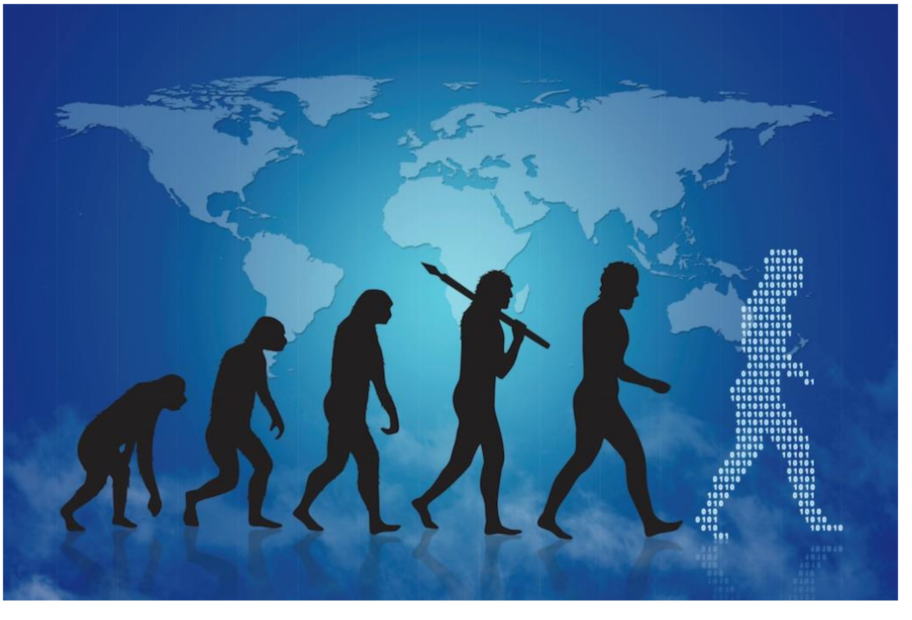
Book of Projects

"The only source of knowledge is experience", Albert Einstein



Steve Wagner 19/07/2017

Summary

Figure 1 Bosch XDK board with all available sensors

Figure 2 Ublox entension with GSM LisaU2

Goal
Migration of several drivers to a new Hardware Abstraction Layer (HAL) design.
Adapt unit and integration tests, also the specifications and the documentation of the drivers

Material used
Bosch XDK prototype board with several internal sensors (Accelerometer, Gyroscope, Magnetometer, Humidity sensor, Pressure sensor, Temperature sensor, Acoustic sensor, Digital Light sensor), TP-Link Router, Wifi-cc3100mod module, Ublox extension board with GSM LisaU2

Programming language
C, Python

Time of work
8 months (08/2016 – 03/2017)

Achievement
Each driver has been correctly migrated to the new HAL design. The unit tests have been implemented and were working as previously. Some specifications and integration tests were deleted, because the new HAL design was managing differently, but other specifications and tests were added to keep the coherence and the security of the drivers

Tools
Eclipse for the programming of the drivers, PyCharm for the management of the integration tests, Stash for the project management, Jenkins as version control systems, GTest for the unit tests, Git for the management of the code

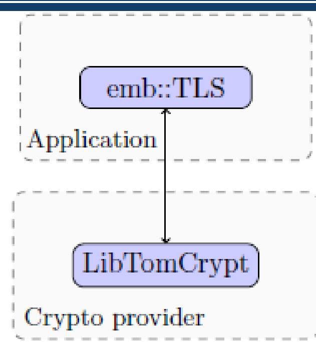


Figure 3 TLS project at the beginning without the interface

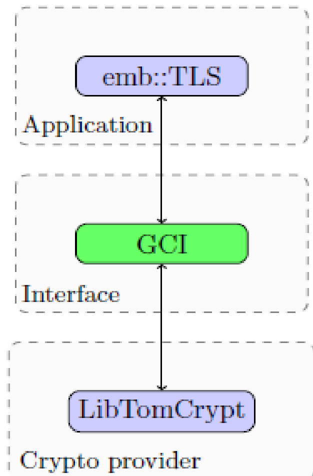


Figure 4 Project at the end with the Generic Cryptographic Interface (GCI)

Goal
Design of a cryptographic interface with the basis of the algorithms needed for any cryptographic project

Implementation of this new interface into an existing TLS security protocol project

Implementation of an open-source cryptographic algorithms into the new interface

Updating existing tests to make them run as before

Programming language
C

Technology
Cryptography, Internet security protocol TLS

Time of work
6 months

Achievement
After learning the basis of Cryptography and the Internet Security Protocol TLS, the project has been split into 4 parts.

First one was the writing of the new interface, which has been approved by the all institute working on the TLS security protocol project.

Then this interface has been implemented into the project. It replaces an old one which wasn't optimized for the use of different cryptographic algorithms.

After implementing the interface, the open-source cryptographic algorithm, which was already used in the project, has been written into the interface until the tests of each part of cryptography were working as before.

The last part was the writing of the bachelor thesis report and the final presentation.

Tools
Eclipse for the programming, Wireshark as sniffer, Redmine for the project management, Git for the code management, Latex for the Bachelor Thesis report, Beamer for the Bachelor Thesis presentation

Figure 5 Starting bloc

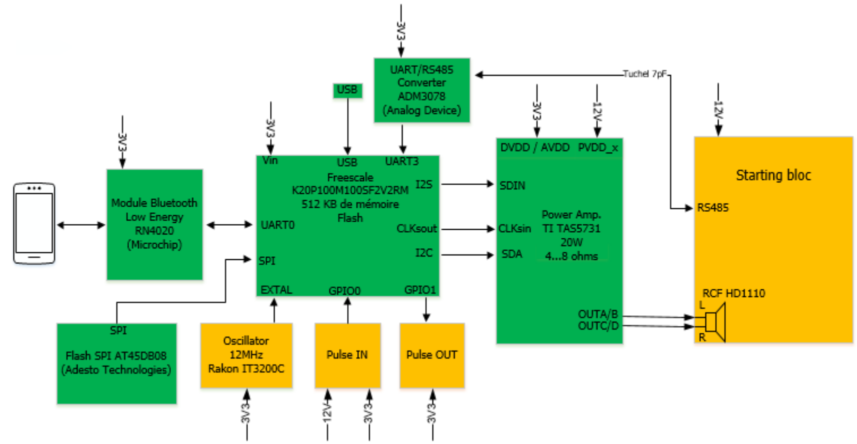


Figure 4 Bloc scheme of the system

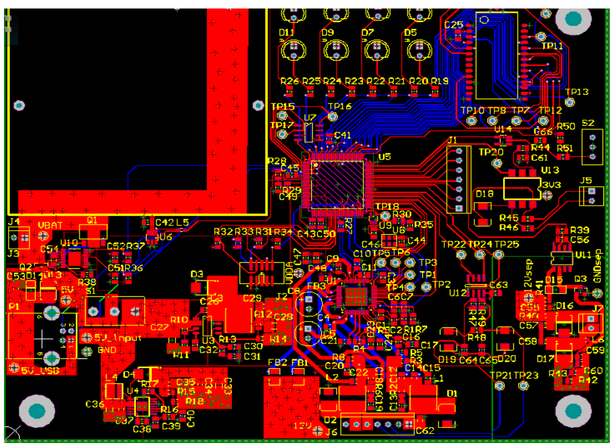


Figure 6 Layout of the final system

Figure 6 Final system

Goal
To Develop a training system that allows connecting one or several starting blocks to a mobile or tablet app via Bluetooth. The training system generates the audio output for the start commands ("on your marks", "set") and the start signal ("*gunshot*"). The system should receive information from the starting block (force curve, reaction time) but also from the app (delay between each sound, number of sprinters) and managed by the system.

The project has been split into two subprojects: Mobile application on Android and IOS, managed by another student and the Embedded Hardware and Software managed by myself

Programming language
C

Achievement
First the specifications had to be understood and all the components needed for the project choose according to the specifications. A bloc scheme was created for more understanding with the customer. Then after agreement of the customer, the electronic scheme and the PCB were realized. During the creation of the board the Bluetooth and the RS485 for the Starting Bloc communication protocol were realized and tested.

The soldering of the SMT component was done with electronic test to avoid any short-circuit.

All the drivers have then been written (Bluetooth, RS485, and Audio) and the whole system realized and tested.

The system was working successfully

The Bachelor thesis report and presentation have then been written.

Tools
Altium Designer for board conception, Eclipse for programming, Visio for the block scheme

Hardware communication
I2C, I2S, UART, GPIO

Gear
Oscilloscope, multimeter, development board Freescale Kinetis K20

Technology
Bluetooth, RS485, Digital Audio Amplifier

Figure 7 Map with the localization of the two embedded boards

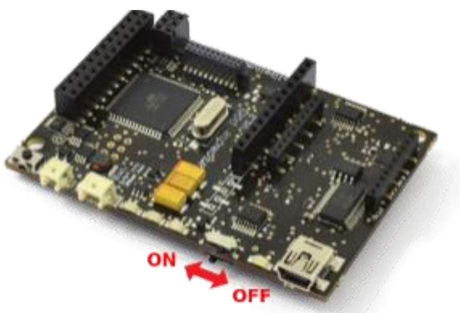


Figure 8 Wapsmote board

Goal
To know the placement of a parcel in real time through embedded systems

Team
2 Students in Bachelor of Informatics specialized in embedded systems (third year)

Programming Software
MBED platform, Waspmote platform

Material used
Ublox C027 and Wapsmote embedded board

Programming language
C, C++

Time of work
50 hours (February 2015 – April 2015)

Methods
The user enters the arrived coordinates and his phone number in a text file. The text file is saved in a web server. The embedded system receives his GPS coordinates every 5 minutes through a GPS module and sends it to the server. The server compares the received coordinate with the saved coordinate. If the coordinates correspond, the server sends a message with the user's phone number to the embedded system. The embedded system sends at this time a SMS to the user to inform him that his parcel is arrived.

Final Result
Each embedded system receives its GPS coordinates and sent them to the server. An SMS is sent to the user when the parcel is arrived

Autonomous Robot Car

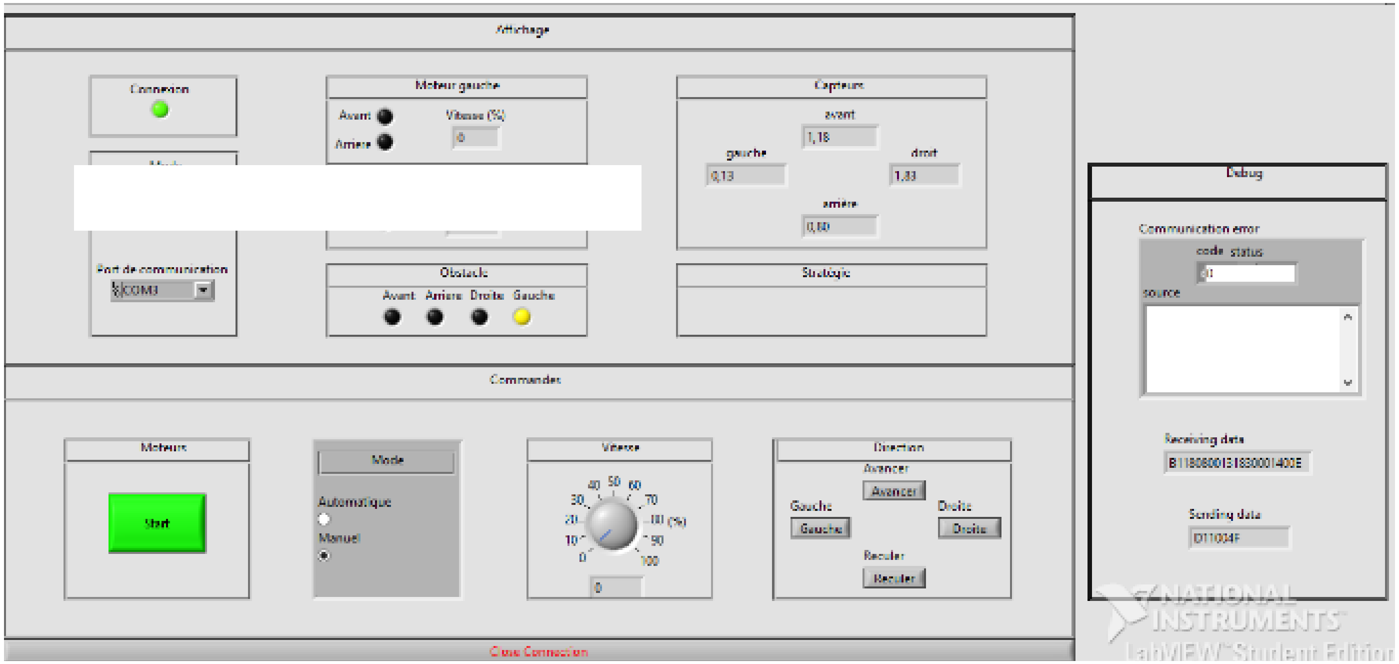


Figure 11 Monitoring with Labview

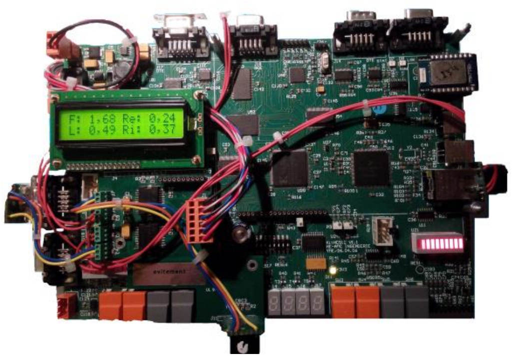


Figure 10 Robot car (top view)

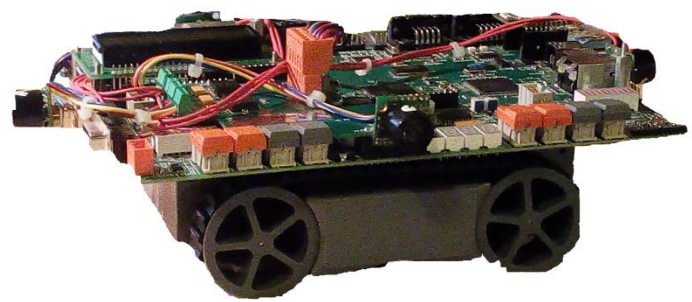


Figure 9 Robot car (left view)

Goal
The robot should move in an unknown environment and avoid the obstacles. It can also be guided through a software interface. A Real Time Operating Systems (uOsII) should be used for the understanding of all the specificity of a RTOS. Two microcontrollers are used on the board. One should manage the input and output of the robot and should also manage the data coming from the other microcontroller. The second one should manage the communication through Bluetooth with the software interface and manage the data from the first microcontroller.

Programming software
Codewarrior, LabVIEW

Programming Language
C

Time of work
150 hours (February 2015 - May 2015)

Method
Ultrasonic sensors are used to determine the distances on each side of the robot. This one move automatically without touching any obstacle. Through the software interface, the robot can be guided by a user through the software interface or move freely. On the software interface each distance of each sensor, between the robot and a possible obstacle, can be visualized. Several tasks, semaphore and mutex were used for the management of the project, as the inputs control or the data to or from the software interface.

Final Result
The robot moves automatically and can be guided through the software interface. On the software interface can the speed, the start and stop of the robot and the automatic and manual mode chosen.

Freescale Cup

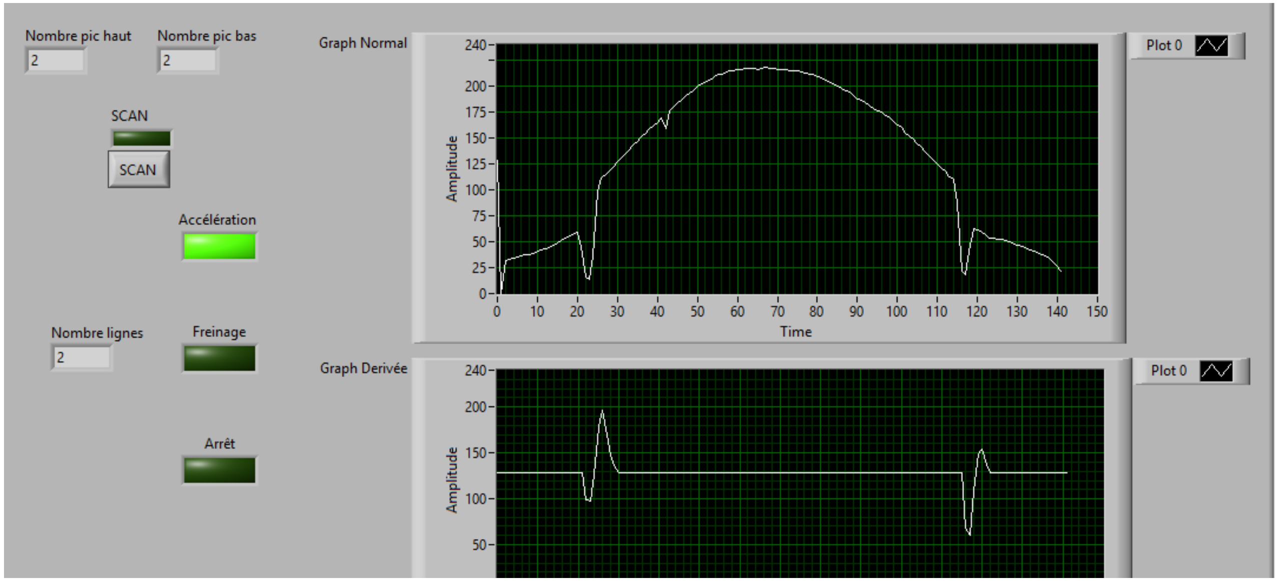


Figure 14 Monitoring Labview

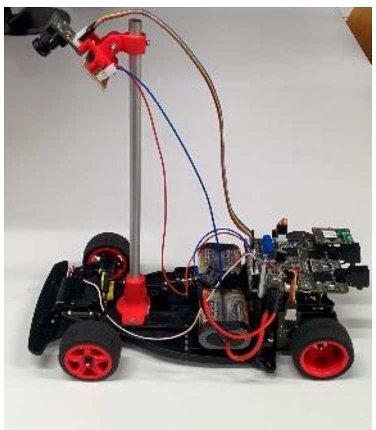


Figure 12 Freescale Cup's car (left view)

Figure 13 Freescale Cup's

Goal
The robot may move as speed as possible on a white circuit, with two lines on each part, without going out of it.

Team
Two students in Bachelor of Informatics specialized in embedded systems (third year).

Programming Software
Codewarrior, LabVIEW

Programming language
C

Time of work
200 hours (September 2014 - February 2015)

Method
The robot has a camera on the top of a stand, which is on the front of the robot and allow the visualization of the circuit. The result coming from the visualization of the camera is processed and, depending on the result, it’s differently interpreted on the engines (acceleration, braking, turn right or left, etc.). The result of the processed image could be visualized on a software interface.

Final Result
The robot can drive without going out of the circuit. The processed image can be visualized on a software interface with the state of the robot (acceleration, breaking, stopped) and the average speed.

Led Controller via Android

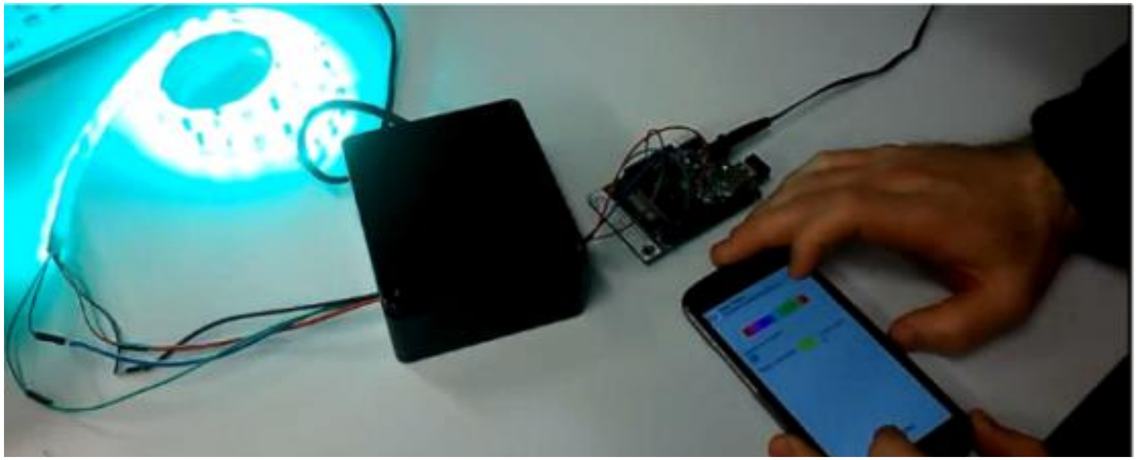


Figure 17 complete assembly with PIC24F board, the transistors (black box) to put the PWM color from the board to the leds

Figure 16 One of the page of the Android application

Figure 15 PIC24F (from Microchip)

Goal
Creation of an android application which allows the change of led color through Bluetooth

Team
Two students in Bachelor of Informatics specialized in embedded systems (third year).

Programming Software
Eclipse (programming of the Android application), MPLAB X IDE (programming of the embedded board)

Time of work
200 hours (September 2014 – February 2015)

Programming Language
C (for the embedded board) and Java Android (for the Android application)

Time of work
200 hours (September 2014 – February 2015)

Method
The color is sent through the Bluetooth from a phone to an embedded board (where the leds are connected). This board receives the frame with the value of the colors and adapts them to send it, as a Pulse Width Modulation (PWM) signal, to four transistors. These four transistors are used to convert this PWM signal to a power signal and have the corresponded color on a led band.

Final Result
The color changes in four methods (phone rotation, movement of shake and through a slider)