CAN based protocol implementation between battery charger and battery management system for electric vehicles

NAYAN KHULBE and CHANDRASHEKHAR CHOUDHARY

Department of Electronic System Design, International Institute of Information Technology, Bangalore-560100

ABSTRACT : The study was conducted at International Institute of Information Bangalore. In the study a protocol based on CAN or Controller Area Network has been implemented for communication between the Battery charger and Battery management system or BMS. Modern days electric vehicles uses Lithium ion batteries for charging which has a Battery management system to monitor various parameters of a battery such as current, voltages and temperature. The implementation of CAN protocol enables the Battery management system to communicate with the battery charger. The BMS instructs the battery charger to perform necessary action when the values of the parameters goes out of range. This helps in preventing any damages to the battery and maintain a healthy life of the battery. The charger also becomes intelligent enough to take decisions without any human monitoring. These charger can directly be deployed on board an electric vehicle.

Key words: Controller Area Network, Battery management system, Battery charger, Piccolo microcontroller

The increase in the environmental pollution has led to the development of electric vehicles which is a clean alternative to our conventional petrol and diesel vehicles (Wu and Zhang, 2017). Electric vehicles have zero exhaust emission which helps in reducing the harmful air pollution. Most of the modern days electric vehicles uses Lithium ion batteries for running, which has a battery management system to communicate with various functional blocks inside a vehicle such as sensors, actuators and battery charger. The aim of our study is to implement a CAN protocol for communication between the battery charger and the Battery management system or BMS. This makes the battery charger intelligent enough to communicate with the Battery management system and take decisions without any human monitoring, which prevents the battery from any damages. CAN or Controller Area Network is a multi master half duplex protocol that supports real-time control with a communication rate of upto 1 Mbps. The idea of CAN was first given by Robert Bosch in 1983 and was first released in 1986. It is widely used in automobile industry for communication among various devices in the vehicle. CAN has advantages over other protocols as it ensures a noise and error free transmission. It eliminates traffic congestion as messages are transmitted based on their priority. CAN is the most preferred protocol to communicate with the Battery management system. The Battery management system of a electric vehicle controls the charging as well as discharging of a battery. The BMS provides data monitoring of various parameters such as current, voltages and temperature (Mustafa Turgut, Raif Beyir, Fecir Duran, 2018). It acts as a protector for battery system by examining the real time states of the battery and keeping these states within the threshold limit. The Battery management system communicates with the battery charger over the CAN network and directs the charger to take necessary action in case the values of current, voltages or temperature cross the threshold limit. The charger also provides various information to the Battery management system such as the current supplied, charger temperature etc. This protocol was implementedand tested at Research and Development lab at International Institute of Information Technology Bangalore.

MATERIALS AND METHODS

1) A Texas TMS320F28069U Piccolo Microcontroller for Battery management system.

The Battery management system is emulated as an application code in Texas microcontroller. It is to be noted that their no actual battery in the test setup. In order to emulate a Battery management system the specifications for a 12V-100Ah Lithium ion battery is hard coded in the controller. The specification taken into account are values of Operational voltage charge and discharge, maximum operating temperature, maximum charge and discharge current. The Piccolo Microcontrollers which is used here, has a 32 bit high

efficiency CPU (Central Processing Unit) which is based on the Harvard architecture. It has a Flash memory of 256KB, a 100KB of RAM (Random Access Memory) and a 2KB of One Time Programmable ROM (Read only memory). Piccolo microcontrollers have various serial port peripherals such as a Serial Peripheral Interface module, two Universal Asynchronous Receiver Transmitter module, one Inter Integrated circuit Bus, one Enhanced CAN module and a Universal Serial Bus. It also has 54 programmable multiplexed General Purpose Input or Output pins [http://www.ti.com/lit/ds/symlink/tms320f28069. pdf].

2) A 100W lithium ion battery charger consisting of Texas TMS320F28069U piccolo microcontroller.

The microcontroller inside the charger also runs an application code which encodes and decodes the messages coming from the BMS. The charger takes necessary action based on the instruction given to it by the Battery management system.

3) Texas Code Composer Studio to run the application code.

Code Composer Studio is an integrated development environment (IDE) to develop application for Texas instruments embedded processors. It is primarily designed as for embedded product design and low level based debugging. However the latest releases are based on unmodified version of eclipse open source IDE, which can be easily extended to support OS (Operating System) level application debug (Linux, Android, Windows Embedded).

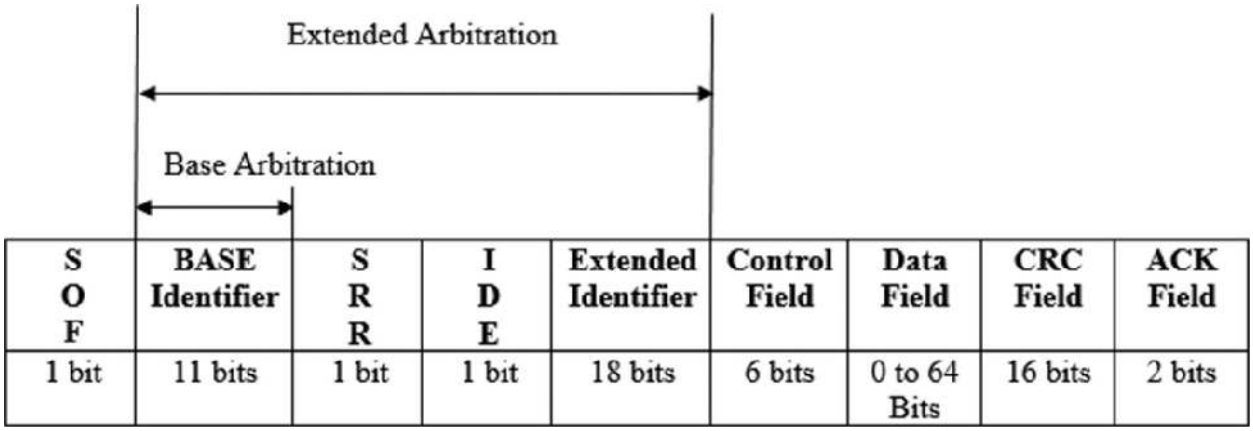
4) JTAG (Joint Test Action Group) interface for programming and debugging the microcontroller.

It is a industry standard for accessing sub-blocks of integrated circuits, making it essential mechanism for debugging embedded devices. JTAG allow the software developer to access the debug modules inside the target CPU and debug the embedded modules at machine level instruction or at high level language source code.

In the study we used Enhanced CAN module which is a part of piccolo microcontroller. It provides the functionality of CAN protocol, version 2.0B. CAN version 2.0B data frame has a standard format which has 11 bits base identifier and a extended format with 29 bits base identifier. This base identifier represents a unique identification for each messages in the data frame. Figure 1 shows the data frame which includes the following (http://www.ti.com/lit/ug/spruh18g/spruh18g.pdf.):

- Start of frame bit.
- Arbitration field which contains RTR (Remote Trasmission Request), SRR (Susbsitute Remote Request) and IDE (Identifier Extension).
- Control field which indicates the number of bytes to be transmitted.
- 8 Bytes of data to be transmitted
- Cycle Redundancy Check (CRC) for error checking.
- Acknowledgment bits.
- End of frame bits.

Figure 2 shows the architecture of the CAN module which is composed of a Message controller, a Protocol Kernel and a CAN transceiver. The Protocol Kernelconsists of a Receive buffer and a Transmit buffer which receives and transmit the messages on the CAN bus [http://www.ti.com/lit/ug/spruh18g/spruh18g.pdf.].



S O F	BASE Identifier	S R R	I D E	Extended Identifier	Control Field	Data Field	CRC Field	ACK Field
1 bit	11 bits	1 bit	1 bit	18 bits	6 bits	0 to 64 Bits	16 bits	2 bits

Fig. 1: CAN Data Frame

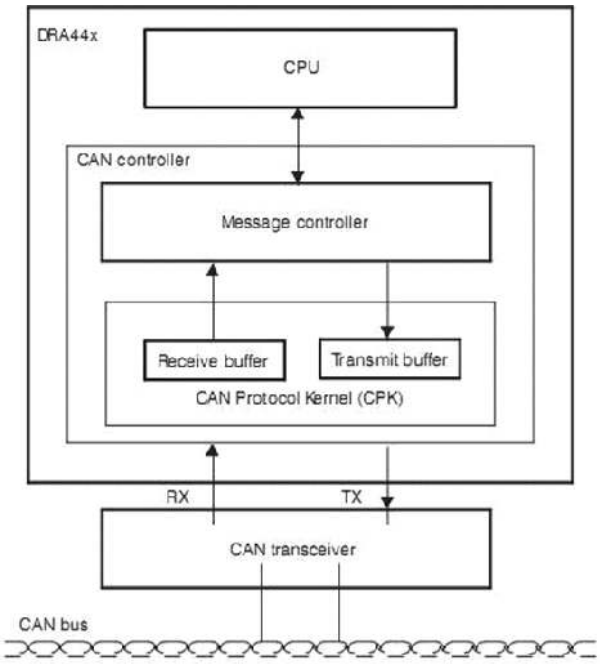


Fig. 2: CAN Controller Architecture

Message controller consists of a RAM and a memory management unit which receives the messages from receive buffer and store them into the RAM and also transmit the messages to the transmit buffer based on their priority.

On the physical layer the CAN transceiver has two dedicated wires for communication. These wires are CAN high (Recessive logic high) and CAN low (Dominant logic low). In the idle state both the wires carry 2.5V. During the communication the CAN high goes to 3.75V and CAN low drops to 1.25V, thus maintaining a differential voltage of 2.5V as shown in Figure 3. This differential prevents CAN bus from any electric noise or inductive spikes.

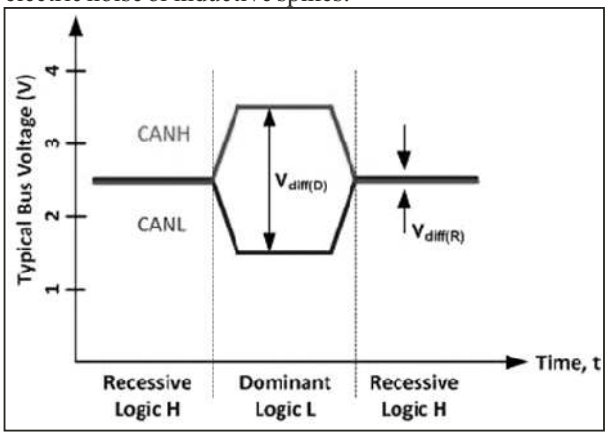


Fig. 3: CAN Differential Voltage

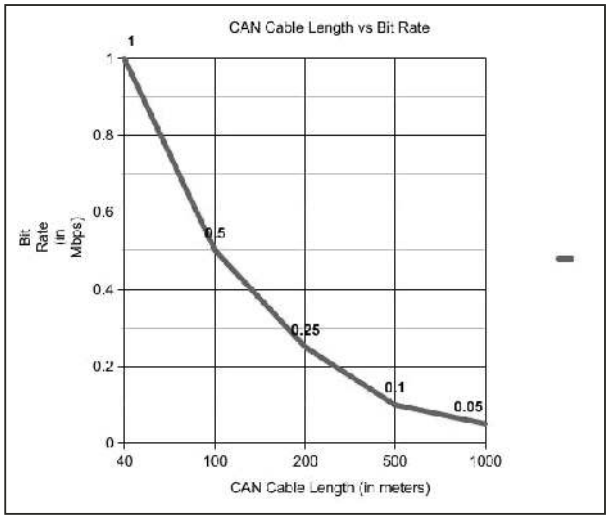


Fig. 3: CAN Cable Length VS Bit Rate

The equation for calculating bit rate is given below.

Bit Rate = (SYSCLKOUT) / (2*BRP*Bit Time) where SYSCLKOUT is the CAN clock frequency, BRP is Bit Rate Prescaler and Bit time is the number of time quanta per bit. In the study we desired to achieve a bit rate of 1Mbps. The Bit Rate Prescaler is a register in the CAN module which we need to set according to our bit rate requirements. In the study the Bit Time of 10 Quanta was considered, the value of which is taken from Texas reference manual (http://www.ti.com/lit/ug/spruh18g/spruh18g.pdf.). The value of CAN clock frequency was taken as 100Mhz. The Bit rate prescaler was calculated as 5 and value is set in the application code.

Texas Code Composer is used to run the application code inside the microcontroller. A JTAG is connected from our system to controller via Universal Serial Bus which is used to load the program into the RAM of the controller. The connection between the BMS and battery charger is established by connecting the CAN high pin of one microntroller to the CAN high pin of another microcontroller. Similarly, CAN low pins are also connected together. The communication between the two nodes starts with a CAN Enable message sent by charger to the BMS. As soon as the BMS decodes the CAN Enable message it sends Charger Enable message to the charger. This message is the instruction for the charger to supply current to the battery.

The BMS keeps monitoring the current, voltage and the temperature of the battery. JTAG interface used for debugging, provides us with the functionality of

Table 1: CAN vs LIN vs Flex Ray

Protocol	Messaging Type	Node control	Datarate (Max)	Physical character
CAN	Event Triggered	Multi-Master	1Mbps	2- wire interface
Flex Ray	Event and Time Triggered	Maste/ Slave	20Mbps	2-wire interface
LIN	Static Messages	Maste/Slave	20Kbps	1-wire interface

changing the real time values of parameters such as current, voltages and temperature in the application code of BMS (https://en.wikipedia.org/wiki/JTAG). In order to test the code these values were varied accordingly. As soon as the BMS finds that the operational charge voltage has reached to its maximum value, which is 15V for Lithium ion battery taken into consideration [https://www.mpow eruk.com/communications.html], it sends a Charger Shutdown message to charger. Similarly, if the value of operational voltage discharge drops down to 9.2V (http://www.claytonpower.com/products /lithium-ion- batteries/specifications), a Charger Shutdown message is sent. The charger upon receiving the message stop charging the battery. Finally a CAN Disable message is sent to disable the communication.

The BMS also takes care of the critical conditions such as if there is sudden increase in temperature of the battery beyond 50 degree celcius, which is maximum for the Lithium ion battery taken into consideration, the BMS sends a Emergency Charger Shutdown message. Similar action was taken if the voltages and the current values crosses a threshold value.

RESULTS AND DISCUSSION

The results shows that CAN is a very reliable protocol. The test was run for several number of iterations and no data loss was seen. CAN can achieve a very high speed of 1Mbps. The study shows that in the event of some critical situations like sudden peak in voltages and current or over temperature as discussed earlier the charger shutdown happened immediately which helps to prevent any damage to the Lithium ion battery. This can only be achieved if the protocol is more reliable and supports faster communication speed. We were able to achieve a maximum bit rate for communication as the cable length was around one meter. The data rate utpo 40 meter was around 1Mbps. After 40 meter as the length is doubled the data rate is almost decreased by two as shown in Figure 3.

The steady states losses, degradation of signal transition time and inter symbol interference are the factors which leads to degradation of bit rate (http://www.tij.co.jp/jp/lit/an/slla270/slla270.pdf).

In the study we used CAN as our protocol to establish communication, but in the automobile industry some other protocols are also used such as LIN (Local Interconnect Network) protocol and Flex Ray (https://www.mpow eruk.com/communications.html), which have their own advantages and disadvantages over CAN. The comparison is shown in Table 1 below.

CONCLUSION

In this study a CAN protocol has been implemented for communication between the battery charger and the Battery management system. The battery management system monitors various parameters of the battery which are considered critical to maintain a healthy battery life and prevent any damage to the battery. This protocol enables the battery charger to communicate with the Battery management system and take decisions accordingly. This makes the battery charger intelligent enough to work without any human monitoring. The CAN protocol which is used for communication makes the overall system more reliable and accurate as no data losses was noticed during the study. The study concludes that a battery charger has been made intelligent enough to be deployed in modern days electric vehicles.

REFERENCES

Y. Wu and L. Zhang (2017). “Can the development of electric vehicles reduce the emission of air pollutants and greenhouse gases in developing countries?”, Transportation Research Part D: Transport and Environment, vol. 51:129–145, DOI: 10.1016/j.trd.2016.12.007.

Mustafa Turgut, Raif Beyir, Fecir Duran, (2018). Can communication based modular type battery management system for electric vehicles. ELEKTRONIKA IR ELEKTROTECHNIKA, ISSN 1392-1215, VOL. 24, NO. 3.

MInxin Zheng, Bojin Qi, Hongje Wu (2008). A Li-ion battery management system based on CAN-bus for electric vehicle. Industrial Electronics and Applications, 2008, ICIEA.

Lithium Ion Battery Specifications, http://www.claytonpower.com/products/lithium-ion- batteries/specifications.

Batterry, Communications Interfaces. https://www.mpoweruk.com/communications.html.

TMS320x2806x Piccolo, Technical Reference Manual, http://www.ti.com/lit/ug/spruh18g/spruh18g.pdf.

TMS320F28069 Piccolo Microcontroller datasheet, http://www.ti.com/lit/ds/symlink/tms320f28069. pdf.

JTAG, https://en.wikipedia.org/wiki/JTAG.

http://www.tij.co.jp/jp/lit/an/slla270/slla270.pdf.

M. Di Natale (2012).Reference Architecture of CAN Based System, Springer, 2012.

N. Raju, Vemulla Pallavi (2013) "Can Protocol Implementation for Industrial Purposes", Internationaal Journal of Computer Science and Mobile Computing.

Chee Kyun Ng, Wei Lun Ng (2010). Review of Researches in Controller Area Network Evolution and Application", Asia Pacific Advance Network.

Received: March 11, 2019
Accepted: July 4, 2019