Implementation and Research of Bootloader for Automobile ECU Remote Incremental Update

Ji Zhang, Xiangyu Zhu, Yong Peng
Clean Energy Automotive Engineering Center of Tongji University, Shanghai 201804, China

***Abstract*—This paper is based on the application of automobile ECU remote update in order to facilitate automotive remote update. At first, related technology of Bootloader was introduced. System design project of automobile ECU Bootloader was completed according to the design requirement. Then "incremental update" was put forwarded as a way of application program update. The paper carried out a detailed program design, which including memory layout, state flow chart, the specific software running processes and software structures. Finally, the results of experiment prove that the way of automobile remote update is more safety and faster than traditional method.**

Keywords-automobile ECU remote update bootloader; incremental update.

I. INTRODUCTION

Currently, the development of automotive electronics technology has become an irreversible trend. Pure mechanical structure is unable to meet our needs. But with the increasing amount of automotive electronics, the amount of ECU and software codes have also been growing dramatically. In 2009, in an article in ,Munich University of Technology, Professor Manfred • Broken indicated that compared with the F-35 fighter's 5.7 million lines codes and the Boeing 787's 6.5 million lines codes[1], there is about 100 million lines of software codes in a modern luxury car. From a certain perspective, now our cars are running on the software codes. With huge amount of software codes, it brings a lot of difficulties for automotive ECU firmware update. If every update operation should be completed by specific technicians, it will be a time-consuming and laborious task. In order to solve this problem, automobile ECU remote update technology provides a more efficient way[2][3].

To achieve automobile ECU remote update, firstly, software update package should be downloaded from the server to the automobile platform through a wireless network. Secondly, update package will be programmed into target ECU though automobile's own network. The second step requires the ECU to have the ability to achieve online upgrade[4].

Bootloader is a segment of codes in the microcontroller which runs before application program. A rational design for Bootloader enables automobile ECU to have the function of online update. Designing an online update Bootloader for automobile ECU is a prerequisite to achieve automobile ECU remote update. This paper is based on automotive remote update, proposing a method of application program incremental update. As to memory layout, state flow chart, the specific software running processes and software structures have been detail researched.

II. REQUIREMENT ANALYSIS OF AUTOMOBILE ECU REMOTE UPDATE

This paper mainly analysis the related function requirements which Bootloader should satisfy during the process of automotive ECU online update. Taking into account of the actual application, ECU Bootloader should not only satisfy the function of application program online update, it should also meet many other property requirements.

1) Security: In order to prevent others to apply for a Bootloader update service request from automobile network to the ECU, which will make an illegal update to the automobile ECU application program. ECU Bootloader must have the appropriate security measures.

2) Operation reliability：ECU Bootloader must have the ability to deal with external abnormal fault, so as to ensure the system to run correctly. In the process of automobile ECU online update, some abnormal situations such as power-fail may happen, ECU Bootloader must have the ways to deal with these exceptional faults, to prevent entering "isolated" state.

3) Accuracy and novelty：The novelty here means that the application update package should be correct and novelty. Under the normal situation, server will send the latest application update package to the vehicle platform. Then the update task will be executed in the vehicle platform. But it cannot exclude the abnormal situation which happens in the sever. For example, the wrong version or old version update packets may be sent out. At the same time, the vehicle platform also is not complete the detection function (version record function abnormal or record update not in time will contribute to this situation). ECU Bootloader is request to have the version check function, to ensure the correctness and novelty of application software version.

III. IMPLEMENT AND RESEARCH OF AUTOMOTIVE ECU REMOTE INCREMENTAL UPDATE

A. Related technology of Bootloader

Bootloader is also called boot load program. This program is the first section of software code which will be executed after hardware equipment power on reset. For different application fields, the work methods are also different.

In the embedded system, Bootloader is running before the operating system. Through this little program, we can initialize hardware equipment and establish the memory mapping table. Thereby, establish the proper system environment; get ready for the final calls of operating system kernel

In general, vehicle ECU doesn't need to load the operating system kernel, the main job of the Bootloader is used to application program codes update, namely the online update function.

Online update function is an implementation of the IAP technology. IAP is the abbreviation of "In Application Programming", meaning "programming in the application ", namely when the program is running, program memory can be erased by program itself. Popular speaking, the procedures could write date or modify data into program memory by itself. This procedure which was programming called the application program. The implementation of this section programming function is Bootloader program. Now the Flash storage technology is widely used in the automobile ECU, flash storage technology support software to erase or write, provided the precondition for the automobile ECU online update.

B. Implementation and research of Incremental update technology

Incremental update is in terms of the whole update. Generally, programming for ECU program, we adopted the way of whole update, erasing the entire memory space at first. Then the whole new codes were written into the target ECU. But sometimes, the change between the new version and the old version is little; there is no need to renewal whole program codes. We can extract the part of the codes which is changed, and then only update part of the codes. Erase FLASH storage space and complete writing. In order to realize the incremental update, when design the application program. Linked scripts need to be reconfigured. The memory space of each function module in the application program should be specified assigned. Without a detailed memory program layout, when we make a change on a functional module, the generated target file is shown in figure 1. As the picture show, without specified designation, each function module of memory space will be closely linked in the default mode. In this way, memory fragmentations will be eliminated and we can ensure good use of memory utilization. But, though the module or the codes are not changed, the memory addresses are also changing. When the program is updated, this part of the code also needs to be erased and rewritten.

By configuring the linker script file, specific memory space being assigned for each function module can solve the above problem. As shown in Figure 2, the modification of function module A before and after, the other module contents and addresses in the controller memory are not changed. The part of update contents are only related to modified module. But because of the memory allocation, in order to fulfill its expansionary function needs, some memory address need to be the reserved for each function module .Each function module will create a certain amount of memory fragmentations which will affect memory utilization. To reduce the memory fragmentations, the most effective way is to reduce the number of variable function module. It needs to make a detailed analysis to the application program module.

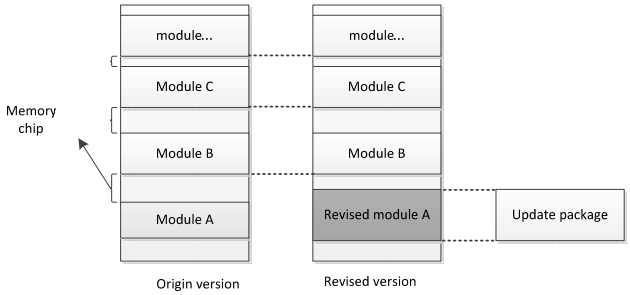

Figure 1. Flash storage space configuring

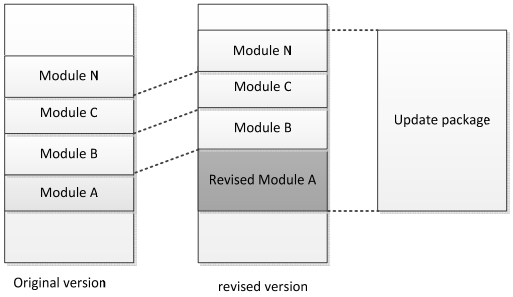

Figure 2. After detail designed Flash storage figure in the default mode

C. ECU flash Space layout

ECU flash layout as shown in figure 3 can effectively support the Bootloader self -update function. It also can prevent the system to enter the isolated state. In the layout application update area, Bootloader area and fixed area are located in the program / Flash code, these three block areas should keep alignment with program / code Flash minimum erase block. In space layout, there is a part of Bootloader area and another part of the application area, the two part codes completely independent. The application program area includes the interrupt vector table and specific application program. This layout also distributes a block data of flash or EEPROM area as user data storage, wherein the backup area is effective when MCU does not contain EEPROM. Due to the lack of support of EEPROM, we can't operate the date by a single byte. When data is rewrite, the whole sector needs to be erased. If a failure happens in the system, all dates will loss, so it needs to allocate another area to backup information. User data storage area includes two parts: the version information area and sign area.

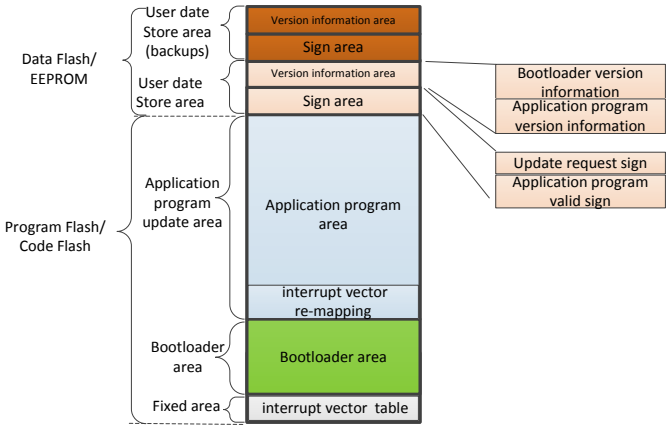

Figure 3. Flash Space layout

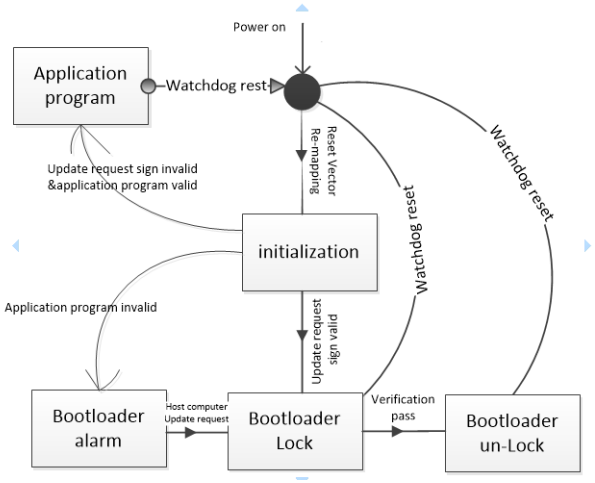

Figure 4. Operational state flow chart

The version information area is used to store Bootloader version information and the application version information. During the process of version information validation, this part of information need to be read and compared with the version information which is sent by host computer. After the update completed, version information is need to re-write. In order to prevent the external abnormal situation, this part of information needs to be stored independent and keep the effectiveness. For example, during the process of Bootloader update, when erase operation is executed, the system is suddenly interrupted. If the version information stored in the Bootloader area, the information will be lost. When power is turned on again, update will be executed again. This time we can't not read version information or unable to complete the version check work. Sign area update request sign is used to support update service from application program to the Bootloader update service by system resetting. Application program validity sign is used to prevent abnormal accident during the application update process, guide exchange sign is used to ensure reliable update for Bootloader.

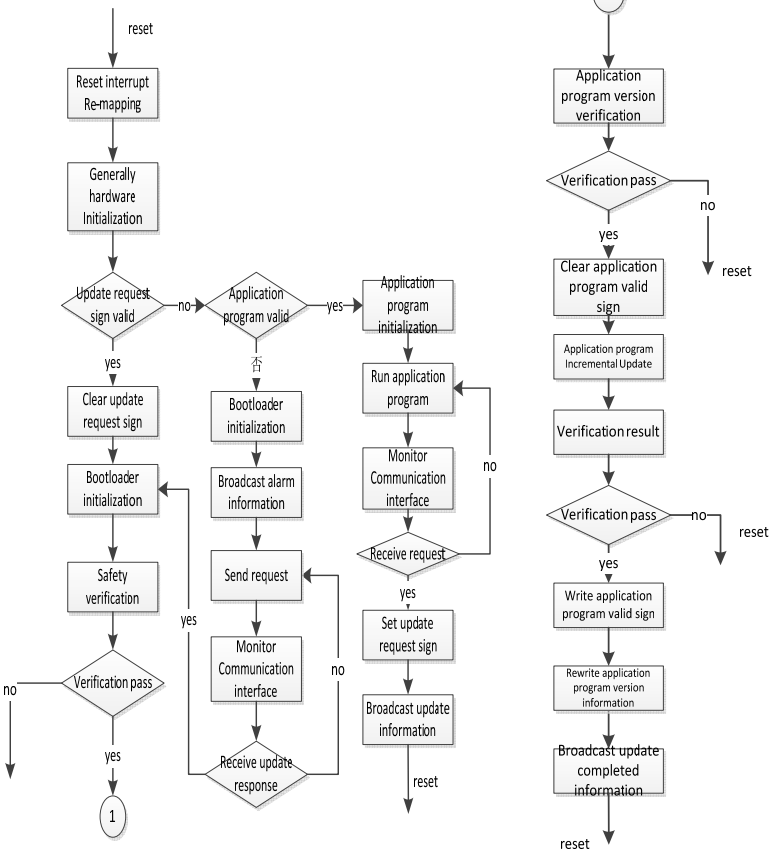

Figure 5. Detailed running flow before ECU enter update service

Fig.6 The Detailed running flow of ECU update service

D. ECU node operation state

The flow graph of ECU nodes operational state, as shown in Figure 4, there are 5 running state: application program running state, system initialization state, Bootloader alarm status, Bootloader lock state and Bootloader unlock state, the last four states are running in the Bootloader area.

E. ECU program running process

This section divide ECU running program process into two parts, the first part is ECU running process before ECU enter the update services, the second part is the running process of ECU update service.

The operation process before ECU enters update service is shown in Figure 5. After ECU is power on, judge "guide exchange area sign" firstly. According to the sign to complete the reset vector remapping, enter the Bootloader effective area, and executing ECU basic initialization task. After initialization work is completed, begin to judge update request sign and application program effectiveness sign. When the update request sign is valid , Bootloader goes into the locked state. If the update request sign is invalid, judging whether the current application program is valid. If the application program is not valid, then enter the Bootloader alarm status. If not going into the application program running state.

After Bootloader enter alarm status, Bootloader relevant initialization job is completed, including communication interface initialization, and then broadcast an alarm message to inform other ECU on the network,

Then send the request to the host computer, and monitor communications interface, if received update response, enter the Bootloader locked.

After entering application program running state, application program initialization job is completed firstly. Then execute application program task. At the same time, periodically monitor communication interface, judging whether receiving the update request from the host computer. If the update request is received, its sign will be written into the user data (interrupt should be turned off before writing and turned on after completing), and an update message will be broadcasted to inform the other ECUs on the network, and finally the watchdog will be reset.

After entering the Bootloader locked state, ECU will communicate with host computer, performing security verification process. If verification is valid, enter the

un-locked state and execute update service. If not watchdog will be interrupted, system will restart. The detail running processes of ECU update service is shown in Figure 6. After entering the update service, ECU will determine that choose to execute application program updates or Bootloader program update according to the date which is sent by host computer.

When performing application program update, make a communication with host computer firstly. And execute the application version check job. If check is fail (the updated version package doesn't newer than the current version), the watchdog will be reset.

If check is passed , application program valid sign in the user date area will be erased before update , (to prevent an accident occurs during the update process, after the system reset will enter the wrong application program), then execute the application program incremental update job (erase by block and write by block). After update is completed, check whether updated result is correct (achieve by calculating the CRC). If the check is not passed, the watchdog will be reset. If check result passes, application valid sign will be written and application program version information in the user date area will be re-written. Finally broadcast messages will send to ECU on the network.

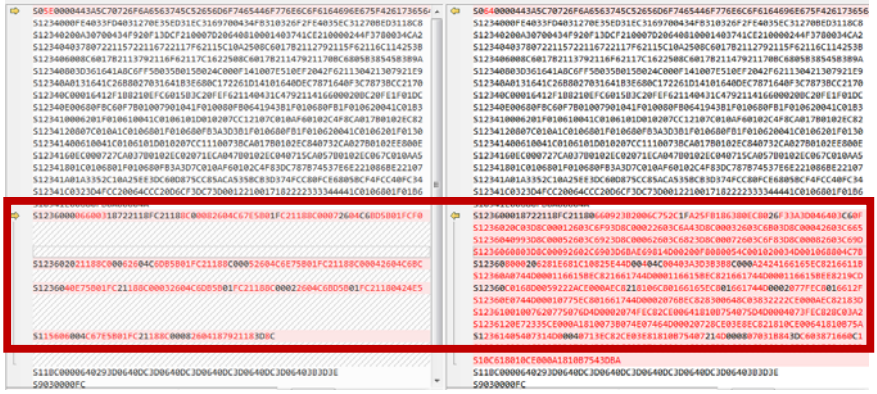

Figure 7. Compared with application program target file of "marquee" and "digital tube"

F. Incremental update validation

In order to validate the "incremental update" of application program, this paper creates two different application program object files: led_sci.S19 (marquee effect), nixie_sci.S19 (digital tube display effect). The memory layouts of these two application programs are the same. only in the validation module area enable different functional modules, in order to achieve different output effect. Comparing of the two target files as shown in Figure 7. As you can see, though allocating memory space in detail, both of them are changed only in the designated validation module area. Therein to, S0 row includes file name (containing path), here is different. There is no relation with specific writing content. Before the specific test, through the BDM interface, using programmer write complete Bootloader1.s19 (including content of fixed area and user data area) write into microchip. After writing successfully, unplug the BDM programmer, turn on super terminal software, select the COM port, and set specific parameters. The baud rate is 38400bps, 8 Data, 1 Stop, No parity checking, and no data flow control, the parameter is request to match with the ECU port parameter. After parameter setting is finished, click OK, it will automatically turn on the connection. Then resets the ECU, because the update request sign is set to be valid in default situation, the main interface of super terminal will print out the online update interface as shown in figure 8.

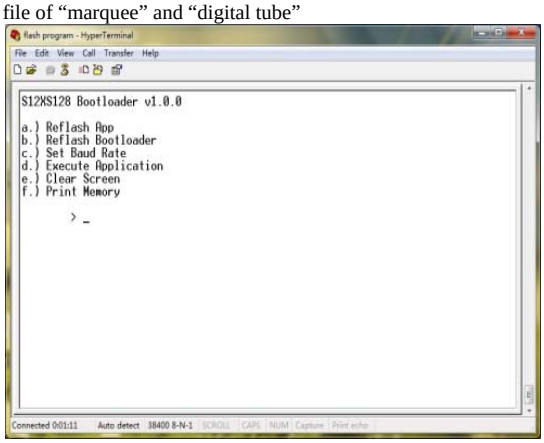

Figure 8. Bootloader online

Before the update of incremental update package，MCU does not monitor external request of application program, led.S19 file is written into MCU by the way of incrementally update. Input character "a"in the main interface, when the ECU receives a request, it will send "C" to the host computer in order to requist file transfer. Open the function of file transmission, select the target file, and choose the Xmodem transport protocol, click on the "send" , it will begin to execute the operation of writing program

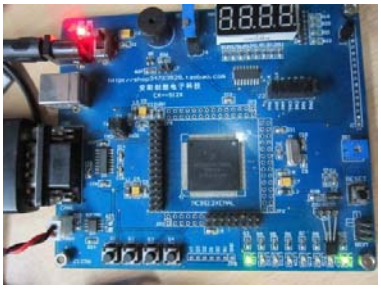

Figure 9. Running effect of marquee

Figure 10. Running effect update operation interface program of digital tube program

After update procedure is completed, "Program success on the main interface!" will be displayed in the main interface. At the same time, if the error occurs during the update process, an error message will also be informed. Next, input character "d" into the main interface; begin to enter the application program. Then ECU terminal starts running Marquee program. The running result is shown in Figure 9. After led.S19 was written correctly, start to perform programming of an incremental update packages. Input the update request "r" into the main interface, ECU will acquire this request when the application program is running start reset ECU, enter into Bootloader update service, and re-printed output interface as shown in Figure 8. Next, perform the same operation process as led.S19 programming, Selecting delta_nixie.S19 as the target file, complete programming of incremental update packages Eventually running the application program, the digital display operate normally, the effect is shown as Figure 10.

Experiments prove that incremental update could effectively reduce the size of update package data and improve the speed of update, but also verify the correctness and validity of Bootloader incremental update method which is presented in this paper.

IV. CONCLUSION

This paper is specific to the prospects of automotive ECU remote update application. The design requirement and realization project of "incremental update" is put forward for automotive ECU remote application update. This paper also thoroughly analysis the technology of Bootloader, ECU running process before entering update service and the detail process of ECU update service. Though the test, verify the correctness and feasibility of Bootloader incremental updates, Using the way of incremental update could effectively reduce the size of software update package and significantly reduce the time of update.

REFERENCES

[1] Charette R N.This car runs on code[J].IEEE Spectrum,2009,46(3):3

[2] Miucic R, Mahmud S M. Wireless multicasting for remote software upload in vehicles with realistic vehicle movement[C].Proc. SAE World Congress,2005:11-14

[3] Lightner, B., Botrego, D., Myers, C., Lowrey, L.H.Wireless diagnostic system and method for monitoring vehicles.U.S. patent 6636790,2003

[4] Rezgui J, Cherkaoui S, Charkroun O. Deterministic for DSRC/802.11p vehicular safety communication[C].2011 7th International Wireless Communications and Mobile Computing Conference,2011,p 595-600

[5] Chen, C.-H..Vehicle security system having wireless function-programming capability.U.S. patent 6184779 ,2001

[6] Shavit M, Gryc A, Miucic R.Firmware update over the air (FOTA) for automotive industry[R]. SAE Technical Paper,2007

[7] Nilsson D K, Larson U E. Secure firmware updates over the air in intelligent vehicles[C].Communications Workshops, 2008. ICC Workshops' 08.IEEE International Conference on. IEEE,2008: 380-384

[8] Nilsson D K, Sun L, Nakajima T. A framework for self-verification of firmware updates over the air in vehicle ECUs[C].GLOBECOM Workshops, 2008 IEEE. IEEE,2008: 1-5

[9] Miucic R, Mahmud S M. Wireless Reprogramming of Vehicle Electronic Control Units[C].Consumer Communications and Networking Conference, 2008. CCNC 2008.5th IEEE. IEEE,2008: 754-755