Innovasic®
Semiconductor
Extended Life Semiconductor Solutions

# IA186EM/IA188EM
# 8-Bit/16-Bit Microcontrollers
# Data Sheet

Published by Innovasic Semiconductor, Inc.
3737 Princeton Drive NE, Suite 130, Albuquerque, NM 87107

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

## CONVENTIONS

| | |
|---|---|
| **Arial Bold** | Designates headings, figure captions, and table captions. |
| Blue | Designates hyperlinks (PDF copy only). |
| Italics | Designates emphasis or caution related to nearby information. Italics is also used to designate variables, refer to related documents, and to differentiate terms from other common words (e.g., "During refresh cycles, the a and ad busses may not have the same address during the address phase of the ad bus cycle." "The hold latency time [time between the hold and hlda] depends on the current processor activity when the hold is received."). |

# ACRONYMS AND ABBREVIATIONS

| | |
|---|---|
| AMD | Advanced Micro Devices |
| BIC | Bus Interface and Control |
| CDRAM | Count for Dynamic RAM |
| CSC | Chip Selects and Control |
| DA | Disable Address |
| DMA | Direct Memory Access |
| EOI | End of Interrupt |
| ISR | Interrupt Service Routine |
| LMCS | Low-Memory Chip Select |
| MC | Maximum Count |
| MDRAM | Memory Partition for Dynamic RAM |
| MILES™ | Managed IC Lifetime Extension System |
| MMCS | Midrange Memory Chip Select |
| NMI | nonmaskable interrupt |
| PCB | peripheral control block |
| PIO | programmable I/O |
| PLL | phase-lock-loop |
| POR | power-on reset |
| PQFP | Plastic Quad Flat Package |
| PSRAM | Pseudo-Static RAM |
| RCU | Refresh Control Unit |
| RoHS | Restriction of Hazardous Substances |
| SFNM | Special Fully Nested mode |
| TQFP | Thin Quad Flat Package |
| UART | Universal Asynchronous Receiver-Transmitter |
| UMCS | Upper Memory Chip Select |

# 1. Introduction

The IA186EM/IA188EM is a form, fit, and function replacement for the original Advanced Micro Devices Am186EM/Am188EM family of microcontrollers. Innovasic produces replacement ICs using its MILES™, or Managed IC Lifetime Extension System cloning technology. This technology produces replacement ICs far more complex than "emulation" while ensuring they are compatible with the original IC. MILES™ captures the design of a clone so it can be produced even as silicon technology advances. MILES™ also verifies the clone against the original IC so that even the "undocumented features" are duplicated.

## 1.1 General Description

The IA186EM/IA188EM family of microcontrollers replaces obsolete Am186EM/188EM devices, allowing customers to retain existing board designs, software compilers/assemblers and emulation tools, thereby avoiding expensive redesign efforts.

The IA186EM/IA188EM microcontrollers are an upgrade for the 80C186/80C 188 microcontroller designs, with integrated peripherals to provide increased functionality and reduce system costs. The Innovasic devices are created to satisfy requirements of embedded products designed for telecommunications, office automation and storage, and industrial controls.

## 1.2 Features

- Pin-for-pin compatible with Am186EM/Am188EM devices
- All features are retained, including:
  - A phase-lock loop (PLL) allowing same crystal/system clock frequency
  - 8086/8088 instruction set with additional 186 instruction set extensions
  - Programmable interrupt controller
  - Two Direct Memory Access (DMA) channels
  - Three 16-bit timers
  - Programmable chip select logic and wait-state generator
  - Dedicated watchdog timer
  - Two independent asynchronous serial ports (UARTs)
    - DMA capability
    - Hardware flow control
    - 7-, 8-, or 9-bit data capability
- Pulse Width Demodulator feature
- Up to 32 programmable I/O pins (PIO)
- Pseudo-static/dynamic RAM controller
- Fully static CMOS design
- 40-MHz operation at industrial operating conditions
- +5 VDC power supply

# 2. Packaging , Pin Descriptions, and Physical Dimensions

Information on the packages and pin descriptions for the IA186EM and the IA188EM is provided separately. Refer to sections, figures, and tables for information on the device of interest.

## 2.1 Packages and Pinouts

The Innovasic Semiconductor IA186EM and IA188EM microcontroller is available in the following packages:

- 100-Pin Thin Quad Flat Package (TQFP), equivalent to original SQFP package
- 100-Plastic Quad Flat Package (PQFP), equivalent to original PQFP package

### 2.1.1 IA186EM TQFP Package

The pinout for the IA186EM TQFP package is as shown in Figure 1. The corresponding pinout is provided in Tables 1 and 2.

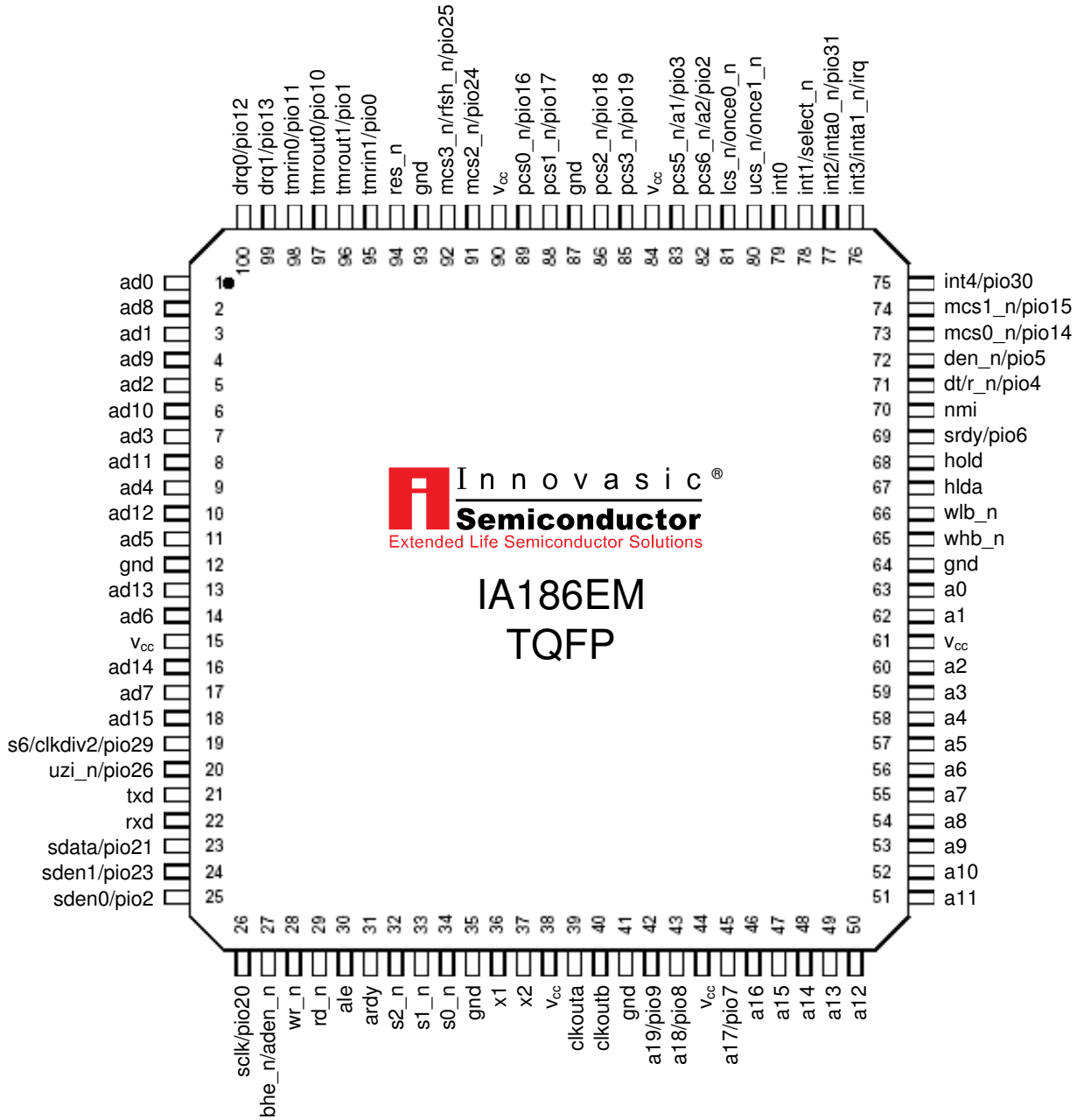



**Figure 1. IA186EM TQFP Package Diagram**

## Table 1. IA186EM TQFP Numeric Pin Listing

| Pin | Name |
|---|---|
| 1 | ad0 |
| 2 | ad8 |
| 3 | ad1 |
| 4 | ad9 |
| 5 | ad2 |
| 6 | ad10 |
| 7 | ad3 |
| 8 | ad11 |
| 9 | ad4 |
| 10 | ad12 |
| 11 | ad5 |
| 12 | gnd |
| 13 | ad13 |
| 14 | ad6 |
| 15 | $v_{cc}$ |
| 16 | ad14 |
| 17 | ad7 |
| 18 | ad15 |
| 19 | s6/clkdiv2/pio29 |
| 20 | uzi_n/pio26 |
| 21 | txd |
| 22 | rxd |
| 23 | sdata/pio21 |
| 24 | sden1/pio23 |
| 25 | sden0/pio22 |
| 26 | sclk/pio20 |
| 27 | bhe_n/aden_n |
| 28 | wr_n |
| 29 | rd_n |
| 30 | ale |
| 31 | ardy |
| 32 | s2_n |
| 33 | s1_n |
| 34 | s0_n |

| Pin | Name |
|---|---|
| 35 | gnd |
| 36 | x1 |
| 37 | x2 |
| 38 | $v_{cc}$ |
| 39 | clkouta |
| 40 | clkoutb |
| 41 | gnd |
| 42 | a19/pio9 |
| 43 | a18/pio8 |
| 44 | $v_{cc}$ |
| 45 | a17/pio7 |
| 46 | a16 |
| 47 | a15 |
| 48 | a14 |
| 49 | a13 |
| 50 | a12 |
| 51 | a11 |
| 52 | a10 |
| 53 | a9 |
| 54 | a8 |
| 55 | a7 |
| 56 | a6 |
| 57 | a5 |
| 58 | a4 |
| 59 | a3 |
| 60 | a2 |
| 61 | $v_{cc}$ |
| 62 | a1 |
| 63 | a0 |
| 64 | gnd |
| 65 | whb_n |
| 66 | wlb_n |
| 67 | hlda |

| Pin | Name |
|---|---|
| 68 | hold |
| 69 | srdy/pio6 |
| 70 | nmi |
| 71 | dt/r_n/pio4 |
| 72 | den_n/pio5 |
| 73 | mcs0_n/pio14 |
| 74 | mcs1_n/pio15 |
| 75 | int4/pio30 |
| 76 | int3/inta1_n/irq |
| 77 | int2/inta0_n/pio31 |
| 78 | int1/select_n |
| 79 | int0 |
| 80 | ucs_n/once1_n |
| 81 | lcs_n/once0_n |
| 82 | pcs6_n/a2/pio2 |
| 83 | pcs5_n/a1/pio3 |
| 84 | $v_{cc}$ |
| 85 | pcs3_n/pio19 |
| 86 | pcs2_n/pio18 |
| 87 | gnd |
| 88 | pcs1_n/pio17 |
| 89 | pcs0_n/pio16 |
| 90 | $v_{cc}$ |
| 91 | mcs2_n/pio24 |
| 92 | mcs3_n/rfsh_n/pio25 |
| 93 | gnd |
| 94 | res_n |
| 95 | tmrin1/pio0 |
| 96 | tmrout1/pio1 |
| 97 | tmrout0/pio10 |
| 98 | tmrin0/pio11 |
| 99 | drq1/pio13 |
| 100 | drq0/pio12 |

## Table 2. IA186EM TQFP Alphabetic Pin Listing

| Name | Pin |
|---|---|
| a0 | 63 |
| a1 | 62 |
| a2 | 60 |
| a3 | 59 |
| a4 | 58 |
| a5 | 57 |
| a6 | 56 |
| a7 | 55 |
| a8 | 54 |
| a9 | 53 |
| a10 | 52 |
| a11 | 51 |
| a12 | 50 |
| a13 | 49 |
| a14 | 48 |
| a15 | 47 |
| a16 | 46 |
| a17/pio7 | 45 |
| a18/pio8 | 43 |
| a19/pio9 | 42 |
| ad0 | 1 |
| ad1 | 3 |
| ad2 | 5 |
| ad3 | 7 |
| ad4 | 9 |
| ad5 | 11 |
| ad6 | 14 |
| ad7 | 17 |
| ad8 | 2 |
| ad9 | 4 |
| ad10 | 6 |
| ad11 | 8 |
| ad12 | 10 |
| ad13 | 13 |

| Name | Pin |
|---|---|
| ad14 | 16 |
| ad15 | 18 |
| ale | 30 |
| ardy | 30 |
| bhe_n/aden_n | 27 |
| clkouta | 39 |
| clkoutb | 40 |
| den_n/pio5 | 72 |
| drq0/pio12 | 100 |
| drq1/pio13 | 99 |
| dt/r_n/pio4 | 71 |
| gnd | 12 |
| gnd | 36 |
| gnd | 41 |
| gnd | 64 |
| gnd | 87 |
| gnd | 93 |
| hlda | 67 |
| hold | 68 |
| int0 | 79 |
| int1/select_n | 78 |
| int2/inta0_n/pio31 | 77 |
| int3/inta1_n/irq | 76 |
| int4/pio30 | 75 |
| lcs_n/once0_n | 81 |
| mcs0_n/pio14 | 73 |
| mcs1_n/pio15 | 74 |
| mcs2_n/pio24 | 91 |
| mcs3_n/rfsh_n/pio25 | 92 |
| nmi | 70 |
| pcs0_n/pio16 | 89 |
| pcs1_npio | 88 |
| pcs2_n/pio18 | 86 |

| Name | Pin |
|---|---|
| pcs3_n/pio19 | 85 |
| pcs5_n/a1/pio3 | 83 |
| pcs6_n/a2/pio2 | 82 |
| rd_n | 29 |
| res_n | 94 |
| rxd/pio23 | 24 |
| s0_n | 34 |
| s1_n | 33 |
| s2_n | 32 |
| s6/clkdiv2/pio29 | 19 |
| sclk/pio20 | 26 |
| sdata/pio21 | 23 |
| sden0/pio22 | 25 |
| sden1/pio23 | 24 |
| srdy/pio6 | 69 |
| tmrin0/pio11 | 98 |
| tmrin1/pio0 | 95 |
| tmrout0/pio10 | 97 |
| tmrout1/pio1 | 96 |
| txd/pio27 | 21 |
| ucs_n/once1_n | 80 |
| uzi_n/pio26 | 20 |
| $V_{CC}$ | 15 |
| $V_{CC}$ | 38 |
| $V_{CC}$ | 44 |
| $V_{CC}$ | 61 |
| $V_{CC}$ | 84 |
| $V_{CC}$ | 90 |
| whb_n | 65 |
| wlb_n | 66 |
| wr_n | 28 |
| x1 | 36 |
| x2 | 37 |

### 2.1.2 IA188EM TQFP Package

The pinout for the IA188EM TQFP package is as shown in Figure 2. The corresponding pinout is provided in Tables 3 and 4.

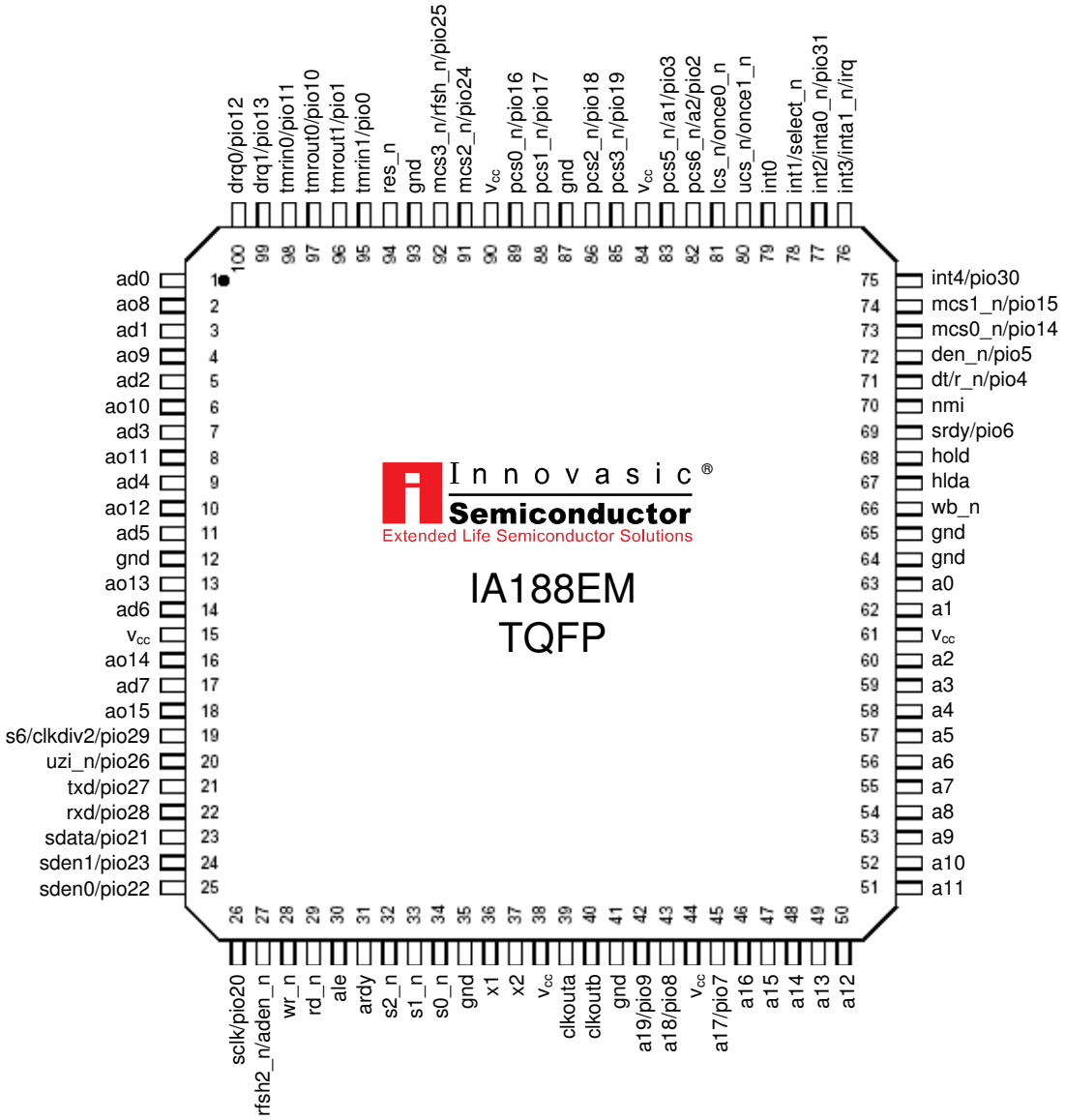



**Figure 2. IA188EM TQFP Package Diagram**

## Table 3. IA188EM TQFP Numeric Pin Listing

| Pin | Name |
|---|---|
| 1 | ad0 |
| 2 | ao8 |
| 3 | ad1 |
| 4 | ao9 |
| 5 | ad2 |
| 6 | ao10 |
| 7 | ad3 |
| 8 | ao11 |
| 9 | ad4 |
| 10 | ao12 |
| 11 | ad5 |
| 12 | gnd |
| 13 | ao13 |
| 14 | ad6 |
| 15 | $v_{cc}$ |
| 16 | ao14 |
| 17 | ad7 |
| 18 | ao15 |
| 19 | s6/clkdiv2/pio29 |
| 20 | uzi_n/pio26 |
| 21 | txd/pio27 |
| 22 | rxd/pio28 |
| 23 | sdata/pio21 |
| 24 | sden1/pio23 |
| 25 | sden0/pio22 |
| 26 | sclk/pio20 |
| 27 | rfsh2_n/aden_n |
| 28 | wr_n |
| 29 | rd_n |
| 30 | ale |
| 31 | ardy |
| 32 | s2_n |
| 33 | s1_n |
| 34 | s0_n |

| Pin | Name |
|---|---|
| 35 | gnd |
| 36 | x1 |
| 37 | x2 |
| 38 | $v_{cc}$ |
| 39 | clkouta |
| 40 | clkoutb |
| 41 | gnd |
| 42 | a19/pio9 |
| 43 | a18/pio8 |
| 44 | $v_{cc}$ |
| 45 | a17/pio7 |
| 46 | a16 |
| 47 | a15 |
| 48 | a14 |
| 49 | a13 |
| 50 | a12 |
| 51 | a11 |
| 52 | a10 |
| 53 | a9 |
| 54 | a8 |
| 55 | a7 |
| 56 | a6 |
| 57 | a5 |
| 58 | a4 |
| 59 | a3 |
| 60 | a2 |
| 61 | $v_{cc}$ |
| 62 | a1 |
| 63 | a0 |
| 64 | gnd |
| 65 | gnd |
| 66 | wb_n |
| 67 | hlda |

| Pin | Name |
|---|---|
| 68 | hold |
| 69 | srdy/pio6 |
| 70 | nmi |
| 71 | dt/r_n/pio4 |
| 72 | den_n/pio5 |
| 73 | mcs0_n/pio14 |
| 74 | mcs1_n/pio15 |
| 75 | int4/pio30 |
| 76 | int3/inta1_n/irq |
| 77 | int2/inta0_n/pio31 |
| 78 | int1/select_n |
| 79 | int0 |
| 80 | ucs_n/once1_n |
| 81 | lcs_n/once0_n |
| 82 | pcs6_n/a2/pio2 |
| 83 | pcs5_n/a1/pio3 |
| 84 | $v_{cc}$ |
| 85 | pcs3_n/pio19 |
| 86 | pcs2_n/pio18 |
| 87 | gnd |
| 88 | pcs1_n/pio17 |
| 89 | pcs0_n/pio16 |
| 90 | $v_{cc}$ |
| 91 | mcs2_n/pio24 |
| 92 | mcs3_n/rfsh_n/pio25 |
| 93 | gnd |
| 94 | res_n |
| 95 | tmrin1/pio0 |
| 96 | tmrout1/pio1 |
| 97 | tmrout0/pio10 |
| 98 | tmrin0/pio11 |
| 99 | drq1/pio13 |
| 100 | drq0/pio12 |

## Table 4. IA188EM TQFP Alphabetic Pin Listing

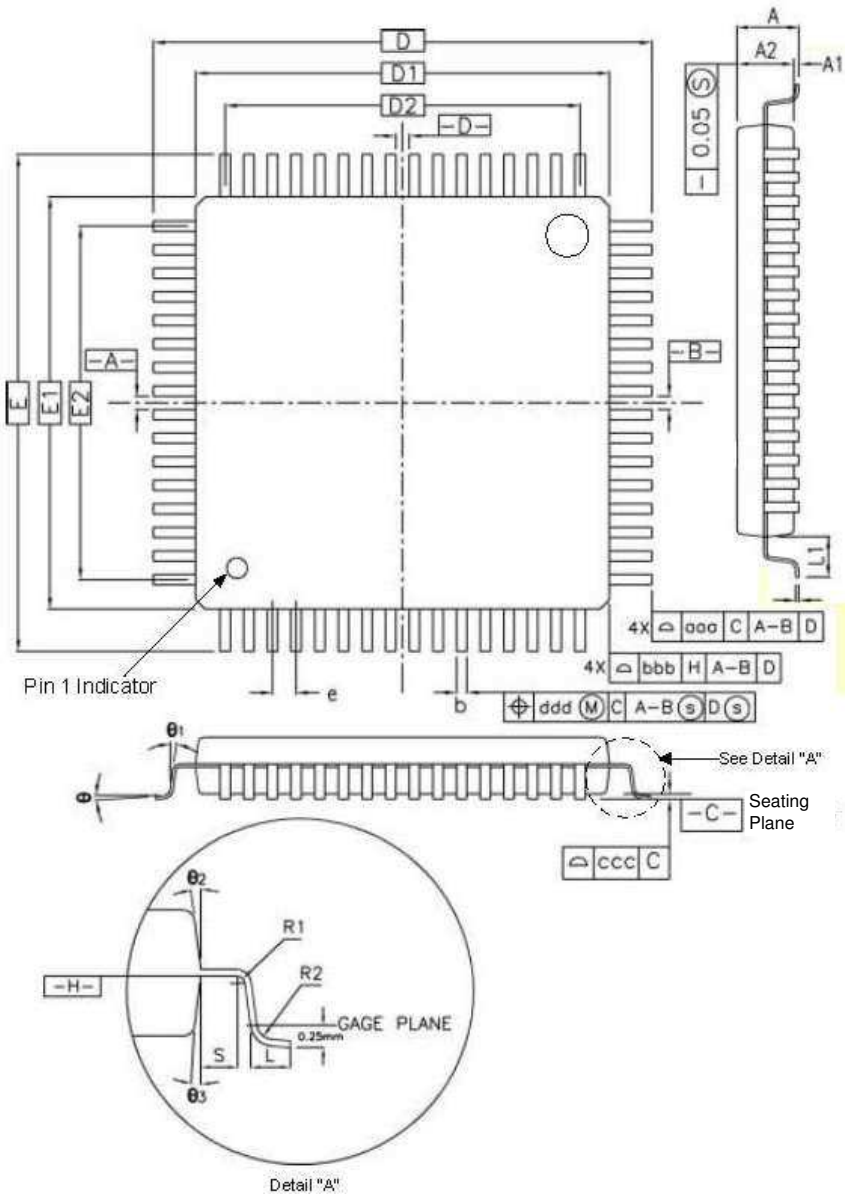
| Name | Pin |
|---|---|
| a0 | 63 |
| a1 | 62 |
| a2 | 60 |
| a3 | 59 |
| a4 | 58 |
| a5 | 57 |
| a6 | 56 |
| a7 | 55 |
| a8 | 54 |
| a9 | 53 |
| a10 | 52 |
| a11 | 51 |
| a12 | 50 |
| a13 | 49 |
| a14 | 48 |
| a15 | 47 |
| a16 | 46 |
| a17/pio7 | 45 |
| a18/pio8 | 43 |
| a19/pio9 | 42 |
| ale | 30 |
| ad0 | 1 |
| ad1 | 3 |
| ad2 | 5 |
| ad3 | 7 |
| ad4 | 9 |
| ad5 | 11 |
| ad6 | 14 |
| ad7 | 17 |
| ao8 | 2 |
| ao9 | 4 |
| ao10 | 6 |
| ao11 | 8 |
| ao12 | 10 |

| Name | Pin |
|---|---|
| ao13 | 13 |
| ao14 | 16 |
| ao15 | 18 |
| ardy | 30 |
| clkouta | 39 |
| clkoutb | 40 |
| den_n/pio5 | 72 |
| drq0/pio12 | 100 |
| drq1/pio13 | 99 |
| dt/r_n/pio4 | 71 |
| gnd | 12 |
| gnd | 35 |
| gnd | 41 |
| gnd | 64 |
| gnd | 65 |
| gnd | 87 |
| gnd | 93 |
| hlda | 67 |
| hold | 68 |
| int0 | 79 |
| int1/select_n | 78 |
| int2/inta0_n/pio31 | 77 |
| int3/inta1_n/irq | 76 |
| int4/pio30 | 75 |
| lcs_n/once0_n | 81 |
| mcs0_n/pio14 | 73 |
| mcs1_n/pio15 | 74 |
| mcs2_n/pio24 | 91 |
| mcs3_n/rfsh_n/pio25 | 92 |
| nmi | 70 |
| pcs0_n/pio16 | 89 |
| pcs1_n/pio17 | 88 |
| pcs2_n/pio18 | 86 |

| Name | Pin |
|---|---|
| pcs3_n/pio19 | 85 |
| pcs5_n/a1/pio3 | 83 |
| pcs6_n/a2/pio2 | 82 |
| rd_n | 29 |
| res_n | 94 |
| rfsh2_n/aden_n | 27 |
| rxd/pio28 | 22 |
| s0_n | 34 |
| s1_n | 33 |
| s2_n | 32 |
| s6/lock_n/clkdiv2/pio29 | 19 |
| sclk/pio20 | 26 |
| sdata/pio21 | 23 |
| sden0/pio22 | 25 |
| sden1/pio23 | 24 |
| srdy/pio6 | 69 |
| tmrin0/pio11 | 98 |
| tmrin1/pio0 | 95 |
| tmrout0/pio10 | 97 |
| tmrout1/pio1 | 96 |
| txd/pio27 | 21 |
| ucs_n/once1_n | 80 |
| uzi_n/pio26 | 20 |
| $V_{CC}$ | 15 |
| $V_{CC}$ | 38 |
| $V_{CC}$ | 44 |
| $V_{CC}$ | 61 |
| $V_{CC}$ | 84 |
| $V_{CC}$ | 90 |
| wb_n | 66 |
| wr_n | 28 |
| x1 | 36 |
| x2 | 37 |

### 2.1.3 TQFP Physical Dimensions

The physical dimensions for the TQFP are as shown in Figure 3.




Legend:

| Symbol | Millimeter Min | Nom | Max | Inch Min | Nom | Max |
|---|---|---|---|---|---|---|
| A | – | – | 1.20 | – | – | 0.047 |
| A1 | 0.05 | – | 0.15 | 0.002 | – | 0.006 |
| A2 | 0.95 | 1.00 | 1.05 | 0.037 | 0.039 | 0.041 |
| b | 0.17 | 0.20 | 0.27 | 0.007 | 0.008 | 0.011 |
| c | 0.09 | – | 0.20 | 0.004 | – | 0.008 |
| D | 16.00 BSC. | | | 0.630 BSC. | | |
| D1 | 14.00 BSC. | | | 0.551 BSC. | | |
| D2 | 12.00 | | | 0.472 | | |
| e | 0.50 BSC. | | | 0.02 BSC. | | |
| E | 16.00 BSC. | | | 0.630 BSC. | | |
| E1 | 14.00 BSC. | | | 0.551 BSC. | | |
| E2 | 12.00 | | | 0.472 | | |
| L | 0.45 | 0.60 | 0.75 | 0.018 | 0.024 | 0.030 |
| L1 | 1.00 REF | | | 0.039 REF | | |
| R1 | 0.08 | – | – | 0.003 | – | – |
| R2 | 0.08 | – | 0.20 | 0.003 | – | 0.008 |
| S | 0.20 | – | – | 0.008 | – | – |
| θ | 0° | 3.5° | 7° | 0° | 3.5° | 7° |
| $\theta_1$ | 0° | – | – | 0° | – | – |
| $\theta_2$ | 11° | 12° | 13° | 11° | 12° | 13° |
| $\theta_3$ | 11° | 12° | 13° | 11° | 12° | 13° |
| Tolerances of Form and Position | | | | | | |
| aaa | 0.20 | | | 0.008 | | |
| bbb | 0.20 | | | 0.008 | | |
| ccc | 0.08 | | | 0.003 | | |
| ddd | 0.08 | | | 0.003 | | |

Note:
Control dimensions are in millimeters.

**Figure 3. TQFP Package Dimensions**

### 2.1.4 IA186EM PQFP Package

The pinout for the IA186EM PQFP package is as shown in Figure 4. The corresponding pinout is provided in Tables 5 and 6.

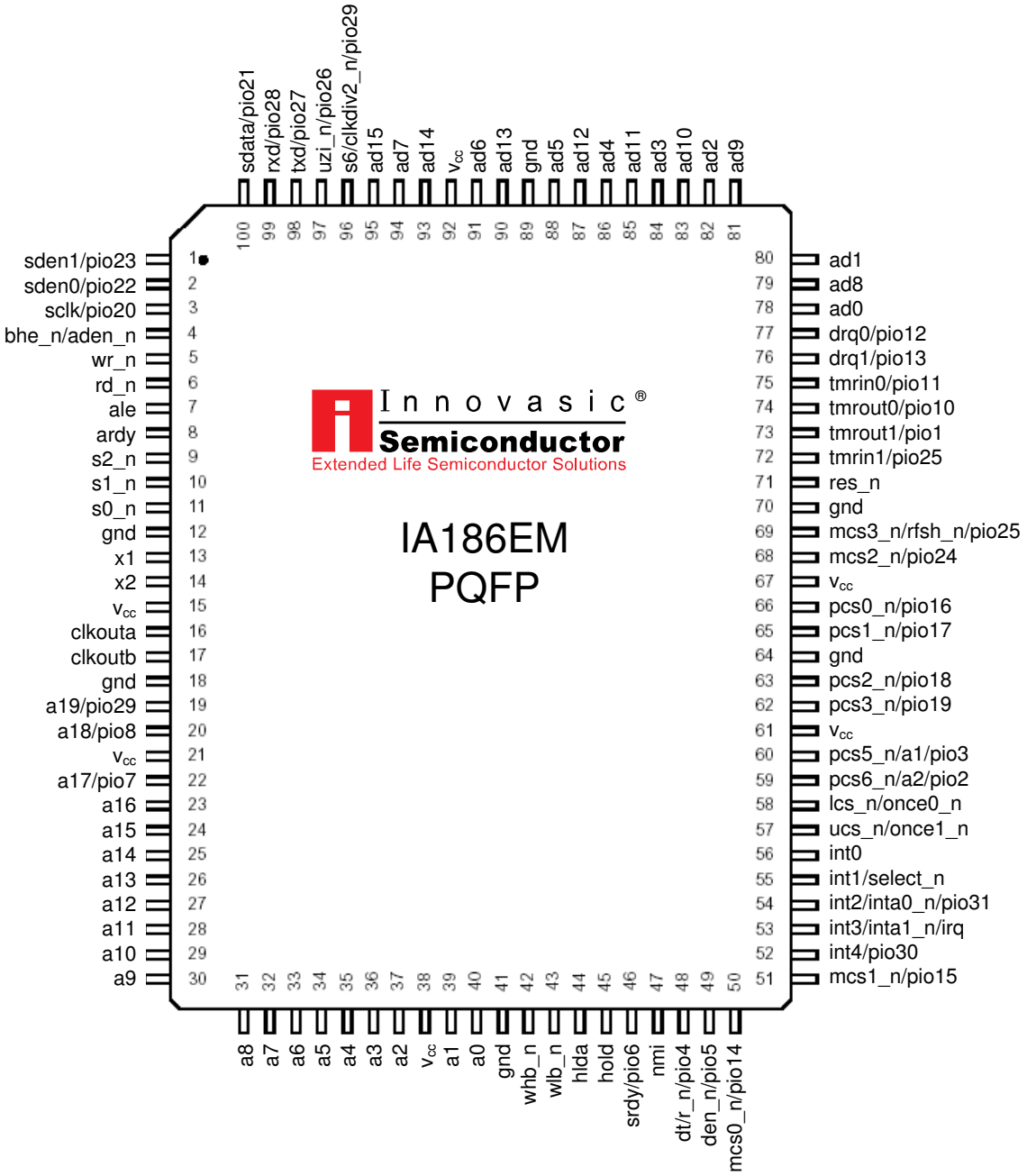



**Figure 4. IA186EM PQFP Package Diagram**

## Table 5. IA186EM PQFP Numeric Pin Listing

| Pin | Name |
|---|---|
| 1 | sden1/pio23 |
| 2 | sden0/pio22 |
| 3 | sclk/pio20 |
| 4 | bhe_n/aden_n |
| 5 | wr_n |
| 6 | rd_n |
| 7 | ale |
| 8 | ardy |
| 9 | s2_n |
| 10 | s1_n |
| 11 | s0_n |
| 12 | gnd |
| 13 | x1 |
| 14 | x2 |
| 15 | $v_{cc}$ |
| 16 | clkouta |
| 17 | clkoutb |
| 18 | gnd |
| 19 | a19/pio29 |
| 20 | a18/pio8 |
| 21 | $v_{cc}$ |
| 22 | a17/pio7 |
| 23 | a16 |
| 24 | a15 |
| 25 | a14 |
| 26 | a13 |
| 27 | a12 |
| 28 | a11 |
| 29 | a10 |
| 30 | a9 |
| 31 | a8 |
| 32 | a7 |
| 33 | a6 |
| 34 | a5 |

| Pin | Name |
|---|---|
| 35 | a4 |
| 36 | a3 |
| 37 | a2 |
| 38 | $v_{cc}$ |
| 39 | a1 |
| 40 | a0 |
| 41 | gnd |
| 42 | whb_n |
| 43 | wlb_n |
| 44 | hlda |
| 45 | hold |
| 46 | srdy/pio6 |
| 47 | nmi |
| 48 | dt/r_n/pio4 |
| 49 | den_n/pio5 |
| 50 | mcs0_n/pio14 |
| 51 | mcs1_n/pio15 |
| 52 | int4/pio30 |
| 53 | int3/inta1_n/irq |
| 54 | int2/inta0_n/pio31 |
| 55 | int1/select_n |
| 56 | int0 |
| 57 | ucs_n/once1_n |
| 58 | lcs_n/once0_n |
| 59 | pcs6_n/a2/pio2 |
| 60 | pcs5_n/a1/pio3 |
| 61 | $v_{cc}$ |
| 62 | pcs3_n/pio19 |
| 63 | pcs2_n/pio18 |
| 64 | gnd |
| 65 | pcs1_n/pio17 |
| 66 | pcs0_n/pio16 |
| 67 | $v_{cc}$ |

| Pin | Name |
|---|---|
| 68 | mcs2_n/pio24 |
| 69 | mcs3_n/rfsh_n/pio25 |
| 70 | gnd |
| 71 | res_n |
| 72 | tmrin1/pio25 |
| 73 | tmrout1/pio1 |
| 74 | tmrout0/pio10 |
| 75 | tmrin0/pio11 |
| 76 | drq1/pio13 |
| 77 | drq0/pio12 |
| 78 | ad0 |
| 79 | ad8 |
| 80 | ad1 |
| 81 | ad9 |
| 82 | ad2 |
| 83 | ad10 |
| 84 | ad3 |
| 85 | ad11 |
| 86 | ad4 |
| 87 | ad12 |
| 88 | ad5 |
| 89 | gnd |
| 90 | ad13 |
| 91 | ad6 |
| 92 | $v_{cc}$ |
| 93 | ad14 |
| 94 | ad7 |
| 95 | ad15 |
| 96 | s6/clkdiv2_n/pio29 |
| 97 | uzi_n/pio26 |
| 98 | txd/pio27 |
| 99 | rxd/pio28 |
| 100 | sdata/pio21 |

## Table 6. IA186EM PQFP Alphabetic Pin Listing

| Name | Pin |
|---|---|
| a0 | 40 |
| a1 | 39 |
| a2 | 37 |
| a3 | 36 |
| a4 | 35 |
| a5 | 34 |
| a6 | 33 |
| a7 | 32 |
| a8 | 31 |
| a9 | 30 |
| a10 | 29 |
| a11 | 28 |
| a12 | 27 |
| a13 | 26 |
| a14 | 25 |
| a15 | 24 |
| a16 | 23 |
| a17/pio7 | 22 |
| a18/pio8 | 20 |
| a19/pio9 | 19 |
| ad0 | 78 |
| ad1 | 80 |
| ad2 | 82 |
| ad3 | 84 |
| ad4 | 86 |
| ad5 | 88 |
| ad6 | 91 |
| ad7 | 94 |
| ad8 | 79 |
| ad9 | 81 |
| ad10 | 83 |
| ad11 | 85 |
| ad12 | 87 |
| ad13 | 90 |

| Name | Pin |
|---|---|
| ad14 | 93 |
| ad15 | 95 |
| ale | 7 |
| ardy | 8 |
| bhe_n/aden_n | 4 |
| clkouta | 16 |
| clkoutb | 17 |
| den_n/pio5 | 49 |
| drq0/pio12 | 77 |
| drq1/pio13 | 76 |
| dt/r_n/pio4 | 48 |
| gnd | 12 |
| gnd | 18 |
| gnd | 41 |
| gnd | 64 |
| gnd | 70 |
| gnd | 89 |
| hlda | 44 |
| hold | 45 |
| int0 | 56 |
| int1/select_n | 55 |
| int2/inta0_n/pio31 | 54 |
| int3/inta1_n/irq | 53 |
| int4/pio30 | 52 |
| lcs_n/once0_n | 58 |
| mcs0_n/pio14 | 50 |
| mcs1_n/pio15 | 51 |
| mcs2_n/pio24 | 68 |
| mcs3_n/rfsh_n/pio25 | 69 |
| nmi | 47 |
| pcs0_n/pio16 | 66 |
| pcs1_n/pio17 | 65 |
| pcs2_n/pio18 | 63 |

| Name | Pin |
|---|---|
| pcs3_n/pio19 | 62 |
| pcs5_n/a1/pio3 | 60 |
| pcs6_n/a2/pio2 | 59 |
| rd_n | 6 |
| res_n | 71 |
| rxd/pio28 | 99 |
| s0_n | 11 |
| s1_n | 10 |
| s2_n | 9 |
| s6/clkdiv2/pio29 | 96 |
| sclk/pio20 | 3 |
| sdata/pio21 | 100 |
| sden0/pio22 | 2 |
| sden1/pio23 | 1 |
| srdy/pio6 | 46 |
| tmrin0/pio11 | 75 |
| tmrin1/pio0 | 72 |
| tmrout0/pio10 | 74 |
| tmrout1/pio1 | 73 |
| txd/pio27 | 98 |
| ucs_n/once1_n | 57 |
| uzi_n/pio26 | 97 |
| $V_{CC}$ | 15 |
| $V_{CC}$ | 21 |
| $V_{CC}$ | 38 |
| $V_{CC}$ | 61 |
| $V_{CC}$ | 67 |
| $V_{CC}$ | 92 |
| whb_n | 42 |
| wlb_n | 43 |
| wr_n | 5 |
| x1 | 13 |
| x2 | 14 |

### 2.1.5 IA188EM PQFP Package

The pinout for the IA188EM PQFP package is as shown in Figure 5. The corresponding pinout is provided in Tables 7 and 8.

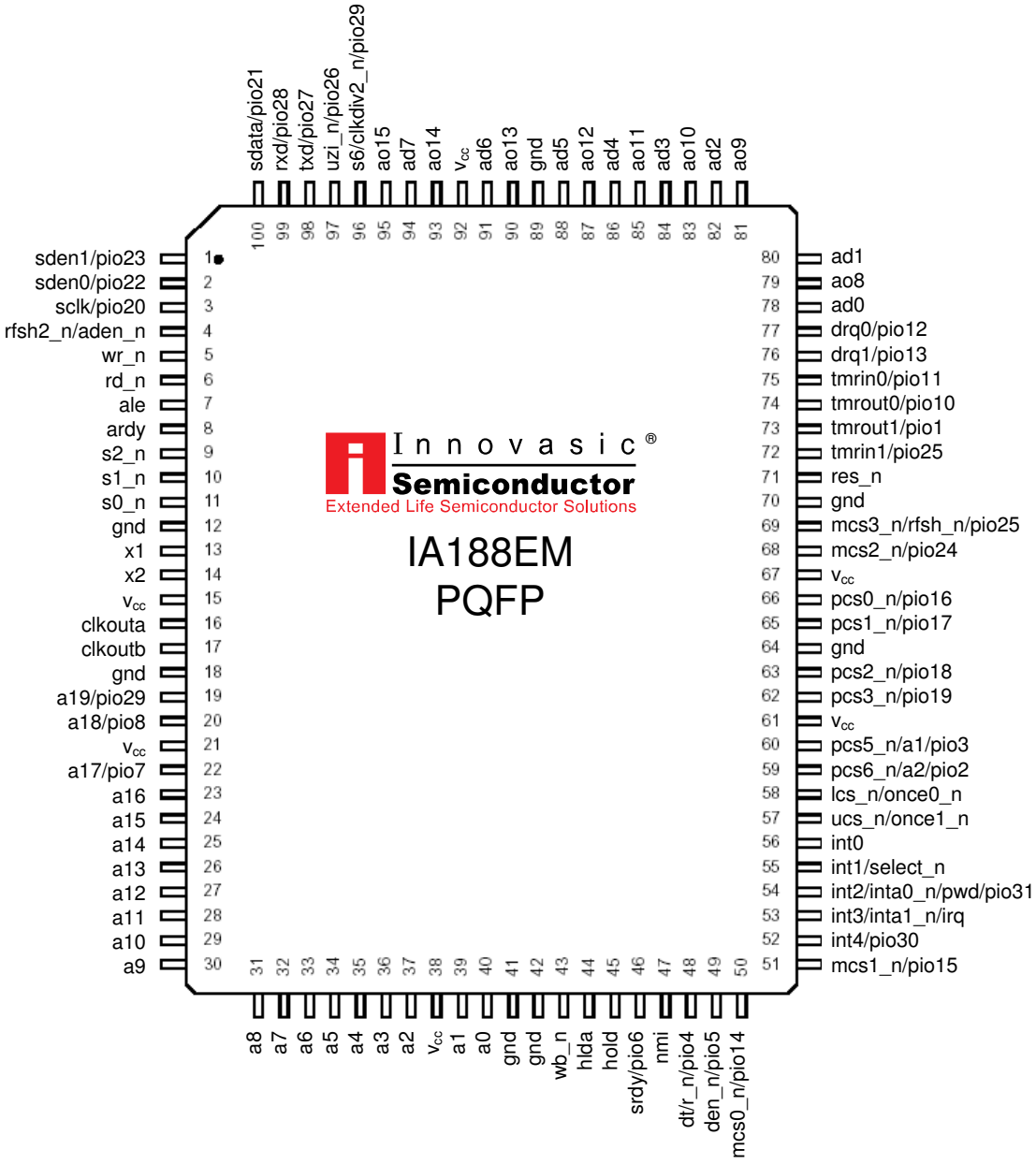



**Figure 5. IA188EM PQFP Package Diagram**

## Table 7. IA188EM PQFP Numeric Pin Listing

| Pin | Name |
|---|---|
| 1 | sden1/pio23 |
| 2 | sden0/pio22 |
| 3 | sclk/pio20 |
| 4 | rfsh2_n/aden_n |
| 5 | wr_n |
| 6 | rd_n |
| 7 | ale |
| 8 | ardy |
| 9 | s2_n |
| 10 | s1_n |
| 11 | s0_n |
| 12 | gnd |
| 13 | x1 |
| 14 | x2 |
| 15 | $v_{cc}$ |
| 16 | clkouta |
| 17 | clkoutb |
| 18 | gnd |
| 19 | a19/pio29 |
| 20 | a18/pio8 |
| 21 | $v_{cc}$ |
| 22 | a17/pio7 |
| 23 | a16 |
| 24 | a15 |
| 25 | a14 |
| 26 | a13 |
| 27 | a12 |
| 28 | a11 |
| 29 | a10 |
| 30 | a9 |
| 31 | a8 |
| 32 | a7 |
| 33 | a6 |
| 34 | a5 |

| Pin | Name |
|---|---|
| 35 | a4 |
| 36 | a3 |
| 37 | a2 |
| 38 | $v_{cc}$ |
| 39 | a1 |
| 40 | a0 |
| 41 | gnd |
| 42 | gnd |
| 43 | wb_n |
| 44 | hlda |
| 45 | hold |
| 46 | srdy/pio6 |
| 47 | nmi |
| 48 | dt/r_n/pio4 |
| 49 | den_n/pio5 |
| 50 | mcs0_n/pio14 |
| 51 | mcs1_n/pio15 |
| 52 | int4/pio30 |
| 53 | int3/inta1_n/irq |
| 54 | int2/inta0_n/pwd/pio31 |
| 55 | int1/select_n |
| 56 | int0 |
| 57 | ucs_n/once1_n |
| 58 | lcs_n/once0_n |
| 59 | pcs6_n/a2/pio2 |
| 60 | pcs5_n/a1/pio3 |
| 61 | $v_{cc}$ |
| 62 | pcs3_n/pio19 |
| 63 | pcs2_n/pio18 |
| 64 | gnd |
| 65 | pcs1_n/pio17 |
| 66 | pcs0_n/pio16 |
| 67 | $v_{cc}$ |

| Pin | Name |
|---|---|
| 68 | mcs2_n/pio24 |
| 69 | mcs3_n/rfsh_n/pio25 |
| 70 | gnd |
| 71 | res_n |
| 72 | tmrin1/pio25 |
| 73 | tmrout1/pio1 |
| 74 | tmrout0/pio10 |
| 75 | tmrin0/pio11 |
| 76 | drq1/pio13 |
| 77 | drq0/pio12 |
| 78 | ad0 |
| 79 | ao8 |
| 80 | ad1 |
| 81 | ao9 |
| 82 | ad2 |
| 83 | ao10 |
| 84 | ad3 |
| 85 | ao11 |
| 86 | ad4 |
| 87 | ao12 |
| 88 | ad5 |
| 89 | gnd |
| 90 | ao13 |
| 91 | ad6 |
| 92 | $v_{cc}$ |
| 93 | ao14 |
| 94 | ad7 |
| 95 | ao15 |
| 96 | s6/clkdiv2_n/pio29 |
| 97 | uzi_n/pio26 |
| 98 | txd/pio27 |
| 99 | rxd/pio28 |
| 100 | sdata/pio21 |

## Table 8. IA188EM PQFP Alphabetic Pin Listing

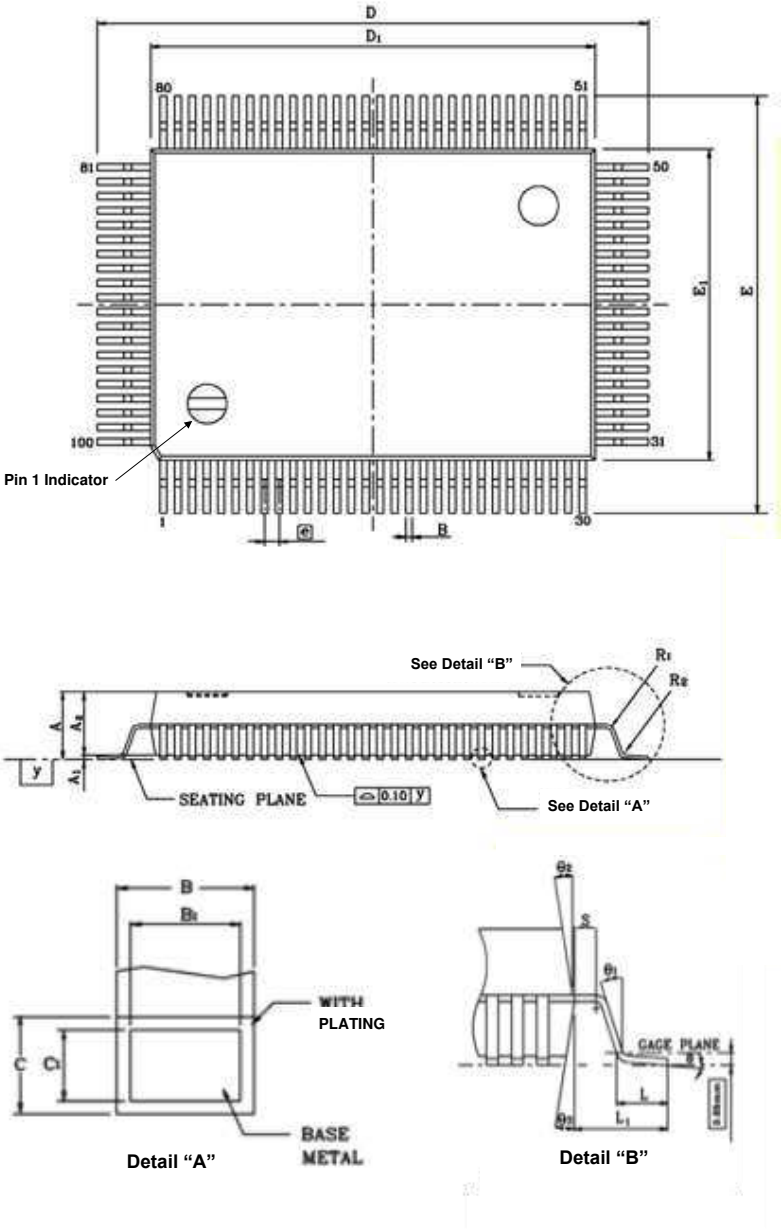
| Name | Pin |
|---|---|
| a0 | 40 |
| a1 | 39 |
| a2 | 37 |
| a3 | 36 |
| a4 | 35 |
| a5 | 34 |
| a6 | 33 |
| a7 | 32 |
| a8 | 31 |
| a9 | 30 |
| a10 | 29 |
| a11 | 28 |
| a12 | 27 |
| a13 | 26 |
| a14 | 25 |
| a15 | 24 |
| a16 | 23 |
| a17/pio7 | 22 |
| a18/pio8 | 20 |
| a19/pio9 | 19 |
| ad0 | 78 |
| ad1 | 80 |
| ad2 | 82 |
| ad3 | 84 |
| ad4 | 86 |
| ad5 | 88 |
| ad6 | 91 |
| ad7 | 94 |
| ale | 7 |
| ao8 | 79 |
| ao9 | 81 |
| ao10 | 83 |
| ao11 | 85 |
| ao12 | 87 |

| Name | Pin |
|---|---|
| ao13 | 90 |
| ao14 | 93 |
| ao15 | 95 |
| ardy | 8 |
| clkouta | 16 |
| clkoutb | 17 |
| den_n/ds_n/pio5 | 49 |
| drq0/pio12 | 77 |
| drq1/pio13 | 76 |
| dt/r_n/pio4 | 48 |
| gnd | 12 |
| gnd | 18 |
| gnd | 41 |
| gnd | 42 |
| gnd | 64 |
| gnd | 70 |
| gnd | 89 |
| hlda | 44 |
| hold | 45 |
| int0 | 56 |
| int1/select_n | 55 |
| int2/inta0_n/pwd/pio31 | 54 |
| int3/inta1_n/irq | 53 |
| int4/pio30 | 52 |
| lcs_n/once0_n | 58 |
| mcs0_n/pio14 | 50 |
| mcs1_n/pio15 | 51 |
| mcs2_n/pio24 | 68 |
| mcs3_n/rfsh_n/pio25 | 69 |
| nmi | 47 |
| pcs0_n/pio16 | 66 |
| pcs1_n/pio17 | 65 |
| pcs2_n/cts1_n/enrx1_n/pio18 | 63 |

| Name | Pin |
|---|---|
| pcs3_n/rts1_n/rtr1_n/pio19 | 62 |
| pcs5_n/a1/pio3 | 60 |
| pcs6_n/a2/pio2 | 59 |
| rd_n | 6 |
| res_n | 71 |
| rfsh2_n/aden_n | 4 |
| rxd/pio28 | 99 |
| s0_n | 11 |
| s1_n | 10 |
| s2_n | 9 |
| s6/lock_n/clkdiv2/pio29 | 96 |
| sclk/pio20 | 3 |
| sdata/pio21 | 100 |
| sden0/pio22 | 2 |
| sden1/pio23 | 1 |
| srdy/pio6 | 46 |
| tmrin0/pio11 | 75 |
| tmrin1/pio0 | 72 |
| tmrout0/pio10 | 74 |
| tmrout1/pio1 | 73 |
| txd/pio27 | 98 |
| ucs_n/once1_n | 57 |
| uzi_n/pio26 | 97 |
| $V_{CC}$ | 15 |
| $V_{CC}$ | 21 |
| $V_{CC}$ | 38 |
| $V_{CC}$ | 61 |
| $V_{CC}$ | 67 |
| $V_{CC}$ | 92 |
| wb_n | 42 |
| wr_n | 5 |
| x1 | 13 |
| x2 | 14 |

### 2.1.6 PQFP Physical Dimensions

The physical dimensions for the PQFP are as shown in Figure 6.




| Legend | | | | | | |
|---|---|---|---|---|---|---|
| | Millimeter | | | Inch | | |
| Symbol | Min | Nom | Max | Min | Nom | Max |
| A | – | – | 3.40 | – | – | 0.134 |
| $A_1$ | 0.25 | – | – | 0.010 | – | – |
| $A_2$ | 2.73 | 2.85 | 2.97 | 0.107 | 0.112 | 0.117 |
| B | 0.25 | 0.30 | 0.38 | 0.010 | 0.012 | 0.015 |
| $B_1$ | 0.22 | 0.30 | 0.33 | 0.009 | 0.012 | 0.013 |
| C | 0.13 | 0.15 | 0.23 | 0.005 | 0.006 | 0.009 |
| $C_1$ | 0.11 | 0.15 | 0.17 | 0.004 | 0.006 | 0.007 |
| D | 23.00 | 23.20 | 23.40 | 0.906 | 0.913 | 0.921 |
| $D_1$ | 19.90 | 20.00 | 20.10 | 0.783 | 0.787 | 0.791 |
| E | 17.00 | 17.20 | 17.40 | 0.669 | 0.677 | 0.685 |
| $E_1$ | 13.90 | 14.00 | 14.10 | 0.547 | 0.551 | 0.555 |
| e | 0.65 BSC. | | | 0.026 BSC. | | |
| L | 0.73 | 0.88 | 1.03 | 0.029 | 0.035 | 0.041 |
| $L_1$ | 1.60 BSC. | | | 0.063 BSC. | | |
| $R_1$ | 0.13 | – | – | 0.005 | – | – |
| $R_2$ | 0.13 | – | 0.30 | 0.005 | – | 0.012 |
| S | 0.20 | – | – | 0.008 | – | – |
| Y | – | – | 0.10 | – | – | 0.004 |
| θ | 0° | – | 7° | 0° | – | 7° |
| $\theta_1$ | 0° | – | – | 0° | – | – |
| $\theta_2$ | 9° | 10° | 11° | 9° | 10° | 11° |
| $\theta_3$ | 9° | 10° | 11° | 9° | 10° | 11° |

Notes:
1. Dimensions $D_1$ and $E_1$ do not include mold protrusion, but mold mismatch is included. Allowable protrusion is 0.25mm/0.010″ per side.
2. Dimension B does not include Dambar protrusion. Allowable protrusion is 0.08mm/0.003″ total in excess of the B dimension at maximum material condition. Dambar cannot be located on the lower radius or the foot.
3. Controlling dimension: millimeter.

**Figure 6. PQFP Package Dimensions**

## 2.2 Pin Descriptions

### 2.2.1 a19/pio9, a18/pio8, a17/pio7, a16–a0—Address Bus (synchronous outputs with tristate)

These pins are the system's source of non-multiplexed I/O or memory addresses and occur a half clkouta cycle before the multiplexed address/data bus (ad15–ad0 for the IA186EM or ao15–ao8 and ad7–ad0 for the IA188EM). The address bus is tristated during a bus hold or reset.

### 2.2.2 ad15–ad8 (IA186EM)—Address/data bus (level-sensitive synchronous inouts with tristate)

These pins are the system's source of time-multiplexed I/O or memory addresses and data. The address function of these pins can be disabled (see bhe_n/aden_n pin description). If the address function of these pins is enabled, the address will be present on this bus during $t_1$ of the bus cycle and data will be present during $t_2$, $t_3$, and $t_4$ of the same bus cycle.

If whb_n is not active, these pins are tristated during $t_2$, $t_3$, and $t_4$ of the bus cycle.

The address/data bus is tristated during a bus hold or reset.

These pins can be used to load the internal Reset Configuration register (RESCON, offset 0F6h) with configuration data during a power-on reset (POR).

### 2.2.3 ad7–ad0—Address/Data bus (level-sensitive synchronous inouts with tristate)

These pins are the system's source of time-multiplexed low-order byte of the addresses for I/O or memory and 8-bit data. The low-order address byte will be present on this bus during $t_1$ of the bus cycle and the 8-bit data will be present during $t_2$, $t_3$, and $t_4$ of the same bus cycle.

The address function of these pins can be disabled (see bhe_n/aden_n pin description).

If wlb_n (IA186EM) is not active, these pins are tristated during $t_2$, $t_3$, and $t_4$ of the bus cycle. The address/data bus is tristated during a bus hold or reset.

### 2.2.4 ao15–ao8 (IA188EM)—Address-only bus (level-sensitive synchronous outputs with tristate)

The address-only bus will contain valid high-order address bits during the bus cycle ($t_1$, $t_2$, $t_3$, and $t_4$) if the bus is enabled.

These pins are combined with ad7–ad0 to complete the multiplexed address bus and are tristated during a bus hold or reset condition.

### 2.2.5 ale—Address Latch Enable (synchronous output)

This signal indicates the presence of an address on the address bus (ad15–ad0 for the IA186EM or ao15–ao8 and ad7–ad0 for the IA188EM), which is guaranteed to be valid on the falling edge of ale.

### 2.2.6 ardy—Asynchronous Ready (level-sensitive asynchronous input)

This asynchronous signal provides an indication to the microcontroller that the addressed I/O device or memory space will complete a data transfer. This active high signal is asynchronous with respect to clkouta and if the falling edge of ardy is not synchronized to clkouta, an additional clock cycle may be added

Signal ardy should be tied high to maintain a permanent assertion of the ready condition. On the other hand, if the ardy signal is not used by the system it should be tied low, which passes control to the srdy signal.

### 2.2.7 bhe_n/aden_n (IA186EM)—Bus High Enable (synchronous output with tristate)/Address Enable (input with internal pull-up)

The bhe_n and address bit ad0 or a0 inform the system which bytes of the data bus (upper, lower, or both) are involved in the current memory access bus cycle as shown Table 9.

**Table 9. Bus Cycle Types for bhe_n and ad0**

| bhe_n | ad0 | Type of Bus Cycle |
|---|---|---|
| 0 | 0 | Word Transfer |
| 0 | 1 | High-Byte Transfer (Bits [15–8]) |
| 1 | 0 | Low-Byte Transfer (Bits [7–0]) |
| 1 | 1 | Refresh |

The bhe_n does not require latching and during bus hold and reset is tristated. It is asserted during $t_1$ and remains so through $t_3$ and $t_w$.

The high- and low-byte write enable functions of bhe_n and ad0 are performed by whb_n and wlb_n, respectively.

When using the ad bus, DRAM refresh cycles are indicated by bhe_n/aden_n and ad0 both being high. During refresh cycles the a and ad busses may not have the same address during the address phase of the ad bus cycle necessitating the use of ad0 as a determinant for the refresh cycle rather than a0.

An additional signal is used for Pseudo-Static RAM (PSRAM) refreshes (see mcs3_n/rfsh_n pin description).

There is a weak internal pull-up on bhe_n/aden_n obviating the need for an external pull-up and reducing power consumption.

Holding aden_n high or letting it float during POR passes control of the address function of the ad bus (ad15–ad0) during LCS and UCS bus cycles from aden_n to the Disable Address (DA) bit in Low-Memory Chip Select (LMCS) and Upper Memory Chip Select (UMCS) registers. When the address function is selected, the memory address is placed on the a19–a0 pins.

Holding aden_n low during POR, both the address and data are driven onto the ad bus independently of the DA bit setting. This pin is normally sampled one clock cycle after the rising edge of res_n.

### 2.2.8 clkouta—Clock Output A (synchronous output)

This pin is the internal clock output to the system. Bits [9–8] and Bits [2–0] of the Power-Save Control register (PDCON) control the output of this pin, which may be tristated, output the crystal input frequency (x1), or output the power save frequency (internal processor frequency after divisor). The clkouta can be used as a full-speed clock source in power-save mode. The AC timing specifications that are clock-related refer to clkouta, which remains active during reset and hold conditions.

### 2.2.9 clkoutb—Clock Output B (synchronous output)

This pin is an additional clock output to the system. Bits [11–10] and [2–0] of the Power-Save Control register (PDCON) control the output of this pin, which may be tristated, output the PLL frequency, or may output the power-save frequency (internal processor frequency after divisor). The clkoutb remains active during reset and hold conditions.

### 2.2.10 den_n/pio5—Data Enable Strobe (synchronous output with tristate)

This pin provides an output enable to an external bus data bus transmitter or receiver. This signal is asserted during I/O, memory, and interrupt acknowledge processes and is deasserted when dt/r_n undergoes a change of state. It is tristated for a bus hold or reset.

### 2.2.11 drq1/pio12–drq0/pio13—DMA Requests (synchronous level-sensitive inputs)

An external device that is ready for DMA channel 1 or 0 to carry out a transfer indicates to the microcontroller this readiness on these pins. They are level triggered, internally synchronized, not latched, and must remain asserted until dealt with.

### 2.2.12 dt/r_n/pio4—Data Transmit or Receive (synchronous output with tristate)

The microcontroller transmits data when dt/r_n is pulled high and receives data when this pin is pulled low. It floats during a reset or bus hold condition.

### 2.2.13 gnd—Ground

Six or seven pins, depending on package, connect the microcontroller to the system ground.

### 2.2.14 hlda—Bus Hold Acknowledge (synchronous output)

This pin is pulled high to signal the system that the microcontroller has ceded control of the local bus, in response to a high on the hold signal by an external bus master, after the microcontroller has completed the current bus cycle. The assertion of hlda is accompanied by the tristating of den_n, rd_n, wr_n, s2_n–s0_n, ad15–ad0, s6, a19–a0, bhe_n, whb_n, wlb_n, and dt/r_n, followed by the driving high of the chip selects ucs_n, lcs_n, mcs3_n–mcs0_n, pcs6_n–pcs5_n, and pcs3_n–pcs0_n. The external bus master releases control of the local bus by the deassertion of hold that in turn induces the microcontroller to deassert the hlda. The microcontroller can take control of the bus if necessary (to execute a refresh for example), by deasserting hlda without the bus master first deasserting hold. This requires that the external bus master be able to deassert hold to permit the microcontroller to access the bus.

### 2.2.15 hold—Bus Hold Request (synchronous level-sensitive input)

This pin is pulled high to signal the microcontroller that the system requires control of the local bus.

The hold latency time (time between the hold and hlda) depends on the current processor activity when the hold is received. A hold request is second only to a DMA refresh request in priority of processor activity requests. If a hold request is received at the moment a DMA transfer starts, the hold latency can be up to 4 bus cycles. (This happens only on the IA186EM when a word transfer is taking place from an odd to an odd address.) This means that the latency may be 16 clock cycles without wait states. Furthermore, if lock transfers are being performed, then the latency time is increased during the locked transfer.

### 2.2.16 int0—Maskable Interrupt Request 0 (asynchronous input)

The int0 pin provides an indication that an interrupt request has occurred, and provided that int0 is not masked, program execution will continue at the location specified by the INT0 vector in the interrupt vector table. Although interrupt requests are asynchronous, they are synchronized internally and may be edge- or level-triggered. To ensure that it is recognized, the assertion of the interrupt request must be maintained until it is handled.

### 2.2.17 int1/select_n—Maskable Interrupt Request 1/Slave Select (both are asynchronous inputs)

The int1 pin provides an indication that an interrupt request has occurred, and provided that int1 is not masked, program execution will continue at the location specified by the int1 vector in the interrupt vector table. Although interrupt requests are asynchronous, they are synchronized internally and may be edge- or level-triggered. To ensure that it is recognized, the assertion of the interrupt request must be maintained until it is handled.

The select_n pin provides an indication to the microcontroller that an interrupt type has been placed on the address/data bus when the internal Interrupt Control Unit is slaved to an external interrupt controller. Before this can occur, however, the int0 pin must have already indicated an interrupt request has occurred.

### 2.2.18 int2/inta0_n/pio31—Maskable Interrupt Request 2 (asynchronous input)/Interrupt Acknowledge 0 (synchronous output)

The int2 pin provides an indication that an interrupt request has occurred, and provided that int2 is not masked, program execution will continue at the location specified by the int2 vector in the interrupt vector table. Although interrupt requests are asynchronous, they are synchronized internally and may be edge- or level-triggered. To ensure that it is recognized, the assertion of the interrupt request must be maintained until it is handled. When int0 is configured to be in cascade mode, int2 changes its function to inta0_n.

The inta0_n function indicates to the system that the microcontroller requires an interrupt type in response to the interrupt request int0 when the microcontroller's Interrupt Control Unit is in cascade mode. The peripheral device that issued the interrupt must provide the interrupt type.

### 2.2.19 int3/inta1_n/irq—Maskable Interrupt Request 3 (asynchronous input)/Interrupt Acknowledge 1 (synchronous output)/Interrupt Acknowledge (synchronous output)

The int3 pin provides an indication that an interrupt request has occurred. If int3 is not masked, program execution will continue at the location specified by the int3 vector in the interrupt vector table. Although interrupt requests are asynchronous, they are synchronized internally and may be edge- or level-triggered. To ensure that it is recognized, the assertion of the interrupt request must be maintained until it is handled. When int1 is configured to be in cascade mode, int3 changes its function to inta1_n.

The inta1_n function indicates to the system that the microcontroller requires an interrupt type in response to the interrupt request int1 when the microcontroller's Interrupt Control Unit is in cascade mode. The peripheral device that issued the interrupt must provide the interrupt type.

The signal on irq allows the microcontroller to output an interrupt request to the external master interrupt controller when the Interrupt Control Unit of the microcontroller is in slave mode.

### 2.2.20 int4/pio30—Maskable Interrupt Request 4 (asynchronous input)

The int4 pin provides an indication that an interrupt request has occurred, and provided that int4 is not masked, program execution will continue at the location specified by the int4 vector in the interrupt vector table. Although interrupt requests are asynchronous, they are synchronized internally and may be edge- or level-triggered. To ensure that it is recognized, the assertion of the interrupt request must be maintained until it is handled.

### 2.2.21 lcs_n/once0_n—Lower Memory Chip Select (synchronous output with internal pull-up)/ONCE Mode Request (input)

The lcs_n pin provides an indication that a memory access is occurring to the lower memory block. The size of the Lower Memory Block and its base address are programmable, with the size adjustable up to 512 Kbytes. The lcs_n is held high during bus hold.

The once0_n pin (ONCE – ON Circuit Emulation) and its companion pin, once1_n, define the microcontroller mode during reset. These two pins are sampled on the rising edge of res_n and if both are asserted low the microcontroller starts in ONCE mode, else it starts normally. In ONCE mode, all pins are tristated and remain so until a subsequent reset. To prevent the microcontroller from entering ONCE mode inadvertently, this pin has a weak pull-up that is only present during reset. This pin is not tristated during bus hold.

### 2.2.22 mcs2_n—mcs0_n (no pio, pio15, pio 14)—Midrange Memory Chip Selects (synchronous outputs with internal pull-up)

The mcs2_n and mcs0_n pins provide an indication that a memory access is in progress to the second or third midrange memory block. The size of the Midrange Memory Block and its base address are programmable. The mcs2_n – mcs0_n are held high during bus hold and have weak pull-ups that are only present during reset.

### 2.2.23 mcs3_n/rfsh_n (pio25)—Midrange Memory Chip Select (synchronous output with internal pull-up)/Automatic Refresh (synchronous output)

The mcs3_n pin provides an indication that a memory access is in progress to the fourth region of the midrange memory block. The size of the Midrange Memory Block and its base address are programmable. The mcs3_n is held high during bus hold and has a weak pull-up that is present only during reset.

The rfsh_n signal is timed for auto refresh to PSRAM or DRAM devices. The refresh pulse is output only when the PSRAM or DRAM mode bit is set (EDRAM register Bit [15]). This pulse is of 1.5 clock-pulse duration with the rest of the refresh cycle made up of a deassertion period such that the overall refresh time is met. This pin is not tristated during a bus hold.

### 2.2.24 nmi—Nonmaskable Interrupt (synchronous edge-sensitive input)

Unlike int4 – int0, this is the highest priority interrupt signal and cannot be masked. Upon the assertion of this interrupt (transition from Low to High), program execution is transferred to the nonmaskable interrupt vector in the interrupt vector table and this interrupt is initiated at the next instruction boundary. For recognition to be assured, the nmi pin must be held high for at least a clkouta period so that the transition from low to high is latched and synchronized internally. The interrupt will begin at the next instruction boundary.

The nmi is not involved in the priority resolution process that deals with the maskable interrupts and does not have an associated interrupt flag. This allows for a new nmi request to interrupt an nmi service routine that is already underway. When an interrupt is taken by the processor the interrupt flag IF is cleared, disabling the maskable interrupts. If the maskable interrupts are reenabled during the nmi service routine (e.g., by use of STI instruction), the priority resolution of maskable interrupts will be unaffected by the servicing of the non-maskable interrupt (NMI).

> Note: For this reason, it is strongly recommended that the NMI interrupt service routine does not enable the maskable interrupts.

### 2.2.25 pcs3_n–pcs0_n (pio19–pio16)—Peripheral Chip Selects 3–0 (synchronous outputs)

The pcs3_n–pcs0_n pins provide an indication that a memory access is underway for the corresponding region of the peripheral memory block (I/O or memory address space). The base address of the peripheral memory block is programmable. The pins are held high during both bus hold and reset. These outputs are asserted with the ad address bus over a 256-byte range each.

### 2.2.26 pcs5_n/a1—Peripheral Chip Select 5 (synchronous output)/Latched Address Bit 1 (synchronous output)

The pcs5_n signal provides an indication that a memory access is underway for the sixth region of the peripheral memory block (I/O or memory address space). The base address of the peripheral memory block is programmable. The pcs5_n is held high during both bus hold and reset. This output is asserted with the ad address bus over a 256-byte range.

This a1 pin provides an internally latched address bit 1 to the system when the EX bit (Bit [7]) in the mcs_n and pcs_n auxiliary (MPCS) register is 0. It retains its previously latched value during a bus hold.

### 2.2.27 pcs6_n/a2—Peripheral Chip Select 6 (synchronous output)/latched Address Bit 2 (synchronous output)

The pcs6_n signal provides an indication that a memory access is underway for the seventh region of the peripheral memory block (I/O or memory address space). The base address of the peripheral memory block is programmable. The pcs6_n is held high during both bus hold and reset. This output is asserted with the ad address bus over a 256-byte range.

The a2 pin provides an internally latched address Bit [2] to the system when the EX bit (Bit [7]) in the mcs_n and pcs_n auxiliary (MPCS) register is 0. It retains its previously latched value during a bus hold.

### 2.2.28 pio31–pio0—Programmable I/O Pins (asynchronous input/output open-drain)

There are 32 individually programmable I/O pins provided (see Table 15, Default Status of PIO Pins at Reset).

### 2.2.29 rd_n—Read strobe (synchronous output with tristate)

The rd_n pin provides an indication to the system that a memory or I/O read cycle is underway. It will not to be asserted before the ad bus is floated during the address to data transition. The rd_n is tristated during bus hold.

### 2.2.30 res_n—Reset (asynchronous level-sensitive input)

The res_n pin forces a reset on the microcontroller. Its Schmitt trigger allows POR generation via an RC network. When this signal is asserted, the microcontroller immediately terminates its present activity, clears its internal logic, and transfers CPU control to the reset address, FFFF0h.

The res_n must be asserted for at least 1 ms. Because it is synchronized internally it may be asserted asynchronously to clkouta. Furthermore, $v_{cc}$ must be within specification and clkouta must be stable for more than four of its clock periods for the period that res_n is asserted.

The microcontroller starts to fetch instructions 6.5 clkouta clock periods after the deassertion of res_n.

### 2.2.31 rfsh2_n/aden_n (IA188EM)—Refresh 2 (synchronous output with tristate)/Address Enable (input with internal pull-up)

The rfsh2_n indicates that a DRAM refresh cycle is being performed when it is asserted low. However, this is not valid in PSRAM mode where mcs3_n/rfsh_n is used instead.

If the aden_n pin is held high during POR, the ad bus (ao15–ao8 and ad7–ad0 for the IA188EM) is controlled during the address portion of the lcs and ucs bus cycles by the DA bit (Bit [7]) in the lcs and ucs registers. If the DA bit is 1, the address is accessed on the a19–a0 pins, reducing power consumption. The weak pull-up on this pin obviates the necessity of an external pull-up.

If the aden_n pin is held low during POR, the ad bus is used for both addresses and data without regard for the setting of the DA bits. The rfsh2_n/aden_n is sampled one crystal clock cycle after the rising edge of res_n and is tristated during bus holds and ONCE mode.

### 2.2.32 rxd/pio28—Receive Data (asynchronous input)

This signal connects asynchronous serial receive data from the system to the asynchronous serial port.

### 2.2.33 s2_n–s0_n—Bus Cycle Status (synchronous outputs with tristate)

These three signals inform the system of the type of bus cycle in progress. The s2_n may be used to indicate whether the current access is to memory or I/O, and s1_n may be used to indicate whether data is being transmitted or received. These signals are tristated during bus hold and hold acknowledge. The coding for these pins is presented in Table 10.

**Table 10. Bus Cycle Types for s2_n, s1_n, and s0_n**

| s2_n | s1_n | s0_n | Bus Cycle |
|---|---|---|---|
| 0 | 0 | 0 | Interrupt acknowledge |
| 0 | 0 | 1 | Read data from I/O |
| 0 | 1 | 0 | Write data to I/O |
| 0 | 1 | 1 | Halt |
| 1 | 0 | 0 | Instruction fetch |
| 1 | 0 | 1 | Read data from memory |
| 1 | 1 | 0 | Write data to memory |
| 1 | 1 | 1 | None (passive) |

### 2.2.34 s6/clkdiv2_n/pio29—Bus Cycle Status Bit 6 (synchronous output)/Clock Divide by 2 (input with internal pull-up)

The s6 signal is high during the second and remaining cycle periods (i.e., $t_2 - t_4$), indicating that a DMA-initiated bus cycle is underway. The s6 is tristated during bus hold or reset.

If the clkdiv2_n signal is held low during power-on-reset, the microcontroller enters clock divide-by-2 mode. In this mode, the PLL is disabled and the processor receives the external clock divided by 2. Sampling of this pin occurs on the rising edge of res_n.

> Note: If this pin is used as pio29 and configured as an input, care should be taken that it is not driven low during POR.

Because this pin has an internal pull-up, it is not necessary to drive the pin high even though it defaults to an input PIO.

### 2.2.35 sclk—Serial Clock (synchronous outputs with tristate)

Because this pin provides a slave device with a synchronous serial clock it permits synchronization of the transmit and receive data exchanges between the slave and the microcontroller. The sclk is the result of dividing the internal clock by 2, 4, 8, or 16, depending on the contents of the Synchronous Serial Control (SSC) register Bits [5–4]. Accessing either the SSR or SSD registers activates the sclk for eight cycles. When sclk is not active, the microcontroller hold is high.

### 2.2.36 sdata—Serial Data (synchronous inout)

The sdata pin connects a slave device to synchronous serial transmit and receive data. The last value is retained on this pin when it is inactive.

### 2.2.37 sden1–sden0—Serial Data Enables (synchronous outputs with tristate)

The sden1–sden0 pins facilitate the transfer of data on ports 1 and 0 of the Synchronous Serial Interface (SSI). Either sden1 or sden0 is asserted by the microcontroller at the start of the data transfer and is de-asserted when the transfer is completed. These pins are held low by the microcontroller when they are inactive.

### 2.2.38 srdy/pio6—Synchronous Ready (synchronous level-sensitive input)

This signal is an active high input synchronized to clkouta and indicates to the microcontroller that a data transfer will be completed by the addressed memory space or I/O device.

In contrast to the Asynchronous Ready (ardy), which requires internal synchronization, srdy permits easier system timing because it already synchronized. Tying srdy high will always assert this ready condition. Tying it low will give control to ardy.

### 2.2.39 tmrin0/pio11—Timer Input 0 (synchronous edge-sensitive input)

This signal may be either a clock or control signal for the internal Timer 0. The timer is incremented by the microcontroller after it synchronizes a rising edge of tmrin0. When not used, tmrin0 must be tied high, or when used as pio11, it is pulled up internally.

### 2.2.40 tmrin1/pio0—Timer Input 1 (synchronous edge-sensitive input)

This signal may be either a clock or control signal for the internal Timer 1. The timer is incremented by the microcontroller after it synchronizes a rising edge of tmrin1. When not used, tmrin1 must be tied high, or when used as pio0, it is pulled up internally.

### 2.2.41 tmrout0/pio10—Timer Output 0 (synchronous output)

This signal provides the system with a single pulse or a continuous waveform with a programmable duty cycle. It is tristated during a bus hold or reset.

### 2.2.42 tmrout1/pio1—Timer Output 1 (synchronous output)

This signal provides the system with a single pulse or a continuous waveform with a programmable duty cycle. It is tristated during a bus hold or reset.

### 2.2.43 txd/pio22—Transmit Data (asynchronous output)

This pin provides the system with asynchronous serial transmit data from the serial port.

### 2.2.44 ucs_n/once1_n—Upper Memory Chip Select (synchronous output)/ONCE Mode Request 1 (input with internal pull-up)

The ucs_n pin provides an indication that a memory access is in progress to the upper memory block. The size of the Upper Memory Block and its base address are programmable, with the size adjustable to 512 Kbytes. The ucs_n is held high during bus hold.

After power-on-reset, ucs_n is active low and program execution begins at FFFF0h. Its default configuration is a 64-Kbyte memory range from F0000h to FFFFFh.

The once0_n pin (ONCE – ON Circuit Emulation) and its companion pin, once1_n, define the microcontroller mode during reset. These two pins are sampled on the rising edge of res_n and if both are asserted low the microcontroller starts in ONCE mode, else it starts normally. In ONCE mode, all pins are tristated and remain so until a subsequent reset. To prevent the microcontroller from entering ONCE mode inadvertently, this pin has a weak pull-up that is only present during reset. This pin is not tristated during bus hold.

### 2.2.45 uzi_n/pio26—Upper Zero Indicate (synchronous output)

This pin allows the designer to determine if an access to the interrupt vector table is in progress by ORing it with Bits [15–10] of the address and data bus (ad15–ad10 on the IA186EM and ao15–ao10 on the IA188EM). The uzi_n is the logical OR of the inverted a19–a16 bits. It asserts in the first period of a bus cycle and is held throughout the cycle.

At reset, uzi_n should be pulled high or allowed to float. If this pin is pulled low at reset, the microcontroller enters a reserved clock test mode.

### 2.2.46 $v_{cc}$—Power Supply (input)

These pins supply power (+5V ±10%) to the microcontroller.

### 2.2.47 whb_n (IA186EM)—Write High Byte (synchronous output with tristate)

The whb_n and wlb_n pins indicate to the system which bytes of the data bus (upper, lower, or both) are taking part in a write cycle. The whb_n is asserted with ad15–ad8 and is the logical OR of bhe_n and wr_n. It is tristated during reset.

### 2.2.48 wlb_n/wb_n—Write Low Byte (IA186EM) (synchronous output with tristate)/Write Byte (IA188EM) (synchronous output with tristate)

The wlb_n and whb_n pins indicate to the system which bytes of the data bus (upper, lower, or both) are taking part in a write cycle. The wlb_n is asserted with ad7–ad0 and is the logical OR of ad0 and wr_n. It is tristated during reset.

On the IA188EM microcontroller, wb_n provides an indication that a write to the bus is occurring. It shares the same early timing as that of the non-multiplexed address bus, and is associated with ad7–ad0. It is tristated during reset.

### 2.2.49 wr_n—Write Strobe (synchronous output)

The wr_n pin indicates to the system that the data currently on the bus is to be written to a memory or I/O device. It is tristated during a bus hold or reset.

### 2.2.50 x1—Crystal Input (input)

The x1 and x2 pins are the connections for a fundamental-mode or third-overtone, parallel-resonant crystal used by the internal oscillator circuit. An external clock source for the microcontroller is connected to x1. The x2 is left unconnected.

### 2.2.51 x2—Crystal Input (input)

The x1 and x2 pins are the connections for a fundamental-mode or third-overtone, parallel-resonant crystal used by the internal oscillator circuit. An external clock source for the microcontroller is connected to x1. The x2 is left unconnected.

## 2.3 Pins Used by Emulators

The following pins are used by emulators:

- a19–a0
- ao15–ao8 (on the IA188EM)
- ad7–ad0
- ale
- bhe_n/aden_n (on the IA186EM)
- clkouta
- rfsh2_n/aden_n (on the IA188EM)
- rd_n
- s2_n–s0_n
- s6/lock_n/clkdiv2_n
- uzi_n

Emulators require that s6/lock_n/clkdiv2_n and uzi_n be configured as their normal functions (i.e., as s6 and uzi_n, respectively). Holding bhe_n/aden_n (IA186EM) or rfsh_n/aden_n (IA188EM) low during the rising edge of res_n, will cause s6 and uzi_n to be configured in their normal functions at reset instead of as PIOs.

# 3. Maximum Ratings, Thermal Characteristics, and DC Parameters

The absolute maximum ratings, thermal characteristics, and DC parameters are provided in Tables 11 through 13, respectively.

### Table 11. IA186EM and IA188EM Absolute Maximum Ratings

| Parameter | Rating |
|---|---|
| Storage Temperature | −65°C to +125°C |
| Voltage on any Pin with Respect to $v_{ss}$ | −0.5V to +($v_{cc}$ + 0.5) V |

### Table 12. IA186EM and IA188EM Thermal Characteristics

| Symbol | Characteristic | Value |
|---|---|---|
| $T_A$ | Ambient Temperature | -40°C to 85°C |

### Table 13. DC Characteristics Over Commercial Operating Ranges

| Symbol | Parameter Description | Test Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply Voltage (@ 5V Operation) | - | 4.5 | 5.5 | V |
| $V_{IL}$ | Input Low Voltage (Except x1) | – | −0.5 | 0.8 | V |
| $V_{IL1}$ | Clock Input Low Voltage (x1) | – | −0.5 | 0.8 | V |
| $V_{IH}$ | Input High Voltage (Except res_n and x1) | – | 2.0 | $v_{cc}$ +0.5 | V |
| $V_{IH1}$ | Input High Voltage (res_n) | – | 2.4 | $v_{cc}$ +0.5 | V |
| $V_{IH2}$ | Clock Input High Voltage (x1) | – | $v_{cc}$–0.8 | $v_{cc}$ +0.5 | V |
| $V_{OL}$ | Output Low Voltages | $I_{OL}$ = 2.5 mA (s2_n–s0_n) | – | 0.45 | V |
| | | $I_{OL}$ = 2.0 mA (other) | – | 0.45 | V |
| $V_{OH}$ | Output High Voltages[a] | $I_{OH}$ = −2.4 mA @ 2.4 V | 2.4 | $v_{cc}$ +0.5 | V |
| | | $I_{OH}$ = −200 µA @ $v_{cc}$ −0.5 | $v_{cc}$ −0.5 | $v_{cc}$ | V |
| $I_{CC}$ | Power Supply Current @ 0°C | $v_{cc}$ = 5.5 V[b] | – | 5.9 | mA/ MHz |
| $I_{LI}$ | Input Leakage Current @ 0.5 MHz | 0.45 V ≤ $V_{IN}$ ≤ $v_{cc}$ | – | ±10 | µA |
| $I_{LO}$ | Output Leakage Current @ 0.5 MHz | 0.45 V ≤ $V_{OUT}$ ≤ $v_{cc}$ [c] | – | ±10 | µA |
| $V_{CLO}$ | Clock Output Low | $I_{CLO}$ = 4.0 mA | – | 0.45 | V |
| $V_{CHO}$ | Clock Output High | $I_{CHO}$ = −500 µA | $v_{cc}$ −0.5 | – | V |

[a]The lcs_n/once0_n, mcs3_n–mcs0_n, ucs_n/once1_n, and rd_n pins have weak internal pullup resistors. Loading the lcs_n/once0_n and ucs_n/once1_n pins in excess of $I_{OH}$ = −200 µA during reset can cause the device to go into ONCE mode.

[b]Current is measured with the device in reset with the x1 and x2 driven and all other non-power pins open but held high or low.

[c]Testing is performed with the pins floating, either during hold or by invoking the ONCE mode.

# 4. Device Architecture

A functional block diagram of the IA186EM/IA188EM is shown in Figure 7. This microcontroller consists of the following functional blocks.

- Bus Interface and Control (BIC)
- Peripheral Control and Registers
- Chip Selects and Control (CSC)
- Programmable I/O
- Clock and Power Management
- DMA
- Interrupt Controller
- Timers
- Asynchronous Serial Ports
- Synchronous Serial Interface

## 4.1 Bus Interface and Control

BIC manages all accesses to external memory and external peripherals. These peripherals may be mapped either in memory space or I/O space. The BIC supports both multiplexed and non-multiplexed bus operations. Multiplexed address and data are provided on the ad15–ad0 bus, while a non-multiplexed address is provided on the a19–a0 bus. The a bus provides address information for the entire bus cycle ($t_1$–$t_4$), while the ad bus provides address information only during the first phase of the bus cycle ($t_1$). For more details regarding bus cycles, see the AC waveforms at the end of this datasheet.

The IA186EM microcontroller provides two signals that serve as byte write enables, write high byte (whb_n) and write low byte (wlb_n). The IA188EM microcontroller requires only a single write byte (wb_n) signal to support its 8-bit data bus. The whb_n is the logical OR of the bhe_n and wr_n. The wlb_n is the logical OR of ad0 and wr_n. The wlb_n is the logical OR of ad0 and wr_n. The wb_n is low whenever a byte is written to the IA188EM data bus ad7–ad0.

The byte write enables are driven in conjunction with the non-multiplexed address bus a19–a0 to support the timing requirements of common SRAMs.

The BIC also provides support for PSRAM devices. PSRAM is supported in only the lower chip select (lcs_n) area. In order to support PSRAM, the CSC must be appropriately programmed (see Section 4.7, Chip Selects).

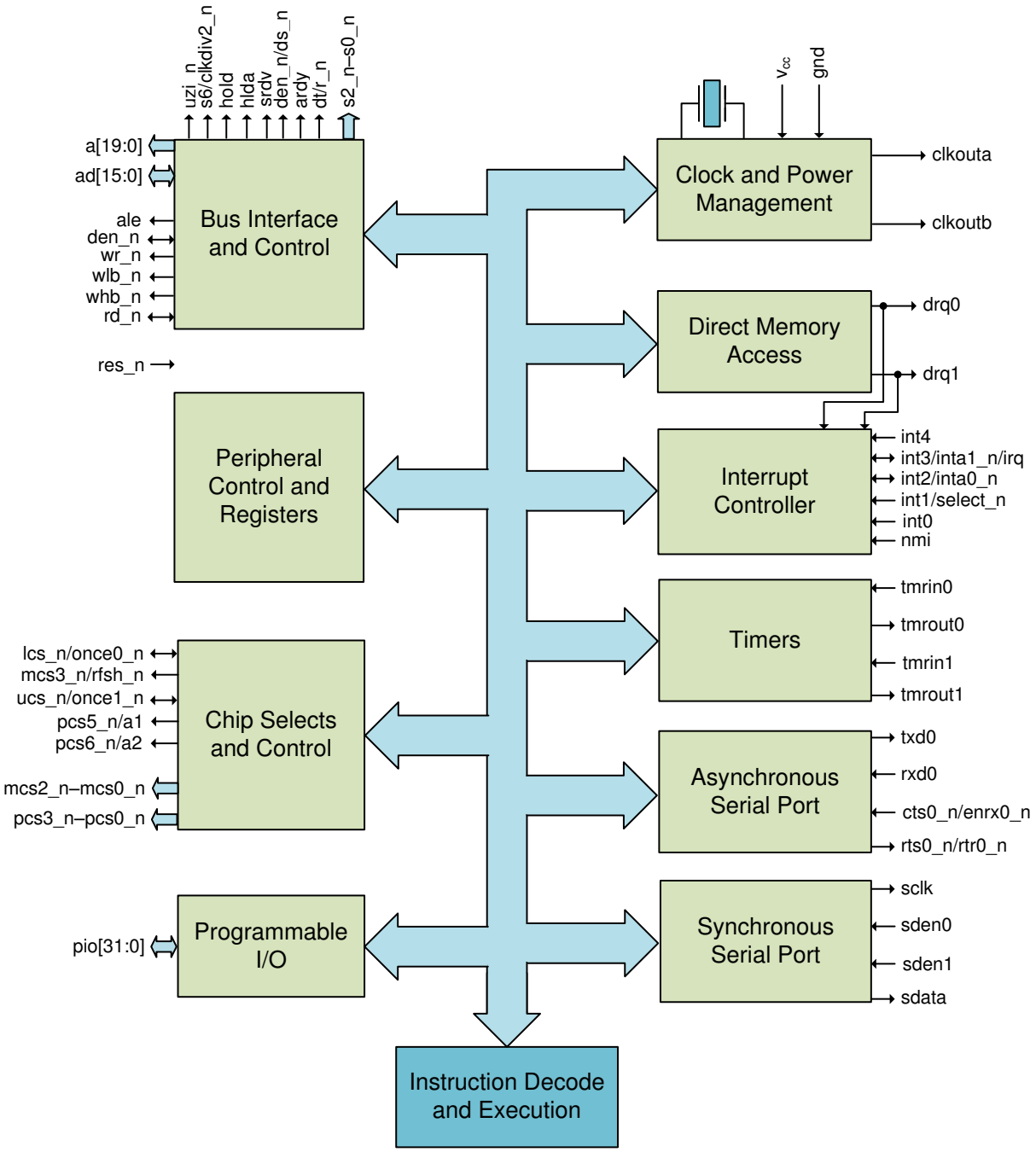



**Figure 7. Functional Block Diagram**

Note: See pin descriptions for pins that share other functions with PIO pins. Pins pwd, int5, int6, rts1_n/rtr1_n, and cts1_n/enrx1_n are multiplexed with int2_n/inta0_n, drq0_n, drq0_n, pcs3_n, and pcs2_n, respectively.

## 4.2 Clock and Power Management

A phase-lock-loop (PLL) and a second programmable system clock output (clkoutb) are included in the clock and power management unit. The internal clock is the same frequency as the crystal but with a duty cycle of 45% to 55 %, as a worst case, generated by the PLL obviating the need for an x2 external clock. A POR resets the PLL (see Figure 8).

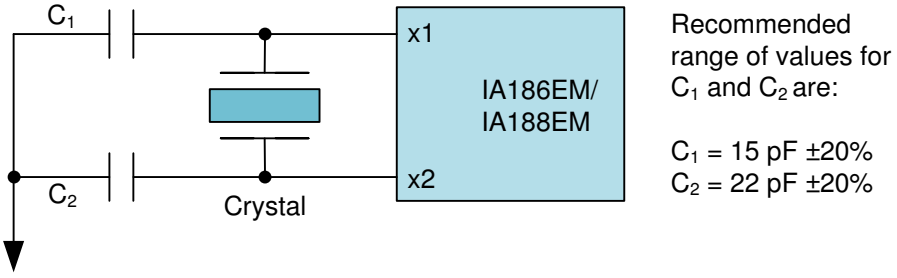



**Figure 8. Crystal Configuration**

## 4.3 System Clocks

If required, the internal oscillator can be driven by an external clock source that should be connected to x1, leaving x2 unconnected.

The clock outputs clkouta and clkoutb may be enabled or disabled individually (Power-Save Control register (PDCON) Bits [11–8]). These clock control bits allow one clock output to run at PLL frequency and the other to run at the power-save frequency (see Figure 9).

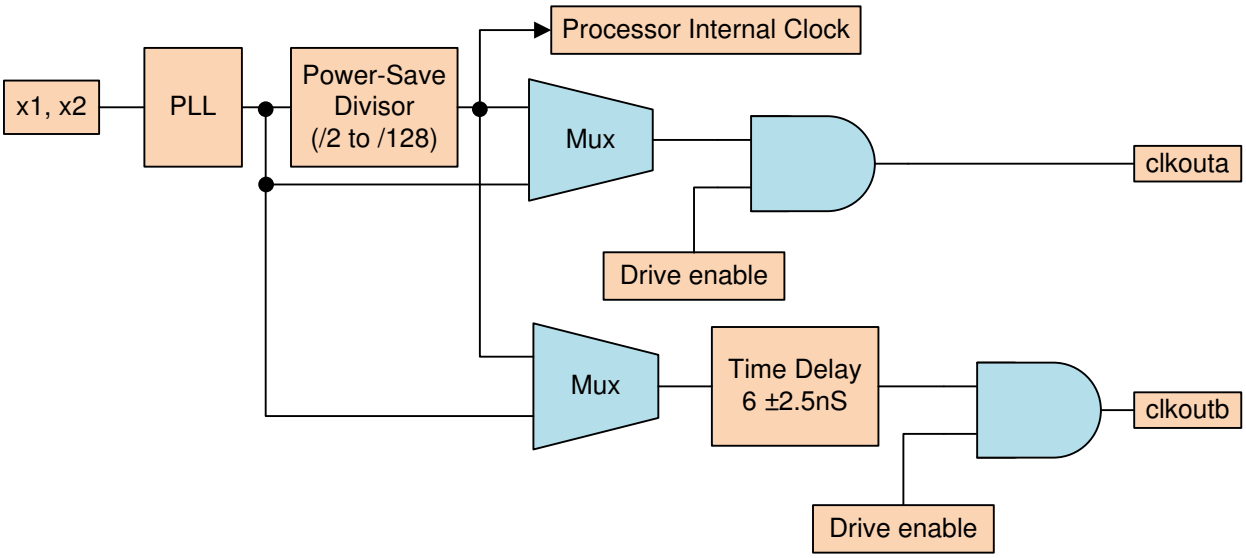



**Figure 9. Organization of Clock**

## 4.4 Power-Save Mode

The operation of the CPU and peripherals operate at a slower clock frequency when in power save mode, reducing power consumption and thermal dissipation. Should an interrupt occur, the microcontroller returns to its normal operating frequency automatically on the internal clock's next rising edge in $t_3$. Any clock-dependent devices should be reprogrammed for the change in frequency during the power-save mode period.

## 4.5 Initialization and Reset

The highest priority interrupt, res_n (Reset) must be held low for 1 mS during power-up to initialize the microcontroller correctly. This operation makes the device cease all instruction execution and local bus activity. The microcontroller begins instruction execution at physical address FFFF0h when res_n becomes inactive and after an internal processing interval with ucs_n is asserted and three wait states. Reset also sets up certain registers to predetermined values and resets the Watchdog timer.

## 4.6 Reset Configuration Register

The data on the address/data bus (ad15–ad0 for the IA186EM, ao15–ao8 and ad7–ad0 for the IA188EM) are written into the Reset Configuration register when reset is low. This data is system dependent and is held in the Reset Configuration register after Reset is de-asserted. This configuration data may be placed on the address/data bus by using weak external pull-up and pull-down resistors or applied to the bus by an external driver, as the processor does not drive the bus during reset. It is a method of supplying the software with some initial data after a reset; for example, option jumper positions.

## 4.7 Chip Selects

Chip-select generation is programmable for memories and peripherals. Programming is also available to produce ready- and wait-state generation plus latched address bits a1 and a2. For all memory and I/O cycles, the chip-select lines are active within their programmed areas, regardless of whether they are generated by the internal DMA unit or the CPU.

There are six chip-select outputs for memories and a further six for peripherals whether in memory or I/O space. The memory chip-selects are able to address three memory ranges, whereas the peripheral chip-selects are used to address 256-byte blocks that are offset from a programmable base address. Writing to a chip-select register enables the related logic even if the pin in question has another function (e.g., if the pin is programmed to be a PIO).

## 4.8 Chip-Select Timing

For normal timing, the ucs_n and lcs_n outputs are asserted with the non-multiplexed address bus.

## 4.9 Ready- and Wait-State Programming

Each of the memory or peripheral chip-select lines can have a ready signal programmed that can be the ardy or srdy signal. The chip-select control registers (UMCS, LMCS, MMCS, PACS, and MPCS) have a single bit that selects whether the external ready signal is to be used or not (R2, Bit [2]). R1 and R0 (Bits [1–0]) in these registers control the number of wait states that are inserted during each access to a memory or peripheral location (from 0 to 3). The control registers for pcs3_n–pcs0_n use three bits, R3, R1–R0 (Bits [3], [1–0]) to provide 5, 7, 9, and 15 wait-states in addition to the original values of 0 to 3 wait states.

In the case where an external ready has been selected as required, internally programmed wait-states will always be completed before the external ready can finish or extend a bus cycle. As an example, consider a system in which the number of wait states to be inserted has been set to 3. The external ready pin is sampled by the processor during the first wait cycle. The access is completed after 7 cycles (4 cycles plus 3 wait cycles) if the ready is asserted. Alternatively, if the ready is not asserted during the first wait cycle, the access is prolonged until ready is asserted and two more wait states are inserted followed by $t_4$.

## 4.10 Chip Select Overlap

Overlapping chip selects are configurations where more than one chip select is asserted for the same physical address. For example, if PCS is configured in I/O space with LCS or any other chip select configured for memory, address 00000h is not overlapping the chip selects.

Note: It is not recommended that multiple chip-select signals be asserted for the same physical address, although it may be inescapable in certain systems. If this is the case, then all overlapping chip-selects must have the same external ready configuration and the same number of wait states to be inserted into access cycles.

Internal signals are employed to access the peripheral control block (PCB) and these signals serve as chip selects that are configured with no wait states and no external ready. Therefore, the PCB can be programmed with addresses that overlap external chip selects only if these chip selects are configured in the same manner.

Note: Caution is advised in the use of the DA bit in the LMCS or UMCS registers when overlapping an additional chip select with either the lcs_n or ucs_n. Setting the DA bit to 1 prevents the address from being driven onto the AD bus for all accesses for which the respective chip select is active, including those for which multiple selects are active.

The mcs_n and pcs_n pins are dual-purpose pins, either as chip selects or PIO inputs or outputs. However, the respective ready- and wait-state configurations for their chip-select function will be in effect regardless of the function for which these two pins are programmed. This requires that even if these pins are configured as PIO and enabled (by writing to the MMCS and MPCS registers for the mcs_n chip selects and to the PACS and MPCS registers for the pcs_n chip selects), the ready- and wait-state settings for them must agree with those for any overlapping chip selects as though they were configured as chip selects.

Although pcs4_n is not available as an external pin, it has ready- and wait-state logic and must follow the rules for overlapping chip-selects. Conversely, pins pcs6_n and pcs5_n have ready- and wait-state logic that is disabled when configured as address bits a2 and a1, respectively.

Note: If chip-select configuration rules are not followed, the processor may hang with the appearance of waiting for a ready signal even in a system where ready (ardy or srdy) is always set to 1.

## 4.11 Upper Memory Chip Select

The ucs_n chip select is for the top of memory. On reset, the microcontroller begins fetching and executing instructions at memory location FFFF0h. As a result, upper memory is usually used for instruction memory. To this end, ucs_n is active on reset and has a memory range of 64 Kbytes (F0000h to FFFFFh) by default, along with external ready required and 3 wait states automatically inserted. The lower boundary of ucs_n is programmable to provide ranges of 64 to 512 Kbytes.

## 4.12 Low Memory Chip Select

The lcs_n chip-select is for lower memory. As the interrupt vector table is at the bottom of memory beginning at 00000h, this pin us usually used for control data memory. Unlike ucs_n, this pin is inactive on reset, but can be activated by any read or write to the LMCS register.

## 4.13 Midrange Memory Chip Selects

There are four midrange chip selects, mcs3_n–mcs0_n, which may be used in a user-located memory block. With some exceptions, the base address of the memory block may be located anywhere in the 1-Mbyte memory address space (those used by the ucs_n and lcs_n chip selects, as well as the pcs6_n, pcs5_n, and pcs3_n–pcs0_n, are excluded). If the pcs_n chip selects are mapped to I/O space, then the MCS address range can overlap the PCS address range.

Both the Midrange Memory Chip Select (MMCS) register and the MCS and PCS auxiliary (MPCS) registers are used to program the four midrange chip selects. The MPCS register is used to configure the block size, whereas the MMCS register configures the base address, the ready condition, and the wait states of the memory block accessed by the mcs_n pin. The chip selects (mcs3_n–mcs0_n) are activated by performing a read or write operation of the MMCS and MPCS registers. The assertion of the MCS outputs occurs with the same timing as the multiplexed AD address bus (ad15–ad0 on the IA186EM or ao15–ao8 and ad7–ad0 on the IA188EM). The a19–a0 may be used for address selection, but the timing will be delayed by a half clock cycle over the timing used for the ucs_n and lcs_n.

## 4.14 Peripheral Chip Selects

There are six peripheral chip selects (pcs6_n, pcs5_n, and pcs3_n–pcs0_n) that may be used within a user-defined memory or I/O block. The base address of this user-defined memory block can be located anywhere within the 1-Mbyte memory address space except for the spaces associated with the ucs_n, lcs_n, and mcs_n chip selects. Or it may be programmed to the 64 Kbyte I/O space. The pcs4_n is not available.

Both the Peripheral Chip Select (PACS) register and the MCS and PCS Auxiliary register (MPCS) registers are used to program the six peripheral chip selects pcs6_n, pcs5_n, and pcs3_n–pcs0_n. The PACS register sets the base address, the ready condition, and the wait states for the pcs3_n–pcs0_n outputs.

The MPCS register configures pcs6_n and pcs5_n pins as either chip selects or address pins a1 and a2, respectively. When these pins are chip selects, the MPCS register also configures them as being active during memory or I/O bus cycles and during their ready and wait states.

None of the pcs_n pins are active at reset. Both the Peripheral Chip Select (PACS) register and the MCS and PCS Auxiliary register (MPCS) registers must be read or written to activate the pcs_n pins as chip selects.

The pcs6_n and pcs5_n may be programmed to have 0 to 3 wait states, whereas pcs3_n–pcs0_n may be programmed to have these and 5, 7, 9, and 15 wait states.

## 4.15 Refresh Control

The Refresh Control Unit (RCU) generates refresh bus cycles. The RCU generates a memory read request after a programmable period of time to the bus interface unit.

The ENA bit in the Enable RCU register (EDRAM) enables refresh cycles, operating off the processor internal clock. If the processor is in power-save mode, the RCU must be reconfigured for the new clock rate.

If the hlda pin is asserted when a refresh request is initiated (indicating a bus hold condition), the processor disables the hlda pin to allow a refresh cycle to be performed. The external circuit bus master must deassert the hold signal for at least one clock period to permit the execution of the refresh cycle.

## 4.16 Interrupt Control

Interrupt requests originate from a variety of internal and external sources that are arranged by the internal interrupt controller in priority order and presented one by one to the processor.

Six external interrupt sources—five maskable (int4–int0) and one nonmaskable (NMI)—are connected to the processor and six internal interrupt sources (three timers, two DMA channels, and the asynchronous serial port that are not brought out to external pins).

The five external maskable interrupt request pins can be used as direct interrupt requests. However, should more interrupts be needed, int3–int0 may be used with the 82C59A-compatible external interrupt controller. By programming the internal interrupt controller to slave mode, a 82C59A-compatible external interrupt controller can be used as the system master. Interrupt nesting can be used in all cases that permit interrupts of a higher priority to interrupt those of a lower priority.

When an interrupt is accepted, other interrupts are disabled, but may be re-enabled by setting the Interrupt Enable Flag (IF) in the Processor Status Flags register during the Interrupt Service Routine (ISR). Setting IF permits interrupts of equal or greater priority to interrupt the currently running ISR.

Further interrupts from the same source will be blocked until the corresponding bit in the In-Service register (INSERV) is cleared. Special Fully Nested mode (SFNM) is invoked for int0 and int1 by the SFNM bit in the INT0 and INT1 control register, respectively, when this bit is set to 1. In this mode, a new interrupt may be generated by these sources regardless of the in-service bit. The following table shows the priorities of the interrupts at POR.

### 4.16.1 Interrupt Types

Table 14 presents interrupt names, types, vector table address, End-of-Interrupt (EOI) type, overall priority, and related instructions.

**Table 14. Interrupt Types**

| Interrupt Name | Interrupt Type | Vector Table Address | EOI Type | Overall Priority | Related Instructions |
|---|---|---|---|---|---|
| Divide Error Exception[a] | 00h | 00h | NA | 1 | DIV, IDIV |
| Trace Interrupt[b] | 01h | 04h | NA | 1A | All |
| Non-maskable Interrupt (NMI) | 02h | 08h | NA | 1B | – |
| Breakpoint Interrupt[a] | 03h | 0ch | NA | 1 | INT3 |
| INT0 Detected Overflow Exception[a] | 04h | 10h | NA | 1 | INT0 |
| Array Bounds Exception[a] | 05h | 14h | NA | 1 | BOUND |
| Unused Opcode Exception[a] | 06h | 18h | NA | 1 | Undefined Opcodes |
| ESC Opcode Exception[a,c] | 07h | 1ch | NA | 1 | ESC Opcodes |
| Timer 0 Interrupt[d,e] | 08h | 20h | 08h | 2A | – |
| Timer 1 Interrupt[d,e] | 12h | 48h | 08h | 2B | – |
| Timer 2 Interrupt[d,e] | 13h | 4ch | 08h | 2C | – |
| Reserved | 09h | 24h | – | – | – |
| DMA 0 Interrupt[e] | 0ah | 28h | 0ah | 3 | – |
| DMA 1 Interrupt[e] | 0bh | 2ch | 0bh | 4 | – |
| INT0 Interrupt | 0ch | 30h | 0ch | 5 | – |
| INT1 Interrupt | 0dh | 34h | 0dh | 6 | – |
| INT2 Interrupt | 0eh | 38h | 0eh | 7 | – |
| INT3 Interrupt | 0fh | 3ch | 0fh | 8 | – |
| INT4 Interrupt[f] | 10h | 40h | 10h | 9 | – |
| Watchdog Timer Interrupt[f] | 11h | 44h | 11h | 9 | – |
| Asynchronous Serial Port Interrupt[f] | 14h | 50h | 14h | 9 | – |
| Reserved | 15h–1fh | 54h–7ch | – | – | – |

Note: If the priority levels are not changed, the default priority level will be used for the interrupt sources.
[a]Instruction execution generates interrupts.
[b]Performed in the same manner as for the 8086 and 8088.
[c]An ESC opcode causes a trap.
[d]Because only one IRQ is generated for the three timers, they share priority level with other sources. The timers have an interrupt priority order among themselves (2A > 2B > 2C).
[e]These interrupt types are programmable in slave mode.
[f]Not available in slave mode.

## 4.17 Timer Control

The IA186EM and IA188EM each have three 16-bit programmable timers. Timer 0 and Timer 1 each has an input and output connected to external pins that permits it to count or to time events as well as to produce variable duty-cycle waveforms or non-repetitive waveforms. Timer 1 can also be configured as a Watchdog timer.

Because Timer 2 does not have external connections, it is confined to internal functions such as real-time coding, time-delay applications, a prescaler for Timer 0 and Timer 1, or to synchronize DMA transfers.

The Peripheral Control Block contains eleven 16-bit registers to control the programmable timers. Each timer-count register holds the present value of its associated timer and may be read from or written to whether or not the timer is in operation. The microcontroller increments the value of the timer-count register when a timer event takes place.

The value stored in a timer's associated maximum count register determines its maximum count value. Upon reaching it, the timer count register is reset to 0 in the same clock cycle that this count was attained. The timer count register does not store this maximum value. Both Timer 0 and Timer 1 have a primary and a secondary maximum count register that permits each to alternate between two discrete maximum values.

Timer 0 and Timer 1 may have the maximum count registers configured in either primary only or both primary and secondary. If the primary only is configured to operate, on reaching the maximum count, the output pin will go low for one clock period. If both the primary and secondary registers are enabled, the output pin reflects the state of the register in control at the time. This generates the required waveform that is dependent on the two values in the maximum count registers.

Because they are polled every fourth clock period, the timers can operate at a quarter of the internal clock frequency. Although an external clock may be used, the timer output may take six clock cycles to respond to the input.

## 4.18 Direct Memory Access (DMA)

DMA frees the CPU from involvement in transferring data between memory and peripherals over either one or both high-speed DMA channels. Data may be transferred from memory to I/O, I/O to memory, memory to memory, or I/O to I/O. DMA channels can be connected to the asynchronous serial port.

The IA186EM supports the transfer of both bytes and words to and from even or odd addresses. It does not support word transfers to memory that is configured for byte accesses. The IA188EM does not support word transfers at all. Each data transfer will take two bus cycles (a minimum of 8 clock cycles).

There are three sources of DMA requests for both DMA channels:

- The channel request pin (drq1–drq0)
- Timer 2
- The system software.

Each channel may be programmed to have a different priority either to resolve a simultaneous DMA request or to interrupt a transfer on the other channel.

## 4.19 DMA Operation

The PCB contains six registers for each DMA channel to control and specify the operation of the channel (see Figure 10):

- Two registers to store a 20-bit source address
- Two registers to store a 20-bit destination address
- One 16-bit transfer-count register
- One 16-bit control register

The number of DMA transfers required is designated in the DMA Transfer Count register and may contain up to 64 Kbytes or words. It will end automatically. DMA channel function is defined by the control registers. Like the other five registers, these may be changed at any time (including during a DMA transfer) and are implemented immediately.

## 4.20 DMA Channel Control Registers

See Section 5.1.8, D1CON (0dah) and D0CON (0cah). The DMA channel control registers specify the following:

- Whether the data destination is in memory or I/O space (Bit [15])
- Whether the destination address is incremented, decremented, or unchanged after each transfer (Bits [14–13])
- Whether the data source is in memory or I/O space (Bit [12])
- Whether the source address is incremented, decremented, or unchanged after each transfer (Bits [11–10])
- Whether DMA transfers cease upon reaching a designated count (Bit [9])
- Whether the last transfer generates an interrupt (Bit [8])
- Synchronization mode (Bits [7–6])

- The relative priority of one DMA channel with respect to the other (Bit [5])
- Acceptance of DMA requests from Timer 2 (Bit [4])
- Byte or Word transfers (Bit [0])

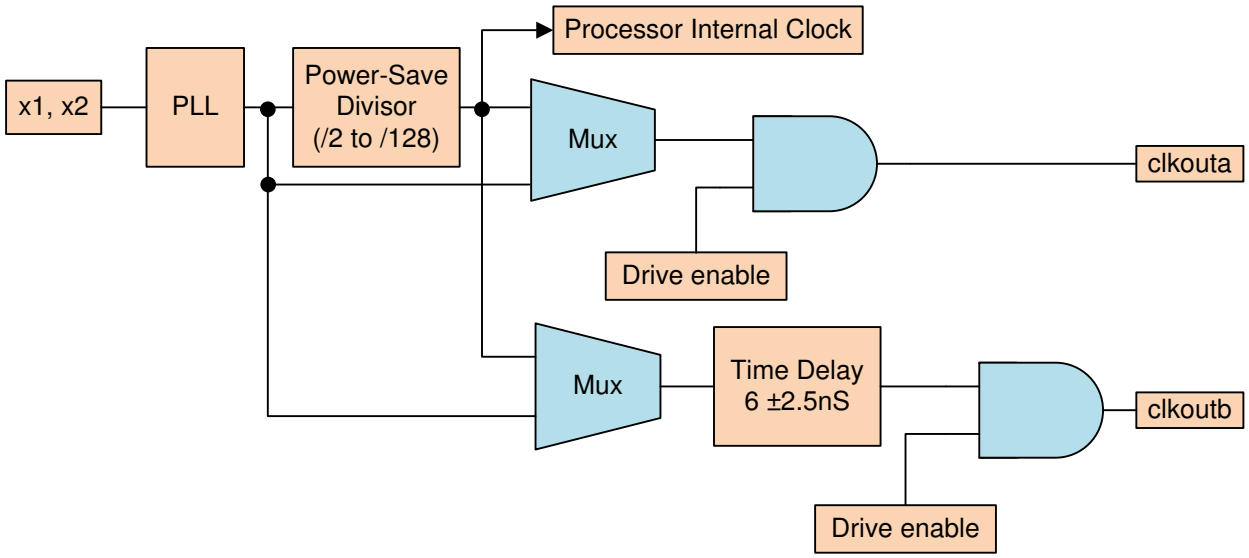



**Figure 10. DMA Unit**

## 4.21 DMA Priority

With the exception of word accesses to odd memory locations or between locked memory addresses, DMA transfers have a higher priority than CPU transfers. Because the CPU cannot access memory during a DMA transfer and a DMA transfer cannot be suspended by an interrupt request, continuous DMA activity will increase interrupt delay. An NMI request halts any DMA activity, however, enabling the CPU to respond promptly to the request.

## 4.22 Asynchronous Serial Port

The asynchronous serial port employs standard industry communication protocols in its implementation of full duplex, bi-directional data transfers. The port can be either the source or destination of DMA transfers.

The following features are supported:

- Full-duplex data transfers
- 7-, 8-, or 9-bit data transfers
- Odd, even, or no parity
- One or two stop bits

- Error detection provided by parity, framing, or overrun errors
- Hardware handshaking is achieved with the following selectable control signals:
  - Clear to send (cts_n)
  - Enable receiver request (enrx_n)
  - Ready to send (rts_n)
  - Ready to receive (rtr_n)

- DMA to and from the port
- The port has its own maskable interrupt
- The port has an independent baud-rate generator
- Maximum baud rate is 1/32 of the processor clock
- Transmit and receive lines are double-buffered

In power-save mode the baud rate generator divide factor must be re-programmed to compensate for the change in clock rate.

### 4.23 Synchronous Serial Port

The synchronous serial port allows the microcontrollers to communicate with ASICs that are required to be programmed but have a pin shortage. The four-pin interface allows half-duplex, bi-directional data transfer at a maximum of 20 Mbits/sec with a 40-MHz CPU clock.

The synchronous serial interface of the IA186EM/ IA188EM operates as the master port in a master/slave arrangement.

There are four pins in the synchronous serial interface for communication with the system elements. These pins are two enables (SDEN0 and SDEN1), a clock (SCLK), and a data pin (SDATA).

In power-save mode, the baud rate generator divide factor must be re-programmed to compensate for the change in clock rate.

### 4.24 Programmable I/O (PIO)

Thirty-two pins are programmable as I/O signals (PIO). Table 15 presents them in both numeric and alphabetic order. Because programming a pin as a PIO disables its normal function, it should be done only if the normal function is not required. A PIO pin can be programmed as an input or output with or without a weak pull-up or pull-down. A PIO pin can be also programmed as an open-drain output. Each PIO pin regains default status after a POR.

## Table 15. Default Status of PIO Pins at Reset

| PIO No. | Associated Pin | Power-On Reset Status |
|---|---|---|
| 0 | tmrin1 | Input with pull-up |
| 1 | tmrout1 | Input with pull-down |
| 2 | pcs6_n/a2 | Input with pull-up |
| 3 | pcs5_n/a1 | Normal operation[a] |
| 4 | dt/r_n | Normal operation[a] |
| 5 | den_n | Normal operation[a] |
| 6 | srdy | Normal operation[a] |
| 7[b] | a17 | Normal operation[a] |
| 8[b] | a18 | Normal operation[a] |
| 9[b] | a19 | Normal operation[a] |
| 10 | tmrout0 | Input with pull-down |
| 11 | tmrin0 | Input with pull-up |
| 12 | drq0 | Input with pull-up |
| 13 | drq1 | Input with pull-up |
| 14 | mcs0_n | Input with pull-up |
| 15 | mcs1_n | Input with pull-up |
| 16 | pcs0_n | Input with pull-up |
| 17 | pcs1_n | Input with pull-up |
| 18 | pcs2_n | Input with pull-up |
| 19 | pcs3_n | Input with pull-up |
| 20 | sclk | Input with pull-up |
| 21 | sdata | Input with pull-up |
| 22 | sden0 | Input with pull-up |
| 23 | sden1 | Input with pull-up |
| 24 | mcs2_n | Input with pull-up |
| 25 | mcs3_n/rfsh_n | Input with pull-up |
| 26[b,c] | uzi_n | Input with pull-up |
| 27 | txd | Input with pull-up |
| 28 | rxd | Input with pull-up |
| 29[b,c] | s6/clkdiv2 | Input with pull-up |
| 30 | int4 | Input with pull-up |
| 31 | int2 | Input with pull-up |

| Associated Pin | PIO No. | Power-On Reset Status |
|---|---|---|
| a17 | 7 | Normal operation[a] |
| a18 | 8 | Normal operation[a] |
| a19 | 9 | Normal operation[a] |
| den_n/ds_n | 5 | Normal operation[a] |
| drq0 | 12 | Input with pull-up |
| drq1 | 13 | Input with pull-up |
| dt/r_n | 4 | Normal operation[a] |
| int2/ | 31 | Input with pull-up |
| int4 | 30 | Input with pull-up |
| mcs0_n | 14 | Input with pull-up |
| mcs1_n | 15 | Input with pull-up |
| mcs2_n | 24 | Input with pull-up |
| mcs3_n/rfsh_n | 25 | Input with pull-up |
| pcs0_n | 16 | Input with pull-up |
| pcs1_n | 17 | Input with pull-up |
| pcs2_n | 18 | Input with pull-up |
| pcs3_n | 19 | Input with pull-up |
| pcs5_n/a1 | 3 | Input with pull-up |
| pcs6_n/a2 | 2 | Input with pull-up |
| rxd | 28 | Input with pull-up |
| s6/clkdiv2 | 29 | Input with pull-up[b,c] |
| sclk | 20 | Input with pull-up |
| sdata | 21 | Input with pull-up |
| sden0 | 22 | Input with pull-up |
| sden1 | 23 | Input with pull-up |
| srdy | 6 | Normal operation[d] |
| tmrin0 | 11 | Input with pull-up |
| tmrin1 | 0 | Input with pull-up |
| tmrout0 | 10 | Input with pull-down |
| tmrout1 | 1 | Input with pull-down |
| txd | 27 | Input with pull-up |
| uzi_n | 26 | Input with pull-up |

[a]Input with pullup option available when used as PIO.
[b]Emulators use these pins and also a15–a0, ad15–ad0 (IA186EM), ale, bhe_n (IA186EM), clkouta, nmi, res_n, and s2_n–s0_n.
[c]If bhe_n/aden_n (IA186EM) or rfsh_n/aden_n (IA188EM) is held low during POR, these pins will revert to normal operation.
[d]Input with pulldown option available when used as PIO.

These default status settings may be changed as desired.

After POR, a19–a17, the three most significant bits of the address bus, start with their normal function, allowing the processor to begin fetching instructions from the boot address FFFF0h. Normal function is also the default setting for dt/r_n, den_n, and srdy after POR.

If the ad15–ad0 bus override is enabled, s6/clkdiv2_n and uzi_n automatically return to normal operation. The ad15–ad0 bus override is enabled if either the bhe_n/aden_n for the IA186EM or the rfsh2_n/aden_n for the IA188EM is held low during POR.

# 5. Peripheral Architecture

## 5.1 Control and Registers

The on-chip peripherals in the IA186EM/IA188EM are controlled from a 256-byte block of internal registers. Although these registers are actually located in the peripherals they control, they are addressed within a single 256-byte block of I/O space and are treated as a functional unit. A list of these registers is presented in Table 16.

Although a named register may be 8 bits, write operations performed on the IA188EM should be 8-bit writes, resulting in 16-bit data transfers to the Peripheral Control Block (PCB) register. Only word reads should be performed to the PCB registers. If unaligned read and write accesses are performed on either the IA186EM or IA188EM, indeterminate behavior may result.

| Note: Adhere to these directions while writing code to avoid errors. |
|---|

## Table 16. Peripheral Control Registers

| Register Name | Offset |
|---|---|
| **Peripheral Control Block Registers** | |
| PCB Relocation Register | FEh |
| Reset Configuration Register | F6h |
| Processor Release Level Register | F4h |
| Power-Save Control Register | F0h |
| Enable RCU Register | E4h |
| Clock Prescaler Register | E2h |
| Memory Partition Register | E0h |
| **DMA Registers** | |
| DMA1 Control Register | DAh |
| DMA1 Transfer Count Register | D8h |
| DMA1 Destination Address High Register | D6h |
| DMA1 Destination Address Low Register | D4h |
| DMA1 Source Address High Register | D2h |
| DMA1 Source Address Low Register | D0h |
| DMA0 Control Register | CAh |
| DMA0 Transfer Count Register | C8h |
| DMA0 Destination Address High Register | C6h |
| DMA0 Destination Address Low Register | C4h |
| DMA0 Source Address High Register | C2h |
| DMA0 Source Address Low Register | C0h |
| **Chip-Select Registers** | |
| pcs_n and mcs_n Auxiliary Register | A8h |
| Mid-Range Memory Chip-Select Register | A6h |
| Peripheral Chip-Select Register | A4h |
| Low-Memory Chip-Select Register | A2h |
| Upper-Memory Chip-Select Register | A0h |
| **Asynchronous Serial Port Register** | |
| Serial Port Baud Rate Divisor Register | 88h |
| Serial Port Receive Register | 86h |
| Serial Port Transmit Register | 84h |
| Serial Port Status Register | 82h |
| Serial Port Control Register | 80h |
| **PIO Registers** | |
| PIO Data 1 Register | 7Ah |
| PIO Direction 1 Register | 78h |
| PIO Mode 1 Register | 76h |
| PIO Data 0 Register | 74h |
| PIO Direction 0 Register | 72h |
| PIO Mode 0 Register | 70h |

| Register Name | Offset |
|---|---|
| **Timer Registers** | |
| Timer 2 Mode and Control Register | 66h |
| Timer 2 Max Count Compare A Register | 62h |
| Timer 2 Count Register | 60h |
| Timer 1 Mode and Control Register | 5Eh |
| Timer 1 Max Count Compare B Register | 5Ch |
| Timer 1 Max Count Compare A Register | 5Ah |
| Timer 1 Count Register | 58h |
| Timer 0 Mode and Control Register | 56h |
| Timer 0 Max Count Compare B Register | 54h |
| Timer 0 Max Count Compare A Register | 52h |
| Timer 0 Count Register | 50h |
| **Interrupt Registers** | |
| Serial Port 0 Interrupt Control Register | 44h |
| Watchdog Timer Control Register | 42h |
| INT4 Interrupt Control Register | 40h |
| INT3 Interrupt Control Register | 3Eh |
| INT2 Interrupt Control Register | 3Ch |
| INT1 Interrupt Control Register | 3Ah |
| INT0 Interrupt Control Register | 38h |
| DMA1 Interrupt Control Register | 36h |
| DMA0 Interrupt Control Register | 34h |
| Timer Interrupt Control Register | 32h |
| Interrupt Status Register | 30h |
| Interrupt Request Register | 2Eh |
| Interrupt In-Service Register | 2Ch |
| Interrupt Priority Mask Register | 2Ah |
| Interrupt Mask Register | 28h |
| Interrupt Poll Status Register | 26h |
| Interrupt Poll Register | 24h |
| End-of-Interrupt (EOI) Register | 22h |
| Interrupt Vector Register | 20h |
| **Serial Port 1 Registers** | |
| Synchronous Serial Receive Register | 18h |
| Synchronous Serial Transmit 0 Register | 16h |
| Synchronous Serial Transmit 1 Register | 14h |
| Synchronous Serial Enable Register | 12h |
| Synchronous Serial Status Register | 10h |

### 5.1.1 RELREG (0feh)

The Peripheral Control Block RELocation REGister maps the entire Peripheral Control Block Register Bank to either I/O or memory space. In addition, RELREG contains a bit that places the interrupt controller in either master or slave mode. The RELREG contains 20ffh at reset (see Table 17).

**Table 17. Peripheral Control Block Relocation Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | S/Mn | Reserved | IO/Mn | RA19– RA8 | | | | | | | | | | | |

- Bit [15]—Reserved.
- Bit [14]—S/Mn → When set to 1, this bit places the interrupt controller into slave mode. When 0, it is in master mode.
- Bit [13]—Reserved.
- Bit [12]—IO/Mn → When set to 1, the Peripheral Control Block is mapped into memory space. When 0, this bit maps the Peripheral Control Block Register Bank into IO space.
- Bits [11–0]—RA19–RA8 → Sets the base address (upper 12 bits) of the Peripheral Control Block Register Bank. RA7–RA0 default to 0. When Bit [12] (IO/Mn) is set to 1, RA19–RA16 are ignored.

### 5.1.2 RESCON (0f6h)

The RESet CONfiguration Register latches user-defined information present at specified pins at the rising edge of reset. The contents of this register are read-only and remain valid until the next reset. The RESCON contains user-defined information at reset (see Table 18).

**Table 18. Reset Configuration Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RC15–RC0 | | | | | | | | | | | | | | | |

- Bits [15–0]—RC15–RC0 → At the rising edge of reset, the values of specified pins (ad15–ad0 for the IA186EM and ao15–ao8 and ad7–ad0 for the IA188EM) are latched into this register.

### 5.1.3 PRL (0f4h)

The Processor Release Level Register contains a code corresponding to the latest processor production release. The PRL is a Read-Only Register. The PRL contains 0400h (see Table 19).

**Table 19. Processor Release Level Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PRL7–PRL0 | | | | | | | | Reserved | | | | | | | |

- Bits [15–8]—PRL7–PRL0 → The latest Processor Release Level.

| PRL Value | Processor Release Level |
|---|---|
| 01h | C |
| 02h | D |
| 03h | E |
| 04h | F |

- Bits [7–0]—Reserved.

### 5.1.4 PDCON (0f0h)

The Power-save CONtrol Register controls several miscellaneous system I/O and timing functions. The PDCON contains 0000h at reset (see Table 20).

**Table 20. Power-Save Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PSEN | Reserved | | | CBF | CBD | CAF | CAD | Reserved | | | | | F2 | F1 | F0 |

- Bit [15]—PSEN → When set to 1, enables the power-save mode causing the internal operating clock to be divided by the value in F2–F0. External interrupts or interrupts from internal interrupts automatically clear PSEN. Software interrupts and exception do not clear PSEN.

  Note: The value of PSEN is not restored upon execution of an IRET instruction.

- Bits [14–12]—Reserved → These bits read back as 0.

- Bit [11]—CBF → When set to 1, the clkoutb output follows the input crystal (PLL) frequency. When 0, it follows the internal clock frequency after the clock divider.

- Bit [10]—CBD → When set to 1, the clkoutb output is pulled low. When 0, it is driven as an output per the CBF bit.

- Bit [9]—CAF → When set to 1, the clkouta output follows the input crystal (PLL) frequency. When 0, it follows the internal clock frequency after the clock divider.

- Bit [8]—CAD → When set to 1, the clkouta output is pulled low. When 0, it is driven as an output per the CBF bit.
- Bits [7–3]—Reserved → These bits read back as 0.
- Bits [2–0]—F2–F0 → These bits control the clock divider as shown below.

Note: PSEN must be 1 for the clock divider to function.

| F2 | F1 | F0 | Divider Factor |
|---|---|---|---|
| 0 | 0 | 0 | Divide by 1 ($2^0$) |
| 0 | 0 | 1 | Divide by 2 ($2^1$) |
| 0 | 1 | 0 | Divide by 4 ($2^2$) |
| 0 | 1 | 1 | Divide by 8 ($2^3$) |
| 1 | 0 | 0 | Divide by 16 ($2^4$) |
| 1 | 0 | 1 | Divide by 32 ($2^5$) |
| 1 | 1 | 0 | Divide by 64 ($2^6$) |
| 1 | 1 | 1 | Divide by 128 ($2^7$) |

### 5.1.5 EDRAM (0e4h)

The Enable RCU Register provides control and status for the refresh counter. The EDRAM register contains 0000h at reset (see Table 21).

**Table 21. Enable Dynamic RAM Refresh Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E | Reserved | | | | | | T8–T0 | | | | | | | | |

- Bit [15]—E → When set to 1, the refresh counter is enabled and msc3_n is configured to act as rfsh_n. Clearing E empties the refresh counter and disables refresh requests. The refresh address is unaffected by clearing E.
- Bits [14–9]—Reserved → These bits read back as 0.
- Bits [8–0]—T8–T0 → These bits hold the current value of the refresh counter. They are read-only.

### 5.1.6 CDRAM (0e2h)

The Clock Prescaler Register determines the period between refresh cycles. The Count for Dynamic RAM (CDRAM) register is undefined at reset (see Table 22).

**Table 22. Count for Dynamic RAM Refresh Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | RC8–RC0 | | | | | | | | |

- Bits [15–9]—Reserved → These bits read back as 0.
- Bits [8–0]—RC8–RC0 → These bits hold the clock count interval between refresh cycles. In power-save mode, the refresh counter value should be adjusted to account for the clock divider value in PDCON.

Note: This value should not be set to less than 18 (12h), else there would never be sufficient bus cycles available for the processor to execute code.

### 5.1.7 MDRAM (0e0h)

The Memory Partition Register holds the a19–a13 address bits of the 20-bit base refresh address. The MDRAM register contains 0000h at reset (see Table 23).

**Table 23. Memory Partition for Dynamic RAM Refresh Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M6–M0 | | | | | | | Reserved | | | | | | | | |

- Bits [15–9]—M6–M0 → Upper bits corresponding to address bits a19–a13 of the 20-bit memory refresh address. These bits are not available on the a19–a0 bus. When using PSRAM mode, M6–M0 must be programmed to 0000000b.
- Bits [8–0]—Reserved → These bits read back as 0.

### 5.1.8 D1CON (0dah) and D0CON (0cah)

DMA CONtrol Registers. DMA Control Registers control operation of the two DMA channels. The D0CON and D1CON registers are undefined at reset, except ST which is set to 0 (see Table 24).

**Table 24. DMA Control Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DM/IOn | DDEC | DINC | SM/IOn | SDEC | SINC | TC | INT | SYN1–SYN0 | | P | TDRQ | Res | CHG | ST | Bn/W |

- Bit [15]—DM/IOn → Destination Address Space Select selects memory or I/O space for the destination address. When DM/IO is set to 1, the destination address is in memory space. When 0, it is in I/O space.

- Bit [14]—DDEC → Destination Decrement. When set to 1, it automatically decrements the destination address after each transfer. The address is decremented by 1 or 2, depending on the byte/word bit (Bn/W, Bit [0]). The address does not change if the increment and decrement bits are set to the same value (00b or 11b).

- Bit [13]—DINC → Destination Increment. When set to 1, it automatically increments the destination address after each transfer. The address is incremented by 1 or 2, depending on the byte/word bit (Bn/W, Bit [0]). The address does not change if the increment and decrement bits are set to the same value (00b or 11b).

- Bit [12]—SM/IOn → Source Address Space Select selects memory or I/O space for the source address. When set to 1, the source address is in memory space. When 0, it is in I/O space.

- Bit [11]—SDEC → Source Decrement. When set to 1, it automatically decrements the destination address after each transfer. The address is decremented by 1 or 2, depending on the byte/word bit (Bn/W, Bit [0]). The address does not change if the increment and decrement bits are set to the same value (00b or 11b).

- Bit [10]—SINC → Source Increment. When set to 1, it automatically increments the destination address after each transfer. The address is incremented by 1 or 2, depending on the byte/word bit (Bn/W, Bit [0]). The address does not change if the increment and decrement bits are set to the same value (00b or 11b).

- Bit [9]—TC → Terminal Count. The DMA decrements the transfer count for each DMA transfer. When set to 1, the source or destination synchronized DMA transfers terminate when the count reaches 0. When 0, they do not. Unsynchronized DMA transfers always end when the count reaches 0, regardless of this bit's setting.

- Bit [8]—INT → Interrupt. When this bit is set to 1, the DMA channel generates an interrupt request on completion of the transfer count. However, for an interrupt to be generated, the TC bit must also be set to 1.

- Bits [7–6]—SYN1–SYN0 → Synchronization Type bits each select channel synchronization types as shown below. The value of these bits is ignored if TDRQ (Bit [4]) is set to 1. A processor reset causes these bits to be set to 11b.

Synchronization Bit Channel Selection

| SYN1 | SYN0 | Sync Type |
|---|---|---|
| 0 | 0 | Unsynchronized |
| 0 | 1 | Source Synchronized |
| 1 | 0 | Destination Synchronized |
| 1 | 1 | Reserved |

- Bit [5]—P → Relative Priority. When set to 1, selects high priority for this channel relative to the other channel during simultaneous transfers.
- Bit [4]—TDRQ → Timer 2 Synchronization. When set to 1, enables DMA requests from Timer 2. When 0, disables them.
- Bit [3]—Reserved.
- Bit [2]—CHG → Change Start Bit. This bit must be set to 1 to allow modification of the ST bit during a write. During a write, when CHG is set to 0, ST is not changed when writing the control word. The result of reading this bit is always 0.
- Bit [1]—ST → Start/Stop DMA Channel. When set to 1, the DMA channel is started. The CHG bit must be set to 1 for this bit to be modified and only during the same register write. A processor reset causes this bit to be set to 0.
- Bit [0]—Bn/W → Byte/Word Select. When set to 1, word transfers are selected. When 0, byte transfers are selected.

Note: Word transfers are not supported if the chip selects are programmed for 8-bit transfers. The IA188EM does not support word transfers

### 5.1.9 D1TC (0d8h) and D0TC (0c8h)

DMA Transfer Count Registers. The DMA Transfer Count registers are maintained by each DMA channel. They are decremented after each DMA cycle. The state of the TC bit in the DMA control register has no influence on this activity. But, if unsynchronized transfers are programmed or if the TC bit in the DMA control word is set, DMA activity ceases when the transfer count register reaches 0. The D0TC and D1TC registers are undefined at reset (see Table 25).

**Table 25. DMA Transfer Count Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TC15–TC0 | | | | | | | | | | | | | | | |

- Bits [15–0]—TC15–TC0 → DMA Transfer Count contains the transfer count for the respective DMA channel. Its value is decremented after each transfer.

### 5.1.10 D1DSTH (0d6h) and D0DSTH (0c6h)

The DMA DeSTination Address High Register. The 20-bit destination address consists of these 4 bits combined with the 16 bits of the respective Destination Address Low Register. A DMA transfer requires that two complete 16-bit registers (high and low registers) be used for both the source and destination addresses of each DMA channel involved. These four registers must be

initialized. Each address may be incremented or decremented independently of each other after each transfer. The addresses are incremented or decremented by two for word transfers and incremented or decremented by one for byte transfers. They are undefined at reset (see Table 26).

**Table 26. DMA Destination Address High Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | DDA19–DDA16 | | | |

- Bits [15–4]—Reserved.
- Bits [3–0]—DDA19–DDA16 → DMA Destination Address High bits are driven onto a19–a16 during the write phase of a DMA transfer.

### 5.1.11 DIDSTL (0d4h) and D0DSTL (0c4h)

DMA DeSTination Address Low Register. The 16 bits of these registers are combined with the 4 bits of the respective DMA Destination Address High Register to produce a 20-bit destination address. They are undefined at reset (see Table 27).

**Table 27. DMA Destination Address Low Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DDA15–DDA0 | | | | | | | | | | | | | | | |

- Bits [15–0]—DDA15–DDA0 → DMA Destination Address Low bits are driven onto a15–a0 during the write phase of a DMA transfer.

### 5.1.12 D1SRCH (0d2h) and D0SRCH (0c2h)

DMA SouRCe Address High Register. The 20-bit source address consists of these 4 bits combined with the 16 bits of the respective Source Address Low Register. A DMA transfer requires that two complete 16-bit registers in the PCB (high and low registers) be used for both the source and destination addresses of each DMA channel involved. Each channel requires that all four address registers be initialized. Each address may be independently incremented or decremented after each word transfer by 2 or by 1 for byte transfers. They are undefined at reset (see Table 28).

**Table 28. DMA Source Address High Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | DSA19–DSA16 | | | |

- Bits [15–4]—Reserved.
- Bits [3–0]—DSA19–DSA16 → DMA Source Address High bits are driven onto a19–a16 during the read phase of a DMA transfer.

### 5.1.13 D1SRCL (0d0h) and D0SRCL (0c0h)

DMA SouRCe Address Low Register. The 16 bits of these registers are combined with the 4 bits of the respective DMA Source Address High register to produce a 20-bit source address. They are undefined at reset (see Table 29).

**Table 29. DMA Source Address Low Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSA15–DSA0 | | | | | | | | | | | | | | | |

- Bits [15–0]—DSA15–DSA0 → DMA Source Address Low bits are placed onto a15–a0 during the read phase of a DMA transfer.

### 5.1.14 MPCS (0a8h)

MCS and PCS (MPCS) Auxiliary Register. Because this register controls more than one type of chip select, it is unlike other chip select control registers. The MPCS register contains information for mcs3_n–mcs0_n, pcs6_n–pcs5_n, and pcs3_n–pcs0_n.

The MPCS register also contains a bit that configures the pcs6_n–pcs5_n pins as either chip selects or as alternate sources for the a2 and a1 address bits. Either a1/a2 or pcs6_n–pcs5_n are selected to the exclusion of the other. When programmed for address bits, these outputs can be used to provide latched address bits for a2 and a1.

The pcs6_n–pcs5_n pins are high and not active on processor reset. When the pcs6_n–pcs5_n are configured as address pins, an access to the MPCS register causes them to activate. They do not require corresponding access to the PACS register to be activated. The value of the MPCS register is undefined at reset (see Table 30).

**Table 30. MCS and PCS Auxiliary Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | M6–M0 | | | | | | | EX | MS | Reserved | | | R2 | R1–R0 | |

- Bit [15]—Reserved → Set to 1.
- Bits [14–8]—M6–M0 mcs_n Block Size → These seven bits determine the total memory block size for the mcs3_n–mcs0_n chip selects. The size is divided equally among them. The relationship between M6–M0 and the size is shown below.

Select Sizes of M6–M0 by Total Block Size

| Total Block Size | Individual Select Size | M6–M0 |
|---|---|---|
| 8K | 2K | 0000001b |
| 16K | 4K | 0000010b |
| 32K | 8K | 0000100b |
| 64K | 16K | 0001000b |
| 128K | 32K | 0010000b |
| 256K | 64K | 0100000b |
| 512K | 128K | 1000000b |

- Bit [7]—EX Pin Selector → This bit determines whether the pcs6_n–pcs5_n pins are configured as chip selects or as alternate outputs for a2 and a1. When set to 1, they are configured as peripheral chip select pins. When 0, they become address bits a1 and a2, respectively.

- Bit [6]—MS Memory/I/O Space Selector → This bit determines whether the pcs_n pins are active during either memory or I/O bus cycles. When set to 1, the outputs are active for memory bus cycles. When 0, they are active for I/O bus cycles.

- Bits [5–3]—Reserved → Set to 1.

- Bit [2]—R2 Ready Mode → This bit influences only the pcs6_n–pcs5_n chip selects. When set to 1, external ready is ignored. When 0, it is required. Values determine the number of wait states to be inserted.

- Bits [1–0]—R1–R0 Wait-State Value → These bits influence only the pcs6_n–pcs5_n chip selects. Their value determines the number of wait states inserted into an access, depending on whether it is to the pcs_n memory or I/O area. Up to three wait states can be inserted (R1–R0 = 00b to 11b).

### 5.1.15 MMCS (0a6h)

Midrange Memory Chip Select (MMCS) Register. Four chip-select pins, mcs3_n–mcs0_n, are provided for use within a user-locatable memory block. Excluding the areas associated with the ucs_n and lcs_n chip selects (and if mapped to memory, the address range of the peripheral chip selects, pcs6_n–pcs5_n and pcs3_n–pcs0_n), the memory block base address can be located anywhere within the 1-Mbyte memory address space. If the pcs_n chip selects are mapped to I/O space, the mcs_n address range can overlap the pcs_n address range.

Two registers program the Midrange Chip Selects. The MMCS register determines the base address, the ready condition, and wait states of the memory block that are accessed through the mcs_n pins. The pcs_n and mcs_n auxiliary (MPCS) register configures the block size. On reset, the mcs3_n–mcs0_n pins are not active. Accessing with a write, both the MMCS and MPCS registers activate these chip selects.

Unlike the ucs_n and lcs_n chip selects, the mcs3_n–mcs0_n outputs assert with the multiplexed ad address bus (ad15–ad0 for the IA186EM and ao15–ao8 and ad7–ad0 for the IA188EM), rather than the earlier timing of the a19–a0 bus. If the a19–a0 bus is used for address selection, the timing is delayed for a half cycle later than that for ucs_n and lcs_n. The value is undefined at reset (see Table 31).

**Table 31. Midrange Memory Chip Select Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BA19–BA13 | | | | | | | Reserved | | | | | | R2 | R1–R0 | |

- Bits [15–9]—BA19–BA13 Base Address → The value of the this pin determines the base address of the memory block that is addressed by the mcs_n chip select pins. These bits correspond to a19–a13 of the 20-bit memory address. The remaining bits a12–a0 of the base address are always 0.
  - The base address may be any integer multiple of the size of the memory clock selected in the MPCS register. For example, if the midrange block is 32 Kbytes, the block could be located at 20000h or 28000h but not at 24000h.
  - If the lcs_n chip select is inactive, the base address of the midrange chip selects can be set to 00000h, because the lcs_n chip select is defined to be 00000h but is unused. Because the base address must be an integer multiple of the block size, a 512K MMCS block size can only be used with the lcs_n chip select inactive and the base address of the midrange chip selects set to 00000h.
- Bits [8–3]—Reserved → Set to 1.
- Bit [2]—R2 Ready mode → This bit determines the mcs_n chip select ready mode. When set to 1, an external ready is ignored. When 0, it is necessary. In each case, the number of wait states inserted in an access is determined by the value of the R1 and R0 bits.
- Bits [1–0]—R1–R0 → Wait-State Value. The value of these bits determines the number of wait states inserted in an access. Up to three wait states can be inserted (R1–R0 = 00b to 11b).

### 5.1.16 PACS (0a4h)

PeripherAl Chip Select Register. These Peripheral Chip Selects are asserted over a 256-byte range with the same timing as the ad address bus. There are six chip selects, pcs6_n–pcs5_n and pcs3_n–pcs0_n, that are used in either the user-locatable memory or I/O blocks. The pcs4_n chip select is not implemented in the IA186EM or IA188EM. Excluding the areas used by the ucs_n, lcs_n, and mcs_n chip selects, the memory block can be located anywhere within the

1-Mbyte address space. These chip selects may also be configured to access the 64-Kbyte I/O space.

Programming the Peripheral Chip Selects uses the Peripheral Chip Select (PACS) and the pcs_n and mcs_n Auxiliary (MPCS) registers. The PACS register establishes the base address, configures the ready mode, and determines the number of wait states for the pcs3_n–pcs0_n outputs.

The MPCS register configures the pcs6_n–pcs5_n pins to be either chip selects or address pins a1 and a2. When these pins are configured as chip selects, the MPCS register determines the ready and wait states for these output pins and whether they are active during memory or I/O bus cycles. These pins are activated as chip selects by writing to the two registers (PACS and MPCS). They are not active on reset. To configure and activate them as address pins, it is necessary to write to both the PACS and MPCS registers. Pins pcs6_n–pcs5_n can be configured for 0 to 3 wait states and pcs3_n–pcs0_n can be programmed for 0 to 15 wait states. The value of the PACS register is undefined at reset (see Table 32).

**Table 32. Peripheral Chip Select Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BA19–BA11 | | | | | | | | | Reserved | | | R3 | R2 | R1–R0 | |

- Bits [15–7]—BA19–BA11 → Base Address bits correspond to Bits [19–11] of the 20-bit programmable base address of the peripheral chip select block and determine the base address. Because I/O addresses are only 16 bits wide, if the pcs_n chip selects are mapped to I/O space, these bits must be set to 0000b. The pcs address ranges are shown below.

  Address Ranges of pcs Chip Selects

| | Range | |
|---|---|---|
| pcs_n Line | Low | High |
| pcs0_n | Base Address | Base Address + 255 |
| pcs1_n | Base Address + 256 | Base Address + 511 |
| pcs2_n | Base Address + 512 | Base Address + 767 |
| pcs3_n | Base Address + 768 | Base Address + 1023 |
| Reserved | NA | NA |
| pcs5_n | Base Address + 1280 | Base Address |
| pcs6_n | Base Address + 1536 | Base Address |

- Bits [6–4]—Reserved → Set to 1.
- Bit [3]—R3 → Wait State Value. See pcs3_n–pcs0_n Wait-State Encoding shown below.

pcs3_n–pcs0_n Wait-State Encoding

| R3 | R1 | R0 | Wait States |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 2 |
| 0 | 1 | 1 | 3 |
| 1 | 0 | 0 | 5 |
| 1 | 0 | 1 | 7 |
| 1 | 1 | 0 | 9 |
| 1 | 1 | 1 | 15 |

- Bit [2]—R2 → Ready Mode. When set to 1, external ready is ignored. When 0, it is required. In each case the number of wait states is determined according to the pcs3_n–pcs0_n Wait-State Encoding shown above.

- Bits [1–0]—R1–R0 → Wait-State Value (see pcs3_n–pcs0_n Wait-State Encoding above). The pcs6_n–pcs5_n and pcs3_n–pcs0_n pins are multiplexed with the PIO pins. For these to function as chip selects, the PIO mode and direction settings for these pins must be set to 0 for normal operation.

### 5.1.17 LMCS (0a2h)

The Low-Memory Chip Select (LMCS) Register configures the LMCS provided to facilitate access to the interrupt vector table located at 00000h or the bottom of memory. The lcs_n pin is not active at reset.

The width of the data bus for the lcs_n space should be configured in the AUXCON register before activating the lcs_n chip select pin, by any write access to the LMCS register. The value of the LMCS register is undefined at reset except DA, which is set to 0 (see Table 33).

**Table 33. Low-Memory Chip Select Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Res | UB2–UB0 | | | Reserved | | | | DA | PSE | Reserved | | | R2 | R1–R0 | |

- Bit [15]—Reserved → Set to 0.

- Bits [14–12]—UB2–UB0 → Upper Boundary. These bits define the upper boundary of memory accessed by the lcs_n chip select. The list below presents the possible block-size configurations (a 512-Kbyte maximum).

LMCS Block-Size Programming Values

| Memory Block Size | Ending Address | UB2–UB0 |
|---|---|---|
| 64K | 0FFFFh | 000b |
| 128K | 1FFFFh | 001b |
| 256K | 3FFFFh | 011b |
| 512K | 7FFFFh | 111b |

- Bits [11–8]—Reserved → Set to 1.
- Bit [7]—DA → Disable Address. When set to 1, the address bus is disabled, providing some measure of power saving. When 0, the address is driven onto the address bus ad15–ad0 during the address phase of a bus cycle. This bit is set to 0 at reset.
  - If bhe_n/aden_n (IA186EM) is held at 0 during the rising edge of res_n, the address bus is always driven, regardless of the setting of DA.
- Bit [6]—PSE → PSRAM Mode Enable. When set to 1, PSRAM support for the lcs_n chip select memory space is enabled. The EDRAM, MDRAM, and CDRAM RCU registers must be configured for auto refresh before PSRAM support is enabled. Setting the enable bit (EN) in the enable RCU register (EDRAM, offset e4h) configures the mcs3_n/rfsh_n as rfsh_n.
- Bits [5–3]—Reserved → Set to 1.
- Bit [2]—R2 → Ready Mode. When set to 1, the external ready is ignored. When 0, it is required. The value of R1–R0 bits determines the number of wait states inserted.
- Bits [1–0]—R1–R0 → Wait-State Value. The value of these bits determines the number of wait states inserted into an access to the lcs_n memory area. This number ranges from 0 to 3 (R1–R0 = 00b to 11b).

### 5.1.18 UMCS (0a0h)

The Upper Memory Chip Select Register configures the UMCS pin, used for the top of memory. On reset, the first fetch takes place at memory location FFFF0h and thus this area of memory is usually used for instruction memory. The ucs_n defaults to an active state at reset with a memory range of 64 Kbytes (F0000h to FFFFFh), external ready required, and three wait states automatically inserted. The upper end of the memory range always ends at FFFFFh. The lower end of this upper memory range is programmable. The value of the UMCS register is F03Bh at reset (see Table 34).

**Table 34. Upper-Memory Chip Select Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | LB2–LB0 | | | Reserved | | | | DA | 0 | Reserved | | | R2 | R1–R0 | |

- Bit [15]—Reserved → Set to 1.
- Bits [14–12]—LB2–LB0 Lower Boundary → These bits determine the bottom of the memory accessed by the ucs_n chip selects. The UMCS Block-Size Programming Values shown below list the possible block-size configurations (a 512-Kbyte maximum).

  UMCS Block-Size Programming Values

| Memory Block Size | Starting Address | LB2–LB0 | Comments |
|---|---|---|---|
| 64K | F0000h | 111b | Default |
| 128K | E0000h | 110b | – |
| 256K | C0000h | 100b | – |
| 512K | 80000h | 000b | – |

- Bits [11–8]—Reserved.
- Bit [7]—DA → Disable Address. When set to 1, the address bus is disabled and the address is not driven on the address bus when ucs_n is asserted, providing some measure of power saving. When 0, the address is driven onto the address bus (ad15–ad0) during the address phase of a bus cycle when ucs_n is asserted. This bit is set to 0 at reset.
  - If bhe_n/aden_n (IA186EM) is held at 0 during the rising edge of res_n, the address bus is always driven, regardless of the setting.
- Bit [6]—Reserved → Set to 0.
- Bits [5–3]—Reserved → Set to 1.
- Bit [2]—R2 Ready Mode → When set to 1, the external ready is ignored. When 0, it is required. The value of the R1–R0 bits determines the number of wait states inserted.
- Bits [1–0]—R1–R0 Wait-State Value → The value of these bits determines the number of wait states inserted into an access to the lcs_n memory area. This number ranges from 0 to 3 (R1–R0 = 00b to 11b).

### 5.1.19 SPBAUD (088h)

Serial Port BAUD Rate Divisor Register. The value in this register determines the number of internal processor cycles in one phase (half-period) of the 32 x serial clock. The contents of

these registers must be adjusted to reflect the new processor clock frequency if power-save mode is in effect. The baud rate divisor may be calculated from:

BAUDDIV = (Processor Frequency/(32 x baud rate)) -1 (Equation 1)

By setting the BAUDDIV to 0000h, the maximum baud rate of 1/32 of the internal processor frequency clock is set. Setting BAUDDIV to 129 (81h) provides a baud rate of 9600 at 40 MHz. The baud rate tolerance is +4.6% to –1.9% with respect to the actual serial port baud rate, not the target baud rate (see Table 35).

**Table 35. Baud Rates**

| Baud Rate | Divisor Based on CPU Clock Rate | | | |
|---|---|---|---|---|
| | 20 MHz | 25 MHz | 33 MHz | 40 MHz |
| 300 | 4166 | 5208 | 6875 | 8333 |
| 600 | 2083 | 2604 | 3437 | 4166 |
| 1200 | 1041 | 1302 | 1718 | 2083 |
| 2400 | 520 | 651 | 859 | 1041 |
| 4800 | 260 | 325 | 429 | 520 |
| 9600 | 130 | 162 | 214 | 260 |
| 14400 | 42 | 53 | 71 | 85 |
| 19200 | 31 | 39 | 53 | 64 |
| 625 Kbaud | 0 | NA | NA | 1 |
| 781.25 Kbaud | NA | 0 | NA | NA |
| 1.041 Mbaud | NA | NA | 0 | NA |
| 1.25 Mbaud | NA | NA | NA | 0 |

The value of the SPBAUD register at reset is undefined (see Table 36).

**Table 36. Serial Port Baud Rate Divisor Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BAUDDIV | | | | | | | | | | | | | | | |

- Bits [15–0]—BAUDDIV Baud Rate Divisor → Defines the divisor for the internal processor clock.

### 5.1.20 SPRD (086h)

Serial Port Receive Data Register. Data received over the serial port are stored in this register until read. The data are received initially by the receive shift register (no software access) permitting data to be received while the previous data are being read.

The RDR bit (Receive Data Ready) in the serial port status register indicates the status of the SPRD register. Setting the RDR bit to 1 indicates there is valid data in the receive register. The value of the SPRD register is undefined at reset (see Table 37).

**Table 37. Serial Port Receive Data Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | RDATA | | | | | | | |

- Bits [15–8]—Reserved.
- Bits [7–0]—RDATA → Holds valid data while the RDR bit of the status register is set.

### 5.1.21 SPTD (084h)

Serial Port Transmit Data Register. Data is written to this register by software, with the values to be transmitted by the serial port. Double buffering of the transmitter allows for the transmission of data from the transmit shift register (no software access) while the next data are written into the transmit register.

The THRE bit in the Serial Port Status register indicates whether there is valid data in the SPDT register. The THRE bit must be a 1 before writing data to this register to prevent overwriting valid data that is already in the SPDT register. The value of the SPTD register is undefined at reset (see Table 38).

**Table 38. Serial Port Transmit Data Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | TDATA | | | | | | | |

- Bits [15–8]—Reserved.
- Bits [7–0]—TDATA → Holds the data to be transmitted.

### 5.1.22 SPSTS (082h)

Serial Port STatuS Register. This register stores information concerning the current status of the port. The status bits are described below.

The value of the SPSTS register is undefined at reset (see Table 39).

**Table 39. Serial Port Status Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | TEMT | THRE | RDR | BRKI | FER | PER | OER |

- Bits [15–7]—Reserved → Set to 0.
- Bit [6]—TEMT Transmitter Empty → When both the transmit shift register and the transmit register are empty, this bit is set indicating to software that it is safe to disable the transmitter. This bit is read-only.

- Bit [5]—THRE Transmit Holding Register Empty → When this bit is 1, the corresponding transmit holding register is ready to accept data. This is a read-only bit.
- Bit [4]—RDR Receive Data Ready → When this bit is 1, the respective SPRD register contains valid data. This is a read/write bit and can be reset only by reading the corresponding receive register.
- Bit [3]—BRKI Break Interrupt → This bit indicates that a break has been received when this bit is set to 1 and causes a serial port interrupt request.

  Note: This bit should be reset by software.

- Bit [2]—FER Framing Error Detected → When the receiver samples the rxd line as low when a stop bit is expected (line high) a framing error is generated setting this bit.

  Note: This bit should be reset by software.

- Bit [1]—PER Parity Error Detected → When a parity error is detected in either mode 1 or 3, this bit is set.

  Note: This bit should be reset by software.

- Bit [0]—OER Overrun Error Detected → When new data overwrites valid data in the receive register (because it has not been read) an overrun error is detected setting this bit.

  Note: This bit should be reset by software.

### 5.1.23 SPCT (080h)

Serial Port ConTrol Register. This register controls both transmit and receive parts of the serial port. The value of the SPCT register is 0000h at reset (see Table 40).

**Table 40. Serial Port Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | TXIE | RXIE | LOOP | BRK | BRKVAL | PMODE | | WLGN | STP | TMODE | RSIE | RMODE |

- Bits [15–12]—Reserved → Set to 0.
- Bit [11]—TXIE Transmitter Ready Interrupt Enable → This bit enables the generation of an interrupt request whenever the transmit holding register is empty (THRE Bit [1]). The respective port does not generate interrupts when this bit is 0. Interrupts continue to be generated as long as THRE and the TXIE are 1.

- Bit [10]—RXIE Receive Data Ready Interrupt Enable → This bit enables the generation of an interrupt request whenever the receive register contains valid data (RDR Bit [1]). The respective port does not generate interrupts when this bit is 0. Interrupts continue to be generated as long as RDR and the RXIE are 1.

- Bit [9]—LOOP Loop Back → The serial port is placed into the loop-back mode when this bit is set.

- Bit [8]—BRK Send Break → When this bit is set to 1, the txd pin is driven low, overriding any data that may be in the course of being shifted out of the transmit shift register.

  Note: See the definitions of long and short break in Section 5.1.2, SPSTS (Serial Port Status Register).

- Bit [7]—BRKVAL Break Value → This is the ninth data bit transmitted when in modes 2 and 3. This bit is cleared at each transmitted word and is not buffered. To transmit data with this bit set high, the following procedure is recommended.

  1. The TEMT bit in the serial port status register must go high.
  2. Set the TB8 bit by writing it to the serial port control register.
  3. Write the transmit character to the serial port transmit register.

  – Serial port 0 is a special case. If this bit is 1, the associated pins are used for flow control overriding the Peripheral Chip Select signals. This bit is 0 at reset.

- Bits [6–5]—PMODE Parity Mode → When this bit is set to 1, the txd pin is driven low, overriding any data that may be in the course of being shifted out of the transmit shift register.

  Note: See the definitions of long and short break in Section 5.1.2, SPSTS (Serial Port Status Register).

- Bit [4]—WLGN Word Length → The number of bits transmitted or received in a frame is determined by the value of this bit. When this bit is 1, the number of data bits in a frame is 8. When 0, it is 7. This bit is 0 at reset.

- Bit [3]—STP Stop Bits → This bit specifies the number of stop bits used to indicate the end of a frame. When this bit is 1, the number of stop bits is 2. When 0, it is 1. This bit is 0 at reset.

- Bit [2]—TMODE Transmit Mode → When this bit is 1, the transmit section of the serial port is enabled. When 0, it is disabled.

- Bit [1]—RSIE Receive Status Interrupt Enable → When an exception occurs during data reception, an interrupt request is generated if enabled by this bit (RSIE = 1). Interrupt requests are made for the error conditions listed in the serial port status register (BRK, OER, PER, and FER). This bit is 0 at reset.

- Bit [0]—RMODE Receive Mode → When this bit is 1, the receive section of the serial port is enabled. When 0, it is disabled. This bit is 0 at reset.

### 5.1.24 PDATA1 (07ah) and PDATA0 (074h)

PIO DATA Registers. When a PIO pin is configured as an output, the value in the corresponding PIO data register bit is driven onto the pin. However, if the PIO pin is configured as an input, the value on the pin is put into the corresponding bit of the PIO data register. Table 41 lists the default states for the PIO pins.

**Table 41. PIO Pin Assignments**

| PIO Number | Associated Pin Name | Power-On Reset Status |
|---|---|---|
| 0 | tmrin1 | Input with pull-up |
| 1 | tmrout1 | Input with pull-down |
| 2 | pcs6/a2 | Input with pull-up |
| 3 | pcs5/a1 | Input with pull-up |
| 4 | dt/r_n | Normal operation[a] |
| 5 | den_n/ds_n | Normal operation[a] |
| 6 | srdy | Normal operation[b] |
| 7[c] | a17 | Normal operation[a] |
| 8[c] | a18 | Normal operation[a] |
| 9[c] | a19 | Normal operation[a] |
| 10 | tmrout0 | Input with pull-down |
| 11 | tmrin0 | Input with pull-up |
| 12 | drq0 | Input with pull-up |
| 13 | drq1 | Input with pull-up |
| 14 | mcs0_n | Input with pull-up |
| 15 | mcs1_n | Input with pull-up |
| 16 | pcs0_n | Input with pull-up |
| 17 | pcs1_n | Input with pull-up |
| 18 | pcs2_n | Input with pull-up |
| 19 | pcs3_n | Input with pull-up |
| 20 | sclk | Input with pull-up |
| 21 | sdata | Input with pull-up |
| 22 | sden0 | Input with pull-down |
| 23 | sden1 | Input with pull-down |

**Table 41. PIO Pin Assignments** *(Continued)*

| PIO Number | Associated Pin Name | Power-On Reset Status |
|---|---|---|
| 24 | mcs2_n | Input with pull-up |
| 25 | mcs3_n/rfsh_n | Input with pull-up |
| 26[c,d] | uzi | Input with pull-up |
| 27 | txd | Input with pull-up |
| 28 | rxd | Input with pull-up |
| 29[c,d] | s6/clkdiv2_n | Input with pull-up |
| 30 | int4 | Input with pull-up |
| 31 | int2 | Input with pull-up |

[a]When used as a PIO pin, it is an input with a pull-up option available.
[b]When used as a PIO pin, it is an input with a pull-down option available.
[c]Emulators use these pins and also a15–a0, ad15–ad0 (IA186EM), ale, bhe_n (IA186EM), clkouta, nmi, res_n, and s2_n–s0_n.
[d]If bhe_n/aden_n (IA186EM) or rfsh2_n/aden (IA188EM) is held low during POR, these pins revert to normal operation.

The value of the PDATA registers is undefined at reset (see Tables 42 and 43).

**Table 42. PDATA 0**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDATA15–PDATA0 | | | | | | | | | | | | | | | |

**Table 43. PDATA 1**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDATA31–PDATA16 | | | | | | | | | | | | | | | |

- Bits [15–0]—PDATA15–PDATA0 PIO Data 0 Bits → This register contains the values of the bits that are either driven on, or received from, the corresponding PIO pins. Depending on its configuration, each pin is either an output or an input. The values of these bits correspond to those in the PIO Direction registers and PIO Mode registers.

- Bits [15–0]—PDATA31–PDATA16 PIO Data 1 Bits → This register contains the values of the bits that are either driven on, or received from, the corresponding PIO pins. Depending on its configuration, each pin is either an output or an input. The values of these bits correspond to those in the PIO direction registers and PIO Mode registers

- The PIO pins may be operated as open-drain outputs by:
  - Maintaining the data constant in the appropriate bit of the PIO data register.
  - Writing the value of the data bit into the respective bit position of the PIO Direction register, so that the output is either 0 or disabled depending on the value of the data bit.

### 5.1.25 PDIR1 (078h) and PDIR0 (072h)

PIO DIRection Registers. Each PIO pin is configured as an input or an output by the corresponding bit in the PIO Direction register (see Table 44).

**Table 44. PIO Mode and PIO Direction Settings**

| PIO Mode | PIO Direction | Pin function |
|---|---|---|
| 0 | 0 | Normal operation |
| 0 | 1 | PIO input with pullup/pulldown |
| 1 | 0 | PIO output |
| 1 | 1 | PIO input without pullup/pulldown |

The value of the PDIR0 register is FC0Fh at reset (see Table 45).

**Table 45. PDIR0**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDIR15–PDIR0 | | | | | | | | | | | | | | | |

The value of the PDIR1 register is FFFFh at reset (see Table 46).

**Table 46. PDIR1**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDIR31–PDIR16 | | | | | | | | | | | | | | | |

- Bits [15–0]—PDIR15–PDIR0 PIO Direction 0 Bits → For each bit, if the value is 1, the pin is configured as an input. If 0, as an output. The values of these bits correspond to those in the PIO data registers and PIO mode registers.

- Bits [15–0]—PDIR31–PDIR16 PIO Direction 1 Bits → For each bit, if the value is 1, the pin is configured as an input. If 0, as an output. The values of these bits correspond to those in the PIO Data registers and PIO Mode registers.

### 5.1.26 PIOMODE1 (076h) and PIOMODE0 (070h)

PIO MODE Registers. Each PIO pin is configured as an input or an output by the corresponding bit in the PIO direction register. The bit number of PMODE corresponds to the PIO number (see Table 44, PIO Mode and PIO Direction Settings). The value of the PIOMODE0 register is 0000h at reset (see Table 47).

**Table 47. PIOMODE0**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PMODE15–PMODE0 | | | | | | | | | | | | | | | |

The value of the PIOMODE1 register is 0000h at reset (see Table 48).

**Table 48. PMODE1**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PMODE31–PMODE16 | | | | | | | | | | | | | | | |

- Bits [15–0]—PMODE15–PMODE0 PIO Mode 0 Bits → For each bit, if the value is 1, the pin is configured as an input. If 0, an output. The values of these bits correspond to those in the PIO data registers and PIO Mode registers.

- Bits [15–0]—PMODE31–PMODE16 PIO Mode 1 Bits → For each bit, if the value is 1, the pin is configured as an input. If 0, an output. The values of these bits correspond to those in the PIO data registers and PIO Mode registers.

### 5.1.27 T1CON (05eh) and T0CON (056h)

Timer 0 and Timer 1 Mode and CONtrol Registers. These registers control the operation of Timer 0 and Timer 1, respectively. The value of the T0CON and T1CON registers is 0000h at reset (see Table 49).

**Table 49. Timer 0 and Timer 1 Mode and Control Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EN | INHn | INT | RIU | Reserved | | | | | | MC | RTG | P | EXT | ALT | CONT |

- Bit [15]—EN Enable Bit → The timer is enabled when the EN bit is 1. The timer count is inhibited when the EN bit is 0. This bit is write-only and can only be written if the INHn bit (Bit [14]) is set to 1 in the same operation.

- Bit [14]—INHn Inhibit Bit → Gates the setting of the enable (EN) bit. This bit must be set to 1 in the same write operation that sets the enable (EN) bit. Otherwise, the EN bit will not be changed. This bit always reads 0.

- Bit [13]—INT Interrupt Bit → An interrupt request is generated when the Count register reaches its maximum, MC = 1, by setting the INT bit to 1. In dual maxcount mode, an interrupt request is generated when the count register reaches the value in Maxcount A or Maxcount B. No interrupt requests are generated if this bit is set to 0. If an interrupt request is generated, and the enable bit is then cleared before the interrupt is serviced, the interrupt request will remain.

- Bit [12]—RIU Register in Use Bit → This bit is set to 1 when the Maxcount Register B is used to compare to the timer-count value. It is 0 when the Maxcount Compare A register is used.

- Bits [11–6]—Reserved → Set to 0.
- Bit [5]—MC Maximum Count → When the timer reaches its maximum count, this bit is set to 1 regardless of the interrupt enable bit. This bit is also set every time Maxcount Compare Register A or B is reached when in dual maxcount mode. If preferred, this bit may be used by software polling rather than by interrupts to monitor timer status.
- Bit [4]—RTG Retrigger Bit → This pin controls the timer function of the timer input pin. When set to 1, the count is reset by a 0 to 1 transition on timrin0 or tmrin1. When 0, a high input on tmrin0 or tmrin1 enables the count and a 1 holds the timer value. This bit is ignored if the external clocking (EXT = 1) bit is set.
- Bit [3]—P Prescaler Bit → P is ignored if external clocking is enabled (EXT = 1). Timer 2 prescales the timer when P is set to 1. Otherwise, the timer is incremented on every fourth clkout cycle.
- Bit [2]—EXT External Clock Bit → This bit determines whether an external or internal clock is used. If EXT is 1, an external clock is used. If 0, an internal is used.
- Bit [1]—ALT Alternate Compare Bit → If set to 1, the timer will count to Maxcount Compare A, reset the count register to 0, count to Maxcount Compare B, reset the count register to 0, and begin again at Maxcount Compare A. If 0, it will count to Maxcount Compare A, reset the count register to 0, and begin again at Maxcount Compare A. Maxcount Compare B is not used in this case.
- Bit [0]—CONT Continuous Mode Bit → When set to 1, the timer runs continuously. When 0, the timer stops after each count run and EN will be cleared. If CONT = 1 and ALT = 1, the respective timer counts to the Maxcount Compare A value and resets, then commences counting to Maxcount Compare B value, resets, and stops counting.

### 5.1.28 T2CON (066h)

Timer 2 Mode and CONtrol Register. This register controls the operation of Timer 2. The value of the T2CON register is 0000h at reset (see Table 50).

**Table 50. Timer 2 Mode and Control Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EN | INHn | INT | Reserved | | | | | | | MC | Reserved | | | | CONT |

- Bit [15]—EN Enable Bit → The timer is enabled when the EN bit is 1. The timer count is inhibited when the EN bit is 0. Setting this bit to 1 by writing to the T2CON register requires that the INH bit be set to 1 during the same write. This bit is write-only and can only be written if the INHn bit (Bit [14]) is set to 1 in the same operation.

- Bit [14]—INHn Inhibit Bit → Gates the setting of the enable (EN) bit. This bit must be set to 1 in the same write operation that sets the enable (EN) bit. This bit always reads 0.
- Bit [13]—INT Interrupt Bit → An interrupt request is generated, by setting the INT bit to 1, when the Count register reaches its maximum, MC = 1.
- Bits [12–6]—Reserved → Set to 0.
- Bit [5]—MC Maximum Count → When the timer reaches its maximum count, this bit is set to 1, regardless of the interrupt enable bit. If preferred, this bit may be used by software polling rather than by interrupts to monitor timer status.
- Bits [4–1]—Reserved → Set to 0.
- Bit [0]—CONT Continuous Mode Bit → The timer will run continuously when this bit is set to 1. The timer will stop after each count run and EN will be cleared if this bit is set to 0.

### 5.1.29 T2COMPA (062h), T1COMPB (05ch), T1COMPA (05ah), T0COMPB (054h), and T0COMPA (052h)

Timer Maxcount COMpare Registers. These registers contain the maximum count value that is compared to the respective count register. Timer 0 and Timer 1 each have two compare registers.

If Timer 0 and/or Timer 1 is/are configured to count and compare first to Register A and then Register B, the tmrout0 or tmrout1 signals can be used to generate various duty-cycle wave forms.

Timer 2 has only one compare register, T2COMPA.

If one of these timer maxcount compare registers is set to 0000h, the respective timer will count from 0000h to FFFFh before generating an interrupt request. For example, a timer configured in this manner with a 40-MHz clock will interrupt every 6.5536 mS.

The value of these registers is undefined at reset (see Table 51).

**Table 51. Timer Maxcount Compare Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TC15–TC0 | | | | | | | | | | | | | | | |

- Bits [15–0]—TC15–TC0 Timer Compare Value → The timer will count to the value in the respective register before resetting the count value to 0.

### 5.1.30 T2CNT (060h), T1CNT (058h), and T0CNT (050h)

These registers are incremented by one every four internal clock cycles if the relevant timer is enabled.

The Increment of Timer 0 and Timer 1 may also be controlled by external signals tmrin0 and tmrin1 respectively, or prescaled by Timer 2.

Comparisons are made between the count registers and maxcount registers and action taken dependent on achieving the maximum count.

The value of these registers is undefined at reset (see Table 52).

**Table 52. Timer Count Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TC15–TC0 | | | | | | | | | | | | | | | |

- Bits [15–0]—TC15–TC0 Timer Count Value → This register has the value of the current count of the related timer that is incremented every fourth processor clock in internal clocked mode. Alternatively, the register is incremented each time the Timer 2 maxcount is reached if using Timer 2 as a prescaler. Timer 0 and Timer 1 may be externally clocked by tmrin0 and tmrin1 signals.

### 5.1.31 SPICON (044h) (Master Mode)

Serial Port Interrupt CONtrol Register. This register controls the operation of the asynchronous serial port interrupt source (SPI, Bit [10] in the Interrupt Request register). The value of this register is 001Fh at reset (see Table 53).

**Table 53. Serial Port Interrupt Control Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | Reserved | MSK | PR2–PR0 | | |

- Bits [15–5]—Reserved → Set to 0.
- Bit [4]—Reserved → Set to 1.
- Bit [3]—MSK Mask → This bit, when 0, enables the serial port to cause an interrupt. When this bit is 1, the serial port is prevented from generating an interrupt.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupt in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown below.

Values of PR2–PR0 by Priority

| Priority | PR2–PR0 |
|---|---|
| (High) 0 | 000b |
| 1 | 001b |
| 2 | 010b |
| 3 | 011b |
| 4 | 100b |
| 5 | 101b |
| 6 | 110b |
| (Low) 7 | 111b |

### 5.1.32 WDCON (044h) (Master Mode)

WatchDog Timer Interrupt CONtrol Register. These registers control the operation of the Watchdog Timer interrupt source. The value of this register is 000Fh at reset (see Table 54).

**Table 54. Watchdog Timer Interrupt Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | Reserved | MSK | PR2–PR0 | | |

- Bits [15–5]—Reserved → Set to 0.
- Bit [4]—Reserved → Set to 0.
- Bit [3]—MSK Mask → This bit, when 0, enables the Watchdog Timer to cause an interrupt. When this bit is 1 prevents the Watchdog Timer from generating an interrupt.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the Watchdog Timer interrupt in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown in the above table.

### 5.1.33 I4CON (040h) (Master Mode)

This register controls the operation of the int4 signal, which is only intended for use in fully nested mode. The interrupt is assigned to type 10h. The value of the I4CON register is 000Fh at reset (see Table 55).

**Table 55. INT4 Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | LTM | MSK | PR2–PR0 | | |

- Bits [15–5]—Reserved → Set to 0.
- Bit [4]—LTM Level-Triggered Mode → The int4 interrupt may be edge- or level-triggered, depending on the value of the bit. If LTM is 1, int4 is active high level-sensitive interrupt. If 0, it is a rising-edge triggered interrupt. The interrupt int4 must remain active (high) until serviced.
- Bit [3]—MSK Mask → The int4 signal can cause an interrupt if the MSK bit is 0. The int4 signal cannot cause an interrupt if the MSK bit is 1.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupt in relation to other interrupt signals. The interrupt priority is the lowest at 7 upon reset. The values of PR2–PR0 are shown in the above table.

### 5.1.34 I3CON (03eh) and I2CON (03ch) (Master Mode)

INT2/INT3 CONtrol Register. The int2 and int3 are designated as interrupt type 0eh and 0fh, respectively, and may be configured as the interrupt acknowledge pins inta0_n and inta1_n in cascade mode. The value of these registers is 000Fh at reset (see Table 56).

**Table 56. INT2/INT3 Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | LTM | MSK | PR2–PR0 | | |

- Bits [15–5]—Reserved → Set to 0.
- Bit [4]—LTM Level-Triggered Mode → The int2 or int3 interrupt may be edge- or level-triggered depending on the value of this bit. If LTM is 1, int2 or int3 is an active high level-sensitive interrupt. If 0, int2 or int3 is a rising-edge-triggered interrupt. The interrupt int2 or int3 must remain active (high) until acknowledged.
- Bit [3]—MSK Mask → The int2 or int3 signal can cause an interrupt if the MSK bit is 0. The int2 or int3 signal cannot cause an interrupt if the MSK bit is 1. The Interrupt Mask Register has a duplicate of this bit.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupt int2 or int3 in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown above.

### 5.1.35 I1CON (03ah) and I0CON (038h) (Master Mode)

INT0/INT1 CONtrol Register. The int0 and int1 are designated as interrupt type 0ch and 0dh, respectively, and may be configured as the interrupt acknowledge pins inta0 and inta1 in cascade mode. The value of these registers is 000Fh at reset (see Table 57).

**Table 57. INT0/INT1 Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | SFNM | C | LTM | MSK | PR2–PR0 | | |

- Bits [15–7]—Reserved → Set to 0.
- Bit [6]—SFNM Special Fully Nested Mode → This bit enables fully nested mode for int0 or int1 when set to 1.
- Bit [5]—C Cascade Mode → This bit enables cascade mode for int0 or int1 when set to 1.
- Bit [4]—LTM Level-Triggered Mode → The int0 or int1 interrupt may be edge- or level-triggered depending on the value of the bit. If LTM is 1, int0 or int1 is an active high-level-sensitive interrupt. If 0, either is a rising-edge-triggered interrupt and must remain active (high) until acknowledged.
- Bit [3]—MSK Mask → The int0 or int1 signal can cause an interrupt if the MSK bit is 0. If it is 1, they cannot. The Interrupt Mask Register has a duplicate of this bit.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupt int0 or int1 in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown above.

### 5.1.36 TCUCON (032h) (Master Mode)

Timer Control Unit Interrupt CONtrol Register. The three timers have their interrupts assigned to types 08h, 12h, and 13h and are configured by this register. The value of this register is 000Fh at reset (see Table 58).

**Table 58. Timer Control Unit Interrupt Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | MSK | PR2–PR0 | | |

- Bits [15–4]—Reserved → Set to 0.
- Bit [3]—MSK Mask → An interrupt source may cause an interrupt if the MSK bit is 0. If 1, it cannot. The Interrupt Mask Register has a duplicate of this bit.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupt in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown above.

### 5.1.37 T2INTCON (03ah), T1INTCON (038h), and T0INTCON (032h) (Slave Mode)

Timer INTerrupt CONtrol Register. The three timers, Timer 2, Timer 1, and Timer 0, each have an interrupt control register, whereas in master mode all three are masked and prioritized in one register (TCUCON). The value of these registers is 000Fh at reset (see Table 59).

**Table 59. Timer Interrupt Control Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | MSK | PR2–PR0 | | |

- Bits [15–4]—Reserved → Set to 0.
- Bit [3]—MSK Mask → Any of the interrupt sources may cause an interrupt if the MSK bit is 0. If 1, they cannot. The Interrupt Mask Register has a duplicate of this bit.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupts in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown above.

### 5.1.38 DMA1CON/INT6CON (036h) and DMA0CON/INT5CON (034h) (Master Mode)

DMA and INTerrupt CONtrol Register. The DMA0 and DMA1 interrupts have interrupt type 0ah and 0bh, respectively. These pins are configured as external interrupts or DMA requests in the respective DMA Control register. The value of these registers is 000Fh at reset (see Table 60).

**Table 60. DMA and Interrupt Control Register (Master Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | MSK | PR2–PR0 | | |

- Bits [15–4]—Reserved → Set to 0.
- Bit [3]—MSK Mask → Any of the interrupt sources may cause an interrupt if the MSK bit is 0. If 1, they cannot. The Interrupt Mask Register has a duplicate of this bit.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupts in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown above.

### 5.1.39 DMA1CON/INT6 (036h) and DMA0CON/INT5 (034h) (Slave Mode)

DMA and INTerrupt CONtrol Register. The two DMA control registers maintain their original functions and addressing that they possessed in Master Mode. These pins are configured as

external interrupts or DMA requests in the respective DMA Control register. The value of these registers is 000Fh at reset (see Table 61).

**Table 61. DMA and Interrupt Control Register (Slave Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | MSK | PR2–PR0 | | |

- Bits [15–4]—Reserved → Set to 0.
- Bit [3]—MSK Mask → Any of the interrupt sources may cause an interrupt if the MSK bit is 0. If 1, they cannot. The Interrupt Mask Register has a duplicate of this bit.
- Bits [2–0]—PR2–PR0 Priority → These bits define the priority of the serial port interrupts in relation to other interrupt signals. The interrupt priority is the lowest at 7 at reset. The values of PR2–PR0 are shown above.

### 5.1.40 INTSTS (030h) (Master Mode)

INTerrupt STatuS Register. The Interrupt status register contains the interrupt request status of each of the three timers, Timer 2, Timer 1, and Timer 0 (see Table 62).

**Table 62. Interrupt Status Register (Master Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DHLT | Reserved | | | | | | | | | | | | TMR2–TMR0 | | |

- Bit [15]—DHLT DMA Halt → DMA activity is halted when this bit is 1. It is set to 1 automatically when any non-maskable interrupt occurs and is cleared to 0 when an IRET instruction is executed. Interrupt handlers and other time-critical software may modify this bit directly to disable DMA transfers. However, the DHLT bit should not be modified by software if the timer interrupts are enabled as the function of this register because an interrupt request register for the timers would be compromised.
- Bits [14–3]—Reserved.
- Bits [2–0]—TMR2–TMR0 Timer Interrupt Request → A pending interrupt request is indicated by the respective timer, when any of these bits is 1.

> Note: The TMR bit in the REQST register is a logical OR of these timer interrupt requests.

### 5.1.41 INTSTS (030h) (Slave Mode)

When nonmaskable interrupts occur, the interrupt status register controls DMA operation and the interrupt request status of each of the three timers, Timer 2, Timer 1, and Timer 0 (see Table 63).

**Table 63. Interrupt Status Register (Slave Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DHLT | Reserved | | | | | | | | | | | | TMR2–TMR0 | | |

- Bit [15]—DHLT DMA Halt → DMA activity is halted when this bit is 1. It is set to 1 automatically when any non-maskable interrupt occurs and is cleared to 0 when an IRET instruction is executed.
- Bits [14–3]—Reserved.
- Bits [2–0]—TMR2–TMR0 Timer Interrupt Request → A pending interrupt request is indicated by the respective timer, when any of these bits is 1.

Note: The TMR bit in the REQST register is a logical OR of these timer interrupt requests.

### 5.1.42 REQST (02eh) (Master Mode)

Interrupt REQueST Register. This is a read-only register and such a read results in the status of the interrupt request bits presented to the interrupt controller. The REQST register is undefined on reset (see Table 64).

**Table 64. Interrupt Request Register (Master Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | SP0 | WD | I4–I0 | | | | | D1–D0 | | Reserved | TMR |

- Bits [15–11]—Reserved.
- Bit [10]—SP0 Serial Port 0 Interrupt Request → This is the serial port interrupt state and when enabled is the logical OR of all the serial port 0 interrupt sources, THRE, RDR, BRKI, FER, PER, and OER.
- Bit [9]—WD Watchdog Timer Interrupt Request → When it is a 1, the watchdog interrupt state indicates that an interrupt is pending.
- Bits [8–4]—I4–I0 Interrupt Requests → Setting any of these bits to 1 indicates that the relevant interrupt has a pending interrupt.
- Bits [3–2]—D1–D0 DMA Channel Interrupt Request → Setting either bit to 1 indicates that the respective DMA channel has a pending interrupt.
- Bit [1]—Reserved.

- Bit [0]—TMR Timer Interrupt Request → This is the timer interrupt state and is the logical OR of the timer interrupt requests. Setting this bit to 1 indicates that the timer control unit has a pending interrupt.

### 5.1.43 REQST (02eh) (Slave Mode)

This is a read-only register and such a read results in the status of the interrupt request bits presented to the interrupt controller. The status of these bits is available when this register is read.

When an internal interrupt request (D1, D0, TMR2, TMR1, or TMR0) occurs, the respective bit is set to 1. The internally generated interrupt acknowledge resets these bits. The REQST register contains 0000h on reset (see Table 65).

**Table 65. Interrupt Request Register (Slave Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | TMR2 | TMR1 | D1–D0 | | Reserved | TMR0 |

- Bits [15–6]—Reserved.
- Bit [5]—TMR2 Interrupt Requests → Setting this bit to 1 indicates that Timer 2 has a pending interrupt.
- Bit [4]—TMR1 Interrupt Requests → Setting this bit to 1 indicates that Timer 1 has a pending interrupt.
- Bits [3–2]—D1–D0 DMA Channel Interrupt Request → Setting either bit to 1 indicates that the respective DMA channel has a pending interrupt.
- Bit [1]—Reserved.
- Bit [0]—TMR0 Timer Interrupt Request → Setting this bit to 1 indicates that Timer 0 has a pending interrupt.

### 5.1.44 INSERV (02ch) (Master Mode)

IN-SERVice Register. The interrupt controller sets the bits in this register when the interrupt is taken. Writing the corresponding interrupt type to the End-of-Interrupt (EOI) register clears each of these bits.

When one of these bits is set, an interrupt request will not be generated by the microcontroller for the respective source. This prevents an interrupt from interrupting itself if interrupts are enabled in the ISR. This restriction is bypassed in Special Fully nested mode for the int0 and int1 sources. The INSERV register contains 0000h on reset (see Table 66).

**Table 66. In-Service Register (Master Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | SPI | WD | I4–I0 | | | | | D1–D0 | | Reserved | TMR |

- Bits [15–11]—Reserved.
- Bit [10]—SPI Serial Port Interrupt Request → This is the serial port 0 interrupt state.
- Bit [9]—WD Watchdog Timer Interrupt In-Service Request → This bit is the In-Service state of the Watchdog Timer.
- Bits [8–4]—I4–I0 Interrupt Requests → Setting any of these bits to 1 indicates that the relevant interrupt has a pending interrupt.
- Bits [3–2]—D1–D0 DMA Channel Interrupt In-Service → This bit is the In-Service state of the respective DMA channel.
- Bit [1]—Reserved.
- Bit [0]—TMR Timer Interrupt Request → This is the timer interrupt state and is the logical OR of the timer interrupt requests. Setting this bit to 1 indicates that the timer control unit has a pending interrupt.

### 5.1.45 INSERV (02ch) (Slave Mode)

This is a read-only register and such a read supplies the status of the interrupt request bits presented to the interrupt controller.

When an internal interrupt request (D1, D0, TMR2, TMR1, and TMR0) occurs, the respective bit is set to 1. The internally generated interrupt acknowledge resets these bits. The REQST register contains 0000h on reset (see Table 67).

**Table 67. In-Service Register (Slave Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | TMR2 | TMR1 | D1–D0 | | Reserved | TMR0 |

- Bits [15–6]—Reserved.
- Bit [5]—TMR2 Timer 2 Interrupt In Service → Timer 2 is being serviced when this bit is set to 1.
- Bit [4]—TMR1 Timer 1 Interrupt In Service → Timer 1 is being serviced when this bit is set to 1.

- Bits [3–2]—D1–D0 DMA Channel Interrupt In Service → The respective DMA channel is being serviced when this bit is set to 1.
- Bit [1]—Reserved.
- Bit [0]—TMR0 Timer Interrupt In Service → Timer 0 is being serviced when this bit is set to 1.

### 5.1.46 PRIMSK (02ah) (Master and Slave Mode)

PRIority MaSK Register. This register contains a value that sets the minimum priority level at which an interrupt can be generated by a maskable interrupt. The PRIMSK register contains 0007h on reset (see Table 68).

**Table 68. Priority Mask Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | PRM2–PRM0 | | |

- Bits [15–3]—Reserved → Set to 0.
- Bits [2–0]—PRM2–PRM0 Priority Field Mask → This three-bit field sets the minimum priority necessary for a maskable interrupt to generate an interrupt. Any maskable interrupt with a numerically higher value than that contained by these three bits is masked. The values of PR2–PR0 are shown below.

Values of PR2–PR0 by Priority

| Priority | PR2–PR0 |
|---|---|
| (High) 0 | 000b |
| 1 | 001b |
| 2 | 010b |
| 3 | 011b |
| 4 | 100b |
| 5 | 101b |
| 6 | 110b |
| (Low) 7 | 111b |

Any unmasked interrupt can generate an interrupt if the priority level is set to 7. On the other hand, if the priority level is set to say 4, only unmasked interrupts with a priority of 0 to 5 are permitted to generate interrupts.

### 5.1.47 IMASK (028h) (Master Mode)

Interrupt MASK Register. The interrupt mask register is read/write. Setting a bit in this register has the same effect as setting the MSK bit in the corresponding interrupt control register. Setting

a bit to 1 masks the interrupt. The interrupt request is enabled when the corresponding bit is set to 0. The IMASK register contains 07fdh on reset (see Table 69).

**Table 69. Interrupt MASK Register (Master Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | SPI | WD | I4–I0 | | | | | D1–D0 | | Reserved | TMR |

- Bits [15–11]—Reserved.
- Bit [10]—SPI Serial Port Interrupt Mask → Setting this bit to 1 is an indication that the asynchronous serial port interrupt is masked.
- Bit [9]—WD Watchdog Timer Interrupt In-Service Request → Setting this bit to 1 indicates that the Watchdog Timer interrupt is masked.
- Bits [8–4]—I4–I0 Interrupt Mask → Setting any of these bits to 1 is an indication that the relevant interrupt is masked.
- Bits [3–2]—D1–D0 DMA Channel Interrupt Mask → Setting this bit to 1 is an indication that the respective DMA channel interrupt is masked.
- Bit [1]—Reserved.
- Bit [0]—TMR Timer Interrupt Mask → When set to 1, it indicates that the timer control unit interrupt is masked.

### 5.1.48 IMASK (028h) (Slave Mode)

Interrupt MASK Register. The interrupt mask register is read/write. Setting a bit in this register is effectively the same as setting the MSK bit in the corresponding interrupt control register. Setting a bit to 1 masks the interrupt request. The interrupt request is enabled when the corresponding bit is set to 0. The IMASK register contains 003dh on reset (see Table 70).

**Table 70. Interrupt MASK Register (Slave Mode)**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | TMR2 | TMR1 | D1–D0 | | Reserved | TMR0 |

- Bits [15–6]—Reserved.
- Bit [5]—TMR2 Timer 2 Interrupt Mask → This bit provides an indication of the state of the mask bit in the Timer Interrupt Control register. When set to 1, it indicates that the interrupt request is masked.

- Bit [4]—TMR1 Timer 1 Interrupt Mask → This bit provides an indication of the state of the mask bit in the Timer Interrupt Control register. When set to 1, it indicates that the interrupt request is masked.
- Bits [3–2]—D1–D0 DMA Channel Interrupt Mask → This bit provides an indication of the state of the mask bit in the respective DMA channel Interrupt Control register. When set to 1, it indicates that the interrupt request is masked.
- Bit [1]—Reserved.
- Bit [0]—TMR0 Timer Interrupt Mask → This bit provides an indication of the state of the mask bit in the Timer Interrupt Control register. When set to 1, it indicates that the interrupt request is masked.

### 5.1.49 POLLST (026h) (Master Mode)

POLL STatus Register. This register reflects the current state of the Poll register and can be read without affecting its contents. However, when the Poll Register is read, it causes the current interrupt to be acknowledged and replaced by the next interrupt. The poll status register is read-only (see Table 71).

**Table 71. POLL Status Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IREQ | Reserved | | | | | | | | | | S4–S0 | | | | |

- Bit [15]—IREQ Interrupt Request → This bit is set to 1 when an interrupt is pending. During this state the S4–S0 bits contain valid data.
- Bits [14–5]—Reserved.
- Bits [4–0]—S4–S0 Poll Status → These bits show the interrupt type of the highest priority pending interrupt.

The interrupt service routine does not begin execution automatically with the IS bit set. Rather, the application software must execute the appropriate ISR.

### 5.1.50 POLL (024h) (Master Mode)

POLL Register. When the Poll Register is read, it causes the current interrupt to be acknowledged and be replaced by the next interrupt. The poll status register reflects the current state of the Poll register and can be read without affecting its contents. The POLL register is read-only (see Table 72).

**Table 72. Poll Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IREQ | Reserved | | | | | | | | | | S4–S0 | | | | |

- Bit [15]—IREQ Interrupt Request → This bit is set to 1 when an interrupt is pending. During this state, the S4–S0 bits contain valid data.
- Bits [14–5]—Reserved.
- Bits [4–0]—S4–S0 Poll Status → These bits show the interrupt type of the highest priority pending interrupt.

### 5.1.51 EOI (022h) End-Of-Interrupt Register (Master Mode)

The In Service flags of the In-Service register are reset when a write is made to the EOI register. The interrupt service routine (ISR) should write to the EOI to reset the IS bit, in the In-Service register, for the interrupt before executing an IRET instruction that ends an interrupt service routine. Because it is most secure, the specific EOI reset is the preferred method for resetting the IS bits. The EOI register is write-only (see Table 73).

**Table 73. End-of-Interrupt Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NSPEC | Reserved | | | | | | | | | | S4–S0 | | | | |

- Bit [15]—NSPEQ Non-Specific EOI → When set to 1, this bit is a non-specific EOI. When 0, it indicates the specific EOI.
- Bits [14–5]—Reserved.
- Bits [4–0]—S4–S0 Source Interrupt Type → These bits show the interrupt type of the highest priority pending interrupt.

### 5.1.52 EOI (022h) Specific End-Of-Interrupt Register (Slave Mode)

Specific End-Of-Interrupt Register. An In-Service flag of a specific priority in the In-Service register, it is reset when a write is made to the EOI register. A three-bit user-supplied priority-level value that points to the in-service bit that is to be reset. Writing this value to this register resets the specific bit. The EOI register is write-only and undefined at reset (see Table 74).

**Table 74. Specific End-of-Interrupt Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | L2–L0 | | |

- Bits [15–3]—Reserved → Write as 0.
- Bits [2–0]—L2–L0 Interrupt Type → The priority or the IS (interrupt service) bit to be reset is encoded in these three bits. Writing to these bits caused the issuance of an EOI for the interrupt type (see Table 14, Interrupt Types).

### 5.1.53 INTVEC (020h) Interrupt Vector Register (Slave Mode)

The CPU shifts left 2 bits (multiplies by 4) an 8-bit interrupt type, generated by the interrupt controller, to produce an offset into the interrupt vector table. The INTVEC register is undefined at reset (see Table 75).

**Table 75. Interrupt Vector Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | T4–T0 | | | | | Reserved | | |

- Bits [15–8]—Reserved → Read as 0.
- Bits [7–3]—T4–T0 Interrupt Type → These five bits contain the five most significant bits of the interrupt types used for the internal interrupt type. The least significant three bits of the interrupt type are supplied by the interrupt controller, as set by the priority level of the interrupt request.
- Bits [2–0]—Reserved → Read as 0.

### 5.1.54 SSR (018h)

Synchronous Serial Receive Register. This register holds the serial data received on the SSI port. The value of the SSR register is undefined at reset (see Table 76).

**Table 76. Synchronous Serial Receive Register**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | SR7-SR0 | | | | | | | |

- Bits [15–8]—Reserved.
- Bits [7–0]—SR7–SR0 → Data received over the SDATA pin.

### 5.1.55 SSD0 (016h) and SSD0 (014h)

Synchronous Serial Transmit Registers. These registers hold the data to be transmitted by the SSI ports. The value of these registers is undefined at reset (see Table 77).

**Table 77. Synchronous Serial Transmit Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | SD7–SD0 | | | | | | | |

- Bits [15–8]—Reserved.
- Bits [7–0]—SD7–SD0 → Data to be transmitted over the SDATA pin.

### 5.1.56 SSC (012h)

Synchronous Serial Control Register. This register controls the operation of the sden1 and sden0 outputs and the baud rate of the SSI port. The sden1 and sden0 outputs are held high when the respective bit is set to 1, but in the event that both DE1 and DE0 are set to 1 then only sden0 will be held high. The value of the SSR register is 0000h at reset (see Table 78).

**Table 78. Synchronous Serial Control Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | SCLKDIV | | Reserved | | DE1 | DE0 |

- Bits [15–6]—Reserved.
- Bits [5–4]—SCLKDIV SCLK Divide → These bits set the SCLK frequency. SCLK is the result of dividing the internal processor clock by 2, 4, 8, or 16 as shown below.

| SCLKDIV | SCLK Frequency Divider |
|---|---|
| 00b | Processor Clock/2 |
| 01b | Processor Clock/4 |
| 10b | Processor Clock/8 |
| 11b | Processor Clock/16 |

- Bits [3–2]—Reserved.
- Bit [1]—DE1 SDEN1 → The SDEN1 bit is held high when this bit is set to 1 and SDEN1 is held low when this bit is set to 0.
- Bit [0]—DE0 SDEN0 → The SDEN0 bit is held high when this bit is set to 1 and SDEN0 is held low when this bit is set to 0.

### 5.1.57 SSS (010h)

Synchronous Serial Status Register. This is a read-only register that indicates the state of the SSI port. The value of the SSR register is 0000h at reset (see Table 79).

**Table 79. Synchronous Serial Status Registers**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | RE/TE | DR/DT | PB |

- Bits [15–3]—Reserved.
- Bit [2]—RE/TE Receive/Transmit Error Detect → This bit is set to 1 when a read of the Synchronous Serial Received register or a write to one of the transmit register is detected while the interface is busy (PB = 1). This bit is reset to 0 when the SDEN output is not active (DE1–DE0 in the SSC register are 00h).
- Bit [1]—DR/DT Data Receive/Transmit Complete → This bit is set to a 1 when the transmission of data Bit [7] is completed (SCLK rising edge) during a transmit or receive operation. This bit is reset by a read of the SSR register, when either the SSD0 or SSD1 register is written, when the SSS register is read (unless the SSI completes an operation and sets the bit in the same cycle), or when both SDEN0 and SDEN1 become inactive.
- Bit [0]—PB SSI Port Busy → This bit indicates that a data transmit or receive is occurring when it is set to 1. When set to 0, it indicates that the port is ready to transmit or receive data.

### 5.2 Reference Documents

Additional information on the operation and programming of the IA186EM/ IA188EM can be found in the following Advanced Micro Devices (AMD) publications:

- Am186™EM and Am188™EM Microcontrollers *User's Manual*, February 1997, Publication 19713.
- Am186™EM/EMLV and Am188™EM/EMLV Preliminary Data Sheet, February 1997, Publication 19168, Rev. E, Amendment 0.

## 6. AC Specifications

Table 80 presents the AC characteristics over commercial operating ranges (40 MHz). Tables 81 and 82 present the alphabetic and numeric keys to waveform parameters, respectively. Figure 11 presents the read cycle. Figure 12 presents the multiple read cycles. Table 83 presents the read cycle timing. Figure 13 presents the write cycle. Figure 14 presents the multiple write cycles. Table 84 presents the write cycle timing.

Figure 15 presents the PSRAM read cycle. Table 85 presents the PSRAM read cycle timing. Figure 16 presents the PSRAM write cycle. Table 86 presents the PSRAM write cycle timing. Figure 17 presents the PSRAM refresh cycle. Table 87 presents the PSRAM refresh cycle timing. Figure 18 presents the interrupt acknowledge cycle. Table 88 presents the interrupt

acknowledge cycle timing. Figure 19 presents the software halt cycle. Table 89 presents the software halt cycle timing. Figure 20 presents the clock—active mode. Figure 21 presents the clock—power-save mode. Table 90 presents the clock timing.

Figure 22 presents the srdy—synchronous ready. Figure 23 presents the ardy—asynchronous ready. Figure 24 presents the peripherals. Table 91 presents the ready and peripheral timing. Figures 25 and 26 present Reset 1 and Reset 2, respectively. Figures 27 and 28 present the bus hold entering and bus hold leaving, respectively. Table 92 presents the reset and bus hold timing.

Figure 29 presents the synchronous serial interface. Table 93 presents the synchronous serial interface timing.

**Table 80. AC Characteristics Over Commercial Operating Ranges (40 MHz)**

| No. | Name | Description | Min | Max |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | – |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 3 | tCHSV | Status Active Delay | 0 | 6 |
| 4 | tCLSH | Status Inactive Delay | 0 | 6 |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 6 | tCLAX | Address Hold | 0 | 12 |
| 8 | tCHDX | Status Hold Time | 0 | – |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| 12 | tAVLL | *ad* Address Valid to ale Low | tCLCH | – |
| 13 | tLLAX | *ad* Address Hold from ale Inactive | tCHCL | – |
| 14 | tAVCH | *ad* Address Valid to Clock High | 0 | – |
| 15 | tCLAZ | *ad* Address Float Delay | 0 | 12 |
| 16 | tCLCSV | mcs_n/pcs_n Inactive Delay | 0 | 12 |
| 17 | tCXCSX | mcs_n/pcs_n Hold from Command Inactive | tCLCH | – |
| 18 | tCHCSX | mcs_n/pcs_n Inactive Delay | 0 | 12 |
| 19 | tDXDL | den_n Inactive to dt/r_n Low | 0 | – |
| 20 | tCVCTV | Control Active Delay 1 | 0 | 10 |
| 21 | tCVDEX | den_n Inactive Delay | 0 | 0 |
| 22 | tCHCTV | Control Active Delay 2 | 0 | 10 |
| 23 | tLHAV | ale High to Address Valid | 7.5 | – |
| 80 | tCLCLX | lcs_n Inactive Delay | 0 | 9 |
| 81 | tCLCSL | lcs_n Active Delay | 0 | 9 |
| 82 | tCLRF | clkouta High to rfsh_n Invalid | 0 | 12 |
| 84 | tLRLL | lcs_n Precharge Pulse Width | tCLCL + tCLCH | – |

**Table 80. AC Characteristics Over Commercial Operating Ranges (40 MHz)** *(Continued)*

| No. | Name | Description | Min | Max |
|---|---|---|---|---|
| **Read Cycle Timing Responses** | | | | |
| 24 | tAZRL | *ad* Address Float to rd_n Active | 0 | – |
| 25 | tCLRL | rd_n Active Delay | 0 | 10 |
| 26 | tRLRH | rd_n Pulse Width | tCLCL | – |
| 27 | tCLRH | rd_n Inactive Delay | 0 | 10 |
| 28 | tRHLH | rd_n Inactive to ale High | tCLCH | – |
| 29 | tRHAV | rd_n Inactive to *ad* Address Active | tCLCL | – |
| 30 | tCLDOX | Data Hold Time | 0 | – |
| **Write Cycle Timing Responses** | | | | |
| 31 | tCVCTX | Control Inactive Delay | 0 | 10 |
| 32 | tWLWH | wr_n Pulse Width | 2tCLCL | – |
| 33 | tWHLH | wr_n Inactive to ale High | tCLCH | – |
| 34 | tWHDX | Data Hold after wr_n | tCLCL | – |
| 35 | tWHDEX | wr_n Inactive to den_n Inactive | tCLCH | – |
| 41 | tDSHLH | ds_n Inactive to ale Inactive | tCLCH | – |
| 59 | tRHDX | rd_n High to Data Hold on *ad* Bus | 0 | – |
| 65 | tAVWL | *a* Address Valid to wr_n Low | tCLCL + tCHCL | – |
| 66 | tAVRL | *a* Address Valid to rd_n Low | tCLCL + tCHCL | – |
| 67 | tCHCSV | clkouta High to lcs_n/usc_n Valid | 0 | 9 |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |
| 87 | tAVBL | *a* Address Valid to whb_n/wlb_n Low | tCHCL -1.5 | tCHCL |
| **Refresh Timing Cycle Parameters** | | | | |
| 79 | tCHRFD | clkouta High to rfsh_n Valid | 0 | 12 |
| 82 | tCLRF | clkouta High to rfsh_n Invalid | 0 | 12 |
| 85 | tRFCY | rfsh_n Cycle Time | 6tCLCL | – |
| 86 | tLCRF | lcs_n Inactive to rfsh_n Active Delay | 2tCLCL | – |
| **clkin Timing** | | | | |
| 36 | tCKIN | x1 Period | 25 | 66 |
| 37 | tCLCK | x1 Low Time | 7.5 | – |
| 38 | tCHCK | x1 High Time | 7.5 | – |
| 39 | tCKHL | x1 Fall Time | – | 5 |
| 40 | tCKLH | x1 Rise time | – | 5 |
| **clkout Timing** | | | | |
| 42 | tCLCL | clkouta Period | 25 | – |
| 43 | tCLCH | clkouta Low Time | TCLCL/2 | – |
| 44 | tCHCL | clkouta High Time | TCLCL/2 | – |
| 45 | tCH1CH2 | clkouta Rise Time | – | 3 |
| 46 | tCL2CL1 | clkouta Fall Time | – | 3 |
| 61 | tLOCK | Maximum PLL Lock Time | – | 0.5 |
| 69 | tCICOA | x1 to clkouta Skew | – | 25 |
| 70 | tCICOB | x1 to clkoutb Skew | – | 35 |

**Table 80. AC Characteristics Over Commercial Operating Ranges (40 MHz)** *(Continued)*

| No. | Name | Description | Min | Max |
|---|---|---|---|---|
| **Ready and Peripheral Timing Requirements** | | | | |
| 47 | tSRYCL | srdy Transition Setup Time | 10 | – |
| 48 | tCLSRY | srdy Transition Hold Time | 3 | – |
| 49 | tARYCH | ardy Resolution Transition Setup Time | 9 | – |
| 50 | tCLARX | ardy Active Hold Time | 4 | – |
| 51 | tARYCHL | ardy Inactive Holding Time | 6 | – |
| 52 | tARYLCL | ardy Setup Time | 9 | – |
| 53 | tINVCH | Peripheral Setup Time | 10 | – |
| 54 | tINVCL | drq Setup Time | 10 | – |
| **Peripheral Timing Responses** | | | | |
| 55 | tCLTMV | Timer Output Delay | 0 | 12 |
| **Reset & Hold Timing Requirements** | | | | |
| 57 | tRESIN | res_n Setup Time | 10 | – |
| 58 | tHVCL | hld Setup Time | 10 | – |
| **Reset and Hold Timing Responses** | | | | |
| 62 | tCLHAV | hlda Valid Delay | 0 | 7 |
| 63 | tCHCZ | Command Lines Float Delay | 0 | 12 |
| 64 | tCHCV | Command Lines Valid Delay (after Float) | 0 | 12 |
| **Synchronous Serial Port Timing Requirements** | | | | |
| 75 | tDVSH | Data Valid to sclk High | 10 | – |
| 77 | tSHDX | sclk High to SPI Data Hold | 3 | – |
| **Synchronous Serial Port Timing Responses** | | | | |
| 71 | tCLEV | clkouta Low to sden Valid | 0 | 12 |
| 72 | tCLSL | clkouta Low to sclk High | 0 | 12 |
| 78 | tSLDV | sclk Low to Data Valid | 0 | 12 |

## Table 81. Alphabetic Key to Waveform Parameters

| No. | Name | Description |
|---|---|---|
| 49 | tARYCH | ardy Resolution Transition Setup Time |
| 51 | tARYCHL | ardy Inactive Holding Time |
| 52 | tARYLCL | ardy Setup Time |
| 87 | tAVBL | *a* Address Valid to whb_n/wlb_n Low |
| 14 | tAVCH | *ad* Address Valid to Clock High |
| 12 | tAVLL | *ad* Address Valid to ale Low |
| 66 | tAVRL | *a* Address Valid to rd_n Low |
| 65 | tAVWL | *a* Address Valid to wr_n Low |
| 24 | tAZRL | *ad* Address Float to rd_n Active |
| 45 | tCH1CH2 | clkouta Rise Time |
| 68 | tCHAV | clkouta High to *a* Address Valid |
| 38 | tCHCK | x1 High Time |
| 44 | tCHCL | clkouta High Time |
| 67 | tCHCSV | clkouta High to lcs_n/usc_n Valid |
| 18 | tCHCSX | mcs_n/pcs_n Inactive Delay |
| 22 | tCHCTV | Control Active Delay 2 |
| 64 | tCHCV | Command Lines Valid Delay (after Float) |
| 63 | tCHCZ | Command Lines Float Delay |
| 8 | tCHDX | Status Hold Time |
| 9 | tCHLH | ale Active Delay |
| 11 | tCHLL | ale Inactive Delay |
| 79 | tCHRFD | clkouta High to rfsh_n Valid |
| 3 | tCHSV | Status Active Delay |
| 69 | tCICOA | x1 to clkouta Skew |
| 70 | tCICOB | x1 to clkoutb Skew |
| 39 | tCKHL | x1 Fall Time |
| 36 | tCKIN | x1 Period |
| 40 | tCKLH | x1 Rise time |
| 46 | tCL2CL1 | clkouta Fall Time |
| 50 | tCLARX | ardy Active Hold Time |
| 5 | tCLAV | *ad* Address Valid Delay |
| 6 | tCLAX | Address Hold |
| 15 | tCLAZ | *ad* Address Float Delay |
| 43 | tCLCH | clkouta Low Time |
| 37 | tCLCK | x1 Low Time |
| 42 | tCLCL | clkouta Period |
| 80 | tCLCLX | lcs_n Inactive Delay |
| 81 | tCLCSL | lcs_n Active Delay |
| 16 | tCLCSV | mcs_n/pcs_n Inactive Delay |
| 30 | tCLDOX | Data Hold Time |

**Table 81. Alphabetic Key to Waveform Parameters** *(Continued)*

| No. | Name | Description |
|---|---|---|
| 7 | tCLDV | Data Valid Delay |
| 2 | tCLDX | Data in Hold |
| 71 | tCLEV | clkouta Low to sden Valid |
| 62 | tCLHAV | hlda Valid Delay |
| 82 | tCLRF | clkouta High to rfsh_n Invalid |
| 27 | tCLRH | rd_n Inactive Delay |
| 25 | tCLRL | rd_n Active Delay |
| 4 | tCLSH | Status Inactive Delay |
| 72 | tCLSL | clkouta Low to sclk Low |
| 48 | tCLSRY | srdy Transition Hold Time |
| 55 | tCLTMV | Timer Output Delay |
| 83 | tCOAOB | clkouta to clkoutb Skew |
| 20 | tCVCTV | Control Active Delay 1 |
| 31 | tCVCTX | Control Inactive Delay |
| 21 | tCVDEX | den_n Inactive Delay |
| 17 | tCXCSX | mcs_n/pcs_n Hold from Command Inactive |
| 1 | tDVCL | Data in Setup |
| 75 | tDVSH | Data Valid to SCLK High |
| 19 | tDXDL | den_n Inactive to dt/r_n Low |
| 58 | tHVCL | hld Setup Time |
| 53 | tINVCH | Peripheral Setup Time |
| 54 | tINVCL | drq Setup Time |
| 86 | tLCRF | lcs_n Inactive to rfsh_n Active Delay |
| 23 | tLHAV | ale High to Address Valid |
| 10 | tLHLL | ale Width |
| 13 | tLLAX | *ad* Address Hold from ALE Inactive |
| 61 | tLOCK | Maximum PLL Lock Time |
| 84 | tLRLL | lcs_n Precharge Pulse Width |
| 57 | tRESIN | res_n Setup Time |
| 85 | tRFCY | rfsh_n Cycle Time |
| 29 | tRHAV | rd_n Inactive to *ad* Address Active |
| 59 | tRHDX | rd_n High to Data Hold on *ad* Bus |
| 28 | tRHLH | rd_n Inactive to ale High |
| 26 | tRLRH | rd_n Pulse Width |
| 77 | tSHDX | sclk High to SPI Data Hold |
| 78 | tSLDV | sclk Low SPI Data Hold |
| 47 | tSRYCL | srdy Transition Setup Time |
| 35 | tWHDEX | wr_n Inactive to den_n Inactive |
| 34 | tWHDX | Data Hold after wr_n |
| 33 | tWHLH | wr_n Inactive to ale High |
| 32 | tWLWH | wr_n Pulse Width |

## Table 82. Numeric Key to Waveform Parameters

| No. | Name | Description |
|---|---|---|
| 1 | tDVCL | Data in Setup |
| 2 | tCLDX | Data in Hold |
| 3 | tCHSV | Status Active Delay |
| 4 | tCLSH | Status Inactive Delay |
| 5 | tCLAV | *ad* Address Valid Delay |
| 6 | tCLAX | Address Hold |
| 7 | tCLDV | Data Valid Delay |
| 8 | tCHDX | Status Hold Time |
| 9 | tCHLH | ale Active Delay |
| 10 | tLHLL | ale Width |
| 11 | tCHLL | ale Inactive Delay |
| 12 | tAVLL | *ad* Address Valid to ALE Low |
| 13 | tLLAX | *ad* Address Hold from ALE Inactive |
| 14 | tAVCH | *ad* Address Valid to Clock High |
| 15 | tCLAZ | *ad* Address Float Delay |
| 16 | tCLCSV | mcs_n/pcs_n Inactive Delay |
| 17 | tCXCSX | mcs_n/pcs_n Hold from Command Inactive |
| 18 | tCHCSX | mcs_n/pcs_n Inactive Delay |
| 19 | tDXDL | den_n Inactive to dt/r_n Low |
| 20 | tCVCTV | Control Active Delay 1 |
| 21 | tCVDEX | den_n Inactive Delay |
| 22 | tCHCTV | Control Active Delay 2 |
| 23 | tLHAV | ale High to Address Valid |
| 24 | tAZRL | *ad* Address Float to rd_n Active |
| 25 | tCLRL | rd_n Active Delay |
| 26 | tRLRH | rd_n Pulse Width |
| 27 | tCLRH | rd_n Inactive Delay |
| 28 | tRHLH | rd_n Inactive to ale High |
| 29 | tRHAV | rd_n Inactive to *ad* Address Active |
| 30 | tCLDOX | Data Hold Time |
| 31 | tCVCTX | Control Inactive Delay |
| 32 | tWLWH | wr_n Pulse Width |
| 33 | tWHLH | wr_n Inactive to ale High |
| 34 | tWHDX | Data Hold after wr_n |
| 35 | tWHDEX | wr_n Inactive to den_n Inactive |
| 36 | tCKIN | x1 Period |
| 37 | tCLCK | x1 Low Time |
| 38 | tCHCK | x1 High Time |
| 39 | tCKHL | x1 Fall Time |
| 40 | tCKLH | x1 Rise time |

**Table 82. Numeric Key to Waveform Parameters** *(Continued)*

| No. | Name | Description |
|---|---|---|
| 42 | tCLCL | clkouta Period |
| 43 | tCLCH | clkouta Low Time |
| 44 | tCHCL | clkouta High Time |
| 45 | tCH1CH2 | clkouta Rise Time |
| 46 | tCL2CL1 | clkouta Fall Time |
| 47 | tSRYCL | srdy Transition Setup Time |
| 48 | tCLSRY | srdy Transition Hold Time |
| 49 | tARYCH | ardy Resolution Transition Setup Time |
| 50 | tCLARX | ardy Active Hold Time |
| 51 | tARYCHL | ardy Inactive Holding Time |
| 52 | tARYLCL | ardy Setup Time |
| 53 | tINVCH | Peripheral Setup Time |
| 54 | tINVCL | drq Setup Time |
| 55 | tCLTMV | Timer Output Delay |
| 57 | tRESIN | res_n Setup Time |
| 58 | tHVCL | hld Setup Time |
| 59 | tRHDX | rd_n High to Data Hold on *ad* Bus |
| 61 | tLOCK | Maximum PLL Lock Time |
| 62 | tCLHAV | hlda Valid Delay |
| 63 | tCHCZ | Command Lines Float Delay |
| 64 | tCHCV | Command Lines Valid Delay (after Float) |
| 65 | tAVWL | *a* Address Valid to wr_n Low |
| 66 | tAVRL | *a* Address Valid to rd_n Low |
| 67 | tCHCSV | clkouta High to lcs_n/usc_n Valid |
| 68 | tCHAV | clkouta High to *a* Address Valid |
| 69 | tCICOA | x1 to clkouta Skew |
| 70 | tCICOB | x1 to clkoutb Skew |
| 71 | tCLEV | clkouta Low to sden Valid |
| 72 | tCLSL | clkouta Low to sclk High |
| 75 | tDVSH | Data Valid to sclk High |
| 77 | tSHDX | sclk High to SPI Data Hold |
| 78 | tSLDV | sclk Low to Data Valid |
| 79 | tCHRFD | clkouta High to rfsh_n Valid |
| 80 | tCLCLX | lcs_n Inactive Delay |
| 81 | tCLCSL | lcs_n Active Delay |
| 82 | tCLRF | clkouta High to rfsh_n Invalid |
| 83 | tCOAOB | clkouta to clkoutb Skew |
| 84 | tLRLL | lcs_n Precharge Pulse Width |
| 85 | tRFCY | rfsh_n Cycle Time |
| 86 | tLCRF | lcs_n Inactive to rfsh_n Active Delay |
| 87 | tAVBL | *a* Address Valid to whb_n/wlb_n Low |

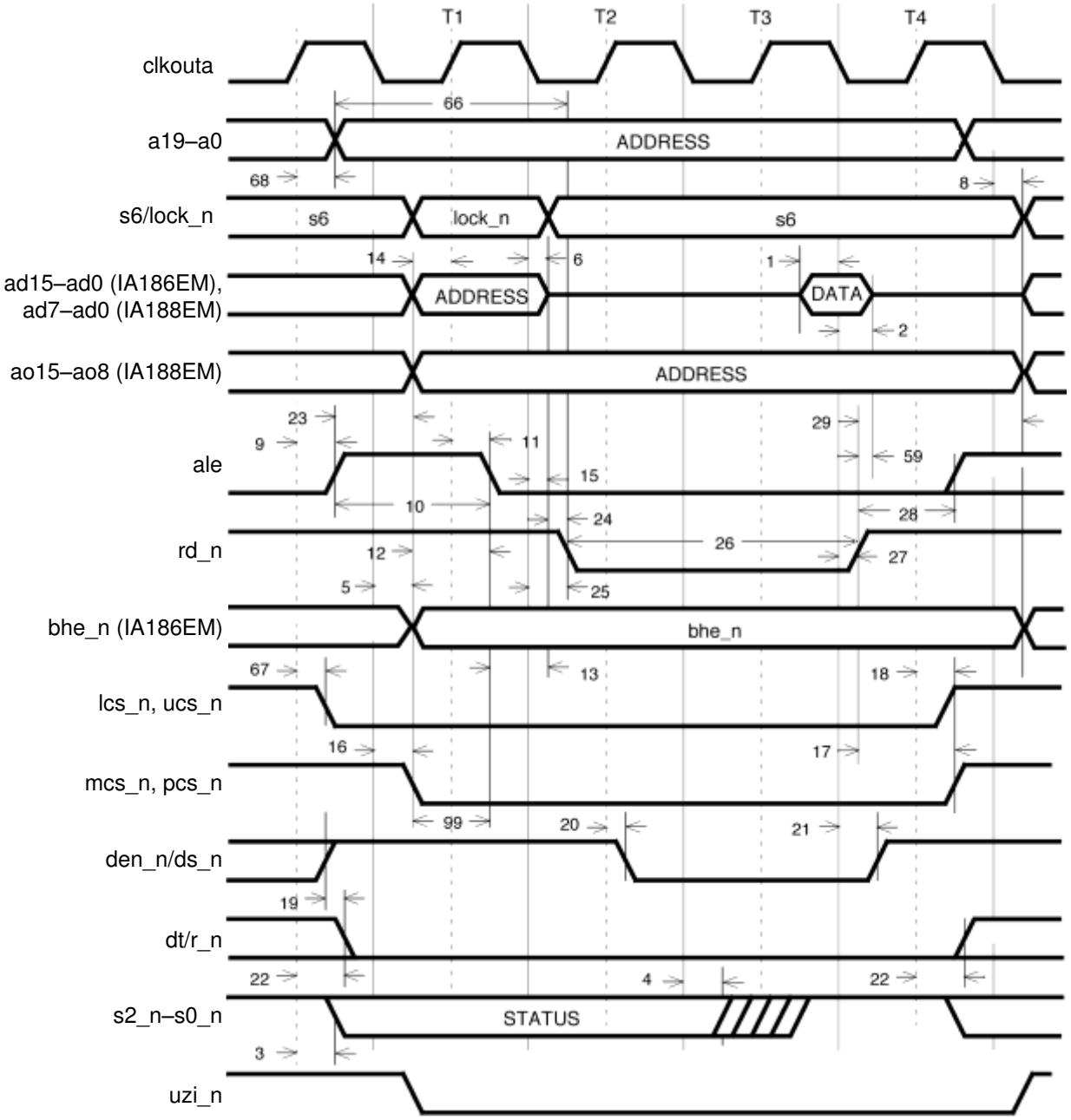



**Figure 11. Read Cycle**

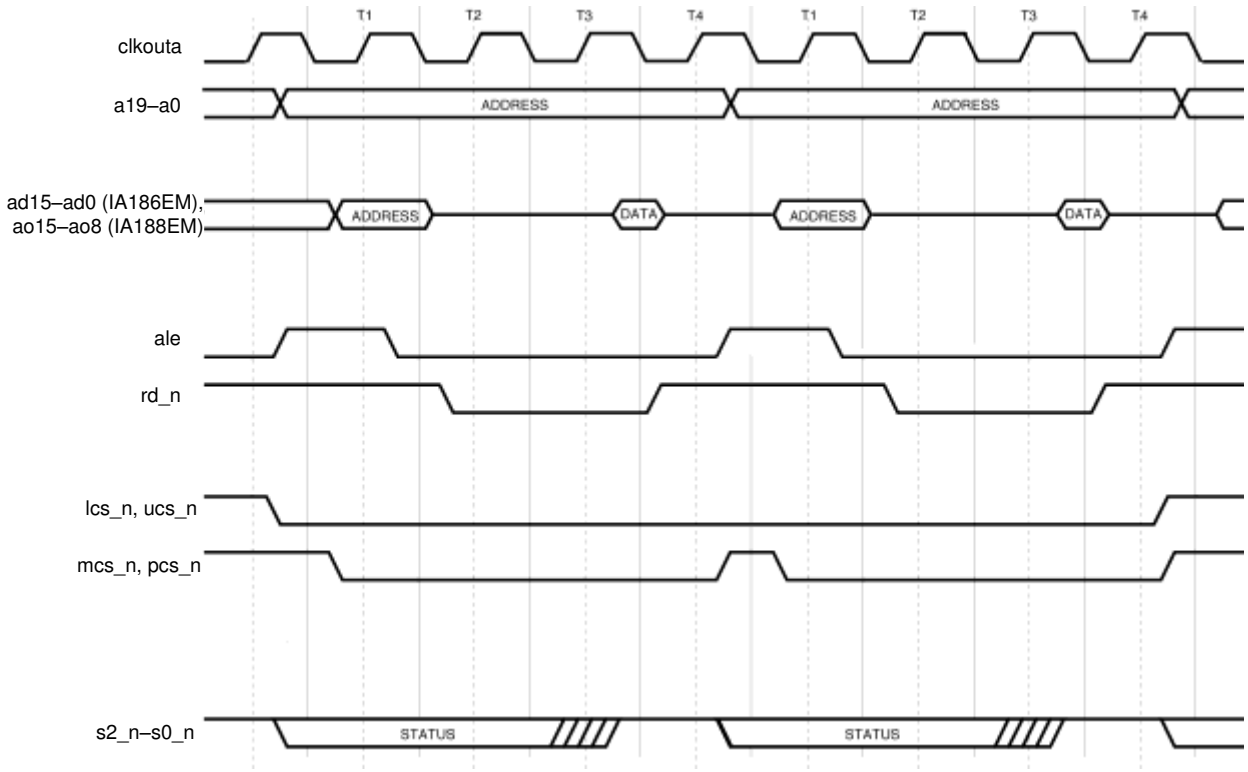



**Figure 12. Multiple Read Cycles**

## Table 83. Read Cycle Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | – |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 3 | tCHSV | Status Active Delay | 0 | 6 |
| 4 | tCLSH | Status Inactive Delay | 0 | 6 |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 6 | tCLAX | Address Hold | 0 | 12 |
| 8 | tCHDX | Status Hold Time | 0 | – |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| 12 | tAVLL | *ad* Address Valid to ale Low | tCLCH | – |
| 13 | tLLAX | *ad* Address Hold from ale Inactive | tCHCL | – |
| 14 | tAVCH | *ad* Address Valid to Clock High | 0 | – |
| 15 | tCLAZ | *ad* Address Float Delay | 0 | 12 |
| 16 | tCLCSV | mcs_n/pcs_n Inactive Delay | 0 | 12 |
| 17 | tCXCSX | mcs_n/pcs_n Hold from Command Inactive | tCLCH | – |
| 18 | tCHCSX | mcs_n/pcs_n Inactive Delay | 0 | 12 |
| 19 | tDXDL | den_n Inactive to dt/r_n Low | 0 | – |
| 20 | tCVCTV | Control Active Delay 1 | 0 | 10 |
| 21 | tCVDEX | den_n Inactive Delay | 0 | 9 |
| 22 | tCHCTV | Control Active Delay 2 | 0 | 10 |
| 23 | tLHAV | ale High to Address Valid | 7.5 | – |
| **Read Cycle Timing Responses** | | | | |
| 24 | tAZRL | *ad* Address Float to rd_n Active | 0 | – |
| 25 | tCLRL | rd_n Active Delay | 0 | 10 |
| 26 | tRLRH | rd_n Pulse Width | tCLCL | – |
| 27 | tCLRH | rd_n Inactive Delay | 0 | 10 |
| 28 | tRHLH | rd_n Inactive to ale High | tCLCH | – |
| 29 | tRHAV | rd_n Inactive to *ad* Address Active | tCLCL | – |
| 66 | tAVRL | *a* Address Valid to rd_n Low | tCLCL + tCHCL | – |
| 67 | tCHCSV | clkouta High to lcs_n/usc_n Valid | 0 | 9 |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |

[a]In nanoseconds.

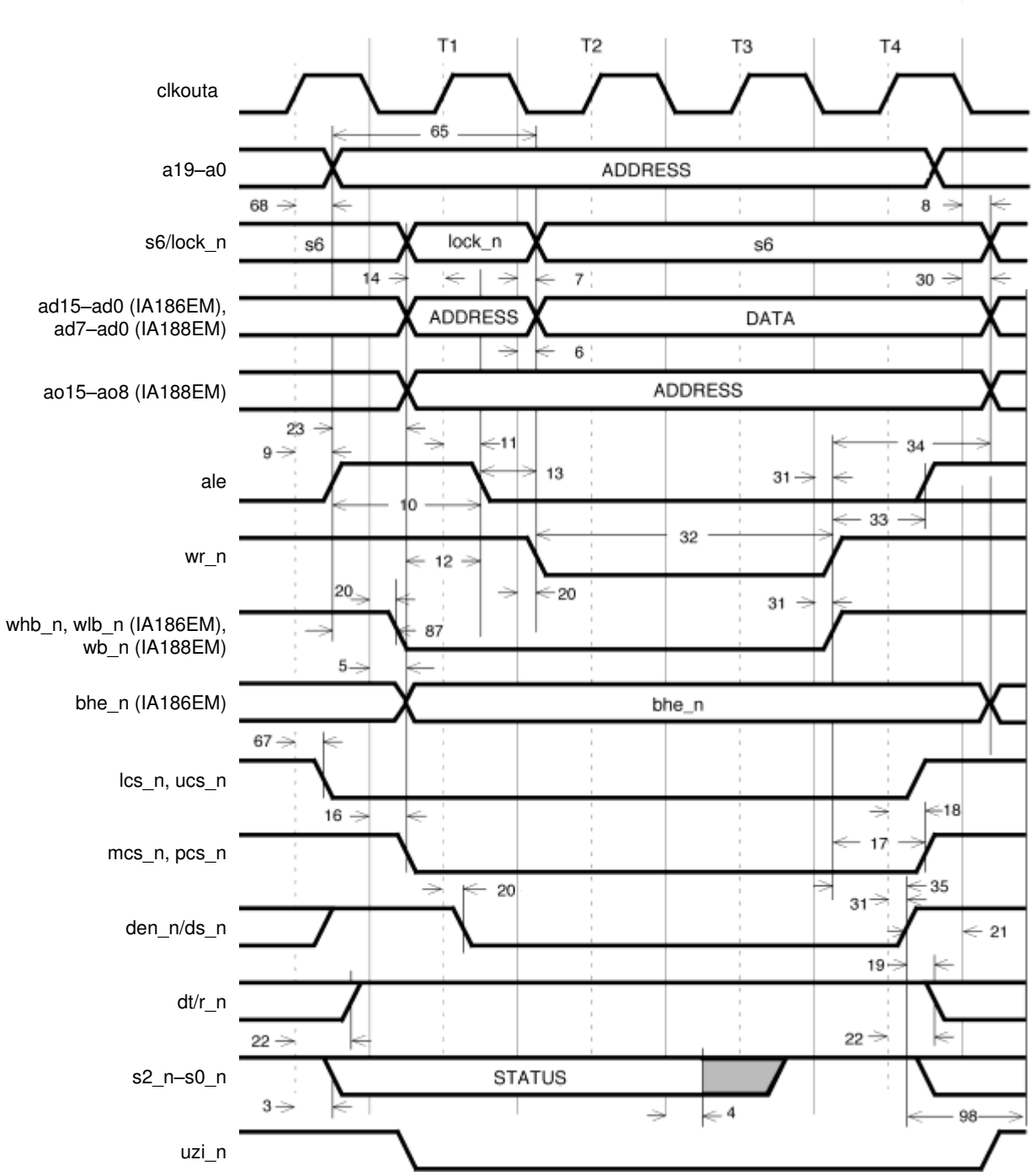



**Figure 13. Write Cycle**

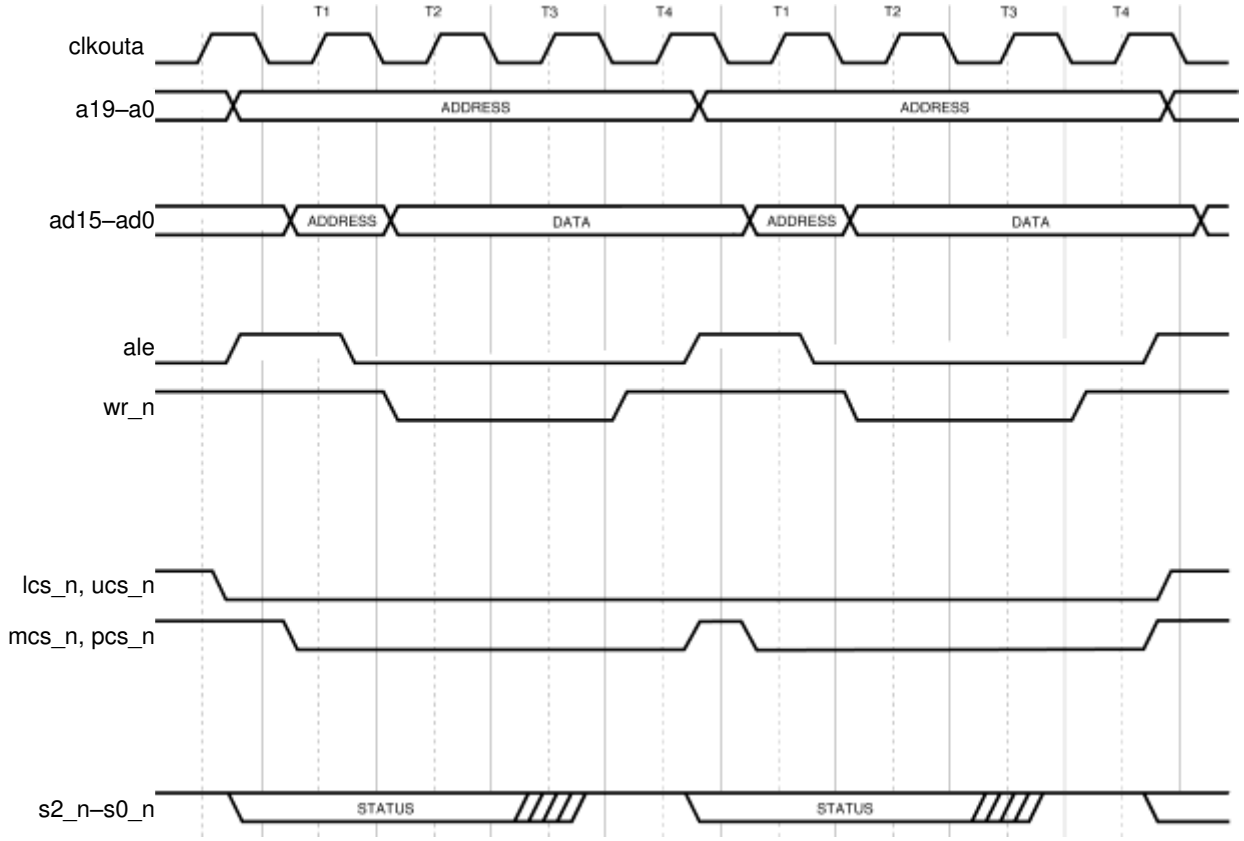



**Figure 14. Multiple Write Cycles**

## Table 84. Write Cycle Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | – |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 3 | tCHSV | Status Active Delay | 0 | 6 |
| 4 | tCLSH | Status Inactive Delay | 0 | 6 |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 6 | tCLAX | Address Hold | 0 | 12 |
| 7 | tCLDV | Data Valid Delay | 0 | 12 |
| 8 | tCHDX | Status Hold Time | 0 | – |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| 12 | tAVLL | *ad* Address Valid to ale Low | tCLCH | – |
| 13 | tLLAX | *ad* Address Hold from ale Inactive | tCHCL | – |
| 14 | tAVCH | *ad* Address Valid to Clock High | 0 | – |
| 16 | tCLCSV | mcs_n/pcs_n Inactive Delay | 0 | 12 |
| 17 | tCXCSX | mcs_n/pcs_n Hold from Command Inactive | tCLCH | – |
| 18 | tCHCSX | mcs_n/pcs_n Inactive Delay | 0 | 12 |
| 19 | tDXDL | den_n Inactive to dt/r_n Low | 0 | – |
| 20 | tCVCTV | Control Active Delay 1 | 0 | 10 |
| 22 | tCHCTV | Control Active Delay 2 | 0 | 9 |
| 23 | tLHAV | ale High to Address Valid | 7.5 | – |
| **Write Cycle Timing Responses** | | | | |
| 30 | tCLDOX | Data Hold Time | 0 | – |
| 31 | tCVCTX | Control Inactive Delay | 0 | 10 |
| 32 | tWLWH | wr_n Pulse Width | 2tCLCL | – |
| 33 | tWHLH | wr_n Inactive to ale High | tCLCH | – |
| 34 | tWHDX | Data Hold after wr_n | tCLCL | – |
| 35 | tWHDEX | wr_n Inactive to den_n Inactive | tCLCH | – |
| 65 | tAVWL | *a* Address Valid to wr_n Low | tCLCL + tCHCL | – |
| 67 | tCHCSV | clkouta High to lcs_n/usc_n Valid | 0 | 9 |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |
| 87 | tAVBL | *a* Address Valid to whb_n/wlb_n Low | tCHCL -1.5 | – |

[a]In nanoseconds.

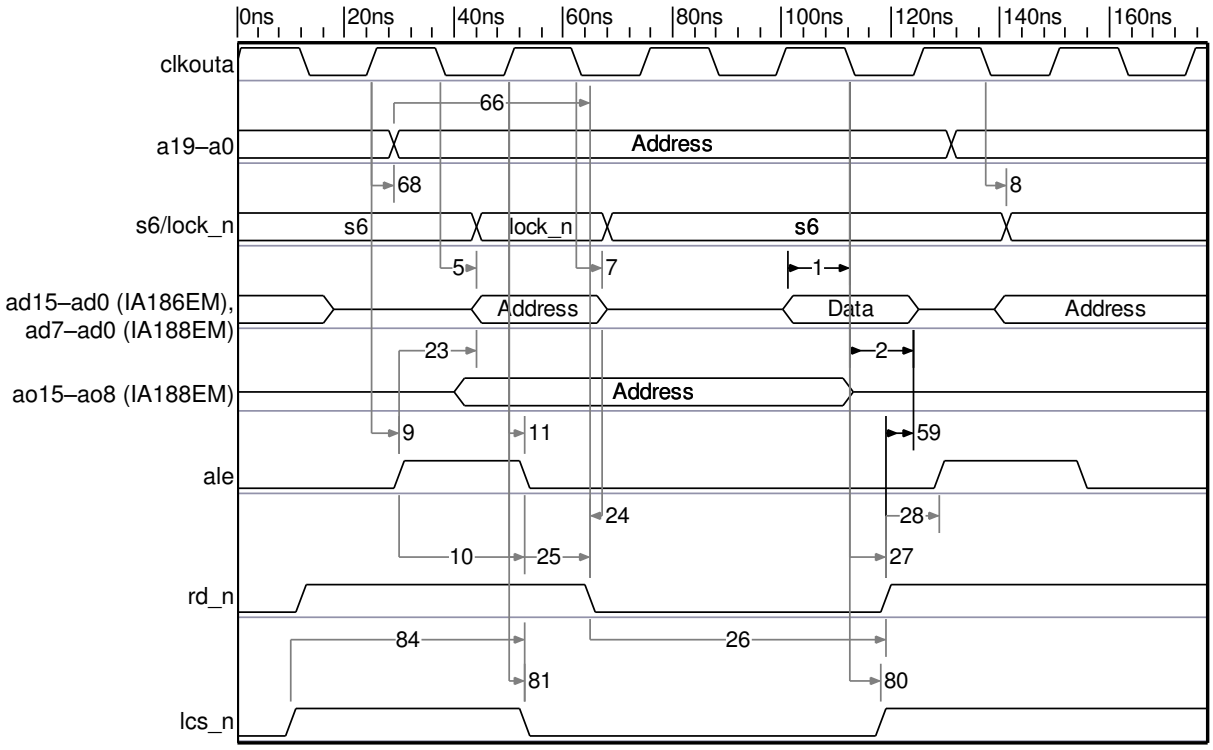



**Figure 15. PSRAM Read Cycle**

## Table 85. PSRAM Read Cycle Timing

| No. | Name | Comment | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | NLL |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 7 | tCLDV | Data Valid Delay | 0 | 12 |
| 8 | tCHDX | Status Hold Time | 0 | – |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCHCL-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| 23 | tLHAV | ale High to Address Valid | 7.5 | – |
| 80 | tCLCLX | lcs_n Inactive Delay | 0 | 9 |
| 81 | tCLCSL | lcs_n Active Delay | 0 | 9 |
| 84 | tLRLL | lcs_n Precharge Pulse Width | tCLCL+ tCLCH | – |
| **Read Cycle Timing Responses** | | | | |
| 24 | tAZRL | *ad* Address Float to rd_n Active | 0 | – |
| 25 | tCLRL | rd_n Active Delay | 0 | 10 |
| 26 | tRLRH | rd_n Pulse Width | tCLCL | – |
| 27 | tCLRH | rd_n Inactive Delay | 0 | 10 |
| 28 | tRHLH | rd_n Inactive to ale High | tCLCH | – |
| 59 | tRHDX | rd_n High to Data Hold on *ad* Bus | 0 | – |
| 66 | tAVRL | *a* Address Valid to rd_n Low | tCLCL+ tCHCL | – |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |

[a]In nanoseconds.

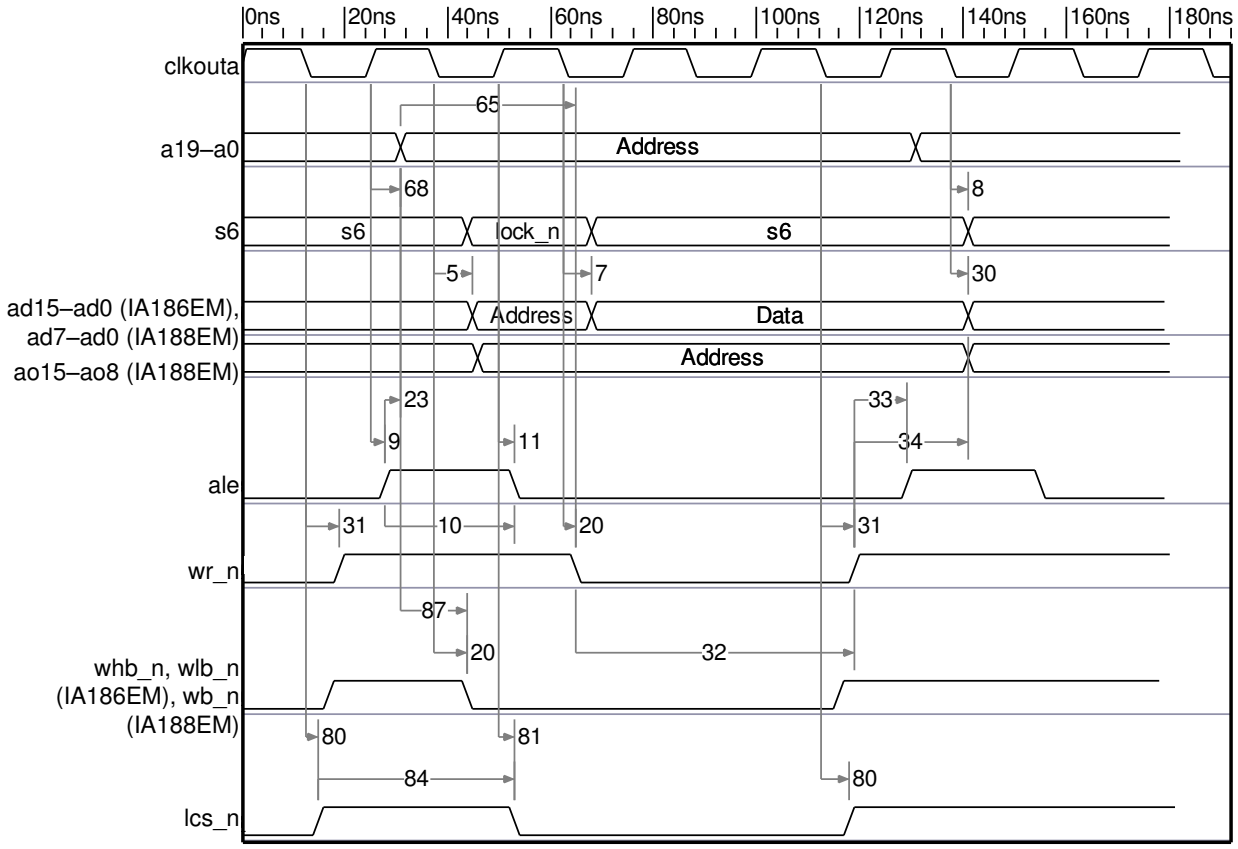



**Figure 16. PSRAM Write Cycle**

## Table 86. PSRAM Write Cycle Timing

| No. | Name | Comment | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | – |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 7 | tCLDV | Data Valid Delay | 0 | 12 |
| 8 | tCHDX | Status Hold Time | 0 | – |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | NULL | NULL |
| 20 | tCVCTV | Control Active Delay 1 | 0 | 10 |
| 23 | tLHAV | ale High to Address Valid | 7.5 | – |
| 80 | tCLCLX | lcs_n Inactive Delay | 0 | 9 |
| 81 | tCLCSL | lcs_n Active Delay | 0 | 9 |
| 84 | tLRLL | lcs_n Precharge Pulse Width | tCLCL+ tCLCH | – |
| Write Cycle Timing Responses | | | | |
| 30 | tCLDOX | Data Hold Time | 0 | – |
| 31 | tCVCTX | Control Inactive Delay | 0 | 10 |
| 32 | tWLWH | wr_n Pulse Width | 2tCLCL | – |
| 33 | tWHLH | wr_n Inactive to ale High | tCLCH | – |
| 34 | tWHDX | Data Hold after wr_n | tCLCL | – |
| 65 | tAVWL | *a* Address Valid to wr_n Low | tCLCL+ tCHCL | – |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |
| 87 | tAVBL | *a* Address Valid to whb_n/wlb_n Low | tCHCL -1.5 | – |

[a]In nanoseconds.

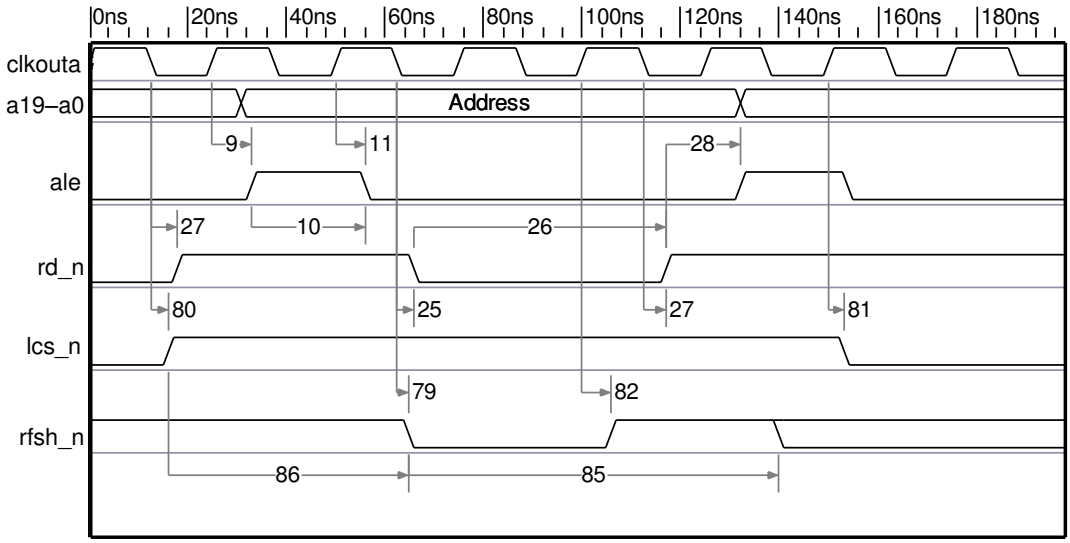



**Figure 17. PSRAM Refresh Cycle**

## Table 87. PSRAM Refresh Cycle

| No. | Name | Comment | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | – |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| **Read/Write Cycle Timing Responses** | | | | |
| 25 | tCLRL | rd_n Active Delay | 0 | 10 |
| 26 | tRLRH | rd_n Pulse Width | tCLCL | – |
| 27 | tCLRH | rd_n Inactive Delay | 0 | 10 |
| 28 | tRHLH | rd_n Inactive to ale High | tCLCH | – |
| 80 | tCLCLX | lcs_n Inactive Delay | 0 | 9 |
| 81 | tCLCSL | lcs_n Active Delay | 0 | 9 |
| **Refresh Cycle Timing Responses** | | | | |
| 79 | tCHRFD | clkouta High to rfsh_n Valid | 0 | 12 |
| 82 | tCLRF | clkouta High to rfsh_n Invalid | 0 | 12 |
| 85 | tRFCY | rfsh_n Cycle Time | 6tCLCL | – |
| 86 | tLCRF | lcs_n Inactive to rfsh_n Active Delay | 2tCLCL | NULL |

[a]In nanoseconds.

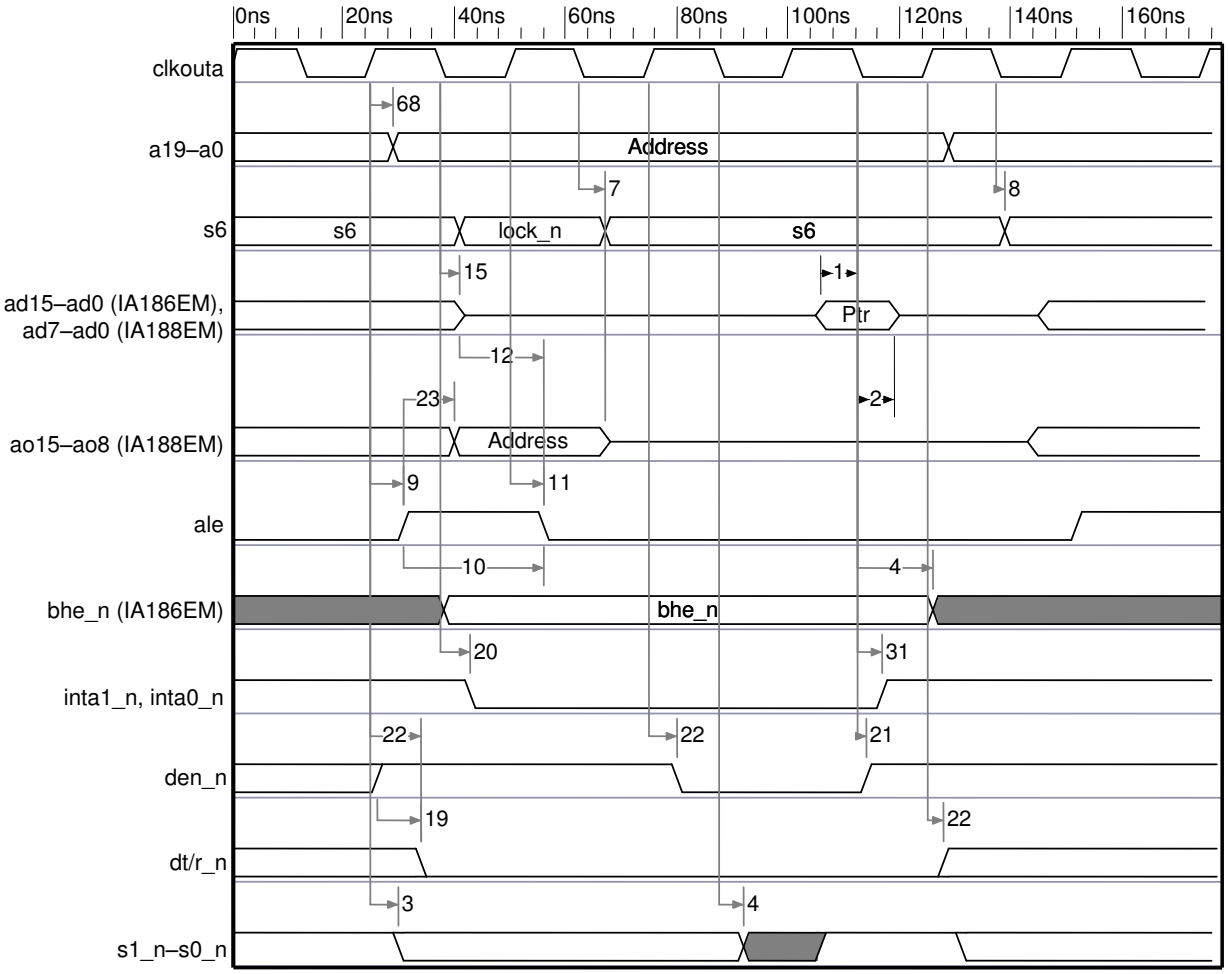



**Figure 18. Interrupt Acknowledge Cycle**

## Table 88. Interrupt Acknowledge Cycle Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Requirements** | | | | |
| 1 | tDVCL | Data in Setup | 10 | – |
| 2 | tCLDX | Data in Hold | 0 | – |
| **General Timing Responses** | | | | |
| 3 | tCHSV | Status Active Delay | 0 | 6 |
| 4 | tCLSH | Status Inactive Delay | 0 | 6 |
| 7 | tCLDV | Data Valid Delay | 0 | 12 |
| 8 | tCHDX | Status Hold Time | 0 | – |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| 12 | tAVLL | *ad* Address Valid to ale Low | tCLCH | – |
| 15 | tCLAZ | *ad* Address Float Delay | 0 | 12 |
| 19 | tDXDL | den_n Inactive to dt/r_n Low | 0 | – |
| 20 | tCVCTV | Control Active Delay 1 | 0 | 10 |
| 21 | tCVDEX | den_n Inactive Delay | 0 | 9 |
| 22 | tCHCTV | Control Active Delay 2 | 0 | 10 |
| 23 | tLHAV | ale High to Address Valid | 7.5 | – |
| 31 | tCVCTX | Control Inactive Delay | 0 | 10 |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |

[a]In nanoseconds.

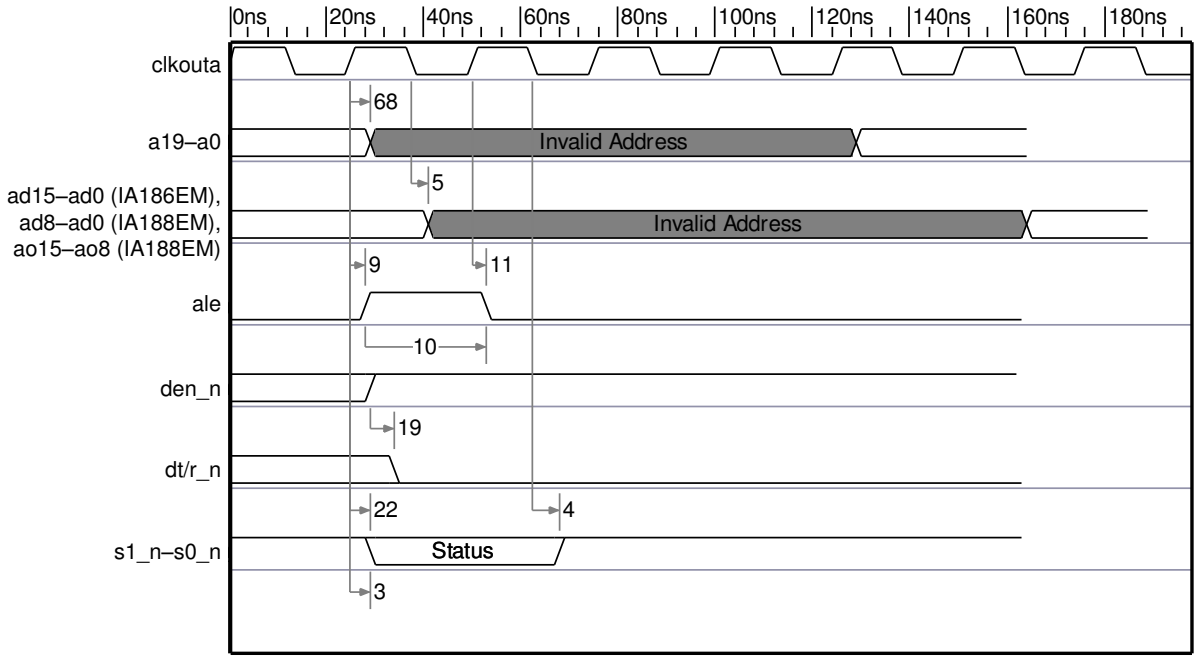



**Figure 19. Software Halt Cycle**

## Table 89. Software Halt Cycle Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **General Timing Responses** | | | | |
| 3 | tCHSV | Status Active Delay | 0 | 6 |
| 4 | tCLSH | Status Inactive Delay | 0 | 6 |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 9 | tCHLH | ale Active Delay | 0 | 8 |
| 10 | tLHLL | ale Width | tCLCH-5 | – |
| 11 | tCHLL | ale Inactive Delay | 0 | 8 |
| 19 | tDXDL | den_n Inactive to dt/r_n Low | 0 | – |
| 22 | tCHCTV | Control Active Delay 2 | 0 | 10 |
| 68 | tCHAV | clkouta High to *a* Address Valid | 0 | 8 |

[a]In nanoseconds.

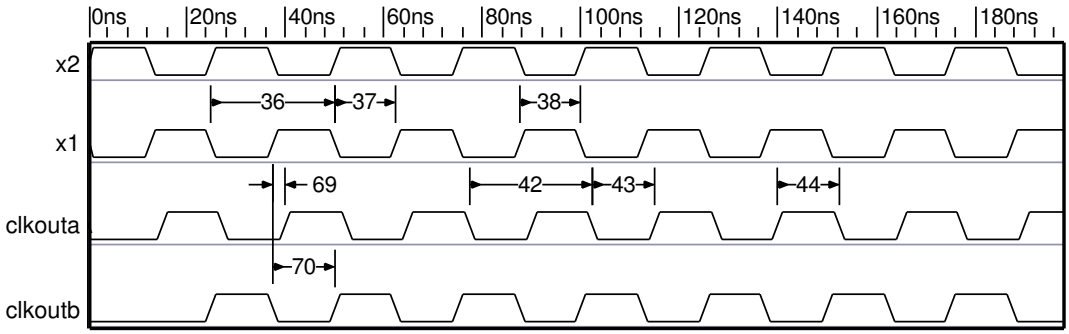



**Figure 20. Clock—Active Mode**

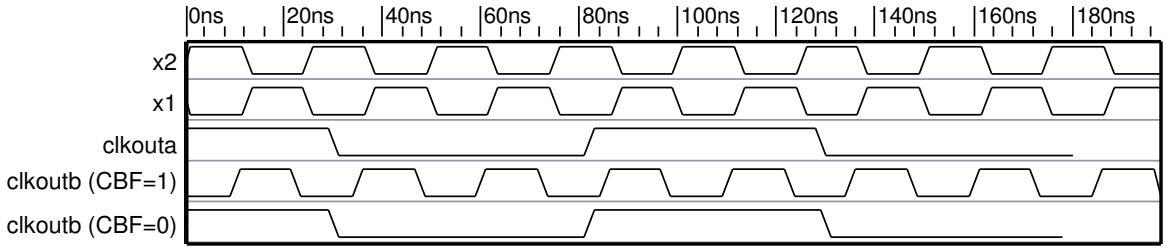



**Figure 21. Clock—Power-Save Mode**

## Table 90. Clock Timing

| No. | Name | Description | Min | Max | Units |
|---|---|---|---|---|---|
| **CLKIN Requirements** | | | | | |
| 36 | tCKIN | x1 Period | 25 | 66 | ns |
| 37 | tCLCK | x1 Low Time | 7.5 | – | ns |
| 38 | tCHCK | x1 High Time | 7.5 | – | ns |
| 39 | tCKHL | x1 Fall Time | – | 5 | ns |
| 40 | tCKLH | x1 Rise time | – | 5 | ns |
| **CLKOUT Timing** | | | | | |
| 42 | tCLCL | clkouta Period | 25 | – | ns |
| 43 | tCLCH | clkouta Low Time | tCLCL/2 | – | ns |
| 44 | tCHCL | clkouta High Time | tCLCL/2 | – | ns |
| 45 | tCH1CH2 | clkouta Rise Time | – | 3 | ns |
| 46 | tCL2CL1 | clkouta Fall Time | – | 3 | ns |
| 61 | tLOCK | Maximum PLL Lock Time | – | 0.5 | ms |
| 69 | tCICOA | x1 to clkouta Skew | – | 25 | ns |
| 70 | tCICOB | x1 to clkoutb Skew | – | 35 | ns |

[a]In nanoseconds.

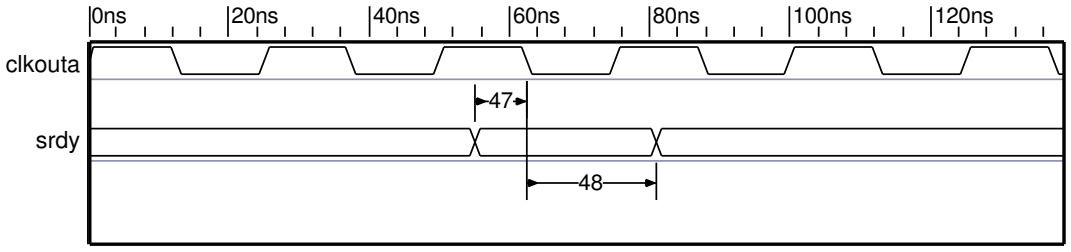



**Figure 22. srdy—Synchronous Ready**

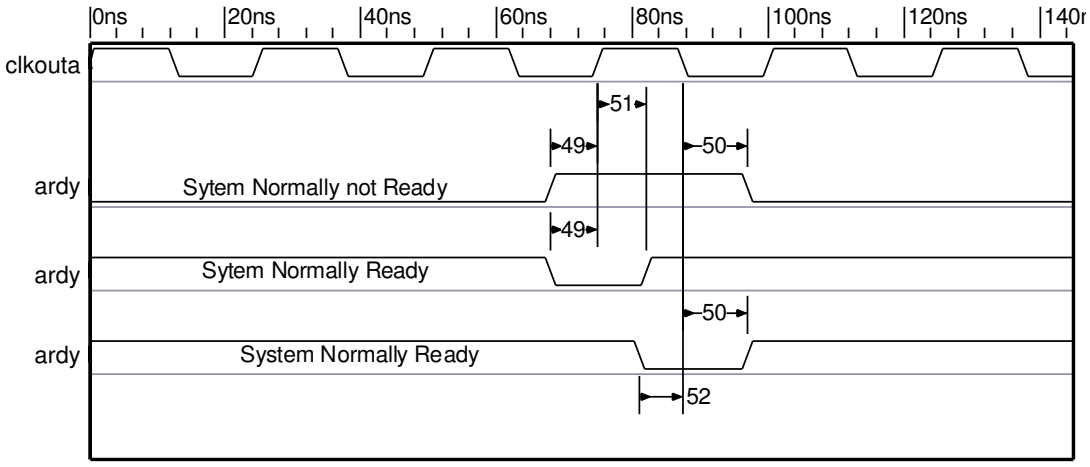



**Figure 23. ardy—Asynchronous Ready**

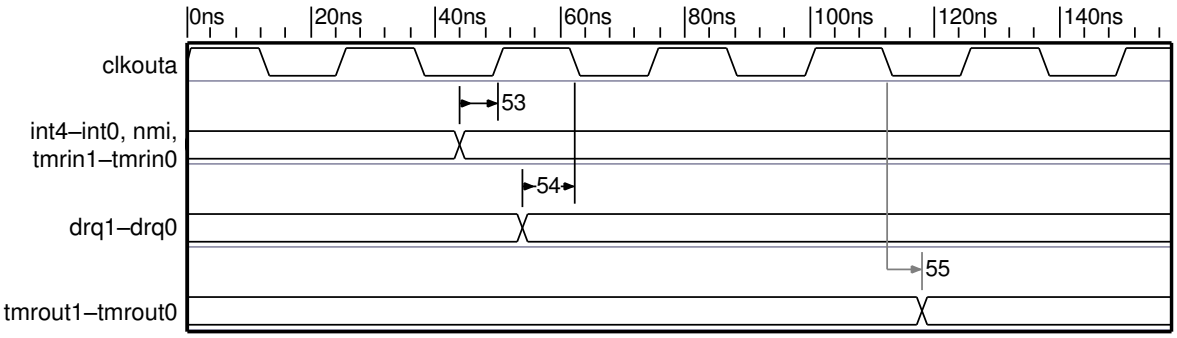



**Figure 24. Peripherals**

## Table 91. Ready and Peripheral Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **Ready and Peripheral Timing Requirements** | | | | |
| 47 | tSRYCL | srdy Transition Setup Time | 10 | – |
| 48 | tCLSRY | srdy Transition Hold Time | 3 | – |
| 49 | tARYCH | ardy Resolution Transition Setup Time | 9 | – |
| 50 | tCLARX | ardy Active Hold Time | 4 | – |
| 51 | tARYCHL | ardy Inactive Holding Time | 6 | – |
| 52 | tARYLCL | ardy Setup Time | 9 | – |
| 53 | tINVCH | Peripheral Setup Time | 10 | – |
| 54 | tINVCL | drq Setup Time | 10 | – |
| **Peripheral Timing Responses** | | | | |
| 55 | tCLTMV | Timer Output Delay | 0 | 12 |

[a]In nanoseconds.

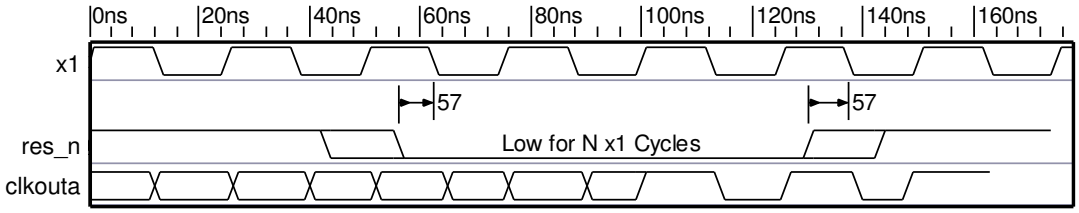



**Figure 25. Reset 1**

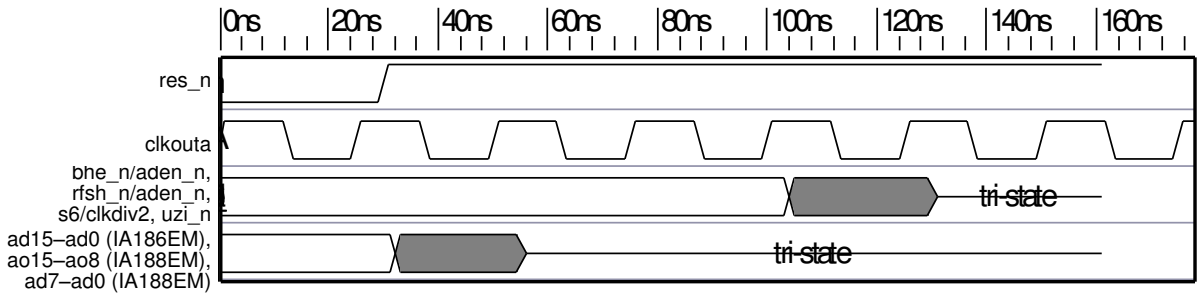



**Figure 26. Reset 2**

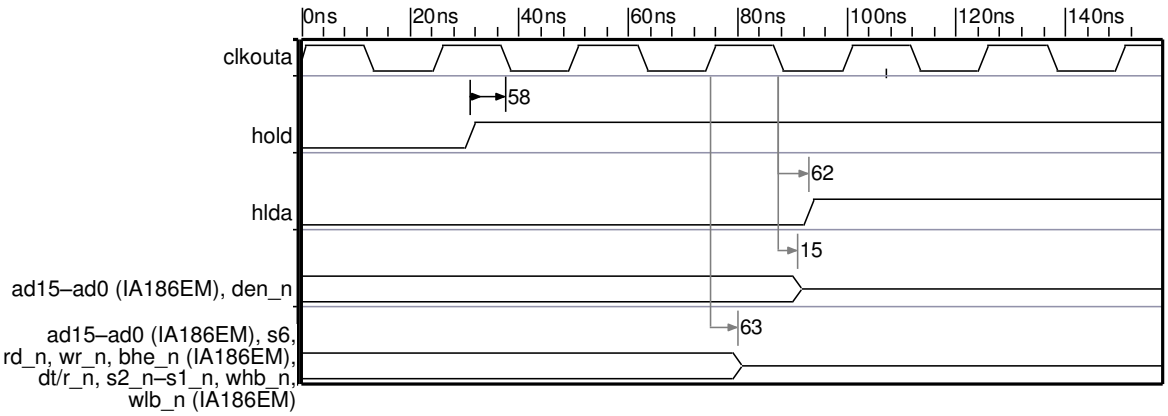



**Figure 27. Bus Hold Entering**

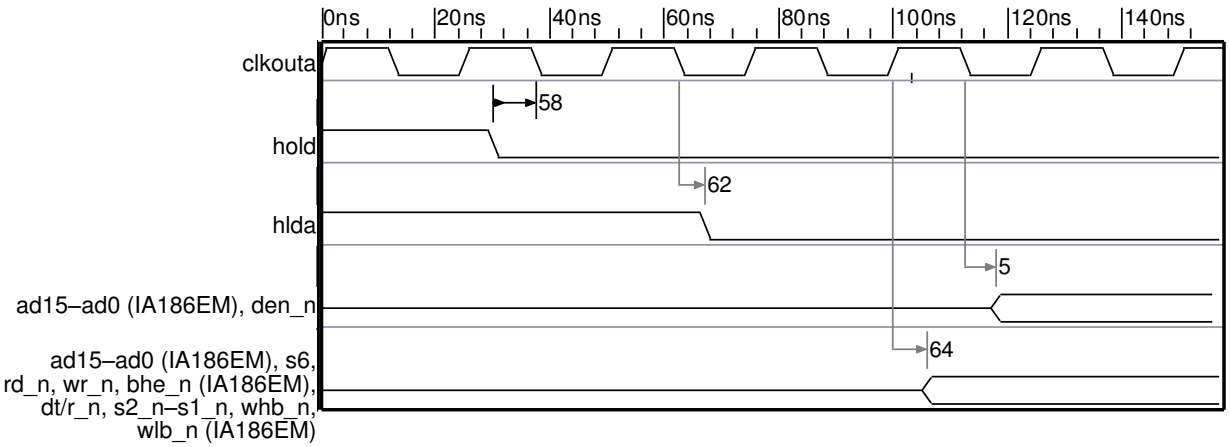



**Figure 28. Bus Hold Leaving**

## Table 92. Reset and Bus Hold Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **Reset and Bus Hold Timing Requirements** | | | | |
| 5 | tCLAV | *ad* Address Valid Delay | 0 | 12 |
| 15 | tCLAZ | *ad* Address Float Delay | tCLCH | – |
| 57 | tRESIN | res_n Setup Time | 10 | – |
| 58 | tHVCL | hld Setup Time | 10 | – |
| **Reset and Bus Hold Timing Responses** | | | | |
| 62 | tCLHAV | hlda Valid Delay | 0 | 7 |
| 63 | tCHCZ | Command Lines Float Delay | 0 | 12 |
| 64 | tCHCV | Command Lines Valid Delay (after Float) | 0 | 12 |

[a]In nanoseconds.

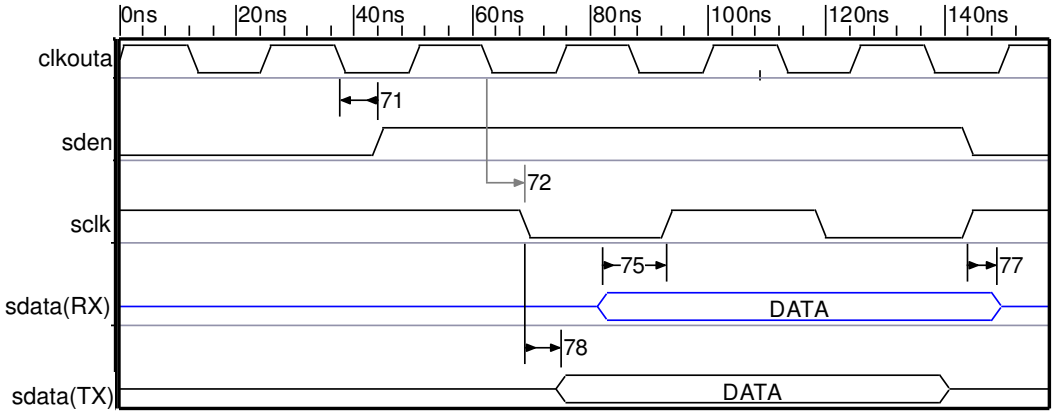



**Figure 29. Synchronous Serial Interface**

### Table 93. Synchronous Serial Interface Timing

| No. | Name | Description | Min[a] | Max[a] |
|---|---|---|---|---|
| **Synchronous Serial Port Timing Requirements** | | | | |
| 75 | tDVSH | Data Valid to sclk high | 10 | – |
| 77 | tSHDX | sclk High to SPI Data Hold | 3 | – |
| **Synchronous Serial Port Timing Responses** | | | | |
| 71 | tCLEV | clkouta Low to sden Valid | 0 | 12 |
| 72 | tCLSL | clkouta Low to sclk Low | 0 | 12 |
| 78 | tSLDV | sclk Low to Data Valid | 0 | 12 |

[a]In nanoseconds.

## 7. Instruction Set Summary Table

Table 94 summarizes each instruction. A key to abbreviations is presented at the end of the table.

### Table 94. Instruction Set Summary

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| AAA | ASCII adjust AL after add | 37 | – | – | 8 | 8 | U | – | – | – | U | U | R | U | R |
| AAD | ASCII adjust AX before divide. | D5 | 0A | – | 15 | 15 | U | – | – | – | R | R | U | R | U |
| AAM | ASCII adjust AL after multiply | D4 | 0A | – | 19 | 19 | U | – | – | – | R | R | U | R | U |
| AAS | ASCII adjust AL after subtract | 3F | – | – | 7 | 7 | U | – | – | – | U | U | R | U | R |
| ADC | Add imm8 to AL with carry | 14 | ib | – | 3 | 3 | R | – | – | – | R | R | R | R | R |
| | Add imm16 to AX with carry | 15 | iw | – | 4 | 4 | | | | | | | | | |
| | Add imm8 to r/m8 with carry | 80 | /2 ib | – | 4/16 | 4/16 | | | | | | | | | |
| | Add imm16 to r/m16 with carry | 81 | /2 iw | – | 4/16 | 4/20 | | | | | | | | | |
| | Add sign extended imm8 to r/m16 with carry | 83 | /2 ib | – | 4/16 | 4/20 | | | | | | | | | |
| | Add byte reg to r/m8 with carry | 10 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Add word reg to r/m16 with carry | 11 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | Add r/m8 to byte reg with carry | 12 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Add r/m16 to word reg with carry | 13 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| ADD | Add imm8 to AL | 04 | ib | – | 3 | 3 | R | – | – | – | R | R | R | R | R |
| | Add imm16 to AX | 05 | iw | – | 4 | 4 | | | | | | | | | |
| | Add imm8 to r/m8 | 80 | /0 ib | – | 4/16 | 4/16 | | | | | | | | | |
| | Add imm16 to r/m16 | 81 | /0 iw | – | 4/16 | 4/20 | | | | | | | | | |
| | Add sign extended imm8 to r/m16 | 83 | /0 ib | – | 4/16 | 4/20 | | | | | | | | | |
| | Add byte reg. to r/m8 | 00 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Add word reg. to r/m16 | 01 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | Add r/m8 to byte reg | 02 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Add r/m16 to word reg | 03 | /r | – | 3/10 | 3/14 | | | | | | | | | |

## Table 94. Instruction Set Summary *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| AND | And imm8 with AL | 24 | ib | | 3 | 3 | 0 | – | – | – | R | R | U | R | 0 |
| | And imm16 with AX | 25 | iw | | 4 | 4 | | | | | | | | | |
| | And imm8 with r/m8 | 80 | /4 ib | | 4/16 | 4/16 | | | | | | | | | |
| | And imm16 with r/m16 | 81 | /4 iw | | 4/16 | 4/20 | | | | | | | | | |
| | And sign-extended imm8 with r/m16 | 83 | /4 ib | | 4/16 | 4/20 | | | | | | | | | |
| | And byte reg. with r/m8 | 20 | /r | | 3/10 | 3/10 | | | | | | | | | |
| | And word reg. with r/m16 | 21 | /r | | 3/10 | 3/14 | | | | | | | | | |
| | And r/m8 with byte reg | 22 | /r | | 3/10 | 3/10 | | | | | | | | | |
| | And r/m16 with word reg | 23 | /r | | 3/10 | 3/14 | | | | | | | | | |
| BOUND | Check array index against bounds | 62 | /r | – | 33–35 | 33–35 | – | – | – | – | – | – | – | – | – |
| CALL | Call near, disp relative to next instruction | E8 | cw | – | 15 | 19 | – | – | – | – | – | – | – | – | – |
| | Call near, reg indirect mem | FF | /2 | – | 13/19 | 17/27 | | | | | | | | | |
| | Call far to full address given | 9A | cd | – | 23 | 31 | | | | | | | | | |
| | Call far to address at m16:16 word | FF | /3 | – | 38 | 54 | | | | | | | | | |
| CBW | Convert byte integer to word | 98 | – | – | 2 | 2 | – | – | – | – | – | – | – | – | – |
| CLC | Clear carry flag | F8 | – | – | 2 | 2 | – | – | – | – | – | – | – | – | – |
| CLD | Clear direction flag | FC | – | – | 2 | 2 | – | 0 | – | – | – | – | – | – | – |
| CLI | Clear interrupt-enable flag | FA | – | – | 2 | 2 | – | – | 0 | – | – | – | – | – | – |
| CMC | Complement carry flag | F5 | – | – | 2 | 2 | – | – | – | – | – | – | – | – | R |
| CMP | Compare imm8 to AL | 3C | ib | | 3 | 3 | R | – | – | – | R | R | R | R | R |
| | Compare imm16 to AX | 3D | iw | – | 4 | 4 | | | | | | | | | |
| | Compare imm8 to r/m8 | 80 | /7 | ib | 3/10 | 3/10 | | | | | | | | | |
| | Compare imm16 to r/m16 | 81 | /7 | iw | 3/10 | 3/14 | | | | | | | | | |
| | Compare sign-extended imm8 to r/m16 | 83 | /7 | ib | 3/10 | 3/14 | | | | | | | | | |
| | Compare byte reg to r/m8 | 38 | /r | | 3/10 | 3/10 | | | | | | | | | |
| | Compare word reg to r/m16 | 39 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | Compare r/m8 to byte reg | 3A | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Compare r/m16 to word reg | 3B | /r | – | 3/10 | 3/14 | | | | | | | | | |
| CMPS | Compare byte ES:[DI] to byte segment:[SI] | A6 | – | – | 22 | 22 | R | – | – | – | R | R | R | R | R |
| | Compare word ES:[DI] to word segment:[SI] | A7 | – | – | 22 | 26 | | | | | | | | | |
| CMPSB | Compare byte ES:[DI] to byte DS:[SI] | A6 | – | – | 22 | 22 | R | – | – | – | R | R | R | R | R |
| CMPSW | Compare word ES:[DI] to word DS:[SI] | A7 | – | – | 22 | 26 | R | – | – | – | R | R | R | R | R |
| CS | CS segment reg override prefix | 2E | – | – | – | – | – | – | – | – | – | – | – | – | – |
| CWD | Convert word integer to double word | 99 | – | – | 4 | 4 | – | – | – | – | – | – | – | – | – |
| DAA | Decimal adjust AL after addition | 27 | – | – | 4 | 4 | U | – | – | – | R | R | R | R | R |
| DAS | Decimal adjust AL after subtraction | 2F | – | – | 4 | 4 | U | – | – | – | R | R | R | R | R |
| DEC | Subtract 1 from r/m8 | FE | /1 | – | 3/15 | 3/15 | R | – | – | – | R | R | R | R | R |
| | Subtract 1 from r/m16 | FF | /1 | – | 3/15 | 3/19 | | | | | | | | | |
| | Subtract 1 from word reg | 48+rw | | | 3 | 3 | | | | | | | | | |
| DIV | Divide unsigned numbers | F6 | mod 110 r/m | – | 29/35 | 29/35 | U | – | – | – | U | U | U | U | U |

**Table 94. Instruction Set Summary** *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| DS | DS segment override prefix | 3E | – | – | – | – | – | – | – | – | – | – | – | – | – |
| ENTER | Create stack frame for nested procedure | C8 | iw ib | – | 22+16 (n–1) | 26+20 (n–1) | – | – | – | – | – | – | – | – | – |
| | Create stack frame for non-nested procedure | C8 | iw 00 | – | 15 | 19 | | | | | | | | | |
| | Create stack frame for nested procedure | C8 | iw 01 | – | 25 | 29 | | | | | | | | | |
| ES | ES segment reg override prefix | 26 | – | – | – | – | – | – | – | – | – | – | – | – | – |
| ESC | Escape - takes a Trap 7 | D8 | /0 | – | – | – | – | – | 0 | 0 | – | – | – | – | – |
| | Escape - takes a Trap 7 | D9 | /1 | – | – | – | | | | | | | | | |
| | Escape - takes a Trap 7 | DA | /2 | – | – | – | | | | | | | | | |
| | Escape - takes a Trap 7 | DB | /3 | – | – | – | | | | | | | | | |
| | Escape - takes a Trap 7 | DC | /4 | – | – | – | | | | | | | | | |
| | Escape - takes a Trap 7 | DD | /5 | – | – | – | | | | | | | | | |
| | Escape - takes a Trap 7 | DE | /6 | – | – | – | | | | | | | | | |
| | Escape - takes a Trap 7 | DF | /7 | – | – | – | | | | | | | | | |
| HLT | Suspend instruction execution | F4 | – | – | 2 | 2 | – | – | – | – | – | – | – | – | – |
| IDIV | Divide Integers AL = AX/(r/m8); AH = remainder | F6 | /7 | – | 44–52 / 50–58 | 44–52 / 50–58 | U | – | – | – | U | U | U | U | U |
| | Divide Integers AX = DX:AX/(r/m16); DX = remainder | F7 | /7 | – | 53–61 / 59–67 | 53–61 / 63–71 | | | | | | | | | |
| IMUL | Multiply Integers AX=(r/m8)*Al | F6 | /5 | – | 25–28 / 31–34 | 25–28 / 31–34 | R | – | – | – | U | U | U | U | R |
| | Multiply Integers DX=(r/m16)*AX | F7 | /5 | – | 34–37 / 40–43 | 34–37 / 44–47 | | | | | | | | | |
| | Multiply Integers (word reg) = (r/m16)*(sign-ext. byte integer) | 6B | /r ib | – | 22–25 | 22–25 | | | | | | | | | |
| | Multiply Integers (word reg) = (word reg)*(sign-ext. byte integer) | 6B | /r ib | – | 22–25 | 22–25 | | | | | | | | | |
| | Multiply Integers (word reg) = (r/m16)*(sign-ext. byte integer) | 69 | /r iw | – | 29–32 | 29–32 | | | | | | | | | |
| | Multiply Integers (word reg) = (word reg)*(sign-ext. byte integer) | 69 | /r iw | – | 29–32 | 29–32 | | | | | | | | | |
| IN | Input byte from imm port to AL | E4 | ib | – | 10 | 10 | – | – | – | – | – | – | – | – | – |
| | Input word from imm port to AX | E5 | ib | – | 10 | 14 | | | | | | | | | |
| | Input byte from port in DX to AL | EC | | – | 8 | 8 | | | | | | | | | |
| | Input word from port in DX to AX | ED | | – | 8 | 12 | | | | | | | | | |
| INC | Increment r/m8 by 1 | FE | /0 | – | 3/15 | 3/15 | R | – | – | – | R | R | R | R | R |
| | Increment r/m16 by 1 | FF | /0 | – | 3/15 | 3/19 | | | | | | | | | |
| | Increment word reg by 1 | 40+rw | – | – | 3 | 3 | | | | | | | | | |
| INS | Input byte from port in DX to ES:[DI] | 6C | – | – | 14 | 14 | – | – | – | – | – | – | – | – | – |
| | Input word from port in DX to ES:[DI] | 6D | | | | | | | | | | | | | |
| INSB | Input byte from port in DX to ES:[DI] | 6C | | | | | | | | | | | | | |

## Table 94. Instruction Set Summary *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| INSW | Input word from port in DX to ES:[DI] | 6D | | | | | | | | | | | | | |
| INT 3 | Generate interrupt 3 (trap to debug) | CC | – | – | 45 | 45 | – | – | 0 | 0 | – | – | – | – | – |
| INT | Generate type of interrupt specified by imm8 | CD | ib | – | 47 | 47 | | | | | | | | | |
| INTO | Generate interrupt 4 if Overflow Flag (O) is 1 | CE | – | – | 48, 4 | 48, 4 | | | | | | | | | |
| IRET | Interrupt return | CF | – | – | 28 | 28 | Restores value of flags reg that was stored on the stack when the interrupt was taken. | | | | | | | | |
| JA | Jump short if above (C & Z = 0) | 77 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNBE | Jump short if not below or equal | | | | | | | | | | | | | | |
| JAE | Jump short if above or equal (C=0) | 73 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNB | Jump short if not below (C=0) | | | | | | | | | | | | | | |
| JNC | Jump short if not carry (C=0) | | | | | | | | | | | | | | |
| JB | Jump short if below (C=1) | 72 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JC | Jump short if carry (C=1) | | | | | | | | | | | | | | |
| JNAE | Jump short if not above or equal (C=1) | | | | | | | | | | | | | | |
| JBE | Jump short if below or equal (C & Z = 0) | 76 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNA | Jump short if not above (C & Z = 0) | | | | | | | | | | | | | | |
| JCXZ | Jump short if CX reg is 0 | E3 | cb | – | 15,5 | 15,5 | – | – | – | – | – | – | – | – | – |
| JE | Jump short if equal (Z=1) | 74 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JZ | Jump short if 0 (Z=1) | | | | | | | | | | | | | | |
| JG | Jump short if greater (Z & S = O) | 7F | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNLE | Jump short if not less or equal (Z & S = O) | | | | | | | | | | | | | | |
| JGE | Jump short if greater or equal (S=O) | 7D | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNL | Jump short if not less (S = O) | | | | | | | | | | | | | | |
| JLE | Jump short if less or equal (Z & S = O) | 7E | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNG | Jump short if not greater (Z & S = O) | | | | | | | | | | | | | | |
| JMP | Jump short direct, disp relative to next instruction | EB | cb | – | 14 | 14 | – | – | – | – | – | – | – | – | – |
| | Jump near direct, disp relative to next instruction | E9 | cw | – | 14 | 14 | | | | | | | | | |
| | Jump near indirect | FF | /4 | – | 11/17 | 11/21 | | | | | | | | | |
| | Jump far direct to doubleword imm address | EA | cd | – | 14 | 14 | | | | | | | | | |
| | Jump m16: 16 indirect and far | FF | /5 | – | 26 | 34 | | | | | | | | | |
| JNE | Jump short if not equal (Z=0) | 75 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNZ | Jump short if not zero (Z=0) | | | | | | | | | | | | | | |
| JNO | Jump short if not overflow (O=1) | 71 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JNP | Jump short if not parity (P=0) | 7B | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JPO | Jump short if parity odd (P=0) | | | | | | | | | | | | | | |
| JNS | Jump short if not sign (S=0) | 79 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JO | Jump short if overflow (O=1) | 70 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JP | Jump short if parity (P=1) | 7A | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| JPE | Jump short if parity (P=1) | | | | | | | | | | | | | | |

## Table 94. Instruction Set Summary *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| JS | Jump short if sign (S=1) | 78 | cb | – | 13, 4 | 13, 4 | – | – | – | – | – | – | – | – | – |
| LAHF | Load AH with low byte of flags reg | 9F | – | – | 2 | 2 | – | – | – | – | – | – | – | – | – |
| LDS | Load DS:r16 with segment offset from memory | C5 | /r | – | 18 | 26 | – | – | – | – | – | – | – | – | – |
| LEA | Load offset for m16 word in 16-bit reg | 8D | /r | – | 6 | 6 | – | – | – | – | – | – | – | – | – |
| LEAVE | Destroy procedure stack frame | C9 | – | – | 8 | 8 | – | – | – | – | – | – | – | – | – |
| LES | Load ES:r16 with segment offset from memory | C4 | /r | – | 18 | 26 | – | – | – | – | – | – | – | – | – |
| LOCK | Asserts lock_n during an instruction execution | F0 | – | – | 1 | 1 | – | – | – | – | – | – | – | – | – |
| LODS | Load byte segment :[SI] in AL | AC | – | – | 12 | 12 | – | – | – | – | – | – | – | – | – |
| | Load word segment :[SI] in AX | AD | | | 12 | 16 | | | | | | | | | |
| LODSB | Load byte DS:[SI] in AL | AC | | | 12 | 12 | | | | | | | | | |
| LODSW | Load word DS:[SI] in AX | AD | | | 12 | 16 | | | | | | | | | |
| LOOP | Decrement count; jump short if CX ≠ 0 | E2 | – | – | 16, 6 | 16, 6 | – | – | – | – | – | – | – | – | – |
| LOOPE | Decrement count; jump short if CX ≠ 0 and Z = 1 | E1 | cb | – | | | | | | | | | | | |
| LOOPZ | Decrement count; jump short if CX ≠ 0 and Z = 1 | | | | | | | | | | | | | | |
| LOOPNE | Decrement count; jump short if CX ≠ 0 and Z = 0 | E0 | cb | – | 16, 6 | 16, 6 | – | – | – | – | – | – | – | – | – |
| LOOPNZ | Decrement count; jump short if CX ≠ 0 and Z = 0 | | | | | | | | | | | | | | |
| MOV | Copy reg to r/m8 | 88 | /r | – | 2/12 | 2/12 | – | – | – | – | – | – | – | – | – |
| | Copy reg to r/m16 | 89 | /r | – | 2/12 | 2/16 | | | | | | | | | |
| | Copy r/m8 to reg | 8A | /r | – | 2/9 | 2/9 | | | | | | | | | |
| | Copy r/m16 to reg | 8B | /r | – | 2/9 | 2/13 | | | | | | | | | |
| | Copy segment reg to r/m16 | 8C | /sr | – | 2/11 | 2/15 | | | | | | | | | |
| | Copy r/m16 to segment reg | 8E | /sr | – | 2/9 | 2/13 | | | | | | | | | |
| | Copy byte at segment offset to AL | A0 | – | – | 8 | 8 | | | | | | | | | |
| | Copy word at segment offset to AX | A1 | – | – | 8 | 12 | | | | | | | | | |
| | Copy AL to byte at segment offset | A2 | – | – | 9 | 9 | | | | | | | | | |
| | Copy AX to word at segment offset | A3 | – | – | 9 | 13 | | | | | | | | | |
| | Copy imm8 to reg | B0+rb | – | – | 3 | 3 | | | | | | | | | |
| | Copy imm16 to reg | B8+rw | | | 3 | 4 | | | | | | | | | |
| | Copy imm8 to r/m8 | C6 | /0 | – | 12 | 12 | | | | | | | | | |
| | Copy imm16 to r/m16 | C7 | /0 | – | 12 | 13 | | | | | | | | | |
| MOVS | Copy byte segment [SI] to ES:[DI] | A4 | – | – | 14 | 14 | – | – | – | – | – | – | – | – | – |
| | Copy word segment [SI] to ES:[DI] | A5 | – | – | 14 | 18 | | | | | | | | | |
| MOVSB | Copy byte DS:[SI] to ES:[DI] | A4 | – | – | 14 | 14 | | | | | | | | | |
| MOVSW | Copy word DS:[SI] to ES:[DI] | A5 | – | – | 14 | 18 | | | | | | | | | |

**Table 94. Instruction Set Summary** *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| MUL | AX = (r/m8) * AL | F6 | /4 | – | 26–28 / 32–34 | 26–28 / 32–34 | R | – | – | – | – | – | – | – | R |
| | DX::AX = (r/m16) * AX | F7 | /4 | | 35–37 / 41–43 | 35–37 / 45–47 | | | | | | | | | |
| NEG | Perform 2's complement negation of r/m8 | F6 | /3 | – | 3/10 | 3/10 | R | – | – | – | R | R | R | R | R |
| | Perform 2's complement negation of r/m16 | F7 | /3 | – | 3/10 | 3/14 | | | | | | | | | |
| NOP | Perform no operation | 90 | – | – | 3 | 3 | – | – | – | – | – | – | – | – | – |
| NOT | Complement each bit in r/m8 | F6 | /2 | – | 3/10 | 3/10 | – | – | – | – | – | – | – | – | – |
| | Complement each bit in r/m16 | F7 | /2 | | 3/10 | 3/14 | | | | | | | | | |
| OR | OR imm8 with AL | 0C | ib | – | 3 | 3 | 0 | – | – | – | R | R | U | R | 0 |
| | OR imm16 with AX | 0D | iw | | 4 | 4 | | | | | | | | | |
| | OR imm8 with r/m8 | 80 | /1 ib | – | 4/16 | 4/16 | | | | | | | | | |
| | OR imm16 with r/m16 | 81 | /1 iw | – | 4/16 | 4/20 | | | | | | | | | |
| | OR imm8 with r/m16 | 83 | /1 ib | – | 4/16 | 4/20 | | | | | | | | | |
| | OR byte reg with r/m8 | 08 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | OR word reg with r/m16 | 09 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | OR r/m8 with byte reg | 0A | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | OR r/m16 with word reg | 0B | /r | – | 3/10 | 3/14 | | | | | | | | | |
| OUT | Output AL to imm port | E6 | ib | – | 9 | 9 | – | – | – | – | – | – | – | – | – |
| | Output AX to imm port | E7 | ib | – | 9 | 13 | | | | | | | | | |
| | Output AL to port in DX | EE | – | – | 7 | 7 | | | | | | | | | |
| | Output AX to port in DX | EF | – | – | 7 | 11 | | | | | | | | | |
| OUTS | Output byte DS:[SI] to port in DX | 6E | – | – | 14 | 14 | – | – | – | – | – | – | – | – | – |
| | Output word DS:[SI] to port in DX | 6F | – | – | | | | | | | | | | | |
| OUTSB | Output byte DS:[SI] to port in DX | 6E | – | – | | | | | | | | | | | |
| OUTSW | Output word DS:[SI] to port in DX | 6F | – | – | | | | | | | | | | | |
| POP | Pop top word of stack into memory word | 8F | /0 | – | 20 | 24 | – | – | – | – | – | – | – | – | – |
| | Pop top word of stack into word reg | 58+rw | – | – | 10 | 14 | | | | | | | | | |
| | Pop top word of stack into DS | 1F | – | – | 8 | 12 | | | | | | | | | |
| | Pop top word of stack into ES | 07 | – | – | | | | | | | | | | | |
| | Pop top word of stack into SS | 17 | – | – | | | | | | | | | | | |
| POPA | Pop DI, SI, BP, BX, DX, CX, & AX | 61 | – | – | 51 | 83 | Values in word at top of stack are copied into FLAGS reg bits. | | | | | | | | |
| POPF | Pop top word of stack into Processor Status Flags reg | 9D | – | – | 8 | 12 | | | | | | | | | |
| PUSH | Push memory word onto stack | FF | /6 | – | 16 | 20 | – | – | – | – | – | – | – | – | – |
| | Push reg word onto stack | 50+rw | – | – | 10 | 14 | | | | | | | | | |
| | Push sign-extended imm8 onto stack | 6A | – | – | 10 | 14 | | | | | | | | | |
| | Push imm16 onto stack | 68 | – | – | 10 | 14 | | | | | | | | | |
| | Push CS onto stack | 0E | – | – | 9 | 13 | | | | | | | | | |
| | Push SS onto stack | 16 | – | – | 9 | 13 | | | | | | | | | |
| | Push DS onto stack | 1E | – | – | 9 | 13 | | | | | | | | | |
| | Push ES onto stack | 06 | – | – | 9 | 13 | | | | | | | | | |
| PUSHA | Push AX, CX, DX, BX, original SP, BP, SI, and DI | 60 | – | – | 36 | 68 | – | – | – | – | – | – | – | – | – |

**Table 94. Instruction Set Summary** *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| PUSHF | Push Processor Status Flags reg | 9C | – | – | 9 | 13 | – | – | – | – | – | – | – | – | – |
| RCL | Rotate 9 bits of C and r/m8 left once | D0 | /2 | – | 2/15 | 2/15 | – | – | – | – | – | – | – | – | – |
| | Rotate 9 bits of C and r/m8 left CL times | D2 | /2 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 9 bits of C and r/m8 left imm8 times | C0 | /2 ib | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 17 bits of C and r/m16 left once | D1 | /2 | – | 2/15 | 2/15 | | | | | | | | | |
| | Rotate 17 bits of C and r/m16 left CL times | D3 | /2 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 17 bits of C and r/m16 left imm8 times | C1 | /2 ib | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| RCR | Rotate 9 bits of C and r/m8 right once | D0 | /3 | – | 2/15 | 2/15 | – | – | – | – | – | – | – | – | – |
| | Rotate 9 bits of C and r/m8 right CL times | D2 | /3 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 9 bits of C and r/m8 right imm8 times | C0 | /3 ib | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 17 bits of C and r/m16 right once | D1 | /3 | – | 2/15 | 2/15 | | | | | | | | | |
| | Rotate 17 bits of C and r/m16 right CL times | D3 | /3 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 17 bits of C and r/m16 right imm8 times | 75 | /3 ib | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| REP INS | Input CX bytes from port in DX to ES:[DI] | F3 | 6C | – | 8+8n | 8+8n | – | – | – | – | – | – | – | – | – |
| | Input CX bytes from port in DX to ES:[DI] | F3 | 6D | – | 8+8n | 12+8n | | | | | | | | | |
| REP LODS | Load CX bytes from segment :[SI] in AL | F3 | AC | – | 6+11n | 6+11n | – | – | – | – | – | – | – | – | – |
| | Load CX words from segment :[SI] in AX | F3 | AD | – | 6+11n | 10+ 11n | | | | | | | | | |
| REP MOVS | Copy CX bytes from segments :[SI] to ES:[DI] | F3 | A4 | – | 8+8n | 8+8n | – | – | – | – | – | – | – | – | – |
| | Copy CX words from segments :[SI] to ES:[DI] | F3 | A5 | – | 8+8n | 12+8n | | | | | | | | | |
| REP OUTS | Output CX bytes from DS:[SI] to port in DX | F3 | 6E | – | 8+8n | 8+8n | – | – | – | – | – | – | – | – | – |
| | Output CX bytes from DS:[SI] to port in DX | F3 | 6F | – | 8+8n | 12+8n | | | | | | | | | |
| REP STOS | Fill CX bytes at ES:[DI] with AL | F3 | AA | – | 8+8n | 8+8n | – | – | – | – | – | – | – | – | – |
| | Fill CX words at ES:[DI] with AL | F3 | AB | – | 8+8n | 12+8n | | | | | | | | | |
| REPE CMPS | Find non-matching bytes in ES:[DI] and segment :[SI] | F3 | A6 | – | 5+22n | 5+22n | – | – | – | – | – | – | – | – | – |
| | Find non-matching words in ES:[DI] and segment :[SI] | F3 | A7 | – | 5+22n | 9+22n | | | | | | | | | |
| REPE SCAS | Find non-AL byte starting at ES:[DI] | F3 | AE | – | 5+15n | 5+15n | | | | | | | | | |
| | Find non-AX word starting at ES:[DI] | F3 | AF | – | 5+15n | 9+15n | | | | | | | | | |
| REPZ CMPS | Find non-matching bytes in ES:DI and segment :[SI] | F3 | A6 | – | 5+22n | 5+22n | | | | | | | | | |
| | Find non-matching words in ES:DI and segment :[SI] | F3 | A7 | – | 5+22n | 9+22n | | | | | | | | | |

**Table 94. Instruction Set Summary** *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| REPZ SCAS | Find non-AL byte starting at ES:DI | F3 | AE | – | 5+15n | 5+15n | | | | | | | | | |
| | Find non-AX word starting at ES:DI | F3 | AF | – | 5+15n | 9+15n | | | | | | | | | |
| REPNE CMPS | Find matching bytes in ES:[DI] and segment :[SI] | F2 | A6 | – | 5+22n | 5+22n | – | – | – | – | – | – | – | – | – |
| | Find matching words in ES:[DI] and segment :[SI] | F2 | A7 | – | 5+22n | 9+22n | | | | | | | | | |
| REPNZ CMPS | Find AL byte starting at ES:[DI] | F2 | A6 | – | 5+22n | 5+22n | | | | | | | | | |
| | Find AX word starting at ES:[DI] | F2 | A7 | – | 5+22n | 9+22n | | | | | | | | | |
| REPNE SCAS | Find matching bytes in ES:DI and segment :[SI] | F2 | AE | – | 5+15n | 5+15n | | | | | | | | | |
| | Find matching words in ES:DI and segment :[SI] | F2 | AF | – | 5+15n | 9+15n | | | | | | | | | |
| REPNZ SCAS | Find AL byte starting at ES:DI | F2 | AE | – | 5+15n | 5+15n | | | | | | | | | |
| | Find AX word starting at ES:DI | F2 | AF | – | 5+15n | 9+15n | | | | | | | | | |
| RET | Return near to calling procedure | C3 | | | 16 | 20 | – | – | – | – | – | – | – | – | – |
| | Return far to calling procedure | CB | data low | data high | 22 | 30 | | | | | | | | | |
| | Return near; pop imm16 parameters | C2 | | | 18 | 22 | | | | | | | | | |
| | Return far; pop imm16 parameters | CA | data low | data high | 25 | 33 | | | | | | | | | |
| ROL | Rotate 8 bits of r/m8 left once | D0 | /0 | – | 2/15 | 2/15 | U | – | – | – | – | – | – | – | R |
| | Rotate 8 bits or r/m8 left CL times | D2 | /0 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 8 bits or r/m8 left imm8 times | C0 | /0 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 16 bits of r/m8 left once | D1 | /0 | – | 2/15 | 2/15 | | | | | | | | | |
| ROL | Rotate 16 bits or r/m8 left CL times | D3 | /0 | – | 5+n/ 17+n | 5+n/ 17+n | U | – | – | – | – | – | – | – | R |
| | Rotate 16 bits or r/m8 left imm8 times | C1 | /0 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| ROR | Rotate 8 bits of r/m8 right once | D0 | /1 | – | 2/15 | 2/15 | U | – | – | – | – | – | – | – | R |
| | Rotate 8 bits or r/m8 right CL times | D2 | /1 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 8 bits or r/m8 right imm8 times | C0 | /1 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 16 bits of r/m8 right once | D1 | /1 | – | 2/15 | 2/15 | | | | | | | | | |
| | Rotate 16 bits or r/m8 right CL times | D3 | /1 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Rotate 16 bits or r/m8 right imm8 times | C1 | /1 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| SAHF | Show AH in low byte of the Status Flags reg | 9E | – | – | 3 | 3 | – | – | – | – | R | R | R | R | R |

**Table 94. Instruction Set Summary** *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| SAL/SHL | Multiply r/m8 by 2, once | D0 | /4 | – | 2/15 | 2/15 | U | – | – | – | – | R | R | R | R |
| | Multiply r/m8 by 2, CL times | D2 | /4 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m8 by 2, imm8 times | C0 | /4 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m16 by 2, once | D1 | /4 | – | 2/15 | 2/15 | | | | | | | | | |
| | Multiply r/m16 by 2, CL times | D3 | /4 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m16 by 2, imm8 times | C1 | /4 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m8 by 2, once | D0 | /4 | – | 2/15 | 2/15 | | | | | | | | | |
| | Multiply r/m8 by 2, CL times | D2 | /4 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m8 by 2, imm8 times | C0 | /4 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m16 by 2, once | D1 | /4 | – | 2/15 | 2/15 | | | | | | | | | |
| | Multiply r/m16 by 2, CL times | D3 | /4 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Multiply r/m16 by 2, imm8 times | C1 | /4 ib | data8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| SAR | Perform a signed division of r/m8 by 2, once | D0 | /7 | – | 2/15 | 2/15 | U | – | – | – | R | R | U | R | R |
| | Perform a signed division of r/m8 by 2, CL times | D2 | /7 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Perform a signed division of r/m8 by 2, imm8 times | C0 | /7 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Perform a signed division of r/m16 by 2, once | D1 | /7 | – | 2/15 | 2/15 | | | | | | | | | |
| | Perform a signed division of r/m16 by 2, Cl times | D3 | /7 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Perform a signed division of r/m16 by 2, imm8 times | C1 | /7 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| SBB | Subtract imm8 from Al with borrow | 1C | ib | – | 3 | 3 | R | – | – | – | R | R | R | R | R |
| | Subtract imm16 from AX with borrow | 1D | iw | data 8 | 4 | 4 | | | | | | | | | |
| | Subtract imm8 from r/m8 with borrow | 80 | /3 ib | – | 4/16 | 4/16 | | | | | | | | | |
| | Subtract imm16 from r/m16 with borrow | 81 | /3 iw | – | 4/16 | 4/20 | | | | | | | | | |
| | Subtract sign-extended imm8 from r/m16 with borrow | 83 | /3 ib | – | 4/16 | 4/20 | | | | | | | | | |
| | Subtract byte reg from r/m8 with borrow | 18 | /r | data8 | 3/10 | 3/10 | | | | | | | | | |
| | Subtract word reg from r/m16 with borrow | 19 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | Subtract r/m8 from r/m8 with borrow | 1A | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Subtract r/m8 reg from byte with borrow | 1B | /r | data 8 | 3/10 | 3/14 | | | | | | | | | |
| SCAS | Compare byte AL to ES:[DI]; update DI | AE | – | – | 15 | 19 | R | – | – | – | R | R | R | R | R |
| | Compare word AL to ES:[DI]; update DI | AF | – | – | 15 | 19 | | | | | | | | | |

## Table 94. Instruction Set Summary *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| SCASB | Compare byte AL to ES:[DI]; update DI | AE | – | – | 15 | 19 | | | | | | | | | |
| SCASW | Compare word AL to ES:[DI]; update DI | AF | – | – | 15 | 19 | | | | | | | | | |
| SHR | Divide unsigned of r/m8 by 2, once | D0 | /7 | – | 2/15 | 2/15 | U | – | – | – | R | R | U | R | 0 |
| | Divide unsigned of r/m8 by 2, CL times | D2 | /7 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Divide unsigned of r/m8 by 2, imm8 times | C0 | /7 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Divide unsigned of r/m16 by 2, once | D1 | /7 | – | 2/15 | 2/15 | | | | | | | | | |
| | Divide unsigned of r/m16 by 2, CL times | D3 | /7 | – | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| | Divide unsigned of r/m16 by 2, imm8 times | C1 | /7 ib | data 8 | 5+n/ 17+n | 5+n/ 17+n | | | | | | | | | |
| SS | SS segment reg override prefix | 36 | – | – | – | – | – | – | – | – | – | – | – | – | – |
| STC | Set the Carry Flag to 1 | F9 | – | | 2 | 2 | – | – | – | – | – | – | – | – | 1 |
| STD | Set the Direction Flag so the source Index (SI) and/or the Destination Index (DI) regs will decrement during string instructions | FD | – | – | 2 | 2 | – | 1 | – | – | – | – | – | – | – |
| STI | Enable maskable interrupts after the next instruction | FB | – | – | 2 | 2 | – | – | 1 | – | – | – | – | – | – |
| STOS | Store AL in byte ES:[DI]; update DI | AA | – | – | 10 | 10 | – | – | – | – | – | – | – | – | – |
| | Store AX in word ES:[DI]; update DI | AB | – | – | 10 | 14 | | | | | | | | | |
| STOSB | Store AL in byte ES:[DI]; update DI | AA | – | – | 10 | 10 | | | | | | | | | |
| STOSW | Store AX in word ES:[DI]; update DI | AB | – | – | 10 | 14 | | | | | | | | | |
| SUB | Subtract imm8 from AL | 2C | ib | – | 3 | 3 | R | – | – | – | R | R | R | R | R |
| | Subtract imm16 from AX | 2D | iw | – | 4 | 4 | | | | | | | | | |
| | Subtract imm8 from r/m8 | 80 | /5 ib | – | 4/16 | 4/16 | | | | | | | | | |
| | Subtract imm16 from r/m16 | 81 | /5 iw | – | 4/16 | 4/20 | | | | | | | | | |
| | Subtract sign-extended imm8 from r/m16 | 83 | /5 ib | – | 4/16 | 4/20 | | | | | | | | | |
| | Subtract byte reg from r/m8 | 28 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Subtract word reg from r/m16 | 29 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | Subtract r/m8 from byte reg | 2A | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | Subtract r/m16 from word reg | 2B | /r | – | 3/10 | 3/14 | | | | | | | | | |
| TEST | AND imm8 with AL | A8 | ib | – | 3 | 3 | 0 | – | – | – | R | R | U | R | 0 |
| | AND imm16 with AX | A9 | iw | – | 4 | 4 | | | | | | | | | |
| | AND imm8 with r/m8 | F6 | /0 ib | data 8 | 4/10 | 4/10 | | | | | | | | | |
| | AND imm16 with r/m16 | F7 | /0 iw | – | 4/10 | 4/14 | | | | | | | | | |
| | AND byte reg with r/m8 | 84 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | AND word reg with r/m16 | 85 | /r | data 8 | 3/10 | 3/14 | | | | | | | | | |
| WAIT | Performs an NOP | 9B | – | – | – | – | – | – | – | – | – | – | – | – | – |

**Table 94. Instruction Set Summary** *(Continued)*

| Instruction | | Opcode - Hex | | | Clock Cycles | | Flags Affected | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mnemonic | Description | Byte 1 | Byte 2 | Bytes 3–6 | IA186EM | IA188EM | O | D | I | T | S | Z | A | P | C |
| XCHG | Exchange word reg with AX | 90 | – | – | 3 | 3 | – | – | – | – | – | – | – | – | – |
| | Exchange AX with word reg | +rw | – | – | 3 | 3 | | | | | | | | | |
| | Exchange byte reg with r/byte | 86 /r | – | – | 4/17 | 4/17 | | | | | | | | | |
| | Exchange r/m8 with byte reg | | – | – | 4/17 | 4/17 | | | | | | | | | |
| | Exchange word reg with r/m16 | 87 /r | – | – | 4/17 | 4/21 | | | | | | | | | |
| | Exchange r/m16 with word reg | | – | – | 4/17 | 4/21 | | | | | | | | | |
| XLAT | Set AL to memory byte segment :[BX+unsigned AL] | D7 | – | – | 11 | 15 | – | – | – | – | – | – | – | – | – |
| XLATB | Set AL to memory byte DS :[BX+unsigned AL] | D7 | – | – | 11 | 15 | | | | | | | | | |
| XOR | XOR imm8 with AL | 34 | ib | – | 3 | 3 | 0 | – | – | – | R | R | U | R | 0 |
| | XOR imm16 with AX | 35 | iw | – | 4 | 4 | | | | | | | | | |
| | XOR imm8 with r/m8 | 80 | /6 ib | – | 4/16 | 4/16 | | | | | | | | | |
| | XOR imm16 with r/m16 | 81 | /6 iw | – | 4/16 | 4/20 | | | | | | | | | |
| | XOR sign-extended imm8 with r/m16 | 83 | /6 ib | – | 4/16 | 4/20 | | | | | | | | | |
| | XOR byte reg with r/m8 | 30 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | XOR word reg with r/m16 | 31 | /r | – | 3/10 | 3/14 | | | | | | | | | |
| | XOR r/m8 with byte reg | 32 | /r | – | 3/10 | 3/10 | | | | | | | | | |
| | XOR r/m16 with word reg | 33 | /r | – | 3/10 | 3/14 | | | | | | | | | |

## 7.1 Key to Abbreviations Used in Instruction Set Summary Table

Abbreviations used in the Instruction Set Summary Table are explained below.

### 7.1.1 Operand Address Byte

The operand address byte is configured as shown below.

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| mod field | | aux field | | | r/m field | | |

### 7.1.2 Modifier Field

The modifier field is defined below.

| mod | Description |
|---|---|
| 11 | r/m is treated as a register field |
| 00 | DISP = 0, disp-low and disp-high are absent, address displacement is 0 |
| 01 | DISP = disp-low sign-extended to 16-bits, disp-high is absent |
| 10 | DISP = disp-high:disp-low |

### 7.1.3 Auxiliary Field

The Auxiliary Field is defined below.

| aux | If mod = 11 and word = 0 | If mod = 11 and word = 1 |
|---|---|---|
| 000 | AL | AX |
| 001 | CL | CX |
| 010 | DL | DX |
| 011 | BL | BX |
| 100 | AH | SP |
| 101 | CH | BP |
| 110 | DH | SI |
| 111 | BH | DI |

Note: When mod ≠ 11, depends on instruction.

### 7.1.4 r/m Field

The r/m field is defined below.

| r/m | Description |
|---|---|
| 000 | EA = (BX) + (SI) + DISP [where EA is the Effective Address] |
| 001 | EA = (BX) + (DI) + DISP |
| 010 | EA = (BP) + (SI) + DISP |
| 011 | EA = (BX) + (DI) + DISP |
| 100 | EA = (SI) + DISP |
| 101 | EA = (DI) + DISP |
| 110 | EA = (BP) + DISP [except if mod = 00, then EA = disp-high:disp-low] |
| 111 | EA = (BX) + DISP |

### 7.1.5 Displacement

The displacement is an 8- or 16-bit value added to the offset portion of the address.

### 7.1.6 Immediate Bytes

The immediate bytes consist of up to 16 bits of immediate data.

### 7.1.7 Segment Override Prefix

The segment override prefix is configured as shown below.

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | SR | SR | 1 | 1 | 0 |

### 7.1.8 Segment Register

The segment register is shown below.

| SR | Segment Register |
|---|---|
| 00 | ES |
| 01 | CS |
| 10 | SS |
| 11 | DS |

## 7.2 Explanation of Notation Used in Instruction Set Summary Table

Notation used in the Instruction Set Summary Table is explained below.

| Parameter | Indication |
|---|---|
| : | The component of the left is the segment for a component located in memory. The component on the right is the offset. |
| :: | The component of the left is concatenated with the component on the right. |

| Operand | Definition |
|---|---|
| imm8 | Immediate byte: signed number between –128 and 127 |
| imm16 | Immediate word: signed number between –32768 and 32767 |
| m | Operand in memory |
| m8 | Byte string in memory pointed to by DS:SI or ES:DI |
| m16 | Word string in memory pointed to by DS:SI or ES:DI |
| r/m8 | General byte register or a byte in memory |
| r/m16 | General word register or a word in memory |

### 7.2.1 Opcode

Opcode parameters and definitions are provided below.

| Parameter | Definition |
|---|---|
| /0 - /7 | The Auxiliary Field in the Operand Address byte specifies an extension (from 000 to 111, i.e., 0 to 7) to the opcode instead of a register. Thus, the opcode for adding (AND) an immediate byte to a general byte register or a byte in memory is “80 /4 ib.” This indicates that the second byte of the opcode is “mod 100 r/m.” |
| /r | The Auxiliary Field in the Operand Address byte specifies a register rather that an opcode extension. The opcode byte specifies which register, either byte size or word size, is assigned as in the aux code above. |
| /sr | This byte is placed before the instruction as shown in Section 7.1.7, Segment Override Prefix. |
| cb | The byte following the Opcode byte specifies the offset. |
| cd | The double word following the Opcode byte specifies the offset and is some cases a segment. |
| ib | Immediate byte—signed or unsigned determined by the Opcode byte. |
| iw | Immediate word—signed or unsigned determined by the Opcode byte. |
| rw | Word register operand as determined by the Opcode byte, aux field. |

### 7.2.2 Flags Affected After Instruction

Flags affected after instruction are shown below.

| | |
|---|---|
| U | Undefined |
| - | Unchanged |
| R | Result-dependent |

# 8. Innovasic/AMD Part Number Cross-Reference Tables

Tables 95 and 96 show Innovasic part numbers cross-referenced with the corresponding AMD part number.

**Table 95. Innovasic/AMD Part Number Cross-Reference for the TQFP**

| Innovasic Part Number | AMD Part Number | Package Type | Temperature Grade |
|---|---|---|---|
| IA186EM-PTQ100I-R-03 lead free (RoHS-compliant) | AM186EM-20VC\W<br>AM186EM-25VC\W<br>AM186EM-33VC\W<br>AM186EM-40VC\W<br>AM186EM-20VI\W<br>AM186EM-25VI\W<br>AM186EM-33VI\W<br>AM186EM-40VI\W | 100-Pin Thin Quad Flat Package (TQFP) | Industrial |
| IA188EM-PTQ100I-R-03 lead free (RoHS-compliant) | AM188EM-20VC\W<br>AM188EM-25VC\W<br>AM188EM-33VC\W<br>AM188EM-40VC\W<br>AM188EM-20VI\W<br>AM188EM-25VI\W<br>AM188EM-33VI\W<br>AM188EM-40VI\W | | |

## Table 96. Innovasic/AMD Part Number Cross-Reference for the PQFP

| Innovasic Part Number | AMD Part Number | Package Type | Temperature Grade |
|---|---|---|---|
| IA186EM-PQF100I-R-03 lead free (RoHS-compliant) | AM186EM-20KC\W<br>AM186EM-25KC\W<br>AM186EM-33KC\W<br>AM186EM-40KC\W<br>AM186EM-20KI\W<br>AM186EM-25KI\W<br>AM186EM-33KI\W<br>AM186EM-40KI\W | 100-Pin Plastic Quad Flat Package (PQFP) | Industrial |
| IA188EM-PQF100I-R-03 lead free (RoHS-compliant) | AM188EM-20KC\W<br>AM188EM-25KC\W<br>AM188EM-33KC\W<br>AM188EM-40KC\W<br>AM188EM-20KI\W<br>AM188EM-25KI\W<br>AM188EM-33KI\W<br>AM188EM-40KI\W | | |

# 9. Errata

The following errata are associated with Version 03 of the IA186EM/IA188EM. A workaround to the identified problem has been provided where possible.

## 9.1 Errata Summary

Table 97 presents a summary of errata.

**Table 97. Summary of Errata**

| Errata No. | Problem | Ver. 03 |
|---|---|---|
| 1 | There is a difference in how priority of timer interrupts are asserted between the original AMD part and the Innovasic part. | Exists |
| 2 | Lock up just after reset is released. | Exists |
| 3 | Intermittent startup. | Exists |
| 4 | Timer Operation in continuous mode. | Exists |
| 5 | DMA interrupt will not bring device out of halt state. | Exists |
| 6 | Does not clear the interrupt request bit for INT0 upon entering the ISR. | Exists |
| 7 | There is a difference in how hardware handshaking for UARTs during a bus hold cycle is handled between the original AMD part and the Innovasic part. | Exists |

## 9.2 Errata Detail

### Errata No. 1

**Problem:** There is a difference in how priority of timer interrupts are asserted between the original AMD part and the Innovasic part.

**Description:** In the original AMD part, timer interrupts cannot be interrupted by another timer interrupt, even if the new timer interrupt is of a higher priority. The Innovasic part will interrupt a timer interrupt with a higher-priority timer interrupt. Additionally, if a lower-priority timer interrupt is interrupted with a higher-priority timer interrupt and another incident of the lower-priority interrupt occurs during the processing of the higher-priority interrupt, upon execution of the EOI, a new lower-priority interrupt will be initiated, possibly orphaning the original lower-priority timer interrupt.

**Workaround:** When using nested interrupts, at the beginning of the interrupt routine before the global interrupts are enabled with a CLI, timer interrupts must be specifically masked. At the

end of the timer interrupt routine being serviced, set the Interrupt Enable Bit in the Process Status Word to globally disable interrupts prior to clearing the timer interrupt being serviced and unmask the appropriate timer interrupts.

### Errata No. 2

**Problem:** Lock up just after reset is released.

**Description:** Usually the first instruction is a long jump to the start of the user's code. In this case, the compiler apparently inserted a short jump instruction with zero displacement before the expected long jump instruction. The OEM device stuttered, but recovered to execute the long jump, while the device instruction pointer was corrupted, causing the lockup. In summary, a short jump with zero displacement is a corner case that does not work in the device.

**Workaround:** Do not use a short jump instruction with zero displacement.

### Errata No. 3

**Problem:** Intermittent startup.

**Description:** Processor either came out of reset normally, or would go into a series of watchdog timeouts. The addition of 10K ohm pullups to the wr_n and rd_n outputs seemed to solve the issue. Further analysis of the OEM device shows the presence of undocumented pullups on these pins, which will pull them high when the reset condition tristates these pins. The device does not include internal pullups on these pins allowing these outputs to float during reset.

**Workaround:** Add 10K ohm pullups to wr_n and rd_n pins to guarantee proper logic levels at the end of reset.

### Errata No. 4

**Problem:** Timer operation in continuous mode.

**Description:** The timers (Timer 0 and Timer 1) do not function per the specification when set in continuous mode with no external timer input stimulus to initiate/continue count.

**Workaround:** None.

### Errata No. 5

**Problem:** DMA interrupt will not bring device out of halt state.

**Description:** When device is in halt state, the interrupt caused by a DMA completion will not bring the CPU out of the halt state.

**Workaround:** Use idle mode instead of halt.

### Errata No. 6

**Problem:** Does not clear the interrupt request bit for INT0 upon entering the ISR.

**Description:** The interrupt bit for INT0 is not cleared until the interrupt routine is complete.

**Workaround:** Do not rely on the bit to be cleared when nesting interrupts.

### Errata No. 7

**Problem:** How hardware handshaking for UARTs during a bus hold cycle is handled differently between the AMD part and the Innovasic part.

**Description:** In the AMD part, hardware handshaking works per the data sheet. The Innovasic part will occasionally drive a handshake signal to the incorrect state during bus hold instead of tristating the pin.

**Workaround:** None. Avoid using hardware handshaking in conjunction with the bus hold operation with the Innovasic part UARTs.

# 10. Revision History

Table 98 presents the sequence of revisions to document IA211050902.

**Table 98. Revision History**

| Date | Revision | Description | Page(s) |
|---|---|---|---|
| August 17, 2005 | 13 | Edition released. | NA |
| January 30, 2008 | 14 | Errata 6 and 7 added. | 130 |
| February 19, 2008 | 15 | Errata 7 clarified. | 130 |
| December 24, 2008 | 16 | Document reformatted to meet publication standards. Added Conventions, Acronyms and Abbreviations, and Summary of Errata table. | All |
| July 15, 2009 | 17 | Added $V_{CC}$ parameters to Table 13. | 42 |
| January 25, 2010 | 18 | Corrected PQFP Package Dimensions table | 29 |
| February 25, 2011 | 19 | Updated section 2.2.46 to clarify that power is $\pm$10%; Removed package options to support the elimination of SnPb lead plating options | 40, 140, 141 |

# 11. For Additional Information

The IA186EM/IA188EM is a form, fit, and function replacement for the original AMD Am186EM/188EM family of microcontrollers. Innovasic produces replacement ICs using its MILES system cloning technology that produces replacement ICs far more complex than "emulation" while ensuring they are compatible with the original IC. MILES captures the design of a clone so it can be produced even as silicon technology advances. MILES also verifies the clone against the original IC so that even the "undocumented features" are duplicated.

The IA186EM/IA188EM family of microcontrollers replaces obsolete AMD Am186EM/188EM devices, allowing customers to retain existing board designs, software compilers/assemblers, and emulation tools and thus avoid expensive redesign efforts.

The Innovasic Support Team wants its information to be complete, accurate, useful, and easy to understand. Please feel free to contact experts at Innovasic with suggestions, comments, or questions at any time.

Innovasic Support Team
3737 Princeton NE, Suite 130
Albuquerque, NM 87107 USA

Phone: +1-505-883-5263 (International)
Fax: (505) 883-5477
Toll Free: (888) 824-4184 (In US)
E-mail: support@innovasic.com
Website: http://www.Innovasic.com