Distributed Embedded Control System using Time Division Scheduling for Implementing Internet of Things to Driverless Car (AutonomousVehicle)

S. Sakthivel Manikandan and S. Vasantharathna

Technical Specialist, Department of Engine Management System, Robert Bosch Engineering and Business Solution Limited, Coimbatore, India.
Professor and Head of the Department, Department of Electrical Engineering, Coimbatore Institute of Technology, Coimbatore, India.

ABSTRACT
Background: The World Health Organization (WHO) has stated in its first ever Global Status Report on Road Safety that speeding in vehicles is one of the main contributing factors in road accidents. This paper presents a solution to reduce the number of collisions in the road with the help of distributed embedded control system model. Both time triggered and event triggered processing are combined in this model. Distributed computing platform along with Internet of Things is implemented for a driverless autonomous car. This computing environment uses time division scheduling mechanism which is interfaced with web is discussed. The model is experimented with real time hardware setup consisting of CAN nodes, sensors, actuator and connectivity to web. The results from experiments show that this computing environment can be applied to a driverless car which can be monitored and controlled over web. Further the smart phone with APPS of this web application can enable us to control the speed of the vehicle if it seems to be abnormal, thus avoid accidents due to manual driving.

INTRODUCTION
Distributed computing can be described as a mechanism to unite the computers connected in a network in such a way that the information or resource can be shared by all the computers. The embedded distributed control systems have a very high usage in the various domains like automotive, industrial automation, robotics, building control and so on. Most of the embedded distributed control systems are real time systems. In general, real time system can be designed in two aspects, an event-triggered approach and time-triggered approach (Kopetz, H., 1993). The event-triggered architecture is basically of a dynamic system that responds to the sensor signal or any other external events. But the time triggered architecture is a static design that responds periodically as programmed before. In hard real time systems the time-triggered architecture is preferred over event-triggered since the hard real time system is a time critical system which should not have jitter. However in the case of automotive domain and other embedded control system, there is a need for mixed triggered, (i.e.) periodically operated time triggered operations and dynamically operated event triggered operations.

In this paper for the automotive engine control the mixed architecture distributed embedded computing is applied. This control is experimented in a driverless car which is also called as autonomous vehicle where the car is automatically driven with the help of ultrasonic sensors, RADAR, LIDAR and camera connected. The system presented by Hattori et al. (Hattori, H et al, 2006) can manage both the time triggered and the event triggered operations. So there is a need of efficient and optimal performance operating system for the mixed architecture. A distributed computing environment based on time-division scheduling was presented to run the time-triggered and event-triggered process simultaneously (Tasuku Ishigooka, 2007). Yuichi Itami et al. had presented two kinds of event-triggered object models, a pure event triggered distributed object model and a data triggered distributed object model (Yuichi Itami, 2008).

The rest of the paper is organized as follows. Section II presents autonomous vehicle overview. In Section III, distributed embedded computing model is explained. Section IV gives the overview of Internet of Things. In Section V, IoT based distributed computing is explained. Finally in Section VI, the implantation of model is presented with results and Conclusions as last section.

Corresponding Author: S. Sakthivel Manikandan, Department of Engine Management System, Robert Bosch Engineering and Business Solution Limited, Coimbatore, India.
Tel:+919994563111; E-mail:sakthivelmanikandan.sundharam@in.bosch.com
II. Autonomous Vehicle Overview:

There is a growing interest in driverless vehicles and the attention is often on long-distance use (e.g. motorways where vehicles can drive in group, automatic speed adaptation to conditions on the road ahead, etc.) However one of the key expectations could be shorter journeys between home, office, and public transport hub (rail station or airport). There is also growing interest in "self service" hire, starting with bicycles and extending to "city cars". If the driverless car concept combined with "self service", people can be traveled more efficiently, often at a lower cost. The autonomous vehicle can be designed with the help of the ultrasonic sensors and the GPS. The GPS plays a major role in the navigation of the car from the internet and the speed of the vehicle can also be identified with this GPS (Ray-Shine Run et al, 2010). And this speed can be displayed in the Internet and it can be controlled if it exceeds the limit.

GPS receiver:

The GPS (Global positioning system) sends the data in bytes such as longitude, latitude, velocity, time and heading. All of them are very useful, but mostly the longitude and latitude values can be used to identify the exact location of the car. As mentioned in the GPS PMB-688 data sheet, one GPS receiver can receive up to 20 parallel satellites, where these data are sent in NMEA protocol. We made a test by receiving the longitude and latitude for different locations, it shows the data and the value is consistent for the change of placements. The Figure.1, shows the value received in different locations by GPS.

![GPS Test data for various locations](image)

**Fig. 1:** GPS Test data for various locations.

Ultrasonic sensor:

The ultrasonic sensor operates with the piezo electric transducer as the sound waves are transmitted and received. This transducer transmits a packet of ultrasonic pulses and the echo pulses which are received converted as voltage. This voltage is given to the microcontroller and this distance can be calculated by using the below mentioned formula:

\[
\text{Test distance} = \frac{(\text{High level time} \times \text{Sound velocity})}{2}
\]

Sound velocity = 340 m/s

As shown in the Figure.2, the sensing range Sd is the active range. This test distance is measured in meters.

![Ultrasonic distance measurement range](image)

**Fig. 2:** Ultrasonic distance measurement range.

GSM Modem:

The GSM modem is used as the medium to transmit the data from the vehicle. The date distance from the ultrasonic sensor and the exact location where the vehicle is located will be transmitted from the vehicle to the centralized computer. The received data is processed and sent to the web server. In some advanced processors...
like ARM Cortex M3, an inbuilt Wi-Fi peripheral is available and data can be uploaded to the server directly.
We have experimented with CAN based microcontroller where controller is interfaced with GSM connectivity.

III. Distributed embedded computing model:

Time Triggered Object Model:

The Figure 3 shows the base class for the embedded control software. The base class named value Object has the attribute named value. The attribute value has three methods to it named revise, scan and fix. The revise method will calculate and store the value of the attribute value. The method scan is used to read the value of the attribute value and the method fix is used to set the value of the attribute value. The revise method automatically executes the calculation. And there are no arguments in the method. If there is a case of an object needs another objects attributes, the first object should call the scan method of second object.

---

Event triggered object model:

The design of automotive control system is made using the CAD/CAE tools MATLAB and SIMULINK (MLSL), ASCET etc. There will not be any control flow in a general block diagram but Simulink tool helps in providing notations such as triggering system.

The Figure 4 shows the pure event-triggered distributed objects. In this example the task of Engine revolution is distributed to two ECUs namely Ultrasonic sensor (ECU 1) and the Engine revolution calculator (ECU2). When an event is sent by the ultrasonic sensor in ECU 1 the revise method of Engine revolution is executed.

---

Time Division scheduling:

There are two ways by which the optimal performance can be achieved one is by reducing the responsive time of the event-triggered distributed processing or by reducing the jitter of the time-triggered processing. As the entire performance of the control system depends on the jitter, the priority can be given to reduce the jitter in the time-triggered processing. Here we present a time division scheduling to combine both the time-triggered process and the event-triggered process. As mentioned before the execution cycle or machine cycle consists of time-triggered and non-time-triggered processing. The figure 5 shows the complete example of the time-triggered objects. In this figure, there are two tasks i.e., two objects named Surrounding of the vehicle and the Injection opening. Both objects are distributed to two ECUs. The Load of the vehicle is at the first ECU (ECU 1) and the Injection valve opening is located at the second ECU (ECU 2). To get the value of the Load of the vehicle the Injection valve opening calls the scan method of the replica of Load of vehicle. Surrounding of the vehicle object is calculated by sensors connected to ECU 1. Injection opening object is considered as actuator where ECU 2 is connected with fan module. This is analogous to an automotive electric drive train.
The attribute value will be copied to the replica to maintain the state of the original value to be consistent. This time division scheduling is applied to the CPU by the RTOS and the communication network on which we work is CAN (Control area network) which allows to do the time triggered and event triggered tasks.

The Figure 6, shows the example of the time chart of time division scheduling. The timer of RTOS of ECU1 and the timer of RTOS of ECU2 is synchronized using the communication protocol CAN. Let the period of the execution cycle is T, this time T is same as the communication cycle of CAN. This execution time T consists of static segment and dynamic segment. The process of time-triggered tasks is executed in the first segment (static segment) and the process of non-time-triggered like event-triggered and data-triggered tasks are executed in the second segment (dynamic segment). Reducing the jitter can be achieved in the time-triggered processing since it is statically programmed and not by the event of any other time-triggered or non-time-triggered tasks.

**IV. Internet of things:**

Since there are several Internet applications are able to transfer the information in internet such as file transfer protocol or web, to differentiate this Internet of things (Thing’s information) from others it must have some particulars to become a new type of internet application. Generally the working of the internet application as a form of sharing information in web is divided into three parts; they are uploading the information, transmitting the information and downloading the information. The difference of Internet of Things is that the information as the object of the information uploading (called uploaded information) has some features different
from other Internet applications i.e., Information of things. The need of the Internet of things is increasing. The service of other internet application may also do this work, but only the uploaded information of Internet of Things is just for the object’s (transducer’s) information. The object’s information is the specialized orientation of Internet of Things. By processing and remotely managing the things, the human welfare can be greatly increased and Internet of Things can utilize the maximum of thing’s entities.

**Architectural review of IoT:**

Similar to TCP/IP, OSI architectural models, the architecture model of the Internet of Things is discussed here. A three layered architecture is generally accepted for this internet of things. These three layers are Perception layer, Network layer and Application layer as shown Figure 7.

**The Perception layer:**

The perception layer is the one where the things which are gathering information are identified. The perception layer may include RFID, Transducers, Camera, GPS, Sensor networks. These things will gather the information from any means; the perception layer’s main task is to identify such things.

**The Network layer:**

The network layer acts as a main part of the Internet of Things, this layer is to transfer the information gathered by things using a communication medium. This layer includes a convergence network where the information is transmitted. The communication medium can be anything which can be wired or wireless such as 3G, Bluetooth, Wi-Fi, ZigBee etc.

**The Application layer:**

The application layer is the one which meets the demand of the industry. This layer is the combination of the IoT’s technical portion and the real time industrial needs.

**Representing Things on Internet:**

As in Figure.8, for representing Things in the Internet, we have Web server where the data is to be displayed and the Web services which will enable the application to communicate and the storage space to store the data.

**Web Server:**

An object processes HTTP requests through a Web server which is augmented directly on the board or externally, here we use the external one with the help of GSM Modem. These requests are processed, and a respective control mechanism is returned for operations on the thing. Thus, an object has a number of processes that enables its virtual access and control.

**Web Services:**

Communication with a thing can be done in various APIs, here we used the .NET and VB based interface to enables the state and functions of things to be communicated to the Web. Thus, an object has a number of communications channels directly accessing its representation and state.

**Storage:**

An object must have capabilities to cache Web resources and status information. The storage can be onboard or remote such as a data center, or private cloud. Hence, an object has a number of storage options.

**V. IoT based distributed computing environment:**

The Figure.9, shows the structure of the distributed computing environment. This structure consists of application program-Internet of Things, a RTOS and a CAN driver. The application program consists of the concept of Internet of Things and the middleware consists of the objects created from the program. The RTOS has a timer based on our time division scheduling. The replication mechanism is available in the middleware and it has the configuration data which is sent by the user from the internet, this configuration data is the information of the particular ECU.
Fig. 8: Connecting the sensor to the internet.

**CAN Network:**

The time division scheduling of the task can be done in the distributed systems with Controller Area Network (CAN). Since the time division scheduling is same as the scheduling of the CAN, the timer of the RTOS is synchronized with the timer of the CAN. Here the CAN frame is composed of the dynamic segment (non-time-triggered process) and the static segment (time-triggered process).

**Middleware:**

The middleware part provides the location transparency with replication. In addition to the time-triggered environment, we provide an event channel mechanism that is taken care by the RTOS.

Fig. 9: Structure of Distributed computing environment.

This middleware executes the replication process in the time-triggered processing and it executes the event triggered processing as executing the engine revolution by sensing the value of the distance from ultrasonic sensor. The replication process uses the two kinds of stubs, original stub which is used to bridge the middleware with the original object and the replica stub to bridge the middleware with the replica object. The original stub for Load of the vehicle is located at the ECU1 and the replica of this Vehicle load object is located at the ECU2.

The distance from the ultrasonic sensor is located at the ECU 1 and the Replica of the distance value from the ultrasonic sensor and the engine revolution are located at the ECU 2. In operation the middleware calls the original stub in the ECU1 to get the attribute value of the distance and that value will be stored in the communication buffer register. Since the maximum data of the CAN is eight bytes, the middleware packs the replica data of the objects into the CAN message packet as long as the total size of the replication data is less than or equal to 8 bytes. After converting it into the packets, the data will be sent through the CAN network. The ECU2 gets the transmitted value of the ECU1 by the middleware and it is stored in the receive buffer register and this value will unpacked by the replica stub to set the attribute value of distance. The authorized users can visit the user portal and above mentioned parameters can be visualized. This application program helps in controlling the speed of the vehicle through internet by providing a data configuration from the web link and that is connected to the middleware of the distributed computing system. The Figure 10, shows the interface of the Thing or an object with internet. Thus the information of an object can be seen in the Internet.

**RESULTS AND DISCUSSION**

The distributed object computing model experiment has been done and evaluated with the help of Microchip’s PIC Microcontroller PIC 18F458. Microcontroller has on-chip CAN controller. The operating clock frequency is 20MHz and the transmitted rate of the CAN is 1Mbps.

\[
\text{Fosc} = 20 \text{ MHz} \\
\text{CAN operating Frequency} = \frac{(\text{Fosc})}{(2 \times \text{PRESCALAR})}
\]
The PRESCALAR value of the CAN is 1, so the CAN operating frequency will be 10 MHz. For transmitting one bit of data CAN protocol is programmed with the nominal bit time which is 1 μs, hence it operates at 1 Mbps data transfer speed. In CAN, a message of 1 byte to 8 bytes of data can be transferred in a single frame and for an ideal condition a frame consists of 111 bits for 8 bytes of data, but practically in average the CAN frame will consist of 114 bits. So it takes 114 μs for the transmission of 8 bytes of data. And the average time for the replication of 8 bytes of data is ideally 228 μs. This value is calculated by the round trip (114+114) replication between the two ECUs.

This value is in the case of no blocking during the arbitration. The replica message request or transmission may be blocked if another message with is transmitting in the CAN bus. The maximum value of the blocking time is equal to the transmission time. If the transmission of a message takes 118 μs, then the blocking time will be 118 μs. The jitter of the synchronization will be caused by the blocking, even when the priority of the message is high due to the busy condition of bus. So we estimated the jitter of the replication will be 118 μs, and this may not be in the case of arbitration failure. The Figure.11, shows the implementation of the Ultrasonic sensor based distance measurement this acts as the ECU1 this node is designed using PIC microcontroller with the CAN Transceiver MCP 2551.

Figure.12, shows the implementation of electric drive train actuator model. This node acts as the ECU2 and the data is received by the CAN transceiver MCP 2551 which is connected to the PIC microcontroller.

The figure.13, shows the complete model consists of ECU1 distance measurement and the ECU2 Engine revolution control which is connected using the CAN network as shown in the figure.
Fig. 13: ECU1 and ECU2 connected through CAN.

The Figure.14, shows the results which is displaying in the hyper terminal. This shows the distance with the proceeding vehicle as Front distance and the distance with the succeeding vehicle as rear distance, the Longitude value and Latitude value of the GPS is also uploaded through the UART port.

The Table.1 shows the results obtained in the experiment. Tabulation is formed for ten different locations with the latitude and longitude value of the GPS along with the distance of proceeding vehicle and the succeeding vehicle.

![Hyper Terminal Result](image)

Fig. 14: Results displaying in hyper terminal.

The figure.15 and figure.16 shows the graphical representation of the distance of the succeeding vehicle and the proceeding vehicle for various locations. The Figure.17 shows the operation of the injection valve in accordance with the distance with the other vehicles measured by the ultrasonic sensor.

The distance of the user vehicle with the other vehicle is available as data in the webpage (user portal). The speed of the vehicle is displayed as controllable data which in turn controlled interactively.

Table 1: Results of a test experiment.

<table>
<thead>
<tr>
<th>LOCATION</th>
<th>LATITUDE/LONGITUDE</th>
<th>PROCEEDING VEHICLE DISTANCE (cm)</th>
<th>SUCCEEDING VEHICLE DISTANCE (cm)</th>
<th>INJECTION VALVE OPENING (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1101.1208,N,0/658.0095,E,0</td>
<td>188</td>
<td>209</td>
<td>92.0</td>
</tr>
<tr>
<td>2</td>
<td>1101.1085,N,0/657.9791,E,0</td>
<td>13</td>
<td>8</td>
<td>5.0</td>
</tr>
<tr>
<td>3</td>
<td>_3.36.4.48/0.051213,N,0</td>
<td>51</td>
<td>101</td>
<td>40.0</td>
</tr>
<tr>
<td>4</td>
<td>1101.1063,N,0/657.9833,E,0</td>
<td>17</td>
<td>29</td>
<td>13.0</td>
</tr>
<tr>
<td>5</td>
<td>1101.0972,N,0/656.9859,E,0</td>
<td>116</td>
<td>134</td>
<td>78.0</td>
</tr>
<tr>
<td>6</td>
<td>_0.10.6.8/0.051213,N,0</td>
<td>72</td>
<td>41</td>
<td>34.0</td>
</tr>
<tr>
<td>7</td>
<td>1101.0910,N,0/657.8712,E,0</td>
<td>45</td>
<td>105</td>
<td>89.0</td>
</tr>
<tr>
<td>8</td>
<td>_0.18.16/2.051213,N,0</td>
<td>125</td>
<td>100</td>
<td>72.0</td>
</tr>
<tr>
<td>9</td>
<td>1101.0979,N,0/657.9849,E,0</td>
<td>182</td>
<td>193</td>
<td>90.0</td>
</tr>
<tr>
<td>10</td>
<td>_1.17.945/8.051213,E,0</td>
<td>79</td>
<td>4</td>
<td>23.0</td>
</tr>
<tr>
<td>11</td>
<td>1101.2813,N,0/655.7192,E,0</td>
<td>123</td>
<td>70</td>
<td>52.0</td>
</tr>
<tr>
<td>12</td>
<td>1101.0837,N,0/659.8503,E,0</td>
<td>174</td>
<td>170</td>
<td>80.0</td>
</tr>
</tbody>
</table>
Fig. 15: Graph – Location – Proceeding Vehicle Distance.

Fig. 16: Graph – Location – Succeeding Vehicle Distance.

Fig. 17: Injection valve opening based on the distance level.
Conclusion:

The devices connected to the internet where the application is monitored and controlled is the latest business model of the industry. In this paper driverless car implementing Internet of Things (IoT) is presented. The distributed embedded computing is emulated with the help of hardware setup consisting of network of ECUs, sensors and actuators. The mixed architecture concept where time triggered and event triggered processing are combined is proposed. Finally an experiment which mimics an autonomous car is carried out in the setup and the speeding of vehicle is limited and controlled over web interactively. In the future the proposed concept can be extended by implementing it in real autonomous automotive power train system, bringing the capability of IoT to automobiles.

REFERENCES


