Revista Internacional de Investigación e Innovación Tecnológica

Página principal: www.riiit.com.mx

Reprogramming system for electronic control units on PHEV and EV vehicles with recharge mode 2. system

Sistema de reprogramación de unidades de control electrónico para vehículos eléctricos e híbridos conectables con sistema de recarga modo 2.

Sanchez-Hernandez, D[a]., Rivas-Silva H. A. [b]

[a] Departamento de posgrados, Centro de Tecnología Avanzada, 45030, Zapopan, Jalisco.
[b] SW Development Group, HELLA KGaA Hueck & Co, Tlaquepaque, Jalisco, México
daniel.shernandez88@gmail.com , hectorantonio.rivas@hella.com

Technological Innovation: Design and implementation of a reprogramming system for electric and hybrid vehicles with AC recharge systems.

Industrial Application Area: Automotive Industry, Automotive recalls, remote interface reprogramming.

Received: 26 April 2017.
Accepted: 21 August 2017.

Resumen

El presente trabajo propone un sistema de reprogramación para mejorar el proceso de llamados a revisión por software en la industria automotriz. El sistema se enfoca en reprogramar unidades de control electrónico de vehículos eléctricos e híbridos que sean capaces de recargarse por medio de un cable de alimentación y usando el cable de alimentación por sí mismo como medio físico de comunicación. El sistema se divide en dos fases; la primera fase describe como el equipo de suministro eléctrico del vehículo, el sistema utilizado para recargar lo vehículos, es utilizado para almacenar un dispositivo que sea capaz de conectarse a la red e identificar si existe un llamado a revisión por software disponible para el vehículo, si existiera, el sistema realizaría la descarga del archivo una vez que el vehículo es conectado a la red de alimentación. La segunda etapa describe el proceso cuando el archivo es descargado y transmitido por medio de la línea de recarga del vehículo utilizando la tecnología de comunicación PLC. En el vehículo, se describe un módulo de control que se encarga de recibir el archivo y de comenzar el proceso de reprogramación para el módulo al que va dirigido el nuevo programa. Un prototipo fue construido para demostrar el concepto del sistema, al finalizar, se realizaron pruebas simulando un escenario de solicitud de reprogramación.

Palabras Clave: Comunicación por línea de transmisión, Reprogramación de Módulos de Control Electrónico, Sistemas de recarga, Vehículos eléctricos.

Abstract

The present work proposes a reprogramming system to improve the software recall process in the automotive industry. The focus of the system is to re-flash an electronic control unit (ECU) of electric and hybrid vehicles that can be capable to be recharged with a wire and by using the energy wire itself as a communication physical layer. The system is divided in two phases; the first phase describes the electrical vehicle supply equipment (EVSE), the system used to recharge the vehicles, how it is used to embedded a system capable to connect to the Network and identify if there is a new software recall available for the vehicle, if an update is found, the EVSE will download the newest version once the car gets connected into the power network. The second phase describes the process for the file download into the vehicle thru the wire by using the power line communications. On the vehicle, the system describes an electronic control unit capable of receiving the file and start the download sequence to the target ECU. A prototype was built to demonstrate the feasibility of the concept and a test was performed simulating a re-flash request.

Key Words: ECU Reprogramming, Electric Vehicles, Powerline Communication, Recharge Systems.

Introduction

As any product, vehicles may contain production and design errors once they have been sold. The errors can represent a considerable economic risk for the company and a high safety risk for the customers.

Once the error has been detected, automotive companies need to inform to their customers about the issue and request them to take the vehicle into the car dealer to get the car fixed. The costs of those corrections are fully covered by the Original Equipment Manufacturer (OEM) which represent a big loss of the incomes and have a negative impact on their image and customer trust.

Over the last years, recalls have increased to a record number. Only in 2014, there were 700 recalls affecting over more than 60 million vehicles all over the world [1].

Table 1.0, shows the number of units recalled by each company.

Table 1. Recalls according with automotive company.

Company	No. of Units
General Motors	27,000,000
Honda	9,000,000
Chrysler	9,000,000
Toyota	6,000,000

The Software programmed into the Electronic Control Units (ECUs) may also be responsible to generate a recall. If a bug is found in the software that compromise the safety of any customer, a recall must be performed.

In July 2015, Charlie Miller and Chris Valasek reported an issue on certain Jeep vehicles. They could control the direction, transmission and breaks using the internet interface of the vehicle [2]. Chrysler had to announce a recall for 1.4 million vehicles in the United States.

Over the last years, the software size has been increased on the ECUs, as an example, in 2007, the BMW series 7 contained 65 Mb,

in 2010, the same car contained 1 Gb of software running in the vehicle [3]. The software incremented 93.5% in size.

A higher number of software means a higher number of bugs in the system; this problem make any company think about improving the way they make software recalls.

1.1 Electric Vehicles and supply equipment.

Electric vehicles can be categorized in three different groups: Hybrid Electric Vehicles (HEV), Plugged Hybrid Electric Vehicles (PHEV) and Electric Vehicles (EV) [4]. HEV and PHEV work in a similar way, the only difference is that PHEV can be recharged with an external power supply which makes them more fuel efficient compared with the HEV type.

The supply equipment for PHEV and EV are divided in conductive and inductive systems. Inductive systems work with two metal sheets that create an electromagnetic field over them and induce a current thru them. Conductive systems only use a wire that is connected directly to the car and the current goes thru the wire. At this moment, inductive systems are still not so efficient and they are still under development [5].

In the automotive industry exists two main standards that classify the supply equipment according with its characteristics. The specification SAE J1772 and the IEC 61851.

These two specifications describe in a similar way the different supply equipment' that exists. For alternate current, the SAE J1772 specify two supply modes, AC Mode 1 and Mode 2.

AC Mode 1 specify a simple home supply connector (NEMA 5-15) with the capacity to supply 120 volts with a maximum current between 12 and 16 amperes [6].

For AC Mode 2, the specification SAE J1772 requires to have an Electric Vehicle Supply Equipment (EVSE) between the home connector and the vehicle. The power supply for the AC Mode 2 is between 208 and 240 volts and the current must be less than 80 amperes [6].

Table 2 describes the modes and characteristics of the standard IEC 61851 [5].

Table 2. Recharge Modes according IEC 61851.

IEC Mode	Recharge time	Type of current	Maximum current
Mode 1	Slow	AC	16 A
Mode 2	Slow	AC	32 A
Mode 3	Slow/Fast	AC	Variable
Mode 4	Fast	DC	Variable

1.2 Power line communication and PEV – EVSE Association

The power line communication (PLC) is a communication protocol that uses the power line to inject a signal to transmit data over it [7]. In the last years, PLC has been used for home applications focused on smart home control and internet signal transmission.

One of the organizations that has worked on the standardization of the PLC is the HomePlug Powerline Alliance. The organization published in May of 2007 the specification HomePlug AV v1.1 which supports a data rate of 200 Mbps [8].

HomePlug Green PHY is a specification created by the consortium based on HomePlug AV. This specification was designed for applications that do not require high bandwidth, but are required to have high reliability [8]. The standard has been used by several products, in the automotive industry, this standard is used to communicate any pluggable electric vehicle (PEV) with the EVSE. The idea behind this association is to have an easier experience with electric vehicles so the customer only

would worry to plug the PEV to the EVSE and the charge and billing will be ensured.

In this paper, a system capable to reprogram an ECU using the recharge wire of an EVSE is proposed. The system shall also be capable of connecting to the network and download a new software version of any ECU related with the vehicle.

There are other studies where wireless reprogramming is proposed [9] [10], nevertheless they focus their system to work with the cellular infrastructure and several repetitions of the data to ensure a correct download process. However, the use of cellular infrastructure can increase the cost of the system as it would need to be rented for several periods to ensure all cars have received the new file. For the system proposed, it's only required to be connected to the Network.

1. *Functional description of the proposed system*

The system can be described in two phases. As shown on figure 1, the first phase comprises three blocks. The first block is the communication block, which is responsible to manage the interaction and validation between the EVSE and the OEM server through the Network. The second block is an application block, which is part of the entire EVSE and manages the request of approval of the customer to start the reprogramming sequence. And finally, the third module is defined as an embedded communication module which contains the sequence to notify and transmit the received file into an embedded board previously sent through the power line.

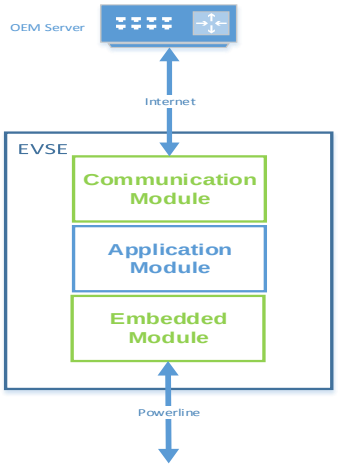

Figure 1. First phase (EVSE Module diagram).

The communication block contains an IP address which is used to communicate through the Network with the OEM server. Once the EVSE is turned on, the communication between the EVSE and the OEM server is established to detect if there's a new software version that needs to be updated. If the EVSE detects that there's a new software version for the registered car, it starts the request and validation sequence to download the new file. The Figure 2 shows the block flowchart of the communication process.

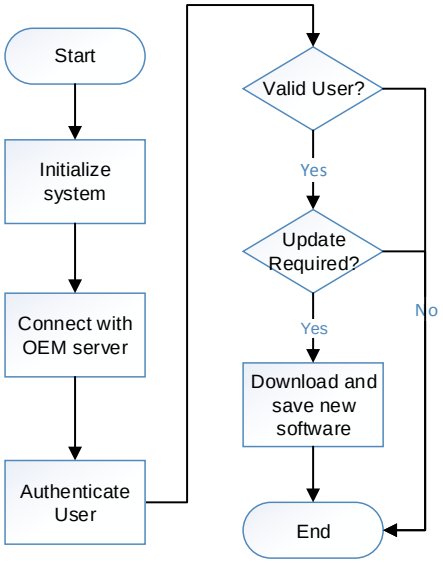

Figure 2. Communication block flowchart.

The application block is responsible to register new users and interact with the customer to get the approval to start the reprogramming session. In the register process, the user must specify the model year and the vehicle information number (VIN) of the car which will be used to verify the existence of new software in the OEM server. Once the communication block detects an update and has downloaded the file, the application will wait for the next recharge cycle of the EV or PHEV for requesting the customer the approval of the event. As the application module knows the size of the file to update, the time of the reprogramming session is computed and informed to the customer so the customer can decide according with his time availability.

The embedded module, packs the data of the new software file into a format that can be downloaded into the ECU. When the vehicle is connected to the EVSE, the embedded module detects the connection and informs the application module the presence of the vehicle.

If the application module provides the permission, the embedded module starts the process to validate the vehicle and finally transfer to the Gateway module the new software file though power line. To get the best approach into the standard download process, the embedded module performs the download process according with the standard ISO 14229 which is the most popular diagnostic standard in the automotive industry. The Figure 3 shows the process of the application module.

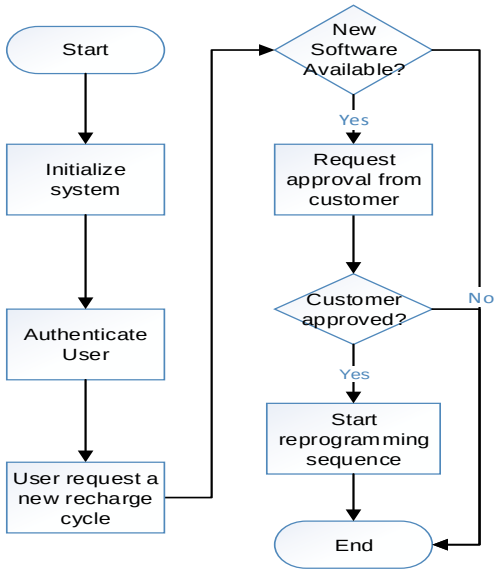

Figure 3. Application Module flowchart.

As shown in fig 4, the second phase comprises two modules. The first one is the Gateway module, it manages the reception and transmission to the EVSE and to the target ECU. The second one is the validation module, which is responsible to verify the integrity between the request and the file to download.

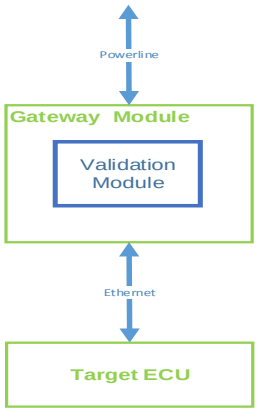

Figure 4. Second phase (Gateway).

The Gateway module is responsible of establishing communication with the EVSE and store the data received into a memory partition. Also, the Gateway module communicates with the target ECU and performs the download sequence according

to the standard ISO 14229. The gateway module is the only ECU in the vehicle with powerline communication capability and it's the responsible to interact with the EVSE module and the target ECU to reprogram.

The validation module performs two verifications in the second phase, the first validation is done between the EVSE module and the Gateway. The validation module ensures the compatibility between the car registered in the EVSE and the actual car. The second validation is done between the file downloaded and the target ECU. As part of the download sequence, the module compares the number part of the ECU and the one in the file and deny or allow the process according with the result.

In previous proposed systems, it is uncertain the right moment to perform the reprogramming because the vehicle needs to be under certain conditions to ensure the safety of the passenger. In this work, the reprograming is performed in stable conditions when the car is recharging and the customer is asked and notified about the process. Nevertheless, there's still a risk that the connector could be disconnected when the reprogramming is in process. This risk can be reduced if the connector is locked when the process is done.

2. Implementation

The Figure 5 shows the entire system interconnected with an Ethernet switch.

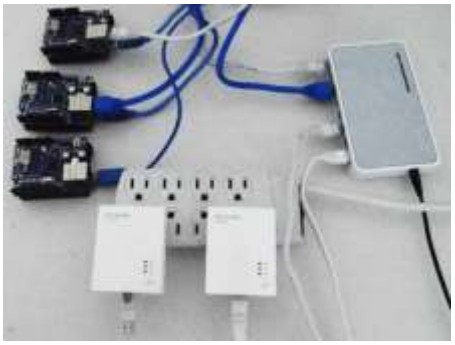

Figure 5. System Prototype Implementation.

Three Arduino UNO boards are used as EVSE, Gateway and the target ECU. The boards include a shield capable to receive and transmit Ethernet frames. The Arduino UNO has an 8-bit microcontroller ATmega328P. This microcontroller is produced by the company ATEMEL, it has 32 KB of Flash Memory, 2KB of RAM and 1KB of EEPROM. The clock speed running by the microcontroller is 16MHz.

Two PLC modules AV500 Nano to transmit and receive data thru Powerline Communication. The modules support the standards HomePlug AV, IEEE802.3 and IEEE802.3u. Having one Ethernet port 10/100Mbps to connect the device to communicate. Each module can encrypt data with the standard AES 128-bits.

3. Experimental results and Analysis

As described in chapter 2 "Functional description of the proposed system", the system consists of two phases for its development. The first one, describes the acquisition of new software version, the second phase manages the transmission and the download process. Figure 5 describes the system architecture of phase one and two.

Figure 6. Block diagram of prototype.

For the phase one, the Communication and Application modules were simulated using a PC with Internet connection. For the prototype development, it was used the

Google Drive Server to have the closest simulation of cloud storage with encrypted data.

For phase two, the validation module in this prototype was implemented in software and was the responsible to validate the file received and the target ECU.

A test application used to verify the feasibility of the project was developed using Atmel Studio. The application consisted of one push button and seven LEDs, when the push button is pressed, the LEDs turned on sequentially starting from the middle and having a small delay between them. Imaging the OEM wants to change this behavior; a new application needs to be reprogramming for safety purposes in the module. The new behavior contained in the new software version, modify the way the LED's are turned on. Instead turning on the LED's starting from the middle, the OEM requires to start from the right and continue turning on with a new diming functionality to the left.

Phase One

The application module is responsible of interactions between the client and the EVSE. In this case, if a new software version is available to download, the application module acts as interface between the client and the system. For the prototype, an interface was developed using Visual Studio 2013 with C#.

To simulate the OEM server, a file and synchronization service was used to store the new file to download. When the system is powered on, an authentication process is performed between the file system and the EVSE module.

The application interface has communication functionalities, user notification and also the download process status. As shown in Figure 7, the Port button shows all the available serial ports in the computer, once the right Port is selected, the "Connect" button is selected to establish a connection between the EVSE and the Arduino board. On the other hand, the button "Download New Software" initialize the communication between the EVSE module, in this case the PC, and the Google Drive Server.

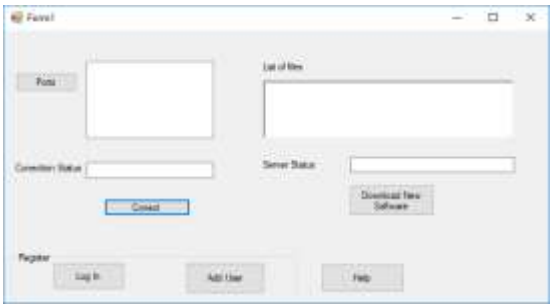

Figure 7. Application software.

To be able to establish communication with the Google Drive Server and access the files, it was necessary to perform a validation sequence were two data elements were used: an email provided by Google specific to the folder and a generated key also by using the Google Drive account. Using the method "*Drive.api.Authentication.AuthenticateServiceAccount*" of the class "*Daimto*" and adding as argument the personal email and the authentication key is how it can be accessed to the Google Drive Server.

Once the session is validated, the module verifies the file stored in the folder. The file shall contain all necessary elements to satisfy the download requirements. To perform this step, an element was created containing a search list comprising the name "EVSE". The method *Drive.api.DaimtoGoogleDriveHelper.GetFiles* was used of the same class *Daimto.* Ideally, in this step, the module should have to search in a folder with a lot of files inside of it and with different parameters. Nevertheless, the word "EVSE" was used to demonstrate the concept. To know if the file stored in the folder has been modified since the last

connection, the method *LastWriteTime* can be used by the application.

The next step once the file and user are validated, is to download the file. In order to perform this task, the method *downloadFile* of the same class *Daimto*. As argument of the method, it was specified the folder to store it.

An option was created to add the user's information. The user's name, the OEM, the Vehicle Identification Number and the Model of the vehicle associated to the user are required to create a new user and stored in the system. The figure 8 shows the format of the Register where the previous information is required. This data is used at the moment to search the right folder where new files will be stored according with each vehicle. Also, a validation process between the EVSE and the vehicle was performed and this same information was used.

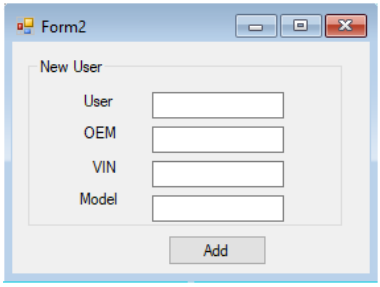

Figure 8. New User Register.

If new software version was detected and needed to be downloaded to the vehicle, a strategy to detect the vehicle connection was implemented. The embedded EVSE module, which is an Arduino board, is used to interact with the network between the EVSE and the car. If the system is turned on, using the Ethernet port, a periodic request message with the ID 0x705 is sent every second for 10 times, if the request is responded with the data field 0x55AA, the connection between the Gateway (vehicle) and the EVSE module (Arduino EVSE) is established, otherwise a new request is generated every minute. Once the module has stablished communication, a request is generated to have the user's approval. The Figure 9 shows the request generated by the application tool. If the user accepts, the download transmission is started.

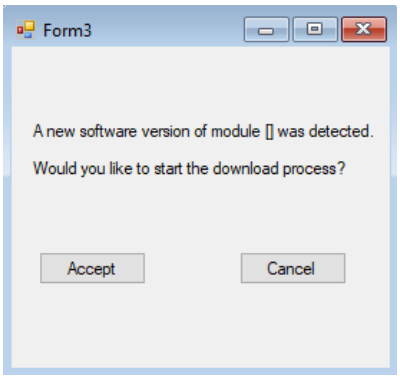

Figure 9. New User Register.

The Figure 10 shows the application when the connection between the PC and Arduino has been established and the connection with the server is authenticated.

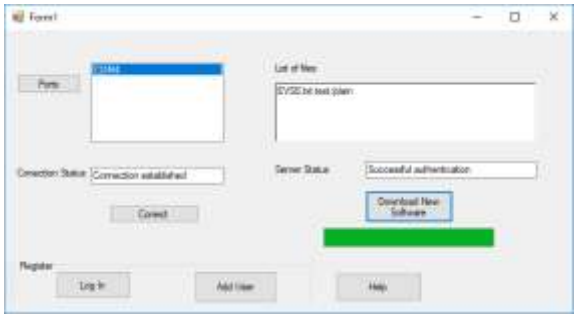

Figure 10. Connection with Server and Arduino board.

Phase Two

Phase two can be divided in two parts, the first part corresponds to the transmission of the Software from the EVSE module to the Gateway module. For this part, each module has one Power Line Communication module used to transmit and receive the data from the Ethernet bus into the power line. The second part of phase two consists of performing the download sequence into the selected ECU. The reference of this

sequence was taken from the ISO 14229 which is a diagnostic specification used in the automotive industry to diagnose the ECU and also re-flash ECU's. Figure 11 shows the Power Line Communication modules.

Figure 11. Power Line Communication Modules.

For the first part of Phase two, the EVSE and Gateway modules were programmed using Atmel Studio, each module was simulated using an Arduino UNO board and the Ethernet shield which contains the integrated circuit W5100 used to transmit and receive Ethernet frames. Figure 12 shows the Arduino and Ethernet shield.

Figure 12. Arduino boards with Ethernet Shield.

The EVSE module was configured with the IP: 192.168.15.20 and the Gateway module has the IP: 192.168.15.250. The sequence to transfer the new file from the EVSE to the Gateway is the following; a) Once the EVSE is turned on and connected to the vehicle, a periodic request will be sent with the data [0x02, 0x3D, 0x00]; b) If the Gateway is present, it will answer with the data [0x55, 0xAA]; c) After the Gateway answer, the EVSE perform a validation process were the VIN number is requested, for this, a DID is sent to read the parameter [0x02, 0x22, 0xF0, 0x00]; d) The Gateway will send back a message with its' VIN, this value is used by the EVSE to compare it with the software received by the Application; e) If the values are the same, the EVSE will send to the Gateway using the service "Transfer Data" the new software file; f) If transmission was successfully received, the Gateway will send back an Acknowledge to the EVSE.

After data transfer is finished, the Gateway performs a download process into the target ECU using the same process that any other diagnostic tool would perform. A test was performed using two different applications. Before and after the reprograming process, the memory was read and stored in a hex file. The Figure 13 shows a small portion of the differences between these two files. The bootloader starts at address 0x3800 so from this memory address, the files shall not be different.

Figure 13. Files before and after reprogramming process.

4. Conclusions and future work

The reprogramming system for pluggable electric vehicles was described with its concept and design. A prototype was implemented to demonstrate the easy implementation of the system and its feasibility of the entire process. A test using two applications was performed, the download sequence was triggered by the Application and downloaded to the target ECU. As the system works using the power line communication and the Network, it is

also feasible to use the recharge cable of the EV and PHEV to transmit from an OEM server from any place with Network connection.

Further work considers to protect the system against cyberattacks since the EVSE will be connected to the Network. Also, security in the Ethernet transmission must be implemented. An evaluation on real conditions must be considered in the future.

5. References

[1] Vlasic, B., & Stout, H. (2014). Auto Industry Galvanized After Record Recall Year. *Business Day*. Retrieved from http://www.nytimes.com/2014/12/31/business/a-year-of-record-recalls-galvanizes-auto-industry-into-action.html?_r=0

[2] Ring, T. (2015). Connected cars–the next targe tfor hackers. *Network Security*, *2015*(11), 11-16.

[3] Pretschner, A., Broy, M., Kruger, I. H., & Stauner, T. (2007). *Software engineering for automotive systems: A roadmap*. Paper presented at the 2007 Future of Software Engineering.

[4] (Maggetto, G. and Van Mierlo, J. 2000. Electric and electric hybrid vehicle technology: a survey. Electric, Hybrid and Fuel Cell Vehicles (Ref. No. 2000/050), IEE Seminar. S.l.: IET, pp. 1/1-111.

[5] Ayob, A., Mahmood, W., Mohamed, A., Wanik, M., Siam, M., Sulaiman, S., Azit, A. and Ali, M. 2014. Review on electric vehicle, battery charger, charging station and standards. Review on electric vehicle, battery charger, charging station and standards. Vol. 7, no. 2, pp. 364-373.

[6] Bohn, T. and Chaudhry, H. 2012. Overview of SAE standards for plug-in electric vehicle. IEEE PES Innovative Smart Grid Technologies (ISGT). S.l.: IEEE, pp. 1-7.

[7] Strobl, M., Waas, T., Moehne, S., Kucera, M., Rath, A., Balbierer, N., & Schingale, A. (2012). Using Ethernet over powerline communication in automotive networks. Paper presented at the Intelligent Solutions in Embedded Systems (WISES), 2012 Proceedings of the Tenth Workshop on.

[8] Latchman, H., Katar, S., Yonge, L. and Gavette, S. 2013. Homeplug AV and IEEE 1901: A Handbook for PLC Designers and Users. 1. S.l.: John Wiley & Sons.

[9] Miucic, R. and Mahmud, S. 2005. Wireless multicasting for remote software upload in vehicles with realistic vehicle movement. *Proc. SAE World Congress*. S.l.: SAE, pp. 11-14.

[10] Miucic, R. and Mahmud, S. 2008. Wireless Reprogramming of Vehicle Electronic Control Units. *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*. S.l.: IEEE,