Development of a method for automotive engine control unit modification for achieving different operating modes using software correction

N Nikolov$^1$ and D Gotseva
Faculty of Computer Systems and Technologies, Technical University – Sofia, 8 Kliment Ohridski Blvd., 1000 Sofia, Bulgaria

$^1$E-mail: n.nikolov@tu-sofia.bg

Abstract. This article describes the development of a method for automotive fuel injection engine control unit (ECU) modification for aimed at achieving operating modes such as Economy (ECO) and Sport using software alteration. The article researches the offered pattern and demonstrates the experimental staging. An automotive ECU, microcontroller and software are used for the experimental stage. The ECO mode is used to achieve fuel economy and the Sport mode are used to achieve more engine power. The microprocessor used to switch between ECU modes is PIC18F14k22 and the ROM flash memory which stores the firmware for the two modes is AM29F400BB. The ECU used in the research is Sagem s2000. In the article, both the hardware and the software modifications are described as well as the experimental results.

1. Introduction
In order to achieve operating modes such as maximum Economy and Power from the same engine, it is necessary to approach it by modification in the software of its operation. Environmental norms are an important thing but they limit the full potential of the internal combustion engine. By limiting the maximum power of a gasoline engine we achieve greater environmental friendliness, smaller amount of harmful gases emitted into the atmosphere and as a result increased economy. Maximum engine dynamics are achieved by higher fuel consumption and maintenance of richer mixtures. Through software setting – using the so called Maps tune or Chip Tuning – of engine modes such as ECO and Sport full coverage of engine capabilities, such as fuel economy and maximum developed power and dynamics, is achieved.

2. Theory and architecture
In order to achieve full fuel economy for gasoline engine, it is necessary to make the mixture lean to a certain extent. To achieve full power, the mixture has to be enriched. For the complete burning of 1 kg of gasoline requires 14.64 kg of air., the mixture must be somewhere between 15 and 16 air/fuel ratio. To achieve full economy and for full power it must be about 13, 13.5 air/fuel ratio. Table 1 shows the different mixtures of gasoline and air [1]. In order to achieve fuel saving and power, maps with $16 \times 16$ values such as Lambda oxygen sensor for different modes, VE table (volumetric efficiency enrichment), ignition angle and throttle sensitivity and accelerator pedal – Driver Wish Torque are...
required to be changed. A 1.4-litre engine: TU3JP with 75 horse power is used for the experiment. The ECU used for fuel injection is Sagem s2000.

Table 1. Various mixtures of gasoline and air (air/fuel ratio) and their application.

| Air/fuel ratio | Lambda | Comment                                      |
|----------------|--------|----------------------------------------------|
| 9.0            | 0.61   | Black smoke/low power                        |
| 11.5           | 0.79   | Approximate rich best torque at wide open throttle |
| 12.2           | 0.83   | Safe best power at wide-open throttle        |
| 13.3           | 0.91   | Approximate lean best torque                 |
| 14.64          | 1.00   | Stoichiometric air/fuel ratio (chemically ideal) |
| 15.5           | 1.06   | Lean cruise                                  |
| 16.5           | 1.13   | Usual best economy                           |
| 18.0           | 1.23   | Carbureted lead burn limit                  |

For the optimal operation of the gasoline engine it is necessary to increase the ignition angle by 1 degree relative to the factory settings (figure 1). Figure 1 shows the ignition angle map, which is size 16×16. This map is the same for both ECO and Sport modes.

![Figure 1. Ignition angle map 16×16.](image)

ECO and Sport modes differ in terms of Lambda oxygen sensor, VE (volumetric efficiency enrichment) and Driver Wish Torque. Figure 2 shows the Lambda oxygen sensor map for ECO mode, where the target air/fuel ratio for full load is 15. Figure 3 shows the map of Lambda Oxygen Sensor for Sport mode, where the target air/fuel ratio for full load is 13.2.

![Figure 2. Lambda oxygen sensor for ECO mode, the target air/fuel ratio for full load is 15.](image)
Figure 3. Lambda oxygen sensor for Sport mode, the target air/fuel ratio for full load is 13.2.

Of great importance for achieving greater economy is the driver wish map, which decides how much fuel the driver wants the ECU to put into the motor. Figure 4 shows the ECO mode driver wish map and Figure 5 shows the driver wish map for Sport mode. From the data in the figures it can be concluded that the low load values for ECO mode are quite reduced and in the modes between 75 and 100%, which correspond to WOT (Wide opening throttle), they are unchanged compared to the original ones. This is done in order for the vehicle to reach the desired power and torque in normal driving. For Sport mode, it appears that the values are well increased from original ones in order to ensure maximum power, acceleration and torque.

Figure 4. Driver wish map for ECO mode.

Figure 5. Driver wish map for Sport mode.
The following formula summarizes the work of the ECU fuel injection software, which is responsible for the operation of the prepared fuel by injectors. The opening times of injectors are adjusted by Pulse width modulation [1]:

\[
\text{Pulse width} = \left(\frac{\text{MAP Voltage}}{5} \times \text{UAP} \times \left(\frac{\text{VE Absolute}}{100}\right) \times \text{TPS} \times \text{CTS} \times \text{IAT} \times \text{EGO} \times \text{SE}\right) + \text{POT} + \text{BTO},
\]

where: MAP – manifold absolute pressure; UAP – user-adjustable pulse width; POT – pulse width offset time at idle; TPS – throttle position change; CTS – engine coolant temperature; IAT – intake air temperature; EGO – required air/fuel correction coefficient based on the exhaust gas oxygen; SE – starting enrichments. All correction coefficients add percentage changes to VE absolute percentage with the exception of battery voltage correction BTO.

3. Experimental stage and hardware realization

In order to switch between ECO and Sport modes, ECU Firmware must be changed. For this purpose, two types of firmware ECO and Sport are prepared. The ECU Sagem s2000 use NOR flash memory AM29F200BB [2], which is 256 kb. In order to be able to combine both modes of operation, we need to use NOR flash memory with double the size of AM29F400BB, which is 512 kb size [3]. The switch between the two modes must be done by submitting logical low and high levels in order to select the first 256 kb block where Sport is and the second 256 kb block where ECO is stored. (figure 6).

![Figure 6. Memory map of firmware in nor flash memory AM29F400BB.](image)

The two modes are selected by submitting low and high level to am29F400BB with address pin A17. But to make the mode switch, Sport and ECO, selection buttons, an indicator of the status of the mode and Reset on the ECU Sagem s2000 processor. The ECU uses a SAC c167CS microprocessor, which is 16 bit, and when switching the new mode, the SAC c167CS must be restarted and IP (instruction pointer) starts again from the reset vector [4]. The need to RESET a SAK c167CS is due to the fact that the two firmware for ECO and Sport modes have different Check Sum for data validity.

Figure 7 shows the time chart to be performed to switch between engine operating modes. For this purpose, the PIC18F14K22 micro controller is used to control the switching between operating modes. Figure 8 shows the picture of the experienced staging – the ECU Sagem s2000 used with pic18F14K22 controller.
4. Software realization

For the Firmware [5] implementation of the PIC18F14K24 micro controller, the programming language is C. RTOS operating system [6-8] with cooperative scheduling Task [9, 10] are created. This RTOS has a TASK which is responsible for reading the buttons, the switching of the upper and lower 256 kbytes, and reseing the SAK C167CS controller of Sagem s2000. The main program contains the following lines:

```c
Init_CPU_Freq_Clock ( void );
Init_Hardware_GPIO ( void );
Init_Hardware_TIMER ( void );
Init_Hardware_IRQ ( void );
Init_Read_EEPROM ( void );
Scheduler_init ( void );
for ( ; ; ); (2)
```

where the initialization of Registers, Timers, CPU Frequency Clock, EEPROM and Scheduler TASK is performed. The main TASKs are: “Task_Scan.Buttons”, “Task_Switch.ECU_Mode”, “Task_Reset.SAC_TC167”, “Task_Control.LED.ECO.Dashboard”. The chosed mode between ECO or Sport are saved in the EEPROM memory. The software executes the logic based on the time chart (figure 7). The following lines show the TASK for RESET of SAK C167CS:

```c
void Task_Reset_SAC_TC167( void ){
    if( FLAG_Enable_RESET_SAC_C167CS_GPIO == TRUE ){
        FLAG_Enable_RESET_SAC_C167CS_GPIO = FALSE;
        delay_ms( 100 );
        RESET_SAC_C167CS_GPIO = LOW ;
        delay_ms( 500 );
        RESET_SAC_C167CS_GPIO = HIGH ;
    }else{
        RESET_SAC_C167CS_GPIO = HIGH ;
    }
    return;
}(3)
```

Figure 7. Time diagram for operating modes CONTROL ECO and Sport via the PIC18F14K22 micro controller.

Figure 8. The test staging – the ECU Sagem s2000 and the PIC18F14K22 micro controller.
 TASK itself shows how the “delay_ms(100)” function performs a software delay of 100 ms required to prepare the reset time. Cooperative scheduling in RTOS kernel is shown in these rows:

```c
void  Scheduler_init( void ){
    for ( ; ; ){
        Task_Scan.Buttons              ( void );
        Task_Switch.ECU_Mode           ( void );
        Task_Reset.SAC_TC167           ( void );
        Task_Control.LED.ECO.Dashboard ( void );
    }
    return;
}
```

5. Experimental results
Several approaches were used to study the results obtained – Dyno Jet bench which measures the power of the wheels of the car and driving on a particular route in order to compare the achieved economy. The results of Dyno Jet are shown in figure 9. The measured parameters are Torque in Nm, Power in hp and air/fuel ratio. The graph shows that the Sport mode achieves a maximum Torque 113.02 Nm and Power 64.98 hp. Comparable air/fuel ratio shows that the engine holds mixtures between 14.8÷15.20 of ECO mode for better combustion of the fuel and in Sport mode – 12 for maximum power. As a result of driving the vehicle in urban – ECO mode, the reduction in fuel consumption from 9 l/100km to 8 l/100km is detected and extra-urban from 5.4 l/100km to 5.0/100km.

![Figure 9. Dyno Jet measurements with parameter comparison such as Torque in Nm, Power in hp and air/fuel ratio.](image)

6. Conclusion
This article describes the achievement of operating modes – maximum Economy and Power from the same engine. This was achieved through a modification in the software of its work. Two ECO and Sport modes are made, thus achieving maximum power and when it is not needed select maximum economy. This is done with ECU maps trimming. The paper describes how the hardware and software
realization of the system is created. The architecture and operating model are shown. A cooperative RTOS – a real time operating system is created for controlling the process. The experimental results are shown in figure 9: the measurement parameter such as Torque in Nm, Power in hp and air/fuel ratio in both modes: ECO and Sport. The contribution of the authors consists in the realization of the software architecture and the cooperative RTOS and also in the experimental staging and the experiment.

References
[1] Hartman J 2013 How to tune and modify automotive engine management systems (Motorbooks Workshop)
[2] Bogdanov L and Ivanov R 2019 Flash programming low power microcontrollers over the Internet IEEE 28th Int. Conf. Electronics (ET) (Sozopol, Bulgaria)
[3] AM29F400B Datasheet (PDF) – Advanced Micro Devices, www.alldatasheet.com, 03.2020
[4] Colaco F and Cardoso F 1999 Flying objects: a modular real time object operating system 11th IEEE NPSS Real Time Conference (Sante Fe, NM, USA) pp 543–6
[5] Bogdanov L V 2019 A Wi-Fi to UART bridge for firmware updates of microcontrollers IEEE 28th Int. Sci. Conf. Electronics (ET) (Sozopol, Bulgaria)
[6] Hoffmann M, Dietrich C and Lohmann D 2013 dOSEK: A dependable RTOS for automotive applications IEEE 19th Pacific Rim International Symposium on Dependable Computing (Vancouver) pp 120–1
[7] Xiao D and Zhang J 2011 A scheme and debugging of RTOS based on OSEK/VDX Int. Conf. on Electric Information and Control Eng. (Wuhan) pp 2924–8
[8] Shuaiqing X, Yang W, Chengjun T, Mingzhi S, Yongsheng Z and Tao X 2011 Transplantation of RTOS uC/OS-II on TriCore processor Int. Conf. on Electric Information and Control Eng. (Wuhan) pp 5598–601
[9] Seo SH, Lee S, Hwang SH and Jeon J W 2006 Analysis of task switching time of ECU embedded system ported to OSEK(RTOS) SICE-ICASE Int. Joint Conf. (Busan) pp 545–9
[10] Yang G, Zhaohui W and Long Y 2005 AlphaOS, an automotive RTOS based on OSEK/VDX: design and test Proc. IEEE Networking, Sensing and Control (Tucson) pp 174–9