A System for Multiple Channel Mixing and Processing of Digital Audio

Mark Rudolph
GOTHAM Signal Arts Corporation
New York, NY 10019, USA

3135 (D-3)
5M/D-3

Presented at the 91st Convention 1991 October 4–8 New York

AES

AN AUDIO ENGINEERING SOCIETY PREPRINT

A System for Multiple Channel Mixing and Processing of Digital Audio

Mark Rudolph
GOTHAM Signal Arts Corporation
New York, New York

I. Abstract

The GOTHAM Signal Arts Sigma-DSP3[1] is an integrated environment for mixing, equalization and dynamics processing of up to 60 digital audio channels and is intended for use in professional music and post production studios. The system design and application software grew from preliminary multi-processor music synthesis work at AT&T Bell Laboratories and continued at GOTHAM Signal Arts Corporation. The system consists of a custom version of the DSP3 ([1], [2]), a massively parallel multi-processor developed at AT&T Bell Laboratories, a high fidelity digital interface to multi-track digital tape machines, an Ethernet connector allowing control information to be sent from existing scanned "moving fader" mixing consoles or computer "virtual" consoles, and a secondary processor for spectral analysis and display. The DSP3 is an array of up to 128 processing nodes, each node consisting of an AT&T DSP32C floating point processor, 256 Kbytes of zero wait state static RAM and a special communications ASIC also developed at AT&T Bell Laboratories. A custom operating system and processing software have been developed for the DSP3 by GOTHAM Signal Arts to allow the necessary communication bandwidth and processing power to handle up to 60 channels of digital audio.

II. Introduction

The Sigma-DSP3 can be thought of as a massively parallel Ethernet device with special audio interface capability (Figures 1 and 2). The system consists of an optional Sun 3 workstation as a software development platform and supports the use of the Sun Network File System for loading of executable files. Also connecting the Sun to the Sigma-DSP3 is a serial connector so that information about the boot-up can be monitored in a Sunview window. The architecture is based around the VME bus and there are spaces for six 6U VME cards of which two are standard, an Excellan Ethernet card and a Plessey 68030 "realtime host" (RTH) card. A second "VSB" bus connects the parallel ports of all DSP32C processors and allows the RTH to read and write directly to and from local DSP32C memory via parallel DMA. A large hard disk is used for storing start-up and application code and allows the Sigma-DSP3 to operate as a "turnkey" system without the Sun. The processor boards are 9U VME cards and each contains 16 nodes. A node consists of one AT&T DSP32C floating point processor, 256 Kbytes of zero wait state RAM and a special communications ASIC. A system uses 3 to 5 processor boards depending on whether it is a 32, 44 or 60 channel version. There is also one 9U "IOS" board which handles packet communications between the RTH board and the processor boards. Finally, the left and right monitor channels are sent via digital interface to a DSP32C card in a a PC with a high resolution VGA monitor. This subsystem does realtime 8K FFT analysis at 10 frames per second and displays the results using log frequency and log amplitude scaling in either 2D or 3D.

The networking capability of the Sigma-DSP3 is based on its ability to create TCP/IP sockets capable of servicing one or more clients such as a processor located in a scanned mixing console or computer virtual console. The system is controlled by messages conforming to a special four byte protocol. The control philosophy adopted is that all of the digital processing and mixing be controllable from familiar analog mixing consoles and that any console capable of connecting to a pre-existing TCP/IP socket be able to control the Sigma-DSP3 by use of a standard message protocol.

The audio interface allows connection to multi-track digital tape machines of up to 48 channels. Also provided is a user specified array of output interface modules allowing communications with processing units or master recorders using analog or any available digital format such as AES/EBU, SDIF, S/PDIF, PRODIGI, JVC format or Yamaha format. The internal data format is 32-bit floating point which provides an internal numerical dynamic range of 1500dB. At the output the master mix and monitor mix are redithered to 16-20 integer bits by dithering modules.

The internal communications scheme of the system is orthogonal and consists of an Ethernet control stream, serial audio input and output directly to and from each node, and packet communications from node to node on a 160 Mbyte/sec. "NPI" bus formed by the current

1 *Sigma-DSP3 is a Trademark [™] of GOTHAM Signal Arts Corporation*

configuration of nodes. The topology used in the Sigma-DSP3 is a simple one directional linear array.

There are three versions of the system which accommodate 32, 44 and 60 channels respectively and are intended for use with existing 24 channel, 32 channel and 48 channel digital tape machines. The processing power of the Sigma-DSP3 ranges from 1 Gflop in its 32 channel *version to 1.64 Gflops in its 60 channel version.*

The software environment is hierarchical (Figure 3). The top layer is Sun/Unix with c-compilers, c++ "front", and DSP32C assembler and simulator. This highest layer is used for software development only and is not part of the final system. The next layer is the Vertex operating system running on the RTH 68030. This layer handles Ethernet communications, booting, loading, configuring, and starting and regulating application procedures in the nodes. Vertex is multi-tasking and features a real-time clock, semaphores, mailboxes and queues for interprocess regulation. The lowest software level consists of the mixing and processing code and a custom operating system kernel of packet communication and topology switching routines used in inter-node communications. In the case of the Sigma-DSP3, all code running on the nodes is written in assembler for maximum speed.

III. Control

The intended control of the Sigma-DSP3 is the set of mixing faders and EQ and dynamics knobs on a scanned analog mixing console, or the movement of a cursor on a high resolution graphics screen of a computer virtual console. The method used is to send four byte messages over a TCP/IP Ethernet stream socket. The Sigma-DSP3 has an Excellan Ethernet VME card and an Ethernet transceiver. The physical connection to the system is a BNC "thinnet" cable to the transceiver. The Sigma-DSP3 has a unique Internet address and creates the server side of a TCP/IP socket during initialization after power-on. Clients on Ethernet need only create their end of the socket and connect to the already existing Sigma-DSP3 socket. Usually there are local operating system functions to accomplish these actions. For example, in Unix the functions are socket() which returns a socket descriptor, and connect() which establishes the software connection using supplied information about the server Sigma-DSP3. Once the socket is established the sending of control messages to the Sigma-DSP3 is exactly analogous to writing data to a file. The function call is identical but uses the socket descriptor in place of a file descriptor. Error checking, buffering and requests for resend are handled invisibly as part of TCP/IP. The Ethernet bandwidth is ten megabits per second so is more than adequate for the stream of fader and knob control information. The system expects four byte messages which communicate channel number, function type and parameter value.

A process running on the real-time host reads incoming messages, translates them into *information for the nodes and writes this information directly into the local memory of the* target node or group of nodes. An arbitrary number of bytes can be written and, in some cases, a semaphore is set in order to prevent the node processor from reading part of a previous data set and part of a new set.

IV. Determinism Via Serial Input Counter

An important technique for making any application more efficient is to make it deterministic. Then there is no need for case testing and the use of a bus can be precisely planned for maximum efficiency. For mixing in particular, the send of each channel packet can be made to occur at a precise time when the bus can be guaranteed to be free, and the arrival of packets at a mix node can be made to occur in pre-determined order so no checking of their source is necessary and mixing scalers can be read from a simple incrementing pointer.

The easiest way to achieve determinism is for all nodes to have an absolute time reference so actions can be scheduled globally to permit precise ordering of events. The DSP3 has no system clock accessible by all nodes. Therefore, one clock taken from the multi-track digital tape machine is used to to control all audio input and output, and each node keeps a local synchronized count based on the arrival or departure of samples on the serial port. Nodes in the system have two general functions. Channel nodes do equalization and dynamics processing, and mix nodes scale and accumulate samples sent from the channel nodes. The basic processing and mix unit is the packet which consists of 70 audio samples, i.e., 1.58ms duration. Channel nodes count samples by counting serial interrupts. Channel nodes always have three packets within their workspace. Packet N-2 was received from the serial port two packet "periods" ago and was loaded into the output FIFO in the previous packet period. In the present packet period packet N-2 waits to be sent to the mix nodes. Packet N-1 is the packet accumulated from serial interrupts during the previous packet segment and is now processed during the non-interrupt periods of the present packet period. Packet N is being accumulated

from the serial interrupts during the present packet period. As each new sample arrives at the serial input buffer an interrupt is generated and within the interrupt service routine a counter is decremented by one. During each packet period the counter is decremented from its initial value of 70 down to zero, at which point the counter is reset to 70 again.

Associated with each channel is a unique sample number used as a packet send "trigger". Within each jump to the serial interrupt service routine the sample counter is decremented and tested to see if it equals the trigger number. If it does, the node first switches configuration to connect its output FIFO to the NPI bus. Then the node sends packet N-2, already loaded in the output FIFO during the previous packet period, to the mix nodes which expect it. In the case of 60 channels, the trigger numbers range from 68 for channel one down to 9 for channel 60. Channel K sends its packet when the sample counter reaches 69-K. The amount of time required to switch configuration and send a packet is significantly less than the 22.675ms between the arrival of the present sample and the next sample. Thus, no interrupts are undetected and hence each channel node sample counter is always accurate. Furthermore, since the samples from all channels arrive at the serial ports of the respective channel nodes at the same time, all channel nodes have identical counter states. Thus, since each channel node completes its packet send within one uniquely assigned packet period, no packet collisions can ever occur and the channel number order of the packets received at the mix nodes is determined.

Mix nodes also keep globally synchronized sample count but in a different manner. The DSP32C features a simple serial DMA capability. With serial DMA enabled, two registers are dedicated to use as serial input pointer and serial output pointer. When a sample is either taken from the output pointer location and placed in the output buffer or arrives at the input buffer and is loaded into the input pointer location, the appropriate pointer register is post-incremented. The range of addresses the pointer takes corresponds to the sample counter values in the channel nodes. For example, at sample counter 70, the pointer is located at some start address A, and when the sample counter decreases to zero the pointer has advanced incrementally to address A + 70*4 (all internal samples are four byte floats). Thus, the location of the pointer can be used to determine what sample count is in the channel nodes. This correspondence between pointer address and sample counter allows the pointer register to be tested against specific addresses in order to trigger mix events which are synchronized with channel node events. In particular, the final "barrier" address (A + 70*4) is used to test if a new packet period has begun and the pointer needs to be reset.

Two further aspects of system timing need to be discussed. First, there is an initial synchronization to ensure that all nodes, both channel and mix, receive the same first "wave" of samples which trigger counter decrements and pointer increments. A random unsynchronized start might occur just at the instant a wave of samples arrived at the serial ports. Minor time skew due to path length differences might cause some nodes to sense a sample arrival while others might not. Those nodes which did not detect a sample arrival will have missed one interrupt or DMA trigger relative to those that did detect a sample arrival, and hence all local counters would not be synchronized. The initial synchronization is achieved by the remote host signalling node number one by writing to a special memory location in node one's local memory. After node one is signaled it waits for a sample to arrive. After the sample arrives (condition input buffer full is set) node one broadcasts a header-only packet to all nodes which causes them to turn on serial interrupts in the case of channel nodes, or turn on serial DMA in the case of mix nodes. After broadcasting, node one also enables its serial interrupt. These actions are achieved in well under the 22.675ms sample time, i.e., well before the next wave of samples. Thus, all nodes are listening for the same approaching wave of samples so that their local counters or pointers are globally synchronized. Finally, a point must be made concerning configuration (Figure 4). A node in the linear array can be configured either with input FIFO enabled to allow packet reception or packet pass-through (configuration one) or with output FIFO connected to allow packet send (configuration two). An attempt to send a packet to or through a node in configuration two, i.e., configured with output FIFO connected, will result in failure. Thus care must be taken to ensure that the path to the target mix nodes consists of nodes all in configuration one. The following method ensures that each node sends its packet only when all nodes between it and its target are in configuration one. First, upon counter reset to 70, all nodes are in configuration one. The linear array is implemented so that the mix nodes are all "to the right" of all channel nodes, and channel node one is furthest "to the left" in the array (Figure 5). At counter time 68, channel node one switches to configuration two and sends its packet. Then at counter time 67, channel node two switches to configuration two and sends its packet, and so on. In each case, all nodes "to the right" of each sending channel node are in configuration one, and all nodes "to the left" are in configuration two. Thus no blocking can occur so all packet sends are successful. Finally, at sample counter 8, all

channel nodes have sent their packets so it is safe them to switch back to configuration one in preparation for the next packet period. Once again, reference to a global sample counter allows safe, predictable deterministic system behavior.

V. Mixing

Mixing is achieved by the scaling and accumulating of each channel packet in the mix node to form a mix packet. The realtime host keeps track of fader and pan positions and on/off information. Whenever a change to any of these parameters occurs a new mix scaler is calculated for the appropriate channel and written to the mix nodes via parallel DMA. A packet sent to a mix node is received in the input FIFO, and each of the 70 samples is multiplied by the current mix scaler for that channel and accumulated in a mix buffer.

The nodes assigned to the task of mixing include those mixing the master left and right outputs, the monitor left and right outputs and the auxiliary mix outputs. The number of nodes assigned to mixing is based on the ratio of the time to send a packet on the bus to the time taken unloading the current channel packet from the FIFO. This ratio is approximately 0.5 to 1, i.e., the packet sends are twice as fast as the FIFO unloads. Thus, in order to fully utilize the bus time segment two mix nodes are used to perform each single channel mix. Odd numbered channel nodes send their packets to the "odd" mix nodes, and even numbered channel nodes send their packets to the "even" mix nodes. In this manner, the bus is fully utilized. Channel node packet sends occur at each sample count whereas mix node FIFO unloads are overlapped by one sample count, odd mix node FIFO unloads starting on odd sample counts and even mix node FIFO unloads starting on even sample counts. The alternation of FIFO unloads between odd and even mix nodes allows sufficient mix processing time since each FIFO unload, scaling and accumulation of a packet requires almost two sample counts to complete. Also, as mentioned before, the order of the channel packets received is always 1, 3, 5, 7, etc. in the odd mix nodes and 2, 4, 6, 8, etc. in the even mix nodes. Therefore, scalers for each distinct channel packet can be read from an incrementing pointer.

Odd and even mix nodes are adjacent to one another, the even node one position further "to the right" in the linear array than the odd node. The mix nodes for a 60 channel system include the following, in order left to right: auxiliary mix one odd, auxiliary mix one even, auxiliary mix two odd, auxiliary mix two even . . . auxiliary mix six odd, auxiliary mix six even, monitor left odd, monitor left even, monitor right odd, monitor right even, master mix left odd, master mix left even, master mix right odd, master mix right even.

After all channel nodes have sent their packets, i.e., sample count 8, the odd mix nodes switch to configuration two, i.e., connect their output FIFOs, and send their accumulated "odd channel mix" to the respective even mix node "mate" which is adjacent. No blocking results since the send is to a nearest right neighbor. The even mix nodes accumulate the full channel mix, left, right or auxiliary as the case may be, and leave it in a buffer to be removed by DMA in the next packet period. After sending the accumulated odd channel mix to its even neighbor, the odd mix node switches back to configuration one, receive or pass-through, to be ready for the next packet period sends. This switch is completed well before the sample counter reaches zero and hence, once again, no blocking ever occurs.

Each odd or even channel node is able to broadcast to all odd or even mix nodes because each node is allowed to have up to three logical bus addresses. Thus each node can be assigned its ordinal number in the linear array, and all odd mix nodes can be assigned one "group ID" while all even mix nodes can be assigned a second group ID. These ID numbers are written to a specific location in the packet header when the packet is built. The ID is used by the communications ASICs to determine whether a packet is to be loaded, loaded and passed on, or simply passed on.

A summary of the sequence of events within each packet period can now be specified for an example 60 channel system. First, channel node one switches to configuration two and broadcasts its packet to all odd mix nodes at sample counter 68. Next, channel node two switches to configuration two and broadcasts its packet to all even mix nodes at sample counter 67. This procedure continues until finally, channel 60 switches to configuration two and broadcasts its packet to all even mix nodes at sample counter 9. Then at sample counter 8 all channel nodes switch back to configuration one, and all odd mix nodes switch to configuration two and send their odd channel mix accumulations to their corresponding right neighbor even mix nodes which accumulate the final mix packets to be sent by serial DMA during the next packet period. Next, all odd mix nodes switch back to configuration one. Finally, when the sample counter reaches zero (or the DMA pointer reaches its barrier address) all node states are incremented and all buffer pointers are set according to the new state, and the sample counter is reset to 70 for the next packet period. The overall mix and monitor system delay is

three packet periods or approximately 5ms. The overall auxiliary mix system delay is about twice that or approximately 10ms.

Additional features of the mix are cut and solo. Cut removes a specified channel from the monitor mix and solo removes all channels except a specific channel from the monitor mix. Cut and solo are applied at the mix node so the application delay is equal to the console scanning delay plus the Ethernet/DMA delay plus two packet periods, i.e., less than 10ms.

VI. Processing

The processing done in each channel node is modeled after the processing section of the Neve VR console. The only difference is the addition of an adjustable delay line in each channel to allow the synchronization of input transients. An example of its use is when differing microphone distances must be compensated for when an ensemble of instruments is recorded in a hall using multiple microphones. Following the delay line the input has a trim control to scale the amplitude of each channel input. Next, there is a phase button which inverts the phase of the input waveform from zero to pi or vice versa. Next there are two frequency adjustable cut filters. The frequency range of the low cut filter is 31.5Hz to 315Hz. The frequency range of the high cut filter is 7500Hz to 18kHz. These filters have 0dB gains in the pass band and roll-off at a slope of -12dB per octave. Next is the dynamics processing section, about which more will be said later. Finally, there is the equalization section. This is a structure of four filters, low, high, mid1 and mid2. The low and high filters can either be set to be shelving filters having gains + 18dB to -18dB, or boost filters having ± 18dB gains and Q values of 0.71 or 2.0. The frequency range of the low filter is 33Hz to 370Hz. The frequency range of the high filter is 1500Hz to 17kHz. The mid filters have ± 18dB gains and Q values of 0.5 to 9.0. The frequency range of the mid1 filter is 190Hz to 2kHz, and that of the mid2 filter is 800Hz to 8.7kHz.

All the filter coefficients are stored as sets of five floating point numbers in the 3Mb of user memory on the RTH board. When a console filter knob is moved a new pointer into the filter coefficient table is computed. Then a block of data is sent via DMA to the local memory of the appropriate channel node. The five coefficients are written to the new coefficient buffer and a semaphore is set to indicate that a change has occurred.

The dynamics processing section is between the two cut filters and the four equalizing filters. The section consists of a compressor/limiter and a gate/expander. The advantage of working

in the digital domain is that present dynamics processing decisions can be made contingent upon the future states of the signal because an entire packet of "future" samples is always available for inspection. Compression is implemented when the input has exceeded a specified amplitude threshold for a specified length of time. When the compression ratio is set at K:1, the portion of the signal to be compressed is decreased in amplitude so that its peak dB level above threshold is K times lower than its original value. For example, if the ratio is 2:1, a signal with peak at 6dB above threshold is reduced to a signal with peak 3dB above threshold. Limiting is compression with a ratio of infinity to one, i.e., clipping. Expansion is the inverse of compression, i.e., a signal with amplitude below a specified threshold is increased so that its new lowest amplitude value is K times higher in dB relative to the threshold than previously. Gating decreases any portion of a signal below a specified threshold level to a new level a specified dB "depth" below the threshold. A second "unmute" level above the threshold can be specified which is the level a signal must attain again before gating is halted.

VII. Analysis

In order to aid the audio engineer in the task of equalization and signal processing in general, the stereo output of the monitor mix is sent as a digital stream to an analysis subsystem. The monitor mix is identical to the master mix unless a solo button is pushed in which case the monitor mix is simply the solo channel (useful for the EQ of a particular channel). The analysis system consists of a PC with VGA monitor and a card containing a DSP32C, 256Kbytes of RAM and a digital interface.

The spectral analysis consists of the following. The left channel, right channel or channel sum is windowed and an 8K FFT is performed ten times per second and displayed in either 2D or 3D graphical form along with the present SMPTE timecode. The linear resolution of the FFT is approximately 5Hz and there are three selectable frequency scales: 20Hz to 20kHz, 2Hz to 2kHz and 0.2Hz to 200Hz. There are also three selectable log amplitude scales: -60dB to 0dB, -90dB to 0dB and -180dB to 0dB. In addition to the graphical display, audio segments can be analyzed and that information captured to hard disk. When a capture is made, the spectral amplitude at each of 512 log frequencies is recorded for each 100ms frame. If the cursor is moved to any log frequency, the SMPTE time corresponding to the occurrence of the amplitude peak in that band is displayed. Then the captured graphics information can be replayed on the screen and halted at the occurrence of the selected peak in order to observe

the spectral amplitudes in the neighborhood of the peak. In addition, a sound segment can be captured to disk file A and the same segment processed can be captured to disk file B. Then each can be averaged and a difference displayed in order to show the frequency response of the processing. Thus the spectral analysis subsystem can be used as a graphical aid to EQ processing and as a test system for revealing the frequency response of exterior processing units such as de-essers, reverberation and other effects modules.

VIII. Conclusion

The GOTHAM Signal Arts Sigma-DSP3 is a high fidelity digital mixing, EQ and dynamics processing system for professional music recording and post-production studios. It can accommodate up to 60 digital audio channels and is designed to interface to existing 24, 32 and 48 channel digital tape machines, and be controlled by the faders and knobs of any scanned analog console or computer virtual console via a simple Ethernet connection and message protocol.

References

[1] R.R. Shively, E.B. Morgan, T.W. Copley and A.L. Gorin, "A High Performance Reconfigurable Parallel Processing Architecture", Proc. Supercomputing '89, November 1989.

[2] R.R. Shively and A.L. Gorin, "A Reconfigurable Fault-Tolerant Systolic Signal Processor", Proc. ICASSP '89, May 1989.

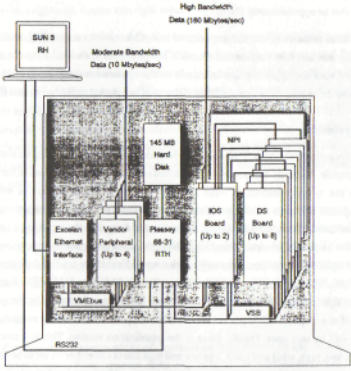

Figure 1: DSP3 Architecture

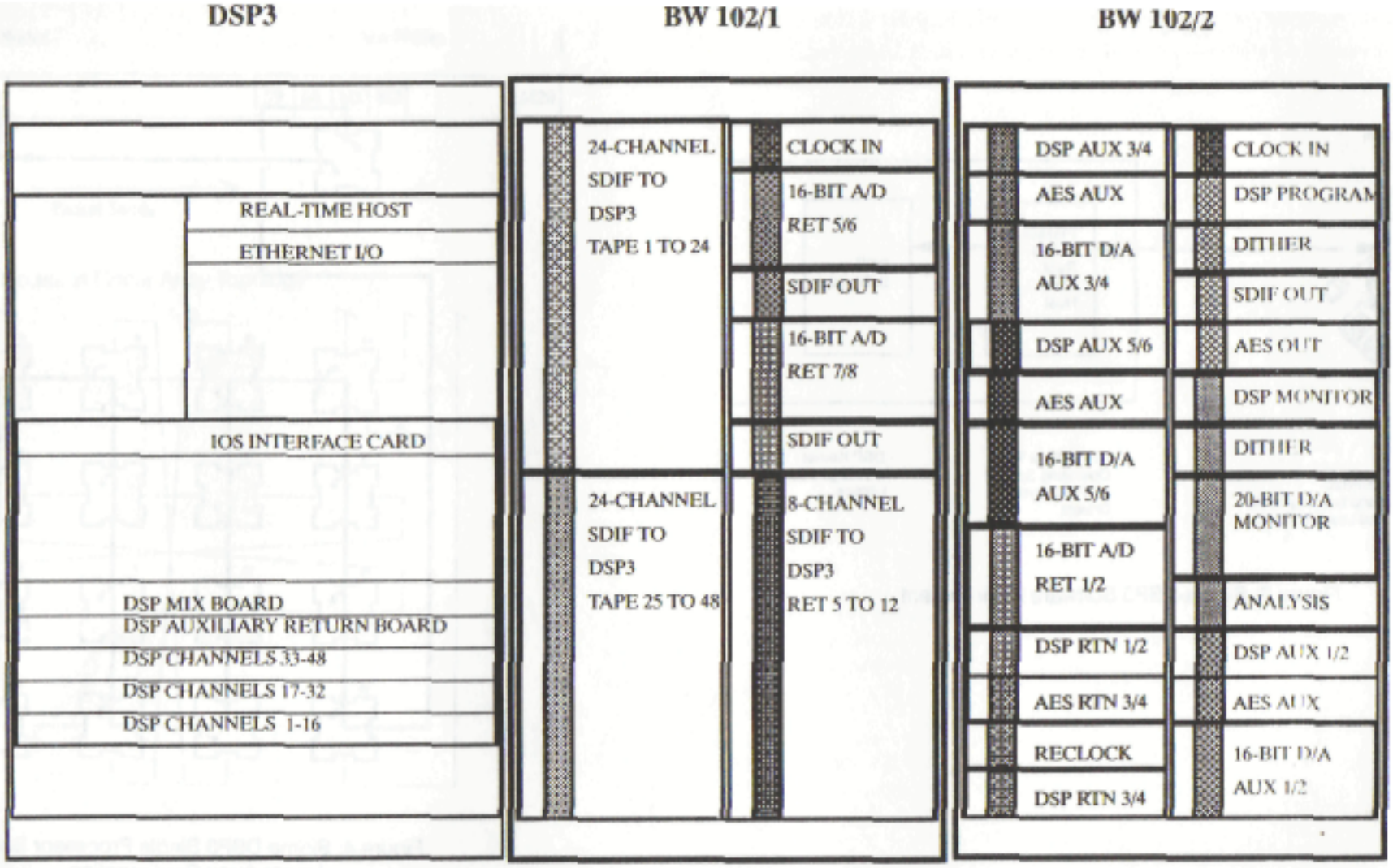

Figure 2: Sigma-DSP3 Architecture

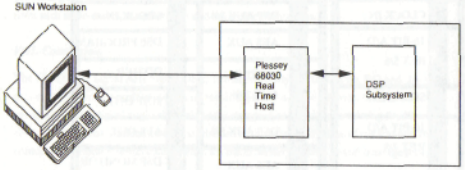

Figure 3: Sigma-DSP3 Software Environment

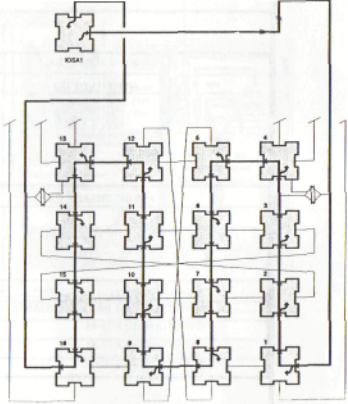

Figure 4: Sigma-DSP3 Single Processor Board
Linear Array Configuration

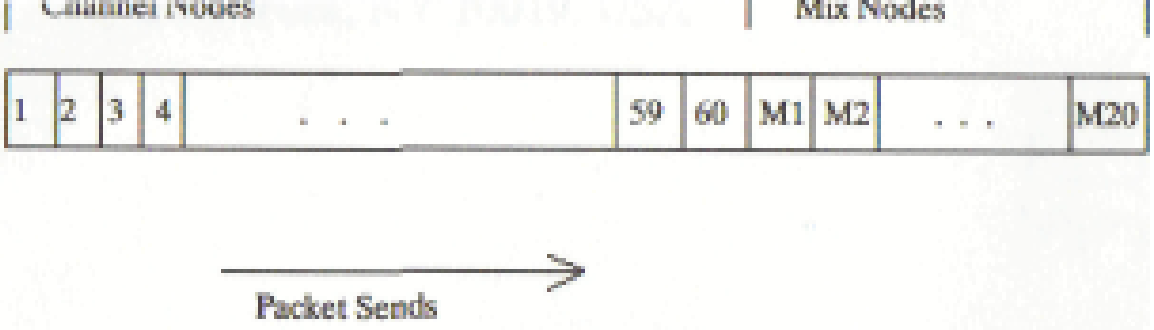

Figure 5: Nodes in Linear Array Topology