General Purpose Input/Output (GPIO)

Le Yang (le.yang@canterbury.ac.nz)

Department of Electrical and Computer Engineering

Office Hours: Link Building A510, Mondays 16:00 – 18:00

Where we're going today

- **GPIO overview**
- GPIO configuration and access
- Example program

GPIO Overview (1)

- A GPIO = a signal pin on an integrated circuit (IC) or circuit board
  - Basic I/O interface for MCUs
  - Customizable (input (button push)/output (LED driving), analogue, communications …)

- Configurable pin modes:
  - Simple input/output (e.g., to read from a button/drive a LED …)
    - Pull-up/Pull-down/No pull ?
    - Push-pull (PP)/Open drain (OD) ?
    - Events/Interrupts ?
  - Analogue (e.g., to monitor the environment using a sensor …)
  - Alternate function (AF) (to be connected to other peripherals and realize their functions)
    - DMA, ADC/DAC, PWM, serial communications …

GPIO Overview (2)

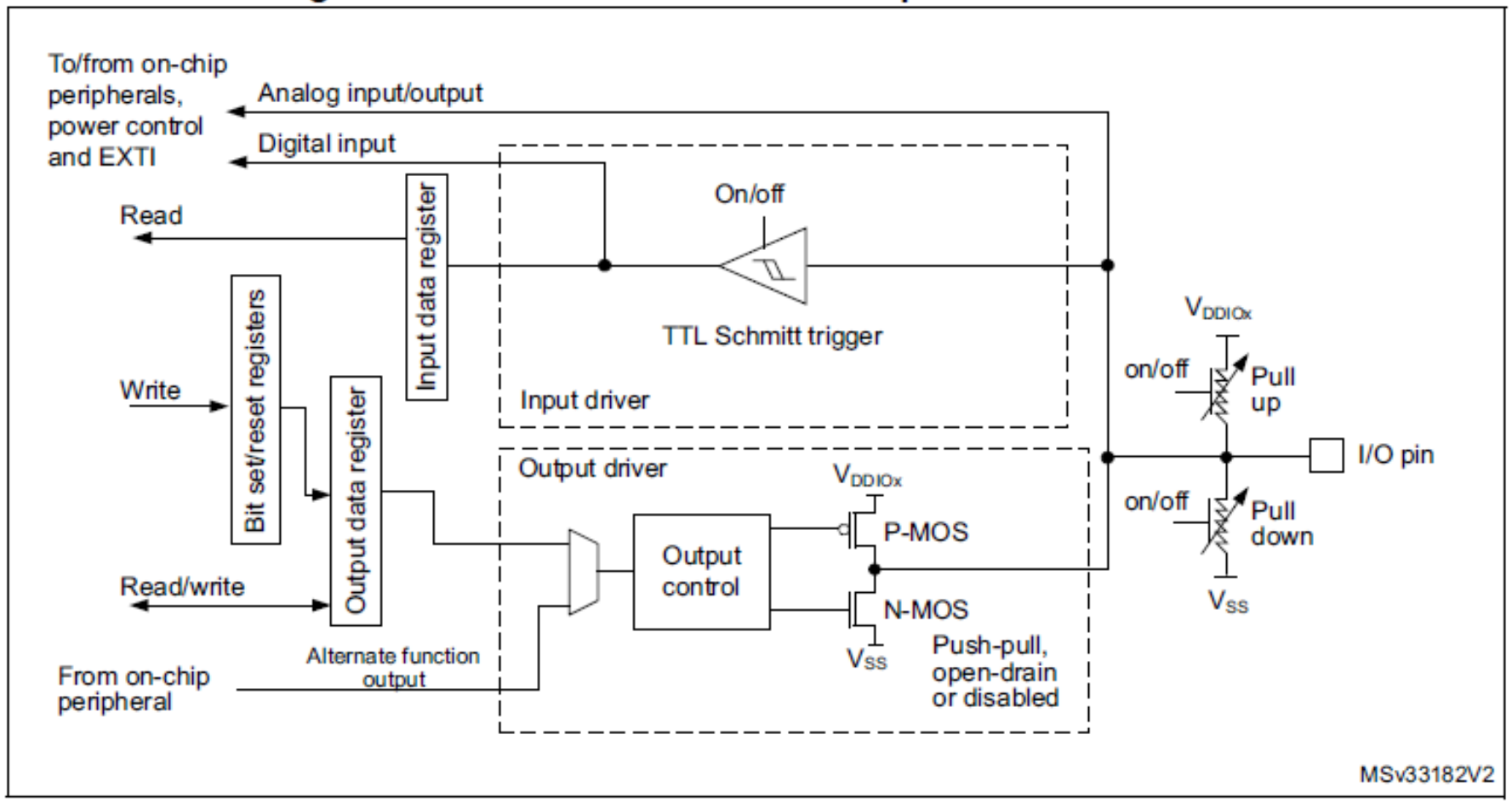
- Exact number of available GPIO pins on a MCU depends on
  - IC package
  - Family of MCUs
  - Board design …

- 76 pins on NUCLEO-C071RB board
  - 4 single-in-line headers, each with 19 pins
  - Collected in two ST morpho pin headers (CN7 & CN10, see the next two slides)

- 60 GPIO pins accessible via 5 ports (PA, PB, PC, PD, PF)
  - 'P' stands for port → "PF0", "PA7", "PB4"
  - Many memory locations are assigned to each GPIO port (why?)

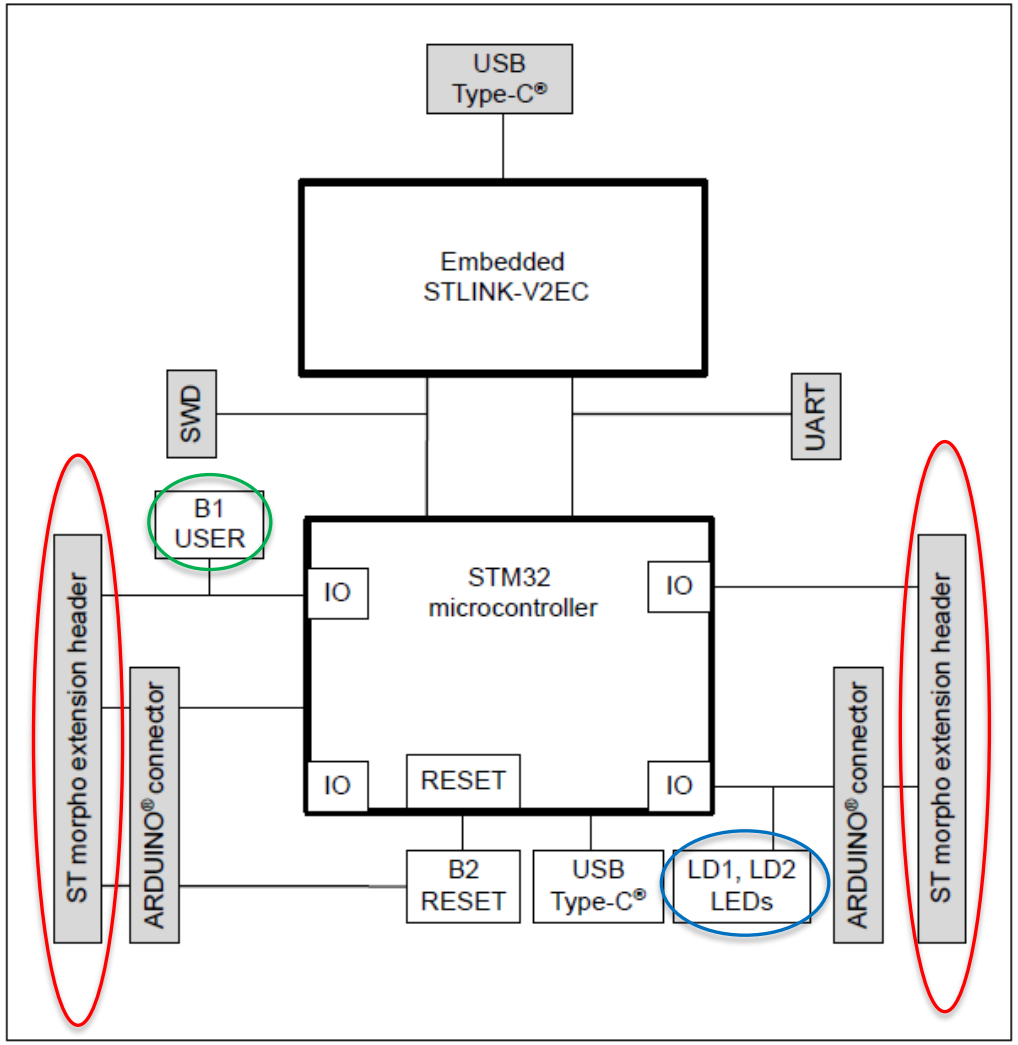
Basic Structure of a GPIO Pin

Figure 16. Basic structure of an I/O port bit



RM0490 STM32C0 series advanced ARM-based 32-bit MCUs.pdf

NUCLEO-C071RB board Block Diagram



ST morpho pin headers (CN7 & CN10)

Output LEDs (LD1 & LD2)

User switch (B1)

UM3353 STM32 NUCLEO64 board (MB2046).pdf

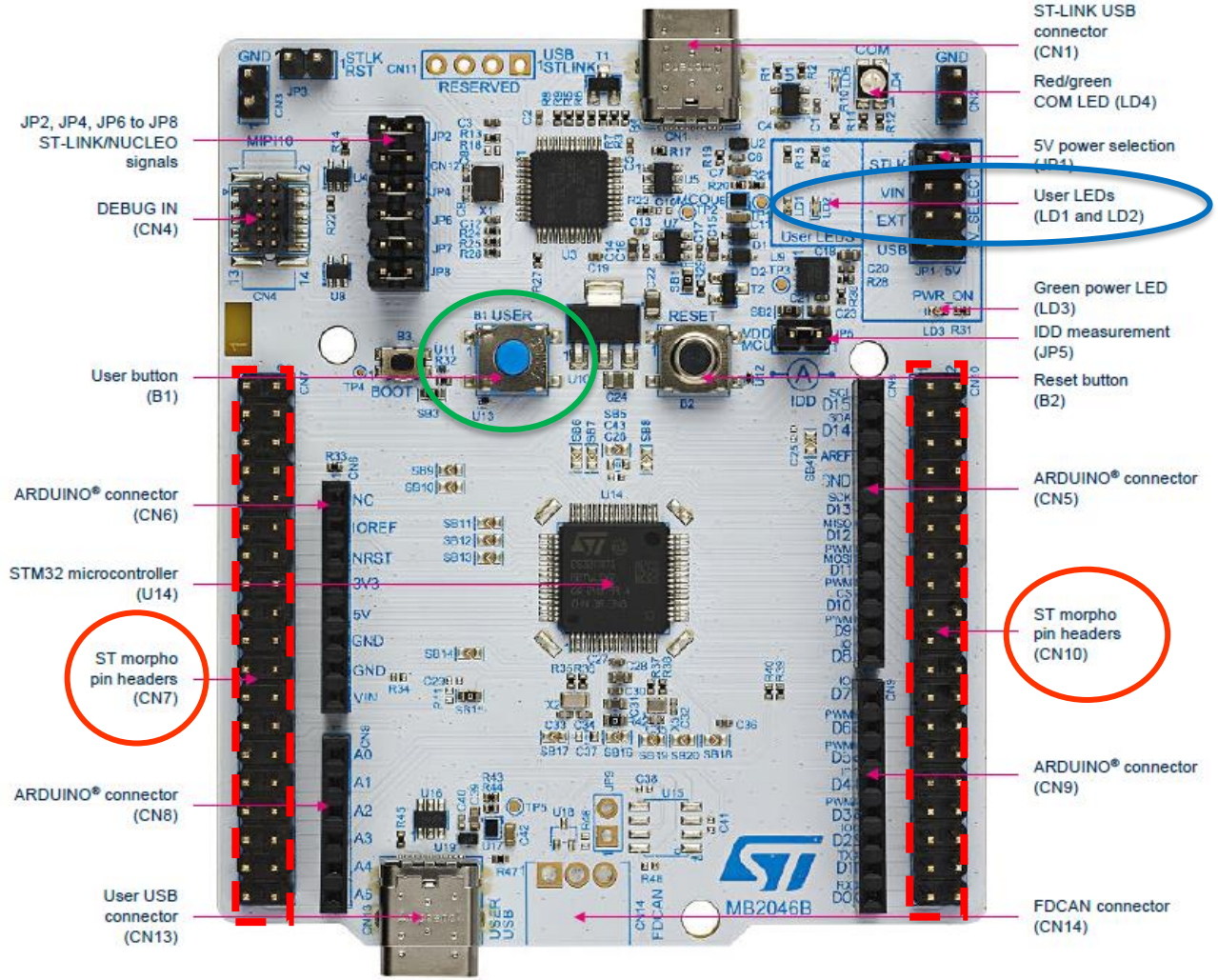


Table 12. Pin assignments for the STM32 on the ST morpho connectors

CN7				CN10			
Pin name	Pin number	Pin number	Pin name	Pin name	Pin number	Pin number	Pin name
PC10	1	2	PC11	PC3	1	2	PC9
PC12	3	4	PD2	PB8	3	4	PC1
VDD	5	6	5V_EXT	PB9	5	6	PA3
PD4	7	8	GND	VREF[4]	7	8	5V_STLK

Table 12 in UM3353 STM32 NUCLEO64 board (MB2046).pdf

Where we’re going today

- GPIO overview
- **GPIO configuration and access**
- Example program

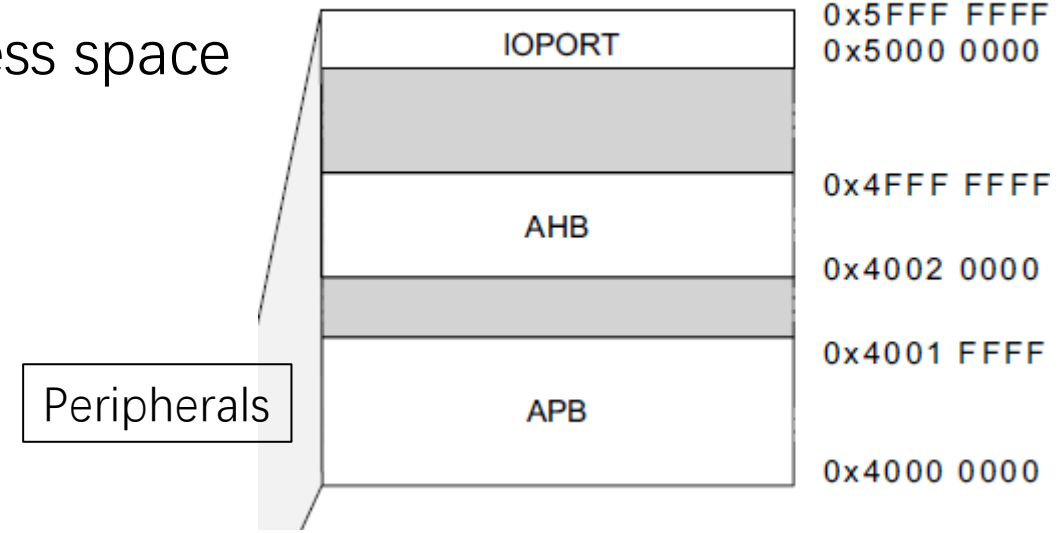
Output LEDs and User Button

- NUCLEO-C071RB board has
  - 2 user LEDs: LD1 (green) and LD2 (blue)
    - LD1 → PA5, active HIGH
    - LD2 → PC9, active LOW
    - To drive a LED, PA5 or PC9 needs to be set as OUTPUT and NO_PULL
  - 1 user button (B1)
    - B1 → PC13, active LOW
    - B1 implemented with a firmware debounce filter
    - To read the user button, PC13 needs to be set as INPUT and PULL_UP

UM3353 STM32 NUCLEO64 board (MB2046).pdf

Drive LD1 via PA5 (1)

- Peripherals mapped to a region of 4GB address space
  - 0x 4000 0000 to 0x5FFF FFFF (0.5GB)
  - PA5: 0x5000 0000 (base) - 0x5000 03FF (1KB)



- Configuration code:

```
int main(void) {
    uint32_t *GPIOA_MODER, *GPIOA_OTYPER;

    //Enable Port A
    __HAL_RCC_GPIOA_CLK_ENABLE();                  // Enable Port A

    GPIOA_MODER = (uint32_t*) (0x50000000);        // Address of the GPIOA_MODER register
    GPIOA_OTYPER = (uint32_t*) (0x50000000 + 0x14); // Address of the GPIOA_ODR register

   *GPIOA_MODER = *GPIOA_MODER | 0x400;            // Set MODER[11:10] = 0x01
   *GPIOA_OTYPER = *GPIOA_OTYPER | 0x00;           // Set OTYPER[5] = 0x00
    .
    .
```

00: Input mode (reset state)
01: General purpose output mode
10: Alternate function mode
11: Analog mode

Page 45, RM0490 STM32C0 series advanced ARM-based 32-bit MCUs.pdf

HAL-based GPIO Pin Configuration (1)

- HAL: Hardware Abstraction Layer
  - A set of libraries enabling the control of peripherals and core features without dealing with low-level details

- HAL_GPIO_Init(GPIO_TypeDef* GPIOx, GPIO_InitTypeDef* GPIO_Init)
  - GPIOx: GPIOA, GPIOB, …
  - GPIO_Init: GPIO pin configuration:

```
typedef struct {
        uint32_t Pin;        // Pin number (e.g., GPIO_PIN_5)
        uint32_t Mode;       // Pin mode (e.g., GPIO_MODE_OUTPUT_PP)
        uint32_t Pull;       // Pull mode (e.g., GPIO_PULLDOWN)
        uint32_t Speed;      // Output speed (e.g., GPIO_SPEED_FREQ_LOW)
        uint32_t Alternate;  // Alternate function (AF)
} GPIO_InitTypeDef;
```

HAL-based GPIO Pin Configuration (2)

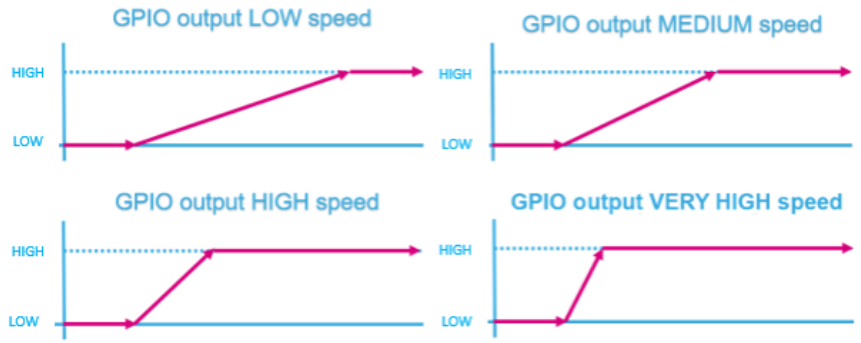
- HAL-based PA5 configuration code:

```
GPIO_InitTypeDef GPIO_InitStruct = {0};

/*Configure GPIO pin : PA5 */
GPIO_InitStruct.Pin = GPIO_PIN_5;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;

HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
```

```
typedef struct {
        uint32_t Pin;        // Pin number
        uint32_t Mode;       // Pin mode
        uint32_t Pull;       // Pull mode
        uint32_t Speed;      // Output speed
        uint32_t Alternate;  // Alternate function
} GPIO_InitTypeDef;
```



UM3029 Description of STM32C0 HAL and Low-layer Drivers.pdf

HAL-based GPIO Pin Access

- To read the status of a GPIO Pin:

  GPIO_PinState HAL_GPIO_ReadPin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);

  - GPIO_PinState: GPIO_PIN_SET (HIGH), GPIO_PIN_RESET (LOW)

- To write to a GPIO Pin:

  void HAL_GPIO_WritePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState);

- To invert the state of a GPIO Pin:

  void HAL_GPIO_TogglePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);

- **Drive LD1 through PA5:**

  HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);

Other HAL_GPIO Functions

- To lock the configuration of a GPIO pin:

  HAL_StatusTypeDef HAL_GPIO_LockPin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin)

  - Any subsequent attempt to change the configure will fail, until RESET

- To set a GPIO Pin to its default RESET status:

  void HAL_GPIO_DeInit(GPIO_TypeDef* GPIOx, uint32_t GPIO_Pin)

  - If we no longer need a GPIO pin, turn it off to save power

Other GPIO Pin Modes

Pin Mode	Description
GPIO_MODE_INPUT	Input Floating Mode
GPIO_MODE_OUTPUT_PP	Output Push Pull Mode
GPIO_MODE_OUTPUT_OD	Output Open Drain Mode
GPIO_MODE_AF_PP	Alternate Function Push Pull Mode
GPIO_MODE_AF_OD	Alternate Function Open Drain Mode
GPIO_MODE_ANALOG	Analog Mode
GPIO_MODE_IT_RISING	External Interrupt Mode with Rising edge trigger detection
GPIO_MODE_IT_FALLING	External Interrupt Mode with Falling edge trigger detection
GPIO_MODE_IT_RISING_FALLING	External Interrupt Mode with Rising/Falling edge trigger detection
GPIO_MODE_EVT_RISING	External Event Mode with Rising edge trigger detection
GPIO_MODE_EVT_FALLING	External Event Mode with Falling edge trigger detection
GPIO_MODE_EVT_RISING_FALLING	External Event Mode with Rising/Falling edge trigger detection

UM3029 Description of STM32C0 HAL and Low-layer Drivers.pdf

Where we’re going today

- GPIO overview
- GPIO configuration and access
- **Example program**

Drive LD1 via PA5 (2)

```
int main(void) {
{
    /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
    HAL_Init();

    /* Configure the system clock */
    SystemClock_Config();

    GPIO_InitTypeDef GPIO_InitStruct = {0};

    /* Enable Port A */
     __HAL_RCC_GPIOA_CLK_ENABLE();

    /* Configure GPIO pin : PA5 */
    GPIO_InitStruct.Pin = GPIO_PIN_5;                    // Pin 5
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;          // Push-Pull output pin
    GPIO_InitStruct.Pull = GPIO_NOPULL;                  // No pull
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;         // Low output speed

    HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);    .         // Configure PA5

    /* Drive LD1 via PA5 */
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);
}
```

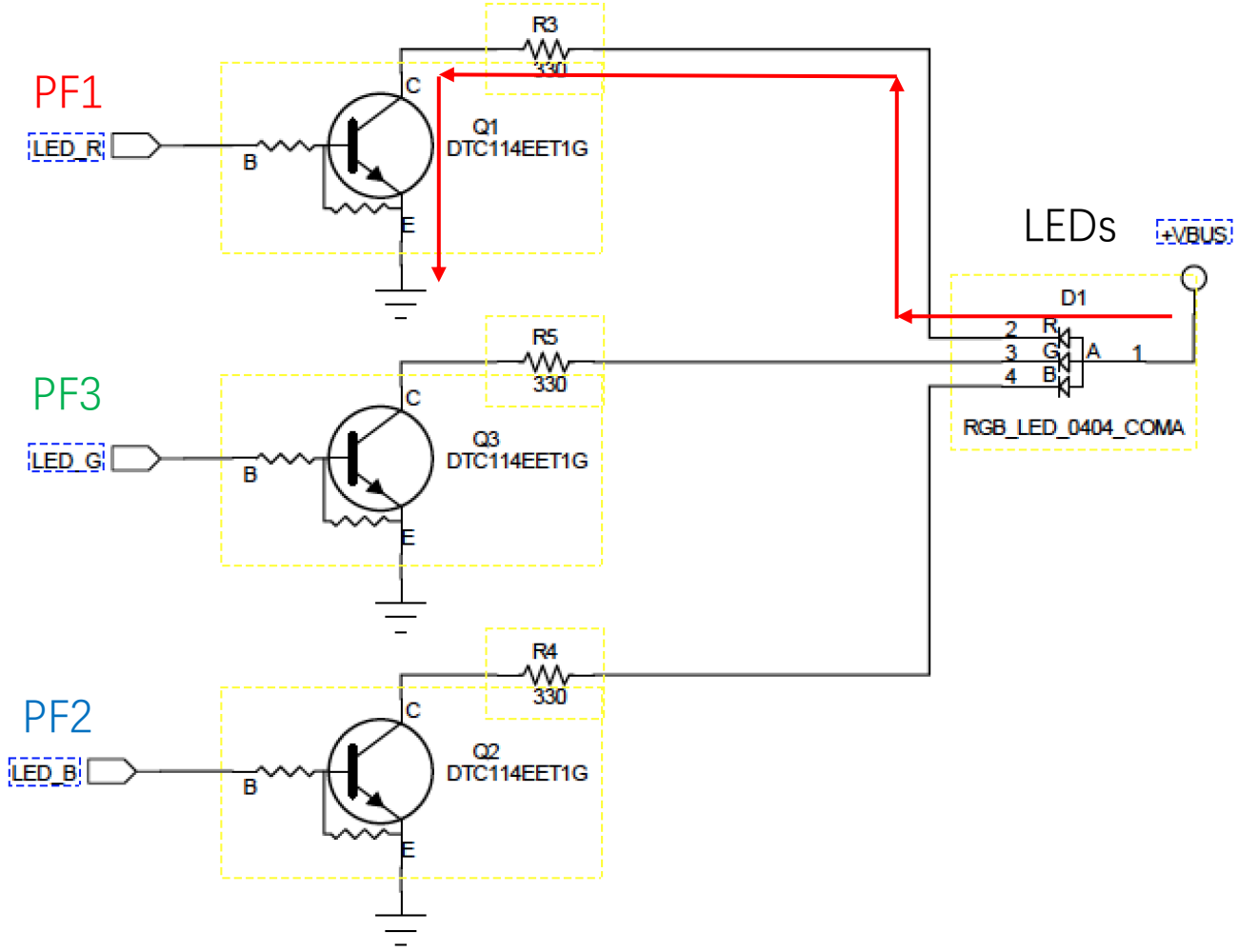
After-Lecture Exercise:
Configure PA5 as in this slide via the CubeIDE pin configuration

Supplementary Materials

- Active high LED circuitry

Active High LED Circuitry

- Transistors as current switches
  - V_{BE} greater than, say, 0.7V
    - Transistors in saturate state
    - V_{CE} smaller than, say, 0.2V, equivalent to closed circuits from +VBUS to GND
- LEDs configured as active high
  - A scheme adopted in most test programs



From Page 20, Tiva C Series TM4C123G Launchpad Users Guide.pdf