DALLAS SEMICONDUCTOR

DS80C390
High-Speed Microcontroller
User's Guide Supplement

www.dalsemi.com

ADDENDUM TO SECTION 1: INTRODUCTION

This document is provided as a supplement to the High-Speed Microcontroller User's Guide, covering new or modified features specific to the DS80C390. ***This document must be used in conjunction with the High-Speed Microcontroller User's Guide, available from Dallas Semiconductor.*** Addenda are arranged by section number, which correspond to sections in the High-Speed Microcontroller User's Guide.

The following additions and changes, with respect to the High-Speed Microcontroller User's Guide, are contained in this document. This document is a work in progress, and updates/additions will be added when available.

Section 1: Introduction
No Changes.

Section 2: Ordering Information
Information on new members of the High-Speed Microcontroller family has been added.

Section 3: Architecture
No Changes. Information containing new architectural features is contained in the DS80C390 Data Sheet.

Section 4: Programming Model
Descriptions of the DS80C390 memory map are included, as well as new/modified Special Function Registers.

Section 5: CPU Timing
Descriptions of the clock multiply modes have been added.

Section 6: Memory Access
Descriptions of the 22-bit expanded address capability have been added. A discussion of how to use the internal 4KB SRAM as a bootloader is also presented.

Section 7: Power Management
Clarified function of ring oscillator and removal of PMM1.

Section 8: Reset Conditions
Complete description of the reset sources, reset output ($\overline{\text{RSTOL}}$) and In-System Disable function.

Section 9: Interrupts
TBD

Section 10: Parallel I/O
Descriptions of changes to the I/O characteristics of ports 1, 4, and 5 has been added.

Section 11: Programmable Timers
Information on the new divide by 13 mode has been added, as well as updated figures showing the effect of the clock multiplier modes on the timers. The function of the Clock Output feature and IrDA Clock Output mode is also described.

Section 12: Serial I/O
No Changes.

Section 13: Timed Access Protection
Additional/changed Timed Access bits in the DS80C390 are listed.

Section 14: Real Time Clock
No Changes. Not applicable to the DS80C390.

Section 15: Battery Backup
No Changes. Not applicable to the DS80C390.

Section 16: Instruction Set Details
Modified timing and cycle count of select instructions in paged and contiguous addressing modes is listed.

Section 17: Troubleshooting
No Changes.

Section 18: Controller Area (CAN) Module
Information on the features and use of the CAN module is provided.

Section 19: Arithmetic Accelerator
Information on the features and use of the DS80C390 arithmetic accelerator is provided.

ADDENDUM TO SECTION 2: ORDERING INFORMATION

The High-Speed Microcontroller family follows the part numbering convention shown below. Note that all combinations of devices are not currently available. Please refer to individual data sheets for the available versions.

DS80C390-FCR

SPEED:	D	18 MHz
	G	25 MHz
	L	33 MHz
	R	40 MHz
TEMPERATURE:	C	0°C to 70°C
	N	-40°C to +85°C
PACKAGE:	M	PLASTIC
	Q	PLCC
	E	THIN PLASTIC QUAD FLAT PACK (TQFP)
	F	PLASTIC QUAD FLAT PACK (PQFP)
	W	WINDOWED CERDIP
	K	WINDOWED CERQUAD
OPERATING VOLTAGE:	0	+5V
	3	+3V OR WIDE VOLTAGE
MEMORY TYPE:	0	ROMless
	3	ROM
	7	EPROM

ADDENDUM TO SECTION 4: PROGRAMMING MODEL

The DS80C390 microprocessor is based on the industry standard 80C52. The core is an accumulator-based architecture using internal registers for data storage and peripheral control. It executes the standard 8051 instruction set. This section provides a brief description of each architecture feature. Details concerning the programming model, instruction set, and register description are provided in Section 4.

The High-Speed Microcontroller, uses several distinct memory areas. These are registers, program memory, and data memory. Registers serve to control on-chip peripherals and as RAM. Note that registers (on-chip RAM) are separate from data memory. Registers are divided into three categories including directly addressed on-chip RAM, indirectly addressed on-chip RAM, and Special Function Registers. The program and data memory areas are discussed under Memory Map. The Registers are discussed under Registers Map.

MEMORY MAP

The DS80C390 microprocessor uses a memory addressing scheme that separates program memory (ROM) from data memory (RAM). Each area is accessed via a 20-bit address bus and 4 chip enables, allowing a maximum address space of 4 MB of program memory and 4 MB of data memory. The program and data segments can overlap since they are accessed in different ways. Program memory is fetched by the microprocessor automatically. These addresses are never written by software. There is one instruction (MOVC) that is used to explicitly read the program area. This is commonly used to read look-up tables. The data memory area is accessed explicitly using the MOVX instruction. This instruction provides multiple ways of specifying the target address

REGISTER MAP

The register map is separate from the program and data memory areas mentioned above. A separate class of instructions is used to access the registers. There are 256 potential register location values. In practice, the High-Speed Microcontroller has 256 bytes of Scratchpad RAM and up to 128 Special Function Registers (SFRs). This is possible since the upper 128 Scratchpad RAM locations can only be accessed indirectly. That is, the contents of a Working Register (described below) will designate the RAM location. Thus a direct reference to one of the upper 128 locations must be an SFR access. Direct RAM is reached at locations 0 to 7Fh (0 to 127). SFRs are accessed directly between 80h and FFh (128 to 255). The RAM locations between 128 and 255 can be reached through an indirect reference to those locations.

Scratchpad RAM is available for general-purpose data storage. It is commonly used in place of off-chip RAM when the total data contents are small. When off-chip RAM is needed, the Scratchpad area will still provide the fastest general-purpose access. Within the 256 bytes of RAM, there are several special purpose areas. These are described as follows:

BIT ADDRESSABLE LOCATIONS

In addition to direct register access, some individual bits in both the RAM and SFR area are also accessible. In the Scratchpad RAM area, registers 20h to 2Fh are bit addressable. This provides 126 (16 * 8) individual bits available to software. The type of instruction distinguishes a bit access from a full register access. In the SFR area, any register location ending in a 0 or 8 is bit addressable.

WORKING REGISTERS

As part of the lower 128 bytes of RAM, there are four banks of general-purpose Working Registers, each bank containing registers R0 through R7. The bank is selected via bits in the Program Status Word register. Since there are four banks, the currently selected bank will be used by any instruction using R0-R7. This allows software to change context by simply switching banks. The Working Registers also

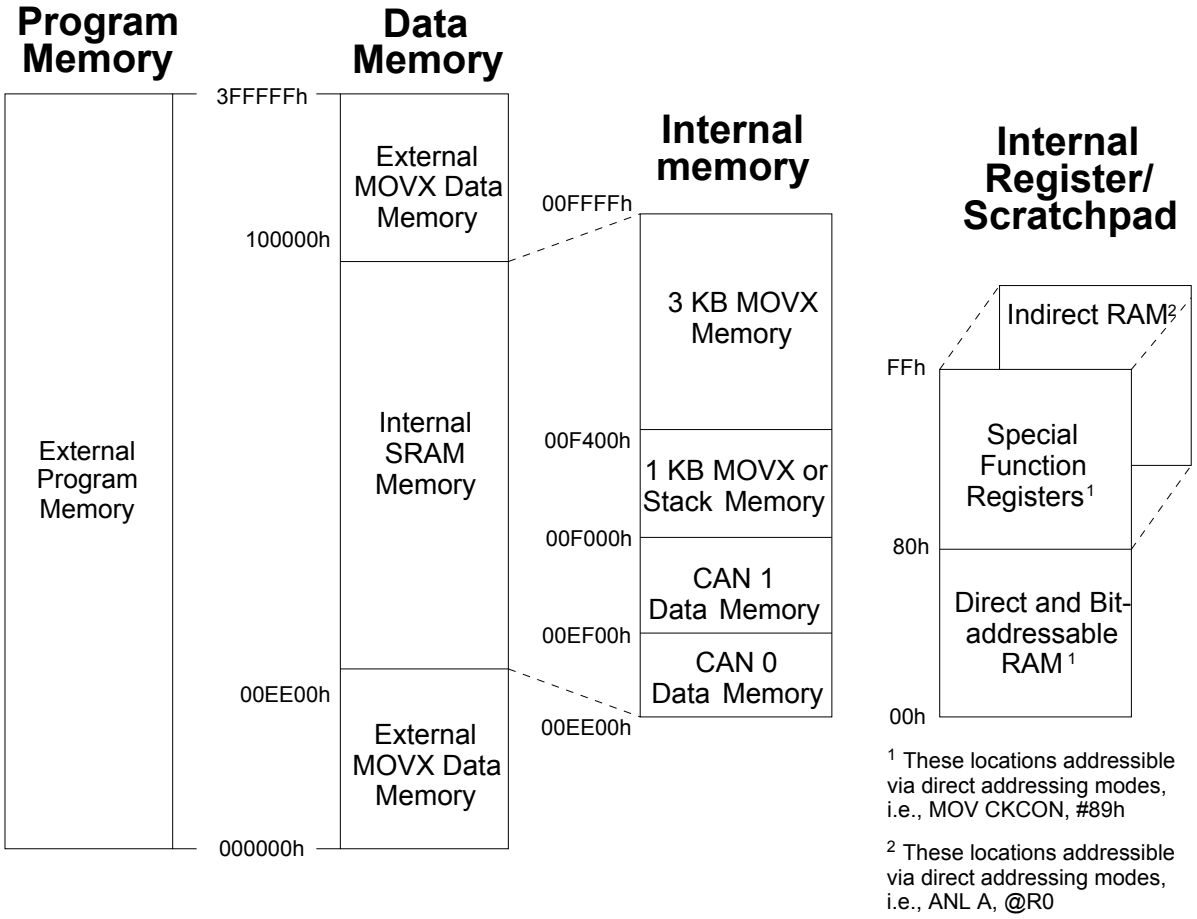
allow their contents to be used for indirect addressing of the upper 128 bytes of RAM. Thus an instruction can designate the value stored in R0 (for example) to address the upper RAM. This value might be the result of another calculation.

STACK

Another use of the Scratchpad area is for the programmer's stack. This area is selected using the Stack Pointer (SP,81h) SFR. Whenever a call or interrupt is invoked, the return address is placed on the stack. It also is available to the programmer for variables, etc. The Stack Pointer will default to 07h on reset, but can be relocated as needed. A convenient location would be the upper RAM area (>7Fh) since this is only available indirectly. The SP will point to the last used value. Therefore, the next value placed on the Stack is put at SP + 1. Each PUSH or CALL will increment the SP by the appropriate value. Each POP or RET will decrement as well.

The DS80C390 supports an optional 10-bit (1 KB) stack. This greatly increases programming efficiency and allows the device to support large programs. When enabled by setting the Stack Address (SA) bit in the ACON register, the lower 1 KB of the 4 KB internal SRAM becomes the memory location used by all instructions that affect the stack. The 10-bit address is formed by concatenating the lower 2 bits of the Extended Stack Pointer (ESP;9Bh) and the 8-bit Stack Pointer (SP;81h). The exact address of the 1 KB is dependent on the setting of the IDM1-0 bits. The 10-bit stack feature is not supported when the 4 KB SRAM is configured as combined program/data memory (IDM1=IDM0=1)

Figure 4- 1 DS80C390 Memory Map (Default Settings)



SPECIAL FUNCTION REGISTERS

Most of the unique features of the High-Speed Microcontroller family are controlled by bits in special function registers (SFRs) located in unused locations in the 8051 SFR map. This allows for increased functionality while maintaining complete instruction set compatibility. The SFRs reside in register locations 80h-FFh and are accessed using direct addressing. SFRs that end in 0 or 8 are bit addressable.

The first table indicates the names and locations of the SFRs used by the DS80C390 and individual bits in those registers. Bits protected by the Timed Access function are shaded. The second table indicates the reset state of all SFR bits. Following these tables is a complete description of DS80C390 SFRs that are new to the 8051 architecture or have new or modified functionality.

SPECIAL FUNCTION REGISTER LOCATION TABLE 4-1

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Address
P4	A19/P4.7	A18/P4.6	A17/P4.5	A16/P4.4	$\overline{CE3}$/P4.3	$\overline{CE2}$/P4.2	$\overline{CE1}$/P4.1	$\overline{CE0}$/P4.0	80h
SP									81h
DPL									82h
DPH									83h
DPL1									84h
DPH1									85h
DPS	ID1	ID0	TSL	-	-	-	-	SEL	86h
PCON	SMOD_0	SMOD0	OFDF	OFDE	GF1	GF0	STOP	IDLE	87h
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	88h
TMOD	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	89h
TL0									8Ah
TL1									8Bh
TH0									8Ch
TH1									8Dh
CKCON	WD1	WD0	T2M	T1M	T0M	MD2	MD1	MD0	8Eh
P1	INT5/P1.7	INT4/P1.6	INT3/P1.5	INT2/P1.4	TXD1/P1.3	RXD1/P1.2	T2EX/P1.1	T2/P1.0	90h
EXIF	IE5	IE4	IE3	IE2	CKRY	RGMD	RGSL	BGS	91h
P4CNT	-	SBCAN	P4CNT.5	P4CNT.4	P4CNT.3	P4CNT.2	P4CNT.1	P4CNT.0	92h
DPX									93h
DPX1									95h
C0RMS0									96h
C0RMS1									97h
SCON0	SM0/FE_0	SM1_0	SM2_0	REN_0	TB8_0	RB8_0	TI_0	RI_0	98h
SBUF0									99h
ESP	-	-	-	-	-	-	ESP.1	ESP.0	9Bh
AP									9Ch
ACON	-	-	-	-	-	SA	AM1	AM0	9Dh
C0TMA0									9Eh
C0TMA1									9Fh
P2	A15/P2.7	A14/P2.6	A13/P2.5	A12/P2.4	A11/P2.3	A10/P2.2	A9/P2.1	A8/P2.0	A0h
P5	$\overline{PCE3}$/P5.7	$\overline{PCE2}$/P5.6	$\overline{PCE1}$/P5.5	$\overline{PCE0}$/P5.4	C1TX/P5.3	C1RX/P5.2	C0RX/P5.1	C0TX/P5.0	A1h
P5CNT	CAN1BA	CAN0BA	SP1EC	C1_I/O	C0_I/O	P5CNT.2	P5CNT.1	P5CNT.0	A2h
C0C	ERIE	STIE	PDE	SIESTA	CRST	AUTOB	ERCS	SWINT	A3h
C0S	BUSOFF	CECE	WKS	RXS	TXS	ER2	ER1	ER0	A4h
C0IR	INTIN7	INTIN6	INTIN5	INTIN4	INTIN3	INTIN2	INTIN1	INTIN0	A5h
C0TE									A6h
C0RE									A7h
IE	EA	ES1	ET2	ES0	ET1	EX1	ET0	EX0	A8h

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Address
SADDR0									A9h
SADDR1									AAh
C0M1C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	ABh
C0M2C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	ACh
C0M3C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	ADh
C0M4C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	AEh
C0M5C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	AFh
P3	$\overline{RD}$ /P3.7	$\overline{WR}$ /P3.6	T1/P3.5	T0/P3.4	INT1/P3.3	INT0/P3.2	TXD0/P3.1	RXD0/P3.0	B0h
C0M6C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	B3h
C0M7C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	B4h
C0M8C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	B5h
C0M9C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	B6h
C0M10C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	B7h
IP	-	PS1	PT2	PS0	PT1	PX1	PT0	PX0	B8h
SADEN0									B9h
SADEN1									BAh
C0M11C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	BBh
C0M12C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	BCh
C0M13C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	BDh
C0M14C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	BEh
C0M15C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	BFh
SCON1	SM0/FE_1	SM1_1	SM2_1	REN_1	TB8_1	RB8_1	TI_1	RI_1	C0h
SBUF1									C1h
PMR	CD1	CD0	SWB	CTM	4X/ $\overline{2X}$	ALEOFF	-	-	C4h
STATUS	PIP	HIP	LIP	-	SPTA1	SPRA1	SPTA0	SPRA0	C5h
MCON	IDM1	IDM0	CMA	-	PDCE3	PDCE2	PDCE1	PDCE0	C6h
TA									C7h
T2CON	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/ $\overline{T2}$	CP/ $\overline{RL2}$	C8h
T2MOD	-	-	-	D13T1	D13T2	-	T2OE	DCEN	C9h
RCAP2L									CAh
RCAP2H									CBh
TL2									CCh
TH2									CDh
COR	IRDACK	C1BPR7	C1BPR6	C0BPR7	C0BPR6	COD1	COD0	CLKOE	CEh
PSW	CY	AC	F0	RS1	RS0	OV	F1	P	D0h
MCNT0	$\overline{LSHIFT}$	CSE	SCB	MAS4	MAS3	MAS2	MAS1	MAS0	D1h
MCNT1	MST	MOF	-	CLM	-	-	-	-	D2h
MA									D3h
MB									D4h
MC									D5h
C1RMS0									D6h
C1RMS1									D7h
WDCON	SMOD_1	POR	EPF1	PF1	WDIF	WTRF	EWT	RWT	D8h
C1TMA0									DEh
C1TMA1									DFh
ACC									E0h
C1C	ERIE	STIE	PDE	SIESTA	CRST	AUTOB	ERCS	SWINT	E3h
C1S	BUSOFF	CECE	WKS	RXS	TXS	ER2	ER1	ER0	E4h
C1IR	INTIN7	INTIN6	INTIN5	INTIN4	INTIN3	INTIN2	INTIN1	INTIN0	E5h
C1TE									E6h
C1RE									E7h
EIE	CANBIE	C0IE	C1IE	EWDI	EX5	EX4	EX3	EX2	E8h
MXAX									EAh

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Address
C1M1C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	EBh
C1M2C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	ECh
C1M3C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	EDh
C1M4C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	EEh
C1M5C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	EFh
B									F0h
C1M6C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	F3h
C1M7C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	F4h
C1M8C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	F5h
C1M9C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	F6h
C1M10C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	F7h
EIP	CANBIP	C0IP	C1IP	PWDI	PX5	PX4	PX3	PX2	F8h
C1M11C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	FBh
C1M12C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	FCh
C1M13C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	FDh
C1M14C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	FEh
C1M15C	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP	FFh

SPECIAL FUNCTION REGISTER RESET VALUES TABLE 4-2

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Address
P4	1	1	1	1	1	1	1	1	80h
SP	0	0	0	0	0	1	1	1	81h
DPL	0	0	0	0	0	0	0	0	82h
DPH	0	0	0	0	0	0	0	0	83h
DPL1	0	0	0	0	0	0	0	0	84h
DPH1	0	0	0	0	0	0	0	0	85h
DPS	0	0	0	0	0	1	0	0	86h
PCON	0	0	SPECIAL	SPECIAL	0	0	0	0	87h
TCON	0	0	0	0	0	0	0	0	88h
TMOD	0	0	0	0	0	0	0	0	89h
TL0	0	0	0	0	0	0	0	0	8Ah
TL1	0	0	0	0	0	0	0	0	8Bh
TH0	0	0	0	0	0	0	0	0	8Ch
TH1	0	0	0	0	0	0	0	0	8Dh
CKCON	0	0	0	0	0	0	0	1	8Eh
P1	1	1	1	1	1	1	1	1	90h
EXIF	0	0	0	0	SPECIAL	SPECIAL	SPECIAL	0	91h
P4CNT	1	0	1	1	1	1	1	1	92h
DPX	0	0	0	0	0	0	0	0	93h
DPX1	0	0	0	0	0	0	0	0	95h
C0RMS0	0	0	0	0	0	0	0	0	96h
C0RMS1	0	0	0	0	0	0	0	0	97h
SCON0	0	0	0	0	0	0	0	0	98h
SBUF0	0	0	0	0	0	0	0	0	99h
ESP	1	1	1	1	1	1	0	0	9Bh
AP	0	0	0	0	0	0	0	0	9Ch
ACON	1	1	1	1	1	0	0	0	9Dh
C0TMA0	0	0	0	0	0	0	0	0	9Eh
C0TMA1	0	0	0	0	0	0	0	0	9Fh
P2	1	1	1	1	1	1	1	1	A0h

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Address
P5	1	1	1	1	1	1	1	1	A1h
P5CNT	0	0	0	0	0	0	0	0	A2h
C0C	0	0	0	0	1	0	0	1	A3h
C0S	0	0	0	0	0	0	0	0	A4h
C0IR	0	0	0	0	0	0	0	0	A5h
C0TE	0	0	0	0	0	0	0	0	A6h
C0RE	0	0	0	0	0	0	0	0	A7h
IE	0	0	0	0	0	0	0	0	A8h
SADDR0	0	0	0	0	0	0	0	0	A9h
SADDR1	0	0	0	0	0	0	0	0	AAh
C0M1C	0	0	0	0	0	0	0	0	ABh
C0M2C	0	0	0	0	0	0	0	0	ACh
C0M3C	0	0	0	0	0	0	0	0	ADh
C0M4C	0	0	0	0	0	0	0	0	AEh
C0M5C	0	0	0	0	0	0	0	0	AFh
P3	1	1	1	1	1	1	1	1	B0h
C0M6C	0	0	0	0	0	0	0	0	B3h
C0M7C	0	0	0	0	0	0	0	0	B4h
C0M8C	0	0	0	0	0	0	0	0	B5h
C0M9C	0	0	0	0	0	0	0	0	B6h
C0M10C	0	0	0	0	0	0	0	0	B7h
IP	1	0	0	0	0	0	0	0	B8h
SADEN0	0	0	0	0	0	0	0	0	B9h
SADEN1	0	0	0	0	0	0	0	0	BAh
C0M11C	0	0	0	0	0	0	0	0	BBh
C0M12C	0	0	0	0	0	0	0	0	BCh
C0M13C	0	0	0	0	0	0	0	0	BDh
C0M14C	0	0	0	0	0	0	0	0	BEh
C0M15C	0	0	0	0	0	0	0	0	BFh
SCON1	0	0	0	0	0	0	0	0	C0h
SBUF1	0	0	0	0	0	0	0	0	C1h
PMR	1	0	0	0	0	0	1	1	C4h
STATUS	0	0	0	1	0	0	0	0	C5h
MCON	0	0	0	1	0	0	0	0	C6h
TA	1	1	1	1	1	1	1	1	C7h
T2CON	0	0	0	0	0	0	0	0	C8h
T2MOD	1	1	1	0	0	1	0	0	C9h
RCAP2L	0	0	0	0	0	0	0	0	CAh
RCAP2H	0	0	0	0	0	0	0	0	CBh
TL2	0	0	0	0	0	0	0	0	CCh
TH2	0	0	0	0	0	0	0	0	CDh
COR	0	0	0	0	0	0	0	0	CEh
PSW	0	0	0	0	0	0	0	0	D0h
MCNT0	0	0	0	0	0	0	0	0	D1h
MCNT1	0	0	1	0	1	1	1	1	D2h
MA	0	0	0	0	0	0	0	0	D3h
MB	0	0	0	0	0	0	0	0	D4h
MC	0	0	0	0	0	0	0	0	D5h
C1RMS0	0	0	0	0	0	0	0	0	D6h
C1RMS1	0	0	0	0	0	0	0	0	D7h
WDCON	0	SPECIAL	0	SPECIAL	0	SPECIAL	SPECIAL	0	D8h
C1TMA0	0	0	0	0	0	0	0	0	DEh
C1TMA1	0	0	0	0	0	0	0	0	DFh
ACC	0	0	0	0	0	0	0	0	E0h

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Address
C1C	0	0	0	0	1	0	0	1	E3h
C1S	0	0	0	0	0	0	0	0	E4h
C1IR	0	0	0	0	0	0	0	0	E5h
C1TE	0	0	0	0	0	0	0	0	E6h
C1RE	0	0	0	0	0	0	0	0	E7h
EIE	0	0	0	0	0	0	0	0	E8h
MXAX	0	0	0	0	0	0	0	0	EAh
C1M1C	0	0	0	0	0	0	0	0	EBh
C1M2C	0	0	0	0	0	0	0	0	ECh
C1M3C	0	0	0	0	0	0	0	0	EDh
C1M4C	0	0	0	0	0	0	0	0	EEh
C1M5C	0	0	0	0	0	0	0	0	EFh
B	0	0	0	0	0	0	0	0	F0h
C1M6C	0	0	0	0	0	0	0	0	F3h
C1M7C	0	0	0	0	0	0	0	0	F4h
C1M8C	0	0	0	0	0	0	0	0	F5h
C1M9C	0	0	0	0	0	0	0	0	F6h
C1M10C	0	0	0	0	0	0	0	0	F7h
EIP	0	0	0	0	0	0	0	0	F8h
C1M11C	0	0	0	0	0	0	0	0	FBh
C1M12C	0	0	0	0	0	0	0	0	FCh
C1M13C	0	0	0	0	0	0	0	0	FDh
C1M14C	0	0	0	0	0	0	0	0	FEh
C1M15C	0	0	0	0	0	0	0	0	FFh

Port 4 (P4)

	7	6	5	4	3	2	1	0
SFR 80h	A19/P4.7	A18/P4.6	A17/P4.5	A18/P4.4	$\overline{CE3}$/P4.3	$\overline{CE2}$/P4.2	$\overline{CE1}$/P4.1	$\overline{CE0}$/P4.0
	RW-0	RW-0	RW-0	RW-0	RW-1	RW-1	RW-1	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

P4.7-0 **Port 4.** This port functions as a general-purpose I/O port. In addition, all the pins have an alternative function associated with the memory interface described below. The selection of general I/O or memory interface function for the Port 4 pins is controlled via the P4CNT(92h) register. Port pins configured as I/O will reflect the state of the corresponding port pin. Port pins assigned to memory interface functions will appear as 1 when read. The associated SFR bit must be programmed to a logic one before the pin can be used in its alternate function capacity. The reset state of this register and the P4CNT register will configure the device to so that A19-A16 function as address lines and $\overline{CE0} - \overline{CE3}$ are active. The first opcode fetch following a reset will therefore be at 00000h with $\overline{CE0}$ asserted.

A19
Bit 7 **Program/Data Memory Address 19.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the A19 memory signal.

A18
Bit 6 **Program/Data Memory Address 18.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the A18 memory signal.

A17
Bit 5 **Program/Data Memory Address 17.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the A17 memory signal.

A16
Bit 4 **Program/Data Memory Address 16.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the A16 memory signal.

$\overline{CE3}$
Bit 3 **Program Memory Chip Enable 3.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the $\overline{CE3}$ memory signal.

$\overline{CE2}$
Bit 2 **Program Memory Chip Enable 2.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the $\overline{CE2}$ memory signal.

$\overline{CE1}$
Bit 1 **Program Memory Chip Enable 1.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the $\overline{CE1}$ memory signal.

$\overline{CE0}$
Bit 0 **Program Memory Chip Enable 0.** When this bit is set to a logic one and the P4CNT register is configured correctly, the corresponding device pin will represent the $\overline{CE0}$ memory signal.

Stack Pointer (SP)

	7	6	5	4	3	2	1	0
SFR 81h	SP.7	SP.6	SP.5	SP.4	SP.3	SP.2	SP.1	SP.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-1	RW-1	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SP.7-0 Bits 7-0 — **Stack Pointer.** This stack pointer identifies current location of the stack. The stack pointer is incremented before every PUSH operation. This register defaults to 07h after reset. When the 10-bit stack is enabled (SA=1), this register will be combined with the extended stack pointer (ESP;9Bh) to form the 10-bit address.

Data Pointer Low 0 (DPL)

	7	6	5	4	3	2	1	0
SFR 82h	PDL.7	PDL.6	PDL.5	PDL.4	PDL.3	PDL.2	PDL.1	PDL.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

DPL.7-0 Bits 7-0 — **Data Pointer Low 0.** This register is the low byte of the standard 80C32 16-bit data pointer. DPL and DPH are used to point to non-scratchpad data RAM.

Data Pointer High 0 (DPH)

	7	6	5	4	3	2	1	0
SFR 83h	DPH.7	DPH.6	DPH.5	DPH.4	DPH.3	DPH.2	DPH.1	DPH.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

DPH.7-0 Bits 7-0 — **Data Pointer High 0.** This register is the high byte of the standard 80C32 16-bit data pointer. DPL and DPH are used to point to non-scratchpad data RAM.

Data Pointer Low 1 (DPL1)

	7	6	5	4	3	2	1	0
SFR 84h	DPL1.7	DPL1.6	DPL1.5	DPL1.4	DPL1.3	DPL1.2	DPL1.1	DL1H.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

DPL1.7-0 Bits 7-0 — **Data Pointer Low 1.** This register is the low byte of the auxiliary 16-bit data pointer. When the SEL bit (DPS.0) is set, DPL1 and DPH1 are used in place of DPL and DPH during DPTR operations.

Data Pointer High 1 (DPH1)

	7	6	5	4	3	2	1	0
SFR 85h	DPH1.7	DPH1.6	DPH1.5	DPH1.4	DPH1.3	DPH1.2	DPH1.1	DPH1.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

DPH1.7-0
Bits 7-0

Data Pointer High 1. This register is the high byte of the auxiliary 16-bit data pointer. When the SEL bit (DPS.0) is set, DPL1 and DPH1 are used in place of DPL and DPH during DPTR operations.

Data Pointer Select (DPS)

	7	6	5	4	3	2	1	0
SFR 86h	ID1	ID0	TSL	0	0	1	0	SEL
	RW-0	RW-0	RW-0	R-0	R-0	R-1	R-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

ID1, ID0
Bits 7-6

Increment/Decrement Function Select. These bits define whether the INC DTPR instruction will increment or decrement the active data pointer as selected by the SEL bit.

ID1	ID0	SEL=0	SEL=1
0	0	Increment DPTR	Increment DPTR1
0	1	Decrement DPTR	Increment DPTR1
1	0	Increment DPTR	Decrement DPTR1
1	1	Decrement DPTR	Decrement DPTR1

TSL
Bit 5

Toggle Select Enable. When set, this bit allows the following five DPTR-related instructions to toggle the SEL bit following execution of the instruction. When TSL=0, DPTR-related instructions will not affect the state of the SEL bit. DPTR-related instructions are:

```
INC  DPTR
MOV  DPTR, #data16
MOVC A, @A+DPTR
MOVX @DPTR, A
MOVX A, @DPTR
```

Bits 4-1

Reserved.

SEL
Bit 0

Data Pointer Select. This bit selects the active data pointer.

0 = Instructions that use the DPTR will use DPL, DPH, DPX.
1= Instructions that use the DPTR will use DPL1 and DPH1, DPX1.

Power Control (PCON)

	7	6	5	4	3	2	1	0
SFR 87h	SMOD_0	SMOD0	OFDF	ODFE	FG1	FG0	STOP	IDLE
	RW-0	RW-0	RW-0*	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset, *=See description

SMOD_0
Bit 7

Serial Port 0 Baud Rate Doubler Enable. This bit enables/disables the serial baud rate doubling function for Serial Port 0.

0 = Serial Port 0 baud rate will be that defined by baud rate generation equation.

1 = Serial Port 0 baud rate will be double that defined by baud rate generation equation.

SMOD0
Bit 6

Framing Error Detection Enable. This bit selects function of the SCON0.7 and SCON1.7 bits.

0 = SCON0.7 and SCON1.7 control the SM0 function defined for the SCON0 and SCON1 registers.

1 = SCON0.7 and SCON1.7 are converted to the Framing Error (FE) flag for the respective Serial Port.

OFDF
Bit 5

Oscillator Fail Detect Flag. When set, this bit indicates that the preceding reset was caused by the detection of the crystal oscillator frequency falling below approximately 30 kHz while the OFDE bit was set. This bit must be cleared by software. This bit not altered (and no reset will be generated) under the following conditions:

1. OFDE=0
2. An oscillator halt associated with entering STOP mode.
3. An oscillator halt associated with running from the internal ring oscillator.

OFDE
Bit 4

Oscillator Fail Detect Enable. When the OFDE=1, a system reset will be generated any time the crystal oscillator frequency falls below approximately 30 kHz. When the OFDE bit is cleared to a logic 0, no reset will be issued when the crystal falls below 30 kHz. The OFDE is cleared to a logic 0 by any reset source.

GF1
Bit 3

General Purpose User Flag 1. This is a bit-addressable, general-purpose flag for software control.

GF0
Bit 2

General Purpose User Flag 0. This is a bit-addressable, general-purpose flag for software control.

STOP
Bit 1

Stop Mode Select. Setting this bit will stop program execution, halt the CPU oscillator, and internal timers, and place the CPU in a low-power mode. This bit will always be read as a 0. Setting this bit while the IDLE=1 will place the device in an undefined state.

IDLE
Bit 0

Idle Mode Select. Setting this bit will stop program execution but leave the CPU oscillator, timers, serial ports, and interrupts active. This bit will always be read as a 0.

Timer/Counter Control (TCON)

	7	6	5	4	3	2	1	0
SFR 88h	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TF1
Bit 7

Timer 1 Overflow Flag. This bit indicates when Timer 1 overflows its maximum count as defined by the current mode. This bit can be cleared by software and is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine.

0 = No Timer 1 overflow has been detected.
1 = Timer 1 has overflowed its maximum count.

TR1
Bit 6

Timer 1 Run Control. This bit enables/disables the operation of Timer 1.

0 = Timer 1 is halted.
1 = Timer 1 is enabled.

TF0
Bit 5

Timer 0 Overflow Flag. This bit indicates when Timer 0 overflows its maximum count as defined by the current mode. This bit can be cleared by software and is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine or by software.

0 = No Timer 0 overflow has been detected.
1 = Timer 0 has overflowed its maximum count.

TR0
Bit 4

Timer 0 Run Control. This bit enables/disables the operation of Timer 0.

0 = Timer 0 is halted.
1 = Timer 0 is enabled.

IE1
Bit 3

Interrupt 1 Edge Detect. This bit is set when an edge/level of the type defined by IT1 is detected. If IT1=1, this bit will remain set until cleared in software or the start of the External Interrupt 1 service routine. If IT1=0, this bit will inversely reflect the state of the $\overline{\text{INT1}}$ pin.

IT1
Bit 2

Interrupt 1 Type Select. This bit selects whether the $\overline{\text{INT1}}$ pin will detect edge or level triggered interrupts.

0 = $\overline{\text{INT1}}$ is level triggered.
1 = $\overline{\text{INT1}}$ is edge triggered.

IE0
Bit 1

Interrupt 0 Edge Detect. This bit is set when an edge/level of the type defined by IT0 is detected. If IT0=1, this bit will remain set until cleared in software or the start of the External Interrupt 0 service routine. If IT0=0, this bit will inversely reflect the state of the $\overline{\text{INT0}}$ pin.

IT0
Bit 0

Interrupt 0 Type Select. This bit selects whether the $\overline{\text{INT0}}$ pin will detect edge or level triggered interrupts.

0 = $\overline{\text{INT0}}$ is level triggered.
1 = $\overline{\text{INT0}}$ is edge triggered.

Timer Mode Control (TMOD)

	7	6	5	4	3	2	1	0
SFR 89h	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

GATE
Bit 7

Timer 1 Gate Control. This bit enable/disables the ability of Timer 1 to increment.

0 = Timer 1 will clock when TR1=1, regardless of the state of $\overline{INT1}$.
1 = Timer 1 will clock only when TR1=1 and $\overline{INT1}$=1.

C/$\overline{T}$
Bit 6

Timer 1 Counter/Timer Select.

0 = Timer 1 is incremented by internal clocks.
1 = Timer 1 is incremented by pulses on T1 when TR1 (TCON.6) is 1.

M1, M0
Bits 5-4

Timer 1 Mode Select. These bits select the operating mode of Timer 1.

M1	M0	Mode
0	0	Mode 0: 8 bits with 5-bit prescale
0	1	Mode 1: 16 bits
1	0	Mode 2: 8 bits with auto-reload
1	1	Mode 3: Timer 1 is halted, but holds its count

GATE
Bit 3

Timer 0 Gate Control. This bit enables/disables that ability of Timer 0 to increment.

0 = Timer 0 will clock when TR0=1, regardless of the state of $\overline{INT0}$.
1 = Timer 0 will clock only when TR0=1 and $\overline{INT0}$=1.

C/$\overline{T}$
Bit 2

Timer 0 Counter/Timer Select.

0 = Timer incremented by internal clocks.
1 = Timer 1 is incremented by pulses on T0 when TR0 (TCON.4) is 1.

M1, M0
Bits 1-0

Timer 0 Mode Select. These bits select the operating mode of Timer 0. When Timer 0 is in mode 3, TL0 is started/stopped by TR0 and TH0 is started/stopped by TR1. Run control from Timer 1 is then provided via the Timer 1 mode selection.

M1	M0	Mode
0	0	Mode 0: 8 bits with 5-bit prescale
0	1	Mode 1: 16 bits
1	0	Mode 2: 8 bits with auto-reload
1	1	Mode 3: Timer 0 is two 8 bit counters.

Timer 0 LSB (TL0)

	7	6	5	4	3	2	1	0
SFR 8Ah	TL0.7	TL0.6	TL0.5	TL0.4	TL0.3	TL0.2	TL0.1	TL0.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TL0.7-0 **Timer 0 LSB.** This register contains the least significant byte of Timer 0.
Bits 7-0

Timer 1 LSB (TL1)

	7	6	5	4	3	2	1	0
SFR 8Bh	TL1.7	TL1.6	TL1.5	TL1.4	TL1.3	TL1.2	TL1.1	TL1.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TL1.7-0 **Timer 1 LSB.** This register contains the least significant byte of Timer 1.
Bits 7-0

Timer 0 MSB (TH0)

	7	6	5	4	3	2	1	0
SFR 8Ch	TH0.7	TH0.6	TH0.5	TH0.4	TH0.3	TH0.2	TH0.1	TH0.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TH0.7-0 **Timer 0 MSB.** This register contains the most significant byte of Timer 0.
Bits 7-0

Timer 1 MSB (TH1)

	7	6	5	4	3	2	1	0
SFR 8Dh	TH1.7	TH1.6	TH1.5	TH1.4	TH1.3	TH1.2	TH1.1	TH1.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TH1.7-0 **Timer 1 MSB.** This register contains the most significant byte of Timer 1.
Bits 7-0

Clock Control (CKCON)

	7	6	5	4	3	2	1	0
SFR 8Eh	WD1	WD0	T2M	T1M	T0M	MD2	MD1	MD0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

WD1, WD0
Bits 7-6

Watchdog Timer Mode Select 1-0. These bits select the watchdog timer time-out period, which determines the timing of the watchdog timer interrupt and the watchdog timer reset.

WD1	WD0	Interrupt time-out	Reset time-out
0	0	2^{17} system clocks	2^{17} + 512 system clocks
0	1	2^{20} system clocks	2^{20} + 512 system clocks
1	0	2^{23} system clocks	2^{23} + 512 system clocks
1	1	2^{26} system clocks	2^{26} + 512 system clocks

The system clock relates to the external clock as follows:

Clock Mode	External clocks per system clock
Frequency Multiplier (4x)	0.25
Frequency Multiplier (2x)	0.5
Divide by 4	1
Power Management Mode	256

T2M
Bit 5

Timer 2 Clock Select. This bit controls the division of the system clock that drives Timer 2. This bit has no effect when the timer is in baud rate generator or clock output modes. Clearing this bit to 0 maintains 80C32 compatibility. This bit has no effect on instruction cycle timing.

0 = Timer 2 uses a divide by 12 of the crystal frequency.
1 = Timer 2 uses a divide by 4 of the crystal frequency.

T1M
Bit 4

Timer 1 Clock Select. This bit controls the division of the system clock that drives Timer 1. Clearing this bit to 0 maintains 80C32 compatibility. This bit has no effect on instruction cycle timing.

0 = Timer 1 uses a divide by 12 of the crystal frequency.
1 = Timer 1 uses a divide by 4 of the crystal frequency.

T0M
Bit 3

Timer 0 Clock Select. This bit controls the division of the system clock that drives Timer 0. Clearing this bit to 0 maintains 80C32 compatibility. This bit has no effect on instruction cycle timing.

0 = Timer 0 uses a divide by 12 of the crystal frequency.
1 = Timer 0 uses a divide by 4 of the crystal frequency.

MD2, MD1, MD0
Bits 2-0

Stretch MOVX Select 2-0. These bits select the time by which external MOVX cycles are to be stretched. This allows slower memory or peripherals to be accessed without using ports or manual software intervention. The $\overline{RD}$ or $\overline{WR}$ strobe will be stretched by the specified interval, which will be transparent to the software except for the increased time to execute to MOVX instruction. All internal MOVX instructions are performed at the 2 machine cycle rate.

MD2	MD1	MD0	Stretch Value	MOVX Duration
0	0	0	0	2 Machine Cycles
0	0	1	1	3 Machine Cycles (reset default)
0	1	0	2	4 Machine Cycles
0	1	1	3	5 Machine Cycles
1	0	0	4	9 Machine Cycles
1	0	1	5	10 Machine Cycles
1	1	0	6	11 Machine Cycles
1	1	1	7	12 Machine Cycles

Port 1 (P1)

	7	6	5	4	3	2	1	0
SFR 90h	P1.7 $\overline{INT5}$	P1.6 INT4	P1.7 $\overline{INT3}$	P1.4 INT2	P1.3 TXD1	P1.2 RXD1	P1.1 T2EX	P1.0 T2
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

P1.7-0
Bits 7-0

General Purpose I/O Port 1. This register functions as the A0-A7 of the non-multiplexed address bus (when the $\overline{MUX}$ pin=1) or a general purpose I/O port (when the $\overline{MUX}$ pin=0). When serving as a general purpose I/O port all the pins have an alternative function listed below. P1.2-7 contain functions that are new to the 80C32 architecture. The Timer 2 functions on pins P1.1-0 are available on the 80C32, but not the 80C31. Each of the functions is controlled by several other SFRs. The associated Port 1 latch bit must contain a logic one before the pin can be used in its alternate function capacity.

$\overline{INT5}$
Bit 7

External Interrupt 5. A falling edge on this pin will cause an external interrupt 5 if enabled.

INT4
Bit 6

External Interrupt 4. A rising edge on this pin will cause an external interrupt 4 if enabled.

$\overline{INT3}$
Bit 5

External Interrupt 3. A falling edge on this pin will cause an external interrupt 3 if enabled.

INT2
Bit 4

External Interrupt 2. A rising edge on this pin will cause an external interrupt 2 if enabled.

TXD1
Bit 3

Serial Port 1 Transmit. This pin transmits the serial port 1 data in serial port modes 1, 2, 3 and emits the synchronizing clock in serial port mode 0.

RXD1
Bit 2

Serial Port 1 Receive. This pin receives the serial port 1 data in serial port modes 1, 2, 3 and is a bi-directional data transfer pin in serial port mode 0.

T2EX
Bit 1

Timer 2 Capture/Reload Trigger. A 1 to 0 transition on this pin will cause the value in the T2 registers to be transferred into the capture registers if enabled by EXEN2 (T2CON.3). When in auto–reload mode, a 1 to 0 transition on this pin will reload the timer 2 registers with the value in RCAP2L and RCAP2H if enabled by EXEN2 (T2CON.3).

T2
Bit 0

Timer 2 External Input. A 1 to 0 transition on this pin will cause timer 2 increment or decrement depending on the timer configuration.

External Interrupt Flag (EXIF)

	7	6	5	4	3	2	1	0
SFR 91h	IE5	IE4	IE3	IE2	CKRY	RGMD	RGSL	BGS
	RW-0	RW-0	RW-0	RW-0	R-*	R-*	RW-*	RT-0

R=Unrestricted Read, W=Unrestricted Write, T=Timed Access Write Only
-n=Value after Reset, *=See description

IE5
Bit 7

External Interrupt 5 Flag. This bit will be set when a falling edge is detected on $\overline{\text{INT5}}$. This bit must be cleared manually by software. Setting this bit in software will cause an interrupt if enabled.

IE4
Bit 6

External Interrupt 4 Flag. This bit will be set when a rising edge is detected on INT4. This bit must be cleared manually by software. Setting this bit in software will cause an interrupt if enabled.

IE3
Bit 5

External Interrupt 3 Flag. This bit will be set when a falling edge is detected on $\overline{\text{INT3}}$ This bit must be cleared manually by software. Setting this bit in software will cause an interrupt if enabled.

IE2
Bit 4

External Interrupt 2 Flag. This bit will be set when a rising edge is detected on INT2. This bit must be cleared manually by software. Setting this bit in software will cause an interrupt if enabled.

CKRY
Bit 3

Clock Ready. The CKRY bit indicates the status of the start-up period delay used by the crystal oscillator and the crystal clock multiplier warm-up period. CKRY=0 indicates the start-up delay is still counting. When the CKRY=1 the counter has completed. This bit is cleared each time the CTM bit in the PMR register is changed from low to high to start the crystal multiplier. Once the CKRY is set, the lockout is removed on the CD1, CD0 bits to select the multiplied crystal clock as a system clock source. This status bit is also cleared each time the crystal oscillator is restarted when exiting Stop mode.

RGMD
Bit 2

Ring Mode Status. This bit indicates the current clock source for the device. This bit is cleared to 0 after a power-on reset, and unchanged by all other forms of reset.

0 = Device is operating from the external crystal or oscillator.
1 = Device is operating from the ring oscillator.

RGSL
Bit 1

Ring Oscillator Select. This bit selects the clock source following a resume from Stop mode. Using the ring oscillator to resume from Stop mode allows almost instantaneous start-up. This bit is cleared to 0 after a power-on reset, and unchanged by all other forms of reset. The state of this bit will be undefined on devices which do not incorporate a ring oscillator.

0 = The device will hold operation until the crystal oscillator has warmed-up.

1 = The device will begin operating from the ring oscillator, and when the crystal warm-up is complete, will switch to the external clock source or oscillator.

BGS
Bit 0

Band-gap Select. This bit enables/disables the band-gap reference during Stop mode. Disabling the band-gap reference provides significant power savings in Stop mode, but sacrifices the ability to perform a power fail interrupt or power-fail reset while stopped. This bit can only be modified with a Timed Access procedure.

0 =The band-gap reference is disabled in Stop mode but will function during normal operation.

1 = The band-gap reference will operate in Stop mode.

Port 4 Control Register (P4CNT)

	7	6	5	4	3	2	1	0
SFR 92h	1	SBCAN	P4CNT.5	P4CNT.4	P4CNT.3	P4CNT.2	P4CNT.1	P4CNT.0
	R-1	RT-0	RT-1	RT-1	RT-1	RT-1	RT-1	RT-1

R=Unrestricted Read, T=Timed Access Write Only, -n=Value after Reset

P4.7-0 **Port 4 Control Register.** This register controls the alternate addressing modes function of Port 4. Programming this register as shown below will assign the alternate functions of Port 4. The associated Port 4 SFR bit must be programmed to a logic one before the pin can be used in its alternate function capacity.

Bit 7 Reserved

SBCAN Bit 6 **Single Bus CAN.** Setting this bit connects both CAN receive inputs (C0RX and C1RX) to P5.1 and drives P5.0 with the logical AND of both CAN transmit outputs (C0TX and C1TX). SBCAN=0 disables the feature and allows the CAN modules to receive/transmit through their respective bus pins. This can be used to create a single "super" CAN module with 30 message centers.

P4CNT.5-P4CNT.3 **Port Pin P4.7-4 Configuration Control Bits**

Bits 5-0 configure the external memory control signals. P4CNT.5-3 determine whether specific P4 pins function as A19-A16 or I/O. The number of external address lines enabled establishes the range for each program chip enable ($\overline{CE0-3}$) and data chip enable ($\overline{PCE0-3}$). When P4CNT.5-3=000b, $\overline{CE0}-\overline{CE3}$ are decoded on 32KB block boundaries.

	Port 4 Pin Function				Max. Memory Size per CEx
P4CNT.5-3	**P4.7**	**P4.6**	**P4.5**	**P4.4**	
000	I/O	I/O	I/O	I/O	32 kbytes
100	I/O	I/O	I/O	A16	128 kbytes
101	I/O	I/O	A17	A16	256 kbytes
110	I/O	A18	A17	A16	512 kbytes
111	A19	A18	A17	A16	1 Mbytes

P4CNT.2-P4CNT.0 **Port Pin P4.3-P4.0 Configuration Control Bits**

P4CNT.2-0 determine whether specific P4 pins function as program chip enable signals or I/O. The memory ranges for each $\overline{CEx}$ signal are determined by P4CNT.5-3. Note that when the appropriate PDCEx bit (MCON.3-0) is set, the corresponding $\overline{CEx}$ pin will function as a combined program/peripheral chip enable, and the respective $\overline{PCE0}-\overline{PCE3}$ will be disabled.

	Port 4 Pin Function			
P4CNT.2-0	**P4.3**	**P4.2**	**P4.1**	**P4.0**
000	I/O	I/O	I/O	I/O
100	I/O	I/O	I/O	$\overline{CE0}$
101	I/O	I/O	$\overline{CE1}$	$\overline{CE0}$
110	I/O	$\overline{CE2}$	$\overline{CE1}$	$\overline{CE0}$
111	$\overline{CE3}$	$\overline{CE2}$	$\overline{CE1}$	$\overline{CE0}$

Data Pointer Extended Register 0 (DPX)

	7	6	5	4	3	2	1	0
SFR 93h								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

DPL.7-0
Bits 7-0

Data Pointer Extended Register 0. This register contains the high-order byte of the 22-bit address (or 23-bit address when CMA=1) when performing operations with Data Pointer 0. This register is ignored when addressing data memory in the 16-bit addressing mode.

Data Pointer Extended Register 1 (DPX1)

	7	6	5	4	3	2	1	0
SFR 95h								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

DPL.7-0
Bits 7-0

Data Pointer Extended Register 1. This register contains the high-order byte of the 22-bit address (or 23-bit address when CMA=1) when performing operations with Data Pointer 1. This register is ignored when addressing data memory in the 16-bit addressing mode.

CAN 0 Receive Message Stored Register 0 (C0RMS0)

	7	6	5	4	3	2	1	0
SFR 96h								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 0 Receive Message Stored Register 0. This register indicates which of CAN 0 message centers 1-8 have successfully received and stored a message since the last read of this register. A logic one in a location indicates a message has been received and stored for that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C0RMS1 register to ascertain the status of all message centers.

C0RMS0.7 **Message Center 8, Message Received and Stored**
Bit 7

C0RMS0.6 **Message Center 7, Message Received and Stored**
Bit 6

C0RMS0.5 **Message Center 6, Message Received and Stored**
Bit 5

C0RMS0.4 **Message Center 5, Message Received and Stored**
Bit 4

C0RMS0.3 **Message Center 4, Message Received and Stored**
Bit 3

C0RMS0.2 **Message Center 3, Message Received and Stored**
Bit 2

C0RMS0.1 **Message Center 2, Message Received and Stored**
Bit 1

C0RMS0.0 **Message Center 1, Message Received and Stored**
Bit 0

CAN 0 Receive Message Stored Register 1 (C0RMS1)

	7	6	5	4	3	2	1	0
SFR 97h								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 0 Receive Message Stored Register 1. This register indicates which of CAN 0 message centers 9-15 have successfully received and stored a message since the last read of this register. A logic one in a location indicates a message has been received and stored for that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C0RMS0 register to ascertain the status of all message centers.

Bit 7 Reserved

C0RMS1.6 **Message Center 15, Message Received and Stored**
Bit 6

C0RMS1.5 **Message Center 14, Message Received and Stored**
Bit 5

C0RMS1.4 **Message Center 13, Message Received and Stored**
Bit 4

C0RMS1.3 **Message Center 12, Message Received and Stored**
Bit 3

C0RMS1.2 **Message Center 11, Message Received and Stored**
Bit 2

C0RMS1.1 **Message Center 10, Message Received and Stored**
Bit 1

C0RMS1.0 **Message Center 9, Message Received and Stored**
Bit 0

Serial Port 0 Control (SCON0)

	7	6	5	4	3	2	1	0
SFR 98h	SM0/FE_0	SM1_0	SM2_0	REN_0	TB8_0	RB8_0	T1_0	R1_0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SM0-2
Bits 7-5

Serial Port Mode These bits control the mode of serial port 0. In addition the SM0 and SM2_0 bits have secondary functions as shown below.

SM0	SM1	SM2	MODE	FUNCTION	LENGTH	PERIOD
0	0	0	0	Synchronous	8 bits	12 t_{CLK}
0	0	1	0	Synchronous	8 bits	4 t_{CLK}
0	1	X	1	Asynchronous	10 bits	Timer 1 or 2 baud rate equation
1	0	0	2	Asynchronous	11 bits	64 t_{CLK} (SMOD=0) 32 t_{CLK} (SMOD=1)
1	0	1	2	Asynchronous w/ Multiprocessor communication	11 bits	64 t_{CLK} (SMOD=0) 32 t_{CLK} (SMOD=1)
1	1	0	3	Asynchronous	11 bits	Timer 1 or 2 baud rate equation
1	1	1	3	Asynchronous w/ Multiprocessor communication	11 bits	Timer 1 or 2 baud rate equation

SM0/FE_0
Bit 7

Framing Error Flag. When SMOD0 (PCON.6)=0, this bit (SM0) is used to select the mode for serial port 0. When SMOD0 (PCON.6)=1, this bit (FE) will be set upon detection of an invalid stop bit. When used as FE, this bit must be cleared in software. Once the SMOD0 bit is set, modifications to this bit will not affect the serial port mode settings. Although accessed from the same register, internally the data for bits SM0 and FE are stored in different locations.

SM1_0
Bit 6

No alternate function.

SM2_0
Bit 5

Multiple CPU Communications. The function of this bit is dependent on the serial port 0 mode.

Mode 0: Selects 12 t_{CLK} or 4 t_{CLK} period for synchronous serial port 0 data transfers.

Mode 1: When set, reception is ignored (RI_0 is not set) if invalid stop bit received.

Mode 2/3: When this bit is set, multiprocessor communications are enabled in modes 2 and 3. This will prevent the RI_0 bit from being set, and an interrupt being asserted, if the 9th bit received is not 1.

REN_0
Bit 4

Receiver Enable. This bit enable/disables the serial port 0 receiver shift register.

0 = Serial port 0 reception disabled.

1= Serial port 0 receiver enabled (modes 1, 2, 3). Initiate synchronous reception (mode 0).

TB8_0 (Bit 3) — **9th Transmission Bit State.** This bit defines the state of the 9th transmission bit in serial port 0 modes 2 and 3.

RB8_0 (Bit 2) — **9th Received Bit State.** This bit identifies that state of the 9th reception bit of received data in serial port 0 modes 2 and 3. In serial port mode 1, when SM2_0=0, RB8_0 is the state of the stop bit. RB8_0 is not used in mode 0.

TI_0 (Bit 1) — **Transmitter Interrupt Flag.** This bit indicates that data in the serial port 0 buffer has been completely shifted out. In serial port mode 0, TI_0 is set at the end of the 8th data bit. In all other modes, this bit is set at the end of the last data bit. This bit must be manually cleared by software.

RI_0 (Bit 0) — **Receiver Interrupt Flag.** This bit indicates that a byte of data has been received in the serial port 0 buffer. In serial port mode 0, RI_0 is set at the end of the 8th bit. In serial port mode 1, RI_0 is set after the last sample of the incoming stop bit subject to the state of SM2_0. In modes 2 and 3, RI_0 is set after the last sample of RB8_0. This bit must be manually cleared by software.

Serial Data Buffer 0 (SBUF0)

	7	6	5	4	3	2	1	0
SFR 99h	SBUF0.7	SBUF0.6	SBUF0.5	SBUF0.4	SBUF0.3	SBUF0.2	SBUF0.1	SBUF0.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SBUF0.7-0 (Bits 7-0) — **Serial Data Buffer 0.** Data for serial port 0 is read from or written to this location. The serial transmit and receive buffers are separate registers, but both are addressed at this location.

Extended Stack Pointer Register (ESP)

	7	6	5	4	3	2	1	0
SFR 9Bh	1	1	1	1	1	1	ESP1	ESP0
	R-1	R-1	R-1	R-1	R-1	R-1	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

Bits 7-2 — Reserved

ESP.1-0 (Bits 1-0) — **Extended Stack Pointer.** This register contains the upper 2 bits of the 10-bit stack pointer. When the SA bit is set, any overflow of the SP from FFh to 00h will increment the ESP by 1, and any underflow of the SP from 00h to FFh will decrement the ESP by 1. The ESP register is ignored when SA = 0, but is still read/write accessible. Configuring the 4K block of SRAM as program and/or data memory (IDM1,IDM0=11b) will disable the extended stack mode. Internal logic will take into consideration the programming conditions imposed by the SA, IDM1 and IDM0 bits within the MCON register, to allow access to the 1K Stack Memory. See ACON register for more detail.

Address Page Register (AP)

	7	6	5	4	3	2	1	0
SFR 9Ch								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

AP.7-0
Bits 7-0

Address Page Register. The AP Register (AP) supports extended program and data addressing (>64KB) capabilities in the 22-bit paged addressing mode (AM1, AM0 = 01b), and is fully compatible with the original 8052 16-bit addressing mode. When executing LJMP or LCALL instructions in paged addressing mode, the microcontroller automatically loads bits 23:16 of the program counter with the contents of the AP register to calculate the new LCALL or LJMP address. The AP register affects only the previous instructions, and is not incremented during a program counter rollover from FFFFh to 0000h. This register is a general purpose SFR when not operating in 22-bit paged mode.

Executing interrupts while in 22-bit paged addressing mode pushes the three bytes of the program counter onto the stack, but not the AP register itself. The AP register should be saved at the beginning of the ISR if it will be modified inside the ISR. Following the execution of a RETI instruction, the processor will automatically reload the entire 24 value of the PC with the original address from the stack, again leaving the contents of the AP register unchanged.

Address Control Register (ACON)

	7	6	5	4	3	2	1	0
SFR 9Dh	1	1	1	1	1	SA	AM1	AM0
	R-1	R-1	R-1	R-1	R-1	RT-0	RT-0	RT-0

R=Unrestricted Read, T=Timed Access Write Only, -n=Value after Reset

Bits 7-3 Reserved

SA
Bit 2

Extended Stack Address Mode Enable. This bit can only be modified via the Timed Access procedure.
0 = All instructions will utilize the traditional 8-bit 8051 stack pointer (SP;81h).
1 = All instructions will utilize the 10-bit stack pointer formed by concatenating the 2 least significant bits of the ESP register with the SP register. Lower 1 KB of internal MOVX memory is used as the stack when this bit is set. This bit cannot be set while IDM1:IDM0=11b.

AM1, AM0
Bits 1-0

Address Mode Control bits. These bits establish the addressing mode for the device. These bits can only be modified via the Timed Access procedure.

AM1	AM0	Addressing Mode
0	0	16-bit Addressing Mode
0	1	22-bit Paged Addressing Mode
1	x	22-bit Contiguous Addressing Mode

CAN 0 Transmit Message Acknowledgement Register 0 (C0TMA0)

	7	6	5	4	3	2	1	0
SFR 9Eh								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 0 Transmit Message Acknowledgement Register 0. This register indicates which of CAN 0 message centers 1-8 have successfully transmitted a message since the last read of this register. A logic one in a location indicates a message has been transmitted from that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C0TMA1 register to ascertain the status of all message centers.

C0TMA0.7 **Message Center 8, Message Transmitted**
Bit 7

C0TMA0.6 **Message Center 7, Message Transmitted**
Bit 6

C0TMA0.5 **Message Center 6, Message Transmitted**
Bit 5

C0TMA0.4 **Message Center 5, Message Transmitted**
Bit 4

C0TMA0.3 **Message Center 4, Message Transmitted**
Bit 3

C0TMA0.2 **Message Center 3, Message Transmitted**
Bit 2

C0TMA0.1 **Message Center 2, Message Transmitted**
Bit 1

C0TMA0.0 **Message Center 1, Message Transmitted**
Bit 0

CAN 0 Transmit Message Acknowledgement Register 1 (C0TMA1)

	7	6	5	4	3	2	1	0
SFR 9Fh								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 0 Transmit Message Acknowledgement Register 1. This register indicates which of CAN 0 message centers 9-15 have successfully transmitted a message since the last read of this register. A logic one in a location indicates a message has been transmitted for that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C0TMA0 register to ascertain the status of all message centers.

Bit 7 Reserved

C0TMA1.6 **Message Center 15, Message Transmitted**
Bit 6

C0TMA1.5 **Message Center 14, Message Transmitted**
Bit 5

C0TMA1.4 **Message Center 13, Message Transmitted**
Bit 4

C0TMA1.3 **Message Center 12, Message Transmitted**
Bit 3

C0TMA1.2 **Message Center 11, Message Transmitted**
Bit 2

C0TMA1.1 **Message Center 10, Message Transmitted**
Bit 1

C0TMA1.0 **Message Center 9, Message Transmitted**
Bit 0

Port 2 (P2)

	7	6	5	4	3	2	1	0
SFR A0h	A15/P2.7	A14/P2.6	A13/P2.5	A12/P2.4	A11/P2.3	A10/P2.2	A9/P2.1	A8/P2.0
	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

P2.7-0
Bits 7-0

Port 2. The Port 2 pins function as an address bus during external memory accesses, and a general purpose I/O port when executing code memory from the internal 4KB SRAM (IDM1,IDM0 = 00b). When executing programs from the internal 4KB SRAM, the contents of this SFR will be driven onto the Port 2 pins. When executing programs from external memory, writes to P2 will have no effect on the state of the Port 2 pins (except during register-indirect MOVX operations).

When executing register-indirect instructions such as MOVX A, @R1, this register supplies the address MSB during data memory operations.

Port 5 (P5)

	7	6	5	4	3	2	1	0
SFR A1h	P5.7 $\overline{\text{PCE3}}$	P5.6 $\overline{\text{PCE2}}$	P5.5 $\overline{\text{PCE1}}$	P5.4 $\overline{\text{PCE0}}$	P5.3 C1TX	P5.2 C1RX	P5.1 C0RX	P5.0 C0TX
	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

General Purpose I/O Port 5. This register functions as a general purpose I/O port. In addition, all the pins have an alternate function listed below. Each of the alternate functions is controlled by and/or influences other SFRs. The associated Port 5 latch bit must contain a logic one before the pin can be used in its alternate function capability.

$\overline{\text{PCE3}}$
Bit 7
Peripheral Chip Enable 3. When enabled via the P5CNT register, this pin will assert the fourth chip enable signal.

$\overline{\text{PCE2}}$
Bit 6
Peripheral Chip Enable 2. When enabled via the P5CNT register, this pin will assert the third chip enable signal.

$\overline{\text{PCE1}}$
Bit 5
Peripheral Chip Enable 1. When enabled via the P5CNT register, this pin will assert the second chip enable signal.

$\overline{\text{PCE0}}$
Bit 4
Peripheral Chip Enable 0. When enabled via the P5CNT register, this pin will assert the first chip enable signal.

C1TX/TXD1
Bit 3
CAN 1 Transmit / Serial Port 1 Transmit. This pin is connected to the transmit data input pin of the CAN 1 transceiver device. Setting the Serial Port 1 External Connection bit (SP1EC, P5CNT.5) configures this pin as the Serial Port 1 transmit signal, disabling the corresponding CAN 1 function.

C1RX/RXD1
Bit 2
CAN 1 Receive / Serial Port 1 Receive. This pin is connected to the receive data output pin of the CAN 1 transceiver device. Setting the Serial Port 1 External Connection bit (SP1EC, P5CNT.5) configures this pin as the Serial Port 1 receive signal, disabling the corresponding CAN 1 function.

C0RX
Bit 1
CAN 0 Receive. This pin is connected to the receive data output pin of the CAN 0 transceiver device.

C0TX
Bit 0
CAN 0 Transmit. This pin is connected to the transmit data input pin of the CAN 0 transceiver device.

Port 5 Control Register (P5CNT)

	7	6	5	4	3	2	1	0
SFR A2h	CAN1BA	CAN0BA	SP1EC	C1_I/O	C0_I/O	P5CNT.2	P5CNT.1	P5CNT.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RT-0	RT-0	RT-0

R=Unrestricted Read, W=Unrestricted Write, T=Timed Access Write Only, -n=Value after Reset

CAN1BA
Bit 7

CAN 1 Bus Active. The CAN1BA signal is a latched status bit that will be set if the respective CAN1 I/O Enabled (P5CNT.4) bit is set and bus activity detected on the CAN 1 bus. Once activity is detected and the bit is set, it will remain set until cleared via application software or a reset.

CAN0BA
Bit 6

CAN 0 Bus Active. The CAN0BA signal is a latched status bit that will be set if the respective CAN0 I/O Enabled (P5CNT.3) bit is set and bus activity detected on the CAN 0 bus. Once activity is detected and the bit is set, it will remain set until cleared via application software or a reset.

SP1EC
Bit 5

Serial Port 1 External Connections. This bit controls whether the Serial Port 1 signals are asserted on P1.2/P1.3 or P5.2/P5.3. Rerouting the serial port signals to Port 5 allows the use of both serial ports (but with the loss of the CAN 1 interface) when Port 1 becomes a dedicated address bus during demultiplexed addressing mode. Note that the corresponding port pins must be set to 1 before they can be used in their serial port or CAN functions.

0 = Serial Port 1 signals are routed to P1.2/P1.3.
Conditions: $\overline{MUX}$ =0, SFR bit P1.2 =1, SFR bit P1.3 =1

1 = Serial Port 1 signals are routed to P5.2/P5.3
Conditions: $\overline{MUX}$ =0, SFR bit P5.2 =1, SFR bit P5.3 =1

C1_I/O
Bit 4

CAN 1 I/O Enable. This bit controls the function of port pins P5.2 and P5.3.

0 = Port pins P5.2 and P5.3 function as general-purpose I/O pins. The alternate Serial Port 1 transmit and receive functions on Port 5 are only possible when this bit is cleared to 0.

1 = Port pins P5.2 and P5.3 are dedicated to the CAN 1 receive and transmit functions.

C0_I/O
Bit 3

CAN 0 I/O Enable. This bit controls the function of port pins P5.0 and P5.1.

0 = Port pins P5.0 and P5.1 function as general-purpose I/O pins.

1 = Port pins P5.0 and P5.1 are dedicated to the CAN 0 receive and transmit functions.

P5CNT.2-P5CNT.0

Port Pin P5.7-P5.4 Configuration Control Bits

These bits, in conjunction with the P4CNT register, control which Port 5 pins (if any) are used for PCEx decoding as shown in the table below. The memory range addressable by each PCEx signal is a function of the total number of address lines (A19-A16) established by the P4CNT register. Note that the chip enable range when using A0-A15 is 32 KB instead of the expected 64 KB. This is to allow the use of more common 32 KB memory devices rather than 64 KB devices.

	Port 5 Pin Function			
P5CNT.2-0	**P5.7**	**P5.6**	**P5.5**	**P5.4**
000	I/O	I/O	I/O	I/O
100	I/O	I/O	I/O	$\overline{\text{PCE0}}$
101	I/O	I/O	$\overline{\text{PCE1}}$	$\overline{\text{PCE0}}$
110	I/O	$\overline{\text{PCE2}}$	$\overline{\text{PCE1}}$	$\overline{\text{PCE0}}$
111	$\overline{\text{PCE3}}$	$\overline{\text{PCE2}}$	$\overline{\text{PCE1}}$	$\overline{\text{PCE0}}$

The memory range addressable by each PCEx signal is a function of the total number of address lines (A19-A16) established by the P4CNT register. Note that the chip enable range when using A0-A15 is 32 KB instead of the expected 64 KB. This is to allow the use of more common 32 KB memory devices rather than 64 KB devices.

	Port 4 Pin Function			
P4CNT.5-3	$\overline{\text{PCE0}}$	$\overline{\text{PCE1}}$	$\overline{\text{PCE2}}$	$\overline{\text{PCE3}}$
000	0 - 32KB	32 - 64KB	64 - 96KB	96 - 128KB
100	0 - 128KB	128 - 256KB	256 - 384KB	384 - 512KB
101	0 - 256KB	256 - 512KB	512 - 768KB	768KB - 1MB
110	0 - 512KB	512 - 1MB	1 - 1.5MB	1.5 - 2MB
111	0 - 1MB	1 - 2M	2 - 3MB	3 - 4MB

CAN 0 Control Register (C0C)

	7	6	5	4	3	2	1	0
SFR A3h	ERIE	STIE	PDE	SIESTA	CRST	AUTOB	ERCS	SWINT
	RW-0	RW-0	RW-0	RW-0	RT-1	RW-0	RW-0	RW-1

R=Unrestricted Read, W=Unrestricted Write, T=Timed Access Write Only, -n=Value after Reset

ERIE
Bit 7

CAN 0 Error Interrupt Enable.

0 = CAN 0 Error Interrupt is disabled.

1 = Setting this bit while the C0IE bit (EIE.6) and Global Interrupt Enable bits (IE.7) are set will generate an interrupt if the CAN 0 Bus Off (BUSOFF) or CAN 0 Error Count Exceeded bit (CECE) bits are set.

STIE
Bit 6

CAN 0 Status Interrupt Enable.

0 = CAN 0 Status Interrupt is disabled.

1 = If the C0IE bit (EIE.6) is set, an interrupt will be generated if the CAN 0 Transmit Status bit (TXS), Receive Status bit (RXS) or the Wake-Up Status bit (WKS) is set. An interrupt will also be generated if the Status Error bits (ER2-0) changes to a non-000b or non-111b state.

PDE
Bit 5

CAN 0 Power Down Enable. Setting this bit places the CAN 0 module into its lowest power mode. The module will enter Power Down mode immediately upon setting this bit, or following the completion of the current reception, transmission, arbitration failure, or error condition on CAN 0. Software can poll the PDE bit to ascertain whether the microcontroller has entered Power Down mode (PDE=1) or is waiting for a current CAN operation to complete (PDE=0) before entering Power Down Mode.

Power Down mode is exited by clearing the PDE bit or by any reset of the microcontroller. The CAN 0 module will resume operation after the receipt of 11 consecutive recessive bits.

The Wake-Up Status bit, WKS, is a logical OR of this bit and the SIESTA bit.

SIESTA
Bit 4

CAN 0 Siesta Mode Enable. Setting this bit places the CAN 0 module into a low power mode. The module will enter Siesta mode immediately upon setting this bit, or following the completion of the current reception, transmission, arbitration failure, or error condition on CAN 0. Software can poll the SIESTA bit to ascertain whether the microcontroller has entered Siesta mode (SIESTA =1) or is waiting for a current CAN operation to complete (SIESTA =0) before entering Siesta Mode.

Siesta mode is exited by clearing the Siesta bit, detecting CAN 0 bus activity, or setting either the CRST or SWINT bits to 1. The CAN 0 module will begin operation after the receipt of 11 consecutive recessive bits.

The Wake-Up Status bit, WKS, is a logical OR of this bit and the PDE bit.

CRST
Bit 3

CAN 0 Reset. Setting this bit via a Timed Access write will reset all CAN 0 registers in the SFR map to their reset default states. The module will reset the registers immediately upon setting this bit, or following the completion of the current reception, transmission, arbitration failure, or error condition on CAN 0. Software can poll the CRST bit to ascertain whether the microcontroller has successfully reset the registers (CRST =1) or is waiting for a current CAN operation to complete (CRST =0) before resetting the registers. Setting the CRST bit also clears the transmit and receive error counters and sets the SWINT bit.

CRST must be cleared by software to remove the CAN reset. The state of the SWINT and BUSOFF bits determines the action of the device when the CRST bit is cleared.

AUTOB
Bit 2

CAN 0 Autobaud. Setting this bit allows the CAN 0 module to establish proper CAN bus timing without disrupting the normal data flow between other nodes on the CAN Bus. When in the autobaud mode, incoming data on the C0RX pin is internally ANDed with transmit data generated by the CAN 0 module. An internal loop back feeds this combined data stream back into the input of the CAN 0 module. At the same time, C0TX pin is placed into a recessive state to prevent driving non-synchronized data (creating CAN Bus errors to other nodes) while attempting to synchronize the processor with the CAN Bus.

With AUTOB = 1, the microcontroller auto-baud algorithm will make use of the CAN 0 Status Register RXS and error status bits to determine when a message is successfully received (when AUTOB =1, a successful receive does not require a store). Each successive baud rate attempt is proceeded by the microcontroller clearing the transmit and receive error counters via a write of 00h to the Transmit Error SFR Register and a read of the CAN 0 Status Register to clear the previous Status Change Interrupt. Note that a write to the Transmit Error SFR Register automatically resets the CAN fault confinement state machine to an initial (error active) state if the error counters are cleared to 00h. If, however, the error counters are programmed to a value greater than 128, the CAN module will be in a error passive state. Appropriate flags are set when the error counter is written with any value. A write of the Status Register is also used to remove the previous error value in the ER2-0 bits. Clearing the error counters will also clear the CECE bit, if set.

When BUSOFF = 1, software is prohibited from writing to the error counters by virtue of the fact that the SWINT bit is also forced to a 0 state during the period that the CAN module performs a bus recovery and power up sequence. Once the CAN module has removed itself from the Bus Off condition it will also clear BUSOFF = 0, set SWINT = 1, and will clear both the transmit and receive error counters to 00h.

ERCS
Bit 1

CAN 0 Error Count Select. This bit selects the number of transmit or receive errors that will cause the CAN 0 Error Count Exceeded bit, CECE (C0S.6), to be set.

0 = CECE bit set when the transmit or receive error counters exceed 95 errors.
1 = CECE bit set when the transmit or receive error counters exceed 127 errors.

SWINT
Bit 0

CAN 0 Software Initialization Enable. This bit enables (SWINT=1) and disables (SWINT=0) software write access to the first 16 bytes of the CAN 0 MOVX SRAM. These bytes contain the CAN 0 Control/Status/Mask Registers. Read access to all bytes in the CAN 0 MOVX SRAM is permitted at all times, regardless of the state of the SWINT bit.

Setting SWINT=1 disables CAN 0 Bus activity, allowing software access to the CAN 0 Control/Status/Mask Registers without corrupting CAN Bus transmission or reception. A special lockout procedure delays the internal assertion of the SWINT bit until all CAN 0 activity has ceased. The following procedure must be followed when setting the SWINT bit to prevent the accidental corruption of CAN Bus activity:

1. Write a 1 to the SWINT bit, starting the internal process to enter the software initialization process.
2. Poll the SWINT bit until it is set. The lockout circuit will hold SWINT=0 if it detects a reception, transmission, or arbitration in progress. When one of these conditions ceases, or if an error occurs, the CAN module will set SWINT=1, indicating that the CAN module is disabled and software can now write to the first 16 bytes of the CAN 0 MOVX SRAM. Attempts to modify the first 16 bytes of the CAN 0 MOVX SRAM while SWINT=0 will fail, leaving the bytes unchanged.

The SWINT bit controls access to several other bits and registers. The CAN 0 Transmit Error Register (C0TE;A6h) and CAN 0 Receive Error Register (C0RE;A7h) are only modifiable while SWINT=1. Setting SWINT=1 automatically clears the SIESTA bit, and attempts to set SWINT=1 and SIESTA=1 in the same write to the COC register will result in SWINT=1 and SIESTA=0.

The BUSOFF bit has a direct interaction with the SWINT bit. When a Bus Off condition is detected (BUSOFF=1), the CAN module will automatically clear SWINT=0 and initiate a bus recovery and power-up sequence. Write access to the SWINT bit is prohibited until the Bus Off condition has been cleared and BUSOFF has been reset to 0.

The SWINT bit is also set automatically following a system reset, the setting of the CRST bit in the CAN 0 Control Register, or programming the CAN Bus Timing Registers (C0BT0, C0BT1 in the MOVX SRAM) to 00h (an invalid state). As a precaution against utilizing the CAN with invalid bus timing, the SWINT bit cannot be cleared while C0BT0=C0BT1=00h. When this bit is cleared, the CAN 0 module will initiate a CAN Bus synchronization after the CAN module executes a power-up sequence (reception of 11 consecutive recessive bits.)

CAN 0 Status Register (C0S)

	7	6	5	4	3	2	1	0
SFR A4h	BUSOFF	CECE	WKS	RXS	TXS	ER2	ER1	ER0
	R-0	R-0	R-0	RW-0	RW-0	R-0	R-0	R-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

BUSOFF
Bit 7

CAN 0 Bus Off. When BUSOFF = 1, the CAN 0 Bus is disabled and is not capable of receiving or transmitting messages. This condition is the result of the transmit error counter reaching a count of 256. When the CAN 0 module detects an error count of 256 the CAN module will automatically set BUSOFF = 1 and clear SWINT = 0.

BUSOFF is cleared to a 0 to enable CAN 0 Bus activity when the CAN module completes both the busoff recovery (128 X 11 consecutive recessive bits) and the power-up sequence (11 consecutive recessive bits). Once the CAN module has completed this relationship it will set SWINT = 1 and will enter into the software initialization state. Once software has cleared SWINT to a 0, the CAN module will be enabled to transmit and receive messages. When BUSOFF = 0, the CAN 0 Bus is enabled to receive or transmit messages. A change in the state of BUSOFF from a previous 0 to a 1 will generate an interrupt if the ERIE, C0IE and EA register bits are set. All microcontroller writes to the SWINT bit are disabled when BUSOFF = 1. Both the transmit and receive error counters are cleared to 00h when the Bus Off condition is cleared by the CAN module (BUSOFF=0).

CECE
Bit 6

CAN 0 Error Count Exceeded. This bit operates in one of two modes, depending on the state of the ERCS bit in the CAN 0 Control Register.

ERCS = 0 (Error count limit=96) In this mode when CECE=1, the interrupt flag indicates that either the CAN 0 Transmit Error Counter or the CAN 0 Receive Error Counter has reached an error count of 96, which represents an exceptionally high number of errors. CECE=0 indicates that both error counters have an error count of less than 96. A 0 to 1 transition of CECE will generate an interrupt if the ERIE, C0IE and IE SFR bits are set.

ERCS = 1 (Error count limit=128) In this mode when CECE=1, the interrupt flag indicates that either the CAN 0 Transmit Error Counter or the CAN 0 Receive Error Counter has reached an error count of 128, which represents an exceptionally high number of errors. CECE = 0 indicates that the current Transmit Error Counter and Receive Error Counter both have an error count of less than 128. A change in the state of CECE from either a previous 0 to a 1 *or from a previous 1 to 0* will generate an interrupt if the ERIE, C0IE and IE SFR bits are set.

WKS
Bit 5

CAN 0 Wake-up Status. When WKS=1, the CAN 0 module is in either SIESTA or Power Down mode. Clearing both the SIESTA and PDE bits will force the WKS=0. A change in the state of WKS from a previous 1 to 0 will generate an interrupt if the STIE, C0IE and IE SFR bits are set.

RXS
Bit 4

CAN 0 Receive Status. This bit indicates whether or not messages have been received since the last read of the CAN 0 Status Register. RXS is only set by the CAN 0 logic and must be cleared by the Microcontroller software, the CRST bit, or a system Reset.

1 = The meaning of RXS=1 is dependent on the Autobaud bit, AUTOB.

AUTOB=0, RXS = 1 indicates that a message has been both successfully received and stored in one of the message centers by CAN 0 since the last read of the CAN 0 Status Register.

AUTOB=1, RXS = 1 indicates that a message has been successfully received by CAN 0 since the last read of the CAN 0 Status Register. Note that messages that are successfully received without errors but do not pass the arbitration filtering will still set the RXS bit.

0 = No messages have been successfully received since the last read of the CAN 0 Status Register.

When STIE= 1 and the RXS bit transitions from 0 to 1, the CAN Interrupt Register (C0IR;A5h) will change to 01h to indicate a pending interrupt due to a change in the CAN Status Register(C0S;A4h). Reading any bit in the C0S register will clear the pending interrupt, causing the C0IR register to change to 00h if no interrupts are pending or the appropriate value if a lower priority message center interrupt is pending. If a second successful reception is detected prior to or after the clearing of the RXS bit in the Status Register, a second status change interrupt flag will be set, issuing a second interrupt. Each new successful reception will generate an interrupt request independent of the previous state of the RXS bit, as long as the CAN Status Register has been read to clear the previous status change interrupt flag. Note that if software changes RXS from 0 to 1, an artificial Status Change Interrupt (STIE=1) will be generated. Thus, if RXS was previously set to 0 and a reception was successful, RXS will be set to 1 and an enabled interrupt may be asserted. An interrupt may be asserted (if enabled) if software changes RXS from 0 to 1. If RXS was previously set to 1 and a reception was successful, RXS remains set and an interrupt may be asserted if enabled. No interrupt will be asserted if software attempts to set RXS=1 while the bit is already set.

TXS
Bit 3

CAN 0 Transmit Status. This bit indicates whether or not one or more messages have been successfully transmitted since the last read of the CAN 0 Status Register. TXS is only set by the CAN 0 logic and is not cleared by the CAN controller but is only cleared via software, the CRST bit, or a system Reset.

1 = A message has been successfully transmitted by CAN 0 (error free and acknowledged) since the last read of the CAN 0 Status Register.

0 = No messages have been successfully transmitted since the last read of the CAN 0 Status Register.

When STIE= 1 and the TXS bit transitions from 0 to 1, the CAN Interrupt Register (C0IR;A5h) will change to 01h to indicate a pending interrupt due to a change in the CAN Status Register. Reading any bit in the C0S register will clear the pending interrupt, causing C0IR to change to 00h if no interrupts are pending or the appropriate value if a lower priority message center interrupt is pending. If a second successful reception is detected prior to or after the clearing of the RXS bit in the Status Register, a second status change interrupt flag will be set, issuing a second interrupt. Each new successful reception will generate an interrupt request independent of the previous state of the RXS bit, as long as the CAN Status Register has been read to clear the previous status change interrupt flag. Note that if software changes TXS from 0 to 1, an artificial Status Change Interrupt (STIE=1) will be generated. Thus, if TXS was previously set to 0 and a reception was successful, TXS will be set to 1 and an enabled interrupt may be asserted. An interrupt may be asserted (if enabled) if software changes TXS from 0 to 1. If TXS was previously set to 1 and a reception was successful, TXS remains set and an interrupt may be asserted if enabled. No interrupt will be asserted if software attempts to set TXS while it is already set.

ER2-0
Bit 2-0

CAN 0 Bus Error Status. These bits indicate the type of error, if any, detected in the last CAN 0 Bus Frame. These bits will be reset to the 111b state following any read of the C0S register (when SWINT=0), allowing software to determine if a new error has been received since the last read of this register. The ER2-0 bits are read only.

If enabled, an interrupt will be generated any time the ER2-0 bits change from 000b or 111b to another value. Errors received while the ER2-0 bits are in a non-000b or 111b state will be ignored, leaving ER2-0 unchanged and no additional interrupts will be generated. This ensures that error conditions will not be lost/overwritten before software has a chance to read the C0S register. Once the C0S register is read and the ER2-0 bits return to 111b, new errors will be processed normally. In the case of simultaneous errors in multiple CAN 0 message centers, only the highest priority error is indicated.

ER2	ER1	ER0	Priority	Error Conditions
0	0	0	N/A	No Error in Last Frame
0	0	1	2	Bit Stuff Error
0	1	0	5	Format Error
0	1	1	4	Transmit Not Acknowledged Error
1	0	0	6(lowest)	Bit 1 Error
1	0	1	1(highest)	Bit 0 Error
1	1	0	3	CRC Error
1	1	1	N/A	No change since last C0S read

The following is a description of the different error types:

Bit Stuff Error: Occurs when the CAN controller detects more than 5 consecutive bits of an identical state are received in an incoming message.

Format Error: Generated when a received message has the wrong format.

Transmit Not Acknowledged Error: Indicates that a data frame was sent and the requested node did not acknowledged the message.

Bit 1 Error: Indicates that the CAN attempted to transmit a message and that when a recessive bit was transmitted, the CAN bus was found to have a dominant bit level. This error is not generated when the bit is a part of the arbitration field (identifier and remote retransmission request).

Bit 0 Error: Indicates that the CAN attempted to transmit a message and that when a dominant bit was transmitted, the CAN bus was found to have a recessive bit level. This error is not generated when the bit is a part of the arbitration field. The Bit 0 Error is set each time a recessive bit is received during the Busoff recovery period.

CRC Error: Generated whenever the calculated CRC of a received message does not match the CRC embedded in the message.

CAN 0 Interrupt Register (C0IR)

	7	6	5	4	3	2	1	0
SFR A5h								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

C0IR.7-0
Bit 7-5

CAN 0 Interrupt Indicator 7-0 This register indicates the status of the interrupt source associated with the CAN 0 module. Reading this register after the generation of a CAN 0 Interrupt will identify the interrupt source as shown in the table below. This register is cleared to 00h following a reset.

C0IR.7-0	Priority	Interrupt Source
00h	N/A	No Pending Interrupt
01h	1 (highest)	Change in the CAN 0 Status Register
02h	2	Message 15
03h	3	Message 1
04h	4	Message 2
05h	5	Message 3
06h	6	Message 4
07h	7	Message 5
08h	8	Message 6
09h	9	Message 7
0Ah	10	Message 8
0Bh	11	Message 9
0Ch	12	Message 10
0Dh	13	Message 11
0Eh	14	Message 12
0Fh	15	Message 13
10h	16 (lowest)	Message 14

The C0IR value will not change unless the previous interrupt source has been acknowledged and removed (i.e., software read of the C0S register or clearing of the appropriate INTRQ bit), even if the new interrupt has a higher priority. If two enabled interrupt sources become active simultaneously, the interrupt of higher priority will be reflected in the C0IR value.

The CAN 0 interrupt source into the interrupt logic is active whenever C0IR is not equal to 00h. Changes in the C0IR value from 00h to a non-zero state, indicate the first interrupt source detected by the CAN module following the non-active interrupt state. The C0IR interrupt values will remain in place until the interrupt source is removed, independent of other higher (or lower) priority interrupts that become active prior to clearing the currently displayed interrupt source.

When the current CAN interrupt source is cleared, C0IR will change to reflect the next active interrupt with the highest priority. The Status Change interrupt will be asserted if there has been a change in the Can 0 Status Register (if enabled by the appropriate ERIE and/or STIE bit) and the CAN Status Interrupt state is set. A message center interrupt will be indicated if the INTRQ bit in the respective CAN Message Control Register is set.

CAN 0 Transmit Error Register (C0TE)

	7	6	5	4	3	2	1	0
SFR A6h								
	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0

R=Unrestricted Read, *= Write only when SWINT=1 and BUSOFF=0, -n=Value after Reset

C0TE.7-0
Bits 7-0

CAN 0 Transmit Error Register. This register indicates the number of accumulated CAN 0 transmit errors. The CAN 0 module responds in different ways to varying number of errors as shown below.

This register can only be modified via software when SWINT=1 and BUSOFF=0. All software writes to this register simultaneously load the same value into the CAN 0 Transmit Error Register and the CAN 0 Receive Error Register. Writing 00h to this register will also clear the CAN 0 Error Count Exceeded bit, CECE (C0S.6). This register is cleared following all hardware Resets and software resets enabled via the CRST bit in the CAN 0 Control Register.

C0TE Value	CAN 0 State
Value < 96	Error active mode, CAN 0 Bus on (BUSOFF=0)
128 > Value ≥ 96	Error active mode, CAN 0 Bus on (BUSOFF=0), warning level
255 ≥ Value ≥ 128	Error passive mode, CAN 0 Bus on (BUSOFF=0)
Value > 255	CAN 0 Bus off (BUSOFF=1)

CAN 0 Receive Error Register (C0RE)

	7	6	5	4	3	2	1	0
SFR A7h								
	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0

R=Unrestricted Read, *= Write only via C0TE register, -n=Value after Reset

C0RE.7-0
Bits 7-0

CAN 0 Receive Error Register. This register indicates the number of accumulated CAN 0 receive errors. All writes to the C0TE register are simultaneously loaded into this register. This register is cleared following all hardware Resets and software resets enabled via the CRST bit in the CAN 0 Control Register.

Interrupt Enable (IE)

	7	6	5	4	3	2	1	0
SFR A8h	EA	ES1	ET2	ES0	ET1	EX1	ET0	EX0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

EA Bit 7
Global Interrupt Enable. This bit controls the global masking of all interrupts except Power-Fail Interrupt, which is enabled by the EPFI bit (WDCON.5).
0 = Disable all interrupt sources. This bit overrides individual interrupt mask settings.
1 = Enable all individual interrupt masks. Individual interrupts will occur if enabled.

ES1 Bit 6
Enable Serial Port 1 Interrupt. This bit controls the masking of the serial port 1 interrupt.
0 = Disable all serial port 1 interrupts.
1 = Enable interrupt requests generated by the RI_1 (SCON1.0) or TI_1 (SCON1.1) flags.

ET2 Bit 5
Enable Timer 2 Interrupt. This bit controls the masking of the Timer 2 interrupt.
0 = Disable all Timer 2 interrupts.
1 = Enable interrupt requests generated by the TF2 flag (T2CON.7).

ES0 Bit 4
Enable Serial Port 0 Interrupt. This bit controls the masking of the serial port 0 interrupt.
0 = Disable all serial port 0 interrupts.
1 = Enable interrupt requests generated by the RI_0 (SCON0.0) or TI_0 (SCON0.1) flags.

ET1 Bit 3
Enable Timer 1 Interrupt. This bit controls the masking of the Timer 1 interrupt.
0 = Disable all Timer 1 interrupts.
1 = Enable all interrupt requests generated by the TF1 flag (TCON.7).

EX1 Bit 2
Enable External Interrupt 1. This bit controls the masking of external interrupt 1.
0 = Disable external interrupt 1.
1 = Enable all interrupt requests generated by the $\overline{\text{INT1}}$ pin.

ET0 Bit 1
Enable Timer 0 Interrupt. This bit controls the masking of the Timer 0 interrupt.
0 = Disable all Timer 0 interrupts.
1 = Enable all interrupt requests generated by the TF0 flag (TCON.5).

EX0 Bit 0
Enable External Interrupt 0. This bit controls the masking of external interrupt 0.
0 = Disable external interrupt 0.
1 = Enable all interrupt requests generated by the $\overline{\text{INT0}}$ pin.

Slave Address Register 0 (SADDR0)

	7	6	5	4	3	2	1	0
SFR A9h	SADDR0.7	SADDR0.6	SADDR0.5	SADDR0.4	SADDR0.3	SADDR0.2	SADDR0.1	SADDR0.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SADDR0.7-0
Bits 7-0

Slave Address Register 0. This register is programmed with the given or broadcast address assigned to serial port 0.

Slave Address Register 1 (SADDR1)

	7	6	5	4	3	2	1	0
SFR AAh	SADDR1.7	SADDR1.6	SADDR1.5	SADDR1.4	SADDR1.3	SADDR1.2	SADDR1.1	SADDR1.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SADDR1.7-0
Bits 7-0

Slave Address Register 1. This register is programmed with the given or broadcast address assigned to serial port 1.

CAN 0 Message Center 1 Control Register (C0M1C)

	7	6	5	4	3	2	1	0
SFR ABh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

MSRDY
Bit 7

CAN 0 Message Center 1 Ready. This bit is used by the Microcontroller to prevent the CAN module from accessing message center 1 while the microcontroller is updating message attributes. These include as identifiers (arbitration registers 0-3), data byte registers 0-7, data byte count (DTBYC3-DTBYC0), direction control (T/R), the extended or standard mode bit (EX/ST), and the mask enables (MEME and MDME) associated with this message center. When this bit is 0, the CAN 0 processor will ignore this message center for transmit, receive, or remote frame request operations.

MSRDY is cleared following a microcontroller hardware reset or a reset generated by the CRST bit in the CAN 0 Control Register, and must also remain in a cleared mode until all the CAN 0 initialization has been completed. Individual message MSRDY controls can be changed after initialization to reconfigure specific messages, without interrupting the communication of other messages on the CAN 0 Bus.

ETI
Bit 6

CAN 0 Message Center 1 Enable Transmit Interrupt. Setting ETI to a 1 will enable a successful CAN 0 transmission in message center 1 to set the INTRQ bit for this message center which in turn will issue an interrupt to the microcontroller. When ETI is cleared to 0 a successful transmission will not set INTRQ bit and will not generate an interrupt. Note that the ETI bit located in Message Center 15 is ignored by the CAN module, since the message center 15 is a receive only message center.

ERI
Bit 5

CAN 0 Message Center 1 Enable Receive Interrupt. Setting ERI to a 1 will enable a successful CAN 0 reception and storage in message center 1 to set the INTRQ bit for this message center which in turn will issue an interrupt to the microcontroller. When ERI is cleared to 0 a successful reception will not set the INTRQ bit and as such will not generate an interrupt.

INTRQ
Bit 4

CAN 0 Message Center 1 Interrupt Request. This bit serves as a CAN interrupt flag, indicating the successful transmission or reception of a message in this message center. INTRQ is automatically set when ERI=1 and message center 1 successfully receives and stores a message. The INTRQ bit is also set to a 1 when ETI is set and the CAN 1 logic completes a successful transmission. The INTRQ interrupt request must be also enabled via the EA global mask in the IE SFR register if the interrupt is to be acknowledged by the microcontroller interrupt logic. This flag must be cleared via software.

EXTRQ
Bit 3

CAN 0 Message Center 1 External Transmit Request. When EXTRQ is cleared to a 0, there are no pending requests by external CAN nodes for this message. When EXTRQ is set to a 1, a request has been made for this message by an external CAN node, but the CAN 0 controller has not yet completed the service request. Following the completion of a requested transmission by a message center programmed for transmission ($T/\overline{R} = 1$), the EXTRQ bit will be cleared by the CAN 0 controller. A remote request is only answered by a message center programmed for transmission ($T/\overline{R} = 1$) when DTUP = 1 and TIH = 0, i.e. when new data was loaded and is not being currently modified by the micro. Note that a message center programmed for a receive mode ($T/\overline{R} = 0$) will also detect a remote frame request and will set the EXTRQ bit in a similar manner, but will not automatically transmit a data frame and as such will not automatically clear the EXTRQ bit.

MTRQ
Bit 2

CAN 0 Message Center 1 Microcontroller Transmit Request. When set, this bit indicates that the message center is requesting that a message be transmitted. The bit is cleared when the transmission is complete, allowing this bit to be used to both initiate and monitor the progress of the transmission. The bit can be set via software or the CAN module, depending on the state of the Transmit/Receive bit in the CAN 0 Message 1 Format Register (located in MOVX space). This bit is cleared when the CRST bit is set, the CAN module experiences a system reset, or the conditions described below. Note that the MTRQ bit located in Message Center 15 is ignored by the CAN module, since the Message Center 15 is a receive only message center.

$T/\overline{R}$ =0 (receive)

When software sets this bit, a remote frame request previously loaded into the message center will be transmitted. The CAN 0 Module will clear this bit following the successful transmission of the frame request message.

T/$\overline{R}$ =1 (transmit)
When software sets this bit, a data frame previously loaded into the message center will be transmitted. When T/$\overline{R}$ = 1, the MTRQ bit will also be set by the CAN 0 controller at the same time that the EXTRQ bit is set by a message request from an external node.

ROW/TIH
Bit 1

CAN 0 Message Center 1 Receive Overwrite/Transmit Inhibit. The Receive Overwrite (ROW) and Transmit Inhibit (TIH) bits share the same bit location. When T/$\overline{R}$ = 0 the bit has the ROW function, serving as a flag that an overwrite of incoming data may have occurred. When T/$\overline{R}$ = 1 the bit has the Transmit Inhibit function, allowing software to disable the transmission of a message while the data contents are being updated.

Receive Overwrite: (T/R = 0, ROW is Read Only)
The CAN 0 controller automatically sets this bit 0 if a new message is received and stored while the DTUP bit was still set. When set, ROW indicates that the previous message was potentially lost and may not have been read, since the microcontroller had not cleared the DTUP bit prior to the new load. When ROW = 0, no new message has been received and stored while DTUP was set to '1' since this bit was last cleared. Note that the ROW bit will not be set when the WTOE bit is cleared to a 0, since all overwrites are disabled. This is due to the fact that even if the incoming message matches the respective message center that as long as DTUP = 1 in the respective message center, the combination of WTOE = 0 and DTUP = 1 will force the CAN module to ignore the respective message center when the CAN is processing the incoming data.

ROW is cleared by the CAN module when software clears the DTUP bit associated with that message center. INTRQ is automatically set when the ERI=1 and message center 1 successfully receives and stores a message.

ROW will reflect the actual message center relationships for message centers 1 to 14. Message center 15 utilizes a special shadow message buffer, and the ROW bit for that message center indicates an overwrite of the buffer as opposed to the actual message center 15. The ROW bit for message center 15 is cleared once the shadow buffer is loaded into the message center 15, and the shadow buffer is cleared to allow a new message to be loaded. The shadow buffer is automatically loaded into message center 15 when the microcontroller clears the DTUP and EXTRQ bits in message center 15.

Transmit Inhibit: (T/R = 1, TIH is unrestricted Read/Write)

The TIH allows the microcontroller to disable the transmission of the message when the data contents of the message are being updated. TIH = 1 directs the CAN 0 controller not to transmit the associated message. TIH = 0 enables the CAN 0 controller to transmit the message. If TIH = 1 when a remote frame request is received by the message center, EXTRQ will be set to a 1. Following the Remote Frame Request and after the microcontroller has established the proper data to be sent, the microcontroller will clear the

TIH bit to a 0, which will allow the CAN module to send the data requested by the previous Remote Frame Request. Note that the TIH bit associated with Message Center 15 is ignored because it is a receive only message center.

DTUP
Bit 0

CAN 0 Message Center 1 Data Updated. This bit indicates that new data has been loaded into the data portion of the message center. The exact function of the DTUP bit is dependent on whether the message center is configured in a receive ($T/\overline{R}$ = 0) or transmit ($T/\overline{R}$ = 1) mode. Some functions are also dependent on the state of the WTOE bit. The DTUP bit is only cleared by a software write to the bit, a system reset, or the setting of the CRST bit.

$T/\overline{R}$ =0 (receive)

In this mode ($T/\overline{R}$ = 0) the DTUP bit is set when new data has been successfully received and is ready to be read by the microcontroller. The exact meaning of the DTUP bit during a message center read is determined by the WTOE bit in the CAN 0 Control Register.

If WTOE = 1 (message center overwrite enabled), DTUP should be polled before and after reading the message center to ascertain if an overwrite of the data occurred during the read. For example, software should clear DTUP before reading the message center and then again after the message center read. If DTUP has been set, then a new message was received and software should read the message center again to read the new data. If DTUP remained cleared, no additional data was received and the data is complete.

If WTOE=0 the processor is not permitted to overwrite this message center, so it is only necessary to clear the DTUP bit after reading the message center.

The state of the DTUP bit in the receive mode does not inhibit remote frame request transmission in the receive mode. The only gating item for remote frame transmission in the receive mode is that the MSRDY and MTRQ bits must both be set.

$T/\overline{R}$ =1 (transmit)

In this mode, software must set TIH =1 and clear DTUP = 0 prior to doing an update of the associated message center. This prevents the CAN module from transmitting the data while the microcontroller is updating it. Once the microcontroller has finished configuring the message center, software must clear TIH = 0 and set MSRDY=MTRQ =DTUP =1, to enable the CAN module to transmit the data.

The CAN module will **not** clear the DTUP after the transmission, but the microcontroller can verify that the transmission has been completed, by checking the MTRQ bit, which will be cleared (MTRQ = 0) after the transmission has been successfully completed.

CAN 0 Message Center 2 Control Register (C0M2C)

	7	6	5	4	3	2	1	0
SFR ACh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M2C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 3 Control Register (C0M3C)

	7	6	5	4	3	2	1	0
SFR ADh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M3C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 4 Control Register (C0M4C)

	7	6	5	4	3	2	1	0
SFR AEh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M4C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 5 Control Register (C0M5C)

	7	6	5	4	3	2	1	0
SFR AFh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M5C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

Port 3 (P3)

	7	6	5	4	3	2	1	0
SFR B0h	P3.7 $\overline{RD}$	P3.6 $\overline{WR}$	P3.5 T1	P3.4 T0	P3.3 $\overline{INT1}$	P3.2 $\overline{INT0}$	P3.1 TXD0	P3.0 RXD0
	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

P3.7-0 Bits 7-0 — **Purpose I/O Port 3.** This register functions as a general purpose I/O port. In addition, all the pins have an alternative function listed below. Each of the functions is controlled by several other SFRs. The associated Port 1 latch bit must contain a logic one before the pin can be used in its alternate function capacity.

$\overline{RD}$ Bit 7 — **External Data Memory Read Strobe.** This pin provides an active low read strobe to an external memory device.

$\overline{WR}$ Bit 6 — **External Data Memory Write Strobe.** This pin provides an active low write strobe to an external memory device.

T1 Bit 5 — **Timer/Counter External Input.** A 1 to 0 transition on this pin will increment Timer 1.

T0 Bit 4 — **Counter External Input.** A 1 to 0 transition on this pin will increment Timer 0.

$\overline{INT1}$ Bit 3 — **External Interrupt 1.** A falling edge/low level on this pin will cause an external interrupt 1 if enabled.

$\overline{INT0}$ Bit 2 — **External Interrupt 0.** A falling edge/low level on this pin will cause an external interrupt 0 if enabled.

TXD0 Bit 1 — **Serial Port 0 Transmit.** This pin transmits the serial port 0 data in serial port modes 1, 2, 3 and emits the synchronizing clock in serial port mode 0.

RXD0 Bit 0 — **Serial Port 0 Receive.** This pin receives the serial port 0 data in serial port modes 1, 2, 3 and is a bi-directional data transfer pin in serial port mode 0.

CAN 0 Message Center 6 Control Register (C0M6C)

	7	6	5	4	3	2	1	0
SFR B3h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M6C Bits 7-0 — Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 7 Control Register (C0M7C)

	7	6	5	4	3	2	1	0
SFR B4h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M7C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 8 Control Register (C0M8C)

	7	6	5	4	3	2	1	0
SFR B5h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M8C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 9 Control Register (C0M9C)

	7	6	5	4	3	2	1	0
SFR B6h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M9C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 10 Control Register (C0M10C)

	7	6	5	4	3	2	1	0
SFR B7h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M10C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

Interrupt Priority (IP)

	7	6	5	4	3	2	1	0
SFR B8h	-	PS1	PT2	PS0	PT1	PX1	PT0	PX0
	-	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

Bit 7 Reserved. Read data is indeterminate.

PS1 Bit 6 **Serial Port 1 Interrupt.** This bit controls the priority of the serial port 1 interrupt.
0 = Serial port 1 priority is determined by the natural priority order.
1 = Serial port 1 is a high priority interrupt.

PT2 Bit 5 **Timer 2 Interrupt.** This bit controls the priority of Timer 2 interrupt.
0 = Timer 2 is determined by the natural priority order.
1 = Timer 2 is a high priority interrupt.

PS0 Bit 4 **Serial Port 0 Interrupt.** This bit controls the priority of the serial port 0 interrupt.
0 = Serial port 0 priority is determined by the natural priority order.
1 = Serial port 0 is a high priority interrupt.

PT1 Bit 3 **Timer 1 Interrupt.** This bit controls the priority of Timer 1 interrupt.
0 = Timer 1 is determined by the natural priority order.
1 = Timer 1 is a high priority interrupt.

PX1 Bit 2 **External Interrupt 1.** This bit controls the priority of external interrupt 1.
0 = External interrupt 1 is determined by the natural priority order.
1 = External interrupt 1 is a high priority interrupt.

PT0 Bit 1 **Timer 0 Interrupt.** This bit controls the priority of Timer 0 interrupt.
0 = Timer 0 is determined by the natural priority order.
1 = Timer 0 is a high priority interrupt.

PX0 Bit 0 **External Interrupt 0.** This bit controls the priority of external interrupt 0.
0 = External interrupt 0 is determined by the natural priority order.
1 = External interrupt 0 is a high priority interrupt.

Slave Address Mask Enable Register 0 (SADEN0)

	7	6	5	4	3	2	1	0
SFR B9h	SADEN0.7	SADEN0.6	SADEN0.5	SADEN0.4	SADEN0.3	SADEN0.2	SADEN0.1	SADEN0.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SADEN0.7-0
Bits 7-0

Slave Address Mask Enable Register 0. This register functions as a mask when comparing serial port 0 addresses for automatic address recognition. When a bit in this register is set, the corresponding bit location in the SADDR0 register will be exactly compared with the incoming serial port 0 data to determine if a receiver interrupt should be generated. When a bit in this register is cleared, the corresponding bit in the SADDR0 register becomes a don't care and is not compared against the incoming data. All incoming data will generate a receiver interrupt when this register is cleared.

Slave Address Mask Enable Register 1 (SADEN1)

	7	6	5	4	3	2	1	0
SFR BAh	SADEN1.7	SADEN1.6	SADEN1.5	SADEN1.4	SADEN1.3	SADEN1.2	SADEN1.1	SADEN1.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SADEN1.7-0
Bits 7-0

Slave Address Mask Enable Register 1. This register functions as a mask when comparing serial port 1 addresses for automatic address recognition. When a bit in this register is set, the corresponding bit location in the SADDR1 register will be exactly compared with the incoming serial port 1 data to determine if a receiver interrupt should be generated. When a bit in this register is cleared, the corresponding bit in the SADDR1 register becomes a don't care and is not compared against the incoming data. All incoming data will generate a receiver interrupt when this register is cleared.

CAN 0 Message Center 11 Control Register (C0M11C)

	7	6	5	4	3	2	1	0
SFR BBh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M11C
Bits 7-0

Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 12 Control Register (C0M12C)

	7	6	5	4	3	2	1	0
SFR BCh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M12C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 13 Control Register (C0M13C)

	7	6	5	4	3	2	1	0
SFR BDh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M13C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 14 Control Register (C0M14C)

	7	6	5	4	3	2	1	0
SFR BEh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M14C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

CAN 0 Message Center 15 Control Register (C0M15C)

	7	6	5	4	3	2	1	0
SFR BFh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C0M15C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 0 Message One Control Register (C0M1C;ABh). Please consult the description of that register for more information.

Serial Port Control (SCON1)

	7	6	5	4	3	2	1	0
SFR C0h	SM0/FE_1	SM1_1	SM2_1	REN_1	TB8_1	RB8_1	TI_1	RI_1
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SM0-2
Bits 7-5

Serial Port 1 Mode. These bits control the mode of serial port 1 as shown below. In addition, the SM0 and SM2 bits have secondary functions as shown below.

SM0	SM1	SM2	MODE	FUNCTION	LENGTH	PERIOD
0	0	0	0	Synchronous	8 bits	12 t_{CLK}
0	0	1	0	Synchronous	8 bits	4 t_{CLK}
0	1	X	1	Asynchronous	10 bits	Timer 1 or 2 baud rate equation
1	0	0	2	Asynchronous	11 bits	64 t_{CLK} (SMOD=0) 32 t_{CLK} (SMOD=1)
1	0	1	1	Asynchronous w/ Multiprocessor communication	11 bits	64 t_{CLK} (SMOD=0) 32 t_{CLK} (SMOD=1)
1	1	0	3	Asynchronous	11 bits	Timer 1 or 2 baud rate equation
1	1	1	3	Asynchronous w/ Multiprocessor communication	11 bits	Timer 1 or 2 baud rate equation

SM0/FE_1
Bit 7

Framing Error Flag. When SMOD0 (PCON.6)=0, this bit (SM0) is used to select the mode for serial port 1. When SMOD0 (PCON.6)=1, this bit (FE) will be set upon detection of an invalid stop bit. When used as FE, this bit must be cleared in software. Once the SMOD0 bit is set, modifications to this bit will not affect the serial port mode settings. Although accessed from the same register, internally the data for bits SM0 and FE are stored in different locations.

SM1_1
Bit 6

No alternate function.

SM2-2
Bit 5

Multiple CPU Communications. The function of this bit is dependent on the serial port 0 mode.

Mode 0: Selects 12 t_{CLK} or 4 t_{CLK} period for synchronous serial port 0 data transfers.

Mode 1: When set, reception is ignored (RI_1 is not set) if invalid stop bit received.

Mode 2/3: When this bit is set, multiprocessor communications are enabled in modes 2 and 3. This will prevent the RI_1 bit from being set, and an interrupt being asserted, if the 9th bit received is not 1.

REN_1
Bit 4

Receive Enable. This bit enables/disables the serial port 1 receiver shift register.

0 = Serial port 1 reception disabled.

1 = Serial port 1 receiver enabled (modes 1, 2, 3). Initiate synchronous reception

(mode 0).

TB8_1 Bit 3 — **9th Transmission Bit State.** This bit defines the state of the 9th transmission bit in serial port 1 modes 2 and 3.

RB8_1 Bit 2 — **9th Received Bit State.** This bit identifies the state for the 9th reception bit received data in serial pot 1 modes 2 and 3. In serial port mode 1, when SM2_1=0, RB8_1 is the state of the stop bit. RB8_1 is not used in mode 0.

TI_1 Bit 1 — **Transmitter Interrupt Flag.** This bit indicates that data in the serial port 1 buffer has been completely shifted out. In serial port mode 0, TI_1 is set at the end of the 8th data bit. In all other modes, this bit is set at the end of the last data bit. This bit must be manually cleared by software.

RI_1 Bit 0 — **Transmitter Interrupt Flag.** This bit indicates that a byte of data has been received in the serial port 1 buffer. In serial port mode 1, RI_1 is set at the end of the 8th bit. In serial port mode 1, RI_1 is set after the last sample of the incoming stop bit subject to the state of SM2_1. In modes 2 and 3, RI_1 is set after the last sample of RB8_1. This bit must be manually cleared by software.

Serial Data Buffer 1 (SBUF1)

	7	6	5	4	3	2	1	0
SFR C1h	SBUF1.7	SBUF1.6	SBUF1.5	SBUF1.4	SBUF1.3	SBUF1.2	SBUF1.1	SBUF1.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

SBUF1.7-0 Bits 7-0 — **Serial Data Buffer 1.** Data for serial port 1 is read from or written to this location. The serial transmit and receive buffers are separate registers, but both are addressed at this location.

Power Management Register (PMR)

	7	6	5	4	3	2	1	0
SFR C4h	CD1	CD0	SWB	CTM	4X/$\overline{2X}$	ALEOFF	1	1
	R*-1	R*-0	RW-0	R*-0	R*-0	RW-0	R-1	R-1

R=Unrestricted Read, W= Unrestricted Write, *= See description below, -n=Value after Reset

CD1, CD0
Bits 7-6

Clock Divide Control 1-0. These bits select the number of crystal oscillator clocks required to generate one machine cycle. Switching between modes requires a transition through the divide by 4 mode (CD1, CD0=01). For example, to go from 1 to 1024 clocks per machine cycle the device must first go from 1 to 4 clocks per cycle, and then from 4 to 1024 clocks per cycle. Attempts to perform an invalid transition will be ignored. The setting of these bits will effect the timers and serial ports as shown below.

Attempts to change these bits to the frequency multiplier (1 or 2 clocks per cycle) setting will fail when running from the internal ring oscillator. In addition, it is not possible to change these bits to the 1024 clocks per machine cycle setting while the switchback enable bit (SWB) is set and any of the switchback sources (external interrupts or serial port transmit or receive activity) are active.

CD1:0	4X/$\overline{2X}$	OSCILLATOR CYCLES PER MACHINE. CYCLE	OSC CYCLES PER TIMER 0/1/2 CLOCK.		OSC CYCLES PER TIMER 2 CLK, BAUD RATE GEN.	OSC CYCLES PER SERIAL PORT CLK, MODE 0		OSC CYCLES PER SERIAL PORT CLK, MODE 2	
			TxM=0	TxM=1		SM2=0	SM2=1	SMOD=0	SMOD=1
00	1	1	12	1	2	3	1	64	32
00	0	2	12	2	2	6	2	64	32
01	x	Reserved							
10	x	4	12	4	2	12	4	64	32
11	x	1024	3072	1024	512	3072	1024	64	32

SWB
Bit 5

Switchback Enable. This bit allows an enabled external interrupt or serial port activity to force the Clock Divide Control bits to the divide by 4 state (10) when the microcontroller is in the divide by 1024 state. Upon internal acknowledgement of an external interrupt, the device will switch modes at the start of the jump to the interrupt service routine. Note that this means that an external interrupt must actually be recognized (i.e., be enabled and not masked by higher priority interrupts) for the switchback to occur. For serial port reception, the switch occurs at the start of the instructions following the falling edge of the start bit.

CTM
Bit 4

Crystal Multiplier Enable. The CTM bit enables/disables the Crystal Clock Multiplier. The CTM bit can be changed only when the CD1 and CD0 bits are set to divide by 4 mode and the RGMD is cleared to 0. When cleared this bit disables the Crystal Clock Multiplier to save energy. Setting this bit enables the Crystal Clock Multiplier, permitting the use of the 1 or 2 clock per machine cycle speeds. The following procedure must be performed when setting the CTM bit.

1. Select the desired clock rate via the 4X/$\overline{2X}$ bit. (4X/$\overline{2X}$ =1, 1 clock per cycle, 4X/$\overline{2X}$ =0, 2 clocks per cycle).
2. Set the CTM bit. At this point the CKRY bit (EXIF.3) will be cleared, indicating the internal clock stabilization period has commenced. Software is prohibited from modifying the CD1, CD0 bits while the CKRY bit is cleared.
3. Poll the CKRY bit until it is set.
4. Change CD0, CD1 bits to 00b.

CTM cannot be changed from a 1 to a 0 while the Crystal Clock Multiplier option is selected via the CD1 and CD0 clock control bits. The CTM is also automatically cleared to a logic 0 when the processor enters into a Stop mode.

4X/$\overline{2X}$
Bit 3

System Clock Multiplier. This bit selects the internal crystal oscillator multiplier setting, which in turn establishes a speed of one or two clocks per machine cycle. This bit can only be altered when the CTM bit is cleared to prevent the corruption of the system clock.

0 = The device operates at a rate of two clocks per machine cycle.

1 = The device operates at a rate of one clock per machine cycle.

ALEOFF
Bit 2

ALE Disable. This bit disables the expression of the ALE signal on the device pin during all on-board program and data memory accesses. External memory accesses will automatically enable ALE independent of the ALEOFF bit.

0 = ALE expression is enabled.

1 = ALE expression is disabled.

Bits 1-0

Reserved. These bits will read 1

Status Register (STATUS)

	7	6	5	4	3	2	1	0
SFR C5	PIP	HIP	LIP	-	SPTA1	SPRA1	SPTA0	SPRA0
	R-0	R-0	R-0	R-*	R-0	R-0	R-0	R-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset, *=See description

PIP
Bit 7
Power Fail Priority Interrupt Status. When set, this bit indicates that software is currently servicing a power-fail interrupt. It is cleared when the program executes the corresponding RETI instruction.

HP
Bit 6
High Priority Interrupt Status. When set, this bit indicates that software is currently servicing a high priority interrupt. It is cleared when the program executes the corresponding RETI instruction.

LIP
Bit 5
Low Priority Interrupt Status. When set, this bit indicates that software is currently servicing a low priority interrupt. It is cleared when the program executes the corresponding RETI instruction.

Bit 4
Reserved. Read value will be indeterminate.

SPTA1
Bit 3
Serial Port 1 Transmit Activity Monitor. When set, this bit indicates that data is currently being transmitted by serial port 1. It is cleared when the internal hardware sets the TI_1 bit. Do not alter the Clock Divide Control bits (PMR.7-6) while this bit is set or serial port data may be lost.

SPRA1
Bit 2
Serial Port 1 Receive Activity Monitor. When set, this bit indicates that data is currently being received by serial port 1. It is cleared when the internal hardware sets the RI_1 bit. Do not alter the Clock Divide Control bits (PMR.7–6) while this bit is set or serial port data may be lost.

SPTA0
Bit 1
Serial Port 0 Transmit Activity Monitor. When set, this bit indicates that data is currently being transmitted by serial port 0. It is cleared when the internal hardware sets the TI_1 bit. Do not alter the Clock Divide Control bits (PMR.7-6) while this bit is set or serial port data may be lost.

SPRA0
Bit 0
Serial Port 0 Receive Activity Monitor. When set, this bit indicates that data is currently being received by serial port 0. It is cleared when the internal hardware sets the RI_1 bit. Do not alter the Clock Divide Control bits (PMR.7-6) while this bit is set or serial port data may be lost.

Memory Control Register (MCON)

	7	6	5	4	3	2	1	0
SFR C6h	IDM1	IDM0	CMA	1	PDCE3	PDCE2	PDCE1	PDCE0
	RT-0	RT-0	RT-0	R-1	RT-0	RT-0	RT-0	RT-0

R=Unrestricted Read, T=Timed Access Write Only, -n=Value after Reset

IDM1, IDM0
Bits 7-6

Internal Data Memory Configuration Bits 1-0. These bits establish both the address and type (data and/or program) of the internal 4 KB internal SRAM as shown in the table below.

Note that a special lockout feature prevents the use of the Program and/or Data Memory configuration (IDM1, IDM0 = 11b) and the 10-bit stack pointer (SA=1) at the same time. The IDM1, IDM0 bits can be set to 11b only when the SA bit (ACON.2) is cleared, and the SA bit cannot be set while the IDM1, IDM0 bits are equal to 11b. Attempts to modify the IDMx or SA bits in these situations will fail and the bit(s) will remain unchanged.

		4 KB Internal SRAM	
IDM1	**IDM0**	**Memory Location**	**Memory Assignment**
0	0	00F000h-00FFFFh	Data Memory
0	1	000000h-000FFFh	Data Memory
1	0	400000h-400FFFh	Data Memory
1	1	400000h-400FFFh	Program and/or Data Memory

CMA
Bit 5

CAN Data Memory Assignment. This bit selects the address of the 256 byte blocks of CAN Data Memory associated with both CAN controllers.

CMA	**CAN 0 Memory Address**	**CAN 1 Memory Address**
0 (default)	00EE00h-00EEFFh	00EF00h-00EFFFh
1	401000h-4010FFh	401100h-4011FFh

Bit 4

Reserved

PDCE3
Bit 3

Program/Data Chip Enable 3. This bit selects whether the $\overline{CE3}$ signal functions as the chip enable for external program memory only (PDCE=0), or as a merged chip enable for program and data memory (PDCE=1). When PDCE=1, the microprocessor will use the $\overline{PSEN}$ signal instead of the $\overline{RD}$ signal when reading from external MOVX memory. The Port 4 Control register (P4CNT) determines the memory range associated with $\overline{CE3}$. This bit is ignored if $\overline{CE3}$ has not been previously enabled via the Port 4 Control register.

PDCE2
Bit 2

Program/Data Chip Enable 2. This bit selects whether the $\overline{CE2}$ signal functions as the chip enable for external program memory only (PDCE=0), or as a merged chip enable for program and data memory (PDCE=1). When PDCE=1, the microprocessor will use the $\overline{PSEN}$ signal instead of the $\overline{RD}$ signal when reading from external MOVX memory. The Port 4 Control register (P4CNT) determines the memory range associated with $\overline{CE2}$. This bit is ignored if $\overline{CE2}$ has not been previously enabled via the Port 4 Control register.

PDCE1
Bit 1

Program/Data Chip Enable 1. This bit selects whether the $\overline{CE1}$ signal functions as the chip enable for external program memory only (PDCE=0), or as a merged chip enable for program and data memory (PDCE=1). When PDCE=1, the microprocessor will use the $\overline{PSEN}$ signal instead of the $\overline{RD}$ signal when reading from external MOVX memory. The Port 4 Control register (P4CNT) determines the memory range associated with $\overline{CE1}$. This bit is ignored if $\overline{CE1}$ has not been previously enabled via the Port 4 Control register.

PDCE0
Bit 0

Program/Data Chip Enable 0. This bit selects whether the $\overline{CE0}$ signal functions as the chip enable for external program memory only (PDCE=0), or as a merged chip enable for program and data memory (PDCE=1). When PDCE=1, the microprocessor will use the $\overline{PSEN}$ signal instead of the $\overline{RD}$ signal when reading from external MOVX memory. The Port 4 Control register (P4CNT) determines the memory range associated with $\overline{CE0}$. This bit is ignored if $\overline{CE0}$ has not been previously enabled via the Port 4 Control register.

Timed Access Register (TA)

	7	6	5	4	3	2	1	0
SFR C7h	TA.7	TA.6	TA.5	TA.4	TA.3	TA.2	TA.1	TA.0
	W-1	W-1	W-1	W-1	W-1	W-1	W-1	W-1

W=Unrestricted Write, -n=Value after Reset

TA.7-0
Bits 7-0

Timed Access. Correctly accessing this register permits modification of timed access protected bits. Write AAh to this register first, followed within 3 cycles by writing 55h. Timed access protected bits can then be modified for a period of 3 cycles measured from the writing of the 55h.

Timer 2 Control (T2CON)

	7	6	5	4	3	2	1	0
SFR C8h	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TF2
Bit 7

Timer 2 Overflow Flag. This flag will be set when Timer 2 overflows from FFFFh or the count equal to the capture register in down count mode. It must be cleared by software. TF2 will only be set if RCLK and TCLK are both cleared to 0.

EXF2
Bit 6

Timer 2 External Flag. A negative transition on the T2EX pin (P1.1) or timer 2 underflow/overflow will cause this flag to set based on the CP/RL2 (T2CON.0), EXEN2 (T2CON.3), and DCEN (T2MOD.0) bits. If set by a negative transition, this flag must be cleared to 0 by software. Setting this bit in software or detection of a negative transition on the T2EX pin will force a timer interrupt if enabled.

CP/$\overline{RL2}$	EXEN2	DCEN	RESULT
1	0	X	Negative transitions on P1.1 will not affect this bit.
1	1	X	Negative transitions on P1.1 will set this bit.
0	0	0	Negative transitions on P1.1 will not affect this bit.
0	1	0	Negative transitions on P1.1 will set this bit.
0	X	1	Bit toggles whenever timer 2 underflows/overflows and can be used as a 17th bit of resolution. In this mode, EXF2 will not cause an interrupt.

RCLK
Bit 5

Receive Clock Flag. This bit determines the serial port 0 timebase when receiving data in serial modes 1 or 3.
0 = Timer 1 overflow is used to determine receiver baud rate for serial port 0.
1 = Timer 2 overflow is used to determine receiver baud rate for serial port 0. Setting this bit will force timer 2 into baud rate generation mode. The timer will operate from a divide by 2 of the external clock.

TCLK
Bit 4

Transmit Clock Flag. This bit determines the serial port 0 timebase when transmitting data in serial modes 1 or 3.
0 = Timer 1 overflow is used to determine transmitter baud rate for serial port 0.
1 = Timer 2 overflow is used to determine transmitter baud rate for serial port 0. Setting this bit will force timer 2 into baud rate generation mode. The timer will operate from a divide by 2 of the external clock.

EXEN2
Bit 3

Timer 2 External Enable. This bit enables the capture/ reload function on the T2EX pin if Timer 2 is not generating baud rates for the serial port.

0 = Timer 2 will ignore all external events at T2EX.

1 = Timer 2 will capture or reload a value if a negative transition is detected on the T2EX pin.

TR2
Bit 2

Timer 2 Run Control. This bit enables/disables the operation of timer 2. Halting this timer will preserve the current count in TH2, TL2.

0 = Timer 2 is halted.

1 = Timer 2 is enabled.

C/$\overline{\text{T2}}$
Bit 1

Counter/Timer Select. This bit determines whether timer 2 will function as a timer or counter. Independent of this bit, timer 2 runs at 2 clocks per tick when used in either baud rate generator or clock output mode.

0 = Timer 2 function as a timer. The speed of timer 2 is determined by the T2M bit (CKCON.5).

1 = Timer 2 will count negative transitions on the T2 pin (P1.0).

CP/$\overline{\text{RL2}}$
Bit 0

Capture/Reload Select. This bit determines whether the capture or reload function will be used for timer 2. If either RCLK or TCLK is set, this bit will not function and the timer will function in an auto-reload mode following each overflow.

0 = Auto-reloads will occur when timer 2 overflows or a falling edge is detected on T2EX if EXEN2=1.

1 = Timer 2 captures will occur when a falling edge is detected on T2EX if EXEN2 = 1.

Timer 2 Mode (T2MOD)

	7	6	5	4	3	2	1	0
SFR C9h	1	1	1	D13T1	D13T2	1	T2OE	DCEN
	R-1	R-1	R-1	RW-0	RW-0	R-1	RW-0	RW-0

R=Unrestricted Read, W= Unrestricted Write, -n=Value after Reset

Bits 7-5 Reserved.

D13T1 Bit 4 **Divide by Thirteen Clock Option For Timer 1**. The D13T1 bit provides an alternate clock source to the Timer 1 in place of the normal external T1 input pin. When D13T1 is cleared to 0 (the default Reset state), the clock source for Timer 1 is supplied through the standard T1 external input pin, the divide by 12 of the oscillator (T1M = 0) or the divide by 4 of the oscillator (T1M = 1), as controlled by T1M and C/$\overline{T}$. When D13T1 is set to a 1 the clock source for Timer 1 is supplied through a separate divide by 13 of the crystal oscillator independent of T1M. The C/$\overline{T}$ bit must also be programmed to a 1 to select the divide by 13 counter.

D13T2 Bit 3 **Divide by Thirteen Clock Option For Timer 2**. The D13T2 bit provides an alternate clock source to the Timer 2 in place of the normal external T2 input pin. When D13T2 is cleared to 0 (the default Reset state), the clock source for Timer 2 is supplied through the standard T2 external input pin, the divide by 12 of the oscillator (T2M = 0) or the divide by 4 of the oscillator (T2M = 1), as controlled by T2M and C/$\overline{T2}$. When D13T2 is set to a 1 the clock source for Timer 2 is supplied through a separate divide by 13 of the crystal oscillator independent of T2M. The C/$\overline{T2}$ bit must also be programmed to a 1 to select the divide by 13 counter.

Bit 2 Reserved.

T2OE Bit 1 **Timer 2 Output Enable.** This bit enables/disables the clock output function of the T2 pin (P1.0). 0 = The T2 pin functions as either a standard port pin or as a counter input for timer 2. 1 = Timer 2 will drive the T2 pin with a clock output if C/T2=0. Also, timer 2 rollovers will not cause interrupts.

DCEN Bit 0 **Down Count Enable.** This bit, in conjunction with the T2EX pin, controls the direction that timer 2 counts in 16-bit auto-reload mode.

DCEN	T2EX	DIRECTION
1	1	Up
1	0	Down
0	X	Up

Timer 2 Capture LSB (RCAP2L)

	7	6	5	4	3	2	1	0
SFR CAh	RCAP2L.7	RCAP2L.6	RCAP2L.5	RCAP2L.4	RCAP2L.3	RCAP2L.2	RCAP2L.1	RCAP2L.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

RCAP2L.7-0
Bits 7-0

Timer 2 Capture LSB. This register is used to capture the TL2 value when timer 2 is configured in capture mode. RCAP2L is also used as the LSB of a 16-bit reload value when timer 2 is configured in auto-reload mode.

Timer 2 Capture MSB (RCAP2H)

	7	6	5	4	3	2	1	0
SFR CBh	RCAP2H.7	RCAP2H.6	RCAP2H.5	RCAP2H.4	RCAP2H.3	RCAP2H.2	RCAP2H.1	RCAP2H.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

RCAP2H.7-0
Bits 7-0

Timer 2 Capture MSB. This register is used to capture the TH2 value when timer 2 is configured in capture mode. RCAP2H is also used as the MSB of a 16-bit reload value when timer 2 is configured in auto-reload mode.

Timer 2 LSB (TL2)

	7	6	5	4	3	2	1	0
SFR CCh	TL2.7	TL2.6	TL2.5	TL2.4	TL2.3	TL2.2	TL2.1	TL2.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TL2.7-0
Bits 7-0

Timer 2 LSB. This register contains the least significant byte of Timer 2.

Timer 2 MSB (TH2)

	7	6	5	4	3	2	1	0
SFR CDh	TH2.7	TH2.6	TH2.5	TH2.4	TH2.3	TH2.2	TH2.1	TH2.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

TL2.7-0
Bits 7-0

Timer 2 MSB. This register contains the least significant byte of Timer 2.

Clock Output Register (COR)

	7	6	5	4	3	2	1	0
SFR CEh	IRDACK	C1BPR7	C1BPR6	C0BPR7	C0BPR6	COD1	COD0	CLKOE
	RT-0	RT-0	RT-0	RT-0	RT-0	RT-0	RT-0	RT-0

R=Unrestricted Read, T=Timed Access Write Only, -n=Value after Reset

IRDACK
Bit 7

IRDA Clock Output Enable. This bit determines which clock signal will be asserted on port pin P3.5 when the CLKOE bit is set. Please consult the description of the CLKOE bit for more information.

0 = The port pin P3.5 will output a signal determined by the Clock Output Divide Bits (COR.1 and COR.2).

1 = The port pin P3.5 will output a signal that is 16 times the programmed baud rate associated with Serial Port 0.

C1BPR7, C1BPR6
Bit 6-5

CAN 1 Baud Rate Prescaler Bits. These bits establish bits 7 and 6 of the eight-bit CAN 1 Baud Rate Prescaler. These bits can not be modified while the SWINT bit in the CAN1 Control Register is cleared to 0. The remaining six bits are located in the CAN 1 Bus Timing Register Zero (C1BT0) located in the CAN MOVX memory.

C0BPR7, C0BPR6
Bit 4-3

CAN 0 Baud Rate Prescaler Bits. These bits establish bits 7 and 6 of the eight-bit CAN 0 Baud Rate Prescaler. These bits can not be modified while the SWINT bit in the CAN 0 Control Register is cleared to 0. The remaining six bits are located in the CAN 0 Bus Timing Register Zero (C0BT0) located in the CAN MOVX memory.

COD1, COD0
Bit 2-1

Clock Output Divide Select bits. These bits select the frequency of signal asserted on port pin P3.5 when CLKOE=1 and IRDACK=0. Please consult the description of the CLKOE bit for more information.

COD1	COD0	P3.5 Output Frequency
0	0	System clock divided by 2
0	1	System clock divided by 4
1	0	System clock divided by 6
1	1	System clock divided by 8

CLKOE
Bit 0

External Clock Output Enable. This bit enables the optional clock output functions on port pin P3.5. Associated bits are shown in the following table.

CLKOE	IRDACK	COD1	COD0	P3.5 Output
0	x	x	x	General purpose I/O or T1 function
1	1	x	x	16 x serial port 0 baud rate
1	0	0	0	System clock divided by 2
1	0	0	1	System clock divided by 4
1	0	1	0	System clock divided by 6
1	0	1	1	System clock divided by 8

Program Status Word (PSW)

	7	6	5	4	3	2	1	0
SFR D0h	CY	AC	F0	RS1	RS0	0V	F1	PARITY
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

CY
Bit 7

Carry Flag. This bit is set when if the last arithmetic operation resulted in a carry (during addition) or a borrow (during subtraction). Otherwise it is cleared to 0 by all arithmetic operations.

AC
Bit 6

Auxiliary Carry Flag. This bit is set to 1 if the last arithmetic operation resulted in a carry into (during addition), or a borrow (during subtraction) from the high order nibble. Otherwise it is cleared to 0 by all arithmetic operations.

F0
Bit 5

User Flag 0. This is a bit-addressable, general-purpose flag for software control.

RS1, RS0
Bits 4-3

Register Bank Select 1–0. These bits select which register bank is addressed during register accesses.

RS1	RS0	REGISTER BANK	ADDRESS
0	0	0	00h – 07h
0	1	1	08h – 0Fh
1	0	2	10h – 17h
1	1	3	18h – 1Fh

OV
Bit 2

Overflow Flag. This bit is set to 1 if the last arithmetic operation resulted in a carry (addition), borrow (subtraction), or overflow (multiply or divide). Otherwise it is cleared to 0 by all arithmetic operations.

F1
Bit 1

User Flag 1. This is a bit-addressable, general-purpose flag for software control.

PARITY
Bit 0

Parity Flag. This bit is set to 1 if the modulo-2 sum of the eight bits of the accumulator is 1 (odd parity); and cleared to 0 on even parity.

Multiplier Control Register Zero (MCNT0)

	7	6	5	4	3	2	1	0
SFR D1h	$\overline{\text{LSHIFT}}$	CSE	SCE	MAS4	MAS3	MAS2	MAS1	MAS0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Read, -n=Value after Reset

$\overline{\text{LSHIFT}}$
Bit 7

Left Shift. This bit works in conjunction with the SCE and CSE bits to determine the direction and path of arithmetic accelerator shift operations as shown below. When $\overline{\text{LSHIFT}}$ =0, shift operations will shift from the LSb to the MSb, and vice versa when $\overline{\text{LSHIFT}}$ =1. $\overline{\text{LSHIFT}}$ does not alter any other type of calculation other than the shift function. The $\overline{\text{LSHIFT}}$ bit is cleared to 0 following a either a system reset or the initialization of the accelerator.

CSE
Bit 6

Circular Shift Enable. This bit works in conjunction with the SCE and $\overline{\text{LSHIFT}}$ bits to determine the direction and path of arithmetic accelerator shift operations as shown below. When CSE=1, shifts of the arithmetic accelerator will wrap bit 31 to bit 0 or vice versa depending on the settings of the $\overline{\text{LSHIFT}}$ bits.

When CSE is cleared to a 0, all left or right shifts will shift cleared bit values into the most significant bit for a right shift and the least significant bit for a left shift. When CSE is set to a 1 and SCB is set to a 1, the most significant bit will be shifted into the 32 Bit Carry Bit when doing a left shift and least most significant bit will be shifted into the 32 Bit Carry Bit when doing a right shift.

The CSE bit is cleared to 0 following a system reset.

SCE
Bit 5

Shift Carry Enable. This bit works in conjunction with the CSE and $\overline{\text{LSHIFT}}$ bits to determine the direction and path of arithmetic accelerator shift operations as shown below.

When SCE=1 the arithmetic accelerator carry bit will be shifted into the LSb for a left shift and into the MSb for a right shift. When SCE=0, shifts will not incorporate the arithmetic accelerator carry bit as a part of the shifting process. If CSE=0 the arithmetic accelerator carry bit will remain unchanged during the shift process. If CSE=1 the MSb will be shifted into the carry bit on a left shift and the least most significant bit of the arithmetic accelerator will be shifted into the carry bit on a right shift. The SCE bit is cleared to 0 following a system reset.

			Arithmetic Accelerator Values After Shift		
SCE	**CSE**	**$\overline{\text{LSHIFT}}$**	**MSb (bit 31)**	**LSb (bit 0)**	**Carry bit**
0	0	0	Previous bit 30	Previous bit 0	unchanged
0	0	1	0	Previous bit 1	unchanged
0	1	0	Previous bit 30	Previous bit 31	unchanged
0	1	1	Previous bit 0	Previous bit 1	unchanged
1	0	0	Previous bit 30	Previous carry	unchanged
1	0	1	Previous carry	Previous bit 1	unchanged
1	1	0	Previous bit 30	Previous carry	Previous bit 31
1	1	1	Previous carry	Previous bit 1	Previous bit 0

MAS4-0
Bits 4-0

Multiplier Register Shift Bits. These bits determine the number of shifts performed when a shift operation is performed with the arithmetic accelerator, and are also used to indicate how many shifts were performed during a previous normalization operation. These bits are cleared to 00000b following a system reset or the initialization of the arithmetic accelerator.

When these bits are cleared to 00000b after loading the arithmetic accelerator, the device will normalize the 32-bit value loaded into the arithmetic accelerator Accumulator, rather than shifting it. Following the normalization operation, the MAS4-0 bits will be modified to indicate how many shifts were performed.

MAS4	MAS3	MAS2	MAS1	MAS0	Number of shifts of Arithmetic Accelerator Accumulator
0	0	0	0	0	Normalization
0	0	0	0	1	Shift by 1
0	0	0	1	0	Shift by 2
0	0	0	1	1	Shift by 3
.	.	.	.	.	.
.	.	.	.	.	.
1	1	1	1	0	Shift by 30
1	1	1	1	1	Shift by 31

Multiplier Control Register One (MCNT1)

	7	6	5	4	3	2	1	0
SFR D2h	MST	MOF	SCB	CLM	1	1	1	1
	RW-0	R-0	RW-0	RW-0	R-1	R-1	R-1	R-1

R=Unrestricted Read, W=Unrestricted Read, -n=Value after Reset

MST Bit 7 — **Multiply/Accumulate Status Flag.** The MST bit serves as a busy flag for the multiplier/accumulate hardware. The bit is set automatically when the processor begins loading data into the MA or MB register, and will remain set until the assigned task is completed. MST is automatically cleared by the multiplier/accumulate hardware once an assigned task is completed and the results are ready for the processor to read. MST=0 also indicates that the accelerator has been initialized and can be loaded with new values. Clearing this bit via software from a previous high state will terminate the current operation and initialize the multiplier, allowing the immediate loading of new data into MA and/or MB to perform a new calculation.

MOF Bit 6 — **Multiply Overflow Flag.** The MOF flag bit is cleared following a either a system reset or the initialization of the accelerator. The MOF bit is automatically set when the accelerator detects a divide by zero, or when the result of the calculation is larger than FFFFh.

SCB Bit 5 — **Shift Carry Bit.** The SCB bit is used as a carry bit for shift operation when SCE bit is set to 1. Note that the SCB will not be cleared at the beginning of a new operation and must be cleared by a write to this bit or a system reset.

CLM Bit 4 — **Clear Accelerator Registers.** Writing a one to this bit will clear the MA, MB, and MC registers. Reading this bit will always return a logic 0.

Bit 3-0 — Reserved

Multiplier A Register (MA)

	7	6	5	4	3	2	1	0
SFR D3h								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Read, -n=Value after Reset

Bits 7-0 — **Multiplier A Register.** The MA Register is used as both a source and result register for various arithmetic accelerator functions. When in the source mode it is loaded with the numerator for divide operations and the multiplicand when performing multiply operations. The MA register also holds the quotient of the divide operations, multiply product, shift results, and mantissa of the normalize function.

The MA register can receive or hold up to a 32-bit result, accessed via a series of sequential writes to or reads from the register. Details of the sequencing are explained in the arithmetic accelerator section of the User's Guide.

Multiplier B Register (B)

	7	6	5	4	3	2	1	0
SFR D4h								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Read, -n=Value after Reset

Bits 7-0 **Multiplier B Register.** The MB Register is used as both a source and result register for various arithmetic accelerator functions. When in the source mode it is loaded with the dividend for divide operations and the multiplier when performing multiply operations. The MA register also holds the remainder of the divide operations.

The MB register can receive or hold up to a 32-bit result, accessed via a series of sequential writes to or reads from the register. Details of the sequencing are explained in the arithmetic accelerator section of the User's Guide.

Multiplier C Register (C)

	7	6	5	4	3	2	1	0
SFR D5h								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Read, -n=Value after Reset

Bits 7-0 **Multiplier C Register.** The MC Register allows access to the 40-bit accumulator register for the arithmetic accelerator. Each time a multiply or divide (but not shift or normalization) function is performed with the arithmetic accelerator the result is added to the previous value in the MC register.

Data is read from the 40-bit accumulator MSB first, and five read operations must be performed to read the entire value. Writes to the accumulator are performed LSB first, but software may write as few registers as needed (i.e., 2 in the case of a 16-bit value) provided the unloaded registers have been previously initialized to 00h. Details of the sequencing are explained in the arithmetic accelerator section of the User's Guide.

All 40 bits of the accumulator are cleared by a system reset, the setting of the CLM bit or the setting of the MST bit in the MCNT1 SFR. The register can also be cleared by performing five writes of 00h to the MC register.

CAN 1 Receive Message Stored Register 0 (C1RMS0)

	7	6	5	4	3	2	1	0
SFR D6h								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 1 Receive Message Stored Register 0. This register indicates which of CAN 1 message centers 1-8 have successfully received and stored a message since the last read of this register. A logic one in a location indicates a message has been received and stored for that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C1RMS1 register to ascertain the status of all message centers.

C1RMS0.7 **Message Center 8, Message Received and Stored**
Bit 7

C1RMS0.6 **Message Center 7, Message Received and Stored**
Bit 6

C1RMS0.5 **Message Center 6, Message Received and Stored**
Bit 5

C1RMS0.4 **Message Center 5, Message Received and Stored**
Bit 4

C1RMS0.3 **Message Center 4, Message Received and Stored**
Bit 3

C1RMS0.2 **Message Center 3, Message Received and Stored**
Bit 2

C1RMS0.1 **Message Center 2, Message Received and Stored**
Bit 1

C1RMS0.0 **Message Center 1, Message Received and Stored**
Bit 0

CAN 1 Receive Message Stored Register 1 (C1RMS1)

	7	6	5	4	3	2	1	0
SFR D7h								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 1 Receive Message Stored Register 1. This register indicates which of CAN 1 message centers 9-15 have successfully received and stored a message since the last read of this register. A logic one in a location indicates a message has been received and stored for that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C1RMS0 register to ascertain the status of all message centers.

Bit 7 Reserved

C1RMS1.6 **Message Center 15, Message Received and Stored**
Bit 6

C1RMS1.5 **Message Center 14, Message Received and Stored**
Bit 5

C1RMS1.4 **Message Center 13, Message Received and Stored**
Bit 4

C1RMS1.3 **Message Center 12, Message Received and Stored**
Bit 3

C1RMS1.2 **Message Center 11, Message Received and Stored**
Bit 2

C1RMS1.1 **Message Center 10, Message Received and Stored**
Bit 1

C1RMS1.0 **Message Center 9, Message Received and Stored**
Bit 0

Watchdog Control (WDCON)

	7	6	5	4	3	2	1	0
SFR D8h	SMOD	POR	EPF1	PFI	WDIF	WTRF	EWT	RWT
	RW-0	RT-*	RW-0	RW-*	RT-0	RW-*	RT-*	RT-0

R=Unrestricted Read, W=Unrestricted Write, T=Timed Access Write Only ,
-n=Value after Reset, *=See Description

SMOD
Bit 7

Serial Modification. This bit controls the doubling of the serial port 1 baud rate in modes 1, 2, and 3.

0 = Serial port 1 baud rate operates at normal speed
1 = Serial port 1 baud rate is doubled.

POR
Bit 6

Power-on Reset Flag. This bit indicates whether the last reset was a power-on reset. This bit is typically interrogated following a reset to determine if the reset was caused by a power-on reset. It must be cleared by a Timed Access write before the next reset of any kind or the software may erroneously determine that another power-on reset has occurred. This bit is set following a power-on reset and unaffected by all other resets.

0 = Last reset was from a source other than a power-on reset
1 = Last reset was a power-on reset.

EPFI
Bit 5

Enable Power fail Interrupt. This bit enables/disables the ability of the internal band-gap reference to generate a power-fail interrupt when V_{CC} falls below approximately 4.5 volts. While in Stop mode, both this bit and the Band-gap Select bit, BGS (EXIF.0), must be set to enable the power-fail interrupt.

0 = Power-fail interrupt disabled.

1 = Power-fail interrupt enabled during normal operation. Power-fail interrupt enabled in Stop mode if BGS is set.

PFI
Bit 4

Power fail Interrupt Flag. When set, this bit indicates that a power-fail interrupt has occurred. This bit must be cleared in software before exiting the interrupt service routine, or another interrupt will be generated. Setting this bit in software will generate a power-fail interrupt, if enabled.

WDIF
Bit 3

Watchdog Interrupt Flag. This bit, in conjunction with the Watchdog Timer Interrupt Enable bit, EWDI (EIE.4), and Enable Watchdog Timer Reset bit (WDCON.1), indicates if a watchdog timer event has occurred and what action will be taken. This bit must be cleared in software before exiting the interrupt service routine, or another interrupt will be generated. Setting this bit in software will generate a watchdog interrupt if enabled. This bit can only be modified using a Timed Access Procedure.

EWT	EWDI	WDIF	RESULT
X	X	0	No watchdog event has occurred.
0	0	1	Watchdog time-out has expired. No interrupt has been generated.
0	1	1	Watchdog interrupt has occurred.
1	0	1	Watchdog time-out has expired. No interrupt has been generated. Watchdog timer reset will occur in 512 cycles if RWT is not strobed.
1	1	1	Watchdog interrupt has occurred. Watchdog timer reset will occur in 512 cycles if RWT is not set using a Timed Access procedure.

WTRF
Bit 2

Watchdog Timer Reset Flag. When set, this bit indicates that a watchdog timer reset has occurred. It is typically interrogated to determine if a reset was caused by watchdog timer reset. It is cleared by a power- on reset, but otherwise must be cleared by software before the next reset of any kind or software may erroneously determine that a watchdog timer reset has occurred. Setting this bit in software will not generate a watchdog timer reset. If the EWT bit is cleared, the watchdog timer will have no effect on this bit.

EWT
Bit 1

Enable Watchdog Timer Reset. This bit enables/disables the ability of the watchdog timer to reset the device. This bit has no effect on the ability of the watchdog timer to generate a watchdog interrupt. The time-out period of the watchdog timer is controlled by the Watchdog Timer Mode Select bits (CKCON.7-6). Clearing this bit will disable the ability of the watchdog timer to generate a reset, but have no affect on the timer itself, or its ability to generate a watchdog timer interrupt. This bit can only be modified using a Timed Access Procedure. This bit is unaffected by all other resets.

0 = A timeout of the watchdog timer will not cause the device to reset.
1 = A timeout of the watchdog timer will cause the device to reset.

RWT
Bit 0

Reset Watchdog Timer. Setting this bit will reset the watchdog timer count. This bit must be set using a Timed Access procedure before the watchdog timer expires, or a watchdog timer reset and/or interrupt will be generated if enabled. The time-out period is defined by the Watchdog Timer Mode Select bits (CKCON.7-6). This bit will always be 0 when read.

CAN 1 Transmit Message Acknowledgement Register 0 (C1TMA0)

	7	6	5	4	3	2	1	0
SFR DEh								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 1 Transmit Message Acknowledgement Register 0. This register indicates which of CAN 1 message centers 1-8 have successfully transmitted a message since the last read of this register. A logic one in a location indicates a message has been transmitted from that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C1TMA1 register to ascertain the status of all message centers.

C1TMA0.7 **Message Center 8, Message Transmitted**
Bit 7

C1TMA0.6 **Message Center 7, Message Transmitted**
Bit 6

C1TMA0.5 **Message Center 6, Message Transmitted**
Bit 5

C1TMA0.4 **Message Center 5, Message Transmitted**
Bit 4

C1TMA0.3 **Message Center 4, Message Transmitted**
Bit 3

C1TMA0.2 **Message Center 3, Message Transmitted**
Bit 2

C1TMA0.1 **Message Center 2, Message Transmitted**
Bit 1

C1TMA0.0 **Message Center 1, Message Transmitted**
Bit 0

CAN 1 Transmit Message Acknowledgement Register 1 (C1TMA1)

	7	6	5	4	3	2	1	0
SFR DFh								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, -n=Value after Reset

CAN 1 Transmit Message Acknowledgement Register 1. This register indicates which of CAN 1 message centers 9-15 have successfully transmitted a message since the last read of this register. A logic one in a location indicates a message has been transmitted for that message center. This register is automatically cleared to 00h when read. This register should always be read in conjunction with the C1TMA0 register to ascertain the status of all message centers.

Bit 7 Reserved

C1TMA1.6 **Message Center 15, Message Transmitted**
Bit 6

C1TMA1.5 **Message Center 14, Message Transmitted**
Bit 5

C1TMA1.4 **Message Center 13, Message Transmitted**
Bit 4

C1TMA1.3 **Message Center 12, Message Transmitted**
Bit 3

C1TMA1.2 **Message Center 11, Message Transmitted**
Bit 2

C1TMA1.1 **Message Center 10, Message Transmitted**
Bit 1

C1TMA1.0 **Message Center 9, Message Transmitted**
Bit 0

Accumulator (A or ACC)

	7	6	5	4	3	2	1	0
SFR E0h	ACC.7	ACC.6	ACC.5	ACC.4	ACC.3	ACC.2	ACC.1	ACC.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

ACC.7-0 **Accumulator.** This register serves as the accumulator for arithmetic operations.
Bits 7-0 It is functionally identical to the accumulator found in the 80C32.

CAN 1 Control Register (C1C)

	7	6	5	4	3	2	1	0
SFR E3h	ERIE	STIE	PDE	SIESTA	CRST	AUTOB	ERCS	SWINT
	RW-0	RW-0	RW-0	RW-0	RT-1	RW-0	RW-0	RW-1

R=Unrestricted Read, W=Unrestricted Write, T=Timed Access Write Only, -n=Value after Reset

ERIE
Bit 7

CAN 1 Error Interrupt Enable.

0 = CAN 1 Error Interrupt is disabled.

1 = Setting this bit while the C1IE bit (EIE.5) and Global Interrupt Enable bits (IE.7) are set will generate an interrupt if the CAN 1 Bus Off (BUSOFF) or CAN 1 Error Count Exceeded bit (CECE) bits are set.

STIE
Bit 6

CAN 1 Status Interrupt Enable.

0 = CAN 1 Status Interrupt is disabled.

1 = If the C1IE bit (EIE.5) is set, an interrupt will be generated if the CAN 1 Transmit Status bit (TXS), Receive Status bit (RXS) or the Wake-Up Status bit (WKS) is set. An interrupt will also be generated if the Status Error bits (ER2-0) change a non-000b or non-111b state.

PDE
Bit 5

CAN 1 Power Down Enable. Setting this bit places the CAN 1 module into its lowest power mode. The module will enter Power Down mode immediately upon setting this bit, or following the completion of the current reception, transmission, arbitration failure, or error condition on CAN 1. Software can poll the PDE bit to ascertain whether the microcontroller has entered Power Down mode (PDE=1) or is waiting for a current CAN operation to complete (PDE=0) before entering Power Down Mode.

Power Down mode is exited by clearing the PDE bit or by any reset of the microcontroller. The CAN 1 module will begin operation after the receipt of 11 consecutive recessive bits.

The Wake-Up Status bit, WKS, is a logical OR of this bit and the SIESTA bit.

SIESTA
Bit 4

CAN 1 Siesta Mode Enable. Setting this bit places the CAN 1 module into a low power mode. The module will enter Siesta mode immediately upon setting this bit, or following the completion of the current reception, transmission, arbitration failure, or error condition on CAN 1. Software can poll the SIESTA bit to ascertain whether the microcontroller has entered Siesta mode (SIESTA =1) or is waiting for a current CAN operation to complete (SIESTA =0) before entering Siesta Mode.

Siesta mode is exited by clearing the Siesta bit, detecting CAN 1 bus activity, or setting either the CRST or SWINT bits to 1. The CAN 1 module will begin operation after the receipt of 11 consecutive recessive bits.

The Wake-Up Status bit, WKS, is a logical OR of this bit and the PDE bit.

CRST
Bit 3

CAN 1 Reset. Setting this bit via a Timed Access write will reset all CAN 1 registers in the SFR map to their reset default states. The module will reset the registers immediately upon setting this bit, or following the completion of the current reception, transmission, arbitration failure, or error condition on CAN 1. Software can poll the CRST bit to ascertain whether the microcontroller has successfully reset the registers (CRST =1) or is waiting for a current CAN operation to complete (CRST =0) before resetting the registers. Setting the CRST bit also clears the transmit and receive error counters and sets the SWINT bit.

CRST must be cleared by software to remove the CAN reset. The state of the SWINT and BUSOFF bits determines the action of the device when the CRST bit is cleared.

AUTOB
Bit 2

CAN 1 Autobaud. Setting this bit allows the CAN 1 module to establish proper CAN bus timing without disrupting the normal data flow between other nodes on the CAN Bus. When in the autobaud mode, incoming data on the C1RX pin is internally ANDed with transmit data generated by the CAN 1 module. An internal loop back feeds this combined data stream back into the input of the CAN 1 module. At the same time, C1TX pin is placed into a recessive state to prevent driving non-synchronized data (creating CAN Bus errors to other nodes) while attempting to synchronize the processor with the CAN Bus.

With AUTOB = 1, the microcontroller auto-baud algorithm will make use of the CAN 1 Status Register RXS and error status bits to determine when a message is successfully received (when AUTOB =1, a successful receive does not require a store). Each successive baud rate attempt is proceeded by the microcontroller clearing the transmit and receive error counters via a write of 00h to the Transmit Error SFR Register and a read of the CAN 1 Status Register to clear the previous Status Change Interrupt. Note that a write to the Transmit Error SFR Register automatically resets the CAN fault confinement state machine to an initial (error active) state if the error counters are cleared to 00h. If, however, the error counters are programmed to a value greater than 128, the CAN module will be in a error passive state. Appropriate flags are set when the error counter is written with any value. A write of the Status Register is also used to remove the previous error value in the ER2-0 bits. Clearing the error counters will also clear the EC96 bit, if set.

When BUSOFF = 1, software is prohibited from writing to the error counters by virtue of the fact that the SWINT bit is also forced to a 0 state during the period that the CAN module performs a bus recovery and power up sequence. Once the CAN module has removed itself from the Bus Off condition it will also clear BUSOFF = 0, set SWINT = 1, and will clear both the transmit and receive error counters to 00h.

ERCS
Bit 1

CAN 1 Error Count Select. This bit selects the number of transmit or receive errors that will cause the CAN 1 Error Count Exceeded bit, CECE (C1S.6), to be set.

0 = CECE bit set when the transmit or receive error counters exceed 95 errors.
1 = CECE bit set when the transmit or receive error counters exceed 127 errors.

SWINT
Bit 0

CAN 1 Software Initialization Enable. This bit enables (SWINT=1) and disables (SWINT=0) software write access to the first 16 bytes of the CAN 1 MOVX SRAM. These bytes contain the CAN 1 Control/Status/Mask Registers. Read access to all bytes in the CAN 1 MOVX SRAM is permitted at all times, regardless of the state of the SWINT bit.

Setting SWINT=1 disables CAN 1 Bus activity, allowing software access to the CAN 1 Control/Status/Mask Registers without corrupting CAN Bus transmission or reception. A special lockout procedure delays the internal assertion of the SWINT bit until all CAN 1 activity has ceased. The following procedure must be followed when setting the SWINT bit to prevent the accidental corruption of CAN Bus activity:

3. Write a 1 to the SWINT bit, starting the internal process to enter the software initialization process.
4. Poll the SWINT bit until it is set. The lockout circuit will hold SWINT=0 if it detects a reception, transmission, or arbitration in progress. When one of these conditions ceases, or if an error occurs, the CAN module will set SWINT=1, indicating that the CAN module is disabled and software can now write to the first 16 bytes of the CAN 1 MOVX SRAM. Attempts to modify the first 16 bytes of the CAN 1 MOVX SRAM while SWINT=0 will fail, leaving the bytes unchanged.

The SWINT bit controls access to several other bits and registers. The CAN 1 Transmit Error Register (C1TE;A6h) and CAN 1 Receive Error Register (C1RE;A7h) are only modifiable while SWINT=1. Setting SWINT=1 automatically clears the SIESTA bit, and attempts to set SWINT=1 and SIESTA=1 in the same write to the C1C register will result in SWINT=1 and SIESTA=0.

The BUSOFF bit has a direct interaction with the SWINT bit. When a Bus Off condition is detected (BUSOFF=1), the CAN module will automatically clear SWINT=0 and initiate a bus recovery and power-up sequence. Write access to the SWINT bit is prohibited until the Bus Off condition has been cleared and BUSOFF has been reset to 0.

The SWINT bit is also set automatically following a system reset, the setting of the CRST bit in the CAN 1 Control Register, or programming the CAN Bus Timing Registers (C1BT0, C1BT1 in the MOVX SRAM) to 00h (an invalid state). As a precaution against utilizing the CAN with invalid bus timing, the SWINT bit cannot be cleared while C1BT0=C1BT1=00h. When this bit is cleared, the CAN 1 module will initiate a CAN Bus synchronization after the CAN module executes a power-up sequence (reception of 11 consecutive recessive bits.)

CAN 1 Status Register (C1S)

	7	6	5	4	3	2	1	0
SFR E4h	BUSOFF	CECE	WKS	RXS	TXS	ER2	ER1	ER0
	R-0	R-0	R-0	RW-0	RW-0	R-0	R-0	R-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

BUSOFF
Bit 7

CAN 1 Bus Off. When BUSOFF = 1, the CAN 1 Bus is disabled and is not capable of receiving or transmitting messages. This condition is the result of the transmit error counter reaching a count of 256. When the CAN 1 module detects an error count of 256 the CAN module will automatically set BUSOFF = 1 and clear SWINT = 0.

BUSOFF is cleared to a 0 to enable CAN 1 Bus activity when the CAN processor completes both the busoff recovery (128 X 11 consecutive recessive bits) and the power-up sequence (11 consecutive recessive bits). Once the CAN module has completed this relationship it will set SWINT = 1 and will enter into the software initialization state. Once software has cleared SWINT to a 0, the CAN module will be enabled to transmit and receive messages. When BUSOFF = 0, the CAN 1 Bus is enabled to receive or transmit messages. A change in the state of BUSOFF from a previous 0 to a 1 will generate an interrupt if the ERIE, C1IE and IE SFR register bits are set. All microcontroller writes to the SWINT bit are disabled when BUSOFF = 1. Both the transmit and receive error counters are cleared to 00 hex when the Bus Off condition is cleared by the CAN module and BUSOFF is cleared to 0.

CECE
Bit 6

CAN 1 Error Count Exceeded. This bit operates in one of two modes, depending on the state of the ERCS bit in the CAN 1 Control Register.

ERCS = 0 (Error count limit=96) In this mode when CECE=1, the interrupt flag indicates that either the CAN 1 Transmit Error Counter or the CAN 1 Receive Error Counter has reached an error count of 96, which represents an exceptionally high number of errors. CECE=0 indicates that both error counters have an error count of less than 96. A 0 to 1 transition of CECE will generate an interrupt if the ERIE, C1IE and IE SFR bits are set.

ERCS = 1 (Error count limit=128) In this mode when CECE=1, the interrupt flag indicates that either the CAN 1 Transmit Error Counter or the CAN 1 Receive Error Counter has reached an error count of 128, which represents an exceptionally high number of errors. CECE = 0 indicates that the current Transmit Error Counter and Receive Error Counter both have an error count of less than 128. A change in the state of CECE from either a previous 0 to a 1 *or from a previous 1 to 0* will generate an interrupt if the ERIE, C1IE and IE SFR bits are set.

WKS
Bit 5

CAN 1 Wake-up Status. When WKS=1, the CAN 1 module is in either SIESTA or Power Down mode. Clearing both the SIESTA and PDE bits will force the WKS=0. A change in the state of WKS from a previous 1 to 0 will generate an interrupt if the STIE, C1IE and IE SFR bits are set.

RXS
Bit 4

CAN 1 Receive Status. This bit indicates whether or not messages have been received since the last read of the CAN 1 Status Register. RXS is only set by the CAN 1 logic and must be cleared by the Microcontroller software, the CRST bit, or a system Reset.

1 = The meaning of RXS=1 is dependent on the Autobaud bit, AUTOB.

AUTOB=0, RXS = 1 indicates that a message has been both successfully received and stored in one of the message centers by CAN 1 since the last read of the CAN 1 Status Register.

AUTOB=1, RXS = 1 indicates that a message has been successfully received by CAN 1 since the last read of the CAN 1 Status Register. Note that messages that are successfully received without errors but do not pass the arbitration filtering will still set the RXS bit.

0 = No messages have been successfully received since the last read of the CAN 1 Status Register.

When STIE= 1 and the RXS bit transitions from 0 to 1, the CAN Interrupt Register (C1IE;A5h) will change to 01h to indicate a pending interrupt due to a change in the CAN Status Register. Reading any bit in the C1S register will clear the pending interrupt, causing the C1IE register to change to 00h if no interrupts are pending or the appropriate value if a lower priority message center interrupt is pending. If a second successful reception is detected prior to or after the clearing of the RXS bit in the Status Register, a second status change interrupt flag will be set, issuing a second interrupt. Each new successful reception will generate an interrupt request independent of the previous state of the RXS bit, as long as the CAN Status Register has been read to clear the previous status change interrupt flag. Note that if software changes RXS from 0 to 1, an artificial Status Change Interrupt (STIE=1) will be generated. Thus, if RXS was previously set to 0 and a reception was successful, RXS will be set to 1 and an enabled interrupt may be asserted. An interrupt may be asserted (if enabled) if software changes RXS from 0 to 1. If RXS was previously set to 1 and a reception was successful, RXS remains set and an interrupt may be asserted if enabled. No interrupt will be asserted if software attempts to set RXS=1 while the bit is already set.

TXS
Bit 3

CAN 1 Transmit Status. This bit indicates whether or not one or more messages have been successfully transmitted since the last read of the CAN 1 Status Register. TXS is only set by the CAN 1 logic and is not cleared by the CAN controller but is only cleared via software, the CRST bit, or a system Reset.

1 = A message has been successfully transmitted by CAN 1 (error free and acknowledged) since the last read of the CAN 1 Status Register.

0 = No messages have been successfully transmitted since the last read of the CAN 1 Status Register.

When STIE= 1 and the TXS bit transitions from 0 to 1, the CAN 1 Interrupt Register (C1IE;A5h) will change to 01h to indicate a pending interrupt due to a change in the CAN Status Register. Reading any bit in the C1S register will clear the pending interrupt, causing the C1IE register to change to 00h if no interrupts are pending or the appropriate value if a lower priority message center interrupt is pending. If a second successful reception is detected prior to or after the clearing of the RXS bit in the Status Register, a second status change interrupt flag will be set, issuing a second interrupt. Each new successful reception will generate an interrupt request independent of the previous state of the RXS bit, as long as the CAN Status Register has been read to clear the previous status change interrupt flag. Note that if software changes TXS from 0 to 1, an artificial Status Change Interrupt (STIE=1) will be generated. Thus, if TXS was previously set to 0 and a reception was successful, TXS will be set to 1 and an enabled interrupt may be asserted. An interrupt may be asserted (if enabled) if software changes TXS from 0 to 1. If TXS was previously set to 1 and a reception was successful, TXS remains set and an interrupt may be asserted if enabled. No interrupt will be asserted if software attempts to set TXS while it is already set.

ER2-0
Bit 2-0

CAN 1 Bus Error Status. These bits indicate the type of error, if any, detected in the last CAN 1 Bus Frame. These bits will be reset to the 111b state following any read of the C1S register (when SWINT=0), allowing software to determine if a new error has been received since the last read of this register. The ER2-0 bits are read only.

The ER2-0 bits are updated any time they change from 000b or 111b to another value. If enabled, an interrupt will be generated at this time. Errors received while the ER2-0 bits are in a non-000b or 111b state will be ignored, leaving ER2-0 unchanged and not generating enabled interrupts. This ensures that error conditions will not be lost/overwritten before software has a chance to read the C1S register. Once the C1S register is read and the ER2-0 bits return to 111b, new errors will be processed normally. In the case of simultaneous errors in multiple CAN 1 message centers, only the highest priority error is indicated.

ER2	ER1	ER0	Priority	Error Conditions
0	0	0	N/A	No Error in Last Frame
0	0	1	2	Bit Stuff Error
0	1	0	5	Format Error
0	1	1	4	Transmit Not Acknowledged Error
1	0	0	6(lowest)	Bit 1 Error
1	0	1	1(highest)	Bit 0 Error
1	1	0	3	CRC Error
1	1	1	N/A	No change since last C1S read

The following is a description of the different error types:

Bit Stuff Error: Occurs when the CAN controller detects more than 5 consecutive bits of an identical state are received in an incoming message.

Format Error: Generated when a received message has the wrong format.

Transmit Not Acknowledged Error: Indicates that a data request message was sent and the requested node did not acknowledged the message.

Bit 1 Error: Indicates that the CAN attempted to transmit a message and that when a recessive bit was transmitted, the CAN bus was found to have a dominant bit level. This error is not generated when the bit is a part of the arbitration field (identifier and remote retransmission request).

Bit 0 Error: Indicates that the CAN attempted to transmit a message and that when a dominant bit was transmitted, the CAN bus was found to have a recessive bit level. This error is not generated when the bit is a part of the arbitration field. The Bit 0 Error is set each time a recessive bit is received during the Busoff recovery period.

CRC Error: Generated whenever the calculated CRC of a received message does not match the CRC embedded in the message.

CAN 1 Interrupt Register (C1IR)

	7	6	5	4	3	2	1	0
SFR E5h								
	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

C1IR.7-0
Bit 7-5

CAN 1 Interrupt Indicator 7-0 This register indicates the status of the interrupt source associated with the CAN 1 module. Reading this register after the generation of a CAN 1 Interrupt will identify the interrupt source as shown in the table below. This register is cleared to 00h following a reset.

C1IR.7-0	Priority	Interrupt Source
00h	N/A	No Pending Interrupt
01h	1 (highest)	Change in the CAN 1 Status Register
02h	2	Message 15
03h	3	Message 1
04h	4	Message 2
05h	5	Message 3
06h	6	Message 4
07h	7	Message 5
08h	8	Message 6
09h	9	Message 7
0Ah	10	Message 8
0Bh	11	Message 9
0Ch	12	Message 10
0Dh	13	Message 11
0Eh	14	Message 12
0Fh	15	Message 13
10h	16 (lowest)	Message 14

The C1IR value will not change unless the previous interrupt source has been acknowledged and removed (i.e., software read of the C1S register or clearing of the appropriate INTRQ bit), even if the new interrupt has a higher priority. If two enabled interrupt sources become active simultaneously, the interrupt of higher priority will be reflected in the C1IR value.

The CAN 1 interrupt source into the interrupt logic is active whenever C1IR is not equal to 00h. Changes in the C1IR value from 00h to a non-zero state, indicate the first interrupt source detected by the CAN module following the non-active interrupt state. The C1IR interrupt values displayed in C1IR will remain in place until the respective interrupt source is removed, independent of other higher (or lower) priority interrupts that become active prior to clearing the currently displayed interrupt source.

When the current CAN interrupt source is cleared, C1IR will change to reflect the next active interrupt with the highest priority. The Status Change interrupt will be asserted if there has been a change in the CAN 1 Status Register (if enabled by the appropriate ERIE and/or STIE bit) and the CAN Status Interrupt state is set. A message center interrupt will be indicated if the

INTRQ bit in the respective CAN Message Control Register is set.

CAN 1 Transmit Error Register (C1TE)

	7	6	5	4	3	2	1	0
SFR E6h								
	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0

R=Unrestricted Read, *= Write only when SWINT=1 and BUSOFF=0, -n=Value after Reset

C1TE.7-0
Bits 7-0

CAN 1 Transmit Error Register. This register indicates the number of accumulated CAN 1 transmit errors. The CAN 1 module responds in different ways to varying number of errors as shown below.

This register can only be modified via software when SWINT=1 and BUSOFF=0. All software writes to this register simultaneously load the same value into the CAN 1 Transmit Error Register and the CAN 1 Receive Error Register. Writing 00h to this register will also clear the CAN 1 Error Count Exceeded bit, CECE (C1S.6). This register is cleared following all hardware Resets and software resets enabled via the CRST bit in the CAN 1 Control Register.

C1TE Value	CAN 1 State
Value < 96	Error active mode, CAN 1 Bus on (BUSOFF=0)
128 > Value ≥ 96	Error active mode, CAN 1 Bus on (BUSOFF=0), warning level
255 ≥ Value ≥ 128	Error passive mode, CAN 1 Bus on (BUSOFF=0)
Value > 255	CAN 1 Bus off (BUSOFF=1)

CAN 1 Receive Error Register (C1RE)

	7	6	5	4	3	2	1	0
SFR E7h								
	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0	R*-0

R=Unrestricted Read, *= Write only via C1TE register, -n=Value after Reset

C1RE.7-0
Bits 7-0

CAN 1 Receive Error Register. This register indicates the number of accumulated CAN 1 receive errors. All writes to the C1TE register are simultaneously loaded into this register. This register is cleared following all hardware Resets and software resets enabled via the CRST bit in the CAN 1 Control Register.

Extended Interrupt Enable (EIE)

	7	6	5	4	3	2	1	0
SFR E8h	CANBIE	C0IE	C1IE	EWDI	EX5	EX4	EX3	EX2
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n = Value after Reset

CANBIE
Bit 7
CAN 0/1 Activity Interrupt Priority. This bit enables/disables the CAN 0/1 Activity Interrupt
0 = Disable the CAN 0/1 Activity Interrupt.
1 = Enable the CAN 0/1 Activity Interrupt.

C0IE
Bit 6
CAN 0 Interrupt Enable. This bit enables/disables the CAN 0 Interrupt
0 = Disable the CAN 0 Interrupt.
1 = Enable the CAN 0 Interrupt.

C1IE
Bit 5
CAN 1 Interrupt Enable. This bit enables/disables the CAN 1 Interrupt
0 = Disable the CAN 1 Interrupt.
1 = Enable the CAN 1 Interrupt.

EWDI
Bit 4
Watchdog Interrupt Enable. This bit enables/disables the watchdog interrupt.
0 = Disable the watchdog interrupt.
1 = Enable interrupt requests generated by the watchdog timer.

EX5
Bit 3
External Interrupt 5 Enable. This bit enables/disables external interrupt 5.
0 = Disable external interrupt 5.
1 = Enable interrupt requests generated by the $\overline{\text{INT5}}$ pin.

EX4
Bit 2
External Interrupt 4 Enable. This bit enables/disables external interrupt 4.
0 = Disable external interrupt 4.
1 = Enable interrupt requests generated by the INT4 pin.

EX3
Bit 1
External Interrupt 3 Enable. This bit enables/disables external interrupt 3.
0 = Disable external interrupt 3.
1 = Enable interrupt requests generated by the $\overline{\text{INT3}}$ pin.

EX2
Bit 0
External Interrupt 2 Enable. This bit enables/disables external interrupt 2.
0 = Disable external interrupt 2.
1 = Enable interrupt requests generated by the INT2 pin.

MOVX Extended Address Register (MXAX)

	7	6	5	4	3	2	1	0
SFR EAh								
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

Bits 7-0

MOVX Extended Address Register. This register is concatenated with P2 and R1 or R0 to form the 22-bit address when executing a MOVX @Ri, A or MOVX A, @Ri instruction in either the 22-bit paged or 22-bit contiguous modes. The DPTR related MOVX instructions do not utilize the P2 and MXAX register. Note that the MXAX register is only used when the processor is operating in either the paged or contiguous addressing modes.

CAN 1 Message Center 1 Control Register (C1M1C)

	7	6	5	4	3	2	1	0
SFR EBh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

MSRDY
Bit 7

CAN 1 Message Center 1 Ready. This bit is used by the Microcontroller to prevent the CAN module from accessing message center 1 while the microcontroller is updating message attributes. These include as identifiers (arbitration registers 0-3), data byte registers 0-7, data byte count (DTBYC3-DTBYC1), direction control ($T/\overline{R}$), the extended or standard mode bit (EX/ST), and the mask enables (MEME and MDME) associated with this message center. When this bit is 0, the CAN 1 processor will ignore this message center for transmit, receive, or remote frame request operations.

MSRDY is cleared following a microcontroller hardware reset or a reset generated by the CRST bit in the CAN 1 Control Register, and must also remain in a cleared mode until all the CAN 1 initialization has been completed. Individual message MSRDY controls can be changed after initialization to reconfigure specific messages, without interrupting the communication of other messages on the CAN 1 Bus.

ETI
Bit 6

CAN 1 Message Center 1 Enable Transmit Interrupt. Setting ETI to a 1 will enable a successful CAN 1 transmission in message center 1 to set the INTRQ bit for this message center which in turn will issue an interrupt to the microcontroller. When ETI is cleared to 0 a successful transmission will not set INTRQ bit and will not generate an interrupt. Note that the ETI bit located in Message Center 15 is ignored by the CAN module, since the message center 15 is a receive only message center.

ERI
Bit 5

CAN 1 Message Center 1 Enable Receive Interrupt. Setting ERI to a 1 will enable a successful CAN 1 reception and storage in message center 1 to set the INTRQ bit for this message center which in turn will issue an interrupt to the microcontroller. When ERI is cleared to 0 a successful reception will not set the INTRQ bit and as such will not generate an interrupt.

INTRQ
Bit 4

CAN 1 Message Center 1 Interrupt Request. This bit serves as a CAN interrupt flag, indicating the successful transmission or reception of a message in this message center. INTRQ is automatically set when ERI=1 and message center 1 successfully receives and stores a message. The INTRQ bit is also set to a 1 when ETI is set and the CAN 1 logic completes a successful transmission. The INTRQ interrupt request must be also enabled via the EA global mask in the IE SFR register if the interrupt is to be acknowledged by the microcontroller interrupt logic. This flag must be cleared via software.

EXTRQ
Bit 3

CAN 1 Message Center 1 External Transmit Request. When EXTRQ is cleared to a 0, there are no pending requests by external CAN nodes for this message. When EXTRQ is set to a 1, a request has been made for this message by an external CAN node, but the CAN 1 controller has not yet

completed the service request. Following the completion of a requested transmission by a message center programmed for transmission ($T/\overline{R}$ = 1), the EXTRQ bit will be cleared by the CAN 1 controller. A remote request is only answered by a message center programmed for transmission ($T/\overline{R}$ = 1) when DTUP = 1 and TIH = 0, i.e. when new data was loaded and is not being currently modified by the micro. Note that a message center programmed for a receive mode ($T/\overline{R}$ = 0) will also detect a remote frame request and will set the EXTRQ bit in a similar manner, but will not automatically transmit a data frame and as such will not automatically clear the EXTRQ bit.

MTRQ
Bit 2

CAN 1 Message Center 1 Microcontroller Transmit Request. When set, this bit indicates that the message center is requesting that a message be transmitted. The bit is cleared when the transmission is complete, allowing this bit to be used to both initiate and monitor the progress of the transmission. The bit can be set via software or the CAN module, depending on the state of the Transmit/Receive bit in the CAN 1 Message 1 Format Register (located in MOVX space). This bit is cleared when the CRST bit is set, the CAN module experiences a system reset, or the conditions described below. Note that the MTRQ bit located in Message Center 15 is ignored by the CAN module, since the Message Center 15 is a receive only message center.

$T/\overline{R}$ =0 (receive)
When software sets this bit, a remote frame request previously loaded into the message center will be transmitted. The CAN 1 Module will clear this bit following the successful transmission of the frame request message.

$T/\overline{R}$ =1 (transmit)

When software sets this bit, a data frame previously loaded into the message center will be transmitted. When $T/\overline{R}$ = 1, the MTRQ bit will also be set by the CAN 1 controller at the same time that the EXTRQ bit is set by a message request from an external node.

ROW/TIH
Bit 1

CAN 1 Message Center 1 Receive Overwrite/Transmit Inhibit. The Receive Overwrite (ROW) and Transmit Inhibit (TIH) bits share the same bit location. When $T/\overline{R}$ = 0 the bit has the ROW function, serving as a flag that an overwrite of incoming data may have occurred. When $T/\overline{R}$ = 1 the bit has the Transmit Inhibit function, allowing software to disable the transmission of a message while the data contents are being updated.

Receive Overwrite: ($T/\overline{R}$ = 0, ROW is Read Only)
The CAN 1 controller automatically sets this bit 0 if a new message is received and stored while the DTUP bit was still set. When set, ROW indicates that the previous message was potentially lost and may not have been read, since the microcontroller had not cleared the DTUP bit prior to the new load. When ROW = 0, no new message has been received and stored while DTUP was set to '1' since this bit was last cleared. Note that the ROW bit will not be set when the WTOE bit is cleared to a 0, since all overwrites are disabled. This is due to the fact that even if the incoming message matches the respective message center that as long as DTUP = 1 in the respective message center, the combination of WTOE

= 0 and DTUP = 1 will force the CAN module to ignore the respective message center when the CAN is processing the incoming data.

ROW is cleared by the CAN module when the microcontroller clears the DTUP bit associated with the same message center. INTRQ is automatically set when the ERI=1 and message center 1 successfully receives and stores a message.

ROW will reflect the actual message center relationships for message centers 1 to 14. Message center 15 utilizes a special shadow message buffer, and the ROW bit for that message center indicates an overwrite of the buffer as opposed to the actual message center 15. The ROW bit for message center 15 is cleared once the shadow buffer is loaded into the message center 15, and the shadow buffer is cleared to allow a new message to be loaded. The shadow buffer is automatically loaded into message center 15 when the microcontroller clears the DTUP and EXTRQ bits in message center 15.

Transmit Inhibit: (T/$\overline{R}$ = 1, TIH is unrestricted Read/Write)

The TIH allows the microcontroller to disable the transmission of the message when the data contents of the message are being updated. TIH = 1 directs the CAN 1 controller not to transmit the associated message. TIH = 0 enables the CAN 1 controller to transmit the message. If TIH = 1 when a remote frame request is received by the message center, EXTRQ will be set to a 1. Following the Remote Frame Request and after the microcontroller has established the proper data to be sent, the microcontroller will clear the TIH bit to a 0, which will allow the CAN module to send the data requested by the previous Remote Frame Request. Note that the TIH bit associated with Message Center 15 is ignored because it is a receive only message center.

DTUP
Bit 0

CAN 1 Message Center 1 Data Updated. This bit indicates that new data has been loaded into the data portion of the message center. The exact function of the DTUP bit is dependent on whether the message center is configured in a receive (T/$\overline{R}$ = 0) or transmit (T/$\overline{R}$ = 1) mode. Some functions are also dependent on the state of the WTOE bit. The DTUP bit is only cleared by a software write to the bit, a system reset, or the setting of the CRST bit.

T/$\overline{R}$ =0 (receive)

In this mode (T/$\overline{R}$ = 0) the DTUP bit is set when new data has been successfully received and is ready to be read by the microcontroller. The exact meaning of the DTUP bit during a message center read is determined by the WTOE bit in the CAN 1 Control Register.

If WTOE = 1 (message center overwrite enabled), DTUP should be polled before and after reading the message center to ascertain if an overwrite of the data occurred during the read. For example, software should clear DTUP before reading the message center and then again after the message center read. If DTUP has been set, then a new message was received and software should read the message center again to read the new data. If DTUP remained cleared, no additional data was received and the data is complete.

If WTOE=0 the processor is not permitted to overwrite this message center, so it is only necessary to clear the DTUP bit after reading the message center.

The state of the DTUP bit in the receive mode does not inhibit remote frame request transmission in the receive mode. The only gating item for remote frame transmission in the receive mode is that the MSRDY and MTRQ bits must both be set.

T/$\overline{\mathbf{R}}$ =1 (transmit)
In this mode, software must set TIH =1 and clear DTUP = 0 prior to doing an update of the associated message center. This prevents the CAN module from transmitting the data while the microcontroller is updating it. Once the microcontroller has finished configuring the message center, software must clear TIH = 0 and set MSRDY=MTRQ =DTUP =1, to enable the CAN module to transmit the data.

The CAN module will **not** clear the DTUP after the transmission, but the microcontroller can verify that the transmission has been completed, by checking the MTRQ bit, which will be cleared (MTRQ = 0) after the transmission has been successfully completed.

CAN 1 Message Center 2 Control Register (C1M2C)

	7	6	5	4	3	2	1	0
SFR ECh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M2C Bits 7-0 — Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 3 Control Register (C1M3C)

	7	6	5	4	3	2	1	0
SFR EDh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M3C Bits 7-0 — Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 4 Control Register (C1M4C)

	7	6	5	4	3	2	1	0
SFR EEh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M4C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 5 Control Register (C1M5C)

	7	6	5	4	3	2	1	0
SFR EFh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M5C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

B Register (B)

	7	6	5	4	3	2	1	0
SFR F0h	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n=Value after Reset

B.7-0 Bits 7-0: **B Register.** This register serves as a second accumulator for certain arithmetic operations. It is functionally identical to the B register found in the 80C32.

CAN 1 Message Center 6 Control Register (C1M6C)

	7	6	5	4	3	2	1	0
SFR F3h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M6C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 7 Control Register (C1M7C)

	7	6	5	4	3	2	1	0
SFR F4h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M7C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 8 Control Register (C1M8C)

	7	6	5	4	3	2	1	0
SFR F5h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M8C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 9 Control Register (C1M9C)

	7	6	5	4	3	2	1	0
SFR F6h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M9C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 10 Control Register (C1M10C)

	7	6	5	4	3	2	1	0
SFR F7h	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M10C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

Extended Interrupt Priority (EIP)

	7	6	5	4	3	2	1	0
SFR F8h	CANBIP	C0IP	C1IP	PWDI	PX5	PX4	PX3	PX2
	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

R=Unrestricted Read, W=Unrestricted Write, -n = Value after Reset

CANBIP
Bit 7

CAN 0/1 Activity Interrupt Priority. This bit controls the priority of the CAN 0/1 Activity Interrupt

0 = The CAN 0/1 Activity Interrupt is a low priority interrupt.

1 = The CAN 0/1 Activity Interrupt is a high priority interrupt.

C0IP
Bit 6

CAN 0 Interrupt Priority. This bit controls the priority of the CAN 0 Interrupt

0 = The CAN 0 Interrupt is a low priority interrupt.

1 = The CAN 0 Interrupt is a high priority interrupt.

C1IP
Bit 5

CAN 1 Interrupt Priority. This bit controls the priority of the CAN 1 Interrupt

0 = The CAN 1 Interrupt is a low priority interrupt.

1 = The CAN 1 Interrupt is a high priority interrupt.

PWDI
Bit 4

Interrupt Priority. This bit controls the priority of the watchdog interrupt.

0 = The watchdog interrupt is a low priority interrupt.

1 = The watchdog interrupt is a high priority interrupt.

PX5
Bit 3

External Interrupt 5 Priority. This bit controls the priority of external interrupt 5.

0 = External interrupt 5 is a low priority interrupt.

1 = External interrupt 5 is a high priority interrupt.

PX4
Bit 2

External Interrupt 4 Priority. This bit controls the priority of external interrupt 4.

0 = External interrupt 4 is a low priority interrupt.

1 = External interrupt 4 is a high priority interrupt.

PX3
Bit 1

External Interrupt 3 Priority. This bit controls the priority of external interrupt 3.

0 = External interrupt 3 is a low priority interrupt.

1 = External interrupt 3 is a high priority interrupt.

PX2
Bit 0

External Interrupt 2 Priority. This bit controls the priority of external interrupt 2.

0 = External interrupt 2 is a low priority interrupt.

1 = External interrupt 2 is a high priority interrupt.

CAN 1 Message Center 11 Control Register (C1M11C)

	7	6	5	4	3	2	1	0
SFR FBh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M11C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 12 Control Register (C1M12C)

	7	6	5	4	3	2	1	0
SFR FCh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M12C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 13 Control Register (C1M13C)

	7	6	5	4	3	2	1	0
SFR FDh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M13C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 14 Control Register (C1M14C)

	7	6	5	4	3	2	1	0
SFR FEh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M14C Bits 7-0: Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

CAN 1 Message Center 15 Control Register (C1M14C)

	7	6	5	4	3	2	1	0
SFR FFh	MSRDY	ETI	ERI	INTRQ	EXTRQ	MTRQ	ROW/TIH	DTUP
	RW-0	RW-0	RW-0	RW-0	RC-0	R*-0	R*-0	R*-0

R=Unrestricted Read, C=Clear Only, *= See description below, -n=Value after Reset

C1M15C Bits 7-0

Operation of the bits in this register are identical to those found in the CAN 1 Message One Control Register (C1M1C;ABh). Please consult the description of that register for more information.

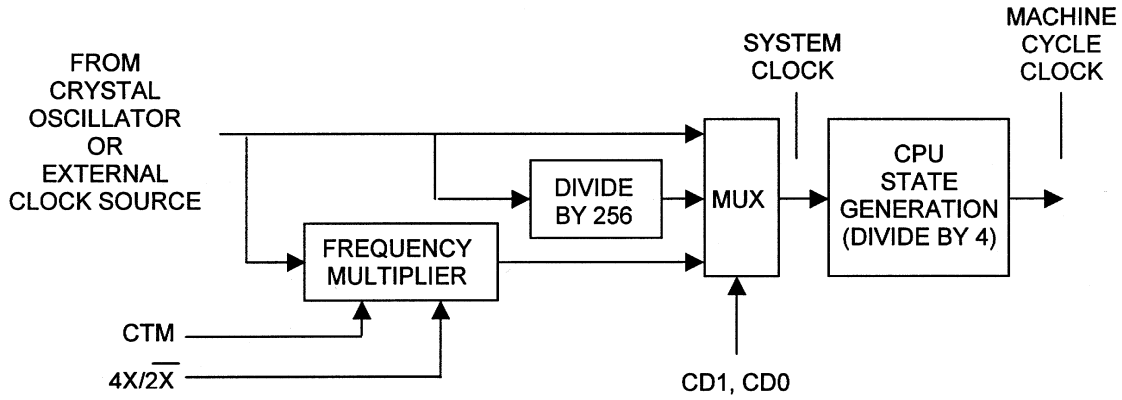
ADDENDUM TO SECTION 5: CPU TIMING

SYSTEM CLOCK SELECTION

The internal clocking options of the DS80C390 differs slightly from that described in the High-Speed Microcontroller User's Guide. Most members of the family offer the option of 4, 256, or 1024 clocks per machine cycle. The DS80C390 can operate at 1, 2, 4 or 1024 clocks per machine cycle. The logical operation of the system clock divide control function is shown below. A 3:1 multiplexer, controlled by CD1, CD0 (PMR.7-6), selects one of three sources for the internal system clock:

- Crystal oscillator or external clock source
- (Crystal oscillator or external clock source) divided by 256
- (Crystal oscillator or external clock source) frequency multiplied by 2 or 4 times.

Figure 5- 1 SYSTEM CLOCK CONTROL DIAGRAM



The system clock control circuitry generates two clock signals that are used by the microcontroller. The *internal system clock* provides the timebase for timers and internal peripherals. The system clock is run through a divide by 4 circuit to generate the *machine cycle clock* that provides the timebase for CPU operations. All instructions execute in one to five machine cycles. It is important to note the distinction between these two clock signals, as they are sometimes confused, creating errors in timing calculations.

Setting CD1, CD0 to 0 enables the frequency multiplier, either doubling or quadrupling the frequency of the crystal oscillator or external clock source. The $4X/\overline{2X}$ bit controls the multiplying factor, selecting twice or four times the frequency when set to 0 or 1, respectively. Enabling the frequency multiplier results in apparent instruction execution speeds of 2 or 1 clocks. Regardless of the configuration of the frequency multiplier, the system clock of the microcontroller can never be operated faster than 40 MHz. This means that the maximum crystal oscillator or external clock source is 10 MHz when using the 4X setting, and 20 MHz when using the 2X setting.

The primary advantage of the clock multiplier is that it allows the microcontroller to use slower crystals to achieve the same performance level. This reduces EMI and cost, as slower crystals are generally more available and thus less expensive.

SYSTEM CLOCK CONFIGURATION Table 5- 1

CD1	CD0	4X/$\overline{2X}$	Name	Clocks/MC	Max. External Frequency
0	0	0	Frequency Multiplier (2X)	2	20 MHz
0	0	1	Frequency Multiplier (4X)	1	10 MHz
0	1	N/A	Reserved		
1	0	N/A	Divide-by-four (Default)	4	40 MHz
1	1	N/A	Power Management Mode	1024	40 MHz

The system clock and machine cycle rate changes one machine cycle after the instruction changing the control bits. Note that the change will affect all aspects of system operation, including timers and baud rates. The use of the switchback feature, described later, can eliminate many of the issues associated with the Power Management Mode's affect on peripherals such as the serial port. Table 5-2 illustrates the effect of the clock modes on the operation of the timers.

EFFECT OF CLOCK MODES ON TIMER OPERATION Table 5- 2

CD1	CD0	4X/$\overline{2X}$	OSC. CYCLES PER MACHINE CYCLE	OSC. CYCLES PER TIMER 0/1/2 CLOCK		OSC. CYCLES PER TIMER 2 CLOCK, BAUD RATE GEN.		OSC. CYCLES PER SERIAL PORT CLOCK MODE 0		OSC. CYCLES PER SERIAL PORT CLOCK MODE 2	
				TxM=1	TxM=0	T2M=1	T2M=0	SM2=0	SM2=1	SMOD=0	SMOD=1
0	0	0	2	12	2	2	2	6	2	64	32
0	0	1	1	12	1	2	2	3	1	64	32
0	1	N/A	Reserved								
1	0	N/A	4	12	4	2	2	12	4	64	32
1	1	N/A	1024	3072	1024	512	512	3072	1024	16,384	8192

Changing the system clock/machine cycle clock frequency

The microcontroller incorporates a special locking sequence to ensure "glitch-free" switching of the internal clock signals. All changes to the CD1, CD0 bits must pass through the 10 (divide-by-four) state. For example, to change from 00 (frequency multiplier) to 11 (PMM), the software must change the bits in the following sequence: 00 10 11. Attempts to switch between invalid states will fail, leaving the CD1, CD0 bits unchanged.

The following sequence must be followed when switching to the frequency multiplier as the internal time source. This sequence can only be performed when the device is in divide-by-four operation. The steps must be followed in this order, although it is possible to have other instructions between them. Any deviation from this order will cause the CD1, CD0 bits to remain unchanged. Switching from frequency multiplier to non-multiplier mode requires no steps other than the changing of the CD1, CD0 bits.

1. Ensure that the CD1, CD0 bits are set to 10, and the RGMD (EXIF.2) bit = 0.
2. Clear the CTM (Crystal Multiplier Enable) bit.
3. Set the 4X/$\overline{2X}$ bit to the appropriate state.
4. Set the CTM (Crystal Multiplier Enable) bit.
5. Poll the CKRDY bit (EXIF.4), waiting until it is set to 1. This will take approximately 65536 cycles of the external crystal or clock source.
6. Set CD1, CD0 to 00. The frequency multiplier will be engaged on the machine cycle following the write to these bits.

ADDENDUM TO SECTION 6: MEMORY ACCESS

EXTERNAL MEMORY INTERFACING

The DS80C390 follows the memory interface convention established by the industry standard 80C32/80C52, but with many added improvements. Most notably, the device incorporates a 22-bit addressing capability that supports up to four megabytes of program memory and four megabytes of data memory. Externally the memory is accessed via a multiplexed or demultiplexed 20-bit address bus/8-bit data bus and four chip enable (active during program memory access) or four peripheral enable (active during data memory access) signals. Multiplexed addressing mode mimics the traditional 8051 memory interface, with the address MSB presented on Port 2 and the address LSB and data multiplexed on Port 0. The multiplexed mode requires an external latch to demultiplex the address LSB and data. When the MUX pin is pulled high, the address LSB and data are demultiplexed, with the address MSB presented on Port 2, address LSB on Port 1, and the data on Port 0. The elimination of the demultiplexing latch removes a delay element in the memory timing, and can in some cases allow the use of slower, less expensive memory devices. The following table illustrates the locations of the external memory control signals.

EXTERNAL MEMORY ADDRESSING PIN ASSIGNMENTS TABLE 6-1

Address/Data Bus	$\overline{CE3}$ - $\overline{CE0}$	$\overline{PCE3}$ - $\overline{PCE0}$	Addr 19-16	Addr 15-8	Addr 7-0	Data Bus
Multiplexed	P4.3-P4.0	P5.7-P5.4	P4.7-P4.4	P2	P0	P0
Demultiplexed	P4.3-P4.0	P5.7-P5.4	P4.7-P4.4	P2	P1	P0

Each upper order address line (A16-A19) and chip or peripheral enable is individually enabled via the P4CNT and P5CNT registers. Enabling upper order address lines controls the maximum size of the external memories that can be addressed, and enabling chip or peripheral enables controls the number of external memories that can be addressed. For example, if P4CNT.5-3 are set to 101b, A17 and A16 will be enabled (along with A15-0), permitting a maximum memory device size of 2^{18} or 256 KB.

The configurable program/code chip enable ($\overline{CEx}$) and MOVX chip enable ($\overline{PCEx}$) signals issued by the microprocessor are used when accessing multiple external memory devices. External chip enable lines are only required if more than one physical block of memory will be used. In the standard 8051 configuration, $\overline{PSEN}$ is used as the output enable for the program memory device, and $\overline{RD}$ and $\overline{WR}$ control the input or output functions of the data (SRAM) device. The chip enables of these devices can be tied to their active state if only one of each will be used. To support a larger amount of memory, however, the microprocessor must generate chip or data enables to select one of several memory devices. The following tables demonstrate how to enable various combinations of high-order address lines and chip enables.

EXTENDED ADDRESS AND CHIP ENABLE GENERATION TABLE 6-2

	Port 4 Pin Function (A19-A16 Address Pins)					Port 4 Pin Function (Code Memory Chip Enables)			
P4CNT.5-3	P4.7	P4.6	P4.5	P4.4	P4CNT.2-0	P4.3	P4.2	P4.1	P4.0
000	I/O	I/O	I/O	I/O	000	I/O	I/O	I/O	I/O
100	I/O	I/O	I/O	A16	100	I/O	I/O	I/O	$\overline{CE0}$
101	I/O	I/O	A17	A16	101	I/O	I/O	$\overline{CE1}$	$\overline{CE0}$
110	I/O	A18	A17	A16	110	I/O	$\overline{CE2}$	$\overline{CE1}$	$\overline{CE0}$

111(default)	A19	A18	A17	A16	111(default)	$\overline{CE3}$	$\overline{CE2}$	$\overline{CE1}$	$\overline{CE0}$

	Port 5 Pin Function (MOVX Memory Chip Enables)			
P5CNT.2-0	**P5.7**	**P5.6**	**P5.5**	**P5.4**
000(default)	I/O	I/O	I/O	I/O
100	I/O	I/O	I/O	$\overline{PCE0}$
101	I/O	I/O	$\overline{PCE1}$	$\overline{PCE0}$
110	I/O	$\overline{PCE2}$	$\overline{PCE1}$	$\overline{PCE0}$
111	$\overline{PCE3}$	$\overline{PCE2}$	$\overline{PCE1}$	$\overline{PCE0}$

The following table illustrates how memory is segmented based on the setting of the Port 4 P4.7-4 Configuration Control bits (P4CNT.5-3)

PROGRAM MEMORY CHIP ENABLE BOUNDARIES TABLE 6-3

P4CNT.5-3	$\overline{CE0}$	$\overline{CE1}$	$\overline{CE2}$	$\overline{CE3}$	Maximum Memory size per Chip Enable
000	0h-7FFFh	8000h-FFFFh	10000h-17FFFh	18000h-1FFFFh	32 kilobytes
100	0h-1FFFFh	20000h-3FFFFh	40000h-5FFFFh	60000h-7FFFFh	128 kilobytes
101	0h-3FFFFh	40000h-7FFFFh	80000h-BFFFFh	C0000h-FFFFFh	256 kilobytes
110	0h-7FFFFh	80000h-FFFFFh	100000h-17FFFFh	180000h-1FFFFFh	512 kilobytes
111(default)	0-FFFFFh	100000h-1FFFFFh	200000h-2FFFFFh	300000h-3FFFFFh	1 megabyte

Following any reset, the device defaults to 16-bit mode addressing. In 16-bit addressing mode the device will be configured with P4.7-P4.4 as address lines and P4.3-P4.0 configured as $\overline{CE3-0}$, with the first program fetch being performed from 00000h with $\overline{CE0}$ active (low).

USING THE COMBINED CHIP ENABLE SIGNALS

The DS80C390 incorporates a feature allowing $\overline{PCEx}$ and $\overline{CEx}$ signals to be combined. This is useful when incorporating modifiable code memory as part of a bootstrap loader or for in-system reprogrammability. Setting the one or more $\overline{PDCE3-0}$ bits (MCON.3-0) causes the corresponding chip enable signal to be asserted for both MOVC and MOVX operations. Write access to combined program and data memory blocks is controlled by the $\overline{WR}$ signal, and read access is controlled by the $\overline{PSEN}$ signal. This feature is especially useful if the design achieves in-system reprogrammability via external Flash memory, in which a single device is accessed via both MOVC instructions (program fetch) and MOVX write operations (updates to code memory).

IMPLEMENTING A BOOTLOADER USING INTERNAL SRAM

The internal 4 KB SRAM of the DS80C390 can be used to implement a bootloader function, allowing in-system reprogrammability. One of the difficulties of implementing a bootloader function with a Flash memory device is that the Flash programming algorithm will not allow instruction fetches (reads) from a device while it is being reprogrammed. The DS80C390 avoids this problem by placing the internal SRAM in the program/data configuration and loading it with a small bootstrap loader transferred from the external Flash memory. The bootloader software then runs out of internal SRAM while the external memory is being reprogrammed.

The following example demonstrates the implementation with a Flash memory. The internal SRAM is first configured as program/data memory, and a small bootloader routine is copied from the external Flash memory into the internal SRAM. The software then jumps to the internal SRAM and begins executing the bootloader program out of the internal SRAM. The bootloader software activates the combined program/chip enable function, if desired, as described above to allow simplified access to the Flash memory. The bootloader program then begins accepting bytes via the serial port or other external interface and copies them via MOVX instructions to the appropriate location in the external Flash until the new program is loaded. The final step is to jump to the starting location in the external Flash and begin execution of the new program. Soon after starting the new program software should disable the combined program/chip enable function and configure the 4 KB SRAM as desired.

The bootloading process is summarized below. Steps 1-4 are performed while executing code from the Flash device.

1. Set/clear CMA bit as desired.
2. Configure 4 KB SRAM as program/data. (IDM1:IDM0=11).
3. Copy user-supplied bootloader into 4KB SRAM.
4. LJMP to beginning of bootloader code.
5. In bootloader code, set PDCEx bits (MCON.3-0) to correspond to CEs controlling the Flash.
6. Bootloader code performs multiple MOVX operations to load new code into Flash.
7. When Flash load is complete, LJMP to starting location of new code. (This often 000000h but does not have to be.) At beginning of new code clear PDCEx bits and configure CMA, IDM1, IDM0 as desired.

ADDENDUM TO SECTION 7: POWER MANAGEMENT

The DS80C390 supports the general power management features of the DS87C520 described in the High-Speed Microcontroller. Exceptions are noted below.

POWER MANAGEMENT MODES

Power management mode 1 (PMM1) is not supported on the DS80C390.

SWITCHING BETWEEN CLOCK SOURCES

The ring oscillator on the DS80C390 is similar to that on the DS80C320. As such it does not support the "run from ring" feature which allows the microprocessor to use the ring oscillator as a clock source after the external crystal has stabilized (CKRY=1).

ADDENDUM TO SECTION 8: RESET CONDITIONS

This section supersedes the corresponding section in the High-Speed Microcontroller User's Guide.

The microprocessor provides several ways to place the CPU in a reset state. It also offers the means for software to determine the cause of a reset. The reset state of most processor bits is not dependent on the type of reset, but selected bits do depend on the reset source. The reset sources and the reset state are described below. The function of the $\overline{\text{RSTOL}}$ pin is also described in this section.

RESET SOURCES

The microprocessor has three ways of entering a reset state, described below. They are:

Power-on/Power Fail Reset
Watchdog Timer Reset
External Reset

Power-on/Fail Reset

The DS80C390 incorporates an internal voltage reference which holds the CPU in the power-on reset state while V_{CC} is below V_{RST}. Once V_{CC} has risen above V_{RST}, the microprocessor will restart the oscillation of the external crystal and count 65536 clock cycles. This helps the system maintain reliable operation by only permitting processor operation when voltage is in a known good state. The processor will then begin software execution at location 0000h.

The processor will exit the reset condition automatically once the above conditions are met. This happens automatically, needing no external components or action. Execution begins at the standard reset vector address of 0000h. Software can determine that a Power-on Reset has occurred using the Power-on Reset flag (POR). It is located at WDCON.6. Since all resets cause a vector to location 0000h, the POR flag allows software to acknowledge that power failure was the reason for a reset.

Software should clear the POR bit after reading it. When a reset occurs, software will be able to determine if a power cycle was the cause. In this way, processing may take a different course for each of the three resets if applicable. When power fails (drops below V_{RST}), the power monitor will invoke the reset state again. This reset condition will remain while power is below the threshold. When power returns above the reset threshold, a full power-on reset will be performed. Thus a brownout that causes V_{CC} to drop below V_{RST} appears the same as a power up.

Watchdog Timer Reset

The Watchdog Timer is a free running timer with a programmable interval. The Watchdog supervises CPU operation by requiring software to reset it before the time-out expires. If the timer is enabled and software fails to clear it before this interval expires, the CPU is placed into a reset state. The reset state will be maintained for two machine cycles. Once the reset is removed, the software will resume execution at 0000h.

The Watchdog Timer is fully described in Section 11. Software can determine that a Watchdog time-out was the reason for the reset by using the Watchdog Timer Reset flag (WTRF). WTRF is located at WDCON.2. Hardware will set this bit to a logic 1 when the Watchdog times out without being cleared by software if EWT=1. If a Watchdog Timer reset occurs, software should clear this flag manually. This allows software to detect the event if it occurs again.

External Reset

If the RST input is taken to a logic 1, the CPU will be forced into a reset state. This will not occur instantaneously, as the condition must be detected and then clocked into the microprocessor. It requires a minimum of two machine cycles to detect and invoke the reset state. Thus the reset is a synchronous operation and the crystal must be running to cause an external reset.

Once the reset state is invoked, it will be maintained as long as RST=1. When the RST is removed, the CPU will exit the reset state within two machine cycles and begin execution at address 0000h. All registers will default to their power-on reset state. There is no flag to indicate that an external reset was applied. However, since the other two sources have associated flags, the RST pin is the default source when neither POR or WTRF is set.

If a RST is applied while the processor is in the Stop mode, the scenario changes slightly. As mentioned above, the reset is synchronous and requires a clock to be running. Since the Stop mode stops all clocks, the RST will first cause the oscillator to begin running and force the program counter to 0000h. Rather than a two machine cycle delay as described above, the processor will apply the full power-on delay (65536 clocks) to allow the oscillator to stabilize.

RESET OUTPUTS

The microprocessor has one reset output, the $\overline{\text{RSTOL}}$ pin.

Reset Output Low ($\overline{\text{RSTOL}}$)

This external output pin is active low whenever the microprocessor is in a reset state. It can be used to signal to external devices than an otherwise invisible internal reset is in progress. It will be active under the following conditions:

When the processor has entered reset via the RST pin
During the crystal warm-up period following a power-on reset or stop mode
During a watchdog timer reset. ($\overline{\text{RSTOL}}$ will be active for 2 machine cycles)
During an oscillator failure (OFDE=1).

RESET STATE

Regardless of the source of the reset, the state of the microprocessor is the same while in reset. When in reset, the oscillator is running, but no program execution is allowed. When the reset source is external, the user must remove the reset stimulus. When power is applied to the device, the power-on delay removes the stimulus automatically.

Resets do not affect the Scratchpad RAM. Thus any data stored in RAM will be preserved. The contents of internal MOVX data memory will also remain unaffected by a reset. Note that if the power supply dips below approximately 2V, the RAM contents may be lost. The minimum voltage required for RAM data retention in not specified. Since it is impossible to determine if the power was lower than 2V prior to the power–on reset, RAM must be assumed lost when POR is set.

The reset state of SFR bits are described in Section 4. Bits which are marked SPECIAL have conditions which can affect their reset state. Consult the individual bit descriptions for more information. Note that the stack pointer will also be reset. Thus the stack is effectively lost during a reset even though the RAM contents are not altered. Interrupts and Timers are disabled. The state of the Watchdog Timer is dependent on the specific device in use. Note that the Watchdog time out defaults to its shortest interval on any reset. I/O Ports are taken to a weak high state (FFh). This leaves each port pin configured with the

data latch set to a 1. Ports do not go to the 1 state instantly when a reset is applied, but will be taken high within two machine cycles of asserting a reset. When the reset stimulus is removed, program execution begins at address 0000h.

IN-SYSTEM DISABLE MODE

The In-System Disable (ISD) feature allows the device to be tristated for in-circuit emulation or board testing. During ISD mode, the device pins will take on the following states:

DEVICE PIN	STATE DURING ISD
Port 0, 1, 2, 3, 4, 5, RST, EA, ALE, $\overline{\text{PSEN}}$, $\overline{\text{MUX}}$	True Tristate
$\overline{\text{RSTOL}}$	Driven per V_{OH3} specification
XTAL1, XTAL2	Oscillator remains active

The following procedure is used to enter ISD mode:

1. Assert reset by pulling RST high,
2. Pull ALE low and pull $\overline{\text{PSEN}}$ high,
3. Verify that P2.7, P2.6, P2.5 are not being driven low,
4. Release RST,
5. Hold ALE low and $\overline{\text{PSEN}}$ high for at least 2 machine cycles,
6. Device is now in ISD mode. Release ALE and $\overline{\text{PSEN}}$ if desired.

Note that pins P2.7, P2.6, P2.5 should not be driven low when RST is released. This will place the device into a reserved test mode. Because these pins have a weak pull-up during reset, they can be left floating. The test mode is only sampled on the falling edge of RST, and once RST is released their state will not effect device operation. In a similar manner, the $\overline{\text{PSEN}}$ and RST pins can be released once ISD mode is invoked, and their state will not effect device operation. The RST pin will also be in a tristate mode, but asserting it in ISD mode will return the device to normal operation.

ADDENDUM TO SECTION 10: PARALLEL I/O

Changes to this section primarily involve the additional functionality associated with Port 4 and 5, and the use of Port 1 as the address LSB in non-multiplexed memory mode. Because the DS80C390 is a ROMless device, Port 0 and 2 do not support general purpose I/O.

Port 1
General Purpose I/O

When the device is operating in multiplexed memory mode ($\overline{MUX}$ pin is tied to a logic low) port 1 serves as a general purpose I/O port. Data written to the port latch serves to set both level and direction of the data on the pin. More detail on the functions of port 1 pins configured for general purpose I/O is provided under the description of port 1 and port 3 in the High-Speed Microcontroller User's Guide.

Non-multiplexed Address Bus A0-A7

When the device is operating in non-multiplexed memory mode ($\overline{MUX}$ pin is tied to a logic high) port 1 serves LSB of the external address bus. When operating as the LSB of the address bus the port 1 pins have extremely strong drivers that allow the bus to move 100 pF loads with the timing shown in the electrical specifications.

When used as an address bus, the A0-7 pins will provide true drive capability for both logic levels. No pull-ups are needed. In fact, pull-ups will degrade the memory interface timing. Members of the High-Speed Microcontroller family employ a two-state drive system on A0-7. That is, the pin is driven hard for a period to allow the greatest possible setup or access time. Then the pin states are held in a weak latch until forced to the next state or overwritten by an external device. This assures a smooth transition between logic states and also allows a longer hold time. In general, the data is held (hold time) on A0-7 until another device overwrites the bus. This latch effect is generally transparent to the user.

Current-limited transitions

The DS80C390 does not employ the current-limited transition feature described in the High-Speed Microcontroller User's Guide.

Ports 4 and 5

Ports 4 and 5 are general purpose I/O ports with optional special functions associated with each pin. Enabling the special function automatically converts the I/O pin to that function. To insure proper operation, each alternate function pin should be programmed to a logic 1.

The drive characteristics of these pins may change depending on whether the pin is configured for general I/O or as the special function associated with that pin. When in I/O mode, the logic 0 is created by a strong pull-down. The logic 1 is created by a strong transition pull-up that changes to a weak pull-up. When a pin is configured in its alternate function, and that function concerns memory interfacing (A16-A17, $\overline{PCE0-3}$, or $\overline{CE0-3}$) the pins will be driven using the stronger memory interface values shown in the DC electrical characteristics of the data sheet.

OUTPUT FUNCTIONS

Although 8051 I/O ports appear to be true I/O, their output characteristics are dependent on the individual port and pin conditions. When software writes a logic 0 to the port for output, the port is pulled to ground. When software writes a logic 1 to the port for output, ports 1, 3, 4, or 5 will drive weak pull-ups (after the strong transition from 0 to 1). Port 0 will go tri-state. Thus as long as the port is not heavily loaded, true

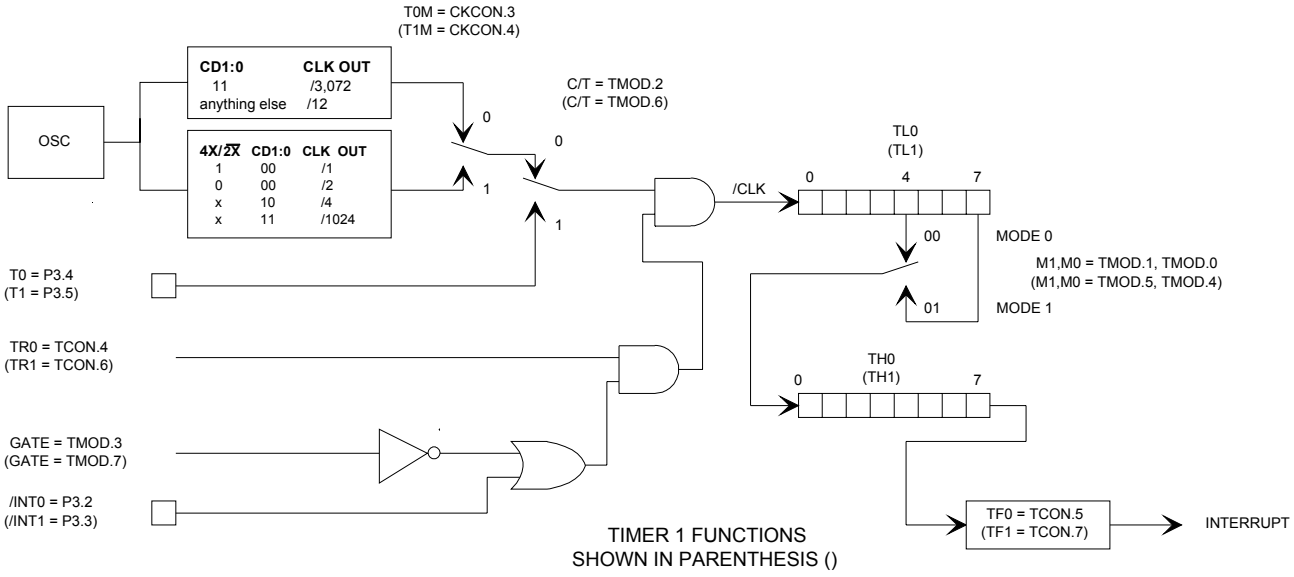
logic values will be output. DC drive capability is provided in the electrical specifications. Note that the DC current available from an I/O port pin is a function of the permissible voltage drop. Transition current is available to help move the port pin from a 0 to a 1. Since the logic 0 driver is strong, no additional drive current is needed in the 1 to 0 direction. The transition current is applied when the port latch is changed from a logic 0 to a logic 1. Simply writing a logic 1 where a 1 was already in place does not change the strength of the pull-up. This transition current is applied for a one half of a machine cycle. The absolute current is not guaranteed, but is approximately 2 mA at 5V.

When serving as an I/O port, the drive will vary as follows. For a logic 0, the port will invoke a strong pull-down. For a logic 1, the port will invoke a strong pull-up for two oscillator cycles to assist with the logic transition. Then, the port will revert to a weak pull-up. This weak pull-up will be maintained until the port transitions from a 1 to a 0. The weak pull-up can be overdriven by external circuits. This allows the output 1 state to serve as the input state as well.

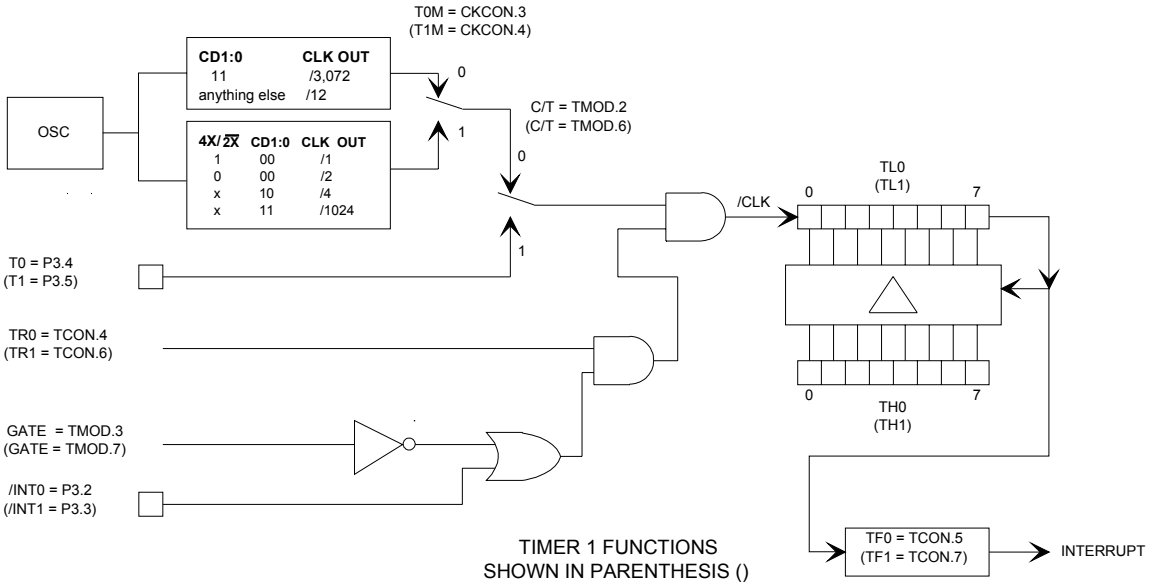
ADDENDUM TO SECTION 11: PROGRAMMABLE TIMERS

The timers of the DS80C390 are very similar to the those of described in the High-Speed Microcontroller User's Guide. The primary changes concern the removal of the PMM2 option and the inclusion of the frequency multiplier settings. The following figures replace the corresponding figures in Section 11 of the High-Speed Microcontroller User's Guide. The effect on the timers is summarized in tabular form in Section 5, EFFECT OF CLOCK MODES ON TIMER OPERATION Table 5- 2.

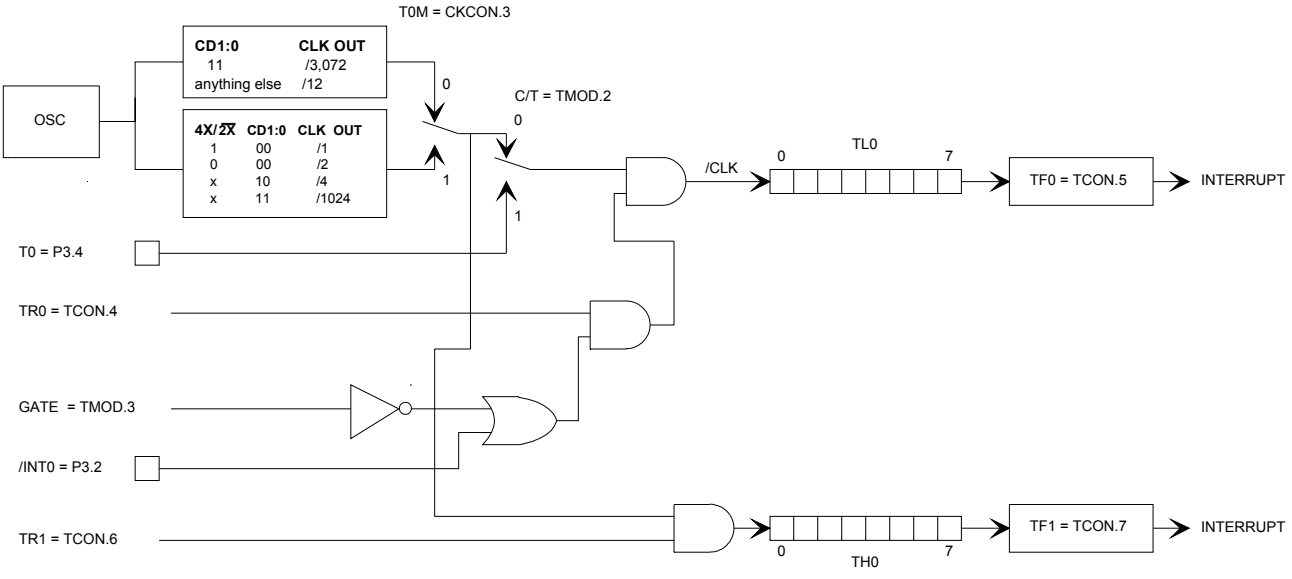
TIMER/COUNTER 0 AND 1
MODES 0 AND 1



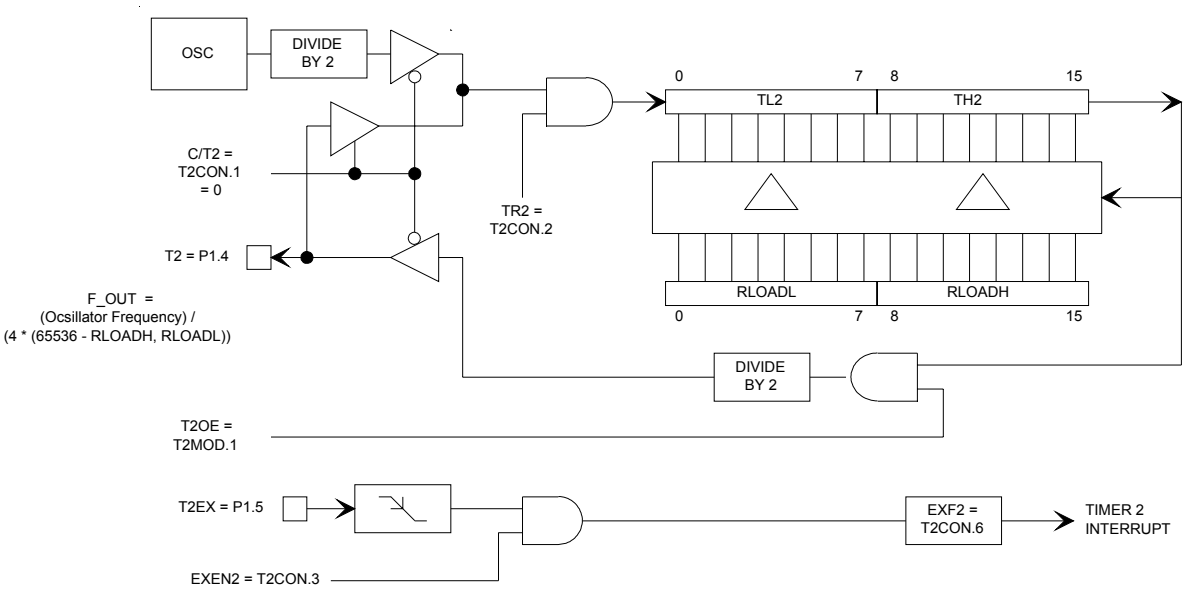
TIMER/COUNTER 0 AND 1
MODE 2



TIMER/COUNTER 0
MODE 3

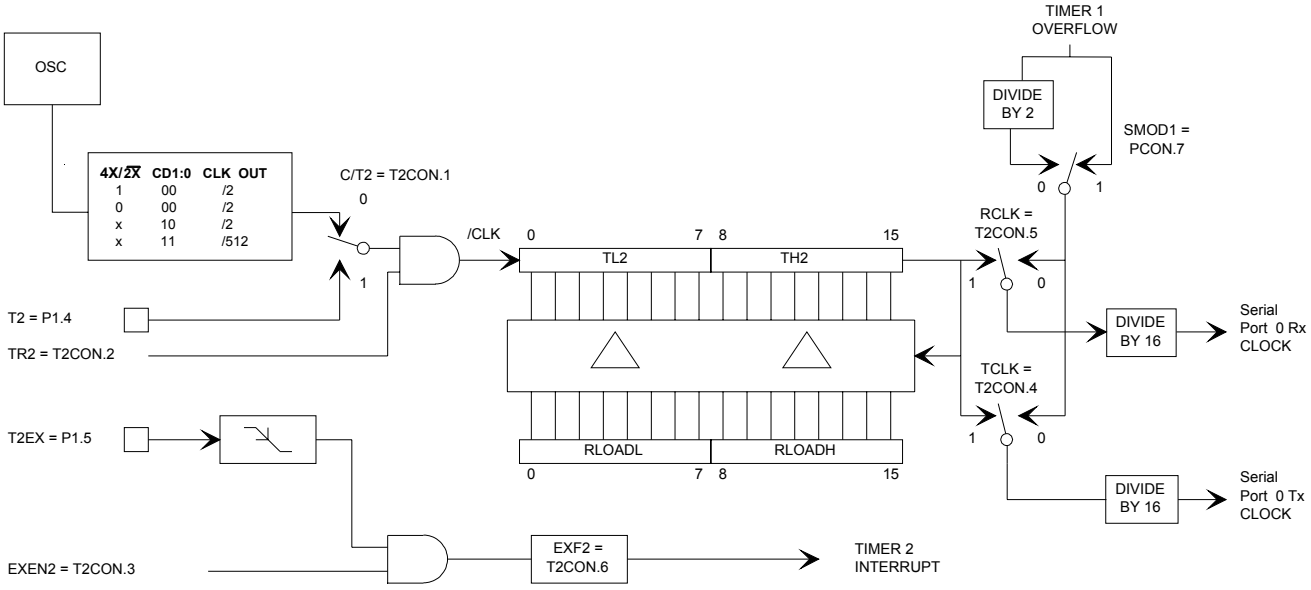


TIMER/COUNTER 2
CLOCK-OUT MODE
(/RL2 = 0)



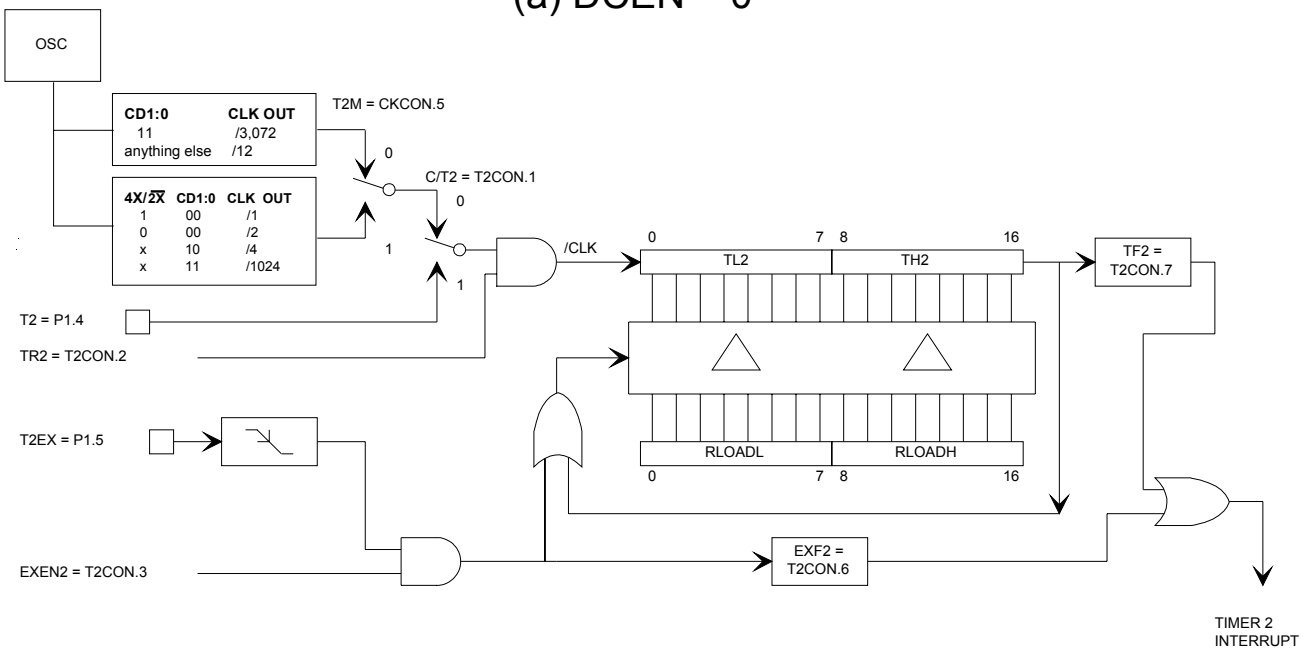
TIMER/COUNTER 2
BAUD RATE GENERATOR MODE

/RL2(T2CON.0) = 0; RCLK(T2CON.5) = 1 or TCLK(T2CON.4) = 1

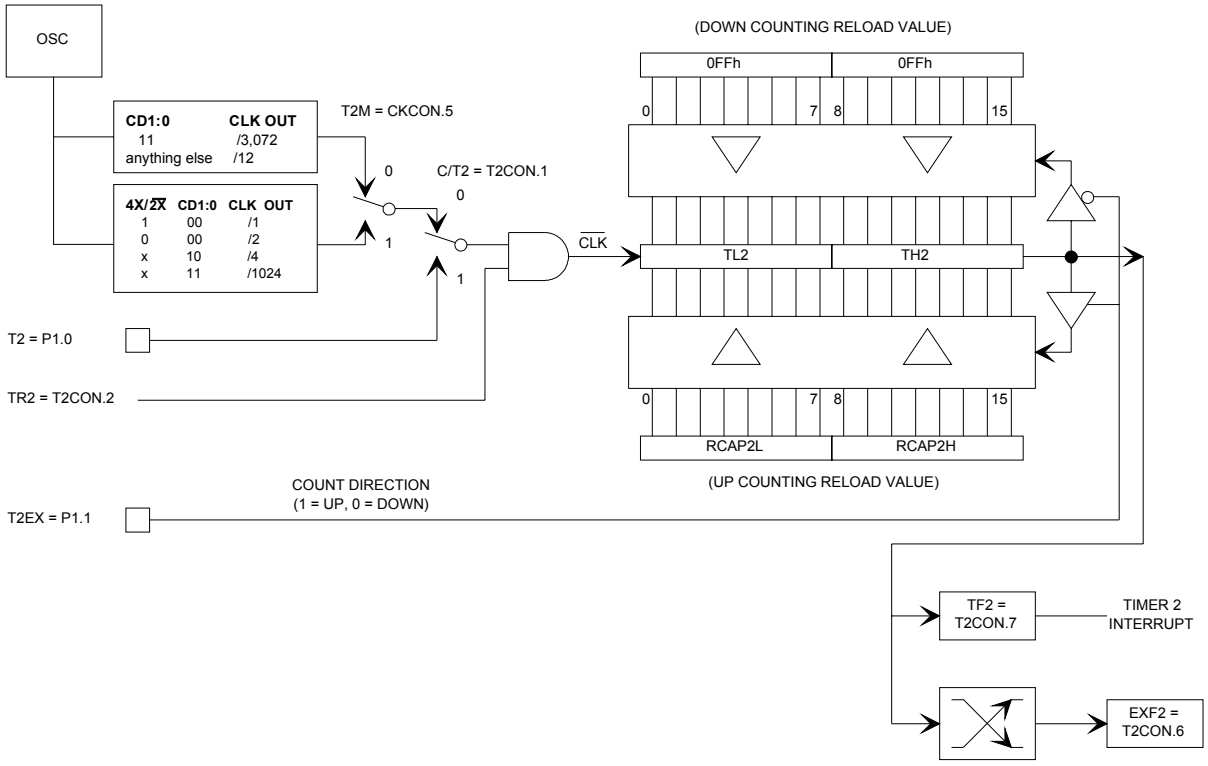


TIMER/COUNTER 2
AUTO RELOAD MODE (/RL2 = 0)

(a) DCEN = 0

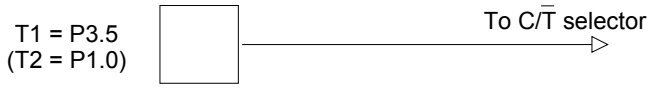


(b) DCEN = 1

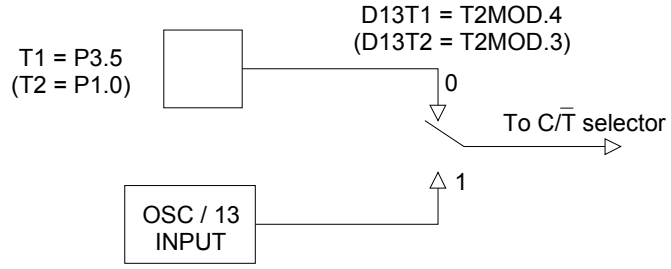


DIVIDE BY 13 OPTION

The other change to the timers associated with the DS80C390 is the inclusion of a divide by 13 option for Timer 1 and Timer 2. The option in independently enabled for each timer by setting the D13T1 (for timer 1) or D13T2 (for timer 2) bits. When enabled by setting the appropriate bits, the timer input from the T1 or T2 external pins will be replaced by a timebase that is OSC/13. The following figure illustrates the operation of these bits.



As shown in High-Speed Microcontroller User's Guide



As implemented in DS80C390 with divide by 13 option

The setting of the divide by 13 bits will affect all operations of timer 1 and all operations of timer except baud rate generator mode. The baud rate generator mode of Timer 2 will not be affected by any setting of the D13T2 bit.

PROGRAMMABLE CLOCK OUTPUT

When enabled, the DS80C390 can output a 50% duty cycle square wave on external pin P3.5. This signal is free-running, and not synchronized to the external clock source. To enable this feature, three conditions must be met:

1. Select the output frequency of system clock divided by 2, 4, 6, or 8 via the Clock Output Divide Select bits (COR.2-1).
2. The External Clock Output Enable bit, CLKOE, must be set (COR.0). Steps 1 and 2 can be combined and must use the Timed Access procedure.
3. The P3.5 latch bit (P3.5) must be set.

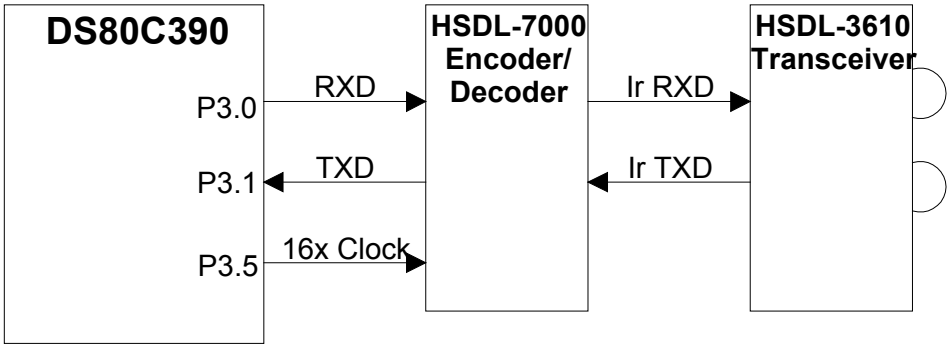
IRDA CLOCK OUTPUT

The Infrared Data Association (IrDA) communication protocol is a popular way to connect physically separated devices up to one meter distant. The physical layer of the protocol is very easy to implement: configure the DS80C390's serial port 0 by selecting crystal speed, baud rate, etc. The microcontroller is then connected to an external IR encoder/decoder which modulates the output of the serial port and communicates with an infrared transceiver. A good reference for the implementation of IrDA interfaces can be found in "IrDA Data Link Design Guide" available from Hewlett-Packard Company, http:/www.hp.com/go/ir.

The DS80C390 incorporates special circuitry that makes it a snap to add IR capability to your design. Most IR encoders require the controlling microprocessor to supply a 16x clock to perform the modulation. The DS80C390 can provide this special 16x clock to the encoder without requiring the use of a timer. After the serial port is configured, set the IRDACK (COR.7) and CLKOE (COR.0) bits using the Timed Access procedure. The P3.5 latch bit (P3.5) must also be set. At this point a clock signal with a frequency of 16 times the serial port 0 baud rate will be presented on P3.5.

The following diagram illustrates how to add IrDA capability to a DS80C390-based system. The receive and transmit signals of serial port 0 are connected directly to an encoder/decoder chip which in turn interfaces to an infrared transceiver. The External Clock Output pin (P3.5) is tied to the 16XCLK pin of the encoder/decoder to provide the modulation clock.

SAMPLE IrDA IMPLEMENTATION



ADDENDUM TO SECTION 13: TIMED ACCESS PROTECTION

A number of Timed Access protected bits are associated with the new features of the DS80C390. Please consult the High-Speed Microcontroller User's Guide for complete information on the use of the Timed Access feature.

POR (WDCON.6):	Power-on Reset
WDIF (WDCON.3):	Watchdog Interrupt Flag
EWT (WDCON.1):	Watchdog Timer Enable
RWT (WDCON.0):	Watchdog Reset
BGS (EXIF.0):	Bandgap Stop
SA (ACON.2):	Stack Address Mode
AM1-AM0 (ACON.1-ACON.0):	Address Mode Bit 1 and Bit 0
IDM1-IDM0 (MCON.7-MCON.6):	Internal Memory Configuration and Location Bits 1 and 0
CMA (MCON.5):	CAN Data Memory Assignment
PDCE3-PDCE.0 (MCON.3-MCON.0)	Program/Data Chip Enables
CRST (C0C.3):	CAN 0 Reset
CRST (C1C.3):	CAN 1 Reset
SBCAN (P4CNT.6):	Single Bus CAN
P4CNT.5-P4CNT.0:	Port 4 Pin Configuration Control Bits
P5CNT.2-P5CNT.0:	Port 5 Pin (P5.7-P5.5) Configuration Control Bits
IRDACK (COR.7):	IRDA Clock Output Enable
C1BPR7-C1BPR6 (COR.6-COR.5):	CAN 1 Baud Rate Pre-scale Bits
C0BPR7-C0BPR6 (COR.4-COR.3):	CAN 0 Baud Rate Pre-scale Bits
COD1-COD0 (COR.2-COR.1):	CAN Clock Output Divide Bit 1 and Bit 0
CLKOE (COR.0):	CAN Clock Output Enable

ADDENDUM TO SECTION 16: INSTRUCTION SET DETAILS

The DS80C390 supports one of three different address modes, selected via the AM1 and AM0 bits in the ACON register. The processor operates in either the traditional 16-bit address mode, 22-bit paged address mode or in a 22-bit contiguous address mode. When operating in the 16-bit addressing mode (AM1, AM0 = 00b), all instruction cycle timing and byte counts will be identical to the 8051 family. Use of the 24-bit paged address mode is binary code-compliant with the traditional (16-bit) 8051 compilers, but allows for up to 4M bytes of program and 4M bytes of data memory to be supported via a new Address Page SFR which supports an internal bank switch mechanism. The 22-bit contiguous mode requires a compiler that supports contiguous program flow over the entire 22-bit address range via the addition of an operand and/or cycles to eight basic instructions.

16-BIT (8051 STANDARD) ADDRESSING MODE

This addressing mode is identical to that used by the 8051 family and most members of the High-Speed Microcontroller. The microcontroller defaults to this mode following a reset. This mode can also be used to run code compiled or assembled for the 22-bit contiguous mode, as long as the following four instructions are not executed：

```
MOV DPTR, #data24,
ACALL addr19
LCALL addr24
LJMP addr24
```

These four branch instructions are the only instructions that will cause the compiler to generate additional operands relative to the 16-bit addressing mode. Note that the number of cycles per instruction may appear different from other instructions, but this is ignored by most assemblers or compilers and as such does not pose a problem with the binary output.

By selecting the 24-bit contiguous mode prior using any one of these four branch instructions, it is possible to run 24-bit contiguous compiled code in the default 16-bit address configuration. Once the AM0 and AM1 bits are set to the 24-bit contiguous address mode, the instructions seen above will execute properly. When the 24-bit paged address mode is selected, all instructions complied under the traditional 16-bit address mode will execute normally at any point in code.

22-BIT PAGED ADDRESSING MODE

The DS80C390 incorporates an internal 8-bit Address Page Register (AP),an 8-bit extended DPTR Register (DPX), an 8-bit extended DPTR1 Register (DPX1), and an 8-bit MOVX Extended Address Register as hardware support for 22-bit addressing in the paged address mode (AM1, AM0 = 01b). This mode has two differences in code execution from the traditional 16-bit mode:

1. The first difference is the additional of one machine cycle when executing the ACALL, LCALL, RET and RETI instructions as well as when the hardware processes an interrupt. This change should be transparent to most compilers, as the byte count remains identical for these instructions.

2. The second instruction involves register indirect MOVX instructions such as `MOVX @Ri, A` or `MOVX A, @Ri`. When in this mode, the MXAX register supplies the upper 8-bits of the 22-bit (or 23-bit if CMA=1 or IDM1=) address bits of the MOVX address. The complete address is formed by concatenating MXAX, P2, and R1 or R0 in this mode. The DPTR-related MOVX instructions do not utilize the P2 and MXAX register.

The DS80C390 supports interrupts from any location in the 22-bit address field. When an interrupt request is acknowledged, the current contents of the 22-bit Program Counter (PC) is pushed onto the stack, and the page value (00h) and the lower 16-bit address of the interrupt vector is then written to the PC before the execution of the LCALL. This means that all interrupt vectors are fetched from address 0000xxh, rather than the current page as defined by the AP register. The RETI instruction will pop the three address bytes from the stack, and will restore these bytes back to the PC at the conclusion of the interrupt service routine. Interrupt service routines that branch over page boundaries must save the current contents of AP before altering the AP register, as it is not automatically saved on the stack. This mechanism will support up to three levels of nesting for interrupts.

One extra machine cycle is required to handle the additional byte associated with the extension to 22-bit addressing. The storage of the 22-bit address during an interrupt, LCALL, or ACALL instruction also requires three bytes of stack memory as opposed to the traditional two bytes in the 16-bit address mode. In this mode, the third byte of the PC (PC[22:16]) is not incremented when the lower 16 bits in the lower two bytes of the PC (PC[15:0]) rolls over from FFFFh to 0000h. In the 22-bit paged address mode PC[22:16] functions only as a storage register which is loaded by the Address Page (AP) register whenever the processor executes either a LJMP, ACALL or LCALL instruction. PC[22:16] is stored and retrieved from the stack with the lower 16-bit of address in PC[15:0] when stack operation is required.

In paged address mode MOVX instructions which utilize the data pointers (such as `MOVX @DPTR, A`) will form the 22-bit data address by concatenating the contents of the currently selected extended DPTR register (DPX or DPX1) with the contents of the DPTR. The values in the DPX and DPX1 registers are not affected when the lower 16 bits of the selected DPTR overflows or underflowed.

To maintain compatibility with existing 8051 compilers, the `JMP @A+DPTR` or the `MOVC A, @A+DPTR` instructions are limited to the current 64 KB page as specified by the upper 7 bits of the current instruction execution address register. The contents of the DPX and DPX1 registers will not affect the operation of either instruction. Note that this differs slightly from the previous discussion of instructions which utilize the data pointer.

The modification of the instructions in the 22-bit page address mode is summarized in the following table.

	INSTRUCTION CODE											
MNEMONIC	**D_7**	**D_6**	**D_5**	**D_4**	**D_3**	**D_2**	**D_1**	**D_0**	**HEX**	**BYTE**	**CYCLE**	**EXPLANATION**
ACALL addr 11	a_{10}	a_9	a_8	1	0	0	0	1	Byte 1	2	4	$(PC_{15:0})=(PC_{15:0})+2$
	a_7	a_6	a_5	a_8	a_3	a_2	a_1	a_0	Byte 2			(SP) = (SP) + 1
												$((SP)) = (PC_{7:0})$
												(SP) = (SP) + 1
												$((SP)) = (PC_{15-8})$
												(SP) = (SP) + 1
												$((SP))=(PC_{23:16})$
												$(PC_{10:0})$=addr11
												$(PC_{23:16})=(AP_{7:0})$

LCALL addr 16	0 a_{15} a_7	0 a_{14} a_6	0 a_{13} a_5	1 a_{12} a_5	0 a_{11} a_3	0 a_{10} a_2	1 a_9 a_1	0 a_8 a_0	12 Byte 2 Byte 3	3	5	$(PC_{15:0})=(PC_{15:0})+3$ (SP) = (SP) + 1 ((SP)) = $(PC_{7:0})$ (SP) = (SP) + 1 ((SP)) = (PC_{15-8}) (SP) = (SP) + 1 ((SP))=$(PC_{23:16})$ (PC)=addr16 $(PC_{23:16})=(AP_{7:0})$
RET	0	0	1	0	0	0	1	0	22	1	5	$(PC_{23:16})$=((SP)) (SP)=(SP)-1 (PC_{15-8})=((SP)) (SP)=(SP)-1 $(PC_{7:0})$=((SP)) (SP)=(SP)-1
RETI	0	0	1	1	0	0	1	0	32	1	5	$(PC_{23:16})$=((SP)) (SP)=(SP)-1 (PC_{15-8})=((SP)) (SP)=(SP)-1 (PC_{7-0})=((SP)) (SP)=(SP)-1

22-BIT CONTIGUOUS ADDRESSING MODE

When the AM1 bit is set, the DS80C390 will operate in its 22-bit contiguous addressing mode. This addressing mode is supported by a full 22-bit Program Counter with eight modified instructions that operate over the full 22-bit address range. All modified branching instructions will automatically store and restore the entire contents of the 22-bit Program Counter. The 22-bit DPTR and DPTR1 registers will function identically to the Program Counter to allow access to the full 22- bit data memory range.

All the DS80C390 instruction opcodes retain binary compatibility to the 8051. Modified instructions are only different with respect to their cycle/byte/operand count and operate within a contiguous 24-bit address field. Note that all instructions which utilize the DPTR register now make use of a full 24-bit register (DPTR=DPX+DPH+DPL and DPTR1=DPX1+DPH1+DPL1). This mode of operation requires software tools (assembler or compiler) specifically designed to accept the modified length of the new instructions.

In addition, the 22-bit contiguous mode utilizes the MXAX register to supply the upper 8-bits of the 22-bit (or 23-bit if CMA=1 or IDM1=) address bits of the MOVX address during register indirect MOVX instructions such as MOVX @Ri, A or MOVX A, @Ri. The complete address is formed by concatenating MXAX, P2, and R1 or R0 in this mode. The DPTR-related MOVX instructions do not utilize the P2 and MXAX register.

The instructions modified to operate in the 24-bit address mode are summarized in the following table.

MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
ACALL addr 19	a_{18}	a_{17}	a_{16}	1	0	0	0	1	Byte 1	3	5	(PC)=(PC)+3 (SP)=(SP)+1 ((SP))=($PC_{7:0}$) (SP)=(SP)+1 ((SP))=($PC_{15:8}$) (SP)=(SP)+1 ((SP))=($PC_{23:16}$) ($PC_{18:0}$)=addr19
	a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	Byte 2			
	a_7	a_6	a_5	a_8	a_3	a_2	a_1	a_0	Byte 3			
AJMP addr 19	a_{18}	a_{17}	a_{16}	0	0	0	0	1	Byte 1	3	5	(PC)=(PC)+3 ($PC_{18:0}$)=addr19
	a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	Byte 2			
	a_7	a_6	a_5	a_8	a_3	a_2	a_1	a_0	Byte 3			
INC DPTR	1	0	1	0	0	0	1	1	A3	1	4	(DPTR)=(DPTR)+1 ($PC_{18:0}$)=addr19
LCALL addr24	0	0	0	1	0	0	1	0	12	4	6	(PC)=(PC)+4 (SP)=(SP)+1 ((SP))=($PC_{7:0}$) (SP)=(SP)+1 ((SP))=($PC_{15:8}$) (SP)=(SP)+1 ((SP))=($PC_{23:16}$) ($PC_{23:0}$)=addr24
	a_{23}	a_{22}	a_{21}	a_{20}	a_{19}	a_{18}	a_{17}	a_{16}	Byte 2			
	a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	Byte 3			
	a_7	a_6	a_5	a_5	a_3	a_2	a_1	a_0	Byte 4			
LJMP addr24	0	0	0	0	0	0	1	0	02	4	5	($PC_{23:0}$)=addr24
	a_{23}	a_{22}	a_{21}	a_{20}	a_{19}	a_{18}	a_{17}	a_{16}	Byte 2			
	a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	Byte 3			
	a_7	a_6	a_5	a_5	a_3	a_2	a_1	a_0	Byte 4			
MOV DPTR, #data24	1	0	0	1	0	0	0	0	90	4	3	(DPX)=#data23:9 (DPH)=#data15:8 (DPL)=#data7:0
	d_{23}	d_{22}	d_{21}	d_{20}	d_{19}	d_{18}	d_{17}	d_{16}	Byte 2			
	d_{15}	d_{14}	d_{13}	d_{12}	d_{11}	d_{10}	d_9	d_8	Byte 3			
	d_7	d_6	d_5	d_5	d_3	d_2	d_1	d_0	Byte 4			
RET	0	0	1	0	0	0	1	0	22	1	5	($PC_{23:16}$)=((SP)) (SP)=(SP)-1 (PC_{15-8})=((SP)) (SP)=(SP)-1 ($PC_{7:0}$)=((SP)) (SP)=(SP)-1
RETI	0	0	1	1	0	0	1	0	32	1	5	($PC_{23:16}$)=((SP)) (SP)=(SP)-1 (PC_{15-8})=((SP)) (SP)=(SP)-1 ($PC_{7:0}$)=((SP)) (SP)=(SP)-1

SECTION 18: CONTROLLER AREA NETWORK (CAN) MODULE

The DS80C390 incorporates two identical CAN controllers (CAN 0 and CAN 1). Each of these CAN units provide operating modes which are fully compliant with the CAN 2.0B specification. The microcontroller interface to the CAN controllers is broken into two groups of registers. To simplify the software associated with the operation of the CAN controllers, all of the global CAN status and controls as well as the individual message center control/status registers are located in the Special Function Register map. The remaining registers associated with the data identification, identification masks, format and data are located in the MOVX space. Each of the SFR and MOVX registers are configured as dual port memories to allow both the CAN controller and the microcontroller access to the required functions.

The basic functions covered by the CAN controllers include the use of 11-bit standard or 29-bit extended acceptance identifiers, as programmed by the microcontroller for each message center. Each CAN unit provides storage for up to 15 messages, with the standard 8 byte data field, in each message. Each of the first 14 message centers is programmable in either a transmit or receive mode. Message center 15 is designed as a receive only message center with a FIFO buffer to prevent the inadvertent loss of data when the microcontroller is busy and is not allowed time to retrieve the incoming message prior to the acceptance of a second message into message center 15. Message 15 also utilizes an independent set of mask registers and Identification registers, which are only applied once an incoming message has not been accepted by any of the first fourteen message centers. A second filter test is also supported for all message centers (1 - 15) to allow the CAN controller to use two separate 8 bit media masks and media arbitration fields to verify the contents of the first two byte of data of each incoming message, before accepting an incoming message. This feature allows the CAN unit to directly support the use of higher CAN protocols which make use of the first and/or second byte of data as a part of the acceptance layer for storing incoming messages. Each message center can also be programmed independently to perform testing of the incoming data with or without the use of the global masks.

Global controls and status registers in each CAN module allow the microcontroller to evaluate error messages, validate new data and the location of such data, establish the bus timing for the CAN Bus, establish the Identification mask bits, and verify the source of individual messages. In addition each message center is individually equipped with the necessary status and controls to establish directions, interrupt generation, identification mode (standard or extended), data field size, data status, automatic remote frame request and acknowledgment, and masked or non-masked identification acceptance testing. Utilizing the Single Bus CAN mode (SBCAN=1) ties the inputs and outputs of both CAN modules together, effectively creating a single CAN module with 30 message centers.

The priority order associated with the CAN module transmitting or receiving a message is determined by the inverse of the number of the message center, and is independent of the arbitration bits assigned to the message center. Thus message center 2 has a higher priority than message center 14. To avoid a priority inversion the CAN modules are configured to reload the transmit buffer with the message of the highest priority (lowest message center number) whenever an arbitration is lost or an error condition occurs.

The following tables illustrate the locations of the MOVX SRAM registers and bits used by the CAN controllers. Following the tables are descriptions of the function of the bits and registers.

MOVX MESSAGE CENTERS FOR CAN 0

CAN 0 CONTROL/STATUS/MASK REGISTERS

Register	7	6	5	4	3	2	1	0	MOVX Data Address[1]
C0MID0	MID07	MID06	MID05	MID04	MID03	MID02	MID01	MID00	xxxx00h
C0MA0	M0AA7	M0AA6	M0AA5	M0AA4	M0AA3	M0AA2	M0AA1	M0AA0	xxxx01h
C0MID1	MID17	MID16	MID15	MID14	MID13	MID12	MID11	MID10	xxxx02h
C0MA1	M1AA7	M1AA6	M1AA5	M1AA4	M1AA3	M1AA2	M1AA1	M1AA0	xxxx03h
C0BT0	SJW1	SJW0	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0	xxxx04h
C0BT1	SMP	TSEG26	TSEG25	TSEG24	TSEG13	TSEG12	TSEG11	TSEG10	xxxx05h
C0SGM0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxxx06h
C0SGM1	ID20	ID19	ID18	0	0	0	0	0	xxxx07h
C0EGM0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxxx08h
C0EGM1	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13	xxxx09h
C0EGM2	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	xxxx0Ah
C0EGM3	ID4	ID3	ID2	ID1	ID0	0	0	0	xxxx0Bh
C0M15M0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxxx0Ch
C0M15M1	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13	xxxx0Dh
C0M15M2	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	xxxx0Eh
C0M15M3	ID4	ID3	ID2	ID1	ID0	0	0	0	xxxx0Fh

CAN 0 MESSAGE CENTER 1

	Reserved								xxxx10h - 11h
C0M1AR0	CAN 0 MESSAGE 1 ARBITRATION REGISTER 0								xxxx12h
C0M1AR1	CAN 0 MESSAGE 1 ARBITRATION REGISTER 1								xxxx13h
C0M1AR2	CAN 0 MESSAGE 1 ARBITRATION REGISTER 2								xxxx14h
C0M1AR3	CAN 0 MESSAGE 1 ARBITRATION REGISTER 3							WTOE	xxxx15h
C0M1F	DTBYC3	DTBYC2	DTBYC1	DTBYC0	T/$\overline{R}$	EX/$\overline{ST}$	MEME	MDME	xxxx16h
C0M1D0-7	CAN 0 MESSAGE 1 DATA BYTES 0 - 7								xxxx17h - 1Eh
	Reserved								xxxx1Fh

CAN 0 MESSAGE CENTERS 2-14

	MESSAGE CENTER 2 REGISTERS (similar to Message Center 1)	xxxx20h - 2Fh
	MESSAGE CENTER 3 REGISTERS (similar to Message Center 1)	xxxx30h - 3Fh
	MESSAGE CENTER 4 REGISTERS (similar to Message Center 1)	xxxx40h - 4Fh
	MESSAGE CENTER 5 REGISTERS (similar to Message Center 1)	xxxx50h - 5Fh
	MESSAGE CENTER 6 REGISTERS (similar to Message Center 1)	xxxx60h - 6Fh
	MESSAGE CENTER 7 REGISTERS (similar to Message Center 1)	xxxx70h - 7Fh
	MESSAGE CENTER 8 REGISTERS (similar to Message Center 1)	xxxx80h - 8Fh
	MESSAGE CENTER 9 REGISTERS (similar to Message Center 1)	xxxx90h - 9Fh
	MESSAGE CENTER 10 REGISTERS (similar to Message Center 1)	xxxxA0h - AFh
	MESSAGE CENTER 11 REGISTERS (similar to Message Center 1)	xxxxB0h - BFh
	MESSAGE CENTER 12 REGISTERS (similar to Message Center 1)	xxxxC0h - CFh
	MESSAGE CENTER 13 REGISTERS (similar to Message Center 1)	xxxxD0h - DFh
	MESSAGE CENTER 14 REGISTERS (similar to Message Center 1)	xxxxE0h - EFh

CAN 0 MESSAGE CENTER 15

-	Reserved								xxxxF0h - F1h
C0M15AR0	CAN 0 MESSAGE 15 ARBITRATION REGISTER 0								xxxxF2h
C0M15AR1	CAN 0 MESSAGE 15 ARBITRATION REGISTER 1								xxxxF3h
C0M15AR2	CAN 0 MESSAGE 15 ARBITRATION REGISTER 2								xxxxF4h
C0M15AR3	CAN 0 MESSAGE 15 ARBITRATION REGISTER 3							WTOE	xxxxF5h
C0M15F	DTBYC3	DTBYC2	DTBYC1	DTBYC0	0	EX/$\overline{\text{ST}}$	MEME	MDME	xxxxF6h
C0M15D0-7	CAN 0 MESSAGE 15 DATA BYTE 0 - 7								xxxxF7h - FEh
	Reserved								xxxxFFh

Notes:
[1]The first two bytes of the CAN 0 MOVX memory address are dependent on the setting of the CMA bit (MCON.5) CMA=0, xxxx=00EE; CMA=1, xxxx=4010.

MOVX MESSAGE CENTERS FOR CAN 1

CAN 1 CONTROL/STATUS/MASK REGISTERS

Register	7	6	5	4	3	2	1	0	MOVX Data Address[1]
C1MID0	MID07	MID06	MID05	MID04	MID03	MID02	MID01	MID00	xxxx00h
C1MA0	M0AA7	M0AA6	M0AA5	M0AA4	M0AA3	M0AA2	M0AA1	M0AA0	xxxx01h
C1MID1	MID17	MID16	MID15	MID14	MID13	MID12	MID11	MID10	xxxx02h
C1MA1	M1AA7	M1AA6	M1AA5	M1AA4	M1AA3	M1AA2	M1AA1	M1AA0	xxxx03h
C1BT0	SJW1	SJW0	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0	xxxx04h
C1BT1	SMP	TSEG26	TSEG25	TSEG24	TSEG13	TSEG12	TSEG11	TSEG10	xxxx05h
C1SGM0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxxx06h
C1SGM1	ID20	ID19	ID18	0	0	0	0	0	xxxx07h
C1EGM0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxxx08h
C1EGM1	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13	xxxx09h
C1EGM2	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	xxxx0Ah
C1EGM3	ID4	ID3	ID2	ID1	ID0	0	0	0	xxxx0Bh
C1M15M0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxxx0Ch
C1M15M1	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13	xxxx0Dh
C1M15M2	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	xxxx0Eh
C1M15M3	ID4	ID3	ID2	ID1	ID0	0	0	0	xxxx0Fh

CAN 1 MESSAGE CENTER 1

	Reserved								xxxx10h - 11h
C1M1AR0	CAN 1 MESSAGE 1 ARBITRATION REGISTER 0								xxxx12h
C1M1AR1	CAN 1 MESSAGE 1 ARBITRATION REGISTER 1								xxxx13h
C1M1AR2	CAN 1 MESSAGE 1 ARBITRATION REGISTER 2								xxxx14h
C1M1AR3	CAN 1 MESSAGE 1 ARBITRATION REGISTER 3							WTOE	xxxx15h
C1M1F	DTBYC3	DTBYC2	DTBYC1	DTBYC0	T/$\overline{R}$	EX/$\overline{ST}$	MEME	MDME	xxxx16h
C1M1D0-7	CAN 1 MESSAGE 1 DATA BYTES 0 - 7								xxxx17h - 1Eh
	Reserved								xxxx1Fh

CAN 1 MESSAGE CENTERS 2-14

	MESSAGE CENTER 2 REGISTERS (similar to Message Center 1)	xxxx20h - 2Fh
	MESSAGE CENTER 3 REGISTERS (similar to Message Center 1)	xxxx30h - 3Fh
	MESSAGE CENTER 4 REGISTERS (similar to Message Center 1)	xxxx40h - 4Fh
	MESSAGE CENTER 5 REGISTERS (similar to Message Center 1)	xxxx50h - 5Fh
	MESSAGE CENTER 6 REGISTERS (similar to Message Center 1)	xxxx60h - 6Fh
	MESSAGE CENTER 7 REGISTERS (similar to Message Center 1)	xxxx70h - 7Fh
	MESSAGE CENTER 8 REGISTERS (similar to Message Center 1)	xxxx80h - 8Fh
	MESSAGE CENTER 9 REGISTERS (similar to Message Center 1)	xxxx90h - 9Fh
	MESSAGE CENTER 10 REGISTERS (similar to Message Center 1)	xxxxA0h - AFh
	MESSAGE CENTER 11 REGISTERS (similar to Message Center 1)	xxxxB0h - BFh
	MESSAGE CENTER 12 REGISTERS (similar to Message Center 1)	xxxxC0h - CFh
	MESSAGE CENTER 13 REGISTERS (similar to Message Center 1)	xxxxD0h - DFh
	MESSAGE CENTER 14 REGISTERS (similar to Message Center 1)	xxxxE0h - EFh

CAN 1 MESSAGE CENTER 15

-	Reserved								xxxxF0h - F1h
C1M15AR0	CAN 1 MESSAGE 15 ARBITRATION REGISTER 0								xxxxF2h
C1M15AR1	CAN 1 MESSAGE 15 ARBITRATION REGISTER 1								xxxxF3h
C1M15AR2	CAN 1 MESSAGE 15 ARBITRATION REGISTER 2								xxxxF4h
C1M15AR3	CAN 1 MESSAGE 15 ARBITRATION REGISTER 3							WTOE	xxxxF5h
C1M15F	DTBYC3	DTBYC2	DTBYC1	DTBYC0	0	EX/$\overline{\text{ST}}$	MEME	MDME	xxxxF6h
C1M15D0-C1M15D7	CAN 1 MESSAGE 15 DATA BYTE 0 - 7								xxxxF7h - FEh
	Reserved								xxxxFFh

Notes:
[1]The first two bytes of the CAN 1 MOVX memory address are dependent on the setting of the CMA bit (MCON.5) CMA=0, xxxx=00EF; CMA=1, xxxx=4011.

CAN MOVX Register Description

Most of the SRAM control registers, including the message centers proper, are mapped into a special location in the MOVX SRAM space. The specific location of the registers is a function of the module number (CAN 0 or CAN 1) and the CMA bit which controls whether the CAN SRAM begins at location 401xxxh or 00Exxxh.

The MOVX CAN Registers consist of a set of one Control/Status/Mask register and 15 message centers. Write access to the Control/Status/Mask registers is only possible when the SWINT bit is set to 1. All message centers for a given CAN module are identical with the exception of 15, which has some minor differences noted in the register descriptions. All of the CAN 1 registers are duplicates of the CAN 0 register set, differing only by address. To simplify the documentation, only one set of registers will be shown, with the following generic notation used for register names and addresses:

n CAN number (0 or 1)

xxxx First four hexadecimal digits of register address

CMA	CAN 0	CAN 1
0	00EE	00EF
1	4010	4011

y Address based on message center number

y	Message center number
1	1
2	2
.	.
A	10
F	15

CAN Media ID Mask Register 0 (CnMID0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx00h								

CAN Media ID Mask Register 1 (CnMID1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx02h								

CAN Media ID Mask Registers 1-0. These registers function as the mask when performing the Media Identification test. This register can only be modified during a software initialization (SWINT=1). If MDME=0, the Media Identification test will not be performed and the contents of these registers is ignored. If MDME=1, the CAN module will perform an additional qualifying test on Data Bytes 0 and 1 of the incoming message, regardless of the state of the EX/$\overline{ST}$ bit. Data byte 1 will be compared against CAN Media Byte Arbitration Register 1 utilizing CnMID1 as a mask, and Data byte 0 will be compared against CAN Media Byte Arbitration Register 0 utilizing CnMID0 as a mask. Any bit in the CnMID1, CnMID0 masks programmed to 0 will ignore the state of the corresponding Data Byte bit when performing the test. Any bit in the CnMID1, CnMID0 masks programmed to 1 will force the state of the corresponding Data Byte bit and CAN Media Byte Arbitration Registers 1 and 0 to match before considering the incoming message a match. Programming either Media ID Mask Register to 00h effectively disables the Media ID test for that byte. As such the CnMID1, CnMID0 masks act as a don't care following a system Reset.

CAN Media Arbitration Register 0 (CnMA0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx01h								

CAN Media Arbitration Register 1 (CnMA1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx03h								

CAN Media Arbitration Register 1-0. These registers function as the arbitration field when performing the Media Identification test. If MDME=0, the Media Identification test will not be performed and the contents of these registers is ignored. If MDME=1, the CAN module will perform an additional qualifying test on Data Bytes 0 and 1 of the incoming message, as mentioned in the description of the CAN Media ID Mask Registers. This register can only be modified during a software initialization (SWINT=1).

CAN Bus Timing Register 0 (CnBT0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx04h	SJW1	SJW0	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0

SJW1, SJW0
Bits 7-6

CAN Synchronization Jump Width Select. These bits specify the maximum number of time quanta (t_{qu}) cycles that a bit may be lengthened or shortened in one resynchronization to compensate for Phase Errors detected by the CAN controller when receiving data. These bits can only be modified during a software initialization (SWINT=1).

SJW1	SJW0	Synchronization Jump Width (Number in parenthesis is SJW value used in bit timing calculations)
0	0	1 t_{qu} (1)
0	1	2 t_{qu} (2)
1	0	3 t_{qu} (3)
1	1	4 t_{qu} (4)

BPR5 - BPR0
Bits 5-0

CAN Baud Rate Prescaler. The sixty four states defined by the binary combinations of the BPR5 - BPR0 bits determine the value of the prescaler, which in turn defines the cycle time associated with one time quanta. These bits can only be modified during a software initialization (SWINT=1).

BPR5	BPR4	BPR3	BPR2	BPR1	BPR0	Baud Rate Prescale Value (BRPV)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
.	.	.	.	.	.	.
.	.	.	.	.	.	.
1	1	1	1	1	0	63
1	1	1	1	1	1	64

CAN Bus Timing Register 1 (CnBT1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx05h	SMP	TSEG26	TSEG25	TSEG24	TSEG13	TSEG12	TSEG11	TSEG10

SMP
Bit 7

CAN Sampling Rate. The Sampling Rate (SMP) bit determines the number of samples to be taken during each receive bit time. Programming SMP = 0 will take only one sample during each bit time. Programming SMP = 1 will direct the CAN logic to take three samples during each bit time, and to use a majority voting circuit to determine the final bit value. When SMP is set to a 1, two additional t_{qu} clock cycles are be added to Time Segment One. SMP should not be set to one when the Baud Rate Prescale Value (BRPV) is less than 4. This bit can only be modified during a software initialization (SWINT=1).

TSEG26-24
Bits 6-4

CAN Time Segment 2 Select. The eight states defined by the TSEG26 - TSEG24 bits determine the number of clock cycles in the Phase Segment 2 portion of the nominal bit time, which occurs after the sample time. These bits can only be modified during a software initialization (SWINT=1).

TSEG26	TSEG25	TSEG24	Time Segment Two Length (Number in parenthesis is TS2_LEN value used in bit timing calculations)
0	0	0	Invalid
0	0	1	2 t_{qu} (2)
0	1	0	3 t_{qu} (3)
.	.	.	.
1	1	0	7 t_{qu} (7)
1	1	1	8 t_{qu} (8)

TSEG13-10
Bits 3-0

CAN Time Segment 1 Select. The sixteen states defined by the TSEG13 - TSEG10 bits determine the number of clock cycles in the Phase Segment 1 portion of the nominal bit time, which occurs before the sample time. These bits can only be modified during a software initialization (SWINT=1).

TSEG13	TSEG12	TSEG11	TSEG10	Time Segment One Length (Number in parenthesis is TS1_LEN value used in bit timing calculations)
0	0	0	0	Invalid
0	0	0	1	2 t_{qu} (2)
0	0	1	0	3 t_{qu} (3)
.	.	.	.	.
1	1	1	0	15 t_{qu} (15)
1	1	1	1	16 t_{qu} (16)

CAN Standard Global Mask Register 0 (CnSGM0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx06h	MASK28	MASK27	MASK26	MASK25	MASK24	MASK23	MASK22	MASK21

CAN Standard Global Mask Register 1 (CnSGM1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx07h	MASK20	MASK19	MASK18	0	0	0	0	0

CAN Standard Global Mask Registers 1-0. These registers function as the mask when performing the 11-bit global identification test on incoming messages for Message Centers 1-14. If MEME=0, the incoming message ID field must match the corresponding message center arbitration value exactly, effectively ignoring the contents of these registers. These registers are only used when performing the standard identification test, and their contents are ignored when EX/$\overline{ST}$ =1. These registers can only be modified during a software initialization (SWINT=1).

Any mask bit in the CnSGM1, CnSGM0 mask programmed to a 0 will create a don't care condition when the respective bit in the incoming message ID field is compared with the corresponding arbitration bits in Message Centers 1-14. Any bit in these masks programmed to a 1 will force the respective bit in the incoming message ID field to match identically with the corresponding arbitration bits in Message Centers 1-14, before said message will be loaded into Message Centers 1-14.

The five least significant bits in the CnSGM1 register are not used, and will not perform any comparison of these bit locations. A read of these bits will produce the last bit values written to these bit locations by the microcontroller or an indeterminate value following a power-up.

CAN Extended Global Mask Register 0 (CnEGM0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx08h	MASK28	MASK27	MASK26	MASK25	MASK24	MASK23	MASK22	MASK21

CAN Extended Global Mask Register 1 (CnEGM1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx09h	MASK20	MASK19	MASK18	MASK17	MASK16	MASK15	MASK14	MASK13

CAN Extended Global Mask Register 2 (CnEGM2)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx0Ah	MASK12	MASK11	MASK10	MASK9	MASK8	MASK7	MASK6	MASK5

CAN Extended Global Mask Register 3 (CnEGM3)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx0Bh	MASK4	MASK3	MASK2	MASK1	MASK0	0	0	0

CAN Extended Global Mask Registers 0-3. These registers function as the mask when performing the Extended Global Identification test (EX/$\overline{ST}$ =1) for Message Centers 1-14. When EX/$\overline{ST}$ =0 the contents of this register will be ignored. These registers can only be modified during a software initialization (SWINT=1).

When EX/$\overline{ST}$ =1, the 29-bits of the message ID will be compared against the 29-bits of the CAN Message Center y Arbitration Registers, using the 29 bits of the CAN Extended Global Mask Registers as a mask. Any bit in the Extended Global Mask Registers set to 0 will ignore the state of the corresponding bit in the incoming message ID field when performing the test. Any bit in the Extended Global Mask Registers set to 1 will force the state of the corresponding bit in the incoming message ID field and CAN message center arbitration Registers 0-3 to match before considering the incoming message a match.

The three least significant bits in the CnEGM3 are not used, and will not perform any comparison of these bit locations. A read of these bits will always return 0, and writes to these bits will be ignored.

Programming all Mask registers to 00h effectively disables the Global ID test for that message, accepting all messages. As such the Global mask registers act as a don't care following a system Reset.

CAN Message Center 15 Mask Register 0 (CnM15M0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx0Ch	MASK28	MASK27	MASK26	MASK25	MASK24	MASK23	MASK22	MASK21

CAN Message Center 15 Mask Register 1 (CnM15M1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx0Dh	MASK20	MASK19	MASK18	MASK17	MASK16	MASK15	MASK14	MASK13

CAN 0 Message Center 15 Mask Register 2 (CnM15M2)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx0Eh	MASK12	MASK11	MASK10	MASK9	MASK8	MASK7	MASK6	MASK5

CAN 0 Message Center 15 Mask Register 3 (CnM15M3)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxx0Fh	MASK4	MASK3	MASK2	MASK1	MASK0	0	0	0

MASK28-MASK0 **CAN Message Center 15 Mask Registers 0-3.** These registers function as the mask when performing the Extended Global Identification test (EX/$\overline{ST}$ =1) for Message Center 15 only. These registers can only be modified during a software initialization (SWINT=1).

When EX/$\overline{ST}$ =1, the 29 bits of the message ID will be compared against the 29 bits of the CAN Message Center 15 Arbitration Registers, using the 29 bits of the CAN Message Center 15 Mask Registers as a mask. When EX/$\overline{ST}$ =0, the 11 bits of the message ID will be compared against the most significant 11 bits of the CAN Message Center 15 Arbitration Registers, using the most significant 11 bits of the CAN Message Center 15 Mask Registers as a mask. Any bit in the CAN Message Center 15 Mask Registers set to 0 will ignore the state of the corresponding bit in the incoming message ID field when performing the test. Any bit in the CAN Message Center 15 Mask Registers set to 1 will force the state of the corresponding bit in the incoming message ID field and CAN message center arbitration Registers 0-3 to match before considering the incoming message a match.

The three least significant bits in the CnM15M3 register are not used, and will not perform any comparison of these bit locations. A read of these bits will always return 0, and writes to these bits will be ignored.

Programming all Mask registers to 00h effectively disables the Message Center 15 ID test, accepting all messages. As such the Message Center 15 mask registers act as a don't care following a system Reset.

CAN MESSAGE CENTER MOVX REGISTER DESCRIPTIONS

CAN Message Center y Arbitration Register 0 (CnMyAR0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy2h	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21

CAN Message Center y Arbitration Register 1 (CnMyAR1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy3h	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13

CAN Message Center y Arbitration Register 2 (CnMyAR2)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy4h	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5

CAN Message Center y Arbitration Register 3 (CnMyAR3)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy5h	ID4	ID3	ID2	ID1	ID0	0	0	WTOE

ID28-ID0

CAN Message Center y Arbitration Registers 0-3. These bits form the arbitration value/identification number for the message center y. When the message center is configured in a transmit mode, these registers will be the source of the 29-bit ID message field (when EX/$\overline{ST}$ =1) or the 11-bit ID message field (when EX/$\overline{ST}$ =0). When EX/$\overline{ST}$ =1, the 29 message ID bits will correspond to ID28-ID0 as shown above. When EX/$\overline{ST}$ =0, the message ID bits 10-0 correspond to ID28-18 in CnMyAR0 and CnMyAR1.

When configured in a receive mode, these registers serve as the arbitration value for message center y, against which incoming messages are compared to ascertain if they are valid for that message center. When EX/$\overline{ST}$ =1, all 29 bits of the arbitration are used, but when EX/$\overline{ST}$ =0, only the most significant 11 bits are used.

Bits 2-1 (CnMyAR3 only)

Reserved – Bits 2 and 1 of the CnMyAR3 register are not used in arbitration. A read of these bits will always return 0, and writes to these bits will be ignored.

WTOE
Bit 0
(CnMxAR3 only)

Write-over Enable. This bit controls the ability of a new message to overwrite an existing message in the corresponding message center in receive mode. The DTUP and EXTRQ bits for the message center in question must also be considered to determine the effect of this bit as shown below. The WTOE bit can only be programmed when the SWINT bit is set.

WTOE	DTUP	EXTRQ	Result when new message detected
0	0	0	There is currently no unread message or pending external frame request in the message center, so the matching message will be written to appropriate message center (1-15)
0	1	x	The message center (1-15) has an unread message or pending external frame request. The incoming matching message will be ignored and the message center remains unchanged. The CAN module will proceed to the next lower priority message center to evaluate the incoming message ID and arbitration bits and related masking operations. (No overwrite)
0	x	1	The message center (1-15) has an unread message or pending external frame request. The incoming matching message will be ignored and the message center remains unchanged. The CAN module will proceed to the next lower priority message center to evaluate the incoming message ID and arbitration bits and related masking operations. (No overwrite)
1	0	x	There is currently no unread message or pending external frame request in the message center, so the matching message will be written to appropriate message center (1-15)
1	1	x	The new matching message will be stored, overwriting the previously stored message. The ROW bit will be set to indicate the overwrite operation.

Special notes for message center 15

The ROW bit in message center 15 is associated with an overwrite of the shadow buffer for message center 15. The EXTRQ and DTUP bits are also shadow buffered to allow the buffered message and the message center 15 value to take on different relationships. The EXTRQ and DTUP values read by software are the current message center 15 values, rather than those of the shadow buffer as is the case with the ROW bit. The shadow buffer is automatically loaded into message center 15 when **both** the DTUP bit and EXTRQ bit are cleared. If either DTUP=1 or EXTRQ=1 when clearing the other, any message in the shadow buffer will not be transferred to the message 15 registers, and any incoming messages for message 15 will be stored in the shadow buffer if WTOE = 1, or will be lost if WTOE = 0.

Special notes concerning remote frames

For remote frames, which can be received by transmit message centers (1-14) in case of a matching identifier, WTOE and EXTRQ are evaluated. If ((WTOE = 1) OR (WTOE = 0 and EXTRQ = 1)), the respective transmit message center (1-14) arbitration bits can be overwritten.

CAN Message Center y Format Register (CnMyF)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy6h	DTBYC3	DTBYC2	DTBYC1	DTBYC0	T/$\overline{R}$	EX/$\overline{ST}$	MEME	MDME

DTBYC3-0
Bits 7-4

Data Byte Count. These bits indicate the number of bytes within the data field of the message. When performing a transmit, software sets the DTBYC bits to establish the number of bytes that are to be transmitted. When receiving a message, the DTBYC bits indicate the (binary) number of bytes of data in the incoming message; i.e., 0000b = 0 data bytes and 1000b = 8 data bytes.

T/$\overline{R}$
Bit 3

Transmit/Receive Select. This bit is programmed by the application software to indicate if the message is to be transmitted (T/$\overline{R}$ = 1) or received (T/$\overline{R}$ = 0). This bit can only be modified when MSRDY = 0.

This bit does not exist for Message Center 15 and will always return 0 when read from Message Center 15.

EX/$\overline{ST}$
Bit 2

Extended or Standard Identifier. This bit determines whether the respective message is to utilize the extended 29-bit Identification format (EX/$\overline{ST}$ = 1) or the standard 11-bit Identification format (EX/$\overline{ST}$ = 0). Message centers programmed for one format will only receive/send extended messages in that format and will ignore the alternate format. This bit can only be modified when MSRDY = 0.

MEME
Bit 1

Message Identification Mask Enable. The MEME bit enables (MEME = 1) or disables (MEME = 0) the use of the Message Identification Masking process, associated with the testing of the Identification field in the incoming message. This bit can only be modified when MSRDY = 0.

0 = The mask registers are ignored when evaluating the identification bits of the incoming message, and the identification bits of the incoming message and the message center arbitration bits must match exactly to allow receipt of the incoming message. This is equivalent to programming the mask with all zeros. An exact match is also required before a remote data request is allowed.

1 = The mask registers are enabled, comparing only those bits message identification and arbitration bits which correspond to a 1 in the mask register.

MDME
Bit 0

Media Identification Mask Enable. The MDME bit enables (MEME = 1) or disables (MEME = 0) the use of the first two bytes of the data field as a message qualifier. This bit can only be modified when MSRDY = 0.

0 = The first two bytes of the data field are ignored and not compared.

1 = The first two data bytes are masked by the respective Media Mask ID Register and then compared with the Media Arbitration Register Zero and One bytes. Only those bits in the first two data bytes and the arbitration registers corresponding to a 1 in the mask register are compared. When MDME=1 the test is also performed before a remote request of data from a remote node is accepted.

CAN Message Center y Data Byte 0 (CnMyD0)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy7h								

CAN Message Center y Data Byte 1 (CnMyD1)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy8h								

CAN Message Center y Data Byte 2 (CnMyD2)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxy9h								

CAN Message Center y Data Byte 3 (CnMyD3)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxyAh								

CAN Message Center y Data Byte 4 (CnMyD4)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxyBh								

CAN Message Center y Data Byte 5 (CnMyD5)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxyCh								

CAN Message Center y Data Byte 6 (C0MyD6)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxyDh								

CAN Message Center y Data Byte 7 (C0MyD7)

MOVX Address[1]	7	6	5	4	3	2	1	0
xxxxyEh								

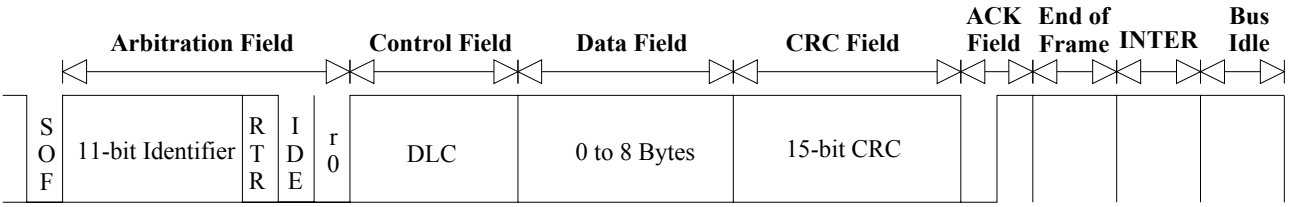
CnMyD0-CnMyD7	**CAN Message Center y Data Bytes 0-7.** These bytes hold data to be transmitted or received data.

Frame Types

The CAN 2.0B protocol specifies two different message formats, the standard 11-bit (CAN 2.0A) and the extended 29-bit (CAN 2.0 B), and four different Frame Types for CAN Bus communications.

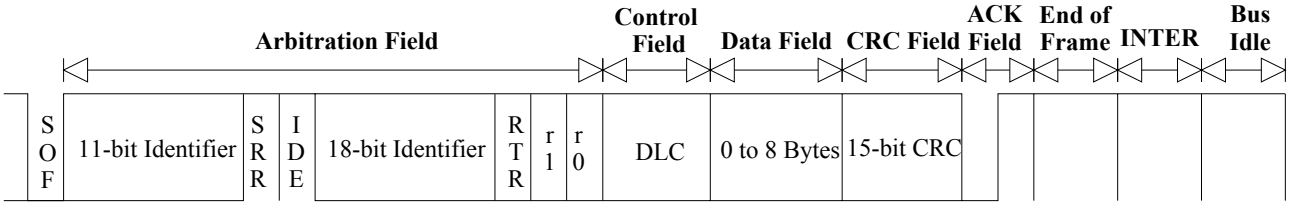
The Standard Format seen below makes use of an 11-bit identifier.

Figure 16- 1 CAN 2.0A Format



The Extended Format seen below makes use of a 29 bit identifier.

Figure 16- 2 CAN 2.0B Format



The four different Frame Types for CAN Bus communications are the Data Frame, the Remote Frame, the Error Frame and the Overload Frame.

Data Frame:

The Data Frame is formulated to carry data from a transmitter to a receiver. The preceding two figures are examples of data frames in the standard and extended formats. The Data Frame is composed of seven fields. These include the Start of Frame, Arbitration Field, Control Field, Data Field, CRC Field, Acknowledge Field and an End of Frame. A description of these fields follows.

Start of Frame - SOF: (Standard and Extended Format)

The Start of Frame is a dominant bit which signals the start of a Data or Remote Frame. The dominant forces a hard synchronization, initiating the CAN controller receive mode.

Arbitration Field: (Standard and Extended Format)

The Arbitration Field contains the identifier of the message and a dominant Remote Request (RTR) bit. The identifier is composed of one field in the standard 11-bit format or two fields in the extended 29-bit format. Two additional bits, the Substitution Remote Request (SRR) bit and the Identifier Extension (IDE) bit, separate the two fields in the Extended Format.

Remote Request (RTR) bit: (Standard and Extended Format)

The Remote Request bit is a dominant bit in Data Frames and a recessive bit in Remote Frames.

Substitution Remote Request (SRR) bit: (Extended Format)

The Substitution Remote Request bit is a recessive bit and is substituted for the RTR bit when using the Extended Format.

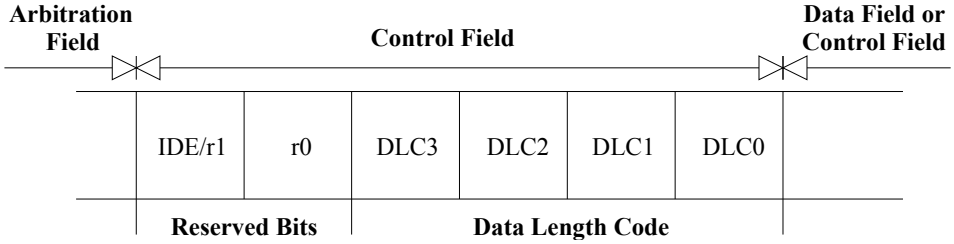
Identifier Extension (IDE) bit: (Extended Format)

The Identifier Extension (IDE) bit is a dominant bit in the Standard Format and a recessive bit in the Extended Format. The IDE bit is located in the Arbitration Field in the Standard Format and is located in the Control Field in the Extended Format.

Control Field: (Standard and Extended Format)

The Control Field is made up of six bits in two fields. The first field is made up of two reserved bits which are transmitted as dominant bits. The second field contains four bits which make up the Data Length Code (DLC). The DLC determines the number of data bytes in the Data Field of the Data Frame and is programmed through the use of the CAN Message Format Registers, located in each of the 15 message centers.

Figure 16- 3 Control Field



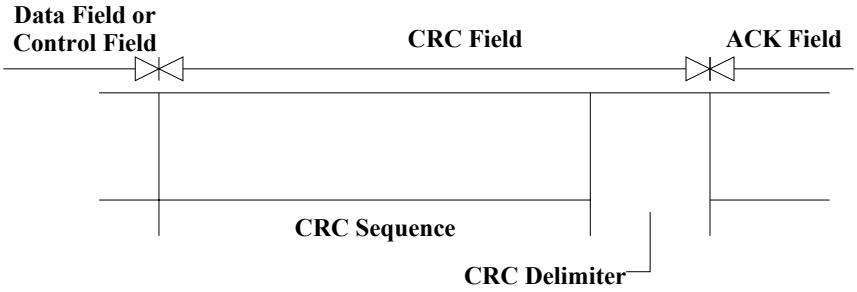
Data Field: (Standard and Extended Format)

The Data Field is made up of 0 to 8 bytes in a Data Frame and 0 bytes in a Remote Frame. The number of data bytes associated with a message center is programmed through the use of the CAN Message Format Registers, located in each of the 15 message centers. The data field contents are saved to the respective message center if the identifier test is successful, no errors are detected through the last bit of the end of frame, and an Error Frame does not immediately following the Data or Remote Frame. The data field is transmitted Least Significant Byte first, with the Msb of each byte transmitted first.

CRC Field: (Standard and Extended Format)

The CRC Field is made up of a 15 bit code which is the computed Cyclic Redundancy Check using the destuffed bits in the Start of Frame, the Arbitration Field, the Control Filed, and the Data Field (when present), and a CRC delimiter. The CRC calculation is limited to 127 bit maximum code word (a shortened BCH Code) with a CRC sequence length of 15 bits.

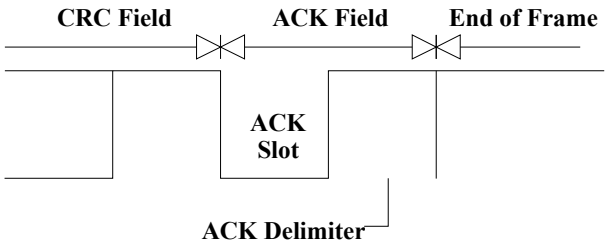
Figure 16- 4 CRC Field



Acknowledge Field (ACK): (Standard and Extended Format)

The ACK Field is made up of two bits. The transmitting node will send two recessive bits in the ACK field. The receiving nodes which have received the message and found the CRC Sequence to be correct will reply by driving the ACK Slot with a dominant bit. The ACK Delimiter is always a recessive bit.

Figure 16- 5 Acknowledge Field



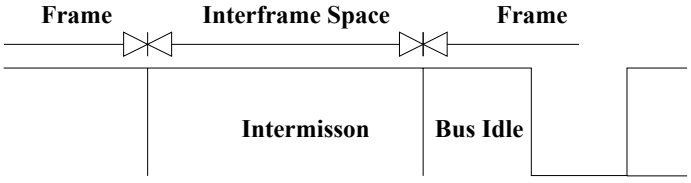
End of Frame: (Standard and Extended Format)

The End of Frame for both the Data and Remote Frame is established by the transmitter by sending seven recessive bits.

Interframe Spacing (Intermission): (Standard and Extended Format)

Data Frames and Remote Frames are separated from preceding frames by three recessive bits termed the Intermission. During the Intermission the only allowed signaling to the bus is by an Overload condition. No node is allowed to start a message transmission of a Data or Remote Frame during this period. If no node becomes active following the Interframe Space an indeterminate number of recessive bit times will transpire in the Bus Idle condition until the next transmission of a new Data or Remote Frame by a node.

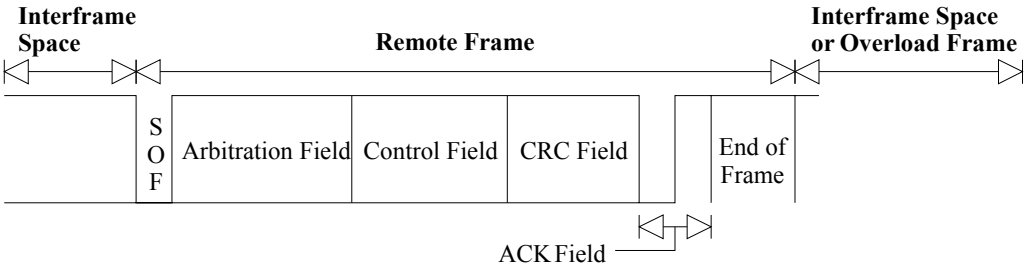
Figure 16- 6 Intermission



Remote Frame: (Standard and Extended Format)

The Remote Frame is transmitted by a CAN controller to request the transmission of the Data Frame with the same identifier. The Remote Frame is composed of seven fields. These include the Start of Frame, Arbitration Field, Control Field, Data Field, CRC Field, Acknowledge Field and an End of Frame.

Figure 16- 7 Remote Frame

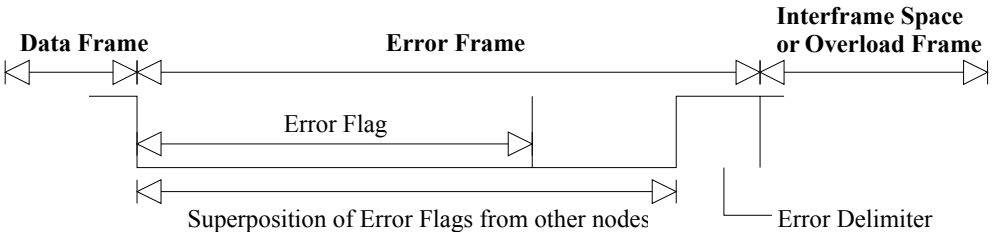


The Remote Frame is used when a CAN processor wishes to request data from another node. Sending a Remote Frame initiates a transmission of data from a source node with the same identifier (masked groups included). The primary bit pattern difference between a Data Frame and a Remote Frame is the RTR bit, which in the Remote Frame is sent as a recessive bit, and in the Data Frame is sent as a dominant bit. The Remote Frame also does not contain a data field, independent of the programmed values in the DTBYC3 - DTBYC0 bits in the respective CAN Message Format Register.

Error Frame:

The Error Frame is transmitted by a CAN controller when the CAN processor detects a bus error. The Error Frame is composed of two different fields. These are 1) the superposition of the Error Flags from different nodes and 2) the Error Delimiter.

Figure 16- 8 Error Frame



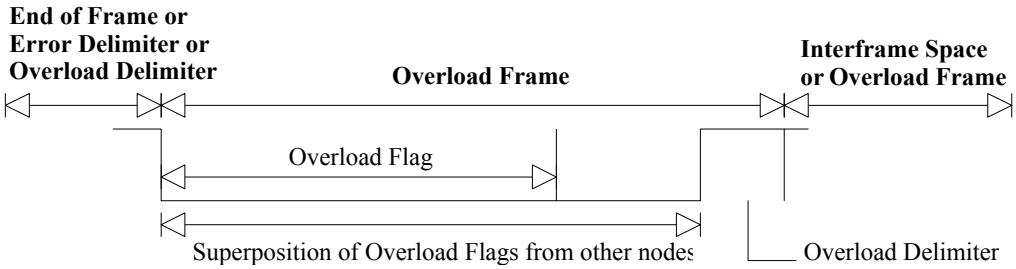
The Error Frame is composed of six dominant bits, which violate the CAN specification bit stuffing rule. If either of the CAN processors detect an error condition, that CAN processor will transmit an Error Frame. When this happens all nodes on the bus will detect the bit stuff error condition and will transmit their own Error Frame. The superpositioning of all of these Error Frames will lead to a total Error Frame length between 6 and 12 bits, depending on the response time and number of nodes in the system. Any messages (Data or Remote Frame) received by the CAN processors (successful or not) which are followed by an Error Frame will be discarded. After the transmission of an Error Flag each CAN processor will send an error delimiter (eight recessive bits) and will monitor the bus until it detects the change from the dominant to recessive bit level. The CAN modules will issue an Error Frame each time an Error Frame is detected. Following a series of Error Frames the CAN modules will enter into an Error Passive Mode. In the Error Passive Mode the CAN processors will transmit six recessive bits, and wait

until six equal bits of the same polarity have been detected. At this point the CAN processor will begin the next internal receive or transmission operation.

Overload Frame:

The Overload Frame provides an extra delay between Data or Remote Frames. The Overload Frame is composed of two different fields: the Overload Flag and the Overload Delimiter.

Figure 16- 9 Overload Frame



There are three conditions which lead to the transmission of an Overload Flag:

1) The internal conditions of a CAN receiver require a delay before the next Data or Remote Frame is sent. The DS80C390 CAN controllers are designed to prevent this condition for data rates at or below the 1 Mbit per second data rate.

2) The CAN processor detects a dominant bit at the first and second bit position of the Intermission.

3) If the CAN processor detects a dominant bit at the eighth bit of an Error Delimiter or Overload Delimiter, it will start transmitting an Overload Frame.

The error counters will not be incremented as a result of number 3. The CAN processor will only start an Overload Frame at the first bit of an expected Intermission if initiated by condition 1. Conditions 2 and 3 will result in the CAN processor transmitting an Overload Frame starting one bit after detecting the dominant bit. The Overload Flag consists of six dominant bits that correspond to an Error Flag. Because the Overload Frame is only transmitted at the first bit time of the Interframe Space, it is possible for the CAN processor to discriminate between an Error Frame and an Overload Frame. The Overload Flag destroys the Intermission field. When such a condition is detected, the CAN processor will detect the Overload condition and will begin transmitting an Overload Frame. After the transmission of an Overload Frame the CAN processors will monitor the bus for a dominant to recessive level change. The CAN processor will then begin the transmission of six additional recessive bits, for a total of seven recessive bits on the bus. The Overload Delimiter consists of eight recessive bits.

Initializing the CAN controllers

Software initialization of each CAN controller begins with the setting of the Software Initialization bit (SWINT) in the appropriate CAN Control SFR Register. When SWINT=1, the respective CAN module is disabled and the corresponding CAN transmit output will be placed in a recessive state. This in turn allows the microcontroller to write information into the CAN MOVX SRAM Control/Status/Mask registers without the possibility of corrupting data transmissions or receptions in progress. Setting SWINT will not clear the receive and transmit error counters, but will allow the microcontroller to write a common value to both error counters via the CAN Transmit Error SFR Register. Consult the description of the SWINT bit for specifics of the software initialization process.

All CAN registers located in the SFR memory map, with the exception of the CAN 0 and CAN 1 Control Registers, are cleared to a 00 Hex following a system Reset. The CAN 0 and CAN 1 Control Registers, are set to 0B Hex following a system Reset. CAN registers located in the MOVX memory map are indeterminate following a system Reset. A system Reset also clears both the receive and transmit error counters in the CAN controllers, takes the CAN processors off line, and sets the SWINT bit in the CAN 0/1 Control Register.

Following a reset, the following general registers must be initialized for proper operation of the CAN modules. These registers are in addition to specific registers associated with mask, format, or specific message centers.

Register	**Significance**
P5CNT (SFR A2h)	C0_I/O (P5CNT.3) must be set to enable CAN 0 pins P5.1 and P5.0. C1_I/O (P5CNT.4) must be set to enable CAN 1 pins P5.2 and P5.3
C0BT0, C0BT1 C1BT0, C1BT1 (MOVX SRAM xxxx04-5)	These MOVX SRAM control registers must be set to configure CAN 0 (C0BT0, C0BT1) or CAN 1 (C1BT0, C1BT1) bus timing. The exact values are dependent on the network configuration and environment.
COR (SFR CEh)	C0BPR7-6 (COR.4-3) must be configured as part of the CAN 0 bus timing C1BPR7-6 (COR.6-5) must be configured as part of the CAN 1 bus timing

CAN Interrupts

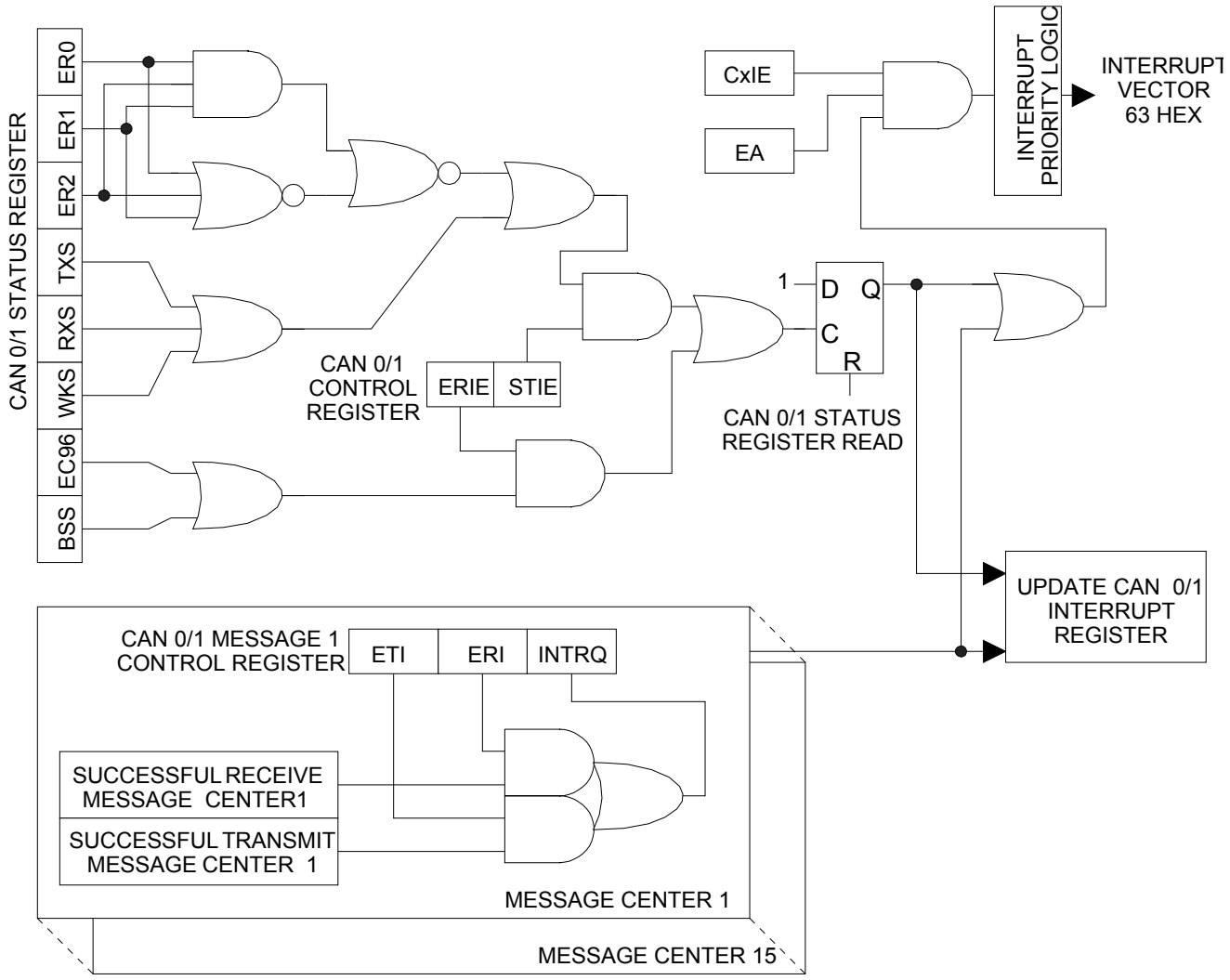
Each CAN processor is assigned one individual interrupt and one common CAN Bus Activity Interrupt which are globally enabled or disabled by the EA bit in the IE SFR register. The CAN 0/1 interrupt is generated by either a receive/transmit acknowledgment from one of the fifteen message centers or an error condition which results in a change in the CAN 0/1 Status Register. These interrupts are enabled via the C0IE or C1IE bit (CAN 0 or CAN 1) in the EIE register. The third CAN related interrupt is common to both CAN systems and is supplied to detect CAN bus activity on either CAN input pin. This interrupt is termed the CAN Bus Activity Interrupt, operates independent of the CAN processor, and is only available if one or both of the CAN processors have been connected to the respective Port 5 pins (via C0_I/O and/or C1_I/O in the Port 5 Control SFR).

CAN 0/1 receive/transmit interrupt sources are derived from a successful transmit or receive of data within one of the fifteen message centers as determined by the INTRQ bit in the associated CAN 0/1 Message (1-15) Control Register. Each message center (1-15) also provides separate receive and transmit interrupt enables via the ETI and ERI bits in the respective CAN 0/1 Message (1-15) Control Register. This allows each message center to be programmed to issue an interrupt request as per the application requirements of the message center. Each source is determined through the use of the CAN 0/1 Interrupt

SFR Register. Software must clear the respective INTRQ bit in the associated CAN 0/1 Message (1-15) Control Register to clear the interrupt source before leaving the interrupt routine.

The CAN 0/1 Interrupt source is connected to a change in the CAN 0/1 Status Register. Each of the status bits in the CAN 0/1 Status Register represents a potential source for the interrupt. To simplify the application and testing of a device, these sources are broken into two groups which are further enabled via the ERIE and STIE bits of the CAN 0/1 Control register. This allows the non-standard errors typically associated with development to be grouped under the STIE enable. These include the successful receive RXS, successful transmit TXS, wake status WKS, and general set of error conditions reported by ER2 - ER0. Also note that since the RXS and TXS bit are cleared by software, if a second message is received or transmitted before the RXS or TXS bits are cleared and after a read of the CAN 0/1 Status Register, a second interrupt will be generated. The remaining error sources comprise the BSS and CECE bits in the CAN 0/1 Status Register. These read-only bits are separately enabled via the ERIE bit in the CAN 0/1 Control register. A read of the CAN 0/1 Status Register is required to clear either of the two groups of Error interrupts. It is possible that multiple changes to the Status Register may occur before the register is read; in that case the Status Register will generate only one interrupt. The following figure provides a graphical illustration of the interrupt sources and their respective interrupt enables.

Figure 16- 10 CAN Interrupt Logic



Arbitration/Masking Considerations

Each CAN processor evaluates CAN bus activity to determine if an incoming message is loaded into one of the 15 message centers. Acceptance of a message is determined by comparing the message's ID or data field against the corresponding arbitration value loaded into each message center and checking if the bits match. Messages that contain bit errors or which fail arbitration are discarded. The incoming message is tested in order against each enabled message center (enabled by the MSRDY bit in the CAN Message Control Register) from 1 to 15. The first message center to successfully pass the test will receive the incoming message and end the testing, and the message is loaded into the respective message center.

The CAN modules support an optional masking feature that restricts arbitration to those bits that are masked with a 1 in the respective masking register. By selectively programming the message center arbitration registers and the related masks, it is possible to allow groups of incoming messages to be loaded into any single message center. Each pair of mask and arbitration registers has the same number of bits as the message ID in the incoming message. When masking is enabled, only those arbitration and identifier bits that correspond to a 1 in the masking register will be compared. Programming a bit in the mask to a 0 will make the comparison of those arbitration and identifier bits a don't care, automatically registering a match between those bits. If all of the bits in the mask are programmed to a 0, any incoming message arbitration field will match with any message center arbitration value. On the other hand, if a mask is programmed with all 1's all of the arbitration and identifier bits must match identically before the incoming message will be loaded into the message center.

The DS80C390 supports two types of arbitration: basic and media. Basic arbitration compares either 29-bits (EX/$\overline{ST}$ =1) or 11-bits (EX/$\overline{ST}$ =0) of the message ID against the corresponding bits in the 4 CAN Arbitration registers (CnMxAR0-3). Each message center can be individually be configured for 29- or 11-bit operation. If the Message Identification Mask Enable bit (MEME) is set, the CAN module will utilize the Standard Global Mask registers (CnSGM0-1) when EX/$\overline{ST}$ =0 or the Extended Global Mask registers (CnEGM0-3) when EX/$\overline{ST}$ =1. In either case, only those bits in the message ID and arbitration registers which correspond with a 1 in the mask register will be compared. Bits corresponding with 0 in the mask register will be ignored, creating a don't care condition. Filling the mask register with all 0s while MEME=1 will cause the arbitration circuitry to automatically match all message IDs. When MEME=0, all ID bits in the incoming message are compared directly (bit for bit) with the respective arbitration bits of the message center.

Media arbitration is an optional second arbitration performed when the Media Identification Mask Enable bit (MDME) is set. Media arbitration compares the first and second byte of the data field in each message against two 8-bit Media Arbitration bytes (stored at locations CnMA0, CnMA1). If the incoming arbitration field matches a specific message arbitration value and the first two data bytes match the two 8-bit Media Arbitration bytes (and no bit errors are detected) the message is loaded into the respective message center. Unlike the Identification Mask Enable (MEME), however, when MDME=0 no testing will be performed of the first two bytes of the incoming data field.

MESSAGE CENTER 15

Message center 15 supports an additional set of set of masks to supplement basic arbitration. While this message center performs basic and media arbitration as per message centers 1-14, it also uses the Cn15M3-0 mask registers perform an additional level of filtering during basic (i.e., not media) arbitration. When determining arbitration for message center 15, the contents of Cn15M3-0 are logically ANDed with either CnEGM3-0 (if EX/$\overline{ST}$ =1 for message center 15) or CnSGM1-0 (if EX/$\overline{ST}$ =0 for message center 15). This ANDed value is then used in place of CnEGM3-0 or CnSGM1-0 when performing basic

arbitration as described previous. If the MDME bit is set then the incoming message must pass the media arbitration test as well.

Message center 15 has a buffered FIFO arrangement to allow up to two received messages to be received without being lost prior to the microcontroller reading of the first message. The first message received by message center 15 is stored in the normal MOVX memory location for Message Center 15, if the previous message has been already read by the microcontroller. If the first message has not been read, then the incoming message is buffered internally until the first message is read, at which time the second message is automatically loaded into the first (MOVX) message 15 slot, allowing software to then read the second message. The CAN module determines if the first message has been read is by software clearing the DTUP bit and the EXTRQ bit. If a third message comes in before the second message has been copied into the MOVX message 15 slot, then the third message will write over the second buffered message. Software should clear the INTRQ bit as well as the DTUP and EXTRQ bit after reading each message in the MOVX message 15 center. The WTOE bit associated with message center 15 has unique operating considerations, described later in the section regarding the function of the WTOE bit.

Transmitting and Receiving Messages

All CAN data is sent and received through message centers. All CAN message centers for both CAN modules are identical with the exception of message center 15. Message center 15 has been designed as a receive only center and is also shadow-buffered to help prevent the loss of incoming messages, when the software is not able to read the first message before the next message is loaded. All message centers, with the exception of message center 15, are capable of four different operations. These are:

Transmitting a data message
Receiving a data message
Transmitting a remote frame request
Receiving a remote frame request

Transmitting Data Messages:
Starting with the lowest numbered message center (highest priority) each CAN module sequentially scans each message center until it finds a message center that is proper enabled for transmission ($T/\overline{R}$ = 1, TIH = 0, DTUP = 1, MSRDY = 1, and MTRQ = 1). The contents of the respective message center is then transferred to the transmit buffer and the CAN module attempts to transmit the message. If successful the appropriate MTRQ bit will be cleared to 0, indicating that the message was successfully sent. Following a successful transmission, loss of arbitration, or an error condition, the CAN module will again search for a properly configured message center, starting with the lowest numbered message center. This search relationship will always allow the highest priority message center to be transmitted, independent of the last successful (MTRQ = 0) or unsuccessful (MTRQ = 1) message transmission.

Receiving Data Messages:
Each incoming data message is compared sequentially with each receive enabled ($T/\overline{R}$ = 0) message center starting with the lowest numbered message center (highest priority) and proceeding to the highest numbered message center. This testing continues until a match is found (incorporating masking functions as required), at which time the incoming message is stored in the respective message center. Higher numbered message centers that are not reviewed prior to the match will not be evaluated during the current message test. When the WTOE=1, the CAN module can overwrite receive message centers that have DTUP = 1, which will in turn set ROW = 1. When WTOE = 0, incoming messages will not overwrite receive message centers that have DTUP = 1.

Message center 15 is a special receive-only, FIFO-buffered message center, designed to receive messages not accepted by the other message centers. The ROW bit in message center 15 is associated with the overwrite of the shadow buffer for message center 15. The EXTRQ and DTUP bits are shadow buffered to allow the buffered message and the message center 15 value to take on different relationships. The EXTRQ and DTUP values read by the microcontroller are not those of the shadow buffer as is the case with the ROW bit, but are the current values associated with message center 15. The shadow buffer is automatically loaded into message center 15 when **both** the DTUP bit and the EXTRQ bit are cleared. If either DTUP or EXTRQ are left set when clearing the other, any message in the shadow buffer will not be transferred to the message 15 registers, and any incoming messages for message 15 will be stored in the shadow buffer (if WTOE = 1), or will be lost if (WTOE = 0).

Transmitting Remote Frame Requests

Starting with the lowest numbered message center (highest priority) each CAN module sequentially scans each message center. When it finds a message center properly enabled to transmit a remote frame ($T/\overline{R}$ = 0, MSRDY = 1, and MTRQ = 1), the contents of the respective message center is then transferred to the transmit buffer and the CAN module attempts to transmit the message. If successful the appropriate MTRQ bit will be cleared to 0, indicating that the message was successfully sent. Following a successful transmission, loss of arbitration, or an error condition, the CAN module will again search for a properly configured message center, starting with the lowest numbered message center. This search relationship will always allow the highest priority message center to be transmitted, independent of the last successful (MTRQ = 0) or unsuccessful (MTRQ = 1) message transmission. The state of the TIH bit does not effect the transmission of a remote frame request.

Receiving/Responding to Remote Frame Requests

The remote frame request is handled like a data frame with data length zero and the EXTRQ and RXS bits are set. Each incoming Remote Frame Request (RFR) message is compared sequentially with each enabled (MSRDY = 1) message center starting with the lowest numbered message center (highest priority) and proceeding to the highest numbered message center. Testing continues until a match is found (incorporating masking functions as required), at which time the incoming RFR message is stored in the respective message center, the DTBYC bits are updated to indicate the requested number of return bytes (DTBYC=0 for a remote frame request), and EXTRQ and MTRQ are both set to 1. When the message is successfully received and stored, an interrupt of the corresponding message center will be asserted if enabled by the ERI bit. The EXTRQ bit can be left set if the message center is reconfigured to perform a transmit ($T/\overline{R}$ = 1) and used in the standard reply of a remote frame operating with transmit message centers. EXTRQ can also be cleared by software if the current message center is not being used to reply to the remote frame request. Higher numbered message centers (lower priority) that are not reviewed prior to the match will not be evaluated during the current message test. Depending on the state of the transmit/receive bit for that message center, the CAN module will perform one of two responses.

If the microcontroller wishes to request data from another node, it first clears the respective MSRDY bit to 0 and then writes the identifier and control bits in this message center, configures the message center as a receive message center ($T/\overline{R}$ = 0) and then sets the MTRQ bit. After a successful transmission, the CAN module will clear MTRQ = 0 and set TXS = 1. In addition to the TXS bit, if the ETI bit is set, the successful transmission will also set the corresponding INTRQ bit. Requesting data from another node is possible in message centers 1 to 14. As seen above the CAN module sends a remote frame request and receives the data frame in the same message center that sent the request. Therefore, only one mailbox is necessary to do a remote request.

Message centers enabled for transmission (T/$\overline{R}$ = 1) will set the EXTRQ and MTRQ bits in the corresponding message center when a remote frame request is successfully received, to mark the message as a 'to be sent' message. The CAN module will attempt to automatically transmit the requested if the message center is fully enabled to do so (MSRDY = 1, TIH = 0, DTUP = 1). After the transmission, the TXS bit in the status register is set, the EXTRQ and MTRQ bits are reset to a 0 and a message center interrupt of the corresponding message center is asserted if enabled by the respective ETI bit. If the transmit inhibit bit is set (TIH = 1), the message center will receive the RFR, modifying the DTBYC and/or arbitration bits if necessary, but the return data will not be transmitted until TIH = 0.

If software wants to modify the data in a message center configured for transmission of an answer to a remote request (EXTRQ set to a 1), the microcontroller must set the TIH = 1 and DTUP = 0. The microcontroller may then access the data byte registers 0 -7, data byte count (DTBYC3–0), the extended or standard mode bit (EX/$\overline{ST}$), and the mask enables (MEME and MDME) of the message center to load the required settings. Following the set up, the software should reset TIH to a 0 and sets DTUP to a 1 bit to signal the CAN that the access is finished. Until the DTUP = 1 and TIH = 0, the transmission of this mailbox is not permitted. Thus, the CAN will transmit the newest data and reset EXTRQ = 0 after the transmission is complete. The message center must first be disabled to change the identifier or the direction control (T/$\overline{R}$).

Message centers enabled for reception (T/$\overline{R}$ = 0) will **not** automatically transmit the requested data. The Remote Frame Request will, however, continue to store the requested number of return bytes in DTBYC and set EXTRQ = 1. No data bytes are received or stored from a remote frame request. The message center can then be configured via software to either function as transmitter (T/$\overline{R}$ = 1) and transmit the requested data, or the microcontroller can configure another message center in a transmit mode (T/$\overline{R}$ = 1) to send the requested data. Note that the MTRQ bit is not set by the loading of a matching remote frame request, when T/$\overline{R}$ = 0. RXS must be previously cleared by software or a system reset.

When a remote frame is received the CAN module can be configure to either automatically transmit data back to the remote node or to allow the microcontroller to intervene and establish the conditions of the transmitting of the return message. The following examples outline various options to respond to remote frame requests.

Case 1) Automatic Reply
CAN Controller receives a remote frame Request (RFR) and automatically transmits data without additional software intervention.

1. Software sets T/$\overline{R}$ = 1, MSRDY = 0, DTUP = 0, and TIH = 1.
2. Software loads data into respective message center.
3. Software sets MSRDY = 1, DTUP = 1 and TIH = 0 in same instruction.
   Note: Software does not change MTRQ = 0 from previous completed transmission
4. CAN does not transmit data (MTRQ = 0), but waits for RFR.
5. CAN successfully receives RFR.
6. CAN forces MTRQ = 1 and EXTRQ = 1
7. CAN loads DTBYC from RFR and ID into arbitration registers.
8. CAN automatically transmits data in respective message center.
9. CAN clears EXTRQ = 0 and MTRQ = 0.

Case 2) **Software-Initiated Reply** (Using TIH as Gating Control)
CAN module wishes to receive an RFR and wait for software to determine when and what will be transmitted in reference to RFR.

1. Software sets T/$\overline{R}$ = 1, MSRDY = 0, DTUP = 0, and TIH = 1.
2. Software then loads data into respective message center.
3. Software sets MSRDY = 1, DTUP = 1 and TIH = 1 in same instruction.
   Note: Software does not change MTRQ = 0 from previous completed transmission
4. CAN does not transmit data (MTRQ = 0), but waits for RFR.
5. CAN successfully receives RFR.
6. CAN forces MTRQ = 1 and EXTRQ = 1.
7. CAN loads DTBYC from RFR and ID into arbitration registers.
8. CAN waits for software to read message center and determine the fact that EXTRQ = 1.
9. Software may load data into message center (or it may already have the data established).
10. Software writes MSRDY = 1, DTUP = 1 and TIH = 0 in same instruction.
11. CAN will automatically transmit data (as per RFR DTBYC) in respective message center.
12. CAN clears EXTRQ = 0 and MTRQ = 0.

Case 3) **Software-Initiated Reply** (Reply via same message center, using TIH as Gating Control)
CAN module wishes to receive an RFR in a receive-configured (T/$\overline{R}$ = 0) message center. When the data is received, the message center will be reconfigured send data back to the remote request node. This relationship is not possible for message center 15.

1. Software sets T/$\overline{R}$ = 0, MSRDY = 1, and DTUP = 0 and awaits either data frame or RFR.
   Note: Software does not change MTRQ = 0 from previous completed transmission
2. CAN successfully receives RFR.
3. CAN forces EXTRQ = 1 and DTUP = 1.
4. MTRQ can not be written to a 1 by the CAN when T/$\overline{R}$ = 0 and is left as MTRQ = 0
5. CAN loads DTBYC from RFR and ID into arbitration registers.
6. CAN waits for Software to read message center and determine the fact that EXTRQ = 1.
7. Software disables message center and converts message center into transmit message center.
8. Software clears MSRDY = 0 to disable message center. Software leaves EXTRQ = 1.
9. Software then forces message center to transmit mode, T/R = 1.
10. Software writes MSRDY = 0, DTUP = 0 and TIH = 1 in preparation to load data.
11. Software loads data into message center.
12. Software writes MSRDY = 1, MTRQ = 1, DTUP = 1 and TIH = 0 in same instruction.
13. Note that Software leaves EXTRQ = 1.
14. CAN will automatically transmit data (as per RFR DTBYC) in respective message center.
15. CAN clears EXTRQ = 0 and MTRQ =0.

Case 4) **Software-Initiated Reply** (Reply via same different center, using TIH as Gating Control) CAN Controller wishes to receive an RFR in a message center (denoted MC1) configured also be able to receive data ($T/\overline{R} = 0$) and to wait for software to select another message center (denoted MC2) to send data back to remote request node.

1. Software sets $T/\overline{R} = 0$, MSRDY = 1, and DTUP = 0 in MC1 and awaits either data frame or RFR.
   Note: Software does not change MTRQ = 0 in MC1 from previous completed transmission.
2. CAN successfully receives RFR in MC1.
3. CAN forces EXTRQ = 1 and DTUP = 1 in MC1.
   MTRQ can not be written to a 1 by the CAN when $T/\overline{R} = 0$ and is left as MTRQ = 0
4. CAN loads DTBYC from RFR and ID into arbitration registers in MC1.
5. CAN waits for Software to read message center and determine the fact that EXTRQ = 1.
6. Software disables in MC1 to transfer information to MC2.
7. Software clears MSRDY = 0 to disable MC1. Software leaves EXTRQ = 1.
8. Software clears MSRDY = 0 in a MC2.
9. Software forces MC2 to transmit mode $T/\overline{R} = 1$.
10. Software loads ID and DTBYC value from MC1 into ID and DTBYC value for MC2.
11. Software writes MSRDY = 0, DTUP = 0 and TIH = 1 in MC2 in preparation to load data to MC2.
12. Software loads data into MC2.
13. Software writes MSRDY = 1, MTRQ = 1 EXTRQ = 0, DTUP = 1 and TIH = 0 in MC2 in same instruction.
    Note that CAN has not set EXTRQ in MC2, and is not required to be set for transmission of data from MC2.
14. CAN will automatically transmit data (as per RFR DTBYC) in MC2.
15. CAN clears MTRQ =0 (leaving previous EXTRQ = 0 cleared).
16. Software sets $T/\overline{R} = 0$, MSRDY = 1, EXTRQ = 0, and DTUP = 0 in MC1 and awaits either next RFR or data frame.
    Note that MTRQ is still cleared in MC1 since MC1 has not been set to a transmit mode.

Remote Frame Handling in Relation to the DTBYC Bits

The DTBYC bits function slightly differently when Remote Frames are used. In that case, the data length code will be overwritten by the data length code field of the incoming remote request frame. These requested data bytes will be sent in the data frame which answers the remote request. The following example demonstrates how the DTBYC bits are modified by a received remote frame request.

1. Assume the microcontroller has programmed the following into a message center:
   DTBYC = 5, data field = 75 AF 43 2E 12 78 90 00
   (Note that only the first through the fifth data bytes will be recognized because DTBYC=5)

2. When the CAN module successfully receives a remote frame with the following data:
   Identifier = ID, DTBYC = 2, RTR = 1.

3. The incoming message will overwrite the identifier and the data length code. The new data in the message center will be:
   DTBYC = 2, data field = 75 AF 43 2E 12 78 90 00
   (Note that only the first and second data bytes are recognized because DTBYC is now 2)

4. The outgoing response will be a data frame containing the following information:
   DTBYC = 2, data field = 75 AF

Important Information Concerning ID Changes when Awaiting Data from a Previous Remote Frame Request:
The use of acceptance filtering (MEME=1) in conjunction with remote frame requests can result in a modification of the message center arbitration registers. When a message center is configured to transmit a remote frame request (MTRQ = 1, EXTRQ = 0, T/$\overline{R}$ = 0 and MSRDY = 1) it is possible for a second Frame Request from an external node to modify the initial Arbitration Register value of the current message center prior to the current message center receiving the requested data if arbitration masks are used. When a remote frame request is received, the message ID is loaded into that message center's arbitration registers. When message identification masking is not used (MEME=0), the message ID will always match the arbitration value, so the process will be transparent. If masking is used, however, the message ID ANDed with the appropriate mask will be loaded into that message center's arbitration registers, resulting in a change of the arbitration values for that message center. To prevent this situation, acceptance filtering should be disabled (MEME = 0) for any message center configured for remote frame handling. An alternate solution would be to disable the overwrite feature for that message center, which also prevents incoming messages from altering the message center ID.

Overwrite Enable/Disable Feature

The Write Over Enable bit (WTOE) located in each message center (CnMxAR3.0) enables or disables the overwriting of unread messages in message centers 1 through 15. Programming WTOE = 1 following a system reset or CRST bit-enabled reset allows newly received messages which pass arbitration to overwrite unread (i.e., message centers with DTUP=1) messages. When an overwrite occurs the Receive Overwrite (ROW) bit in the respective CAN Message Control Register will be set. When WTOE = 0, message centers which have data waiting to be read (indicated by DTUP = 1) or transmitted (EXTRQ=1) will not be overwritten by incoming data.

Special care must be taken when reading data from a message center with the overwrite feature enabled (WTOE = 1). The caution is needed because the WTOE bit, when set, allows an incoming message to overwrite the message center. If this overwrite occurs at the same time that software is attempting to read several bytes from the message center (such as a multi-byte data field), it is possible that the read could return a mix of information from the old and overwriting messages. To avoid this situation, software should clear the DTUP bit to 0 prior to reading the message center, and then verify afterwards that the DTUP bit remained at 0. If DTUP remains cleared after the read, no overwrite occurred and the returned data was correct. If DTUP = 1 after the read then software again should clear DTUP = 0 and re-read the message center, since a possible overwrite has occurred. The original message will be lost (as planned since WTOE=1), but new message should be available on the next read.

One important use of the WTOE bit is to allow the microprocessor to program multiple message centers with the same ID when operating in the receive mode, with WTOE=0. This allows the CAN module to store multiple incoming messages in a series of message centers, creating a large storage area for high-speed recovery of large amounts of data. The CPU is required to manage the use of these message centers to keep track of the incoming data, but the use of multiple message centers and disabling of their

overwrite (WTOE=0) function prevents the module from potentially losing data during a high-speed data transfer. In transmit mode, the WTOE bit prevents a message center ID from being overwritten by an incoming remote frame.

The following examples demonstrate the use of the WTOE and other bits when using multiple message centers with the same arbitration value. Case 2 illustrates the approach described above for configuring multiple message centers to capture a large amount of data at a relatively high rate.

Case 1: WTOE=1 (Overwrites allowed)

1. Software configures message center 1 & 2 with the same arbitration value (abbreviated AV).
2. Software configures message center 1 & 2 to receive ($T/\overline{R}$ = 0) and to allow message overwrite WTOE=1.
3. The first message received that matches AV will be stored into message center 1, DTUP = 1.
4. The second message that matches AV will be stored into message center 1, DTUP = ROW = 1.
5. The third message that matches AV will be stored into message center 1.
6. etc.

Note that in this example message center 2 will never receive a message, and that if software does not read message center 1 before the second message is received, the first message will be lost.

Case 2: WTOE=0 (Overwrites disabled)

1. Software configures message center 1 & 2 with the same arbitration value (abbreviated AV).
2. Software configures message center 1 & 2 to receive ($T/\overline{R}$ = 0) and to disable message overwrite WTOE=0.
3. The first message received that matches AV will be stored in message center 1, DTUP = 1.
4. The second message received that matches AV will be stored in message center 2, DTUP = 1
5. Software reads message center 1 and then programs message center 1 DTUP = 0.
6. The third message received that matches AV will be stored into message center 1, DTUP = 1.
7. Software reads message center 2 and then programs message center 2 DTUP = 0.
8. The fourth message received that matches AV will be stored into message center 2, DTUP = 1
9. etc.

Note that in this example message center 1 or 2 will never be overwritten. The user must insure that the proper number of message centers be allocated to the same arbitration value when using this arrangement. If software fails to read the allocated message group, an incoming message may be lost without software realizing it (ROW is never set when WTOE = 0). To put a message center back into operation software must force DTUP = 0 and EXTRQ = 0. This indicates that software has read the message center.

Special Considerations for Message Center 15

Message center 15 incorporates a shadow message center used to buffer incoming messages, in addition to the standard message center registers. When the message center is empty (DTUP=EXTRQ=0), incoming messages are loaded directly into the message center registers. When the message center has unread data (DTUP =1) or a pending remote frame request (EXTRQ = 1), incoming messages are loaded into the shadow message center. Unread contents of the shadow message center are automatically loaded into the message center when it becomes empty (DTUP=0). An overwrite condition is possible when both the message center 15 and shadow message centers are full.

The response of message center 15 to the overwrite condition is dependent on the Writeover Enable (WTOE) bit. When overwrite is enabled (WTOE = 1) and there is unread data (DTUP =1) or a pending

remote frame request (EXTRQ = 1), successfully received messages are stored in the shadow message center, overwriting existing data. If the shadow message center contained previously unread data at the time of the overwrite, the message center 15 ROW bit will be set. If the shadow message center was empty at the time of the overwrite, then the incoming message will overwrite the previous message in the shadow buffer and ROW will be set to a 1. Note that the message center 15 ROW bit reflects only an overwrite of the shadow message center, not the message center registers as with message centers 1-14.

When WTOE = 0 and there is unread data (DTUP =1) or a pending remote frame request (EXTRQ = 1) in message center 15 and there is already a message stored in the shadow buffer, incoming messages will not be stored in either the message center or shadow buffers.

Using the Autobaud Feature

It is sometimes necessary to connect a CAN node to a network with an unknown baud rate. The autobaud feature of the DS80C390 provides a simple way for the CAN module to determine the baud rate of the network and reconfigure the DS80C390 to operate at that baud rate. Special hardware inside the CAN module allows it to interface to a fully functional CAN bus and perform the autobaud feature without disturbing other CAN nodes.

The theory behind the CAN autobaud feature is relatively simple. If a CAN module operating at a particular baud rate listens in on a CAN bus operating at a different baud rate, it will see a random bit stream. Because the bit stream does not conform to the CAN 2.0B protocol, a large number of bus errors (bit 0 error, bit 1 error, bit stuff error, etc.) will be seen by the "listening" CAN. These errors will increment the appropriate CAN error counter registers. With only a moderate amount of CAN traffic, enough errors will quickly accumulate to set the CAN Error Count Exceeded (CECE) bit in the DS80C390 CAN module. This can be used as an indicator that the DS80C390 is not operating at the same baud rate as the CAN. The DS80C390 would then adjust its baud rate and repeat the process.

If, after a period of time, only a small number of errors have accumulated (most likely due to normal transmission noise), then the DS80C390 is operating at the correct baud rate. The autobaud process is further simplified by the fact that most networks only operate at a small number of values. For example, DeviceNet operates at 125 kbps, 250 kbps, and 500 kbps, so a device attempting to autobaud to a DeviceNet network would only have to test three baud rates.

The autobaud feature for a CAN module is enabled by setting the appropriate Autobaud bit (C0C.2 or C1C.2). Setting this bit activates a special loopback circuit within the CAN module that logically ANDs incoming network data received on the RX pin with the TX pin of the CAN module. While the autobaud bit is set, the CAN module will disable its transmit output and place it in the recessive (high) state, so that error frames generated by the autobauding CAN module do not disturb other devices on the network during the procedure. Also, while in this state, messages are received but not stored.

The following user-defined software procedure can be used in conjunction with the autobaud feature to determine the baud rate of the network.

1) Set CRST=1 to disable bus activity. Setting this bit also sets the SWINT bit, enabling access to Control/Status registers, and also clears the CxRE and CxTE registers,
2) Configure bus timing registers to set desired baud rate,
3) Set autobaud bit =1,
4) Set SWINT=0 to enable CAN module and begin listening for errors,
5) Delay approximately 500 ms (allow enough time for >128 errors to occur),

6) If CAN Error Count Exceeded (CECE) bit is set, baud rate is incorrect. Select a new baud rate and repeat procedure. If CECE bit is not set, the DS80C390 CAN module is set to the correct baud rate.

Bus Off / Bus Off Recovery and Error Counter Operation

Each CAN module contains two SFRs that allow software to monitor and modify (under controlled conditions) the error counts associated with the transmit and receive error counters in each CAN module. These registers can be read at any time. Writing the CAN Transmit Error Counter registers updates both the Transmit Error Counter registers and the Receive Error Counter registers with the same value. Details are given in the SFR description of these registers. These counters are incremented or decremented according to CAN specification version 2.0B, summarized in the rules below. The error counters are initialized by a CRST = 1 or a system reset to 00h. The error counters remain unchanged when the CAN module enters and exits from a low power mode via the SIESTA or PDE bit.
Changes to the error counters are performed according to the following rules. This level of detail is not necessary for the average CAN user, and full information is provided in the CAN 2.0B specification. More than one rule may apply to a given message.

Condition	**Effect on error counters**
Error detected by receiver, unless the detected error was a bit error during the sending of an active error flag or an overload flag.	Receive Error Counter incremented by 1.
Receiver detects a dominant bit as the first bit after sending an error flag.	Receive Error Counter incremented by 8.
Transmitter sends an error flag. Note, however, that the transmit error count will not change if: 1) The transmitter is error passive and detects an acknowledgement error because of not detecting a dominant acknowledge and does not detect a dominant bit while sending its passive error flag. 2) Or, if the transmitter sends an error flag because a stuff error occurred during arbitration, and has been sent as recessive but monitored as dominant.	Transmit Error Counter incremented by 8.
Transmitter detects a bit error while sending an active error flag or an overload flag.	Transmit Error Counter incremented by 8.
Receiver detects a bit error while sending an active error flag or an overload flag.	Receive Error Counter incremented by 8.
Node detects the 14th consecutive dominant bit (in case of an active error flag or an overload flag), or detects the 8th consecutive dominant bit following a passive error flag, or after a sequence of additional eight consecutive dominant bits.	Transmit Error Counter incremented by 8. Receive Error Counter incremented by 8.
Message is successfully transmitted (acknowledge received and no error until end of frame is complete)	Transmit Error Count is decremented by 1 (unless it was already 0).

A message has been successfully received (reception without error up to the acknowledge slot and the successful sending of the acknowledge bit), and the receive error count was between 1 and 127.	Receive Error Counter decremented by 1.
A message has been successfully received (reception without error up to the acknowledge slot and the successful sending of the acknowledge bit), and the receive error count was greater than 127.	Receive Error Counter is set to a value between 119 and 127.

A node is error passive when the transmit error count equals or exceeds 128, or when the receive error count equals or exceeds 128. An error condition letting a node become error passive causes the node to send an active error flag. An error passive node becomes error active again when both the transmit error count and the receive error count are less than or equal to 127.

A node is bus off when the transmit error count is greater than or equal to 256. A bus off node will become error active (no longer bus off) with its error counters both set to 0 after 128 occurrence of 11 consecutive recessive bits have been monitored on the bus.

After exceeding the error passive limit (128), the receive error counter will not be increased any further. When a message was received correctly, the counter is set again to a value between 119 and 127 (compare with CAN 2.0B specification). After reaching the "bus off" status, the transmit error counter is undefined while the receive error counter is cleared and changes its function. The receive error counter will be incremented after every 11 consecutive recessive bits on the bus. These 11 bits correspond to the gap between two messages on the bus. If the receive error counter reaches the count 128, following the bus off recovery sequence, the CAN module changes automatically back to the status of "bus on" and then sets SWINT = 1. After setting SWINT, all internal flags of the CAN module are reset and the error counters are cleared. A recovery from a bus off condition does not alter any of the previously programmed MOVX memory values and will also not alter SFR registers, apart from the transmit and receive error SFR registers and the error conditions displayed in CAN Status Register. The bus timing will remain as previously programmed.

Bit Timing

Bit timing in the CAN 2.0B specification is based on a unit called the nominal bit time. The nominal bit time is further subdivided into four specific time periods.

1. The SYNC_SEG time segment is where an edge is expected when synchronizing to the CAN Bus.

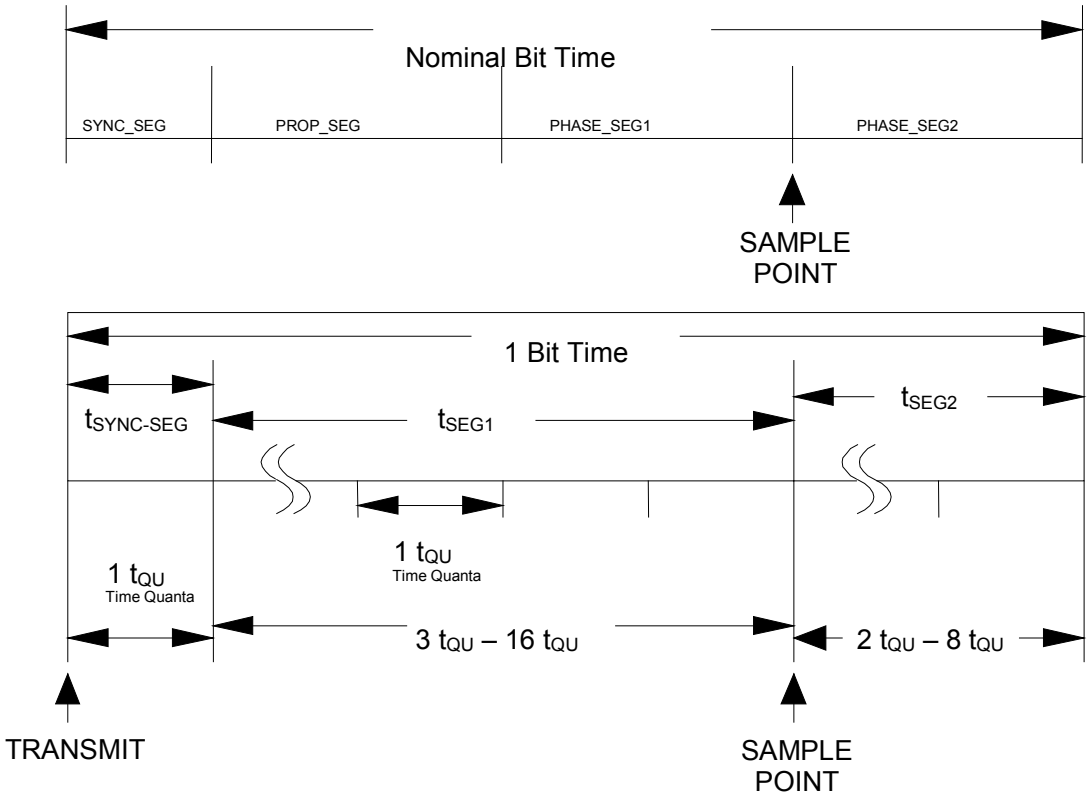
2. The PROP_SEG time segment is provided to compensate for the physical times associated with the CAN Bus network

3 & 4. The PHASE_SEG1 and PHASE_SEG2 time segments compensate for edge phase errors. The PHASE_SEG1 and PHASE_SEG2 time segments can be lengthened or shorted through the use of the SJW1 and SJW0 bits in the CAN 0/1 Bus Timing Register Zero.

The CAN bus is data is evaluated at the sample point. A time quantum (t_{QU}) is a unit of time derived from the division of the microprocessor crystal oscillator by both the Baud Rate Prescaler (programmed by the BPR5 - BPR0 bits in the CAN 0/1 Bus Timing register) and the System Clock Divider (programmed by the SCD2 - SCD0 bits in the Clock Output Register). Combining the PROP_SEG and PHASE_SEG1 time segments into one time period termed t_{TSEG1} and equating the SYNC_SEG time segment to t_{SYNC_SEG}

and PHASE_SEG2 to t_{TSEG2}, provides the basis for the time segments outlined below and in the CAN Bus Timing SFR Register descriptions. These are shown in the following figure.

Figure 16- 11 Bit Timing



The CAN 0/1 Bus Timing Register Zero (C0BT0/C1BT0) contains the control bits for the PHASE_SEG1 and PHASE_SEG2 time segments as well as the Baud Rate Prescaler (BPR5-0) bits. CAN 0/1 Bus Timing Register One (C0BT1/C1BT1) controls the sampling rate, the Time Segment Two bits that control the number of clock cycles assigned to the Phase Segment 2 portion, and the Time segment One bits that determine the number of clock cycles assigned to the Phase Segment 1 portion. The value of both of the Bus Timing registers are automatically loaded into the CAN module following each software change of the SWINT bit from a 1 to a 0 by the microcontroller. The bit timing parameters must be configured before starting operation of the CAN module. These registers can only be modified during a software initialization (SWINT = 1), when the CAN module is NOT in a bus off mode, and after the removal of a system reset or a CAN reset. To avoid unpredictable behavior of the CAN module the Bus Timing Registers should never be written with all zeros. To prevent this the SWINT is forced to 0 when TSEG1 = TSEG2 = 00h.

The timing of the various time segments is determined via the following formulae. Most users will never need to perform these calculations, as other devices already attached to the network will dictate the bus timing parameters.

$$t_{QU} = \frac{BRPV \cdot CCD}{F_{OSC}}$$

$$t_{SYNC\_SEG} = 1 \cdot t_{QU}$$

$$t_{TSEG1} = (TS1\_LEN) \cdot t_{QU}$$

$$t_{TSEG2} = (TS2\_LEN) \cdot t_{QU}$$

$$t_{SJW} = (SJW) \cdot t_{QU}$$

$$t_{QU} \text{ per bit} = \frac{1}{\text{baud rate} \cdot t_{QU}}$$

(only integer values are permitted.)

where BPRV is the CAN baud rate prescaler value found in the description of the C0BT0/C1BT0 registers, F_{OSC} is the crystal or external oscillator frequency of the microprocessor, and TS1_LEN and TS2_LEN are listed in the description of the TSEG26-24 and TSEG13-10 bits in the CAN Bus Timing Register 1. SJW is listed in the description of the SJW1-0 bits in the CAN Bus Timing Register 0. The CCD is the CAN clock divide value, calculated from the following table.

CD1	CD0	4X/$\overline{2X}$	CCD
0	0	1	0.5
0	0	0	1
1	0	x	2
1	1	x	512

The following restrictions apply to the above equations:

$$t_{TSEG1} \geq t_{TSEG2}$$
$$t_{TSEG2} \geq t_{SJW}$$
$$t_{SJW} < t_{TSEG1}$$
$$2 \leq TS1\_LEN \leq 16$$
$$2 \leq TS2\_LEN \leq 8$$
$$(TS1\_LEN + TS2\_LEN + 1) \leq 25$$

The nominal bit time applies when a synchronization edge falls within the t_{SYNC_SEG} period. The maximum bit time occurs when the synchronization edge falls outside of the t_{SYNC_SEG} period, and the synchronization jump width time is added to perform the resynchronization.

$$\text{nominal bit time} = t_{SYNC\_SEG} + t_{TSEG1} + t_{TSEG2}$$

$$= \frac{(BRPV)(CCD)[1 + (TS1\_LEN) + (TS2\_LEN)]}{F_{OSC}}$$

$$\text{maximum bit time} = t_{SYNC\_SEG} + t_{TSEG1} + t_{TSEG2} + t_{SJW}$$

$$= \frac{(BRPV)(CCD)[1 + (TS1\_LEN) + (TS2\_LEN) + (SJW)]}{F_{OSC}}$$

$$\text{CAN baud rate} = \frac{F_{OSC}}{(BRPV)(CCD)[1 + (TS1\_LEN) + (TS2\_LEN)]}$$

Threefold Bit Sampling:

The DS80C390 supports the ability perform one or three samplings of each bit, based on the SMP bit (CxBT1.7). The single sample mode (SMP=0) is available in all settings and takes one sample during each bit time. The triple sampling mode (SMP=1) samples each bit three times for increased noise immunity. This mode can only be used when the baud rate prescale value (BPRV) is greater than 3.

Bus Rate Timing Example:

The following table shows a few example bit timing settings for common oscillator frequency and baud rate selections. Because of the large number of variables, there are many combinations not shown that can achieve a desired baud rate. There are a number of approaches to determining all the bit timing factors, but this utilizes the most common, i.e., the oscillator frequency and baud rate have already been determined by system constraints.

Additional Bit Timing Examples (assumes CCD=1)

F_{OSC}	Baud rate	BRPV	CCD	t_{QU}	t_{QU} per bit	TS1_LEN	TS2_LEN	SJW	SMP=1 Permitted?
40 MHz	1 Mbps	2	2	100 ns	10	5	4	3	NO
	500 kbps	4	2	200 ns	10	5	4	3	YES
	250 kbps	5	2	250 ns	16	10	5	4	YES
	125 kbps	10	2	500 ns	16	10	5	4	YES
16 MHz	1 Mbps	1	2	125 ns	8	4	3	4	NO
	500 kbps	1	2	125 ns	16	10	5	4	NO
	250 kbps	2	2	250 ns	16	10	5	4	NO
	125 kbps	4	2	500 ns	16	10	5	4	YES
8 MHz	1 Mbps	1	1	125 ns	8	4	3	2	NO
	500 kbps	1	1	125 ns	16	10	5	4	NO
	250 kbps	1	1	250 ns	16	10	5	4	NO
	125 kbps	2	2	500 ns	16	10	5	4	NO

As an aid to understanding, the following is an explanation of how the table row illustrating an oscillator frequency of 16 MHz and a CAN baud rate of 125 kbps is derived.

Various combinations of BRPV are selected until one is located that meets the "t_{QU} per bit" criteria, i.e., an integer value less than 24. Selecting BRPV=4, the previously described equations state that there should be 16 t_{QU} per bit. That leaves 16-1 or 15 t_{QU} remaining for TS1_LEN and TS2_LEN, which are arbitrarily assigned as shown. Because BRPV > 3, the triple sampling feature (SMP=1) may be used if desired.

SECTION 19: ARITHMETIC ACCELERATOR

The DS80C390 incorporates an arithmetic accelerator which performs 32- and 16-bit calculations while maintaining 8051 software compatibility. Math operations are performed by sequentially loading three special registers. The mathematical operation is determined by the sequence in which three dedicated SFRs (MA, MB and MC) are accessed, eliminating the need for a special step to choose the operation. The arithmetic accelerator has four functions: multiply, divide, shift right/left, and normalize. The normalize function facilitates the conversion of 4-byte unsigned binary integers into floating point format. An integral 40-bit accumulator, described later, supports multiply-and-add and divide-and-add operations. The following table shows the operations supported by the math accelerator and their time of execution.

ARITHMETIC ACCELERATOR EXECUTION TIMES TABLE 19-1

Operation	Result	Execution Time
32-bit/16-bit divide	32-bit quotient, 16-bit remainder	36 t_{CLCL}
16-bit/16-bit divide	16-bit quotient, 16-bit remainder	24 t_{CLCL}
16-bit/16-bit multiply	32-bit product	24 t_{CLCL}
32-bit shift left/right	32-bit result	36 t_{CLCL}
32-bit normalize	32-bit mantissa, 5 bit exponent	36 t_{CLCL}

The following is a brief summary of the bits and registers used in conjunction with arithmetic acceleration operations. Please consult the SFR listing in Section 4for a complete description of all these registers.

$\overline{\text{LSHIFT}}$ MCNT0.7 — **Left Shift.** This bit determines whether shift operations proceed from LSb to MSb or vice versa.

CSE MCNT0.6 — **Circular Shift Enable.** This bit determines whether shift operations will wrap between the LSb and MSb.

SCE MCNT0.5 — **Shift Carry Enable.** This bit determines whether the arithmetic accelerator carry bit is included in the shift process.

MAS4-0 MCNT0.4-0. — **Multiplier Register Shift Bits.** When performing a shift operation, these bits determine how many shifts to perform. Following a normalize operation, these bits will contain indicate the number shifts performed.

MST MCNT1.7 — **Multiply/Accumulate Status Flag.** This bit serves as a busy flag for the arithmetic accumulator operations.

MOF MCNT1.6 — **Multiply Overflow Flag.** This bit is set when a divide by zero or when the result of a calculation exceeds FFFFh.

SCB MCNT1.5 — **Shift Carry Bit.** This bit serves as the carry bit during arithmetic accelerator shift operations when SCE=1. This bit must be cleared (or set) via software as desired before each new shift operation.

CLM MCNT1.4 — **Clear Accelerator Registers.** Setting this bit clears the MA, MB, and MC registers.

MA — **Multiplier A Register.** This register is used as both a source and result register for

MA.7-0	various arithmetic accelerator functions.
MB MB.7-0	**Multiplier B Register.** This register is used as both a source and result register for various arithmetic accelerator functions.
MC MC.7-0	**Multiplier C Register.** This register serves as the 40-bit accumulator of the arithmetic accelerator.

The following procedures illustrate how to use the arithmetic accelerator. The MA and MB registers must be loaded and read in the order shown for proper operation, although accesses to any other registers can be performed between access to the MA or MB registers. An access to the MA, MB, or MC registers out of sequence will corrupt the operation, requiring the software to clear the MST bit to restart the math accelerator state machine.

Divide (32/16 or 16/16)

The divide operation utilizes a 32 or 16 bit dividend and a 16-bit divisor. The dividend is loaded into MA (four bytes in the case of a 32-bit dividend, 2 bytes for a 16-bit dividend) and the 16-bit divisor is loaded into MB. The quotient is stored in MA and the remainder in MB. The optional test of the MOF bit can be performed to detect a divide by zero operation if software has not previously checked for a non-zero divisor.

1. Load MA with dividend LSB.
2. *Load MA with dividend LSB+1**
3. *Load MA with dividend LSB+2**
4. Load MA with dividend MSB.
5. Load MB with divisor LSB.
6. Load MB with divisor MSB.
7. Poll the MST bit until cleared (9 machine cycles).
8. Check MOF bit (MCNT1.6) to see if divide by zero occurred. (optional)
9. Read MA to retrieve the quotient MSB.
10. Read MA to retrieve the quotient LSB+2.
11. Read MA to retrieve the quotient LSB+1.
12. Read MA to retrieve the quotient LSB.
13. Read MB to retrieve the remainder MSB.
14. Read MB to retrieve the remainder LSB.

*Steps 2 and 3 not performed for 16 bit dividend.

Multiply (16x16)

This function multiplies two 16-bit values in MA and MB and places the 32-bit product into MA. If the product exceeds FFFFh then the Multiply Overflow Flag (MOF) will be set

1. Load MB with multiplier LSB.
2. Load MB with multiplier MSB.
3. Load MA with multiplicand LSB.
4. Load MA with multiplicand MSB.
5. Poll the MST bit until cleared (6 machine cycles).
6. Read MA for product MSB.
7. Read MA for product LSB+2.
8. Read MA for product LSB+1.
9. Read MA for product LSB.
10. Check MOF bit (MCNT1.6) to see if product exceeded FFFFh. (optional)

Shift right/left

The shift function rotates the 32 bits of the MA register as directed by the control bits of the MCNT0 register. MA will contain the shifted results following the operation. Note that the multiplier register shift bits (MCNT.4-0) must be set to a nonzero value or the Normalize function will be performed instead of the desired shift operation.

1. Load MA with data LSB.
2. Load MA with data LSB+1.
3. Load MA with data LSB+2.
4. Load MA with data MSB.
5. Configure MCNT0 register as required.
6. Poll the MST bit until cleared. (9 machine cycles)
7. Read MA for result MSB.
8. Read MA for result LSB+2.
9. Read MA for result LSB+1.
10. Read MA for result LSB.

Normalize

The normalize function is used to convert four byte unsigned binary integers into floating point format by removing all leading zeros by shift left operations. Following the operation MA will contain the normalized value (mantissa) and the MAS4-0 bits will contain the number of shifts performed (characteristic).

1. Load MA with data LSB.
2. Load MA with data LSB+1.
3. Load MA with data LSB+2.
4. Load MA with data MSB.
5. Write 00000b to the MAS4-0 bits in the MCNT0 register.
6. Poll the MST bit until cleared. (9 machine cycles)
7. Read MA for mantissa MSB.
8. Read MA for mantissa LSB+2.
9. Read MA for mantissa LSB+1.
10. Read MA for mantissa LSB.
11. Read MAS4-0 to determine the number of shifts performed.

40-BIT ACCUMULATOR

The accelerator also incorporates an automatic accumulator function, permitting the implementation of multiply-and-accumulate and divide-and-accumulate functions without any additional delay. Each time the accelerator is used for a multiply or divide operation, the result is transparently added to a 40-bit accumulator. This can greatly increase speed of DSP and other high-level math operations.

The accumulator can be accessed any time the Multiply/Accumulate Status Flag (MCNT1;D2h) is cleared. The accumulator is initialized by performing five writes to the Multiplier C Register (MC;D5h), LSB first. The 40-bit accumulator can be read by performing five reads of the Multiplier C Register, MSB first.