2007-01-0702

Model-Based Design Study and Evaluation of New HMI Concepts for Vehicle Multimedia, Climate Control and Navigation Systems

Paul F. Smith
The MathWorks, Inc.

Jie Chen, Hongxing Hu
Panasonic Electronic Devices Corp. of America

ABSTRACT

The application of Model-Based Design has been well established, documented, and highly refined practice for the development of embedded control systems like those used for power train or chassis controls.

Model-Based Design, when applied to in-vehicle electronic systems with human-machine interface (HMI) elements, presents unique challenges and opportunities.

Historically, development of the software to drive audio, climate control, navigation or other driver facing functions in a vehicle have been developed with traditional processes. These could include textural requirements and hand coding or application of structured design and analysis tools followed by hand coding. Production of the first working prototype using these traditional methods is much longer than with Model-Based Design. This is an emerging trend in the automotive body electronics design community.

This paper will describe the key design elements of the new HMI concept being developed and the development process used to produce a working prototype. Some key advantages of Model-Based Design for demonstration system development will be addressed. The next process steps to move the demonstration towards production will also be discussed.

INTRODUCTION

To improve drive safety and operational convenience, a new human machine interface (HMI) switch control concept was proposed and studied with a driver simulator-based simulation analysis [1]. This new concept is composed of a remote multi-functional switch module, located proximate to the driver such as on the steering wheel, a control module, and a visual display unit such as head-up-display (HUD) unit that provides the driver with indicia of the function controlled by each switch during operation. It offers an integrated control interface for the driver of a vehicle in operating such on-board devices as audio and multimedia, HVAC, and navigation systems. Furthermore, to improve functionality of the HMI switch control, a touch panel switch with figure writing input and character recognition capability was then proposed [2]. This newly added feature would provide a fast program search (for albums, songs, street names, etc…) mechanism that can be in particular meaningful for future car multimedia or infotainment systems. In order to make the design feasible for a wide range of vehicle applications, it is assumed that only a space-limited display will be available. Such kind of display may be the display on driving information center, navigation panel, or a simple vacuum fluorescent display (VFD) unit on top of dashboard that is not as sophisticated as the HUD unit.

The concepts of Model-Based Design were applied to provide for rapid requirements prototyping and analysis, system specification, and demonstration of these new HMI control concepts. Special block and state diagram based models were developed to represent the functionality of the system. Execution of the specification model in a PC-based hardware-in-loop simulation system including a prototype steering wheel switch module and a VFD unit was performed. This approach avoids the expensive and lengthy coding steps typically required to build functioning prototypes with traditional processes and allows the engineer to rapidly try out new ideas and also communicate new design concepts with internal and external customer groups. Properly architected, these models are compact, easily modified and extended and are relatively simple to read and understand.

THE HMI CONCEPT

Many different HMI design methods like haptic switches, touchpad switches, and speech/voice recognition have been evaluated, and their advantages and disadvantages have been studied [3-6]. A new concept HMI system that features multi-function switches as input devices to

control the infotainment system and a HUD as visual feedback was proposed [1] and implemented on a 2004 Grand Prix for concept demonstration [2]. Figure 1 shows the HMI switch control concept. In this design, the HUD projects the critical driving information and other necessary information onto the windshield and forms a imaginary image at the distance of the front bumper, which gives the driver visual feedback without forcing a change of viewing angle. It significantly reduces the distraction to the driver by eliminating the need to look away from the road for some other information. The multi-function switches with integrated menu navigation logic serve for input of many different control functions with a very limited number of switches, thus reducing the package space requirements. A control module (HMI ECU) hosts the menu navigation logic and controls the HUD display based on the switch input. This HMI design enhances driver's operation convenience and improves safety by keeping the driver's hands on the steering wheel and eyes on the road. Additionally, the reduced number of switches may also help OEMs to reduce the cost and allow more flexibility in interior design.

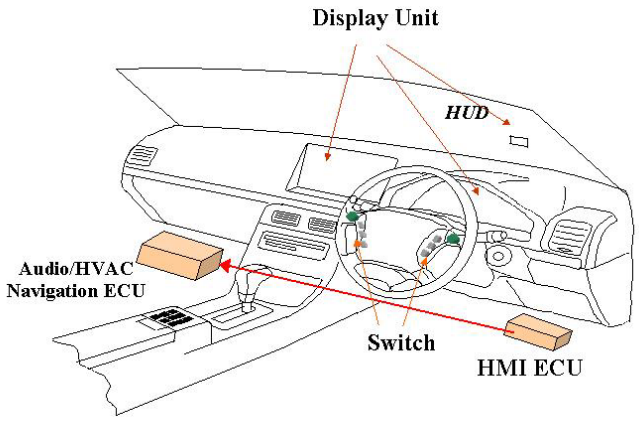

Figure 1. The HMI switch control concept

The current in-vehicle implementation uses a new OEM production radio, and includes most of the frequently used radio operations, including volume control, band, CD, forward/backward seeking, preset program selection, equalizer control, XM category and program selection. The HMI system consists of three modules, namely the steering wheel switch module, the heads-up display (HUD) module, and the gateway module (see Figure 2). Each module is controlled by a Motorola Star12 microchip with the proper memory size for the required software and communicates with each other through CAN (Controller Area Network). Vehicle information which comes in J1850 format is translated to CAN format by gateway module for HUD to display.

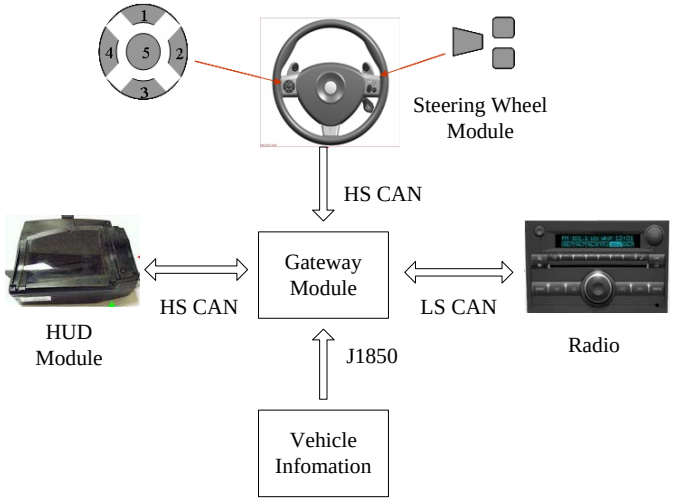

Figure 2. HMI system configuration

The study indicated that

- The menu navigation logic design has a significant influence on the user's overall rating of the system. How to design the menu logic such that the most frequently used functions are easy to access, as well as how many menu layers are within the users' comfort level need to be carefully studied[2].

- The HUD graphics design, including display contents, timing and layout is another key to the HMI design[2].

- Adding more control functions like Bluetooth cell phone, external media player control will add significant value to the system. Among these additional functions, iPod control appears to be the most important one based on the market trend.

- Different input device like touch panel that can interpret user's input including write and motion may bring additional convenience to operation.

- Other kind of display like the driving information center, the navigation panel, or a separate VFD (Vacuum Fluorescent Display) unit on top of dashboard need to be evaluated to provide display solution for different vehicle configurations. These commonly used display units are not as sophisticated as a HUD but have advantage in cost reduction.

In order to address these considerations in a time and cost efficient manner, a software method using Model-Based Design is adopted in the present study for rapidly concept prototyping. As it will be shown in the following section, this design approach will give portable HMI concept demo model that can be easily used for engineering evaluation and customer presentation. With Model-Based Design, new design change and modification may also be easily achieved and evaluated, which results in a significantly reduced development cycle. In this development work, the focus is on how to

control an iPod by using touch panel input with a secondary small display unit on dash.

DEVELOPMENT PROCESS

Model-Based Design has been well documented in previous publications [7-19]. The system described in the preceding section was initially prototyped using this approach. The key advantage of using Model-Based Design for prototyping was the separation of the functionality development from the in-vehicle integration. This results in speedy requirements validation and specification refinement. When it was time to build the system into a vehicle, the engineers were confident the system behavior was correct.

Models of the display menu dynamics were formed in Simulink® and Stateflow® from The MathWorks, Inc. These models allowed for rapid evaluation of the system requirements through simulation of system behavior. This approach allowed for and encouraged multiple iterations in the design and refinement of the underlying requirements. This approach is contrary to the view that up-front design and analysis leads to getting things right "the first time". Engineers tend to get things right through tinkering and multiple, evolutionary design improvements. Simulation models facilitate this rapidly converging design approach. Removing the hardware implementation details from the design cycle resulted in a net time save over the more conventional hardware based build and test design cycles still in use today.

Special model components are built to allow for the interfacing of the hardware switches to the PC-based simulation. These components are then archived and were reused when exploring modification of the design to include character recognition and a touch pad interface, which will be described in more detail below.

Porting the system to a vehicle required hand integration on the ECU and integration with existing electronic systems in the vehicle. Future work includes employing automatically generated code to further speed the movement to an in-vehicle based demonstration system and ultimately a production system.

Based on the above method, a model-based demo module developed (see Figure 3). This demo module includes two prototyped hardware units – a steering wheel switch module and a small VFD display unit, as well as a PC-based radio simulation controlled by the real hardware input. All supporting software including switch interface, menu navigation logic, and character recognition algorithm are PC-based too.

Figure 3: Model-based demo module

All system design endeavors begin with well documented and understood requirements. It is important to note that models embody requirements. They do not eliminate the need for requirements. With that said, models, due to their inherent ability to simulate behaviors, become a medium for rapid validation and evolution of requirements. Certain subtleties of, in this case, audio menus to be displayed are not understood until they can be seen animating on a screen. Simulation and animation allows for pruning of inefficiencies in the HMI. With HMI design efficiency interaction with the driver is a key design consideration. The menus presented to the driver must be both complete and present minimal distraction to the driver. Figure 4 is an example of the kind of written requirements that drove the development of behavioral models. In the diagram, the hardware interface the driver will use is presented and the basics of the operation of the various buttons and touch screen are described. The display menu layout is shown in the upper right corner of Figure 4.

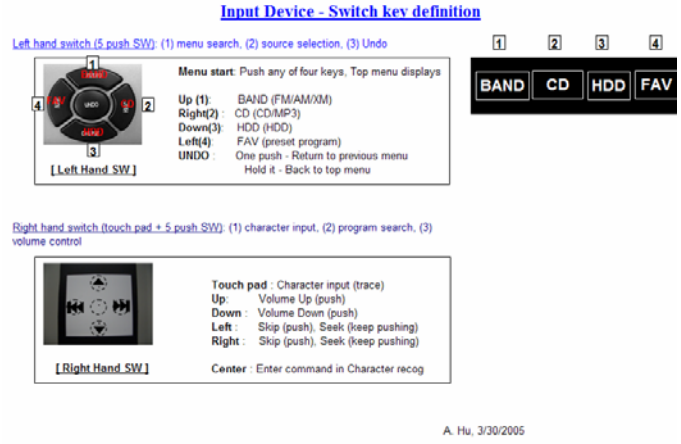

Figure 4: Example textural requirements document

As you drill down through the requirements of the HMI, painstaking detail on behavior of the menu system and underlying in vehicle systems must be described before the model can begin. Figure 5 shows one such diagram

that details expected behavior of the “Search by Artist” sub-menu.

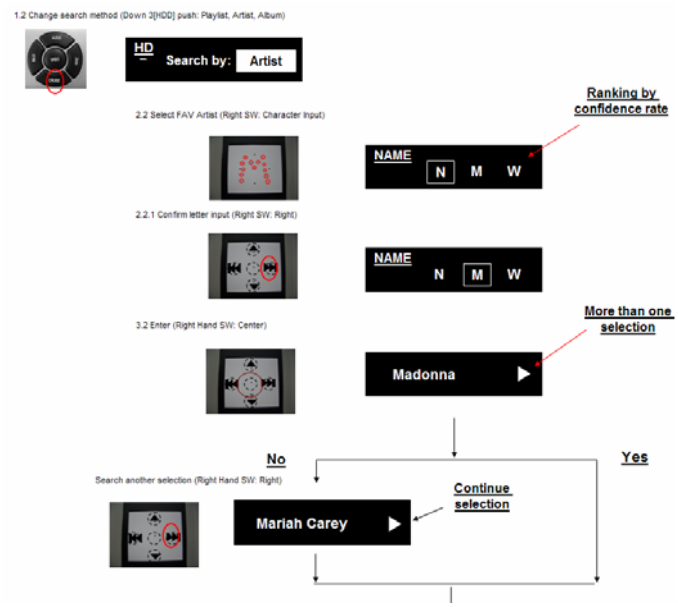

Figure 5: Example detailed requirements

Model-Based Design does not eliminate the need for detailed requirements. In fact, it is dependent on complete requirements just as traditional design methodologies are. Model-Based Design provides the added advantage that requirements defects, under-specifications, or conflicting specifications will be uncovered during the initial design phases when it is relatively cheap to make changes (before coding and hardware implementation). The graphical nature of the models developed aids in detailed visualization of the system architecture and behavior, neither of which can be done easily with a traditional design approach involving software coding before system behavior is demonstrated.

Applying Model-Based Design to HMI design allows the engineers to avoid the high cost and labor intensive activities of development of custom video display drivers, graphics file handling on embedded processors, or handling of custom in-vehicle communications protocols. These are unique challenges when compared with some of the other (powertrain, chassis) vehicle applications that have previously adopted Model-Based Design for powertrain or chassis systems development.

MODEL ARCHITECTURE

To gain a better understanding of these concepts, selected snippets of the model architecture are shown below. The high level view of the behavioral model is focused primarily on the control of the audio system. Extension of the concepts and tooling to navigation, climate control or other system do not introduce any new modeling concepts and are hence excluded from this paper.

The diagram in Figure 6 is the top level of the model in the graphical design language of Simulink. The outline of the system subset that will be shown in the following diagrams is given below in tree view:

HID_Radio
- Menu Logic Controller
  - Radio.Normal
    - HDD
      - Searchmode.Palylist
- Keyboard and Mouse Input
- Graphics Interface

At the top level major blocks describe the interface to the hardware switches connected to the PC-based simulation, the menu control logic, and the display driving logic.

The interface to the hardware switches has been customized to the particular hardware choices and is hence not entirely reusable. Conceptually, much of the interface between switches and the simulation is embodied in custom scripting or C code and incorporated in the model to act as an external interface. During simulation, button presses or activity on the touchpad will be decoded by this block and converted into a form that can drive event based state diagrams that model the menu behaviors in the Stateflow® design language.

On the right side of Figure 6 is a custom block that routes signals to the graphical output display that eventually ends up on a display unit. For simulation purposes, the display unit is simulated as a regular video display monitor and hence the only controls needed are PC-based services for driving a display. This greatly simplifies the engineers’ task during the initial phases of development as hardware details associated with control of a production display unit can be ignored.

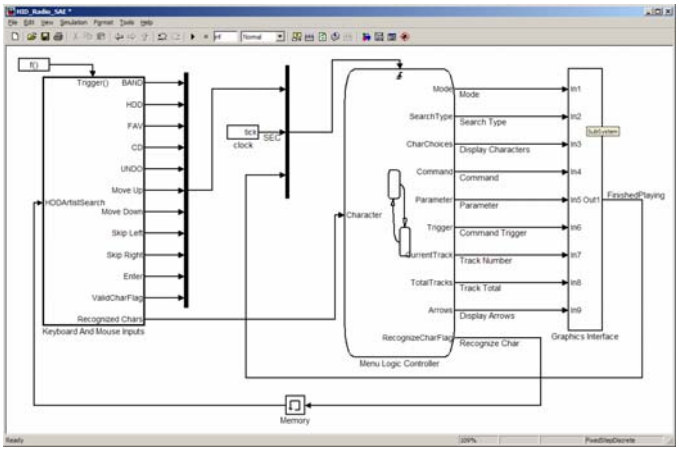

Figure 6: Top level model

At the top level of the Menu Control Logic is a state diagram that specifies system behavior from first initialization to normal “wait for input” states as shown in Figure 7. A graphical function is used to isolate and perform initialization actions on the audio system. Isolation allows the designer to quickly replace the

PC-based simulation actions from an alternate set when the system is taken to in-vehicle implementation by replacing just this graphical function (square box on the lower part of the diagram). This approach of graphically isolating hardware interfaces is used throughout the successful application of Model-Based Design. The alternative, co-mingling these functions with the rest of the design, leads to rework later during the hardware realization stages of the development.

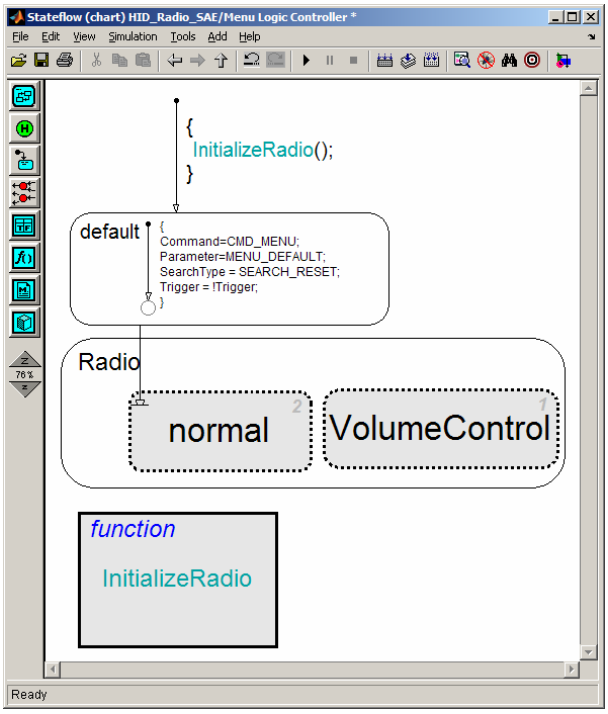

Figure 7: Main menu control state diagram

Drilling down further inside the "normal" sub state in Figure 7, the high level menu dynamics state diagram is found and is shown in Figure 8. This is the switch yard for dispatching events from the sensed hardware switches to control the various functions performed by the driver of the vehicle when operating the audio system. Getting the structure of this level of the model correct is essential to easy extension, debugging or redesign of menu behaviors. For this particular problem, sub states for audio band selection, preset programming, hard drive based audio file selection and CD player operation are shown. Navigation from any state to any other is allowed and is easily modeled in the design language. It is literally a few clicks on the mouse to add a new submenu or change the relationship between the menu items. Using traditional design hardware based development platforms with hand coding methods might require hours or days to perform the same level of reconfiguration.

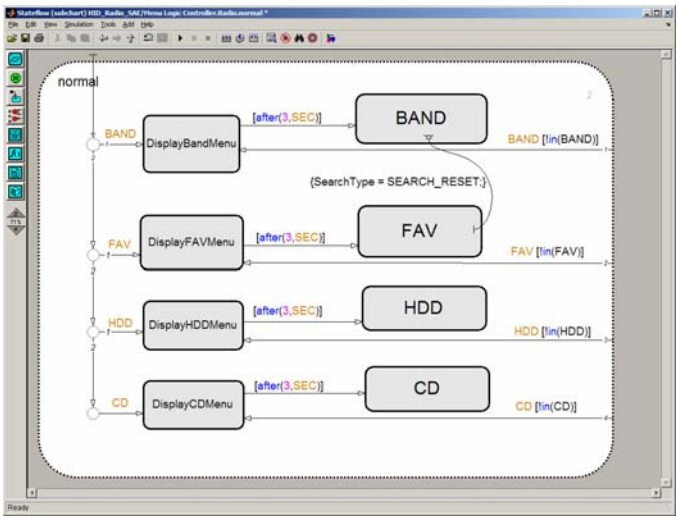

Figure 8: Main audio menu state diagram

Continuing to drill down into the heart of the system, we see the hard drive based audio file submenu state diagram labeled HDD in Figure 8 above. The contents of the HDD state are shown in Figure 9. Here the driver can select by artist or play list. Further special proprietary character recognition algorithms are integrated that allow the user to trace out letters on the keypad in the steering wheel and hit a "search" spot on the touch pad. The system then returns to the display unit a list of matches and the driver selects the music they want to play.

Again the idea of isolation of the various pieces of the algorithm (menu controls from character recognition, in this case) allows the designers to work on the different pieces of their system independently or in parallel. Movement to the vehicle is then streamlined by easily splicing in or out implementations of the modeled behavior that run efficiently in the target hardware. This keeps the concept design free of fixed point or device specific implementations details. It also then allows the design to compartmentalize these details out of view even when the system moves closer to production implementation.

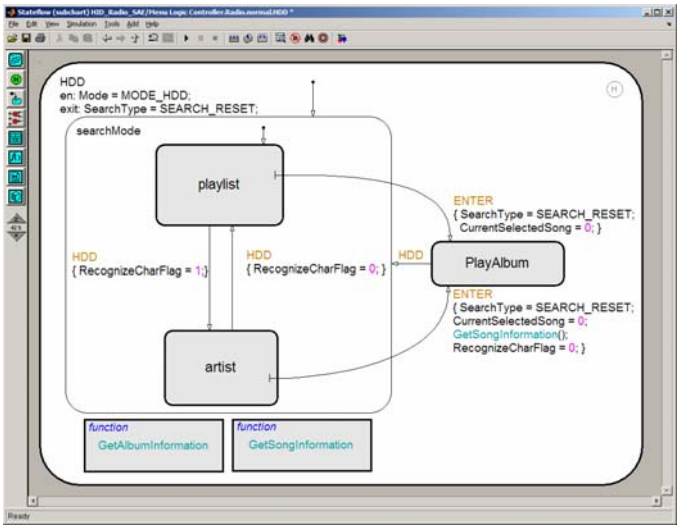

Figure 9: HDD menu state diagram

The final layer of detail, an example of which is shown in Figure 10, specifies exactly what happens in each menu state when the various buttons are pushed. Abstract action language allows for isolation of implementation from bigger picture behavioral operation of the system. MBD allows for interactive animation of the diagrams shown here while the system is simulating which aids in the detailed understanding of the subtleties in the requirements. Commonly, this results in the discovery of requirements defects or shortcomings.

The diagram shown in Figure 10 demonstrates the integration of both state based logic specification and flow chart based specification. Within the state, a detailed flow chart is built which describes behavior when each steering wheel button is pushed. This is the bottom of the hierarchy in the model. No further specification is needed illustrating one key advantage of this approach – simple, compact, easy to maintain models.

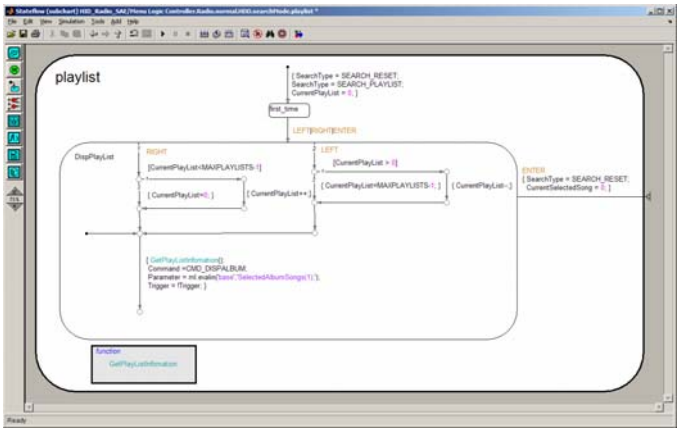

Figure 10: Integrated state and flow logic

PRODUCTION IMPLEMENTATION NEXT STEPS

The movement from modeled behaviors to an in-vehicle implementation is not a trivial one. Given a properly architected model, the transition can be managed in a methodical and systematic fashion. The key to making the transition a smooth one is the model architecture used during concept development. If implementation details which are different between PC-based simulation and in-vehicle demonstration are co-mingled with the behavioral models, the engineers will have to first unravel the two and then rebuild the models. This will result in new errors being introduced. Sequestering the interface to the hardware implementation is essential in avoiding this rework and error injection point.

For this work two stages of vehicle implementation are required. The first is to build a demonstration vehicle. The second is to build a production system.

The development of a demonstration vehicle can further leverage Model-Based Design. Rapid-prototyping systems are currently being used by powertrain and chassis system designers. Similar techniques can be applied to driver information systems. [8,9,19]. Automatic code generation directly from the model can be targeted at a special prototyping hardware platform with enough memory and processing power to handle the task of concept prove-out. This typically implies floating point processors, standard CAN interfaces and supporting model blocks to drive them and a series of basic input/output device support (AtoD, DtoA, DIO, etc…).

The final implementation integrated into a production vehicle brings additional constraints and design considerations. The final implementation may be in fixed-point software code. It will likely be integrated with other electronic systems under control and hence have to share processor resources (memory and CPU). The code generated from the models will have to integrate cleanly with legacy code that drives platform services (operating systems, input/output device drivers). The process and tool chain for integrating code will also have to be addressed. Things like memory map management, data structure definitions, function calling interfaces, calibration or data acquisition device interfacing and other similar issues will all have to integrate smoothly. Finally the code generated from models needs to be compact and fast. Tremendous progress has been made over the last decade to this end and many have concluded that for most applications automatic code generation is on par with average hand coding [9,12,13,19].

There are capabilities built into the tool chain for Model-Based Design that facilitate or automate many of the tasks outlined above [10,11,14,16-19]. It does not eliminate the tasks themselves. By systematically applying the concepts described above for architecting models and refining the system through the various phases of prototype demonstration, the HMI designer can have confidence that the main functionality contained in their design work – the interface to the driver – can be developed with confidence prior to complicating the problem with production or pre-production hardware.

CONCLUSION

The application of Model-Based Design has been a well established, documented, and highly refined practice for the development of embedded control systems like those used for powertrain or chassis controls.

Model-Based Design, when applied to in-vehicle electronic systems with human-machine interface (HMI) elements, presents unique challenges and opportunities.

The use of mixed continuous and discrete state and flow logic driven models to validate and refine requirements for HMI based in-vehicle systems has been shown to be an effective technique to isolate the design engineer from the error-prone and labor intensive hardware implementation. This is especially important during concept development, which is a critical design stage for the HMI developer. Such software-based concept model can also be used as a demo module for initial customer

presentation and evaluation. This may really help the design engineers to gain customers' feedback at an early stage of such advanced technology development at a cost effective way.

A complex menu navigation system was prototyped and shown to minimize driver distraction and provide quick and easy access to important audio system control functions. The simulation models provided a framework for rapid system evolution, encouraging engineers to do what they do best.

Future work to pursue automatically generated embeddable code from specification models for HMI systems is only possible when models are properly architected. Given the correct model structure, many different designs can be quickly and efficiently evaluated, thus giving confidence to the engineer that they have the "right" design prior to the costly hardware implementation phase.

REFERENCES

1. Kiyotake Sasanouchi, Yasushi Ishiai, Tsuyoshi Tanaka, Junichi Yukawa and Austin Hu, "Development of the HMI system that improves the safety and operational convenience by the combination of steering wheel switch and head up display", SAE Paper 2005-01-0438 (2005)

2. Hongxing Hu, Jie Chen, Junichi Yukawa, Leah Luo, Tsuyoshi Tanaka, Masashi Tada, Kiyotaka Sasanouchi, Kouichi Santo, Katsu Saito, Yasushi Ishiai and Ted Cates, "Study and Evaluation of New HMI Switch Control Concepts Through Demo Vehicle and Module Development", SAE Paper 2006-01-0815 (2006)

3. N. Takubo, M. Kihira, "Analysis of glances on navigation devices on the basis of the accident data", Proceedings of Symposium on Mobile Interactions and Navigation", pp. 109-112 (2002)

4. Paul Green and Jonatha, "Future in-car information systems: input from focus groups", SAE Paper 920614 (1992)

5. P. Knoll and W. Koenig, " Human-machine interface: how to make it simple & effective", Convergence 2002 International Congress on Transportation Electronics, Oct. 2000, SAE paper No. 2000-01-C109.

6. K. Itoh, Y. Miki, N. Yoshitsugu, N. Kubo and S. Mashimo, "Evaluation of a voice-activated system using a driving simulator", SAE Paper 2004-01-0232 (2004 SAE International Congress)

7. The MathWorks, Inc. – Automotive Technical Literature: http://www.mathworks.com/industries/auto/technicalliterature.html

8. Grantley Hodge, Jian Ye, Walt Stuart, "Multi-Target Modelling for Embedded Software Development for Automotive Applications", SAE Paper 2004-01-269

9. Jeff Thate, Larry Kendrick, Siva Nadarajah, "Caterpillar Automatic Code Generation", SAE Paper 2004-01-0894

10. Arvind Hosagrahara, Paul Smith, "Measuring Productivity and Quality in Model-Based Design", SAE Paper 2005-01-1357

11. Jon Friedman, Jason Ghidella, "Using Model-Based Design for Automotive Systems Engineering - Requirements Analysis of the Power Window Example", SAE 2006-01-1217

12. Tom Erkkinen, "Safety-Critical Software Development using Automatic Production Code Generation", SAE Paper 2007-01-1493

13. Paul Smith, Sameer Prabhu, Jon Friedman, "Best Practices for Establishing a Model-Based Design Culture", SAE Paper 2007-01-0777

14. Brian McKay, "Open Architecture Solution for Hardware-in-the-Loop (HIL) Testing", SAE Paper 2007-01-0501

15. Jonathan Friedman, Chris Fillyaw, Sameer Prabhu, "Creating Human Machine Interface (HMI) Based Tests within Model-Based Design", SAE Paper 2007-01-0780

16. Proceedings from The MathWorks International Automotive Conferences:

- http://www.mathworks.com/company/events/programs_de/iac2004/iac_confirm.html
- http://www.mathworks.com/industries/auto/iac/presentations.html
- http://www.mathworks.com/industries/auto/iac06/presentations.html

CONTACT

Paul Smith
Paul.Smith@mathworks.com
www.mathworks.com

DEFINITIONS, ACRONYMS, ABBREVIATIONS

AtoD – Analog to Digital

CAN – Controller Area network

CPU – Central Processing Unit

DIO – Digital Input/Output

DtoA – Digital to Analog

ECU – Electronic Control Unit

HMI – Human Machine Interface

HUD – Heads Up Display

HVAC – Heating, Ventilation, Air Conditioning

MBD – Model-Based Design

VFD – Vacuum Fluorescent Display