Implementation of Fault Tolerant Network Management System for CAN Bus using CANopen

Vardhaman Tiwatane, Deelip Omana

- Department of Advanced Embedded System Design, International Institute of Information Technology, Pune (India) -

Embedded systems are now-a-days complex distributed systems (e.g. automobiles, elevators, building climate control systems, and aircrafts) with various demands on networking capabilities. These Networked Control Systems are Distributed Control Systems (DCS) where controllers, sensors, actuators and other system components communicate over a network. It is easy to imagine that according to the high complexity of the networked electronic systems, the risk of failures of the system is increasing. So there is a need to provide high levels of reliability within the network. To accomplish this task; a network management system is used.

CANopen protocol is a networking system based on CAN serial bus. It has been widely applied into networked control systems. The CANopen networking functionality is implemented in one of the CAN nodes. This node is responsible for conducting networking functionality such as mode control, error control and configuration control. The network is vulnerable if this node fails. This paper presents a fault tolerant system, which deals with failure of so called Network Manager and sorting out this situation by providing a redundant node. The redundant node is provided with optional functionality of Network Management along with its original functionality of Slave Node.

Keywords **— CAN, CANopen, fault tolerant system, network management, NMT master/slave, OSI model, and redundant master.**

I. Introduction

Embedded system has grown from stand-alone systems to highly integrated and networked systems [1]. Embedded networks are essential for the unified, efficient and cost-effective exchange of electronic information within embedded communication networks. Each system should communicate with the other systems in the network, sharing information, sending and responding to requests as needed [2]. The embedded communication network employs different microcontrollers/microprocessors devices. These devices are connected together based on strict rules, which determine exactly how network units shall communicate.

After tremendous growth of inter-networked embedded systems, a new wave of technological achievement in the form of ubiquitous communication can now be observed. Thus various demands on the communication have led to the development of many different network protocols [3].

Fault tolerant systems are those systems, which remain operational in the presence of faults; these faults should not propagate and affect the rest of the system [4]. This paper deals with the fault tolerant network management system for CAN bus. As per the guidelines given in the CANopen [5], one of the nodes on the bus can be configured as network master. This node carries out the network management for whole of the CAN bus. In case of failure of this node the network turns into an unmanageable state. This paper elaborates the technique in which a redundant node on the existing CAN bus, detects the failure state and take over the control of the network.

II. CAN and CANopen

Controller Area Network (CAN) [6], originally designed for the automotive industry, but now has also become popular in industrial automation as well as many other applications [7]. It is a serial bus communication protocol developed by Bosch, in the early 1980s, as a method for enabling robust serial communication [1] [8]. A CAN bus with three nodes is shown in Figure. 1.

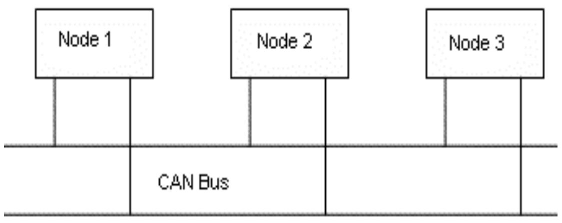

Figure 1 Three nodes connected on CAN bus

CAN have been developed as an international standard and is documented in ISO 11898 (for applications up to 1 Mbps) and ISO 11519 (for applications up to 125 Kbps) [9]. Since the CAN specifications defines the data link layer in the ISO Open Systems Interconnect (OSI) model, higher layer protocol (HLP) specifications outline the application layer. CANopen is one such HLP.

CANopen [5] is standardized embedded network protocol based on the CAN serial bus with high flexible configuration capabilities. It was originally designed for industrial automation distributed applications. CANopen networks are also used in off-road vehicles, maritime electronics, medical equipment, and railways. CANopen is now owned and maintained by CAN in Automation (CiA), the International Users and Manufacturers Group". CANopen uses the international CAN standard, ISO 11898 as the basis for communication. The seven-layer structure defined in the ISO/OSI reference model can be mapped to the CANopen protocol stack, as illustrated in Figure 2[10] [11].

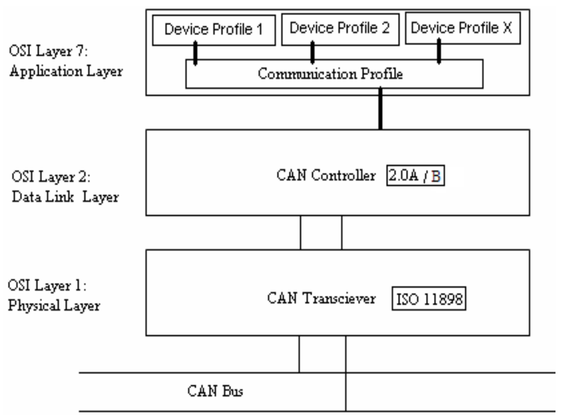

Figure 2. CAN and CANopen standards in the OSI network model [10] [11].

The CANopen standard consists of an addressing scheme, several small communication protocols and an application layer defined by a device profile. The fundamental part of the device profile is the object dictionary [11]. This consists of a mixture of data objects, communication objects and commands or actions. CANopen services give the user full access to the object dictionary in a device allowing reading and writing of data and commands. Data and commands are implemented using a 16-bit index addressing mechanism together with 8-bit sub-index.

The communication protocols have support for network management, device monitoring and communication between nodes, including a simple transport layer for message segmentation/de-segmentation. CAN data link layer is normally implemented in hardware. The basic CANopen device and communication profiles are given in the CAN in Automation draft standard 301[12]. The lower level protocol implementing the data link and physical layer is usually CAN [13].

III. Network management usung CANopen

Generally, the network management protocols reside in the Network layer of the OSI (Open Systems Interconnection) Reference Model for communication.

Network management in a CANopen network is based on the NMT and DBT services defined in the CAN application layer (CAL). It is one of the service elements of the application layer in the CAN Reference Model [14]. It is used to control devices on the network in a co-ordinated and well-defined manner. NMT master performs network management using different network protocols such as Module control protocols, Error control protocols, and Configuration control protocols.

The NMT serves to configure, initialize, and handle errors in a CAN network [12]. CANopen network management provides following functionality:

- Dynamic or static distribution of CAN identifiers for communication and error services.
- Control over node operating states and communication modes for individual or multiple nodes.
- Life-guarding [15].

The CANopen network management is node-oriented and follows a master/slave structure. One node in a CANopen network is assigned the role of the network (NMT) master. The other nodes are NMT slaves. Each NMT Slave is uniquely identified by its module ID (i.e. node ID). The master does not arbitrate the other nodes' access to the bus in any way: it simply acts to co-ordinate network behavior. Each NMT slaves in the network has a node object which is uniquely identified in the network by its NMT address. For each NMT slaves there exists one remote node object with the same NMT address (as of the NMT slave) on the NMT master. It is possible that the module acting as an NMT master can also be configured as an NMT slave.

NMT master controls the communication state of the other nodes (NMT slaves) in a CANopen network, e.g. change the state of all nodes in a defined way from the "Pre-operational" state (in this state no transmission of process data is yet possible, only the configuration and parameterization of nodes) to the "Operational" state (in this state the exchange of process data is also possible). In this way the state of the individual network node can be controlled in a well-defined way. A NMT Slave represents that part of a node, which is responsible for the node's NMT functionality [5] [15].

CANopen specifications [12] also define the node guarding protocol that enables the network manager to determine whether a node is still alive on the network and also to determine the nodes operating state. Figure 3 shows the NMT node-guarding format [11].

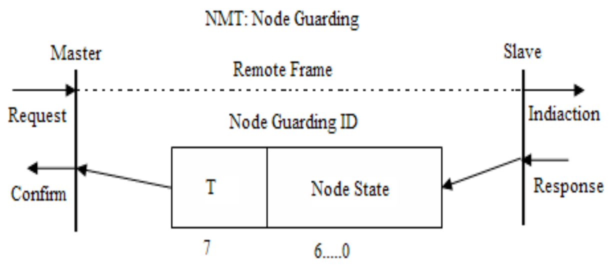

Figure 3. Node guarding format [11].

CANopen is a standardized application defined on top of CAN and widely used in Europe for the application of CAN in distributed industrial automation. It provides users a platform for device profiling and network management. The preceding section gives an overview of fault tolerant systems that use Node-Monitoring/Guarding service to detect node failure in the network [16].

IV. NMT model

CAN network consists of modules that are connected by one physical CAN bus. All the nodes are configured with networking features. These features are characterized as NMT master feature and NMT slave features. A module having NMT master feature, is a network manager on the CAN bus. It carries out various tasks required to monitor the whole network. A module having NMT slave features is a module able to respond properly to the queries of network master.

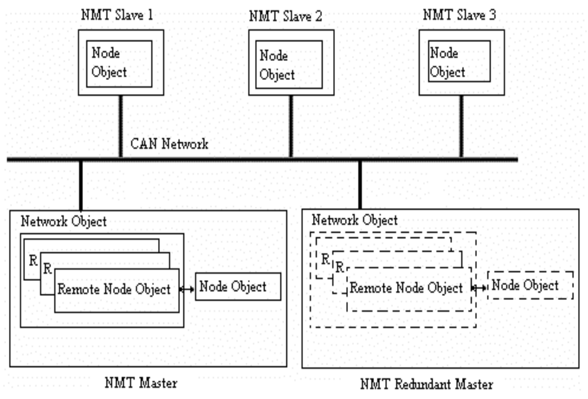

Figure 4. NMT Model.

The CAN module uses three objects to model a CAN network. These objects and its existence are depicted in Figure 4.

- *Network object* represents set of all modules in a CAN network. A network can contain at most 255 modules. This object exists in a NMT master.
- *Remote node object* exists on the NMT master. A remote node object represents each module in the network that is managed by the NMT services.
- Node object exists in a NMT slave. Each module that is managed by the NMT services is represented by a node object on that module [17].

Figure 4 shows a Network Management Model where NMT master polls each NMT slave at regular time interval (guarding time). The response of the NMT slave contains the state of the node object of that NMT slave. If this state doesn't match the state of the corresponding remote node object, it will be indicated that a remote error has occurred through the network event services.

V. Fault tolerant network management system for CAN

CANopen network management follows master/slave structure. NMT master acts as a network manager. NMT services uses a node guarding protocol to detect the node failures in the network. If the node within a specified time window doesn't respond the NMT master signals a node guarding event to its application. This is the case if one of the nodes is failed in the network. But suppose if the master node fails then the entire network will not be operational and will come to stand-still, which is undesirable for the some safety critical applications.

In Fault Tolerant Network Management System for CAN (FTNMSCAN), within the existing network one of the slave nodes (temperature sensor) is assigned an additional functionality of a redundant master node. If master node fails, then the redundant master will take the control of the bus and will act as NMT master. In normal operating condition the redundant node executes its assigned task. In the failure mode this redundant node has an additional functionality as network master.

VI. Implementation details

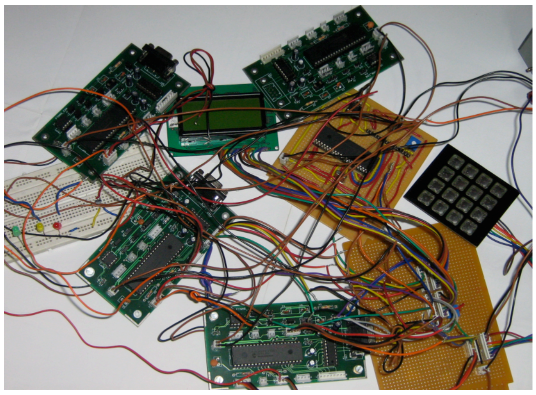

Figure 5 FTNMSCAN setup

Figure 5 shows setup for FTNMSCAN. Five boards have been designed using PIC18F458 microcontroller, which has on-chip CAN controller. The transceiver used is the MCP2551.

- Three slave nodes (namely, LCD, Temperature Sensor and Keypad).
- One master node.
- One Troubling node.

Of the three-slave nodes temperature sensor node has additional functionality as redundant NMT master. Network operational details are categorized into two modes *Normal mode* and *Failure mode*

A. Normal Mode

Initially after power up, when the nodes are in pre-operational state, respective LEDs blink at PORTE.0 implies that the network is in pre-operational state.

NMT Master:

In addition to controlling the state machines of the slaves on the network; NMT master provides services for monitoring devices on the network through node guarding protocol. When LED is 'ON' at PORTE.1 implies that NMT master is operational. If this node fails, LED at PORTE.1 is 'OFF'

NMT Slaves:

Each NMT slave and its node object is uniquely identified in the network by its NMT address. For each NMT slave, there must exist a remote node object with the same NMT address on the NMT master. A node object and remote node object that have the same NMT address are called peers. Each remote node object communicates with its peer via the NMT protocol. The NMT protocol uses a node ID to address a NMT slave.

1. Temperature Sensor Node:

When this node enters into preoperational state respective LED is' ON' at PORTE.0. Sensor node transmits the sensed temperature sample periodically on the bus.

2. LCD Node:

When this node enters into preoperational state respective LED is 'ON' at PORTE.0. This node receives data from temperature sensor node and displays temperature in degree centigrade or degree Fahrenheit depending upon what key is pressed by the user at keypad node. The default mode for display is degree centigrade.

3. Keypad Node:

When this node enters into preoperational state respective LED is 'ON' at PORTE.0. Two keys are used in this application. If any of the keys is pressed an appropriate packet is transmitted by the node on the bus. LCD node receives this packet and reconfigures LCD display based on the received packet information (i.e. whether to display temperature in degree centigrade or Fahrenheit).

B. Failure Mode

When the device becomes operational, after 2 minutes of time the Troubling node is activated. The function of the troubling node is that whenever NMT master polls for Node Guarding it generates an error message on the bus it continues doing this until the NMT master goes into bus off state. To detect the absence of NMT master, the NMT redundant master tests internally, whether the node guarding is taking place in the defined time interval (life guarding time). If yes, it continues observing the activities being executed by NMT master and updates the remote node objects. If no, it takes over the control and start polling individual NMT slaves with its own ID. When redundant master is in action, the respective LED at PORTE.1 is "ON' indicating that redundant master has taken over the bus.

VII. Future enhancements and conclusions

CANopen is an application layer protocol based on CAN network. CANopen enables the communication between devices of different manufacturers and guarantees an interchangeability of devices. Hot plugging feature which offers dynamic distribution of CAN identifiers can be implemented by using Distributor (DBT) service of CANopen. This feature is important for design of CAN nodes based on the customer's requirements. Fault tolerant Network Management system uses the service provided by network management to enable smooth and efficient operation of the complex networks. One can modify the existing network to his own need.

Acknowledgment

The authors would like to acknowledge the anonymous reviewers for their constructive criticism. The authors also wish to acknowledge and thank Director, Deputy Director, Head of the Department - Embedded Systems and all faculty members of I^{2}IT, Pune.

Vardhaman P. Tiwatane
Institute: International Institute of Information Technology, Pune (India)
Address: P-14, Rajiv Gandhi Infotech Park, Hinjewadi, Pune - 411057, India
Phone: 0091-20-22933441
Fax:: 0091-20-22934191
E-mail: vardhamant@isquareit.ac.in

Deelip Babaso Omana
Institute: International Institute of Information Technology, Pune (India)
Address: P-14, Rajiv Gandhi Infotech Park, Hinjewadi, Pune - 411057,India
Phone: +91 9922688647
E-mail:dbomana@gmail.com,
deelipo_june06@isquareit.ac.in

References

[1] J. Froberg, K. Sandstrom, C. Norstrom, H. Hansson, J. Axelsson, and B. Villing.A comparative case study of distributed network architectures for different automotive applications. In *Handbook on Information Technology in Industrial Automation*. IEEE Press and CRC Press, 2004.

[2] ZOU Lian-ying, ZOU Xue-cheng. Co-design for an SoC embedded network controller, *Journal of Zhejiang University SCIENCE A* ISSN 1009-3095 (Print); ISSN 1862-1775. [Online] Available: http://www.springerlink.com/index/540H1870 2581780T.pdf

[3] Andrew S. Tanenbaum,Computer Networks, 4/E, Prentice Hall,2003.

[4] Philip P. Shirvani and Edward J. McCluskey, Fault-Tolerant Systems in A Space Environment: The CRC ARGOS Project. CRC Technical Report No. 98-2 (CSL TR No. 98-774) December 1998 [Online] Available: http://www-crc.stanford.edu/crc_papers/CRC-TR-98-2.pdf

[5] CANopen, an overview [Online]. Available: http://www.can-cia.org/canopen/

[6] CAN Specifications Version 2.0 [Online]. Available: http://www.semiconductors.boschs.de/pdf/can2spec.pdf

[7] Bruce Negley, Getting Control Through CAN, Oct 1, 2000, [Online] Available: http://www.sensorsmag.com/articles/1000/18/main.shtml

[8] Dominique Paret, Roderick, MA Riesco (Translator), Multiplexed Networks for Embedded Systems CAN, LIN, Flexray, Safe-by-Wire..., John Wiley & Son's, Ltd.

[9] Keith Pazul, Controller Area Network (CAN) Basics. AN713, [Online] Available: http://www.cl.cam.ac.uk/research/srg/HAN/Lambda/webdocs/an713.pdf

[10] H. Boterenbrood, CANopen high-level protocol for CAN-bus. [Online]. Available: http://www.nikhef.nl/pub/departments/ct/po/doc/CANopen30.pdf

[11] M Farsi, K Ratcliff and Manuel Barbosa, An Introduction to CANopen, Mechatronics and Automation, Proceedings of the 2006 IEEE International Conference.

[12] CAN in Automation, CANopen, Application Layer and Communication Profile, CiA Draft Standard 301, Version 4.02. 13 February 2002.

[13] O. Pfeiffer, A. Ayre, C. Keydel: *Embedded Networking with CAN* and CANopen, RTC Books, ISBN 0-929392-78-7, 2003.

[14] Mohammad Farsi and Karl Ratcliff, An Introduction To Canopen And Canopen Communication Issues, CANopen Implementation (Digest No. 1997/384), IEE Colloquium.

[15] CAN in Automation (CiA). [Online]. Available: http://www.diakom.ru/el/communication/can/can_org.pdf

[16] K.H. Johansson, M. Torngren, and L. Nielson, "Vehicle Applications of Controller Area Network" [Online]. Available: http://www.s3.kth.se/~kallej/papers/can_necs_handbook05.pdf

[17] CAN in Automation, CANopen, CAN Application Layer for Industrial applications, CiA Draft Standard 201 to 207 version 1.1