THE DATASHEET OF

DS80C390-FNR+

www.win-source.net 0086-755-83957316

DS80C390
Dual CAN High-Speed Microprocessor

GENERAL DESCRIPTION

The DS80C390 is a fast 8051-compatible microprocessor with dual CAN 2.0B controllers. The redesigned processor core executes 8051 instructions up to 3X faster than the original for the same crystal speed. The DS80C390 supports a maximum crystal speed of 40MHz, resulting in apparent execution speeds of 100MHz (approximately 2.5X). An optional internal frequency multiplier allows the microprocessor to operate at full speed with a reduced crystal frequency, reducing EMI. A hardware math accelerator further increases the speed of 32-bit and 16-bit multiply and divide operations as well as high-speed shift, normalization, and accumulate functions.

The *High-Speed Microcontroller User's Guide* and *High-Speed Microcontroller User's Guide: DS80C390 Supplement* must be used in conjunction with this data sheet. **Download both at: www.maxim-ic.com/microcontrollers.**

APPLICATIONS

Industrial Controls
Factory Automation
Medical Equipment
Agricultural Equipment
Gaming Equipment
Heating, Ventilation, and Air Conditioning

FEATURES

- **80C52 Compatible**
- **High-Speed Architecture**
- **4kB Internal SRAM Usable as Program/ Data/Stack Memory**
- **Enhanced Memory Architecture**
- **Two Full-Function CAN 2.0B Controllers**
- **Two Full-Duplex Hardware Serial Ports**
- **Programmable IrDA Clock**
- **High Integration Controller**
- **16 Interrupt Sources with Six External**
- **Available in 64-Pin LQFP, 68-Pin PLCC**

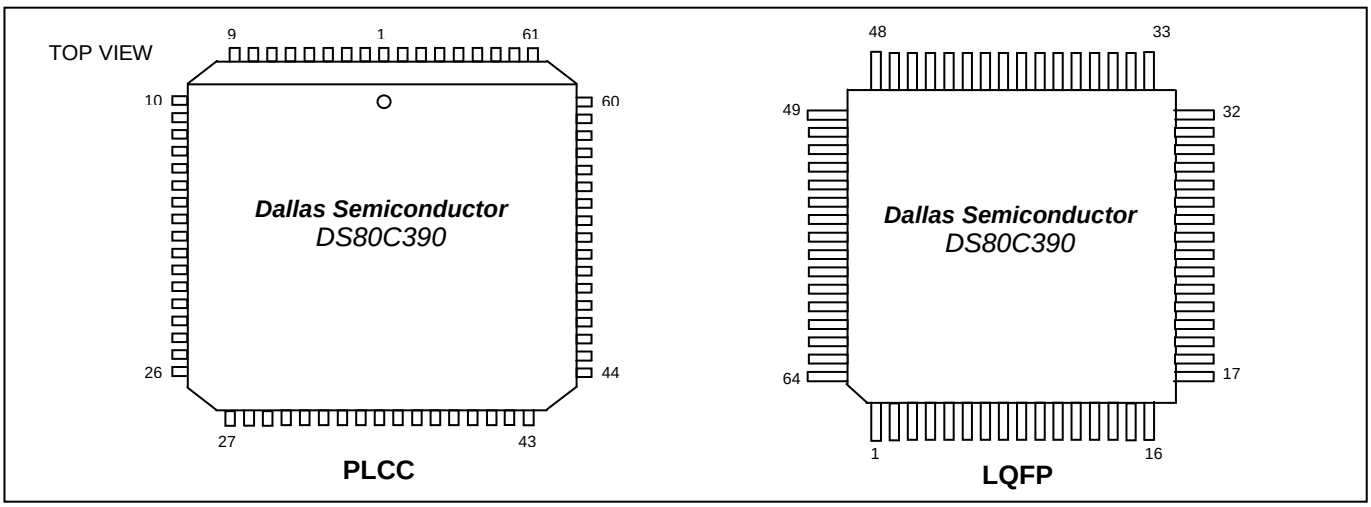
See page 29 for a complete list of features.

ORDERING INFORMATION

PART	TEMP RANGE	PIN-PACKAGE
DS80C390-QCR	0°C to +70°C	68 PLCC
DS80C390-QCR+	0°C to +70°C	68 PLCC
DS80C390-QNR	-40°C to +85°C	68 PLCC
DS80C390-QNR+	-40°C to +85°C	68 PLCC
DS80C390-FCR	0°C to +70°C	64 LQFP
DS80C390-FCR+	0°C to +70°C	64 LQFP
DS80C390-FNR	-40°C to +85°C	64 LQFP
DS80C390-FNR+	-40°C to +85°C	64 LQFP

+Denotes a lead(Pb)-free/RoHS-compliant device.

PIN CONFIGURATIONS



Note: *Some revisions of this device may incorporate deviations from published specifications known as errata. Multiple revisions of any device may be simultaneously available through various sales channels. For information about device errata, click here: www.maxim-ic.com/errata.*

ABSOLUTE MAXIMUM RATINGS

Voltage Range on Any Pin Relative to Ground……-0.3V to (V_{CC} + 0.5V)
Voltage Range on V_{CC} Relative to Ground……-0.3V to +6.0V
Operating Temperature Range……..-40°C to +85°C
Storage Temperature Range……...-55°C to +125°C
Soldering Temperature…..……..See IPC/JEDEC J-STD-020

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to the absolute maximum rating conditions for extended periods may affect device reliability.

DC ELECTRICAL CHARACTERISTICS (Note 10)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS
Supply Voltage	V_{CC}	V_{RST}	5.0	5.5	V
Power-Fail Warning	V_{PFW}	4.10	4.38	4.60	V
Minimum Operating Voltage	V_{RST}	3.85	4.13	4.35	V
Supply Current, Active Mode (Note 1)	I_{CC}		80	150	mA
Supply Current, Idle Mode (Note 2)	I_{IDLE}		40	75	mA
Supply Current, Stop Mode (Note 3)	I_{STOP}		1	120	µA
Supply Current, Stop Mode, Bandgap Enabled (Note 3)	I_{SPBG}		150	350	µA
Input Low Level	V_{IL}	-0.5		+0.8	V
Input High Level	V_{IH}	2.0		V_{CC} +0.5	V
Input High Level for XTAL1, RST	V_{IH2}	0.7 x V_{CC}		V_{CC} +0.5	V
Output Low Voltage for Port 1, 3, 4, 5 at I_{OL} = 1.6mA	V_{OL1}			0.45	V
Output Low Voltage for Port 0, 1, 2, 4, 5, $\overline{RD}$, $\overline{WR}$, $\overline{RSTOL}$, $\overline{PSEN}$, and ALE at I_{OL} = 3.2mA (Note 5)	V_{OL2}			0.45	V
Output High Voltage for Port 1, 3, 4, 5 at I_{OH} = -50µA (Note 4)	V_{OH1}	2.4			V
Output High Voltage for Port 1, 3, 4, 5 at I_{OH} = -1.5mA (Note 6)	V_{OH2}	2.4			V
Output High Voltage for Port 0, 1, 2, 4, 5, $\overline{RD}$, $\overline{WR}$, $\overline{RSTOL}$, $\overline{PSEN}$, and ALE at I_{OH} = -8mA (Note 5, 7)	V_{OH3}	2.4			V
Input Low Current for Port 1, 3, 4, 5 at 0.45V (Note 8)	I_{IL}			-55	µA
Logic 1 to 0 Transition Current for Port 1, 3, 4, 5 (Note 9)	I_{T1}			-650	µA
Input Leakage Current for Port 0 (Input Mode Only)	I_L	-300		+300	µA
RST Pulldown Resistance	R_{RST}	50		170	kΩ

Note 1: Active current measured with 40MHz clock source on XTAL1, V_{CC} = RST = 5.5V, all other pins disconnected.
Note 2: Idle mode current measured with 40MHz clock source on XTAL1, V_{CC}= 5.5V, RST = $\overline{EA}$ = V_{SS}, all other pins disconnected.
Note 3: Stop mode current measured with XTAL1 = RST = $\overline{EA}$ = V_{SS}, V_{CC} = 5.5V, all other pins disconnected.
Note 4: RST = V_{CC}. This condition mimics operation of pins in I/O mode.
Note 5: Applies to port pins when they are used to address external memory or as CAN interface signals.
Note 6: This measurement reflects the port during a 0-to-1 transition in I/O mode. During this period a one-shot circuit drives the ports hard for two clock cycles. If a port 4 or 5 pin is functioning in memory mode with pin state of 0 and the SFR bit contains a 1, changing the pin to an I/O mode (by writing to P4CNT) will not enable the 2-cycle strong pullup. During Stop or Idle mode the pins switch to I/O mode, and so port 2 and port 1 (in nonmultiplexed mode) will not exhibit the 2-cycle strong pullup when entering Stop or Idle mode.
Note 7: Port 3 pins 3.6 and 3.7 have a stronger than normal pullup drive for one oscillator period following the transition of either the $\overline{RD}$ or $\overline{WR}$ from a 0-to-1 transition.
Note 8: This is the current required from an external circuit to hold a logic low level on an I/O pin while the corresponding port latch bit is set to 1. This is only the current required to *hold* the low level; transitions from 1 to 0 on an I/O pin also have to overcome the transition current.
Note 9: Ports 1(in I/O mode), 3, 4, and 5 source transition current when being pulled down externally. It reaches its maximum at approximately 2V.
Note 10: Specifications to -40°C are guaranteed by design and not production tested.

AC ELECTRICAL CHARACTERISTICS—(MULTIPLEXED ADDRESS/DATA BUS) (Note 10, Note 11)

PARAMETER	SYMBOL	CONDITIONS	40MHz		VARIABLE CLOCK		UNITS
			MIN	MAX	MIN	MAX	
Oscillator Frequency	$1 / t_{CLCL}$	External oscillator	0	40	0	40	MHz
		External crystal	1	40	1	40	
ALE Pulse Width	t_{LHLL}				$0.375\ t_{MCS} - 5$		ns
Port 0 Instruction Address or $\overline{CE0–4}$ Valid to ALE Low	t_{AVLL}				$0.125\ t_{MCS} - 5$		ns
Address Hold After ALE Low	t_{LLAX1}				$0.125\ t_{MCS} - 5$		ns
ALE Low to Valid Instruction In	t_{LLIV}					$0.625\ t_{MCS} - 20$	ns
ALE Low to $\overline{PSEN}$ Low	t_{LLPL}				$0.125\ t_{MCS} - 5$		ns
$\overline{PSEN}$ Pulse Width	t_{PLPH}				$0.5\ t_{MCS} - 8$		ns
$\overline{PSEN}$ Low to Valid Instruction In	t_{PLIV}					$0.5\ t_{MCS} - 20$	ns
Input Instruction Hold After $\overline{PSEN}$	t_{PXIX}		0		0		ns
Input Instruction Float After $\overline{PSEN}$	t_{PXIZ}					$0.25\ t_{MCS} - 5$	ns
Port 0 Address to Valid Instruction In	t_{AVIV1}					$0.75\ t_{MCS} - 22$	ns
Port 2, 4 Address to Valid Instruction In	t_{AVIV2}					$0.875\ t_{MCS} - 30$	ns
$\overline{PSEN}$ Low to Address Float	t_{PLAZ}			0		0	ns

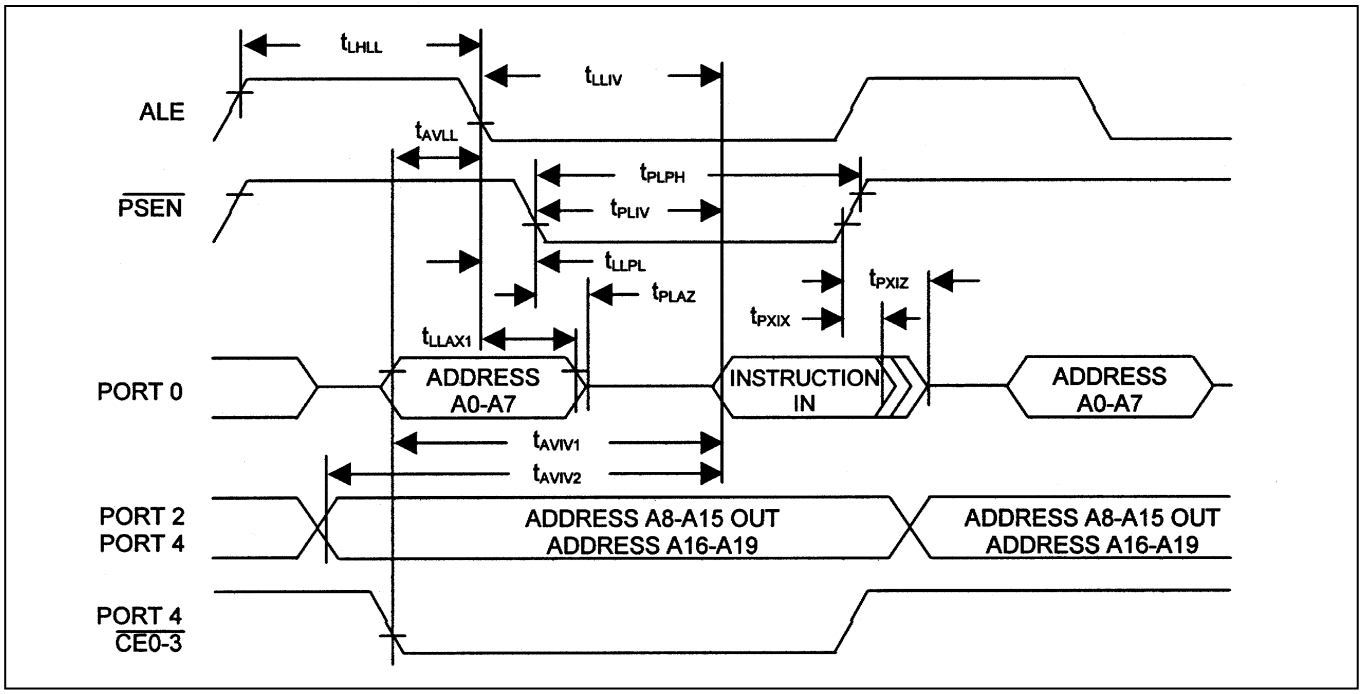
Note 11: All parameters apply to both commercial and industrial temperature operation unless otherwise noted. The value t_{MCS} is a function of the machine cycle clock in terms of the processor's input clock frequency. These relationships are described in the *Stretch Value Timing* table. All signals characterized with load capacitance of 80pF except Port 0, ALE, $\overline{PSEN}$, $\overline{RD}$, and $\overline{WR}$ with 100pF. Interfacing to memory devices with float times (turn off times) over 25ns can cause bus contention. This does not damage the parts, but causes an increase in operating current. Specifications assume a 50% duty cycle for the oscillator. Port 2 and ALE timing changes in relation to duty cycle variation. Some AC timing characteristic drawings contain references to the CLK signal. This waveform is provided to assist in determining the relative occurrence of events and cannot be used to determine the timing of signals relative to the external clock. AC timing is characterized and guaranteed by design but is not production tested.

AC SYMBOLS

The DS80C390 uses timing parameters and symbols similar to the original 8051 family. The following list of timing symbols is provided as an aid to understanding the timing diagrams.

SYMBOL	FUNCTION
t	Time
A	Address
C	Clock
CE	Chip Enable
D	Input Data
H	Logic Level High
L	Logic Level Low
I	Instruction
P	$\overline{\text{PSEN}}$
Q	Output Data
R	$\overline{\text{RD}}$ Signal
V	Valid
W	$\overline{\text{WR}}$ Signal
X	No longer a valid logic level.
Z	Tri-State

Figure 1. Multiplexed External Program Memory Read Cycle



MOVX CHARACTERISTICS (MULTIPLEXED ADDRESS/DATA BUS) (Note 12)

PARAMETER	SYMBOL	MIN	MAX	UNITS	STRETCH VALUES C_{ST} (MD2:0)
MOVX ALE Pulse Width	t_{LHLL2}	0.375 t_{MCS} - 5		ns	C_{ST} = 0
		0.5 t_{MCS} - 5		ns	1 ≤ C_{ST} ≤ 3
		1.5 t_{MCS} - 10		ns	4 ≤ C_{ST} ≤ 7
Port 0 MOVX Address, $\overline{CE0–4}$, $\overline{PCE0–4}$ Valid to ALE Low	t_{AVLL2}	0.125 t_{MCS} - 5		ns	C_{ST} = 0
		0.25t_{MCS} - 5		ns	1≤ C_{ST} ≤ 3
		1.25 t_{MCS} - 10		ns	4 ≤ C_{ST} ≤ 7
Address Hold After MOVX Read/Write	t_{LLAX2} t_{LLAX3}	0.25t_{MCS}-5		ns	C_{ST} = 0
		0.125 t_{MCS} - 5		ns	1≤ C_{ST} ≤ 3
		1.25 t_{MCS} - 5		ns	4 ≤ C_{ST} ≤ 7
$\overline{RD}$ Pulse Width	t_{RLRH}	0.5 t_{MCS} - 6		ns	C_{ST} = 0
		C_{ST} x t_{MCS} - 10		ns	1 ≤ C_{ST} ≤ 7
$\overline{WR}$ Pulse Width	t_{WLWH}	0.5 t_{MCS} - 6		ns	C_{ST} = 0
		C_{ST} x t_{MCS} - 10		ns	1 ≤ C_{ST} ≤ 7
$\overline{RD}$ Low to Valid Data In	t_{RLDV}		0.5 t_{MCS} - 20	ns	C_{ST} = 0
			C_{ST} x t_{MCS} - 25	ns	1 ≤ C_{ST} ≤ 7
Data Hold After Read	t_{RHDX}	0		ns	
Data Float After Read	t_{RHDZ}		0.25 t_{MCS} - 5	ns	C_{ST} = 0
			0.5t_{MCS} - 5	ns	1 ≤ C_{ST} ≤ 3
			1.5 t_{MCS} - 5	ns	4 ≤ C_{ST} ≤ 7
ALE Low to Valid Data In	t_{LLDV}		0.625 t_{MCS} - 20	ns	C_{ST} = 0
			(C_{ST} + 0.25) x t_{MCS} - 20	ns	1 ≤ C_{ST} ≤ 3
			(C_{ST} + 1.25) x t_{MCS} - 20	ns	4 ≤ C_{ST} ≤ 7
Port 0 Address, Port 4 CE, Port 5 PCE to Valid Data In	t_{AVDV1}		0.75 t_{MCS} - 26	ns	C_{ST} = 0
			(4C_{ST} + 0.5) x t_{MCS} - 30	ns	1≤ C_{ST} ≤ 3
			(4C_{ST} + 2.5) x t_{MCS} - 30	ns	4 ≤ C_{ST} ≤ 7
Port 2, 4 Address to Valid Data In	t_{AVDV2}		0.75 t_{MCS} - 30	ns	C_{ST} = 0
			(4C_{ST} + 0.5) x t_{MCS} - 30	ns	1 ≤ C_{ST} ≤ 3
			(4C_{ST} + 2.5) x t_{MCS} - 30	ns	4 ≤ C_{ST} ≤ 7
ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{LLWL}	0.125 t_{MCS} - 5	0.125 t_{MCS} + 10	ns	C_{ST} =0
		0.25t_{MCS} - 5	0.25t_{MCS} + 10	ns	1 ≤ C_{ST} ≤ 3
		1.25 t_{MCS} - 5	1.25 t_{MCS} + 10	ns	4 ≤ C_{ST} ≤ 7
Port 0 Address, Port 4 CE, Port 5 PCE to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVWL1}	0.25 t_{MCS} - 11		ns	C_{ST} = 0
		0.5t_{MCS} - 11		ns	1 ≤ C_{ST} ≤ 3
		2.5 t_{MCS} - 11		ns	4 ≤ C_{ST} ≤ 7
Port 2, 4 Address to or $\overline{WR}$ Low	t_{AVWL2}	0.375 t_{MCS} - 11		ns	C_{ST} = 0
		0.625t_{MCS} - 11		ns	1 ≤ C_{ST} ≤ 3
		2.625 t_{MCS} - 11		ns	4 ≤ C_{ST} ≤ 7
Data Valid to $\overline{WR}$ Transition	t_{QVWX}	-8		ns	
Data Hold After $\overline{WR}$ High	t_{WHQX}	0.25 t_{MCS} - 8		ns	C_{ST} = 0
		0.5t_{MCS} - 10		ns	1 ≤ C_{ST} ≤ 3
		1.5 t_{MCS} - 10		ns	4 ≤ C_{ST} ≤ 7
$\overline{RD}$ Low to Address Float	t_{RLAZ}		See Note 12		
$\overline{RD}$ or $\overline{WR}$ High to ALE, Port 4 CE or Port 5 PCE High	t_{WHLH}	-5	+10	ns	C_{ST} = 0
		0.25 t_{MCS} - 7	0.25 t_{MCS} + 5	ns	1 ≤ C_{ST} ≤ 3
		1.25 t_{MCS} - 7	1.25 t_{MCS} +10	ns	4 ≤ C_{ST} ≤ 7

Note 12: All parameters apply to both commercial and industrial temperature operation. C_{ST} is the stretch cycle value determined by the MD2:0 bits. t_{MCS} is a time period shown in the *t_{MCS} Time Periods* table. All signals characterized with load capacitance of 80pF except Port 0, ALE, $\overline{PSEN}$, $\overline{RD}$, and $\overline{WR}$ with 100pF. Interfacing to memory devices with float times over 25ns can cause bus contention and an increase in operating current. Specifications assume a 50% duty cycle for the oscillator; port 2 and ALE timing changes in relation to duty cycle variation. Some AC timing characteristic drawings show the CLK signal, provided to determine the relative occurrence of events and not the timing of signals relative to the external clock. During the external addressing mode, weak latches maintain the previously driven value from the processor on Port 0 until Port 0 is overdriven by external memory; and on Port 1, 2 and 4 for one XTAL1 cycle prior to change in output address from Port 1, 2, and 4.

Figure 2. Multiplexed 9-Cycle Address/Data $\overline{CE0\text{-}3}$ MOVX Read/Write Operation

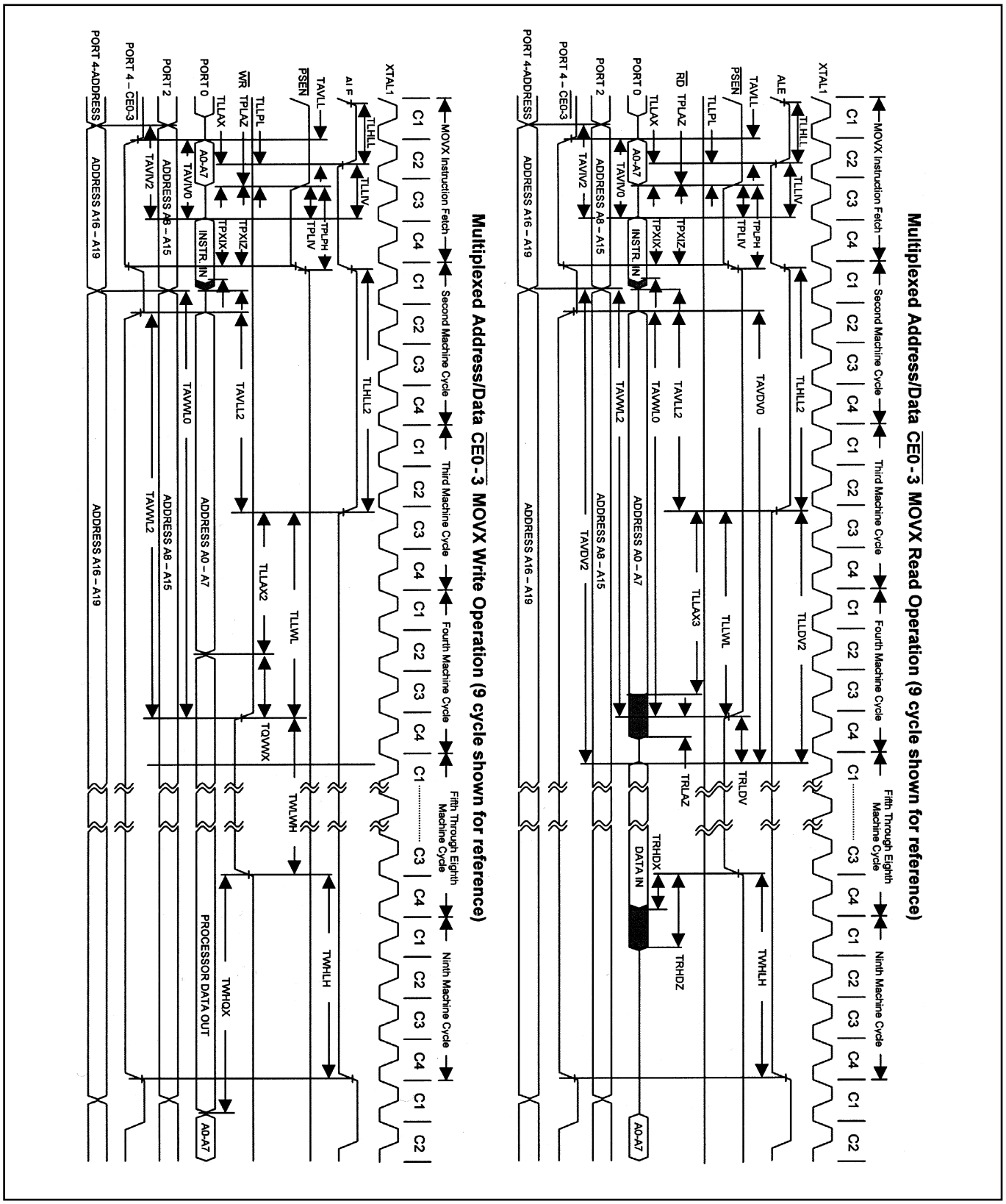


Figure 3. Multiplexed 9-Cycle Address/Data $\overline{\text{PCE0-3}}$ MOVX Read/Write Operation

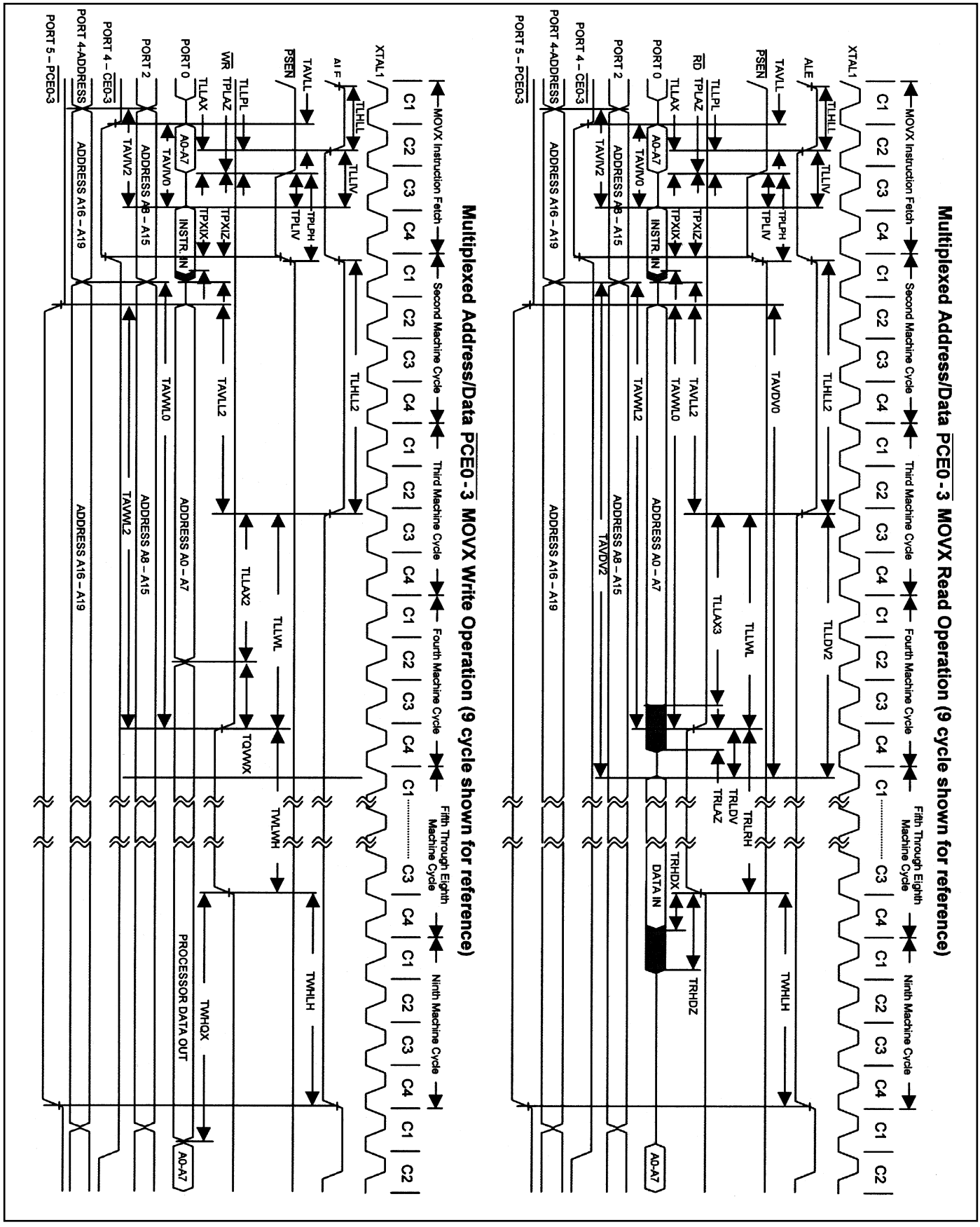


Figure 4. Multiplexed 2-Cycle Data Memory $\overline{\text{PCE0-3}}$ Read or Write

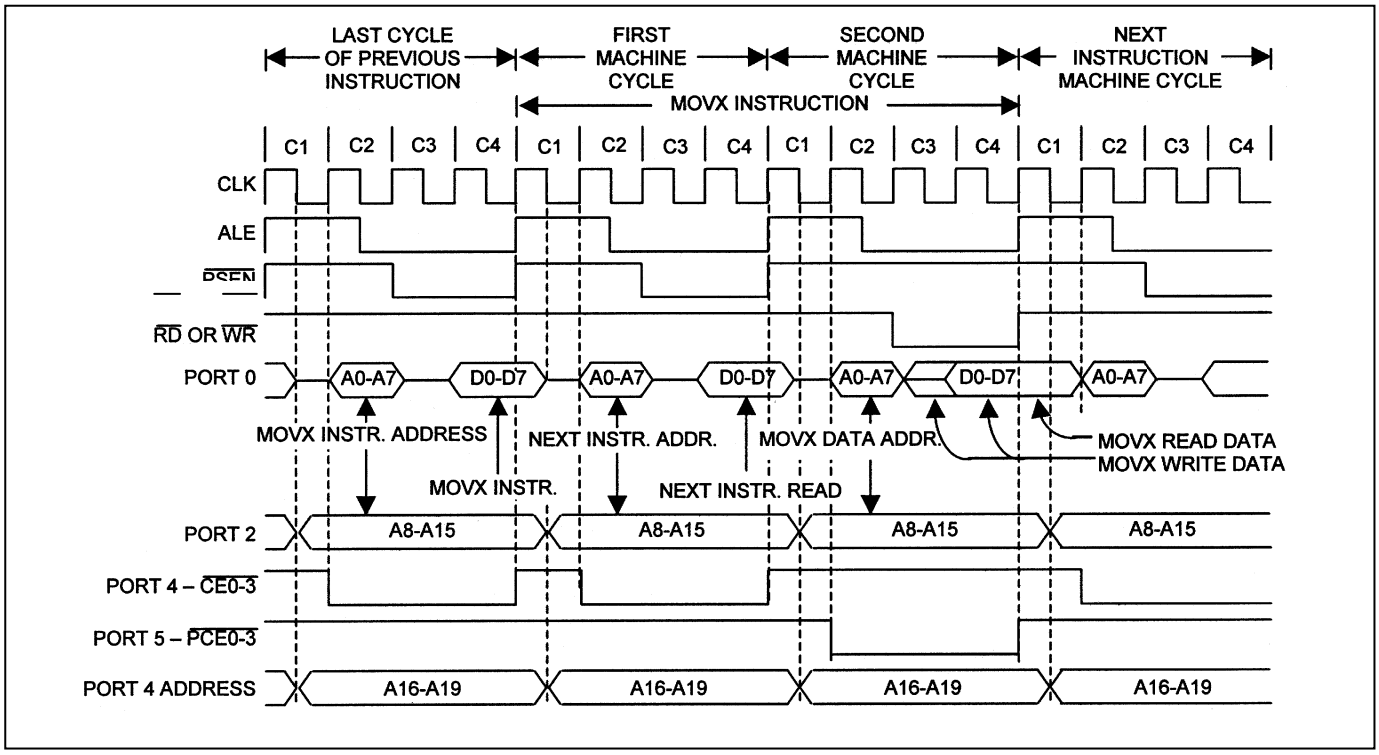


Figure 5. Multiplexed 2-Cycle Data Memory $\overline{\text{CE0-3}}$ Read

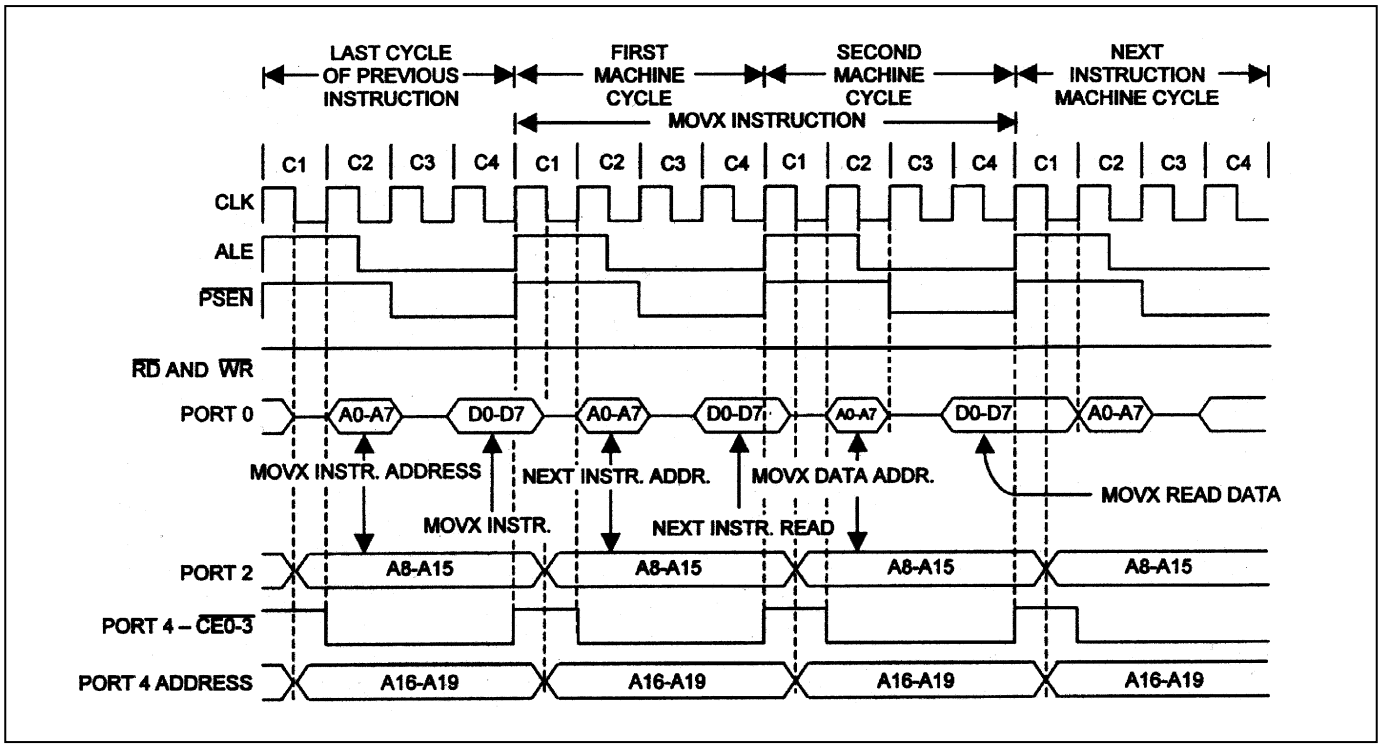


Figure 6. Multiplexed 2-Cycle Data Memory $\overline{\text{CE0-3}}$ Write

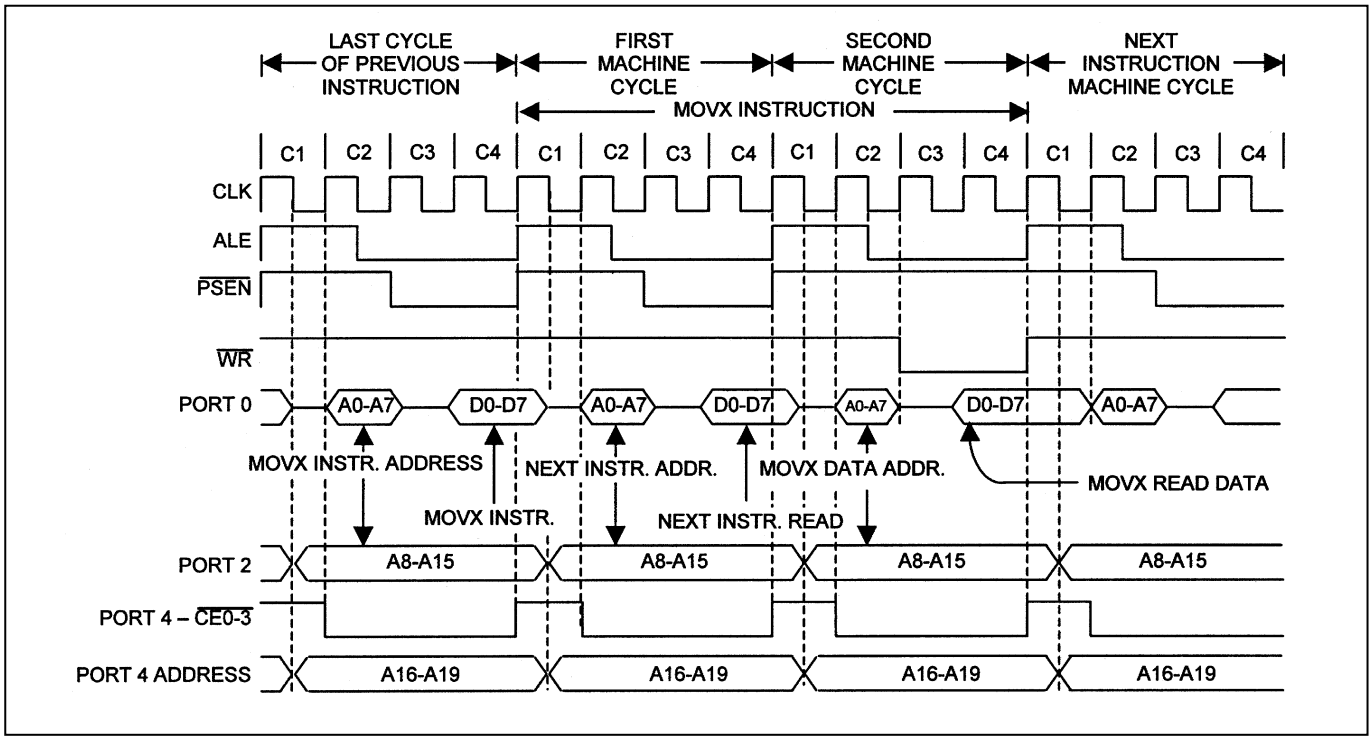


Figure 7. Multiplexed 3-Cycle Data Memory $\overline{\text{PCE0-3}}$ Read or Write

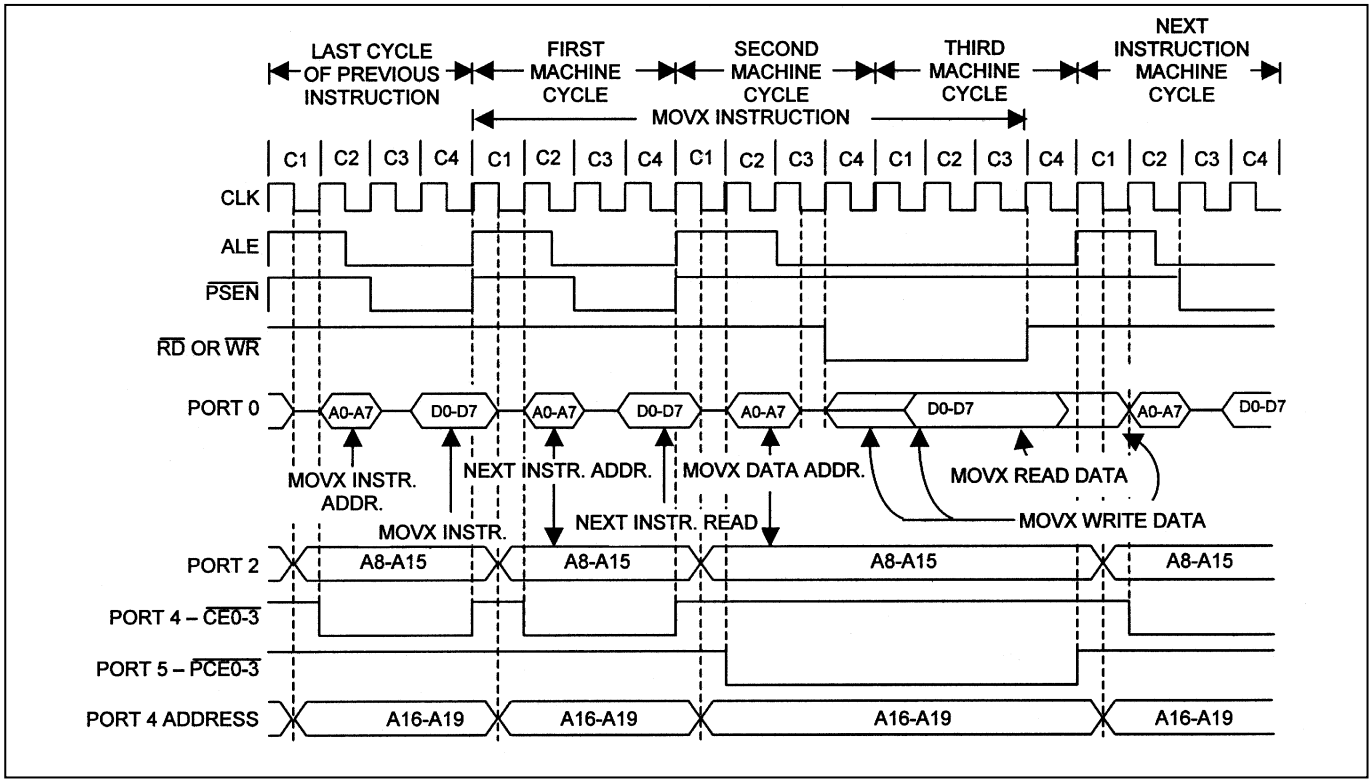


Figure 8. Multiplexed 3-Cycle Data Memory $\overline{CE0\text{-}3}$ Read

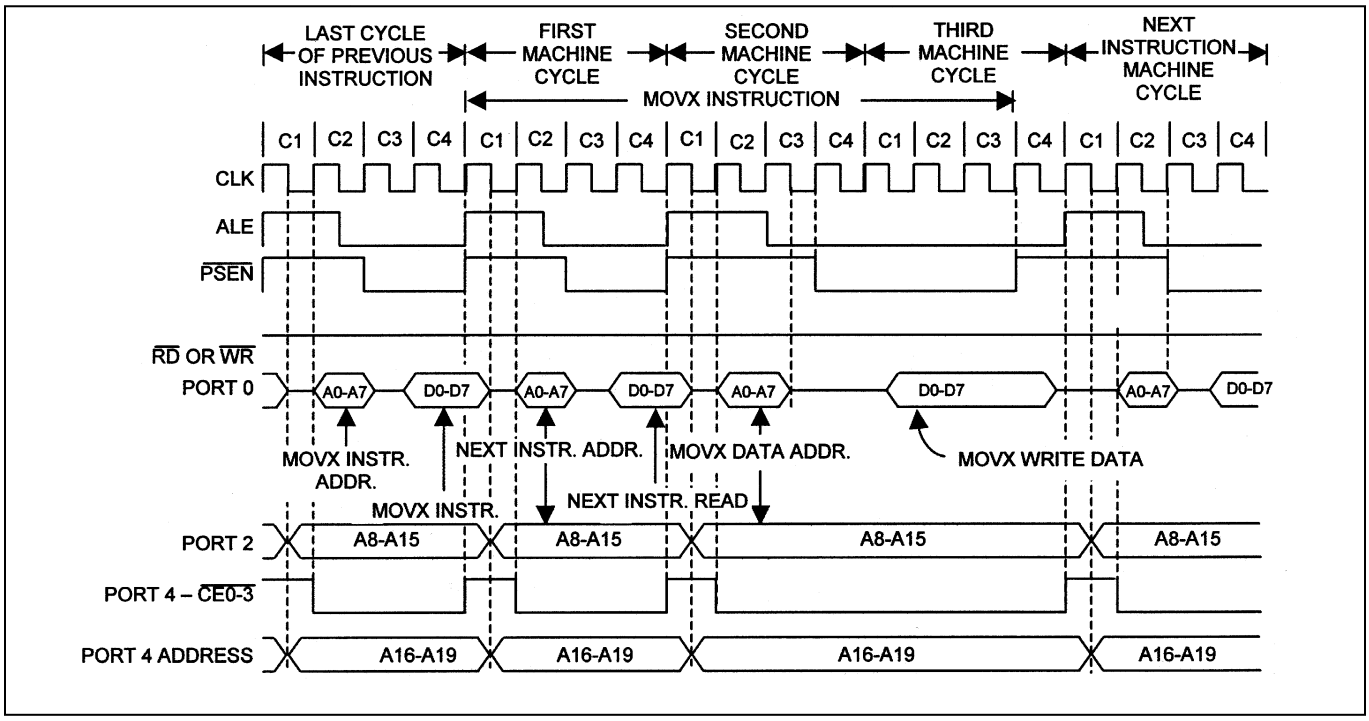


Figure 9. Multiplexed 3-Cycle Data Memory $\overline{CE0\text{-}3}$ Write

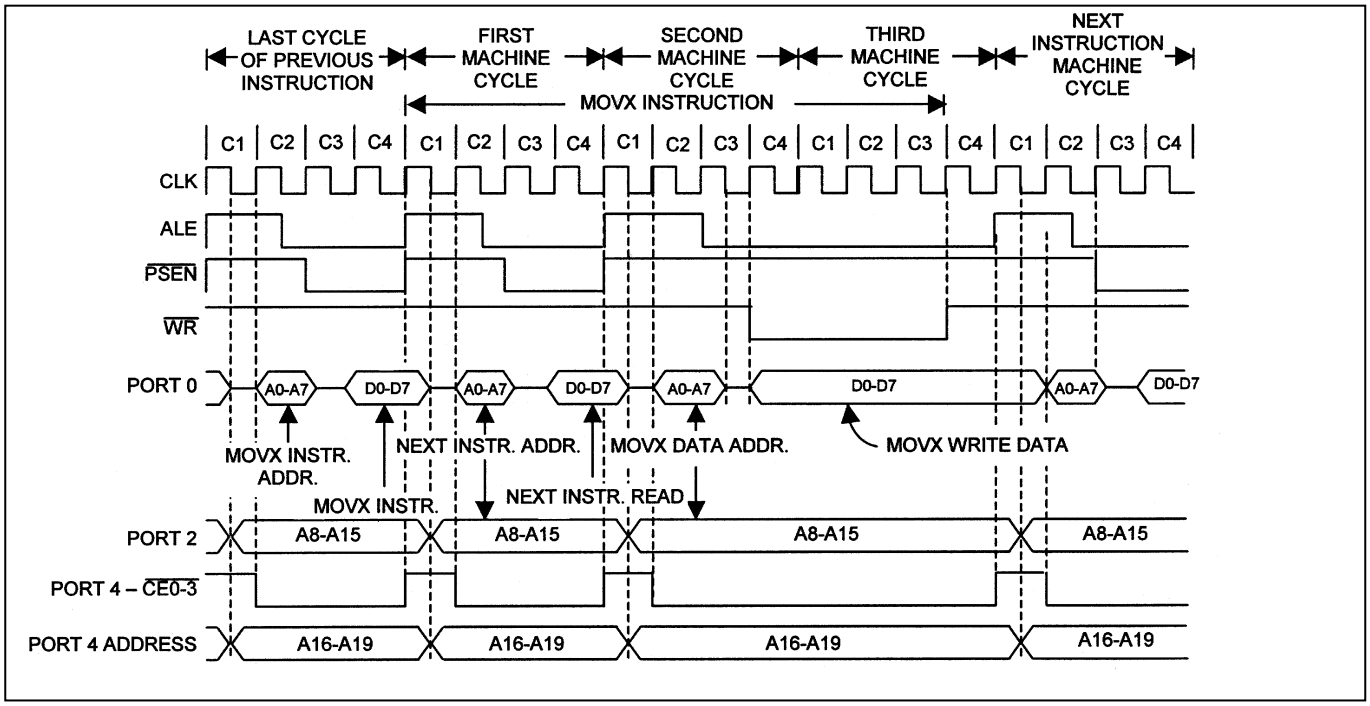


Figure 10. Multiplexed 9-Cycle Data Memory $\overline{\text{PEC0-3}}$ Read or Write

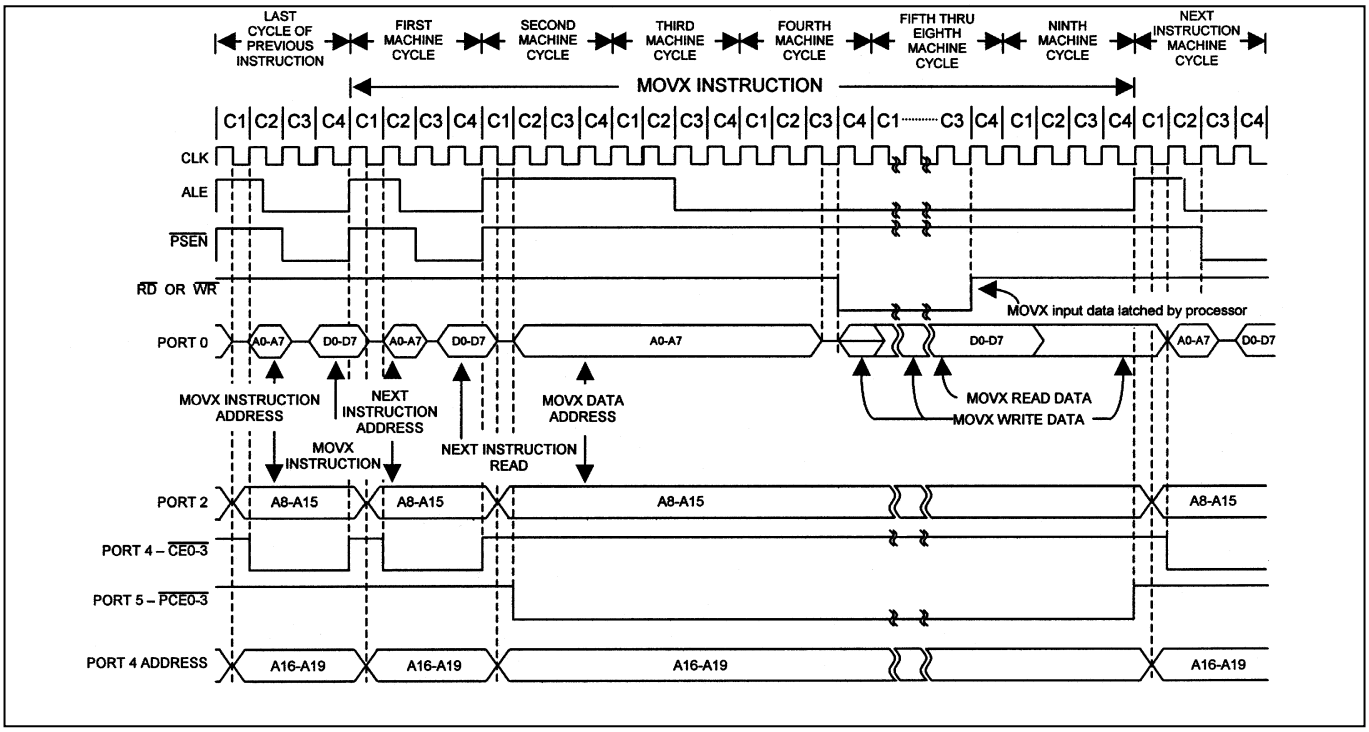


Figure 11. Multiplexed 9-Cycle Data Memory $\overline{\text{CE0-3}}$ Read

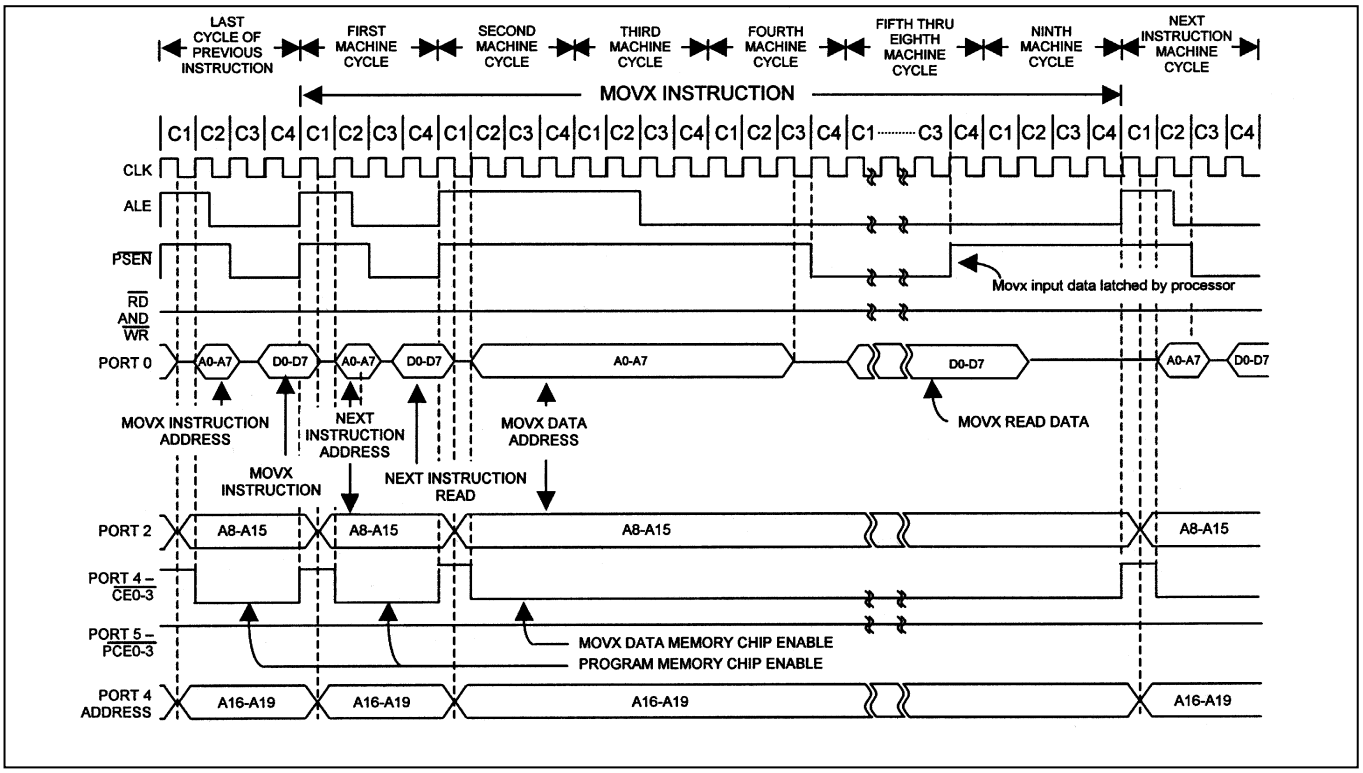
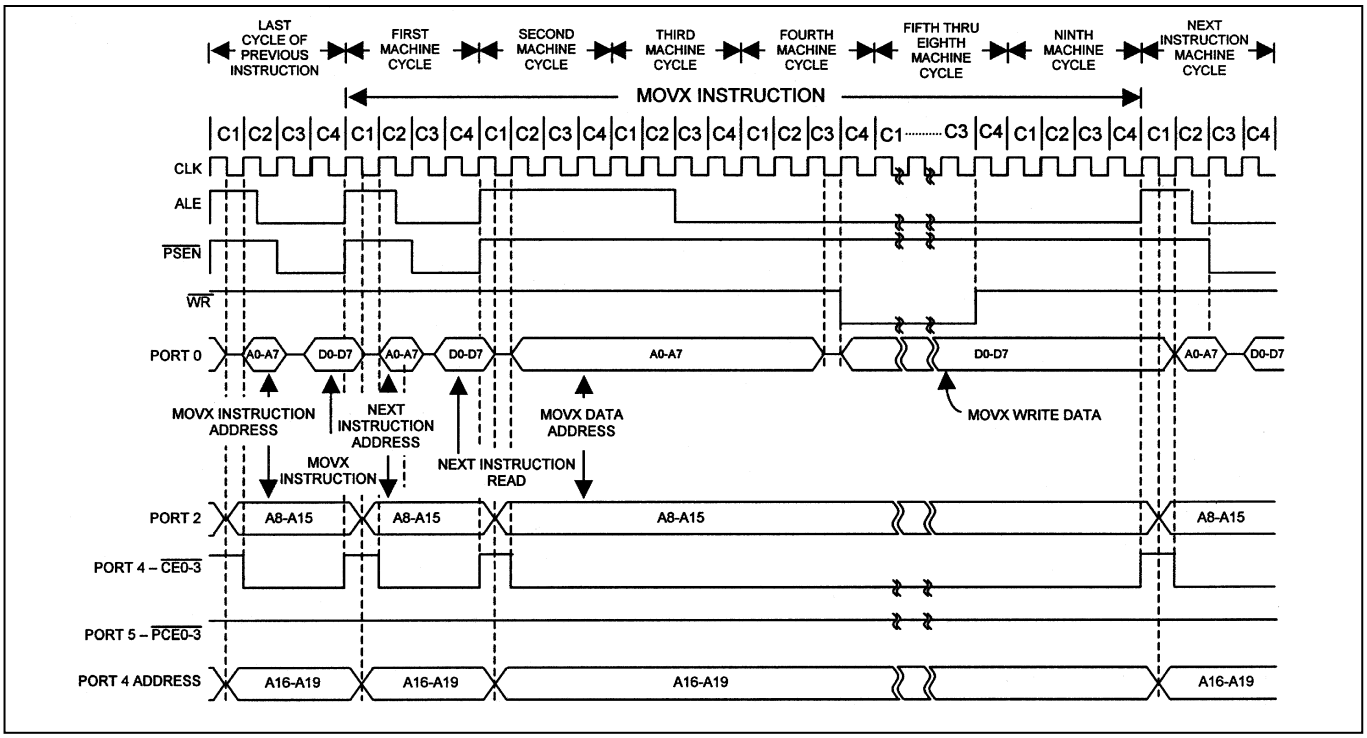


Figure 12. Multiplexed 9-Cycle Data Memory $\overline{CE0\text{-}3}$ Write

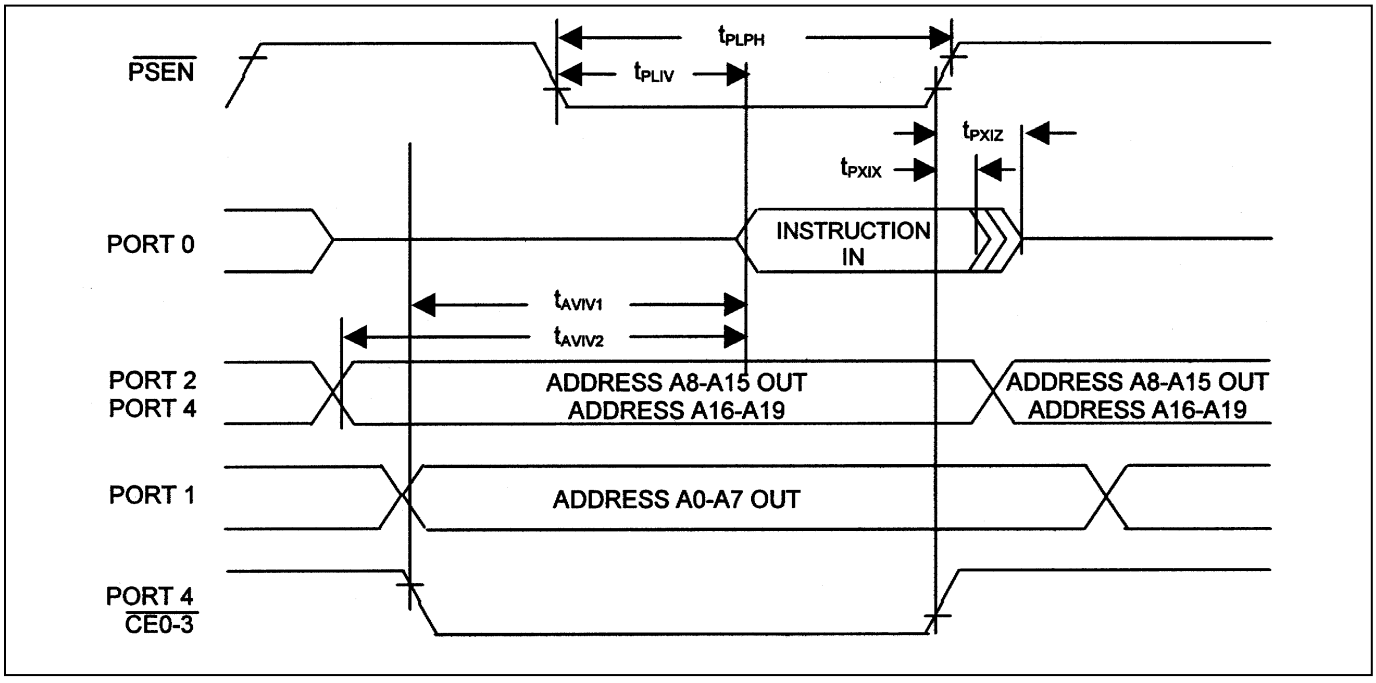


ELECTRICAL CHARACTERISTICS—(NONMULTIPLEXED ADDRESS/DATA BUS) (Note 13)

PARAMETER	SYMBOL	CONDITIONS	40MHz		VARIABLE CLOCK		UNITS
			MIN	MAX	MIN	MAX	
Oscillator Frequency	$1 / t_{CLCL}$	External oscillator	0	40	0	40	MHz
		External crystal	1	40	1	40	
$\overline{PSEN}$ Pulse Width	t_{PLPH}				$0.5\ t_{MCS} - 8$		ns
$\overline{PSEN}$ Low to Valid Instruction In	t_{PLIV}					$0.5\ t_{MCS} - 20$	ns
Input Instruction Hold After $\overline{PSEN}$	t_{PXIX}		0		0		ns
Input Instruction Float After $\overline{PSEN}$	t_{PXIZ}					See MOVX Characteristics	ns
Port 1 Address, Port 4 CE to Valid Instruction In	t_{AVIV1}					$0.75\ t_{MCS} - 22$	ns
Port 2, 4 Address to Valid Instruction In	t_{AVIV2}					$0.875\ t_{MCS} - 30$	ns

Note 13: All parameters apply to both commercial and industrial temperature operation unless otherwise noted. The value t_{MCS} is a function of the machine cycle clock in terms of the processor's input clock frequency. These relationships are described in the *Stretch Value Timing* table. All signals characterized with load capacitance of 80pF except Port 0, ALE, $\overline{PSEN}$, $\overline{RD}$, and $\overline{WR}$ with 100pF. Interfacing to memory devices with float times (turn off times) over 25ns can cause bus contention. This does not damage the parts, but causes an increase in operating current. Specifications assume a 50% duty cycle for the oscillator. Port 2 and ALE timing changes in relation to duty cycle variation. Some AC timing characteristic drawings contain references to the CLK signal. This waveform is provided to assist in determining the relative occurrence of events and cannot be used to determine the timing of signals relative to the external clock.

Figure 13. Nonmultiplexed External Program Memory Read Cycle



MOVX CHARACTERISTICS (NONMULTIPLEXED ADDRESS/DATA BUS)

PARAMETER	SYMBOL	MIN	MAX	UNITS	STRETCH VALUES C_{ST} (MD2:0)
$\overline{RD}$ Pulse Width	t_{RLRH}	0.5 t_{MCS} - 6		ns	C_{ST} = 0
		C_{ST} x t_{MCS} - 6			1 ≤ C_{ST} ≤ 7
$\overline{WR}$ Pulse Width	t_{WLWH}	0.5 t_{MCS} - 6		ns	C_{ST} = 0
		C_{ST} x t_{MCS} - 6			1 ≤ C_{ST} ≤ 7
$\overline{RD}$ Low to Valid Data In	t_{RLDV}		0.5 t_{MCS} - 20	ns	C_{ST} = 0
			C_{ST} x t_{MCS} - 25		1 ≤ C_{ST} ≤ 7
Data Hold After Read	t_{RHDX}	0		ns	
Data Float After Read	t_{RHDZ}		0.125 t_{MCS} - 5	ns	C_{ST} = 0
			0.375t_{MCS} - 5		1 ≤ C_{ST} ≤ 3
			1.375 t_{MCS} - 5		4 ≤ C_{ST} ≤ 7
Port 1 Address, Port 4 CE, Port 5 PCE to Valid Data In	t_{AVDV1}		0.75 t_{MCS} - 26	ns	C_{ST} = 0
			(4C_{ST} + 0.5) x t_{MCS} - 30		1 ≤ C_{ST} ≤ 3
			(4C_{ST} + 2.5) x t_{MCS} - 30		4 ≤ C_{ST} ≤ 7
Port 2, 4 Address to Valid Data In	t_{AVDV2}		0.75 t_{MCS} - 30	ns	C_{ST} = 0
			(4C_{ST} + 0.625) x t_{MCS} - 30		1 ≤ C_{ST} ≤ 3
			(4C_{ST} + 2.625) x t_{MCS} - 30		4 ≤ C_{ST} ≤ 7
Port 0 Address, Port 4 CE, Port 5 PCE to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVWL1}	0.25 t_{MCS} - 11		ns	C_{ST} = 0
		0.5 t_{MCS} - 11			1 ≤ C_{ST} ≤ 3
		2.5 t_{MCS} - 11			4 ≤ C_{ST} ≤ 7
Port 2, 4 Address to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVWL2}	0.375 t_{MCS} - 11		ns	C_{ST} = 0
		0.625t_{MCS} - 11			1 ≤ C_{ST} ≤ 3
		2.625 t_{MCS} - 11			4 ≤ C_{ST} ≤ 7
Data Valid to $\overline{WR}$ Transition	t_{QVWX}	-8		ns	
Data Hold After $\overline{WR}$ High	t_{WHQX}	0.25 t_{MCS} - 8		ns	C_{ST} = 0
		0.5t_{MCS} - 10			1 ≤ C_{ST} ≤ 3
		1.5 t_{MCS} - 10			4 ≤ C_{ST} ≤ 7
$\overline{RD}$ or $\overline{WR}$ High to ALE, Port 4 CE or Port 5 PCE High	t_{WHLH}	-5	10	ns	C_{ST} = 0
		0.25 t_{MCS} - 7	0.25 t_{MCS} + 10		1 ≤ C_{ST} ≤ 3
		1.25 t_{MCS} - 7	1.25 t_{MCS} + 10		4 ≤ C_{ST} ≤ 7

Figure 14. Nonmultiplexed 9-Cycle Address/Data $\overline{CE0\text{-}3}$ MOVX Read/Write Operation

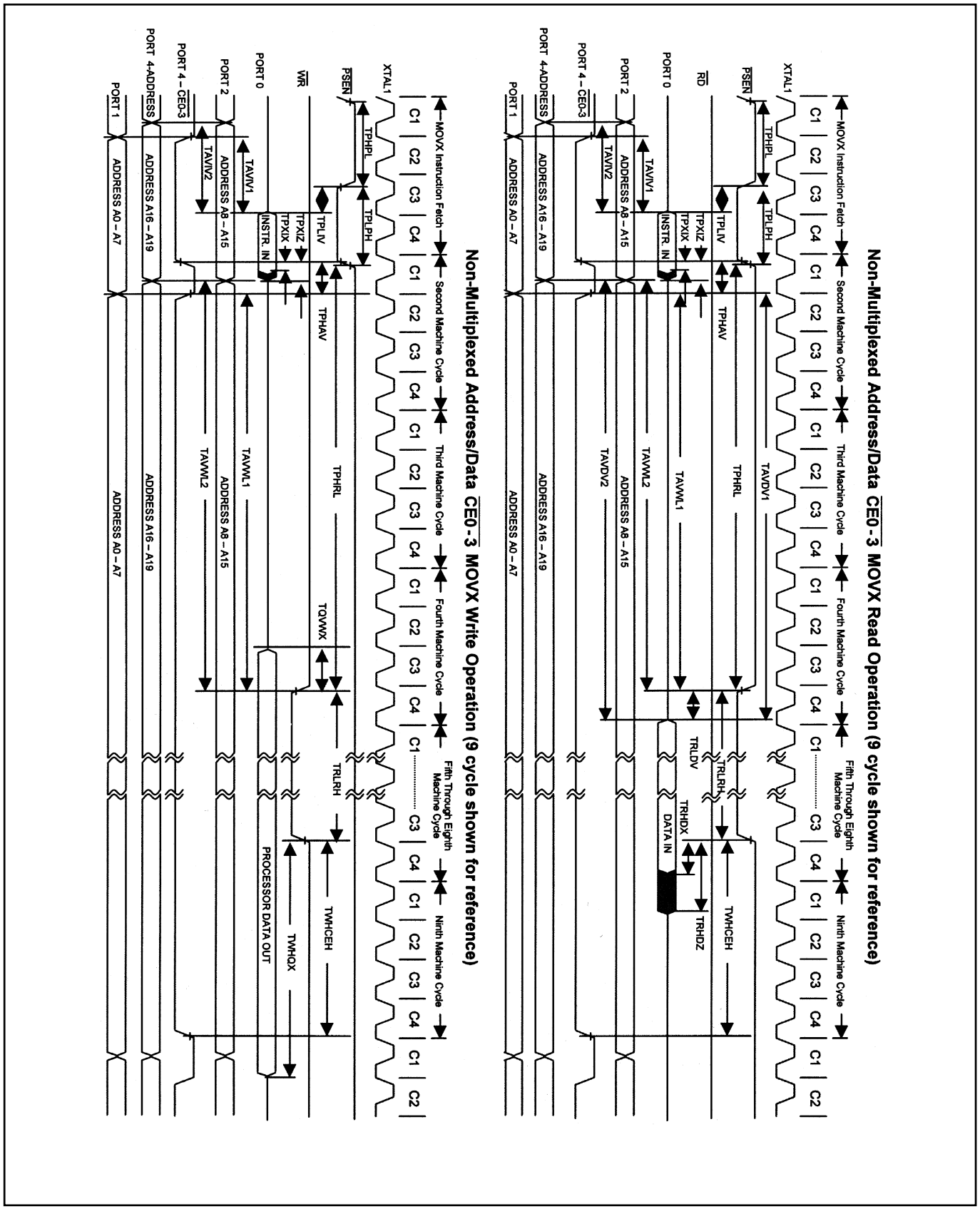


Figure 15. Nonmultiplexed 9-Cycle Address/Data $\overline{\text{PCE0-3}}$ MOVX Read/Write Operation

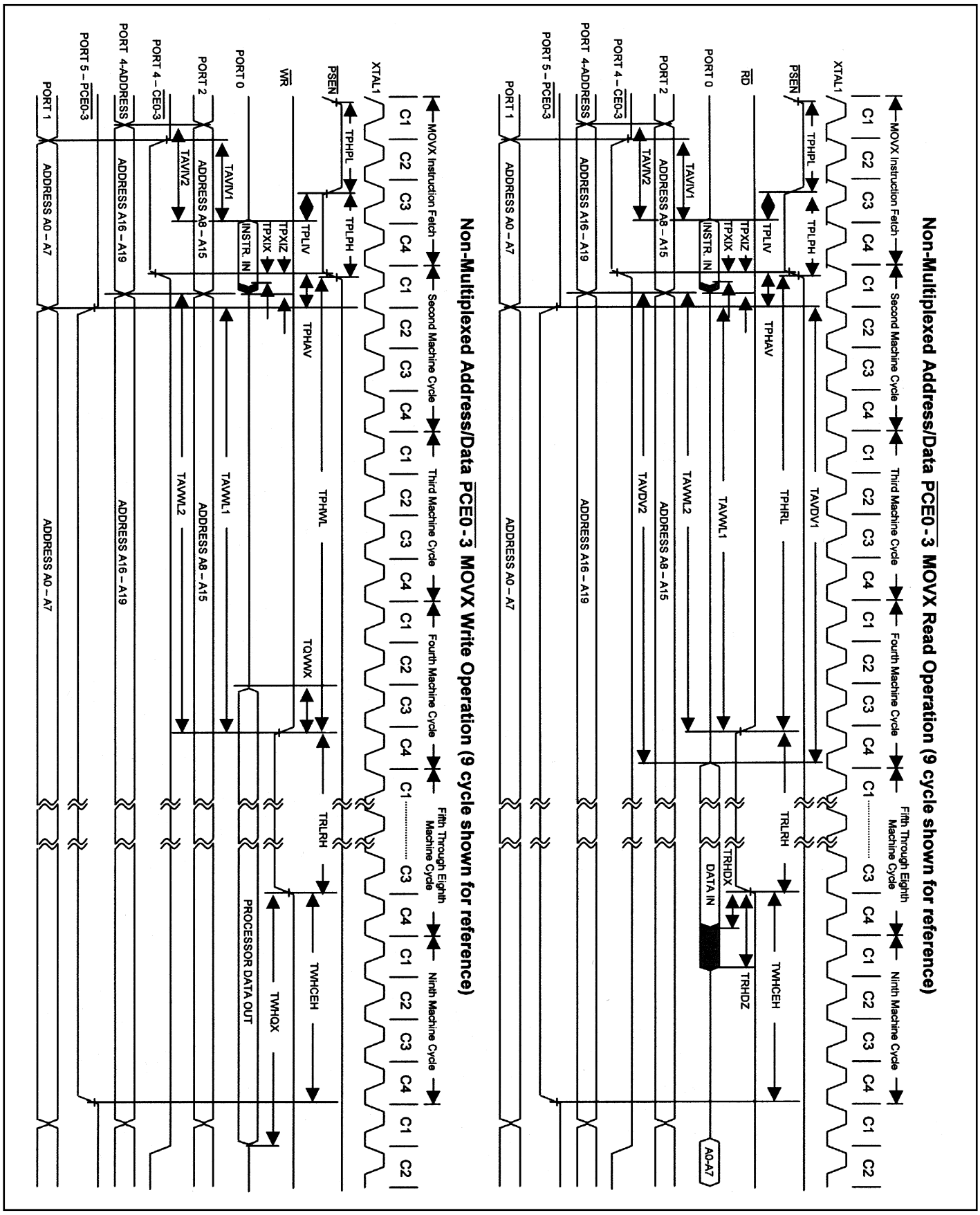


Figure 16. Nonmultiplexed 2-Cycle Data Memory $\overline{\text{PCE0-3}}$ Read or Write

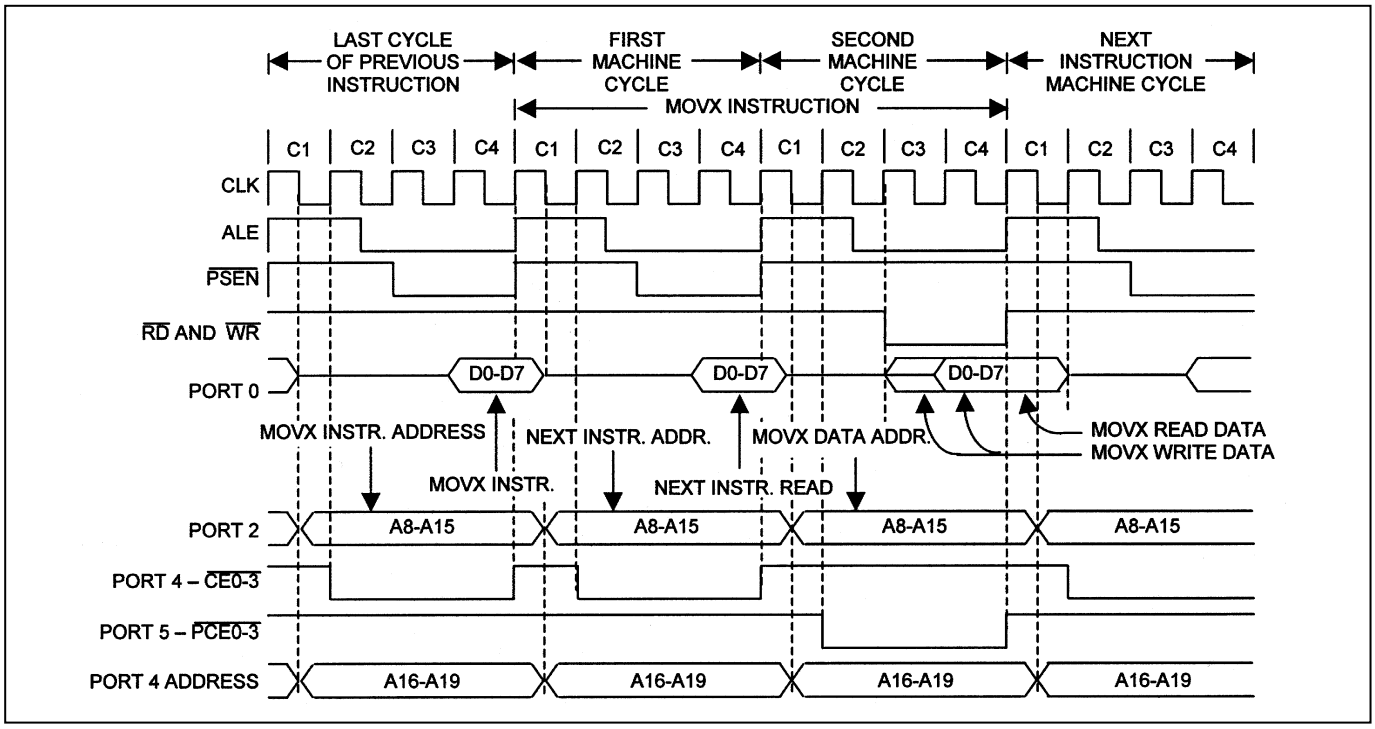


Figure 17. Nonmultiplexed 2-Cycle Data Memory $\overline{\text{CE0-3}}$ Read

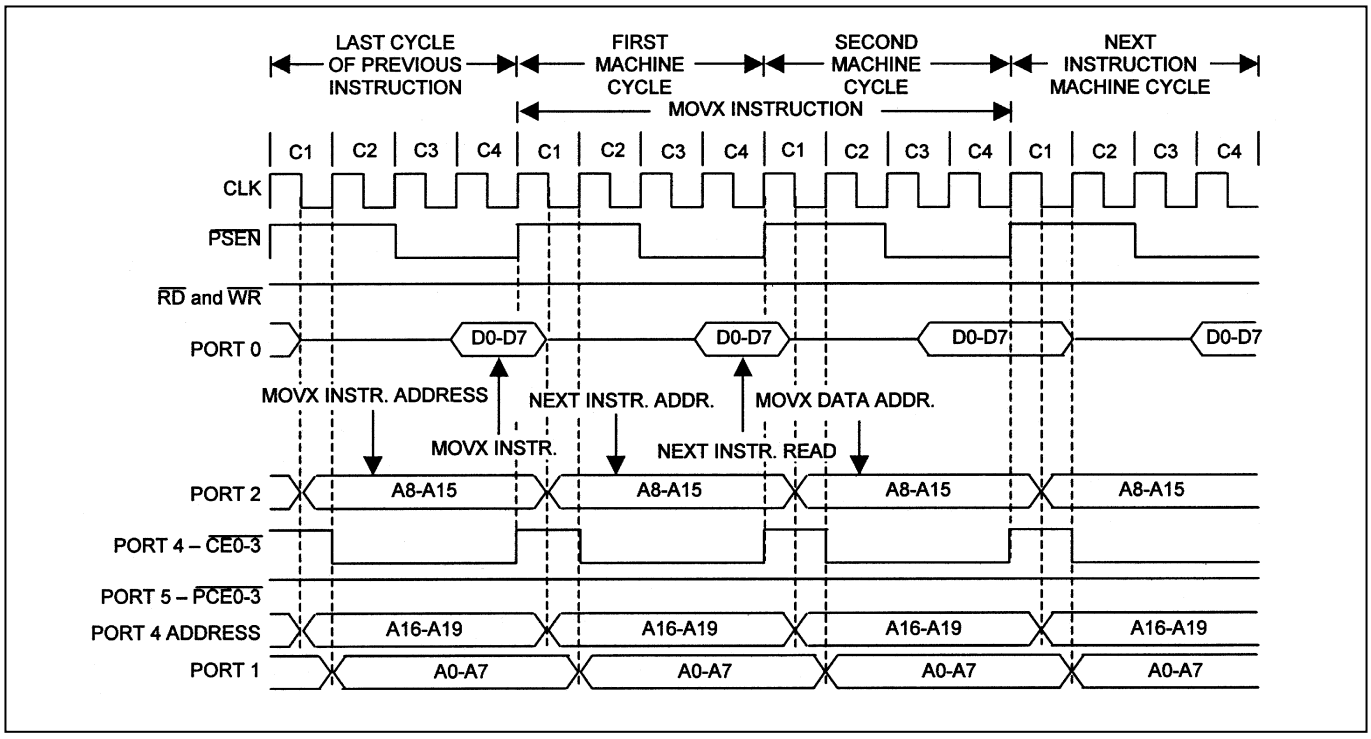


Figure 18. Nonmultiplexed 2-Cycle Data Memory $\overline{CE0\text{-}3}$ Write

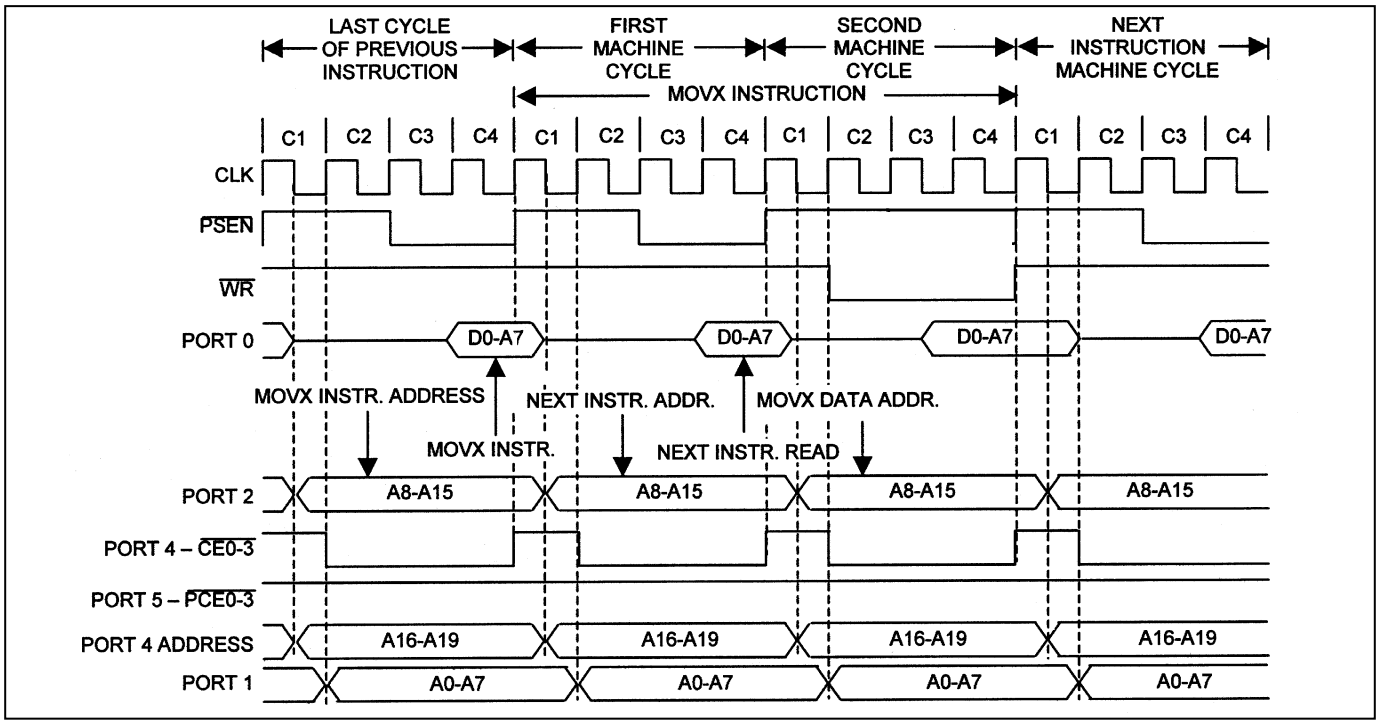


Figure 19. Nonmultiplexed 3-Cycle Data Memory $\overline{PEC0\text{-}3}$ Read or Write

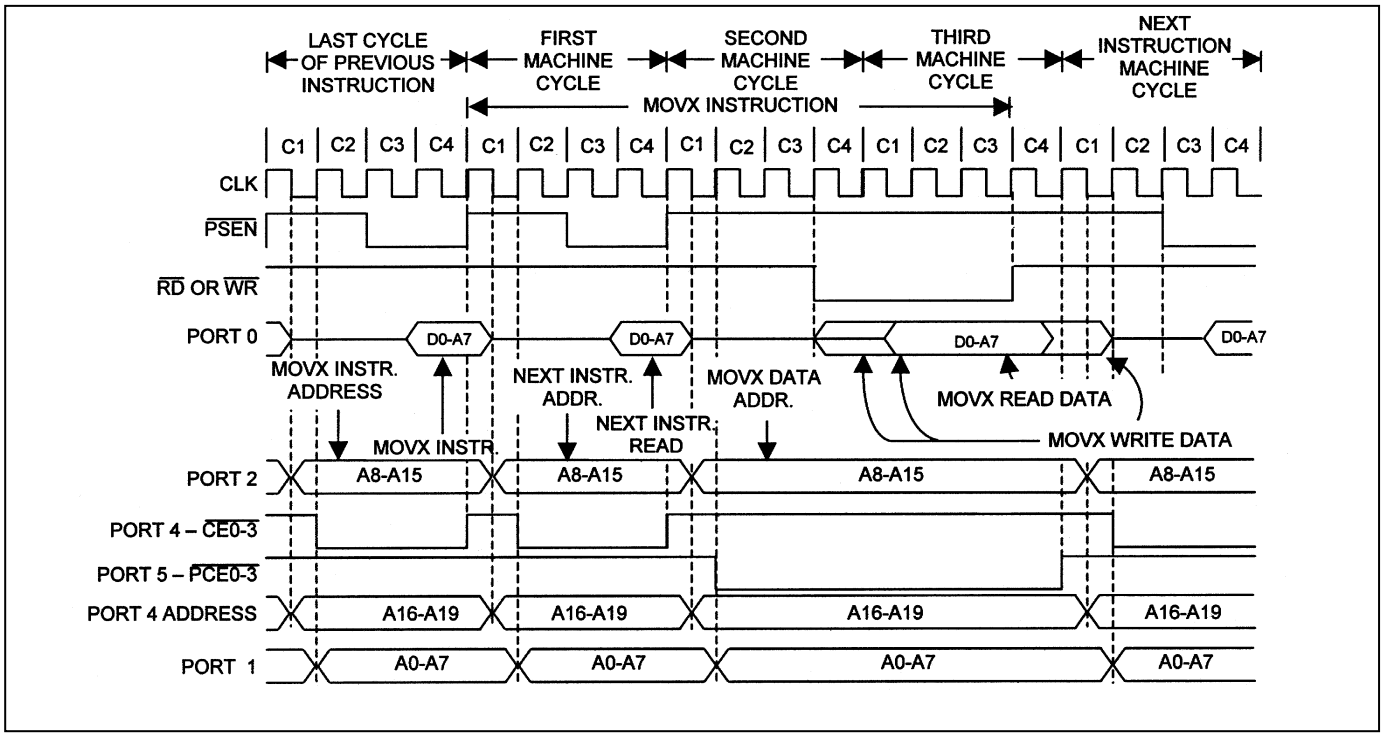


Figure 20. Nonmultiplexed 3-Cycle Data Memory $\overline{CE0\text{-}3}$ Read

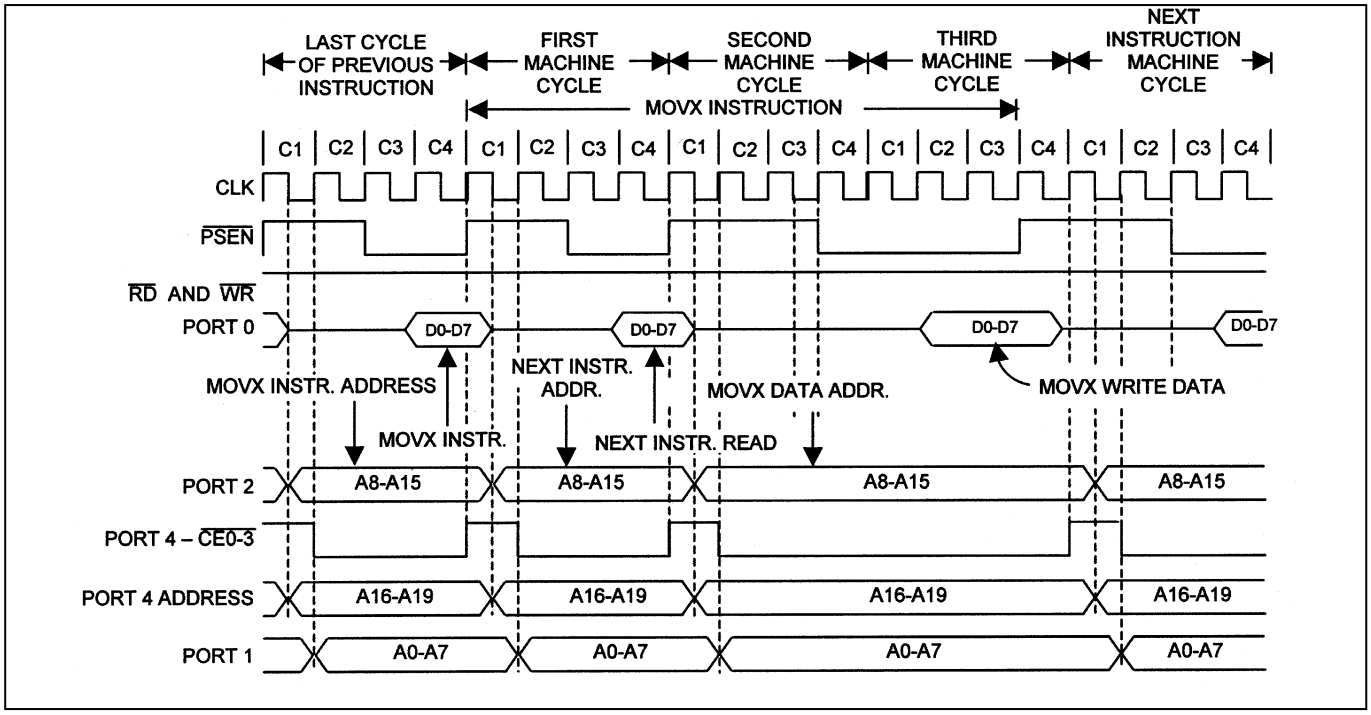


Figure 21. Nonmultiplexed 3-Cycle Data Memory $\overline{CE0\text{-}3}$ Write

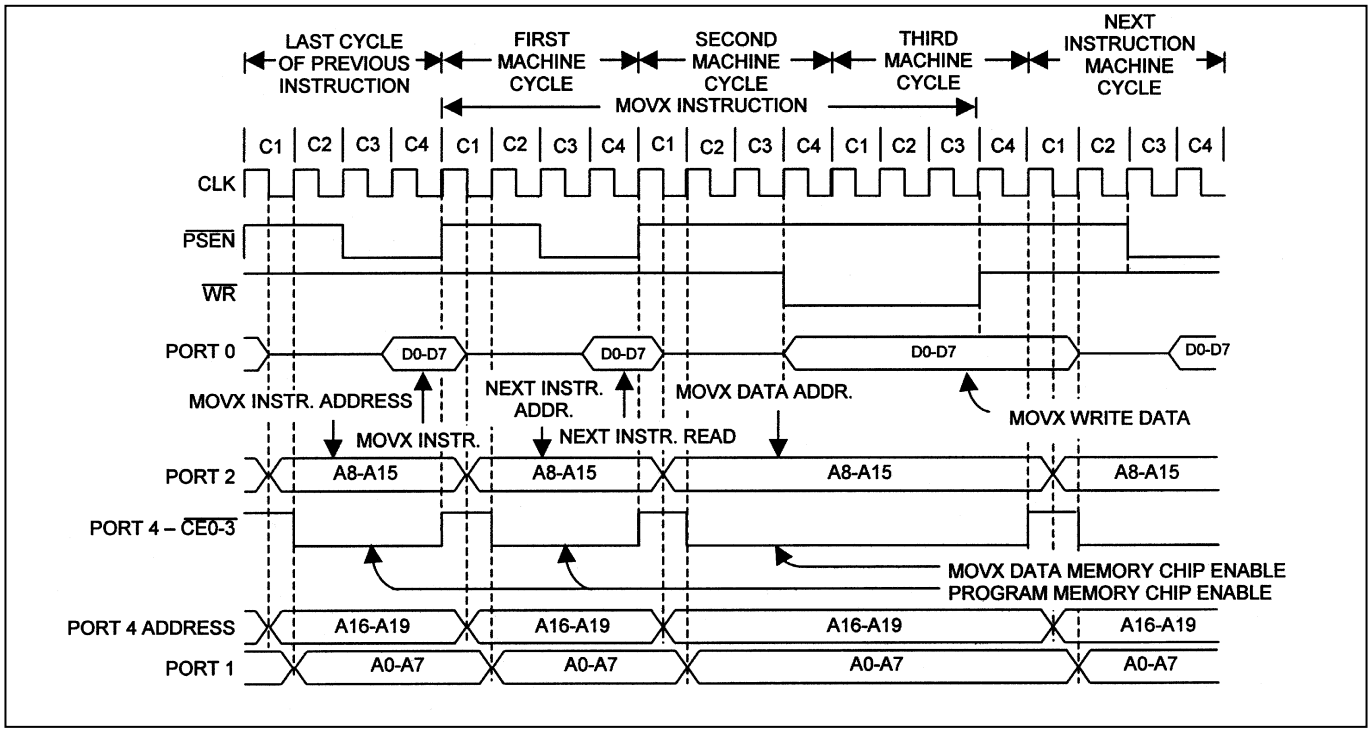


Figure 22. Nonmultiplexed 9-Cycle Data Memory $\overline{\text{PCE0-3}}$ Read or Write

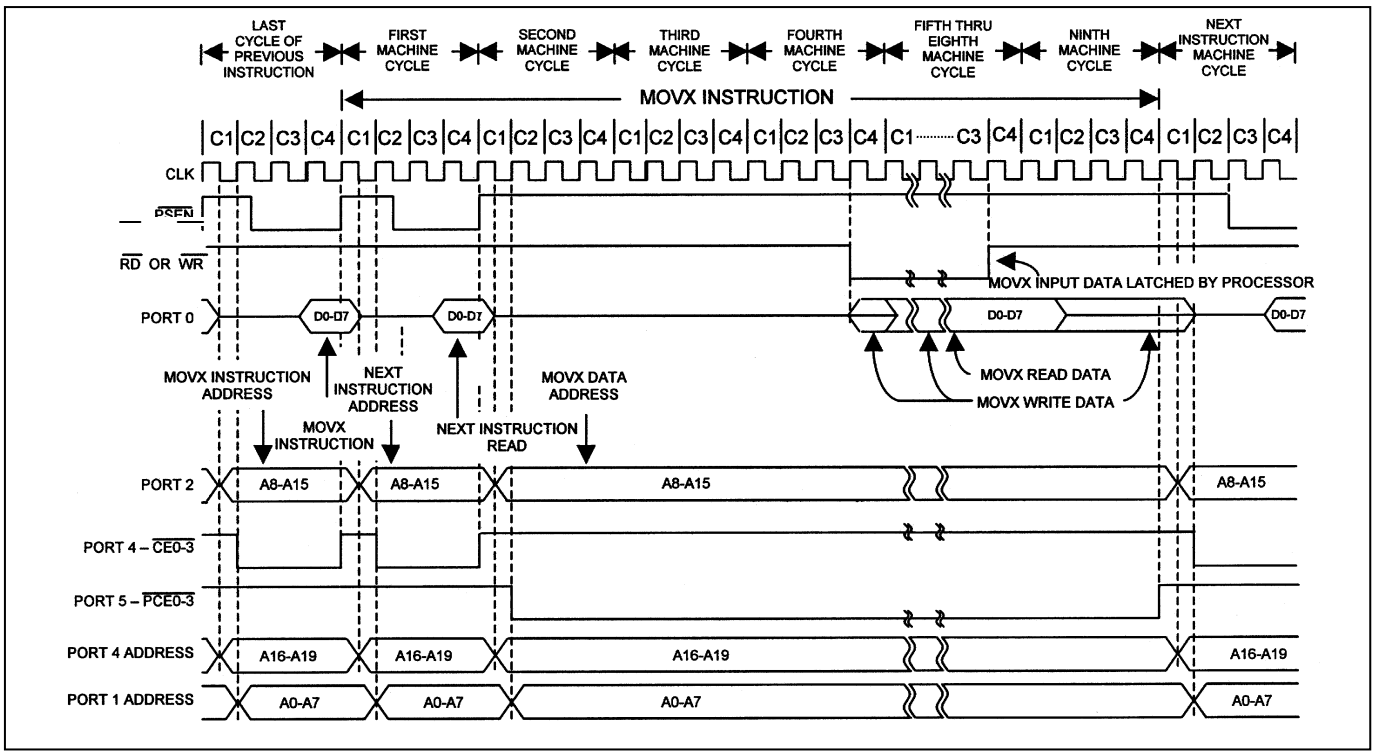


Figure 23. Nonmultiplexed 9-Cycle Data Memory $\overline{\text{CE0-3}}$ Read

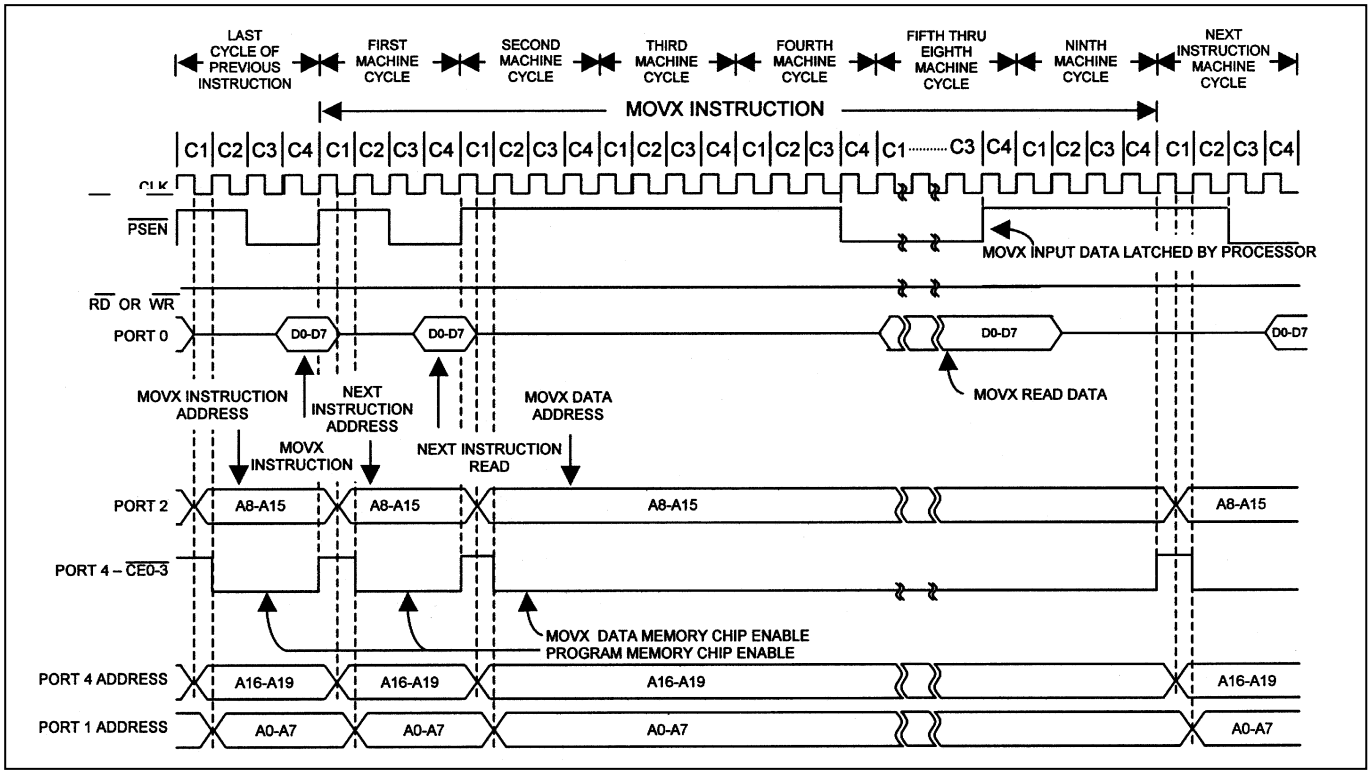
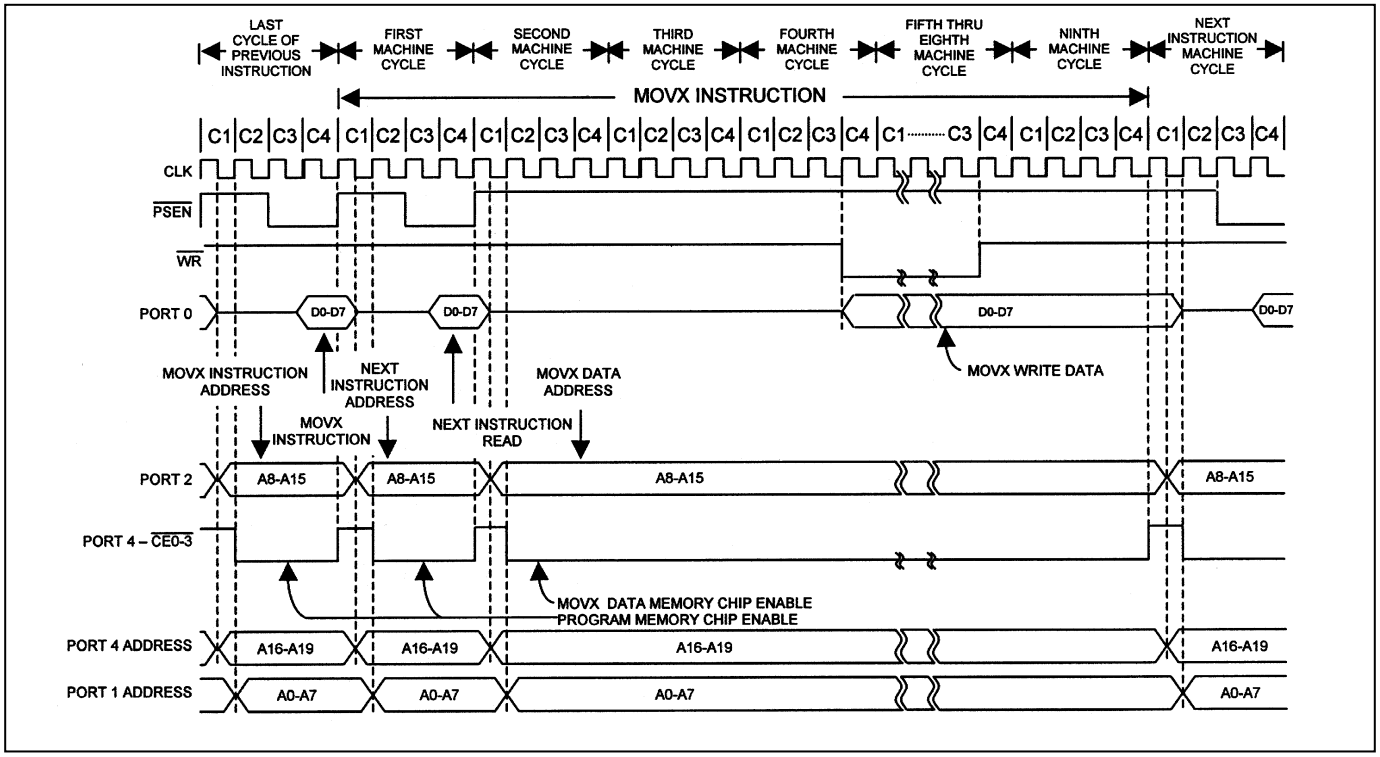


Figure 24. Nonmultiplexed 9-Cycle Data Memory $\overline{CE0\text{-}3}$ Write



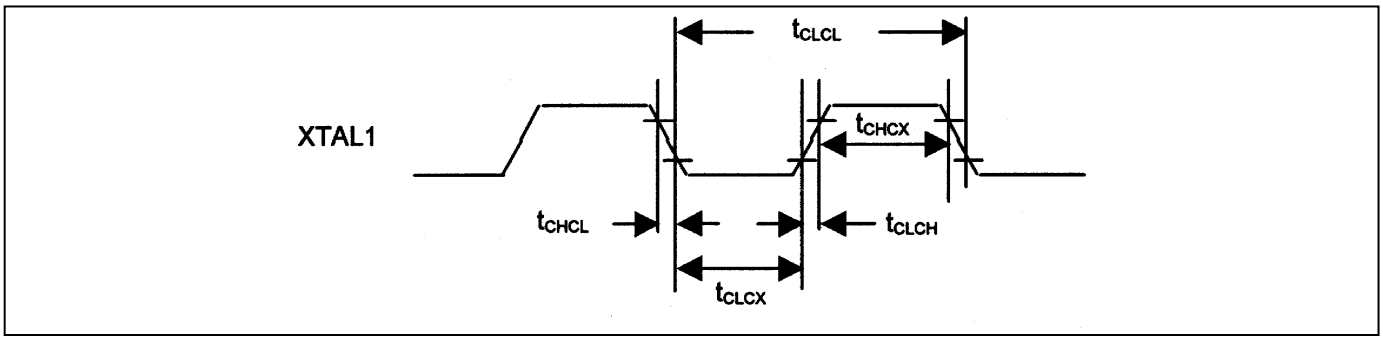
t_{MCS} TIME PERIODS

SYSTEM CLOCK SELECTION			t_{MCS}
4X/$\overline{2X}$	CD1	CD0	
1	0	0	1 t_{CLCL}
0	0	0	2 t_{CLCL}
X	1	0	4 t_{CLCL}
X	1	1	1024 t_{CLCL}

EXTERNAL CLOCK CHARACTERISTICS

PARAMETER	SYMBOL	MIN	MAX	UNITS
Clock High Time	t_{CHCX}	8		ns
Clock Low Time	t_{CLCX}	8		ns
Clock Rise Time	t_{CLCH}		4	ns
Clock Fall Time	t_{CHCL}		4	ns

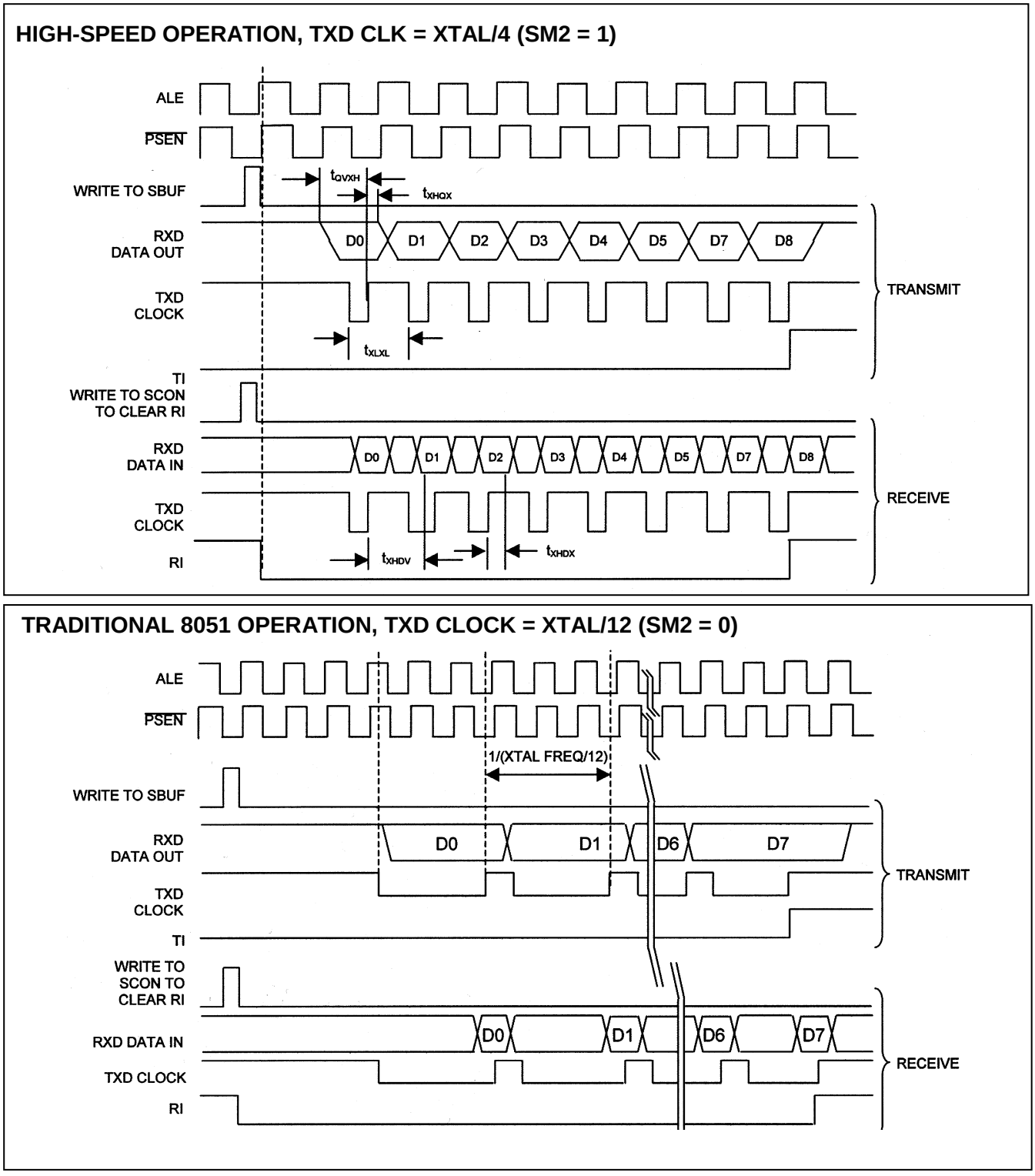
Figure 25. External Clock Drive



SERIAL PORT MODE 0 TIMING CHARACTERISTICS

PARAMETER	SYMBOL	CONDITIONS	TYP	UNITS
Serial Port Clock Cycle Time	t_{XLXL}	SM2 = 0:2 clocks per cycle	12 t_{CLCL}	ns
		SM2 = 1:4 clocks per cycle	4 t_{CLCL}	
Output Data Setup to Clock Rising	t_{QVXH}	SM2 = 0:12 clocks per cycle	10 t_{CLCL}	ns
		SM2 = 1:4 clocks per cycle	3 t_{CLCL}	
Output Data Hold from Clock Rising	t_{XHQX}	M2 = 0:12 clocks per cycle	2 t_{CLCL}	ns
		SM2 = 1:4 clocks per cycle	t_{CLCL}	
Input Data Hold After Clock Rising	t_{XHDX}	SM2 = 0:12 clocks per cycle	t_{CLCL}	ns
		SM2 = 1:4 clocks per cycle	0	
Clock Rising Edge to Input Data Valid	t_{XHDV}	SM2 = 0:12 clocks per cycle	11 t_{CLCL}	ns
		SM2 = 1:4 clocks per cycle	2 t_{CLCL}	

Figure 26. Serial Port 0 (Synchronous Mode)



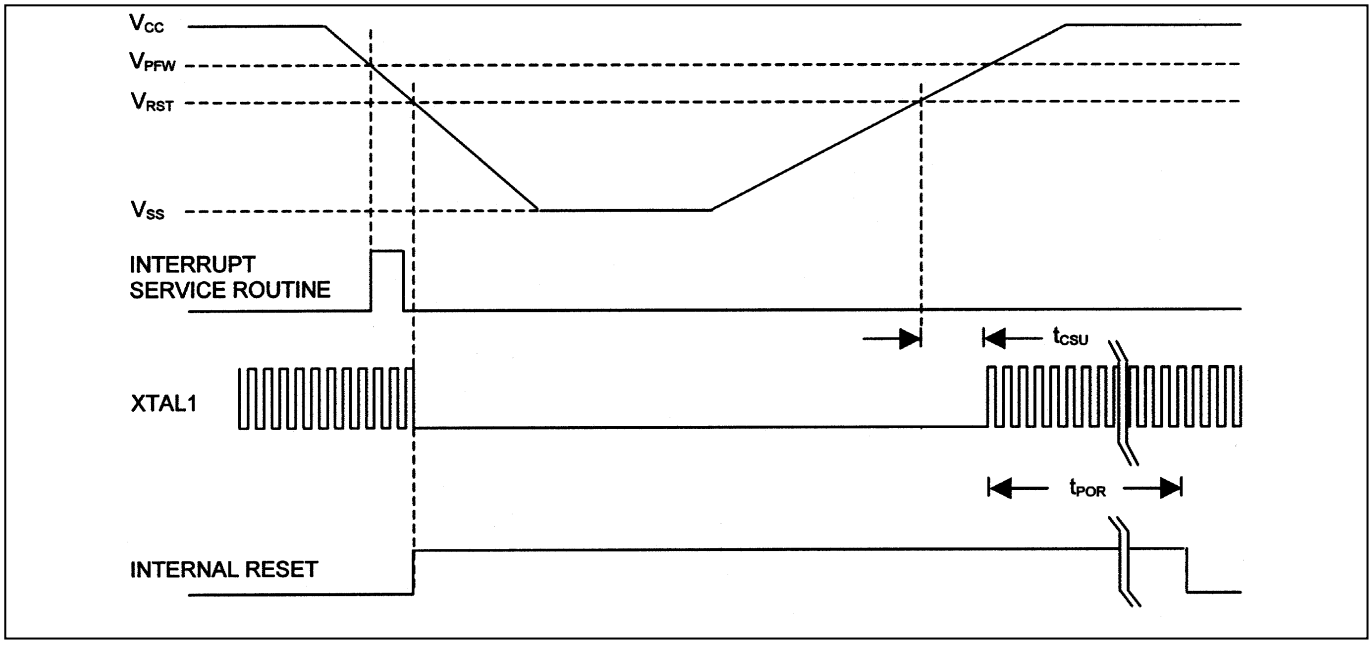
POWER-CYCLE TIMING CHARACTERISTICS

PARAMETER	SYMBOL	TYP	MAX	UNITS
Crystal Startup Time (Note 14)	t_{CSU}	1.8		ms
Power-On Reset Delay (Note 15)	t_{POR}		65,536	t_{CLCL}

Note 14: Startup time for crystals varies with load capacitance and manufacturer. Time shown is for an 11.0592MHz crystal manufactured by Fox Electronics.

Note 15: Reset delay is a synchronous counter of crystal oscillations during crystal startup. Counting begins when the level on the XTAL1 input meets the V_{IH2} criteria. At 40MHz, this time is approximately 1.64ms.

Figure 27. Power-Cycle Timing



PIN DESCRIPTION

PIN		NAME	FUNCTION
LQFP	PLCC		
8, 22, 40, 56	17, 32, 51, 68	V_{CC}	**+5V**
9, 25, 41, 57	1, 18, 35, 52	GND	**Digital Circuit Ground**
46	57	ALE	**Address Latch Enable, Output.** When the $\overline{MUX}$ pin is low, this pin outputs a clock to latch the external address LSB from the multiplexed address/data bus on Port 0. This signal is commonly connected to the latch enable of an external transparent latch. ALE has a pulse width of 1.5 XTAL1 cycles and a period of four XTAL1 cycles. When the $\overline{MUX}$ pin is high, the pin will toggle continuously if the ALEOFF bit is cleared. ALE is forced high when the device is in a reset condition or if the ALEOFF bit is set while the $\overline{MUX}$ pin is high.
45	56	$\overline{PSEN}$	**Program Store Enable, Output.** This signal is the chip enable for external ROM memory. $\overline{PSEN}$ provides an active-low pulse and is driven high when external ROM is not being accessed.
47	58	$\overline{EA}$	**External Access Enable, Input.** This pin must be wired to GND for proper operation.
26	36	$\overline{MUX}$	**Multiplex/Demultiplex Select, Input.** This pin selects if the address/data bus operates in multiplexed ($\overline{MUX}$ = 0) or demultiplexed ($\overline{MUX}$ = 1) mode.
2	11	RST	**Reset, Input.** The RST input pin contains a Schmitt voltage input to recognize external active-high reset inputs. The pin also employs an internal pulldown resistor to allow for a combination of wired-OR external reset sources. An RC circuit is not required for power-up, as the device provides this function internally.
3	12	$\overline{RSTOL}$	**Reset Output Low, Output.** This active-low signal is asserted: When the processor has entered reset through the RST pin, During crystal warmup period following power-on or stop mode, During a watchdog timer reset (2 cycles duration), During an oscillator failure (if OFDE = 1), Whenever $V_{CC} \leq V_{RST.}$
23	33	XTAL2	**XTAL1, XTAL2.** Crystal oscillator pins support fundamental mode, parallel resonant, and AT-cut crystals. XTAL1 is the input if an external clock source is used in place of a crystal. XTAL2 is the output of the crystal amplifier.
24	34	XTAL1	
55	67	AD0/D0	**AD0–7 (Port 0), I/O.** When the $\overline{MUX}$ pin is wired low, Port 0 is the multiplexed address/data bus. While ALE is high, the LSB of a memory address is presented. While ALE falls, the port transitions to a bidirectional data bus. When the $\overline{MUX}$ pin is wired high, Port 0 functions as the bidirectional data bus. Port 0 cannot be modified by software. The reset condition of Port 0 pins is high. No pullup resistors are needed.
54	66	AD1/D1	
53	65	AD2/D2	
52	64	AD3/D3	
51	63	AD4/D4	
50	62	AD5/D5	
49	61	AD6/D6	
48	59	AD7/D7	

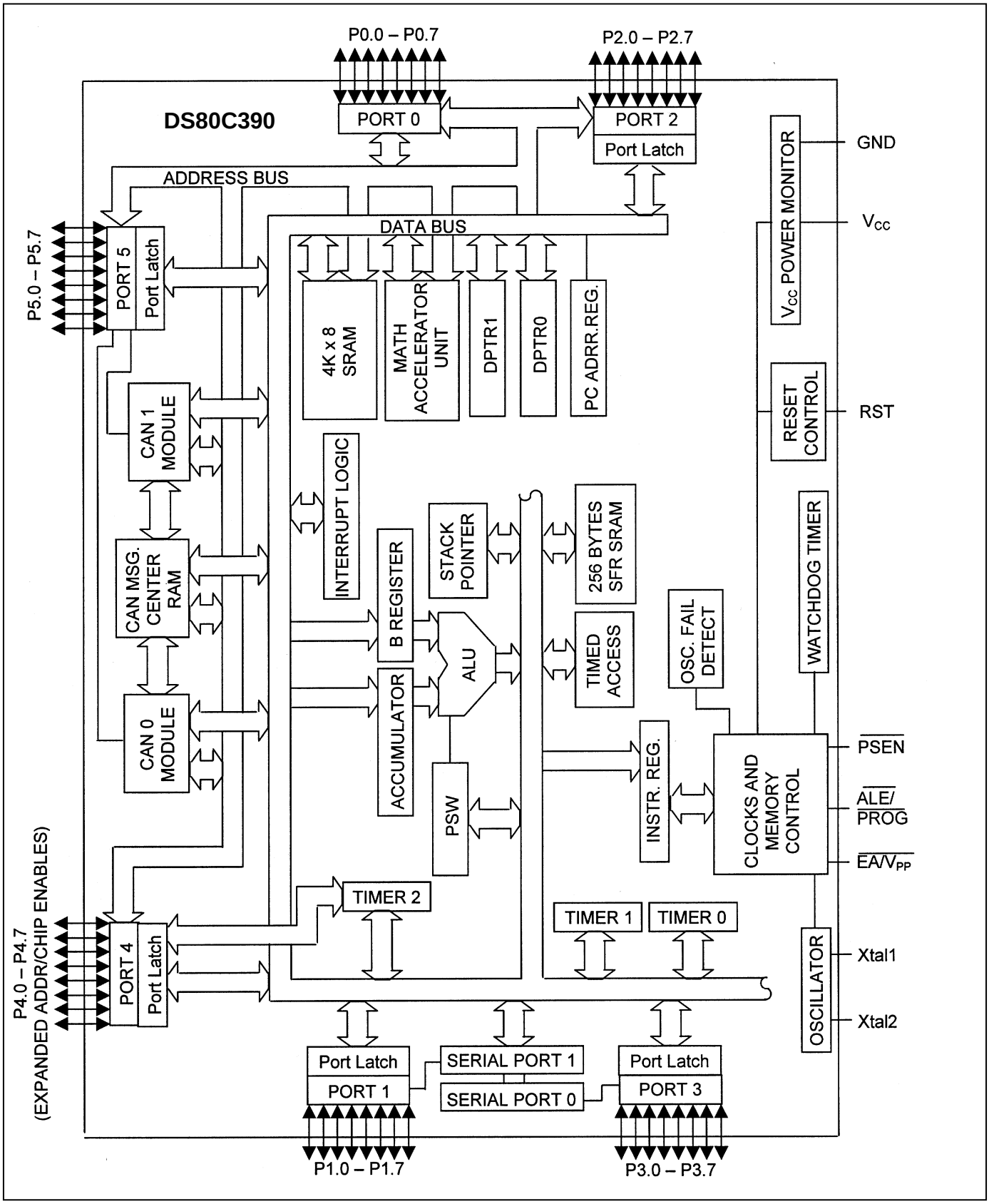
PIN DESCRIPTION (continued)

PIN		NAME	FUNCTION
LQFP	PLCC		
58–64, 1	2–8, 10	P1.0–P1.7	**Port 1, I/O.** Port 1 can function as an 8-bit bidirectional I/O port, the nonmultiplexed A0–A7 signals (when the $\overline{MUX}$ pin = 1), and as an alternate interface for internal resources. Setting the SP1EC bit relocates RXD1 and TXD1 to Port 5. The reset condition of Port 1 is all bits at logic 1 through a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, a strong pulldown is activated that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 activates a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state. **Port** **Alternate Function**
58	2	A0	P1.0 T2 External I/O for Timer/Counter 2
59	3	A1	P1.1 T2EX Timer/Counter 2 Capture/Reload Trigger
60	4	A2	P1.2 RXD1 Serial Port 1 Input
61	5	A3	P1.3 TXD1 Serial Port 1 Output
62	6	A4	P1.4 INT2 External Interrupt 2 (Positive Edge Detect)
63	7	A5	P1.5 $\overline{INT3}$ External Interrupt 3 (Negative Edge Detect)
64	8	A6	P1.6 INT4 External Interrupt 4 (Positive Edge Detect)
1	10	A7	P1.7 $\overline{INT5}$ External Interrupt 5 (Negative Edge Detect)
35	46	A8 (P2.0)	**A15–A8 (Port 2), Output.** Port 2 serves as the MSB for external addressing. The port automatically asserts the address MSB during external ROM and RAM access. Although the Port 2 SFR exists, the SFR value never appears on the pins (due to memory access). Therefore, accessing the Port 2 SFR is only useful for MOVX A, @Ri or MOVX @Ri, A instructions, which use the Port 2 SFR as the external address MSB.
36	47	A9 (P2.1)	
37	48	A10 (P2.2)	
38	49	A11 (P2.3)	
39	50	A12 (P2.4)	
42	53	A13 (P2.5)	
43	54	A14 (P2.6)	
44	55	A15 (P2.7)	
4–7, 10–13	13–16, 19–22	P3.0–P3.7	**Port 3, I/O.** Port 3 functions as an 8-bit bidirectional I/O port and as an alternate interface for several resources found on the traditional 8051. The reset condition of Port 1 is all bits at logic 1 through a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, the device activates a strong pulldown that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 activates a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state. **Port** **Alternate Function**
4	13		P3.0 RXD0 Serial Port 0 Input
5	14		P3.1 TXD0 Serial Port 0 Output
6	15		P3.2 $\overline{INT0}$ External Interrupt 0
7	16		P3.3 $\overline{INT1}$ External Interrupt 1
10	19		P3.4 T0 Timer 0 External Input
11	20		P3.5 T1/XCLK Timer 1 External Input/External Clock Output
12	21		P3.6 $\overline{WR}$ External Data Memory Write Strobe
13	22		P3.7 $\overline{RD}$ External Data Memory Read Strobe

PIN DESCRIPTION (continued)

PIN		NAME	FUNCTION
LQFP	PLCC		
34–27	45, 44, 42–37	P4.0–P4.7	**Port 4, I/O.** Port 4 can function as an 8-bit, bidirectional I/O port, and as the source for external address and chip enable signals for program and data memory. Port pins are configured as I/O or memory signals via the P4CNT register. The reset condition of Port 1 is all bits at logic 1 via a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, the device activates a strong pulldown that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 will activate a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state.
34 33 32 31 30 29 28 27	45 44 42 41 40 39 38 37		**Port** **Alternate Function** P4.0 $\overline{CE0}$ Program Memory Chip Enable 0 P4.1 $\overline{CE1}$ Program Memory Chip Enable 1 P4.2 $\overline{CE2}$ Program Memory Chip Enable 2 P4.3 $\overline{CE3}$ Program Memory Chip Enable 3 P4.4 A16 Program/Data Memory Address 16 P4.5 A17 Program/Data Memory Address 17 P4.6 A18 Program/Data Memory Address 18 P4.7 A19 Program/Data Memory Address 19
21–14	31–27, 25–23	P5.0–P5.7	**Port 5, I/O.** Port 5 can function as an 8-bit, bidirectional I/O port, the CAN interface, or as peripheral enable signals. Setting the SP1EC bit will relocate the RXD1 and TXD1 functions to P5.3-P5.2 as described in the *High-Speed Microcontroller User's Guide: DS80C390 Supplement*. The reset condition of Port 1 is all bits at logic 1 via a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, the device activates a strong pulldown that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 will activate a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state. **Port** **Alternate Function** P5.0 C0TX CAN0 Transmit Output P5.1 C0RX CAN0 Receive Input P5.2 C1RX CAN1 Receive Input (optional RXD1) P5.3 C1TX CAN1 Transmit Output (optional TXD1) P5.4 $\overline{PCE0}$ Peripheral Chip Enable 0 P5.5 $\overline{PCE1}$ Peripheral Chip Enable 1 P5.6 $\overline{PCE2}$ Peripheral Chip Enable 2 P5.7 $\overline{PCE3}$ Peripheral Chip Enable 3
21 20 19 18 17 16 15 14	31 30 29 28 27 25 24 23		
	9, 26, 43, 60	N.C.	**Not Connected.** Reserved. These pins are reserved for use with future devices in this family and should not be connected.

Figure 28. Block Diagram



FEATURES

- **80C52 Compatible**
  8051-Instruction-Set Compatible
  Four 8-Bit I/O Ports
  Three 16-Bit Timer/Counters
  256 Bytes Scratchpad RAM
- **High-Speed Architecture**
  4 Clocks/Machine Cycle (8051 = 12)
  Runs DC to 40MHz Clock Rates
  Frequency Multiplier Reduces Electromagnetic Interference (EMI)
  Single-Cycle Instruction in 100ns
  16/32-Bit Math Coprocessor
- **4kB Internal SRAM Usable as Program/Data/Stack Memory**
- **Enhanced Memory Architecture**
  Addresses Up to 4MB External
  Defaults to True 8051-Memory Compatibility
  User-Enabled 22-Bit Program/Data Counter
  16-Bit/22-Bit Paged/22-Bit Contiguous Modes
  User-Selectable Multiplexed/Nonmultiplexed Memory Interface
  Optional 10-Bit Stack Pointer
- **Two Full-Function CAN 2.0B Controllers**
  15 Message Centers Per Controller
  Standard 11-Bit or Extended 29-Bit Identification Modes
  Supports DeviceNet™, SDS, and Higher Layer CAN Protocols
  Disables Transmitter During Autobaud
  SIESTA Low-Power Mode
- **Two Full-Duplex Hardware Serial Ports**
- **Programmable IrDA Clock**
- **High-Integration Controller Includes**:
  Power-Fail Reset
  Early-Warning Power-Fail Interrupt
  Programmable Watchdog Timer
  Oscillator-Fail Detection
- **16 Interrupt Sources with Six External**
- **Available in 64-Pin LQFP and 68-Pin PLCC**

DETAILED DESCRIPTION

The DS80C390 features two full-function controller area network (CAN) 2.0B controllers. Status and control registers are distributed between SFRs and 512 bytes of internal MOVX memory for maximum flexibility. In addition to standard 11-bit or 29-extended message identifiers, the device supports two separate 8-bit media masks and media arbitration fields to support the use of higher-level CAN protocols such as DeviceNet and SDS.

All of the standard 8051 resources such as three timer/counters, serial port, and four 8-bit I/O ports (plus two 8-bit ports dedicated to memory interfacing) are included in the DS80C390. In addition it includes a second hardware serial port, seven additional interrupts, programmable watchdog timer, brownout monitor, power-fail reset, and a programmable output clock that supports an IrDA interface. The device provides dual data pointers with increment/decrement features to speed block data memory moves. It also can adjust the speed of MOVX data memory access from 2 to 12 machine cycles for flexibility in addressing external memory and peripherals.

The device incorporates a 4kB SRAM, which can be configured as various combinations of MOVX memory, program memory, and optional stack memory. A 22-bit program counter supports access to a maximum of 4MB of external program memory and 4MB of external data memory. A 10-bit stack pointer addresses up to 1kB of MOVX memory for increased code efficiency.

A new power-management mode (PMM) is useful for portable or power-conscious applications. This feature allows software to switch from the standard machine cycle rate of 4 clocks per cycle to 1024 clocks per cycle. For example, at 12MHz standard operation has a machine cycle rate of 3MHz. In PMM at the same external clock speed, software can select 11.7kHz machine cycle rate. There is a corresponding reduction in power consumption when the processor runs slower.

The EMI reduction feature allows software to select a reduced electromagnetic interference (EMI) mode by disabling the ALE signal when it is unneeded. The device also incorporates active current control on the address and data buses, reducing EMI by minimizing transients when interfacing to external circuitry.

80C32 COMPATIBILITY

The DS80C390 is a CMOS 80C32-compatible microcontroller designed for high performance. Every effort has been made to keep the core device familiar to 80C32 users while adding many new features.

DeviceNet is a trademark of Open DeviceNet Vendor Association, Inc.

Because the device runs the standard 8051 instruction set, in general, software written for existing 80C32-based systems will work on the DS80C390. The primary exceptions are related to timing-critical issues, since the high-performance core of the microcontroller executes instructions much faster than the original. Memory interfacing is performed identically to the standard 80C32. The high-speed nature of the DS80C390 core slightly changes the interface timing, and designers are advised to consult the timing diagrams in this data sheet for more information.

The DS80C390 provides the same timer/counter resources, full duplex serial port, 256 bytes of scratchpad RAM and I/O ports as the standard 80C32. Timers default to a 12 clocks-per-machine cycle operation to keep timing compatible with original 8051 systems, but can be programmed to run at the faster four clocks-per-machine cycle if desired. New hardware functions are accessed using special function registers that do not overlap with standard 80C32 locations.

This data sheet provides only a summary and overview of the DS80C390. Detailed descriptions are available in the *High-Speed Microcontroller User's Guide: DS80C390 Supplement*. This data sheet assumes a familiarity with the architecture of the standard 80C32. In addition to the basic features of that device, the DS80C390 incorporates many new features.

PERFORMANCE OVERVIEW

The DS80C390's higher performance comes not just from increasing the clock frequency but also from a more efficient design. This updated core removes the dummy memory cycles that are present in a standard, 12 clocks-per-machine cycle 8051. In the DS80C390, the same machine cycle takes 4 clocks. Thus the fastest instruction, one machine cycle, executes three times faster for the same crystal frequency. The majority of instructions on the DS80C390 see the full 3-to-1 speed improvement, while a few execute between 1.5 and 2.4 times faster. Regardless of specific performance improvements, all instructions are faster than the original 8051.

Improvement of individual programs depends on the actual mix of instructions used. Speed-sensitive applications should make the most use of instructions that are three times faster. However, the large number of 3-to-1 improved op codes makes dramatic speed improvements likely for any arbitrary combination of instructions. These architecture improvements and the submicron CMOS design produce a peak instruction cycle in 100ns (10 MIPS). The dual data pointer feature also allows the user to eliminate wasted instructions when moving blocks of memory.

INSTRUCTION SET SUMMARY

All instructions perform exactly the same functions as their 8051 counterparts. Their effect on bits, flags, and other status functions is identical. However, the timing of instructions is different, both in absolute and relative number of clocks. The absolute timing of software loops can be calculated using a table in the *High-Speed Microcontroller User's Guide: DS80C390 Supplement*. However, counter/timers default to run at the traditional 12 clocks per increment. In this way, timer-based events occur at the standard intervals with software executing at higher speed. Timers optionally can run at the faster four clocks per increment to take advantage of faster processor operation.

The relative time of two DS80C390 instructions might differ from the traditional 8051. For example, in the original architecture the "MOVX A, @DPTR" instruction and the "MOV direct, direct" instruction required the same amount of time: two machine cycles or 24 oscillator cycles. In the DS80C390, the MOVX instruction takes as little as two machine cycles, or eight oscillator cycles, but the "MOV direct, direct" uses three machine cycles, or 12 oscillator cycles. While both are faster than their original counterparts, they now have different execution times. This is because the device usually uses one instruction cycle for each instruction byte. Examine the timing of each instruction for familiarity with the changes. Note that a machine cycle now requires just four clocks, and provides one ALE pulse per cycle. Many instructions require only one cycle, but some require five. Refer to the *High-Speed Microcontroller User's Guide: DS80C390 Supplement* for details and individual instruction timing.

SPECIAL FUNCTION REGISTERS (SFRs)

Special function registers (SFRs) control most special features of the microcontroller, allowing the device to have many new features but use the same instruction set as the 8051. When writing software to use a new feature, an equate statement defines the SFR to an assembler or compiler. This is the only change needed to access the new function. The DS80C390 duplicates the SFRs contained in the standard 80C52. Table 1 shows the register addresses and bit locations. Many are standard 80C52 registers. The *High-Speed Microcontroller User's Guide: DS80C390 Supplement* contains a full description of all SFRs.

Table 1. SFR Locations

REGISTER	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	ADDRESS
P4	P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	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	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	A0h
P5	P5.7	P5.6	P5.5	P5.4	P5.3	P5.2	P5.1	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	BSS	EC96/128	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
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	P3.7	P3.6	T1	T0	INT1	INT0	TXD0	RXD0	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

Table 1. SFR Locations (continued)

REGISTER	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	ADDRESS
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	EPFI	PFI	WDIF	WTRF	EWT	RWT	D8h
C1TMA0									DEh
C1TMA1									DFh
ACC									E0h
C1C	ERIE	STIE	PDE	SIESTA	CRST	AUTOB	ERCS	SWINT	E3h
C1S	BSS	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
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

Note: Shaded bits are timed-access protected.

ON-CHIP ARITHMETIC ACCELERATOR

An on-chip math accelerator allows the microcontroller to perform 32-bit and 16-bit multiplication, division, shifting, and normalization using dedicated hardware. 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 normalize function facilitates the conversion of 4-byte unsigned binary integers into floating point format. Table 2 shows the operations supported by the math accelerator and their time of execution.

Table 2. Arithmetic Accelerator Execution Times

OPERATION	RESULT	EXECUTION TIME (t_{CLCL})
32-Bit/16-Bit Divide	32-Bit Quotient, 16-Bit Remainder	36
16-Bit/16-Bit Divide	16-Bit Quotient, 16-Bit Remainder	24
16-Bit/16-Bit Multiply	32-Bit Product	24
32-Bit Shift Left/Right	32-Bit Result	36
32-Bit Normalize	32-Bit Mantissa, 5-Bit Exponent	36

Table 3 demonstrates the procedure to perform mathematical operations using the hardware math 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 corrupts the operation, requiring the software to clear the MST bit to restart the math accelerator state machine. Consult the description of the MCNT0 SFR for details of how the shift and normalize functions operate.

Software must ensure that the input value for the normalize operation is not zero or the function will not complete. Compilers such as the one from Keil Software have updated their libraries and compensate for this condition.

Table 3. Arithmetic Accelerator Sequencing

DIVIDE (32/16 OR 16/16)	MULTIPLY (16 X 16)
Load MA with dividend LSB. *Load MA with dividend LSB + 1.** L*oad MA with dividend LSB + 2.** Load MA with dividend MSB. Load MB with divisor LSB. Load MB with divisor MSB. Poll the MST bit until cleared. (9 machine cycles). Read MA to retrieve the quotient MSB. *Read MA to retrieve the quotient LSB + 2.***** *Read MA to retrieve the quotient LSB + 1.***** Read MA to retrieve the quotient LSB. Read MB to retrieve the remainder MSB. Read MB to retrieve the remainder LSB.	Load MB with multiplier LSB. Load MB with multiplier MSB. Load MA with multiplicand LSB. Load MA with multiplicand MSB. Poll the MST bit until cleared. (6 machine cycles). Read MA for product MSB. Read MA for product LSB + 2. Read MA for product LSB + 1. Read MA for product LSB.
SHIFT RIGHT/LEFT	**NORMALIZE**
Load MA with data LSB. Load MA with data LSB + 1. Load MA with data LSB + 2. Load MA with data MSB. Configure MCNT0 register as required Poll the MST bit until cleared. (9 machine cycles) Read MA for result MSB. Read MA for result LSB + 2. Read MA for result LSB + 1. Read MA for result LSB.	Load MA with data LSB. Load MA with data LSB + 1. Load MA with data LSB + 2. Load MA with data MSB.****** Load MCNT0 with 00h. Poll the MST bit until cleared. (9 machine cycles) Read MA for mantissa MSB. Read MA for mantissa LSB + 2. Read MA for mantissa LSB + 1. Read MA for mantissa LSB. Read MCNT0.4–MCNT0.0 for exponent.

**Not performed for 16-bit numerator.*

***Not performed for 16/16 divide.*

****Value to be normalized must be nonzero.*

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 anytime 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.

MEMORY ADDRESSING

The DS80C390 incorporates three internal memory areas:

- 256 bytes of scratchpad (or direct) RAM
- 4kB of SRAM configurable as various combinations of MOVX data memory, stack memory, and MOVC program memory
- 512 bytes of RAM reserved for the CAN message centers.

Up to 4MB of external memory is addressed 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. Three different addressing modes are supported, as selected by the AM1, AM0 bits in the ACON SFR.

16-Bit Address Mode

Memory is accessed by 16-bit address mode similarly to the traditional 8051. It is op-code compatible with the 8051 microprocessor and identical to the byte and cycle count of the Dallas Semiconductor High-Speed Microcontroller family. A device operating in this mode can access up to 64kB of program and data memory. The device defaults to this mode following any reset.

22-Bit Paged-Address Mode

The 22-bit paged-address mode retains binary-code compatibility with the 8051 instruction set, but adds one machine cycle to the ACALL, LCALL, RET, and RETI instructions with respect to Dallas Semiconductor's High-Speed Microcontroller family timing. This is transparent to standard 8051 compilers. Interrupt latency is also increased by one machine cycle. In this mode, interrupt vectors are fetched from 0000xxh.

22-Bit Contiguous Address Mode

The 22-bit contiguous addressing mode uses a full 22-bit program counter, and all modified branching instructions automatically save and restore the entire program counter. The 22-bit branching instructions such as ACALL, AJMP, LCALL, LJMP, MOV DPTR, RET, and RETI instructions require an assembler, compiler, and linker that specifically supports these features. The INC DPTR is lengthened by one cycle but remains byte-count-compatible with the standard 8051 instruction set.

Internally, the device uses a 22-bit program counter. The lowest order 22 bits are used for memory addressing, with a special 23rd bit used to map the 4kB SRAM above the 4MB memory space in bootstrap loader applications. Address bits 16–23 for the 22-bit addressing modes are generated through additional SFRs dependent on the type of instruction as shown in Table 4.

Table 4. Extended Address Generation

INSTRUCTION	ADDRESS BITS 23–16	ADDRESS BITS 15–8	ADDRESS BITS 7–0
MOVX instructions using DPTR	DPX;93h	DPH;83h	DPL;82h
MOVX instructions using DPTR1	DPX1;95h	DPH1;85h	DPL1;84h
MOVX instructions using @Ri	MXAX;EAh	P2;A0h	Ri
Addressing program memory in 22-bit paged mode	AP;9Ch	—	—
10-bit stack pointer mode	—	ESP;9Bh	SP;81h

INTERNAL MOVX SRAM

The DS80C390 contains 4kB of SRAM that can be configured as user accessible MOVX memory, program memory, or optional stack memory. The specific configuration and locations are governed by the internal data memory configuration bits (IDM1, IDM0) in the memory control register (MCON;C6h). Note that when the SA bit (ACON.2) is set, the first 1kB of the MOVX data memory is reserved for use by the 10-bit expanded stack. Internal memory accesses will not generate $\overline{WR}$, $\overline{RD}$, or $\overline{PSEN}$ strobes.

The DS80C390 can configure its 4kB of internal SRAM as combined program and data memory. This allows the application software to execute self-modifiable code. The technique loads the 4kB SRAM with bootstrap loader software, and then modifies the IDM1 and IDM0 bits to map the 4kB starting at memory location 40000h. This allows the system to run the bootstrap loader without disturbing the 4MB external memory bus, making the device in-system reprogrammable for flash or NV RAM.

Table 5. Internal MOVX SRAM Configuration

IDM1	IDM0	CMA	MEMORY		
			MOVX DATA	CAN MESSAGE	SHARED PROGRAM/DATA
0	0	0	00F000h–00FFFFh	00EE00h–00EFFFh	—
0	0	1	00F000h–00FFFFh	401000h–4011FFh	—
0	1	0	000000h–000FFFh	00EE00h–00EFFFh	—
0	1	1	000000h–000FFFh	401000h–4011FFh	—
1	0	0	400000h–400FFFh	00EE00h–00EFFFh	—
1	0	1	400000h–400FFFh	401000h–4011FFh	—
1	1	0	—	00EE00h–00EFFFh	400000h–400FFFh*
1	1	1	—	401000h–4011FFh	400000h–400FFFh*

**10-bit expanded stack is not available in shared program/data memory mode.*

EXTERNAL MEMORY ADDRESSING

The enabling and mapping of the chip-enable signals is done through the Port 4 control register (P4CNT;92h) and memory control register (MCON; 96h). Table 7 shows which chip-enable and address line signals are active on Port 4. Following reset, the device will be configured with P4.7–P4.4 as address lines and P4.3–P4.0 configured as $\overline{CE3\text{-}0}$, with the first program fetch being performed from 00000h with $\overline{CE0}$ active. The following tables illustrate which memory ranges are controlled by each chip enable as a function of which address lines are enabled.

Table 6. External Memory Addressing Pin Assignments

ADDRESS/DATA BUS	$\overline{CE3–CE0}$	$\overline{PCE3–PCE0}$	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

Table 7. Extended Address and Chip-Enable Generation

P4CNT.5–3	PORT 4 PIN FUNCTION				P4CNT.2–0	PORT 4 PIN FUNCTION			
	P4.7	P4.6	P4.5	P4.4		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)	$\overline{CE3}$	$\overline{CE2}$	$\overline{CE1}$	$\overline{CE0}$

Table 8. Program Memory Chip-Enable Boundaries

P4CNT.5–3	$\overline{CE0}$	$\overline{CE1}$	$\overline{CE2}$	$\overline{CE3}$
000	0h–7FFFh	8000h–FFFFh	10000h–17FFFh	18000h–1FFFFh
100	0h–1FFFFh	20000h–3FFFFh	40000h–5FFFFh	60000h–7FFFFh
101	0h–3FFFFh	40000h–7FFFFh	80000h–BFFFFh	C0000h–FFFFFh
110	0h–7FFFFh	80000h–FFFFFh	100000h–17FFFFh	180000h–1FFFFFh
111(default)	0–FFFFFh	100000h–1FFFFFh	200000h–2FFFFFh	300000h–3FFFFFh

The DS80C390 incorporates a feature allowing PCE and CE 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 $\overline{PDCE3–0}$ (MCON.3–0) bits causes the corresponding chip-enable signal to function 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 through both MOVC instructions (program fetch) and MOVX write operations (updates to code memory). In this case, the internal SRAM is placed in the program/data configuration and loaded with a small bootstrap loader program stored in the external flash memory. The device then executes the internal bootstrap loader routine to modify/update the program memory located in the external flash memory.

STRETCH MEMORY CYCLES

The DS80C390 allows user-application software to select the number of machine cycles it takes to execute a MOVX instruction, allowing access to both fast and slow off-chip data memory and/or peripherals without glue logic. High-speed systems often include memory-mapped peripherals such as LCDs or UARTs with slow access times, so it may not be necessary or desirable to access external devices at full speed. The microprocessor can perform a MOVX instruction in as little as two machine cycles or as many as twelve machine cycles. Accesses to internal MOVX SRAM always use two cycles. Note that stretch cycle settings affect external MOVX memory operations only and that there is no way to slow the accesses to program memory other than to use a slower crystal (or external clock).

External MOVX timing is governed by the selection of 0 to 7 stretch cycles, controlled by the MD2–MD0 SFR bits in the clock-control register (CKCON.2–0). A stretch of zero results in a 2-machine cycle MOVX instruction. A stretch of seven results in a MOVX of 12 machine cycles. Software can dynamically change the stretch value depending on the particular memory or peripheral being accessed. The default of one stretch cycle allows the use of commonly available SRAMs without dramatically lengthening the memory access times.

Stretch cycle settings affect external MOVX timing in three gradations. Changing the stretch value from 0 to 1 adds an additional clock cycle each to the data setup and hold times. When a stretch value of 4 or above is selected, the interface timing changes dramatically to allow for very slow peripherals. First, the ALE signal is lengthened by 1 machine cycle. This increases the address setup time into the peripheral by this amount. Next, the address is held on the bus for one additional machine cycle increasing the address hold time by this amount. The $\overline{WR}$ and $\overline{RD}$ signals are then lengthened by a machine cycle. Finally, during a MOVX write the data is held on the bus for one additional machine cycle, thereby increasing the data hold time by this amount. For every stretch value greater than 4, the setup and hold times remain constant, and only the width of the read or write signal is increased. These three gradations are reflected in the *AC Electrical Characteristics*, where the eight MOVX timing specifications are represented by only three timing diagrams.

The reset default of one stretch cycle results in a three-cycle MOVX for any external access. Therefore, the default off-chip RAM access is not at full speed. This is a convenience to existing designs that use slower RAM. When maximum speed is desired, software should select a stretch value of zero. When using very slow RAM or peripherals, the application software can select a larger stretch value.

The specific timing of MOVX instructions as a function of stretch settings is provided in the *Electrical Specifications* section of this data sheet. As an example, Table 9 shows the read and write strobe widths corresponding to each stretch value.

Table 9. Data Memory Cycle Stretch Values

MD2	MD1	MD0	STRETCH CYCLE COUNT	MOVX MACHINE CYCLES	$\overline{RD}$, $\overline{WR}$ PULSE WIDTH (IN OSCILLATOR CLOCKS)			
					t_{MCS} (4X/$\overline{2X}$ = 1 CD1:0 = 00)	t_{MCS} (4X/$\overline{2X}$ = 0 CD1:0 = 00)	t_{MCS} (4X/$\overline{2X}$ = X CD1:0 = 10)	t_{MCS} (4X/$\overline{2X}$ = X CD1:0 = 11)
0	0	0	0*	2	0.5 t_{CLCL}	1 t_{CLCL}	2 t_{CLCL}	2048 t_{CLCL}
0	0	1	1**	3	t_{CLCL}	2 t_{CLCL}	4 t_{CLCL}	4096 t_{CLCL}
0	1	0	2	4	2 t_{CLCL}	4 t_{CLCL}	8 t_{CLCL}	8192 t_{CLCL}
0	1	1	3	5	3 t_{CLCL}	6 t_{CLCL}	12 t_{CLCL}	12,288 t_{CLCL}
1	0	0	4	9	4 t_{CLCL}	8 t_{CLCL}	16 t_{CLCL}	16,384 t_{CLCL}
1	0	1	5	10	5 t_{CLCL}	10 t_{CLCL}	20 t_{CLCL}	20,480 t_{CLCL}
1	1	0	6	11	6 t_{CLCL}	12 t_{CLCL}	24 t_{CLCL}	24,576 t_{CLCL}
1	1	1	7	12	7 t_{CLCL}	14 t_{CLCL}	28 t_{CLCL}	28,672 t_{CLCL}

**All internal MOVX operations execute at the 0 Stretch setting.*
***Default stretch setting for external MOVX operations following reset.*

EXTENDED STACK POINTER

The DS80C390 supports both the traditional 8-bit and an extended 10-bit stack pointer that improves the performance of large programs written in high-level languages such as C. Enable the 10-bit stack pointer feature by setting the stack address mode bit, SA (ACON.2). The bit is cleared following a reset, forcing the device to use an 8-bit stack located in the scratchpad RAM area. When the SA bit is set, the device will address up to 1kB of stack memory in the first 1kB of the internal MOVX memory. The 10-bit stack pointer address is generated by concatenating the lower two bits of the extended stack pointer (ESP;9Bh) and the traditional 8051 stack pointer (SP;81h). The 10-bit stack pointer cannot be enabled when the 4kB of SRAM is mapped as both program and data memory.

ENHANCED DUAL DATA POINTERS

The DS80C390 contains two data pointers, DPTR0 and DPTR1, designed to improve performance in applications that require high data throughput. Incorporating a second data pointer allows the software to greatly speed up block data (MOVX) moves by using one data pointer as a source register and the other as the destination register.

DPTR0 is located at the same address as the original 8051 data pointer, allowing the DS80C390 to execute standard 8051 code with no modifications. The second data pointer, DPTR1, is split between the DPH1 and DPL1 SFRs, similar to the DPTR0 configuration. The active data pointer is selected with the data pointer select bit SEL (DPS.0). Any instructions that reference the DPTR (i.e., MOVX A, @DPTR), will select DPTR0 if SEL = 0, and DPTR1 if SEL = 1. Because the bits adjacent to SEL are not implemented, the state of SEL (and thus the active data pointer) can be quickly toggled by the `INC DPS` instruction without disturbing other bits in the DPS register.

Unlike the standard 8051, the DS80C390 has the ability to decrement as well as increment the data pointers without additional instructions. When the `INC DPTR` instruction is executed, the active DPTR increments or decrements according to the ID1, ID0 (DPS.7-6), and SEL (DPS.0) bits as shown. The inactive DPTR is not affected.

Table 10. Data Pointer Auto Increment/ Decrement Configuration

ID1	ID0	SEL	INC DPTR RESULT
X	0	0	Increment DPTR0
X	1	0	Decrement DPTR0
0	X	1	Increment DPTR1
1	X	1	Decrement DPTR1

Another useful feature of the device is its ability to automatically switch the active data pointer after a DPTR-based instruction is executed. This feature can greatly reduce the software overhead associated with data memory block moves, which toggle between the source and destination registers. When the toggle-select bit (TSL;DPS.5) is set to 1, the SEL bit (DPS.0) is automatically toggled every time one of the following DPTR-related instructions is executed.

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

As a brief example, if TSL is set to 1, then both data pointers can be updated with two INC DPTR instructions. Assume that SEL = 0, making DPTR the active data pointer. The first INC DPTR increments DPTR and toggles SEL to 1. The second instruction increments DPTR1 and toggles SEL back to 0.

```
INC DPTR
INC DPTR
```

CLOCK CONTROL AND POWER MANAGEMENT

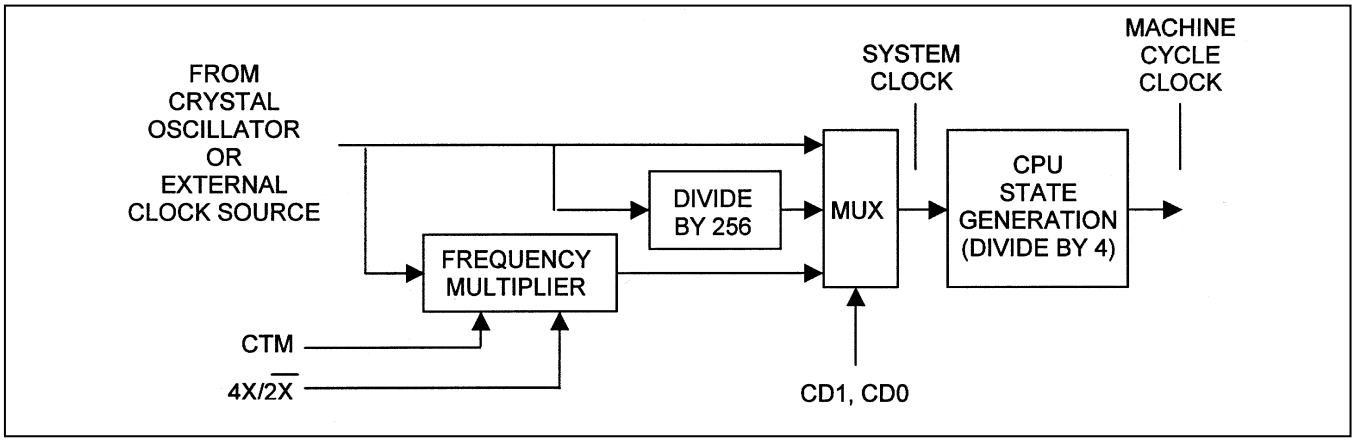
The DS80C390 includes a number of unique features that allow flexibility in selecting system clock sources and operating frequencies. To support the use of inexpensive crystals while allowing full speed operation, a clock multiplier is included in the processor's clock circuit. Also, in addition to the standard 80C32 idle and power-down (Stop) modes, the DS80C390 provides a new power management mode. This mode allows the processor to continue instruction execution, yet at a very low speed to significantly reduce power consumption (below even idle mode). The DS80C390 also features several enhancements to stop mode that make this extremely low-power mode more useful. Each of these features is discussed in detail below.

System Clock Control

As mentioned previously, the microcontroller contains special clock-control circuitry that simultaneously provides maximum timing flexibility and maximum availability and economy in crystal selection. The logical operation of the system clock-divide control function is shown in Figure 29. 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 29. 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 time base 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 time base 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 40MHz. This means that the maximum crystal oscillator or external clock source is 10MHz when using the 4X setting, and 20MHz 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.

Table 11. System Clock Configuration

CD1	CD0	4X/$\overline{2X}$	FUNCTION	CLOCKS PER MACHINE CYCLE	MAX EXTERNAL FREQUENCY (MHz)
0	0	0	Frequency Multiplier (2X)	2	20
0	0	1	Frequency Multiplier (4X)	1	10
0	1	N/A	Reserved	—	—
1	0	N/A	Divide-by-4 (Default)	4	40
1	1	N/A	Power Management Mode	1024	40

The system clock and machine cycle rate changes one machine cycle after the instruction changing the control bits. Note that the change affects all aspects of system operation, including timers and baud rates. The use of the switchback feature, described later, can eliminate many of the problems associated with the PMM.

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-4) 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-4 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/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 65,536 cycles of the external crystal or clock source.
6) Set CD1, CD0 to 00. The frequency multiplier is engaged on the machine cycle following the write to these bits.

OSCILLATOR-FAIL DETECT

The microprocessor contains a safety mechanism called an on-chip oscillator-fail-detect circuit. When enabled, this circuit causes the processor to be held in reset if the oscillator frequency falls below 40kHz. In operation, this circuit complements the watchdog timer. Normally, the watchdog timer is initialized so that it times out and causes a processor reset in the event that the processor loses control. In the event of a crystal or external oscillator failure, however, the watchdog timer does not function and there is the potential for the processor to fail in an uncontrolled state. The use of the oscillator-fail-detect circuit forces the processor to a known state (i.e., reset) even if the oscillator stops.

The oscillator-fail-detect circuitry is enabled when software sets the enable bit OFDE (PCON.4) to 1. Please note that software must use a timed-access procedure (described later) to write this bit. The OFDF (PCON.5) bit also sets to 1 when the circuitry detects an oscillator failure, and the processor is forced into a reset state. This bit can only be cleared to 0 by a power-fail reset or by software. The oscillator-fail-detect circuitry is not activated when the oscillator is stopped due to the processor entering stop mode.

POWER MANAGEMENT MODE (PMM) AND SWITCHBACK

Power consumption in PMM is less than in idle mode, and approximately one quarter of that consumed in divide-by-four mode. While PMM and Idle modes leave the power-hungry internal timers running, PMM runs all clocked functions such as timers at the rate of crystal divided by 1024, rather than crystal divided by 4. Even though instruction execution continues in PMM (albeit at a reduced speed), it still consumes less power than idle mode. As a result there is little reason to use idle mode in new designs.

When enabled, the switchback feature allows serial ports and interrupts to automatically switch back from divide by 1024 (PMM) to divide-by-4 (standard speed) operation. This feature makes it very convenient to use the PMM in real-time applications. Software can simply set the CD1 and CD0 clock control bits to the 4 clocks-per-cycle mode to exit PMM. However, the microcontroller provides hardware alternatives for automatic Switchback to standard speed (divide-by-4) operation.

Setting the SFR bit SWB (PMR.5) to 1 enables the switchback feature. Once it is enabled, and when PMM is selected, two possible events can cause an automatic switchback to divide-by-4 mode. First, if an interrupt occurs and is acknowledged, the system clock reverts from PMM to divide-by-4 mode. For example, if $\overline{\text{INT0}}$ is enabled and the CPU is not servicing a higher priority interrupt, then switchback occurs on $\overline{\text{INT0}}$. However, if $\overline{\text{INT0}}$ is not enabled or the CPU is servicing a higher priority interrupt, then activity on $\overline{\text{INT0}}$ does not cause switchback to occur.

A switchback can also occur when an enabled UART detects the start bit indicating the beginning of an incoming serial character or when the SBUF register is loaded initiating a serial transmission. Note that a serial character's start bit does not generate an interrupt. The interrupt occurs only on reception of a complete serial word. The automatic switchback on detection of a start bit allows timer hardware to return to divide-by-4 operation (and the correct baud rate) in time for a proper serial reception or transmission. So with switchback enabled and a serial port enabled, the automatic switch to divide-by-4 operation occurs in time to receive or transmit a complete serial character as if nothing special had happened.

STATUS

The status register (STATUS;C5h) provides information about interrupt and serial port activity to assist in determining if it is possible to enter PMM. The microprocessor supports three levels of interrupt priority: power-fail, high, and low. The PIP (power-fail priority interrupt status; STATUS.7), HIP (high-priority interrupt status; STATUS.6), and LIP (low-priority interrupt status; STATUS.5) status bits, when set to logic 1, indicate the corresponding level is in service.

Software should not rely on a lower-priority level interrupt source to remove PMM (switchback) when a higher level is in service. Check the current priority service level before entering PMM. If the current service level locks out a desired switchback source, then it would be advisable to wait until this condition clears before entering PMM. Alternately, software can prevent an undesired exit from PMM by intentionally entering a low priority interrupt service level before entering PMM. This will prevent other low priority interrupts from causing a switchback.

Entering PMM during an ongoing serial port transmission or reception can corrupt the serial port activity. To prevent this, a hardware lockout feature ignores changes to the clock divisor bits while the serial ports are active. Serial port activity can be monitored via the serial port activity bits located in the status register.

IDLE MODE

Setting the IDLE bit (PCON.0) invokes the idle mode. Idle leaves internal clocks, serial ports, and timers running. Power consumption drops because memory is not being accessed and instructions are not being executed. Since clocks are running, the idle power consumption is a function of crystal frequency. It should be approximately one-half of the operational power at a given frequency. The CPU can exit idle mode with any interrupt or a reset. Because PMM consumes less power than idle mode, as well as leaving timers and CPU operating, idle mode is no longer recommended for new designs, and is included for backward software compatibility only.

STOP MODE

Setting the STOP bit of the power control register (PCON.1) invokes stop mode. Stop mode is the lowest power state (besides power off) since it turns off all internal clocking. All processor operation ceases at the end of the instruction that sets the STOP bit. The CPU can exit stop mode via an external interrupt, if enabled, or a reset condition. Internally generated interrupts (timer, serial port, watchdog) cannot cause an exit from stop mode because internal clocks are not active in stop mode.

BANDGAP SELECT

The DS80C390 provides two enhancements to stop mode. As described below, the device provides a band-gap reference to determine power-fail interrupt and reset thresholds. The bandgap select bit, BGS (RCON.0), controls the bandgap reference. Setting BGS to 1 keeps the bandgap reference enabled during stop mode. The default or reset condition of the bit is logic 0, which disables the bandgap during stop mode. This bit has no control of the reference during full power, PMM, or idle modes.

With the bandgap reference enabled, the power-fail reset and interrupt are valid means for leaving stop mode. This allows software to detect and compensate for a power-supply sag or brownout, even when in stop mode. In stop mode with the bandgap enabled, I_{CC} is higher compared to with the bandgap disabled. If a user does not require a power-fail reset or interrupt while in stop mode, the bandgap can remain disabled. Only the most power-sensitive applications should disable the bandgap reference in stop mode, as this results in an uncontrolled power-down condition.

RING OSCILLATOR

The second enhancement to Stop mode reduces power consumption and allows the device to restart instantly when exiting stop mode. The ring oscillator is an internal clock that can optionally provide the clock source to the microcontroller when exiting stop mode in response to an interrupt.

During stop mode the crystal oscillator is halted to maximize power savings. Typically, 4ms to 10ms is required for an external crystal to begin oscillating again once the device receives the exit stimulus. The ring oscillator, by contrast, is a free-running digital oscillator that has no startup delay. Setting the ring oscillator select bit, RGSL (EXIF.1), enables the ring oscillator feature. If enabled, the microcontroller uses the ring oscillator as the clock source to exit stop mode, resuming operation in less than 100ns. After 65,536 oscillations of the external clock source (not the ring oscillator), the device clears the ring-oscillator-mode bit, RGMD (EXIF.2), to indicate that the device has switched from the ring oscillator to the external clock source.

The ring oscillator runs at approximately 10MHz but varies over temperature and voltage. As a result, no serial communication or precision timing should be attempted while running from the ring oscillator since the operating frequency is not precise. The default state exits stop mode without using the ring oscillator.

TIMED-ACCESS PROTECTION

Selected SFR bits are critical to operation, making it desirable to protect them against an accidental write operation. The timed-access procedure prevents an errant processor from accidentally altering bits that would seriously affect processor operation. The timed-access procedure requires that the write of a protected bit be immediately preceded by the following two instructions:

```
MOV   0C7h, #0AAh
MOV   0C7h, #55h
```

Writing an AAh followed by a 55h to the timed-access register (location C7h) opens a three-cycle window that allows software to modify one of the protected bits. If the instruction that seeks to modify the protected bit is not immediately preceded by these instructions, the write is ignored. The protected bits are:

WDCON.6	POR	Power-On Reset Flag
WDCON.3	WDIF	Watchdog Interrupt Flag
WDCON.1	EWT	Watchdog Reset Enable
WDCON.0	RWT	Reset Watchdog Timer
RCON.0	BGS	Bandgap Select
ACON.2	SA	Stack Address Mode
ACON.1–0	AM1–AM0	Address Mode Select bits
MCON.7–6	IDM1–IDM0	Internal Memory Configuration and Location bits
MCON.5	CMA	CAN Data Memory Assignment
MCON.3–0	PDCE3–PDCE.0	Program/Data Chip Enables
C0C.3	CRST	CAN 0 Reset
C1C.3	CRST	CAN 1 Reset
P4CNT.6	SBCAN	Single Bus CAN
P4CNT.5–0		Port 4 Pin Configuration Control Bits
P5CNT.2–0	P5.7–P5.5	Configuration Control Bits
COR.7	IRDACK	IRDA Clock Output Enable
COR.6–5	C1BPR7–C1BPR6	CAN 1 Baud Rate Prescale Bits
COR.4–3	C0BPR7–C0BPR6	CAN 0 Baud Rate Prescale Bits
COR.2–1	COD1–COD0	CAN Clock Output Divide Bit 1 and Bit 0
COR.0	CLKOE	CAN Clock Output Enable

EMI REDUCTION

One of the major contributors to radiated noise in an 8051-based system is the toggling of ALE. The microcontroller allows software to disable ALE when not used by setting the ALEOFF (PMR.2) bit to 1. When ALEOFF = 1, ALE automatically toggles during an off-chip MOVX. However, ALE remains static when performing on-chip memory access. The default state of ALEOFF is 0 so ALE normally toggles at a frequency of XTAL/4.

PERIPHERAL OVERVIEW

The DS80C390 provides several of the most commonly needed peripheral functions in microcomputer-based systems. New functions include a second serial port, power-fail reset, power-fail interrupt flag, and a programmable watchdog timer. In addition, the microcontroller contains two CAN modules for industrial communication applications. Each of these peripherals is described in the following paragraphs. More details are available in the *High-Speed Microcontroller User's Guide* and the *DS80C390 Supplement*.

SERIAL PORTS

The microcontroller provides a serial port (UART) that is identical to the 80C52. In addition it includes a second hardware serial port that is a full duplicate of the standard one. This second port optionally uses pins P1.2 (RXD1) and P1.3 (TXD1). It has duplicate control functions included in new SFR locations. The second serial port can alternately be mapped to P5.2 and P5.3 to allow use of both serial ports in nonmultiplexed mode.

Both ports can operate simultaneously but can be at different baud rates or even in different modes. The second serial port has similar control registers (SCON1, SBUF1) to the original. The new serial port can only use Timer 1 for baud-rate generation.

The SCON0 register provides control for serial port 0 while its I/O buffer is SBUF0. The registers SCON1 and SBUF1 provide the same functions for the second serial port. A full description of the use and operation of both serial ports can be found in the *High-Speed Microcontroller User's Guide: DS80C390 Supplement*.

WATCHDOG TIMER

The watchdog is a free-running, programmable timer that can set a flag, cause an interrupt, and/or reset the microcontroller if allowed to reach a preselected timeout. It can be restarted by software.

A typical application uses the watchdog timer as a reset source to prevent software from losing control. The watchdog timer is initialized, selecting the timeout period and enabling the reset and/or interrupt functions. After enabling the reset function, software must then restart the timer before its expiration or the hardware will reset the CPU. In this way, if the code execution goes awry and software does not reset the watchdog as scheduled, the processor is put in a known good state: reset.

Software can select one of four timeout values as controlled by the WD1 and WD0 bits. Timeout values are precise since they are a function of the crystal frequency. When the watchdog times out, it sets the watchdog timer-reset flag (WTRF = WDCON.2), which generates a reset if enabled by the enable watchdog-timer reset (EWT = WDCON.1) bit. Both the enable watchdog-timer reset and the reset watchdog timer control bits are protected by timed-access circuitry. This prevents errant software from accidentally clearing or disabling the watchdog.

The watchdog interrupt is useful for systems that do not require a reset circuit. It set the WDIF (watchdog interrupt) flag 512 clocks before setting the reset flag. Software can optionally enable this interrupt source, which is independent of the watchdog-reset function. The interrupt is commonly used during the debug process to determine where watchdog-reset commands must be located in the application software. The interrupt also can serve as a convenient time base generator or can wake up the processor from power-saving modes.

The clock control (CKCON) and the watchdog control (WDCON) SFRs control the watchdog timer. CKCON.7 and CKCON.6 (WD1 and WD0, respectively) select the watchdog timeout period. Of course, the 4X/$\overline{2X}$ (PMR.3) and CD1:0 (PMR.7:6) system clock-control bits also affect the timeout period. Table 12 shows the timeout selection.

Table 12. Watchdog Timeout Values

4X/$\overline{2X}$	CD1:0	WATCHDOG INTERRUPT TIMEOUT				WATCHDOG RESET TIMEOUT			
		WD1:0 = 00	WD1:0 = 01	WD1:0 = 10	WD1:0 = 11	WD1:0 = 00	WD1:0 = 01	WD1:0 = 10	WD1:0 = 11
1	00	2^{15}	2^{18}	2^{21}	2^{24}	2^{15}+512	2^{18}+512	2^{21}+512	2^{24}+512
0	00	2^{16}	2^{19}	2^{22}	2^{25}	2^{16}+512	2^{19}+512	2^{22}+512	2^{25}+512
x	01	2^{17}	2^{20}	2^{23}	2^{26}	2^{17}+512	2^{20}+512	2^{23}+512	2^{26}+512
x	10	2^{17}	2^{20}	2^{23}	2^{26}	2^{17}+512	2^{20}+512	2^{23}+512	2^{26}+512
x	11	2^{25}	2^{28}	2^{31}	2^{34}	2^{25}+512	2^{28}+512	2^{31}+512	2^{34}+512

Table 12 demonstrates that for a 33MHz crystal frequency, the watchdog timer can produce timeout periods from 3.97ms (2^{17} x 1/33MHz) to over 2 seconds (2.034 = 2^{26} x 1/33MHz) with the default setting of CD1:0 (=10). This wide variation in timeout periods allows very flexible system implementation.

In a typical initialization, the user selects one of the possible counter values to determine the timeout. Once the counter chain has completed a full count, hardware sets the interrupt flag (WDIF = WDCON.3). Regardless of whether the software makes use of this flag, there are then 512 clocks left until the reset flag (WTRF = WDCON.2) is set. Software can enable (1) or disable (0) the reset using the enable watchdog-timer-reset (EWT = WDCON.1) bit.

POWER-FAIL RESET

The microcontroller incorporates an internal precision bandgap voltage reference and comparator circuit that provide a power-on and power-fail reset function. This circuit monitors the processor's incoming power supply voltage (V_{CC}), and holds the processor in reset while V_{CC} is below the minimum voltage level. When power exceeds the reset threshold, a full power-on reset is performed. In this way, this internal voltage monitoring circuitry handles both power-up and power-down conditions without the need for additional external components.

Once V_{CC} has risen above V_{RST}, the device automatically restarts the oscillator for the external crystal and counts 65,536 clock cycles before program execution begins at location 0000h. This helps the system maintain reliable operation by only permitting processor operation when the supply voltage is in a known good state. Software can determine that a power-on reset has occurred by checking the power-on reset flag (POR;WDCON.6). Software should clear the POR bit after reading it.

POWER-FAIL INTERRUPT

The bandgap voltage reference that sets a precise reset threshold also generates an optional early warning power-fail interrupt (PFI). When enabled by software, the processor vectors to ROM address 0033h if V_{CC} drops below V_{PFW}. PFI has the highest priority. The PFI enable is in the watchdog control SFR (EPFI;WDCON.5). Setting this bit to logic 1 enables the PFI. Application software can also read the PFI flag at WDCON.4. A PFI condition sets this bit to 1. The flag is independent of the interrupt enable and must be cleared by software.

EXTERNAL RESET PINS

The DS80C390 has reset input (RST) and reset output ($\overline{\text{RSTOL}}$) pins. The $\overline{\text{RSTOL}}$ pin supplies an active-low reset when the microprocessor is issued a reset from either a high on the RST pin, a timeout of the watchdog timer, a crystal oscillator fail, or an internally detected power fail. The timing of the $\overline{\text{RSTOL}}$ pin is dependent on the source of the reset.

RESET TYPE/SOURCE	$\overline{\text{RSTOL}}$ DURATION
Power-On Reset	65,536 t_{CLCL} (as described in *Power Cycle Timing Characteristics*)
External Reset	<1.25 machine cycles
Power Fail	65,536 t_{CLCL} (as described in *Power Cycle Timing Characteristics*)
Watchdog Timer Reset	2 machine cycles
Oscillator-Fail Detect	65,536 t_{CLCL} (as described in *Power Cycle Timing Characteristics*)

INTERRUPTS

The microcontroller provides 16 interrupt sources with three priority levels. All interrupts, with the exception of the power-fail interrupt, are controlled by a series combination of individual enable bits and a global interrupt-enable, EA (IE.7). Setting EA to 1 allows individual interrupts to be enabled. Clearing EA disables all interrupts regardless of their individual enable settings.

The three available priority levels are low, high, and highest. The highest priority level is reserved for the power-fail interrupt only. All other interrupt priority levels have individual priority bits that, when set to 1, establish the particular interrupt as high priority. In addition to the user-selectable priorities, each interrupt also has an inherent natural priority, used to determine the priority of simultaneously occurring interrupts. The available interrupt sources, their flags, their enables, their natural priority, and their available priority selection bits are identified in Table 13.

Table 13. Interrupt Summary

NAME	DESCRIPTION	VECTOR	NATURAL PRIORITY	FLAG BIT	ENABLE BIT	PRIORITY CONTROL BIT
PFI	Power-Fail Interrupt	33h	0	PFI (WDCON.4)	EPFI (WDCON.5)	N/A
INT0	External Interrupt 0	03h	1	IE0 (TCON.1)**	EX0 (IE.0)	PX0 (IP.0)
TF0	Timer 0	0Bh	2	TF0 (TCON.5)*	ET0 (IE.1)	PT0 (IP.1)
INT1	External Interrupt 1	13h	3	IE1 (TCON.3)**	EX1 (IE.2)	PX1 (IP.2)
TF1	Timer 1	1Bh	4	TF1 (TCON.7)*	ET1 (IE.3)	PT1 (IP.3)
SCON0	TI0 or RI0 from Serial Port 0	23h	5	RI_0 (SCON0.0); TI_0 (SCON0.1)	ES0 (IE.4)	PS0 (IP.4)
TF2	Timer 2	2Bh	6	TF2 (T2CON.7)	ET2 (IE.5)	PT2 (IP.7)
SCON1	TI1 or RI1 from Serial Port 1	3Bh	7	RI_1 (SCON1.0); TI_1 (SCON1.1)	ES1 (IE.6)	PS1 (IP.6)
INT2	External Interrupt 2	43h	8	IE2 (EXIF.4)	EX2 (EIE.0)	PX2 (EIP.0)
INT3	External Interrupt 3	4Bh	9	IE3 (EXIF.5)	EX3 (EIE.1)	PX3 (EIP.1)
INT4	External Interrupt 4	53h	10	IE4 (EXIF.6)	EX4 (EIE.2)	PX4 (EIP.2)
INT5	External Interrupt 5	5Bh	11	IE5 (EXIF.7)	EX5 (EIE.3)	PX5 (EIP.3)
C0I	CAN0 Interrupt	6Bh	12	various	C0IE (EIE.6)	C0IP (EIP.6)
C1I	CAN1 Interrupt	73h	13	various	C1IE (EIE.5)	C1IP (EIP.5)
WDTI	Watchdog Timer	63h	14	WDIF (WDCON.3)	EWDI (EIE.4)	PWDI (EIP.4)
CANBUS	CAN0/1 Bus Activity	7Bh	15	various	CANBIE (EIE.7)	CANBIP (EIP.7)

Unless marked, all flags must be cleared by the application software.
**Cleared automatically by hardware when the service routine is entered.*
***If edge-triggered, flag is cleared automatically by hardware when the service routine is entered. If level-triggered, flag follows the state of the interrupt pin.*

CONTROLLER AREA NETWORK (CAN) MODULE

The DS80C390 incorporates two CAN controllers that are fully compliant with the CAN 2.0B specification. CAN is a highly robust, high-performance communication protocol for serial communications. Popular in a wide range of applications including medical, heating, ventilation, and industrial control, the CAN architecture allows for the construction of sophisticated networks with a minimum of external hardware.

The CAN controllers support the use of 11-bit standard or 29-bit extended acceptance identifiers for up to 15 messages, with the standard 8-byte data field, in each message. Fourteen of the 15 message centers are programmable in either transmit or receive modes, with the 15th designated as a FIFO-buffered, receive-only message center to help prevent data overruns. All message centers support two separate 8-bit media masks and media arbitration fields for incoming message verification. This feature supports the use of higher-level 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 test incoming data with or without the use of the global masks.

Global controls and status registers in each CAN unit allow the microcontroller to evaluate error messages, generate interrupts, locate and validate new data, establish the CAN bus timing, establish identification mask bits, and verify the source of individual messages. Each message center is individually equipped with the necessary status and control bits to establish direction, identification mode (standard or extended), data field size, data status, automatic remote frame request and acknowledgment, and perform masked or non-masked identification acceptance testing.

COMMUNICATING WITH THE CAN MODULE

The microcontroller interface to the CAN modules is divided into two groups of registers. All the global CAN status and control bits as well as the individual message center control/status registers are located in the SFR map. The remaining registers associated with the message centers (data identification, identification/arbitration masks, format, and data) are located in MOVX data space. The CMA bit (MCON.5) allows the message centers to be mapped to either 00EE00h–00EEFFh (CMA = 0) or 401000h–4011FFh (CMA = 1), reducing the possibility of a memory conflict with application software. Note that setting the CMA bit employs a special 23rd address bit that is only used for addressing CAN MOVX memory. The DS80C390's internal architecture requires that the device be in one of the two 22-bit addressing modes when the CMA bit is set to correctly use the 23rd bit and access the CAN MOVX memory. A special lockout feature prevents the accidental software corruption of the control, status, and mask registers while a CAN operation is in progress. Each CAN processor uses 15 message centers. Each message center is composed of four specific areas, including the following:

1) Four arbitration registers (C0MxAR0–3 and C1MxAR0–3) that store either the 11-bit or 29-bit arbitration value. These registers are located in the MOVX memory map.
2) A format register (C0MxF and C1MxF) that informs the CAN processor as to the direction (transmit or receive), the number of data bytes in the message, the identification format (standard or extended), and the optional use of the identification mask or media mask during message evaluation. This register is located in the MOVX memory map.
3) Eight data bytes for storage of 0 to 8 bytes of data (C0MxD0–7 and C1MxD0–7), which are located in the MOVX memory map.
4) Message control registers (C0MxC and C1MxC), which are located in the SFR memory for fast access.

Each of the message centers is identical with the exception of message center 15. Message center 15 has been designed as a receive-only center, and is also buffered through the use of a two-message FIFO to help prevent message loss in a message-overrun situation. The receipt of a third message before either of the first two are read will overwrite the second message, leaving the first message undisturbed.

Modification of the CAN registers located in MOVX memory is protected through the SWINT bits, with one bit protecting each respective CAN module. Consult the description of this bit in the *High-Speed Microcontroller User's Guide: DS80C390 Supplement* for more information. Each CAN module contains a block of control/status/mask registers, 14 functionally identical message centers, plus a 15th message center that is receive-only and incorporates a buffered FIFO. The following tables describe the organization of the message centers located in MOVX space.

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/$\overline{ST}$	MEME	MDME	xxxxF6h
C0M15D0–C0M15D7	CAN 0 MESSAGE 15 DATA BYTE 0–7								xxxxF7h–Feh
	Reserved								xxxxFFh

[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/$\overline{ST}$	MEME	MDME	xxxxF6h
C1M15D0–C1M15D7	CAN 1 MESSAGE 15 DATA BYTE 0–7								xxxxF7h–Feh
	Reserved								xxxxFFh

[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 INTERRUPTS

The DS80C390 supports three interrupts associated with the CAN controllers. One interrupt is dedicated to each CAN controller, providing receive/transmit acknowledgments from each of its 15 message centers. The remaining interrupt, the CAN bus activity interrupt, is used to detect CAN bus activity on the C0RX or C1RX pins.

The message center interrupts are enabled/disabled by individual ETI (transmit) and ERI (receive) enable bits in the corresponding message control register (located in SFR memory) for each message center. All the message center interrupts of each CAN module are Ored together into their respective CAN interrupt. The successful transmission or receipt of a message sets the INTRQ bit in the corresponding message control register (located in SFR memory). This bit can only be cleared through software. In addition, the global interrupt-enable bit (IE.7) and the specific CAN interrupt-enable bit, EIE.6 (CAN0) or EIE.5 (CAN1), must be correctly set to acknowledge a message center interrupt.

Interrupt assertion of error and status conditions associated with the CAN modules is controlled by the ERIE and STIE bits located in the CAN control registers, C0C and C1C.

ARBITRATION AND MASKING

After a CAN module has ascertained that an incoming message is bit-error-free, the identification field of that message is then compared against one or more arbitration values to determine if they will be loaded into a message center. Each enabled message center (see the MSRDY bit in the *CAN Message Control Register*) is tested in order from 1 to 15. The first message center to successfully pass the test receives the incoming message and ends the testing. Using masking registers allows the use of more complex identification schemes, as tests can be made based on bit patterns rather than an exact match between all bits in the identification field and arbitration values. Each CAN processor also incorporates a set of five masks to allow messages with different IDs to be grouped and successfully loaded into a message center. Note that some of these masks are optional as per the bits shown in the *Arbitration/Masking Feature Summary* table (Table 14).

There are several possible arbitration tests, varying according to which message center is involved. If all the enabled tests succeed, the message is loaded into the respective message center. The most basic test, performed on all messages, compares either 11 (CAN 2.0A) or 29 (CAN 2.0B) bits of the identification field to the appropriate arbitration register, based on the EX/$\overline{ST}$ bit in the CAN 0/1 format register. The MEME bit (C0MxF.1 or C1MxF.1) controls whether the arbitration and ID registers are compared directly or through a mask register. A special set of arbitration registers dedicated to message center 15 allows added flexibility in filtering this location.

If desired, further arbitration can be performed by comparing the first two bytes of the data field in each message against two 8-bit media arbitration register bytes. The MDME bit in the CAN message center format registers (C0MxF.0 or C1MxF.0) either disables (MDME = 0) arbitration, or enables (MDME = 1) arbitration using the media ID mask registers 0–1.

If the 11-bit or 29-bit arbitration and the optional media-byte arbitration are successful, the message is loaded into the respective message center. The format register also allows the microcontroller to program each message center to function in a receive or transmit mode through the T/$\overline{R}$ bit, and to use from 0 to 8 data bytes within the data field of a message. Note that message center 15 can only be used in a receive mode. To avoid a priority inversion, the DS80C390 CAN processors 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.

Table 14. Arbitration/Masking Feature Summary

TEST NAME	ARBITRATION REGISTERS	MASK REGISTERS	CONTROL BITS AND CONDITIONS
Standard 11-Bit Arbitration (CAN 2.0A)	Message Center Arbitration Registers 0–1 (Located in each Message Center, MOVX memory)	Standard Global Mask Registers 0–1 (Located in each CAN Control/Status/Mask Register bank, MOVX memory)	EX/$\overline{ST}$ = 0 MEME = 0: Mask register ignored. ID and arbitration register must match exactly. MEME = 1: Only bits corresponding to 1 in mask register are compared in ID and arbitration registers.
Extended 29-Bit Arbitration (CAN 2.0B)	Message Center Arbitration Registers 0–3 (Located in each Message Center, MOVX memory)	Extended Global Mask Registers 0–3 (Located in each CAN Control/Status/Mask Register bank, MOVX memory)	EX/$\overline{ST}$ = 1 MEME = 0: Mask register ignored. ID and arbitration register must match exactly. MEME = 1: Only bits corresponding to 1 in mask register are compared in ID and arbitration registers.
Media Byte Arbitration	Media Arbitration Registers 0–3 (Located in each CAN Control/Status/Mask Register bank, MOVX memory)	Media ID Mask Registers 0–1 (Located in each CAN Control/Status/Mask Register bank, MOVX memory)	MDME = 0: Media byte arbitration disabled. MDME = 1: Only bits corresponding to 1 in Media ID mask register are compared between data bytes 1 and 2 and Media arbitration registers.
Message Center 15, Standard 11-Bit Arbitration (CAN 2.0A)	Message Center 15 Arbitration Registers 0–1 (Located in Message Center 15, MOVX memory)	Message Center 15 Mask Registers 0–1 (Located in each CAN Control/Status/Mask Register bank, MOVX memory)	EX/$\overline{ST}$ = 0 MEME = 0: Mask register ignored. ID and arbitration register must match exactly. MEME = 1: Message center 15 mask registers are ANDed with Global Mask register. Only bits corresponding to 1 in resulting value are compared in ID and arbitration registers.
Message Center 15, Extended 29-Bit Arbitration (CAN 2.0B)	Message Center 15 Arbitration Registers 0–3 (Located in Message Center 15, MOVX memory)	Message Center 15 Mask Registers 0–3 (Located in each CAN Control/Status/Mask Register bank, MOVX memory)	EX/$\overline{ST}$ = 1 MEME = 0: Mask register ignored. ID and arbitration register must match exactly. MEME = 1: Message center 15 mask registers are ANDed with Global Mask register. Only bits corresponding to 1 in resulting value are compared in ID and arbitration registers.

MESSAGE BUFFERING/OVERWRITE

If a message center is configured for reception ($T/\overline{R}$ = 0) and the previous message has not been read (DTUP = 1), then the disposition of an incoming message to that message center is controlled by the WTOE bit (located in CAN Arbitration Register 3 of each message center). When WTOE = 0, the incoming message is discarded and the current message is untouched.

If the WTOE bit is set, the incoming message is received and written over the existing data bytes in that message center. The receiver overwrite bit (ROW) is also set in the corresponding message center control register, located in SFR memory.

Message center 15 is unique in that it incorporates a buffer that can receive up to two messages without loss. If a message is received by message center 15 while it contains an unread message, the new incoming message is held in an internal buffer. When the CAN processor reads the message-center-15 memory location and then clears DTUP = INTRQ = EXTRQ = 0, the contents of the internal buffer is automatically loaded into the message-center-15 MOVX-memory location.

The message-center-15 WTOE bit controls what happens if a third message is received when both the message-center-15 MOVX-memory location and the buffer contain unread messages. If WTOE = 0, the new message is discarded, leaving the message-center-15 MOVX-memory location and the buffer untouched. If WTOE = 1, then the third message writes over the buffered message but leaves the message-center-15 MOVX-memory location untouched.

ERROR COUNTER INTERRUPT GENERATION

Each CAN module can be independently configured to alert the microprocessor when either 96 or 128 errors have been detected by the transmit or receive error counters. The error count select bit, ERCS (C0C.1 or C1C.1) selects whether the limit is 96 (ERCS = 0) or 128 (ERCS = 1) errors. When the error limit is exceeded, the CAN error count exceeded bit, CECE (C0S.6 or C1S.6), bit is set. If the ERIE, C0IE (or C1IE), and EA SFR bits are configured, an interrupt is generated. If the ERCS bit is set, the device generates an interrupt when the CECE bit is set or cleared, if the interrupt is enabled.

BIT TIMING

Bit timing of the CAN transmission can be adjusted per the CAN 2.0B specification. The CAN 0/1 bus timing register zero (C0BT0 and C1BT0)—located in the control/status/mask register block in MOVX memory—controls the PHASE_SEG1 and PHASE_SEG2 time segments as well as the baud-rate prescaler (BPR5–BPR0). The CAN 0/1 bus timing register one (C0BT1 and C1BT1) contains the controls for the sampling rate and the number of clock cycles assigned to the Phase Segment 1 and 2 portions of the nominal bit time. The values of both bus timing registers are automatically loaded into the CAN processor following each software change of the SWINT bit from a 1 to a 0 by the microcontroller. The bit timing parameters must be set before starting operation of the CAN processor. These registers can only be modified during a software initialization, (SWINT = 1), when the CAN processor 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 processor, the software cannot clear the SWINT bit when TSEG1 and TSEG2 are both cleared to 0.

PACKAGE INFORMATION

For the latest package outline information and land patterns, go to **www.maxim-ic.com/packages**.

PACKAGE TYPE	PACKAGE CODE	DOCUMENT NO.
68 PLCC	Q68-1	**21-0049**
64 LQFP	C64L-2	**21-0083**

REVISION HISTORY

REVISION	DESCRIPTION
062299	Initial preliminary release.
090799	Clarifies that unused/unimplemented bits in the CAN MOVX SRAM read 0. Corrected the t_{MCS} time period table. Corrected multiplexed 2-cycle date memory $\overline{CEO-3}$ read figure to show $\overline{RD}$ and $\overline{WR}$ inactive.
110199	Corrected P5.2 and P5.3 pin descriptions. Corrected description of sequence to activate the crystal frequency multiplier. Corrected references to PQFP to read LQFP. Added $\overline{RSTOL}$ timing information.
032904	Official release (removed "preliminary" status). Abs max soldering temp now references JEDEC standard. AC and DC specifications updated to reflect final characterization data. Clarified DC characteristics Note 6 concerning port 4 and 5. Removed Figure 1. *Typical I_{CC} vs. Frequency.* Added t_{LLAX3} specification (identical to t_{LLAX2}). Clarified that t_{RLAZ} is held weak latch until overdriven by external memory. Removed t_{PXIZ}, t_{PHAV}, t_{PHWL}, and t_{PHRL} from nonmultiplexed address/data bus table. Corrected PSEN trace in Figure 10 to not show assertion during MOVX write. Corrected Table 3 to show unnecessary steps during 16/16 divide. Supplied approximate oscillator-fail detection frequency. Removed text references to Stop mode current. Corrected location of PT2 in Table 14.
022305	In *Absolute Maximum Ratings* section (page 2): Removed "A" from IPC/JEDEC J-STD-020A specification to support lead-free devices. In *DC Electrical Characteristics* table (page 2): Changed V_{PFW} MIN to 4.10V from 4.20V Changed V_{PFW} MAX to 4.60V from 4.55V Changed V_{RST} MIN to 3.85V from 3.95V Changed V_{RST} MAX to 4.35V from 4.3V Changed V_{IH2} MIN reference to 0.7 x V_{CC} from 0.7 x V_{DD} Added Note 10 In *AC Electrical Characteristics* table (page 3): Added note to (now) Note 11 that AC timing is characterized and guaranteed by design but is not production tested.
060805	Added lead-free part numbers to *Ordering Information* table.
110905	Added new paragraph to page 33 stating "Software must ensure that the input value for the normalize operation is not zero or the function will not complete. Compilers such as the one from Keil Software have updated their libraries and compensate for this condition." Table 3: clarified text under "Normalize" function. Changed "Configure MCNTO register as required." To "Load MCNT0 with 00h."
050115	Removed automotive references in the *Applications* and *CONTROLLER AREA NETWORK (CAN) MODULE* sections

Maxim Integrated cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim Integrated product. No circuit patent licenses are implied. Maxim Integrated reserves the right to change the circuitry and specifications without notice at any time.

Maxim Integrated Products, 160 Rio Robles, San Jose, CA 95134 408-601-1000

© 2015 Maxim Integrated Products \
The Maxim logo is a registered trademark of Maxim Integrated Products, Inc.

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View DS80C390-FNR+ on WIN SOURCE
- Maxim Integrated Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management