Honeywell

FACE Aligned EGI Platform Specific Service Adapter

US Army Aviation FACE™ TIM Paper by:
Honeywell Areospace

Matt Warpinski
Software Engineer

Robert Gjullin
Systems Staff Engineer

November, 2015

Table of Contents

Executive Summary

This paper describes the lessons learned based on a Honeywell effort to apply FACE Technical Standard 2.1 to an existing legacy Honeywell product in order to understand and identify:

- Issues associated with using FACE with legacy equipment
- Issues associated with learning how to apply FACE Technical Standard 2.1
- Issues associated with integration into an external lab environment

For this effort, Honeywell focused on the widely deployed EGI-764 navigation Line Replaceable Unit (LRU) as a representative example of a legacy system used by all the services and whose primary interface (MIL-STD-1553) is ubiquitous in military platforms. Having to deal with the scenario of legacy equipage will likely be unavoidable for many years for both government and industry when designing and integrating avionics hardware and software into military platforms.

Introduction

Today's avionics system development and integration has reached a level of complexity and size that any significant mission system development on a military platform is at risk of incurring significant cost and schedule impacts. Current military procurement approaches tend to favor development of avionics improvements that are aligned to specific platforms that have their own unique requirements that often do not translate well to other avionics systems on other platforms. This platform based approach provides little support or incentive to standardize architectures, hardware, and software so that development risk, cost, and schedule are reduced.

Open System Architectures (OSA) and Model Based Development (MBD) are potential mitigating solutions to this issue that seek to encourage the idea that architecture standards help constrain system development by establishing a core framework by which system interfaces encourage the idea of hardware and software component modularity and portability.

The Future Airborne Capability Environment (FACE) standard attempts to incorporate both OSA and MBD concepts to create a framework focused primarily on avionics software development with the goal of enabling more effective and efficient development and integration for both existing and future military platforms.

As a member of the FACE consortium and a developer of military and commercial avionics systems, Honeywell sees that the platform specific, stovepipe development approach will not be cost effective in the future given the trajectory of military spending pressures and the need to respond to quickly to emerging world threats.

It was with this in mind that Honeywell decided to explore the implications of adapting our existing products to interface with software that will be developed to the FACE architecture standard. Specialty Avionics LRUs like radios, Traffic Collision Avoidance Systems (TCAS), and Embedded GPS/INS (EGI) navigation units will be around for the foreseeable future and avionics software requires a cost effective approach to integrate with them. Toward this end, Honeywell created a FACE aligned prototype adapter component within the FACE architecture and supported its integration into a bigger, external avionics system whose goal was to demonstrate a mission re-planning function within the ARMY MIS Lab in Huntsville Alabama. This paper outlines the general approach and the lessons learned by Honeywell as a result of this effort.

Legacy EGI

For this effort, Honeywell focused on the widely deployed family of military aircraft Embedded GPS/INS (EGI) systems. They are self-contained, all-attitude, tightly coupled navigation systems providing outputs of linear and angular acceleration, linear and angular velocity, position, attitude (roll, pitch), platform azimuth, magnetic and true heading, altitude, body angular rates, time tags, and Coordinated Universal Time (UTC) synchronized time.

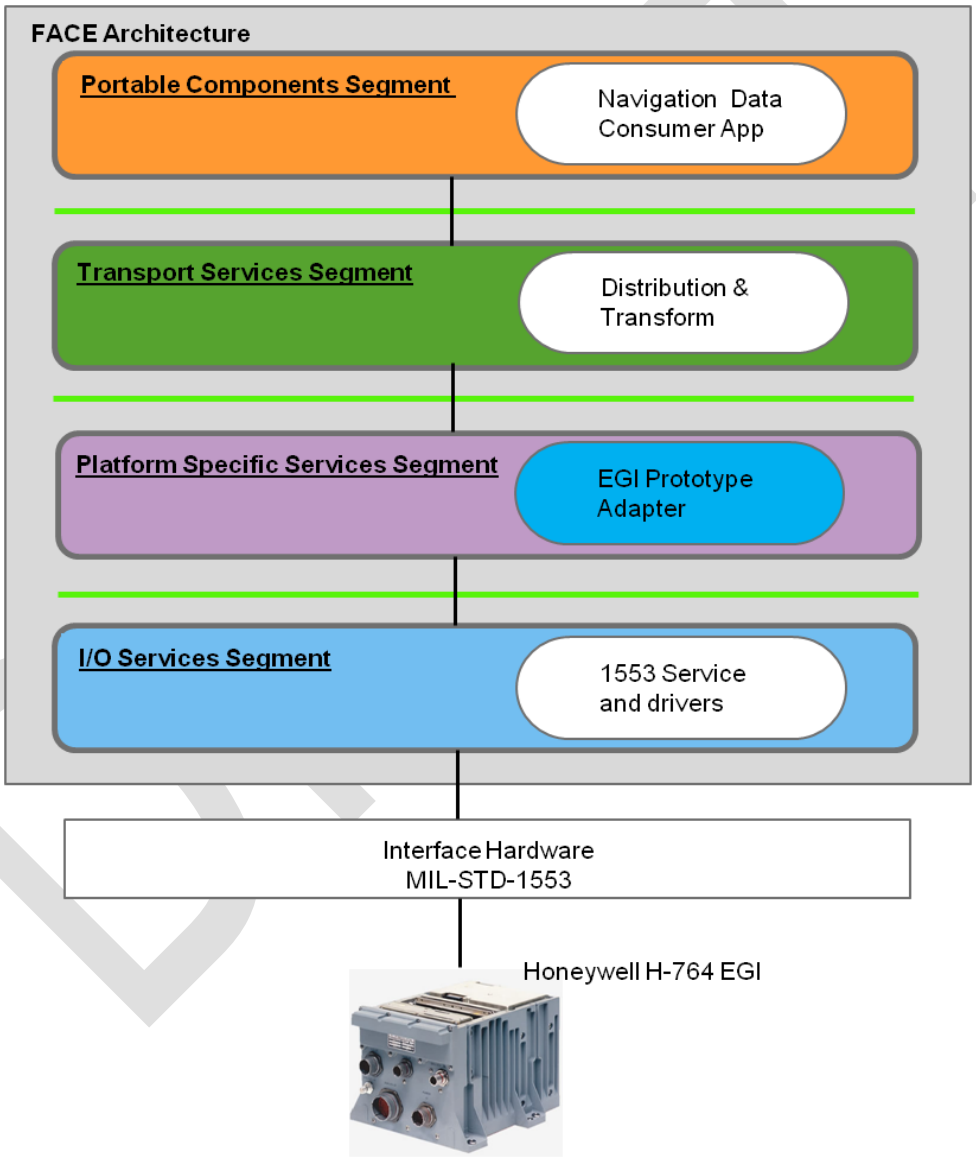

Figure 1: EGI adaptation for the FACE standard

EGI PSS Adapter Integration Lessons Learned

The primary H-764 EGI interface is physically a MIL-STD-1553 implementation which is specified in an Interface Control Document (ICD) that is close to 1000 pages long. For this demonstration, Honeywell selected 4 of its messages that provide high and low rate position, velocity, acceleration, and attitude information as the most generalized common data required by most platforms. **Figure 1** shows the H-764 interfacing with the I/O Services Segment (ISS) which was responsible for controlling the 1553 bus, capturing the H-764 1553 message data, and providing that data to the EGI prototype adapter (shown in blue) in the Platform Specific Services Segment (PSSS).

In this scenario, the integrator provided a 1553 Bus Controller (BC) for the system in which the EGI is a Remote terminal (RT) device. This of course means that the integrator had to study and understand the existing ICD in enough detail to manage the 1553 initialization, protocols, timing, and message traffic requirements just as they would have in a non-FACE implementation. In this case, the IOS 1553 service created and populated a buffer (in the message format specified by the FACE Technical Standard) with the 4 H-764 messages based on the rate at which they were provided. The PSSS EGI prototype adapter independently translated these messages into the data model specification format (discussed below) and stored the "one source of truth" navigation data locally for consumption by any Portable Components Segment (PCS) consumer applications that required it. The route re-planning demonstration mostly used low rate navigation data and the system performed as expected. No attempt was made to measure latency or timing associated with the high rate data.

Data Modeling

The 1553 message structure and content for the H-764 EGI has been added to over the years to accommodate a number of platforms specific needs and thus represents a superset of their requirements. This has led to a situation where multiple instances of a given data parameter (such as position or velocity) are transmitted in different reference frames and/or units. Additionally, the formats of the data are driven by the MIL-STD-1553 data formats and thus are structured as 16 bit words which usually capture numeric data as 2's complement scaled integers. These are the very types of issues that FACE wants to address with a combination of the data model and the Transport Services Segment (TSS).

Two decisions were required right up front in the data modeling effort:

- What format to translate and store the 1553 scaled integer data into for local storage?
- Should the data model design be aligned to the existing message structure and content?

For the first question, it should not matter what the local representation of the locally stored data is as long as it maintains the minimum accuracy, precision, and timing requirements of the original representation. Furthermore, if there are overlapping representations of the same data that differ only by things like units or reference frames, but not by accuracy, precision, filtering or timing, then those items can be consolidated to a single representation in the local storage.

For this effort, Honeywell selected integers and floating point formats for numeric values as well as Boolean and Enumerations for individual bit fields. In all cases, the selections were based on the original ICD specifications In the FACE architecture, the integrator may adapt the format and reference frames for PCS applications via the Distribution and Transform functions within the TSS of **Figure 1**.

The second question was more focused on whether to simply organize the local data in the same way that the original messages are structured (for simplicity relative to the installed base) or should the data model be designed to optimize some aspect of performance such as memory usage or timing/latency within the system. The intrinsic goal of portability can be achieved in either case. The argument for retaining the message data as currently structured by the EGI is based on the premise that existing platform applications have grown to expect certain data to be packaged together and provided as a block of information. This may reduce the transition impacts for training and implementation in legacy platform updates. The argument for optimizing the data structure to be orthogonal and efficient is self-evident but it quickly leads to the next question; what are you optimizing and why? This is a question that the integrator of a future system (or Family of Systems) would have to answer based on the particular mission requirements and the implementation of the avionics mission system itself. Since there is no driver for any particular optimization, Honeywell elected to retain the current message structure and bundling of data in the local data storage, thus maintaining the time correlation of the message data. It should be noted that if an alternative packaging of data was desired, the TSS can be made to accommodate the desired distribution and transform functions.

Finally, Honeywell used the standard FACE library in Enterprise Architect (EA) to create the model. Honeywell's observation is that it took significant time to create model entries in EA and then export them for conformance checking in the GME tool suite.

PSSS Function

After development of the Data Model the next step is in GME. Honeywell used the GME tool inside the SDK/IDK tool suite to model 4 components:

- Input Output 1553 driver
- Platform Specific Services EGI Adapter
- TS Component
- Portable Component Display

The link between the PSS EGI Adapter and TS Component and the TS Component and the Portable Component Display both used the Data Model that was created above. We were able to model this in GME and create auto generated code for all 4 components.

Within the PSS EGI adapter we created the logic to take raw 1553 messages and convert them into a FACE aligned data model. We focused on 4 1553 sub address messages 16, 17, 18, and 19 from the Honeywell EGI ICD. These 4 messages gave us all the data needed to provide a platform with all necessary data from the EGI. These 4 messages fully populated the data model.

Test Environment

In development of the EGI PSS, we needed an easy way to test the PSS software component. In order to do this a simple FACE framework was needed. Again we relied on the FACE SDK/ISK tool kit, using the SDK/ISK to create the addition IOS/TSS/PCS FACE components to move data through the system.

The PCS components focus was simply to display pertinent information for development. We made a quick OpenGL text display application that allowed us to display the relevant data so we could test the accuracy of that data transformations. The TSS component is needed to move the data from the PSSS to the PCS, in our application the TSS does not do anything other than move data.

Two considerations were needed when deciding how to design the IOS component. First, we did not have enough development 1553 devices to allow for development at our work stations so we needed a work around without using 1553. Second, without using 1553, we still needed to send 1553 FACE packets to the PSS component. We handled these by using UDP messages from our EGI sim to the IOS driver, then unpacking the UDP messages and packing it into the 1553 FACE structure to send to the PSS. This allowed the PSS to behave as it was getting 1553 messages without needing a 1553 bus attached to our development environment. As we were not delivering an IOS component this would not impact the final system but provided ease of development.

On the simulation side, Honeywell has extensive knowledge using X-Plane. We have existing X-Plane plugins that convert X-Plane data into 1553 packets. As noted above we needed to slightly modify one of these plugins to allow for sending on UDP traffic to our UDP->1553 IOS component. X-Plane allowed us to adhoc fly and test all the EGI data on the selected 1553 message subaddresses.

Integration Lessons Learned

Development of the PSS component was done with auto generated code from the Vanderbilt ISIS SDK/IDK toolset. Code generated with this toolset is designed to work in the included ARINC 653 ACM emulator. This emulator is not fully 653 compliant and contains some ACM specific function calls, specifically logging messages throughout the auto generated code.

The MIS lab that we were integrating into was using VxWorks 653. Before porting code over we needed to ensure our code would work in the VxWorks environment. As noted above we needed to get rid of the ACM logging messages, for integration we converted those to printf statement and needed to add stdio.h. We needed to add required asset.h header file, and remove the apex/face_acm_wrapper.hpp header file.

Integration for this effort went smooth with only one major issue. Honeywell and the MIS team made different assumptions for reading data from the IOS. The PSS component reads 1553 sub address messages 16, 17, 18, and 19. Each entry into the PSS requires 4 read calls to IOS to receive each of those messages.

Honeywell's development IOS centered on a while loop that read data from IOS. Each time we called read we would inspect the message header and process the data based on the message header. We continued to call IOS as long as we got "NO_ERROR" enumeration.

```
while(data still to send)
{
        Read()
        Process Data (Switch on 16, 17, 18, or 19)
}
SendData()
```

MIS assumpted that we would call IOS with a prepopulated header requesting message 16, 17, 18, or 19. IOS would return the data for that specific message. In this assumption IOS only maintained the latest data for each subaddress. After requesting each subaddress and processing that data PSS would then send the data.

```
Read( header.Subaddress = 16 )
Process16()

Read( header.Subaddress = 17 )
Process17()

Read( header.Subaddress = 18 )
Process18()

Read( header.Subaddress = 19 )
Process19()
```

SendData()

We elected to go with the MIS assumption because it was easier to change the PSS to call IOS as expected than to change IOS for the PSS assumptions.

Within the autogenerated code is the EGI_PSS_Process() function. This is the main process that is called each time to run the PSS. Within the ACM emulator environment that Honeywell used for development, this function would be called every time for PSS. Within the VxWorks environment this function would only be called once. We needed to add a while loop and periodic_wait() function call in the EGI_PSS_Process function to account for the changes between operating environments. Creating the following function:

```
void EGI_PSS_Process(void)
{
        printf("Invoked EGI_PSS_INTERFACE\n");

        while (1)
        {
                sub_address = 16;
                readAndProcess(sub_address);
                sub_address = 17;
                readAndProcess(sub_address);
                sub_address = 18;
                readAndProcess(sub_address);
                sub_address = 19;
                readAndProcess(sub_address);

                sendHighSpeedVector();
                sendStateVector1();
                sendTime();
                RETURN_CODE_TYPE RETURN_CODE;
                PERIODIC_WAIT(&RETURN_CODE); //wait for next period
        }
}
```

As a last integration note, Honeywell and MIS has licenses for different VxWorks environments. Upon compiling the two together we received a handful of errors. Two compiler flags were needed to allow Honeywell to build an object file in VxWorks 3.x and link in MIS's VxWorks 2.x enviorment. Those flags needed are:

- -fno-exceptions
- -fno-use-cxa-atexit

Summary

In this effort, Honeywell determined that the concept of adapting a legacy avionics LRU and its functionality to the FACE architecture with the goal of abstracting its peculiar interfaces away is achievable. The effort to do so still requires an integrator to understand all aspects of the LRUs interface and functionality. The design of the data model is not necessarily a trivial effort. It will require trade-off decisions to be made by future integrators and those trades may lead to different decisions depending on whether the mission system being created or updated is on a legacy platform or for a whole Family of Systems (FoS) like being envisioned by the FVL program.

Honeywell believes that initiatives such as the FACE standard are going to be required if the government is ever going to be able to address portability and reuse of hardware and software avionics systems. The concepts of a standard that drives to control and manage "one source of truth" data and interfaces that minimize the opportunity for closed, stovepipe architectures are key.

About the Author(s)

Matt Warpinski is a Software Engineer with Honeywell Aerospace Advanced Technology. He is a member of the FACE Consortium and has actively been participating for over a year as a member of General Enhancement and Transport Services Technical Subcommittees. He was responsible for taking the developed Data Model and software and integrating into the MIS lab.

Robert Gjullin is a Systems Engineer with Honeywell Aerospace Advanced Technology. He is supporting Honeywell's efforts for FACE, Joint Common Architecture (JCA), and Future Vertical Lift (FVL) activities. He was responsible as the overall Principle Investigator for the FACE EGI adaptor.

About The Open Group FACE™ Consortium

The Open Group Future Airborne Capability Environment (FACE™) Consortium, was formed in 2010 as a government and industry partnership to define an open avionics environment for all military airborne platform types. Today, it is an aviation-focused professional group made up of industry suppliers, customers, academia, and users. The FACE Consortium provides a vendor-neutral forum for industry and government to work together to develop and consolidate the open standards, best practices, guidance documents, and business strategy necessary for acquisition of affordable software systems that promote innovation and rapid integration of portable capabilities across global defense programs.

Further information on FACE Consortium is available at www.opengroup.org/face.

About The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through IT standards. With more than 500 member organizations, The Open Group has a diverse membership that spans all sectors of the IT community – customers, systems and solutions suppliers, tool vendors, integrators, and consultants, as well as academics and researchers – to:

- Capture, understand, and address current and emerging requirements, and establish policies and share best practices
- Facilitate interoperability, develop consensus, and evolve and integrate specifications and open source technologies
- Offer a comprehensive set of services to enhance the operational efficiency of consortia
- Operate the industry's premier certification service

Further information on The Open Group is available at www.opengroup.org.