International Journal of Research & Development Organisation

ISSN-4567-7860

JOURNAL OF ELECTRICAL AND ELECTRONINS ENGINEERING

ARDUINO BASED VEHICLE COLLISION DETECTION USING CAN PROTOCOL

S.Shanmathi
Dept of ECE, Bannari Amman Institute of Technology, Sathyamangalam-638401
e-mail- lunashanmathi@gmail.com

Prof.C.Kamalanathan
Dept of ECE, Bannari Amman Institute of Technology, Sathyamangalam-638401
e-mail- kamalanadhan@gmail.com

***Abstract*—Vehicle accidents are one of the leading causes of fatalities. Because of not getting help on time, poor emergency facilities, people lose their lives. This project deals with the optimum solution to this drawback. This paper presents the development and implementation of a digital driving system for a semi-autonomous vehicle to improve the driver-vehicle interface. It uses an ARDUINO based data acquisition system that uses ADC to bring all control data from analog to digital format and visualize through LCD. The communication module used in this project is embedded networking by CAN which has efficient data transfer. In order to reduce point to point wiring harness in vehicle automation, CAN is suggested as a means for data communication within the vehicle environment. The benefits of CAN bus based network over traditional point to point schemes will offer increased flexibility and expandability for future technology insertions.**

***Keywords*—Control Area Network (CAN), collision avoidance system**

I. INTRODUCTION

The problem of vehicle accident is part of an endless list of disasters that could occur anywhere anytime. According to the association for safe international road travel , about 1.24 million die and 50 million are injured on the roads of the world every year. So this system is proposed where mistakes done by driver are eliminated. Most of the intelligent car systems have monitoring system only. Antilock brakes, speed sensors and other automatic systems are present in sports cars and other luxury cars only. But these cars are not affordable to everyone. So, a system needs to be developed which can be implemented in every car. A collision avoidance system is a system of sensors that is placed within a car to warn its driver of any dangers that may lie ahead on the road. Some of the dangers that these sensors can pick up on include how close the car is to other cars surrounding it, how much its speed needs to be reduced while going around a curve, and how close the car is to going off the road.

Ultrasonic sensor is adapted to measure the distance with respect to the previous car. For rear-end end collision avoidance subsystem, the currently available ultrasonic sensors for vehicles are adopted for approaching cars with relatively low speed. While the rough reading of distance data cannot be applied directly, an intelligent approach is proposed to process the raw distance read out of sensors to produce appropriate warning signals. Also an alcoholic sensor is included in the car to monitor the person in the car; if the person appears to be drunk the transmission will be automatically switched off. If accident occurs then bump sensor detects accident and immediately sends SMS to hospitals and police station about location of accident.

II. OVERVIEW OF CAN PROTOCOL

A. Basic CAN

The original CAN interfaces available were later called "Basic" CAN interface. Basic CAN interfaces only offer a limited number of receive buffers and filters (typically 1 to 3)[2]. If a node using such a controller needs to listen to a number of different messages (different CAN message identifiers), the filters usually have to be set "wide open" causing an interrupt with every single message on the bus. Obviously, the microcontroller will get many CAN interrupts, as it has to check in software if a message can be ignored or needs to be worked on.

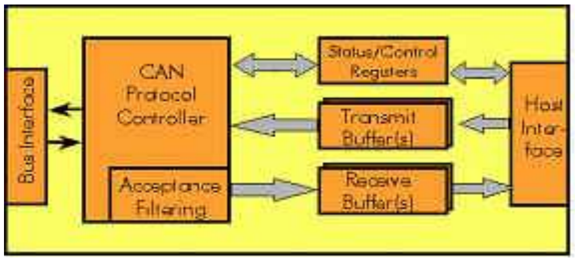

FIG 1. Block Diagram of CAN

B. Full CAN

The next generation of CAN controllers was the so-called "Full-CAN" implementation. Full CAN controllers use a number of message objects (typically 15) with each being bi-directional (can be configured to either transmit or receive), each having its own transmit/receive buffer for one message and each having one filter match register. This allows setting a message object to only listen for exactly one message (one identifier). As long as the total number of messages a node needs to listen to is smaller than the number of message objects available, these interfaces are very efficient. They will only cause an interrupt to the MCU if a "wanted" message was received.

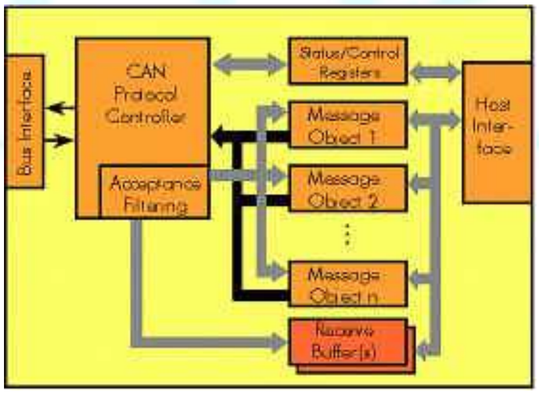

FIG 2. Full CAN

However, this mechanism does not offer any protection from a back-to-back worst-case scenario. Each message object has a single buffer and a matching incoming message will override the buffers' contents, so potentially messages can get lost. As long as a buffer is configured for a single message identifier, this scenario is not too bad, as it is unlikely that the producer of that message produces them back-to-back. But as soon as any of the message objects is configured to receive multiple CAN identifiers - the microcontroller needs to be prepared that these could come in back-to-back. On a 1Mbit CAN network that means about 50 microseconds from receive interrupt occurrence to a potential overwrite of that message by the next incoming message.

C. FIFOs

The only way to get around the back-to-back message problem and the high performance and timing demands on the interrupt service routine is with a receive FIFO buffer (First In - First Out). A typical implementation features a number of filters that include both match and mask registers. Upon a filter match, the incoming message is moved into the FIFO buffer. An interrupt request to the MCU is made depending on configuration: either a certain fill-level is reached or a high priority filter received the last incoming message. Even if such a FIFO can only hold 64 bytes it is still big enough to improve the back-to-back scenario mentioned earlier. If the FIFO is configured to cause an interrupt with every single incoming (matched) message, the MCU has at least 500 bit times until the FIFO will overflow. That is about 10 times more time available to the MCU than with Basic CAN or Full CAN implementations.

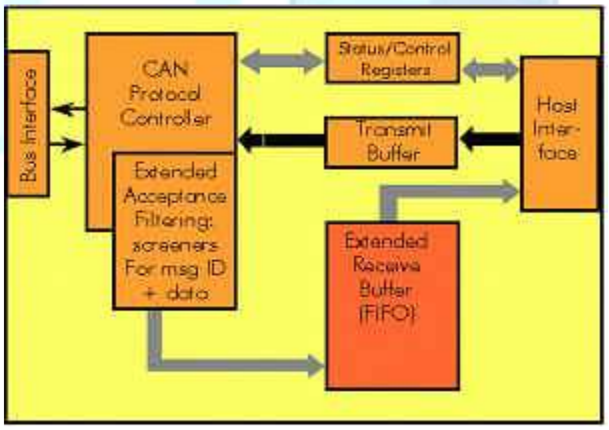

FIG 3. CAN Networking

On the downside, messages in the FIFO cannot overtake each other. So if the FIFO already contains several messages and now an additional, but high priority message comes in - the MCU first needs to process all messages previously stored in the FIFO before it gets access to the high priority message. In a Full CAN interface it is up to the interrupt service routine in which order the message objects are checked and it would very well be possible that a higher priority message can "overtake" previously received lower priority messages.

D. Enhanced Full CAN With Receive FIFO

The latest developments do not have standardized names as chip manufacturers came up with their own customized improvements for the CAN interfaces. Several chip manufacturers now offer devices that combine the benefits of Full CAN and a FIFO. The most powerful approach is a Full CAN implementation with a dedicated FIFO for each single message object. Although powerful, these are also the most complex controllers to configure, especially if each individual FIFO can be freely located in RAM and can be of individual lengths. Another alternative is to be able to take a Full CAN implementation and be able to concatenate message objects to a FIFO. So instead of one message object only having one buffer, a FIFO can be formed "borrowing" the buffers of other message objects. Although fairly flexible, the disadvantage is obvious: with each buffer added to a FIFO, one message object is lost. So the value of this feature increases with a high number of message objects, but decreases if the number of message objects decreases, too.

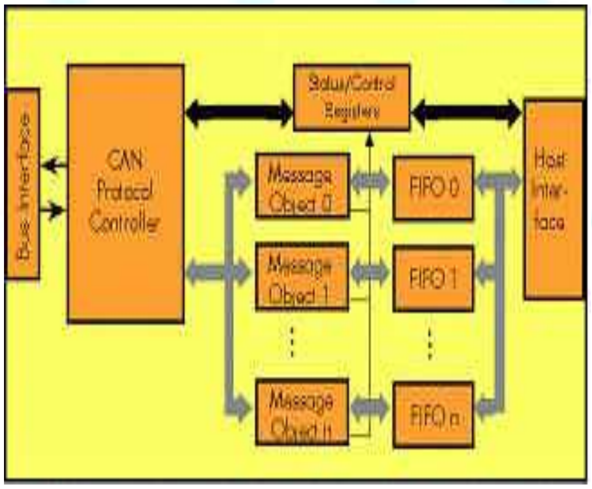

FIG 4. CAN Interfaces

II. DESIGN PROCEDURE

The proposed block diagram for CAN bus communication system is as shown in Figure 3. In this system the ARDUINO board using ATMEGA 328 processor which is interfacing with Controller Area Network to transmit the signal at 50 microseconds. The sensor is connecting with ARDUINO board and transmitting the signal via CAN protocol.

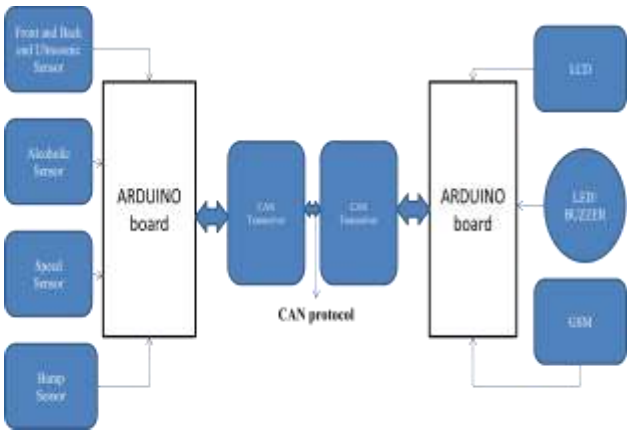

FIG 5.Block Diagram

The alcoholic sensor will sense whether the driver is drunk and if the driver is drunk then the driver will not be allowed to start the car. The speed sensor will monitor the speed of the car and if found high then warning will be given to the driver using an alarm. Here the sensors will communicate with the output devices using CAN (Control Area Network) protocol which will be implemented in the ARDUINO controller.

A. Alcoholic sensor

Alcohol Sensor for use in Breathalyser's or in an alarm unit, to detect the presence of alcohol vapors. This sensor unit offers very high sensitivity, combined with a fast response time. An alcoholic sensor is to monitor the person in the car. If the person appears to be drunk the transmission will be automatically switched off..

Features:

• High Sensitivity
• Detection Range: 10 - 1,000 ppm Alcohol
• Fast Response Time: <10s
• Heater Voltage: 5.0V
• Dimensions: 16mm Diameter, 10mm High excluding pins, Pins - 6mm High

B. Ultrasonic sensor

Ultrasonic sensor is adapted to measure the distance with respect to the previous car. While the car is in motion the distance of another car is measured and accordingly warning signals are given to the driver. The sensor will send out an 8 cycle burst of ultrasound at 40khz and raise its echo line high. It then listens for an echo, and as soon as it detects one it lowers the echo line again. The echo line is therefore a pulse whose width is proportional to the distance to the object. By timing the pulse it is possible to calculate the range in inches/centimeters or anything else. If nothing is detected then the sensor will lower its echo line anyway after about 36Ms.

C. Bump Sensor

The bump sensor detects accidents and if accident is detected then a message is send a message to hospital and police station about location of accident. Piezoelectric plate is the special type of sensor which is used to sense the mechanical vibration. Piezoelectric plate converts the mechanical vibration to electrical signal. The converted electrical signal is in the range of small milli voltage signal.

D. Speed sensor

This sensor monitors the speed of the car and if the speed is found to be more than a prescribed level then a warning signal will be given to the driver.

E. ARDUINO Controller

The ARDUINO Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

IV. PROPOSED ALGORITHM

Algorithm for the proposed system is divided in two parts as follows: Transmitter and Receiver

A. Transmitter

Algorithm for transmitter side which consists of sensors, microcontroller and CAN (MCP2515) is as follows:

1. Initialize SPI (Serial Peripheral Interface).

2. Initialize LCD.

3. Initialize CAN (MCP2515).

4. Provide impulse to ultrasonic sensor.

5. Measure distance from other car and display on LCD.

6. Transmit distance via CAN (MCP2515).

7. If alcohol sensed send X else go to step 8.

8. If lanes cutting detected send Y else go to step 9.

9. Sense speed and if speed goes beyond range send Z else go to step 10.

10. Check for impact and if impact detected sends A else go to step 4.

B. Receiver

Algorithm for transmitter side which consists output devices, ARDUINO microcontroller and CAN (MCP2515) is as follows:

1. Initialize SPI (Serial Peripheral Interface).

2. Initialize LCD.

3. Initialize CAN (MCP2515).

4. Send acknowledgment to the transmitter.

5. Receive distance data from CAN of transmitter and if distance is less then display warning signal on LCD.

6. If X is received then displays “Car cannot be started” else go to step 7.

7. If Y is received then display “Wrong lane” else go to step 8.

8. If Z is received then turn on buzzer else go to step9.

9. If A is detected then send SMS through GSM else go to step 5.

VI. CONCLUSION

With the advancement of technology in every walk of life the importance of safety of vehicle has increased. This project introduces an embedded system with a combination of CAN bus systems. With the rapid development of embedded technology, high performance embedded processor is penetrated into the auto industry, which is low cost, high reliability and other features to meet the needs of the modern automobile industry. With ARDUINO as the main controller and it makes full use of the high performance of ATMEGA 328, high-speed reduction of CAN bus communication control networks and instrument control so as to achieve full sharing of data between nodes and enhance their collaborative work. This system features efficient data transfer among different nodes in the practical applications.

ACKNOWLEDGMENT

This work is supported by my guide Prof.C Kamalanathan and my institution.

REFERENCES

[1] Ashwini S. Shinde, Prof. Vidhyadhar B. Dharmadhikari, "Controller Area Network for Vehicle Automation" International Journal of Emerging Technology and Advanced Engineering www.ijetae.com ISSN 2250-2459, Volume 2, Issue 2, February 2012.

[2]. Li Ran, Wu Junfeng, Wang Haiying, Li Gechen. "Design Method of CAN BUS Network Communication Structure for Electric Vehicle", IFOST 2010 Proceedings IEEE.

[3] Mazran Esro, Amat Amir Basari, Siva Kumar S, A. Sadhiqin M I, Zulkifli Syariff, "Controller Area Network (CAN) Application in Security System" World Academy of Science, Engineering and Technology 35 2009.

[4] Chin E. Lin, Hung-Ming Yen, "A Prototype Dual Can-Bus Avionics System For Small Aircraft Transportation System" 2006 IEEE.

[5] Chin E. Lin, S. F. Tai, H. T. Lin, T. P. Chen, P. K. Chang, C. C. Kao "Prototype Of A Small Aircraft Avionics Using Hybrid Data Bus Technology" 2005 IEEE

[6] Chin E. Lin, H. M. Yen, "Reliability And Stability Survey On Can-Based Avionics Network For Small Aircraft" -7803-9307- 4/05/2005 IEEE.

[7] Feng Xia, Xiaohua Dai, Zhi Wang, and Youxian Sun, "Feedback Based Network Scheduling of Networked Control Systems"2005 IEEE.

[8] Yujia Wang, Hao Su, Mingjun Zhang., "CAN-Bus-Based Communication System Research for Modular underwater Vehicle", 2011 IEEE DOI 10.1109/ICICTA.2011

[9] V. Claesson, C. Ekelin, N. Suri, "The event-triggered and time- triggered medium-access methods", 6th IEEE International Symposium on Object-Oriented Real-Time Distributed computing, May 14-16, 2003, pp. 131-134.