

Evolution of the Space Shuttle Primary Avionics Software and Avionics for Shuttle Derived Launch Vehicles

2011-01-2575

Published
10/18/2011Roscoe C. Ferguson
United Space Alliance

Copyright © 2011 SAE International

doi:10.4271/2011-01-2575

ABSTRACT

As a result of recommendation from the Augustine Panel, the direction for Human Space Flight has been altered from the original plan referred to as Constellation. NASA's Human Exploration Framework Team (HEFT) proposes the use of a Shuttle Derived Heavy Lift Launch Vehicle (SDLV) and an Orion derived spacecraft (salvaged from Constellation) to support a new flexible direction for space exploration. The SDLV must be developed within an environment of a constrained budget and a preferred fast development schedule. Thus, it has been proposed to utilize existing assets from the Shuttle Program to speed development at a lower cost. These existing assets should not only include structures such as external tanks or solid rockets, but also the Flight Software which has traditionally been a "long pole" in new development efforts.

The avionics and software for the Space Shuttle was primarily developed in the 70's and considered state of the art for that time. As one may argue that the existing avionics and flight software may be too outdated to support the new SDLV effort, this is a fallacy if they can be evolved over time into a "modern avionics" platform. The "gold" of the flight software is the "control loop" algorithms of the vehicle. This is the Guidance, Navigation, and Control (GNC) software algorithms. This software is typically the most expensive to develop, test, and verify. The "control loop" software algorithms could be extracted and evolved to execute on technology compatible with the legacy system embedded within a SDLV avionics platform. It is also possible to package the GNC algorithms into an emulated version of the original computer (via Field Programmable Gate Arrays or FPGAs), thus becoming a "GNC on a Chip" solution.

INTRODUCTION

As a result of recommendation from the Augustine Panel, the direction for Human Space Flight has been altered from the original plan referred to as Constellation. NASA's Human Exploration Framework Team (HEFT) proposes the use of a Shuttle Derived Heavy Lift Launch Vehicle (SDLV) and an Orion derived spacecraft (salvaged from Constellation) to support a new flexible direction for space exploration. The intent is to start development of the Shuttle Derived Heavy Lift Launch Vehicle (Figure 1) in 2011 (or shortly afterwards) with possible flights as early as 2016. The SDLV must be developed within an environment of a constrained budget and a preferred fast development schedule. Thus, it has been proposed to utilize existing assets from the Shuttle Program to speed development at a lower cost. These existing assets should not only include structures such as external tanks or solid rockets, but also the Flight Software which has traditionally been a "long pole" in new development efforts. The avionics and software for the Space Shuttle was primarily developed in the 70's and considered state of the art for that time. As one may argue that the existing avionics and flight software may be too outdated to support the new SDLV effort, this is a fallacy if they can be evolved over time into a "modern avionics" platform. The technology may be outdated, but the avionics concepts and flight software algorithms are not. The reuse of existing avionics and software also allows for the reuse of development, verification, and operations facilities. This is another hidden or forgotten cost that can over strain a budget and schedule due to challenges such as parallel development efforts.

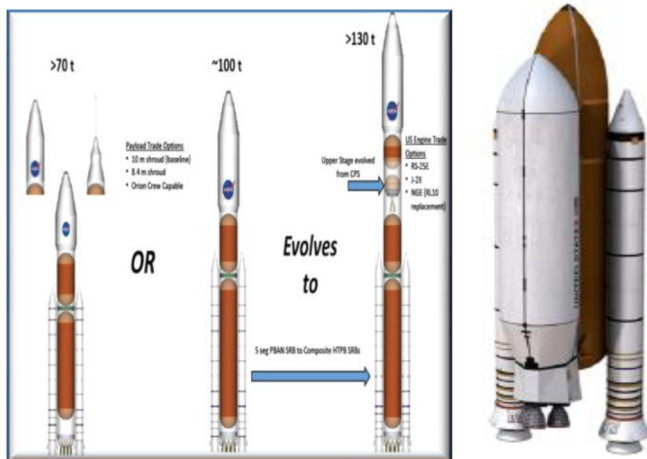


Figure 1. Shuttle Derived Launch Vehicles (Courtesy of NASA)

This paper discusses an approach to *evolve* the current Space Shuttle avionics and software assets to support the proposed SDLV. The keyword is *evolve* in that these assets can support the fast development of such a vehicle, but then be gradually evolved over time towards more modern platforms as budget and schedule permits. The “gold” of the flight software is the “control loop” algorithms of the vehicle. This is the Guidance, Navigation, and Control (GNC) software algorithms. This software is typically the most expensive to develop, test, and verify. Thus, the approach is to preserve the GNC flight software, while first evolving the supporting software (such as Command and Data Handling, Caution and Warning, Telemetry, etc.). This can be accomplished by gradually removing the “support software” from the legacy flight software leaving only the GNC algorithms. The “support software” could be re-developed for modern platforms, while leaving the GNC algorithms to execute on technology compatible with the legacy system. It is also possible to package the GNC algorithms into an emulated version of the original computer (via Field Programmable Gate Arrays or FPGAs), thus becoming a “GNC on a Chip” solution where it could live forever to be embedded in modern avionics platforms.

OVERVIEW OF SPACE SHUTTLE AVIONICS AND PRIMARY AVIONICS SOFTWARE

AVIONICS

The Space Shuttle avionics system consists of multiple computers and multiple buses (Figure 2). The computer set consists of five AP-101S General Purpose Computers (GPCs) which are from the IBM Modular Military Computer (MMC) technology line [1].

A GPC can be separated into a CPU and Input Output Processor (IOP). Each GPC has twenty-six independent processors consisting of a CPU, Master Sequence Controller (MSC), and twenty-four Bus Control Elements (BCEs). The MSC and BCEs are part of the IOP. The CPU performs general processing, while the IOP performs input and output processing [1].

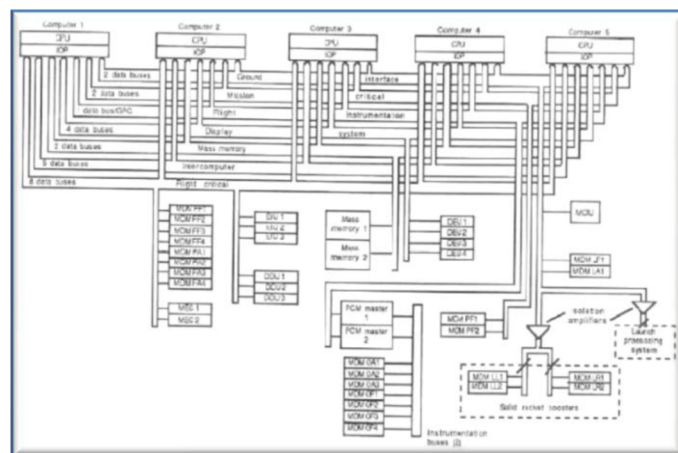


Figure 2. Space Shuttle Avionics (Courtesy of NASA)

The IOP interfaces to the twenty-four serial data buses. Twenty-three of the buses are cross strapped (available to all GPCs) while the twenty-fourth is not. The twenty-four buses are grouped into seven categories. These are Inter-computer (five buses), Mass Memory (two buses), Flight Instrumentation (five non-cross-strapped buses, one per GPC), Payload (two buses), Launch Data (two buses), Display (four buses), and Flight Critical and Sensor Control (eight buses). The eight Flight Critical and Sensor Control Buses are organized into four strings. Each string consists of two buses. The concept provides two paths to the IO devices. This renders four logical buses. However, if a bus path fails, the option is provided to switch to the alternate path via a port mode operation [1].

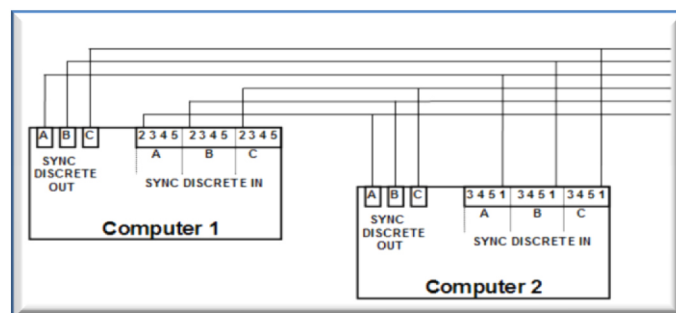


Figure 3. Synchronization Discrete Set (Courtesy of NASA)

The IOP provides a discrete interface to support input and output operations for switches and controls. The discrete set also provides a fast communication path between GPCs to support synchronization operations (Figure 3) [1].

PRIMARY AVIONICS FLIGHT SOFTWARE (PASS)

The Primary Avionics Flight Software or PASS is the center piece of the safety critical avionics of the Space Shuttle. Without it, the vehicle is inoperable. Its failure can result in catastrophic failures resulting in the loss of life or the inability to achieving mission success [1]. The software has a proven track record of low error rates and high quality [2] [3]. These attributes were obtained via an enormous investment in all aspects of the software life cycle of continuous process improvements over 30 years.

The software is composed of system software and applications. The system software interfaces with the hardware and provides services to applications. It is an aggregate of the Flight Computer Operating System (FCOS), System Control, and User Interface. Applications provide the behavior of the vehicle [1].

FCOS is the operating system and is responsible for the low-level control over the general purpose computer (GPC). It provides support for Process Management, IO Management (“device drivers” for the IO devices), and Data Processing System (DPS) Configuration. FCOS also provides built-in support for redundancy management operations such as identical input/output and synchronization [1]. Systems Control provides support for the initialization and configuration of the system including the bus network [1].

The User Interface manages the interface between the crew and the software. Input from the crew is routed to the proper application as required. In addition to crew support, the User Interface also provides support for communication with the launch center over the Launch Data Bus (LDB) for pre-launch operations and the Mission Control Center (MCC) via Radio Frequency (RF) transmission.

Applications are composed of Guidance, Navigation, and Control (GNC), Systems Management (SM), and Vehicle Utility (VU) [8]. GNC is responsible for managing vehicle position and velocity for ascent, orbit, and entry. SM is responsible for managing and monitoring the systems of the vehicle and payloads. This includes performing fault detection, annunciation, and control of the environmental systems. VU provides support for the testing, integration, and certification of the vehicle during ground and in-flight operations. It provides both built-in self-test functions and command scripting under the control of the ground and/or crews to support vehicle testing. The command scripting is managed by VU's on-board Test Control Supervisor (TCS)

logic. The TCS accepts commands one at a time or in the form of a series of commands (procedure) for which required processes are automatically sequenced to completion.

The design of the PASS is based on a layered architecture (Figure 4). FCOS is the bottom layer and provides an interface through Supervisor Calls (SVCs). These calls and their data parameters are encapsulated for applications using a macro interface. The macro interface supports services such as input/output request, bus configuration, overlay requests, synchronization services, process management, interrupt enable/disable, error handling, and time formatting.

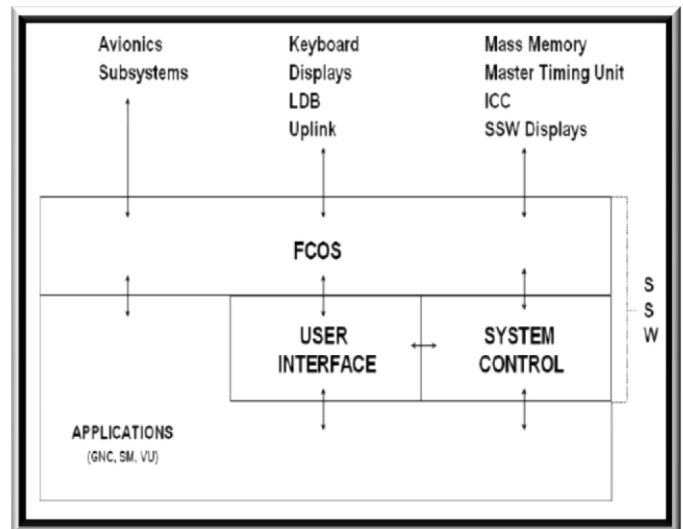


Figure 4. PASS Architecture

“GPC ON A CHIP” RESEARCH

From 2004-2006, research was conducted on the feasibility of implementing Space Shuttle avionics assets in Field Programmable Gate Array (FPGA) technology [4] [5] [6]. This technology allows digital electronic designs to be represented in a Hardware Description Language (HDL) where it can be synthesized to hardware components. One such component is a FPGA, which is a reprogrammable device (packaged as a chip) that can be embedded as part of electronic hardware boards. Use of such technology provides benefits such as the reduction of risk due to obsolescence and the ability to evolve hardware designs over time. Obsolescence is reduced because the digital designs are realized in a HDL that can be synthesized to new targets as technology changes over time. Designs can be evolved for the same reason. The HDL can be modified to fix hardware design flaws or incorporate new functionality and synthesized to the FPGA as needed.

This research proposed a conceptual approach where the digital design of the Space Shuttle GPC could be implemented in a HDL and synthesized to a FPGA to support

the reuse of the flight software on a SDLV. At the high level, the approach would allow for the instruction set of the GPC to be executed on a FPGA. The benefits include the reuse of FSW and the existing infrastructure required for developing, testing, and verifying the FSW. Another benefit would be that the key avionics assets could become expendable whereas new GPC and/or Multiplexer/Demultiplexer (MDM) devices could be synthesized from the HDL onto FPGAs as required.

Using this approach, the system could be evolved due to the flexibility of having the legacy digital design in a HDL. For example, the design of the IOP could be modified to support varying bus technologies, while isolating the changes from the legacy FSW. The instruction set supporting IO would be unchanged, but the design could support other technology such as 1553 or RS 422. Another aspect would be that a single GPC from the legacy system could exist as one of multiple cards in the backplane of a modern avionics computer. The other boards could provide advanced functionality not part of the legacy system such as advanced failure monitoring algorithms or special interfaces to support the payloads for the SDLV. Such a platform is shown in [Figure 5](#) from “Use of Programmable Gate Array Technology in Future Space Avionics”.

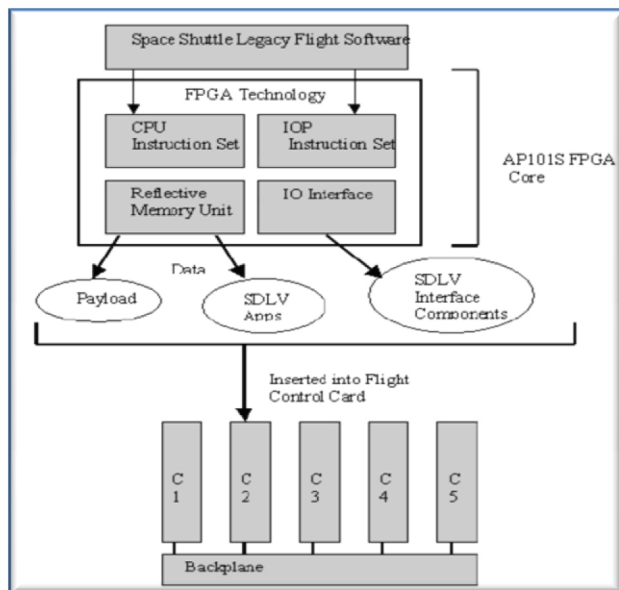


Figure 5. Legacy System in Conceptual Platform

During the research effort, the legacy Space Shuttle FSW was executed on an Altera FPGA development kit containing a prototype of a GPC on a chip. In essence, the prototype version implemented GPC functionality in a 40 mm by 40 mm package requiring less than 1 mw of power vs. the original 19.55 inches long by 7.62 inches high by 10.2 inches wide version requiring 500 w of power ([Figure 6](#)).

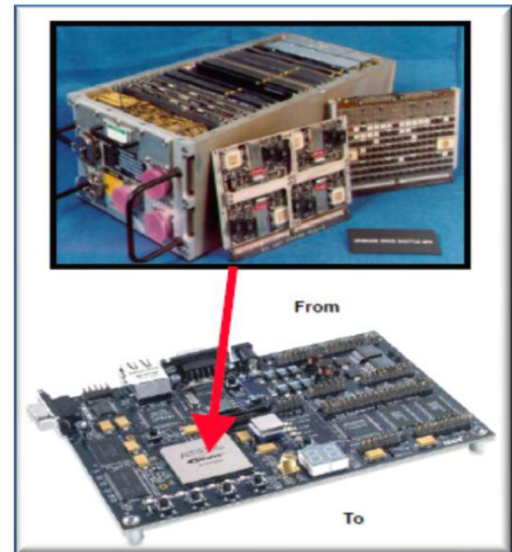


Figure 6. Prototype Repackage of GPC to FPGA

The FSW was executed on the setup in [Figure 7](#) which also provided support for legacy Shuttle displays ([Figure 8](#)).

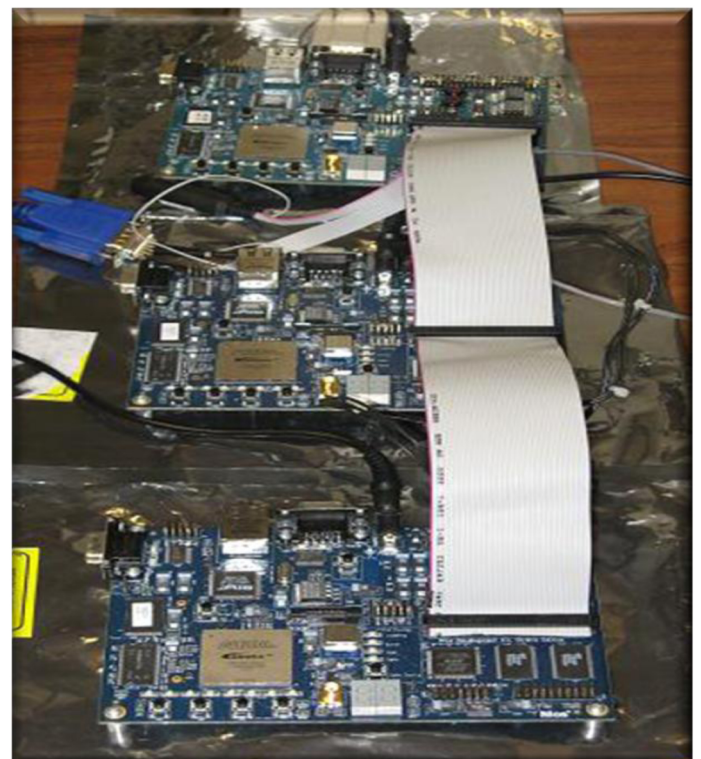


Figure 7. Prototype Setup