

Accelerated and Integrated Real Time Testing Process Based on Two Universal Controllers on Rapid Controller Prototyping

ZhENCHUN Xia and FENG Gao

MB SIM Technology Co., Ltd

Kazuhide Togai

Mitsubishi Motors Corporation

Hiroki Yamaura

Mitsubishi Automotive Engineering Co., Ltd

Copyright © 2008 SAE International Inc.

ABSTRACT

Rapid Controller Prototyping (RCP) is an efficient method for design & development of ECU (Electronic Controller Unit) at early stage. Usually, RCP requires firstly performing Software-in-the-loop simulation and then connecting universal controller (e.g. MicroAutoBox) to real controlled system for testing of controller functionality. During this process, it is likely that some problems related to signal configuration and real time characteristics occur and consequently give rise to unexpected results, e.g., sensor signals or controlling signals produce large deviation and possibly damage components of real system under severe condition. On the other hand, it cannot make sure that the real time characteristics of designed controller are suitable just after applying Software-in-the-loop simulation. One problem often encountered is that the sampling time of controller is not suitable for real system very well, even though it has been tested by Software-in-the-loop testing in simulation environment. One approach of rapid controller prototyping based on two universal controllers is presented. It is performed after the Software-in-the-loop simulation and before real time testing of real system. It is able to check efficiently the real time characteristics of designed controller as well as configuration of I/O interface. Majority of problems or errors related to real time characteristics or interface configuration can be found quickly and corrected immediately. In this approach, the used two universal controllers have different roles: one is to perform the designed controller functionality and the other is to simulate the controlled system. All interface signals are configured and then the two universal controllers are connected by sensor signals line and CAN line according to the specifications of real system. Based on the approach, the real time closed loop testing can be conducted and real time characteristics of system can be checked and analyzed. This approach improves greatly

reliability of designed controller and reduces the duration of RCP. It facilitates the testing of designed controller on real system as well as decreases risk of damage on real system to the minim. It is also convenient to develop and test diagnosis functionality of controller. This approach is applied on the design & development of one new type of TCU (transmission controller unit) and proves high efficiency and capability for developing ECU.

INTRODUCTION

In the last decades, the development of ECU (electronic controller unit) in automotive industry has experienced large changes, which results from requirements for higher efficiency of development, shorter time-to-market, more reliable quality and more accurate controlling functionality.

However, in the early stage of ECU development, one traditional way in which ECU is designed developed and tested gives rise to more and more difficulties to shorten the time-to-market, meanwhile achieve high quality. This traditional way is based on C code programming development style and process.

C code has been one popular and efficient tool in a variety of industrial fields in the last decades. Especially in embedded system fields, C code and its programming style is one efficient interface between hardware system and programmer. Unfortunately, with the development of complex system in automotive industry, the codes performing real time functionality is becoming more and more complicated and gives rise to tanglesome structure. This leads to terrible difficulty in terms of readability and maintenance. Especially when the controlling functionality needed to be updated, the modification and debug process can be one nightmare under such conditions.

In the late 1970s, CAD (Computer Aided Design) begins to develop into design field of industry. Diagram style design attracts more and more attention. New graphic design tool & platform emerges, e.g. Matlab, Simulink, etc. Matlab is the abbreviation of Matrix laboratory. As its name indicates, Initially Matlab is developed to deal with complicated matrix calculation. As well known, the matrix calculation using C coding is one large burden programmers have to bear. Matlab integrates the matrix calculation algorithm in terms of m. file and Simulink blocks. With more and more toolboxes used, Matlab/Simulink begins to develop powerful environment for graphic design.

Real time Workshop is developed to perform automatic code generation. More tools are designed to perform the automatic transfer from graphic design to codes that can be run in real time on target hardware. Targetlink is one efficient tool that can basically produce production code.

Based on the CAD and the development of modeling & automatic coding, from the early 1990s, the rapid controller prototyping begins to be applied from laboratory to practices in automotive industry. A lot of big automotive companies in the world gradually perceive the importance of RCP. One of important forces is the pressure coming from more and more complicated software structure and faster time-to-market with still high quality. In the past, one complete design of ECU needs at least dozens of months from the beginning of specification to production. Compared with this traditional development flow, RCP fully utilizes flexible and efficient development methods, which includes graphic specification, model establishment, automatic coding generation, fast and real time testing on real system, etc. However, in the 1990s, automatic coding generation did not develop into maturity stage; automatic generated codes just are used for testing and invalidation of controller algorithm. Its efficiency and precise can not reach production level. After the testing using RCP, software engineer still needs to manually program the code into ECU.

Entering 21 Century, with the further maturity of modeling tools, RCP hardware platform and automatic coding technology, RCP gradually develops into its industrial practices stage. One interesting example is some big automotive companies even begin to perform such projects: transferring paramount C codes to Simulink model. One of its purposes is to perform bypass testing to add new functionality, and another one is to achieve high readability for update of functionality in future. As everyone can see, facing at least several thousands hundred lines of C code, only reading and maintaining them seems more and more difficult tasks and need large manpower and time.

MAIN SYSTEM FRAMEWORK

RCP mainly includes the following important steps: functionality specification, modeling according to functionality specification, software-in-the-loop testing, C code automatic generation into target hardware, real time testing on real time system, correcting and updating functionality according to testing results, finally writing the functionality code into ECU. RCP schematic diagram is indicated in Fig. 1.

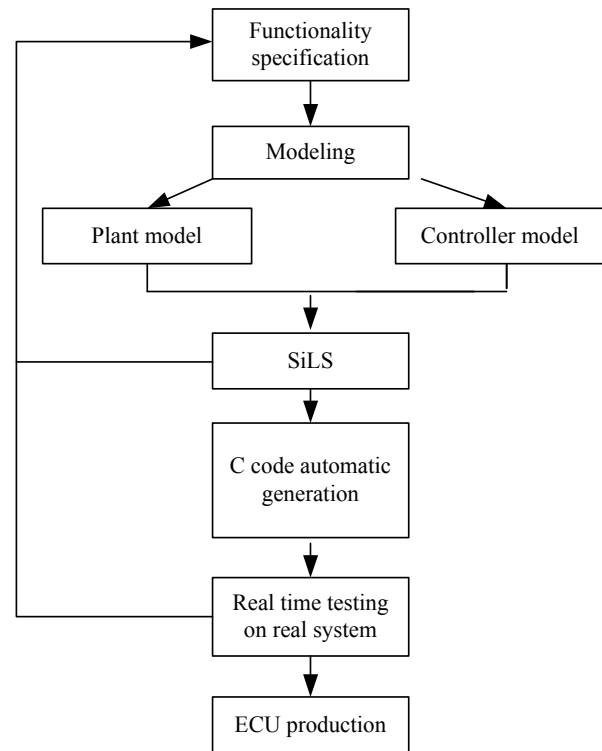


Fig. 1 RCP schematic diagram

Functionality specification of ECU is important portion in the development flow. It will directly determine what type and what function should be realized. In traditional development, functionality specification of ECU has to be finished based on different system levels; from high level of automotive company to sub-level of OEM, and deeply to different develop team. The phenomenon results from the fact that different system level has different platform and environment. They cannot be integrated into one complete development flow. This is changed completely by RCP. The automotive company (or OEM) can directly describe specification completely. The specification will not be separated into different system levels; instead it is directly realized by modeling. This can be viewed as one graphic specification. In early stage of RCP, some manpower and time are needed to establish model. However it is worthy doing so. These models will accelerate the development flow greatly; moreover, they can be re-used in other development projects.

The plant model and controller model are needed for Software-in-the-loop simulation. One example of SiLS results is indicated in Fig. 2.

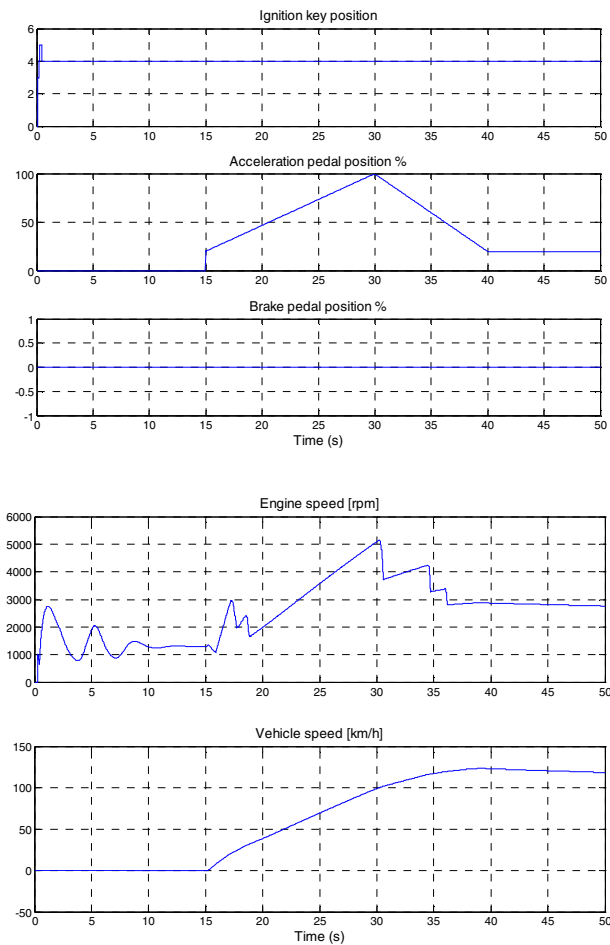


Fig. 2 Software-in-the-loop Simulation

The whole system, including plant model and controller model, needs to be modeled based on mechanical, electrical and controlling dynamics. Appropriate sampling time and calculation methods (e.g. Euler method) needs to be selected to actuate the model of system.

The purpose of SiLS is to perform the initial testing of controller functionality. During SiLS, most malfunctions or errors can be detected by means of massive testing with different testing scenarios. SiLS proves the tremendous importance in the development of controller. SiLS can validate at least three aspects efficiently: controlling functionality, signal flow, sampling time scope. Controlling functionality can be directly tested through the interaction between plant model and controller model. Signal flow realized in the model of whole system will lead to clear understanding & tracking which is critical to master the behaviors of system. Sampling time scope is another crucial issue. Under large numbers of signals conditions, different characteristics of signals will give rise to large difficulty to analyze the proper sampling time. SiLS can easily solve this. Appropriate sampling time will actuate system normally; otherwise system will probably be activated abnormally, even put into chaos.

In the last ten years, RCP has been playing important role in ECU R & D field of automotive industry. RCP is utilized to develop a variety of ECUs for different component in vehicle, e.g. engine ECU, ESP ECU, transmission ECU, and roof ECU, etc. RCP greatly accelerates the procedure of ECU development and undoubtedly exposes the traditional C code programming of ECU development to large pressure.

However, the progress of state-of-the-art is unlimited. When researchers and engineers have been dedicated themselves to developing different ECU to achieve faster time-to-market with high quality, they also find out that there are still big space to improve the RCP schematic diagram indicated in Fig. 1.

In Fig. 1, the “C code automatic generation” refers to that the controller model is automatically transferred into C code and downloaded into target hardware, e.g. AutoBox, MicroAutoBox and PC-compatible board, etc. Then the target hardware is connected to the real system for real time testing. Before this real time testing on real system, just the software-in-the-loop is performed. Possible problems often, even inevitably occur: the target hardware that acts as controller to actuate & control the real system possibly cannot meet the requirements of real time operation. Even under extreme condition, it cannot be completely avoided that the controlling signal will exceed normal scope, therefore possibly cause the damage of components in real system. For example, the PWM signal from the target hardware will possibly exceed its normal scope and lead to the abnormal action of motor. It is dangerous if action of motor cannot be controlled precisely. The root of such problem lies in the difference between software simulation and real time running: the software-in-the-loop simulation cannot completely simulate & indicate the interactions of system in real time environment.

It is not difficult to find the reasons for such situation. Normally software-in-the-loop environment is based on non real time operation system, e.g. Windows operation system. In running circumstance of such system, the operation of CPU is split into pieces of time. The pieces of time are assigned to different working of software program. How fast and how long time one program runs and needs is determined by the number and duration of the pieces of time which are assigned to the program. For example, the running time of one model system in Matlab/Simulink environment doesn't have direct relationship with real time. If the CPU frequency and internal memory is large enough, the model system may take only one tenth of time to finish running. On the other hand, if a model system is very complex and needs large memory, it will take far too longer simulation time than needed in real time environment. It is not to one's surprising that one model in certain operation system can take several hours to finish one simulation which simulation time is set to just several minutes.

Another important reason is related to the issues of interruption. In Matlab/Simulink, there is no interruption concept existing, neither hardware interruption nor