OA-II VEH COM System Design

DR00003

Rev: A01

Jinzhi Cai

2019–07–22

Table of Contents

1 Introduction

1.1 Scope

This document is describe a design for the OA-II VEH Computing and Operation Module.

1.2 Purpose

The goal for the design is to build a system that allow real time operation and multimedia processing happen in the same time during the flight.

2 Revision History

Rev#	Editor	Delta	Date
A01	Jinzhi Cai	Initialize	2019-7-20

Table 1: Summary of Revision History

3 Requirement Analysis

3.1 Failure Recovery

In the OA-II VEH system requirement,failure recovery is always the most important part. The main requirement for failure recovery is in below.

Backup For all the mission critical system, it mast have at least have one backup unit.

Fast Switch When emergent scenario has been detected, the system will switch to backup unit within limited time and functioning.

Hierarchy Failure Recovery System For different emergent scenario, failure recovery system have different plan to deal with to protect device and data.

3.2 Realtime Operating System

In the OA-II VEH system requirement, realtime operating system is the key to proform all the critical procedure.

Predictable Execution Time All the procedure will need to execute on time with limited of error.

Interface with the Main System The RTOS will need to be have to receive command from other system.

Execution Feedback When a procedure is executed, it have to be comfirm with sensor.

4 Current Controller Analysis

4.1 Arduino

Arduino is a common use in DIY electronics. It have an abundant library for different IC. It also friandly for the beginner. However, most of the arduino library is not effection and the interface choose is limited. Its main clock is running at 16MHz which is not enough for support a functional linux system.

4.2 STM32 Serise

STM32 Serise is a common use microcontroller and it support up to 400MHz main clock. This kind of microcontroller also support to running a operating system. However, as the arduino, it lack the freedom to choose the the interface.

4.3 Market SoC System

In the market, it have a lot of different SoC system, such as raspberry Pi and BeagleBone. It do come with a lot of different interface. However, it do not have a high speed interface what can use to connect between different system.

4.4 FPGA Soft Core Processor

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer after manufacturing – hence the term "field-programmable".[**?**] Almost every FPGA company will have it own soft core processor. Those processor existing in code form and allow user to rebuild them in the FPGA chip as long as the chip have enough resource. Even the soft core processor allow user to costume the structure, the speed of soft core processor will not as fast as the SoC processor. However, it is fairly easy to interface to other system.

4.5 SoC FPGA

The SoC FPGA basicly also is a kind of FPGA. The only different it embedded a powerful processor as know as hard core processor. It help fuse the benefit between the SoC chip and FPGA chip. On one hand, it allow big system running in the powerful hard core processor. In the same time, the flexibility of FPGA chip allow costume interface and upgrade ability. However, SoC FPGA have less logic resource compare with the classic FPGA and weak proessing power compare with SoC which in the same price.

5 Recommend System Design

5.1 SoC FPGA

In this design, the main processor will be a SoC FPGA that have about 70K LUT logic resource and dual core 800MHz - 1GHz hard core processor.

IP core or Logic	Resource Cost
Blackbone Bus Controller	50K(Max)
Low Speed Sensor Controller	10K
Soft Core Processor	9K
Debug JTAG Core	1K
Summery	70K

Table 2: Table of Logic Resource

The hard core processor will be collecting data from the bus and sensor controller. It will storage those data and use it to resolve the rocket attitude and location. The result will transfer to command and send to the realtime soft core processor to control the PAM.
The FRU will have the same hardware structure as the MCU, but constantly get data from the MCU for error checking.

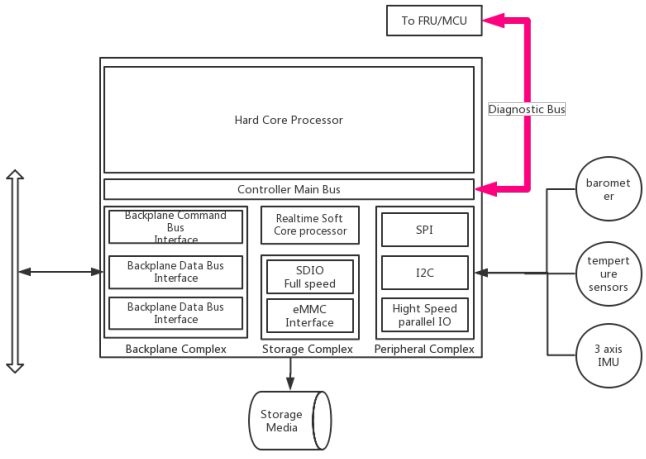

Figure 1: Block Diagram For COM System