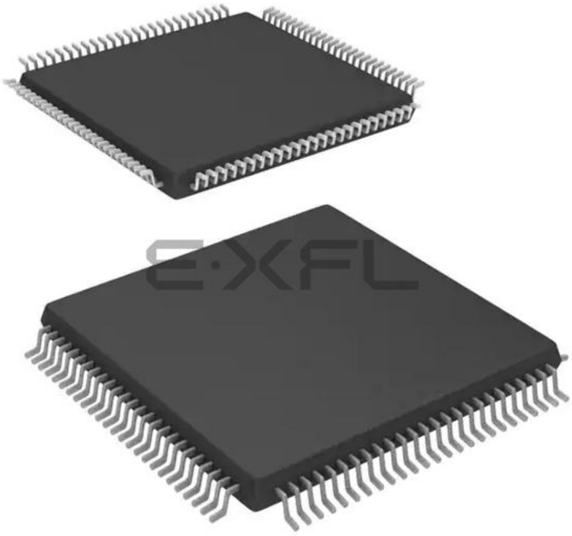
Analog Devices Inc. - IA186EM-PTQ100I-R-03 Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	-
Core Size	8/16-Bit
Speed	40MHz
Connectivity	UART/USART
Peripherals	DMA, PWM, WDT
Number of I/O	32
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	-
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-TQFP
Supplier Device Package	100-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/ia186em-ptq100i-r-03

2.1.1 IA186EM TQFP Package

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

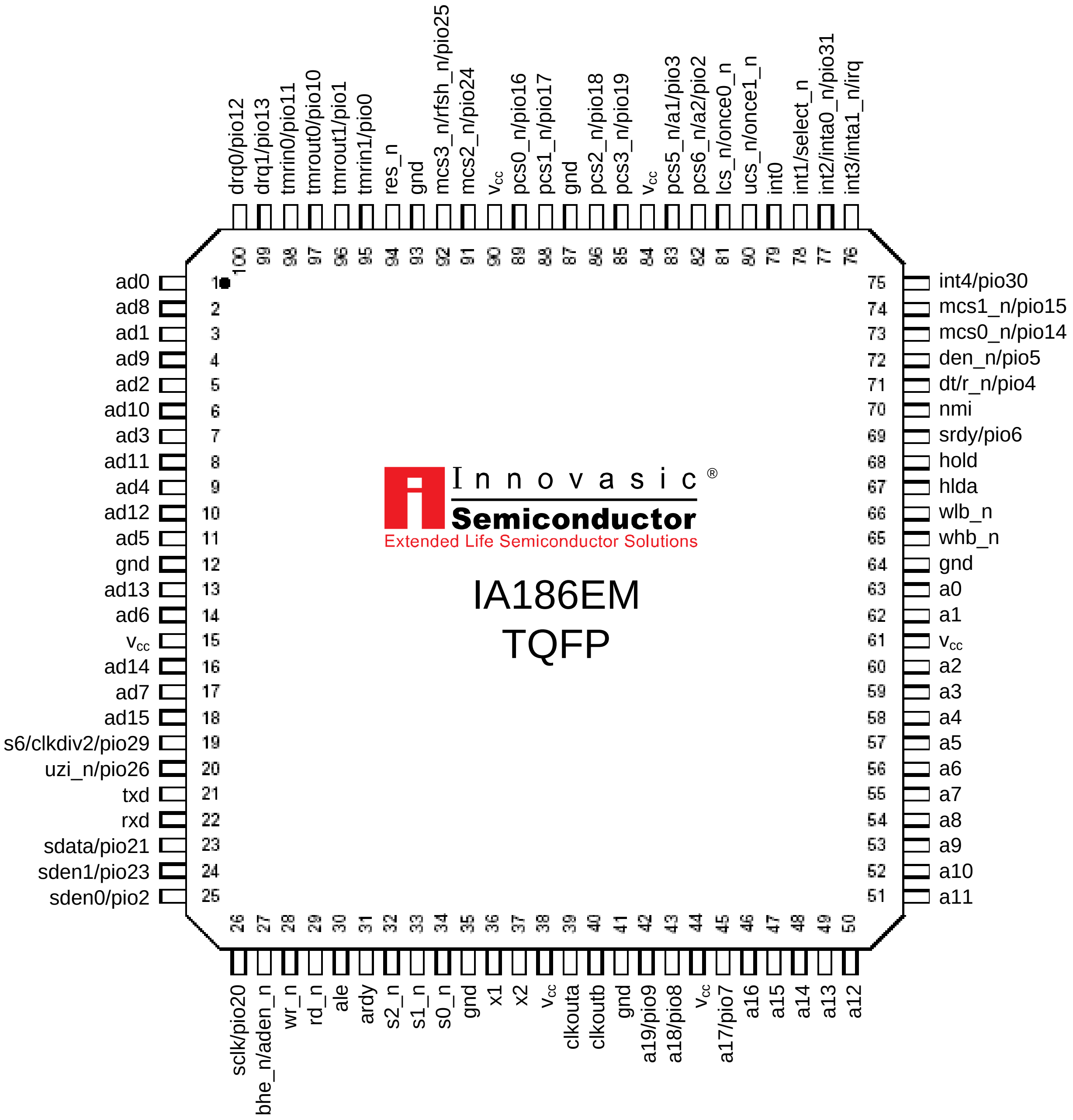


Figure 1. IA186EM TQFP Package Diagram

Table 2. IA186EM TQFP Alphabetic Pin Listing

Name	Pin
a0	63
a1	62
a2	60
a3	59
a4	58
a5	57
a6	56
a7	55
a8	54
a9	53
a10	52
a11	51
a12	50
a13	49
a14	48
a15	47
a16	46
a17/pio7	45
a18/pio8	43
a19/pio9	42
ad0	1
ad1	3
ad2	5
ad3	7
ad4	9
ad5	11
ad6	14
ad7	17
ad8	2
ad9	4
ad10	6
ad11	8
ad12	10
ad13	13
ad14	16
ad15	18
ale	30
ardy	30
bhe_n/aden_n	27
clkouta	39
clkoutb	40
den_n/pio5	72
drq0/pio12	100
drq1/pio13	99
dt/r_n/pio4	71
gnd	12
gnd	36
gnd	41
gnd	64
gnd	87
gnd	93
hlda	67
hold	68
int0	79
int1/select_n	78
int2/inta0_n/pio31	77
int3/inta1_n/irq	76
int4/pio30	75
lcs_n/once0_n	81
mcs0_n/pio14	73
mcs1_n/pio15	74
mcs2_n/pio24	91
mcs3_n/rfsh_n/pio25	92
nmi	70
pcs0_n/pio16	89
pcs1_npio	88
pcs2_n/pio18	86
pcs3_n/pio19	85
pcs5_n/a1/pio3	83
pcs6_n/a2/pio2	82
rd_n	29
res_n	94
rxd/pio23	24
s0_n	34
s1_n	33
s2_n	32
s6/clkdiv2/pio29	19
sclk/pio20	26
sdata/pio21	23
sden0/pio22	25
sden1/pio23	24
srdy/pio6	69
tmrin0/pio11	98
tmrin1/pio0	95
tmrout0/pio10	97
tmrout1/pio1	96
txd/pio27	21
ucs_n/once1_n	80
uzi_n/pio26	20
V_{cc}	15
V_{cc}	38
V_{cc}	44
V_{cc}	61
V_{cc}	84
V_{cc}	90
whb_n	65
wlb_n	66
wr_n	28
x1	36
x2	37

Table 6. IA186EM PQFP Alphabetic Pin Listing

Name	Pin
a0	40
a1	39
a2	37
a3	36
a4	35
a5	34
a6	33
a7	32
a8	31
a9	30
a10	29
a11	28
a12	27
a13	26
a14	25
a15	24
a16	23
a17/pio7	22
a18/pio8	20
a19/pio9	19
ad0	78
ad1	80
ad2	82
ad3	84
ad4	86
ad5	88
ad6	91
ad7	94
ad8	79
ad9	81
ad10	83
ad11	85
ad12	87
ad13	90
ad14	93
ad15	95
ale	7
ardy	8
bhe_n/aden_n	4
clkouta	16
clkoutb	17
den_n/pio5	49
drq0/pio12	77
drq1/pio13	76
dt/r_n/pio4	48
gnd	12
gnd	18
gnd	41
gnd	64
gnd	70
gnd	89
hlda	44
hold	45
int0	56
int1/select_n	55
int2/inta0_n/pio31	54
int3/inta1_n/irq	53
int4/pio30	52
lcs_n/once0_n	58
mcs0_n/pio14	50
mcs1_n/pio15	51
mcs2_n/pio24	68
mcs3_n/rfsh_n/pio25	69
nmi	47
pcs0_n/pio16	66
pcs1_n/pio17	65
pcs2_n/pio18	63
pcs3_n/pio19	62
pcs5_n/a1/pio3	60
pcs6_n/a2/pio2	59
rd_n	6
res_n	71
rxd/pio28	99
s0_n	11
s1_n	10
s2_n	9
s6/clkdiv2/pio29	96
sclk/pio20	3
sdata/pio21	100
sden0/pio22	2
sden1/pio23	1
srdy/pio6	46
tmrin0/pio11	75
tmrin1/pio0	72
tmrout0/pio10	74
tmrout1/pio1	73
txd/pio27	98
ucs_n/once1_n	57
uzi_n/pio26	97
V_{cc}	15
V_{cc}	21
V_{cc}	38
V_{cc}	61
V_{cc}	67
V_{cc}	92
whb_n	42
wlb_n	43
wr_n	5
x1	13
x2	14

2.1.5 IA188EM PQFP Package

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

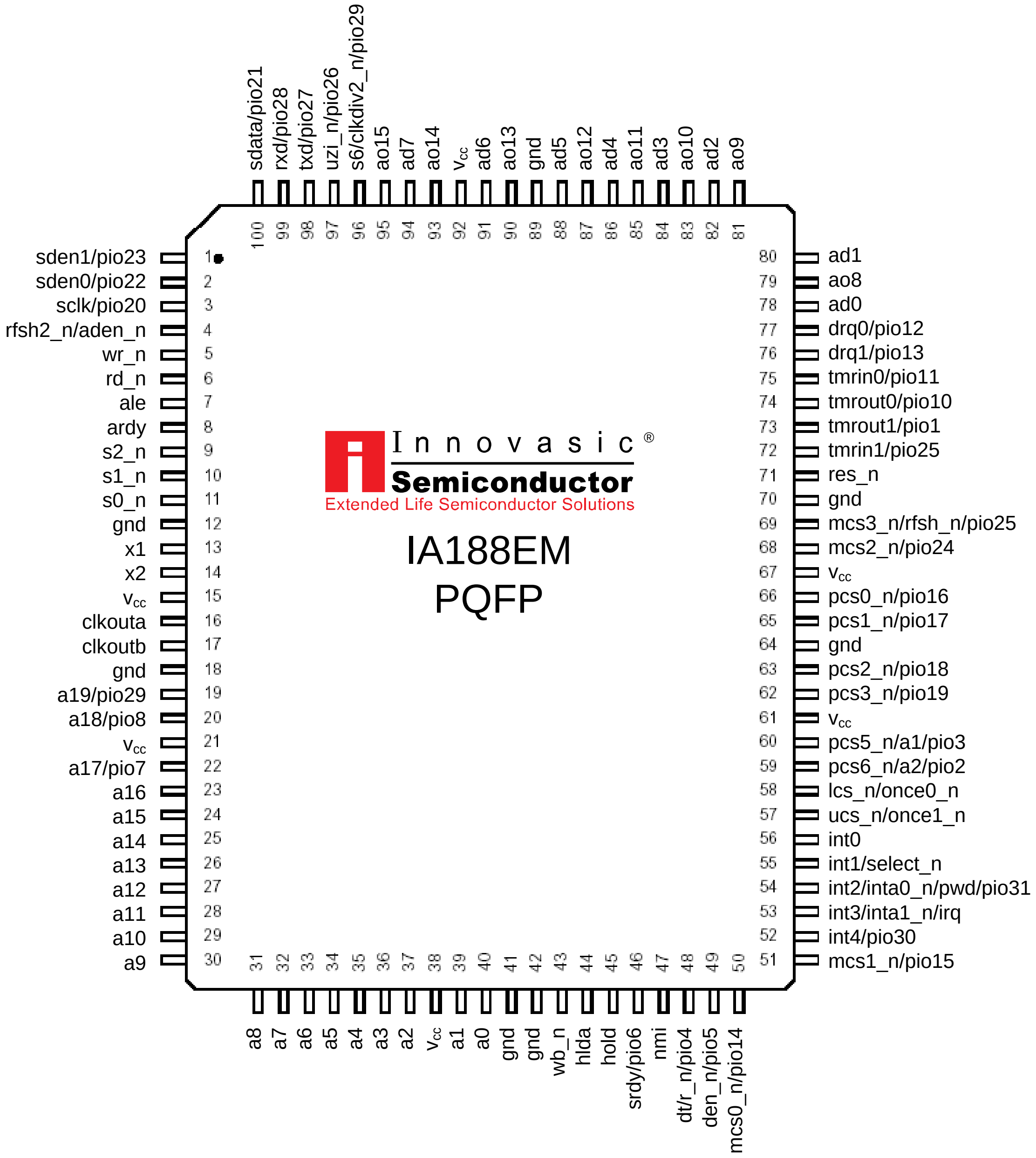


Figure 5. IA188EM PQFP Package Diagram

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

4.11 Upper Memory Chip Select

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

4.17 Timer Control

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

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

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

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

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

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

4.18 Direct Memory Access (DMA)

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

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

Table 15. Default Status of PIO Pins at Reset

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

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

[a] Input with pullup option available when used as PIO.

[b] Emulators use these pins and also a15–a0, ad15–ad0 (IA186EM), ale, bhe_n (IA186EM), clkouta, nmi, res_n, and s2_n–s0_n.

[c] If bhe_n/aden_n (IA186EM) or rfsh_n/aden_n (IA188EM) is held low during POR, these pins will revert to normal operation.

[d] Input with pulldown option available when used as PIO.

Select Sizes of M6–M0 by Total Block Size

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

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

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

- Bits [5–3]—Reserved → Set to 1.

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

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

5.1.15 MMCS (0a6h)

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

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

LMCS Block-Size Programming Values

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

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

5.1.18 UMCS (0a0h)

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

- Bit [14]—INHn Inhibit Bit → Gates the setting of the enable (EN) bit. This bit must be set to 1 in the same write operation that sets the enable (EN) bit. This bit always reads 0.

- Bit [13]—INT Interrupt Bit → An interrupt request is generated, by setting the INT bit to 1, when the Count register reaches its maximum, MC = 1.

- Bits [12–6]—Reserved → Set to 0.

- Bit [5]—MC Maximum Count → When the timer reaches its maximum count, this bit is set to 1, regardless of the interrupt enable bit. If preferred, this bit may be used by software polling rather than by interrupts to monitor timer status.

- Bits [4–1]—Reserved → Set to 0.

- Bit [0]—CONT Continuous Mode Bit → The timer will run continuously when this bit is set to 1. The timer will stop after each count run and EN will be cleared if this bit is set to 0.

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

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

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

Timer 2 has only one compare register, T2COMPA.

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

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

Table 51. Timer Maxcount Compare Registers

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

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

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

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

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

Table 56. INT2/INT3 Control Register

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

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

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

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

Table 83. Read Cycle Timing

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

[a] In nanoseconds.

Table 84. Write Cycle Timing

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

[a] In nanoseconds.

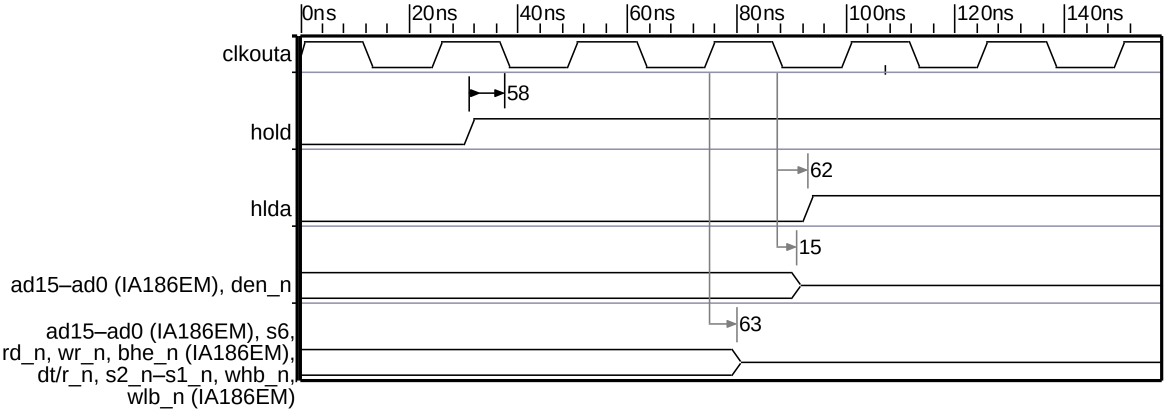


Figure 27. Bus Hold Entering

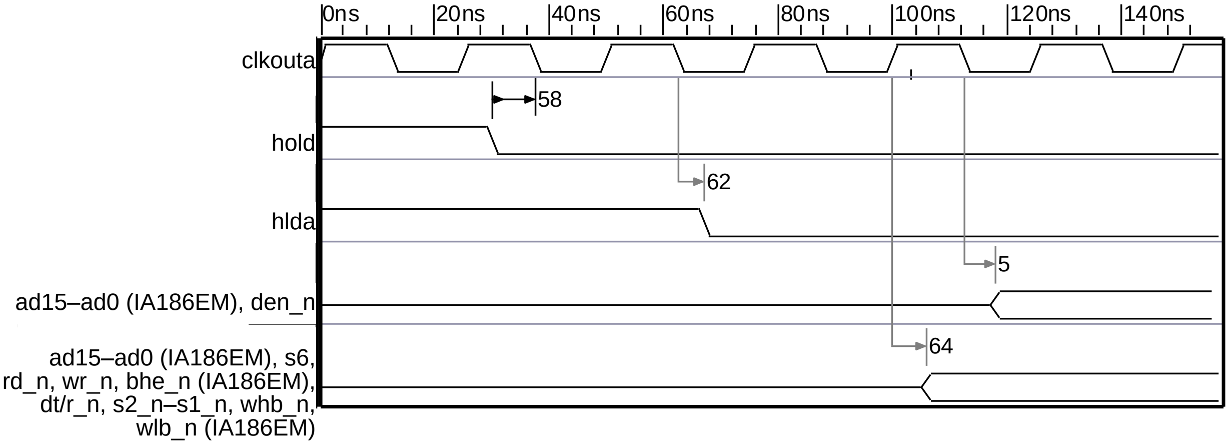


Figure 28. Bus Hold Leaving

Table 94. Instruction Set Summary *(Continued)*

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

Table 94. Instruction Set Summary *(Continued)*

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

7.2.1 Opcode

Opcode parameters and definitions are provided below.

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

7.2.2 Flags Affected After Instruction

Flags affected after instruction are shown below.

U	Undefined
-	Unchanged
R	Result-dependent

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

Workaround: Use idle mode instead of halt.

Errata No. 6

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

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

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

Errata No. 7

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

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

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