Winbond Electronics Corporation America

VOICE DSP

ISD-T360SA

VoiceDSP™ Digital Speech Processor with CID, CID on Call Waiting (CIDCW), Full-Duplex Speakerphone, Master/Slave CODEC Interface and Multiple Flash Support

The VoiceDSP™ product family combines multiple digital signal processing functions on a single chip for cost-effective solutions in telephony, automotive and consumer applications.

The VoiceDSP processor offers necessary features to modern telephony products, such as: high-quality speech record and playback, electrical and acoustic echo cancellation for full-duplex hands-free speakerphone operation.

The VoiceDSP Caller ID Type I feature complies with the Bellcore (US), French, Spanish, Dutch and Japanese standards. For the Bellcore (US) standard, the VoiceDSP implements CID Type II. The VoiceDSP implements the receiver side for data transmitted from the central office to the subscriber.

The T360SA VoiceDSP can be used in various applications:

- Digital telephony with add-on speech processing: Digital Telephone Answering Machines (DTADs), hands-free speakerphone operation for ISDN, DECT, Digital Spread Spectrum, and analog cordless applications; CT0/1 Base stations.
- An add-on chip for corded telephones featuring DTAD functions and/or full-duplex, hands-free speakerphone operation.
- Stand-alone digital answering machines with full-duplex, hands-free speakerphone operation.
- Voice memo recording
- Automotive applications employing full-duplex speakerphone operations for hands-free, in-car communications, and for car status and information announcements.

Based on ISD's unique concept which combines 16-bit DSP (Digital Speech Processor) and 16-bit RISC core technology, the ISD-T360SA is a high-performing chip solution for various applications. To facilitate incorporating the VoiceDSP processor, it features system support functions such as an interrupt control unit, codec interface (master, slave), Microwire interface to the system microcontroller, as well as a memory handler for Flash memory devices. Design of high-end, price optimized systems are possible with ISD's VoiceDSP flexible system interfaces (codec, microcontroller, and memory management support).

The ISD-T360SA processor operates as a peripheral controlled by the system microcontroller via an enhanced, serial Microwire interface. The system microcontroller typically controls the analog circuits, buttons and display, as well as activates functions through commands. The VoiceDSP executes these commands and returns status information to the Microcontroller.

The VoiceDSP software resides in the on-chip ROM. It includes DSP-based algorithms, system support functions, and a software interface to hardware peripherals.

February 2000

ISD · 2727 North First Street, San Jose, CA 95134 · TEL: 408/943-6666 · FAX: 408/544-1787 · http://www.isd.com

FEATURES AT A GLANCE

DTAD MANAGEMENT

- Highest quality speech recording in PCM format for music on hold or other OGM (Out Going Message) recording and IVS
- Selectable high-quality speech compression rate of 4.7 Kbit/s, 6.7 Kbit/s or 8.7 Kbit/s
- Up to approximately 15 minutes recording on a 4-Mbit Flash
- Up to 4 hour speech recording
- High-quality music compression for music on hold (64 Kbit/s PCM)
- Programmable message tag for message categorization, e.g., Mailboxes, Incoming Messages (ICM), OutGoing Messages (OGM)
- Message management
- Skip forward or backward during message playback
- Variable speed playback
- Real-time clock: Day of Week, Hours, Minutes
- Multi-lingual speech synthesis using International Vocabulary Support (IVS)
- Vocabularies available in: English, Japanese, Mandarin, German, French and Spanish
- Conversation Recording
- Software Automatic Gain Control (AGC)
- Supports Caller ID Type I for the Bellcore (US), French, Spanish, Dutch and Japanese standards
- Supports Caller ID on call waiting (Caller ID Type II) for the Bellcore (US) standard
- Stores caller numbers
- Automatic storage of Caller ID data of Incoming Messages (ICM)

SPEAKERPHONE

- Digital full-duplex speakerphone
- Acoustic- and line-echo cancellation
- Continuous on-the-fly monitoring of external and internal conditions (acoustic and line) provides high-quality, hands-free, conversation in a changing environment
- Minimum microcontroller control intervention (Launch-and-forget)
- Supports: On, Off, Mute, and Hold functions
- Additional Software Automatic Gain Control for improved speakerphone performance

CALL AND DEVICE MANAGEMENT

- Digital volume control
- Least cost routing support (LCR)
- Power-down mode
- 3.3V or 5V selectable power supply
- 32.7 MHz internal operation using 4.096 MHz external crystal oscillator
- Available in the PQFP 100-pin package
- DTMF generation and detection
- Telephone line functions including busy, dial tone detection and VOX detection
- Single tone generation
- DTMF detection during message record and playback

PERIPHERAL CONTROL

Codec

- μ-Law, A-Law, and 16-bit linear codec input support
- Selectable master/slave codec interface. In addition, the slave mode is implemented with respect to IOM-2™/GCI specifications
- Supports two selectable in-coming lines in slave mode without speakerphone for DTAD recording
- Supports one in-coming line in slave mode and one master speakerphone interface
- Supports up to 32 user-selectable speech channels in slave mode
- Software Automation Gain Control (AGC)
- Supports long-frame and short-frame codecs
- Single/double bit clock rate for slave mode
- On-chip codec clock generation

Memory

- Supports up to four 4-Mbit, four 8-Mbit, or four 16-Mbit Flash devices from Toshiba or Samsung
- The number of messages that can be stored is limited only by memory size
- Direct access to message memory
- Message storage contains all data in a concatenated chain of memory blocks.
- Memory mapping and product floor test included
- Supports external vocabularies, using Flash memory or expansion ROM

Microwire

- MICROWIRE slave interface to an external microcontroller
- Sophisticated command language to optimize system code size

INTERNATIONAL VOCABULARY SUPPORT (IVS)

For manufacturing recorded voice prompt and speech synthesis, the ISD International Vocabulary Support delivers pre-recorded voice prompts in the same high-quality of the user-recorded speech. For complete control over quality and memory management, the IVS features adjustable speech compressions. In addition, several pre-recorded voice prompt sets are available in various languages for further convenience.

Available Languages:

- English
- Japanese
- Mandarin
- German
- French
- Spanish

You can also develop a new vocabulary using the ISD-T360 IVSTOOL application. This PC-Windows95/98/2000™-based application synthesizes recorded .wav files into the ISD-T360SA's various compression rates (including PCM). The IVSTool application supports various languages, including their unique grammar structures.

ISD's VoiceDSP products store IVS vocabularies on either Flash memory or expansion ROM memories, thus DTAD manufacturers can design a product for multiple countries, featuring various languages.

For more details about IVS, refer to the *IVS User's Guide*.

Figure 1-1: ISD-T360SA Block Diagram—Basic Configuration with Four 4/8/16Mbit NAND Flash Devices (Samsung/Toshiba)

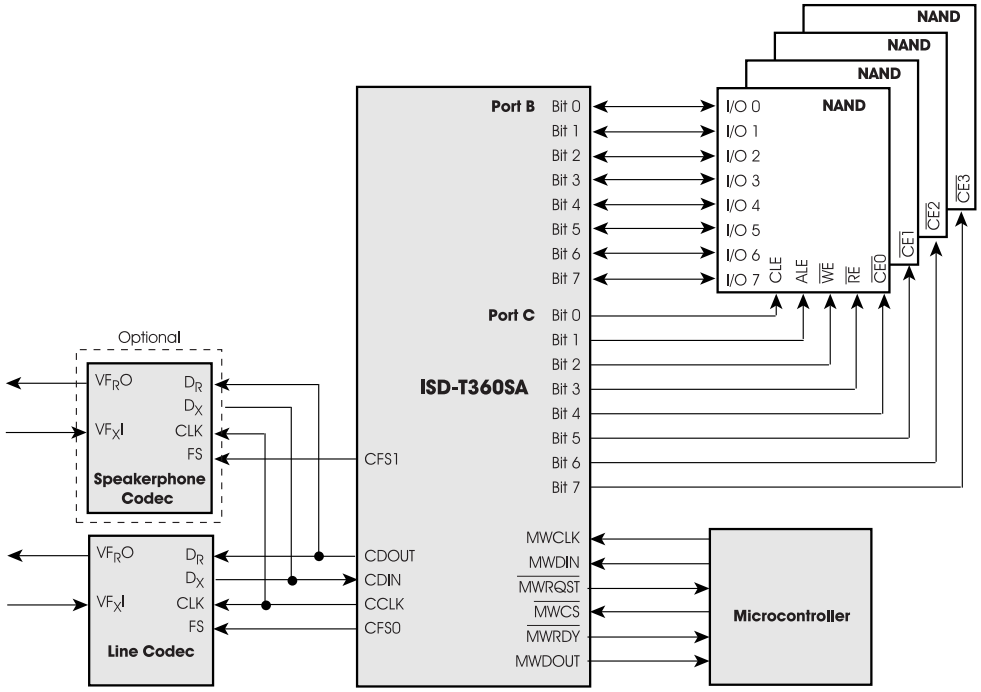


Figure 1-2: ISD-T360SA Block Diagram-Basic Configuration with IVS EPROM

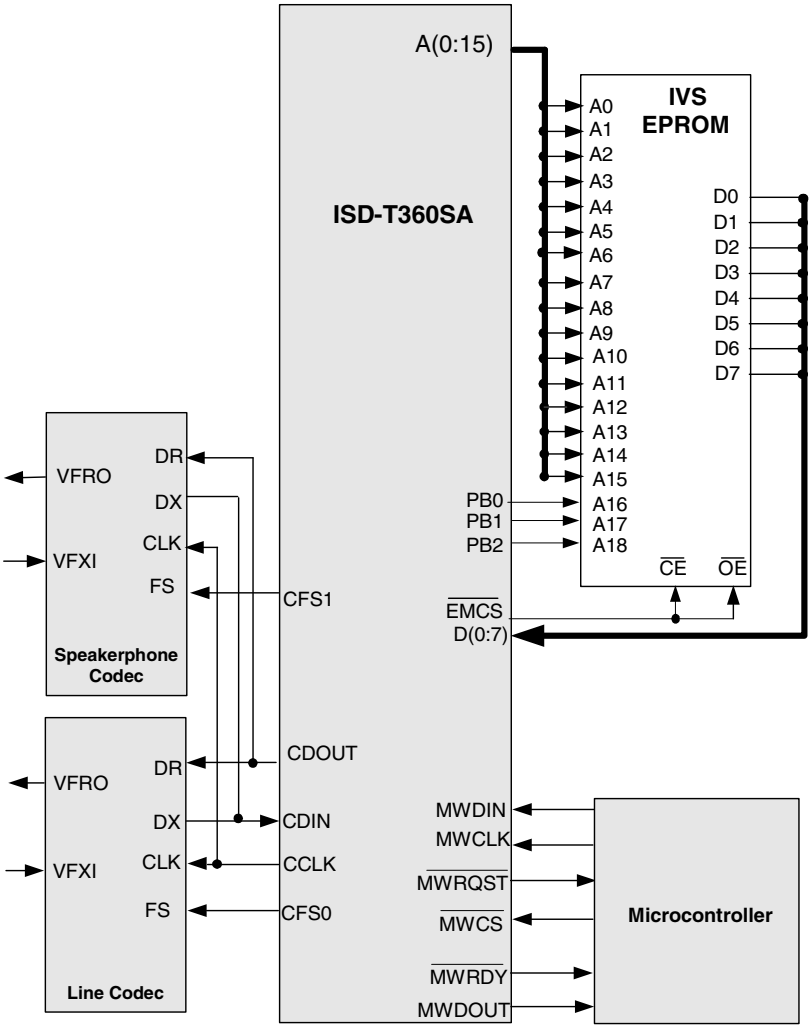


Table of Contents

Chapter 1—HARDWARE

1.1 PIN ASSIGNMENT

The following sections detail the pins of the ISD-T360SA processor. Slashes separate the names of signals that share the same pin.

1.1.1 PIN-SIGNAL ASSIGNMENT

Table 1-1 shows all the pins and the signals that use them in different configurations. It also shows the type and direction of each signal.

Figure 1-1: 100-PQFP Package Connection Diagram

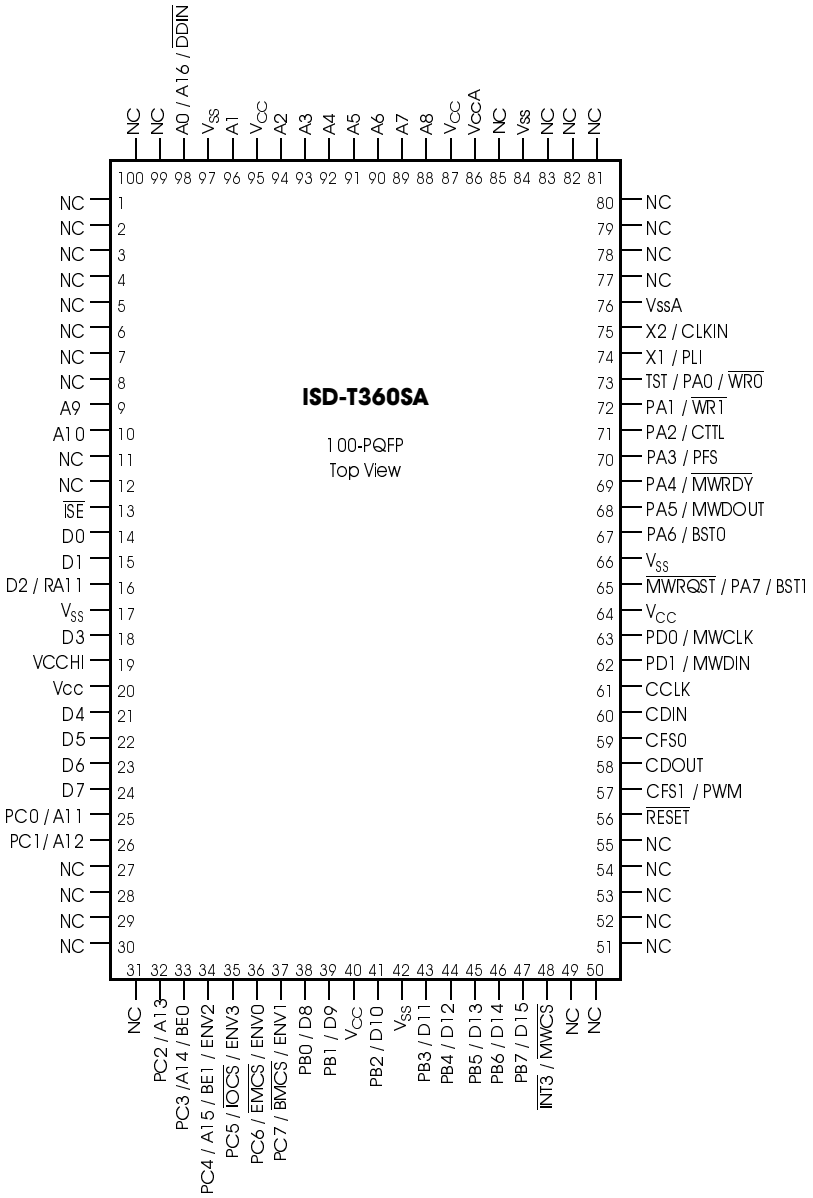


Table 1-1: VoiceDSP Pin Signal Assignment

Pin Name	Signal Name	Type	Description
A(0:15)	A(0:15)	Output	Address bits 0 through 15
CCLK	CCLK	I/O	Codec Master/slave Clock
CDIN	CDIN	Input	Data Input from Codec
CDOUT	CDOUT	Output	Data Output to Codec
CFS0	CFS0	I/O	Codec 0 Frame Synchronization
CFS1	CFS1	Output	Codec 1 Frame Synchronization
D(0:7)	D(0:7)	I/O	Data bits 0 through 7
$\overline{\text{EMCS}}$/ENV0	$\overline{\text{EMCS}}$	Output	Expansion Memory Chip Select
$\overline{\text{EMCS}}$/ENV0	ENV0	Input[1]	Environment Select
MWCLK	MWCLK	Input[4]	MICROWIRE Clock
$\overline{\text{MWCS}}$	$\overline{\text{MWCS}}$	Input[4]	MICROWIRE Chip Select
MWDIN	MWDIN	Input[4]	MICROWIRE Data Input
MWDOUT	MWDOUT	Output	MICROWIRE DATA Output
$\overline{\text{MWRDY}}$	$\overline{\text{MWRDY}}$	Output	MICROWIRE Ready
$\overline{\text{MWRQST}}$	$\overline{\text{MWRQST}}$	Output	MICROWIRE Request Signal
PB(0:7)[2,3]	PB(0:7)	I/O	Port B, bits 0 through 7
PC(0:7)	PC(0:7)	Output	Port C, bits 0 through 7
$\overline{\text{RESET}}$	$\overline{\text{RESET}}$	Input[4]	Reset
$\overline{\text{TST}}$	$\overline{\text{TST}}$	Input	Test pin
V_{CC}	V_{CC}	Power	3.3 V power supply pin
$V_{CC}A$	$V_{CC}A$	Power	3.3 V analog circuitry power supply pin
$V_{CC}HI$	$V_{CC}HI$	Power	5 V power supply pin. Connect to V_{CC} if 3.3 V power supply is used.
V_{SS}	V_{SS}	Power	Ground for on-chip logic and output drivers
$V_{SS}A$	$V_{SS}A$	Power	Ground for on-chip analog circuitry
X1	X1	Oscillator	Crystal Oscillator Interface
X2/CLKIN	X2	Oscillator	Crystal Oscillator Interface

1. *Input during reset, otherwise output.*
2. *Virtual address lines for IVS ROM, bits [0:2]*
3. *Chip select lines for Flash devices, bits [3:6]*
4. *Schmitt trigger input.*

1.2 DESCRIPTION

This section provides details of the functional characteristics of the VoiceDSP processor. It is divided into the following sections:

- Resetting
- Clocking
- Power-Down Mode
- Power and Grounding
- Memory Interface
- Codec Interface

1.2.1 RESETTING

The $\overline{\text{RESET}}$ pin is used to reset the VoiceDSP processor.

On application of power, $\overline{\text{RESET}}$ must be held low for at least t_{pwr} after V_{CC} is stable. This ensures that all on-chip voltages are completely stable before operation. Whenever $\overline{\text{RESET}}$ is applied, it must also remain active for not less than t_{RST}, see Table 1-9 and Table 1-10. During this period, and for 100 ms after, the $\overline{\text{TST}}$ signal, which is an internal signal, must be high. This can be done with a pull-up resistor on the $\overline{\text{TST}}$ pin.

The value of $\overline{\text{MWRDY}}$ is undefined during the reset period, and for 100 ms after. The microcontroller should either wait before polling the signal for the first time, or the signal should be pulled high during this period.

Upon reset, the ENV0 signal is sampled to determine the operating environment. During reset, the $\overline{\text{EMCS}}$/ENV0 pin is used for the ENV0 input signals. An internal pull-up resistor sets ENV0 to 1.

After reset, the same pin is used for $\overline{\text{EMCS}}$.

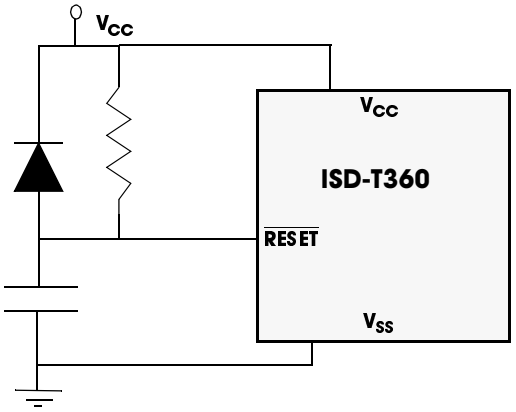
System Load on ENV0

For any load on the ENV0 pin, the voltage should not drop below V_{ENVh} (see Table 1-8).

If the load on the ENV0 pin causes the current to exceed 10 μA, use an external pull-up resistor to keep the pin at 1.

Figure 1-2 shows a recommended circuit for generating a reset signal when the power is turned on.

Figure 1-2: Recommended Power-On Reset Circuit



1.2.2 CLOCKING

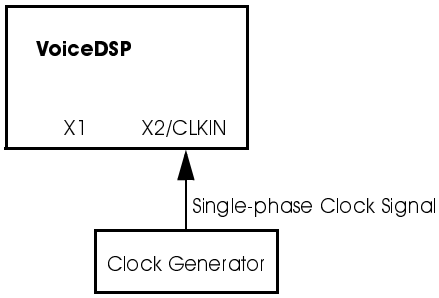
The VoiceDSP processor provides an internal oscillator that interacts with an external clock source through the X1 and X2/CLKIN pins. Either an external single-phase clock signal, or a crystal oscillator, may be used as the clock source.

External Single-Phase Clock Signal

If an external single-phase clock source is used, it should be connected to the CLKIN signal as shown in Figure 1-3, and should conform to the voltage-level requirements for CLKIN stated in "ELECTRICAL CHARACTERISTICS" on page 1-16.

NOTE the CLKIN signal is not 5V tolerant.

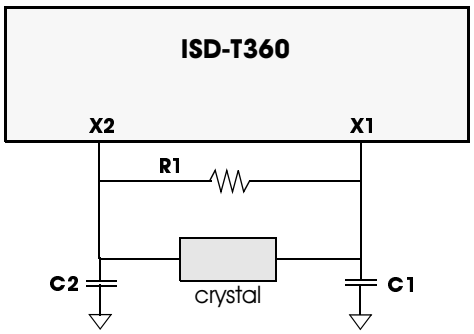
Figure 1-3: External Clock Source



Crystal Oscillator

A crystal oscillator is connected to the on-chip oscillator circuit via the X1 and X2 signals, as shown in Figure 1-4.

Figure 1-4: Connections for an External Crystal Oscillator



Keep stray capacitance and inductance, in the oscillator circuit, as low as possible. The crystal resonator, and the external components, should be as close to the X1 and X2/CLKIN pins as possible, to keep the trace lengths in the printed circuit to an absolute minimum.

You can use crystal oscillators with maximum load capacitance of 20 pF, although the oscillation frequency may differ from the crystal's specified value.

Table 1-2 lists the components in the crystal oscillator circuit

Table 1-2: Components of Crystal Oscillator Circuit

Component	Values	Tolerance
Crystal Resonator	4.096 MHz	50 PPM
Resistor R1	10 MΩ	5%
Capacitors C1, C2	33 pF	20%

1.2.3 POWER-DOWN MODE

Power-down mode is useful during a power failure or in a power-saving model, when the power source for the processor is a backup battery or in battery-powered devices, while the processor is in idle mode.

In power-down mode, the clock frequency of the VoiceDSP processor is reduced and some of the processor modules are deactivated. As a result, the ISD-T360SA consumes considerably less power than in normal-power mode. Although the VoiceDSP processor does not perform all its usual functions in power-down mode, it does retain stored messages and maintain the date and time.

NOTE In power-down mode all the chip select signals, CS0 to CS3, are set to 1. To guarantee that there is no current flow from these signals to the Flash devices, the power supply to these devices must not be disconnected.

The ISD-T360SA stores messages and all memory management information in Flash memory. When Flash memory is used for memory management, power does not need to be maintained to the processor to preserve stored messages.

To keep power consumption low during power-down mode, the $\overline{\text{RESET}}$, $\overline{\text{MWCS}}$, MWCLK and MWDIN signals should be held above V_{CC} – 0.5 V or below V_{SS} + 0.5 V.

1.2.4 POWER AND GROUNDING

Power Pin Connections

The ISD-T360SA can operate over two supply voltage ranges 3.3 V ±10% and 5 V ±10%. The power supply and ground pins (V_{CC}, V_{SS}, $V_{CC}A$, $V_{SS}A$ and $V_{CC}HI$) must be connected as shown in Figure 1-5 when operating in a 3.3 V environment, and as shown in Figure 1-6 when operating in a 5 V environment. Failure to correctly connect the pins may result in damage to the device.

The Capacitor and Resistor values are given in Table 1-3.

Table 1-3: Components of Supply Circuit

Component	Values	Tolerance
Resistor R1	10 Ω	5%
Capacitors C1, C2, C3, C4, C5, C6, C7	0.1 µF Ceramic	20%
Capacitors C8, C9, C10, C11, C12, C13, C14	2.2 µF Tantalum	20%

Figure 1-5: 3.3 V Power Connection Diagram

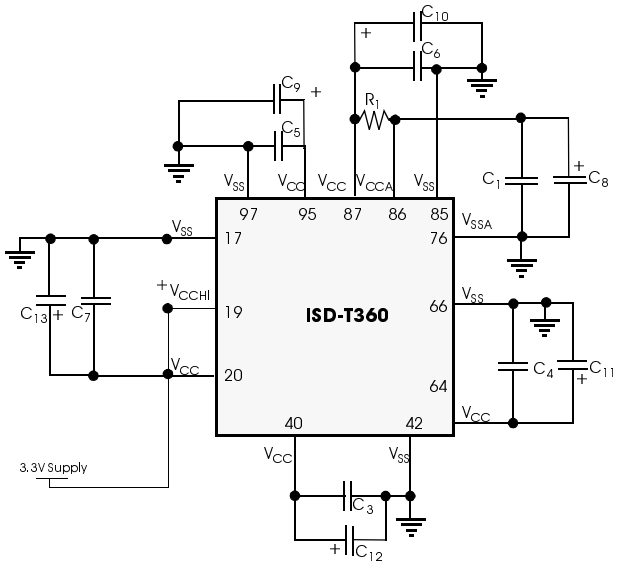
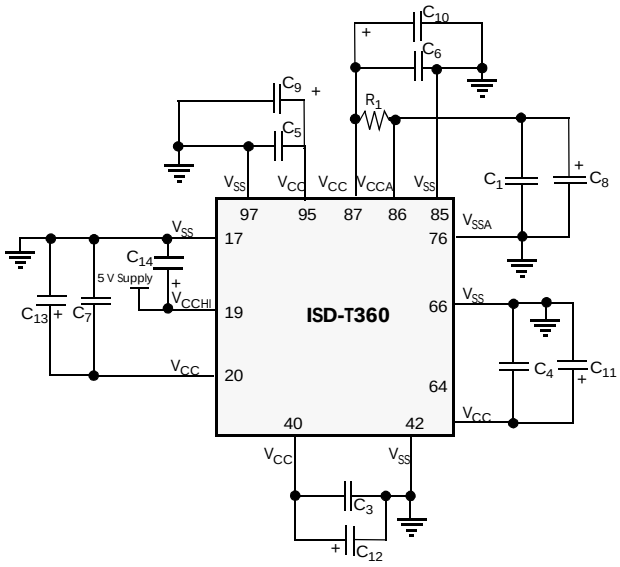


Figure 1-6: 5 V Power Connection Diagram



For optimal noise immunity, the power and ground pins should be connected to V_{CC} and the ground planes, respectively, on the printed circuit board. If V_{CC} and the ground planes are not used, single conductors should be run directly from each V_{CC} pin to a power point, and from each GND pin to a ground point. Avoid daisy-chained connections. The VoiceDSP does not perform its usual functions in power-down mode but it still preserves stored messages, maintains the time of day.

When you build a prototype, using wire-wrap or other methods, solder the capacitors directly to the power pins of the VoiceDSP processor socket, or as close as possible, with very short leads.

1.2.5 MEMORY INTERFACE

Flash Support

The ISD-T360SA VoiceDSP supports Flash devices for storing recorded data, thus, power can be disconnected to the ISD-T360SA without losing data. The ISD-T360SA supports NAND Flash device interfaces, such as TC58V16BFT, TC5816BFT, TC58A040F, KM29N040T, KM29W8000T, and KM29W16000AT. The ISD-T360SA may be connected to up to four Flash devices, resulting with maximum recording storage of 16-Mbits x 4 = 64 Mbits (up to 4 hours of recording time).

The following flash devices are supported:

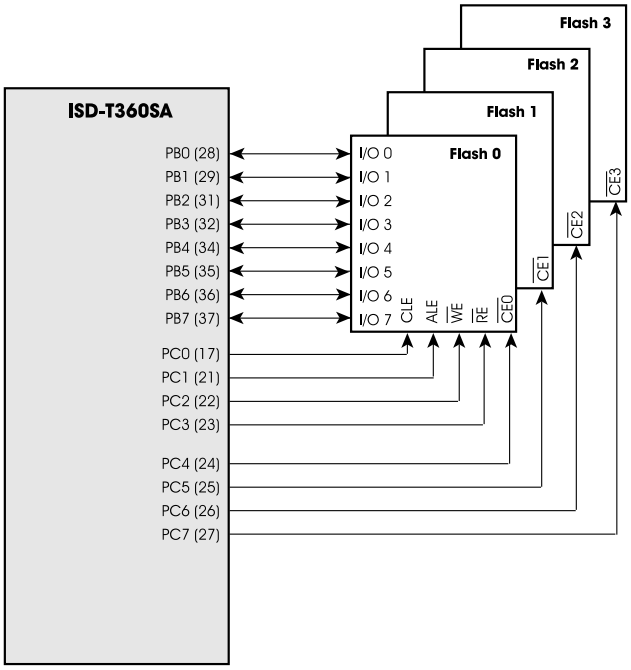
Table 1-4: Supported Flash Devices

Manufacturer	Memory Device Name	Characteristics	Operating Voltage	Memory Size
Toshiba	TC58V16BFT	2Mx8	3.3 V	16-Mbit LV
Toshiba	TC5816BFT	2Mx8	5.0 V	16-Mbit
Samsung	KM29W040T	512Kx8	2.7V - 5.0 V	4-Mbit
Samsung	KM29W8000T	1Mx8	2.7V - 5.0 V	8-Mbit
Samsung	KM29W16000AT	2Mx8	2.7V - 5.0 V	16-Mbit

Internal Memory Organization

The Flash devices detailed in Table 1-4 divide internally into basic 4-Kbyte block units. The ISD-T360SA uses one block on each device for memory management, leaving the rest of the blocks available for recording. Using at least one block for a single recorded message yields a maximum of NUM_OF_BLOCKS_IN_MEM – 1 (see Table 2-10 for parameter definition) messages per device.

Figure 1-7: Memory Interface with Four 4/8/16 Mbit, NAND Flash Devices (Samsung, Toshiba)



NAND Flash (Samsung, Toshiba)

The VoiceDSP processor supports up to four NAND interface Flash memory devices for storing messages. A flash device with the NAND interface uses a single 8bit I/O port to set the address and access the data. The ISD-T360SA supports three types of Flash volumes (4Mbit, 8Mbit and 16Mbit as listed inTable 1-4) while all the connected Flash devices must be of the same type. Ports B and C are used to connect ISD-T360SA to the Flash devices using port B for address and data transfer and port C for communication control and chip select. Connecting less than four Flash devices requires connecting the Flash devices sequentially, starting from PC4 up to PC7 (see Figure 1-7). Upon initialization, the ISD-T360SA scans the flash devices to detect defected blocks. The defected blocks are marked in a special map, which is located in the last block of each device.

Flash Endurance

A Flash memory may be erased a limited number of times. To maximize the Flash use, the memory manager utilizes the Flash's blocks evenly (i.e., each block is erased more or less the same number of times), to ensure that all blocks have the same lifetime. Refer to the respective Flash memory device data sheets for specific endurance specifications.

A VoiceDSP processor message uses at least one block. The maximum recording time depends on four factors:

1. The basic compression rate (4.7 Kbit/s, 6.7 Kbit/s or 8.7 Kbit/s).
2. The amount of silence in the recorded speech
3. The number of bad blocks
4. The number of recorded messages. (The basic memory allocation unit for a message is a 4-Kbyte block, which means that half a block, on average, is wasted per recorded message.)

Table 1-5: Recording Time with No Silence Compression

Memory Size	Compression Rate	Total Recording Time
4 Mbit	4.7 Kbit/s	14.5 Minutes
4 Mbit	6.7 Kbit/s	10.2 Minutes
4 Mbit	8.7 Kbit/s	7.8 Minutes
8 Mbit	4.7 Kbit/s	29.1Minutes
8 Mbit	6.7 Kbit/s	20.4 Minutes
8 Mbit	8.7 Kbit/s	15.7 Minutes
16 Mbit	4.7 Kbit/s	58.1Minutes
16 Mbit	6.7 Kbit/s	40.8 Minutes
16 Mbit	8.7 Kbit/s	31.4 Minutes
32 Mbit	4.7 Kbit/s	116.2 Minutes
32 Mbit	6.7 Kbit/s	91.5 Minutes
32 Mbit	8.7 Kbit/s	62.8Minutes

Rom interface

IVS vocabularies can be stored in either Flash memory and/or ROM. The VoiceDSP processor supports IVS ROM devices through an Expansion Memory mechanism. Up to 64 Kbytes (64K x 8) of Expansion Memory are directly supported. Nevertheless, the processor uses bits of the on-chip port (PB) to further extend the 64 Kbytes address space up to 0.5 Mbytes address space.

ROM is connected to the VoiceDSP processor using the data bus, D(0:7), the address bus, A(0:15), the extended address signals, $\overline{EA}$(16:18), and Expansion Memory Chip Select, $\overline{EMCS}$, controls. The number of extended address pins to use may vary, depending on the size and configuration of the ROM. ISD-T360SA configured with NAND Flash memory cannot support extension ROM.

1.2.6 THE CODEC INTERFACE

The ISD-T360SA provides an on chip interface for analog and digital telephony, supporting master and slave codec interface modes. In master mode, the ISD-T360SA controls the operation of the codec for use in analog telephony. In the slave mode, the ISD-T360SA codec interface is controlled by an external source. This mode is used in digital telephony (i.e., ISDN or DECT lines). The slave mode is implemented with respect to IOM-2™/GCI specifications.

See Table 1-6 for codec options for the ISD-T360SA (ISD supports compatible codecs in addition to those listed below).

The codec interface supports the following features:

- Master Mode or Slave Mode.
- 8- or 16-bit Channel Width.
- Long (Variable) or Short (Fixed) Frame Protocol.
- Single or Double Bit (Slave Mode Only) Clock Rate.
- Single or Dual Channel Codecs
- One or Two Codecs
- Multiple Clock And Sample Rates.
- One or Two Frame Sync Signals

Table 1-6: Supported Codec Options

Manufacturer	Codec Device Name	Characteristics	Operating Voltage	Conversion Type
National Semiconductor	TP3054	Single codec	5 V	μ-Law
National Semiconductor	TP 3057	Single codec	5 V	A-Law
Oki	MSM7533V	Dual codec	5 V	μ-Law, A-Law
Oki	MSM 7704	Dual codec	3.3 V	μ-Law, A-Law, LV
Lucent	T7502	Dual codec	5 V	A-Law
Lucent	T7503	Dual codec	5 V	μ-Law
Motorola	MC145482	Single codec	5 V	16-bit linear
Motorola	MC145484	Single codec	5 V	μ-Law, A-Law
Winbond	W6612	Dual codec	5 V	μ-Law, A-Law

Codec Configuration Options

The codec interface may be configured so that the ISD-T360SA is connected to either:

- one single codec (the default option) (using channel 0)
- two single codecs (using channels 0, 2)
- one dual codec (using channel 0, 1)

The Codec Interface Signals

The codec interface consists of five signals: CDIN, CDOUT, CCLK, CFS0 and CFS1. The first codec is connected to CDIN, CDOUT, CCLK and CFS0 (see Figure 1-9). The second codec is connected to CDIN, CDOUT, CCLK and CFS1 (see Figure 1-10).

Data is read from the codec through the CDIN input pin, while data is simultaneously transferred to the codec through the CDOUT output pin (see Figure 1-8).

The Codec Frame Sync 0 signal (CFS0) is generated by the codec interface in master mode, and is generated externally in slave mode.

The Codec Frame Sync 1 signal (CFS1) is generated by the codec interface in master and slave modes. The signal is generated with a delay relative to the CFS0 signal. CFS1 signal may be delayed between 0 and 255 CCLK cycles (see Figure 1-8).

The Codec Clock (CCLK) signal is used to synchronize the data in and out of the codec interface. The CCLK signal is used to indicate a delay (relative to CFS0) of data and CFS1 signals. The CCLK signal is generated according to the programmed clock rate in master mode, and is generated externally in slave mode.

In Slave Mode operation, the CCLK signal is input to the ISD-T360SA and controls the frequency of the codec interface operation. The CCLK may take on any frequency between 500 KHz and 2.048 MHz.

In master mode, the codec interface is configured to single bit rate clock. In slave mode, a double bit rate clock is available as well. When using double bit rate clock, the codec interface divides the frequency in half and the resulting clock is used as CCLK. Note that the signals are sampled on the falling edge of CCLK.

Short-frame / Long-frame Formats

The codec may be configured in one of the following formats:

- A short-frame format (see Figure 1-24)
- A long-frame format (see Figure 1-25)

In short-frame operation, CFS0 and CFS1 are active for one CCLK cycle. In long-frame operation, CFS0 and CFS1 are active for the width of the transaction.

Short-frame and long-frame formats are available in both master and slave modes (for more information refer to the software description on page 2-9).

Channel Width

The Codec interface supports both 8-bit and 16-bit channel width in both master and slave modes. The channel width determines how many data bits are transferred in a single codec transaction per channel. Refer to Table 1-7 for typical codec applications.

Table 1-7: Typical Codec Applications

Application	Codec Type	No. of Channels	Master/ Slave	Channel Width (No. Bits)	Long/ Short Frame Protocol	Bit Rate	CCLK Freq. (MHz)	Sample Rate (Hz)	No. of Frame Syncs
Analog μ-Law	single	1	Master	8	short or long	1	2.048	8000	1
ISDN—8 bit digital—A-Law	dual	2	Slave	8	short	1 or 2	2.048	8000	1
Linear	single	1	Master	16	short	1	2.048	8000	1
IOM-2/GCI	single or dual	1–2	Slave	8	short	1 or 2	1.536	8000	1
266 Compatibility	single or dual	1 or 2	Master	8	long or short	1	2.048	8000	1 or 2

Figure 1-8: Codec Protocol-Short Frame—8-Bit Channel Width

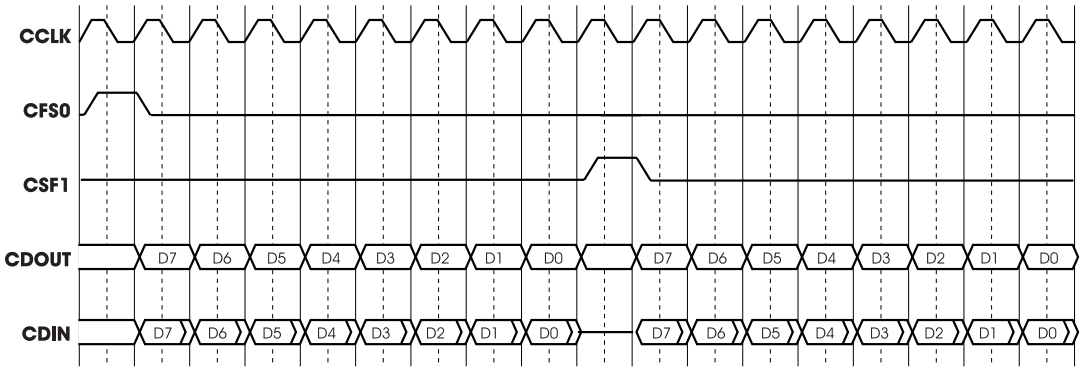


Figure 1-9: Codec Interface with One Single Codec, NSC TP3054, for Single Line Operation

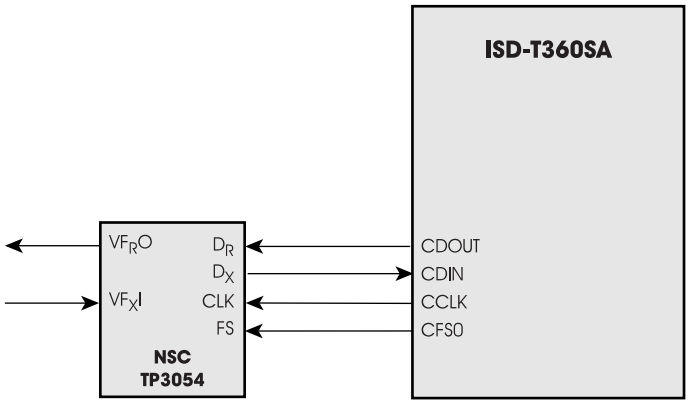


Figure 1-10: Codec Interface Diagram with Two, Single Codecs, NSC TP3054, and NSC TP3057, for Speakerphone Operation

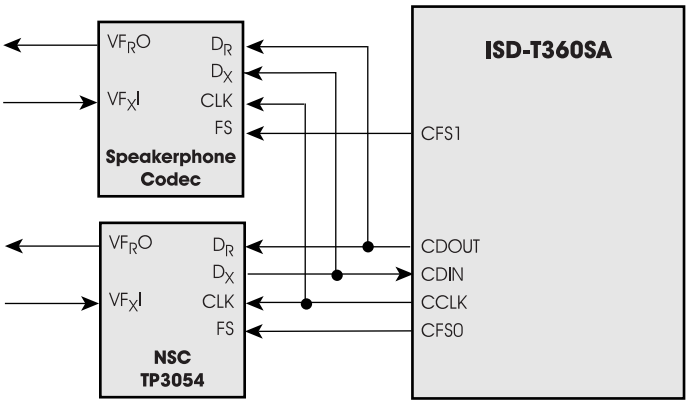


Figure 1-11: Codec Interface for Dual Line or Single Line and Speakerphone Operation with OKI Dual Codec

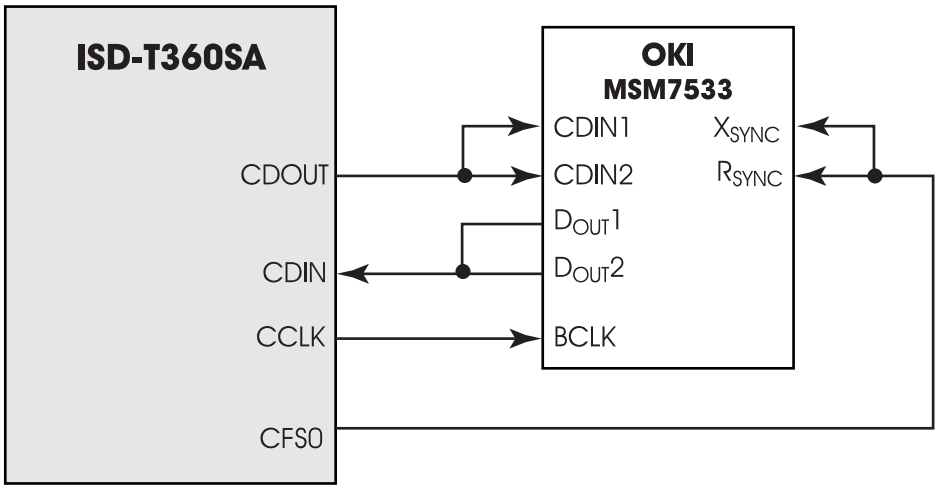


Figure 1-12: Codec Interface for Dual Line or Single Line and Speakerphone with Lucent Dual Codec

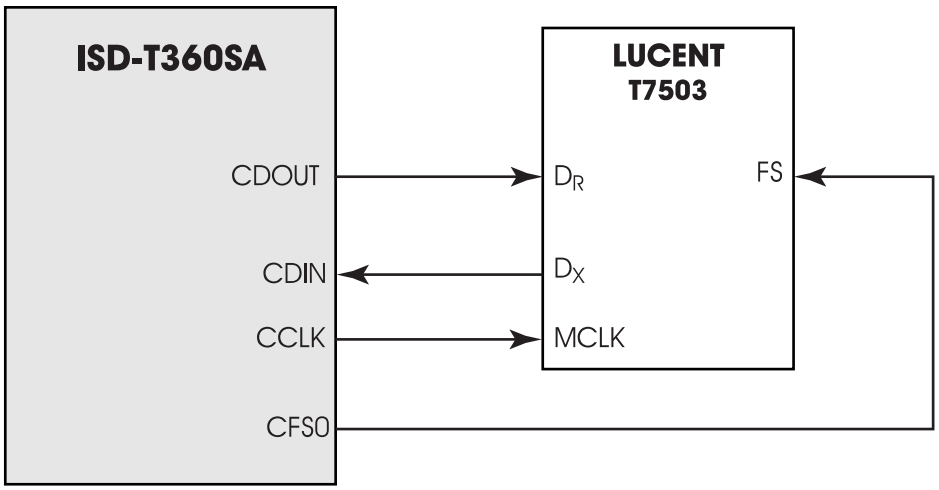


Figure 1-13: Codec Interface with one single Motorola codec, MC145482, for single line operation

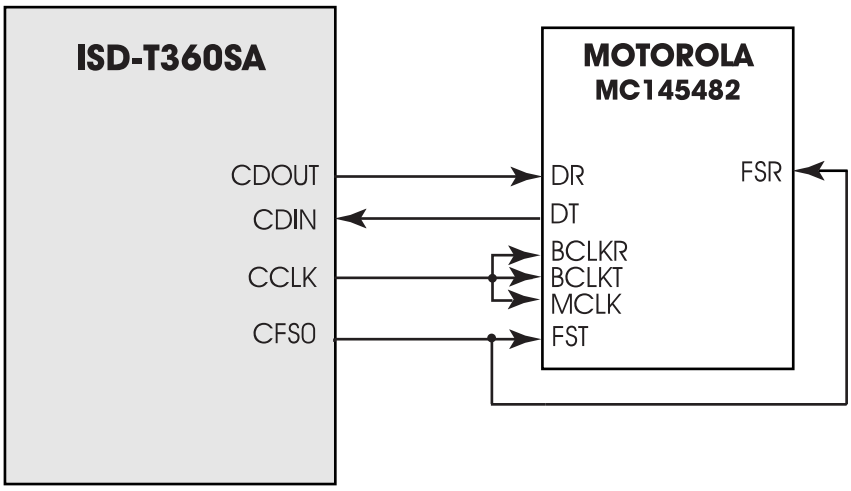


Figure 1-14: Codec Interface with two single Motorola codecs, MC145482, for speakerphone operation

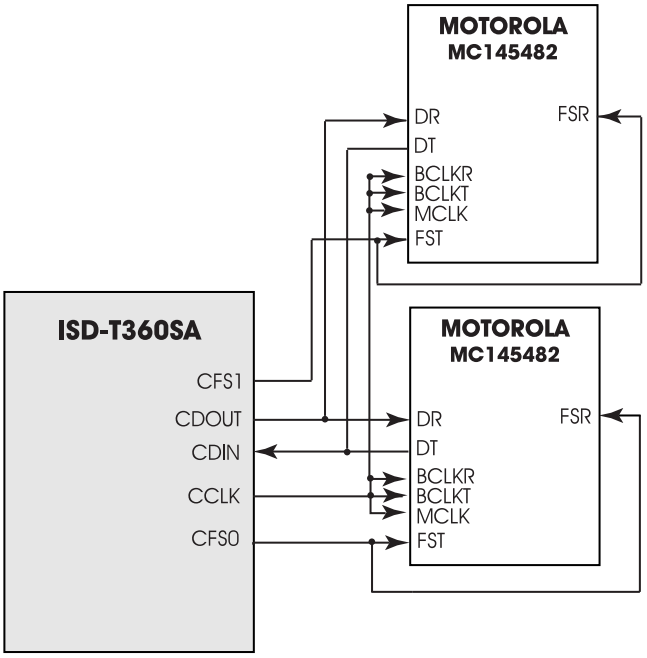
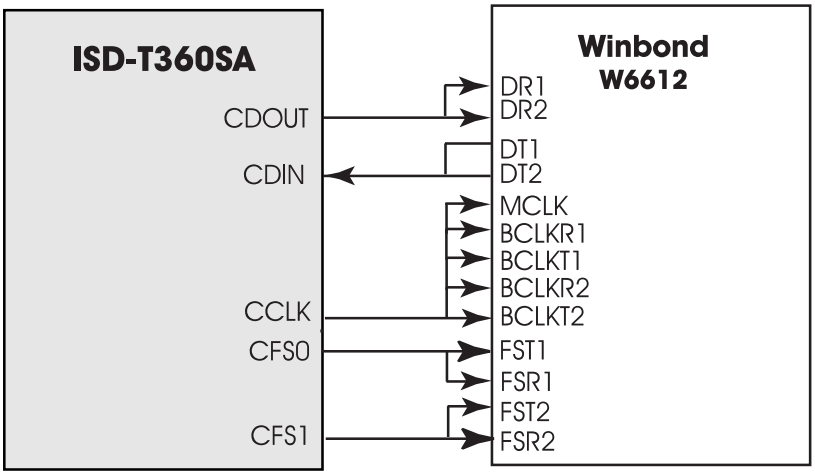


Figure 1-15: Codec Interface for Dual Line or Single Line and Speakerphone Operation with Winbond Dual Codec



NOTE: *To use this codec, you must define the codec interface to operate with two single codecs.*

1.3 SPECIFICATIONS

1.3.1 ABSOLUTE MAXIMUM RATINGS

Storage temperature	–65°C to +150°C
Temperature under bias	0°C to +70°C
All input and output voltages with respect to GND	–0.5 V to +6.5 V

NOTE *Absolute maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended; operation should be limited to the conditions specified below.*

1.3.2 ELECTRICAL CHARACTERISTICS

T_A = 0ºC to +70ºC, V_{CC} = 5 V ±10% Or 3.3 V ±10%, GND = 0 V

Table 1-8: Electrical Characteristics
(All Parameters with Reference to V_{CC} = 3.3 V)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
C_X	X1 and X2 Capacitance[1]			17.0		pF
I_{CC1}	Active Supply Current	Normal Operation Mode, Running Speech Applications[2]		40.0	80.0	mA
I_{CC2}	Standby supply current	Normal Operation Mode, DSPM Idle[2]		30.0		mA
I_{CC3}	Power-down Mode Supply Current	Power-down Mode[2,3]			12.0	mA
I_L	Input Load Current	0 V ≤ V_{IN} ≤ V_{CC}	–5.0[7]		5.0	µA
I_O (Off)	Output Leakage Current (I/O pins in Input Mode)	0 V ≤ V_{OUT} ≤ V_{CC}	–5.0[7]		5.0	µA
t_{WRa}	$\overline{WR0}$ Active	After R.E. CTTL, T1			t_{CTp}/ 2 + 2	
t_{WRCSh}	$\overline{WR0}$ Hold after $\overline{EMCS}$[4]	R.E. $\overline{EMCS}$ R.E. to R.E. $\overline{WR0}$	7.5			
t_{WRh}	$\overline{WR0}$ Hold	After R.E. CTTL	t_{CTp}/2 – 6			
t_{WRia}	$\overline{WR0}$ Inactive	After R.E. CTTL, T3			t_{CTp}/ 2 + 2	
V_{ENVh}	ENV0 Input, High Voltage		2.0			V
V_{Hh}	CMOS Input with Hysteresis, Logical 1 Input Voltage		2.1			V
V_{Hl}	CMOS Input with Hysteresis, Logical 0 Input Voltage				0.8	V
V_{Hys}	Hysteresis Loop Width[1]		0.5			V

Table 1-8: Electrical Characteristics
(All Parameters with Reference to V_{CC} = 3.3 V)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IH}	TTL Input, Logical 1 Input Voltage		2.0		V_{CC} + 0.5	V
V_{IL}	TTL Input, Logical 0 Input Voltage		–0.5[7]		0.8	V
V_{OH}	Logical 1 TTL, Output Voltage	I_{OH} = –0.4 mA	2.4			V
V_{OHWC}	MMCLK, MMDOUT and $\overline{EMCS}$ Logical 1, Output Voltage	I_{OH} = –0.4 mA	2.4			V
		I_{OH} = –50 µA[5]	V_{CC} – 0.2			V
V_{OL}	Logical 0, TTL Output Voltage	I_{OL} = 4 mA			0.45	V
		I_{OL} = 50 µA[5]			0.2	V
V_{OLWC}	MMCLK, MMDOUT and $\overline{EMCS}$ Logical 0, Output Voltage	I_{OL} = 4 mA			0.45	V
		I_{OL} = 50 µA[5]			0.2	V
V_{XH}	CLKIN Input, High Voltage	External Clock[6]	2.0			V
V_{XL}	CLKIN Input, Low Voltage	External Clock[6]			0.8	V

1. *Guaranteed by design.*
2. *I_{OUT} =0, T_A 25°C, V_{CC} = 3.3 V for V_{CC} pins and 3.3 V or 5 V on V_{CCHI} pins, operating from a 4.096 MHz crystal and running from internal memory with Expansion Memory disabled.*
3. *All input signals are tied to 0 (above V_{CC} – 0.5 V or below V_{SS} + 0.5 V), except ENV0, which is tied to V_{CC}.*
4. *Measured in power-down mode. The total current driven, or sourced, by all the VoiceDSP processor's output signals is less than 50 µA.*
5. *Guaranteed by design, but not fully tested.*
6. *CLKIN signal is not 5V tolerant.*
7. *Negative hold times are allowed since they are relative to the internal signal CTTL*

1.3.3 SWITCHING CHARACTERISTICS

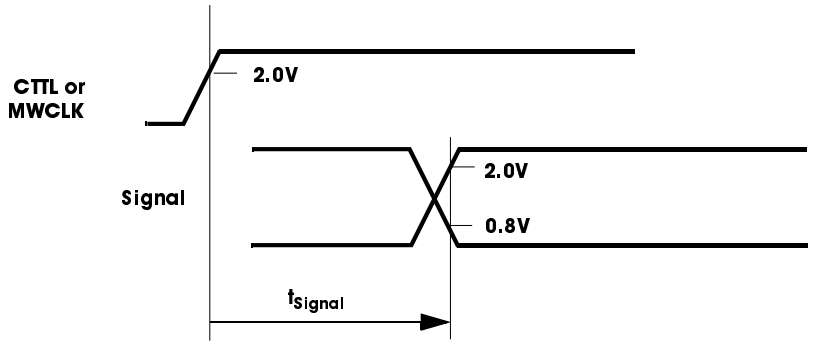
Definitions

All timing specifications in this section refer to 0.8 V or 2.0 V on the rising or falling edges of the signals, as illustrated in Figure 1-16 through Figure 1-22, unless specifically stated otherwise.

Maximum times assume capacitive loading of 50pF. CLKIN crystal frequency is 4.096 MHz.

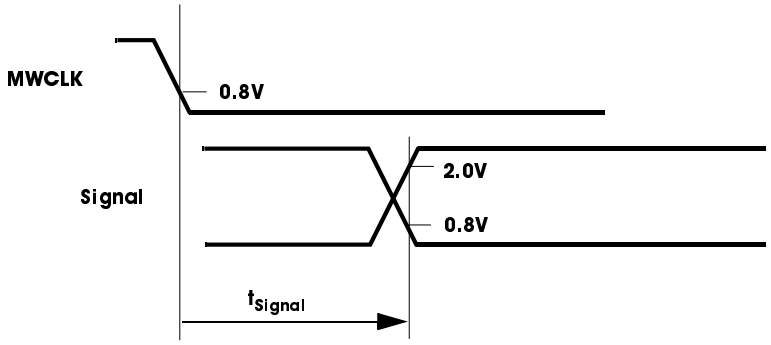
NOTE CTTL is an internal signal and is used as a reference to explain the timing of other signals. See Figure 1-31.

Figure 1-16: Synchronous Output Signals (Valid, Active and Inactive)



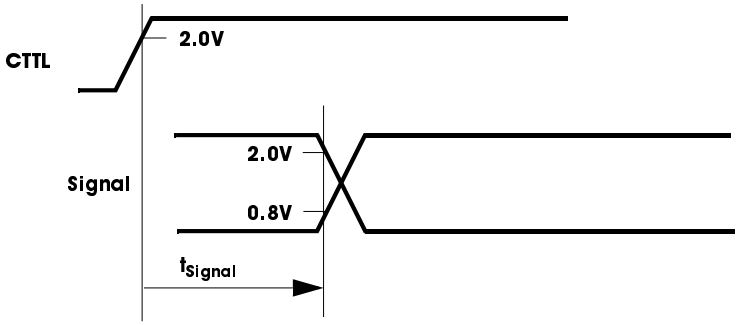
NOTE: *Signal valid, active or inactive time, after a rising edge of CTTL or MWCLK.*

Figure 1-17: Synchronous Output Signals (Valid)



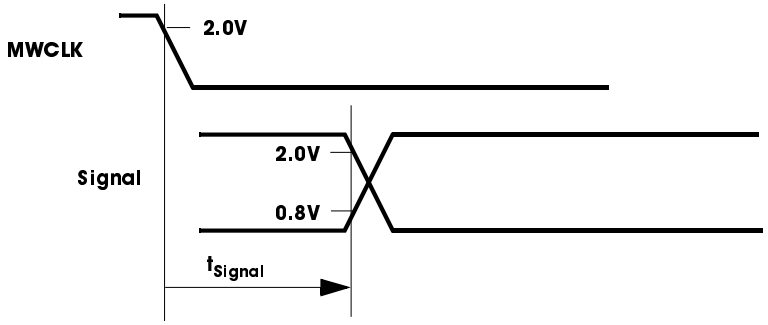
NOTE: *Signal valid time, after a falling edge of MWCLK.*

Figure 1-18: Synchronous Output Signals (Hold), after Rising Edge of CTTL



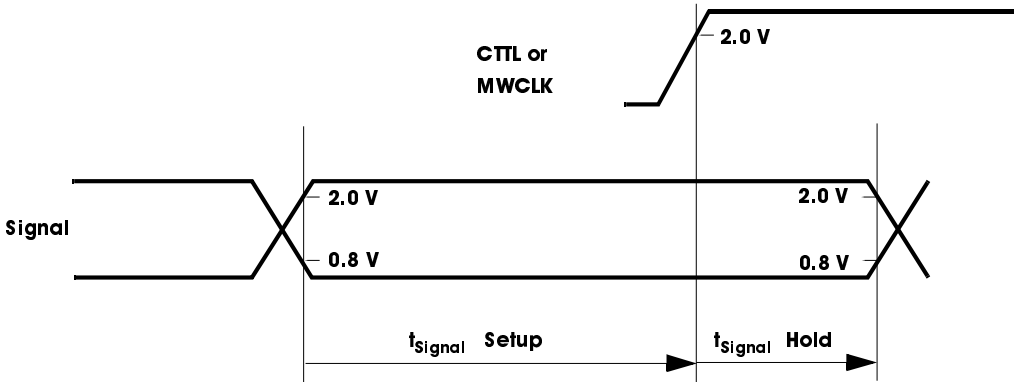
NOTE: *Signal hold time, after a rising edge of CTTL.*

Figure 1-19: Synchronous Output Signals (Hold), after Falling Edge of MWCLK



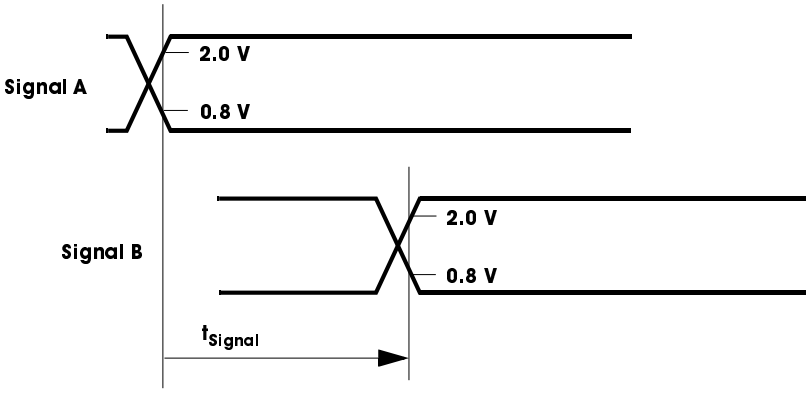
NOTE: *Signal hold time, after a falling edge of MWCLK.*

Figure 1-20: Synchronous Input Signals



NOTE: *Signal setup time, before a rising edge of CTTL or MWCK, and signal hold time after a rising edge of CTTL or MWCK*

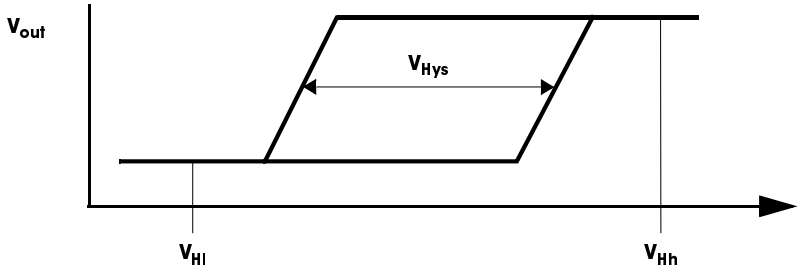
Figure 1-21: Asynchronous Signals



NOTE: *Signal B starts after rising or falling edge of signal A.*

The $\overline{\text{RESET}}$ has a Schmitt trigger input buffer. Figure 1-22 shows the input buffer characteristics.

Figure 1-22: Hysteresis Input Characteristics



1.3.4 SYNCHRONOUS TIMING TABLES

In this section, R.E. means Rising Edge and F.E. means Falling Edge.

Table 1-9: Output Signals

Symbol	Figure	Description	Reference Conditions	Min (ns)	Max (ns)
$t_{AIh\ [10:0]}$		Address Hold	After R.E. CTTL	-1.5[3]	
$t_{Ahh\ [15:11]}$		Address Hold	After R.E. CTTL	0.0	
t_{Av}		Address Valid	After R.E. CTTL, T1		9.0
t_{CCLKa}		CCLK Active	After R.E. CTTL		12.0
t_{CCLKh}		CCLK Hold	After R.E. CTTL	0.0	
t_{CCLKia}		CCLK Inactive	After R.E. CTTL		12.0
t_{CDOh}		CDOUT Hold	After R.E. CTTL	-2.0[3]	
t_{CDOv}		CDOUT Valid	After R.E. CTTL		12.0
t_{CTp}		CTTL Clock Period[1]	R.E. CTTL to next R.E. CTTL	30.5	250,000
t_{EMCSa}		$\overline{EMCS}$ Active	After R.E. CTTL, T2W1		12.0
t_{EMCSh}		$\overline{EMCS}$ Hold	After R.E. CTTL	0.0	
t_{EMCSia}		$\overline{EMCS}$ Inactive	After R.E. CTTL T3		12.0
t_{FSa}		CFS0 Active	After R.E. CTTL		25.0
t_{FSh}		CFS0 Hold	After R.E. CTTL	-2.5[3]	
t_{FSia}		CFS0 Inactive	After R.E. CTTL		25.0
t_{MWDOf}		MICROWIRE Data Float[1]	After R.E. $\overline{MWCS}$		70.0
t_{MWDOh}		MICROWIRE Data Out Hold[2]	After F.E. MWCLK	0.0	
t_{MWDOnf}		MICROWIRE Data No Float[2]	After F.E. $\overline{MWCS}$	0.0	70.0
t_{MWDOv}		MICROWIRE Data Out Valid[2]	After F.E. MWCLK		70.0
t_{MWITOp}		MWDIN to MWDOUT	Propagation Time		70.0
t_{MWRDYa}		$\overline{MWRDY}$ Active	After R.E. of CTTL	0.0	40.0
$t_{MWRDYia}$		$\overline{MWRDY}$ Inactive	After F.E. MWCLK	0.0	70.0
t_{PCh}		PC hold	After R.E. CTTL	-0.5[3]	
t_{PABh}		PA, PB and $\overline{MWRQST}$ hold	After R.E. CTTL, T2W1	-3.0[3]	
t_{PABCv}		PB and $\overline{MWRQST}$	After R.E. CTTL, T2W1		12.0

1. *In normal operation mode, t_{CTp} must be 30.5 ns; in power-down mode, t_{CTp} must be 50,000 ns.*
2. *Guaranteed by design, but not fully tested.*
3. *Negative hold times are allowed since they are relative to the internal signal CTTL.*

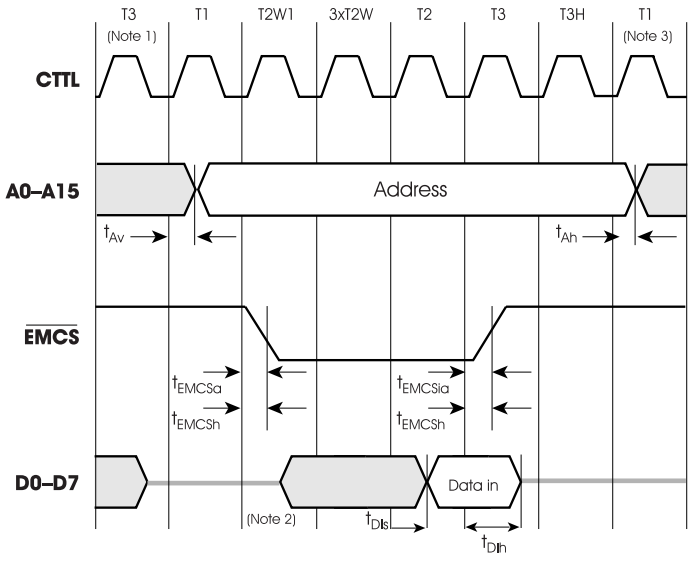
Table 1-10: Input Signals

Symbol	Figure	Description	Reference Conditions	Min (ns)	Max (ns)
t_{CCLKSp}		Codec Clock Period (slave)	R.E. CCLK to next R.E. CCLK	244	
t_{CCLKSh}		Codec Clock High (slave)	At 2.0 V (both edges)	120	
t_{CCLKSl}		Codec Clock Low (slave)	At 0.8 V (both edges)	120	
t_{CDIh}		CDIN Hold	After R.E. CTTL	0.0	
t_{CDIs}		CDIN Setup	Before R.E. CTTL	25.0	
t_{CFSOSs}		CFS0 Setup	Before R.E. CCLK	TBD	
t_{CFSOSh}		CFS0 Hold	After R.E. CCLK	TBD	
t_{DIh}		Data in Hold (D0:7)	After R.E. CTTL T1, T3 or TI	0.0	
t_{DIs}		Data in Setup (D0:7)	Before R.E. CTTL T1, T3 or TI	19.0	
t_{MWCKh}		MICROWIRE Clock High (slave)	At 2.0 V (both edges)	100.0	
t_{MWCKl}		MICROWIRE Clock Low (slave)	At 0.8 V (both edges)	100.0	
t_{MWCKp}		MICROWIRE Clock Period (slave)[1]	R.E. MWCLK to next R.E. MWCLK	2.5 μs	
t_{MWCLKh}		MWCLK Hold	After $\overline{\text{MWCS}}$ becomes inactive	50.0	
t_{MWCLKs}		MWCLK Setup	Before $\overline{\text{MWCS}}$ becomes active	100.0	
t_{MWCSh}		$\overline{\text{MWCS}}$ Hold	After F.E. MWCLK	75.0	
t_{MWCSs}		$\overline{\text{MWCS}}$ Setup	Before R.E. MWCLK	100.0	
t_{MWDIh}		MWDIN Hold	After R.E. MWCLK	50.0	
t_{MWDIs}		MWDIN Setup	Before R.E. MWCLK	100.0	
t_{PWR}		Power Stable to $\overline{\text{RESET}}$ R.E.[2]	After V_{CC} reaches 4.5 V	30.0 ms	
t_{RSTw}		$\overline{\text{RESET}}$ Pulse Width	At 0.8 V (both edges)	10.0 ms	
t_{Xh}		CLKIN High	At 2.0 V (both edges)	$t_{X1p}/2 - 5$	
t_{Xl}		CLKIN Low	At 0.8 V (both edges)	$t_{X1p}/2 - 5$	
t_{Xp}		CLKIN Clock Period	R.E. CLKIN to next R.E. CLKIN	244.4	

1. *Guaranteed by design, but not fully tested in power-down mode.*
2. *Guaranteed by design, but not fully tested.*

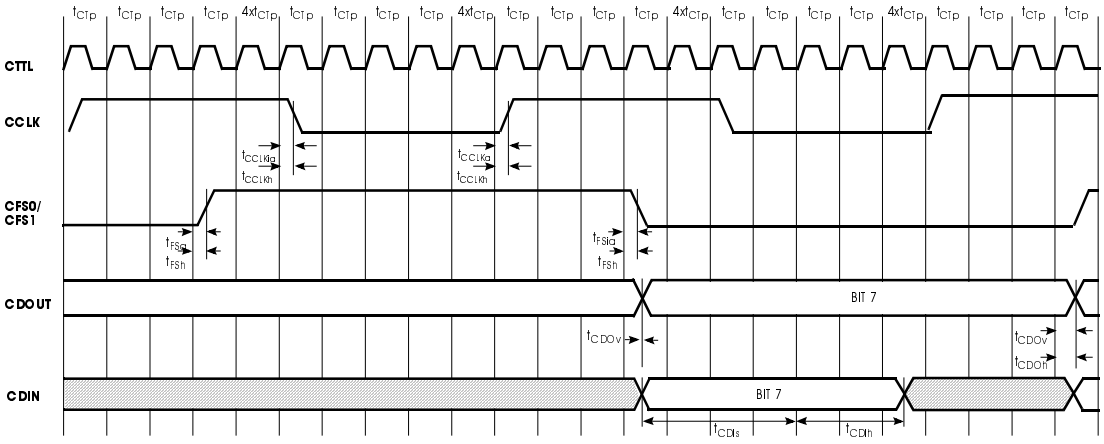
1.3.5 TIMING DIAGRAMS

Figure 1-23: ROM Read Cycle Timing



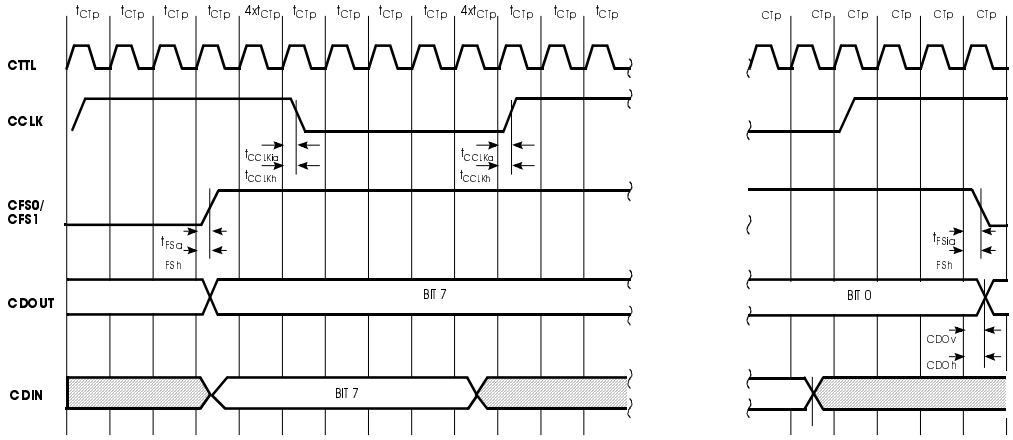
1. *This cycle may be either TI (Idle), T3 or T3H.*
2. *Data can be driven by an external device at T2W1, T2W, T2 and T3.*
3. *This cycle may be either TI (Idle) or T1.*

Figure 1-24: Codec Short Frame Timing



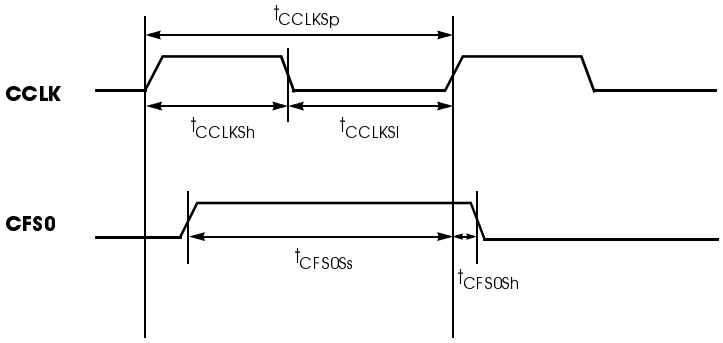
NOTE: *The CCLK and CFS0 timing is shown for Master Mode only. For Slave Mode, see Figure 1-26.*

Figure 1-25: Codec Long Frame Timing



NOTE: *The CCLK and CFS0 timing is shown for Master Mode only. For Slave Mode, see Figure 1-26.*

Figure 1-26: Slave Codec CCLK and CFS0 Timing



NOTE: *For CFS1, CDIN, CDOUT timing, see Figure 1-24 and Figure 1-25.*

Figure 1-27: MICROWIRE Transaction Timing—Data Transmitted to Output

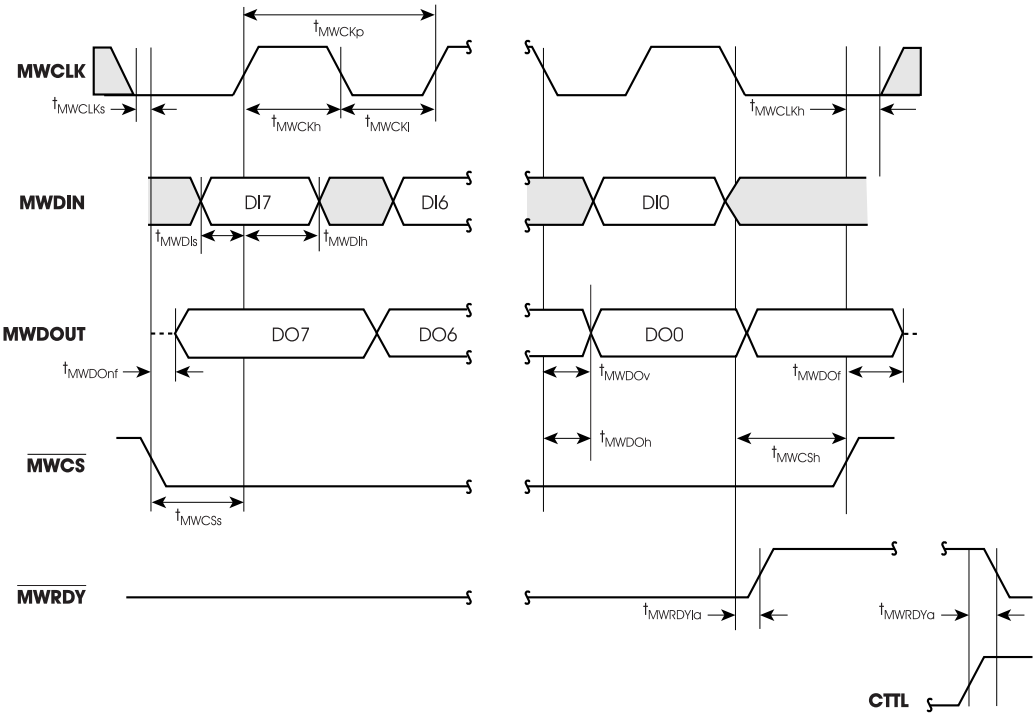


Figure 1-28: MICROWIRE Transaction Timing—Data Echoed to Output

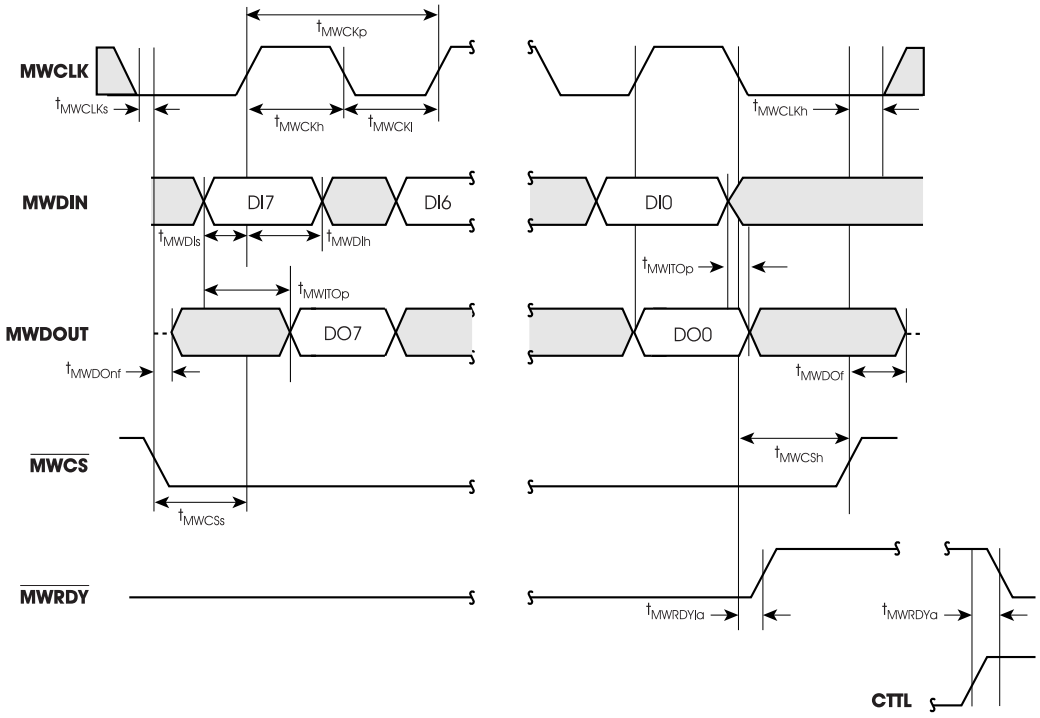
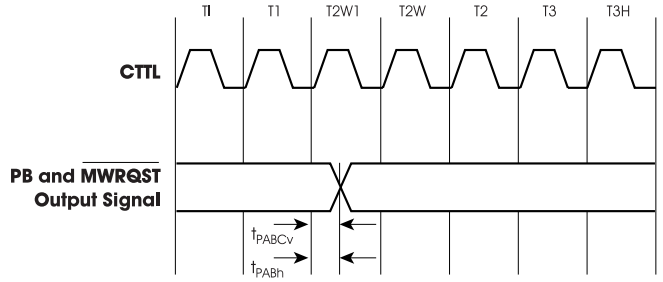


Figure 1-29: Output Signal Timing for Port PB and $\overline{\text{MWRQST}}$



NOTE: *This cycle may be either TI (Idle), T2, T3 or T3H.*

Figure 1-30: CLKIN Timing

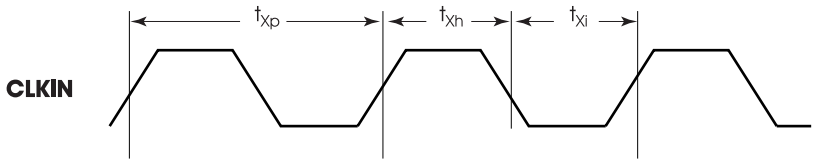


Figure 1-31: CTTL Timing

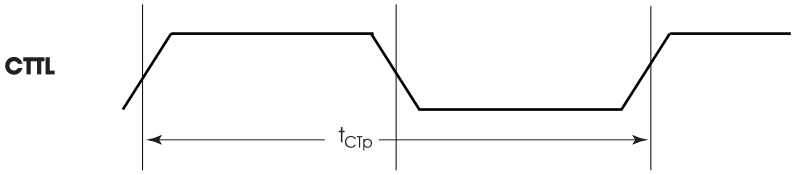


Figure 1-32: Reset Timing When Reset Is Not at Power-Up

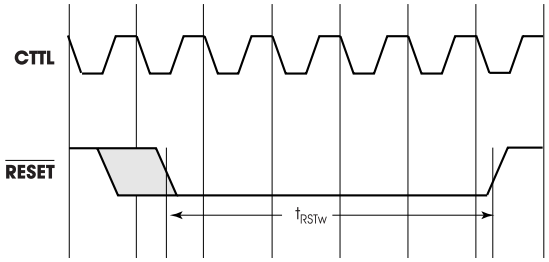
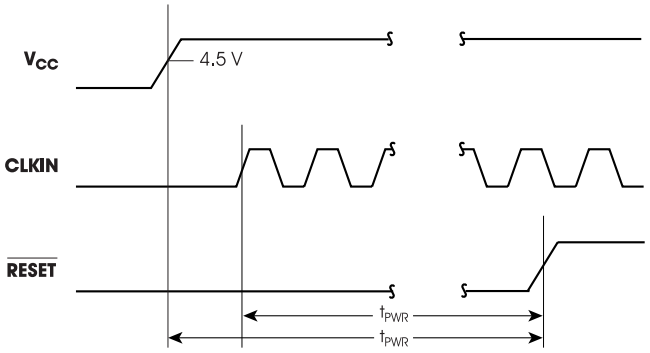


Figure 1-33: Reset Timing When Reset Is at Power-Up



Chapter 2—SOFTWARE

The VoiceDSP software resides in the on-chip ROM. It includes DSP-based algorithms, system support functions and a software interface to hardware peripherals.

2.1 SYSTEM OPERATION

This section provides details of the system support functions and their principle operation. It is divided into the following subjects:

- The State Machine
- Command Execution
- Event Handling
- Message Handling
- Tone Generation
- Initialization and Configuration
- Power Down Mode (PDM)

2.1.1 THE STATE MACHINE

The ISD-T360SA operates in two modes, normal mode (DTAD) and speakerphone mode. To change the mode use the Set Speakerphone Mode (SSM) command. The VoiceDSP processor functions as a state machine under each mode. It changes state either in response to a command sent by the microcontroller, after execution of the command is completed, or as a result of an internal event (e.g. memory full or power failure). For more information see "VoiceDSP PROCESSOR COMMANDS—QUICK REFERENCE TABLE" on page 2-28. The VoiceDSP states in DTAD mode are described below.

RESET

The VoiceDSP processor is initialized to the RESET state after a full hardware reset by the $\overline{\text{RESET}}$ signal (See "RESETTING" on page 1-3). In this state the processor detectors (VOX, constant energy, call progress tones and DTMF) are not active. In all other states, these detectors are active. (See the SDET and RDET commands for further details).

IDLE

This is the state from which most commands are executed. As soon as a command and all its parameters are received, the processor starts executing the command.

RECORD

In this state a message is compressed (unless stored in PCM format) and recorded into the message memory.

PLAY

In this state a message is decompressed (unless stored in PCM format), and played back.

SYNTHESIS

An individual word or a sentence is synthesized from an external vocabulary in this state.

TONE_GENERATE

In the TONE_GENERATE state, the VoiceDSP processor generates single or DTMF tones.

MSG_OPEN

The VoiceDSP processor either reads or writes 32 bytes to the message memory, or sets the message read/write pointer on a 32 byte boundary.

CID

In this state, the VoiceDSP processor receives the Caller ID data into an internal buffer.

CIDCW

In this state, the VoiceDSP processor starts the CIDCW process by sending an ACK signal to the central office, and then changes to CID state.

POWER DOWN MODE

In this state, the power consumption is reduced for the VoiceDSP processor. For more information, refer to the Power Down Mode section on page 2-5.

2.1.2 COMMAND EXECUTION

A VoiceDSP processor command is represented by an 8-bit opcode. Some commands require parameters. Some commands have return values. Commands are either synchronous or asynchronous.

SYNCHRONOUS COMMANDS

A synchronous command must complete execution before the microcontroller can send a new command (e.g. GMS, GEW). A command sequence begins when the microcontroller sends an 8-bit opcode to the processor, followed by the command's parameters (if any). The VoiceDSP processor then executes the command and, if required, transmits a return value to the microcontroller. Upon completion, the processor notifies the microcontroller that it is ready to accept a new command.

ASYNCHRONOUS COMMANDS

An asynchronous command starts execution in the background and notifies the microcontroller, which can send more commands while the current command is still running (e.g. R, P). After receiving an asynchronous command, such as P (Playback), R (Record), SW (Say Words) or GT (Generate Tone), the VoiceDSP processor switches to the appropriate state and executes the command until finished or a S (Stop) or PA (Pause) command is received from the microcontroller. When completed, the EV_NORMAL _END event is set and the processor switches to the IDLE state.

"VoiceDSP PROCESSOR COMMANDS—QUICK REFERENCE TABLE" on page 2–28 displays all the processor commands, the valid source states in which these commands are valid, and the states resulting from the command.

2.1.3 EVENT HANDLING

STATUS WORD

The 16-bit status word indicates events that occur during normal operation. The VoiceDSP processor activates the $\overline{\text{MWRQST}}$ signal, to indicate a change in the status word. This signal remains active until the processor receives a GSW (Get Status Word) command.

For detailed description of the Status Word and the meaning of each bit, see "GSW Get Status Word" on page 2-46.

ERROR WORD

The 16-bit error word indicates errors that occurred during execution of the last command. If an error is detected, the command is not processed; the EV_ERROR bit in the status word is set to 1, and the $\overline{\text{MWRQST}}$ signal is activated.

ERROR HANDLING

When the microcontroller detects the active $\overline{\text{MWRQST}}$ signal, it issues the GSW command, deactivating the $\overline{\text{MWRQST}}$ signal. Then, the microcontroller tests the EV_ERROR bit in the status word, and, if set, sends the GEW (Get Error Word) command to read the error word for details.

For detailed description of the Error Word and the meaning of each bit, see "GEW Get Error Word" on page 2-43.

2.1.4 MESSAGE HANDLING

A message is the basic unit on which most of the VoiceDSP commands operate. A VoiceDSP processor message, stored on a flash memory device, can be regarded as a computer file stored on a flash mass-storage device.

The ISD-T360SA manages messages for a wide range of applications, which require different levels of DTAD functionality. The VoiceDSP processor features advanced memory-organization features such as multiple OutGoing Messages (OGMs), mailboxes, and the ability to distinguish between InComing Messages (ICMs) and OGMs.

A message is created with either the R (Record) or the CMSG (Create Message) command. Once created, the message is assigned a time-and-day stamp and a message tag which is read by the microcontroller. Caller ID information can be automatically attached to the message by setting the CID_RECORD tunable parameter. (For more information about the tunable parameter, refer to Table 2-13.) The R command takes voice samples from the codec, compresses them, and stores them in the message memory.

When a message is created with the CMSG command the data to be recorded is provided by the microcontroller, via the WMSG (Write Message) command and not through the codec. Here, the data is transferred directly to the message memory, and not compressed by the ISD-T360SA voice compression algorithm.

WMSG, RMSG (Read Message) and SMSG (Set Message Pointer) are message-data access commands used to store and read data to or from any location in the message memory (see "VoiceDSP PROCESSOR COMMANDS—QUICK REFERENCE TABLE" on page 2-28 for more details). Using these commands, the microcontroller can utilize messages for features such as a Telephone Directory and Caller Numbers List (Caller IDs of those who called but did not leave a message.)

A message can be played back (P command) and deleted (DM command). Redundant data (e.g. trailing tones or silence) can be removed from the message tail with the CMT (Cut Message Tail) command.

The PA (Pause) and RES (Resume) commands, respectively, suspend the P (Playback) and R (Record) commands, and resume them from the point at which they were suspended.

CURRENT MESSAGE

The GTM (Get Tagged Message) command selects the current message. Most message handling commands (P, DM, RMSG) operate on the current message.

Deleting the current message does not cause a different message to become current; the current message is undefined. If you issue the GTM command to skip to the next message, the first message, newer than the just deleted message, becomes the current message.

MESSAGE TAG

Each message has a 2-byte message tag which is used to categorize messages, and implement features such as OutGoing Messages, mailboxes and different handling of old and new messages.

The tag is created during the R (Record) command. Use the GMT (Get Message Tag) and SMT (Set Message Tag) commands to handle message tags.

NOTE *Message tag bits can only be cleared and are set only when a message is first created. This limitation, inherent in Flash memories, allows bits to be changed only from 1 to 0 (changing bits from 0 to 1 requires a special erasure procedure). However, the usual reason for updating an existing tag is to mark a message as old. This can be done when a message is first created by using one of the bits as a new/old indicator, setting the bit to 1 and later clearing it when necessary.*

2.1.5 TONE GENERATION

The VoiceDSP processor generates DTMF tones and single-frequency tones from 300Hz to 3000Hz in increments of 100Hz. The ISD-T360SA tone generation conforms to the EIA-470-RS standard. Note, however, that value of some tunable parameters may need adjusting to meet the standard specifications since the energy level of generated tones depends on the analog circuits used.

1. Tune the DTMF_GEN_TWIST_LEVEL parameter to control the twist level of the generated DTMF tones.
2. Use the VC (Volume Control) command, and tune the TONE_GEN_LEVEL parameter, to control the energy level at which these tones are generated.
3. Use the GT (Generate Tone) command to specify the DTMF tones, and the frequency at which single tones are generated.

Refer to Table 2-5, VC command and GT command of the Command Description section for further details of the relevant tunable parameters and commands.

NOTE The DTMF detector performance is degraded during tone generation, especially if the frequency of the generated tone is close to the frequency of one of the DTMF tones.

2.1.6 INITIALIZATION AND CONFIGURATION

Use the following procedures to initialize the VoiceDSP processor:

NORMAL INITIALIZATION

Reset the VoiceDSP processor by activating the RESET signal. (See "RESETTING" on page 1-3.)

1. Issue a CFG (Configure VoiceDSP processor) command to change the configuration according to your environment.
2. Issue an INIT (Initialize System) command to initialize the VoiceDSP firmware.
3. Issue a series of TUNE commands to adjust the VoiceDSP processor to the requirements of your system.

NOTE The tuning of NUM_OF_BLOCKS_IN_MEM (tunable parameter number 62) should be done before the CFG command. This tune operation should be executed when using flash memory other than 4Mb. The tuning of NUM_OF_BLOCKS_FOR_TEST (tunable parameter number 63) should be done before the INIT command.

TUNABLE PARAMETERS

The VoiceDSP processor can be adjusted to the specific system's requirements using a set of tunable parameters. These parameters are set to their default values after reset and can be later modified with the TUNE command. By tuning these parameters, you can control various aspects of the VoiceDSP processor's operation, such as silence compression, tone detection, and no-energy detection.

Table 2-4 to Table 2-13 of the Command Description section describes all the tunable parameters in detail.

2.1.7 POWER-DOWN MODE

The PDM (Go To Power-Down Mode) command switches the ISD-T360SA to power-down mode. The purpose of the PDM command is to save power during battery operation, or for any other power saving cause. During power-down mode only basic functions, such as time and date update, are active (for more details refer to POWER-DOWN MODE description on page 1-4).

This PDM command may only be issued when the processor is in IDLE mode (for an explanation of the ISD-T360SA states, see "The State Machine" on page 2-1). If it is necessary to switch to power-down mode from any other state, the controller must first issue a S (Stop) command to switch the processor to the IDLE state, and then issue the PDM command. Sending any command while in power-down mode resets the VoiceDSP processor detectors, and returns it to normal operation mode.

NOTE *Entering or exiting power-down mode can distort the real-time clock by up to 500 µs. Thus, to maintain the accuracy of the real-time clock, enter or exit the power-down mode as infrequently as possible.*

2.2 PERIPHERALS

This section provides details of the peripherals interface support functions and their principle operation. It is divided into the following subjects:

- Microcontroller Interface (Slave MICROWIRE)
- Memory Interface
- Codec Interface

2.2.1 MICROCONTROLLER INTERFACE

MICROWIRE/PLUS™ is a synchronous serial communication protocol minimizes the number of connections, and thus the cost, of communicating with peripherals.

The VoiceDSP MICROWIRE interface implements the MICROWIRE/PLUS interface in slave mode, with an additional ready signal. It enables a microcontroller to interface efficiently with the VoiceDSP processor application.

The microcontroller is the protocol master and provides the clock for the protocol. The VoiceDSP processor supports clock rates of up to 400 KHz. This transfer rate refers to the bit transfer; the actual throughput is slower due to byte processing by the VoiceDSP processor and the microcontroller.

Communication is handled in bursts of eight bits (one byte). In each burst the VoiceDSP processor is able to receive and transmit eight bits of data. After eight bits have been transferred, an internal interrupt is issued for the VoiceDSP processor to process the byte, or to prepare another byte for sending. In parallel, the VoiceDSP processor sets $\overline{\text{MWRDY}}$ to 1, to signal the microcontroller that it is busy with the byte processing. Another byte can be transferred only when the $\overline{\text{MWRDY}}$ signal is cleared to 0 by the VoiceDSP processor. When the VoiceDSP processor transmits data, it expects to receive the value 0xAA before each transmitted byte. The VoiceDSP processor reports any status change by clearing the $\overline{\text{MWRQST}}$ signal to 0.

If processor command's parameter is larger than one byte, the microcontroller transmits the Most Significant Byte (MSB) first. If a return value is larger than one byte, the VoiceDSP processor transmits the MSB first.

The following signals are used for the interface protocol. Input and output are relative to the VoiceDSP processor.

INPUT SIGNALS

MWDIN

MICROWIRE Data In. Used for input only, for transferring data from the microcontroller to the VoiceDSP processor.

MWCLK

MICROWIRE Clock. Serves as the synchronization clock during communication. One bit of data is transferred on every clock cycle. The input data is available on MWDIN and is latched on the clock rising edge. The transmitted data is output on MWDOUT on the clock falling edge. The signal should remain low when switching $\overline{\text{MWCS}}$.

$\overline{\text{MWCS}}$

MICROWIRE Chip Select. The $\overline{\text{MWCS}}$ signal is cleared to 0, to indicate that the VoiceDSP processor is being accessed. Setting $\overline{\text{MWCS}}$ to 1 causes the VoiceDSP processor to start driving MWDOUT with bit 7 of the transmitted value. Setting the $\overline{\text{MWCS}}$ signal resets the transfer-bit counter of the protocol, so the signal can be used to synchronize between the VoiceDSP processor and the microcontroller.

To prevent false detection of access to the VoiceDSP processor due to spikes on the MWCLK signal, use this chip select signal, and toggle the MWCLK input signal, only when the VoiceDSP processor is accessed.

OUTPUT SIGNALS

MWDOUT

MICROWIRE Data Out. Used for output only, for transferring data from the VoiceDSP processor to the microcontroller. When the VoiceDSP processor receives data it is echoed back to the microcontroller on this signal, unless the received data is 0xAA. In this case, the VoiceDSP processor echoes a command's return value.

$\overline{\text{MWRDY}}$

MICROWIRE Ready. When active (0), this signal indicates that the VoiceDSP processor is ready to transfer (receive or transmit) another byte of data.

This signal is set to 1 by the VoiceDSP processor after each byte transfer has been completed. It remains 1, while the VoiceDSP processor is busy reading the byte, writing the next byte, or executing the received command (after the last parameter has been received). $\overline{\text{MWRDY}}$ is cleared to 0 after reset. For proper operation after a hardware reset, this signal should be pulled up.

$\overline{\text{MWRQST}}$

MICROWIRE Request. When active (0), this signal indicates that new status information is available. $\overline{\text{MWRQST}}$ is deactivated (set to 1), after the VoiceDSP processor receives a GSW (Get Status Word) command from the microcontroller. After reset, this signal is active (0) to indicate that a reset occurred. $\overline{\text{MWRQST}}$, unlike all the signals of the communication protocol, is an asynchronous line that is controlled by the VoiceDSP firmware.

SIGNAL USE IN THE INTERFACE PROTOCOL

After reset, both $\overline{\text{MWRQST}}$ and $\overline{\text{MWRDY}}$ are cleared to 0.

The $\overline{\text{MWRQST}}$ signal is activated to indicate that a reset occurred. The EV_RESET bit in the status register is used to indicate a reset condition.

The GSW command should be issued after reset to verify that the EV_RESET event occurred, and to deactivate the $\overline{\text{MWRQST}}$ signal.

While the $\overline{\text{MWCS}}$ signal is active (0), the VoiceDSP processor reads data from MWDIN on every rising edge of MWCLK. VoiceDSP processor also writes every bit back to MWDOUT. This bit is either the same bit which was read from MWDIN (in this case it is written back as a synchronization echo after some propagation delay), or it is a bit of a value the VoiceDSP processor transmits to the microcontroller (in this case it is written on every falling edge of the clock).

When a command has more than one parameter/return-value, the parameters/return-values are transmitted in the order of appearance. If a parameter/return-value is more than one byte long, the bytes are transmitted from the most significant to the least significant.

The $\overline{\text{MWRDY}}$ signal is used as follows:

1. Active (0) $\overline{\text{MWRDY}}$ signals the microcontroller that the last eight bits of data transferred to/from the voice module were accepted and processed (see below).
2. The $\overline{\text{MWRDY}}$ signal is deactivated (set to 1 by the VoiceDSP processor) after 8-bits of data were transferred to/from the VoiceDSP processor. The bit is set following the falling edge of the eighth MWCLK clock-cycle.
3. The $\overline{\text{MWRDY}}$ signal is activated (cleared to 0) by the VoiceDSP processor when it is ready to receive the first parameter byte (if there are any parameters) and so on till the last byte of parameters is transferred. An active $\overline{\text{MWRDY}}$ signal after the last byte of parameters indicates that the command was parsed and (if possible) executed. If that command has a return value, the microcontroller must read the value before issuing a new command.
4. When a return value is transmitted, the $\overline{\text{MWRDY}}$ signal is deactivated after every byte, and activated again when the VoiceDSP processor is ready to send another byte, or to receive a new command.
5. The $\overline{\text{MWRDY}}$ signal is activated (cleared to 0) after reset, and after a protocol time-out. (See "INTERFACE PROTOCOL TIME-OUTS")

The $\overline{\text{MWRQST}}$ signal is used as follows:

1. The $\overline{\text{MWRQST}}$ signal is activated (cleared to 0), when the status word is changed.
2. The $\overline{\text{MWRQST}}$ signal remains active (0), until the VoiceDSP processor receives a GSW command.

Figure 1-27 and Figure 1-28 illustrate the sequence of activities during a MICROWIRE data transfer between VoiceDSP and the microcontroller.

INTERFACE PROTOCOL TIME-OUTS

Depending on the VoiceDSP processor's state, if more than 100 milliseconds elapse between the assertion of the $\overline{\text{MWRDY}}$ signal and the transmission 8th bit of the next byte pertaining to the same command transaction, a time-out event occurs, and the VoiceDSP processor responds as follows:

1. Sets the error bit in the status word to 1.
2. Sets the EV_TIMEOUT bit in the error word to 1.
3. Activates the $\overline{\text{MWRQST}}$ signal (clears it to 0).
4. Activates the $\overline{\text{MWRDY}}$ signal (clears it to 0).
5. Waits for a new command. (After a time-out occurs, i.e., the microcontroller received $\overline{\text{MWRQST}}$ during the command transfer, or result reception, the microcontroller must wait at least four milliseconds before issuing the next command.)

ECHO MECHANISM

The VoiceDSP processor echoes back to the microcontroller all the bits received by the VoiceDSP processor. Upon detection of an error in the echo, the microcontroller should stop the protocol clock, which eventually causes a time-out error (i.e., ERR_TIMEOUT bit is set in the error word).

For commands that have a return value, the microcontroller must transmit a byte value of 0xAA for each byte returned. In response, the VoiceDSP processor transmits the return values instead of echoing the 0xAA byte.

Upon detection of an error the VoiceDSP processor activates the $\overline{\text{MWRQST}}$ signal, and sets the ERR_COMM bit in the error word.

2.2.2 MEMORY INTERFACE

DEVICE NUMBER AND TYPE

The VoiceDSP processor supports various types of Flash memory devices. Up to four devices may be connected to the VoiceDSP, where all the connected devices must be of the same type. Each memory device may be of 4Mbit, 8Mbit or 16Mbit; thus a total of 64Mbit non-volatile memory may be connected for message storage (up to 4 hours of voice recording).

See "MEMORY INTERFACE" on page 1-6, for detailed description of the supported Flash and the hardware connectivity.

Use the CFG command to define the type and number of installed memory devices (see "CFG Configure VoiceDSP config_value" on page 2-32).

MEMORY DEVICE SIZE

The memory manager handles the memory devices in basic units of 4Kbyts blocks. This approach is defined due to the nature of Flash devices where the basic unit that can be erased is a 4Kbytes block.

Memory blocks cannot be shared by different voice messages. Therefore, the maximum number of messages per memory device, equals to the number of memory blocks minus one (one block per device is used for memory management).

The size of the connected memory devices, is defined by the number of memory blocks in each device. Refer to tunable parameter index 62, in Table 2-10, for detailed description of the available number of blocks for Flash.

PRODUCTION LINE TESTING

In many cases it is desired to test the ISD-T360SA in the production line as part of the whole application. Usually in these cases, the testing time is an important factor and should be minimized as possible. The initialization time of the memory devices is significant and should be avoided during production (Refer to Table 1-4). Therefore, a dedicated parameter is defined in order to allow a production line testing while using a small part of the real connected memory size.

It should be noted that in case of power failure during the production line testing, the connected memory devices should be replaced, and the process should be repeated. Refer to parameter index 63, in Table 2-10, for further explanation of the production line testing.

2.2.3 CODEC INTERFACE

SUPPORTED FUNCTIONALITY

The VoiceDSP processor supports analog and digital telephony in various configurations. For analog telephony the VoiceDSP operates in master mode, where it provides the clock and the synchronization signals. It supports a list of single channel and dual channel codecs, as listed in Table 1-6. For digital telephony the VoiceDSP operates in slave mode, where the control signals are provided by an external source.

The codec interface is designed to exchange data in short frame format as well as in long frame format. The channel width may be either 8 bits (u-Law format or A-Law format), or 16 bits (linear format). In slave mode the clock may be divided by two, if required (two bit rate clock mode).

The VoiceDSP processor supports up to 2 voice channels, where the line should be connected as channel 0 (in master mode or in slave mode - depends on the configuration), and the speakerphone (speaker and microphone) should be connected as channel 1 or as channel 2, depends on the configuration (channel 2 is always connected as master).

See "The Codec Interface" on page 1-9, for detailed description of the supported codec devices and the hardware connectivity.

Use the CFG command to define the codec mode (master or slave), the data frame format (short or long), the channel width (8 bits or 16 bits), the clock bit rate (single or dual) and the number and type of codecs (one or two, single channel or dual channel). See "CFG Configure VoiceDSP config_value" on page 2-32.

DATA CHANNEL TIMING

To provide additional flexibility (e.g., when using speakerphone), the codec interface provides a programmable delay of data and synchronization signals, relative to the CFS0 signal.

Refer to tunable parameters index 65 to index 69, in Table 2-11, for detailed description of the delay registers and their significance.

2.3 ALGORITHM FEATURES

This section provides details of the VoiceDSP algorithms and their principle operation. It is divided into the following subjects:

- VCD (Voice Compression and Decompression)
- DTMF Detection
- Tone and Energy Detection (Call Progress)
- Speakerphone
- Speech Synthesis
- Caller ID
- SW Automatic Gain Control

2.3.1 VCD (VOICE COMPRESSION AND DECOMPRESSION)

The VoiceDSP processor implements a state of the art VCD algorithm of the CELP family. The algorithm provides 3 compression rates that can be selected dynamically. PCM recording (no compression) is also provided.

The lowest compression rate of 4.7 Kbit/s enables about 30 minutes of recording on an 8-Mbit device. The mid compression rate of 6.7 Kbit/s provides about 20 minutes of voice recording time. The highest compression rate of 8.7 Kbit/s, which provides the best compressed voice quality, stores approximately 10 minutes on an 8-Mbit device. For detailed information about recording times refer to Table 1-5.

Before recording each message, the microcontroller selects one of the three compression rates, or PCM recording, with the compression_rate parameter of the R (Record) command. During message playback the VoiceDSP processor reads this one byte parameter and selects the appropriate speech decompression algorithm.

IVS vocabularies can be prepared in either of the three compression rates, or in PCM format, using the IVS tool. All messages in a single vocabulary must be recorded using the same algorithm. (See the *IVS User's Guide* for more details). During speech synthesis, the VoiceDSP processor automatically selects the appropriate speech decompression algorithm.

VARIABLE SPEED PLAYBACK

This feature increases or decreases the speed of messages and synthesized messages during playback. Use the SPS (Set Playback Speed) to set the speed of message playback. The new speed applies to all recorded messages and synthesized messages (only if synthesized using IVS), until changed by another SPS command. If this command is issued while the VoiceDSP processor is in the PLAY state, the speed also changes for the message currently being played.

The speedup / slowdown algorithm is designed to maintain the pitch of the original speech. This approach provides the same speech tone while playback speed varies.

CONVERSATION RECORDING / PLAYBACK

The VoiceDSP supports the conversation recording feature, which enables the user to simultaneously record the far end caller and near end caller. The VoiceDSP simultaneously records data from channels 0 and 1 by summing the two channels and then compressing the summation. The data is then saved in the flash memory. When Playing the compressed message, the data is simultaneously broadcast to both channels. This feature is available for all compression rates. This feature can be operated using the SLC command. For more details, refer to the SLC command in the Command Description section.

PCM RECORDING

The VoiceDSP is capable of recording data in PCM format (that is the original samples format either in 8 bit u-Law format, 8 bit A-law format or 16 bit linear format). The PCM data uses more storage space, but it provides the highest quality for OGM, music-on-hold or IVS data. The PCM recording may be selected as one of the available compression rates during the R command (compression_rate = 0).

Variable Speed Playback is not accessible during PCM recording and playback.

2.3.2 DTMF DETECTION

The VoiceDSP processor detects DTMF tones, thus enabling remote control operations. Detection is active throughout the operation of the VoiceDSP processor. Detection can be configured using the SDET (Set Detectors Mask) command, which controls the reporting of the occurrence of tones, and the RDET (Reset Detectors) command which resets the detectors. The accuracy of the tone length, as reported by the tone detectors, is ±10 ms.

DTMF detection may be reported at the starting point, ending point, or both. The report is made through the status word (for further details, see GSW command).

For further details about tunable parameters refer to Table 2-6 of the Command description section. The DTMF detector performance, as measured on the line input using an ISD-DS360-DAA board, is summarized below in Table 2-1.

Table 2-1: DTMF Detector Performance*

	Play/IVS Synthesis	Record/Idle
Detection Sensitivity[1]	Performance depends on the message being played.[2]	–34 dBm
Accepted DTMF Length[3]	>50 ms	>40 ms
Frequency Tolerance	±1.5%	±1.5%
S/N Ratio	12 dB	12 dB
Minimum Spacing[4]	>50 ms	>45 ms
Normal Twist	8 dB	8 dB
Reverse Twist[5]	4 dB or 8 dB	4 dB or 8 dB

* Performance depends on the DAA design. For reliable DTMF detection:
- A hardware echo canceller, that attenuates the echo by at least 6 dBm, is required during playback.
- The HW AGC, if present, must be disabled during playback.

1. *Tune parameters 60 and 61 affect DTMF detection sensitivity. The detection sensitivity is -34 dBm when these tunable parameters get their maximum value of 5.*
2. *Performance with echo canceler is 10 dB better than without echo canceler. For a silent message, detection sensitivity is –34 dBm, with echo canceler.*
3. *The accuracy of reported DTMF tones is ±10 ms.*
4. *If the interval between two consecutive identical DTMF tones is less than, or equal to, 20 ms, the two are detected as one long DTMF tone. If the interval between two consecutive identical DTMF tones is between 20 ms and 45 ms, separate detection is unpredictable.*
5. *Determined by the DTMF_REV_TWIST tunable parameter value.*

DTMF SW AGC

In order to remove the linkage between the HW AGC and the detection level of the DTMF detector, two new tunable parameters are added. These tunable parameters define the gain of the SW AGC for DTMF signals.

DTMF_DET_AGC_IDLE - SW AGC for DTMF detection in Idle and Record states. When incrementing this tunable by 1, the dynamic range is increased by 3 dB.

DTMF_DET_AGC_PLAY - SW AGC for DTMF detection in Play and Tone_Generate states. When incrementing this tunable by 1, the dynamic range is increased by 3 dB.

For more information on the SW AGC commands, refer to index values 60 and 61 in Table 2-6 of the Command Description section.

ECHO CANCELLATION

Echo cancellation is a technique used to improve the performance of DTMF detection during speech synthesis, tone generation, and OGM playback. For echo cancellation to work properly, HW AGC must not be active in parallel. Thus, to take advantage of echo cancellation, the microcontroller must control the HW AGC, if exists, (i.e., disable the HW AGC during PLAY, SYNTHESIS and TONE_GENERATE states and enable it again afterwards). If HW AGC can not be disabled, do not use echo cancellation.

The microcontroller should use the CFG command to activate/deactivate echo cancellation.

NOTE *Normally, a HW AGC is not required with The ISD-T360SA, since SW AGC is optional for the VCD algorithm, DTMF detection and the speakerphone module.*

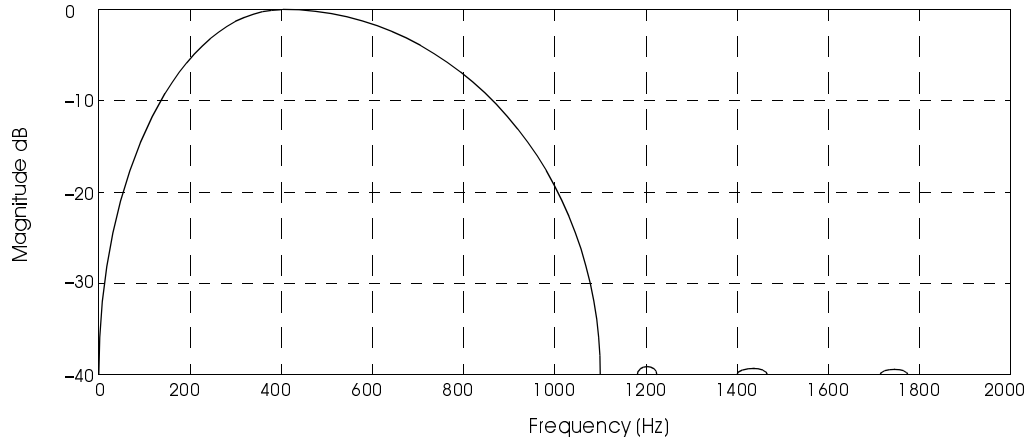
2.3.3 TONE AND ENERGY DETECTION (CALL PROGRESS)

The VoiceDSP processor detects busy and dial tones, constant energy level, and no-energy (VOX). This enables call progress tracking. Detection is active throughout the operation of the VoiceDSP processor. Detection can be configured using the SDET (Set Detectors Mask) command, which controls the reporting of the occurrence of tones, and the RDET (Reset Detectors) command which resets the detectors. The accuracy of the tone length, as reported by the tone detectors, is ±10 ms.

TUNABLE PARAMETERS

Tunable parameters control the detection of busy and dial tones, constant energy level (in the frequency range 200–3400Hz), and no-energy. These parameters should be tuned to fit the system hardware. In addition, changes may be required to the tunable parameters according to the setting (On or Off) of the HW Automatic Gain Control (HW AGC), if exists. For more information refer to Table 2-7 and Table 2-8 of the Command Description section.

Figure 2-1: Busy and Dial-Tone Band-Pass Filter Frequency Response



BUSY AND DIAL TONES

Busy and dial-tone detectors work with a band-pass filter that limits the frequency range in which tones can be detected to 0–1100Hz. Figure 2-2 shows the frequency response of this band-pass filter.

The design of the busy-tone detector allows very high flexibility in detecting busy tones with varying cadences.

The tunable parameters are divided into four sets:

1. Busy Tone On-time and Off-time Range Specification:

   BUSY_DET_MIN_ON_TIME
   BUSY_DET_MIN_OFF_TIME
   BUSY_DET_MAX_ON_TIME
   BUSY_DET_MAX_OFF_TIME

2. Busy Tone Cadence Control Specification

   BUSY_DET_VERIFY_COUNT
   BUSY_DET_DIFF_THRESHOLD
   BUSY_DET_TONE_TYPE

   BUSY_DET_VERIFY_COUNT determines the number of On/Off cadences that detector should detect before reporting busy tone presence.

   BUSY_DET_DIFF_THRESHOLD describes the maximum allowed difference between two compared On or Off periods, as determined by the BUSY_DET_TONE_TYPE tunable parameter.

   BUSY_DET_TONE_TYPE specifies the type of cadences that are supported.

   Legal values are:

   Two cadences only
   Three cadences only
   Both two and three cadences.

   The acceptance criteria for two cadences:

   [E1–E3] < BUSY_DET_DIFF_THRESHOLD
   and
   [S1–S3] < BUSY_DET_DIFF_THRESHOLD

   The acceptance criteria for three cadences:

   [E1–E4] < BUSY_DET_DIFF_THRESHOLD
   and
   [S1–S4] < BUSY_DET_DIFF_THRESHOLD

3. Busy and Dial Tone Energy Thresholds

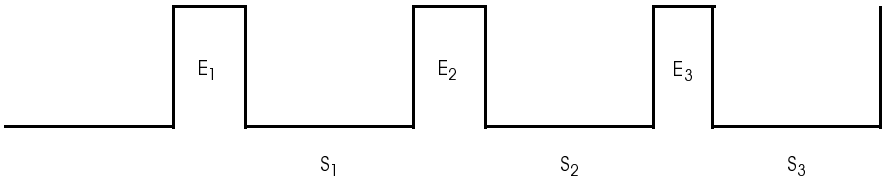
   TONE_DET_ON_ENERGY THRESHOLD

   TONE_DET_OFF_ENERGY THRESHOLD

4. Busy Detection Time

   BUSY_DET_MIN_TIME

Figure 2-2: Busy-Tone Detector—Default Cadence Specification



$[E_1 - E_3] < 100$ ms $[S_1 - S_3] < 100$ ms $100 < E_i < 1680$ ms $70 < S_i < 1220$ ms

CONSTANT ENERGY

The constant-energy detector reports the presence of constant energy in the range of 200Hz to 3400Hz. It is intended to detect both white and pink noise and can be used to detect line disconnection during recording.

It is recommended to use the constant energy mechanism in conjunction with the no-energy (VOX) mechanism.

The following tunable parameters control the operation of the constant-energy detector:

CONST_NRG_DET_TIME_COUNT
CONST_NRG_DET_TOLERANCE_TIME
CONST_NRG_DET_LOW_THRESHOLD
CONST_NRG_DET_HIGH_THRESHOLD

NO ENERGY (VOX)

The no-energy detector reports when the energy in the frequency range of 200Hz to 3400Hz remains below a pre-programmed threshold for a pre-programmed time-out. A programmable tolerance is allowed.

It is recommended to use the no-energy (VOX) mechanism in conjunction with the constant-energy mechanism.

The following tunable parameters control the operation of the no-energy (VOX) mechanism:

VOX_DET_ENERGY_THRESHOLD
VOX_DET_TIME_COUNT
VOX_DET_TOLERANCE_TIME

2.3.4 FULL-DUPLEX SPEAKERPHONE

The speakerphone feature lets the user communicate through a telephone line, using the unit's speaker and microphone instead of its handset. The speakerphone processes signals sent from the line to the speaker, and from the microphone to the line. It also performs the necessary switching, attenuation and echo cancellation on the signals present on the line/speaker.

The ISD-T360SA speakerphone is simple to use; it requires no special hardware or training for the echo cancelers. The gain control is fully digital, which eliminates the need for analog gain control hardware.

The speakerphone features two types of echoes, the electrical echo (line or circuit) and the acoustic echo. The electrical echo is a result of an imperfect impedance match between the 4- to 2-wire interface (hybrid) and the line impedance. The electrical echo, relatively short term, has a transfer function that varies slowly. The second echo, the acoustic echo, is a line impedance returning from the speaker to the microphone. This echo is relatively long term, and its transfer function may vary quite quickly if anyone, or anything, moves in the room. Both echoes must be canceled to achieve a high-quality hands-free system.

For more details of the speakerphone tunable parameters refer to Table 2-9 of the Command Description section.

SPEAKERPHONE TERMINOLOGY

Send Path

The signal path from the microphone (near-end speaker) to the line (far-end listener). The microphone is the input port, and line-out is the output port of this signal path.

Receive Path

The signal path from the line (far-end speaker) to the loudspeaker (near-end listener). The line-in is the input port, and the speaker is the output port for this signal path.

AEC

Acoustic Echo Controller. The part in the speakerphone algorithm that controls the echo in the send path.

EEC

Electric Echo Controller. The part in the speakerphone algorithm that controls the echo in the receive path.

SPEAKERPHONE MODES OF OPERATION

Full-Duplex (ON)

The speakerphone works in full-duplex mode, meaning both parties can speak and hear each other simultaneously. In this mode both the acoustic and electric echo controllers are active. The VoiceDSP processor tone detectors are not active in this mode.

Mute

In this mode of operation, the speakerphone generates silence to the line. The near-end listener can hear the far-end speaker but not vice versa. Tone detectors are not active.

Hold

During Hold mode silence is generated to the line and the speaker, and neither side can hear the other.

Restart

In Restart mode the speakerphone re-initializes itself to the last speakerphone mode (full-duplex, transparent or mute). This mode should be used to resume the speakerphone operation when there is a significant change in the environmental conditions (e.g., parallel pickup) that may affect the speakerphone quality.

Transparent

While in Transparent mode, the speakerphone works in full-duplex mode but without echo cancellation.

Samples from the microphone are transferred to the line, and samples from the line are transferred to the speaker, with no processing. This mode should be used only for tuning and testing the system.

Silence

In silence mode, only the electric echo cancellor is active. This is in order to improve CIDCW CAS detection performance. Refer to CID Type II Detailed Description on page 2-24

Listen

In Listen mode the line is audible on the speaker, and the processor tone detectors are active.

During Listen mode, dialing with the GT command and call progress detection are possible, since the busy and dial tone detectors are active.

The following pseudo-code demonstrates how to make a call from speakerphone mode:

Figure 2-3: Speakerphone Pseudo Code Representation

```
while () {
    EV = wait_event()
    case EV of:
        spkr_button_pressed:
            if (speakerphone_on) {
                SSM 0   // Put VoiceDSP in idle mode
                first_digit = TRUE
                deactivate_digit_timeout_event()
            }
            else
                SSM 1   // Put VoiceDSP in full-duplex speakerphone mode
        digit_pressed:
            if (first_digit) {
                SSM 4       // Enter LISTEN mode
                first_digit = FALSE
            }
            GT <dtmf_of_digit> // Dial the digit
            S           // Stop. Note that after the S command
                        // the VoiceDSP is still in speakerphone mode
            enable_digit_timeout_event() // To "guess" when dialing is completed.
        digit_timeout_event:
            SSM 1       // Dialing is completed, Go back to full-duplex mode
            deactivate_digit_timeout_event()
}
```

2.3.5 SPEECH SYNTHESIS

Speech synthesis is the technology used to create messages out of predefined words and phrases stored in a vocabulary.

There are two kinds of predefined messages: fixed messages (voice menus in a voice-mail system) and programmable messages (time-and-day stamp, or the *You have n messages* announcement in a DTAD).

A vocabulary includes a set of predefined words and phrases, needed to synthesize messages in any language. Applications which support more than one language require a separate vocabulary for each language.

INTERNATIONAL VOCABULARY SUPPORT (IVS)

IVS is a mechanism by which the VoiceDSP processor utilizes several vocabularies stored on an external storage device. IVS enables the ISD-T360SA to synthesize messages with the same meaning, but in different languages, from separate vocabularies.

IVS Features

- Multiple vocabularies stored on a single storage device.
- Plug-and-play. The same microcontroller code is used for all languages.
- Synthesized and recorded messages use the same voice compression algorithm to achieve equal quality.
- Argumented sentences. (For example: *You have <n> messages*.)
- Auto-synthesized time-and-day stamp (driven by the VoiceDSP processor's clock).
- Support for various language and sentence structures:

- One versus many. (For example: You *have one message* versus *You have two messages*.)
- None versus many. (For example: *You have no messages* versus *You have two messages*.)
- Number synthesis (English—*Eighty* versus French—*Quatre-vingt*).
- Word order (English—*Twenty-one* versus German—*Einundzwanzig*).
- Days of the week (Monday through Sunday versus Sunday through Saturday).

VOCABULARY DESIGN

There are several issues, sometimes conflicting, which must be addressed when designing a vocabulary.

Vocabulary Content

If memory space is not an issue, the vocabulary could contain all the required sentences, each recorded separately.

If memory space is a concern, the vocabulary must be compact; it should contain the minimum set of words and phrases required to synthesize all the sentences. The least memory is used when phrases and words that are common to more than one sentence are recorded only once, and the IVS tool is used to synthesize sentences out of them.

A good combination of sentence quality and memory space is achieved if you take the "compact" approach, and extend it to solve pronunciation problems. For example, the word twenty is pronounced differently when used in the sentences *You have twenty messages* and *You have twenty-two messages*. To solve this problem, words that are pronounced differently should be recorded more than once, each in the correct pronunciation.

Vocabulary Recording

When recording vocabulary words, there is a compromise between space and quality. The words should be recorded and saved in a compressed form, and you should use the best voice compression for that purpose. However, lower compression rates do affect the voice quality.

Another issue to consider is the difference in voice quality between synthesized and recorded messages (e.g., between time-and-day stamp and ICMs in a DTAD environment). It is more pleasant to the human ear to hear both messages have the same sound quality.

Vocabulary Access

Sometimes compactness and high quality are not enough. There should be a simple and flexible interface to access the vocabulary elements. Not just the vocabulary but the code to access the vocabulary should be compact.

When designing for a multi-lingual environment, there are even more issues to consider. Each vocabulary should be able to handle language-specific structures and designed in a cooperative way with the other vocabularies so that the code to access each vocabulary is the same. When you use the command to synthesize the sentence *Monday, 12:30 P.M.*, you should not care in what language it is going to be played back.

IVS VOCABULARY COMPONENTS

This section describes the basic concept of an IVS vocabulary, its components, and the relationships between them.

Basic Concepts

An IVS vocabulary consists of words, sentences, and special codes that control the behavior of the algorithm which VoiceDSP processor uses to synthesize sentences.

Word Table

The words are the basic units in the vocabulary. Create synthesized sentences by combining words in the vocabulary. Each word in the vocabulary is given an index which identifies it in the word table.

Note that, depending on the language structures and sentences synthesized, you may need to record some words more than once in the vocabulary. For example, if you synthesize the sentences: *you have twenty messages* and *you have twenty-five messages*, the word *twenty* is pronounced differently. In this example, *twenty* should be defined as two different words.

Number Tables

The number tables allow you to treat numbers differently depending on the context.

Example 1: The number 1 can be announced as *one* as in *message number one* or as first as in *first message*.

Example 2: The number 0 can be announced as *no* as in *you have no messages* or as *oh* as in *monday, eight oh five* A.M.

A separate number table is required for each particular type of use. The number table contains the indices of the words in the vocabulary that are used to synthesize the number. Up to nine number tables can be included in a vocabulary.

Sentence Table

The sentence table describes the predefined sentences in the vocabulary. The purpose of this table is to make the microcontroller that drives the VoiceDSP processor independent of the language being synthesized. For example, if the Flash and/or ROM memory contains vocabularies in various languages, and the first sentence in each vocabulary means you have n messages, the microcontroller switches languages by issuing the following command to VoiceDSP processor:

```
SV <storage_media>, <vocabulary_id> -Select a
new vocabulary
```

The microcontroller software is thus independent of the grammar of the language in use. The sentences consist of words, which are represented by their indices in the vocabulary.

Sentence 0

All sentences but one are user defined. The VoiceDSP processor treats the first sentence in the sentence table (sentence 0) specially, to support time-and-day stamp. The processor assumes that the sentence is designed for both system time, and message time-and-day stamp announcement, and has one argument which is interpreted as follows:

0	System time is announced
1	The time-and-day stamp of the current message is announced.

Example 1: When the microcontroller sends the command: SAS 0, 0

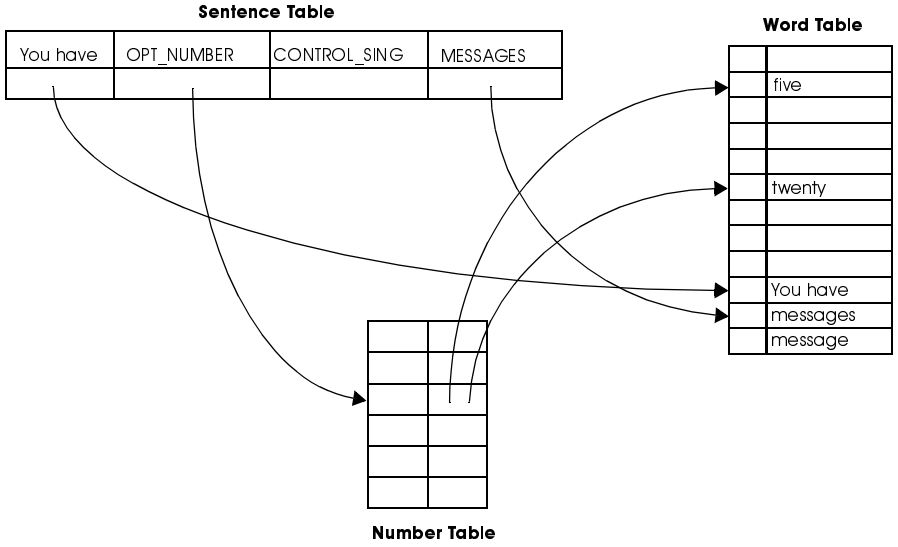
The system time and day is announced.

Example 2: When the microcontroller sends the command: SAS 0, 1

The current message time-and-day stamp is announced.

The following Figure 2-4 shows the interrelationship between the three types of tables.

Figure 2-4: The Interrelationship between the Word, the Number, and the Sentence Tables



Control and Option Codes

The list of word indices alone cannot provide the entire range of sentences that the VoiceDSP processor is able to synthesize. IVS control and option codes send special instructions to control the speech synthesis algorithm's behavior in the processor.

For example, if the sentence should announce the time of day, the VoiceDSP processor should be able to substitute the current day and time in the sentence. These control words do not represent recorded words, rather they instruct the processor to take special actions.

THE IVS TOOL

The IVS tool includes two utilities:

1. The DOS-based IVS Compiler
2. IVSTOOL for Windows. A Windows 95/98/2000 utility

The tools help create vocabularies for the VoiceDSP processor. They take you from designing the vocabulary structure, through defining the vocabulary sentences, to recording the vocabulary words.

IVS Compiler

The IVS compiler runs on MS-DOS (version 5.0 or later) and enables you to insert your own vocabulary, (i.e., basic words and data used to create numbers and sentences, as directories and files in MS-DOS). The IVS compiler then outputs a binary file containing that vocabulary. In turn, this information can be burned into an EPROM or Flash memory to be used by the VoiceDSP software.

IVS Voice Compression

Each IVS vocabulary can be compiled with either the 4.7 Kbit/s, the 6.7 Kbit/s or the 8.7 Kbit/s voice compression algorithm, or in PCM format. Define the bit rate before compilation. The VoiceDSP processor automatically selects the required voice decompression algorithm when the SV command chooses the active vocabulary.

Graphical User Interface (GUI)

The IVS package includes a Windows utility to assist the vocabulary designer to synthesize sentences. With this utility, you can record words, compose sentences and listen to words and sentences in the specific compression rate quality selected.

HOW TO USE THE IVS TOOL WITH THE VOICEDSP PROCESSOR

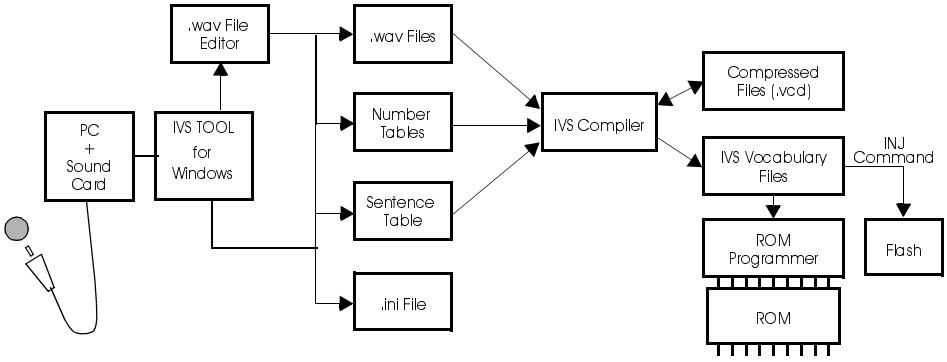
The IVS tool creates IVS vocabularies, and stores them as a binary file. This file is burnt into a ROM device or programmed into a Flash memory device using the INJ (Inject IVS) command. The VoiceDSP processor SO (Say One Word) command is used to select the required vocabulary. The SW (Say Words), SO, SS (Say Sentence) and SAS (Say Argumented Sentence) commands are used to synthesize the required word or sentence. The typical vocabulary-creation process using the IVS Tool software is as follows:

1. Design the vocabulary.
2. Create the vocabulary files (as described in detail below). Use IVS TOOL for Windows to simplify this process.
3. Record the words using any standard PC sound card and sound editing software, that can create.wav files.
4. Run the IVS compiler to compress the.wav files, and compile them and the vocabulary tables into an IVS vocabulary file.
5. Repeat steps 1 to 4 to create a separate IVS vocabulary for each language that you want to use. Note that each language file must have a different ID number. To specify an ID Number, go to the "Vocabulary ID" option in the Vocabulary/Build Options menu.
6. Exit IVS Tool and burn the IVS vocabulary files into a ROM (or Flash memory) device. Use the INJ (Inject IVS) command to program the data into a Flash device.
7. For injecting vocabularies with multiple languages into the flash memory:
   - From a DOS prompt, issue the following DOS command: copy voc1.bin + voc2.bin voc_both.bin (where voc1.bin and voc2.bin are the two vocabulary files and voc_both.bin is the combined file.)
   - The two files are now combined into a single file.
   - Inject the united vocabulary file using the INJ command. (See page 2-52 for details.)
   - To switch between the two languages in real time operation, use the SV command. (See page 2-64 for details.) The SV *type id* command switches between specified vocabulary IDs.

Once the vocabulary is in place, the speech synthesis commands of the VoiceDSP processor can be used to synthesize sentences.

Figure 2-5 shows the vocabulary-creation process for a single table on a ROM or Flash memory device.

Figure 2-5: Creation of an IVS Vocabulary



2.3.6 CALLER ID

Calling Identity Delivery (CID) is a telephone service that allows a customer to receive information about a calling party. The information includes details about the call, such as date and time, the calling party's number and name, and the called party's number. ISD_T360SA supports CID on-hook (also called CID Type I) for Bellcore (US), French, Spanish, Dutch and Japanese standards, and CID off-hook (also called CID Type II) for Bellcore standard.

CID TYPE I

CID Type I is used while the telephone is on-hook, that is, when a customer's line is open to receive a call. The CID information transferred is about a new incoming call. The data transmission occurs during the long silent interval between the first and the second power ringing patterns.

In Bellcore standard, CID features include Calling Number Delivery (CND) and Calling Name Delivery (CNAM), while other standards support similar features.

CID TYPE II

CID Type II is used while the telephone is off-hook, that is, when a customer's line is busy. In this case, a customer is engaged in another conversation, while receiving a new incoming call (a waited call). The CID information transferred is about the waited call. The transmission of the information takes place immediately after the customer is alerted to the new call, so the customer can use this information to decide whether to take the new waited call.

CID Type II, also called Calling Identity Delivery on Call Waiting (CIDCW), is supported for Bellcore standard.

It should be noticed that CID Type II is capable of everything CID Type I is capable of, and in addition it supports the CIDCW service. Therefore, the CIDCW service is always activated together with the CND and CNAM services.

MESSAGE FORMAT

The Caller ID information is transmitted in forms of data messages. A message format is defined according to the specific CID standard.

The Bellcore (US) standard supports two message formats: Single Data Message Format (SDMF) and Multiple Data message Format (MDMF).

In the on-hook case, the data is transmitted in one of these formats, according to the CID feature:

- SDMF for the Calling Number Delivery (CND) feature, which provides a calling party's number along with the date and time of the call.
- MDMF for the Calling Name Delivery (CNAM) feature, which provides a calling party's name along with the date and time of the call.

In the off-hook case, the data is transmitted in MDMF for the Calling Identity Delivery on Call Waiting (CIDCW) feature, which provides CID features information for waited calls.

Single Data Message Format (SDMF) transmits information as a series of data words specifying message type, message length, message data, and error detection information, as illustrated in Figure 2-6.

Figure 2-6: Single Data Message Format (SDMF)

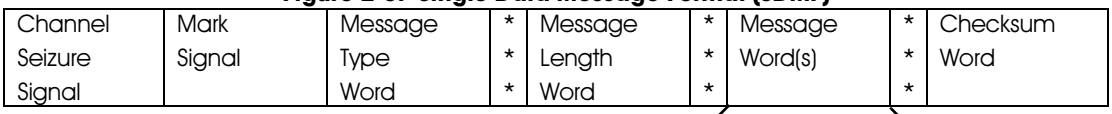
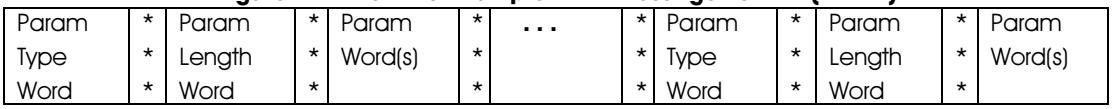


Figure 2-7: Detail of Multiple Data Message Format (MDMF)



Multiple Data Message Format (MDMF) is similar to SDMF, while the Message Word(s) consist of parameter message(s). The parameter message consists of one or more parameters. Each parameter is a series of data words specifying parameter type, parameter length, and parameter data, as illustrated in Figure 2-7.

NOTE: The stars between words indicate mark bits received in order to maintain a continuous signal.

The message type word in both formats and the parameter type word in the MDMF contain values defining specific services

The message length word in both formats contains the number of words in the message following it, excluding the checksum word.

For on-hook data transmission, each data message is preceded by a channel seizure signal and a Mark signal. For off-hook data transmission, each data message is preceded by only the Mark signal.

Other standards support similar message formats to transmit the CID data, according to the requirements of the standard.

The VoiceDSP processor provides the same interface to the Caller ID data, using the GCID (Get Caller ID) command, regardless of the selected standard. (For more information refer to the GCID command on page 2-38).

CID TYPE I – DETAILED DESCRIPTION

Interface with the SPCS

The CID feature requires a data transmission between the SPCS (Stored Program Control System) which is the Central Office, and the CPE (Customer Premises Equipment) which is the customer.

In Bellcore standard, data transmission occurs during the long silent interval between the first and the second power ringing patterns. The transmission begins at least 500 ms, but no later than 1.5 seconds, after the end of the first power-ringing pattern. The transmission ends at least 200 ms before the start of the second power-ringing pattern.

If the line goes off-hook before or during data transmission, data transmission stops.

Other standards support similar timing requirements for the on-hook data transmission, according to the requirements of the standard.

Interface with the microcontroller

During an on-hook case, when the line is idle (not engaged), a new incoming call may be received. In order to activate CID Type I, the microcontroller sends the ACID command (with CID Type I as a parameter) to the VoiceDSP.

If the CID physical layer is Bell202 or V.23, the ACID command should be sent by the microcontroller after the first ring, but before the second ring (timing should be according to the relevant standard). It is the responsibility of the microcontroller's ring detection algorithm to determine when the first ring has ended.

If the CID physical layer is DTMF, the ACID command should be sent by the microcontroller after it detects voltage polarity change on the line.

By receiving this command, the VoiceDSP clears the CID buffer, activates the CID modem, and enters the CID state. While in the CID state, the VoiceDSP processor receives the CID data into the CID buffer. (If during the session the microcontroller sends a STOP command to the VoiceDSP, the session should be stopped).

After the modem session has been completed, and the CID buffer has been loaded with the new CID data, the VoiceDSP processor deactivates the modem and reports an EV_NORMAL_END event. If the second ring occurs before the VoiceDSP processor reports an event, the microcontroller should terminate the CID modem session with the S (Stop) command.

If an error is encountered during the session, the VoiceDSP processor sets the ERR_CID bit in the error word, and reports an EV_ERROR event. In this case, details of the error are included in the CID data.

Whether an error was detected or not, the CID data is saved in an internal buffer and can be retrieved by the microcontroller with the GCID command. If the modem session is terminated with the S (Stop) command, the contents of the CID buffer are undefined. For detailed description of the CID buffer, refer to the GCID command on page 2-38.

If a message is recorded, the accompanying CID data can be automatically saved as part of the message, as controlled by the CID_RECORD tunable parameter. Refer to Table 2-13 of the Command Description section for detailed information of the tunable parameter.

CID TYPE II (CIDCW) – DETAILED DESCRIPTION

Interface with the SPCS

The CIDCW feature requires a handshaking sequence and a data transmission between the SPCS (Stored Program Control System) which is the Central Office, and the CPE (Customer Premises Equipment) which is the customer.

The SPCS starts the SPCS-CPE handshaking sequence by muting transmission to and from the far-end, and transmitting alerting sequence to the near-end. The alerting sequence consists of the Subscriber Alerting Signal (SAS) and the CPE Alerting Signal (CAS). The purpose of the SAS is to alert the subscriber to a new waiting call. The CAS is a machine detectable signal that is used to alert the CPE to prepare for data reception.

Upon detection of the CAS by the CPE, the CPE temporarily mutes the subscriber's handset and disables the keypad to prevent near-end interference. Within 100 ms after the end of the CAS, the CPE responds to the SPCS with an Acknowledgement signal (ACK) – specified by tunable parameters – indicating its readiness. (For more details of the Acknowledgement tunable parameters refer to Table 2-13 of the Command Description section).

Data transmission begins only after the SPCS detects the ACK. The SPCS transmits the message to the customer's CPE and reestablishes the voice path by unmuting the far-end shortly after the end of data transmission. Within 50 ms after the data is received, the CPE unmutes the handset and enables the keypad. The CPE also unmutes the near-end in case it times out waiting for data (500 ms after sending the ACK signal).

Interface with the microcontroller

During an off-hook case, when a customer is engaged in a telephone call, a new incoming call (a waited call) may occur. Therefore, from the beginning of a telephone call, the CPE should be able to detect a CAS signal that may be sent by the SPCS. This means that the microcontroller should send a SDET command to the VoiceDSP at the beginning of a call, to tell the VoiceDSP to activate the CAS detector. By receiving this command, the VoiceDSP activates the CAS detector, and is ready to detect a CAS.

It is preferable that the microcontroller would send the SDET command at the beginning of the call itself, rather than when the CPE goes off-hook, before sending the DTMF digits. But if the microcontroller would send the command when the CPE goes off-hook, it should send an SDET command before and after each DTMF, to deactivate and activate respectively the CAS detection.

During the time the CAS detector is activated, the following commands are the only commands that the microcontroller is allowed to send: SDET, GEW, GSW and S (Stop).

If a CAS is received and detected, the VoiceDSP sends an EV_CAS event to the microcontroller, indicating the detection of a CAS. The VoiceDSP deactivates the CAS detector, and the microcontroller disables the keypad and mutes the handset (or deactivates the speakerphone, in case the VoiceDSP was activated in full-duplex speakerphone mode).

Then the microcontroller sends the ACID command (with CID Type II as a parameter) to the VoiceDSP. By receiving this command, the VoiceDSP first enters a new state, CIDCW state, and sends an ACK to the SPCS (specified by the tunable parameter CID_ACK_TYPE). While in the CIDCW state, the following commands are the only commands that the microcontroller is allowed to send: GEW, GSW and S (Stop). Then the VoiceDSP enters the CID state, the same state that it enters when activating CID Type I. While in this state, it clears the CID buffer, activates the CID modem, and receives the CID data.

After the modem session has been completed, and the CID buffer has been loaded with the new CID data, the VoiceDSP processor deactivates the modem and reports either an EV_NORMAL_END event (in case the session succeeded) or EV_ERROR (in case the session didn't succeed). An EV_ERROR event is reported also if the CID data hasn't been received within 500 ms from the end of the ACK signal.

After the process ends, the microcontroller should send the GCID command to the VoiceDSP, in order to receive the CID information. In case of an error, the error details are in the CID status bytes (bytes 0 and 1 of the CID buffer). The GCID command should be sent before the next activation of the CAS detector by the microcontroller. For detailed description of the CID buffer, refer to the GCID command on page 2-38.

Within 50 ms after the CID data is received, the microcontroller should enable the keypad and unmute the handset (or activate the speakerphone, in case the VoiceDSP is in a speakerphone mode).

If the CPE returns to the on-hook case during the process (either during the handshaking sequence or during the data transmission), the process should be stopped. In this case, the microcontroller should send a STOP command to the VoiceDSP to stop the process. The process also stops in any case the microcontroller sends a STOP command to the VoiceDSP. In the case that the modem session is terminated with the S (Stop) command, the contents of the CID buffer are undefined.

CIDCW can work in three different modes regarding the presence of a speakerphone:

1. The basic mode – The VoiceDSP does not implement the speakerphone feature (there is only one codec).
2. Full-duplex speakerphone mode – The VoiceDSP implements the speakerphone feature, and currently is activated in full-duplex speakerphone mode (there are two codecs).
3. Silence speakerphone mode – The VoiceDSP implements the speakerphone feature, and currently is activated in silence speakerphone mode (there are two codecs).

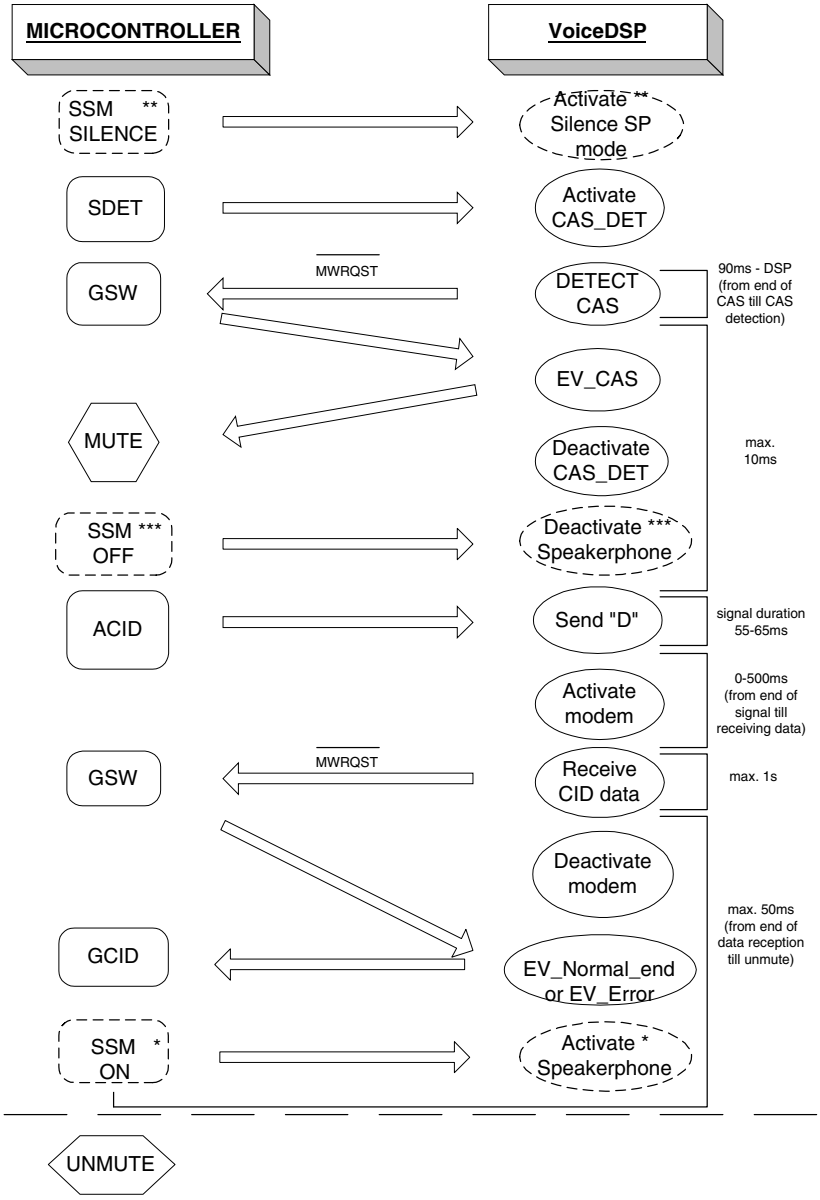
In modes 2 and 3, an echo cancellation can be operated for the CAS detection.

Figure 2-8 on the following page illustrates the interface between the VoiceDSP and the microcontroller in the three possible modes.

NOTE *Note: In the Full-duplex speakerphone mode, after the activation of the 'SSM OFF' command, the Microcontroller should not activate an 'SSM ON' command before it receives an 'EV_NORMAL_END' or an 'EV_ERROR'.*

In the Silence speakerphone mode, the 'SSM SILENCE' command should also be activated in the event a user starts to speak through the speakerphone (a full-duplex speakerphone mode), and then moves to speak through the handset (a silence speakerphone mode).

Figure 2-8: Three possible modes of Interface between VoiceDSP and microcontroller



* Full-duplex Speakerphone Mode

** Silence Speakerphone Mode

*** Full-duplex and Silence Speakerphone Modes

CALLER ID MODEM

In both CID Type I and CID Type II, the VoiceDSP processor activates the CID modem in order to receive the data from the SPCS. It should be noticed that the CID modem's functionality is connected to the HW AGC activation in the following way:

When Bell202 or V.23 is used as the physical layer and if HW AGC is implemented, HW AGC must not be active in parallel. Therefore, the microcontroller must disable the HW AGC before activating the modem.

When working in slave mode, the samples provided to the modem should be in 7.273 kHz sampling rate. Samples provided to all other modules can be in 8.0 kHz sampling rate.

TEST MODE

This mode of operation allows easier tuning of the analog circuits to the CID modem. It is available only for CID standards that use either Bell202 or V.23 modems. The CID test mode provides access to the following test points:

BL	Bit Level of the reception. During reception of the CID data, each bit is output to the CIDBL pin.
CI	Carrier Indication. During reception of the CID data, the CIDCI pin provides the Carrier Indication.

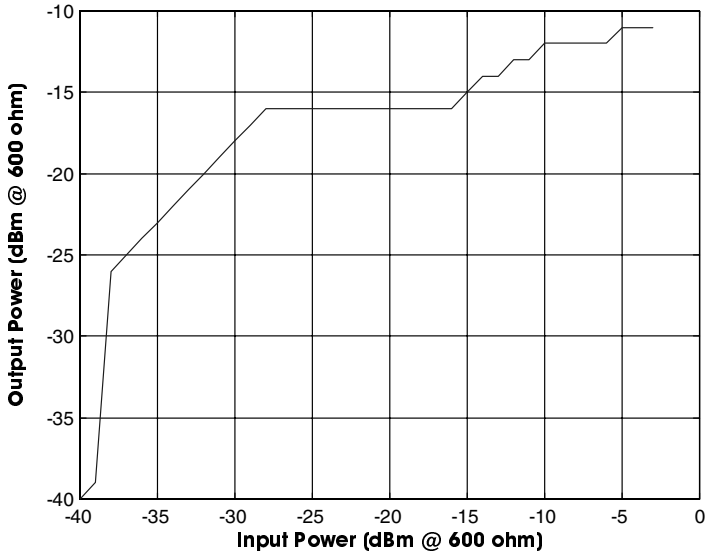
2.3.7 SW AGC

The Software Automatic Gain Control (SWAGC) algorithm can be activated with the compression algorithm or with the Speakerphone algorithm. When used with the compression algorithm, it regulates the input signal to a dynamic range that can provide higher compression quality. When used with the speakerphone algorithm, it amplifies the input signal from the line to the speaker and this improves the quality for attenuated far end speakers.

The SWAGC algorithm senses the energy level and updates the signal gain in order to amplify low energy signals and to avoid signal saturation. The SWAGC feature eliminates the need for an external hardware AGC, thus reducing hardware costs and complexity. Hardware AGC may be used with the compression algorithm to avoid signal saturation prior to sampling the signal. Note that the hardware AGC cannot be used with the speakerphone algorithm.

A set of tunable parameters is available for controlling the SWAGC algorithm. Different values can be used for the compression and speakerphone algorithms. The SWAGC may even be turned completely off for either one of the algorithms. The way to use different tunable values is by updating the tunable parameter values prior to activating each one of the algorithms. For more details on the SWAGC tunable parameters, see Table 2-4 in the Command Description section.

Figure 2-9: SWAGC Characteristics with Default Tunable Parameters Values



2.4 VOICEDSP PROCESSOR COMMANDS—QUICK REFERENCE TABLE

Table 2-2: Speech Commands

Command		Description	Opcode Hex	Source State	Result State	Command Parameters		Return Value	
Name	S/A*					Description	Bytes	Description	Bytes
ACID	A	Activate Caller ID	2D	IDLE	CID	Type	1	None	-
CCIO	S	Configure Codec I/O	34	RESET, IDLE	No change	Config_value	1	None	-
CFG	S	Configure VoiceDSP	01	RESET	No change	Config_value	3	None	-
CMSG	S	Create Message	33	IDLE	MSG_OPEN	Tag, Num_of_ blocks, comp_rate	2+2+1	None	-
CMT	S	Cut Message Tail	26	IDLE	No change	Time_length	2	None	-
CVOC	S	Check Vocabulary	2B	IDLE	No change	None	-	Test result	1
DM	S	Delete Message	0A	IDLE	No change	None	-	None	-
DMS	S	Delete Messages	0B	IDLE	No change	Tag_ref, Tag_mask	2 + 2	None	-
GCFG	S	Get Configuration Value	02	RESET, IDLE	No change	None	-	Version	1
GCID	S	Get Caller ID	2E	IDLE	IDLE	Src, Offset, N	3	Caller ID Data	N
GEW	S	Get Error Word	1B	All states	No change	None	-	Error word	2
GI	S	Get Information item	25	PLAY, RECORD, SYNTHESIS, TONE_ GENERATE, IDLE	No change	Item	1	Item value	2
GL	S	Get Length	19	IDLE	No change	None	-	Message length	2
GMS	S	Get Memory Status	12	IDLE	No change	Type	1	Remaining memory blocks	2
GMT	S	Get Message Tag	04	IDLE	No change	None	-	Message tag	2
GNM	S	Get Number of Messages	11	IDLE	No change	Tag_ref, Tag_mask	2 + 2	Number of messages	2
GSW	S	Get Status Word	14	All states	No change	None	-	Status word	2
GT	A	Generate Tone	0D	IDLE	TONE_ GENERATE	Tone (single Tone or DTMF)	1	None	-
GTD	S	Get Time and Day	0E	IDLE	No change	Time_day_ option	1	Time and day	2
GTM	S	Get Tagged Message	09	IDLE	No change	Tag_ref, Tag_mask,Dir	2+2+1	Message found	1

Table 2-2: Speech Commands (Continued)

Command		Description	Opcode Hex	Source State	Result State	Command Parameters		Return Value	
Name	S/A*					Description	Bytes	Description	Bytes
GTUNE	S	Get Tunable Parameter	06	IDLE, RESET	No change	Index	1	Parameter_ value	2
INIT	S	Initialize System	13	RESET, IDLE	IDLE	None	-	None	-
INJ	S	Inject IVS data	29	IDLE	No change	N, $byte_1$..$byte_n$	4 + n	None	-
MR	S	Memory Reset	2A	RESET, IDLE	No change	None	-	None	-
P	A	Playback	03	IDLE	PLAY	None	-	None	-
PA	S	Pause	1C	PLAY, RECORD, SYNTHESIS, TONE_GENERATE, IDLE*	No change	None	-	None	-
PDM	S	Go To Power-Down Mode	1A	IDLE	No change	None	-	None	-
R	A	Record Message	0C	IDLE	RECORD	Tag (message Tag), Compression_ rate	2 + 1	None	-
RDET	S	Reset Detectors	2C	IDLE	No change	Detectors_ reset_mask	1	None	-
RES	S	Resume	1D	PLAY, RECORD, SYNTHESIS, TONE_GENERATE IDLE*	No change	None	-	None	-
RMSG	S	Read Message	32	IDLE, MSG_OPEN	MSG_OPEN	None	-	Data	32
S	S	Stop	00	All states but RESET	IDLE	None	-	None	-
SAS	A	Say Argumented Sentence	1E	IDLE	SYNTHESIS	Sentence_n, Arg	1 + 1	None	-
SB	S	Skip Backward	23	PLAY, IDLE*	No change	Time_length	2	None	-
SCID	S	Save Caller ID Data	35	IDLE	IDLE	Dest, Offset, N, Data	3 + N	None	-
SDET	S	Set Detectors Mask	10	IDLE	No change	Detectors_ mask	1	None	-
SE	S	Skip to End of Message	24	PLAY, IDLE*	No change	None	-	None	-
SETD	S	Set Time and Day	0F	IDLE	No change	Time_and_ day	2	None	-
SF	S	Skip Forward	22	PLAY, IDLE*	No change	Time_length	2	None	-
SLC	S	Set Line Channels	38	IDLE	No change	Active_chann els	1	None	-
SMSG	S	Set Message Pointer	30	IDLE, MSG_OPEN	MSG_OPEN	Num_of_ pages	2	None	-
SMT	S	Set Message Tag	05	IDLE	No change	Message_tag	2	None	-

Table 2-2: Speech Commands (Continued)

Command		Description	Opcode Hex	Source State	Result State	Command Parameters		Return Value	
Name	S/A*					Description	Bytes	Description	Bytes
SO	A	Say One Word	07	IDLE	SYNTHESIS	Word_number	1	None	-
SPS	S	Set Playback Speed	16	PLAY, SYNTHESIS, IDLE	No change	Speed	1	None	-
SS	A	Say Sentence	1F	IDLE	SYNTHESIS	Sentence_n	1	None	-
SV	S	Set Vocabulary Type	20	IDLE	No change	Type, Id	1 + 1	None	-
SW	A	Say Words	21	IDLE	SYNTHESIS	N, $word_1$...$word_n$	1 + N	None	-
TUNE	S	Tune Parameters	15	IDLE, RESET	No change	Index, Parameter_ value	1 + 2	None	-
VC	S	Volume Control	28	PLAY, SYNTHESIS, IDLE, TONE_GENERATE	No change	*vol_level* (increment/ decrement)	1	None	-
WMSG	S	Write Message	31	IDLE, MSG_OPEN	MSG_OPEN	Data	32	None	-

NOTE: ** Command is valid in IDLE state, but has no effect.*
S = Synchronous command
A= Asynchronous command

Table 2-3: Speakerphone Commands

Command		Description	Opcode Hex	Source State	Result State	Command Parameters		Return Value	
Name	S/A*					Description	Bytes	Description	Bytes
GEW	S	Get Error Word	1B	TONE_GENERATE, IDLE	No change	None	-	Error word	2
GSW	S	Get Status Word	14	TONE_GENERATE, IDLE	No change	None	-	Status word	2
GT	A	Generate Tone	0D	IDLE	TONE_GEN ERATE	Tone (Single Tone or DTMF)	1	None	-
RDET	S	Reset Detectors	2C	IDLE	No change	Detectors_reset _mask	1	None	-
S	S	Stop	00	TONE_GENERATE, IDLE	IDLE	None	-	None	-
SDET	S	Set Detectors Mask	10	IDLE	No change	Detectors_ mask	1	None	-
SSM	S	Set Speakerphone Mode	2F	IDLE	No change	Mode	1	None	-
TUNE	S	Tune Parameters	15	IDLE, RESET	No change	Index, Parameter_ value	1 + 2	None	-
VC	S	Volume Control	28	TONE_GENERATE, IDLE	No change	*vol_level* (increment/ decrement)	1	None	-

NOTE: ** Command is valid in IDLE state, but has no effect.*
S = Synchronous command
A = Asynchronous command

2.5 COMMAND DESCRIPTION

The commands are listed in alphabetical order.

The execution time for all commands, when specified, includes the time required for the microcontroller to receive the return value, where appropriate.

The execution time does not include the protocol timing overhead, i.e., the execution times are measured from the moment that the command is detected as valid until the command is fully executed.

NOTE: Each command description includes an example application of the command. The examples show the opcode issued by the microcontroller and the response returned by the VoiceDSP processor. For commands which require a return value from the processor, the start of the return value is indicated by a thick vertical line.

ACID Activate Caller ID *type*

Activates the Caller ID Type *type* process.

The Caller ID *type* may be one of the following:

0 Caller ID Type I

1 Caller ID Type II (CIDCW)

In CID Type I, the VoiceDSP processor changes to CID state and activates the CID modem. In CID Type II, the VoiceDSP processor first changes to CIDCW state and sends an ACK to the SPCS. Then it changes to CID state and activates the modem as in Type I.

When the modem session completes, the VoiceDSP processor reports an EV_NORMAL_END event. If the session does not complete successfully, the VoiceDSP processor sets the ERR_CID bit in the error word and reports an EV_ERROR event. The error details are in the CID status byte (bytes 0 and 1 of the CID buffer).

In both cases, the microcontroller can retrieve the contents of the CID buffer, using the GCID command.

If the modem session is terminated with the S (Stop) command, the contents of the CID buffer are undefined.

NOTE: 1. When ACID starts execution, the previous contents of the CID buffer are destroyed.
2. In Slave Mode, the sampling rate for FSK modem should be 7.273 kHz for proper FSK detection.

Example

ACID 00			
Byte sequence:	Microcontroller	2D	00
	VoiceDSP	2D	00
Description:	Activate CID Type I		

CCIO Configure Codec I/O *config_value*

Configures the voice sample paths in various states. It should be used to change the default VoiceDSP processor configuration. It is relevant only when two codecs are used and speakerphone is connected.

The *config_value* parameter is encoded as follows:

Bit 0

Loopback control.

0 Loopback disabled (default).

1 Loopback enabled. In the RECORD state, the input samples are echoed back, unchanged (i.e., no volume control), to the same codec.

Bit 1

Codec input control.

0 Input is received via the line codec (default).

1 Input is received via the speakerphone codec.

Bits 2–3

Codec output control.

00 In PLAY, IDLE, SYNTHESIS and TONE_GENERATE states, output is to both codecs. In RECORD state, output is to the non-input codec (no volume control). If the loopback control bit is set, output in RECORD state is to both codecs as well (default).

01 Output in all states is to the line codec.

10 Output in all states is to the speakerphone codec.

11 Output in all states is to both codecs.

Bits 4–7

Reserved.

Example

CCIO 01			
Byte sequence:	Microcontroller	34	01
	VoiceDSP	34	01
Description:	Enable loopback		

CFG Configure VoiceDSP *config_value*

Configures the VoiceDSP processor in various hardware environments. It should be used to change the default processor configuration.

The *config_value* parameter is encoded as follows:

Bits 0–3

Memory type.

0000 No Memory (default).

0001 Reserved.

0010 Reserved.

0011 Reserved.

0100 Samsung NAND Flash.

0101 Toshiba NAND Flash.

0110 Reserved.

0111 Reserved.

1000 - 1111 Reserved.

Bits 4–5

Number of installed memory devices.

00 1 (Default)

01 2

10 3

11 4

Bit 6

Caller ID Test Mode selection:

0 Normal Mode

1 Test Mode: the carrier presence on pin CIDCI. The CID bit value on pin CIDBL. Refer to Caller ID Test Mode on page 2-27 for more details

Bits 7-10

Caller ID Application Layer:

0000 Bellcore (US)

0001 French

0010 Dutch

0011 Japanese

0100 Spanish

0101 - 1111 Reserved

Bits 11-14
Reserved

Bit 15
Echo Cancellation Control (for DTMF Detection).

0 Echo cancellation off.

1 Echo cancellation is on during playback. (recommended)

Echo cancellation improves the performance of DTMF detection during playback. Echo cancellation can be turned on only with a system that can disable HW AGC (if present) during playback. A system featuring HW AGC, that cannot be controlled by the microcontroller (i.e., disabled or enabled), must not turn on this bit.

Bit 16

Clock bit rate (in Slave Mode only).

0 One bit rate clock (default).

1 Two bit rate clock.

Bit 17

Codec configuration.

0 Short-frame format (default).

1 Long-frame format (guaranteed by design but not tested).

Bits 18–19

Codec type.

00 16-bit linear (default).

01 μ-Law.

10 A-Law.

Bit 20

Codec interface mode.

0 Master codec interface (default).

1 Slave codec interface.

Bits 21–22

Number and type of codecs

00 One single codec (default).

01 Two single codecs.

10 One dual codec.

11 Reserved.

The codecs should be connected as follows:

Telephone line or equivalent - always connected as channel 0.

Speaker and microphone - connected as channel 1 in case of one dual codec, connected as channel 2 in case of two single codecs.

Bit 23

Reserved.

Example

CFG 188013					
Byte sequence:	Microcontroller	01	18	80	13
	VoiceDSP	01	18	80	13
Description:	Configure the VoiceDSP to work with: Single codec in Slave Mode and A-Law compressed samples. Data in Short Frame format and Single Bit Rate interface. Two Serial Toshiba Flash devices. Echo Cancellation for DTMF detection is On.				

CMSG Create Message *tag num_of_blocks compression_rate*

Creates a new message with a message tag *tag*, and compression rate *compression_rate* allocates *num_of_blocks* 4-Kbytes blocks for the new message, and sets the message pointer to the beginning of the message data. CMSG switches the VoiceDSP processor to the MSG_OPEN state.

The memory space available for the message data is computed as follows:

(127 x *num_of_blocks* – 2) x 32 bytes.

Once a message is open (the processor is in the MSG_OPEN state), the message pointer can be set to any position on a page (32 bytes) boundary within the message with the SMSG command. Modify the message contents with the WMSG command, and read with the RMSG command.

The microcontroller must issue an S command to close the message and switch the VoiceDSP processor to the IDLE state.

If the memory is full, EV_MEMFULL is set in the status word and no message is created. If the memory is not full but there is insufficient memory and the processor cannot allocate more memory space, EV_MEMLOW is set in the status word and no message is created.

The compression rate may be defined as 0 for PCM recording or either 1,2,3 for compression rates 4.7 Kbit/s, 6.7 Kbit/s, 8.7 Kbit/s respectively. See section "VCD" on page 2-10 for a description of the compression algorithm.

Example

CMSG 0101 0001						
Byte sequence:	Microcontroller	33	01	01	00	01
	VoiceDSP	33	01	01	00	01
Description:	Create a new message with a tag=0101, and allocate 1 block (4 Kbytes) for its data.					

CMT Cut Message Tail *time_length*

Cut *time_length* units, in 10 ms segments, off the end of the current message. The maximum value of *time_length* is 6550. In case of silence, cut-time accuracy is 0.1 to 0.2 seconds (depends on compression rate).

If time_length is greater than 6550 the ERR-PARAM is set in the ERR_WORD.

NOTE *If time_length is longer than the total duration of the message, the EV_NORMAL_END event is set in the status word and the message becomes empty but not deleted.*

A compressed frame represents 21 ms of speech, thus the minimum meaningful parameter is 3, (i.e., a 30 ms cut).

CMT 1 or CMT 2 have no effect.

The CMT command can not be used on data messages.

In PCM recording mode the accuracy of µ-, A-Law configuration is 0.50 sec while the accuracy of linear configuration is 0.25 sec.

Use the DM (Delete Message), or DMS (Delete Messages) command to delete the message.

Example

CMT 02BC				
Byte sequence:	Microcontroller	26	02	BC
	VoiceDSP	26	02	BC
Description:	Cut the last seven seconds of the current message.			

CVOC Check Vocabulary

Checks (checksum) if the IVS data of the currently selected vocabulary was correctly programmed to the ROM or Flash device.

If the vocabulary data is correct the return value is 1. Otherwise the return value is 0.

If the current vocabulary is undefined, ERR_INVALID is reported.

Example

CVOC			
Byte sequence:	Microcontroller	2B	AA
	VoiceDSP	2B	01
Description:	Check the current vocabulary. The VoiceDSP processor responds that the vocabulary is OK.		

DM Delete Message

Deletes the current message. Deleting a message clears its message tag.

Deleting the current message does not cause a different message to become current. The current message is undefined. If, for example, you issue the GTM command to skip to the next message, the first message that is newer than the just deleted message is selected as the current message.

The memory space released by the deleted message is immediately available for recording new messages.

Example

DM		
Byte sequence:	Microcontroller	0A
	VoiceDSP	0A
Description:	Delete current message.	

DMS Delete Messages *tag_ref tag_mask*

Deletes all messages whose message tags match the tag_ref parameter. Only bits set in tag_mask are compared i.e., a match is considered successful if:

message tag and *tag_mask* = *tag_ref* and *tag_mask*

where and is a bitwise AND operation.

After the command completes execution, the current message is undefined. Use the GTM command to select a message to be the current message.

The memory space released by the deleted messages is immediately available for recording new messages.

Example

DMS FFC2 003F						
Byte sequence:	Microcontroller	0B	FF	C2	00	3F
	VoiceDSP	0B	FF	C2	00	3F

Description:	Delete all old incoming messages from mailbox Number 2 in a system where the message tag is encoded as follows: Bits 0–2: mailbox ID 8 mailboxes indexed: 0 to 7 Bit 3: new/old message indicator 0—Message is old 1—Message is new Bits 4–5: message type 00—ICM/memo 01—OGM 10—Call transfer message Bits 6–15: not used *Note: the description of the tag is an example only. All bits of the tag are user-definable.*

GCFG Get Configuration Value

Returns one byte with the following information:

Bits 0–7

Magic number, which specifies the VoiceDSP firmware version.

Example

GCFG			
Byte sequence:	Microcontroller	02	AA
	VoiceDSP	02	01
Description:	Get the VoiceDSP processor magic number. The VoiceDSP processor responds that it is Version 1.		

GCID Get CID *src, offset, length*

Returns the Caller ID data.

The *src* may be one of the following:

0 returns the current content of the CID buffer.

1 returns the CID data of the current message.

offset specifies the start position in the CID buffer.

length specifies the number of bytes to retrieve.

The structure of the CID buffer, for all except the Japanese CID, is shown below.

Byte 0

First byte of status word.

Each bit represents either error (E), warning (W), or status (S). Whenever ERR_CID is set in the VoiceDSP processor error word, one of the (E) bits is set to indicate the error details.

When set to **1**, the bits have the following meanings:

Bit 0
Message format error

Bit 1
Parameter redundancy warning

Bit 2
Notification message detection (French CID only)

Bit 3
Parameter type warning

Bit 4
Absence of name parameter (1 if parameter detected)

Bit 5
Absence of caller number parameter (1 if parameter detected)

Bit 6
Command parameter indication for notification message (1 if parameter detected) (French CID only)

Bit 7
MARK minimum length violation
(for CID Type I – set by the tunable parameter CID_PARAM4)
(for CID Type II – set by the tunable parameter CID_PARAM7)

Byte 1

Second byte of status word.

Each bit represents either error (E), warning (W), or status (S). Whenever ERR_CID is set in the VoiceDSP processor error word, one of the (E) bits is set to indicate the error details.

When set to **1**, the bits have the following meanings:

Bit 0
End of message

Set after the last byte of the CID data has been received. If not set, it indicates that the session was not completed. The EV_NORMAL_END event, in the VoiceDSP processor status word, is set in parallel to this bit.

Bit 1
Checksum error

Bit 2
No carrier of the FSK signal

Bit 3
Reports status of receiving CID data and setting a timer
(for CID Type I – set by the tunable parameter CID_TIMEOUT0)
(for CID Type II – set by the tunable parameter CID_PARAM6)

Bit 4
Report status of waiting for CID data and setting a timer
(for CID Type I – set by the tunable parameter CID_TIMEOUT1)
(for CID Type II – set by the tunable parameter CID_PARAM5)

Bit 5
Report status of receiving the seizure signal and setting a timer (set by tunable parameter CID_TIMEOUT2)

Bit 6
Report status of waiting for the message type and setting a timer (set by the tunable parameter CID_TIMEOUT3)

Bit 7
Field overflow

Bytes 2-21
Caller Number field. Null padded.

If the telephone number is absent, byte 2 is used to indicate the reason:

'P' Private (ASCII Character 'P')

'O' Out-of-area / unavailable

Bytes 22-41
Caller Name. Null padded.

If the caller name is absent, byte 22 indicates the reason:

'P' Private (ASCII Character 'P')

'O' Out-of-area / unavailable

Bytes 42-43
Month – ASCII code in the range 01 to 12, where byte 42 is the MSB.

Bytes 44-45
Day – ASCII code in the range 01 to 31, where byte 44 is the MSB.

Bytes 46-47

Hour – ASCII code in the range 00 to 23, where byte 46 is the MSB.

Bytes 48-49
Minute – ASCII code in the range 00 to 59, where byte 48 is the MSB.

Bytes 50-69
Called Number Field (Spanish CID only).

Byte 70
Call type (Spanish CID) / Command parameter (French CID notification message).

Japanese CID:

The Japanese CID buffer contains a total of 130 bytes. Its structure is as follows:

Byte 0
Reserved.

Byte 1
Status word. When set to 1, the bits have the following meanings:

Bit 0
End of message (status)

Bit 1
Reserved

Bit 2
No carrier (error)

Bit 3
More than tunable parameter CID_TIMEOUT0 time has elapsed from the ACID command until the end of the message

Bit 4
More than tunable parameter CID_TIMEOUT1 time has elapsed from the ACID command until the start of the message

Bit 5
Buffer overflow (more than 128 bytes received)

Bit 6
Reserved

Bit 7
Reserved

Bytes 2-129

CID data, including all control, CRC, message bytes together with parity bits. The data is null (0) padded. It is not parsed, but appears in the buffer as received.

Valid Range

The value of *offset* plus *length* must not be larger than the size of the CID buffer in use. The maximum value is 130 for Japanese CID, when using CID buffer (i.e., *src* = 0), and 71 for all other cases.

If this condition is violated, ERR_PARAM is set.

Example

GCID 00 00 30				
Byte sequence:	Microcontroller	2E 00 00 30	AA	AA
	VoiceDSP	2E 00 00 30	48-byte CID data	
Description:	VoiceDSP returns 48 first bytes of CID data (from the current CID buffer)			

GEW Get Error Word

Returns the 2-byte error word.

ERROR WORD

The 16-bit error word indicates errors that occurred during execution of the last command. If an error is detected, the command is not processed; the EV_ERROR bit in the status word is set to 1, and the $\overline{\text{MWRQST}}$ signal is activated (set to 0).

The GEW command reads the error word. The error word is cleared during reset and after execution of the GEW command.

If errors ERR_COMMAND or ERR_PARAM occur during the execution of a command that has a return value, the return value is undefined. The microcontroller must still read the return value, to ensure proper synchronization.

15 9	8	7	6	5	4	3	2	1	0
Res	ERR_CID	ERR_ INVALID	ERR_ TIMEOUT	ERR_ COMM	Res	ERR_ PARAM	ERR_ COMMAND	ERR_ OPCODE	Res

The bits of the error word are used as follows:

ERR_OPCODE

Illegal opcode. The VoiceDSP processor does not recognize the opcode.

ERR_COMMAND

Illegal command sequence. The command is not legal in the current state.

ERR_PARAM

Illegal parameter. The value of the parameter is out of range, or is not appropriate for the command.

ERR_COMM

Microcontroller MICROWIRE communication error.

ERR_TIMEOUT

Time-out error. Depending on the VoiceDSP processor's state, more than 100 milliseconds elapsed between the arrival of two consecutive bytes (for commands that have parameters).

ERR_INVALID

Command can not be performed in current context.

ERR_CID

Error during CID detection.

Example

GEW				
Byte sequence:	Microcontroller	1B	AA	AA
	VoiceDSP	1B	00	02
Description:	Get the VoiceDSP processor error word (typically sent after GSW when EV_ERROR is reported in the status word). The VoiceDSP processor responds: ERR_OPCODE:			

GI Get Information *item*

Returns the 16-bit value specified by *item* from one of the internal registers of the VoiceDSP processor.

item may be one of the following:

0 The duration of the last detected DTMF tone, in 10 ms units. The return value is meaningful only if DTMF detection is enabled, and the status word shows that a DTMF tone was detected.

1 The duration of the last detected busy tone in 10 ms units.

2 The energy level of the samples in the last 10 ms.

3 The energy level of the samples, in the last 10 ms, that are in the frequency range described in Figure 2-1 on page 2-12. The return value is meaningful only if one of the tone detectors is enabled (bits 0,1 of the detectors mask; see the description of SDET command).

The return value is unpredictable for any other value of *item*.

Example

GI 0					
Byte sequence:	Microcontroller	25	00	AA	AA
	VoiceDSP	25	00	00	06
Description:	Get the duration of the last detected DTMF tone. The VoiceDSP processor responds: 60 ms.				

GL Get Length

Returns the length of the current message in multiples of 4 Kbytes (blocks).

The returned value includes the message directory information (64 bytes for the first block and 32 bytes for every other block), the message data, and the entire last block of the message, even if the message occupies only a portion of the last block. Since a memory block includes 4096 bytes, the returned length

may be bigger than the actual message length by up to 4095 bytes. The minimum length of a message is one block.

Example

GL				
Byte sequence:	Microcontroller	19	AA	AA
	VoiceDSP	19	00	04
Description:	Get the length of the current message. The VoiceDSP processor responds: 4 i.e., the message occupies 16384 (4 * 4096) bytes.			

GMS Get Memory Status *type*

Returns the total remaining memory blocks as a 16bit unsigned integer. The parameter *type* must be 0.

The estimated remaining recording time may be calculated as follows:

Time = (Num_of_blocks x 4096 x 8) / (Compression_rate x 1000)

This estimate assumes no silence compression: a real recording may be longer, according to the amount of silence detected and compressed.

Example

GMS 00					
Byte sequence:	Microcontroller	12	00	AA	AA
	VoiceDSP	12	00	00	28
Description:	Return the remaining memory blocks. The VoiceDSP responds: 40 blocks.				

GMT Get Message Tag

Returns the 16-bit tag associated with the current message. If the current message is undefined, ERR_INVALID is reported.

Example

GMT				
Byte sequence:	Microcontroller	04	AA	AA
	VoiceDSP	04	00	0E
Description:	Get the current message tag. In a system where the message tag is encoded as described in the DMS command, the VoiceDSP processor return value indicates that the message is a new ICM in mailbox Number 6.			

GNM Get Number of Messages *tag_ref tag_mask*

Returns the number of messages whose message tags match the *tag_ref* parameter. Only bits set in *tag_mask* are compared, a match is considered successful if:

message tag and *tag_mask* = *tag_ref* and *tag_mask*

where and is a bitwise AND operation.

The *tag_ref* and *tag_mask* parameters are each two bytes; the return value is also two bytes long.

See "Message Tag" on page 2-3 for a description of message-tag encoding. If *tag_mask* = 0, the total number of all existing messages is returned, regardless of the *tag_ref* value.

Example

GNM FFFE 0003								
Byte sequence:	Microcontroller	11	FF	FE	00	03	AA	AA
	VoiceDSP	11	FF	FE	00	03	00	05
Description:	Get the number of messages which have bit 0 cleared, and bit 1 set, in their message tags. VoiceDSP processor responds that there are five messages which satisfy the request.							

GSW Get Status Word

Returns the 2-byte status word.

STATUS WORD

The VoiceDSP processor has a 16-bit status word to indicate events that occur during normal operation. The VoiceDSP processor asserts the $\overline{\text{MWRQST}}$ signal (clears to 0), to indicate a change in the status word. This signal remains active until the VoiceDSP processor receives a GSW command.

The status word is cleared during reset, and upon a successful GSW command.

15	14	13	12	11	10	9	8	7	6	5	4	3 0
EV_ DTMF	EV_ RESET	EV_ VOX	EV_ CONST_ NRG	EV_ CAS	EV_ MEMLOW	EV_ DIALTONE	EV_ BUSY	EV_ ERROR	EV_ MEMFULL	EV_ NORMAL_ END	EV_ DTMF_ END	EV_ DTMF_ DIGIT

The bits in the status word are used as follows:

EV_DTMF_DIGIT

DTMF digit. A value indicating a detected DTMF digit. (See the description of DTMF code in the GT command.)

EV_DTMF_END

1 = Ended detection of a DTMF tone. The detected digit is held in EV_DTMF_DIGIT.

EV_NORMAL_END

1 = Normal completion of operation, e.g., end of message playback.

EV_MEMFULL

1 = Memory is full.

EV_ERROR

1 = Error detected in the last command. You must issue the GEW command to return the error code and clear the error condition.

EV_BUSY

1 = Busy tone detected. Use this indicator for call progress and line disconnection.

EV_DIALTONE

1 = Dial tone detected. Use this indicator for call progress and line disconnection.

EV_MEMLOW

1 = Not enough memory. (See CMSG command for further details.)

EV_CAS

1 = CAS tone detected. Use this signal for CIDCW.

EV_CONST_NRG

1 = A period of constant energy was detected. Use this indicator for line disconnection. (See CONST_NRG_DET_TIME_COUNT in Table 2-8.)

EV_VOX

1 = A period of silence (no energy) was detected on the telephone line. Use this indicator for line disconnection. (See VOX_DET_TIME_COUNT in Table 2-8.)

EV_RESET

When the VoiceDSP processor completes its power-up sequence and enters the RESET state, this bit is set to 1, and the $\overline{\text{MWRQST}}$ signal is activated (cleared to 0). Normally, this bit changes to 0 after performing the INIT command. If this bit is set during normal operation of the VoiceDSP processor, it indicates an internal VoiceDSP processor error. The microcontroller can recover from such an error by re-initializing the system.

EV_DTMF

1 = Started detection of a DTMF tone.

Example

GSW				
Byte sequence:	Microcontroller	14	AA	AA
	VoiceDSP	14	00	40
Description:	Get the VoiceDSP processor Status Word (typically sent after the $\overline{\text{MMRQST}}$ signal is asserted by the VoiceDSP processor which indicates a change in the status word). The VoiceDSP processor responds that the memory is full.			

GT Generate Tone *tone*

Generates the tone specified by the 1-byte tone parameter. The VoiceDSP state changes to TONE_GENERATE. The tone generation continues until an S command is received.

A DTMF or a single frequency tone may be generated as shown:

To generate a DTMF encode the bits as follows:

Bit 0

1

Bits 1–4

DTMF code.

Where the DTMF code is encoded as follows:

Value (Hex)	DTMF Digit
0 to 9	0 to 9
A	A
B	*
C	#
D	B
E	C
F	D

Bits 5–7

0

To generate a single frequency tone encode the bits as follows:

Bit 0

0

Bits 1-5

3–30

The value in bits 1–5 is multiplied by 100 to generate the required frequency (300Hz–3000Hz).

Bits 6-7

0

The VoiceDSP processor does not check for the validity of the tone specification. Invalid specification yields unpredictable results.

Example

GT 20			
Byte sequence:	Microcontroller	0D	20
	VoiceDSP	0D	20
Description:	Generate a single-frequency 1600Hz tone.		

GTD Get Time and Day *time_day_option*

Returns the time and day as a 2-byte value. *time_day_option* may be one of the following:

0 Get the system time and day.

1 Get the current message time-and-day stamp.

Any other *time_day_option* returns the time-and-day stamp of the current message.

Time and day are encoded as follows:

Bits 0–2

Day of the week (1 through 7).

Bits 3–7

Hour of the day (0 through 23).

Bits 8–13

Minute of the hour (0 through 59).

Bits 14–15

00 The time was not set before the current message was recorded.

11 The time was set, i.e., the SETD (Set Time of Day) command was executed.

NOTE If the current message is undefined, and time_day_option is 1, an ERR_INVALID error is reported.

Example

GTD 1					
Byte sequence:	Microcontroller	0E	01	AA	AA
	VoiceDSP	0E	01	E8	29
Description:	Get the current message time-and-day stamp. The VoiceDSP processor responds that the message was created on the first day of the week at 5:40 A.M. The return value also indicates that the SETD command was used to set the system time and day before the message was recorded. *Note: If the SAS command is used to announce the time-and-day stamp, "Monday" is announced as the first day of the week. For an external vocabulary, the announcement depends on the vocabulary definition (See the* IVS User's Manual *for more details).*				

GTM Get Tagged Message *tag_ref tag_mask dir*

Selects the current message, according to instructions in *dir*, to be the first, n^{th} next or n^{th} previous message which complies with the equation:

message tag and *tag_mask* = *tag_ref* and *tag_mask*.

where and is a bitwise AND operation.

dir is one of the following:

0 Selects the first (oldest) message.

-128 Selects the last (newest) message.

n Selects the n^{th} next message starting from the current message.

–n Selects the n^{th} previous message starting from the current message.

NOTE To select the n^{th}, or $-n^{th}$, message with a given tag to be the current message you must first select the first message (dir=0), or the last message (dir=-128), that complies with the above equation, and then issue another GTM command with n-1 (for next message), or -n+1 (for previous message), as a parameter, to skip to the n^{th}, or $-n^{th}$, message respectively.

If a message is found, it becomes the current message and 1 (TRUE) is returned. If no message is found, the current message remains unchanged and 0 (FALSE) is returned.

If dir is not 0, and the current message is undefined the return value is unpredictable. After the command execution the current message may either remain undefined or changed to any existing message. The only exception is when the GTM command is executed just after the DM command. (See the DM command for further details.)

To access the n^{th} message, when n > 127, a sequence of GTM commands is required.

Example

GTM FFCE 003F 00								
Byte sequence:	Microcontroller	09	FF	CE	00	3F	00	AA
	VoiceDSP	09	FF	CE	00	3F	00	01

Description:	Select the oldest of the new ICMs, in mailbox number 6, to be the current message, for a system where the message tag is encoded as described in the example for the DMS command.
	The VoiceDSP processor returns a value indicates that there is such a message.
	The following pseudo-code demonstrates how to play all new ICMs in mailbox number 6. The messages are marked as old after being played:
	
	Return_val = GTM(FFCE, 003F, 00) /*Get the oldest message with the defined tag*/
	While (Return_Val == TRUE)
	Begin
	P() /* Play */
	Message_tag = GMT() /* Get message tag */
	SMT(FFF7) /* Mark the message as 'old' */
	GTM(FFCE, 003F, 01) /* Get next message with the same tag */
	End							

GTUNE Get Tune *index*

Gets the value of the tunable parameter identified by *index* (one byte) as the 2-byte value, *parameter_value*. This command may be used to read and identify the parameter value that was set to tune the VoiceDSP.

If *index* does not point to a valid tunable parameter, ERR_PARAM is set in the error word.

This index parameter should be in a hexidecimal value.

The GTUNE command may be used in IDLE state or RESET state. If TUNE command was not used to set the tunable parameters, then the GTUNE command will read the default parameter value.

Example

GTUNE 17					
Byte sequence:	Microcontroller	06	17	AA	AA
	VoiceDSP	06	17	02	BC
Description:	Get the minimum period for busy detection ComactSPEECH responds: 700 (7 seconds).				

Table 2-4 through Table 2-13 describe the tunable parameters, their index numbers in decimal, and their default values.

INIT Initialize System

Execute this command after the VoiceDSP processor has been configured (see CFG command). INIT performs a soft reset of the VoiceDSP as follows:

- Initializes the message directory information.
- Messages are not deleted. To delete the messages, use the DM and DMS commands.
- Sets the detectors mask to 0.
- Sets the volume level that is controlled by the VC command, to 0.
- Sets the playback speed to normal (0).
- Switches to the IDLE state.
- Initializes the tone detectors.

The current message is undefined after INIT execution.

The tunable parameters are not affected by this command. They are set to their default values only during RESET.

Example

INIT		
Byte sequence:	Microcontroller	13
	VoiceDSP	13
Description:	Initialize the VoiceDSP processor.	

INJ Inject IVS Data *n byte*$_1$*... byte*$_n$

Injects vocabulary data of size *n* bytes to good Flash blocks.

This command programs Flash devices, on a production line, with IVS vocabulary data. It is optimized for speed; all VoiceDSP processor detectors are suspended during execution of the command. Use the CVOC command to check whether programming was successful.

If there is not enough memory space for the vocabulary data, ERR_PARAM is set in the error word, and execution stops.

Flash blocks that include IVS data can not be used for recording, even if only one byte of the block contains IVS data (e.g., if the vocabulary size is 4K + 100 bytes, two blocks of the Flash are not available for message recording).

Example

INJ 00000080 Data							
Byte sequence:	Microcontroller	29	00	00	00	80	Vocabulary Data
	VoiceDSP	29	00	00	00	80	Echo of Data
Description:	Inject 128 bytes of vocabulary data.						

MR Memory Reset

Erases all memory blocks and initializes the memory and message management. Bad blocks, and blocks which are used for IVS vocabularies, are not erased. This command can be issued in either RESET or IDLE states.

NOTE When Memory Reset is used in RESET state, it must be issued after the CFG command is issued, or the memory type and number of devices are not defined. In this case the result is unpredictable.

NOTE The command erases all messages and should be used with care.

Example

MR		
Byte sequence:	Microcontroller	2A
	VoiceDSP	2A
Description:	Erase all memory blocks.	

P Playback

Begins playback of the current message. The VoiceDSP processor state changes to PLAY. When playback is complete, the VoiceDSP processor sets the EV_NORMAL_END bit in the status word, and activates (clears to 0) the $\overline{\text{MWRQST}}$ signal. The state then changes to IDLE. Playback can be paused with the PA command, and can be resumed later with the RES command. Playback can be stopped with the S command.

If the current message is undefined, ERR_INVALID is reported.

Example

P		
Byte sequence:	Microcontroller	03
	VoiceDSP	03
Description:	Play the current message.	

PA Pause

Suspends the execution of the current GT, P, R, SAS, SO, SW, or SS commands. The PA command does not change the state of the VoiceDSP processor; execution can be resumed with the RES command.

NOTE DTMF and tone detectors remain active during Pause, Play and Record commands.

Example

PA		
Byte sequence:	Microcontroller	1C
	VoiceDSP	1C
Description:	Suspend playback of current message.	

PDM Go To Power-Down Mode

Switches the VoiceDSP processor to power-down mode (see "POWER-DOWN MODE" on page 1-4 for details). Sending any command while in power-down mode resets the processor detectors, and returns it to normal operation mode.

NOTE If an event report is pending ($\overline{MWRQST}$ is active) and not processed by the microcontroller prior to issuing the PDM command, the event is lost.

Example

PDM		
Byte sequence:	Microcontroller	1A
	VoiceDSP	1A
Description:	Put the VoiceDSP processor in power-down mode.	

R Record *tag compression_rate*

Records a new message with message tag *tag* and compression rate *compression_rate*. The VoiceDSP processor state changes to RECORD. The R command continues execution until stopped by the S command. Recording can be paused with the PA command, and can be resumed later with the RES command.

If the memory becomes full, recording stops and EV_MEMFULL is set in the status word.

See "Message Tag" on page 2-3 for a description of message-tag encoding.

The compression rate may be defined as 0 for PCM recording or either 1,2,3 for compression rates 4.7 Kbits/s, 6.7 Kbits/s, 8.7 Kbits/s respectively. See "VCD" on page 2-10 for a description of the compression algorithm.

NOTE A time-and-day stamp is automatically attached to each message. Before using the R command for the first time, use the SETD command. Failure to do so results in undefined values for the time-and-day stamp.

The CID_RECORD tunable parameter (index 51) may be used to attach the contents of the CID buffer to the message memory during recording.

To read CID information attached to the message, use the GCID command. See page 2-38 for details.

Example

R 000E 03					
Byte sequence:	Microcontroller	0C	00	0E	03
	VoiceDSP	0C	00	0E	03
Description:	Record a new ICM in mailbox Number 6 in a system where the message tag is encoded as described in the example of the DMS command. The compression rate is defined as 8.7 Kbit/s				

RDET Reset Detectors *detectors_reset_mask*

Resets the VoiceDSP processor tone and energy detectors according to the value of the detectors_reset_mask parameter. A bit set to 1 in the mask, resets the corresponding detector. A bit cleared to 0 is ignored.

The 1-byte *detectors_reset_mask* is encoded as follows:

Bit 0

Reset the busy and dial tone detectors.

Bits 1–3

Reserved. Must be cleared to 0.

Bit 4

Reset the constant energy detector.

Bit 5

Reset the no energy (VOX) detector.

Bit 6

Reset the DTMF detector.

Bit 7

Reserved. Must be cleared to 0.

Example

RDET 20			
Byte sequence:	Microcontroller	2C	20
	VoiceDSP	2C	20
Description:	Reset the VOX detector.		

RES Resume

Resumes the activity that was suspended by the PA, SB, SE or SF commands.

Example

RES		
Byte sequence:	Microcontroller	1D
	VoiceDSP	1D
Description:	Resume playback which was suspended by either the PA, SF or SB command.	

RMSG Read Message

Returns 32 bytes of data from the current position of the message pointer, and advances the message pointer by 32 bytes.

If the VoiceDSP processor was in the IDLE state, the command opens the current message, switches the VoiceDSP processor to the MSG_OPEN state, sets the message pointer to the beginning of the message data, and returns the 32 bytes of data.

The microcontroller must issue an S command to close the message, and switch the VoiceDSP processor to the IDLE state.

If the current message is undefined, ERR_INVALID is reported.

Trying to read beyond the end of the message sets the EV_NORMAL_END event, and the VoiceDSP processor switches to the IDLE state. In this case, the return value is undefined and should be ignored.

Example

RMSG					
Byte sequence:	Microcontroller	32	AA	AA	...
	VoiceDSP	32	32 bytes of data		
Description:	Read 32 bytes from the current message memory.				

S Stop

Stops execution of the current command and switches the VoiceDSP processor to the IDLE state. S may be used to stop the execution of CMSG, SMSG, WMSG, RMSG and all asynchronous commands. See Table 2-2 on page 2-28 for details.

This command may also be used to switch the VoiceDSP processor state to IDLE in order to issue commands that are allowed to execute only while in the IDLE state.

Example

S		
Byte sequence:	Microcontroller	00
	VoiceDSP	00
Description:	Stop current activity (e.g., playback, recording) and change the VoiceDSP processor to IDLE state.	

SAS Say Argumented Sentence *sentence_n arg*

Announces sentence number *sentence_n* of the currently selected vocabulary, and passes *arg*. *sentence_n* and *arg* are each 1-byte long. The VoiceDSP processor state changes to SYNTHESIS.

When playing is complete, the VoiceDSP processor sets the EV_NORMAL_END bit in the status word, and activates the $\overline{\text{MWRQST}}$ signal. The state then changes to IDLE.

If the current vocabulary is undefined, ERR_INVALID is reported.

Example

SAS 00 03				
Byte sequence:	Microcontroller	1E	00	03
	VoiceDSP	1E	00	03
Description:	Announce the first sentence in the sentence table of the currently selected vocabulary with '3' as the actual parameter.			

SB Skip Backward *time_length*

Skips backward in the current message *time_length* units, in 0.2 second segments, and pauses message playback. The RES command must be issued to continue playback. *time_length* is a 2-byte parameter that can have any value up to 320 (64 seconds). The skip accuracy is five percent. SB is meaningful only in the PLAY state.

If the beginning of the message is detected during the execution of the SB command, execution terminates, the EV_NORMAL_END bit in the status register sets, the $\overline{\text{MWRQST}}$ signal activates, and the processor switches to the IDLE state.

If *time_length* is greater than 320, ERR_PARAM is set in the error word.

Playback speed does not affect the behavior of this command.

Example

SB 0019				
Byte sequence:	Microcontroller	23	00	19
	VoiceDSP	23	00	19
Description:	Skip backwards five seconds from the current position in the message being played.			

SCID Save Caller ID Data *dest, offset, len, <data>*

Save the CID data into the Caller ID buffer, or as part of the current message. If no current message exists, error flag is set in the status word, and ERR_INVALID is set in the error word.

The *dest* value can be either 0 or 1:

0 Save the data into the Caller ID buffer

1 Save the data in the message directory

The *offset* value shows where to start saving in the destination buffer (‘*dest*)

The *len* value shows number of bytes to save

The *data* value is *len* bytes to be saved

Valid range: *len* is non-zero, 0 < *offset* + *len* <= 71

NOTE *If a message directory contains Caller ID data, that was saved as part of the R (Record) command, do not save Caller ID data using the SCID command.*

Example

SCID 00 00 30 <48-byte CIDdata>		
Byte sequence:	Microcontroller	35 00 00 30 <48-byte CID data>
	VoiceDSP	35 00 00 30 <48-byte CID data>
Description:	VoiceDSP stores 48 bytes of Caller ID data into the beginning of the Caller ID buffer.	

SDET Set Detectors Mask *detectors_mask*

Controls the reporting of detection of tones and energy detectors according to the value of the *detectors_mask* parameter. A bit set to 1 in the mask, enables the reporting of the corresponding detector. A bit cleared to 0 disables the reporting.

Disabling reporting of a detector does not stop or reset the detector.

The 1-byte *detectors_mask* is encoded as follows:

Bit 0

Report detection of a busy tone.

Bit 1

Report detection of a dial tone.

Bit 2

Reserved. Must be cleared to 0.

Bit 3

Report detection of CAS (CIDCW notification).

Bit 4

Report detection of a constant energy.

Bit 5

Report detection of no energy (VOX) on the line.

Bit 6
Report the ending of a detected DTMF.

Bit 7
Report the start of a detected DTMF (up to 40 ms after detection start).

Example

SDET BB			
Byte sequence:	Microcontroller	10	BB
	VoiceDSP	10	BB
Description:	Set reporting of all VoiceDSP processor detectors, except for end-of-DTMF.		

SE Skip to End of Message

This command is valid only in the PLAY state. When invoked, playback is suspended (as for the PA command), and a jump to the end of the message is performed. Playback remains suspended after the jump.

Example

SE		
Byte sequence:	Microcontroller	24
	VoiceDSP	24
Description:	Skip to end of current message.	

SETD Set Time and Day *time_and_day*

Sets the system time and day as specified by the 2-bytes *time_and_day* parameter. The *time_and_day* parameter is encoded as follows:

Bits 0–2
Day of the week (1 through 7).

Bits 3–7
Hour of the day (0 through 23).

Bits 8–13
Minute of the hour (0 through 59).

Bits 14–15
Must be set to 1.

If *time_and_day* value is not valid, ERR_PARAM is set in the error word.

Example

SETD DE09				
Byte sequence:	Microcontroller	0F	DE	09
	VoiceDSP	0F	DE	09
Description:	Set time and day to Monday 1.30 A.M. (where Monday is the first day of the week)			

SF Skip Forward *time_length*

Skips forward in the current message *time_length* units, in 0.2 second segments, and causes message playback to pause. The RES command must be issued to continue playback. *time_length* is a 2-byte parameter that can have any value up to 320 (64 seconds). The skip accuracy is five percent. This command is meaningful only in the PLAY state.

If the end of the message is detected during execution of SF, execution of the command terminates, the EV_NORMAL_END bit in the status word sets, the $\overline{\text{MWRQST}}$ signal activates, and the processor switches to the IDLE state.

If *time_length* is greater than 320, ERR_PARAM is set in the error word.

Playback speed does not affect the behavior of this command.

Example

SF 0019				
Byte sequence:	Microcontroller	22	00	19
	VoiceDSP	22	00	19
Description:	Skip forward five seconds from the current position in the message being played.			

SLC Set Line Channels *active_channels*

Configures the VoiceDSP to activate/deactivate channel 1. The activation of channel 1 is used to enable the conversation recording feature. When activating this feature, the VoiceDSP sums channel 0 and channel 1 during the record command and then compresses the sum. When playing the message, it is decompressed and broadcasted to the active channels as set by this command. Note that channel 0 is always active and is used as the default line channel.

Use the following parameter values to activate the channels:

Active Channel value	Channel 1
1	Non-active
3	Active

Example

SLC 03			
Byte sequence:	Microcontroller	0x38	03
	VoiceDSP	0x38	03
Description:	Enables both channel 0 and 1		

SMSG Set Message Pointer *num_of_pages*

Sets the message pointer to *num_of_pages* x 32 bytes from the beginning of the current message data.

If the VoiceDSP processor was in the IDLE state, the command opens the current message and switches the VoiceDSP processor to the MSG_OPEN state. The microcontroller must issue an S command to close the message, and switch the VoiceDSP processor to the IDLE state.

If *num_of_pages* x 32 is greater than the message length, EV_NORMAL_END is set in the status word, the message pointer is set to the end of the message, and the VoiceDSP processor switches to the IDLE state.

If the current message is undefined, ERR_INVALID is reported.

Example

SMSG 000A				
Byte sequence:	Microcontroller	30	00	0A
	VoiceDSP	30	00	0A
Description:	Set the message pointer to 10 pages (320 bytes) from the beginning of the current message data.			

SMT Set Message Tag *message_tag*

Sets the tag of the current message. The 2-byte *message_tag* can be used to implement mailbox functions by including the mailbox number in the tag, or to handle old and new messages differently by using one bit in the tag to mark the message as old or new. See "Message Tag" on page 2-3.

To change the message tag, you should first get the tag using the GMT command, read the tag, modify it, and write it back.

NOTE Message tag bits can only be cleared. Message tag bits are set only when a message is first created.

If the current message is undefined, ERR_INVALID is reported.

Example

SMT FFF7				
Byte sequence:	Microcontroller	05	FF	F7
	VoiceDSP	05	FF	F7
Description:	Mark the current message as old in a system where the message tag is encoded as described in the example of the DMS command. *Note that the VoiceDSP processor ignores bits in the tag which are set to 1; only bit 3 is modified in the message tag.*			

SO Say One Word *word_number*

Plays the word number *word_number* in the current vocabulary. The 1-byte *word_number* may be any value from 0 through the index of the last word in the vocabulary. The VoiceDSP processor state changes to SYNTHESIS.

When playback of the selected word has been completed, the VoiceDSP processor sets the EV_NORMAL_END bit in the status word, and activates the $\overline{\text{MWRQST}}$ signal. The state then changes to IDLE.

If *word_number* is not defined in the current vocabulary, or if it is an IVS control or option code, ERR_PARAM is set in the error word.

If the current vocabulary is undefined, ERR_INVALID is reported.

Example

SO 00			
Byte sequence:	Microcontroller	07	00
	VoiceDSP	07	00
Description:	Announce the first word in the word table of the currently selected vocabulary.		

SPS Set Playback Speed *speed*

Sets the speed of message playback as specified by speed. The new speed applies to all recorded messages and synthesized messages (only if synthesized using IVS), until changed by another SPS command. If this command is issued while the VoiceDSP processor is in the PLAY state, the speed also changes for the message currently being played.

Speed may be one of 13 values, from –6 to +6. A value of 0 represents normal speed. If *speed* is not in the –6 to +6 range, ERR_PARAM is set in the error word.

NOTE A negative speed value represents an increase in speed, a positive value represents a decrease in speed.

NOTE The playback speed control is not applicable when the stored messages or the IVS data are not compressed (Stored in PCM format).

The change in speed is approximate, and depends on the recorded data. In any case, if i < j, playback speed with parameter i is the same or faster than with parameter j.

Example

SPS FB			
Byte sequence:	Microcontroller	16	FB
	VoiceDSP	16	FB
Description:	Set playback speed to –5.		

SS Say Sentence *sentence_n*

Say sentence number *sentence_n* of the currently selected vocabulary. *sentence_n* is 1-byte long. The VoiceDSP processor state changes to SYNTHESIS.

If the sentence has an argument, 0 is passed as the value for this argument.

When playing has been completed, the VoiceDSP processor sets the EV_NORMAL_END bit in the status word, and activates the $\overline{\text{MWRQST}}$ signal. The state then changes to IDLE.

If *sentence_n* is not defined in the current vocabulary, ERR_PARAM is set in the error word.

If the current vocabulary is undefined, ERR_INVALID is reported.

Example

SS 00			
Byte sequence:	Microcontroller	1F	00
	VoiceDSP	1F	00
Description:	Announce the first sentence in the sentence table of the currently selected vocabulary.		

SSM Set Speakerphone Mode *mode*

Sets the speakerphone to the *mode* mode of operation. The command is valid when the VoiceDSP processor is in IDLE state. *mode* can be one of:

0	OFF	Deactivate the speakerphone, and return the VoiceDSP processor to normal operation mode.
1	ON	Put the VoiceDSP processor in speakerphone mode and activate speakerphone in full-duplex mode i.e., with full cancellation of both the acoustic and the electrical echoes. Tone detectors are not active. Gains in the Send and Receive paths are set by the relevant tunable parameters.
2	TRANSPARENT	Activate the speakerphone with no echo cancellation (this mode is used for system tuning).
3	MUTE	Activate the speakerphone, while generating silence to the line. The near-end-listener can hear the far-end-speaker, but not vice versa. Tone detectors are not active.
4	LISTEN	The line is audible on the speaker. Tone detectors are active. This mode is used for call generation.
5	Reserved.	
6	RESTART	Restart the current speakerphone mode. This mode differs from ON; it does not require full initialization of the speakerphone. It should be used to resume the speakerphone operation to adjust to an environment change (e.g., parallel pickup).
7	HOLD	Stop the codec interrupts. Neither side can hear each other.

8 SILENCE The speakerphone is disabled. Electric echo cancellation is active for the CAS detection (CIDCW).

See "Full-duplex Speakerphone" on page 2-14 for more details.

NOTE Only commands that are specified in Table 2-3, are active during all speakerphone modes (other than 0).

Example

SSM 01			
Byte sequence:	Microcontroller	2F	01
	VoiceDSP	2F	01
Description:	Put the VoiceDSP processor into Speakerphone mode, and set the speakerphone to full-duplex mode.		

SV Set Vocabulary Type *type id*

Selects the vocabulary table to be used for voice synthesis. The vocabulary type is set according to the 1-byte *type* parameter:

0 For compatibility only.

1 External vocabulary in ROM.

2 External vocabulary in Flash.

3-7 Reserved.

The host is responsible for selecting the current vocabulary, with SV command, before using an SAS, SO, SS or SW command. Each external vocabulary table has a unique id which is part of the vocabulary internal header (See the *IVS User's Guide* for more details). If type is 1 or 2, the VoiceDSP processor searches for the one byte *id* parameter in each vocabulary table header until a match is found.

If the *id* parameter does not point to a valid IVS vocabulary, ERR_PARAM is set in the error word.

Example

SV 02 03				
Byte sequence:	Microcontroller	20	02	03
	VoiceDSP	20	02	03
Description:	Select the vocabulary with vocabulary-id 3, which resides on a Flash, as the current vocabulary.			

SW Say Words *n word*$_1$ . . . *word*$_n$

Plays n words, indexed by *word*$_1$ to *word*$_n$ (≤ 9). The VoiceDSP processor state changes to SYNTHESIS.

On completion, the EV_NORMAL_END bit in the status word is set, and the $\overline{\text{MWRQST}}$ signal goes low. The state then changes to IDLE.

If one of the words is not defined in the current vocabulary, or if it is an IVS control or option code, or if n > 8, ERR_PARAM is reported.

If the current vocabulary is undefined, ERR_INVALID is reported.

Example

SW 02 00 00					
Byte sequence:	Microcontroller	21	02	00	00
	VoiceDSP	21	02	00	00
Description:	Announce the first word, in the word table of the currently selected vocabulary, twice.				

TUNE Tune *index parameter_value*

Sets the value of the tunable parameter identified by *index* (one byte) to the 2-byte value, *parameter_value*. This command may be used to tune the DSP algorithms to a specific Data Access Arrangement (DAA) interface, or to change other parameters. If you do not use TUNE, the VoiceDSP processor uses default values.

If *index* does not point to a valid tunable parameter, ERR_PARAM is set in the error word.

NOTE *The tunable parameters are assigned with their default values on application of power. The INIT command does not affect these parameters.*

The index parameter of this command should be in hexidecimal numbers while the index of the tunable parameter should be a decimal number.

Example

TUNE 17 02BC					
Byte sequence:	Microcontroller	15	17	02	BC
	VoiceDSP	15	17	02	BC
Description:	Set the minimum period for busy detection to 700 (7 seconds).				

The tables 2-4 through 2-13 describe the tunable parameters, their index numbers and their default values, grouped by their functionality.

Table 2-4: TUNABLE PARAMETERS: Software Automatic Gain Control

Index	Parameter Name	Description	Default
70	SWAGC enabling: AGC_enable_flag	Enables the SWAGC. Value 1 to enable, value 0 to disable. Different values of the tunable parameter can be used for the compression and Speakerphone algorithms. In order to change the value of the tunable parameter use the TUNE command prior to entering each of the two algorithms. Required value for speakerphone mode is 0 - disabling the SWAGC. Legal values: 0,1	1
86	Input threshold for gain modification: AGC_VAD_THR	Power threshold for Voice activity detection. Only signals stronger than the threshold are considered voice and are increased by the SWAGC algorithm. The default value is equivalent to –38dBm input. Each multiplication of the value by 2 adds approximately 3dB to the input level. Different values of the tunable parameter can be used for the compression and Speakerphone algorithms. In order to change the value of the tunable parameter use the TUNE command prior to entering each of the two algorithms. Legal values: 0 - 32767	4
87	SWAGC time scale operation: AGC_DECAY_COEFF	Determines the desired time scale for SWAGC algorithm operation. Increasing this parameter decreases the time scale, and the SWAGC reacts quicker to changes in input levels and can become more sensitive to fluctuations in signal power. Default value is equivalent to 5 sec per 12dB change. Different values of the tunable parameter can be used for the compression and Speakerphone algorithms. In order to change the value of the tunable parameter use the TUNE command prior to entering each of the two algorithms. Legal values: 0 - 32767	2048
85	SWAGC low power bound: AGC_POWER_LOW	Low bound of desired signal power. The SWAGC algorithm amplifies the gain whenever the output signal power decreases below AGC_POWER_LOW. The default value is equivalent to –17dBm output power. Each multiplication of the value by 2 adds approximately 3dB to the output power. Different values of the tunable parameter can be used for the compression and Speakerphone algorithms. In order to change the value of the tunable parameter use the TUNE command prior to entering each of the two algorithms. Legal values: 0 - 32767	512

Table 2-4: TUNABLE PARAMETERS: Software Automatic Gain Control

Index	Parameter Name	Description	Default
11	SWAGC High power bound: AGC_POWER_HIGH	High bound of desired signal power. The SWAGC algorithm attenuates the gain whenever the output signal power increases above AGC_POWER_HIGH. The default value is equivalent to −11dBm output power. Each multiplication of the value by 2 adds approximately 3dB to the output power. Different values of the tunable parameter can be used for the compression and Speakerphone algorithms. In order to change the value of the tunable parameter use the TUNE command prior to entering each of the two algorithms. Required value for speakerphone mode is 5000. This value should be given only if the SWAGC is enabled in this mode. Legal values: 0 - 16384	2048
88	SWAGC maximum gain step: AGC_MAX_GAIN	This tunable parameter controls the maximum gain of the SWAGC algorithm. Adding 1 to the tunable parameter increases the gain by 6dB. Default is equivalent to 12dB gain. Different values of the tunable parameter can be used for the compression and Speakerphone algorithms. In order to change the value of the tunable parameter use the TUNE command prior to entering each of the two algorithms. Legal values: 0 - 3	2

Table 2-5: TUNABLE PARAMETERS: Tone Generation and Message Playback

Index	Parameter Name	Description	Default
27	DTMF Generation: DTMF_GEN_TWIST_LEVEL	A one-byte value that controls the twist level of a DTMF tone, generated by the GT command, by controlling the energy level of each of the two tones (low frequency and high frequency) composing the DTMF tone. The Least Significant Nibble (LSN) controls the low tone and the Most Significant Nibble (MSN) controls the high tone. The energy level of each tone, as measured at the output of a TP3054 codec (before the DAA) connected to the VoiceDSP processor is summarized in the following table: *Nibble Value* — *Tone Energy (dB-Volts)* 0 — 0 1 — –17.8 2 — –14.3 3 — –12.9 4 — –12.4 5 — –12.0 6 — –11.9 7 — –11.85 8 - 15 — –11.85 The volume of the generated DTMF tone during measurements was 6. (TONE_GEN_LEVEL + *vol_level* = 6). *Note: the vol_level parameter is controlled by the VC command.* For the default level, the high tone is –14.3 dBV and the low tone is –12.4 dBV, which gives a DTMF twist level of 1.9 dB. The energy level of a single generated tone is the level of the low tone.	66
16	Tone Generation: TONE_GEN_LEVEL	Controls the energy level at which DTMF and other tones are generated. Each unit represents 3 dB. The default level is the reference level. For example, if you set this parameter to 4, the energy level is 6 dB less than the default level. The actual output level is the sum of TONE_GEN_LEVEL and the *vol_level* parameter, controlled by the VC command. The tones are distorted when the level is set too high. *Legal values: 0 ≤ TONE_GEN_LEVEL + vol_level ≤ 12.*	6
21	VCD Playback and Voice Synthesis: VCD_PLAY_LEVEL	Controls the energy during playback and external voice synthesis. Each unit represents 3 dB. The default level is the reference level. For example, if you set this parameter to 4, the energy level is 6 dB less than the default level. The actual output level is the sum of VCD_PLAY_LEVEL and the *vol_level* parameter (controlled by the VC command). Speech is distorted when the level is set too high. *Legal values: 0 ≤ VCD_PLAY_LEVEL + vol_level ≤ 12.*	6

Table 2-6: TUNABLE PARAMETERS: DTMF Detection

Index	Parameter Name	Description	Default
17	Energy Level: DTMF_DET_MIN_ENERGY	Minimum energy level at which DTMF tones are detected. If you divide (multiply) the value by 2, the detection sensitivity decreases (increases) by 3 dB. *Legal values: 8 to 4096*	32
24	Echo Canceler: DTMF_DET_ECHO_DELAY	The near-echo delay in samples. The sampling rate is 8000Hz (i.e., 125 ms per sample). *Legal values: 0 to 8.*	4
26	Twist Level: DTMF_DET_REV_TWIST	Controls the reverse twist level at which the VoiceDSP processor detects DTMF tones. While the normal twist is set at 8 dB, the reverse twist can be either 4 dB (default) or 8 dB (if this parameter is set to 1).	0
60	SW AGC: DTMF_DET_AGC_IDLE	SW AGC for DTMF in Idle/Record modes. When incrementing the tunable by 1, the dynamic range is increased by 3 dB. *Legal values: 0 to 5.*	0
61	SW AGC: DTMF_DET_AGC_PLAY	SW AGC for *PLAY* and *TONE_GENERATE* modes. When incrementing the tunable by 1, the dynamic range is increased by 3 dB. *Legal values: 0 to 5.*	3

Table 2-7: TUNABLE PARAMETERS: Tone Detection

Index	Parameter Name	Description	Default
18	Dial Tone: TONE_DET_TIME_COUNT	Controls the duration of a tone before it is reported as a dial tone, in 10 msec units. The accuracy of the constant is ±10 ms. *Legal values: 0 to 65535.*	700
19	Busy and Dial Tone: TONE_DET_ON_ENERGY_THRESHOLD	Minimum energy level at which busy and dial tones are detected as ON (after 700Hz filtering). If you divide (multiply) the value by 2 you get about 3 dB decrease (increase) in the threshold. The mapping between energy level and the parameter value is as follows (measured on the codec output when a 400Hz tone was injected to the codec input): *Tunable value* — *Energy threshold (dB-Volts)* 10 — –31.8 20 — –28.6 100 — –21.7 500 — –14.7 8000 — –2.5 *Legal values: 0 to 65535.*	160

Table 2-7: TUNABLE PARAMETERS: Tone Detection

Index	Parameter Name	Description	Default
20	Busy and Dial Tone: TONE_DET_OFF_ENERGY_ THRESHOLD	Maximum energy level at which busy and dial tones are detected as OFF (after 700Hz filtering). If you divide (multiply) the value by 2 you get about 3 dB decrease (increase) in the threshold. The mapping between energy level and the parameter value is the same as for TONE_DET_ON_ENERGY_THRESHOLD *Legal values: 0 to 65535.*	110
23	Busy Tone: BUSY_DET_MIN_TIME	Minimum time period for busy detection, in 10 ms units. The accuracy of the constant is ±10 ms. *Legal values: 0 to 65535.*	600
53	Busy Tone: BUSY_DET_MIN_ON_TIME	Minimum period considered as On period for busy tone detection. Note that for weak signals: (–30 dB and below) the maximum value is 12 (i.e., 120 ms minimum detection time). Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 10 to 1000.*	10
54	Busy Tone: BUSY_DET_MAX_ON_TIME	Maximum period considered as On for busy-tone detection. Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 10 to 1000.*	170
55	Busy Tone: BUSY_DET_MIN_OFF_TIME	Minimum period considered as Off for busy-tone detection. Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 5 to 1000.*	5
56	Busy Tone: BUSY_DET_MAX_OFF_TIME	Maximum period considered as On for busy-tone detection. Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 5 to 1000.*	130
57	Busy Tone: BUSY_DET_VERIFY_COUNT	Number of On/Off cadences that must be detected prior to reporting busy-tone presence. *Legal values: 9 to 127.*	9
58	Busy Tone: BUSY_DET_TONE_TYPE	Specifies the type of busy tone to detect: 1 —Two cadences 2 —Three cadences 3 —Both two and three cadences	1
59	Busy Tone: BUSY_DET_DIFF_THRESHOLD	The maximum allowed difference between two compared On or Off periods. Unit: 10 ms. *Legal values: 0 to 255.*	9

Table 2-8: TUNABLE PARAMETERS: Energy Detection

Index	Parameter Name	Description	Default
10	Silence (VOX): VOX_DET_ENERGY_THRESHOLD	This parameter determines the minimum energy level at which voice is detected. Below this level, it is interpreted as silence. If you divide (multiply) the value by 2, you get approximately a 3db decrease (increase) in the threshold. The mapping between energy level and the parameter value is as follows (measured on the codec output when a 400Hz tone was injected into the codec input): *Tunable value* — *Energy threshold (dB-Volts)* 10 — –45 20 — –42 40 — –39 80 — –36 160 — –33 320 — –30 640 — –27 1280 — –24 2560 — –21 *Legal values: 1 to 32767.*	12
12	Silence (VOX): VOX_DET_TIME_COUNT	This parameter, in units of 10 ms, determines the period of silence before the VoiceDSP processor reports silence. The accuracy of the constant is ±10 ms. *Legal values: 0 to 65535.*	700
22	Silence (VOX): VOX_DET_TOLERANCE_TIME	Controls the maximum energy-period, in 10 ms units, that does NOT reset the vox detector. *Legal values: 0 to 255.*	3
47	Constant Energy: CONST_NRG_DET_TIME_COUNT	Minimum elapsed time until the VoiceDSP processor reports constant energy level. Units: 10 ms. Accuracy: ±10 ms *Legal values: 1 to 65534*	700
48	Constant Energy: CONST_NRG_DET_TOLERANCE_TIME	Variations in constant energy, up to this time, do not reset the constant energy detector. Units: 10 ms. *Legal values: 0 to 255*	5
49	Constant Energy: CONST_NRG_DET_LOW_THRESHOLD	Determines the minimum energy level that is treated as constant energy. The minimum energy is calculated as follows: $(1—1/2^{CONST\_NRG\_DET\_LOW\_THRESHOLD}) * average\_energy$ *Legal values: 1 to 16*	1
50	Constant Energy: CONST_NRG_DET_HIGH_THRESHOLD	Determines the maximum energy level that is treated as constant energy. The maximum energy is calculated as follows: $(1 + 1/2^{CONST\_NRG\_DET\_HIGH\_THRESHOLD}) * average\_energy$ *Legal values: 0 to 16*	1

Table 2-9: TUNABLE PARAMETERS: Speakerphone

Index	Parameter Name	Description	Default
31	Acoustic Echo Canceler (AEC): SP_AEC_PRIORITY_BIAS	Controls the bias in priority between the Send and Receive paths. If send priority-bias is preferred, the value should be greater than zero. For no priority bias, the value should be zero. For priority bias for the Receive path, the value should be negative. Steps are 3 dB each (e.g., +3 is 9 dB bias for the Send path, –2 is 6 dB bias for the Receive path). *Legal values: –4 to 4.* *Note: this parameter is represented as a signed byte in the system. When reading its value with the GTUNE command, ignore the upper byte of the returned value.*	0
32	Acoustic Echo Canceler (AEC): SP_AEC_COUPLING_ LOSS_THRESHOLD	This parameter limits the acoustic return loss. Its value (SP_AEC_COUPLING_LOSS_THRESHOLD / 32767) is compared with the RMS value of S_{out} divided by the RMS value of R_{in}, during a single-talk event. The loop gain is decreased, if necessary, to control the TCL level. For SP_AEC_COUPLING_LOSS_THRESHOLD = 32767 this loop is disabled. *Legal values: 0 to 32767.*	2047
34	Acoustic Echo Canceler (AEC): SP_AEC_LR_LEVEL	Controls the speakerphone gain from the microphone to the line-out. The total attenuation, or gain, depends on both the analog gain and this value. The gain is: K * signal where: K = SP_AEC_LR_LEVEL/4096. *Legal values: 0 to 16000.*	14000
35	Electric Echo Canceler (EEC): SP_EEC_LR_LEVEL	Controls the speakerphone gain from the line-in to the speaker. The total attenuation, or gain, depends on both the analog gains and this value. The gain is: K * signal where: K = (SP_EEC_LR_LEVEL/4096) * $(2^{(6 + vol\_level)/2})$ *(vol_level is controlled by the VC command)* *Legal values: 0 to 400.*	281

Table 2-9: TUNABLE PARAMETERS: Speakerphone

Index	Parameter Name	Description	Default
36	Acoustic Echo Canceler (AEC): SP_AEC_CLIP_POS	Specifies the positive peak-value at which the analog circuit of the speaker saturates. Codec analog full scale corresponds to μLAW full scale values after expansion. Assume that positive saturation occurs at amplitudes higher than those of a sine wave at X [dBm0]. The SP_AEC_CLIP_POS value is set as: SP_AEC_CLIP_POS = $32636 * 10^{((X-3.17)/20)}$ *Note: a sine wave with amplitude 4 * 8159 = 32636 corresponds to 3.17 dBm0.* Example: For X = –6.2761 dBm0, the value is: SP_AEC_CLIP_POS = $32636 * 10^{((-6.2761-3.17)/20)} = 0.3371 * 32636 = 11000$; *Legal values: 0 to 32767.*	16000
37	Acoustic Echo Canceler (AEC): SP_AEC_CLIP_NEG	Specifies the negative peak value at which the analog circuit of the speaker saturates. Codec analog full scale corresponds to μLAW full scale values after expansion.The value of SP_AEC_CLIP_NEG is set as shown for SP_AEC_CLIP_POS, above. *Legal values: –32768 to 0.*	–16000
38	Electric Echo Canceler (EEC): SP_EEC_CLIP_POS	Specifies the positive peak value at which the analog circuit of the line-out saturates. Codec analog full scale corresponds to μLAW full scale values after expansion. The value of SP_EEC_CLIP_POS is set as shown for SP_AEC_CLIP_POS, above. *Legal values: 0 to 32767.*	16000
39	Electric Echo Canceler (EEC): SP_EEC_CLIP_NEG	Specifies the negative peak value at which the analog circuit of the line-out saturates. Codec analog full scale corresponds to μLAW full scale values after expansion. The value of SP_EEC_CLIP_NEG is set as shown for SP_AEC_CLIP_POS, above. *Legal values: –32768 to 0.*	–16000
43	Acoustic Echo Canceler (AEC): SP_AEC_VOX_HYST	Controls the hysteresis in near-talker detection. (The speakerphone state machine has a built-in hysteresis mechanism to prevent fluctuations in the talker identification process i.e., identifying the active side.) The value of this parameter is a dimensionless number, which should be evaluated during the tuning process for specific hardware. Larger values for the parameter correspond to a wider hysteresis loop. Negative values increase the probability that the state machine remains in the last state. *Legal values: –127 to 127.* *Note: this parameter is represented as a signed byte in the system. When reading its value with the GTUNE command, ignore the upper byte of the returned value.*	10

Table 2-9: TUNABLE PARAMETERS: Speakerphone

Index	Parameter Name	Description	Default
44	Electric Echo Canceler (EEC): SP_EEC_VOX_HYST	Controls the hysteresis in far-talker detection. (The speakerphone state machine has a built-in hysteresis mechanism to prevent fluctuations in the talker identification process i.e., identifying the active side.) The value of this parameter is a dimensionless number, which should be evaluated during the tuning process for specific hardware. Larger values for the parameter correspond to a wider hysteresis loop. Negative values increase the probability that the state machine remains in the last state. *Legal values: –127 to 127.* *Note: this parameter is represented as a signed byte in the system. When reading its value with the GTUNE command, ignore the upper byte of the returned value.*	10
45	Acoustic Echo Canceler (AEC): SP_AEC_DTD_TH	Controls the sensitivity of the system. Low values correspond to high sensitivity, with a greater false alarm probability (i.e., an echo is considered a real talker). High values correspond to low sensitivity, with slower switching. This parameter is affected by the loop gain and the specific hardware characteristics. *Legal values: 0 to 127.*	73
46	Electric Echo Canceler (EEC): SP_EEC_DTD_TH	Controls the sensitivity of the system. Low values correspond to high sensitivity, with a greater false alarm probability (i.e., an echo is considered a real talker). High values correspond to low sensitivity, with slower switching. This parameter is affected by the loop gain and the specific hardware characteristics. *Legal values: 0 to 127.*	82
33	Attenuation: SP_BLOCK_LEVEL	Controls the maximum attenuation level of the speakerphone suppressors. It affects the speakerphone stability and its subjective quality. The maximum attenuation is calculated according to: $SP\_BLOCK\_LEVEL/2^{28}$ *Legal values: 550 to 32000.*	10922
42	Tone Generation: SP_TONE_GEN_ LEVEL	Controls the energy level at which DTMF, and other tones, are generated to the line (codec 0) while the speakerphone is active. Each unit represents 3 dB. *Legal values: 0 to 10.* *Note: the energy level at which the tones are generated to the speaker (codec 1) while the speakerphone is active, is controlled by the TONE_GEN_LEVEL tunable parameter and the vol_level.* *Note: vol_level is controlled by the VC command.*	6

NOTE: *Refer to the Software Automatic Gain Control tunable parameters, listed in Table 2-4, for more details on speakerphone-related tunable parameters.*

Table 2-10: TUNABLE PARAMETERS: Memory Support

Index	Parameter Name	Description	Default
62	Memory Device Size: NUM_OF_BLOCKS_IN_MEM	Defines the number of blocks (each block is of 4096 bytes) in every Flash memory device. The number and type of connected devices are defined by the CFG command. *Flash Device Size (Mbits)* — *Number of Blocks Value* 4 — 128 8 — 256 16 — 512 *Note: this parameter must be tuned before the CFG command.*	128
63	Memory Size for Testing NUM_OF_BLOCKS_FOR_TEST	Defines the number of blocks (each block is of 4096 bytes) in every Flash memory device for production line testing purposes. The number should be small to minimize testing time during the production sequence. However, the number of blocks should be larger than the number of expected bad blocks in the memory device. In case of value=0, no production test is performed. In any case other than value= 0, the number of blocks is defined by the parameter value, and a production testing cycle is performed after RESET. *Legal values: 0 to the value of tune 62.* *Note: If power fails during production testing cycle, the memory status is unpredicted. The memory device should be replaced and the production test should be repeated.* *Note: this parameter must be tuned before the INIT command.*	0

Table 2-11: TUNABLE PARAMETERS: Codec Support (Samples)

Index	Parameter Name	Description	Default
65	Channel 0 Delay: CFRD0	*The delay of codec channel 0 from Frame Synch 0 (CFS0) to start of valid data.* Legal values: 0 to 255	1
66	Channel 1 Delay: CFRD1	*The delay of codec channel 1 from Frame Synch 0 (CFS0) to start of valid data.* Legal values: 0 to 255	9
67	Channel 2 Delay: CFRD2	*The delay of codec channel 2 from Frame Synch 0 (CFS0) to start of valid data.* Legal values: 0 to 255	17

Table 2-11: TUNABLE PARAMETERS: Codec Support (Samples)

Index	Parameter Name	Description	Default
68	Frame Synch Delay: CFSD	*The delay of Frame Synch 1 (CFS1) from Frame Synch 0 (CFS0).* Legal values: 0 to 255	16
69	Data Valid Delay: CFET	*The delay between Frame Synch 0 (CFS0) to end of valid data of all channels.* Legal values: 0 to 255	25

Table 2-12: TUNABLE PARAMETERS: Clocking

Index	Parameter Name	Description	Default
79	PLL_Configuration: PLL_MODE	*Controls the internal clock PLL multiplication factor, which generates the master system clock. The default value (0) allows using external crystal (or oscillator) of 4.096Mhz. Setting this value to 1 allows using external crystal (or oscillator) of 65.536Mhz.* Legal values: 0, 1	0

Table 2-13: TUNABLE PARAMETERS: Caller ID

Index	Parameter Name	Description	Default
0	CID_TIMEOUT0	CID Type I: Time-out for Caller ID detection, after activation of the ACID command, in 10 ms units. Accuracy is +/- 10 ms. If Caller ID detection is not completed within this period, ERR_CID is reported. If the value is 0, there is no check for time-out. *Legal values:0 to 65535*	0

Table 2-13: TUNABLE PARAMETERS: Caller ID

Index	Parameter Name	Description	Default
1	CID_TIMEOUT1	CID Type I: For CID physical layer Bell202 or V.23: Time-out for seizure signal detection, after activation of the ACID command, in 10 ms units. Accuracy is +/- 10 ms. Compute the value as follows: · The maximum time, as defined in the standard, less the time between end of first ring and activation of the ACID command, plus 20 ms. · French CID: The value should be 152, assuming zero delay between the end of the first ring and ACID activation. · Spanish CID: Recommended value is 204 (2 s standard recommendation + 40 ms for CID modem delay). · The default value is for the Bellcore CID where the time-out is not defined. · For CID physical layer DTMF: Maximum time for starting code detection (see CID_START_END_CODES), after activation of the ACID command, in 10 ms units. · Dutch CID: The value should be 80. If this time-out elapses, ERR_CID is reported. If the value is 0, there is no check for time-out. *Legal values: 0 to 65535*	0
2	CID_TIMEOUT2	CID Type I: For CID physical layer Bell202 or V.23: Minimum length of the seizure signal in 10 ms units. Accuracy is +/- 10 ms. French CID: The value should be 8. The default value is for the Bellcore and Spanish CID, where the time-out is not defined. For CID physical layer DTMF: Maximum time between two consecutive DTMF codes. Dutch CID: The value should be 50. If this time-out elapses, ERR_CID is reported. If the value is 0, there is no check for time-out. *Legal values: 0 to 65535*	0

Table 2-13: TUNABLE PARAMETERS: Caller ID

Index	Parameter Name	Description	Default
3	CID_TIMEOUT3	CID Type I: For CID physical layer Bell202 or V.23: Time-out for message type detection after valid seizure signal is 10 ms units. Accuracy is +/- 10 ms. French Caller ID: the value should be 46. The default value is for the Bellcore and Spanish CID where the time-out is not defined. If this time-out elapses, ERR_CID is reported. If the value is 0, there is no check for time-out. *Legal values: 0 to 65535* For CID with DTMF physical layer: Defines the maximum number of DTMF codes (digits) in the message body that is supported by the standard. Dutch CID: the value should be 17. Spanish CID: the value should be 42. No error is reported if the number of codes exceeds the parameter value. However, the field overflow bit in the status byte of the CID data will be set. (See the GCID command for more details). *Legal values: 0 to 46*	0
28	CID_RECEIVE_SENSITIVITY	CID Type I and Type II: If the CID physical layer is Bell202 or v.23: defines the minimum energy level, in dBm, on the codec input, above which the CID signal is detected. *Legal values: -26 to -38* *Note: this parameter is represented as a signed byte in the system. When reading its value with the GTUNE command, ignore the upper byte of the returned value.*	-36
30	CID_START_END_CODES	CID Type I: If the CID physical layer is DTMF: Defines the starting DTMF code (high nibble) and ending DTMF code (low nibble). The default value follows the Dutch CID specification, which defines the 'D' DTMF as the starting code and the 'C' DTMF as the ending code of the CID data.	254
51	CID_RECORD	CID Type I: 1 – Copy CID buffer to message memory during recording. 0 – Do not copy CID buffer.	1

Table 2-13: TUNABLE PARAMETERS: Caller ID

Index	Parameter Name	Description	Default
52	CID_PARAM4	CID Type I: Time-out for minimum MARK length in 10ms units. Accuracy allowed in 10ms is +/- 10ms. Recommended value for Spanish CID is 3. *Legal values: 0 to 65535*	0
71	CID_ACK_TYPE	CID Type II: The kind of DTMF that is generated as ACK. The values should be decoded according to the GT command. The default value (31) stands for the "D" DTMF. *Legal values: 1 to 31, only odd numbers*	31
72	CID_ACK_LEN	CID Type II: The length of the ACK signal generated, in 10 ms units. Accuracy allowed in 10 ms is +/- 5 ms. Note that the actual length of the signal is 1 less than the value of the tunable. The default value (7) stands for 60ms. *Legal values: 0 to 255*	7
84	CID_ACK_VOLUME	CID Type II: The volume of the DTMF that is generated as the ACK signal. Each unit represents 3dB. The default level is the reference level. For more details, refer to tunable index 16 in Table 2-5. *Legal values: $0 \leq CID\_ACK\_VOLUME + vol\_level \leq 12$*	6
73	CID_PARAM5	CID Type II: Time-out for MARK detection (for start of CID data), after the ACK signal ended, in 10 ms units [Equivalent to CID_TIMEOUT1 of Type I]. The default value (54) stands for 500ms. *Legal values: 0 to 65535*	0
74	CID_PARAM6	CID Type II: Length of CID data, in 10 ms units [Equivalent to CID_TIMEOUT0 of Type I]. *Legal values: 0 to 65535*	0
75	CID_PARAM7	CID Type II: Time-out for minimum MARK length, in 10 ms units. Accuracy allowed in 10 ms is +/- 10 ms [Equivalent to CID_PARAM4 of Type I]. *Legal values: 0 to 65535*	0
76	CID_CAS_THRESH	CID Type II: The relative threshold between level of input signal and level of the input signal on the CAS tone band. Increase in the value of this parameter causes increase in rejection as CAS detection. *Legal values: 0 to 32767*	1800

Table 2-13: TUNABLE PARAMETERS: Caller ID

Index	Parameter Name	Description	Default
77	CID_CAS_THRESH_MIN	CID Type II: The lowest tone's level which can be accepted as CAS. *Legal values: 0 to 32767*	180
78	CID_CAS_THRESH_MAX	CID Type II: The highest tone's level which can be accepted as CAS. *Legal values: 0 to 32767*	2600

VC Volume Control *vol_level*

Controls the energy level of all the output generators (playback, tone generation, voice synthesis and speakerphone), with one command. The resolution is ±3 dB.

The actual output level is composed of the tunable level variable, plus the *vol_level*. The valid range for the actual output level of each output generator is defined in Table 2-5.

For example, if the tunable variable VCD_PLAY_LEVEL (parameter number 21) is 6, and *vol_level* is –2, then the output level equals VCD_PLAY_LEVEL + *vol_level* = 4.

Example

VC 04			
Byte sequence:	Microcontroller	28	04
	VoiceDSP	28	04
Description:	Set the volume level to VCD_PLAY_LEVEL + 4.		

WMSG Write Message *data*

Writes 32 bytes of data to the current position of the message pointer, and advances the message pointer by 32 bytes.

If the VoiceDSP processor is in IDLE state, the command opens the current message, switches the VoiceDSP processor to MSG_OPEN state, sets the message pointer to the beginning of the message data, and writes the 32 bytes of data.

To add data at the end of an existing message, issue the SMSG command to the last page of the message. Issue the WMSG command with a buffer consisting of 32 FF bytes (this has no effect on the current data in the page). A subsequent WMSG command adds a new block to the message, and writing continues at the beginning of the new block.

The microcontroller must issue an S command to close the message and switch the VoiceDSP processor to IDLE state.

If the current message is undefined, ERR_INVALID is reported.

NOTE When updating an existing message, bits can only be cleared, but not set.

Example

WMSG 32 bytes			
Byte sequence:	Microcontroller	31	32 bytes of data to write
	VoiceDSP	31	echo 32 bytes of data
Description:	Write 32 bytes in the message memory.		

Chapter 3—SCHEMATIC DIAGRAMS

3.1 APPLICATION INFORMATION

The following schematic diagrams are extracted from a VoiceDSP demo unit, based on the ISD-ES360 board.

The demo includes three basic clusters:

- COP888EEG Microcontroller
- VoiceDSP processor cluster working with 3.3v, including two TP3054 codecs and an ISD-T360SA controlling Samsung 16Mb NAND flash.
- User interface that includes a 12-key (4x3) keypad.

For more details about the demo, please refer to the ISD-ES360 Demo Operating Instructions.

Figure 3-1: VoiceDSP ISD-T360SA

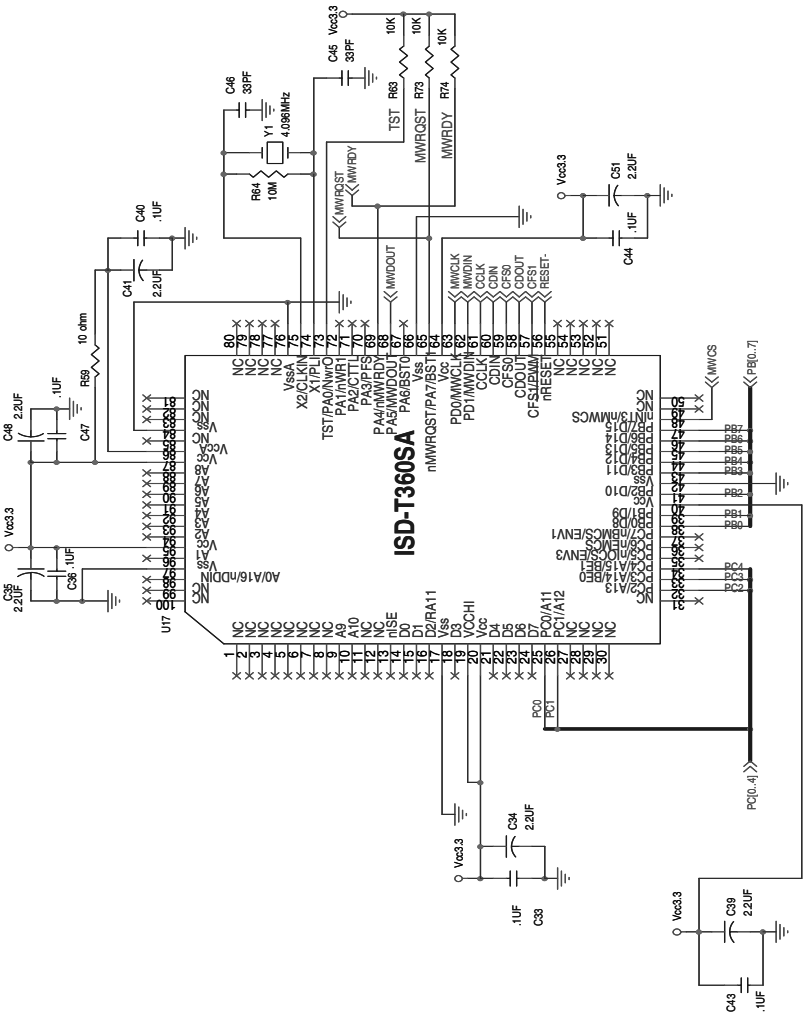


Figure 3-2: Nand Flash Device

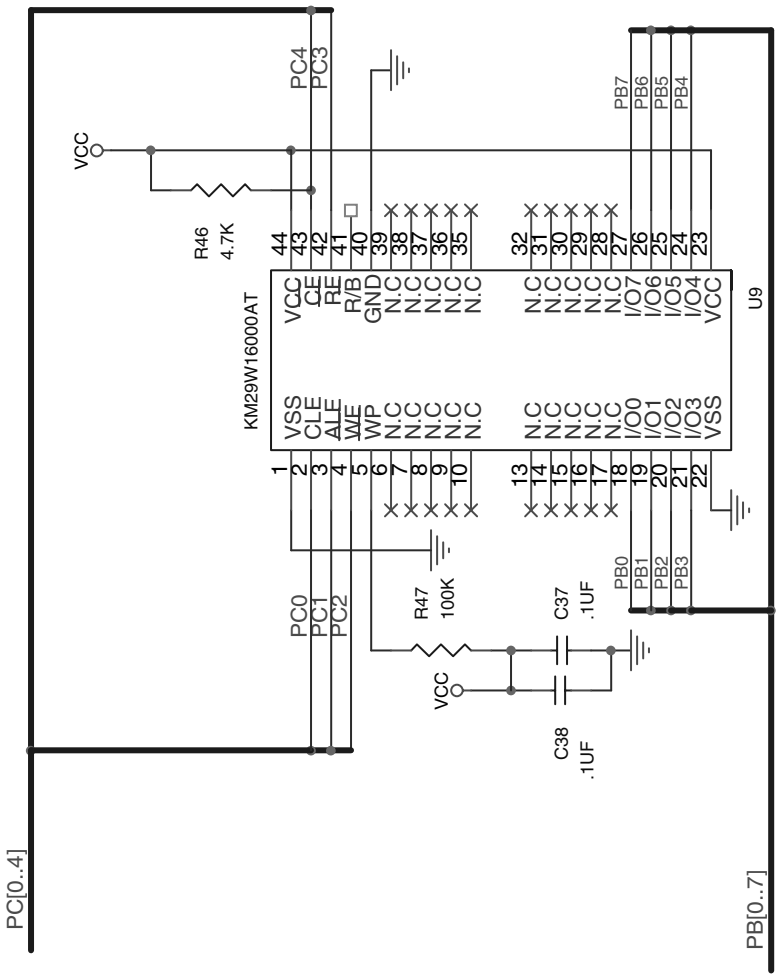


Figure 3-3: Line and Speakerphone Codec

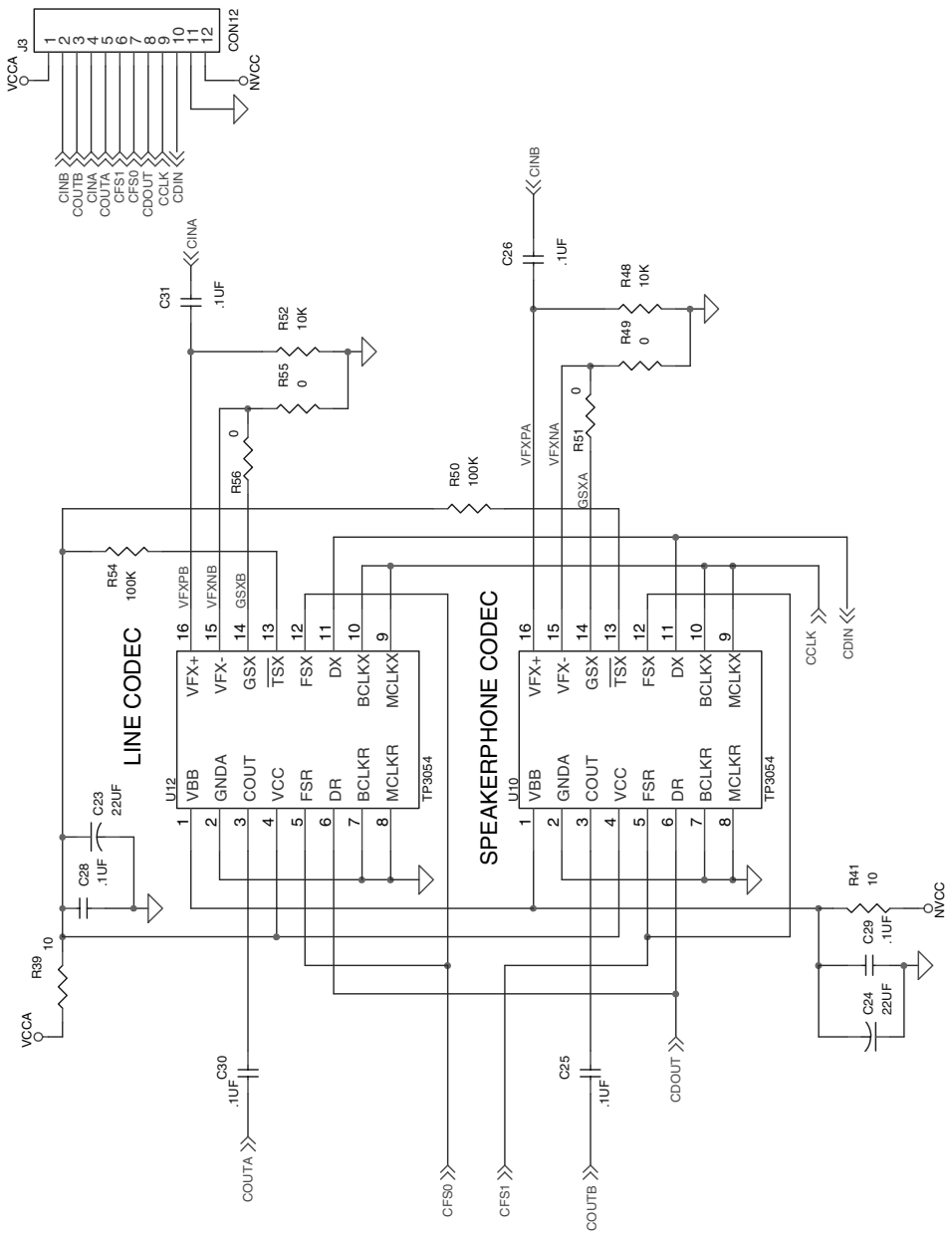
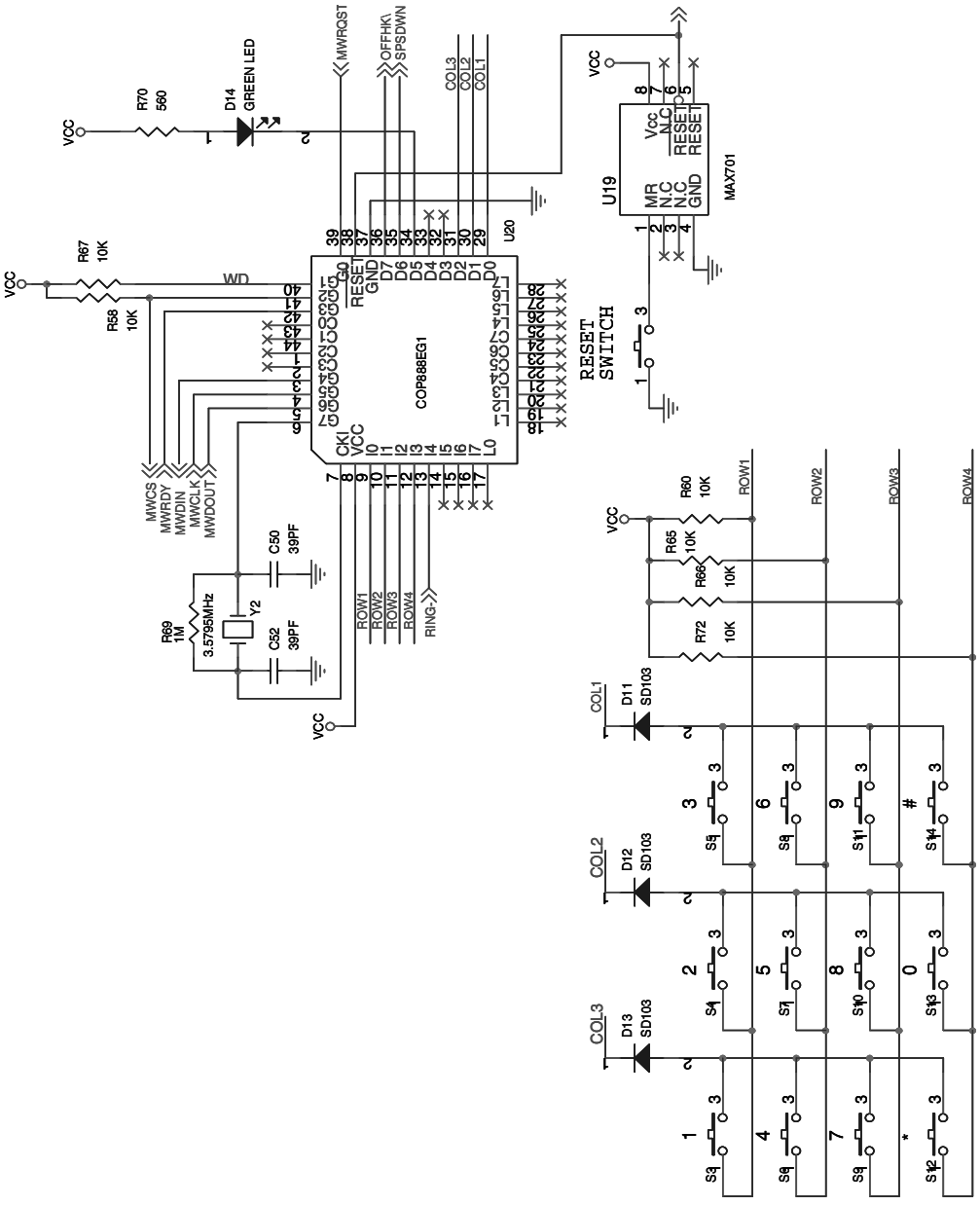
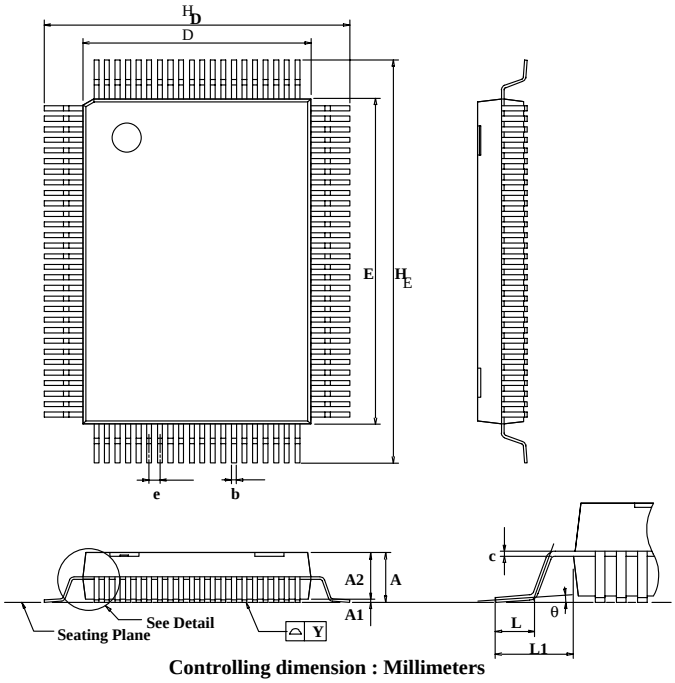


Figure 3-4: Microcontroller and User Interface



Chapter 4—PHYSICAL DIMENSIONS

Figure 4-1: 100L PQFP (14x20x2.75mm footprint 3.2mm) IQC



Controlling dimension : Millimeters

Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	0.130	—	—	3.30
A1	0.010	0.014	0.018	0.25	0.35	0.45
A2	0.101	0.107	0.113	2.57	2.72	2.87
b	0.008	0.012	0.016	0.20	0.30	0.40
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.547	0.551	0.555	13.90	14.00	14.10
E	0.783	0.787	0.791	19.90	20.00	20.10
e	0.020	0.026	0.032	0.50	0.65	0.80
H_D	0.669	0.677	0.685	17.00	17.20	17.40
H_E	0.905	0.913	0.921	23.00	23.20	23.40
L	0.025	0.031	0.037	0.65	0.80	0.95
L1	—	0.063	—	—	1.60	—
Y	—	—	0.003	—	—	0.08
θ	0°	—	7°	0°	—	7°