Antonio Roldao[1], Martin Unwin[1] and Tanya Vladimirova[2]

[1]Surrey Satellite Technology Ltd.
[2]Surrey Space Centre

ABSTRACT

As Surrey Satellite Technology Limited (SSTL) moves beyond Low Earth Orbit (LEO) missions, there is a need to develop more robust and radiation tolerant systems. Such systems must be flexible and fast enough to meet customer requirements while keeping costs down. Such flexibility has become possible through the extraordinary developments in Field Programmable Gate Arrays (FPGAs) technologies. With families ranging from simple low cost, to full-blown hybrids that include multiple embedded processors, FPGAs provide an ideal alternative solution to Application-Specific Integrated Circuits (ASICs). In shifting from Commercial Off-The-Shelf (COTS) ASICs and moving towards FPGA based systems, SSTL is able to take advantage of an unprecedented freedom. Whereas before, FPGAs were only used to provide glue-logic and simple functionality, such as of a Triple Modular Redundancy (TMR), now these devices have become capable of agglomerating the Central Processing Unit (CPU) as well as various peripherals.

This paper describes recent developments at SSTL, where an existing FPGA design consisting of an interface and an Error Detection and Correction (EDAC) module was extended to integrate a controller area network (CAN) module. Several CAN intellectual property (IP) cores were investigated for this application. The European Space Agency's (ESA) HurriCANe core was selected as it is designed for space use, it is small enough for the targeted FPGA, and the source is available for European space applications.

The design and the experiences with the development of this IP Core will be introduced including the purpose-built wrapper and glue logic. It will be described how, by taking advantage of the soft-core nature, the CAN core was adapted to better mesh with the CAN protocol adapted by SSTL for Spacecraft Usage (CAN-SU). Analysis of the advantages and disadvantages of taking such an approach, compared with either using COTS or by having a fully featured System-on-Chip (SoC) solution will be presented. The paper will also provide a glimpse at the new developments of a Satellite Generic SoC (SG-SoC) that incorporates all the basic functionality of an On Board Computer (OBC), from which by simply plugging-in additional modules, this SG-SoC can easily become a task specific sub-system, such as a Star Sensor or a GPS receiver.

Introduction

Traditional space industry is extremely conservative in terms of both methodologies and systems. This combination fuels a vicious cycle where missions are infrequent and expensive. Since there's no opportunity to fail, conservative engineering methods and lengthy verification procedures are applied. Hi-reliability parts and custom-made system are also the norm. These approaches require extensive resource usage and are extremely time consuming, therefore missions are infrequent and expensive. In such approaches quality is supreme, cost and performance secondary.

Fig 1 - Traditional Space Approach

In SSTL's space approach quality, price and performance are all factors in the customer's decision. Because flight opportunities are frequent, systems need not to be perfect, so they are less expensive. Engineering is practical, performance and cost-driven. Satellites are also built in batches and try to evolve as much as possible from heritage designs. Commercial of the Shelf (COTS) parts are managed into an acceptable risk level, thus reducing overall costs. Therefore missions are cheaper and more frequent. [1]

Fig 2 - SSTL's Space Approach

This section gives a real example on applying such an approach to the development of a GPS receiver. It describes the usage of the CAN COTS protocol, managing radiation effects by applying TMR and the use of FPGAs, as a method of reducing costs and managing risk. These experiences include the study of the inner-workings of CAN, the comparison between different available IP CAN cores, the study and implementation of a memory TMR IP core, the adaptation of the selected CAN core (HurriCANe) to better suit CAN-SU, the integration of the CAN and TMR core with a custom built wrapper and finally the development of drivers and test software.

CAN - Controller Area Network

Controller Area Network (CAN) is a standardized serial data communication protocol. Also known the ISO 11898 standard, this protocol was initially aimed at the automotive industry. Currently CAN is used on a large variety of machines, from intelligent houses to satellites. Originally developed by the German company Robert Bosch in the 1986, this standard has provided a cost-effective, reliable and robust communications bus.

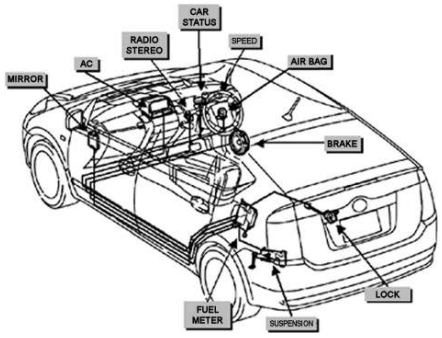

Fig 3 – Car using CAN

This standard posses some features that make it especially attractive which include:

- **Mature Standard** – CAN has been around for almost twenty years. This enabled the protocol to mature giving room to various tools and products. Because it's widely used in the massive automotive industry, units that implement this standard are cheap, reliable and readily available.

- **Hardware Implemented** – Because this protocol is implemented in silicon it is possible to combine other functionalities, such as error handling and fault confinement facilities. It's also possible to optimize the available bandwidth and achieve higher transmission speeds.

- **Simple Transmission Medium** – This protocol needs only a pair of twisted wires to function. However one of its features is being able to operate with only one wire, in case of fault.

- **Error Handling** – This is one the strong points of CAN. With extensive mechanisms to detect CAN is a strong reference for safety critical systems.

- **Fault Confinement –** A single node can occupy the bus and thus render the whole medium useless. This is why CAN was developed with fault detection and confinement mechanisms, where faulty nodes get automatically disconnected until the bus returns to normal operation.

- **Collision Detection and Resolution –** Whenever two nodes start transmitting simultaneously instead of having a scrambler signal being sent, as it happens with the Ethernet protocol, CAN resolves to allow the highest priority node to carry-on while the lowest gives-in. However the lowest priory node is still able to receive the message. There's also a special error frame that is sent every time there's an error.

Using this protocol data can be sent at rates up-to 1 Megabits per second (this limitation is easily overcome using capable equipment). Data is sent on frames that consist of 8 fields: [2]

3. **Control** – Consisting of six bits; two reserved bits (r0 and r1) and a four bit Data Length Code (DLC). The DLC indicates the number of bytes in the Data Field that follows.

4. **Data** – Contains a range from zero to eight data bytes.

5. **CRC** – Comprises of a fifteen bit cyclic redundancy check code and a recessive delimiter bit.

6. **ACKnowledge** – Consisting of two bits; the first, being the Slot bit, which is transmitted as recessive, but is simultaneously over-written by a dominant bit transmitted from any other nodes as an indication of successfully receiving the message; the second bit is a recessive delimiter bit.

7. **End of Frame** – Comprises of seven recessive bits indicating the end of frame

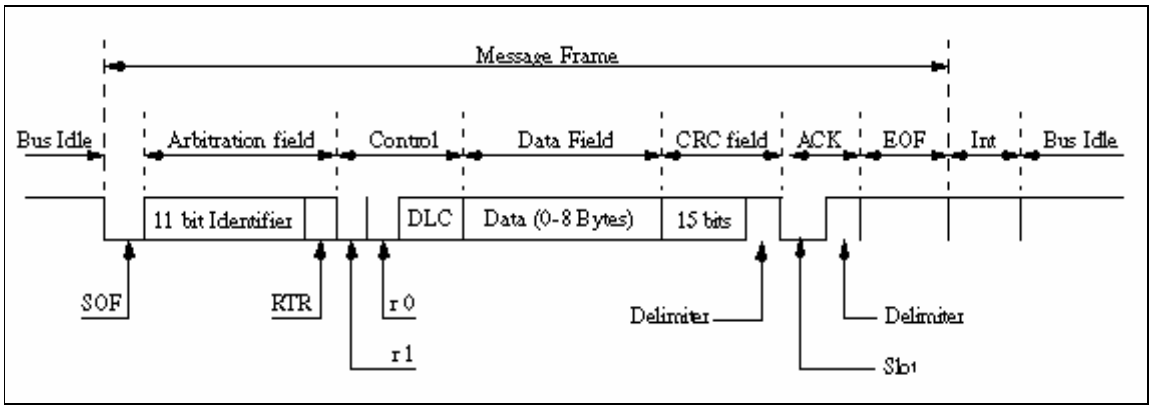

Fig 4 – Controller Area Frame Fields

1. **Start of Frame** – This indicates the beginning of a message frame.

2. **Arbitration** – This field contains the message identifier and the Remote Transmission Request (RTR) bit. The RTR bit is used to discriminate between a transmitted Data Frame and a request of data from a remote node.

The **Arbitration** field is used for two purposes, to identifying the class/message type and to resolve any collisions. The message priority is encoded using the identifier. The lower this identifier is the higher its priority. This is illustrated on Figure 5 where three nodes start transmitting a frame at the same time.

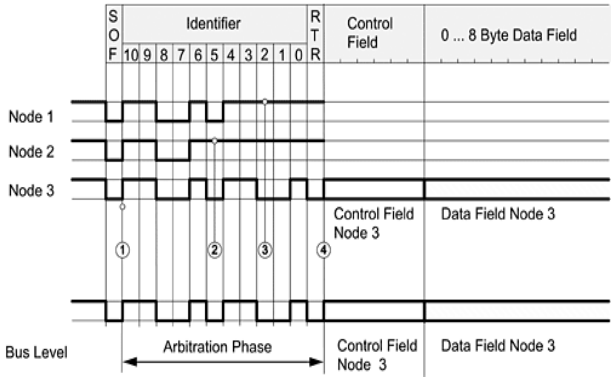

Fig 5 – CAN Collision Resolution

Three nodes start competing for the bus at ①. They transmit the first five bits of their identifiers as they are the same. At ② node two gets into listening mode as it has lower priority, same thing happens to node one at ③. Finally node three that has the highest priority gets to transmit its message ④.

CAN was developed for safety critical applications as such it is highly redundant and error prone. To accomplish this, a series of mechanism where implemented to detect and isolate faulty nodes. Detecting errors is accomplished using 5 mechanisms, three at the message level and two at the bit level:

Message Level

1. **Cyclic Redundancy Check** – A 16 bit function of the transmitted data.
2. **Frame Checks** – Certain values must appear at certain positions.
3. **Acknowledgement** – Acknowledgements must be received.

Bit Level

1. **Bus Bit Level** – An error is flagged if the bus level is not as expected.
2. **Bit Stuffing** – Within a message transmission no more than five consecutive values of the same bit are allowed. To prevent this, an opposite bit value is added. This value is then automatically removed.

In case an error is detected by any node, an **Error Frame** is sent out, thus preventing any other node from accepting the message and ensuring consistency of data throughout the bus.

During normal operation, a CAN node will be in **Error Active Mode**, but once its Error Counter Register (ECR) contains more than 127 errors it switches into **Error Passive Mode**. In this later mode, the node will handle errors in more sensitive matter. If the ECR exceeds the value of 255 then the node enters into **Bus-Off Mode**. In this mode the device will cease to be active on the bus until re-initialized by the user. However communications between the other nodes will remain operational.

A total residual error probability for undetected corrupted messages less than "message error rate * 4.7 * 10^-11" puts this protocol in a very safe position. [2]

EDAC - Error Detection and Correction

As a result of the Earth's magnetic properties a combination of electrically charged particles gets trapped around it. These particles form two belts, the Van Allen Belts. The inner belt is mainly populated by protons, a heavier particle. The outer in turn is mostly comprised of electrons. Such radiations possess a hazard, not only to electronic system but also biological life.

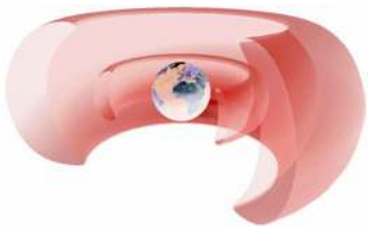

Fig 6 – Van Allen Belts

As satellites go beyond Low Earth (LEO) Orbits, as is the case with GSTBv2, it is necessary to consider radiation hardening. At these orbits, radiation levels can be more than a one bit upset per megabyte of SRAM a day. To protect against this, one can either use customized expensive solutions or manage the risk into an acceptable level.

To address these radiation hazards a simple and relatively cheap Triple Modular Redundant (TMR) system can be implemented. Such a system consists on replicating events on-to three equal basic sub-systems, therefore allowing for one to go wrong without compromising the system. This principle can be applied to very complex systems or to simple ones. In the case of a memory system that is divided into read and write cycles, this would require the data to be processed in the following order:

A VHDL IP Core that implemented this functionally was developed using an asynchronous logic. This core also included some additional functionality to decode chip-selects to their memory address space. This allowed for RAMs and the Flash device to have different wait states configurations.

EDACAN

The SGR-GEO initial design contemplated a dedicated CAN controller. However the schematic was arranged so that in the future it would be possible to integrate this device into the FPGA. To achieve this goal it was required to investigate on the various CAN cores available. Important points to consider included the FPGA's (an Actel 54SXA32) physical limitation in terms of gates on and the overall costs.

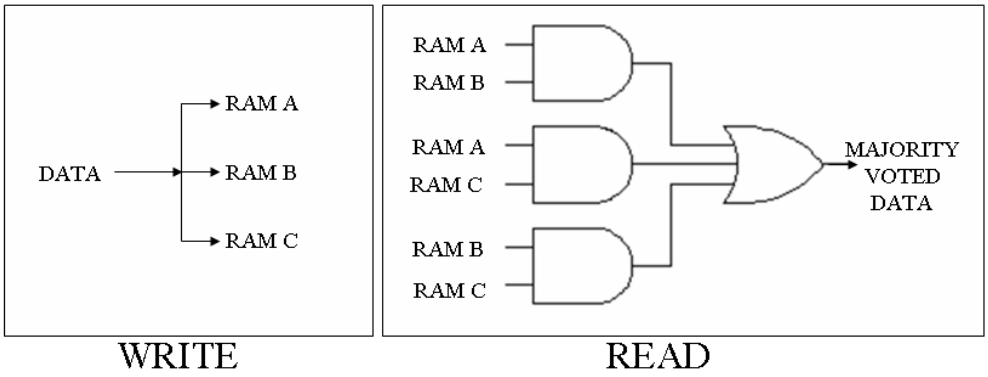

Fig 7 – TMR Memory Write/Read

During the write cycle data is written equally into each individual RAM bank. During the read cycle data must pass a pooling in which the majority vote is propagated to the output.

Using a Karnaugh map it's easy to demonstrate how any bit flip (or Single Event Upset) is corrected.

Table 1 - Karnaugh Map

C\AB	00	01	11	10
0	0	0	1	0
1	0	1	1	1

These various CAN cores were considered as possible candidates:

- **HurriCANe** – A European Space Agency development. It was already available at SSTL through an agreement previously signed. Though due to the lack of documentation it was never tested.
- **Actel CAN IP Solution** – Provided by the same manufacturer as the FPGA, would be complete with support and documentation. However it would not be free.

- **Open Cores CAN** – This core is freely available under the General Purpose License. However it was too big for the target FPGA.

- **CAST CAN Solution** – From a well organized group, this core included a tested and verified solution for the target FPGA. But as with the Actel's solution, it would not be free.

After spending time understanding how does CAN work and exploring the way these functionalities were implemented on the HurriCANe core, confidence around this solution increased. However other alternatives weren't straight away discarded.

HurriCANe is a Soft IP core that implements the CAN 2B specifications. Initially it was developed as prototype to study the internals of CAN, then as low priority task, was gradually improved into a full controller. It is being used in SMART-1 and on ATV (Automated Transfer Vehicle). [3]

To carryout this project the Actel's Libero Integrated Development Environment (IDE) in conjunction with Mentor Graphics Modelsim were used.

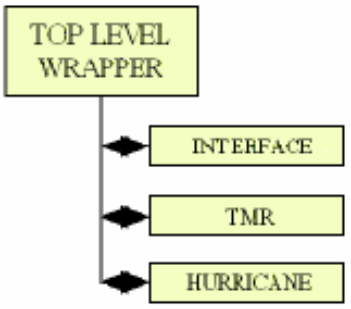

Fig 9 – EDACAN Top Wrapper

A Wrapper that incorporates both the TMR and the CAN controller was developed. This Wrapper also included other additional logic used for the decoding of Chip Selects (CS). With this logic it was possible to allocate certain memory spaces, with their own wait-states, to each device.

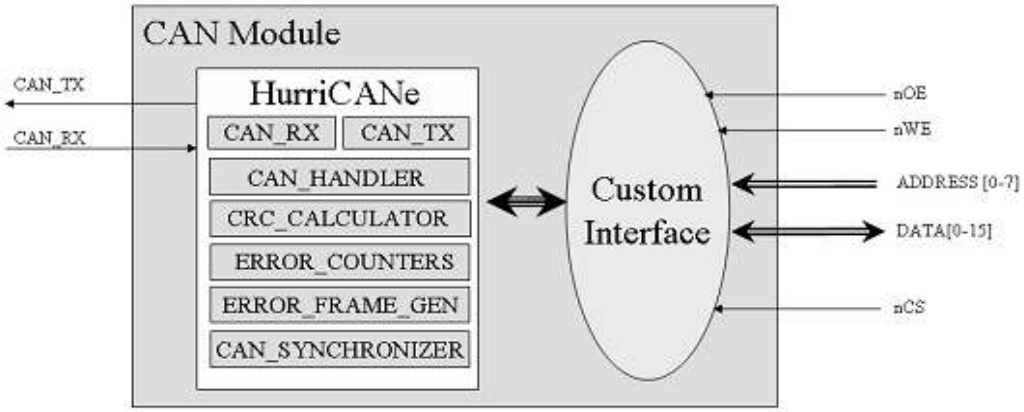

Fig 8 – HurriCANe IP Core Internals

With a better understanding of HurriCANe and with the successful simulation of its operation, it was decided to build a custom made wrapper that would integrate its functionality with the TMR. This TMR was initially based on a previous work done by David Cooke. This small sized VHDL project was an ideal starting point for someone without any FPGA and HDL experience.

With the integration of EDACAN into the interfacing FPGA, it was possible to eliminate the need for the SJA1000 CAN peripheral integrated circuit. This resulted in a compact and less power consuming receiver.

In order to operate this custom tailored core it was necessary to develop low-level drivers from scratch. These drivers were implemented using a mixture of ANSI C and ARM assembly language. Having completed both the hardware and the driving software, a multitude of tests were performed. These tests which verified the proper operation of the Controller Area Network and the Triple Modular Redundant memories were successfully carried out. Finally the receiver was integrated into the spacecraft and if everything goes accordingly to plan it will be in space before 2006.

Future Developments

Following the tendency to further integrate and allow for a soft-core flexible system, the next step is to integrate the micro-controller and other peripherals onto the FPGA. To achieve this integration, various micro-processors soft-core are being examined to determine a flexible solution that could be reused in other sub-systems. This Satellite Generic System-on-Chip would then only require the addition of specific modules to become a specific subsystem, such as a GPS Receiver.

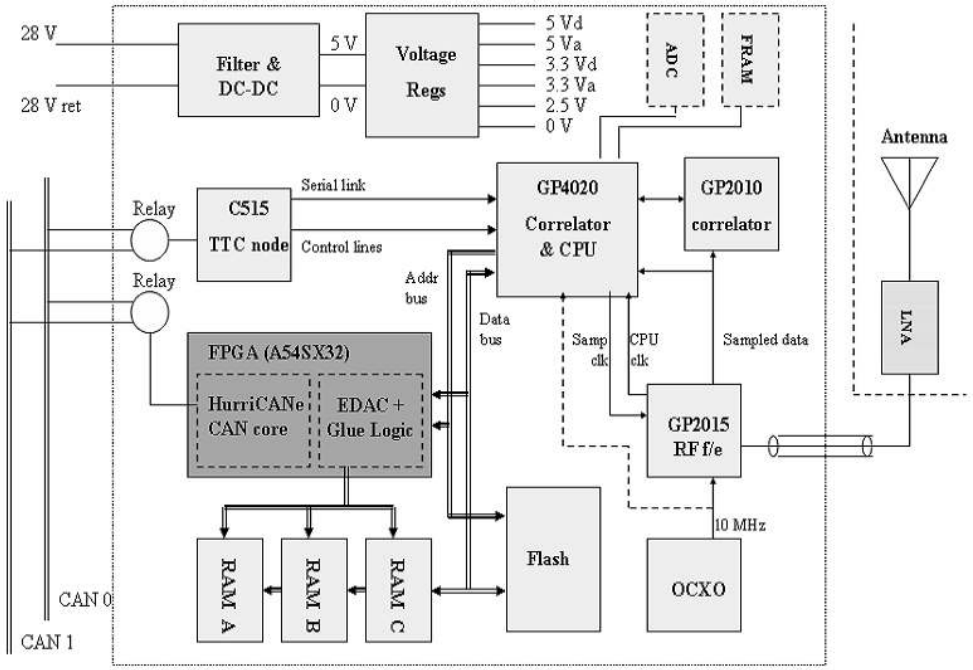

Fig 10 – Integrated EDACAN SGR-GEO Design

Summary

The design demonstrated, using the SSTL's engineering approach, how it is possible to mitigate radiation hazards and integrate a GPS receiver trough the usage of COTS devices. This soft-core design also allows for a flexible solution that can be adapted to any specific application. In this case it was specially tailored to better suit SSTL's CAN-SU protocol. This solution also provides a path for radiation hardened system simply by implementing the core in a RadHard FGPA.

References

[1] SSTL Engineering Approach Presentation, EDPS, Sir Martin Sweeting, 2003

[2] Bosch CAN 2 specification manual V 2.0, Robert Bosch GmbH, 1991

[3] HurriCANe - VHDL CAN Controller core Version 4.6, Luca Stagnaro, ESA, 2