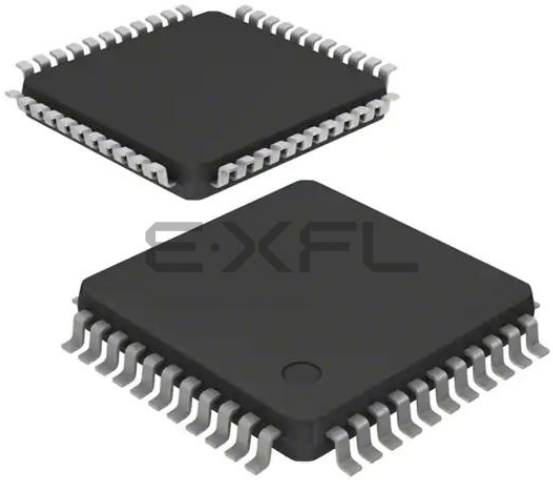


Welcome to E-XFL.COM

Understanding Embedded - DSP (Digital Signal Processors)

Embedded - DSP (Digital Signal Processors) are specialized microprocessors designed to perform complex mathematical computations on digital signals in real-time. Unlike general-purpose processors, DSPs are optimized for high-speed numeric processing tasks, making them ideal for applications that require efficient and precise manipulation of digital data. These processors are fundamental in converting and processing signals in various forms, including audio, video, and communication signals, ensuring that data is accurately interpreted and utilized in embedded systems.

Applications of Embedded - DSP (Digital Signal Processors)

Details	
Product Status	Active
Type	Fixed Point
Interface	External Peripheral Interface
Clock Rate	20MHz
Non-Volatile Memory	OTP (8kB)
On-Chip RAM	1kB
Voltage - I/O	5.00V
Voltage - Core	5.00V
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8937120asg

GENERAL DESCRIPTION (Continued)

Note: All signals with an overline are active Low. For example in RD/$\overline{WR}$, RD is active High and $\overline{WR}$ is active Low.

The power connections follow the convention described below:

Connection	Circuit	Device
Power	V_{CC}	V_{DD}
Ground	GND	V_{SS}

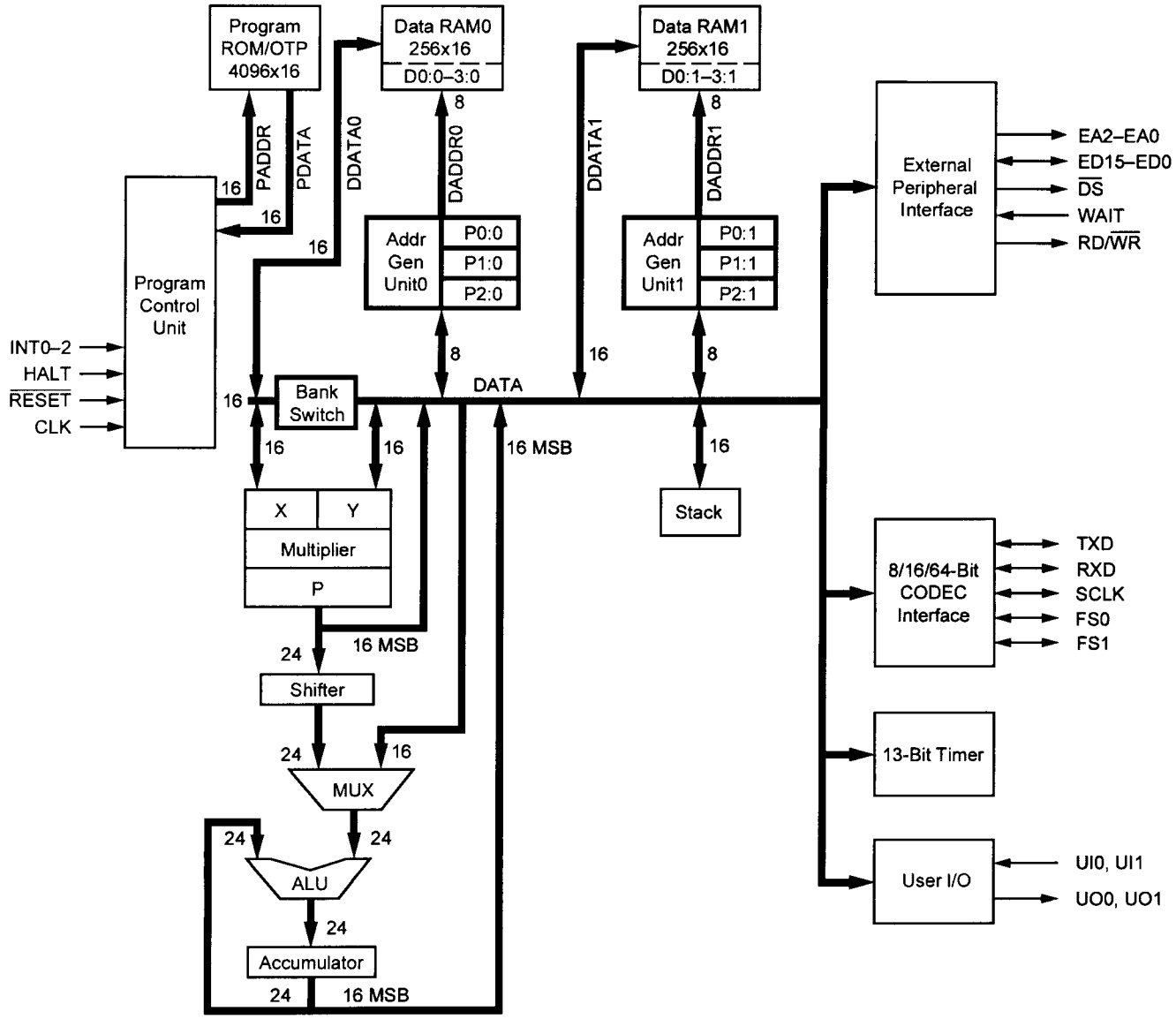


Figure 1. Z893x1 Functional Block Diagram

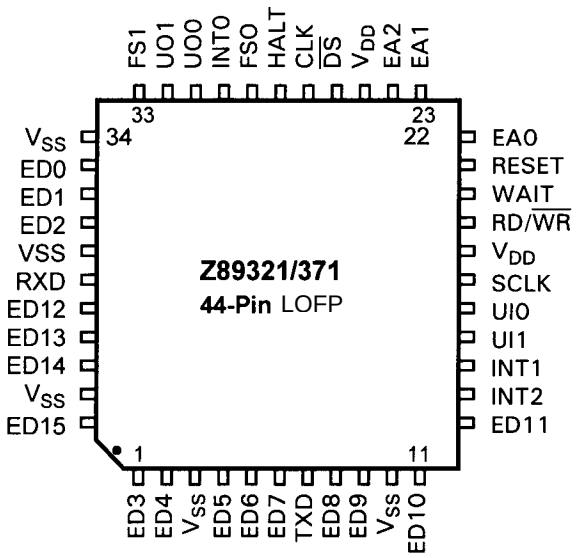


Figure 5. Z89321/371 44-Pin LOFP Pin Assignments

Table 3. Z89321/371 44-Pin LOFP Pin Identification

No.	Symbol	Function	Direction
1	ED3	External Data Bus	In/Out
2	ED4	External Data Bus	In/Out
3	V_{SS}	Ground	
4	ED5	External Data Bus	In/Out
5	ED6	External Data Bus	In/Out
6	ED7	External Data Bus	In/Out
7	TXD	Serial Output Data	Output
8	ED8	External Data Bus	In/Out
9	ED9	External Data Bus	In/Out
10	V_{SS}	Ground	
11	ED10	External Data Bus	In/Out
12	ED11	External Data Bus	In/Out
13	INT2	Interrupt	Input
14	INT1	Interrupt	Input
15	UI1	User Input	Input
16	UI0	User Input	Input
17	SCLK	CODEC Serial Clock	Output
18	V_{DD}	Power Supply	Input
19	RD/$\overline{WR}$	Read/$\overline{Write}$ select for ED bus	Output
20	WAIT	Wait state	Input
21	RESET	Reset	Input
22	EA0	External Address bus	Output

Table 3. Z89321/371 44-Pin LOFP Pin Identification

No.	Symbol	Function	Direction
23	EA1	External Address bus	Output
24	EA2	External Address bus	Output
25	V_{DD}	Power Supply	Input
26	$\overline{DS}$	Data Strobe for ED Bus	Output
27	CLK	Clock	Input
28	HALT	Stop execution	Input
29	FS0	Frame Sync–CODEC Ch. 0	Output
30	INT0	Interrupt	Input
31	UO0	User Output	Output
32	UO1	User Output	Output
33	FS1	Frame Sync–CODEC Ch. 1	Output
34	V_{SS}	Ground	
35	ED0	External Data Bus	In/Out
36	ED1	External Data Bus	In/Out
37	ED2	External Data Bus	In/Out
38	V_{SS}	Ground	
39	RXD	Serial Input Data	Input
40	ED12	External Data Bus	In/Out
41	ED13	External Data Bus	In/Out
42	ED14	External Data Bus	In/Out
43	V_{SS}	Ground	
44	ED15	External Data Bus	In/Out

AC ELECTRICAL CHARACTERISTICS

Table 5. V_{DD} = 5V ±10%, T_A = 0°C to +70°C for "S" Temperature Range (T_A = –40°C to +85°C for "E" temperature range, unless otherwise noted)

Symbol	Parameter	Min [ns]	Max [ns]
Clock			
TCY	CLK Cycle Time	50	31250
CPWH	CLK Pulse Width High	21	
CPWL	CLK Pulse Width Low	21	
Tr	CLK Rise Time		2
Tf	CLK Fall Time		2
External Peripheral Bus			
DSVALID	$\overline{DS}$ Valid Time from CLK Fall	0	15
DSHOLD	$\overline{DS}$ Hold Time from CLK Rise	0	15
EASET	EA Setup Time to $\overline{DS}$ Fall	10	
EAHOLD	EA Hold Time from $\overline{DS}$ Rise	4	
RWSET	Read/Write Setup Time to $\overline{DS}$ Fall	10	
RWHOLD	Read/Write Hold Time from $\overline{DS}$ Rise	0	
RDSET	Data Read Setup Time to $\overline{DS}$ Rise	15	
RDHOLD	Data Read Hold Time from $\overline{DS}$ Rise	0	
WRVALID	Data Write Valid Time from $\overline{DS}$ Fall		5
WRHOLD	Data Write Hold Time from $\overline{DS}$ Rise	2	
Reset			
RSET	Reset Setup Time to CLK Fall for synchronous operation	15	
RWIDTH	Reset Low Pulse Width	2 TCY	
RRISE	Reset Rise Time		50
Interrupt			
INTSET	Interrupt Setup Time to CLK Fall	7	
INTWIDTH	Interrupt Low Pulse Width	1 TCY	
Halt			
HSET	Halt Setup Time to CLK Rise	4	
HHOLD	Halt Hold Time from CLK Rise	12	
Wait State			
WSET	Wait Setup Time to CLK Rise	20	
WHOLD	Wait Hold Time from CLK Rise	10	
CODEC Interface			
SSET	SCLK Setup Time from CLK Rise		15
FSSET	FSYNC Setup Time from SCLK Rise		7
TXSET	TXD Setup Time from SCLK Rise		7
RXSET	RXD Setup Time to SCLK Fall	7	
RXHOLD	RXD Hold Time from SCLK Fall	0	

TIMING DIAGRAMS

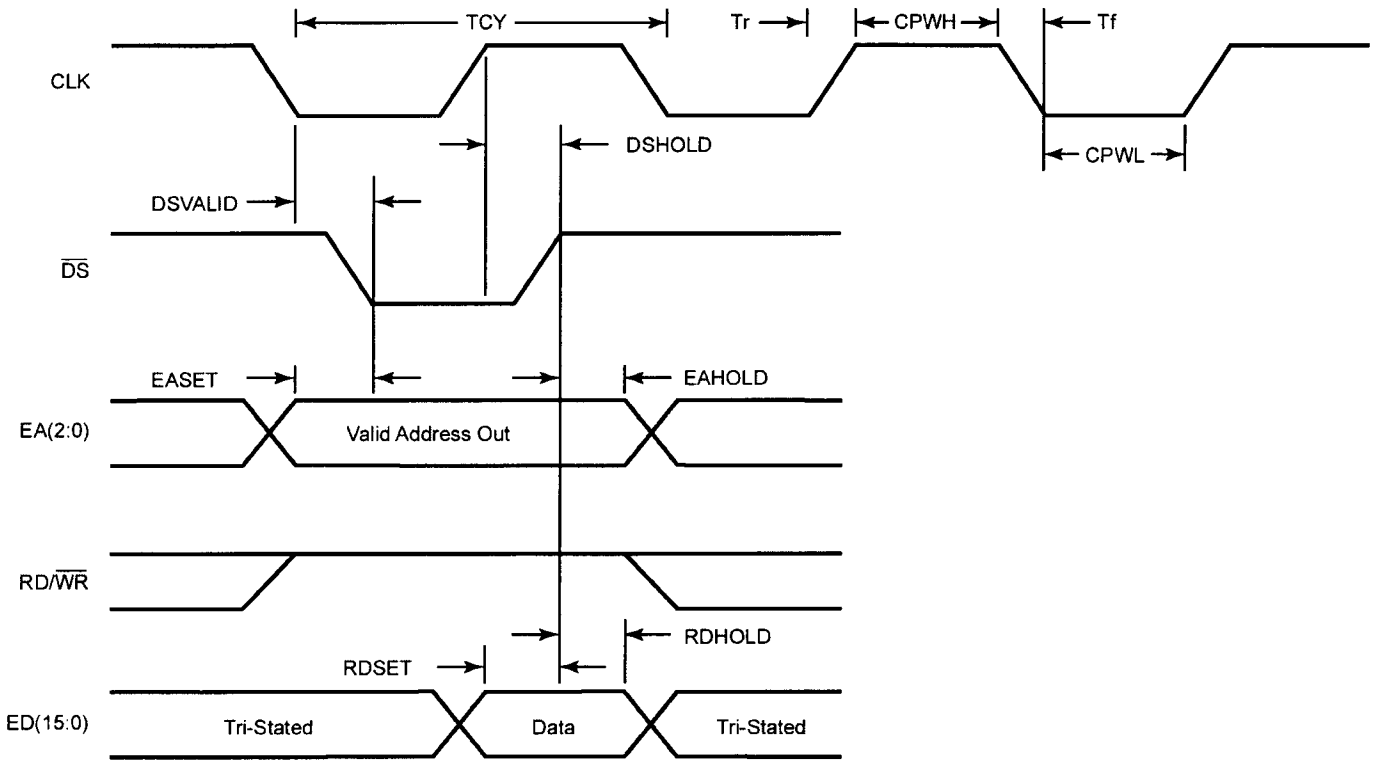


Figure 7. Read Timing

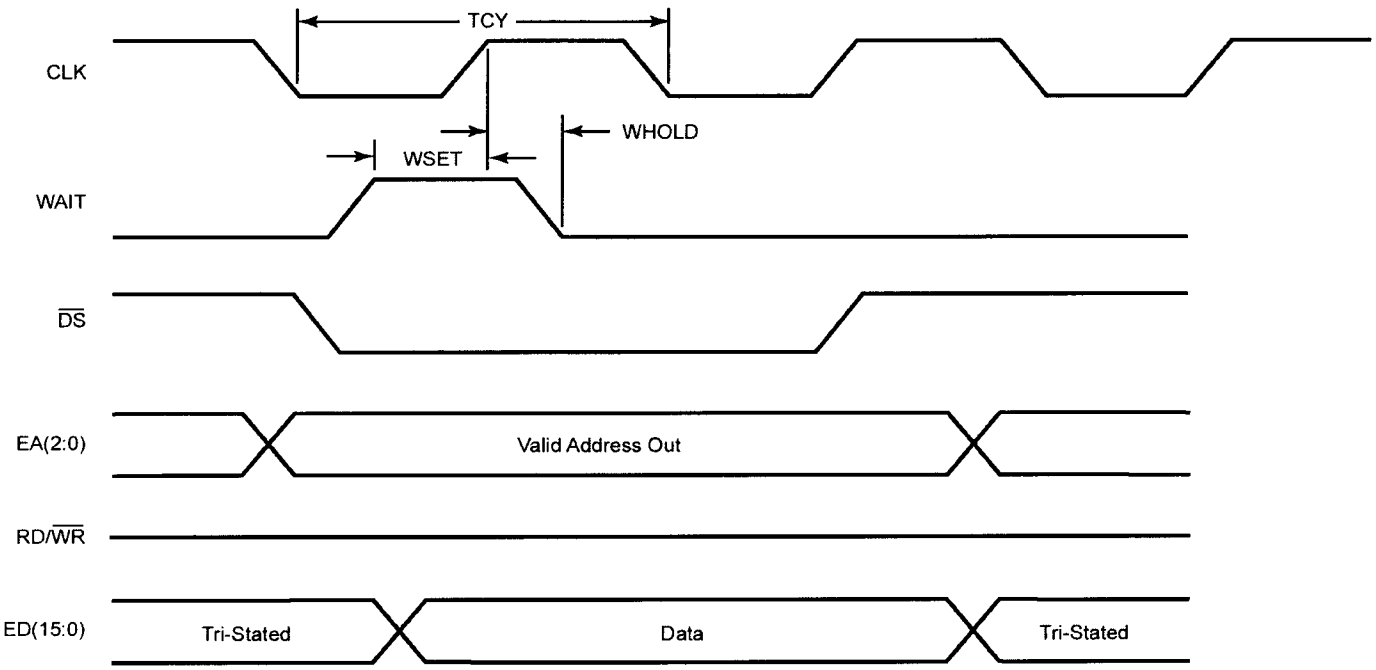


Figure 8. External Data (ED) Bus Read Timing Using WAIT Pin

TIMING DIAGRAMS (Continued)

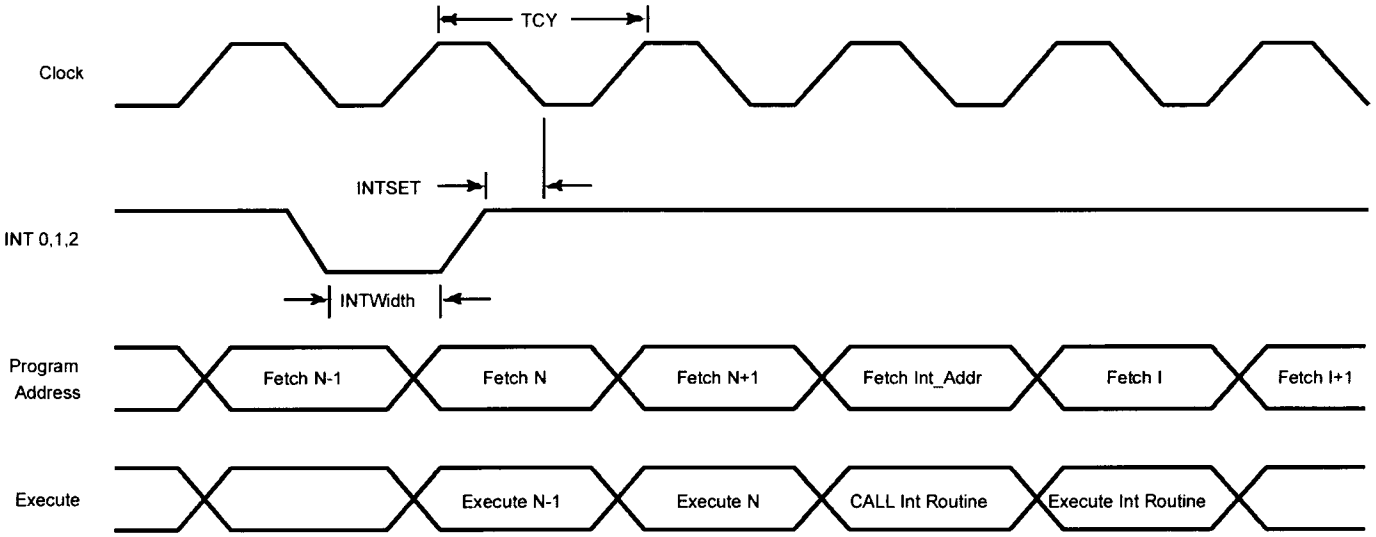


Figure 11. Interrupt Timing

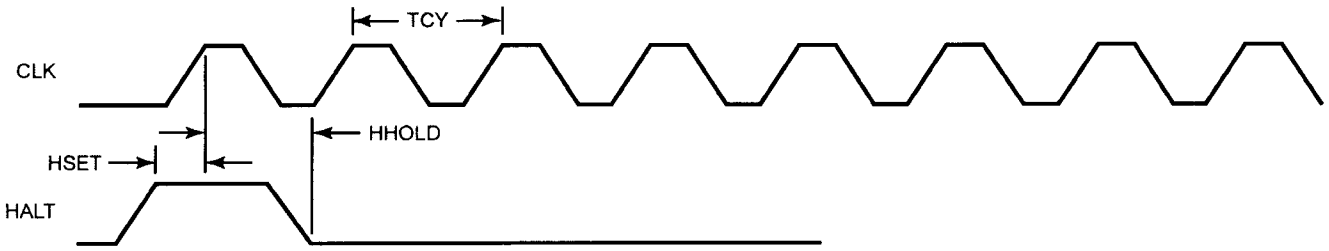


Figure 12. HALT Timing

Interrupts. The Z893x1 has three positive edge-triggered interrupt inputs. An interrupt is serviced at the end of an instruction execution. Two machine cycles are required to enter an interrupt instruction sequence. The PC is pushed onto the stack. At the end of the interrupt service routine, a RET instruction is used to pop the stack into the PC. The priority of the interrupts is INT0 = highest, INT2 = lowest. When those peripherals are enabled, INT1 is dedicated to the CODEC Interface and INT2 is dedicated to the 13-bit timer.

The Set-Interrupt-Enable-Flag (SIEF) instruction enables the interrupts. Interrupts are automatically disabled when entering an interrupt service routine. Before exiting an interrupt service routine, the SIEF instruction can be used to re-enable interrupts.

Registers. The Z893x1 has 19 internal registers and up to seven user-defined 16-bit external registers (EXT0–EXT6). The external register address space for EXT4–EXT6 is used by the Z893x1 internal peripherals. Disabling a peripheral allows access to these addresses for general-purpose use.

External Register Usage. The external registers EXT0–EXT6 are accessed using the External Address Bus EA2–EA0, the External Data Bus (ED Bus) ED15–ED0, and control signals $\overline{DS}$, WAIT, and RD/$\overline{WR}$. These registers provide a convenient data transfer capability with external peripherals. Data transfers can be performed in a single-cycle. An internal Wait-State generator is provided to accommodate slower external peripherals. A single wait state can be implemented through control register EXT7–2. For additional wait states, the WAIT pin can be used. The WAIT pin is monitored only during execution of a read or write instruction to external peripherals on the ED bus.

Wait-State Generator. An internal Wait-State generator is provided to accommodate slow external peripherals. A single Wait-State can be implemented through a control register. For additional states, a dedicated pin (WAIT) can be held Low. The WAIT pin is monitored only during execution of a read or write instruction to external peripherals (ED bus).

CODEC Interface. The CODEC Interface provides the necessary control signals for transmission of CODEC information to/from the processor. The CODEC Interface accommodates external 8-bit PCM or 16/64-bit linear CODECs. The CODEC Interface can also be used with external A/D and D/A converters. The interface can also be used as a high-speed serial port.

µ-Law Compression. The CODEC Interface provides optional hardware µ-Law compression from 13-bit format to 8-bit format. Decompression is performed in software using a 128-word lookup table.

Timers. Two programmable timers, a general purpose 13-bit Timer, and a dedicated 12-bit Counter/Timer are provided to support the CODEC Interface. The 13-bit Timer can be operated in either continuous or one-shot mode. If the CODEC Interface is not enabled, its 12-bit Counter/Timer is also available for general-purpose use.

MEMORY MAP

Program Memory. Programs of up to 4K words can be masked into internal ROM (Z89321) or programmed into an OTP (Z89371). Four locations are dedicated to the vector addresses for the three interrupts (0FFDH–0FFFH) and the starting address following a $\overline{\text{RESET}}$ (0FFCH). Internal ROM is mapped from 0000H to 0FFFH, and the highest location for program instructions is 0FFBH.

Internal Data RAM. All Z893x1 family members have internal 512 x 16-bit data RAM organized as two banks of 256 x 16-bit words each (RAM0 and RAM1). The three addressing modes available to access the data RAM are direct addressing, short form direct, and register indirect.

The contents of both data RAM banks can be read simultaneously and loaded into the X and Y inputs of the multiplier during a multiply instruction.

The addresses for each data RAM bank are:

0–255 (0000H–00FFH) for RAM0
256–511 (0100H–01FFH) for RAM1

Data RAM Pointers. In register indirect, each data RAM bank is addressed by one of three data RAM address pointers:

Pn:b, where
n = pointer number = 0, 1, or 2
b = bank = 0 or 1,

thus,

P0:0, P1:0, P2:0 for RAM0
P0:1, P1:1, P2:1 for RAM1

In auto-increment, loop-increment, and loop-decrement indirect addressing, the pointer is automatically modified.

The data RAM pointers, which may be read or written directly, are 8-bit registers connected to the lower byte of the internal 16-bit DDATA Bus.

Program Memory Pointers. The first 16 locations of each data RAM bank can be used as pointers to locations in Program Memory. These locations can be an efficient way to address coefficients. The programmer selects a pointer location using two bits in the status register and two bits in the operand. At any one time, there are eight usable pointers, four per bank, and the four pointers are in consecutive locations.

Dn:b, where
n = pointer number = 0, 1, 2, or 3
b = bank = 0 or 1,

thus,

D0:0, D1:0, D2:0, D3:0 for RAM0
D0:1, D1:1, D2:1, D3:1 for RAM1

For example, if S3/S4 = 01 in the status register, then D0:0/D1:0/D2:0/D3:0 refer to register locations 4/5/6/7 in data RAM Bank 0.

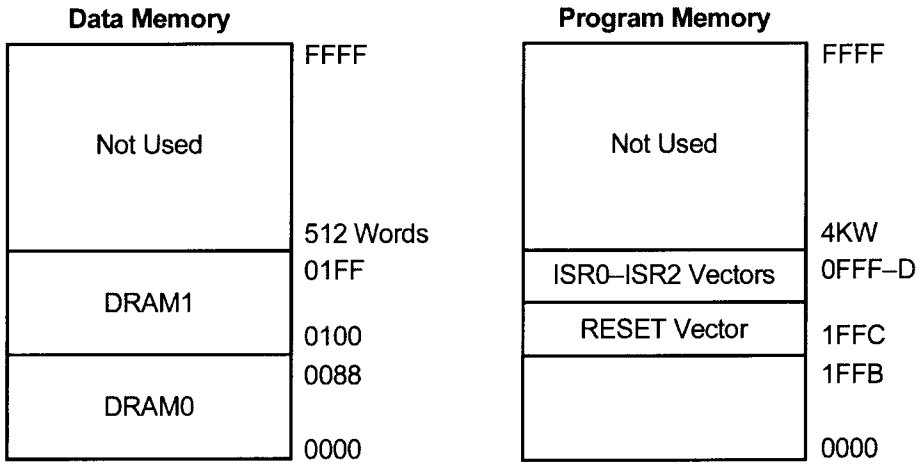


Figure 16. Memory Map

REGISTERS (Continued)

Table 8. RPL Description

S2	S1	S0	Loop Size
0	0	0	256
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

The following are not actually registers; however, they have a read/write function that acts primarily the same way as the hardware registers do on the chip:

Register	Register Definition
BUS	DDATA Bus
Dn:b	Program Memory Pointers
EXTn	External Registers

BUS is a read-only register which, when accessed, returns the contents of the D-Bus. BUS is used for emulation only.

Dn:b refers to locations in RAM that can be used as a pointer to locations in program memory. These locations make the Z89321/371 capable for coefficient addressing. The programmer decides which location to choose from based on two bits in the status register and two bits in the operand; only the lower 16 possible locations in RAM can be specified. At any one time, there are eight usable pointers, four per bank, and the four pointer are in consecutive locations in RAM. For example, if S3/S4=1 in the status register, then D0:0/D1:0/D2:0/D3:0 refer to register locations 4/5/6/7 in RAM Bank 0.

Note: When the data pointers are being written to, a number is actually being loaded to Data RAM. In effect, these data pointers can be used as a limited method for writing to RAM.

EXT0–EXT3 are used to map external peripherals into the address space of the processor.

Note: The actual register RAM does not exist on the chip, but would exist as part of the external device (such as an A/D result latch). The External Address Bus, EA2–EA0, the External Data Bus, ED15–ED0 and the control signals $\overline{DS}$, WAIT and RD/$\overline{WR}$, are used to access external peripherals.

EXT4 is used by the 13-bit Timer. If the Timer is disabled, then this address can be used to access an external peripheral on the External Data Bus.

EXT5 and **EXT6** are used by the CODEC Interface channels 0 and 1 respectively. If a CODEC channel is disabled, the corresponding address can be used to access an external peripheral.

EXT7 is used to program wait states for EXT0–EXT6, and is not available for accessing an external peripheral.

If both the Timer and CODEC Interface are disabled, there are 7 addresses available to access external peripherals.

If both the Timer and CODEC Interface are enabled, there are 4 addresses available to access external peripherals.

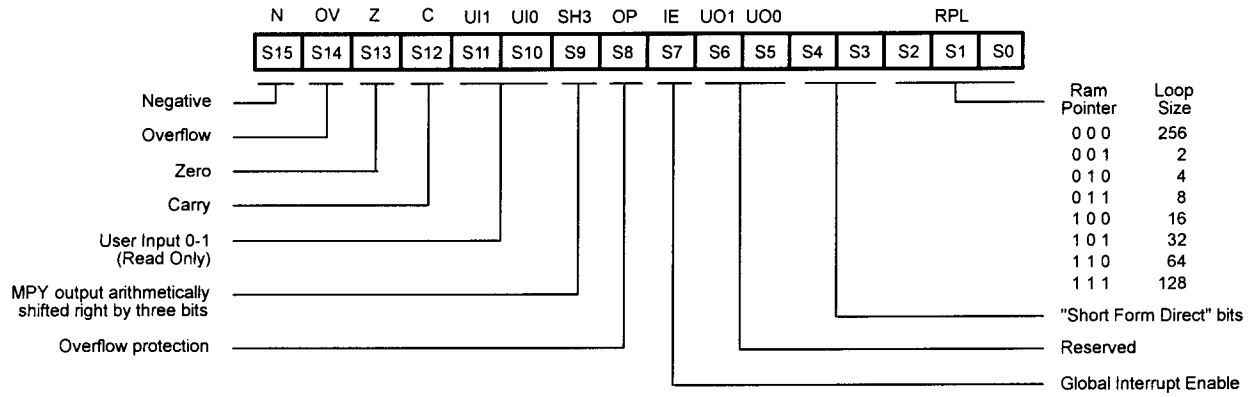


Figure 17. Status Register

13-BIT GENERAL PURPOSE TIMER

The General-Purpose Timer can be enabled or disabled. At power-on or $\overline{\text{RESET}}$, the counter is enabled. When the Timer is disabled, it can only be re-enabled by another $\overline{\text{RESET}}$. The Timer operates in a continuous or one-shot mode, and can be stopped. The Timer utilizes a 13-bit down-counter.

Continuous Mode With a load instruction, the user sets the Timer to run the mode, selects the clock source, and loads a non-zero count value:

1. When the down-counter reaches zero, an interrupt is generated on INT2,
2. The non-zero count value is automatically reloaded into the down-counter,
3. The process continues at step #1.

One-Shot Mode With a load instruction, the user sets the Timer to run the mode, selects the clock source, and loads a non-zero count value:

4. When the down-counter reaches zero, an interrupt is generated on INT2,
5. The user interrupt service routine must load a zero value into the Count Operation bit (D14 of EXT4),
6. The process stops.

Timing Intervals If the Timer clock source is CLK/2:

Time Interval = (count value) x (2/CLK)

Timer Frequency = (CLK/2) / (count value)

where CLK denotes the system clock frequency.

Extended Timing Intervals The Timer interval can be extended beyond 13-bits by using the Timer in conjunction with the CODEC Interface Counter/Timer. The count is thus extended to a maximum of 25 bits:

- 12-bits from the CODEC counter/timer
- 13-bits from the Timer

If the Timer clock source is the CODEC counter output:

Time Interval =(Timer count value) x (CODEC counter/timer period)

Timer Freq. = (CODEC counter/timer freq.) ÷ (Timer count value)

Timer Interrupt Behavior The following clarifies the behavior of the Timer interrupt:

- While the Timer is enabled, it utilizes theINT2 service routine address.
- The Timer is enabled after $\overline{\text{RESET}}$; however, the Timer is in stop mode.
- The INT2 pin has an internal pull-down.
- When the Timer is in run mode, it generates an interrupt each time it counts down to zero.
- When the Timer is disabled, INT2 can be controlled by an external peripheral.

Note: If the Timer is to be disabled, and an external peripheral is driving INT2, it should hold INT2 High while the Timer is being disabled.

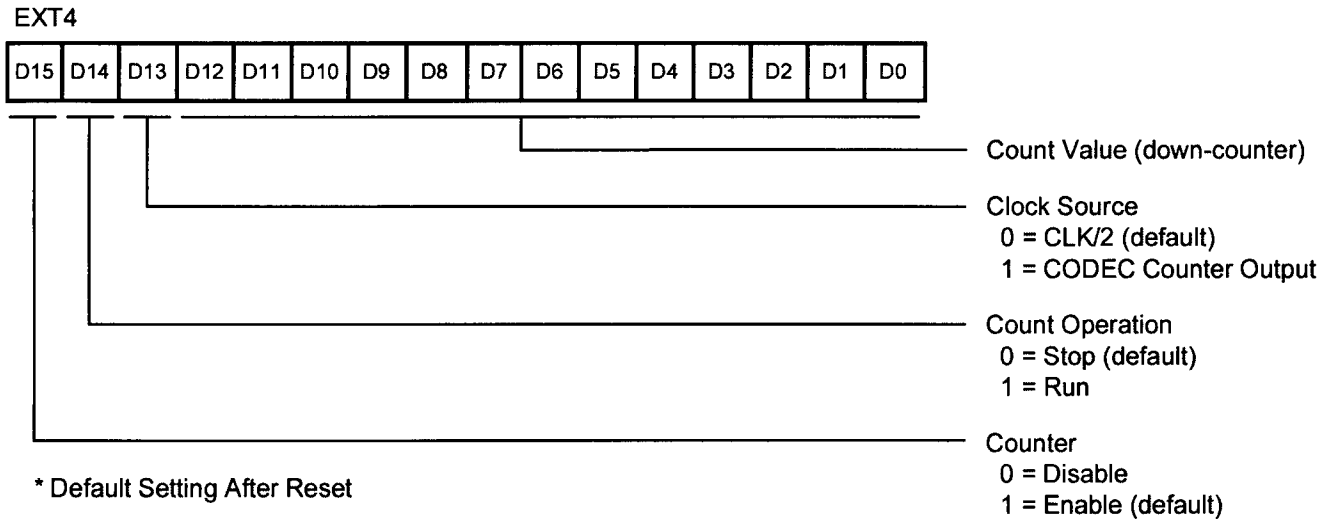


Figure 18. Timer Register EXT4

CODEC INTERFACE

Overview

The CODEC Interface not only supports a variety of external 8-bit, 16-bit linear, 64-bit sigma-delta stereo CODECs, and external A/D and D/A Converters, but the interface can also be used as a general purpose high-speed serial port. The CODEC Interface includes optional hardware μ-Law compression. The CODEC Interface is designed to support both Half-Duplex and Full-Duplex operation. The CODEC Interface is designed to operate in master mode only. The CODEC Interface generates a serial clock and two Frame Sync signals, which allows for two channels of data.

Hardware

The CODEC Interface hardware uses six 16-bit registers, μ-Law compression logic, and general-purpose control logic to control transfers to/from the appropriate registers.

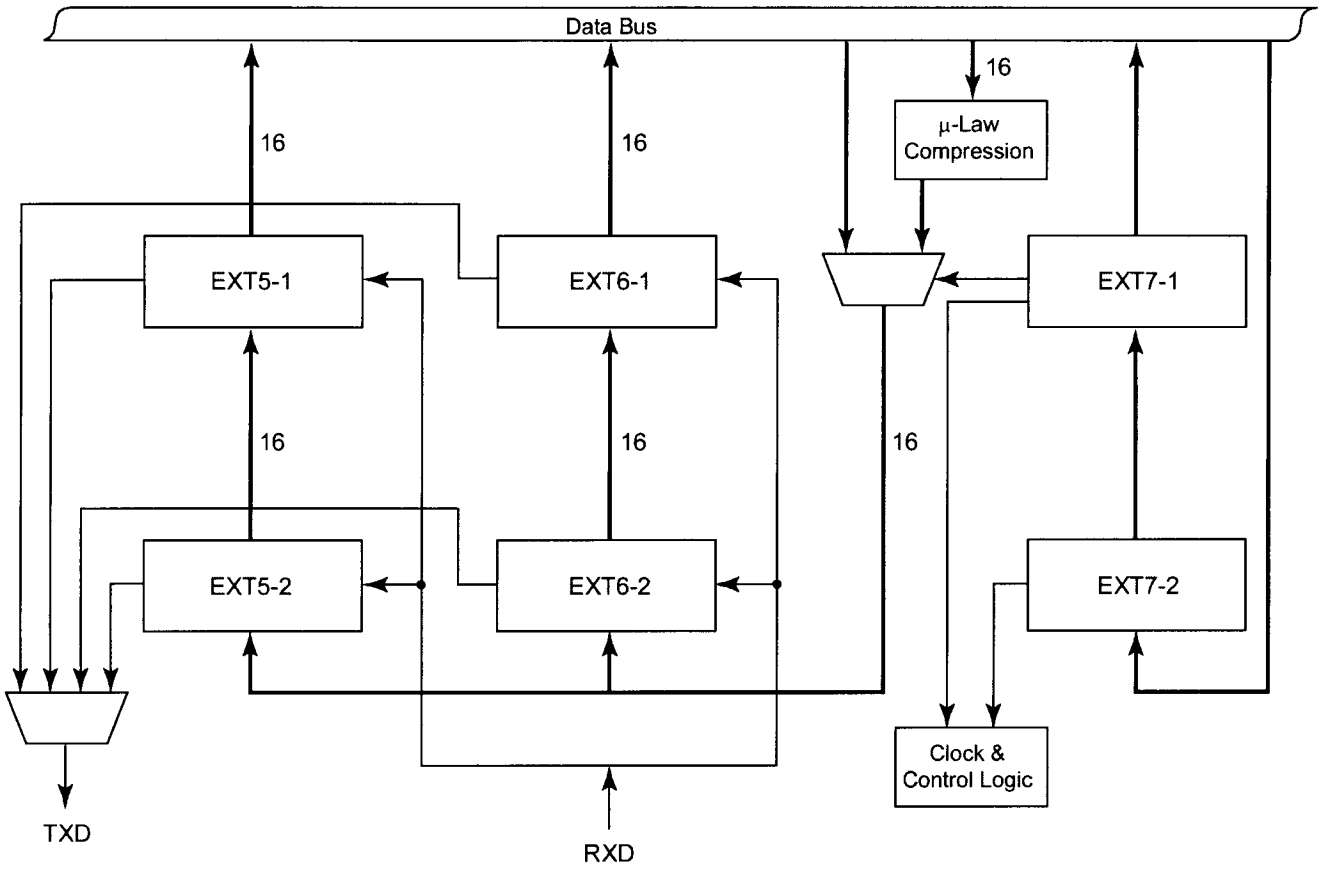


Figure 19. CODEC Interface Block Diagram

CODEC Interface Control Signals

SCLK. Serial Clock (output). This pin provides the clock signal for operating the external CODEC. A 4-bit prescaler is used to divide down the system clock (CLK) to produce the desired output frequency of SCLK. An internal divide-by-two is performed on CLK before passing it to the SCLK prescaler:

$$SCLK = (CLK/2) \div PS$$

where PS = 2's complement of the 4-bit Pre-Scaler value (PS is an up-counter).

TXD. Serial Output Data (output). This pin provides 8, 16, and 64-bit data transfers. Each bit is clocked out of the processor by the rising edge of SCLK, with the MSB transmitted first.

RXD. Serial Input Data (input). This pin provides 8, 16, and 64-bit data transfers. Each bit is clocked into the processor by the falling edge of SCLK, with the MSB received first.

FS0, FS1. Frame Sync 0 and Frame Sync 1 (output). These pins are used to mark data transfer/receive frames. The rising and falling edge of the Frame Sync signals indicate the beginning and the end of each serial data transmission.

CODEC Interface Interrupt Behavior

When the transmission of serial data is completed, the CODEC Interface generates an internal interrupt which vectors to the INT1 service routine address. This interrupt is coincident with the falling edge of FS1. The following clarifies the behavior of the CODEC Interface interrupt:

- While the CODEC Interface is enabled, it utilizes the INT1 service routine address.
- The CODEC Interface will be disabled after $\overline{\text{RESET}}$.
- The INT1 pin has an internal pull-down.
- If INT1 is tied High, the CODEC Interface generates an interrupt at the end of each frame transfer.
- If INT1 is not connected, or tied Low, the CODEC Interface not only generates an interrupt when first enabled, but generates an interrupt at the end of each frame transfer.
- When the CODEC Interface is disabled, INT1 can be controlled by an external peripheral.

Note: In single channel applications, use Channel 1 because INT1 coincides with FS1, not FS0.

Registers

The CODEC Interface registers (EXT5, EXT6 and EXT7) each act as a 2-deep FIFO. See the CODEC Interface Block Diagram for more information, Figure 19. Two operations may be required for some data transfers.

EXT5 and **EXT6.** The CODEC Interface constantly transfers and receives data during normal operation. The reading of receive data, and the writing of transmit data, are interleaved.

An example of Channel 1 operation in 8 or 16-bit mode, where one can wait for the input data, is as follows:

```
LD<dest>, EXT6          ; Read previous input data from
                          EXT6–1
LD EXT6, <xmit data>    ; Push current data from EXT6–2
                          to EXT6–1
                        ; Load EXT6–2 with data to be
                          transmitted
```

To obtain the input data as soon as it arrives, and extra instruction is required:

```
LD EXT6,<anything>      ; Push current input data from
                          EXT6–2 to EXT6–1
LD<test>, EXT6          ; Read current input data from
                          EXT6–1
LD EXT6, <xmit data>    ; Load EXT6–2 with data to be
                          transmitted
```

For 64-bit mode, one can use the following code sequence:

```
LD <Ch. 0 MSW input dest>, ; Get MSW of Ch. 0 input
EXT5
LD <Ch. 1 MSW input dest>, ; Get MSW of Ch 1 input
EXT6
LD EXT5,                ; Move LSB of Ch. 0 input
<Ch. 0 MSW output data>   and Load MSW of output
LD EXT6,                ; Move LSB of Ch. 1 input
<Ch. 1 MSW output data>   and Load MSW of output
LD<Ch. 0 LSW input test>, ; Get LSW of Ch. 0 input
EXT5
LD<Ch. 1 LSW input test>, ; Get LSW of Ch. 1 input
EXT6
LD EXT5,                ; Load LSW of Ch. 0 output
<Ch. 0 LSW output data>
LD EXT6,                ; Load LSW of Ch. 1 output
<Ch. 1 LSW output data>
```

Note: EXT# denotes EXT5 or EXT6.

In the 8 and 16-bit modes, EXT5–2 and EXT6–2 are the shift registers for Channel 0 and Channel 1, respectively. In 8-bit mode, the 8-bits reside in the least significant byte for both transmit and receive. In 64-bit mode, the output/input order is EXT 5–1 first, followed by EXT5–2, EXT6–1, and finally by EXT6–2. In all modes, the MSB is shifted out/in first.

Channel 0 uses FS0, EXT5–1, and EXT5–2. Channel 1 uses FS1, EXT6–1, and EXT6–2.

EXT7. This register contains the configuration information for the CODEC Interface and the Wait-State Generator. In normal operation, the user writes configuration data for EXT7–1 followed by configuration data for EXT7–2.

```
Write EXT7 LD EXT7, <config data1>  ; Move data to 7–2
           LD EXT7, <config data2>  ; Move data to 7–1
```

CODEC INTERFACE (Continued)

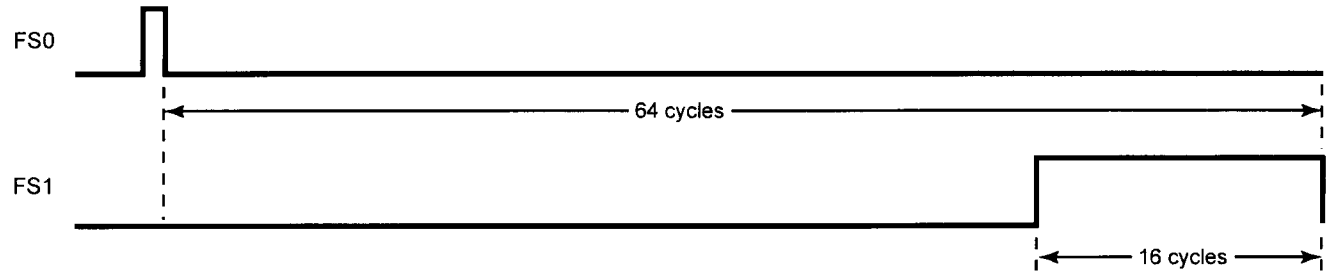


Figure 20. 64-Bit CODEC Frame Synchronization

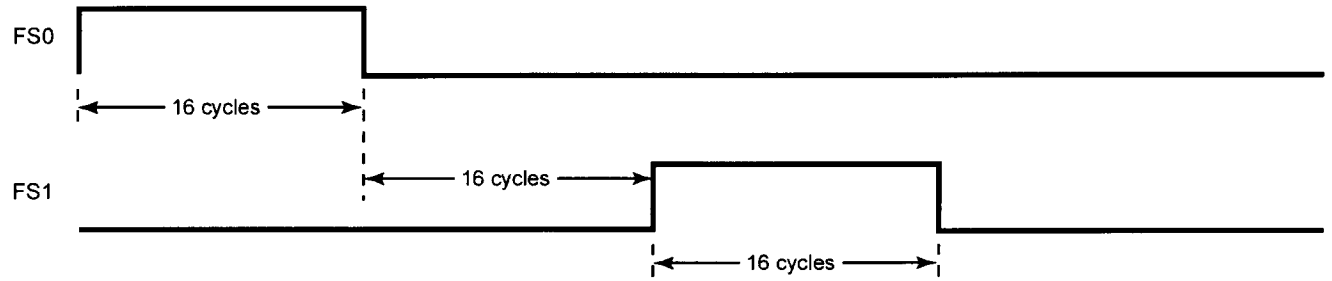


Figure 21. 16-Bit CODEC Frame Synchronization

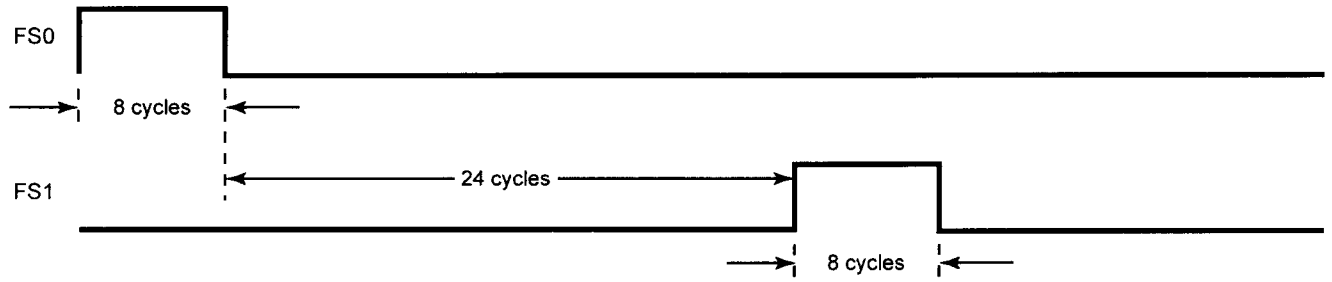


Figure 22. 8-Bit CODEC Frame Synchronization

CODEC INTERFACE (Continued)

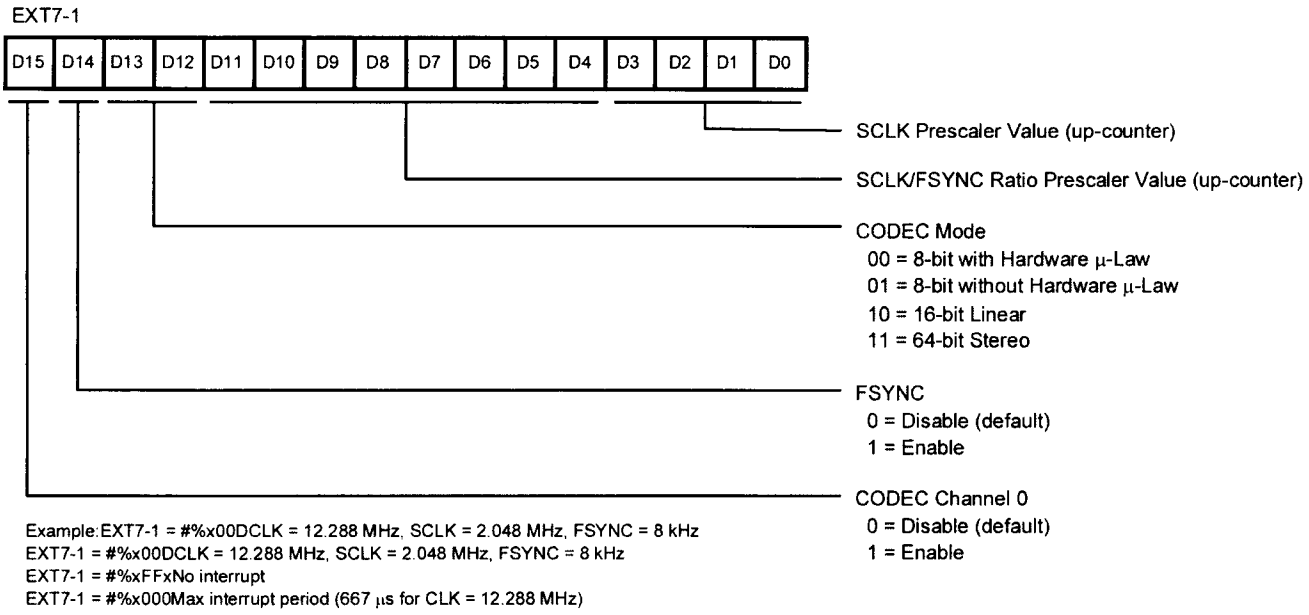


Figure 23. CODEC Interface Control Register

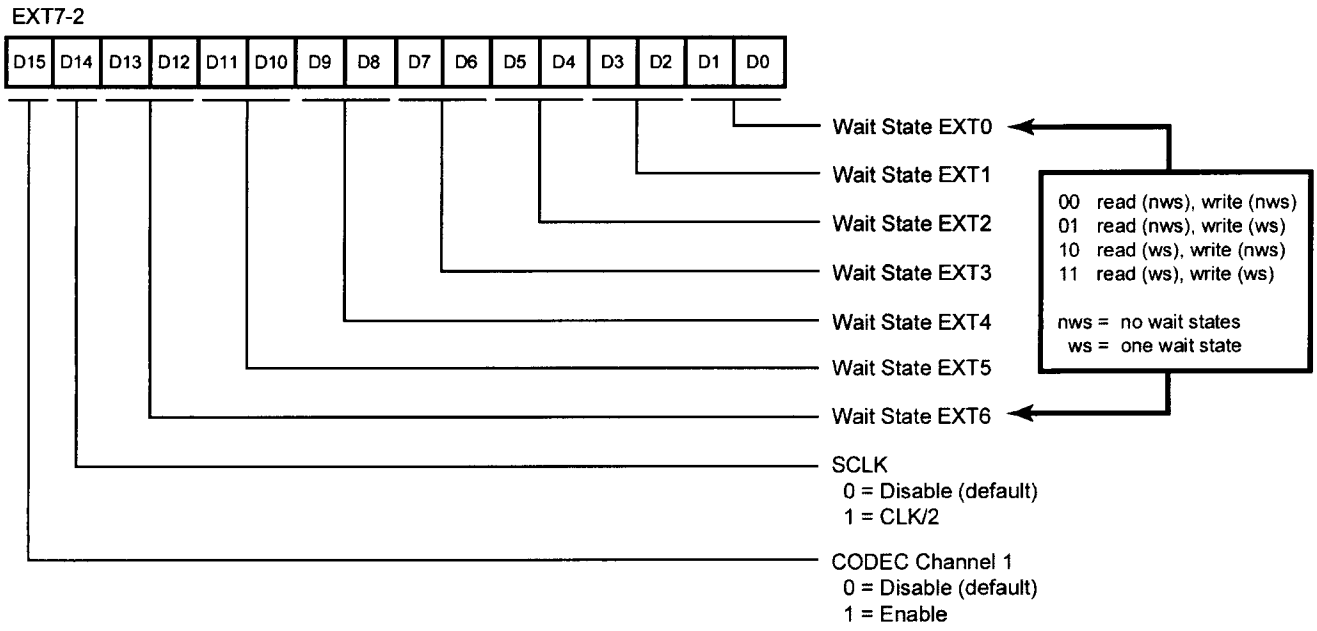


Figure 24. Wait-State Generator and CODEC Interface Control Register

CONDITION CODES

The following Instruction Description defines the condition codes supported by the DSP assembler. If the instruction description refers to the < cc > (condition code) symbol in one of its addressing modes, the instruction will only execute if the condition is true.

Code	Description
C	Carry
EQ	Equal (same as Z)
F	False
IE	Interrupts Enabled
MI	Minus
NC	No Carry
NE	Not Equal (same as NZ)
NIE	No Interrupts Enabled
NOV	No Overflow
NU0	Not User Zero
NU1	Not User One
NZ	Not zero
OV	Overflow
PL	Plus (Positive)
U0	User Zero
U1	User One
UGE	Unsigned Greater Than or Equal (Same as NC)
ULT	Unsigned Less Than (Same as C)
Z	Zero

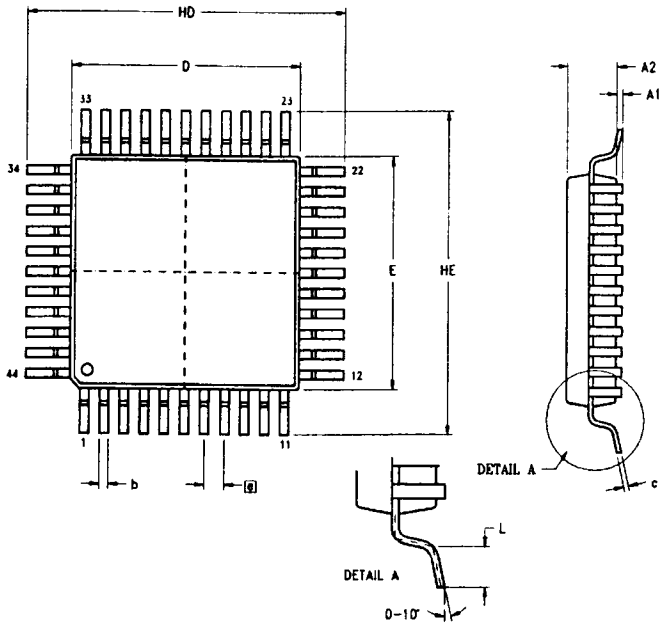
Inst.	Description	Synopsis	Operands	Words	Cycles	Examples
Notes: 1. When <dest> is <hwregs>, <dest> cannot be P. 2. When <dest> is <hwregs> and <src> is <hwregs>, <dest> cannot be EXTn if <src> is EXTn, <dest> cannot be X if <src> is X, <dest> cannot be SR if <src> is SR. 3. When <src> is <accind> <dest> cannot be A.						
MLD	Multiply	MLD <src1>,<src2> [,<bank switch>]	<hwregs>,<regind> <hwregs>,<regind>, bank switch> <regind>,<regind> <regind>,<regind>, <bank switch>	1 1 1 1	1 1 1 1	MLD A,@P0:0+LOOP MLD A,@P1:0,OFF MLD @P1:1,@P2:0 MLD @P0:1,@P1:0,ON
Notes: 1. If src1 is <regind> it must be a bank 1 register. Src2's <regind must be a bank 0 register. 2. <hwregs> for src1 cannot be X. 3. For the operands <hwregs>, <regind> the <bank switch> defaults to OFF. For the operands <regind>, the <bank switch> defaults to ON.						
MPYA	Multiply and add	MPYA <src1>,<src2> [,<bank switch>]	<hwregs>,<regind> <hwregs>,<regind>, bank switch> <regind>,<regind> <regind>,<regind>, <bank switch>	1 1 1 1	1 1 1 1	MPYA A,@P0:0 MPYA A,@P1:0,OFF MPYA @P1:1,@P2:0 MPYA@P0:1,@P1:0,ON
Notes: 1. If src1 is <regind> it must be a bank 1 register. Src2's <regind> must be a bank 0 register. 2. <hwregs> for src1 cannot be X. 3. For the operands <hwregs>, <regind> the <bank switch> defaults to OFF. For the operands <regind>, the <bank switch> defaults to ON.						
MPYS	Multiply and subtract	MPYS <src1>,<src2> [,<bank switch>]	<hwregs>,<regind> <hwregs>,<regind>, bank switch> <regind>,<regind> <regind>,<regind>, <bank switch>	1 1 1 1	1 1 1 1	MPYS A,@P0:0 MPYS A,@P1:0,OFF MPYS @P1:1,@P2:0 MPYS @P0:1,@P1:0,ON
Notes: 1. If src1 is <regind> it must be a bank 1 register. Src2's <regind> must be a bank 0 register. 2. <hwregs> for src1 cannot be X. 3. For the operands <hwregs>, <regind> the <bank switch> defaults to OFF. For the operands <regind>, <regind> the <bank switch> defaults to ON.						
NEG	Negate	NEG <cc>,A	<cc>, A A	1 1	1 1	NEG MI,A NEG A
NOP	No operation	NOP	None	1	1	NOP
OR	Bitwise OR	OR <dest>,<src>	A,<pregs> A,<dregs> A,<limm> A,<memind> A,<direct> A,<regind> A,<hwregs> A,<simm>	1 1 2 1 1 1 1 1	1 1 2 3 1 1 1 1	OR A,P0:1 OR A, D0:1 OR A,#%2C21 OR A,@@P2:1+ OR A, %2C OR A,@P1:0–LOOP OR A,EXT6 OR A,#%12
POP	Pop value from stack	POP <dest>	<pregs> <dregs> <regind> <hwregs>	1 1 1 1	1 1 1 1	POP P0:0 POP D0:1 POP @P0:0 POP A

INSTRUCTION DESCRIPTIONS (Continued)

Inst.	Description	Synopsis	Operands	Words	Cycles	Examples
PUSH	Push value onto stack	PUSH <src>	<pregs>	1	1	PUSH P0:0
			<dregs>	1	1	PUSH D0:1
			<regind>	1	1	PUSH @P0:0
			<hwregs>	1	1	PUSH BUS
			<limm>	2	2	PUSH #12345
			<accind>	1	3	PUSH @A
			<memind>	1	3	PUSH @@P0:0
RET	Return from subroutine	RET	None	1	2	RET
RL	Rotate Left	RL <cc>,A	<cc>,A	1	1	RL NZ,A
			A	1	1	RL A
RR	Rotate Right	RR <cc>,A	<cc>,A	1	1	RR C,A
			A	1	1	RR A
SCF	Set C flag	SCF	None	1	1	SCF
SIEF	Set IE flag	SIEF	None	1	1	SIEF
SLL	Shift left logical	SLL	[<cc>,]A	1	1	SLL NZ,A
			A	1	1	SLL A
SOPF	Set OP flag	SOPF	None	1	1	SOPF
SRA	Shift right arithmetic	SRA<cc>,A	<cc>,A	1	1	SRA NZ,A
			A	1	1	SRA A
SUB	Subtract	SUB<dest>,<src>	A,<pregs>	1	1	SUB A,P1:1
			A,<dregs>	1	1	SUB A,D0:1
			A,<limm>	2	2	SUB A,#%2C2C
			A,<memind>	1	3	SUB A,@D0:1
			A,<direct>	1	1	SUB A,%15
			A,<regind>	1	1	SUB A,@P2:0–LOOP
			A,<hwregs>	1	1	SUB A,STACK
			A,<simm>	1	1	SUB A, #%12
XOR	Bitwise exclusive OR	XOR <dest>,<src>	A,<pregs>	1	1	XOR A,P2:0
			A,<dregs>	1	1	XOR A,D0:1
			A,<limm>	2	2	XOR A,#13933
			A,<memind>	1	3	XOR A,@@P2:1+
			A,<direct>	1	1	XOR A,%2F
			A,<regind>	1	1	XOR A,@P2:0
			A,<hwregs>	1	1	XOR A,BUS
			A,<simm>	1	1	XOR A, #%12

Bank Switch Operand. The third (optional) operand of the MLD, MPYA and MPYS instructions represents whether the bank switch is set ON or OFF. To more clearly represent this capacity, the keywords ON and OFF are used to state the status of the switch. These keywords are referenced in the instruction descriptions through the <bank switch> symbol. The most notable capability this item provides is that a source operand can be multiplied by itself (squared).

PACKAGE INFORMATION (Continued)



SYMBOL	MILLIMETER		INCH	
	MIN	MAX	MIN	MAX
A1	0.05	0.25	.002	.010
A2	2.00	2.25	.078	.089
b	0.25	0.45	.010	.018
c	0.13	0.20	.005	.008
HD	13.70	14.15	.539	.557
D	9.90	10.10	.390	.398
HE	13.70	14.15	.539	.557
E	9.90	10.10	.390	.398
e	0.80 TYP		.0315 TYP	
L	0.60	1.20	.024	.047

NOTES:
1. CONTROLLING DIMENSIONS : MILLIMETER
2. LEAD COPLANARITY : MAX .10 (.004")

Figure 27. 44-Pin LQFP Package Diagram

ORDERING INFORMATION

Z89321
Z8932120PSC
Z8932120VSC
Z8932120VEC
Z8932120FSC
Z8932120FEC
Z89371
Z8937120PSC
Z8937120VSC
Z8937120FSC

For fast results, contact your local ZiLOG sales office for assistance in ordering the part desired.

Codes

Package	P = Plastic DIP
	V = Plastic PLCC
	F = Plastic LOFP
Temperature	S = 0°C to +70°C
	E = –40°C to 85°C
Speed	20 = 20 MHz
Environmental	C = Plastic Standard

Example:

Z 89321 20 V S C is a Z89321, 20 MHz, PLCC, 0° to +70°C, Plastic Standard Flow

- C — Environmental Flow
- S — Temperature
- V — Package
- 20 — Speed
- 89321 — Product Number
- Z — ZiLOG Prefix

Pre-Characterization Product:

The product represented by this document is newly introduced and ZiLOG has not completed the full characterization of the product. The document states what ZiLOG knows about this product at this time, but additional features or non-conformance with some aspects of the document may be found, either by ZiLOG or its customers in the course of further application and characterization work. In addition, ZiLOG cautions that delivery may be uncertain at times, due to start-up yield issues.

Customer Support

For answers to technical questions about the product, documentation, or any other issues with Zilog's offerings, please visit Zilog's Knowledge Base at http://www.zilog.com/kb.

For any comments, detail technical questions, or reporting problems, please visit Zilog's Technical Support at http://support.zilog.com.