# YAZAKI Final Report

## Semester i: Automotive Embedded Technology

November 30th 2018

Pedro Antonio Gámez

A00369538

Sharif Nasser Kadamani

A00820367

Digital System and Robotic Engineers

**Professors:**

Dr. Alfonso Ávila

Dr. Graciano Dieck

Mtra. Jakeline Marcos

Mtro. Artemio Aguilar

Dr. Raúl Peña

Dr. Eleazar Reyes

**Engineers:**

Eng. Luis Saracho

Eng. Isaías Acosta

Eng. Javier Baca

Eng. Edgar Ríos

Eng. Fernando Villarreal

Eng. Alejandro Quijas

Eng. Romy Bompart

Eng. Rogelio Espinoza

Eng. Daniel Velásquez

Eng. Jesús López

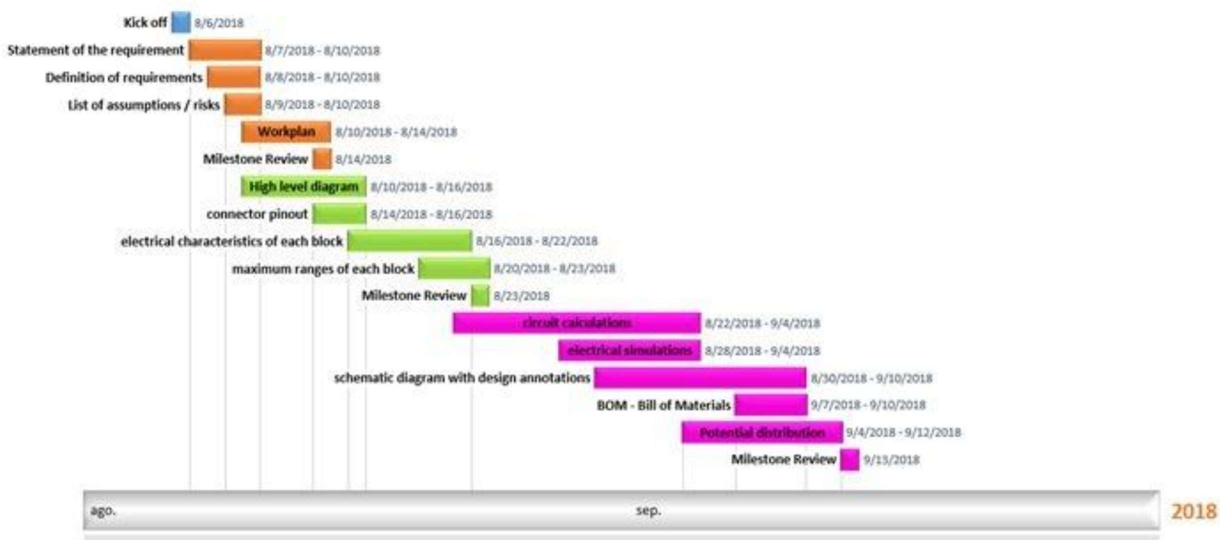
Instituto Tecnológico y de Estudios Superiores de Monterrey

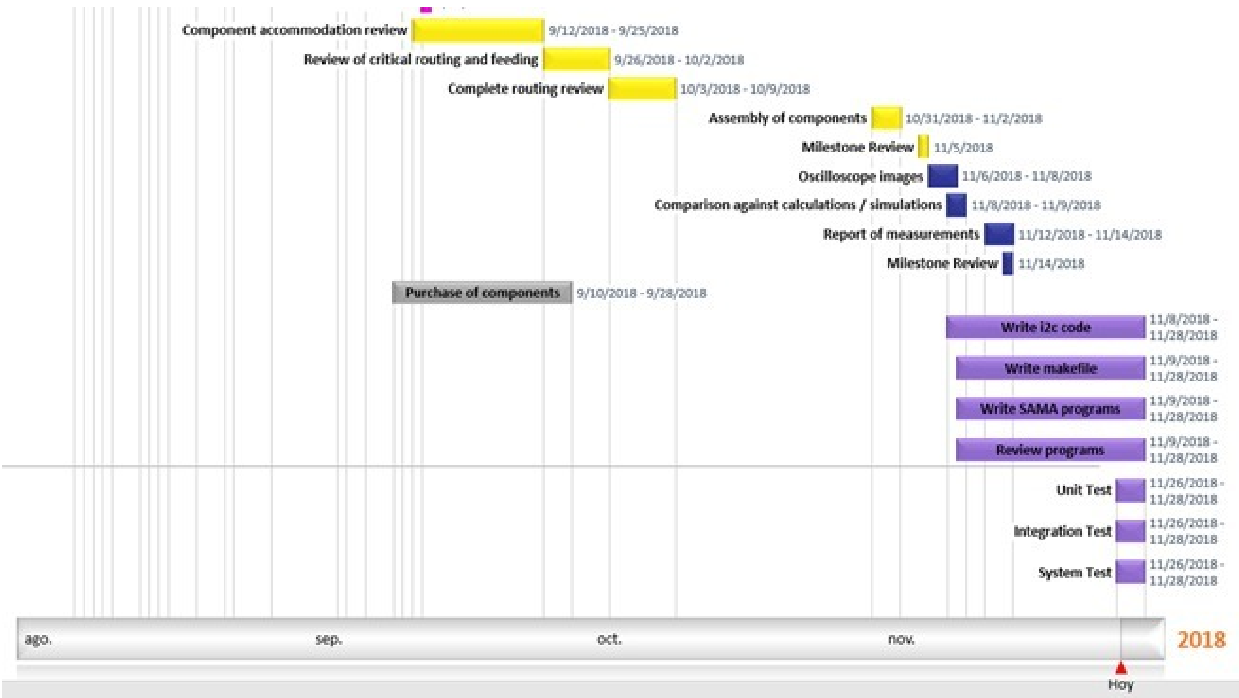
MONTERREY, NUEVO LEON

# Table of Contents

# Project Managment

## Project Schedule



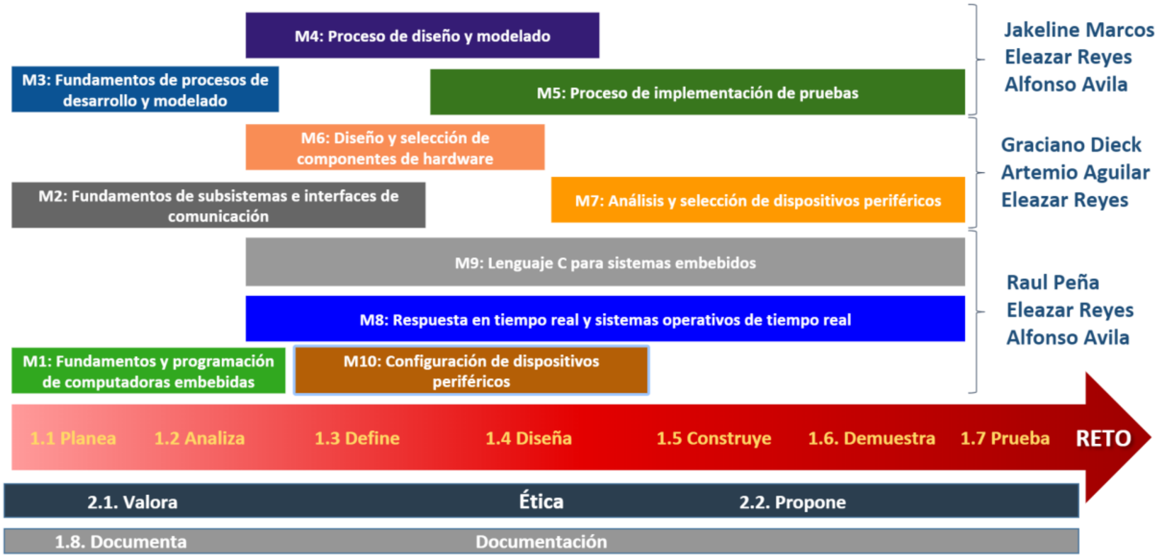



## Software and Hardware Project Schedule (YAZAKI)

| | ACTIVITY | PLAN START | Plan Duration |
|---|---|---|---|
| | Kick off | 1 | 1 |
| **Análisis de requerimientos** | Enunciado del requerimiento | 2 | 4 |
| | Definición de requisitos | 3 | 3 |
| | lista de suposiciones/riesgos | 4 | 2 |
| | Plan de trabajo | 5 | 2 |
| **Arquitectura de hardware** | Diagrama de alto nivel | 5 | 4 |
| | pinout de conectores | 7 | 2 |
| | características eléctricas de cada bloque | 9 | 4 |
| | rangos máximos de cada bloque | 11 | 3 |

| | | | |
|---|---|---|---|
| **Diseño electrónico** | cálculos de circuitos | 13 | 10 |
| | simulaciones eléctricas | 17 | 6 |
| | diagrama esquemático con anotaciones de diseño | 19 | 8 |
| | BOM - Bill of Materials | 25 | 2 |
| | Distribución de potencia | 22 | 7 |
| **Diseño de Tablero** | Revisión de acomodo de componentes | 28 | 10 |
| | Revisión de ruteo crítico & de alimentación | 38 | 5 |
| | Revisión de ruteo completo | 43 | 5 |
| | Ensamble de componentes | 63 | 3 |
| **Verificación de circuitos** | Imágenes de osciloscopio | 67 | 2 |
| | Comparación contra cálculos/simulaciones | 69 | 2 |
| | Reporte de mediciones | 71 | 2 |
| **Actividades de espera** | Compra de componentes | 26 | 15 |
| | **Fabricación de tablero** | 48 | 15 |
| **Creación de Código** | Escribir código Makefile | 48 | 7 |
| | Escribir código I2c | 42 | 20 |
| | Escribir código SPI | 42 | 20 |
| | Realizar diseño en StoryBoard Suite | 68 | 10 |

## Schedule based on Semestre i modules




## RACI Chart

| | | ROBODUO | | YAZAKI | TEC |
|---|---|---|---|---|---|
| | **ACTIVITY** | **Pedro Antonio Gámez** | **Sharif Nasser** | **Engineers** | **Professors** |
| | Kick off | | | | |
| **Análisis de requerimientos** | Enunciado del requerimiento | R | R | I | A |
| | Definición de requisitos | R | R | I | A |
| | lista de suposiciones/riesgos | R | R | I | A |
| | Plan de trabajo | R | R | I | A |
| **Arquitectura de hardware** | Diagrama de alto nivel | R | C | I | A |

| | | | | | |
|---|---|---|---|---|---|
| | pinout de conectores | R | R | I | A |
| | características eléctricas de cada bloque | R | R | I | A |
| | rangos máximos de cada bloque | C | R | I | A |
| **Diseño electrónico** | cálculos de circuitos | R | C | I | I |
| | simulaciones eléctricas | R | C | I | I |
| | diagrama esquemático con anotaciones de diseño | R | C | I | I |
| | BOM - Bill of Materials | R | C | I | I |
| | Distribución de potencia | R | C | I | I |
| **Diseño de Tablero** | Revisión de acomodo de componentes | R | C | I | I |
| | Revisión de ruteo crítico & de alimentación | R | C | I | I |
| | Revisión de ruteo completo | R | C | I | I |
| | Ensamble de componentes | A | A | R | I |
| **Verificación de circuitos** | Imágenes de osciloscopio | R | R | I | I |
| | Comparación contra cálculos/simulaciones | R | C | I | I |

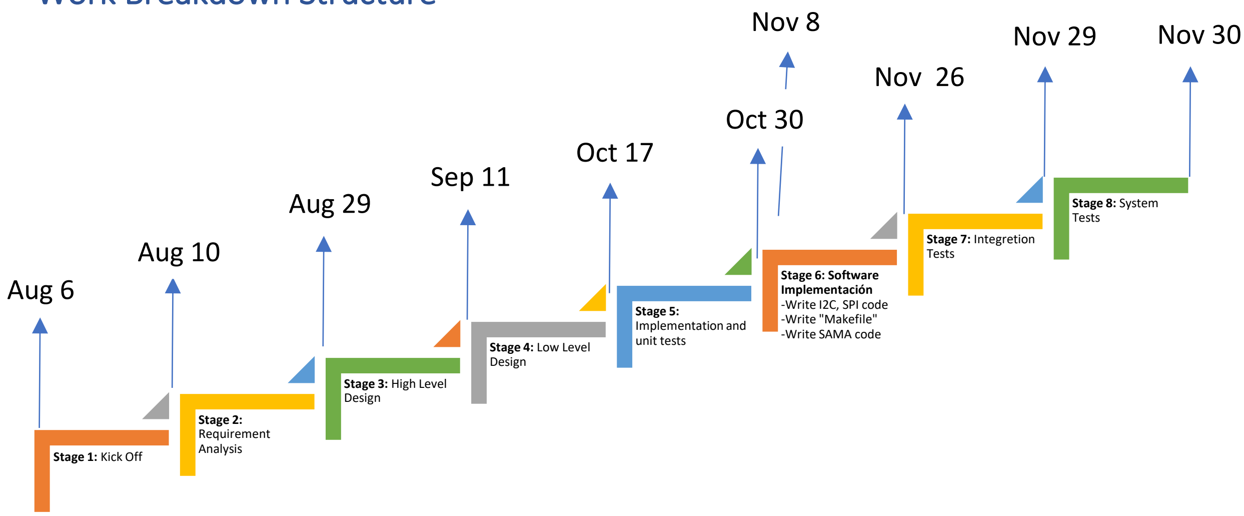
| | | | | | |
|---|---|---|---|---|---|
| | Reporte de mediciones | R | C | I | I |
| **Actividades de espera** | Compra de componentes | A | A | I | R |
| | **Fabricación de tablero** | I | I | A | R |
| **Creación de Código** | Escribir código Makefile | C | R | I | I |
| | Escribir código I2c | C | R | I | I |
| | Escribir código SPI | C | R | I | I |
| | Realizar diseño en StoryBoard Suite | C | R | I | I |

R = Responsible, A = Accountable, C = Consulted, I = Informed

| Actividad | Cluster Team | | Yazaki Team | |
|---|---|---|---|---|
| | Pedro Gámez | Sharif Nasser | Ingenieros | Estatus |
| Diagrama de Actividad Display Graphics | C | R | I | Approved |
| Diagrama de Actividad Power on | C | R | I | Approved |

| | | | | |
|---|---|---|---|---|
| Diagrama de Actividad Adjust Backlight | R | C | I | Approved |
| Diagrama de Casos de Uso | C | R | I | Approved |
| Deployment Diagram | C | R | I | Approved |
| High level Diagram | R | C | I | Approved |
| QnA | R | R | I | Approved |
| SRS | R | R | I | Approved |
| Electrical Requirement format | C | R | I | Approved |
| Simulaciones | R | C | I | Approved |
| BOM | R | C | I | Approved |
| XDxDesigner Schematic | R | C | I | Approved |
| Cluster Storyboard | C | R | I | Approved |
| xPCB Layout | R | C | I | Approved |
| Software | C | R | I | Approved |
| Tests | R | R | I | Approved |

## Work Breakdown Structure




## Project Charter

**Title:** Kick off stage for the project "Embedded Automotive Technology"; Familiarization with concepts, objectives and requirements.

**Date:** 10/08/18

**Team Name:** ROBODUO

**Team Members:** Pedro Antonio Gámez, Sharif Nasser

**Objective**

That the students know the project, become familiar with the concepts, objectives and requirements as well as generate a work plan and lists of assumptions / risks for it.

**Scope**

**Problem Statement**

- Start Project; requirements analysis.

**What are we going to do?**

1. Kick off and requirement analysis

1.1 Plan the solution's development

1.2 Analyze the device requirements

1.3 Know the useful interfaces of both hardware and software for the development of the project

1.4 List the assumptions /risks of the project

1.5 Design a schematic of how we assume that the embedded system will be built

1.5.1 Know and do research about the individual components of both software and hardware

2. Software architecture

2.1 Design high- and low-level diagrams

2.2 Pinouts characteristics

3. Electronic design

3.1 Calculations of the values needed to build the embedded system

4. PCB design and tests

4.1 Components assembly

5. Circuit performance and integrations tests

6. Project delivery

**What are we not going to do?**

1. PCB's manufacturing

2. Components purchase

**In what will Yazaki help us?**

1. The engineers will advise us during all stages of the project development; they'll assist us by giving us by sharing their knowledge with us and giving us the right tools so that we can develop the project properly

**People involved**

- Professors
- Students
- Yazaki's teamwork

**Risks**

- Getting defective components
- Having mistakes in the schematic design
- Having mistaken at the PCB design
- Having issues between teammates
- Having flaws in programming
- Bad teamwork
- Bad engineers-professors-students communication

**Responsible signatures:**

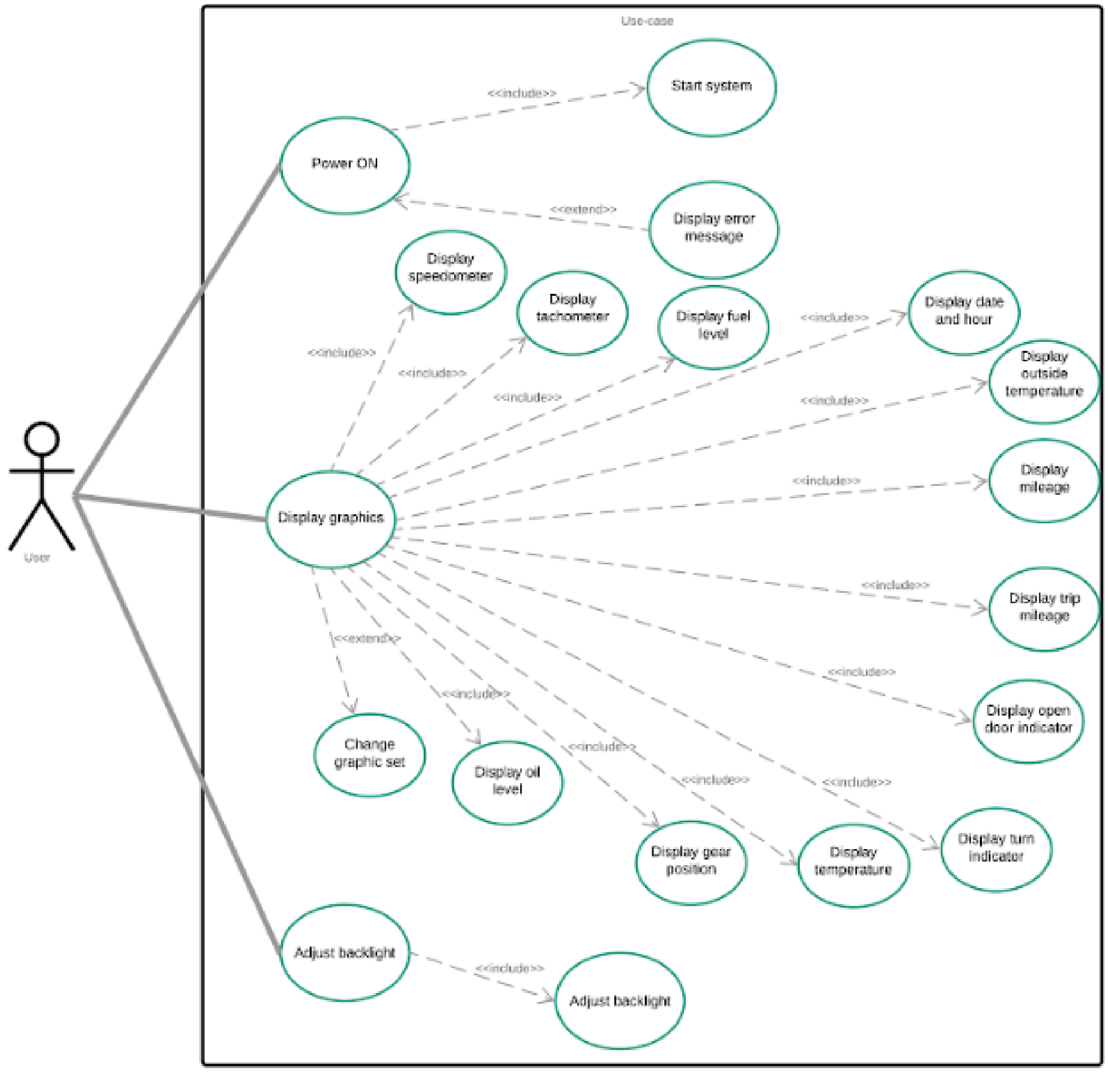
| | | |
|---|---|---|
| Project Manager | YAZAKI | ITESM |

## Requirement Specification

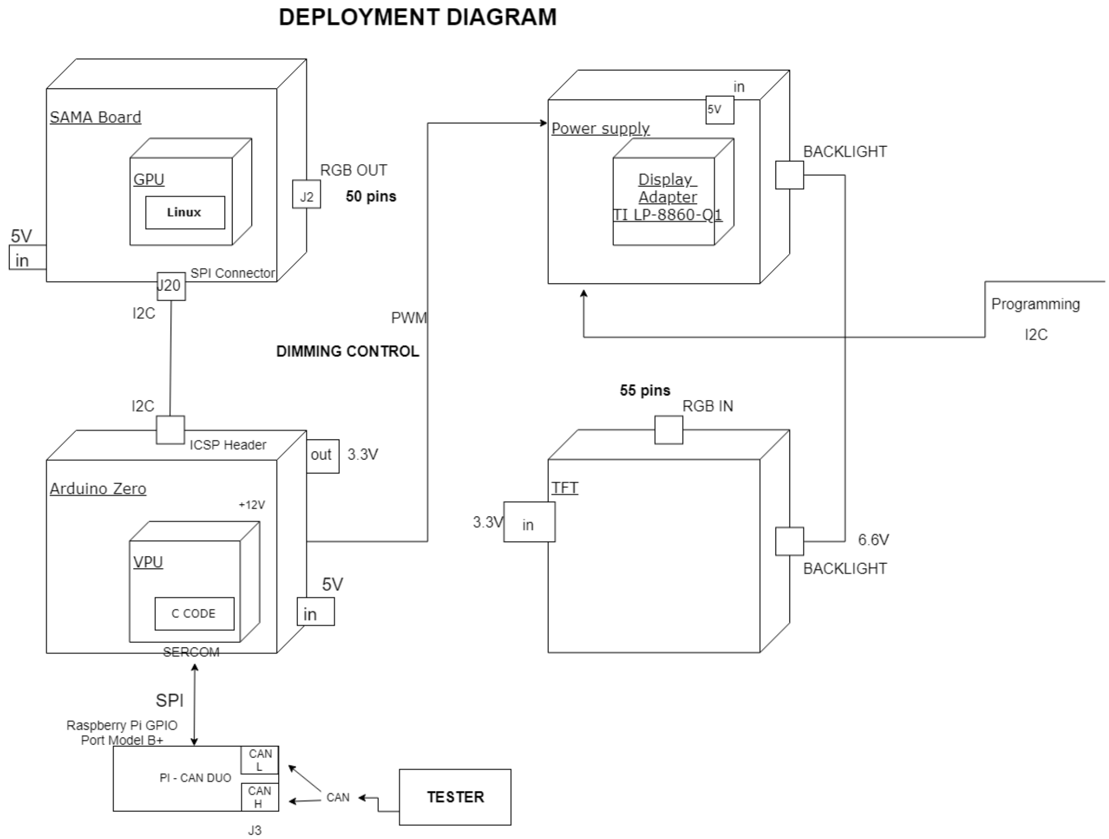
Check Appendix A (System Requirements Specification)

## High-level design

## Use Case Diagram

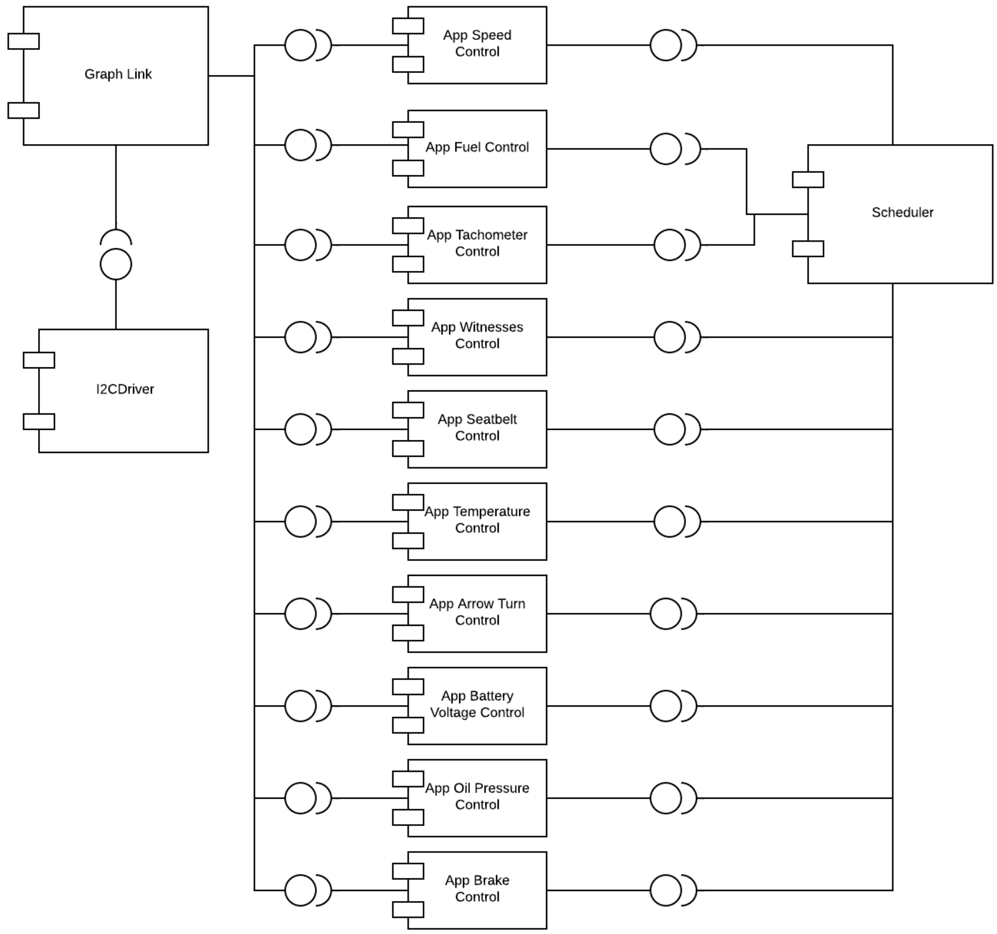


## Deployment Diagram

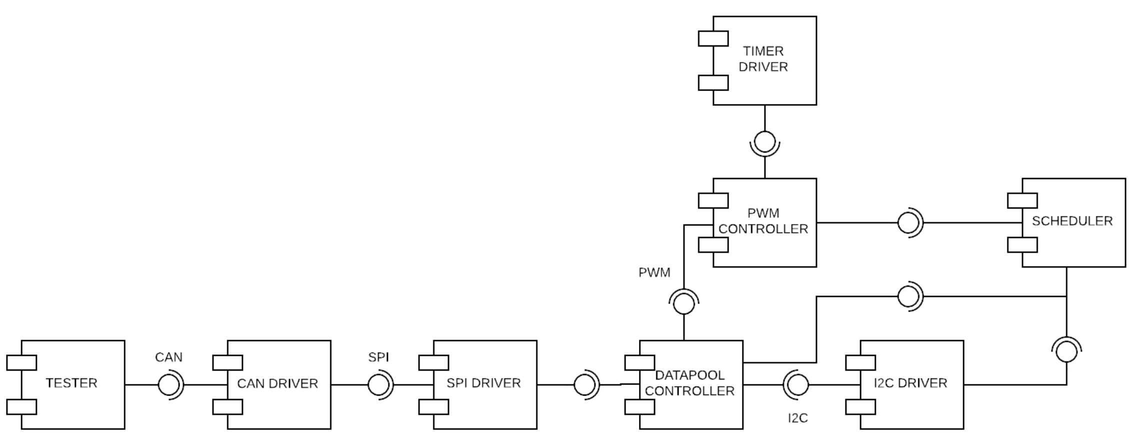


## Component Diagram

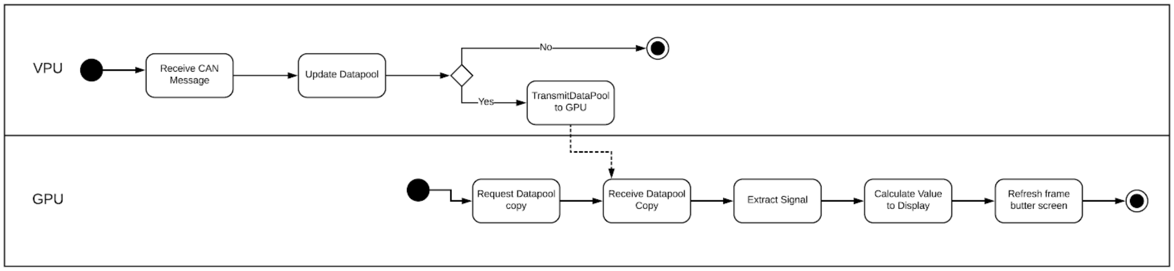
In GPU:



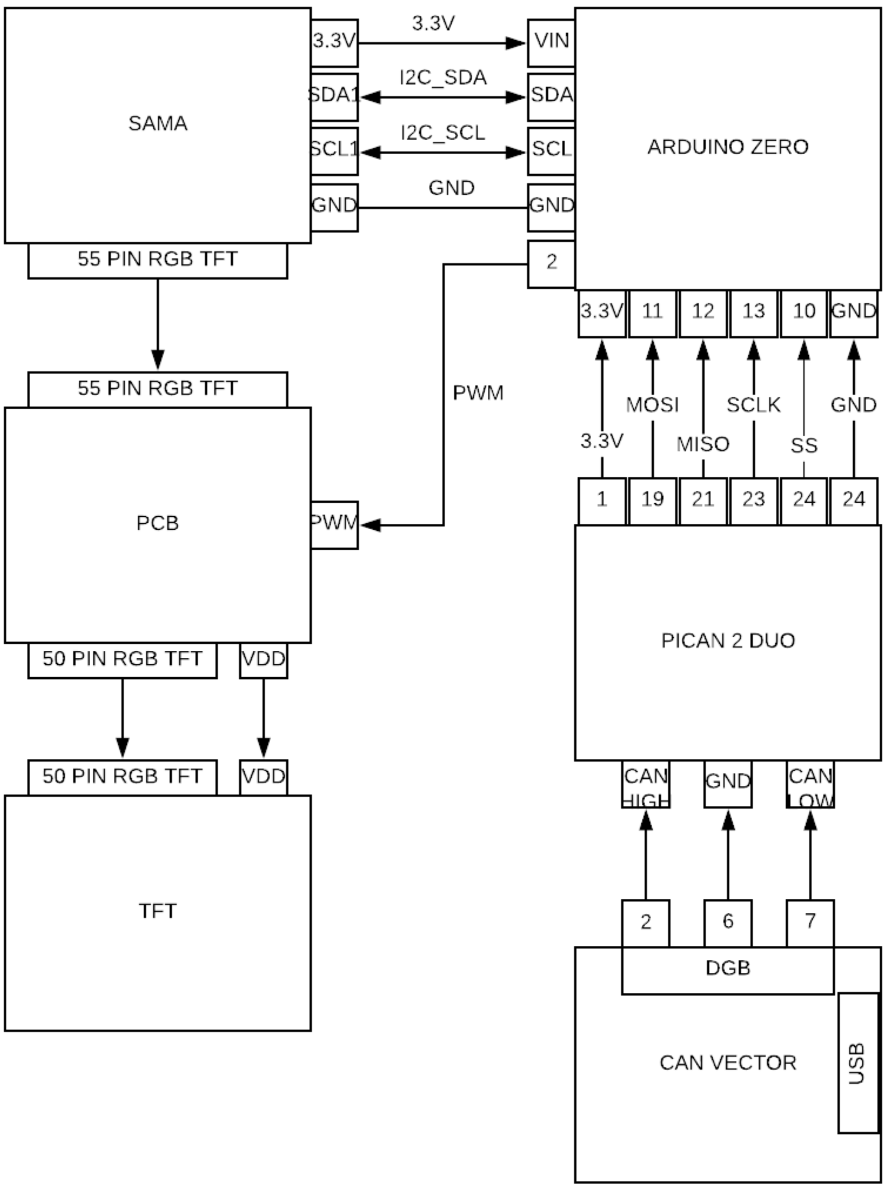

In VPU:



## Activity Diagrams

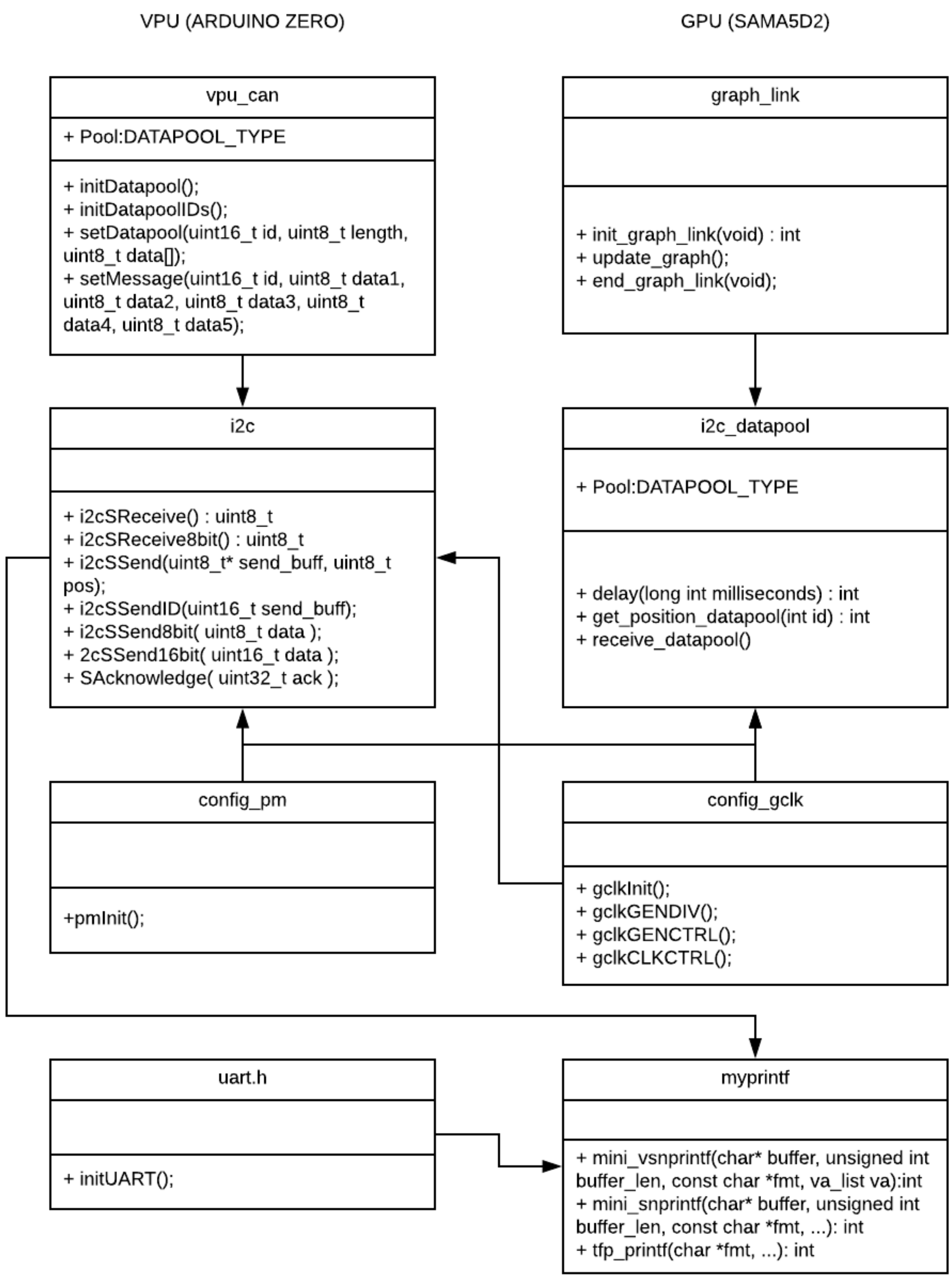



## Microcontroller/Component PIN assignments

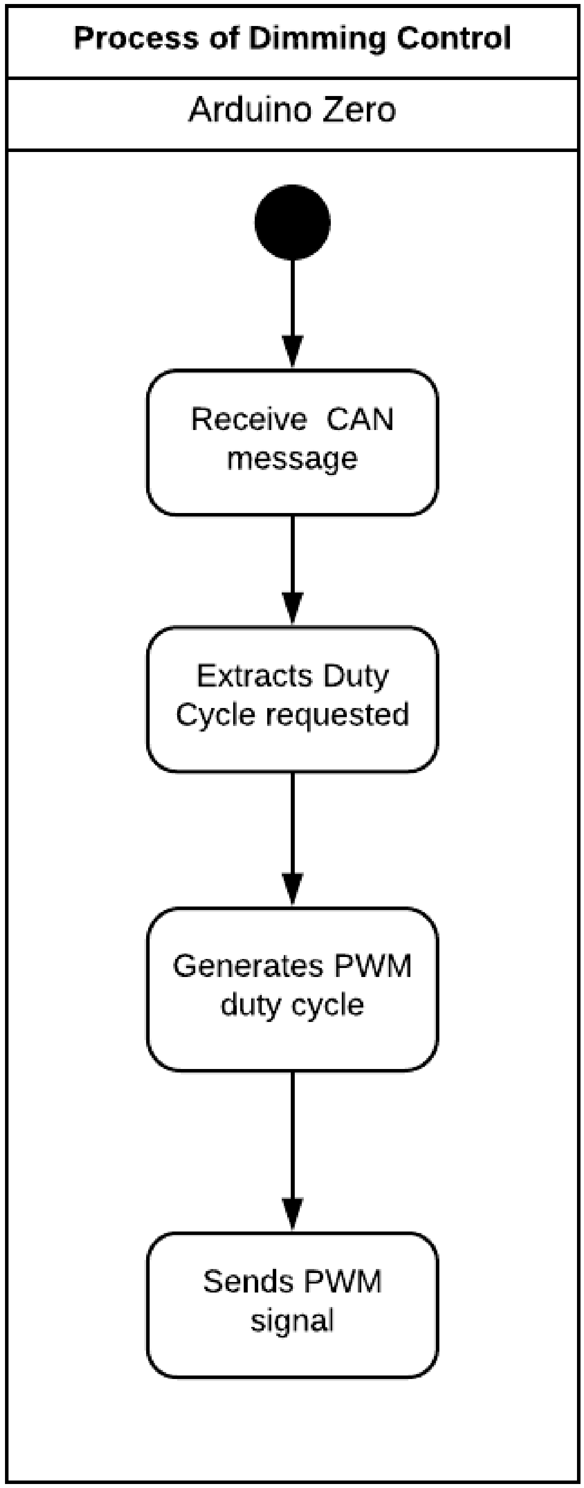


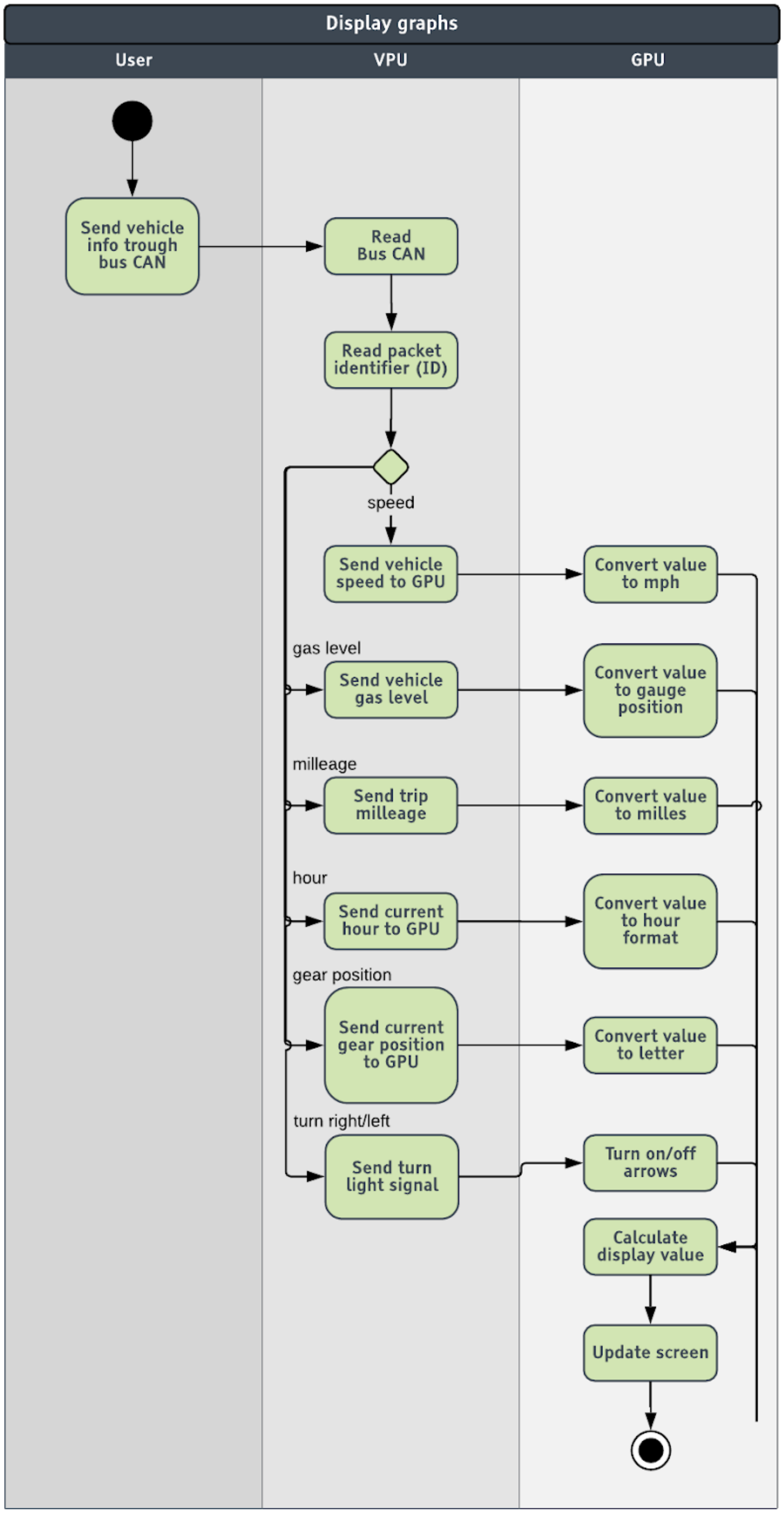
# Low-level design

## Class diagrams

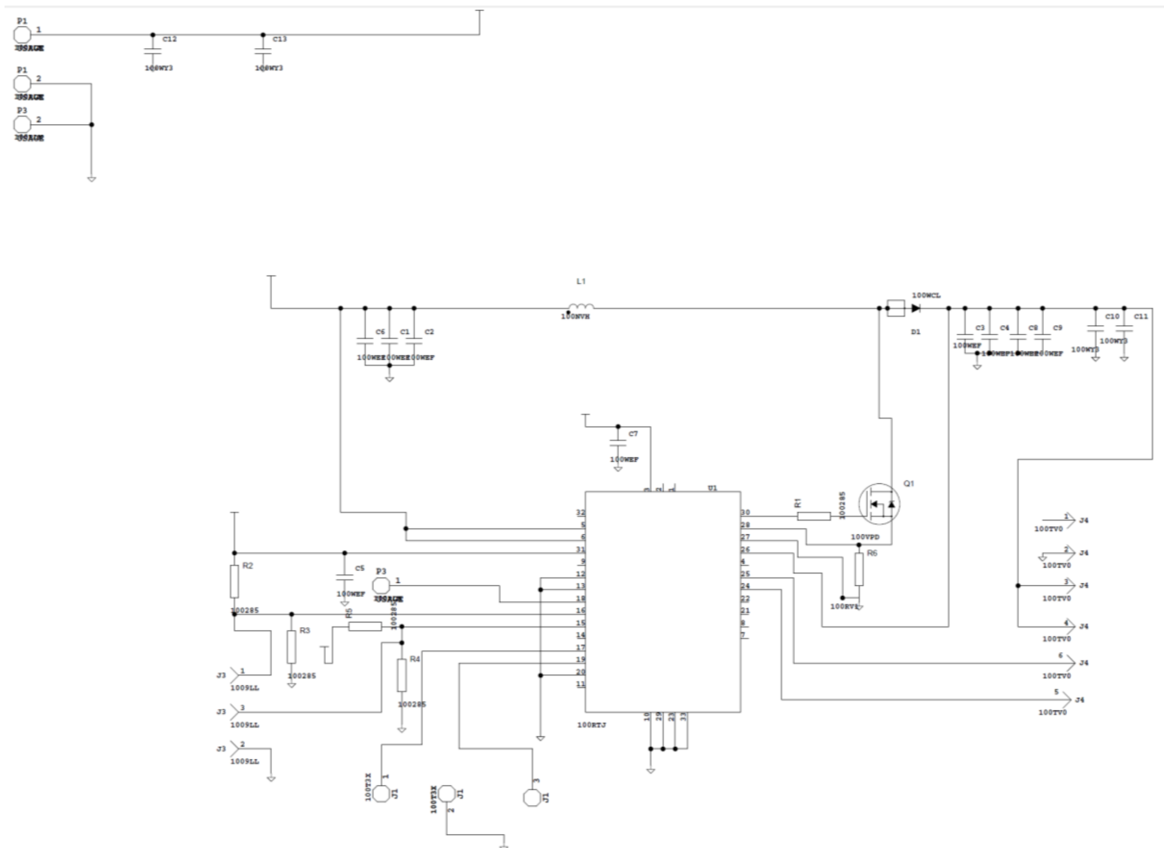


## Activity diagrams (per method)

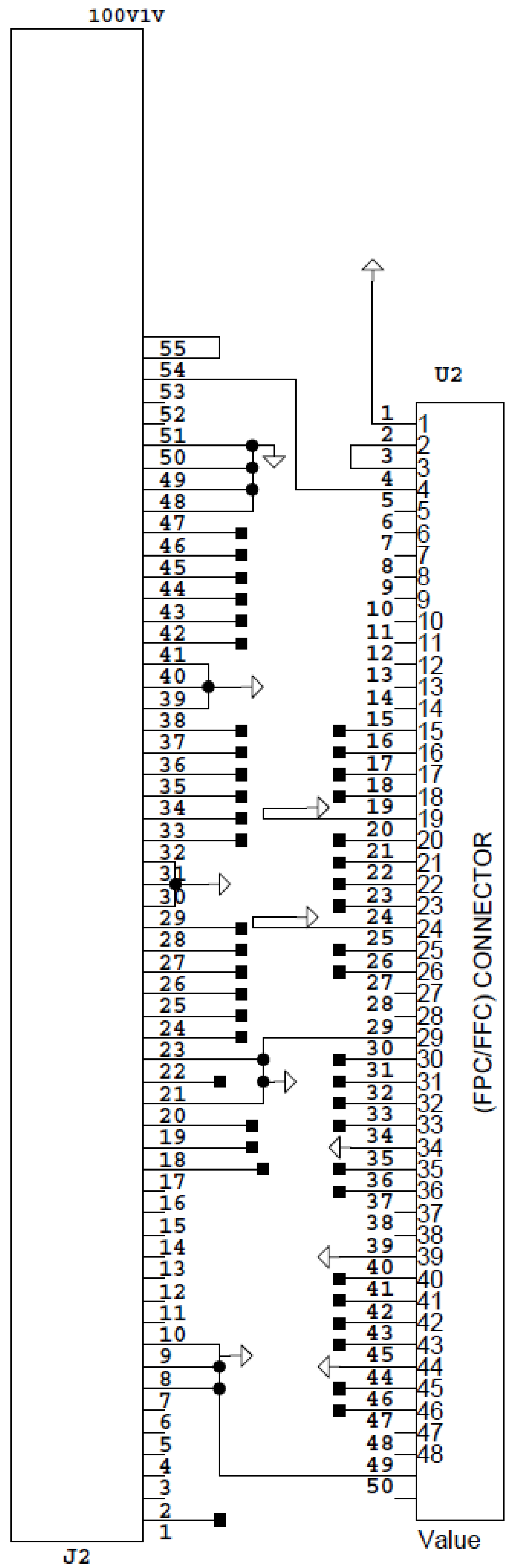



Display graphs
User
VPU
GPU
Send vehicle info trough bus CAN
Read Bus CAN
Read packet identifier (ID)
speed
Send vehicle speed to GPU
Convert value to mph
gas level
Send vehicle gas level
Convert value to gauge position
milleage
Send trip milleage
Convert value to milles
hour
Send current hour to GPU
Convert value to hour format
gear position
Send current gear position to GPU
Convert value to letter
turn right/left
Send turn light signal
Turn on/off arrows
Calculate display value
Update screen

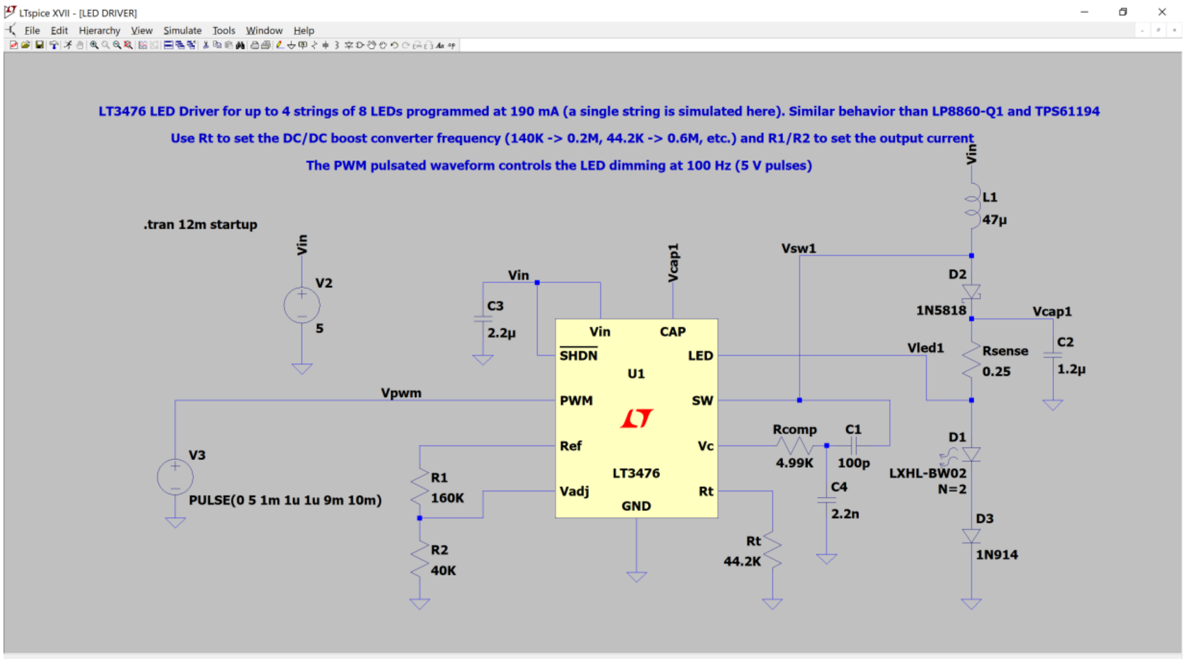
# PCB schematic/ simulations

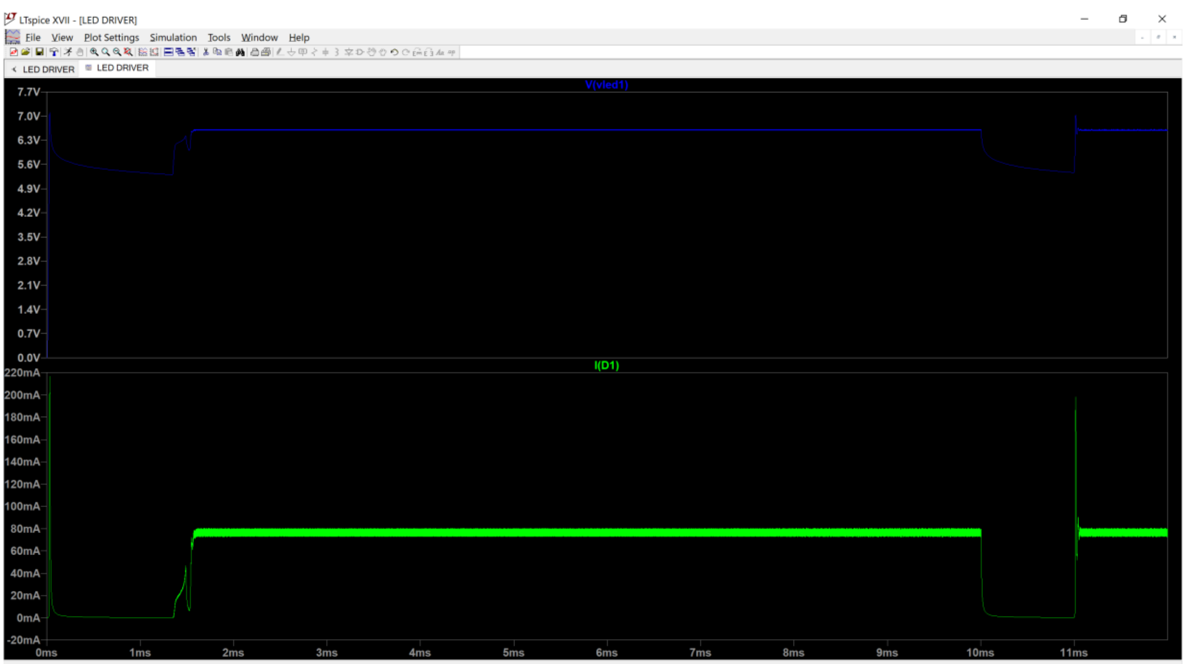


Connections between 50-55 pin connectors

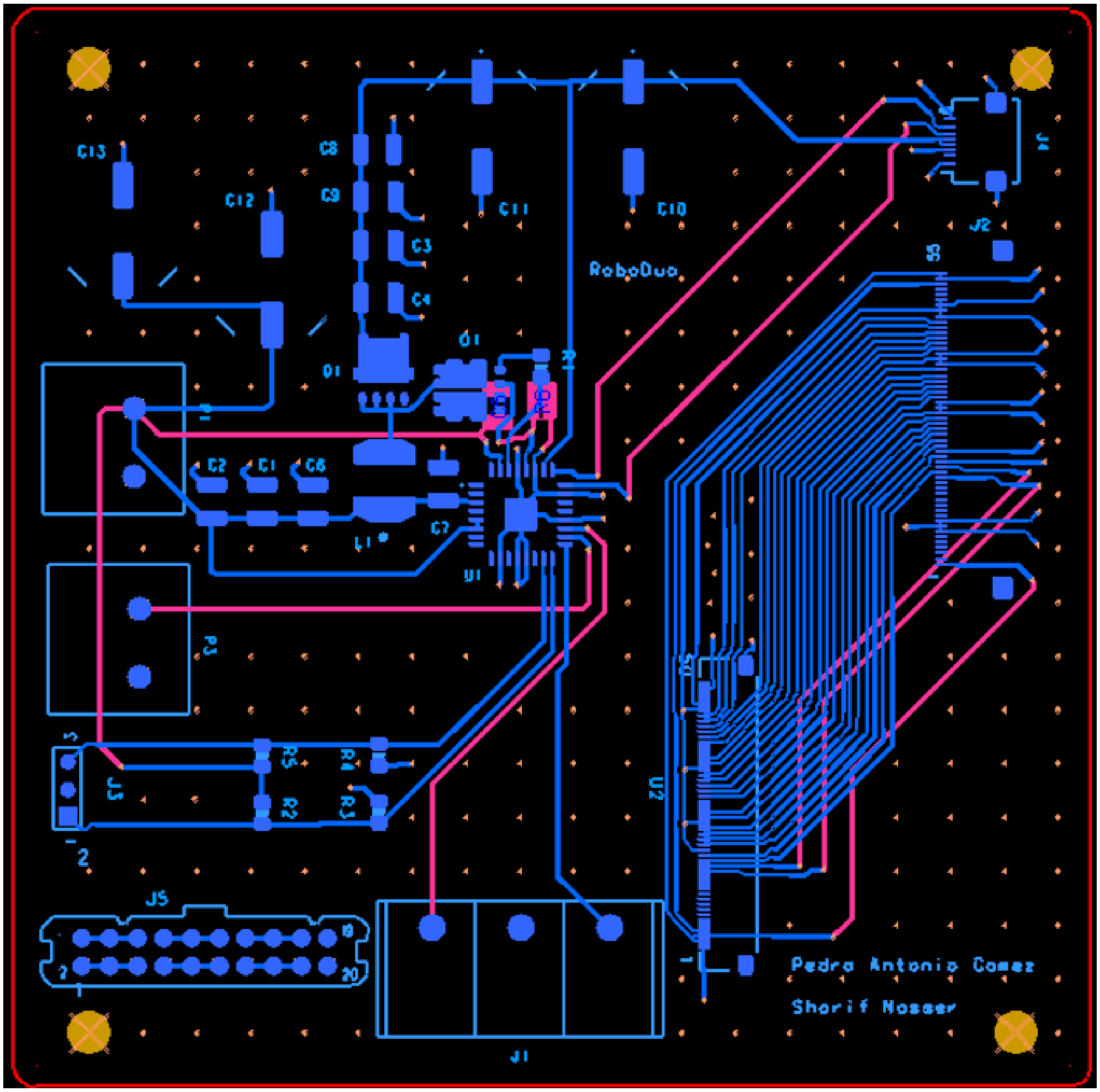


Simulation were not made for the LP8860 Q1, instead we simulated the LT3476 another LED driver, which we then learned was nothing like the driver that we were to work with.

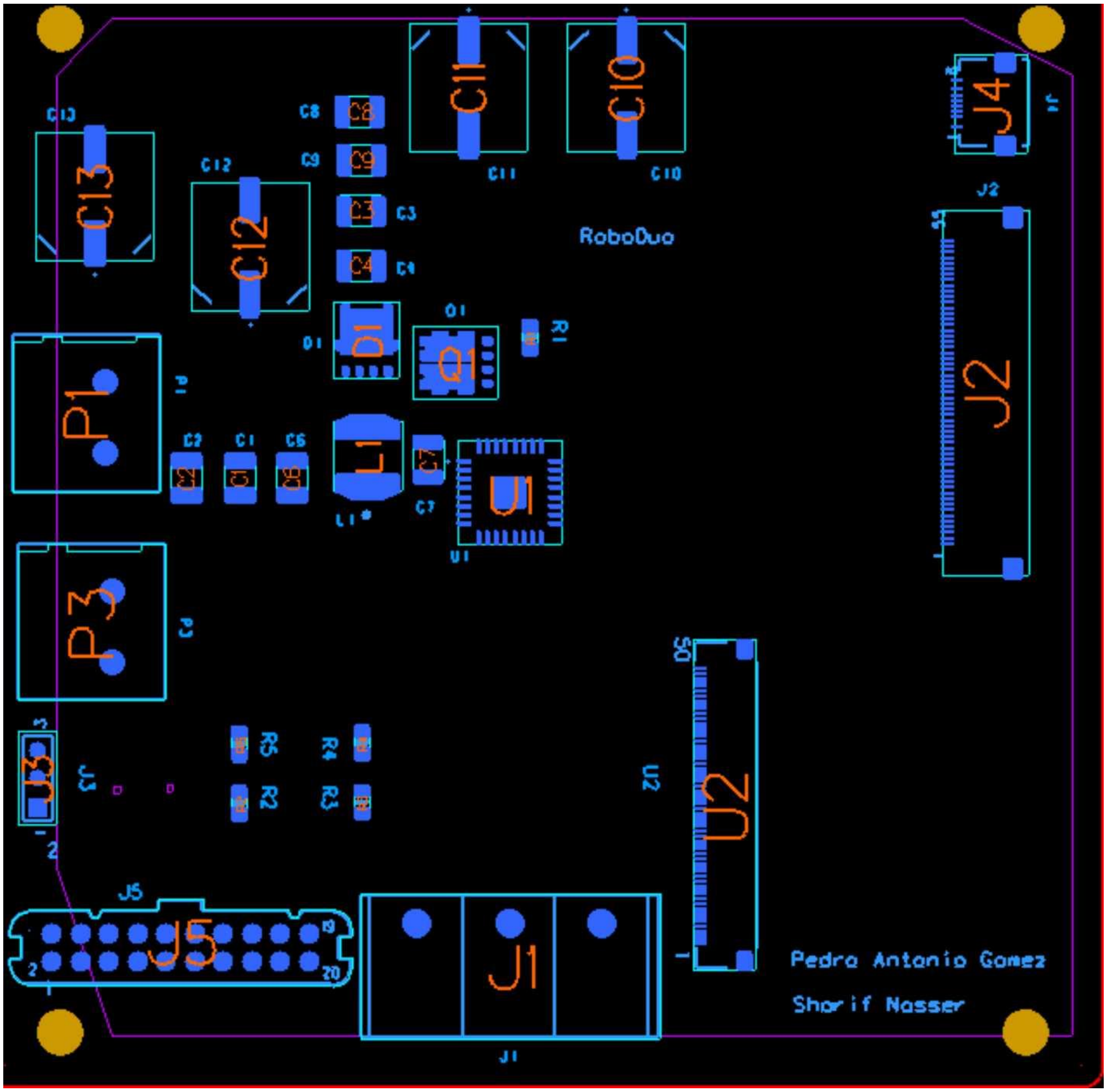


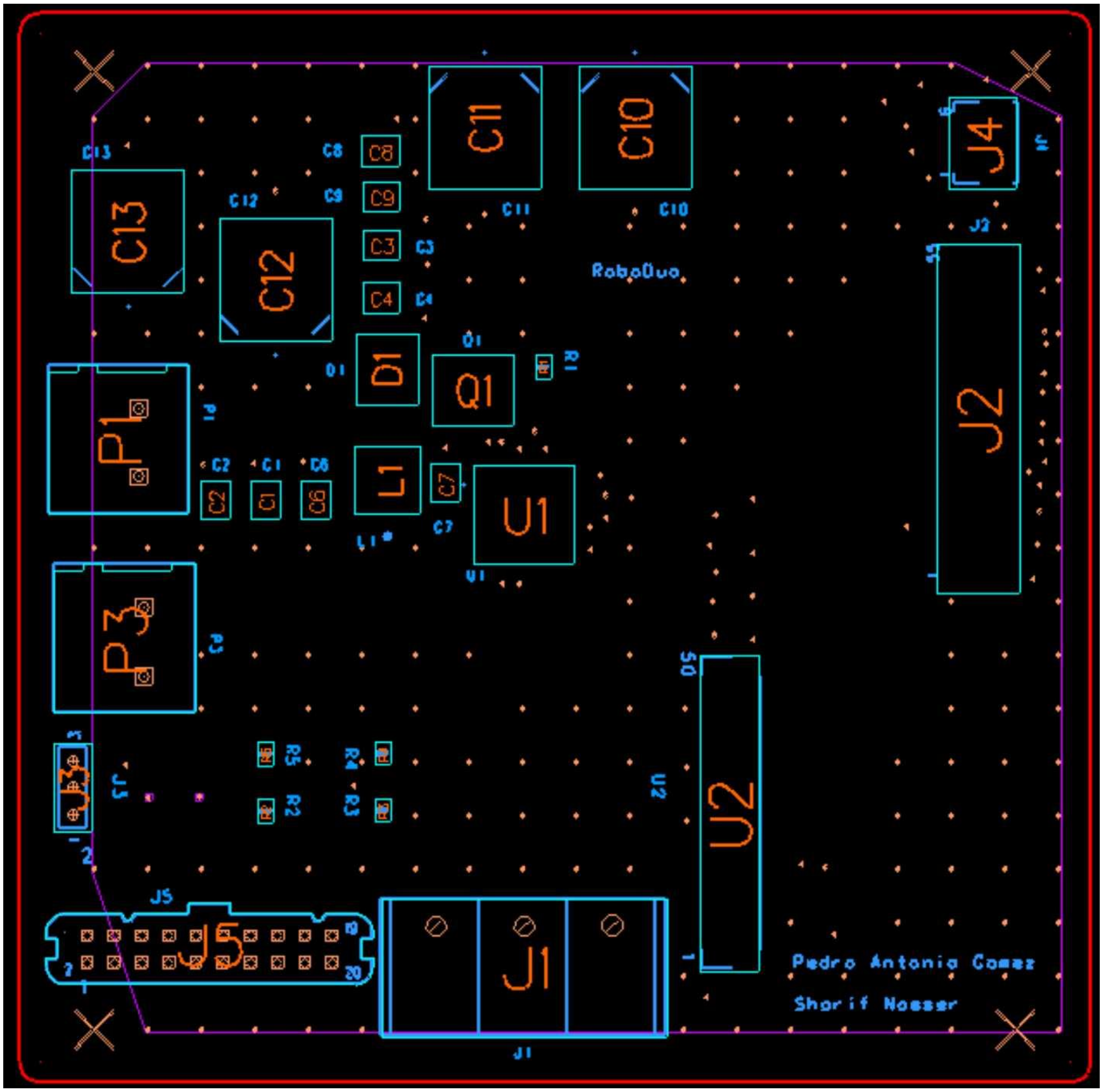



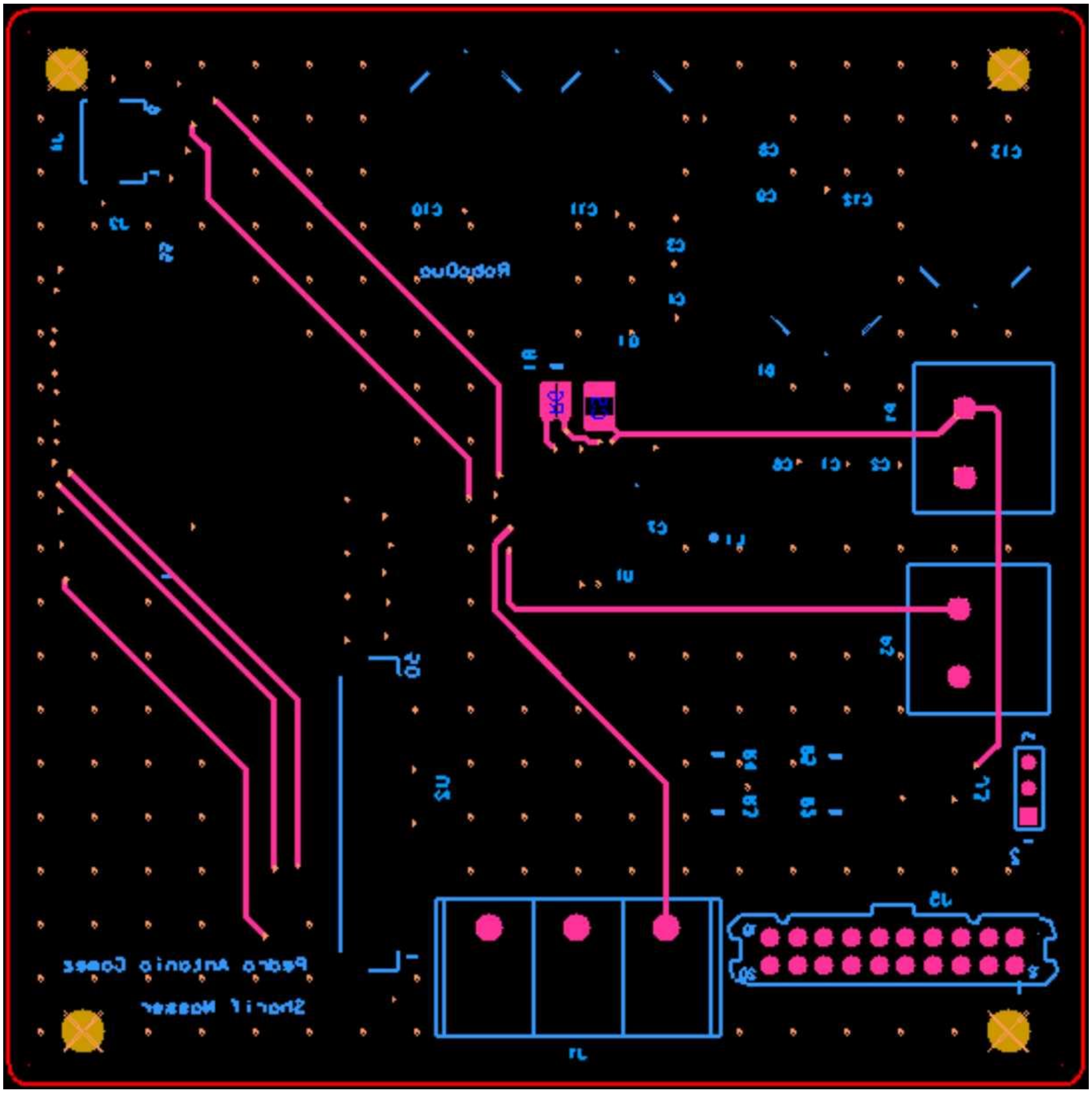
## PCB layout



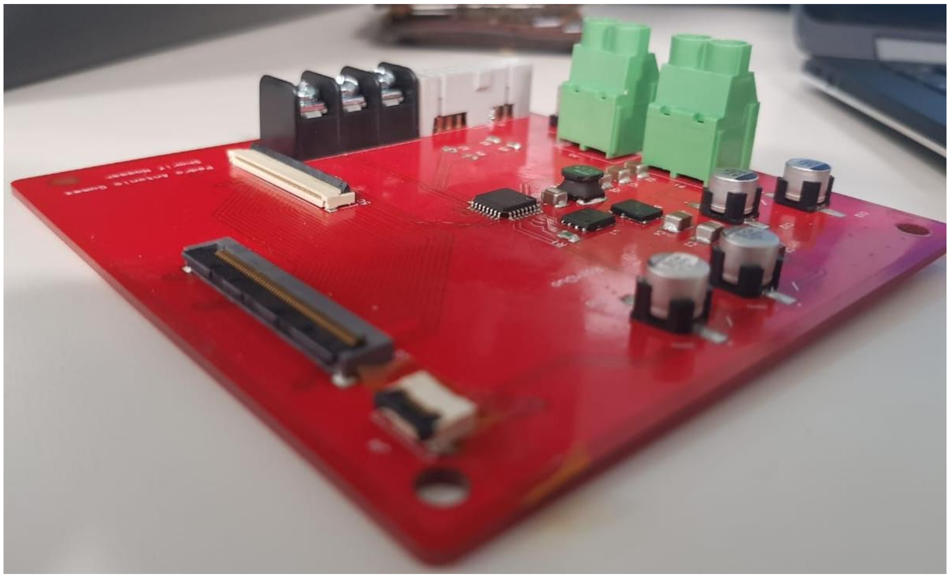
C13
C12
C8
C9
C3
C4
C11
C10
RoboDuo
J4
J2
D1
Q1
P1
C2
C1
C6
C7
L1
U1
P3
U2
J3
R5
R4
R2
R3
J5
J1
Pedro Antonio Gomez
Sharif Nasser

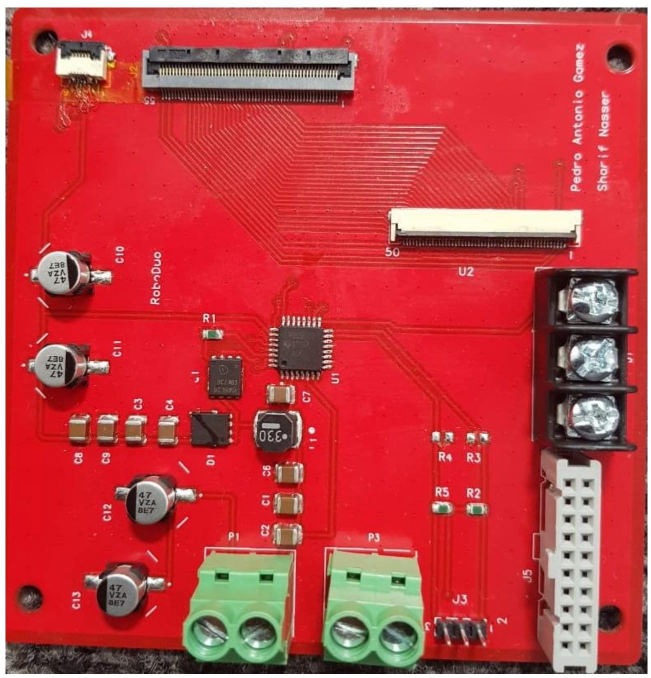

RoboDuo
Pedro Antonio Gomez
Sharif Nasser


C11
C10
C13
C12
C8
C9
C3
C4
D1
Q1
P1
L1
C2
C1
C6
C7
U1
J4
J2
P3
J3
U2
J5
J1
RoboDuo



## Manufactured PCB





# Description of System tasks and timeline schedule

The system is built by a PICAN 2 DUO, an Arduino Zero, a SAMA5D2, an electronic circuit that interfaces the RGB connectors between the SAMA and the TFT, which is the last element of it. This fundamentally works with the communication between the Vehicle Processor Unit, which is an Arduino Zero, and a Graphic Processor Unit, a SAMA5D2.

The PICAN 2 DUO counts with a CAN controller that receives data in this protocol and sends it to the VPU through SPI communication.

In the case of the VPU, it has three important tasks: to receive the CAN data sent by the PICAN 2 DUO through SPI communication and storage it in a datapool array saved in memory, to send a copy of this datapool to the GPU through I2C communication, and change the PWM duty cycle sent to the backlight according to the information received by CAN. These three functions are executed as a loop in the order described above.

In the case GPU, it is in charge of: receiving the datapool copy through I2C and saving it in storage memory, to stablish the link with the TFT and constantly (in real time) update the graphs displayed with the new received data, and also has a thread, that executes along with the two previous tasks described, that counts time for displaying the hour in the TFT or counting the time for the turn-arrows blinking.

In order to stablish the RGB connection between the GPU and the TFT, there is an interface manufactured in the PCB that converts the 50-pin-connector in the GPU to the 55-pin-connector of the TFT. It also counts with the LED driver that supplies with voltage and controls the TFT backlight directly by a PWM input that is received from the VPU.

# Implementation

## Codes

Appendix B

## List of files in VPU:

- main.c
- i2c.h
- I2c.c
- can.h
- can.c
- config_gclk.h
- config_gclk.c
- config_pm.h
- config_pm.c
- config_port.h

- config_port.c
- config_sercomI2C.h
- config_sercomI2C.c
- mcp_can.h
- mcp_can_dfs.h
- mcp_can.c
- spi.h
- spi.c
- timers.h
- timers.c
- uart.h
- uart.c
- myprintf.h
- myprintf.c

### Number of lines-of-code: 875

### Size of executable file:

- Program Memory Usage : 6228 bytes 2.4 % Full
- Data Memory Usage : 9500 bytes 29.0 % Full

### List of files in GPU:

- main.c
- i2c_datapool.h
- I2c_datapool.c
- graph_link.h
- graph_link.c
- ClusterIO_events.h

### Number of lines-of-code: 468

### Size of executable file:

- Source code: 890.5kB
- Graphics:

## BOM (Bill of materials)

| Reference / Description | Value | Qty | Reference in PCB | Manufacturer | Part# | pulg | mm | L | W | H | #xDxDatabook | Check in PCB |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Capacitor CIN | 10uF | 4 | C1-9 | KEMET X7R | C1210C106K8RACTU | 1210 | 3225 | 3.2mm | 2.5mm | 1.9mm | 100WEF | YES |
| Capacitor COUT | 10uF | 4 | C1-9 | KEMET X7R | C1210C106K8RACTU | 1210 | 3225 | 3.2mm | 2.5mm | 1.9mm | 100WEF | YES |
| Capacitor VDD5V | 10uF | 1 | C1-9 | KEMET X7R | C1210C106K8RACTU | 1210 | 3225 | 3.2mm | 2.5mm | 1.9mm | 100WEF | YES |
| Inductor L1 | ~~22uH~~/33uH | 1 | L1 | TAIYOYUDEN | NRS6045T330MMGKV | | 6045 | 6mm | 6mm | 4.5mm | 100NVH | YES |
| Schottky | 60 V 15 A | 1 | D1 | On Semiconductor | NRVTS8100MFST1G | SO-8 FL | | 4.56mm | 4.53mm | 1.3mm | 100WCL | YES |
| Resistor RGD | 10 Ohms | 1 | R1 | KOA speer | SG73G2ATTD10R0D | '0802' | 2012 | 2mm | 1.25mm | 0.5mm | 100285 | YES |
| Resistor Rsense | 25mOhms | 1 | R6 | KOA speer | WU732B15TTD25L0F | 0612 | 1632 | 1.6mm | 3.2mm | 0.6mm | 100RV1 | YES |
| nMOSFET Boost | 60 V 20 A | 1 | Q1 | infineon | IPG20N06S4L-26A | PG-TDSON-8 | | 5.9mm | 5.15mm | 1mm | 100VPD | YES |
| LP-8860 | | 1 | U1 | Texas Instruments | | | | | | | 100RTJ | YES |
| Connector 55 | | 1 | J2 | HIROSE ELECTRIC | FH28-55S-0.5SH(11) | | | 6.5mm | 32.08mm | 5.4mm | 100V1V | YES |
| Connector 50 | | 1 | U2 | OMRON | XF2M-5015-1A | | | 4.9mm | 29.1mm | 3.2mm | 100WNN | YES |
| Connector 6 | | 1 | J4 | IRISO ELECTRONICS | IMSA-12001S-06Y903 | | | 7.2mm | 24mm | 3.2mm | 100TV0 | YES |
| PULLUP Resistor | 1.5K | 2 | R2-3 | KOA speer | SG73G2ATTD1501D | '0802' | 2012 | 2mm | 1.25mm | 0.5mm | 100285 | YES |
| PULLDOWN Resistor | 1.5K | 2 | R4-R5 | KOA speer | SG73G2ATTD1501D | '0802' | 2012 | 2mm | 1.25mm | 0.5mm | 100285 | YES |
| Buck Capacitors | 47uF | 4 | C10-13 | PANASONIC | EEH-ZA1V470V | 1030 | | 7.8mm | 7.8mm | 5.8mm | 100WY3 | YES |
| Terminal Block 2Pos | | 3 | P1, P3 | OST | OSTT7020150 | | | | | | 100RLM | YES |
| 3 Pin Header | | 1 | J3 | KEC | 146282-3 | | | | | | 1009LL | YES |
| Terminal Block 3Pos | | 1 | J1 | Molex | 387007503 | | | | | | 100T3X | YES |
| Connector 20 Pins | | 1 | J5 | 3M Interconnect Solutions | 8520-4500PL | | | | | | 100VLY | YES |

# Tests

## Unit Test

| | Storyboard Gear Position implementation 25-Nov-2018 | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez, Sharif Nasser | | | |
| Description | Test that we can implent a new element in the Crank Story board to add the gear position | | | |
| Function | Add gear position in Crank StoryBoard | | | |
| Components | Storyboard .gapp, graph_link | | | |
| Hardware | PC with StoryBoard Software | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Create New Visuals in external application, or use text in Storyboard as new visual | Expect to create the new text for the gear position P R N D L and overlap them all | We now have the GearPostion in the CrankSoftware | OK |
| | Step1: Add an event to each letter of the PRND | Expect to create an event for each letter having a byte of information 0 being invisible an 1 visible | Correct creation of each Letter overlapped | OK |
| | Step 2: Create LUA file, and simulation | Create the Lua file to implement the information accordingly to the information that is being given and run the simulation | Correct simulation of the PRND texts. | OK |
| | Step 3: Testing in changing values | We expect the TFT to display the image and update the information displayed accordingly to the datapool values changed in the Canoe program | Video 2:Changing values and seeing the distinct forms | OK |
| Comments | In Canoe we send 0 1 2 3 4 5 6 to change the PRND,D1,D2,L and sending a value that is not in between 0 and 6 will leave us in the last established gear position. | | | |

| | **Changing PWM 29-Nov-2018** | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez, Sharif Nasser | | | |
| Description | Changing the PWM for the LED Driver Dimming | | | |
| Function | Sending a specific PWM cycle for the dimming | | | |
| Components | Port.c, Pm.c, gclk.c, timer. c, main | | | |
| Hardware | PCwith Atmel Studio | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Create the port for the PWM | Assign a correct port this case PA14 (PIN 2) to the PWM outout | Port assigned as output with PWM | OK |
| | Step1: Set Power manager for port | Assign the power that will flow through the PIN | Correct creation PORT | OK |
| | Step 2: Set GCLK an Timer | Assign the gclk and set the timer for the high to low change. And establish the correct frequency and duty cycle to the output. | Get the correct PWM frequency and duty cycle according to whats established. | OK |
| | Step 3: Testing in changing the PWM | Get the correct duty cycles at the output | Figure 13. Shows PWMs outputs at different duty cycles. | OK |
| Comments | | | | |

| | **Printing established Datapool in Arduino 23-Nov-2018** | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez, Sharif Nasser | | | |
| Description | Test that with only the computer with Atmel installed I can print the datapool and extract a specific signal, in this case the Gear Postion, seeing the datapool in the putty | | | |
| Function | Establish a predefined datapool, establish addresses, read datapool, extract signal, display datapool. | | | |
| Components | Datapool data , control (main). Uartdriver | | | |
| Hardware | PC with Atmel and Serial reader, FTDI , Arduino Zero | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Establish a datapool using structures and define the addresses of all the elements | We expect to create a correct union struct for the datapool with the elements information of the vehicle in it. | Figure 1. shows the defining of the addresses of the elements of the datapool Figure 2. Datapool structure | OK |
| | Step1: Start the program | Initialize all the components and elements in this case using the uart driver for the serial reading of the datapool. | Figure 3. Initializing the program in main | OK |
| | Step2: Start Putty with the right baud rate and read the buffer. | The program should continue execution and the rx_buffer should contain the datapool which shall be read at the using the putty interface | Figure 4.First try with DLC error marking 0 Figure 5.Putty reading | No, we had a minor where we corrected it by initializing correctly the length of the DLC |
| Comments | We had a minor problem which had to do with the DLC which was quickly resolved | | | |

## Integration Test

| | Initializing TFT display with Static image 25-Nov-2018 | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez | | | |
| Description | Test that we can initialize the TFT display using the export file from the Storyboard, to just display the static image to the display | | | |
| Function | Display a static image to the TFT display | | | |
| Components | Graphics driver, Yocto Project, Ubuntu in virtual machine | | | |
| Hardware | PC with Linux, Sama, TFT display | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Establish the correct connections between components, export Storyboard file and send it to the SAMA Board | Expect to have the correct export of the Storyboard | Figure 9. Storyboard Configuration, were we can observe the edited image that will be sent to the SAMA | OK |
| | Step1: Export Storyboard configuration to SAMA | Through scp transfer send the storyboard configuration to the SAMA | Succesful transfer form the Storyboard to the SAMA board | OK |
| | Step 2: Execute SAMA and kill the application launcher task | Once the program is executed in the display is the application launcher which we must kill before sending the updated graph from the storyboard. | Figure 10. Application launcher in display that is being killed next | OK |
| | Step 3: Declare environmental variables and Run the executable code to display graphics. | We expect the TFT to display the image in the correct resolution that we exported from the Storyboard, sending it through the SAMA. | Figure 11. Shows the excecutable instruccionens sent to the SAMA board. Figure 12. Shows the correct imaged displayed at the TFT display. | OK |
| Comments | At first we had an error in the aspect ratio, so we had the image cutout, for that we needed to make various changes in resolution to get the image correctly displayed | | | |

| | **Can Reading through SPI 23-Nov-2018** | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez, Sharif Nasser | | | |
| Description | Test that we can read the CAN values established form the Canoe, send them thorugh SPI form Pican to the Arduino reading the values using the UART | | | |
| Function | Read the values from Canoe to Arduino correctly in real time | | | |
| Components | Canoe , control (main). Uartdriver, SPI driver, MCPCANdriver, Myprintf driver | | | |
| Hardware | PC with Atmel and Serial reader, FTDI , Arduino Zero, PICAN, Vector | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Establish a datapool in Canoe with the respective addreses | We expect to create a correct datapool in the Canoewith the same addresses as above | Figure 6. shows the defining of the addresses of the elements of the datapool in Canoe | OK |
| | Step1: Initializa UART, SPI communication. Initialize Datapool | Initialize all the communication and elements in this case using the uart driver for the serial reading of the datapool. And the SPI communication between the Pican and Arduino | Figure 7. Initializing the program in main with the communication protocols correctly | OK |
| | Step2: Send data from Canoe to Pican and to Arduino | The data must travel in CAN form using the vector to the Pican and from there to the arduino in SPI form | It worked correctly and we can see the values once we print them using the UART | OK |
| | Step3: Print updated Can values | The program should continue execution and the rx_buffer should contain the datapool which shall be read at the using the putty interface | Figure 8. Datapool printed correctly al the way form the Canoe | OK |
| Comments | There were no problems once we could correctly send from the same arduino, there were just minimal changes to the previous unit test. | | | |

| | Changing the values in the TFT display 25-Nov-2018 | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez | | | |
| Description | Test that we can read the CAN values established form the Canoe, send them thorugh SPI form Pican to the Arduino reading the values using the UART | | | |
| Function | Read the values from Canoe to Arduino correctly in real time | | | |
| Components | Graphics driver, Yocto Project, Ubuntu in virtual machine | | | |
| Hardware | PC with Linux, Sama, Piican, Arduino Zero, | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Establish the correct connections between components, have Storyboard export ready and loaded to Sama, and Datapool established in Canoe program. | Expect to have the correct export of the Storyboard, Correct datapool established in canoe and the hardware correctly connected | Figure 9. Storyboard Configuration, were we can observe the edited image that will be sent to the SAMA Figure 12. Shows the physical connections made between hardware | OK |
| | Step1: Export Storyboard configuration, Initializa SPI communication in arduino. | Succesful Storyboard transfer and arduino initialization | Succesful transfer form the Storyboard to the SAMA board and arduino ready from Canoe and sending to SAMA | OK |
| | Step 2: Initialize I2C communication in SAMA and execute SAMA and kill the application launcher task. | Succesful initialization of Arduino and Sama as well as killing launcher app to begin sending the new graphics | Figure 10. shows initial application launcher | OK |
| | Step 3: Update at Datapool values in Canoe to be send to Can. | We expect the TFT to display the image and update the information displayed accordingly to the datapool values changed in the Canoe program | Video 1: Shows the Display updating its values when changing at the Canoe Video 2: Shows an error were we could refresh the value | OK |
| Comments | We made various test like sending a signal much higher than the normally permitted, for example sending a 500 mph, were 200 is the maximum, for this we limit the storyboard to only display its maximum which is 200. we also had some issues were it was quickly fixed because we were sending the values pointer insted of the value. | | | |

## System Test

| | **SYSTEM TEST 25-Nov-2018** | | | |
|---|---|---|---|---|
| Team | RoboDuo | | | |
| Responsible | Pedro Gámez,Sharif Nasser | | | |
| Description | Test the whole system | | | |
| Function | Initialize all the components and run all the communication protocols. | | | |
| Components | Graphics driver, Yocto Project, Ubuntu in virtual machine, PC, Arduino Code, Sam Code | | | |
| Hardware | PC with Linux, Sama, Piican, Arduino Zero, | | | |
| Procedure | | Expected behavior | Actual behavior | Result |
| | Establish the correct connections between components, I2C wires, SPI wires, Power and Ground cables. | Have a correct connection between every component, avoiding non connection errors | Correct hardware connections. Figure 12. Shows the physical connections | OK |
| | Step1: Run Arduino Code and open all the programs | Open CANoe , Run Ubuntu yocto project | All the programs running correctly | OK |
| | Step 2: Initialize the and run the SAMA code | We expect to have all the programs running correctly and the code to run correctly and have the display functioning and the Canoe sending values | We can observe all elements working correctly without errors | OK |
| | Step 3: Tests all the components and the display | Send information from the Canoe and have that information be updated in the TFT display. | Video 3: We can observe the whole system test changing values in can and displaying it at the TFT. | OK |
| Comments | Luckily, all the components worked as expected and correctly, although this system test is also to be tried with the PCB which has the LED driver, although is must still be ought to be tested with the PCB (the connectors for the display do work. | | | |

# Tests Evidence

Figure 1.

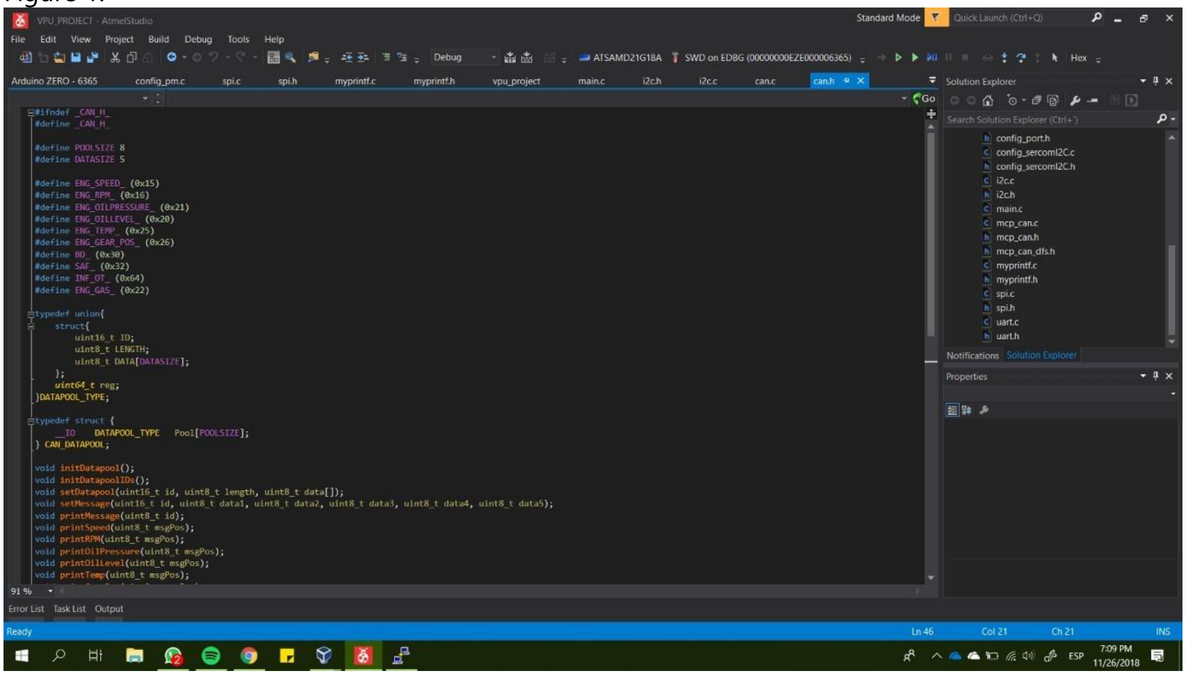



Figure 2.

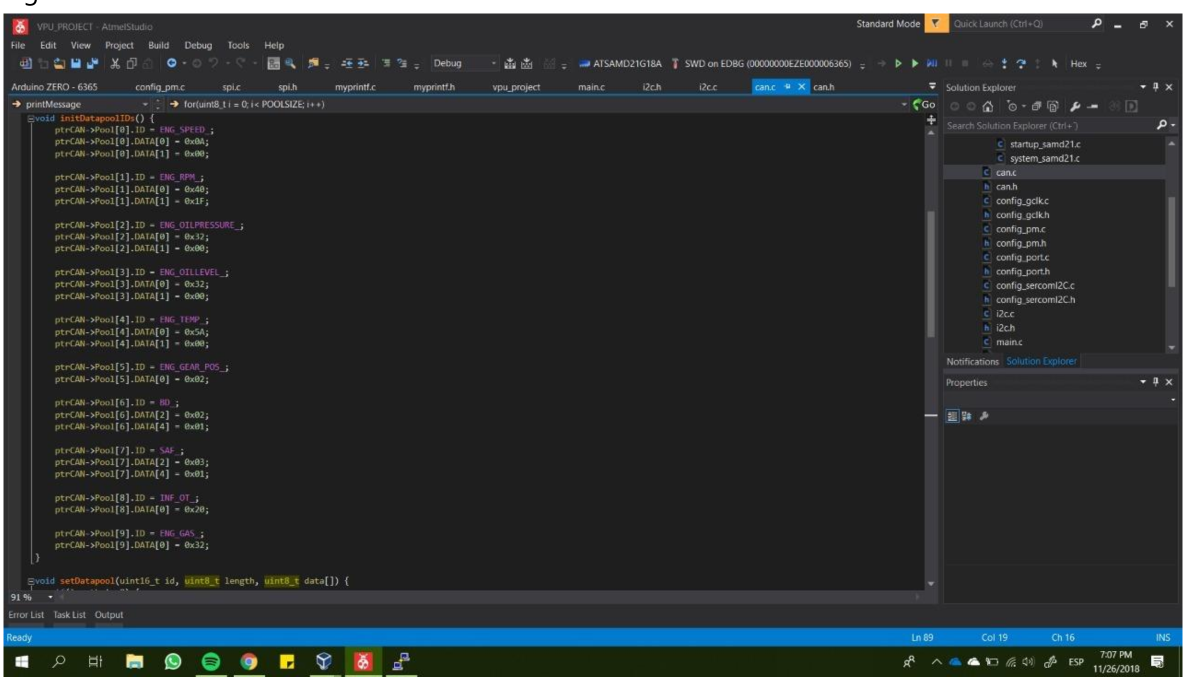



Figure 3.

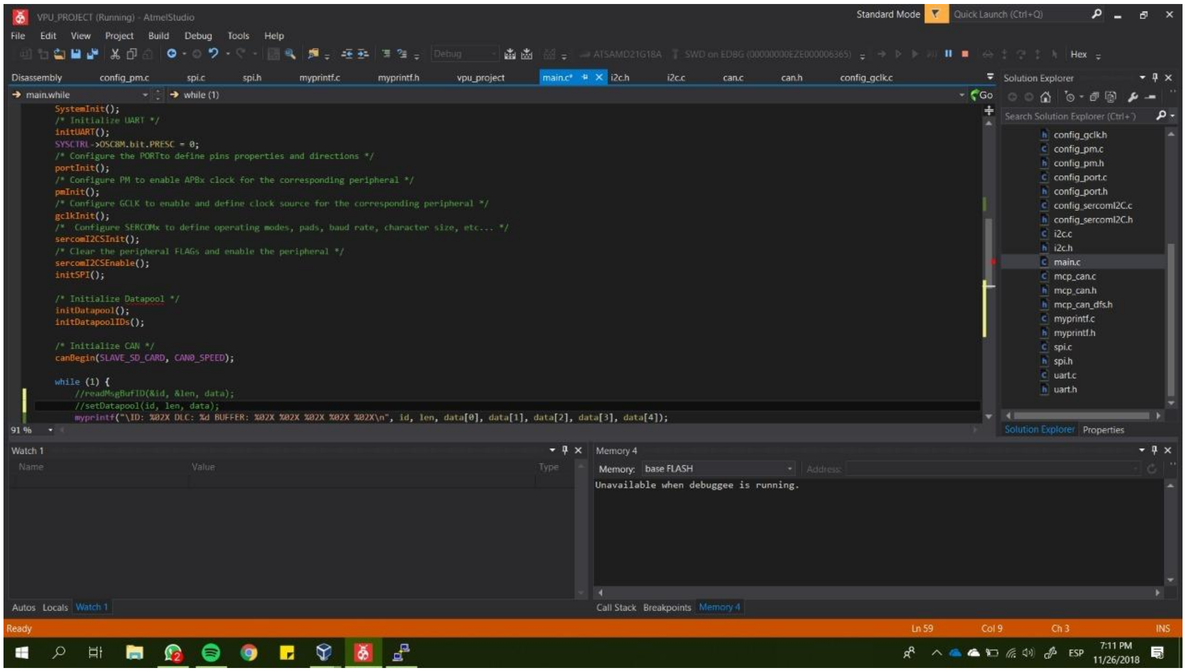



Figure 4.

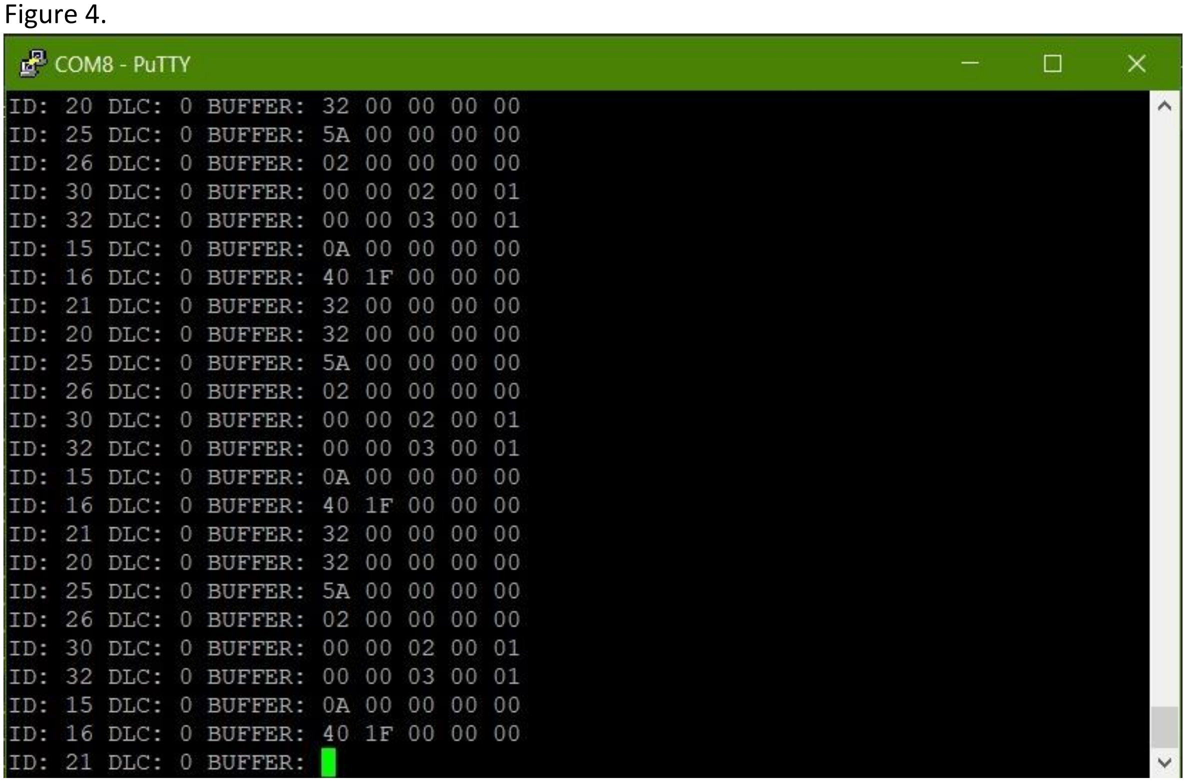



Figure 5.

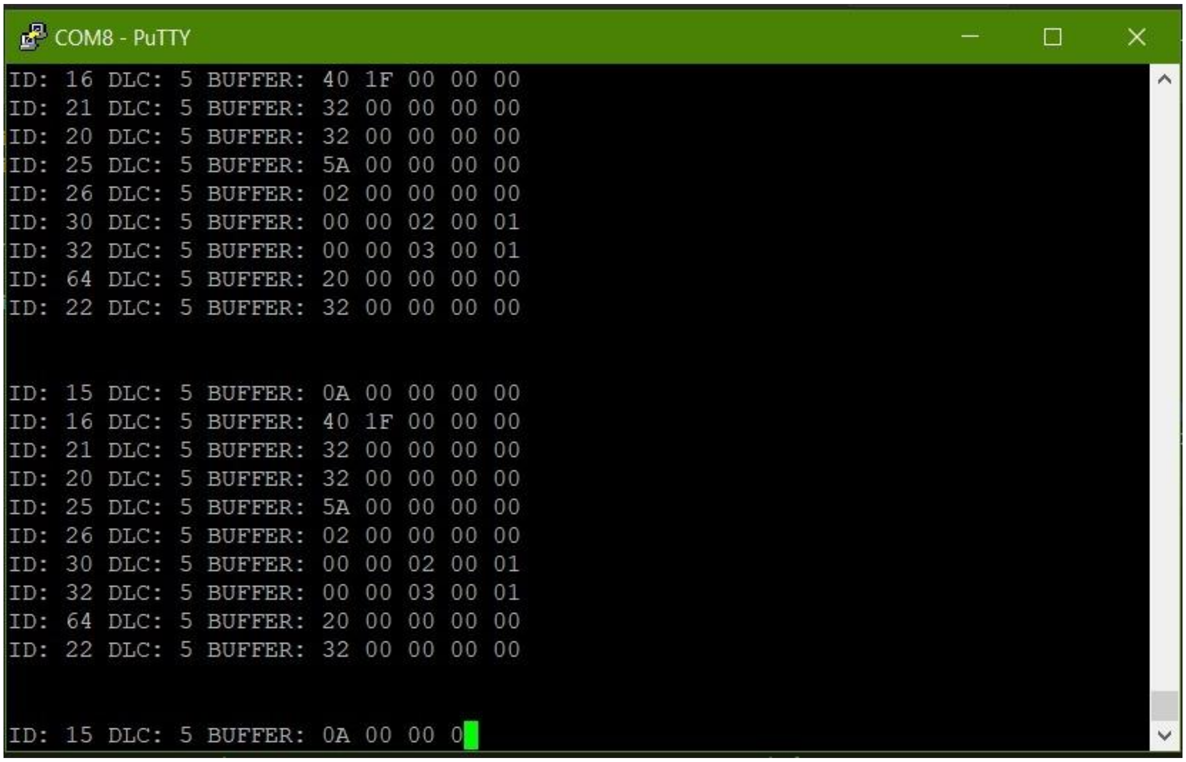



Figure 6.

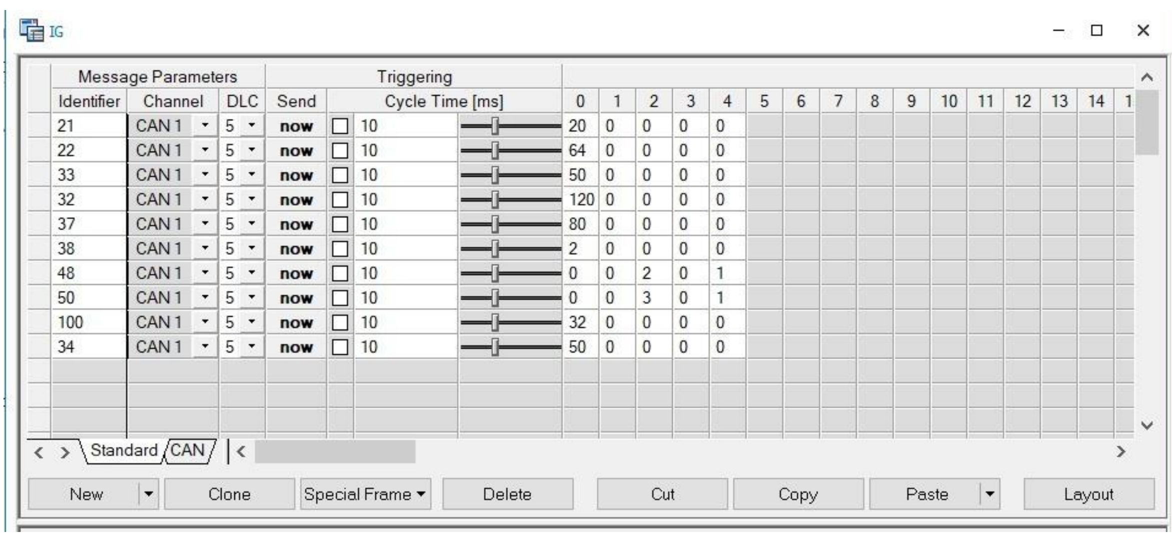



Figure 7.

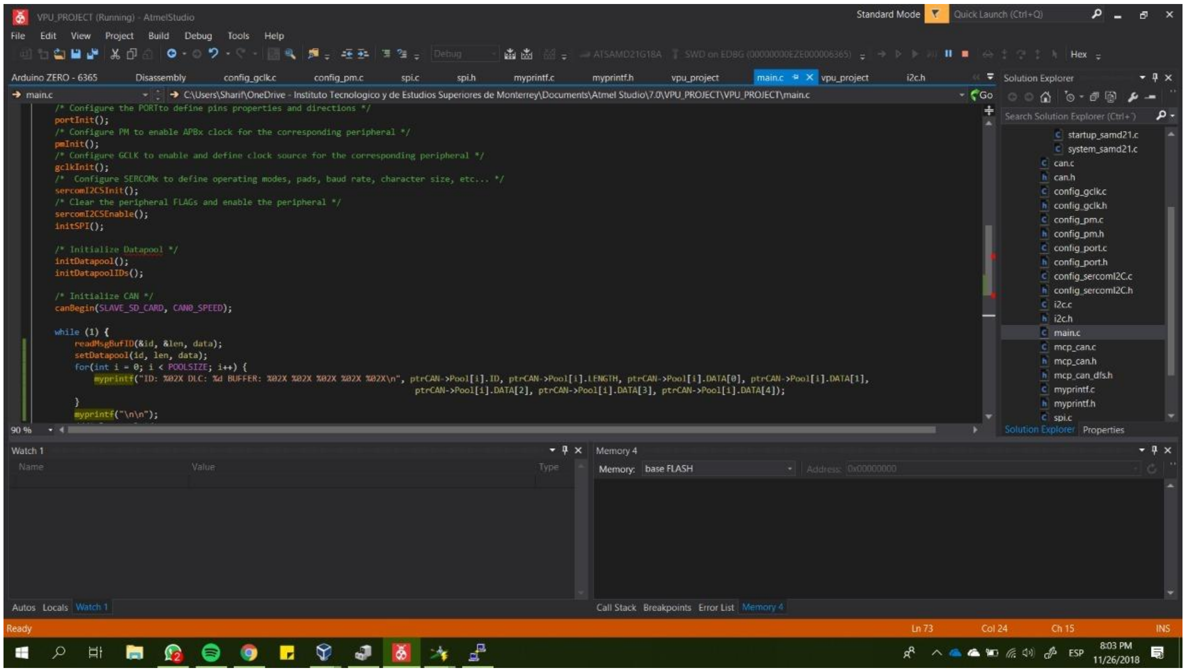



Figure 8.

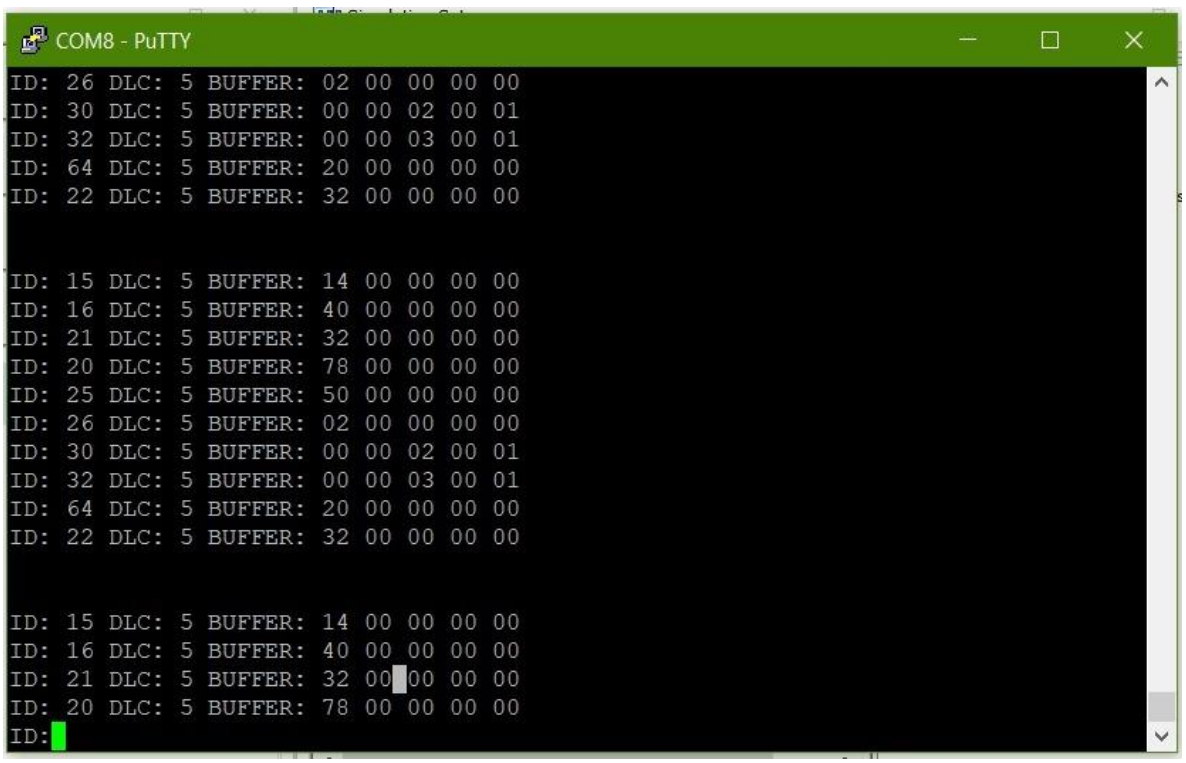



Figure 9.

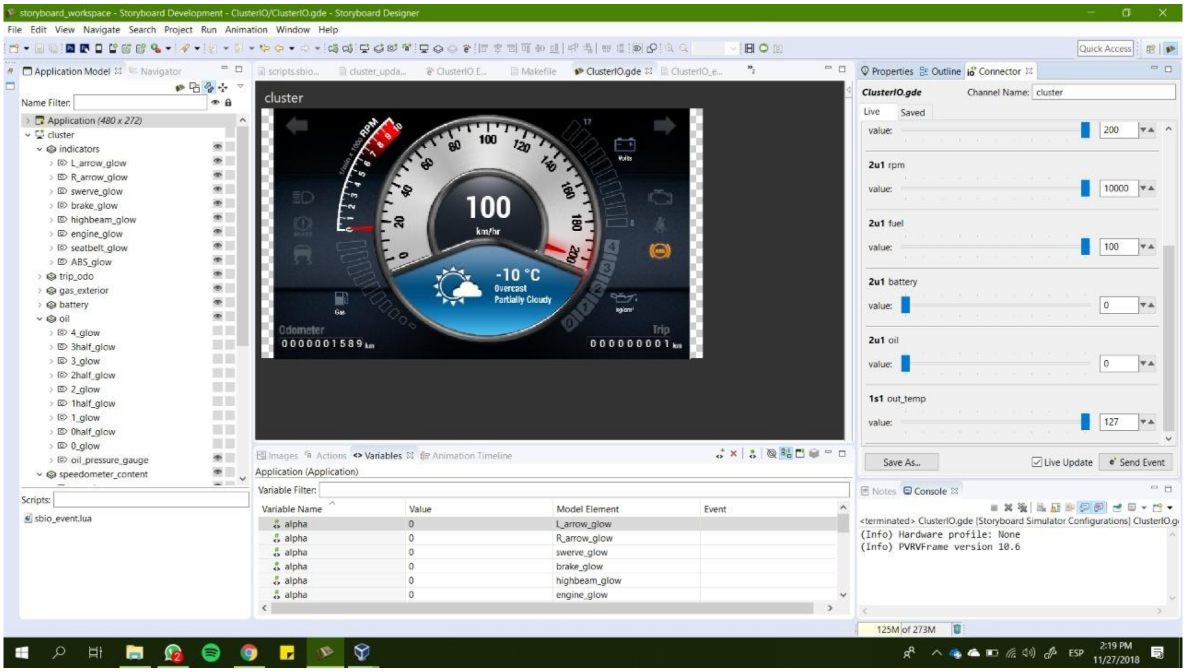


Figure 10.

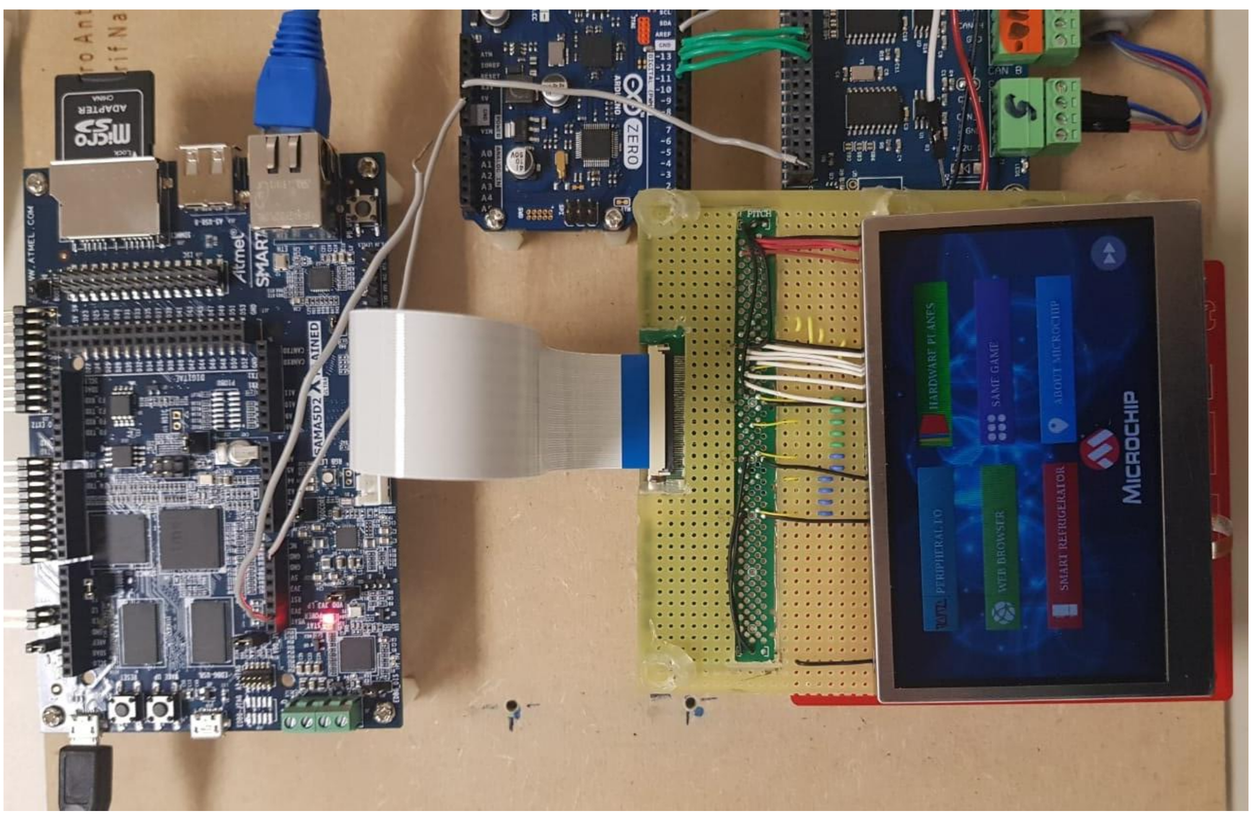
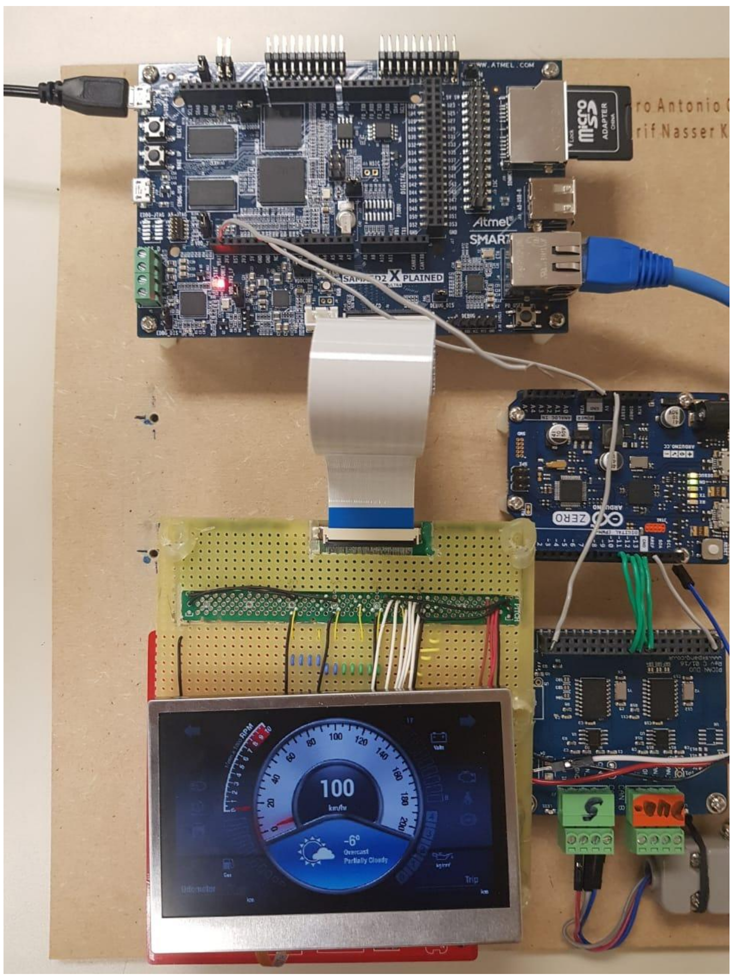


Figure 11.


Figure 12.



Video 1.

https://drive.google.com/file/d/1S4Rp3hOnc8uHFqs4TseQQIZOoOFwMPG9/view?usp=drivesdk

Video 2.

https://drive.google.com/file/d/1U0rOlEeuFqbLii0HBMK4XentWDKHt8ID/view?usp=drivesdk

Video 3.

https://drive.google.com/file/d/1jEmB9WClN75Y6lGYY3MuP1v3Y80_HdyC/view?usp=drivesdk