L-3 COMMUNICATIONS GNS
MAXPRO WIRELESS PRODUCT UPGRADE
PROJECT DESIGN SPECIFICATIONS (DRAFT 1.3)

PREPARED BY :

ART LARSON, MIKE GRUDSKY, PAUL MURRAY, ERIC KRIEG, STEVE BULACK
L-3 Communications Inc.
700 Dresher Road
Horsham PA 19044

TABLE OF CONTENTS

1 Overview

The current L-3 GNS project is to provide an updated product to support communications services from four inbound E1 Truck Communications channels compressed onto the output trunk through a single E1 communications interface. The current project schedule requires the system to be delivered to L-3's client site in the September 1999 timeframe. The issue identified here is to determine the realistic scope of the work and to provide a manageable a task work breakdown to support and assist in the management of the project's progress. To meet schedule the software will be developed using a phased approach.

1.1 Technical Background

In the current project L-3 GCN is upgrading/enhancing the existing Hardware/Software Architecture for the MaxPro Wireless Systems. This product will support a 4:1 data compression for E1 communications thus reducing the customer's telecommunications cost by 75% of what would be required without using uncompressed E1 communications on the same network. An E1 circuit has a bandwidth of 2.048 Mbs (E1 Span). This is divided into 64 Kbs Channels. Thus an individual E1 has 32 channels of 64 Kbs bandwidth available. A single channel is normally used to carry voice/fax/data communications. Sometimes as requested from customers, the entire channel and/or a set of channels can be dedicated for a designated customers (leased line). The MaxPro System supports processing communications through 4 uncompressed E1 trunks or 128 channels and then using a 4:1 compression send/receive this through another E1 trunk using only 32 channels. This system consists of several major components:

- A chassis that utilizes a PCM system bus architecture. The PCM bus supports 16 serial lines communications at a data rate of 2.048 Mhz. The chassis contains 22 backplane board interface connections.
- An IPC bus used to support communications between the various boards in the rack. The system has two IPC channels operating at 2.048 Mhz.
- DSPR Boards are used to receive E1 data streams and to compress/decompress data on specific channels. The legacy system DSPR boards have only a 2:1 Compression ratio and the new MaxPro DSPR-32 board has a 4:1 compression. A signal DSPR-32 board supports management of compression/decompression of 32 channels. This means that only one DSPR-32 board is needed in the chassis to manage the data stream traffic of an entire E1 span on compressed data (32 channels).
- A CPU board used to interface with the external interfaces and manage the MaxPro Wireless System functions. One of the main tasks for this card is to configure the mapping of E1 circuit resources between the 4 Uncompressed E1 boards to the DSPR-32 board which interface to the compressed E1 board. The CPU board processes user commands to setup connections between the compressed E1 channels and the uncompressed E1 channels. This is managed by the CPU board sending messages to the other boards in the system which direct these boards to map and establish connections between theE1 trunk line on the compressed board's channel to that in one of the uncompressed E1 board trunk line channels.
- E1 Transcoder cards these interface with up to four E1 trunk lines of uncompressed and one compressed E1 trunk line link. The E1 Boards in the system are configured to operate with one of the two releases of the software. The boards use identical hardware but the embedded firmware is different and as result they function quite differently. The firmware in the first board type deals with interfacing to trunk lines of uncompressed E1 data stream communications. Here the firmware is used to extract the E1 data and map this onto the PCM bus. The firmware in the later type of E1, is used to extract data compressed via the DSPR-32 boards in the system presented to the E1 board via the PCM bus and to then translate the data onto the E1 lines sending/receiving compressed E1 communications. The software in these boards will be different from that in the current E1 2:1 Transcoder systems. In the new system the uncompressed and compressed E1 boards are to both interface with the DSPR-32. In the uncompressed boards most likely only a review of the code is needed for this purpose. A possible change may be needed in the area where the mapping representation of the E1 channels to the PCM channels in the DSPR-32 may be required. In the compressed E1 the mapping representation of the compressed E1 channels to the DSPR-32 and uncompressed E1 boards will be different. This will result in software changes being required for this area.

1.2 E1 Communications CCITT G.704 Standard [35]

According to CCITT G.704 Standard, E1 Data communications supports a data stream of 2.048 Mbs. This is separated into 8000 Frames per second. Between each frame is a frame alignment signal, (FAS – X0011011), used to align the electronics to process the next data stream. The frame data is further divided into 32 Time Slots (0..31) or Channels. Each Channel processes one byte of data per frame. This scheme supports 32 channels each with a capacity to manage 8000 bytes per second or 64 Kbs. Channel 16 is defined by the standard to support Channel Associated Signals (CAS) and/or Channel Control Signals (CCS). The standard defines support for voice communications over channels 1..15 and 17..31. Two of the channels (Channel 0 and 16) are reserved by the standard for other processing defined in the standard. Channels 1..15 and 17..31 are assigned to telephone numbers and identified as telephone channels 1..30.

1.2.1 PCM Bus [1], [3]

The PCM bus consists of 16 serial lines. A line supports data I/O at 2.048 Mbs. Each line supports thirty two 64 kbs channels. The PCM bus bandwidth represents the ability to support 512 unidirectional channels or 256 bi-directional channels. Data communications between the MaxPro System will occur via the PCM serial bus. Channels are assigned via the CPU card. The 16 lines support Time Dimensioned Multiplexing (TDM) serial streaming consisting of 32 channel frames. The total capacity is 16 x 32 = 512 channels. The bit transmission rate 2.048 Mbs corresponds to 64 Kbs/channel/second or 8000 bytes per channel per second. A TDM time frame of 125 mircoseconds (8000 frames per second) is used for management of the channel data. A single 125 microsecond timeframe supports a data steam of one byte of 8 bits per channel for each of the 32 channels. The Mitel MT8985 Digital Switch chips are used to communicate with the PCM bus. The chips support connections to 8 input serial lines and 8 output serial lines. The Mitels perform management of the data stream for each of these 8 serial input lines, breaking the data stream into 125 microsecond frames each line supports 32 channels of 8 bit wide data streams or 256 bytes of data in a 125 microsecond time frame. The chips are double buffered allowing the boards 125 microseconds of processing time before having to deal with the next frame. An interface between the framers/transceivers is provide via the Mitel chips. 4 Mitels together can provide the ability to connect to and from any of the 512 PCM bus channels. The input and output lines of the Mitels can be interconnected to support loopback for diagnostic testing. In order to support the proper framing of the data streams each board in the chassis has both Mitels and Framer Chips PCM6341 E1 Framer Chips. The data stream for E1 Communications is defined in the CITT G.704 Standard. An E1 supports. 256 bits of data per frame which is allocated into 32 Time Slots or Channels. Between each Frame a Framer code is used to delimit the frame and to allow the electronics to synchronize with the data stream. The job of the framer is to perform this function and to detect error conditions that may occur.

These chips are present in all the boards (CPU, DSPR, E1). The Mitel chips perform time slot interface (TSI) functions and allow the DSPR to receive/transmit data to on the specified time slots. The DSPR is configured with Mitel chips for management of compressed and uncompressed E1 traffic. The chips are used for unidirectional traffic management Network to User E1 traffic and for User to Network traffic and the other two to manage both directions of the data stream traffic to/from the compressed E1.

1.2.2 IPC Bus Communications [1], [3]

Within the MaxPro System a second internal bus is available for communications. This bus is known as the IPC bus. Each board in the rack is able to communicate with the CPU board via the IPC bus. Two IPC buses exist in the configuration (data rate is 2.048 Mbs). The primary and the backup. The system is redundant here to prevent a signal point of failure.

On each board in the MaxPro System, is a M68360 Motorola CPU. These embedded processors utilize an embedded operating system call RTEMS [9]. This allows for standardized modular software development across the board in the configuration. The firmware within each to M68360 CPU's in the rack board is a module used for IPC communications this enables the main CPU card to communicate, download firmware and data along management of the system status/diagnostics required.

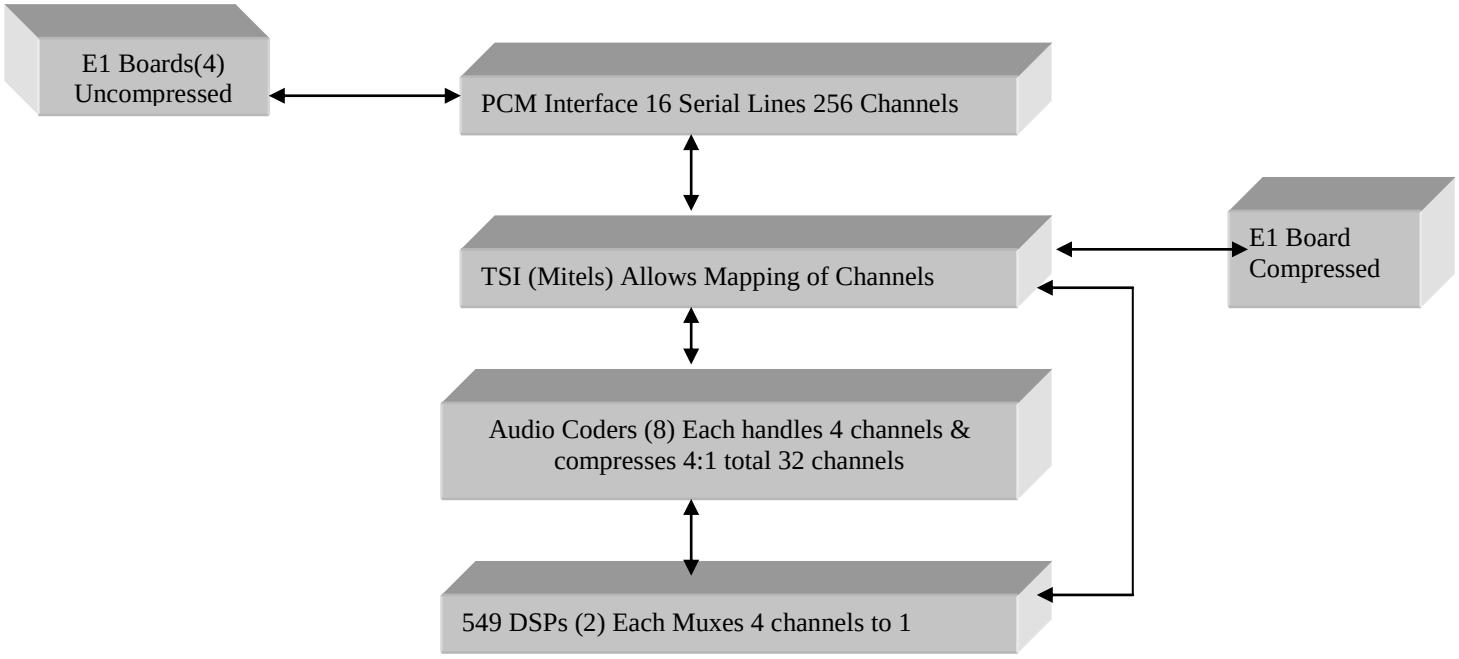
1.2.3 DSPR-32 Card (4:1 Transcoder) [1], [2], [3], [8],[27]

The current software development required to support the product enhancement will entail software changes to the CPU card, E1 Card and reworking the DSPR code due to the hardware changes required to support the product upgrades. In the new board, the hardware will consist of 4 MT8985 (TSI Chips) used to interface to the PCM bus. The PCM bus consists of 16 serial lines. Each line supports 32 channels or 256 bi-directional channels. Two MT8985's (8 Serial lines each) will mange the traffic inbound to the board and the other pair will manage traffic transmitted back out onto the PCM bus. This architecture gives the board the capability to send/receive data streams through any timeslot.

The MT8985's interface with the Audio Codes AC481xxA-C Chips (AC's) , (TI TMS320C549's with prepackaged firmware). The board has 8 of these configured each is able to handle 4 64 Kbs uncompressed traffic channels. These chips are able to compress a 64 Kbs data stream into a 9600 data stream. These chips upon initialization/reset are to be loaded with firmware to operate properly. Each Audio Code Chip is to handle the communications traffic of 4 of the 32 channels from an uncompressed E1 circuit and is able to compress the data stream using an internal 4:1 data compression algorithm. These chips are used to receive the traffic and compress/uncompress E1 data stream traffic.

After the Audio Codes process the data stream, the information is then processed by the two TI TMS320C549 DSP chips (549's). Each 549 is able to handle data stream traffic from 16 compressed channels. This means each 549 will interface with 4 Audio coder devices. They will manage framing of data to send/receive information via two MT8985's on the board used to send/receive compressed/uncompressed E1 communications. The total capacity of the DSPR-32 is able to interface with 40 PCM bus bi-directional time slots [27].

Each 549 has internal to the chip 32 Kword of RAM available for customized programming. The 549's interface with the onboard M68360 CPU via 16 bytes of dual ported ram which is word addressable. Since the AC chips are also 549's they also have the same dual ported RAM available. This is the mechanism that will be used communicate and control the new 549's and the AC chips on the DSPR. This interface enables the M68360 CPU to interact with the 549 chip to setup/reset, configure, download and to manage desired operations available. In the initial phase of the project the DSPR board will have all the firmware preloaded in Flash EPROM memory. The AC chip firmware from the vendor enables the AC chips to compress/uncompress data in the E1 64 Kbs channels presented to the chip. In this new DSPR configuration, each AC is to manage 4 channels. Each 549 manages 4 AC chips. When the AC processing is complete the 549's will receive the data for 4 549's which is a total of 16 channels of data (1/2) of the E1's 32 channel communications per 549 chip on the DSPR board. In this architecture the 549's will be responsible to frame the communications and perform further processing to be identified in the design phase. After the 549 processing is complete the next task at hand for the new 549 firmware is to then send to processed communications back out onto the PCM bus via the TSI interface.


E1 Boards(4)
Uncompressed
PCM Interface 16 Serial Lines 256 Channels
TSI (Mitels) Allows Mapping of Channels
E1 Board
Compressed
Audio Coders (8) Each handles 4 channels &
compresses 4:1 total 32 channels
549 DSPs (2) Each Muxes 4 channels to 1

The onboard M68360 CPU contains the RTEMS embedded operating system along with the IPC bus communications module used to interact with the other boards in the rack configuration. The new hardware architecture in the upgraded DSPR board is different enough that this effort will require new software along with new design and testing of the changes. Some code can be ported onto the new system to save schedule time this can be done for the 360 CPU firmware, and using the vendor's AC firmware. Some untested code at the present time exists for the 549 that allows one to frame 549 communications. However all the firmware required to handle the interactions with the other interfaces on the DSPR board are presently not available. The new design will need to factor in the following interfaces and functional operations.

- M68360
  - Port existing firmware onto new board
  - Analysis to determine changes required here
    - Host Interface 360 CPU to 549's
    - Download 549's firmware (bootstrap)
    - Download AC firmware via the 549 after bootstrap.
    - Interfaces changes via IPC bus identified
    - Updates to the 360 CPU Commands required
    - Status monitoring of board 549's/AC's/TSI states
    - Reset/Error Processing
    - Alarm/Error Management Processing and Interface to CPU board
- 549
  - Determine bootstrap operation
  - Define 360 to 549 Interface function/operations
  - 549 firmware download operation
  - Host Interface to AC chips
    - Setup/Reset, Configure
  - Download AC firmware into AC chips
  - Reset/Error Processing
  - Management of AC's Communications
    - Framing
    - Further Processing
    - Transmit Results via TSI interface
  - Host interface to TSI
    - TSI Interface Operations(Setup/Reset, Configure, Transmit)
  - Compressed Communications Management Operations
  - Decompress Communications Management Operations
- Audio Coders
  - Review vendor firmware functions
  - Prototype Test operations desired for design/development
  - Determine Host Interface Operations to 549 chips
  - Determine Interface Operations to AC chips
  - Compress/Uncompress communications management
- TSI
  - Interface to 549's
    - Setup/Reset, Configure, Transmit/Receive

1.2.4 CPU Card [3]

The CPU board contains an M68360 CPU that uses the RTEMS Kernel, This uses the IPC bus to support HDLC communications to the other boards in the rack. This CPU performs the following functions: 1) Communications with the External Console, 2) Internal communications via IPC communications using HDLC Downloads program code and the DSP's on power up or when required. 3) Processes communications between the DSPR boards and the E1 boards over the TSI interface, 4) Controls

operations of the TSI. The Software modules identified in the legacy systems are the following: a) POST, b) FPGA Image Download, c) IPC Communications of HDLC bus, d) Global TSI Management, e) Communications status monitor and statistics, f) Peripheral card status, monitor and reset, g) OAM communications (Ethernet), h) Peer communications and synchronization.

The CPU interfaces with several leds alarm indicators (Minor, Major, Critical Alarms) and several status leds (Operational, On Line, Administrative). A manual reset is provide here also. The CPU boards interfaces also with an RS232 Port, an Ethernet 10BaseT connection, and a bank of flash ram.

The CPU card is configured such that when either an operator implements a hardware reset or the software in the CPU performs a reset that a signal is set to the other cards (DSPR, E1) to perform a hardware reset. The CPU can selectively send a reset to an individual board in the chassis if desired. The CPU FPGA has a Bit Error Rate Test (BERT) function able to test individual 64 kbs channels. This feature allows the FPGA to monitor anyone of the 128 channels and determine if data framing overruns are occurring. This feature allows the system monitor individual channels of the E1 boards to determine if any errors are present.

1.2.4.1 CPU Board Host CPU

The host CPU is a Motorola MC68360. The board has 2.0 Mb of RAM and 0.5 Mb of Flash RAM. The host CPU performs the following functions: a) Communicates with the main CPU board via the IPC bus, b) Host the RTEMS operating system c) Downloads program code d) Allows for control via local debug RS232 Port e) Controls board status indicators n) Support diagnostic loopback testing

1.2.4.2 CPU IPC Bus Communications

The board interfaces with the IPC bus via Serial Communications hardware. Two IPC buses are supported IPC Bus A and IPC Bus B. When the board powers up one is setup as the primary and another as the backup. During power up initialization the on board 360 CPU initializes the Serial Communications hardware to enable IPC bus communications with the Primary CPU board.

1.2.5 T1E1 Card [1], [3], [18]

The E1 board provides external interfaces to the E1 trunks. The E1 compressed board is able to manage 4 E1 spans of information available on the PCM bus perform processing to timeslots in both receive and transmit directions. Each E1 contains a M68360 CPU along with an on board DSP. The DSP is used to interface with the E1 trunk data and to transpose the traffic to/from the PCM bus channels. The following lists the legacy system functions of the E1 board: a) POST, b) FPGA image download, c) DSP Image download, d) IPC communications, e) TSI management, f) Communications status etc., g) E1 Framer setup/management.

The board contains four framers, a Host CPU M68360, Clock Selection/rate adjustment logic, frame pulse generator, a Time Slot Interchanger system bus interface. An FPGA provides clock select/generation, timing, monitoring. The M68360 CPU has 512K x 32 RAM and 256K of Flash RAM.

The E1 board has six status indicators. These are connected to LEDS to display RED, GREEN or YELLOW. Four of the LEDS indicate the status of the states of the 4 spans. One is for Administration status and another is for general alarms management indications.

1.2.5.1 T1E1 Data Flow

Incoming E1 analog PCM data streams are managed by the framer/transceivers which are set to the clock rate of 2.048 MHz. The framers establish frame synchronization. The clock rate is converted to the 4.096 MHz TSI System Clock and associated to the 8 Mhz TSI framing pulse. These operations allow the TSI to stream in PCM data into the TSI allocations. In the opposite direction TSI slot data connects to the framer/transceiver(s) driving the outbound E1 link. The onboard host 360 CPU receives IPC messages

which allow for the configuration of the time slot format to be an E1 data stream. The hardware can support both T1 and E1 data stream clock rates.

1.2.5.2 T1E1 Card Host CPU

The host CPU is a Motorola MC68360. The board has 2.0 Mb of RAM and 0.5 Mb of Flash RAM. The host CPU performs the following functions: a) Communicates with the main CPU board via the IPC bus, b) Host the RTEMS operating system c) Downloads program code e) Controls operation of the TSI f) Controls operations of the framer/transceiver and reports to CPU board g) Sends signaling messages for received signal changes h) Converts signaling messages received into time slot signaling code i) Controls selection of local and system timing j) controls operation of timing monitors in FPGA along with reading status k) Allows for control via local debug RS232 Port l) Controls communication via PCM timeslots m) Controls board status indicators n) Support diagnostic loopback testing

1.2.5.3 T1E1 Framer/Transceivers PM6341 E1XC [19], [20]

The interface between the system PCM bus and the T1/E1 framers/transceivers is provided via the Mitel MT8985 chips. Together 4 MT8985(s) provide the ability to connect any 512 PCM channels to any of the 4 Transceivers/Framers and from any Transceiver/Framer to any of the 512 PCM channels. The Framer/Transceiver chips used on the board are the PM6341 devices.

On the receive side the PM6341 Framer/Transceiver is used to interface with the PCM bus to frame/align data stream into 2.048 Mhz digital signal. Here also the chip supports detection of various alarm conditions such as 1) loss of signal, 2) loss of frame, 3) loss of signaling multiframe, 4) reception of remote alarm signal, 5) reception of remote multiframe signal , 6) Alarm indication Signal (AIS), 7) Timeslot 16 alarm indication signal. The chip supports performance monitoring and accumulates the following types of errors: 1) CRC-4 errors, 2) far end block errors, 3) framing end bit errors.

On the transmit side generates a basic 2.048 Mhz E1 signal. The chip supports transmission of HDLC messages on a data link. The data link can be inserted into timeslot 16.

The PM6341 supports a parallel port visible in the boards CPU I/O memory. This port controls operation of the device.

An interface to the PCM Interface supports connections to the PCM data streams.

1.2.5.4 PCM Bus and TSI Communications

Data communications between boards in the system will take place via the PCM serial bus. This bus has 16 lines supporting TDM serial streaming consisting of 32 channel frames. Channels are assigned by the CPU card via IPC bus communications messaging. Once channel connections are established data stream communication can be setup through the connections.

1.2.5.5 T1E1 FPGA [29]

An onboard FPGA (Xilinx) contains the support logic. The device is connected in a peripheral mode and must have its program downloaded via the CPU. This supports clock synchronization and selection modes for E1 or T1. The logic functions of the FPGA are managed by the firmware downloaded.

The FPGA has a set of Input/Output Blocks (Ports) (IOBs). These interface to internal Configurable Logic Blocks (CLB). These are used to perform specific logical gate operations such as an Add, Or or Xor using up to five arguments. The FPGA has Configuration Memory which is downloaded during startup. The Logic Program defines Programmable Interconnections associating resources of the FPGA. These instructions route the paths of the IOB's and CLB's into logic networks. The FPGA uses these operations

to perform high speed data stream operations to support the interaction between the TSI and Framers as required.

1.2.5.6 T1E1 IPC Bus Communications [21], [28]

The board interfaces with the IPC bus via Serial Communications hardware. Two IPC buses are supported IPC Bus A and IPC Bus B. When the board powers up one is setup as the primary and another as the backup. During power up initialization the on board 360 CPU initializes the Serial Communications hardware to enable IPC bus communications with the Primary CPU board.

1.2.6 PC Console Graphic User Interface

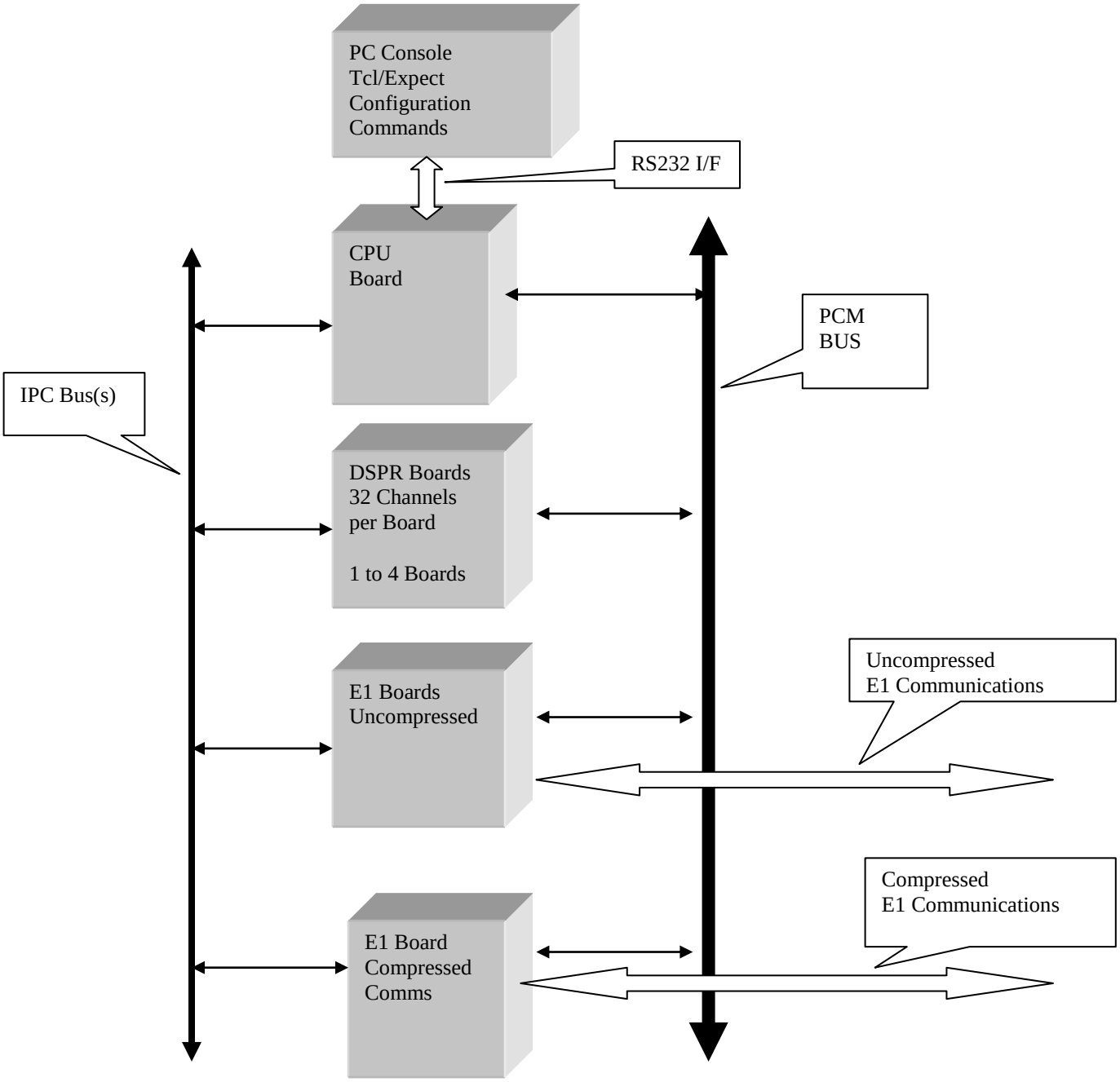
An additional task identified for the project is to provide a capability to configure and manage the boards in the chassis via a PC Console. This can be done via an RS232 interface between the CPU board and the PC console. The CPU board will parse the messages determine if the command is to be processed locally by the CPU board or routed to another board in the chassis. The CPU code will send messages through IPC communications to the board which is to perform the command. This feature will enable the development and test teams to manage/debug/test operations faster. In the new system the PC Console Interface will be used to send records to the CPU board to configure board parameters and to manage status.

Note: For next phase of the project an additional feature to enable remote configuration/management of another chassis will be to utilize ½ of a 64kbs E1 channel send receive commands will need to be considered. The work for this function at this time will be deferred until after the next release of the system.

1.2.7 Integration/Testing

Upon completion of the changes two levels of testing are required to validate the proper functional operations of the new architecture. On the first level will be unit level/board level testing. This is expected to be completed by the person responsible for the task assigned to them. The second level is integration testing, this involves both the person(s) responsible for development along with person/group that is to be responsible to verify the system operates as required. For system level testing a need exists to have a testbed available for the project. At the present time, a platform configuration appears to be viable for this purpose. Here one will set up four E1 lines interfaced to testing equipment to generate input communications. This will be connected into a rack via an E1 card used to process uncompressed E1 information. This E1 card will take the information and place onto the PCM bus. Prior to this point the testor(s) will properly configure the architecture as defined for the test. The new DSPR cards will also be available here. The DSPR board will compressed the information and place the data back onto the PCM bus. Next the E1 card with the firmware to manage compressed E1 communications will receive the information and send out onto an E1 line. At this point to same equipment, time etc. a loopback configuration will be established to redirect the compressed E1 data back into that E1 card. Here the compressed data is translated back onto the PCM bus directed back into the DSPR's which then uncompresses the information and send back onto the PCM bus. At this point the other card is to take the uncompressed E1 communications and send back out to the destination. In this case this will be the monitoring equipment set up to validate the results with those expected for the tests. In these test(s), each of the 32 E1 channels will need to be tested to verify the results of the operations.

1.2.8 System Configuration



2 Current Software Project Schedule Activities

2.1 Task Work Break Down

The Software Development team is now in place to support development of this project. Mike Grudsky is working on the Hardware development and is working in the capacity as project technical leader. Paul Murray is working as the software architecture technical lead, Eric Krieg is working on the E1 board software along with the PC/Interface to the CPU, Steve Bulack is working on the CPU board, Art Larson is working in a project supporting role to handle software design and project software issues.

Someone also needs to be dedicated establishing the integration and testing environment. One needs to assume that this will take several weeks of effort also. From this preliminary analysis, the project seems to have several potential critical paths not just the one identified. These are the ones identified so far the are the following:

- Software Design and Architecture Documentation
- Scope of the work to handle the DSPR software development
- Potential critical path on the CPU card upgrades and integration of these function through out the other features of the system
- PC Console Testing Code (Tcl/Expect), CPU Board SW updates, E1 Board SW Updates and DSPR SW updates. This can be reduced as a potential critical path with a good SW design that specifies the functionality to be put into each of these components. This will also reduce the critical path potential problem with integration.
- Integration and Testing the new system. To make sure this part of the work does not take a long time at the point when the Software Design/Architecture is defined someone needs to then focus in on this area to be ready validate the functionality. First they will need to set up the integration/testing environment quickly next they will need to design/devise system tests and the means to verify the results. They will need to extract from the software design the command functions presently in the CPU interface to the external environment along with the new features to be supported in the present software development/design effort. This will need to first validate functionality along with the performance desired. This person will need to work closely with the team to support rapid validation of features under development thus reducing the time in development.
- To reduce project development time, One needs to device some baseline functionality to work data through the entire system and to then add additional features in the next build.

2.1.1 Software Design/Architecture

- Document System Architecture
  - Review current Software Architecture in each Board

- Software Design Documentation
  - Identify new functional features to be added to the system and to each board
  - Defined the new features and specify the new software architecture
  - DSPR Design/Analysis
  - Identify E1 Software Upgrades
  - CPU Board Software Design
  - PC Console Interface Testing via Tcl/Expect using PC Testing environment

- Message Interface Specifications
  - Define the PC Console commands to./from chassis (Message Interface Description)
  - Review and identify commands present and required in each board
  - Define the Message Interface to/from PC Console via CPU board to other boards in the rack.
  - Specify the Message Operations to be supported via the PC Console interface

- System Test Plan
  - Integration Setup/Design

2.1.2 DSPR Coding /Testing

- Analysis/Design
- Design Review
- Code the DSPR M68360 updates
- Code the TMS320x549 functions
- Review E1 functions with 549
- Unit Test
- Integration System Test with E1 and CPU
- Debug/Fix

2.1.3 E1 Coding/Testing

- Review current E1 Software/Identify updates required
- Code the new E1 Interface Functions
- Code updates required for working with new DSPR if needed
- Unit Test
- Integration System Test with DSPR and CPU
- Debug/Fix

2.1.4 CPU Board Coding/Testing

- Review Current Code
  - Reduce existing functionality and rebuild to support current architecture
  - Code new interfaces (External, E1, DSPR)
  - Code new Interfaces to support additional commands
  - Code functions to interact with the DSPR and E1 as identified and needed
  - Code PC Console Operator Interface to/from RS232
  - Code GUI Interface to PC functions
  - Rework Chassis Management Functions
  - Rework Resource Management Functions
  - Rework DSPR/CPU Commands
  - Rework Traffic Management function to properly route new PC Console messages through system.
  - Rework Async Resource Management feature
  - Rework Channel Management feature
  - Rework Alarms Status Management
  - Code Flash RAM Management Functions

2.1.5 PC Console Interface via Tcl/Expect (GUI on PC)

- Learn Tcl/Expect Utilities
- Design/Analysis of functions/features
- Code interface functions to CPU board
- Code interfaces via CPU to E1, DSPR
- Add Data Dump Feature
- Add E1 Interface functions features
- Create a utility to configure the system via CPU

2.1.6 Support and Management of the Project

This task involves supporting a the project and updating information required.

2.1.7 Integration Testing

Here a system test plan will be used to verify functional operations identified

3 High Level Software Design

3.1 Common Software

This section defines software components common to all the M68360 Boards in the system.

3.1.1 Common M68360 RTEMS Processes

3.1.1.1 Start Up Processing

When the system boards power up a Real Time Operating System, (RTEMS) is used to manage operations. This is an embedded multitasking executive, that allows tasking priority/preemption, interprocess communications, supports device drivers for the board interfaces thus allowing the tasks to perform simple read/write operation to send/receive messages. Tasks can be event driven such that a task can be suspended waiting for message I/O and/or a message timeout event to occur. The executive supports timer functions also. RTEMS supports tasks using system wide event flags also. RTEMS supports 32 global event flags. Eight of these are used for the task scheduling management of the message extensions. The executive scheduler allows tasks to set/use event flags. Processes can be suspended to wait on an event flag to become active.

In each of the M68360's the first code to start running is the board support software bspstart.c this code performs operations specific to starting up the system. In this routine bsp_start() a call is made to start up the RTEMS executive via the call to rtems_intialize_executive(). After this call is made the RTEMS operating system manages further processing in the 360 CPUs. The first task RTEMS starts is the System Initialization Task init() usually found in init.c for the specific board. It is this process that then starts up the other processes in the system such as IPC_RX, IPC_TX, RMON etc.

In each of the board configurations a common set of processes will be present. These tasks are the following:

- Init – Start up Initialization Task. When each board powers on diagnostics are executed and if successful they set the Board State LED to GREEN otherwise they set the LED to YELLOW. When the initialization task runs, it will read the value of the led to determine the board's health based on the results of the power up diagnostics testing.
- IPC TX – Interprocessor Transmit Task used to send messages over the IPC bus to another board in the chassis.
- IPC RX – Interprocessor Receive Task used to receive messages over the IPC bus to from another board over the IPC bus.
- RMON – RTEMS Monitor Process this is used to read commands and to process the messages. The messages and supporting routines will be different in each of the boards. This task also creates a console read task (crd) which reads the input data stream and placing the input data into a message queue processed by RMON.

3.1.1.2 RTEMS Message Extensions [11], [14]

Source code for RTEMS is available here, to more easily support communications, an extension to the interprocess communications was added into the code (msg interprocess communications extension). When a RTEMS process is created each process is assigned a set message queues. The process interacts with the queues with a set of simple operations:

- Msg_allocate – Allocate message buffer
- Msg_free – Deallocate message buffer space
- Msg_put – Transmit message
- Msg_get - Receive a message, a timeout argument is provided here also.

When a call is made to send/receive some of the arguments are the Queue ID to be used and the Global Process ID to receive the message. When the system starts up, RTEMS provides access to a global data area that has information on the process ids known to the system. The message system provides a set of definitions used to specify a board and the task ids. This enables one to perform chassis wide communications across the boards to/from specific tasks.

Presently a set up eight message queues are provided with each process created some of these queues are identified here:

- **Command Queue** – This is used to set up parameters for management of a task
- **Response Queue** - Used to receive response messages
- **Blocking Response Queue** – Task will be suspended here when the a request is made to receive a message from this queue.
- **User to Network Queue** – Message data arriving externally into the board to task's queue
- **Network to User Queue** – Message data locally going towards the IPC bus
- **Application Queues** – An additional set of three more queues is available these are available for use to manage requirements specific to an application.

When a call is made to the message interface function one of the parameters to send a message is the message id of the destination. This id consists of the network address, task id and the queue id one of the eight above. In the current system architecture predefined include files provide these parameters.

3.1.1.3 RTEMS Monitor Process (RMON)

The monitor process allows for a standard set up commands common to all M68360 boards to be processed. The facility also allows one to extend the command set to perform additional operations that are specific to a given board environment. The standard command set identified is the following:

- help -- get command information
- exit -- invoke a fatal RTEMS error
- symbol -- show entries from symbol table
- config -- show system configuration
- itask -- list init tasks
- mpci -- list mpci config
- task -- show task information
- queue -- show message queue information
- extension -- user extensions
- driver -- show information about named drivers
- dname -- show information about named drivers
- object -- generic object information
- node -- specify default node for commands that take id's
- curnode -- print current node
- defnode -- print or set default target console node
- defpid -- print or set default target console process

3.2 CPU Design [12],[13],[14],[15],[16]

3.2.1 Overview

When the CPU board is powered up the Software loads up device drivers, runs local diagnostics, initializes the interface chips, downloads the FPGA, sets up IPC communications to enable interaction with the DSPR and E1 boards.

3.2.2 Start Up

When the CPU M68360 starts up, the CPU Supervisor task executed during the RTEMS initialization. This task in turn starts up the the following task:

- Initialization Task (INIT) – Start Up Task, creates other tasks and then goes into a loop monitoring FPGA alarms.
- IPC transmit task (IPC_TX) – Receives message status data from boards in the system.
- IPC receive task (IPC_RX) – Sends messages to other boards via IPC bus
- User Interface Task (CPU_UI) – Processes user commands
- Resource Manager Task (RES) – Manages board/chassis configuration DSPR and E1/T1 board status.
- The Console Read Task. (CRS) – Reads RS232 Commands send to CPU_UI)
- Debug Monitor (DMON) this is a development debugging task not present is the production code
- RTEMS Monitor (RMON)

3.2.2.1 Firmware Download Support for the DSPR-32 Initialization

After the CPU is operational. The DSPR will also power up. When both the CPU and the DSPR are operational and have established communications. The DSPR will interact with the CPU to then download firmware for the 549 and the Audio Coders. Here also the CPU will provide configuration data identified. In the initial phase the DSPR-32 board itself will contain most of the firmware to be downloaded into its 549's, Audio Coders, FPGA etc. The DSPR-32 will also be setup with a default configuration. The job of the CPU is to download into the DSPR-32 configuration data required.

3.2.2.2 Dynamic Configuration of DSPR - TBD

The CPU is to support dynamic configuration of DSPR parameters that are as of yet to be identified.

3.2.2.3 Firmware Download for T1E1 Board(s) Initialization

In addition to supporting the DSPR, the CPU is to synchronize with each of the T1E1 cards as they power up and support firmware/configuration downloading. As with the DSPR board, the T1E1 boards will have firmware in the board memory used to load the DSP, FPGA etc. The T1E1 boards are to also power up with a default set of configuration parameters. When communications is established with the CPU board will send initial configuration parameters.

3.2.2.4 Dynamic Configuration of T1E1 Boards

The CPU is to support dynamic configuration of T1E1 data. An additional feature here is that updates have been made to the T1E1 software to allow new configuration parameters to be sent down into the board. An issue here is to determine if this functionality requires additional work in the PC to support the PC interface and will updates to the CPU resource manager be also required.

3.2.3 User Interface Task (CPU_UI)

The current CPU software provides for management of connections, configuration information on both the local CPU and the other boards in the chassis, build commands, list configuration status etc. The current set of commands are allowed to be batched. This means that the entire set of commands which can then be executed or aborted. The following is the set of commands available at the present time. As noted in the commands this system supports 4:1 E1 compression since each E1 support 32 64 Kbs channels this means the number of clear channel is 128 and the number of compressed E1 channels is 32. The software in the system has predefined the mapping of the channels of the uncompressed E1 boards to specific channels on the compressed E1 board. This means the user interface commands only need to provide a definition to connect/disconnect a channel path. The underlying software will then reference the predefined channel mapping and establish to the proper mapping for the system. The chassis has 22 slot positions for boards of which one is reserved for the CPU board. This means that 21 slots positions are available in the chassis for placement of the DSPR-32 and the T1E1 boards.

Note1: The DSPR-32 actual data compression is actually about 5.3 :1 this translated into mapping 120 channels of voice/data communications into 22.5 channels on the compressed E1 board. The clear channel 16 for each one of the E1 boards E1 Board A thru E1 Board D have 120 channels of voice/fax/data communications that is compressed into 22.5 channels on the compressed E1 board. The clear channel 16 for boards A..D are mapped into the E1 compressed on clear channels using another 4 clear channels of the 32 channels available.

3.2.3.1 User Interface Commands

// Channel Connect/Disconnect Commands
channel connect <E1 Board Id 1..4> < Channel No 1..31> cmp // Compressed {Default compresssed}
or
channel connect <channel no 1..128> clear // Uncompressed

channel disconnect <E1 Board Id 1..4> < Channel No 1..31>
or
channel disconnect <channel no 1..128> clear

channel list // List Channel Assignments

// Board Definition Commands
board define <Board ID 1..21> dsp // Define Board as DSPR Card
board define <Board ID 1..21> net // Define Board as Compressed E1 Card
board define <Board ID 1..21> user // Define Board as Uncompressed E1 Card

board undefine <Board ID 1..21> //Deassign Board - Disables operations with board in the board slot id

// Framer Synchronization Commands
clock set <primary|secondary> internal framer id 1..4 // Set CPU Mitel clock framer
clock set <primary|secondary> net framer id 1..4 // Set E1 Compressed Board Mitel clock framer
clock set <primary|secondary> user framer id 1..4 // Set E1 Compressed Board Mitel clock framer

execute // Execute the command batch
discard // Abort the command batch

3.2.3.1.1 User Interface Commands Updates - TBD

For the initial phase the plan is to support only the existing User Interface commands. At a subsequent time additional commands may be provided as required. For the initial MaxPro System. The supporting infrastructure for the user interface commands will require review and analysis to determine impact of the new changes. Areas to be identified are the following:

- DSPR-32 Board Configuration commands
- E1 uncompressed Board Configuration Commands
- E1 compressed Board Configuration Commands

3.2.3.2 User Interface Command Message Processing

The User interface task (CPU_UI) receives text messages as defined above from the user and responds by sending an appropriate message to the T1/E1 Card Address synchronously creating the message request(s) and waiting for the T1/E1 response. The User Interface Task parses the command and sends the request messages to the CPU Resource Management Task. The Resource Management Task then examines the current chassis configuration and then issues messages to the T1/E1 boards. The Resource manager uses the message type defined for processing user commands which is:

// Channel Command Processing

- MSG_TYPE_PROCCMD
  message subtypes:
  - PROC_SET_USER_MAPPING
  - PROC_SET_USER_MAPPING_FORCE

In response to these messages, the following responses may be received from the T1/E1 boards:

- MSG_TYPE_ACK

- MSG_TYPE_NACK
  message subtypes:
- MAPPING_GENERAL_FAILURE
  - MAPPING_VALIDATION_FAILURE
    details for this subtype:
    TOO_MANY_COMPRESSED_CHANNELS
    TOO_MANY_CLEAR_CHANNELS
- MAPPING_RESOURCE_FAILURE
- MAPPING_ACTIVATION_FAILURE

In implementing the other user interface commands to the T1E1 Board, the CPU_UI task also sends these messages to the resource manager.

MSG_TYPE_PROCCMD
subtype:

PROC_GET_USER_MAPPING // Channel List Command Request

PROC_SET_BOARD_DEF // Board Definition Commands
PROC_GET_BOARD_DEF

In response to these messages, the following responses may be received:
MSG_TYPE_ACK
MSG_TYPE_NACK

3.2.4 RTEMS Monitor Command Extensions – TDB

3.2.5 CPU Flash RAM Interface (NEW)

In the new system, the CPU is to provide a mechanism to store download programs for the boards along configuration parameters in the on board FLASH RAM. Here the goal will be to support the following:

- Provide an Interface to Read/Write Flash RAM Parameters on the main CPU board via the IPC messaging system to allow other boards in the system to read/write data to/from the flash ram available on the main CPU board.
- Provide and Interface to Download Programs into the DSPR for the new 549 firmware and the code for the CPU board FPGA.(Phase II)
- Provide a means for users define a new set of default configuration parameters for each of the board configurations and at the same time retain a known functional default set of configuration parameters to be used in case the user defined parameters result in a operational malfunction.
- Provide a means to down load into the E1 boards FPFA code and DSP code for both the compressed and uncompressed (Phase II)
- Provide a means to download into the DSPR board 549 Code and Codec Code. (Phase II)

- Provide an interface to PC via RS232 to support PC Configuration download of E1 configuration data from the PC into Flash RAM. Subsequent to this point provide a message interface to download the E1 board with the PC configuration data. This will require analysis of the PC record formats and definition of new messages/format to load records into the E1(s). In addition, one needs to determine what action will initiate the download operation into the E1 board(s). [Note this will require one to synchronize with the E1 configuration design for this purpose]

3.2.6 Diagnostics

The current CPU system supports diagnostics upon startup.

- In the subsequent phase more software may be required here to support additional diagnostics such as being able to run diagnostics on the other chassis cards (DSPR, E1). Here also we may need to consider loopback communications tests.

3.2.7 CPU IPC Messaging

3.2.7.1 CPU IPC Transmit Task (IPC_TX)

Handles IPC messages sent to the CPU board via the Siemens Serial Communications Chip that is connected to the chassis IPC Bus. The messages received are the following:

- SYSCMD_INIT – Message Request to Initialize a Board
- SYSCMD_RESET – Message Request to Reset a Board
- SYSCMD_CONFIG – Message Request to Configure a Board's Configuration Parameters
- SYSCMD_DEVSTART - Message Request to Start a device
- SYSCMD_DEVSTOP – Message Request to stop a device
- SYSCMD_SETLOOPBACK – Message Request to board(s) to go into Loopback Testing Mode
- SYSCND_CLRLOOPBACK – Message Request to board(s) to exit loopback mode
- SYSCMD_PRTCONFIG – Request to fetch Configuration data from other boards
- SYSCMD_PRTSTAT – print statistics request of other boards for perform data on framers performance and alarms
- SYSCMD_CLRSTAT – message request to clear statistics performance tables in the boards

3.2.7.2 CPU IPC Receive Task (IPC_RX)

This task receives CPU IPC messages to the other boards via the Siemens Serial Communications Chip through the IPC bus.

- The only messages identified so far sent for the CPU IPC Receive Task are ACK/NACKs (Acknowledge/Not Acknowledge Messages).

3.2.8 CPU Resource Management Task (RES)

The resource management module in the CPU provides dynamic resource allocation/deallocation of T1/E1 channels resources. The concept of a path (circuit connect), is a primary theme in the functionality of the resource manager A path is an ordered collection connected resources. A path exist between an entry point (e.g. timeslot of an E1 stream) and an exit point (outbound E1 channel). For every path of connecting uncompressed E1 data stream to a compressed E1 stream there is to be a symmetrical path connecting channels in the opposite direction.. The resources involved can be backplane timeslots. This establishes a path in the user to network direction and the opposite symmetric path will be in the network to user direction. It is necessary to be able to group together resources on criteria other than path. For example four backplane slots carrying compressed traffic to/from a DSPR card shall be as one 256 kbs stream (using 4:1 compression). A path is defined as a path entity/node represented in the resource management data structures. A path node has a resource id and a connection state indicator. The state of a connection can be one of the following: Unknown, Connected, Connection Pending, Disconnected, Disconnect Pending,

Connect Failure, Disconnect Failure. The CPU resource manager relies on the abilities of the individual cards to set up a connection between resources. The resource identification scheme is set up to be recognized in all the software running in each of the boards M68360 CPUs. The IPC mechanism is used to communicate requests to the individual cards. The request IPC messages support calls to connect, disconnect and get connection status using the source and destination resource ids. The resource id is defined as combination of the following: 1) slot number (Chassis board id), 2) Resource type and 3) Resource number. The resource types are the following: a) Unknown, b) Backplane Timeslot, c) E1 Timeslot, d) Codec, f) Subchannel, g) aggregate channel (mulitple backplane time slots), h) compressed channel (9600 bits of compressed channel data representss 64000 bits at the decompressed end of an E1 board) and j) Supervisory channel (control communications for LAPD communications).

The resource number has the following specific meaning specific to the resource type:

- The resource number of a backplane type is bus id << 8 | stream <<5| timeslot.
- The resource number of an E1 timeslot is span <<5|timeslot (channel 0..31)
- The resource number of a Codec/549 is To Be Designed (TBD)
- The resource number of a subchannel is TBD in the new architecture
- The resource number of a an aggregate channel is the backplane timeslot number of an even channel pair.
- The resource number of a compressor channel is 0-14 and corresponds to the DSPR .

Note: The resource manager deals with a data structure known as the the Time Slot Interchange (TSI) mapping. This group of data structures enables one to represent the association between aggregate E1 channels assigned to a compressed E1 data streams. This mapping representation is hardcoded and thus the mapping of uncompressed E1 channel data streams to compressed E1 channels is defined with this mapping.

3.2.8.1 CPU Resource Manager Configuration and TSI Mapping - TBD

In the MaxPro software upgrade analysis will be required to update the TSI mapping to properly deal with the new architecture.

3.2.9 Chassis Management – TBD

The CPU board is responsible for the management of MaxPro chassis configuration. The issue is to determine how the current chassis management features operate and to then define the requirements needed to support the new DSPR configuration along with other functions identified here.

3.2.10 CPU Error and Alarm Management - TBD

The task here is to determine what work needs to be done to support alarms/error interaction with the new DSPR board.

The CPU board is to be able to monitor and receive alarm messages from the various boards in the chassis. Here they alarm management function will need to determine if the alarm in from local event specific to the CPU board or from another board in the configuration. A message specification may be required here to support this feature along with a determination as to what functionality will be provided in the first phase of the project

3.2.11 External Interfaces

3.2.11.1 CPU to DSPR Interface

The following messages are used by the CPU to manage operations of this board

- SYSCMD_INIT – Message Request to Initialize a Board
- SYSCMD_RESET – Message Request to Reset a Board
- SYSCMD_CONFIG – Message Request to Configure a Board’s Configuration Parameters

- SYSCMD_DEVSTART - Message Request to Start a device
- SYSCMD_DEVSTOP – Message Request to stop a device
- SYSCMD_SETLOOPBACK – Message Request to board(s) to go into Loopback Testing Mode
- SYSCND_CLRLOOPBACK – Message Request to board(s) to exit loopback mode
- SYSCMD_PRTCONFIG – Request to fetch Configuration data from other boards
- SYSCMD_PRTSTAT – print statistics request of other boards for perform data on framers performance and alarms
- SYSCMD_CLRSTAT – message request to clear statistics performance tables in the boards

3.2.11.2 CPU to E1 Interface

The following messages are used by the CPU to manage operations of this board

- SYSCMD_INIT – Message Request to Initialize a Board
- SYSCMD_RESET – Message Request to Reset a Board
- SYSCMD_CONFIG – Message Request to Configure a Board's Configuration Parameters
- SYSCMD_DEVSTART - Message Request to Start a device
- SYSCMD_DEVSTOP – Message Request to stop a device
- SYSCMD_SETLOOPBACK – Message Request to board(s) to go into Loopback Testing Mode
- SYSCND_CLRLOOPBACK – Message Request to board(s) to exit loopback mode
- SYSCMD_PRTCONFIG – Request to fetch Configuration data from other boards
- SYSCMD_PRTSTAT – print statistics request of other boards for perform data on framers performance and alarms
- SYSCMD_CLRSTAT – message request to clear statistics performance tables in the boards

3.2.11.3 PC to CPU Interface – TBD

3.2.11.4 CPU to External Chassis (Box to Box) Interface – Phase II

This interface will allow the system to send/receive messages from/to another system. The functionality here will be not be addressed in the initial phase. This communications will utilized ½ of a clear channel allocated by the resource manager to perform this communications.

3.3 DSPR-32 Design [27], [36] - TBD

3.3.1 Start Up

As with the other boards the DSPR will first run the board support diagnostics found in bspstart.c which will the start the RTEMS operating system. When the DSPR M68360 starts up, the DSPR Supervisor task executed during the RTEMS initialization. This task in turn starts up the IPC transmit task, IPC receive task, Resource Manager Task (RMON) and the Console Read Task, DSPR OAM task (Used to manage configuration parameters) will be started. During startup the M68360 code will extract programs present in the on board flash memory to download the 549 chips with the new DSPR code. After the 549 chips are loaded the M68360 will then download the vendor firmware into each of the Audio Coder Chips. In addition the FPGA firmware will also be downloaded to enable proper board operations. After the board is initialized, the DSPR will communicate with the CPU to obtain and update its configuration parameters.

3.3.2 Operational Mode

The DSPR role will be to support communications processing required between the E1 compressed and the E1 uncompressed boards. This means that based on the configuration setup parameters defined for the circuit paths the DSPR will merge, route data stream traffic between these two components in both

directions. Data coming from the compressed E1 board can be configured to be either compressed or uncompressed. When the data is uncompressed the DSPR will use the configuration parameters defined for a given connection and take the E1 compressed board 64 Kbs channel data and place this information onto the PCM bus via the Mitels onto the corresponding serial line and channel frame as defined in the configuration to which to map data stream to. This data stream is then received by the E1 uncompressed board from the PCM bus and then transmitted out the corresponding E1 line and channel defined in its configuration parameters to which the data stream is to be routed towards. When the DSPR receives compressed data from the E1 compressed board the processes is the same except for an additional step to decompress data stream is applied. Processing of data streams in the opposite directions follows the same algorithm in the opposite direction.

Note: The CPU resource manager is the system component responsible for management of the data stream connections as they are to be routed through the system from/to the E1 compressed to/from the DSPR and to/from the E1 compressed data streams. When the connections are configured the CPU board will send information to the corresponding DSPR, E1 compressed board, and the E1 uncompressed boards to ensure that all the system components properly represent the circuit connections.

3.3.3 DSPR-32 Alarm and Error Management - TDB

The DSPR will be responsible to manage the board local error conditions that may arise. When an error does occur the board is to perform notification via Alarm(s) sending information about the alarm/error via an IPC communications to the main CPU board.

3.3.4 External Interfaces

3.3.4.1 DSPR-32 to CPU Interface

The following messages are received from the CPU to manage operations of this board

- SYSCMD_INIT – Message Request to Initialize a Board
- SYSCMD_RESET – Message Request to Reset a Board
- SYSCMD_CONFIG – Message Request to Configure a Board's Configuration Parameters
- SYSCMD_DEVSTART - Message Request to Start a device
- SYSCMD_DEVSTOP – Message Request to stop a device
- SYSCMD_SETLOOPBACK – Message Request to board(s) to go into Loopback Testing Mode
- SYSCND_CLRLOOPBACK – Message Request to board(s) to exit loopback mode
- SYSCMD_PRTCONFIG – Request to fetch Configuration data from other boards
- SYSCMD_PRTSTAT – print statistics request of other boards for perform data on framers performance and alarms
- SYSCMD_CLRSTAT – message request to clear statistics performance tables in the boards

3.3.4.2 DSPR-32 to E1 [36]

The DSPR has to interface with both type of E1 cards managing the flow of data streams both directions. This means that interfaces exist for this purpose. These are identified as the following:

3.3.4.2.1 DSPR-32 to E1 compressed

The E1 compressed interfaces to the DSPR-32 via the PCM bus. The DSPR-32 compresses 64 kbs data streams into 9.6 kbs data streams but added PAD overhead resulting in 12 kbs data streams per channel. The data is sent in PCM frames each carries 30 bytes of compressed data traffic (30 voice channels).

3.3.4.2.1.1 Output Frame from DSPR-32 via PCM bus to E1 Compressed PCM Frame

000x000x000x000x000x000x000x000x000x000x000x000x000x000xSSSS = 256 bits/PCM Frame

0 – Data Payload Nibble , x – PAD/Header Nibble , S – Sync Nibble

3.3.4.2.1.2 Clear Channel Output to E1 Compressed

The E1 channels configured to be transmitted as clear channels will not be compressed by the DSPR-32. Here the TSI configuration (Mitels, Framers) will be setup to merely send that clear frame channel data in a separate PCM TSI which is then combined by the E1 Compressed board.

3.3.4.2.2 E1 compressed to DSPR-32

3.3.4.2.3 DSPR-32 to E1 uncompressed - TBD

3.3.4.2.4 E1 uncompressed to DSPR-32 - TBD

3.4 T1E1 Design [18] ,[36]

The E1 M68360 CPU performs the following functions :

- Hosts RTEMS Kernel
- Communicates with main CPU board via IPC channel
- Downloads Program Code into the FPGA.
- Controls operations of the TSI
- Controls operations of the framer/transceivers
- Reads Link status of framer/transceiver and reports to CPU board
- Sends signaling message for received signaling changes
- Converts signaling messages received into time slot signaling codes
- Controls selection of the local and/or system timing
- Controls operation of timing monitors in FPGA and reads status
- Control communication channels via PCM timeslots
- Controls board status indicators
- Supports diagnostic testing and loopback control
- Communicates with CPU board to provide error/alarm status (new)

The interface between the framers/transceivers is provided via the four Mitel chips. Together they provide the ability to connect from any of the 512 PCM channels to any of the 4 framer/transceivers 512 PCM channels. Each Mitel has 8 serial inputs and 8 serial outputs. Each input and output consists of a pair of 32 64Kbs channels multiplexed to form a 2048 kbit/s stream. The input and output Digital switches are interconnected to support loopbacks for diagnostic testing.

3.4.1 Start Up

When the CPU M68360 starts up, the E1 Supervisor task executed during the RTEMS initialization. This task in turn starts up the

- Init – T1E1 Supervisor/Start Up Task
- IPC transmit task (IPC_TX)
- IPC receive task (IPC_RX)
- RTEMS Monitor Task (RMON)
- the Console Read Task. (CONS)
- T1E1OAM task (Manages configuration parameters)

3.4.1.1 T1E1OAM – E1 Operations Administration and Management Task

This task receives configuration control messages from CPU. This information is used to setup functions identified. The task also reports back to the CPU status and performance information back to the CPU. In addition the task will send to the CPU notifications of failure conditions. When data stream framing errors occur this function will be signaled. In this case the task will then send a message to the CPU to indicate this condition. The task activates based on a one second timer event. At this point the process polls the framer for status and send the status back to the CPU. Another function of the task is to update the on board LED to reflect the current states of the framers, general errors and/or administration operations.

3.4.1.1.1 T1E1OAM Task Command Extensions

In addition to the standard command set. The E1 board has extended the command set to incorporate the following:

- setclocksource -- set pcm clock source.
  Syntax: setclocksource <bus>
  <bus> => none|busa|busb|both
- fconfig -- set default framer config
  Syntax: fconfig <fmt>
  <fmt> => e1_frame|e1_nf|e1_crc|e1_nf_crc
- bts_config-- set default bts framer config
  Syntax: bts_config <fmt>
- ctsi -- connect through tsi
  Syntax: ctsi <in|out|local> <str>:<chan>
- dtsi -- disconnect tsi stream:chan
  Syntax: dtsi <in|out|local> <str>:<chan>
- ltsi -- list active tsi connections
- otsi -- output data on tsi stream

3.4.1.1.2 T1E1OAM Task System Commands Supported

In addition to these command this task also support the following System Command Requests from the CPU board.

- init -- perform module init and reset
- reset -- module reset
- start -- enter normal operations
- stop -- disable normal operations
- loopback -- enter loopback mode
- clrloop -- exit loopback & reset
- prtconfig -- print current configuration
- prtstat -- print device status
- clrstat -- clear device statistics

3.4.2 T1E1 Design Configuration Parameters – TBD

In the MaxPro architecture review of the management of circuit paths in the compressed version of the E1 board is required to be examined/updated to make sure that the software properly represent the current mapping circuit paths in a 4:1 channel compression algorithm versus the 2: 1 compression used in the prior system

3.4.3 T1E1 Error /Alarm Management [19],[20]

One of the current features of the E1 software is to monitor data communications errors. The task here is to interface to the Framer/Transceivers on the T1E1 Board and obtain the error conditions that may occur. The interface to these chips (PM6341 E1XC) support notification of the following types of alarms.

3.4.3.1 T1E1 Data Stream Reception Framer Alarms

- Reception of a remote alarms and remote multiframe alarm
- Reception of an alarm indication signal (AIS)
- Supports accumulation of alarms counters for the following type
  1. CRC-4 errors to 1000/sec
  2. Far end block errors to 1000/sec
  3. Frame sync errors to 127/sec
  4. Line code violations to 8191/sec

3.4.3.2 T1E1 Alarm Interface to the CPU - TBD

When alarms/errors are detected on board via the PM6341 Framer Chips. The software in the T1E1 360 is to build alarms messages and then transmit this information to the main CPU board. The CPU is to the manage the CPU alarm indicators to set the relays (RED, YELLOW, GREEN) this is the Major, Minor, Critical alarm indicators..

3.4.4 T1E1 PC Configuration Downloads – TBD

This function is to allow the circuit definition to be defined in a PC file and then downloaded into the system via the Main CPU's RS232 port interface. The CPU is to then send IPC configuration messages to the E1 boards to properly configure the circuit definitions. The task here will require analysis in the CPU's messaging system along with the resource management function.

3.4.5 E1 Compressed [36]

The E1 board that manages compressed data streams will contain firmware different from the boards that manage uncompressed data streams. This board will receive data stream configuration data from the CPU to define correspondence between the channels and the compressed data traffic. The current system supports a 4:1 data compression. This means that the boards data stream of 2.048 Mbs compressed can represent maximum the data stream of 8.192 Mbs if all the channels are compressed. The system however allows one to configure 64 Kbs data streams to be clear or compressed dependent on the desired user connections. In this case then the E1 compressed will only be able to manage a data stream less then the maximum. When the system connections are configured the E1 will be provided with information that directs the mapping of the data streams to the corresponding PCM serial lines and channels to which the data is to be connected to. The other end of the connection is with the DSPR-32.

3.4.6 E1 Uncompressed

The MaxPro system supports up to 4 E1 boards each connected to a E1 line of 2.048 Mbs. The function of these boards is to extract inbound E1 data streams and to place the traffic onto the PCM bus. This traffic is then received by the DSPR which determines based on the configuration parameters if the data is to be compressed or uncompressed. In the opposite direction these boards interface with the DSPR-32 data streams which places traffic onto the PCM bus. The DSPR will uncompress data streams that arrived compressed and place the information onto the corresponding PCM bus local identified in the configuration parameters that define the circuit connection between the compressed to uncompressed channels.

3.4.7 External Interfaces

3.4.7.1 E1 to DSPR-32 Interface

The DSPR has to interface with both type of E1 cards managing the flow of data streams both directions. This means that interfaces exist for this purpose. These are identified as the following:

3.4.7.1.1 DSPR-32 to E1 compressed [36]

The E1 compressed interfaces to the DSPR-32 via the PCM bus. The DSPR-32 compresses 64 kbs data streams into 9.6 kbs data streams but added PAD overhead resulting in 12 kbs data streams per channel.

The data arrives in PCM frames each carries 30 bytes of compressed data traffic (30 voice channels). A pair of consecutive bytes in the PCM frame contains 12 bits of data and 4 bits header. The E1 board combines two consecutive 12 bits of payload into 24 bits (3 bytes) of E1 compressed data sent out over 3 consecutive time slot channels. This represents 16 uncompressed channels or ½ of an E1 span. 3 compressed E1 TSI = 16 Uncompressed TSI.

3.4.7.1.2 Input Frame from DSPR-32 via PCM bus to E1 Compressed

000x000x000x000x000x000x000x000x000x000x000x000x000x000x000xSSSS = 256 bits/PCM Frame

0 – Data Payload Nibble , x – PAD/Header Nibble , S – Sync Nibble

3.4.7.1.3 E1 Compressed to E1 Trunk Line

The compressed data stream is combined removing the Header. Each pair of consecutive 12 bits of payload data is combined into 24 bits (3 bytes/3 E1 compressed TSI/channels). This shows that fifteen twelve bit payload data stream is combined into 7.5 groups of of 3 channel data/22.5 E1 compressed channels.

Note: The remaining half of this 23rd channel is reserved for box to box communications that is to be developed in the next phase of the software development.

This leaves 9 channels in the E1 frame, one channel 0 is used already as defined in the standard. The remaining 8 channels are available for clear channel communications. Since channel 16 of each of the four uncompressed E1 boards was not factored in here out of the 8 clear channels 4 are reserved for clear channel 16 communications. This leaves a remainder of 4 channels available clear channel communications.

3.4.7.1.3.1 Output E1/PCM Frame to E1 Compressed Trunk Line

The format of the output to the E1 compressed PCM framer presented to the E1 trunk link is the following

```
000000 000000 000000 000000 000000 000000 000000 000xxx xxxxxx xxxxxx xxxx
  1..3    4..6    7..9   10..12  13..15  16..18  19..21  22..24  25..27  28..30  31..32    E1 Channels
```

0 – Payload Nibble of compressed channels, x –clear channel nibbles (64 nibbles/32 byte E1 frame)

3.4.7.1.3.2 Merging Compressed E1 data with Uncompressed E1 data onto the E1 trunk line - TBD

The MaxPro chassis can be configured to allocate the clear channels of the compressed E1 board to any of the desired 128 of the four uncompressed E1 spans. The configuration commands are used to set the system up for this purpose. As shown only 4 spare channels are available for this purpose.

When the MaxPro System is configured clear channel data is received in a separate PCM frame and combined into the single compressed PCM frame sent out on the E1 Trunk Line used to send/receive

3.4.7.1.4 E1 Compressed to DSPR-32 [36]

In processing data streams coming into the MaxPro system via an E1 trunk line assigned for processing compressed E1 data the format is the same as that used in the output to the E1 trunk line. The E1 frame will contain a data stream with both the compressed and clear channel data. The E1 clear channel data is routed to the E1 uncompressed channels (1..128) bypassing the DSPR-32 decompression processing. The configuration setup is used to define the mapping of the clear channel data to the corresponding uncompressed channels.

3.4.7.1.4.1 Input E1/PCM Frame from Trunk Line to E1 Compressed [36]

The E1 compressed receives E1 frames from another MaxPro system. The format of the input E1 frame is the following:

000000 000000 000000 000000 000000 000000 000000 000xxx xxxxxx xxxxxx xxxx
1..3 4..6 7..9 10..12 13..15 16..18 19..21 22..24 25..27 28..30 31..32 E1 Channels

0 – Payload Nibble of compressed channels, x –clear channel nibbles (64 nibbles/32 byte E1 frame)

3.4.7.1.4.2 Inbound Compressed E1/PCM Frame to DSPR-32

The E1 compressed frame received will be separated into the compressed data and the clear channel data.

3.4.7.1.4.2.1 E1 Compressed Clear Channel Processing - TBD

The clear channel data will be routed to the designated uncompressed E1 channels and defined in the configuration setup.

3.4.7.1.4.2.2 E1 Compressed Frame Data Processing

The E1 compressed interfaces to the DSPR-32 via the PCM bus. The DSPR-32 decompresses 9.6 kbs data streams into 64 kbs data streams. This data is then routed to the corresponding E1 uncompressed channels as defined in the configuration setup.

The data arrives in PCM frames where 22.5 bytes (45 nibbles) of the 32 channel frame represents the compressed data.. This formatted and mapped into consecutive groups of 12 bits of data and 4 bits of header (16 bits) for the DSPR-32.

3.4.7.1.4.2.3 Output Frame from E1 Compressed to DSPR-32 via PCM bus

The format of the compressed data output in a PCM bus frame sent to the DSPR-32 board is the following:

000x000x000x000x000x000x000x000x000x000x000x000x000x000xSSSS = 256 bits/PCM Frame

0 – Data Payload Nibble , x – PAD/Header Nibble , S – Sync Nibble

3.4.7.1.5 DSPR-32 to E1 uncompressed TBD

3.4.7.1.6 E1 uncompressed to DSPR-32 TBD

3.4.7.2 E1 to CPU Interface

The following messages are received from the CPU to manage operations of this board

- SYSCMD_INIT – Message Request to Initialize a Board
- SYSCMD_RESET – Message Request to Reset a Board
- SYSCMD_CONFIG – Message Request to Configure a Board's Configuration Parameters
- SYSCMD_DEVSTART - Message Request to Start a device
- SYSCMD_DEVSTOP – Message Request to stop a device
- SYSCMD_SETLOOPBACK – Message Request to board(s) to go into Loopback Testing Mode
- SYSCND_CLRLOOPBACK – Message Request to board(s) to exit loopback mode
- SYSCMD_PRTCONFIG – Request to fetch Configuration data from other boards

- SYSCMD_PRTSTAT – print statistics request of other boards for perform data on framers performance and alarms
- SYSCMD_CLRSTAT – message request to clear statistics performance tables in the boards

4 System Integration and Testing [25], [26],[30]

In the new system one will need to perform testing at several different levels. The first will be the unit level tests. These test are managed by the developer. To validate software components. After this phase the software will be placed into the configuration management system and the Integration/Testing group will then rebuild the software and perform integration tests. Here the test group can use some of the existing testbed environment from the prior system the 2:1 Digicall system testing system. This environment uses the GL Communications Testing facility which is a PC based system the uses to PC E1 emulation boards. One board is used to send an E1 data stream and another to receive the results. This facility has many features available to support testing. The system however, is not able to handle data communications in the compressed mode. In spite of this drawback, the testbed can be used as an initial test suite to exercise the new equipment configured to operate in the uncompressed mode. In this case, the operational functionality of the channels can be checked out using the existing testbed. After the system is tested using the current testing suite in the uncompressed channel mode, one can then reconfigure the system to operate in the compressed mode. In this case a simple test will be to use a standard communication message that is transmitted though each of the channels via a simple loopback configuration. Each channel pair will be tested to verify that the system operates properly. If communications fails on a specific pair then the task is to isolate further via reconfiguration of the data streams for the pair that fails remapped into two other channels that are known to operate properly this will then isolate the problem to a single channel. At this point, the development team will be notified of the problem to perform further technical analysis and to provide a correction. The software updates will be submitted into the configuration management system and then rebuilt by the test team which will then retest the problem(s) identified.

The testing will entail verification of the operations identified in MaxPro system. At one level the functionality in each of the boards will need to be verified. This is assumed to be the responsibility of the developers. This tests are to be provided to the test group.

4.1 Board Level Testing

4.1.1 CPU Board Level Testing - TBD

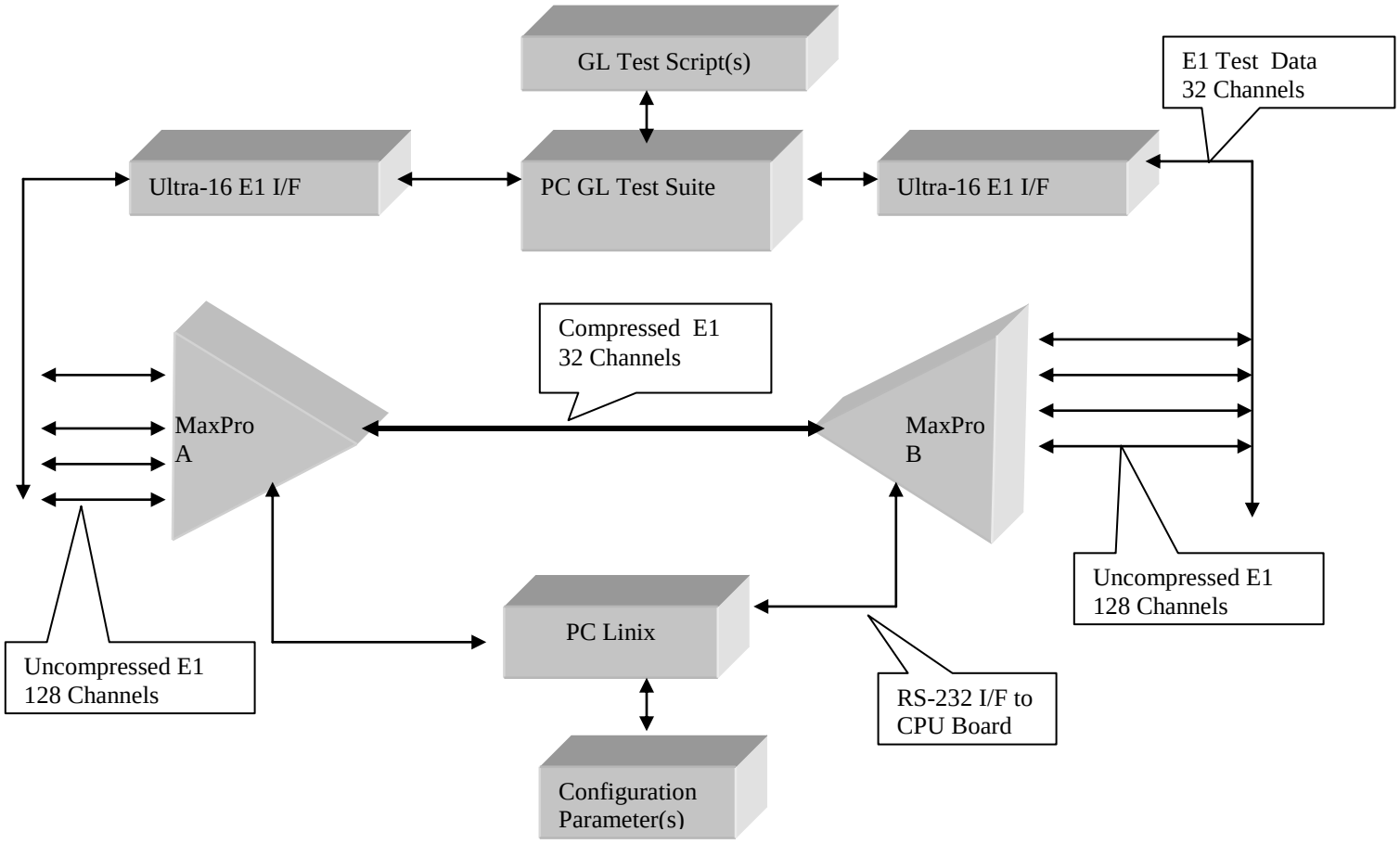
4.1.2 T1E1 Board Level Testing - TBD

4.1.3 DSPR-32 Board Level Testing - TBD

4.2 System Level Testing

4.2.1 Testing Environment and Equipment [31]

- IBM PC with a pair of GL Communications E1 Communication Interface Cards (GL PC) along with the supporting Win/98/NT based vendor software. This product allows the testor to generate tones, signals, record data streams, playback/edit data in files that can be used customize the software generation. The GL Communications software also allows the tester to develop testing scripts using the product's script control language. Scripts can be executed in either a batch or interactive mode. An important feature to note with configuration of the E1 PC Interface boards is that they can be setup to operate in a loopback mode. Also the E1 PC Interface cards (Ultra-16) support bi-directional E1 data streams (2.048 Mbs/sec in both directions).
- A second IBM PC running Linux 6.0 along with Tcl/Tk and Expect (Linux PC). This will be interface to an RS232 port in the MaxPro CPU board. The configuration and setup data will be defined in files which are downloaded from this PC into the MaxPro System. Since the CPU supports dynamic configuration and setup this interface will be used during testing to support the testing operations.
- Two MaxPro Chassis(es) each configured with a CPU Board containing the new firmware, a new DSPR-32 card with the new firmware, 4 E1 Boards with the firmware for processing uncompressed data streams and one E1 Board with the firmware for processing compressed E1 data streams. For most of the testing only one MaxPro Chassis will be needed the exception is Box to Box Testing.



4.2.2 System Level Tests

The system testing will entail testing functional operations of the MaxPro System as defined in the functional operational processing for the first phase of the development. These test will cover testing the following functionality:

- **Tone Generation Testing** – Sending Testing Signal between two MaxPro systems and verification that voice/tone data operates properly with all the communications channels
- **MaxPro Configuration Testing** – These operations will involve exercising the configuration setup testing for the system to verify configuration setup processing operates properly.
- **Failure Condition Testing** – The testing here will entail setting conditions that result in known fails and verify the system handles the conditions properly.
- **Loop Back Testing** – This testing will entail verification of the system to make sure the loopback functions of the system operate properly.
- **Production Testing** – Here the task is to define functional testing to be used to verify the MaxPro system operation as standard testing procedures to be used in the quality assurance verification of the systems produced in the production environment.

4.2.3 Tone Generation Testing – TBD

4.2.4 Configuration Testing - TBD

4.2.5 Uncompressed E1 Channel Communications Loopback Testing

This test is used to verify that the MaxPro system is able to send and receive data streams uncompressed through the E1 spans. This test will involve setting up loopback circuits using the Linux PC Interface to send configuration commands to the MaxPro Chassis. The compressed E1 Board is to be configured to send the data stream towards the external network on portion of span used by one uncompressed E1 and on the other fraction of the span to receive data streams from the network. This board is to be configured in a loopback mode mapping the outbound TSI data streams to the inbound TSI. This will be done via the Linux PC configuration setup. The GL PC will be connect its E1 board interface connections to the uncompressed E1 Boards. At the present time we have only two E1 board simulators in this PC configuration. If we obtain an additional pair then all the data streams in the MaxPro can be tested at the same time. If we are unable to do this then we will test the system using one board to send data streams out on one uncompressed E1 and the other on one of the other uncompressed E1 boards. Once the system is connected and configured, the PC GL testing will proceed. This will test single span of uncompressed E1 traffic in one direction. Next the task is reverse the data stream in opposite direction to make sure bi-directional communications work properly. When this part of the test is done successfully this will verify that the E1 compressed board works in directions when in the uncompressed mode. This will also verify that the DSPR-32 supports uncompressed data communications. At this point however, we have only tested one E1 span of the uncompressed data streams. The Linux PC will then be used to reconfigure the loopback paths to test the remaining data streams available. At this point the GL PC will again test by sending test data streams on the other half of the E1's span and verify the data received by the E1 connected to receive obtains the data properly. Next the GL PC will send the data streams in the opposite direction and verify that portion of the E1 data stream works properly. At this point only 50% of the testing is complete.

The GL PC E1 host interface cards are to then be connected to the other two uncompressed E1 boards and the above steps repeated again. Upon successful verification of the data stream transmission this will verify the MaxPro Chassis operates properly in the uncompressed mode.

4.2.6 Compressed E1 Communications Loopback Testing - TBD

In this testing operation, the DSPR-32 board is to be configured to operate in the compressed mode. The PC Linux will be used to configure

4.2.7 Uncompressed E1 Communications Testing (Box to Box) - TBD

In this case two MaxPro chassis will be required.

4.2.8 Compressed E1 Communications Testing (Box to Box) - TBD

Here also two MaxPro chassises will be required.

4.2.9 Failure Condition Testing – TBD

4.2.10 Production Testing – TBD

4.2.11

5 References

1. Aydin Telecom, Electrical Hardware Specification for DIGICALL DSPR, George Crisafulli, Dec. 6, 96.
2. DSPR Card Framer/Mux/DeMux Software Detailed Design, Review Copy, Alex Khusidman, Michael Grudsky, Dec. 20, 98
3. 4 to 1 E1 Transcoder , Top Level Design, Mike Grudsky, Alex Khusidman, Michael Grudsky, Oct 6, 98.
4. Initial Project Schedule, for Digicall Product Upgrade, Michael Grudsky, 5/30/99.
5. Texas Instruments, Product Description, TMS320C54X, TMS320LC54X, TMS320VC54x, Fixed-Point Digital Signal Processors. SPRS039B, Feb 96, Revised, Feb 98.
6. Mitel MT8985, Preliminary Information, Aug 93.
7. Audio Codes, AC481xxA-C, Voice over Packet Network, Designer's Manual, Revision, 1.10-1, Catalog No. LTRT-00042.
8. Technical Interview with Mike Grudsky, Paul Murray and Alex Khusidman, Eric Krieg at L3 GCN on 6/5/99 with Art Larson Decision Software Systems.
9. RTEMS C User's Guide Edition 4.0 RTEMS 4.0.0 October 1998, On-Line Applications Research Corp.www.oarcorp.com
10. The GMS System – Overview of Wireless Communications
11. RTEMS – "The msg interprocessor communications extension", Paul Murray L-3 Communications
12. CPU Block Diagram, J. Paquet, Aydin, 10-10-95
13. FPGA Block Diagram, J. Paquet, Aydin, 10-3-95
14. Event Flag Usage ,1/5/1999, Aydin
15. Digicall CPU Module Programmer's Model, 10/11/95 Rev. 1, Aydin
16. Digicall CPU Module Description, 6/24/96, Rev. 3
17. Resource Management Module (CPU), Alex Khusidman, Aydin, 1998
18. T1/E1 Card Electrical Hardware Specification Preliminary, Aydin Corp., Jan 8, 1997, Joseph Foran
19. E1 Framer/Transceiver Data Sheet, E1XC, PM6341, Issue 8, June 1998, PMC-Sierra Inc.
20. Meeting ETS 300 011 Requirements with the E1XC, PM6341, Issue 3, March 1998, PMC-Sierra Inc.
21. ICs for Communications, Enhanced Serial Communications Controller, ESCC2, Version 3.2, User's Manual 07.96 Siemens
22. Exploring Expect, Don Libes, O'Reilly & Associates Inc. ISBN 1-56592-090-2
23. Graphical Application with Tcl & Tk, 2nd Edition, Eric Foster-Johnson, M&T Books., ISBN 1-55851-569-0
24. GNU Distribution for Tcl and Tk www.gnu.org Web Site.
25. Scripted Control Language User's Guide, GL Communications Inc., Ver 2.17, April 9, 1999
26. GL T1/E1 Application Development Toolkit for Win NT/95, Programmers Guide and Reference Manual. Version 1.0, GL Communications.
27. DSPR-32 Card, Electrical Hardware Specification for 4 to 1 Transcoder, L-3 Communications, Global Network Solutions, July 13, 1999 Michael Grudsky
28. Data Communications ICs, Enhanced Serial Communications Controller, ESCC2, ESCC8, Siemens, Version 10.94
29. Xilinx, XC3000 Series, Field Programmable Gate Arrays, (XC3000A/L, XC3100A/L0., Nov. 9, 1998 Version 3.1 Production Description
30. Test Plan for 4 to 1 Transcoder, April 2, 1999, Aydin Communications
31. Scripted Control Language User's Guide, GL Communications Inc. Ver 2.17, April 9, 1999.
32. User's Guide for Win98/NT Software, Revised June 29, 1999, GL Communications Inc.
33. RFC 2494 Definitions of Managed Objects for DS0 and DS0 Bundle Interface Type
34. RFC 2495 Definitions of Managed Objects for the DS1, E1, DS2 and E2 Interface Types
35. CCITT G.704, General Aspects of Digital Transmission Systems, Terminal Equipment(s), Synchronous Frame Structures Used at Primary and Secondary Hierarchical Levels.
36. T1E1 Card Compressor DSP, Aydin, 2/19/99, Alex Khusidmang

37. General Aspects of Digital Transmission Systems; Terminal Equipments, Recommendations G.700-G.772, CCITT, Blue Book Vol. III - Fascicle III.4, ISBN 92-61-03341-5, Nov. 1988

6 Glossary of Terms

- AC – Audio Code Chips
- 549 – TI TMS320C549 Digital Signal Processing Chips
- CAS – Channel Associated Signaling (Applies to E1 only uses TSI 16)
- CCS – Common Channel Signal (Applied to E1 Only used TSI 16)
- CRC-4 – Four Bit Cyclic Redundancy Check used in E1 Communications
- IPC - InterProcessor Communications
- DSPR – Digital Signal Processing (DSP) Resource Card
- DSP – Digital Signal Processing
- FAS – Frame Alignment Signal
- FPGA – Functional Programmable Gate Array
- HDLC – High Level Data Link Control
- Kbs – Kilo Bits Per Second
- Mbs – Mega Bits Per Second
- Nibble – 4 bits
- Network to User – Data Stream sent into the PCM backplane from a board
- OAM – Operations, Administration, and Maintenance
- Octets – 8 bits/1 byte
- PAD – Packet Assembly/Disassembly
- PCM – Pulse Coding Modulation
- RF- Radio Frequency
- SPAN – 2.048 Mhs Data Stream (Bandwidth of an E1 Board – 32 channels @ 64 Kbs)
- TBD – To Be Defined
- TRX – Transmit/Receiver
- TSI – Time Slot Interchanger refers to Mitel 8985 chips.
- User to Network – Data Streams coming into a board from the PCM backplane