Telecommunications Interfacing to the TMS32010

APPLICATION REPORT: SPRA128

Author: Jeff Robillard
Semiconductor Systems Engineering

Digital Signal Processing Solutions
1989

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Telecommunications Interfacing to the TMS32010

Abstract

This report describes the implementation of a full-duplex 2.4-kbit/s LPC vocoder on a single TMS32010 device.

Product Support on the World Wide Web

Our World Wide Web site at www.ti.com contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.

INTRODUCTION

Signal processing has long been a tool used to improve performance in telecommunications systems. The advent of digital signal processing has given the telecommunications industry a powerful instrument to further enhance performance and reliability. Speech recognition, speech coding, speech synthesis, and high-speed modems are examples of applications now possible because of digital signal processing. Until recent years, real-time digital signal processing was solely the domain of minicomputers. The Texas Instruments TMS320 family offers low-cost VLSI implementations of digital signal processing functions. The TMS32010, the first-generation processor of the TMS320 family, is a high-speed, 16/32-bit digital signal processing microprocessor/microcomputer, ideally suited for telecommunications applications.

The purpose of this application report is to facilitate TMS32010 design by showing various telecommunications applications. The interfaces described in this report can be used in the following applications:

1. Digital modem
2. Echo cancellation
3. Analog repeaters
4. Handsfree telephone (speakerphone)
5. Noise reduction
6. Digital speech interpolation (DSI)
7. Analog-switched network simulator
8. Voice and data encryption
9. Simultaneous voice and data transmission
10. Speech coding and decoding, such as Linear Predictive Coding (LPC)
11. ADPCM transcoding
12. Voice mail
13. DTMF encoding and decoding
14. System identification.

This application report consists of two major sections. The first section details various hardware building blocks required for system interfacing. This provides a simple means of constructing a TMS32010-based system for many telecommunications applications. Included are circuit diagrams and their functional descriptions. These standard telecommunications interfaces provide a realistic medium for implementing digital signal processing algorithms based on the TMS32010. Standard serial and parallel interfaces permit connection to most digital Private Branch Exchange (PBX) backplanes and Time Division Multiplex (TDM) systems. Standard combo-codec filters provide analog interfaces that conform to CCITT recommendations. Description of hardware PCM companding for use with combo-codec filters is useful with time-critical algorithms. Microphone, loudspeaker, and host system interfaces are also included.

The second section of the report examines the following TMS32010 applications based on the building blocks described in the first section:

1. Standalone analog interface
2. Telephony test-set interface
3. PBX backplane interface
4. 2/4-wire transformer interface
5. Three-way conference interface
6. ADPCM interface
7. DTMF detection interface.

HARDWARE INTERFACE BLOCKS AND CIRCUITRY

The hardware circuit described in this section is divided into distinct modules or blocks. This modular approach allows flexibility in accommodating a wide range of telecommunications applications. These blocks can be placed together to fit a specific application. The following hardware blocks are detailed in the next sections of this application report:

1. TMS32010 and support circuit
2. Timing and control circuit
3. PCM linearization circuit
4. PBX/TDM interface circuit
5. Analog interface circuit
6. Host interface circuit.

The system block diagram, shown in Figure 1, illustrates how these blocks fit together around the TMS32010, the focal point of all activities. All blocks are directly connected to the TMS32010 data bus. The TMS32010 accesses all interfaces, except the program memory, using its I/O ports. Each interface is assigned to an I/O port. Port decode logic is used to decode the port address from the TMS32010 I/O space.

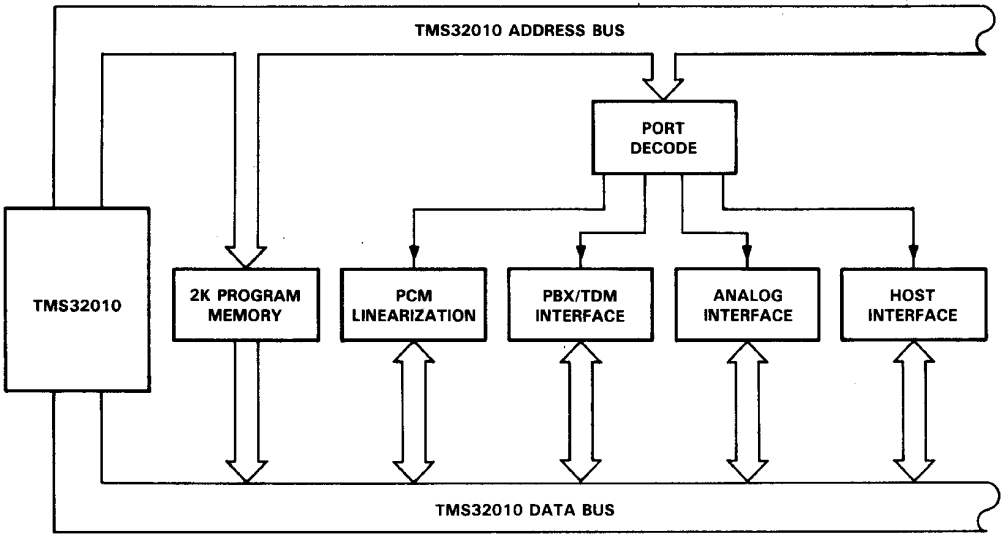


Figure 1. System Block Diagram

TMS32010 and Support Circuit

The TMS32010 serves as the hardware engine that performs the DSP algorithms. It interfaces to its peripherals via its I/O ports. Support circuitry for the TMS32010 includes memory and port decode logic and synchronization flip-flops. Table 1 indicates the TMS32010 port assignments for the application hardware constructed for this application report.

Table 1. TMS32010 Port Assignments

PORT	ASSIGNMENT
0	Analog interface 1
1	Analog interface 2, Microphone and loudspeaker interface
2	PBX/TDM interface 1
3	PBX/TDM interface 2
4	μ/A-law to two's-complement linear conversion
5	Two's-complement linear to μ/A-law conversion*
6	Host interface
7	Not used

*To perform conversions, a write to port 4 is required. The converted value is read at port 5.

The TMS32010 can be reset by momentarily activating the reset switch, which provides a 6-ms pulse to the $\overline{RS}$ input to the TMS32010. The device has one hardware interrupt ($\overline{INT}$) and one software interrupt ($\overline{BIO}$). The $\overline{BIO}$ (branch on I/O) input to the TMS32010 is considered the low-priority interrupt and is used for synchronizing to the 8-kHz framing pulse that drives the telecommunications interfaces (i.e., combo-codec filters and PBX/TDM interfaces). The $\overline{INT}$ input is considered the high-priority interrupt and is used for communicating with the host processor via the host interface (see Table 2 for interrupt assignments). The $\overline{INT}$ input becomes active when a command from the host is waiting to be processed. The reset and interrupt signals going to the TMS32010 are all synchronized.

Table 2. TMS32010 Interrupt Assignments

INTERRUPT	ASSIGNMENT
$\overline{RS}$	Reset. Vector to location >000H in program memory. Initialization of all pointers and internal registers recommended.
$\overline{INT}$	Hardware interrupt. Vector to location >002H in program memory. Interrupt service routine for performing host command processing.
$\overline{BIO}$	Software interrupt. Program in wait state when using the BIOZ instruction, waiting for 8-kHz synchronization signal.

Figure 2 shows the TMS32010 and support circuitry. The TMS32010 (U1) has a 16-bit data bus that interfaces to all ports and program memory. All port decoding is accomplished with two SN74ALS138 decoders (U6 and U7), which provide the write and read port strobes, respectively.

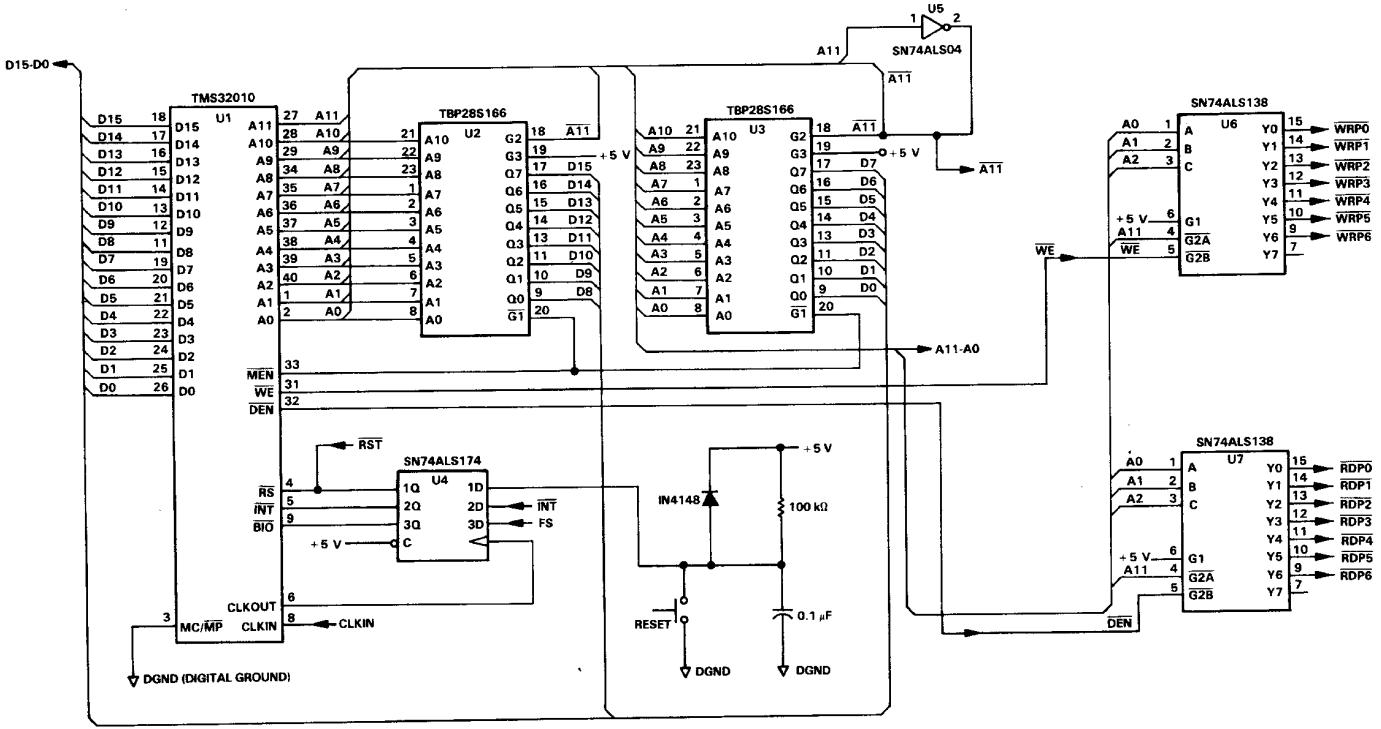


Figure 2. TMS32010 and Support Circuit

The reset and interrupts are synchronized to the TMS32010 with an SN74ALS174 D-type flip-flop (U4). The TMS32010 12-bit address bus is used for addressing program memory and the individual ports. This address bus drives the TBP28S166 program memory PROMs (U2 and U3) and the port decode logic (U6 and U7). The PROMs are mapped into the lower 2K address space (>000H to >7FFH). Address bit A11 is used for decoding the lower and upper 2K program memory spaces.

Timing and Control Circuit

The timing and control circuit supplies all the required clocking, timing, and control signals to operate the TMS32010 and interfaces. A 20.48-MHz clock is provided for the TMS32010. Derived from this clock are all the timing signals to operate the combo-codec filters, such as the 128-kHz clock for shifting data and the 8-kHz framing pulse for synchronization purposes (see Figure 3). External frame synchronization is permitted with an 8-kHz framing pulse input. Note that the internal clock always operates asynchronously with the external system clock.

All timing and control signals are derived from a 20.48-MHz clock source (see Figure 4). The inverters (U12) are configured as an oscillator with a 20.48-MHz crystal and 10-kohm resistor. A decode counter (U8) divides the 20.48-MHz clock by 10 to produce a 2.048-MHz clock signal required by the combo-codec filters. A dual 4-bit binary counter (U9) divides the 2.048-MHz clock by 16 to produce a 128-kHz clock signal used by the combo-codec filters and related support circuitry to shift in/out serial data. U9 also divides the 128-kHz clock by 16 to produce an 8-kHz framing strobe (FS) used in the serial/parallel data conversion process and as a synchronization signal to the TMS32010. U10 and U11 are used to synchronize counters U8 and U9 to an external asynchronous 8-kHz framing pulse (FP). This permits synchronization to an external system, such as a PBX, on a frame-by-frame basis.

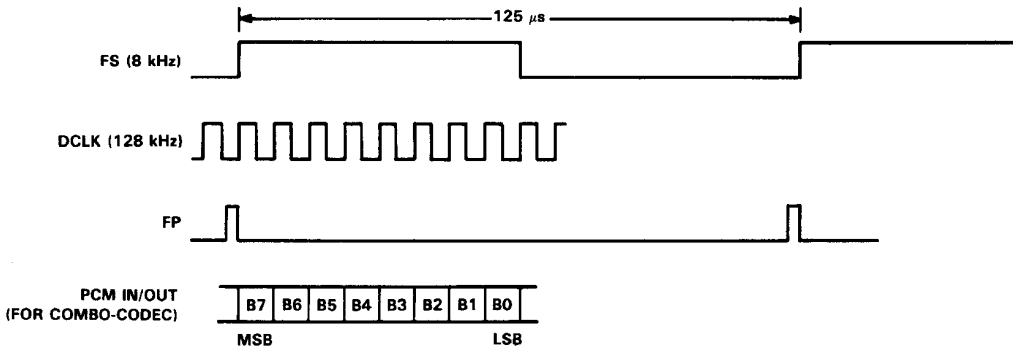


Figure 3. Clocking and Timing Diagram

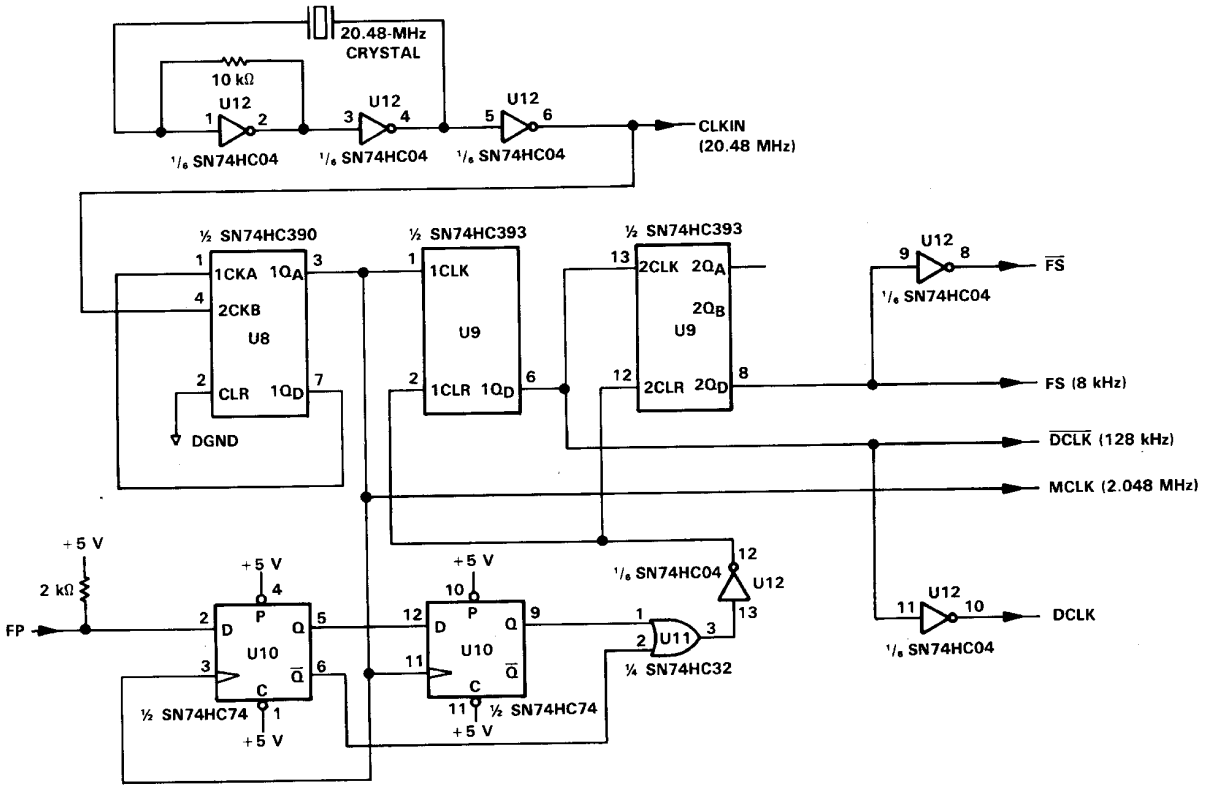


Figure 4. Timing and Control Circuit

PCM Linearization Circuit

In digital telecommunications systems, speech is transmitted in an 8-bit nonlinear PCM code. μ-law and A-law are the two most popular PCM coding schemes. For linear digital signal processing, the PCM code must be converted to two's-complement linear codes for processing by the TMS32010. After processing, the two's-complement linear code must be converted to the nonlinear PCM code. This conversion process can be performed in software on the TMS32010. A TMS32010 source code for the conversion process is provided in the application report, "Companding Routines for the TMS32010."[1] In certain cases, there may not be enough time in an 8-kHz framing period to accomplish this conversion process.

For time-critical applications, hardware linearization is used to provide a faster conversion process. The conversion process is accomplished by using table lookup ROMs. To convert from one code to another, a value is written to port 4, followed by a read from port 4 or 5. Port 4 is read to convert from a nonlinear code value to a two's-complement linear value. Port 5 is read to convert a two's-complement linear value to a nonlinear code value. The TMS32010 performs a port write followed by a port read in 800 ns when driven by a 20-MHz clock source.

The linearization circuit uses a lookup table to perform the conversion process (see Figure 5). U18 converts 14-bit two's-complement linear PCM to 8-bit μ/A-law PCM. U19 converts μ/A-law 8-bit PCM to 14-bit two's-complement linear PCM. Data to be converted is clocked into two SN74ALS574 octal D-type edge-triggered flip-flops (U13 and U14) when the TMS32010 writes to port 4. U13 and U14 contain the least- and most-significant bytes of data, respectively. μ/A-law data is buffered to the data bus via U16. Data is enabled onto the data bus when the TMS32010 reads port 5. Note that only the least-significant byte is active on the data bus. Linear data is buffered to the data bus via U15 and U17, which contain the most- and least-significant bytes of data, repectively. Data is enabled onto the data bus when the TMS32010 reads port 4. The most-significant byte of data is clocked into U15 on the rising edge of the $\overline{\text{MEN}}$ output from the TMS32010.

FORTRAN programs that compute μ-law PCM to 14-bit two's-complement linear PCM and vice versa are provided in the appendix. This computation generates the object file (in the INTEL data record format) for purposes of burning-in PROMs and EPROMs. A lookup ROM can also be programmed to generate patterns for test signal generator and DTMF encoder applications.

PBX/TDM Interface Circuit

The PBX/TDM interface permits connection to serial and parallel digital PBX backplanes and TDM systems. An example of a TDM system is a D1 channel bank that provides T1 carrier digital transmission on two-wire trunks between central offices. Each trunk provides 24 8-bit channels multiplexed in time. Each channel is intended to carry 8-bit companded PCM voice. The application hardware does not support any signalling functions in a T1 carrier connection.

Two PBX/TDM interfaces are provided in the application circuit to accommodate separate connections to receive (RX) and transmit (TX) paths. The hardware, as shown in Figure 6, is configured to interface onto an 8-bit parallel bus or serial PCM highway. The parallel bus interface provides three-state I/O, and the serial interface provides open-collector or three-state I/O. Selection between parallel or serial interface on the application circuit is hardware-selectable via a jumper. Each PBX/TDM interface occupies an I/O port on the TMS32010 I/O port map, as shown in Table 1. These interfaces are assigned to ports 2 and 3. The TMS32010 is synchronized to these ports via an 8-kHz framing pulse on its $\overline{\text{BIO}}$ input.

The two interfaces in this application circuit operate identically; therefore, only one interface, as shown in Figure 6, is described. The serial data input (SERIAL PCM IN) is converted to 8-bit parallel data via shift register U23. All timing and control signals are provided by the external system interface. SYSCLK is the external synchronous clock that drives U23. $\overline{\text{ITSEN}}$ (input time-slot enable) is synchronous with SYSCLK and clocks the 8-bit parallel data from U23 into U20 at an 8-kHz rate. Parallel data P7-P0 is clocked into U21 with the $\overline{\text{ITSEN}}$ signal at an 8-kHz rate. U20 and U21 are the storage registers for serial and parallel PCM data, repectively. Access to either of these registers by the TMS32010 is controlled on the application circuit via a jumper. The jumper routes the read port 2 signal from the TMS32010 to either U20 or U21, thereby selecting the serial or parallel data inputs. The PCM data output from the TMS32010 is clocked into U22 when a write to port 2 occurs. The parallel data in U22 is buffered to the system data bus via U26. U26 is an SN74LS244 line driver that provides high-current drivers to the external system PCM data bus. Data is enabled onto the external system PCM data bus when $\overline{\text{OTSEN}}$ (output time-slot enable) is active low.

For serial interfacing, the $\overline{\text{OTSEN}}$ signal loads the parallel data at U22 into shift register U24. Data is shifted to the PCM highway using the external system clock SYSCLK. The serial input pin (pin 10) is connected to the serial output pin (pin 9) on U24 to permit broadcasting the same PCM data on two or more consecutive time-slots on the PCM highway. $\overline{\text{OTSEN}}$ enables the data onto the PCM highway by controlling buffer U25. U25 drives either a three-state or open collector/drain PCM highway. U18 provides the logic which configures a three-state driver on U25 as an open-collector/drain driver.

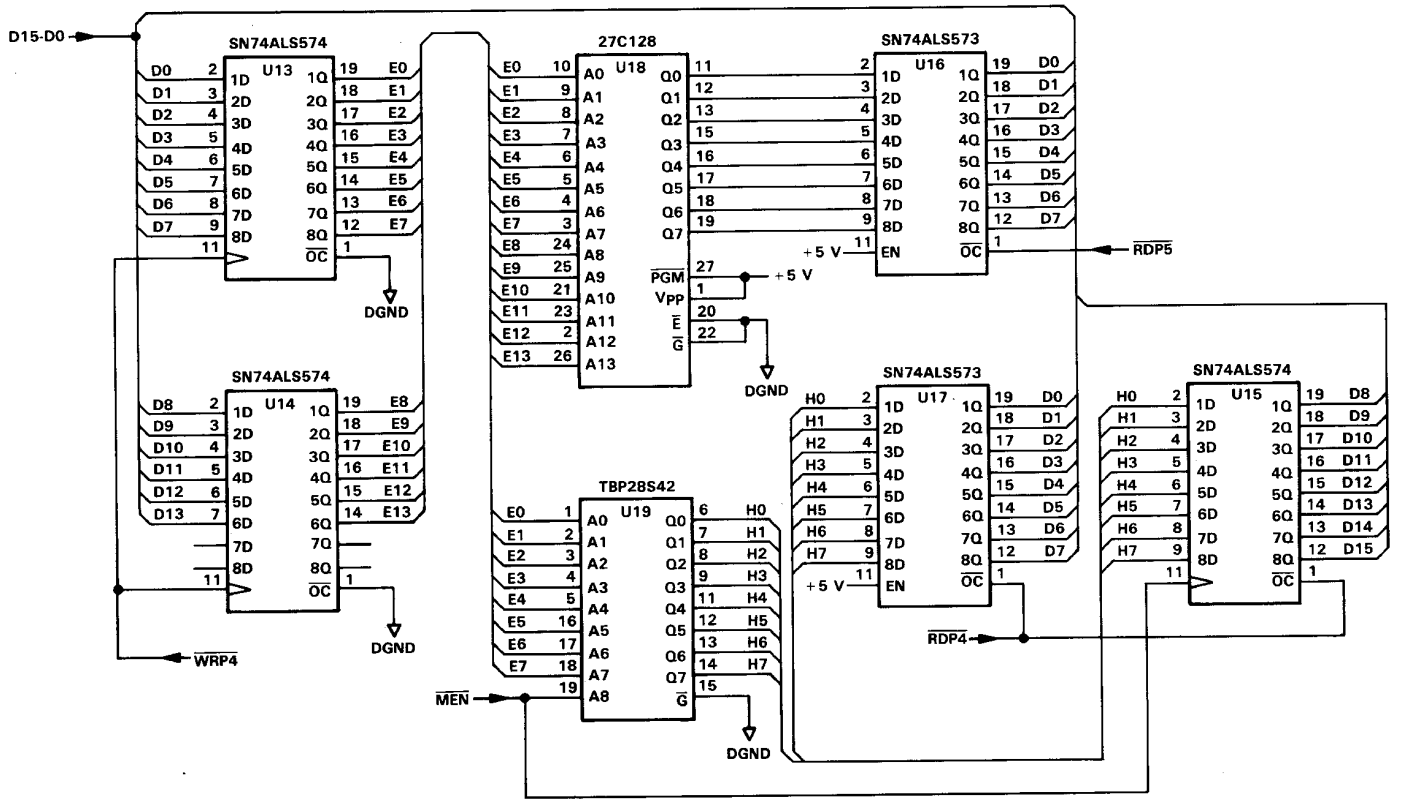


Figure 5. PCM Linearization Circuit

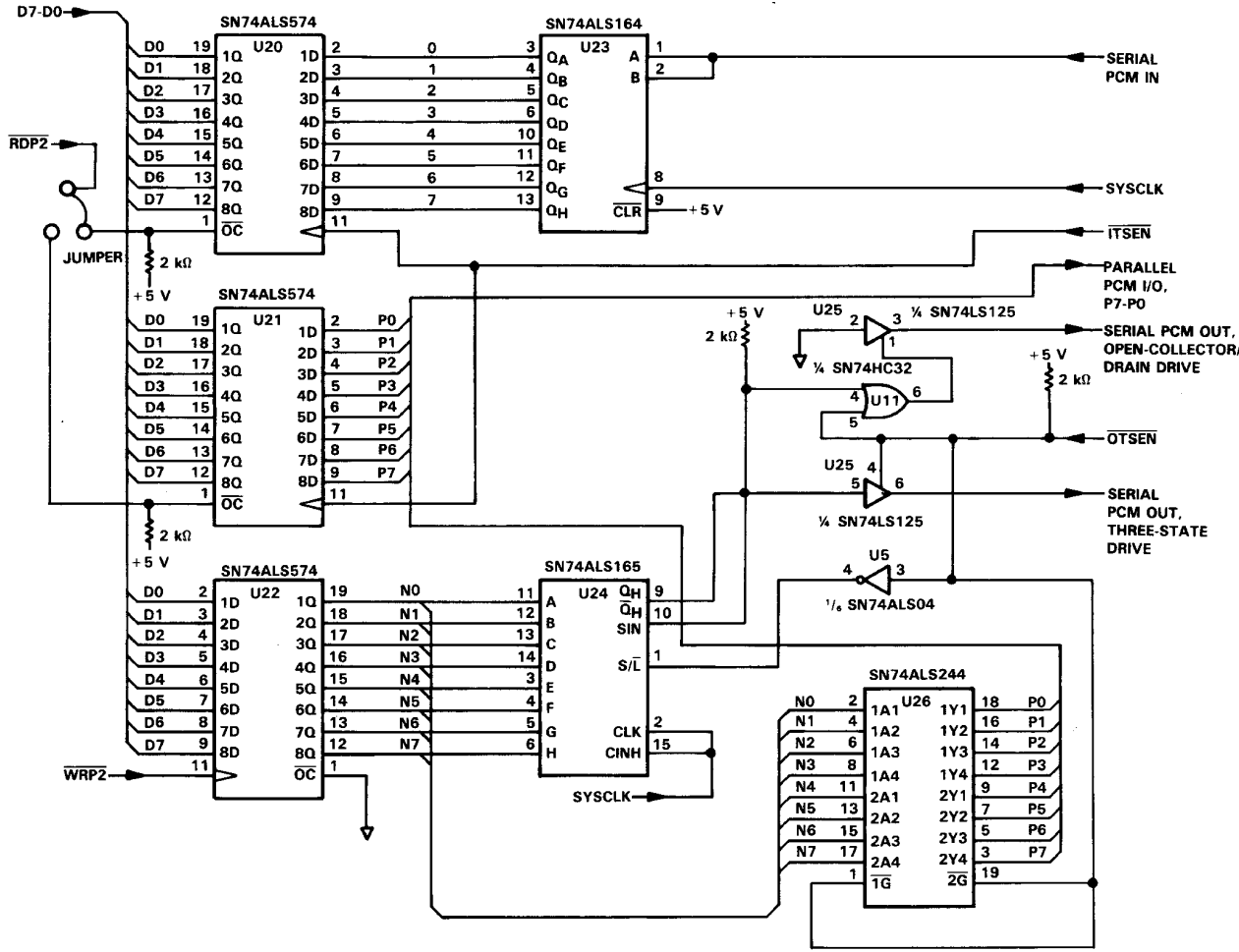


Figure 6. PBX/TDM Interface Circuit

Analog Interface Circuit

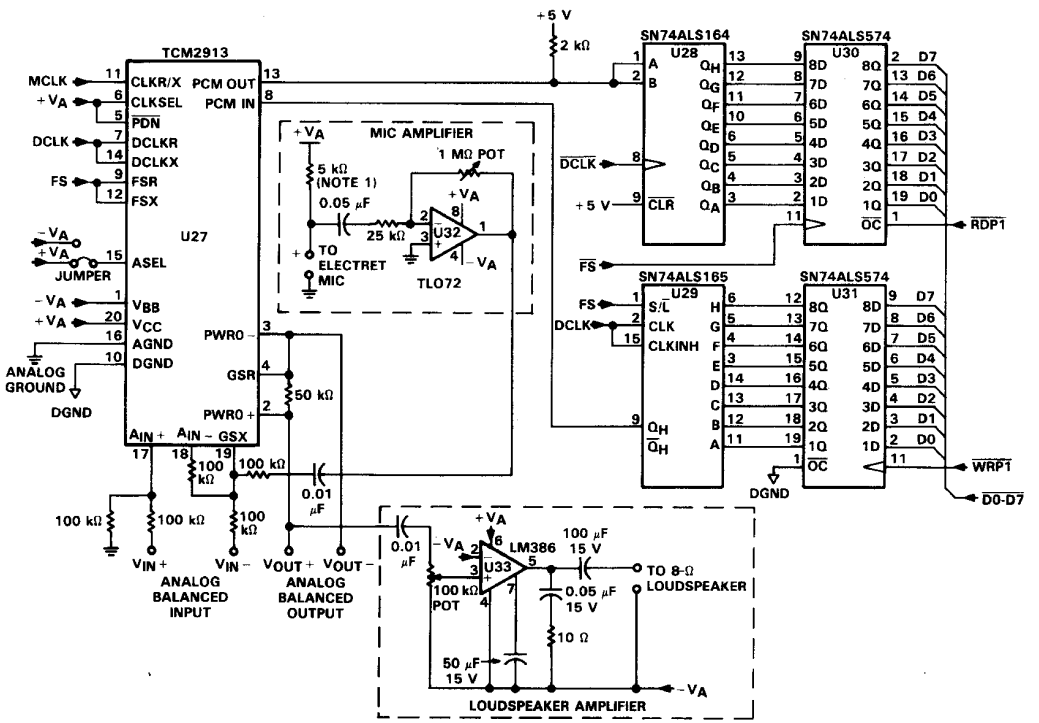
Two analog interfaces are provided to access receive (RX) and transmit (TX) analog paths. These interfaces conform to CCITT recommendations through the use of industry-standard combo-codec filters. μ/A-law operation can be hardware-selected via a jumper on the application circuit, as shown in Figure 7. One of the interfaces also contains microphone and loudspeaker amplifiers, which can be used in a handsfree speakerphone application. The gain of both amplifiers is adjustable using potentiometers. The overall gain through the TMS32010 and combo-codec filters is fixed at 3 dB. Each analog interface occupies an I/O port on the TMS32010 I/O port map (see Table 1). These interfaces are assigned to ports 0 and 1. The analog interface containing the microphone and loudspeaker amplifiers is at port 1. The application hardware sets the electret condenser microphone current at 1 mA. The loudspeaker amplifier can drive a standard 8-ohm loudspeaker.

The transmission signal reference points are governed by the combo-codec filters.[2] These reference points for unbalanced line interfacing are summarized as follows:

INPUT: 0 dBmO = 1.064-V RMS 1020-Hz sine wave between $V_{IN}+$ and analog ground; $V_{IN}+$ not connected.

OUTPUT: 0 dBmO = 1.503-V RMS 1020-Hz sine wave between $V_{OUT}+$ or $V_{OUT}-$ and ground.

Referring to Figure 7, the TCM2913 (U27) combo-codec filters[2] contain A/D and D/A converters and filters required for analog interfacing. A jumper is used to select either μ/law or A-law operation on the combo-codec filters by controlling the voltage level of ASEL (pin 15) on U27. A +5 voltage selects μ-law, and a −5 voltage selects A-law. Analog and digital grounds should be kept completely separate to avoid introducing digital switching noise in U27 and ground loops. The analog input to U27 is configured as a unity gain difference amplifier. U32 is the microphone amplifier that is configured as an inverting amplifier. The amplifier gain is controlled by a 1-Mohm potentiometer. The



NOTES: 1. Electret MIC current set at 1 mA.
2. Analog supply:
$+V_A = +5$ V
$-V_A = -5$ V

Figure 7. Analog Interface Circuit

amplifier input is designed to interface to an electret condenser microphone. The microphone current is set at 1 mA via a 5-kohm resistor connected to the analog +5 V supply. The DC voltage component is removed by capacitively coupling the amplifier. The 0.05-μF capacitor sets the highpass filter corner frequency at roughly 120 Hz. U32 is connected to U27, and the input gain at U27 is set at unity. The output transmission level at U27 is set at a 0-dB loss by strapping the PWR and GSR pins together.[2]

The loudspeaker amplifier is capacitively coupled to the U42 output, and fixed with a gain of 26 dB. However, a 100-kohm potentiometer attenuates the input, thereby acting as a volume control. The loudspeaker amplifier can drive an 8-ohm load. PCM data is clocked out of U27 at the PCM output pin (PCM OUT) at a rate of 128 kbit/s. U28 is a shift register that converts the serial data from U27 to 8-bit parallel data clocked into register U30 at an 8-kHz rate. The TMS32010 accesses register U30 when a read to port 1 occurs, which enables 8-bit data onto the least-significant byte of the 16-bit data bus. When data is being sent to the interface, the TMS32010 writes to port 1, which clocks 8-bit data into register U31. This data resides on the least-significant byte of the 16-bit data bus. Data at U31 is loaded into shift register U29 at an 8-kHz rate. This data is shifted out in serial format at a 128-kbit/s rate to the PCM input pin (PCM IN) of U27. The 128-kHz clock signal used for shifting serial data and the 8-kHz loading signal are provided by the system timing and control circuit as DCLK and FS,[2] respectively.

Host Interface Circuit

The TMS32010 can be externally controlled (if desired) by a host processor via the host interface. The host interface contains three registers: an acknowledgement register, ACKR (U34); a command register, COMR (U35); and a status register, SR (U36). Figure 8 shows the host interface

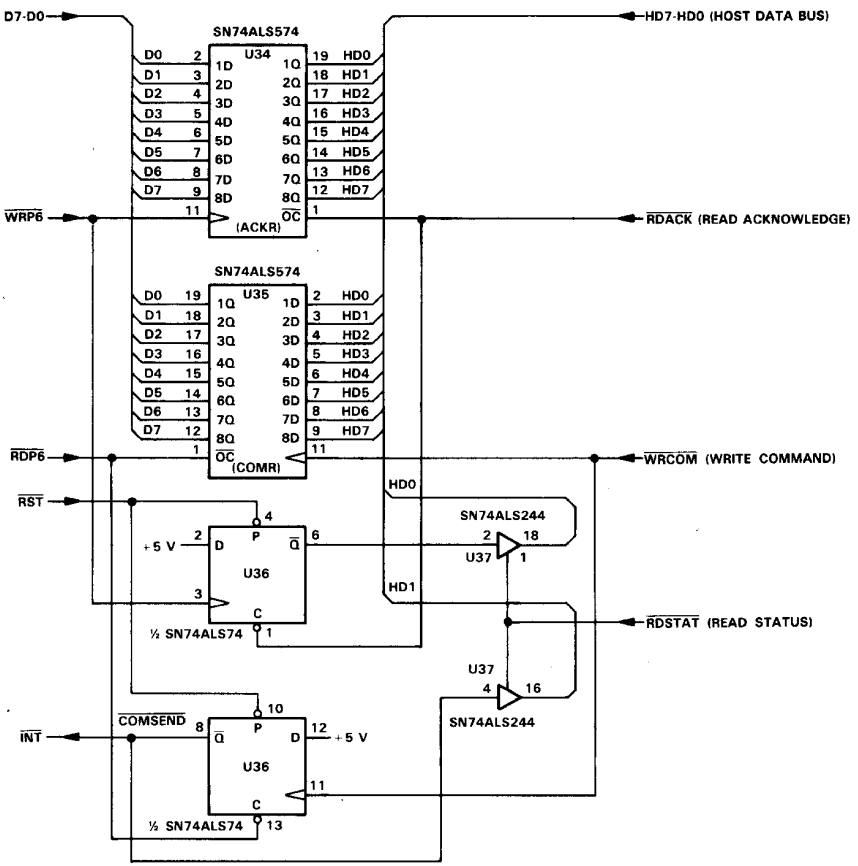


Figure 8. Host Interface Circuit

circuit. The host writes commands to the COMR, which can be read by the TMS32010. The commands are completely defined by the user. The TMS32010 writes to the ACKR, which can be read by the host. The COMR and ACKR are assigned to port 6 in the TMS32010 I/O map. The read and write strobes from the TMS32010 and the host control the bits in the SR that are used for handshaking purposes with the host. Only the host can read the contents of the status register. For PBX applications, the host is located within the PBX.

Handshaking signals are provided to control data transfers across the host interface. The host can access these signals by reading a two-bit SR, which resides on the two least-signficant bits of the host 8-bit data bus. The SR should reside in the host I/O port map or memory map. When reading the SR, the host must mask out the six most-significant bits since they are unknown. When a system reset occurs, the SR is set to 3H (xxxxxx11). When the host writes a command to the COMR, the SR is set to 1H (xxxxxx01), and the TMS32010 is interrupted. The TMS32010 then reads the command value in the COMR. This sets the SR to 3H.

When the TMS32010 completes processing the command, it responds to the host by writing an acknowledgement value to the ACKR, which sets the SR to 2H. The host reads the SR to determine if the TMS32010 has processed the command. If the command has been processed, the host reads the acknowledgement value in the ACKR. This sets the SR to 3H, which indicates that the command transaction has been satisfactorily completed by the TMS32010 and the host. The acknowledgement value can be dummy values or data being retrieved from the TMS32010. If the TMS32010 has not completed processing the command, the SR will read 1H or 3H. If the SR stays at 1H or 3H for an abnormally long period of time, a possible fault or problem is indicated. Table 3 shows the possible SR values and their interpretation.

Table 3. Host Interface Status Register Interpretation

HOST*	SR VALUE	COMSEND (bit 1)	COMACK (bit 0)	DESCRIPTION
A	0H	0	0	These conditions should not occur. Potential problems are indicated.
A	1H	0	1	
A	2H	1	0	
A	3H	1	1	Idle command processing state.
B	0H	0	0	Not allowed. Potential problem is indicated.
B	1H	0	1	The TMS32010 has not yet started processing the host command. Potential problem is indicated.
B	2H	1	0	The TMS32010 has completed processing the host command. Data is waiting for the host.
B	3H	1	1	The TMS32010 has started, but not completed, processing the host command.

* "HOST" refers to the command transaction state. "A" indicates that the command transaction is completed, or that a host command has not been sent. "B" indicates that a host command has been sent and that a command transaction is in progress.

The TMS32010 can process commands from the host in the following suggested manner: The command value written by the host, with an offset added to it, becomes the command vector or subroutine for the TMS32010. This is accomplished in software with a sequence of actions using the TMS32010. The interrupt service routine reads the host command value at port 6 (COMR) and stores it in internal data RAM (using the IN instruction). This value is then stored in the accumulator (using the LAC instruction). Using the AND instruction, the command value is masked to zero the 24 most-significant bits, and the new value is stored in memory. Masking is required because the eight most-significant bits on the TMS32010 data bus are unknown when reading the host interface port. A minimum offset of 4 is added to the masked command value in the accumulator (using the ADDS or ADD instruction). The offset is added to the command value to ensure that the resulting command vector does not overlap the reset and interrupt vector spaces. The next step is to vector to the subroutine via the CALA instruction. The CALA instruction uses the value in the accumulator as the subroutine vector. This forces the TMS32010 to begin executing a subroutine located at the command vector value in program memory space. Shown below is the source code sequence for the TMS32010.

```
ISR  IN    x,6  ; read host command in COMR
     LAC   x,0  ; load host command in ACC
     AND   y    ; mask host command
     ADD   z    ; add an offset to host command
     CALA       ; call subroutine
```

Note that ISR (interrupt service routine) is a label. During the TMS32010 initialization routine, the values >FFH and >4 are stored in locations y and x in data memory, respectively.

The TMS32010 communicates with the host on a byte data-transfer basis. The least-significant byte of the host and TMS32010 data bus is used. When the host issues a command, it writes one byte of data to COMR (U35), as shown in Figure 8. $\overline{\text{WRCOM}}$ (write command) is the write strobe from the host and is active low. Both flip-flops in U36 represent the SR. $\overline{\text{WRCOM}}$ sets a flip-flop in U36, thereby setting $\overline{\text{COMSEND}}$ (command send) low and interrupting the TMS32010. $\overline{\text{COMSEND}}$ is bit 1 in the SR. During the TMS32010's interrupt service routine, the TMS32010 reads host commands at port 6, which enables data from U35 onto the data bus. The port 6 read strobe sets $\overline{\text{COMSEND}}$ high, indicating that the command is being processed. When the TMS32010 responds to the command, it writes one byte into the ACKR U34 at port 6. This sets the second flip-flop in U36, thus forcing $\overline{\text{COMACK}}$ (command acknowledge) low. This indicates command processing is completed. $\overline{\text{COMACK}}$ is bit 0 in the SR. The host reads the ACKR (U34) by enabling data onto its bus using the $\overline{\text{RDACK}}$ (read acknowledge) strobe, which is active low and sets $\overline{\text{COMACK}}$ high. The host can read the SR (U36) with strobe $\overline{\text{RDSTAT}}$ (read status register), which is active low. U36 sets $\overline{\text{COMACK}}$ and $\overline{\text{COMSEND}}$ high when the TMS32010 is reset. A timing diagram illustrating the host transaction is shown in Figure 9.

INTERFACING APPLICATIONS

In this second section of the report, several telecommunications applications are discussed. First, an analog interface hardware application is examined, which is useful in self-contained standalone designs, such as analog repeaters or analog PBX designs. Then, a telephony test-set application is shown. This is useful as a self-contained standalone test tool or as a diagnostic function within a PBX. Finally, digital PBX applications in the following areas are described: PBX backplane interface, 2/4 wire transformer interface, three-way conference interface, ADPCM interface, and DTMF detection interface. It is important to note that the telecommunications designer can combine several applications using one TMS32010 to perform multiple functions.

These applications utilize the interface blocks previously described and the TMS32010 for performing digital signal processing algorithms. However, the scope of this report is not to discuss the DSP algorithms, but rather to demonstrate how these applications can be made possible using the telecommunications interface circuits. Application reports are available, which implement digital signal processing algorithms for telecommunications applications using the TMS320 family.[3,4,5]

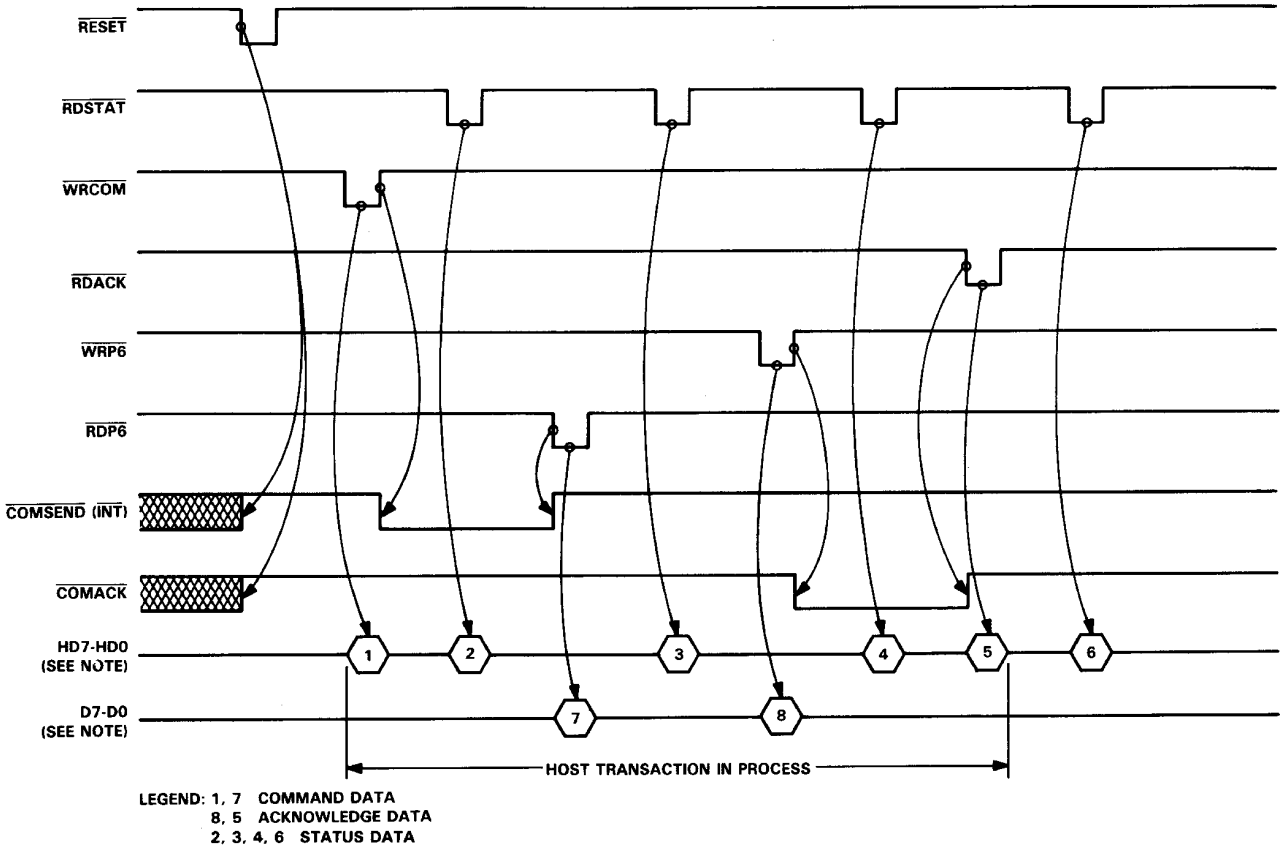
Standalone Analog Interface

The standalone analog interface is useful where remote or independent operation is required. Some examples of standalone applications are adaptive repeaters and handsfree telephones (speakerphones), as shown in Figure 10.

The standalone analog interface contains elements from the following hardware blocks previously described:

1. TMS32010 and support circuit,
2. Timing and control circuit, and
3. Analog interface circuit.

Figure 10 illustrates the hardware configuration for a handsfree telephone application. The TMS32010 performs adaptive filtering and voice-switching functions. The analog interface circuit provides two combo-codec filters to interface to the analog telephone line, loudspeaker, and microphone. The combo-codec filters perform filtering of the analog I/O, and A/D and D/A conversion. The serial data streams to and from the combo-codec filters are converted to parallel format that permits interfacing to the TMS32010's parallel I/O. The TMS32010 easily interfaces to combo-codec filters using 8-bit shift registers and octal D-type flip-flop registers. Both combo-codec filters share one set of shift registers at the serial interface. The timing and control circuit provides all the signals required to synchronize and control the combo-codec filters, registers, and TMS32010.



NOTE: Only host transaction data bursts shown.

Figure 9. Host Transaction Timing Diagram

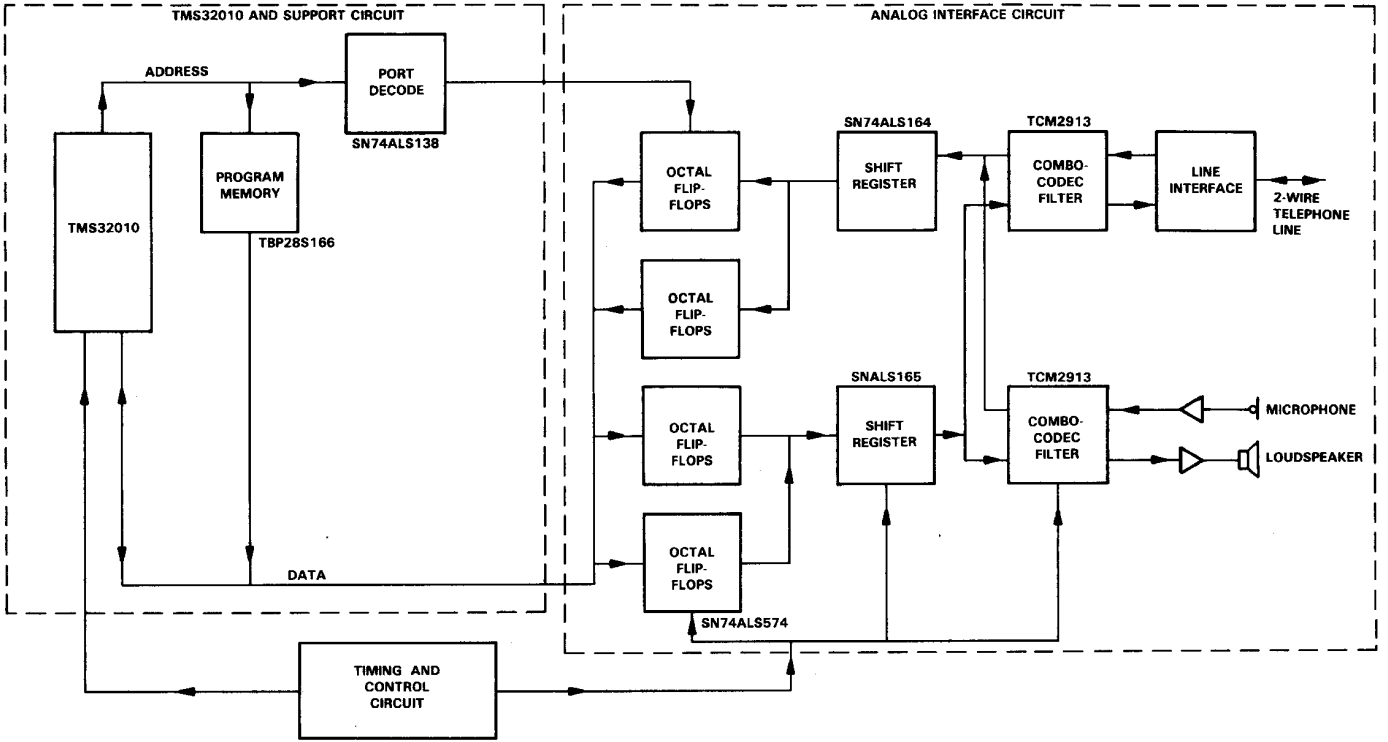


Figure 10. Handsfree Telephone System Diagram

Telephony Test-Set Interface

A telephony test-set is a useful tool for testing analog interfaces in telecommunications. It can also be used in a PBX design where analog line testing diagnostics are required. The application shown in Figure 11 is a standalone system.

A test-set application uses the following hardware blocks:

1. TMS32010 and support circuit,
2. Timing and control circuit,
3. Analog interface circuit,
4. Host interface circuit, and
5. PCM linearization circuit (optional).

Balanced inputs and outputs are required in a test-set application. The TMS32010 can be programmed to measure noise, distortion, signal levels, etc., and serve as a network simulator for analog telephony circuits. The electrical performance of the combo-codec filters is a limiting factor on measurements and simulations that can be performed. Figure 11 shows a possible telephony test-set system configuration. All measurements are under control of a host processor. The host interfaces to a control input, such as a keyboard or human interface I/O, and a display, such as a CRT. Measurement parameters can be modified via the control interface, and the actual measured values displayed. All transfers between the host and application circuit are performed via the host interface.

A possible test-set function may be a sine-wave signal generator. The TMS32010 produces a digital sine wave, which is converted to the analog domain via the combo-codec filters in the analog interface circuit. TMS32010 source code for a precision sine-wave generator is given in the application report, "Precision Digital Sine-Wave Generation with the TMS32010."[6] Note that this algorithm produces linear samples that must be converted to μ-law or A-law PCM samples. TMS32010 source code for performing this conversion can be found in the application report, "Companding Routines for the TMS32010."[1] Hardware conversion is also possible using the PCM linearization circuit, described in the previous major section of this report.

PBX Backplane Interface

A backplane is an interconnect system with its mechanical, electrical, pin assignments, and interaction protocol rigidly defined. Mechanically, a backplane is a printed circuit board with connectors that reside at the rear of a card-cage, providing electrical interconnection between cards plugged into these connectors. Architecturally, a backplane has a set of protocols that control data transfers between interconnected cards. This application report concerns itself only with some architectural aspects of possible PBX backplanes.

Two types of backplanes are used in most PBX designs: serial and parallel. The serial backplane is the most popular. Interfacing with standard codecs is simple, because they contain serial input and output ports. In some modern PBXs, a parallel backplane is used, which eases hardware interfacing since standard byte-wide logic can be used. In either system,

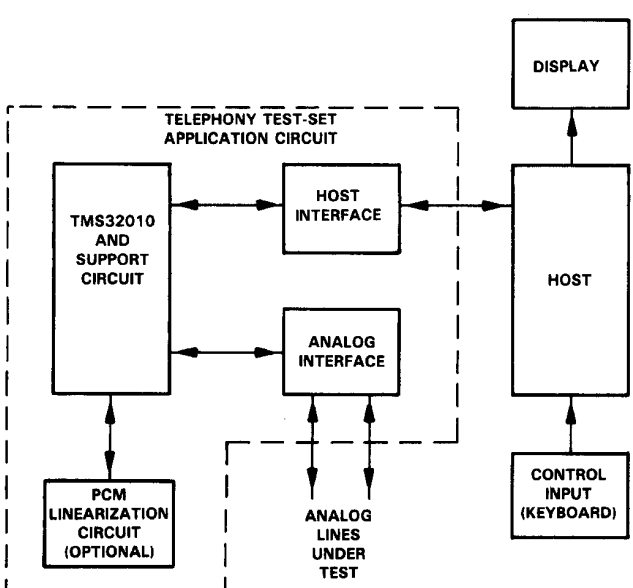


Figure 11. Telephony Test-Set System Diagram

this application circuit can serve as a signal processing module to perform specialized functions, such as DTMF detection, echo cancellation, noise reduction, and conferencing. The host processor determines which PCM channels are assigned to the signal processing module.

A typical PBX backplane contains one or more serial PCM highways or an 8/16-bit PCM bus, system address and data buses, and system control lines (see Figure 12). The system address bus is used by the host processor to access all the peripherals or modules connected to the backplane. The data transfers occur across the data bus between the host processor and modules. The system control lines are typically read and write strobes used by the host processor to control these data transfers. The system clock and 8-kHz framing pulse are used for synchronizing access to the PCM highway or bus. In most systems, time-slot enable control lines are provided to access individual PCM channels. The time-slot enable control lines are provided by either the backplane with a control card using time-slot decode circuitry or card-resident time-slot decode circuitry. The application circuit does not provide any time-slot decoding; however, it provides the flexibility to interface to most systems through the external signals $\overline{\text{ITSEN}}$ and $\overline{\text{OTSEN}}$.

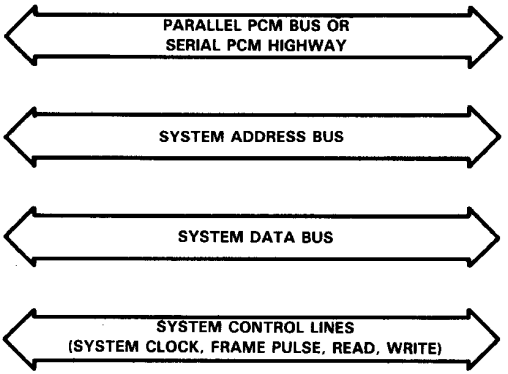


Figure 12. Typical Backplane Configuration

In Figure 13, the framing pulse is used to indicate the start of a frame. The framing pulse, usually one system clock cycle wide, is synchronous with the system clock and synchronizes all peripherals connected to the PCM highway or bus. Many PCM systems are based on a 2.048-MHz system clock, which provides 32 channels on a serial PCM highway or 256 channels on an 8-bit parallel PCM bus.

The backplane application hardware requires the following blocks:

1. TMS32010 and support circuit,
2. Timing and control circuit,
3. Two PBX/TDM interface circuits for use with receive and transmit channels,
4. Host interface circuit, and
5. PCM linearization circuit for critical real-time applications (optional).

Each PBX/TDM interface contains a serial input port, serial open-drain/collector, three-state output ports, and an 8-bit wide three-state parallel output port. These ports directly connect to the PCM bus or highway on the backplane. An external device or circuit is required to provide the time-slot enables $\overline{\text{ITSEN}}$ and $\overline{\text{OTSEN}}$ (see Figure 14). The host determines which channels reside in time-slots on the PCM backplane, and writes information to a time-slot controller residing in the peripheral or module connected to the backplane. The controller then furnishes the input and output time-slot enables $\overline{\text{ITSEN}}$ and $\overline{\text{OTSEN}}$, respectively. $\overline{\text{ITSEN}}$ assigns to the application circuit the PCM data in the corresponding time-slot on the PCM backplane. $\overline{\text{OTSEN}}$ enables PCM data from the application circuit onto the appropriate time-slot on the PCM backplane. The application hardware permits broadcasting PCM data onto several time-slots on the PCM backplane.

2/4-Wire Transformer Interface

The 2/4-wire transformer is useful for interfacing directly to 2-wire trunks or telephone lines. The transformer application requires the following hardware blocks:

1. TMS32010 and support circuit, and
2. Analog interface circuit.

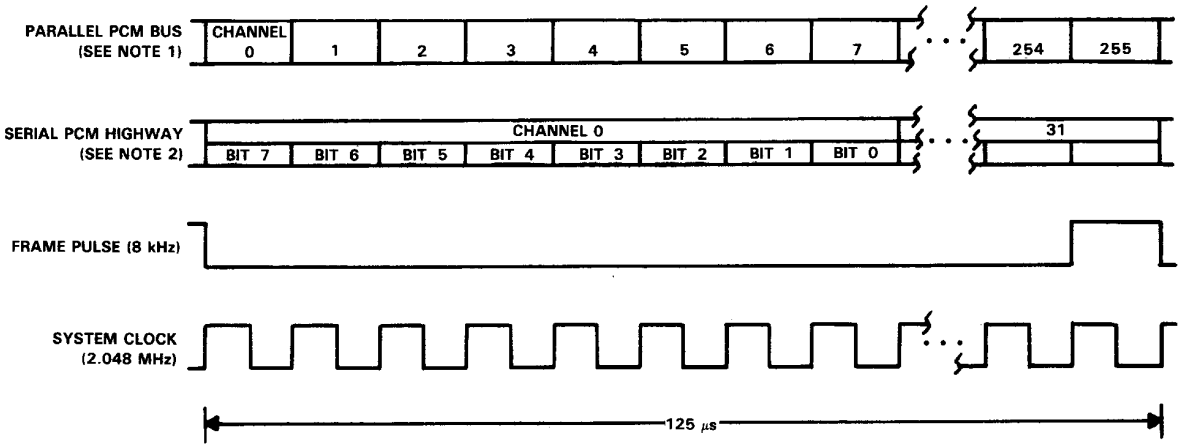
Figure 15 shows how a 2/4-wire transformer is connected to the analog interface circuit. $V_{OUT}+$ drives a load resistor (RL) in series with the transformer. This series arrangement produces a 6-dB loss between $V_{OUT}+$ and the telephone line. The load resistor balances the transmission line. The input is arranged so that the output signal reflected back at the transformer to input $V_{IN}+$ is minimized. This is accomplished by subtracting $V_{OUT}+$ using the $V_{IN}-$ input in a difference amplifier configuration. The return loss will be a function of matching RL to the telephone-line impedance. A mismatch between the load RL and the telephone line will worsen the return loss. The transmission-signal reference points were described in the analog interface circuit section. Information on 2/4-wire analog interfacing that conforms to FCC Rules Part 68 can be found in reference [7].

Three-Way Conference Interface

A popular PBX feature is three-way conferencing. In analog PBXs, conferencing is easily accomplished with op-amps in a summing configuration. The problem is not trivial in digital PBXs. μ/A-law encoded PCM samples cannot be directly summed together with standard adders (i.e., SN74LS283s) because of the nonlinear encoding scheme used. These PCM values must first be converted to linear values, summed, then converted back to μ/A-law PCM. This function is possible with the TMS32010. As mentioned previously, the PCM samples can be linearized in software[1] or in hardware.

For a three-way conference, the application hardware (see Figure 16) contains the following modules:

1. TMS32010 and support circuit,
2. Timing and control circuit,



NOTES: 1. Each time-slot corresponds to one channel, 8-bits wide. There are 256 channels per 125-μs frame.
2. Each time-slot corresponds to one channel, 8-bits long. There are 32 channels per 125-μs frame.

Figure 13. Possible Backplane Timing

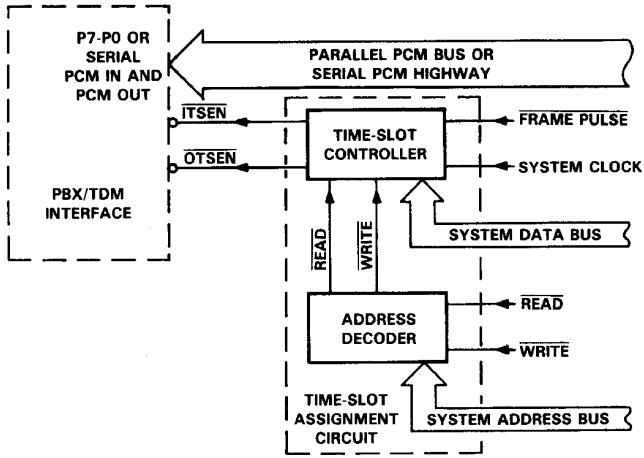
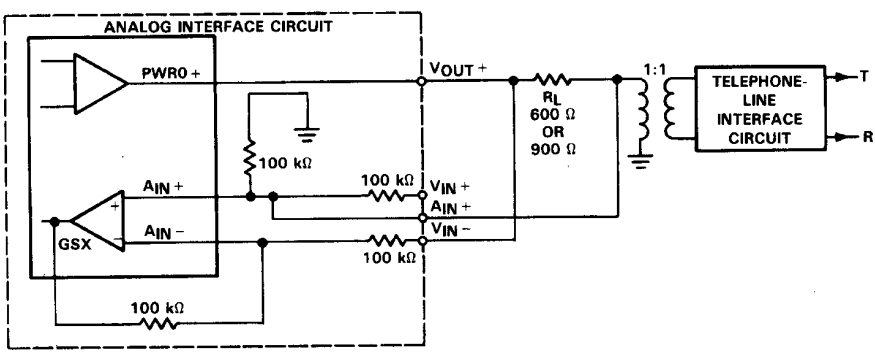


Figure 14. Typical Backplane Interface



NOTES: 1. R_L balances telephone line.
2. 6-dB loss from V_{OUT} and to telephone line.
3. 6-dB gain from telephone line to GSX.

Figure 15. 2/4-Wire Transformer Connection

3. Three PBX/TDM interface circuits, and
4. PCM linearization circuit (optional).

Three-way conferencing is the simplest method of conferencing; however, some system considerations are required. Return loss and noise performance are most important factors in a conferencing design. Return loss refers to the amount of attenuation of a reflected signal: the larger the return loss, the smaller the reflected signal. Reflections occur where there are 2/4-wire conversions in the network. In long network paths, such as satellite links, these reflections manifest themselves as echoes. (The application report, "Digital Voice Echo Canceller with a TMS32020,"[5] describes the implementation of an echo canceller using the TMS32020, the second-generation digital signal processor of the TMS320 family.) Return loss affects the "singing" margin and echo performance of the conference. "Singing" refers to an oscillatory state in a closed-loop system that occurs when excessive gain is inserted in the loop. Summing of two or more parties into a conference decreases the return loss, thereby increasing the chances of "singing" and degrading echo performance. In a three-way conference, the simplest method to reduce the effects of return loss is to insert transmission loss into the receive (RX) or transmit (TX) paths.

In the three-way conference node shown in Figure 17, each party is connected only to the other two parties, not to itself. Normally, the return loss is dominated by the party with the worst reflections. In a worst-case condition, if these two parties have similar reflection characteristics, the return loss decreases by 6 dB. To offset this, the transmission levels at summing junctions are decreased by 6 dB, i.e., a 6-dB loss insertion at TX IN or RX OUT. The loss insertion

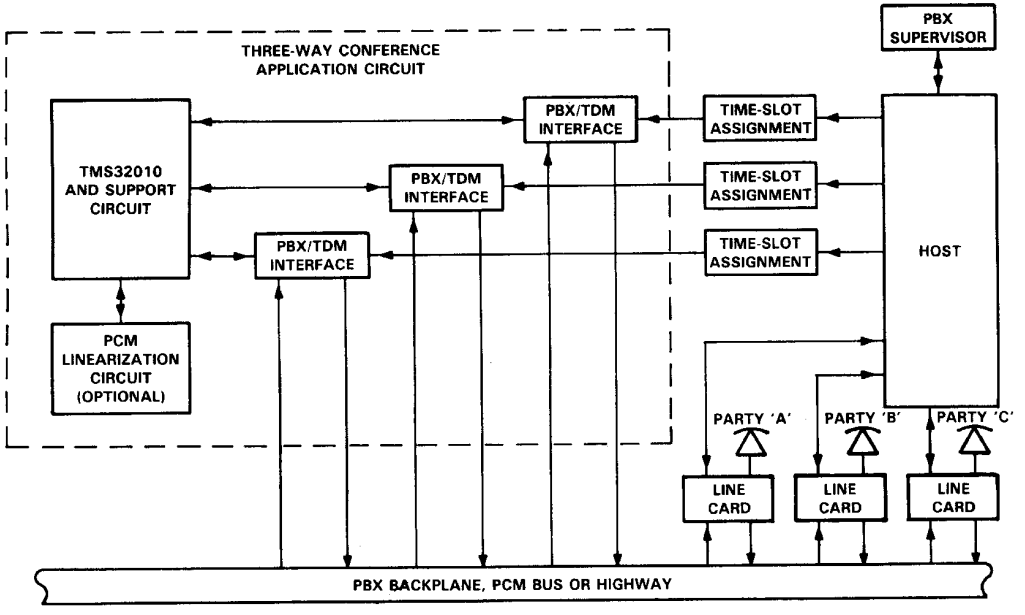


Figure 16. Three-Way Conference System Diagram

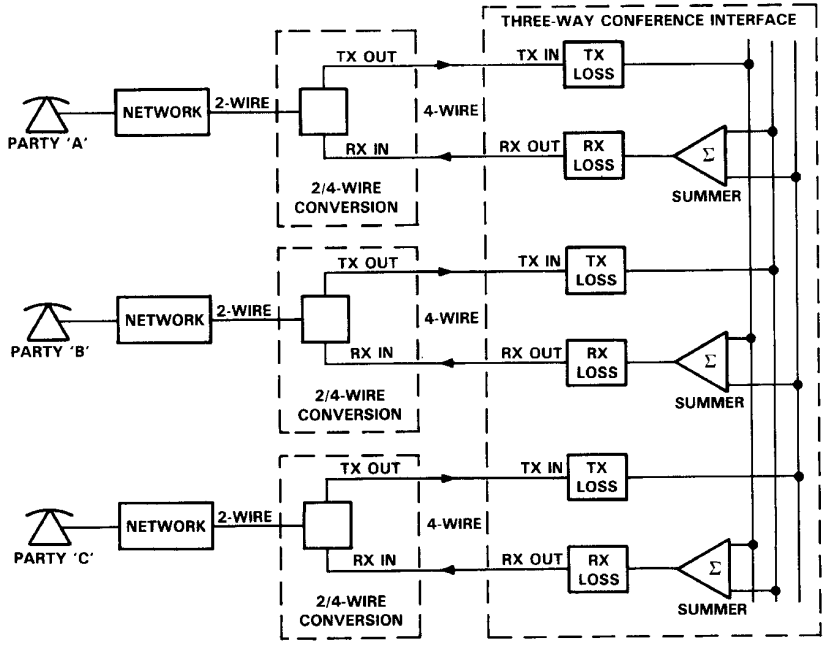


Figure 17. Three-Way Conference Configuration

slightly degrades noise performance. Noise performance refers to the subjective transmission quality of band-limited voice vs. band-limited noise. The noise performance of a conference, where all parties are summed together, always degrades to the party with the worst signal-to-noise ratio. In conferences where a large number of parties are connected together, voice-switching is used to limit the number of talkers summed together at one time, thereby limiting echo and noise performance degradation. Stringing together several three-way conferences to make a larger conference is not recommended, because noise performance will be substantially reduced due to compounded 6-dB insertion losses. The removal of this loss would likely degrade the echo performance of the conference and increase the chances of "singing".

The TMS32010 source code in Figure 18 can be used for a three-way conference. PCM linearization can be performed in hardware or software. The user can insert the appropriate linearization code in the space provided. Memory locations TA, TB, and TC correspond to the TX IN inputs at ports X, Y, and Z, respectively. Memory locations RA, RB, and RC correspond to the RX OUT outputs at port X, Y, and Z, respectively. The PBX supervisor maps the TMS32010 ports to their appropriate time-slots on the backplane. The 6-dB insertion loss is accomplished by a 15-position right-shift on the TX IN samples when loading the accumulator. This results in an effective single-position left-shift when using the upper 16-bits of the accumulator. Note that in the following code, WAIT is a label.

```
WAIT  BIOZ  WAIT   ; wait for 8-kHz interrupt
      IN    TA,X   ; input ports to memory
      IN    TB,Y
      IN    TC,Z
*
* Insert mu-/A-law to two's-complement linear
* conversion code here for ports X, Y, and Z.
      LAC   TA,15  ; sum ports X and Y for
*                  ; port Z
      ADD   TB,15
      SACH  RC,0
      LAC   TB,15  ; sum ports Y and Z for
*                  ; port X
      ADD   TC,15
      SACH  RA,0
      LAC   TA,15  ; sum ports X and Z for
*                  ; port Y
      ADD   TC,15
      SACH  RB,0
*
* Insert two's-complement linear to mu-/A-law
* conversion code here for ports X, Y, and Z.
      OUT   RA,X   ; output memory locations
*                  ; to ports
      OUT   RB,Y
      OUT   RC,Z
      B     WAIT
      END
```

Figure 18. Three-Way Conferencing Source Code

ADPCM Interface

Adaptive Differential Pulse Code Modulation (ADPCM) is a method of increasing voice-band transmission capacity. ADPCM is used in telecommunications systems to transcode 64-kbit/s PCM to 32-kbit/s ADPCM, thereby doubling the capacity of a transmission system. An implementation of a full-duplex ADPCM on a single TMS32010 is described in the application report, "32-kbit/s ADPCM with the TMS32010."[3] Although it does not provide bit-by-bit compatibility with CCITT recommendations, it follows the recommended model. The same report also describes the TMS32010 implementation of a half-duplex 32-kbit/s ADPCM algorithm recommended by CCITT.

The following paragraphs briefly examine two of many possible ADPCM applications: voice mail and transcoding in a PBX design.

Voice-mail in a PBX application is possible by using the ADPCM transcoder and the host processor. The application hardware for a voice-mail function, shown in Figure 19, requires the following modules:

1. TMS32010 and support circuit,
2. PBX/TDM interface circuit, and
3. Host interface circuit.

The ADPCM algorithm is stored on the TMS32010's program memory ROMs. The PBX supervisor instructs the host to initiate the voice-mail function and maps the PBX/TDM interface circuit to the appropriate time-slot on the backplane. The host initiates the transaction by writing a command to the application circuit and waits for ADPCM samples. The TMS32010 reads the μ/A-law PCM sample from the PBX/TDM interface. It then performs the PCM to ADPCM transcoding function. The ADPCM sample is then written to the host interface. The host reads the ADPCM samples at the host interface and stores them in a high-capacity storage device, such as a hard disk or bubble memory. At some point, the PBX supervisor instructs the host to terminate the transaction with the application circuit. When the mail is to be delivered, the PBX supervisor again instructs the host to initiate the transaction with the application circuit, and maps the PBX/TDM interface to the appropriate time-slot on the backplane. The host initiates the transaction by writing a command to the application circuit. The application circuit then waits for ADPCM samples from the host. The host retrieves the ADPCM sample from the storage device and writes it to the host interface. The TMS32010 reads the ADPCM sample, then performs the ADPCM to

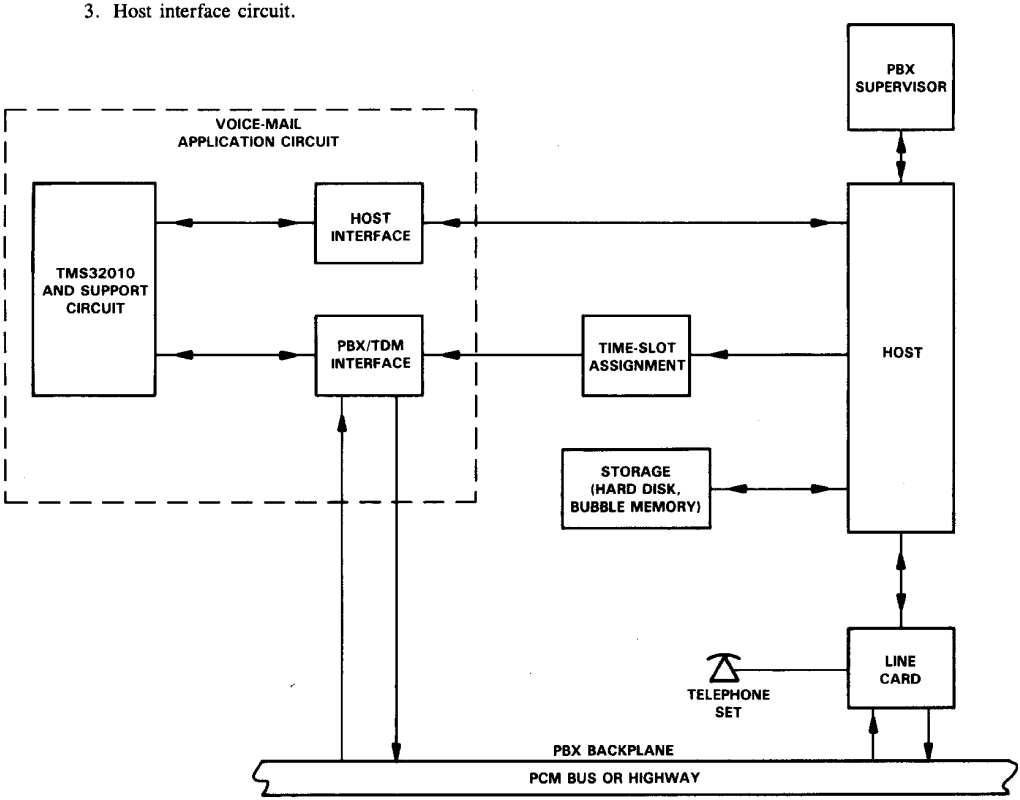


Figure 19. Voice-Mail System Diagram

PCM transcoding function, and writes the PCM sample to the PBX/TDM interface.

Transcoding within a PBX is possible between an 8-bit PCM highway and a 4-bit ADPCM highway. The transcoding application requires the following hardware blocks, as shown in Figure 20:

1. TMS32010 and support circuit,
2. Timing and control circuit, and
3. Two PBX/TDM interface circuits.

The PBX supervisor assigns the PBX/TDM interfaces to their respective time-slots on the PCM and ADPCM highways. The TMS32010 reads the sample from one of the interfaces, performs the transcoding function, then writes the sample to the second interface. The reverse operation is also applied, thereby providing full-duplex operation.

DTMF Detection Interface

In modern PBXs, DTMF detection is used to implement call features. DTMF signalling from the station-set is interpreted by the PBX for various features, such as outside line access, call-forwarding, conferencing, and voice-mail. DTMF detection is an integral part of any PBX design. DTMF detection is a popular signal processing function easily implemented on the TMS32010. TMS32010 source code for the DTMF detector is given in reference [4]. Multiple-channel DTMF detection is possible with a single TMS32010.

The DTMF detection application contains the following modules:

1. TMS32010 and support circuit,
2. Timing and control circuit,
3. Host interface circuit,
4. PBX/TDM interface circuit, and
5. PCM linearization circuit (optional).

The DTMF detection algorithm[4] requires PCM linearization. When the DTMF detection algorithm used is time-critical and software linearization is not possible, the PCM linearization circuit is needed. A system diagram for DTMF detection is shown in Figure 21.

An example of system operation is as follows: The PBX supervisor is informed that a station-set is in an off-hook condition. The PBX supervisor prepares for DTMF detection by mapping the PBX/TDM interface and station-set to the same time-slot, and instructing the host to detect digits and implement features. Note that the station-set is interfaced to the PBX backplane with a line-card. The host writes a command to the host interface instructing the application circuit to begin DTMF detection. The host waits for digits to be returned at the host interface. The TMS32010 reads samples from the PBX/TDM interface and implements the DTMF detection algorithm. When a DTMF digit is detected, the value is written to the host interface. The host reads the digit and implements the requested feature. The host then writes a command to the host interface, instructing the application circuit to terminate DTMF detection.

SUMMARY

The TMS32010 is well suited for a variety of telecommunications applications, ranging from trunk circuits to PBX systems. The hardware described in this application report allows the designer to apply some new ideas to end-product designs. Various digital PBX applications in the following areas were described: PBX backplane, 2/4-wire transformer connection, telephony test-set, three-way conferencing, ADPCM transcoding, and DTMF detection. Standalone systems applications, such as analog interfaces and testing, were also shown. All the applications can be transposed to the TMS32020, the second-generation digital signal processor.

The TMS32010 offers cost-effective solutions for many telecommunications applications requiring digital signal processing. Further cost reduction is possible using the TMS320M10. This mask version of the TMS32010 provides a 1523-word on-chip program ROM. Texas Instruments provides easy-to-use low-cost tools for circuit and algorithm development. The entire TMS320 family of digital signal processors is extensively supported by a staff of application engineers at the factory and in the field.

REFERENCES

1. *Companding Routines for the TMS32010* (Application Report), Texas Instruments (1984).
2. *Types TCM2913, TCM2914, TCM2916, TCM2917 Combined Single-Chip PCM Codec and Filter* (Data Sheet SCTS012), Texas Instruments (1983).
3. J.B. Reimer, M.L. McMahan, and M. Arjmand, *32-kbit/s ADPCM with the TMS32010* (Application Report), Texas Instruments (1985).
4. P.Mock, "Add DTMF Generation and Decoding to DSP-μP Designs," *EDN* (March 21, 1985).
5. D.G. Messerschmitt, D.J. Hedberg, C.R. Cole, A. Haoui, and P. Winship, *Digital Voice Echo Canceller with a TMS32020* (Application Report), Texas Instruments (1985).
6. *Precision Digital Sine-Wave Generation with the TMS32010* (Application Report), Texas Instruments (1984).
7. G. Dash, "Understand FCC Rules When Designing Telecomm Equipment," *EDN* (May 16, 1985).

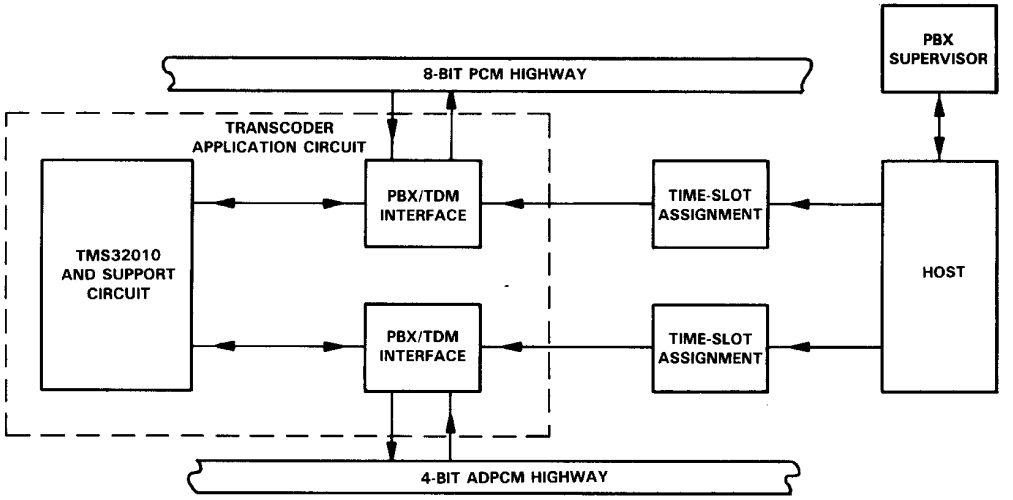


Figure 20. PBX Transcoder System Diagram

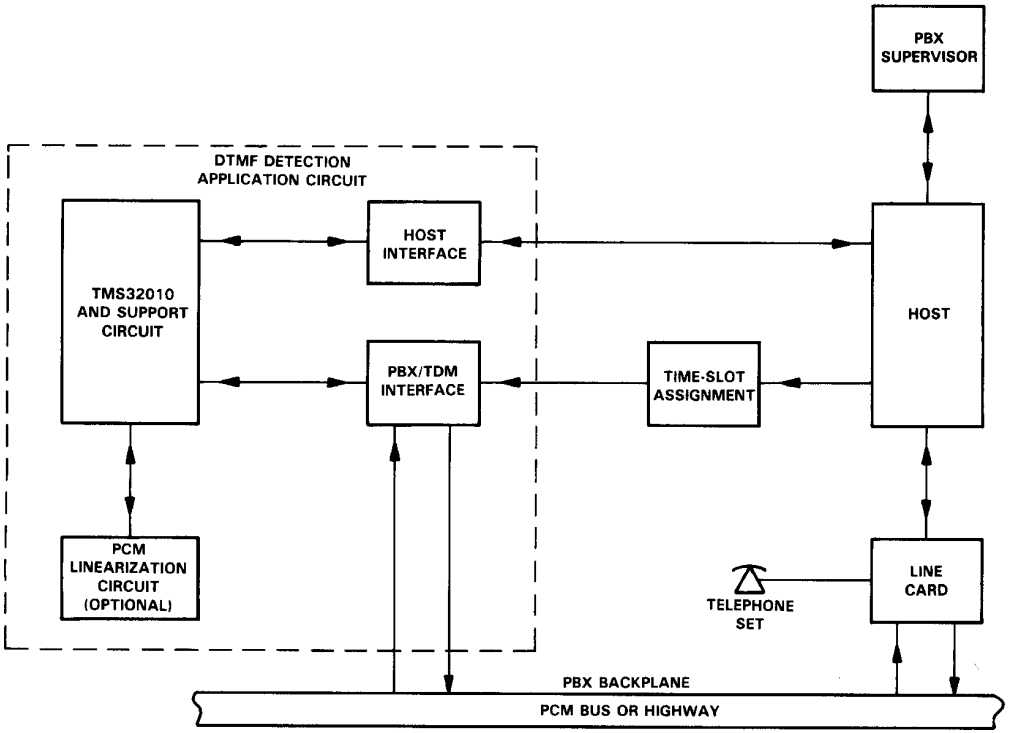


Figure 21. DTMF Detection System Diagram

APPENDIX

PCM Linearization Code

```
D Line# 1     7
      1 C*****************************************************************
      2 C
      3 C THIS PROGRAM COMPUTES MU-LAW PCM TO 14-BIT TWO'S COMPLEMENT
      4 C LINEAR CODES FOR PURPOSES OF PROGRAMMING PROMS AND EPROMS.
      5 C
      6 C GENERATED IS AN OBJECT FILE IN THE INTEL DATA RECORD FORMAT.
      7 C
      8 C WRITTEN BY JEFF ROBILLARD, TEXAS INSTRUMENTS INC.
      9 C              AUGUST 6, 1985
     10 C
     11 C*****************************************************************
     12 C
     13 C PROGRAM WRITTEN ON THE TEXAS INSTRUMENTS PROFESSIONAL COMPUTER
     14 C USING MICROSOFT FORTRAN.
     15 C
     16 C*****************************************************************
     17 C
     18 C MU-LAW LINEARIZATION FORMULA:
     19 C S=SEGMENT
     20 C Q=QUANTIZATION CHORD
     21 C
     22 C LIN=(2**S(2*Q+33))-33
     23 C
     24 C*****************************************************************
     25 C
     26 C DEFINE CONSTANTS AND INITIALIZE
     27 C
     28        INTEGER S,Q,LIN,LINBB
     29        DIMENSION LINBB(512)
     30 C
     31 C 2'S COMPLEMENT LINEARIZATION CALCULATION
     32 C
     33        DO 100 J=1,128,1
1    34        JJ=J-1
1    35        Q=MOD(JJ,16)
1    36        S=JJ/16
1    37        LIN=(2**S*(2*Q+33))-33
1    38        LINBB(J)=LIN/256
1    39        LINBB(128+J)=(65536-LIN)/256
1    40        LINBB(256+J)=MOD(LIN,256)
1    41        LINBB(384+J)=MOD((65536-LIN),256)
1    42      100 CONTINUE
     43        LINBB(129)=0
     44        LINBB(385)=0
     45        CALL HEXFILE (LINBB,'U-LIN.HEX',512)
     46        STOP'LINEARIZATION COMPLETED'
     47        END

Name    Type         Offset P Class

J       INTEGER*4      2050
JJ      INTEGER*4      2062
LIN     INTEGER*4      2074
LINBB   INTEGER*4         2
MOD                          INTRINSIC
Q       INTEGER*4      2066
S       INTEGER*4      2070
     48 C
     49 C INTEL RECORD FORMAT GENERATOR
```



```
D Line# 1     7
     50 C
     51        SUBROUTINE HEXFILE (DEC,LABL,SIZE)
     52        CHARACTER LABL*9,REC,HX
     53        INTEGER DEC,SIZE,SUM,ADDR,CH
     54        DIMENSION DEC(SIZE),REC(43),HX(16),CH(43)
     55        OPEN(1,FILE=LABL,STATUS='NEW')
     56        HX(1)='0'
     57        HX(2)='1'
     58        HX(3)='2'
     59        HX(4)='3'
     60        HX(5)='4'
     61        HX(6)='5'
     62        HX(7)='6'
     63        HX(8)='7'
     64        HX(9)='8'
     65        HX(10)='9'
     66        HX(11)='A'
     67        HX(12)='B'
     68        HX(13)='C'
     69        HX(14)='D'
     70        HX(15)='E'
     71        HX(16)='F'
     72        CH(2)=1
     73        CH(3)=0
     74        CH(8)=0
     75        CH(9)=0
     76        REC(1)=':'
     77        I=SIZE/16
     78        DO 1000 J=1,I,1
1    79        ADDR=(J-1)*16
1    80        CH(4)=ADDR/4096
1    81        CH(5)=MOD(ADDR,4096)/256
1    82        CH(6)=MOD(ADDR,256)/16
1    83        CH(7)=MOD(ADDR,16)
1    84        DO 2000 L=1,16,1
2    85        LL=(L-1)*2+10
2    86        CH(LL)=DEC(ADDR+L)/16
2    87        CH(LL+1)=MOD(DEC(ADDR+L),16)
2    88   2000 CONTINUE
1    89        SUM=0
1    90        DO 3000 M=2,40,2
2    91        SUM=SUM+(CH(M)*16)
2    92   3000 CONTINUE
1    93        DO 3100 M=3,41,2
2    94        SUM=SUM+CH(M)
2    95   3100 CONTINUE
1    96        SUM=256-MOD(SUM,256)
1    97        CH(42)=SUM/16
1    98        CH(43)=MOD(SUM,16)
1    99        DO 4000 N=2,43,1
2   100        REC(N)=HX(CH(N)+1)
2   101   4000 CONTINUE
1   102        WRITE(1,1300) (REC(K2),K2=1,43,1)
1   103   1000 CONTINUE
    104        WRITE(1,1400)
    105        CLOSE(1)
    106   1300 FORMAT(43(A1))
    107   1400 FORMAT(':00000001FF')
    108        RETURN
```

```
D Line# 1      7
    109        END

Name    Type          Offset P Class

ADDR    INTEGER*4       2326
CH      INTEGER*4       2138
DEC     INTEGER*4          0 *
HX      CHAR*1          2121
I       INTEGER*4       2310
J       INTEGER*4       2314
K2      INTEGER*4       2358
L       INTEGER*4       2330
LABL    CHAR*9             4 *
LL      INTEGER*4       2342
M       INTEGER*4       2350
MOD                             INTRINSIC
N       INTEGER*4       2354
REC     CHAR*1          2078
SIZE    INTEGER*4          8 *
SUM     INTEGER*4       2346


Name    Type            Size    Class

HEXFIL                          SUBROUTINE
MAIN                            PROGRAM


 Pass One    No Errors Detected
             109 Source Lines
```



```
D Line# 1      7
      1 C*****************************************************************
      2 C
      3 C THIS PROGRAM COMPUTES 14-BIT TWO'S COMPLEMENT LINEAR TO MU-LAW PCM
      4 C CODE FOR PURPOSES OF PROGRAMMING PROMS AND EPROMS.
      5 C
      6 C GENERATED IS AN OBJECT FILE IN THE INTEL DATA RECORD FORMAT.
      7 C
      8 C WRITTEN BY JEFF ROBILLARD, TEXAS INSTRUMENTS INC.
      9 C             AUGUST 6, 1985
     10 C
     11 C*****************************************************************
     12 C
     13 C PROGRAM WRITTEN ON THE TEXAS INSTRUMENTS PROFESSIONAL COMPUTER
     14 C USING MICROSOFT FORTRAN.
     15 C
     16 C*****************************************************************
     17 C
     18 C MU-LAW TO LINEAR CONVERSION:
     19 C S=SEGMENT
     20 C Q=QUANTIZATION CHORD
     21 C
     22 C DETERMINE S BY TABLE LOOK-UP
     23 C Q=INT(INT((LIN+33)/2**S)/2
     24 C
     25 C*****************************************************************
     26 C
     27 $STORAGE:2
     28       INTEGER L,S,CPAND
     29       INTEGER*2 ULAW
     30       DIMENSION ULAW(16384)
     31       DO 100 J=1,32,1
  1  32       S=0
  1  33       L=J-1
  1  34       ULAW(J)=CPAND(L,S)
  1  35   100 CONTINUE
     36       DO 110 J=33,96,1
  1  37       S=1
  1  38       L=J-1
  1  39       ULAW(J)=CPAND(L,S)
  1  40   110 CONTINUE
     41       DO 120 J=97,224,1
  1  42       S=2
  1  43       L=J-1
  1  44       ULAW(J)=CPAND(L,S)
  1  45   120 CONTINUE
     46       DO 130 J=225,480,1
  1  47       S=3
  1  48       L=J-1
  1  49       ULAW(J)=CPAND(L,S)
  1  50   130 CONTINUE
     51       DO 140 J=481,992,1
  1  52       S=4
  1  53       L=J-1
  1  54       ULAW(J)=CPAND(L,S)
  1  55   140 CONTINUE
     56       DO 150 J=993,2016,1
  1  57       S=5
  1  58       L=J-1
  1  59       ULAW(J)=CPAND(L,S)
```

```
D Line# 1      7
1     60   150 CONTINUE
      61       DO 160 J=2017,4064,1
1     62       S=6
1     63       L=J-1
1     64       ULAW(J)=CPAND(L,S)
1     65   160 CONTINUE
      66       DO 170 J=4065,8192,1
1     67       S=7
1     68       L=J-1
1     69       ULAW(J)=CPAND(L,S)
1     70   170 CONTINUE
      71       DO 200 J=1,8192,1
1     72       ULAW(16384-J+1)=128+ULAW(J)
1     73   200 CONTINUE
      74       CALL HEXFILE (ULAW,'LIN-U.HEX',16384)
      75       STOP'U-LAW COMPANDING COMPLETED'
      76       END

Name    Type         Offset P Class

CPAND   INTEGER*2              FUNCTION
J       INTEGER*2     32770
L       INTEGER*2     32778
S       INTEGER*2     32776
ULAW    INTEGER*2         2


      77 C
      78 C U-LAW COMPAND FUNCTION
      79 C
      80       INTEGER FUNCTION CPAND(LIN,SEG)
      81       INTEGER LIN,SEG
      82       CPAND=SEG*16+(((LIN+33)/(2**SEG))-33)/2
      83       RETURN
      84       END

Name    Type         Offset P Class

LIN     INTEGER*2         0 *
SEG     INTEGER*2         4 *


      85 C
      86 C INTEL RECORD FORMAT GENERATOR
      87 C
      88       SUBROUTINE HEXFILE (DEC,LABL,SIZE)
      89       CHARACTER LABL*9,REC,HX
      90       INTEGER DEC,SIZE,SUM,ADDR,CH
      91       DIMENSION DEC(SIZE),REC(43),HX(16),CH(43)
      92       OPEN(1,FILE=LABL,STATUS='NEW')
      93       HX(1)='0'
      94       HX(2)='1'
      95       HX(3)='2'
      96       HX(4)='3'
      97       HX(5)='4'
      98       HX(6)='5'
      99       HX(7)='6'
     100       HX(8)='7'
     101       HX(9)='8'
```



```
D Line# 1      7
     102       HX(10)='9'
     103       HX(11)='A'
     104       HX(12)='B'
     105       HX(13)='C'
     106       HX(14)='D'
     107       HX(15)='E'
     108       HX(16)='F'
     109       CH(2)=1
     110       CH(3)=0
     111       CH(8)=0
     112       CH(9)=0
     113       REC(1)=':'
     114       I=SIZE/16
     115       DO 1000 J=1,I,1
1    116       ADDR=(J-1)*16
1    117       CH(4)=ADDR/4096
1    118       CH(5)=MOD(ADDR,4096)/256
1    119       CH(6)=MOD(ADDR,256)/16
1    120       CH(7)=MOD(ADDR,16)
1    121       DO 2000 L=1,16,1
2    122       LL=(L-1)*2+10
2    123       CH(LL)=DEC(ADDR+L)/16
2    124       CH(LL+1)=MOD(DEC(ADDR+L),16)
2    125   2000 CONTINUE
1    126       SUM=0
1    127       DO 3000 M=2,40,2
2    128       SUM=SUM+(CH(M)*16)
2    129   3000 CONTINUE
1    130       DO 3100 M=3,41,2
2    131       SUM=SUM+CH(M)
2    132   3100 CONTINUE
1    133       SUM=256-MOD(SUM,256)
1    134       CH(42)=SUM/16
1    135       CH(43)=MOD(SUM,16)
1    136       DO 4000 N=2,43,1
2    137       REC(N)=HX(CH(N)+1)
2    138   4000 CONTINUE
1    139       WRITE(1,1300) (REC(K2),K2=1,43,1)
1    140   1000 CONTINUE
     141       WRITE(1,1400)
     142       CLOSE(1)
     143   1300 FORMAT(43(A1))
     144   1400 FORMAT(':00000001FF')
     145       RETURN
     146       END

Name    Type         Offset P Class

ADDR    INTEGER*2     32934
CH      INTEGER*2     32840
DEC     INTEGER*2         0 *
HX      CHAR*1        32823
I       INTEGER*2     32926
J       INTEGER*2     32928
K2      INTEGER*2     32950
L       INTEGER*2     32936
LABL    CHAR*9            4 *
LL      INTEGER*2     32942
M       INTEGER*2     32946
```

```
D Line# 1     7
MOD                              INTRINSIC
N       INTEGER*2      32948
REC     CHAR*1         32780
SIZE    INTEGER*2          8 *
SUM     INTEGER*2      32944


Name    Type           Size    Class

CPAND   INTEGER*2              FUNCTION
HEXFIL                         SUBROUTINE
MAIN                           PROGRAM

 Pass One     No Errors Detected
             146 Source Lines
```