DESIGN AND DEVELOPMENT OF FAULT TOLERANT CAN-LIN GATEWAY FOR IN-VEHICLE COMMUNICATION

Nimish Kumar1, Rinki Sharma2, B. Ramya 3
1- (Engg.) Student, 2- Assistant Professor, 3- Senior Lecturer
Automotive Engineering Centre
M. S. Ramaiah School of Advanced Studies, Bangalore

Abstract

In last few decades, the rising numbers of sensors, actuators and ECUs have increased the complexity of automotive network. Moreover, multiple network communication protocols such as CAN, LIN, FlexRay, MOST are used in today’s cars to meet the various requirements of in-vehicle applications. While CAN is the traditional automotive field-bus used in engine management, LIN provides a low-cost network solution for car window and door applications. In many instances data has to be shared between various networks using different protocols. Network gateway(GW) enables this data exchange with optimum utilization of the available data / signals. Typically over 25-45 ECUs connected in a vehicle with two to four different network protocols exchange approximately 100-300 signals and data [1],[2]. An error free data transfer from one protocol to another through GW ECU is a challenging and critical task. Failure or error in data exchange can lead to safety problems or inconvenience to the passenger.

GW failure may happen due to failure in software algorithm or hardware. In this paper, a fault tolerant GW proposed to tackle failure in hardware. Design configuration depends on number of controllers, transceivers used, criticality of subsystem and software algorithm implemented. Basic configurations are proposed highlighting the advantages and disadvantages.

Out of five basic configurations proposed, penta configuration fault tolerant CAN-LIN GW design is developed with limited features using PIC18F4580, MCP2551 and TPIC1021 and its feasibility tested under lab environment. Each controller is assumed to support one protocol path. Software control algorithm is developed for CAN-LIN GW. Experimental and simulation results are presented and discussed.

Key Words: Network, Fault Tolerant Gateway, CAN, LIN, Automotive Communication

Abbreviations

CAD Computer Aided Design
CFD Computational Fluid Dynamics
EGR Exhaust Gas Recirculation
MPI Multi-Point Injection
PIV Particle Image Velocimetry
PISO Pressure-Implicit with Splitting of Operators
VVT Variable Valve Timing
WOT Wide Open Throttle
CAN Controller Area Network
ECU Electronic Control Unit
FT Fault Tolerant
GPRS General Packet Radio Service
GW Gateway
LCD Liquid Crystal Display
LIN Local Interconnect Area
MOST Media-Oriented System Transport
NM Network Manager
OEM Original Equipment Manufacture
OSEK OS Offene Systeme and deren Schnittstellen für die Elektronik in Kraftfahrzeugen; In English: Open Systems and their Interfaces for the Electronics in Motor Vehicles)
PDU Protocol Data Unit
USART Universal Synchronous/Asynchronous and Receiver/Transmitter
UART Universal Asynchronous Receiver and Transmitter

1. INTRODUCTION

In early days of networking, generic UART was used by GM, Ford and Chrysler. These big three were vertically integrated and not highly dependent on external suppliers. However in Europe, car manufactures were dependent highly on external suppliers. The same trend is increasing now in US. Use of proprietary protocol stated posing difficulties in system design. Standard protocols allow modules from different suppliers to easily link together. In 1994 to classify the various standards, SAE has defined basic categories such as Class A(10kbps), B(10-125 kbps), C (125 to 1 Mbps), and D(>1Mbps) based on network speed and function. Door information which exists in Class A (LIN) must be able to communicate with CLASS C (CAN, power-train module) networks. This led to development of communication network gateway (GW). A GW is a device, which translates and transfers data from one protocol to another. GWs are required to connect different domains of vehicle such as body, power-train, chassis and/or telematics functions.

Many researchers have worked towards the development of GW implementation for in-vehicle protocols. They have proposed various software and hardware for improvement of features, increase flexibility and robustness of GW. Almost every GWs has its own merits and demerits.

Vivekanandan et. al [3] emphasizes the importance of configurable network in vehicle where CAN is the backbone network and LIN as sub network connected...
through GW. Evolution of networks has mainly three advantages. First, reduce the bulk of wiring harness. Second, sensor data is available to any ECUs and third reduce use of number of sensors which reduces overall cost, increases flexibility and further weight. Requirement of such a network would be a diagnostic manager, network layer, signal interaction layer, network management, data link layer, CAN/LIN drivers and bootloader. Configurability and scalability helps reducing development time and cost for OEMs.

Hyan Seo, et.al[4] explains a flexible network GW based on proven communication controllers between the CAN and FlexRay protocols for ECU embedded systems. Flexibility is brought by using independent control unit, which decreases the work load from the main ECU. The control unit proposed has high system frequency and program memory to handle complex routine information in real time and ensure data integrity due to dedicated ECU. The control unit reduces the interrupt burden of the main ECU, while message buffer RAM is used to store all messages transmitted and received.

Khanapurkar, et. al [5] have proposed a FPGA design of vehicle black box for storing and retrieving data of various ECUs on automotive CAN network, which increases network diagnostic capability.

Yong, et. al [6] describes improved design of CAN-GPRS (general packet radio service) with high end 32 bit microprocessor to bridge the in-vehicle CAN bus network and mobile communication network to remote hosts.

Zhu and Jackman [7] proposed Simulink based SimEvent tool for simulation environment and demonstrated for the two protocols GW.

Yoon Moon et al., [8] proposed GW ECU architecture using three protocols LIN-CAN-FlexRay GW with diagnostic functionalities. The system consists of MC9S12DP256 (Host), MFR4200 (FlexRay controller), LIN transceiver, CAN transceiver, and FlexRay transceiver.

Horner [9] proposed a universal GW (LIN, CAN & FlexRay) based on flexible post build configuration with AUTOSAR standard Configuration Conformance Class. New requirements can be added to the already built GW without much change in the application code.

Seo et al., [10] describes reliable GW mechanism using OSEK OS, NM and log recorder for CAN, LIN and FlexRay. Developed GW message are processed and sent based on the priority assigned for full-preempt task and hence increases reliability of software architecture. Diagnostic features were implemented by keeping log-information. Authors have proposed addition of fault tolerance capability.

From literature survey requirement for a reliable and fault tolerant (FT) GW is found to be important and continuous research is going on to achieve the same. A block diagram of currently available GW and proposed fault tolerant gateway is shown in Figure 1. Conventional system cannot handle protocol hardware failure (Tx/Rx). Proposed fault tolerant GW has dual path to transmit message on to destination network bus. Main ECU will monitor health of the Tx/Rx hardware.

In case of failure of primary network, secondary path will be used to transmit the data. The health monitoring data will be recorded and display error or path failure in a LCD.

**Fig. 1 Conventional vs. FT GW**

### 2. DESIGN CONFIGURATION

GWs are prone to error due to failure in software algorithm or hardware. A FT GW should work even in case of software or hardware failure. Design configuration of FT GW depends on number of controllers and transceivers per controller used and number of paths and software algorithm implemented. Basic configurations proposed are shown in Figure 2.

1. **Single Controller GW:** Such GW module will have one ECU with two or more protocols transceiver. Control, design and diagnostic features implementation are easy. Limitation: GW fails in case of hardware failure of ECU.

2. **Dual Controller GW:** Such design of GW will have two ECUs with single transceiver of each protocol. Each ECU will act as GW. In case of primary ECU failure, GW control will be transferred to secondary ECU. Control and design will be easy. Diagnosis features have to be supported by both controllers, which will increase the complexity.

3. **Tri Controller GW:** Tri GW module consists of three ECUs where each ECU will have one transceiver for each protocols. Three level of FT redundancy is provided by such design, since data can flow from three different ECU and paths. Separate software algorithm can be to bring redundancy in GW design. Design for such system is complex and cost increases with increase in number of ECU.

4. **Quad Controller GW:** Such GW is designed with four ECUs each having single protocol support. Quad GW will provide minimum of five alternative paths for data flow. Also, software and hardware redundancy in the system will increase the fault tolerance capability. Design and implementation of diagnostic features will be complex.

**Fig. 2 Basic design configuration of FT GW**
5. Penta Controller GW: Such a design will provide two network paths. Four ECUs with single protocol transceiver are connected with main ECUs. Separate software algorithm can be implemented for each protocol ECU. Design and implementation of diagnostic features will be easy since it will be controlled by main ECU. Increase in number of controller will increase cost. Depending on the application and redundancy level required, other configurations can be designed from above basic configurations compromising over limitation, benefits and cost of each other. Redundancy in software algorithm can be used which will further enhance reliability and fault tolerance capability.

Feasibility study of such a design is shown in this work. CAN- LIN protocols, which are commonly used in mid-range vehicle, have been selected and implemented in penta controller GW design for the feasibility study of GW. Similar software algorithm is used in both CAN and LIN controllers to reduce complexity.

3. PENTA CONTROLLER DESIGN

Block diagram of penta controller GW is shown in Figure 3. It has following important components.

- Main Microcontroller - Connected with LIN-1, LIN-2 slave node, CAN-1 and CAN-2 nodes.
- LIN-1 and 2 – Two paths for transferring CAN message to LIN bus and vice-versa.
- CAN-1 and 2 - Two paths for transferring LIN message to CAN bus and vice-versa.
- LIN-M - LIN Master on LIN bus is used for testing the functionality of the GW.
- CAN-N – CAN Node connected to CAN bus for testing the functionality of the GW.

Three main components are main controller, CAN 1 & 2 controllers and LIN 1 & 2 controllers, which controls functionality of the system. Main controller communicates with LIN and CAN controller through software UART that is used for serial bus communication and can be developed using any two controller ports. In this study software UART build-in library function of MPLAB IDE C18 compiler is used. Four hard-coded software UART files has been implemented for two CAN and LIN controllers flow path. CAN and LIN controllers are connected to respective transceivers through predefined ports in the controllers. These controllers monitor the health of the bus and send or receive messages from the bus to main controller and vice-versa.

A top-level flowchart implemented in main controller showing algorithm for message flow and selection of GW control path is shown in Figure 5. Main controller continuously checks for the message from attached protocol bus and as well as the health of the primary paths. In case message is received from CAN, it extracts the data and sends the same to the LIN primary path. In case the primary path is unhealthy (due to hardware failure) the data will be sent to secondary path maintaining the robustness of the GW. Similarly, the data received from LIN will be handled. Time taken to transfer the control from primary to secondary path depends on number of instruction cycles used in the application and clock frequency. Current application uses 105 instruction cycles with 1 MHz crystals used in main controller. Hence, time taken is 105 μs.
3.1 Software Architecture

Software architecture is shown in the Figure 4. CAN 2.0 [11] and LIN 1.2 [12] protocol specification has been followed. “MCmain.c” controls both CAN and LIN (slave) controller’s primary and secondary data flow path. It also checks the health of the software UART path by sending and receiving expected data at regular interval. Respective algorithm for monitoring health of the CAN and LIN bus is developed in “can.c” and “lin.c”. “Node.c” and “lin_master.c” contains input/output functionality used for the testing the GW. “Lcd.c” has output function for display unit.

The software design for CAN node in proposed FT GW have similar control algorithm in both primary and secondary controllers for simplification. Separate control algorithm should be used to bring software redundancy. High-level CAN node control algorithm is shown in Figure 6. CAN module perform two tasks. First one is to receive message from main controller and send or reply appropriately to CAN bus or main controller and vice versa. Other is to check health of the CAN bus. In PIC18F4880 control register TxB0 [14], is set high, if CAN bus gets continuously more than 255 errors. Time taken for each error counts depends on the hardware used.

Similar algorithm has been implemented for LIN. LIN protocol is based on single master and multiple slave concept. High-level control algorithm for LIN slave used in GW is shown in Figure 7. In this setup, LIN slave (LIN1 & LIN2) is used in design, which will avoid user interface for the GW. LIN master is configured to test the GW design. Designer needs to program the LIN master for these two slaves with LIN slave GW ID=07 (currently used in the program). LIN slaves perform similar task as CAN nodes. LIN bus hardware failure is dependent on timer count. One timer count is equal to \((2^{16}-1)\) instruction cycle. After each 25 timer count LIN master sends health check message to LIN slave. In current application, if LIN slave does not receive health check message from LIN master up to 100 timer counts, it raises a flag indicating failure in LIN bus path. Time taken to identify the LIN bus failure in current application is approximately 1 sec with 12 MHz crystal. It is again dependent on application code optimization and hardware used.

3.2 Hardware Selection

Hardware’s used for implementation penta controller FT gate is tabulated in Table 1. Basic requirement of the controller is CAN Tx/Rx port for interfacing CAN transceiver and USART/Enhanced-USART for interfacing LIN transceiver. Establishing communication between main controller and other ECUs, software UART requires 5 ports for each controller. One LCD display (8 pins) is connected with main controllers for displaying errors and paths used by the GW. Port configuration for the main controller is shown in Figure 8. From CAN1 and CAN2 two Tx/Rx ports are connected to CAN transceivers. E-USART ports from LIN1/LIN2 controllers are connected to LIN transceivers. Other pins in main micro-controller unit are used for LED’s and switches.
Fig. 7 High-level control algorithm in LIN slave

Table 1. Hardware description

<table>
<thead>
<tr>
<th>Module</th>
<th>Hardware</th>
<th>Voltage</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Controller</td>
<td>PIC18F45 80</td>
<td>5 V</td>
<td>40 pins required for connecting 4 microcontroller</td>
</tr>
<tr>
<td>Can module</td>
<td>PIC18F45 80/ MCP2551</td>
<td>5 V</td>
<td>CAN interface/CAN transceiver</td>
</tr>
<tr>
<td>LIN module</td>
<td>PIC18F45 80/ TPIC1021</td>
<td>5/12 V</td>
<td>EUART/UART and LIN transceiver</td>
</tr>
<tr>
<td>LCD module</td>
<td>LCD</td>
<td>5 V</td>
<td>2 line display</td>
</tr>
</tbody>
</table>

Fig. 8 Main Micro-controller hardware connection

4. TESTING AND VALIDATION

Simulation of each CAN node has been performed in Vector CANoe tool. Block diagram for the CANoe tool is shown in Figure 9. Test results are shown in Figure 10. Different messages were sent and received from CANoe successfully.

Functional test has been performed on developed FT GW hardware setup under lab environment. Experimental setup is shown in Figure 11. Apart for the 5 controllers in the GW, one controller each for LIN (Master unit) and CAN is used in respective bus. A display unit as output and two switches as input has been connected and programmed. Two switches are used for sending two different messages. Messages received and sent are displayed in the corresponding display unit. Another display unit attached to the main controller displays send or receive message and path used. In case of failure of path and error in bus, it displays the same in real time.

Hardware failures were demonstrated by removing the Tx/Rx pin from the bus and GW. GW has been tested for single and failure of both paths. Accurate data transfer is received from both the buses. In case of primary path failure, GW uses secondary path. If primary path is connected, it automatically detects and uses primary path.

Non-functional requirement such as memory usage and cost has been evaluated. 85% of the controller memory has been unutilized, which can be used for implementing new features. Cost of the setup is...
calculated to be not more than $25 (excluding design and development cost), where as CAN-LIN gate from COSMOL cost nearly $1200 [13].

Fig. 11 Experimental setup

5. CONCLUSION

Fault tolerant GW is a necessity for critical system application, which requires data from other network bus Basic functions of a FT GW are demonstrated in this work. GW proposed notifies real time GW failure error. Enhanced diagnostic features can be added.

6. REFERENCES


