Dogo Rangsang Research Journal
ISSN : 2347-7180

UGC Care Group I Journal
Vol-10 Issue-07 No. 8 July 2020

Simulated Platform for Automotive ECU Systems

Dr.Udhayshankar.C
Associate Professor
Department of Electrical and Electronics
Kumaraguru College of Technology
(Affiliated to Anna University)
Coimbatore, India

Kishanraj.V.K
M.E(Power Electronics and Drives)
Department of Electrical and Electronics
Kumaraguru College of Technology
(Affiliated to Anna University)
Coimbatore, India

***Abstract*— Most of the automotive based components require modelling-based software development and testing automation which are necessary for product development and testing. Most of the Automotive ECUs (Electronic Control Units) require the development and testing on a hardware platform which makes difficult for development and testing of hardware and embedded software development. This paper focuses mainly on the virtual development and testing platform for automotive ECU development which paves way for parallel and faster development and validation of automotive based ECUs. Simulated platform consists of simulation-based environment wherein the developed software components can be validated without any hardware dependency and also will have a scope of testing automation on the platform.**

Keywords—ECU, Simulated platform, embedded software, microcontroller, hardware, communication, testing, automotive

I. INTRODUCTION

This document is a final report from my final semester project as a post graduate student. My mentor Dr.Udhayshankar.C from department of Electrical and Electronics at Kumaruguru college of Technology, directed my efforts for the concept and simulation of Simulated Platform for Automotive ECU Systems via a software tool for establishing a development and testing platform for automotive ECU systems

In general, development and testing/validating an automotive system requires electronic hardware and software interfaces to be embedded with each other. Due to limited timeline for development, parallel development and testing becomes complex. Simulated platform for automotive ECU systems is one solution for the existing complexity. Using this platform, model based/ controller-based software development and validation can be done without hardware dependency.

II. COMPONENTS

A. *Diagnostics*

Vehicle diagnostics [8] is a concept which is introduced to identify a problem before it occurs and also to provide fail safe protection when the vehicle is under operation. There are many protocols involved in implementing diagnostics in vehicle. Major vehicle manufacturers standardized the protocol due to the importance of need of diagnostics in vehicles.

The diagnostics is communicated throughout the vehicle by different ECUs (Electronic Control Units) using CAN (Controller Area Network) communication [14]. Diagnostics is an important component in the software architecture which is useful in identifying the faults occurring in vehicle system.

Diagnostics has different standards as specified in ISO automotive requirements. Any diagnostic protocol should be as per the standard specification.

B. *Model components*

Model based software development is essential for any automotive system [12]. UML (Unified Modelling Language) is a concept used in model-based software development. Integrating the different software layers is required for the system functionality [10]. Block diagram of interaction between modules is illustrated fig (i). The layers include

- MCAL (MicroController Abstraction Layer)
- Complex device drivers
- Middleware
- Application module
- User application interfaces

III. BLOCK DIAGRAM

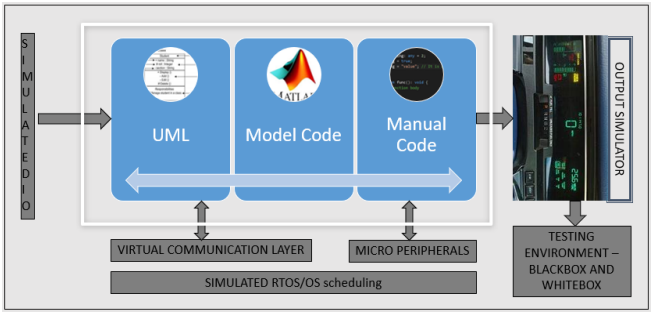
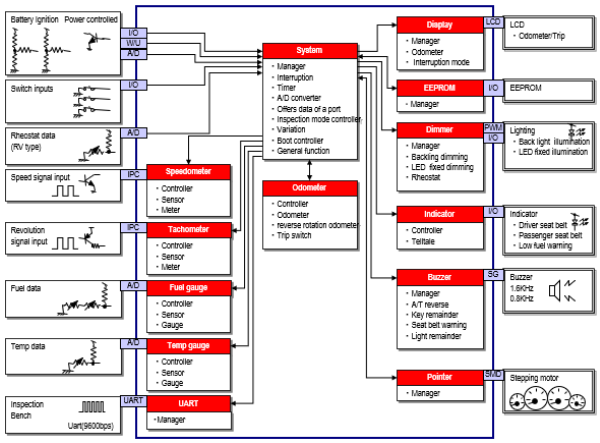


Fig (i) Block Diagram

A. *Simulated IO*

The Input and Output simulation [15] for analog and digital signals can be given to the platform as a simulated interface. This is achieved through the predefined signal parameters which are provided through MATLAB [7] stub codes where the values are provided through a scheduler as represented in Fig (ii).



Fig(ii) Software structure

B. *Simulated OS/ Scheduler*

- Schedulers act as a Real Time Operating system providing the task timer for the operations to be handled. Based on the application requirements, the OS (Operating System) can be hard real time or soft real time.
- Timing parameters are set in accordance with the clock frequency of the micro-controllers. Interrupt handlers are used to process maskable and non-maskable interrupts in the system.
- Bootloader is a software component which is used for reflash the microcontroller through any communication protocol. OS routine for bootloader and application will be different.

C. *Peripherals*

Peripherals [10] are broadly classified into GPIO (General Purpose Input Output) and Controlled GPIOs. These are the integral part in any micro controller architecture. GPIOs can be configured as either inputs or outputs. As an input, a GPIO pin tells the microcontroller what voltage is present on the pin

D. *Communication layer*

The basic communication protocols [11] used in embedded automotive systems are

- UART (Universal Asynchronous Receiver/Transmitter)
- SPI (Serial Peripheral Interface)
- CAN (Controller Area Network)
- I2C (Inter Integrated Circuit)
- LIN (Local Interconnect Network)
- Ethernet
- MOST (Media Oriented System Transport)

Communication layer [16] has the CDDs (Complex Device Drivers) and components in accordance with OSI (Open Systems Interconnection) architecture model. Module interface is illustrated in Fig(ii) and (iii). The communication can be confirmed by acknowledging the reception of signals through simulated interface. The data-loss will also be prevented by using this method.

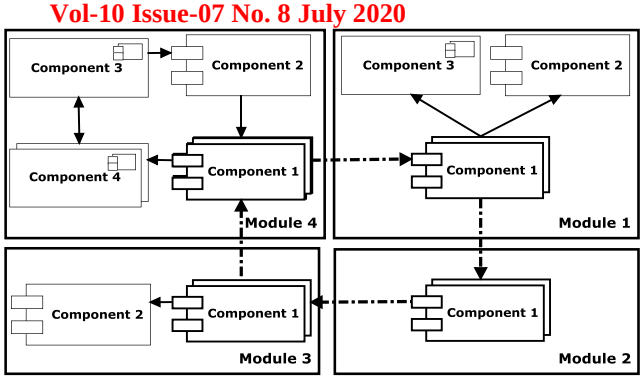


Fig (iii) Module interaction

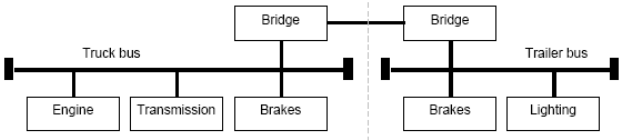


Fig (iv) Communication nodes

E. *Output Simulator*

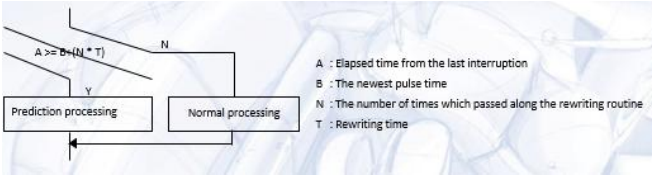
- The Output simulator [17] is used for displaying the functionality. It is achieved through software tools such as Altia or MATLAB[7] as illustrated in Fig (iv). Graphical representation will be helpful in understanding the behavior of the system [15]. Any defect in the system can be logged using log file which can be used to recreate the simulation.

Fig(v) Output simulation

- The simulated output is again connected to any testing interfaces for performing Black-box or white box testing.

F. *Testing Interface*

The testing interface [9] is a closed loop control [13] where the expected and observed values/result is validated. An example of speed calculation is explained as follows. Fig (vi) shows the speed calculation.

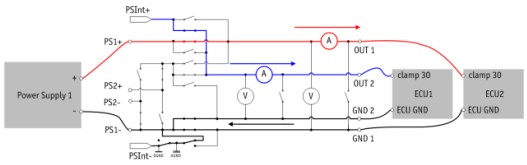


Fig(vi) - Speed Calculation

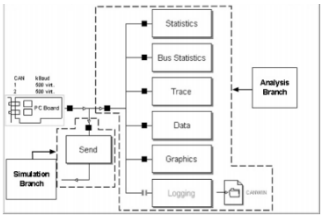
- Prediction: The oldest pulse time is transposed to prediction time among the copied pulse width data, and a moving average deviation is calculated.
- Normal: The moving average deviation of the copied pulse width data is calculated.
- Weighted average value calculation of a directions angle. Directions angle calculated at the time of the update of speed calculate to weight averaged.
- Calculation of the amount of displacement: Difference of the last weighted average result and this weighted average result is calculated, and it considers as the amount of displacement
- Moving average deviation calculation of the amount of displacement. Moving average deviation of the amount of displacement is calculated.

IV. RESULT

The simulation and test report panel are represented with the appropriate results.

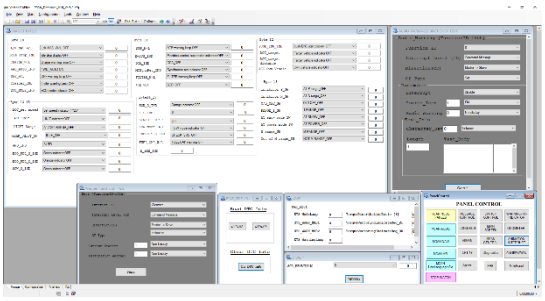


Fig(vii - a) – Screenshot of a Hardware Configuration window



Fig(vii - b) – Screenshot of a module Configuration

Testing panels are useful for monitoring the functional behavior of the system. Test results are obtained through log files with different file formats.



Fig(viii) – Screenshot of a testing panel

V. CONCLUSION

Thus, the Simulated platform offers solution for software development and testing without the dependency of hardware. The timeline required for the development process is reduced and the development cost can also be reduced by using this method. It is a virtual interface of hardware used in automotive systems which can be a solution to overcome the drawback of hardware failures.

Automated test environments will be helpful for diagnosing the failures thus reducing the defects in the system before implementing it in the actual automotive application.

ACKNOWLEDGMENT

Thanks to Dr.Udhayshankar.C for his devoted guidance and support for this project. The clarity of his comments in the workflow allowed me to complete the project and demonstrate the simulation with result.

REFERENCES

[1] SAE International automotive standards website. Available: https://www.sae.org/
[2] International Organization for Standardization website. https://www.iso.org/
[3] Automotive Software Architecture standards website. https://www.autosar.org/
[4] TUV SUD standard: https://www.tuvsud.com
[5] Automotive architecture example. Available: https://www.aptiv.com/newsroom/article/evolution-of-vehicle-architecture
[6] Vector Knowledge base online materials. Available: https://www.vector.com/in/en/support-downloads/knowledgebase/
[7] Lee, W., Yoon, M., & Sunwoo, M. (2003). A cost-and time-effective hardware-in-the-loop simulation platform for automotive engine control systems. Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 217(1), 41-52
[8] Montazeri-Gh, M., Nasiri, M., & Jafari, S. (2011). Real-time multi-rate HIL simulation platform for evaluation of a jet engine fuel controller. Simulation Modelling Practice and Theory, 19(3), 996-1006.
[9] Kong, F., Zhang, L., Zeng, J., & Zhang, Y. (2007, August). Automatic measurement and control system for vehicle ECU based on CAN bus. In 2007 IEEE International Conference on Automation and Logistics (pp. 964-968). IEEE.
[10] Mouzakitis, A., Copp, D., Parker, R., & Burnham, K. (2009). Hardware-in-the-loop system for testing automotive ECU diagnostic software. Measurement and control, 42(8), 238-245.
[11] Lu, J., Guo, Y. Q., & Wang, H. Q. (2008, December). Rapid prototyping real-time simulation platform for digital electronic engine control. In 2008 2nd International Symposium on Systems and Control in Aerospace and Astronautics (pp. 1-5). IEEE.
[12] Palladino, A., Fiengo, G., & Lanzo, D. (2012). A portable hardware-in-the-loop (HIL) device for automotive diagnostic control systems. ISA transactions, 51(1), 229-236.
[13] Short, M., & Pont, M. J. (2008). Assessment of high-integrity embedded automotive control systems using hardware in the loop simulation. Journal of Systems and Software, 81(7), 1163-1183.
[14] Nissimagoudar, P. C., Mane, V., Gireesha, H. M., & Iyer, N. C. (2020). Hardware-in-the-loop (HIL) Simulation Technique for an Automotive Electronics Course. Procedia Computer Science, 172, 1047-1052.
[15] Sini, J., & Violante, M. (2020). A simulation-based methodology for aiding advanced driver assistance systems hazard analysis and risk assessment. Microelectronics Reliability, 109, 113661.
[16] Chen, S., Chen, Y., Zhang, S., & Zheng, N. (2019). A novel integrated simulation and testing platform for self-driving cars with hardware in the loop. IEEE Transactions on Intelligent Vehicles, 4(3), 425-436.

[17] Guissouma, H., Lauber, A., Mkadem, A., & Sax, E. (2019, April). Virtual Test Environment for Efficient Verification of Software Updates for Variant-Rich Automotive Systems. In 2019 IEEE International Systems Conference (SysCon) (pp. 1-8). IEEE.