8XC196L*x* Supplement to 8XC196K*x*, 8XC196J*x*, 87C196CA User’s Manual

August 2004

Order Number: 272973-003

Information in this document is provided in connection with Intel products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The 8XC196Lx, 8XC196Kx, 8XC196Jx and 87C196CA microprocessors may contain design defects or errors known as errata which may cause the products to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at http://www.intel.com.

*Third-party brands and names are the property of their respective owners.

CHAPTER 1
GUIDE TO THIS MANUAL

CHAPTER 2
ARCHITECTURAL OVERVIEW

CHAPTER 3
ADDRESS SPACE

CHAPTER 4
STANDARD AND PTS INTERRUPTS

CHAPTER 5
I/O PORTS

CHAPTER 6
SYNCHRONOUS SERIAL I/O PORT

CHAPTER 7
EVENT PROCESSOR ARRAY

CHAPTER 8
J1850 COMMUNICATIONS CONTROLLER

CHAPTER 9
MINIMUM HARDWARE CONSIDERATIONS

CHAPTER 10
SPECIAL OPERATING MODES

CHAPTER 11
PROGRAMMING THE NONVOLATILE MEMORY

APPENDIX A
SIGNAL DESCRIPTIONS

GLOSSARY

FIGURES

FIGURES

TABLES

1

Guide to This Manual

CHAPTER 1
GUIDE TO THIS MANUAL

This document is a supplement to the *8XC196Kx, 8XC196Jx, 87C196CA Microcontroller Family User's Manual*. It describes the differences between the 8XC196L*x* and the 8XC196K*x* family of microcontrollers. For information not found in this supplement, please consult the *8XC196Kx, 8XC196Jx, 87C196CA Microcontroller Family User's Manual* (order number 272258) or the 8XC196L*x* datasheets listed in the "Related Documents" section of this chapter.

1.1 MANUAL CONTENTS

This supplement contains several chapters, an appendix, a glossary, and an index. This chapter, Chapter 1, provides an overview of the supplement. This section summarizes the contents of the remaining chapters and appendixes. The remainder of this chapter provides references to related documentation.

Chapter 2 — Architectural Overview — compares the features of the 8XC196L*x* microcontroller family with those of the 8XC196K*x* microcontroller family and describes the 87C196LA, LB internal clock circuitry.

Chapter 3 — Address Space — describes the addressable memory space of the 52-pin 8XC196L*x*, lists the peripheral special-function registers (SFRs), and provides tables of WSR values for windowing higher memory into the lower register file for direct access.

Chapter 4 — Standard and PTS Interrupts — describes the additional interrupts for the 87C196LB's J1850 communications controller peripheral and the SFRs that support those interrupts.

Chapter 5 — I/O Ports — describes the port differences and explains the change in the port reset state from a "logic 1" to a "logic 0" on the 87C196LA, LB.

Chapter 6 — Synchronous Serial I/O Port — describes the enhanced synchronous serial I/O (SSIO) port and explains how to program the two additional peripheral SFRs.

Chapter 7 — Event Processor Array — describes the event processor array channel differences.

Chapter 8 — J1850 Communications Controller — describes the 87C196LB's integrated J1850 controller and explains how to configure it.

Chapter 9 — Minimum Hardware Considerations — describes device reset options through the reset source register, and discusses hardware design considerations.

Chapter 10 — Special Operating Modes — illustrates the internal clock control circuitry of the 87C196LA, LB and describes how to enter and exit on-circuit emulation (ONCE) mode.

Chapter 11 — Programming the Nonvolatile Memory — describes the memory maps and recommended circuits to support programming of the 87C196LA, LB's 24 Kbytes of OTPROM.

Appendix A — Signal Descriptions — provides reference information for the 8XC196L*x* device pins, including descriptions of the pin functions, reset status of the I/O and control pins, and package pin assignments.

Glossary — defines terms with special meaning used throughout this supplement.

Index — lists key topics with page number references.

1.2 RELATED DOCUMENTS

Table 1-1 lists additional documents that you may find useful in designing systems incorporating the 8XC196L*x* microcontrollers.

Table 1-1. Related Documents

Title and Description	Order Number
8XC196Kx, 8XC196Jx, 87C196CA Microcontroller Family User's Manual	272258
87C196LA-20 MHz CHMOS 16-Bit Microcontroller Automotive datasheet	272806
87C196LB-20 MHz CHMOS 16-Bit Microcontroller Automotive datasheet	272807
83C196LD CHMOS 16-Bit Microcontroller Automotive datasheet	272805

2

Architectural Overview

CHAPTER 2 ARCHITECTURAL OVERVIEW

This chapter describes architectural differences between the 8XC196L*x* (87C196LA, 87C196LB, and 83C196LD) and the 8XC196K*x* (8XC196K*x*, 8XC196J*x*, and 87C196CA) microcontroller families. Both the 8XC196L*x* and the 8XC196K*x* are designed for high-speed calculations and fast I/O, and share a common architecture and instruction set with few deviations. This chapter provides a high-level overview of the deviations between the two families.

NOTE

> This supplement describes two product families within the MCS® 96 microcontroller family. For brevity, the name 8XC196L*x* is used when the discussion applies to all three L*x* controllers. Likewise, the name 8XC196K*x* is used when the discussion applies to all the K*x*, J*x*, and CA controllers.

2.1 MICROCONTROLLER FEATURES

Table 2-1 lists the features of the 8XC196L*x* and the 8XC196K*x*.

Table 2-1. Features of the 8XC196L*x* and 8XC196K*x* Product Famiies

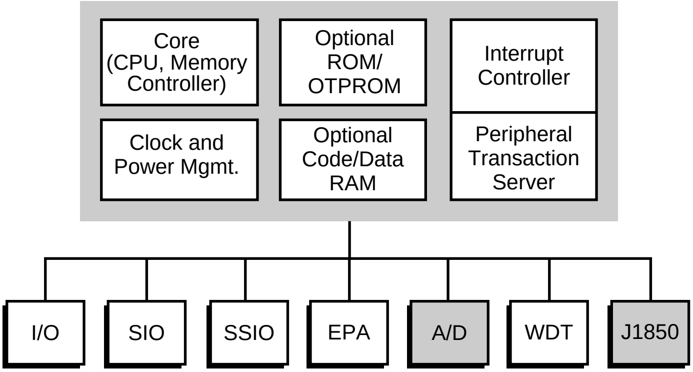
Device	Pins	OTPROM/ EPROM/ ROM (1)	Register RAM (2)	Code RAM	I/O Pins	EPA Pins	SIO/ SSIO Ports	A/D	CAN	J1850	Ext. Interrupt Pins
87C196LA	52	24 K	768	—	41	6	3	6	—	—	1
87C196LB	52	24 K	768	—	41	6	3	6	—	1	1
83C196LD	52	16 K	384	—	41	6	3	—	—	—	1
8XC196JV	52	48 K	1536	512	41	6	3	6	—	—	1
8XC196KT	68	32 K	1024	512	56	10	3	8	—	—	2
8XC196JT	52	32 K	1024	512	41	6	3	6	—	—	1
87C196CA	68	32 K	1024	256	51	6	3	6	1	—	2
8XC196KR	68	16 K	512	256	56	10	3	8	—	—	2
8XC196JR	52	16 K	512	256	41	6	3	6	—	—	1

NOTES:

1. Optional. The second character of the device name indicates the presence and type of nonvolatile memory. 80C196*xx* = none; 83C196*xx* = ROM; 87C196*xx* = OTPROM or EPROM.
2. Register RAM amounts include the 24 bytes allocated to core SFRs and the stack pointer.

2.2 BLOCK DIAGRAM

Figure 2-1 is a simplified block diagram that shows the major blocks within the microcontroller. Observe that the slave port peripheral does not exist on the 8XC196L*x*.



Note:
The J1850 peripheral is unique to the 87C196LB device.
The A/D peripheral is unique to the 87C196LA, LB devices.

A5253-01

Figure 2-1. 8XC196L*x* Block Diagram

2.3 INTERNAL TIMING

The 87C196LA, LB clock circuitry (Figure 2-2) implements a phase-locked loop and clock multiplier circuitry, which can substantially increase the CPU clock rate while using a lower-frequency input clock. The clock circuitry accepts an input clock signal on XTAL1 provided by an external crystal or oscillator. Depending on the value of the PLLEN pin, this frequency is routed either through the phase-locked loop and multiplier or directly to the divide-by-two circuit. The multiplier circuitry can double the input frequency (F_{XTAL1}) before the frequency (f) reaches the divide-by-two circuitry. The clock generators accept the divided input frequency (f/2) from the divide-by-two circuit and produce two nonoverlapping internal timing signals, PH1 and PH2. These signals are active when high.

NOTE

This manual uses lowercase “f” to represent the internal clock frequency. For the 87C196LA and LB, f is equal to either F_{XTAL1} or $2F_{XTAL1}$, depending on the clock multiplier mode, which is controlled by the PLLEN input pin.

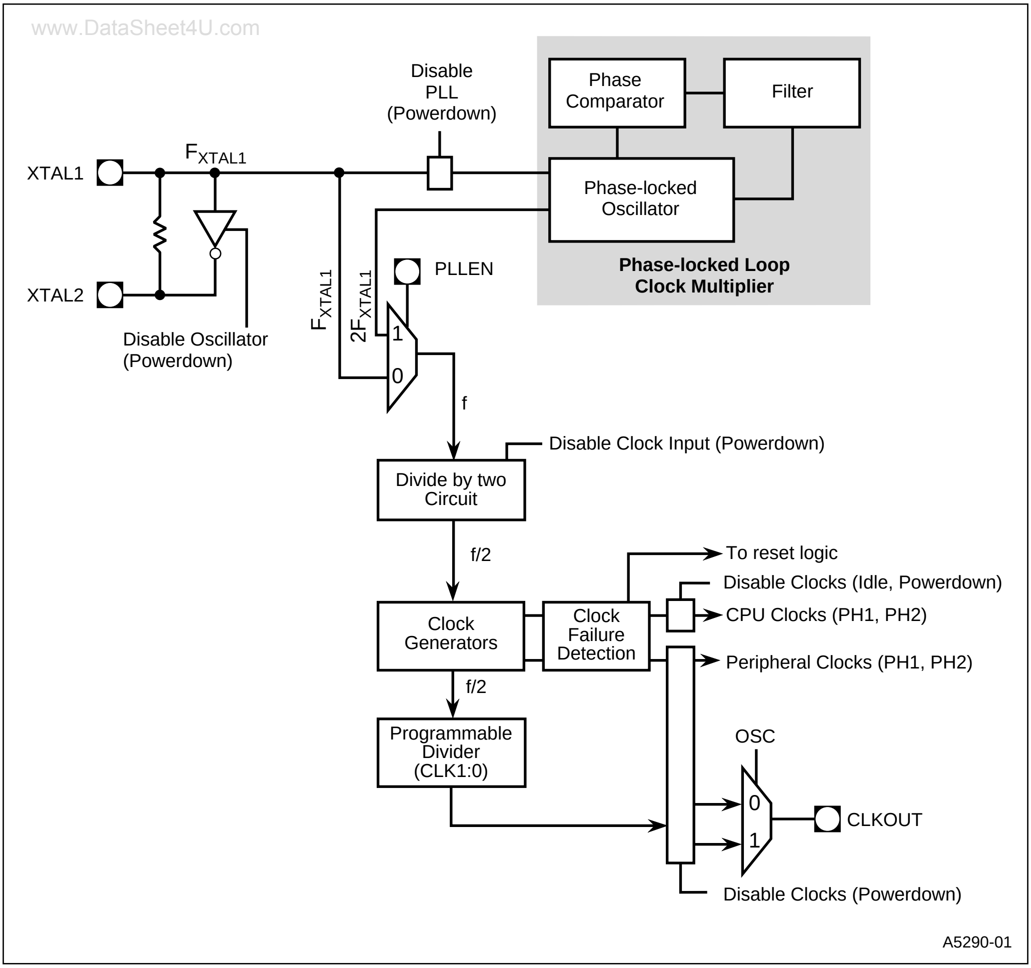


Figure 2-2. Clock Circuitry (87C196LA, LB Only)

The rising edges of PH1 and PH2 generate the internal CLKOUT signal (Figure 2-3). The clock circuitry routes separate internal clock signals to the CPU and the peripherals to provide flexibility in power management. It also outputs the CLKOUT signal on the CLKOUT pin. Because of the complex logic in the clock circuitry, the signal on the CLKOUT pin is a delayed version of the internal CLKOUT signal. This delay varies with temperature and voltage.

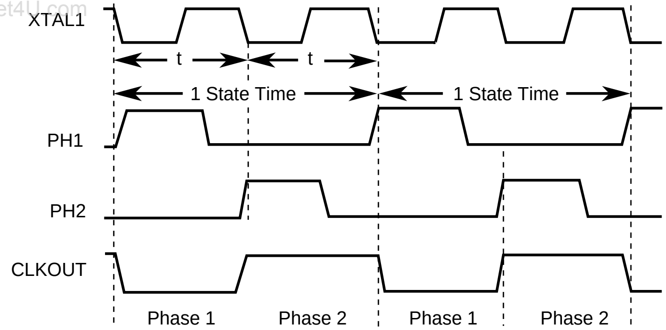


Figure 2-3. Internal Clock Phases (Assumes PLL is Bypassed)

The combined period of phase 1 and phase 2 of the internal CLKOUT signal defines the basic time unit known as a *state time* or *state*. Table 2-2 lists state time durations at various frequencies.

Table 2-2. State Times at Various Frequencies

f (Frequency Input to the Divide-by-two Circuit)	State Time
8 MHz	250 ns
12 MHz	167 ns
16 MHz	125 ns
20 MHz	100 ns

The following formulas calculate the frequency of PH1 and PH2, the duration of a state time, and the duration of a clock period (t).

$$\text{PH1 (in MHz)} = \frac{f}{2} = \text{PH2} \qquad \text{State Time (in µs)} = \frac{2}{f} \qquad t = \frac{1}{f}$$

Because the device can operate at many frequencies, this manual defines time requirements (such as instruction execution times) in terms of state times rather than specific measurements. Datasheets list AC characteristics in terms of clock periods (t; sometimes called T_{osc}).

Figure 2-4 illustrates the timing relationships between the input frequency (F_{XTAL1}), the operating frequency (f), and the CLKOUT signal with each PLLEN pin configuration. Table 2-3 details the relationships between the input frequency (F_{XTAL1}), the PLLEN pin, the operating frequency (f), the clock period (t), and state times.

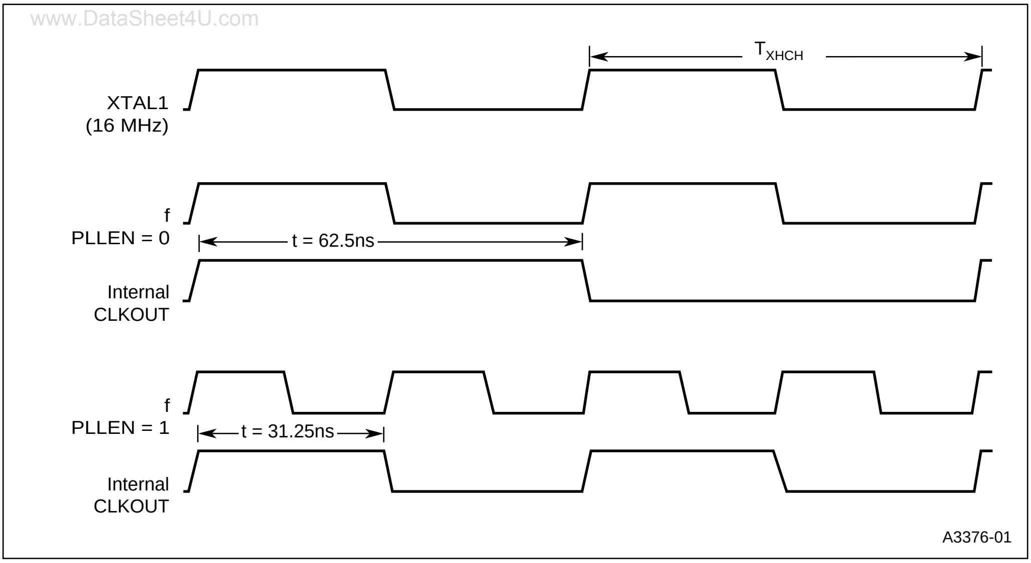


Figure 2-4. Effect of Clock Mode on Internal CLKOUT Frequency

Table 2-3. Relationships Between Input Frequency, Clock Multiplier, and State Times

F_{XTAL1} (Frequency on XTAL1)	PLLEN	Multiplier	f (Input Frequency to the Divide-by-two Circuit)	t (Clock Period)	State Time
4 MHz	0	1	4 MHz	250 ns	500 ns
8 MHz	0	1	8 MHz	125 ns	250 ns
12 MHz	0	1	12 MHz	83.5 ns	167 ns
16 MHz	0	1	16 MHz	62.5 ns	125 ns
20 MHz	0	1	20 MHz	50 ns	100 ns
4 MHz	1	2	8 MHz	125 ns	250 ns
8 MHz	1	2	16 MHz	62.5 ns	125 ns
10 MHz	1	2	20 MHz	50 ns	100 ns

2.4 EXTERNAL TIMING

You can control the output frequency on the CLKOUT pin by programming two uneraseable PROM bits. Figure 2-5 illustrates the read-only USFR1, which reflects the state of the unerasable PROM bits. You can select one of three frequencies: f/2, f/4, or f/8. As Figure 2-2 on page 2-3 shows, the configurable divider accepts the output of the clock generators (f/2) and further divides that frequency to produce the desired output frequency. The CLK1:0 bits control the divisor (divide f/2 by either 1, 2, or 4).

USFR1 (read only)

Address: 1FF2H
Reset State: XXH

The UPROM special-function register 1 (USFR1) reflects the status of unerasable, programmable read-only memory (UPROM) locations. This read-only register reflects the status of two bits that control the output frequency on CLKOUT.

7							**0**
—	—	—	—	—	—	CLK1	CLK0

Bit Number	Bit Mnemonic	Function
7:2	—	Reserved.
1:0	CLK1:0	CLKOUT Control These bits reflect the programmed frequency of the CLKOUT signal: **CLK1 CLK0** 0 0 divide by 1 (CLKOUT = f/2) 0 1 divide by 2 (CLKOUT = f/4) 1 0 divide by 4 (CLKOUT = f/8) 1 1 divide by 1 (CLKOUT = f/2)

Figure 2-5. Unerasable PROM 1 (USFR1) Register (LA, LB Only)

To program these bits, write the correct value to the locations shown in Table 2-4 using slave programming mode. During normal operation, you can determine the values of these bits by reading the UPROM SFR (Figure 2-5).

You can verify a UPROM bit to make sure it programmed, but you cannot erase it. For this reason, Intel cannot test the bits before shipment. However, Intel does test the features that the UPROM bits enable, so the only undetectable defects are (unlikely) defects within the UPROM cells themselves.

Table 2-4. UPROM Programming Values and Locations

To set this bit	Write this value	To this location
CLK0	0001H	0768H
CLK1	0002H	0728H

2.5 INTERNAL PERIPHERALS

The internal peripheral modules provide special functions for a variety of applications. This section provides a brief description of the peripherals that differ between the 8XC196L*x* and the 8XC196K*x* families.

2.5.1 I/O Ports

The I/O ports of the 8XC196L*x* are functionally identical to those of the 8XC196J*x*. However, on the 87C196LA and LB the reset state level of all 41 general-purpose I/O pins has changed from a weak logic “1” (wk1) to a weak logic “0” (wk0).

2.5.2 Synchronous Serial I/O Port

The synchronous serial I/O (SSIO) port on the 8XC196L*x* has been enhanced, implementing two new special function registers (SSIO0_CLK and SSIO1_CLK) that allow you to select the operating mode and configure the phase and polarity of the serial clock signals.

2.5.3 Event Processor Array

The 8XC196L*x*’s event processor array (EPA) is functionally identical to that of the 8XC196J*x*, except that it has only two EPA capture/compare channels without pins instead of four. In addition the LD has no compare-only channels.

2.5.4 J1850 Communications Controller

The 87C196LB microcontroller has a peripheral not found on the 8XC196K*x* microcontrollers or any other L*x* microcontroller, the J1850 peripheral. The J1850 communications controller manages communications between multiple network nodes. This integrated peripheral supports the 10.4 Kb/s VPW (variable pulse-width) medium-speed, class B, in-vehicle network protocol. It also supports both the standard and in-frame response (IFR) message framing as specified by the *Society of Automotive Engineering* (*SAE) J1850* (revised May 1994) technical standards.

2.6 DESIGN CONSIDERATIONS

With the exception of a few new multiplexed functions, the 8XC196L*x* microcontrollers are pin compatible with the 8XC196J*x* microcontrollers. The 8XC196J*x* microcontrollers are 52-lead versions of 8XC196K*x* microcontrollers. For registers that are implemented in both the 8XC196L*x* and the 8XC196J*x*, configure the 8XC196L*x* register as you would for the 8XC196J*x* unless differences are noted in this supplement.

3

Address Space

CHAPTER 3
ADDRESS SPACE

This chapter describes the differences in the address space of the 8XC196L*x* from that of the 8XC196K*x*.

3.1 ADDRESS PARTITIONS

Table 3-1 is an address map of the 8XC196L*x* and 8XC196K*x* microcontroller family members.

Table 3-1. Address Map

Device and Hex Address Range						Description	Addressing Modes
CA	JR, KR	LD	LA, LB	JT, KT	JV		
FFFF A000	FFFF 6000	FFFF 6000	FFFF 8000	FFFF A000	FFFF E000	External device (memory or I/O) connected to address/data bus	Indirect or indexed
9FFF 2080	5FFF 2080	5FFF 2080	7FFF 2080	9FFF 2080	DFFF 2080	Program memory (internal nonvolatile or external memory); see Note 1	Indirect or indexed
207F 2000	207F 2000	207F 2000	207F 2000	207F 2000	207F 2000	Special-purpose memory (internal nonvolatile or external memory)	Indirect or indexed
1FFF 1FE0	1FFF 1FE0	1FFF 1FE0	1FFF 1FE0	1FFF 1FE0	1FFF 1FE0	Memory-mapped SFRs	Indirect or indexed
1FDF 1F00	1FDF 1F00	1FDF 1F00	1FDF 1F00	1FDF 1F00	1FDF 1F00	Peripheral SFRs (Includes J1850 SFRs on 87C196LB)	Indirect, indexed, or windowed direct
1EFF 1E00	—	—	—	—	—	CAN SFRs	Indirect, indexed, or windowed direct
1DFF 1C00	1EFF 1C00	1EFF 1C00	1EFF 0300	1EFF 1C00	1EFF 1E00	External device (memory or I/O) connected to address/data bus; (future SFR expansion; see Note 2)	Indirect or indexed
—	—	—	—	—	1DFF 1C00	Register RAM	Indirect, indexed, or windowed direct

NOTES:

1. After a reset, the device fetches its first instruction from 2080H.
2. The content or function of these locations may change in future device revisions, in which case a program that relies on a location in this range might not function properly.

Table 3-1. Address Map (Continued)

Device and Hex Address Range						Description	Addressing Modes
CA	JR, KR	LD	LA, LB	JT, KT	JV		
1BFF 0500	1BFF 0500	1BFF 0600	—	1BFF 0600	1BFF 0600	External device (memory or I/O) connected to address/data bus	Indirect or indexed
04FF 0400	04FF 0400	—	—	05FF 0400	05FF 0400	Internal code or data RAM	Indirect or indexed
—	03FF 0200	05FF 0180	—	—	—	External device (memory or I/O) connected to address/data bus	Indirect or indexed
03FF 0100	01FF 0100	017F 0100	02FF 0100	03FF 0100	03FF 0100	Upper register file (general-purpose register RAM)	Indirect, indexed, or windowed direct
00FF 0000	00FF 0000	00FF 0000	00FF 0000	00FF 0000	00FF 0000	Lower register file (register RAM, stack pointer, and CPU SFRs)	Direct, indirect, or indexed

NOTES:

1. After a reset, the device fetches its first instruction from 2080H.
2. The content or function of these locations may change in future device revisions, in which case a program that relies on a location in this range might not function properly.

3.2 REGISTER FILE

Figure 3-1 compares the register file addresses of the 8XC196L*x* and 8XC196K*x*. The register file in Figure 3-1 is divided into an upper register file and a lower register file. The upper register file consists of general-purpose register RAM. The lower register file contains general-purpose register RAM along with the stack pointer (SP) and the CPU special-function registers (SFRs).

Table 3-2 lists the register file memory addresses. The RALU accesses the lower register file directly, without the use of the memory controller. It also accesses a *windowed* location directly (see “Windowing” on page 3-6). The upper register file and the peripheral SFRs can be windowed. Registers in the lower register file and registers being windowed can be accessed with register-direct addressing.

NOTE

> The register file must not contain code. An attempt to execute an instruction from a location in the register file causes the memory controller to fetch the instruction from external memory.

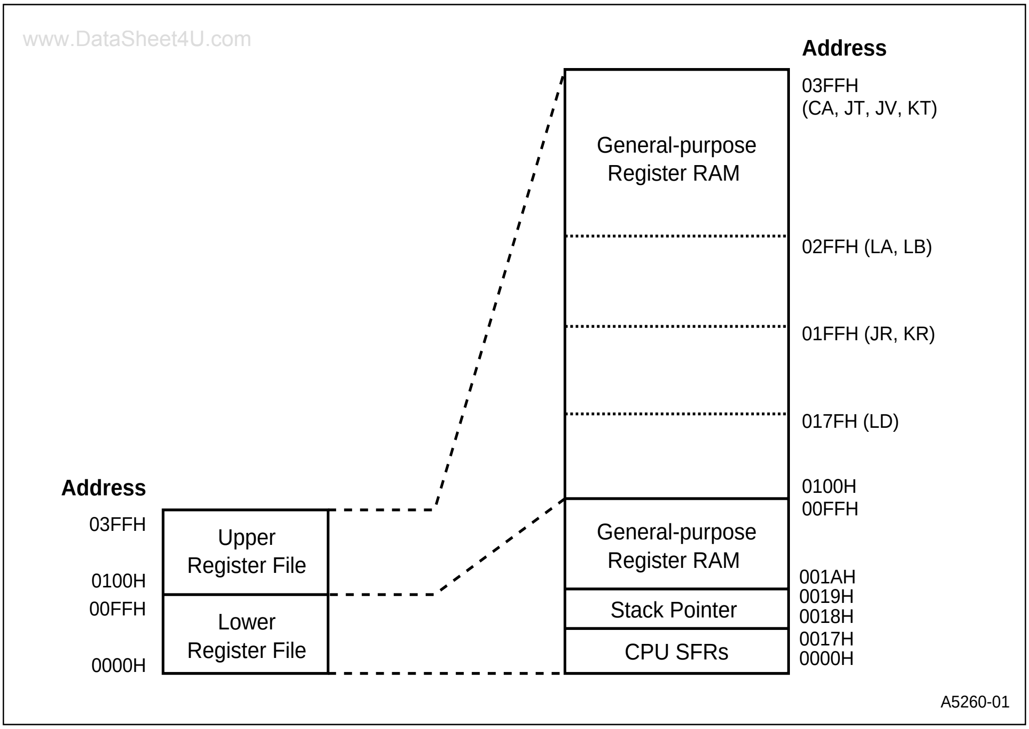


Figure 3-1. Register File Address Map

Table 3-2. Register File Memory Addresses

Device and Hex Address Range					Description	Addressing Modes
JV	CA,JT,KT	LA, LB	JR, KR	LD		
1DFF 1C00	—	—	—	—	Register RAM	Indirect, indexed, or windowed direct
03FF 0100	03FF 0100	02FF 0100	01FF 0100	017F 0100	Upper register file (register RAM)	Indirect, indexed, or windowed direct
00FF 001A	00FF 001A	00FF 001A	00FF 001A	00FF 001A	Lower register file (register RAM)	Direct, indirect, or indexed
0019 0018	0019 0018	0019 0018	0019 0018	0019 0018	Lower register file (stack pointer)	Direct, indirect, or indexed
0017 0000	0017 0000	0017 0000	0017 0000	0017 0000	Lower register file (CPU SFRs)	Direct, indirect, or indexed

3.3 PERIPHERAL SPECIAL-FUNCTION REGISTERS

Table 3-3 lists the peripheral SFR addresses. Highlighted addresses are unique to the 8XC196L*x*.

Table 3-3. 8XC196L*x* Peripheral SFRs

Ports 3, 4, 5, and UPROM SFRs		
Address	**High (Odd) Byte**	**Low (Even) Byte**
1FFEH	P4_PIN	P3_PIN
1FFCH	P4_REG	P3_REG
1FFAH	SLP_CON	SLP_CMD
1FF8H	Reserved	SLP_STAT
1FF6H	P5_PIN	USFR
1FF4H	P5_REG	P34_DRV
1FF2H	P5_DIR	**USFR1 (LA, LB)**
1FF0H	P5_MODE	Reserved
1FEEH	Reserved	Reserved
1FECH	Reserved	Reserved
1FEAH	Reserved	Reserved
1FE8H	Reserved	Reserved
1FE6H	Reserved	Reserved
1FE4H	Reserved	Reserved
1FE2H	Reserved	Reserved
1FE0H	Reserved	Reserved

Ports 0, 1, 2, and 6 SFRs		
Address	**High (Odd) Byte**	**Low (Even) Byte**
1FDEH	Reserved	Reserved
1FDCH	Reserved	Reserved
1FDAH	Reserved	P0_PIN
1FD8H	Reserved	Reserved
1FD6H	P6_PIN	P1_PIN
1FD4H	P6_REG	P1_REG
1FD2H	P6_DIR	P1_DIR
1FD0H	P6_MODE	P1_MODE
1FCEH	P2_PIN	Reserved
1FCCH	P2_REG	Reserved
1FCAH	P2_DIR	Reserved
1FC8H	P2_MODE	Reserved
1FC6H	Reserved	Reserved
1FC4H	Reserved	Reserved
1FC2H	Reserved	Reserved
1FC0H	Reserved	Reserved

† Must be addressed as a word.

Table 3-3. 8XC196L*x* Peripheral SFRs (Continued)

SIO and SSIO SFRs

Address	High (Odd) Byte	Low (Even) Byte
1FBEH	Reserved	Reserved
1FBCH	SP_BAUD (H)	SP_BAUD (L)
1FBAH	SP_CON	SBUF_TX
1FB8H	SP_STATUS	SBUF_RX
1FB6H	**SSIO1_CLK**	Reserved
1FB4H	**SSIO0_CLK**	SSIO_BAUD
1FB2H	SSIO1_CON	SSIO1_BUF
1FB0H	SSIO0_CON	SSIO0_BUF

A/D SFRs (LA, LB Only)

Address	High (Odd) Byte	Low (Even) Byte
1FAEH	AD_TIME	AD_TEST
1FACH	Reserved	AD_COMMAND
1FAAH	AD_RESULT (H)	AD_RESULT (L)

EPA Interrupt SFRs

Address	High (Odd) Byte	Low (Even) Byte
1FA8H	Reserved	EPAIPV
1FA6H	Reserved	EPA_PEND1
1FA4H	Reserved	EPA_MASK1
†1FA2H	EPA_PEND (H)	EPA_PEND (L)
†1FA0H	EPA_MASK (H)	EPA_MASK (L)

Timer 1, Timer 2, and EPA SFRs

Address	High (Odd) Byte	Low (Even) Byte
†1F9EH	TIMER2 (H)	TIMER2 (L)
1F9CH	Reserved	T2CONTROL
†1F9AH	TIMER1 (H)	TIMER1 (L)
1F98H	Reserved	T1CONTROL
1F96H	Reserved	Reserved
1F94H	Reserved	Reserved
1F92H	Reserved	**RST_SRC**
1F90H	Reserved	Reserved

EPA SFRs

Address	High (Odd) Byte	Low (Even) Byte
†1F8EH	COMP1_TIME (H)	COMP1_TIME (L)
1F8CH	Reserved	COMP1_CON
†1F8AH	COMP0_TIME (H)	COMP0_TIME (L)
1F88H	Reserved	COMP0_CON
†1F86H	EPA9_TIME (H)	EPA9_TIME (L)
1F84H	Reserved	EPA9_CON
†1F82H	EPA8_TIME (H)	EPA8_TIME (L)
1F80H	Reserved	EPA8_CON
†1F7EH	EPA7_TIME (H)	EPA7_TIME (L)
1F7CH	Reserved	EPA7_CON
†1F7AH	EPA6_TIME (H)	EPA6_TIME (L)
1F78H	Reserved	EPA6_CON
†1F76H	EPA5_TIME (H)	EPA5_TIME (L)
1F74H	Reserved	EPA5_CON
†1F72H	EPA4_TIME (H)	EPA4_TIME (L)
1F70H	Reserved	EPA4_CON
†1F6EH	EPA3_TIME (H)	EPA3_TIME (L)
†1F6CH	EPA3_CON (H)	EPA3_CON (L)
†1F6AH	EPA2_TIME (H)	EPA2_TIME (L)
1F68H	Reserved	EPA2_CON
†1F66H	EPA1_TIME (H)	EPA1_TIME (L)
†1F64H	EPA1_CON (H)	EPA1_CON (L)
†1F62H	EPA0_TIME (H)	EPA0_TIME (L)
1F60H	Reserved	EPA0_CON

J1850 SFRs (LB Only)

Address	High (Odd) Byte	Low (Even) Byte
1F5EH	Reserved	Reserved
1F5CH	Reserved	Reserved
1F5AH	Reserved	Reserved
1F58H	Reserved	**J_DLY**
1F56H	Reserved	Reserved
1F54H	Reserved	**J_CFG**
1F52H	**J_STAT**	**J_RX**
1F50H	**J_CMD**	**J_TX**

† Must be addressed as a word.

3.4 WINDOWING

Windowing maps a segment of higher memory (the upper register file or peripheral SFRs) into the lower register file. The window selection register (WSR) selects a 32-, 64- or 128-byte segment of higher memory to be windowed into the top of the lower register file space. Table 3-4 lists the WSR values for windowing the upper register file for both the 8XC196L*x* and 8XC196K*x*.

Table 3-4. Windows

Base Address	WSR Value for 32-byte Window (00E0–00FFH)	WSR Value for 64-byte Window (00C0–00FFH)	WSR Value for 128-byte Window (0080–00FFH)
Peripheral SFRs			
1FE0H	7FH (Note)	3FH (Note)	1FH (Note)
1FC0H	7EH		
1FA0H	7DH	3EH	
1F80H	7CH		
1F60H	7BH	3DH	1EH
1F40H	7AH		
1F20H	79H	3CH	
1F00H	78H		
CAN Peripheral SFRs (87C196CA Only)			
1EE0H	77H	3BH	1DH
1EC0H	76H		
1EA0H	75H	3AH	
1E80H	74H		
1E60H	73H	39H	1CH
1E40H	72H		
1E20H	71H	38H	
1E00H	70H		
Register RAM (87C196JV Only)			
1DE0H	6FH	37H	1BH
1DC0H	6EH		
1DA0H	6DH	36H	
1D80H	6CH		
1D60H	6BH	35H	1AH
1D40H	6AH		
1D20H	69H	34H	
1D00H	68H		

NOTE: Locations 1FE0–1FFFH contain memory-mapped SFRs that cannot be accessed through a window. Reading these locations through a window returns FFH; writing these locations through a window has no effect.

Table 3-4. Windows (Continued)

Base Address	WSR Value for 32-byte Window (00E0–00FFH)	WSR Value for 64-byte Window (00C0–00FFH)	WSR Value for 128-byte Window (0080–00FFH)
Register RAM (87C196JV Only; Continued)			
1CE0H	67H	33H	19H
1CC0H	66H		
1CA0H	65H	32H	
1C80H	64H		
1C60H	63H	31H	18H
1C40H	62H		
1C20H	61H	30H	
1C00H	60H		
Upper Register File (CA, JT, JV, KT)			
03E0H	5FH	2FH	17H
03C0H	5EH		
03A0H	5DH	2EH	
0380H	5CH		
0360H	5BH	2DH	16H
0340H	5AH		
0320H	59H	2CH	
0300H	58H		
Upper Register File (CA, JT, JV, KT, LA, LB)			
02E0H	57H	2BH	15H
02C0H	56H		
02A0H	55H	2AH	
0280H	54H		
0260H	53H	29H	14H
0240H	52H		
0220H	51H	28H	
0200H	50H		
Upper Register File (CA, JR, JT, JV, KR, KT, LA, LB)			
01E0H	4FH	27H	13H
01C0H	4EH		
01A0H	4DH	26H	
0180H	4CH		

NOTE: Locations 1FE0–1FFFH contain memory-mapped SFRs that cannot be accessed through a window. Reading these locations through a window returns FFH; writing these locations through a window has no effect.

Table 3-4. Windows (Continued)

Base Address	WSR Value for 32-byte Window (00E0–00FFH)	WSR Value for 64-byte Window (00C0–00FFH)	WSR Value for 128-byte Window (0080–00FFH)
Upper Register File (CA, JR, JT, JV, KR, KT, LA, LB, LD)			
0160H	4BH	25H	12H
0140H	4AH		
0120H	49H	24H	
0100H	48H		

NOTE: Locations 1FE0–1FFFH contain memory-mapped SFRs that cannot be accessed through a window. Reading these locations through a window returns FFH; writing these locations through a window has no effect.

4

Standard and PTS Interrupts

CHAPTER 4
STANDARD AND PTS INTERRUPTS

The interrupt structure of the 8XC196L*x* is the same as that of the 8XC196J*x*. The only difference is that the slave port interrupts (INT08:06) now support the J1850 controller peripheral.

4.1 INTERRUPT SOURCES, VECTORS, AND PRIORITIES

Table 4-1 lists the 8XC196Lx's interrupts sources, default priorities (30 is highest and 0 is lowest), and vector addresses.

Table 4-1. Interrupt Sources, Vectors, and Priorities

Interrupt Source	Mnemonic	Interrupt Controller Service			PTS Service		
		Name	Vector	Priority[†]	Name	Vector	Priority
Nonmaskable Interrupt	NMI[†]	INT15	203EH	30	—	—	—
EXTINT Pin	EXTINT	INT14	203CH	14	PTS14	205CH	29
Reserved	—	INT13	203AH	13	PTS13	205AH	28
SIO Receive	RI	INT12	2038H	12	PTS12	2058H	27
SIO Transmit	TI	INT11	2036H	11	PTS11	2056H	26
SSIO Channel 1 Transfer	SSIO1	INT10	2034H	10	PTS10	2054H	25
SSIO Channel 0 Transfer	SSIO0	INT09	2032H	09	PTS09	2052H	24
J1850 Status (LB only)	J1850ST	INT08	2030H	08	PTS08	2050H	23
Reserved (LA, LD)	—	INT08	2030H	08	PTS08	2050H	23
Unimplemented Opcode	—	—	2012H	—	—	—	—
Software TRAP Instruction	—	—	2010H	—	—	—	—
J1850 Receive (LB only)	J1850RX	INT07	200EH	07	PTS07	204EH	22
Reserved (LA, LD)	—	INT07	200EH	07	PTS07	204EH	22
J1850 Transmit (LB only)	J1850TX	INT06	200CH	06	PTS06	204CH	21
Reserved (LA, LD)	—	INT06	200CH	06	PTS06	204CH	21
A/D Conv. Complete (LA, LB)	AD_DONE	INT05	200AH	05	PTS05	204AH	20
Reserved (LD)	—	INT05	200AH	05	PTS05	204AH	20
EPA Capture/Compare 0	EPA0	INT04	2008H	04	PTS04	2048H	19
EPA Capture/Compare 1	EPA1	INT03	2006H	03	PTS03	2046H	18
EPA Capture/Compare 2	EPA2	INT02	2004H	02	PTS02	2044H	17
EPA Capture/Compare 3	EPA3	INT01	2002H	01	PTS01	2042H	16
EPA Capture/Compare 6–9, EPA 0–3, 8–9 Overrun, EPA Compare 0–1[†††], Timer 1 Overflow, & Timer 2 Overflow	EPA*x* [††]	INT00	2000H	00	PTS00	2040H	15

[†] The NMI pin is not bonded out on the 8XC196L*x*. To protect against glitches, create a dummy interrupt service routine that contains a RET instruction.

[††] These interrupts are individually prioritized in the EPAIPV register. Read the EPA pending registers (EPA_PEND and EPA_PEND1) to determine which source caused the interrupt.

[†††] 87C196LA, LB only. The 83C196LD has no EPA compare-only channels.

4.2 INTERRUPT REGISTERS

This section describes the changes in the interrupt register bit definitions for the 8XC196L*x* family.

4.2.1 Interrupt Mask Registers

Figures 4-1 and 4-2 illustrate the interrupt mask registers for the 8XC196L*x* microcontrollers.

INT_MASK

Address: 0008H
Reset State: 00H

The interrupt mask (INT_MASK) register enables or disables (masks) individual interrupt requests. (The EI and DI instructions enable and disable servicing of all maskable interrupts.) INT_MASK is the low byte of the processor status word (PSW). PUSHF or PUSHA saves the contents of this register onto the stack and then clears this register. Interrupt calls cannot occur immediately following a push instruction. POPF or POPA restores it.

	7							**0**
LA	—	—	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
LB	J1850RX	J1850TX	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
LD	—	—	—	EPA0	EPA1	EPA2	EPA3	EPA*x*

Bit Number	Function
7:0†	Setting a bit enables the corresponding interrupt. **Bit Mnemonic** — **Interrupt Description** J1850RX — J1850 Receive (LB only) J1850TX — J1850 Transmit (LB only) AD — A/D Conversion Complete (LA, LB) EPA0 — EPA Capture/Compare Channel 0 EPA1 — EPA Capture/Compare Channel 1 EPA2 — EPA Capture/Compare Channel 2 EPA3 — EPA Capture/Compare Channel 3 EPA*x*†† — Shared EPA interrupt †† EPA 6–9 capture/compare channel events, EPA 0–1 compare channel events†††, EPA 0–3 and 8–9 capture/compare overruns, and timer overflows can generate this multiplexed interrupt. The EPA mask and pending registers decode the EPA*x* interrupt. Write the EPA mask registers to enable the interrupt sources; read the EPA pending registers to determine which source caused the interrupt. ††† 87C196LA, LB only.

† Bits 6–7 are reserved on the 87C196LA, and bits 5–7 are reserved on the 83C196LD. For compatibility with future devices, write zeros to these bits.

Figure 4-1. Interrupt Mask (INT_MASK) Register

INT_MASK1

Address: 0013H
Reset State: 00H

The interrupt mask 1 (INT_MASK1) register enables or disables (masks) individual interrupt requests. (The EI and DI instructions enable and disable servicing of all maskable interrupts.) INT_MASK1 can be read from or written to as a byte register. PUSHA saves this register on the stack and POPA restores it.

	7							0
LB	NMI	EXTINT	—	RI	TI	SSIO1	SSIO0	J1850ST

	7							0
LA, LD	NMI	EXTINT	—	RI	TI	SSIO1	SSIO0	—

Bit Number	Function
7:0†	Setting a bit enables the corresponding interrupt. **Bit Mnemonic** — **Interrupt Description** NMI†† — Nonmaskable Interrupt EXTINT — EXTINT Pin Reserved — — RI — SIO Receive TI — SIO Transmit SSIO1 — SSIO1 Transfer SSIO0 — SSIO0 Transfer J1850ST — J1850 Status (LB only) †† NMI is always enabled. This nonfunctional mask bit exists for design symmetry with the INT_PEND1 register. Always write zero to this bit.

† Bit 5 is reserved on the 8XC196L*x* devices, and bit 0 is reserved on the 87C196LA and 83C196LD. For compatibility with future devices, always write zeros to these bits.

Figure 4-2. Interrupt Mask 1 (INT_MASK1) Register

4.2.2 Interrupt Pending Registers

Figures 4-3 and 4-4 illustrate the interrupt pending registers for the 8XC196L*x* microcontrollers.

INT_PEND

Address: 0009H
Reset State: 00H

When hardware detects an interrupt request, it sets the corresponding bit in the interrupt pending (INT_PEND or INT_PEND1) registers. When the vector is taken, the hardware clears the pending bit. Software can generate an interrupt by setting the corresponding interrupt pending bit.

	7							0
LA	—	—	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
	7							**0**
LB	J1850RX	J1850TX	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
	7							**0**
LD	—	—	—	EPA0	EPA1	EPA2	EPA3	EPA*x*

Bit Number	Function
7:0†	Any set bit indicates that the corresponding interrupt is pending. The interrupt bit is cleared when processing transfers to the corresponding interrupt vector. **Bit Mnemonic** **Interrupt Description** J1850RX J1850 Receive (LB only) J1850TX J1850 Transmit (LB only) AD A/D Conversion Complete (LA, LB) EPA0 EPA Capture/Compare Channel 0 EPA1 EPA Capture/Compare Channel 1 EPA2 EPA Capture/Compare Channel 2 EPA3 EPA Capture/Compare Channel 3 EPA*x*†† Shared EPA Interrupt †† EPA 6–9 capture/compare channel events, EPA 0–1 compare channel events†††, EPA 0–3 and 8–9 capture/compare overruns, and timer overflows can generate this shared interrupt. Write the EPA mask registersto enable the interrupt sources; read the EPA pending registers to determine which source caused the interrupt. ††† 87C196LA, LB only.

† Bits 6–7 are reserved on the 87C196LA, and bits 5–7 are reserved on the 83C196LD. For compatibility with future devices, write zeros to these bits.

Figure 4-3. Interrupt Pending (INT_PEND) Register

INT_PEND1

Address: 0012H
Reset State: 00H

When hardware detects an interrupt request, it sets the corresponding bit in the interrupt pending (INT_PEND or INT_PEND1) registers. When the vector is taken, the hardware clears the pending bit. Software can generate an interrupt by setting the corresponding interrupt pending bit.

	7							0
LB	NMI	EXTINT	—	RI	TI	SSIO1	SSIO0	J1850ST
	7							**0**
LA, LD	NMI	EXTINT	—	RI	TI	SSIO1	SSIO0	—

Bit Number	Function
7:0†	Any set bit indicates that the corresponding interrupt is pending. The interrupt bit is cleared when processing transfers to the corresponding interrupt vector. **Bit Mnemonic** **Interrupt Description** NMI Nonmaskable Interrupt EXTINT EXTINT Pin Reserved — RI SIO Receive TI SIO Transmit SSIO1 SSIO 1 Transfer SSIO0 SSIO 0 Transfer J1850ST J1850 Status (LB only)

† Bit 5 is reserved on the 8XC196L*x* devices and bit 0 is reserved on the 87C196LA and 83C196LD. For compatibility with future devices, always write zeros to these bits.

Figure 4-4. Interrupt Pending 1 (INT_PEND1) Register

4.2.3 Peripheral Transaction Server Registers

Figures 4-5 and 4-6 illustrate the PTS interrupt select and service registers for the 8XC196L*x* microcontrollers.

PTSSEL Address: 0004H
Reset State: 0000H

The PTS select (PTSSEL) register selects either a PTS microcode routine or a standard interrupt service routine for each interrupt request. Setting a bit selects a PTS microcode routine; clearing a bit selects a standard interrupt service routine. In PTS modes that use the PTSCOUNT register, hardware clears the corresponding PTSSEL bit when PTSCOUNT reaches zero. The end-of-PTS interrupt service routine must reset the PTSSEL bit to re-enable the PTS channel.

	15							**8**
LA	—	EXTINT	—	RI	TI	SSIO1	SSIO0	—
	7							**0**
	—	—	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
	15							**8**
LB	—	EXTINT	—	RI	TI	SSIO1	SSIO0	J1850ST
	7							**0**
	J1850RX	J1850TX	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
	15							**8**
LD	—	EXTINT	—	RI	TI	SSIO1	SSIO0	—
	7							**0**
	—	—	—	EPA0	EPA1	EPA2	EPA3	EPA*x*

<table>
<tr><th>Bit Number</th><th>Function</th></tr>
<tr><td>14:0†</td><td>Setting a bit causes the corresponding interrupt to be handled by a PTS microcode routine.
The PTS interrupt vector locations are as follows:
<table>
<tr><th>Bit Mnemonic</th><th>Interrupt</th><th>PTS Vector</th></tr>
<tr><td>EXTINT</td><td>EXTINT pin</td><td>205CH</td></tr>
<tr><td>Reserved†</td><td>—</td><td>205AH</td></tr>
<tr><td>RI</td><td>SIO Receive</td><td>2058H</td></tr>
<tr><td>TI</td><td>SIO Transmit</td><td>2056H</td></tr>
<tr><td>SSIO1</td><td>SSIO 1 Transfer</td><td>2054H</td></tr>
<tr><td>SSIO0</td><td>SSIO 0 Transfer</td><td>2052H</td></tr>
<tr><td>J1850ST (LB)</td><td>J1850 Status</td><td>2050H</td></tr>
<tr><td>J1850RX(LB)</td><td>J1850 Receive</td><td>204EH</td></tr>
<tr><td>J1850TX(LB)</td><td>J1850 Transmit</td><td>204CH</td></tr>
<tr><td>AD (LA, LB)</td><td>A/D Conversion Complete</td><td>204AH</td></tr>
<tr><td>EPA0</td><td>EPA Capture/Compare Channel 0</td><td>2048H</td></tr>
<tr><td>EPA1</td><td>EPA Capture/Compare Channel 1</td><td>2046H</td></tr>
<tr><td>EPA2</td><td>EPA Capture/Compare Channel 2</td><td>2044H</td></tr>
<tr><td>EPA3</td><td>EPA Capture/Compare Channel 3</td><td>2042H</td></tr>
<tr><td>EPA<i>x</i>††</td><td>Multiplexed EPA</td><td>2040H</td></tr>
</table>
†† PTS service is not useful for shared interrupts because the PTS cannot readily determine the source of these interrupts.</td></tr>
</table>

† Bit 13 is reserved on the 8XC196L*x* devices and bits 6–8 are reserved on the 87C196LA and 83C196LD. For compatibility with future devices, write zeros to these bits.

Figure 4-5. PTS Select (PTSSEL) Register

PTSSRV

Address:	0006H
Reset State:	0000H

The PTS service (PTSSRV) register is used by the hardware to indicate that the final PTS interrupt has been serviced by the PTS routine. When PTSCOUNT reaches zero, hardware clears the corresponding PTSSEL bit and sets the PTSSRV bit, which requests the end-of-PTS interrupt. When the end-of-PTS interrupt is called, hardware clears the PTSSRV bit. The end-of-PTS interrupt service routine must set the PTSSEL bit to re-enable the PTS channel.

	15							8
LA	—	EXTINT	—	RI	TI	SSIO1	SSIO0	—
	7							**0**
	—	—	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
	15							**8**
LB	—	EXTINT	—	RI	TI	SSIO1	SSIO0	J1850ST
	7							**0**
	J1850RX	J1850TX	AD	EPA0	EPA1	EPA2	EPA3	EPA*x*
	15							**8**
LD	—	EXTINT	—	RI	TI	SSIO1	SSIO0	—
	7							**0**
	—	—	—	EPA0	EPA1	EPA2	EPA3	EPA*x*

<table>
<tr><th>Bits</th><th>Function</th></tr>
<tr><td>14:0†</td><td>A bit is set by hardware to request an end-of-PTS interrupt for the corresponding interrupt through its standard interrupt vector.
The standard interrupt vector locations are as follows:
<table>
<tr><th>Bit Mnemonic</th><th>Interrupt</th><th>Standard Vector</th></tr>
<tr><td>EXTINT</td><td>EXTINT pin</td><td>203CH</td></tr>
<tr><td>Reserved†</td><td>—</td><td>203AH</td></tr>
<tr><td>RI</td><td>SIO Receive</td><td>2038H</td></tr>
<tr><td>TI</td><td>SIO Transmit</td><td>2036H</td></tr>
<tr><td>SSIO1</td><td>SSIO 1 Transfer</td><td>2034H</td></tr>
<tr><td>SSIO0</td><td>SSIO 0 Transfer</td><td>2032H</td></tr>
<tr><td>J1850ST (LB)</td><td>J1850 Status</td><td>2030H</td></tr>
<tr><td>J1850RX (LB)</td><td>J1850 Receive</td><td>202EH</td></tr>
<tr><td>J1850TX (LB)</td><td>J1850 Transmit</td><td>202CH</td></tr>
<tr><td>AD (LA, LB)</td><td>A/D Conversion Complete</td><td>202AH</td></tr>
<tr><td>EPA0</td><td>EPA Capture/Compare Channel 0</td><td>2028H</td></tr>
<tr><td>EPA1</td><td>EPA Capture/Compare Channel 1</td><td>2026H</td></tr>
<tr><td>EPA2</td><td>EPA Capture/Compare Channel 2</td><td>2024H</td></tr>
<tr><td>EPA3</td><td>EPA Capture/Compare Channel 3</td><td>2022H</td></tr>
<tr><td>EPA<i>x</i>††</td><td>Multiplexed EPA</td><td>2020H</td></tr>
</table>
†† PTS service is not useful for shared interrupts because the PTS cannot readily determine the source of these interrupts.</td></tr>
</table>

† Bit 13 is reserved on the 8XC196L*x* devices and bits 6–8 are reserved on the 87C196LA and 83C196LD. For compatibility with future devices, write zeros to these bits.

Figure 4-6. PTS Service (PTSSRV) Register

5

I/O Ports

CHAPTER 5
I/O PORTS

The I/O ports of the 8XC196L*x* are functionally identical to those of the 8XC196J*x*. However, on the 87C196LA and LB, the reset state level of all 41 general-purpose I/O pins has changed from a weak logic “1” (wk1) to a weak logic “0” (wk0). This chapter outlines the differences between the 87C196LA, LB and the 8XC196K*x* controllers.

5.1 I/O PORTS OVERVIEW

Table 5-1 provides an overview of the 8XC196L*x* and 8XC196K*x* I/O ports.

Table 5-1. Microcontroller Ports

Port	Pins	Type	Configuration Options	Associated Peripheral or System Function
Port 0	8 (K*x*) 6 (CA, J*x*, L*x*)	Standard	Input-only	A/D converter (not supported on LD)
Port 1	8 (K*x*) 4 (CA, J*x*, L*x*)	Standard	Complementary Open-drain	EPA and timers
Port 2	8 (K*x*) 6 (CA, J*x*, L*x*)	Standard	Complementary Open-drain	J1850 (LB only), SIO, interrupts, bus control, clock gen.
Port 3	8	Memory mapped	Complementary Open-drain	Address/data bus
Port 4	8	Memory mapped	Complementary Open-drain	Address/data bus
Port 5	8 (K*x*) 3 (CA, J*x*, L*x*)	Memory mapped	Complementary Open-drain	Bus control, slave port
Port 6	8 (K*x*) 6 (CA, J*x*, L*x*)	Standard	Complementary Open-drain	EPA, SSIO

5.2 INTERNAL STRUCTURE FOR PORTS 1, 2, 5, AND 6 (BIDIRECTIONAL PORTS)

Figure 5-1 shows the logic for driving the output transistors, Q1 and Q2. Consult the datasheet for specifications on the amount of current that each port can source or sink.

In I/O mode (selected by clearing a port mode register bit*)*, the port data output and the port direction registers are input to the multiplexers. These signals combine to drive the gates of Q1 and Q2 so that the output is high, low, or high impedance.

In special-function mode (selected by setting a port mode register bit), SFDIR and SFDATA are input to the multiplexers. These signals combine to drive the gates of Q1 and Q2 so that the output is high, low, or high impedance. Special-function output signals clear SFDIR; special-function input signals set SFDIR. Even if a pin is to be used in special-function mode, you must still initialize the pin as an input or output by writing to the port direction register.

Resistor R1 provides ESD protection for the pin. Input signals are buffered. The standard ports use Schmitt-triggered buffers for improved noise immunity. Port 5 uses a standard input buffer because of the high speeds required for bus control functions. The signals are latched into the port pin register sample latch and output onto the internal bus when the port pin register is read.

The falling edge of RESET# turns on transistor Q3, which remains on for about 300 ns, causing the pin to change rapidly to its reset state. The active-low level of RESET# turns on transistor Q4, which weakly holds the pin low. Q4 remains on, weakly holding the pin low, until your software writes to the port mode register.

NOTE

P2.7 is an exception. After reset, P2.7 carries the CLKOUT signal (half the crystal input frequency) rather than being held low. When CLKOUT is selected, it is always a complementary output.

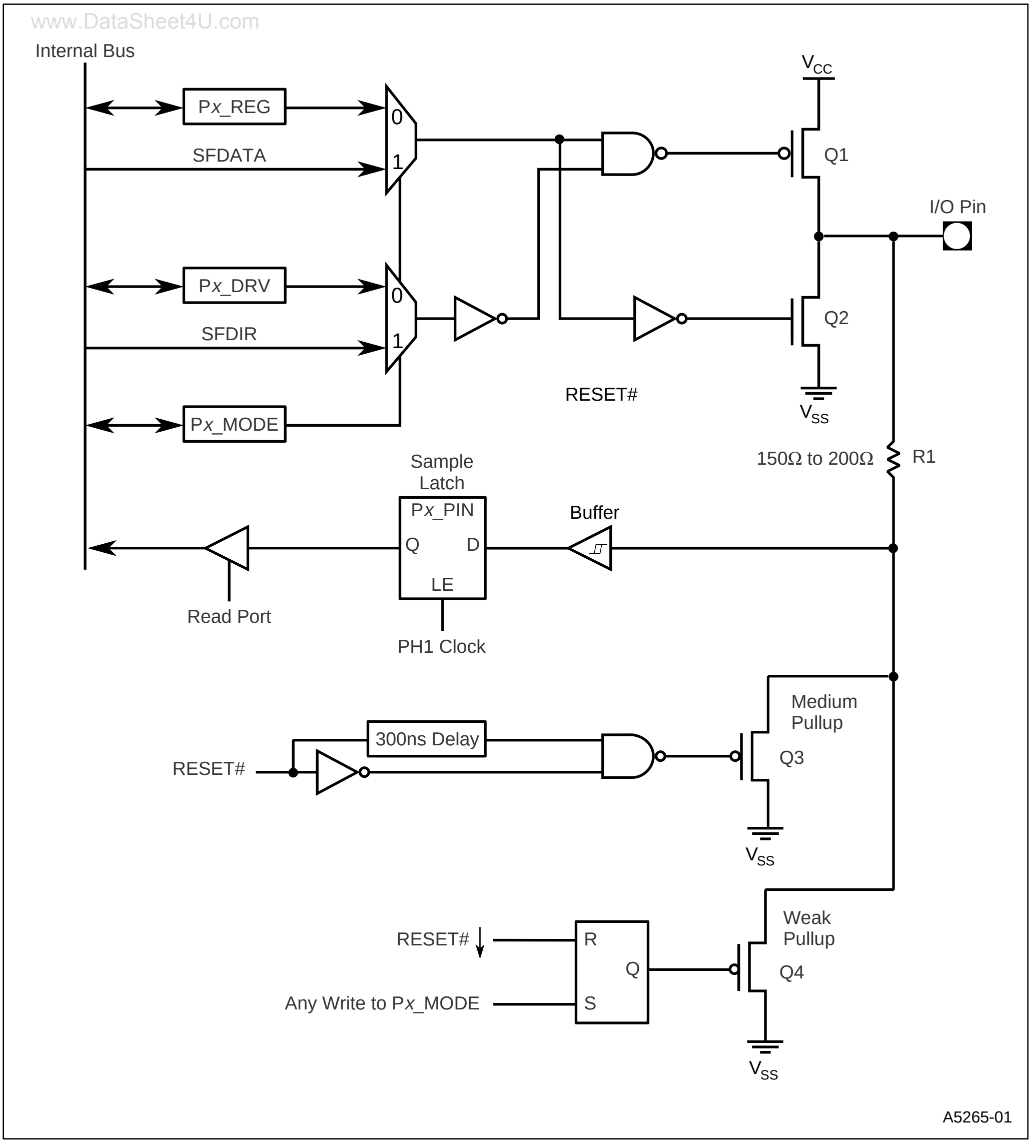


Figure 5-1. Ports 1, 2, 5, and 6 Internal Structure (87C196LA, LB Only)

5.2.1 Configuring Ports 1, 2, 5, and 6 (Bidirectional Ports)

Using the port mode register, you can individually configure each pin for port 1, 2, 5, and 6 to operate either as a general-purpose I/O signal (I/O mode) or as a special-function signal (special-function mode). In either mode, three configurations are possible: complementary output, high-impedance input, or open-drain output. The port direction and data output registers select the configuration for each pin. Complementary output means that the microcontroller drives the signal high or low. High-impedance input means that the microcontroller floats the signal. Open-drain output means that the microcontroller drives the signal low or floats it. For I/O mode, the port data output register determines whether the microcontroller drives the signal high, drives it low, or floats it. For special-function mode, the on-chip peripheral or system function determines whether the microcontroller drives the signal high or low for complementary outputs.

The pins for ports 1, 2, 5, and 6 are weakly pulled low during and after reset. Initializing the pins by writing to the port mode register turns off the weak pull-downs. To ensure that the ports are initialized correctly, follow this suggested initialization sequence:

1. Write to P*x*_DIR to configure the individual pins. Clearing a bit configures a pin as a complementary output. Setting a bit configures a pin as a high-impedance input or open-drain output.
2. Write to P*x*_MODE to select either I/O or special-function mode. Writing to P*x*_MODE (regardless of the value written) turns off the weak pull-downs. Even if the entire port is to be used as I/O (its default configuration after reset), **you must write to P*x*_MODE to ensure that the weak pull-downs are turned off**.
3. Write to P*x*_REG.

   For complementary output configurations:
   In I/O mode, write the data that is to be driven by the pins to the corresponding P*x*_REG bits. In special-function mode, the value is immaterial because the on-chip peripheral or system function controls the pin. However, you must still write to P*x*_REG to initialize the pin.

   For high-impedance input or open-drain output configurations:
   In I/O mode, write to P*x*_REG to either float the pin, making it available as a high impedance input, or pull it low. Setting the corresponding P*x*_REG bit floats the pin; clearing the corresponding P*x*_REG bit pulls the pin low. In special-function mode, if the on-chip peripheral uses the pin as an input signal, you must set the corresponding P*x*_REG bit so that the pin can be driven externally. If the on-chip peripheral uses the pin as an output signal, the value of the corresponding P*x*_REG bit is immaterial because the on-chip peripheral or system function controls the pin. However, you must still write to P*x*_REG to initialize the pin.

5.2.2 Special Bidirectional Port Considerations

This section outlines special consideration for using the pins of ports 1, 2, 5, and 6.

1. After reset, your software must configure the device to match the external system. This accomplished by writing appropriate configuration data into P*x*_MODE. Writing to P*x*_MODE not only configures the pins but also turns off the transistor that weakly holds the pins low. For this reason, even if your port is to be used as it is configured at reset, you should still write data into P*x*_MODE.
2. P2.6/TXJ1850 is the enable pin for ONCE mode. Because a high input during reset can cause the device to enter ONCE mode or a reserved test mode, caution must be exercised

in using this pin. Be certain that your system meets the V_{IH} specifications during reset to prevent inadvertent entry into ONCE mode or a test mode.

3. Following reset, P2.7/CLKOUT carries the strongly driven CLKOUT signal. It is not held low. When P2.7/CLKOUT is configured as CLKOUT, it is always a complementary output.

5.3 INTERNAL STRUCTURE FOR PORTS 3 AND 4 (ADDRESS/DATA BUS)

Figure 5-2 shows the logic of ports 3 and 4. Consult the datasheet for specifications on the amount of current ports 3 and 4 can source and sink.

During reset, the active-low level of RESET# turns off Q1 and Q2 and turns on transistor Q4, which weakly holds the pin low. Resistor R1 provides ESD protection for the pin. During normal operation, the device controls the port through BUS CONTROL SELECT, an internal control signal.

When the device needs to access external memory, it clears BUS CONTROL SELECT, selecting ADDRESS/DATA as the input to the multiplexer. ADDRESS/DATA then drives Q1 and Q2 as complementary outputs.

When external memory access is **not** required, the device sets BUS CONTROL SELECT, selecting P*x*_REG as the input to the multiplexer. P*x*_REG then drives Q1 and Q2. If P34_DRV is set, Q1 and Q2 are driven as complementary outputs. If P34_DRV is cleared, Q1 is disabled and Q2 is driven as an open-drain output requiring an external pull-up resistor. With the open-drain configuration (BUS CONTROL SELECT set and P34_DRV cleared) and P*x*_REG set, the pin can be used as an input. The signal on the pin is latched in the P*x*_PIN register. The pins can be read, making it easy to see which pins are driven low by the device and which are driven high by external drivers while in open-drain mode.

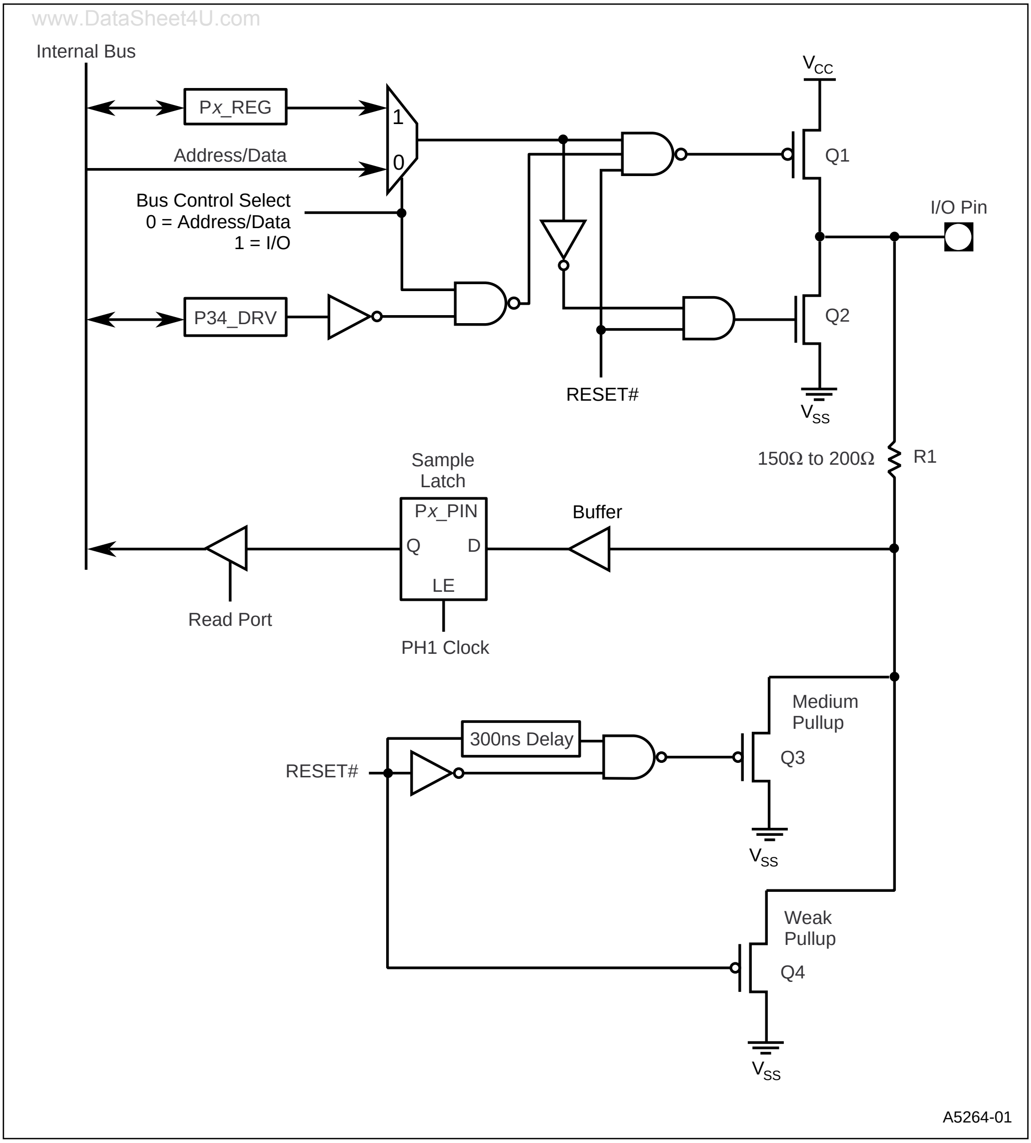


Figure 5-2. Ports 3 and 4 Internal Structure (87C196LA, LB Only)

6

Synchronous Serial I/O Port

CHAPTER 6
SYNCHRONOUS SERIAL I/O PORT

The synchronous serial I/O (SSIO) port on the 8XC196L*x* has been enhanced, implementing two new special function registers (SSIO0_CLK and SSIO1_CLK) that allow you to select the operating mode and configure the phase and polarity of the serial clock signals.

6.1 SSIO 0 CLOCK REGISTER

The SSIO 0 clock (SSIO_CLK) register selects the phase and polarity for the SC0 clock signal. In standard mode, SC0 is channel 0’s clock signal. In duplex and channel-select modes, SC0 is the common clock signal for both SSIO channels.

SSIO0_CLK

Address: 1FB5H
Reset State: 00H

The SSIO 0 clock (SSIO0_CLK) register configures the serial clock for channel 0. In standard mode, the SC0 is channel 0’s clock signal. In duplex and channel-select modes, SC0 is the common clock signal for both SSIO channels.

7							**0**
—	—	—	—	—	—	PHAS	POLS

Bit Number	**Bit Mnemonic**	**Function**
7:2	—	Reserved; for compatibility with future devices, write zeros to these bits.
1	PHAS	Phase and Polarity Select For normal transfers, these bits determine the idle state of the serial clock and select the serial clock signal edge on which the SSIO samples incoming data bits or shifts out outgoing data bits. These bits are ignored for handshaking transfers. Use SSIO0_ CON to select the type of data transfer (normal or handshaking) for channel 0. **For transmissions** PHAS POLS 0 0 low idle state; shift on falling edges 0 1 high idle state; shift on rising edges 1 0 low idle state; shift on rising edges 1 1 high idle state; shift on falling edges **For receptions** PHAS POLS 0 0 low idle state; sample on rising edges 0 1 high idle state; sample on falling edges 1 0 low idle state; sample on falling edges 1 1 high idle state; sample on rising edges
0	POLS	

Figure 6-1. SSIO 0 Clock (SSIO0_CLK) Register

For transmissions, SSIO0_CLK determines whether the SSIO shifts out data bits on rising or falling clock edges. For receptions, SSIO0_CLK determines whether the SSIO samples data bits on rising or falling clock edges.

6.2 SSIO 1 CLOCK REGISTER

SSIO1_CLK selects the SSIO mode of operation (standard, duplex, or channel-select), enables the channel-select master contention interrupt request, and selects the phase and polarity for the serial clock (SC1) for channels. In standard mode, use this register to configure the serial clock for channel 1.

SSIO1_CLK

Address: 1FB7H
Reset State: 00H

The SSIO 1 clock (SSIO1_CLK) register selects the SSIO mode of operation (standard, duplex, or channel-select), enables the channel-select master contention interrupt request, and selects the phase and polarity for the serial clock (SC1) for channel 1.

7							**0**
—	—	CHS	DUP	CONINT	CONPND	PHAS	POLS

Bit Number	Bit Mnemonic	Function
7:6	—	Reserved; for compatibility with future devices, write zeros to these bits.
5	CHS	These bits determine the SSIO operating mode. **CHS DUP** 0 0 standard mode 0 1 duplex mode 1 0 channel-select half-duplex mode (uses SD1 only) 1 1 channel-select full-duplex mode (uses both SD0 and SD1)
4	DUP	
3	CONINT	Master Contention Interrupt For channel-select master operations, the SSIO sets the master contention interrupt pending bit (CONPND) when the CHS# pin is externally activated. In a system with multiple masters, an external master activates the CHS# signal to request control of the serial clock. CONINT determines whether the SSIO sets both CONPND and the SSIO0 interrupt pending bit or only CONPND when the CHS# pin is externally activated. 0 = SSIO sets only CONPND 1 = SSIO sets both CONPND and the SSIO0 interrupt pending bit This bit is valid for channel-select master operations and ignored for all other operations.

Figure 6-2. SSIO 1 Clock (SSIO1_CLK) Register

SSIO1_CLK (Continued)

Address: 1FB7H
Reset State: 00H

The SSIO 1 clock (SSIO1_CLK) register selects the SSIO mode of operation (standard, duplex, or channel-select), enables the channel-select master contention interrupt request, and selects the phase and polarity for the serial clock (SC1) for channel 1.

7							0
—	—	CHS	DUP	CONINT	CONPND	PHAS	POLS

Bit Number	Bit Mnemonic	Function
2	CONPND	Master Contention Interrupt Pending For channel-select master operations, the SSIO sets this bit when the CHS# pin is externally activated. In a system with multiple masters, an external master activates the CHS# signal to request control of the serial clock. This bit is valid for channel-select master operations and ignored for all other operations.
1	PHAS	Phase and Polarity Select For normal transfers, these bits determine the idle state of the serial clock and select the serial clock signal edge that the SSIO samples incoming data bits or shifts out outgoing data bits. **For transmissions** PHAS POLS 0 0 low idle state; shift on falling edges 0 1 high idle state; shift on rising edges 1 0 low idle state; shift on rising edges 1 1 high idle state; shift on falling edges **For receptions** PHAS POLS 0 0 low idle state; sample on rising edges 0 1 high idle state; sample on falling edges 1 0 low idle state; sample on falling edges 1 1 high idle state; sample on rising edges These bits are ignored for duplex and channel-select modes; these modes use SC0 as the common clock signal. The SSIO0_CLK register contains the phase and polarity select bits for the SC0 clock signal. These bits are also ignored for handshaking transfers. Use SSIO1_ CON to select the type of data transfer (normal or handshaking) for channel 1.
0	POLS	

Figure 6-2. SSIO 1 Clock (SSIO1_CLK) Register (Continued)

For transmissions, SSIO1_CLK determines whether the SSIO shifts out data bits on rising or falling clock edges. For receptions, SSIO1_CLK determines whether the SSIO samples data bits on the rising or falling clock edges.

7

Event Processor Array

CHAPTER 7
EVENT PROCESSOR ARRAY

The EPA on the 8XC196L*x* is functionally identical to that of the 8XC196J*x*; however, the 8XC196L*x* has only two capture/compare channels without pins instead of four. In addition, the 83C196LD has no compare-only channels.

7.1 EPA FUNCTIONAL OVERVIEW

Table 7-1 lists the capture/compare (with and without pins) and compare-only channels for each device in the 8XC196L*x* and 8XC196K*x* families.

Table 7-1. EPA Channels

Device	Capture/Compare Channels With Pins	Capture/Compare Channels Without Pins	Compare-only Channels
8XC196LA, LB	EPA3:0 and EPA9:8	EPA7:6	COMP1:0
8XC196LD	EPA3:0 and EPA9:8	EPA7:6	—
87C196CA, 8XC196J*x*	EPA3:0 and EPA9:8	EPA7:4	COMP1:0
8XC196K*x*	EPA9:0	—	COMP1:0

The 8XC196L*x*'s EPA performs input and output functions associated with two timer/counters, timer 1 and timer 2, as depicted in Figures 7-1 and 7-2.

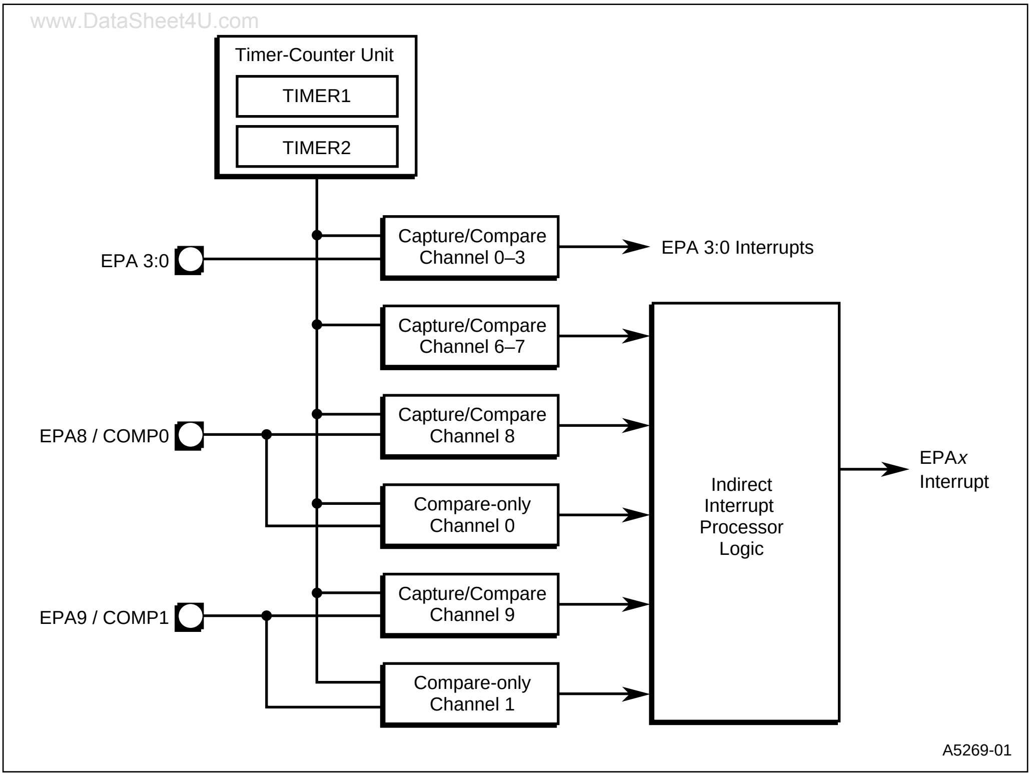


Figure 7-1. EPA Block Diagram (87C196LA, LB Only)

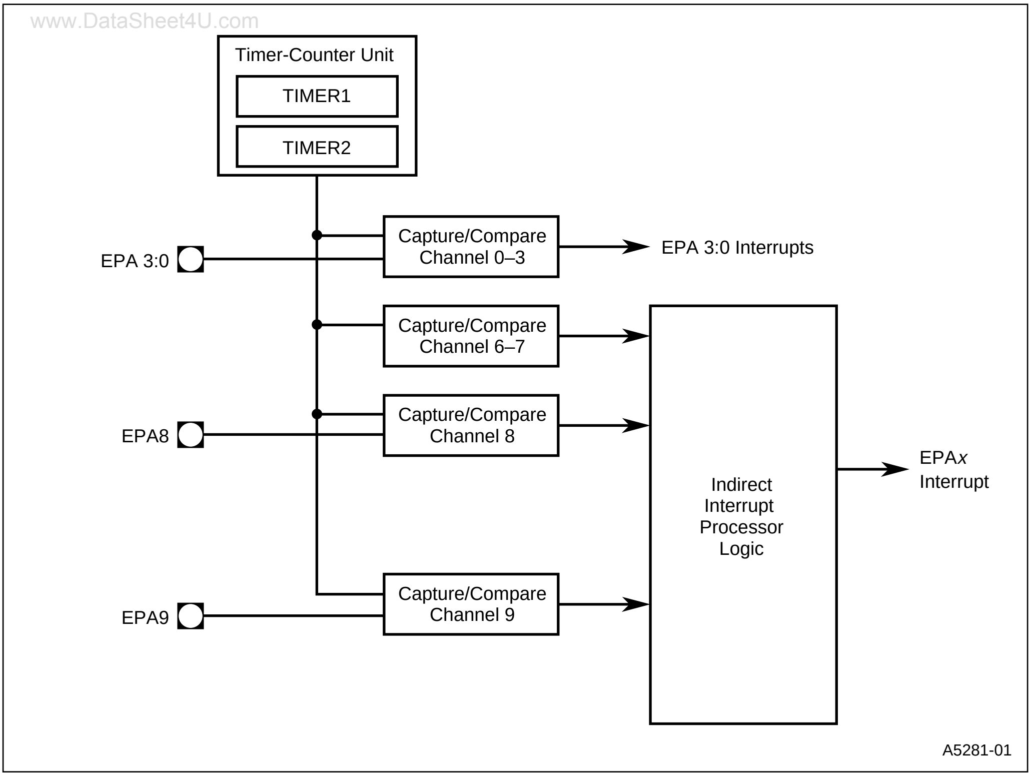

Timer-Counter Unit
TIMER1
TIMER2
EPA 3:0
Capture/Compare Channel 0–3
EPA 3:0 Interrupts
Capture/Compare Channel 6–7
EPA8
Capture/Compare Channel 8
Indirect Interrupt Processor Logic
EPAx Interrupt
EPA9
Capture/Compare Channel 9
A5281-01

Figure 7-2. EPA Block Diagram (83C196LD Only)

7.1.1 EPA Mask Registers

Figures 7-3 and 7-4 illustrate the EPA mask registers, EPA_MASK and EPA_MASK1, for the 8XC196L*x* microcontroller family.

EPA_MASK

Address: 1FA0H
Reset State: 0000H

The EPA interrupt mask (EPA_MASK) register enables or disables (masks) interrupts associated with the shared EPA*x* interrupt.

	15							**8**
Lx	—	—	EPA6	EPA7	EPA8	EPA9	OVR0	OVR1
	7							**0**
	0VR2	OVR3	—	—	—	—	OVR8	OVR9

Bit Number	Function
15:0[†]	Setting a bit enables the corresponding interrupt as a EPA*x* interrupt source. The shared EPA*x* interrupt is enabled by setting its interrupt enable bit in the interrupt mask register (INT_MASK.0 = 1).

[†] Bits 2–5 and 14–15 are reserved on the 8XC196L*x* device family. For compatibility with future devices, write zeros to these bits.

Figure 7-3. EPA Interrupt Mask (EPA_MASK) Register

EPA_MASK1

Address: 1FA4H
Reset State: 00H

The EPA interrupt mask 1 (EPA_MASK1) register enables or disables (masks) interrupts associated with the multiplexed EPA*x* interrupt.

7							**0**
—	—	—	—	COMP0[†]	COMP1[†]	OVRTM1	OVRTM2

Bit Number	Function
7:4	Reserved; for compatibility with future devices, write zeros to these bits.
3:0[†]	Setting a bit enables the corresponding interrupt as a multiplexed EPA*x* interrupt source. The multiplexed EPA*x* interrupt is enabled by setting its interrupt enable bit in the interrupt mask register (INT_MASK.0 = 1). [†] 87C196LA, LB only; reserved on 83C196LD.

Figure 7-4. EPA Interrupt Mask 1 (EPA_MASK1) Register

7.1.2 EPA Pending Registers

Figures 7-5 and 7-6 illustrate the EPA pending registers, EPA_PEND and EPA_PEND1, for the 8XC196L*x* microcontroller family.

EPA_PEND

Address: 1FA2H
Reset State: 0000H

When hardware detects a pending EPA6–9 or OVR0–3, 8–9 interrupt request, it sets the corresponding bit in the EPA interrupt pending register (EPA_PEND or EPA_PEND1). The EPAIPV register contains a number that identifies the highest priority, active, shared interrupt source. When EPAIPV is read, the EPA interrupt pending bit associated with the EPAIPV priority value is cleared.

	15							8
Lx	—	—	EPA6	EPA7	EPA8	EPA9	OVR0	OVR1
	7							0
	0VR2	OVR3	—	—	—	—	OVR8	OVR9

Bit Number	Function
15:0†	Any set bit indicates that the corresponding EPA*x* interrupt source is pending. The bit is cleared when software reads the EPA interrupt priority vector register (EPAIPV).

† Bits 2–5 and 14–15 are reserved on the 8XC196L*x* device family. For compatibility with future devices, write zeros to these bits.

Figure 7-5. EPA Interrupt Pending (EPA_PEND) Register

EPA_PEND1

Address: 1FA6H
Reset State: 00H

When hardware detects a pending EPA*x* interrupt, it sets the corresponding bit in the EPA interrupt pending register (EPA_PEND or EPA_PEND1). The EPAIPV register contains a number that identifies the highest priority, active, multiplexed interrupt source. When EPAIPV is read, the EPA interrupt pending bit associated with the EPAIPV priority value is cleared.

7							0
—	—	—	—	COMP0†	COMP1†	OVRTM1	OVRTM2

Bit Number	Function
7:4	Reserved; always write as zeros.
3:0†	Any set bit indicates that the corresponding EPA*x* interrupt source is pending. The bit is cleared when the EPA interrupt priority vector register (EPAIPV) is read. † 87C196LA, LB only; reserved on 83C196LD.

Figure 7-6. EPA Interrupt Pending 1 (EPA_PEND1) Register

7.1.3 EPA Interrupt Priority Vector Register

Figure 7-7 illustrates the EPA interrupt priority vector (EPAIPV) register for the 8XC196L*x* microcontroller family.

EPAIPV

Address: 1FA8H
Reset State: 00H

When an EPA*x* interrupt occurs, the EPA interrupt priority vector (EPAIPV) register contains a number that identifies the highest priority, active, multiplexed interrupt source (see Table 7-2).

EPAIPV allows software to branch via the TIJMP instruction to the correct interrupt service routine when EPA*x* is activated. Reading EPAIPV clears the EPA pending bit for the interrupt associated with the value in EPAIPV. When all the EPA pending bits are cleared, the EPA*x* pending bit is also cleared.

7							**0**
—	—	—	PV4	PV3	PV2	PV1	PV0

Bit Number	Bit Mnemonic	Function
5:7	—	Reserved; for compatibility with future devices, write zeros to these bits.
4:0	PV4:0	Priority Vector These bits contain a number from 01H to 14H corresponding to the highest-priority active interrupt source. This value, when used with the TIJMP instruction, allows software to branch to the correct interrupt service routine.

Figure 7-7. EPA Interrupt Priority Vector Register (EPAIPV)

Table 7-2. EPA Interrupt Priority Vectors

Value	Interrupt	Value	Interrupt	Value	Interrupt
14H	—	0DH	OVR1	06H	OVR8
13H	—	0CH	OVR2	05H	OVR9
12H	EPA6	0BH	OVR3	04H	COMP0†
11H	EPA7	0AH	—	03H	COMP1†
10H	EPA8	09H	—	02H	OVRTM1
0FH	EPA9	08H	—	01H	OVRTM2
0EH	OVR0	07H	—	00H	None

† 87C196LA, LB only; reserved on 83C196LD.

8

J1850 Communications Controller

CHAPTER 8 J1850 COMMUNICATIONS CONTROLLER

The J1850 communications controller manages communications between multiple network nodes. This integrated peripheral supports the 10.4 Kb/s VPW (variable pulse width) medium-speed class B in-vehicle network protocol. It also supports both the standard and in-frame response (IFR) message framing as specified by the *Society of Automotive Engineering* (*SAE) J1850* (revised May 1994) technical standards. Its lower cost per node makes it suitable for diagnostics and non-real-time data sharing in applications with high numbers of nodes. This chapter details the integrated J1850 controller and explains how to configure it.

8.1 J1850 FUNCTIONAL OVERVIEW

The integrated J1850 communications controller transfers messages between network nodes according to the J1850 protocol. The complete J1850 communications protocol solution includes an on-chip, J1850 digital-logic controller working with an external analog bus transceiver circuit. Figure 8-1 illustrates the J1850 protocol with the J1850 controller integrated on the 87C196LB 16-bit microcontroller and a standalone J1850 bus transceiver device. The example uses the *Harris HIP7020* as the remote transceiver device.

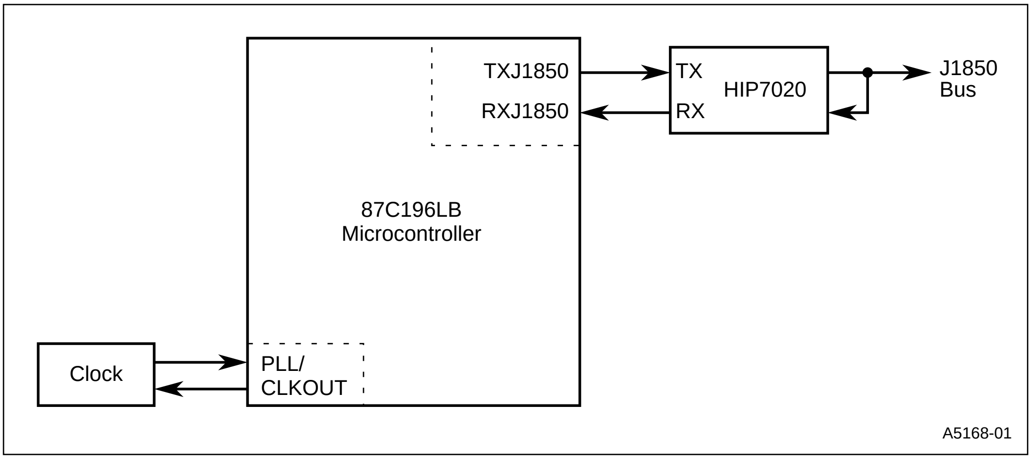


Figure 8-1. Integrated J1850 Communications Protocol Solution

The benefit of an integrated, J1850 protocol solution is threefold:

- Minimizes CPU overhead for reception and transmission of J1850 messages.
- Frees up serial and parallel communications ports for other purposes.
- Offers significant printed-circuit board area savings when compared with conventional standalone protocol devices.

The J1850 controller can handle network protocol functions including message frame sequencing, bit arbitration, in-frame response (IFR) messaging, error detection, and delay compensation.

The J1850 communications controller (Figure 8-2) consists of a control state machine (CSM), symbol synchronization and timing (SST) circuitry, six control and status registers, transmit and receive buffers, and an interrupt handler.

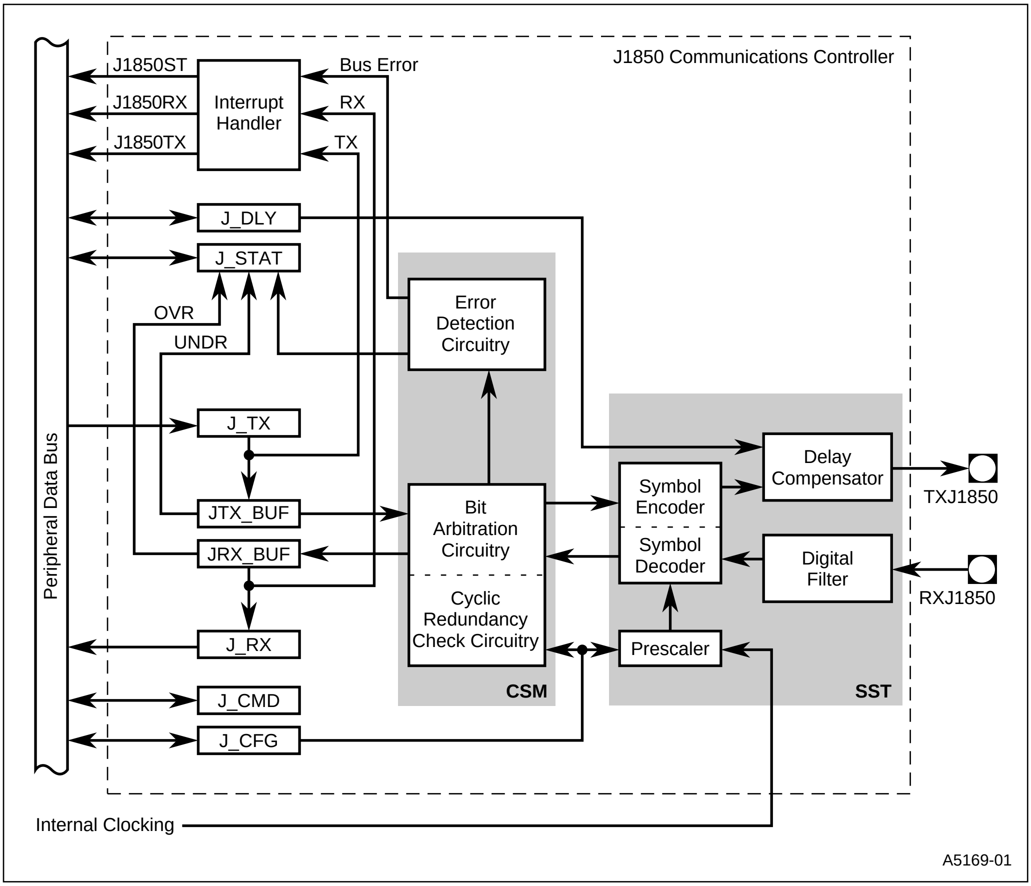


Figure 8-2. J1850 Communications Controller Block Diagram

8.2 J1850 CONTROLLER SIGNALS AND REGISTERS

Table 8-1 describes the J1850 controller's pins, and Table 8-2 describes the control and status registers.

Table 8-1. J1850 Controller Signals

Signal	Type	Description
RXJ1850	I	Receive Carries digital symbols from a remote transceiver to the J1850 controller.
TXJ1850	O	Transmit Carries digital symbols from the J1850 controller to a remote transceiver.

Table 8-2. Control and Status Registers

Mnemonic	Address	Description
J_CFG	1F54H	J1850 Configuration Program this byte register to specify the oscillator prescaler divisor, mode of operation, and normalization bit format. You must write to this register during the initialization sequence.
J_CMD	1F51H	J1850 Command Program this byte register to specify the number of bytes to be transmitted in the next message frame. This register also monitors the status of the message transmission in progress, and it can abort, ignore, or retry a message if necessary. Read this register to determine the status of transmissions in progress.
J_DLY	1F58H	J1850 Delay Compensation Program this byte register to define the length of the delay time through the external transceiver to compensate for the inherent propagation delays and to accurately resolve bus contention during arbitration. You must write to this register during the initialization sequence.
J_RX	1F52H	J1850 Receiver Read this byte register to receive data in byte increments from the J1850 bus to the microcontroller CPU. This register is buffered to allow for reception of a second data byte while the first data byte is being read.
J_STAT	1F53H	J1850 Status Read this byte register to determine the current status of the receive and transmit buffers and the J1850 interrupt sources. You can also determine bus status and in-frame response messaging status. All bits of this register are cleared when read, with the exception of the BUS_STAT bit.
J_TX	1F50H	J1850 Transmitter Program this byte register to transmit data in byte increments to the J1850 bus from the microcontroller CPU. This register is buffered to allow for writing of a second data byte while the first data byte is being shifted out.

Table 8-2. Control and Status Registers (Continued)

Mnemonic	Address	Description
INT_MASK	0008H	Interrupt Mask Bits 6 and 7 in this register enable and disable the J1850 receive and transmit interrupt requests, respectively.
INT_MASK1	0013H	Interrupt Mask 1 Bit 0 in this register enables and disables the J1850 bus error interrupt request.
INT_PEND	0009H	Interrupt Pending Bits 6 and 7 in this register, when set, indicate pending J1850 receive and transmit interrupt requests, respectively.
INT_PEND1	0012H	Interrupt Pending 1 Bit 0 in this register, when set, indicates a pending J1850 bus error interrupt request.
PTSSEL	0004H	PTS Select Bits 6, 7, and 8 of this word register select either a PTS service request or a standard interrupt service request for J1850TX, J1850RX, and J1850ST interrupts, respectively.
PTSSRV	0006H	PTS Service Bits 6, 7, and 8 of this word register are set by hardware to request an end-of-PTS interrupt for the J1850.

8.3 J1850 CONTROLLER OPERATION

This section describes the control state machine (which contains the cyclic redundancy check generator) and the symbol synchronization and timing circuitry for J1850 transmissions and receptions.

8.3.1 Control State Machine

The control state machine (CSM) represents the engine of the digital circuitry portion of the J1850 communications controller. The CSM handles all message framing for standard and in-frame response (IFR) messaging, data validation, bus contention, bit arbitration, and error detection.

8.3.1.1 Cyclic Redundancy Check Generator

The *cyclic redundancy check (CRC)* generator circuitry calculates and checks the CRC byte generated for both transmitted and received standard messages as specified by *SAE J1850* protocol specification for class B in-vehicle networks. The CRC calculation is a code byte of information that verifies the proper reception or transmission of your message. The calculated CRC code byte is always appended as the last byte of your transmitted message. On reception, the calculated CRC checksum byte always results in a value of C4H for valid messages. An invalid CRC checksum during reception signals the presence of an error in your incoming message, which immediately sets the J1850 bus error (J1850BE) bit in the J_STAT register (Figure 8-19 on page 8-21).

8.3.1.2 Bus Contention

Bus contention arises when multiple nodes attempt to access and transmit message frames across the J1850 bus simultaneously. This creates a conflict on the bus. The recognition of conflicting symbols or bits on the bus is referred to as *contention detection*. For example, if a node observes a difference between a symbol it transmits to the J1850 bus and the symbol that it detects on the bus, that node has detected contention to the transmission of its message frame. Only one message frame from one node vying for the bus wins arbitration on each symbol or bit of its frame. This winning message frame does not experience or detect contention. The message frames that were not awarded arbitration will experience contention.

8.3.1.3 Bit Arbitration

A *bit arbitration* scheme is used to resolve such conflicts as bus contention. The J1850 protocol uses the carrier sense multiple access (CSMA) bit arbitration scheme. Bit arbitration is the process of settling conflicts that occur when multiple nodes attempt to transmit one bit or symbol at a time across a single bus. A symbol is simply a timing-level formatted bit. By definition, a node that detects contention has lost arbitration and will discontinue transmitting any further symbols remaining in its message frame. Remaining nodes vying for the bus will continue to send their symbols until the next instance of contention is detected or arbitration is awarded. This process continues until a complete message frame from one node has been transmitted. For details on this arbitration scheme, refer to the “Bit Arbitration Example” on page 8-7.

8.3.1.4 Error Detection

The J1850 controller’s error detection logic monitors the bus for four error conditions, and sets the J1850BE interrupt pending bit in the J_STAT register if an error occurs. The following list describes each error type:

- CRC error — the calculated CRC checksum received on incoming messages has a value other than C4H (the expected value for all received message frames).
- bus symbol timing error — the symbol stream on the J1850 bus contains an invalid symbol. An invalid symbol is any signal that is between 8 µs and 34 µs in duration.
- incomplete byte error — an EOD/EOF symbol occurred,but was not on a byte boundary; the number of bits recieved was not a multiple of eight.
- no echo — the message is transmitted; however, the transmission’s echo back through the feedback loop to the receiver has not been detected within the allowable 60 µs window.

8.3.2 Symbol Synchronization and Timing Circuitry

The symbol synchronization and timing (SST) circuitry consists of a clock prescaler, digital filter, delay compensation circuitry, and synchronization and symbol encoding/decoding circuitry. The SST supports Huntzicker encoding of symbols, which entails 10.4 Kb/s variable pulse-width (VPW) operation for valid edge detection on message receptions.

8.3.2.1 Clock Prescaler

Because the 87C196LB microcontroller can operate at a variety of input frequencies (F_{XTAL1}), the clock prescaler circuitry is used to provide a single, internal clock frequency (f/2) to ensure that the J1850 peripheral is clocked at the proper operating frequency. This is accomplished through the programmable clock prescaler bits, PRE1:0 in the J_CFG register (Figure 8-17 on page 8-18). The prescale bits support input frequencies of 8, 12, 16, and 20 MHz on the XTAL1 pin. With the phase-locked loop (PLL) circuitry enabled, the prescale bits can support input frequencies of 4, 6, 8, and 10 MHz on the XTAL1 pin.

Table 8-3 details the relationships between the input frequency, the configuration of PLL, the internal clock frequency, and the prescaler bits.

Table 8-3. Relationships Between Input Frequency, PLL, and Prescaler Bits

F_{XTAL1}		Internal Clock Frequency (f/2)	PRE1	PRE0
PLL Disabled	PLL Enabled			
8 MHz	4 MHz	4 MHz	0	0
12 MHz	6 MHz	6 MHz	0	1
16 MHz	8 MHz	8 MHz	1	0
20 MHz	10 MHz	10 MHz	1	1

8.3.2.2 Digital Filter

To automatically reject noise spikes of 8 µs or less in duration, the J1850 controller uses a digital filter between the RXJ1850 input pin and the symbol synchronization logic.

A *noise spike* is defined as an active or passive state pulse that is shorter in duration than a valid receive symbol at that state. A valid receive symbol is at least 34 µs in duration. Any symbol captured on the bus between 8 µs and 34 µs in duration is considered invalid and is flagged by the J_STAT register as a bus-symbol timing error.

8.3.2.3 Delay Compensation

Because the digital portion of the J1850 protocol is integrated onto the microcontroller and physically separated from the transceiver and J1850 bus, control over critical timing parameters of various manufacturers' remote transceivers is required. The delay compensation circuitry addresses this requirement by providing the flexibility to compensate for propagation delay and pulse-width variations among various transceivers. The compensation circuitry synchronizes itself to the leading edge of each input symbol, which allows for accurate detection of bus contention during bit arbitration. The delay compensation is programmable through the J_DLY register (Figure 8-18 on page 8-20).

8.3.2.4 Symbol Encoding and Decoding

The J1850 protocol supports the Huntzicker encoding method, which is based on *variable pulse-width (VPW)* bus modulation. VPW modulation is a forced high/low symbol transition formatting scheme that tracks the duration between two consecutive transitions and the level of the bus, active or passive (Figure 8-3).

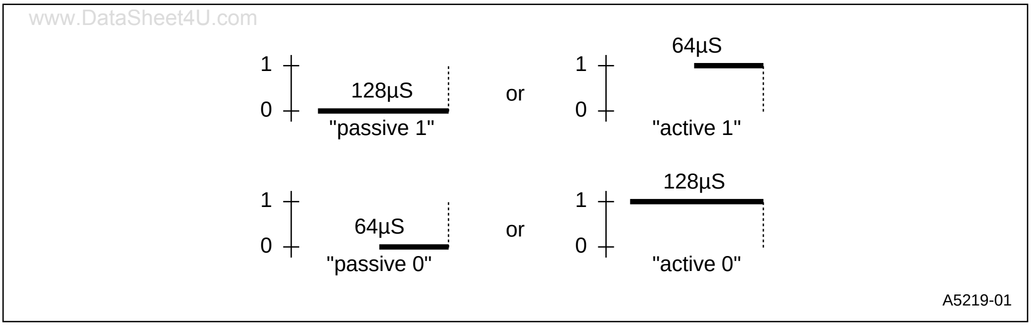


Figure 8-3. Huntzicker Symbol Definition for J1850

A symbol is defined as a timing-level formatted bit. The VPW symbol timing requirements stipulate that there is one symbol per transition and one transition per symbol. This ensures that a message frame will always result in a uniform square waveform of varying level durations. Figure 8-4 depicts a typical Huntzicker formatted data byte of hex value CCH.

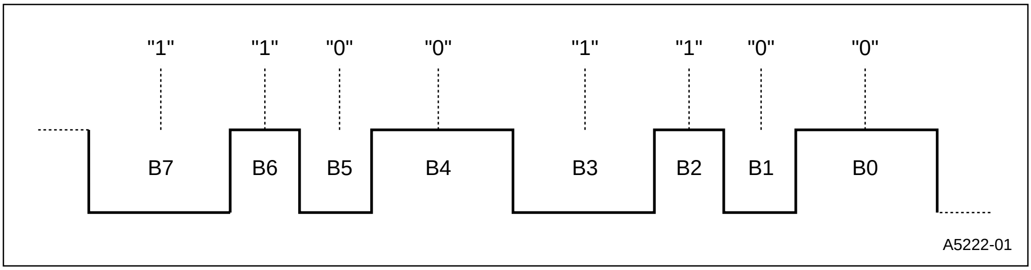


Figure 8-4. Typical VPW Waveform

Bits 7 and 3 carry logic level 1 data; however, they are represented by a passive-level symbol in keeping with the VPW requirements. Bits 4 and 0 carry logic level 0 data and are represented by an active-level symbol.

8.3.3 Bit Arbitration Example

The drive capacity of each symbol establishes the priority for arbitration. By definition, an active bus level is a driven state, and a passive bus level is a non-driven, or idle, state. A driven bus state is always given priority over an idle bus in arbitration. An “active 0” state has priority over an “active 1” state in arbitration, because the “active 0” state is driven over a longer duration, 128 µs versus the “active 1” state’s drive time of 64 µs. Similarly, a “passive 0” state has priority over a “passive 1” state, because the “passive 0” state comes out of its idle state in a shorter period of time, 64 µs versus the “passive 1” state’s idle time of 128 µs.

For example, Figure 8-5 illustrates four nodes vying for the bus. Node B is the first node to discontinue transmitting when it attempts to transmit a “passive 1” symbol onto the bus. At the point of arbitration, nodes A, C, and D are all transmitting an “active 0” symbol, thus the idle state of the “passive 1” symbol is overruled in favor of the driven state of the “active 0” symbol.

Node C is the next node to discontinue transmitting when it attempts to take control of the bus by transmitting an “active 1” symbol. However, nodes A and D maintain control by continuing to drive the bus with an “active 0” symbol.

Finally, node D discontinues transmitting when its attempt to hold the bus in an idle state is overruled by the driven state of the “active 1” symbol on node A. Thus, node A is awarded arbitration.

The busline signal, detected on the bus by the receiver, reflects node A’s message, as this is the only node that did not experience contention.

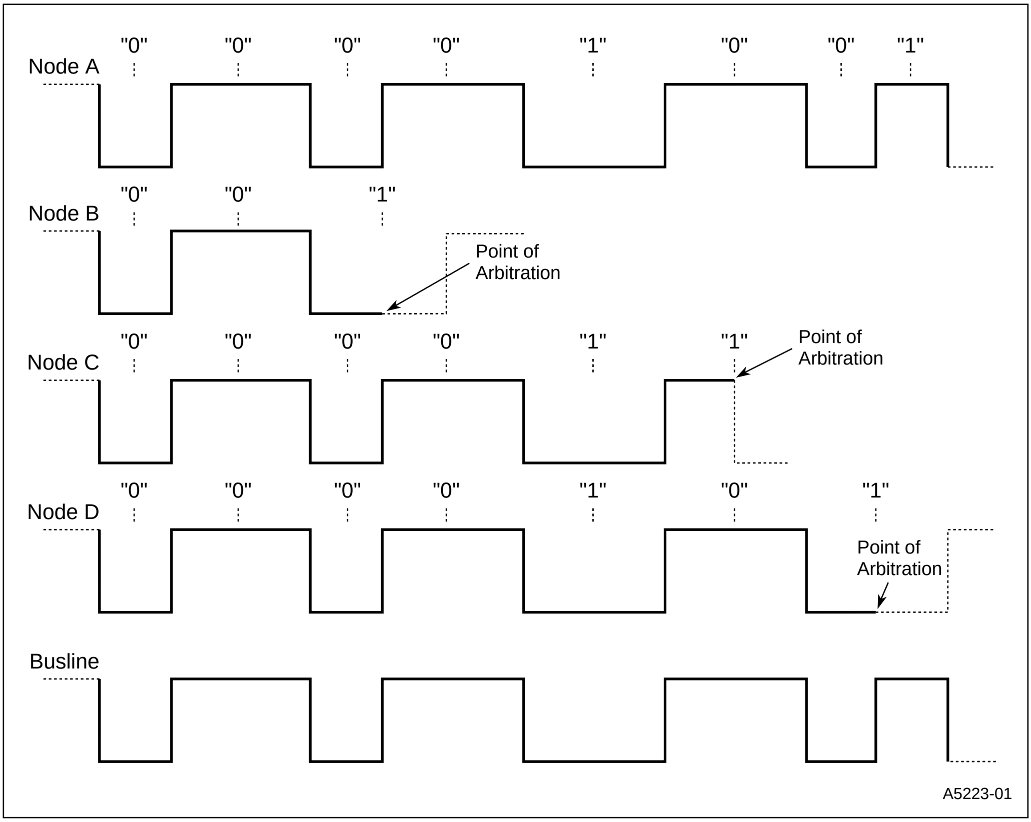
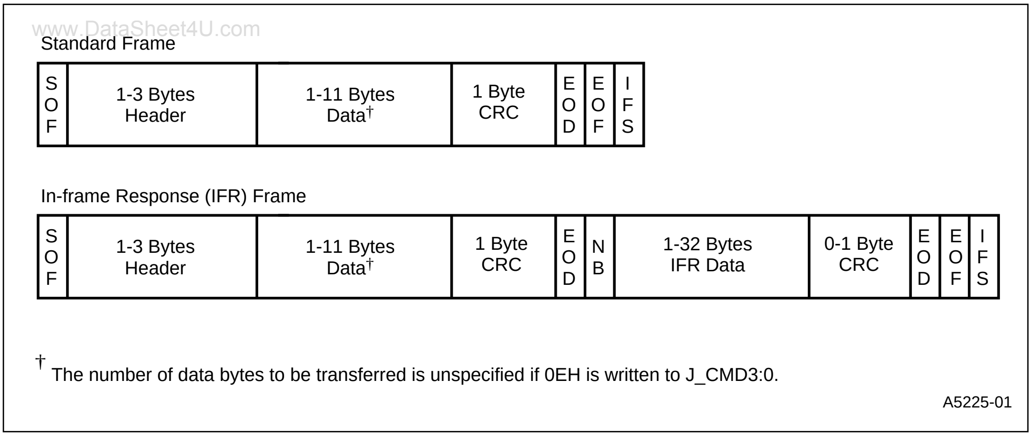


Figure 8-5. Bit Arbitration Example

8.4 MESSAGE FRAMES

A message transmission or reception is transferred within a message frame that adds control and error-detection bits to the content of the message. The frame for an IFR message differs slightly from that for a standard message, but they contain similar information (Figure 8-6).



† The number of data bytes to be transferred is unspecified if 0EH is written to J_CMD3:0.

Figure 8-6. J1850 Message Frames

A standard message frame is initiated by the responder and contains no more than 11 data bytes to be transmitted. An IFR message is a request initiating the recipient(s) to respond by transmitting data within the same frame. The following subsections describe each of the messaging forms.

8.4.1 Standard Messaging

A standard message frame can best be described as a "send mode only" format that is initiated by the responder either to request information or to reply to a received message from a remote node. In addition to the actual data that is being transmitted, the standard message is composed of a header (1–3 bytes), a CRC byte, and a series of start and end symbols.

8.4.1.1 Header

The header provides general information on the physical network and the necessary interface requirements. For a complete description of the header, refer to the *Society of Automotive Engineering* (*SAE) J1850* specifications (revised May 1994).

8.4.1.2 CRC Byte

The CRC byte, calculated through the cyclic redundancy check generator, is a checksum value that verifies the accuracy of the data message transmitted onto the bus. The CRC byte is appended to all data messages and optionally appended to IFR response messages. Upon reception, the CRC byte is compared with the value C4H. If the values match, the transmitted message is valid; otherwise, it is invalid, and an error flag in the J_STAT register is set.

8.4.1.3 Normalization Bit

The normalization bit (NB), found only in IFR messaging, defines the start of the IFR message response data. The NB is triggered by bit J_CMD.6 and is transmitted after an end-of-data (EOD) symbol is detected on the bus. The timing format of the NB is assigned by the J_CFG register

(J_CFG.7) and considers whether the IFR message response has a CRC byte appended. Figure 8-7 depicts the *SAE* preferred, active-level state bit format timing for the NB.

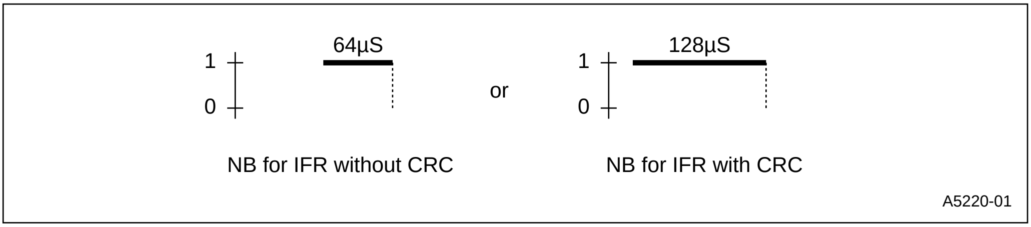


Figure 8-7. Huntzicker Symbol Definition for the Normalization Bit

8.4.1.4 Start and End Message Frame Symbols

Five symbols are used to mark the start and end of a message frame and to allow the J1850 bus to properly recognize the interruption of a message transmission or reception. Figure 8-8 illustrates the formats and their respective timing.

The following is a description of each symbol:

- start of frame (SOF) — this symbol signals the start of a message frame. This is an active-level state symbol only and appears once per frame.
- end of data (EOD) — this symbol signals the end of the data transmission. This is a passive-level state symbol only. It appears twice in IFR messaging: at the end of the initial request data field and at the end of the IFR data field.
- end of frame (EOF) — this symbol signals the end of a message frame and returns the bus to an idle state. This is a passive-level state symbol only. It appears once per frame.
- in-frame separation (IFS) — the timing of this symbol allows for proper synchronization of multiple nodes during back-to-back transmissions. Nodes contending for the bus must comply with one of two conditions before transmitting:
  - — wait for the IFS minimum timing to expire
  - — wait for a rising edge on the bus after the EOF minimum timing has expired
- break (BRK) — this symbol signals an interruption during a bus transmission. At the point of termination, all nodes are reset. This is an active-level state symbol.

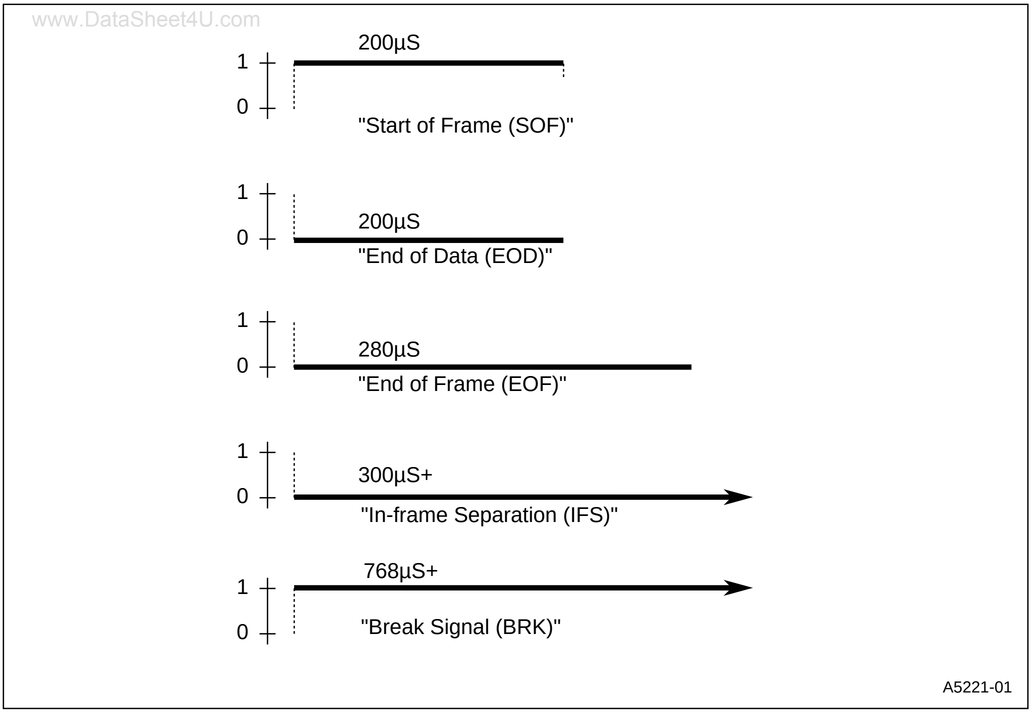


Figure 8-8. Definition for Start and End of Frame Symbols

Table 8-4 details the symbol timing characteristics supported by the 87C196LB.

Table 8-4. Huntzicker Symbol Timing Characteristics

Name	Symbol	Bus Level	T_{TX}min	T_{TX}nom	T_{TX}max	T_{RX}min	T_{RX}max	Units
Logic Level 0	0	Passive	60	64	68	34	<96	µs
		Active	122	128	134	96	<163	µs
Logic Level 1	1	Passive	122	128	134	96	<163	µs
		Active	60	64	68	34	<96	µs
Start of Frame	SOF	Active	193	200	207	163	<239	µs
End of Data	EOD	Passive	193	200	207	163	<239	µs
End of Frame	EOF	Passive	271	280	289	239	<300	µs
In-frame Separation	IFS	Passive	>300	—	—	>300	—	µs
Break	BRK	Active	768	—	—	>239	—	µs

NOTE: Timings are based on the standard bus rate of 10.4 Kb/s. When operating in 4x mode, the bus rate becomes 41.6 Kb/s and all symbol timings are one fourth that shown.

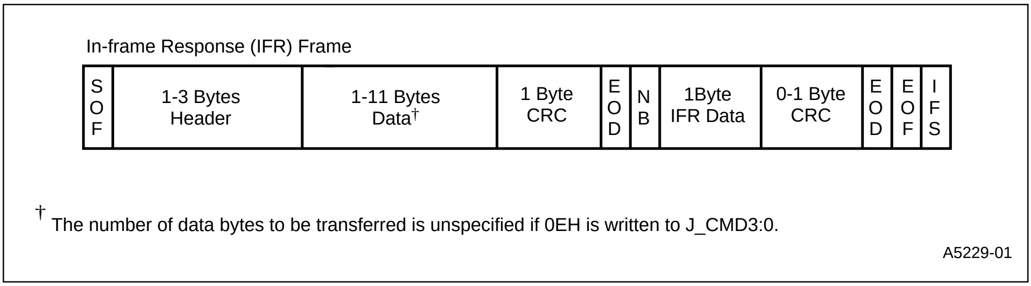
8.4.2 In-frame Response Messaging

There are three types of in-frame response (IFR) message framings: type 1 (a single byte from a single responder), type 2 (a single byte from multiple responders), and type 3 (multiple bytes from a single responder). Like the standard message frame, the IFR frame is composed of header, data, and CRC bytes, and a series of start and end symbols. Unlike the standard message frame, the actual length of the IFR message frame will differ based on the desired response.

Consider the following example: a system's controller (the requestor) requests an information update from each of four nodes (the responders) in the system. With type 1 messaging, the controller can receive a limited information update if it sends out four separate transmissions. With type 2 messaging, the controller can receive a limited information update by sending one message. With type 3 messaging, the controller can receive unlimited information; however, it will require four separate transmissions. The following subsections detail this example for the three IFR messaging types.

8.4.2.1 IFR Messaging Type 1: Single Byte, Single Responder

No IFR messaging type carries a distinct advantage or disadvantage over the other messaging types. IFR messaging type 1 (Figure 8-9) is ideal for use when requesting small amounts of information from a single source in your system. In the above example, suppose you want to know how many pounds of pressure each of the four remote node sites experienced after the controller sent out a request to each node sensor to exert a given amount of pressure. If you use type 1 messaging, the controller will send four separate serial messages to the remote node sites in the system and wait for their responses. Keeping the data timing a constant, the CPU overhead of transmitting these messages alone amounts to a minimum of 4.96 ms (refer to Table 8-4 on page 8-11 for all symbol timings).

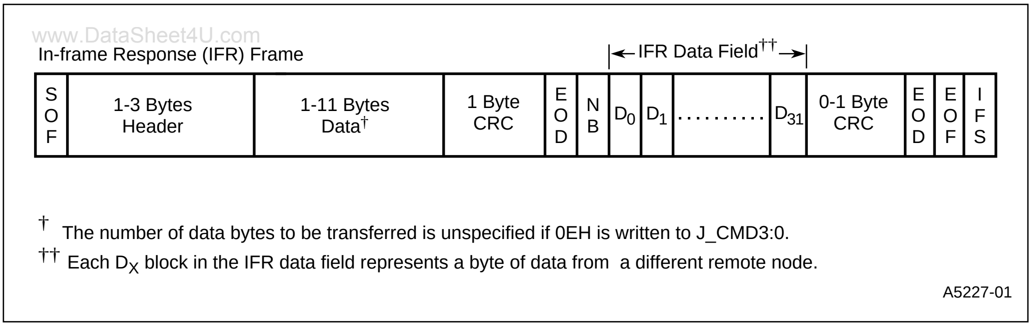


† The number of data bytes to be transferred is unspecified if 0EH is written to J_CMD3:0.

Figure 8-9. IFR Type 1 Message Frame

8.4.2.2 IFR Messaging Type 2: Single Byte, Multiple Responders

When response time is the highest consideration, IFR messaging type 2 is desirable. IFR type 2 messaging can monitor up to 32 remote nodes on a given request (see Figure 8-10). While it allows only one byte of information per response, in many cases a single byte of information is more than adequate. In our example, suppose that each node sensor detected a pressure of 75 P.S.I. (pounds per square inch). The response (the value 75) would take a single byte, 46H, to communicate the reply. The maximum overhead required is 1.24 ms, or one fourth the time it would take type 1 messaging to achieve the same results.



† The number of data bytes to be transferred is unspecified if 0EH is written to J_CMD3:0.

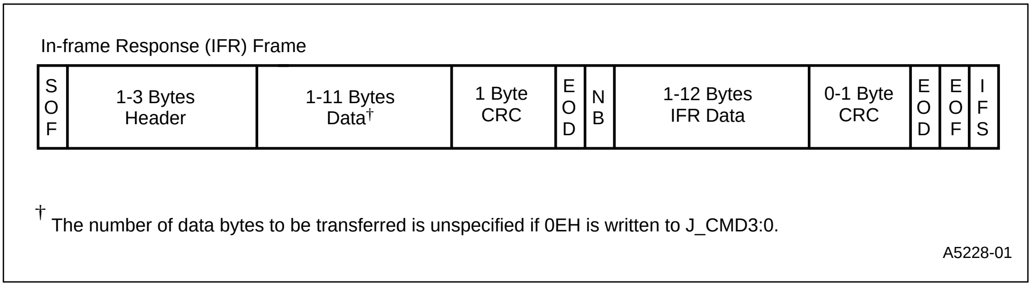
†† Each D_X block in the IFR data field represents a byte of data from a different remote node.

A5227-01

Figure 8-10. IFR Type 2 Message Frame

8.4.2.3 IFR Messaging Type 3: Multiple Bytes, Single Responder

IFR messaging type 3 (Figure 8-11) is ideal for requesting large amounts of information from a single source in your system. You can compile up to 12 bytes of data from a remote node on a single request. In our example, for the same amount of CPU overhead as IFR type 1 messaging exhausted (4.96 ms), you can gather up to twelve times as much information.



† The number of data bytes to be transferred is unspecified if 0EH is written to J_CMD3:0.

A5228-01

Figure 8-11. IFR Type 3 Message Frame

8.5 TRANSMITTING AND RECEIVING MESSAGES

The J1850 controller can transmit and receive messages in either standard or IFR form.

8.5.1 Transmitting Messages

To transmit a standard message, prepare the message in register RAM and then write it to the J1850 transmit (J_TX) register (Figure 8-12) one byte at a time.

J_TX

Address:	1F50H
Reset State:	00H

The J1850 transmitter (J_TX) register transfers data in byte increments to the J1850 bus from the microcontroller CPU. This register is buffered to allow for transmission of a second data byte while the first data byte is being shifted out. This byte register can be read or written, and is addressable through *windowing*.

7	0
Transmit Byte	

Bit Number	Bit Mnemonic	Function
7:0	DB7:0	Data Bits These eight bits compose the data byte to be transmitted to the J1850 bus.

Figure 8-12. J1850 Transmitter (J_TX) Register

Transmitting the message requires that you first program the J1850 command (J_CMD) register to specify the number of bytes you want to transfer across the J1850 bus. The number of bytes specified must include the header byte(s). After the start of frame (SOF) symbol is put on the bus, the first header byte is transferred to J_TX for transmission. This byte will automatically be transferred into the J1850 transmit buffer (JTX_BUF) and the second byte of the message frame will be written to J_TX. The transfer of the first byte to JTX_BUF triggers the transmission process and generates the J1850 transmission (J1850TX) interrupt (if it is enabled), signaling that J_TX is available for another byte (Figure 8-13).

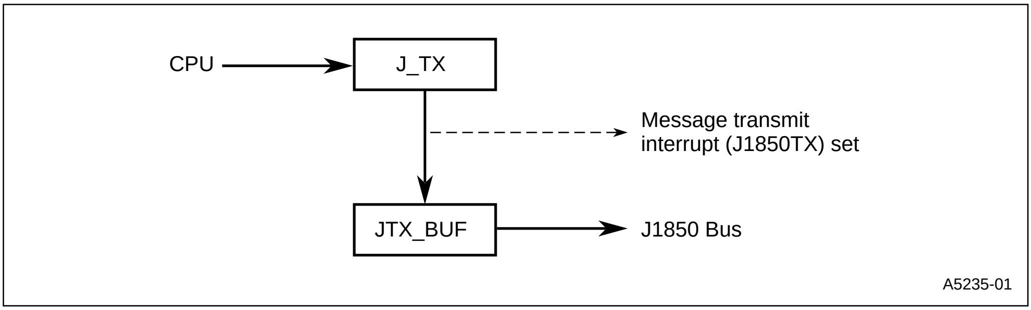


Figure 8-13. J1850 Transmit Message Structure

After the byte in JTX_BUF is transmitted, the byte residing in J_TX is automatically shifted into JTX_BUF, freeing J_TX for another byte. This process continues until the CSM has resolved the number of message bytes (MSG3:0) programmed into the J_CMD register.

If the last message byte being transmitted is shifted out before the MSG*x* count expires, a J1850ST core interrupt is generated and the OVR_UNDR (J_STAT.3) bit records a transmitter underflow error in the J_STAT register.

NOTE

An overrun condition can occur on transmission if the transmit buffer, JTX_BUF, is overwritten.

8.5.2 Receiving Messages

For a message reception, after a SOF is detected on the bus, the controller starts to shift data symbols into the J1850 receive buffer (JRX_BUF) until an entire data byte has been received. This byte is automatically transferred into the J1850 receive (J_RX) register (Figure 8-14) and the subsequent byte is written into the empty JRX_BUF.

J_RX

Address: 1F52H
Reset State: 00H

The J1850 receiver (J_RX) register transfers received data in byte increments from the J1850 bus to the microcontroller CPU. This register is buffered to allow for reception of a second data byte while the first data byte is being read. This byte register can be read or written, and is addressable through *windowing*.

7	0
Receive Byte	

Bit Number	Bit Mnemonic	Function
7:0	DB7:0	Data Bits These eight bits compose the last data byte received from the J1850 bus.

Figure 8-14. J1850 Receiver (J_RX) Register

The transfer of the first byte to J_RX triggers the reception process and generates the J1850 reception (J1850RX) interrupt (if it is enabled), signaling that JRX_BUF is available for another byte (Figure 8-15).

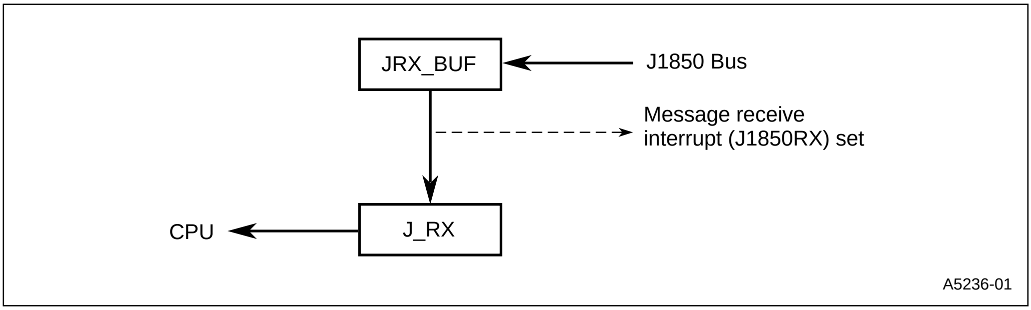


Figure 8-15. J1850 Receive Message Structure

After J_RX is read, the byte residing in JRX_BUF is automatically shifted into J_RX, freeing JRX_BUF for another reception. This process continues until an end of data (EOD) symbol is encountered.

If a third byte is received before J_RX is read, a J1850ST core interrupt is generated and the OVR_UNDR (J_STAT.3) bit records a receiver overrun error in the J_STAT register.

8.5.3 IFR Messages

In-frame response (IFR) messaging is identical in setup to standard messaging for both transmission and reception. It uses the same registers to configure, communicate, and control data. The difference is that the requestor initiating the IFR message sequence writes the message specifying a response from either one or more nodes in the system. Framing a message in this manner bypasses needless CPU overhead that can result from lengthy EOF symbols, and it gives you a faster response to the information you are accessing from remote sites in your system. (Refer to "In-frame Response Messaging" on page 8-12 for a detailed explanation).

8.6 PROGRAMMING THE J1850 CONTROLLER

This section explains how to configure the J1850 controller. Several registers combine to control the configuration: the command register, the configuration register, the delay compensation register, and the status register.

Programming the J1850 controller requires that you first program the configuration and delay registers during initialization. You need to program these two registers only once per initialization sequence.

After initialization, you must first program the command register, followed by either the receive or transmit register, and then the status register.

8.6.1 Programming the J1850 Command (J_CMD) Register

The J1850 command register (Figure 8-16) determines the messaging type, specifies the number of bytes to be transmitted in the next message frame, and updates the status of the message transmission in progress.

J_CMD

Address:	1F51H
Reset State:	00H

The J1850 command (J_CMD) register determines the messaging type, specifies the number of bytes to be transmitted in the next message frame, and updates the status of the message transmission in progress. This byte register can be directly addressed through *windowing*. You must write to this register prior to transmitting every message.

7							0
AUTO	IFR	IGNORE	ABORT	MSG3	MSG2	MSG1	MSG0

Bit Number	Bit Mnemonic	Function
7	AUTO	Automatic Transmit Retry This bit, when arbitration is lost on the first byte of your message, prompts the transmitter to automatically retry until the byte is successfully transmitted. Automatic retry applies only to the first byte. 0 = normal operation 1 = enable automatic retry
6	IFR	In-frame Response Indicator This bit signals that a normalization bit (NB) is to be sent after an end-of-data symbol is detected on the bus and that the subsequent byte written to the J1850 transmitter (J_TX) register is an in-frame response (IFR). 0 = standard messaging 1 = next byte written to J_TX is an IFR
5	IGNORE	Ignore Incoming Message This bit instructs the bus to ignore the incoming message until an EOF symbol is detected. The bit is cleared after an EOF symbol is detected. 0 = normal operation 1 = ignore incoming message
4	ABORT	Abort Transmission This bit aborts any transmission in progress and flushes the transmit buffer (JTX_BUF). To prevent another node from mistakenly assuming that the last byte was a CRC byte, two extra ‘1’s are appended. 0 = normal operation 1 = abort transmission in progress
3:0	MSG3:0	Message These four bits specify the number of bytes to be transmitted in the next message frame. This number includes the header, but not the CRC byte. In normal messaging, the maximum number of bytes you can transmit in a message frame is eleven. **MSG3:0** — **Operation** — **Purpose** FH — Termination byte — Terminate block transmission EH — Block transmission — Transmit unspecified number of bytes DH — Reserved — — CH — Reserved — — B:0H — Normal messaging — Transmit specified number of bytes

Figure 8-16. J1850 Command (J_CMD) Register

8.6.2 Programming the J1850 Configuration (J_CFG) Register

The J1850 configuration register (Figure 8-17) selects the proper oscillator prescaler, initiates a transmission break for debugging, invokes clock quadrupling operation, and selects the normalization bit format.

J_CFG

Address: 1F54H
Reset State: 00H

The J1850 configuration (J_CFG) register selects the proper oscilator prescaler, initiates transmission break for debug, invokes clock quadrupling operation, and selects the normalizartion bit format. This byte register can be directly addressed through *windowing*. All J1850 bus activity is ignored until you first write to this register.

7							**0**
NBF	IFR3	4XM	TXBRK	RXPOL	—	PRE1	PRE0

Bit Number	**Bit Mnemonic**	**Function**
7	NBF	Normalization Bit Format This bit specifies which normalization bit (NB) format is to be used. **IFR with CRC Byte** — **IFR without CRC Byte** 0 = active long NB — 0 = active short NB 1 = active short NB — 1 = active long NB
6	IFR3	Type 3 IFR Messaging This bit selects type 3 IFR messaging, which supports the in-frame transfer of an unspecified number of data bytes. 0 = normal operation 1 = type 3 IFR messaging
5	4XM	Oscillator Quadruple (4x) Mode This bit allows the J1850 peripheral to operate at four times the normal bit transfer rate (41.6 Kb/s versus 10.4 Kb/s). 0 = normal operation 1 = 4x mode operation
4	TXBRK	Transmission Break This bit will terminate any transmission in progress by writing a break (BRK) symbol to the bus. 0 = normal operation 1 = transmit BRK symbol onto bus
3	RXPOL	Receive Polarity This bit changes the polarity of the receive symbol. 0 = normal operation – Rx input inverted 1 = receive polarity enabled – Rx input non-inverted

Figure 8-17. J1850 Configuration (J_CFG) Register

J_CFG

Address: 1F54H
Reset State: 00H

The J1850 configuration (J_CFG) register selects the proper oscilator prescaler, initiates transmission break for debug, invokes clock quadrupling operation, and selects the normalizartion bit format. This byte register can be directly addressed through *windowing*. All J1850 bus activity is ignored until you first write to this register.

7							0
NBF	IFR3	4XM	TXBRK	RXPOL	—	PRE1	PRE0

Bit Number	Bit Mnemonic	Function
2	—	Reserved; for compatibility with future devices, write zero to this bit.
1:0	PRE1:0	J1850 Oscillator Prescaler These bits ensure proper operation of the J1850 peripheral at the supported input frequencies (F_{XTAL1}). **PRE1** **PRE0** **F_{XTAL1}** 0 0 8 MHz 0 1 12 MHz 1 0 16 MHz 1 1 20 MHz

Figure 8-17. J1850 Configuration (J_CFG) Register (Continued)

8.6.3 Programming the J1850 Delay Compensation (J_DLY) Register

The J1850 delay compensation register (Figure 8-18) allows you to program the necessary delay time through the external transceiver to compensate for the inherent propagation delays and to accurately resolve bus contention during arbitration.

J_DLY | Address: 1F58H
Reset State: 00H

The J1850 delay (J_DLY) register allows you compensate for the inherent propagation delays and to accurately resolve bus contention during arbitration. This byte register can be directly addressed through *windowing*.

7							0
—	—	—	DLY4	DLY3	DLY2	DLY1	DLY0

Bit Number	Bit Mnemonic	Function
7:5	—	Reserved; for compatibility with future devices, write zeros to these bits.
4:0	DLY4:0	Delay Time These five bits specify the desired propagation delay between the J1850 controller circuitry and the off-chip transceiver device, in units of microseconds (μs).

Figure 8-18. J1850 Delay (J_DLY) Register

8.6.4 Programming the J1850 Status (J_STAT) Register

The J1850 status register (Figure 8-19) provides the current status of the message and the four interrupt sources associated with the J1850 protocol.

J_STAT

Address: 1F53H
Reset State: 00H

The J1850 status (J_STAT) register provides the current status of the message transfer, the receive and transmit buffers, and the four interrupt sources associated with the J1850 protocol. This byte register can be directly addressed through *windowing*. You must write to this register before transmitting each message. Reading this register clears all bits except BUS_STAT.

7							**0**
IFR_RCV	BUS_CONT	BUS_STAT	BRK_RCV	OVR_UNDR	MSG_TX	MSG_RX	J1850BE

Bit Number	Bit Mnemonic	Function
7	IFR_RCV	In-frame Response Received This bit indicates whether the IFR byte has been received and is ready to be read from the J1850 receiver (J_RX) register. 0 = no action 1 = IFR byte received
6	BUS_CONT	J1850 Bus Contention This bit indicates whether bus contention has been detected and arbitration has been lost. 0 = no action 1 = bus contention
5	BUS_STAT	J1850 Bus Status This bit indicates whether a transmission or reception is in progress on the J1850 bus. 0 = J1850 bus idle 1 = J1850 bus busy
4	BRK_RCV	Break Received This bit indicates whether a BRK symbol has been detected on the J1850 bus. 0 = no action 1 = BRK symbol detected
3	OVR_UNDR	Receive Overrun/Transmit Underflow Interrupt This bit indicates whether a receive buffer overrun (OVR) or transmit buffer underflow (UNDR) has occurred. An overrun occurs when a symbol is received while both J_RX and JRX_BUF contain unread bytes. An underflow occurs when a transmission is attempted while both J_TX and JTX_BUF are empty. 0 = normal operation 1 = OVR or UNDR detected

Figure 8-19. J1850 Status (J_STAT) Register

J_STAT

Address: 1F53H
Reset State: 00H

The J1850 status (J_STAT) register provides the current status of the message transfer, the receive and transmit buffers, and the four interrupt sources associated with the J1850 protocol. This byte register can be directly addressed through *windowing*. You must write to this register before transmitting each message. Reading this register clears all bits except BUS_STAT.

7 **0**

IFR_RCV	BUS_CONT	BUS_STAT	BRK_RCV	OVR_UNDR	MSG_TX	MSG_RX	J1850BE

Bit Number	Bit Mnemonic	Function
2	MSG_TX	Message Transmit Interrupt This bit signals the successful transmission of a message upon detecting the EOD symbol. 0 = no action 1 = message transmitted
1	MSG_RX	Message Receive Interrupt This bit signals the successful reception of a message upon detecting the EOD symbol. 0 = no action 1 = message received
0	J1850BE	J1850 Bus Error Interrupt This bit is set if one or more of the following conditions occur: • the calculated CRC for a received message does not equal C4H • an incomplete byte is received on the bus • an invalid bus symbol is detected on the bus • a transmission occurs and the feedback through the receiver is not detected within 60 µs

Figure 8-19. J1850 Status (J_STAT) Register (Continued)

9

Minimum Hardware Considerations

CHAPTER 9
MINIMUM HARDWARE CONSIDERATIONS

This chapter discusses the major hardware consideration differences between the 8XC196L*x* and the 8XC196K*x*. The 8XC196L*x* has implemented a reset source SFR that reveals the source of the most recent reset request.

9.1 IDENTIFYING THE RESET SOURCE

The reset source (RSTSRC) register indicates the source of the last reset that the microcontroller encountered (Figure 9-1). If more than one reset occurs at the same time, all of the corresponding RSTSRC bits are set. Reading this SFR clears all the register bits.

RSTSRC

Address: 1F92H
Reset State: XXH[1]

The reset source (RSTSRC) register indicates the source(s) of the last reset that the microcontroller encountered.

7							0
—	—	—	—	CFDRST	WDTRST	SFWRST	EXTRST

Bit Number	Bit Mnemonic	Function
7:4	—	Reserved; for compatibility with future devices, write zeros to these bits.
3	CFDRST	Clock Failure Detection Reset When set, this bit indicates that a failed clock caused the last reset.
2	WDTRST	Watchdog Timer Reset When set, this bit indicates that the watchdog timer caused the last reset.
1	SFWRST	Software Reset When set, this bit indicates that either the RST instruction or the IDLPD instruction used with an illegal key caused the last reset.
0	EXTRST	External Reset When set, this bit indicates that the RESET# pin being asserted caused the last reset.

NOTE:
1. The State of the RSTSRC register is inderterminate on a V_{CC} power up condition. All other reset states will have the corresponding reset event bit set in the register.

Figure 9-1. Reset Source (RSTSRC) Register

9.2 DESIGN CONSIDERATIONS FOR 8XC196LA, LB, AND LD

With the exception of a few new multiplexed functions, the 8XC196L*x* microcontrollers are pin compatible with the 8XC196J*x* microcontrollers. The 8XC196J*x* microcontrollers are 52-lead versions of 8XC196K*x* microcontrollers.

Follow these recommendations to help maintain hardware and software compatibility between the 8XC196L*x*, 8XC196K*x*, and future microcontrollers.

- **Bus width.** Since the 8XC196L*x* has neither a WRH# nor a BUSWIDTH pin, the microcontroller cannot dynamically switch between 8- and 16-bit bus widths. Program the CCBs to select 8-bit bus mode.
- **Wait states.** Since the 8XC196L*x* has no READY pin, the microcontroller cannot rely on a READY signal to control wait states. Program the CCBs to limit the number of wait states (0, 1, 2, or 3).
- **EPA6–EPA7.** These functions exist in the 8XC196L*x*, but the associated pins are omitted. You can use these functions as software timers, to start A/D conversions (on 87C196LA and LB only), or to reset the timers.
- **Slave port.** Since the 8XC196L*x* has no P5.1/SLPCS and P5.4/SLPINT pins, you cannot use the slave port.
- **ONCE mode.** On the 8XC196L*x*, the ONCE mode entry function is multiplexed with P2.6 (and TXJ1850 on the 87C196LB) rather than with P5.4 as it is on the 8XC196K*x* (P5.4/SLPINT/ONCE).
- **NMI.** Since the 8XC196L*x* has no NMI pin, the nonmaskable interrupt is not supported. Initialize the NMI vector (at location 203EH) to point to a RET instruction. This method provides glitch protection only.
- **I/O ports.** The following port pins do not exist in the 8XC196L*x*: P0.0–P0.1, P1.4–P1.7, P2.3 and P2.5, P5.1 and P5.4–P5.7, P6.2 and P6.3. Software can still read and write the associated P*x*_REG, P*x*_MODE, and P*x*_DIR registers. Configure the registers for the omitted pins as follows:

  — Clear the corresponding P*x*_DIR bits. (Configures pins as complementary outputs.)

  — Clear the corresponding P*x*_MODE bits. (Selects I/O port function.)

  — Write either “0” or “1” to the corresponding P*x*_REG bits. (Effectively ties signals low or high.)

  Do not use the bits associated with the omitted port pins for conditional branch instructions. Treat these bits as reserved.
- **Auto programming.** During auto programming, the 8XC196L*x* supports only a 16-bit, zero-wait-state bus configuration.

10

Special Operating Modes

CHAPTER 10
SPECIAL OPERATING MODES

The 8XC196Lx's idle and powerdown modes are the same as those of the 8XC196Kx. However, the clock circuitry has changed, and the on-circuit emulation (ONCE) special-purpose mode operation has changed slightly because of the new reset state pin levels that have been implemented.

10.1 INTERNAL TIMING

The 87C196LA and LB clock circuitry (Figure 10-1) implements a phase-locked loop and clock multiplier circuitry, which can substantially increase the CPU clock rate while using a lower-frequency input clock.

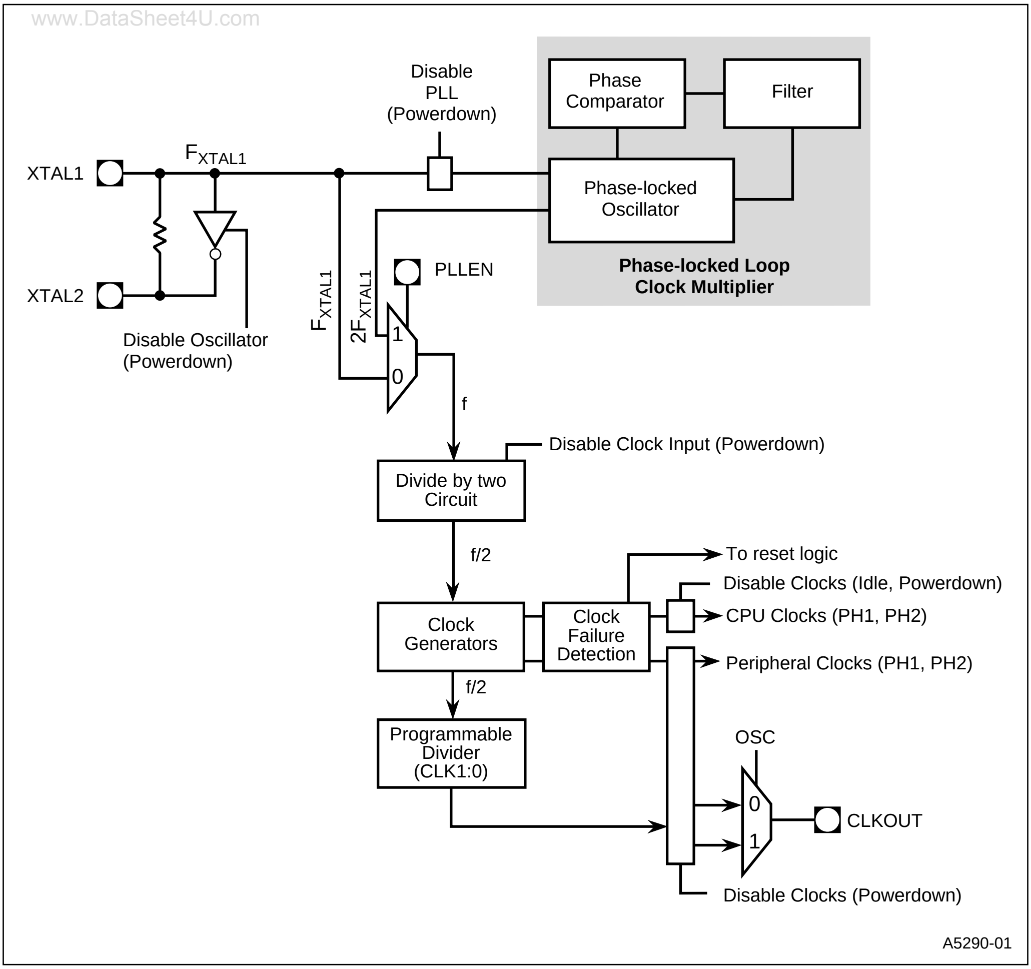


Figure 10-1. Clock Circuitry (87C196LA, LB Only)

10.2 ENTERING AND EXITING ONCE MODE

ONCE mode isolates the device from other components in the system to allow printed-circuit-board testing or debugging with a clip-on emulator. During ONCE mode, all pins except XTAL1, XTAL2, V_{SS}, and V_{CC} are weakly pulled either high or low. During ONCE mode, RESET# must be held high or the device will exit ONCE mode and enter the reset state.

On the 87C196LA and LB, the reset state level of all 41 general-purpose I/O pins has changed from a weak logic “1” (wk1) to a weak logic “0” (wk0). ONCE shares a package with port pin 2.6. Asserting and holding the ONCE signal high during the rising edge of RESET# causes the device to enter ONCE mode. To prevent accidental entry into ONCE mode, configure this pin as an output. If you choose to configure this pin as an input, always hold it low during reset and ensure that your system meets the V_{IH} specification to prevent inadvertent entry into ONCE mode.

11

Programming the Nonvolatile Memory

CHAPTER 11
PROGRAMMING THE NONVOLATILE MEMORY

The 87C196LA and LB microcontrollers contain 24 Kbytes (2000–7FFFH) of one-time-programmable read-only memory (OTPROM). OTPROM is similar to EPROM, but it comes in a windowless package and cannot be erased. You have the option of programming the OTPROM yourself or having the factory program it as a quick-turn ROM product (the latter option may not be available for all devices).

NOTE

> In this supplement, OTPROM refers to the device's internal read-only memory, whether it is EPROM or OTPROM, and EPROM refers specifically to EPROM devices.

The 87C196LA and LB programming signals, registers, and procedures are the same as those of the 87C196K*x*. This chapter describes the differences in memory mapping and programming circuits for the 87C196LA and LB.

11.1 SIGNATURE WORD AND PROGRAMMING VOLTAGE VALUES

The 8XC196L*x*'s programming voltage values are the same as those of the 8XC196K*x*; however, the signature word value differs. Table 11-1 lists the signature word and programming voltage values.

Table 11-1. Signature Word and Programming Voltage Values

Device	Signature Word		Programming V_{CC}		Programming V_{PP}	
	Location	Value	Location	Value	Location	Value
87C196LA	0070H	871BH	0072H	40H	0073H	0A0H
87C196LB	0070H	871BH	0072H	40H	0073H	0A0H

11.2 OTPROM ADDRESS MAP

The OTPROM contains customer-specified special-purpose and program memory (Table 11-2). The 128-byte special-purpose address partition is used for interrupt vectors, the chip configuration bytes (CCBs), and the security key. Several locations are reserved for testing or for use in future products. Write the value (20H or FFH) indicated in Table 11-2 to each reserved location. The remainder of the OTPROM is available for code storage.

Table 11-2. 87C196LA, LB OTPROM Address Map

Address Range (Hex)	Description
7FFF 2080	Program memory
207F 205E	Reserved (each location must contain FFH)
205D 2040	PTS vectors
203F 2030	Upper interrupt vectors
202F 2020	Security key
201F 201C	Reserved (each location must contain FFH)
201B	Reserved (must contain 20H)
201A	CCB1
2019	Reserved (must contain 20H)
2018	CCB0
2017 2016	OFD flag for QROM or MROM codes†
2015 2014	Reserved (each location must contain FFH)
2013 2000	Lower interrupt vectors

† Intel manufacturing uses this location to determine whether to program the OFD bit. Customers with quick-ROM (QROM) or masked-ROM (MROM) codes who desire oscillator failure detection should equate this location to the value 0CDEH.

11.3 SLAVE PROGRAMMING CIRCUIT AND ADDRESS MAP

Figure 11-1 shows the circuit diagram and Table 11-3 details the address map for slave programming of the 87C196LA and LB devices.

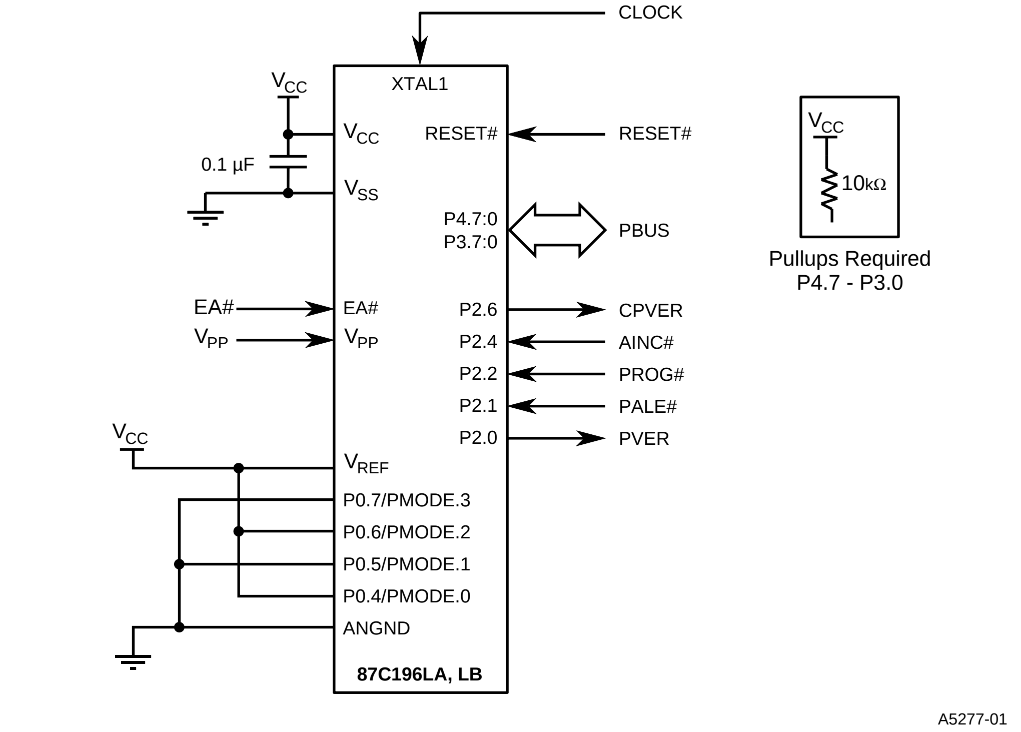


Figure 11-1. Slave Programming Circuit

Table 11-3. Slave Programming Mode Address Map

Description	Address	Comments
OTPROM	2000–7FFFH	OTPROM Cells
OFD	0778H	OTPROM Cell
DED[†]	0758H	UPROM Cell
DEI[†]	0718H	UPROM Cell
PCCB	0218H	Test EPROM
Programming V_{CC}	0072H	Read Only
Programming V_{PP}	0073H	Read Only
Signature word	0070H	Read Only

[†] These bits program the UPROM cells. Once these bits are programmed, they cannot be erased, and dynamic failure analysis of the device is impossible.

11.4 SERIAL PORT PROGRAMMING CIRCUIT AND ADDRESS MAP

Figure 11-2 shows the circuit and Table 11-4 details the address map for serial port programming.

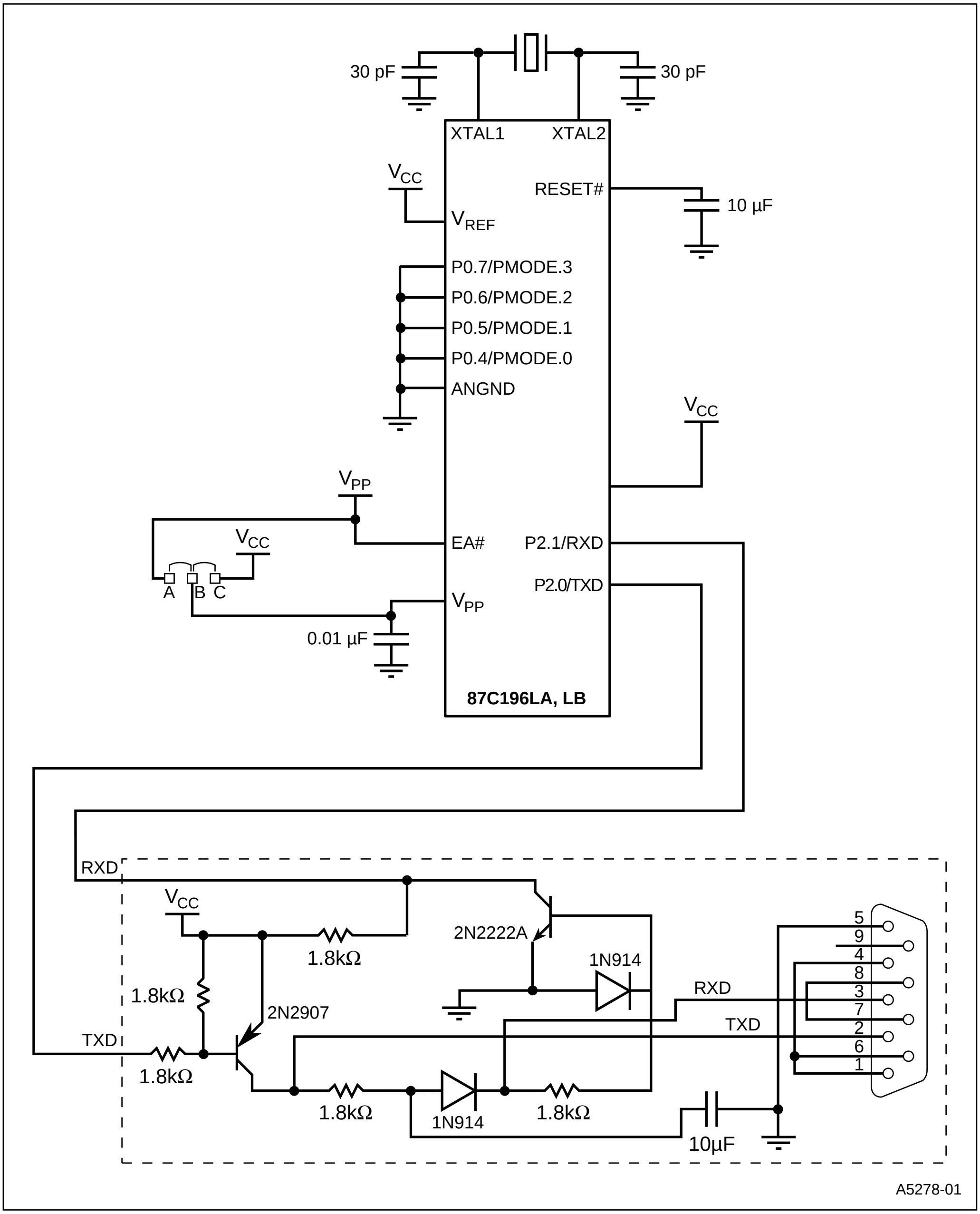


Figure 11-2. Serial Port Programming Circuit

Table 11-4. Serial Port Programming Mode Address Map

Description	Address Range	
	Normal Operation	Serial Port Programming Mode
Internal OTPROM	2000–7FFFH	A000–FFFFH
External memory	—	4000–9FFFH
Do not address	—	2400–3FFFH
Test ROM and RISM	—	2000–23FFH

A

Signal Descriptions

APPENDIX A
SIGNAL DESCRIPTIONS

This appendix provides reference information for the pin functions of the 8XC196L*x* microcontrollers.

A.1 FUNCTIONAL GROUPINGS OF SIGNALS

Tables A-1, A-2, and A-3 list the signal assignments for the 8XC196L*x* microcontrollers, grouped by function. A diagram of each microcontroller shows the pin location of each signal.

Table A-1. 87C196LA Signals Arranged by Functional Categories

Addr & Data

Name	Pin
AD0	22
AD1	21
AD2	20
AD3	19
AD4	18
AD5	17
AD6	16
AD7	15
AD8	14
AD9	13
AD10	12
AD11	11
AD12	10
AD13	9
AD14	8
AD15	7

Input/Output

Name	Pin
P0.2 / ACH2	33
P0.3 / ACH3	34
P0.4 / ACH4	35
P0.5 / ACH5	36
P0.6 / ACH6	37
P0.7 / ACH7	38
P1.0 / EPA0 / T2CLK	44
P1.1 / EPA1	43
P1.2 / EPA2 / T2DIR	42
P1.3 / EPA3	41
P2.0 / TXD	27

Input/Output (Cont'd)

Name	Pin
P2.1 / RXD	28
P2.2	29
P2.4	30
P2.6	31
P2.7	32
P3.0	22
P3.1	21
P3.2	20
P3.3	19
P3.4	18
P3.5	17
P3.6	16
P3.7	15
P4.0	14
P4.1	13
P4.2	12
P4.3	11
P4.4	10
P4.5	9
P4.6	8
P4.7	7
P5.0	2
P5.2	6
P5.3	5
P6.0 / EPA8 / COMP0	45
P6.1 / EPA9 / COMP1	46
P6.4 / SC0	47
P6.5 / SD0	48
P6.6 / SC1	49
P6.7 / SD1	50

Program Control

Name	Pin
AINC#	30
CPVER	31
PACT#	32
PALE#	28
PBUS.0	22
PBUS.1	21
PBUS.2	20
PBUS.3	19
PBUS.4	18
PBUS.5	17
PBUS.6	16
PBUS.7	15
PBUS.8	14
PBUS.9	13
PBUS.10	12
PBUS.11	11
PBUS.12	10
PBUS.13	9
PBUS.14	8
PBUS.15	7
PMODE.0	35
PMODE.1	36
PMODE.2	37
PMODE.3	38
PROG#	29
PVER	27

Processor Control

Name	Pin
EA#	24
EXTINT	29
PLLEN	6
RESET#	23
XTAL1	52
XTAL2	51

Bus Cont & Status

Name	Pin
ADV# / ALE	2
CLKOUT	32
RD#	5
WR# / WRL#	6

Power & Ground

Name	Pin
ANGND	39
V_{CC}	26
V_{PP}	4
V_{REF}	40
V_{SS}	3
V_{SS1}	1
V_{SS1}	25

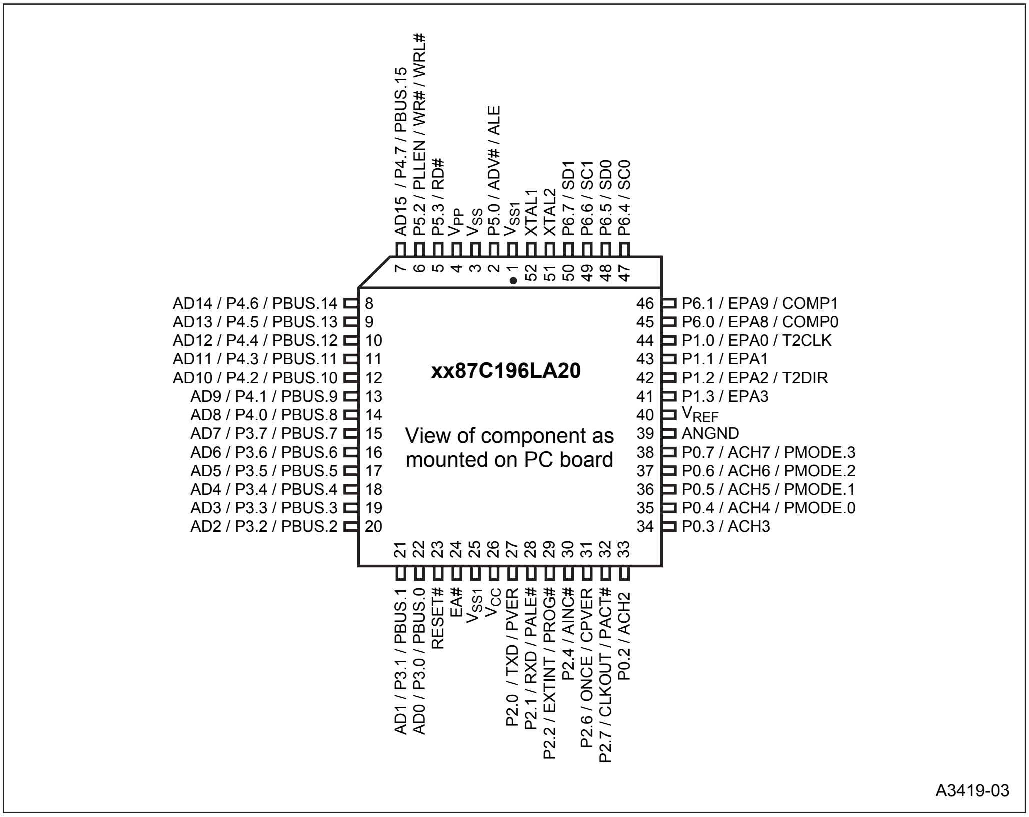


Figure A-1. 87C196LA 52-pin PLCC Package

Table A-2. 87C196LB Signals Arranged by Functional Categories

Addr & Data	
Name	**Pin**
AD0	22
AD1	21
AD2	20
AD3	19
AD4	18
AD5	17
AD6	16
AD7	15
AD8	14
AD9	13
AD10	12
AD11	11
AD12	10
AD13	9
AD14	8
AD15	7

Input/Output	
Name	**Pin**
P0.2 / ACH2	33
P0.3 / ACH3	34
P0.4 / ACH4	35
P0.5 / ACH5	36
P0.6 / ACH6	37
P0.7 / ACH7	38
P1.0 / EPA0 / T2CLK	44
P1.1 / EPA1	43
P1.2 / EPA2 / T2DIR	42
P1.3 / EPA3	41
P2.0 / TXD	27

Input/Output (Cont'd)	
Name	**Pin**
P2.1 / RXD	28
P2.2	29
P2.4 / RXJ1850	30
P2.6 / TXJ1850	31
P2.7	32
P3.0	22
P3.1	21
P3.2	20
P3.3	19
P3.4	18
P3.5	17
P3.6	16
P3.7	15
P4.0	14
P4.1	13
P4.2	12
P4.3	11
P4.4	10
P4.5	9
P4.6	8
P4.7	7
P5.0	2
P5.2	6
P5.3	5
P6.0 / EPA8 / COMP0	45
P6.1 / EPA9 / COMP1	46
P6.4 / SC0	47
P6.5 / SD0	48
P6.6 / SC1	49
P6.7 / SD1	50

Program Control	
Name	**Pin**
AINC#	30
CPVER	31
PACT#	32
PALE#	28
PBUS.0	22
PBUS.1	21
PBUS.2	20
PBUS.3	19
PBUS.4	18
PBUS.5	17
PBUS.6	16
PBUS.7	15
PBUS.8	14
PBUS.9	13
PBUS.10	12
PBUS.11	11
PBUS.12	10
PBUS.13	9
PBUS.14	8
PBUS.15	7
PMODE.0	35
PMODE.1	36
PMODE.2	37
PMODE.3	38
PROG#	29
PVER	27

Processor Control	
Name	**Pin**
EA#	24
EXTINT	29
PLLEN	6
RESET#	23
XTAL1	52
XTAL2	51

Bus Cont & Status	
Name	**Pin**
ADV# / ALE	2
CLKOUT	32
RD#	5
WR# / WRL#	6

Power & Ground	
Name	**Pin**
ANGND	39
V_{CC}	26
V_{PP}	4
V_{REF}	40
V_{SS}	3
V_{SS1}	1
V_{SS1}	25

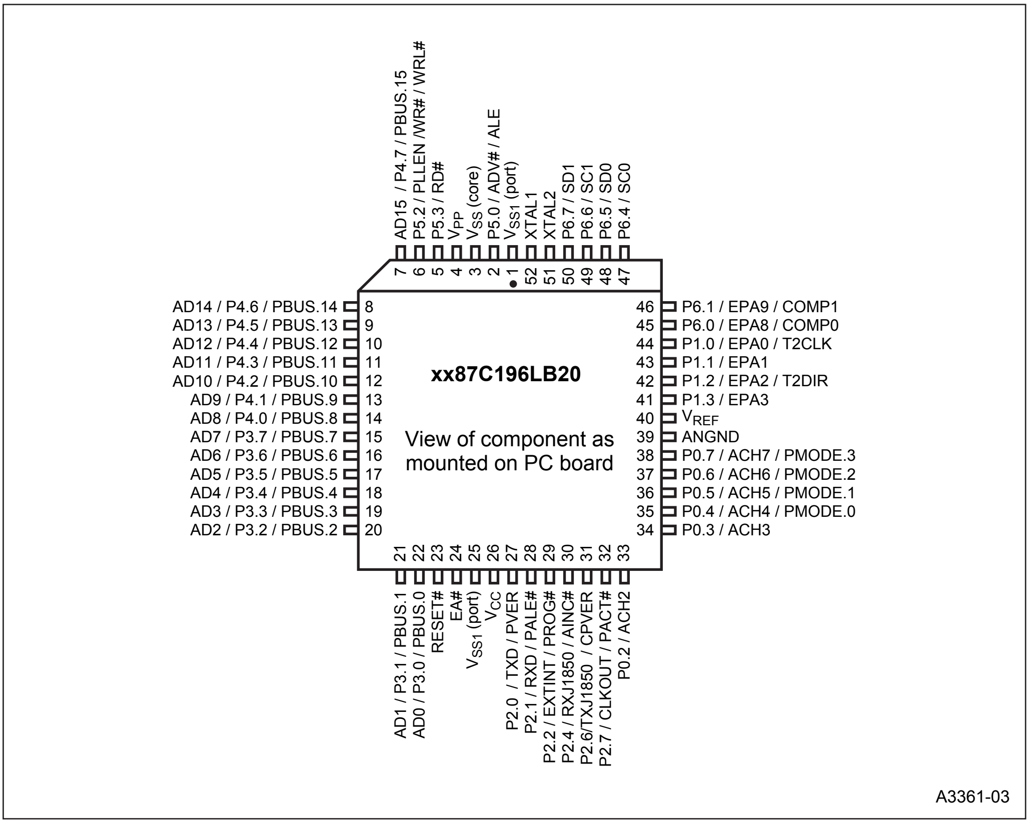


Figure A-2. 87C196LB 52-pin PLCC Package

Table A-3. 83C196LD Signals Arranged by Functional Categories

Addr & Data	
Name	**Pin**
AD0	22
AD1	21
AD2	20
AD3	19
AD4	18
AD5	17
AD6	16
AD7	15
AD8	14
AD9	13
AD10	12
AD11	11
AD12	10
AD13	9
AD14	8
AD15	7

Input	
Name	**Pin**
P0.2	33
P0.3	34
P0.4	35
P0.5	36
P0.6	37
P0.7	38

Input/Output	
Name	**Pin**
P1.0/EPA0/T2CLK	44
P1.1/EPA1	43
P1.2/EPA2/T2DIR	42
P1.3/EPA3	41
P2.0/TXD	27
P2.1/RXD	28
P2.2	29
P2.4	30
P2.6	31
P2.7	32
P3.0	22
P3.1	21
P3.2	20
P3.3	19
P3.4	18
P3.5	17
P3.6	16
P3.7	15
P4.0	14
P4.1	13
P4.2	12
P4.3	11
P4.4	10
P4.5	9
P4.6	8
Input/Output (Cont'd)	
Name	**Pin**
P4.7	7
P5.0	2
P5.2	6
P5.3	5
P6.0/EPA8	45
P6.1/EPA9	46
P6.4/SC0	47
P6.5/SD0	48
P6.6/SC1	49
P6.7/SD1	50

Power & Ground	
Name	**Pin**
V_{CC}	26
V_{CC}	40
V_{PP}	4
V_{SS}	1
V_{SS}	3
V_{SS}	25
V_{SS}	39

Processor Control	
Name	**Pin**
CLKOUT	32
EA#	24
EXTINT	29
ONCE#	31
RESET#	23
XTAL1	52
XTAL2	51

Bus Control & Status	
Name	**Pin**
ADV#/ALE	2
RD#	5
WR#/WRL#	6

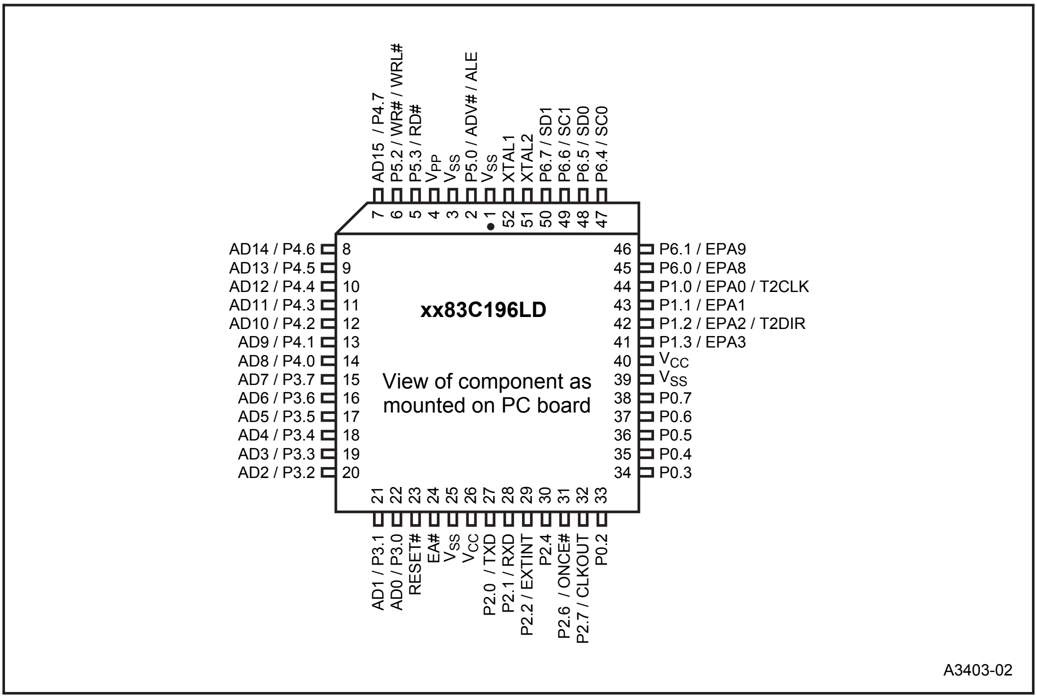


Figure A-3. 83C196LD 52-pin PLCC Package

A.2 DEFAULT CONDITIONS

Table A-5 lists the values of the signals for the 87C196LA and 87C196LB during various operating conditions. Table A-6 lists the same information for the 83C196LD. Table A-4 defines the symbols used to represent the pin status. Refer to the DC characteristics table in the datasheet for actual specifications for V_{OL}, V_{IL}, V_{OH}, and V_{IH}.

Table A-4. Definition of Status Symbols

Symbol	Definition	Symbol	Definition
0	Voltage less than or equal to V_{OL}, V_{IL}	MD0	Medium pull-down
1	Voltage greater than or equal to V_{OH}, V_{IH}	MD1	Medium pull-up
HiZ	High impedance	WK0	Weak pull-down
LoZ0	Low impedance; strongly driven low	WK1	Weak pull-up
LoZ1	Low impedance; strongly driven high	ODIO	Open-drain I/O

Table A-5. 87C196LA, LB Default Signal Conditions

Port Signals	Alternate Functions	During RESET# Active	Upon RESET# Inactive (Note 6)	Idle	Power-down
P0.7:2	ACH7:2	HiZ	HiZ	HiZ	HiZ
P1.0	EPA0/T2CLK	WK0	WK0	(Note 1)	(Note 1)
P1.1	EPA1	WK0	WK0	(Note 1)	(Note 1)
P1.2	EPA2/T2DIR	WK0	WK0	(Note 1)	(Note 1)
P1.3	EPA3	WK0	WK0	(Note 1)	(Note 1)
P2.0	TXD	WK0	WK0	(Note 1)	(Note 1)
P2.1	RXD	WK0	WK0	(Note 1)	(Note 1)
P2.2	EXTINT	WK0	WK0	(Note 1)	(Note 1)
P2.4	RXJ1850 (LB only)	WK0	WK0	(Note 1)	(Note 1)
P2.6	ONCE/TXJ1850 (LB only)	MD0	MD0	(Note 1)	(Note 1)
P2.7	CLKOUT	CLKOUT active, LoZ0/1	CLKOUT active, LoZ0/1	(Note 1)	(Note 2)
P3.7:0	AD7:0	WK0	HiZ	(Note 4)	(Note 4)
P4.7:0	AD15:8	WK0	HiZ	(Note 4)	(Note 4)
P5.0	ALE/ADV#	WK0	WK0	(Note 1)	(Note 1)
P5.2	WR#/WRL#	WK0	WK0	(Note 1)	(Note 1)
P5.3	RD#	WK0	WK0	(Note 1)	(Note 1)
P6.0	EPA8/COMP0	WK0	WK0	(Note 1)	(Note 1)
P6.1	EPA9/COMP1	WK0	WK0	(Note 1)	(Note 1)
P6.4	SC0	WK0	WK0	(Note 1)	(Note 1)
P6.5	SD0	WK0	WK0	(Note 1)	(Note 1)
P6.6	SC1	WK0	WK0	(Note 1)	(Note 1)
P6.7	SD1	WK0	WK0	(Note 1)	(Note 1)
—	EA#	WK1 (Note 5)	WK1	WK1	WK1
—	RESET#	LoZ0	MD1	MD1	MD1
—	V_{PP}	HiZ	HiZ	LoZ1	LoZ1
—	XTAL1	Osc input, HiZ	Osc input, HiZ	Osc input, HiZ	Osc input, HiZ
—	XTAL2	Osc output, LoZ0/1	Osc output, LoZ0/1	Osc output, LoZ0/1	(Note 3)

NOTES:

1. If P*x*_MODE.*y* = 0, port is as programmed.
   If P*x*_MODE.*y* = 1, pin is as specified by P*x*_DIR and the associated peripheral.
2. If P2_MODE.7 = 0, pin is as programmed. If P2_MODE.7 = 1, pin is LoZ0.
3. If XTAL1 = 0, pin is LoZ1. If XTAL1 = 1, pin is LoZ0.
4. If EA# = 0, port is HiZ. If EA# = 1, port is open-drain I/O.
5. Although EA# is weakly pulled high, do not allow it to float. Always tie EA# to V_{CC} if it is not connected to an external device.
6. The values in this column are valid until your software writes to P*x*_MODE.

Table A-6. 83C196LD Default Signal Conditions

Port Signals	Alternate Functions	During RESET# Active	Upon RESET# Inactive (Note 6)	Idle	Power-down
P0.7:2	—	HiZ	HiZ	HiZ	HiZ
P1.0	EPA0/T2CLK	WK1	WK1	(Note 1)	(Note 1)
P1.1	EPA1	WK1	WK1	(Note 1)	(Note 1)
P1.2	EPA2/T2DIR	WK1	WK1	(Note 1)	(Note 1)
P1.3	EPA3	WK1	WK1	(Note 1)	(Note 1)
P2.0	TXD	WK1	WK1	(Note 1)	(Note 1)
P2.1	RXD	WK1	WK1	(Note 1)	(Note 1)
P2.2	EXTINT	WK1	WK1	(Note 1)	(Note 1)
P2.4	—	WK1	WK1	(Note 1)	(Note 1)
P2.6	ONCE	MD1	MD1	(Note 1)	(Note 1)
P2.7	CLKOUT	CLKOUT active, LoZ0/1	CLKOUT active, LoZ0/1	(Note 1)	(Note 2)
P3.7:0	AD7:0	WK1	HiZ	(Note 4)	(Note 4)
P4.7:0	AD15:8	WK1	HiZ	(Note 4)	(Note 4)
P5.0	ALE/ADV#	WK1	WK1	(Note 1)	(Note 1)
P5.2	WR#/WRL#	WK1	WK1	(Note 1)	(Note 1)
P5.3	RD#	WK1	WK1	(Note 1)	(Note 1)
P6.0	EPA8	WK1	WK1	(Note 1)	(Note 1)
P6.1	EPA9	WK1	WK1	(Note 1)	(Note 1)
P6.4	SC0	WK1	WK1	(Note 1)	(Note 1)
P6.5	SD0	WK1	WK1	(Note 1)	(Note 1)
P6.6	SC1	WK1	WK1	(Note 1)	(Note 1)
P6.7	SD1	WK1	WK1	(Note 1)	(Note 1)
—	EA#	WK1 (Note 5)	WK1	WK1	WK1
—	RESET#	LoZ0	MD1	MD1	MD1
—	V_{PP}	HiZ	HiZ	LoZ1	LoZ1
—	XTAL1	Osc input, HiZ	Osc input, HiZ	Osc input, HiZ	Osc input, HiZ
—	XTAL2	Osc output, LoZ0/1	Osc output, LoZ0/1	Osc output, LoZ0/1	(Note 3)

NOTES:

1. If P*x*_MODE.*y* = 0, port is as programmed.
   If P*x*_MODE.*y* = 1, pin is as specified by P*x*_DIR and the associated peripheral.
2. If P2_MODE.7 = 0, pin is as programmed. If P2_MODE.7 = 1, pin is LoZ0.
3. If XTAL1 = 0, pin is LoZ1. If XTAL1 = 1, pin is LoZ0.
4. If EA# = 0, port is HiZ. If EA# = 1, port is open-drain I/O.
5. Although EA# is weakly pulled high, do not allow it to float. Always tie EA# to V_{CC} if it is not connected to an external device.
6. The values in this column are valid until your software writes to P*x*_MODE.

Glossary

GLOSSARY

This glossary defines acronyms, abbreviations, and terms that have special meaning in this manual. (Chapter 1 discusses notational conventions and general terminology.)

absolute error The maximum difference between corresponding actual and ideal *code transitions*. Absolute error accounts for all deviations of an actual A/D converter from an ideal converter.

accumulator A register or storage location that forms the result of an arithmetic or logical operation.

actual characteristic A graph of output code versus input voltage of an actual *A/D converter.* An actual characteristic may vary with temperature, supply voltage, and frequency conditions.

A/D converter Analog-to-digital converter. An internal peripheral that converts an analog input to a digital value.

ALU Arithmetic-logic unit. The part of the *RALU* that processes arithmetic and logical operations.

assert The act of making a signal active (enabled). The polarity (high or low) is defined by the signal name. Active-low signals are designated by a pound symbol (#) suffix; active-high signals have no suffix. To assert RD# is to drive it low; to assert ALE is to drive it high.

attenuation A decrease in amplitude; voltage decay.

bit A binary digit.

BIT A single-bit operand that can take on the Boolean values, “true” and “false.”

bit arbitration The process of settling conflicts that occur when multiple nodes attempt to transmit a bit or symbol across a single bus at the same time.

break-before-make The property of a multiplexer which guarantees that a previously selected channel is deselected before a new channel is selected. (That is, break-before-make ensures that the *A/D converter* will not short inputs together.)

byte Any 8-bit unit of data.

BYTE An unsigned, 8-bit variable with values from 0 through 2^8–1.

CCBs Chip configuration bytes. The chip configuration registers (*CCRs*) are loaded with the contents of the CCBs after a reset.

CCRs Chip configuration registers. Registers that define the environment in which the microcontroller will be operating. The chip configuration registers are loaded with the contents of the *CCBs* after a reset.

channel-to-channel matching error The difference between corresponding *code transitions* of actual characteristics taken from different *A/D converter* channels under the same temperature, voltage, and frequency conditions. This error is caused by differences in *DC input leakage* and on-channel resistance from one multiplexer channel to another.

characteristic A graph of output code versus input voltage; the *transfer function* of an *A/D converter.*

chip-select unit The integrated module that selects an external memory device during an external bus cycle.

clear The “0” value of a bit or the act of giving it a “0” value. See also *set*.

code 1) A set of instructions that perform a specific function; a program.
2) The digital value output by the *A/D converter.*

code center The voltage corresponding to the midpoint between two adjacent *code transitions* on the *A/D converter.*

code transition The point at which the *A/D converter*’s output code changes from “Q” to “Q+1.” The input voltage corresponding to a code transition is defined as the voltage that is equally likely to produce either of two adjacent codes.

code width The voltage change corresponding to the difference between two adjacent *code transitions*. Code width deviations cause *differential nonlinearity* and *nonlinearity* errors.

contention The detection of conflicting symbols or bits on the bus.

crosstalk See *off-isolation.*

DC input leakage Leakage current from an analog input pin to ground or to the reference voltage (V_{REF}).

deassert The act of making a signal inactive (disabled). The polarity (high or low) is defined by the signal name. Active-low signals are designated by a pound symbol (#) suffix; active-high signals have no suffix. To deassert RD# is to drive it high; to deassert ALE is to drive it low.

demultiplexed bus The configuration in which the microcontroller uses separate lines for address and data (address on A20:0; data on AD15:0 for a 16-bit bus or AD7:0 for an 8-bit bus). See also *multiplexed bus*.

differential nonlinearity The difference between the actual *code width* and the ideal one-LSB code width of the *terminal-based characteristic* of an A/D converter. It provides a measure of how much the input voltage may have changed in order to produce a one-count change in the conversion result. *Differential nonlinearity* is a measure of local code-width error; *nonlinearity* is a measure of overall code-transition error.

doping The process of introducing a periodic table Group III or Group V element into a Group IV element (e.g., silicon). A Group III impurity (e.g., indium or gallium) results in a *p-type material*. A Group V impurity (e.g., arsenic or antimony) results in an *n-type material*.

double-word Any 32-bit unit of data.

DOUBLE-WORD An unsigned, 32-bit variable with values from 0 through 2^{32}–1.

EPA Event processor array. An integrated peripheral that provides high-speed input/output capability.

ESD Electrostatic discharge.

external address A 21-bit address is presented on the microcontroller's pins. The address decoded by an external device depends on how many of these address pins the external system uses. See also *internal address*.

f Lowercase "f" represents the frequency of the internal clock.

far constants Constants that can be accessed only with extended instructions. See also *near constants*.

far data Data that can be accessed only with extended instructions. See also *near data*.

feedthrough The *attenuation* from an input voltage on the selected channel to the A/D output after the *sample window* closes. The ability of the *A/D converter* to reject an input on its selected channel after the sample window closes.

FET Field-effect transistor.

full-scale error The difference between the ideal and actual input voltage corresponding to the final (full-scale) *code transition* of an *A/D converter.*

hold latency The time it takes the microcontroller to assert HLDA# after an external device asserts HOLD#.

ideal characteristic The *characteristic* of an ideal *A/D converter.* An ideal characteristic is unique: its first *code transition* occurs when the input voltage is 0.5 LSB, its full-scale (final) code transition occurs when the input voltage is 1.5 LSB less than the full-scale reference, and its code widths are all exactly 1.0 LSB. These properties result in a conversion without *zero-offset, full-scale*, or *linearity* errors. *Quantizing error* is the only error seen in an ideal A/D converter.

input leakage Current leakage from an input pin to power or ground.

input series resistance The effective series resistance from an analog input pin to the *sample capacitor* of an *A/D converter.*

integer Any member of the set consisting of the positive and negative whole numbers and zero.

INTEGER A 16-bit, signed variable with values from -2^{15} through $+2^{15}-1$.

internal address

The 24-bit address that the microcontroller generates. See also *external address*.

interrupt controller

The module responsible for handling interrupts that are to be serviced by *interrupt service routines* that you provide. Also called the *programmable interrupt controller (PIC)*.

interrupt latency

The total delay between the time that an interrupt is generated (not acknowledged) and the time that the microcontroller begins executing the *interrupt service routine* or *PTS routine*. Determine the instruction in your code that has the longest execution time and use that execution time in calculating interrupt latency.

interrupt service routine

A software routine that you provide to service a standard interrupt request.

interrupt vector

A location in *special-purpose memory* that holds the starting address of an *interrupt service routine*.

J1850

An integrated communications controller peripheral that supports the 10.4 Kb/s *variable pulse-width (VPW)* medium-speed, class B, in-vehicle network protocol.

ISR

See *interrupt service routine.*

linearity errors

See *differential nonlinearity* and *nonlinearity.*

LONG-INTEGER

A 32-bit, signed variable with values from -2^{31} through $+2^{31}-1$.

LSB

1) Least-significant bit of a byte or least-significant byte of a word.

2) In an A/D converter, the reference voltage divided by 2^n, where *n* is the number of bits to be converted. For a 10-bit converter with a reference voltage of 5.12 volts, one LSB is equal to 5.0 millivolts ($5.12 \div 2^{10}$).

LSW

Least-significant word of a double-word or quad-word.

maskable interrupts

All interrupts except stack overflow, unimplemented opcode, and software trap. Maskable interrupts can be disabled (masked) by the individual mask bits in the interrupt mask registers, and their servicing can be disabled by the DI (disable interrupt service) instruction. Each *maskable interrupt* can be assigned to the *PTS* for processing.

monotonic

The property of *successive approximation* converters which guarantees that increasing input voltages produce adjacent *codes* of increasing value, and that decreasing input voltages produce adjacent codes of decreasing value. (In other words, a converter is monotonic if every code change represents an input voltage change in the same direction.) Large *differential nonlinearity* errors can cause the converter to exhibit nonmonotonic behavior.

MSB

Most-significant bit of a *byte* or most-significant byte of a *word*.

MSW

Most-significant word of a double-word or quad-word.

multiplexed bus

The configuration in which the microcontroller uses both A20:0 and AD15:0 for address and also uses AD15:0 for data. See also *demultiplexed bus*.

***n*-channel FET**

A field-effect transistor with an *n*-type conducting path (channel).

***n*-type material**

Semiconductor material with introduced impurities (*doping*) causing it to have an excess of negatively charged carriers.

near constants

Constants that can be accessed with nonextended instructions. Constants in page 00H are near constants. See also *far constants*.

near data

Data that can be accessed with nonextended instructions. Data in page 00H is near data. See also *far data*.

no missing codes

An A/D converter has *no missing codes* if, for every output code, there is a unique input voltage range which produces that code only. Large *differential nonlinearity* errors can cause the converter to miss codes.

nonlinearity

The maximum deviation of *code transitions* of the *terminal-based characteristic* from the corresponding code transitions of the *ideal characteristic*.

nonmaskable interrupts

Interrupts that cannot be masked (disabled) and cannot be assigned to the PTS for processing. The nonmaskable interrupts are stack overflow, unimplemented opcode, software trap, and NMI. The DI (disable interrupt service) and EI (enable interrupt service) instructions have no effect on nonmaskable interrupts.

***npn* transistor**

A transistor consisting of one part *p-type material* and two parts *n-type material*.

off-isolation

The ability of an *A/D converter* to reject (isolate) the signal on a deselected (off) output.

***p*-channel FET**

A field-effect transistor with a *p*-type conducting path.

***p*-type material**

Semiconductor material with introduced impurities (*doping*) causing it to have an excess of positively charged carriers.

PC

Program counter.

phase-locked loop

A component of the clock generation circuitry. The phase-locked loop (PLL) and the input pin (PLLEN) combine to enable the microcontroller to attain its maximum operating frequency with an external clock whose frequency is either equal to or one-half that maximum frequency or with an external oscillator whose frequency is one-half that maximum frequency.

PIC

Programmable interrupt controller. The module responsible for handling interrupts that are to be serviced by *interrupt service routines* that you provide. Also called simply the *interrupt controller.*

PIH

Peripheral interrupt handler. An integrated module that provides interrupt vectors for specific *EPA* interrupt requests to the *interrupt controller* or *PTS.*

PLL

See *phase-locked loop.*

prioritized interrupt

NMI, stack overflow, or any *maskable interrupt*. Two of the *nonmaskable interrupts* (unimplemented opcode and software trap) are not prioritized; they vector directly to the *interrupt service routine* when executed.

program memory

A partition of memory where instructions can be stored for fetching and execution.

protected instruction

An instruction that prevents an interrupt from being acknowledged until after the next instruction executes. The protected instructions are DI, EI, DPTS, EPTS, POPA, POPF, PUSHA, and PUSHF.

PSW

Processor status word. The high byte of the PSW is the status byte, which contains one bit that globally enables or disables servicing of all maskable interrupts, one bit that enables or disables the *PTS*, and six Boolean flags that reflect the state of the current program. The low byte of the PSW is the INT_MASK register. A PUSHA or POPA instruction saves or restores both bytes (PSW + INT_MASK); a PUSHF or POPF saves or restores only the PSW.

PTS

Peripheral transaction server. The microcoded hardware interrupt processor.

PTSCB

See *PTS control block*.

PTS control block

A block of data required for each *PTS interrupt*. The microcode executes the proper *PTS routine* based on the contents of the PTS control block.

PTS cycle

The microcoded response to a **single** PTS interrupt request.

PTS interrupt

Any *maskable interrupt* that is assigned to the *PTS* for interrupt processing.

PTS mode

A microcoded response that enables the *PTS* to complete a specific task quickly.

PTS routine

The entire microcoded response to multiple PTS interrupt requests. The PTS routine is controlled by the contents of the *PTS control block*.

PTS transfer

The movement of a single byte or word from the source memory location to the destination memory location.

PTS vector

A location in *special-purpose memory* that holds the starting address of a *PTS control block*.

QUAD-WORD

An unsigned, 64-bit variable with values from 0 through 2^{64}–1. The QUAD-WORD variable is supported only as the operand for the EBMOVI instruction.

quantizing error

An unavoidable A/D conversion error that results simply from the conversion of a continuous voltage to its integer digital representation. Quantizing error is always ± 0.5 LSB and is the only error present in an ideal *A/D converter.*

RALU

Register arithmetic-logic unit. A part of the CPU that consists of the *ALU*, the *PSW*, the master *PC*, the microcode engine, a loop counter, and six registers.

repeatability error

The variation in *code transitions* when comparing a number of *actual characteristics* from the same converter on the same channel with the same temperature, voltage, and frequency conditions. The amount of repeatability error depends on the comparator's ability to resolve very similar voltages and the extent to which random noise contributes to the error.

reserved memory

A memory location that is reserved for factory use or for future expansion. Do not use a reserved memory location except to initialize it.

resolution

The number of input voltage levels that an *A/D converter* can unambiguously distinguish between. The number of useful bits of information that the converter can return.

sample capacitor

A small (2–3 pF) capacitor used in the *A/D converter* circuitry to store the input voltage on the selected input channel.

sample delay

The time period between the time that *A/D converter* receives the "start conversion" signal and the time that the *sample capacitor* is connected to the selected channel.

sample delay uncertainty

The variation in the *sample delay.*

sample time

The period of time that the *sample window* is open. (That is, the length of time that the input channel is actually connected to the *sample capacitor.*)

sample time uncertainty
The variation in the *sample time*.

sample window
The period of time that begins when the *sample capacitor* is attached to a selected channel of an *A/D converter* and ends when the sample capacitor is disconnected from the selected channel.

sampled inputs
All input pins, with the exception of RESET#, are sampled inputs. The input pin is sampled one state time before the read buffer is enabled. Sampling occurs during PH1 (while CLKOUT is low) and resolves the value (high or low) of the pin before it is presented to the internal bus. If the pin value changes during the sample time, the new value may or may not be recorded during the read.

RESET# is a level-sensitive input. EXTINT is normally a sampled input; however, the powerdown circuitry uses EXTINT as a level-sensitive input during powerdown mode.

SAR
Successive approximation register. A component of the *A/D converter.*

set
The “1” value of a bit or the act of giving it a “1” value. See also *clear.*

SFR
Special-function register.

SHORT-INTEGER
An 8-bit, signed variable with values from $–2^7$ through $+2^7–1$.

sign extension
A method for converting data to a larger format by filling the upper bit positions with the value of the sign. This conversion preserves the positive or negative value of signed integers.

sink current
Current flowing **into** a device to ground. Always a positive value.

source current
Current flowing **out of** a device from V_{CC}. Always a negative value.

SP
Stack pointer.

special interrupt
Any of the three *nonmaskable interrupts* (unimplemented opcode, software trap, or NMI).

special-purpose memory

A partition of memory used for storing the *interrupt vectors*, *PTS vectors*, chip configuration bytes, and several reserved locations.

standard interrupt

Any *maskable interrupt* that is assigned to the *interrupt controller* for processing by an *interrupt service routine*.

state time (or state)

The basic time unit of the microcontroller; the combined period of the two internal timing signals, PH1 and PH2. Because the microcontroller can operate at many frequencies, this manual defines time requirements in terms of *state times* rather than in specific units of time.

successive approximation

An A/D conversion method that uses a binary search to arrive at the best digital representation of an analog input.

t

Lowercase “t” represents the period of the internal clock.

temperature coefficient

Change in the stated variable for each degree Centigrade of temperature change.

temperature drift

The change in a specification due to a change in temperature. Temperature drift can be calculated by using the *temperature coefficient* for the specification.

terminal-based characteristic

An *actual characteristic* that has been translated and scaled to remove *zero-offset error* and *full-scale error.* A terminal-based characteristic resembles an *actual characteristic* with zero-offset error and full-scale error removed.

transfer function

A graph of output *code* versus input voltage; the *characteristic* of the *A/D converter.*

transfer function errors

Errors inherent in an analog-to-digital conversion process: *quantizing error*, *zero-offset error*, *full-scale error*, *differential nonlinearity*, and *nonlinearity.* Errors that are hardware-dependent, rather than being inherent in the process itself, include *feedthrough*, *repeatability*, *channel-to-channel matching*, *off-isolation*, and V_{CC} *rejection* errors.

UART

Universal asynchronous receiver and transmitter. A part of the serial I/O port.

V_{CC} rejection

The property of an A/D converter that causes it to ignore (reject) changes in V_{CC} so that the *actual characteristic* is unaffected by those changes. The effectiveness of *V_{CC} rejection* is measured by the ratio of the change in V_{CC} to the change in the *actual characteristic*.

VPW

Variable pulse-width. A forced high/low symbol transition formatting scheme that tracks the duration between two consecutive transitions and the level of the bus, active or passive.

wait state

Time spent waiting for an operation to take place. Wait states are added to external bus cycles to allow a slow memory device to respond to a request from the microcontroller.

watchdog timer

An internal timer that resets the microcontroller if software fails to respond before the timer overflows.

WDT

Watchdog timer. An internal timer that resets the microcontroller if software fails to respond before the timer overflows.

word

Any 16-bit unit of data.

WORD

An unsigned, 16-bit variable with values from 0 through 2^{16}–1.

zero extension

A method for converting data to a larger format by filling the upper bit positions with zeros.

zero-offset error

An ideal *A/D converter*'s first *code transition* occurs when the input voltage is 0.5 LSB. Zero-offset error is the difference between 0.5 LSB and the actual input voltage that triggers an A/D converter's first code transition.

Index

INDEX