15. A NEW ACCELEROMETER RECORDING SYSTEM FOR SHUTTLE USE

Dr. Byron Lichtenberg, Payload Systems, Inc.

ABSTRACT

Microgravity investigators are interested in enhancing the capabilities and improving the information return from accelerometers used in microgravity research. In addition to improving the accelerometer sensors, efforts should be directed towards using recent advances in microprocessor technology and system design techniques to improve:

a) Sensor calibration and temperature compensation;
b) Online data display and analysis;
c) Data reduction and information storage.

Results from the above areas of investigation should be combined in an integrated design for a spaceflight microgravity accelerometer package.

Good morning, thank you, it's nice to be here. I appreciate the opportunity to come and talk to all of you. As Dan said, we are not quite developing a new accelerometer, we are really developing an accelerometer data processing and recording system, which you are going to see in just a minute. It stemmed from some of our experiences on Spacelab 1 and some of the experiences that a lot of you have had out there in trying to get timely processed accelerometer data from the space shuttle, and from your experiments as well, and trying to figure out how to correlate these to particular operations that went on during experiment. We are going to talk about a system called Python. For all you NASA people, don't try to figure out what that stands for because it's not an acronym, it's just a name. Python is a microgravity accelerometer system made up of a variety of parts. Figure 1 shows a list of some of the design goals that we set for ourselves when we tried to put it together.

- Designed for use on the space shuttle.
- High performance accelerometers with temperature compensation to measure 1 micro-g in three orthogonal axes.
- Detachable sensor head allows micro-g acceleration measurements at the experiment location.
- Spaceflight-qualified version available which fits inside a middeck locker.
- Stand-alone system significantly reduces shuttle interface documentation and verification, and permits software refinements until shortly before launch.
- Removable high capacity data storage cassettes retain data during mission and allow early postflight data retrieval.
- Downlink data transmission capability.
- User-friendly system minimizes training time.
- Modular structure allows customization for a wide range of experiment requirements.
- Fully supported through mission integration and flight by PSI space experiment engineering personnel

FIGURE 1. PYTHON MICROGRAVITY ACCELEROMETER SYSTEM

First, it is designed for use on space shuttle, although that does not preclude use on the space station or free-flyers; it was designed initially for crew attended use on the shuttle. There will be a space flight qualified version because all the pieces we have used here have been flown or are going to fly and are space flight qualified. Because I'm a user, we wanted to make sure it's user friendly --that helps you understand how it works; it is easier for training people that use it, and for making changes and modifications. It has a detachable sensor head because we initially targeted this for the middeck, or spacelab module experiments (indoor sports where people are doing a lot of different experiments). It is a stand-alone system which significantly reduces interaction with the shuttle general purpose computers or the spacelab computer. This means you can do late software development and make late changes in your equipment just prior to flight, carry a small micro floppy disc and run your software without having to worry about going through all the extensive shuttle and spacelab verification procedures. We incorporated a data storage system, which is a magnetic tape cartridge system so that you can bring your data back immediately after flight. We will have downlink data transmission capability through the shuttle or spacelab, and Figure 2 shows what led us to doing this. The problem that we saw early on is that there is no available system now that can be used to control experiments, and that can be used on a variety of experiments with generic equipment. Also, there is no flight-certified data acquisition and display system that stands alone. A lot of systems do data acquisition, data storage, or downklink but there isn't one unified system that allows the crew to take a look at processed data during the flight. As I said before, we want to stay away from the Shuttle GPC and spacelab computers because of their limited availability, limited resources and capabilities as well as the difficult problem of interfacing with them.

When we take that process and apply it to our problem today, which is materials science and accelerometer and acceleration measurements, a little background is in order. There have been a lot of different systems developed to measure acceleration in the micro-g or

THERE IS NO AVAILABLE SYSTEM TO CONTROL SPACE EXPERIMENTS THAT CAN BE USED BY A VARIETY OF DISCIPLINES THROUGH CUSTOMIZATION OF A GENERIC GROUP OF EQUIPMENT.

THERE IS NO FLIGHT-CERTIFIED DATA ACQUISITION AND HANDLING SYSTEM THAT ENABLES REAL OR NEAR REAL-TIME PROCESSING AND DISPLAY THAT IS INDEPENDENT OF THE SHUTTLE OR SPACELAB COMPUTERS - BOTH OF WHICH ARE EITHER LIMITED IN THEIR PERFORMANCE OR ACCESSIBILITY AND REQUIRE EXTENSIVE INTERFACE VERIFICATION.

FIGURE 2. PROBLEM STATEMENT

milli-g ranges, but all of them have their own internal accelerometers that are not available to the users. The user has to rely on someone else's data. Remember we heard yesterday that some of the ASIP and HIRAP measurements were taken near the center of the mass of the vehicle and do not necessarily correspond to what a particular experiment might see if it is located, for example, in the middeck. So, a lot of these users develop their own systems and that leads to a lot of expense, or unnecessary duplication.

Somtimes they have an inappropriate bandwidth. As Dr. Hamacher mentioned yesterday, his scheme was to sample the peak acceleration during any one second, but he can't really reconstruct any time history from that. All he can do is put an envelope around the maximum acceleration values.

Another problem is that it takes a long time to get data back. Some of the materials science experimenters on Spacelab 1 and 2 waited over a year to get their reduced data.

Python is a modular, generic system that relies on commercially available equipment. It is built around a bus system with a user interface computer (we are using a GRiD case III, which I have here and will talk about later). We are using an STD bus so we have a distributed system that allows user interactive and iterative design. We can support a variety of sensors; we are not locked to any one accelerometer or thermocouple or other type of sensor. We are designing it for science and engineering applications, not for safety of flight or reliability issues, so we think we can push NASA a little bit in their flight qualification. Everything that we are using now has been flight-certified or flight-qualified or there is a version that's flown, so we are not looking at the tremendous problem of flight qualification. We want to stress the fact that this is not a safety-of-flight type of equipment and, therefore, we should be able to use more state-of-the-art equipment.

As I said, we use the STD bus modules and the STD system, which are widely commercially available. We do have customized software modules available to do a variety of things and we are concentrating on

distributed processors. As shown in Figure 3, our design philosphy is reverse from some of the other programs. We feel that now most hardware is really cheap, it's the custom-developed software that is much more expensive, so our philosophy is to separate these functions. Our data acquisition, data processing, and data handling systems design should be able to accommodate most hardware. We are not locked into any particular chip or particular system. We are trying to move beyond portable software, which in the past has been transferable from one machine to another. We did some software development that runs on a Compaq or IBM AT and they run without modification on the briefcase GRiD machine. This system is able to support different levels of hardware without being locked into any one system, as shown by the three goals on the bottom of Figure 3. We are going to do verification and validation on individual system elements such as plug-in modules, which makes it easier to customize to individual user needs.

Figure 4 shows some of the system elements that we are proposing for our Python system. The heart of the system is the STD data acquisition and control system that includes generic accelerometers, generic data storage devices, a generic user interface computer, and downlink interface.

For accelerometers, Sundstrand QA-2000's are the initial choice; because we feel that they have the best performance/cost ratio, as well as good power, weight, volume, frequency characteristics, and g-sensitivity. I'm not here to sell Sundstrands; if somebody wants to donate some MESAs, we can accommodate them. But if we are going to do our own development, then the Sundstrands look pretty good. Figure 5 shows some of the accelerometer characteristics on which we based our decision. The resolution accuracy that is claimed for the Sundstrand is 1 micro-g, and that is probably good enough for what we are doing on the shuttle or on the space station for quite awhile. It has good long-term stability, and they judiciously pick their accelerometers and test them so that they are well characterized in terms of the thermal characteristics. Sundstrand has a 4th order model that gives some idea of the histories of the temperature dependence of dc offset and thermal bias coeffici-

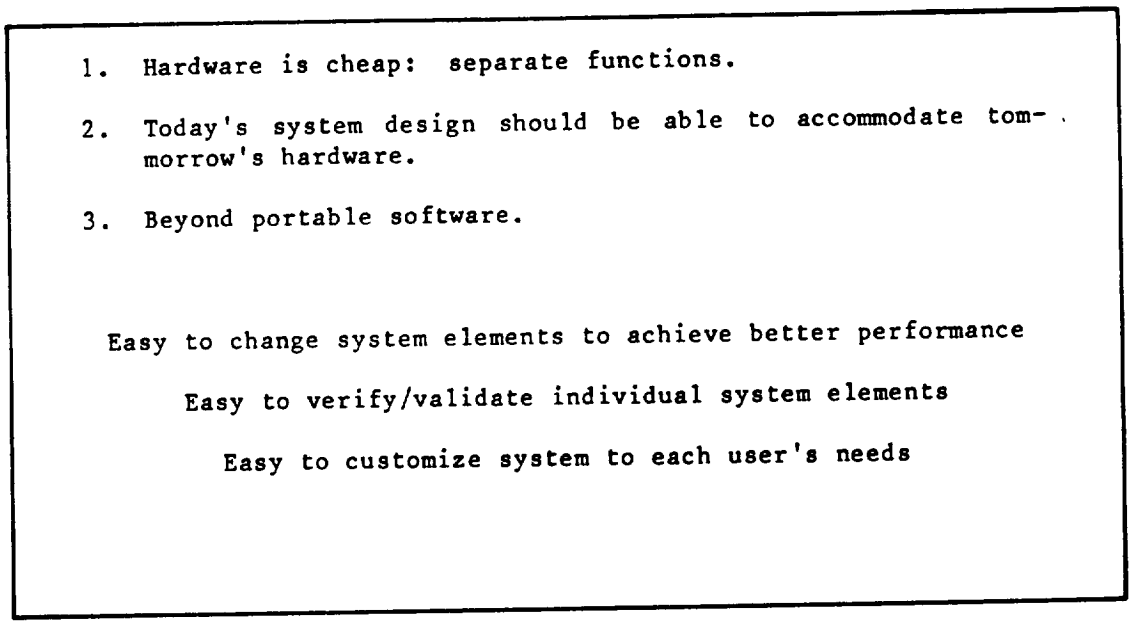

FIGURE 3. PYTHON DESIGN PHILOSOPHY

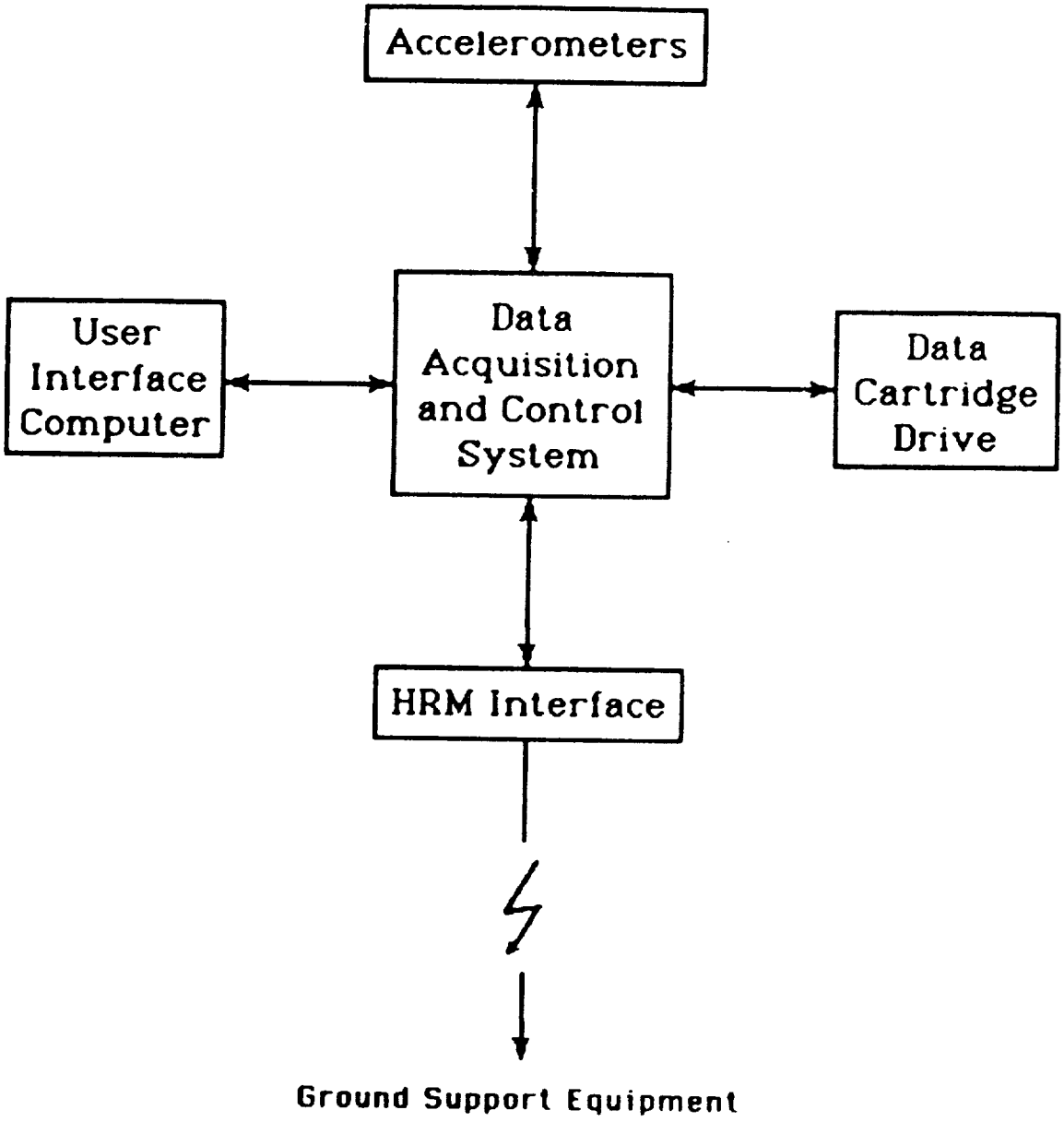

FIGURE 4. DATA ACQUISITION UNIT SYSTEM BLOCK DIAGRAM

INDEPENDENT OF PYTHON BUT CURRENT SPECIFICATIONS BASED ON SUNDSTRAND QA-2000

LOW POWER

LIGHTWEIGHT

HIGH PERFORMANCE TO COST RATIO

HIGH PERFORMANCE SERVO DESIGN

- RESOLUTION AND ACCURACY BETTER THAN 1 MICRO-G
- LONG TERM STABILITY OF DC BIAS, SCALE FACTORS, AND TEMPERATURE DEPENDENCE
- THERMAL CHARACTERISTICS WELL MODELLED AND DOCUMENTED
- SPACE FLIGHT CERTIFIED

FIGURE 5. ACCELERATION SENSORS

ents. So we can put those models into our system on the bus and do a real-time version of that data. Also, the QA-2000 sensors are already space-qualified.

The data acquisition and control systems and STD bus system were chosen from the large amount of commercially available equipment that works. There are CPU boards, memory boards, A/D boards, thermocouple boards, D/A boards, and open architecture boards available. Also, the small size means a low form factor so it's much smaller than some of the other busses that you might be familiar with. It is flexible enough to accommodate new processor chips and a new 19- or 20-bit A/D that's going to be available. There are also a lot of chips available in CMOS for low-power consumption and radiation protection. As I said before, our user interface computer is based on a GRiD case that was chosen because it is IBM compatible, and previous GRiD hardware has flown on the Shuttle. The GRiD case is now being certified for flight on the IML mission. We feel good about that but, as another example, if Hewlett-Packard or any other manufacturer gave us a certified lap top, we would not be against using their machine either.

For mass storage we are looking at a 3M data cartridge drive system, a version of which has already flown and has the ability to store 67 megabytes of data. It is a small cassette type of cartridge that you could put in your shirt pocket and bring home.

Figure 6 gives you an idea of how much recording time there is on one cassette. This is based on a 12-bit A/D word, three-channels of sample acceleration data, and 67-Mbyte capacity on the tape. We can change the sampling rates depending on what frequency ranges one is interested in. For example, a kilohertz sample rate in each channel gives 3¼ hours of continuous recording, before it is necessary to change the tape. At 200 Hz, you can get 16 hours of recording time, and if you only use 20 Hz bandwidth to get the 17-Hz signal of interest, you can go 163 hours (or nearly 7 days). If you want to run at the higher sampling rates, just bring several cassettes on board and have a crew member change them once every eight hours or so.

BASED ON A 12 BIT WORD, 3 CHANNELS OF SAMPLED DATA, 67 MBYTE CAPACITY

SAMPLING RATE	CONTINUOUS RECORDING
1 kHZ	3.25 hours
200 Hz	16.3 hours
20 Hz	6.8 days
1 Hz	136 days

FIGURE 6. CONTINUOUS RECORDING TIMES

Our Python system can simultaneously sample all the channels of data. Right now we are assuming three channels of acceleration (X, Y, Z) but more can be added. Each A/D board has eight differential inputs so one board can handle many more channels than the current baseline configuration.

We are using an interesting overlap scheme to get both dynamic range and resolution. By using two analog-to-digital converters per channel, we can get not only the dynamic range we want (which is a micro-g up to 1 g) but also the resolution in the vernier range, which is a micro g. It depends on how you scale the analog signal for the wide-scale range. If we have 1 micro-g resolution with a 12-bit A/D converter, we cover the range from 0 to $\pm$ 2 milli-g's. If we have a ½ milli-g/bit resolution on the high-range scale, we have an overlap. The wide range scale goes from ½ milli-g up to $\pm$1 g, at ½ milli-g resolution. You could change this to ¼ milli-g resolution to cover the range ¼ milli-g to $\pm$½ g.

We can simultaneously do digital filtering from 0 to 400 Hz because that looks like the frequency range of interest but that is also flexible. We can do on-line temperature compensation with stored models of the hysteresis curves. We can do dc offset corrections by letting the instrument float freely in the module for several seconds and as we have heard, the free-floating time before wall contact should last several seconds. In a confined space in the module or middeck of the orbiter, we can get less than 1 micro-g, and then use that as our dc offset with stored models of the hysteresis curves. We can also take measurements before and after the experiment to get a good idea of what happens to the sensor drift during a several day experiment.

Python can do Fourier transforms from the instrument coordinates to any experiment coordinates. People talk about power spectral densities and using fast Fourier transform algorithms. We can actually set up several different ranges so we can cover the low frequency range by taking long sampling periods of data and averaging points together. We can simultaneously do higher frequency FFT transforms as well.

We feel it is important, as Bob Naumann thought yesterday, to detect acceleration events. These are events that have an acceleration value above a user-set threshold. The goal is to measure the onset time of some excursion above the threshold, the peak acceleration, the time that the acceleration fell below the threshold, and the integrated value (the area under the curve). This gives a measure of the energy in the signal. This processing can be done simultaneously with FFTs and long-term running averages of acceleration.

We can also transfer data to the ground through the shuttle or the spacelab. We can display data to the crew members. Dr. Hamacher said yesterday that crew training can help in reducing the noise acceleration levels on board. Well, we feel this might be true, but if the crew has some feedback of acceleration, which they haven't had so far in real time, they can watch the values while they do a maneuver, or slam a door shut and see the effects of it right there. You can have a voice log so that it's all very nicely time-coordinated in one package. This can be with the full range of capabilities to command and control a variety of experiments in the indoor spaces.

The displays, commands, and controls can be used to alert crew members to off-nominal situations, give them access to the equipment, and let them take a look at the recorded data so that they can make real-time decisions about whether or not to be concerned. I certainly endorse the philosophy that if there are more scientists up there, they can truly assess the quality of data, not do data analysis on board, but take a look at it and see whether or not they want to change something.

Figure 7 shows a blow-up of the GRiD display. We decided to use a Lotus 1-2-3 type of menu across the top of the screen. We have a variety of commands to set A/D rates, stop A/D conversion, or exit back to a main menu. For example, if we were setting up an A/D rate we would set it in hertz, and the input shown in Figure 7 is 1,000 Hz. The tape and the time are self-explanatory, the data are time-tagged on the headers. We also have some digital data on the display that show the X,Y,Z axes of time-averaged RMS values. You can pick a time, say 10

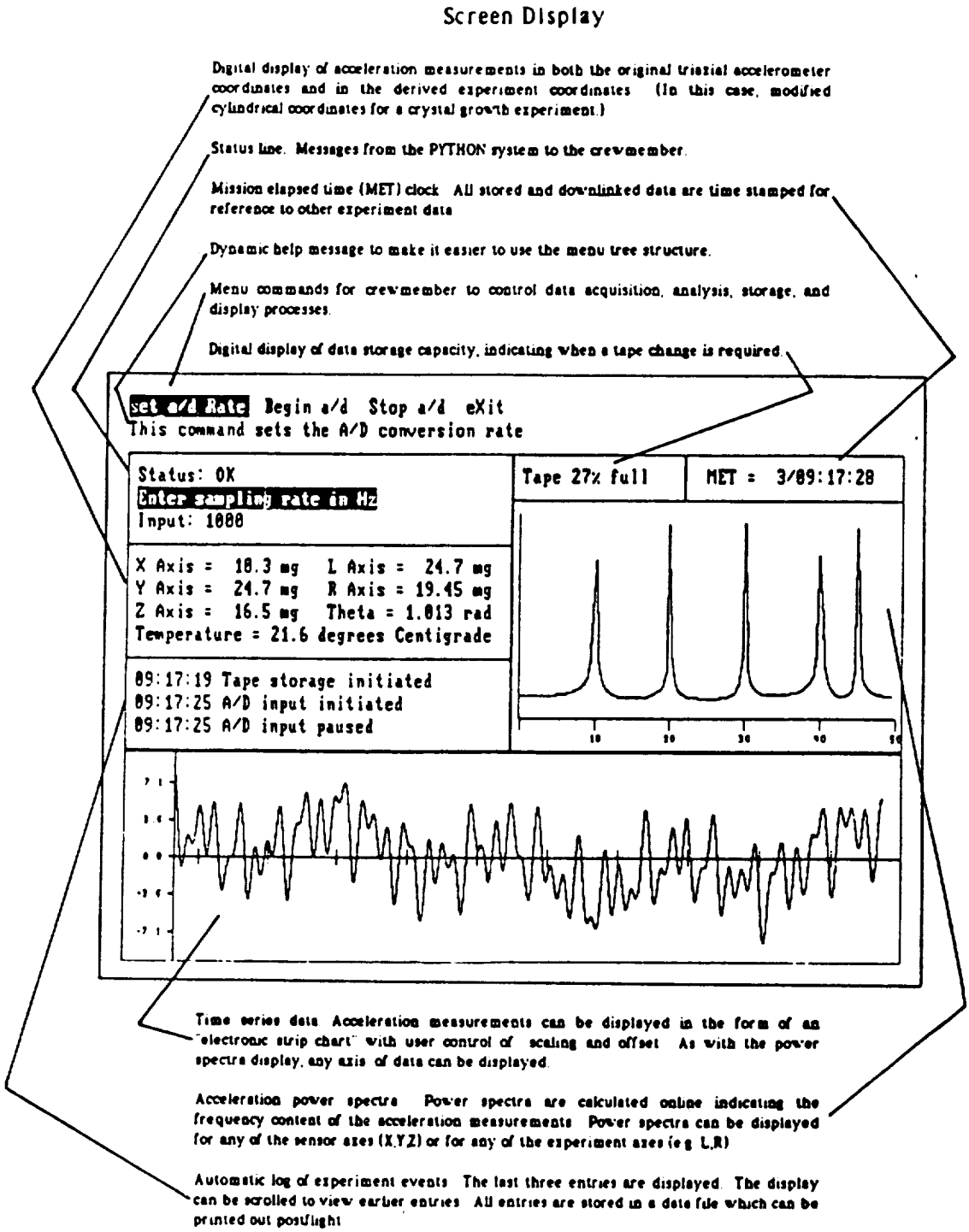

FIGURE 7. SCREEN DISPLAY

sec, and do an RMS so that once every 10 sec these quasi-steady dc values are updated. The temperature of the accelerometer package is used for temperature compensation. Those are examples of changes you can make. We have a log that gives you the time tag and what the commands were. It allows you to record the acceleration events as well and correlate them with the log.

Python can display other axes of data also, because it's not always true that your accelerometer axes (X,Y,Z) are appropriate to the experiment. Figure 7 shows cylindrical coordinates that could be appropriate if you're growing a crystal with a Bridgman technique, where you want longitudinal and radial acceleration.

Figure 7 also shows a quasi-random generated noise, some sine signal, and time axis data, but basically raw accelerometer data, with appropriate scaling. We also have the ability to have coprocessor chips on the bus system to do Fourier transforms. Figure 7 shows power spectral densities as well.

We are trying to put together a system that is flexible, customizable, useful, and gives you data the way you need it, when you need it, which is at the time the experiment is being conducted. We think our Python system has met the goals that we set for it, as shown in Figure 3. Again, just to recap, it was developed for manual use on the shuttle and space station. We have a space-qualified version that will be available. It is user-friendly with menu systems and custom software. Basing it on modular structure, we can swap components. We have disk drives so you can bring up your software at the last minute if you want one of the crew members to carry the disk in a shirt pocket. Python features a detachable sensor head, with the accelerometer of your choice or, if you trust us, our choice. And it is a stand-alone system so that we can reduce the verification and software integration requirements with the orbiter or space station. We have high-capacity data storage as well as data downlink transmission capability to all of you. The downlink transmission can be either the raw data, or the processed data. You can have a very low bit rate stream going down, which will give you just power spectral densities and maybe a 10-sec quasi-dc acceleration level.

Question: How do you get your data to downlink? I didn't know there were any wires from the shuttle middeck lockers.

Lichtenberg: The question was, how do you get your data down from the middeck? There are several ways. What we would prefer to do is use the payload data interleaver system, which will take data streams up to something like 44 kbits/sec, and downlink that. You can go through flex MDMs if you want also.

Question: Do you have hard wiring in the middeck lockers to do that?

Lichtenberg: Yes, there are some signal panels up in the payload specialist's compartment in the aft flight deck area, and I believe there have been wires for the continuous flow electrophoresis system that have been put in there that we could use. We end up going through the payload specialist station up in the middeck.

Ulf Merbold, ESA/ESTEC: One of the, I think, most important advantages of spacelab is the capability to do all the experiments interactively with the ground. How would you then hook that system into the data acquisition system on the Shuttle?

Lichtenberg: Okay, for the Spacelab, you're absolutely right. The question was, that the major advantage of doing crew-attended manned operations in Spacelab or the Shuttle is the ability to do things interactively with the ground so that you certainly want to be able to have these data down on the ground. Going in Spacelab it's very easy to get a dedicated port to the high-rate multiplexer with this, and, depending on what data streams you want to see, you can either set it up as a fairly low rate signal, in the range of tens of kilobits per second, or if you want to get all the data down you can go a little bit higher and have the raw data come down through a dedicated HRM port. So there's several different ways of doing it. It's our intention to allow that capability to get data down either from the module or from the middeck-type experiments.

Question: How can you synchronize your system with the shuttle or other accelerometers?

Lichtenberg: We have a question about how we can synchronize our system with the shuttle system or other accelerometers. We can do the time set function, and our feeling is we can get probably to within a tenth of a second. If you need it more than that, there are ways of pulling off the UTC, universal time, coming from the Orbiter, so that you can get a time code input. Our feeling was that somewhere less than a second of having a crew member input the GMT, or MET, into the computer when we start out will be good enough, at least it's going to get you in the ball park within one-half to one second.

Bob Naumann, NASA/MSFC: What does the system sell for now?

Lichtenberg: What does the system sell for? That's a good question. We have several systems. We have a ground-based version, and that would probably not consist of the GRiD because you can use an IBM, Compaq, or an AT or whatever, AT&T 6300, any IBM-compatible can be used, so we would put the ground-based system together and it's hard to come up with a price because we're not sure what you want.

Naumann: How about for the flight-qualified version?

Lichtenberg: Again, a little bit depends on what you want, but I can tell you that it certainly is going to be much less than the 1½, 2 million dollars that CDC is going to charge for their new little CMOS space flight computer, and we have data storage and also data displays. Much, much less than that.

Naumann: If your sample rate is high enough to reproduce the actual acceleration of the sample accelerometers, then you could conceivably average that and display a time-average up there if the user selects the right constants, is that correct?

Lichtenberg: The question was, it seems that our sampling rates are high enough that we can get an accurate reconstruction of the raw data and then do processing on it to do time averages and things, and that's exactly correct, Bob. We can set the A-to-D sampling rates. We figure that, probably something like a 200-Hz sampling rate is reasonable, that'll give you a reconstruction of at least 100 Hz or

below, so you're below the Nyquist frequency, but again it is settable, so if you have a very high vibration signal that you want to analyze, you can set the A-to-D rates at say 1,000 Hz and use a whole range of accelerometers. The Sundstrands are good to something like 600 Hz.

Question: Have you actually done that and shown that one can take all this grass, this noise, and actually time-average it and get zero?

Lichtenberg: Okay. Good question. No, we haven't done that yet. We're still in the process of developing the software.

Naumann: This is our big problem. The accelerometers are certainly sensitive enough, but when you try to time-average to get an average dc level, which is what most experimenters want, you get nonsense out. So it's kind of like trying to track a microvolt signal when you have millivolt noise.

Lichtenberg: The question is about the ability to take the data and to analyze as we're talking about, which is to have a fairly rapidly timed signal and to obtain a good average of it. RMS values seem like a pretty good way of doing it.

Ravindra Lal, Alabama A&M: Can the instrument take more than one sensor by a switch? Can the system take more than one sensor?

Lichtenberg: Oh, yes, can the system take more than one sensor? Yes. The STD bus cage that we're using has the ability to handle something like 8 to 10 cards. An individual A-to-D card, analog to digital card, can handle eight different signals. So you can stack up something like 80 different sensors without much of a problem at all. And what we would plan to do is to have distributed processing out there so we'd have some CPUs out in the bus that can do these dedicated tasks and then just bring the data in to display it on the user interface computer.