Application and experience of CAN as a low cost OBDH bus system

Adrian Woodroffe and Paul Madle

Surrey Satellite Technologies Ltd, University of Surrey, Guildford, GU2 7XH, UK.
Email: a.woodroffe@sstl.co.uk p.madle@sstl.co.uk
Phone: +44 1483 689278
Fax: +44 1483 689503

1.0 Introduction

This paper gives an overview of SSTLs use of CAN bus on its recent missions. It gives a description of the CAN standard and the SSTL CAN topology. It also goes into depth concerning software protocols and chipsets (including COTS) used in order to implement CAN in space.

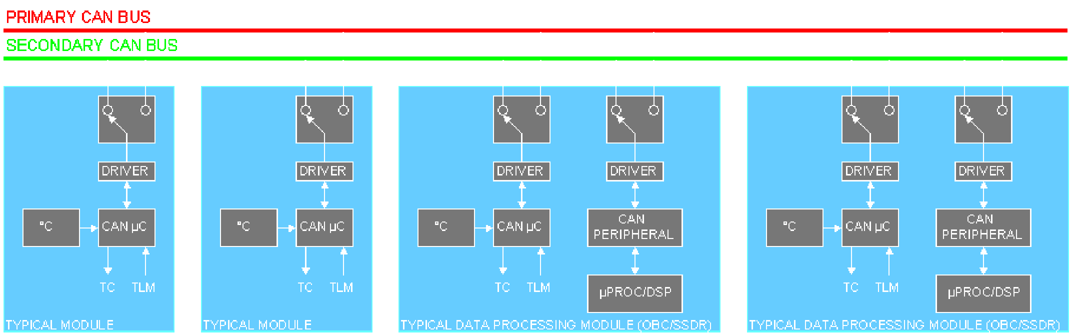
2.0 CAN Overview

The Controller Area Network (CAN) is a bus system used for the communication of discrete packets of information across a distributed electronic system. It is ideally suited for real time commanding.

CAN was spawned from the automotive industry in the late 80's to handle the increasing connectivity requirements of in-car electronic systems. CAN was originally developed by Bosch and is now formalised in ISO-11898 Parts 1 and 2.
CAN features include prioritization of messages, configuration flexibility, multicast reception with time synchronization, system wide data consistency, multimaster and automatic retransmission of corrupted messages as soon as the bus is idle again.

3.0 SSTL CAN implementation

SSTL has been using CAN as an on-board telemetry/telecommand bus since 1995. Initially CAN was used with a centralised TTC system (similar to TTC-B-01 HLC,BLC) on FASAT-Alpha/Bravo. SSTL then migrated to a complete CAN solution. The complete CAN system has now flown on 6 LEO missions (UoSAT-12, SNAP-1, AlSAT-1, UKDMC, NigeriaSAT-1 & BilSAT-1). The following CAN topology has been tried and tested:



Nodes within the spacecraft connect to both primary and redundant busses via a latching relay. On power up, the relay switches to communicate on the primary bus. If a node does not receive a CAN message for 5 minutes, it assumes bus failure and switches to the redundant bus. Modules connected to the bus fall into 2 categories. A data processing module such as an On Board Computer (OBC) has 2 connections to the bus. One is through an 8-bit CAN microcontroller which provides telemetry when the main processor is off. The other is via a CAN peripheral connected to the main processor. Alternatively, less intelligent modules are connected to the bus via the 8-bit microcontroller which performs all the control/telemetry gathering needed.

SSTL has developed its own relatively simple higher layer protocol on top of CAN. CAN for spacecraft usage (CAN-SU) forces peer to peer addressing and is optimised for telemetry, telecommand and buffer transfer.

4.0 COTS CAN Solution

SSTL specialise in providing low cost, predominantly LEO missions to a variety of civil and military markets. It has been the ongoing philosophy to fly exclusively commercial off the shelf technology (COTS) while mitigating the risk of radiation effects by module level redundancy and passive fail-safe system design. The use of COTS components allows SSTL to rapidly utilise new technology as it enters to commercial market. For example, SSTL first flew CAN bus technology in 1996 and now uses CAN as the backbone for all missions. See table 4.1 for components flown:

Components flown	Number of missions flown
Philips CAN Tranceiver: Physical CAN driver, current production	4
Phillips PCA82C250: Physical CAN driver, going obsolete	10
Philips P87C592: CAN microcontroller, obsolete	10
Philips CAN 8-bit peripheral, current production	4
Philips PCA82C200: CAN 8-bit peripheral, obsolete	10
Infineon: 8-bit CAN microcontroller (A/D, PWM etc. 8051), current production	6
Microchip CAN SPI peripheral	4

Table 4.1 – COTS CAN components used on recent missions

5.0 Radiation Tolerant CAN Solution (RadCAN)

Recently SSTL has started moving out of LEO missions under the British National Space Centre funded GEMINI small GEO platform program. A COTS approach is no longer applicable. In order to meet the radiation environment and reliability (as GEMINI has deployable panels and therefore no passive fail-safe attitude) a more traditional space industry solution was used whist retaining our expertise in CAN based system architectures.

The single chip CAN microcontroller functionality of the COTS 8051 was re-created with a discrete microcontroller, memory, EPROM, ADC and FPGA. This provides a Latch-up immune, highly SEU tolerant, 100Krad solution. A comparison of the cost of migrating from COTS to Hi-Rel is shown in Table 5.1.

	COTS Infineon 8051	RadCAN
Size	1"sq	>6"sq
Mass	<5g	>50g
Power	<0.75W (max), 0.20W (nominal)	Estimate 1W
Total Dose	<10Krad	100Krads
SEE Protection	None	Highly SEU tolerant, latch-up immune
Component Cost	<$10	>$15000
Export Issues	None	Full ITAR

Table 5.1 – Comparison of a Radiation Tolerant to COTs CAN solution

6.0 Future Work

Although compared with MIL-STD-1553, CAN is a low power solution. Power savings are always welcome. SSTL are therefore looking at low power CAN solutions using 3.3volt technology. Work is presently under way to develop a 3.3volt CAN node using a PIC 8-bit microcontroller, replacing the 8051. Initial tests show the PIC consuming a third of the power of the 8051 running at approx 3.5 MIPS as opposed to the 8051's 1 MIP.

As SSTL moves out of purely LEO missions, the need for more robust, radiation tolerant systems increases. The next generation of RadCAN will be a System on a chip solution (SoC) integrating VHDL IP cores for a micro-controller, CAN core, memory and EDAC in a single FPGA. (Currently looking at the Actel RT version of the AX1000). For applications not requiring microcontroller functionality, RadCAN Lite is being developed as just the CAN core with a simple state-machine.

7.0 Conclusion

The use of the CAN bus on SSTL spacecraft has been highly successful and despite the use of COTS technology, no CAN node failures have been observed. CAN is an ideal architecture for LEO missions and with the developments outlined above, will be an attractive alternative to more traditional space bus architecture for all missions.

8.0 References

1. http://www.caen.it/micro/rd_casta.html: a Rad-Hard 8051 microcontroller with on chip CAN hardware.
2. http://www.can.bosch.com/: bosch's CAN page
3. http://www.can.bosch.com/docu/can2spec.pdf : the Bosch CAN 2.0 B specification
4. http://www.can-cia.org/: CAN in automation
5. http://www.vector-informatik.com/canlist/: an informative CAN mailing list