Table of Contents

ISD-T267SC CompactSPEECH™ Digital Speech Processor with Caller ID Support and Multiple Supplier Flash Support

Figures, Tables and Graphs in the ISD-T267SC Datasheet

CompactSPEECH™

ISD-T267SC CompactSPEECH™ Digital Speech Processor with Caller ID Support and Multiple Supplier Flash Support

GENERAL DESCRIPTION

The ISD-T267SC is a member of Information Storage Device's CompactSPEECH Digital Speech Processor family. This processor provides Digital Telephone Answering Device (DTAD) functionality to embedded systems.

The CompactSPEECH processor interfaces with Toshiba's TC58A040F and Samsung's KM29N040T Flash memory devices to provide a cost-effective solution for DTAD and integrated DTAD applications.

The CompactSPEECH processor integrates the functions of a traditional Digital Signal Processing (DSP) chip and the CR16A, a 16-bit general-purpose RISC core implementation of the CompactRISC™ architecture. It contains system support functions such as Interrupt Control Unit, Codec interface, MICROWIRE interfaces to a microcontroller and Flash memory, WATCHDOG timer, and a Clock Generator.

The CompactSPEECH processor operates as a slave peripheral that is controlled by an external microcontroller via a serial MICROWIRE interface. In a typical DTAD environment, the microcontroller controls the analog circuits, buttons and display, and activates the CompactSPEECH processor by sending it commands. The CompactSPEECH processor executes the commands and returns status information to the microcontroller.

The CompactSPEECH firmware implements voice compression and decompression, tone detection and generation, message storage management, speech synthesis for time-and-day stamp, and supports user-defined voice prompts in various languages.

The CompactSPEECH Caller ID feature complies with the Bellcore standard, used in the US, French, Spanish, Japanese and Dutch standards. It implements the receiver side for data transmitted from the central office to the subscriber.

The CompactSPEECH Processor applies echo-cancellation techniques to support high-quality DTMF tone detection during message playback.

The CompactSPEECH processor can synthesize messages in various languages via the International Vocabulary Support (IVS) mechanism. The ISD-T267SC can store vocabularies on either Flash memory or Expansion ROM memories. DTAD manufacturers can thus create machines that "speak" in different languages, simply by using other vocabularies. For more details about IVS, refer to the *IVS User's Manual*.

FEATURES

- Selectable speech compression rate of 5.2 Kbit/s and 7.3 Kbit/s, plus silence compression with each rate
- Up to 16 minutes recording on a 4-Mbit Flash memory devices (more than 1 hour total recording time on four devices)
- Automatic storage of Caller ID data of InComing Messages (ICM)
- Call screening (input signal echoed to codec output)
- Supports Samsung and Toshiba Flash memory devices
- Supports the US (Bellcore), French, Spanish, Japanese and Dutch Caller ID standards.
- Supports long-frame and short-frame codecs
- Interface to μ-Law codec
- Multi-lingual speech synthesis using International Vocabulary Support (IVS)
- Vocabularies available in: English, Japanese, Mandarin, German, French and Spanish
- Supports external vocabularies, using Flash memory devices or expansion ROM
- DTMF generation and detection
- Telephone line functions, including busy and dial tone detection
- Single tone generation
- DTMF detection during OutGoing Message playback
- MICROWIRE slave interface to an external microcontroller
- Supports up to four 4-Mbit Flash memory devices
- MICROWIRE master interface to Flash memory devices
- The number of messages that can be stored is limited only by memory size
- Direct access to message memory
- Programmable message tag for message categorization, e.g., Mailboxes, InComing Messages (ICM), OutGoing Messages (OGM)
- Digital volume control
- Variable speed playback
- Real-time clock: Day of Week, Hours, Minutes
- Designed around the CR16A, a 16-bit general-purpose RISC core implementation of the CompactRISC architecture
- 16-bit architecture and implementation, 20.48 MHz operation
- On-chip DSP Module (DSPM) for high-speed DSP operations
- On-chip codec clock generation and interface
- Power-down mode
- Stores caller numbers
- Storage and management of messages
- Skip forward or backward during message playback
- Supports prerecorded vocabularies on Flash memory
- Available in PLCC 68-pin, and PQFP 100-pin packages

Figure i: Block Diagram—ISD-T267SC Basic Configuration with Samsung Flash

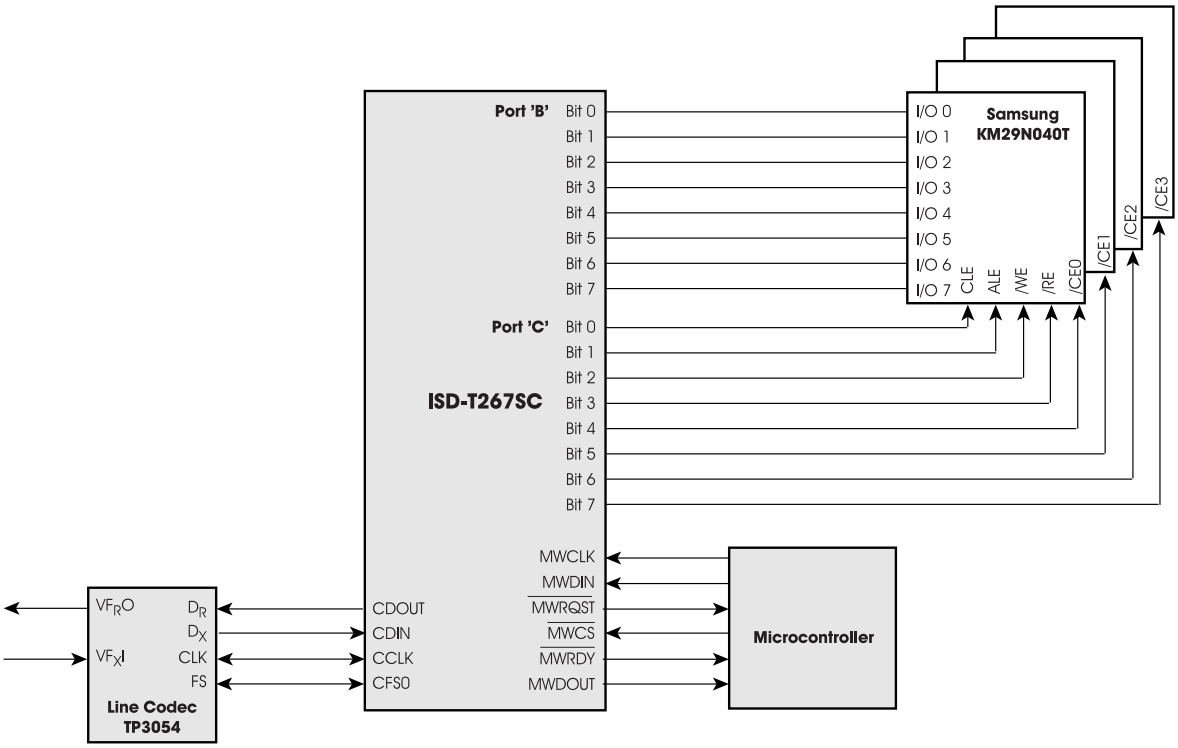
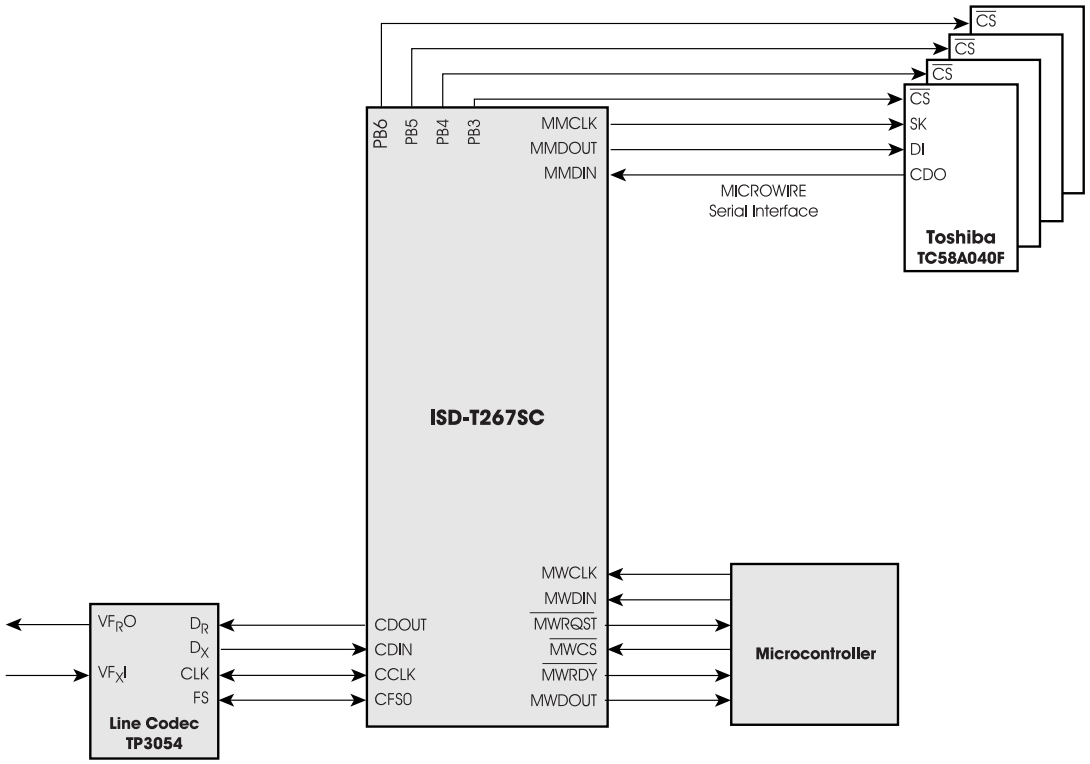
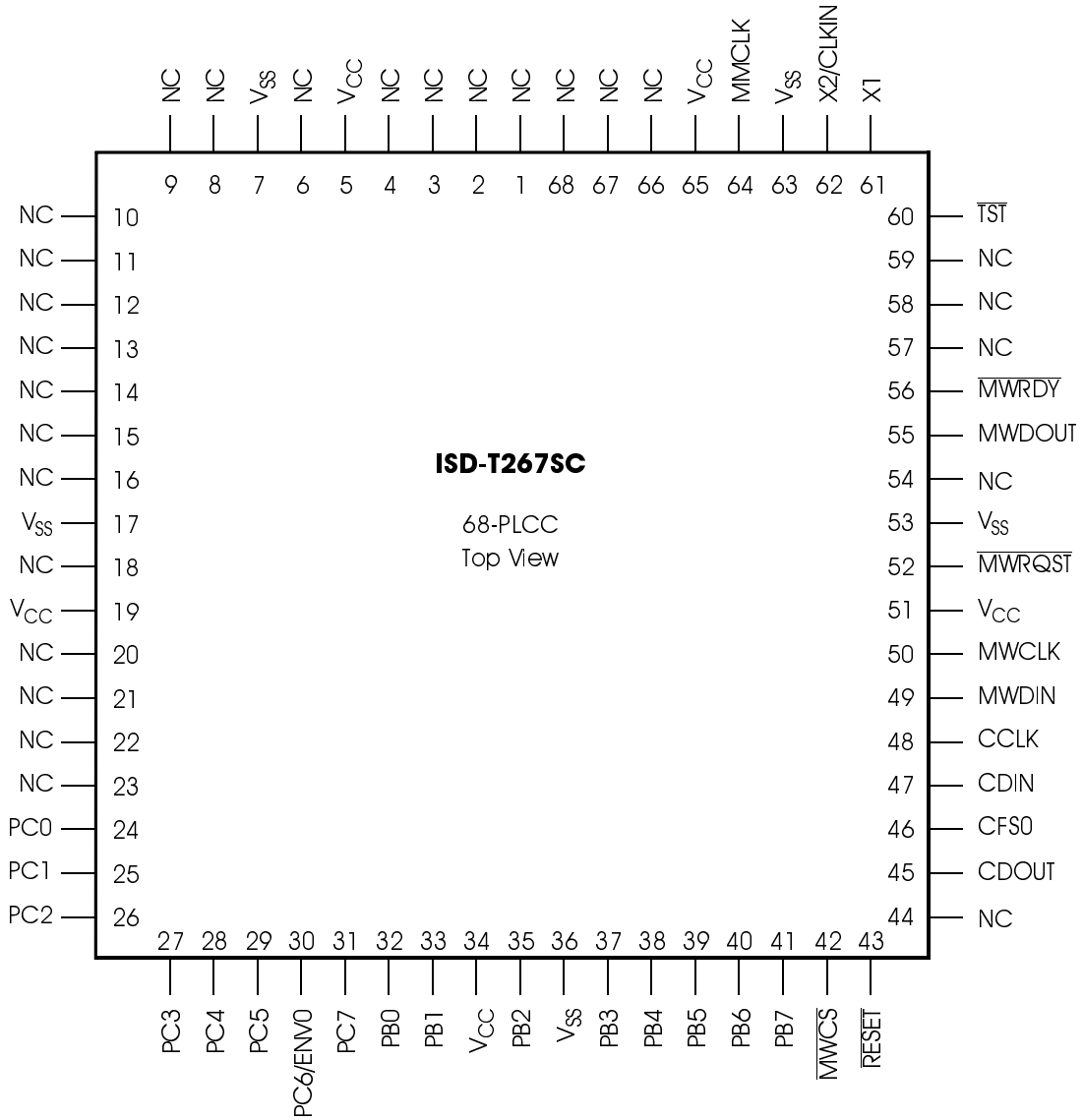


Figure ii: Block Diagram—ISD-T267SC Basic Configuration with the Toshiba Flash



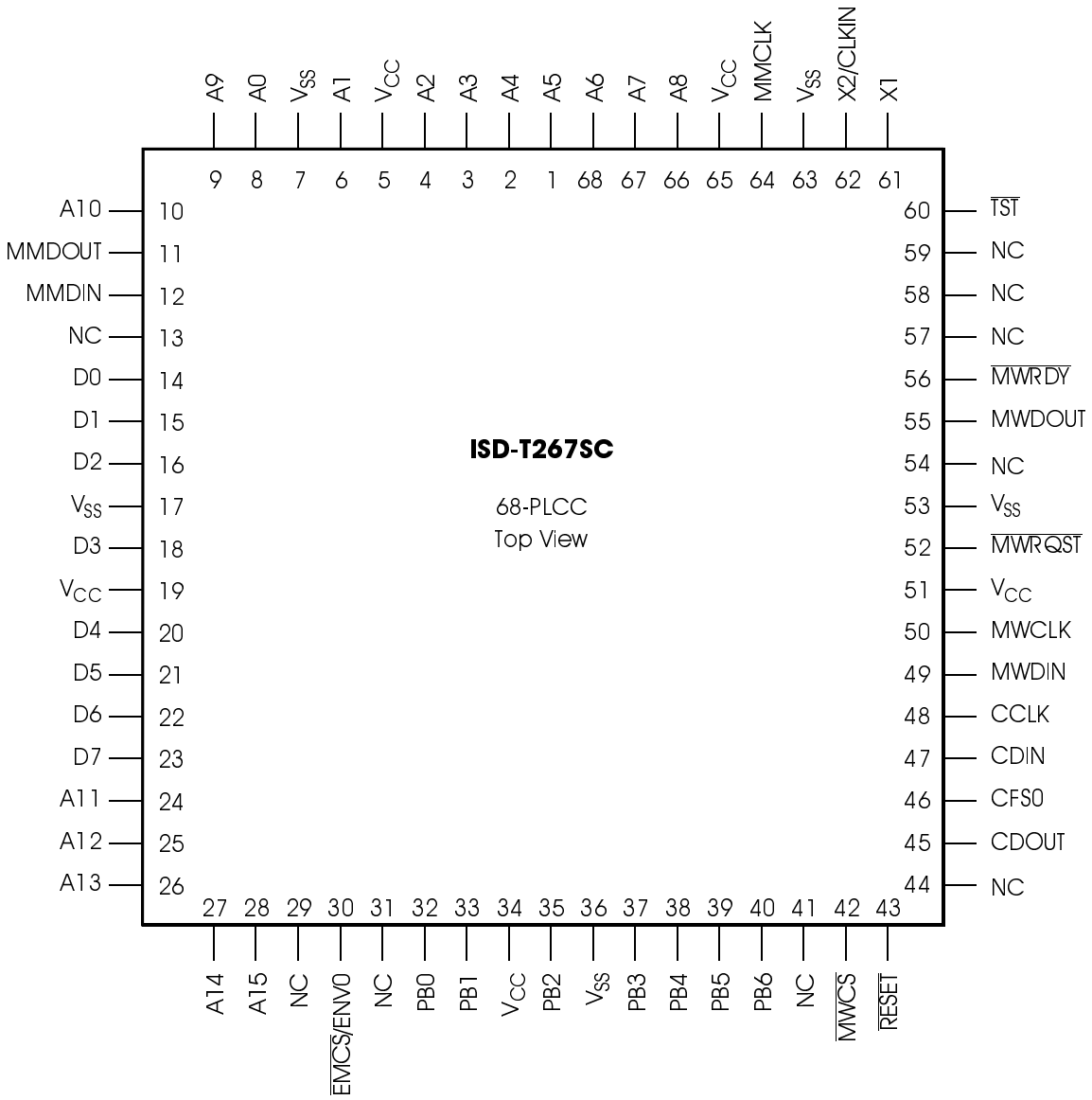
Chapter 1—HARDWARE

Figure 1-1: Pin Assignment in the 68-PLCC Package—Connection Diagram for Samsung Flash Memory



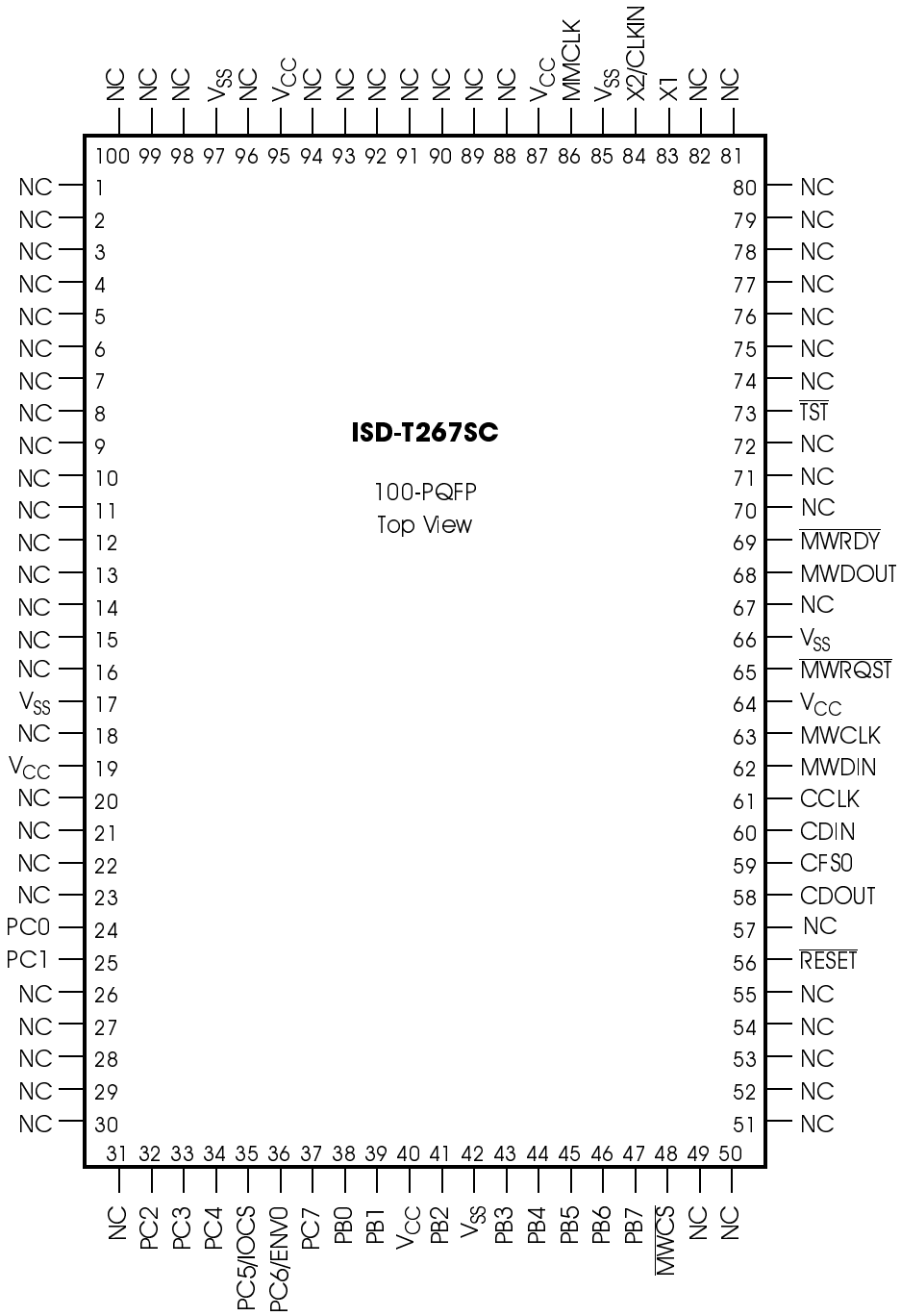
NOTE: *Pins marked NC should not be connected.*

Figure 1-2: Pin Assignment in the 68-PLCC Package—Connection Diagram for Toshiba Flash Memory



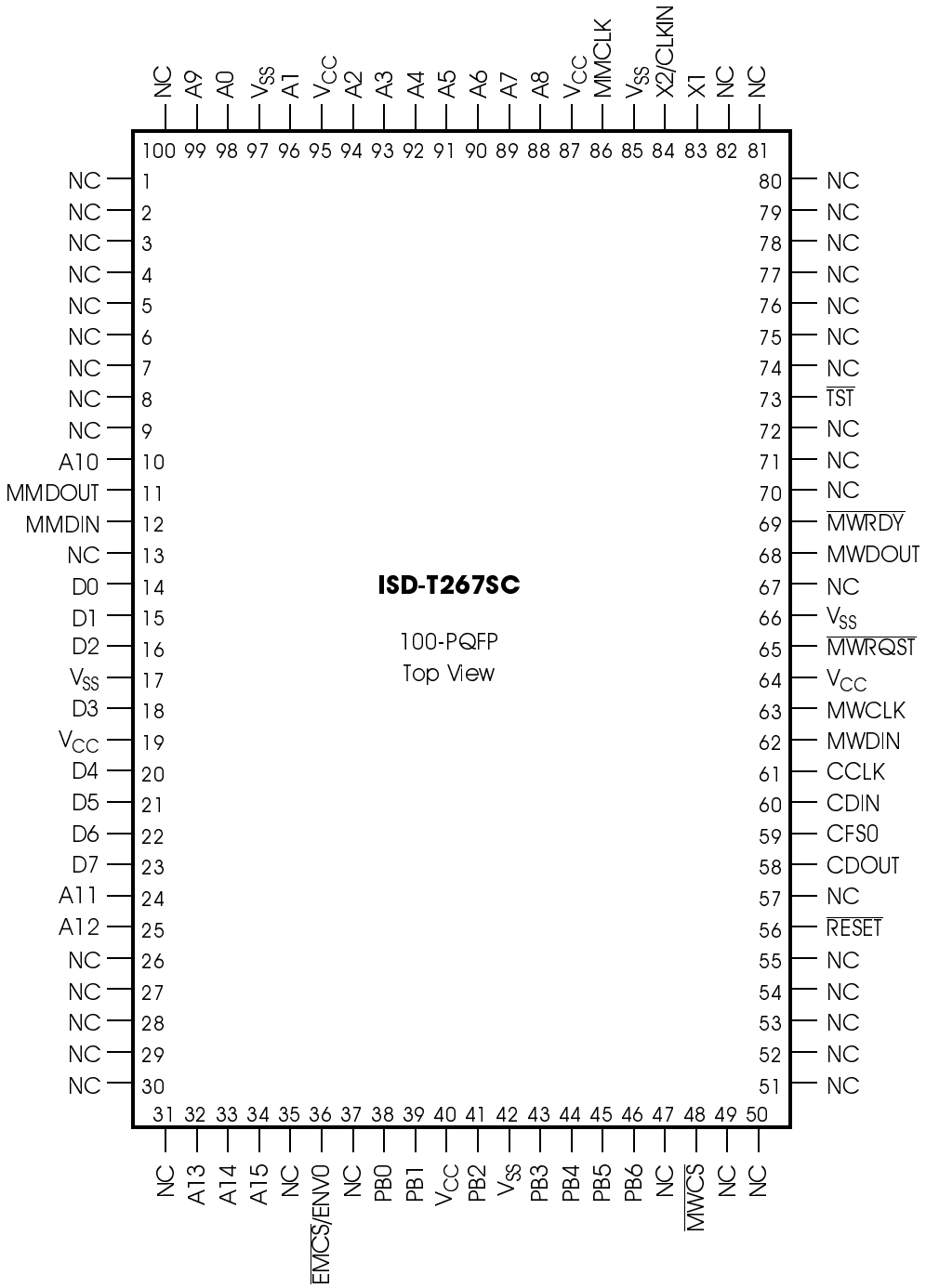
NOTE: *Pins marked NC should not be connected.*

Figure 1-3: Pin Assignment in the 100-PQFP Package—Connection Diagram for Samsung Flash Memory



NOTE: *Pins marked NC should not be connected.*

Figure 1-4: Pin Assignment in the 100-PQFP Package—Connection Diagram for Toshiba Flash Memory



NOTE: *Pins marked NC should not be connected.*

PIN ASSIGNMENT

The following sections detail the pins of the ISD-T267SC processor. Slashes separate the names of signals that share the same pin.

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.

Table 1-1: CompactSPEECH Pin—Signal Assignment

Pin Name	Type	Signal Name	I/O
A(0–15)	TTL	A(0–15)	Output
A(11–15)[1]	TTL	PC0–PC4	Output
B(0–7)[2]	TTL	(D8–D15)	Input/Output
BMCS	TTL	PC7	Output
CCLK	TTL	CCLK	Output
CDIN	TTL	CDIN	Input
CDOUT	TTL	CDOUT	Output
CFS0	TTL	CFS0	Output
D(0–7)	TTL	D(0–7)	Input/Output
$\overline{\text{EMCS}}$/ENV0	TTL1[3] CMOS[4]	$\overline{\text{EMCS}}$, PC6 ENV0	Output Input
MMCLK	TTL1[3]	MMCLK	Output
MMDIN	TTL	MMDIN	Input
MMDOUT	TTL1[3]	MMDOUT	Output
MWCLK	TTL	MWCLK	Input
$\overline{\text{MWCS}}$	TTL[3]	$\overline{\text{MWCS}}$	Input
MWDIN	TTL	MWDIN	Input
MWDOUT	TTL	MWDOUT	Output
$\overline{\text{MWRDY}}$	TTL	$\overline{\text{MWRDY}}$	Output
$\overline{\text{MWRQST}}$	TTL	$\overline{\text{MWRQST}}$	Output
PB(0–2)[5]	TTL	EA(16–18)	Output
PB(3–6)[6]	TTL	CS(0–3)	Output
$\overline{\text{RESET}}$	Schmitt[7]	$\overline{\text{RESET}}$	Input
V_{CC}	Power	V_{CC}	

Table 1-1: CompactSPEECH Pin—Signal Assignment (Continued)

Pin Name	Type	Signal Name	I/O
V_{SS}	Power	V_{SS}	
$\overline{WR0}$/$\overline{TST}$	TTL	$\overline{WR0}$ $\overline{TST}$	Output Input
X1	XTAL	X1	OSC
X2/CLKIN	XTAL TTL	X2 CLKIN	OSC Input

1. *Port B pins are shared with high section of Data Bus D8–D15.*
2. *Port C is shared with address lines A11–A15.*
3. *TTL1 output signals provide CMOS levels in the steady state, for small loads.*
4. *Input during reset, CMOS level input.*
5. *Virtual address lines for IVS ROM.*
6. *Chip select lines for Serial Flash devices.*
7. *Schmitt trigger input.*

FUNCTIONAL DESCRIPTION

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

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

RESETTING

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

Upon power up, $\overline{RESET}$ 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 $\overline{RESET}$ is applied, it must also remain active for not less than t_{RST}. During this period, and for 100 μs after, the $\overline{TST}$ signal must be high. This can be done with a pull-up resistor on the $\overline{TST}$ pin.

The value of $\overline{MWRDY}$ is undefined during the reset period, and for 100 μs 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 $\overline{EMCS}$/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 $\overline{EMCS}$.

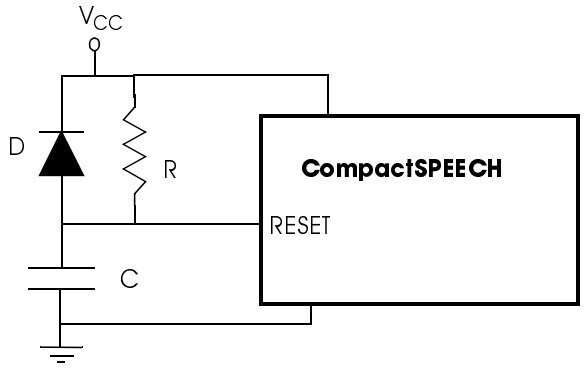
System Load on ENV0

For any load on the ENV0 pin, the voltage should not drop below V_{ENVh}.

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 logic 1.

Figure 1-5 shows a recommended circuit for generating a reset signal when the power is turned on.

Figure 1-5: Recommended Power-On Reset Circuit



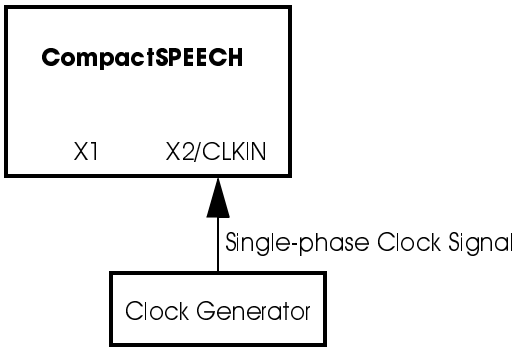
CLOCKING

The CompactSPEECH 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-, and should conform to the voltage-level requirements for CLKIN stated in "Electrical Characteristics" on page 1-14.

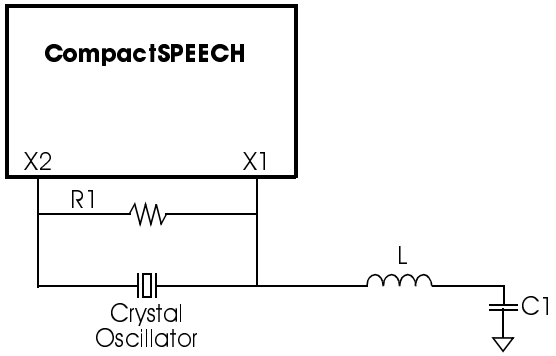
Figure 1-6: External Clock Source



Crystal Oscillator

A crystal oscillator is connected to the on-chip oscillator circuit via the X1 and X2 signals, as shown in Figure 1-7.

Figure 1-7: 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: Crystal Oscillator Component List

Component	Parameters	Values	Tolerance
Crystal Oscillator	Resonance Frequency Third Overtone Type Maximum Serial Resistance Maximum Shunt Capacitance Maximum Load Capacitance	40.96 MHz Parallel AT-Cut 50 Ω 7 pF 12 pF	N/A
Resistor R1		10 MΩ	5%
Capacitor C1		1000 pF	20%
Inductor L		3.9 µH	10%

POWER-DOWN MODE

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

In power-down mode, the clock frequency of the CompactSPEECH processor is reduced, and some of the processor modules are deactivated. As a result, the CompactSPEECH processor consumes much less power than in normal-power mode (less than 1.5 µA). Although the CompactSPEECH processor does not perform all its usual functions in power-down mode, it still keeps stored messages and maintains the time and day.

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 memory devices, the power supply to these devices must not be disconnected.*

The CompactSPEECH processor stores messages, and all memory management information, in Flash memory. Thus, there is no need to maintain the power to the processor to preserve stored messages when a Flash device is used. If the microcontroller's real-time clock (and not the CompactSPEECH processor's real-time clock) is used to maintain the time and day, neither the Flash nor the CompactSPEECH processor require battery backup during power failure. In this case, when returning to normal mode, the microcontroller should perform the initialization sequence, as described in "Initialization" on page 2–16, and use the SETD command to set the time and day.

To keep power consumption low in 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.

The PDM (Go To Power-down Mode) command switches the CompactSPEECH processor to power-down mode. (For an explanation of the CompactSPEECH processor commands, see "Command Description" on page 2–21.) It may only be issued when the CompactSPEECH processor is in the IDLE state. (For an explanation of the CompactSPEECH processor states, see "Initialization" on page 2–16.) If it is necessary to switch to power-down mode from any other state, the controller must first issue an S command to switch the CompactSPEECH processor to the IDLE state, and then issue the PDM command. Sending any command while in power-down mode resets the CompactSPEECH processor detectors, and returns the CompactSPEECH processor 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.*

POWER AND GROUNDING

The CompactSPEECH processor requires a single 5 V power supply, applied to the V_{CC} pins.

The grounding connections are made on the GND pins.

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.

Use decoupling capacitors to keep the noise level to a minimum. Attach standard 0.1 µF ceramic capacitors to the V_{CC} and GND pins, as close as possible to the CompactSPEECH processor.

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

MEMORY INTERFACE

Flash Support

The ISD-T267SC CompactSPEECH supports 4-Mbit and 16-Mbit, byte wide, Flash devices for storing messages. These Flash devices are organized in 4 Kbyte blocks. The ISD-T267SC can use such devices for message recording without any affect on voice quality, if they conform to Flash/Aflash specifications.

There are two major limitations imposed by current Flash technology: block erasure time and Flash endurance. Both these limitations are handled by the CompactSPEECH firmware.

Block Erasure

In a Flash environment, an erase operation is also required. You must ensure that a memory location, which as previously written, is erased prior to writing. The basic unit that can be read or written, is a byte; the basic unit that can be erased is an entire 4 Kbyte block.

Block erasure takes time. The following erasure times are quoted from Samsung's and Toshiba's datasheets for devices supported by the ISD-T267SC. For current information refer to Samsung's KM29N040T and Toshiba's TC58A040F data sheets.

	Block Erasure Time	Total Memory Erasure Time
Samsung	8 msec (typ))	1.25 sec (max)
Toshiba	7 msec (typ))	1.27 sec (max)

A Flash memory can not be written while erasure is in progress. During erasure, access to the Flash is not allowed. The CompactSPEECH, however, accepts commands which do not require Flash access (e.g., Get Status) during erasure.

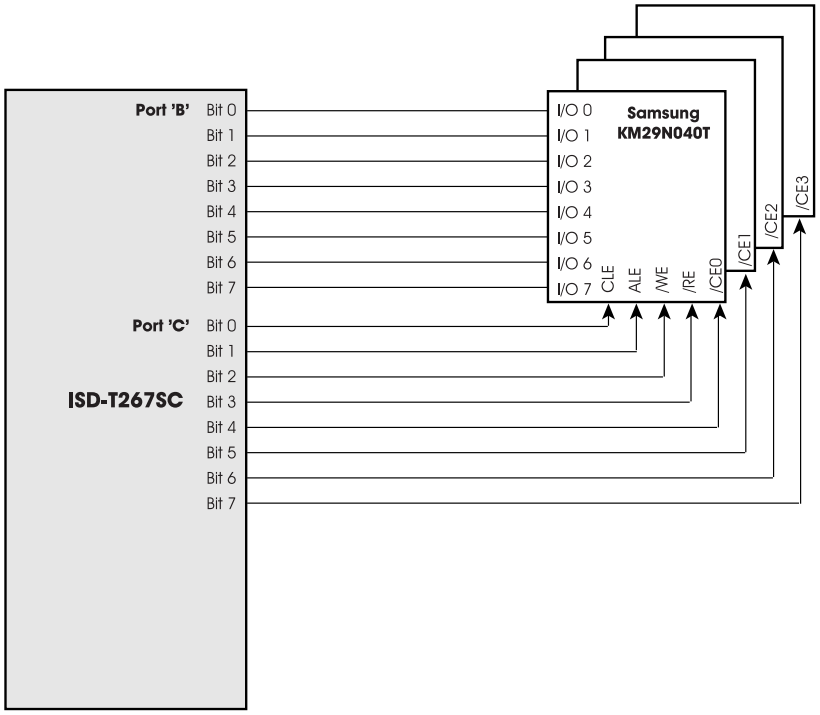
Flash Interface

The CompactSPEECH processor supports up to four Toshiba's TC58A040F 4-Mbit Flash memory devices or up to four Samsung's KM29N040T Flash memory devices for storing messages.

Samsung's KM29N040T

The CompactSPEECH processor supports up to four Samsung KM29N040T NAND Flash memory without any external logic. The KM29N040T Flash memory uses a byte wide interface to control Read/Write functions and data path. The CompactSPEECH processor interfaces to the Samsung KM29N040T through Port "B" and Port "C." Port "B" is a bidirectional data path for command and data information. See Figure 1-8. Port "C" is used for controlling Read, Write, and Chip enable (CLE) signal. Refer to the Samsung KM29N040T NAND Flash datasheet for more information.

Figure 1-8: Samsung's KM29N040T Flash Memory Diagram



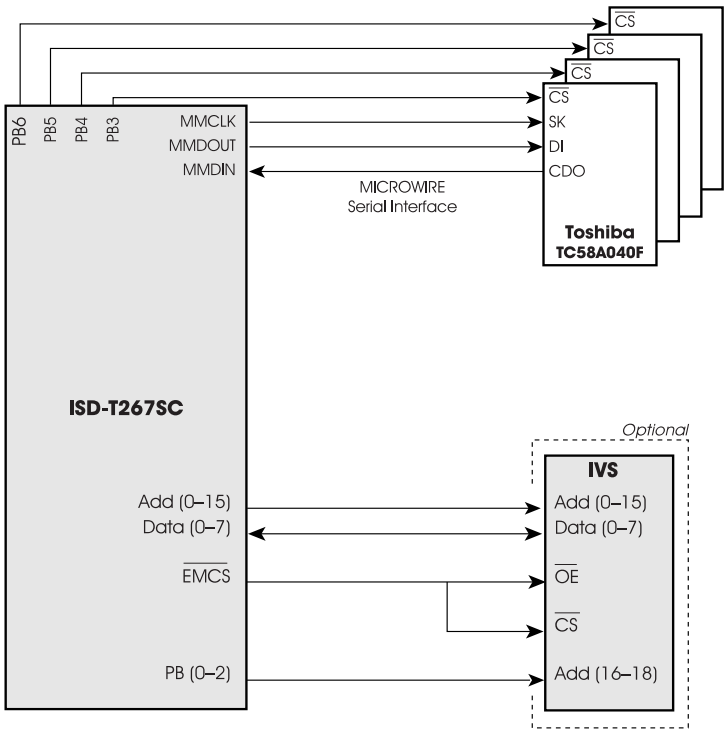
Toshiba's TC58A040F Serial Flash

The TC58A040F is organized as 128 blocks of 128 pages, each containing 32 bytes. A block is the smallest unit that can be erased, and is 4 Kbytes in size. See Figure 1-9.

Not all 128 blocks are available for recording. Up to 10 blocks may contain bad bits, and one block is write-once and holds the locations of these unusable blocks.

For further information about Toshiba's TC58A040F, please refer to the TC58A040F datasheet.

Figure 1-9: Toshiba's TC58A040F Flash Memory Diagram



Flash Endurance

The Flash memory may be erased up to 100,000 times. To reduce the effect of this limitation, 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.

Consider the following extensive usage of all the Flash memory device 's blocks:

1. Record 15 minutes of messages (until the memory is full).
2. Playback 15 minutes (all the recorded messages).
3. Delete all messages.

Assuming a Flash memory device is used in this manner 24 times a day, its expected lifetime is:

Flash Lifetime = 100,000 / (24 * 365) = 11.4 years

Thus the Flash memory device will last for over ten years, even when used for twelve hours of recording per day.

Note, that if an Flash memory device is used, then, under the same conditions, it will last for more than 20 years.

Message Organization and Recording Time

A CompactSPEECH processor message uses at least one block.

The maximum recording time depends on four factors:

1. The basic compression rate (5.2 Kbit/s or 7.3 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)

Assuming a single message recorded in all the available memory space of a 4-Mbit device with no bad blocks, the maximum recording time using 5.2 Kbit/s compression is as follows:

Table 1-3: Recording Time on a 4-Mbit Device

Silence	Total Recording Time
0	13 minutes and 9 seconds
10%	14 minutes and 25 seconds
15%	15 minutes and 7 seconds
20%	15 minutes and 47 seconds
25%	16 minutes and 25 seconds

ROM Interface

IVS vocabularies can be stored in either Flash and/or ROM. The CompactSPEECH processor supports IVS ROM devices through Expansion Memory. Up to 64 Kbytes (64K × 8) of Expansion Memory are supported directly. Nevertheless, the CompactSPEECH 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 CompactSPEECH processor using the data bus, D(0:7), the address bus, A(0:15), the extended address signals, 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.

NOTE *The Samsung (KM29N040T) parallel Flash memory device does not support external ROM.*

Reading from Expansion Memory

An Expansion Memory read bus cycle starts at T1, when the data bus is in TRI-STATE, and the address is driven on the address bus. $\overline{EMCS}$ is asserted (LOW) on a T2W1 cycle. This cycle is followed by three T2W cycles and one T2 cycle. Data is sampled by the ISD-T267SC at the end of the T2 cycle.

The transaction is terminated at T3, when $\overline{EMCS}$ becomes inactive (HIGH). The address remains valid until T3 is complete. A T3H cycle is added after the T3 cycle. The address remains valid unit the end of T3H, see

$\overline{WR0}$ is inactive (HIGH) during the read bus cycle.

CODEC INTERFACE

The CompactSPEECH processor provides an on-chip interface to a serial codec. This interface supports codec operation in long or short-frame formats. The format is selected with the CFG command.

The codec interface uses four signals CDIN, CDOUT, CCLK and CFS0.

The CDIN input pin and the CDOUT, CCLK and CFS0 output pins are connected to the codec.

Data is transferred to the codec through the CDOUT pin. Data is read from the codec through the CDIN pin.

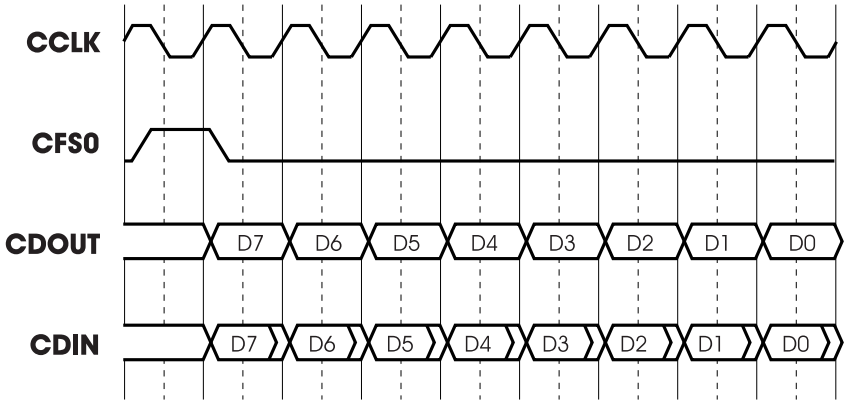
Short Frame Protocol

When short frame protocol is configured, eight data bits are exchanged with each codec in each frame, i.e., CFS0 cycle.

Data transfer starts when CFS0 is set to 1 for one CCLK cycle. The data is then transmitted, bit-by-bit, via the CDOUT output pin. Concurrently, the received data is shifted in via the CDIN input pin. Data is shifted one bit in each CCLK cycle.

Figure 1-10 shows how the codec interface signals behave when short frame protocol is configured.

Figure 1-10: Codec Protocol—Short Frame



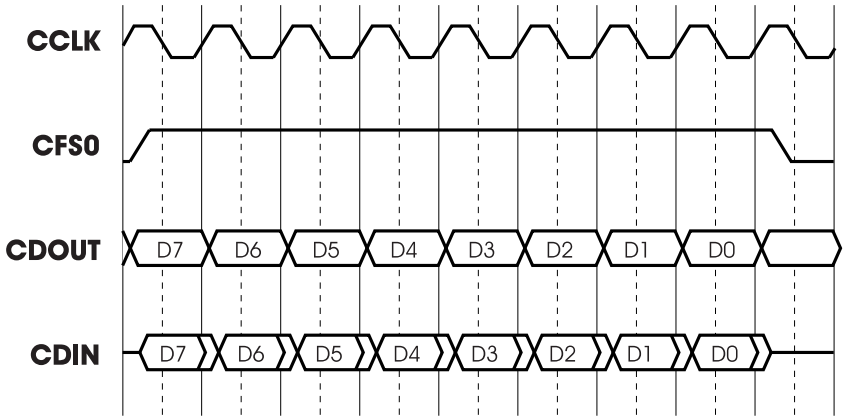
Long Frame Protocol

When long frame protocol is configured, eight data bits are exchanged with each codec, as for the short frame protocol. However, for the long frame protocol, data transfer starts by setting CFS0 to 1 for eight CCLK cycles.

Simultaneously, the data for the first codec is shifted out bit-by-bit, via the CDOUT output pin, as in short frame protocol. Concurrently, the received data is shifted in through the CDIN input. The data is shifted one bit in each CCLK cycle.

Figure 1-11 shows how the codec interface signals behave when long frame protocol is configured.

Figure 1-11: Codec Protocol—Long Frame



SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Storage temperature	–65°C to +150°C
Temperature under bias	0°C to +70°C
All input or 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.*

ELECTRICAL CHARACTERISTICS

T_A = 0°C to +70°C, V_{CC} = 5 V ±10%, GND = 0 V

Table 1-4: Electrical Characteristics

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]		65.0	80.0	µA
I_{CC2}	Standby Supply Current	Normal Operation Mode, DSPM Idle[2]		40.0		µA
I_{CC3}	Power-down Mode Supply Current	Power-down Mode[2,3]			1.5	µA

Table 1-4: Electrical Characteristics (Continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_L	Input Load Current[4]	$0 V \le V_{IN} \le V_{CC}$	–5.0		5.0	µA
I_O (Off)	Output Leakage Current (I/O pins in Input Mode)	$0 V \le V_{OUT} \le V_{CC}$	–5.0		5.0	µA
V_{ENVh}	ENV0 High Level, Input Voltage		3.6			V
V_{Hh}	CMOS Input with Hysteresis, Logical 1 Input Voltage		3.6			V
V_{Hl}	CMOS Input with Hysteresis, Logical 0 Input Voltage				1.1	V
V_{Hys}	Hysteresis Loop Width[1]		0.5			V
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		0.8	V
V_{OH}	Logical 1 TTL, Output Voltage	I_{OH} = –0.4 µA	2.4			V
V_{OHWC}	MMCLK, MMDOUT and $\overline{EMCS}$ Logical 1, Output Voltage	I_{OH} = –0.4 µA	2.4			V
		I_{OH} = –50 µA[5]	V_{CC} – 0.2			V
V_{OL}	Logical 0, TTL Output Voltage	I_{OL} = 4 µA			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 µA			0.45	V
		I_{OL} = 50 µA[5]			0.2	V
V_{XH}	CLKIN Input, High Voltage	External Clock	2.0			V
V_{XL}	CLKIN Input, Low Voltage	External Clock			0.8	V

1. *Guaranteed by design.*
2. *I_{OUT} = 0, T_A = 25°C, V_{CC} = 5 V, operating from a 40.96 MHz crystal, and running from internal memory with Expansion Memory disabled.*
3. *All input signals are tied to 1 or 0 (above V_{CC} – 0.5 V or below V_{SS} + 0.5 V).*
4. *Maximum 20 µA for all pins together.*
5. *Measured in power-down mode. The total current driven, or sourced, by all the CompactSPEECH processor's output signals are less than 50 µA.*

SWITCHING CHARACTERISTICS

Definitions

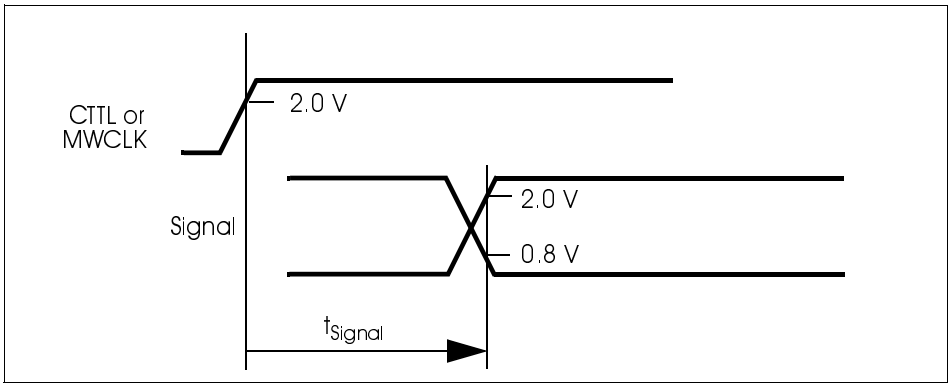
All timing specifications in this section refer to 0.8 V or 2.0 V on the rising or falling edges of the signals, as illustrated in Figures 1-12 through 1-18, unless specifically stated otherwise.

Maximum times assume capacitive loading of 50 pF.

CLKIN crystal frequency is 40.96 MHz.

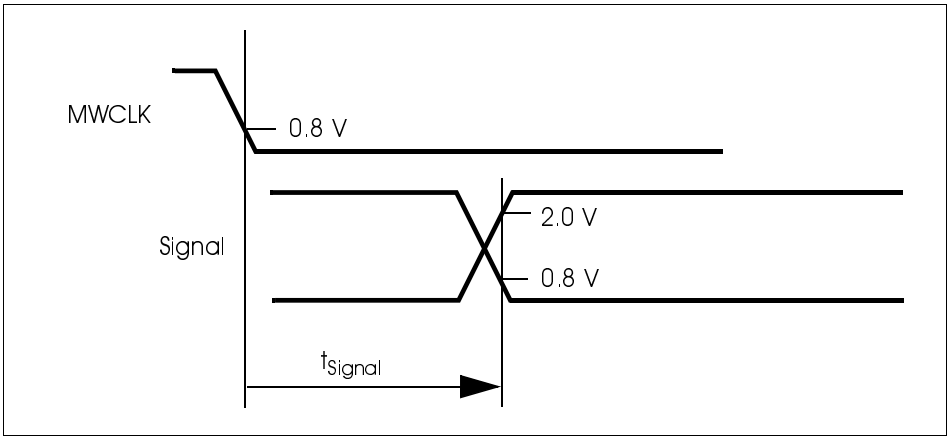
NOTE *CTTL is an internal signal and is used as a reference to explain the timing of other signals. See Figure 1-26.*

Figure 1-12: Synchronous Output Signals (Valid, Active and Inactive)



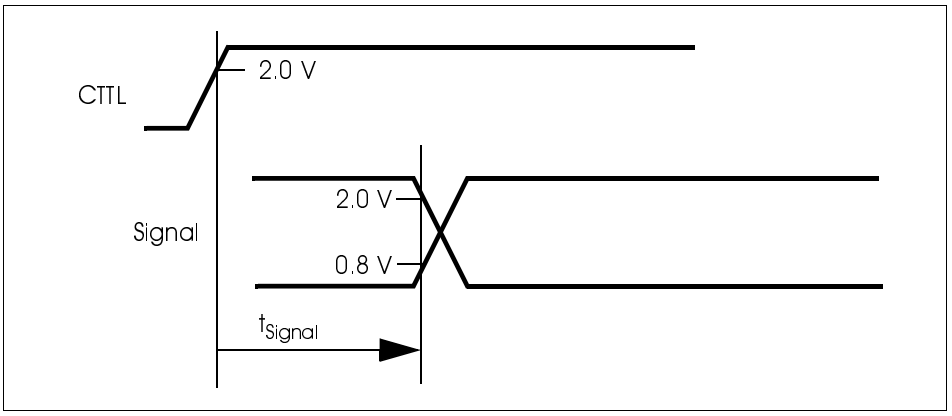
NOTE: *Signal valid, active or inactive time, after a rising edge of CTTL or MWCLK.*

Figure 1-13: Synchronous Output Signals (Valid)



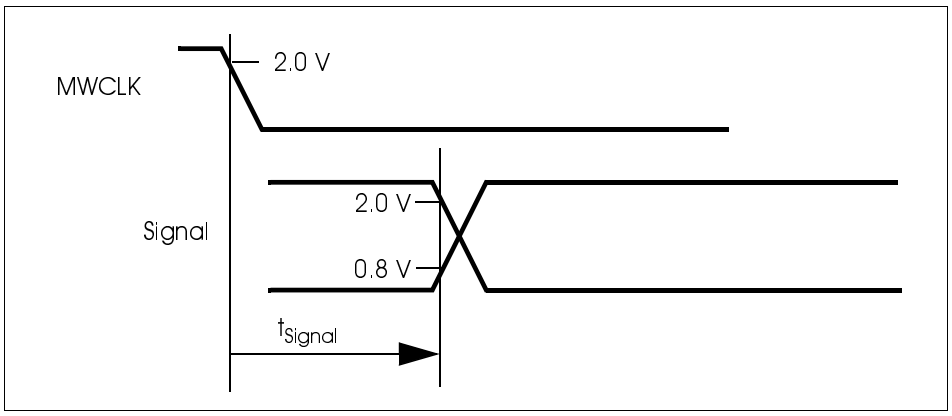
NOTE: *Signal valid time, after a falling edge of MWCLK.*

Figure 1-14: Synchronous Output Signals (Hold)



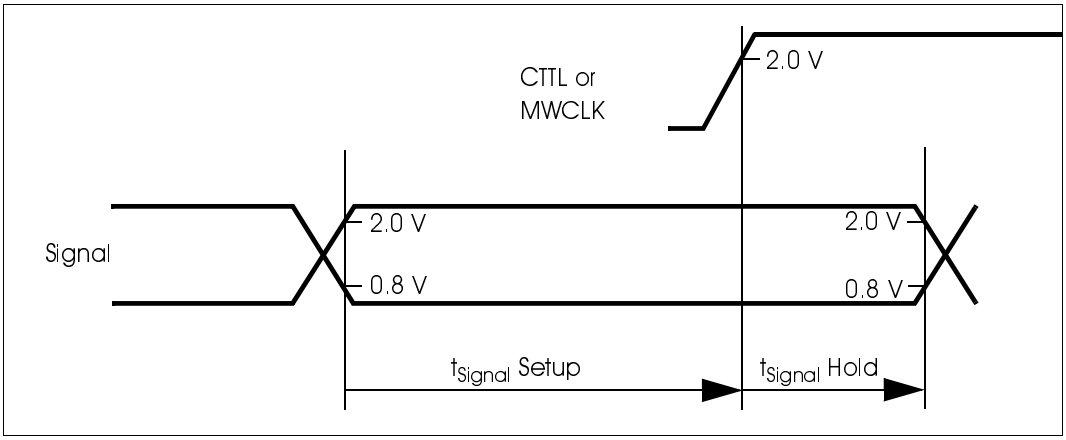
NOTE: *Signal hold time, after a rising edge of CTTL.*

Figure 1-15: Synchronous Output Signals (Hold)



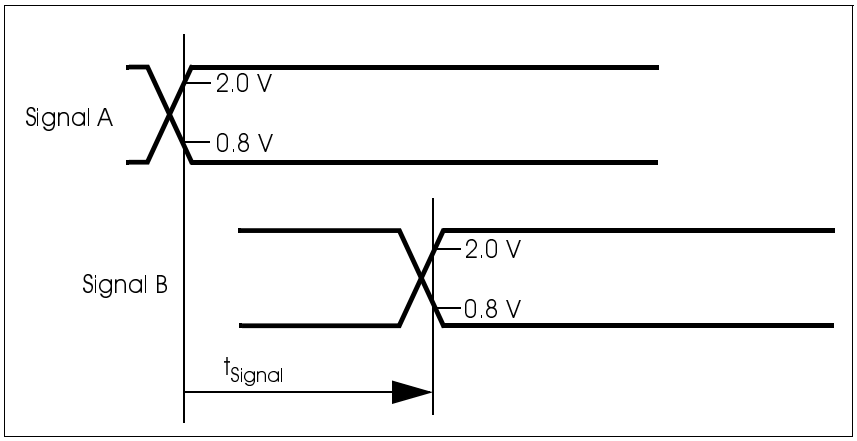
NOTE: *Signal hold time, after a falling edge of MWCLK.*

Figure 1-16: Synchronous Input Signals



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

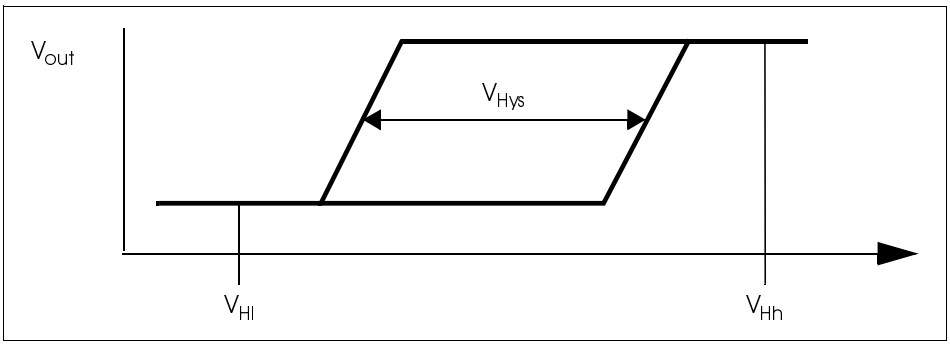
Figure 1-17: Asynchronous Signals



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

The $\overline{\text{RESET}}$ signal has a Schmitt trigger input buffer. Figure 1-18 shows the characteristics of the input buffer.

Figure 1-18: Hysteresis Input Characteristics



SYNCHRONOUS TIMING TABLES

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

Output Signals

Table 1-5: Output Signals

Symbol	Figure	Description	Reference Conditions	Min (ns)	Max (ns)
t_{Ah}	1-21	Address Hold	After R.E. CTTL	0.0	
t_{Av}	1-21	Address Valid	After R.E. CTTL, T1		12.0
t_{CCLKa}	1-19	CCLK Active	After R.E. CTTL		12.0
t_{CCLKh}	1-19	CCLK Hold	After R.E. CTTL	0.0	
t_{CCLKia}	1-19	CCLK Inactive	After R.E. CTTL		12.0
t_{CDOh}	1-19	CDOUT Hold	After R.E. CTTL	0.0	
t_{CDOv}	1-19	CDOUT Valid	After R.E. CTTL		12.0
t_{CTp}	1-26	CTTL Clock Period[1]	R.E. CTTL to next R.E. CTTL	48.8	50,000
t_{EMCSa}	1-21	$\overline{EMCS}$ Active	After R.E. CTTL, T2W1		12.0
t_{EMCSh}	1-21	$\overline{EMCS}$ Hold	After R.E. CTTL	0.0	
t_{EMCSia}	1-21	$\overline{EMCS}$ Inactive	After R.E. CTTL T3		12.0
t_{FSa}	1-19	CFS0 Active	After R.E. CTTL		25.0
t_{FSh}	1-19	CFS0 Hold	After R.E. CTTL	0.0	
t_{FSia}	1-19	CFS0 Inactive	After R.E. CTTL		25.0
t_{MMCLKa}	1-24	Master MICROWIRE Clock Active	After R.E. CTTL		12.0
t_{MMCLKh}	1-24	Master MICROWIRE Clock Hold	After R.E. CTTL	0.0	
$t_{MMCLKia}$	1-24	Master MICROWIRE Clock Inactive	After R.E. CTTL		12.0
t_{MMDOh}	1-24	Master MICROWIRE Data Out Hold	After R.E. CTTL	0.0	
t_{MMDOv}	1-24	Master MICROWIRE Data Out Valid	After R.E. CTTL		12.0
t_{MWDOf}	1-22	MICROWIRE Data Float[2]	After R.E. $\overline{MWCS}$		70.0
t_{MWDOh}	1-22	MICROWIRE Data Out Hold[2]	After F.E. MWCK	0.0	
t_{MWDOnf}	1-22	MICROWIRE Data No Float[2]	After F.E. $\overline{MWCS}$	0.0	70.0
t_{MWDOv}	1-22	MICROWIRE Data Out Valid[2]	After F.E. MWCK		70.0
t_{MWITOp}	1-23	MWDIN to MWDOUT	Propagation Time		70.0
t_{MWRDYa}	1-22	$\overline{MWRDY}$ Active	After R.E. of CTTL	0.0	35.0

Table 1-5: Output Signals (Continued)

Symbol	Figure	Description	Reference Conditions	Min (ns)	Max (ns)
$t_{MWRDYia}$	1-22	$\overline{MWRDY}$ Inactive	After F.E. MWCLK	0.0	70.0
t_{PABCh}	1-25	PB and $\overline{MWRQST}$	After R.E. CTTL	0.0	
t_{PABCv}	1-25	PB and $\overline{MWRQST}$	After R.E. CTTL, T2W1		12.0

1. In normal operation mode t_{CTp} must be 48.8 ns; in power-down mode, t_{CTp} must be 50,000 ns.

2. Guaranteed by design, but not fully tested.

Input Signals

Table 1-6: Input Signals

Symbol	Figure	Description	Reference Conditions	Min (ns)
t_{CDIh}	1-19	CDIN Hold	After R.E. CTTL	0.0
t_{CDIs}	1-19	CDIN Setup	Before R.E. CTTL	11.0
t_{DIh}	1-21	Data in Hold (D0:7)	After R.E. CTTL T1, T3 or TI	0.0
t_{DIs}	1-21	Data in Setup (D0:7)	Before R.E. CTTL T1, T3 or TI	15.0
t_{MMDINh}	1-24	Master MICROWIRE Data In Hold	After R.E. CTTL	0.0
t_{MMDINs}	1-24	Master MICROWIRE Data In Setup	Before R.E. CTTL	11.0
t_{MWCKh}	1-22	MICROWIRE Clock High (slave)	At 2.0 V (both edges)	100.0
t_{MWCKl}	1-22	MICROWIRE Clock Low (slave)	At 0.8 V (both edges)	100.0
t_{MWCKp}	1-22	MICROWIRE Clock Period (slave)[1]	R.E. MWCLK to next R.E. MWCLK	2.5 µs
t_{MWCLKh}	1-22	MWCLK Hold	After $\overline{MWCS}$ becomes inactive	50.0
t_{MWCLKs}	1-22	MWCLK Setup	Before $\overline{MWCS}$ becomes active	100.0
t_{MWCSh}	1-22	$\overline{MWCS}$ Hold	After F.E. MWCLK	50.0
t_{MWCSs}	1-22	$\overline{MWCS}$ Setup	Before R.E. MWCLK	100.0
t_{MWDIh}	1-22	MWDIN Hold	After R.E. MWCLK	50.0
t_{MWDIs}	1-22	MWDIN Setup	Before R.E. MWCLK	100.0
t_{PWR}	1-28	Power Stable to $\overline{RESET}$ R.E.[2]	After V_{CC} reaches 4.5 V	30.0 ms

Table 1-6: Input Signals (Continued)

Symbol	Figure	Description	Reference Conditions	Min (ns)
t_{RSTw}	1-27	$\overline{RESET}$ Pulse Width	At 0.8 V (both edges)	10.0 ms
t_{Xh}	1-26	CLKIN High	At 2.0 V (both edges)	$t_{X1p}/2 - 5$
t_{Xl}	1-26	CLKIN Low	At 0.8 V (both edges)	$t_{X1p}/2 - 5$
t_{Xp}	1-26	CLKIN Clock Period	R.E. CLKIN to next R.E. CLKIN	24.4

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

2. *Guaranteed by design, but not fully tested.*

TIMING DIAGRAMS

Figure 1-19: Codec Short Frame Timing

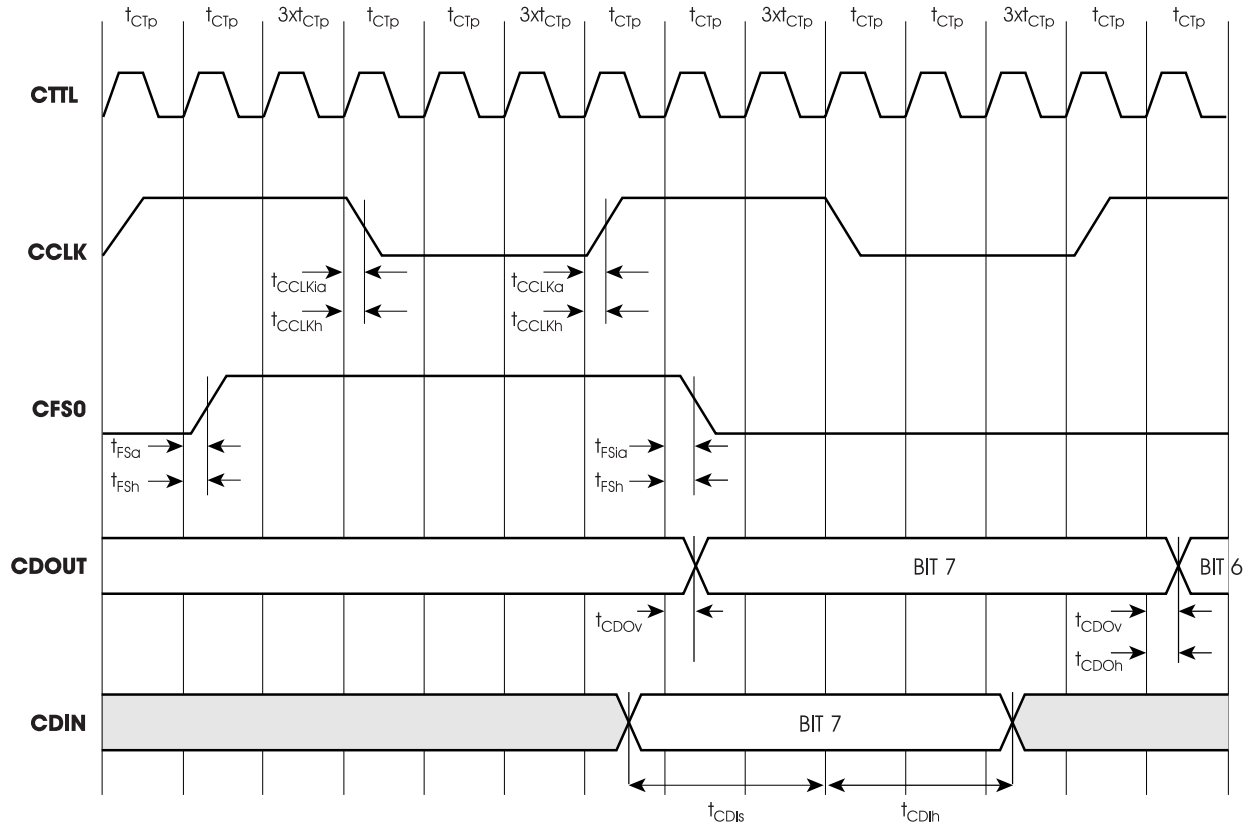


Figure 1-20: Codec Long Frame Timing

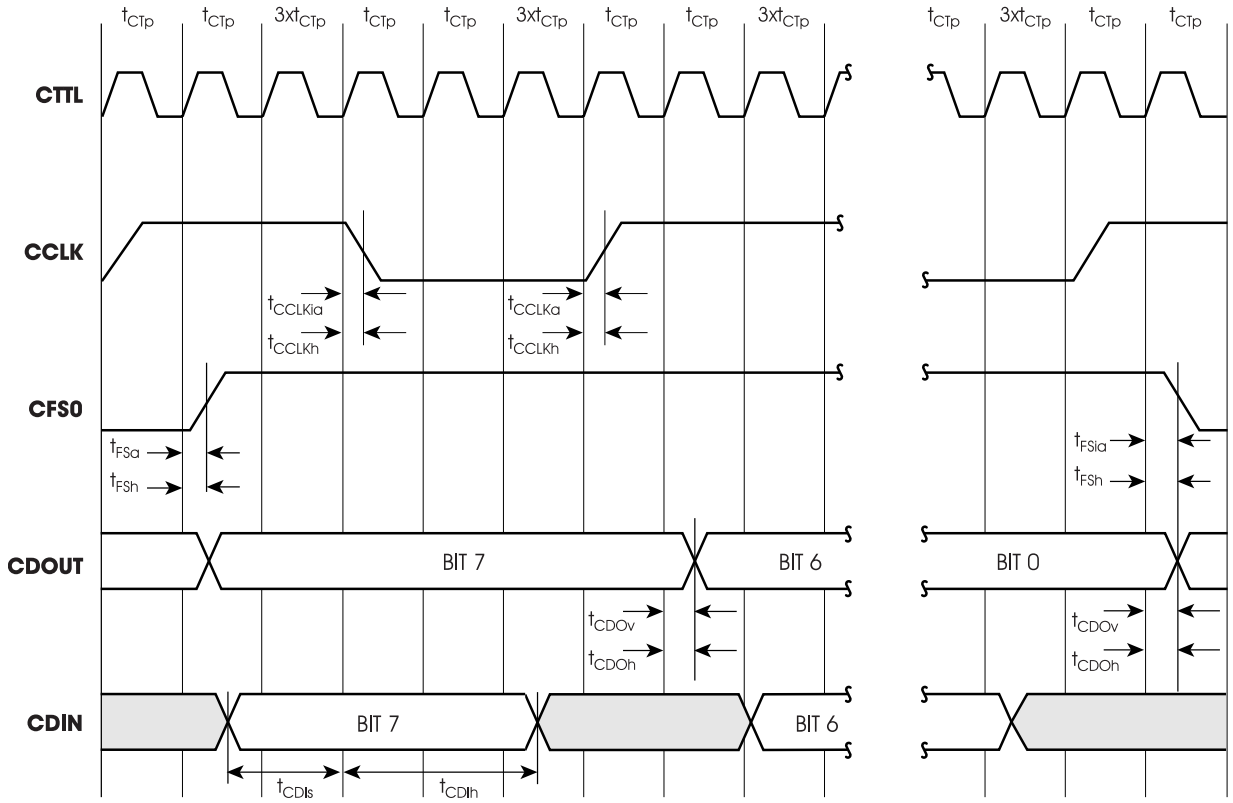
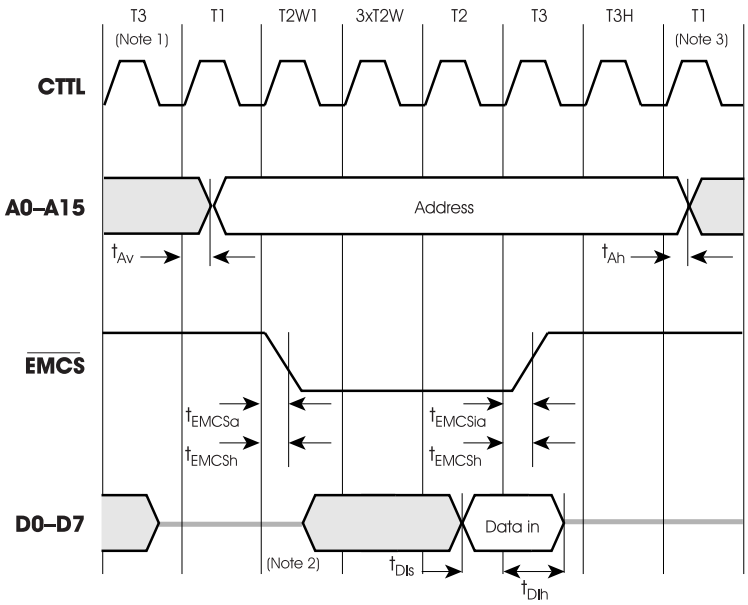


Figure 1-21: ROM Read Cycle Timing



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 1-22: MICROWIRE Transaction Timing—Data Transmitted to Output

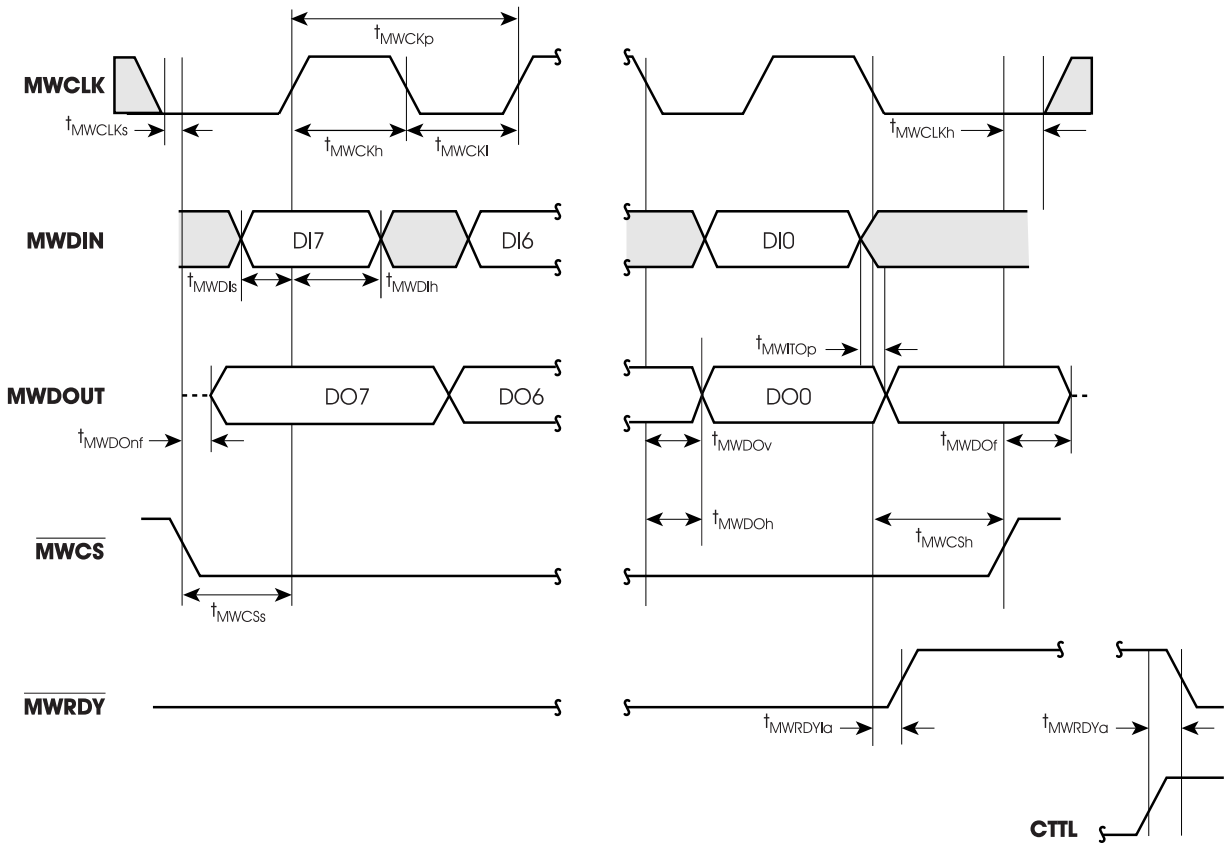


Figure 1-23: MICROWIRE Transaction Timing—Data Echoed to Output

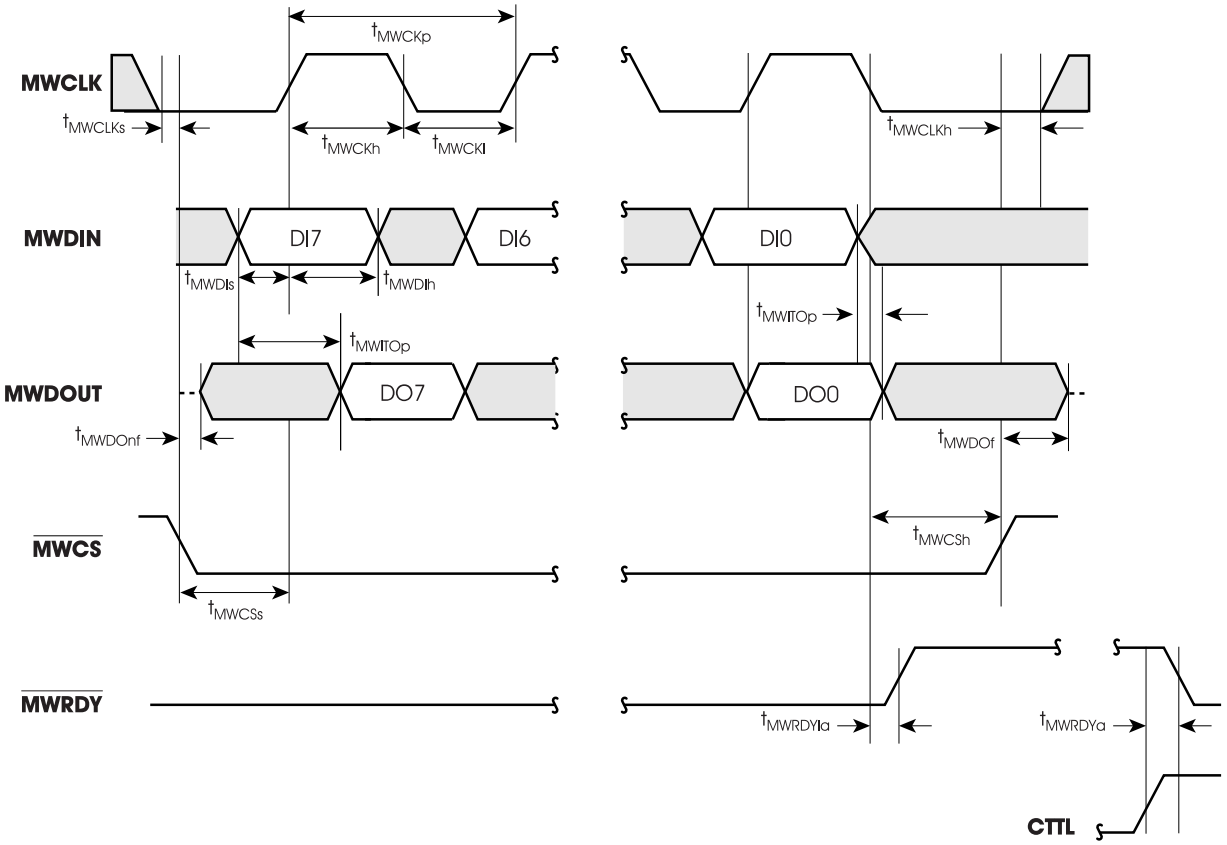


Figure 1-24: Master MICROWIRE Timing

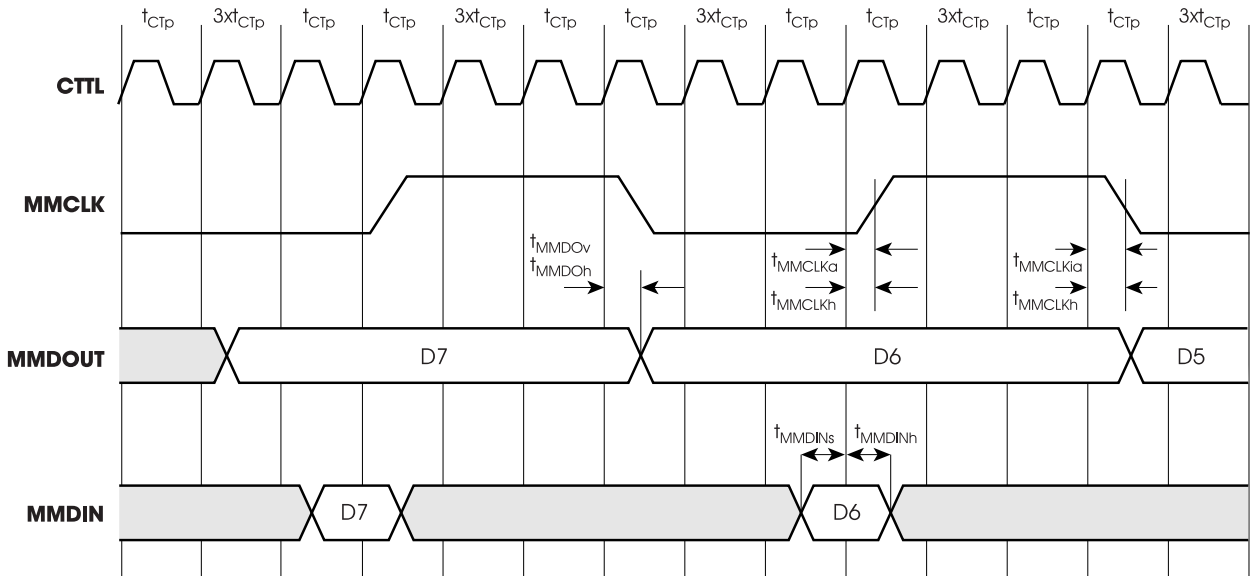
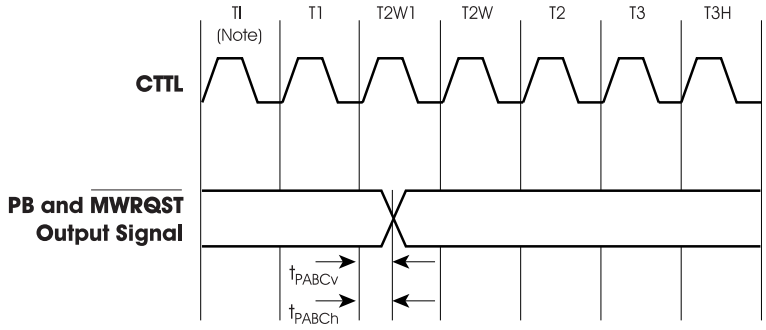


Figure 1-25: Output Signal Timing for Port PB and $\overline{\text{MWRQST}}$



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

Figure 1-26: CTTL and CLKIN Timing

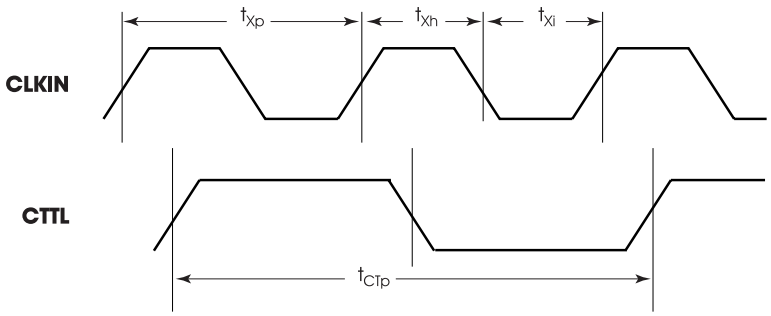


Figure 1-27: Reset Timing When Reset Is Not at Power-Up

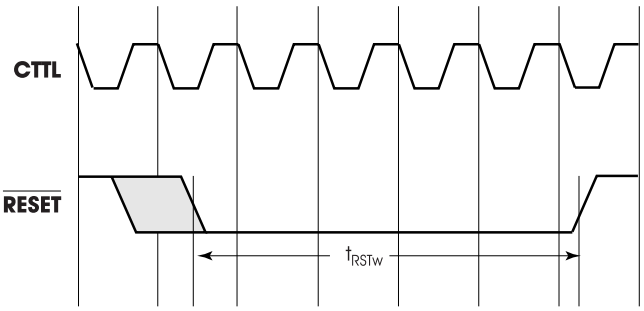
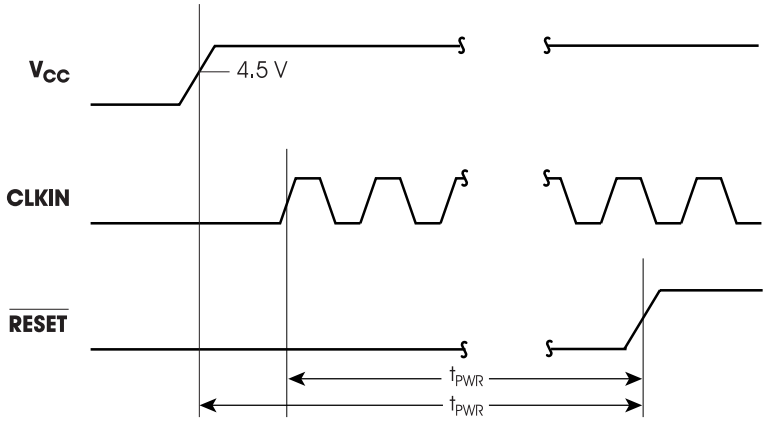


Figure 1-28: Reset Timing When Reset Is at Power-Up



Chapter 2—SOFTWARE

OVERVIEW

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

DSP-BASED ALGORITHMS

- Speech compression and decompression
- DTMF detector with echo canceler
- VOX and constant energy detector, and dial-tone detector
- Caller ID modem
- Digital volume control

SYSTEM SUPPORT

- Command interface to an external microcontroller
- Memory and message manager
- IVS support
- Tone generator
- Real-time clock handler
- Power-down mode support

PERIPHERALS SUPPORT

- Flash interface (Master MICROWIRE handler)
- Microcontroller interface (Slave MICROWIRE handler)
- Codec interface

COMPACTSPEECH PROCESSOR COMMANDS—QUICK REFERENCE TABLE

Table 2-1: 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	None	-	None	-
CCIO	S	Configure Codec I/O	34	RESET, IDLE	No change	Config_value	1	None	-
CFG	S	Configure CompactSPEECH	01	RESET	No change	Config_value	2	None	-
CMSG	S	Create Message	33	IDLE	No change	Tag, Num_of_blocks	2+2	None	-
CMT	S	Cut Message Tail	26	IDLE	No change	Length of time	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, Config_value	2
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	Index	1	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	Recording time left	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 or DTMF	1	None	-
GTD	S	Get Time and Day	0E	IDLE	No change	Time/Day option	1	Time/day	2
GTM	S	Get Tagged Message	09	IDLE	No change	Tag_ref, Tag_Mask, Dir	2+2+1	Message found	1
INIT	S	Initialize System	13	RESET, IDLE	IDLE	None	-	None	-
INJ	S	Inject IVS data	29	RESET, 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	Message tag	2	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

Table 2-1: Speech Commands (Continued)

Command		Description	Opcode Hex	Source State	Result State	Command Parameters		Return Value	
Name	S/A					Description	Bytes	Description	Bytes
S	S	Stop	00	All states but RESET	IDLE	None	-	None	-
SAS	A	Say Argumented Sentence	1E	IDLE	SYNTHESIS	Sentence_n arg1	1 + 1	None	-
SB	S	Skip Backward	23	PLAY, IDLE*	No change	Length of time	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/Day	2	None	-
SF	S	Skip Forward	22	PLAY, IDLE*	No change	Length of time	2	None	-
SMSG	S	Set Message	30	IDLE, MSG_OPEN	MSG_OPEN	Num_of_pages	2	None	-
SMT	S	Set Message Tag	05	IDLE	No change	Message tag	2	None	-
SO	A	Say One Word	07	IDLE	SYNTHESIS	Word number	1	None	-
SPS	S	Set Playback Speed	16	PLAY, SYNTHESIS, IDLE	No change	Speed value	1	None	-
SS	A	Say Sentence	1F	IDLE	SYNTHESIS	Sentence_n	1	None	-
SV	S	Set Vocabulary Type	20	IDLE	No change	Mode, Id	1 + 1	None	-
SW	A	Say Words	21	IDLE	SYNTHESIS	N, $word_1$ . . . $word_n$	1 + N	None	-
TUNE	S	Tune	15	IDLE	No change	Index, Value	1 + 2	None	-
VC	S	Volume Control	28	PLAY, SYNTHESIS, IDLE, TONE_GENERATE	No change	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

THE STATE MACHINE

The CompactSPEECH processor functions as a state machine. 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).

The CompactSPEECH processor may be in one of the following states:

RESET

The CompactSPEECH processor is initialized to this state after a full hardware reset by the $\overline{\text{RESET}}$ signal (see "Resetting" on page 1-6). CompactSPEECH processor detectors (VOX, constant energy, call progress tones and DTMF tones) are not active. In all other states, the 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 CompactSPEECH processor starts executing the command.

PLAY

In this state a message is decompressed and played back.

RECORD

In this state a message is compressed and recorded into the message memory.

SYNTHESIS

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

TONE_GENERATE

The CompactSPEECH processor generates single or DTMF tones.

MSG_OPEN

The CompactSPEECH processor either reads or writes, 32 bytes from or to, the message memory, or sets the message Read/Write pointer on a 32-byte boundary.

CID

The CompactSPEECH processor receives the Caller ID data into an internal Caller ID buffer.

COMMAND EXECUTION

A CompactSPEECH processor command is represented by an 8-bit opcode. Some commands have parameters, and some 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 starts when the microcontroller sends an 8-bit opcode to the CompactSPEECH processor, followed by the command's parameters (if any).

The CompactSPEECH processor executes the command and, if required, transmits a return value to the microcontroller. Upon completion, the CompactSPEECH 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 CompactSPEECH processor switches to the appropriate state and executes the command until it is completed, or an S (Stop) or PA (Pause) command is received from the microcontroller.

When an asynchronous command execution is completed, the EV_NORMAL_END event is set, and the CompactSPEECH processor switches to the IDLE state.

"CompactSPEECH Processor Commands—Quick Reference Table" on page 2–2 provides a table which shows all the CompactSPEECH processor commands, the source states in which these commands are valid, and the result states which the CompactSPEECH processor enters as a result of the command.

STATUS WORD

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

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 that the $\overline{\text{MWRQST}}$ signal is active, it should issue the GSW (Get Status Word) command which deactivates the $\overline{\text{MWRQST}}$ signal. Then, the microcontroller should test the EV_ERROR bit in the status word, and, if it is set, send the GEW (Get Error Word) command to read the error word for details of the error.

For a detailed description of each of the CompactSPEECH processor commands, see "Command Description" on page 2–21.

TUNABLE PARAMETERS

The CompactSPEECH processor can be adjusted to your system's requirements. For this purpose the CompactSPEECH processor supports a set of tunable parameters, which 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 CompactSPEECH processor's operation, such as silence compression, tone detection, no-energy detection, etc.

Table 2-3 describes all the tunable parameters in detail. "Command Description" on page 2–21 describes the TUNE command.

MESSAGES

The CompactSPEECH processor message manager supports a wide range of applications which require different levels of DTAD functionality.

The message-organization scheme and the message tag support 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 the basic unit on which most of the CompactSPEECH processor commands operate. A CompactSPEECH processor message, stored on a Flash memory device, can be regarded as a computer file stored on a mass-storage device.

A message is created with either the R or the CMSG (Create Message) command.

When a message is created, it is assigned a time-and-day stamp, and a message tag which can be read by the microcontroller. Caller ID information can be automatically attached to the message by setting the CID_RECORD tunable parameter.

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 but not the codec. The data is trans-

ferred directly to the message memory. It is not compressed by the CompactSPEECH processor voice compression algorithm.

WMSG, RMSG (Read Message) and SMSG (Seek Message) are a complete set of message-data access commands that can be used to store and read data to/from any location in the message memory (see "Command Description" on page 2–21 for more details about these commands). Using these commands, messages can be used by the microcontroller to implement such features as a Telephone Directory and Caller Numbers List (Caller IDs of all 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, temporarily suspend the P and R commands, and then allow them to resume execution from where they were suspended.

CURRENT MESSAGE

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

Deleting the current message does not cause a different message to become current. The current message is undefined. If, however, 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.

MESSAGE TAG

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

The most significant bit of the message tag (bit 15) is used to indicate the speech compression rate. The microcontroller should program it before recording (0 for 5.2 Kbit/s, 1 for 7.3 Kbit/s). The CompactSPEECH processor reads the bit before message playback to select the appropriate decompression algorithm.

The GMT and SMT commands may be used to handle message tags.

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

This limitation is inherent in Flash memories, which only allow bits to be changed from 1 to 0 (changing bits from 0 to 1 requires a special erasure procedure, see "Command Execution" on page 2–4). However, the main reason for updating an existing tag is to mark a message as old, and this can be done by using one of the bits as a new/old indicator, setting it to 1 when a message is first created, and clearing it when necessary.

SPEECH COMPRESSION

The CompactSPEECH processor implements two speech compression algorithms. One algorithm with 5.2 Kbit/s compression rate which enables from 14 to 16 minutes of recording on a 4-Mbit device, while the other which uses a 7.3 Kbit/s compression rate supports 10 to 12 minutes of recording. Both compression rates assume 10 percent silence.

Before recording each message, the microcontroller can select one of the two algorithms by programming bit 15 of the message tag.

During message playback the CompactSPEECH processor reads this bit and selects the appropriate speech decompression algorithm.

IVS vocabularies can be prepared in either the 5.2 Kbit/s or the 7.3 Kbit/s compression format using the IVS tool. All the messages in a single vocabulary must be recorded using the same

algorithm. (See the *IVS User's Manual* for further details). During speech synthesis, the CompactSPEECH processor automatically selects the appropriate speech decompression algorithm.

TONE AND NO-ENERGY DETECTORS

The CompactSPEECH processor detects DTMF, busy and dial tones, constant energy level, and no-energy (VOX). This enables remote control operations and call progress. Detection is active throughout the operation of the CompactSPEECH 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

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 in "GSW Get Status Word" on page 2–33).

The DTMF detector performance, as measured on the line input using an ISD-TDB266-DAA board, is summarized below (see Table 2-2).

Table 2-2: DTMF Detector Performance[1]

	Play/IVS Synthesis	Record/Idle
Detection Sensitivity	Performance depends on the message being played.[2]	–36 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

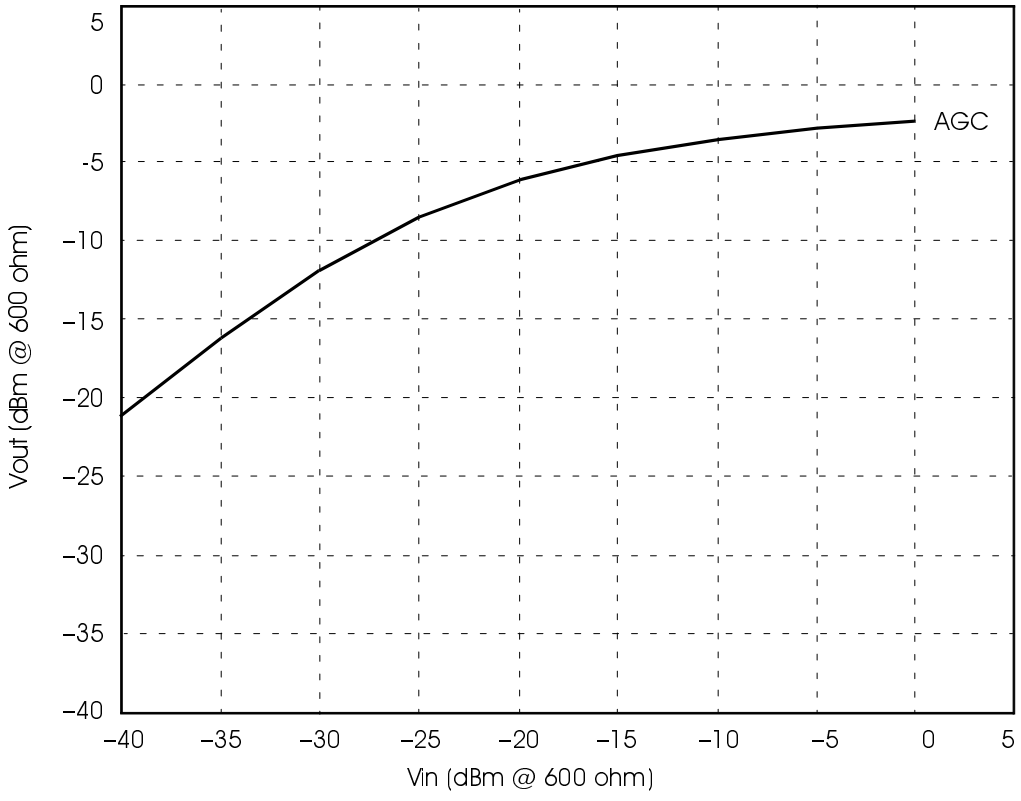
1. *Performance depends on the DAA design. For reliable DTMF detection:*
   - *The hardware AGC must be disabled during playback.*
   - *A hardware echo-canceler, that attenuates the echo by at least 6 dBm, is required during playback.*
   - *To achieve –36 dBm sensitivity in RECORD/IDLE state, use a hardware AGC that amplifies weak signals by 18 dBm. To pass Mitel tests, we also recommend that the AGC attenuate high signals (approximately 3 dBm for signals in the proximity of 0 dBm). See Graph 2-1 for the AGC's characteristics.*
   - *To achieve –34 dBm, sensitivity when playing a silence-message, the hardware should supply a fixed-gain of 6 dBm.*
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 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 DTMF tones is between 20 ms and 45 ms, separate detection is unpredictable.*
5. *Determined by the DTMF_REV_TWIST tunable parameter value.*

ECHO CANCELLATION

Echo cancellation is a technique used to improve the performance of DTMF tone detection during speech synthesis, tone generation, and OGM playback. For echo cancellation to work properly, AGC must not be active in parallel. To take advantage of echo cancellation the microcontroller must control the AGC, (i.e., disable the AGC during PLAY, SYNTHESIS, and TONE_GENERATE states and enable it again afterwards). If AGC can not be disabled, do not use echo cancellation. The microcontroller should use the CFG command to activate/deactivate echo cancellation. (For further details, see "Command Description" on page 2–21.)

Echo cancellation applies only to DTMF tones. Busy, constant energy, and dial-tone detection is not affected by this technique. This implies that the performance of the busy and dial-tone detector during message playback depends on the message being played.

Graph 2-1: Transfer Functions of AGC Circuit from Line



TUNABLE PARAMETERS

Tunable parameters control the detection of busy and dial tones, constant energy level (in the frequency range 200–3400 Hz), and no-energy. You should tune these parameters to fit your hardware. In addition, you may need to change the tunable parameters according to the setting (On or Off) of the Automatic Gain Control (AGC). For more information see the TUNE command in "TUNE Tune index parameter_value" on page 2–51.

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 0–1100 Hz. Graph 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. Figure 2-1 shows the default specification for a busy tone.

The tunable parameters are divided into five sets:

1. Busy tone on-time and off-time range specification:

   BUSY_MIN_ON_TIME
   BUSY_MIN_OFF_TIME
   BUSY_MAX_ON_TIME
   BUSY_MAX_OFF_TIME

2. Busy tone cadence control specification

   BUSY_VERIFY_COUNT
   BUSY_TONE_TYPE
   CADENCE_DELTA

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

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

   BUSY_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] < CADENCE_DELTA
   and
   [S1–S3] < CADENCE_DELTA

   The acceptance criteria for three cadences:

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

3. Busy and Dial Tone Energy Thresholds

   TONE_ON_ENERGY THRESHOLDS
   TONE_OFF_ENERGY THRESHOLDS

4. Busy Detection Time

   MIN_BUSY_DETECT_TIME

5. Improved DTMF Sensitivity.

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

   DTMF_GAIN_LEVEL_AT_IDLE_MODE—SW AGC for DTMF in idle/record modes. When incrementing the tunable by 1, the dynamic range is increased by 3 dB.

   DTMF_GAIN_LEVEL_AT_PLAY_MODE—SW AGC for DTMF detection in play and tone generation modes. When incrementing the tunable by 1, the dynamic range is increased by 3 dB.

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

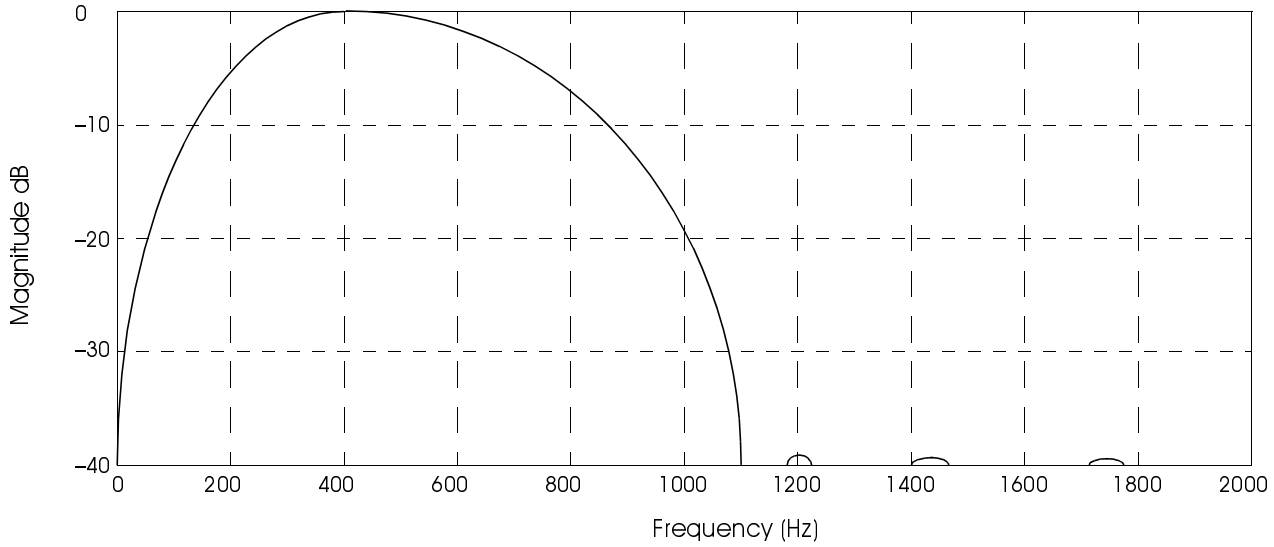
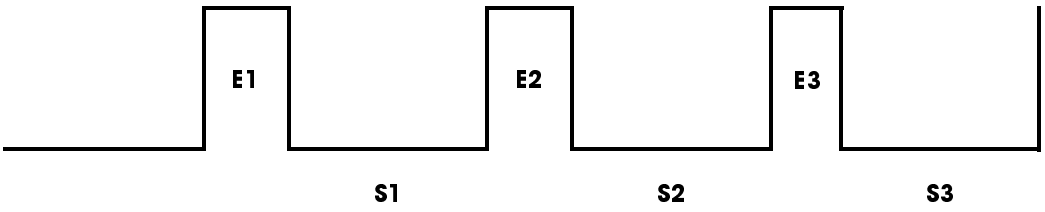


Figure 2-1: Busy-Tone Detector—Default Cadence Specification



[E1–E3] < 100 ms [S1–S3] < 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 200 Hz to 3400 Hz. It is intended to detect both white and pink noise, and can be used to detect line disconnection during recording.

We recommend that you 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_TIME > COUNT
CONST_NRG_TOLERANCE_TIME
CONST_NRG_LOW_THRESHOLD
CONST_NRG_HIGH_THRESHOLD

NO ENERGY (VOX)

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

We recommend that you use the constant-energy mechanism in conjunction with the no-energy (VOX) mechanism. The following tunable parameters control the operation of the no-energy (VOX) mechanism:

VOX_ENERGY_THRESHOLD
VOX_TIME_COUNT
VOX_TOLERANCE_TIME

TONE GENERATION

The CompactSPEECH processor can generate DTMF tones and single-frequency tones from 300 Hz to 3000 Hz in increments of 100 Hz. CompactSPEECH processor tone generation conforms to the EIA-470-RS standard. Note, however, that you may have to change the value of some tunable parameters in order to meet the standard specifications since the energy level of generated tones depends on the analog circuits being used.

- Tune the DTMF_TWIST_LEVEL parameter to control the twist level of the generated DTMF tones.
- Use the VC command, and tune the TONE_GENERATION_LEVEL parameter, to control the energy level at which these tones are generated.
- Use the GT command to specify the DTMF tones, and the frequency at which single tones are generated.
- 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.

CALLER ID

The CompactSPEECH Caller ID feature provides complete support for the Bellcore standard used in the United States, and meets the French, Japanese, Spanish, and Dutch standards as well.

The CompactSPEECH processor activates the Caller ID modem when it receives the ACI (Activate Caller ID) command from the microcontroller.

After the modem session has been completed, if successful, the CompactSPEECH processor reports an EV_NORMAL_END event. If any error is encountered during the session, the CompactSPEECH processor reports an ERR_CID error. In this case, details of the error are included in the Caller ID data.

Whether an error was detected or not, the Caller ID data is saved in an internal buffer, and can be retrieved by the microcontroller with the GCID command. See the description of the Caller ID buffer in the GCID command in "GCID Get CID src, offset, length" on page 2–27. If a message is recorded, the accompanying CID data can be automatically saved as part of the message. This is controlled by the CID_RECORD tunable parameter (see the TUNE command in "TUNE Tune index parameter_value" on page 2–51).

THE CALLER ID MODEM

For the Caller ID modem to function correctly, if either Bell202 or V.23 are used as the modem, AGC must not be active in parallel. The microcontroller must disable the AGC before activating the modem.

If DTMF is used as the physical layer, AGC must be active for best performance.

MESSAGE FORMAT

For the Bellcore standard, the CompactSPEECH processor supports CND (Caller Number Delivery) and CNAM (Calling Name Delivery) in SDMF (Single Data Message Format) and MDMF (Multiple Data Message Format).

For the French standard, Caller Number and Caller Name are supported in the "Call Message" service.

The CompactSPEECH processor supports the Notification messages of the French CID. All the parameters are similar to the parameters of the Identification messages, with the addition of the Command parameter. The presence of the Command parameter is reflected in the CID status byte. In addition, Absence of Name in the French CID is supported. This is indicated in the CID status byte (see the GCID command for details).

For the Dutch standard the Caller ID data includes a 17 digit telephone number.

For the Japanese Caller ID the buffer size is 130. CompactSPEECH processor holds two status bytes, and up to 128 bytes of CID information.

The CompactSPEECH processor provides the same interface to the Caller ID data using the GCID command regardless of the selected standard.

For more details about the ACID and GCID commands, and the various Caller ID specific tunable parameters, refer to "Command Description" on page 2–21.

TEST MODE

This mode of operation allows easier tuning of the analog circuits to the CID modem. It is only available for CID standards that use either Bell 202 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.

SPEECH SYNTHESIS

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

There are two kinds of predefined messages: fixed messages (e.g., voice menus in a voice-mail system) and programmable messages (e.g., 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 CompactSPEECH processor can use several vocabularies stored on an external storage device. IVS enables CompactSPEECH processor to synthesize messages with the same meaning, but in different languages, from separate vocabularies.

Among IVS features:

- Multiple vocabularies are 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 CompactSPEECH 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. On one hand, the words should be recorded and saved in a compressed form, and you would like to use the best voice compression for that purpose. On the other hand, the higher the compression rate, the worse 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 incoming messages (ICMs) in a DTAD environment). It is more pleasant to the human ear to hear them both in the same 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 only the vocabulary, but also the code to access it should be compact.

When designing for a multi-lingual environment, there are 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 CompactSPEECH processor uses to synthesize sentences.

Word Table

The words are the basic units in the vocabulary. You 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 that you wish to synthesize, 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. They should, therefore, 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 CompactSPEECH processor independent of the language being synthesized.

For example, if the Flash memory and/or ROM contain 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 CompactSPEECH 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 CompactSPEECH processor treats the first sentence in the sentence table, i.e., sentence 0, in a special way to support time-and-day stamp. It assumes that the sentence is designed for both system time and message time-and-day stamp announcement. The first sentence has one argument which is interpreted as follows:

0 System time is announced

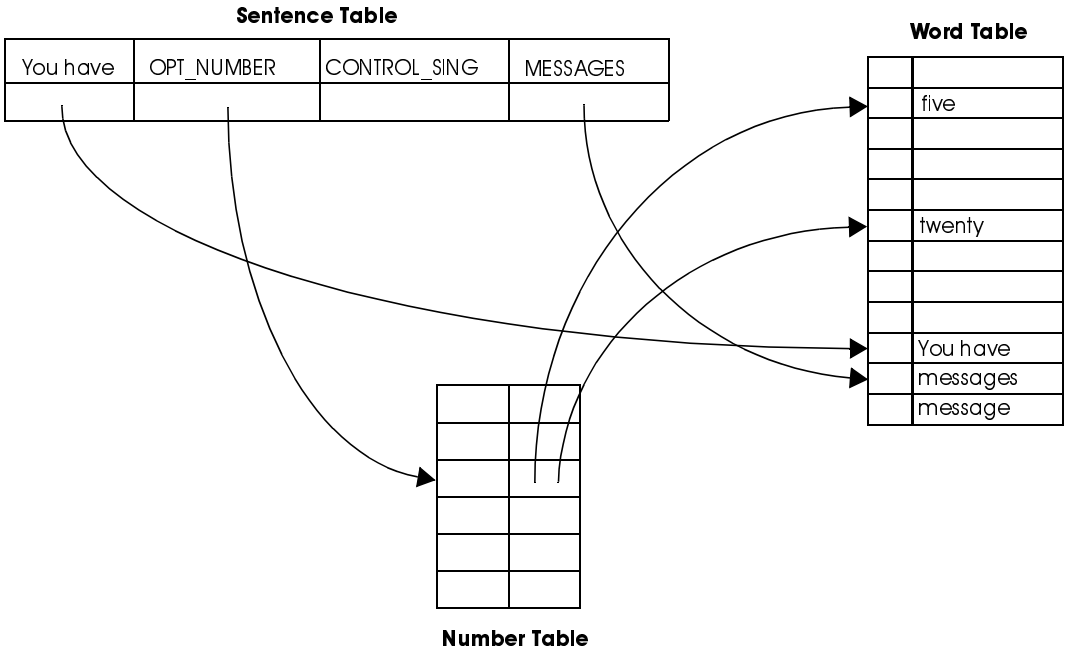
1 The time-and-day stamp of the current message is announced.

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 2-2 shows the interrelationship between the three types of tables:

Figure 2-2: The Interrelationship of a Word Table, a Sentence Table, and a Number Table



Control and Option Codes

The list of word indices alone cannot provide the entire range of sentences that the CompactSPEECH processor can synthesize. IVS control and option codes are used as special instructions that control the behavior of the speech synthesis algorithm in the CompactSPEECH processor.

For example, if the sentence should announce the time of day, the CompactSPEECH 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 CompactSPEECH processor to take special actions.

THE IVS TOOL

The IVS tool includes two utilities:

- The DOS-based IVS Compiler.
- IVSTOOL for Windows. A Windows 3.1 based utility.

The tools allow you to create vocabularies for the CompactSPEECH processor. They take you all the way 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). It allows 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. This information can be burned into an EPROM or Flash memory for use by the CompactSPEECH software.

Voice Compression

Each IVS vocabulary can be compiled with either the 5.2 Kbit/s or the 7.3 Kbit/s voice compression algorithm. You define the compression rate before compilation. The CompactSPEECH processor automatically selects the required voice decompression algorithm when the SV command is used to select the active vocabulary.

Graphical User Interface (GUI)

The IVS package includes a Windows utility that assists the vocabulary designer to synthesize sentences. With this utility, you can both compose sentences and listen to them.

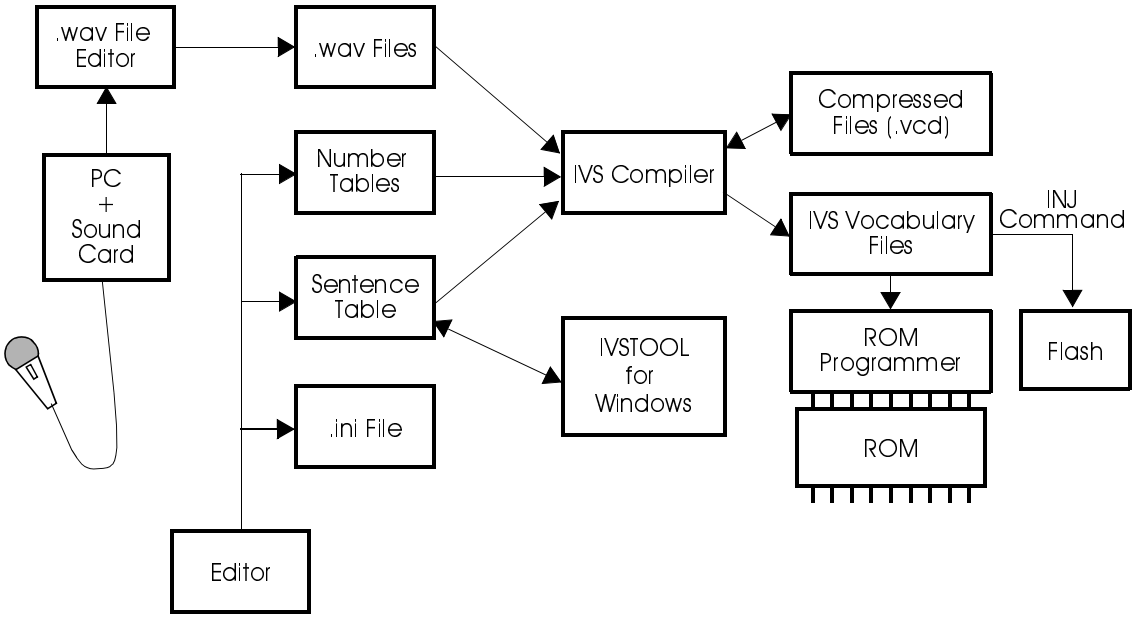
HOW TO USE THE IVS TOOL WITH THE COMPACTSPEECH PROCESSOR

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 command. The CompactSPEECH processor SV command is used to select the required vocabulary. The SW, SO, SS, and SAS commands are used to synthesize the required word or sentence. The typical vocabulary-creation process is as follows:

1. Design the vocabulary.
2. Create the vocabulary files (as described in detail below). Use IVSTOOL for Windows 3.1 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 the .wave files as well as 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.
6. 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 memory device.
7. Once the vocabulary is in place, the speech synthesis commands of the CompactSPEECH processor can be used to synthesize sentences.

Figure 2-3 shows the vocabulary-creation process for a single table on a ROM or Flash memory device.

Figure 2-3: Creation of an IVS Vocabulary



INITIALIZATION

Use the following procedures to initialize the CompactSPEECH processor:

1. **Failure Mode**Reset the CompactSPEECH processor by activating the $\overline{\text{RESET}}$ signal. (See "Resetting" on page 1-6.)
2. Issue a CFG (Configure CompactSPEECH processor) command to change the configuration according to your environment.
3. Issue an INIT (Initialize System) command to initialize the CompactSPEECH firmware.
4. Issue a series of TUNE commands to adjust the CompactSPEECH processor to the requirements of your system.

MICROWIRE SERIAL INTERFACE

MICROWIRE/PLUS™ is a synchronous serial communication protocol, originally implemented in National Semiconductor's COPS™ and HPC™ families of microcontrollers to minimize the number of connections, and thus the cost, of communicating with peripherals.

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

The microcontroller is the protocol master, and provides the clock for the protocol. The CompactSPEECH 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 CompactSPEECH processor and the microcontroller.

Communication is handled in bursts of eight bits (one byte). In each burst the CompactSPEECH processor is able to receive and transmit eight bits of data. After eight bits have been transferred, an

internal interrupt is issued for the CompactSPEECH processor to process the byte, or to prepare another byte for sending. In parallel, the CompactSPEECH 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 CompactSPEECH processor. When the CompactSPEECH processor transmits data, it expects to receive the value 0xAA before each transmitted byte. The CompactSPEECH processor reports any status change by clearing the $\overline{\text{MWRQST}}$ signal to 0.

If a parameter of a CompactSPEECH processor command is bigger than one byte, the microcontroller should transmit the Most Significant Byte (MSB) first. If a return value is bigger than one byte, the CompactSPEECH processor transmits the MSB first.

SIGNAL DESCRIPTION

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

INPUT SIGNALS

MWDIN

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

MWCLK

This signal 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's rising edge. The transmitted data is output on MWDOUT on the clock's 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 CompactSPEECH processor is being accessed. Setting $\overline{\text{MWCS}}$ to 0 causes the CompactSPEECH 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 CompactSPEECH processor and the microcontroller.

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

OUTPUT SIGNALS

MWDOUT

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

$\overline{\text{MWRDY}}$

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

This signal is set to 1 by the CompactSPEECH processor after each byte transfer has been completed. It remains 1, while the CompactSPEECH 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 CompactSPEECH 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 CompactSPEECH 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 CompactSPEECH processor reads data from MWDIN on every rising edge of MWCLK. The CompactSPEECH 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 CompactSPEECH 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 CompactSPEECH processor) after 8-bits of data were transferred to/from the CompactSPEECH 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 CompactSPEECH processor when it is ready to receive the first parameter byte (if there are any parameters) and so on until the last parameter byte is transferred. An active $\overline{\text{MWRDY}}$ signal after the last parameter byte 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 then re-activated when the CompactSPEECH processor is ready to send another byte or to receive a new command.

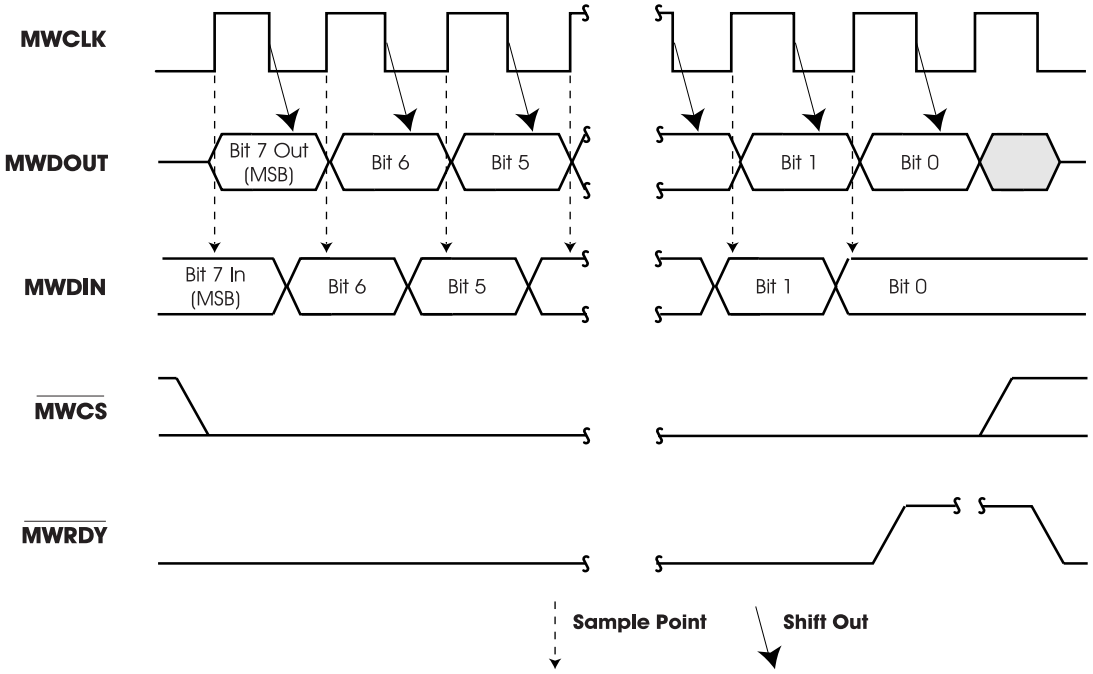
   The $\overline{\text{MWRDY}}$ signal is activated (cleared to 0) after reset and after a protocol time-out. (See "INTERFACE PROTOCOL ERROR HANDLING" on page 2–19.)

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 CompactSPEECH processor receives a GSW command.

Figure 2-4 illustrates the sequence of activities during a MICROWIRE data transfer.

Figure 2-4: Sequence of Activities During a MICROWIRE Byte Transfer



INTERFACE PROTOCOL ERROR HANDLING

Interface Protocol Time-outs

Depending on the CompactSPEECH processor's state, if more than 100 milliseconds elapse between the assertion of the $\overline{\text{MWRDY}}$ signal and the transmission of the eighth bit of the next byte of the same command transaction, a time-out event occurs, and the CompactSPEECH 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 CompactSPEECH processor echoes back to the microcontroller all the bits received by the CompactSPEECH processor. Upon detection of an error in the echo the microcontroller should stop the protocol clock, which will eventually cause a time-out error (i.e., ERR_TIMEOUT bit is set in the error word).

NOTE *When a command has a return value, the CompactSPEECH processor transmits bytes of the return value instead of the echo value.*

The CompactSPEECH processor transmits a byte as an echo when it receives the value 0xAA from the microprocessor. Upon detection of an error, the CompactSPEECH processor activates the $\overline{\text{MWRQST}}$ signal and sets the ERR_COMM bit in the error word.

THE MASTER MICROWIRE INTERFACE

The CompactSPEECH processor's Master MICROWIRE controller implements the MICROWIRE/PLUS interface in master mode. It enables the CompactSPEECH processor to control Flash memory devices. Several devices may share the Master MICROWIRE channel. This can be implemented by connecting device selection signals to general purpose output ports.

MASTER MICROWIRE DATA TRANSFER

Signals

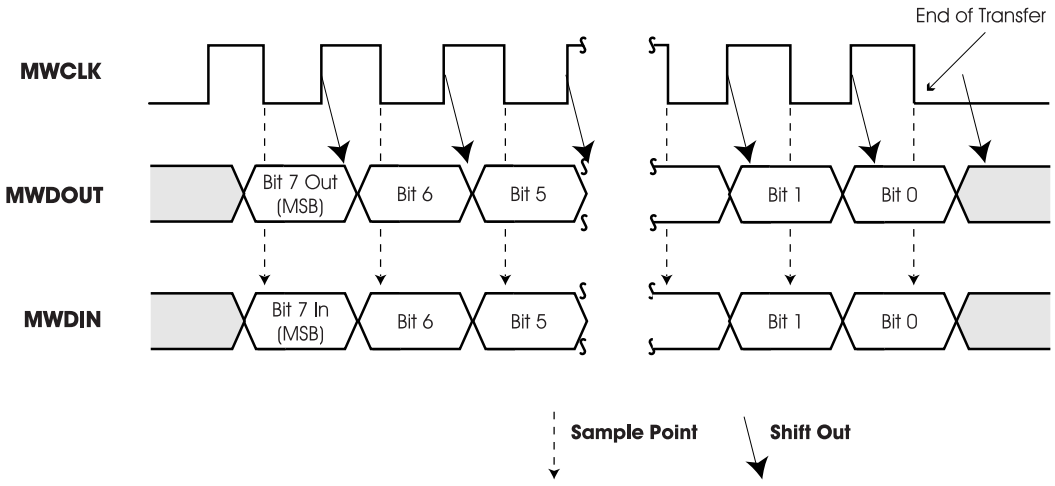
The Master MICROWIRE controller's signals are the Master MICROWIRE serial clock (MMCLK), the Master MICROWIRE serial data out (MMDOUT) signal and the Master MICROWIRE serial data in (MMDIN) signal.

The Master MICROWIRE controller can handle up to four Flash memory devices. The CompactSPEECH processor uses the signals, CS0–CS3, as required for the number of devices in use, as device chip-select signals.

Clock for Master MICROWIRE Data Transfer

Before data can be transferred, the transfer rate must be determined and set. The rate of data transfer on the Master MICROWIRE is determined by the Master MICROWIRE serial clock (MMCLK) signal. This rate is the same as the Codec clock (CCLK) signal. As long as the Master MICROWIRE is transferring data, the codec interface must be enabled and its sampling rate should not be changed.

Figure 2-5: Master MICROWIRE Data Transfer



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 retrieve 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 CompactSPEECH processor. For commands which require a return value from the CompactSPEECH processor, the start of the return value is indicated by a thick vertical line.*

ACID Activate Caller ID

When the CompactSPEECH processor receives the ACID command: it clears the Caller ID buffer, activates the Caller ID modem, and enters the CID state. When the Caller ID physical layer is Bell202 or V.23, this 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 physical layer is DTMF the command should be sent by the microcontroller after it detects voltage polarity change on the line.

When the modem session has been completed and the Caller ID buffer loaded with the new Caller ID data, the CompactSPEECH processor deactivates the modem and reports an EV_NORMAL_END event. If the second ring occurs before the CompactSPEECH processor reports an event, the microcontroller should terminate the Caller ID modem session with the S command.

If the modem session is not successful, the CompactSPEECH processor sets the ERR_CID bit in the error word, and reports an EV_ERROR event. The microcontroller can still retrieve the contents of the Caller ID buffer using the GCID command. The error details are in the Caller ID status byte (bytes 0 and 1 of the Caller ID buffer).

If the modem session is terminated with the S command, the contents of the Caller ID buffer are undefined.

NOTE: *When ACID starts execution, the previous contents of the Caller ID buffer are destroyed.*

Before activating the ACID command, AGC must be OFF for Dutch Caller ID, and ON for others.

For French CID, the message type time-out is measured from the end of the SMMR (see the tunable parameter CID_PARAM3).

Example

ACID		
Byte sequence:	Microcontroller	2D
	CompactSPEECH	2D
Description:	Activate Caller ID modem.	

CCIO Configure Codec I/O *config_value*

Configures the voice sample paths in various states. It should be used to change the default CompactSPEECH processor configuration.

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 codec.

Bits 1–7

Reserved.

Example

CCIO 01			
Byte sequence:	Microcontroller	34	01
	CompactSPEECH	34	01

CFG Configure CompactSPEECH *config_value*

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

The *config_value* parameter is encoded as follows:

Bit 0

Codec configuration.

0 Short-frame format (default).

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

Bit 1

Reserved.—must be cleared to 0.

Bit 2

Echo cancellation control

0 Echo cancellation off (default).

1 Echo cancellation is on during playback.

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

Bit 3

Reserved.—must be cleared to 0.

Bits 4–5

00 Reserved

01 Reserved

10 Toshiba's TC58A040F Flash

11 Samsung's KM29N040T Flash

The default value is 10, the Serial Flash.

Bits 6–7

Reserved—must be cleared to 00.

Bits 8–10

Number of installed Flash devices Valid range 1 . . . 4 Flash devices. Default is 1.

Bit 11

Caller ID Test Mode selection

0 Normal Mode

1 Test Mode: the carrier presence on pin CIDCI the CID bit value on pin CIDBL

Bits 12–14

Caller ID Application layer.

000 US, Bellcore

001 French

010 Dutch

011 Japanese

100 Spanish

Bit 15

Reserved—must be cleared to 0.

NOTE: *The CompactSPEECH processor automatically detects the type of Flash memory device in use, i.e., TC58A040F.*

Example

CFG 0324				
Byte sequence:	Microcontroller	01	03	24
	CompactSPEECH	01	03	24
Description:	Configure the CompactSPEECH to work with: Codec that supports short-frame format. Three, TC58A040F, Flash devices. Echo cancellation on.			

CMSG Create Message *tag num_of_blocks*

Creates a new message with a message tag *tag*, allocates *num_of_blocks* 4 Kbytes blocks for it, and sets the message pointer to the beginning of the message data. The command switches the CompactSPEECH 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 i.e., the CompactSPEECH 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. The message contents can be modified with the WMSG command, and read with the RMSG command.

The microcontroller must issue an S command to close the message and switch the CompactSPEECH 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 not enough memory and the CompactSPEECH processor can not allocate the required memory space for the message, EV_MEMLOW is set in the status word and no message is created.

Example

CMSG 0101 01						
Byte sequence:	Microcontroller	33	01	01	00	01
	CompactSPEECH	33	01	01	00	01
Description:	Create a new message, and allocate 4 Kbytes for its data.					

CMT Cut Message Tail *time_length*

Cut *time_length* units, each of 10 ms duration, off the end of the current message. The maximum value of *time_length* is 6550. Cut-time accuracy is ±0.14 seconds.

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 is not deleted. Use the DM (Delete Message) or DMS (Delete Messages) commands to delete the message.*

A compressed frame represents 26.5 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.

Example

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

CVOC Check Vocabulary

Checks (checksum) if the IVS data was correctly programmed to the ROM or Flash memory 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
	CompactSPEECH	2B	01
Description:	Check the current vocabulary. The CompactSPEECH 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
	CompactSPEECH	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 message is immediately available for recording new messages.

Example

DMS FFC2 003F						
Byte sequence:	Microcontroller	0B	FF	C2	00	3F
	CompactSPEECH	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 6–14: not used Bit 15: compression rate *Note: the description of the tag is an example only. All bits of the tag are user-definable.*					

GCFG Get Configuration Value

Returns a sequence of two bytes with the following information:

Bits 0–7

Magic number, which specifies the CompactSPEECH firmware version.

Bits 8–9

Memory type.

00 Reserved

01 Reserved

10 Toshiba's TC58A040F Flash

11 Samsung's KM29N040T Flash

The command should be used together with the CFG and INIT commands during CompactSPEECH processor initialization. See the CFG command for more details, and an example of a typical initialization sequence.

Example

GCFG				
Byte sequence:	Microcontroller	02	AA	AA
	CompactSPEECH	02	02	01
Description:	Get the CompactSPEECH processor magic number. The CompactSPEECH processor responds that it is Version 1, with Serial Flash			

GCID Get CID *src, offset, length*

Returns the Caller ID data.

src may be one of the following:

0 returns the current content of the Caller ID buffer.

1 returns the Caller ID 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 Caller ID buffer, for all except the Japanese CID, is shown below.

Note that the Dutch Caller ID contains two status bytes, and up to 17 bytes of caller number.

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 CompactSPEECH 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 (Set by the tunable parameter CID_PARAM4).

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 CompactSPEECH 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 (S).

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

Bit 1 Checksum error.

Bit 2 No carrier.

Bit 3 Time-out 0 (See the CID_PARAM0 tunable parameter for details) (E).

Bit 4 Time-out 1 (See the CID_PARAM1 tunable parameter for details) (E).

Bit 5 Time-out 2 (See the CID_PARAM2 tunable parameter for details) (E).

Bit 6 Reserved.

Bit 7 Field overflow (W).

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

'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).

The Japanese Caller ID 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_PARAM0 time has elapsed from the ACID command until the end of the message.

Bit 4 More than tunable parameter CID_PARAM1 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

Caller ID 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						
Byte sequence:	Microcontroller	2E	00	AA	...	AA
	CompactSPEECH	2E	00	48-byte Caller ID data		
Description:	CompactSPEECH returns 48 bytes of Caller ID data.					

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 CompactSPEECH 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 CompactSPEECH 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 Caller ID detection.

Example

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

GI Get Information *item*

Returns the 16-bit value specified by item from one of the internal registers of the CompactSPEECH 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 Graph 2-2. 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
	CompactSPEECH	25	00	00	06
Description:	Get the duration of the last detected DTMF tone. The CompactSPEECH processor responds: 60 MS				

GL Get Length

Returns the length of the current message in multiples of 32 bytes.

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 Flash 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 i.e., an empty message occupies 4 Kbytes (the message length is: 4096/32 = 128).

Example

GL					
Byte sequence:	Microcontroller	19	AA	AA
	CompactSPEECH	19	02	00
Description:	Get the length of the current message. The CompactSPEECH processor responds: 512 i.e., the message occupies 16384 (512 * 32) bytes			

GMS Get Memory Status *type*

Returns the estimated total remaining recording time in seconds as a 16-bit unsigned integer. This estimate assumes 5.2 Kbit/s with no silence compression: a real recording may be longer, according to the amount of silence detected and compressed.

The return value is dependent on the value of the *type* parameter as follows:

0 The remaining recording time is returned.

1 Returns 0 (For compatibility only).

2 Same as 0 (For compatibility only).

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

Example

GMS 0					
Byte sequence:	Microcontroller	12	00	AA	AA
	CompactSPEECH	12	00	01	40
Description:	Return the remaining recording time. The CompactSPEECH processor responds: 320 seconds				

GMT Get Message Tag

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

Example

GMT				
Byte sequence:	Microcontroller	04	AA	AA
	CompactSPEECH	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 CompactSPEECH 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 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.

The *tag_ref* and *tag_mask* parameters are each two bytes; the return value is also 2-byte long.

For example, if *tag_ref* = 42_{16}, and *tag_mask* = $3F_{16}$, the number of existing old messages whose user-defined tag is 2 is returned. See "MESSAGE TAG" on page 2–6 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
	CompactSPEECH	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. CompactSPEECH processor responds that there are five messages which satisfy the request.							

GSW Get Status Word

Returns the 2-byte status word.

Status Word

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

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

EV_DIALTONE

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

EV_VOX

1 = A period of silence (no energy) was detected on the telephone line. Use this indicator for line disconnection. (See VOX_TIME_COUNT in Table 2-3.)

EV_CONST_NRG

1 = A period of constant energy was detected. Use this indicator for line disconnection. (See CONST_NRG_TIME_COUNT in Table 2-3.)

EV_RESET

When the CompactSPEECH 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 CompactSPEECH processor, it indicates an internal CompactSPEECH 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
	CompactSPEECH	14	00	40
Description:	Get the CompactSPEECH processor Status Word (typically sent after the $\overline{\text{MMRQST}}$ signal is asserted by the CompactSPEECH processor which indicates a change in the status word). The CompactSPEECH processor responds that the memory is full.			

GT Generate Tone *tone*

Generates the tone specified by the 1-byte tone parameter, until an S command is received.

Specify the tone by setting the bits of tone as follows:

Bit 0

1

Bits 1–4

DTMF code.

Where the DTMF code is encoded as follows:

Value (Hex) 0 to 9	*DTMF Digit 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 (300 through 3000 Hz).

Bits 6, 7

0

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

Example

GT 0D20			
Byte sequence:	Microcontroller	0D	20
	CompactSPEECH	0D	20
Description:	Generate a single-frequency 1600 Hz 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 of day 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

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
	CompactSPEECH	0E	01	E8	29
Description:	Get the current message time-and-day stamp. The CompactSPEECH 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} message with a given tag to be the current message you must first select the first message that complies with the above equation, and then issue another GTM command with n – 1 as a parameter, to skip to the n^{th} message.*

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 change 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 0								
Byte sequence:	Microcontroller	09	FF	CE	00	3F	00	AA
	CompactSPEECH	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 CompactSPEECH processor return value indicates that there is such a message. The following pseudo-code demonstrates how to play all new ICMs. The messages are marked after being played.
	
	In mailbox number 6:
	
	Return_val = GTM(FFCE, 003F, 1)
	While (ReturnVal == TRUE)
	Begin
	P() /* Play */
	Message_tag = GMT() /* get message tag */
	SMT(FFF7) /* Mark the message as 'old' */
	GTM(FFCE, 003F, 1) /* Get next with same tag */
	End							

INIT Initialize System

Execute this command after the CompactSPEECH processor has been configured (see CFG and GCFG commands).

Performs a soft reset of the CompactSPEECH processor 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
	CompactSPEECH	13
Description:	Initialize the CompactSPEECH 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 CompactSPEECH 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 plus 100 bytes, two blocks of the Flash are not available for message recording).

Example

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

MR Memory Reset

Erases all good Flash blocks and initializes the CompactSPEECH processor (i.e., does exactly what the INIT command does). 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: *The command erases all messages and should be used with care.*

Example

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

P Playback

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

If the current message is undefined, ERR_INVALID is reported.

Example

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

PA Pause

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

NOTE: *DTMF and tone detectors remain active during Pause.*

Example

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

PDM Go To Power-down Mode

Switches the CompactSPEECH processor to power-down mode (see "Power-down Mode" on page 1-8 for details). Sending any command while in power-down mode resets the CompactSPEECH processor detectors, and returns the CompactSPEECH processor to normal operation mode.

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

Example

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

R Record *tag*

Records a new message with message tag *tag*. The CompactSPEECH 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

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 contents of the Caller ID buffer are automatically attached to each message. After command execution, the contents of the buffer are undefined. It is the microcontroller's responsibility to trace the validity of the Caller ID data. The message tag can be used for this purpose.

Example of a typical recording session:

- (ICM) The microcontroller detects the first ring.
- (ICM) The microcontroller sends the ACID command to activate the Caller ID modem.
- (ICM) The CompactSPEECH processor reports EV_NORMAL_END to indicate a successful Caller ID session. The contents of the Caller ID buffer are valid.
- (ICM, OGM, memo) The microcontroller sends the R command, and uses one of the message tag bits to indicate the Caller ID validity. When recording a memo or an OGM, the Caller ID modem is not usually activate, and the contents of the Caller ID buffer are therefore meaningless. Later, when the microcontroller uses the GCID command to retrieve the Caller ID data, it should read the message tag to verify that the Caller ID data is valid.

Example

R 000E				
Byte sequence:	Microcontroller	0C	00	0E
	CompactSPEECH	0C	00	0E
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.			

RDET Reset Detectors *detectors_reset_mask*

Resets the CompactSPEECH 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
	CompactSPEECH	2C	20
Description:	Reset the VOX detector.		

RES Resume

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

Example

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

RMSG Read Message *data*

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

If the CompactSPEECH processor was in the IDLE state, the command opens the current message, switches the CompactSPEECH 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 CompactSPEECH 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 CompactSPEECH processor switches to the IDLE state. In this case, the return value is undefined and should be ignored.

Example

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

RRAM Read Memory

Exists for compatibility only. Use RMSG instead.

S Stop

Stops execution of the current command and switches the CompactSPEECH processor to the IDLE state. S may be used to stop the execution of SMSG, WMSG, RMSG all asynchronous commands.

Example

S		
Byte sequence:	Microcontroller	00
	CompactSPEECH	00
Description:	Stop current activity (e.g., playback, recording) and put the CompactSPEECH processor in IDLE state.	

SAS Say Argumented Sentence *sentence_n arg*

Announces sentence number *sentence_n* of the currently selected vocabulary, and passes *arg* to it. *sentence_n* and *arg* are each 1-byte long.

When playing is complete, the CompactSPEECH processor sets the EV_NORMAL_END bit in the status word, and activates the $\overline{\text{MWRQST}}$ signal.

If the current vocabulary is undefined, ERR_INVALID is reported.

Example

SAS 00 03				
Byte sequence:	Microcontroller	1E	00	03
	CompactSPEECH	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, each of 0.2 seconds duration, and causes message playback to pause. *time_length* is a 2-byte parameter that can have any value up to 320, i.e., 64 seconds. The skip accuracy is 5%. This command is meaningful only in the PLAY state. The RES command must be issued to continue playback.

If the beginning of the message is detected during the execution of the SB command, execution is terminated, the EV_NORMAL_END bit in the status register is set, the $\overline{\text{MWRQST}}$ signal is activated, and the CompactSPEECH 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 19			
Byte sequence:	Microcontroller	23	19
	CompactSPEECH	23	19
Description:	Skip back 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.

dest can be either 0 or 1.

0 Save the data into the Caller ID buffer.

1 Save the data in the message directory.

Offset shows where to start saving in the destination buffer (‘*dest*’)

len Shows number of bytes to save.

data *len* bytes to be saved.

Valid range: *len* is non-zero and 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 0048	
Description:	Stores 48 bytes of Caller ID data in the Caller ID buffer.

SDET Set Detectors Mask *detectors_mask*

Controls the reporting of detection for tones and VOX 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.

Bits 2–3

Reserved. Must be cleared to 0.

Bit 4

Report detection of a constant energy.

Bit 5

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

(The VOX attributes are specified with the tunable parameters VOX_TIME_COUNT and VOX_ENERGY_LEVEL.)

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 A3			
Byte sequence:	Microcontroller	10	A3
	CompactSPEECH	10	A3
Description:	Set reporting of all CompactSPEECH 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
	CompactSPEECH	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 bits 0–13 in 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

These bits 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
	CompactSPEECH	0F	DE	09
Description:	Set time and day to Monday 1.30 A.M.			

SF Skip Forward *time_length*

Skips forward in the current message *time_length* units, each of 0.2 seconds duration, and causes message playback to pause. *time_length* is a 2-byte parameter that can have any value up to 320, i.e., 64 seconds. The skip accuracy is 5 percent. This command is meaningful only in the PLAY state. The RES command must be issued to continue playback.

If the end of the message is detected during execution of SF, execution of the command is terminated and the EV_NORMAL_END bit in the status word is set, the $\overline{\text{MWRQST}}$ signal is activated and the CompactSPEECH 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 19				
Byte sequence:	Microcontroller	22	00	19
	CompactSPEECH	22	00	19
Description:	Skip forward five seconds from the current position in the message being played.			

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 (*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 CompactSPEECH processor switches to the IDLE state.

Example

SMSG 10				
Byte sequence:	Microcontroller	30	00	0A
	CompactSPEECH	30	00	0A
Description:	Set the message pointer to 32 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 2–6.

To change the tag of a message, we recommend that you read the message 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.

Bit 15 of the message tag is used to select the voice compression algorithm and should not be modified after recording.

Example

SMT FF F7				
Byte sequence:	Microcontroller	05	FF	F7
	CompactSPEECH	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 CompactSPEECH 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.

When playback of the selected word has been completed, the CompactSPEECH processor sets the EV_NORMAL_END bit in the status word, and activates the $\overline{\text{MWRQST}}$ signal.

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
	CompactSPEECH	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 CompactSPEECH 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.

Note that a negative *speed* value represents an increase in speed, a positive value represents a decrease in speed.

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.

If *speed* is not in the –6 to +6 range, ERR_PARAM is set in the error word.

Example

SPS FB			
Byte sequence:	Microcontroller	16	FB
	CompactSPEECH	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.

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

When playing has been completed the CompactSPEECH processor sets the EV_NORMAL_END bit in the status word and activates the $\overline{\text{MWRQST}}$ signal.

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
	CompactSPEECH	1F	00
Description:	Announce the first sentence in the sentence table of the currently selected vocabulary.		

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 memory.
All others	Reserved.

The host is responsible for selecting the current vocabulary, with SV, before using an SO, SW, SS or SAS command.

Each external vocabulary table has a unique *id* which is part of the vocabulary internal header (See the *IVS User's Manual* for more details). If *type* is 1 or 2, the CompactSPEECH 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
	CompactSPEECH	20	02	03
Description:	Select the vocabulary with vocabulary-id 3, which resides on Flash memory, as the current vocabulary.			

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

Plays *n* words, indexed by *word*$_1$ to *word*$_n$. On completion, the EV_NORMAL_END bit in the status word is set, and the $\overline{\text{MWRQST}}$ signal goes low.

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
	CompactSPEECH	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 CompactSPEECH 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.*

Table 2-3 describes the tunable parameters, their index numbers and their default values.

Table 2-3: Tunable Parameters

Index	Parameter Name	Description	Default
0	CID_PARAM0	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
1	CID_PARAM1	For Caller ID physical layer Bell202 or V.23: Time-out for SMMR 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 Caller ID: value should be 152 assuming zero delay between the end of the first ring and ACID activation. Spanish Caller ID: recommended value is 204 (2 ms standard recommendation +40 ms for CID modem delay). The default value is for the US Caller ID where the time-out is not defined. For Caller ID 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 Caller ID: 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_PARAM2	For Caller ID physical layer Bell202 or V.23: minimum length of the SMMR signal in 10 ms units. Accuracy is ±10 ms. French Caller ID: the value should be 8. The default value is for the US and Spanish Caller ID, where the time-out is not defined. For Caller ID physical layer DTMF: Maximum time between two consecutive DTMF codes. Dutch Caller ID: 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 2-3: Tunable Parameters (Continued)

Index	Parameter Name	Description	Default
3	CID_PARAM3	For Caller ID Physical Layer Bell202 Or V.23: Time-out for message type detection after valid SMMR signal is 10 ms units. Accuracy is ±10 ms. French Caller ID: the value should be 46. The default value is for the US and Spanish Caller ID 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 Caller ID with DTMF physical layer: Defines the maximum number of DTMF codes (digits) in the message body that is supported by the standard. Dutch Caller ID: the value should be 17. Spanish Caller ID: 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 Caller ID data will be set. (See the GCID command for more details.) *Legal values: 0 to 46.*	0
4	_SIL_THRESHOLD	Prevents speech from being interpreted as silence. The silence detection algorithm has an adaptive threshold, which is changed according to the noise level. This parameter is, therefore, only the initial threshold level. *Legal values: 9216 to 13824 in 512 (6 dB) steps.*	11264
5	_SIL_THRESHOLD_STEP	Defines the adaptive threshold changes step. If this threshold is too low, the threshold converges too slowly. If it is too high, silence detection is highly sensitive to any noise. *Legal values: 3 to 48.*	12
6	_SIL_BURST_THRESHOLD	The minimum time period for speech detection during silence. As this threshold increases, the time period interpreted as silence increases. If this threshold is too low, a burst of noise is detected as speech. If it is too high, words may be partially cut off. *Legal values: 1 to 3.*	2
7	_SIL_HANG_THRESHOLD	The minimum time period for silence detection, during speech. As this threshold increases, the time period interpreted as silence decreases. If this threshold is too low, words may be partially cut off. If it is too high, no silence is detected. *Legal values: 8 to 31.*	15
8	_SIL_ENABLE	Silence compression control. 0 turns silence compression off.	1

Table 2-3: Tunable Parameters (Continued)

Index	Parameter Name	Description	Default
9	_ENERGY_FACTOR	Determines the energy level used to synthesize silence. For the default value, the energy levels of the synthesized silence and the recorded silence are the same. If you divide (multiply) the default value by two, the synthesized silence is 6 dB less (more) than the level of the recorded silence. *Legal values: 1024 to 16384.*	8192
10	VOX_ENERGY_THRESHOLD	This parameter determines the minimum energy level at which voice is detected. Below this level, it is interpreted as silence. *Valued* — *BV* 1 — –59 2 — –55.2 3 — –52.7 4 — –50.2 6 — –48.6 12 — –45.3 24 — –42.2 48 — –39.5 96 — –36.4 192 — –33.4 384 — –30.3 768 — –27.4 1536 — –24.4 3072 — –18.4 12288 — –15.4 24596 — –12.4 32767 — –11.1 *Legal values: 1 to 32767.*	12
11	Reserved.		
12	VOX_TIME_COUNT	This parameter, in units of 10 ms, determines the period of silence before the CompactSPEECH processor reports silence. The accuracy of the constant is ±10 ms. *Legal values: 0 to 65535.*	700
13–15	Reserved.		
16	TONE_GENERATION_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_GENERATION_LEVEL and the VOL_LEVEL variable, controlled by the VC command. The tones are distorted when the level is set too high. *Legal values: 0 ≤ TONE_GENERATION_LEVEL + VOL_LEVEL ≤ 12.*	6

Table 2-3: Tunable Parameters (Continued)

Index	Parameter Name	Description	Default
17	DTMF_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: 32 to 4096*	32
18	TONE_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	TONE_ON_ENERGY_ THRESHOLD	Minimum energy level at which busy and dial tones are detected as On (after 700 Hz 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 400 Hz 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
20	TONE_OFF_ENERGY_ THRESHOLD	Maximum energy level at which busy and dial tones are detected as Off (after 700 Hz 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_ON_ENERGY_THRESHOLD. *Legal values: 0 to 65535.*	110
21	VCD_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_LEVEL and the VOL_LEVEL variable, controlled by the VC command. Speech is distorted when the level is set too high. *Legal values: 0 ≤ VCD_LEVEL + VOL_LEVEL ≤ 12.*	6
22	VOX_TOLERANCE_TIME	Controls the maximum energy-period, in 10 ms units, that does NOT reset the VOX detector. *Legal values: 0 to 255.*	3

Table 2-3: Tunable Parameters (Continued)

Index	Parameter Name	Description	Default
23	MIN_BUSY_DETECT_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
24	ECHO_DELAY	The near-echo delay in samples. The sampling rate is 8000 Hz (i.e., 125 µs per sample). *Legal values: 0 to 16.*	4
25	Reserved.		
26	DTMF_REV_TWIST	Controls the reverse twist level at which the CompactSPEECH 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
27	DTMF_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 CompactSPEECH 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_GENERATION_LEVEL + VOL_LEVEL = 6). 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
28	CID_RECEIVE_SENSITIVITY	If the Caller ID physical layer is Bell202 or V.23, CID_RECEIVE_SENSITIVITY defines the minimum energy level, in dBm, on the codec input, above which the Caller ID signal is detected. *Legal values: –26 to –38.*	–36
29	Reserved.		

Table 2-3: Tunable Parameters (Continued)

Index	Parameter Name	Description	Default
30	CID_START_END_CODES	If the Caller ID physical layer is DTMF: Defines the starting DTMF code (high nibble) and ending DTMF code (low nibble). The default value follows the Dutch Caller ID specification which defines the 'D' DTMF as the starting code and the 'C' DTMF as the ending code of the Caller ID data.	254
31–46	Reserved.		
47	CONST_NRG_TIME_COUNT	Minimum elapsed time until the CompactSPEECH processor reports constant energy level. Units: 10 ms. Accuracy: ±10 ms *Legal values: 1 to 65534*	700
48	CONST_NRG_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	CONST_NRG_LOW_THRESHOLD	Determines the minimum energy level that is treated as constant energy. The minimum energy is calculated as follows: $(1 - 1/2^{CONST\_NRG\_LOW\_THRESHOLD}) * average\_energy$ *Legal values: 1 to 16*	1
50	CONST_NRG_HIGH_ THRESHOLD	Determines the maximum energy level that is treated as constant energy. The maximum energy is calculated as follows: $(1 + 1/2^{CONST\_NRG\_HIGH\_THRESHOLD}) * average\_energy$ *Legal values: 0 to 16*	1
51	CID_PARAM4	2 bytes: US, French and Spanish Caller ID: minimum mark length. Recommended value for Spanish CID is 3.	0
52	CID_RECORD	1—Copy CID buffer to message memory during recording. 0—Do not copy CID buffer.	1
53	BUSY_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_MAX_ON_TIME	Maximum period considered as On for busy-tone detection. Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 10 to 1000.*	168

Table 2-3: Tunable Parameters (Continued)

Index	Parameter Name	Description	Default
55	BUSY_MIN_OFF_TIME	Minimum period considered as Off for busy-tone detection. Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 5 to 1000.*	7
56	BUSY_MAX_OFF_TIME	Maximum period considered as On for busy-tone detection. Unit: 10 ms. Accuracy is ±20 ms. *Legal values: 5 to 1000.*	122
57	BUSY_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_TYPE	Specifies the type of busy tone to detect: 1—Two cadences 2—Three cadences 3—Both two and three cadences	1
59	CADENCE_DELTA	The maximum allowed difference between two compared On or Off periods. Unit: 10 ms. *Legal values: 0 to 1000.*	9
60	DTMF_GAIN_LEVEL_AT_IDLE_MODE	SW AGC for DTMF in idle/record modes. When incrementing the tunable by 1, the dynamic range is increased by 3 dB.	0
61	DTMF_GAIN_LEVEL_AT_PLAY_MODE	Software AGC for play mode and tone generation modes. When incrementing the tunable by 1, the dynamic range increases by 3 dB.	

Example

TUNE 23 700					
Byte sequence:	Microcontroller	15	17	02	BC
	CompactSPEECH	15	17	02	BC
Description:	Set the minimum period for busy detection to seven seconds.				

VC Volume Control *vol_level*

Controls the energy level of all the output generators (playback, tone generation, and voice synthesis), 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 2-3.

For example, if the tunable variable VCD_LEVEL is 6, and *vol_level* is –2, then the output level equals VCD_LEVEL + *vol_level* = 4.

Example

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

WMSG Write Message *Data*

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

If the CompactSPEECH processor is in the IDLE state, the command opens the current message, switches the CompactSPEECH processor to the 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 0xFF (this has no affect on the current data in the page). A subsequent WMSG command adds a new block to the message, and writing continues at the new block.

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

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

If the current message is undefined, ERR_INVALID is reported.

Example

WMSG 32 bytes			
Byte sequence:	Microcontroller	31	32 bytes of data to write
	CompactSPEECH	31	echo 32 bytes of data
Description:	Create a message with tag = 01, and write 32 bytes in the message memory.		

Chapter 3—SCHEMATIC DIAGRAMS

The following schematic diagrams are extracted from a CompactSPEECH processor demo unit, based on the ISD-TDB266 board.

This demo includes three basic clusters:

- COP888EEG Microcontroller.
- CompactSPEECH processor cluster, including a TP3054 codec, and an ISD-T267SC controlling a Serial Flash device.
- User interface that includes one 16-digit LCD, and a 16-key (4 x 4) keypad.

For more details about the demo please refer to the *ISD Digital Answering Machine Demo Operating Instructions*.

NOTE *If IVS resides in Flash memory, and not in ROM, the address- and data-line connections are not required, and the layout is much simpler.*

Figure 3-1: CompactSPEECH Schematic Diagram

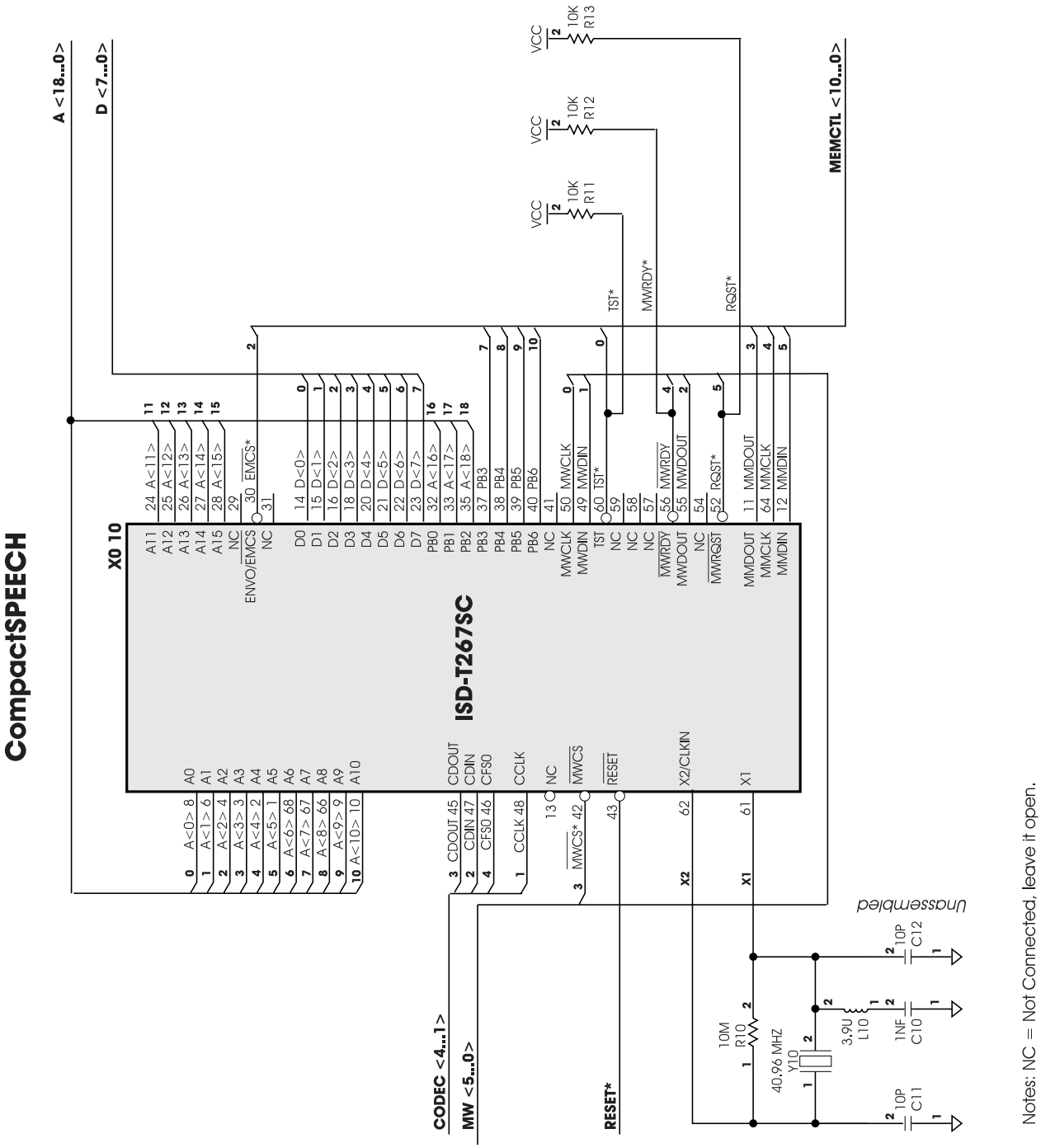


Figure 3-2: Flash Schematic Diagram

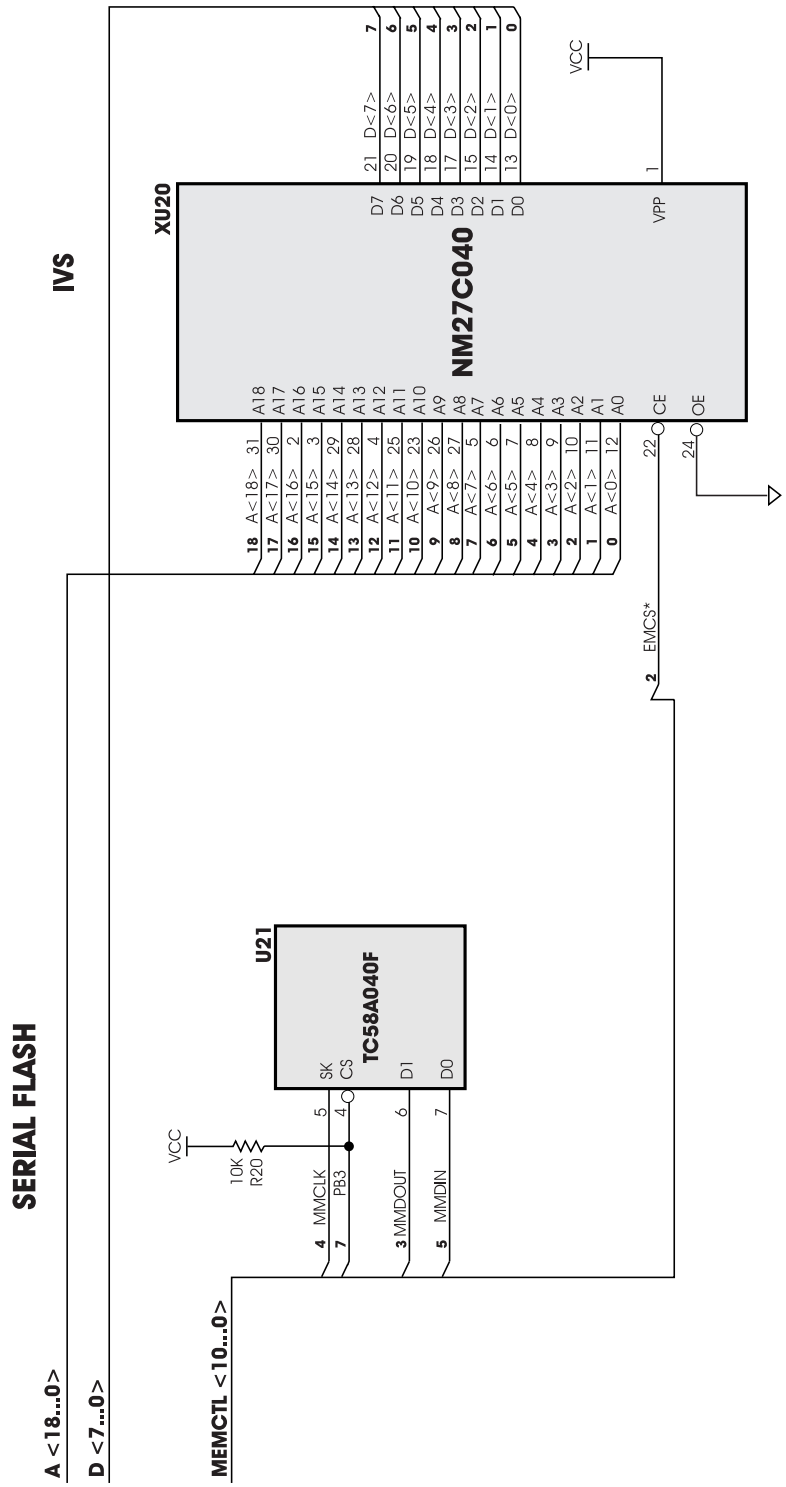


Figure 3-3: Codec Circuit Schematic Diagram

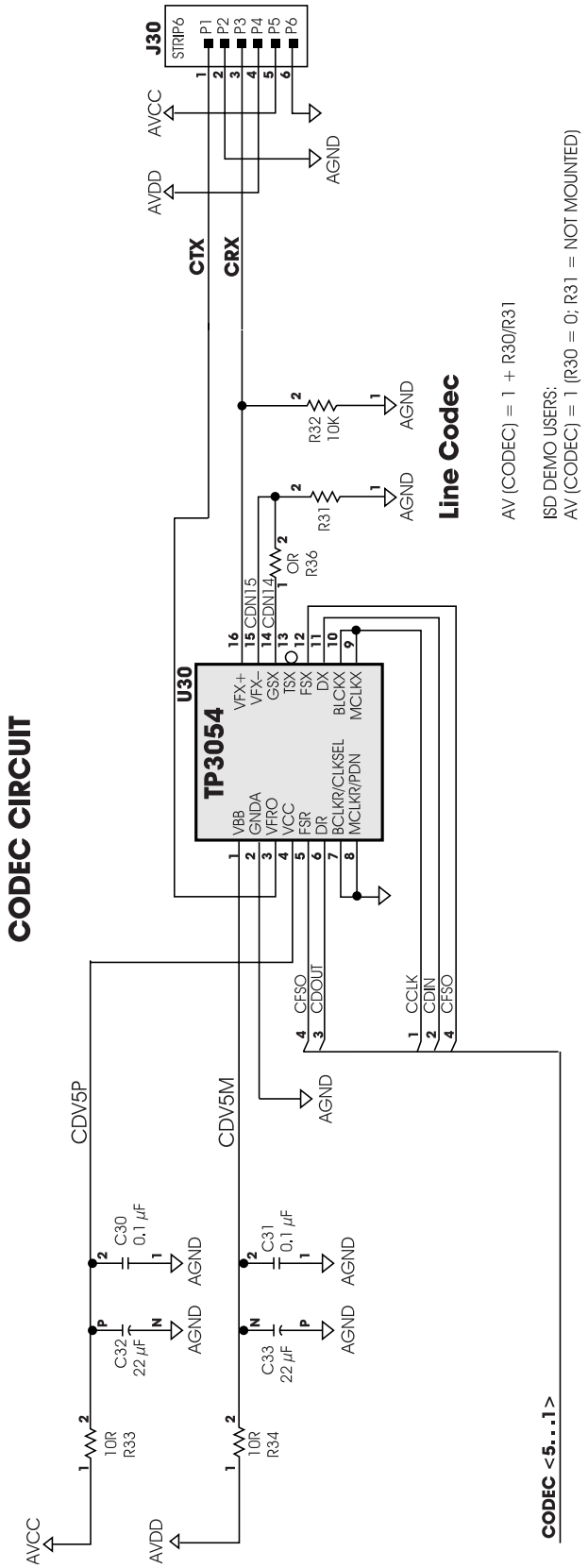


Figure 3-4: Microcontroller Schematic Diagram

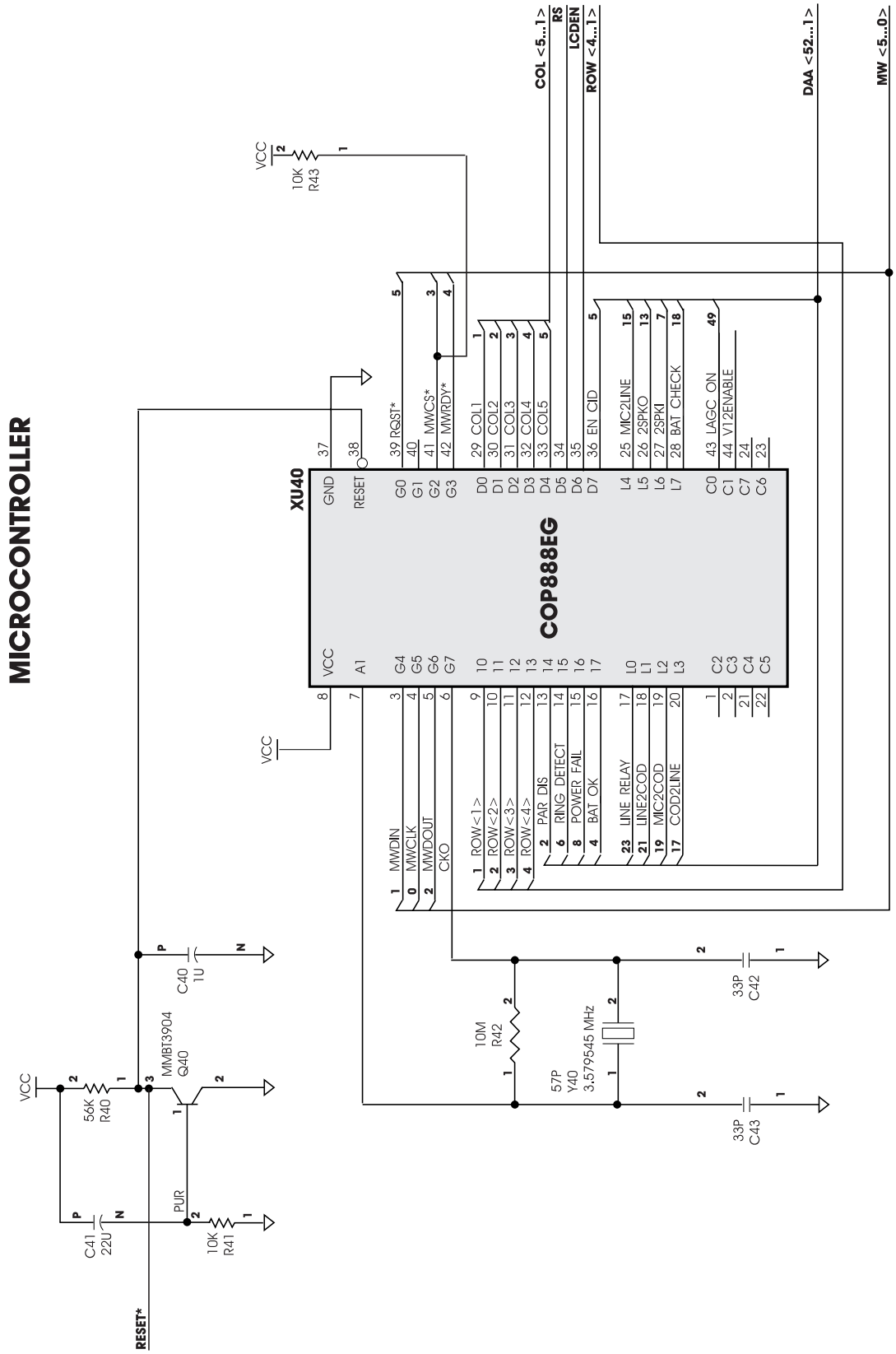
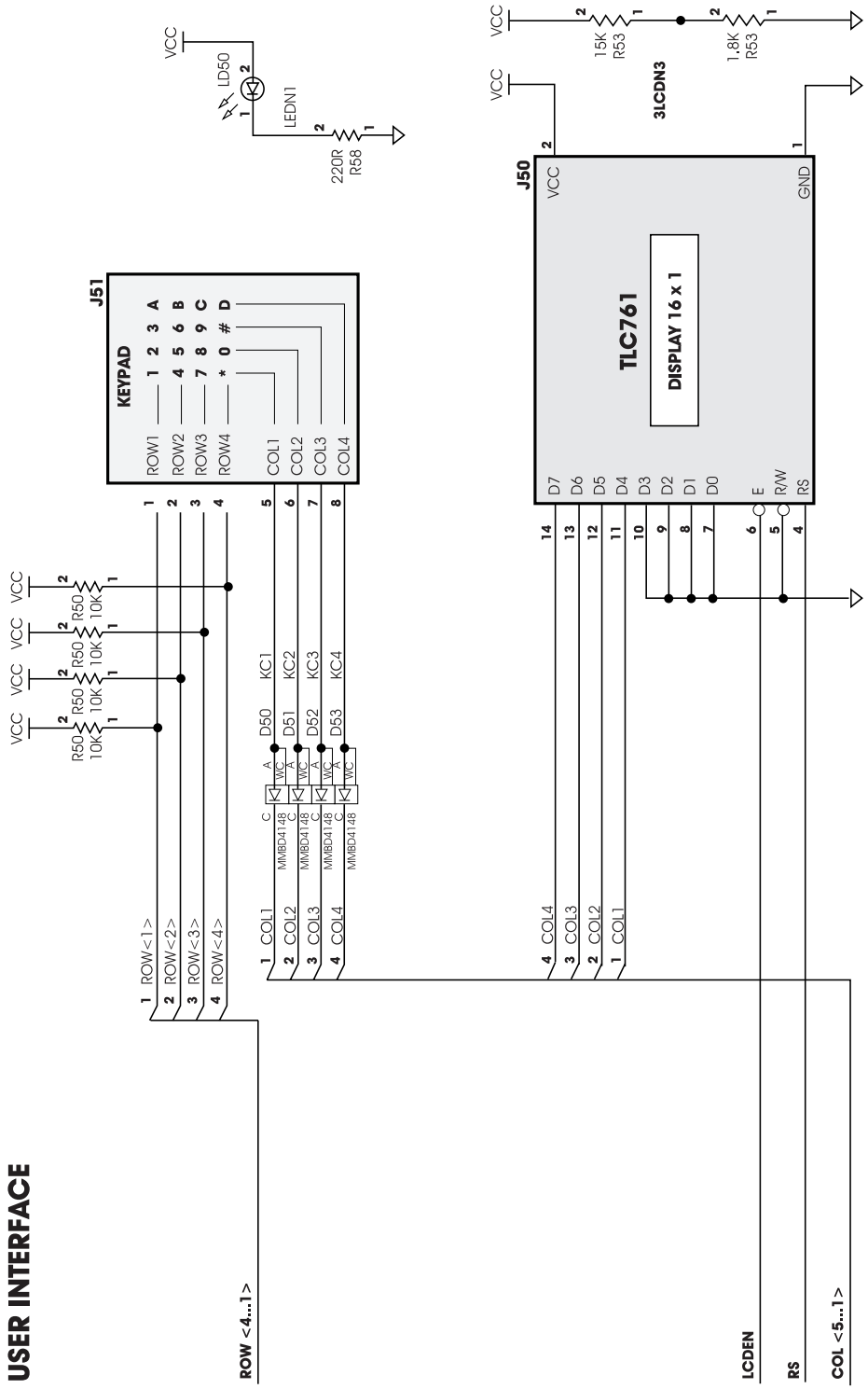


Figure 3-5: User Interface Schematic Diagram

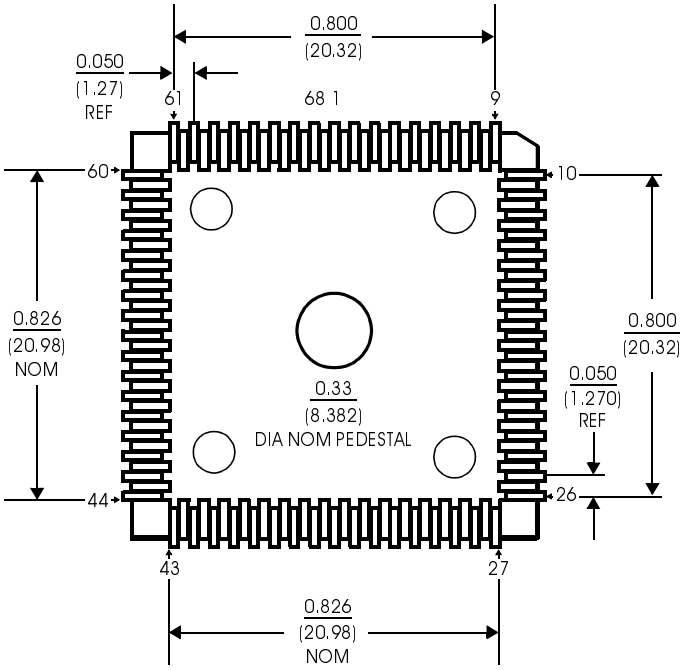


Chapter 4—PHYSICAL DIMENSIONS

Figure 4-1: 68-Pin Plastic Leaded Chip Carrier (J)—Order Number ISD-T267SC/J

BOTTOM VIEW

Inches (millimeters)



SIDE VIEW

Inches (millimeters)

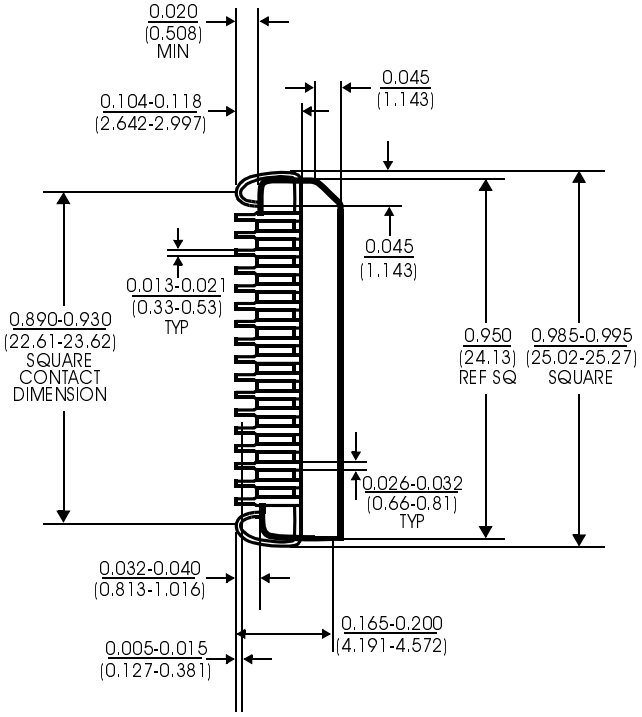
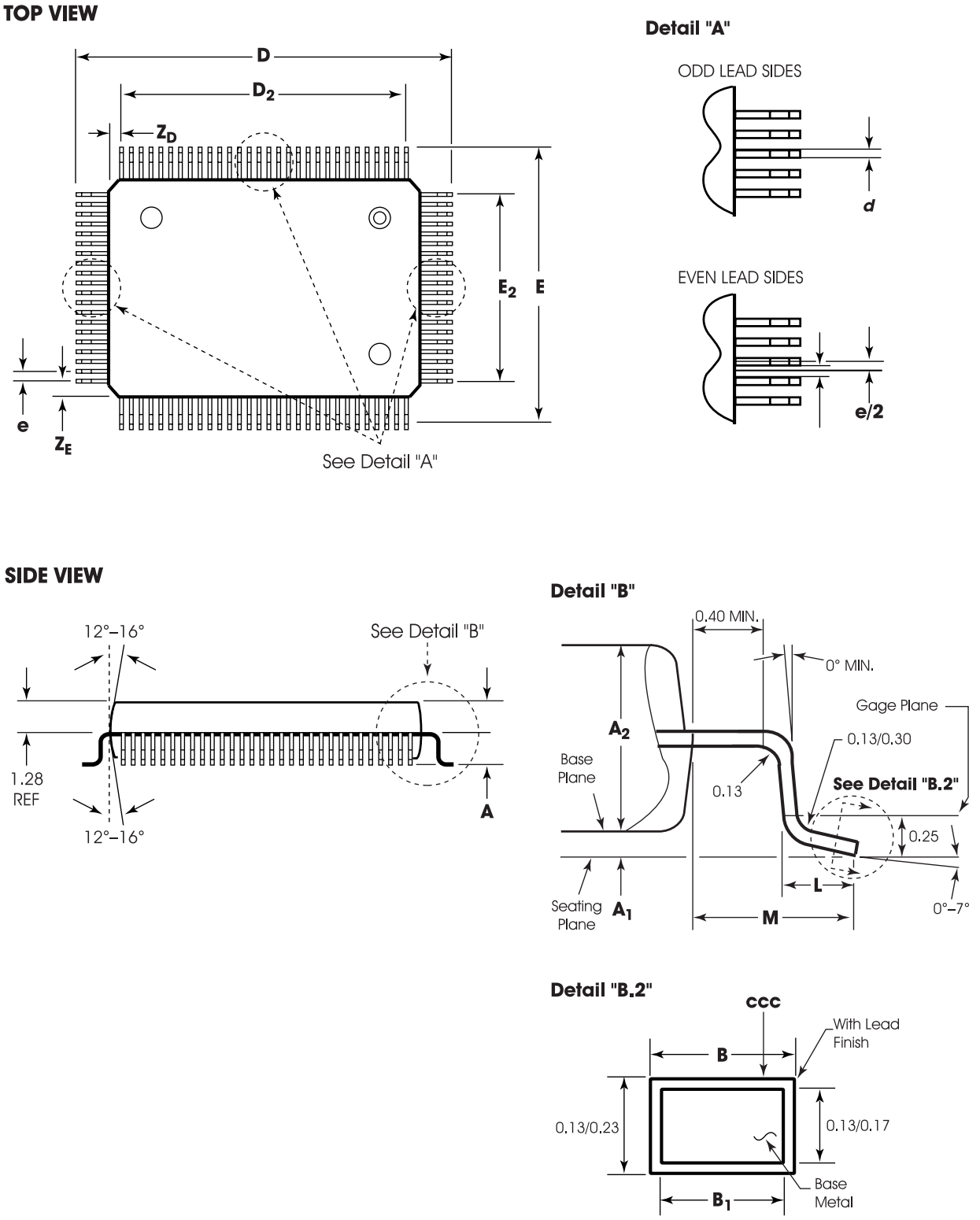


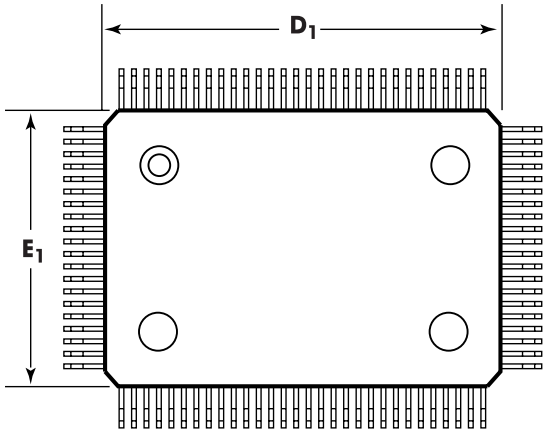
Figure 4-2: Package Outline Top and Side Views: MQFP. 14 x 20 Body, 1.60/0.33 mm Form, 2.71 mm Thick[1]



1. All dimensions are in millimeters. All dimensions and tolerances conform to ANSI Y14.5-1982.

Figure 4-3: Package Outline Bottom View: MQFP. 14 x 20 Body, 1.60/0.33 mm Form, 2.71 mm Thick[1]

BOTTOM VIEW



1. All dimensions are in millimeters. All dimensions and tolerances conform to ANSI Y14.5-1982.

Table 4-1: Package Outline, MQFP.—Order Number ISD-T267SC/Q

Symbol	Name	(All Dimensions in Millimeters)		
		MIN	NOM.	MAX.
A		—	3.04	3.40
A_1	Stand off	0.25	0.33	—
A_2	Body Thickness	2.57	2.71	2.87
D	Lead to lead length		23.20	
D_1	Body length		20.00	
D_2			18.85	
Z_D			0.58	
E	Lead to lead width		17.20	
E_1	Body width		14.00	
E_2			12.35	
Z_E			0.83	
L	Foot length	0.73	0.88	1.03
M	Lead form		1.60	
N	Number of pins		100	
f	Lead width		0.33	
e	Lead to lead pitch		0.65	
B		0.22	—	0.38
B_1	Lead width	0.22	0.30	0.33
ccc	Lead thickness	—	0.13	—
N_D	No. of pins, left side		30	
N_E	No. of pins, right side		20	

IMPORTANT NOTICES

The warranty for each product of ISD (Information Storage Devices, Inc.), is contained in a written warranty which governs sale and use of such product. Such warranty is contained in the printed terms and conditions under which the product is sold, or in a separate written warranty supplied with the product. Please refer to such written warranty with respect to its applicability to certain applications of such product.

These products may be subject to restrictions on use. Please contact ISD, for a list of the current additional restrictions on these products. By purchasing these products, the purchaser of these products agrees to comply with such use restrictions. Please contact ISD for clarification of any restrictions described herein.

ISD, reserves the right, without further notice, to change the ISD CompactSPEECH product specifications and/or information in this document and to improve reliability, functions and design.

ISD assumes no responsibility or liability for any use of the ISD CompactSPEECH products. ISD conveys no license or title, either expressed or implied, under any patent, copyright, or mask work right to the ISD CompactSPEECH products, and ISD makes no warranties or representations that the ISD CompactSPEECH products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Application examples and alternative uses of any integrated circuit contained in this publication are for illustration purposes only and ISD makes no representation or warranty that such applications shall be suitable for the use specified.

Information contained in this ISD CompactSPEECH data sheet supersedes all data for the ISD CompactSPEECH products published by ISD prior to January, 1998.

This data sheet and any future addendum to this data sheet is (are) the complete and controlling ISD CompactSPEECH product specifications. In the event any inconsistencies exist between the information in this and other product documentation, or in the event that other product documentation contains information in addition to the information in this, the information contained herein supersedes and governs such other information in its entirety.

2045 Hamilton Ave.
San Jose, California 95125
Tel: 408/369-2400
Fax: 408/369-2422
http://www.isd.com

Part No. 2201297D267SC