A New Design & Implementation of CAN Bus Protocol for Data Encryption System

Sushil Kumar Dwivedi,Deepika Sharma,Meghraj Pardesi

***Abstract*---Control area network (CAN) isused for automotive application. The goal to make the system more reliable, safe and efficient while decreasing wiring harness weight and complexity, we will modify the system "*A New Design &Implementation of CAN Bus Protocol for Data Encryption System*", which aims to develop a Cost effective, reliable, light weighted, deterministic and robust real time fast and secured embedded control system for Industrial Automation. A large number of low cost hardware platforms such as Raspberry Pi, In our system we are used raspberry pi model B, in which Bluetooth and Wi-Fi both are inbuilt Arduino, Argos Mote etc. are available that do not provide any inbuilt wireless module. In this paper, CAN Bus Protocol on Raspberry Pi is proposed as an integrated solution. Here, we can also have encrypted the data before transmission to provide the higher security. As a result, various industrial applications can be controlled with automation by using new version of Raspberry Pi model B and also the useful transmitted data between two or more devices cannot be hacked by any unauthorized user. ACAN bus project is carried out to fully utilize the application of CAN bus system in building automation, security system and data acquisition system, any software application & IOT technology.**

***Index Terms*---CAN bus Protocol, Raspberry Pi 3 Model-B,Putty Software, Router.**

I. INTRODUCTION

The controller Area (CAN) was developed by Robert Bosch in 1985. It is an international standardization organization (ISO) defined serial communication bus originally developed for the automotive industry to replace the complex wiring harness with a two wire bus. The Proposed work is modified with the features of Raspberry Pi after implementing CAN Bus protocol on it. CAN bus protocol is designed for a high speed serial communication and to provide automation for industrial, medical and in other fields. This CAN Bus Protocol provides an inexpensive durable network that helps multiple devices to communicate with one another. CAN protocol specifications include cyclic redundancy code to perform error checking. The objective of this protocol is to detect and handle error and provide security for data transmission. Another important concept, proposed in the design of the protocol, is to improve the data security, stability and ability of the CAN bus protocol.

In this system, the transmitted data is first converted into an encrypted form and then it is decoded only by the authorized person. So the data is fully secured and cannot be hacked by any unauthorized person. The objective of our project is to eliminate the use of CAN Bus Module hardware and to reduce the high cost of the system and developed a cost effective, reliable, efficient, deterministic, and robust real time distributed embedded control system. When for using the CAN protocol, we want to purchase a CAN bus module which is very costly and also its connection with other devices are very complex. Wired communication between remote devices becomes difficult and costly due to cabling, socket fittings and requires high maintenance cost. These problems were overcome by the concept of wireless media. When we transmit over data from one device to another device that cannot be secured. To reduce this problem, we will modify a system called as **"A new design & implementation of CAN Bus protocol for data encryption system".** Offer implementation of this system, the raspberry-pi model B is integrated with CAN BUS Protocol and because of this protocol, the Raspberry-pi is integrated with new specification.These signals are in the form of binary with error so, it is necessary to identify the bit error. At the receiver end Raspberry Pi find the error which are occur in any bit and it correct the code after that it fetch the original information. This modification of system is done with the help of programming. After that implementation, there is no need to purchase an external CAN Bus module for transmitting our data from one deice to another because CAN Protocol is internally implemented in Raspberry-pi 3 model B

II. OVERALL SYSTEM DESIGN

CAN (Controller Area Network) is a serial bus system used to communicate between several embedded 8-bit and 16-bit microcontrollers. It was originally designed for use in the automotive industry but is used today in many other systems (e.g. home appliances and industrial machines). Highest Baud Rate is 1Mb. CAN uses a message oriented transmission protocol. Raspberry Pi is a SOC board on which we can implement a CAN Bus Protocol. First we open the terminal of Raspberry Pi where we can write a code for CAN Protocol implementation. This code is written in C language. The below figure explains the working of Raspberry Pi with CAN bus implementation. When the data is send from one system then it goes to another Raspberry Pi these data is also in encrypted form. Here Raspberry Pi auto detect the signals.

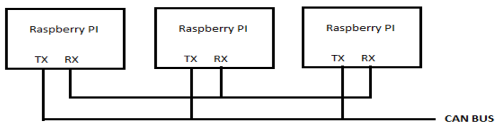


Fig 1 : Raspberry Pi Bus.

III. RASPBERRY PI 3 MODEL B

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom (UK) by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. Basically we have used third generation Raspberry Pi which is Raspberry Pi 3 Model B in our system to improve the speed and performance of data transmission and reception. It replaced the Raspberry Pi 2 Model B in February 2016. The Raspberry Pi 3 Model B with BCM2387 ARM Cortex - A53 Quad Core Processor powered Single Board Computer running at 1.2 GHz as 10 times the performance of a Raspberry Pi 1. It also contains inbuilt Bluetooth 4.1, BLE (Bluetooth Low Energy) and 802.11n Wireless LAN facility which is not provided by old versions of Raspberry Pi. The Raspberry Pi 3 has an identical form factor to the previous Raspberry Pi 2 (and Pi 1 Model B+) and has complete compatibility with Raspberry Pi 3 model B. An interface for output unit such as VGA Monitor, touch screen and input unit such as keyboard, keypad, and mouse.

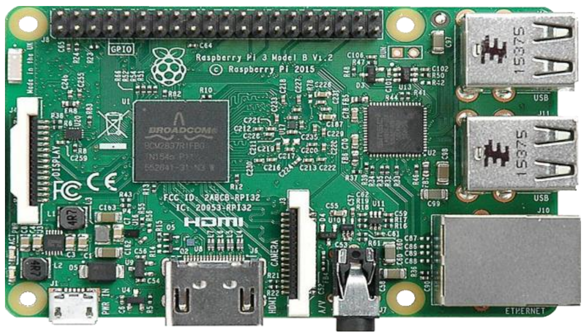
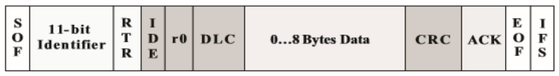


Fig 2 : Structure of Raspberry Pi 3 Model B

IV. CAN BUS PROTOCOL

CAN is a broadcast digital bus designed to operate at speeds from 20kb/s to 1Mb/s, standardized as ISO/DIS 11898 for high speed applications (500 kbit/s) and ISO 11519-2 for lower speed applications (125Kbit/s). The transmission rate depends on the bus length and transceiver speed. CAN is an attractive solution for embedded control systems because of its low cost, light protocol management, the deterministic resolution of the contention, and the built-in features for error detection and retransmission. Raspberry Pi model B supports the CAN communication standard that are today widely available as well as sensors and actuators that are manufactured for communicating data over CAN.

CAN networks are today successfully replacing point-to-point connections in many application domains, including automotive, avionics, plant and factory control, elevator controls, medical devices and possibly more. The CAN standard has been developed with the objective of time determinism and support for reliable communication. At first we design the frame work of CAN Bus Protocol which are given

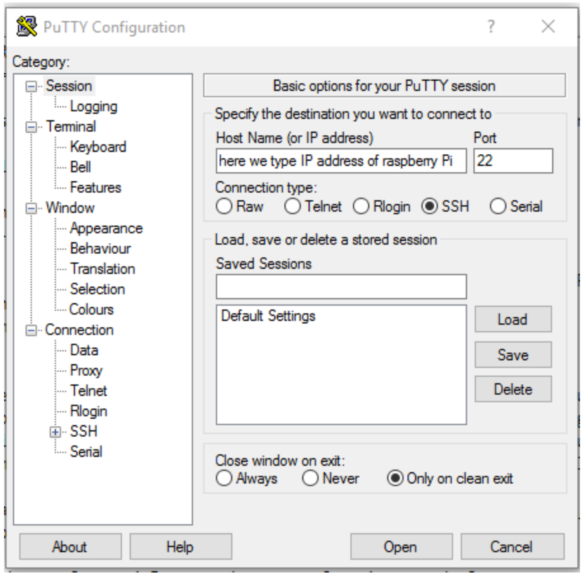


SOF	11-bit Identifier	RTR	IDE	r0	DLC	0…8 Bytes Data	CRC	ACK	EOF	IFS

Fig 3 : CAN Frame.

V. PUTTY

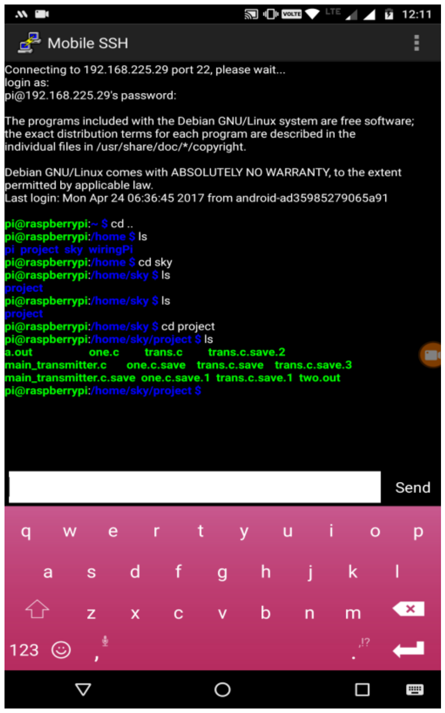
PuTTY is a free software and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. The name "PuTTY" has no definitive meaning. PuTTY was originally written for Microsoft Windows, but it has been ported to various other operating systems. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and unofficial ports have been contributed to platforms such as Symbian, Windows Mobile and Windows Phone. PuTTY as written and is maintained primarily by Simon Tatham and is currently beta software. PuTTY is a free SSH. SSH is a network protocol that allows you to logging in to a multi-user computer from another computer, over a network. On the computer you sit at, you run a client, which makes a network connection to the other computer (the server). The network connection carries your keystrokes and commands from the client to the server, and carries the server's responses back to you. Using this type of interface, there is no need for you to be sitting at the same machine you are typing commands to. The commands, and responses, can be sent over a network, so you can sit at one computer and give commands to another one, or even to more than one. Protocols can also be used for other types of keyboard-based interactive session. In particular, there are a lot of bulletin boards, talker systems and MUDs (Multi-User Dungeons) which support access using Telnet.



Specify the destination you want to connect to
Host Name (or IP address): here we type IP address of raspberry Pi
Port: 22
Connection type: Raw Telnet Rlogin [✓ SSH] Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Close window on exit: Always Never [✓ Only on clean exit]

Fig 4 : PuTTY configuration

With the help of smartphone we are able to access IOT technology the mobile SSH an android utility allows usto connect our android devices to raspberry pi over the same network using SSH protocol. Raspberry pi can be access from anywhere in the network.



```
Connecting to 192.168.225.29 port 22, please wait...
login as:
pi@192.168.225.29's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Apr 24 06:36:45 2017 from android-ad35985279065a91

pi@raspberrypi:~ $ cd ..
pi@raspberrypi:/home $ ls
pi  project  sky  wiringPi
pi@raspberrypi:/home $ cd sky
pi@raspberrypi:/home/sky $ ls
project
pi@raspberrypi:/home/sky $ ls
project
pi@raspberrypi:/home/sky $ cd project
pi@raspberrypi:/home/sky/project $ ls
a.out                  one.c         trans.c        trans.c.save.2
main_transmitter.c     one.c.save    trans.c.save   trans.c.save.3
main_transmitter.c.save  one.c.save.1  trans.c.save.1  two.out
pi@raspberrypi:/home/sky/project $
```

Fig 5 : Operate on Smartphone

In Raspberry Pi, we make the instructions which follows the frame of CAN Bus Protocol. Explanation of that instructions are –

Remote Transmission Request (RTR) :

Uint8_trtr_tx = 0x00;

Identifier Extension (IDE) :

Uint_t ide _tx = 0x01;

Ro (Reserved bit) :

uint 8_tx = 0x01

Data Length Code (DLC) :

dlc_tx>=16;

0…..8 bytes data :

In this frame data are enter by user

Cyclic Redundancy Check (CRC) :

CRC check the 16 bit (15 bits plus delimiter)

uint 16_t crc _ tx = 0;

uint 8_t crc _tx_low = 0;

uint 8_t crc tx_tx_high = 0;

Acknowledge (ACK) :

Ack_rx = serialGetchar (fd);

End Of Frame (EOF) :

eof_rx = rx = serialGetchar(fd);

Inter Frame Space (IFS) :

ifs_rx = serialGetchar (fd)

The above instruction are the basic instructions which are used to implement a CAN Protocol and to give initial values.

In this system, we use C language to implement CAN Bus Protocol. We know that the CAN Bus module transmits data from one device to another which is not highly secured. In our system, data is also transmitted which are also in encrypted form. Here when we transmit the data we will send the interrupt in bit. We know that when the data is sending; the data is transmitted which are in digital form. Digital data is in the form of binary it means that in the form of “0” and “1”. When we send the data, at first data is converted into binary form. The implementation of CAN bus protocol are sending the bit error with the sending message. CAN Bus protocol transmitter is the user input with receiver output pins which are passively piled with high internally. When the input is zero, the Raspberry pi is automatically receives to a recessive bus state on both input and output pins.

The allocation of priority to message in the identifier is a feature of can that makes it particularly attractive for use within a real time control environment. Zero(0) the binary message identifier numbers, the priority of its is high. Each user recognizer are consists of zero has the highest priority of the message on the network. It retains the bus foremost thelongest. When two nodes are start to transmit simultaneously. The nodes are sending the last user identifier bit as a zero while the other nodes are sending a one (recessive) retains control of the CAN bus protocol andcomplete its message. The higher priority bits are always overwrites a recessive on bit in a CAN bus. The transmitted nodes are automatically monitors each bit of its own transmission.

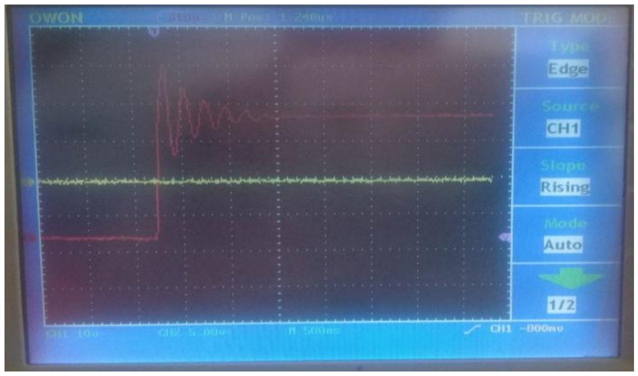


Fig 6 : Transmissionof CAN bus data

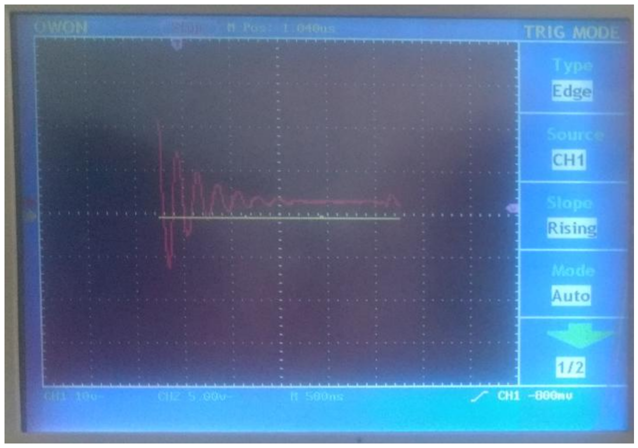


Fig 7 : Receiving of CAN bus data

Proposed work is modified the features of Raspberry Pi after implementing CAN Bus protocol on it. CAN bus protocol is design for a high speed serial communication and to provide automation for industrial, medical and in other fields. This CAN Bus Protocol provide an inexpensive durable network that helps multiple devices communicate with one another. CAN protocol specification include cyclic redundancy code to perform error checking. The objective of this protocol is to detect and handling error and provide security for data transmission. Another important concept, proposed in the design of the protocol, is to improve the data security, stability and ability of the CAN bus protocol.

In this system, the transmitted data is first converted into an encrypted form and then it is decoded only by the authorized person. So the data is fully secured and cannot be used by any unauthorized person.

The objective of our project is to eliminate the use of CAN Bus Module hardware and to reduce the high cost of the system and developed a cost effective, reliable, efficient, deterministic, and robust real time distributed embedded control system.

REFERENCES

[1] Garima, Nidhi Agarwal Research Associate & Dr. S.R.N Reddy Associate Professor CSE Dept. IGDTUW, Delhi, India on "Design & Development of Daughter Board for Raspberry Pi to support Bluetooth Communication using UART " published by International Conference on Computing, Communication and Automation (ICCCA2015).

[2] Mr. Shripad Deshpande Syncspace Solution Pvt.Ltd & Thakor Bhishmapalsinh litendrasinh M.E, VLSI & Embedded System Design Gujarat Technical University, India on " Implementation of CAN Bus Protocol on XENOMAI RTOS on ARM Platform for Industrial Automation " published by International Conference on Computation of Power, Energy Information and Communication (ICCPEIC2016).

[3] Pradhan Suvendu Kedareswar, Venkatasubramanian Krishnamoorthy from School of Electronics, VIT, Chennai, India on " A CAN Protocol Based Embedded System to Avoid Rear-End Collision of Vehicles " published by IEEE 2015.

[4] A. A. Salunkhe, Pravin P kamble, & Rohit Jadhav from Asst. Professor, Department Of Electronics Engineering, Walchand College Of Engg., Sangli, Maharashtra, India on " Design And Implementation of CAN bus Protocol for Monitoring Vehicle Parameters " published by IEEE International Conference On Recent Trends In Electronics Information Communication Technology, May 20-21, 2016, India.

[5] Michiel van Osch & Scott A. Smolka, Department of Math and Computing Science Eindhoven University of Technology, Eindhoven, The Netherlands on "Finite-State Analysis of the CAN Bus Protocol" published by IEEE International Symposium on High Assurance Systems Engineering (HASE'01) 1530-2059/01 $17.00 © 2001 IEEE.

[6] Tao Yizheng, Tang Dingyong, Gaoshan, Shenhao Institute of Computer Application, China Academy of Engineering Physics, Mianyang, Sichuan, China on **"Design and Implementation of USB Key-based JavaEE Dual-factor Authentication System"** published by International Conference on Information Management, Innovation Management and Industrial Engineering © 2009 IEEE.

[7] Artal J.S., Caraballo J. & Dufo R. Department of Electrical Engineering University School of Engineering and Architecture, EINA. University of Zaragoza, Spain on **"CAN/LIN-Bus Protocol. Implementation of a low-cost serial communication network"** published by ©2014 IEEE.

Sushil KumarDwivedi- B.E, ECE dept,R.G.P.V, Bhopal,India.

Deepika Sharma- MIST, Asst. Professor, ECE dept M.Tech, ECE dept,R.G.P.V, Bhopal, India

Meghraj Pardesi- B.E, CSE dept,R.G.P.V, Bhopal, India.