Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004

Managing Complexity in Large-Scale Control System Design

Anthony Phillips[1] Diana Yanakiev Fangjun Jiang
Research and Advanced Engineering
Ford Motor Company

***Abstract*—**Continued advances in onboard computing technology are allowing automotive companies to accelerate the level of functionality delivered on new vehicles through electronics and software. New features such as hybrid powertrains and vehicle stability control are driving the need for vehicle-level supervisory control. In this work, the supervisory controller was defined in a modular and hierarchical structure in order to achieve better reusability and flexibility in incorporating new technologies into existing applications. In addition, a database tool was developed for the purpose of managing the complexity associated with maintaining the modular and hierarchical structure. Finally, the application of this work to advanced development of new hybrid vehicle prototypes is described.

I. Introduction

Automotive controls have historically been limited to the powertrain (engine and transmission) and brake systems within the vehicle with only minimal communication between them. As customer and regulatory demands in the areas of performance, fuel economy, and emissions continue to grow, the need for more intelligence and coordination within the vehicle increases. Automakers are responding by leveraging cost and speed improvements in microprocessors and communication networks that enable the introduction of more advanced subsystems (e.g. advanced vehicle stability control systems and hybrid vehicle powertrains). Managing the control system development and coordination of these new systems is rapidly becoming more complex. At the same time, the faster development times being dictated by today's economic environment require more reusability and portability of controller software from one project to the next.

In the past, new features that were added to the vehicle were routinely contained within a single subsystem. For instance, cruise (vehicle speed) control can be fully contained in the engine control subsystem. Similarly, anti-lock brake control can be completely contained within the brake control subsystem. Little or no interaction with other subsystems was required for implementation of these new features. More recently, companies are beginning to add more sophisticated features (e.g. vehicle dynamic stability control or hybrid electric powertrains) to meet customer demands and remain compliant with government regulations. These new features result in much more complex and highly interacting control systems within the vehicle. A vehicle-level supervisory control strategy is required to manage these interactions and ensure that the total control system meets the overall requirements for the vehicle. In order to ensure the most reusability and flexibility of this supervisory control strategy, it needs to be modular and hierarchical. Managing the complexity of the design of this control strategy is a challenging task.

The paper will focus on describing the control system management process and how it has been applied to several recent advanced hybrid vehicle projects.

II. Vehicle System Control

In today's vehicles, most major subsystems (engine, transmission, brakes, etc.) have a corresponding embedded microprocessor that controls the subsystem function by managing a complex set of sensors and actuators. Vehicles normally also have an even greater number of less sophisticated processors installed as part of the overall electronic system that is responsible for operating the "creature comfort" features such as door locks, seats, power windows, etc. In total, a vehicle may have tens of microprocessors connected through networks and hardwire interfaces [1].

In the earliest implementations of electronic controls in automobiles, the major subsystems operated autonomous to one another. In this case the driver interacted directly with the subsystem (e.g. through the accelerator pedal/throttle cable linkage to the engine). The subsystem controller determined the driver's intent through a sensor and used this information, along with other measured and inferred vehicle and subsystem information, to control the subsystem's actuators to achieve the desired response. With increasing computing power, new vehicle-level features were developed and employed that relied on coordinated

[1]Corresponding author: aphilli8@ford.com, 313-594-4717, P.O. Box 2053, MD 1170, SRL, Dearborn, MI 48167

interaction between the major subsystems (e.g. traction control). This interaction has traditionally been managed through peer-to-peer coordinated control designs. Supported by faster and more robust communication protocols, today's vehicles employ an even great number of features and technologies that require interaction between major subsystems. Figure 1 shows an example of some high-level vehicle functions and the subsystems that they impact.

Clearly, complexity management within new vehicle applications is increasing at a rapid rate. Managing subsystem interactions through peer-to-peer coordination is becoming increasingly more difficult. As a result, new complex vehicle applications are often based instead on a supervisory or hierarchical control structure [2]-[5]. In this type of structure (see Figure 2), a vehicle system control (VSC) strategy is used to manage all of the interactions between the subsystem controllers. Since the driver dictates high-level vehicle commands (accelerate, decelerate, turn, etc.), the VSC also handles the interface to the driver. The measured information from the driver is used in the highest levels of the hierarchy to determine overall vehicle requests that are later cascaded to the subsystem controllers.

Since, by its nature, the VSC incorporates the high level functions that help give the vehicle its character, manufacturers typically want to develop these functions internally. On the other hand, some of the major subsystem controls are seen as commodities and are supplied by the subsystem vendors. By standardizing the interfaces between the VSC functions and the major subsystem functions, there are added advantages of flexibility in sourcing of subsystem vendors and reusability of functions.

III. FUNCTIONAL VERSUS PHYSICAL ARCHITECTURE

The vehicle system control strategy described above and shown in Figure 2 represents a functional control architecture. This is distinctly different from a physical (electrical) architecture (example shown in Figure 3). Whereas the functional architecture describes the relationship of the various control elements or tasks, the physical architecture specifies the interconnections of the electrical components of the control system. In traditional implementations of automotive controls, the functional architecture has coincided with the physical architecture. In other words, all functions associated with a particular subsystem have resided in the subsystem's corresponding hardware control module.

As more and more vehicle-level functions are developed and implemented on automobiles, there is a temptation to move to a physical architecture that mimics the functional architecture described in Figure 2. There are numerous reasons why a separate hardware module for the VSC is not desirable, however. Cost, failure mode management, communication network bandwidth, robustness, and loading, and even supplier relationships can all weigh against a hierarchical physical architecture.

Clearly if a VSC-centric functional architecture is to be implemented in a vehicle without a central VSC hardware module, the vehicle-level functions need to be distributed amongst the existing hardware modules. To accomplish this the internal VSC structure must be well defined and modular to the extent that it can be split across the hardware modules. In order to accommodate the ever-increasing level of functionality within vehicles, the internal structure must also be flexible enough to support reallocations for new applications and technologies.

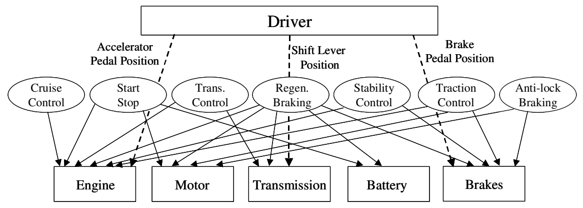


Figure 1. Interactions between vehicle-level functions and major subsystems.

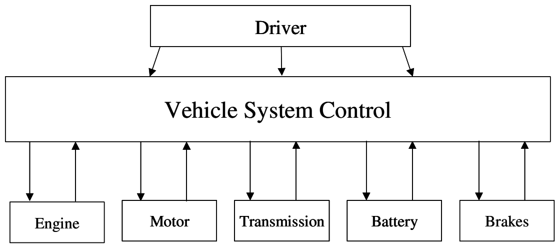


Figure 2. Subsystem coordination through vehicle system control hierarchy.

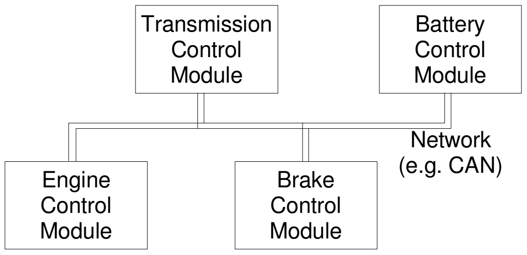


Figure 3. Typical physical (electrical) architecture in an automotive control system.

IV. ATOMIC FUNCTION CONCEPT

In automotive applications, control functionality is implemented primarily through software. Typically the software is managed in relatively large elements called "features". Each feature is a set of related functionality that is of a size that is possible to be maintained by a single engineer. For example spark control, fuel control, and idle speed control would all be examples of engine control features. Since each new vehicle application tends to be a little different from the last, these features have multiple versions that need to be closely tracked. New versions are normally evolved from older versions in order to retain the majority of the feature content that is unchanged from the previous applications.

For each vehicle application, the appropriate versions of all relevant features are combined to form a single control “strategy”. See Figure 4. Since the overall strategy evolves throughout the vehicle development process, new versions of each strategy need to be tracked as well. For any given vehicle application, the control strategy consists of over one hundred features.

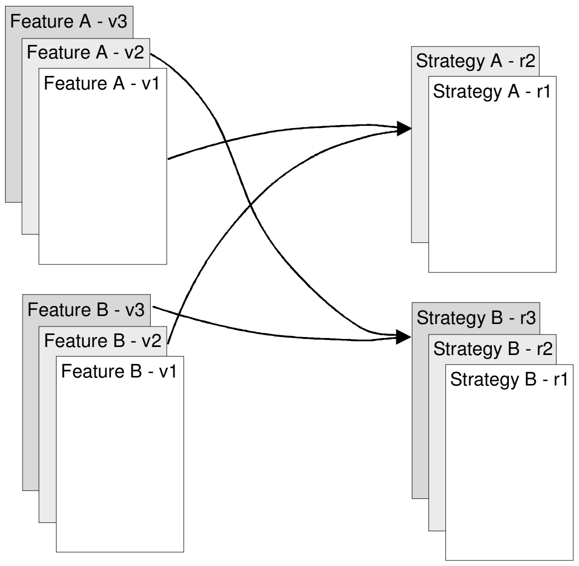


Figure 4. Feature and strategy elements in a typical control system management process.

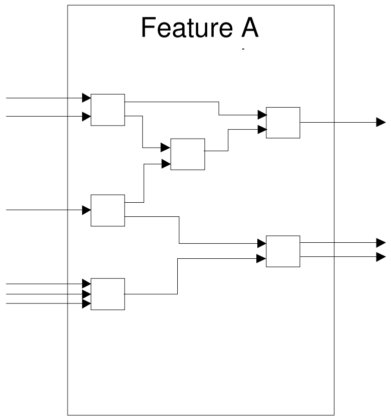


Figure 5. Several atomic functions shown as elements of a single feature.

In order to achieve the modularity and flexibility required to institute a VSC-centric control structure, the control system needs to be managed at a finer granularity than either the strategy or feature level. With this challenge in mind, we have been using a new process for developing advanced vehicle control solutions. Specifically, the vehicle control system is broken down into "atomic" functions that are portable and reusable and that have standard, generic interfaces. Typically, a feature, as described above, might consist of several atomic functions. See Figure 5. Each atomic function consists of a single algorithm designed to achieve a single function. The function can contain any type of open-loop, closed-loop, or logical control algorithm. The expectation is that an atomic function would never be split apart in order to allocate it to more than one hardware module. One example of an atomic function is *accelerator pedal interpreter*. See Figure 6. This function has inputs of accelerator pedal position and vehicle speed and an output of desired wheel torque. Its only function is to convert driver demand (as measured by pedal position) into a desired wheel torque.

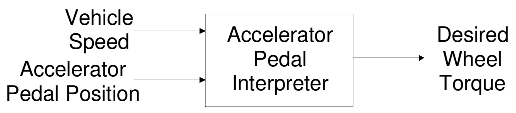


Figure 6. Accelerator pedal interpreter as an atomic function.

Equally important to the modularity provided by the atomic functions is the ability to now organize the internal structure of the VSC hierarchically. See Figure 7. Here the atomic functions are represented in the hierarchy as F1, F2, etc. Elements H1, H2, etc., referred to as subsystems, represent the structure of allowed interactions between the functions in the hierarchy. Algorithms exist only at the atomic function level. Accordingly, in an implementation, only the atomic functions are allocated to hardware modules. The subsystems become virtual at the allocation phase. They only exist to define the structure of the functional interactions.

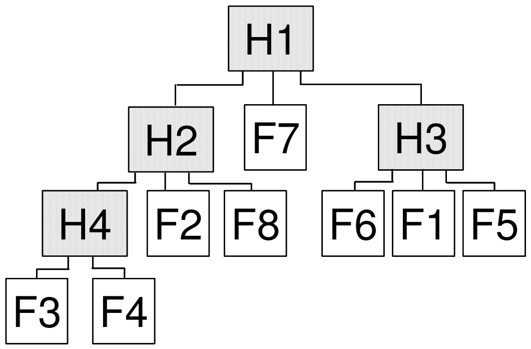


Figure 7. Generic hierarchical structure within the VSC.

There are various advantages to the atomic function approach. First, this approach isolates the majority of the control strategy to new function changes. Whereas in the feature-based approach, versions of entire features need to be tracked for small changes to functions, the modularity and hierarchical nature of this approach allows individual function changes to be tracked separately with no updates needed for the remaining functions. This results in a greater level of reusability and a lesser need to track large versions of control strategy with only minor changes. Second, responsibility for redesign of the atomic functions can easily be distributed across engineers during the design phase of a project and the functions can be readily lumped into control code "releases" for hardware-in-the-loop or in-vehicle development. Next, by having standard interfaces for each function, early negotiations with suppliers on control architecture definition can be minimized (whether or not the supplier or the OEM develops any given function). Finally, the functions are of a size that are portable. That is, each function is small enough so that for any given vehicle application, there should never be a need to split the function across hardware modules for the implementation. Also, because of their portable nature, each function's hardware allocation can be reconsidered for new vehicle or technology applications.

V. DATABASE TOOL

Clearly, one aspect of this new process is that there are many more functions than there were features for a given vehicle application. Consider the set of functions F1-F7 shown in Figure 8. This set may represent all of the VSC functions used in a given vehicle application. For the next application, new vehicle requirements may drive the need to create additional functions F8-F11 but the requirements may also obviate the need for a portion of the original functions. Tracking the definition, interface definition, attributes and usage of these functions is a considerable challenge.

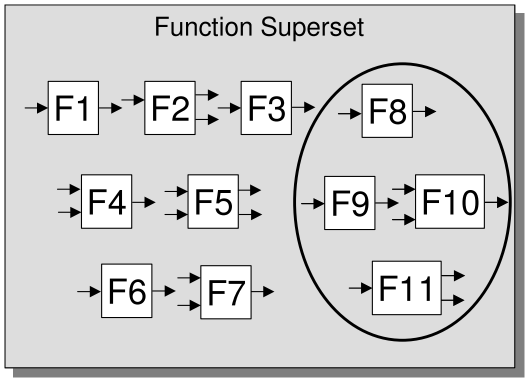
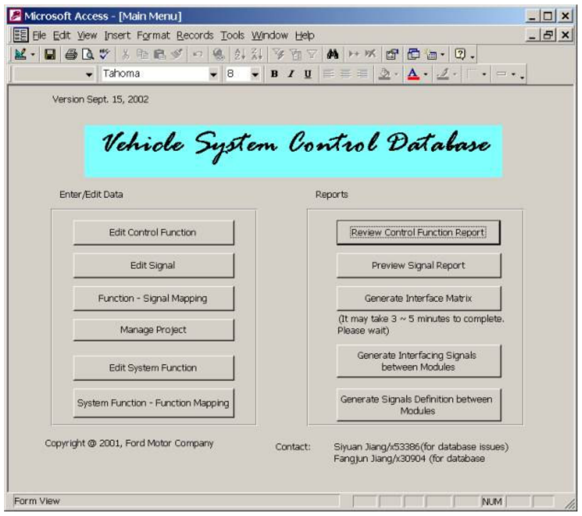


Figure 8. Function superset for existing and new applications.

To facilitate the functional management process, a prototype database tool has been developed for tracking the functions, signals, code releases, and hardware allocations related to each new project. Figure 9 shows the user interface for the database tool.

In its prototype instantiation as shown here, the tool allows users to enter or edit information about both functions and signals independently. As every new function is added the user is prompted to specify the input signals used by that particular function. Similarly, with each new signal, a unique originating function must be designated. The tool ensures proper signal connectivity and usage even as new functions are added to large, existing designs.

In addition to allowing the user to track relationships between signals and functions within the entire superset, the tool also allows the user to enter information about system functions (vehicle-level features). Some examples of these might be vehicle stability control or engine start/stop. Using the "System Function – Function Mapping" feature of the tool, the user can then associate specific atomic functions with each vehicle feature. This aspect of the tool provides a good way to track atomic functions back to vehicle-level requirements.


Microsoft Access - [Main Menu]
File Edit View Insert Format Records Tools Window Help
Tahoma
8
Enter/Edit Data
Edit Control Function
Edit Signal
Function - Signal Mapping
Manage Project
Edit System Function
System Function - Function Mapping
Reports
Review Control Function Report
Preview Signal Report
Generate Interface Matrix
Generate Interfacing Signals between Modules
Generate Signals Definition between Modules
Form View
NUM

Version Sept. 15, 2002

Vehicle System Control Database

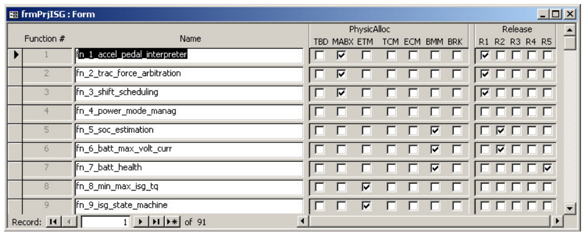
(It may take 3 ~ 5 minutes to complete. Please wait)

Copyright @ 2001, Ford Motor Company

Contact: Siyuan Jiang/x53386(for database issues)
Fangjun Jiang/x30904 (for database

Figure 9. Primary user interface for VSC database tool.

Finally, the tool also allows the user to manage function usage by project (vehicle application). Figure 10 shows the dialogue box for entering project specific information about each function. Specifically, each function can be associated with a specific hardware allocation for that particular project. In the example shown, function number 7 (fn_7_batt_health) is allocated to the battery management module (BMM) while function 8 (fn_8_min_max_isg_tq) is allocated to the electric traction motor module (ETM).



frmPrjISG : Form

		PhysicAlloc							Release				
Function #	Name	TBD	MABX	ETM	TCM	ECM	BMM	BRK	R1	R2	R3	R4	R5
1	fn_1_accel_pedal_interpreter	☐	☑	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
2	fn_2_trac_force_arbitration	☐	☑	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
3	fn_3_shift_scheduling	☐	☑	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
4	fn_4_power_mode_manag	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
5	fn_5_soc_estimation	☐	☐	☐	☐	☐	☑	☐	☐	☑	☐	☐	☐
6	fn_6_batt_max_volt_curr	☐	☐	☐	☐	☐	☑	☐	☐	☑	☐	☐	☐
7	fn_7_batt_health	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐	☐	☑
8	fn_8_min_max_isg_tq	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
9	fn_9_isg_state_machine	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐

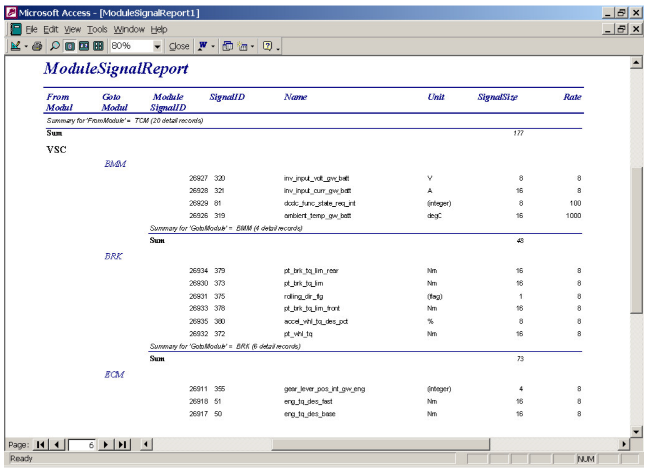
Record: 1 of 91

Figure 10. Project management dialogue box within database management tool.

Another aspect that can be managed at the project level is the linkage of each function to a specific code release during vehicle development. This feature facilitates staging the functions for more disciplined development.

Once the relationships between functions, signals, and hardware allocations are established in the database, it is a simple matter to generate the hardware interface specification for any given project. The "Generate Interfacing Signals between Modules" feature of the tool automatically establishes this module-to-module specification in a form that readily supports definition of messages under a network protocol. Figure 11 shows an example output for this feature. In this application, a dedicated VSC hardware module was used. Specifically shown are the signals required for the interface from the VSC to the battery management module (BMM), brake control module (BCM) and the engine control module (ECM, partial).

By managing the functions and their attributes and interface specifications through this database tool, it becomes much easier to assess the impact of function changes on the rest of the system, to reuse the functions in subsequent vehicle applications, and to track physical allocation of the functions across the vehicle control hardware.



Microsoft Access - [ModuleSignalReport1]
File Edit View Tools Window Help
80%
Close
Page: 6
Ready
NUM

ModuleSignalReport

From Modul	Goto Modul	Module SignalID	SignalID	Name	Unit	SignalSize	Rate
Summary for 'FromModule' = TCM (20 detail records)							
Sum						177	
VSC							
	BMM						
		26927	320	inv_input_volt_gw_batt	V	8	8
		26928	321	inv_input_curr_gw_batt	A	16	8
		26929	81	dcdc_func_state_req_int	(integer)	8	100
		26926	319	ambient_temp_gw_batt	degC	16	1000
		Summary for 'GotoModule' = BMM (4 detail records)					
		Sum				48	
	BRK						
		26934	379	pt_brk_tq_lim_rear	Nm	16	8
		26930	373	pt_brk_tq_lim	Nm	16	8
		26931	375	rolling_dir_flg	(flag)	1	8
		26933	378	pt_brk_tq_lim_front	Nm	16	8
		26935	380	accel_whl_tq_des_pct	%	8	8
		26932	372	pt_whl_tq	Nm	16	8
		Summary for 'GotoModule' = BRK (6 detail records)					
		Sum				73	
	ECM						
		26911	355	gear_lever_pos_int_gw_eng	(integer)	4	8
		26918	51	eng_tq_des_fast	Nm	16	8
		26917	50	eng_tq_des_base	Nm	16	8

Figure 11. Module-to-module interface specification report from database tool.

VI. Application Examples

The processes and tools described in this report have been applied across several hybrid vehicle development projects over the past few years. Development of the process and tools was motivated by the desire to create a common control strategy to support two distinct vehicle applications of an integrated starter-generator (ISG) system. A diagram of the ISG system for one of the applications is shown in Figure 12. The system consisted of an electric motor sandwiched between the engine and transmission in a rear wheel drive vehicle. The vehicle also had electro-hydraulic brakes (EHB - not shown) that allowed series regenerative braking.

The process and tools were further motivated as the development team moved from the ISG project on to an electric four wheel drive (E4WD) application (see Figure 13). This vehicle consisted of a conventional rear wheel drive powertrain coupled with an electric motor on the front axle. This vehicle also had EHB to facilitate series regenerative braking. Although the vehicle architectures were radically different, many of the VSC functions could be carried over to the new application. For instance both applications required functions such as *battery state of charge estimation*, *charging power determination*, and *torque coordination at the wheels*. Clearly it was desirable to reuse as much of the ISG control strategy as possible.

Figure 14 shows an immediate benefit of the modular structure of the VSC functions. Figure 14a shows the major atomic function elements of the torque control feature for a conventional (non-hybrid) vehicle application with electronic throttle control. These elements exist as part of an entire feature in the traditional engine control strategy. The driver demanded accelerator torque enters on the left where it is arbitrated with

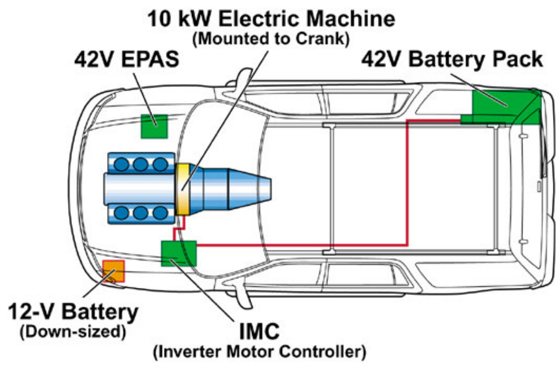


Figure 12. Diagram of the Integrated Starter-Generator vehicle.

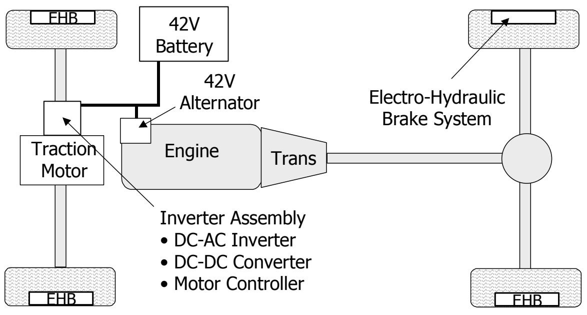
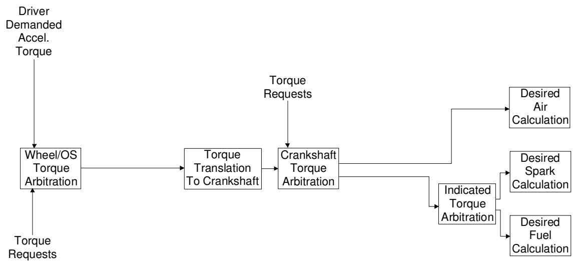
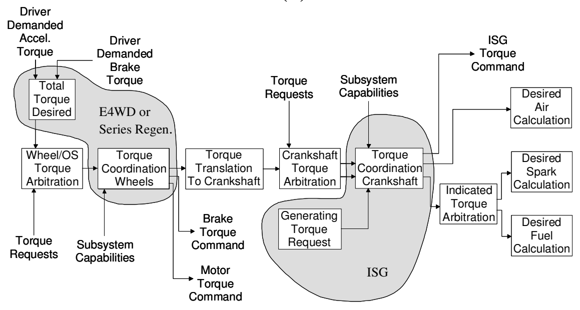


Figure 13. Diagram of the Electric Four Wheel Drive vehicle.



Driver Demanded Accel. Torque
Torque Requests
Wheel/OS Torque Arbitration
Torque Translation To Crankshaft
Crankshaft Torque Arbitration
Desired Air Calculation
Desired Spark Calculation
Indicated Torque Arbitration
Desired Fuel Calculation
Torque Requests
(a)



Driver Demanded Accel. Torque
Driver Demanded Brake Torque
Total Torque Desired
E4WD or Series Regen.
Wheel/OS Torque Arbitration
Torque Coordination Wheels
Torque Translation To Crankshaft
Torque Requests
Crankshaft Torque Arbitration
Subsystem Capabilities
Torque Coordination Crankshaft
ISG Torque Command
Desired Air Calculation
Desired Spark Calculation
Indicated Torque Arbitration
Desired Fuel Calculation
Generating Torque Request
ISG
Brake Torque Command
Motor Torque Command
Torque Requests
Subsystem Capabilities
(b)

Figure 14. a) Atomic functions in torque control feature for conventional vehicle, b) Atomic functions for torque control in various hybrid vehicle configurations.

other requests for torque at the wheels (from, e.g., traction control). From there the torque request is translated to the engine crankshaft domain where it is again arbitrated with other requests (from, e.g., the transmission). Eventually the torque request is split into a desired air calculation and desired spark and fuel calculations that are then sent to the various actuators. Figure 14b shows the same functional elements with the addition of those required to control a vehicle with an ISG and/or E4WD and series regenerative braking systems. With only the addition of a *torque coordination crankshaft* function, the same control architecture could be used for the ISG vehicle as was used for the conventional vehicle. Similarly, by adding a *torque coordination wheels* function, either/both E4WD or/and series regenerative braking could be realized with disrupting the existing functions. The modular nature of the functions also aids in their allocation to hardware modules within the vehicle. As mentioned above, there are many reasons why an independent VSC hardware module is not desirable in a vehicle application. For these reasons, the VSC is distributed across the various modules. Figure 15 shows a conceptual drawing of how the VSC was distributed in the ISG project. In this case, portions of the VSC were allocated to the engine, transmission, and motor control modules. By having well-defined interfaces for the atomic functions and a modular structure, it is relatively straightforward to modify the allocation of functions for subsequent applications.

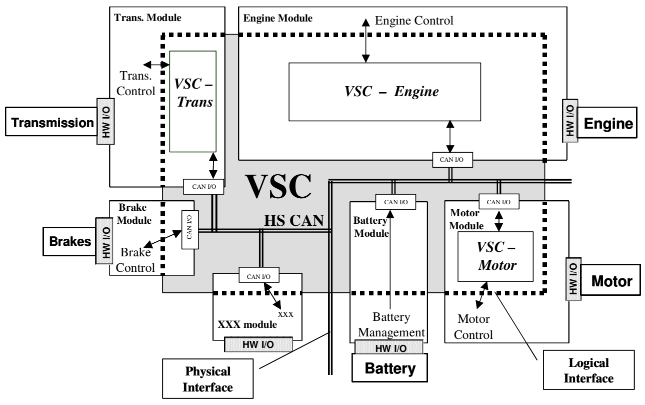


Figure 15. Physical allocation of VSC functionality within the ISG vehicle application.

VII. Conclusion

A novel method for managing complexity in large-scale control system design has been introduced. A hierarchical vehicle system control structure with modular elements and standardized interfaces forms the basis for this new method. In addition, a database tool has been presented that provides many advantages in the control system management process. Significant improvements in reusability and design flexibility have been achieved through the use of this new process. Finally, specific application to advanced development of hybrid vehicles is described.

Acknowledgment

The authors wish to thank Bengt Noren, Robert Eriksson, Per Jurland, Anders Lasson, and Jonas Jerrelind of Volvo Car Corporation and Tony Tsakiris, Pete Grutter, Hong Jiang, and Dave Bidner of Ford Motor Company, all of which contributed ideas to this work. Special thanks goes to Siyuan Jiang of Ford who put in significant effort to codify the database tool.

References

[1] Costlow, T., "Automotive Networking," *Automotive Engineering International*, vol. XX, pp. 80-83, May 2003.

[2] Phillips, A. M., Jankovic, M., Bailey, K. E., "Vehicle System Controller Design for a Hybrid Electric Vehicle", *Proceedings of the 2000 IEEE Conference on Control Applications*, Anchorage, Alaska, pp. 297-302, September 25-27, 2000.

[3] Rushton, G., Zakarian, A., Grigoryan, T., "Development of Modular, Electrical, Electronic, and Software System Architectures for Multiple Vehicle Platforms," SAE Technical Paper 2003-01-0139, Detroit MI, March 2003.

[4] Ormerod, J., Fussey, P., "Development of a Control System for a Mild Hybrid Vehicle," Proceedings of the International Workshop on Modeling, Emissions and Control in Automotive Engines, (MECA 2001) Salerno, Italy, September, 2001.

[5] Larsen, M., De La Salle, S., Reuter, D., "A Reusable Control System Architecture for Hybrid Powertrains," SAE Technical Paper 2002-01-2808, San Diego, CA, October 2002.