Introduction to the e-puck2 robot and TP1

Francesco Mondada
IMT - STI - EPFL

Introduction

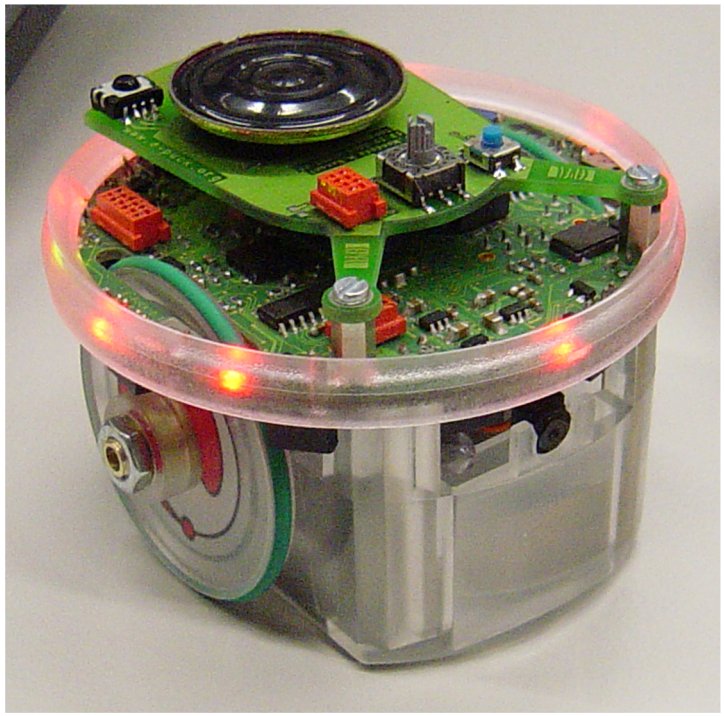
The goal of this lesson is to get you introduced to some elements of the e-puck2 mobile robot and prepare you for the TP1

We will have an overview on:

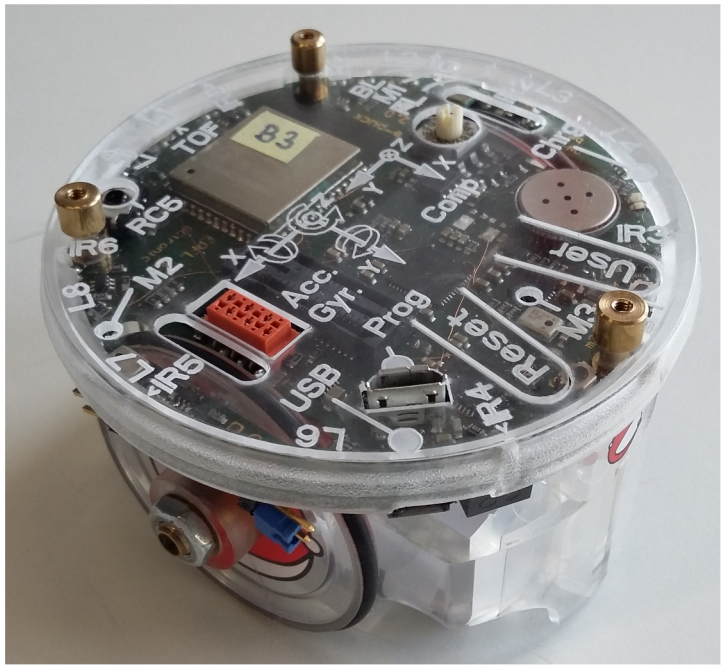
- Mechanical structure
- Hardware architecture
- Schematics for the microcontroller and GPIO
- Link between schematics and software
- Understanding about the robot as a system

The e-puck project

e-puck



e-puck (2004)

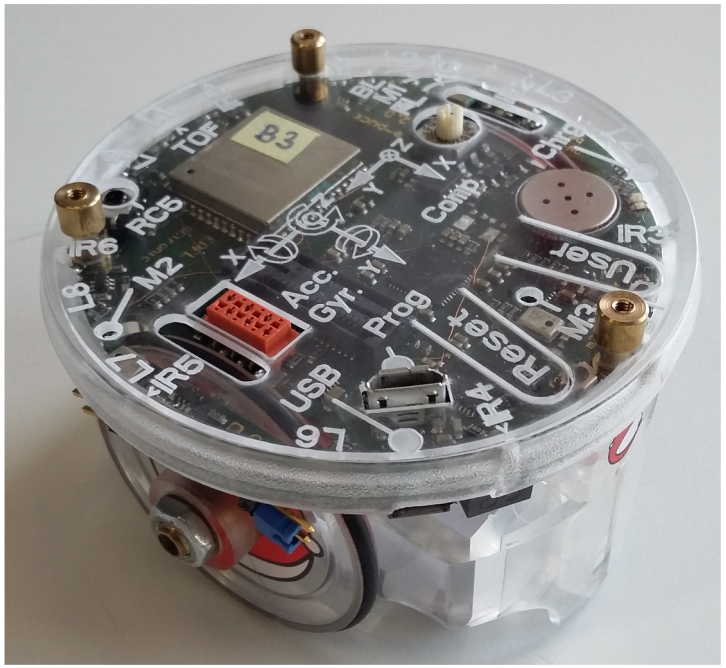


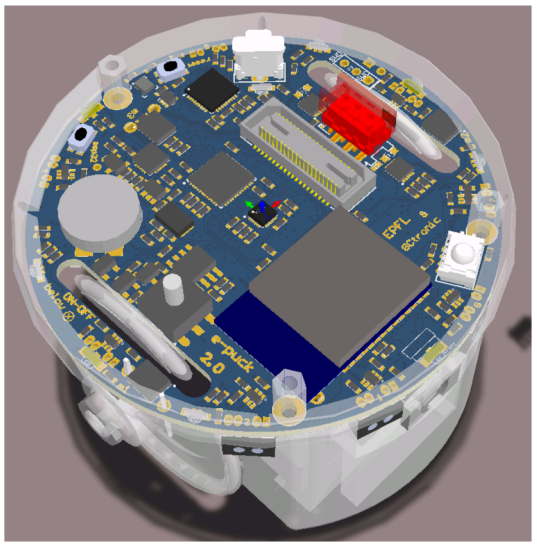
e-puck2 (2018)

The e-puck2 mobile robot

Main features:

- Cylindrical, Ø 70mm
- STM32f407 processor
- Two stepper motors
- Multiple LEDs
- Many sensors:
  - ✓Camera
  - ✓Sound
  - ✓IR proximity sensors
  - ✓IMU (Acc., Gyro., ~~Mag.~~)
  - ✓Distance sensor (ToF)
- Li-ion accumulator
- Bluetooth/Wifi wireless
- USB communication
- Onboard Debug Server (GDB)
- Open hardware



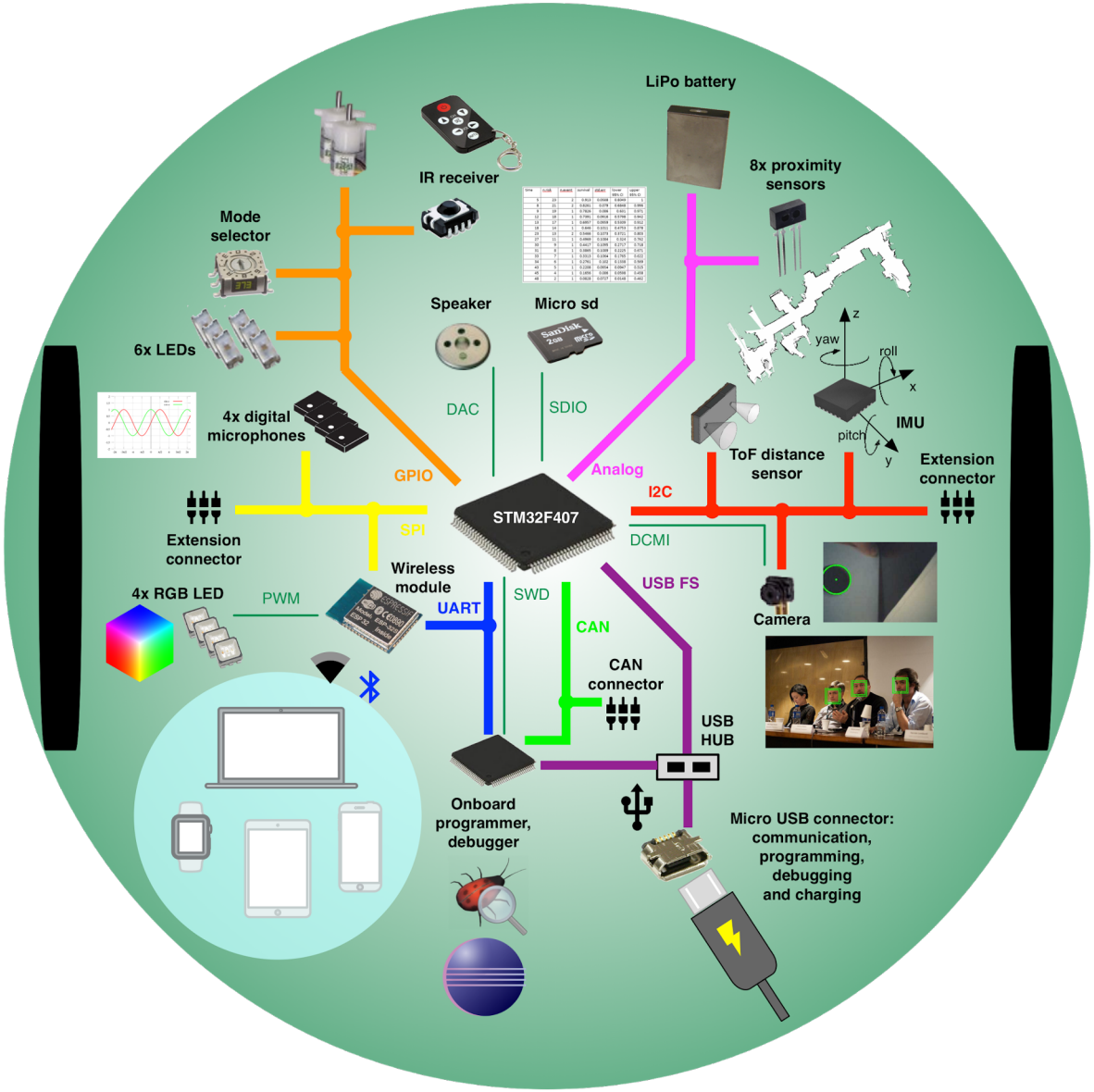


The e-puck2 open hardware license

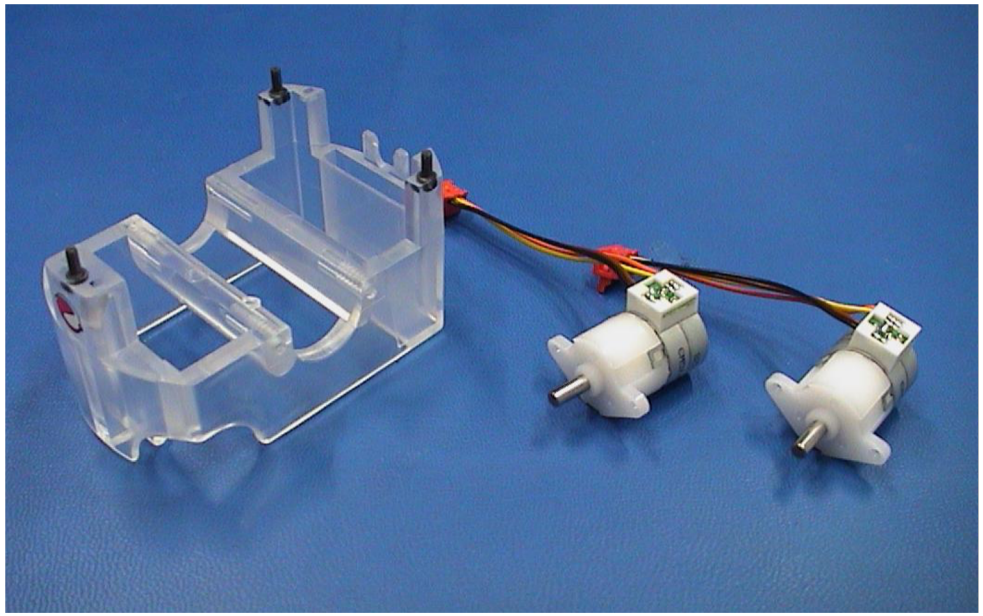
The specifications of the e-puck2 mobile robot are "open source hardware". You can redistribute them and/or modify them under the terms of the e-puck2 Robot Open Source Hardware License as published by EPFL. You should have received a copy of the EPFL e-puck2 Robot Open Source Hardware License along with these specifications; if not, write to the Ecole Polytechnique Fédérale de Lausanne (EPFL), Industrial Relations Office, Station 10, 1015 Lausanne, Switzerland.

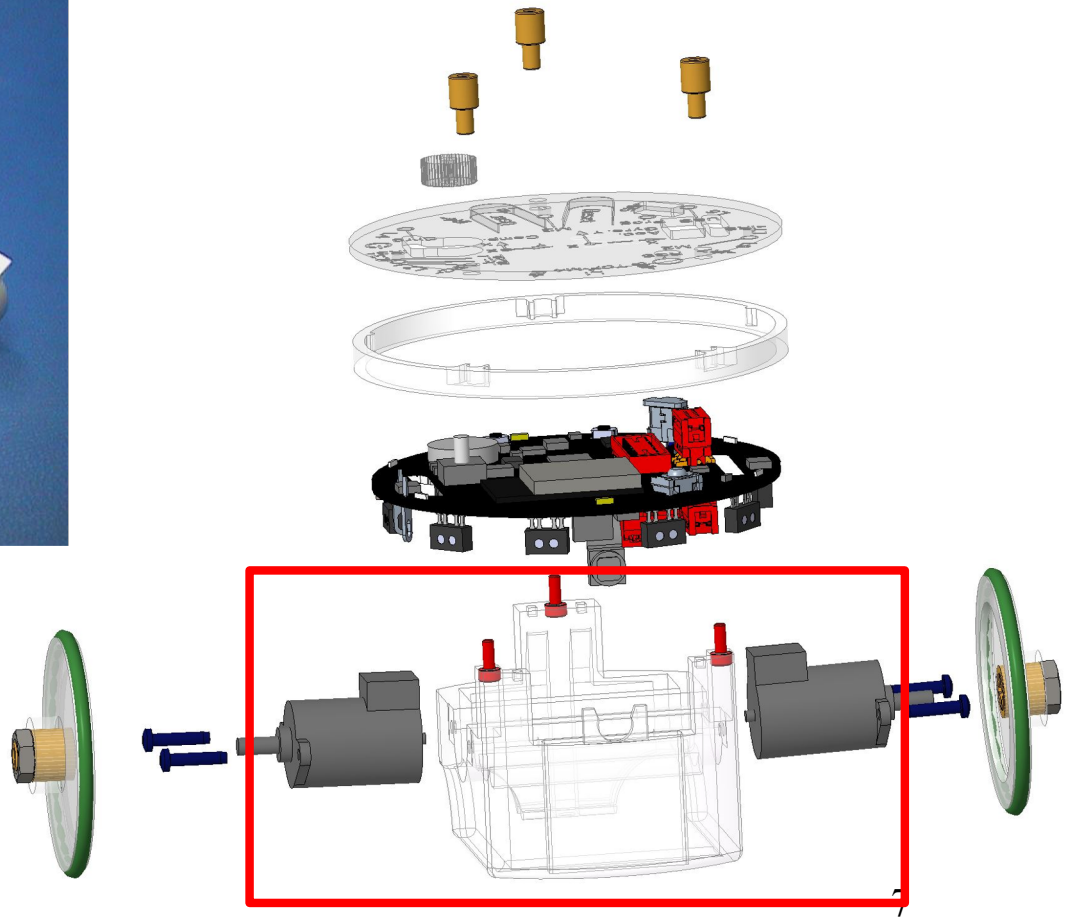
These specifications are distributed in the hope that they will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. For more details, see the EPFL e-puck2 Robot Open Source Hardware License.

e-puck2 structure

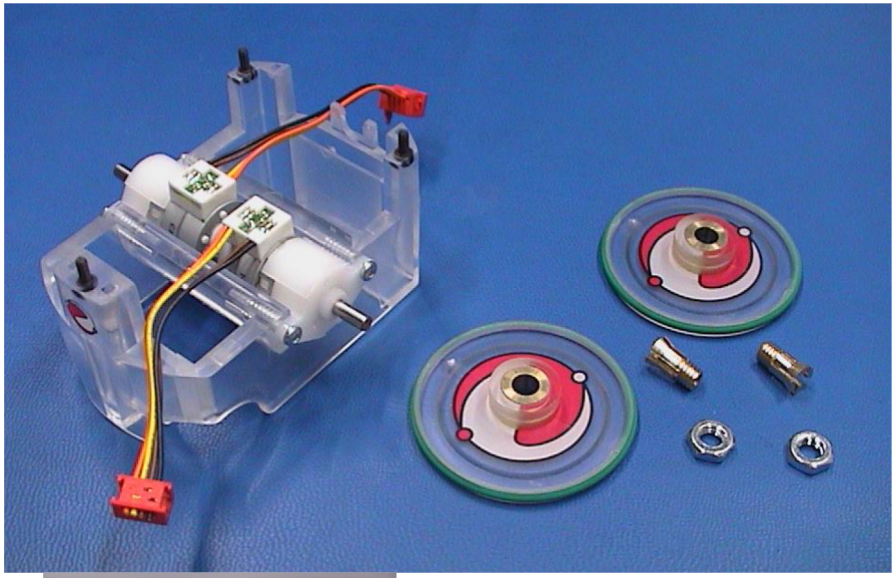


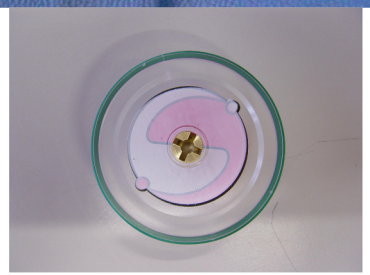
e-puck2 mobile robot mechanical structure

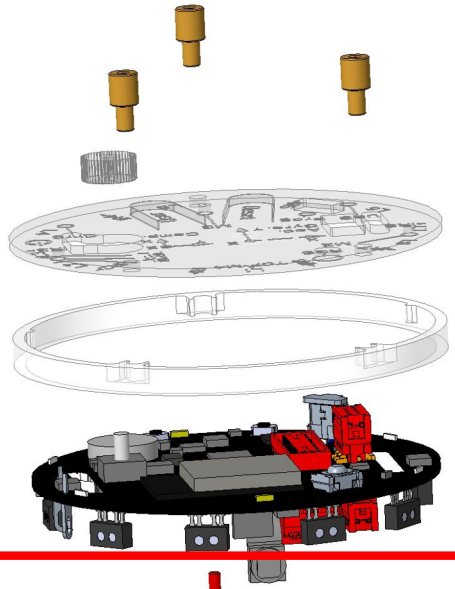


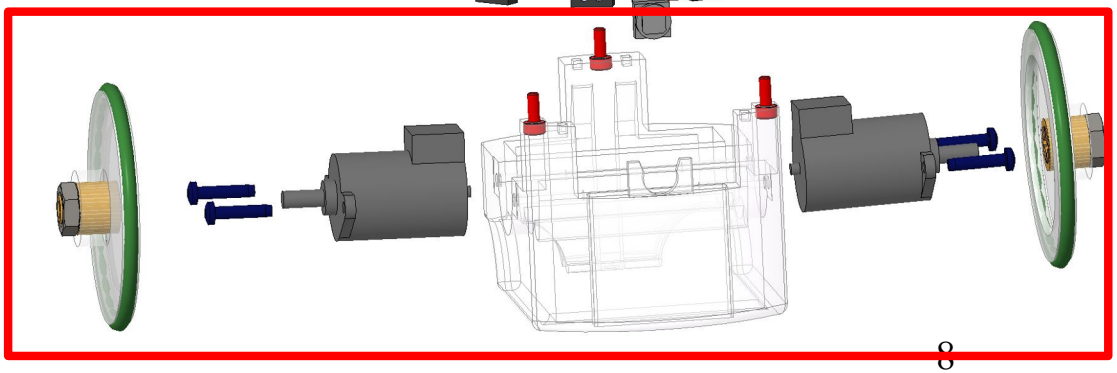


e-Puck mobile robot mechanical structure

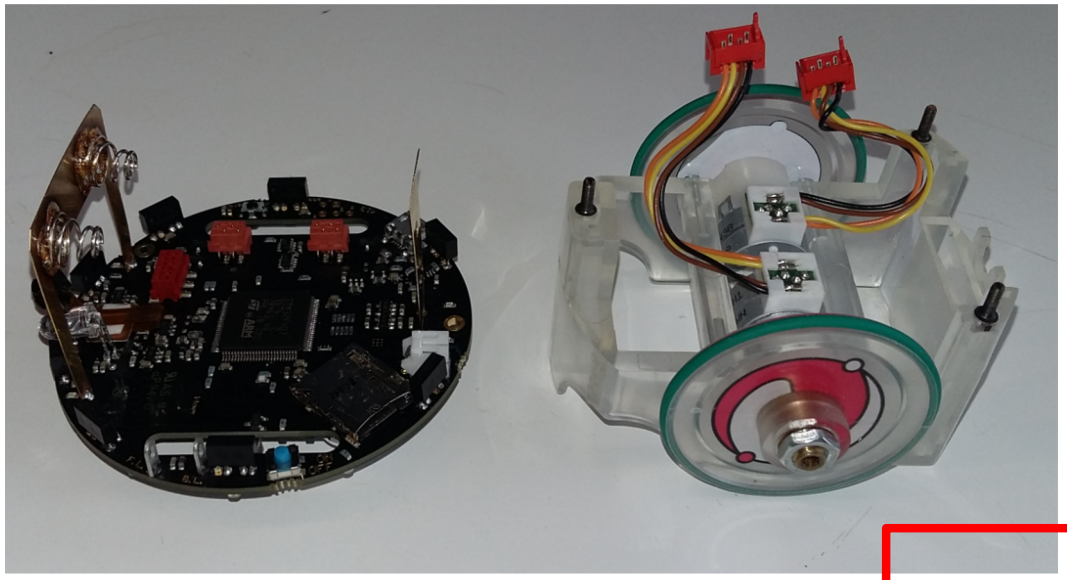


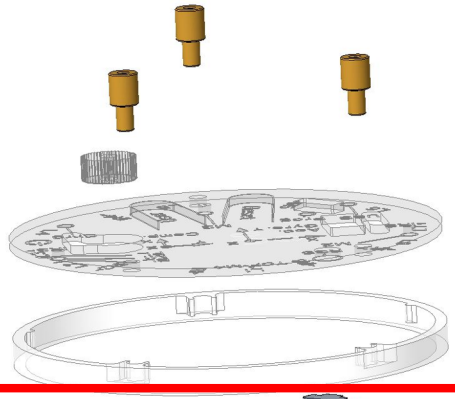


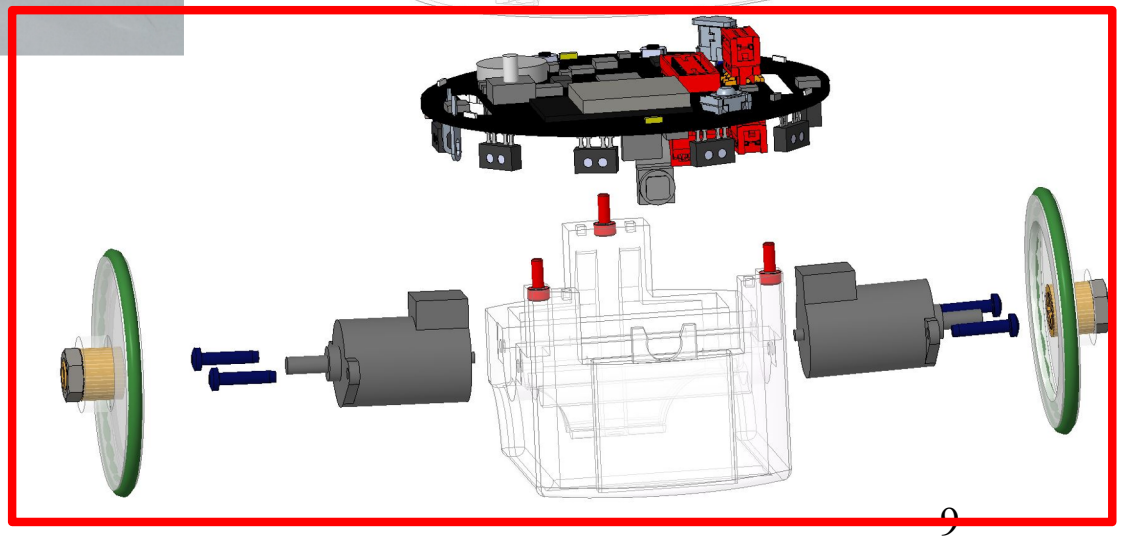




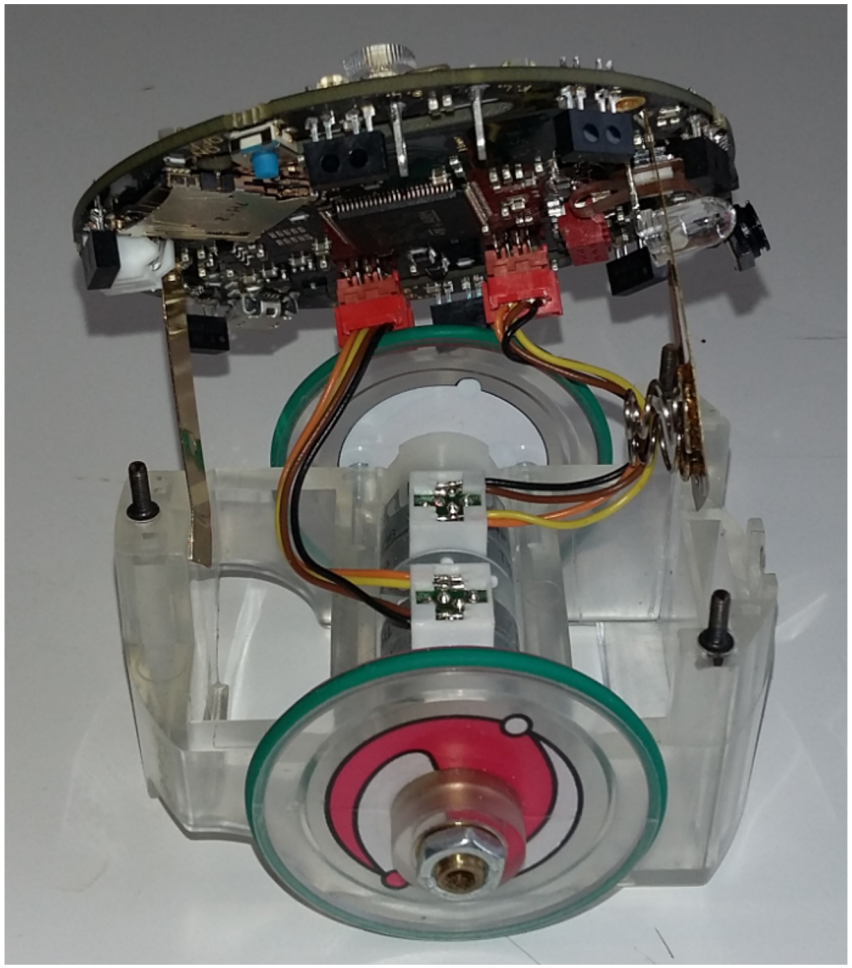
e-Puck mobile robot mechanical structure

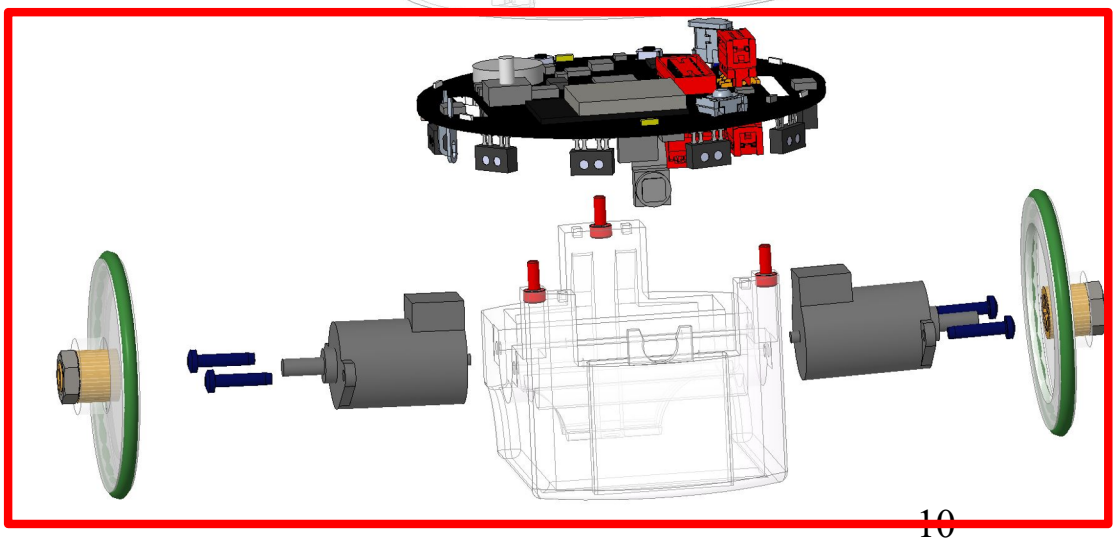




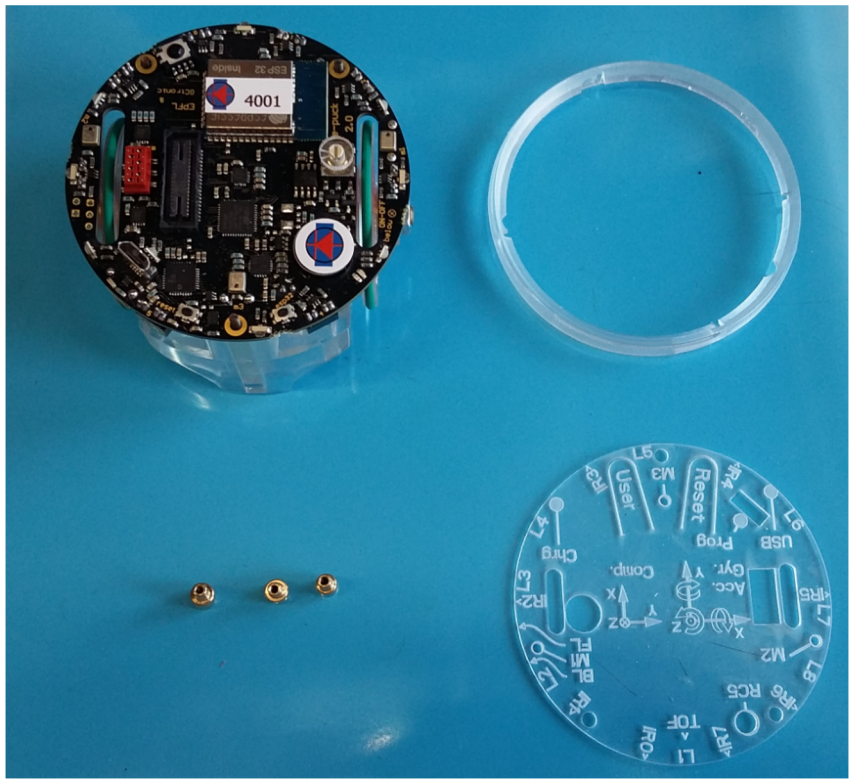


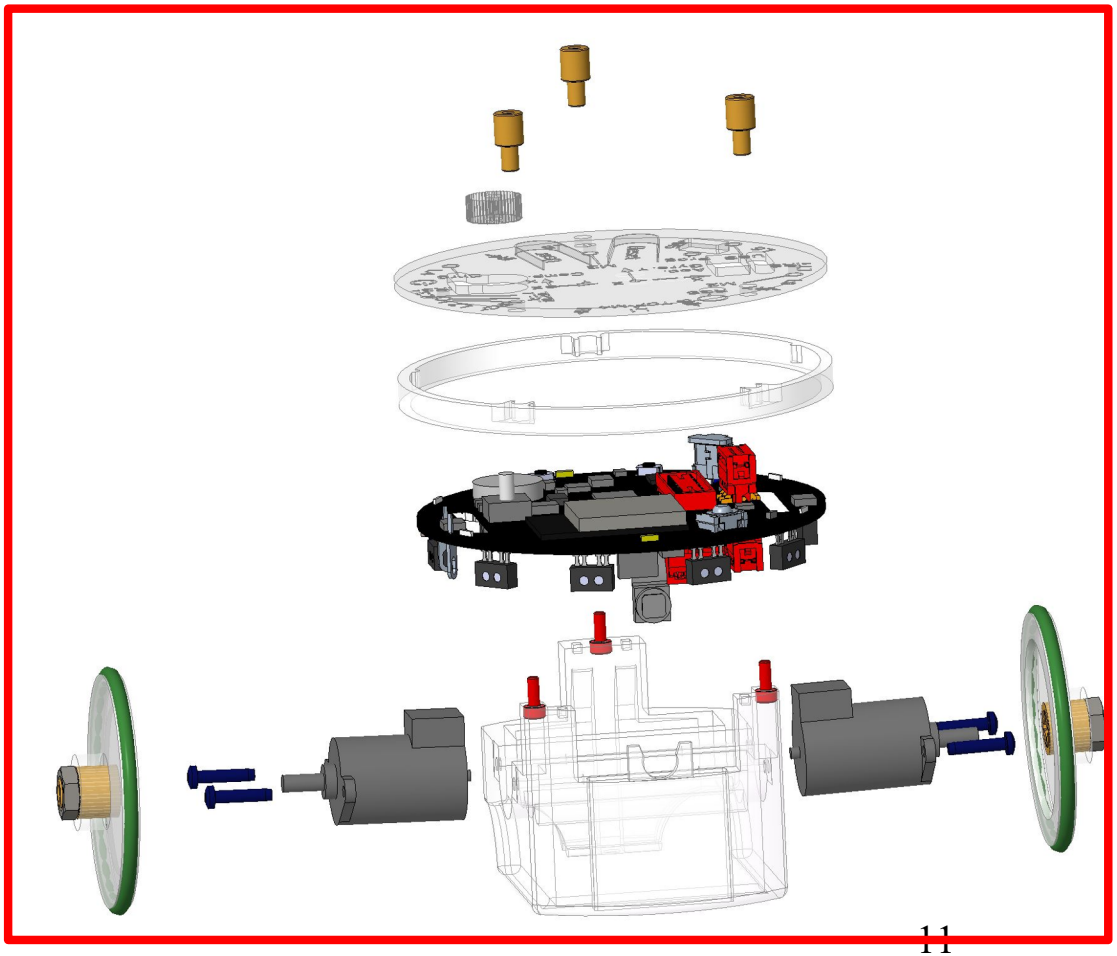
e-Puck mobile robot mechanical structure



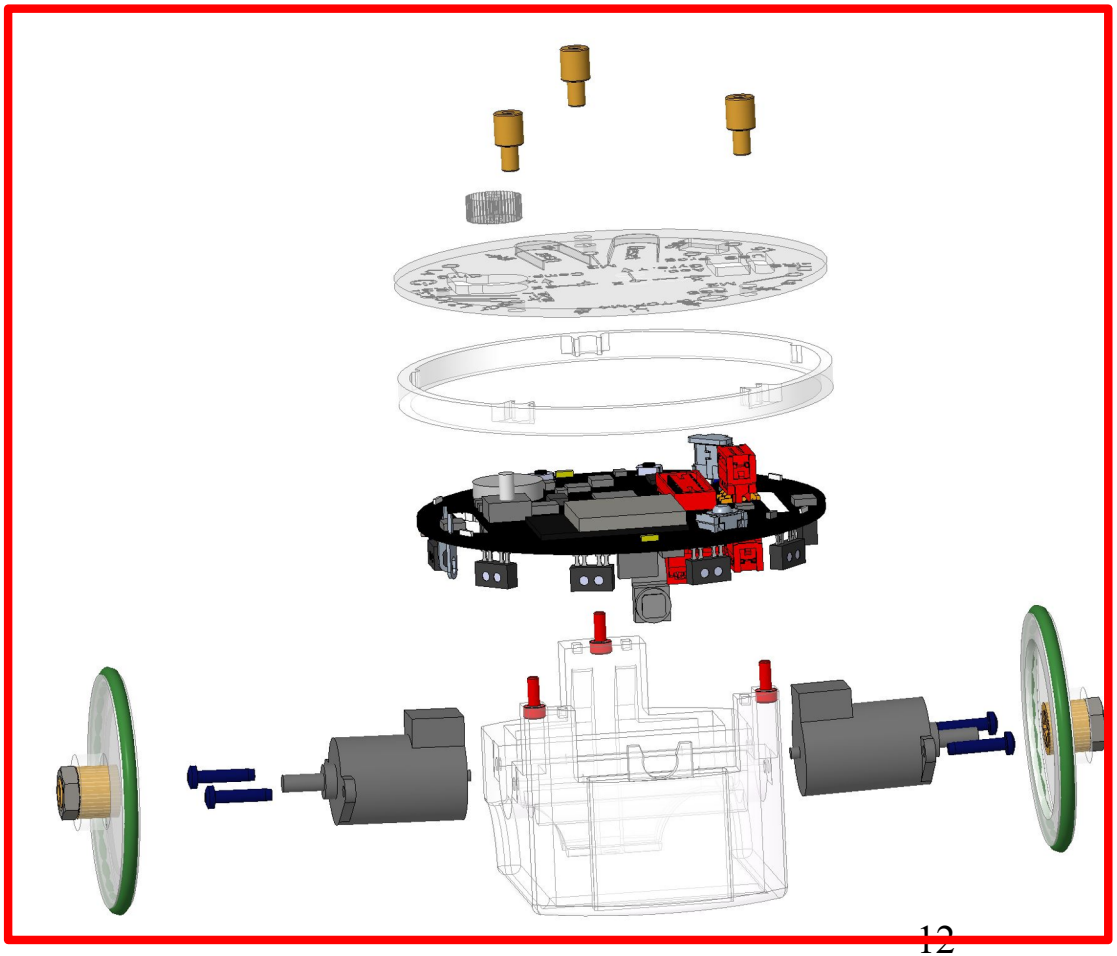


e-Puck mobile robot mechanical structure

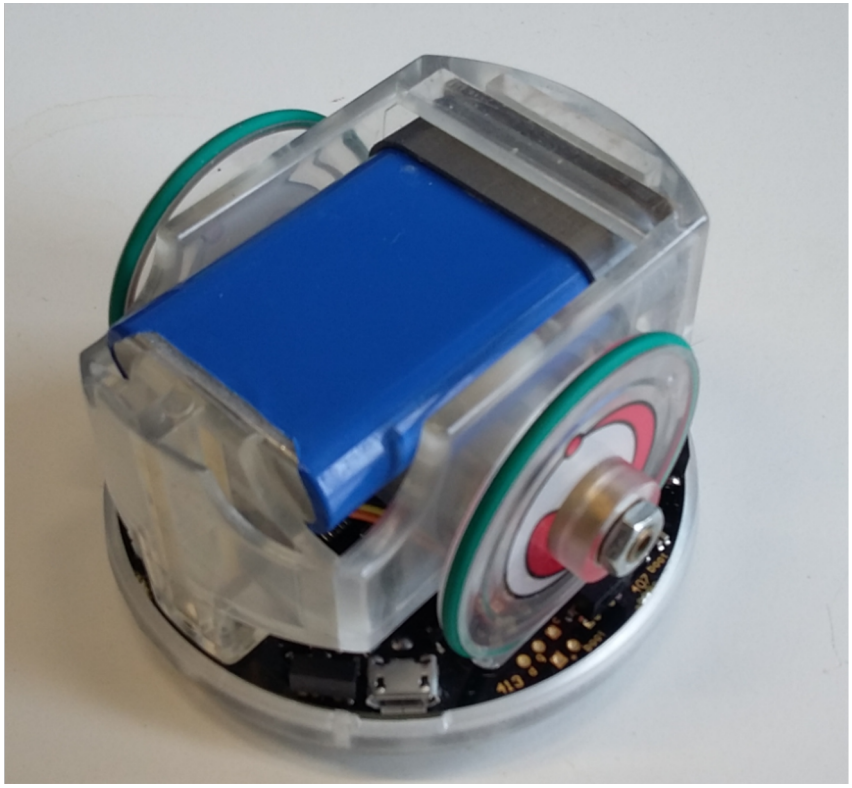


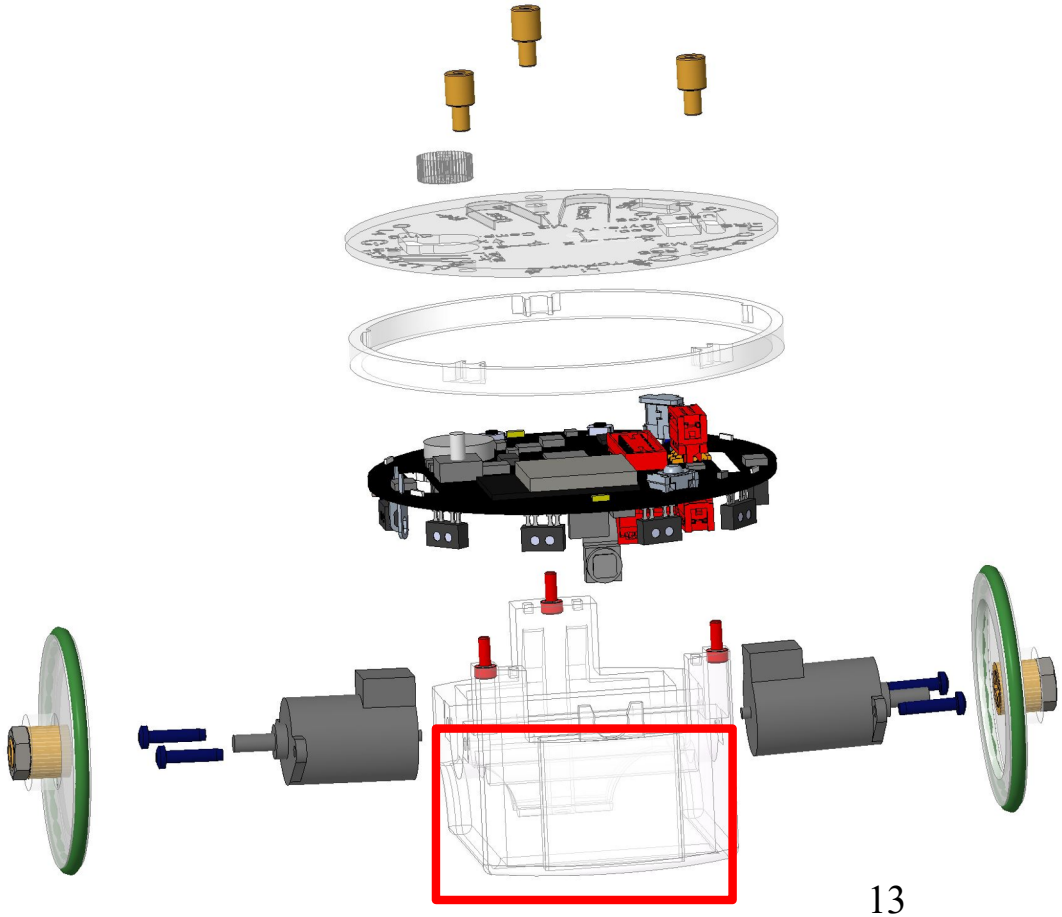


e-Puck mobile robot mechanical structure

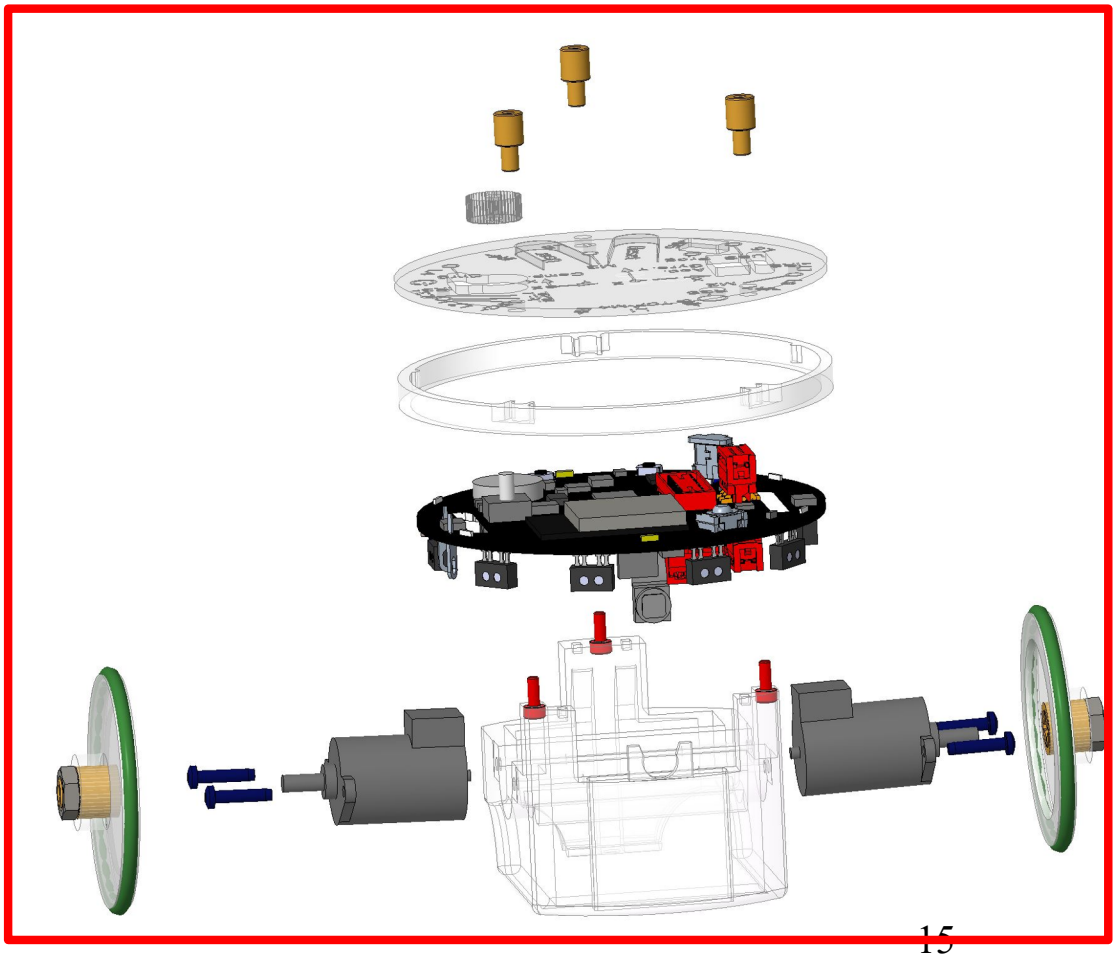


e-Puck mobile robot mechanical structure

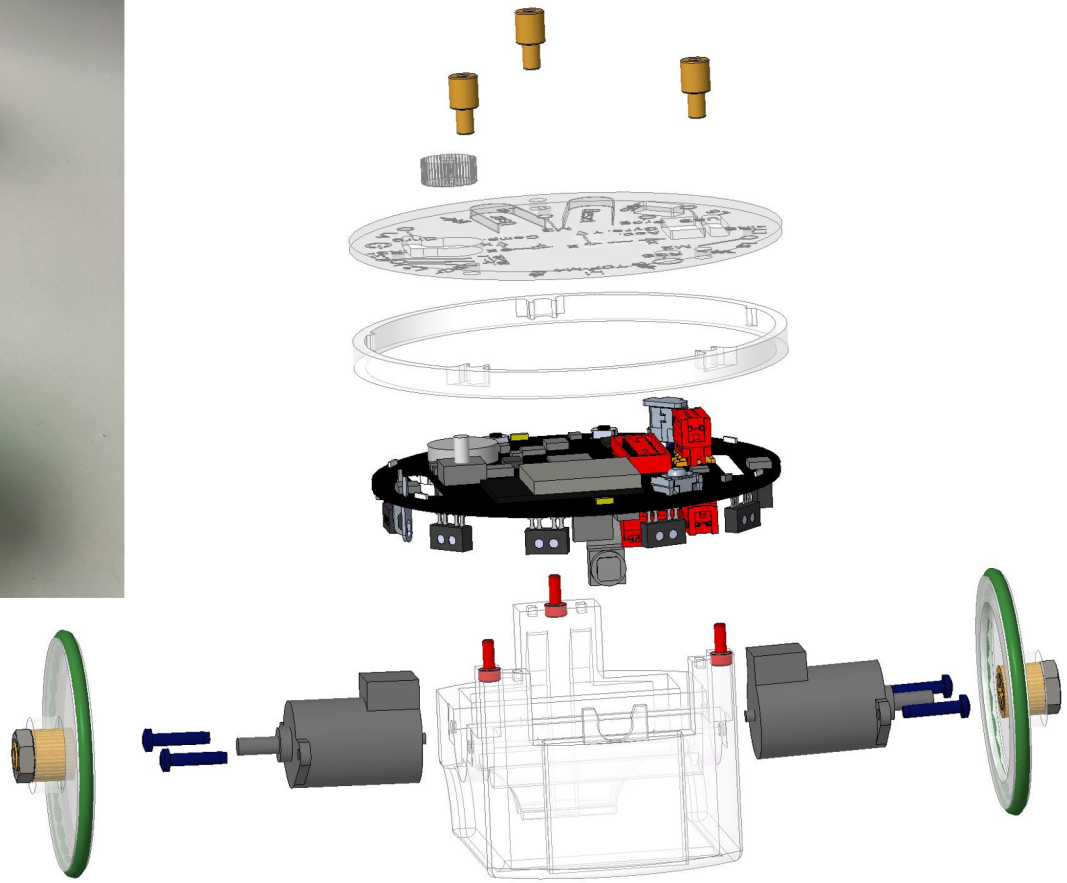




e-Puck mobile robot mechanical structure



e-Puck mobile robot mechanical structure



Documentation

The goal of the practicals is also to train you to find the information to resolve issues linked with embedded systems, electronics and programming by yourselves, using for instance the documentations of the microcontroller.

STM 32 Datasheet

Regroup the technical data of the microcontroller

life.augmented

STM32F405xx
STM32F407xx

ARM Cortex-M4 32b MCU+FPU, 210DMIPS, up to 1MB Flash/192+4KB RAM, USB OTG HS/FS, Ethernet, 17 TIMs, 3 ADCs, 15 comm. interfaces & camera

Datasheet - production data

Features

- Core: ARM® 32-bit Cortex®-M4 CPU with FPU, Adaptive real-time accelerator (ART Accelerator™) allowing 0-wait state execution from Flash memory, frequency up to 168 MHz, memory protection unit, 210 DMIPS/ 1.25 DMIPS/MHz (Dhrystone 2.1), and DSP instructions
- Memories
- Up to 1 Mbyte of Flash memory
- Up to 192+4 Kbytes of SRAM including 64-Kbyte of CCM (core coupled memory) data RAM
- Flexible static memory controller supporting Compact Flash, SRAM, PSRAM, NOR and NAND memories
- LCD parallel interface, 8080/6800 modes
- Clock, reset and supply management
  - 1.8 V to 3.6 V application supply and I/Os
  - POR, PDR, PVD and BOR
  - 4-to-26 MHz crystal oscillator
  - Internal 16 MHz factory-trimmed RC (1% accuracy)
  - 32 kHz oscillator for RTC with calibration
  - Internal 32 kHz RC with calibration
- Low-power operation
  - Sleep, Stop and Standby modes
  - V_{BAT} supply for RTC, 20×32 bit backup registers + optional 4 KB backup SRAM
- 3×12-bit, 2.4 MSPS A/D converters: up to 24 channels and 7.2 MSPS in triple interleaved mode
- 2×12-bit D/A converters
- General-purpose DMA: 16-stream DMA controller with FIFOs and burst support

LQFP64 (10 × 10 mm)
LQFP100 (14 × 14 mm)
LQFP144 (20 × 20 mm)
LQFP176 (24 × 24 mm)

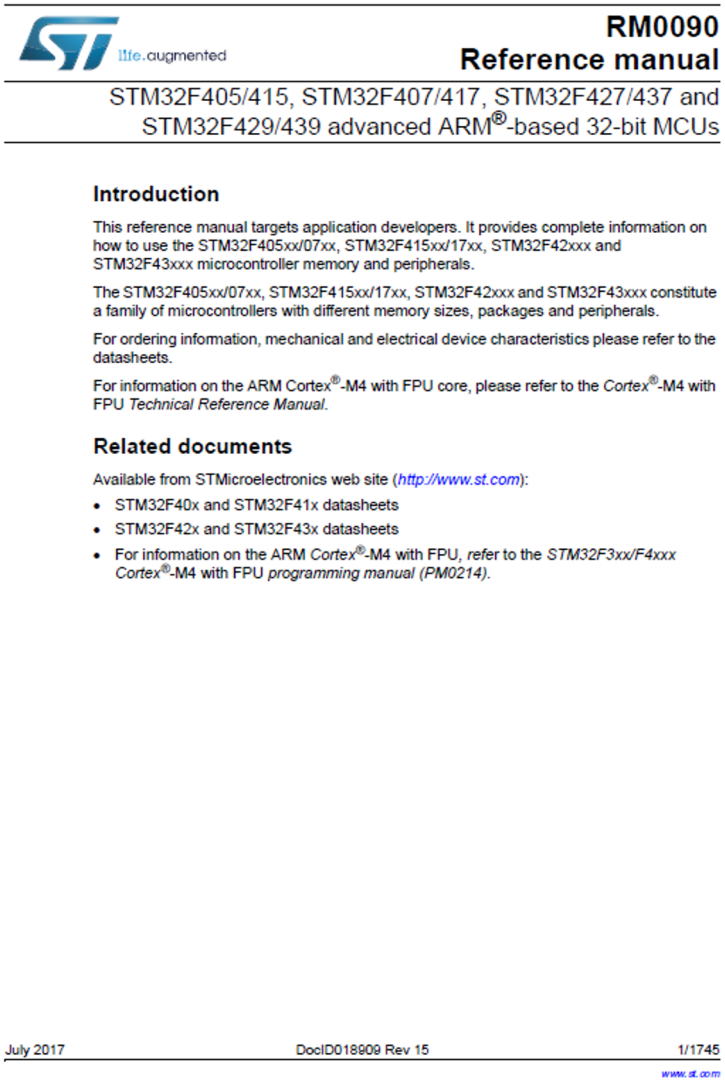
WLCSP90
(4.223x3.969 mm)

UFBGA176
(10 × 10 mm)

- Up to 17 timers: up to twelve 16-bit and two 32-bit timers up to 168 MHz, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
- Debug mode
  - Serial wire debug (SWD) & JTAG interfaces
  - Cortex-M4 Embedded Trace Macrocell™
- Up to 140 I/O ports with interrupt capability
  - Up to 136 fast I/Os up to 84 MHz
  - Up to 138 5 V-tolerant I/Os
- Up to 15 communication interfaces
  - Up to 3 × I^2C interfaces (SMBus/PMBus)
  - Up to 4 USARTs/2 UARTs (10.5 Mbit/s, ISO 7816 interface, LIN, IrDA, modem control)
  - Up to 3 SPIs (42 Mbits/s), 2 with muxed full-duplex I^2S to achieve audio class accuracy via internal audio PLL or external clock
  - 2 × CAN interfaces (2.0B Active)
  - SDIO interface
- Advanced connectivity
  - USB 2.0 full-speed device/host/OTG controller with on-chip PHY
  - USB 2.0 high-speed/full-speed device/host/OTG controller with dedicated DMA, on-chip full-speed PHY and ULPI
  - 10/100 Ethernet MAC with dedicated DMA: supports IEEE 1588v2 hardware, MII/RMII

STM 32 Reference Manual

Application and programming (memory, registers, peripherals) of the microcontroller



life.augmented

RM0090
Reference manual

STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced ARM®-based 32-bit MCUs

Introduction

This reference manual targets application developers. It provides complete information on how to use the STM32F405xx/07xx, STM32F415xx/17xx, STM32F42xxx and STM32F43xxx microcontroller memory and peripherals.

The STM32F405xx/07xx, STM32F415xx/17xx, STM32F42xxx and STM32F43xxx constitute a family of microcontrollers with different memory sizes, packages and peripherals.

For ordering information, mechanical and electrical device characteristics please refer to the datasheets.

For information on the ARM Cortex®-M4 with FPU core, please refer to the *Cortex®-M4 with FPU Technical Reference Manual*.

Related documents

Available from STMicroelectronics web site (*http://www.st.com*):

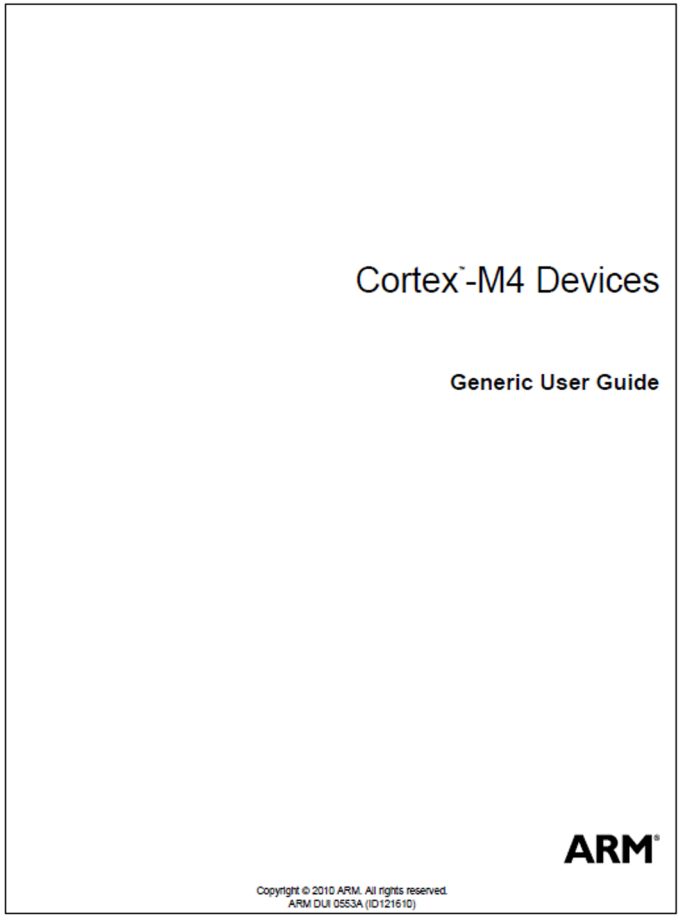
- STM32F40x and STM32F41x datasheets
- STM32F42x and STM32F43x datasheets
- For information on the ARM *Cortex®-M4 with FPU, refer to the STM32F3xx/F4xxx Cortex®-M4 with FPU programming manual (PM0214).*

July 2017 DocID018909 Rev 15 1/1745

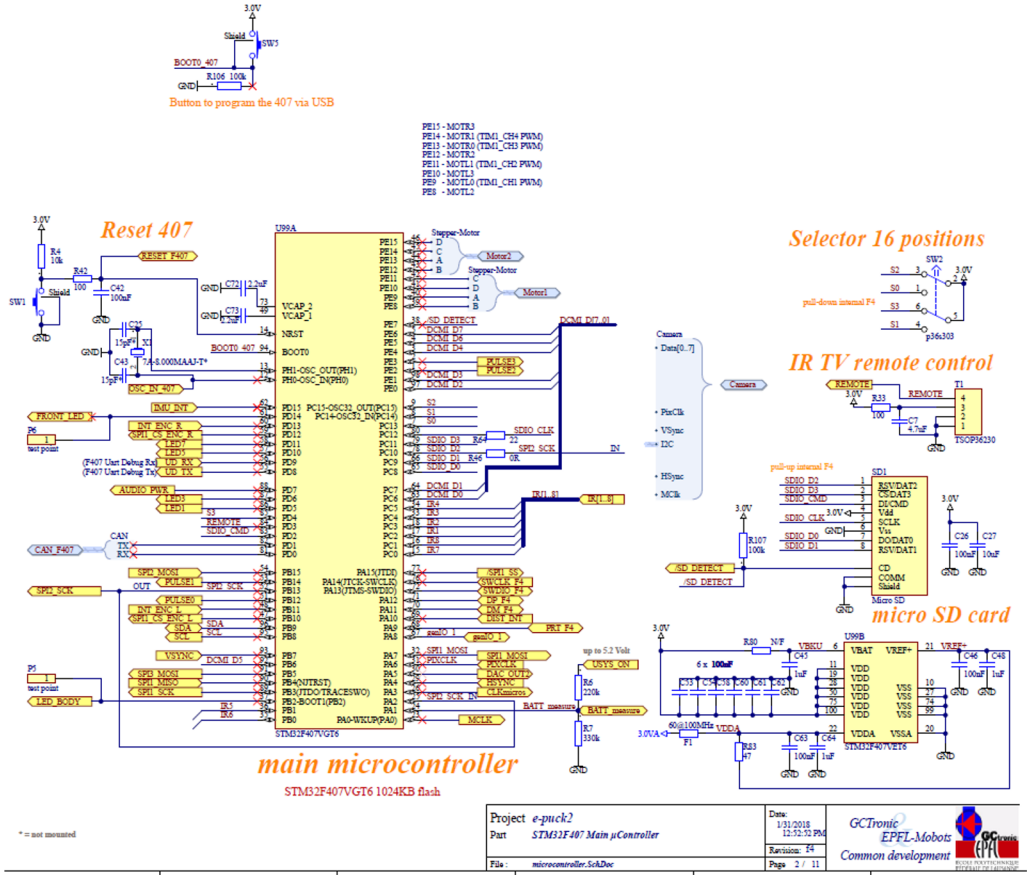
www.st.com

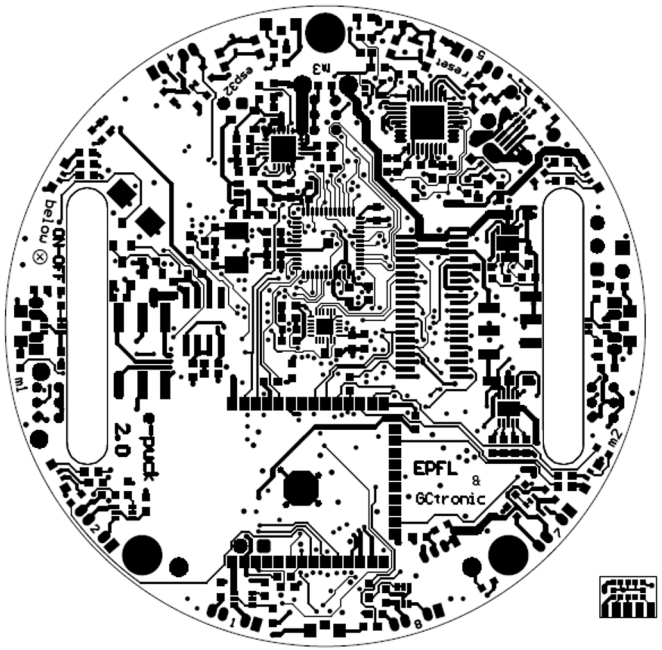
Cortex-M4 Generic User Guide

Architecture of the ARM processor, programming of the processor (registers, assembly instructions)

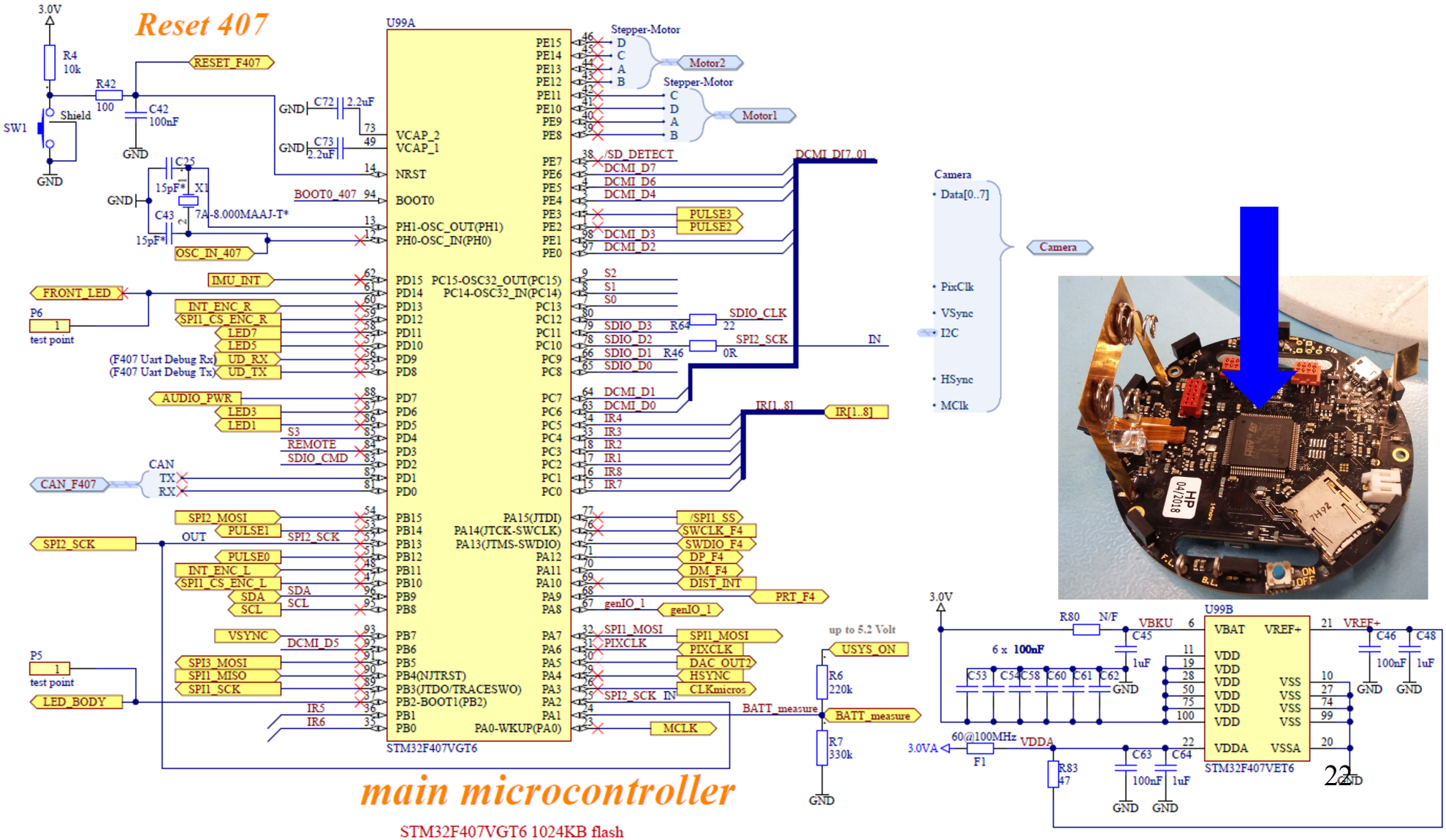


Dossier Electronique



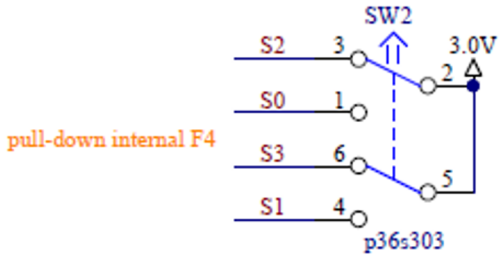


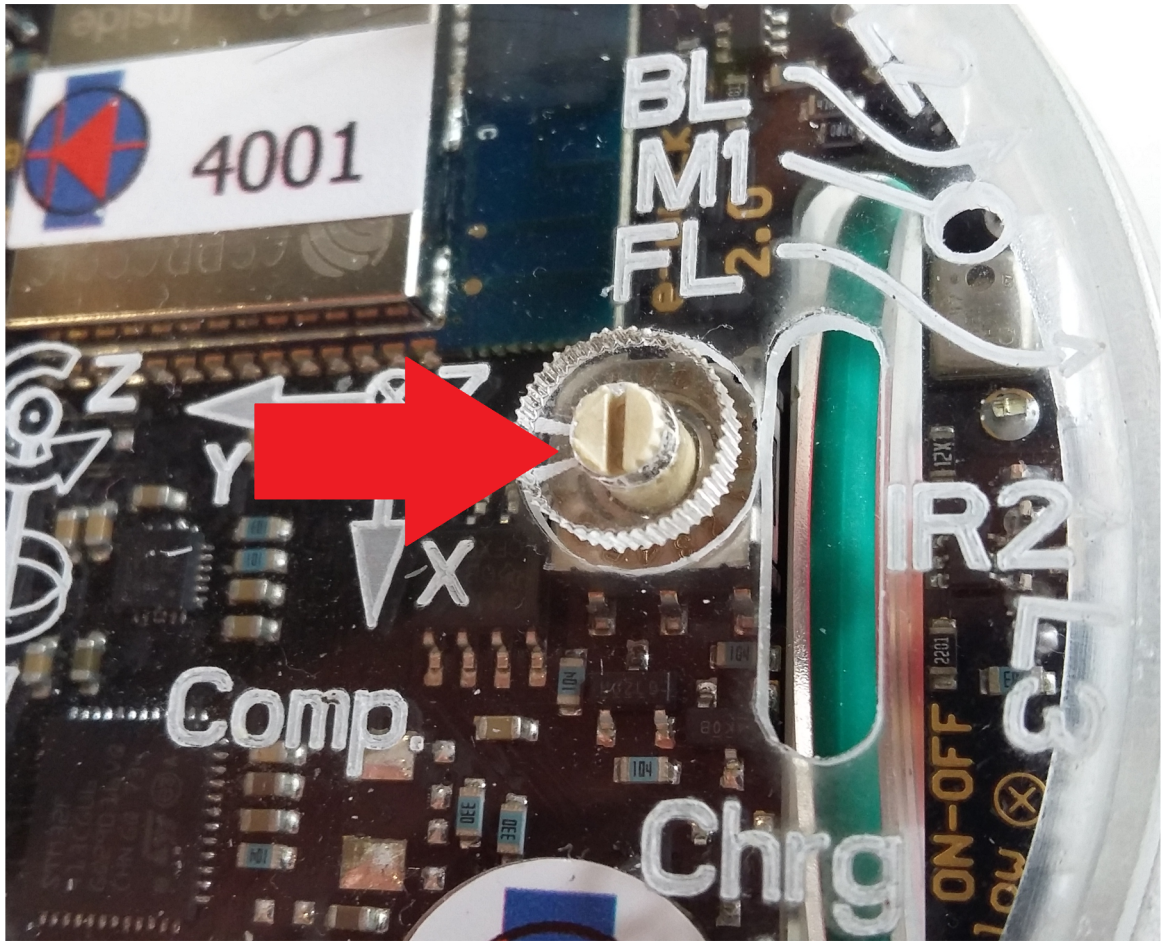
Schematics: Microcontroller



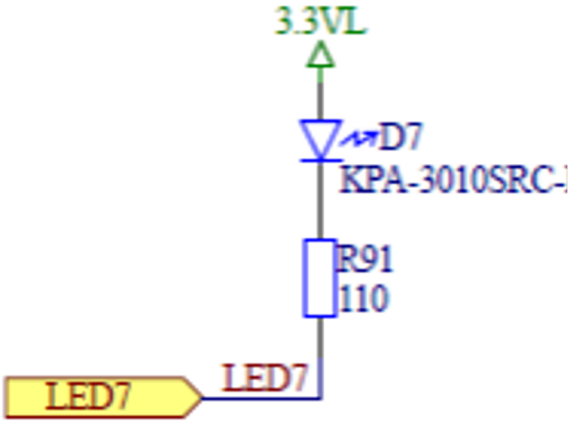
Schematics: Selector

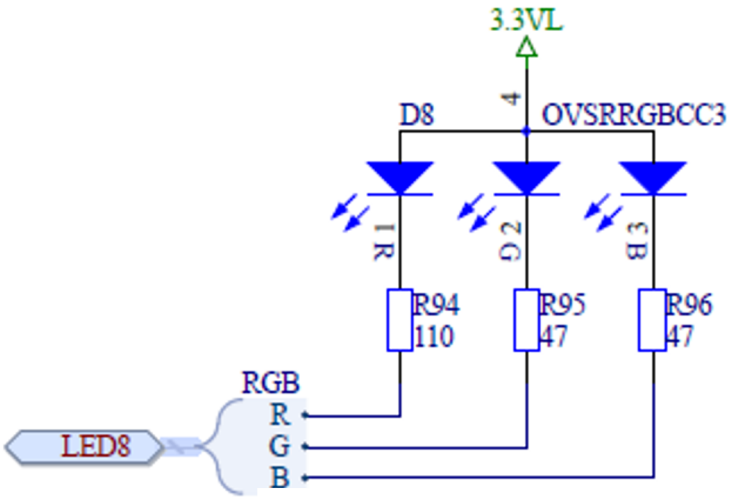
Selector 16 positions

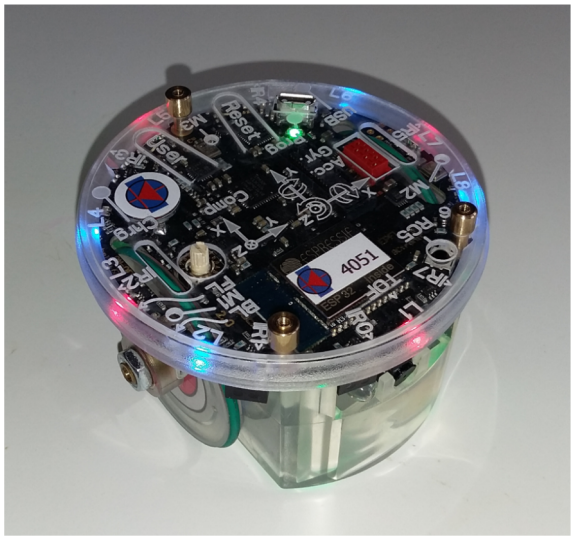


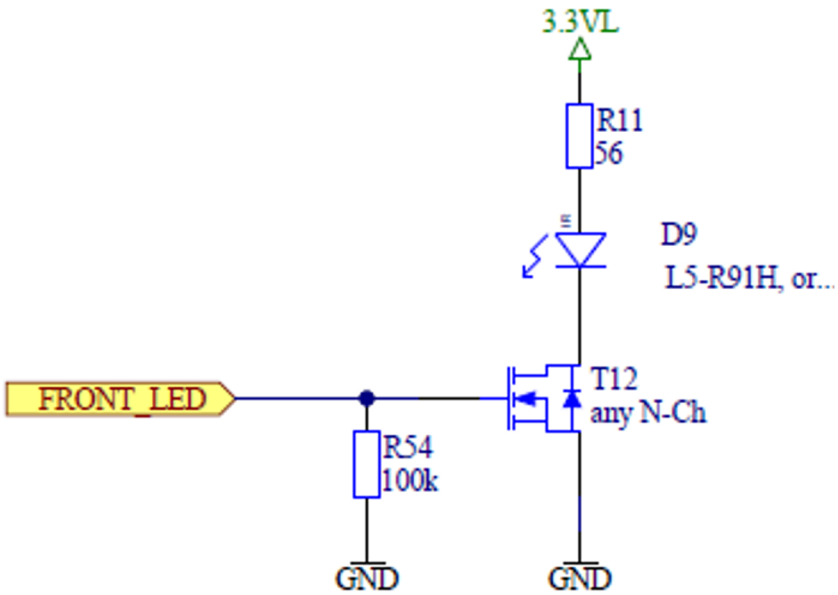


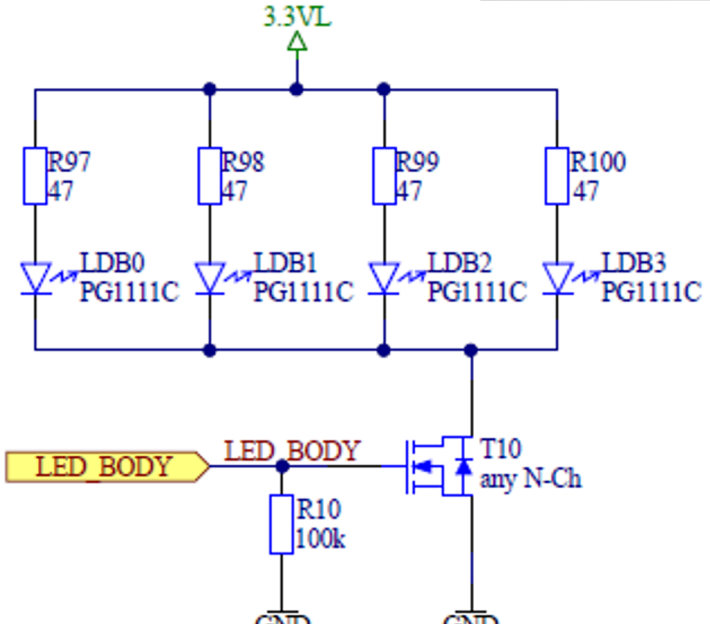
Schematics: LEDs (different types)



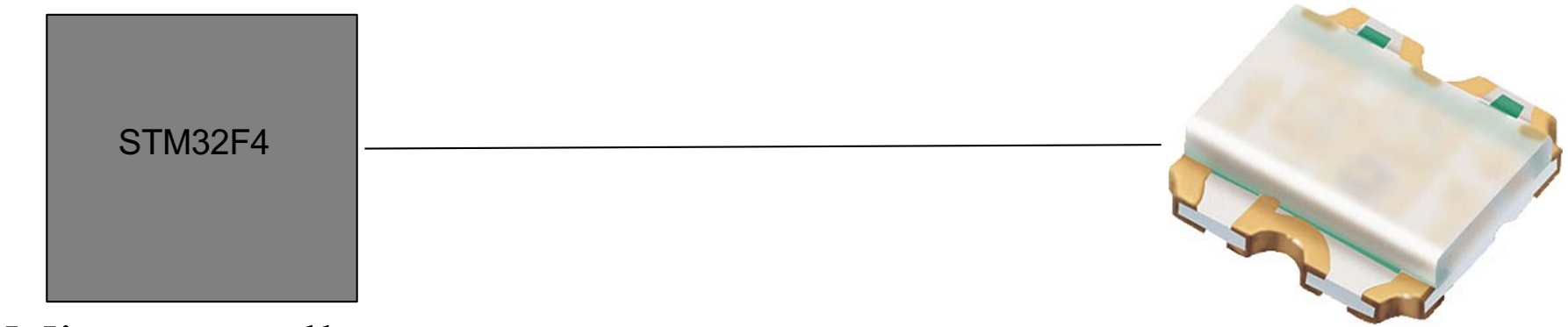




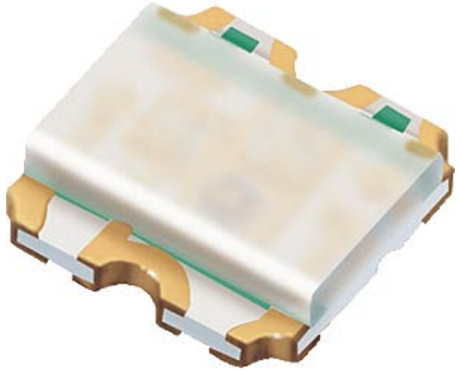


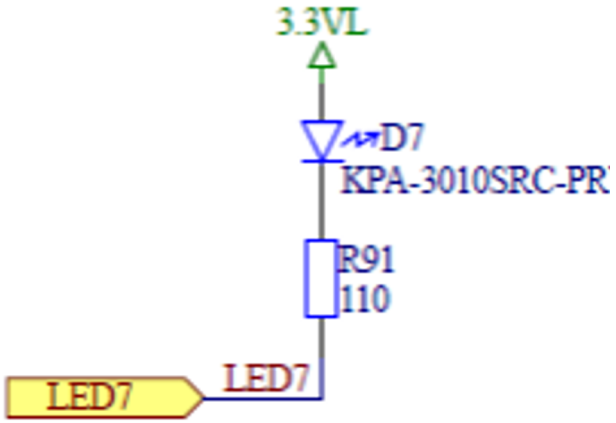


Interface

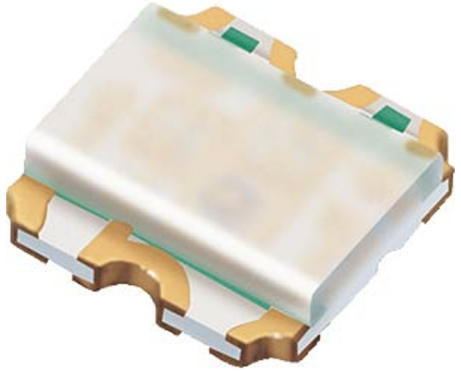


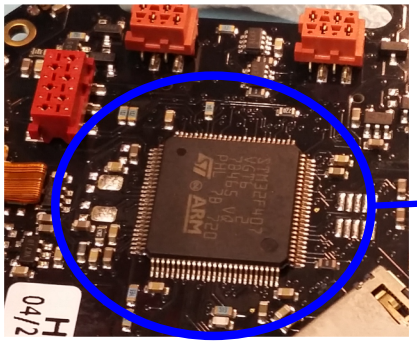
Interface: Schematic

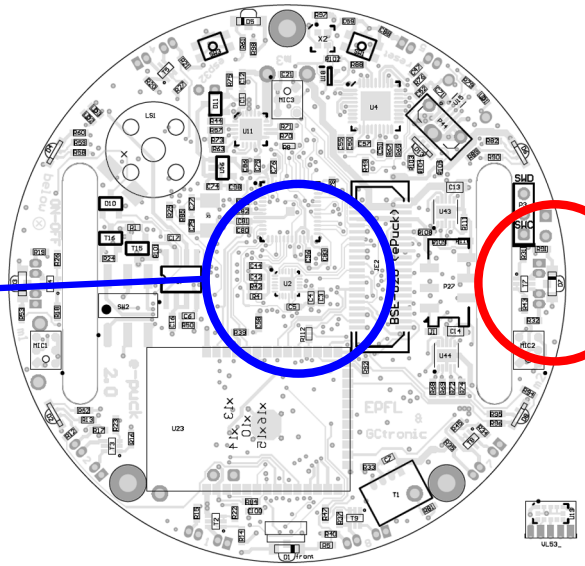




Interface: Physical

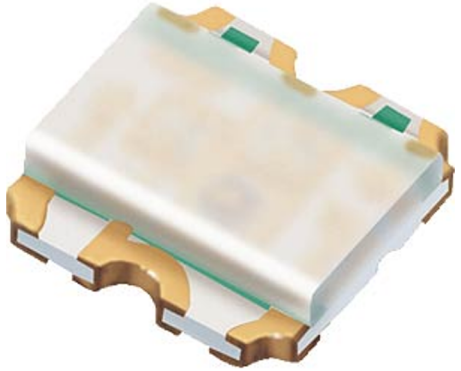


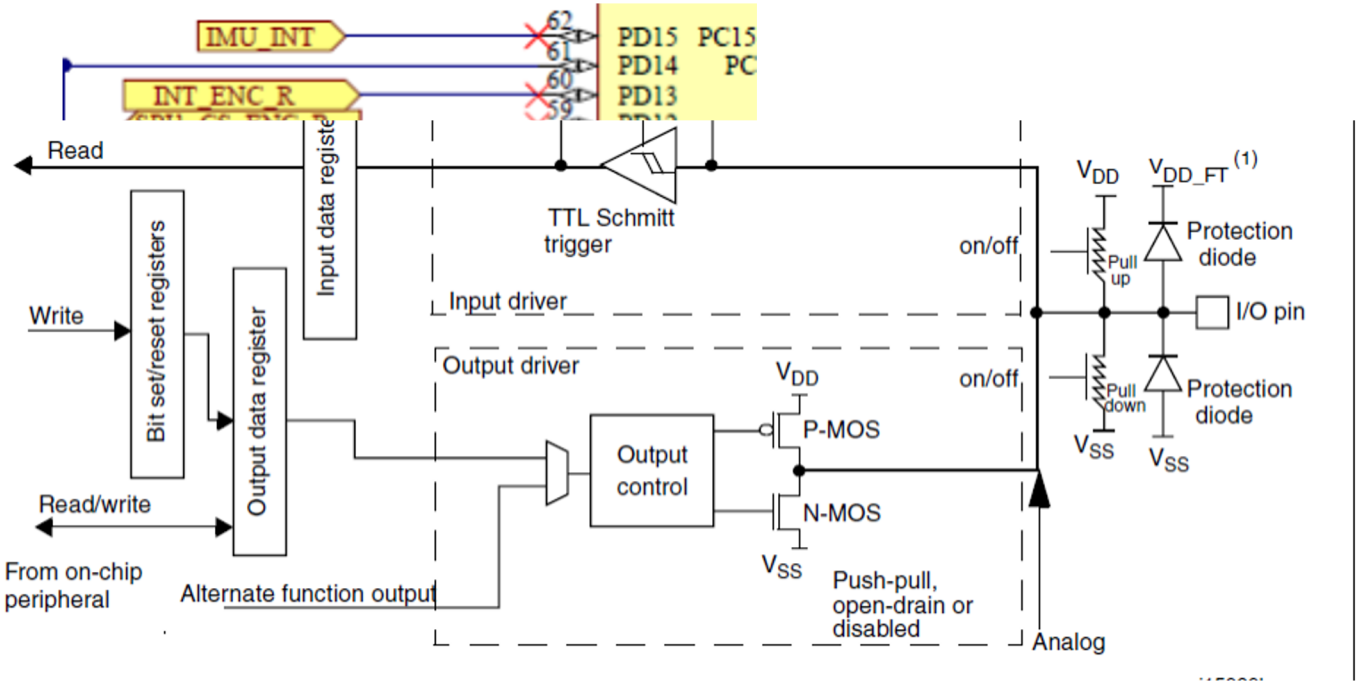


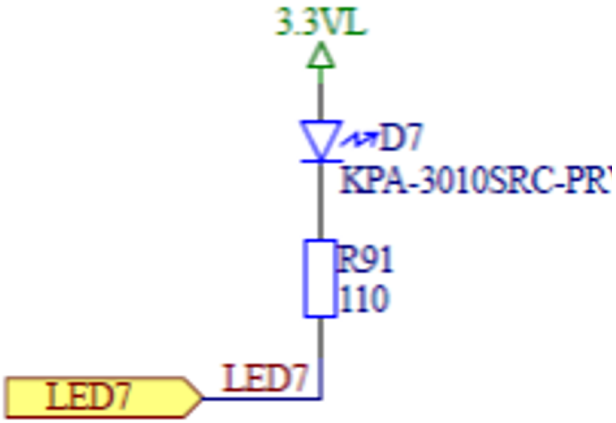


Interface: microcontroller / hardware

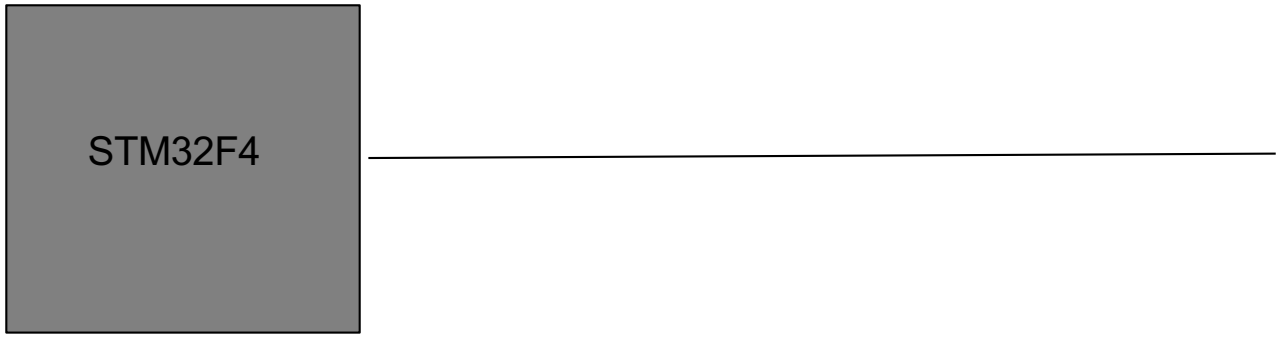
STM32F4

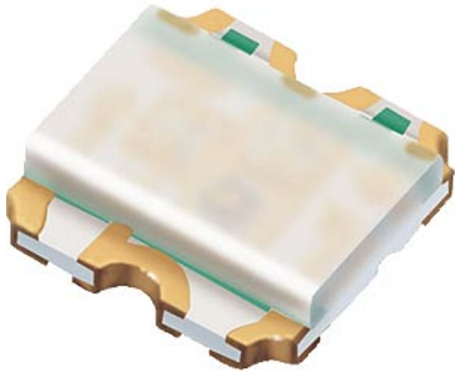


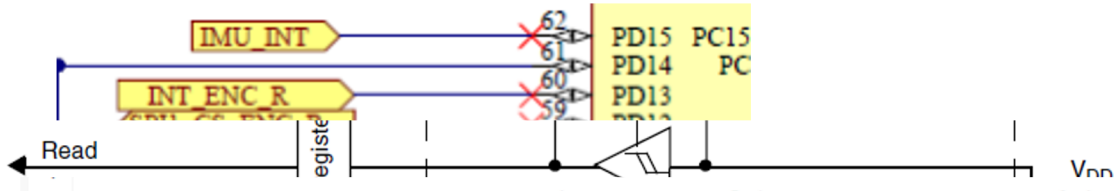




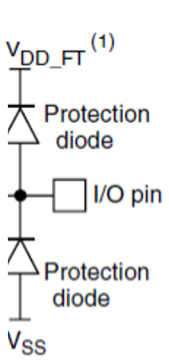
Interface: microcontroller / firmware

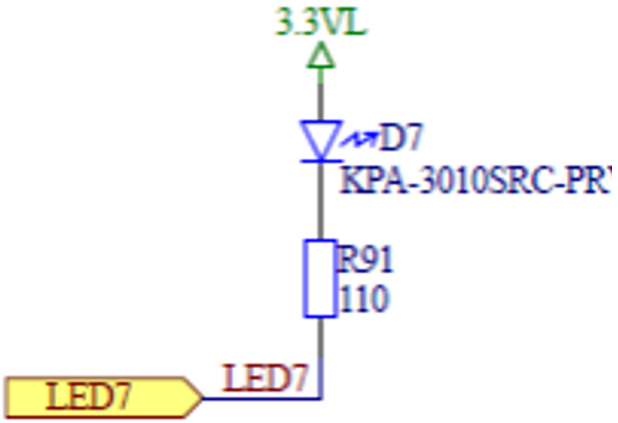




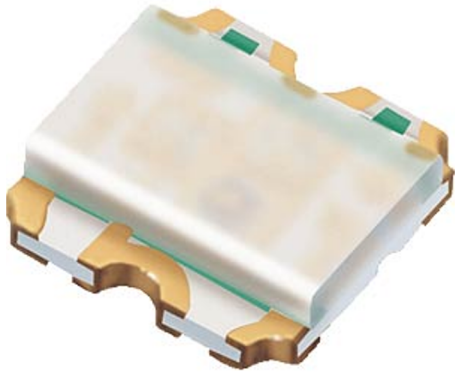



```
5  void gpio_config_output_opendrain(GPIO_TypeDef *port, unsigned int pin)
6  {
7    // Output type open-drain : OTy = 1
8    port->OTYPER |= (1 << pin);
9
10   // Output data low : ODRy = 0
11   port->ODR &= ~(1 << pin);
12
13   // Floating, no pull-up/down : PUPDRy = 00
14   port->PUPDR &= ~(3 << (pin * 2));
15
16   // Output speed highest : OSPEEDRy = 11
17   port->OSPEEDR |= (3 << (pin * 2));
18
19   // Output mode : MODERy = 01
20   port->MODER = (port->MODER & ~(3 << (pin * 2))) | (1 << (pin * 2));
```

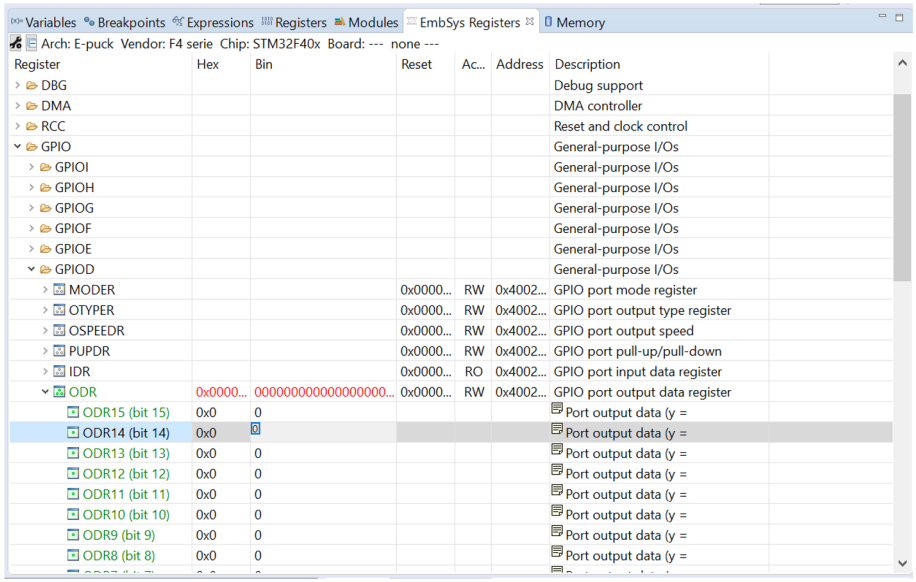




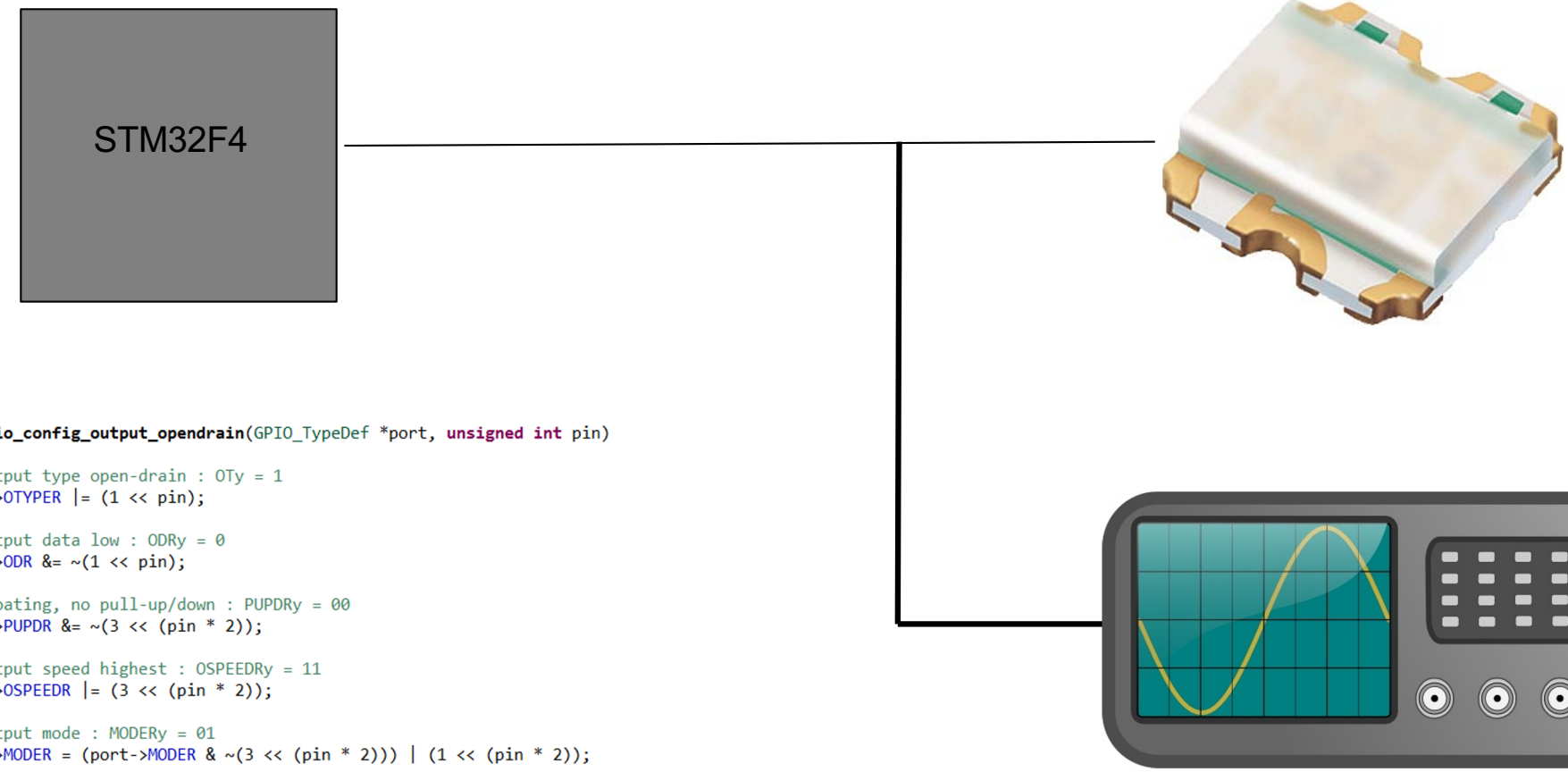
Tools: Debug



```
5  void gpio_config_output_opendrain(GPIO_TypeDef *port, unsigned int pin)
6  {
7    // Output type open-drain : OTy = 1
8    port->OTYPER |= (1 << pin);
9
10   // Output data low : ODRy = 0
11   port->ODR &= ~(1 << pin);
12
13   // Floating, no pull-up/down : PUPDRy = 00
14   port->PUPDR &= ~(3 << (pin * 2));
15
16   // Output speed highest : OSPEEDRy = 11
17   port->OSPEEDR |= (3 << (pin * 2));
18
19   // Output mode : MODERy = 01
20   port->MODER = (port->MODER & ~(3 << (pin * 2))) | (1 << (pin * 2));
21 }
```



Tools: Oscilloscope




```
5  void gpio_config_output_opendrain(GPIO_TypeDef *port, unsigned int pin)
6  {
7    // Output type open-drain : OTy = 1
8    port->OTYPER |= (1 << pin);
9
10   // Output data low : ODRy = 0
11   port->ODR &= ~(1 << pin);
12
13   // Floating, no pull-up/down : PUPDRy = 00
14   port->PUPDR &= ~(3 << (pin * 2));
15
16   // Output speed highest : OSPEEDRy = 11
17   port->OSPEEDR |= (3 << (pin * 2));
18
19   // Output mode : MODERy = 01
20   port->MODER = (port->MODER & ~(3 << (pin * 2))) | (1 << (pin * 2));
21 }
```

TP1: Structure

- Getting hand on the Eclipse Environment and e-puck2
- Configuring GPIOs in C language in output/input
- Configuring Timer/Interrupt in C language
- Measurements of the pin signal

Enjoy TP1!