Proposal for an Ultra-Cheap, Fault-Tolerant Command & Data Handling System

Alexandria Lin*[1], James Roach,+[2] David L. Cotten^[1]
*alexandria.lin@uga.edu, +james.h.roach@nasa.edu, ^dcotte1@uga.edu
[1]Small Satellite Research Laboratory, University of Georgia, [2]NASA Ames Research Center

Introduction

In recent years, the space industry has undergone a transformation with the emergence of low-cost, miniaturized electronics that possess advanced capabilities in computing power and robustness. Commercial-off-the shelf (COTS) hardware is increasingly being utilized in a variety of small satellite missions, extending from low-earth orbit missions to deep space. Despite these recent advances, many of these systems are still far more expensive than consumer electronics due to the rigors of the space environment that can pose a threat to spacecraft.

We propose a fault-tolerant system using BeagleBone Black boards and FPGA that is capable of performing all the command- and data handling needs of a small satellite at a fraction of the cost of current solutions on the market. Our proposal will address the problems posed by power, radiation, compatibility, and redundancy considerations as well as the steps our system takes to mitigate those problems. The adoption of this system could assist in the growth of inexpensive but complex satellites on an unprecedented scale.

Overview of the BeagleBone Black

BeagleBoards are a line of low-powered, low-cost, open-source computers with a range of peripherals for rapid prototyping. The latest edition of the BeagleBoard, the BeagleBone Black (BBB), has 512MB of RAM, a clock rate of 1 GHz, and 2GB of onboard eMMC flash memory. The BBB has flight heritage for Linux-based CubeSat missions. However, these missions are not long-term due to the short life expectancy of the BBB boards in LEO, and place BBBs squarely in simplistic sub-Class D missions of this type.

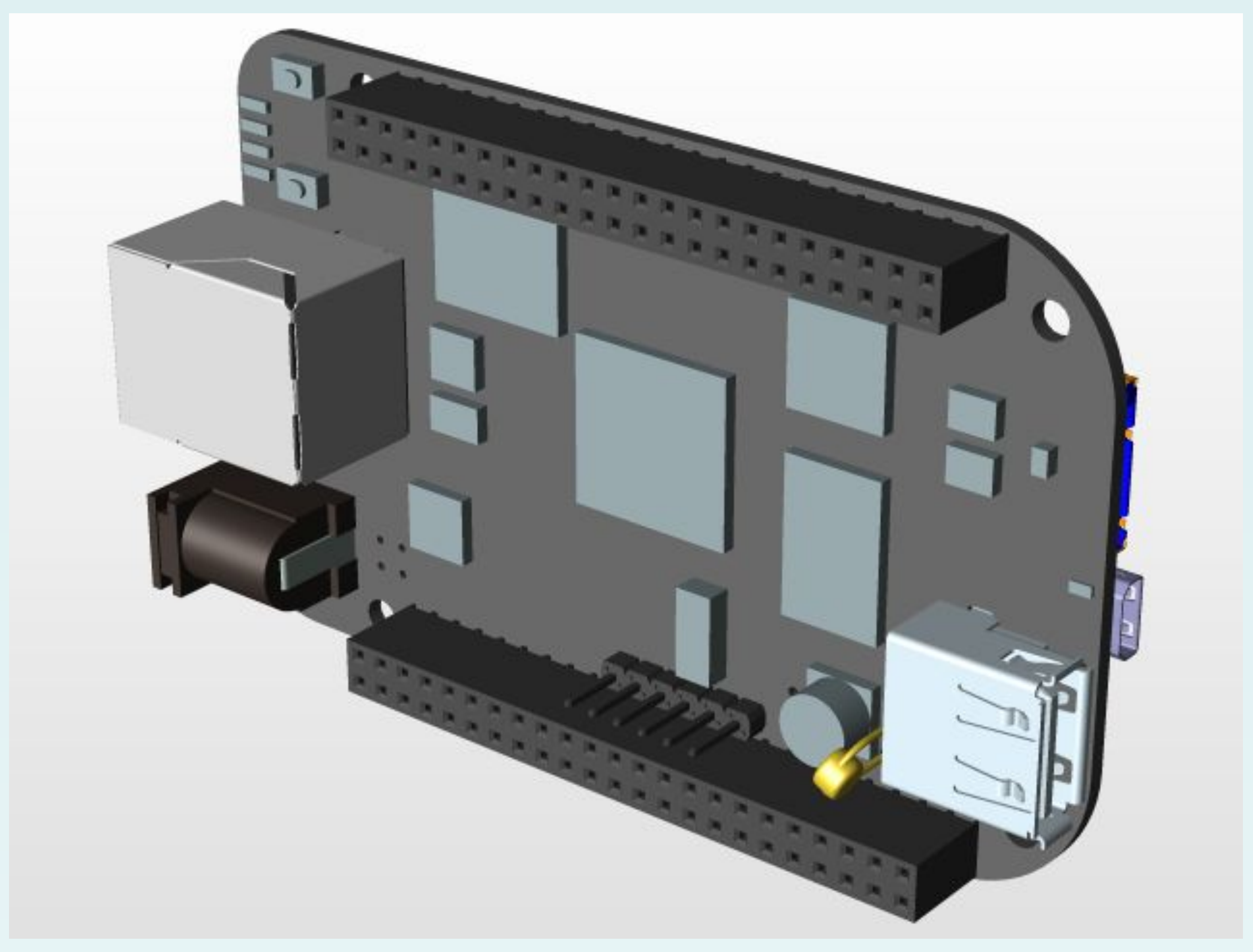

Figure 1: CAD Model of BeagleBone Black

Approach

Our proposed fault-tolerant system implements a form of practical Byzantine fault tolerance (pBFT) using a cluster of four redundant BeagleCore (BCM1.ETR) modules, which are a smaller, more ruggedized version of the BBB boards with wider operating temperature/humidity ranges and are more industrialized. The number of redundant nodes is derived from Lamport's paper that concluded a minimum of *3n+1* nodes are needed to tolerate *n* Byzantine faults with the use of digital signatures. The BCMs write telemetry and mission data to a shared flash memory block over SPI, which prevents use of the erroneous micro-SD connector and the failure of a single BCM to cause lost data.

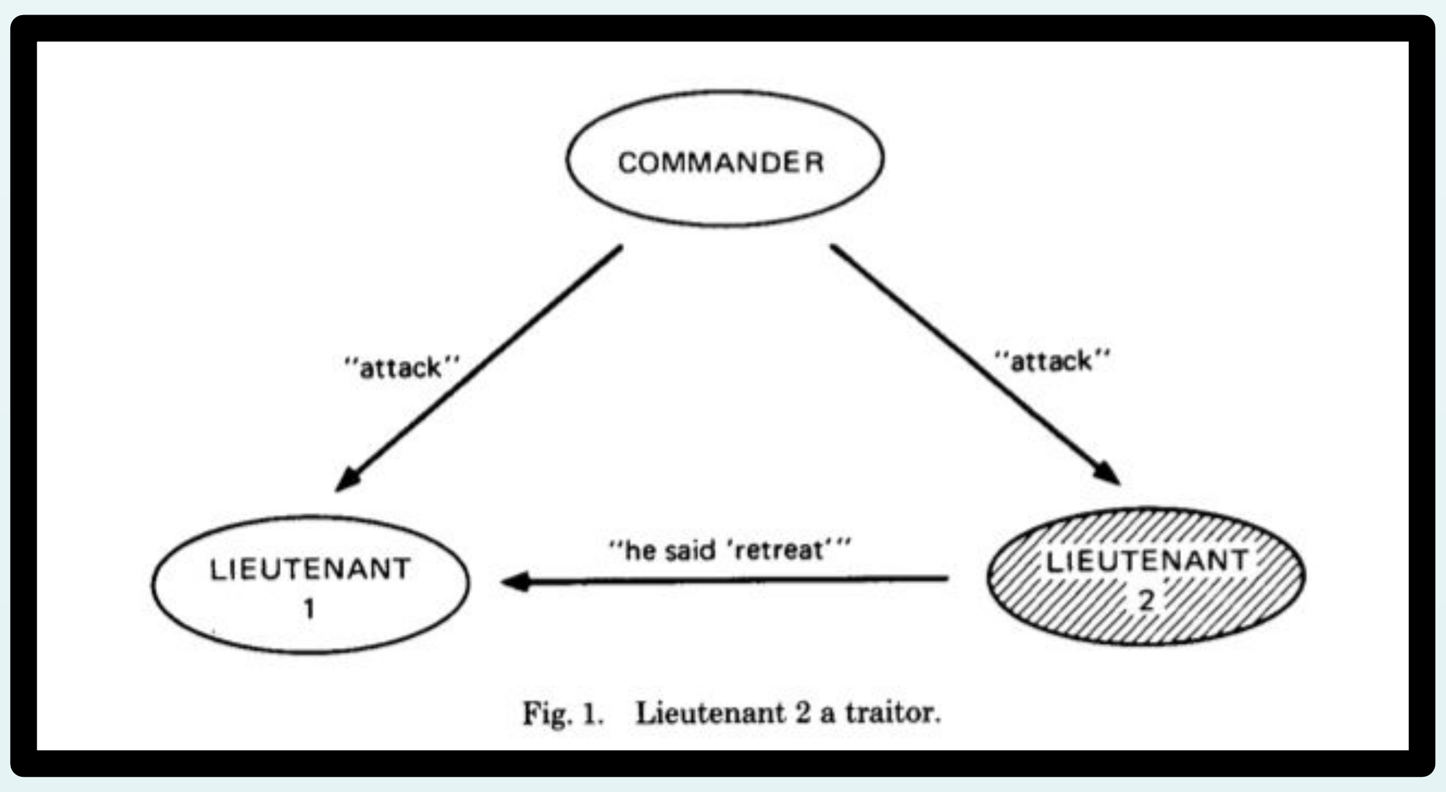

Figure 2: Example of Byzantine Allegory

Sensors and payloads are routed to the cluster using a breakout board. Additionally, although the BCM cluster only has an added latency of a microsecond, the design includes a low-power FPGA that is directly interfaced with the components in the core avionics stack of the spacecraft such that when critical errors are detected (i.e. from the EPS), the FPGA can respond with minimal latency and place the satellite in a safe state and perform other crucial tasks.

Cluster Operation

A single BeagleCore module is designated as the "primary" node in the cluster, with the other nodes in a heartbeat/idle state. When a request from ground is received or internal data handling tasks are being completed, the primary node broadcasts the requests to the two other nodes. All nodes perform the operation and send a reply back to the primary, which successfully completes the operation if and only if three or more of the replies contain the same result and the other nodes are in agreement.

The nodes also periodically transmit heartbeats to all other nodes; should a node not receive a response from the primary node first, the primary node itself has a minority result, or the other nodes are not in agreement, the other nodes use a TMR voter comparator to designate a new primary node. Even if a BeagleCore fails, nominal operation can proceed, and decision making is computationally inexpensive. At any given time, the FPGA is allowed to override the cluster decisions.

Software Fault Tolerance

By using pBFT for a higher level of fault tolerance and recovery, the onboard software itself should not try to catch and raise exceptions if faults are detected, but should be developed using *n*-version programming to reduce the number of possible result inconsistencies.

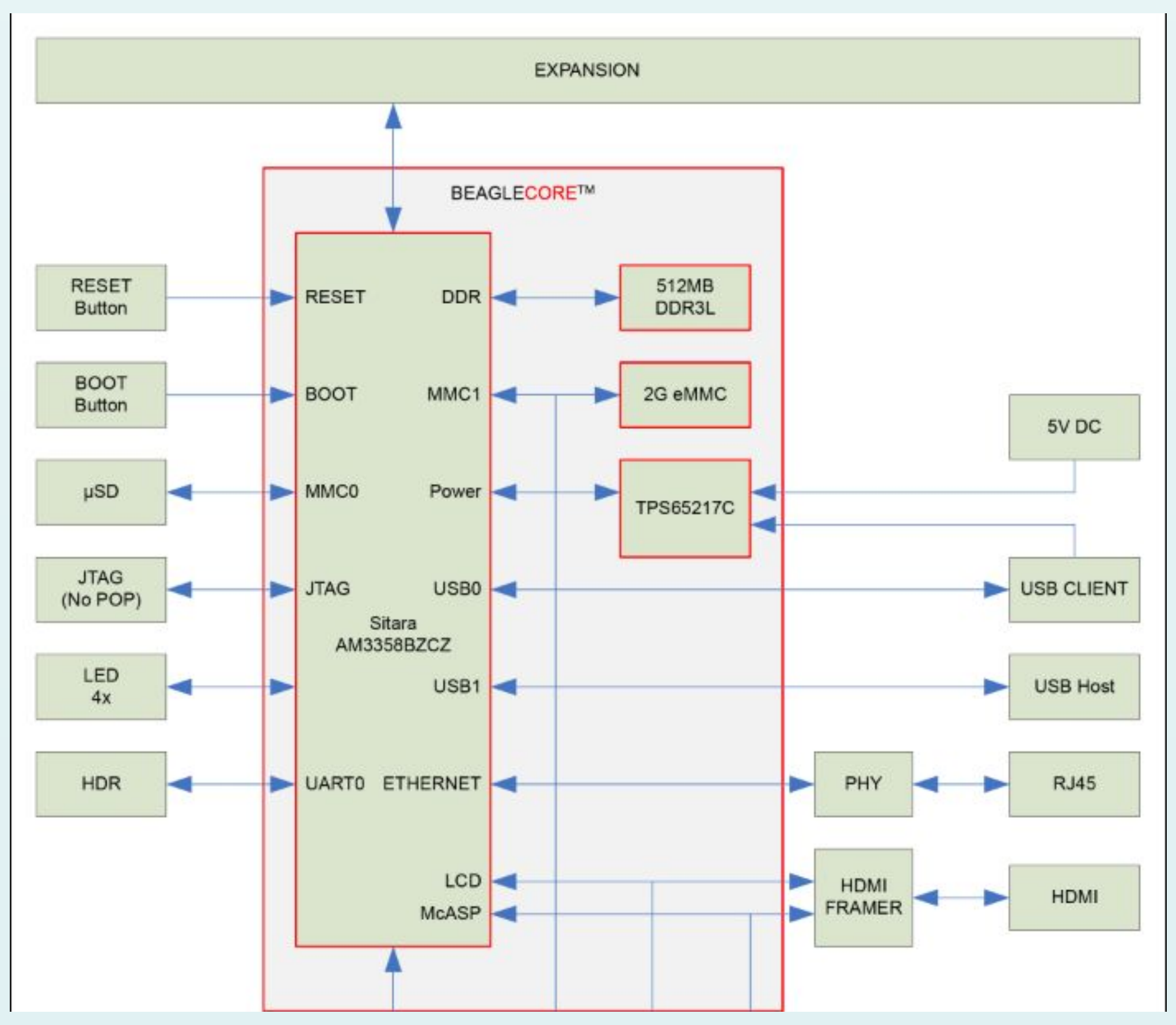

Figure 3: Block Diagram of BCM

Discussion and Future Work

We need to further analyze the power consumption of the current design, design the breakout board for sensor input and the FPGA, and begin prototyping the system.

References

Lamport, L. and Shostak, R. "The Byzantine Generals Problem". SRI International, 1982

Castro, M., and Liskov, B. "Practical Byzantine Fault Tolerance." Proceedings of the Third Symposium on Operating Systems Design and Implementation, Feb. 1999.