Journal Home Page Apollo 11 Journal

Apollo 11 Program Alarms

I thought you might be interested in some more detail about the 1201 and 1202 program alarms that occurred durning the Apollo 11 lunar landing. I and my good friend Don Eyles were two of the 'young experts' at the MIT Instrumentation Lab - Draper Lab - who worked on the software for the LEM guidance computer. Graduated from MIT the same year you graduated from CalTech (1966), and Don is a year or two older. I joined the Lab in January 1967, and Don had been there for about six months. Because the more experienced people at the lab were concentrating on getting the Command Module Computer software right, the two kids were given the responsibility for programming the LM powered-flight routines.

Don was responsible for the LM P60's (Lunar Descent), while I was responsible for the LM P40's (which were) all other LM powered flight except for P12, which was the Lunar Ascent program; we didn't concentrate on getting up from the Moon until a little later. You may have come across the tag "BURNBABY" in connection with the LM powered flight software. That was us. We might not have been out on the streets, but we did listen to the news, and the two biggest news stories were Viet Nam and Black Power, the latter including H. Rap Brown and his exhortations to 'Burn Baby, Burn' -- this was 1967, after all.

You have to constantly keep in mind the amazing - to anyone using a PC today - constraints we had to work with in programming the LGC. There were 36,864 15-bit words of what we called "Fixed" memory, which today would be called ROM, and 2048 words of "Erasable" memory or RAM. With only rare exceptions, all of the executable code was in the Fixed memory, along with constants and other similar data. Erasable memory was used for variable data, counters, and the like. With so little Erasable memory available, we were forced to use the same memory address for different purposes at different times. Thus, a location whose contents might be altitude-over-the-lunar-surface during the landing stage might have contained the results of a sextant sighting of a navigational star from the alignment program. I think there were some memory locations that were shared seven ways. You can imagine the testing we had to do to ensure that the same memory location was not being used by more than one program at the same time.

The only time that programs were executed out of erasable was when we had to "patch" the program after it had been released and the fixed memory configuration had been manufactured. The most

famous incident was on Apollo 14, when Don figured out how to patch the program to ignore the faulty Abort switch. This patch was radioed up and the crew entered it manually.

You also have to remember that, long before Bill Gates, we had developed a real-time multi-tasking operating system. There were interrupt-driven, time-dependent tasks - e.g., turn the LM Descent Engine on at the correct time - as well as priority-ordered jobs that dealt with less time-critical things. Each scheduled job has some erasable memory to use while it was executing. This memory was used for intermediate computational results, rather data. For example, we might have used an Erasable memory location with the mnemonic name TGO to contain the calculated time of engine burn for a maneuver. This was not stored in the memory allocated to individual jobs, so that it could be shared between programs. Each job was allocated a "core set" of 12 erasable memory locations. If a job required more temporary storage, the scheduling request asked for a VAC - vector accumulator - which had 44 erasable words. There were seven core sets and five VAC areas.

When a job was to be scheduled, a call would be made to the appropriate executive routine - sort of like a DOS call today. If the job to be scheduled required a VAC area, the operating system would scan the five VAC areas to find one which was available. After finding and reserving a VAC area, the core sets would be scanned to find an available core set. Scanning for a VAC area would be skipped if the scheduling request specified "NOVAC". In any case, if there were no VAC areas available, the program would branch to the Alarm/Abort routine and set Alarm 1201. Similarly, if no core sets were available, the program would branch to Alarm/Abort and set Alarm 1202.

So what was happening during Apollo 11, as I recall, was that repeated jobs to process rendezvous radar data (that of course were not really there) were scheduled because a misconfiguration of the radar switches. Thus, the core sets got filled up and a 1202 alarm was generated. The 1201 that came later in the landing was because the scheduling request that caused the actual overflow was one that had requested a VAC area.

What happened next in either case was what you described as, 'The computer has been programmed to recognize this data as being of secondary importance and will ignore it while it does more important computations.' It was a little more than that, and had been the subject of a great deal of testing before the software had been released. The software rebooted and reinitialized the computer, and then restarted selected programs at a point in their execution flow near where they had been when the restart occurred. To give you an example in today's terms, right now I have Windows95 with Netscape Communicator active as I compose this message. In the background, an audio CD is playing. WordPerfect is open, but has no active document, and Quicken has my checking account open. If I were to reboot now, all of those programs would be closed down. If Windows was a smart as the Apollo LGC executive, after the reboot the CD might not be playing and WordPerfect might not be there, but I would come back to this message composition window, with the same text displayed, and Quicken would have my checking account open to the proper place.

On Apollo 11, each time a 1201 or 1202 alarm appeared, the computer rebooted, restarted the important stuff, like steering the descent engine and running the DSKY to let the crew know what was going on, but did not restart all the erroneously-scheduled rendezvous radar jobs. The NASA guys in the MOCR knew - because MIT had extensively tested the restart capability - that the mission could go forward.

Journal Home Page Apollo 11 Journal