White Paper

Implementing a Flexible CPLD-Only Digital Dashboard for Automobiles

Introduction

An automotive dashboard acts as a nerve center that consolidates all information pertaining to the safe driving and management of the vehicle, and displays it for the driver. In the present digital age, the vehicle's digital instrumentation system is required to monitor all key functions, and even be personalized. The industry requirements have led to a number of semiconductor solutions, ranging from ASSPs to full-custom devices. These may be fixed-function solutions with restricted or lacking product development options that designers need for solution flexibility. In comparison, a scalable solution can support multiple similar applications, such as a full product line of vehicles without any cost overhead. This type of custom solution can deal with all requirements for a lower price.

This paper discusses a radical and innovative architecture using CPLDs to eliminate the use of microcontrollers and its drivers completely, thereby obtaining a low-cost, low-power digital solution for analog dashboard clusters. This analog dashboard solution (ADS) helps to effectively implement a digital car network and exploit the advantages of the digital world.

Traditional Solutions for a Dashboard Cluster

Traditionally, the real-time outputs from the instruments such as speedometer, odometer, etc. were obtained mechanically and displayed using analog drivers. However, with the digitization of these data inputs, stepper motors and LEDs replaced meters and gauges. Expensive microcontrollers were used to process and display the digital output, which is of no practical significance on a dashboard. ASSPs next appeared, leading to high NRE costs, which limited upgradeability and improvements. The product life span and support for different product lines also was a major factor in favor of a cheap programmable alternative. A developed product must cater to different vehicle requirements and support use of different sensors without any major changes. These limitations render the traditional solutions costly.

Stepper motors used in pointer-dial type displays are devices that convert electrical pulses into discrete mechanical movements. The shaft or spindle of a stepper motor rotates in discrete step increments when electrical control pulses are applied to it in the proper sequence. The motors' rotation has several direct relationships to the applied input pulses. As a digital instrument cluster usually employs stepper motors to emulate the performance and visual efficacy of an analog dial and pointer display and simultaneously to provide excellent positional accuracy demanded in a digital setting, micro-stepping these motors becomes necessary to achieve smooth and non-quantitized motion of the pointer. In addition, as the number of measured samples that can be broadcast from the vehicle sensors to their respective meters is limited by the bandwidth of the digital link between them, the samples for each quantity displayed in the instrument cluster are made available after defined time intervals. A scheme for overcoming the absence of continuous information to the display becomes increasingly necessary in such clusters to ensure that the pointer motion makes up for the time no data is available to the meter from the sensors. These challenges increase the processing power required and thereby the cost of a completely digital dashboard system, deterring its use in vehicles owing to the unfavorable price/performance ratio.

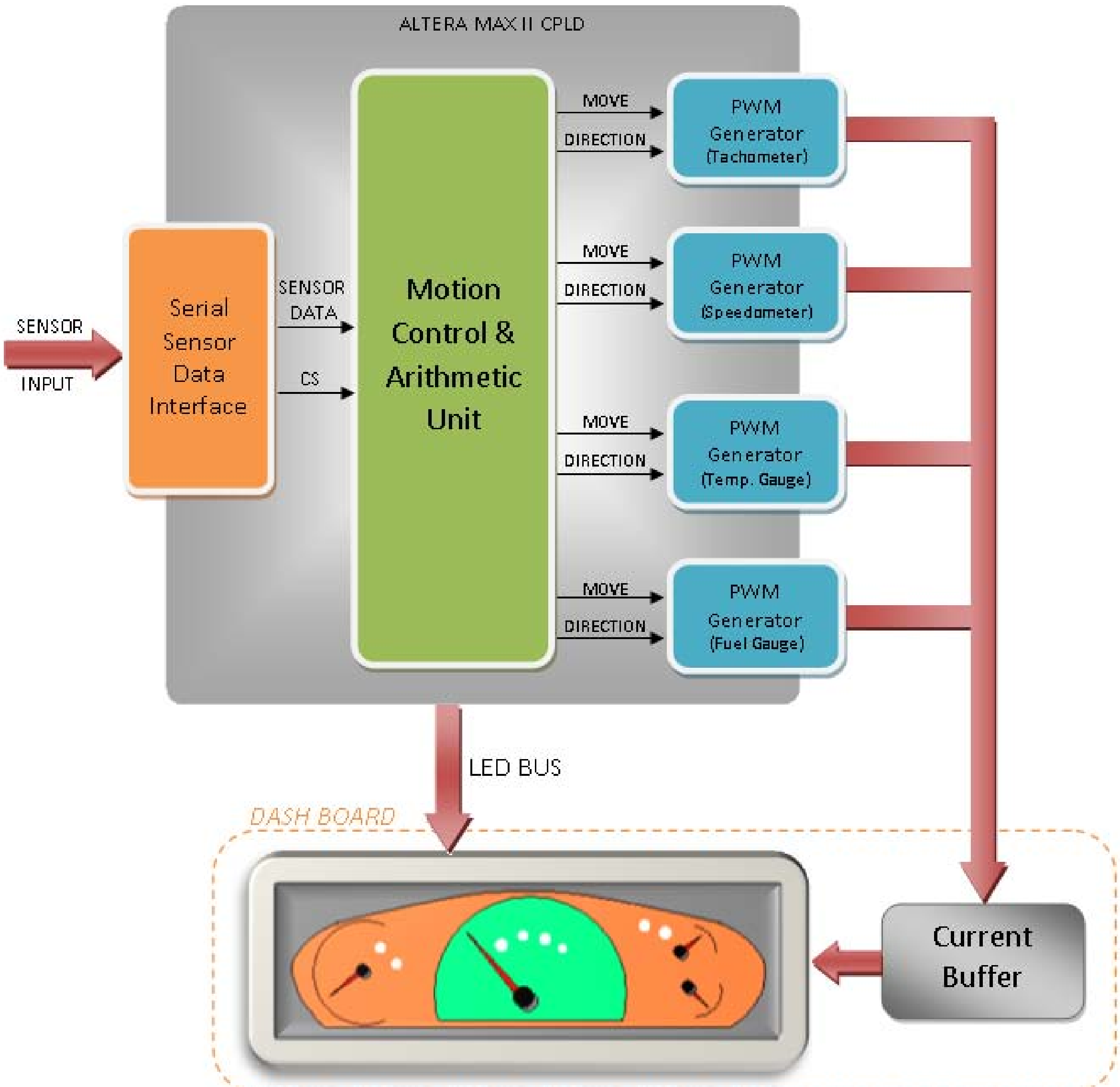
CPLD-Based Dashboard Cluster Controller

The limitation of expensive solutions can be overcome simply by using a CPLD. The ADS offers customers much more flexibility during the design cycle because design iterations are simply a matter of changing the programming file, and the results of design changes can be seen immediately in working parts. Additionally, it is possible to add new features or upgrade products that already are in the field, thereby making room for effective technology deployment and meeting specific user and product requirements. The development cycle is reduced greatly due to the availability of such ADSs. Using the same basic system, it is possible to incorporate slight modifications to support different devices on new product lines.

WP-01072-1.0

Altera has developed this CPLD-based ADS, enabling product developers and manufacturers to choose from several devices based on their needs and without concern about semiconductor component obsolescence. The ADS's selling point is the drastically low cost of implementation while still supporting higher end features and immense scope for future expansion. The architecture uses Altera® MAX® II CPLDs and includes six modules: the serial sensor data unit, the main motion control and arithmetic unit, and four pulse-width modulation (PWM) generators. The serial sensor data unit takes in input from the sensors, the main motion control and arithmetic unit performs all the calculations required, and the PWM generators provide appropriate control signals to all the phases of the stepper motor, taking in commands from the main module. A block diagram of a sample ADS is shown in Figure 1.

Figure 1. CPLD-Based ADS Block Diagram

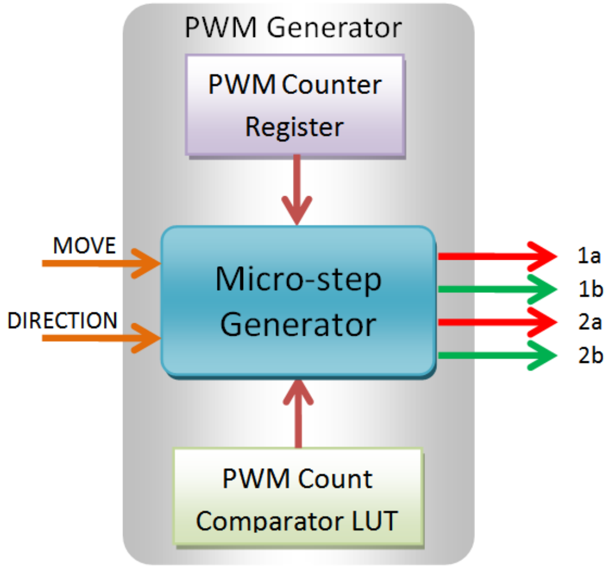


To further describe the elements of the CPLD-based ADS architecture, it will be useful to understand the function of different blocks and how micro-stepping is performed for the stepper motor that drives the pointer dials, using a low-bandwidth vehicle data network. Because sensor data is not continuously required for all the displays on the cluster, the system holds the pointer positions when data is not available. For smooth action, the move commands sent to the pointers are a function of the current deflection to make sure there are no jerky step changes in the pointers.

PWM Generators

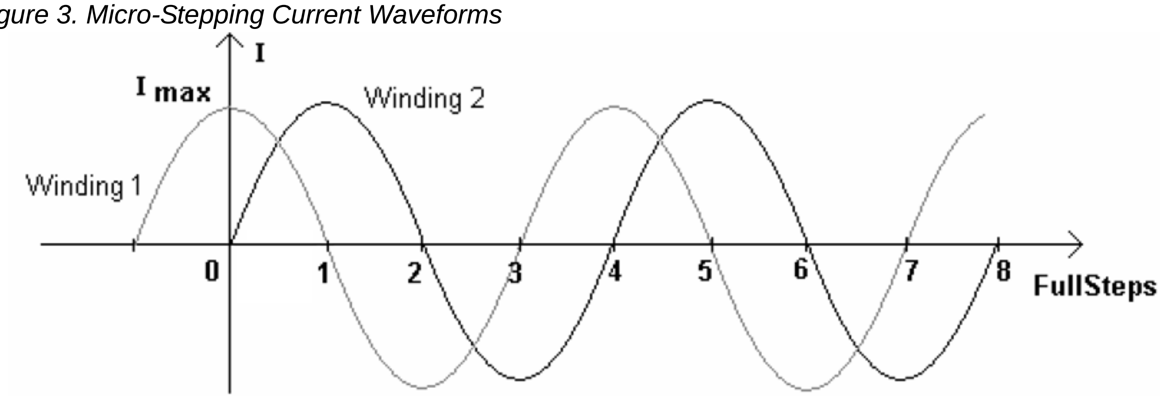
Four PWM generator modules drive the pointer-dial stepper motors to indicate data from different sensors. A basic block diagram of the PWM generator is shown in Figure 2.

Figure 2. Block Diagram of the PWM Generator



Micro-stepping is used to drive the motors. As mentioned earlier, this makes the movement of the pointer smooth. In micro-stepping, the resultant magnetic field developed by the motor is created at positions that are not in line with either of the field coils, but at some angle to them. This way the holding torque can be developed at more positions, allowing the rotor to be held in space between the two extreme field coil axes. When a field coil is energized, the magnetic flux produced by it is proportional to the current flowing through it. If both field coils are excited, the resultant direction of the current, and hence the resultant magnetic field, is obtained by the vector sum of the currents in the two windings. Therefore, if the current in the windings of the stepper motor is changed incrementally, such that a set of equally spaced positions of the resultant magnetic field is created, the stepping resolution of the motor can be increased. Using this underlying principle, the step of a stepper motor is divided into sub-steps called micro-steps over which the shaft is actually moved. Figure 3 shows the sinusoidal current waveforms for the two windings during micro-stepping.

Figure 3. Micro-Stepping Current Waveforms



The current corresponding to each of the micro-steps is calculated from the sine wave. When voltage in the form of PWM waves with pre-calculated duty cycle values is applied to each of the motor windings, the required current for a particular micro-step can be made to flow through them. Here the windings behave as R-L circuits, filtering the PWM pulses and thereby maintaining a constant current.

The function of the four PWM modules is to keep a constant supply of PWM pulses of a fixed duty cycle value to the stepper motor, thus making the pointer move at a particular speed and keeping the four stepper motors in constant motion. They receive move commands along with direction input from the motion control units and apply appropriate voltages across the windings to move each motor one micro-step in the desired direction.

Motion Control and Arithmetic Unit

The delta generator is the main component in the motion control and arithmetic unit. This module receives the target deflection for each of the four stepper motors from the sensors. It also keeps track of the current deflection of the motors. When it receives the target deflection, the delta generator sends two signals to the respective PWM modules, depending on the range that must be spanned by the shaft. One signal is a trigger pulse that tells the PWM module to change the current duty cycle value to the next value, i.e., to speed up. So when the range to be spanned is large, the pulses are sent at a high rate and the PWM module steps through the various duty cycle values very quickly, thus turning the stepper motor at a high speed. When the pointer reaches the target deflection, the trigger pulses to the PWM module stop and the PWM module supplies pulses of a constant duty cycle value, effectively holding the shaft in its position. The other signal from the delta generator tells the PWM module which direction the pointer should move.

The delta generator has second important function. The sample values vary continuously, however, the data representing this information is sent intermittently. That is to say, at one instant, data for the first meter might be 30 units, and the next data that comes in might be 45 or 50 units. If this kind of abrupt information is represented directly on the stepper motor, the pointer is going to move in jerks. To avoid such movement, the delta generator uses an update number referred to as δ. Each time it receives a new value for the target deflection, the generator recalculates δ for that particular pointer. This incremental number δ is then added to the current deflection of the pointer at a rate much higher than the rate at which a new target deflection is received. This is explained mathematically as:

$$\delta = \frac{\textit{Target deflection} - \textit{Current deflection}}{\textit{Update rate} \times \textit{Sample time}}$$

The choice of update rate and sampling time should be such that even for the greatest possible numerator in the above formula, $1/(\textit{Update rate})$ is always a decimal value. After calculating δ using the above formula, it is then added to the current deflection every seconds and the new current deflection value is obtained. During this entire operation, the unit’s place of the current deflection is continuously monitored. Every time it changes, a command to move the stepper motor of the meter by one micro-step in the required direction is sent. If the target deflection is greater than the current deflection, the direction is clockwise, and if it is smaller, the direction is counterclockwise.

Thus in the interval between the receipt of two target deflection values, the pointer is moved at a faster rate, through smaller steps, to reach the target, thus resulting in a smooth movement. In essence, it extrapolates the value appropriately to cover the gaps and keep the motion smooth irrespective of the input variations.

In addition to controlling the motion of the stepper motors, this module also sends commands to light up particular indicator lights on the panel according to the sensor input received.

Sensor Data Input Unit

The sensor data input unit uses an SPI interface, a serial bus standard that supports communication with slow peripheral devices that are accessed intermittently. These devices communicate using a master/slave relationship, where the master initiates the data frame. When the master generates a clock and selects a slave device, data may be transferred in either or both directions simultaneously. An SPI slave module has been implemented at the CPLD’s back end for input to the system. The sensor data is ported to the arithmetic and motion control unit and is the target deflection of a particular sensor or new LED data. The data is sent along with the target address, so that sensor data identifies its source.

Demonstration Design

The ADS used for the demonstration design has four pointer-dial displays: tachometer, speedometer, temperature indicator, and fuel gauge, plus a few LED indicators for turn, seat belt, hand brake, etc. As mentioned earlier, a SPI slave took in time-multiplexed sensor input signals. The data-sampling rates for different sensors were the usual rates of incoming data for an automobile. The tachometer received data every 125 ms since it must respond very quickly to rapid changes in input. Based on similar information, the speedometer, temperature gauge, and fuel gauge data rates were kept at 250 ms, 500 ms, and 1000 ms, respectively. These data were sent and shifted in time according to the system's data rate determined by a PC host following standard SPI protocol. Other than this data, the input bus was free for other automobile environment operations. The incoming data format was a typical address-data format.

The command sent to the slave was a 16-bit packet that identified and addressed one of the four-stepper motors and provided the target deflection. This piece of data was taken by the SPI slave and handed over to the control and arithmetic unit for processing. The 16 addresses were used to identify four pointer-dial sensor outputs and 12 LED indicators. Following the four bits used for addressing, the sensor data was sent. The speedometer and tachometer data was 12 bits each, while temperature and fuel gauge data were kept at 10 bits each since a high resolution is not required for them. The LED data was one bit.

VID-29 stepper motors were used for the pointers, and three field poles were placed symmetrically around an 8-pole rotor. The motor had two field coils, which generated the magnetic field that was distributed to the field poles. Due to its construction and geometry, the resultant magnetic field created in the motor rotated uniformly when the sinusoidal currents, phase shifted by 60 degrees, were driven through its coils. At any point, the angular displacement of the rotor due to this field was proportional to the phase angle of the current's sine wave. The rotor moved by two full steps (or 360 degrees) during one sine wave cycle. With each full step further broken down into 12 micro-steps, the step resolution of the motor was reduced to one-twelfth of a degree at its shaft. The current magnitude corresponding to each of these micro-steps was calculated by dividing one period of the sine wave into 24 equal parts.

This technique of calculating the driving signal was not very effective against a goal of reducing angular error. Other factors such as motor construction and air gap flux leakage needed to be considered to ensure that the rotor was displaced by a constant angle in every micro-step. Formulae for calculating these driving current values for each micro-step were provided by the VID-29 manufacturers as a function of the peak current (I_{MAX}). This information, along with results obtained from SPICE simulations, was used to get the right set of duty cycle values for the smooth motion of the stepper motors.

All this information was processed in the PWM generator to provide the required output at the four winding terminals of the motors. The PWM generator controlled the motors using two inputs, "move" (used to make the rotor move by one micro-step) and "direction" (used to specify the direction of motion). The motion control unit processed the input deflection and generated move and direction signals appropriately.

Resource Utilization

Altera MAX II CPLDs were used to implement this proposed ADS. The device density of these LUT-based CPLDs are measured in terms of logic elements (LEs), and are available in 240-, 570-, and 1270-LE variants. The ADS architecture can be successfully implemented on a 570 LE CPLD using 555 LEs along with a 4.4-MHz internal oscillator to eliminate the need for an external clock circuit. These size constraints do not sacrifice the overall performance of the system.

The CPLD-based ADS can be scaled up easily for even more accuracy and higher-end functionality for different platforms. It can support different assembly topologies for different classes of vehicles with minimal changes in programming and almost no additional expenditure, as the development time decreases exponentially. The ADS can be coupled with more common and robust automotive digital data networks for instant induction into the manufacturing process. This provides the flexibility of choosing an appropriately sized CPLD for implementation along with all the add-on functionalities.

Conclusion

Using a low-cost, low-logic density CPLD, a complex and critical ADS can be achieved with predominantly off-the-shelf microcontrollers and expensive custom silicon solutions. This scalable and flexible architecture overcomes many of the shortcomings of traditional dashboard solutions. Because re-programmability is the inherent advantage of the ADS, the re-usability of the design means additional solutions can be developed quickly using the growing library of available intellectual property (IP) and cores. Reconfiguration is easy and newly developed products reach the customer faster and more easily. NRE costs can be recovered easily because of long product life, and manufacturers can prolong the lifecycle of an already developed product without having to make new NRE investments.

Further Information

- "Digital System for Controlling Mechanical Instrument Gauges," U.S. Patent No. 4939675: http://patft.uspto.gov/
- Microchip AN822: "Stepper Motor Micro Stepping With PIC18C452": www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1824&appnote=en011997
- "C2000 for Camera Dome Control," Yanming Luo, Texas Instruments Developer's Conference, June 2007: http://focus.ti.com/download/tidc/pdfs/asia/2007/T2_5_Yanming_Luo_C2000_for_Camera_Dome_Control.pdf
- *A Flexible Architecture for Fisheye Correction in Automotive Rear-View Cameras:* www.altera.com/literature/wp/wp-01073-flexible-architecture-fisheye-correction-automotive-rear-view-cameras.pdf
- *Creating Low-Cost Intelligent Display Modules With an FPGA and Embedded Processor*: www.altera.com/literature/wp/wp-01074-creating-low-cost-intelligent-display-modules-with-fpga.pdf
- *Applying Graphics to FPGA-Based Solutions*: www.altera.com/literature/wp/wp-01075-applying-graphics-to-fpga-based-solutions.pdf
- *Using LEDs as Light-Level Sensors and Emitters*: www.altera.com/literature/wp/wp-01076-leds-as-light-level-sensors-and-emitters.pdf

101 Innovation Drive
San Jose, CA 95134
www.altera.com