Offline Analysis of Sensor Can Protocol Logs Without Can/Vector Tool Usage

R.S.Sandhya Devi, P.Sivakumar, Sukanya M

Abstract: ***This paper presents Offline Sensor CAN protocol Log file Analysis. Windows platform C language Compiler tool is selected to replace the conventional Analysis using Vector CANoe tool because of its advantages in terms of simplicity, flexibility, license-free hardware, less execution time and portability. In the proposed method, a C script was developed to tokenize the desired CAN elements into array of structures for various Analysis using any C compiler windows platform tool. According to the Analysis, the CAN elements of interest are brought and Analysis is performed. An optimized C script developed gives the expected result similar to results obtained in Vector CANoe using CAPL script. Performance can be analyzed in terms of execution time, and cross checking the Analysis results obtained is done using both CAPL script and C script. The purpose of this optimized technique is to ensure Test Analysis and can be made possible even without costly Vector hardware license and to obtain Test results offline with less processing time***

Keywords: CAN Protocol, CAN channel, RAdio Detection And Ranging, CAN Access Programming Language.

I. INTRODUCTION

Vector CANoe is one of the Universal development tool for CAN bus systems which helps in observing, analyzing and supplementing data traffic on the bus line [6]. Its main purpose is to emulate a node on the bus, system environment for testing a node and ensures link between 2 buses of different speeds, creates test generator for studying the physical layer (arbitrations). It has built-in functions to list the bus data traffic (Tracing), to display data segments of specific messages, to transmit predefined and replay recorded messages. CANoe supports the developer in implementing the diagnostic functionality of an ECU. They provide access to the diagnostic interface for testing ECUs. CANoe may be used both as a diagnostic tester and to simulate ECU diagnostics. A simulation can be generated manually or automatically from the underlying communication data base. This remaining bus simulation to monitor the communication behavior of complete networks or individual ECUs is the basis for the subsequent analysis and testing phases. CAPL is a procedural and event driven programming language that allows to setup a customized platform for testing distributed system software, by emulating the behavior of nodes.

For CAN-based networks, modules, and distributed embedded systems, CAN Access Programming Language, (CAPL) makes it possible to program the CANalyzer/CANoe for developer-specific applications that use the CAN protocol.

Although CANoe marks its significance, the cost of Canoe License is expensive [1]. Also, the processing time is more. For example, log file of 3600 seconds will take an hour on running CANoe in simulated bus mode when given as input. As it is concerned on requirement for testers to get the results in stipulated time, the need for less execution time becomes mandatory for developing and integrating CAPL scripts. So to achieve less processing time, a solution is to develop a simulator which imitates the operation of a real-world process or system in windows platform. Since log file of size more than 1 GB is given for Analysis using Vector CANoe it takes in terms of minutes as its execution time, but instead if C script based log file is developed and given through command prompt it takes only few seconds to execute, which marks its advantage. In order to achieve it, an integrated C script is developed. This replaces the existing Analysis in CAPL using Vector CANoe as an optimized solution.

II. ANALYSIS USING VECTOR CANOE TOOL

Vector CANoe offers many different ways to stimulate ECUs in the network [1]. CANoe represents the state-of-the-art test environment. It is the ideal testing tool for the entire system for efficient ECU testing. CANoe is a comprehensive software tool with intuitive operation for analysis and stimulation of bus communication [3]. CANalyzer is used to check whether and what type of communication is occurring on the bus. It can also be used to send or receive log data.

Some of the advantages of CANoe includes Intuitive operation like easy observation, analysis and supplementing of the data traffic, flexible analysis capabilities by configurable function blocks, e.g. Filter, Interactive Generator or Replay seamless logging of bus data and replay for offline analysis, flexibly programmable, e.g. for extensive analysis tasks, allows user programming via CAPL [2] which can be viewed in Figure 1.CANoe has Measurement Setup and simulation Setup windows which are the two required windows which marks its importance.

Revised Manuscript Received on December 08, 2018.

R.S.Sandhya Devi, Assistant Professor, Department of Electrical and Electronics Engineering, Kumaraguru College of Technology, Coimbatore, Tamilnadu, India,

Dr.P.Sivakumar, Assistant Professor, Department of Electrical and Electronics Engineering PSG College of Technology, Coimbatore, Tamilnadu, India

Sukanya M, Assistant Professor, Department of Electrical and Electronics Engineering ,PSG College of Technology, Coimbatore, Tamilnadu, India

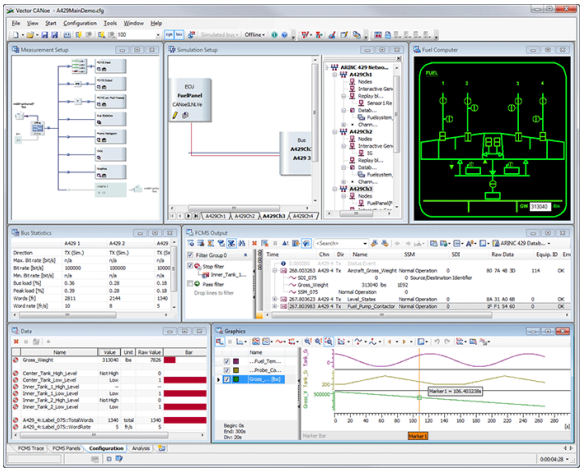

Figure 1 Vector CANoe Windows

A. Measurement Setup Window

CANalyzer is controlled and configured from the data flow diagram or Measurement Setup. Modes of Operation: Online Mode & Offline Mode

- Online Mode: CAN PC-card acts as a data source for CANalyzer
- Offline Mode: Data log file serves as the data source for CANalyzer
- CANalyzer supports a maximum of 32 (virtual and real) channels
- Multiple CAN cards can be used to transmit and receive messages[2]
- Configuration of the Hardware is a must before using it. Each channel can be parameterized independently - independent bus systems with differing speeds.
- The hardware is initialized at the start of a measurement. Loading log file to run in offline mode and creating destination folder to where the object files have to be stored is enhanced.

The trace window displays when messages and data occurs. The Write window displays message data numerically and symbolically. Logging window does record and playback messages.[3]

Executable programs that are part of the CANalyzer are:

- CANdb++ Editor
- define N/W nodes, messages & properties
- define & position signals within a CAN message and defines physical units, symbolic values of a signal
- CAPL Browser
- create CAPL programs for the transmit and analysis branches
- Can get help from Databases
- CANalyzer Main Program
- measure and simulate CAN systems

B. Simulation Setup Window

Simulation setup window is mainly used to simulate networks at the beginning of the development process, the network is fully simulated as shown in Figure 2. In this phase it is possible to operate CANoe with or without a physical bus. Simulation with real bus. In the former case it is sufficient to connect the card's two CAN controllers to the bus. In this case the operating mode in the simulation dialog remains set to Real Bus. All messages generated in the simulation setup are then placed on the real bus. When real bus and controllers are not used then it is possible to operate CANoe in pure simulation mode. Switch the operating mode in the simulation mode by changing the dialog from Real Bus to Simulated Bus. Bus access (sending and receiving messages) is then simulated [2]. In simulation mode, i.e. without a physical bus, the program emulates the functionality of the CAN chip on the interface card. In this mode, the bus baud rate is the only system parameter to be configured in the card's configuration dialog. The CAPL scripts can also be developed in CAPL browser for that particular ECU node emulator as shown in Figure 3. Similarly upload any *.CAN files in any ECU node for simulation. In same way in Replay block exported log/ascii file can be loaded with simulated mode in case if real time sensor datas from Test bench with Real bus mode cannot be made possible.

Periodic messages with variant CAN channel, Message ID, direction, payloads and data length code [7] can be made possible with adding Generater Block can be generated. Additionally filtering of specific Message IDs, Timestamp, CAN channels can be made possible using Trace Window [4].

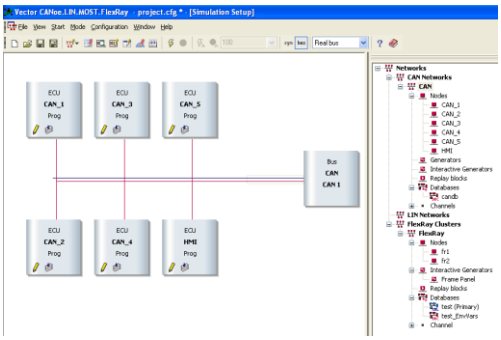

Figure 2 Vector CANoe Simulation Setup Window

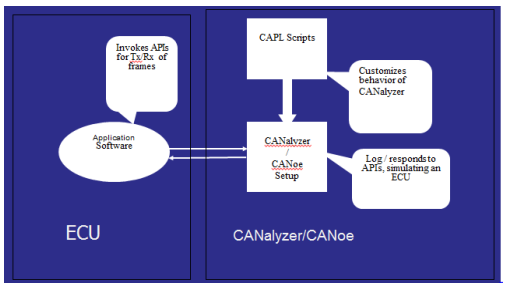

Figure 3 Link between Emulator ECU and CANoe

III. CAPL PROGRAMMING ENVIRONMENT

A. CAPL features

Most important features of CAPL scripting is that it allows to create, modify, and maintain CAPL programs which can interface with a wide variety of inputs, outputs and other functions as shown in Figure 5. Handling of Start-stop events, keyboard entry events, the ability to transmit and receive CAN messages, interaction with the serial port and parallel port, the use of timers [5].

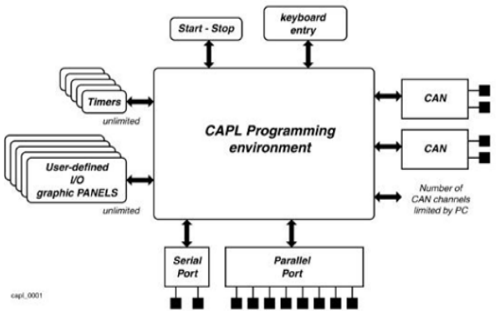

Figure 5 CAPL Programming features

B. Event Handling in CAPL

An event is a specific occurrence, or a condition, that may invoke some necessary action. For e.g, Timeout of a Timer.CAPL has a host of built in functions, that can perform required event handling. The Built in functions cover periodic, event triggered and continuous event handling.

C. Message Handling in CAPL

A message is a piece of data,that encapsulates information that is specific to an entity (ECU/node). A message has a predefined 'identifier'. All information in a message is exchanged over a network in the form of 'frames'. The frames have standard formats. CAPL supports direct message field.

CAPL has in built function such as CancelTimerstops that prevents the timer event procedure from being executed.When an active timer that has been set with setTimer().

Syntax: cancelTimer(Timer t);

Parameter: timer or msTimer variable

Eg: on key F1

{

cancelTimer(delay);

}

Since log file of size more than 1 GB is given for Analysis using Vector CANoe it takes in terms of minutes as its execution time in simulated bus mode. Even running ASCII file in offline mode takes in terms of several minutes to check for signals inside messages in trace window, which again marks as disadvantage. So to avoid both these issues a solution is to develop a C script which could possibly minimize the check through time and execution time with log files given for analysis.

IV. WINDOWS PQLATFORM COMPILER C-SCRIPT

A. Processing ASCII file

The desired elements of interest from given log file was tokenized and stored into array of structures and number of elements in structure is as requirement. Before storing conversion of tokenized elements into its corresponding datatype is done for later usage for Testcases. According to Analysis purpose the stored structure elements are used for processing. Analysis outputs are resulted as text or *.csv files to cross check the performance of both C script and CAPL scripts. This task was made possible using C language and compiler used is any C compiler in windows platform which can be an open source software and ensures results to be outputted with less execution time.

B. Master Control Xml Script

For integrating Testcases, further a requirement of Master Control became mandatory. As a solution a MASTER XML SCRIPT was designed. This consists detailed description of User's requirement matching with Testcase purpose given in detail. Also Enable/Disable option to Enable or Disable one or more Testcases becomes possible. With this setting desired CAN channel was made possible as CANalyzer supports maximum of 32 channels. Together as future scope addition of more Testcases can be made possible such that this XML script ensures flexibility and simplicity of executing Testcases.A Snap shot of Master XML script is shown in Figure 6.

```
<!-- Analysis Selection values: 1 => Testcase_9 enabled and 0 => Testcase_9 disabled -->
<!-- Set CAN_Channel value: 2 => PCAN_Channel enabled -->
<!-- Set CAN_Channel value: 4 => VCAN_Channel enabled -->

<ENABLE_DISABLE_TESTCASE>

<TESTCASE_1> 0 </TESTCASE_1> <PCAN_CHANNEL_1> 4 </PCAN_CHANNEL_1>
<TESTCASE_2> 0 </TESTCASE_2> <PCAN_CHANNEL_2> 4 </PCAN_CHANNEL_2>
<TESTCASE_3> 0 </TESTCASE_3> <PCAN_CHANNEL_3> 4 </PCAN_CHANNEL_3>
<TESTCASE_4> 0 </TESTCASE_4> <PCAN_CHANNEL_4> 2 </PCAN_CHANNEL_4>
<TESTCASE_5> 0 </TESTCASE_5> <PCAN_CHANNEL_5> 2 </PCAN_CHANNEL_5>
<TESTCASE_6> 0 </TESTCASE_6> <PCAN_CHANNEL_6> 2 </PCAN_CHANNEL_6>
<TESTCASE_7> 0 </TESTCASE_7> <PCAN_CHANNEL_7> 2 </PCAN_CHANNEL_7>
<TESTCASE_8> 1 </TESTCASE_8> <PCAN_CHANNEL_8> 2 </PCAN_CHANNEL_8>
<TESTCASE_9> 0 </TESTCASE_9> <PCAN_CHANNEL_9> 2 </PCAN_CHANNEL_9>

</ENABLE_DISABLE_TESTCASE>
```

Figure 6 Master_control_XML_script

C. Sequence Check Analysis

Sequence check in this context means checking if the desired Sequence is obtained till the end of ASCII file by checking order for desired Message IDs using both C script and CAPL script which gives similar results. Check was also done for avoiding interruption of other Message IDs for desired CAN channel which is shown in Figure 7 and Figure 8. To check for synchronization of four RADARs, each message from four RADARs is taken and checked for occurance of and repetition in correct order without allowing any other Interruption Messages and form any other CAN channel other than desired channel mentioned.

D. Periodicity Analysis

Delta-time or Periodicity means the difference in time from its first occurance to its next occurance calculated in milli seconds. .Periodicity for desired Message IDs from four RADARs namely REAR RIGHT(RR), REAR LEFT(RL), FRONT LEFT(FL), and FRONT RIGHT(FR) was cross checked which is required to be around desired milli seconds range for both C script and CAPL script. Delta-time for one particular MessageID was compared for both scripts. Periodicity is calculated to check if all messages received form four sensors are synchronized.

E. Sequence check of Status Messageswith Timestamp synchronization Analysis

Test bench of any Car with Side Radar version SRR3 consist of four RADARs occupying REAR, FRONT and SIDES. Detections from these RADARs related to obstacle is mapped into desired detections containing messages.Radar datas from four sensors RR, RL, FR, FL consists of detections form both Rare Right and Rare Left,out of which certain detections are Messages form REAR RIGHT of radar and REAR LEFT of radar with one Status messages FRONT LEFT and FRONT RIGHT of radar . FR and FL consist of only one Status Messages i.e the last message of the detections received.The requirement is to check if all these detections in sequenceRR, FL, RL, FR. Along with it, cross checked results of, if Timer value of Status Messages of RR,RL,FL,FR are equal. To confirm synchronization of all four Status Messages from four RADARs periodicity or delta time is cross verified for one of one particular status message.

V. SIMULATION RESULT AND DISCUSSION

A. Comparison of results for Sequence Check Analysis

On checking if the desired Sequence is obtained till the end of ASCII file by checking order for desired Message IDs using both C script and CAPL script which gives similar results. Check was also done for avoiding interruption of other Message IDs for desired CAN channel which is shown in Figure.7 and Figure.8.

```
Message_Id, Timestamp, CAN_channel
Error:, Received  Message_ID_3 before Message_ID_2 ,0.000278

Error:, Received Message_ID_4 before Message_ID_3 ,0.018628

Message_ID_1, 0.027443, 2
Message_ID_2, 0.046243, 2
Message_ID_3, 0.049651, 2
Message_ID_4, 0.068092, 2
    Correct Sequence, cycle_completed is, 1

Message_ID_1, 0.077439, 2
Message_ID_2, 0.096272, 2
Message_ID_3, 0.099655, 2
Message_ID_4, 0.117907, 2
    Correct Sequence, cycle_completed is, 2

Message_ID_1, 0.127549, 2
Message_ID_2, 0.146096, 2
Message_ID_3, 0.149495, 2
Message_ID_4, 0.168190, 2
    Correct Sequence, cycle_completed is, 3
```

Figure 7 C script message sequence check

Source	Message
System	Start of measurement 04:01:52.278 pm
System	01-0003 CAN 1 (Classical CAN) simulated bus with 500000 BPS.
System	01-0003 CAN 2 (Classical CAN) simulated bus with 500000 BPS.
System	01-0003 CAN 3 (Classical CAN) simulated bus with 500000 BPS.
System	01-0003 CAN 4 (Classical CAN) simulated bus with 500000 BPS.
System	and animation factor = 1.
CAPL / .NET	CAN_Channel Message_ID Timestamp
CAPL / .NET	Message_ID_1 2 0.008880
CAPL / .NET	Message_ID_2 2 0.027150
CAPL / .NET	Message_ID_3 2 0.028820
CAPL / .NET	Message_ID_4 2 0.047510
CAPL / .NET	Correct Sequence, cycle completed is, 1
CAPL / .NET	Message_ID_1 2 0.058930
CAPL / .NET	Message_ID_2 2 0.078090
CAPL / .NET	Message_ID_3 2 0.079570
CAPL / .NET	Message_ID_4 2 0.098260
CAPL / .NET	Correct Sequence, cycle completed is, 2
CAPL / .NET	Message_ID_1 2 0.108880
CAPL / .NET	Message_ID_2 2 0.127040
CAPL / .NET	Message_ID_3 2 0.128700
CAPL / .NET	Message_ID_4 2 0.147860
CAPL / .NET	Correct Sequence, cycle completed is, 3

Figure 8 CAPL Script Message Sequence Check

B. Comparison of Periodicity for Periodicity Analysis

Periodicity for desired Message IDs from four RADARs namely REAR RIGHT(RR), REAR LEFT(RL), FRONT LEFT(FL), and FRONT RIGHT(FR) was cross checked which is required to be around desired milli seconds range for both C script and CAPL script. Delta-time for one particular MessageID was compared for both scripts in Figure. 9 and Figure. 10. Periodicity is calculated to check if all messages received form four sensors are synchronized.

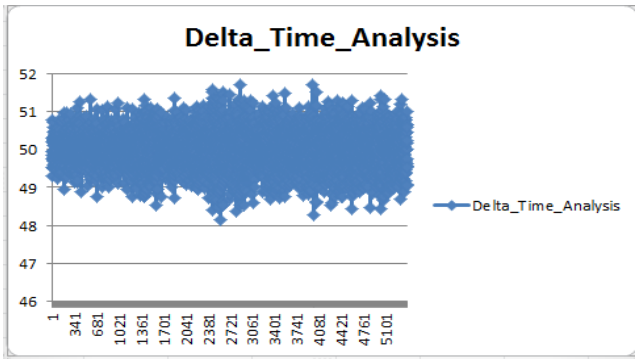

Figure 9 Snap Shot Of C Script Delta Time Analysis

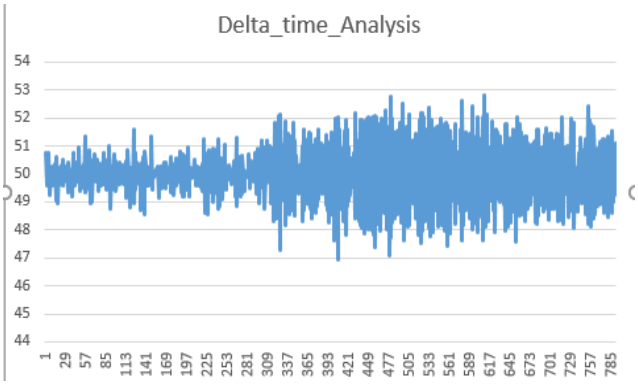

Figure 10 Snap Shot Of CAPL Script Delta Time Analysis

C. Comparison of Sequence Check of Status Messages with Timestamp synchronization Analysis

Radar datas from four sensors RR, RL, FR, FL consists of detections form both Rare Right and Rare Left, out of which certain detections are Messages form REAR RIGHT of radar and REAR LEFT of radar with one Status messages FRONT LEFT and FRONT RIGHT of radar . FR and FL consist of only one Status Messages i.e the last message of the total detections received. The requirement is to check if all these detections in sequenceRR, FL, RL, FR. Along with it, cross checked results of, if Timer value of Status Messages are equal. To confirm synchronization of all four Status Messages from four RADARs periodicity or delta time is cross verified for one particular status message which is shown in Figure. 11 and Figure. 12.

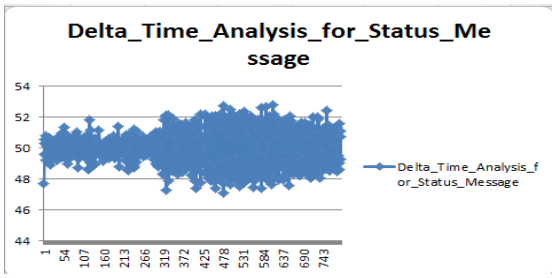

Figure 11 Snap Shot Of C Script Delta Time Analysis

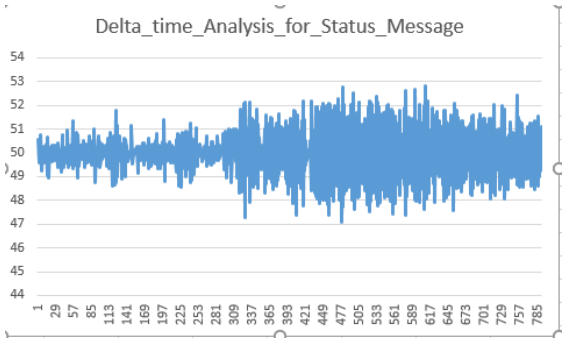

Figure 12 Snap Shot Of CAPL Script Delta Time Analysis

D. Overall Performance Comparison

On comparing the Analysis results of all Tests done, found similar results for both CAPL and C scripts which marks C script to be an optimized solution for performing Analysis using stored tokenized datas. Along with this an important concern to avoid costly Vector Hardware License is achieved. Together with this, reduced execution time in terms of few seconds for running C script compared to several minutes taken for running CAPL scripts is achieved. The reduced compilation time and execution time of C script developed for running the analysis can seen in Figure 13 and Figure 14.Since log file of size more than 1 GB is given for Analysis using Vector CANoe it takes in terms of minutes as its execution time, but instead if we develop a C script and give log file through command prompt it takes only few seconds to execute, which marks its advantage.

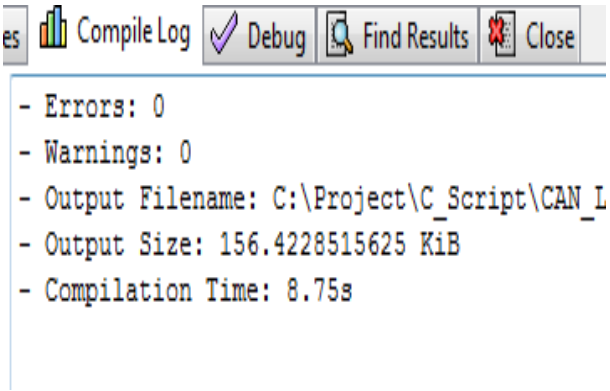

Figure 13 Snap Shot Of C Script With Reduced Compilation Time

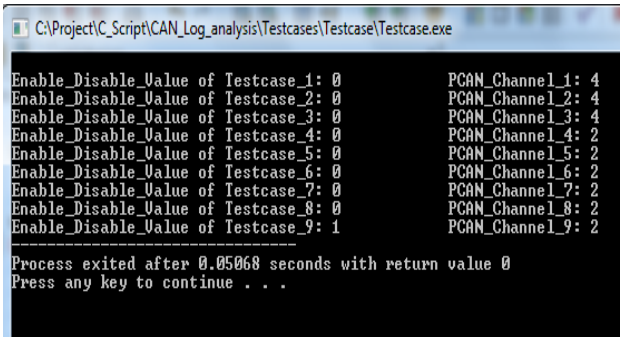

Figure 14 Snap Shot Of C Script With Reduced Execution Time

VI. CONCLUSION

On validating and comparing the results for similar Analysis using CAPL script and C script, which was found to be same, wherein C script marks its importance in CAN diagnostics with its reduced execution time and avoided to overload of expensive hardware license. As a future work more Analysis can be made to perform on tokenized log datas.

REFERENCES

1. Fang Zhou, Shuqin Li, Xia Hou: "Development Method of Simulation and Test System for Vehicle Body CAN Bus Based on CANoe", Proceedings of the 7th World Congress on Intelligent Control and Automation June 25 - 27, 2008
2. Wu Kuanming: "The Theory and Design of Application System of CAN Bus", Beijing: Beijing University of Aeronautics and Astronautics Press, 2001. 18-34.
3. Vector Informatik GmbH. CANoe Manual (Version 5.2). 2005.
4. Yang Li, Yan Weisheng, Gao Jian, Zhang Lichuan: "CAN System Development Based on CANoe Measurement and Control Technology", 2007, 26(4):66-67, 75.
5. Zhang Xinbo, Sun Zechang, Luo Feng: "Study on CAN Body Network Simulation with CANoe", Journal of Jiangsu University (Natural Science Edition), 2003, 24(5): 36-39.
6. Sivakumar, P., B. Vinod, RS Sandhya Devi, and ER Jayasakthi Rajkumar. "Real-time task scheduling for distributed embedded system using MATLAB toolboxes." Indian Journal of Science and Technology 8, no. 15 (2015).
7. Paret, Dominique. Multiplexed networks for embedded systems: CAN, LIN, Flexray, Safe-by-Wire... John Wiley & Sons, 2007.
8. Mathankumar M., Suryaprakash S., Thirumoorthi P., Rajkanna U, Development of smart car security system using multi sensors, International Journal of Pure and Applied Mathematics (IJPAM), 117(22), pp. 19-23, 2017.
9. Nethaji Kumar D, L.Bharathi, R Mahadevan," Polynomial Time Routing Algorithm To Identify Shortest Path In A Distributed Wireless Networks", International Journal Of Innovations In Scientific Andnal Of Innovations In Scientific And Engineering Research, Vol .4, Iss. 10,2017, Pp. 204-208.
10. Vijayanandh R , Senthil Kumar M, Vasantharaj C , Raj Kumar G, Soundarya S ," Numerical Study On Structural Health Monitoring For Unmanned Aerial Vehicle", Journal Of Advanced Research In Dynamical And Control Systems, Vol. 9, Sp– 6 , 2017, Pp. 1937-1958.
11. A.Amsaveni And K.Anusha," A Circularly Polarized Triangular Slot Reconfigurable Antenna For Wireless Applications", International Journal Of Pure And Applied Mathematics, Vol.116, No.11, 2017,Pp. 81-89.