EVOLUTIONARY FACTORS IN THE DEVELOPMENT OF A REALTIME MULTIPROCESSOR SYSTEM

William F. Trover
Associate Director
Advanced Systems
Teledyne Controls
Los Angeles, California

ABSTRACT

Architectural decisions made three years ago in the design of a high speed preprocessor system for realtime data processing at sustained rates of 200k to 300k parameters per second were driven by the need to provide expansion flexibility and to permit the user to program application algorithms through the use of a high level language. The original design concept was a two bus architecture which would accept and merge data from up to 8 data sources with the required number of parallel computers driven by the realtime processing needs - not the 1.5M wps aggregate throughput capability. Other configuration variables were to enable the use of an optional raw data circular (wrap around) file for intermaneuver or anomaly analysis, the number of analog and discrete outputs for strip chart and visual displays, and the ability to support a wide range of processed data throughputs to one or more host computers.

As a result of future defined requirements, the expansion capability ultimately grew to allow up to 30 data sources, 256 analog outputs, and 196 discrete outputs. A concurrent study of the engine and airborne test community showed that in many applications over 50% of the processing was restricted to repetitive computations such as FFTs and first order EU conversions. Although bit slice processors were much faster than general purpose Application Processors (APs), nobody in the user community said they wanted to write microcode to install their application programs. As the first customer's requirements could be easily handled by adding a few APs, the initial system design concentrated only on general purpose processors with provisions being made for the future addition of special purpose digital signal processors to co-reside with the general purpose APs. At the some time, much of the rotary wing test community's data processing was highly floating point intensive so the AP processor was designed with an independent floating point processor using the fastest possible device technology.

The original two bus architecture using industry standard Versa and VME buses evolved as the design matured to a six bus architecture capable of supporting up to 60 parallel processors. The use of industry standard buses has permitted successful development of configurations using a wide range of third party processors and peripherals from a variety of sources. Larger system configurations are implemented by a multi-chassis structure with functions arranged so that no realtime bus is unterminated or physically longer than 19 inches. The simultaneous software development supporting these changes and encompassing 25 man-years of work is beyond the scope of this paper and will be covered in a separate publication.

INTRODUCTION

In the fall of 1984, Teledyne started development of a Realtime Multi-Processor System (RMPS) to satisfy the need for realtime data merging and preprocessing at sustained data rates of 200k to 300k parameters/second. The launch customer for this program was, Bell Helicopter-Textron Inc. (BHTI) who was developing a new realtime PCM ground station to concurrently support four test aircraft and a ground shake test facility generating an aggregate data rate of about 900k wps. The key design approach which caused BHTI to select RMPS was the use of parallel, general purpose floating point computers instead of the faster bit-slice machines that required writing the processing algorithms in microcode. So as to provide system growth capability, it was determined that the non-processed data throughput needed to be 2M wps so that the upper processing limit that could be accommodated was 1.5M wps.

An open bus architecture was selected for the RMPS because Digital Equipment Corporation had been very successful in permitting third party vendors to have access to their Unibus and Q-Bus as had Motorola with their open Versabus and VME bus. At the same time proprietary buses, such as the Mercury bus (Aydin Monitor), the PCD bus (Fairchild-Weston) and the FLO bus (Loral) were not well accepted because the user was forced to go back to the manufacturer for support and the addition of new functionality. The initial requirements were to merge 3 PCM streams so the original architecture was structured to merge 8 streams or data sources, again to provide system growth. Early in the system development, the need was identified to expand the number of data sources from 8 to 15. By using mechanical expansion, coupled with electrical expansion the number of data sources which could be accommodated concurrently grew to thirty sources.

Box product integration was one of the main features of the RMPS architecture that attracted our customer’s attention. Converting classical box products to card products and integrating them into the computer system produced obvious hardware cost savings. Along with these cost savings came increased speed, lower power consumption and improved reliability along with significantly less cabinet/rack space at the subsystem level. The

initial integration only included classical IRIG frame and subframe synchronizers, the PCM simulator, and the analog and discrete output units. Later, came 1553B bus interfaces, IRIG time code interfaces and, just recently, IRIG bit synchronizers. The analog and discrete outputs required by the original customer were only 32 channels each. Here again, to support future growth, the design was sized for 256 analogs and 196 discretes. This later proved to be a critical decision as subsequent customers have specified the full 256 analog outputs in a single system.

One of the most interesting changes in the evolution of the RMPS design was the number of total buses in the system. The original proposed architecture only had two buses: an input bus where the data was merged and distributed, and the output bus where the processed data was collected and sent to the host. All other preprocessor brands used two buses, but those buses were generally limited to about 4 MHz. Thus, the use of the 10 MHz capability of the Motorola Versa/VME bus would permit two 10 MHz buses to provide all of the margin of safety needed for the 2M wps throughput/1.5M wps processing capabilities established as the RMPS design objectives. As the design progressed, the original two buses grew to seven buses, six parallel buses and one serial bus.

System packaging become one of the critical design problems. It doesn't do any good to develop hardware and software to support 30 data sources, 60 computers, 256 analog and 196 discrete outputs, if you can't find a way of connecting all of this hardware together and not significantly slow it down so you can't provide the throughput which generated the need for large quantities of functional units. Daisy chaining buses to 6 or 7 chassis (as advertised by some suppliers for expansion) means that a 10 MHz bus is reduced to 2 or 3MHz. Thus, the packaging was arranged to place like functional modules in separate chassis with a system star configuration where no single physical bus is over 19 inches in length and it is terminated in its characteristic impedance, with electrically isolated bus repeaters for inter-chassis connections. Even the simplest system uses a Versabus and a VME bus chassis back-to-back in the same mechanical assembly to minimize the physical problems.

From program start to manufacturing assembly and test of production systems took nearly three years. Along the way there were many design improvements and upgrades of the functional modules. As the module performance went up, the costs came down, as the new device technology is denser and uses less power and the chips cost less. With a sound, modular system architecture, module upgrades have been incorporated independent of each other. The result is that the preprocessor system will be able to be significantly improved in the future.

INTEGRATION CONCEPTS

The semiconductor industry for the post decade has proven that greater levels of functional integration have reduced costs and improved reliability. It was believed that if this integration were moved up one level into the telemetry front-end system architecture, similar cost savings and increased system reliability could be realized. PCM ground stations for years had been composed of lots of box products integrated together. Even the "preprocessor" when first introduced by Fairchild-Weston was a box product. The initial integration efforts in this area were pioneered by Decom Systems who integrated frame synchronizers, subframe synchronizers and the PCM simulator into a single assembly. The first level of integration proposed in the RMPS was to combine the integrated PCM decoms and the standalone DAC box, used to drive strip chart recorders and discrete indicators, with the preprocessor computers. A functional block diagram of the original proposed RMPS configuration is depicted in Figure 1.

It was necessary to be able to account for the data coming from different sources so it could be ID tagged and segregated by source as it flowed through the preprocessor. This is one of the most important integration features of the design and it was implemented by some new concepts. By ID tagging the data in the frame and subframe synchronizers instead of in the preprocessor, it is possible to immediately tag the data by both parameter (PCM word) and data source. Also, by going to a 16-bit ID tag instead of the classical 10-bit or 12-bit tag, four bits can identify one of 16 sources, while the LSB 12 bits account for one of 4096 parameters (or PCM words) per data source.

Another advantage of integrating the PCM decoms with the computers is that the frame and subframe synchronizers can be used to provide the first level of data thinning. This is achieved by implementing the decom so that when on ID tag is not assigned to the PCM time slot in the frame/subframe synchronizer, those PCM words are not passed to the preprocessor for further processing.

The next level of integration is to be able to put high speed disk controllers and disk drives on the processor. This capability provides two options for high speed disk recording. One is the raw data disk used as a circular wraparound raw data file of the last n-seconds of data for anomaly analysis (if something in the test breaks), or for intermaneuver analysis where it is desirable to review the last test mode while the test pilot positions the aircraft to either repeat the last test mode or go on to the next test mode. The wrap file size is determined by two variables: the aggregate incoming word rate, and the size of the disks used for wrap file recording. These same disks can also be used for realtime EU data recording so as to reduce the bandwidth of the data sent to the host for graphics/hardcopy utilization. The system was designed to permit the integration of two disk controllers so that one set of controller/disks can be used for the wrap file function (recording raw data),

while the second set of controller/disks can be used for processed (EU converted and/or compressed) data recording.

By integrating the DAC box with the preprocessor, it is not necessary to send the EU converted data to the host computer and then back to the DAC box to generate the EU analog outputs to drive strip charts. This, of course, dramatically reduces the DMA bandwidth between the preprocessor and the host. Finally, by placing the digital to analog and discrete output modules on the preprocessor's processed data bus (the Versa/VME buses in RMPS), data can be concurrently sent to the Global Memory buffers for DMA transfer to the host, while also being output as analog and discrete data for strip chart and event indicator displays.

DATA MERGING AND ID TAGGING

As previously stated, ID tagging of the data in the source modules permits the source module to function as the first level of data compression. The use of a 16-bit ID tag can trade off numbers of parameters per data source for total numbers of parameters. Initially, it was thought that 4096 parameters from 16 different sources would provide a conservative design from the standpoint of system growth. Then one customer said he would have between 32k and 34k measurands on one test aircraft. Because the decoms from DSI were already designed, it was not possible to increase the 16-bit tag in the data source module. Thus, the next level of parameter ID control was the Programmable Data Distributor (PDD), through which all of the data passed for retagging and routing to the appropriate AP for data processing.

It was decided to provide the PDD with two Data Source bus ports with a FIFO in each port. This would permit the connection of two data source module chassis so that the total number of data sources could be expanded from 15 to 30. To overcome the absolute 65k parameter limit of a 16-bit ID tag, the PDD was used to provide a tag transformation function. Because the PDD knew from which of the two 15-channel data source buses each measurand come from, it could expand the source module ID tag word. A 40-bit ID tag was selected for system growth and low cost data routing. Thus, the PDD could route any measurand to any of 8 destinations in parallel (the "broadcast" technique advertised by some competitive preprocessors) and it would expand the ID tag from 16 to 18 bits for internal processing. One additional reason for ID tag transformation in the PDD is to permit the system to process flight critical data continuously during intermaneuver or anomaly analysis periods where the primary data source is the wrap file. In this mode of operation, you can have concurrent processing of the some parameters one from realtime flight critical data plus the same parameter from the raw data wrap file. One bit of the ID tag tells the system whether a particular measurand is fresh data or stale data from the

wrap file. Thus, it is easy to see why the PDD must have a much greater tag/routing capability than the 16- bit tog coming from a source module.

The final consideration in ID tagging is to devise a system concept where it is not necessary to send the ID tag from the preprocessor to the host computer. This use of tags would essentially double the DMA bandwidth which, in turn, would drive up the size (and cost) of the host computer to service the preprocessor in realtime. By segregating the data inside the preprocessor by data source and then building separate synchronous data buffers/data source for transfer to the host computer, it is not necessary to carry the ID tag out of the preprocessor. The DMA buffers for synchronous data only require a header that identifies the data source, the number of frames in that buffer, and the start and stop IRIG times of the data contained in that buffer.

The only buffers sent to the host that require ID tags are the asynchronous buffers that not only require ID an tag, but also a time tag to indicate when the event occurred (e.g., limit exceedance, asynchronous compression algorithm output, Boolean outputs, events). The asynchronous buffers usually represent only about 10% of the total data so the ability to send synchronous buffers per source without tags reduces the DMA bandwidth to an absolute minimum.

DATA DISTRIBUTION

It would have been possible to “broadcast” the data from the Data Source bus (originally a DSI design) directly to the APs. This, however, would have restricted the RMPS architecture to only 16 sources and 4096 measurands/source. Now this may initially sound adequate, but you must consider two important things. First, an ID tag must be available for each derived parameter as well as for a single measurand that is produced by concatenation of two PCM words. This is currently the way all ARINC 429, ARINC 629, STANAG 3910 and MIL-STD-1553B data is processed by PCM data acquisition systems. Next, many people use all 14 or 28 tracks of a longitudinal head instrumentation recorder to acquire data from many different sources on a test aircraft. Thus, 15 sources cannot be considered adequate for all system applications.

All data from a single test aircraft should be merged for concurrent processing. When up to 28 data sources (a 28 track recorder) must be merged and a significant number of tags must be reserved for concatenated and derived measurands (up to 50% of the data on some current tests with multiple data buses), you can easily see that even 65k tags could be totally inadequate for a very large test program.

The solution to the ID tagging limitation problems was to merge the data from all sources and pass it through the PDD. Since all data passes through the PDD, it can be the choke

point in the preprocessor system. By using the data source tags to determine a memory location in the PDD Look-Up Memory (LUM), you can transform the source tag to permit many functions to occur that could not be handled with a simple 16-bit tag. The LUM was configured as a 65k memory, 40 bits wide with 120 nanosecond access time. The PDD has its own 68000 processor (it is essentially 1/2 of a dual AP) used for setup and control, and as the wrap file buffer controller and the Global Memory Management Unit. A functional block diagram of the PDD is presented in Figure 2.

It was initially determined that it would be necessary to “broadcast” some critical data to all parallel computers so that time critical processes could start concurrently. It was felt that the Data Input (DI) bus was not the correct method for achieving this function as some of the FIFOs on the DI bus ports might be nearly full while others may be empty, and therefore the broadcast data would not reach all AP computers at the same time. Thus, the next bus, called the Inter-Processor Bus, was added in parallel with the DI bus between the PDD and all APs. Bits of the tag transformed by the PDD are used to broadcast a PCM word to any or all of the different destinations currently implemented: the wrap file buffer, a specific AP for processing, all APs for correlated processes, and/or the global memory for data that is already in processed form. This left four additional destinations for future system growth.

WRAPAROUND DATA FILE

Many flight test data users want to be able to reprocess data in near real time during the intermaneuver period between the time the test aircraft completes one test mode and the time it starts the next test mode or prepares to rerun the previous mode. This is one use of the raw data wraparound disk file. The other use of the wrap file is for anomaly analysis when something “breaks” on the test aircraft. The wrap file is required because statistically the test engineers are never looking at the parameter(s) that “break” when there is a problem on the test aircraft.

The wrap file was another function which necessitated expanding the number of system buses from the original design of two buses. At very high data ingestion rates it is necessary to “broadcast” data to both the APs for processing and to the wrap file disk controller for recording. This fourth bus become the wrap file bus. It permits data to be recorded or read from the wrap file disk(s) without adding any data bandwidth to the Data Source bus or the Processed Data bus. The size of the wrap file is set by one controllable variable and one non-controllable variable. The controllable variable is the total disk capacity of the wrap file disks. This can be as low as 50 megabytes and as high as 2.4 gigabytes, because the system can have a separate wrap file disk controller and one Interphase V/SMD 4200 controller can support up to four 689MB (unformatted) Fujitsu Eagle disk drives.

The preprocessor host computer, the RMPS Control Processor (CP), does not require disk accesses in realtime. Thus, one disk can be the system disk and the 2nd through 4th disks can be the wrap file disks. The disk controller is dual ported so the system disk is accessed by the CP across the Processed Data (Versabus) bus while the wrap file disk(s) are accessed via the second port on the disk controller which is connected to the Wrap File bus. The wrap file buffer smooths out any discontinuity in data flow caused by bus or disk controller contentions.

APPLICATION PROCESSORS

The Application Processors were configured as two separate autonomous computers: (1) a Motorola 68000 CPU with 0.5 megabytes of dedicated RAM, and (2) an independent Floating Point Processor (FPP). Making the FPP independent was necessary for helicopter and jet engine testing where the processing was skewed very heavily towards floating point algorithms. This is especially important for processes such as min/max value detection, harmonic analyses, FFTs, etc. A functional block diagram of a Dual Application Processor is presented in Figure 3.

The Weitek 1032/1033 APU/ALU devices were selected for the FPP, as they were the fastest devices available. This FPP implementation took more PC landscape, much more power, and produced higher hardware costs than a bit-slice engine, but it generated about a 5 MFLOP FPP which was the fastest floating point processor that could be designed in the ‘84/85 time frame. This proved to be a good design decision, because just today (4 years later) devices are becoming available which will afford significant improvements in FPP performance. By totally decoupled the FPP from the basic 68000 processor, AP throughput could be speeded up, as the 68000 processor could perform fixed point operations while the FPP was performing floating point operations.

The APs were the next place where the number of system buses expanded. The original concept was for the PDD to accept data from the Data Source bus and then, with the use of FIFOs, time share the same bus to send transformed data/tag pairs from the PDD to the APs. In the detailed design it proved impractical and too risky to multiplex the Data Source bus. Thus, a fourth bus, the DI bus, was added to send data plus 18 bit tags (transformed from 16 bits by the PDD LUM) to the APs.

The next problem to solve in timing was how to easily differentiate between a data/tag pair, routed to a specific AP for processing, from a measurand such as time or an event that must go to all APs concurrently to start timed critical processes. Thus was born the IP bus. Data broadcast on the IP bus by the PDD caused an interrupt on all APs to accept this measurand, so time or event critical processes could be started concurrently in separate computers.

The IP bus, however, was never predicted to be heavily loaded, so it become the raw data recirculation bus for derived parameters being computed in different APs that used the same measurand as one of the terms of the algorithm. For example, Impact Pressure is used in IAS, CAS, TAS, Mach, etc., and these derived parameters need to be able to be processed in different APs so that there is no data staleness as would be experienced if they were all computed in the some AP. Data sent to a specific AP on the DI bus could be returned to the PDD for further tag transformation and routing to a different AP. Thus, via recirculation, the same measurand could be processed in two different APs nearly concurrently.

Next there was a need to control each of the APs during a test run and to be able to dynamically change processes or start and stop them from another source. This capability is provided by the Serial Bus S-Bus) which connects one port of a DUART on each 68000 in the system (on each decom, the PDD and each AP) to the RMPS Control Processor. The second DUART on each AP was provided for dynamic testing purposes but has been used by some customers to connect that AP directly to the Host so associated processes in the Host can be monitored and controlled directly without going through the RMPS CP.

The key design consideration in a multiprocessor system is to keep data from getting overwritten or lost due to bus arbitrations. This, of course, is minimized by having four buses on each AP. However, this did not provide enough protection, so all bus interfaces were configured with FIFO buffers to allow for asynchronous operations. Now, the system had grown from the original two bus architecture to a system with five buses. A functional block diagram of the RMPS at this phase of the design is presented in Figure 4.

A SYSTEM HOST

Any time two or more computers are interconnected in a system, one computer must be the Master that controls all the other processors. The RMPS Host had two principal functions: to communicate with the ground station host, and accept programs downloaded to it and store them on the local disk. At system initialization time, it loaded all programmable modules with the realtime programs stored on the system disk. At that time it reverted to realtime operations of system housekeeping and diagnostics. It also was used to control the changes in realtime operations.

The selection of the RMPS CP (Host) presented a real dilemma. This was because off-the-shelf 68000 based monoboard computers that were the best from the hardware standpoint, did not have an acceptable operating system and operating systems are not something you want to design. To permit interactive control from either the host or a terminal connected to the CP, the OS needed to be multitasking. The best solution today would be the MicroVAX board product from DEC. Unfortunately, four years ago that product was not

available. The best board computer Was a Motorola CPU board with local memory. Unfortunately, the Versados operating system at that time was totally unacceptable. A compromise had to be made and Charles River Data System's CP-32 with the UNOS operating system was selected. The CP-32 has two CPUs, a main processor with a cache memory, and a separate processor for the RS-422 I/O ports (up to 8 are supported). The negative feature of the CP-32 was that it had no on-board memory so it had to use a part of the system's global memory for its own programs. However, its low realtime workload, plus its local cache, only added about 5% to the Versabus bandwidth so it appeared to be the best compromise. Selection of the CP-32 turned out to be acceptable, as later CRDS introduced the CP-20 which is a software compatible dual 68020 based version of the CP-32. Thus, the RMPS Host functions can be speeded up in applications where that is necessary by just upgrading (exchanging) the CP board.

GLOBAL MEMORY MANAGEMENT

The key to RMPS being able to generate the minimum DMA bandwidth to the ground station host in real time, is by dropping all ID tags (except for asynchronous outputs) in the APs, and establishing synchronous processed data buffers for each data source in the system's global memory. This, of course, requires some memory management functions which were best performed in real time by the 68000 processor on the PDD that is only used for setup and control.

The key design consideration here is Versabus arbitration. When you have a large number of computers aspiring to be bus masters you need a high speed round robin arbitration scheme. This is the most critical throughput limitation point after the PDD. In order to cut arbitration time essentially in half, a look-ahead arbitration scheme was implemented where the next master is being set up while the current master is transferring processed data from the AP to the global memory buffers.

PREPROCESSOR TO HOST DATA TRANSFERS

As stated earlier, separate synchronous data buffers are established in global memory for each data source. These buffers contain no ID tags, just positional data in the same time sequence that it was received from the source. Since asynchronous data requires not only an ID tag but time tags for each measurand, a single asynchronous buffer is established for all data sources. In addition, there is the Current Value Table (CVT) that is used to drive displays. CVTs are set up for periodic (one to five times/second) transfer to the Host. Asynchronous buffers can be transferred either periodically, when they are full, or upon request from the Host. Synchronous buffers are transferred when they are filled, and ping-pong buffers are used, so one buffer from one source can be DMA'd to the host while the

other is being filled by the APs. To make DMA transfers efficient, large ping-pong buffers (up to 128k bytes) can be set up for each data source.

All data is sent from the APs across the Versabus to the global memory for buffering. From global memory, data is sent to the analog/discrete outputs and to the Host DMA channel. It was decided to use a dual ported memory for the global memory and another bus, the RAM bus, for connection of the DMA channel. The use of dual ported global memory permitted the APs to fill the buffers from the Versabus while the DMA channels emptied the buffers via the RAM bus. Thus, only two moves per measurand are required for processed data on the Versabus. One from the AP to global memory and one from global memory to the analog and discrete output modules on the buffered VME extension of the Versabus. Fully expanded RMPS configuration with two data sources buses and the addition of the RAM bus is presented in Figure 5.

ANALOG/DISCRETE OUTPUTS

Integrating the analog and discrete outputs into the preprocessor brought some problems, but also many more benefits than keeping them separate as other preprocessor system architectures had done. The negative consideration is that analog/discrete outputs did nearly double the Versabus bandwidth when many (256) high frequency analog signals are used to drive high speed strip chart recorders. However, the integration of this function permitted precise output timing to be achieved for shake test applications or other closed loop control functions where precise, even time intervals of the DAC outputs must precisely match the data source input word rates, so as not to generate harmonic distortion in the shaker waveforms.

This capability is achieved by placing FIFOs on the DAC cards feeding each D/A converter. The clock for timing each DAC can thus be extracted from the word clock on the decom that the control channel data output come from. In this manner, the output word rate is crystal controlled and synchronized to the input word rate. Integrating the analog and discrete output functions in the preprocessor also significantly offloads the Host computer I/O requirements so it can be downsized. In fact, with general purpose computers in the preprocessor, the only thing that sizes the ground station computer is the number of concurrent users and the amount of time required for batch processing offline. Thus, many realtime ground stations have been implemented with a MicroVAX class of computer with all realtime processing performed in the preprocessor.

LOCAL EU DATA RECORDING

The ability to locate large, fast disks on the preprocessor permits compressed, EU converted data to be recorded in real time on the preprocessor’s disks. This, in turn,

reduces the Host DMA bandwidth to a very low level in real time for support of graphics terminal and hard copy devices. Here is where the selection of the Interphase V/SMD 3200 dual ported disk controller proved to be a good choice. Initially, this disk controller only supported two concurrent disk drives. However, with large disk drives, such as the Fujitsu Eagle (689MB unformatted capacity), up to 1.2 gigabytes of EU data storage at rates up to about 1.3 MB/second could be obtained. More recently, the V/SMD 4200 has become available which supports up to 4 disk drives, and the throughput has been increased to 2.2 MB/second sustained.

The dual porting of the disk drive permits the disks to be dedicated to the wrap file function in real time and to store programs downloaded from the host in offline operations. In addition, UNOS supports two disk controllers so that total EU disk storage capacity and throughput can be nearly doubled to 4.8 gigabytes from the figures achievable from a single controller.

NETWORK INTERFACES

The selection of the Versa/VME open bus architecture has recently brought new benefits. There are now available Ethernet LAN interfaces which permit processed data to be routed in real time via the LAN to workstations. This further offloads the host computer in real time, and it is a vehicle for workstations to have access to the CVTs on an as-required basis.

SYSTEM PACKAGING

System packaging did not have as many evolutionary changes in the RMPS development cycle as the hardware architecture. The selection of the Versabus chassis for the computers was driven initially by the frame and subframe synchronizers being existing products. The card cage was mounted vertically with a smoked glass door on the front so the status lights on the decoms, the PDD and the APs could be seen at one time through the front door. All I/O cables were to enter through single or double height VME cards with the VME cage mounted, backplane to backplane, with the Versacard cage. An illustration of this dual chassis concept is presented in Figure 6.

The card cages were hinged on one side so that, when the assembly is pulled from the cabinet on slides, the Versacard cage can be swung aside to provide easy access to both the VME and Versabus backplanes. Backplane to backplane cables (on the P2) are used for signal transfer between function modules in the different cages in the rear of the cabinet. The original packaging concept was to have all user interface cables connect to the system via the VME cage. Unfortunately, this was not always possible because the disk controller and DMA channel were only available on VME card configurations which

required Versa to VME adapter cards for use in the Versacard cage. Because these purchased cards were only double height cards (triple height VME was not then available), the disk drive connector and the host CPU DMA cable connector were located by the manufacturer on the front of the cards. Thus, in the end, the clean Versabus architecture with no cables on the front of the cards, was violated because all needed functional modules were not available on the Versacard form factor.

Separate cooling plenums with low noise squirrel cage blowers were added on the top of each card cage so as to blow down against normal convection and optimize the heat transfer from the cards. Card spacing was reduced from the 1.0 inch O.C. standard to 0.9 inches to increase the module capacity per chassis. Two APs, each with 0.5 megabytes of RAM and two floating point processors, were packaged as a motherboard/daughterboard two-card assembly so as to be able to pack the maximum amount of computer power in a single card slot of the chassis.

Analog and discrete outputs were packaged on VME cards and installed in the VME cage. An active Versa-to-VME repeater was required to connect the Versabus to the VME bus so as to have each bus no longer than 19 inches and have it terminated in its characteristic impedance. The bus repeater slowed down the VME bus, but in realtime operation traffic on the VME bus was undirectional. With data going from the global memory in the Versabus chassis to the analog/discrete output cards in the VME chassis, the slowdown of the bus was transparent to system operation.

The PDD was a two card functional unit that provided a movable barrier to divide the Versabus cage backplane into source modules (PCM decoms/1553B bus listeners) and the computer modules. It permitted the same pins on the backplane to be used for the Data Source bus and the DI bus, with the DS bus stopping at the PDD Controller Card, and the DI bus starting at the PDD Distributor Card. Internal buses of the PDD were used for data transfers between the two cards. The relocatability of the PDD assembly permitted a system to be reconfigured in the field to accept more computers or source modules depending on the primary objective of the system expansion. For very large systems requiring over 18 Versabus card slots, two chassis are co-located in adjacent cabinets and all source modules and the PDD are located in one chassis, while all of the AP computers, the CP-32 host, disk controller and DMA channel in the second chassis. With this mechanical configuration, five card slots were used by the host and its peripherals, leaving 13 slots for dual AP assemblies (26 floating point computers). No systems to date have required more dual AP modules than this, so the next level of system expansion, the addition of a second computer chassis (a 3 chassis system), has not been necessary.

SYSTEM UPGRADES

There have been several upgrades of modules and capabilities that are retrofittable back into earlier delivered systems by reconfiguring the backplane (the wirewrap portions of the backplane). The first upgrade was the conversion of the decom channels from a two-board frame/subframe synchronizer operating at 10M bps to a single board frame/subframe synchronizer operating at 20M bps. This upgrade was accomplished over a year ago. The next upgrade was a speed increase in the PDD (the system's potential choke point) to a stable 4M wps throughput (i.e., 2 million data words plus 2 million ID tags per second accepted from the Source Bus(es) with the PDD performing tag transformation and routing to the dual APs). This was accomplished concurrently with the decom upgrade to permit higher aggregate input rates.

Several dual AP upgrades have been implemented. The first was just an increase of the clock speed from 12.5 MHz to 16 MHz. The next was an increase in memory speed followed by the changeout of the I6MHz 68000 for a 20 MHz 68020. This last change, completed this year, doubled the AP computer's speed. Another system upgrade was to increase the Versabus arbitration speed so that more dual APs can be accommodated without slowing the system down.

The most recent upgrade has been the development of a 20M bps tunecible bit synchronizer on two triple-height, double-depth VME cards. The use of triple-height, double-depth VME cards for the bit synchronizers means the entire system can now be converted to VME as this size card is within a few millimeters of the size of the Versabus card. Thus, all Versabus cards can be converted to VME by just making a new PC layout to match the VME pinouts. This eliminates the need for box product bit synchronizers and reduces the cabinet count from three to two cabinets for large preprocessor systems. It also simplifies cabling and improves reliability. At the some time that the bit synchronizer integration was implemented, the speed of the global memory was more than doubled to permit elimination of the need for the RAM bus. This, of course, reduces system costs, as the RAM bus adapter is deleted and the backplane wiring costs are reduced.

These upgrades hove permitted sustained computational throughputs to over 750k wps with a very broad mix of processing algorithms. This has been achieved without using more than 10 Dual APs (20 computers). The next upgrade of the APs will be the use of 30 MHz 68030s which should quadruple the processing speed per AP. Thus, it appears the 1.5M wps fully expanded sustained computational capability of the system will never need more than the two chassis of a present large system and less than the 26 computers on 13 assemblies which can be installed in a single computer chassis.

CONCLUSIONS

There have been many architectural changes between the initial two-bus parallel processor design proposed and the large systems currently being delivered with two chassis and six buses. The data flow through the system has been improved by the addition of more parallel buses, which was made physically possible by the initial selection of the Versabus chassis with its much larger P1 /P2 pin capacity. The selection of the general purpose floating point computer over the faster microcoded bit-slice machines has proven to be the best choice in terms of customer acceptance. Many users have been able to easily add their own algorithms (Aerospatiale added over 100 algorithms themselves) without the need for microcoding. The upwardly compatible Motorola 68000 CPU line has permitted several improvements in computer power throughput without any obsolescence in costly software. There is still more hardware throughput capability coming from Motorola, so it appears we may never have to add an unfriendly bit-slice polynomial conversion engine. However, should the market (or the competition) make that necessary, the architecture in hardware and software already exists for the coexistence of bit-slice EU engines and general purpose floating point computers. The ability to incorporate third party board level array processors means that significant expansion in FFT and similar types of calculations can be accommodated as the need arises. Thus, the RMPS becomes a living product that is continually upgraded to produce greater performance and/or lower cost per MIP. Its open system architecture makes it easy to expand and more friendly to the customer.

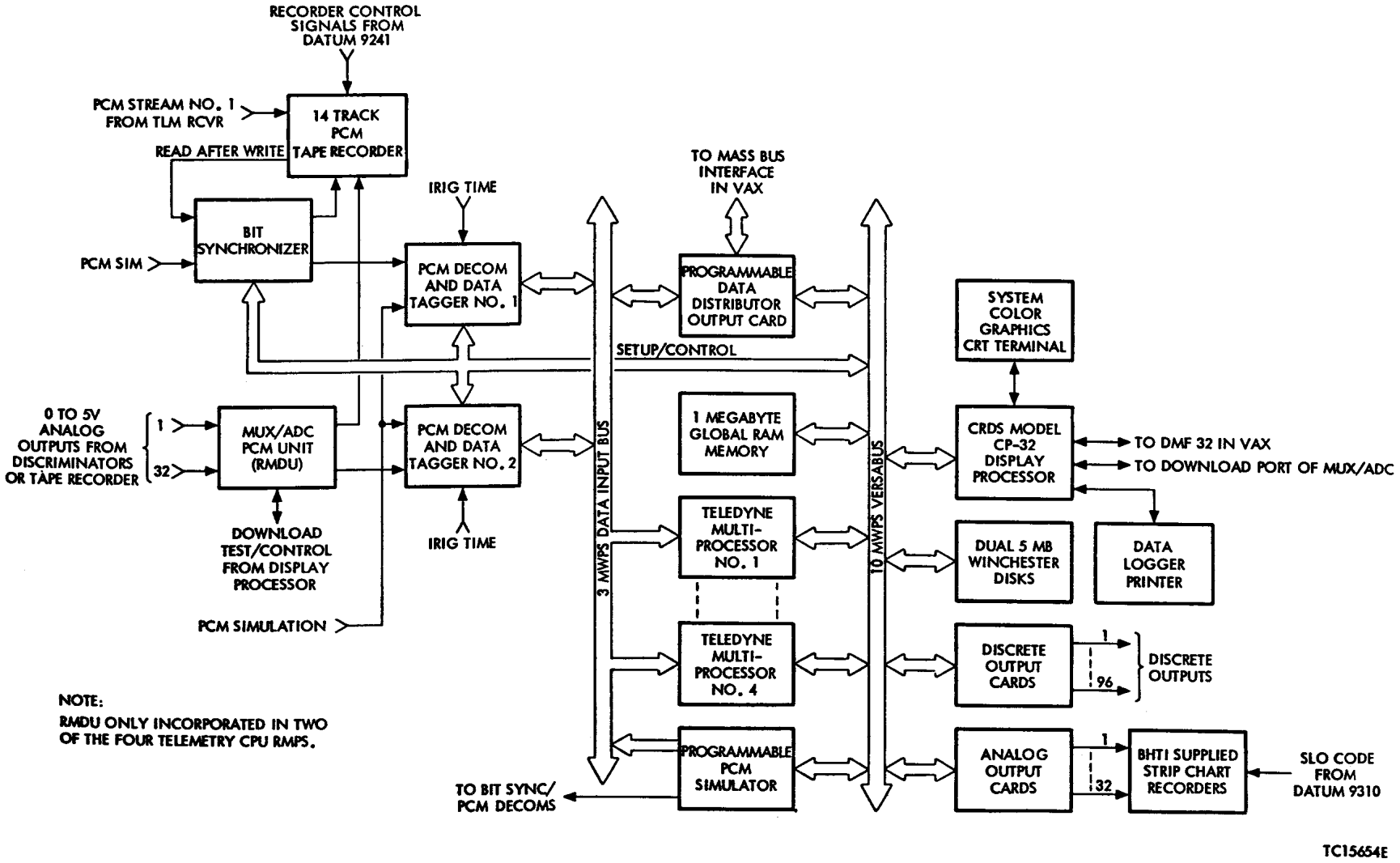

FIGURE 1. ORIGINAL PROPOSED TWO BUS RMPS CONFIGURATION

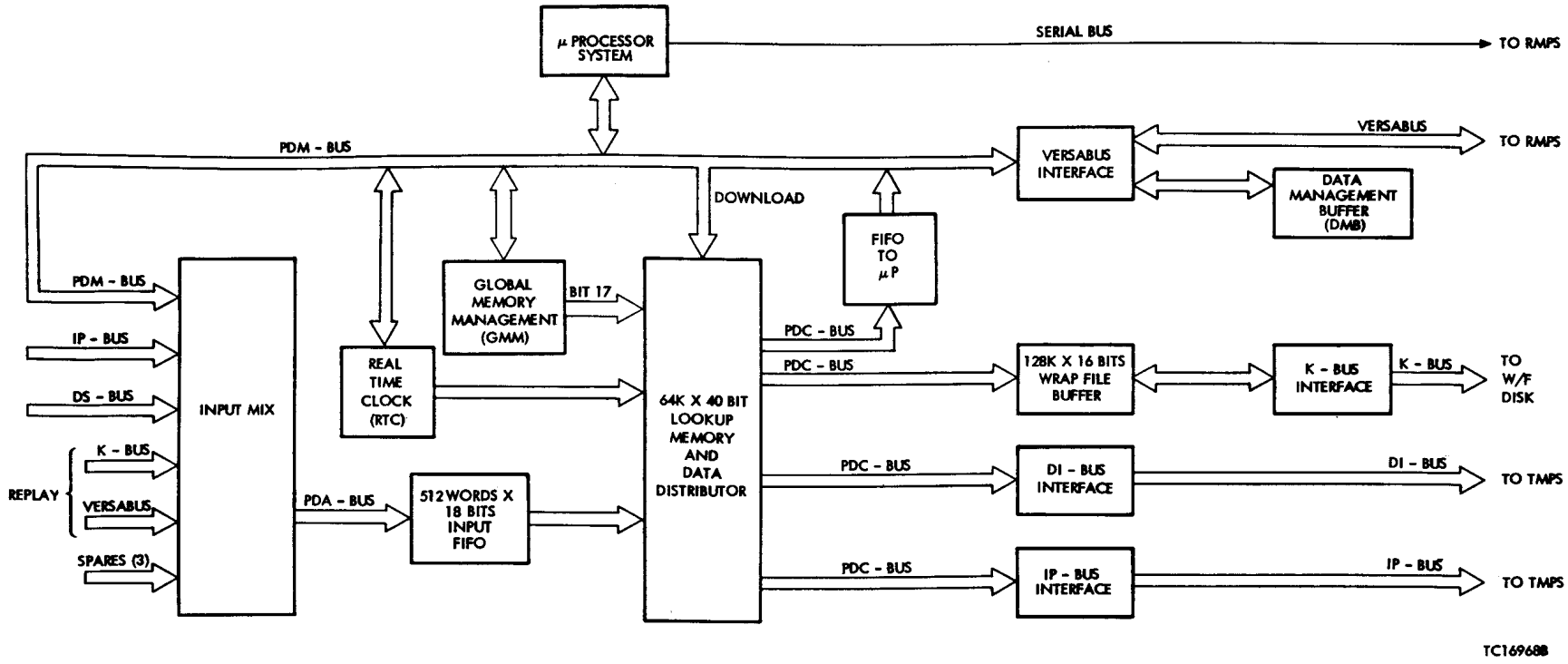

FIGURE 2. PROGRAMMABLE DATA DISTRIBUTOR POD

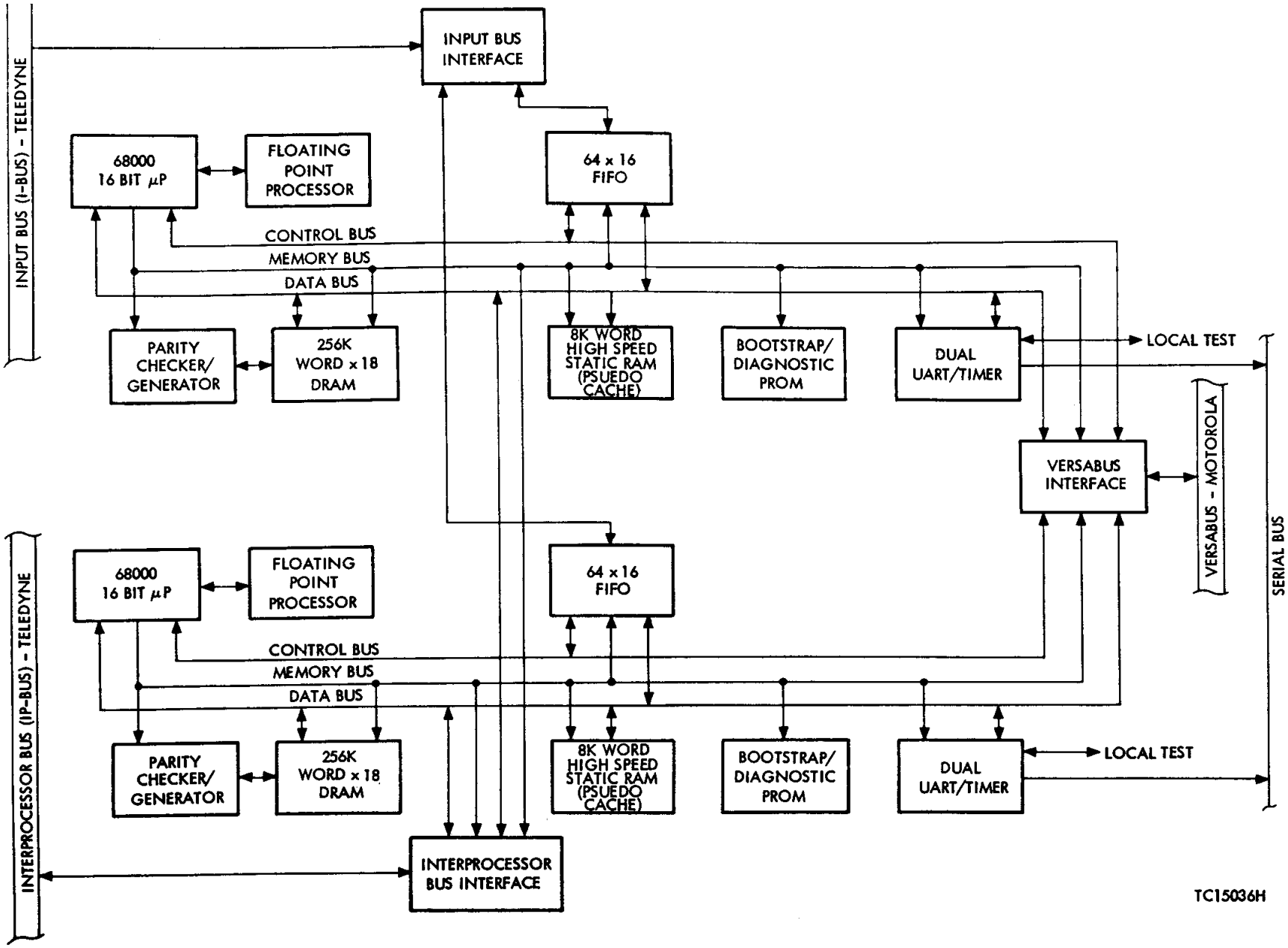

FIGURE 3. DUAL APPLICATION PROCESSOR

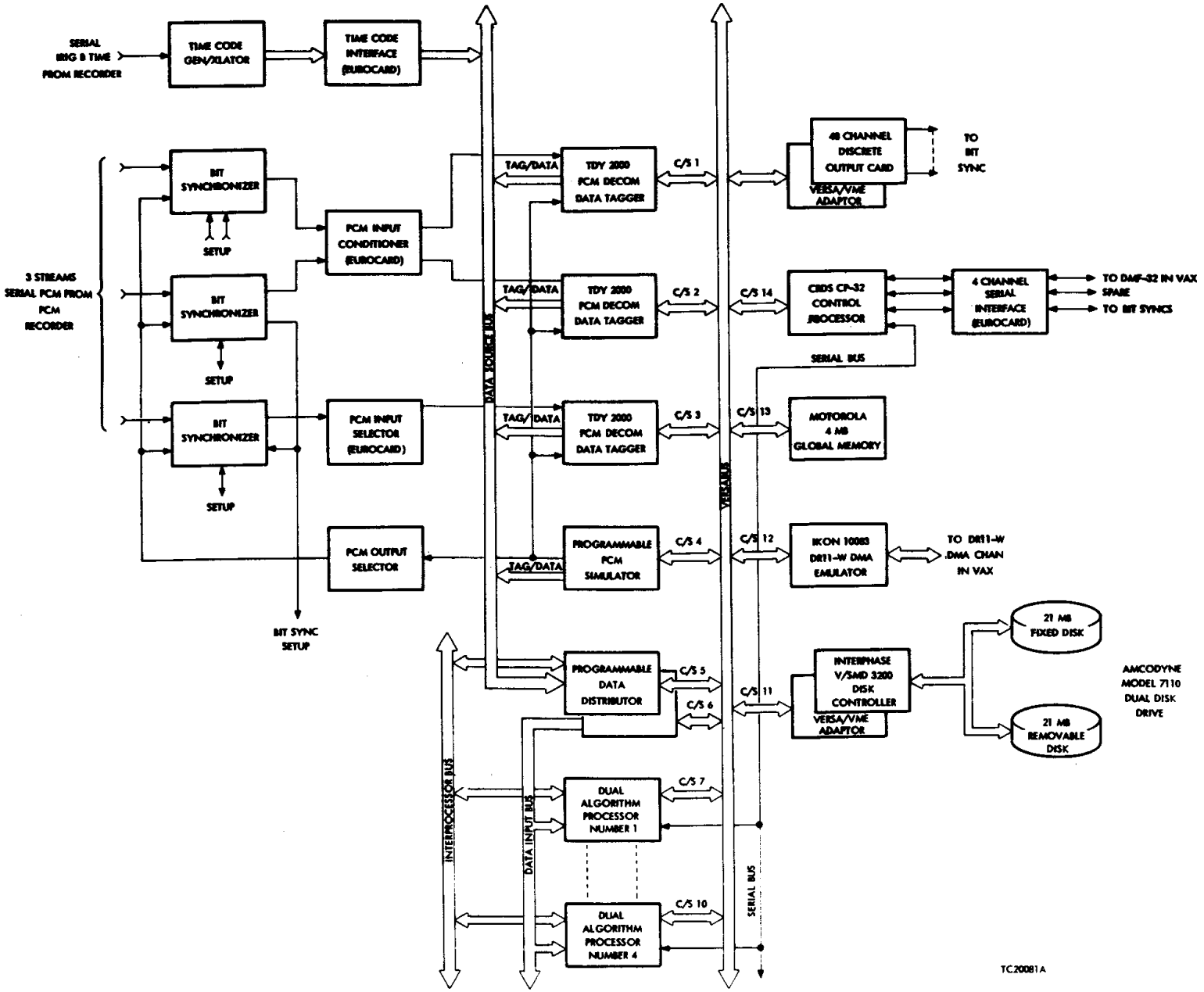

FIGURE 4. RMPS ARCHITECTURE EXPANDED TO FIVE PARALLEL BUSES

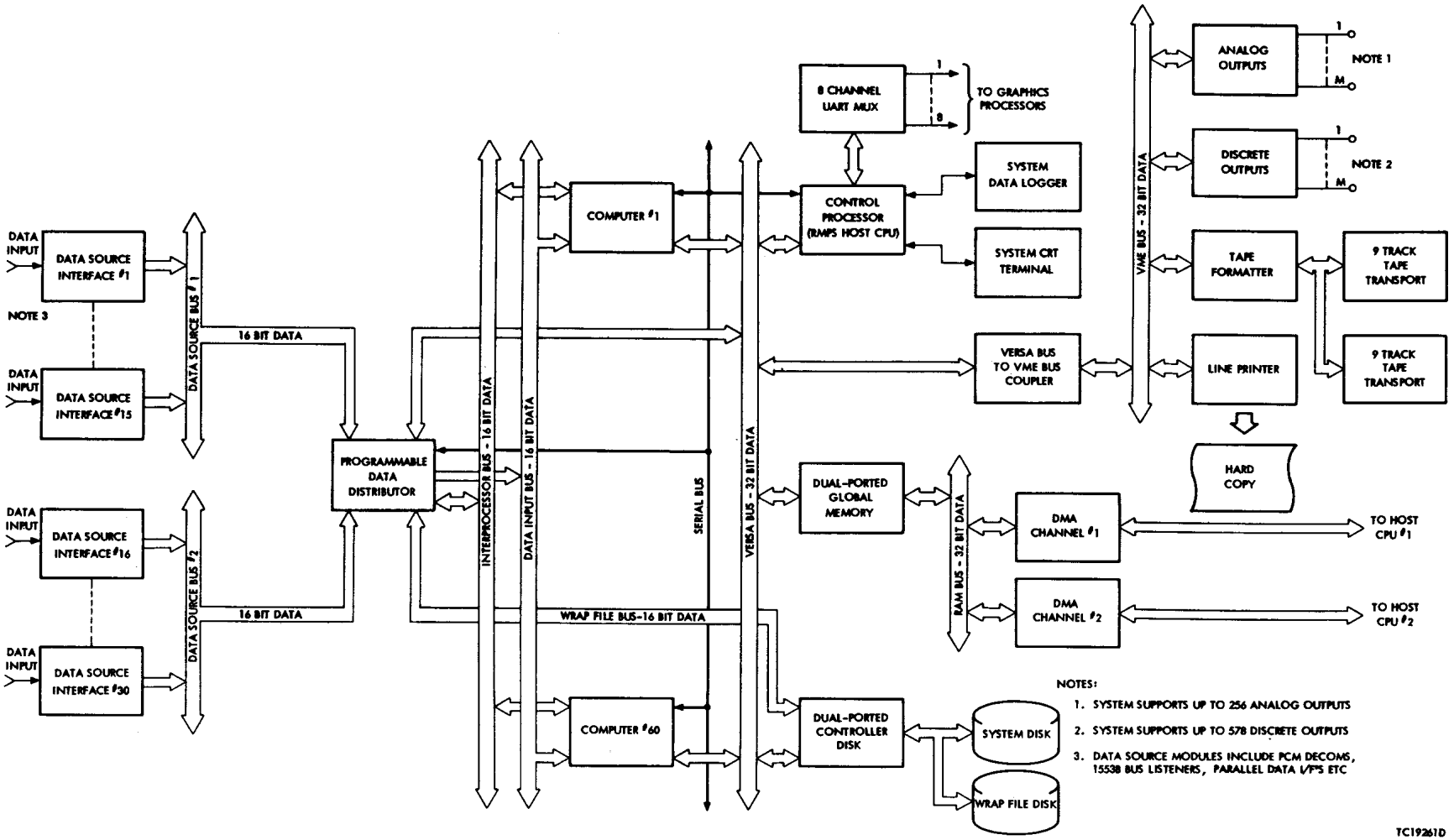

FIGURE 5. FULLY EXPANDED RMPS CONFIGURATION WITH SEVEN PARALLEL BUSES

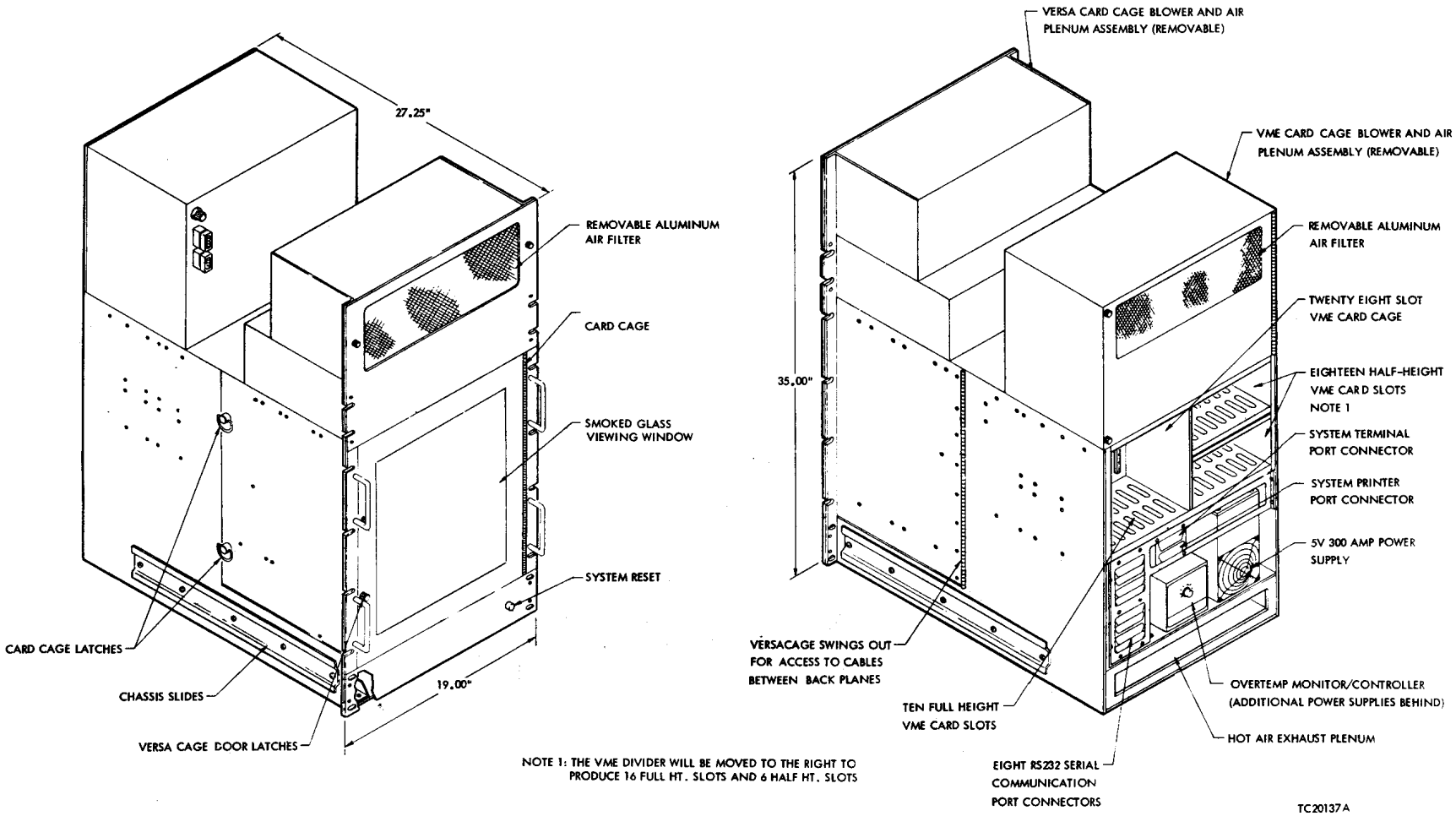

FIGURE 6. DUAL CHASSIS PACKAGING CONFIGURATION