BUS STRUCTURED SOFTWARE FOR A MODERN PCM DECOMMUTATOR

MICHAEL A. CRAWFORD
PRINCIPAL ENGINEER

RALPH F. SWEITZER
PRINCIPAL ENGINEER

LORAL DATA SYSTEMS
SAN DIEGO, CALIFORNIA

ABSTRACT

The expanding requirement in Modern Telemetry Systems for Real-Time Data Processing has necessitated the commutation of a vast majority of the data processing functions into Front End Processors. Even the fastest of Host Processors has proven incapable of keeping pace with high speed data rates (up to 4 Megawords). The commutation of processing power into the telemetry front end has elicited the employment of distributive processing techniques in order to attain the desired throughput.

A distributive processing system architecture achieves high processing throughput by apportioning data analysis functions. By defining and programming unique processing nodes to selectively acquire, distribute, compress, and/or convert data, extensive simultaneous operations are executable. Hardware merged bus structures have lent themselves conveniently to this method of data distribution and control. Conversely, conventional software structures are unsuited to distributive processing architectures which must support a broad spectrum of modular configurations. Primarily, this is evidenced when the composite system software must be repetitively customized as additional processing power or new capabilities are incorporated. Composite software that delivers a high degree of system configuration adaptability is nominally large and complex, is limited in application, depletes system memory resources and complicates sustaining software maintenance. In addition, an undesirable human interface is normally unavoidable with composite software since it requires that the user learn the specific front end system's terminology and individual components.

Bus Structure Software consigns itself to effectively support distributive processing techniques providing for adaptive system configurations. This disquisition will address the concepts of bus structured software and its application to distributive processing. Furthermore, this paper will discuss the architectural capability to service a wide range of telemetry users without specialized system tailoring. A typical implementation of this

convention, the Advanced Decommutation System (ADS) designed by LORAL DATA SYSTEMS, San Diego, California will also be presented.

CONCEPT

Bus Structured Hardware has quickly become the industry standard within the last decade. The concept of a software bus architecture has not evolved as quickly. This conceivably is due to the ambiguity as to what constitutes a software bus and in what application this architecture would be advantageous.

A bus by definition provides an intermediary medium for transferring information and processing control. A software bus, in this context, can be viewed as an inter-connecting link between the system resources and the functionally dissimiliar application modules. A system software architecture centered around Bus Structured Software is capable of determining the unique system's modular configuration during initialization (system generation). As additional processing power or new capabilities are required they are simply added and become part of the composite system during the system generation process.

The dimension, size, and capability of the software is highly adaptive in that various application software modules can be combined, added, or deleted to formulate a composite system configuration that is user application oriented. Within a bus architecture, application modules designed for decommutation and related purposes need not be an integral part of the basic system package yet can share system resources with and function independently of other modules within the system.

MATHEMATICAL MODEL

A comprehensive model for bus structured software designed for a telemetry decommutation application consists of two distinct entities; (1) a system package and (2) a collection of Software Application Modules (SAM's). The system software is an autonomous body designed for no specific configuration, that provides all the fundamental facilities and utilities required to accommodate a variety of data acquisition, processing, management, distribution, and display processes. Conversely a SAM, serving as a processing node, is a personality module designed to provide a specific telemetry or analysis function required for the user's system application.

System Software

The general purpose system software consists of a batch of facilities designed to be resident and operate independent of the specific system configuration. The nucleus of the system software is the operating system.

The functions of the operating system are (See Figure 1)

- System Generation
- Task Management
- Memory Management
- Command Processing
- Display Processing/Operator Interface

Inclusive within the system software are generic facility and utility modules required in typical data processing applications:

- Data Analysis and Evaluation
- General Purpose Routines
- Math Processing
- Task Scheduling
- Standard Drivers

Software ApplicatioD Modules (SAM)

The system utilization is derived from the selected coalition of application modules. Individual SAM's formulate the system's configuration to the specific user's requirements. A combination of both hardware and software is normally required to accomplish any specific function. In this framework, SAM's typically provide the programming, operator displays, and control for each of the fundamental hardware application modules. For example, In decommutation applications, these modules could provide for bit synchronization, frame synchronization, data distribution, data compression, etc. Thus, each processing node is responsible for programming itself according to its own specific design and application, and provide operator controls and monitoring using the inherent system resources.

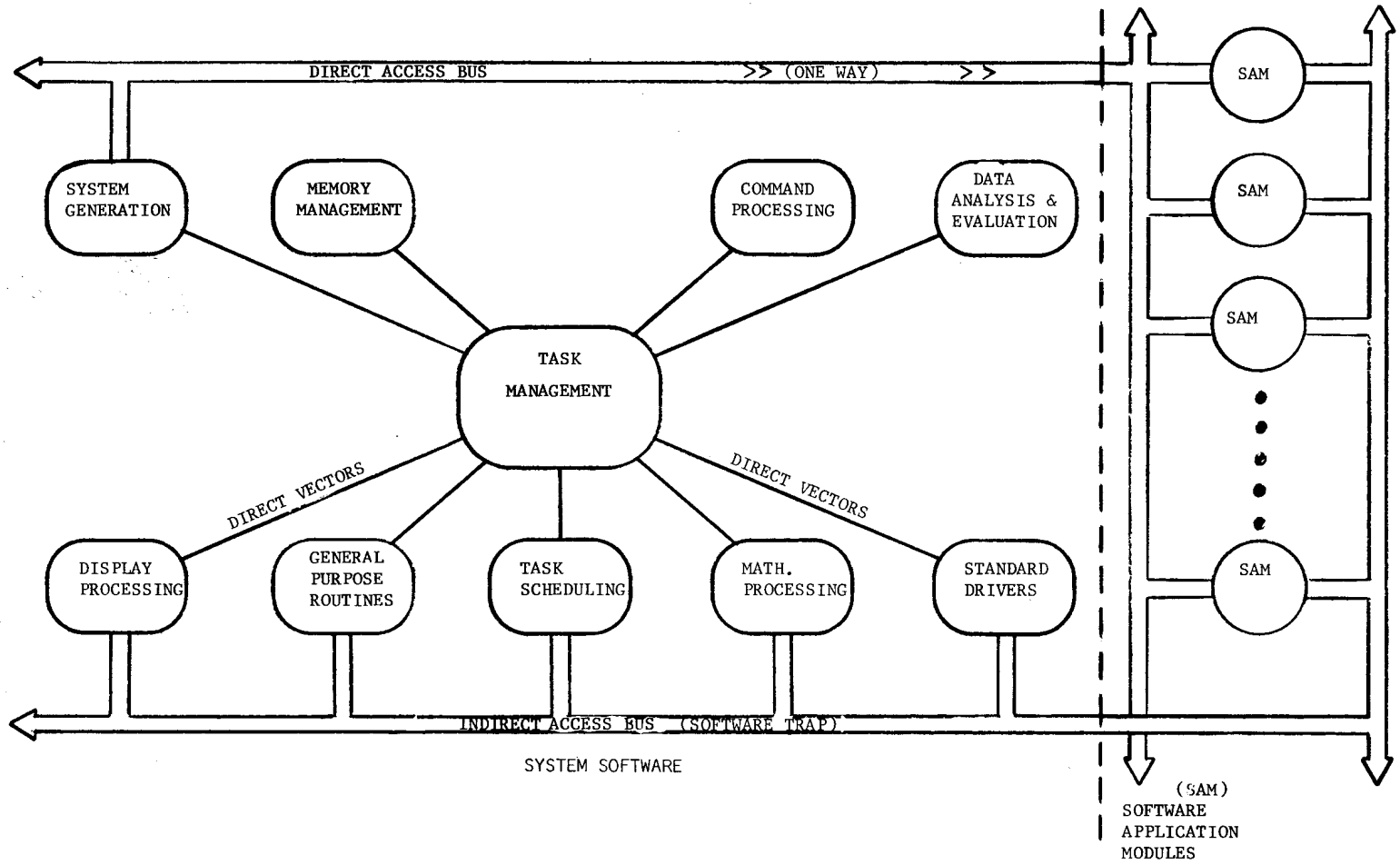

FIGURE 1. BUS STRUCTURED SYSTEM ARCHITECTURE

Software Busses

Application modules and system software communicate over two types of busses:

a) Direct Access
b) Indirect Access

The system generation function is accomplished over the direct access bus. During system generation the entire memory mapped spectrum is searched for the presence of application modules. As each module is recognized, control of system facilities is released to allow that module to complete initialization requirements. In addition, the module specifies its unique memory requirements, allowing random access memory (RAM) within the basic system package to be dedicated to this module's use. System generation is continued until all modules have been identified, logged, and initialized. SAM's can not directly access the operating system, however, for system generation or other similiar type functions the operating system directly accesses the application modules.

Once initialized intercommunication is accomplished over an indirect access bus. During initialization all modules requiring access to system facilities on a single or repetitive type basis must schedule themselves using the system task scheduler. A Scheduling queue is used for task management. The task scheduling and additional system resources are available using a software trap. This allows a module to call a resource without the knowledge of its location, eliminating the requirements for module and system interlinking.

PROCESSING POWER/FLEXIBILITY

Bus Structure Software derives its maximum processing power and flexibility when combined with a hardware bus to formulate a total distributive processing architecture (See Figure 2). The Operating System and facilities operate in the foreground providing for system generation, initialization, operator interface, and for the interactive programming/reprogramming of application modules. In the background, application modules functioning as processing nodes acquiring, gathering, merging, processing, converting, etc., raw data in real-time at hardware processing speeds. Applying this distributive processing technique, data throughput can be maintained independent of foreground processing speeds.

As application modules are added or deleted, the system generation function expands or compresses to constitute the total system configuration. This allows a single operating system to accommodate an extensive variety of data analysis requirements without modification. In addition, the system configuration is defined by the number and types of specific application modules without user intervention. Module or operating system

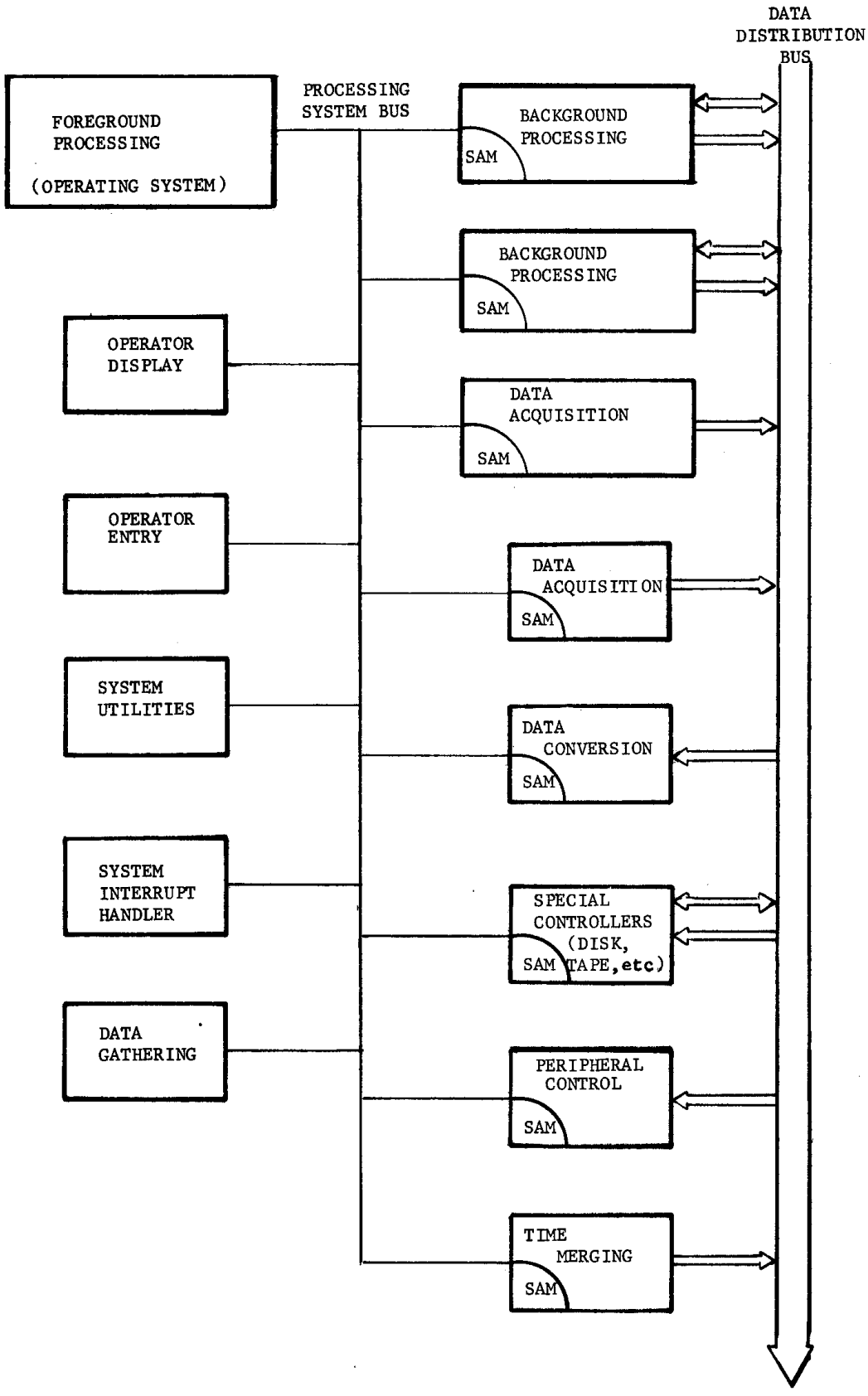

FIGURE 2. DISTRIBUTIVE PROCESSING ARCHITECTURE

software is easily changed independently; not requiring a modification to other software elements. Memory requirements are also minimized since it is based upon the given user configuration. This results in reduced costs and eliminates, in many cases, software and/or hardware the user has no requirement for. In addition, a bus structure software architecture virtually eliminates the need for users to understand the specific systems terminology and hardware/software components. This method of auto-system generation is also highly adaptive to production line fabrication and testing.

PREPROCESSOR APPLICATION

To exemplify the convention of bus structured software within a telemetry system, the typical application of a PCM Decommutator Preprocessor has been selected. The functions of the preprocessor are extensive with the intent of achieving high data throughput while reducing data processing and overhead requirements of the host/mainframe computer system. The Advanced Decommutation System (ADS) designed by LORAL DATA SYSTEMS, San Diego, California delivers such preprocessing capabilities.

The ADS is a complete data acquisition and real-time processing system designed to meet the data analysis requirements of telemetry ground stations. The unit combines the functional elements of the decommutator with a new distributed architecture processing system. The system functions as a complete small scale ground station or as a real-time preprocessor for a major installation. As a stand-alone system the unit provides the capability to acquire, distribute, limit-check, compress and display the results of flight test measurements. These capabilities can also be applied as a preprocessor for a host computer.

In this application the unit serves by relieving the host computer of many real-time processing requirements allowing a more effective, application specific workload. ADS functional characteristics include:

- Bit Synchronization
- Frame Synchronization
- Subframe Synchronization
- Data Distribution
- Data Compression
- Analog/Digital Outputs
- Computer-Compatible Interface
- Operator Displays and Entry (Alpha/Graphics and Keyboard)
- Format Storage/Retrieval
- Data Processing

- Peripheral Control (Disk, Tape, etc.)
- Time Translation
- PCM Simulation

All these operations are directed high-speed under operator control using a distributive processing technique. A 16-bit state-of-the-art (8086) microprocessor provides the operator interface and data display. Programmable bit-slice processors perform data compression, data analysis, alarm gathering and data conversion in real-time. The real-time processors can be used to drive analog or digital ports with processed data (i.e., Nonlinear Engineering Units Conversion).

The distributed processing architecture accommodates real-time data processing by apportioning data analysis functions without the loss of data samples and virtually eliminating aliasing discrepancies. Within the bus structured software, real-time processors are readily added dependent upon the individual user's processing density and speed requirements.

MUXbus

All processing of data within the ADS is accomplished on a high-speed (4 Megaword/Sec) Multiplexed Bus (MUXbus) routed throughout the ADS network. The MUXbus supports the distributive processing architecture by establishing an intermedium between multiple data sources and data processing nodes. Decommutation application modules perform bit, frame, subframe, synchronization and distribute data throughout the system on the MUXbus.

System Architecture

The system hardware and software are designed to support any complement of application modules. The basic system package consists of an autonomous system software assembly as defined previously with associated system hardware. The function of the system package is to provide control and facilities for application modules. In addition, the system package includes a generic set of system directives organized within a complete operator and system interface. Typical directives entail commands allowing the operator to load, save, display, create or set system parameters and programming information. Furthermore, the system software package enables the operator to define which data analysis functions, calculations or displays are desired for each data parameter, including provisions for:

- Data Compression
- Data Conversion (linear/non-linear)
- Data Distribution

- Special Algorithms
- Derived Parameters
- Graphic Cross Plots
- Bar Charts
- Data Display
- Histograms

Modular Architecture

The application modules provide the data recovery sources, distributive processing resources, and special or standard equipment controllers based upon the extent of the specific user's preprocessing requirements. The bus structured modular architecture provides the unique capability to solve complex system problems. An extensive number of various configurations and capabilities are available without hardware or software modification. The key element in this structure is the covalent bond of the hardware and software components of each module into an unseparable entity. Software programs, in the form of firmware for each application module, are a physical and integral part of the hardware module (P.C.Board in most cases). Figure 3 is an illustration of a typical user configuration.

Expandability to conform to system requirement is a foremost advantage of the modular architecture. For example, multiple PCM streams can be simultaneously acquired and merged into a composite stream by the simple addition of another set of decommutation modules.

- Bit Sync
- Frame Sync
- Subframe Sync
- Data Distributor

Other similiar system problems can be readily resolved in the same manner requiring neither hardware or software modification to the existing package.

Application modules may also take the form of peripheral controllers, e.g.,

- Time Code Translators/Generators
- Disc Controllers
- Magnetic Tape Controllers
- Plotters

to relieve host processors of data recording and retrieval functions. Figure 4 depicts the inherent ground station preprocessing capabilities afforded by this architecture.

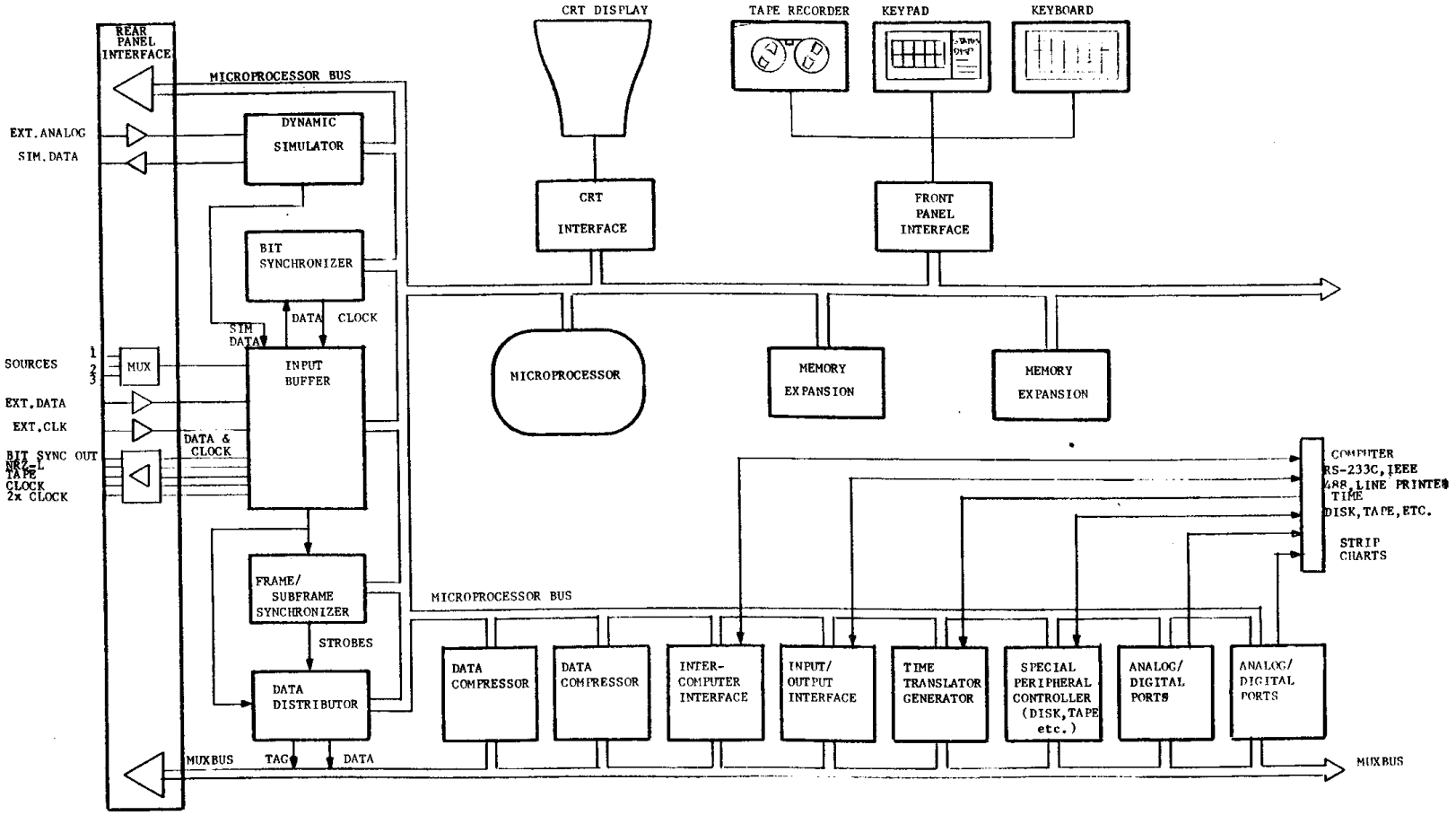

FIGURE 3. ADS FUNCTIONAL BLOCK DIAGRAM

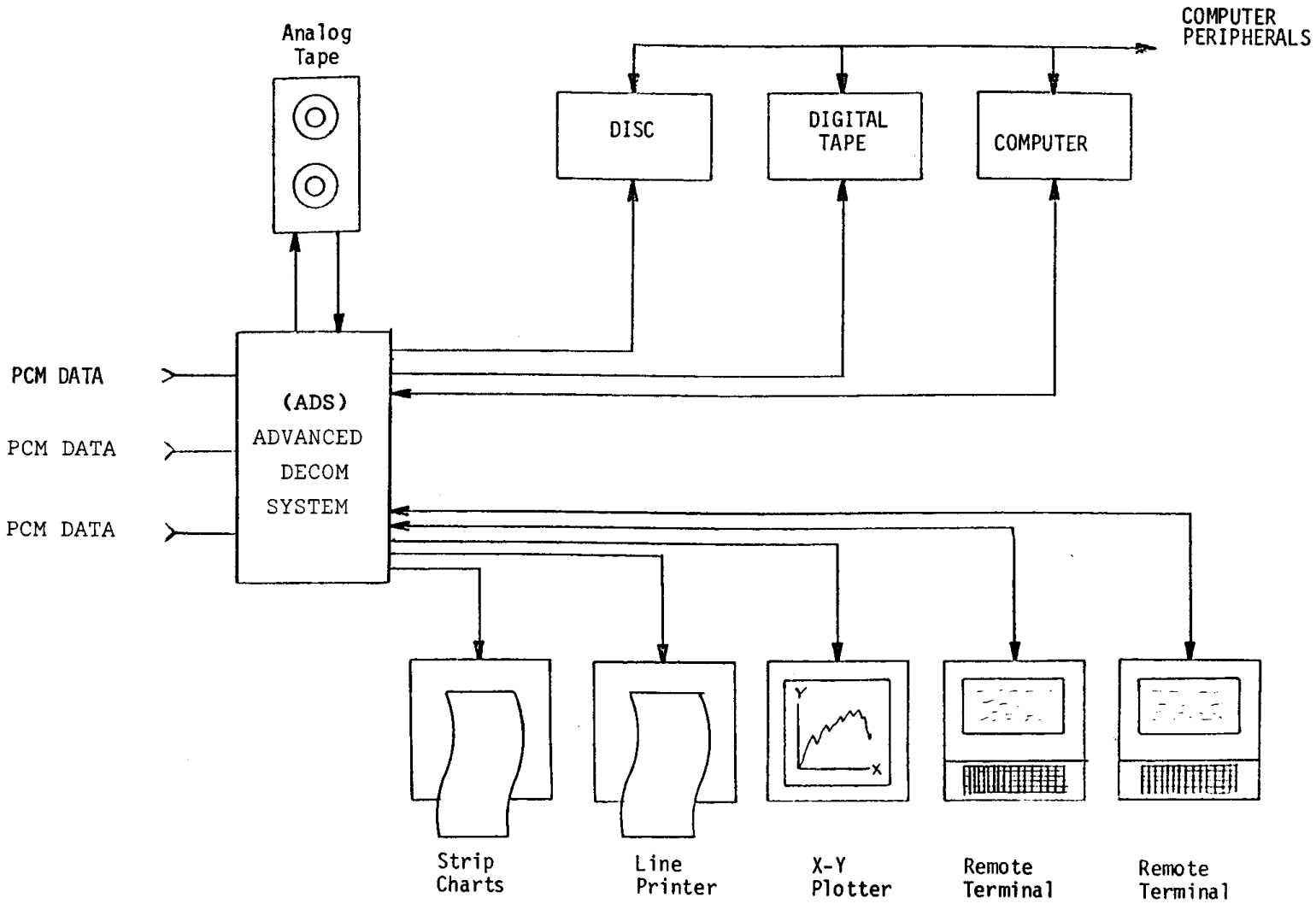

FIGURE 4. ADS GROUND STATION - REAL TIME TELEMETRY PROCESSING

CONCLUSION

Bus Structure Software allows for the standardization of software packages and still supports the extensive variety of user configurations and applications. In the telemetry environment it reduces and/or eliminates development and customizing efforts and costs. It is highly adaptive to specific user requirements without demanding the operator direct the system generation process. Architecturally, bus structured software provides a means for development of a human interface that simplifies system operation and training.

Bus Structured Software, when combined within a system bus architecture, adds dimensions to product developments that are likely to set new standards for telemetry systems. The practicality of this convention is also likely to extend into other system applications involving general purpose data acquisition and analysis.