THE BELL SYSTEM TECHNICAL JOURNAL

DEVOTED TO THE SCIENTIFIC AND ENGINEERING ASPECTS OF ELECTRICAL COMMUNICATION

Volume 60 February 1981 Number 2

A Digital Concentrator for the SLC™-96 System

By R. J. CANNIFF

(Manuscript received April 22, 1980)

The SLC™-96 subscriber loop carrier system is a digital subscriber carrier system serving up to 96 single-party customers. The system can be configured with an optional plug-in which digitally concentrates two standard, 1.544 megabit, serial pulse-code modulation (PCM) bit streams into a single stream, thereby concentrating 48 customers onto a single T1 digital line. The concentrator employs a custom NMOS LSI chip providing a full-access, time-slot interchange function. It has microcomputer controllers at the two ends of the system to control time-slot assignments. The development of the concentrator involved challenges in chip design, software design, and performance testing.

I. INTRODUCTION

As the state of the art in electronics has advanced, so have the inroads of electronics into the loop plant as a supplement to or replacement for cable.[1] The application of pair-gain systems to loops has been of particular importance in recent years. The SLC™-96 subscriber loop carrier systems is the latest in a line of digital loop carrier systems to provide improved features and reduced cost.[2] The SLC-96 system serves up to 96 single-party subscribers between a Central Office Terminal (COT) and Remote Terminal (RT) using standard pulse-code modulation (PCM) coding over facilities such as T1. System features include a variety of channel units, channel and drop testing provisions, and spare digital line switching. System versatility is further enhanced by three modes of operation (each mode uses one additional T1 line for protection):

Mode I: Carrier only; 96 full-time channels on four working T1 lines.

Mode II: Carrier concentrator; 96 channels on two working T1 lines.

Mode III: Special services only; 48 special service channels on two working T1 lines.

This article discusses the Time Assignment Unit (TAU) which is the concentrator employed in Mode II operation. In a fully equipped Mode II system there are actually two identical, but independent, concentrators, each concentrating 48 lines onto 24 time-slots of a T1 line, and thereby eliminating two main T1 lines (see Fig. 1). In many cases the saved T1 lines provide more advantage than is apparent. The benefits arise in applications limited by apparatus case size for holding repeaters, by small cables where the purpose of pair-gain systems is to avoid adding new cable, and, similarly, in situations that use several systems in parallel along the same route where the number of saved pairs can be very significant. For long-distance systems, of course, the saved lines result in important cost savings.

Section II of this article discusses the traffic-handling ability of the TAU and the traffic administration provisions. This is followed by a functional description of the concentrator operation in Section III. Section IV discusses first the hardware features provided by the custom Time-Slot Interchange (TSI) chip and then enumerates the features provided by software. Selected implementation details for both hardware and software are expanded upon in Section V. Performance testing, that is, the equipment and resources that were used to adequately stress the concentrator to confirm its design, is discussed in Section VI.

II. TRAFFIC CONSIDERATIONS

The concentrator performs a full-access, digital time-slot interchange function. The very conservative two-to-one concentration ratio that was chosen provides for the inclusion of a limited number of dedicated (unconcentrated) special-service circuits as well as a large number of multiparty circuits. It is important to note that a single channel unit for special service occupies the same physical space as a standard unit for dual voice frequency and thus reduces the number of lines to be concentrated by two. Also, the special-service unit requires only a single T1 time slot. Thus, regardless of how many special-service units are inserted, the concentration ratio remains fixed at two to one, but the group size is reduced. The blocking probability for concentrated lines increases as more special-service units are added, because of the reduced group size. By design, the number of special-service channel units is limited to eight and must be physically inserted in the last four channel unit positions of each shelf.

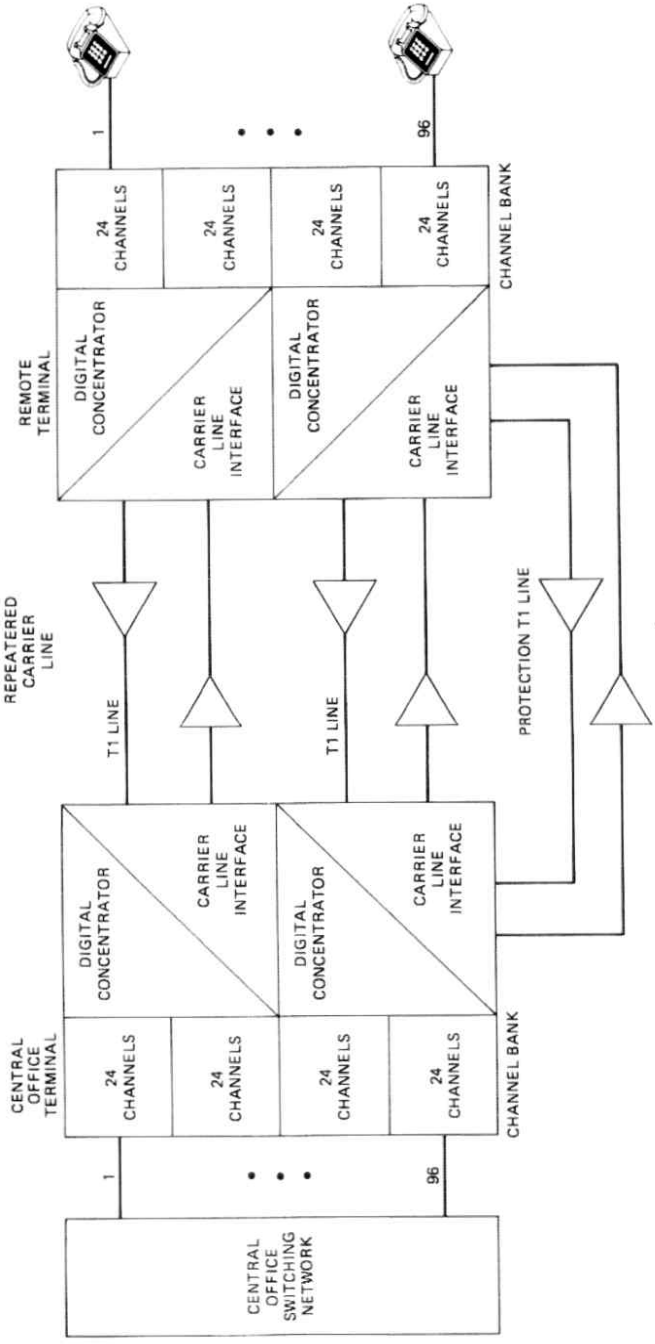


Fig. 1—The SLC™-96 system-carrier concentrator configuration—Mode 2.

For heavy loading or mandatory monitoring purposes, six traffic administration aids are provided:

(*i*) A two-digit numeric display is provided on the faceplate of the COT TAU which displays, upon demand, the peak traffic in hundred call seconds per hour (CCS), per concentrated T1 time slot since the internal traffic register was last cleared. The register is cleared by means of a pin jack on the faceplate.

(*ii*) The same display will also indicate the number of blocked calls, up to a maximum of 15, since the internal blocked calls register was last cleared. This register is cleared by means of the pin jack.

(*iii*) A traffic lamp is provided on the faceplate, in conjunction with a minor alarm, that indicates there has been two or more blocked calls for two out of three weeks running. This alarm must be manually cleared through the pin jack.

(*iv*) A relay closure is provided to remote to an electronic switching system (ESS) office that all T1 time slots are in use. The office can then divert terminating calls and provide its own reorder tone while keeping individual line blockage statistics. (In a non-ESS Central Office this diversion will not occur and the TAU will provide a digital reorder tone.)

(*v*) A second relay is provided that outpulses peak weekly traffic in CCS, once per week, to a remote traffic-monitoring register at the rate of one pulse per second.

(*vi*) A third relay is provided that outpulses the number of blocked calls as they occur to a remote blocked calls monitoring register at the rate of 1 pulse per second. There is no saturation limit here as there was in the faceplate display.

III. TAU FUNCTIONAL DESCRIPTION

The following definitions are given to clarify all discussion that follows: A line refers to a subscriber loop at the RT or a wire pair appearance at the COT. A channel unit is the physical plug-in serving one or two lines and providing the per-line circuit functions. A channel is the electrical path from a channel unit to the Transmit Receive Unit (TRU) serving it, and the time slot reserved for the line into and out of the TRU on the 1.544 megabit serial PCM busses. Channels enter or leave the concentrator as time slots from or to the TRUs. Time slots on the T1 line interfacing to the concentrator are referred to as trunks, because they provide a limited number of shared paths between the two terminals of the system, in analogy with traditional trunking facilities between central offices.

3.1 System block diagram

Figure 2 shows a simplified system block diagram of the TAUs. The microcomputer controllers are realized using the Bell Laboratories

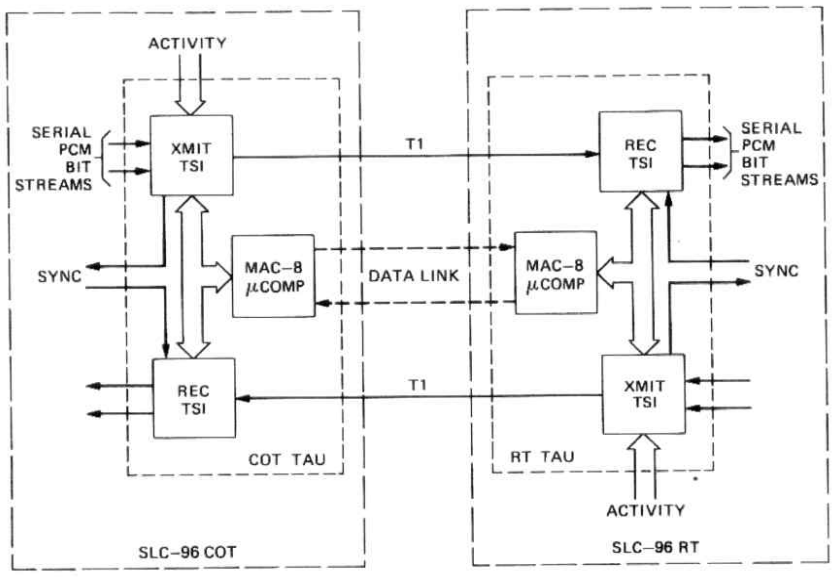


Fig. 2—TAU system block diagram.

designed MAC-8 microprocessor.[3] The MAC-8 microcomputers maintain control over the Time-Slot Interchange (TSI) chips and talk to each other over a data-link channel derived from the subframe bits on the T1 line. The transmit TSI selectively combines two 1.544 megabit, serial PCM bit streams, one from each of the TRUs it serves. (A TRU performs the A/D, D/A, and framing tasks for 24 channels.) The resulting T1 signal is sent to the receive TSI at the other end of the system where it is expanded into two streams, for sending to corresponding TRUs. The TSIs provide full access, meaning that any one of the incoming (outgoing) 48 lines or time slots has access to any one of the 24 outgoing (incoming) trunks, where trunks are time slots on the T1 line. The COT TAU handles trunk assignments and, in general, controls the concentrator. The RT TAU acts more as a slave. When the MAC-8 controller assigns a line to a specific trunk, that line will keep the trunk for the duration of the call and in no way inhibits any of the other lines from accessing any of the other trunks.

A line-service request, called "activity," is picked up by the transmit TSI by accessing the A and B bit signaling busses on the system backplane. A and B bits are the standard nomenclature for per-channel signaling that indicates off-hook, ringing, etc. The activity is stored in memory in the TSI from which it can be retrieved by the MAC-8 through the TSI microcomputer address and data ports. Activity at the RT is passed over the data link to the COT where the line/trunk assignments are determined.

The transmit and receive TSIs must be synchronized to the TRUs they serve and appropriate signals are provided for this purpose. No synchronization is assumed between transmit and receive TSIs. Further, no synchronization is assumed between the MAC-8, the TSIs, and the data link. The TSIs are accessed by means of a handshake procedure. The data-link frame signals are polled to determine when message processing is needed.

If all 24 trunks are busy, provision is needed for feeding a fast-busy (overflow) tone to the COT channel units. This is done digitally through the receive TSI, resulting in significant cost savings in the channel units. The receive TSI allows the assignment of up to 24 lines to "busy trunks" whereby the selected lines receive a fast-busy tone in PCM form as read from a code-word table stored in the MAC-8 program memory. Signaling information (A and B bits) is also stored in these codes and thereby allows the channel unit to recognize that it is getting the fast-busy tone and accordingly trip ringing without charge, prior to applying the tone on the blocked customer's line.

3.2 Circuit block diagram

Figure 3 shows a simplified schematic for the COT TAU. The RT TAU is nearly identical with the basic exceptions that Output Port 2 is removed and there is no Read Only Memory 2 (ROM2). The Random Access Memory (RAM), Read Only Memory (ROM), Input Port, and Output Ports connect to the MAC-8 bus as in any normal microcomputer. The custom TSI chips were also designed to connect directly on the bus and appear to the MAC-8 as programmable peripheral chips. The MAC-8 talks to the TSI chips by means of a handshake procedure

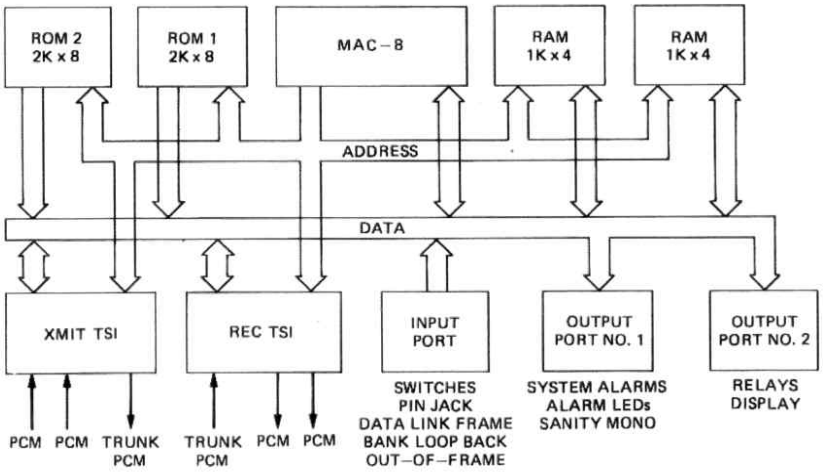


Fig. 3—COT TAU schematic.

using the MAC-8 Data Ready line. The clock frequency for the TSI and MAC-8 is 1.544 MHz. Three to nine wait states result, as required and generated by the TSI on a read or write. The TSI chip generates and responds to the necessary signals for system synchronization.

Output Port 1 provides four alarm light emitting diodes (LEDs) on the faceplate and three system alarms. For the COT TAU, the LEDs indicate alarms for COT, RT, traffic, and special-service channels. The RT TAU has only two LEDs indicating RT alarm and unit alarm for special-service channels. The system alarms are minor, major, and major with channel unit disable. Output Port 1 also provides an output for strobing the sanity monostable, a timeout device that checks on proper sequencing of the program. Output Port 2 provides the dual-digit numeric display for indicating traffic and blocked calls on the faceplate and also services the three relays used for remoting concentrator status. The Input Port provides for the display selection switches and the internal register-clearing pin jack, all mounted on the faceplate. It further provides inputs for the data-link frame signals which are polled to determine data-link message requests. Also, the Input Port allows accessing a receiver out-of-frame signal and a bank loop-back signal.

The COT TAU has 4K bytes of program memory and the RT TAU has 2K bytes. Each are realized using 2K byte ROM chips. At both COT and RT, 1K bytes of RAM are provided, though the COT uses less than one-third of the available memory and the RT uses only about one-fourth. The TAU plug-ins are printed circuit cards measuring approximately 4 inches by 10 inches as pictured in Fig. 4. It was required that the TAUs be sized to physically replace the Line Interface Units for the T1 lines that are not needed in the concentrated mode. Power supplies for the TAU are 5V and 12V, with typical dissipation being about 3 watts. Special requirements had to be met for the RT TAU so that it could work over the temperature range of −40 to +85 degrees Celsius.

IV. HARDWARE AND SOFTWARE FEATURES

4.1 Features of the TSI

The TSI chip was designed to be universal, in the sense that it is package lead and microcomputer programmable for use at COT or RT, for either the transmit or receive function. In making a universal chip with all the features mentioned below, it is possible, and very desirable, to reuse pieces of the hardware inside the chip. Thus, for example, pieces of RAM and other hardware used for busy trunk assignments in the receive TSI mode are alternatively used in the transmit TSI mode for the activity and "TNEN" collection (discussed later). Similarly, many input and output package leads serve dual purposes. In addition to the basic features, several very important additional features are

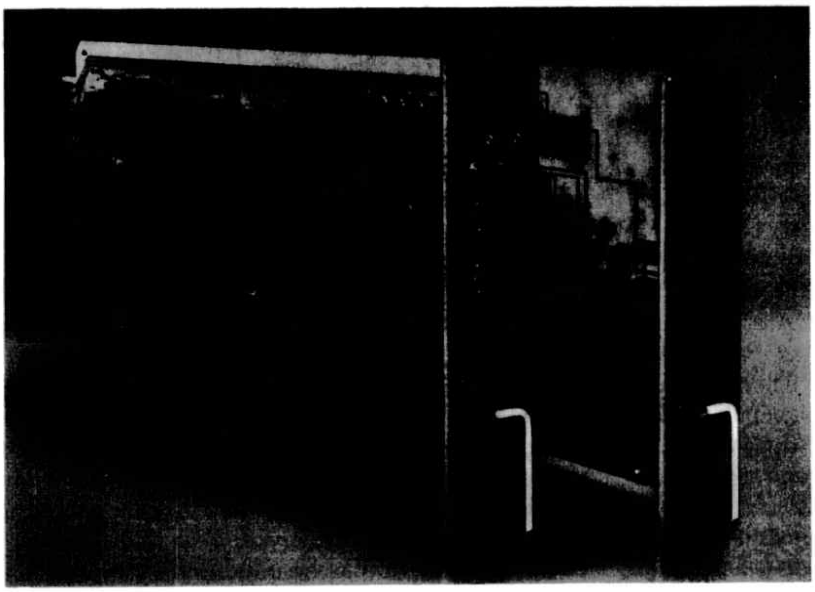


Fig. 4—TAU circuit packs.

provided by the TSI chip. Most of these features cost very little hardware, that is, they increase chip complexity very little, while providing useful features.

Figure 5 is a block diagram of the TSI chip. The heart of the chip is a RAM used for PCM data, trunk assignment information, busy trunk-assignment information, activity data, "TNEN" data, per-line test bits, 1 byte of fast-busy tone buffer, and 2 bytes of ROM. Incoming and outgoing serial PCM data is handled in bytes inside the chip by feeding the data through serial-to-parallel registers (Registers 1 and 2) and parallel-to-serial registers (Registers 2 and 3). The use of Register 2 depends on whether the TSI is performing in the transmit or receive mode. The frame bit is normally stored in the FR FF (frame flip-flop). The main-control logic provides the signals for controlling all the chip's registers, multiplexers, and RAM. Multiplexers are included for selecting address and data for the RAM and for selecting output onto the microcomputer data bus.

4.1.1 Interchanging time slots

The time-slot interchange is executed by the method of writing data into a RAM sequentially and reading it selectively for the transmit concentrator function and vice versa for the receive function. The trunk assignments (24 bytes), which are addresses of the desired PCM line data, are stored in the RAM, yet are used to address the RAM

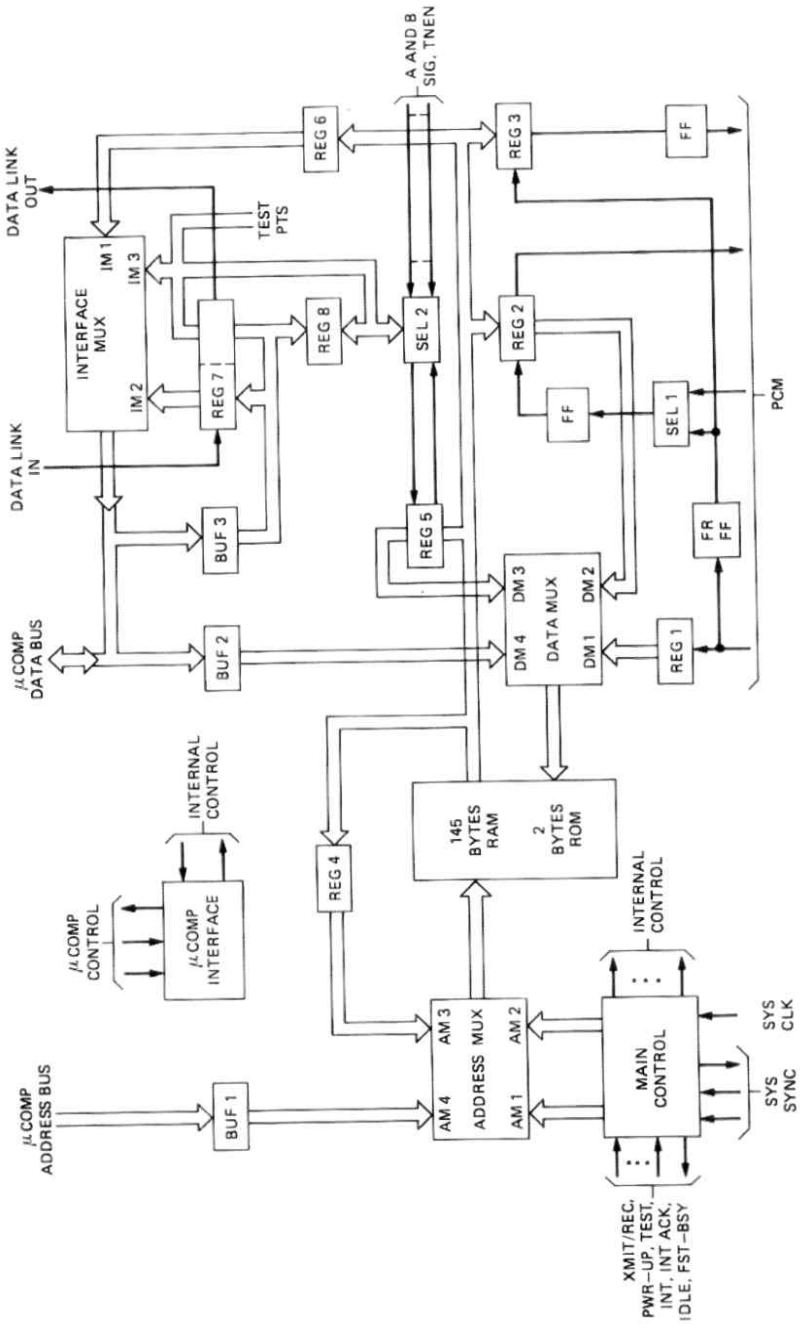


Fig. 5—Time-slot interchange chip.

selectively by a feedback register (Register 4) to the address bus. Each of the trunk (busy trunk) assignments are fed back in turn to address the desired line memory. In the transmit mode, if a trunk is unassigned, the ROM locations are accessed to send idle code on the T1 line. In the receive mode, if a line is unassigned, idle code from the PCM memory is read out for the line or idle is forced by means of a cleared "enable bit" (discussed later).

Since the concentrator is positioned between the TRUs and the T1 line, A and B bit-signaling information is already contained in the PCM bit streams when they are received by the TSI. It is necessary, therefore, that the TSI insure integrity of signaling frames. Signaling frames are the 6th and 12th frames of a 12-frame sequence and contain the per-channel A and B bit signaling information in the least significant bit of the PCM code words. Integrity is maintained by having two RAM sections of PCM data (48 bytes each). While writing (reading) in one section sequentially, the other section is read (written) selectively; the roles are reversed every frame. The frame bit is correspondingly delayed to match up with the outgoing data. The result is that signaling-frame integrity is maintained while the data experiences a fixed delay of approximately two frames (250 microseconds) end-to-end regardless of the line or trunk.

4.1.2 Microcomputer access

The microcomputer has access to the TSI memory through the address and data ports of the TSI chip (see Fig. 5). This allows all the memory locations, including the PCM data memory, to be written and read. The main control block of the TSI contains a frame counter for controlling all chip sequencing. When the counter is in a state not needed for a specific internal function, that clock cycle can be used to respond to a microcomputer read or write request. The result is a variable number of wait states, as mentioned earlier, because the microprocessor can request a TSI access at an arbitrary time. The microcomputer talks to the TSI by means of Read/Write, Select, and Busy leads that connect to the microcomputer interface circuitry. On a read operation the data is held in Register 6 so that the processor can use as much time as necessary to recognize the data. Some of the address space of the TSI is reserved for addressing the data-link register (Register 7) and activity-mode control register (Register 8).

4.1.3 System synchronization

The TSI must be synchronized to the TRUs to know where every bit in the incoming and outgoing bit streams is located. For the transmit function, the TSI puts out a superframe synchronization signal which the TRUs can accept and lock to. For the receive function the TSI

accepts an out-of-frame and synchronization signal from the TRU. The receive function is complicated, however, by the reframing process. Since the TSI exists on the T1 line side of the TRU, the reframing executed by the TRU is carried out after the bit stream has passed through the TSI. This implies the need for special modes of operation while the system is out-of-frame to assure reframe.

4.1.4 Digitally generated, fast-busy tone

A feature of the TSI in the receive mode is the ability to feed a digitally generated, fast-busy tone to a terminating connection when all trunks are busy. In order to receive the fast-busy tone, a line must be assigned to a "busy trunk." The assignment mechanism is nearly identical to assigning a line to a normal trunk. The assignment consists of an address of the line location into which the fast-busy tone bytes are to be placed. These assignments are fed back in turn through Register 4 to select the desired lines. Up to 24 lines can receive the tone byte simultaneously. The source of the current PCM tone byte is a holding register (Register 5) which is indirectly updated every frame by the microcomputer, through the fast-busy tone buffer byte in RAM. Two signaling codes (A=B=0 or A=B=1) can be sent out with the tone byte by making the least significant bit always 0 or 1. Only one of these codes is used in the TAU for signaling the channel units. The fast-busy tone is simulated by a sequence of 48 PCM coded bytes stored in the program ROM which emulates the dual-tone frequency needed.

4.1.5 A and B signaling bit collection

One very important side feature of the TSI in the transmit mode is that it gathers A and B bit information to supply the microcomputer with line activity information. The simplest and fastest way to collect this data is to tap into the A and B bit busses on the system backplane. The A and B bit data are available there every frame and, because the TSI and the TRU are synchronized, the precise time for each line's A and B bit data is known.

Since the concentrator is interested only in activity and not in the precise A and B bit signaling states, the A and B information is condensed. No activity (idle) is signaled by the channel units as A = B = 0 or A = B = 1, depending on whether the location is COT or RT. Thus, the A and B bit collection hardware just looks for a deviation from the idle pattern. It is further desirable to have an elementary filtering effect so that if there is any activity within a certain time period during which the chip is told to collect activity, that activity is caught and held, with the result that the microcomputer is not required to make numerous, closely time-spaced searches for activity.

As a result, the TSI chip offers three modes for activity collection:

sift for zeroes, sift for ones, or no collection. Since the A and B bit information is combined, all 48 lines of activity information are stored in 48 bits or six bytes of the RAM. The collection mode control is set by the microcomputer in Register 8 (see Fig. 5). Selector 2 and Register 5 are used to collect each byte of data before being transferred to the RAM for storage.

4.1.6 TNEN bit collection

Another important side feature of the TSI chip is that it gathers the so-called "TNEN" bits. The per-channel TNEN bit tells the TRU to encode PCM data for the channel or to send digital data off a backplane bus. Since two channels are associated with a single physical channel unit, the two corresponding TNEN bits can be used as an indication of the class of service desired by a channel unit. Thus, four different types of channel unit can be identified based on the permutations of the two TNEN bits, and the concentrator can then take the required action. For example, standard dual voice-frequency units get concentrated service while single voice special and single data special units get a permanent trunk.

The TSI chip in the transmit mode collects TNEN bits in almost the same way it collects activity (using Registers 5 and 8 and Selector 2), because the TNEN bits are available on the backplane bus in the same format as the A and B bits. The only difference is that there is only one TNEN bit per line and no sifting is performed, that is, they are just collected. The TSI cannot collect activity and TNEN simultaneously since they use common hardware. The TNEN bits are stored in different memory bytes in the RAM, however; thus, by means of a fourth collection mode (in addition to the three activity modes), the microcomputer can get a snapshot of the TNEN bits between normal activity collection. This can be done in just a little more than one frame of time so that no significant activity is lost.

4.1.7 Per-channel enable bits

An additional feature of the TSI, and one that has proved to be very powerful for real-time testing, is the provision of per-channel enable bits. The cost of this memory is low because it employs the upper two bits of the trunk assignment bytes which are not needed for addressing the 48 possible lines. Each of the 48 enable bits (two per trunk assignment location) operates independently on its associated line and inhibits PCM writes into the line memory in the transmit TSI or forces idle PCM to the line in the receive TSI mode, regardless of the contents of the PCM line memory. This feature allows the microcomputer to read and write the line memory to check for faults without worrying about the data being overwritten or about extraneous data being sent

to an idle channel. Since the enable bit feature does not inhibit the line from being assigned to a trunk, it allows, with the aid of the microcomputers at both COT and RT, PCM test codes to circulate around the entire connection loop prior to customer cut through. This checks approximately 90 percent of the hardware involved in a connection and, as implemented, adds only about 15 ms to the connect delay time. This feature employs pairs of enable bits as they are read out to Register 4 in conjunction with the trunk assignment for that time slot. While in Register 4, the enable bits can cancel writing data to the RAM from Register 1 (Transmit Mode) or force Registers 2 or 3 to be loaded with idle code (Receive Mode) for the particular lines that correspond to the time slot in question.

4.1.8 Data-link register

Another auxiliary feature built into the TSI to relieve real-time constraints on the processor is an 11-bit data-link shift register (Register 7). The data link, as seen by the concentrator, consists of 11-bit packets of data, every 9 ms, in a serial format. The data-link register is loaded or unloaded in parallel by means of the microcomputer address and data ports, after which it is shifted asynchronously by the data-link clock. The microcomputer polls the data-link frame signals separately to determine when to read or write the shift register.

4.1.9 Initialization (power-up) sequence

A final feature of the TSI is a power-up and initialization sequence. By means of an external RC network, a latch internal to the TSI is set upon power up. Then after the clock starts, a 12-frame sequence must be passed through before the chip comes out of initialization. This dual method of RC timeout and clock timeout assures a robust initialization sequence that assures all memory is initialized and all trunks and busy trunks are deassigned. The initialization sequence is also very useful for manufacturing testing, as is an additional lead that allows breaking up the main-control counter sequence.

4.2 Program features

The software for the TAU MAC-8s was developed on a UNIX™ time-sharing-system (see Ref. 4). A Bell Laboratories microprocessor development tool for the MAC-8, called PLAID, was used for debugging and testing the code.[3]

The COT and RT programs are written in MAC-8 assembly language and are designed to fit into the available 4K bytes and 2K bytes of ROM, respectively. Assembly language was used, not only because of limited program capacity, but also because of stringent real-time constraints, which exist in part because of the decision to keep hard-

Table I—Code breakdown for the COT TAU functions

Function		Percent of Code
Call processing and message handling		38%
Self-diagnostics		37%
Data base consistency checks	(9%)	
Line/trunk fault handling	(10%)	
RAM, ROM, MAC-8 tests	(10%)	
Initialization and alarm filtering	(8%)	
Traffic and blocked calls recording		17%
Channel unit identification		5%
Other		3%

ware at a minimum where software can do the job. For this particular application, these decisions are still justified after the fact. It is also true that understandable code can be written easily in MAC-8 assembly because of its C-like syntax, thereby negating some normal aversion to assembly-level programming.[3] A characteristic of the code is that it is very heavy in register instructions, as would be expected for byte and time-efficiency reasons. (The MAC-8 employs 16 general-purpose registers residing in RAM.) Careful attention was given to register usage so that data required over long segments of a routine, or even between subroutines, could remain as register variables.

4.2.1 Software statistics

Table I shows a usage breakdown of the COT TAU code. The fundamental job of call processing and message handling represents only 38 percent of the code. The importance of self-diagnostics is obvious and reflects the concern and effort that was expended in this area.

Table II shows a breakdown of the code in terms of routines, instructions, and bytes. A large percentage of the code runs in response to interrupts generated by data-link message requests. For example, all trunk assignments and deassignments initiated by the COT TAU are triggered by the need to form a new data-link message for transmittal to the RT. A total of 3894 bytes (95 percent) of the available 4096 bytes are used. Table III lists the bytes of RAM required by the COT TAU MAC-8 program. Only 286 bytes (28 percent) of the available 1024 bytes are used.

Table II—Statistics for COT TAU MAC-8 assembly-language program

	Interrupt Processing	Background Processing	Total
Routines	28	6	34
Instructions	923	418	1341
Code (bytes)	2644	1199	3843
Bytes/instruction	2.86	2.87	2.87
Percent of code	69%	31%	100%
Data table (bytes)	49	2	51

Table III—Statistics for COT TAU R/W memory use

Program variables	64 bytes
Line/trunk data base	90 bytes
Stack allowance	100 bytes
MAC-8 registers	32 bytes
Total	286 bytes

4.2.2 Program flow

The COT TAU program can be envisioned as a Main (background) routine that runs when no other processing is needed and interrupt routines that respond to real-time call-processing requests.

The Main routine performs most of the sanity and memory-checking tests. Another job is to manipulate the "TNEN" bits into masks used to force trunk assignments or no assignments in response to the types of channel units that are inserted in the SLC™-96 bank. Several alarm filters are maintained; the Main routine examines these filters and outputs the correct system and faceplate alarms. The traffic and blocked-call calculations are also performed by the Main routine. The results are passed to an interrupt routine, which times the output for displaying on the COT TAU faceplate and for outpulsing via the relays. The Main routine also performs some housekeeping chores.

An interrupt from the receive TSI occurs every frame (125 μs). Because of the digitally generated, fast-busy tone feature, whereby PCM code words are read from the processor ROM to the receive TSI, this high-speed interrupt is needed. This interrupt is then counted down to time other functions for the interrupt routines (see Fig. 6). Every 2 ms the Data-Link Polling and PCM Test routine (Poll routine) is entered. This routine polls the data-link frame signals to determine when a data-link message must be transmitted or received. As a result, the Transmit and Receive Message routines are executed every 9 ms. The PCM test portion of the Poll routine refers to a function performed at the time of assigning a line to a trunk. Prior to customer cut-through, test PCM codes are circulated COT to RT to COT. The transfer of the test codes from receive TSI to transmit TSI is done by the processor by sampling at the 2 ms rate. Finally, the two-digit numeric display on the COT TAU faceplate is multiplexed at 6 ms intervals by the Display Mux routine.

4.2.3 Data-link message protocol

As mentioned previously, the concentrator data link consists of 11-bit framed messages. These 11-bit messages are grouped together in a protocol providing error protection by means of redundancy. All messages except "Idle" are 33-bit messages made up of three sequential packets of 11 bits. For digital central-office compatibility, normal

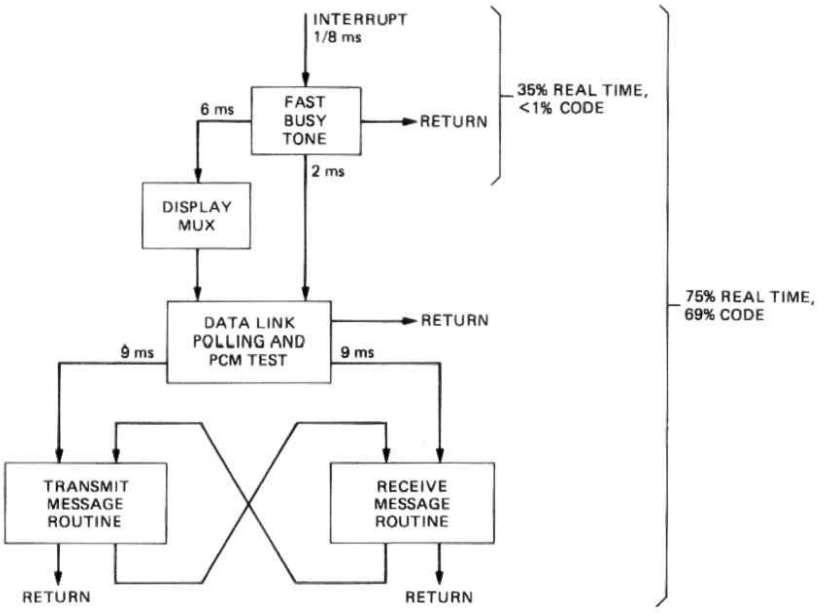


Fig. 6—COT TAU interrupt routines.

messages communicate change-in-state information. Provision is made for updating of assignments and activity as required.

The COT transmits a trunk/line assignment (deassignment) by three identical, sequential 11-bit submessages. The RT looks for a two out of three match to respond, thus providing error protection. The RT similarly sends activity as three identical submessages. Update information is needed to periodically assure that the COT and RT TAUs data bases are in agreement. Updates are sent as a header plus the message and its complement. This biases these messages toward getting through correctly, or not getting through at all, a desirable condition for update messages. The COT sends assignment updates whenever it desires and at the request of the RT by an "Assignment Update Request" message. The RT sends activity updates only at the request of the COT by means of an "Activity Update Request" message. A "Looping Test" message is a periodic message initiated by the COT to test the continuity of the data link. A "No Alarm" message is sent by the RT periodically as a fail-safe way of sending an alarm message to the COT. Care had to be taken in selecting the code words for the messages to assure that message boundaries could be determined, and also to provide error protection across message boundaries since all messages butt end-to-end.

4.2.4 Activity filtering

Activity filtering is provided in both the COT and RT, in addition to the rudimentary filtering provided by the TSI chip. Two bit-up/down counters are employed as filters using unequal attack-decay, with variable thresholds providing hysteresis. These filters provide noise immunity and delay for bridging over dial pulsing, switch-hook flashes, and silent intervals of ringing so that trunks are not deassigned and reassigned during these intervals. These filters negate the need for analog filters on the channel units to perform this function.

4.2.5 Processing trunk requests

The COT controls trunk (busy trunk) assignments and deassignments based on the COT and RT activity. As mentioned earlier, a connection test is done when a line is assigned to a trunk, whereby PCM test codes are circulated COT to RT to COT prior to cut-through of the line. If no trunks are available, the line is assigned to a busy trunk at the COT only and the line then receives the digitally generated fast-busy tone. In the case of a blocked call caused by activity from the RT, that call is transferred to a trunk when it becomes available. Thus, in this case, an RT customer would experience delayed dial tone.

4.2.6 Consistency checks

To assure that all pieces of the data base relating to line and trunk status are correctly correlated, consistency routines have necessarily been implemented. If a conflict arises (for example, a line is assigned to two trunks), corrective action is taken. Such conflicts only arise because of glitches or memory faults, but must be guarded against. In fact, detecting and reacting to "soft" and "hard" errors was one of the most challenging aspects of the software work.

Another related aspect was the requirement that the processor be able to recover from an arbitrary R/W memory state, because it is a stand-alone computer. This required careful consideration and thorough testing to determine that, for instance, the processor would not "hang" if a flag bit accidentally flipped. The ability to detect and react sanely to a genuine fault is also related to these problems. Verifying that software works correctly under the above-mentioned conditions is difficult. The TAU programs were tested by observing the reaction to random data in the processor RAM, and also by forcing bit faults by means of special hardware. These tests were excellent in pointing up several software bugs.

4.2.7 Responding to circuit faults

The TAU software is designed to be very tolerant of R/W memory faults because approximately 65 percent of the R/W memory that is

used (including TSI) is dedicated to per-line functions. Being able to isolate a fault to a single line allows the system to continue operation with a minor alarm condition. The effect is equivalent to a reduced "system crash" failure rate for the TAU plug-in. It is estimated that the equivalent TAU lifetime will be increased by approximately 40 percent because of the fault-responding software. That is, this software would often allow the unit to be replaced before it caused a system crash. This software represents only about a 10 percent overhead in code (cf. Table I).

4.2.8 Identifying channel-unit types

Another software function is processing the "TNEN" information. By associating two TNEN bits, the type of channel unit plugged into a particular physical slot can be determined. The transmit TSI picks up this information and stores it in its own memory from which the processor can obtain it. The result of the processing is essentially two masks. One mask, when combined with the A and B signaling bit activity, generates permanent activity. The other mask forces no activity. Thus a special-service unit plugged into the correct physical slot can be given permanent service or, if plugged into an illegal slot, can be denied service and the condition alarmed. The channel-unit information is also used to condition traffic calculations, since traffic (in CCS) applies only to concentrated trunks.

4.2.9 Measuring traffic and blocked calls

As mentioned previously, a function of the COT software is to calculate and store information related to traffic and blocked calls. A basic software consideration is that some of this information is held for very long periods. The faceplate traffic alarm is based on two or more blocked calls for two out of three weeks running. The traffic and blocked calls displayed on the faceplate are stored indefinitely. Thus, it was necessary to provide storage protection for these pieces of information. For simplicity, the approach used was to triple-store the data and recover them by a two out of three match. This includes not only data, but also the long-term software timers.

4.2.10 Alarm filters

The TAU has the ability to output both minor and major system alarms and to light alarm LEDs on its faceplate. To control these alarm outputs, the software maintains several alarm filters. These filters are up/down counters with a natural decay (down count) built in. To maintain an alarm condition, the appropriate filter must be incremented periodically or fail to be incremented, depending on its use. These alarm filters are maintained for various purposes. For example, one filter checks that the interrupt routines are periodically serviced.

4.2.11 Diagnostic and initialization routines

Finally, the software also includes necessary diagnostic routines. A ROM checksum and a processor maze test are performed continuously. A RAM test is also done continuously, but without slowing down call processing or missing activity. This is done by testing one byte of RAM at a time from an interrupt routine. A few bytes of RAM that are directly involved in the highest-speed interrupt are tested by an indirect method. As alluded to earlier, methods are also used to detect that the processor periodically passes through various portions of the interrupt routines. A sanity monostable is also employed and is strobed on each cycle of the main routine. Power-up initialization routines are provided based on duplicated bytes that are set in the transmit TSI after a power-up or a power supply glitch. Initialization of the data base is not done based solely on a sanity monostable timeout or processor reset since this could be caused by a glitch, which would not be a reason to take down all line/trunk connections.

V. SELECTED IMPLEMENTATION DETAILS

5.1 The time-slot interchange chip

5.1.1 Organization of the TSI RAM

The NMOS TSI chip uses a custom-designed RAM in conjunction with polycells (standard catalog gate functions) for the register and control logic. Figure 7 shows a functional block diagram of the RAM. The operation of the RAM is slaved to a Master Clock (MC) input signal, as is nearly all on-chip circuitry. Address, input data, and output data are all latched internal to the RAM. Separate data input and data output busses exist. The memory is broken into three sections of 49 bytes each. The upper two address bits are decoded to select one of the three sections. Since four possible combinations of the two address leads exist, the fourth combination, not needed for addressing a memory section, is used for addressing the data-link register (Register 7, Fig. 5), activity/TNEN control register (Register 8), and two test bits, thus making these isolated circuits appear as part of the RAM memory space.

The bottom six address leads are decoded to select one of the 49 byte locations within a memory section. To avoid unnecessary transistors and a resulting slowdown of memory operation, a full decoding of the six bits was not done. Valid addresses for the six least-significant bits are decimal 0 to 48 and 63. Addresses 0 to 47 access the 48 bytes used for PCM storage in Sections 1 and 2 of the memory and access the 48 bytes used for trunk and busy-trunk assignments in Section 3. Address 48 or 63 selects the remaining byte. This byte is an all-zero ROM byte in Sections 1 and 2 and a read/write byte in Section 3. The ROM bytes are used to send all-ones (by an inversion) on the T1 line

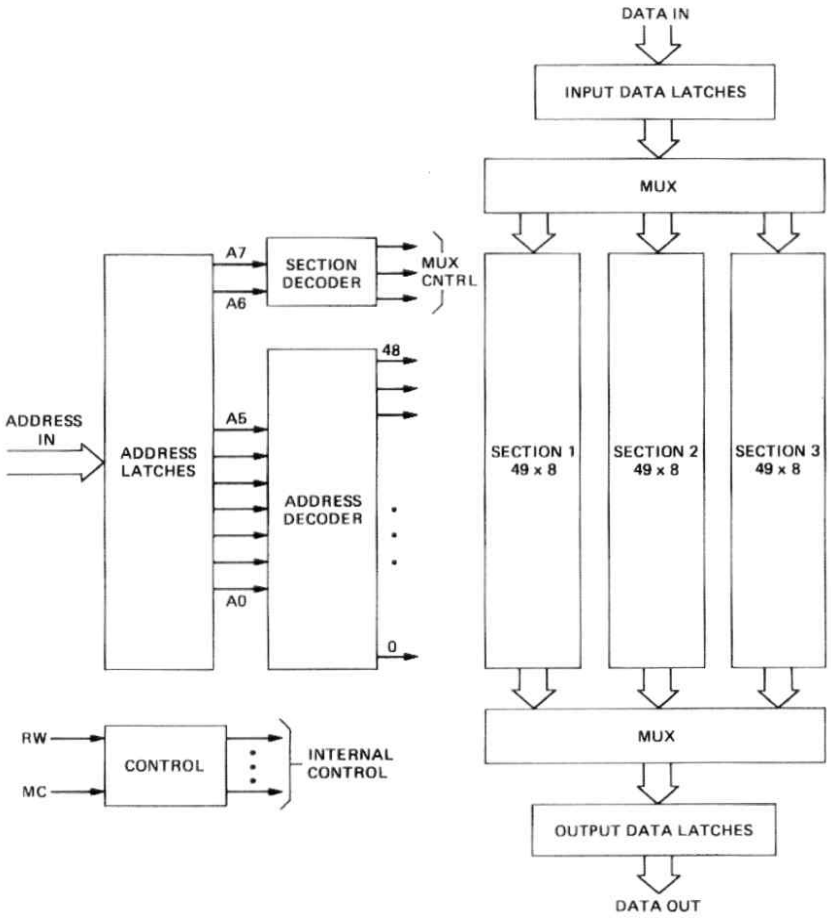


Fig. 7—Block diagram of TSI RAM.

for idle trunks. Unassigned trunks are written to "line" number 48 or 63 which, when fed back, will address one of the ROM bytes. The 49th byte in Section 3 is used as a buffer store for the fast-busy tone bytes written to the TSI by the microcomputer. Since the tone byte must be stable during an entire frame, a buffer is needed to allow the microcomputer the flexibility of writing the next tone byte any time during a frame.

The memory presents a timing interface for the remaining circuitry on the TSI chip (see Fig. 8). For a read or write operation, the address, read/write signal, and input data must be stable at the end of the first half cycle of Master Clock (MC). The memory latches the address, read/write, and input data on the rising edge of MC. For a write

operation, the write will be completed by the end of the MC cycle (Fig. 8). On a read operation, a peculiarity of the memory is that the data appears in the next MC cycle after the cycle that commanded the read. The data becomes valid some time after the start of the cycle and remains valid for the rest of the cycle. Each cycle of MC can be used to perform either a read or write. The TSI design always mandates one or the other. In memory cycles when no useful function is needed, a memory read is performed but the data is not accepted by any of the registers attached to the memory output data bus.

5.1.2 TSI memory layout

For programming purposes, the TSI can be treated simply as a block of memory. Some memory bytes are used for different purposes depending on whether the TSI is functioning as a transmitter or receiver. The 24-trunk assignments are stored in the bottom six bits of the even-addressed bytes in Section 3 of the RAM. The six bits are set to a number between 0 and 47 to indicate which of the 48 lines is assigned to that trunk. If no line is assigned, either 48 or 63 can be written into the six bits. The upper two bits in each of the trunk-assignment locations are called enable bits and operate independent of the trunk assignment. Taken together, there are 48 enable bits (two in each trunk location). Each bit corresponds to a particular line and, if cleared, inhibits PCM read-in to the PCM data locations if the TSI is a transmitter, or forces idle PCM data out of the TSI, regardless of the PCM data memory contents, if the TSI is a receiver.

For receive TSI operation at the COT, the odd-address locations in Section 3 contain the busy trunk assignments. Again, as with the trunks, each location is written to a number between 0 and 47 to assign a line to a busy trunk, or written to 48 or 63 if the busy trunk is not assigned. The upper two bits of the busy trunks have no special use. If

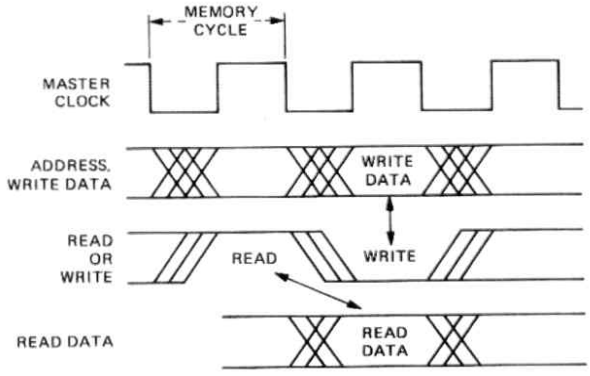


Fig. 8—TSI memory operation.

the TSI is operating as a transmitter, then there are no busy trunks, but some of the same locations are used for holding activity and TNEN bits. Each of the activity or TNEN bytes holds eight data bits, one for each of the eight lines. Thus there are six bytes of activity and six bytes of TNEN to hold data for the 48 lines.

5.1.3 Internal control

The control logic on the TSI chip regulates the flow of data between registers and memory within the TSI chip, and the selection of times appropriate for microcomputer accesses. The main element of this control is a 12-frame counter. By assigning states of this counter to various functions, control of the chip is achieved.

5.1.3.1 Trunk assignments. To understand the assignment of states to specific functions, first consider the steps that must be performed to accomplish the basic time-slot interchange function for a TSI operating in the transmit mode. Serial unconcentrated data is coming in from master and slave TRUs and is being converted to eight bit parallel data in Registers 1 and 2 (see Fig. 5). Every eight cycles of MC (1.544 MHz Master Clock), two PCM words must be written to the memory, one from Register 1 and one from Register 2. Similarly, every eight cycles, one PCM word must be read from the memory and written to Register 3 for transmittal on a T1 trunk. (As noted earlier, the trunk data is read from one section of the PCM memory while the unconcentrated data is being written into the other section.) To read out the trunk data, however, the line assignment for that trunk must first be read out to Register 4, to provide the address for the line data to be read out. Thus, these basic operations consume four out of every eight MC cycles. For the receive TSI operation the trunk data comes into Register 1 and exists unconcentrated through Registers 2 and 3, similarly requiring four cycles.

5.1.3.2 Busy trunk assignments. In the receive TSI operation at the COT where busy trunks are needed, two additional cycles out of every eight are used, one for reading the busy trunk assignment to Register 4 and one for writing the current fast-busy tone byte from holding Register 5 to the PCM line memory of the addressed line. The fast-busy byte is then read out to Register 2 or 3 via the normal sequential readout. The current fast-busy tone byte is read from the fast-busy buffer location in memory to holding Register 5 during the frame bit state (193rd count of the control counter). Thus the receive TSI at the COT uses a total of six out of every eight cycles plus the one extra state occurring each frame.

5.1.3.3 Activity/TNEN. In the transmit TSI mode, states other than the four basic states are used for transferring activity and TNEN information from/to Register 5. These states occur once every 32

states rather than once every 8 states, because it takes that long to collect a byte of activity or TNEN data.

5.1.3.4 Microcomputer access. In all modes of operation, no more than six out of the eight states are used for internal operations; the other two states are reserved for potential use by the microcomputer to perform a read or write of the memory. Waiting for an available state gives rise to the wait states encountered by the microcomputer on a TSI access. The two internally unused states out of every eight are evenly spaced to minimize the delay seen by the microcomputer.

5.1.4 System synchronization

5.1.4.1 ***Transmit TSI mode.*** In the transmit mode the TSI chip establishes its own reference based on its internal counter state. It then forms a super-frame system synchronization signal for sending to the TRUs (a unique pattern over 12 frames). The signal consists of two pulses, one in frame 2 and one in frame 12. Each pulse must be only 162 ns wide to synchronize the 6.176 MHz count-down circuits in the TRUs (6.176 MHz is the fundamental crystal-controlled clock in the SLC-96 system from which the T1 rate is derived). The TRUs and the TAU receive the same clock and then the TAU must carefully time the pulse with respect to the clock edges. Because of the short delays involved here, the pulse had to be timed with TTL external to the custom NMOS chip. The TSI chip puts out a pulse that spans two 6.176 MHz periods which is then gated externally to produce the desired pulse. The effect of the pulses that are sent to the TRUs is to synchronize the counters in the TRUs.

As the slave digroup PCM enters the TSI, it is delayed by one clock period by means of an FF (see Fig. 5) so that it arrives one MC cycle later than the master PCM. The outgoing trunk PCM is also delayed one cycle by means of an FF, which exists for matching outgoing PCM streams in the receive mode. The frame bit from the master digroup is picked off and saved in the FR FF for re-insertion into the outgoing trunk PCM bit stream (which is why this stream is called the master).

5.1.4.2 Receive TSI mode. Synchronization in the receive mode is very different from synchronization in the transmit mode, because the timing is controlled by the reframing circuit contained in the TRU. This circuit locks on the frame bit in the received T1 bit stream and thus can provide the timing reference and reframe signals needed by the receive TSI in the TAU.

The timing reference is provided in the form of a 4 KHz (two frame) clock signal from the master digroup TRU. The edge of this clock waveform is used to trigger the TSI control counter to a predetermined state chosen to synchronize the TSI to the incoming bit stream from the T1 line interface unit. When calculating the position of this bit

stream relative to the synchronization signal provided by the TRU, the delay in passing through the TSI must be taken into account.

Reframing is a very important consideration for receive TSI operation. Since the reframing process occurs downstream from the TSI, the TSI must assume a special mode of operation during the reframe process. This mode must pass the concentrated trunk bit stream directly through the TSI unchanged to assure that the frame bit is received by the TRUs (the timeslot interchange function can lose the frame bit if the TSI chip is out of sync). Once the TRUs reframe, the receive TSI can resume normal operation. To assure that the TRU is not thrown out-of-frame when the receive TSI returns to its normal sequencing after a reframe, it is necessary that the relative position of the frame bit not change. That is, during the time that the TSI is passing the received concentrated bit stream directly through, it must insert the same number of cycles of delay as it will when it is operating normally. Also note that the scrambled time slots sent to the channel units during reframe are of no consequence because the channel units are not enabled to receive the information.

A further consideration during the reframe process is that the 4 KHz synchronization signal slips as the TRU searches for the frame bit. Thus this signal cannot be allowed to preset the TSI control counter during this time or confusion can result. The TRU provides an out-of-frame signal that can be used to gate the synchronization clock and also force the TSI chip into its special reframe mode of operation. Once the out-of-frame signal indicates the TRU is reframed, the synchronization clock is allowed to once again force the counter state in the receive TSI.

During the reframe mode the TSI performs as follows. All incoming trunk data, instead of being selectively written to the assigned line memory locations, is sequentially written to all the slave PCM memory locations in the same section of PCM memory as the selective writes would have been done. The sequential readout of each line's data in the alternate section of PCM memory continues as normal, except that the slave data is also forced out to the master digroup. In this way the concentrated bit stream is passed directly through the TSI with the same delay experienced by the bit streams in the normal mode of operation. Note that if the receive TSI is out of synchronization (as it is assumed to be if a reframe is needed), then the frame bit is not being put into the frame flip-flop (FR FF). Instead, an arbitrary bit out of the frame sequence is being inserted in the FR FF, depending on the relative state of the TSI control counter. The actual frame bit, then, is being stored somewhere in the PCM memory. The bit stream passes unchanged through the TSI, however, with the bit stream being put back together correctly as it exists from the TSI.

After the reframe process terminates, the control counter is resynchronized as mentioned above. Since the frame bit was arbitrarily located in memory, the resynchronization will, in general, cause the loss of one frame bit. The lost frame bit will be in error with 50 percent probability assuming a random bit is inserted in its place. By selecting the rising edge of the synchronization clock, the frame bit that is lost is a signaling frame bit. Thus there is no concern, during the resynchronization, about approaching the master frame bit error threshold that would throw the TRU back out-of-frame. A signaling frame bit error will, at most, delay the reframing of the signaling bit extraction circuit in the TRU.

One other step must be taken in the TSI to complete the reframing process. Since the slave PCM memory locations were used to hold data while the TSI was passing the concentrated bit stream directly through, it is desirable to clean out this memory, that is, write these memory locations to idle PCM code. This is done automatically by the hardware during the two frames that follow the control counter resynchronization (two frames are needed to initialize both sections of PCM memory).

The TSI adds approximately four frames of time to the reframe process for PCM data. This is an increase of 0.5 ms to an average reframe time of 25 ms. The TSI may also add up to one super frame of delay to the receipt of A and B signaling bits by the channel units because of the erroneous signaling frame bit. This is equivalent, worst case, to the loss of one additional A and B signaling bit.

5.1.5 Chip development

The development of the Time-Slot Interchange (TSI) chip was justified on the basis of cost, power, and space. Further the chip is a universal design that can function as a transmitter or receiver, at COT or RT, so that only a single custom design is required. The breadboard for the TSI was built using 96 off-the-shelf integrated circuits. The chip is realized with NMOS polycells and a custom NMOS RAM using 5 micron rules on a chip 258 × 367 mils, packaged in a 40-pin DIP. Typical power dissipation is 750 mW. The design cycle from the first paper design to first chips took about two years. The chip contains 432 polycells, 145 bytes of static RAM, and two bytes of ROM. There are about 10,000 transistors, 70 percent of which are used in the RAM. Extensive logic and timing simulations were required to verify the design.

5.1.6 Internal timing considerations

The TSI clock frequency of 1.544 MHz mandated careful consideration of timing delays in the design of subcircuits for the TSI. "Regular power" NMOS polycell gates can give delays of approximately 50 ns. "High-power" and "super-power" gates must be used for shorter delay

times with corresponding increases in power dissipation. Certain critical paths in the TSI did require the use of high-power and super-power gates.

This method of meeting timing constraints, however, was only supplementary to the decision to build a highly synchronous design. By clocking nearly all subcircuits directly with MC (Master Clock), a very clean timing plan was developed. That is, all propagation delays, setup times, and hold times could easily be calculated with respect to an MC edge. Memory operations were also specified with respect to MC (previously discussed). Such a design avoids accumulating long strings of delays that can cause problems at the expense of, in some cases, additional gates. For example, all the registers have their FFs directly clocked by MC. Gating around and between FFs determine the function the register performs such as hold, shift, or load. With this type of design, where all action takes place on a clock edge, the remaining portion of a cycle can be used for propagation delays of the signals that determine what function will be performed when the next clock edge arrives. This design technique is very valuable for high-speed designs (high speed relative to the technology limitations).

Another timing consideration that should be mentioned is the expected clock duty cycle variations in the receive TSI mode. MC in the receive mode is the recovered T1 line clock. As such, duty cycle variations can be expected. Final assumptions were for a 60/40 duty cycle worst case in either direction. This implies a need for the TSI chip to work, equivalently, at a higher frequency (viz. 40/40). Timing must apply for worst-case device, power supply, and temperature variations, as well, leading to the test-clock rate at room temperature of 2.5 MHz.

5.2 COT software details

5.2.1 Real-time constraints

A basic problem in structuring the software was to determine a method for handling the real-time constraints. All the interrupt routines run indirectly off the fast-busy tone interrupt which occurs once every eighth of a millisecond (recall Fig. 6). The message routines can take milliseconds to execute, however. Thus it was necessary to enable the interrupt upon leaving the Fast-Busy Tone routine to enter the other interrupt routines. The result is that several levels of interrupt can exist on the stack. The MAC-8 microprocessor automatically saves the condition register and return address on the stack when an interrupt occurs, and thus conveniently allows nesting of interrupts. Since the Main routine may have been nested in subroutines when it was interrupted and the interrupt routines, themselves, may have been nested in subroutines at the time of another interrupt, the stack must be large enough to hold all the return addresses. A few data values are

also sometimes stored on the stack. Fifty bytes was determined to be adequate by considering worst-case nesting levels and including margin for unexpected levels caused by an interrupt glitch or an accidental bit flip (a subroutine call stores two bytes on the stack, an interrupt three bytes).

5.2.2 Time-share processing

To get a better feeling for how the processing gets time-shared, refer to Fig. 9. This time diagram is an example of the routines that might be processed in response to a sequence of interrupts. The Fast-Busy routine is executed after every interrupt, followed by a return to the routine which had been executing. This time line starts out by assuming the processor is in the Main routine when an interrupt occurs. The interrupt causes the Fast-Busy routine to be executed. After 16 executions of the Fast-Busy routine, a 2 ms timeout occurs causing the Poll routine to be entered (recall Fig. 6). The Poll routine can be interrupted several times, before it finishes, by the Fast-Busy routine. After the Poll routine finishes, it is assumed that the Transmit Message routine needs to be executed. This routine runs for milliseconds and so will be interrupted not only by the Fast-Busy routine but also by the Poll routine. After the Transmit Message routine terminates, it is assumed necessary to process the Receive Message routine and so control switches to it. Finally, the Receive Message routine finishes and control returns to the Main routine.

The Fast-Busy, Poll, and Display Mux routines are fast enough to finish before they would be called upon to re-execute. However, they would not cause serious problems even if they were re-entered because of some glitch. The message routines are much more complex, however, and can cause some serious consequences (confusing trunk assignments, for example) if they are re-entered. Normally these routines would be finished before being called upon again (as necessary to meet the demands of the data links for messages every 9 ms). However, to provide a degree of protection from the havoc that could occur if they were re-entered, a flag is maintained that indicates the message routine needs processing or is in the process of being executed. This flag is set by the Poll routine when it determines that the message routine needs processing, and is cleared at the end of the message routine after all processing is completed. These flags also provide the means by which control can be transferred directly from the Transmit Message routine to the Receive Message routine or vice versa (recall Fig. 6).

5.2.3 Data-link polling

The data link provided the TAU by the SLC-96 Data Link Unit (DLU) consists of 11 bit packets of serial data every 9 ms. The TAU is provided with a frame signal that remains high for 2.75 ms and low for

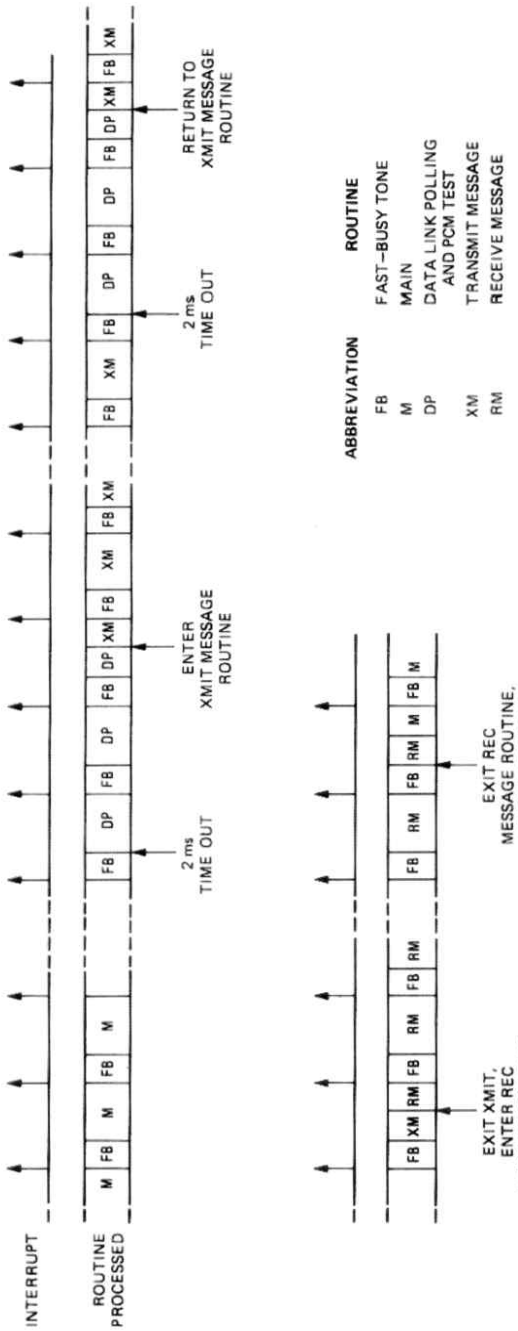


Fig. 9—An example of a routine processing sequence.

6.25 ms. During the high portion of the frame, serial data is transmitted or received at a 4 KHz rate (1 bit every 0.25 millisecond), by combining a 4 KHz clock provided by the DLU with the frame signal. As Fig. 10 indicates, the time while the frame signal is low is available for loading or unloading, in parallel, the shift register provided in the TSI chip. Figure 10 also indicates that the frame signal is polled every 2 ms. The polling is asynchronous and so only one possible phasing of the polling with respect to the frame signal is shown. The polling scheme works by noting 1 to 0 transitions of the frame signal and using this event as a trigger for processing the message routine and loading (unloading) the shift register. Both the Transmit Message routine and the Receive Message routine write or read, respectively, the shift register immediately upon entry; thus, if the routine is delayed or takes a while to process, it is assured that the data-link shift register is loaded or unloaded prior to the next rise in the frame signal.

The actual data-link polling is complicated by the fact that there are two data-link frame signals, one for transmit and one for receive. The data links are asynchronous with respect to each other (or at least out of phase with any random phase), and so all relative phasings must be considered. The message routines are designed to run to completion before transferring control to the other message routine, if needed. Therefore, a message routine can be delayed and real-time constraints must be considered. Since a request to process both message routines may occur at the same time, a priority of processing had to be established.

Figure 11 shows two possible phasings of the data-link frame signals and the processing sequences that could correspondingly result (the Receive Message routine is given priority). Note that message routine

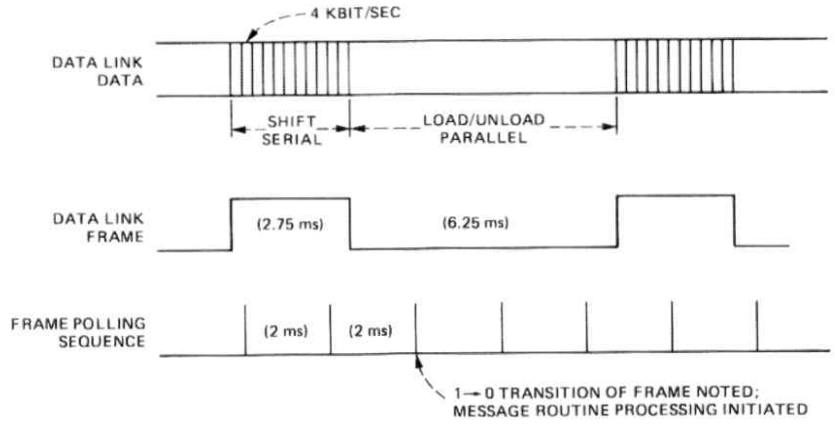
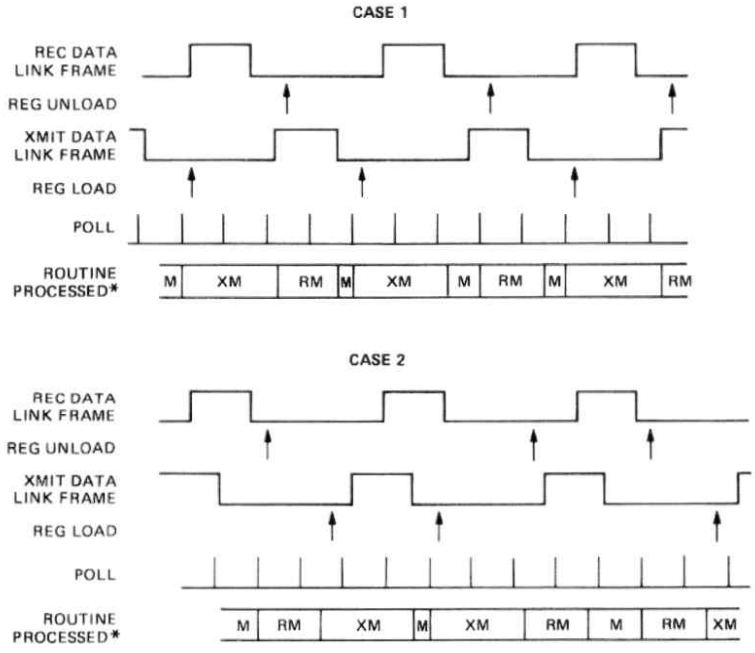


Fig. 10—Data-link message processing.



*NOTES:

1. M = MAIN ROUTINE
   RM = RECEIVE MESSAGE ROUTINE (3 ms RUN TIME)
   XM = TRANSMIT MESSAGE ROUTINE (4.5 ms RUN TIME)
2. RM HAS PRIORITY OVER XM
3. LOW LEVEL INTERRUPTS IGNORED IN REPRESENTATION

Fig. 11—Data-link message sequencing.

processing need not alternate; instead, the Transmit Message routine may be processed twice in a row followed by the Receive Message routine being processed twice in a row as the result of 2-ms polling of the 9-ms frame interval.

If both frame signals are aligned and the polling was such that the transition from 1 to 0 is not noted until almost 2 ms after it occurs, then the Receive Message routine is constrained to be less than 4.25 ms. Enough time must be allowed to enter the Transmit Message routine and put out the new submessage, prior to the rise of the transmit data-link frame signal.

Since only a single flag is used with each message routine to indicate that the routine needs processing or is in processing, and since requests for processing can occur 8 ms apart, the sum of the execution time of the Transmit and Receive Message routines is constrained to less than 8 ms. If, under some unusual circumstance, the processing takes more

than 8 ms, the result is that the next occurrence of the Transmit or Receive Message routine is not processed and a data-link submessage is therefore lost. But because of the redundancy and error protection built into the data-link messages, a message will probably not be lost.

5.2.4 Time-critical routines

The most time-critical routine is the Fast-Busy routine. Great care was taken to make this routine as fast as possible; two MAC-8 registers are used, one for an auto-increment pointer to the fast-busy word data table (register b4) and one as a temporary holding register for the tone byte (register a5), since a memory-to-memory transfer instruction does not exist. (In MAC-8 assembly code, "b" registers are 16-bit general-purpose registers and "a" registers are 8-bit general-purpose registers.) For the purpose of saving bytes and improving speed throughout the entire TAU program, two eight-bit registers (a3 and a13) are used for flags. One of these flags is used for the silent interval of the fast-busy tone (the tone is pulsed on for a 0.25 s, off for a 0.25 s).

Table IV gives the cycle count on an immediate return path of the Fast-Busy routine. The variable number of cycles for the write instruction is due to the variable number of wait states for a TSI access. Assuming that 67 cycles of each frame (out of the available 193) are used for the Fast-Busy routine implies that 35 percent of the real time is used in this routine.

A 2-ms software timer is derived from the data table pointer (register b4) in the fastest possible way by a test on bit 4. For this scheme to work, the Poll routine, when it is entered, immediately increments the pointer by 16 so that only a single 2-ms timeout is indicated by the bit test on the pointer. Thus the fast-busy code table of 48 bytes is actually stored in ROM as three tables of 16 bytes, separated in address space by 16 locations. Worst case, the data table pointer must be set to its new value in less than one frame of time from the interrupt, to assure it has the correct value when the next interrupt occurs. The values for the fast-busy code words were derived by calculating linear samples of the tone and then using a translation table to companded PCM codes. The least significant bit was forced to always be one as needed for A

Table IV—Listing 1: Fast-Busy routine code fragment

Cycles	Instruction	Comment
10	(interrupt preamble)	
7	if (bit(7, a3)) goto ia1;	/* jump if silent interval */
15	a5 = *b4++;	/* fetch busy word */
16-22	BSYWD = a5;	/* write word to rec TSI */
7	if (bit(4, a4)) goto ia2;	/* check polling timer */
9	ireturn( );	
64-70	(Total cycles)	

and B bit signaling to the channel units. The code inverse actually gets sent to the channel units because of a data inversion in passing through the TSI.

The Poll routine is less critical than the Fast-Busy routine by a factor of 16, but it is still important to consider worst-case paths. The routine is held to a minimum, again, by careful use of register instructions and arranging the conditional branches for minimum worst-case path. The execution time of the Transmit and Receive Message routines must also be considered, because they need to be fast enough to service the data-link requests upon demand, to avoid further complication of the algorithms. An example of what can be done to minimize worst-case timing paths is the execution of the consistency checks and updates in the Transmit Message routine on the passes through this routine that do not require the calculation of a new transmit data-link message. That is, since a message is composed of three 11-bit submessages which are calculated and saved until needed, no new message calculation is required while the first two of these submessages are being sent, and so there is real time available for the consistency and update routines to use.

The Main routine has no significant real-time constraints; however, it is still necessary to know the approximate cycle time of the routine to allow an adequate timeout for the sanity monostable. The cycle time of the Main routine is, of course, strongly affected by the running time of the interrupt routines. (It is necessary to strobe the sanity monostable from the Main routine, rather than an interrupt routine, since the latter might continue to strobe the monostable while allowing a return to an unknown loop at an arbitrary location, rather than to the Main routine.)

5.2.5 Line/trunk data base

The processor RAM is used for several purposes as was given in Table III. The line/trunk data base stores copies of the 24 trunk and 24 busy-trunk assignments. These copies are used during assignment searches and acted upon by the consistency routines. The transmit and receive TSI assignments are updated from this copy. These copies are maintained because they can be accessed more quickly than the information in the TSIs.

The line/trunk data base also holds information concerning the 48 lines the concentrator serves. This information is split into six data groups, each group holding information for eight lines. The information is stored in bitwise correlation with the eight bits of line activity as collected in a single byte by the transmit TSI. For each group, seven bytes are stored. Bytes 0 and 1 are the TNEN masks for forcing activity or no activity. Bytes 2 and 3 are the activity filters, stored as least-

significant bits and most-significant bits of two bit up/down counters. Byte 4 stores the status of activity from the RT as received over the data link. The trunk and busy-trunk status in bytes 5 and 6 indicate whether a line is assigned to a trunk or busy trunk, respectively. These bits are maintained to provide a fast method of determining whether an assignment (deassignment) is needed. Without them, one would have to scan all the trunk assignments for every line that had activity to determine which line needed assigning. This could not be done within the time constraints of generating a data-link message "upon demand." All the bytes of data in each group are arranged so that all eight lines can be processed simultaneously by byte operations, thus performing activity filtering, TNEN masking, and service request determination very quickly.

5.2.6 Handling faulty lines and trunks

A significant feature of the software is the ability to allow concentrator operation in the presence of partial faults. Before assignment to either a trunk or busy trunk, extensive local memory checks are made on the trunk (busy trunk) assignment, enable bit, PCM data, and status bit memory locations. A failure causes an alarm to be raised and possibly a line/trunk fault to be stored. If local tests pass, then the PCM looping test is set up for a trunk assignment. A failure of this test can also cause a fault to be stored.

The routines that are provided allow for the detection of memory or connect (disconnect) failures at the time of line/trunk assignment (deassignment). If a particular line/trunk combination fails, that combination is put in a fault store and periodically retried. One such fault gives rise to a minor alarm, two such faults shut down the system. The question is always raised, why not try to assign the line to a different trunk if the first one fails? This is not as easy as it sounds. The basic problem is one of fault isolation. The line/trunk *combination* is fundamental in finding the fault; splitting them up could easily cause a loss in the ability to refind and retest the fault, to maintain an alarm. The result could easily be intermittent alarms or a faulty trunk that wanders from line to line, possibly causing random customer complaints. In any case, it would require a lot more software with diminishing returns. It is also true that, for a given line/trunk combination fault, hardware considerations give a higher probability to the line being at fault than the trunk. In short, it seems very acceptable, and is fairly straightforward in software, to keep the line/trunk combination as a means for maintaining an alarm while allowing all other customers normal service.

A fault consists of a trunk and line pair. The stored trunk number is the lower byte of the address to the trunk or busy-trunk assignments

in the processor RAM. A line number is stored as 0 to 47. If the fault was trapped during a deassignment, the "line" number may be 48 or 63. Both 48 and 63 are used because TSI will allow either one as a no assignment number. If memory fails and will not allow deassignment to 63, then 48 is tried. An empty fault store is designated by all ones for the line number and all zeroes for the trunk number.

If a fault is to be retried, the trunk and line numbers are checked for validity and then a jump to the appropriate trunk (busy trunk) assignment (deassignment) algorithm is made. Also, to assure that the fault is not lost because of changes in the faulted line's activity, permanent activity is maintained by setting the activity filter for that line to a count of 3. A fault will be retried approximately every 1.8 s and thereby continue to increment an alarm filter if the fault persists.

5.2.7 PCM looping test

For a trunk assignment, the PCM looping test is always executed. First, the test is initiated in the Transmit Message routine. This process consists of clearing the enable bits in transmit and receive TSIs for the line under test, so that the test codes will not be overwritten or sent to the channel units. Also, the first PCM test code (alternating 1's and 0's) is written into the transmit TSI PCM memory locations and a PCM test timer is set for an 80-ms timeout. Since the line is assigned to a trunk at the COT, the test codes are received at the RT as soon as the RT receives the trunk assignment message over the data link. The enable bits at the RT are automatically cleared upon receipt of the assignment message.

The operation of the test at the COT is then picked up by the PCM test portion of the Poll routine. This routine will sample the receive TSI every 2 ms looking for the PCM test code that should be returned by the RT. When the code is received, it is complimented and sent to the RT. When the complimented code is received at the COT, the COT sends a test termination code (all 1's) to the RT for 10 ms and then sets the COT TSI enable bits, thereby cutting through the customer at the COT. The RT will correspondingly set the enable bits at the RT when it receives the test termination code.

If a timeout occurs, the COT will deassign the trunk and store the line/trunk combination for later retry. If the RT fails to see the test termination code after a timeout from receipt of the assignment message, it will simply deassign the trunk (only the COT records the trunk and line that gave rise to a fault). If a trunk (busy trunk) deassignment is requested, the disconnect is performed in a straightforward manner. Some memory tests are performed and may result in the storing of a fault. This fault may involve only a trunk if the problem is in writing the unassigned "line number."

VI. PERFORMANCE TESTING

Performance testing of the TAU was a very big part of the project. When all the hardware and software effort expended in testing is considered, the testing job was nearly as big as the basic job of designing the TAU circuits, custom chip, and software. Because of the complexity and compactness of the design, thorough and sophisticated testing was essential.

6.1 Early-project testing

The TAU project began with the design of the TSI. TSI breadboards were built and debugged using a scope and logic analyzer. Because the TSIs use a handshake arrangement in talking to the microcomputer bus, manual switches can be used to simulate the bus signals, and thus a working microcomputer was not needed at this stage. Later, after the TSI design was well on its way to being realized as a chip, the microcomputer designs were finalized and built. The microcomputers were debugged using simple programs and a logic analyzer.

6.2 Mid-project testing

Once serious programming began, a versatile test set was needed in addition to the MAC-8 development system (PLAID). A MAC-8-based test set was designed that allows the display of all 24 trunk assignments on numeric displays by examining the data link messages that flow from COT TAU to RT TAU. It also employs A and B bit and TNEN bit generator cards that store information for all 48 lines that the concentrator is working with and allows the operator to manually set these bits to simulate channel units. It also displays the received A and B bits for the selected channels.

This test set was invaluable for tracing bugs in the TAU software as it developed. Some of the intermittent and transitory phenomena was especially visible on the displays. It was also very nice for observing the results of simulated memory faults, which was done with another piece of test hardware, consisting mainly of EPROMs with the selected bits to be faulted marked in the EPROM. Most of the software for the TAU was written and debugged using this MAC-8-test set in combination with the PLAID.

Another capability that was developed later and was very useful in debugging the code was a data-link monitor. This allows the concentrator data-link messages in both directions to be displayed on a CRT in a correlated fashion. The program allows the storage of messages occurring over approximately 2.3 s and allows triggering on a particular message pattern with "don't care" conditions. The stored messages can then be viewed by scrolling forward and backward. The data-link monitor was especially useful for checking out the software that

determines message priority. It was also useful in noting response messages, such as a trunk assignment message leaving the COT TAU in response to an activity message received from the RT TAU. Another use was in viewing widely-spaced periodic messages such as the data-link "Looping Message."

6.3 Late-project testing

A development that paralleled the TAU and which proved very useful to us for system tests and final TAU software tests was the "Traffic Generating System" (TGS). This development was initiated to simulate realistic traffic on digital lines for the testing of a digital switch. It was decided to develop the hardware and software necessary to use the SLC-96 carrier system for simulating traffic on a T1 line. This hardware and software also serves as a debugging tool for the TAU, while the TAU also provided a shakedown test for the traffic-generating system.

"Signaling interface boards" that perform similar to the original MAC-8 test set are used to simulate channel activity in TGS, that is, A, B, and TNEN bits for all channels are stored in a RAM and read out in the proper sequence. Similarly, received A and B bits are stored. One signaling interface board serves 24 channels. The RAM is writable and readable through an I/O port to a DEC–LSI-11. This hardware is flexible, portable, and used simply by plugging the simulator card into a channel unit position in the SLC-96 system bank. The LSI-11 is tied through a satellite processor link to a host UNIX system (see Fig. 12).

Software for TGS allows writing "scripts" in C programming language that describe what a channel does and when it does it. Library routines are available that simulate, for example, dial pulsing and ringing. Many scripts can be run simultaneously using shared library routine code. The system thus allows simulating realistic traffic conditions.

By using TGS and the data-link monitor, the final TAU boards were exercised very thoroughly. One program that was written measured the connect delay time from an A or B signaling bit change. This program was expanded to make thousands of random calls, measure the delays, and store the results in a UNIX file for later graphing of delay distributions. Most of the delay and distribution of delay is attributable to the delay and asynchronism of the data link. Originating calls are delayed more than terminating calls because of the delay in transmitting RT activity to the COT. Average connect delay from the COT is 70 ms, with 100 ms from the RT.

Delay measurements were repeated with a random-error generator used to insert errors on the T1 lines. At an error rate of 2×10^{-4}, which is worse than a functioning SLC-96 system will see, the only noticeable change in the connect delays was an increase of a few milliseconds in the delay. This checked out the error protection built into the data-link messages.

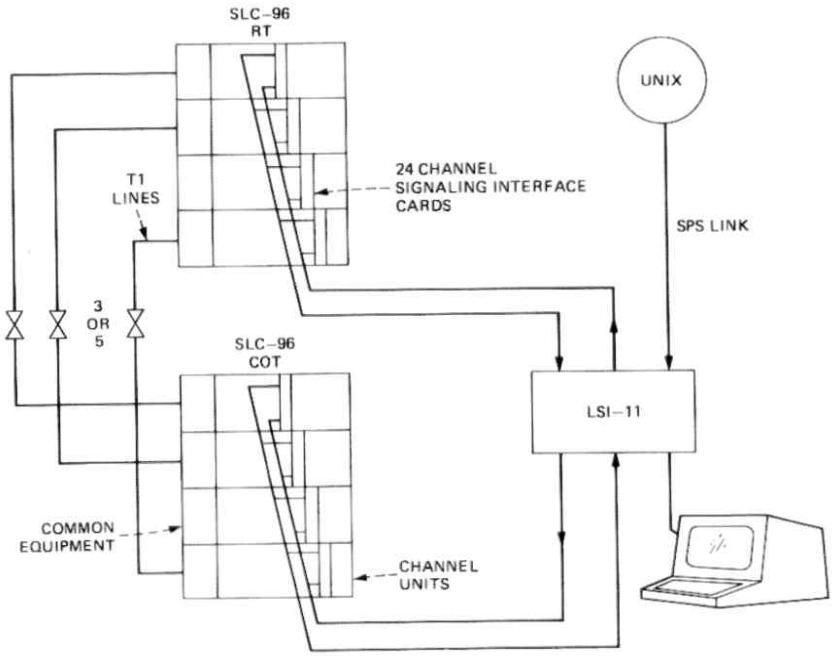


Fig. 12—Traffic generating system for TAU testing.

Another program that was written simulated two simultaneous calls and measured the connect delay of the delayed call. These measurements showed an increase in the connect delay of approximately 50 ms for the second call. Programs were also written to manipulate the TNEN bits, with and without A and B activity, and thereby simulated each type of channel unit in each physical position. Other programs checked blocked-call functioning, the generation of fast-busy tone, and normal ringing and dialing. The traffic generating system was undoubtedly very important in establishing confidence in the final hardware/software design.

VII. CONCLUSION AND ACKNOWLEDGMENTS

The SLC-96 carrier system TAU demonstrates that modern electronics economically provide improved features in the loop plant. The ability to integrate the Time-Slot Interchange function onto a single chip made this development possible. By digital concentration, the TAU reduced the number of T1 lines needed by the SLC-96 system from five to three. Because provision is made for special-service circuits to be given unconcentrated trunks, a separate system is not needed to provide a few special interfaces. Traffic measurement and extensive

maintenance features were also successfully integrated into the firmware control of the TAU.

The author wishes to especially acknowledge Lary Range for performing extensive simulations on the TSI chip and John Beck for his software-design assistance. The layout and design of the TSI chip were directed by Gil Mowery. Sam Arnold, Brian Redman, and Doug Corey were responsible for the software design of TGS. Credit is also due to many other individuals who were associated with all phases of this project.

REFERENCES

1. "The Loop Plant," B.S.T.J., *57*, No. 4 (April 1978).
2. S. Brolin et al., "Inside the New Digital Subscriber Loop System," Bell Laboratories Record, *58*, No. 4 (April 1980).
3. H. D. Rovegno, "A Support Environment for MAC-8 Systems," B.S.T.J., *57*, No. 6, Part 2 (July–August 1978), pp. 2251–63.
4. "UNIX Time-Sharing System," B.S.T.J., *57*, No. 6, Part 2 (July–August 1978).