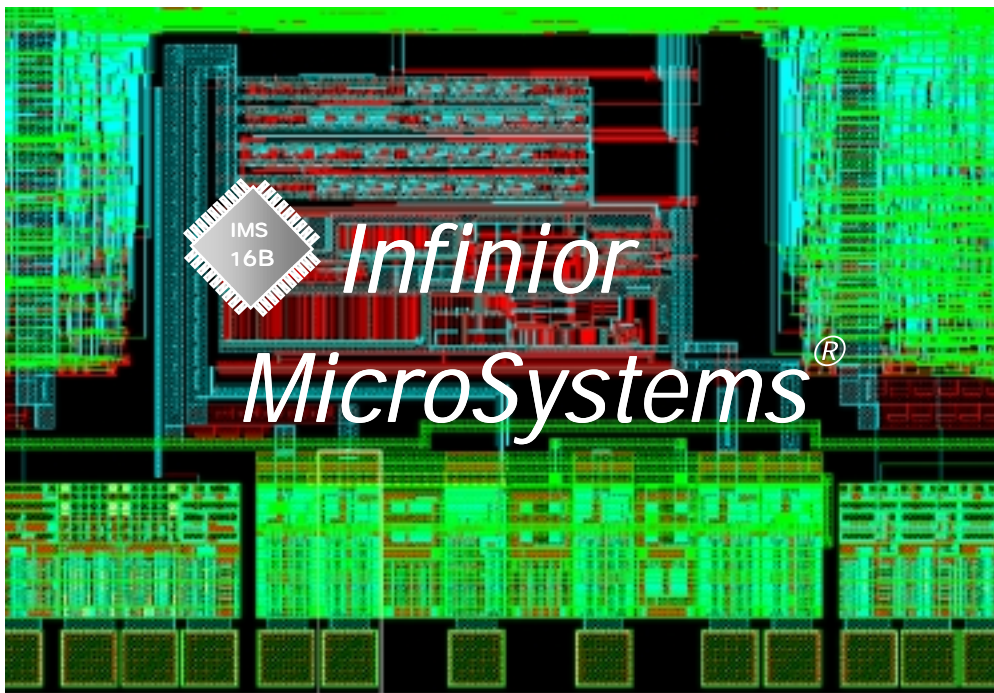


*Infinior MicroSystems'*  
*Tahoe® 16bit Embedded processor*

*User's Manual*



**January 2001**

**Infinior Microsystems**

Infinior Microsystems reserves the right to modify any products described herein at any time without notice in order to improve function or design. Infinior Microsystems does not assume any liability arising from the application or the use of any product described herein; neither does it convey any license under patent rights nor imply the rights of others.

Copyright 2001 by Infinior Microsystems:

No part of this document may be reproduced in any form or means without the prior written permission of Infinior Microsystems.

IMS16B is a registered trademark of Infinior Microsystems.

The IMS16B is available from the following manufacturer:

Infinior MicroSystems  
Accufar Bldg, 234-10, Jamsil-Dong,  
Songpa-Gu, Seoul, Korea 138-220  
Tel: +82-2-2202-0666  
FAX: +82-2-2202-9889  
[www.infinior.com](http://www.infinior.com)  
[sales@infinior.com](mailto:sales@infinior.com)

---

## Contents

---

- 1. Introduction 6
  - 1.1 Overview 6
  - 1.2 IMS16 Families and Load Map 7
  
- 2. Key Features and Applications 9
  - 2.1 Key Feature 10
  - 2.2 Applications 10
  
- 3. System Overview 11
  - 3.1 Pin outs 11
  - 3.2 Pin List 12
    - 3.2.1 Pad Description 14
  - 3.3 Block Diagram 15
  - 3.4 Pin Description 16
  - 3.5 Architecture Overview 24
  - 3.6 Bus Operation 25
  
- 4. Programming 27
  - 4.1 Register Set 27
  - 4.2 Processor Status Flag Register 28
  - 4.3 Memory Organizations and Address Generation 28
  - 4.4 Segments 29
  - 4.5 Addressing Modes 30
  - 4.6 I/O Space 31
  - 4.7 Data Types 31
  - 4.8 Instruction Set 32
  
- 5. Peripherals 34
  
- 6. Peripheral Control Register 37
  - 6.1 Peripheral Control Block Relocation Register 37
  - 6.2 Power-Save Control Register 38
  - 6.3 Initializations and Processor Reset 38
  
- 7. Chip Select Unit 39
  - 7.1 Upper Memory Chip Select Register 40
  - 7.2 Low Memory Chip Select Register 41
  - 7.3 Midrange Memory Chip Select Register 42
  - 7.4 PCS and MCS Auxiliary Register 43
  - 7.5 Peripheral Chip Select Register 44
  
- 8. Refresh Control Unit 45
  - 8.1 Memory Partition Register 45
  - 8.2 Clock Prescaler Register 45
  - 8.3 Enable RCU Register 46

- 9. Interrupt Control Unit 47
  - 9.1 Definitions of Interrupt Terms 47
  - 9.2 Interrupt Conditions and Sequence 49
  - 9.3 Interrupt Priority 49
  - 9.4 Software Exceptions, Traps and NMI 50
  - 9.5 Fully Nested Mode 51
  - 9.6 Operation in a Polled Environment 52
  - 9.7 End-of-Interrupt Write to the EOI Register 52
  - 9.8 Programmable Priority 52
  - 9.9 Trigger Mode 53
  - 9.10 Interrupt Control Registers 53
  - 9.11 INT0 and INT1 Control Registers 54
  - 9.12 INT2 and INT3 Control Registers 55
  - 9.13 INT4 Control Register 56
  - 9.14 Timer and DMA Interrupt Control Registers 57
  - 9.15 Watchdog Timer Interrupt Control Register 58
  - 9.16 UART Serial Interrupt Control Register 59
  - 9.17 Interrupt Status Register 60
  - 9.18 Interrupt Request Register 61
  - 9.19 In-Service Register 62
  - 9.20 Priority Mask Register 63
  - 9.21 Interrupt Mask Register 64
  - 9.22 Poll Status Register 65
  - 9.23 Poll Register 66
  - 9.24 End-of-Interrupt Register 67
  
- 10. Timer Control Unit 68
  - 10.1 Timer 0 and Timer 1 Mode and Control Registers 69
  - 10.2 Timer 2 Mode and Control Register 70
  - 10.3 Timer Count Registers 71
  - 10.4 Timer Maxcount compare Registers 72
  
- 11. DMA Control Unit 73
  - 11.1 DMA Operation 74
  - 11.2 DMA Control Registers 75
  - 11.3 DMA Transfer Count Registers 77
  - 11.4 DMA Destination Address High Register 78
  - 11.5 DMA Destination Address Low Register 78
  - 11.6 DMA Source Address High Register 79
  - 11.7 DMA Source Address Low Register 79
  - 11.8 DMA Requests 80
  - 11.9 DMA Acknowledge 80
  - 11.10 DMA Priority 80
  - 11.11 DMA Programming 81
  - 11.12 DMA Channels on Reset 81
  
- 12. UART(Asynchronous) Serial Interface 82
  - 12.1 Programmable Registers 82
  - 12.2 Serial Port Control Register 83
  - 12.3 Serial Port Status Register 85

- 12.4 Serial Port Transmit Data Register 86
- 12.5 Serial Port Receive Data Register 86
- 12.6 Serial Port Baud Rate Divisor Register 87
  
- 13. Synchronous Serial Interface 88
  - 13.1 Synchronous Serial Status Register 89
  - 13.2 Synchronous Serial Control Register 90
  - 13.3 Synchronous Serial Transmit Registers 91
  - 13.4 Synchronous Serial Receive Register 91
  
- 14. Programmable I/O Unit 92
  - 14.1 PIO Mode 1 Register 93
  - 14.2 PIO Mode 0 Register 93
  - 14.3 PIO Direction 1 Register 94
  - 14.4 PIO Direction 0 Register 94
  - 14.5 PIO Data Register 1 94
  - 14.6 PIO Data Register 0 94
  
- 15. I2C Serial Interface 95
  - 15.1 I2C Address Register 97
  - 15.2 I2C Control Register 97
  - 15.3 I2C Status Register 98
  - 15.4 I2C Data Register 99
  - 15.5 I2C SCL Period Setting Register 100
  - 15.6 I2C Hold Time Setting Register 101
  
- 16. SDRAM Control Unit 102
  - 16.1 Overview of SDRAM Commands 104
  - 16.2 Basic Operations 107
  - 16.3 SDRAM Control Registers 110
  - 16.4 SDRAM Enable Register 110
  - 16.5 SDRAM Auto-refresh Duty Cycle Register 111
  - 16.6 SDRAM Mode Setting Register 112
  
- APPENDIX 113
  - A. Electrical Characteristics 114
    - A.1 DC Electrical Characteristics 114
    - A.2 AC Electrical Characteristics 115
      - A.2.1 Reset Waveforms 115
      - A.2.2 Read Cycle Waveforms 116
      - A.2.3 Write Cycle Waveforms 117
      - A.2.4 SDRAM Initial Waveforms 118
      - A.2.5 SDRAM Read/Write Waveforms 119
  
  - B. Package 120
  
  - C. Typical Applications 122

**1****Introduction**

---

**1.1 Overview**

Infinior Microsystems is the leading provider of highly integrated core IP, silicon and system solutions that enable Internet multimedia appliances. Using advanced design methodologies and proprietary technologies, We designs, develops and supplies SOC(System On a Chip) products and leading edge systems for communication markets, mobile computing, digital set-top box, VoIP/VoDSL and embedded networking devices.

Infinior Microsystems believes that one of its key competitive advantages is its high performance, highly integrated IP core based SOC design technologies encompassing the complete design space from systems to silicon.

High performance 16bit microcontroller with variety of peripherals like I2C, SDRAM, MAC controller and PCI bridge function still remains instruction level compatibility with industry standard processor such as Intel 80C186, AMD Am186 and NEC V25 series.

The IMS16B microcontroller is a performance-enhanced implementations of the industry standard microprocessors with powerful up-to date peripherals such as SDRAM controller and an I2C controller.

The IMS16U microcontroller is an enhanced version of IMS16B. In addition to IMS16B peripherals, IMS16U microcontroller also has USB Control Unit.

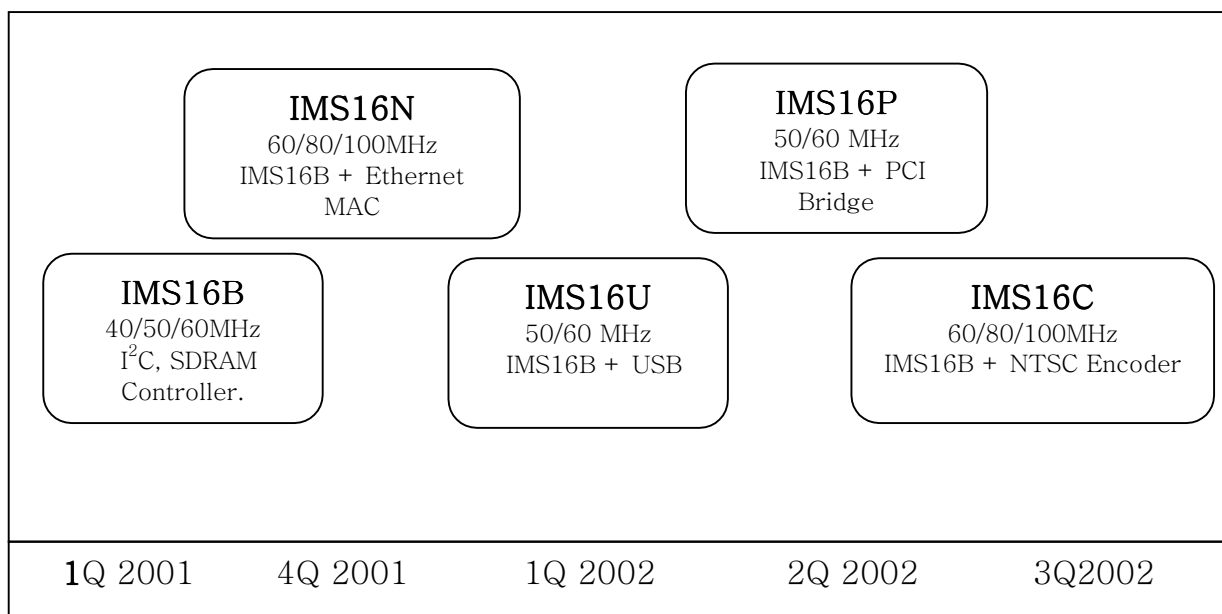
The IMS16P microcontroller is an enhanced version of IMS16B. In addition to IMS16B peripherals, IMS16P microcontroller also has PCI Interface Control Unit.

The IMS16N microcontroller is an enhanced version of IMS16B. In addition to IMS16B peripherals, IMS16N microcontroller also has MAC Control Unit.

The IMS16C microcontroller is an enhanced version of IMS16B. In addition to IMS16B peripherals, IMS16C microcontroller also has NTSC Encoder Control Unit.

## 1.2 IMS16 Families and Load Map

	<b>IMS16B</b>	<b>IMS16U</b>	<b>IMS16P</b>	<b>IMS16N</b>	<b>IMS16C</b>
Speed	40/50/60MHz @3.3V	50/60/80MHz @3.3V	50/60/80MHz @3.3V	60/80/100MHz @3.3V	60/80/100MHz @3.3V
Address Bus	20bit	20bit	20bit	20bit	20bit
Data Bus	16bit	16bit	16bit	16bit	16bit
Package	128pin QFP				
Deliverable	Silicon/IP/Verilog	IP/Verilog	IP/Verilog	IP/Verilog	IP/Verilog
Available	1Q 2001	1Q 2002	2Q 2002	4Q 2001	3Q 2002
<b>Additional Peripherals</b>					
I2C	Yes	Yes	Yes	Yes	Yes
SDRAM Controller	Yes	Yes	Yes/No	Yes	Yes
USB	No	Yes	No	Yes/No	No
PCI Interface	No	No	Yes	No	No
MAC Controller	No	No	No	Yes	Yes
NTSC Encoder	No	No	No	No	Yes



## 1.2 IMS16B

This Page is Left Blank Intentionally !!

**2****Key Features and Applications**

The IMS16B is a performance-enhanced implementations of the industry standard AMD Am186EM® microprocessors with powerful up-to date peripherals such as SDRAM controller and an I2C controller. This microcontroller integrates the functions of the CPU, non-multiplexed address bus, timers, chip selects, interrupt controller, DMA controller, asynchronous serial port, synchronous serial interface and programmable I/O pins on one chip.

Compared to the existing 16bit microcontrollers, the IMS16B enables system designers to reduce the size, power consumption, and cost of embedded systems, while increasing functionality and performance. Development environments are also widely available.

The IMS16B has been totally rearchitected from the ground up using the latest design techniques to produce an efficient, high clock-rate CPU core. It provides higher performance than Am186EM and the Intel 80C186 at the same clock frequency. Low power applications not needing this increased performance can run at 1/2 the clock frequency and still obtain more throughputs.

The IMS16B has been designed to meet the most common requirements of embedded products developed for the office automation, mass storage, communications and general embedded markets. This device is ideal for use in a broad range of communications applications, including ISDN terminal adapters, low-end routers, digital subscriber line (xDSL) modems, PBX applications, digital phones and key telephone systems. This makes the IMS16B ideal devices for designs requiring high performance, serial communications, and a glueless bus interface to external memory systems.

## 2.1 Key Features

- 100% software compatible with the Intel 8086/80186®
- Supported by widely available native x86 development tools
- 1Mbyte Memory address space, 64Kbyte I/O space
- 16-bit ALU
- IMS16B includes all AMD Am186EM peripherals
  - Peripheral Interface Logic
  - Chip-Select and Ready Control Logic
  - 2 Channel DMA Controllers
  - 3 Programmable 16-bit Timers
  - Interrupt Controller
  - Refresh Control Unit
  - Power Save Logic
  - Chip Select Unit
  - Synchronous & Asynchronous Serial Ports
  - Programmable I/O (PIO)
- I<sup>2</sup>C Controller
- SDRAM Controller
- PLL Management Unit
- 1/2, x2, x5 Clock Mode
- Fully synchronous and static design
- 40/50/60Mhz operation in 3.3V
- 0.5um 3.3V CMOS Process
- 3.3V I/O pad (5V non-tolerant)
- 128-Pin Plastic Quad Flat Pack (PQFP)

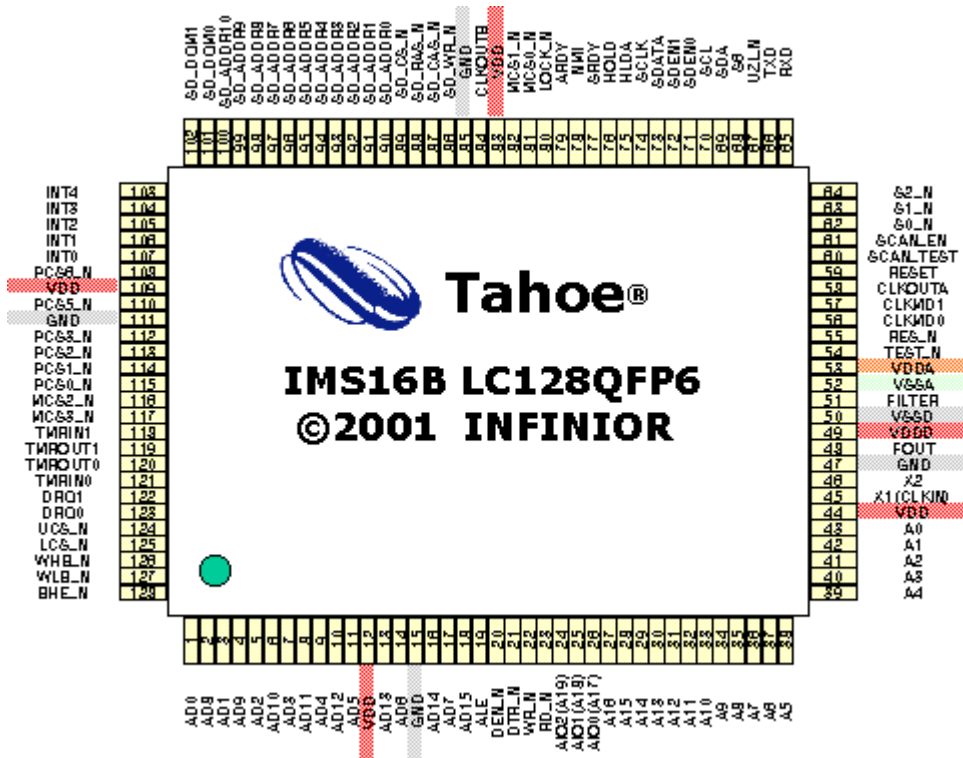
## 2.2 Applications

- General Purpose Applications
  - Toys, Automotive, Instrumentation, Medical, Process and Industrial control
- PC Peripheral Applications
  - Disk and optical drive storage, Retail and consumer products, Various Peripheral Systems and PCI Add-on cards
- Telecommunication Applications
  - Fax Machine, ADSL Modem cards, Key phone System
- Networking Applications
  - VoIP Client, Intelligent Ethernet Hub, SOHO Router

**3**

**System Overview**

**3.1 Pin Outs**



Note:  
128-pin Plastic Quad Flat Package(PQFP)

### 3.2 Pin List

Pin No.	Pin Name	Pad Name
1	AD0	BST4
2	AD8	BST4
3	AD1	BST4
4	AD9	BST4
5	AD2	BST4
6	AD10	BST4
7	AD3	BST4
8	AD11	BST4
9	AD4	BST4
10	AD12	BST4
11	AD5	BST4
12	VDD	
13	AD13	BST4
14	AD6	BST4
15	GND	
16	AD14	BST4
17	AD7	BST4
18	AD15	BST4
19	ALE	OB4
20	DEN_N	BST4
21	DTR_N	BST4
22	WR_N	OT4
23	RD_N	OT4
24	AIO2 (A19)	BST4
25	AIO1 (A18)	BST4
26	AIO0 (A17)	BST4
27	A16	OT4
28	A15	OT4
29	A14	OT4
30	A13	OT4
31	A12	OT4
32	A11	OT4

Pin No.	Pin Name	Pad Name
33	A10	OT4
34	A9	OT4
35	A8	OT4
36	A7	OT4
37	A6	OT4
38	A5	OT4
39	A4	OT4
40	A3	OT4
41	A2	OT4
42	A1	OT4
43	A0	OT4
44	VDD	
45	X1 (CLKIN)	PSOSCM3
46	X2	
47	GND	
48	FOUT	OB4
49	VDDD	Digital VDD for PLL
50	VSSD	Digital VSS for PLL
51	FILTER	OB4
52	VSSA	Analog VSS for PLL
53	VDDA	Analog VDD for PLL
54	TEST_N	ISU
55	RES_N	ISU
56	CLKMD0	ISD
57	CLKMD1	ISD
58	CLKOUTA	OT4
59	RESET	OB4
60	SCAN_TEST	ISD
61	SCAN_EN	ISD
62	S0_N	OT4
63	S1_N	OT4
64	S2_N	OT4

### 3.2 Pin List (Contd.)

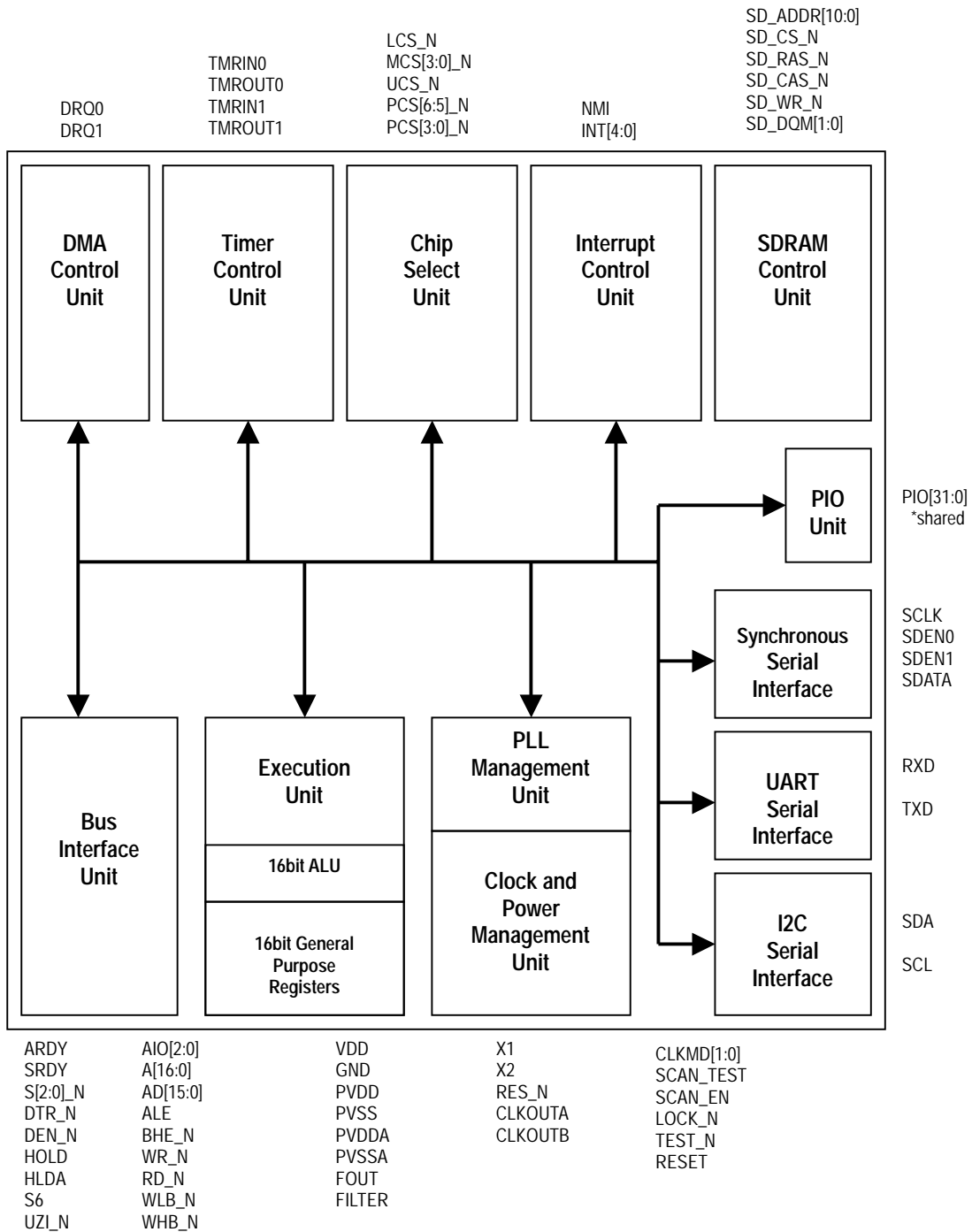
Pin No.	Pin Name	Pad Name
65	RXD	BSUT4
66	TXD	BSUT4
67	UZI_N	BSUT4
68	S6	BSUT4
69	SDA	BSUD4
70	SCL	BSUD4
71	SDEN0	BSDT4
72	SDEN1	BSDT4
73	SDATA	BSUT4
74	SCLK	BSUT4
75	HLDA	OB4
76	HOLD	ISD
77	SRDY	BST4
78	NMI	ISD
79	ARDY	ISU
80	LOCK_N	OT4
81	MCS0_N	BSUT4
82	MCS1_N	BSUT4
83	VDD	
84	CLKOUTB	OT4
85	GND	
86	SD_WR_N	OB4
87	SD_CAS_N	OB4
88	SD_RAS_N	OB4
89	SD_CS_N	OB4
90	SD_ADDR0	OB4
91	SD_ADDR1	OB4
92	SD_ADDR2	OB4
93	SD_ADDR3	OB4
94	SD_ADDR4	OB4
95	SD_ADDR5	OB4
96	SD_ADDR6	OB4

Pin No.	Pin Name	Pad Name
97	SD_ADDR7	OB4
98	SD_ADDR8	OB4
99	SD_ADDR9	OB4
100	SD_ADDR10	OB4
101	SD_DQM0	OB4
102	SD_DQM1	OB4
103	INT4	BSUT4
104	INT3	BSUT4
105	INT2	BSUT4
106	INT1	ISD
107	INT0	ISD
108	PCS6_N	BSUT4
109	VDD	
110	PCS5_N	BSUT4
111	GND	
112	PCS3_N	BSUT4
113	PCS2_N	BSUT4
114	PCS1_N	BSUT4
115	PCS0_N	BSUT4
116	MCS2_N	BSUT4
117	MCS3_N	BSUT4
118	TMRIN1	BSUT4
119	TMROUT1	BSUT4
120	TMROUT0	BSUT4
121	TMRIN0	BSUT4
122	DRQ1	BSUT4
123	DRQ0	BSUT4
124	UCS_N	OT4
125	LCS_N	OT4
126	WHB_N	OT4
127	WLB_N	OT4
128	BHE_N	OT4

### 3.2.1 Pad Description

Pad name	Comments
PSOSCM3	3.3V Crystal Oscillator Pads. 50Mhz ~ 100Mhz clkIn
ISU	3.3V CMOS Input Only Pads With Pull-up Register and Schmitt Trigger non-inverting
ISD	3.3V CMOS Input Only Pads With Pull-down Register and Schmitt Trigger non-inverting.
OB4	3.3V CMOS Output Pads with 4mA drive
OT4	3.3V CMOS 3-State Output Pads with 4mA drive
BST4	3.3V CMOS 3-State I/O Pads with 4mA drive
BSDT4	3.3V CMOS 3-State I/O Pads With Pull-down Register with 4mA drive
BSUT4	3.3V CMOS 3-State I/O Pads With Pull-up Register with 4mA drive
BSUD4	3.3V CMOS Open Drain Bi-Directional Pads With Pull-up Register with 4mA drive

### 3.3 Block Diagram



### 3.4 Pin Description

Pin Name	Type	Description															
<b>A16 - A0</b>	<b>Address Bus (3-state output)</b>	These pins supply non-multiplexed memory or I/O addresses to the system one-half of a CLKOUTA period earlier than the multiplexed address and data bus. During a bus hold or reset condition, the address bus is in a high-impedance state.															
<b>AIO2 - AIO0 (A19/PIO9, A18/PIO8, A17/PIO7)</b>	<b>Address, PIO Bus (3-state I/O)</b>	These pins are used address A19-A17 and programmable Input/output pin PIO9-PIO7.															
<b>AD15-AD0</b>	<b>Address and Data Bus (3-state I/O)</b>	These time-multiplexed pins supply partial memory or I/O addresses, as well as data, to the system. This bus supplies the 16bit address to the system during the first period of a bus cycle (t1), and it supplies data to the system during the remaining periods of that cycle (t2, t3, and t4). During a bus hold or reset condition, the address and data bus is in a high-impedance state. During a power-on reset, the address and data bus pins can also be used to load system configuration information into the internal reset configuration register.															
<b>ALE</b>	<b>Address Latch Enable (output)</b>	This pin indicates to the system that an address appears on the address and data bus. The address is guaranteed valid on the trailing edge of ALE. This pin is not three-stated during a bus hold or reset.															
<b>ARDY</b>	<b>Asynchronous Ready (input)</b>	This pin indicates to the microcontroller that the addressed memory space or I/O device will complete a data transfer. The ARDY pin accepts a rising edge that is asynchronous to CLKOUTA and is active High. The falling edge of ARDY must be synchronized to CLKOUTA. To always assert the ready condition to the microcontroller, tie ARDY High. If the system does not use ARDY, tie the pin Low to yield control to SRDY.															
<b>BHE_N</b>	<b>Bus High Enable (3-state output)</b>	During a memory access, this pin and the least-significant address bit (AD0 or A0) indicate to the system, which bytes of the data bus (upper, lower, or both) participate in a bus cycle. The BHE_N and AD0 pins are encoded as shown in Table 1. BHE_N is asserted during t1 and remains asserted through t3 and tw. BHE_N does not need to be latched. BHE_N floats during bus hold and reset. On the IMS16B, WLB_N and WHB_N implement the functionality of BHE_N and AD0 for high and low byte write enables. <table border="1" data-bbox="730 1509 1404 1711"> <thead> <tr> <th>BHE_N</th> <th>AD0</th> <th>Type of Bus Cycle</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Word Transfer</td> </tr> <tr> <td>0</td> <td>1</td> <td>High Byte Transfer with valid data on Dout(15:8)</td> </tr> <tr> <td>1</td> <td>0</td> <td>Low Byte Transfer with valid data on Dout(7:0)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table>	BHE_N	AD0	Type of Bus Cycle	0	0	Word Transfer	0	1	High Byte Transfer with valid data on Dout(15:8)	1	0	Low Byte Transfer with valid data on Dout(7:0)	1	1	Reserved
BHE_N	AD0	Type of Bus Cycle															
0	0	Word Transfer															
0	1	High Byte Transfer with valid data on Dout(15:8)															
1	0	Low Byte Transfer with valid data on Dout(7:0)															
1	1	Reserved															
<b>CLKMD0, CLKMD1</b>	<b>Core Clock Mode (input)</b>	CLKMD0 and CLKMD1 signals are used for core clock configuration. These pins define PLL frequency coefficient. For example, If IMS16B is assigned 10Mhz through the external clock source(X1), when CLKMD=00b, it can be working 5Mhz internally, when CLKMD=01b, it can be working 20Mhz internally. IMS16B also provides user programmable PLL mode. During reset, AD[15:0] is sampled to give PLLREG new PLL coefficients. Note that PLLREG sampling time lasts only 64 external clock(X1)															

		<p>cycle times.</p> <table border="1"> <thead> <tr> <th>CLKMD[1:0]</th> <th>CORE CLOCK</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>External clock source bypass mode (1/2)</td> </tr> <tr> <td>01</td> <td>X2 PLL clock mode</td> </tr> <tr> <td>10</td> <td>X5 PLL clock mode</td> </tr> <tr> <td>11</td> <td>User Programmable PLL mode</td> </tr> </tbody> </table> <p>Table 2. core clock mode</p> <p>Note:  <math>F_{out} = (M+8) / ((P+2) * 2^S) * F_{in}</math> where P, M, S are decimal.                      PLLREG(P[5:0] &amp; M[7:0] &amp; S[1:0]) &lt;= AD[15:0] during reset.</p>	CLKMD[1:0]	CORE CLOCK	00	External clock source bypass mode (1/2)	01	X2 PLL clock mode	10	X5 PLL clock mode	11	User Programmable PLL mode
CLKMD[1:0]	CORE CLOCK											
00	External clock source bypass mode (1/2)											
01	X2 PLL clock mode											
10	X5 PLL clock mode											
11	User Programmable PLL mode											
<b>CLKOUTA</b>	<b>Clock Output A (3- state output)</b>	This pin supplies the internal clock to the system. Depending on the value of the power-save control register(PDCON), CLKOUTA operates at either the crystal input frequency (X1), the power-save frequency, or is three-stated. CLKOUTA remains active during reset and bus hold conditions.										
<b>CLKOUTB</b>	<b>Clock Output B (3- state output)</b>	This pin supplies the internal clock to the system. Depending on the value of the power-save control register(PDCON), CLKOUTB operates at either the crystal input frequency (X1), the power-save frequency, or is three-stated. CLKOUTA remains active during reset and bus hold conditions. This pin can be used to supply the internal clock to the external SDRAM CLK pin.										
<b>DEN_N/PIO5</b>	<b>Data Enable (3- state output)</b>	This pin supplies an output enable to an external data-bus transceiver. DEN_N is asserted during memory, I/O, and interrupt acknowledge cycles. DEN_N is deasserted when DTR_N changes state. DEN_N floats during a bus hold or reset condition.										
<b>DRQ1 - DRQ0 (DRQ1/PIO13, DRQ0/PIO12)</b>	<b>DMA Requests (input)</b>	These pins indicate to the microcontroller that an external device is ready for DMA channel 1 or channel 0 to perform a transfer. DRQ1 - DRQ0 are level-triggered and internally synchronized. The DRQ signals are not latched and must remain active until serviced.										
<b>DTR_N/PIO4</b>	<b>Data Transmit or Receive (output, three-state, synchronous)</b>	This pin indicates which direction data should flow through an external data-bus transceiver. When DTR_N is asserted High, the microcontroller transmits data. When this pin is deasserted Low, the microcontroller receives data. DTR_N floats during a bus hold or reset condition.										
<b>FILTER</b>	<b>Filter Out (output)</b>	For PLL test. Pump out is connected to the FILTER. A 560pF Capacitor is connected between this pin and analog ground pin.										
<b>FOUT</b>	<b>PLL Clock Out (output)</b>	PLL clock output for test.										
<b>GND</b>	<b>Core ground (input)</b>	These pins connect the system ground to the IMS16B.										
<b>HLDA</b>	<b>Bus Hold Acknowledge (output)</b>	This pin is asserted High to indicate to an external bus master that the microcontroller has released control of the local bus. When an external bus master requests control of the local bus (by asserting HOLD), the microcontroller completes the bus cycle in progress and then relinquishes control of the bus to the external bus master by asserting HLDA and floating DEN_N, RD_N, WR_N, S2_N - S0_N, AD15 - AD0, S6, A19 - A0, BHE_N, WHB_N, WLB_N, and DTR_N, and then driving the chip selects UCS_N, LCS_N, MCS3_N - MCS0_N, PCS6_N - PCS5_N, and PCS3_N - PCS0_N High. When the external bus master has finished using the local bus, it indicates this to the microcontroller by deasserting HOLD. The microcontroller responds by deasserting HLDA.										
<b>HOLD</b>	<b>Bus Hold Request (input)</b>	This pin indicates to the microcontroller that an external bus master needs control of the local bus. The IMS16B microcontroller' HOLD latency time is a function of the activity occurring in the processor when the HOLD request is										

		received.
<b>INT0</b>	<b>Maskable Interrupt Request 0 (input)</b>	This pin indicates to the microcontroller that an interrupt request has occurred. If the INT0 pin is not masked, the microcontroller transfers program execution to the location specified by the INT0 vector in the microcontroller interrupt vector table. Interrupt requests are synchronized internally and can be edge-triggered or level-triggered. To guarantee interrupt recognition, the requesting device must continue asserting INT0 until the request is acknowledged.
<b>INT1</b>	<b>Maskable Interrupt Request 1 (input)</b>	This pin indicates to the microcontroller that an interrupt request has occurred. If INT1 is not masked, the microcontroller transfers program execution to the location specified by the INT1 vector in the microcontroller interrupt vector table. Interrupt requests are synchronized internally and can be edge-triggered or level-triggered. To guarantee interrupt recognition, the requesting device must continue asserting INT1 until the request is acknowledged.
<b>INT2/PIO31</b>	<b>Maskable Interrupt Request 2 (input)</b>	This pin indicates to the microcontroller that an interrupt request has occurred. If the INT2 pin is not masked, the microcontroller transfers program execution to the location specified by the INT2 vector in the microcontroller interrupt vector table. Interrupt requests are synchronized internally and can be edge-triggered or level-triggered. To guarantee interrupt recognition, the requesting device must continue asserting INT2 until the request is acknowledged.
<b>INT3</b>	<b>Maskable Interrupt Request 3 (input)</b>	This pin indicates to the microcontroller that an interrupt request has occurred. If the INT3 pin is not masked, the microcontroller then transfers program execution to the location specified by the INT3 vector in the microcontroller interrupt vector table. Interrupt requests are synchronized internally, and can be edge-triggered or level-triggered. To guarantee interrupt recognition, the requesting device must continue asserting INT3 until the request is acknowledged.
<b>INT4/PIO30</b>	<b>Maskable Interrupt Request 4 (input)</b>	This pin indicates to the microcontroller that an interrupt request has occurred. If the INT4 pin is not masked, the microcontroller then transfers program execution to the location specified by the INT4 vector in the microcontroller interrupt vector table. Interrupt requests are synchronized internally, and can be edge-triggered or level-triggered. To guarantee interrupt recognition, the requesting device must continue asserting INT4 until the request is acknowledged.
<b>LCS_N</b>	<b>Lower Memory Chip Select (output, internal pullup)</b>	This pin indicates to the system that a memory access is in progress to the lower memory block. The base address and size of the lower memory block are programmable up to 512 Kbytes. LCS is held High during a bus hold condition.
<b>LOCK_N</b>	<b>LOCK out (3-state output)</b>	LOCK_N output indicates that other system bus masters are not to gain control of the system bus while LOCK_N is active low. The LOCK_N signal is requested by the LOCK prefix instruction and is activated at the beginning of the first data cycle associated with the instruction following the LOCK prefix. It remains active until the completion of that instruction. No instruction prefetching will occur while LOCK_N is asserted. When executing more than one LOCK instruction, always make sure there are 6 bytes of code between the end of the first LOCK instruction and the start of the second LOCK instruction.
<b>MCS3_N/PIO25</b>	<b>Midrange Memory Chip Select 3 (output, internal pullup)</b>	This pin indicates to the system that a memory access is in progress to the fourth region of the midrange memory block. The base address and size of the midrange memory block are programmable. MCS3_N is held High during a bus hold condition. In addition, this

		pin has a weak internal pullup resistor that is active during reset.
<b>MCS2_N - MCS0_N (MCS2/PIO24, MCS1/PIO15, MCS0/PIO14)</b>	<b>Midrange Memory Chip Selects (output, internal pullup)</b>	<p>These pins indicate to the system that a memory access is in progress to the corresponding region of the midrange memory block. The base address and size of the midrange memory block are programmable.</p> <p>MCS2_N - MCS0_N are held High during a bus hold condition. In addition, they have weak internal pullup resistors that are active during reset.</p>
<b>NMI</b>	<b>Nonmaskable Interrupt (input)</b>	<p>This pin indicates to the microcontroller that an interrupt request has occurred. The NMI signal is the highest priority hardware interrupt and, unlike the INT4 - INT0 pins, cannot be masked. The microcontroller always transfers program execution to the location specified by the nonmaskable interrupt vector in the microcontroller interrupt vector table when NMI is asserted.</p> <p>Although NMI is the highest priority interrupt source, it does not participate in the priority resolution process of the maskable interrupts. There is no bit associated with NMI in the interrupt in-service or interrupt request registers.</p> <p>This means that a new NMI request can interrupt an executing NMI interrupt service routine. As with all hardware interrupts, the IF (interrupt enable flag) is cleared when the processor takes the interrupt, disabling the maskable interrupt sources.</p> <p>However, if maskable interrupts are re-enabled by software in the NMI interrupt service routine, via the STI instruction for example, the fact that an NMI is currently in service will not have any effect on the priority resolution of maskable interrupt requests.</p> <p>For this reason, it is strongly advised that the interrupt service routine for NMI does not enable the maskable interrupts.</p> <p>An NMI transition from Low to High is latched and synchronized internally, and it initiates the interrupt at the next instruction boundary. To guarantee that the interrupt is recognized, the NMI pin must be asserted for at least one CLKOUTA period.</p>
<b>PCS3_N - PCS0_N (PCS3_N/PIO19, PCS2_N/PIO18, PCS1_N/PIO17, PCS0_N/PIO16)</b>	<b>Peripheral Chip Selects (output)</b>	<p>These pins indicate to the system that a memory access is in progress to the corresponding region of the peripheral memory block (either I/O or memory address space). The base address of the peripheral memory block is programmable.</p> <p>PCS3_N - PCS0_N are held High during a bus hold condition. They are also held High during reset. PCS4 is not available on the IMS16B.</p> <p>Unlike the UCS_N and LCS_N chip selects, the PCS_N outputs assert with the multiplexed AD address bus.</p> <p>Note also that each peripheral chip select asserts over a 256byte address range, which is twice the address range covered by peripheral chip selects in the 80C186 and 80C188 microcontrollers.</p>
<b>PCS5_N/A1/PIO3</b>	<b>Peripheral Chip Select 5 (output)  Latched Address Bit 1 (output)</b>	<p><b>PCS5_N:</b> This pin indicates to the system that a memory access is in progress to the sixth region of the peripheral memory block (either I/O or memory address space). The base address of the peripheral memory block is programmable. PCS5_N is held High during a bus hold condition. It is also held High during reset.</p> <p>Unlike the UCS_N and LCS_N chip selects, the PCS_N outputs assert with the multiplexed AD address bus.</p> <p>Note also that each peripheral chip select asserts over a 256 byte address range, which is twice the address range covered by peripheral chip selects in the 80C186.</p> <p><b>A1:</b> When the EX bit in the MCS_N and PCS_N auxiliary register is 0, this pin supplies an internally latched address bit 1 to the system. During a bus hold condition, A1 retains its previously</p>

<p><b>PCS6_N/A2/PIO2</b></p>	<p><b>Peripheral Chip Select 6 (output)</b></p> <p><b>Latched Address Bit 2 (output)</b></p>	<p>latched value.</p> <p><b>PCS6_N:</b> This pin indicates to the system that a memory access is in progress to the seventh region of the peripheral memory block (either I/O or memory address space). The base address of the peripheral memory block is programmable. PCS6_N is held High during a bus hold condition or reset. Unlike the UCS_N and LCS_N chip selects, the PCS outputs assert with the multiplexed AD address bus. Note also that each peripheral chip select asserts over a 256 byte address range, which is twice the address range covered by peripheral chip selects in the 80C186.</p> <p><b>A2:</b> When the EX bit in the MCS and PCS Auxiliary Register is 0, this pin supplies an internally latched address bit 2 to the system. During a bus hold condition, A2 retains its previously latched value.</p>																																																																														
<p><b>PIO31 - PIO0 (Shared)</b></p>	<p><b>Programmable I/O Pins (input/output, asynchronous, open-drain)</b></p>	<p>The IMS16B microcontroller provides 32 individually programmable I/O pins. Each PIO can be programmed with the following attributes: PIO function (enabled/disabled), direction (input/output), and weak pullup or pulldown. The pins that are multiplexed with PIO31 - PIO0 are listed in Table 3.</p> <p>After power-on reset, the PIO pins default to various configurations. The column titled Power-On Reset Status in Table 2 lists the defaults for the PIOs. The system initialization code must reconfigure any PIOs as required.</p> <p>The A19 - A17(AIO2 –AIO0) address pins default to normal operation on power-on reset, allowing the processor to correctly begin fetching instructions at the boot address FFFF0h. The DTR_N, DEN_N, and SRDY pins also default to normal operation on power-on reset.</p> <table border="1" data-bbox="726 1205 1358 1946"> <thead> <tr> <th>PIO</th> <th>Associated</th> <th>Power-On Reset Status</th> </tr> </thead> <tbody> <tr><td>0</td><td>TMRIN1</td><td>Input with pullup</td></tr> <tr><td>1</td><td>TMROUT1</td><td>Input with pulldown</td></tr> <tr><td>2</td><td>PCS6_N</td><td>Input with pullup</td></tr> <tr><td>3</td><td>PCS5_N</td><td>Input with pullup</td></tr> <tr><td>4</td><td>DTR_N</td><td>Normal operation (1)</td></tr> <tr><td>5</td><td>DEN_N</td><td>Normal operation (1)</td></tr> <tr><td>6</td><td>SRDY</td><td>Normal operation (2)</td></tr> <tr><td>7</td><td>AIO0(A17)</td><td>Normal operation (1)</td></tr> <tr><td>8</td><td>AIO1(A18)</td><td>Normal operation (1)</td></tr> <tr><td>9</td><td>AIO2(A19)</td><td>Normal operation (1)</td></tr> <tr><td>10</td><td>TMROUT0</td><td>Input with pulldown</td></tr> <tr><td>11</td><td>TMRIN0</td><td>Input with pullup</td></tr> <tr><td>12</td><td>DRQ0</td><td>Input with pullup</td></tr> <tr><td>13</td><td>DRQ1</td><td>Input with pullup</td></tr> <tr><td>14</td><td>MCS0_N</td><td>Input with pullup</td></tr> <tr><td>15</td><td>MCS1_N</td><td>Input with pullup</td></tr> <tr><td>16</td><td>PCS0_N</td><td>Input with pullup</td></tr> <tr><td>17</td><td>PCS1_N</td><td>Input with pullup</td></tr> <tr><td>18</td><td>PCS2_N</td><td>Input with pullup</td></tr> <tr><td>19</td><td>PCS3_N</td><td>Input with pullup</td></tr> <tr><td>20</td><td>SCLK</td><td>Input with pullup</td></tr> <tr><td>21</td><td>SDATA</td><td>Input with pullup</td></tr> <tr><td>22</td><td>SDEN0</td><td>Input with pulldown</td></tr> <tr><td>23</td><td>SDEN1</td><td>Input with pulldown</td></tr> <tr><td>24</td><td>MCS2_N</td><td>Input with pullup</td></tr> </tbody> </table>	PIO	Associated	Power-On Reset Status	0	TMRIN1	Input with pullup	1	TMROUT1	Input with pulldown	2	PCS6_N	Input with pullup	3	PCS5_N	Input with pullup	4	DTR_N	Normal operation (1)	5	DEN_N	Normal operation (1)	6	SRDY	Normal operation (2)	7	AIO0(A17)	Normal operation (1)	8	AIO1(A18)	Normal operation (1)	9	AIO2(A19)	Normal operation (1)	10	TMROUT0	Input with pulldown	11	TMRIN0	Input with pullup	12	DRQ0	Input with pullup	13	DRQ1	Input with pullup	14	MCS0_N	Input with pullup	15	MCS1_N	Input with pullup	16	PCS0_N	Input with pullup	17	PCS1_N	Input with pullup	18	PCS2_N	Input with pullup	19	PCS3_N	Input with pullup	20	SCLK	Input with pullup	21	SDATA	Input with pullup	22	SDEN0	Input with pulldown	23	SDEN1	Input with pulldown	24	MCS2_N	Input with pullup
PIO	Associated	Power-On Reset Status																																																																														
0	TMRIN1	Input with pullup																																																																														
1	TMROUT1	Input with pulldown																																																																														
2	PCS6_N	Input with pullup																																																																														
3	PCS5_N	Input with pullup																																																																														
4	DTR_N	Normal operation (1)																																																																														
5	DEN_N	Normal operation (1)																																																																														
6	SRDY	Normal operation (2)																																																																														
7	AIO0(A17)	Normal operation (1)																																																																														
8	AIO1(A18)	Normal operation (1)																																																																														
9	AIO2(A19)	Normal operation (1)																																																																														
10	TMROUT0	Input with pulldown																																																																														
11	TMRIN0	Input with pullup																																																																														
12	DRQ0	Input with pullup																																																																														
13	DRQ1	Input with pullup																																																																														
14	MCS0_N	Input with pullup																																																																														
15	MCS1_N	Input with pullup																																																																														
16	PCS0_N	Input with pullup																																																																														
17	PCS1_N	Input with pullup																																																																														
18	PCS2_N	Input with pullup																																																																														
19	PCS3_N	Input with pullup																																																																														
20	SCLK	Input with pullup																																																																														
21	SDATA	Input with pullup																																																																														
22	SDEN0	Input with pulldown																																																																														
23	SDEN1	Input with pulldown																																																																														
24	MCS2_N	Input with pullup																																																																														

		<table border="1"> <tr><td>24</td><td>MCS2_N</td><td>Input with pullup</td></tr> <tr><td>25</td><td>MCS3_N/RFSH</td><td>Input with pullup</td></tr> <tr><td>26</td><td>UZI_N</td><td>Input with pullup</td></tr> <tr><td>27</td><td>TXD</td><td>Input with pullup</td></tr> <tr><td>28</td><td>RXD</td><td>Input with pullup</td></tr> <tr><td>29</td><td>S6</td><td>Input with pullup</td></tr> <tr><td>30</td><td>INT4</td><td>Input with pullup</td></tr> <tr><td>31</td><td>INT2</td><td>Input with pullup</td></tr> </table> <p style="text-align: center;">Table 3. Numeric PIO Pin Assignments</p> <p>Note:                      1. When used as a PIO, input with pullup option available.                      2. When used as a PIO, input with pulldown option available.</p>	24	MCS2_N	Input with pullup	25	MCS3_N/RFSH	Input with pullup	26	UZI_N	Input with pullup	27	TXD	Input with pullup	28	RXD	Input with pullup	29	S6	Input with pullup	30	INT4	Input with pullup	31	INT2	Input with pullup												
24	MCS2_N	Input with pullup																																				
25	MCS3_N/RFSH	Input with pullup																																				
26	UZI_N	Input with pullup																																				
27	TXD	Input with pullup																																				
28	RXD	Input with pullup																																				
29	S6	Input with pullup																																				
30	INT4	Input with pullup																																				
31	INT2	Input with pullup																																				
<b>PVDDA, PVSSA</b>	<b>Analog vdd, vss (input)</b>	PLL Core Analog power supply and ground																																				
<b>PVDD, PVSS</b>	<b>Digital vdd, vss (input)</b>	PLL Core Digital power supply and ground																																				
<b>RESET</b>	<b>Reset out(output)</b>	Reset Output indicates that the CPU is being reset, and can be used as a system reset, It is active high, synchronized with the processor clock, and lasts an integer number of clock periods corresponding to the length of the RES_N signal.																																				
<b>RD_N</b>	<b>Read Strobe (3-state output)</b>	This pin indicates to the system that the microcontroller is performing a memory or I/O read cycle. RD_N is guaranteed not to be asserted before the address and data bus is floated during the address-to-data transition. RD_N floats during a bus hold condition																																				
<b>RES_N</b>	<b>Reset (input)</b>	This pin requires the microcontroller to perform a reset. When RES_N is asserted, the microcontroller immediately terminates its present activity, clears its internal logic, and CPU control is transferred to the reset address FFFF0h. RES_N must be held Low for at least 1 ms. RES_N can be asserted asynchronously to CLKOUTA because RES_N is synchronized internally. For proper initialization, VCC must be within specifications, and CLKOUTA must be stable for more than four CLKOUTA periods during which RES_N is asserted. This input is provided with Schmitt trigger to facilitate power-on reset generation via an RC network.																																				
<b>RXD/PIO28</b>	<b>Receive Data (input)</b>	This pin supplies asynchronous serial receive data from the system to the internal UART of the microcontroller.																																				
<b>S2_N - S0_N</b>	<b>Bus Cycle Status (3-state output)</b>	<p>These pins indicate to the system the type of bus cycle in progress. S2_N can be used as a logical memory or I/O indicator, and S1_N can be used as a data transmit or receive indicator. S2_N - S0_N float during bus hold and hold acknowledges conditions.</p> <p>The S2_N - S0_N pins are encoded as shown in Table 4.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>S2_N</th> <th>S1_N</th> <th>S0_N</th> <th>Bus Cycle</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>Interrupt acknowledge</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>Read data from I/O</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>Write data to I/O</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>Halt</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>Instruction fetch</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Read data from memory</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>Write data to memory</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>None (passive)</td></tr> </tbody> </table> <p style="text-align: center;">Table 4. Bus Cycle Encoding</p>	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)
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)																																			
<b>S6/PIO29</b>	<b>Bus Cycle Status Bit 6 (output)</b>	During the second and remaining periods of a cycle (t2, t3, and t4), this pin is asserted High to indicate a DMA-initiated bus cycle. During a bus hold or reset condition, S6 floats. If S6 is to be used as PIO29 in input mode, the device driving PIO29 must not drive the pin Low during power-on reset. S6/PIO29 defaults to a PIO input with pullup, so the pin does not																																				

		need to be driven High externally.
<b>SCAN_EN</b>	<b>Scan enable (input)</b>	SCAN_EN is used for scan test. Active high.
<b>SCAN_TEST</b>	<b>Scan test (input)</b>	SCAN_EN is used for scan test. Active high.
<b>SCL</b>	<b>I2C serial clock (3-state IN/OUT open drain)</b>	I2C Serial Clock input/output(open-drain)
<b>SCLK/PIO20</b>	<b>Serial Clock (output)</b>	This pin supplies the synchronous serial interface (SSI) clock to a slave device, allowing transmit and receive operations to be synchronized between the microcontroller and the slave. SCLK is derived from the microcontroller internal clock and then divided by 2, 4, 8, or 16 depending on register settings. An access to any of the SSR or SSD registers activates SCLK for eight SCLK cycles. When SCLK is inactive, it is held High by the microcontroller.
<b>SDA</b>	<b>I2C Serial Data (3-state IN/OUT open drain)</b>	I2C Serial Data input/output(open-drain)
<b>SD_ADDR10 - SD_ADDR0</b>	<b>SDRAM control (output)</b>	This is the address bus connected to the SDRAM
<b>SD_CS_N</b>	<b>SDRAM control (output)</b>	This is the active low SDRAM chip select pin.
<b>SD_CAS_N</b>	<b>SDRAM control (output)</b>	This is the active low SDRAM column address select pin.
<b>SD_LAS_N</b>	<b>SDRAM control (output)</b>	This is the active low SDRAM low address select pin.
<b>SD_WR_N</b>	<b>SDRAM control (output)</b>	This is the active low SDRAM write pin.
<b>SD_DQM1 - SD_DQM0</b>	<b>SDRAM control (output)</b>	This is the active high byte mask signal. When high, no data can be read to written to the SDRAM byte.
<b>SDATA/PIO21</b>	<b>Serial Data (3-state I/O)</b>	This pin transmits synchronous serial interface (SSI) data to and from a slave device.
<b>SDEN1/PIO23, SDEN0/PIO22</b>	<b>Serial Data Enables (output)</b>	These pins enable data transfers on port 1 and port 0 of the synchronous serial interface (SSI). The microcontroller asserts either SDEN1 or SDEN0 at the beginning of a transfer and deasserts it after the transfer is complete. When SDEN1–SDEN0 are inactive, they are held Low by the microcontroller.
<b>SRDY/PIO6</b>	<b>Synchronous Ready (input)</b>	This pin indicates to the microcontroller that the addressed memory space or I/O device will complete a data transfer. The SRDY pin accepts an active High input synchronized to CLKOUTA. Using SRDY instead of ARDY allows relaxed system timing because of the elimination of the one-half clock period required to internally synchronize ARDY. To always assert the ready condition to the microcontroller, tie SRDY High. If the system does not use SRDY, tie the pin Low to yield control to ARDY.
<b>TEST_N</b>	<b>TEST for WAIT instruction (input)</b>	TEST_N is examined by the WAIT instruction. If the TEST_N input is high when "WAIT" execution begins, instruction execution will suspend. TEST_N will be re-sampled until it goes low, at which time execution will resume. If interrupts are enabled while the processor is waiting for TEST_N, interrupts will be serviced. This pin is synchronized internally.
<b>TMRIN0/PIO11</b>	<b>Timer Input 0 (input)</b>	This pin supplies a clock or control signal to the internal microcontroller timer 0. After internally synchronizing a Low-to-High transition on TMRIN0, the microcontroller increments the timer. TMRIN0 must be tied High if not being used.
<b>TMRIN1/PIO0</b>	<b>Timer Input 1 (input)</b>	This pin supplies a clock or control signal to the internal microcontroller timer 1. After internally synchronizing a Low-to-High transition on TMRIN1, the microcontroller increments the

		timer. TMRIN1 must be tied High if not being used.
<b>TMROUT0/PIO10</b>	<b>Timer Output 0 (output)</b>	This pin supplies the system with either a single pulse or a continuous waveform with a programmable duty cycle. TMROUT0 is floated during a bus hold or reset.
<b>TMROUT1/PIO1</b>	<b>Timer Output 1 (output)</b>	This pin supplies the system with either a single pulse or a continuous waveform with a programmable duty cycle. TMROUT1 can also be programmed as a watchdog timer. TMROUT1 is floated during a bus hold or reset.
<b>TXD/PIO27</b>	<b>Transmit Data (output)</b>	This pin supplies asynchronous serial transmit data to the system from the internal UART of the microcontroller.
<b>UCS_N</b>	<b>Upper Memory Chip Select (output)</b>	This pin indicates to the system that a memory access is in progress to the upper memory block. The base address and size of the upper memory block are programmable up to 512 Kbytes. UCS is held High during a bus hold condition. After power-on reset, UCS is asserted because the processor begins executing at FFFF0h and the default configuration for the UCS chip select is 64 Kbytes from F0000h to FFFFFh.
<b>UZI_N/PIO26</b>	<b>Upper Zero Indicate (output, synchronous)</b>	UZI—This pin lets the designer 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. UZI is the logical OR of the inverted A19 - A16 bits, and it asserts in the first period of a bus cycle and is held throughout the cycle. This signal should be pulled High.
<b>VDD</b>	<b>Digital Power supply (input)</b>	Digital Power Supply
<b>VDDA</b>	<b>Analog Power supply (input)</b>	Analog Power Supply for PLL
<b>VSSA</b>	<b>Analog ground(input)</b>	Analog Ground for PLL
<b>WHB_N</b>	<b>Write High Byte (3-state output)</b>	This pin and WLB_N indicate to the system which bytes of the data bus (upper, lower, or both) participate in a write cycle. In 80C186 designs, this information is provided by BHE_N, AD0, and WR_N. However, by using WHB_N and WLB_N, the standard system interface logic and external address latch that were required are eliminated. WHB_N is asserted with AD15 - AD8. WHB_N is the logical OR of BHE_N and WR_N. This pin floats during reset.
<b>WLB_N</b>	<b>Write Low Byte (3-state output)</b>	This pin and WHB_N indicate to the system which bytes of the data bus (upper, lower, or both) participate in a write cycle. In 80C186 designs, this information is provided by BHE_N, AD0, and WR_N. However, by using WHB_N and WLB_N, the standard system interface logic and external address latch that were required are eliminated. WLB_N is asserted with AD7 - AD0. WLB_N is the logical OR of AD0 and WR_N. This pin floats during reset.
<b>WR_N</b>	<b>Write Strobe (3-state output)</b>	This pin indicates to the system that the data on the bus is to be written to a memory or I/O device. WR_N floats during a bus hold or reset condition
<b>X1</b>	<b>Crystal Input (input)</b>	This pin and the X2 pin provide connections for a fundamental mode or third-overtone parallel-resonant crystal used by the internal oscillator circuit. To provide the microcontroller with an external clock source, connect the source to the X1 pin and leave the X2 pin unconnected.
<b>X2</b>	<b>Crystal Output (output)</b>	This pin and the X1 pin provide connections for a fundamental mode or third-overtone parallel-resonant crystal used by the internal oscillator circuit. To provide the microcontroller with an external clock source, leave the X2 pin unconnected and connect the source to the X1 pin.

### 3.5 Architecture Overview

The IMS16B core is comprised of two major functions the Execution Unit (EU), and the Bus Interface Unit (BIU). Both units operate independent and asynchronous to the other. The EU is responsible for instruction execution and effective address calculations. The BIU performs all bus cycles and maintains the instruction queue.

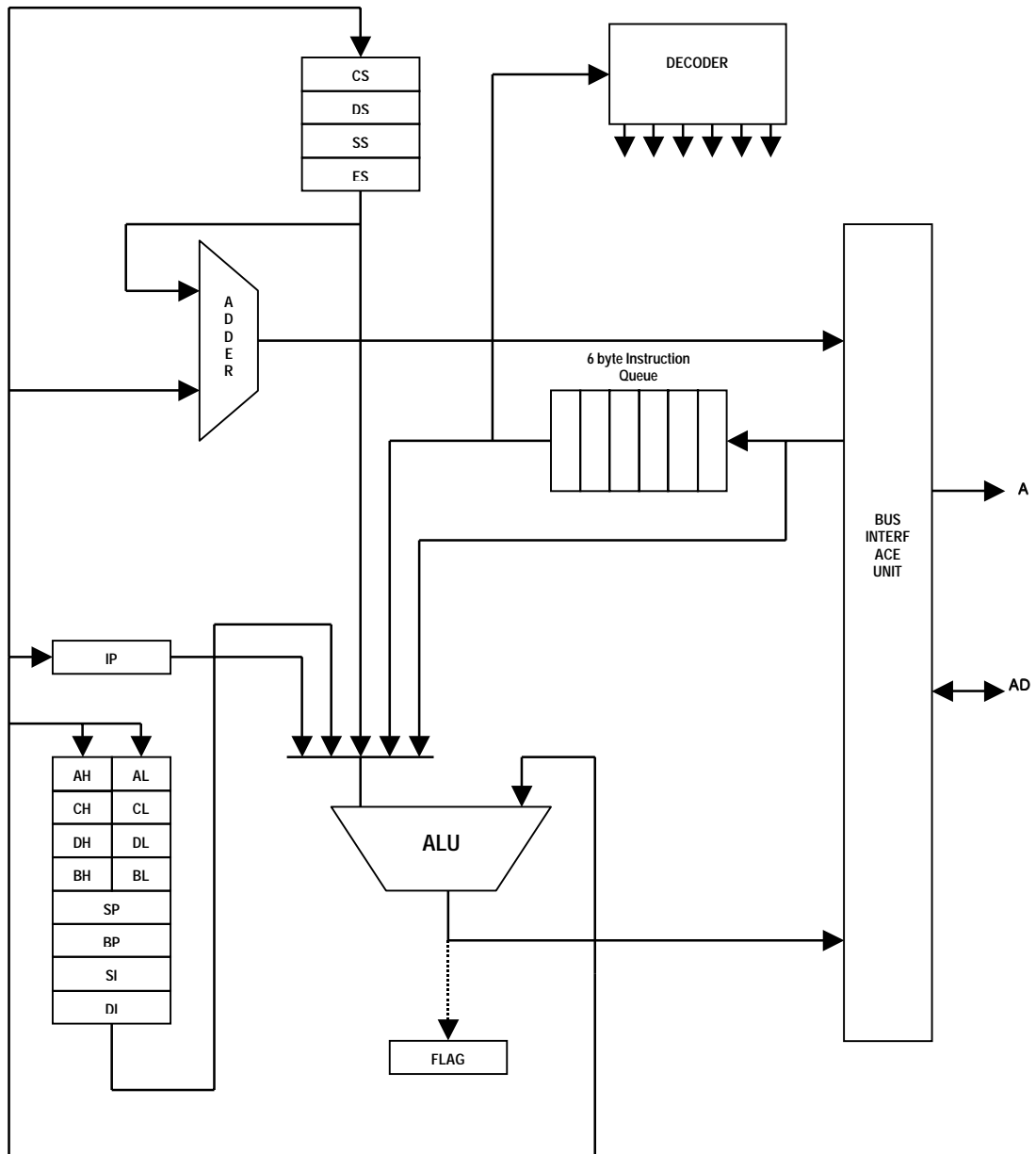


Figure 3.5 IMS16B Core Architecture

### 3.6 Bus Operation

The industry-standard 80C186 and 80C188 microcontrollers use a multiplexed address and data (AD) bus. The address is present on the AD bus only during the t1 clock phase. The IMS16B provides the multiplexed AD bus and a non-multiplexed address (A) bus. The A bus provides an address to the system for the complete bus cycle (t1 - t4).

Read and write bus cycles can be performed to memory or I/O space. Figure 3.6.1 and Figure 3.6.2 shows the affected signals during a normal read or write operation for an IMS16B microcontroller. The address and data will be multiplexed onto the AD bus.

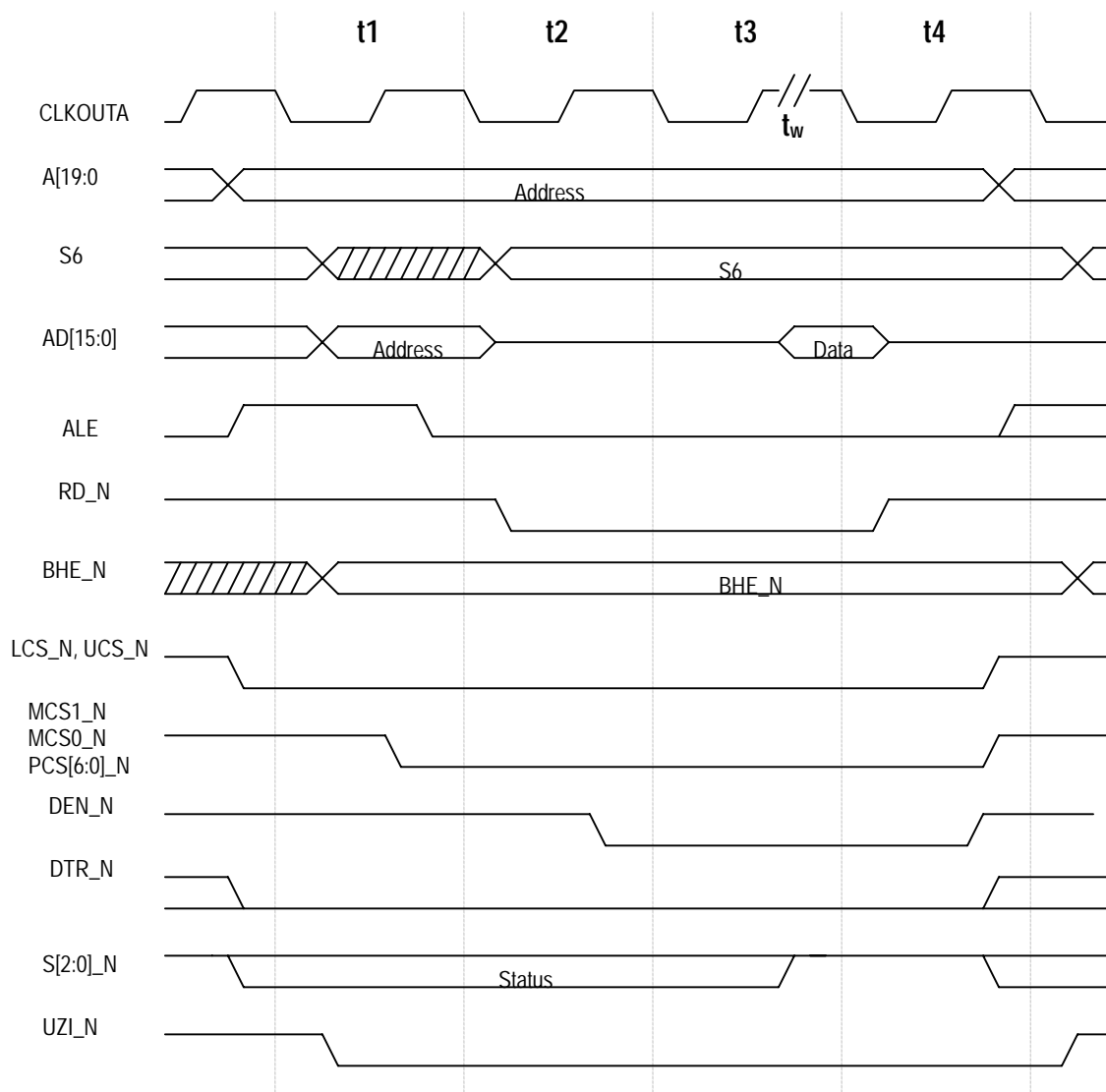


Figure 3.6.1 IMS16B Read Cycle Waveforms

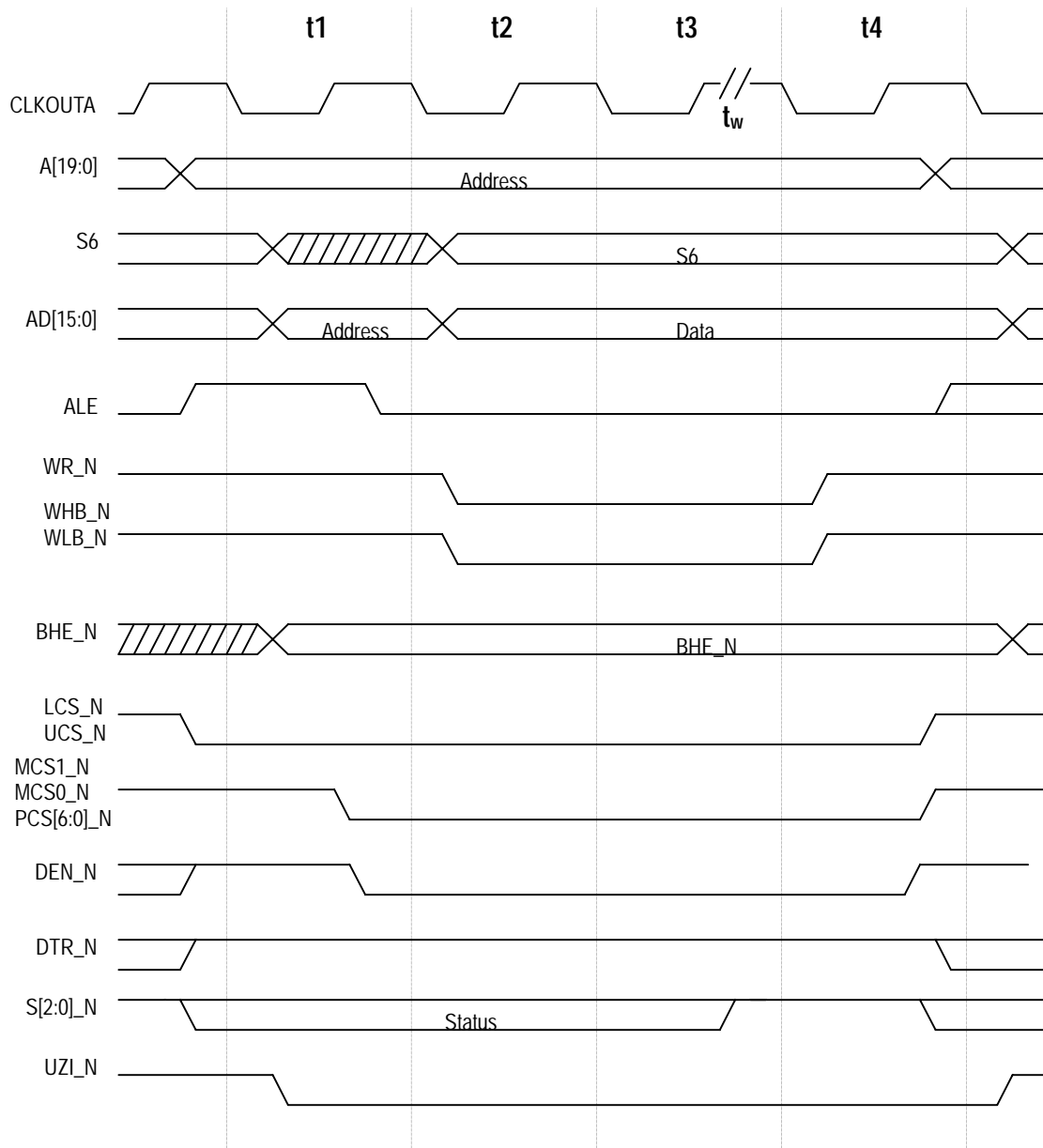


Figure 3.6.2 IMS16B Write Cycle Waveforms

# 4

## Programming

All members of the IMS16 series microprocessors, including the IMS16B, contain the same basic set of registers, instructions, and addressing modes, and are compatible with the original industry-standard 186/188 parts.

### 4.1 Register Set

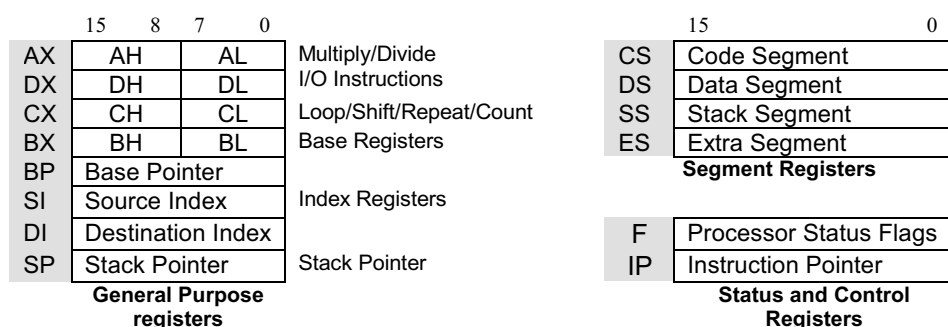


Figure 4.1 Programming Model

The base architecture of the IMS16B has 14 registers, as shown in Figure 4.1. These registers are grouped into the following categories:

■ **General Purpose Registers**

Eight 16-bit general purpose registers can be used for arithmetic and logical operands. Four of these (AX, BX, CX, and DX) can be used as 16-bit registers or split into pairs of separate 8-bit registers (AH, AL, BH, BL, CH, CL, DH, and DL). The Destination Index (DI) and Source Index (SI) registers are used for data movement and string instructions. The Base Pointer (BP) and Stack Pointer (SP) registers are used for the stack segment and point to the bottom and top of the stack, respectively.

Some of the general purpose registers also have dedicated functions with certain instructions as noted in Figure 4.1. The AX and DX registers are used during multiply, divide and I/O instructions. The CX register is used during instructions that have a repeat count for loop, shift and repeat. BX and BP can be used as base pointers used during effective address calculations. SI and DI can be used as index registers during effective address calculations.

Four of the general-purpose registers (BP, BX, DI, and SI) can also be used to determine offset addresses of operands in memory. These registers can contain base addresses or indexes to particular locations within a segment. The addressing mode selects the specific registers for operand and address calculations.

All stack operations (POP, POPA, POPF, PUSH, PUSHA, PUSHF) utilize the stack pointer. The Stack Pointer register is always offset from the Stack Segment (SS) register, and no segment override is allowed.

■ **Segment Registers**

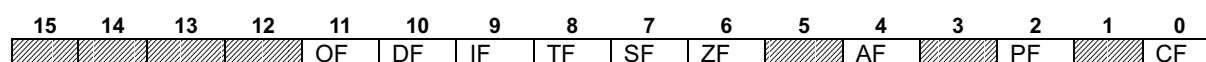
Four 16-bit special-purpose registers (CS, DS, ES, and SS) select, at any given time, the segments of memory that are immediately addressable for code (CS), data (DS and ES), and stack (SS) memory. CS is used for instruction prefetch and immediate data. DS is used for all other data references. SS is used for all stack pushes and pops; any memory reference that uses BP as a base register. ES is used for all string instruction references that use the DI register as an index.

■ **Status and Control Registers**

Two 16-bit special-purpose registers record or alter certain aspects of the processor state - the Instruction Pointer (IP) register contains the offset address of the next sequential instruction to be executed and the Processor Status Flags (FLAGS) register contains status and control flag bits.

### 4.2. Processor Status Flags Register

The 16-bit processor Status Flags register records specific characteristics of the result of logical and arithmetic instructions and controls the operation of the microcontroller within a given operating mode. After an instruction is executed, the value of the flags may be set, cleared/reset, unchanged, or undefined. The term undefined means that the flag value prior to the execution of the instruction is not preserved, and the value of the flag after the instruction is executed cannot be predicted.



Bit	Name	Function
11	Overflow Flag	Set if the signed result cannot be expressed within the number of bits in the destination operand, cleared otherwise
10	Direction Flag	Used during string instructions to determine auto decrement (DF=1), or auto increment (DF=0) of index value.
9	Interrupt-Enable Flag	When set, enables maskable interrupts to cause the CPU to transfer control to a location specified by an interrupt vector.
8	Trace Flag	When set, a trace interrupt occurs after instructions execute. TF is cleared by the trace interrupt after the processor status flags are pushed onto the stack. The trace service routine can continue tracing by popping the flags back with an interrupt return (IRET) instruction.
7	Sign Flag	Set equal to high-order bit of result (0 if 0 or positive, 1 if negative).
6	Zero Flag	Set if result is 0; cleared otherwise.
4	Auxiliary Carry	Set on carry from or borrow to the low-order 4 bits of the AL general-purpose register; cleared otherwise.
2	Parity Flag	Set if low-order 8 bits of result contain an even number of 1's; cleared otherwise.
0	Carry Flag	Set on high-order bit carry or borrow; cleared otherwise.

**Figure 4.2 Status Flags Register**

### 4.3. Memory Organizations and Address Generation

Memory is organized in sets of segments. Each segment is a linear contiguous sequence of 64K (2<sup>16</sup>) 8-bit bytes. Memory is addressed using a two-component address that consists of a 16-bit segment value and a 16-bit offset. The offset is the number of bytes from the beginning of the segment (the segment address), to the data or instruction that is being accessed.

The processor forms the physical address of the target location by taking the segment address, shifting

it to the left 4 bits (multiplying by 16), and adding this to the 16-bit offset. The result is the 20-bit address of the target data or instruction. This allows for a 1-Mbyte physical address size. Figure 4.3 shows physical address generation example.

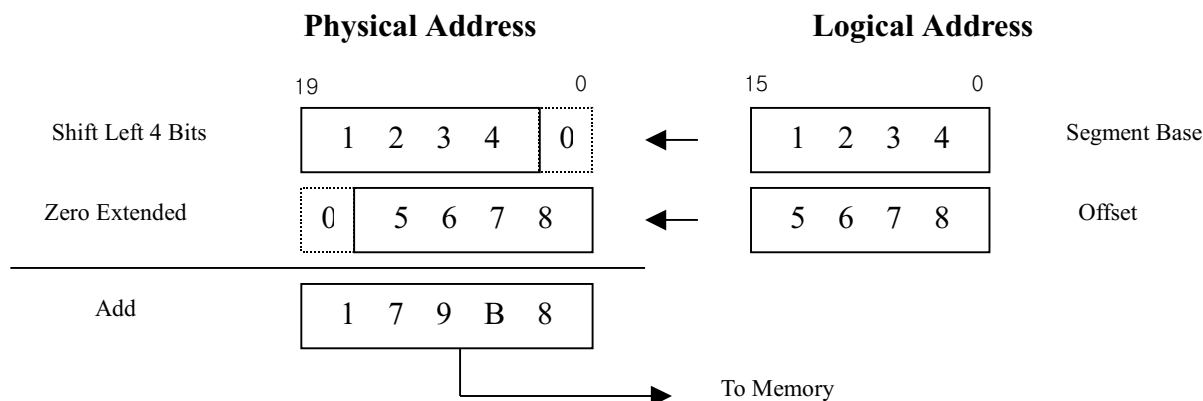


Figure 4.3 Physical Address Generation

### 4.4 Segments

The IMS16B uses four segment registers:

1. **Data Segment (DS):** The processor assumes that all accesses to the program's variables are from the 64K space pointed to by the DS register. The data segment holds data, operands, etc.
2. **Code Segment (CS):** This 64K space is the default location for all instructions. All code must be executed from the code segment.
3. **Stack Segment (SS):** The processor uses the SS register to perform operations that involve the stack, such as pushes and pops. The stack segment is used for temporary space.
4. **Extra Segment (ES):** Usually this segment is used for large string operations and for large data structures. Certain string instructions assume the extra segment as the segment portion of the address. The extra segment is also used (by using segment override) as a spare data segment.

When a segment is not defined for a data movement instruction, it's assumed to be a data segment. An instruction prefix can be used to override the segment register. For speed and compact instruction encoding, the segment register used for physical address generation is implied by the addressing mode used (see Figure 4.4).

Memory Reference Needed	Segment Register Used	Implicit Segment Selection Rule
Local Data	Data(DS)	All data references
Instructions	Code(CS)	Instructions, Immediate data
Stack	Stack(SS)	All stack pushes and pops Any memory references that use the BP register
External Data(Global)	Extra(ES)	All string instruction references that use the DI register as an index

**Figure 4.4 Segment Register Selection Rules**

**4.5. Addressing Modes**

The IMS16B microcontroller uses eight categories of addressing modes to specify operands.  
 - Two addressing modes are provided for instructions that operate on register or immediate operands.  
 - Six addressing modes are provided to specify the location of an operand in a memory segment.

**Register and Immediate Operands**

Register Operand Mode : The operand is located in one of the 8- or 16-bit registers.

Immediate Operand Mode :The operand is included in the instruction.

**Memory Operands**

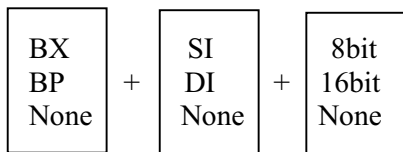
A memory-operand address consists of two 16-bit components: a segment value and an offset. The segment value is supplied by a 16 bit segment register either implicitly chosen by the addressing mode or explicitly chosen by a segment override prefix.

The offset, also called the effective address, is calculated by summing any combination of the following three address elements:

1. Displacement : an 8-bit or 16-bit immediate value contained in the instruction
2. Base : contents of either the BX or BP base registers
3. Index : contents of either the SI or DI index registers

Any carry from the 16-bit addition is ignored. Eight-bit displacements are sign-extended to 16-bit values.

Combinations of the above three address elements define the following six memory addressing modes:



1. Direct Mode : The operand offset is contained in the instruction as an 8- or 16-bit displacement element.
2. Register Indirect Mode : The operand offset is in one of the registers BP, BX, DI, or SI.
3. Based Mode : The operand offset is the sum of an 8- or 16-bit displacement and the contents of a base register (BX or BP).
4. Indexed Mode : The operand offset is the sum of an 8- or 16-bit displacement and the contents of an index register (DI or SI).
5. Based Indexed Mode : The operand offset is the sum of the contents of a base register (BP or BX) and an index register (DI or SI).
6. Based Indexed Mode with Displacement : The operand offset is the sum of a base register's contents, an index register's contents, and an 8-bit or 16-bit displacement.

Addressing Mode	Example
Direct	mov ax, ds:4
Register Indirect	mov ax, [si]
Based	mov ax, [bx]4
Indexed	mov ax, [si]4
Based Indexed	mov ax, [si][bx]
Based Indexed with displacement	mov ax, [si][bx]4

### Figure 4.5 Memory Addressing Mode Examples

## 4.6. I/O Space

The I/O space consists of 64K 8-bit or 32K 16-bit ports. The IN and OUT instructions address the I/O space with either an 8-bit port address specified in the instruction, or a 16-bit port address in the DX register. Eight-bit port addresses are zero-extended so that A15–A8 are Low. I/O port addresses 00F8h through 00FFh are reserved.

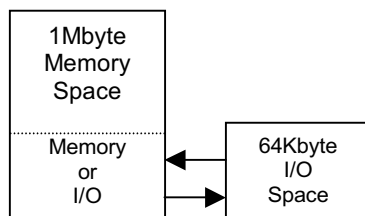


Figure 4.6 Memory and I/O Space

## 4.7. Data Types

The IMS16B supports the following data types:

### Integer

A signed binary numeric value contained in an 8-bit byte or a 16-bit word. All operations assume a two's complement representation.

### Ordinal

An unsigned binary numeric value contained in an 8-bit byte or a 16-bit word.

### Double Word

A signed binary numeric value contained in two sequential 16-bit addresses, or in a DX:AX register pair.

### Quad Word

A signed binary numeric value contained in four sequential 16-bit addresses.

### BCD

An unpacked byte representation of the decimal digits 0-9.

### ASCII

A byte representation of alphanumeric and control characters using the ASCII standard of character representation.

### Packed BCD

A packed byte representation of two decimal digits (0-9). One digit is stored in each nibble (4 bits) of the byte.

### String

A contiguous sequence of bytes or words. A string can contain from 1 byte up to 64 Kbytes.

### Pointer

16-bit or 32-bit quantities composed of a 16-bit offset component or a 16-bit segment base component plus a 16-bit offset component.

In general, individual data elements must fit within defined segment limits.

### 4.8. Instruction Set

This chapter contains descriptions of the brief overview of IMS16B Instruction set constitution. The IMS16B shares the standard 186 instruction set. An instruction can reference from zero to several operands. An operand can reside in a register, in the instruction itself, or in memory.

OPCODE w	mod reg r/m	(disp-low)	(disp-high)	(data)	(data)
1 byte or 2bytes		8bit or 16bit displacement if memory operand needs.		8bit or 16bit immediate data	

r/m	mod				
	00	01	10	11	
				w=0	w=1
000	(BX) + (SI)	(BX) + (SI) + D8	(BX) + (SI) + D16	AL	AX
001	(BX) + (DI)	(BX) + (DI) + D8	(BX) + (DI) + D16	CL	CX
	(BP) + (SI)	(BP) + (SI) + D8	(BP) + (SI) + D16	DL	DX
011	(BP) + (DI)	(BP) + (DI) + D8	(BP) + (DI) + D16	BL	BX
100	(SI)	(SI) + D8	(SI) + D16	AH	SP
101	(DI)	(DI) + D8	(DI) + D16	CH	BP
110	DIRECT	(BP) + D8	(BP) + D16	DH	SI
111	(BX) + [DI]	(BX) + D8	(BX) + D16	BH	DI

reg	w = 0	w = 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

Segment Register	Code
ES	00
CS	01
SS	10
DS	11

AAA	ASCII adjust for addition	AAD	ASCII adjust for division
AAM	ASCII adjust for multiplication	AAS	ASCII adjust for subtraction
ADC	Add byte or word with carry	ADD	Add byte or word
AND	Logical AND byte or word	BOUND	Detects values outside prescribed range
CALL	Call procedure	CBW	Convert byte to word
CLC	Clear carry flag	CLD	Clear direction flag
CLI	Clear interrupt-enable flag	CMC	Complement carry flag
CMP	Compare byte or word	CMPS	Compare byte or word string
CWD	Convert word to double word	DAA	Decimal adjust for addition
DAS	Decimal adjust for subtraction	DEC	Decrement byte or word by 1
DIV	Divide byte or word unsigned	ENTER	Format stack for procedure entry
ESC	Escape to extension processor	HLT	Halt until interrupt or reset
IDIV	Integer divide byte or word	IMUL	Integer multiply byte or word
IN	Input byte or word	INC	Increment byte or word by 1
INS	Input bytes or word string	INT	Interrupt
INTO	Interrupt if overflow	IRET	Interrupt return
JA/JNBE	Jump if above/not below or equal	JAE/JNB	Jump if above or equal/not below
JB/JNAE	Jump if below/not above or equal	JBE/JNA	Jump if below or equal/not above

JC	Jump if carry	JCXZ	Jump if register CX = 0
JE/JZ	Jump if equal/zero	JG/JNLE	Jump if greater/not less or equal
JGE/JNL	Jump if greater or equal/not less	JL/JNGE	Jump if less/not greater or equal
JLE/JNG	Jump if less or equal/not greater	JMP	Jump
JNC	Jump if not carry	JNE/JNZ	Jump if not equal/not zero
JNO	Jump if not overflow	JNP/JPO	Jump if not parity/parity odd
JNS	Jump if not sign	JO	Jump if overflow
JP/JPE	Jump if parity/parity even	JS	Jump if sign
LAHF	Load AH register from flags	LDS	Load pointer using DS
LEA	Load effective address	LEAVE	Restore stack for procedure exit
LES	Load pointer using ES	LOCK	Lock bus during next instruction
LODS	Load byte or word string	LOOP	Loop
LOOPE/ LOOPZ	Loop if equal/zero	LOOPNE/ LOOPNZ	Loop if not equal/not zero
MOV	Move byte or word	MOVS	Move byte or word string
MUL	Multiply byte or word unsigned	NEG	Negate byte or word
NOP	No operation	NOT	Logical NOT byte or word
OR	Logical Inclusive OR byte or word	OUT	Output byte or word
POP	Pop word off stack	POPA	Pop all general register off stack
POPF	Pop flags off stack	PUSH	Push word onto stack
PUSHA	Push all general registers onto stack	PUSHF	Push flags onto stack
RCL	Rotate left through carry byte or word	RCR	Rotate right through carry byte or word
REP	Repeat	REPE/REPZ	Repeat while equal/zero
REPNE/ REPNZ	Repeat while not equal/not zero	RETO	Return from procedure
ROL	Rotate left byte or word	ROR	Rotate right byte or word
SAHF	Store AH register in flags SF, ZF, AF, PF, and CF	SAL	Shift left arithmetic byte or word
SAR	Shift right arithmetic byte or word	SBB	Subtract byte or word with borrow
SCAS	Scan byte or word string	SHL	Shift left logical byte or word
SHR	Shift right logical byte or word	STC	Set carry flag
STD	Set direction flag	STI	Set interrupt-enable flag
STOS	Store byte or word string	SUB	Subtract byte or word
TEST	Test (Logical AND, flags only set) byte or word	XCHG	Exchange byte or word
XLAT	Translate byte	XOR	Logical exclusive OR byte or word

Figure 4.8 Instruction Set of IMS16B

## 5

## Peripherals

This section documents the operation of the IMS16B integrated peripherals. The IMS16B includes the following peripherals:

- Peripheral Interface Logic
- Chip-Select and Ready Control Logic
- 2 Channel DMA controllers
- 3 Programmable 16-bit Timers
- Interrupt Controller
- Refresh Control Unit
- Power-Save Logic
- I2C controller
- SDRAM Controller

The peripheral control block can be mapped into either memory or I/O space. The base address of the control block must be on an even 256-byte boundary (i.e., the lower eight bits of the base address are 00h). Internal logic recognizes control block addresses and responds to bus cycles. During bus cycles to internal registers, the bus controller signals the operation externally (i.e., the RD, WR, status, address, and data lines are driven as in a normal bus cycle), but the data bus, SRDY, and ARDY are ignored.

At reset, the Peripheral Control Block Relocation register is set to 20FFh, which maps the control block to start at FF00h in I/O space. An offset map of the 256-byte peripheral control register block is shown in Figure 5

Offset (hex)	Register Name
FE	Peripheral Control Block Relocation Register
FC	I2C Hold Count Register
FA	I2C Div Count Register
F8	I2C Data Register
F6	I2C Status Register
F4	I2C Control Register
F2	I2C Address Register
F0	PDCON Power-Save Control Register
EE	SDRAM Mode Set Register
EC	SDRAM Auto Refresh Duty Cycle Register
EA	SDRAM Enable Register
E4	Enable RCU Register
E2	Clock Prescaler Register
E0	Memory Partition Register
DA	DMA 1 Control Register
D8	DMA 1 Transfer Count Register
D6	DMA1 Destination Address High Register

D4	DMA 1 Destination Address Low Register
D2	DMA 1 Source Address High Register
D0	DMA 1 Source Address Low Register
CA	DMA 0 Control Register
C8	DMA 0 Transfer Count Register
C6	DMA0 Destination Address High Register
C4	DMA 0 Destination Address Low Register
C2	DMA 0 Source Address High Register
C0	DMA 1 Source Address Low Register
A8	PCS_N AND MCS_N Auxiliary Register
A6	Midrange Memory Chip Select Register
A4	Peripheral Chip Select Register
A2	Low Memory Chip Select Register
A0	Upper Memory Chip Select Register
88	Serial Port Baud Rate Divisor Register
86	Serial Port Receive Register
84	Serial Port Transmit Register
82	Serial Port Status Register
80	Serial Port Control Register
7A	PIO Data 1 Register
78	PIO Direction 1 Register
76	PIO Mode 1 Register
74	PIO Data 0 Register
72	PIO Direction 0 Register
70	PIO Mode 0 Register
66	Timer 2 Mode/Control Register
62	Timer 2 Max count Compare A Register
60	Timer 2 Count Register
5E	Timer 1 Mode/Control Register
5C	Timer 1 Max count Compare B Register
5A	Timer 1 Max count Compare A Register
58	Timer 1 Count Register
56	Timer 0 Mode/Count Register
54	Timer 0 Max count Compare B Register
52	Timer 0 Max count Compare A Register
50	Timer 0 Count Register
4A	I2C Interrupt Controller Register
44	UART Serial Interrupt Control Register
42	Watchdog Timer Control Register
40	INT4 Control Register
3E	INT3 Control Register
3C	INT2 Control Register
3A	INT1 Control Register
38	INT0 Control Register
36	DMA 1 Interrupt Control Register
34	DMA 0 Interrupt Control Register
32	Timer Interrupt Control Register
30	Interrupt Status Register
2E	Interrupt Request Register
2C	In-service Register
2A	Priority Mask Register
28	Interrupt Mask Register
26	Poll Status Register

24	Poll Register
22	End-of Interrupt Register
18	Synchronous Serial Receive Register
16	Synchronous Serial Transmit 0 Register
14	Synchronous Serial Transmit 1 Register
12	Synchronous Serial Enable Register
10	Synchronous Serial Status Register

**Figure 5 IMS16B Peripheral Control Block Register Map**

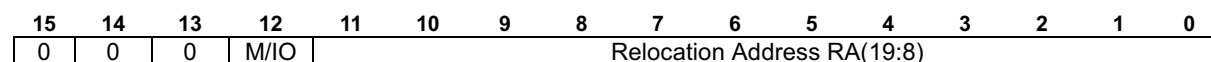
# 6

## Peripheral Control Register

### 6.1 Peripheral Control Block Relocation Register (RELREG, Offset FEh)

The peripheral control block is mapped into either memory or I/O space by programming the Peripheral Control Block Relocation (RELREG) register. This register is a 16-bit register at offset FEh from the control block base address. The RELREG register provides the upper 12 bits of the base address of the control block. The control block is effectively an internal chip select range. Other chip selects can overlap the control block only if they are programmed to zero wait states and ignore external ready.

If the control register block is mapped into I/O space, the upper four bits of the base address must be programmed as 0000b (since I/O addresses are only 16 bits wide). In addition to providing relocation information for the control block, the RELREG register contains a bit that places the interrupt controller into either slave mode or master mode. At reset, the RELREG register is set to 00FFh, which maps the control block to start at FF00h in I/O space.



**Reset value : 0000\_0000\_1111\_1111b**

Bit	Name	Function
12	Memory/I/O Space	When set to 1, the peripheral control block(PCB) is located in memory space. When set to 0, the PCB is located in I/O space.
11-0	Relocation Address Bits	RA19–RA8 define the upper address bits of the PCB base address. The lower eight bits (RA7–RA0) default to 00h. RA19–RA16 are ignored when the PCB is mapped to I/O space.

**Figure 6.1 Peripheral Control Block Relocation Register**

## 6.2 Power-Save Control Register (PDCON, Offset F0h)

The value of the PDCON register is 0000h at reset.

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

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function																																				
15	Enable Power-Save mode	When set to 1, enables Power-Save mode and divides the internal operating clock by the value in F2–F0. PSEN is automatically cleared when an external interrupt, including those generated by on-chip peripheral devices, occurs. The value of the PSEN bit is not restored by the execution of an IRET instruction. Software interrupts (INT instruction) and exceptions do not clear the PSEN bit, and interrupt service routines for these conditions should do so if desired. This bit is 0 after processor reset.																																				
11	CLKOUTB Output Frequency	When set to 1, CLKOUTB follows the crystal input (PLL) frequency. When set to 0, CLKOUTB follows the internal processor frequency (after the clock divisor). Set to 0 on reset.																																				
10	CLKOUTB Drive Disable	When set to 1, CBD three-states the clock output driver for CLKOUTB. When set to 0, CLKOUTB is driven as an output. Set to 0 on reset.																																				
9	CLKOUTA Output Frequency	When set to 1, CLKOUTA follows the crystal input (PLL) frequency. When set to 0, CLKOUTA follows the internal processor frequency (after the clock divisor). Set to 0 on reset.																																				
8	CLKOUTA Drive Disable	When set to 1, CAD three-states the clock output driver for CLKOUTA. When set to 0, CLKOUTA is driven as an output. Set to 0 on reset.																																				
2-0	Clock Divisor Select	Controls the division factor when Power-Save mode is enabled. Allowable values are as follows: <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>F2</th><th>F1</th><th>F0</th><th>Divider Factor</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>Divide by 2</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>Divide by 8</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>Divide by 16</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>Divide by 32</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>Divide by 64</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Divide by 128</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>Divide by 256</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>Divide by 512</td></tr> </tbody> </table>	F2	F1	F0	Divider Factor	0	0	0	Divide by 2	0	0	1	Divide by 8	0	1	0	Divide by 16	0	1	1	Divide by 32	1	0	0	Divide by 64	1	0	1	Divide by 128	1	1	0	Divide by 256	1	1	1	Divide by 512
F2	F1	F0	Divider Factor																																			
0	0	0	Divide by 2																																			
0	0	1	Divide by 8																																			
0	1	0	Divide by 16																																			
0	1	1	Divide by 32																																			
1	0	0	Divide by 64																																			
1	0	1	Divide by 128																																			
1	1	0	Divide by 256																																			
1	1	1	Divide by 512																																			

Figure 6.2 Power-Save Control Register

## 6.3 Initializations and Processor Reset

Processor initialization or startup is accomplished by driving the RES\_N input pin Low. RES\_N must be Low during power-up to ensure proper device initialization. RES\_N forces the IMS16B microcontroller to terminate all execution and local bus activity. No instruction or bus activity occurs as long as RES\_N is active. After RES\_N is deasserted and an internal processing interval elapses, the microcontroller begins execution with the instruction at physical location FFFF0h.

## 7

## Chip Select Unit

The IMS16B microcontroller contains logic that provides programmable chip select generation for both memories and peripherals. In addition, the logic can be programmed to provide ready or wait-state generation and latched address bits A1 and A2. The chip select lines are active for all memory and I/O cycles in their programmed areas, whether they are generated by the CPU or by the integrated DMA unit. The IMS16B microcontroller provides six chip select outputs for use with memory devices and six more for use with peripherals in either memory space or I/O space. The six memory chip selects can be used to address three memory ranges. Note that only one chip select may be programmed to be active for any memory location at a time. (Chip selects cannot overlap)

Offset	Register Mnemonic	Register Name	Affected Pins	Comments
A0h	UMCS	Upper Memory Chip Select	UCS_N	Ending address is fixed at FFFFh
A2h	LMCS	Lower Memory Chip Select	LCS_N	Starting address is fixed at 00000h
A4h	PACS	Peripheral Chip Select	PCS6_N - PCS5_N PCS3_N - PCS0_N	Block size is fixed at 256 bytes
A6h	MMCS	Midrange Chip Select	MCS3_N - MCS0_N	Starting address and block size are programmable
A8h	MPCS	PCS and MCS Auxiliary	PCS6_N - PCS5_N PCS3_N - PCS0_N MCS3_N - MCS0_N	Affects both PCS and MCS chip selects

**Figure 7 Chip Select Register Summary**

Except for the UCS chip select, chip selects are not activated until the associated registers have been accessed. (An access is any read or write operation.) For this reason, the chip select registers should not be read by the processor initialization code until after they have been written with valid data. The LCS chip select is activated when the LMCS register is accessed, the MCS chip selects are activated after both the MMCS and MPCS registers have been accessed, and the PCS chip selects are activated after both the PACS and MPCS registers have been accessed.

### 7.1 Upper Memory Chip Select Register (UMCS, Offset A0h)

The IMS16B provides the UCS\_N chip select pin for the top of memory. On reset, the microcontroller begins fetching and executing instructions starting at memory location FFFF0h, so upper memory is usually used as instruction memory. To facilitate this usage, UCS\_N defaults to active on reset with a default memory range of 64Kbytes from F0000h to FFFFh, external ready required, and three wait states automatically inserted.

The UCS\_N memory range always ends at FFFFh. The lower boundary is programmable. The Upper Memory Chip Select is configured through the UMCS register.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	LB2	LB1	LB0	1	1	1	1	1	1	1	1	R2	R1	R0

Reset value : 1111\_1111\_1111\_1011b

Bit	Name	Function																				
13-11	Lower Boundary	The LB2-LB0 bits define the lower bound of the memory accessed through the UCS_N chip selects. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Memory Block Size</th> <th>Starting Address</th> <th>LB[2:0]</th> <th>Comments</th> </tr> </thead> <tbody> <tr> <td>64K</td> <td>F0000h</td> <td>111b</td> <td>Default</td> </tr> <tr> <td>128K</td> <td>E0000h</td> <td>110b</td> <td></td> </tr> <tr> <td>256K</td> <td>C0000h</td> <td>100b</td> <td></td> </tr> <tr> <td>512K</td> <td>80000h</td> <td>000b</td> <td></td> </tr> </tbody> </table>	Memory Block Size	Starting Address	LB[2:0]	Comments	64K	F0000h	111b	Default	128K	E0000h	110b		256K	C0000h	100b		512K	80000h	000b	
Memory Block Size	Starting Address	LB[2:0]	Comments																			
64K	F0000h	111b	Default																			
128K	E0000h	110b																				
256K	C0000h	100b																				
512K	80000h	000b																				
2	Ready Mode	The R2 bit is used to configure the ready mode for the UCS_N chip select. If R2 is set to 0, external ready is required. If R2 is set to 1, external ready is ignored. In each case, the processor also uses the value of the R1-R0 bits to determine the number of wait states to insert. R2 defaults to 0 at reset.																				
1-0	Wait-State Value	The value of R1-R0 determines the number of wait states inserted into an access to the UCS_N memory area. From zero to three wait states can be inserted (R1-R0 = 00b to 11b). R1-R0 default to 11b at reset.																				

Figure 7.1 Upper Memory Chip Select Register

## 7.2 Low Memory Chip Select Register (LMCS, Offset A2h)

The IMS16B microcontroller provides the LCS\_N chip select pin for the bottom of memory. Since the interrupt vector table is located at 00000h at the bottom of memory, the LCS\_N pin has been provided to facilitate this usage. The LCS\_N pin is not active on reset, but any read or write access to the LMCS register activates this pin.

The Low Memory Chip Select is configured through the LMCS register.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	UB2	UB1	UB0	0	0	0	0	0	1	1	1	R2	R1	R0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function																				
14-12	Upper Boundary	The UB2-UB0 bits define the upper bound of the memory accessed through the LCS_N chip selects. <table border="1" data-bbox="466 936 1252 1086"> <thead> <tr> <th>Memory Block Size</th><th>Ending Address</th><th>UB[2:0]</th><th>Comments</th></tr> </thead> <tbody> <tr> <td>64K</td><td>0FFFFh</td><td>000b</td><td>Default</td></tr> <tr> <td>128K</td><td>1FFFFh</td><td>001b</td><td></td></tr> <tr> <td>256K</td><td>3FFFFh</td><td>011b</td><td></td></tr> <tr> <td>512K</td><td>7FFFFh</td><td>111b</td><td></td></tr> </tbody> </table>	Memory Block Size	Ending Address	UB[2:0]	Comments	64K	0FFFFh	000b	Default	128K	1FFFFh	001b		256K	3FFFFh	011b		512K	7FFFFh	111b	
Memory Block Size	Ending Address	UB[2:0]	Comments																			
64K	0FFFFh	000b	Default																			
128K	1FFFFh	001b																				
256K	3FFFFh	011b																				
512K	7FFFFh	111b																				
2	Ready Mode	The R2 bit is used to configure the ready mode for the LCS_N chip select. If R2 is set to 0, external ready is required. If R2 is set to 1, external ready is ignored. In each case, the processor also uses the value of the R1-R0 bits to determine the number of wait states to insert. R2 defaults to 0 at reset.																				
1-0	Wait-State Value	The value of R1-R0 determines the number of wait states inserted into an access to the LCS_N memory area. From zero to three wait states can be inserted (R1-R0 = 00b to 11b). R1-R0 defaults to 00b at reset.																				

Figure 7.2 Lower Memory Chip Select Register

### 7.3 Midrange Memory Chip Select Register (MMCS, Offset A6h)

The IMS16B provides four chip select pins, MCS3\_N - MCS0\_N, for use within a user-locatable memory block. The base address of the memory block can be located anywhere within the 1Mbyte memory address space, exclusive of the areas associated with the UCS\_N and LCS\_N chip selects (and, if they are mapped to memory, the address range of the Peripheral Chip Selects, PCS6\_N - PCS5\_N and PCS3\_N - PCS0\_N). The MCS address range can overlap the PCS address range if the PCS chip selects are mapped to I/O space.

The Midrange Memory Chip Selects are programmed through two registers. The Midrange Memory Chip Select (MMCS) register determines the base address and the ready condition and wait states of the memory block accessed through the MCS pins. The PCS and MCS Auxiliary (MPCS) register is used to configure the block size. The MCS3\_N - MCS0\_N pins are not active on reset. Both the MMCS and MPCS registers must be accessed with a read or write to activate these chip selects.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA19	BA18	BA17	BA16	BA15	BA14	BA13	1	1	1	1	1	1	R2	R1	R0

**Reset value : 0000\_0000\_0000\_0000b**

Bit	Name	Function
15-9	Base Address	The base address of the memory block that is addressed by the MCS chip select pins is determined by the value of BA19 - BA13. These bits correspond to bits A19 - A13 of the 20-bit memory address. Bits A12 - A0 of the base address are always 0.
2	Ready Mode	The R2 bit is used to configure the ready mode for the MCS_N chip select. If R2 is set to 0, external ready is required. If R2 is set to 1, external ready is ignored. In each case, the processor also uses the value of the R1 - R0 bits to determine the number of wait states to insert. R2 defaults to 0 at reset.
1-0	Wait-State Value	The value of R1 - R0 determines the number of wait states inserted into an access to the MCS_N memory area. From zero to three wait states can be inserted (R1 - R0 = 00b to 11b). R1 - R0 defaults to 00b at reset.

**Figure 7.3 Midrange Memory Chip Select Register**

The base address can be set to any integer multiple of the size of the memory block size selected in the MPCS register. For example, if the midrange block is 32 Kbytes, the block could be located at 10000h or 18000h but not at 14000h. The base address of the midrange chip selects can be set to 00000h only if the LCS chip select is not active. This is due to the fact that the LCS base address is defined to be address 00000h and chip select address ranges are not allowed to overlap. Because of the additional restriction that the base address must be a multiple of the block size, a 512K MMCS block size can only be used when located at address 00000h, and the LCS chip selects must not be active in this case. Use of the MCS chip selects to access low memory allows the timing of these accesses to follow the AD address bus rather than the A address bus. Locating a 512K MMCS block at 80000h always conflicts with the range of the UCS chip select and is not allowed.

### 7.4 PCS and MCS Auxiliary Register (MPCS, Offset A8h)

The PCS and MCS Auxiliary (MPCS) register differs from the other chip select control registers in that it contains fields that pertain to more than one type of chip select. The MPCS register fields provide program information for MCS3\_N - MCS0\_N as well as PCS6\_N - PCS5\_N and PCS3\_N - PCS0\_N. In addition to its function as a chip select control register, the MPCS register contains a field that configures the PCS6\_N - PCS5\_N pins as either chip selects or as alternate sources for the A2 and A1 address bits. When programmed to provide address bits A1 and A2, PCS6\_N - PCS5\_N cannot be used as peripheral chip selects. These outputs can be used to provide latched address bits for A2 and A1. On reset, PCS6\_N - PCS5\_N are not active. If PCS6\_N - PCS5\_N are configured as address pins, an access to the MPCS register causes the pins to activate. No corresponding access to the PACS register is required to activate the PCS6\_N - PCS5\_N pins as addresses.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	M6	M5	M4	M3	M2	M1	M0	EX	MS	1	1	1	R2	R1	R0

Reset value : 0000\_0001\_0000\_0000b

Bit	Name	Function																																
14 -8	MCS_N Block Size	<p>This field determines the total block size for the MCS3 - MCS0 chip selects. Each individual chip select is active for one quarter of the total block size. Only one of the M6 - M0 bits can be set at any time. If more than one of the M6 - M0 bits is set, unpredictable operation of the MCS lines occurs.</p> <table border="1"> <thead> <tr> <th>Total Block Size</th> <th>Individual Select Size</th> <th>M6 - M0</th> <th>Comments</th> </tr> </thead> <tbody> <tr> <td>8K</td> <td>2K</td> <td>0000001b</td> <td>Default</td> </tr> <tr> <td>16K</td> <td>4K</td> <td>0000010b</td> <td></td> </tr> <tr> <td>32K</td> <td>8K</td> <td>0000100b</td> <td></td> </tr> <tr> <td>64K</td> <td>16K</td> <td>0001000b</td> <td></td> </tr> <tr> <td>128K</td> <td>32K</td> <td>0010000b</td> <td></td> </tr> <tr> <td>256K</td> <td>64K</td> <td>0100000b</td> <td></td> </tr> <tr> <td>512K</td> <td>128K</td> <td>1000000b</td> <td></td> </tr> </tbody> </table>	Total Block Size	Individual Select Size	M6 - M0	Comments	8K	2K	0000001b	Default	16K	4K	0000010b		32K	8K	0000100b		64K	16K	0001000b		128K	32K	0010000b		256K	64K	0100000b		512K	128K	1000000b	
Total Block Size	Individual Select Size	M6 - M0	Comments																															
8K	2K	0000001b	Default																															
16K	4K	0000010b																																
32K	8K	0000100b																																
64K	16K	0001000b																																
128K	32K	0010000b																																
256K	64K	0100000b																																
512K	128K	1000000b																																
7	Pin Selector	This bit determines whether the PCS6_N - PCS5_N pins are configured as chip selects or as alternate outputs for A2 - A1. When this bit is set to 1, PCS6_N - PCS5_N are configured as peripheral chip select pins. When EX is set to 0, PCS5_N becomes address bit A1 and PCS6_N becomes address bit A2.																																
6	Memory / IO Space Selector	This bit determines whether the PCS pins are active during memory bus cycles or I/O bus cycles. When MS is set to 1, the PCS outputs are active for memory bus cycles. When MS is set to 0, the PCS outputs are active for I/O bus cycles.																																
2	Ready Mode	This bit applies only to the PCS6-PCS5 chip selects. If R2 is set to 0, external ready is required. If R2 is set to 1, external ready is ignored. In each case, the processor also uses the value of the R1-R0 bits to determine the number of wait states to insert.																																
1-0	Wait-State Value	These bits apply only to the PCS6-PCS5 chip selects. The value of R1-R0 determines the number of wait states inserted into an access to the PCS memory or I/O area. From zero to three wait states can be inserted (R1-R0 = 00b to 11b).																																

Figure 7.4 PCS and MCS Auxiliary Register

### 7.5 Peripheral Chip Select Register (PACS, Offset A4h)

Each peripheral chip select asserts over a 256-byte address range. The IMS16B provides six chip selects, PCS6\_N - PCS5\_N and PCS3\_N - PCS0\_N, for use within a user-locatable memory or I/O block. (PCS4\_N is not implemented on the IMS16B.) The base address of the memory block can be located anywhere within the 1Mbyte memory address space, exclusive of the areas associated with the UCS\_N, LCS\_N, and MCS chip selects, or they can be configured to access the 64Kbyte I/O space.

The Peripheral Chip Selects are programmed through two registers: the Peripheral Chip Select (PACS) register and the PCS and MCS Auxiliary (MPCS) register. The Peripheral Chip Select (PACS) register determines the base address, the ready condition, and the wait states for the PCS3\_N - PCS0\_N outputs. The PCS and MCS Auxiliary (MPCS) register contains bits that configure the PCS6\_N - PCS5\_N pins as either chip selects or address pins A1 and A2. When the PCS6\_N - PCS5\_N pins are chip selects, the MPCS register also determines whether PCS chip selects are active during memory or I/O bus cycles and specifies the ready and wait states for the PCS6\_N - PCS5\_N outputs.

The PCS pins are not active on reset. Both the PACS and MPCS registers must be accessed with a read or write to activate the PCS pins as chip selects. PCS6\_N - PCS5\_N can be configured and activated as address pins by writing only the MPCS register. No corresponding access to the PACS register is required in this case. PCS3\_N - PCS0\_N can be configured for zero wait states to 15 wait states. PCS6\_N - PCS5\_N can be configured for zero wait states to three wait states.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA19	BA18	BA17	BA16	BA15	BA14	BA13	BA12	BA11	1	1	1	1	R2	R1	R0

**Reset value : 0000\_0000\_0000\_0000b**

Bit	Name	Function																											
15 -7	Base Address	<p>The base address of the peripheral chip select block is defined by BA19 - BA11 of the PACS register. BA19 - BA11 correspond to bits 19 - 11 of the 20-bit programmable base address of the peripheral chip select block. Bit 6 of the PACS register corresponds to BA10. When the PCS chip selects are mapped to I/O space, BA19 - 16 must be programmed to 0000b because the I/O address bus is only 16-bits wide.</p> <table border="1"> <thead> <tr> <th>PCS_N Pins</th> <th colspan="2">Range</th> </tr> <tr> <td></td> <th>Low</th> <th>High</th> </tr> </thead> <tbody> <tr> <td>PCS0_N</td> <td>Base Address</td> <td>Base Address + 255(FFh)</td> </tr> <tr> <td>PCS1_N</td> <td>Base Address + 256</td> <td>Base Address + 511(1FFh)</td> </tr> <tr> <td>PCS2_N</td> <td>Base Address + 512</td> <td>Base Address + 767(2FFh)</td> </tr> <tr> <td>PCS3_N</td> <td>Base Address + 768</td> <td>Base Address + 1023(3FFh)</td> </tr> <tr> <td>Reserved</td> <td>N/A</td> <td>N/A</td> </tr> <tr> <td>PCS5_N</td> <td>Base Address + 1280</td> <td>Base Address + 1535(5FFh)</td> </tr> <tr> <td>PCS6_N</td> <td>Base Address + 1536</td> <td>Base Address + 1791(6FFh)</td> </tr> </tbody> </table>	PCS_N Pins	Range			Low	High	PCS0_N	Base Address	Base Address + 255(FFh)	PCS1_N	Base Address + 256	Base Address + 511(1FFh)	PCS2_N	Base Address + 512	Base Address + 767(2FFh)	PCS3_N	Base Address + 768	Base Address + 1023(3FFh)	Reserved	N/A	N/A	PCS5_N	Base Address + 1280	Base Address + 1535(5FFh)	PCS6_N	Base Address + 1536	Base Address + 1791(6FFh)
PCS_N Pins	Range																												
	Low	High																											
PCS0_N	Base Address	Base Address + 255(FFh)																											
PCS1_N	Base Address + 256	Base Address + 511(1FFh)																											
PCS2_N	Base Address + 512	Base Address + 767(2FFh)																											
PCS3_N	Base Address + 768	Base Address + 1023(3FFh)																											
Reserved	N/A	N/A																											
PCS5_N	Base Address + 1280	Base Address + 1535(5FFh)																											
PCS6_N	Base Address + 1536	Base Address + 1791(6FFh)																											
2	Ready Mode	<p>The R2 bit is used to configure the ready mode for the PCS3_N - PCS0_N chip selects. If R2 is set to 0, external ready is required. External ready is ignored when R2 is set to 1. In each case, the processor also uses the value of the R3 and R1 - R0 bits to determine the number of wait states to insert. The ready mode for PCS6_N - PCS5_N is configured through the MPCS register.</p>																											
1-0	Wait-State Value	<p>The value of R3 and R1 - R0 determines the number of wait states inserted into a PCS3_N - PCS0_N access. Up to 15 wait states can be inserted. See the discussion of</p>																											

		bit 3 (R3) for the wait-state encoding of R1 - R0.
--	--	--

**Figure 7.5 Peripheral Chip Select Register**

# 8

## Refresh Control Unit

The Refresh Control Unit (RCU) automatically generates refresh bus cycles. After a programmable period of time, the RCU generates a memory read request to the bus interface unit. The Refresh Control Unit operates off the processor internal clock. If the power-save mode is in effect, the Refresh Control Unit must be reprogrammed to reflect the new clock rate. If the HLDA pin is active when a refresh request is generated (indicating a bus hold condition), then the microcontroller deactivates the HLDA pin in order to perform a refresh cycle. The circuit external bus master must remove the HOLD signal for at least one clock to allow the refresh cycle to execute.

Offset	Register Mnemonic	Register Name
E0h	MDRAM	Memory Partition Register
E2h	CDRAM	Clock Pre-Scaler Register
E4h	EDRAM	Enable RCU Register

**Figure 8 Refresh Control Unit**

### 8.1 Memory Partition Register (MDRAM, Offset E0h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M6	M5	M4	M3	M2	M1	M0	0	0	0	0	0	0	0	0	0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
15 - 9	Refresh Base	Upper bits corresponding to address bits A19 - A13 of the 20-bit memory refresh address. These bits are cleared to 0 at reset.

**Figure 8.1 Memory Partition Register**

### 8.2 Clock Prescaler Register (CDRAM, Offset E2h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RC8	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
8 - 0	Refresh Counter Reload Value	Contains the value of the desired clock count interval between refresh cycles. The counter value should not be set to less than 18 (12h), otherwise there would never be sufficient bus cycles available for the processor to execute code.

**Figure 8.2 Clock Prescaler Register**

### 8.3 Enable RCU Register (EDRAM, Offset E4h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E	0	0	0	0	0	0	T8	T7	T6	T5	T4	T3	T2	T1	T0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
15	Enable RCU	Enables the refresh counter unit when set to 1. Clearing the E bit at any time clears the refresh counter and stops refresh requests, but it does not reset the refresh address. Set to 0 on reset.
8 - 0	Refresh Count	This read-only field contains the present value of the down counter which triggers refresh requests.

**Figure 8.3 Enable RCU Register**

## 9

## Interrupt Control Unit

---

The IMS16B microcontroller can receive interrupt requests from a variety of sources, both internal and external. The internal interrupt controller arranges these requests by priority and presents them one at a time to the CPU. There are six external interrupt sources on the IMS16B microcontroller: five maskable interrupt pins (INT4 - INT0) and the non-maskable interrupt (NMI) pin. There are seven internal interrupt sources that are not connected to external pins: three timers, two DMA channels, one asynchronous serial port and one I2C serial port. Interrupt Controller has the ability to resolve priority among simultaneous interrupt requests. **Note that IMS16B only supports Master Mode, and Fully Nested Mode.**

### 9.1 Definitions of Interrupt Terms

#### ■ Interrupt Vector Table

The interrupt vector table is a memory area of 1Kbyte beginning at address 00000h that holds up to 256 four-byte address pointers containing the address for the interrupt service routine for each possible interrupt type. For each interrupt, an 8-bit interrupt type identifies the appropriate interrupt vector table entry. Interrupts 00h to 1Fh are reserved. The processor calculates the index to the interrupt vector table by shifting the interrupt type left 2 bits (multiplying by 4).

#### ■ Maskable and Non-Maskable Interrupts

Interrupt types 08h through 1Fh are maskable. Of these, only 08h through 14h are actually in use. The maskable interrupts are enabled and disabled by the interrupt enable flag (IF) in the processor status flags, but the INT command can execute any interrupt regardless of the setting of IF. Interrupt types 00h through 07h and all software interrupts (the INT instruction) are non-maskable. The non-maskable interrupts are not affected by the setting of the IF flag.

The IMS16B provides two methods for masking and unmasking the maskable interrupt sources. Each interrupt source has an interrupt control register that contains a mask bit specific to that interrupt. In addition, the Interrupt Mask register is provided as a single source to access all of the mask bits. If the Interrupt Mask register is written while interrupts are enabled, it is possible that an interrupt could occur while the register is in an undefined state. This can cause interrupts to be accepted even though they were masked both before and after the write to the Interrupt Mask register. Therefore, the Interrupt Mask register should only be written when interrupts are disabled. Mask bits in the individual interrupt control registers can be written while interrupts are enabled, and there will be no erroneous interrupt operation.

#### ■ Interrupt Enable Flag (IF)

The interrupt enable flag (IF) is part of the processor status flags. If Interrupt Enable Flag is set to 1, maskable interrupts are enabled and can cause processor interrupt. (Individual maskable interrupts can still be disabled by means of the mask bit in each control register.) If Interrupt Enable Flag is set to 0, all maskable interrupts are disabled. The IF flag does not affect the NMI or software exception interrupts (interrupt types 00h to 07h), and it does not affect the execution of any interrupt through the INT instruction.

### ■ Interrupt Mask Bit

Each of the interrupt control registers for the maskable interrupts contains a mask bit (MSK). If MSK is set to 1 for a particular interrupt, that interrupt is disabled regardless of the IF setting.

### ■ Interrupt Priority

The column titled Overall Priority in Figure 9.1 shows the fundamental priority breakdown for the interrupts at power-on reset. The non-maskable interrupts 00h through 07h are always prioritized ahead of the maskable interrupts. The maskable interrupts can be reprioritized by reconfiguring the PR2 - PR0 bits in the interrupt control registers. The PR2 - PR0 bits in all the maskable interrupts are set to priority level 7 (lowest level) at power-on reset.

### ■ Software Interrupts

Software interrupts can be initiated by the INT instruction. Any of the 256 possible interrupts can be initiated by the INT instruction. INT 21h causes an interrupt to the vector located at 00084h in the interrupt vector table. INT FFh causes an interrupt to the vector located at 003FCh in the interrupt vector table. Software interrupts are not maskable and are not affected by the setting of the IF flag.

### ■ Software Exceptions

A software exception interrupt occurs when an instruction causes an interrupt due to some condition in the processor. Interrupt types 00h, 01h, 03h, 04h, 05h, 06h, and 07h are software exception interrupts. Software exceptions are not maskable and are not affected by the setting of the IF flag.

Interrupt Name	Interrupt Type	Vector Table Address	Default Priority	Related Instructions	Application Notes
Divide Error Exception	00h	00h	1	DIV, IDIV	1
Trace Interrupt	01h	04h	1A	All	2
Non-Maskable Interrupt(NMI)	02h	08h	1	All	
Breakpoint Interrupt	03h	0Ch	1	INT	1
INT0 Detected Overflow Exception	04h	10h	1	INT0	1
Array Bounds Exception	05h	14h	1	BOUND	1
Unused Opcode Exception	06h	18h	1	Undefined Opcodes	1
ESC Opcode Exception	07h	1Ch	1	ESC	1
Timer 0 Interrupt	08h	20h	2A		3
Timer 1 Interrupt	12h	48h	2B		3
Timer 2 Interrupt	13h	4Ch	2C		3
DMA 0 Interrupt	0Ah	28h	4		
DMA 1 Interrupt	0Bh	2Ch	5		
INT0 Interrupt	0Ch	30h	6		
INT1 Interrupt	0Dh	34h	7		
INT2 Interrupt	0Eh	38h	8		
INT3 Interrupt	0Fh	3Ch	9		
INT4 Interrupt	10h	40h	10		
Watchdog Timer Interrupt	11h	44h	14		
UART Serial Port Interrupt	14h	50h	15		
I2C Serial Port Interrupt	17h	5Ch	13		

**Figure 9.1 IMS16B Interrupt Types**

#### NOTES:

Default priorities for interrupt sources are used only if the user does not program each source to a unique priority level.

1. Generated as a result of an instruction execution.

2. Performed in the same manner as the 80C186.

3. All three timers make up a single interrupt request from the interrupt controller. Because of this they share the same priority level. However, each timer has a defined priority with respect to each other. Priority level 2A is the highest, then 2B, followed by 2C.

## 9.2 Interrupt Conditions and Sequence

### ■ Non-Maskable Interrupts

Non-maskable interrupts—the trace interrupt, the NMI interrupt, and software interrupts [both user-defined (INT) and software exceptions]—are serviced regardless of the setting of the interrupt enable flag (IF) in the processor status flags.

### ■ Maskable Hardware Interrupts

In order for maskable hardware interrupt requests to be serviced, the IF flag must be set by the STI instruction, and the mask bit associated with each interrupt must be reset.

### ■ The Interrupt Request

When an interrupt is requested, the internal interrupt controller verifies that the interrupt is enabled and that there are no higher priority interrupt requests being serviced or pending. If the interrupt request is granted, the interrupt controller uses the interrupt type to access a vector from the interrupt vector table. Each interrupt type has a four-byte vector available in the interrupt vector table. The interrupt vector table is located in the 1024 bytes from 00000h to 003FFh. Each four-byte vector consists of a 16-bit offset (IP) value and a 16-bit segment (CS) value. The 8-bit interrupt type is shifted left 2 bit positions (multiplied by 4) to generate the index into the interrupt vector table.

### ■ Interrupt Servicing

A valid interrupt transfers execution to a new program location based on the vector in the interrupt vector table. The next instruction address (CS:IP) and the processor status flags are pushed onto the stack. The interrupt enable flag (IF) is cleared after the processor status flags are pushed on the stack, disabling maskable interrupts during the interrupt service routine (ISR). The segment:offset values from the interrupt vector table are loaded into the code segment (CS) and the instruction pointer (IP), and execution of the ISR begins.

### ■ Returning from the Interrupt

The interrupt return (IRET) instruction pops the processor status flags and the return address off the stack. Program execution resumes at the point where the interrupt occurred. The interrupt enable flag (IF) is restored by the IRET instruction along with the rest of the processor status flags. If the IF flag was set before the interrupt was serviced, interrupts are re-enabled when the IRET is executed. If there are valid interrupts pending when the IRET is executed, the instruction at the return address is not executed. Instead, the new interrupt is serviced immediately. If an ISR intends to permanently modify the value of any of the saved flags, it must modify the copy of the Processor Status Flags register that was pushed onto the stack.

## 9.3 Interrupt Priority

Figure 9.1 shows the predefined types and overall priority structure for the IMS16B. Non-maskable interrupts (interrupt types 0-7) are always higher priority than maskable interrupts. Maskable interrupts have a programmable priority that can override the default priorities relative to one another.

### ■ Non-Maskable Interrupts and Software Interrupt Priority

The non-maskable interrupts from 00h to 07h and software interrupts (INT instruction) always take

priority over the maskable hardware interrupts. Within the non-maskable and software interrupts, the trace interrupt has the highest priority, followed by the NMI interrupt, followed by the remaining non-maskable and software interrupts. After the trace interrupt and the NMI interrupt, the remaining software exceptions are mutually exclusive and can only occur one at a time, so there is no further priority breakdown.

#### ■ Maskable Hardware Interrupt Priority

Beginning with interrupt type 8 (the Timer 0 interrupt), the maskable hardware interrupts have both an overall priority (see Figure 9.1) and a programmable priority. The programmable priority is the primary priority for maskable hardware interrupts. The overall priority is the secondary priority for maskable hardware interrupts. Since all maskable interrupts are set to a programmable priority of seven on reset, the overall priority of the interrupts determines the priority in which each interrupt is granted by the interrupt controller until programmable priorities are changed by reconfiguring the control registers. The overall priority levels shown in Figure 9.1 are not the same as the programmable priority level that is associated with each maskable hardware interrupt. Each of the maskable hardware interrupts has a programmable priority from zero to seven, with zero being the highest priority.

For example, if the INT4 - INT0 interrupts are all changed to programmable priority six and no other programmable priorities are changed from the reset value of seven, then the INT4 - INT0 interrupts take precedence over all other maskable interrupts. (Within INT4 - INT0, INT0 takes precedence over INT1, and INT1 takes precedence over INT2, etc., because of the underlying hierarchy of the overall priority.)

## 9.4 Software Exceptions, Traps and NMI

User programming cannot mask the following predefined interrupts.

#### ■ Divide Error Exception (Interrupt Type 00h)

Generated when a DIV or IDIV instruction quotient cannot be expressed in the number of destination bits.

#### ■ Trace Interrupt (Interrupt Type 01h)

If the trace flag (TF) in the Processor Status flags register is set, the trace interrupt is generated after most instructions. This interrupt allows programs to execute in single-step mode. The interrupt is not generated after prefix instructions like REP, instructions that modify segment registers like POP DS, or the WAIT instruction. Taking the trace interrupt clears the TF bit after the processor status flags are pushed onto the stack. The IRET instruction at the end of the single step interrupts service routine restores the processor status flags (and the TF bit) and transfers control to the next instruction to be traced. Pushing the processor status flags onto the stack, setting the TF flag on the stack, and then popping the flags initiate trace mode.

#### ■ Non-Maskable Interrupt—NMI (Interrupt Type 02h)

The NMI pin provides an external interrupt source that is serviced regardless of the state of the IF (interrupt enable flag) bit. No external interrupt acknowledge sequence is performed for an NMI interrupt. A typical use of NMI is to activate a power failure routine.

#### ■ Breakpoint Interrupt (Interrupt Type 03h)

An interrupt caused by the 1-byte version of the INT instruction (INT3).

**■ INTO Detected Overflow Exception (Interrupt Type 04h)**

Generated by an INTO instruction if the OF bit is set in the Processor Status Flags (FLAGS) register.

**■ Array BOUNDS Exception (Interrupt Type 05h)**

Generated by a BOUND instruction if the array index is outside the array bounds. The array bounds are located in memory at a location indicated by one of the instruction operands. The other operand indicates the value of the index to be checked.

**■ Unused Opcode Exception (Interrupt Type 06h)**

Generated if execution is attempted on undefined opcodes.

**■ ESC Opcode Exception (Interrupt Type 07h)**

Generated if execution of ESC opcodes (D8h-DFh) is attempted. The microcontrollers do not check the escape opcode trap bit. The return address of this exception points to the ESC instruction that caused the exception. If a segment override prefix preceded the ESC instruction, the return address points to the segment override prefix. All numeric coprocessor opcodes cause a trap. The IMS16B does not support the numeric coprocessor interface.

**■ Interrupt Acknowledge**

Interrupts can be acknowledged in two different ways : the internal interrupt controller can provide the interrupt type or an external interrupt controller can provide the interrupt type. The processor requires the interrupt type as an index into the interrupt vector table. When the internal interrupt controller is supplying the interrupt type, no bus cycles are generated. The only external indication that an interrupt is being serviced is the processor reading the interrupt vector table. When an external interrupt controller is supplying the interrupt type, the processor generates two interrupt acknowledge bus cycles. The interrupt type is written to the AD7-AD0 lines by the external interrupt controller during the second bus cycle.

**■ Interrupt Controller Reset Conditions**

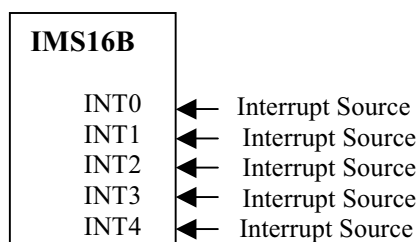
On reset, the interrupt controller performs the following nine actions:

1. All special fully nested mode (SFNM) bits are reset, implying fully nested mode.
2. All priority (PR) bits in the various control registers are set to 1. This places all sources at the lowest priority (level 7).
3. All level-triggered mode (LTM) bits are reset to 0, resulting in edge-triggered mode.
4. All interrupt in-service bits are reset to 0.
5. All interrupt request bits are reset to 0.
6. All mask (MSK) bits are set to 1. All interrupts are masked.
7. All cascade (C) bits are reset to 0 (non-cascade).
8. The interrupt priority mask is set to 7, allowing interrupts of all priorities.
9. The interrupt controller is initialized to master mode.

## 9.5 Fully Nested Mode

In fully nested mode, five pins are used as direct interrupt requests as in Figure 9.5. The interrupt types for these five inputs are generated internally. An in-service bit is provided for every interrupt source. If a lower-priority device requests an interrupt while the in-service bit (IS) is set for a higher priority interrupt, the interrupt controller does not generate any interrupt. In addition, if another interrupts request occurs from the same interrupt source while the in-service bit is set; the interrupt controller

generates no interrupt. This allows interrupt service routines operating with interrupts enabled to be suspended only by interrupts of equal or higher priority than the in-service interrupt. When an interrupt service routine is completed, the proper IS bit must be reset by writing the interrupt type to the EOI register. This is required to allow subsequent interrupts from this interrupt source and to allow servicing of lower-priority interrupts. A write to the EOI register should be executed at the end of the interrupt service routine just before the return from interrupt instruction.



**Figure 9.5 Fully Nested Mode Interrupt Controller Connections**

## 9.6 Operation in a Polled Environment

To allow reading of the Poll register information without setting the indicated in-service bit, the IMS16B microcontroller provides a Poll Status register in addition to the Poll register. Poll register information is duplicated in the Poll Status register, but the Poll Status register can be read without setting the associated in-service bit. These registers are located in two adjacent memory locations in the peripheral control block. The interrupt controller can be used in polled mode if interrupts are not desired. When polling, interrupts are disabled and software polls the interrupt controller as required. The interrupt controller is polled by reading the Poll Status register. Bit 15 in the Poll Status register indicates to the processor that an interrupt of high enough priority is requesting service. Bits 4 - 0 indicate to the processor the interrupt type of the highest priority source requesting service. After determining that an interrupt is pending, software reads the Poll register (rather than the Poll Status register), which causes the in-service bit of the highest priority source to be set.

## 9.7 End-of-Interrupt Write to the EOI Register

A program must write to the EOI register to reset the in-service (IS) bit when an interrupt service routine is completed. There are two types of writes to the EOI register: specific EOI and non-specific EOI. Non-specific EOI does not specify which IS bit is to be reset. Instead, the interrupt controller automatically resets the IS bit of the highest priority source with an active service routine. Specific EOI requires the program to send the interrupt type to the interrupt controller to indicate the source IS bit that is to be reset. Specific reset is applicable when interrupt nesting is possible or when the highest priority IS bit that was set does not belong to the service routine in progress.

## 9.8 Programmable Priority

Each interrupt source is equipped with a programmable priority level within its control register. These bits can be programmed to one of eight different priorities (000b - 111b). Priority level 000b has the highest priority and 111b has the lowest priority. These are the bits that determine which interrupt

source will have priority over another.

## 9.9 Trigger Mode

The five external interrupts (INT4 - INT0) can have their inputs configured for either edge-triggered or level-triggered mode. In either mode all interrupt inputs are active high. When the inputs are configured for level-triggered mode (LTM=1) an interrupt request will occur when ever the input is pulled high. The input must remain high long enough for the processor to recognize it. If the input continues to stay high it will cause another interrupt only after the first interrupt has been processed. When the inputs are configured for edge-triggered mode (LTM=0) an interrupt request will occur when the input makes a low to high transition.

## 9.10 Interrupt Control Registers

The interrupt control registers for master mode are shown in Figure 9.10

Offset	Register Mnemonic	Register Name	Associated Pins	Comments
22h	EOI	EOI Register		Write-only Register
24h	POLL	Poll Register		Read-only Register
26h	POLLST	Poll Status Register		Read-only Register
28h	IMASK	Interrupt Mask Register	INT4 - INT0 DRQ1 - DRQ0	
2Ah	PRIMSK	Priority Mask Register		
2Ch	INSERV	In-Service Register	INT4 - INT0 DRQ1 - DRQ0	
2Eh	REQST	Interrupt Request Register	INT4 - INT0 DRQ1 - DRQ0	Read-only Register
30h	INTSTS	Interrupt Status Register		
32h	TCUCON	Timer Interrupt Control Register	TMRIN1 TMRIN0 TMROUT1 TMROUT0	
34h	DMA0CON	DMA0 Interrupt Control Register	DRQ0	
36h	DMA1CON	DMA1 Interrupt Control Register	DRQ1	
38h	I0CON	INT0 Control Register	INT0	
3Ah	I1CON	INT1 Control Register	INT1	
3Ch	I2CON	INT2 Control Register	INT2	
3Eh	I3CON	INT3 Control Register	INT3	
40h	I4CON	INT4 Control Register	INT4	
42h	WDCON	Watchdog Timer Interrupt Control Register		
44h	SPICON	UART Interrupt Control Register	TXD, RXD	
4Ah	I2CCON	I2C Interrupt Control Register	SCL, SDA	

**Figure 9.10 Interrupt Control Registers**

**9.11 INT0 and INT1 Control Registers  
(I0CON, Offset 38h, I1CON, Offset 3Ah)  
(Master Mode)**

The INT0 interrupt is assigned to interrupt type 0Ch. The INT1 interrupt is assigned to interrupt type 0Dh. When cascade mode is enabled for INT0 by setting the C bit of I0CON to 1, the INT2 pin becomes INTA0, the interrupt acknowledge for INT0. When cascade mode is enabled for INT1 by setting the C bit of I1CON to 1, the INT3 pin becomes INTA1, the interrupt acknowledge for INT1.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	SFNM	C	LTM	MSK	PR2	PR1	PR0

Reset value : 0000\_0000\_0000\_1111b

Bit	Name	Function																		
6	Special Fully Nested Mode	When set to 1, enables special fully nested mode.																		
5	Cascade Mode	When set to 1, this bit enables cascade mode.																		
4	Level Triggered Mode	This bit determines whether the microcontroller interprets an INT0 or INT1 interrupt request as edge- or level-sensitive. A 1 in this bit configures INT0 or INT1 as an active High, level-sensitive interrupt. A 0 in this bit configures INT0 or INT1 as a Low-to-High, edge-triggered interrupt. In either case, INT0 or INT1 must remain High until they are acknowledged.																		
3	Mask	This bit determines whether the INT0 or INT1 signal can cause an interrupt. A 1 in this bit masks this interrupt source, preventing INT0 or INT1 from causing an interrupt. A 0 in this bit enables INT0 or INT1 interrupts.																		
2-0	Priority Level	This field determines the priority of INT0 or INT1 relative to the other interrupt signals. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

**Figure 9.11 INT0 and INT1 Control Registers**

### 9.12 INT2 and INT3 Control Registers (I2CON, Offset 3Ch, I3CON, Offset 3Eh) (Master Mode)

The INT2 interrupt is assigned to interrupt type 0Eh. The INT3 interrupt is assigned to interrupt type 0Fh. The INT2 and INT3 pins can be configured as interrupt acknowledge pins INTA0 and INTA1 when cascade mode is implemented.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	LTM	MSK	PR2	PR1	PR0

Reset value : 0000\_0000\_0000\_1111b

Bit	Name	Function																		
4	Level Triggered Mode	This bit determines whether the microcontroller interprets an INT2 or INT3 interrupt request as edge- or level-sensitive. A 1 in this bit configures INT2 or INT3 as an active High, level-sensitive interrupt. A 0 in this bit configures INT2 or INT3 as a Low-to-High, edge-triggered interrupt. In either case, INT2 or INT3 must remain High until they are acknowledged.																		
3	Mask	This bit determines whether the INT2 or INT3 signal can cause an interrupt. A 1 in this bit masks this interrupt source, preventing INT2 or INT3 from causing an interrupt. A 0 in this bit enables INT2 or INT3 interrupts.																		
2-0	Priority Level	This field determines the priority of INT2 or INT3 relative to the other interrupt signals. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

Figure 9.12 INT2 and INT3 Control Registers

### 9.13 INT4 Control Register (I4CON, Offset 40h) (Master Mode)

The IMS16B microcontroller provides INT4, an additional external interrupt pin. This input behaves like INT3 - INT0 on the 80C186/188 microcontroller with the exception that INT4 is only intended for use as a nested-mode interrupt source.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	LTM	MSK	PR2	PR1	PR0

Reset value : 0000\_0000\_0000\_1111b

Bit	Name	Function																		
4	Level Triggered Mode	This bit determines whether the microcontroller interprets an INT4 interrupt request as edge- or level-sensitive. A 1 in this bit configures INT4 as an active High, level-sensitive interrupt. A 0 in this bit configures INT4 as a Low-to-High, edge-triggered interrupt. In either case, INT4 must remain High until they are acknowledged.																		
3	Mask	This bit determines whether the INT4 signal can cause an interrupt. A 1 in this bit masks this interrupt source, preventing INT4 from causing an interrupt. A 0 in this bit enables INT4 interrupts.																		
2-0	Priority Level	This field determines the priority of INT4 relative to the other interrupt signals. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

Figure 9.13 INT4 Control Registers

**9.14 Timer and DMA Interrupt Control Registers**  
**(TCUCON, Offset 32h, DMA0CON, Offset 34h, DMA1CON, Offset 36h)**  
**(Master Mode)**

The three timer interrupts are assigned to interrupt type 08h, 12h, and 13h. All three timer interrupts are configured through TCUCON, offset 32h. The DMA0 interrupt is assigned to interrupt type 0Ah. The DMA1 interrupt is assigned to interrupt type 0Bh.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	MSK	PR2	PR1	PR0

Reset value : 0000\_0000\_0000\_1111b

Bit	Name	Function																		
3	Mask	This bit determines whether the corresponding signal can generate an interrupt. A 1 masks this interrupt source. A 0 enables the corresponding interrupt.																		
2-0	Priority Level	Sets the priority level for its corresponding source. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

**Figure 9.14 Timer and DMA Interrupt Control Registers**

### 9.15 Watchdog Timer Interrupt Control Register (WDCON, Offset 42h) (Master Mode)

The IMS16B microcontroller provides an additional on-chip interrupt source, the watchdog timer. This timer is constructed from existing 80C186 microcontroller pins. It is implemented by connecting the TMROUT1 output to an additional internal interrupt to create the watchdog timer interrupt. This interrupt is assigned to interrupt type 11h.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	MSK	PR2	PR1	PR0

Reset value : 0000\_0000\_0000\_1111b

Bit	Name	Function																		
3	Mask	This bit determines whether the watchdog timer can cause an interrupt. A 1 in this bit masks this interrupt source, preventing the watchdog timer from causing an interrupt. A 0 in this bit enables watchdog timer interrupts.																		
2-0	Priority Level	This field determines the priority of the watchdog timer relative to the other interrupt signals. <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

Figure 9.15 Watchdog Timer Interrupt Control Register

### 9.16 UART Serial Interrupt Control Register (SPICON, Offset 44h) (Master Mode)

The Serial Port Interrupt Control register controls the operation of the asynchronous serial port interrupt source (SPI, bit 10 in the Interrupt Request register). This interrupt is assigned to interrupt type 14h.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	MSK	PR2	PR1	PR0

Reset value : 0000\_0000\_0000\_1111b

Bit	Name	Function																		
3	Mask	This bit determines whether the serial port can cause an interrupt. A 1 in this bit masks this interrupt source, preventing the serial port from causing an interrupt. A 0 in this bit enables serial port interrupts.																		
2-0	Priority Level	This field determines the priority of the serial port relative to the other interrupt signals. <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

Figure 9.16 UART Serial Port Interrupt Control Register

### 9.17 Interrupt Status Register (INTSTS, Offset 30h) (Master Mode)

The Interrupt Status (INTSTS) register indicates the interrupt request status of the three Timers.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHLT	0	0	0	0	0	0	0	0	0	0	0	0	TMR2	TMR1	TMR0

**Reset value : 0000\_0000\_0000\_0000b**

Bit	Name	Function
15	DMA Halt	When set to 1, halts any DMA activity. This pin is automatically set to 1 when non-maskable interrupts occur and is reset when an IRET instruction is executed. Time-critical software, such as interrupt handlers, can modify this bit directly to inhibit DMA transfers. Because of the function of this register as an interrupt request register for the timers, the DHLT bit should not be modified by software when timer interrupts are enabled.
2-0	Timer Interrupt Request	When set to 1, these bits indicate that the corresponding timer has an interrupt request pending.

**Figure 9.17 Interrupt Status Register**

## 9.18 Interrupt Request Register (REQST, Offset 2Eh) (Master Mode)

The hardware interrupt sources have interrupt request bits inside the interrupt controller. A read from this register yields the status of these bits. The Interrupt Request register is a read-only register. The IMS16B microcontroller defines four new bits to report the state of INT4, the Watchdog Timer, the asynchronous serial port and I2C serial port.

For internal interrupts (SPI, WD, D1, D0, TMR and I2C), the corresponding bit is set to 1 when the device requests an interrupt. The bit is reset during the internally generated interrupt acknowledge.

For INT4 - INT0 external interrupts; the corresponding bit (I4 - I0) reflects the current value of the external signal. The device must hold this signal High until the interrupt is serviced. Generally the interrupt service routine signals the external device to remove the interrupt request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	I2C	0	0	UART	WDT	I4	I3	I2	I1	I0	D1	D0	0	TMR

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
13	I2C Interrupt Request	When this bit is set to 1, the I2C Serial port has an interrupt pending.
10	UART Serial Port Interrupt Request	This bit indicates the interrupt state of the serial port. It is set when the serial port generates an interrupt request and cleared when the interrupt acknowledge cycle occurs
9	Watchdog Timer Interrupt Request	When this bit is set to 1, the Watchdog Timer has an interrupt pending.
8-4	Interrupt Request	When set to 1, the corresponding INT pin has an interrupt pending (i.e., when INT0 is pending, I0 is set). These bits reflect the status of the external pin.
3-2	DMA Channel Interrupt Request	When set to 1, the corresponding DMA channel has an interrupt pending.
0	Timer Interrupt Request	This bit indicates the state of the timer interrupts. This bit is the logical OR of the timer interrupts requests. When set to a 1, this bit indicates that the timer control unit has an interrupt pending.

**Figure 9.18 Interrupt Request Register**

### 9.19 In-Service Register (INSERV, Offset 2Ch) (Master Mode)

The IMS16B microcontroller defines three new bits to report the in-service state of INT4, the Virtual Watchdog Timer, the asynchronous serial port and I2C serial port.

The interrupt controller sets the bits in the INSERV register when the interrupt is taken. Writing the corresponding interrupt type to the End-of-Interrupt (EOI) register clears each bit in the register.

When an in-service bit is set, the microcontroller will not generate an interrupt request for the associated source, preventing an interrupt from interrupting itself if interrupts are enabled in the ISR. In-Service Register is set to 0000h on reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	I2C	0	0	UART	WDT	I4	I3	I2	I1	I0	D1	D0	0	TMR

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
13	I2C Interrupt In-Service	This bit indicates the in-service state of the I2C serial port.
10	UART Serial Port Interrupt In-Service	This bit indicates the in-service state of the UART asynchronous serial port.
9	Watchdog Timer Interrupt In-Service	This bit indicates the in-service state of the Watchdog Timer.
8-4	Interrupt In-Service	These bits indicate the in-service state of the corresponding INT pin.
3-2	DMA Channel Interrupt In-Service	These bits indicate the in-service state of the corresponding DMA channel.
0	Timer Interrupt In-Service	This bit indicates the state of the in-service timer interrupts. This bit is the logical OR of all the timer interrupt status bits. When set to a 1, this bit indicates that the corresponding timer interrupt status bit is in-service.

**Figure 9.19 In-Service Register**

### 9.20 Priority Mask Register (PRIMSK, Offset 2Ah) (Master Mode)

The Priority Mask (PRIMSK) register provides the value that determines the minimum priority level at which maskable interrupts can generate an interrupt. The value of PRIMSK at reset is 0007h.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	PRM2	PRM1	PRM0

Reset value : 0000\_0000\_0000\_0111b

Bit	Name	Function																		
2-0	Priority Field Mask	<p>This field determines the minimum priority that is required in order for a maskable interrupt source to generate an interrupt. Maskable interrupts with programmable priority values that are numerically higher than this field are masked. The possible values are zero (000b) to seven (111b).</p> <table border="1"> <thead> <tr> <th>Priority</th> <th>PR2 – PR0</th> </tr> </thead> <tbody> <tr> <td>(HIGH) 0</td> <td>000b</td> </tr> <tr> <td>1</td> <td>001b</td> </tr> <tr> <td>2</td> <td>010b</td> </tr> <tr> <td>3</td> <td>011b</td> </tr> <tr> <td>4</td> <td>100b</td> </tr> <tr> <td>5</td> <td>101b</td> </tr> <tr> <td>6</td> <td>110b</td> </tr> <tr> <td>(LOW) 7</td> <td>111b</td> </tr> </tbody> </table>	Priority	PR2 – PR0	(HIGH) 0	000b	1	001b	2	010b	3	011b	4	100b	5	101b	6	110b	(LOW) 7	111b
Priority	PR2 – PR0																			
(HIGH) 0	000b																			
1	001b																			
2	010b																			
3	011b																			
4	100b																			
5	101b																			
6	110b																			
(LOW) 7	111b																			

Figure 9.20 Priority Mask Register

## 9.21 Interrupt Mask Register (IMASK, Offset 28h) (Master Mode)

The IMS16B microcontroller define three new bits to report the mask state of the INT4 Control, Watchdog Timer Interrupt Control, Serial Port Interrupt, and I2C Interrupt Control registers. The Interrupt Mask (IMASK) register is a read/write register. Programming a bit in the IMASK register has the effect of programming the MSK bit in the associated control register.

Do not write to the interrupt mask register while interrupts are enabled. To modify mask bits while interrupts are enabled, use the individual interrupt control registers. The IMASK register is set to 3FFDh on reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	I2C	1	1	UART	WDT	I4	I3	I2	I1	I0	D1	D0	0	TMR

Reset value : 0011\_1111\_1111\_1101b

Bit	Name	Function
13	I2C Interrupt Mask	When set to 1, this bit indicates that the I2C serial port interrupt is masked.
10	UART Serial Port Interrupt Mask	When set to 1, this bit indicates that the asynchronous serial port interrupt is masked.
9	Watchdog Timer Interrupt Mask	When set to 1, this bit indicates that the Watchdog Timer interrupt is masked.
8-4	Interrupt Mask	When set to 1, an I4 - I0 bit indicates that the corresponding interrupt is masked.
3-2	DMA Channel Interrupt Mask	When set to 1, a D1 - D0 bit indicates that the corresponding DMA channel interrupt is masked.
0	Timer Interrupt Mask	When set to 1, this bit indicates that interrupt requests from the timer control unit are masked.

**Figure 9.21 Interrupt Mask Register**

### 9.22 Poll Status Register (POLLST, Offset 26h) (Master Mode)

The Poll Status (POLLST) register mirrors the current state of the Poll register. The POLLST register can be read without affecting the current interrupt request. But when the Poll register is read, the current interrupt is acknowledged and the next interrupt takes its place in the Poll register.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IREQ	0	0	0	0	0	0	0	0	0	0	S4	S3	S2	S1	S0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function																																												
15	Interrupt Request	Set to 1 if an interrupt is pending. When this bit is set to 1, the S4 - S0 field contains valid data.																																												
4-0	Poll Status	Indicates the interrupt type of the highest priority pending interrupt. These bits are only valid if IREQ = 1. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Interrupt Type S[4:0]</th> <th>Interrupt Name</th> </tr> </thead> <tbody> <tr><td>00h</td><td>Divide Error Exception</td></tr> <tr><td>01h</td><td>Trace Interrupt</td></tr> <tr><td>02h</td><td>Non-Maskable Interrupt(NMI)</td></tr> <tr><td>03h</td><td>Breakpoint Interrupt</td></tr> <tr><td>04h</td><td>INT0 Detected Overflow Exception</td></tr> <tr><td>05h</td><td>Array Bounds Exception</td></tr> <tr><td>06h</td><td>Unused Opcode Exception</td></tr> <tr><td>07h</td><td>ESC Opcode Exception</td></tr> <tr><td>08h</td><td>Timer 0 Interrupt</td></tr> <tr><td>12h</td><td>Timer 1 Interrupt</td></tr> <tr><td>13h</td><td>Timer 2 Interrupt</td></tr> <tr><td>0Ah</td><td>DMA 0 Interrupt</td></tr> <tr><td>0Bh</td><td>DMA 1 Interrupt</td></tr> <tr><td>0Ch</td><td>INT0 Interrupt</td></tr> <tr><td>0Dh</td><td>INT1 Interrupt</td></tr> <tr><td>0Eh</td><td>INT2 Interrupt</td></tr> <tr><td>0Fh</td><td>INT3 Interrupt</td></tr> <tr><td>10h</td><td>INT4 Interrupt</td></tr> <tr><td>11h</td><td>Watchdog Timer Interrupt</td></tr> <tr><td>14h</td><td>UART Serial Port Interrupt</td></tr> <tr><td>17h</td><td>I2C Serial Port Interrupt</td></tr> </tbody> </table>	Interrupt Type S[4:0]	Interrupt Name	00h	Divide Error Exception	01h	Trace Interrupt	02h	Non-Maskable Interrupt(NMI)	03h	Breakpoint Interrupt	04h	INT0 Detected Overflow Exception	05h	Array Bounds Exception	06h	Unused Opcode Exception	07h	ESC Opcode Exception	08h	Timer 0 Interrupt	12h	Timer 1 Interrupt	13h	Timer 2 Interrupt	0Ah	DMA 0 Interrupt	0Bh	DMA 1 Interrupt	0Ch	INT0 Interrupt	0Dh	INT1 Interrupt	0Eh	INT2 Interrupt	0Fh	INT3 Interrupt	10h	INT4 Interrupt	11h	Watchdog Timer Interrupt	14h	UART Serial Port Interrupt	17h	I2C Serial Port Interrupt
Interrupt Type S[4:0]	Interrupt Name																																													
00h	Divide Error Exception																																													
01h	Trace Interrupt																																													
02h	Non-Maskable Interrupt(NMI)																																													
03h	Breakpoint Interrupt																																													
04h	INT0 Detected Overflow Exception																																													
05h	Array Bounds Exception																																													
06h	Unused Opcode Exception																																													
07h	ESC Opcode Exception																																													
08h	Timer 0 Interrupt																																													
12h	Timer 1 Interrupt																																													
13h	Timer 2 Interrupt																																													
0Ah	DMA 0 Interrupt																																													
0Bh	DMA 1 Interrupt																																													
0Ch	INT0 Interrupt																																													
0Dh	INT1 Interrupt																																													
0Eh	INT2 Interrupt																																													
0Fh	INT3 Interrupt																																													
10h	INT4 Interrupt																																													
11h	Watchdog Timer Interrupt																																													
14h	UART Serial Port Interrupt																																													
17h	I2C Serial Port Interrupt																																													

Figure 9.22 Poll Status Register

### 9.23 Poll Register (POLL, Offset 24h) (Master Mode)

When the Poll register is read, the current interrupt is acknowledged and the next interrupt takes its place in the Poll register. The Poll Status register mirrors the current state of the Poll register, but the Poll Status register can be read without affecting the current interrupt request.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IREQ	0	0	0	0	0	0	0	0	0	0	S4	S3	S2	S1	S0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function																																												
15	Interrupt Request	Set to 1 if an interrupt is pending. When this bit is set to 1, the S4 - S0 field contains valid data.																																												
4-0	Poll Status	Indicates the interrupt type of the highest priority pending interrupt. Reading the Poll register acknowledges the highest priority pending interrupts and enables the next interrupt to advance into the register. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Interrupt Type S[4:0]</th> <th>Interrupt Name</th> </tr> </thead> <tbody> <tr><td>00h</td><td>Divide Error Exception</td></tr> <tr><td>01h</td><td>Trace Interrupt</td></tr> <tr><td>02h</td><td>Non-Maskable Interrupt (NMI)</td></tr> <tr><td>03h</td><td>Breakpoint Interrupt</td></tr> <tr><td>04h</td><td>INT0 Detected Overflow Exception</td></tr> <tr><td>05h</td><td>Array Bounds Exception</td></tr> <tr><td>06h</td><td>Unused Opcode Exception</td></tr> <tr><td>07h</td><td>ESC Opcode Exception</td></tr> <tr><td>08h</td><td>Timer 0 Interrupt</td></tr> <tr><td>12h</td><td>Timer 1 Interrupt</td></tr> <tr><td>13h</td><td>Timer 2 Interrupt</td></tr> <tr><td>0Ah</td><td>DMA 0 Interrupt</td></tr> <tr><td>0Bh</td><td>DMA 1 Interrupt</td></tr> <tr><td>0Ch</td><td>INT0 Interrupt</td></tr> <tr><td>0Dh</td><td>INT1 Interrupt</td></tr> <tr><td>0Eh</td><td>INT2 Interrupt</td></tr> <tr><td>0Fh</td><td>INT3 Interrupt</td></tr> <tr><td>10h</td><td>INT4 Interrupt</td></tr> <tr><td>11h</td><td>Watchdog Timer Interrupt</td></tr> <tr><td>14h</td><td>UART Serial Port Interrupt</td></tr> <tr><td>17h</td><td>I2C Serial Port Interrupt</td></tr> </tbody> </table>	Interrupt Type S[4:0]	Interrupt Name	00h	Divide Error Exception	01h	Trace Interrupt	02h	Non-Maskable Interrupt (NMI)	03h	Breakpoint Interrupt	04h	INT0 Detected Overflow Exception	05h	Array Bounds Exception	06h	Unused Opcode Exception	07h	ESC Opcode Exception	08h	Timer 0 Interrupt	12h	Timer 1 Interrupt	13h	Timer 2 Interrupt	0Ah	DMA 0 Interrupt	0Bh	DMA 1 Interrupt	0Ch	INT0 Interrupt	0Dh	INT1 Interrupt	0Eh	INT2 Interrupt	0Fh	INT3 Interrupt	10h	INT4 Interrupt	11h	Watchdog Timer Interrupt	14h	UART Serial Port Interrupt	17h	I2C Serial Port Interrupt
Interrupt Type S[4:0]	Interrupt Name																																													
00h	Divide Error Exception																																													
01h	Trace Interrupt																																													
02h	Non-Maskable Interrupt (NMI)																																													
03h	Breakpoint Interrupt																																													
04h	INT0 Detected Overflow Exception																																													
05h	Array Bounds Exception																																													
06h	Unused Opcode Exception																																													
07h	ESC Opcode Exception																																													
08h	Timer 0 Interrupt																																													
12h	Timer 1 Interrupt																																													
13h	Timer 2 Interrupt																																													
0Ah	DMA 0 Interrupt																																													
0Bh	DMA 1 Interrupt																																													
0Ch	INT0 Interrupt																																													
0Dh	INT1 Interrupt																																													
0Eh	INT2 Interrupt																																													
0Fh	INT3 Interrupt																																													
10h	INT4 Interrupt																																													
11h	Watchdog Timer Interrupt																																													
14h	UART Serial Port Interrupt																																													
17h	I2C Serial Port Interrupt																																													

Figure 9.23 Poll Register

### 9.24 End-of-Interrupt Register (EOI, Offset 22h) (Master Mode)

The End-of-Interrupt (EOI) register is a write-only register. Writing to the EOI register resets the in-service flags in the In-Service register. Before executing the IRET instruction that ends an interrupt service routine (ISR), the ISR should write to the EOI register to reset the IS bit for the interrupt. The specific EOI reset is the most secure method to use for resetting IS bits.

Example code for a specific EOI reset:

```

...
... ;ISR code
...
mov dx, EOI_ADDR
exit: mov ax, int_type ;load the interrupt type in ax
      out dx, ax ;write the interrupt type to EOI
      popa
      iret ;return from interrupt
    
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NSPEC	0	0	0	0	0	0	0	0	0	0	S4	S3	S2	S1	S0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
15	Non-Specific EOI	The NSPEC bit determines the type of EOI command. When written as a 1, NSPEC indicates non-specific EOI. When written as a 0, NSPEC indicates the specific EOI interrupt type in S4–S0.
4-0	Source EOI Type	Specifies the EOI type of the interrupt that is currently being processed.

**Figure 9.24 End-of-Interrupt Register**

## 10

## Timer Control Unit

There are three 16-bit programmable timers in the IMS16B. Timers 0 and 1 are highly versatile and are each connected to two external pins (TIMIN0, TIMOUT0, TIMIN1, TIMOUT1). These two timers can be used to count or time external events, or they can be used to generate non-repetitive or variable-duty-cycle waveforms.

Timer 1 can also be configured as a watchdog timer. The watchdog timer provides a mechanism for detecting software crashes or hangs. The TMROUT1 output is internally connected to the watchdog timer interrupt. Software developers must first program the TIMER1 Mode/Control, Count, and Max Count registers, and then program the Watchdog Timer Interrupt Control register. The TIMER1 Count register must be reloaded at intervals less than the TIMER1 max count to assure the watchdog interrupt is not taken. If the code crashes or hangs, the TIMER1 countdown can cause a watchdog interrupt.

Timer 2 is not connected to any external pins. It can be used for real-time coding and time-delay applications. It can also be used as a presale to timer 0 and timer 1 or as a DMA request source.

Offset	Register Mnemonic	Register Name
50h	T0CNT	Timer 0 Count
52h	T0CMPA	Timer 0 Maxcount Compare A
54h	T0CMPB	Timer 0 Maxcount Compare B
56h	T0CON	Timer 0 Mode/Control
58h	T1CNT	Timer 1 Count
5Ah	T1CMPA	Timer 1 Maxcount Compare A
5Ch	T1CMPB	Timer 1 Maxcount Compare B
5Eh	T1CON	Timer 1 Mode/Control
60h	T2CNT	Timer 2 Count
62h	T2CMPA	Timer 2 Maxcount Compare A
N/A	N/A	N/A
66h	T2CON	Timer 2 Mode/Control

**Figure 10 Timer Control Unit Register Summaries**

The timer-count registers contain the current value of a timer. The timer-count registers can be read or written at any time, regardless of whether the corresponding timer is running. The microcontroller increments the value of a timer-count register each time a timer event occurs. When the timer reaches the maximum value, it resets to 0 during the same clock cycle. In addition, timers 0 and 1 have a secondary maximum-count register.

Using both the primary and secondary maximum-count registers lets the timer alternate between two maximum values. If the timer is programmed to use only the primary maximum-count register, the timer output pin switches Low for one clock cycle, the clock cycle after the maximum value is reached. If the timer is programmed to use both of its maximum-count registers, the output pin creates a waveform by indicating which maximum-count register is currently in control. The duty cycle and frequency of the waveform depend on the values in the alternating maximum-count registers.

Each timer is serviced on every fourth clock cycle. Therefore, a timer can operate at a maximum speed of one-quarter of the internal clock frequency. A timer can be clocked externally at the same

maximum frequency of one-fourth of the internal clock frequency. However, because of internal synchronization and pipelining of the timer circuitry, the timer output takes up to six clock cycles to respond to the clock or gate input. The timers are run by the processor's internal clock. If power-save mode is in effect, the timers operate at the reduced power-save clock rate.

### 10.1 Timer 0 and Timer 1 Mode and Control Registers (T0CON, Offset 56h, T1CON, Offset 5Eh)

These registers control the functionality of timer 0 and timer 1. The value of T0CON and T1CON at reset is 0002h.

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

Reset value : 0000\_0000\_0000\_0010b

Bit	Name	Function
15	Enable Bit	This is the count enable bit for the timer. When set, the timer is enabled to increment. When cleared, the timer is disabled from counting. If CONT=0, the EN bit is automatically cleared upon a maximum count.
14	Inhibit Bit	INH is a write protect for the EN bit. If 1 during a write operation to the control registers, then EN is modified by the write. If 0 during a write, then EN is not modified by the write. This bit is not stored in the control register, and is always 0 when read.
13	Interrupt Bit	When set to 1, an interrupt request is generated when the count register equals a maximum count. If the timer is configured in dual maxcount Mode, an interrupt is generated each time the count reaches maxcount A or maxcount B. When INT is set to 0, the timer will not issue interrupt requests. If the enable bit is cleared after an interrupt request has been generated but before the pending interrupt is serviced, the interrupt request will still be present.
12	Register in Use Bit	When the Maxcount Compare A register is being used for comparison to the timer count value, this bit is set to 0. When the Maxcount Compare B register is being used, this bit is set to 1.
5	Maximum Count Bit	The MC bit is set to 1 when the timer reaches a maximum count. In dual maxcount mode, the bit is set each time either Maxcount Compare A or B register is reached. This bit is set regardless of the timer interrupt-enable bit. The MC bit can be used to monitor timer status through software polling instead of through interrupts.
4	Retrigger Bit	Determines the control function provided by the timer input pin. When set to 1, a 0 to 1 edge transition on TMRIN0 or TMRIN1 resets the count. When set to 0, a High input enables counting and a Low input holds the timer value. This bit is ignored when external clocking (EXT=1) is selected.
3	Prescaler Bit	When set to 1, the timer is prescaled by timer 2. When set to 0, the timer counts up every fourth CLKOUT period. This bit is ignored when external clocking is enabled (EXT=1).
2	External Clock Bit	When set to 1, an external clock is used. When set to 0, the internal clock is used.
1	Alternate Compare Bit	When set to 1, the timer counts to maxcount compare A, then resets the count register to 0. Then the timer counts to maxcount compare B, resets the count register to zero, and starts over with maxcount compare A. If ALT is clear, the timer counts to maxcount compare A and then resets the count register to zero and starts counting again against maxcount compare A. In this case, maxcount compare B is not used.

0	Continuous Mode Bit	When set to 1, CONT causes the associated timer to run in the normal continuous mode.
---	---------------------	---

**Figure 10.1 Timer 0 and Timer 1 Mode and Control Registers**

**10.2 Timer 2 Mode and Control Register (T2CON, Offset 66h)**

This register controls the functionality of timer 2. The value of T2CON at reset is 0000h.

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

Reset value : 0000\_0000\_0000\_0000b

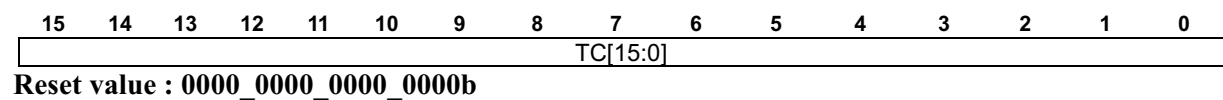
Bit	Name	Function
15	Enable Bit	When EN is set to 1, the timer is enabled. When set to 0, the timer is inhibited from counting. Do not write to this bit unless the INH bit is set to 1 during the same write.
14	Inhibit Bit	Allows selective updating of enable (EN) bit. When INH is set to 1 during a write, EN can be modified on the same write. When INH is set to 0 during a write, writes to EN are ignored. This bit is not stored and is always read as 0.
13	Interrupt Bit	When INT is set to 1, an interrupt request is generated when the count register equals a maximum count. When INT is set to 0, the timer will not issue interrupt requests. If the EN enable bit is cleared after an interrupt request has been generated, but before the pending interrupt is serviced, the interrupt request remains active.
5	Maximum Count Bit	The MC bit is set to 1 when the timer reaches its maximum count. This bit is set regardless of the timer interrupt-enable bit. The MC bit can be used to monitor timer status through software polling instead of through interrupts.
0	Continuous Mode Bit	When CONT is set to 1, it causes the associated timer to run continuously. When set to 0, EN is cleared after each timer counts sequence and the timer halts on reaching the maximum count.

**Figure 10.2 Timer 2 Mode and Control Register**

### 10.3 Timer Count Registers (T0CNT, Offset 50h, T1CNT, Offset 58h, T2CNT, Offset 60h)

These registers can be incremented by one every four internal processor clocks. Timer 0 and timer 1 can also be configured to increment based on the TMRIN0 and TMRIN1 external signals, or timer 2 can rescale them.

The count registers are compared to maximum count registers and various actions are triggered based on reaching a maximum count.



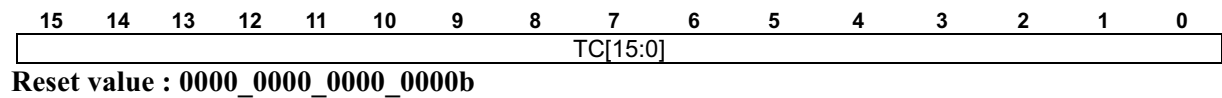
Bit	Name	Function
15 - 0	Timer Count Value	This register contains the current count of the associated timer. The count is incremented every fourth processor clock in internal clocked mode, or each time the timer 2 maxcount is reached if prescaled by timer 2. Timer 0 and timer 1 can be configured for external clocking based on the TMRIN0 and TMRIN1 signals.

**Figure 10.3 Timer Count Registers**

**10.4 Timer Maxcount Compare Registers  
(T0CMPA, Offset 52h, T0CMPB, Offset 54h, T1CMPA, Offset 5Ah,  
T1CMPB, Offset 5Ch, T2CMPA, Offset 62h)**

These registers serve as comparators for their associated count registers. Timer 0 and timer 1 each have two maximum count compare registers. Timer 0 and timer 1 can be configured to count and compare to register A and then count and compare to register B. Using this method, the TMROUT0 or TMROUT1 signals can be used to generate waveforms of various duty cycles.

Timer 2 has one compare register, T2CMPA. If a maximum count compare register is set to 0000h, the timer associated with that compare register will count from 0000h to FFFFh before requesting an interrupt. With a 40-MHz clock, a timer configured this way interrupts every 6.5536 ms.



Bit	Name	Function
15	Timer Count Value	This register contains the maximum value a timer will count to before resetting its count register to 0.
- 0		

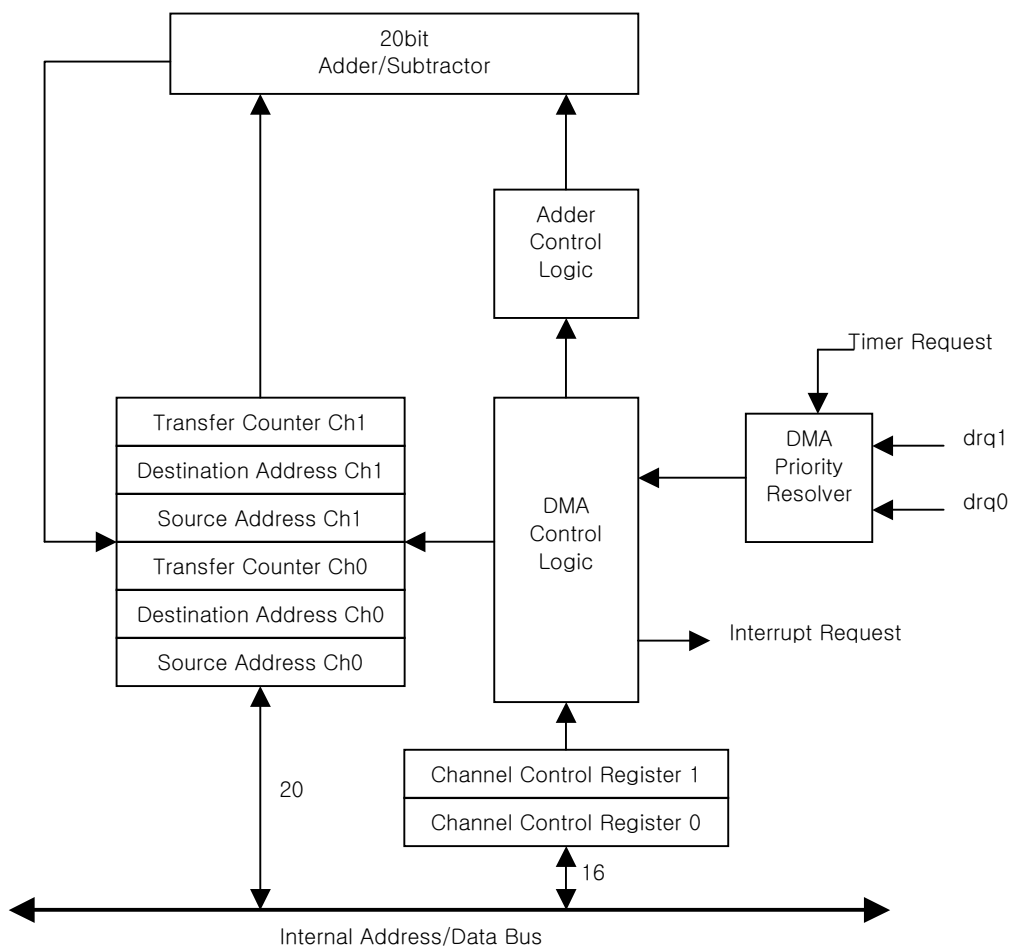
**Figure 10.4 Timer Maxcount Compare Registers**

**11**

**DMA Control Unit**

Direct memory access (DMA) permits transfer of data between memory and peripherals without CPU involvement. The DMA unit in the IMS16B microcontroller provides two high-speed DMA channels. Data transfers can occur between memory and I/O spaces (e.g., memory to I/O) or within the same space (e.g., memory-to-memory or I/O-to- I/O).

Either bytes or words can be transferred to or from even or odd addresses on the IMS16B. Two bus cycles (a minimum of eight clocks) are necessary for each data transfer. Each channel accepts a DMA request from one of two sources: the channel request pin (DRQ1, DRQ0) or Timer 2. The two DMA channels can be programmed with different priorities to resolve simultaneous DMA requests, and transfers on one channel can interrupt the other channel. Figure 11 Shows DMA Controller Block Diagram in IMS16B.



**Figure 11 DMA Controller Block Diagram**

## 11.1 DMA Operation

The format of the DMA control block is shown in Figure 11.1. Six registers in the peripheral control block define the operation of each channel. The DMA registers consist of a 20-bit source address (2 registers), a 20-bit destination address (2 registers), a 16-bit transfer count register, and a 16-bit control register.

Offset	Register Mnemonic	Register Name
C0h	D0SRCL	DMA 0 Source Address Low
C2h	D0SRCH	DMA 0 Source Address high
C4h	D0DSTL	DMA 0 Destination Address Low
C6h	D0DSTH	DMA 0 Destination Address High
C8h	D0TC	DMA 0 Transfer Count
CAh	D0CON	DMA 0 Control
D0h	D1SRCL	DMA 1 Source Address Low
D2h	D1SRCH	DMA 1 Source Address high
D4h	D1DSTL	DMA 1 Destination Address Low
D6h	D1DSTH	DMA 1 Destination Address High
D8h	D1TC	DMA 1 Transfer Count
DAh	D1CON	DMA 1 Control

**Figure 11.1 DMA Controller Register Summary**

The DMA transfer count register (DTC) specifies the number of DMA transfers to be performed. Up to 64 Kbytes or 64K words can be transferred with automatic termination. The DMA control registers define the channel operations (see Figure 11.1). All registers can be modified or altered during any DMA activity. Any changes made to these registers are reflected immediately in DMA operation. The sections on the following pages describe the control registers that are used to configure and operate the two DMA channels.

## 11.2 DMA Control Registers (D0CON, Offset CAh, D1CON, Offset DAh)

The DMA control registers determine the mode of operation for the DMA channels. These registers specify the following options:

- Whether the destination address is memory or I/O space
- Whether the destination address is incremented, decremented, or maintained constant after each transfer
- Whether the source address is memory or I/O space
- Whether the source address is incremented, decremented, or maintained constant after each transfer
- If DMA activity ceases after a programmed number of DMA cycles
- If an interrupt is generated after the last transfer
- The mode of synchronization
- The relative priority of one DMA channel with respect to the other DMA channel
- Whether timer 2 DMA requests are enabled or disabled
- Whether bytes or words are transferred

The DMA channel control registers can be changed while the channel is operating. Any changes made during DMA operations affect the current DMA transfer. The value of D0CON and D1CON at reset is 0000h.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM IO	DD EC	DI NC	SM IO	SD EC	SIN C	TC	INT	SYN 1	SYN 0	P	TDR Q	0	CHG	ST	BW

Reset value : 0000\_0000\_0000\_0000b

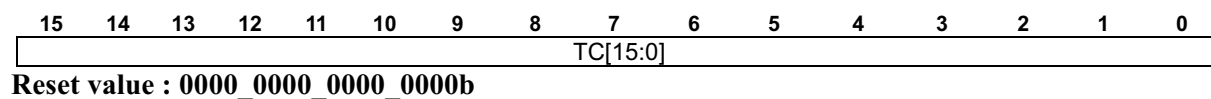
Bit	Name	Function
15	Destination Address Space Select	Selects memory or I/O space for the destination address. When DMIO is set to 1, the destination address is in memory space. When set to 0, the destination address is in I/O space.
14	Destination Decrement	When DDEC is set to 1, the destination address is automatically decremented after each transfer. The address decrements by 1 or 2, depending on the byte/word bit (BW, bit 0). The address remains constant if the increment and decrement bits are set to the same value (00b or 11b).
13	Destination Increment	When DINC is set to 1, the destination address is automatically incremented after each transfer. The address increments by 1 or 2, depending on the byte/word bit (BW, bit 0). The address remains constant if the increment and decrement bits are set to the same value (00b or 11b).
12	Source Address Space Select	When SMIO is set to 1, the source address is in memory space. When set to 0, the source address is in I/O space.
11	Source Decrement	When SDEC is set to 1, the source address is automatically decremented after each transfer. The address decrements by 1 or 2 depending on the byte/word bit (BW, bit 0). The address remains constant if the increment and decrement bits are set to the same value (00b or 11b).
10	Source Increment	When SINC is set to 1, the source address is automatically incremented after each transfer. The address increments by 1 or 2 depending on the byte/word bit (B/W, bit 0). The address remains constant if the increment and decrement bits are set to the same value (00b or 11b).
9	Terminal Count	The DMA decrements the transfer count for each DMA transfer. When TC is set to 1, source or destination synchronized DMA transfers terminate when the count reaches 0. When TC is set to 0, source or destination

		synchronized DMA transfers do not terminate when the count reaches 0. Unsynchronized DMA transfers always terminate when the count reaches 0, regardless of the setting of this bit.															
8	Interrupt	When INT is set to 1, the DMA channel generates an interrupt request on completion of the transfer count. The TC bit must also be set to generate an interrupt.															
7-6	Synchronization Type	The SYN1, SYN0 bits select channel synchronization as shown below. <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>SYN1</th> <th>SYN0</th> <th>Sync Type</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Unsynchronized</td> </tr> <tr> <td>0</td> <td>1</td> <td>Source Synch</td> </tr> <tr> <td>1</td> <td>0</td> <td>Destination Synch</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table>	SYN1	SYN0	Sync Type	0	0	Unsynchronized	0	1	Source Synch	1	0	Destination Synch	1	1	Reserved
SYN1	SYN0	Sync Type															
0	0	Unsynchronized															
0	1	Source Synch															
1	0	Destination Synch															
1	1	Reserved															
5	Relative Priority	When P is set to 1, it selects high priority for this channel relative to the other channel during simultaneous transfers.															
4	Timer Enable/Disable Request	When TDRQ is set to 1, it enables DMA requests from timer 2. When set to 0, TDRQ disables DMA requests from timer 2.															
2	Change Start Bit	This bit must be set to 1 during a write to allow modification of the ST bit. When change start bit is set to 0 during a write, ST is not altered when writing the control word.															
1	Start/Stop DMA Channel	The DMA channel is started when the start bit is set to 1. This bit can be modified only when the change start bit is set to a 1 during the same register writes.															
0	Byte/Word Select	On the IMS16B microcontroller, when BW is set to 1, word transfers are selected. When BW is set to 0, byte transfers are selected.															

**Figure 11.2 DMA Control Registers**

### 11.3 DMA Transfer Count Registers (D0TC, Offset C8h, D1TC, Offset D8h)

Each DMA channel maintains a 16-bit DMA Transfer Count register (DTC). This register is decremented after every DMA cycle, regardless of the state of the TC bit in the DMA Control register. However, if the TC bit in the DMA control word is set or if unsynchronized transfers are programmed, DMA activity terminates when the Transfer Count register reaches 0.



Bit	Name	Function
15-0	DMA Transfer Count	Contains the transfer count for a DMA channel. Value is decremented by 1 after each transfer.

**Figure 11.3 DMA Transfer Count Registers**

### 11.4 DMA Destination Address High Register (High Order Bits) (D0DSTH, Offset C6h, D1DSTH, Offset D6h)

Each DMA channel maintains a 20bit destination and a 20bit source register. Each register takes up two full 16-bit registers (the high register and the low register) in the peripheral control block. For each DMA channel to be used, all four registers must be initialized. These registers can be individually incremented or decremented after each transfer. If word transfers are performed, the address is incremented or decremented by 2 after each transfer. If byte transfers are performed, the address is incremented or decremented by 1.

Each register can point into either memory or I/O space. The user must program the upper four bits to 0000b in order to address the normal 64K I/O space. Since the DMA channels can perform transfers to or from odd addresses, there is no restriction on values for the destination and source address registers. Higher transfer rates can be achieved on the IMS16B microcontroller if all word transfers are performed to or from even addresses so that accesses occur in single, 16-bit bus cycles.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	DDA[19:0]			

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
3-0	DMA Destination Address High	These bits are driven onto A19-A16 during the write phase of a DMA transfer.

Figure 11.4 DMA Destination Address High Register

### 11.5 DMA Destination Address Low Register (Low Order Bits) (D0DSTL, Offset C4h, D1DSTL, Offset D4h)

The Figure 11.5 shows the DMA Destination Address Low Register.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DDA[15:0]															

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
15-0	DMA Destination Address Low	These bits are driven onto A15-A0 during the write phase of a DMA transfer.

Figure 11.5 DMA Destination Address Low Register

### 11.6 DMA Source Address High Register (High Order Bits) (D0SRCH, Offset C2h, D1SRCH, Offset D2h)

Each DMA channel maintains a 20bit destination and a 20bit source register. Each register takes up two full 16bit registers (the high register and the low register) in the peripheral control block. For each DMA channel to be used, all four registers must be initialized. These registers can be individually incremented or decremented after each transfer. If word transfers are performed, the address is incremented or decremented by 2 after each transfer. If byte transfers are performed, the address is incremented or decremented by 1.

Each register can point into either memory or I/O space. The user must program the upper four bits to 0000b in order to address the normal 64K I/O space. Since the DMA channels can perform transfers to or from odd addresses, there is no restriction on values for the destination and source address registers. Higher transfer rates can be achieved on the IMS16B microcontroller if all word transfers are performed to or from even addresses so that accesses occur in single, 16-bit bus cycles.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	DSA[19:0]			

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
3-0	DMA Source Address High	These bits are driven onto A19-A16 during the read phase of a DMA transfer.

Figure 11.6 DMA Source Address High Register

### 11.7 DMA Source Address Low Register (Low Order Bits) (D0SRCL, Offset C0h, D1SRCL, Offset D0h)

The Figure 11.7 shows the DMA Destination Address Low Register.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSA[15:0]															

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
15-0	DMA Source Address Low	These bits are driven onto A15-A0 during the read phase of a DMA transfer.

Figure 11.7 DMA Source Address Low Register

## 11.8 DMA Requests

Data transfers can be either source or destination synchronized - either the source of the data or the destination of the data can request the data transfer. DMA transfers can also be unsynchronized (i.e., the transfer takes place continually until the correct number of transfers has occurred).

During source synchronized or unsynchronized transfers, the DMA channel can begin a transfer immediately after the end of the previous DMA transfer, and a complete transfer can occur every two bus cycles or eight clock cycles (assuming no wait states). When destination synchronization is performed, data is not fetched from the source address until the destination device signals that it is ready to receive it.

When destination synchronized transfers are requested, the DMA controller relinquishes control of the bus after every transfer. If no other bus activity is initiated, another DMA cycle begins after two processor clocks. This allows the destination device time to remove its request if another transfer is not desired.

When the DMA controller relinquishes the bus during destination-synchronized transfers, the CPU can initiate a bus cycle. As a result, a complete bus cycle is often inserted between destination-synchronized transfers.

Type of Synchronization	CPU Running	CPU Halted
Unsynchronized	10 Mbytes/s	10 Mb/s
Source Synchronization	10 Mb/s	10 Mb/s
Destination Synchronization	6.6 Mb/s	8 Mb/s

**Figure 11.8 Maximum DMA Transfer Rates at 40Mhz for IMS16B**

## 11.9 DMA Acknowledge

No explicit DMA acknowledge signal is provided. But, status bit S6 pin of IMS16B will be set to 1 during DMA operations, and reset to 0 during CPU bus cycles.

## 11.10 DMA Priority

The DMA channels can be programmed so that one channel is always given priority over the other, or they can be programmed to alternate cycles when both have DMA requests. The default priority for alternating cycles is DMA0(highest), followed by DMA1(lowest).

DMA cycles always have priority over internal CPU cycles except between internally locked memory accesses or word accesses to odd memory locations. However, an external bus hold takes priority over an internal DMA cycle.

Because an interrupt request cannot suspend a DMA operation and the CPU cannot access memory during a DMA cycle, interrupt latency time suffers during sequences of continuous DMA cycles. An NMI request, however, causes all internal DMA activity to halt. This allows the CPU to respond

quickly to the NMI request.

### 11.11 DMA Programming

DMA cycles occur whenever the ST bit of the control register is set. If synchronized transfers are programmed, a DRQ must also be generated. Therefore, the source and destination transfer address registers and the transfer count register (if used) must be programmed before the ST bit is set. Each DMA register can be modified while the channel is operating. If the CHG bit is set to 0 when the control register is written, the ST bit of the control register will not be modified by the write. If multiple channel registers are modified, an internally Locked string transfer should be used to prevent a DMA transfer from occurring between updates to the channel registers.

### 11.12 DMA Channels on Reset

On reset, the state of the DMA channels is as follows:

- The ST bit for each channel is reset.
- Any transfer in progress is aborted.
- The values of the transfer count registers, source address registers, and destination address registers are undefined.

## 12

**UART(Asynchronous) Serial Interface**

The IMS16B microcontroller provides an asynchronous serial port. The asynchronous serial port is a two-pin interface that permits full-duplex bidirectional data transfer. The asynchronous serial port supports the following features:

- Full-duplex operation
- 7-bit or 8-bit data transfers
- Odd parity, even parity, or no parity
- 1 or 2 stop bits
- Loopback mode
- Ability to break character transmit
- Does not contain DMA support

The asynchronous serial port transmit and receive sections are double-buffered. Break character recognition, framing, parity, and overrun error detection are provided. The user programs exception interrupt generation. The transmit/receive clock is based on the internal processor clock internally divided down to the serial port operating frequency. If power-save mode is in effect, the divide factor must be reprogrammed. The serial port permits 7-bit and 8-bit data transfers. DMA transfers through the serial port are not supported. The serial port generates one interrupt for all serial port events (transmit complete, data received, or error). The Serial Port Status register contains the reason for the serial port interrupt. The interrupt type assigned to the serial port is 14h. The serial port can be used in power-save mode, but the transfer rate must be adjusted to correctly reflect the new internal operating frequency and the serial port must not receive any information until the frequency is changed.

To transmit a word of data, the word must be written to the transmit data holding register while the register is empty. When the transmit shift register is empty the word will be transferred freeing the holding register for the next word of data. Therefore the next word of data may be written while the previous word is still being transmitted. The software may poll the transfer holding register empty bit in the status register, or the asynchronous serial port may be configured to generate an interrupt whenever the holding register is empty.

When the receive portion of the serial port receives a valid word of data the data ready bit in the status register will be asserted. The software then has until the next word of data is received to read the previous word. If the word is not read on time it will be overwritten by the next word of data and the overrun error bit in the status register will be set.

**12.1 PROGRAMMABLE REGISTERS**

The asynchronous serial port is programmed through the use of five, 16-bit peripheral registers.

Offset	Register Mnemonic	Register Name
80h	SPTC	Serial Port Control
82h	SPSTS	Serial Port Status
84h	SPTD	Serial Port Transmit Data
86h	SPRD	Serial Port Receive Data

88h	SPBAUS	Serial Port Baud Rate Divisor
-----	--------	-------------------------------

**Figure 12.1 Asynchronous Serial Port Register Summary**

### 12.2 Serial Port Control Register (SPCT, Offset 80h)

The Serial Port Control register controls both transmit and receive sections of the serial port. The value of SPCT at reset is 0000h.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TXI E	RXI E	LO OP	BR K	BRK VAL	PMO DE1	PMO DE0	WLG N	STP	TMO DE	RSI E	RM ODE

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function								
11	Transmit Holding Register Empty Interrupt Enable	This bit enables the serial port to generate an interrupt for transmit holding register empty condition, indicating that the serial port is ready to accept a new character for transmission. If this bit is 1 and the Serial Port Transmit Holding register does not contain valid data, the serial port generates an interrupt request. This bit is reset to 0.								
10	Receive Data Ready Interrupt Enable	This bit enables the serial port to generate an interrupt for the receive data ready condition. If this bit is 1 and the Serial Port Receive Buffer register contains data that has been received on the serial port, the serial port generates an interrupt request. This bit is reset to 0.								
9	Loopback	Setting this bit to 1 places the serial port in the loopback mode. In this mode, the TXD output is set High and the transmit shift register is connected to the receive shift register. Data transmitted by the transmit section is immediately received by the receive section. The loopback mode is provided for testing the serial port. This bit is reset to 0.								
8	Send Break	Setting this bit to 1 causes the serial port to send a continuous level on the TXD output. A break is a continuous Low on the TXD output for duration of more than one frame transmission time. The BRKVAL bit determines the level driven on the TXD output. To use the transmitter to time the frame, set the BRK bit when the transmitter is empty (indicated by the TEMT bit of the Serial Port Status register), write the serial port transmit holding register, then wait until the TEMT bit is again set before resetting the BRK bit. Since the TXD output is held constant while BRK is set, the data written to transmit holding register will not appear on the pin. This bit is reset to 0.								
7	Break Value	This bit determines the output value transmitted on the TXD pin during a send break operation. If BRKVAL is 1, a continuous High level is driven on the TXD output. If BRKVAL is 0, a continuous Low level is driven on the TXD output. Only a continuous Low value (BRKVAL=0) will result in a break being detected by the receiver. This bit is reset to 0.								
6-5	Parity Mode	This field specifies how parity generation and checking are performed during transmission and reception. If parity checking and generation is selected, a parity bit is received or sent in addition to the specified number of data bits. The value of PMODE after power-on reset is 00b. <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>PMODE</th> <th>Parity</th> </tr> </thead> <tbody> <tr> <td>0 X</td> <td>None(No parity bit in frame)</td> </tr> <tr> <td>1 0</td> <td>Odd(Odd number of 1s in frame)</td> </tr> <tr> <td>1 1</td> <td>Even(Even number of 1s in frame)</td> </tr> </tbody> </table>	PMODE	Parity	0 X	None(No parity bit in frame)	1 0	Odd(Odd number of 1s in frame)	1 1	Even(Even number of 1s in frame)
PMODE	Parity									
0 X	None(No parity bit in frame)									
1 0	Odd(Odd number of 1s in frame)									
1 1	Even(Even number of 1s in frame)									
4	Word Length	This bit determines the number of bits transmitted or received in a frame. If WLG N is 0, the serial port sends and receives 7 bits of data per frame. If WLG N is 1, the serial port sends and receives 8 bits of data per frame. The value of WLG N after power-on reset is 0.								
3	Stop Bits	A 0 in the STP bit specifies that one stop bit is used to signify the end of a								

		frame. A 1 in this bit specifies that two stop bits be used to signify the end of a frame. The value of STP after power-on reset is 0.
2	Transmit Mode	The TMODE bit enables data transmission and controls the operational mode of the serial port for the transmission of data. If TMODE is 0, the transmit section and transmit interrupts of the serial port are disabled. If TMODE is 1, the transmit section of the serial port is enabled. The value of TMODE after power-on reset is 0.
1	Receive Status Interrupt Enable	This bit enables the serial port to generate an interrupt because of an exception during reception. If this bit is 1 and the serial port receives a break, or experiences a framing error, parity error, or overrun error, the serial port generates a serial port interrupt. The value of RSIE after power-on reset is 0.
0	Receive Mode	This field enables data reception and controls the operational mode of the serial port for the reception of data. If RMODE is 0, the receive section and receive interrupts of the serial port are disabled. If RMODE is 1, receive section of the serial port is enabled. The value of RMODE after power-on reset is 0.

**Figure 12.2 Serial Port Control Register**

### 12.3 Serial Port Status Register (SPSTS, Offset 82h)

The Serial Port Status register indicates the status of transmits and receives sections of the serial port.

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

Reset value : 0000\_0000\_0110\_0000b

Bit	Name	Function
6	Transmit Empty	The TEMT bit is 1 when the transmitter has no data to transmit and the transmit shift register is empty. This indicates to software that it is safe to disable the transmit section. This bit is read-only.
5	Transmit Holding Register Empty	When the THRE bit is 1, transmit holding register contains invalid data and can be written with data to be transmitted. When the THRE bit is 0, transmit holding register cannot be written because it contains valid data that has not yet been copied to the transmit shift register for transmission. If transmit interrupts are enabled by the TMODE and TXIE fields, a serial port interrupt request is generated when the THRE bit is 1. The THRE bit is reset automatically by writing transmit holding register. This bit is read-only, allowing other bits of the Serial Port Status register to be written (i.e., resetting the BRKI bit) without interfering with the current data request.
4	Receive Data Ready	When the RDR bit is 1, receive buffer register contains data that can be read. When the RDR bit is 0, receive buffer register does not contain valid data. This bit is read-only. If receive interrupts are enabled by the RMODE and RXIE fields, a serial port interrupt request is generated when the THRE bit is 1. Reading receive buffer register resets the RDR bit.
3	Break Interrupt	The BRKI bit is set to indicate that a break has been received. If the RSIE bit is 1, the BRKI bit being set causes a serial port interrupt request. The BRKI bit should be reset by software.
2	Framing Error	The FER bit is set to indicate that a framing error occurred during reception of data. If the RSIE bit is 1, the FER bit being set causes a serial port interrupt request. The FER bit should be reset by software.
1	Parity Error	The PER bit is set to indicate that a parity error occurred during reception of data. If the RSIE bit is 1, the PER bit being set causes a serial port interrupt request. The PER bit should be reset by software.
0	Overrun Error	The OER bit is set when an overrun error occurs during reception of data. If the RSIE bit is 1, the OER bit being set causes a serial port interrupt request. The OER bit should be reset by software.

Figure 12.3 Serial Port Status Register

### 12.4 Serial Port Transmit Data Register (SPTD, Offset 84h)

Software writes this register with data to be transmitted on the serial port. The transmitter is double-buffered, and the transmit section copies data from the transmit data register to the transmit shift register (which is not accessible to software) before transmitting the data.

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

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
7-0	Transmit Data	This field is written with data to be transmitted on the serial port. The THRE bit in the Serial Port Status register indicates whether there is valid data in the SPTD register. To avoid overwriting data in the SPTD register, the THRE bit should be read as a 1 before writing this register. Writing this register causes the THRE bit to be reset.

Figure 12.4 Serial Port Transmit Data Register

### 12.5 Serial Port Receive Data Register (SPRD, Offset 86h)

This register contains data received over the serial port. The receiver is double-buffered, and the receive section can be receiving a subsequent frame of data in the receive shift register (which is not accessible to software) while the receive data register is being read by software.

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

Reset value : Not Defined

Bit	Name	Function
7-0	Receive Data	This field contains data received on the serial port. The RDR bit of the Serial Port Status register indicates valid data in the SPRD register. To avoid reading invalid data, the RDR bit should be read as a 1 before the SPRD register is read. Reading this register causes the RDR bit to be reset.

Figure 12.5 Serial Port Receive Data Register

## 12.6 Serial Port Baud Rate Divisor Register (SPBAUD, Offset 88h)

This register specifies a clock divisor for the generation of the serial clock that controls the serial port. The serial clock rate is 16 times the baud rate of transmission or reception of data. The SPBAUD register specifies the number of internal processor cycles in one phase (half period) of the 16x serial clock. If power-save mode is in effect, the baud rate divisor must be reprogrammed to reflect the new processor clock frequency.

A general formula for the baud rate divisor is:

$$BAUDDIV_{(decimal)} = (\text{Internal Processor Clock} / (32 * \text{Baud Rate})) - 1$$

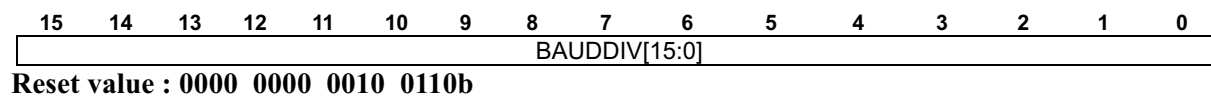
The maximum baud rate is 1/32 of the internal processor clock and is achieved by setting BAUDDIV=0000h.

For a 40MHz internal processor clock (If PLL is not enabled, externally applied clock frequency must be 80Mhz. That is, core clock speed is a half of external clock), a baud rate of 9600 can be achieved with

$$40000000 / (32 * 9600) = 40000000 / 307200 = 130.208$$

$$130 - 1 = 129$$

BAUDDIV=129 (81h). A 1% error applies.



Bit	Name	Function
15-0	Baud Rate Divisor	This field specifies the divisor for the internal processor clock that generates one phase (half period) of the serial clock. The serial clock operates at 16 times the data transmission or reception baud rate.

**Figure 12.6 Serial Port Baud Rate Divisor Register**

## 13

## Synchronous Serial Interface

In addition to the UART serial port, the IMS16B provides a synchronous serial interface. Unlike the asynchronous serial port, the SSI operates in a master/slave configuration. The IMS16B microcontroller operates as the master port. The SSI interface provides four pins for communicating with system components: two enables (SDEN0 and SDEN1), a clock (SCLK), and a data pin (SDATA). Figure 13 shows synchronous serial interface register summary.

Offset	Register Mnemonic	Register Name
10h	SSS	Synchronous Serial Status
12h	SSC	Synchronous Serial Control
14h	SSD1	Synchronous Serial Transmit 1
16h	SSD0	Synchronous Serial Transmit 0
18h	SSR	Synchronous Serial Receive

**Figure 13 Synchronous Serial Interface Register Summaries**

The SDEN1, SDEN0 enable pins can be enabled for up to two peripheral devices. Transmit and receive operations are synchronized between the master (IMS16B) and slave (peripheral) by means of the SCLK output. SCLK is derived from the processor internal clock divided by 2, 4, 8, or 16, as specified by the SSC register. SCLK is only driven during data transmit or receive operations. The inactive state of SCLK is High. If power-save mode is in effect, the SCLK frequency is affected by the reduced processor clock frequency.

Data is transferred across the SDATA input/output pin. Data is driven on the falling edge of SCLK and latched on the rising edge of SCLK. The least-significant bit of the data is shifted first for both transmit and receive operations. During write operations, the processor holds data for one-half of an SCLK period following the transfer of the last data bit. SDATA has a weak keeper that holds the last value of SDATA on the pin.

### 13.1 Synchronous Serial Status Register (SSS, Offset 10h)

This read-only register indicates the state of the SSI port. The value of the SSS register at reset is 0000h.

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

**Reset value : 0000\_0000\_0000\_0000b**

Bit	Name	Function
2	Receive/Transmit Error Detect	This bit is set when the SSI detects either a read of the Synchronous Serial Receive register or a write to one of the transmit registers while the SSI is busy (PB=1). This bit is reset when the SDEN output is inactive (bits DE1 - DE0 in the SSC register are both 0).
1	Data Receive/Transmit Complete	The DR/DT bit is set at the end of the transfer of data bit 7 (SCLK rising edge) during transmit or receive operation. This bit is reset when the SSR register is read, when one of the SSD0 or SSD1 registers 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.
0	SSI Port Busy	When the PB bit is set, transmit or receive operation is in progress. When PB is reset, the port is ready to transmit or receive data.

**Figure 13.1 Serial Port Status Register**

### 13.2 Synchronous Serial Control Register (SSC, Offset 12h)

This read/write register controls the operation of the SDEN0 - SDEN1 outputs and the transfer rate of the SSI port. The SDEN0 and SDEN1 outputs are asserted when a 1 is written to the corresponding bit. However, in the case when both DE0 and DE1 are set, only SDEN0 will be asserted. The value of the SSC register at reset is 0002h.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	SCLKDIV[1:0]	0	0	0	DE1	DE0

Reset value : 0000\_0000\_0000\_0010b

Bit	Name	Function										
5-4	SCLK Divide	<p>These bits determine the SCLK frequency. SCLK is derived from the internal processor clock by dividing by 2, 4, 8, or 16. If power-save mode is in effect, the SCLK frequency is affected by the reduced processor clock frequency.</p> <table border="1"> <thead> <tr> <th>SCLKDIV</th> <th>SCLK Frequency Divider</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>Processor clock / 2</td> </tr> <tr> <td>01</td> <td>Processor clock / 4</td> </tr> <tr> <td>10</td> <td>Processor clock / 8</td> </tr> <tr> <td>11</td> <td>Processor clock / 16</td> </tr> </tbody> </table>	SCLKDIV	SCLK Frequency Divider	00	Processor clock / 2	01	Processor clock / 4	10	Processor clock / 8	11	Processor clock / 16
SCLKDIV	SCLK Frequency Divider											
00	Processor clock / 2											
01	Processor clock / 4											
10	Processor clock / 8											
11	Processor clock / 16											
1	SDEN1 Enable	When this bit is set to 1, the SDEN1 pin is held High. When DE1 is set to 0, the SDEN1 pin is Low.										
0	SDEN0 Enable	When this bit is set to 1, the SDEN0 pin is held High. When DE0 is set to 0, the SDEN0 pin is Low.										

Figure 13.2 Synchronous Serial Control Register

### 13.3 Synchronous Serial Transmit 1 Register (SSD1, Offset 14h) Synchronous Serial Transmit 0 Register (SSD0, Offset 16h)

The Synchronous Serial Transmit 1 and 0 registers contain data to be transferred from the processor to the peripheral on a write operation. Only the least-significant 8 bits of the register are used. A write to either of these registers while the port is not busy will cause transmit to start and the busy bit to be set. A write to either of these registers while the busy bit is set will cause the error bit in the status register to be set and will not affect the ongoing transmit or receive.

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

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
7-0	Send Data	Data to transmit over the SDATA pin. Bit 0 is transmitted first. Bit 7 is transmitted last.

Figure 13.3 Synchronous Serial Transmit Register

### 13.4 Synchronous Serial Receive Register (SSR, Offset 18h)

The Synchronous Serial Receive (SSR) register contains the data transferred from the peripheral to the processor on a read operation. Only the least-significant 8 bits of the register are used. A receive data transmission is initiated by reading the SSR register while the port is not busy (PB bit in SSS register is 0) and one or both of the enable bits (DE1-DE0 in the SSC register) is set. A receive transmission is not initiated by reading the SSR register when neither of the enable bits is set (DE[1:0] = 00b). This allows the software to read the received data without initiating another receive transmission. A read of the Synchronous Serial Receive register while the port is busy (PB bit is set in the SSS register) sets the RE/TE (Receive/Transmit Error) bit in the SSS register and returns an indeterminate value. Such a read does not generate additional data transfers.

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

Reset value : Not defined

Bit	Name	Function
7-0	Receive Data	Data received over the SDATA pin.

Figure 13.4 Synchronous Serial Receive Register

## 14

## Programmable I/O Unit

The 32 pins on the IMS16B microcontroller are available as user-programmable I/O signals (PIOs). Each of these pins can be used as a PIO if the normal function of the pin is not needed. If a pin is enabled to function as a PIO signal, the normal function is disabled and does not affect the pin. A PIO signal can be configured to operate as an input or output with or without internal pullup or pulldown resistors.

After power-on reset, the PIO pins default to various configurations. The column titled Power-On Reset State in Figure 14 lists the defaults for the PIOs. The system initialization code must reconfigure PIOs as required.

The AIO[2:0](A[19:17]) address pins default to normal operation on power-on reset, allowing the processor to correctly begin fetching instructions at the boot address FFFF0h. The DTR\_N, DEN\_N, and SRDY pins also default to normal operation on power-on reset.

PIO No.	Associated Pin	Power-On Reset Status
0	TMRIN1	Input with pullup
1	TMROUT1	Input with pulldown
2	PCS6_N	Input with pullup
3	PCS5_N	Input with pulldown
4	DTR_N	Normal operation
5	DEN_N	Normal operation
6	SRDY	Normal operation
7	AIO2(A19)	Normal operation
8	AIO1(A18)	Normal operation
9	AIO0(A17)	Normal operation
10	TMROUT0	Input with pulldown
11	TMRIN0	Input with pullup
12	DRQ0	Input with pullup
13	DRQ1	Input with pullup
14	MCS0_N	Input with pullup
15	MCS1_N	Input with pullup
16	PCS0_N	Input with pullup
17	PCS1_N	Input with pullup
18	PCS2_N	Input with pullup
19	PCS3_N	Input with pullup
20	SCLK	Input with pullup
21	SDATA	Input with pullup
22	SDEN0	Input with pulldown
23	SDEN1	Input with pulldown
24	MCS2_N	Input with pullup
25	MCS3_N	Input with pullup
26	UZI_N	Input with pullup
27	TXD	Input with pullup
28	RXD	Input with pullup
29	S6	Input with pullup
30	INT4	Input with pullup
31	INT2	Input with pullup

**Figure 14 PIO Pin Assignments**

PIO Mode	PIO Direction	Pin Function
0	0	Normal operation
0	1	PIO input with pullup/pulldown
1	0	PIO output
1	1	Reserved

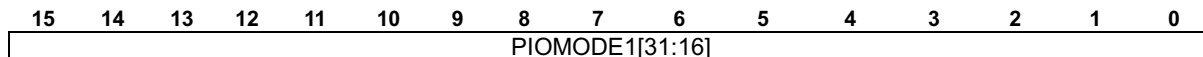
**Figure 14-1 PIO Mode and PIO Direction Settings**

Offset	Register Mnemonic	Register Name
70h	PIOMODE0	PIO Mode 0 Register
72h	PDIR0	PIO Direction 0 Register
74h	PDATA0	PIO Data Register 0
76h	PIOMODE1	PIO Mode 1 Register
78h	PDIR1	PIO Direction 1 Register
7Ah	PDATA1	PIO Data Register 1

**Figure 14-2 Programmable I/O's Register Summary**

### 14.1 PIO Mode 1 Register (PIOMODE1, Offset 76h)

The value of PIOMODE1 at reset is 0000h.

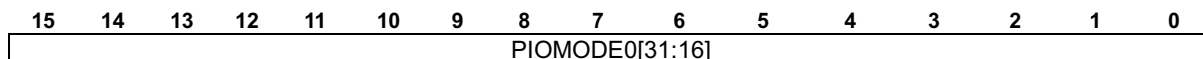


Bit	Name	Function
15-0	PIO Mode Bits[31:16]	This field with the PIO direction registers determines whether each PIO pin performs its pre-assigned function or is enabled as a custom PIO signal. The most significant bit of the PMODE field determines whether PIO31 is enabled, the next bit determines whether PIO30 is enabled, and so on.

**Figure 14.1 PIO Mode 1 Register**

### 14.2 PIO Mode 0 Register (PIOMODE0, Offset 70h)

The value of PIOMODE0 at reset is 0000h.



Bit	Name	Function
15-0	PIO Mode Bits[15:0]	This field is a continuation of the PMODE field in the PIO Mode 1 register.

**Figure 14.2 PIO Mode 0 Register**



## Figure 14.6 PIO Data 0 Register I<sup>2</sup>C Serial Interface

The I2C bus is a popular serial, two-wire interface used in many systems because of its low overhead. The two-wire interface minimizes interconnections so ICs have fewer pins, and the number of traces required on printed circuit boards is reduced. Capable of 100 KHz operations, each device connected to the bus is software addressable by a unique address with a simple Master/Slave protocol.

The I2C bus consists of two wires, serial data (SDA) and serial clock (SCL), which carry information between the devices connected to the bus. The number of devices connected to the same bus is limited only by a maximum bus capacitance of 400pF. Both the SDA and SCL lines are bidirectional lines, connected to a positive supply voltage via a pull-up resistor. When the bus is free, both lines are High. The output stages of devices connected to the bus must have an open-drain or open-collector in order to perform the wired-AND function.

Each device on the bus has a unique address and can operate as either a transmitter or receiver. In addition, devices can also be configured as Masters or Slaves. A Master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. Any other device that is being addressed is considered a Slave. The I2C protocol defines an arbitration procedure that insures that if more than one Master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted. The IMS16B I2C controller supports the arbitration and clock synchronization procedures defined in the I2C specification.

Data transfers on the I2C bus are initiated with a START condition and are terminated with a STOP condition. Normal data on the SDA line must be stable during the High period of the clock. The High or Low state of the data line can only change when SCL is Low. The START condition is a unique case and is defined by a High-to-Low transition on the SDA line while SCL is High. Likewise, the STOP condition is a unique case and is defined by a Low-to-High transition on the SDA line while SCL is High. The definitions of data, START, and STOP insure that the START and STOP conditions will never be confused as data. This is shown in Figure 15.

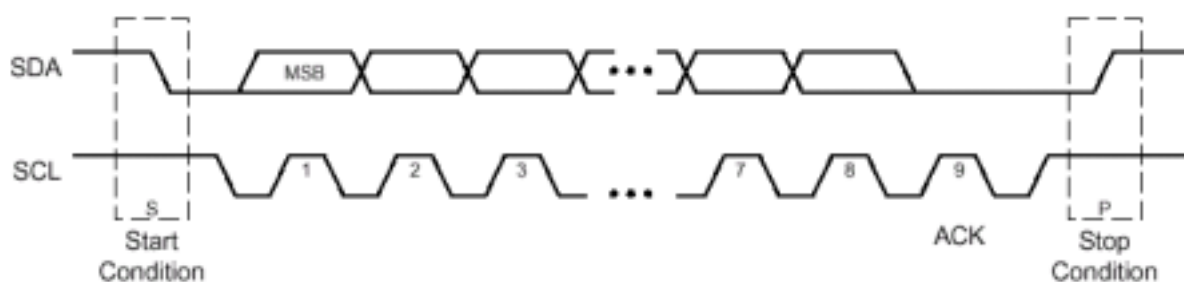


Figure 15. Data Transfer on the I2C Bus

Each data packet on the I2C bus consists of eight bits of data followed by an acknowledge bit so one complete data byte transfer requires nine clock pulses. Data is transferred with the most significant bit first (MSB). The transmitter releases the SDA line during the acknowledge bit and the receiver of the data transfer must drive the SDA line low during the acknowledge bit to acknowledge receipt of the data. If a Slave-receiver does not drive the SDA line Low during acknowledge bit, this indicates that the Slave-receiver was unable to accept the data and the Master can then generate a STOP condition to

abort the transfer. If the Master-receiver does not generate acknowledge, this indicates to the Slave-transmitter that this byte was the last byte of the transfer.

Standard communication on the bus between a Master and a Slave is composed of four parts: START, Slave address, data transfer, and STOP.

The I2C protocol defines a data transfer format for both 7-bit and 10-bit addressing. The implementation of the I2C controller in the IMS16B supports the 7-bit address format. After the START condition, a Slave address is sent. This address is seven bits long followed by an 8th-bit which is the read/write bit. A '1' indicates a request for data (read) and a '0' indicates a data transmission (write).

Only the Slave with the calling address that matches the address transmitted by the Master responds by sending back an acknowledge bit by pulling the SDA line Low on the 9-th clock. Once successful Slave addressing is achieved, the data transfer can proceed byte-by-byte as specified by the read/write bit. The Master can terminate the communication by generating a STOP signal to free the bus. However, the Master may generate a START signal without generating a STOP signal first. This is called a repeated START.

Figure 15-1 shows I2c Serial Interface Register Summary.

Offset	Register Mnemonic	Register Name
F2h	MADR	I2C Address Register
F4h	MBCR	I2C Control Register
F6h	MBSR	I2C Status Register
F8h	MBDR	I2C Data I/O Register
FAh	MDIVCNT	I2C Divide Count Register
FCh	MHOLDCNT	I2C Hold Count Register

**Figure 15-1 Synchronous Serial Interface Register Summary**

### 15.1 I2C Address Register (MADR, Offset F2h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	SADDR[7:1]							0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
7-1	Slave Address	Address Used by the I2C controller when in Slave mode.

Figure 15.1 I2C Serial Interface Address Register

### 15.2 I2C Control Register (MBCR, Offset F4h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	MEN	MIEN	MSTA	MTX	TXAK	RPTSTA	0	0

Reset value : 0000\_0000\_0000\_0000b

Bit	Name	Function
7	I2C Controller Enable	This bit must be set before any other MBCR bits have any effect.
6	Interrupt Enable	'1' enables interrupts. An interrupt occurs if MIF bit in the status register is also set. '0' disable interrupts but does not clear any currently pending interrupts.
5	Master/Slave Mode Select	When changed from '0' to '1', the I2C controller generates a START condition in Master mode. When this bit is cleared, a STOP condition is generated and the I2C controller switches to Slave mode. If this bit is cleared, however, because arbitration for the bus has been lost, a STOP condition is not generated.
4	Transmit/Receive Mode Select	This bit selects the direction of Master/Slave Transfers. - '1' selects an I2C transfer - '0' selects an I2C receive
3	Transmit Acknowledge Enable	This bit specifies the value driven onto the SDA line during acknowledge cycles for both Master and Slave receivers - '1' : no acknowledge is generated - '0' : acknowledge generated Since Master receivers indicate the end of data reception by not acknowledging the last byte of the transfer, this bit is the means for the IMS16B microcontroller to end a Master receiver transfer.
2	Repeated Start	Writing a '1' to this bit generates a repeated START condition on the bus if the I2C controller is the current bus Master. This bit is always read as '0'. Attempting a repeated START at the wrong time if another Master owns the bus results in a loss of arbitration.

Figure 15.2 I2C Serial Interface Control Register

### 15.3 I2C Status Register (MBSR, Offset F6h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	MCF	MAAS	MBB	MAL	0	SRW	MIF	RXAK

Reset value : 0000\_0000\_0000\_000xb

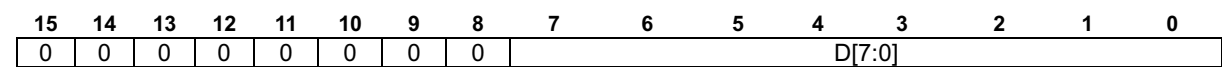
Bit	Name	Function
7	Data Transferring Bit	While on byte of data is being transferred, this bit is cleared. It is set by the falling edge of the ninth clock of a byte transfer. - '1' transfer is complete - '0' transfer in progress
6	Addressed as Slave Bit	When the address on the I2C bus matches the Slave address in the MADR Register, the I2C controller is being addressed as a Slave and switches to Slave mode.
5	Bus Busy Bit	This bit indicates the status of the I2C bus. This bit is set when a START condition is detected and cleared when a STOP condition is detected. - '1' indicates the bus is busy - '0' indicates the bus is idle
4	Arbitration Lost Bit	This bit is set by hardware when arbitration for the I2C bus is lost. The software writing a '0' to this bit must clear this bit.
2	Read/Write from/to Slave	When the I2C controller has been addressed as a Slave (MAAS is set), this bit indicates the value of the read/write bit sent by the Master. This bit is only valid when a complete transfer has occurred and no other transfers have been initiated. - '1' indicates Master reading from Slave - '0' indicates Master writing to Slave
1	Interrupt Bit	This bit is set when an interrupt is pending, which causes a processor interrupt request if MIEN is set. The software writing a "0" to this bit in the interrupt service routine must clear this bit.
0	Received Acknowledge Bit	This bit reflects the value of the SDA signal during the acknowledge cycle of the transfer. - '1' indicates that no acknowledge was received - '0' indicates that an acknowledge was received

Figure 15.3 I2C Serial Interface Status Register

### 15.4 I2C Data Register (MBDR, Offset F8h)

This register contains data to/from the I2C bus. Physically, two byte-wide registers at the same address implement this register, one for the I2C transmits data and one for the I2C received data.

In transmit mode, data written into this register is output on the I2C bus, in receive mode, this register contains the data received from the I2C bus. In receive mode, the received I2C data is placed in this register after each complete transfer. Note that in receive mode, it is assumed that the programmer will be able to read this register during the next I2C transfer.



**Reset value : 0000\_0000\_0000\_0000b**

Bit	Name	Function
7-0	I2C Data	I2C Data Register

**Figure 15.4 I2C Serial Interface Data Register**

### 15.5 I2C SCL Setting Register (MDIVCNT, Offset FAh)

This register contains dividing value for generating SCL signal. Internal clock of IMS16B is used for generating SCL period. Core clock period is the same as clkouta or clkoutb. The equation is as follows:

$$\text{SCL period} = (\text{clkouta}) / (n + 6) \quad , \text{where } n \text{ is integer.}$$

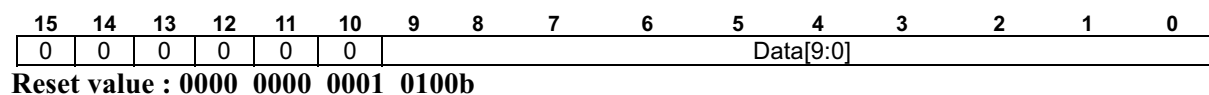
For example, If IMS16B core clock speed is 12.5Mhz, and if n is 20 then

$$\text{SCL period} = 12.5\text{Mhz} / (20 + 6) = 480\text{Khz}$$

That is, If 480Khz SCL period is necessary, programmer must write 10100b(20<sub>dec</sub>) to I2C SCL Setting register.

If IMS16B is operating at 50Mhz core clock speed, and if programmer needs 100Khz SCL signal, dividing value is

$$100\text{Khz} = 50\text{Mhz} / (n+6) \Rightarrow 100 * (n+6) = 50\text{K} \Rightarrow n+6 = 500 \Rightarrow n = 494(111101110\text{b}).$$



Bit	Name	Function
7-0	MDIVCNT	SCL divide count value

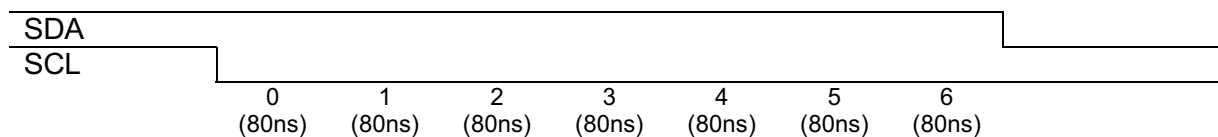
**Figure 15.5 I2C SCL Setting Register**

### 15.6 I2C Hold Time Setting Register (MHOLDCNT, Offset FCh)

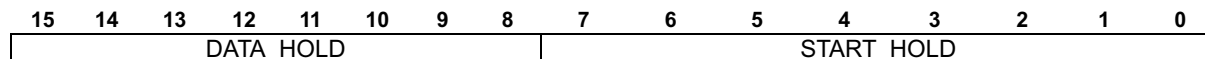
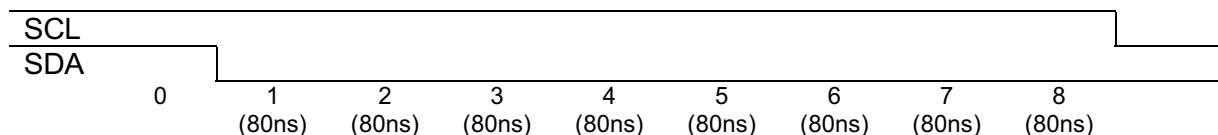
This register contains hold time values for I2C controller. SDA data hold time and SCL START hold time is programmable. Each hold time equations are as follows:

Data hold time = ( n + 1),  
 START hold time = ( n + 1)      where n is integer.

For example, If period of clkouta is 80 ns, and n = 6, then Data hold time is 560ns.



For example, If period of clkouta is 80 ns, and n = 7, then START hold time is 640ns.



Reset value : 0000\_0110\_0000\_1000b

Bit	Name	Function
15-8	DATA HOLD	Data hold period
7-0	START HOLD	START hold period

Figure 15.6 I2C Hold Time Setting Register

16

SDRAM Control Unit

Synchronous DRAMs (SDRAMs) provide a significant improvement in bandwidth performance over traditional asynchronous DRAMs such as "FPM" (Fast Page Mode) and "EDO" (Extended Data Out). Synchronous DRAMs latch input address, data, and control signals on the clock rising edge, freeing the controller from having to drive address and control for the entire read or write transaction. After a preset number of clock cycles, the data is available on the output latches of the SDRAM for a READ, or can be written into its memory for a WRITE. Figure 16 shows MICRON's 16Mbit SDRAM functional block diagram.

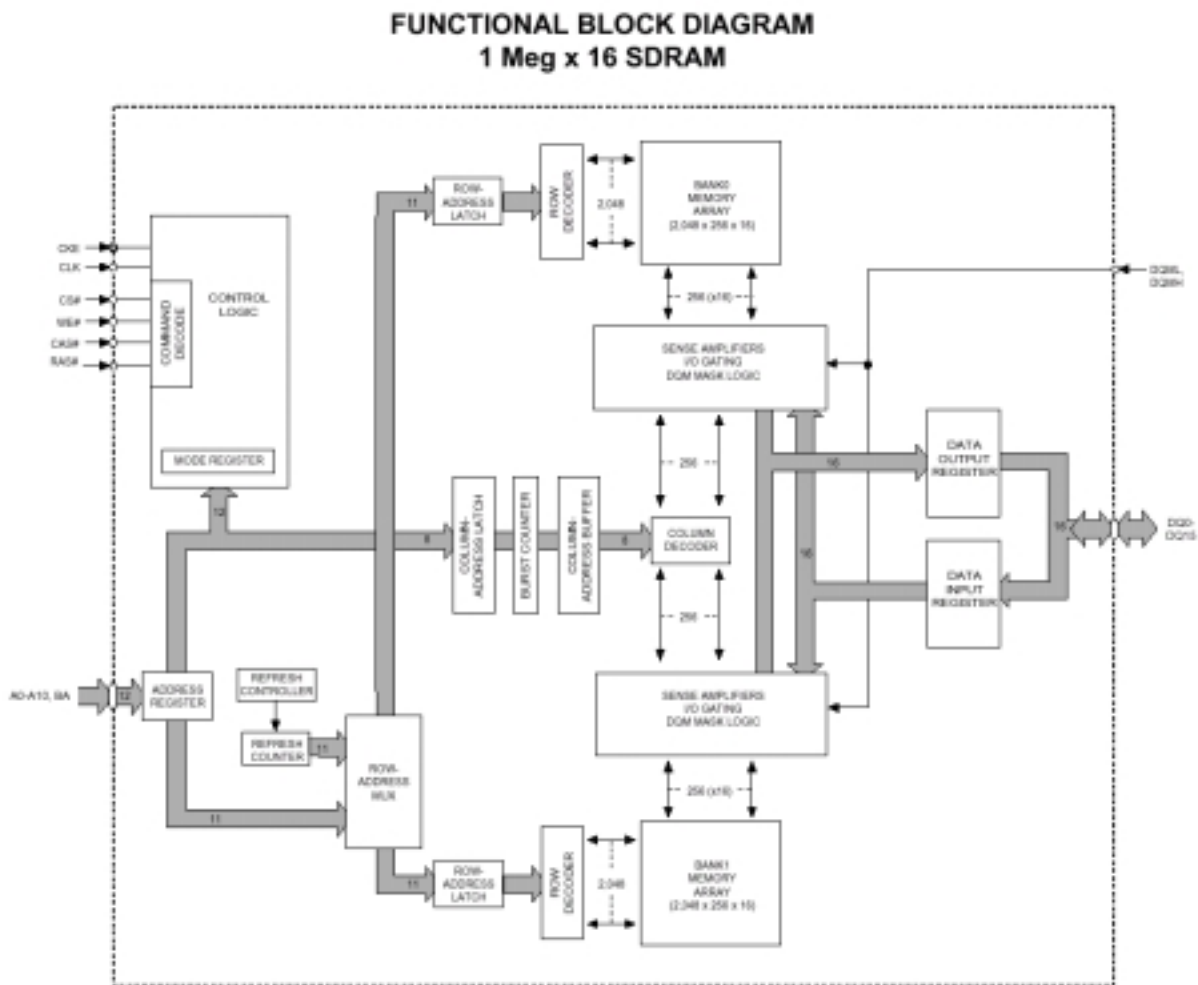


Figure 16 MICRON's 16Mbit SDRAM Block Diagram

In general, SDRAMs are multi-bank DRAM arrays that operate at 3.3V and include a command-driven synchronous interface (all signals are registered on the positive edge of the clock signal). For example, Figure 16 shows a dual 512K x 16 array architecture. Each of the two 512K x 16bit banks is organized as 2,048 rows by 256 columns by 16 bits. Read and write accesses to the SDRAM are burst oriented; accesses start at a selected location and continue for a programmed number of locations in a

programmed sequence.

Accesses begin with the registration of the active low command, followed by a read or write command. The address bits registered coincident with the active low commands are used to select the intended bank and row. The SDRAM must be initialized according to manufacturer specifications. Initialization usually consists of a sequence of precharge-all-banks, auto-refresh, and mode-register-set commands.

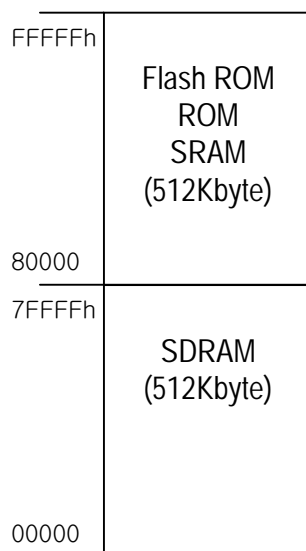
The majority of SDRAM employs a simple set of signal combinations called commands to carry out the basic IO. Most of these commands are one clock cycle in duration, and are clocked by the SDRAM on the rising edge. A proper setup and hold times must be observed. A sequence of these commands comprises the primitive operation of read, write and refresh. Figure 16 illustrates a typical set of command used by most SDRAMs.

The 1M x 16 SDRAM is organized as 512K x 16 x 2 banks. The bank selection is done by a pin called BA (bank address). The address organization of these SDRAMs are 11 rows by 8 columns. During the row address strobing, A[10:0] provides the row to select and during the column strobing, A[7:0] provides the column to select (A[10:8] should be set to logic low).

The IMS16B microcontroller has 20bit address bus. That is, it can only access 1M byte memory area. Currently SDRAM manufactures are making SDRAMs ranging from 16Mbits to 512Mbits. Due to the lack of address resources, IMS16B can use 512Kbyte memory area only. It means that if programmer has 16Mbit SDRAM, only bank 0 or bank 1 can be used for IMS16B memory access.

Note that In IMS16B, there is no bank selection pin. Bank selection pin in SDRAM must be tied to low to select bank 0. In using microcontrollers such as Am186 or 80C186, 512kbyte SDRAM size can be big enough in general use.

The IMS16B has SDRAM enable register. So, It can be disabled when power is critical or any other device is necessary for SDRAM area. SDRAM Controller is enabled at power-on reset. Figure 16-1 shows IMS16B Memory map with respect to SDRAM area. **SDRAM area can be configured through LMCS register.**



**Figure 16-1 IMS16B Memory Map with respect to SDRAM**

## 16.1 OVERVIEW OF SDRAM COMMANDS

### ■ Mode register set command

This command is used to program the SDRAM's mode register. The mode register controls the operation of the SDRAM, including the CAS latency, burst type, burst length, test mode and other vendor specific options. Most SDRAMs do not initialize the mode register upon power up, thus it is critical this register be initialized prior to the normal use. The SDRAM controller initializes the mode register following every system reset with a default value. During the mode register programming, the SDRAM receives the data from A[10:0] and BA, rather than from the data bus.

Most SDRAMs follows the bit field definition illustrated below, but make sure to consult the specific SDRAM vendor specs as they might contain subtle differences.

Figure 16.1 shows mode register definition for SDRAMs.

<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
res	WBL		TM		CAS Latency			BT	Burst Length		

Bit	Name	Function																								
9	Write Burst Length	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>A9</th> <th>Length</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Burst</td> </tr> <tr> <td>1</td> <td>Single bit</td> </tr> </tbody> </table>	A9	Length	0	Burst	1	Single bit																		
A9	Length																									
0	Burst																									
1	Single bit																									
8-7	Test Mode	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>A[8:7]</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Mode register set</td> </tr> <tr> <td>1</td> <td>res</td> </tr> <tr> <td>2</td> <td>res</td> </tr> <tr> <td>4</td> <td>res</td> </tr> </tbody> </table>	A[8:7]	Type	0	Mode register set	1	res	2	res	4	res														
A[8:7]	Type																									
0	Mode register set																									
1	res																									
2	res																									
4	res																									
6-4	CAS Latency	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>A[6:4]</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>res</td> </tr> <tr> <td>1</td> <td>res</td> </tr> <tr> <td>2</td> <td>2</td> </tr> <tr> <td>4</td> <td>3</td> </tr> <tr> <td>5</td> <td>res</td> </tr> <tr> <td>6</td> <td>res</td> </tr> <tr> <td>7</td> <td>res</td> </tr> </tbody> </table>	A[6:4]	Type	0	res	1	res	2	2	4	3	5	res	6	res	7	res								
A[6:4]	Type																									
0	res																									
1	res																									
2	2																									
4	3																									
5	res																									
6	res																									
7	res																									
3	Burst Type	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>A3</th> <th>Length</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>sequential</td> </tr> <tr> <td>1</td> <td>interleave</td> </tr> </tbody> </table>	A3	Length	0	sequential	1	interleave																		
A3	Length																									
0	sequential																									
1	interleave																									
2-0	Burst Length	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>A[2:0]</th> <th>BT=0</th> <th>BT=1</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>2</td> <td>2</td> </tr> <tr> <td>2</td> <td>4</td> <td>4</td> </tr> <tr> <td>4</td> <td>8</td> <td>8</td> </tr> <tr> <td>5</td> <td>res</td> <td>res</td> </tr> <tr> <td>6</td> <td>res</td> <td>res</td> </tr> <tr> <td>7</td> <td>page</td> <td>res</td> </tr> </tbody> </table>	A[2:0]	BT=0	BT=1	0	1	1	1	2	2	2	4	4	4	8	8	5	res	res	6	res	res	7	page	res
A[2:0]	BT=0	BT=1																								
0	1	1																								
1	2	2																								
2	4	4																								
4	8	8																								
5	res	res																								
6	res	res																								
7	page	res																								

Figure 16.1 Mode register definition for general SDRAMs

■ Row address strobe and bank active command

This command is used to select the bank and the row where the data access is to take place. The BA input selects the bank, while A[10:0] provides the row address.

■ Precharge

The precharge command is used to begin the precharge operation to the selected bank(s). When A[10] is high, both banks are activated, while when A[10] is low, the bank selected by BA is activated.

■ Column address and write command

This command is used to select the column of the selected bank where the write is to take place. It is important that the bank selected at the row address strobe be selected again. The address bus A[7:0] selects the column. A[10:8] should be logic low.

■ Column address and read command

This command is used to select the column of the selected bank where the read is to take place. The address bus A[7:0] selects the column. A[10:8] should be logic low.

■ Auto refresh command

This command is used to start the internal refresh cycle. An internal row counter is adjusted to point to the next row.

Figure 16.1-1 shows timing diagrams of basic commands for SDRAM control.

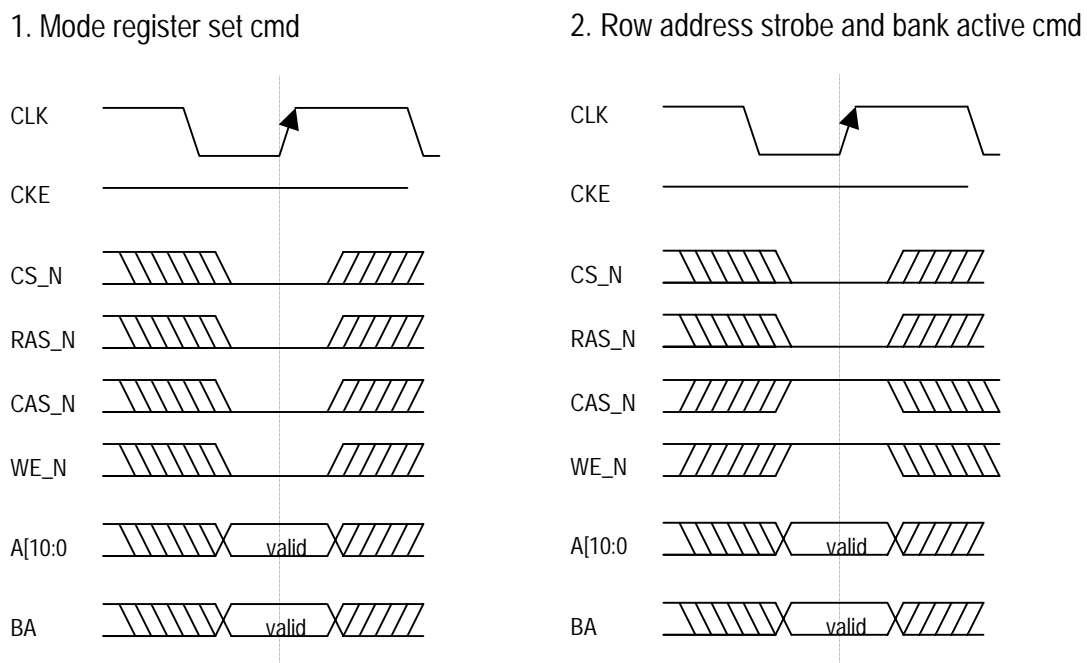
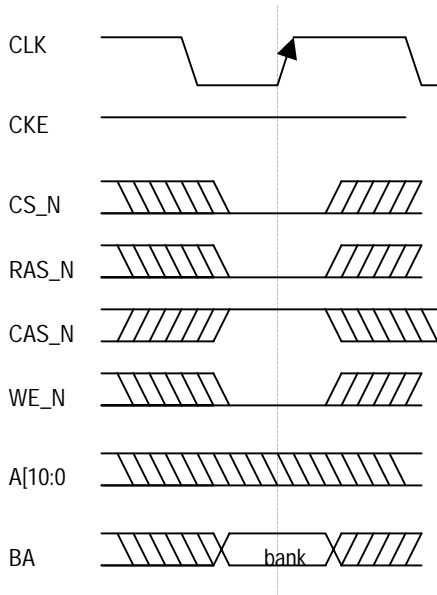
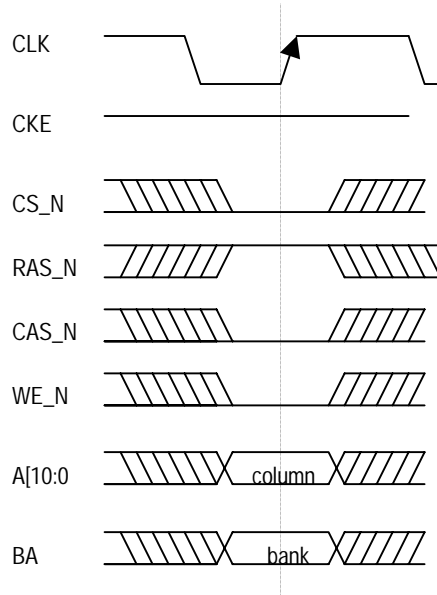


Figure 16.1-1 Timing Diagrams of basic commands for SDRAM control

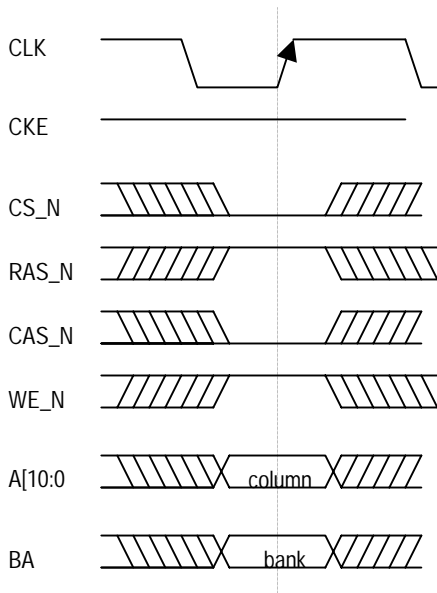
3. Precharge cmd



4. Column address and write cmd



5. Column address and read cmd



6. Autorefresh cmd

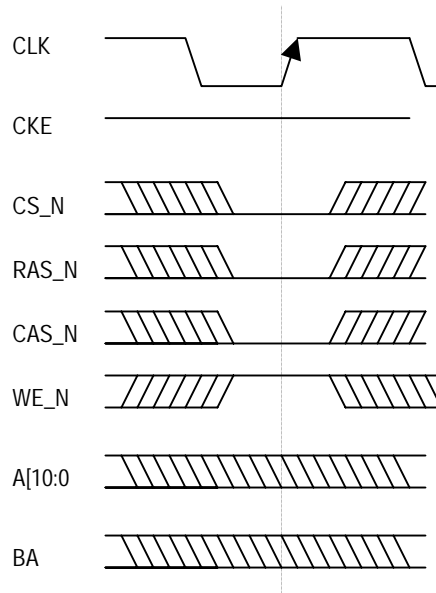


Figure 16.1-1 Timing Diagrams of basic commands for SDRAM control(Contd)

## 16.2 Basic Operation

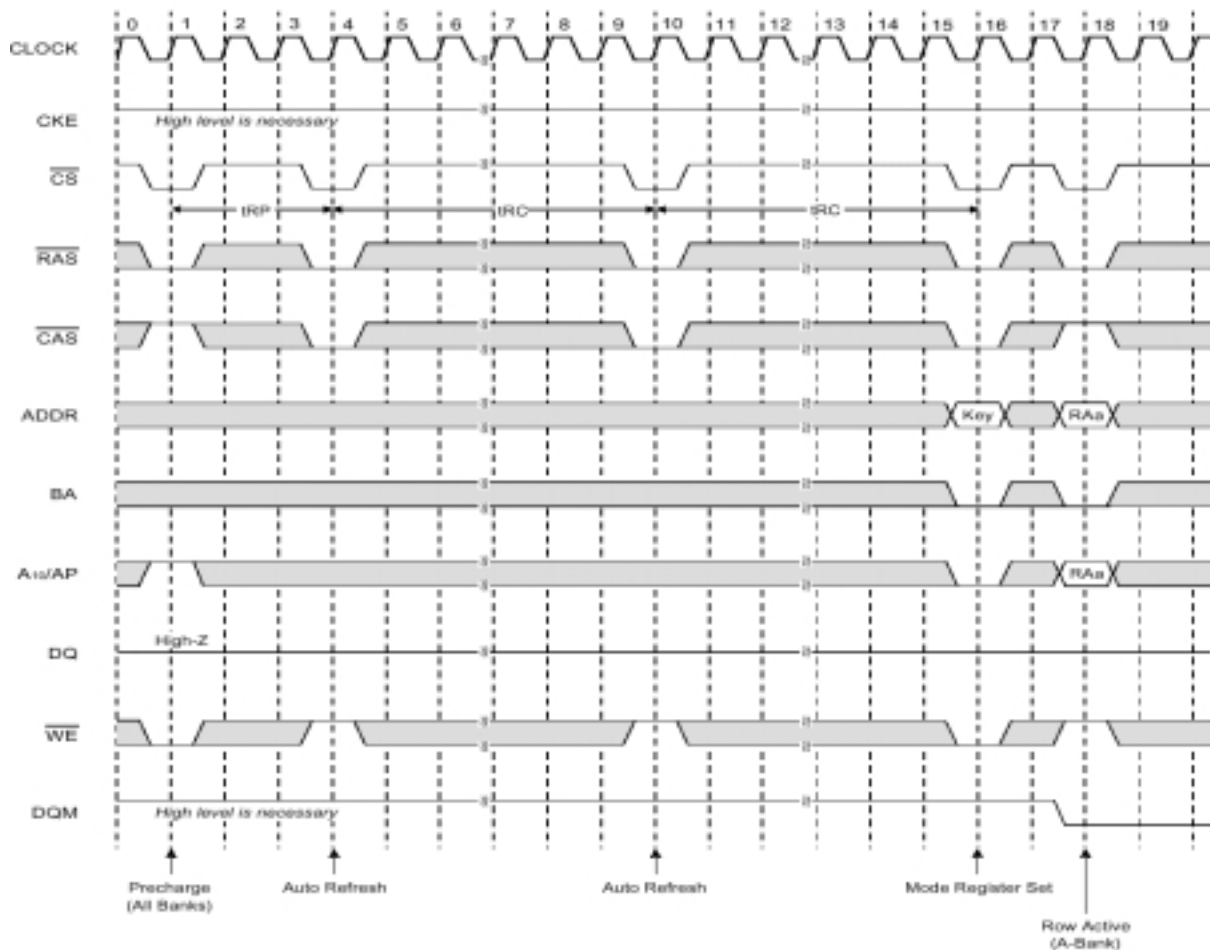
The following section describes the basic operations of single word write, single word read and refresh. Burst write and burst read are not presently supported. Upon power-up, the SDRAM's mode register must be initialized for proper operation. This is due to the fact that most SDRAMs do not initialize the mode register following a reset. However, before the register can be initialized, the SDRAM must be subjected to an initialization sequence.

Notice that following power up (or reset), a repeated number of auto refresh command is necessary. The power-up sequence is also highly dependent on the SDRAM maker, so consult the specification. In general, the sequence consists of the following:

1. After a stable power (Vcc) has been reached, the SDRAM should see a stable clock for about 200uS. During this time, no valid command should be issued.
2. Both banks must be precharged (precharge command)
3. One or more auto-refresh commands must be issued.
4. The mode register can now be initialized.

IMS16B SDRAM controller generates control signals for above power-up sequences for SDRAM automatically at power-on reset.

**Power Up Sequence**

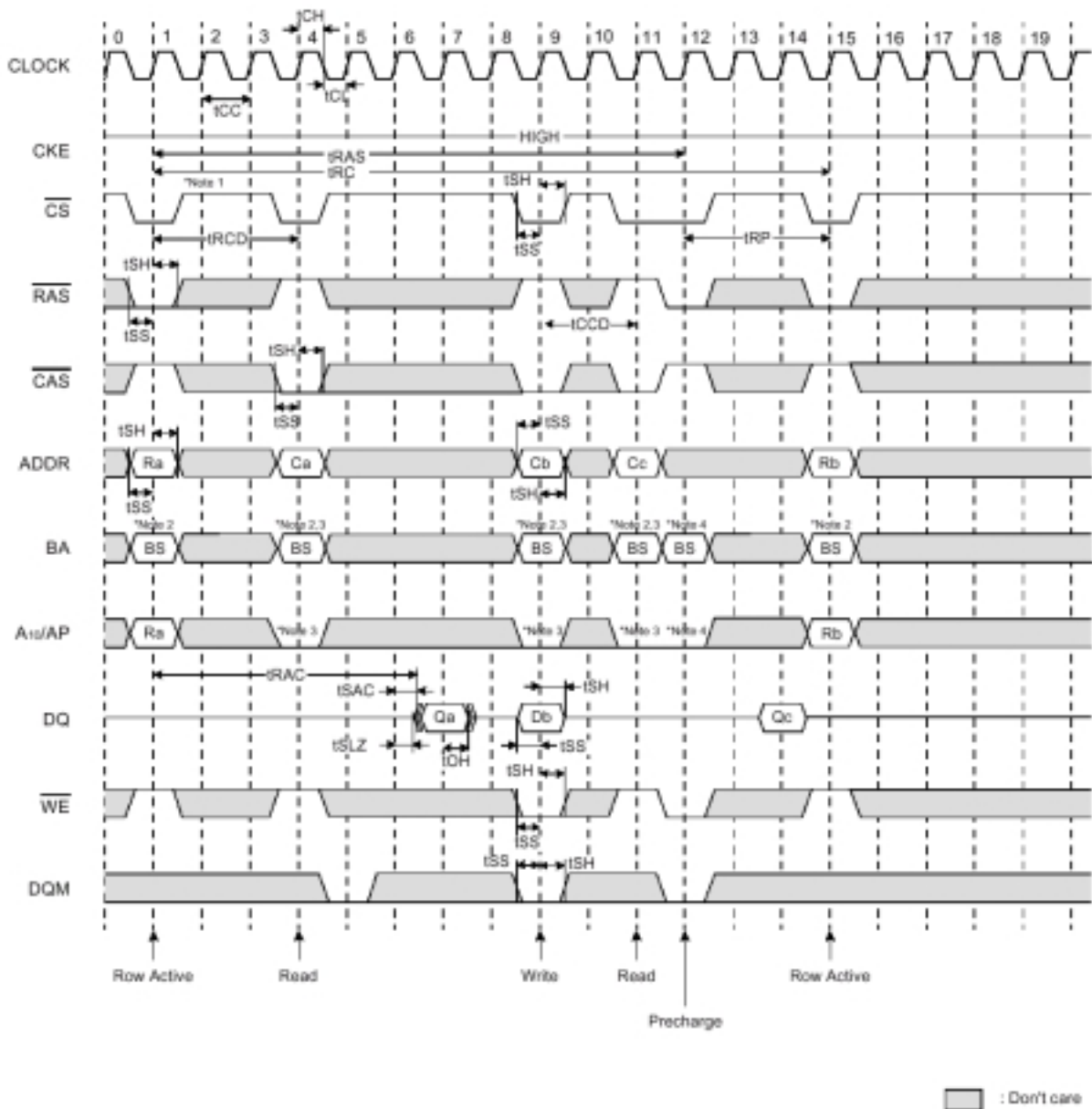


**Figure 16.2 Timing Diagrams of Power-Up Sequence (Samsung SDRAM)**

The basic read and write operation in Figure 16.2-1 consist of the following command sequences:

1. Row address strobe and bank active command,
2. Wait to meet t<sub>RCD</sub>
3. Column address and read command
4. Wait to meet CAS latency
5. Column address and write command
6. Column address and read command
7. Precharge all bank command
8. Wait to meet t<sub>RP</sub>
9. Wait to meet t<sub>RC</sub> before a new row activate command

**Single Bit Read-Write-Read Cycle(Same Page) @CAS Latency=3, Burst Length=1**



**Figure 16.2-1 Timing Diagrams of read and write Sequence (Samsung SDRAM)**

Once the SDRAM receives the column address strobe command, it makes the data available on its bus CAS latency cycles later. The CAS latency is programmed through the mode register. In theory this value can be programmed from 2 to 7 cycles, but most SDRAM vendors limit the choices to 2 or 3 cycles. After the data has been read, the SDRAM controller issues a precharge-all command to ready for the next data access.

The write operation inherently takes fewer cycles than the write because there are not CAS latencies involved. The SDRAM latched in the data on the rising edge of column strobe command.

The refresh operation consists of initiating the internal auto-refresh cycle of the SDRAM, and consists of the following command sequence:

1. N number of autorefresh commands
2. Delay to meet tRC before issuing another command

N represents the number of rows to be refreshed. Following the autorefresh command, the SDRAM cannot accept any new commands until tRC later. This parameter varies by SDRAM vendor.

The number of autorefresh cycles, N, to be issued depends on the type of refresh desired. As previously mentioned, in a typical 16Mbit SDRAM, there are 2048 rows per bank. And the typical refresh interval is 32mS. For an evenly distributed refresh type, this represents 1 refresh operation every 15.6uS(32ms/2048). Alternatively, a “burst” type of refresh can perform all 2048 refresh operations every 32mS. The latter approach is highly advantageous for very slow hosts. Thus N can take a value from 1 to 2048.

Note that the IMS16B SDRAM controller does not support self-refresh mode.

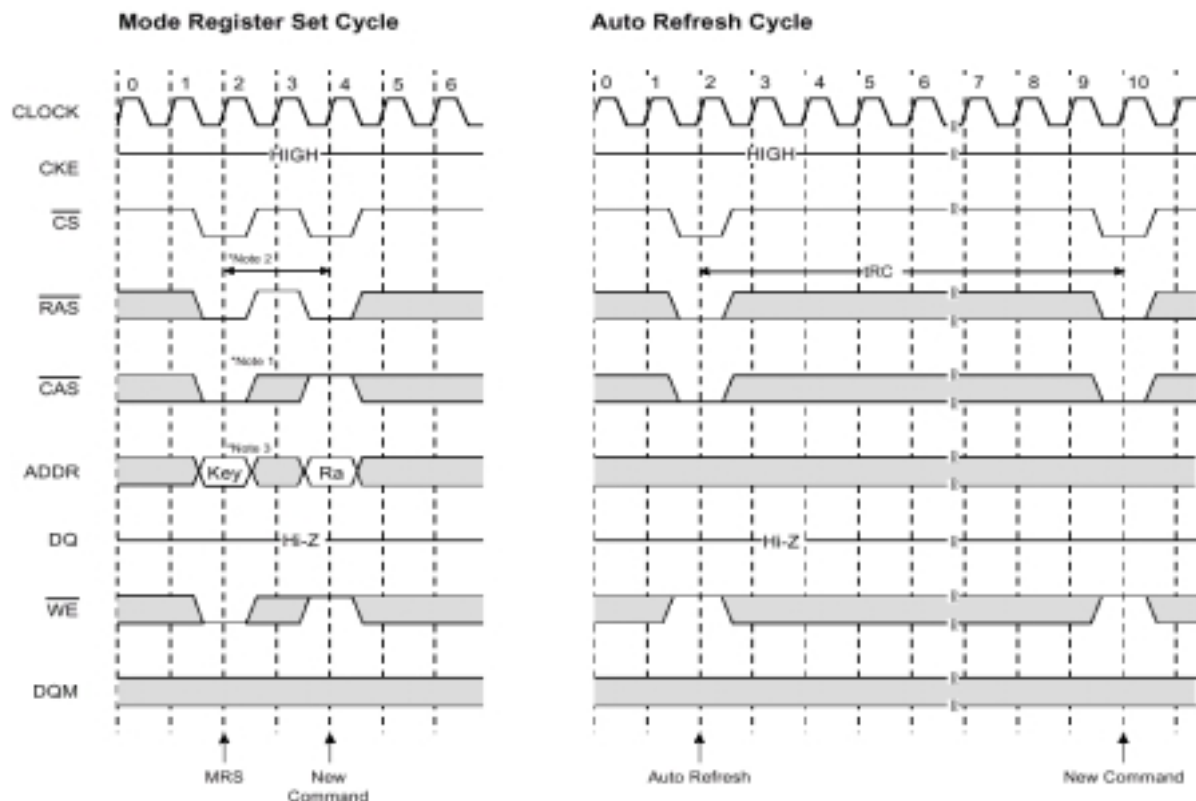


Figure 16.2-2 Timing Diagrams of mode register set and auto refresh cycle (Samsung SDRAM)

## 16.3 SDRAM CONTROL REGISTERS

The IMS16B microcontroller has 3 peripheral registers related to SDRAM controller. Only Writing is available to this registers. Figure 16.3 shows SDRAM Control Register Summary.

Offset	Register Mnemonic	Register Name
EAh	SDEN	SDRAM Enable Register (w)
ECh	SDDUTY	SDRAM Autorefresh Duty Cycle Register (w)
EEh	SDMODE	SDRAM Mode Set Register (w)

Figure 16.3 SDRAM Control Register Summary

## 16.4 SDRAM Enable Register (SDEN, Offset EAh)

When this bit is cleared, SDRAM Controller does not generate any control signals. This bit is set on reset.

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

Reset value : 0000\_0000\_0000\_0001b

Bit	Name	Function
0	SDRAM EN	This bit enables SDRAM Controller. This bit is set on power-on reset.

Figure 16.4 SDRAM Enable Register

***If external SRAM is only used, it recommends that EN bit in SDRAM enable register must be disabled. Because SDRAM access requires 7 external clock times, but SRAM access uses 4 cycle times without external ready.***

### 16.5 SDRAM Auto-refresh Duty Cycle Register (SDDUTY, Offset ECh)

Most of the storage cells of SDRAM need to be refreshed every 32ms to maintain data. An auto refresh cycle accomplishes refresh of a single row of storage cells. The internal counter increments automatically on every auto refresh cycle to refresh all the rows. An auto refresh command is issued by asserting low on CS, RAS and CAS with high on CKE and WE.

This register contains dividing value for generating autorefresh command. Internal clock of IMS16B is used for generating autorefresh command period. Core clock period is the same as CLKOUTA or CLKOUTB. The equation is as follows:

$$\text{Autorefresh Duty Cycle} = \text{SDRAM Refresh Time} / (\text{SDRAM Row Num} * \text{core clock})$$

For example, If IMS16B core clock speed is 50Mhz(20ns), and if SDRAM refresh time is 32ms, and SDRAM Row number is 2048 then

$$\text{Autorefresh Duty Cycle} = 32 * 10^{-3} / (2048 * 20 * 10^{-9}) = 781.25$$

That is, If auto-refresh commend is generated once every 781 core clock cycles, it refreshes 2048 rows in 32ms. In this case, programmer must write 1100001101b(781<sub>dec</sub>) to SDRAM Auto-refresh Duty Cycle Register.

Default value at power-on is 750<sub>dec</sub>(2EEh). It assumed that core clock speed is 50Mhz and refresh cycle is once every 15us for 2048 rows(2k/32ms).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	Data[10:0]										

Reset value : 0000\_0010\_1110\_1110b

Bit	Name	Function
10-0	SDDUTY	SDRAM Auto-refresh Duty Cycle Value.

Figure 16.5 SDRAM Auto-refresh Duty Cycle Register

### 16.6 SDRAM Mode Set Register (SDMODE, Offset EEh)

Figure 16.6 shows SDRAM mode set register. Power-on reset value is 0020h.

	10	9	8	7	6	5	4	3	2	1	0
	res	WBL	TM	CAS Latency			BT	Burst Length			

Bit	Name	Function		
9	Write Burst Length	<b>A9</b>	<b>Length</b>	
		0	Burst	
		1	Single bit	
8-7	Test Mode	<b>A[8:7]</b>	<b>Type</b>	
		0	Mode register set	
		1	res	
		2	res	
		4	res	
6-4	CAS Latency	<b>A[6:4]</b>	<b>Type</b>	
		0	res	
		1	res	
		2	2	
		4	3	
		5	res	
		6	res	
		7	res	
3	Burst Type	<b>A3</b>	<b>Length</b>	
		0	Sequential	
		1	Interleave	
2-0	Burst Length	<b>A[2:0]</b>	<b>BT=0</b>	<b>BT=1</b>
		0	1	1
		1	2	2
		2	4	4
		4	8	8
		5	res	res
		6	res	res
		7	Page	res

**Figure 16.6 SDRAM Mode Set Register**

## APPENDIX

---

- APPENDIX A. Electrical Characteristics**
- APPENDIX B. Package**
- APPENDIX C. Typical Applications**
  - C.1 Hardware RESET Circuit**
  - C.2 PLL Frequency Synthesizer Circuit**
  - C.3 Typical Memory Interface**
  - C.4 EVM Board Schematic**

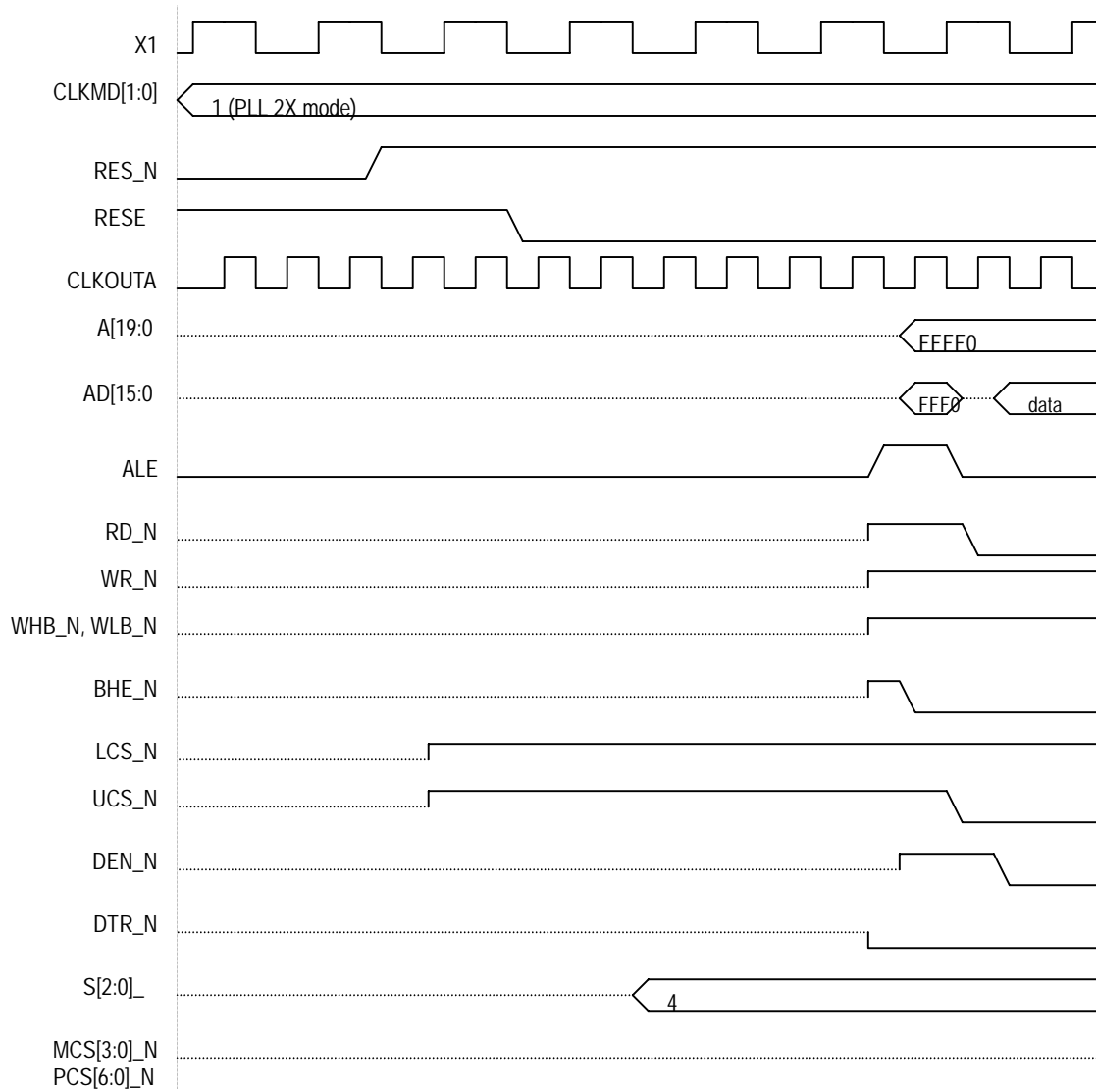
## A. Electrical Characteristics

### A.1 DC Electrical Characteristics

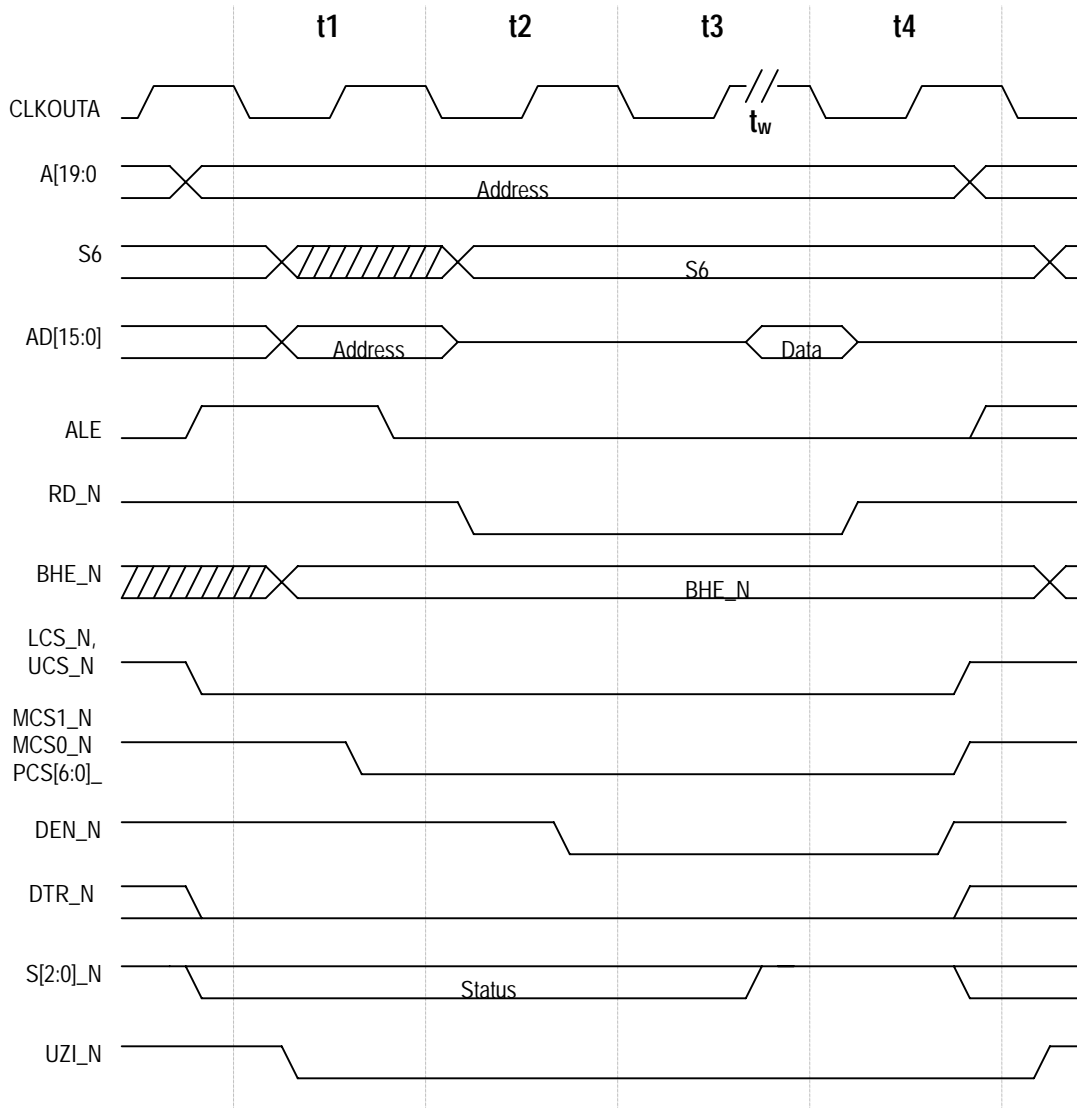
Parameters	Minimum	Maximum	Unit	Conditions
Power Supply	3.0	3.6	V	
ViL, low level input voltage CMOS input	-0.5	0.3 x VDD	V	Guaranteed input low voltage
ViH, high level input voltage CMOS Input	0.7 x VDD	VDD + 0.5	V	Guaranteed input high voltage
Junction temperature	0	100	°C	
Main Clock Frequency	-	60	Mhz	Guaranteed Operating Frequency

## A.2 AC Electrical Characteristics

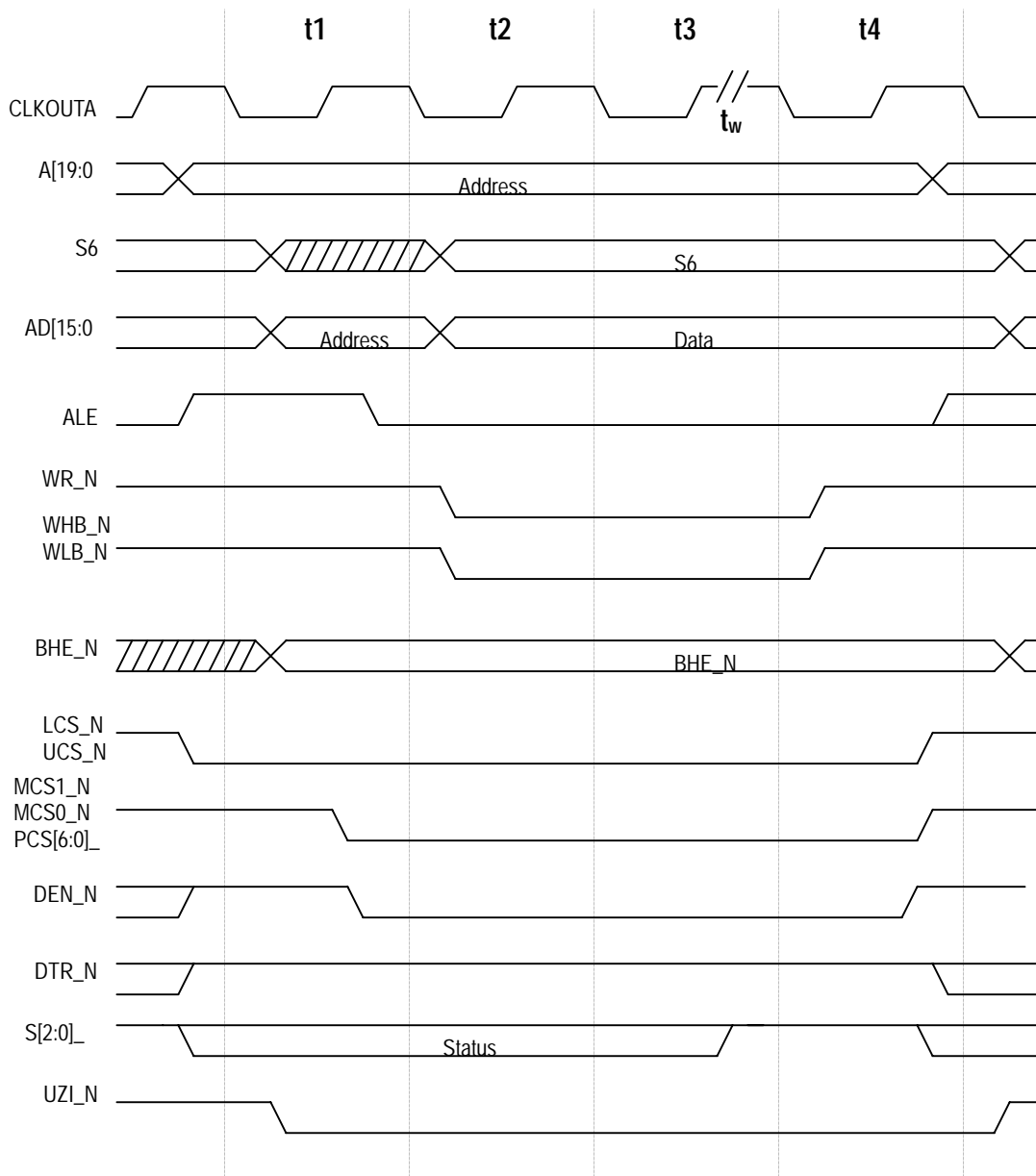
### A.2.1 Reset Waveforms



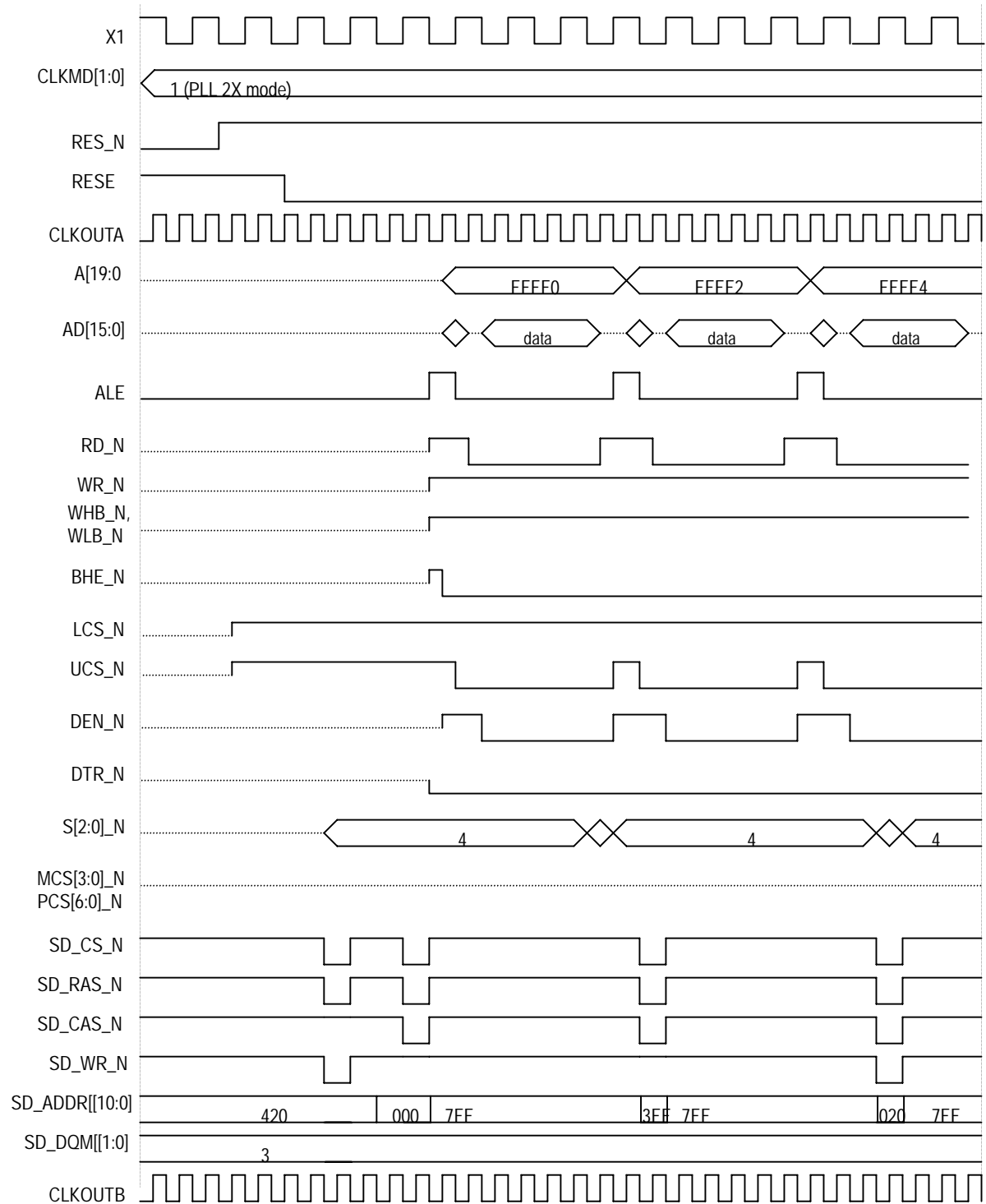
### A.2.2 Read Cycle Waveforms



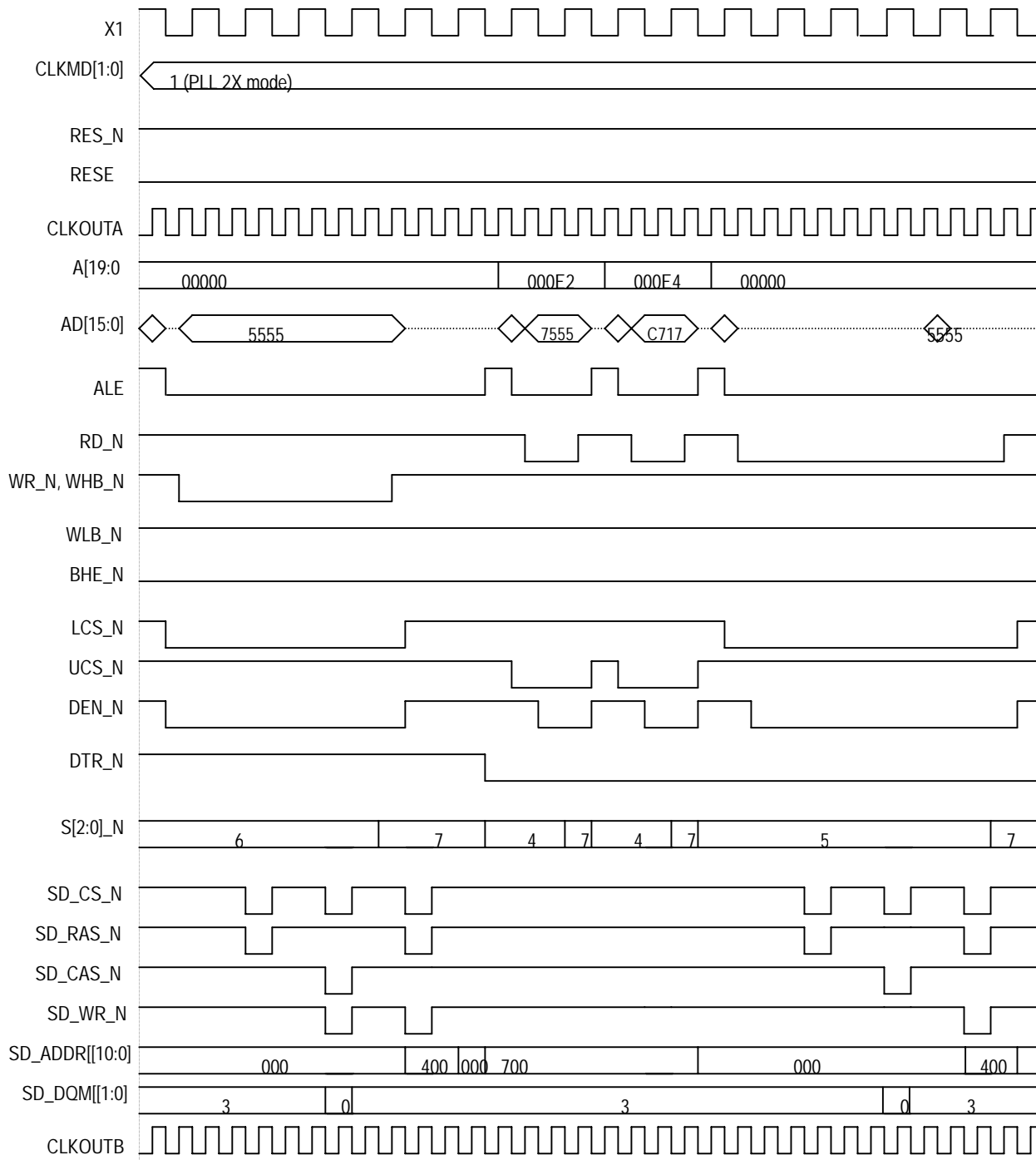
### A.2.3 Write Cycle Waveforms



### A.2.4 SDRAM Initial Waveforms



### A.2.5 SDRAM Read/Write Waveforms



## B. Package

The Plastic Quad Flatpack (PQFP) is a high-density, low-cost package for high leadcount applications. It uses a smaller lead-to-lead spacing than the PLCC and has gull-wing leads which permit better inspection of leads and solder joints. PQFPs are assembled with the latest technology of low stress die-attach material and molding compound.

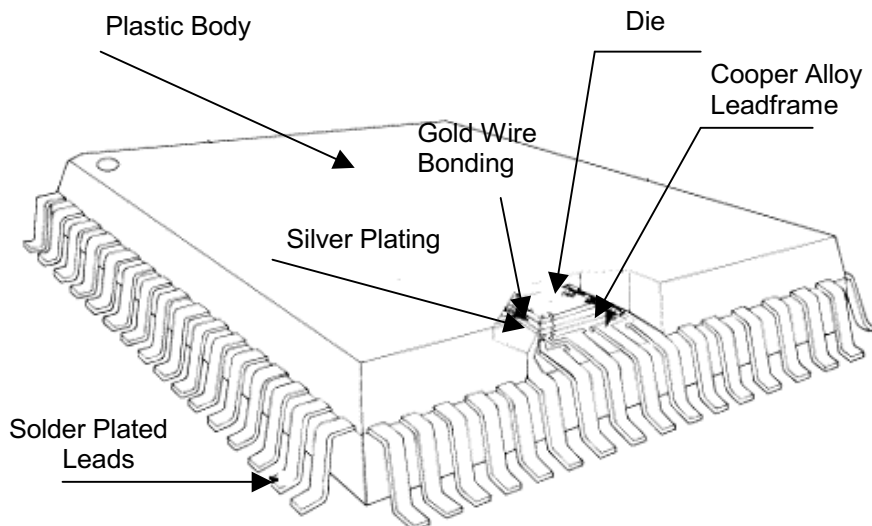


Figure B.1 Packaging

Dimensions in Millimeters

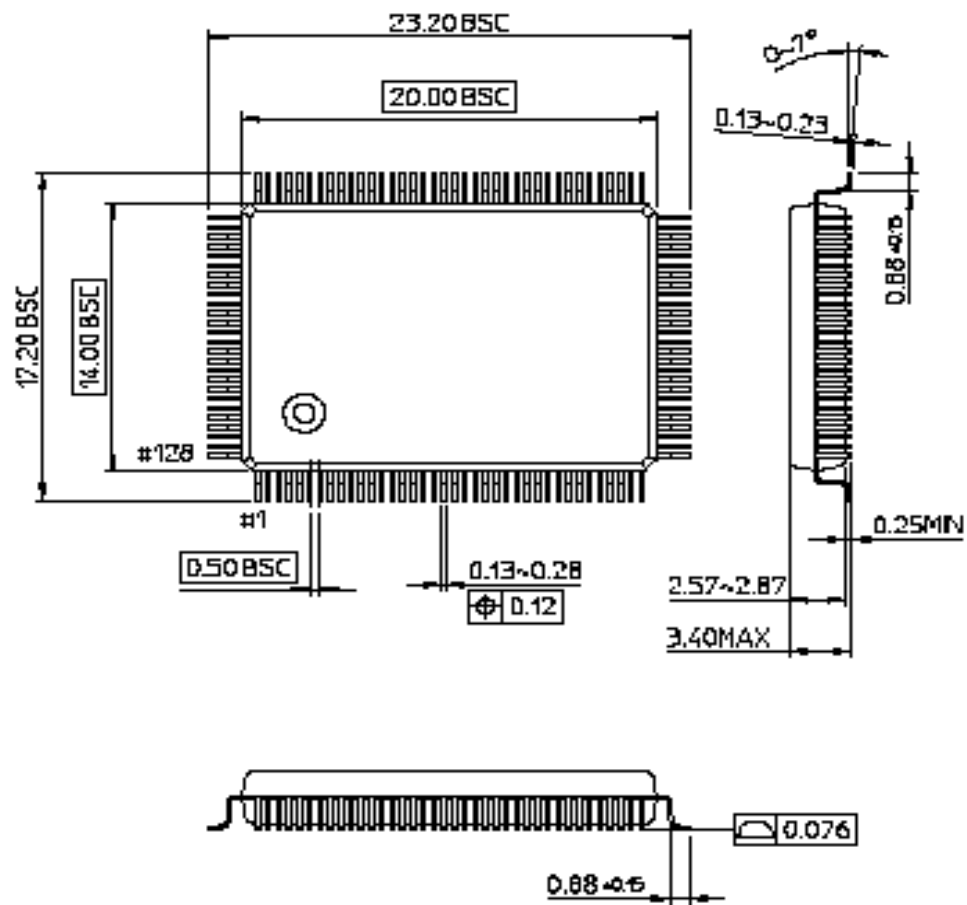


Figure B.2 Physical Dimension

## C. Typical Applications

This Page is left Blank Intentionally!!

### C.2 PLL Frequency Synthesizer Circuit

The IMS16B has a PLL frequency synthesizer on chip to provide frequency multiplication capabilities. The PLL circuit consists of a pre-divider, a PFD(Phase/Frequency Detector), a charge pump, a VCO (Voltage Controlled Oscillator), a post-scaler and a external loop filter.

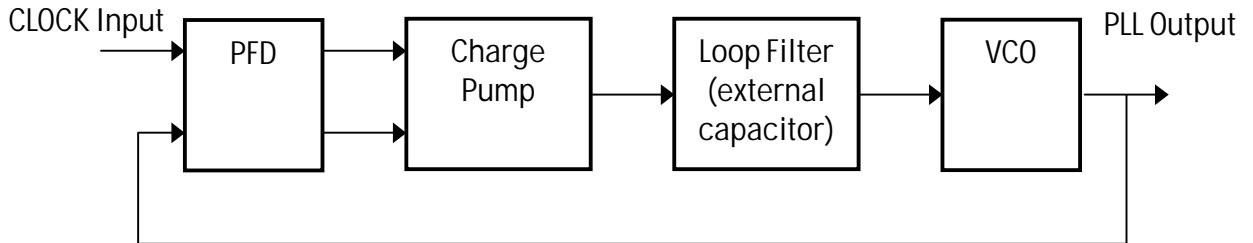


Figure C.2 Main Component of PLL Frequency Synthesizer

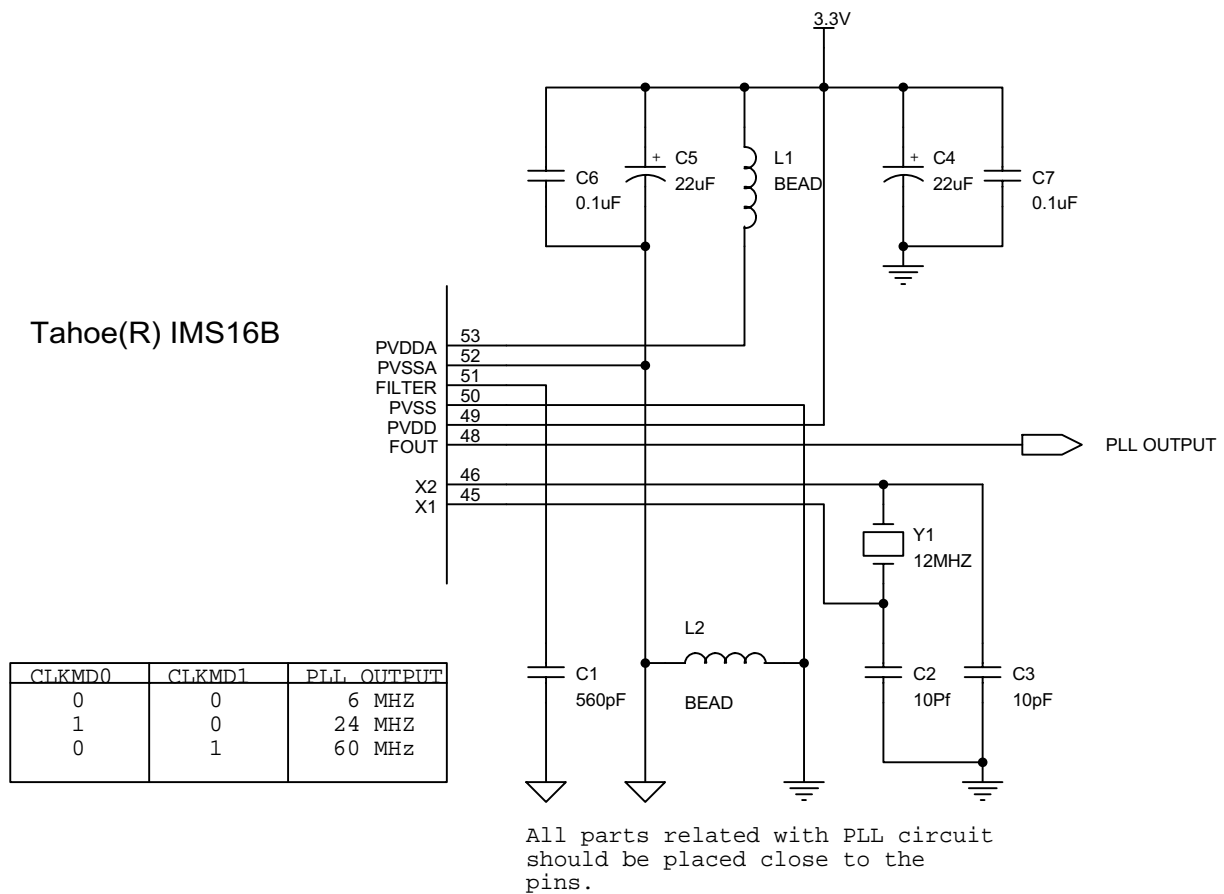


Figure C.3 Example circuit for PLL Frequency Synthesizer

Phase tolerance and jitter of the PLL circuit are independent of the input and PLL frequencies, those are affected by the noise frequency in the power line. Jitter increases when the noise level increases. For signal compatibility with the PLL circuit, a CMOS-level crystal or oscillator is recommended. The capacitor C2 and C3 are varied with the characteristics of crystal Y1. C4 and C5 de-coupling capacitors are 22uF electrolytic capacitors. 10uF capacitors would work well also. C6 and C7 de-coupling capacitors are 0.1uF ceramic capacitors. Loop filter C1 is 560pF ceramic capacitor should be placed as close as possible to the FILTER PIN (Pin 51 in ).

### C.3 Evaluation Board

Infinior MicroSystems offers a simple evaluation board with boot code for embedded network applications.

This evaluation board has two MAC controller interfaces and one serial port with 4Mbit Flash ROM and 2Mbit SRAM. It has one MAC controller with CAT-5 interface and another MAC controller and a 16Mbit SDRAM can be mounted as optional.

For software debugging using a popular ROM emulator which supports 16bit bus, it has 100 mil standard connector JP2.

When using a ROM emulator or booting from ROMs from JP2 connector, S2 should be configured properly.

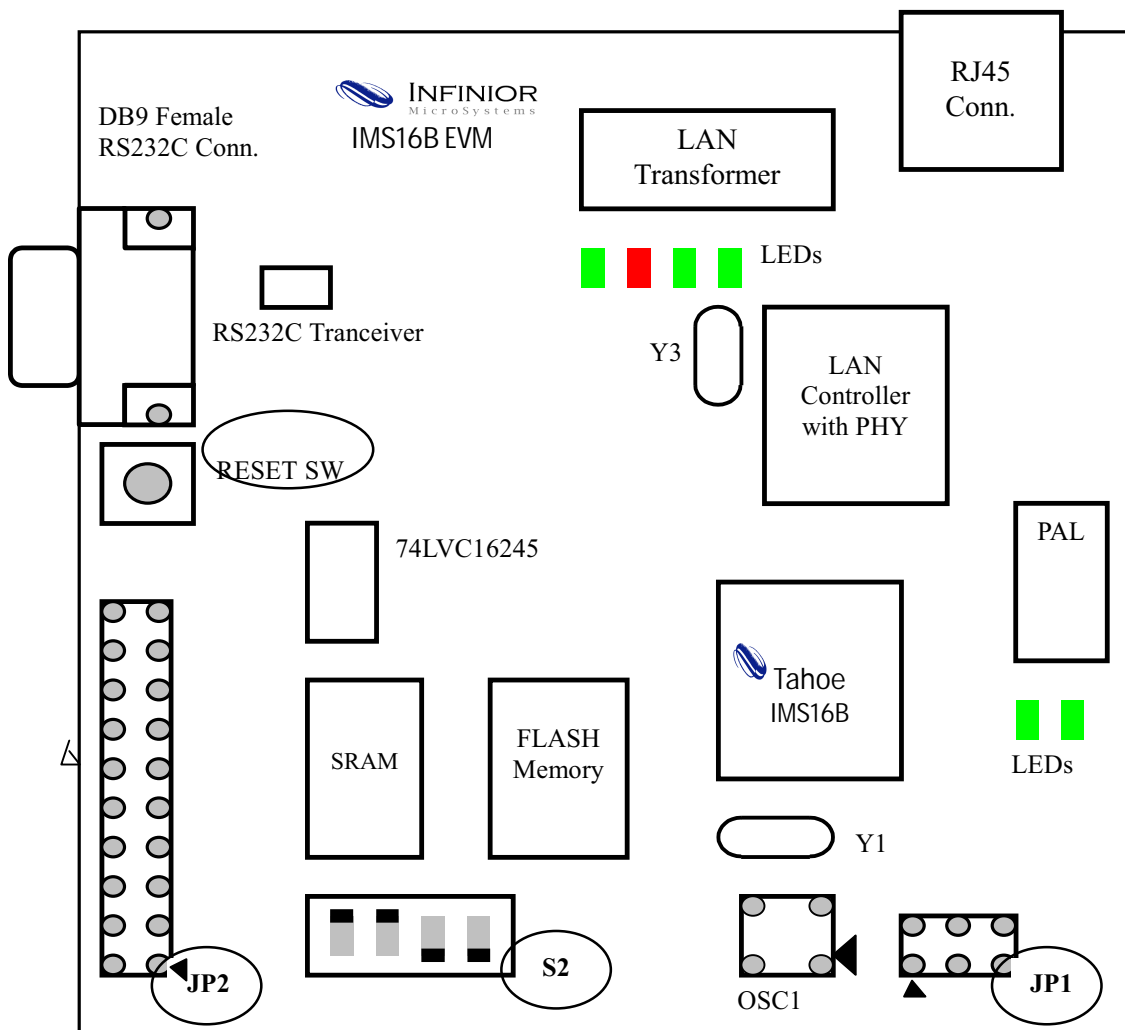


Figure C.4 Tahoe® IMS16B Evaluation Board Layout

The boot option by S2 is shown in Figure C-5.

1	2	3	4	Option
ON	ON	OFF	OFF	Boot from on-board Flash ROM and using on-board SRAM
OFF	OFF	ON	ON	Boot from boot device on JP2 and using a SRAM and/or other Memories on JP2
OFF	ON	ON	OFF	Boot from boot device on JP2 and using on-board SRAM

Figure C-5 S2 Configuration

Clock Mode can be configured externally by JP1. The configuration is listed in Figure C-6. In any clock mode, there should be only one clock device is on the board in Y1 or OSC1. By default, In the evaluation board has 12MHz crystal at Y1 position. If it is needed to adjust the clock frequency by experiment, Y1 has to be taken out and use of OSC1 connector to adjust oscillator clock frequencies.

1-3 Pin	3-5 Pin	2-4 Pin	4-6 Pin	Clock Mode		PLL Operation Mode
				CLKMD0	CLKMD1	
ON	OFF	ON	OFF	0	0	PLL Bypass Mode Internal clock is divided by 2 of input CLOCK
ON	OFF	OFF	ON	1	0	x 2 of input CLOCK
OFF	ON	ON	OFF	0	1	x 5 of input CLOCK

Figure C-6 JP1 Clock Mode Configuration

For using a ROM emulator, installing a daughter board is needed. In the daughter board, lower 8-bit ROM device should be located in IC reference number U8 and upper 8bit ROM device should be in U9 socket. Please refer to Figure C-7 to install the daughter board properly.

To use of this evaluation board in specific applications, there are a couple of software routines are available. For example, a Boot Loader, monitor program, RTOS, and TCP/IP routine can be supplied. For more information and technical support please contact us at sales@infinior.com.

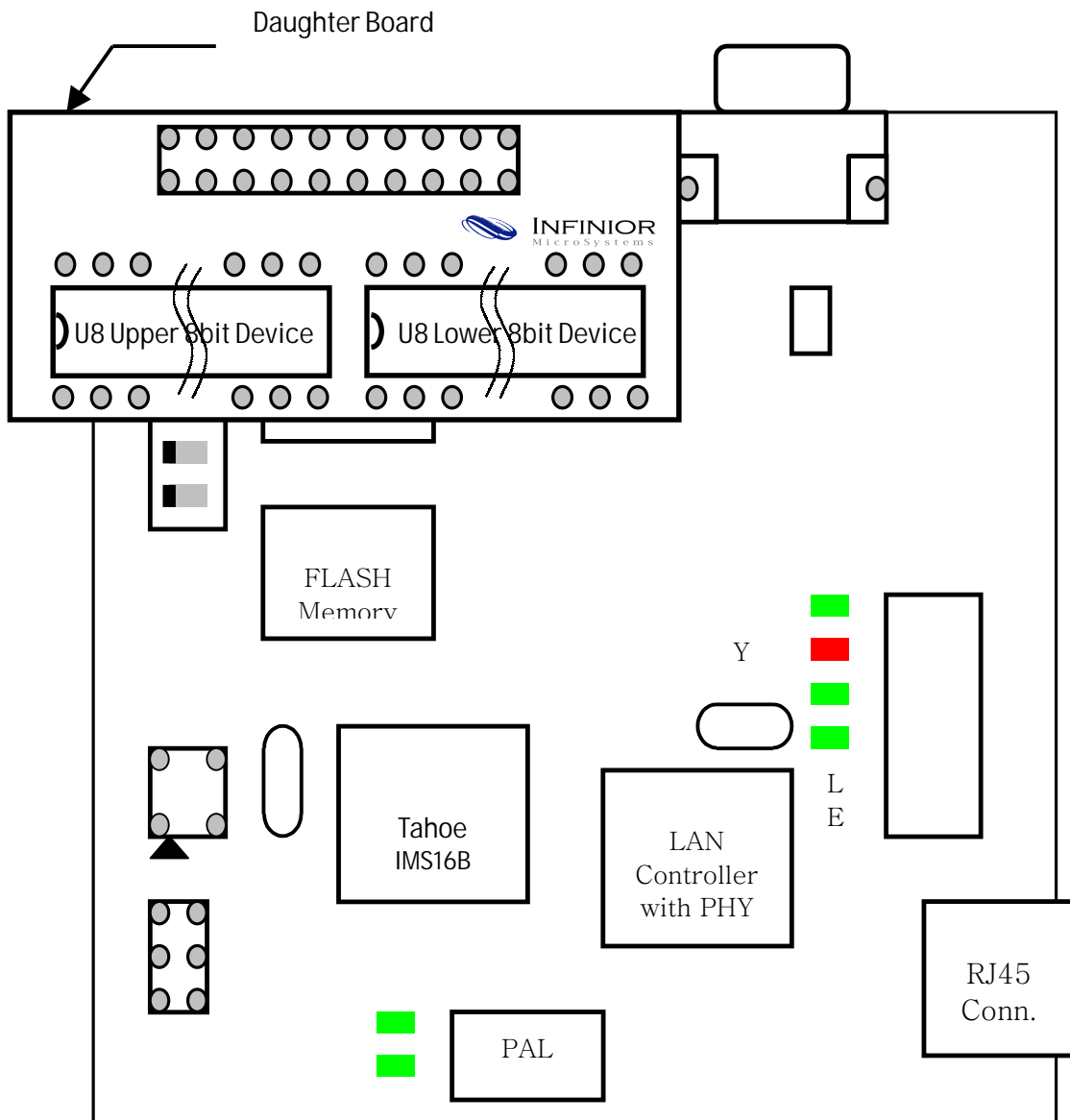


Figure C-7 Daughter Board Installation

### C.3-1 LAN Interface

IMS16B EVM board has two MAC controllers on board. In general, in most of applications only one MAC controller would be installed and the base address is started from 200h via 8bit-wide bus interface. Normally, PCS0 will be used for chip select for this MAC controller. Interrupt from this MAC controller is connected to INT0 of IMS16B MCU.