STM hands-on session

Paolo Burgio
paolo.burgio@unimore.it

UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

High Performance
Real Time Lab

" Programming is a skill best acquired by practice and example rather than from books.

ALAN TURING

Our guy (IoT node)

2x USB
(Use this one!)

SIM/Wifi

General Purpose
I/O ports (GPIO)

Reset ;)

Core SoC

- 32-bit ARM Cortex
-   1MiB Flash mem
-   128kB SRAM

General P
I/O ports

3

Software

Micro-kernel

› No OS, need to flash al memory regions

ST proprietary

› STM32 CubeIDE

› Debug via STLink (won't see this)

How to work

› No way is to compile our code directly on IoT Node

› Cross-compilation *via* the CubeIDE

› Flash the whole OS+program via USB

A simple application

Create a new "Blink" project

› File -> New Project

› Then. Select the MCU (or the board)

› **DO NOT initialize the peripherals in default mode!!! (**for this time..)

STM32L475?? MCU (or B-L475E-IOT01A1 board)

IDE

IDE workspace_1.10.1 - Blink/Core/Src/main.c - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help

📁 Project Explorer ✕

∨ IDE Blink
  > 🔧 Binaries
  > 🔲 Includes
  ∨ 📁
    > .c main.c ← **Main file (the one with "main")**
      .c stm32l4xx_hal_msp.c
      .c stm32l4xx_it.c
      .c syscalls.c
      .c sysmem.c
      .c system_stm32l4xx.c
  > 📁 Startup
  > 📁 Drivers

Some generated files. Do not touch them...

main.c ✕   MX Blink.ioc

```
83    /* USER CODE END SysInit */
84
85    /* Initialize all configured peripheral
86    MX_GPIO_Init();
87    /* USER CODE BEGIN 2 */
88
89    /* USER CODE END 2 */
90
91    /* Infinite loop */
92    /* USER CODE BEGIN WHILE */
93
94    while (1)
95    {
96      /* USER CO
97
98      /* USER CODE B
99
100     // write pin state
```

Initialization/setup + infinite loop (Arduino-like)

Configure LEDs

We want to create **an alias** for GPIOs

› So we don't need to change code when we change LEDs

Let's configure PA5

Leds and GPIOs

LED1 (green) => PA5 pin

LED2 (green) => PB14 pin

General Purpose I/O Ports

Our interface towards the external world

› Also supports Arduino Uno R3

› Let's skip this...

GPIOs are divided into two **board blocks**, and five **SoC ports**

› CN1,2

› Port A, B, C, D, E

› (not all ports are available on the board!!!)

B-L475E-IOT01A2
Arduino™ Uno Rev 3 connectors

Write on GPIO PINs

`stm32l4xx_hal_gpio.h`

```c
void HAL_GPIO_TogglePin (GPIO_TypeDef *GPIOx,
                         int16_t GPIO_Pin );


GPIO_PinState HAL_GPIO_ReadPin (GPIO_TypeDef *GPIOx,
                                int16_t GPIO_Pin );


void HAL_GPIO_WritePin (GPIO_TypeDef *GPIOx,
                        int16_t GPIO_Pin,
                        GPIO_PinState PinState);
```

Project Explorer

- Blink
  - Binaries
  - Includes
  - Core
    - Inc
    - Src
      - main.c
      - stm32l4xx_hal_msp.c
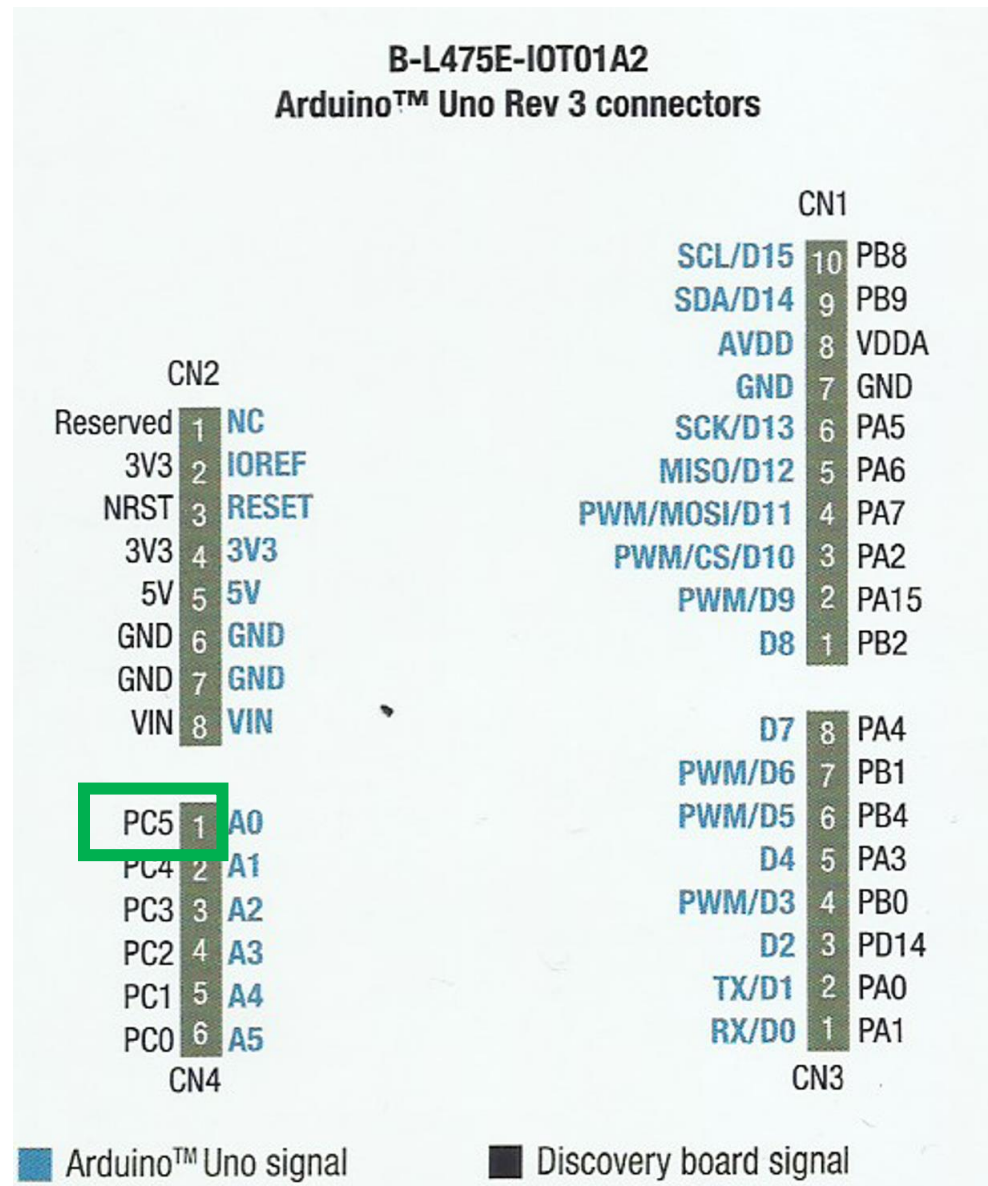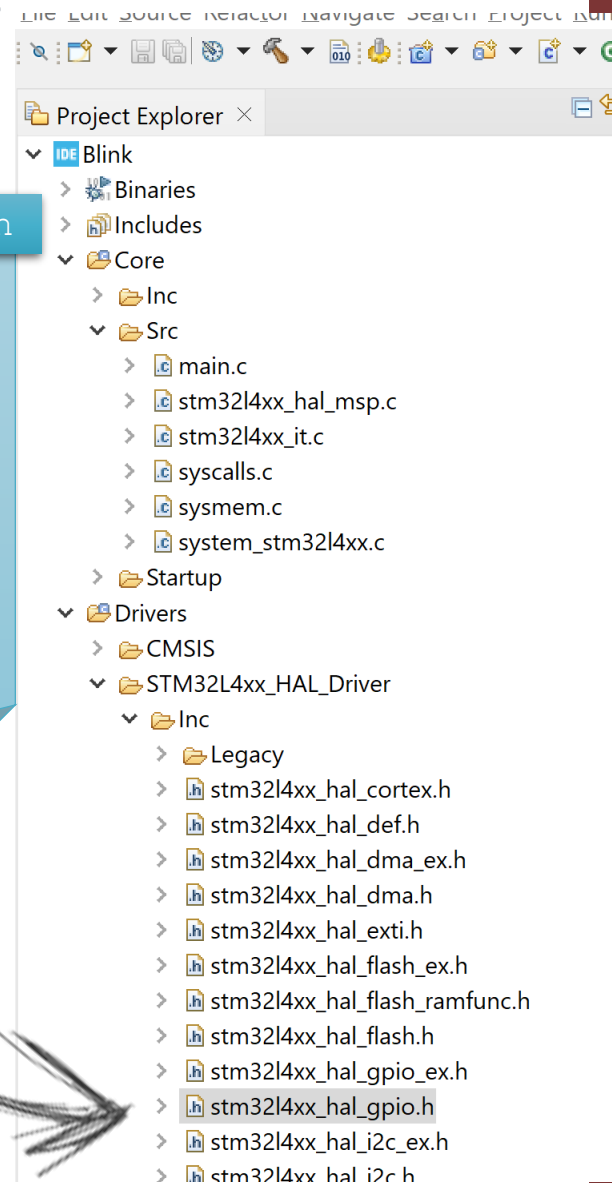   - stm32l4xx_it.c
      - syscalls.c
      - sysmem.c
      - system_stm32l4xx.c
    - Startup
  - Drivers
    - CMSIS
    - STM32L4xx_HAL_Driver
      - Inc
        - Legacy
        - stm32l4xx_hal_cortex.h
        - stm32l4xx_hal_def.h
        - stm32l4xx_hal_dma_ex.h
        - stm32l4xx_hal_dma.h
        - stm32l4xx_hal_exti.h
        - stm32l4xx_hal_flash_ex.h
        - stm32l4xx_hal_flash_ramfunc.h
        - stm32l4xx_hal_flash.h
        - stm32l4xx_hal_gpio_ex.h
        - stm32l4xx_hal_gpio.h
        - stm32l4xx_hal_i2c_ex.h
        - stm32l4xx_hal_i2c.h
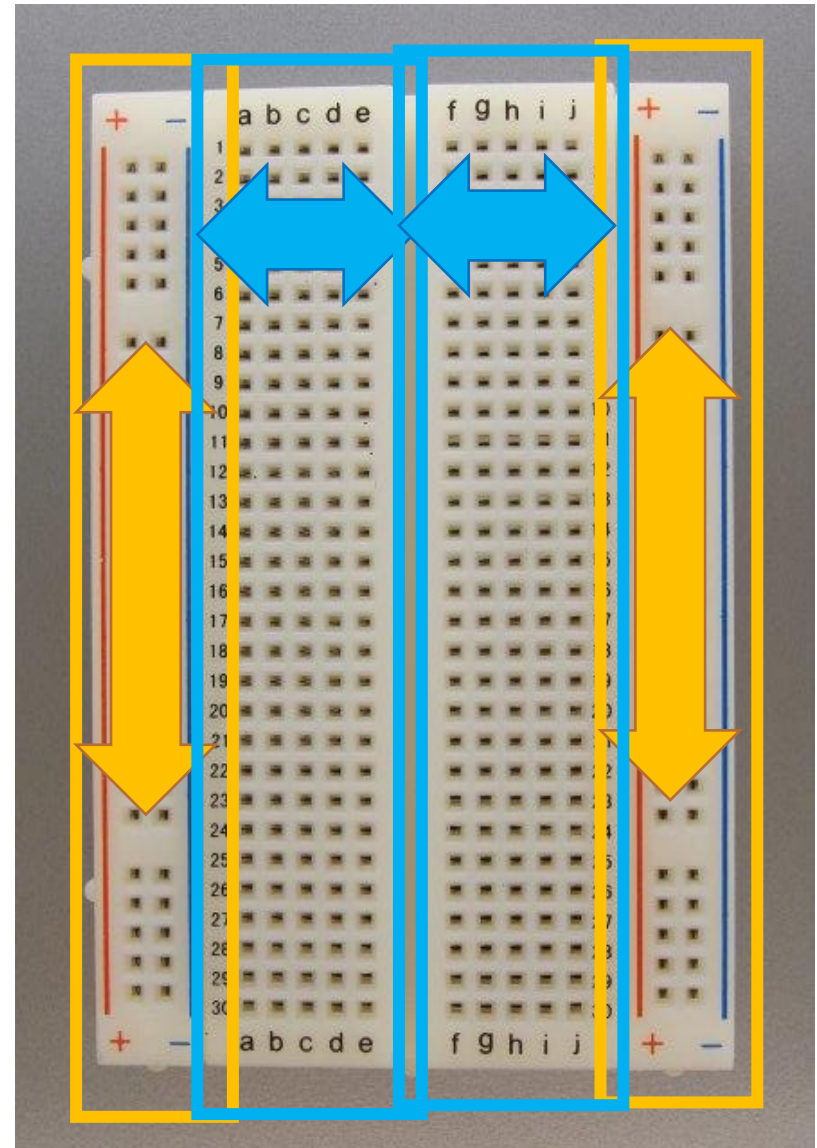
It's a generated file!!

12

Let's play!

Breadboard

Provides electrical connectivity

› Vertical vs. horizontal rails

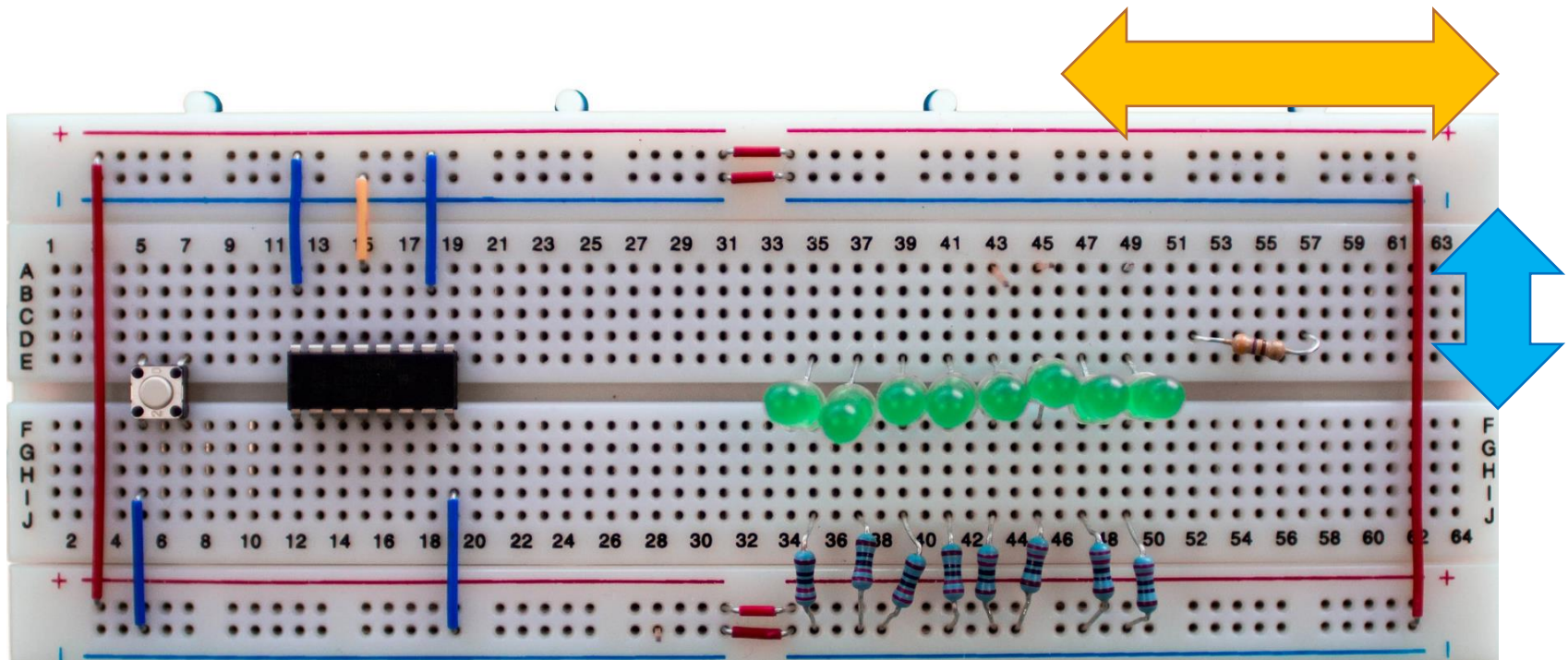› (Typically, power vs other)

› Can use jumper wires

Breadboard

The two sides of the **+** and **-** rails are wired together

› Typically, used for power/GND

Brought to the internal rails with jumper wires

› Where core/chip and other stuff reside
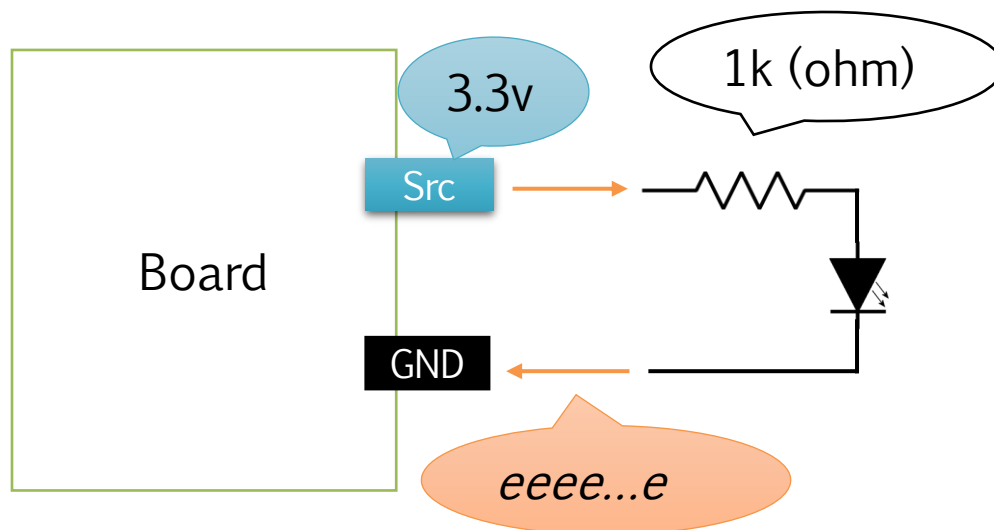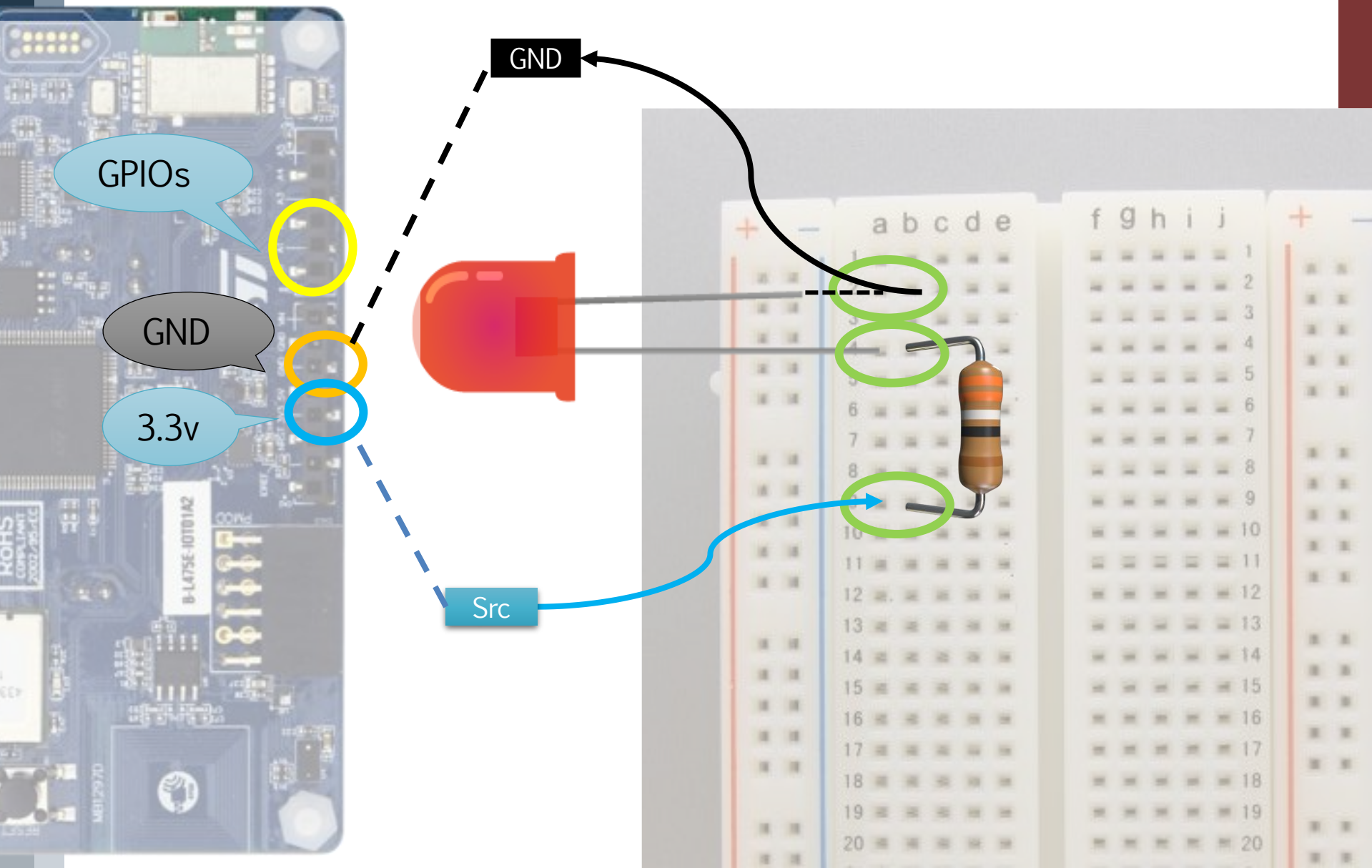
Finally...LEDs

Light Emitting Diodes

› You feed with electrons; they light up

› They have a side!!!!

› They need a resistance to lower the charge

Wrong wiring => you burn them...

GPIOs

GND
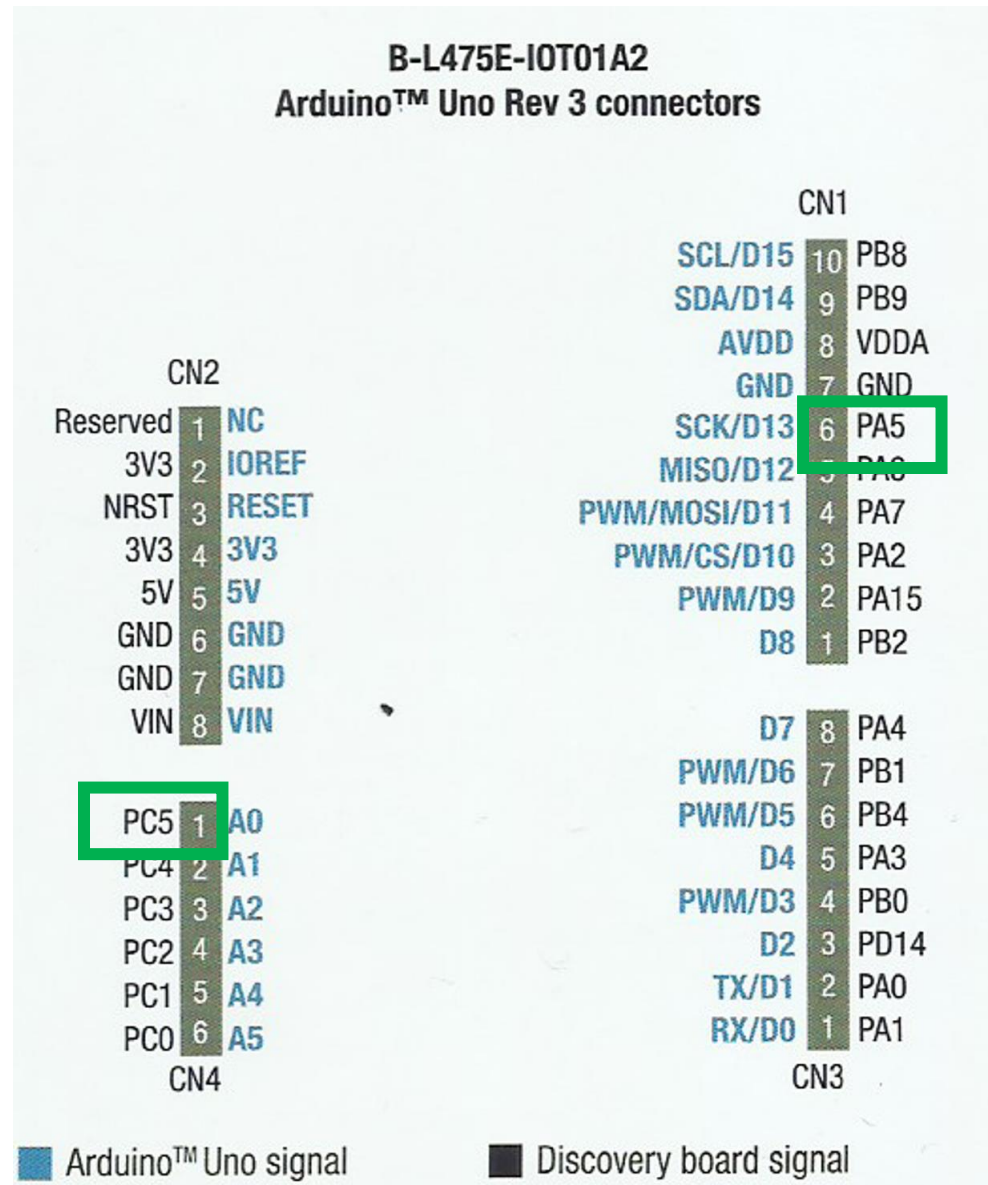
3.3v

GND

Src

a b c d e   f g h i j

Let's play with Pins

PA5 is also in board pinout

› Connect our led to them
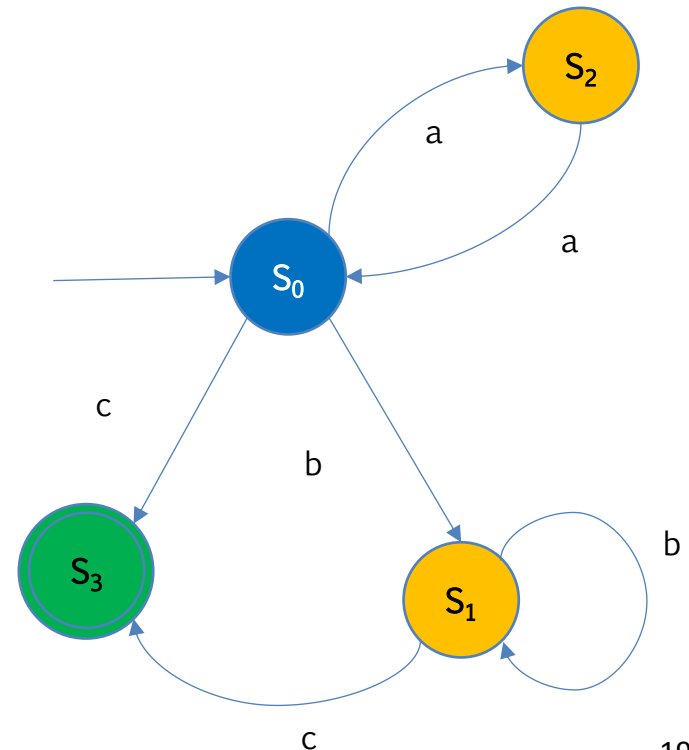
› PB14 is not...

B-L475E-IOT01A2
Arduino™ Uno Rev 3 connectors

Exercise

› Implement the Moore machine of the FSM that understands whether a words is from L

"Identify even sequences of a (even empty),

followed by one, or more, or no, b, ended by c"

› ..and turns on the corresponding led color
  - **Blue** => GPIO 0
  - **Red (error state)** => GPIO 1
  - **Yellow** => GPIO 2
  - **Green** => GPIO 3
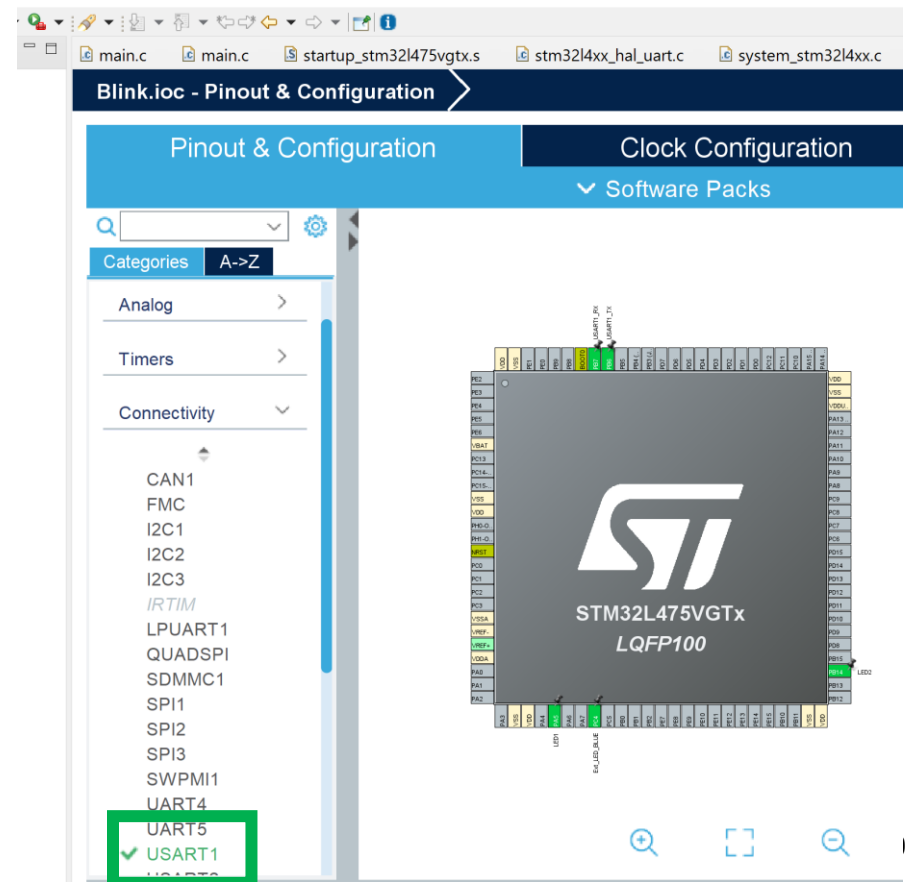

S_2

a

a

S_0

c

b

S_3

b

S_1

c

19

Serial communication

Universal asynchronous receiver-transmitter – UART

› "Asynchronous" -> One channel for TX, one channel for RX

USART Universal Synchronous/Asynchronous Receiver/Transmitter

› w/bitstream

› USART1 in our board

Configure USART1

- › PB6 for Tx

- › PB7 for Rx

Write code

```c
/* USER CODE BEGIN 3 */

uint8_t Test[] = "Hello World !!!\r\n"; //Data to send

HAL_StatusTypeDef retval =
        HAL_UART_Transmit(&huart1,Test,sizeof(Test),10);

//Let's add some delay
HAL_Delay(1000);


/* USER CODE END 3 */
```

› Copy-paste this in your main loop

System header

```
/**
 * @brief Send an amount of data in blocking mode.
 * [...]
 * @param huart UART handle.
 * @param pData Pointer to data buffer (u8 or u16 data elements).
 * @param Size Amount of data elements (u8 or u16) to be sent.
 * @param Timeout Timeout duration.
 * @retval HAL status
 */
HAL_StatusTypeDef HAL_UART_Transmit(UART_HandleTypeDef *huart,
const uint8_t *pData, uint16_t Size, uint32_t Timeout);
```

› Returns "check"

On your machine... (1)

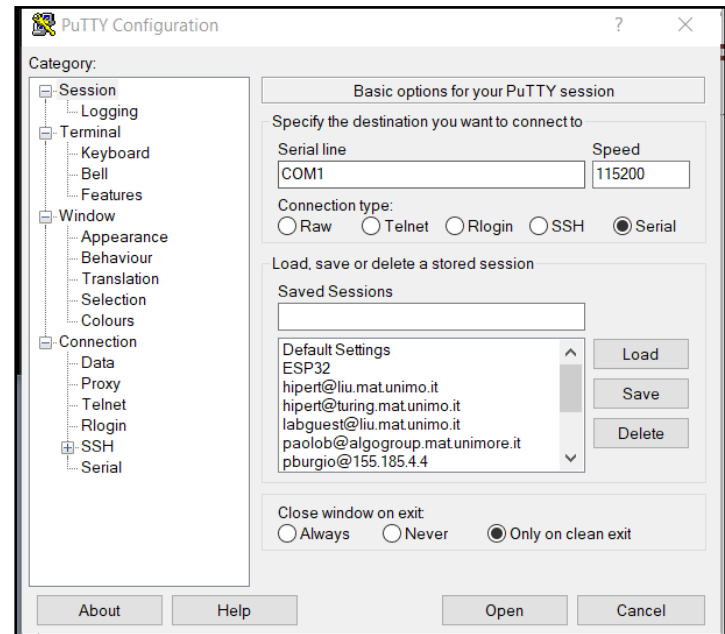First, test with a "standard" serial Monitor

› Linux

 – `sudo apt install minicom`

 – Serial/USB ports are typically `/dev/ttySOMETHING`

› Windows

 – Putty

 – Serial/USB ports are COMx

115200 Baud, no parity, 8 bit

On your machine... (2)

Programmatically read from serial/USB

› C++
  – https://github.com/imabot2/serialib

› Python
  – pySerial

References

Course website

› http://hipert.unimore.it/people/paolob/pub/Industrial_Informatics/index.html

My contacts

› paolo.burgio@unimore.it

› http://hipert.mat.unimore.it/people/paolob/

Resources

› A "small blog -> http://www.google.com

› Serial comms
  – https://wiki.st.com/stm32mcu/wiki/STM32StepByStep:Step3_Introduction_to_the_UART
  – https://github.com/imabot2/serialib