**AUSTRALIAN JOURNAL OF BASIC AND APPLIED SCIENCES**

ISSN:1991-8178 EISSN: 2309-8414
Journal home page: www.ajbasweb.com

# Temperature dependence of dynamic viscosity of vegetable oils: argan, colza and sunflower

[1]Pardeep Kaur and [2]Chanpreet Kaur Toor

*[1]Student, Chandigarh Engineering College, Landran, VLSI Design (Electronics and Communication Engineering), Pardeep Kaur, Landran, Punjab, India.*
*[2]Assistant Professor, Chandigarh Engineering College, Landran , Electronics and Communication Engineering, Chanpreet Kaur Toor, Landran, Punjab, India.*

**Address For Correspondence:**
Pardeep Kaur, Student, Chandigarh Engineering College, Landran, VLSI Design (Electronics and Communication Engineering), Pardeep Kaur, Landran, Punjab, India.

**ARTICLE INFO**

***Article history:***
*Received 18 February 2017*
*Accepted 15 May 2017*
*Available online 18 May 2017*

***Keywords:***
*CAN- Controller Area Network, ID-Identifier, Message Scheduling, Deadline period.*

**ABSTRACT**

**Background:** Various algorithms are proposed for the arithmetic calculation of tough priority orders which solves the problem of extending accessible message sets and dynamic scheduling algorithm for increasing low priority needs. First of all limits are determined for the priorities of new messages. After this the most strong priority order that keeps the IDs of the accessible messages is computed. **Objectives:** In this paper CAN protocol is implemented on an FPGA. **Results:** The Controller Area Network (CAN) is a serial communication protocol. This protocol is invented primarily for connecting electronic control modules. These modules are used in automotive applications. It needs large levels of data incorruptibility and rates of data up to 1 Mega bits per second. The use of the CAN for automobiles communication needs finding appropriate and robust preference levels so every message transferred on the bus meets its specific deadline and bears potential communication errors. **Conclusion:** The problem of message scheduling is removed by making the message identifiers correspond to every message active in CAN protocol. It shows that the id window changes with every message.

## INTRODUCTION

The Controller area Network protocol is a message based standard developed by Bosch within the Eighties. The first development of CAN protocol was mainly in the transport industry which is originated in the form of vehicles such as spacecrafts and other vehicles of transportation. The CAN protocol is also broadly used in industrial and manufacturing automation. It has its applications in several goods like medical industry, weaving machines, and construction automation and production machinery. In these protocol square measures last two layers of open system interconnection (OSI) model and regulate the physical and electric circuit layers. For a lot of systems, higher-layer protocols square measures are need of economic development transformation and their operation. Such protocols are used in applications as a way to talk to the design of identifiers related to application messages. CAN bus is used in industrial and automotive applications. CAN protocol is used in almost every safety and economy feature of automobiles.

### *What Is Can:*

An electronic vehicle has a set-up of electronic devices to share information and data among themselves. An example is of a transmission control unit. It changes the ratio of gear automatically with the ratio of varying



**To Cite This Article:** Avneet Kaur and Chanpreet Kaur Toor., Temperature dependence of dynamic viscosity of vegetable oils: argan, colza and sunflower. ***Aust. J. Basic & Appl. Sci.,*** *11(8): 87-93, 2017*

speed. The engine control unit and various sensors in the system provide information to the transmission control unit. Each electronic device has an ECU/MCU (electronic/microcontroller control unit). ECU/MCU has its individual set of rules and regulations to transfer information. If two or more devices want to correlate they should have the required hardware and software so that they should communicate with each other. When CAN was not introduced in vehicles, each electronic device was linked to other gadgets by wires which worked well, the functions in the system were restricted at that time (Wu, Y.J., J.G. Chung, 2015). The figure describes the outlook of point to point connection of different nodes.

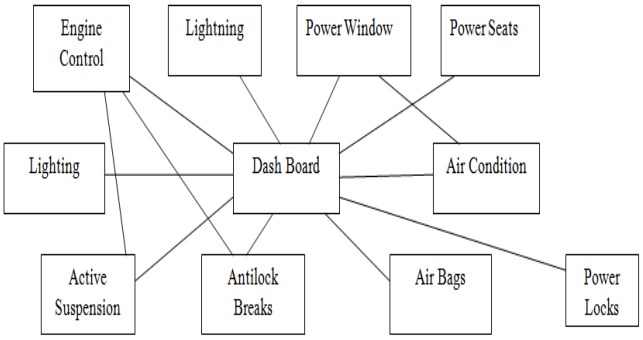


**Fig. 1:** P2P Wiring Connection

If automotive engineers want to exchange the real time information, the major problem was in linking the electronic control units of different devices. CAN protocol is considered to deal with this problem. The CAN protocol sets the rules and regulations through which the several electronic devices can exchange information with each other over a common serial bus. It decreases the point to point connections to great extent and in turn also reduced the largeness and intricacy of the system. Following figure shows connection of various tools in CAN protocol.

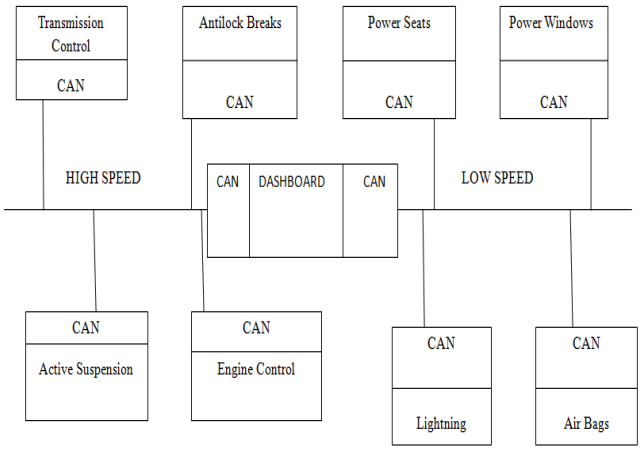


**Fig. 2:** Connection of Tools in CAN

***1) Architecture:***

CAN protocol is a multi-master serial bus. CAN is basically a standard for connecting Electronic Control Units. These ECU's are also called nodes. For the CAN network to communicate, two or more nodes are required. The ECU can be a simple input/output device or an embedded PC having a CAN interface. The node can also be an entry point for a standard computer which allows it to transmit data over a universal serial bus or to the tools on a CAN network by Ethernet port. The bus should be kept within a minimum and maximum common mode bus voltage according to the ISO specifications (Lee, H., *et al*., 2015). But these specifications do not explain the way to maintain this bus with in maximum and minimum range.

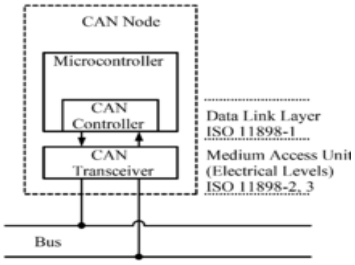


**Fig. 3:** CAN Bus Node

***2) Requirements of a node:-***

- Central processing unit, microprocessor: It is the task of host processor to decide what messages should be transmitted and what the received messages mean.
- All the control devices, sensors and actuators are linked to host processor.
- CAN controller is an essential part of the microcontroller.

***Receiving:***

The acknowledged serial bits from the bus are stored by CAN controller receiving. The acknowledged serial bits from the bus are stored by CAN controller until an entire message is available. This message is then fetched by the host processor.

***Sending:***

Message is transmitted to CAN controller by host processor. Then the CAN controller transmits the bits on bus when it is free. These bits are transmitted serially.

Transceiver: It Defines Medium Access Unit [MAU] standards

***Receiving:***

Data stream is converted by MAU from the CAN bus levels to the levels which are used by CAN controller. MAU has defensive circuit for the protection of CAN controller.

***Transmitting:***

The data stream from the CAN controller to CAN bus levels is converted here.

***3) Message formats:***

There are four types of message formats in CAN. These formats are information, remote, overload, and error frames. In this, only information frame is discussed. First bit is start-of-frame (SOF) bit. It is followed by eleven-bit identifier and the RTR bit. The symbol and the remote transmission request bit are part of the arbitration field. Management field has total six bits. Percentage of information bytes is described by this management field that must be included in data field. Length of knowledge frame is 0 to 8 bytes. There is cyclic redundancy verification field for checking communication errors. The 2 bit acknowledgment field is used to grant valid CAN frame. The last bit is end-of-frame bit.

| SOF | Identifier | RTR | Control | Data | CRC | ACK | EOF |
|---|---|---|---|---|---|---|---|

**Fig. 4:** Message Frame of CAN

***4) Arbitration:***

Arbitration is that the means which handles the disputes related to bus access. Every unit will initiate to broadcast a message when the bus is free. Possible disputes, as an outcome of quite one unit begin to send out at the identical time. Symbol is sent by the each unit which travels all the way through arbitration section. This symbol is then compared with the level that was monitored on the bus. If this comparison comes out to be equal, the entity continues to transmit. If an overriding level is found by the unit on the bus, whereas it was trying to launch a recessive level, then it equal transmission.

***Proposed Methodology:***

The problem in Controller area network is message scheduling and it can be resolved by making the identifiers dynamic corresponding to each message. This means that with each message, the id window alters.

These message identifiers are chosen from a predefined message identifier window. The assortment of message ID's from that window is also dynamic. This means that selection order is not fixed .So that any identifier which is requesting the bus access at random time can have random id. It happens only if they are not a part of emergency messages. Moreover this if any message belongs to emergency message group then it must be given right to use as soon as it is enable.

| | | | | | |
|---|---|---|---|---|---|
| Id1 | Id2 | Id3 | Id4 | Id5 | Id6 |

**Fig. 5:** Message Identifier Window

Figure 5 shows the message identifier window from where the ID's are selected randomly. These ID's are dissimilar for each application for arbitration.

***Pseudo code:***

1. Start
2. Allocate every application a distinctive message id from identifier window
3. Emergency messages are allotted with smallest id (decimal value)
4. Two or three applications enabled to test bus negotiation
   (Allotted ids operate as weights to the equation)

$$accessgrant = \sum_{i=1}^{N} w_i * application_i$$

5. Verify for weights for the preferred enables
6. $if(w_i < w_{i+1}) i \in N\{..\}$

(Covers both normal and emergency messages)

$w_i$ is approved to access the bus

7. $endif$
8. End

## RESULTS AND DISCUSSIONS

The worst case reaction time of the system to each message is given by

$$W_M^{n+1} = \min(\alpha, FixedId) + \sum \frac{W_M^n + \tau_{bit} + D_M}{T_M} \quad (1)$$

$$R_M = W_M^{n+1} + C_M \quad (2)$$

Here $R_M$ is the response time of worst case. $\alpha$ is the number of bits, $\tau_{bit}$ is the time mandatory to transfer 1 bit,$D_M$ is the deadline time and $T_M$ is the period of the message, $W_M$ is the step value for Message M and $C_M$ is the normalized bit value. Table 1 shows the considered values of period and deadline of the message. Table 2 shows the actual order of processing of the message based on the count of bits and the fixed or assigned id to each message.

**Table 1:** Period and Deadline of the Message

| Message M | M1 | M2 | M3 | M4 | M5 | M6 |
|---|---|---|---|---|---|---|
| $T_M$/ms | 10.0 | 5.0 | 5.0 | 10.0 | 10.0 | 10.0 |
| $D_M$/ms | 2.0 | 5.0 | 5.0 | 5.0 | 5.0 | 7.0 |

| Message M | M7 | M8 | M9 | M10 | M11 | M12 |
|---|---|---|---|---|---|---|
| $T_M$/ms | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 20.0 |
| $D_M$/ms | 8.0 | 10.0 | 10.0 | 10.0 | 10.0 | 15.0 |

Table 1 values are used to calculate the worst case response time using the formula given in equation 1 and 2. Firstly the step value for the message is calculated depending on the previous value and other parameters. Because CAN is message id specific not application specific. The step value is used along the normalized bit value to calculate the worst case response time as shown in table 2. So the worst case response time is directly dependent on the number of message bits or the message id assigned to the message. The response time value helps to decide the message priority which helps in the proper utilization of the CAN bus by not letting the other messages in queue for a longer period of time.

**Table 2:** Worst Case Response Time and Order of Message

| Message M | M1 | M2 | M3 | M4 | M5 | M6 |
|---|---|---|---|---|---|---|
| o(M) | 1 | 4 | 9 | 2 | 8 | 6 |
| $R_M$/ms | 0.13 | 0.26 | 0.58 | 0.18 | 0.55 | 0.51 |
| $\alpha$ [bit] | 128 | 132 | 420 | 158 | 512 | 412 |
| Fixed Id | 2 | 396 | 132 | 12 | 42 | 100 |

| Message M | M7 | M8 | M9 | M10 | M11 | M12 |
|---|---|---|---|---|---|---|
| o(M) | 11 | 3 | 12 | 5 | 7 | 10 |
| $R_M$/ms | 0.65 | 0.19 | 0.91 | 0.49 | 0.55 | 0.63 |
| $\alpha$ [bit] | 324 | 154 | 448 | 240 | 264 | 308 |
| Fixed Id | 1282 | 11 | 1803 | 204 | 266 | 452 |

Here, o (M) is order of the message, *a* [bit] is no. of bits in message id and $R_M$ is the response time. The order of message is checked from the response time calculated from the formula. As small is the response time, smaller is the order of message and that message with smaller order is assigned priority. The worst case response time is an important parameter in deciding the priority order of the messages to be delivered. As it depends on various different parameters like number of message bits and the fixed id assigned to each message, the response time is an essential parameter in deciding the access of the bus (Schmidt, K.W., 2014). The message with the lowest response time is assigned the top priority.

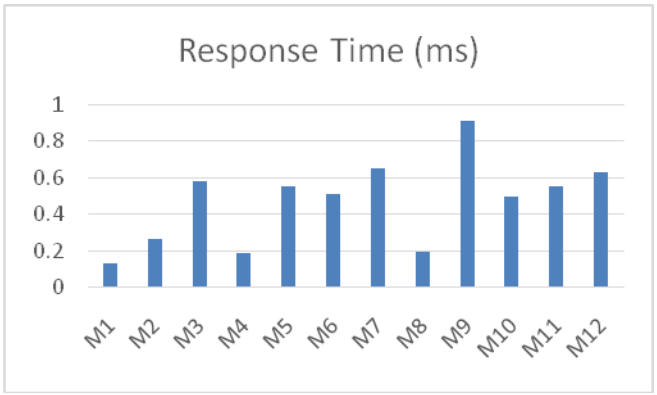


**Fig. 6:** Graph of Worst Case Response Time

Figure 6 shows the graphical results of the worst case response time for the proposed design. Accessing a bus and response time are inversely proportional. The lower is the response time of a message the higher the probability of selection to access the bus.

***Simulation Results:***

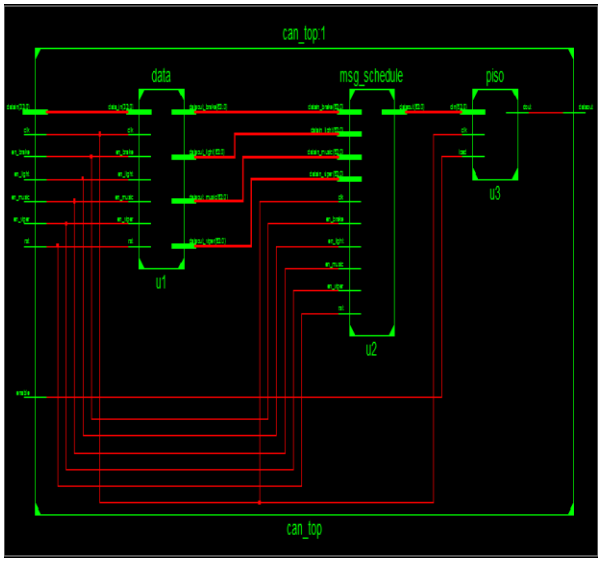


**Fig. 7:** CAN Top Module

The proposed method is simulated using the Xilinx Virtex4 FPGA in Xilinx ISim and the coding is completed in Xilinx ISE environment using Verilog language. Figure 6 describes the top module implementation of the proposed methodology. Message id attached with data is transferred in the form of application to message scheduling. Message scheduling extract the smaller message id and then it is transferred to PISO and figure 7 shows the simulation of the top module.

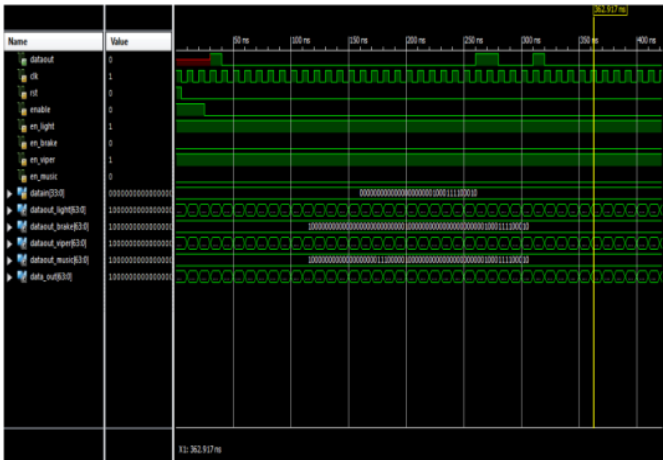

**Fig. 8:** Simulation of Top Module

**Table 3:** Resource Utilization

| Logic | Proposed |
|---|---|
| Number of Slices | 130 |
| Number of Slice Flip Flops | 129 |
| Number of 4 input Look up tables | 238 |
| Latency (ns) | 3.793 |

Resource Utilization is the number of resources which are used on Xilinx Virtex 4 FPGA by the implemented circuit. These resources are provided in stipulations of number of Slices used, number of LUTs and LUT- FF pairs and latency.

***Conclusion And Future Scope:***

Message id allocation and arbitration in CAN protocol is widely researched issue in recent years. Many id allocation techniques have been proposed and implemented on FPGA. The message id allocation is both static and dynamic. In case of static the same id is allocated to every message of a particular application which allows arbitration to be biased. In the present work a window of message ids is used and ids are selected randomly from that window for granting the bus access to the application. Worst case response time of each message is also calculated so as to optimize the bus usage and prioritize the messages based on their ids. In future the methodology must be tested for other emergency related applications.

## REFERENCES

Al Saadi, I.M., Dynamic Message Transmission Scheduling Using Can Protocol.

Barnola-Sampera, M., D. Heredero-Peris, R. Villafafila-Robles, D. Montesinos-Miracle, J. Bergas-Jane, N. Vidal-Tejedor, 2014. Charging/discharging process for electric vehicles: Proposal and emulation. InElectric Vehicle Conference (IEVC), 2014 IEEE International., pp: 1-8.

Bossa, J.L., G.A. Magallán, C.H. De Angelo, G.O. García, 2010. Implementation of a Supervisory Control System for an Electric Vehicle. InIndustry Applications (INDUSCON), 2010 9th IEEE/IAS International Conference, 8: 1-5.

Chang, P.L., Y.X. Guo, M.D. Jeng, 2011. Telematics gateway and power saving method for electric vehicles. InSystems, Man, and Cybernetics (SMC), 2011 IEEE International Conference, pp: 780-785.

Hu, J., H. Li, X. Sheng, K. Cao, F. Yan, 2012. Development and Research of Vehicle Fault Diagnosis System. InPower and Energy Engineering Conference (APPEEC), pp: 1-5.

Hui, D., H. Bo, W. Dafang, Z. Guifan, 2011. The ECU control of diesel engine based on CAN. InIntelligent Computation Technology and Automation (ICICTA), 2011 International Conference on 1: 734-736.

Lee, H., K. Choi, K. Chung, J. Kim, K. Yim, 2015. Fuzzing CAN Packets into Automobiles. InAdvanced Information Networking and Applications (AINA), 2015 IEEE 29th International Conference, pp: 817-821.

Li, Y., C. Yu, J. Yan, H. Li, 2010. The design of ZigBee communication convertor based on CAN. InComputer Application and System Modeling (ICCASM), 2010 International Conference, 6: V6-245.

Nakamura, M., M. Ohara, A. Saysanasongkham, M. Arai, K. Sakai, S. Fukumoto, 2014. Hybrid ARQ for DC-DC Converter Noise in Controller Area Networks. InParallel Processing Workshops (ICCPW), 2014 43rd International Conference, pp: 375-379.

Schmidt, K.W., 2014. Robust priority assignments for extending existing controller area network applications. IEEE Transactions on Industrial Informatics., 10(1): 578-85.

Sharma, M., N. Prashar, 2015. A Novel Approach for Dynamic Message ID Allocation in Application Specific can Architecture. International Journal of Signal Processing, Image Processing and Pattern Recognition., 8(8): 169-76.

Shokry, H., M. Shedeed, S. Hammad, M. Shalan, A. Wahdan, 2009. Hardware EDF scheduler implementation on controller area network controller. InDesign and Test Workshop (IDT), 2009 4th International., pp: 1-6.

Sun, H., H. Zeng, J. Guo, 2011. Bus data acquisition and romote monitoring system based on CAN bus and GPRS. InConsumer Electronics, Communications and Networks (CECNet), 2011 International Conference, pp: 1094-1097.

Wu, Y.J., J.G. Chung, 2015. Efficient controller area network data compression for automobile applications. Frontiers of Information Technology & Electronic Engineering, 16(1): 70-8.

Yun, D.S., J.W. Lee, S.K. Lee, O.C. Kwon, 2012. Development of the eco-driving and safe-driving components using vehicle information. InICT Convergence (ICTC), 2012 International Conference, pp: 561-562.

Zhao, S., N. Li, 2012. Chassis dynamometer for hybrid electric vehicle based on controller area network. InIntelligent Control and Information Processing (ICICIP), 2012 Third International Conference, pp: 352-354.

Zhu, M., X. Wu, K. Wu, J. Zhou, 2011. Development of measurement and control system for vehicular powertrain testbench in HEV applications. InRemote Sensing, Environment and Transportation Engineering (RSETE), 2011 International Conference, pp: 3063-3067.

Zuo, Y.H., C.L. Xiang, Q.D. Yan, W.D. Wang, H. Liu, H.C. Li, 2010. Distributed Communication System Design Based on Can Bus for Parallel-Serial Hybrid Electrical Vehicles. InInformation Science and Applications (ICISA), 2010 International Conference, pp: 1-8.