Flight Code Validation Simulator

SAN095-1616C
CONF-9510136-1

Brent A. Sims*

Sandia National Laboratories, Albuquerque, New Mexico 87185

An End-To-End Simulation capability for software development and validation of missile flight software on the actual embedded computer has been developed utilizing a 486 PC, i860 DSP coprocessor, embedded flight computer and custom dual port memory interface hardware. This system allows real-time interrupt driven embedded flight software development and checkout. The flight software runs in a Sandia Digital Airborne Computer (SANDAC) and reads and writes actual hardware sensor locations in which IMU (Inertial Measurement Unit) data resides. The simulator provides six degree of freedom real-time dynamic simulation, accurate real-time discrete sensor data and acts on commands and discretizes from the flight computer. This system was utilized in the development and validation of the successful premier flight of the Digital Miniature Attitude Reference System (DMARS) in January of 1995 at the White Sands Missile Range on a two stage attitude controlled sounding rocket.

I. Introduction

This paper describes a system to develop and validate flight software and hardware interactions for rocket systems. This system may be used for any computer software/hardware interaction which requires simulation and off-line validation prior to the desired use or event.

The application for this system is to provide an environment in which flight software, including inertial navigation, attitude control (spinning and 3-axis stabilized), guidance and flight sequencing, is exercised in real-time and as close to actual flight conditions as possible. Typically, for a given flight no less than forty perturbations to the simulated trajectory and systems performance are run against the flight software/hardware to prove its validity and correct response to predicted perturbations.

The system, designed to fulfill these requirements, incorporates an i860 DSP board on the AT bus of a 486 PC. Custom hardware was designed to interface between the i860 and the flight computer and its flight Input/Output (I/O) boards. Advantages of this system are the relatively low cost and transportability, since most rocket launches and hardware checkouts occur at remote site locations.

The remainder of this paper describes the hardware and software in the flight system and the simulation system. It also explains the results of the test and, consequently, the success of the simulation and validation computer hardware and software.

II. Rocket System Hardware

The rocket system utilized on this flight was a two stage spin stabilized sounding rocket with attitude control and autonomous linearized guidance for dispersion reduction and flight sequence control. The first stage consisted of a Terrier Mk 12 rocket

motor. The second stage was an Improved Orion rocket motor. This system was rail launched at the White Sands Missile Range from Launch Complex 36 off of the NASA Athena Launcher (see figure 1).

Figure 1

*Senior Member of Technical Staff, Member of AIAA

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Figure 2

The basic systems required for the rocket include; Power System, IMU/Attitude Control System, Flight Sequencer, Fireset, Telemetry System and Payload. The power system utilizes batteries for flight and external power supplies via umbilical cables for ground and pre-launch testing. The IMU/Attitude Control System and flight sequencer is contained within the DMARS unit. This system provides state information of the vehicle for proper orientation, flight sequencing, and payload release conditions. The fireset supplies initiation signals to all ordnance items at the appropriate times and conditions. Some fireset functions are supplied on a fixed time basis and others are performed by request of the flight computer based on the correct states. The telemetry system combines analog measurements and digital data streams from the flight computer and other on board hardware and sends the data down on a radio frequency (RF) link to the range ground station for real-time display and post flight processing.

The IMU is contained in the DMARS, providing a roll-stabilized platform for strapdown inertial navigation (see figure 2). The platform provides 4-axes of gyro information (one redundant) and 3 axes of accelerometer information. Also, the SANDAC flight computer, a Motorola 68020, 33

MHz with floating point capability, resides in the unit with internal communication paths to the sensors, along with serial, parallel and discrete communication paths to external electronics and sensors.

Telemetry information is provided from the SANDAC computer via the external parallel port to a custom programmable Pulse Coded Modulation (PCM) encoder. This encoder serializes the contents of a First-In-First-Out (FIFO) at the desired bit rate and then supplies an interrupt at the desired point for the computer to refill the FIFO. The serialized data is both sent out a serial link and also modulated on the base band of an S-Band Transmitter for RF transmission.

III. Rocket System Software

The navigation, guidance and control (NG&C) software for this vehicle resides in the SANDAC computer incorporated in the DMARS (see figure 3). This software is broken down into three separate functional modules; inertial navigation, guidance & control, and flight sequencing. The inertial navigator reads the IMU sensors and compensates the raw data to provide correct attitude and position information. The guidance and control module utilizes the attitude and position states and acts on them to orient the vehicle and or modify the motor ignition time to remove the first stage dispersion in order to force the vehicle to pass through the desired target position. The flight sequencer effectively ties all the modules together to provide for attitude control, guidance and discrete events at the appropriate times throughout the flight of the vehicle. These decisions are based on time, attitude and position states.

The flight code is written in the "C" language, with assembly language used to incorporate interrupt service routines and operating system calls. The VRTX Real-Time Operating System is utilized for discrete real-time software control. The code is compiled on a PC using the Greenhills "C" cross compiler and assembled utilizing the Quelo Linker and Assembler tool set.

IV. Simulation System Hardware

The simulation system is hosted on a 486/66 MHz PC. The real-time simulation code is run interrupt driven on an Alacron i860 AT Bus DSP Board. A custom daughter card was designed to provide dual port memory addressing and interrupt

Figure 3

capability to and from the i860 board. It also provides a parallel port connection to the DMARS sensor emulation interface board which is directly connected to the SANDAC flight computer and the upper and lower I/O boards. The upper and lower I/O and the SANDAC computer are actual DMARS hardware. These were identical to the flight hardware and were built as if incorporated in a flight unit. A functional block diagram of this system is shown in figure 4.

V. Simulation System Software

The simulation software resides on the PC and is downloaded to the i860 at run time. The simulation code was written in FORTRAN 77 and incorporates models and algorithms which have been proven on many past programs. The interface from the FORTRAN simulation code to the Alacron i860 and the custom daughter card was written in the "C" language to provide for use of Alacron included real-time software calls, addressing of the dual port memory, and interrupt service routines.

The simulation code is broken down into a number of modules. These consist of; six degree of freedom rigid body dynamic equations of motion,

aerodynamic inputs to the equations, missile sequencing for aerodynamic tables and mass property variations, sensor dynamic models, I/O and interrupt driven interface for real-time communication with the embedded flight computer and its interface hardware.

The dynamic equations of motion represent the rigid body dynamics of the vehicle with forcing functions of table lookup aerodynamics, atmospheric models, rocket motor thrust models, attitude control system thrust models, rotating earth or inertial and rail launch models. The aerodynamic models are based on wind tunnel testing and previous flight experience. Missile sequencing effectively provides for sequencing of the correct motor thrust tables, aerodynamic tables and attitude control based on time from lift-off. The interrupt driven interface provides the communication path to the flight hardware for transfer of simulated sensor data and response to discrete control parameters. This module provides the real-time aspect to the simulation and the true programmable interface for hardware sensor mapping.

VI. Validation of Flight System Software

Software Validation System

Figure 4

Once hardware checkouts, air bearing testing of the attitude control system (ACS) and preliminary simulation runs are completed then the software is ready to be validated. The validation of flight software is done in an attempt to exercise most possible scenarios that may occur and to prove that the software/hardware acts and reacts as required. For a rocket system some of the perturbation runs include; high/low performance rocket motors, angular error in initial pointing due to rail launcher mis-positioning, sensor bias and scale factor variations, mass property variations, ACS thrust variations and combinations of these and other variations.

Typically, a pass through the series of validation runs will provide a case in which the software does not handle as expected or desired. In this case, the software is modified and re-tested to confirm proper operation. Also, some set of basic runs must be revalidated in order to confirm the change has not effected the flight software adversely with respect to other modules and effects.

VII. Results of Flight Test

The premier flight of the DMARS, which utilized this validation system for the first time, was highly successful. The flight had an anomaly during first stage and second stage separation, prior to second stage ignition, which considerably depressed the trajectory from the nominal design parameters. Due to the fault tolerant design of the flight software, and the use of the simulator for exercising the flight software in many off nominal cases, the orientation was modified in real-time and the payload release was replanned for optimum data gathering. The primary outcome of the test was nominal due to the design for anomalies and the realistic exercising of the software and hardware.

This simulator has since been used to validate flight software for two additional Sandia Strypi sounding rocket launches conducted in June of 1995.

VIII. Conclusions

In general, this system provided a method to exercise hardware and software which usually may not be exercised in a flight like condition without actually launching the vehicle and expending the considerable resources required for the one-time only launch. With this simulation system the missile can be "launched" many times with many possible off nominal conditions. The outcome can be analyzed for possible software modification requirements or for assurance that the software reacts correctly to the known conditions. Also, the simulations run with this system are the highest fidelity which may be run without actually launching the vehicle. This is due to utilization of actual hardware and software exercised in a real-time environment with representative simulation models.

The most important aspect of this system is that it could be modified to work on any application which incorporates some type of sensors and logic. The main reason for using this simulator is to exercise a system which needs to be validated for fault tolerant response and might not be able to be tested in a realistic manner prior to its usage.

Acknowledgements:

This work was supported by the United States Department of Energy under Contract DE-AC04-94AL85000.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.