NASA Technical Memorandum 107104
AIAA–96–1127

Description and Simulation of a Fast Packet Switch Architecture for Communication Satellites

Jorge A. Quintana
Lewis Research Center
Cleveland, Ohio

and

Paul J. Lizanich
Analex Corporation
Brook Park, Ohio

Prepared for the
16th International Communications Satellite Systems Conference
cosponsored by AIAA, CASI, AAAF, DGLR, and IEICE
Washington, DC, February 25–29, 1996

National Aeronautics and
Space Administration

DESCRIPTION AND SIMULATION OF A FAST PACKET SWITCH ARCHITECTURE FOR COMMUNICATION SATELLITES

Jorge A. Quintana
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135

and

Paul J. Lizanich
Analex Corporation
Brook Park, Ohio 44142

Abstract

The NASA Lewis Research Center has been developing the architecture for a multichannel communications signal processing satellite (MCSPS) as part of a flexible, low-cost meshed-VSAT (very small aperture terminal) network. The MCSPS architecture is based on a multifrequency, time-division-multiple-access (MF-TDMA) uplink and a time-division multiplex (TDM) downlink. There are eight uplink MF-TDMA beams, and eight downlink TDM beams, with eight downlink dwells per beam. The information-switching processor, which decodes, stores, and transmits each packet of user data to the appropriate downlink dwell onboard the satellite, has been fully described by using VHSIC (Very High Speed Integrated-Circuit) Hardware Description Language (VHDL). This VHDL code, which was developed in-house to simulate the information switching processor, showed that the architecture is both feasible and viable. This paper describes a shared-memory-per-beam architecture, its VHDL implementation, and the simulation efforts.

Introduction

Over the past several years, new satellite communications systems that use a meshed-VSAT (very small aperture terminal) architecture have been developed. These systems are geared toward applications requiring data rates of 1.544/2.048 Mb/s.[1] NASA envisioned a need for a meshed-VSAT system that would provide low data rate capability for interactive data, voice, facsimile and video conferencing applications. Such a network would have to be capable of single-point and multicast (multiple-point) transmission, while still being competitive with existing terrestrial communication systems.[1] The multichannel communications signal processing satellite (MCSPS) architecture, shown in Fig. 1, is based on a multifrequency, time-division-multiple-access (MF-TDMA) uplink and a time-division multiplex (TDM) downlink. There are eight uplink MF-TDMA beams, and eight downlink TDM beams, with eight downlink dwells per beam. The advantages of this architecture are discussed in a previous paper.[1]

As Fig.1 shows, the information-switching processor (ISP) is the most important subsystem in the architecture. This portion of the MCSPS was developed and simulated in-house by using VHSIC Hardware Description Language (VHDL). NASA Lewis Research Center's computer-aided-design capabilities combined VHDL code with schematics to simulate this architecture fully. Thus, VHDL describes in software the hardware characteristics of digital components. The code's flexibility allows it to be compiled and synthesized to create application-specific integrated circuits (ASIC's). These devices may be easily reprogrammed when necessary; thus, both debugging time and hardware complexity are reduced.

Background

The MCSPS architecture supports both the more conventional circuit-switched user data as well as packet-switched user data. By allocating bandwidth only on demand, it allows destination-directed packet switching of the user data, a more efficient use of the spectrum than the more traditional circuit switching. However, allocating bandwidth on demand has the potential to cause contention and congestion problems.

Contention occurs when multiple users try to transmit data to the same place simultaneously. We can eliminate this problem easily by using a TDM bus speed that is higher than the combined rates of all the transmitting users. For this architecture, restricting the total bandwidth of user data through the satellite to a throughput of less than 1 Gb/s (the current level of technology) eliminates contention.

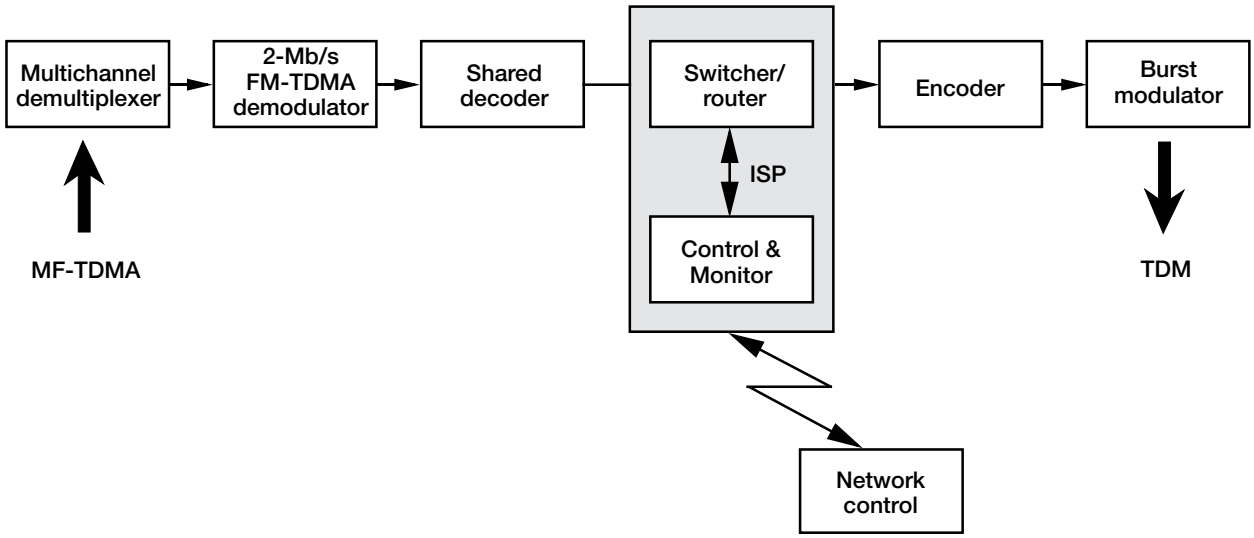

Fig. 1 Meshed VSAT Processing Satellite.

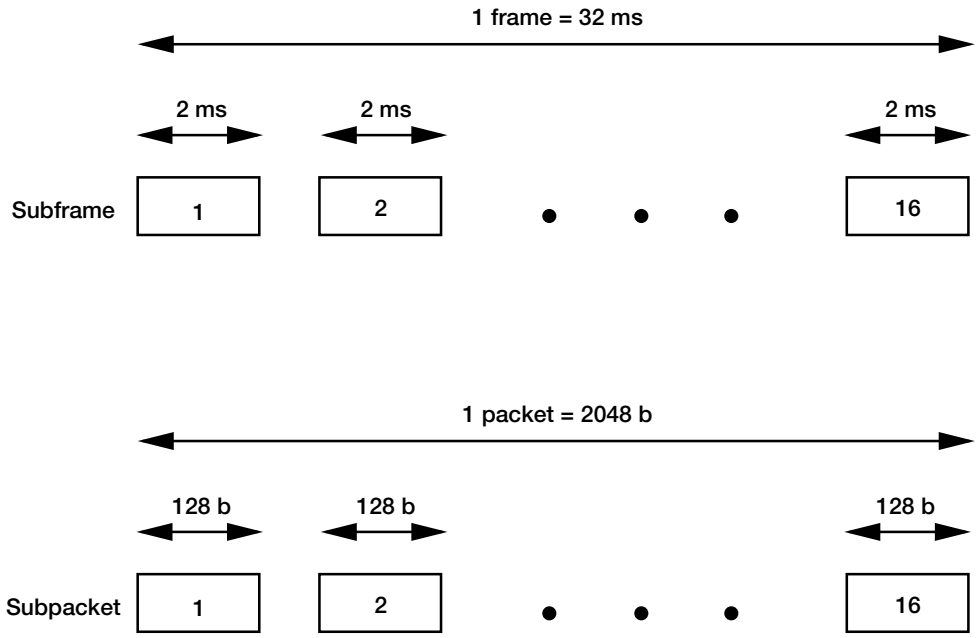

Fig. 2 Frame and packet structure.

Congestion occurs because of the unpredictable traffic patterns in a real system. By monitoring traffic patterns, we can develop network control algorithms to allocate system resources dynamically. Furthermore, the statistical data on traffic characteristics could be used to establish a potential threshold for a network controller to prevent congestion. Although the shared-memory-per-beam architecture is contention-free, possible congestion control algorithms have to be investigated further.

To greatly reduce the amount of onboard memory storage required, the system is designed to use a subpacket scheme,[1] as shown in Fig. 2. Each 2048-b-long packet of user information is broken down into sixteen 128-b subpackets. The first subpacket is a header subpacket containing all the necessary destination information; it is followed by 15 subpackets containing user data. The ISP uses the header information to properly route each of the 15 data subpackets to its correct beam and destination dwell.

Uplink data are transmitted at 2.048 Mb/s by using 1 of 32 frequencies (per uplink beam) in an MF-TDMA scheme.[1] The uplink frame is 32 ms long to allow

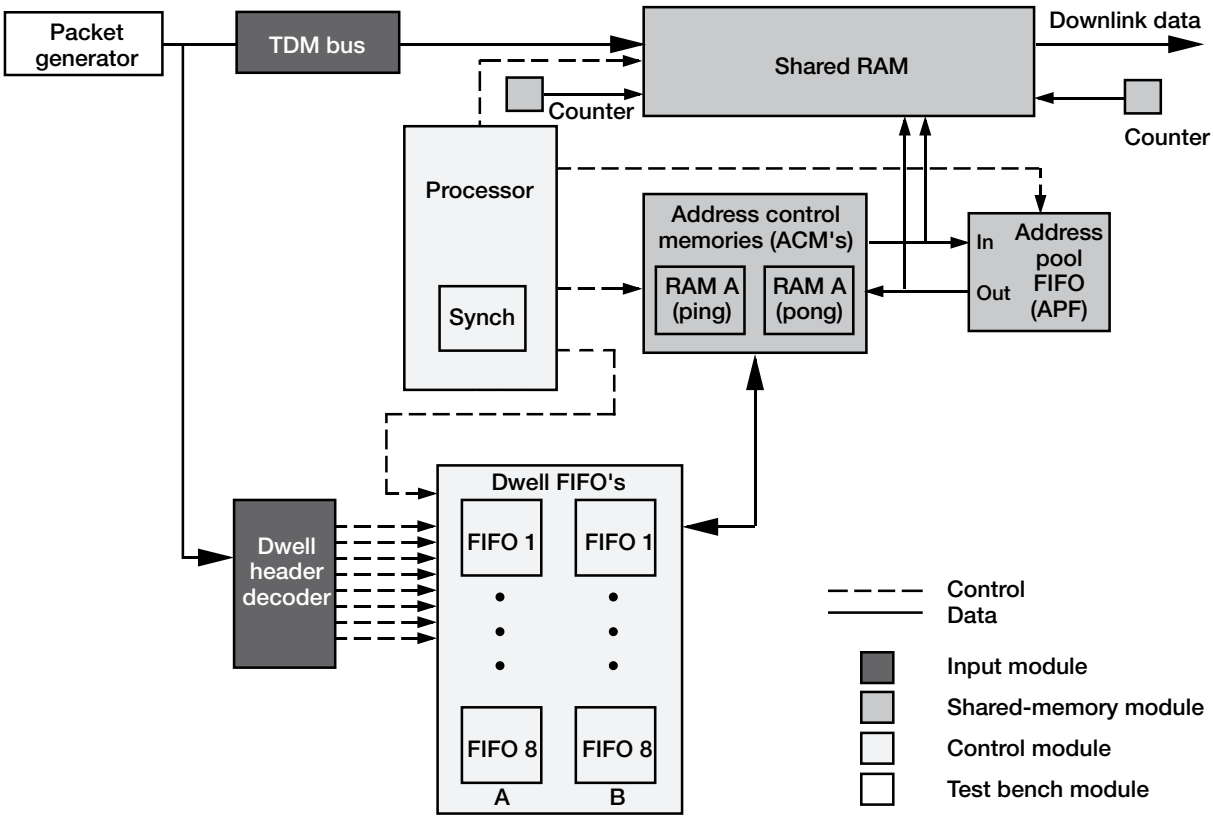

Fig. 3 Shared-memory-per-beam architecture.

transmission of one complete packet/frame. To take full advantage of the subpacket scheme, the uplink frame is further subdivided into 16 subframes, the first of which is dedicated to header subpackets and the remaining 15 to data subpackets (Fig. 2).

The ISP reads the incoming headers and decodes each packet's destination. To further save both onboard memory storage space and uplink bandwidth, the packets can be transmitted either to one downlink dwell (point-to-point connection) or to multiple downlink dwells (multicast).

The satellite downlink frame lags the uplink frame by approximately one subframe. This lag allows the entire header subframe to be processed before the data are transmitted down in a TDM manner at 160 Mb/s.

Architecture

The ISP design is based on a shared-memory-per-beam architecture that is fully discussed in "Fault Tolerant Onboard Packet Switch Architecture for Communication Satellites: Shared Memory Per Beam Approach."[2] The architecture consists of three major subsystems: the input module, the control module, and the shared-memory module. A test bench was also added to simplify packet generation and to test the system's functionality. A detailed block diagram of the architecture is shown in Fig. 3.

Input Module

The input module consists of two different VHDL codes; these are the TDM Bus and the Dwell Header Decoder. The TDM Bus synchronizes the control and data buses while decoding the packet's destination. Since this information is contained only in the header subframe, the VHDL block stores the destination for the next 15 subframes to route the subpackets to the appropriate downlink beam.

Once the proper destination beam has been enabled, the Dwell Header Decoder creates the *write* signal for the destination Dwell FIFO's (first-in, first-out) memories, depending on point-to-point or multicast connection. If the multicast bit is set, then all eight Dwell FIFO's will be enabled for writing; if it is not set, then only the appropriate FIFO is enabled. Although true multicast is allowed at a beam level (subpackets can be transmitted to multiple beams), the switch performs broadcast (transmission to **all** dwells) only at a dwell level.[1]

Control Module

The control module contains three VHDL codes: the Synch, the Processor, and the Dwell FIFO's. The Synch block provides basic system timing for all blocks in the

architecture, and thus makes the system fully synchronous. The Processor monitors and controls all the blocks in the architecture to ensure proper operation of the switch. Since the Address Control Memories (ACM's) and the Dwell FIFO's are used in a "ping-pong" configuration, the Processor has to monitor the frame and subframe counts to properly create *write* and *read* signals for these blocks. Because of the complexity of and the strict timing inherent to the Processor, it is by far the most important piece of VHDL code in the architecture.

The Dwell FIFO's store addresses that point to the Shared RAM during the header subframe. In the subsequent 15 subframes (data), these FIFO's are read sequentially with the help of the Processor. Although the dwell times initially were assigned to be 12.5 percent (1/8) of the subframe time (2 ms), they are fully programmable on a frame-by-frame basis. The dwell times manage the number of subpackets to be read in each dwell by controlling the duration of the FIFO's *read* signal.

The Dwell FIFO's, along with the almost-full flags, can also be used as a monitor for congestion control. Although such a situation was not included in the simulation, the codes can detect a possible overflow condition and signal the transmitting terminal to slow the transmission rate. This capability allows various congestion control solutions to be implemented.

Shared -Memory Module

The shared-memory module has four different blocks: the Shared RAM, two Counters, the ACM, and the Address Pool FIFO (APF). The Counters provide the two least significant address bits (four 32-b words/subpacket) to write and read the data from the Shared RAM. Since all the modules in the architecture work at a subpacket rate (128 b/subpacket), and the Shared RAM works at a word rate (32 b/word), these Counters are needed to provide *read* and *write* addresses to the RAM.

The Shared RAM is a dual port memory (10k x 32 b) shared by all eight destination dwells. This memory reads the next available address from the APF to write the data, and uses the ACM address to transmit it back to the user.[2]

The ACM contains a pair of "ping-pong" RAM memories. That is, data are written into one memory (ping) while the second memory is being read out (pong). These memories are ping-ponged on a subframe basis and their *read*/*write* lines are controlled by the Processor. Both memories are written sequentially, but read randomly (depending on the Dwell FIFO being read); thus temporal switching occurs.[2] The ACM also creates a multicast signal to prevent addresses of multicasted subpackets from being written prematurely into the APF.

The APF provides available addresses to the Shared RAM to write the incoming data. At startup, this FIFO is full (i.e., all addresses are available), and it is read by the Shared RAM on a subpacket basis. Since the Shared RAM always reads the next available address from the APF, it is extremely important to keep the addresses of multicast subpackets from being written back prematurely into the APF.[2] Such premature writing may allow the Shared RAM to read the address (since it is available) before it has been used by all eight destination Dwell FIFO's. This would result in erroneous data being sent down to the user.

Test Bench Module

To simulate this architecture fully, an ISP test bench consisting of a subpacket generator module was created. The generator created 6 multicast and 21 single-destination subpackets/subframe. Each subpacket was created so that it was easily traced by filling the fields with numbers representing the frame, subframe, source (user), beam, and destination dwell, as shown in Fig. 4. In total, 69 downlink subpackets of the 80 available are used in this example, with dwell usage ranging from 60 to 100 percent.

During the simulation process, several modifications were made to the original concept. The original concept called for a 20k x 16-b RAM. To comply with realistic bus speed and memory access times, the TDM Bus was reconfigured to be 32 b wide to increase the cycle time to a more reasonable 61 ns. This change in the bus width subsequently affected the TDM Bus word rate and packet generation, which were simulated at 60 ns. The uplink-to-downlink data-rate ratios (525.288-to-160 Mb/s) were to be kept approximately the same, so the downlink word cycle time of 192 ns was chosen.

For simulation purposes only, the uplink frame was reduced to 256 subpackets of data per subframe, instead of the original 8192, because it was impractical to simulate and verify so many subpackets. This resulted in the

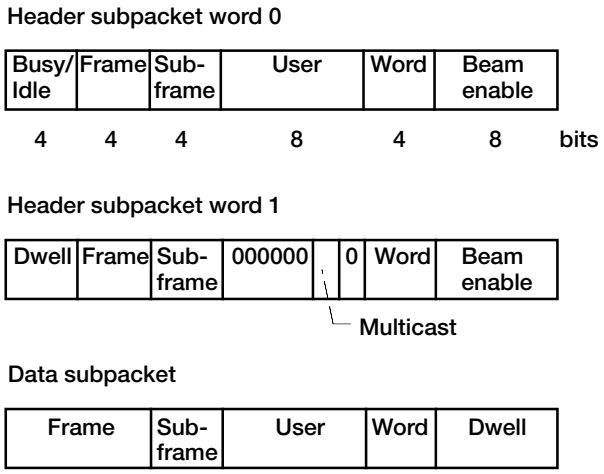

Fig. 4 Packet generator format.

proportional scaling back of several other parameters. The downlink frame length was scaled back to 80 subpackets/subframe and the sizes of the Shared RAM, APF, ACM, and Dwell FIFO's were also appropriately reduced. This system configuration and the use of "generic" statements that allowed users to change system parameters with ease increased the system flexibility and significantly reduced the amount of CPU time needed to simulate and debug the architecture.

Data Flow

The flow of the data can be divided into two phases: the uplink phase (data going from the ground to the satellite) and the downlink phase (data from the satellite back to the ground).

Incoming packets are processed by the TDM Bus and the Dwell Header Decoder to determine their beam and dwell destination. Once the proper beam and dwell(s) within the beam have been enabled, the subpacket is written into the Shared RAM, as illustrated in Fig. 5. The RAM always fetches the next available address from the APF to write the uplink data.[2]

At startup, the APF is full because all the possible *write* addresses are available for the Shared RAM. Once an address is read from the APF, it is stored in ACM A or B, depending on whether the subframe is even (A) or odd (B). Then, the ACM address is written in the appropriate Dwell FIFO, becoming a double pointer to the RAM's *write* address.[2] This process is then repeated for all the subpackets in the first subframe.

In the next subframe, incoming subpackets continue to be read while the previous subframe is being read. The data flow for the downlink is illustrated in Fig. 5. First, the Dwell FIFO's are read sequentially, one FIFO at a time. These FIFO's contain the ACM address that points to the Shared RAM address to be read. Once the ACM address from the FIFO is read, then the ACM is read. The ACM contains the most significant bits of the Shared RAM's *read* address. The least significant bits are provided by a Counter, which reads out the four words contained in a subpacket. Once the Shared RAM is read, its reading address is written back into the APF to show that a subpacket has been transmitted from that location and the location is now available for writing again.

However, this sequence presents a problem for multicast subpackets. In case of a multicast, all eight Dwell FIFO's contain the same ACM address that points to a common RAM location. Therefore, a distinction between multicast and nonmulticast packets must be made

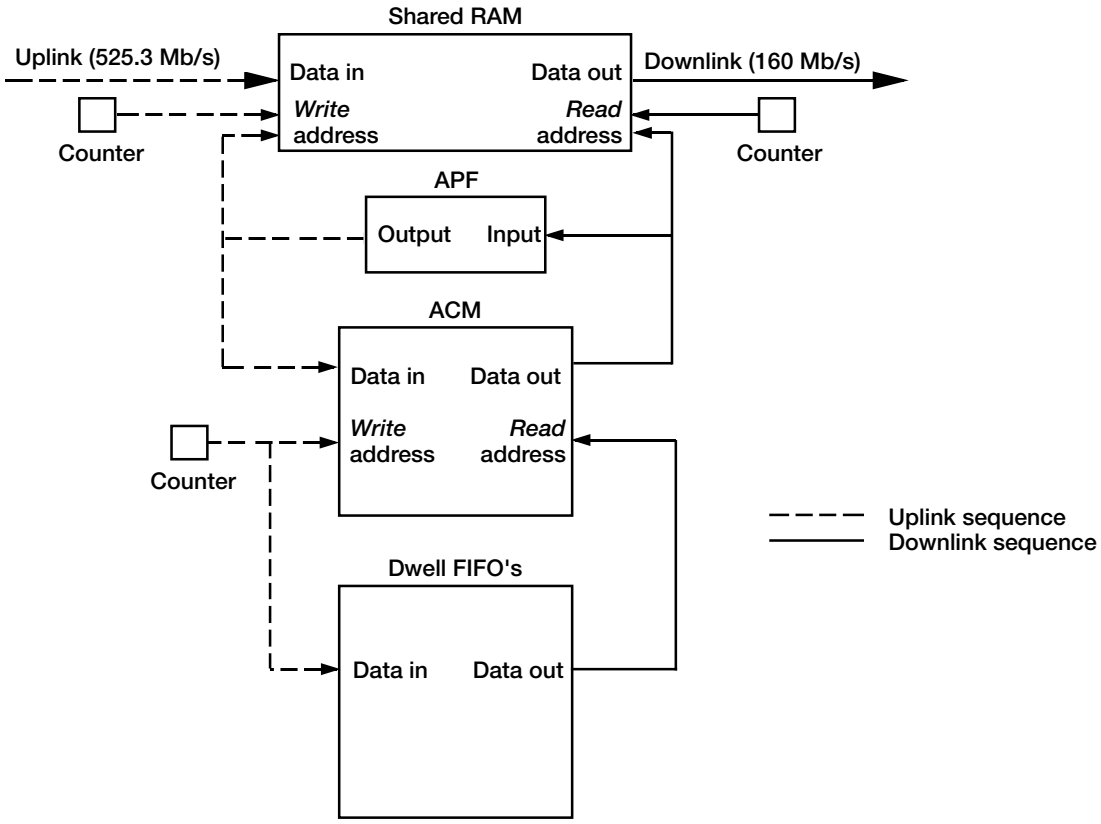

Fig. 5 Data flow for architecture.

to avoid writing the address back prematurely into the APF. If this distinction is not made, the RAM might read the address from the APF (since it is available) and then write an incoming packet in that location before **all** the dwells have read the multicasted subpacket. Such an overwrite will result in an unacceptable transmission error.

To preclude this problem, a multicast signal that distinguished between multicast and nonmulticast packets was created. In the nonmulticast case, the Shared RAM's address of the transmitted subpacket is written **immediately** into the APF. During multicasting, the addresses are **not** written back into the APF until the **last** dwell time (when the eighth Dwell FIFO is read), to avoid an overwrite situation.

A useful example of the architecture's functionality is provided in Fig. 6. As shown in this figure, the subpackets are written into the Shared RAM. The "writing" address is then stored in the ACM. Next, the ACM address is written in the appropriate Dwell FIFO(s). Subpacket 1, for instance,

Data In

Subpacket	Destination
1	Dwell 0
2	Dwell 7
3	Multicast
4	Dwell 4
5	Multicast
6	Multicast
7	Dwell 2
8	Dwell 1
9	Dwell 1
10	Dwell 6

Shared RAM

Subpacket	Address
1	1A
2	1B
3	1C
4	1D
5	1E
6	1F
7	20
8	21
9	22
10	23

Address Control Memory

Address	
ACM	Shared RAM
00	1A
01	1B
02	1C
03	1D
04	1E
05	1F
06	20
07	21
08	22
09	23

Dwell FIFO's

ACM address					
0	00	02	04	05	--
1	02	04	05	07	08
2	02	04	05	06	--
3	02	04	05	--	--
4	02	03	04	05	--
5	02	04	05	--	--
6	02	04	05	09	--
7	01	02	04	05	--

Data Out

Subpackets	Destination
1, 3, 5, 6	FIFO 0
3, 5, 6, 8, 9	FIFO 1
3, 5, 6, 7	FIFO 2
3, 5, 6	FIFO 3
3, 4, 5, 6	FIFO 4
3, 5, 6	FIFO 5
3, 5, 6, 10	FIFO 6
2, 3, 5, 6	FIFO 7

Fig. 6 Switching example.

is going to dwell 0. Once the subpacket is written into the Shared RAM, the RAM address (1A) is stored in the ACM (location 00). Then, this location (00) is written in Dwell FIFO 0, since this is the destination dwell for this subpacket. For a multicast subpacket, the process is the same, except that the ACM address is written in **all** Dwell FIFO's (see subpacket 3).

On the downlink, each Dwell FIFO is read sequentially. These FIFO's provide the *read* address to the ACM, which contains the Shared RAM's *read* address of the subpacket to be transmitted down.

Summary of Results

A shared-memory-per-beam approach for an onboard destination-directed packet switch (DDPS) architecture has been fully simulated as part of a multichannel signal processing satellite (MCSPS), an in-house effort at NASA Lewis Research Center. The various components of this architecture were described by VHSIC Hardware Description Language (VHDL). The simulation consisted of transmitting 27 uplink subpackets /subframe (21 for single point and 6 multicasted), which resulted in 69 downlink subpackets/subframe from the 21 single-point subpackets plus 48 multicast subpackets (6 times 8 dwells).

The architecture was simulated for the duration of three frames and was tested for

-- Proper clock generation
-- Spatial switching (beam enable)
-- Temporal switching (dwell enables)
-- Single-point and multicast connectivity
-- Ping-pong functionality for ACM
-- Ping-pong functionality for dwell FIFO's
-- Adjustable dwell times
-- APF functionality

Possible algorithms for congestion control were not included in the simulation because of the complexity and variety of these algorithms. However, generic statements were used to allow for potential changes in the system requirements; thus, the architecture is fully flexible and reconfigurable. Source codes are available and are well documented for future implementations of the design.

References

1. Ivancic, W.D; Shalkhauser, M.J.; Quintana, J.A.: A Network Architecture for a Geostationary Communication Satellite. IEEE Comm., July 1994.
2. Shalkhauser, M.J.; Quintana, J.A.; Soni, N.J.: Fault Tolerant Onboard Packet Switch Architecture for Communication Satellites: Shared Memory Per Beam Approach, AIAA Paper 94-1101, Feb. 1994 (also NASA TM-106397).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
December 1995

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
Description and Simulation of a Fast Packet Switch Architecture for Communication Satellites

5. FUNDING NUMBERS
WU–235–01–04

6. AUTHOR(S)
Jorge A. Quintana and Paul J. Lizanich

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER
E–9994

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM–107104

11. SUPPLEMENTARY NOTES
Prepared for the 16th International Communications Satellite Systems Conference cosponsored by AIAA, CASI, AAAF, DGLR, and IEICE, Washington, DC, February 25–29, 1996. Jorge A. Quintana, NASA Lewis Research Center and Paul J. Lizanich, Analex Corporation, 3001 Aerospace Parkway, Brook Park, Ohio 44142 (work funded by NASA Contract NAS3–25776). Responsible person, Jorge A. Quintana, organization code 5650, (216) 433–6519.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Categories 17 and 32

This publication is available from the NASA Center for Aerospace Information, (301) 621–0390.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

The NASA Lewis Research Center has been developing the architecture for a multichannel communications signal processing satellite (MCSPS) as part of a flexible, low-cost meshed-VSAT (very small aperture terminal) network. The MCSPS architecture is based on a multifrequency, time-division-multiple-access (MF-TDMA) uplink and a time-division multiplex (TDM) downlink. There are eight uplink MF-TDMA beams, and eight downlink TDM beams, with eight downlink dwells per beam. The information-switching processor, which decodes, stores, and transmits each packet of user data to the appropriate downlink dwell onboard the satellite, has been fully described by using VHSIC (Very High Speed Integrated-Circuit) Hardware Description Language (VHDL). This VHDL code, which was developed in-house to simulate the information switching processor, showed that the architecture is both feasible and viable. This paper describes a shared-memory-per-beam architecture, its VHDL implementation, and the simulation efforts.

14. SUBJECT TERMS
Onboard processing; Congestion; VHDL; Packet switching

15. NUMBER OF PAGES
09

16. PRICE CODE
A02

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT