IOSR Journal of Engineering (IOSRJEN)
ISSN: 2250-3021 Volume 2, Issue 6 (June 2012), PP 1374-1381
www.iosrjen.org

Plug and Play CAN-A Time Triggered Approach

Venkataraman Padmaja[#1],

Dept. of Electronics & Telecommunication, PG Student Maharashtra Academy of Engineering,Alandi Pune University of Pune

***Abstract**: — The use of networks for communication between the Electronic Control Units (ECU) of a vehicle in production cars dates from the beginning of the 90s. The specific requirements of the different car domains have led to the development of a large number of automotive networks such as LIN, J1850, CAN, FlexRay, MOST, etc.*

*Designing a CAN network is a lengthy task due to required modifications in the software and hardware and then validating the system. To solve this issue, it is proposed to develop a complete new protocol layer for the first time to have Plug and Play capability in CAN architecture. This is the additional application layer proposed on top of the CAN layer. Paper also discusses possibility of implementing this Plug and Play system with time triggered mixed planning based scheduling for real time applications. Finally one proposed test case is discussed to evaluate performance of this solution on the field.***

***Keywords**: - CAN, Communication, Plug and Play*

I. INTRODUCTION

The safety, comfort and performance requirements and thus the functions of in-vehicle systems have been increasing steadily. As a result, there has been an increase in the number of electronic control units (ECU) and communication signals with more complex interrelations between them to meet the requirements. This result reveals the need for more robust, dependable and efficient high-speed in-vehicle communication. Such systems contain hard real-time messages that have strict timing requirements. The exchange of these messages in the network is conducted by the in-vehicle communication protocols that can be classified as event-triggered, time-triggered and hybrid networks. These networks are expected to schedule real-time messages to provide timeliness in communication for a healthy run of the system.

Today automotive electronic systems contain electronic control units, sensors and actuators that are distributed and embedded in vehicles. The use of such systems is increasing as mechanical and hydraulic parts are replaced or new functions are added to them. Most of these are real-time systems that possess strict timing requirements in terms of deadlines and response time jitter. For instance, in modern cars nearly 2500 signals are exchanged by up to 70 electronic control units both of which tend to increase with higher demands on safety, comfort, functions and cost. Electronic control units, referred to as the main processing units, of automotive systems form several networks that have different properties, regarding their architectures, services and functions depending on communication requirements. One of the most important requirement is the providing these networks with the integrity and interoperability.

FTTCAN networks

Flexible real-time systems are capable of operating in dynamic environments, i.e., those where a complete specification is not possible and where operational requirements may change during system lifetime. However, flexibility in real-time systems must be achieved without jeopardizing the understandability and predictability of the system temporal behaviour allowing safe and predictable upgrades with negligible down-time. Two important properties to support such flexibility in composability (the subsystems' properties remain valid upon integration in the whole system) and scalability (the extensibility of the system is not limited by some predefined upper limit). In distributed real-time systems, the communication system has a central role in determining either the composability or scalability of a distributed architecture.

With the planning scheduler, the table is rebuilt every fixed duration period of time called a plan. Changes to the message set can be easily accommodated in the next plan thus improving the level of system flexibility. Each plan is divided into a fixed number of constant duration elementary cycles (Ec). All time attributes of messages are expressed as integer multiples of the duration of one Ec. Consequently, within each of these cycles, each message can be transmitted only once. The duration of the Ec determines the temporal resolution of the communication system.

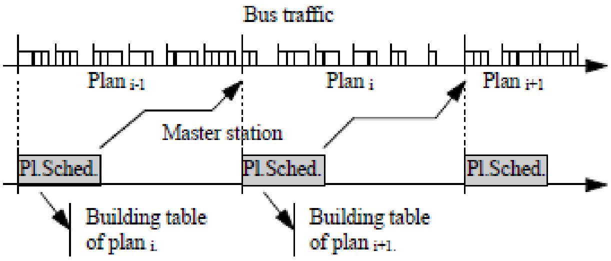


Fig.1.The Planning Scheduler

Given this temporal resolution, any transmission delay within the duration of one Ec is irrelevant. Thus, the order by which messages are transmitted within one Ec is also irrelevant. When realizing centralized scheduling in CAN, the master sends a special control message, the master message, in the beginning of each Ec. This message indicates which data messages must be transmitted during that particular Ec. The order by which the data messages are transmitted is left for the CAN native distributed arbitration to sort out. It is called centralized scheduling, and not arbitration, because it is the scheduling activity that is centralized while the arbitration is still distributed.

The main advantages of such a communication system are:

- Compromise between flexibility and run-time overhead achieved by the planning scheduler
- Implicit synchronization of all nodes using the regular transmission of the master message;
- Simplicity of the slave nodes' network interface;
- Allows messages' phase control in the scheduling table in order to improve the scheduler's performance and stability of messages periods;
- Presents an early detection of missed deadlines and allows the use of mixed scheduling in order to maintain stability under transient overloads.

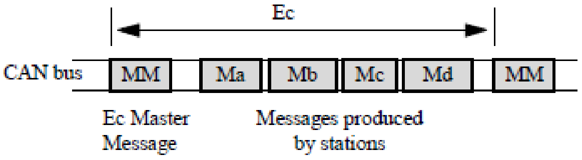


Fig 2.Message transmitted during one Ec

Flexible time-triggered CAN (FTTCAN) is an alternative protocol, which makes use of the dual-phase elementary cycle concept to combine time and event-triggered communication with temporal isolation. Moreover, the time-triggered traffic is scheduled online, facilitating the online admission control of dynamic requests for periodic communication because the respective requirements are held centrally in just one local table. With online admission control, the protocol supports the time-triggered traffic in a flexible way and guarantee timeliness, which is so-called dynamic planning-based scheduling paradigm.

Dynamic planning-based scheduling

Unlike in TTCAN, Dynamic planning-based scheduling paradigm of FTTCAN is implemented according to consecutive planning–based elementary cycles (ECs) instead of a static system matrix based on TDMA.

In FTTCAN, the bus time is slotted in consecutive ECs with fixed duration (TEc time units) that are initiated by a special message, the trigger message (TM) sent by a specific node called the master node. To support both periodic and sporadic traffic each EC is divided into two parts, asynchronous window and synchronous window. In the former, a set of sporadic messages are sent using CAN’s native arbitration mechanism. In the latter, traffic is sent according to the schedule delivered in the TM. That is TM contains in its data field the schedule for the synchronous traffic (time-triggered traffic) that shall be sent within this EC. The schedule for each EC is calculated and planned online by the master node with a planning-based table, synchronous requirements table (SRT).

A mixed planning-based scheduling (MPS)

The RM algorithm is probably the most used priority assignment in real-time applications, because it is very easy to implement. Indeed, an RM scheduler can be implemented just by assigning each task a fixed priority level in reversely proportional to its period, so the message with the smallest period receives the highest priority. DM algorithm is companion piece of RM, which scheduler can be implemented just by assigning each task a fixed priority level inversely proportional to its deadline, so the message with the smallest deadline receives the highest priority. Dynamic mixed planning-based scheduling (MPS) policy based on FTTCAN, which employs RM for the synchronous window and DM for the synchronous window to resolve eventual collisions the schedulability of the MPS, network performance analysis under the MPS is absolutely necessarily.

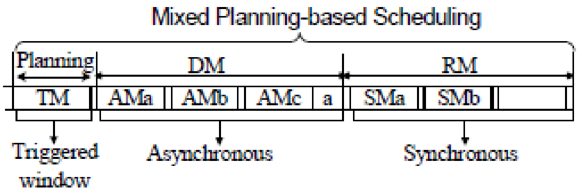


Fig.3. Dynamic mixed planning based scheduling policy

RC4 algorithm is implemented in software, so the complexity is less and it is cheaper as the software can be easily changed according to the requirements.

II. METHODOLOGY

In the currently running time triggered network, all the messages are configured to run at specific time. So the complete scheduling of the network fails if we want to change any node, add new node or delete any node. This leads to redesign a complete network with new message structure and also with new scheduling; followed by complete retesting of the vehicle performance. This results in investment of time and money. The average development time for addition of new module to a network is several months and the cost of thousands of dollars including development and validation.

Advantages of having plug and play included

- Addition/Deletion of CAN module do not result in redesigning of network, saving cost and time
- Allows to do immediate experimentation and also quick optimization of the network

A completely new protocol layer proposed to have Plug and Play capability in CAN. This is the additional application layer on top of the CAN layer.

The protocol layer is implemented with FTTCAN MPS (Mixed Planning Based Scheduling) for hard real time applications and to support the Plug and Play capability. As the FTTCAN MPS scheduling calculated scheduling at every elementary cycle, adding and deleting the node becomes very much simpler to implement.

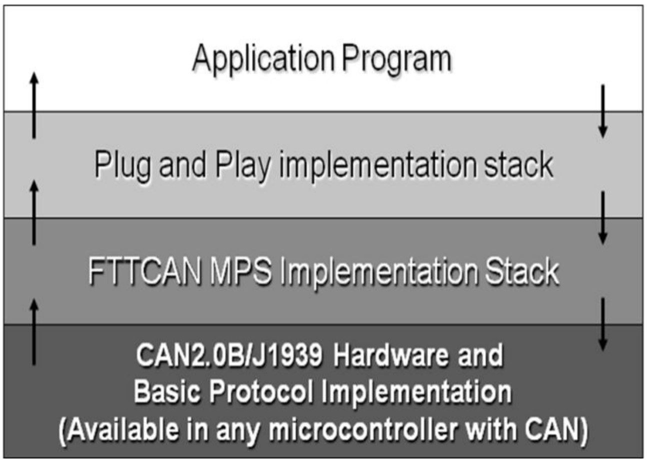

Application Program
Plug and Play implementation stack
FTTCAN MPS Implementation Stack
CAN2.0B/J1939 Hardware and Basic Protocol Implementation (Available in any microcontroller with CAN)

Fig 4. Conceptual Software Block diagram

The plug and play and other scheduling protocol is embedded in the CAN message data. The predefined CAN message ID represents all the Plug and Play related messages in its data.

This is the new concept with pseudo Master Slave arrangement, which allows the plug and play configuration. I.e. any new network node can be easily plugged in to the network without redesigning the network.

The master-slave mode is only to maintain the nodes in the system and for diagnostic purpose. But in general operating condition, the system works as a multi master mode, in which any node can send the data to any of other node in the network.

The master node decides the dynamic scheduling in the network based in time slots for synchronous and asynchronous messages. Maintenance and diagnostic port can be connected to the pseudo-master node, which maintains the required lists needed for the scheduling, plug and play and diagnostic related information.

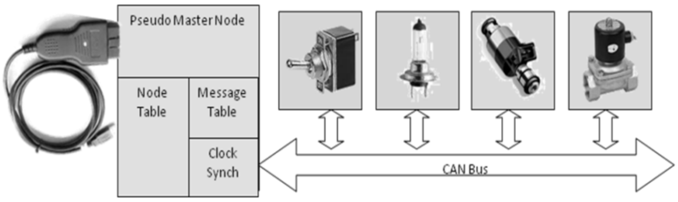


Fig 5. Block diagram of the test system

.

The Software Protocol

The software protocol is based on the FTTCAN MPS algorithm. The basic implementation of this algorithm gives the facility to schedule the asynchronous and synchronous messages very effectively as per the time schedule which is as per DM and RM scheduling. The main advantage is that the message timing is calculated at every elementary cycle (EC). This gives the complete flexibility in changing the message schedule in runtime, which is very important for the plug and play operation.

To enable the Plug and Play capability, few file structures and messages need to be added. This forms an additional Plug and Play implementation stack on top of the FTTCAN MPS implementation stack. FTTCAN MPS stack is based on the basic CAN hardware driver implementation.

The CAN communication is completely based on messages. So for any additional operation, we need additional messages. It is very much difficult to add different messages for the plug and play operations. So only one dedicated message can be added for completely new stack implementation. The type of plug and play operation can be part of the data byte of the dedicated CAN message.

Plug & Play ID	Plug & Play Message Type	Data

Fig 6. Plug and Play CAN Message Structure

There are different types of data structures implemented for the Plug and Play operations in order to have the fail safe operation and have the complete diagnostic information related to network.

Node Information File - The node information file is used for defining a node in the network. The file is the exclusive property of a node, and has the identification and message information. The node information is used in diagnostic and scheduling information to identify the messages and error location.

Message Information File – The message information file is the exclusive property of the node and has complete list of the messages, their trigger (event/periodic) and timing information.

Schedule Information File - This file stores the information about the schedule in the network. The schedule consists of various operations for transferring data from source (publisher) to various destinations (subscriber).

Master also maintains two tables

Node Table – This is the list of all the nodes connected in the network and are active

Message Table – This is the list of messages along with their timing information for all the active nodes listed in the node table.

Master creates the schedule file using the FTT-MPS scheduling algorithm with DM (deadline monotonic) for asynchronous and RM (rate monotonic) for synchronous messages.

III. IMPLEMENTATION

There are two different modes of operation; Operational Mode and Maintenance mode. The operational mode is the normal execution mode. The maintenance mode is used to add and delete the node from the network.

In the conventional network it is not possible to do so as all the messages and their timing are hardcoded and the schedule is done as per that hardcoded messages. Any addition and deletion of the node do not affect this schedule.

There are two basic operations in plug and play, adding a node to the network and deleting the node from the network.

Adding a node is accomplished by following sequence of operations

1. Enter to the maintenance node
2. Ask for the new node information broadcast
3. New node sends the node information and message information file to pseudo-master node
4. Add entry to the node and message table
5. Recalculate the scheduling information including new node
6. Send the scheduling information to all the nodes to start the operation.

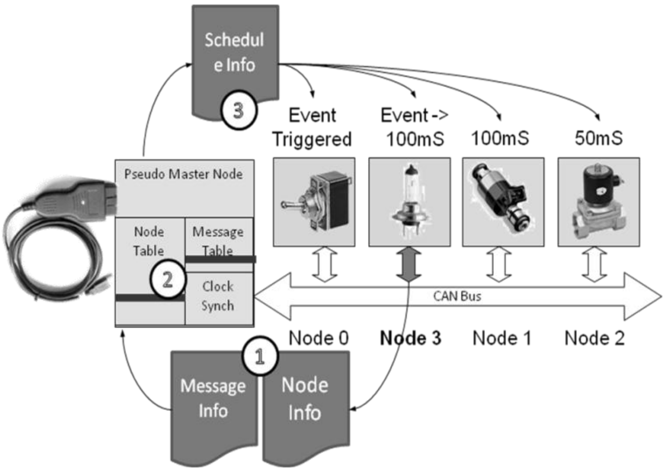


Fig 7. Node Addition (1 – Get Node and message information from added node, 2 – Make entry in the node table and message table, 3 – Send the recalculated schedule info to all the nodes)

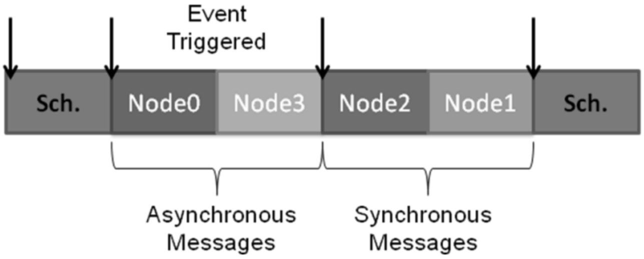


Fig 8. Node Addition - Message Schedule

Deleting a node is a runtime operation and do not need the network to be in the maintenance mode. This is very important as the damaged or deleted node need to be removed immediately from the network to avoid bandwidth loss. This is accomplished by following sequence of operations

1. Master keeps track on the messages on the network
2. If any message not received in the given slot for certain duration, master removes its entry from the schedule file and recalculates the schedule
3. New schedule information is sent to all the nodes to update their message transmission timing
4. When the network goes in the maintenance mode, master rescans all the nodes and then removes the faulty / deleted node entry from the node table and message table
5. Recalculate the scheduling information removing deleted node
6. Send the scheduling information to all the nodes to start the operation.

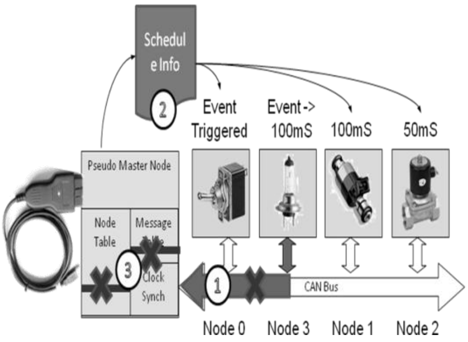


Fig 9. Node Deletion (1 – Node message missing at the defined time indicating faulty/disconnected node, 2 – Update the schedule info file and resend to all other nodes, 3 – In the maintenance mode, remove the deleted node entry from node table and message table)

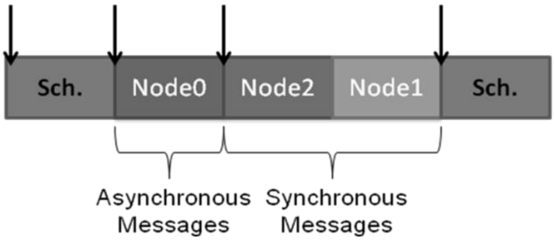


Fig 10. Node Deletion - Message Schedule

The figure 8 and 10 shows the effect of the plug and play node addition and deletion on the timing of messages.

In the conventional system, this message schedule and timing diagram cannot be modified and does not change even if you add or delete any node in the network.

Test Setup

To evaluate the Plug n Play system we can use a 'benchmark' network model of a prototype Traffic Light System. The network model describes a set of signals sent between three different nodes. The set of signals provide a good example to illustrate the application of CAN bus as in Fig to complex distributed real-time control systems.

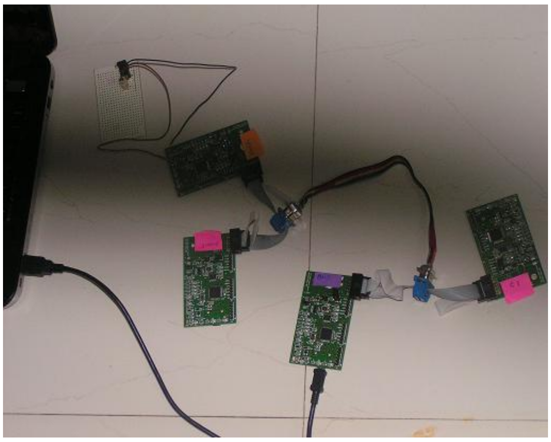


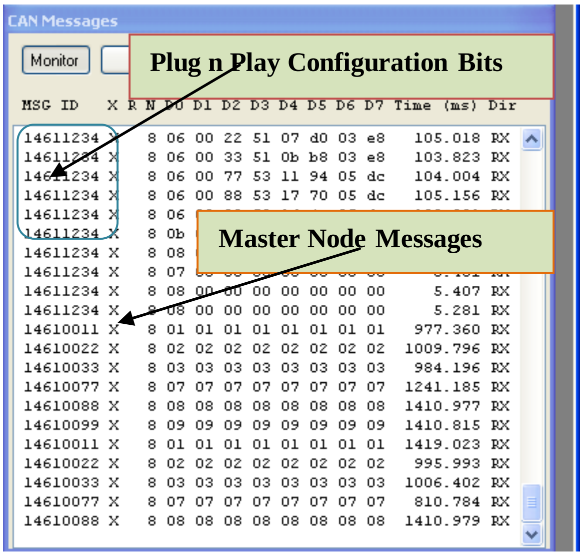
Fig.10. Test Setup

Figure above shows the connection of Master with 2 clients and Bus Monitor. Each module will be triggering 3 messages. Message from every module has the deadline assigned to it with respect to previous message. Any one node changes its timing ie add or remove, the other node understands when the load button of Master is pressed. Master request for the message information from all the nodes connected in the network and creates the schedule file based on each message from all the nodes. The schedule file is downloaded in all the nodes which give them information when they have to send the message.

If a new client is connected we don't have to change any code for the schedule, just press the master load button to acknowledge the addition on the node. Schedule is recalculated automatically including the new node and the trigger adjusted based on the new node addition. Same scenario repeats in the node remove condition also wherein the scheduling adjusted as per the current client connected. This will happen only when the Master load button is pressed or else the same old schedule will be repeated. Once the Load button is pressed, system acknowledges the removal of client.

With this protocol, any client can be added or removed from the network without distributing the entire setup and the schedule gets adjusted automatically without any change in code provided they follow same message structure as we have defined in the protocol

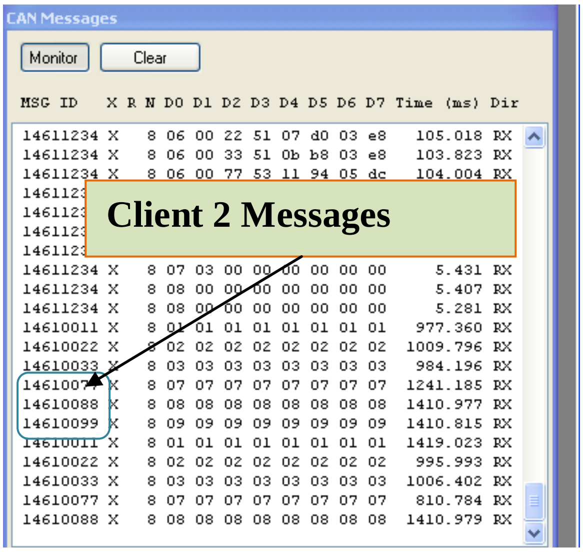
Test Results:



```
MSG ID    X R N D0 D1 D2 D3 D4 D5 D6 D7 Time (ms) Dir

14611234 X   8 06 00 22 51 07 d0 03 e8   105.018 RX
14611234 X   8 06 00 33 51 0b b8 03 e8   103.823 RX
14611234 X   8 06 00 77 53 11 94 05 dc   104.004 RX
14611234 X   8 06 00 88 53 17 70 05 dc   105.156 RX
14611234 X   8 06
14611234 X   8 0b
14611234 X   8 08
14611234 X   8 07
14611234 X   8 08 00 00 00 00 00 00 00     5.407 RX
14611234 X   8 08 00 00 00 00 00 00 00     5.281 RX
14610011 X   8 01 01 01 01 01 01 01 01   977.360 RX
14610022 X   8 02 02 02 02 02 02 02 02  1009.796 RX
14610033 X   8 03 03 03 03 03 03 03 03   984.196 RX
14610077 X   8 07 07 07 07 07 07 07 07  1241.185 RX
14610088 X   8 08 08 08 08 08 08 08 08  1410.977 RX
14610099 X   8 09 09 09 09 09 09 09 09  1410.815 RX
14610011 X   8 01 01 01 01 01 01 01 01  1419.023 RX
14610022 X   8 02 02 02 02 02 02 02 02   995.993 RX
14610033 X   8 03 03 03 03 03 03 03 03  1006.402 RX
14610077 X   8 07 07 07 07 07 07 07 07   810.784 RX
14610088 X   8 08 08 08 08 08 08 08 08  1410.979 RX
```

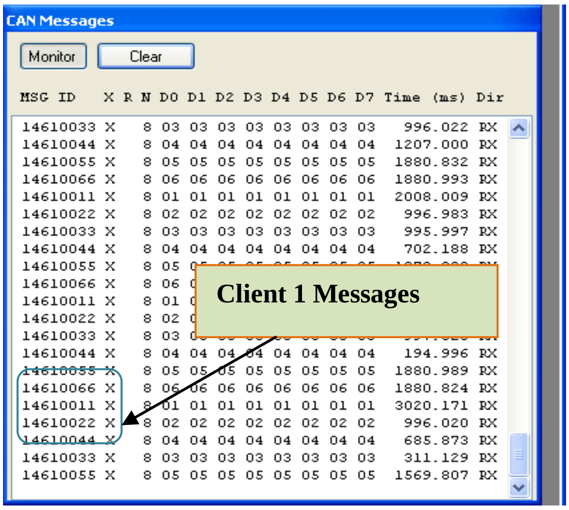
Fig.11 the output of the system with Plug n Play configuration bits messages and the Master node messages.



```
MSG ID    X R N D0 D1 D2 D3 D4 D5 D6 D7 Time (ms) Dir

14611234 X   8 06 00 22 51 07 d0 03 e8   105.018 RX
14611234 X   8 06 00 33 51 0b b8 03 e8   103.823 RX
14611234 X   8 06 00 77 53 11 94 05 dc   104.004 RX
146112
146112
146112
146112
14611234 X   8 07 03 00 00 00 00 00 00     5.431 RX
14611234 X   8 08 00 00 00 00 00 00 00     5.407 RX
14611234 X   8 08 00 00 00 00 00 00 00     5.281 RX
14610011 X   8 01 01 01 01 01 01 01 01   977.360 RX
14610022 X   8 02 02 02 02 02 02 02 02  1009.796 RX
14610033 X   8 03 03 03 03 03 03 03 03   984.196 RX
14610077 X   8 07 07 07 07 07 07 07 07  1241.185 RX
14610088 X   8 08 08 08 08 08 08 08 08  1410.977 RX
14610099 X   8 09 09 09 09 09 09 09 09  1410.815 RX
14610011 X   8 01 01 01 01 01 01 01 01  1419.023 RX
14610022 X   8 02 02 02 02 02 02 02 02   995.993 RX
14610033 X   8 03 03 03 03 03 03 03 03  1006.402 RX
14610077 X   8 07 07 07 07 07 07 07 07   810.784 RX
14610088 X   8 08 08 08 08 08 08 08 08  1410.979 RX
```

Fig. 12 the output of the system with Client 2 node messages



```
MSG ID    X R N D0 D1 D2 D3 D4 D5 D6 D7 Time (ms) Dir
14610033 X   8 03 03 03 03 03 03 03 03   996.022 RX
14610044 X   8 04 04 04 04 04 04 04 04  1207.000 RX
14610055 X   8 05 05 05 05 05 05 05 05  1880.832 RX
14610066 X   8 06 06 06 06 06 06 06 06  1880.993 RX
14610011 X   8 01 01 01 01 01 01 01 01  2008.009 RX
14610022 X   8 02 02 02 02 02 02 02 02   996.983 RX
14610033 X   8 03 03 03 03 03 03 03 03   995.997 RX
14610044 X   8 04 04 04 04 04 04 04 04   702.188 RX
14610055 X   8 05
14610066 X   8 06
14610011 X   8 01
14610022 X   8 02
14610033 X   8 03
14610044 X   8 04 04 04 04 04 04 04 04   194.996 RX
14610055 X   8 05 05 05 05 05 05 05 05  1880.989 RX
14610066 X   8 06 06 06 06 06 06 06 06  1880.824 RX
14610011 X   8 01 01 01 01 01 01 01 01  3020.171 RX
14610022 X   8 02 02 02 02 02 02 02 02   996.020 RX
14610044 X   8 04 04 04 04 04 04 04 04   685.873 RX
14610033 X   8 03 03 03 03 03 03 03 03   311.129 RX
14610055 X   8 05 05 05 05 05 05 05 05  1569.807 RX
```

Fig 13.the output of the system with Client 1 node messages

IV.CONLUSION

The proposed system is implemented to give a new direction in the CAN communication system and will make the Plug and Play application for time triggered CAN more beneficial for the automotive and industrial applications reducing the time and cost of the redevelopment / maintenance.

The system gives various advantages due to its implementation strategy, layered approach and pseudo master concept. These advantages include but not limited to

- Plug and Play Capability
- Remote Configuration Capability
- Maintenance Capability
- Highest Predictability - Due to time triggered architecture, the timing and node participation is known in advance. This gives highest predictability in terms of data transmission, inter-communications delays, cycle time and data latency.
- Lowest Data Redundancy - Due to known data transmission and standard proven hardware for interfacing, frame packaging and error handling overhead are very less, giving low data redundancy.
- Low jitter - Due to predefined timings and schedule, intercommunication gap is known and small. This gives small jitter in the communication.
- Continuous status monitoring - in order to keep the safety, reliability and correct communication within the cluster, it is very essential that all the nodes which are the part of the schedule are active and not faulty all the time of communication. To ensure that, a continuous status monitoring is essential to mark the non working nodes and inform this to other nodes in order to take the necessary action

This is definitely a upgraded approach towards faster network design and modification, and will accelerate the network design activity reducing time and cost.

REFERENCES

[1] http://www.can-cia.de, 2004. Homepage of the organization CAN in Automation (CiA).

[2] CAN specification version 2.0. Robert Bosch GmbH, Stuttgart, Germany, 1991.

[3] Leen, G., Heffernan, D.: TTCAN: a new time-triggered controller area network. Microprocessors and Microsystems 26, 2002, pp.77-94

[4] Almeida, L., Pedreiras, P., Fonseca, J.A.G.: The FTT-CAN protocol: why and how. Industrial Electronics, IEEE Transactions on 49, 2002, pp.1189-1201

[5] Plug and Play CAN Uday Wali

[6] A Real-Time Planning-based Scheduling Policy with CAN for Automotive Communication Systems Wanke Cao1, Cheng Lin1, Wei Zhou2, Fengchun Sun1

[7] Message routing in multi-segment FTT networks *www.cse.wustl.edu/~cdgill/WPDRTS04/.../Mesg-rout- ftt-nets-V3.ppt*

[8] VERIFYING REAL-TIME PROPERTIES OF CAN BUS http://people.cs.aau.dk/~apr/ITVDRTS06/Seminar2/Hanzalek04a.pdf

[9] http://www.st.com/internet/mcu/product/215105.jsp

[10] Basic in-application programming example using the STM8 I2C and SPI peripherals http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/APPLICATION_NOTE/CD00190266.pdf

[11] Getting started with the STM8S and STM8A http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/APPLICATION_NOTE/CD00194637.pdf

[12]http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010406

[13]http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010405