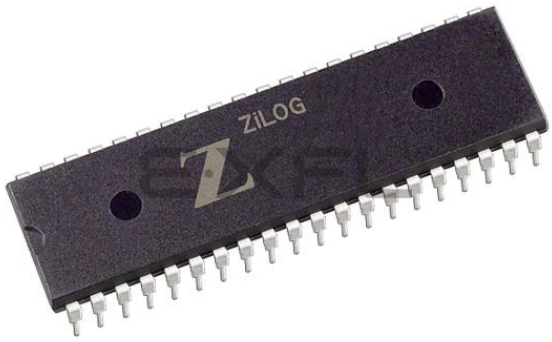


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	Z8
Core Size	8-Bit
Speed	20MHz
Connectivity	UART/USART
Peripherals	DMA
Number of I/O	32
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	4.75V ~ 5.25V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.620", 15.75mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z88c0020psc

ter pointer remains unchanged, the three bits from the address always point to an address within the same eight registers.

The register pointers can be moved by changing the five high bits in control registers R214 for RP0 and R215 for RP1.

The working registers can also be accessed by using full 8-bit addressing. When an 8-bit logical address in the range 192 to 207 (CO to CF) is specified, the lower nibble is used similarly to the 4-bit addressing described above. This is shown in section b of Figure 9.

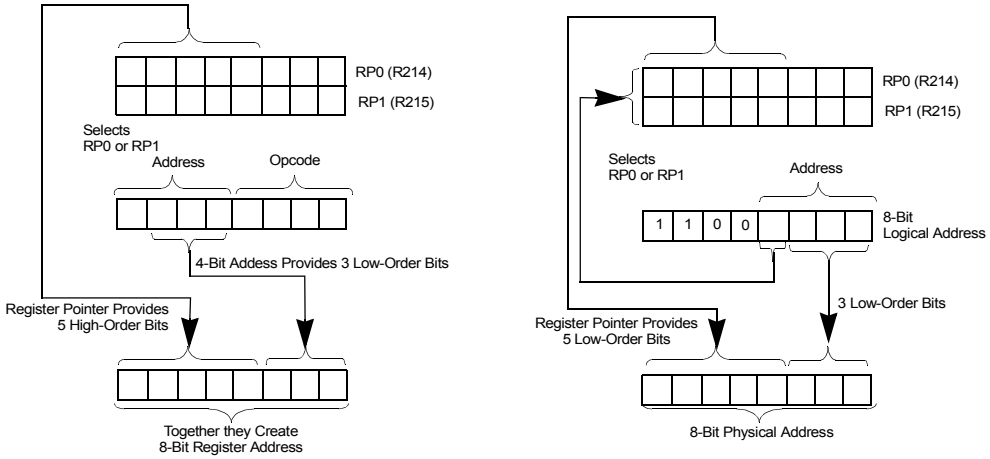


Figure 9.Working Register Window

Since any direct access to logical addresses 192 to 207 involves the register pointers, the physical registers 192 to 207 can be accessed only when selected by a register pointer. After a reset, RPO points to R192 and RP1 points to R200.

Register List

Super-8 Registers lists the Super8 registers. For more details, see Figure 10.

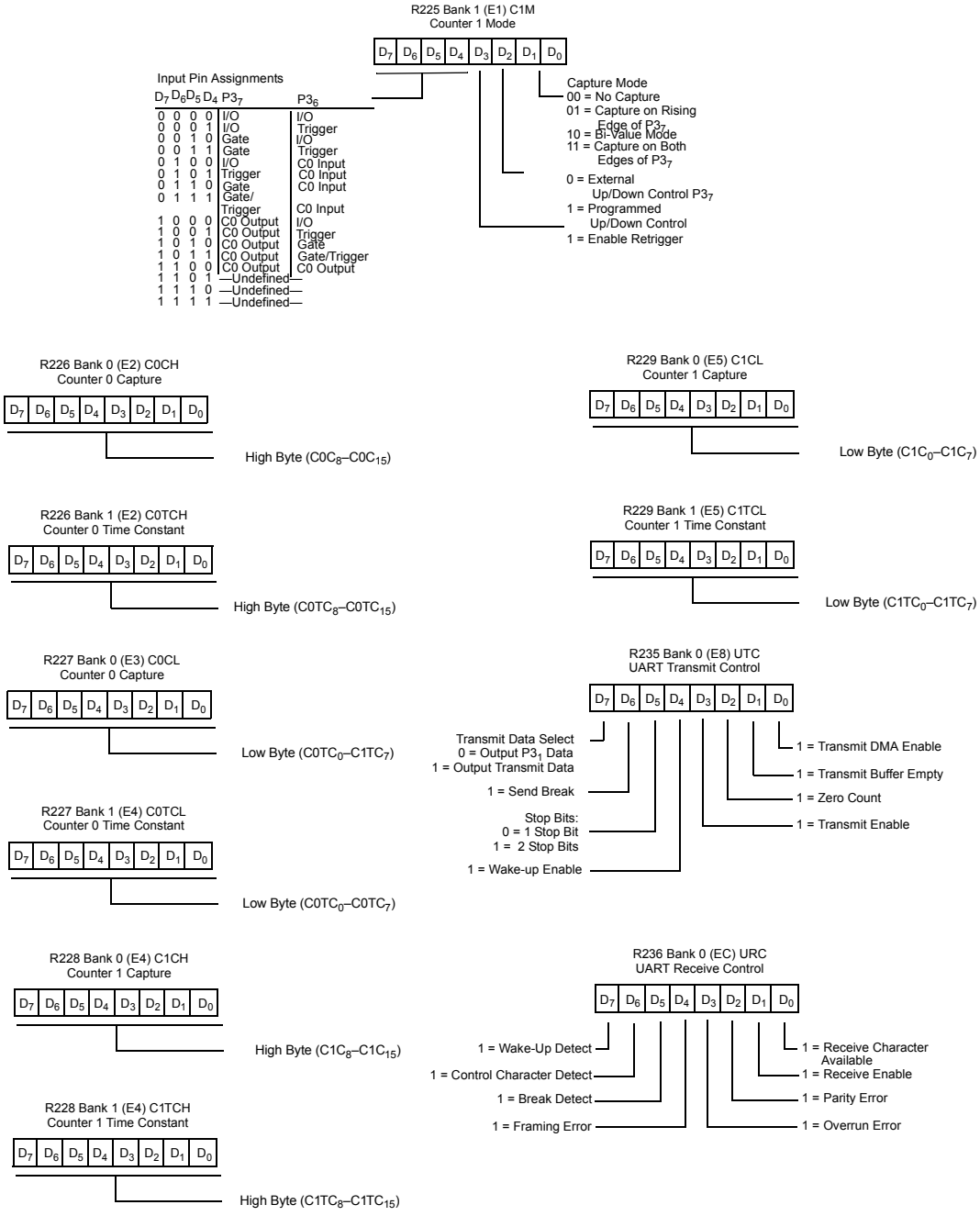


Figure 12.Mode and Control Registers (Continued)

Port 1

In the ROMless device, Port 1 is configured as a byte-wide address/data port. It provides a byte-wide multiplexed address/data path. Additional address lines can be added by configuring Port 0.

The ROM and Protopack Port 1 can be configured as above or as an I/O port; it can be a byte-wide input, open-drain output, or push-pull output. It can be placed under handshake control or handshake channel 0.

Ports 2 and 3

Ports 2 and 3 provide external control inputs and outputs for the UART, handshake channels, and counter/timers. The pin assignments appear in Table 3.

Bits not used for control I/O can be configured as general-purpose I/O lines and/or external interrupt inputs.

Those bits configured for general I/O can be configured individually for input or output. Those configured for output can be individually configured for open-drain or push-pull output.

All Port 2 and 3 input pins are Schmitt-triggered.

The port address for Port 2 is R210, and for Port 3 is R211.

Table 17.Pin Assignments for Ports 2 and 3

Port 2		Port 3	
Bit	**Function**	**Bit**	**Function**
0	UART receive clock	0	UART receive data
1	UART transmit clock	1	UART transmit data
2	Reserved	2	Reserved
3	Reserved	3	Reserved
4	Handshake 0 input	4	Handshake 1 input/$\overline{\text{WAIT}}$
5	Handshake 0 output	5	Handshake 1 output/$\overline{\text{DM}}$
6	Counter 0 input	6	Counter 1 input
7	Counter 0 I/O	7	Counter 1 I/O

Port 4

Port 4 can be configured as I/O only. Each bit can be configured individually as input or output, with either push-pull or open-drain outputs. All Port 4 inputs are Schmitt-triggered.

Port 4 can be placed under handshake control of handshake channel 0. Its register address is R212.

UART

The UART is a full-duplex asynchronous channel. It transmits and receives independently with 5 to 8 bits per character, has options for even or odd bit parity, and a wake-up feature.

Data can be read into or out of the UART via R239, Bank 0. This single address is able to serve a full-duplex channel because it contains two complete 8-bit registers-one for the transmitter and the other for the receiver.

Pins

The UART uses the following Port 2 and 3 pins:

Port/Pin	UART Function
2/0	Receive Clock
3/0	Receive Data
2/1	Transmit Clock
3/1	Transmit Data

Transmitter

When the UART's register address is specified as the destination (dst) of an operation, the data is output on the UART, which automatically adds the start bit, the programmed parity bit, and the programmed number of stop bits. It can also add a wake-up bit if that option is selected.

If the UART is programmed for a 5-, 6-, or 7-bit character, the extra bits in R239 are ignored.

Serial data is transmitted at a rate equal to 1, 1/16, 1/32 or 1/64 of the transmitter clock rate, depending on the programmed data rate. All data is sent out on the falling edge of the clock input.

Carry Flag. This flag is set to 1 if the result from an arithmetic operation generates a carry out of, or a borrow into, bit 7.

After rotate and shift operations, it contains the last value shifted out of the specified register.

It can be set, cleared, or complemented by instructions.

Condition Codes

The flags C, Z, S, and V are used to control the operation of conditional jump instructions.

The opcode of a conditional jump contains a 4-bit field called the condition code (cc). This specifies under which conditions it is to execute the jump. For example, a conditional jump with the condition code for "equal" after a compare operation only jumps if the two operands are equal.

The condition codes and their meanings are given in Condition Codes and Meanings.

Addressing Modes

All operands except for immediate data and condition codes are expressed as register addresses, program memory addresses, or data memory addresses. The addressing modes and their designations are:

- Register (R)
- Indirect Register (IR)
- Indexed (X)
- Direct (DA)
- Relative (RA)
- Immediate (IM)
- Indirect (IA)

Table 18.Condition Codes and Meanings

Binary	Mnemonic	Flags	Meaning
0000	F	-	Always false
1000	-	-	Always true
0111[1]	C	C = 1	Carry
1111[1]	NC	C = 0	No carry

Table 18.Condition Codes and Meanings

Binary	Mnemonic	Flags	Meaning
0110[1]	Z	Z = 1	Zero
1110[1]	NZ	Z = 0	Not zero
1101	PL	S = 0	Plus
0101	MI	S = 1	Minus
0100	OV	V = 1	Overflow
1100	NOV	V = 0	No overflow
0110[1]	EQ	Z = 1	Equal
1110[1]	NE	Z = 0	Not equal
1001	GE	(S XOR V)= 0	Greater than or equal
0001	LT	(S XOR V)= 1	Less than
1010	GT	(Z OR (S XOR V))= 0	Greater than
0010	LE	(Z OR (S XOR V))= 1	Less than or equal
1111[1]	UGE	C=0	Unsigned greater than or equal
0111[1]	ULT	C=1	Unsigned less than
1011	UGT	(C = 0 AND Z = 0)= 1	Unsigned greater than
0011	ULE	(C OR Z)= 1	Unsigned less than or equal

1. Has condition codes that relate to two different mnemonics but test the same flags. For example, Z and EQ are both True if the Zero flag is set, but after an ADD instruction, Z would probably be used, while after a CP instruction, EQ would probably be used.

Registers can be addressed by an 8-bit address in the range of 0 to 255. Working, registers can also be addressed using 4-bit addresses, where five bits contained in a register pointer (R218 or R219) are concatenated with three bits from the 4-bit address to form an 8-bit address.

Registers can be used in pairs to generate 16-bit program or data memory addresses.

Notation and Encoding

The instruction set notations are described in Table 5.

Functional Summary of Commands

Figure 17 shows the formats followed by a quick reference guide to the commands.

Table 19.Instruction Set Notations

Notation	Meaning	Notation	Meaning
cc	Condition code (see Table 4)		DA Direct address (between 0 and 65535)
r	Working register (between 0 and 15)		RA Relative address
rb	Bit of working register	IM	Immediate
r0	Bit 0 of working register	IML	Immediate long
R	Register or working register		dst Destination operand
RR	Register pair or working register pair (Register always start on an even-number boundary)	pairs	src Source operand
		@	Indirect
IA	Indirect address	SP	Stack
Ir	Indirect working register	PC	Program
IR	Indirect register or indirect working register		IP
Irr	Indirect working register pair	FLAGS	Flags
IRR	Indirect register pair or indirect working register	pair	RP
X	Indexed	#	Immediate
XS	Indexed, short offset	%	Hexadecimal
XL	Indexed, long offset	OPC	Opcode

Table 20.Instruction Summary (Continued)

Instruction and Operation	Address Mode dst	Address Mode src	Opcode Byte (Hex)	Flags Affected C	Z	S	V	D	H
SMR(0)←0									
DIV dst, src									
dst ÷ src	RR	R	94	*	*	*	*	-	-
dst (Upper)←Quotient	RR	IR	95						
dst (Lower)←Remainder	RR	IM	96						
DJNZ r, dst	RA	r	rA	-	-	-	-	-	-
r←r - 1			(r = 0 to F)						
if r = 0									
PC←PC + dst									
EI			9F	-	-	-	-	-	-
SMR(0)←1									
ENTER			1F	-	-	-	-	-	-
SP←SP - 2									
@SP←IP									
IP←PC									
PC←@IP									
IP←IP + 2									
EXIT			2F	-	-	-	-	-	-
IP←@SP									
SP←SP + 2									
PC←@IP									
IP←IP + 2									
INC dst	r		rE	-	*	*	*	-	-
dst←dst + 1			(r = 0 to F)						
	R		20						
	IR		21						

Table 20.Instruction Summary (Continued)

Instruction and Operation	Address Mode dst	Address Mode src	Opcode Byte (Hex)	Flags Affected C	Z	S	V	D	H
PC←@SP; SP←SP + 2									
RL dst C←dst(7) dst(0)←dst(7) dst(N + 1)←dst(N) N = 0 to 6	R IR		90 91	*	*	*	*	-	-
RLC dst dst(0)←C C←dst(7) dst(N + 1)←dst(N) N = 0 to 6	R IR		10 11	*	*	*	*	-	-
RR dst C←dst(0) dst(7)←dst(0) dst(N)←dst(N + 1) N = 0 to 6	R IR		E0 E1	*	*	*	*	-	-
RRC dst C←dst(0) dst(7)←C dst(N)←dst(N + 1) N = 0 to 6	R IR		C0 C1	*	*	*	*	-	-
SB0 BANK←0			4F	-	-	-	-	-	-
SB1 BANK←1			5F	-	-	-	-	-	-
SBC dst, src dst←dst – src – C	Note[1]		3[ ]	*	*	*	*	1	*
SCF			DF	1	-	-	-	-	-

INSTRUCTIONS

Table 22.Super8 Instructions

Mnemonic	Operands	Instruction
Load Instructions		
CLR	dst	Clear
LD	dst, src	Load
LDB	dst, src	Load bit
LDC	dst, src	Load program memory
LDE	dst, src	Load data memory
LDCD	dst, src	Load program memory and decrement
LDED	dst, src	Load data memory and decrement
LDCI	dst, src	Load program memory and increment
LDEI	dst, src	Load data memory and increment
LDCPD	dst, src	Load program memory with pre-decrement
LDEPD	dst, src	Load data memory with pre-decrement
LDCPI	dst, src	Load program memory with pre-increment
LDEPI	dst, src	Load data memory with pre-increment
LDW	dst, src	Load word
POP	dst	Pop stack
POPUD	dst, src	Pop user stack (decrement)
POPUI	dst, src	Pop user stack (increment)
PUSH	src	Push stack
PUSHUD	dst, src	Push user stack (decrement)
PUSHUI	dst, src	Push user stack (increment)
Arithmetic Instructions		
ADC	dst, src	Add with carry
ADD	dst, src	Add
CP	dst, src	Compare
DA	dst	Decimal adjust
DEC	dst	Decrement

The levels are shown in Figure 20.

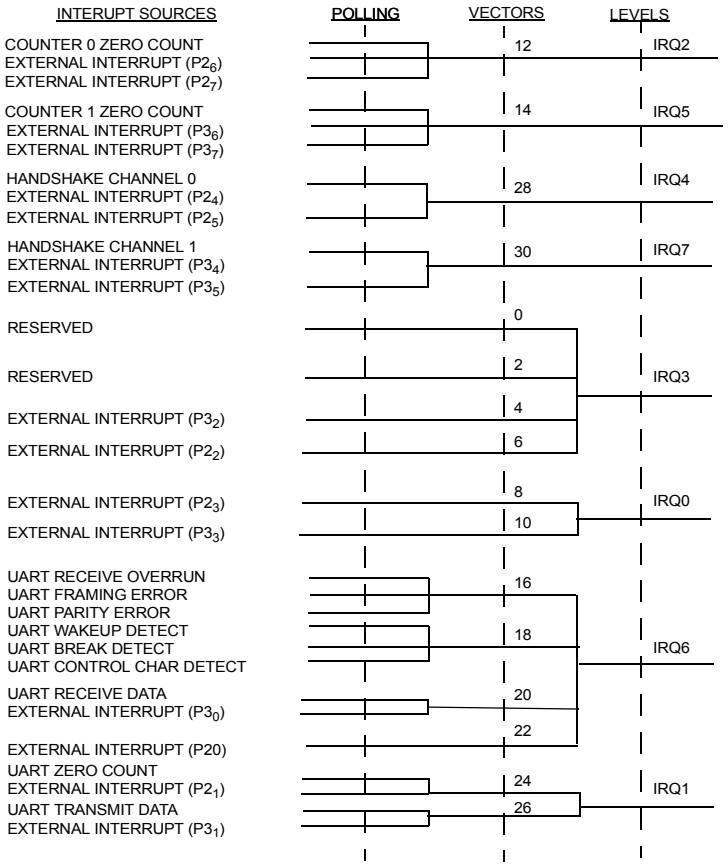


Figure 20.Interrupt Levels and Vectors

Enables

- Interrupts can be enabled or disabled as follows:
- Interrupt enable/disable. The entire interrupt structure can be enabled or disabled by setting bit 0 in the System Mode register (R222).
- Level enable. Each level can be enabled or disabled by setting the appropriate bit in the Interrupt Mask register (R221).
- Level priority. The priority of each level can be controlled by the values in the Interrupt Priority register (R255, Bank 0).
- Source enable/disable. Each interrupt source can be enabled or disabled in the sources' Mode and Control register.

Service Routines

Before an interrupt request can be granted, a) interrupts must be enabled, b) the level must be enabled, c) it must be the highest priority interrupting level, d) it must be enabled at the interrupting source, and e) it must have the highest priority within the level.

If all this occurs, an interrupt request is granted.

The Super8 then enters an interrupt machine cycle that completes the following sequence:

- It resets the Interrupt Enable bit to disable all subsequent interrupts.
- It saves the Program Counter and status flags on the stack.
- It branches to the address contained within the vector location for the interrupt.
- It passes control to the interrupt servicing routine.

When the interrupt servicing routine has serviced the interrupt, it should issue an interrupt return (IRET) instruction. This restores the Program Counter and status flags and sets the Interrupt Enable bit in the System Mode register.

Fast Interrupt Processing

The Super8 provides a feature called fast interrupt processing, which completes the interrupt servicing in 6 clock periods instead of the usual 22.

Two hardware registers support fast interrupts. The Instruction Pointer (IP) holds the starting address of the service routine, and saves the PC value when a fast interrupt occurs. A dedicated register, FLAG', saves the contents of the FLAGS register when a fast interrupt occurs.

To use this feature, load the. address of the service routine in the Instruction Pointer, load the level number into the Fast Interrupt Select field, and turn on the Fast Interrupt Enable bit in the System Mode register.

When an interrupt occurs in the level selected for fast interrupt processing, the following occurs:

- The contents of the Instruction Pointer and Program Counter are swapped.
- The contents of the Flag register are copied into FLAG'.
- The Fast Interrupt Status Bit in FLAGS is set.
- The interrupt is serviced.
- When IRET is issued after the interrupt service outline is completed, the Instruction Pointer and Program Counter are swapped again.

- The contents of FLAG' are copied back into the Flag register
- The Fast Interrupt Status bit in FLAGS is cleared.

The interrupt servicing routine selected for fast processing should be written so that the location after the IRET instruction is the entry point the next time the (same) routine is used.

Level or Edge Triggered

Because internal interrupt requests are levels and interrupt requests from the outside are (usually) edges, the hardware for external interrupts uses edge-triggered flip-flops to convert the edges to levels.

The level-activated system requires that interrupt-serving software perform some action to remove the interrupting source. The action involved in serving the interrupt may remove the source, or the software may have to actually reset the flip-flops by writing to the corresponding Interrupt Pending register.

STACK OPERATION

The Super8 architecture supports stack operations in the register file or in data memory. Bit 1 in the external Memory Timing register (R254 bank 0) selects between the two.

Register pair 216-217 forms the Stack Pointer used for all stack operations. R216 is the MSB and R217 is the LSB.

The Stack Pointer always points to data stored on the top of the stack. The address is decremented prior to a PUSH and incremented after a POP.

The stack is also used as a return stack for CALLs and interrupts. During a CALL, the contents of the PC are saved on the stack, to be restored later. Interrupts cause the contents of the PC and FLAGS to be saved on the stack, for recovery by IRET when the interrupt is finished.

When the Super8 is configured for an internal stack (using the register file), R217 contains the Stack Pointer. R216 may be used as a general-purpose register, but its contents are changed if an overflow or underflow, occurs as the result of incrementing or decrementing the stack address during normal stack operations.

User-Defined Stacks

The Super8 provides for user-defined stacks in both the register file and program or data memory. These can be made to increment or decrement on a push by the choice of opcodes. For example, to implement a stack that grows from low

DC CHARACTERISTICS

Table 23.DC Characteristics

Symbol	Parameter	Min	Max	Unit	Condition
V_{CH}	Clock Input High Voltage	3.8	V_{CC}	V	Driven by External Clock Generator
V_{CL}	Clock Input Low Voltage	-0.3	0.8	V	Driven by External Clock Generator
V_{IH}	Input High Voltage	2.2	V_{CC}	V	
V_{IL}	Input Low Voltage	-0.3	0.8	V	
V_{RH}	Reset Input High Voltage	3.8	V_{CC}	V	
V_{RL}	Reset Input Low Voltage	-0.3	0.8	V	
V_{OH}	Output High Voltage	2.4		V	I_{OH} = -400 µA
V_{OL}	Output Low Voltage		0.4	V	I_{OL} = +4.0 mA
I_{IL}	Input Leakage	-10	10	µA	
I_{OL}	Output Leakage	-10	10	µA	
I_{IR}	Reset Input Current		-50	µA	
I_{CC}	VCC Supply Current		320	rnA	

INPUT HANDSHAKE TIMING

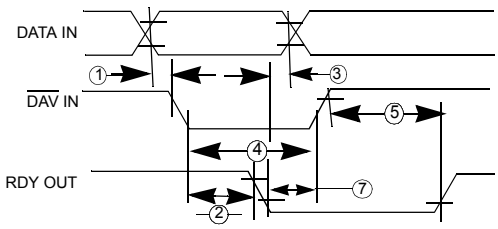


Figure 22.Fully Interlocked Mode

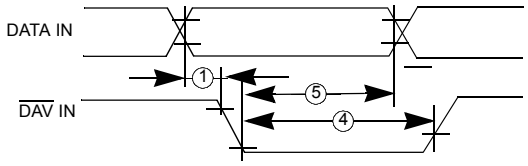


Figure 23.Strobed Mode

AC CHARACTERISTICS (20 MHz)

Input Handshake

Table 24.AC Characteristics (20 MHz) Input Handshake

Number	Symbol	Parameter	Min	Max	Notes[1,2]
1	TsDI(DAV)	Data In to Setup Time	0		
2	TdDAVIf(RDY)	$\overline{DAV}$ ↓ Input to RDY ↓ Delay		200	Note[3]
3	ThDI(RDY)	Data In Hold Time from RDY ↓	0		
4	TwDAV	$\overline{DAV}$ In Width	45		
5	ThDI(DAV)	Data In Hold Time from $\overline{DAV}$ ↓	130		
6	TdDAV(RDY)	$\overline{DAV}$ ↑ Input to RDY ↑ Delay		100	Note[4]
7	TdRDYf(DAV)	RDY ↓ Output to $\overline{DAV}$ ↑ Delay	0		

1. Times are preliminary and subject to change.
2. Times given are in ns.
3. Standard Test Load
4. This time assumes user program reads data before $\overline{DAV}$ Input goes high. RDY does not go high before data is read.

OUTPUT HANDSHAKE TIMING

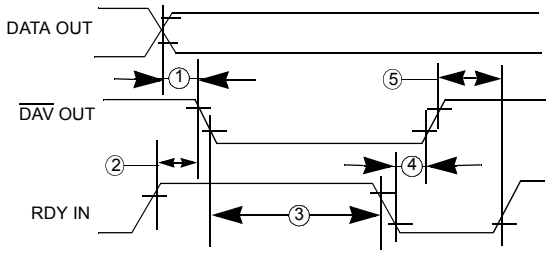


Figure 24.Fully Interlocked Mode

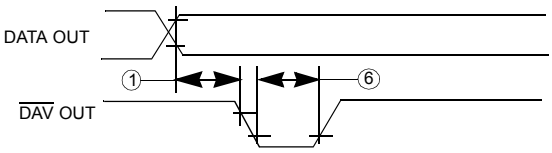


Figure 25.Strobed Mode

AC CHARACTERISTICS (12 MHz, 20 MHz)

Output Handshake

Table 25.AC Characteristics (12 MHz, 20 MHz) Output Handshake

Number	Symbol	Parameter	Min	Max	Notes[1,2]
1	TdDO(DAV)	Data Out to $\overline{DAV}$ ↓ Delay	90		Note[3,4]
2	TdRDYr(DAV)	RDY ↑ Input to $\overline{DAV}$ ↓ Delay	0	110	Note[3]
3	TdDAVOf(RDY)	$\overline{DAV}$ ↓ Output to RDY ↓ Delay	0		
4	TdRDYf(DAV)	RDY ↓ Input to $\overline{DAV}$ ↑ Delay	0	110	Note[3]
5	TdDAVOr(RDY)	$\overline{DAV}$ ↑ Output to RDY ↑ Delay	0		
6	TwDAVO	$\overline{DAV}$ Output Width	150		Note[4]

1. Times are preliminary and subject to change.
2. Times given are in ns.
3. Standard Test Load
4. Time given is for zero value in Deskew Counter. For nonzero value of n where n = 1, 2,. . . 15 add 2 x n x TpC to the given time.

AC CHARACTERISTICS (12 MHz)

Read /Write

Table 26.AC Characteristics (12 MHz) Read/Write

			Normal Timing		Extended Timing		
Number	**Symbol**	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Notes[1,2]**
1	TdA(AS)	Address Valid to $\overline{AS}$ ↑ Delay	35		115		
2	TdAS(A)	$\overline{AS}$ ↑ to Address Float Delay	65		150		
3	TdAS(DR)	$\overline{AS}$ ↑ to Read Data Required Valid		270		600	Note[3]
4	TWAS	$\overline{AS}$ Low Width	65		150		
5	TdA(DS)	Address Float to $\overline{DS}$ ↓	20		20		
6a	TWDS(Read)	$\overline{DS}$ (Read) Low Width	225		470		Note[3]
6b	TwDS(Write)	$\overline{DS}$ (Write) Low Width	130		295	1	Note[3]
7	TdDS(DR)	$\overline{DS}$ ↓ to Read Data Required Valid		180		420	Note[3]
8	ThDS(DR)	Read Data to $\overline{DS}$ ↑ Hold Time	0		0		
9	TdDS(A)	$\overline{DS}$ ↑ to Address Active Delay	50		135		
10	TdDS(AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	60		145		
11	TdDO(DS)	Write Data Valid to $\overline{DS}$ (Write) ↓ Delay	35		115		
12	TdAS(W)	$\overline{AS}$ ↑ to Wait Delay		220		600	Note[4]
13	ThDS(W)	$\overline{DS}$ ↑ to Wait Hold Time	0		0		
14	TdRW(AS)	R/$\overline{W}$ Valid to $\overline{AS}$ ↑ Delay	50		135		

1. All times are in ns and are for 12 MHz input frequency.
2. Timings are preliminary and subject to change
3.) Wait states add 167 ns to these times.
4.) Auto-wait states add 167 ns to this time..

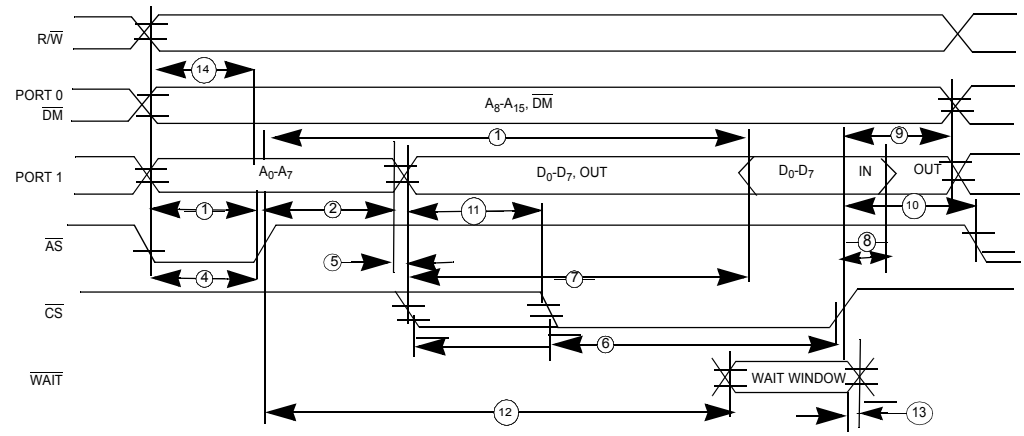


Figure 26.External Memory Read and Write Timing

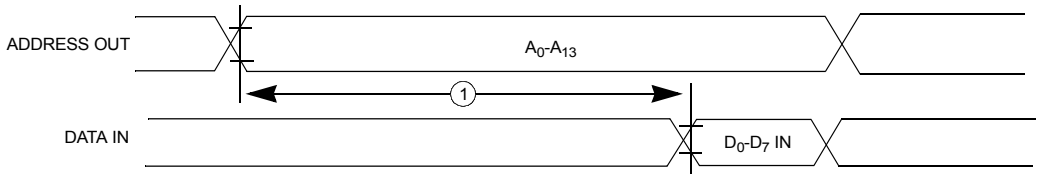


Figure 27.EPROM Read Timing

AC CHARACTERISTICS (20 MHz)

EPROM Read Cycle

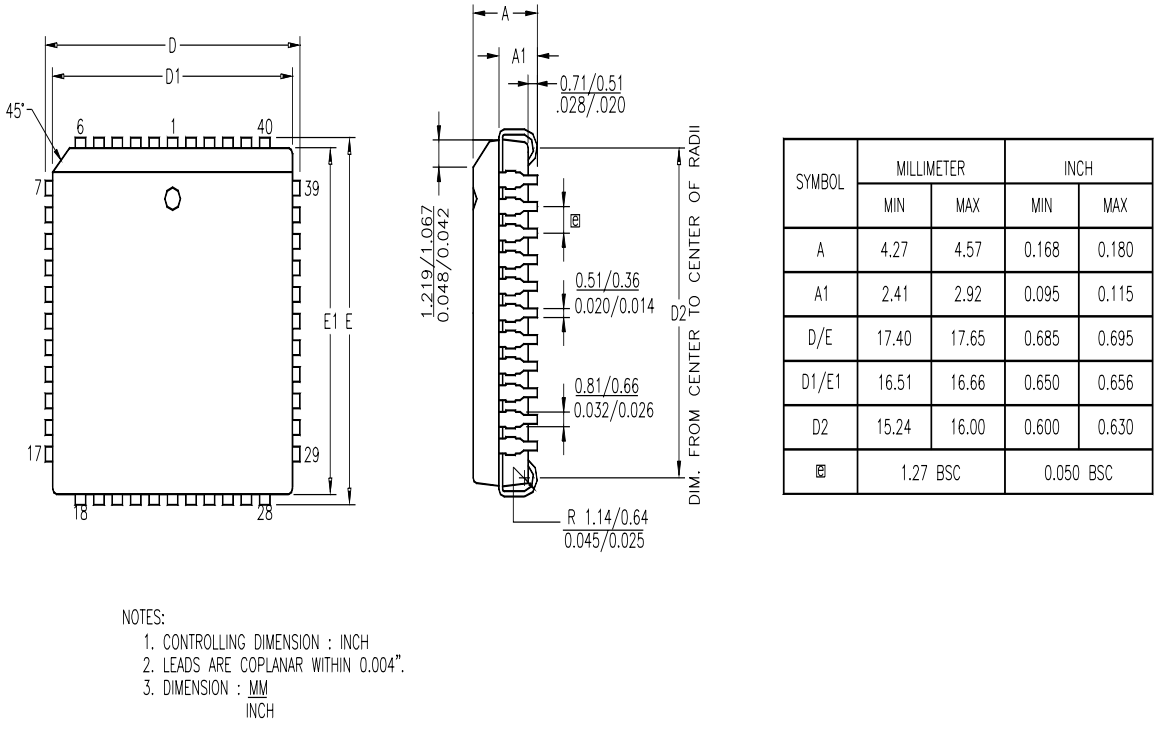
Example:

Table 28.AC Characteristics (20 MHz) EPROM Read Cycle

Number	Symbol	Parameter	Min	Max	Notes[1,2]
1	TdA(DR)	Address Valid to Read Data Required Valid	170		Note[3]

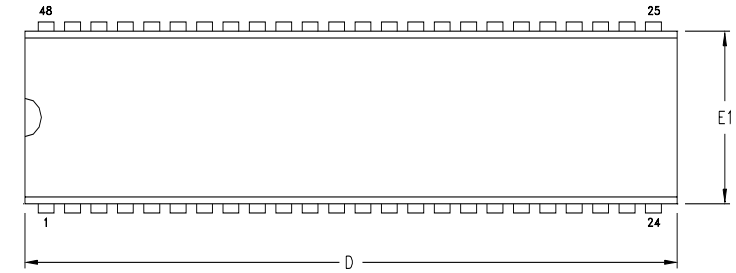
1. All times are in ns and are for 12 MHz input frequency.
2. Timings are preliminary and subject to change.
3.) Wait states add 167 ns to these times.

Packaging Information



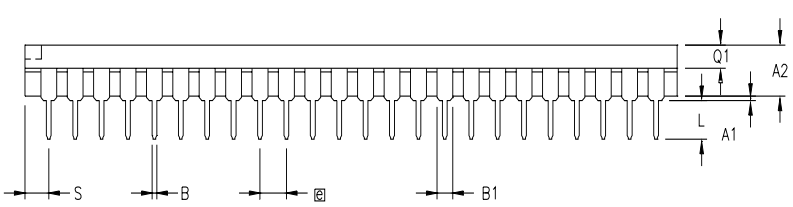
SYMBOL	MILLIMETER		INCH	
	MIN	MAX	MIN	MAX
A	4.27	4.57	0.168	0.180
A1	2.41	2.92	0.095	0.115
D/E	17.40	17.65	0.685	0.695
D1/E1	16.51	16.66	0.650	0.656
D2	15.24	16.00	0.600	0.630
e	1.27 BSC		0.050 BSC	

Figure 28.44-Pin PLCC



SYMBOL	MILLIMETER		INCH	
	MIN	MAX	MIN	MAX
A1	0.38	0.81	.015	.032
A2	3.68	4.19	.145	.165
B	0.38	0.53	.015	.021
B1	1.02	1.52	.040	.060
C	0.23	0.38	.009	.015
D	61.98	62.74	2.440	2.470
E	15.24	15.75	.600	.620
E1	13.72	14.22	.540	.560
e	2.54 BSC		.100 BSC	
eA	15.49	16.76	.610	.660
L	3.18	3.81	.125	.150
Q1	1.52	1.91	.060	.075
S	1.52	2.29	.060	.090

CONTROLLING DIMENSIONS : INCH



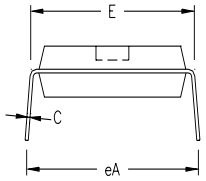


Figure 29.48-Pin DIP