INTEGRATION OF GROUND AND ON-BOARD SYSTEM FOR TERMINAL COUNT

Charles A. Abner
Guidance, Digital & Software Systems Division
Kennedy Space Center

Don H. Townsend
Avionics Systems Division
Johnson Space Center

ABSTRACT

This paper discusses the challenge faced in the development of an integrated ground and on-board system for Space Shuttle terminal count management. The criteria considered in designing this system are outlined with some attention given to examples of problems encountered in the process of maturing the design.

INTRODUCTION

The integrated launch system developed in the Space Shuttle program requires a closely coordinated effort between the ground system and the on-board system. The system had to be structured so that it would be flexible enough for more rapid reconfiguration than in past programs. This was true not only for ground systems but also for the vehicle as well. This is a brief overview of the development of the Space Shuttle terminal count integrated monitor and control system.

SOFTWARE MANAGEMENT IS THE KEY

The Apollo launch support systems was composed of two computers, one located in the mobile launcher and directly locked to the vehicle, and one located in the Launch Control Center. These two computers were connected via a data link which provided fixed telemetry streams to the Launch Control Center for vehicle systems evaluation. The majority of the monitoring function was done by Firing Room personnel looking at meters, lights, and plotters driven by the Launch Control Center computer. Thus, the Apollo launch approach was essentially a fixed system which was structured to provide a single sequential flow for vehicle checkout and launch.

In contrast with the more restrictive approach for terminal count management as was used in Apollo, the challenge arose in the Shuttle era to allow a more flexible design which would be adaptable to the varied configurations of the launch vehicle. In order to achieve this flexibility a software architecture was designed for both on-board and ground systems to readily adapt to any requirement changes. This approach not only allowed for the initial development of the integrated launch system but also supports the basic concept of a multi-mission Space Shuttle Program.

SOFTWARE DESIGN STRUCTURE

The on-board systems management software structure was designed based on individual system inputs as to command/monitoring requirements. The terminal count and launch requirements were considered by the systems during the definition and design of this software structure.

The ground systems software structure was designed to meet total system checkout requirements. These include the requirements for terminal count capabilities.

CONTROLLING DOCUMENTS

During the process of defining these two software systems a interface definition document (CPDS-150) was developed. The primary purpose of this document was to baseline and establish configuration control of the Launch Data Bus (LDB) interface between the ground and on-board systems. This interface definition encompasses both day-to-day test and checkout requirements as well as specific terminal count requirements.

Various documents exist which control the requirements for terminal count activities. The on-board software requirements (for terminal count) were specified in the Functional Subsystem Software Requirements, Sequencing Requirements, STS 81-0026 (FSSR-26). The Launch Commit Criteria (LCC), JSC 16007 and KSC S00000-3, documents were established and are maintained by JSC and KSC. The Ground Launch Sequencer Description Document (GLSDD-OMI-S9005) establishes the parameter monitoring and sequencing requirements for ground systems and vehicle systems activities in support of terminal count and launch. The above documents not only relate software design requirements but also contain real-time anomaly guidelines such as hold/scrub situations.

SEQUENCE CONTROL

VEHICLE SOFTWARE

The flight GN&C software load supports all of the vehicle terminal count requirements. These requirements include both the on-board sequencing as well as the system software necessary to support ground initiated tasks. The on-board sequencing software has interfaces through the GN&C systems software with both the Backup Flight System (BFS) computer and the Space Shuttle Main Engine (SSME) computers. Through the use of these three software sets all Shuttle systems are managed/monitored for terminal count.

GROUND SOFTWARE

The Launch Processing System (LPS) application software supports all the terminal count requirements for ground systems and vehicle interfaces. Most of the terminal count activities are incorporated in the Ground Launch Sequencer (GLS) application software. The remaining terminal count activites are controlled/monitored by systems engineers via their own application software. The vehicle interface for these activities is supported by the on-board GN&C system software via the LDB and the PCM system. This interface provides the LPS access necessary to control all vehicle systems.

LIMITATIONS

It is now appropriate that a few software design limitations be discussed in order that the reader fully appreciates how the challenge of interfacing the ground to on-board systems with the required flexibility was achieved. Trade-offs of total software design capabilities were considered which resulted in today's limitations.

One of the primary limitations was that encountered with LDB structure. Design considerations were vehicle weight and hardware design flexibility in choosing a serial interface over parallel interfaces for the LDB. Additionally, software complexity and memory allocation were drivers in the decision process. The resulting LDB structure allows 240 msec to complete one ground to on-board transaction. This transaction may consist of a one-to-one task such as an operate valve request, a predetermined and stored sequence of one-to-one tasks, or a special coded flight software request for use in terminal count. A 120 msec timing may be achieved by the ground requesting to inhibit the on-board response available in the protocol. The serial data operation was designed with a redundant capability. This redundancy/design includes the necessary data bus hardware as well as the ground and on-board software.

Another limitation which affected the terminal count protocol was the design of ground software. Each firing room console (terminal count activities are controlled at the integration console in the firing room) has the capability of running six application tasks in addition to several other system software tasks. There are time critical functions in terminal count opera-

tions which require stringent control of these six application tasks in order to avert a console throughput (timing) problem. Increased console throughput activities result in delayed and inconsistent LDB operations.

Limitations for vehicle systems management arose as a result of constraining the number of systems parameters to be processed and monitored on-board. This limitation is taken care of by the ground system monitoring and processing of additional parameters.

The combination of the above limitations and several other factors led to the biggest challenge which was managing terminal count time critical events and time critical operations.

DESIGN CRITERIA

The design criteria of coordinating vehicle/ground clocks, performing retries of parameter statusing and command executions, real time data manipulations, and managing potential recycle/scrub activities greatly affected the integrated terminal count software structure.

TIME MANAGEMENT

Of prime importance in the integration of terminal count activities is the management and control of the ground and vehicle clocks. Greenwich Mean Time (GMT) is the reference used by both the ground and on-board software systems. By use of this reference the countdown clock is initialized and controlled by ground system commands. Countdown time is used to initiate all of the terminal count events required by the vehicle and ground. The primary challenge in this areas was the detailed analysis and coordination required to place each function at a specified time with relation to its associated terminal count events. An example of the need for integrated timing requirements was the disposition of the delta that existed between the ground and on-board time due to the on-board software delaying for main engine/vehicle structure stabilization prior to the Solid Rocket Booster (SRB) ignition. This integrated effort was required to achieve a common T-0 lift-off reference.

RETRIES

The initial design requirement was to provide retry logic for anomalous conditions prior to initiating corrective action or proceeding to the next countdown task. Due to previously mentioned LDB timing constraints in conjunction with the requirement for nominal terminal count tasks, the retry logic was found inappropriate and unachievable in most cases. Software verification and validation testing gave the confidence required to assure that retry logic was not mandatory for reasons of safety or for the high probability of an unsuccessful software/hardware transaction. The data transmission "glitch" problems experienced during the Apollo era which originally drove the retry design requirement were found to be non-existent with the Shuttle hardware. The vehicle application software was structured to provide retry capability for certain tasks based upon a systems analysis of potential problems which could be encountered. Even though the ground application software utilizes minimum retry logic, provisions do exist in the ground systems software to retry unsuccessful INPUT/OUTPUT transactions three times.

REAL-TIME OPERATIONS

The capability of manipulating all terminal count events to react to real time event and parameter deviations was another requirement consideration in software design. In keeping with this requirement, the software was designed to allow the bypassing of any terminal count task and to change the limits of any terminal count analog measurement. Provisions were made in the vehicle software for bypassing of certain vehicle monitor/command tasks in response to ground system inputs.

RECYCLE/SCRUB

Another criteria which influenced the integrated software design was capability to perform a recycle or scrub operation at any time during the terminal count. Vehicle safing requirements and ground/vehicle clock synchronization requirements were the primary drivers in implementing the recycle/scrub requirement. Consideration for the implementation must allow for a continually changing vehicle configuration and the need for both ground and vehicle software applications to track the current status. In the design of the recycle/scrub software logic an analysis of the independent on-board software to vehicle systems interface and ground systems management procedures was performed to assure that the vehicle could be placed in a safe configuration. For example the vehicle application software was programmed to go through its predetermined set of recycle/scrub safing commands and then terminate all activity to assure no interference with the ground systems tasks. The ground systems primarily manage the reconfiguration of the vehicle based on the countdown time at which a recycle/scrub was requested.

VERIFICATION AND VALIDATION

As previously mentioned, the primary integration effort required of the ground/vehicle for terminal count operations was the detailed timing analyses to verify time critical operations. In addition to the individual development center's analysis and verification processes, a highly integrated test activity was implemented at the Lyndon B. Johnson Space Center (JSC) Shuttle Avionics Integration Laboratory (SAIL) and at the John F. Kennedy Space Center (KSC) during actual integrated vehicle functional testing. The results of this testing were fed back into the ground and vehicle software design requirements. Software changes were made after analysis and trade-off studies were performed to determine whether adjustments could best be made on the ground or on the vehicle. Software change lead time constraints played an important part in this decision process. Numerous changes were implemented due to the timing analysis and testing results. Adjustments to Launch Commit Criteria also caused numerous changes. Any change always required additional analysis and a test program to insure that no unforeseen problems were created or compounded.

EXAMPLE PROBLEMS ENCOUNTERED

It would be appropriate at this point to discuss several situations which were confronted during the process of integrating the terminal count.

SRB LOCKOUT MANAGEMENT

At a specific point in the countdown sequence, T-40 seconds, the SRB Multiplexer-Demultipler's (MDM's) LL1/LR1 moules 0 and 4 are commanded to the lockout state in preparation for lift-off. At T-13 this same lockout function is performed for LL2/LR2. Initially, in order to achieve this, the ground system was required to issue 20 LDB one-to-one transactions. These commands required an unacceptable amount of time to accomplish at this point in terminal count. This situation was analyzed by JSC and KSC and the best practical solution was found to be a change to flight software. It provided an explicitly coded software function which would issue the necessary commands to accomplish the lockout of LL1/LR1 or LL2/LR2 based upon a single LDB transaction. This transaction provided a positive response to the ground system to assure each module was locked. The following is the list of commands for LL1/LR1 that was initially performed from the ground via one-to-one transactions which required approximately 2.5 seconds for each set of MDM's.

- READ LL1 MDM BITE
- ISSUE LOCK LH SRB MDM LL1 MOD 0
- READ LL1 MDM BITE
  IF BITE / 1000_{16} EXIT SEQUENCE AND SET RESPONSE TO VERIFY FAIL
- ISSUE LOCK LH SRB MDM LL1 MOD 4
- READ LL1 MDM BITE
  IF BITE / 1000_{16} EXIT SEQUENCE AND SET RESPONSE TO VERIFY FAIL

- READ LR1 MDM BITE
- ISSUE LOCK RH SRB MDM LR1 MOD 0
- READ LR1 MDM BITE
  IF BITE / 1000_{16} EXIT SEQUENCE AND SET RESPONSE TO VERIFY FAIL
- ISSUE LOCK RH SRB MDM LR1 MOD 4
- READ LR1 MDM BITE
  IF BITE / 1000_{16} EXIT SEQUENCE AND SET RESPONSE TO VERIFY FAIL
  EXIT SEQUENCE AND SET RESPONSE TO VERIFY

The set of commands to perform the LL2/LR2 lockout would be identical to the above with the LL1's replaced by LL2's and LR1's replaced by LR2's. The flight software change reduced the ground system commands to one LDB transaction for each set of MDM lockouts.

It was also determined that in the case of a recycle/scrub in which the MDM's had been locked, the ground systems was unable to reliably time the unlock sequence using one-for-one commands (700 $\pm$ 100 msec). The difficulty was due to the unpredictability of LDB traffic and ground software activity. It was determined that an existing "pulse mode" option, available in a test and checkout configuration, would best provide the required capability in the prelaunch flight configuration. Flight software was changed to provide this capability.

With this integrated effort the SRB MDM's may be locked and unlocked in a highly efficient and reliable mode.

SRB HYDRAULIC POWER UNIT (HPU) START & GIMBAL PROFILE MONITOR

One of the most delicate tasks in the integrated terminal count sequence is the startup and monitor of the SRB HPU's which occurs inside of T-30 seconds. Many hours of design, testing, and data analysis were used to insure that the HPU's could be started in a timely manner without an overspeed condition and that the SRB gimbal profile which immediately follows the startup sequence would execute properly.

During the HPU startup sequence one GLS design requirement had to be waived in order to allow the numerous HPU prestart commands to be issued in the time available. The requirement was that any GLS command could be individually bypassed or have its command state altered during a real time firing room environment. For the HPU prestart sequence it was decided to use the software capability of issuing multiple commands via a single LDB transaction.

The potential inadvertant runaway overspeed at startup is monitored by the use of an LPS application software technique known as Control Logic. Control Logic design allows the monitoring of PCM parameters and an associated predetermined response independent of the actions required of the normal application software. The turbine speed monitored for startup is higher than the normal range of turbine speed. The normal GLS application software monitors the normal range of the HPU turbine speed after startup has occurred.

In the process of vehicle testing flow for the first Space Shuttle launch it was found that the enabling of this normal monitoring function was about 100 msec early and caused a shutdown of the HPU due to an overspeed condition. This anomoly, which was caused by a controlled higher initial speed in the start as requested by the HPU controller was not considered in GLS timing and resulted in another detailed analyses of the ground to vehicle timing for terminal count. The GLS enabling of overspeed monitoring was subsequently adjusted to correct this situation.

At T-21 seconds the SRB Gimbal profile is initiated by the ground software. This 4 second test of establishing a positive, negative, and null position of the SRB Tilt and Rock actuators required many hours of testing and analysis. Actual vehicle testing along with the SAIL facility produced the necessary data to provide the confidence of a good system checkout and a "go for launch" status.

RECYCLE AFTER SSME START ENABLE COMMAND

An example of an integrated task which included the SSME controller is the potential main engine shutdown initiated after the "start enable" command has been issued. Once the SSME con-

troller has received start enable, it must receive the "start" command within 5 seconds or "start enable" must be commanded again. The on-board application software is structured such that once "start enable" is commanded the only command that may be sent is "start". In the event that a recycle occurs after "start enable" has been commanded, the time out between the two commands in the SSME controller will occur but the on-board count cannot be resumed without a reload of the computer.

The decision was made to change the on-board application software to reinitialize upon recycle and thus allow a resumption of the count without reloading the on-board computer program.

STACKED RESUME

Whenever the on-board application software is requested by the ground or detects a limit violation which causes it to go into a "hold", the ground has to issue a "resume" count request. In the initial design, the on-board system software would accept and save a "resume" count request sent by the ground even though a "hold" was not in effect. Then, when the next hold condition occurred the stored "resume" would be executed immediately, which essentially would negate the hold. This condition was discovered in testing and changes to both on-board and ground software now precludes the possibility of the on-board inadvertantly resuming the count.

SUMMARY

The challenges of the Integrated Ground and On-board Terminal Count development may be summarized in three categories: task integration, software management, timing analysis. The many functions that must be considered and implemented into a terminal count sequence require a super integrated effort among many contractors and many disciplines. The incorporation of these functions into the software require a well managed software development organization. The verification/validation that all functions will be performed in an efficient and timely manner requires a dedicated test team approach.

The successful launch of STS-1 and subsequent flights is proof that the challenge of these three important functions have been met and are continuing to be met for each launch.