IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 5, 2018, accepted August 31, 2018, date of publication September 26, 2018, date of current version October 17, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2870061

Model-Based Development of an Engine Control Module for a Spark Ignition Engine

BRUNO MARTIN DE ALCÂNTARA DIAS[1], ARMANDO ANTONIO MARIA LAGANÁ[1], JOÃO FRANCISCO JUSTO[1], LEOPOLDO RIDEKI YOSHIOKA[1], MAX MAURO DIAS SANTOS[2], (Senior Member, IEEE), AND ZONGHUA GU[3], (Senior Member, IEEE)

[1]Escola Politécnica, Universidade de São Paulo, São Paulo 05424-970, Brazil
[2]Departmento de Eletrônica, Universidade Tecnológica Federal do Paraná, Ponta Grossa 84016-210, Brazil
[3]Department of Applied Physics and Electronics, Umeå University, 901 87 Umeå, Sweden

Corresponding author: Zonghua Gu (zonghua.gu@umu.se)

This work was supported in part by Brazilian Agencies FAPESP, CNPq, Capes, and the U.S. Department of Commerce under Grant BS123456 and in part by NSFC under Grant 61672454.

ABSTRACT A Spark ignition (SI) engine is a complex, multi-domain component of the vehicle powertrain system. The engine control module (ECM) for an SI engine must achieve both high performance and good fuel efficiency. In this paper, we present a model-based development methodology for an open architecture ECM, addressing the entire development lifecycle including a control algorithm design, parameter calibration, hardware/software implementation, and verification/validation of the final system, both with bench tests on a dynamometer and in a real vehicle on the road. The ECM is able to achieve similar performance as the original proprietary ECM provided by the original equipment manufacturer. Its flexible and modular design enables easy extensibility with new control algorithms, and development of new engine types.

INDEX TERMS Engine management system, spark ignition engine, engine control module, electronic control unit.

I. INTRODUCTION

Over the last decades, the automotive industry has been increasingly incorporating new engine technologies [1], with major innovations in the Electrical/Electronic (E/E) systems [2]. The Engine Management System (EMS) consists of the Engine Control Module (ECM), as well as various electrical/electronic components such as sensors, actuators, relays and. It monitors the real-time operating conditions of the engine and issues control commands to multiple actuators, such as injector, spark plug, and throttle valve [3]. The ECM receives input from sensors and uses this information to control engine speed and torque through fuel injection, throttle, and ignition subsystems [4], [5], with the objective of optimizing engine performance and fuel efficiency, and to comply with emission standards.

Commercial ECMs are typically developed for a specific engine, in which the control strategy and calibration parameters are established for specific engine working conditions. Additionally, due to proprietary nature of commercial products, details of the ECM hardware and software are generally not disclosed. Researchers lack an ECM platform, with an open architecture, which could allow experimentation on a wide range of conditions, from experiments with new control theories, exploratory research by varying control functions or parameters, and new control algorithms. Here, we present a methodology for developing an ECM for a SI engine, including the complete workflow of design, development, testing, and calibration of the following modules: 1) air–fuel ratio control; 2) electronic throttle control; 3) idle speed control, and 4) ignition timing control. This methodology will accelerate and facilitate flexible integration of new control and sensing techniques in futures research work, in order to improve engine efficiency and reduce emission.

There are several references related to engine management, but we have been motivated to use two important and classical references that handle about engine modeling and control. Then, [16] and [17] show important aspects for engine management systems in level of modeling, simulation, control strategies, experiments and calibration. Therefore, the textbooks served as background to us design control strategies and experimental aspects.

This paper is organized as follows. Section II presents the development workflow based on an SI engine in a commercial vehicle; Section III shows the design of subsystems

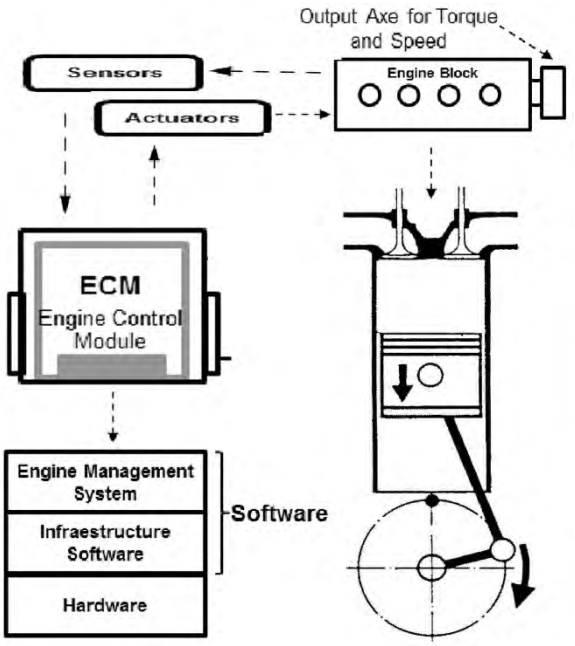

FIGURE 1. The overall architecture of the EMS controlling an SI engine.

controllers for injection, admission and ignition; Sections IV and V show the hardware and software design for the ECM software and hardware; Section VI shows the verification procedures for subsystems of injection, admission and ignition; Section VII shows detailed descriptions of the validation process to assessment of the engine performance and comparison by means of the ECM completely developed from scratch. Finally, Section VIII shows the main outcomes and lessons learned reached on this challenge project which enabled show in details the experimental results of the ECM on a real engine.

II. DEVELOPMENT WORKFLOW

Fig. 1 shows the overall architecture of engine block, sensors/actuators, ECM, hardware/software and engine management system. Fig. 2 shows the ECM that runs several software tasks for controlling fuel injection time, spark advance angle, dwell time, idle speed, and electronic throttle valve opening angle.

The gas pedal input serves as the main reference for the entire controller. The reference engine speed, measured in Revolutions Per Minute (RPM), is computed from the gas pedal and used as reference input to the RPM Proportional-Integral (PI) controller, which evaluates the error between the reference engine speed and the actual engine speed, and computes the throttle valve value based on the well-known PI control algorithm. The throttle valve value is used as the reference input to the throttle PI controller, which sends Pulse-Width Modulation (PWM) commands to the physical throttle valve, with feedback signal from the Throttle Position Sensor (TPS). The advance ignition controller takes as input the engine speed in RPM and pre-stored maps, and computes the spark advance. The dwell time controller takes as input the engine speed, voltage of the

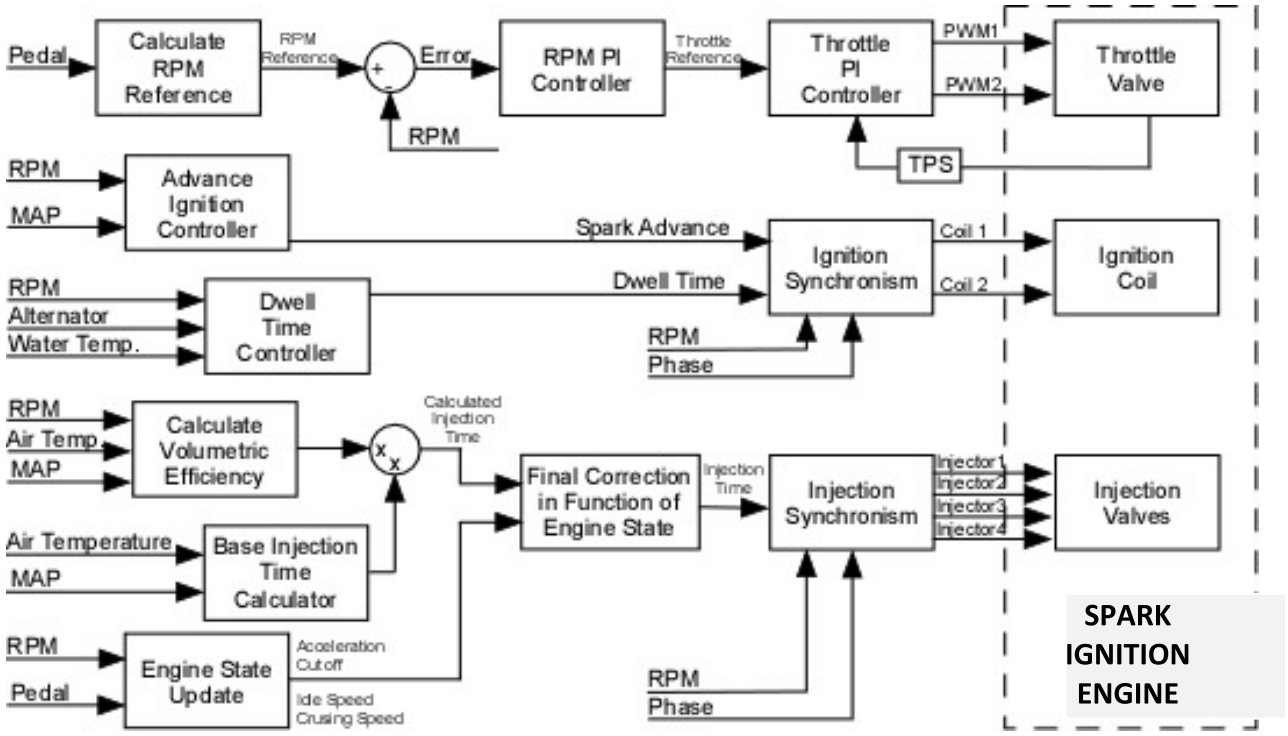

FIGURE 2. Block diagram of the control functions of the EMS.

alternator and the engine coolant temperature, and computes the dwell time. The injection time is computed based on the density of air inside the engine intake. The final injection time is adjusted based on the current engine state and the volumetric efficiency. This adjustment increases the injection time during acceleration and decreases the injection time, or cuts off injection, during deceleration.

Fig. 3 shows the overall model-based development workflow of the ECM.

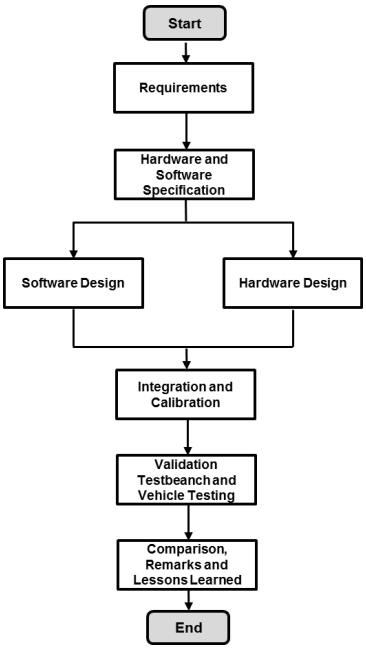

FIGURE 3. Workflow of model-based ECM development.

III. CONTROLLER DESIGN

Fig. 4 shows the three subsystems of the engine control system [7]: admission, injection, and ignition, which control air mass, fuel mass, and electric spark timing, respectively.

Fig. 5 shows typical sensors and actuators in the ECM. Besides the main capability to manage and control all those sensors and actuators, it is important to have some extra general-purpose input/output pins in the EMS developed here. Those extra pins could allow implementing more functionalities in future research.

Fig. 6 shows the nominal engine timing of the EA 113 engine (described in Section VII), operating in a

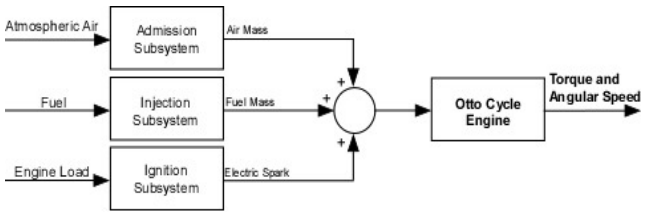

FIGURE 4. The SI engine control system.

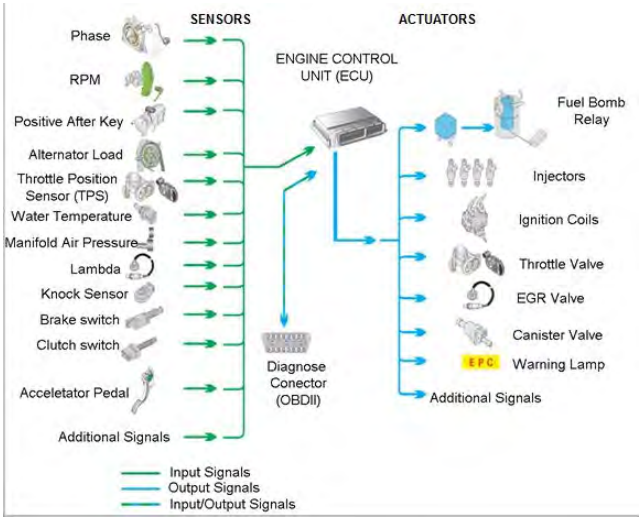

FIGURE 5. Typical set of sensors and actuators in a SI engine.

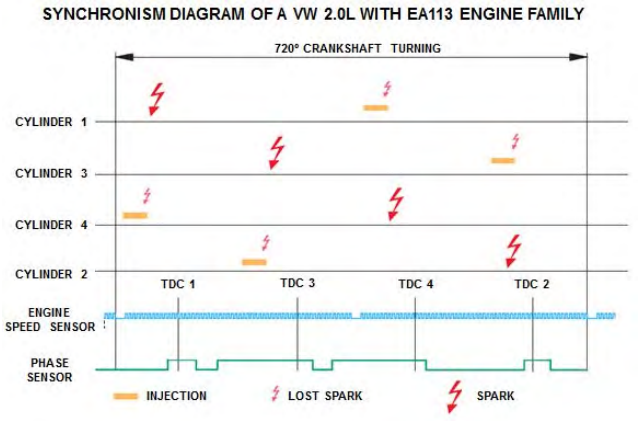

FIGURE 6. Nominal fuel injection and spark timing for EA 113 engine.

dual-coil system. Each coil is connected to two twin cylinders (1-4 or 2-3). While one cylinder receives a spark in the end of compression phase, its twin receives a spark in the end of exhaustion phase. The ECM developed here can control up to 8 individual coils, allowing this platform to control V8 engines.

A. IGNITION SUBSYSTEM CONTROLLER

The ignition subsystem must accurately perform the combustion of the compressed air/fuel mixture inside the engine cylinder, under dynamic operating conditions with fluctuations in the flow patterns of that mixture. The time interval during which the coil must remain energized should be long enough (about 5.8ms according to documentation) to ensure a spark capable of initiating combustion of the fuel mixture. The ignition timing is determined from the engine rotation speed and intake air mass, and controls the spark advance angle relative to Top Dead Cylinder (TDC) for all rotation speed combinations and air mass loads.

B. INJECTION SUBSYSTEM CONTROLLER

The fuel injection subsystem controls the amount of fuel, based on input parameters (intake air mass, manifold air pressure, and atmosphere air temperature), and the current engine operating condition, to ensure stoichiometric fuel mass ratio inside the engine chambers, e.g., the fuel/air mixture ratio should be high to warm up the engine in cold starts and providing the optimum torque according to the driver command [2].

The ideal gas equation is used to correlate air density, pressure, and temperature in the intake manifold. Similarly, it is possible to determine the mass of required fuel for the engine operation state. The injection time is a function of the fuel mass density speed according to equation (1) [8]:

$$T_{inj} = \frac{\rho_0 \left(\frac{\mathrm{P_{MAP}}}{\mathrm{P_0}}\right) \left(\frac{\mathrm{T_0}}{\mathrm{T_{MAP}}}\right) \mathrm{N_{cylinder}} \mathrm{V_{piston}} \mathrm{E_v}}{V_{injector\_flow} \mathrm{N_{injectors}} \left(\mathrm{A/C}\right)} \tag{1}$$

where:

T_{inj}	Injection Time output [μs]
ρ_0	Atmosphere Air Density at Sea Level = 1.299 $\left[\frac{kg}{m^3}\right]$
P_0	Atmosphere Air Pressure at Sea Level = 101.3 [KPa]
T_0	Ambient Temperature = 288 [Kelvin]
P_{MAP}	Manifold Absolute Pressure [KPa]
T_{MAP}	Manifold Intake Air Temperature [Kelvin]
$N_{cylinder}$	Engine Cylinder Number [$*$]
V_{piston}	Engine Piston Volume [m^3]
E_v	Volumetric Efficiency [$*$]
$V_{injector\_flow}$	Injectors Valve Fuel Flow $\left[\frac{\mu g}{\mu s}\right]$
$N_{injectors}$	Number of Injection Valves [$*$]
$\left(\frac{A}{C}\right)$	Lambda to reach stoichiometry fuel ratio [$*$]

Most of the parameters in equation (1) are constant. The parameters that are dependent on the current state of the engine are: manifold absolute pressure, intake air temperature, and volumetric efficiency (E_v). Equations (2), (3), and (4) illustrate how the volumetric efficiency are calculated by the ECM:

$$E_v = \frac{\mathrm{E_{real}}}{\mathrm{E_{theoric}}} \tag{2}$$

$$E_{real} = \mathrm{V_t} \left(\frac{\mathrm{P_{MAP}}}{\mathrm{R} \cdot \mathrm{T_{MAP}}}\right) \left(\frac{\mathrm{RPM}}{2}\right) \tag{3}$$

$$E_{theor} = V_t \left(\frac{RPM_{max}}{2}\right) Ev_{max} \tag{4}$$

where:

V_t	Total Engine Volume $\left[m^3\right]$
Ev_{max}	Maximum Volumetric Efficiency [$*$]
P_{MAP}	Manifold Absolute Pressure [KPa]
T_{MAP}	Manifold Intake Air Temperature [°C]
RPM	Engine Revolution per Minute $\left[\mathrm{min}^{-1}\right]$
RPM_{max}	Maximum Engine Revolution per Minute $\left[\mathrm{min}^{-1}\right]$
R	Universal Gas Constant=8.134 $\left[\frac{\mathrm{J}}{\mathrm{mol}\,.°\mathrm{C}}\right]$

C. ADMISSION SUBSYSTEM CONTROLLER

The admission subsystem (corresponding to the Throttle PI Controller in Fig. 2) controls the throttle valve opening angle, which in turn determines the air/fuel ratio. The controller input is the throttle reference position, and the output is a PWM signal to control the DC motor of the throttle valve. The actual throttle valve opening angle is measured by the TPS, usually integrated on the throttle valve body [9]. The throttle valve consists of a DC motor, a return spring, transmission gears, and a valve plate. The main parameters for the admission subsystem are: manifold absolute pressure, intake air temperature and air flow.

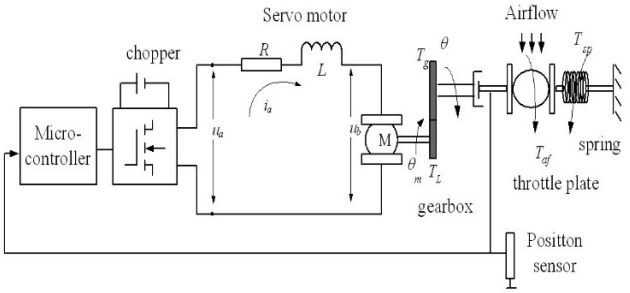

FIGURE 7. Physical model of the electronic throttle control.

Fig. 7 shows the physical model of the valve dynamics [10]. In the figure, *R, L, u a* and *ia*, are the armature resistance, inductance, voltage, and current, respectively; u_b is the counter electromotive constant with respect to engine speed; $\boldsymbol{\theta_m}$ is angular position of the motor shaft; $\boldsymbol{M}$ is output torque from the motor shaft; T_L is motor load torque; T_g is the torque transmitted from gears; T_{af} is the torque; T_{sp} is the air flow; $\boldsymbol{\theta}$ is angular position of the throttle valve plate; $T_{sp} = K_{sp}(\theta - \theta_0)$, where θ_0 is the initial angle of the return spring torque; and K_{sp} is the spring elastic coefficient.

Fig. 8 shows the plant model for the dynamics of the throttle valve corresponding to Fig. 7:

$$\left(\left((U(s) - k_b \Omega_t(s) N) * \left(\frac{1}{Ls + R}\right) * k_b\right) \times N - c_{m0} N - T_{SP} - c_{t0}\right) * \left(\frac{1}{\left(J_g + N^2 J_m\right) s + c_t + N^2 c_m}\right) = \Omega_t(s) \tag{5}$$

where K_b is the counter electromotive constant with respect to engine speed; C_m is the viscous friction coefficient of the motor shaft; C_{m0} is the static friction coefficient of the motor

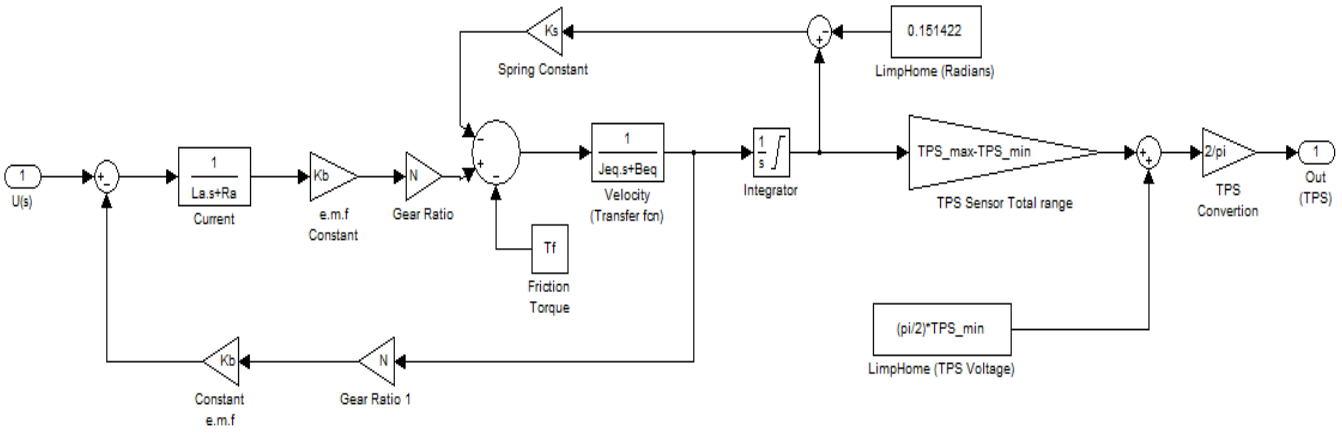

FIGURE 8. Simulink block diagram of electronic throttle control.

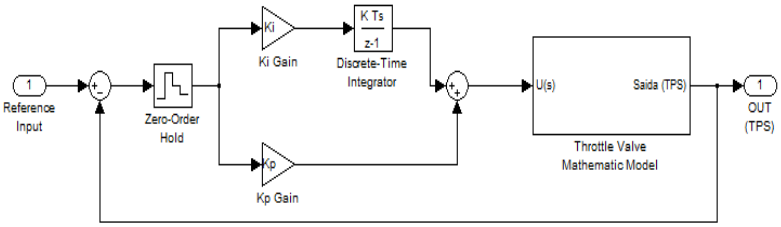

FIGURE 9. Block diagram of position controller.

shaft; J_m is the inertia of the motor; C_t is the viscous friction coefficient of the throttle valve; C_{t0} is the static friction coefficient and J_g is inertia; N is the internal throttle gear ratio. This model does not consider nonlinearities introduced by the variable position of the limp-home springs and the Coulomb friction. To calibrate and define all the throttle parameters, we performed several tests with an electronic throttle valve identical to the one in the EA133 engine.

Fig. 9 shows the throttle valve PI controller. Since the throttle valve does not need to close or open quickly, a PI controller can achieve good control performance. We use PI control instead of PID control, which may cause output fluctuations due to the derivative term. The proportional (K_p) and integral (K_i) gains of the PI controller are tuned with bench tests and the real engine. The proportional gain turns out to be relatively high, in order to compensate for the load imposed by the intake air suction in the idle speed condition.

D. SYSTEM IDENTIFICATION OF THE ENGINE

We present system identification of the engine dynamics, i.e., how the engine speed (RPM) varies depending on throttle valve angular position. We use the classic system identification techniques [11], and carry out experiments to obtain a linear model that characterizes the input/output relationship. Although there are some nonlinearities in SI engines, a linear model is often adequate for control design purposes [12].

The controller must start the engine and keep it in idle speed, controlling successfully all the three main subsystems. We needed to determine how the engine responds when the throttle valve angle changes, and build a look-up table relating throttle position, fuel injection and engine rotation, while keeping the following constant: fuel injection, dwell time, spark advance angle, and load on the engine.

The vehicle is mounted on an inertial dynamometer with a 2.5 ton constant load. We used the National Instruments PCI 6514 DAQ board, and the LabVIEW® software for data acquisition of engine input and output signals. The results are visualized with DIAdem®. The engine speed is set to 1000 RPM with the vehicle in constant speed; the fuel injection time is set to 10 ms; the advance ignition angle is set to 10 degrees. All of them are kept constant in the tests. We start with the throttle valve in its initial angular position (limp home), and apply a 20-degree step in the angular position of the throttle valve in the test. Fig. 10 shows the test results. The horizontal axis represents time (in seconds) and the vertical axis represents voltage (in volts). The green curve is the electronic trigger signal, which shows the exact instant in which the 20-degree step was applied to the throttle. The red channel represents the increase in the engine rotation speedin RPM. The Digital-to-Analog converter (DAC) of the microcontroller transforms digital values of 12 bits within the microcontroller program memory into an analog voltage available in one of its output pins. The resolution at this conversion rate is 60mV/100 rpm, i.e. the voltage output of the DAC output pin increases 60 mV for every 100 RPM. Channel 1 (rotation, red) shows the response of the rotation

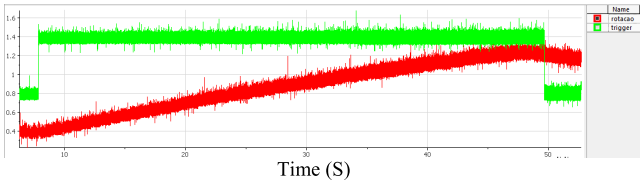

FIGURE 10. Engine speed response with step size of 20 degrees in the throttle valve angle. The green and red curves show respectively the step input reference signal, and the engine RPM response signal.

signal; Channel 2 (trigger, green) shows the trigger signal, indicating when the step input is applied.

It takes the engine approximately 40 seconds to stabilize around 5000 RPM. The results show that, within this operating range, the engine RPM behaves as a first-order system with dead-time behavior. The resulting model transfer function is:

$$G(s) = \frac{K}{\tau \cdot s + 1} \cdot e^{-\theta s} = \frac{1,55}{35{,}2.s + 1} \cdot e^{-0{,}08s} \qquad (6)$$

E. ENGINE SPEED CONTROLLER

Based on the engine plant model in Eq. 6, we design an engine speed controller (corresponding to the RPM PI controller in Fig. 2) that takes as input the engine RPM reference value, and outputs the throttle reference angular position, as input to the throttle PI controller described in Section C.

The specific engine used has a large dead time, of ∼200 ms to change its current state. Rapid changes of the throttle angular position may cause an effect known as ''drowning'', when the engine halts [13]. The choice of the time constant of the closed loop should be conservative to prevent control instability, especially at idle speed. We choose the time constant of 200 ms. Eqs. 7 and 8 show the proportional (K_p) and integral (K_i) gains of the PI controller:

$$K_p = \frac{1}{\mathrm{K}} \cdot \frac{\tau}{\theta + \tau_\mathrm{f}} = \frac{35,2}{1,55 * (0,08 + 0,2)} = 81,1 \qquad (7)$$

$$K_i = \frac{K_p}{\tau} = \frac{81,1}{35,2} = 2.3 \qquad (8)$$

We adopt PWM duty cycle of 86%, since bench tests show a small difference between the PWM duty cycle of 100% or 86% for controlling the throttle valve.

The PI controller is very effective in cruise, full load and deceleration (cut-off) modes. In idle speed, the system has satisfactory performance with a few oscillations between 800 and 900 RPM, although tests performed with the idle speed reference at 1000 RPM are more stable. This is due to mechanical losses of the engine, which have more impact at lower engine angular speed. When the engine has an abrupt acceleration, the RPM PI controller is inefficient without adjusting the percentage of fuel mass injected into the cylinder. Therefore, in the final implemented system, we added a block to correct the function of the engine state in the fuel injection subsystem. This block corrects by 50% the value of the final calculated injection time if the driver presses the accelerator pedal abruptly, with the maximum injection time set to be 18 ms.

IV. HARDWARE DESIGN

The ECM receives signals from engine sensors, determines the control actions and sends control instructions to actuators, in a closed loop control [14]. The ECM architecture (Fig. 11) consists of the following elements: (i) microcontroller, (ii) signal conditioning circuits, (iii) RAM memory, (iv) power line, and (v) CAN and Bluetooth modules for external communication.

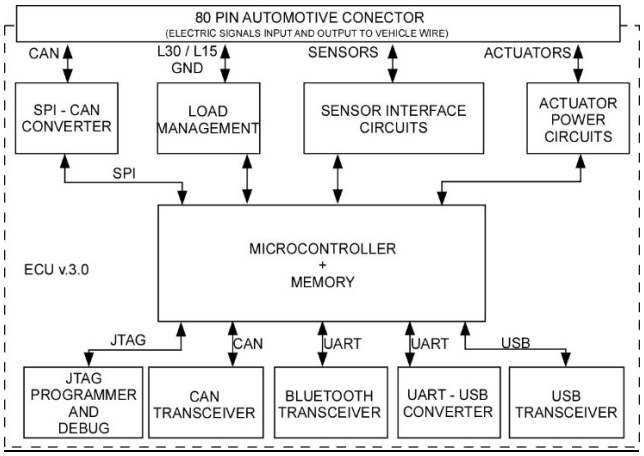

FIGURE 11. The ECM hardware block diagram.

We use the ARM Cortex M4 Kinetis K40 microcontroller (MK40X256ZLQ100), with a 32-bit CPU core with maximum speed of 100 MHz, and SP instructions delivering 1.25 Dhrystone MIPS. We use the NXP CodeWarrior™ Integrated Development Environment, and a J-Link® programmer with JTAG interface to write the firmware and configuration parameters into the microcontroller flash and RAM memories.

All sensor signals must be conditioned before they are sent to the ECM. While the ECM operates at 3.3 V, engine sensors operate at 5 or 12 V. Therefore, the following input signals must be conditioned: positive after key, engine RPM, Top Dead Cylinder (TDC), phase, Manifold Absolute Pressure (MAP), coolant temperature, intake air temperature, throttle position, accelerator pedal position, lambda, and knock sensors. The Kinetis microcontroller provides a maximum current of 20 mA in their output pins, while some actuators require up to a few Amps to activate. The following output signals must be conditioned: automotive relays (≅60mA), throttle valves (≅5A), fuel injection valves (≅2A), and ignition coils (≅5A).

We use the Proteus®software (Labcenter Electronics®) to develop the Printed Circuit Board (PCM). Fig. 12 illustrates the layout of the tracks and the positioning of the finished components, the 3D simulation of the upper/lower faces of the printed circuit layout, and the design of all four-layer tracks.

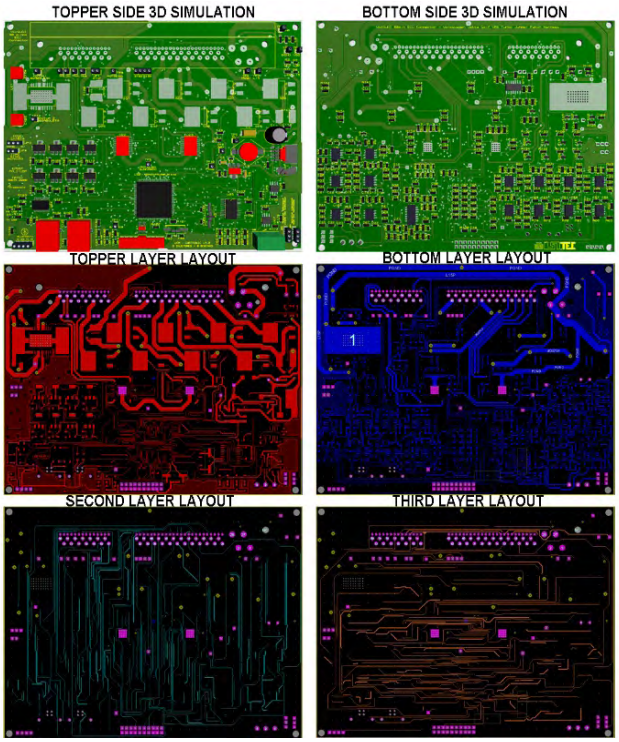

FIGURE 12. Layout of the four layers in the PCB.

The circuit consists of four layers. The top and bottom layers are designed for currents larger than 1A. The central layer is designed for digital and communication tracks. The circuit is built in a fiberglass plate with four printed faces, metalized holes, and hot air finish. Fig. 13 shows the printed board, including the electronic components, the microcontroller, the JTAG recording connector, the USB connectors, and the CAN network connector, and Bluetooth communication antenna.

V. SOFTWARE DESIGN

Fig. 14 shows the software architecture, consisting of three layers. The main layer is executed at initialization time, to determine the initial settings of all registers available to the microcontroller, including memory allocation for ignition maps and timing, and operating frequency of the microcontroller CPU (100 MHz). It also enables the Interrupt Service Routine (ISR) triggered by the positive after key signal, which initiates the synchronism and the Real Time Operating Systems (RTOS) layers. After initialization is completed, the synchronism and RTOS layers start to run (Fig. 15). The synchronism layer manages all ISRs, and the RTOS layer manages all application tasks with a time-triggered cyclic executive scheduler. ISRs in the synchronism layer have higher priorities, and can preempt application tasks in the RTOS layer.

Under typical operating conditions, the engine takes about 100 ms to change its current state. As a rule-of-thumb, a full cycle of the RTOS scheduler should not exceed 10% of the

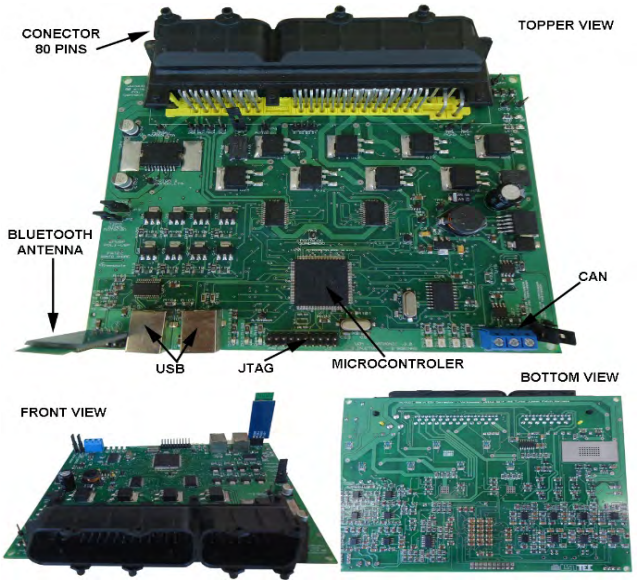

FIGURE 13. The ECM development board.

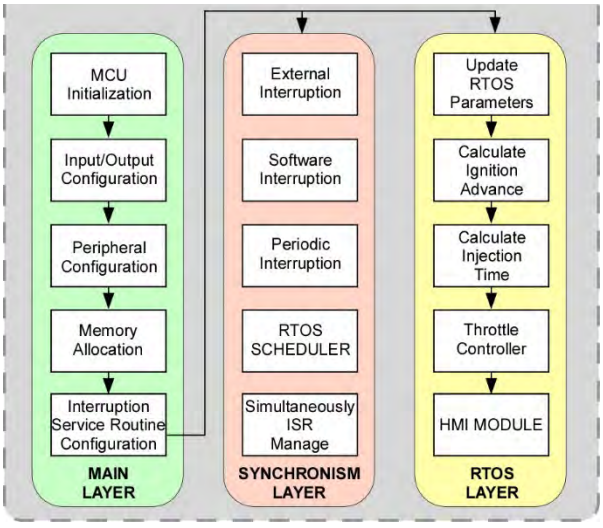

FIGURE 14. Block diagram of the ECM software, with three layers.

engine response time, so we set the scheduling cycle to be 10 ms, which is divided into 10 time slots of 1 ms each for 10 tasks. Each task is assigned a maximum time budget of 1 ms. We use a global data structure of RTOS_VEM vector (Engine State Vector) for data sharing among tasks, which can contain up to 61 engine state variables.

Fig. 16 presents the application Gantt chart, showing one execution scenario where the RTOS tasks may be preempted by ISRs in the synchronism layer. If any task exceeds its timing budget, a warning message is sent to the diagnostic system (human machine interface).

For verification and validation, we first perform verification of each subsystem, followed by validation of the whole system design to guarantee the desired performance and make comparisons. We present more discussions in detail next.

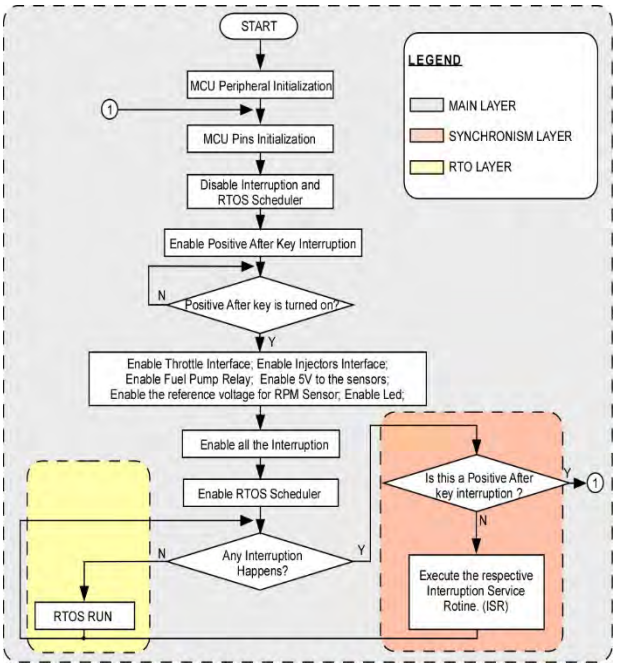

FIGURE 15. The middleware flowchart.

VI. VERIFICATION OF THE SUBSYSTEMS

We use the EA 113 family engine, which is available in a vehicle lab model Volkswagen Polo 1.6l and 2.0l [15]. The EA 113 is a typical SI engine, hence the ECM can be easily extended to other engine types.

The engine has transverse mounting and, at rotation of 2400 RPM, develops a maximum torque of 170 Nm. Its maximum power is 85 kW (116 hp) when the rotation reaches 5200 RPM. This engine synchronizes fuel injection and spark ignition, based on information of the variable reluctance sensor, placed in the crankshaft, and the phase sensor, placed in the camshaft. Fig. 17 shows the technical specification of this engine, including power and torque curves, cylinder capacity, compression ratio, and performance achieved with the original EMS [15].

A. VERIFICATION OF THE IGNITION SUBSYSTEM

The ignition subsystem aims to achieve effective combustion of the compressed mixture in the combustion chamber. The minimum dwell time to satisfactory combustion is 5.8 ms, determined from bench tests and technical datasheet of the original coil used in this engine. The dwell time varies according to the primary supply voltage provided by the battery/alternator of the vehicle, reaching 8 ms in extreme situations, such as during engine startup.

In addition to the coil dwell time, the ignition subsystem (corresponding to the Advance Ignition Controller in Fig. 2) controls the ignition advance angle to increase the pressure on the fuel mixture and, consequently, increase in engine power output. A map in the microcontroller flash memory stores the mapping from all possible engine speeds (in RPM) and Manifold Absolute Pressures (MAPs) (in KPa) to their corresponding ignition advance angles. The map is built by collecting experimental data with an inertial dynamometer, which allows setting the correct ignition timing and ignition advance angle at any engine condition. A load type dynamometer is the best way to accomplish this, since the engine can be held in a steady speed and MAP while the timing and angle are adjusted to achieve maximum output. In the experiment, the ignition advance angle is incremented step by step until the knocking phenomenon occurs. Table 1 shows the experimentally-measured data. Using Matlab®, the data in Table 1 is interpolated with two-dimensional interpolation to obtain the ignition advance angle map with 100 RPM resolution (Fig. 18). The graph serves as a lookup table from the engine rotation speed and the MAP to the corresponding ignition advance angle.

To calculate the engine rotation speed, the microcontroller measures the time interval between two consecutive teeth

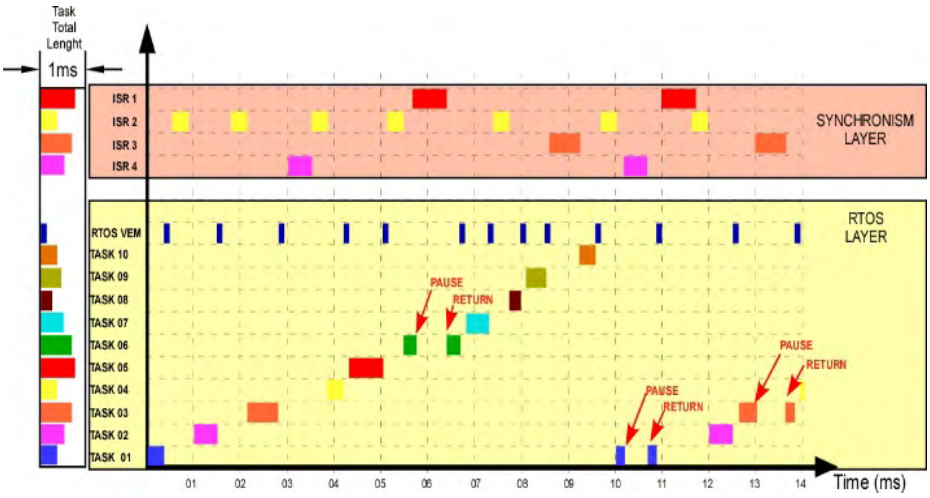

FIGURE 16. Gantt chart for synchronism and RTOS layer concurrent tasks.

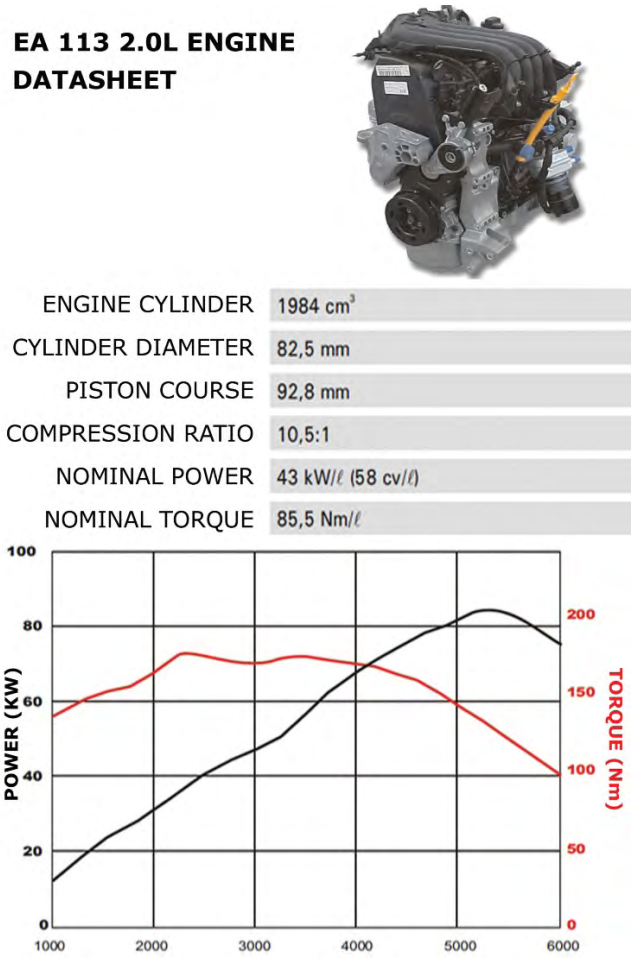

FIGURE 17. Technical specification of the EA 113 engine.

TABLE 1. Ignition advance angle map built using a dynamometer.

RPM \ MAP(KPa)	10	20	30	40	50	60	70	80	90	100	110	120
800	09	09	09	09	10	10	10	10	10	10	10	10
1000	09	09	10	10	12	12	12	12	12	12	11	12
1500	10	10	10	10	13	13	13	13	13	13	11	10
2000	15	15	15	15	15	15	15	15	15	15	14	13
2500	18	18	18	18	17	17	15	15	15	15	15	14
3000	22	22	22	21	20	19	16	16	16	16	16	16
3500	25	25	25	24	22	20	16	16	16	16	16	16
4000	29	29	29	27	24	21	18	18	18	18	18	18
4500	30	30	30	28	25	22	18	18	18	18	18	18
5000	30	30	30	29	27	24	19	19	19	19	19	19
5500	30	30	30	28	27	24	19	19	19	19	19	19
6000	30	30	30	29	27	24	19	19	19	19	19	19

from the RPM sensor. An integer value stored in the microcontroller timer register denotes the time interval between two consecutive teeth. Eq. 9 is used to obtain the engine rotation speed (measured in RPM/100 instead of RPM to reduce the integer range):

$$RPM_{/100} = \frac{10000}{TimeRegster * TimerPeriod_{(\mu s)}} \tag{9}$$

As an example, let's assume the engine rotation speed is 1000 RPM. Then the engine rotation period is 60 ms. Assume that one complete turn of the engine has 60 teeth, then the time interval between two consecutive teeth is 1 ms. If the microcontroller timer period is 1.28 μs, the timer register value should be 781 (1ms/1.28μs). Plugging in these values in Eq. 9, we obtain the correct result of RPM/100=10. An 8-bit timer has the maximum timer register value of 255, hence a 16-bit timer is needed to support the timer register value of 781.

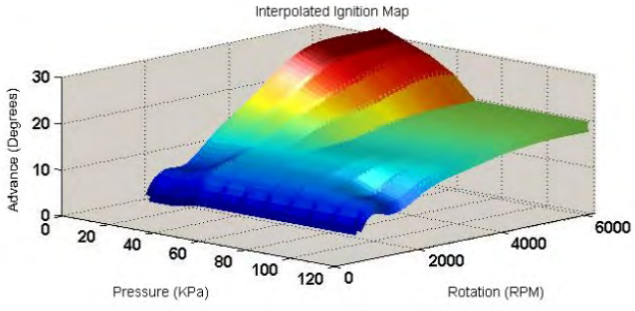

FIGURE 18. Ignition advance angle map, obtained by interpolation of data in Table I. The blue color indicates the engine operating region with lower power but more energy efficiency, while the red color indicates the operating region with higher power but less energy efficiency.

B. VERIFICATION OF THE INJECTION SUBSYSTEM

In order to implement the fuel injection subsystem, a physical model was used, described in Eqs. 1 and 2 [2]. The ECM sends a control pulse to control the injector opening time interval by opening and closing the injector valves, which determines the mass of fuel injected into the chambers.

The fuel injection task runs with period of 10 ms. The task has input parameters the manifold absolute pressure, the engine rotation speed, and the intake air temperature from the RTOS state vector "RTOS_VEM". It computes the base injection time with Eq. 1 and the volumetric efficiency with Eq. 2. The final injection time is corrected by multiplying the calculated injection time and the current volumetric efficiency in the range of [0, 1]. The fuel injection subsystem has time step resolution of 100 μs. The software stores all time values as integer multiples of the time step resolution in the RTOS state vector.

C. VERIFICATION OF THE ADMISSION SUBSYSTEM

In order to validate the mathematical model of the throttle valve, bench tests are carried out on the EA 133 electronic throttle pedal and throttle valve, and an input step signal was applied in the controller. Fig. 19 shows that the model provides a proper description of the admission subsystem.

Fig. 20 shows performance of the throttle valve PI controller. The reference signal of the accelerator pedal (green line) starts from the initial position, goes to the fully-activated position and then returns to the initial position. The throttle position sensor signal (red line) closely follows the accelerator pedal signal, indicating good control performance. The bottom graph shows the controller PWM output signal acting on both DC motors of the throttle valve. When the error between the pedal and the TPS is positive, the reference

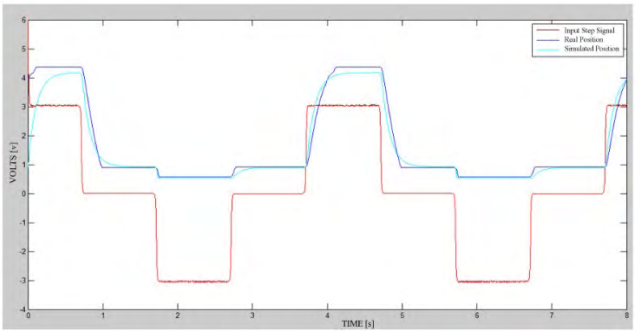

FIGURE 19. Results of the throttle valve model. The figure shows input signal (red line) with a 3.3V step, and real (dark blue line) and Simulink®simulated (light blue line) positions of the throttle valve.

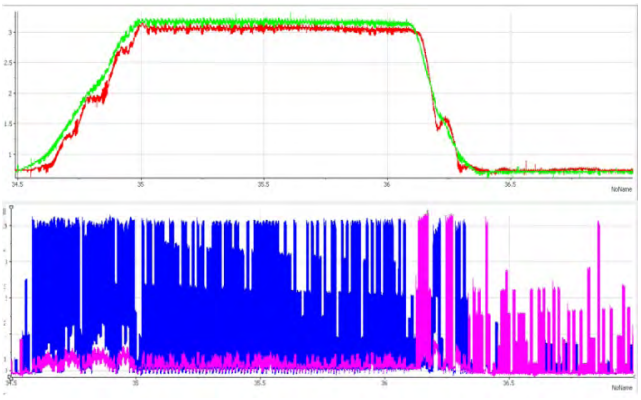

FIGURE 20. Final result of the admission subsystem controller.

(pedal) is larger than the current throttle valve position, the controller applies a PWM signal to the first throttle DC motor (blue color), causing the throttle valve to increase its angular position. When the error is negative, that is, the reference is smaller than the current position of the throttle valve, the controller applies a signal to the second throttle DC motor (pink color), causing the throttle valve to decrease its angular position.

The small steady-state error in Fig. 20, is due to disturbances such as friction and nonlinearities in the spring system of the throttle valve. The real dynamics of the throttle valve are compared with simulated dynamics, using the same input signal, in order to validate the model and its parameters.

VII. VALIDATION TESTS

The ECM is validated with bench tests followed by tests in an actual vehicle. A computer system equipped with a data acquisition board is used. The tests are performed with an inertial dynamometer (DynoTech 7201), in which the main cylinder has a combined mass of approximately 2500 kg. We obtained results in all relevant engine modes, including start, idle speed, acceleration, and deceleration. This section describes the results obtained in the final validation test.

A. VALIDATION OF VOLUMETRIC EFFICIENCY

We use software provided by the supplier of the dynamometer, called Dynercial 2, to calculate the mechanical loss between the vehicle wheel and the engine output shaft through the inertia of the scanning system. The first test aims to validate the synchronism layer and the influence of the volumetric efficiency in the engine output power curve. We calculate the volumetric efficiency of the engine and correct the injection time when engine speed exceeds 3000 RPM. The result of this first test is shown in Fig. 21, which shows a significant increase in engine power when the firmware starts to compute the volumetric efficiency and corrects the injection time as a function of engine filling dynamics. The area between dotted lines indicates the transition moment between the entire system without calculating the volumetric efficiency and after starting the correction of the injection time according to the volumetric efficiency. Without calculating the volumetric efficiency, a maximum 20 hp on the wheel is obtained. When the engine exceeded 3000 RPM and the firmware started to correct the injection time as a function of the current volumetric efficiency of the engine, a significant gain in engine power (red line) and torque (blue line) is obtained, reaching in 85.3 hp at 4950 RPM as maximum engine power output.

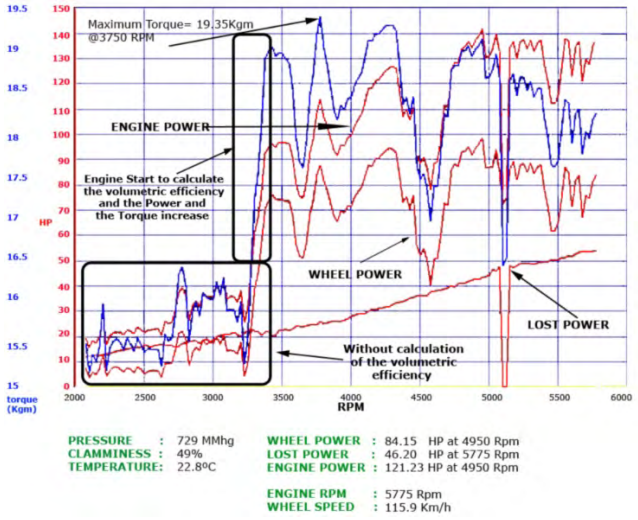

FIGURE 21. Engine volumetric efficiency validation test at the dynamometer.

B. FINAL TESTS

Now we perform validation of the entire system, including all three subsystems. We performed the first test without the RPM PI controller; only with the throttle PI controller. Fig. 21 shows that, with a sharp increase in engine power and torque attributed to the calculation of the volumetric efficiency, the engine shows oscillations in the measured power, with a reduction from its 80 hp at 4200 RPM to 37 hp at 4600 RPM. This oscillation occurs because the throttle valve angular position control is incorrect due to lack of the RPM PI controller. The input reference for the throttle valve controller is the driver's accelerator pedal. The dynamometer requires that the pedal to be fully pressed, so the test is performed with the throttle valve fully open. By comparing results obtained in

this test with the nominal power curve of the OEM electronic control unit of the engine [15], we conclude that the power curve measured in this test is far below the nominal capacity of the engine.

We subsequently add the RPM PI controller to correct the angular position of the throttle valve, and achieved significantly improved performance compared to the first test. Fig. 22 shows the result with the added RPM PI controller. The final test results in a maximum of 95.7 hp at 5475 RPM. A number of additional tests are performed by varying the proportional and integral gains of the RPM PI controller, in order to increase the maximum power on the wheel and get closer to the maximum nominal power of this engine (116 hp). The engine reached 108 hp, but the measured power exhibited instabilities and fluctuations. Therefore, we decided to be conservative, and reduced the controller gains to linearize the power curve. This resulted in reduced maximum engine power, but maintained the linear power curve with small oscillations, which is desirable feature.

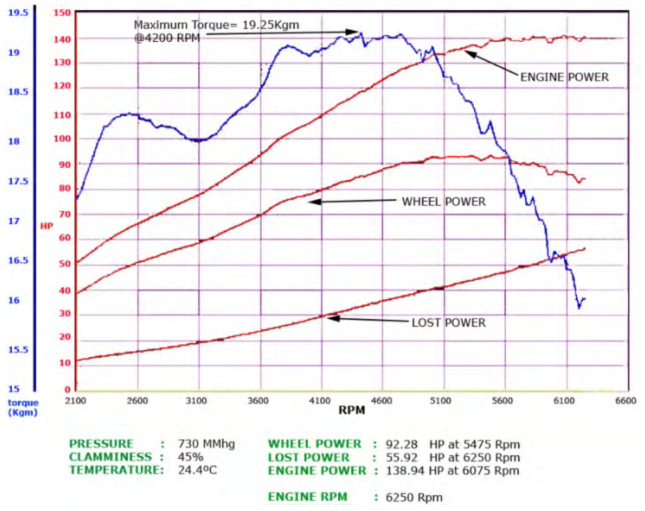

FIGURE 22. The final engine power and torque test result.

C. VALIDATION OF SYNCHRONISM LAYER

Fig. 23 presents the results of the final test on the dynamometer using DIAdem® software, showing the operation of the synchronism layer. The figure shows the signals of the four injection valves and the two coils in a complete Otto cycle, which involves two full revolutions of the crankshaft. To illustrate the synchronization with the engine, the phase and RPM sensor signals are also illustrated. The value of advance ignition angle of the first and second coils changes after the fourteenth teeth signal. The results of Fig. 23 are similar to the nominal synchronism timing, proving the effectiveness of the synchronism layer operation.

D. ANALYSIS

The loads and noises in a real engine are significantly higher than the values presented in theoretical models and influenced negatively the calibration and synchronism between the firmware and the engine. Some noises can be present

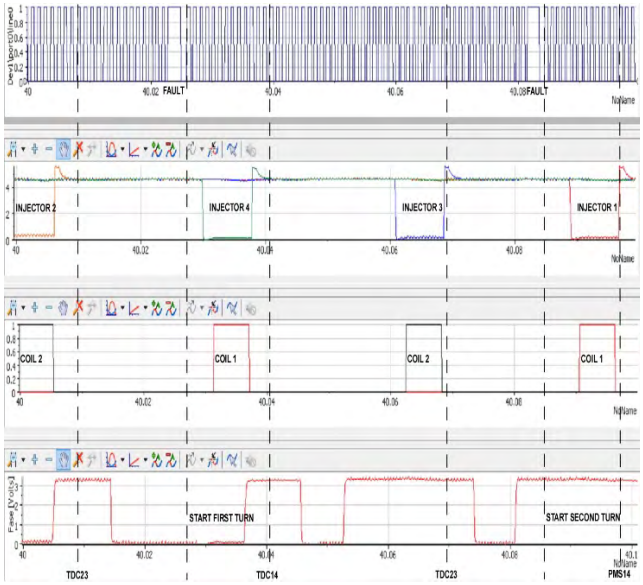

FIGURE 23. Validation of the synchronism layer.

in the signals from the sensors, which are sent to the ECM. The ECM must handle it appropriately, and this may require adjustments in the controllers by using a Kalman Filter to remove the noise.

The results show that our system controls appropriately the engine RPM speed around a reference value, while maintaining excellent results in the power curve analysis with a load applied to the main drive shaft. The RPM PI controller provides a considerable increase in the linearity of the engine power curve comparing to the first test result shown in Fig. 22 when the controller was still not implemented.

Our tests indicated that even small variations in engine control parameters may lead to instability and large variations in power output and fuel consumption. This shows that the ECM is crucial for proper operation of the SI engine, by adjusting parameters for changes on engine operating conditions.

VIII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a complete methodology for development of an ECM of an SI engine, in order to provide a useful reference for ECM developers. While this paper focuses on a single ECU, we plan to address in-vehicle distributed embedded systems connected by industry-standard bus protocols such as CAN-FD [18] or FlexRay [19]. We also hope to integrate the AUTOSAR standard [20], [21] into our development methodology.

REFERENCES

[1] R. K. Jurgen, *Automotive Electronics Handbook*, 2nd ed. New York, NY, USA: McGraw-Hill, 1999.
[2] B. B. Gmbh, *Bosch Automotive Electrics and Automotive Electronics: Systems and Components, Networking and Hybrid Drive*, 5th ed. Berlin, Germany: Springer-Verlag, Nov. 2013, p. 521
[3] B. Ashok, S. D. Ashok, and C. R. Kumar, "A review on control system architecture of a SI engine management system," *Annu. Rev. Control*, vol. 41, pp. 94–118, 2016.

[4] L. Glielmo, F. Vasca, and C. Rossi, "Architecture for electronic control unit tasks in automotive engine control," in *Proc. IEEE Int. Symp. Comput.-Aided Control Syst. Design (CACSD)*, Sep. 2000, pp. 42–47.
[5] J. Cook, J. Sun, J. H. Buckland, I. V. Kolmanovsky, H. Peng, and J. W. Grizzle, "Automotive powertrain control—A survey," *Asian J. Control*, vol. 8, no. 3, pp. 237–261, Sep. 2006.
[6] W. Ribbens, *Understanding Automotive Electronics: An Engineering Perspective*, 8th ed. Oxford, U.K.: Butterworth Heinemann, 2017.
[7] R. van Basshuysen and F. Schaefer, *Internal Combustion Engine Handbook*, 2nd ed. Pittsburgh, PA, USA: SAE International, 2016.
[8] D. Kjellqvist, "Concepts, strategies and controller for gasoline engine management," M.S. thesis, Luleå Univ. Technol., Luleå, Sweden, 2005, p. 80.
[9] Robert Bosch Gbmh, *Automotive Handbook*, 9th ed. Pittsburgh, PA, USA: SAE International, 2014, p. 1550. [Online]. Available: https://www.sae.org/publications/books/content/bosch9/
[10] R. Chen, L. Mi, and W. Tan, "Adaptive fuzzy logic based sliding mode control of electronic throttle," *J. Comput. Inf. Syst.*, vol. 8, pp. 3253–3260, Apr. 2012.
[11] K. Ogata, *Modern Control Engineering*, 5th ed. London, U.K.: Pearson, 2012, p. 930.
[12] C. Wang and F. Luo, "Software development of automotive engine electronic control unit," in *Proc. Int. Conf. Power Energy Syst. (ICPE)*, vol. 13, 2012, pp. 147–152.
[13] J. A. Cook and B. K. Powell, "Modeling of an internal combustion engine for control analysis," *IEEE Control Syst. Mag.*, vol. 8, no. 4, pp. 20–26, Aug. 1988.
[14] P. Terreni and R. Gentili, "Closed-loop electronic fuel injection for spark-ignited engines," *IEEE Trans. Veh. Technol.*, vol. VT-35, no. 1, pp. 30–38, Feb. 1986.
[15] V. Brazil, "Engines 1.6l and 2.0l Polo," Volkswagen Acad., São Bernardo do Campo, Brazil, Tech. Rep., 2007, p. 47.
[16] R. Isermann, *Engine Modeling and Control: Modeling and Electronic Management of Internal Combustion Engines*. Berlin, Germany: Springer-Verlag, 2014, p. 637.
[17] L. Guzzella and C. H. Onder, *Introduction to Modeling and Control of Internal Combustion Engine Systems*. Berlin, Germany: Springer-Verlag, 2010, p. 362.
[18] R. De Andrade, K. N. Hodel, J. F. Justo, A. A. M. Lagana, M. M. Santos, and Z. Gu, "Analytical and experimental performance evaluations of CAN-FD bus," *IEEE Access*, vol. 6, pp. 21287–21295, 2018.
[19] Z. Gu, G. Han, H. Zeng, and Q. Zhao, "Security-aware mapping and scheduling with hardware co-processors for FlexRay-based distributed embedded systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 10, pp. 3044–3057, Oct. 2016.
[20] Q. Zhao, Z. Gu, and H. Zeng, "Design optimization for AUTOSAR models with preemption thresholds and mixed-criticality scheduling," *J. Syst. Archit.*, vol. 72, pp. 61–68, Jan. 2017.
[21] Q. Zhao, Z. Gu, H. Zeng, and N. Zheng, "Schedulability analysis and stack size minimization with preemption thresholds and mixed-criticality scheduling," *J. Syst. Archit.*, vol. 83, pp. 57–74, Feb. 2018.

BRUNO MARTIN DE ALCÂNTARA DIAS graduated in automotive electronics from Fatec Santo André in 2011 and received the master's degree in electrical engineering from the Escola Politécnica, Universidade de São Paulo (POLI-USP), in 2015, and the Executive M.B.A. degree in project management from Fundação Getúlio Vargas in 2016. He is currently pursuing the Ph.D. degree in electrical engineering with POLI-USP. He is also the Project Manager of R&D Projects at the HION TECH Company and a Researcher at USP.

ARMANDO ANTONIO MARIA LAGANÁ received the degree in electrical engineering from the School of Engineering Maua in 1975 and the Ph.D. degree in electrical engineering from the Escola Politécnica, Universidade de São Paulo, in 1994. He is currently a Professor with the Department of Electronic Systems Engineering, Escola Politécnica, Universidade de São Paulo. He is involved in the automotive electronics field with an emphasis on engine management and his previous experience is in the field of materials and processes of microelectronics.

JOÃO FRANCISCO JUSTO received the B.Sc. and M.Sc. degrees in physics from the University of São Paulo, in 1988 and 1991, respectively, and the Ph.D. degree in nuclear engineering from the Massachusetts Institute of Technology in 1997. He is currently a Professor of electrical engineering with the Escola Politécnica, Universidade de São Paulo. He has experience in the computational modeling of electronic material and embedded electronics.

LEOPOLDO RIDEKI YOSHIOKA received the bachelor's degree in electronics engineering from the Instituto Tecnologico de Aeronautica, Sao Jose dos Campos, Brazil, in 1984, and the Ph.D. degree from the Tokyo Institute of Technology, Japan, in 1991. He was an Assistant Professor at the Tokyo Institute of Technology from 1991 to 1992. He is currently an Assistant Professor with the Department of Electronic Systems Engineering, Escola Politécnica, Universidade de São Paulo. His research areas are intelligent mobility and automotive systems.

MAX MAURO DIAS SANTOS (M'15–SM'17) received the bachelor's degree in electrical engineering from the Instituto Católico de Minas Gerais, Brazil, in 1993, and the M.E. and Ph.D. degrees in electrical engineering from the Universidade Federal de Santa Catarina, Brazil, in 1996 and 2004, respectively. He was a Post-Doctoral Fellow in electrical engineering with the Universidade de Aveiro, Portugal, from 2005 to 2006. He is currently an Assistant Professor with the Electronic Department, Universidade Tecnológica Federal do Paraná, Ponta Grossa, Brazil. His main research areas are embedded systems, automotive systems, and industrial automation.

ZONGHUA GU (SM'–) received the Ph.D. degree from the Department of Electrical Engineering and Computer Science, University of Michigan at Ann Arbor, in 2004. He is currently a Professor with the Department of Applied Physics and Electronics, Umeå University, Sweden. His research area is embedded and cyber-physical Systems.