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	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z88c0020vsg

Table of Contents

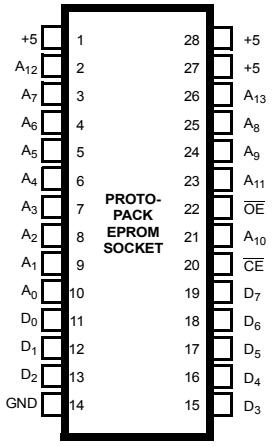


Figure 5.Pin Assignments 28-Pin Piggyback Socket

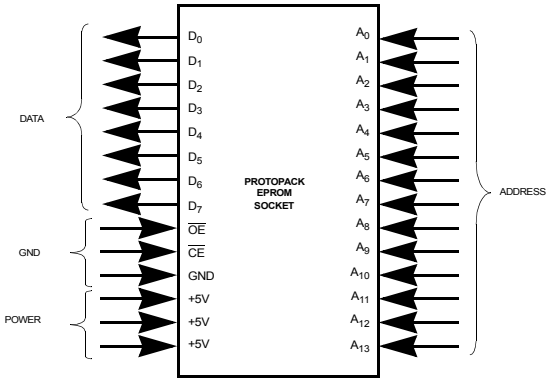


Figure 6.Pin Functions 28-Pin Piggyback Socket

Protopack

This part functions as an emulator for the basic microcomputer. It uses the same package and pin-out as the basic microcomputer but also has a 28-pin "piggy back" socket on the top into which a ROM or EPROM can be installed. The socket is designed to accept a type 2764 EPROM.

This package permits the protopack to be used in prototype and final PC boards while still permitting user program development. When a final program is devel-

The 16-bit counters can operate independently or be cascaded to perform 32-bit counting and timing operations. The DMA controller handles transfers to and from the register file or memory. DMA can use the UART or one of two ports with handshake capability.

The architecture appears in the block diagram (Figure 7).

PIN DESCRIPTIONS

The Super8 connects to external devices via the following TTL-compatible pins:

$\overline{\text{AS}}$. *Address Strobe* (output, active Low). $\overline{\text{AS}}$ is pulsed Low once at the beginning of each machine cycle. The rising edge indicates that addresses R/$\overline{\text{W}}$ and $\overline{\text{DM}}$, when used, are valid.

$\overline{\text{DS}}$. *Data Strobe* (output, active Low). $\overline{\text{DS}}$ provides timing for data movement between the address/data bus and external memory. During write cycles, data output is valid at the leading edge of $\overline{\text{DS}}$. During read cycles, data input must be valid prior to the trailing edge of $\overline{\text{DS}}$.

$P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$. *Port I/O Lines* (input/output). These 40 lines are divided into five 8-bit I/O ports that can be configured under program control for I/O or external memory interface.

In the ROMless devices, Port 1 is dedicated as a multiplexed address/data port, and Port 0 pins can be assigned as additional address lines; Port 0 non-address pins may be assigned as I/O. In the ROM and protopack, Port 1 can be assigned as input or output, and Port 0 can be assigned as input or output on a bit by bit basis.

Ports 2 and 3 can be assigned on a bit-for-bit basis as general I/O or interrupt lines. They can also be used as special-purpose I/O lines to support the UART, counter/timers, or handshake channels.

Port 4 is used for general I/O.

During reset, all port pins are configured as inputs (high impedance) except for Port 1 and Port 0 in the ROMless devices. In these, Port 1 is configured as a multiplexed address/data bus, and Port 0 pins PO_0-PO_4 are configured as address out, while pins $P0_5$-PO_7 are configured as inputs.

$\overline{\text{RESET}}$. *Reset* (input, active Low). Reset initializes and starts the Super8. When it is activated, it halts all processing; when it is deactivated, the Super8 begins processing at address 0020H.

ROMless. (input, active High). This input controls the operation mode of a 68-pin Super8. When connected to VCC, the part functions as a ROMless Z8800. When connected to GND, the part functions as a Z8820 ROM part.

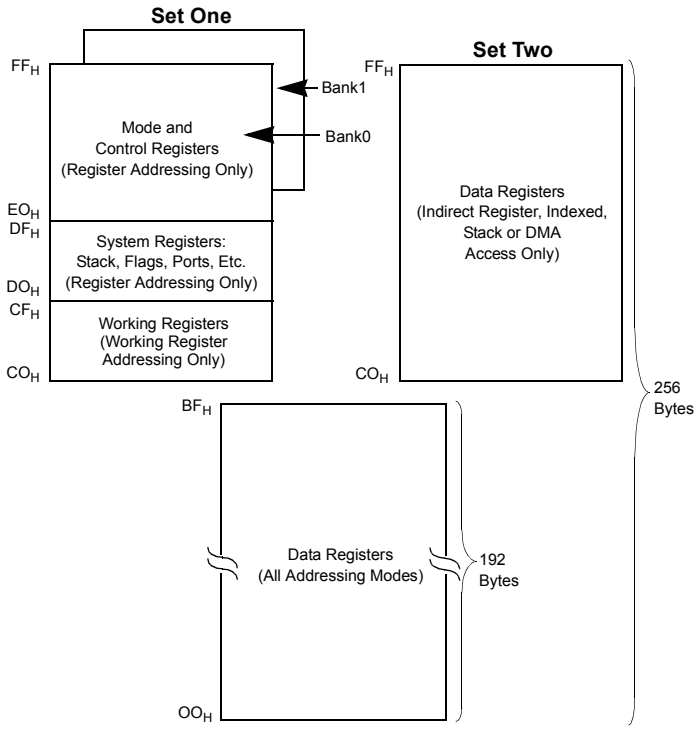


Figure 8.Super8 Registers

Working Register Window

Control registers R214 and R215 are the register pointers, RPO and RP1. They each define a moveable, 8-register section of the register space. The registers within these spaces are called working registers.

Working registers can be accessed using short 4-bit addresses. The process, shown in section a of Figure 9, works as follows:

- The high-order bit of the 4-bit address selects one of the two register pointers (0 selects RPO; 1 selects RP1).
- The five high-order bits in the register pointer select an 8-register (contiguous) slice of the register space.
- The three low-order bits of the 4-bit address select one of the eight registers in the slice.

The net effect is to concatenate the five bits from the register pointer to the three bits from the address to form an 8-bit address. As long as the address in the regis-

Table 15.Super-8 Registers (Continued)

Address				
Decimal	**Hexadecimal**		**Mnemonic**	**Function**
		Bank 1	WUMCH	Wakeup Match Register
255	FF	Bank 0	IPR	Interrupt Priority Register
		Bank 1	WUMSK	Wakeup Match Register

MODE AND CONTROL REGISTERS

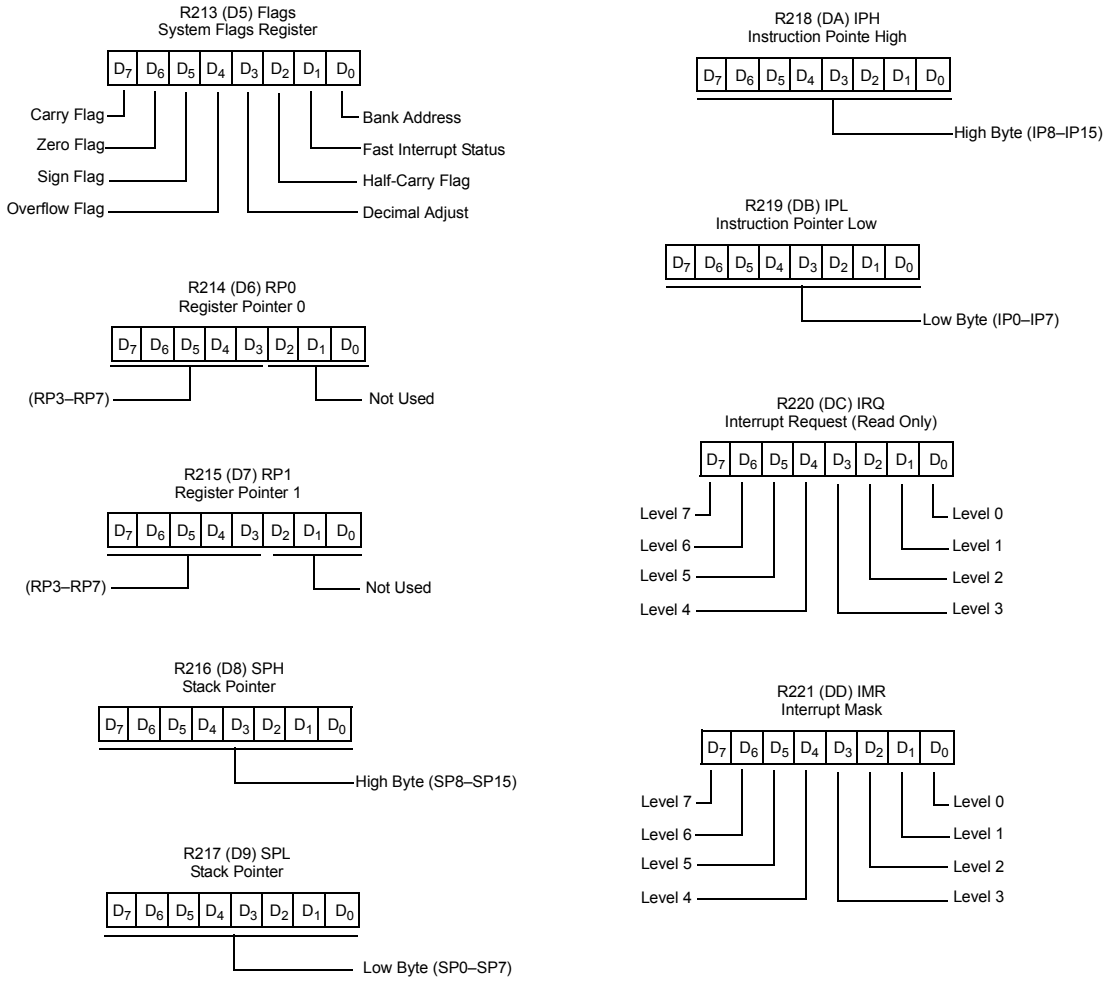


Figure 10.Mode and Control Registers

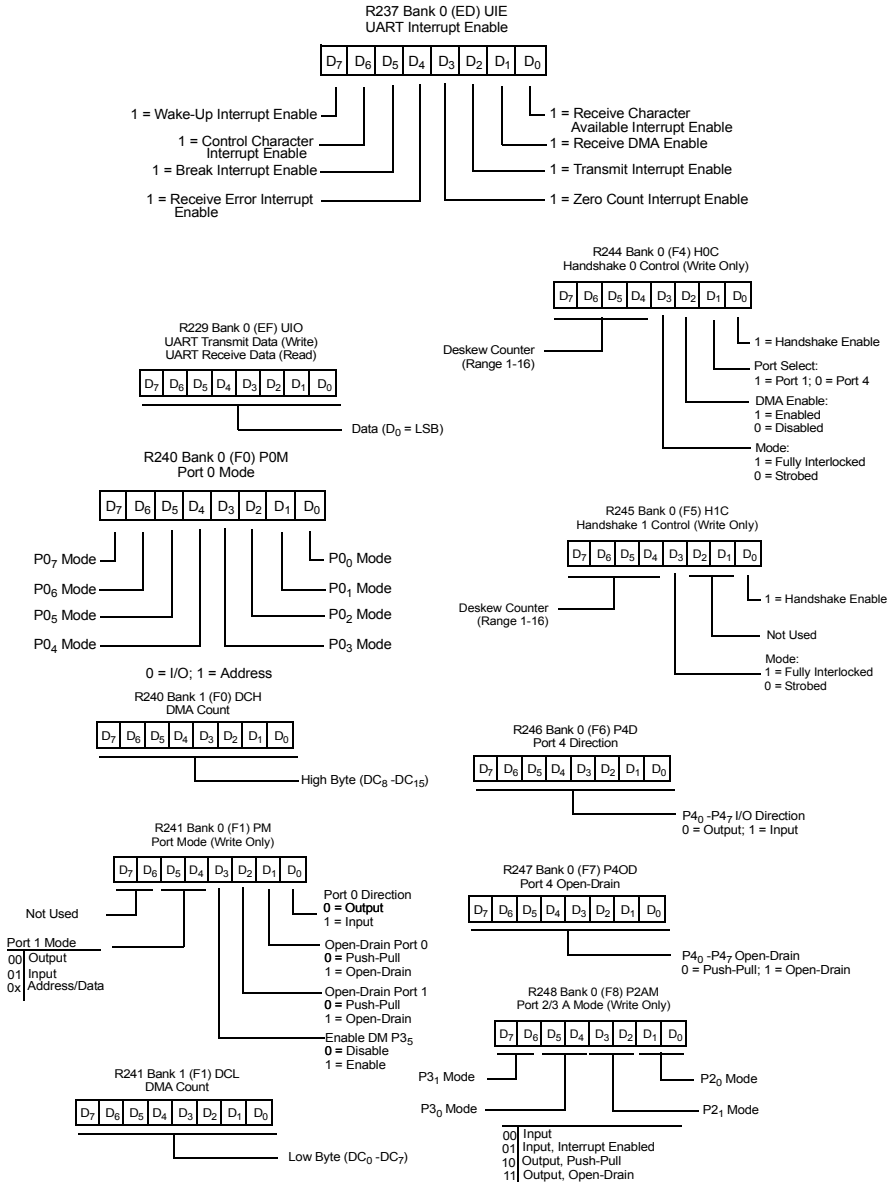


Figure 13. Mode and Control Registers (Continued)

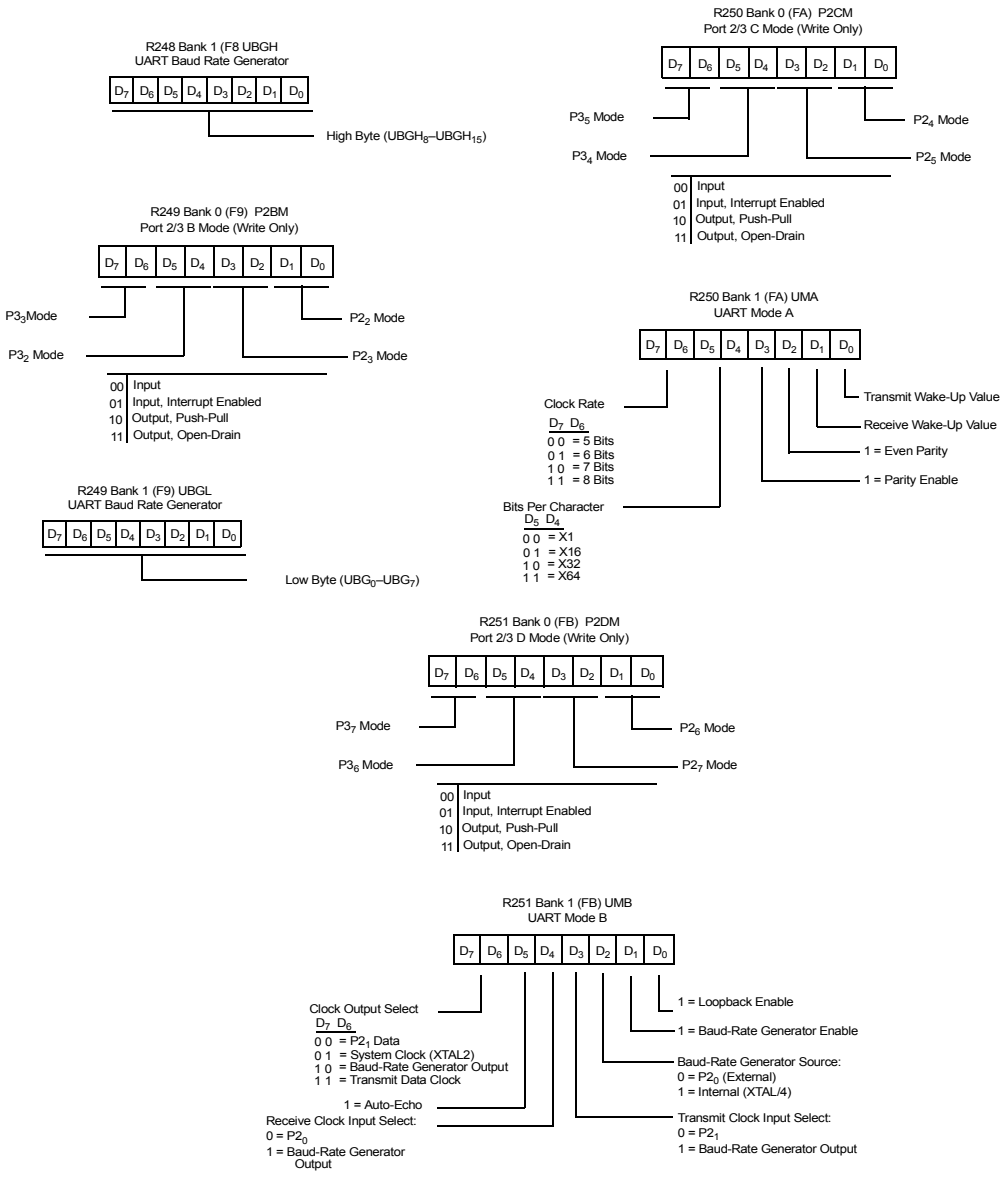


Figure 14.Mode and Control Registers (Continued)

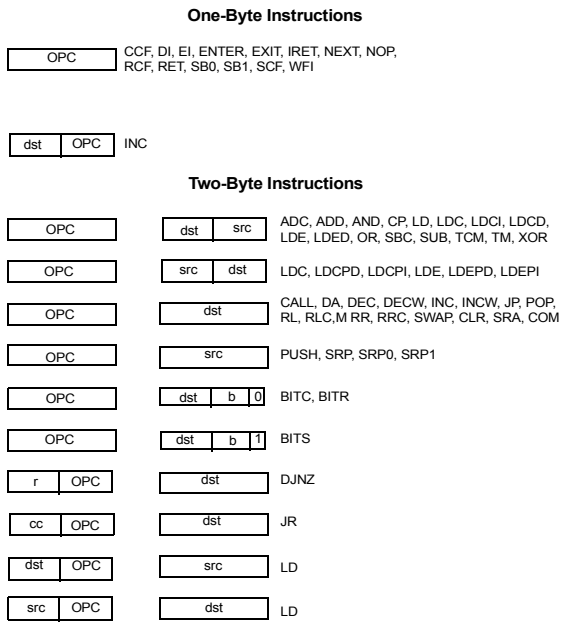


Figure 17.Instruction Formats

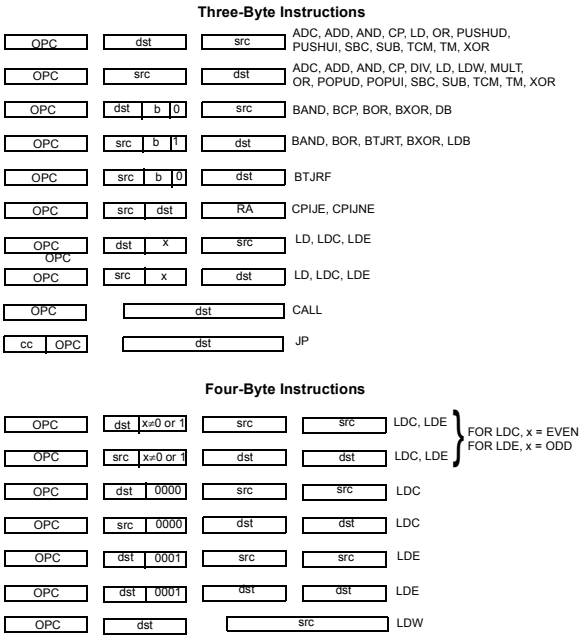


Figure 18.Instruction Formats (Continued)

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
dst←src	RR	IR	85						
	RR	IM	86						
NEXT			0F	-	-	-	-	-	-
PC←@IP									
IP←IP + 2									
NOP			FF	-	-	-	-	-	-
OR dst, src	Note[1]		4[ ]	-	*	*	0	-	-
dst←dst OR src									
POP dst		R	50	-	-	-	-	-	-
dst←@SP;		IR	51						
SP←SP + 1									
POPUD dst, src	R	IR	92	-	-	-	-	-	-
dst←src									
IR←IR - 1									
POPUI dst, src	R	IR	93	-	-	-	-	-	-
dst←src									
IR←IR + 1									
PUSH src		R	70	-	-	-	-	-	-
SP←SP - 1; @SP←src		IR	71						
PUSHUD dst, src	IR	R	82	-	-	-	-	-	-
IR←IR - 1									
dst←src									
PUSHUI dst, src	IR	R	83	-	-	-	-	-	-
IR←IR + 1									
dst←src									
RCF			CF	0	-	-	-	-	-
C←0									
RET			AF	-	-	-	-	-	-

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
C←1									
SRA dst dst(7)←dst(7) C←dst(0) dst(N)←dst(N + 1) N = 0 to 6	R IR		D0 D1	*	*	*	0	-	-
SRP src RP0←IM RP0←IM + 8		IM	31	-	-	-	-	-	-
SRP0 RP0←IM		IM	31	-	-	-	-	-	-
SRP1 RP1←IM		IM	31	-	-	-	-	-	-
SUB dst, src dst←dst – src	Note[1]		2[ ]	*	*	*	*	1	*
SWAP dst dst(0-3)↔dst(4-7)	R IR		F0 F1	-	*	*	U	-	-
TCM dst, src (NOT dst) AND src	Note[1]		6[ ]	-	*	*	0	-	-
TM dst, src dst AND src	Note[1]		7[ ]	-	*	*	0	-	-
WFI			3F	-	-	-	-	-	-
XOR dst, src dst←dst XOR src	Note[1]		B[ ]	-	*	*	0	-	-

1. These instructions have an identical set of addressing modes, which are encoded for brevity. The first opcode nibble identifies the command, and is found in the table above. The second nibble, represented by a [ ], defines the addressing mode as shown in Table 6.

Table 21.Second Nibble

Addr Mode		Lower Opcode Nibble[1]
dst	src	
r	r	[2]
r	Ir	[3]
R	R	[4]
R	IR	[5]
R	IM	[6]

1. For example, to use an opcode represented as x[ ] with an "RR" addressing mode, use the opcode "x4."
   0= Cleared to Zero
   1= Set to One
   -= Unaffected
   *= Set or reset, depending on result of operation.
   U= Undefined

SUPER-8 OPCODE MAP

Upper Nibble (Hex) \ Lower Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	6 DEC R_1	6 DEC IR_1	6 ADD r_1,r_2	6 ADD r_1,Ir_2	10 ADD R_2,R_1	10 ADD IR_2,R_1	10 ADD R_1,IM	10 BOR* r_0-R_b	6 LD r_1,R_2	6 LD r_2,R_1	12/10 DJNZ r_1,RA	12/10 JR cc,RA	6 LD r_1,RM	12/10 JP cc,DA	6 INC r_1	14 NEXT
1	6 RLC R_1	6 RLC IR_1	6 ADC r_1,r_2	6 ADC r_1,Ir_2	10 ADD R_2,R_1	10 ADC IR_2,R_1	10 ADC R_1,IM	10 BCP r_1,b,R_2								20 ENTER
2	6 INC R_1	6 INC IR_1	6 SUB r_1,r_2	6 SUB r_1,Ir_2	10 SUB R_2,R_1	10 SUB IR_2,R_1	10 SUB R_1,IM	10 BXOR* r_0-R_b								22 EXIT
3	10 JP IRR_1	NOTE C	6 SBC r_1,r_2	6 SBC r_1,Ir_2	10 SBC R_2,R_1	10 SBC IR_2,R_1	10 SBC R_1,IM	NOTE A								6 WFI
4	6 DA R_1	6 DA IR_1	6 OR r_1,r_2	6 OR r_1,Ir_2	10 OR R_2,R_1	10 OR IR_2,R_1	10 OR R_1,IM	10 LDB* r_0-Rb								6 SBO
5	10 POP R_1	10 POP IR_1	6 AND r_1,r_2	6 AND r_1,Ir_2	10 AND R_2,R_1	10 AND IR_2,R_1	10 AND R_1,IM	10 BITC r_1,b								6 SBI
6	6 COM R_1	6 COM IR_1	6 TCM r_1,r_2	6 TCM r_1,Ir_2	10 TCM R_2,R_1	10 TCM IR_2,R_1	10 TCM R_1,IM	10 BAND* r_0-Rb								
7	10/12 PUSH R_2	12/14 PUSH IR_2	6 TM r_1,r_2	6 TM r_1,Ir_2	10 TM R_2,R_1	10 TM IR_2,R_1	10 TM R_1,IM	NOTE B								
8	10 DECW IRR_1	10 DECW IR_1	10 PUSHUD IR_1,R_2	10 PUSHUI IR_2,R_2	24 MULT R_2,RR_1	24 MULT IR_2,RR_1	24 MULT IM,RR_1	10 LD r_1,x,r_2								6 DI
9	6 RL R_1	6 RL IR_1	6 POPUD IR_2,R_1	6 POPUI IR_2,R_1	28/12 DIV R_2,RR_1	28/12 DIV IR_2,RR_1	28/12 DIV IM,RR_1	10 LD r_2,x,r_1								6 EI
A	10 INCW RR_1	10 INCW IR_1	6 CP r_1,r_2	6 CP r_1,Ir_2	10 CP R_2,R_1	10 CP IR_2,R_1	10 CP R_1,IM	NOTE D								14 RET
B	6 CLR R_1	6 CLR IR_1	6 XOR r_1,r_2	6 XOR r_1,Ir_2	10 XOR R_2,R_1	10 XOR IR_2,R_1	10 XOR R_1,IM	NOTE E								16/6 IRET
C	6 RRC IRR_1	6 RRC IR_1	16/18 CPIJE Ir,r_2,RA	12 LDC* r_1,Irr_2	10 LDW RR_2,RR_1	10 LDW IR_2,RR_1	12 LDW RR_1,IML	6 LD r_1,Ir_2								6 RCF
D	6 SRA R_1	6 SRA IR_1	16/18 CPIJNE Ir_1,r_2,RA	12 LDC* r_2,Irr_1	20 CALL IA_1		10 LD IR_1,IM	6 LD $Ir1,r_2$								6 SCF
E	6 RR R_1	6 RR IR_1	16 LDCD* r_1,Irr_2	16 LDCI* r_1,Irr_2	10 LD R_2,R_1	10 LD IR_2,R_1	10 LD R_1,IM	18 LDC* r_1,Irr_2,xs								6 CCF
F	8 SWAP R_1	8 SWAP IR_1	16 LDCPD* r_2,Irr_1	16 LDCPI* r_2,Irr_1	18 CALL IRR_1	18 CALL R_2,IR_1	18 CALL DA_1	18 LDC* r_1,Irr_1,xs								6 NOP

Note A

16/18 BTJRF r_2,b,RA	16/18 BTJRT r_2,b,RA

Note B

8 BITR r_1,b	8 BITS r_1,b

Note C

6 SRP IM	6 SRP0 IM	6 SRP1 IM

Note D

20 LDC* r_1,Irr_2,xL	20 LDC* r_1,DA_2

Note E

20 LDC* r_2,Irr_2,xL	20 LDC* r_2,DA_1

Legend:
r = 4-bit address
R = 8-bit address
b = bit number
R_1 or r_1 = dsts address
R_2 or r_2 = src address

***Examples:**
BORr_0-R_2
is BORr_1,b,R_2
or BORr_2,b,R_1
LDCr_1,Irr_2
is LDCr_1,Irr_2 = program
or LDEr_1,Irr_2 = data

Sequence:
Opcode, first, second, third operands

NOTE: The blank areas are not defined.

Figure 19.Opcode Map

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

Table 22.Super8 Instructions (Continued)

Mnemonic	Operands	Instruction
SRP1	src	Set register pointer one

INTERRUPTS

The Super8 interrupt structure contains 8 levels of interrupt, 16 vectors, and 27 sources.

Interrupt priority is assigned by level, controlled by the Interrupt Priority register (IPR). Each level is masked (or enabled) according to the bits in the Interrupt Mask register (IMR), and the entire interrupt structure can be disabled by clearing a bit in the System Mode register (R222).

The three major components of the interrupt structure are sources, vectors, and levels. These are shown in Figure 20 and discussed in the following paragraphs.

Sources

A source is anything that generates an interrupt. This can be internal or external to the Super8 MCU. Internal sources are hardwired to a particular vector and level, while external sources can be assigned to various external events.

Vectors

The 16 vectors are divided unequally among the eight levels. For example, vector 12 belongs to level 2, while level 3 contains vectors 0, 2, 4, and 6.

The vector number is used to generate the address of a particular interrupt servicing routine; therefore all interrupts using the same vector must use the same interrupt handling routine.

Levels

Levels provide the top level of priority assignment. While the sources and vectors are hardwired within each level, the priorities of the levels can be changed by using the Interrupt Priority register (see Figure 15 for bit details).

If more than one interrupt source is active, the source from the highest priority level is serviced first. If both sources are from the same level, the source with the lowest vector has priority. For example, if the UART Receive Data bit and UART Parity Error bit are both active, the UART Parity Error bit is serviced first because it is vector 16, and UART receive data is vector 20.

addresses to high addresses in the register file, use PUSHUI and POPUD. For a stack that grows from high addresses to low addresses in data memory, use LDEI for pop and LDEPD for push.

COUNTER/TIMERS

The Super8 has two identical independently programmable 16-bit counter/timers that can be cascaded to produce a single 32-bit counter. They can be used to count external events, or they can obtain their input internally. The internal input is obtained by dividing the crystal frequency by four.

The counter/timers can be set to count up or down, by software or external events. They can be set for single or continuous cycle counting, and they can be set with a bi-value option, where two preset time constants alternate in loading the counter each time it reaches zero. This can be used to produce an output pulse train with a variable duty cycle.

The counter/timers can also be programmed to capture the count value at an external event or generate an interrupt whenever the count reaches zero. They can be turned on and off in response to external events by using a gate and/or a trigger option. The gate option enables counts only when the gate line is Low; the trigger option turns on the counter after a transient High. The gate and trigger options used together cause the counter/timer to work in gate mode after initially being triggered.

The control and status register bits for the counter/timers are shown in Figure 7.

DMA

The Super8 features an on-chip Direct Memory Access (DMA) channel to provide high bandwidth data transmission capabilities. The DMA channel can be used by the UART receiver, UART transmitter, or handshake channel 0. Data can be transferred between the peripheral and contiguous locations in either the register file or external data memory. A 16-bit count register determines the number of transactions to be performed; an interrupt can be generated when the count is exhausted. DMA transfers to or from the register file require six CPU clock cycles; DMA transfers to or from external memory take ten CPU clock cycles, excluding wait states.

ABSOLUTE MAXIMUM RATINGS

Voltage on all pins with respect to ground	-0.3 V to+7.0 V
Ambient Operating Temperature	See Ordering Information
Storage Temperature	-65 °C to + 150 °C

Stresses greater than these may cause permanent damage to the device. This is a stress rating only; operation of the device under conditions more severe than those listed for operating conditions may cause permanent damage to the device. Exposure to absolute maximum ratings for extended periods may also cause permanent damage.

STANDARD TEST CONDITIONS

Figure 21 shows the setup for standard test conditions. All voltages are referenced to ground, and positive current flows into the reference pin.

Standard conditions are:

- + 4.75 V ≤ VCC ≤ + 5.25V
- GND = 0 V
- 0°C ≤ TA ≤ + 70°C

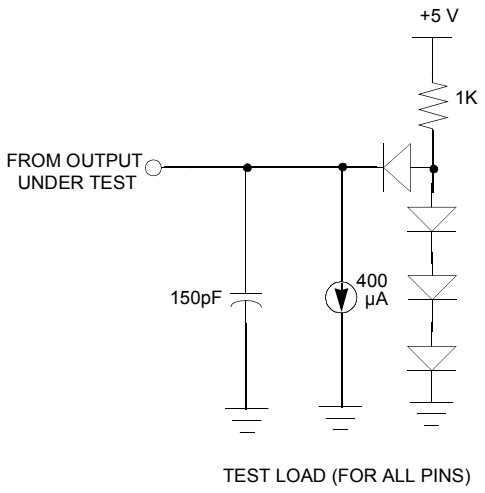


Figure 21.Standard Test Load

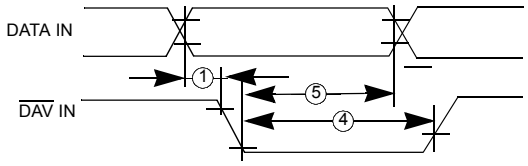


Figure 23.Strobed Mode

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.

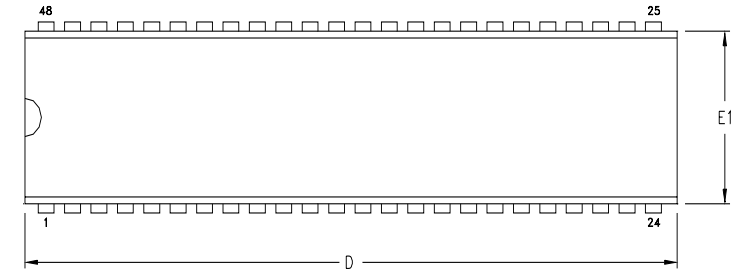
AC CHARACTERISTICS (12 MHz)

Read /Write

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

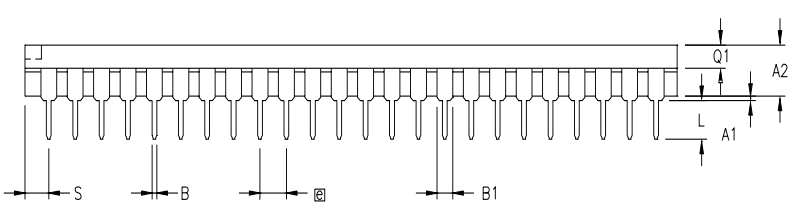
			Normal Timing		Extended Timing		
Number	**Symbol**	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Notes[1,2]**
1	TdA(AS)	Address Valid to $\overline{AS}$ ↑ Delay	35		115		
2	TdAS(A)	$\overline{AS}$ ↑ to Address Float Delay	65		150		
3	TdAS(DR)	$\overline{AS}$ ↑ to Read Data Required Valid		270		600	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..



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



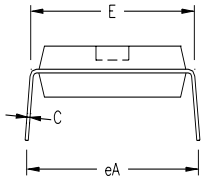


Figure 29.48-Pin DIP