

An ARM 7 Based Temperature Measurement System Using CAN Bus

Ms. A.A. Paturkar *, Dr.P.T.Karule **, Mr. A.S.Dikholkar ***

*(Department of Electronics Engineering, Yeshwantrao Chavan College of Engineering, Nagpur
Email: aasawari.paturkar@yahoo.com)

** (Professor, Department of Electronics Engineering, Yeshwantrao Chavan College of Engineering, Nagpur
Email:ptkarule@rediffmail.com)

*** (Asst. Professor, Department of Electronics Engineering, Yeshwantrao Chavan College of Engineering, Nagpur
Email:amit.dikholkar@gmail.com)

ABSTRACT :

Controller Area Network (CAN) is an effective choice for the automotive industries due to its simplicity, low cost nature, and in addition to that it provide connectivity with multiple nodes with single wiring pair. This paper aims in describing an ARM 7 LPC 2129 based Temperature measurement system with Controller Area Network (CAN) bus. Objective of this paper is to construct a hardware proposal for communication between nodes with CAN bus. A Node to Node communication link has been established which are connected via CAN bus so as to observe and control the temperature values, transfer on the CAN bus and send the control signals onto the another node. The hardware boards are designed to be capable of supporting high speed bit rate up to 1Mbps. The CAN communication module can transmit or receive data following CAN Protocol. With the help of the software used, the CAN-BUS communication would be clear to the nodes of the ARM 7 Controllers in the system. The software part is done in KEIL μ vision-4 using Embedded –

Keywords- CAN Bus, ARM-7 LPC2129, Temperature sensor, communication module.

I. INTRODUCTION

The high demand of automation processes with focus basically on factory automation has required of new technologies in the modern communication systems such as the automation systems (PCs, PLCs, etc.) and the field devices (sensors, actuators etc.)

Nowadays, major communication networks can be divided into four types, namely: IP Core Network/Internet, Wireless LAN, 3G/4G Cellular Network and Ad-hoc PAN [1]. The common usage for these networks is to carry text, audio and video content. Recently, some of these networks have been utilized in industrial automation to monitor and control industrial plants.

Another type of networks is the Controller Area Network. [2] CAN is intended as a communication network between the control units in vehicles. Nowadays, CAN applications are gaining ground and it is extending to industrial automation including marine and aircrafts electronics, factories, cars, trucks and many others. The backbone of the Controller Area Network is a fast serial bus that is designed to provide a reliable, efficient, and very economical link between sensors and actuators. CAN use a twisted-

pair cable to communicate at speeds up to 1 Mbits/sec, with up to 40 devices. CAN was originally developed to simplify the wiring in automobiles. In the past, automobile manufacturers used to connect devices in vehicles using point-to-point wiring systems.

As more electronics and controllers are deployed to monitor and control vehicles, wiring started to become more complex, bulky, heavy and expensive. Automotive industry starts to reduce massive wires complexity with dedicated CAN link that provides low-cost, robust network, and multi-master [6] communication system.

II. CAN CONCEPTS:

CAN was originally developed by Robert Bosch (Germany, 1986) when Mercedes requested a communication system between three electronic control units in vehicles [6]. It is a multi-master, multicast protocol without routing. Point to point communication was not suitable anymore, and the need of using a multi-master communication system became necessary. Although this origin can be traced to automotive industry, industrial automation rapidly showed the need of using such a popular bus system

and currently CAN is not restricted to Automotive Industries. It satisfies the communication needs of a wide range of applications, from high speed networks to low cost multiplexing. Multiple CANs with different speeds can be used in particular system. A CAN port is a two-wire, half duplex, high-speed network system that can reach a throughput up to 1 Mbits/sec. Data, control commands and devices status can be transmitted and/or received in well structured frames. Theoretically, CAN is capable of linking up to 2032 devices on a single network; due to hardware limitation, only 110 nodes can be linked-up to construct a single network.

CAN (Version 2.0) have two different standards: CAN 2.0 A, standard CAN, using 11 bits for node identification; and the other standard is CAN 2.0 B or extended CAN, using 29 bits for node identification. With 11 bit, 2,048 unique messages are possible, whereas 536 million unique messages are possible with 29 bit identifier [6].

In a controller area network, all nodes - recipients - can see all messages and can accept or ignore any messages according to the system design. There are two ISO standards classifying the two CAN's physical layer in terms of data rate; ISO 11898 which can handle speed up to 1 Mbit/sec and ISO 11519 that can handle speed up to 125 Kbit/sec, as shown in table.

	Standard	Signal Rate	Identifier
Low Speed CAN	ISO 11519	125 kbps	11 bit
CAN 2.0A	ISO 11898	1 Mbps	11 bit
CAN 2.0 B	ISO 11898	1 Mbps	29 bit

Table 1: IEEE Structure

CAN have the following properties:

- Prioritization of messages.
- Guarantee of latency times.
- Configuration flexibility.
- Multicast reception with time synchronization.
- System wide data consistency.
- Error detection and error signaling.

2.1 CAN Protocol and Terminology

Controller Area Network (CAN) is a multi-master serial data bus which uses Carrier Sense Multiple Access/ Collision Resolution (CSMA/CR) to determine access. CAN was designed as a simple and robust broadcast bus capable of operating at speeds of up to 1 Mbit/s. Message transfer over CAN is controlled by 4 different types of frame: Data frames, Remote Transmit Request (RTR) frames, Overload frames and Error frames. Each CAN data frame is required to have a unique identifier. Identifiers may be 11-bit (standard format) or 29-bit (extended format).

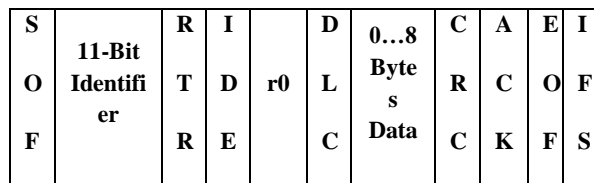


Figure 1: A Standard CAN Frame.

The identifier serves two purposes beyond simply identifying the message. First, the identifier is used as a priority to determine which message among those contending for the bus will be transmitted next. Second, the identifier may be used by receivers to filter out messages that they are not interested in, and so reduce the load on the receiver's host controller. In this paper we are more focused on data frames together with remote frames for requesting data from another node.

2.2 Priority Based Arbitration

Priority of an Identifier defines a fixed message during bus access while Arbitration in CAN protocol tells us that, when the bus is free any node may start to transmit a message, if multiple nodes start transmitting at the same time the bus access conflict is resolved by bit-wise arbitration using identifier. If Data Frame and Remote Frame with the same identifier are initiated at the same time, the data Frame prevails over the Remote Frame. The CAN physical layer supports two states termed dominant ('0') and recessive ('1'). If two or more CAN controllers are transmitting at the same time and at least one of them transmits a '0' then the value on the bus will be a '0'. This mechanism is used to control access to the bus and also to signal errors. The CAN protocol calls for nodes to wait until a bus idle period is detected before attempting to transmit. If two or

more nodes start to transmit at the same time, then by monitoring each bit on the bus, each node can determine if it is transmitting the highest priority message (with a numerically lower identifier) and should continue or if it should stop transmitting and wait for the next bus idle period before trying again. As the message identifiers are unique, a node transmitting the last bit of the identifier field, without detecting a '0' bit that it did not transmit must be transmitting the message with the lowest numerical value and hence the highest priority that was ready at the start of arbitration. This node then continues to transmit the remainder of its message, all other nodes having backed off. The requirement for a node to be able to overwrite a recessive bit, and the transmitting node detect this change, limits the combination of physical length and speed of CAN bus. The duration of each bit must be sufficient for the signal to propagate the length of the network. This limits the maximum data rate to 1Mbit/s for a network up to 40m in length or to 125Kbit/s for a 500m long network. The arbitration mechanism employed by CAN means that messages are sent as if all the nodes on the network shared a single global priority based queue. The above high level description is a somewhat Simplified view of the timing behavior of CAN. CAN does not have a global concept of time, rather each CAN controller typically has its own clock which, within a tolerance specified by the protocol, may drift with respect to the clocks of other nodes.

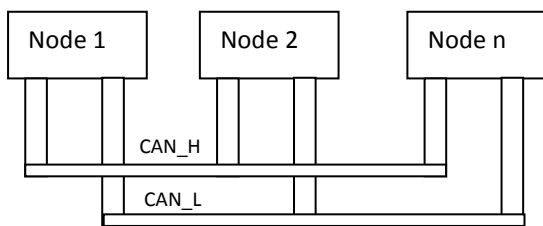


Figure 2: Can Nodes

The CAN protocol therefore requires that nodes re-synchronize on each message transmission. Specifically, every node must synchronize to the leading edge of the start of frame bit caused by whichever node starts to transmit first. Normally, CAN nodes are only allowed to start transmitting when the bus is idle. Thus, when the bus is idle beyond the 3-bit inter-frame space and a node n starts to transmit a message beginning with the dominant start of frame bit ("0"), then all the other nodes

synchronies on the leading edge of this bit and become receivers – i.e. they are not permitted to transmit until the bus next becomes idle. In this case any message that becomes ready for transmission after the leading edge of the start of frame bit has to wait for the next bus idle period before it can enter into arbitration.

However, to avoid problems due to clock drift, the CAN protocol also specifies that, if a CAN node has a message ready for transmission and detects a dominant bit at the 3rd bit of the inter-frame space, it will interpret this as a start of frame bit, and, with the next bit, start transmitting its own message with the first bit of the identifier without first transmitting a start of frame bit and without becoming a receiver⁵. Again the leading edge of the start of frame bit causes synchronization. This behavior ensures that any messages that become ready for transmission, whilst another message is being sent on the bus, are entered into the next round of arbitration, irrespective of any, within tolerance, clock drift.

III. PROPOSED SCHEME

Proposed system includes the ARM7 controller as the main controller connected with several ARM 7 controllers [8] acting as Device 1, Device 2. Here we are trying to communicate two nodes of ARM7 controller using CAN bus. Baud rate should be constant throughout the communication so as to have synchronized communication. When this data is received by the Device 1 or Device 2, it is acknowledged by the Peripherals connected to the devices such as LCD.

ARM7 can communicate with Device 1 or Device 2 by one to one communication or they can also have intercommunication which will be controlled by main controller.

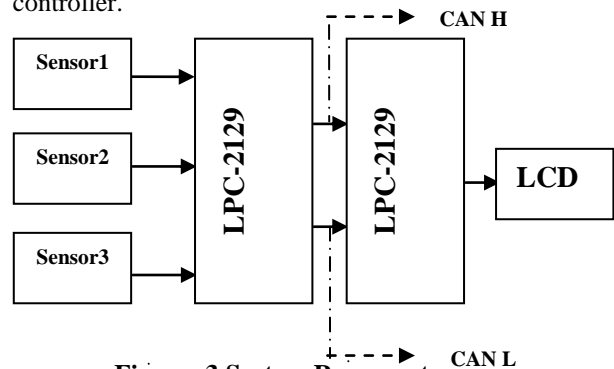


Figure: 3 System Represent

Figure above shows general block representation of the module. The main objective of this scheme is to establish communication between two devices so that a process of operation is carried out with the peripherals.

CAN bus uses two dedicated wires for communication. The wires are called CAN high and CAN low.

The two logic levels are written on to the twisted pair as follows, logic 1 is represented by bus idle with both wires held halfway between 0 & VCC. Logic 0 is represented by both wires being differentially driven.

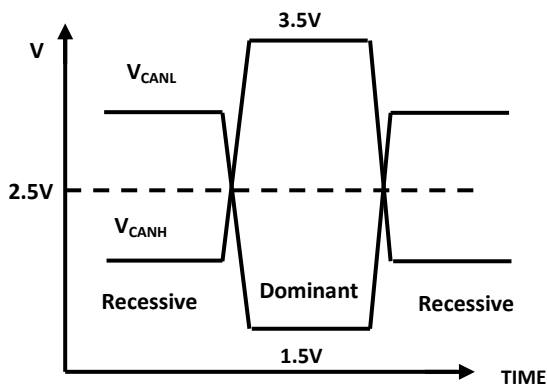


Figure 4: CAN-H & CAN-L Voltage Level

When the CAN bus is in idle mode, both lines carry 2.5V. When data bits are being transmitted, the CAN high line goes to 3.75V and the CAN low drops to the 1.25 V line as shown above. Since communication relies on a voltage differential between the two bus lines, the CAN bus is NOT sensitive to inductive spikes, electrical fields or Mbit/s.

IV. WORKING PRINCIPLE

Initially we are monitoring Temperature of the Industrial area with help of the temperature sensors which are connected to different channels of 10 bit ADC (channel.1:sensor,channel.2: sensor2, channel.3:sensor3) which is inbuilt for lpc2129 controller.

Those Temperature values which we have collected from the ADC are transmitted to the Controlling section board with the help of CAN engine. Before transmitting we need to initialize CAN-registers and start Transmitting the data from

CAN of the Temperature Sensor board to Controlling section Board.

Data values are transferred with CAN-H and CAN-L signals. Received Temperature Values from CAN of Sensor board are displayed on to the LCD by converting it to ASCII Character values. Initialization of CAN registers for Transmission as well as reception of data includes, Setting Frame structure by providing values for Identifier (11 bit/29 bit), data rate, enabling Priority control, enabling Interrupt control using Vector address for Interrupt. Here we are using data Frame and Remote Frame format for sending and receiving data.

Temperature sensor T-103 used here is a Thermistor with 10Kilo Ohms as a variable resistor with range of temperature as 10°C to 35°C. (49 °F to 95 °F).advantages of sensor used here are sensitivity, Accuracy, Low cost, flexible, surface mount etc..

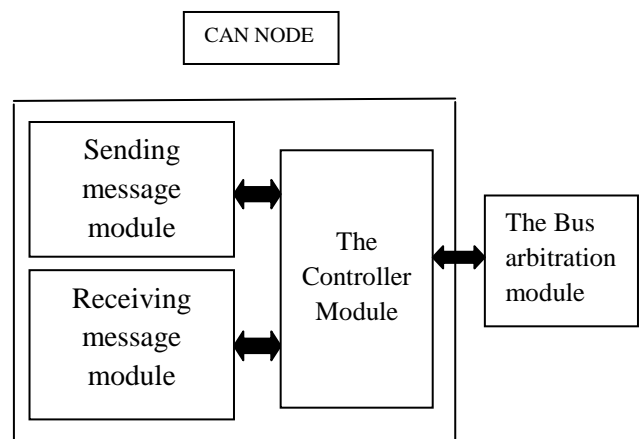


Figure 5: Communication Module

Communication using CAN Bus is achieved through Arbitration i.e. when the CAN Bus is free any node may start to transmit a message. If multiple nodes start transmitting at the same time the bus access conflict is resolved by bit-wise arbitration using the identifier. During arbitration every transmitting node compares the level of the bit transmitted with the level that is monitored on the bus. If these levels are equal, then the node will continue to transmit. When a recessive level bit is sent, but a dominant level is monitored, the node has lost arbitration and must withdraw without sending any further bits. Proposed scheme act as a multi-master i.e. when the bus is free any node may start to send a message. The highest

priority node will be successful in sending the message.

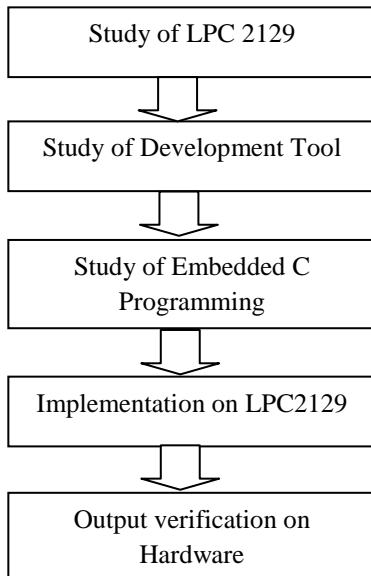


Figure 6: Flow Chart of scheme

V. SIMULATION RESULTS

Window shown below gives initialized Can Transmission registers.

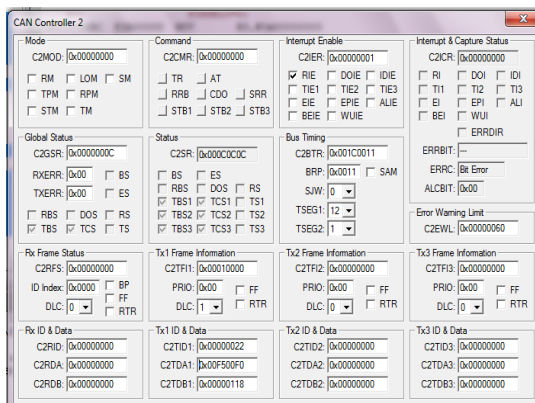


Figure 7: CAN register window by Keil software

Number	States	#	ID (Hex)	Dir	Len	Data (Hex)
10	794448572	2	022	Xmit	1	FO
11	868163539	2	022	Xmit	1	FO
12	941878506	2	022	Xmit	1	FO
13	1015593473	2	022	Xmit	1	FO
14	1089308440	2	022	Xmit	1	FO
15	1163023407	2	022	Xmit	1	FO
16	1236738374	2	022	Xmit	1	FO
17	1310453341	2	022	Xmit	1	FO
18	1384168308	2	022	Xmit	1	FO
19	1457883275	2	022	Xmit	1	FO
20	1531598242	2	022	Xmit	1	FO
21	1605313209	2	022	Xmit	1	FO
22	1679028176	2	022	Xmit	1	FO
23	1752743143	2	022	Xmit	1	FO
24	1826458110	2	022	Xmit	1	FO
25	1900173077	2	022	Xmit	1	FO
26	1973888044	2	022	Xmit	1	FO
27	2047603011	2	022	Xmit	1	FO
28	2121317978	2	022	Xmit	1	FO
29	2195032945	2	022	Xmit	1	FO

Figure 8: CAN Communication window

VI. EXPERIMENTAL VERIFICATION

This system real-timely test and control information which is passed by the Can bus. According to the control information, it collects multichannel temperature, then passes the temperature signal to controlling section by the Can bus. Experiments shows, this system has the advantage of collection temperature precise, high reliability, and communication distance. So it can be widely used in various industrial controls.

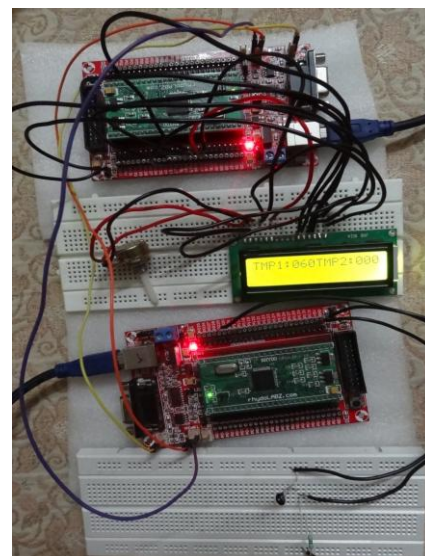


Figure 9: Implemented Hardware

VII. CONCLUSION:

The CAN protocols are adequate solutions for modular system design. The protocols guarantee deterministic communications for the modular system to supports real-time position and direction of the applied system without major disruption on the existing system. ARM7 Controller is the emerging technology now days and it is also a step to discover the latest technological progress in the field of Serial

Communication. Our project is a step put forward in this field.

REFERENCES:

- [1] Liu Wei, "The Design of Communication between Single Chips", 978-0-7695-4008-5/10 \$26.00 © 2010 IEEE
- [2] Sushil Thale, "CAN based Control of DC-DC Converters in Distributed Generation Units Operating in Master Slave Configuration" *2012 IEEE International Conference on Power Electronics, Drives and Energy Systems December 16-19, 2012, Bengaluru, India*
- [3] Chen Yueping, "Design and Realization of Fire Alarm System Based on CAN Bus", 1-4244-1135-1/07/\$25.00 ©2007 IEEE.
- [4] Thale Sushil and Agarwal Vivek, "Controller Area Network (CAN) based Smart Protection Scheme for Solar PV, Fuel Cell, Ultra-Capacitor and Wind Energy System based Microgrid," 38th Photovoltaic Specialists Conference (PVSC 2012), Austin, Texas, USA, 3- 8 June 2012.
- [5] Best, R.J., Morrow, D.J., Lavery, D.M. and Crossley, P.A., "Synchrophasor Broadcast Over Internet Protocol for Distributed Generator Synchronization," *IEEE Transactions on Power Delivery*, vol. 25, no. 4, pp. 2835 - 2841, Oct. 2010
- [6] H. F. Othman, "Controller Area Networks: Evolution and Applications" 2006 IEEE.
- [7] Mr. VIJAY BHAMARE, "Can Based Real Time Implementation In Automobile Using Arm", *(IJARCET) Volume 2, Issue 3, March 2013*
- [8] Mani kanth Alapati, "Design & Implementation of Automotive Intelligent Node CAN bus in Hybrid Electric Vehicles using ARM7", *(IJARECE) Volume 1, Issue 5, November 2012*
- [9] Fang Li, CAN (Controller Area Network) Bus Communication System Based on Matlab/Simulink, 978-1-4244-2108-4/08 © 2008 IEEE
- [10] Syed Misbahuddin, "Improving Inter-processor Data Transfer Rates over Industrial Networks", 1-4244-1031-2/07 ©2007 IEEE