This document will help to know how the project is going on.

[Introduction written – OK]

1. Arduino
2. What is an Arduino?
   1. History and features of microcontrollers
   2. Arduino Family
   3. The UNO and NANO
   4. MEGA (ATMEGA 2560)
3. RFID kit with thirty lessons from the switch button to the password lock
   1. What you need

Doing electronics in your garage or bedroom requires a minimum of equipment. The advantage of Arduino and all the modules are you just need few tools to begin electronics, so you won’t need to spend a lot of money. Here is the list of mandatory equipment:

* 1. Controlling LED by button

[Structure]

* + - 1. Introduction
      2. Components
      3. Principle
      4. Schematic
      5. Code
  1. Controlling LED by PWM
  2. Interactive LED flowing lights
  3. RGB LED
  4. LED display (dot-matrix,
  5. Photoresistor
  6. Tilt switch
  7. Button switch
  8. Buzzer
  9. Answer machine
  10. Controlling voice by light
  11. Voltmeter
  12. Password lock
  13. Flame sensor
  14. Voice sensor
  15. Temperature sensor
  16. Water level sensor
  17. Humidity sensor
  18. Shift register 74HC595

Segment display, add more output, etc..

<https://www.arduino.cc/en/Tutorial/ShiftOut>

<http://eskimon.fr/tuto-arduino-901-ajouter-des-sorties-num%C3%A9riques-%C3%A0-larduino-le-74hc595>

<https://learn.adafruit.com/adafruit-arduino-lesson-4-eight-leds/the-74hc595-shift-register>

* 1. Relay module
  2. Joystick

Simple joystick using potentiometer.

* 1. RTC

Simple use, we’ll see later how to bind it with an SD card to make a DataLogger.

* 1. Serial monitor

Practising the serial monitor of the Arduino.

<http://forum.arduino.cc/index.php?topic=396450.0>

1. Communication
   1. Wired
      1. UART

[Structure]

* + - 1. Introduction

Explain the history, where we can find the bus, the applications …

* + - 1. Protocol

Describe the protocol of the bus

* + - 1. Principle

In this part, will describe the electronic way (or circuit) to produce create the bus and the signals of the communication. For example, the CAN module will be presented with the main chip MCP2515 that cares about the CAN protocole. The circuit is shown with all the components and their role if I am sure about the purpose of each components. If it is inside the AVR, I will explain the registers used for this purpose.

* + - 1. Example

This part will show basic examples, advices and tricks according to my experience.

* + 1. I²C
    2. SPI
    3. CAN – MCP2515 chip
    4. Ethernet
  1. Wireless
     1. Bluetooth – HC-05 chip
     2. Radio 2.5G NRF24L01+PA+LNA SMA
     3. RFID RC-522
     4. WIFI with ESP8266 and/or ESP32

1. Sensors and they board
   1. Ultra-Sonic sensor HC-SR04

[Structure]

I. Introduction

The use of the technology and how it works. For instance, the ultrasonic waves. Where they are used, what is the physics behind it.

II. The chip

This part presents the sensor on the PCB, the circuit if available and the different signals read by the oscilloscope to show the electronic behavior. This part is mainly to introduce sensors and their filtering or other circuits with basic and advanced electronics. For example, the CAN module will be presented with the main chip MCP2515 that cares about the CAN protocole. The circuit is shown with all the components and their role if I am sure about the purpose of each components. If not, I will put a comment to let the other who will read my notes to explain me.

III. The code

Here will see the library (if used) and explain how the code works. It is mainly to explain how it is possible to “convert” the electronic information to another language. Depending on the difficulty of the sensor, I will let some blank as the previous part. In this case, a radar using the ultrasonic waves, see <https://howtomechatronics.com/projects/arduino-radar-project/> using Processing.

IV. Project examples

The last part is the most pragmatic: I will show some of the project I made with the sensors, or for those which I didn’t make a sufficient project, I will leave URL of interesting projects.

**What subjects are covered** ? OP AMP, AVR Timer,

Sources:

<https://uglyduck.ath.cx/ep/archive/2014/01/Making_a_better_HC_SR04_Echo_Locator.html>

<http://extremeelectronics.co.in/avr-tutorials/interfacing-ultrasonic-rangefinder-with-avr-mcus-%E2%80%93-avr-tutorial/>

<https://os.mbed.com/components/HC-SR04/>

<http://www.ezdenki.com/ultrasonic.php>

<http://extremeelectronics.co.in/avr-tutorials/avr-timers-an-introduction/>

<http://extremeelectronics.co.in/avr-tutorials/timers-in-compare-mode-part-i/>

<http://extremeelectronics.co.in/avr-tutorials/input_output_ports/>

* 1. Infrared sensors HC-SR501

<http://henrysbench.capnfatz.com/henrys-bench/arduino-sensors-and-input/arduino-hc-sr501-motion-sensor-tutorial/>

* 1. Laser sensor VL53L0X

<https://github.com/pololu/vl53l0x-arduino>

<https://learn.adafruit.com/adafruit-vl53l0x-micro-lidar-distance-sensor-breakout/overview>

* 1. Microwave sensor CDM324 induction radar

<https://www.limpkin.fr/index.php?post/2017/02/22/Making-the-Electronics-for-a-24GHz-Doppler-Motion-Sensor>

* 1. Speed encoder HC-02K and PID controller

Introduction: the purpose of this sensor

Principle: how it works and how it is mounted on a PCB? Schematic, explanation if available

Example(s): How to use it with the Arduino through an example, here a PID controller, see <https://create.arduino.cc/projecthub/tolgadurudogan/tricks-for-controlling-dc-motors-3a05a5?ref=platform&ref_id=424_trending___&offset=1> .

* 1. Accelerometer & gyroscope GY-521 MPU-6050

<http://tiptopboards.free.fr/arduino_forum/viewtopic.php?f=2&t=28>

<https://maker.pro/arduino/tutorial/how-to-interface-arduino-and-the-mpu-6050-sensor>

* 1. GPS module NEO-6M-GPS6MV2

<http://www.instructables.com/id/How-to-Communicate-Neo-6M-GPS-to-Arduino/>

1. Graphics/Information display
   1. Classic liquid crystal display (LCD 2\*16)

Introduction

<https://howtomechatronics.com/tutorials/arduino/lcd-tutorial/>

* 1. OLED 128\*64

<https://projetsdiy.fr/ssd1306-mini-ecran-oled-i2c-128x64-arduino/>

<http://www.instructables.com/id/Monochrome-096-i2c-OLED-display-with-arduino-SSD13/>

Bitmap generator: <http://javl.github.io/image2cpp/>

* 1. Tactile Display

<https://howtomechatronics.com/tutorials/arduino/arduino-tft-lcd-touch-screen-tutorial/>

<https://learn.adafruit.com/adafruit-gfx-graphics-library/graphics-primitives>

1. Storing data
   1. Type of memory
      1. Flash
      2. EEPROM
   2. SD card

<https://www.carnetdumaker.net/articles/lire-et-ecrire-des-donnees-sur-une-carte-sd-avec-une-carte-arduino-genuino/>

<https://learn.adafruit.com/adafruit-micro-sd-breakout-board-card-tutorial?view=all>

<http://makecourse.weebly.com/sd-card-data-logging.html>

* 1. Useful example: Data logger

<https://learn.sparkfun.com/tutorials/openlog-hookup-guide>

<https://publiclab.org/notes/cfastie/04-30-2017/data-logger-shield-for-nano>

1. Arduino and C++ Oriented-Object Programming (OOP)
   1. Basics C++ OOP

This is not a course about C++ and OOP but rather a reminder of the features of Oriented Object in order to introduce the Oriented Object way to Arduino.

* 1. OOP with Arduino

Once we have masterd basic projects like blinking LED and simple sensors, it’s time to move on to bigger and better projects. That involves combining bits and pieces of simple sketeches and trying to make them work together. As we saw, Arduino is a very simple processor with no operating system instead of a Raspberry or Beaglebone, that can only run one program at a time.

That doesn’t mean we can’t manage multiple tasks on a n Arduino. We just need to use a different approach. Since there is no OS to helps us, we have to take matters into our own hands.

Remember the blink without delay, one of the most famous example of a state machine. When the code begins to be big by using many functions for example and it is repetitive with the “objects” it uses, we use OOP.

We have already used files and classes using OOP. For example, the object Serial is a subclass of the Stream serial.

[Sources]

<https://paulmurraycbr.github.io/ArduinoTheOOWay.html>

<https://www.arduino.cc/en/Reference/APIStyleGuide>

<https://www.arduino.cc/en/Hacking/LibraryTutorial>

<https://learn.adafruit.com/multi-tasking-the-arduino-part-1?view=all>

* 1. Qt Application (more generally, GUI application)

It is useful for a project to have a User Interface (UI) to print information of sensors, to have buttons to control the INPUT of the Arduino. The first UI we have seen so far is the Serial Monitor and the Serial Plotter of the Arduino’s IDE. It will depend on the purpose of your project, but if your boards are connected to your computer, it may be a good idea to have an UI. For most of my projects, I used the Qt library which is a really huge library. I won’t do a course about Qt for two reasons: it will take more than a book to cover all the feature and I am not skilled enough to pretend teaching Qt. However, I will show you how to create a pretty simple application and give you the keys to continue for bigger project. Projects : serial plotter, gui car

Qt installation

To install the library, please refers to these website [URL]. It is important to install the library properly. I don’t have particular advice but follow all the steps carefully.

What library

As I said, Qt is a huge, deep and wide library that have almost a library for evertying : Data base, Sensors, etc. For our needs, we’ll use QSerialPort which is designed to connect devices with serial communication.

* 1. Examples (Protocole & Parseur, CAN, …)

1. Deeper look inside the Arduino: the AVR 8-bits microcontroller

In this part we will see the last additional feature of the AVR microcontroller, the Assembly language and how the microcontroller is written. SO far we used library function, written by the Arduino’s collaborators but what we don’t know is, how these function interacts with the processor. This prat won’t be a course about microcontroller architecture and programming, but it will introduce you to it.

* 1. AVR architecture and assembly language
  2. AVR programing in C
  3. AVR Timer programing in assembly and C
  4. AVR Interrupt programing in assembly and C
  5. AVR Serial Port programing in assembly and C
  6. To go farther

[Sources]

The AVR microcontroller and embedded systems using assembly and C

1. …
2. Electronics

The last part deals with electronics in general, for instance basic and advanced circuits. I will follow the same structure as the book “Learning the art of electronics” which is a lab course-like. However, my part won’t be a rea course because I’m not certified to do that and don’t have enough skills, but I’ll show what I have learned to do. This is more a library of many tips and circuits. Useful websites:

<https://www.electronics-tutorials.ws/>

<https://www.allaboutcircuits.com/>

<https://www.khanacademy.org/science/electrical-engineering>

<https://wiki.analog.com/university/courses/electronics/text/electronics-toc>

So here is the plan:

1. Analog

[Structure] Deals with analog components and circuits: resistor, diodes, capacitor, inductor, transistor, operational amplifier, voltage regulator…

* 1. Voltage and current, reminder of the basics
  2. Capacitors and inductors
  3. Diodes
  4. Transistors
  5. Operational amplifier
  6. Voltage regulator
  7. Famous components
     1. 555 timer IC

<https://www.jameco.com/Jameco/workshop/TechTip/555-timer-tutorial.html>

<https://www.electronics-tutorials.ws/waveforms/555_timer.html>

<http://www.instructables.com/id/555-Timer-Emulator-for-Arduino/>

<http://www.instructables.com/id/555-Timer/>

* + 1. TL494

<http://afrotechmods.com/tutorials/2011/11/28/lm317-adjustable-voltage-regulator/>

<https://www.homemade-circuits.com/simplest-pwm-modified-sine-wave/>

<https://www.electronics-tutorials.ws/blog/variable-voltage-power-supply.html>

1. Digital

[Structure] This part is bound with the first part, microcontroller, because it is what microcontroller is made of: digital. However, it needs analog device to create digital device like TTL or CMOS. So we will improve our knowledge about digital. The plan is inspired by the book “Fundamentals of digital circuits” written by A. Anand Kumar.

* 1. Number systems
  2. Binary codes
  3. Logic gates
  4. Flip-flops
  5. Shift register
  6. Counter
  7. Sequential circuits and algorithmic state machines
  8. Logic families (TTL, IIL, ECL, MOS, CMOS)
  9. Analog-to-Digital and Digital-to-Analog converters
  10. Memories

1. Electro-mechanics

[Structure] The last part to explain motors : DC, stepper, servo, brushless, etc.. We won’t work about AC motors because I don’t have any of them. Most of the features will have already been seen like servo, stepper and PID controller for DC.