Evaluation of Input/Output ECU in Trucks

[1] C Girish, [2] Dr. Chethana K, [3]Sreekanth Kancham
[1] M.Tech VLSI D and ESECE, [2] ECE, DSCE, [3] Advanced Engineering
[1,2] Dayananda Sagar college of Engineering, Bangalore, India
[3] VIPL, VOLVO INDIA PRIVATE LIMITED, Bangalore, India

Abstract: The "Evaluation of I/O ECU in Trucks" which focuses on hardware and software level aspects of the Input/output ECU (I/O ECU). The I/O ECU interfaces with the sensors and actuators of the truck and the controlling action is taken by the High End ECU which interacts with the I/O ECU by exchange of data over a communication channel.

***Index Terms*- Input / Output Electronic Control Unit, High Side Driver, Low Side Driver, High End ECU and Serial Peripheral Interface.**

I. INTRODUCTION

This paper focuses on the evaluation of the hardware and software platforms of the I/O ECU and its subsystems adopted in the trucks.

The evaluation is done by developing a bare-metal code in the software environment and then imparting the program to specific hardware target and test for its functionality. The software developed should be generic and must be able to interface with the homogenous and heterogeneous platform hardware modules.

The I/O ECU in commercial vehicles refers to the system or modules that directly interacts with the sensors and actuators of the vehicle and then relay it to the High End ECU for processing. The hardware subsystems of the I/O ECU includes the low power Microcontroller, Low Side Drivers and High Side Drivers, Analog to Digital Convertors, Shift registers and CAN

II. PROPOSED SYSTEM

A. I/O ECU

The I/O ECU is the customized base board with variant functionalities to perform by the on board peripherals, which include the High side drivers(HSD), Low side drivers(LSD), CAN communication and shift registers [1] etc. The I/O ECU has S32K144 low power microcontroller an ARM cortex M4 32-bit with three SPI communication channel.

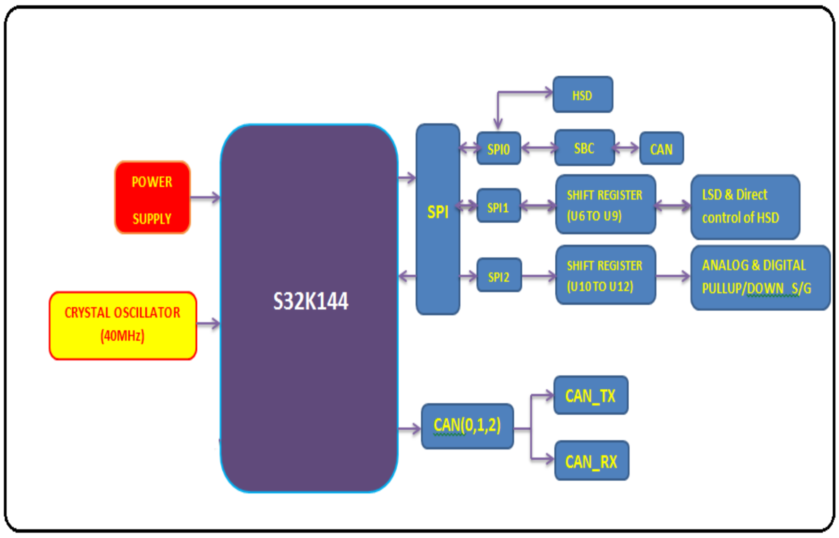

POWER SUPPLY
CRYSTAL OSCILLATOR (40MHz)
S32K144
SPI
SPI0
SPI1
SPI2
HSD
SBC
CAN
SHIFT REGISTER (U6 TO U9)
SHIFT REGISTER (U10 TO U12)
LSD & Direct control of HSD
ANALOG & DIGITAL PULLUP/DOWN S/G
CAN(0,1,2)
CAN_TX
CAN_RX

Fig1. Simplified Block Diagram of I/O ECU

The shift register is configured in the daisy chain arrangement that moves or shifts the data along the shift registers in the form of serial to parallel data conversion, where the data coming out of the shift register are signifying a particular entity which will activate a peripheral [6].

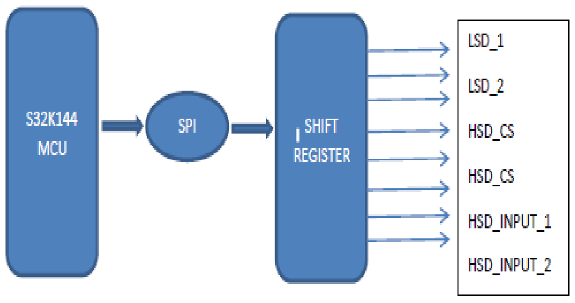
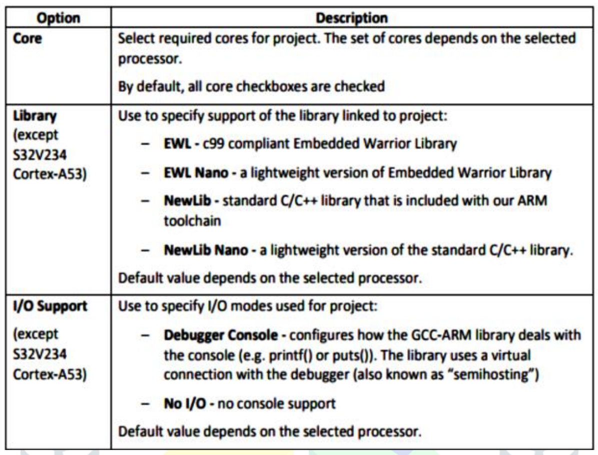
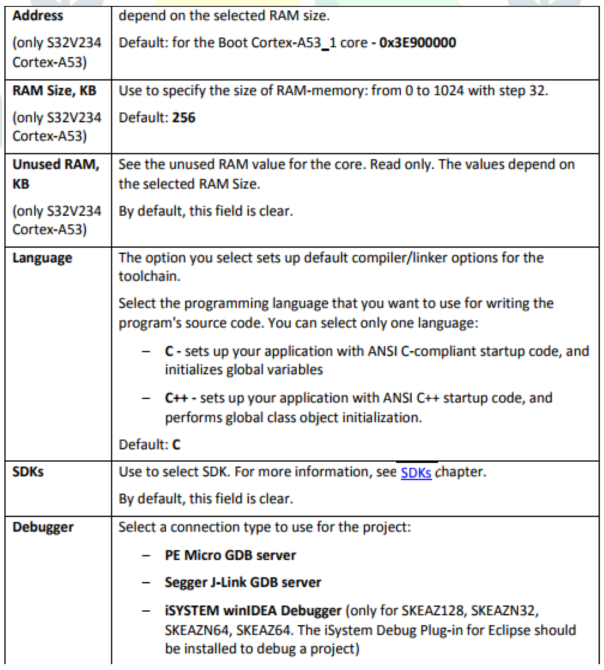


Fig 2. Interaction of MCU and the Shift Register.

The interaction between the shift register and the Microcontroller Unit (MCU) is through the on chip Serial Peripheral Interface (SPI).

Option	Description
Core	Select required cores for project. The set of cores depends on the selected processor. By default, all core checkboxes are checked
Library (except S32V234 Cortex-A53)	Use to specify support of the library linked to project: – **EWL** - c99 compliant Embedded Warrior Library – **EWL Nano** - a lightweight version of Embedded Warrior Library – **NewLib** - standard C/C++ library that is included with our ARM toolchain – **NewLib Nano** - a lightweight version of the standard C/C++ library. Default value depends on the selected processor.
I/O Support (except S32V234 Cortex-A53)	Use to specify I/O modes used for project: – **Debugger Console** - configures how the GCC-ARM library deals with the console (e.g. printf() or puts()). The library uses a virtual connection with the debugger (also known as "semihosting") – **No I/O** - no console support Default value depends on the selected processor.
Address (only S32V234 Cortex-A53)	depend on the selected RAM size. Default: for the Boot Cortex-A53_1 core - **0x3E900000**
RAM Size, KB (only S32V234 Cortex-A53)	Use to specify the size of RAM-memory: from 0 to 1024 with step 32. Default: **256**
Unused RAM, KB (only S32V234 Cortex-A53)	See the unused RAM value for the core. Read only. The values depend on the selected RAM Size. By default, this field is clear.
Language	The option you select sets up default compiler/linker options for the toolchain. Select the programming language that you want to use for writing the program's source code. You can select only one language: – **C** - sets up your application with ANSI C-compliant startup code, and initializes global variables – **C++** - sets up your application with ANSI C++ startup code, and performs global class object initialization. Default: **C**
SDKs	Use to select SDK. For more information, see SDKs chapter. By default, this field is clear.
Debugger	Select a connection type to use for the project: – **PE Micro GDB server** – **Segger J-Link GDB server** – **iSYSTEM winIDEA Debugger** (only for SKEAZ128, SKEAZN32, SKEAZN64, SKEAZ64. The iSystem Debug Plug-in for Eclipse should be installed to debug a project)

III. RESULTS OF EVALUATION

Base Board evaluation is done at both levels of integrity that is the software and the hardware. The evaluation of the HSD is as follows down below.

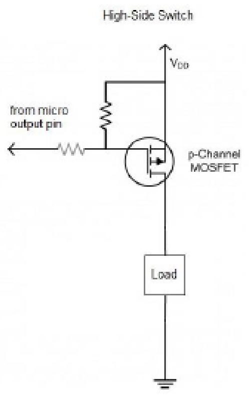


Fig 3. High Side Driver Evaluation

The evaluation of the HSD is at the input voltage of 10V and the output voltage is at the range of 0 to 10V, the test of Pulse Width Modulation (PWM) of 50%, and 90% was tested and the functionality is optimum without the glitch or noise. The shift register configures the same to multiple number of HSD.

The evaluation the LSD is as follows below,

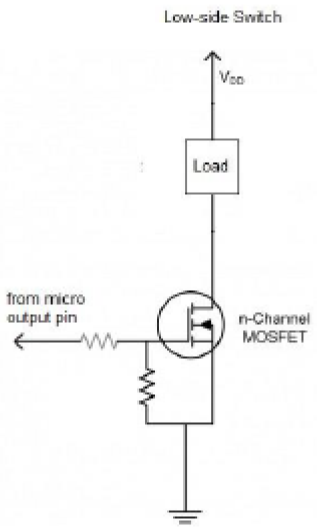


Fig 4. Low Side Driver Evaluation

The load of bulb with rating 24V, 10W is connected to the drain of the LSD and the Microcontroller pin tends to drive the load, the voltage at the load is 24V(VDD) and the other terminal of the load at drain of LSD. The shift register output (5V) drives the gate of the LSD, which intern drives the switching action of LSD. With the timer support the LSD can be driven for the PWM functionality with the varying Duty Cycle (DC).

IV. CONCLUSION

The evaluation of the subsystems of the I/O ECU and its communication interface such as Serial Peripheral Interface (SPI) and the CAN communication makes it possible to communicate between the I/O ECU and the High End ECU exchanging data over a communication channel.

REFERENCES

[1] Peter Waszecki , Philipp Mundhenk , Sebastian Steinhorst , Martin Lukasiewycz ,Ramesh Karri," Automotive Electrical and Electronic Architecture Security via Distributed In-Vehicle Traffic Monitoring ", 2017, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.

[2] C. Lyu, D. Gu, Y. Zeng, and P. Mohapatra, "PBA: Prediction-based authentication for vehicle-to-vehicle communications," IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 1, pp. 71–83, 2015.

[3] C.-W. Lin, Q. Zhu, and A. Sangiovanni-Vincentelli, "Security-aware modeling and efficient mapping for CAN-based real-time distributed automotive systems," Embedded Systems Letters, IEEE, vol. 7, no. 1, pp. 11–14, 2015

[4] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive CAN networks–practical examples and selected short-term countermeasures," in Computer Safety,Reliability, and Security, pp. 235–248, Springer, 2008.

[5] S. Chakraborty, S. Kunzli, and L. Thiele, "A general framework for analyzing system properties in platform-based embedded system designs," in Proceedings of the Conference on Design, Automation and Test in Europe (DATE), pp. 190–195, 2003.