Illini Formula Electric
Display Software Specification
Andrew Smith

2018 Display Design Specification

Purpose

The purpose of the display is to display important vehicle statistics in an easy to read and elegant fashion. This will aid the driver in competition and help them drive consistently and quickly. The driver will be able to gauge how quickly they are traveling and the remaining battery level as well as see the fault status, and traction control status.

System Overview

The display system is a fairly complex system that will be built on top of the data acquisition network. The display system has multiple parts, the physical display and display controller, a microcontroller to act as the brain, and a CAN interface module (Figure 1).

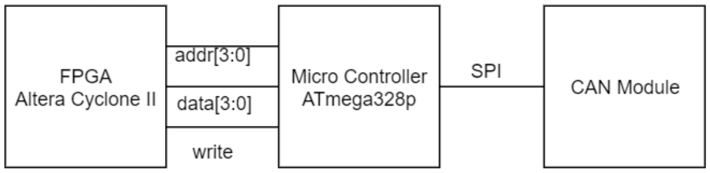

Figure 1: High Level View

Hardware

The physical display that I will be using is a 5.0 inch TFT digital display with 8-bit color resolution. It will be controlled by a small Altera Cyclone II FPGA because the microcontroller does not have the capability to write the 8-bit color to the display. Colors will be palatized to conserve memory on the FPGA. Figure 2 is a concept of the GUI where the orange bars will grow or shrink based on the speed or battery level. The GUI was designed to mimic our IFE logo and still be easy to read at a glance while driving.

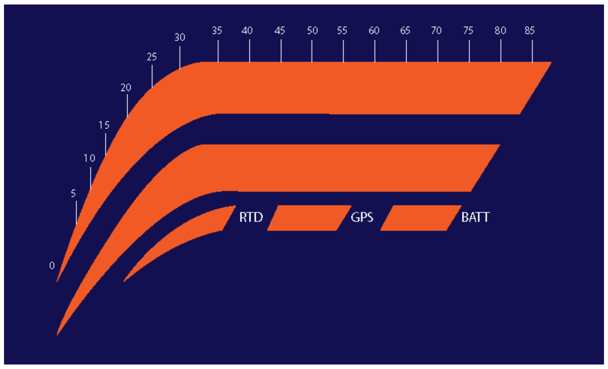

Figure 2: Concept of GUI

The micro controller is the main device on display board. It will interface with the data acquisition network through the CAN module, and tell the FPGA how much of the status bars to display via addressable registers. The microcontroller will be listening on the CAN network for the message IDs that contain the data to be displayed, scaling the data and writing to the FPGA's registers. I plan on using an ATmega328p for its small footprint and its low power draw.

The CAN module is a small system with two ICs that add an easy to use serial can interface to devices. We will be using a Microchip MCP 2515 CAN Controller with a Microchip MCP 2551 CAN transceiver to control and receive messages from the can network. The microcontroller will receive message data via SPI.

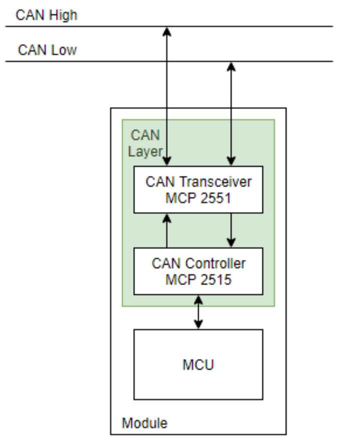

Figure 3: CAN Interface system overview

The system will also have some simple power circuitry with a low pass filter to filter out high frequency noise in the supply voltage, and some voltage regulators to step the power down to 5V, 3.3V, and 1.8V.

Software

This project is going to require a lot of software development. I will need to write a device driver to interface the microcontroller with the CAN system as well as message handler software. I will also have to write a driver so that microcontroller can interface with the FPGA and write to its registers. I will also have to write quite a bit of System Verilog so that the FPGA can draw to the display properly.

Mechanical

I will work with the mechanical captain to design a simple 3D printed housing to contain the hardware as well as make it easy to mount to the dash of the vehicle.

Manufacturing

Currently I am designing the schematic in EAGLE and will be laying out a custom PCB to keep the sizing of the device small, as well as make the device easy to integrate onto the dash of the vehicle. The layout will be sent to a PCB manufacturer to print the board and I will assemble the components onto the board.

Testing

This will be by far the most daunting part but I intend to develop a test bench to be able to send dummy CAN data to the device to simulate it being on the car. Luckily there is not much analog circuitry on the board so most of the testing will be software testing and debugging. I have included LEDs to be able to toggle when debugging.

Current State of Design

Currently I have just figured out the architecture of the sensor network and I have started the schematic of the display board. I have completed the programmer circuitry and configuration loader circuitry for the FPGA as well as the programmer circuitry for the microcontroller. I still have to add the display connector and the power circuitry, as well as finish adding connecting the CAN module.

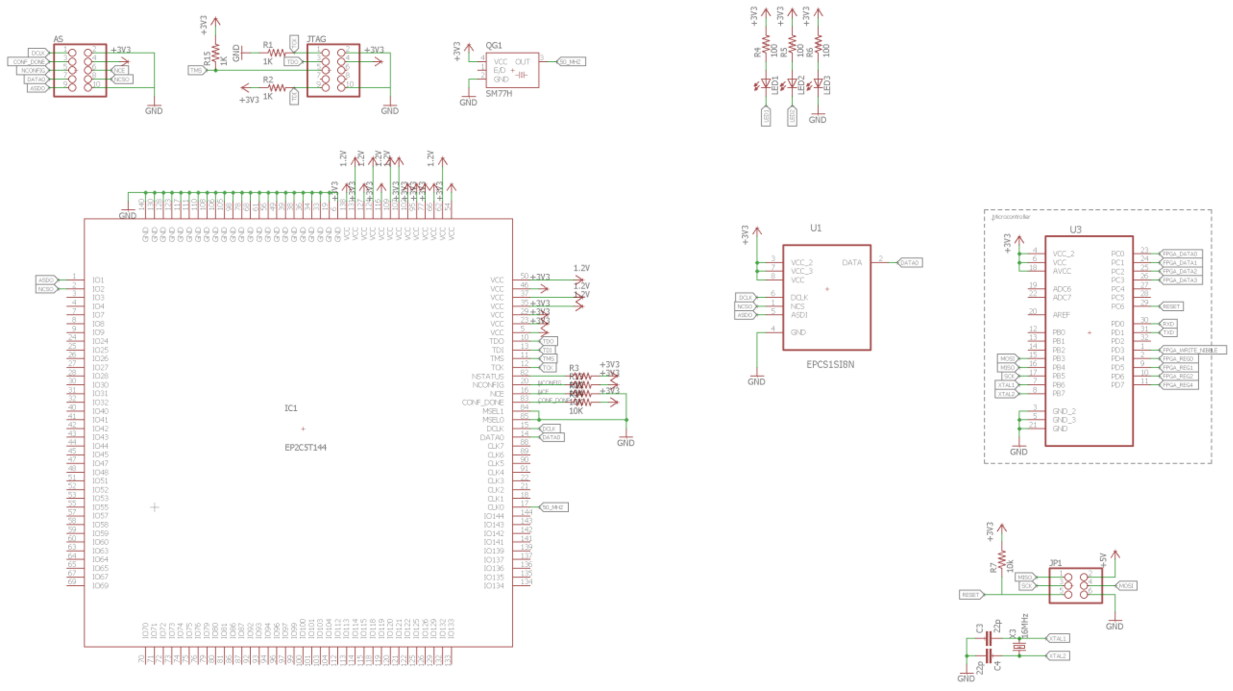

Figure 4: Current design state