STORE-AND-FORWARD MESSAGE RELAY USING MICROSATELLITES: THE UOSAT-3 PACSAT COMMUNICATIONS PAYLOAD

Jeffrey W. Ward[1]

One of the most promising applications for small satellites in the 10-50 kg class is store-and-forward message relay. A single store-and-forward message relay satellite in a polar orbit can provide a global communications network carrying electronic mail, digitised voice, images or computer data. With appropriate choice of link characteristics, small, low-cost ground terminals can be used. When designing an inexpensive microsatellite system to provide store-and-forward communications to small ground terminals, the engineer must challenge the standard assumptions made concerning such things as link frequency, modulation techniques, error-control, and multiple-access arbitration.

Beginning with experiments on its UoSAT-2 satellite, the Spacecraft Engineering Research Unit at the University of Surrey (UK) - in collaboration with AMSAT and VITA - has been investigating store-and-forward communications using microsatellites. The UoSAT-2 store-and-forward transponder used a relatively slow 8-bit CPU with only 96 kbytes of message store, but it has been used by stations world-wide, demonstrating system feasibility. The latest experiments undertaken by Surrey will qualify a commercial-capacity microsatellite store-and-forward system. The onboard transponder is based on a 8-MHz, 16-bit, 80C186 CPU, multi-tasking operating software and 4 Mbytes of RAM message store. The UoSAT/SST modular microsatellite bus provides 9600 baud FSK communications links and other support facilities for the store and forward mission. This payload was launched on the UoSAT-3 satellite in January 1990, and is now successfully operating in orbit.

---

1. Research Fellow and PhD candidate, UoSAT Spacecraft Engineering Research Unit, University of Surrey, Guildford, United Kingdom, GU2 5XH.

INTRODUCTION

History of Store-and-Forward Satellites

The desire for real-time telephony spanning the continents and the oceans drove the early satellite communications market. After initial experimentation, little thought was given to the use of low-Earth orbits for communications satellites. Before the exodus to the "Clark belt", however, engineers had demonstrated that store-and-forward message relay using non-geostationary satellites could be useful. In 1958, the first ever satellite communications was a store-and-forward relay using a tape recorder on the SCORE satellite. The second communications satellite, Courier, was also a store-and-forward system, with three on-board tape recorders which could be commanded to record and play back digital or analogue messages. The teletypes of the 1950s and 1960s were not matched to the high-bandwidth, limited-duration communications links provided by a 'moving' satellite, and so the technique was abandoned in favour of real-time relay using GEO.

The expanding use of computers in the 1970s led to renewed interest in store-and-forward message relay - initially through terrestrial networks (e.g. ARPANET). In 1973, MITRE Corporation published a paper proposing a "message courier" satellite which bears striking resemblance to the systems which were eventually implemented a decade later.[1] This was the only published reference to store-and-forward satellites until the 1980s, when microcomputers and solid-state memories became reliable and sophisticated enough to implement the ideas.

The development of operational civilian store-and-forward communications satellites was pioneered by two groups: satellite builders in the Amateur Radio Satellite Service, and relief/development agencies working in developing countries. These groups have a common interest in spreading communications facilities beyond the limits of existing public telephone networks. Amateur Radio satellite builders were designing and operating increasingly sophisticated, inexpensive microsatellites. For example, the UoSAT-1 satellite launched in 1981 carried two microprocessors which could be re-programmed in orbit. At the same time, Y. Pal proposed using a store-and-forward satellite for United Nations communications in equatorial regions.[2] This concurrence of interests resulted in the construction of a Digital Communications Experiment (DCE) for launch on the UoSAT-2 satellite in 1984. The DCE was designed and built by AMSAT in North America, funded by the Volunteers In Technical Assistance (VITA - an international development organisation based in the U.S.A.). The host satellite was built by the Spacecraft Engineering Research Unit at the University of Surrey, which has long-standing ties with the Amateur Satellite Service.

The UoSAT-2 Digital Communications Experiment

The UoSAT-2 DCE was the first modern civilian store-and-forward satellite transponder. It used an NSC-800 CPU, 28 kbytes of program memory, and 96 kbytes of memory for message storage. Despite this simplicity (relative to today's desktop microcomputers), the DCE conclusively demonstrated that a single, small (60 kg) low-cost (approximately £500k) satellite could be combined with inexpensive groundstation equipment to form an effective global communications network. Fixed stations in England, the U.S.A., Australia, New Zealand,

the U.S.S.R., West Germany, Italy, South Africa, and Nicaragua, as well as isolated stations in Pakistan and the Antarctic, regularly use the UoSAT-2 DCE. During the winter of 1988, the DCE was used for the first store-and-forward satellite communications from the Antarctic, and in 1988 and 1989 the ad-hoc network of Amateur Radio stations using the DCE passed 4 Mbytes of traffic.

Two other store-and-forward microsatellites were launched in the 1980s. In 1985, the Global Low Orbiting Message Relay satellite (GLOMR) was released from a Space Shuttle Getaway Special Canister. GLOMR was built by Defence Systems Inc. for DARPA, and little technical information about the mission has been made publicly available. The onboard computer was from the RCA 1802 family, and the communications links were 9.6 kbps phase-shift keying. The other store-and-forward mission was FUJI-OSCAR-12, an Amateur Radio communications satellite launched on a Japanese H-1 launcher in 1986. The FO-12 transponder uses an NSC-800 CPU and 1 Mbyte of dynamic RAM. Between 1987 and 1989, FO-12 was available for general access to all stations in the Amateur Radio service, and was used by more than 300 groundstations.

As a result of three successful store-and-forward microsatellite missions in the 1980s, several similar missions have already been launched in 1990, and more are planned. Using the experience gained from the UoSAT-2 DCE and the FO-12 store-and-forward mission, satellite engineers from AMSAT-NA and the University of Surrey independently developed two new store-and-forward transponder designs. Five of the 6 microsatellites launched on Ariane V35 in January were equipped with these "second-generation" store-and-forward communications transponders, which involve new microprocessor and memory technologies. Four of these were the AMSAT-NA Microsats. The fifth was Surrey Satellite Technology's UoSAT-3, carrying the PACSAT Communications Experiment.

THE PACSAT COMMUNICATIONS EXPERIMENT HARDWARE

The primary payload on the UoSAT-3 satellite is the PACSAT Communications Experiment (PCE). The University of Surrey's company Surrey Satellite Technology designed the payload, under a development contract with VITA, who propose to use similar transponders on a dedicated VITA store-and-forward satellite. Thus, the UoSAT-3 PCE is proving hardware and software for upcoming commercial store-and-forward missions. To provide a realistic traffic load for these tests, the PCE operates in both the Amateur Satellite Service (funded by AMSAT-UK) and on experimental non-amateur frequencies licensed by the FCC for VITA's pilot tests.

The specific integrated circuit devices chosen for the PCE were selected to provide the desired level of communications service (with some margin for expansion), to be compatible with existing software and hardware development systems, and to support widely-used link-level communications protocols. Demonstrated reliability in space was not one of the selection criteria, since 8-bit microprocessors and asynchronous communications devices were not deemed capable of serving upcoming operational missions. A miniaturised design using surface-mounted components and multi-layer PC boards was also ruled out. These techniques would have significantly increased the cost and development time for the transponder. Since the UoSAT-3 modular bus provided 1600 cm^2 area for the transponder PCBs, we could afford the convenience of

through-hole mounted devices on double-sided PCBs. A block-diagram of the final design is shown in Figure 1.

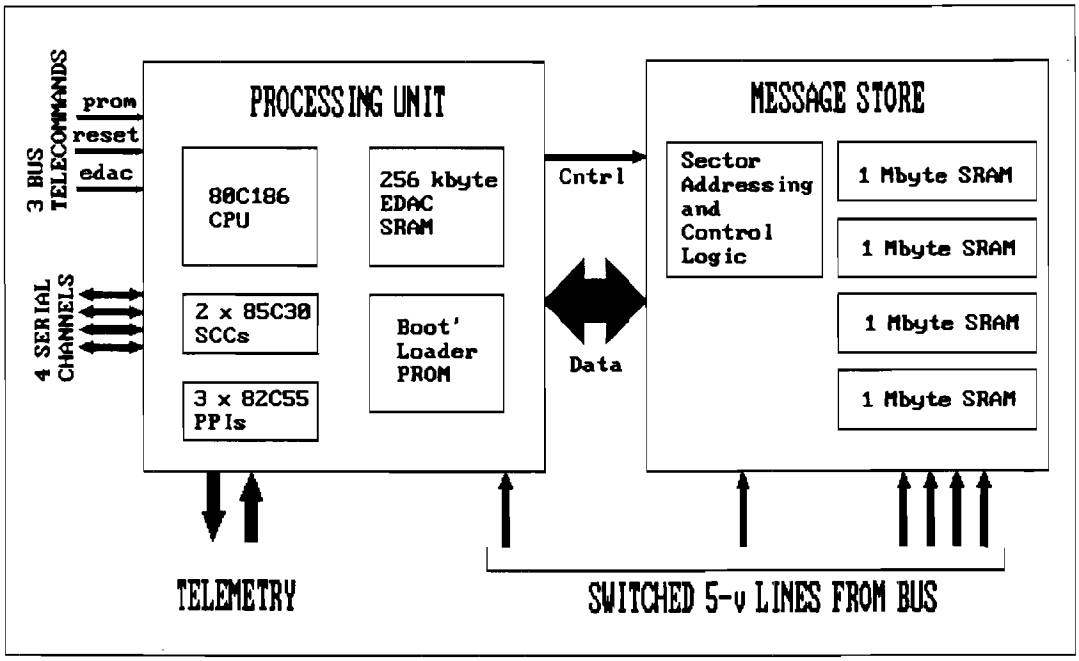

Fig. 1 - Block Diagram of UoSAT-3 PCE

Central Processing Unit

We selected the Intel 80C186 CPU over two other contenders, the NEC-V40 and the Intel 80C188. All three of these processors are software compatible with the Intel 8088 - a great advantage when one considers the wealth of development tools resulting from the popularity of the IBM PC. The 80C186 and 80C188 CPUs provide on-chip counter/timers, memory select logic, clock oscillators and direct memory access (DMA) controllers. These built-in peripherals reduce the complexity of the hardware design, increasing the overall reliability of the payload . Since UoSAT-3 was being designed for an early 1989 launch, device availability influenced the final selection. The V40 was available only in a standard temperature range component. Only samples of the 80C188 were available. The 80C186, though, was available in a commercial temperature range device, with the prospect of a full MIL-883 device to come.[1]

Program Memory

The 80C186 requires a full 16-bit wide memory data bus, rather than the 8-bit data path needed by the V40 and 80C188. This increases the number of components in the circuit, but provides a greater memory bandwidth.

---

1. The M80C186, a MIL 883C compliant version of the 80C186 has since become available from Intel.

We needed enough program memory to run a multitasking operating system and several tasks, and estimated that 256 kbytes would serve. CMOS static RAMS were chosen over dynamics, due to their lower power consumption and lower single-event upset sensitivity. Error-detection and correction (EDAC) circuits similar to those used on the UoSAT-2 DCE were chosen.[3] These circuits use a 4-bit Hamming code to protect the 8 data bits of each byte. Since a 16-bit data bus was to be protected, we investigated more efficient 16-bit EDAC systems, but all of those identified required significant additional hardware.[1] We chose 32k X 8-bit CMOS SRAMs to fit 256 kbytes of RAM with Hamming code EDAC into the available area.

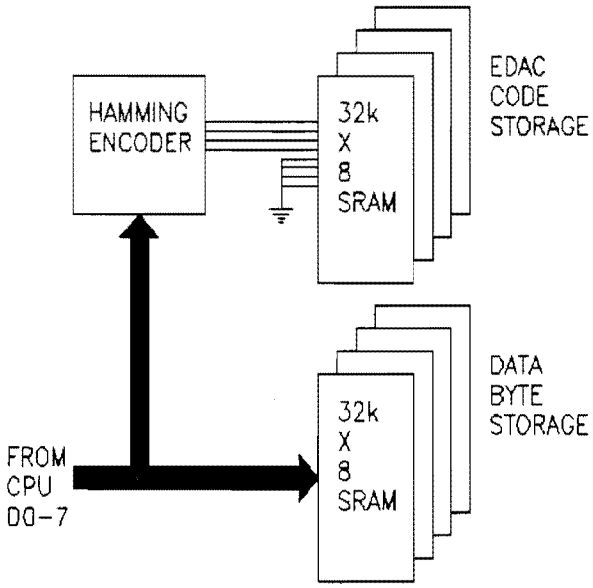

Fig. 2 - Block diagram showing half of the EDAC RAM.

Theoretically we needed only 24 bits of memory (three 8-bit wide SRAMs) for a 16-bit data bus and two 4-bit Hamming code words. In practice, however, Intel processors do not always write 16-bit data words; they sometimes use only half of the bus. To accommodate this byte write cycle, each of the Hamming code words must be independently updatable, so they cannot both be stored in the same SRAM. Hence, the PCE program RAM is implemented as two independent 8-bit, EDAC-protected memories. In each half of the memory, one 8-bit wide SRAM stores the data bits, and another stores the 4 Hamming code bits. The 4 remaining bits of the code RAM are unused (Fig. 2).

Where possible, the PCE design does not rely on a single type of component from a single manufacturer. The PCE program memory uses 32 kbyte SRAMs from 3 different manufacturers (Hybrid/Hitachi, EDH, and NEC) to provide some protection against systematic failure. If one of

---

1. CMOS versions of VLSI EDAC chips are now becoming available. For example, the 54PCT633 from Performance Semiconductor Company is being used in one of the onboard computers on the UoSAT-F microsatellite.

the EDAC decoders or code word RAMs fails, the EDAC system can be bypassed. This would allow some operation of the payload, albeit without protection against SEUs.

Serial Input/Output

The serial I/O devices selected for the UoSAT-3 PCE had to support both asynchronous communications (used for the onboard data handling network), and synchronous communications (used for the packet radio user access protocols). The operating system chosen for the PCE already included software drivers for the Zilog 80C30 serial communications controller (SCC), and this chip would have been the natural choice. Unfortunately 80C30s were not available in time for the UoSAT-3 mission so the similar 85C30 was used. Neither of these chips is a perfect match for the 80C186, and an undesirable amount of 'glue logic' was used to interface the SCCs to the Intel bus. The two 85C30 SCCs provide 4 bidirectional I/O channels, which can be operated at any standard speed from 1200 bits/sec. to 64 kbits/sec. Two channels are used for synchronous communications with groundstations, one of the remaining channels is used for the 9.6 kbits/sec. onboard data handling network, and the fourth is used as a dedicated link to the spacecraft's telecommand subsystem. The synchronous communications channels are connected to the 80C186 DMA controllers, providing simultaneous DMA transfers for uplink and downlink. Multiplexers in the PCE and in the UoSAT-3 bus allow re-assignment of serial input sources and output destinations under on-board or groundstation command.

Parallel Input / Output

The PCE uses parallel I/O to receive data from the navigation magnetometer, to request and receive spacecraft telemetry measurements, to set the battery charge regulator, and to control the 4-mbyte message storage memory. Three Harris 82C55 parallel I/O devices provide 72 programmable lines. The 82C55 was used on the UoSAT-2 DCE and continues to perform flawlessly after five and a half years in orbit.

Message Storage Subsystem

We calculated that several megabytes (mbytes) of message storage would be required on an operational store-and-forward satellite. We considered only CMOS SRAM for this storage, although tape-recorders or bubble memory could have been used. Data from UoSAT-2 shows that SRAMs are reliable, and that the SEUs - even in large devices - can be corrected by software EDAC. Thus, SRAMs have no disadvantages. They have no moving parts, consume negligible power when idle, and require no special interface circuits. The PCE includes four mbytes of message storage SRAM, enough to store just over 2000 pages of compressed text.

Although we did not want to manufacture our own PCBs with surface mount components, several mbytes of through-hole mounting SRAM simply would not fit in the available area. The solution was to use hybrids, consisting of four or eight surface mounted SRAMs on a PCB carrier. The carrier then mounts to the host PCB with standard through-hole pins. Hybrid Memory Products manufactured the hybrids for the PCE. We used both 256 kbyte vertical-mounting single-inline packages and 128 kbyte dual-inline packages. The surface mounted monolithics used on the hybrid carriers are Hitachi and Toshiba 32 kbyte SRAMs. Four mbytes of

hybrid SRAM, with associated buffering, fit easily on the 300 mm X 270 mm PCB. [1]

The 80C186 CPU cannot directly address more than one mbyte of memory. Therefore, the message storage RAM must be paged into the main memory space or connected through some other interface. Paging into the main memory space has the advantage of fast access and operational flexibility. The disadvantages of paging are close electrical coupling of the message memory with the CPU main address and data bus (decreasing reliability of this critical bus), and increased physical complexity of the interface (16 data lines, 10 or more address lines and handshaking). As an alternative to paging, we used a bidirectional parallel I/O port between the PCE CPU and the message memory. Accessing the message memory is similar to accessing a disk-drive: the CPU writes a sector address to the interface port, and then reads or writes a block of data from the storage RAM. The interface uses eight data lines and five handshaking lines provided by one of the 82C55 parallel I/O chips.

Within the message storage subsystem, the four mbytes are divided into four 1-mbyte banks. Each bank has its own switched power supply from the spacecraft bus, and the banks are isolated from one another by bidirectional buffers. Internal subdivision means that none of the memory chips is a single point failure node for the entire memory system. If destructive latchup occurs, perhaps shorting the ground and power rails, the offending bank can simply be turned off. This also allows banks to be turned off when not in use, either for power savings or as protection against radiation dose gate pollution.

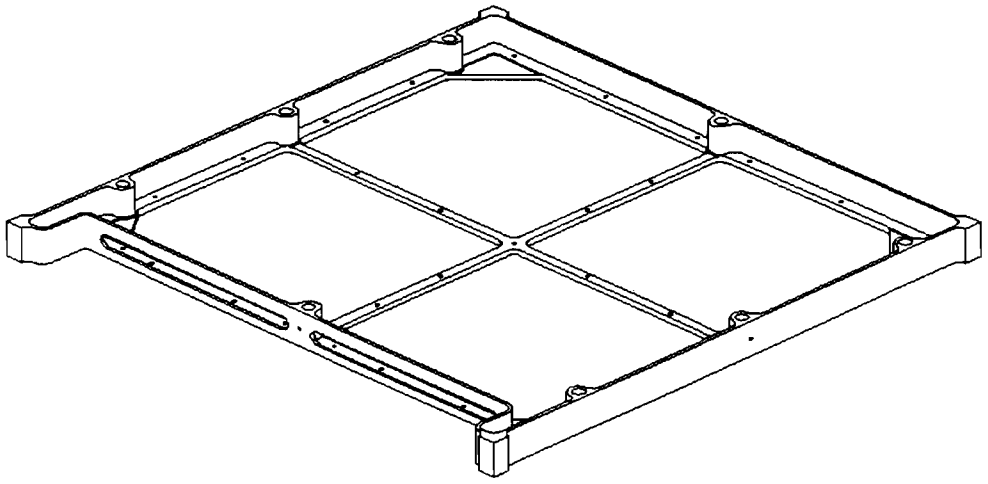

Figure 3 - UoSAT Module Box, approximately 300 X 300 mm.

1. The advance from 32 kbyte to 128 kbyte CMOS SRAMs since the design of UoSAT-3 will allow us to quadruple storage capacity without changing volume. The UoSAT-F message storage subsystem will provide 16 mbytes of RAM.

Mechanical Characteristics

The PCE comprises two 290 X 320 mm PCBs, each of which fits into a standard UoSAT module box (Fig. 3). One PCB carries the CPU, program memory and I/O systems. The other carries the message storage memory. The main spacecraft wiring harness interconnects the PCE PCBs and connects them to the UoSAT-3 spacecraft bus.

Electrical Characteristics

It is difficult to reliably measure the current consumption of a device which has a high-frequency variable duty cycle - such as an 8-MHz CPU and memory subsystem. Filtering characteristics and response time of the telemetry system make absolute measurements difficult to obtain. The table (Fig. 4) provides approximate indication of power consumption in the CPU module during different operating regimes. Running the bootloader, the CPU is executing from ROM and continuously polling the all four I/O channels. The kernel, on the other hand, halts the CPU when there are no tasks ready to run. Thus, the bootloader measurements show the highest power consumption for the payload, while the lightly-loaded kernel shows the idle condition.

Operating Mode	PCE Current (mA at 5-v)	PCE CPU Temp. (C)	'Ambient' Temp. (C)
Bootloader	130	20	7
Kernel	60	17	7

Figure 4 - PCE CPU Current Consumption and Temperature.

Thermal Characteristics

When we were preparing flight model PCE for thermal vacuum tests (conducted at the Royal Aerospace Establishment), we were made aware of a general concern about heat dissipation in onboard computers. Gough and Wolliscroft state that "heat dissipation can be a major problem on both rocket and satellite flights. Even CMOS microprocessors when running at high clock rates can dissipate a few hundred milliwatts...this requires thermally conducting heat sinks to the main structure."[4] The only heat conducting path from the PCE 80C186 to the spacecraft structure is through the 67 pins of the pin-grid array to the PCB. Although we had experienced no problems with the two NSC800s on UoSAT-2 or the CDP 1802s on UoSAT-1 and -2, we wondered if the 8 MHz clock speed of the 80C186 might result in undesirable heat buildup. To answer this question, a temperature sensor was added directly to the 80C186 CPU. Temperature measurements made during thermal vacuum tests and also in orbit show that the CPU runs between 10 and 13 degrees C hotter than the surrounding stack; well within the CPU's +85° C maximum temperature limit.

THE PACSAT COMMUNICATIONS EXPERIMENT SOFTWARE

A store-and-forward 'transponder' is actually an appropriately equipped onboard computer programmed for message switching and packet communications. In this case the design, selection and implementation

of the system software are as important as the flight hardware. As with both previous UoSAT satellites, we load all operational software after launch. There is only a bootstrap loader in ROM, and no programs are kept loaded in RAM during the launch. This approach has several desirable consequences. Because the software can be uploaded after launch, it can be modified to take advantage unexpected experimental opportunities or to respond to changing mission characteristics. Especially when the time between mission commencement and launch is short, it is important to be able to continue software development after launch. Because we do not need to keep software loaded during launch, we can launch the satellite without any onboard computers running. The power to all onboard systems is routed through a switch which remains open until the satellite is safely separated from the launcher, at which time the satellite is under the control of hardware telemetry and telecommand systems. This is a simple and safe configuration to adopt, and has significantly eased the concerns of primary payload customers considering UoSATs as secondary payloads on their launches. For all of these reasons, the only software on the PCE at launch is a robust bootstrap loader in PROM. Both the multi-tasking operating system and the communications application software are loaded after the satellite is in orbit.

The Quadron Multi-Tasking Operating System

The UoSAT-2 DCE uses a single application program and no underlying "operating system", but the flexibility of multi-tasking systems was demonstrated on the AMSAT Phase-III satellites and on the UoSAT-2 OBC. We wanted a multi-tasking operating system for the new PCE onboard computer. H. Price - a principal in UoSAT's store-and-forward communications experiments since UoSAT-2 - made available an 80186 operating system tailored to real-time communications applications. The operating system was developed by Price and others at Quadron Service Corporation[1], to run on the IBM Real-time Interface Co-Processor (RIC), an 80186-based communications card for IBM PC family computers. Through a memorandum of understanding between Surrey Satellite Technology and Quadron, the operating system was tailored to the UoSAT-3 PCE (as well as to the AMSAT-NA V-40 onboard computer).

The operating system - Quadron Communications Facility or qCF - provides a preemptive multi-tasking scheduler, inter-task communications using named pipes, and comprehensive drivers for character and frame-based I/O. qCF provides a powerful development environment for applications programmers. Applications are written in the C language, compiled and linked by an unmodified Microsoft C compiler. For the first time, onboard computer programmers can write in a widely known, high-level programming language, calling operating system services through a rich applications programmer's interface. Tasks can be designed and debugged on a standard IBM-compatible PC, using source-level symbolic debuggers. For more thorough ground testing, the multi-tasking kernel and all tasks can be run on an IBM RIC card and monitored in-situ by the developer's PC. Then, by linking the program with special libraries, a satellite-ready executable file can be created.

---

1. Quadron Service Corporation, 133 E. De La Guerra, Suite #10, Santa Barbara, CA 93101.

The effectiveness of this approach has already been proven. Since launch, a team of programmers in the U.K. and the U.S.A. (most working part time) has produced and debugged the following:

AX.25 protocol driver which supplies level-2 communications link facilities to any other task;

File system for the mass storage memory, accessed through C library functions (e.g. fopen), with software EDAC protection against SEUs;

Housekeeping tasks customized for each of the five microsatellites running qCF;

UoSAT-3 Cosmic Particle and Total Dose Experiment server which uses the onboard data handling network;

Store-and-forward message switching system (alpha version).

Figure 5 shows the tasks currently running on the PCE - comprising nearly 256 kbytes of executable code.

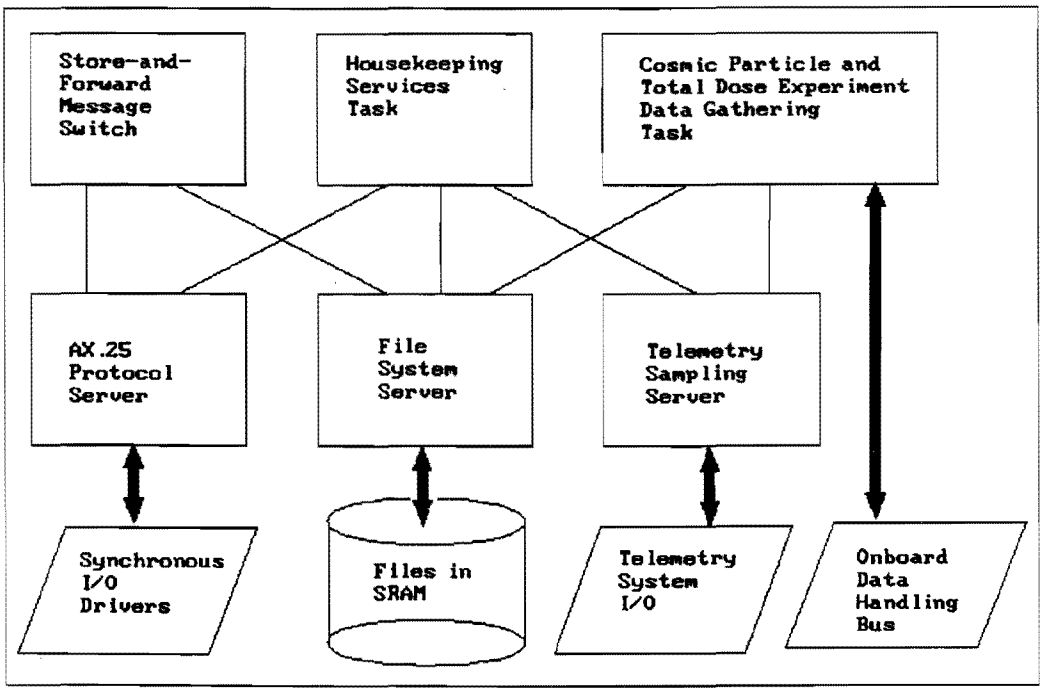

Figure 5 - PCE Software Tasks and Inter-Task Links

PROTOCOL DEVELOPMENT

When developing a low-cost satellite communications system, it is important to take advantage of existing components and subsystems. This is especially important when offering a facility in the Amateur Satellite Service. Although in some areas we have been forced away from proven, available groundstation systems (e.g. in our choice of 9.6 kbit/sec. FSK modulation), available items are suitable in other areas. The AX.25 communications link protocol is a good example of this. AX.25

is a version of the CCITT X.25 Level 2 protocol which has been modified to work in a multi-station radio environment.[5] Many relatively low-cost AX.25 packet assembler/disassemblers are available, and the protocol has had wide acceptance outside of the amateur service.

Using both the connected mode and datagram features of AX.25, we are implementing a suite of protocols tailored to satellite store-and-forward message relay. The satellite's downlink is, naturally, a broadcast medium, and so a point-to-multipoint Broadcast Protocol has been developed to transmit messages of wide interest (e.g. satellite ephemeris). On the uplink, and for certain downlink transactions, a virtual circuit protocol is required. For this we are using a full-duplex binary file transfer protocol on top of the AX.25 connected mode. In both instances the satellite acts as a message transfer agent in an electronic mail system, and standard message encapsulation headers have been defined. User agent software in the ground terminals can scan the message data-base on the satellite, to find messages of interest to specific users. A query and selection system has been defined for this purpose.

These protocols have all been tailored for the specific environment in which they will be used. As a result, they are not intended for open interconnection with terrestrial networks, but they use uplink and downlink bandwidth, onboard storage, and onboard processing power efficiently. The author and H. Price designed and specified these protocols, and are currently implementing both spacecraft server and groundstation client software.[6]

Multiple-Access Protocols

The uplinks of a store-and-forward communications satellite are multiple-access channels, with many stations wishing to share limited bandwidth. A multiple-access protocol must be selected to manage access to this uplink in an efficient and/or equitable manner. While multiple-access protocols for GEO satellite networks, packet-radio networks, and hard-wired computer networks have been studied extensively over the years, the case of a low-Earth orbiting message store-and-forward satellite is sufficiently different to merit its own investigation, and this is the subject of the author's doctoral research.

The simplest approach to channel sharing is for each station to ignore the others and transmit whenever it needs to. When two stations transmit simultaneously, both frames are destroyed. The destroyed frames are not acknowledged, and they are eventually re-transmitted. This is the pure ALOHA protocol, developed for a packet radio network at the University of Hawaii in the 1970s.[7] The classic throughput analysis of an ALOHA shows that the maximum throughput is just over 18% of the channel bandwidth. By synchronising packet transmissions, slotted ALOHA can achieve 36% throughput. Neither of these figures is acceptable when one considers a satellite with a 10-minute visibility window.

One can increase the throughput efficiency of random access channels by avoiding collisions. Unfortunately, the simplest collision avoidance scheme - carrier sense multiple access - is not useful for a store-and-forward satellite system, since contending ground stations will not usually be able to hear one another's carriers (due to frequency separation or geographical distribution). Busy-tone multiple

access (BTMA) could be used on a store-and-forward satellite; the satellite would transmit a busy tone on the downlink whenever the uplink was occupied, and stations wishing to transmit would only do so when the busy tone was not present. Unfortunately, the upper bound on throughput of a BTMA uplink for a satellite in an 850 km LEO is 49.8%. This does not seem high enough to justify inclusion of a busy tone generator on the satellite and detectors in the groundstations.

To increase uplink throughput efficiency, collisions must be reduced or eliminated by assigning a specific fraction of the uplink resources to each groundstation. Time division multiple access (TDMA) assigns all of the uplink RF bandwidth to one groundstation for a limited time, whilst frequency division multiple access (FDMA) divides the RF bandwidth into several uplink channels and assigns each channel to one groundstation. TDMA and FDMA protocols can use either fixed-assignments or demand-assignments. Fixed assignments, wherein each station is permanently assigned a specific time or frequency slot, are inappropriate for LEO store-and-forward message relay, because the number and identity of the groundstations in the satellite footprint changes continuously. Contention-free access for a LEO store-and-forward satellite must use demand assignment, providing variable uplink allocations to groundstations as they enter and leave the satellite's footprint.

A central station can control channel assignments, or all stations can implement a distributed control algorithm. Distributed control protocols are require broadcast channels (i.e. GEO satellite transponders), and are not suitable for LEO message store-and-forward satellites. LEO store-and-forward satellites are better equipped for centrally controlled demand assignment protocols, with the onboard computer assigning uplink capacity. The primary argument against central control in communications networks is that failure of the central node means failure of the network. This argument carries no weight in the present situation, because the network already depends upon the satellite's onboard computer for message storage and retrieval. Using this computer to manage uplink assignments does not create a new single-point failure node.

In a centrally controlled network, stations may only transmit their data packets when they have been granted channel access by the central station. Several methods can be used by the central station (or 'controller') to determine which station should be granted access. The controller can poll all stations to determine which have data to transmit. Polling all stations is inefficient if the number of stations to be polled is large and the number of stations with data to transmit is small. An alternative to polling is probing, in which the controller polls groups of stations rather than single stations.

A third alternative is split-channel reservation multiple access (SRMA).[8] SRMA uses two transmission channels from the stations to the controller. A station with data to transmit first sends a request packet to the controller on the request channel. Only if the controller grants the request does the station transmit data on the message channel. The request channel is a random-access channel, and the message channel is contention-free. The request and message channels may be separate physical channels by frequency splitting, or they may be multiplexed onto a single physical channel using TDMA. The original

paper proposes separate physical channels, and analyses this case in detail. The important result of these calculations is that SRMA can achieve throughput efficiency of 80% or higher.

The author intends to investigate the use of SRMA for LEO store-and-forward satellites. The onboard computer will act as network controller, transmitting a special frame on the downlink to signal the opening of the request channel. The request channel will be a limited duration time division of the uplink communications channel. Upon hearing this frame, groundstations wishing to gain the communications channel will transmit their reservation request frames in an ALOHA manner. In the worst case, none of these frames are received and the cycle is repeated. Otherwise, the onboard computer receives one or more reservation requests and grants the uplink channel to one of the requesting groundstations. The selected groundstation has contention-free access to the satellite until its message has been transferred, and then the cycle is repeated.

If more than one reservation request is received by the satellite, requests can be sorted by the onboard computer. The sorting algorithm might grant high priority access to emergency transmissions, portable ground terminals, or stations which are about to lose the satellite's footprint.

An SCRMA protocol for UoSAT-3 is still in the early stages of design. We will use the UoSAT-3 PCE to test practical implementations of the protocol, and ground-based simulations to evaluate scenarios which cannot be generated in orbit. We hope to achieve throughput efficiency well above 50% (i.e. higher than any of the random-access protocols) without significantly increasing the complexity or expense of the groundstation and satellite hardware.[1]

ON-ORBIT COMMISSIONING

UoSAT-3 was launched on the first flight of the Ariane Structure for Auxiliary Payloads (ASAP) - Ariane V35 - with SPOT-2 the primary customer. UoSAT-4 and the four AMSAT Microsats were also on this ASAP.

The most urgent task for the PCE was to collect data from the Cosmic Particle Experiment (CPE) and Total Dose Experiment (TDE) on UoSAT-3. The first data collection software, confirmed correct operation of the PCE, the CPE/TDE and the onboard data sharing bus. This task (running under qCF) collected periodic measurements from the CPE/TDE and broadcast them in simple ASCII format using the 9.6 kbits/sec. FSK downlink.

The initial software also included a digital packet relay facility which could be used by radio amateurs testing their 9.6 kbits/sec groundstation equipment. (This link speed is 8 times faster than any used regularly on previous AMSAT satellites.) Much to our surprise, over 30 amateur stations from all parts of the world accessed the satellite during the first two weeks of these experimental operations.

---

1. This research is funded by the United Kingdom Science and Engineering Research Council grant GR/F 65576.

Some stations used simple omni-directional antennas; providing limited verification of our link budgets.

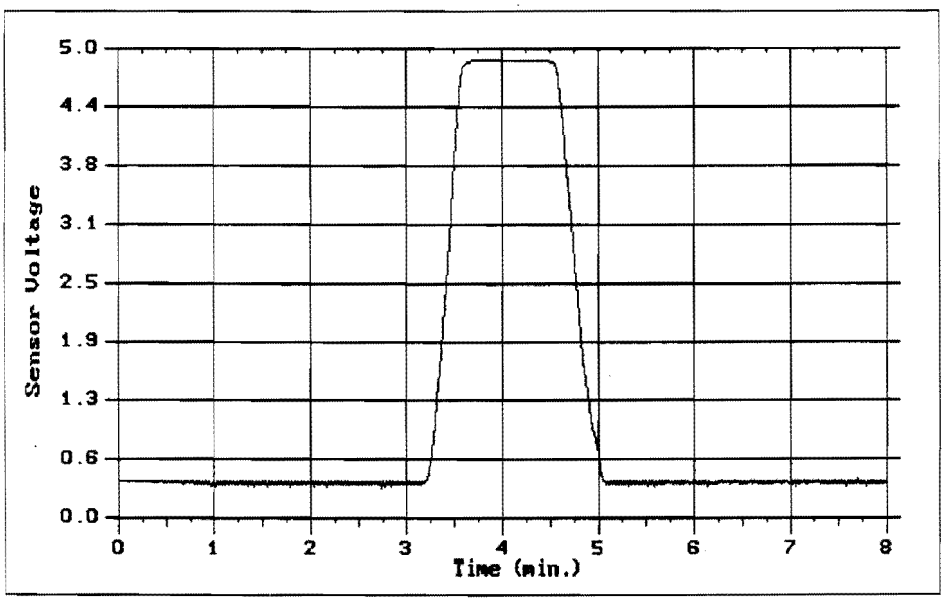

Figure 6 - Sun Sensor Data Survey Collected by PCE.

We next installed the RAM file system and a more complex CPE/TDE data collection task. This task collects data from the CPE/TDE every five minutes, and stores each day's data in a separate RAM file. Over thirty kbytes of data are collected every 24 hours, and downloaded using the Broadcast Protocol. We also loaded a housekeeping task which samples telemetry in real-time and can store telemetry sampled at any point in the orbit. (Figure 6 shows a telemetry survey taken by this software, a short-duration, 1-second resolution survey of the sun detector, showing an encounter.)

The latest phase in commissioning was the installation and testing of the full AX.25 link drivers, and a preliminary version of the store-and-forward message handling software. This was completed in late July.

Complete commissioning of the store-and-forward transponder software is expected shortly, with the inclusion of the data-base query server for message selection. Once this software has been installed, the transponder will be opened for access by radio amateurs and by VITA groundstations.

CONCLUSIONS

The UoSAT-3 PACSAT Communications Experiment has been successfully commissioned in orbit. With a modern 16-bit microprocessor running at 8-MHz, it is one of the most powerful onboard computers ever launched on a civilian satellite. The qCF multitasking operating system allows this power to be used by groundstation programmers working in a well-known high-level language. The PCE CPU design, once proven in orbit, will be the basis for the standard onboard computer in several SST/UoSAT modular satellites currently under consideration.

During the remainder of the UoSAT-3 mission, the PCE will be used to experiment with store-and-forward communications. The University of Surrey will investigate multiple access protocols and spacecraft design issues. Other groups, specifically the Volunteers In Technical Assistance, will be use the transponder for operational demonstrations, showing how dedicated store-and-forward microsatellites can provide communications to areas not sufficiently covered by existing telecommunications networks.

REFERENCES

(1) Brandon, W.T., 1973, **A Data Courier Satellite System Concept**, 21st International Convention on Communications, 8-13 October, Vol:21, pp. 921-935, Genova Italy.

(2) Pal, Y., 1982, **International Round Table on Alternative Space Futures and the Human Condition**, 8-10 March, New York USA.

(3) Ward, J.W., 1988, **Observations of Single-Event Memory Upsets on the UoSAT-2 Satellite,** Proceedings of the 2nd Annual AIAA/USU Conference on Small Satellites, 18-21 September, Logan UT USA.

(4) M.P. Gough and L.J.C. Wolliscroft, **Microprocessors in Space Instrumentation**, Space Technology, Vol 9, No 3, 1989.

(5) Fox, T., 1984, **AX.25 Amateur Packet-Radio Link-Layer Protocol**, American Radio Relay League, Newington CT, USA.

(6) Complete protocol specifications will be published in the Proceedings of the 9th ARRL Computer Networking Conference, 22 September, 1990, London, Ontario, Canada.

(7) Abramson, N., 1977, **The Throughput of Packet Broadcasting Channels**, IEEE Transactions on Communications, January, Vol:15 No:1, pp. 117-127.

(8) Tobagi, F.A., and Kleinrock, L., 1976, **Packet Switching in Radio Channels: Part III**, IEEE Transactions on Communications, August, Vol:24 No:8, pp. 832-844.