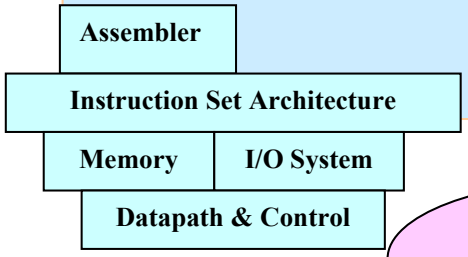
ELEC 3300
Introduction to Embedded Systems

Topic 4

Embedded System Structure

Prof. Tim Woo

Course Overview



Introduction to Embedded Systems

More about Embedded Systems

Basic Computer Structure

MCU Main Board

Digital and Analog Interfacing

USART, IEEE1394, USB, I2C and SPI

Microcontroller Structure

A/D Port

CPU

Serial Port

Buffering and Direct Memory Access (DMA)

External Memory Port

Memory, Interfacing to Memory, Memory Timing and applications

External Interrupt Port

Interrupt Organization

External Timer Port

Timer and Counter

Interfacing LCD

Simple I/O Port

Motor Interfacing

In this course, STM32 is used as a driving vehicle for delivering the concepts.

To be covered	In progress	Done

Expected Outcomes

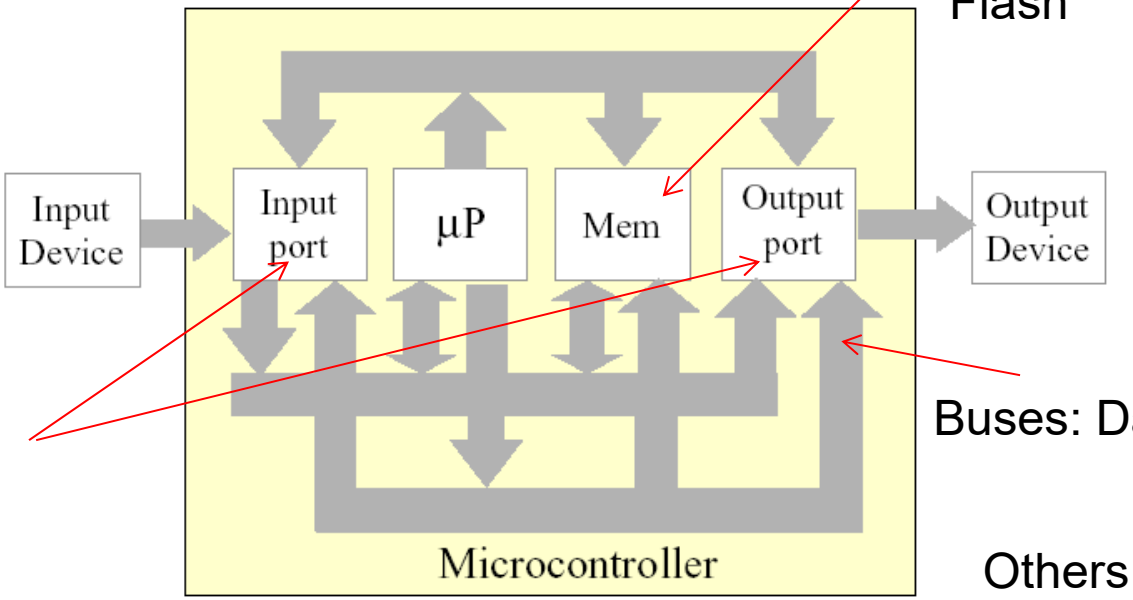
- On successful completion of this topic, you will be able to
  - Summarize the features of ARM micro-controller
  - Describe the bus architecture
  - Understand the memory organization and its map
  - Introduce the Cyclic Redundancy Check (CRC) calculation unit
  - Illustrate examples of the pin definitions of several features
  - Configure a General Purpose I/O for input / output signal

Microcontroller Features

- Processor with memory & I/O ports integrated on the same chip

RAM and SRAM: Volatile memory
ROM: Non Volatile
EPROM: Non Volatile, but erasable by UV light
Flash: Can write (like RAM) but non volatile (like ROM). Electrically erasable.

- Program memory : ROM / EPROM / Flash
- Data memory: RAM / SRAM / Flash



I/O ports

Buses: Data / Address / Control

Others: Timers and Interrupts

You have to write your own device drivers to control the external devices

ARM Microcontroller: Features of STM32

- There are 3 general documents for your reference:
  - Data sheet: STM32F103ZET6-STMicroelectronics-datasheet-7543760.pdf
    - Provides summary of features, pin layouts, pin definitions, electrical characteristics, etc.
  - Reference Manual: STM32_Reference_Manual.pdf
    - Provides complete information on how to use the processor such as registration information of communication protocol, ADC, etc.
    - This could help in writing the codes for initialization and implementation.
  - Programming Manual: STM32_Cortex_M3_Programming_Manual.pdf
    - Provides information for application and system-level software, such as efficient processor core, system and memories, fast interrupt handling, etc.

ARM Microcontroller: Features of STM32

Where are they?

CPU

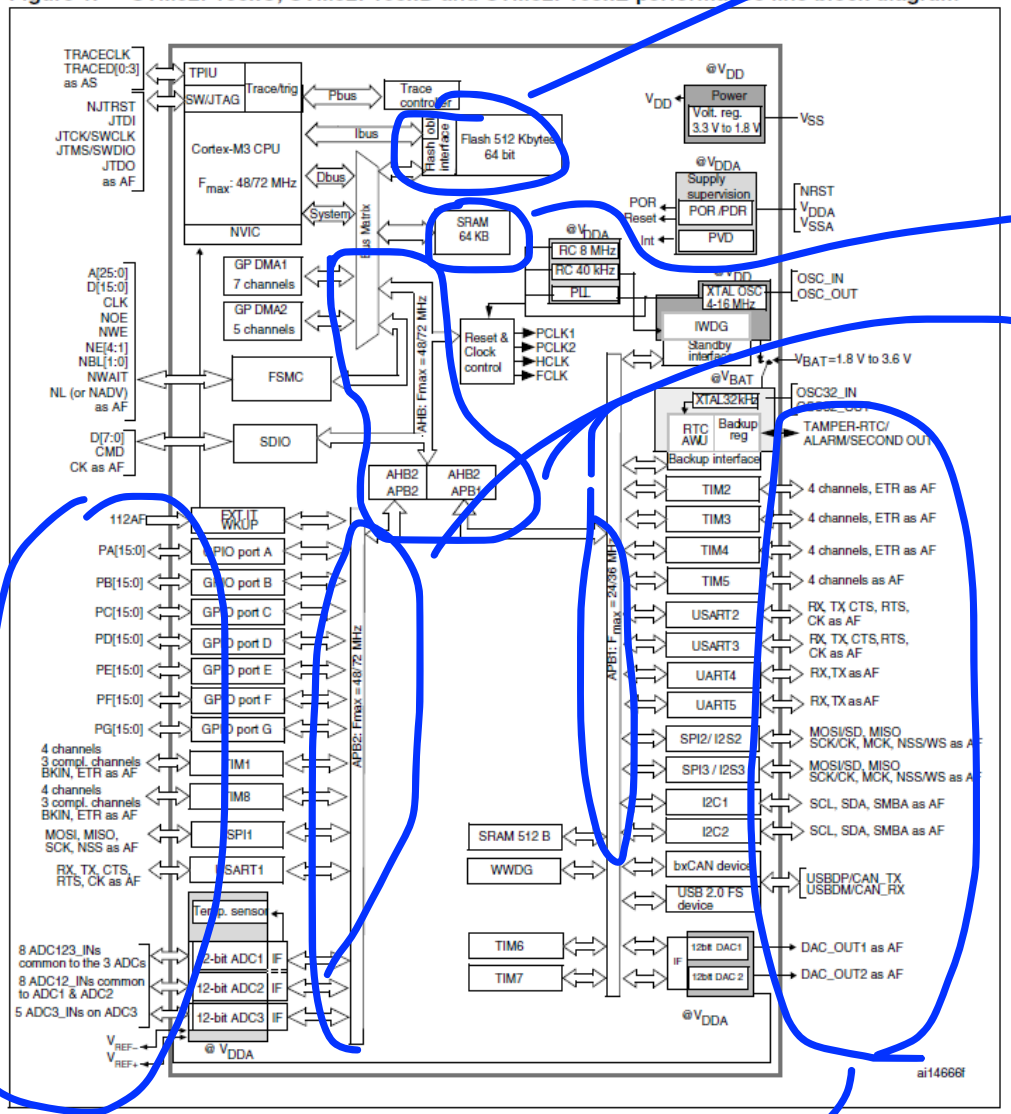
I/O port

Program memory

Data memory

Buses

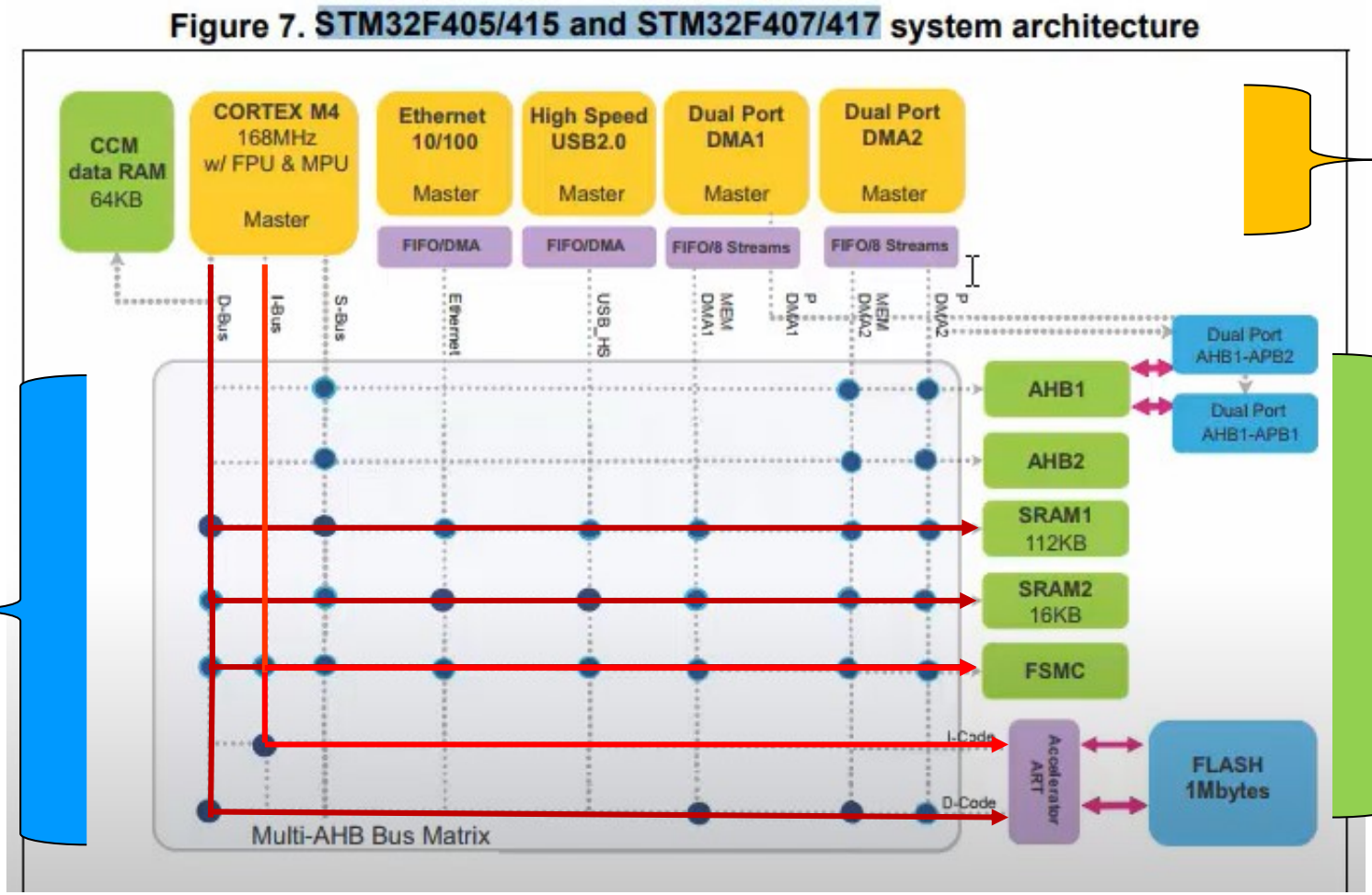
Figure 1. STM32F103xC, STM32F103xD and STM32F103xE performance line block diagram



ARM Microcontroller

- The STM32 can support two configurations:
  - general devices (without Ethernet access)
  - connectivity line (with Ethernet access)
- You have to choose the configuration in the beginning.

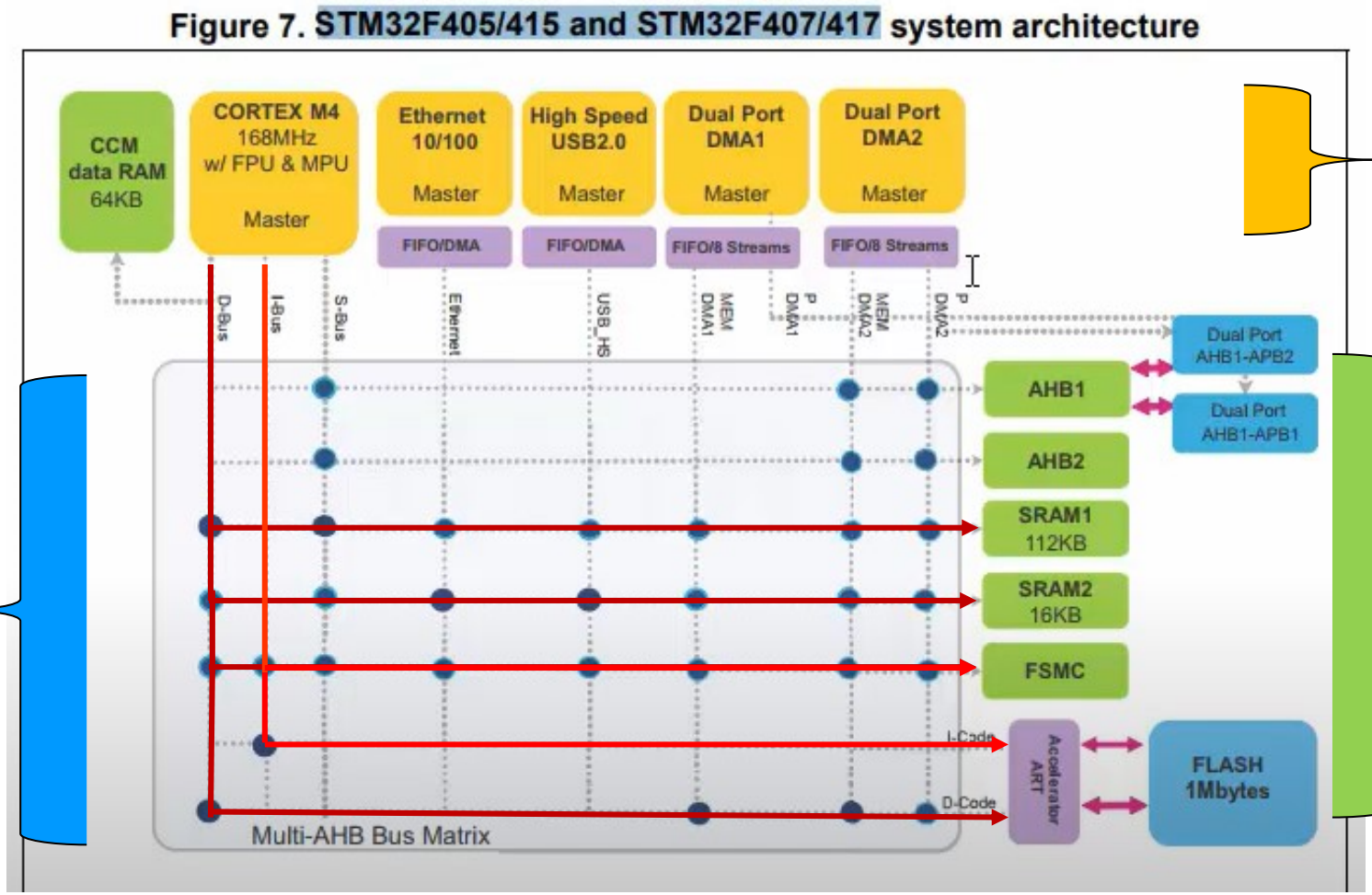
ARM Microcontroller: Bus architecture



- Connected path for communication between Master and Slave

Communication between the processor (master) and peripherals (slave).

ARM Microcontroller: Bus architecture

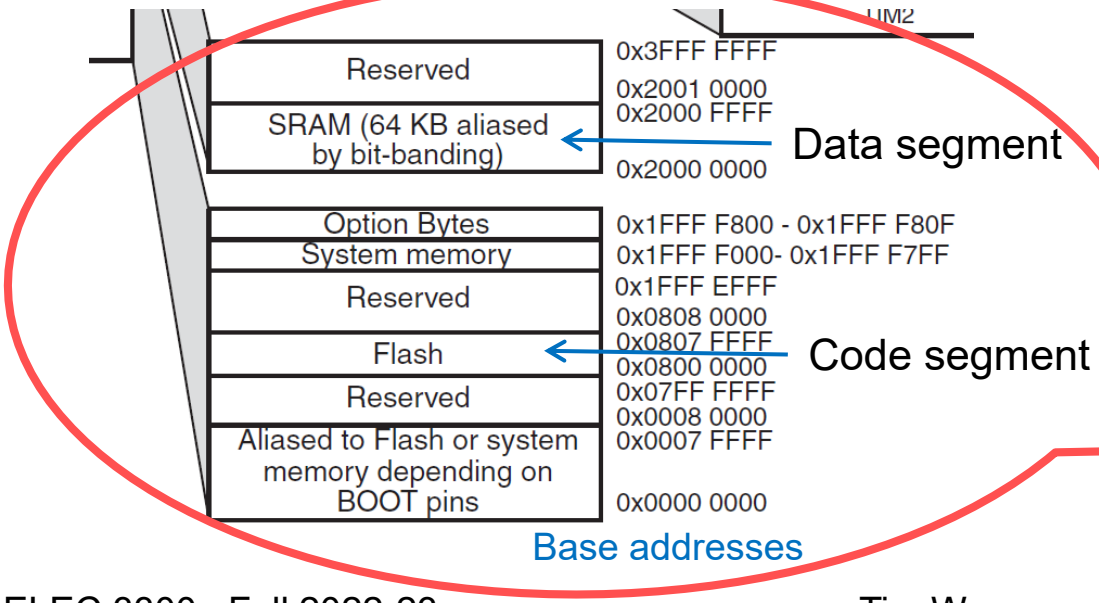


The **bus matrix** provides access from a master to a slave, enabling concurrent access and efficient operation even when several high-speed peripherals work simultaneously.

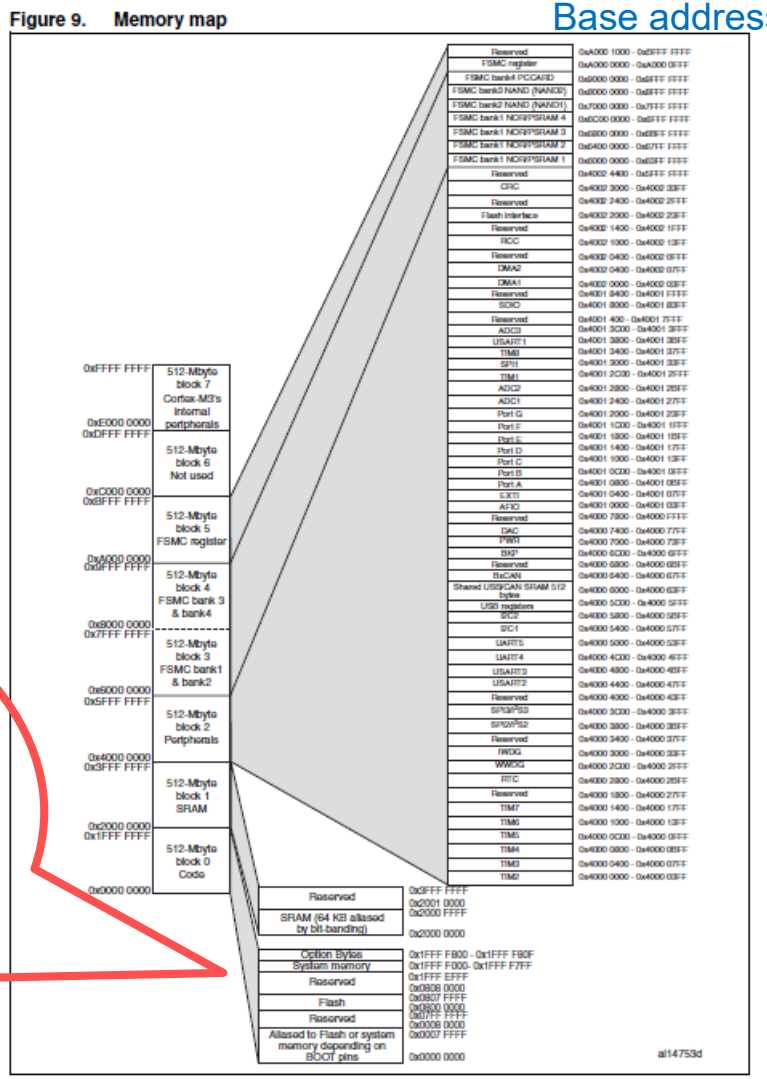
ARM Microcontroller: Memory Map

- Memory plays an important role in the micro-controller. It tells you where to store and read the instructions, data, status of information.

- Reference: Figure 9 in datasheet *



Base addresses



* : STM32F103ZET6-STMicroelectronics-datasheet-7543760.pdf

ARM Microcontroller: Memory Map

- It also tells you the location of the registers for different features.
- Reference: Figure 9 in datasheet *

Register for ADC1

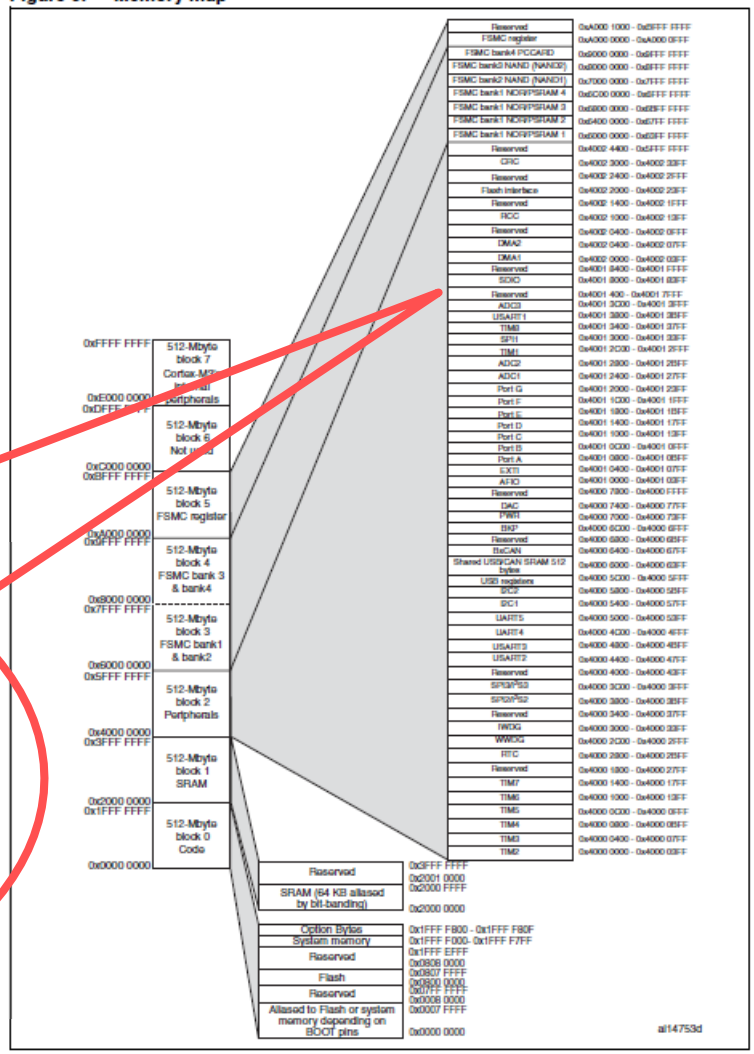
Register for Port D

Base addresses

TIM1	0x4001 2C00 - 0x4001 2FFF
ADC2	0x4001 2800 - 0x4001 2BFF
ADC1	0x4001 2400 - 0x4001 27FF
Port G	0x4001 2000 - 0x4001 23FF
Port F	0x4001 1C00 - 0x4001 1FFF
Port E	0x4001 1800 - 0x4001 1BFF
Port D	0x4001 1400 - 0x4001 17FF
Port C	0x4001 1000 - 0x4001 13FF
Port B	0x4001 0C00 - 0x4001 0FFF
Port A	0x4001 0800 - 0x4001 0BFF
EXTI	0x4001 0400 - 0x4001 07FF

Base addresses

Figure 9. Memory map



* : STM32F103ZET6-STMicroelectronics-datasheet-7543760.pdf

ARM Microcontroller: CRC

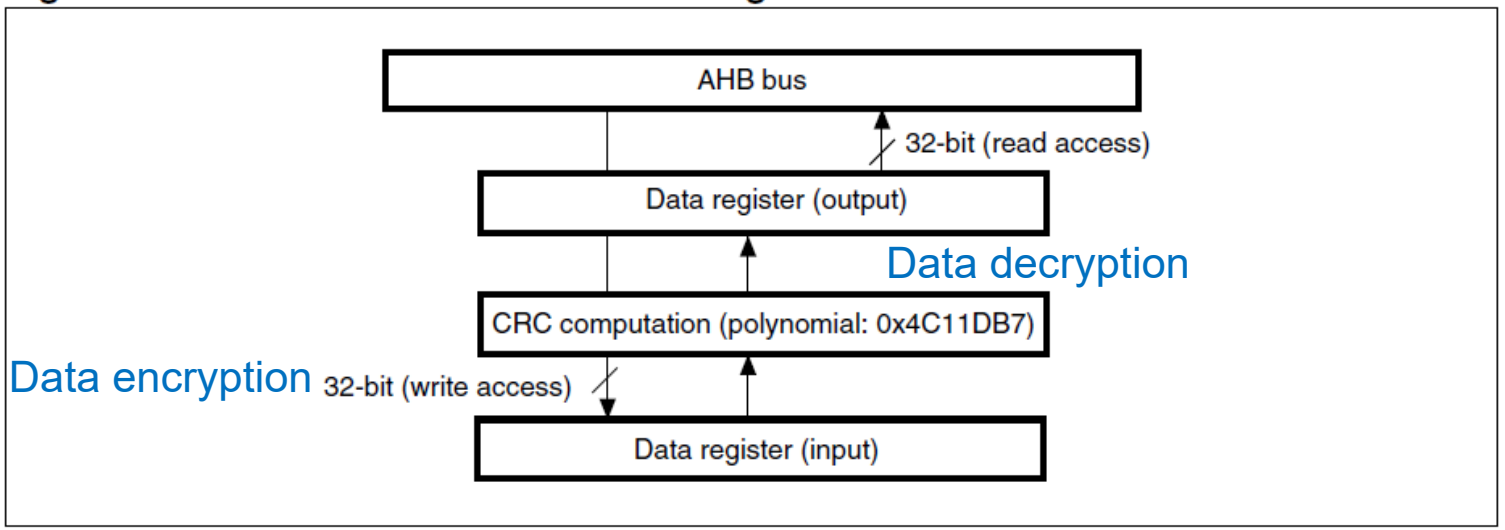
- CRC (Cyclic Redundancy Check)
  - Error-detecting code to detect accidental changes to raw data.
  - Verify data transmission or storage integrity.
  - On retrieval, the calculation is repeated and, in the event the check values do not match, corrective action can be taken against data corruption.
  - CRC uses Generator Polynomial which is available on both sender and receiver side.
  - An example generator polynomial is of the **form like $x^3 + x + 1$**. This generator polynomial represents key 1011.
  - Another example is $x^2 + 1$ that represents key …..?

    101.

ARM Microcontroller: CRC Calculation Unit

- Gets a CRC code from a 32-bit data word and a fixed generator polynomial.

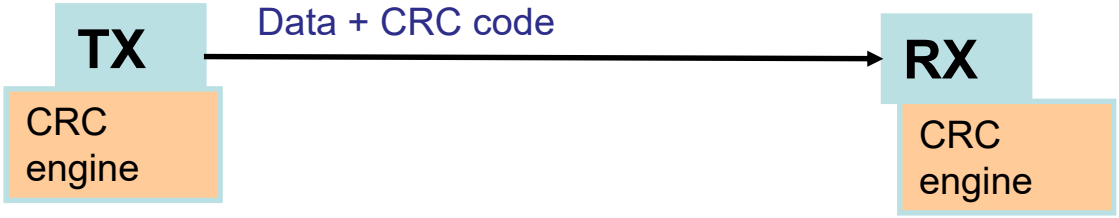
Figure 3. CRC calculation unit block diagram #



Basic CRC Algorithm:

1. Start with an initial CRC value (example: 0xFF).
2. Then XOR it with your data.
3. If the result of step 2 has MSB = 0, then simply left shift that number by 1.
4. If the result from step 2 has MSB = 1, then left shift by 1 and also XOR it with a chosen polynomial.
5. The final value is check sum or CRC value.

ARM Microcontroller: CRC Illustration



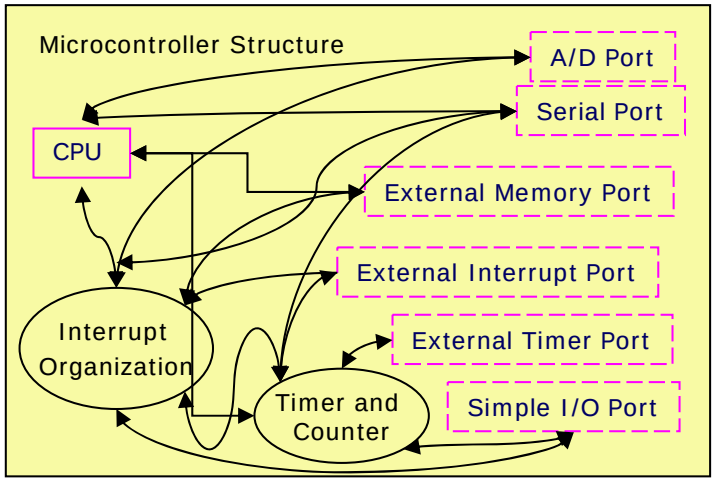
- The Transmitter (TX) will run the data it wants to send through a CRC algorithm.
- It will then have a CRC code that will be unique to that data. So TX sends its data along with the CRC code to the RX.
- Now RX receives the data and CRC code.
- RX will run the data through the same CRC algorithm.
- RX now will compare the CRC it received with the one it calculated and if they match then the data is not corrupted.
- If the CRC codes do not match then something is corrupted.

About High density pin definitions

ARM Microcontroller: High density pin definitions

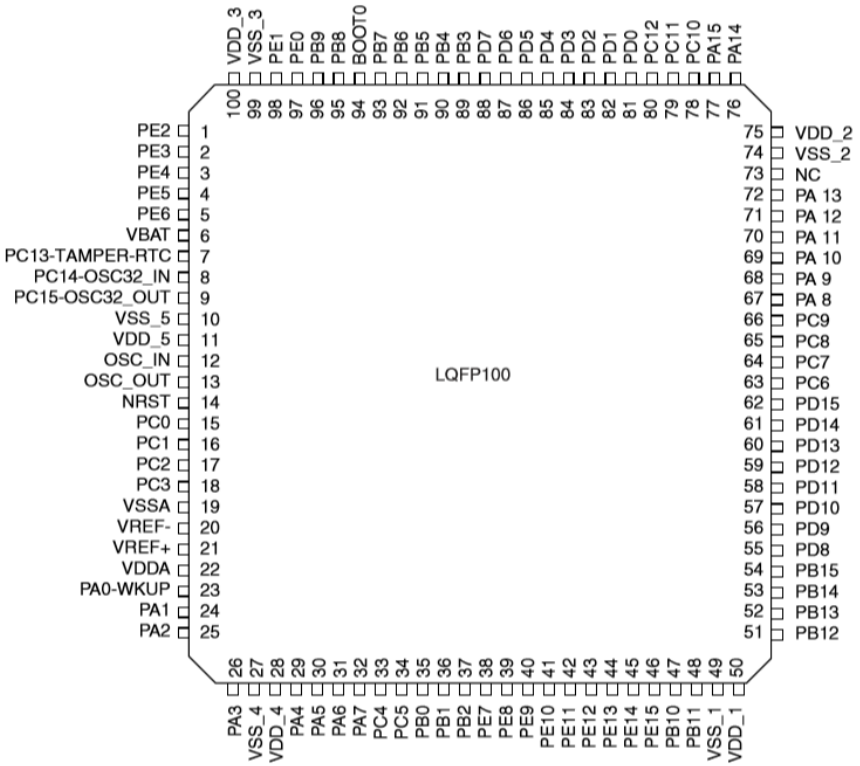
- Pin assignment from Data Sheets
- Reference: Table 5 in datasheet *
- General Purpose I/O

Recall the overview pictures



GPIO	Pin numbers
PA0-PA15	23,24,25, 26, 29, 30, 31, 32, 67,68, 69, 70, 71,72, 76, 77
PB0-PB15	
PC0-PC15	15,16,17,18, 33,34,63,64, 65,66,78,79, 80, 7, 8, 9
PD0-PD15	
PE0-PE15	

STM32F103VE, LQFP-100



* : STM32F103ZET6-STMicroelectronics-datasheet-7543760.pdf

ARM Microcontroller: High density pin definitions

- In general, all the tasks are not executed simultaneously.
- This allows the sharing of pin definition for several tasks. (multiple roles in same pin)

- Analogy:

Table 5. High-density STM32F103xx pin definitions (continued)

Pins						Pin name	Type(1)	I / O Level(2)	Main function(3) (after reset)	Alternate functions(4)	
BGA144	BGA100	WLCSP64	LQFP64	LQFP100	LQFP144					Default	Remap
K1	H1	-	-	20	31	V_{REF-}	S		V_{REF-}		
L1	J1	F7 (7)	-	21	32	V_{REF+}	S		V_{REF+}		
M1	K1	G8	13	22	33	V_{DDA}	S		V_{DDA}		
J2	G2	F6	14	23	34	PA0-WKUP	I/O		PA0	WKUP/USART2_CTS(8) ADC123_IN0 TIM2_CH1_ETR TIM5_CH1/TIM8_ETR	
K2	H2	E6	15	24	35	PA1	I/O		PA1	USART2_RTS(8) ADC123_IN1/ TIM5_CH2/TIM2_CH2(8)	

GPIO (default)
Wake up
USART
ADC
Timer

ARM Microcontroller: High density pin definitions

- Example: We have 3 ADCs, and they may share same pins.

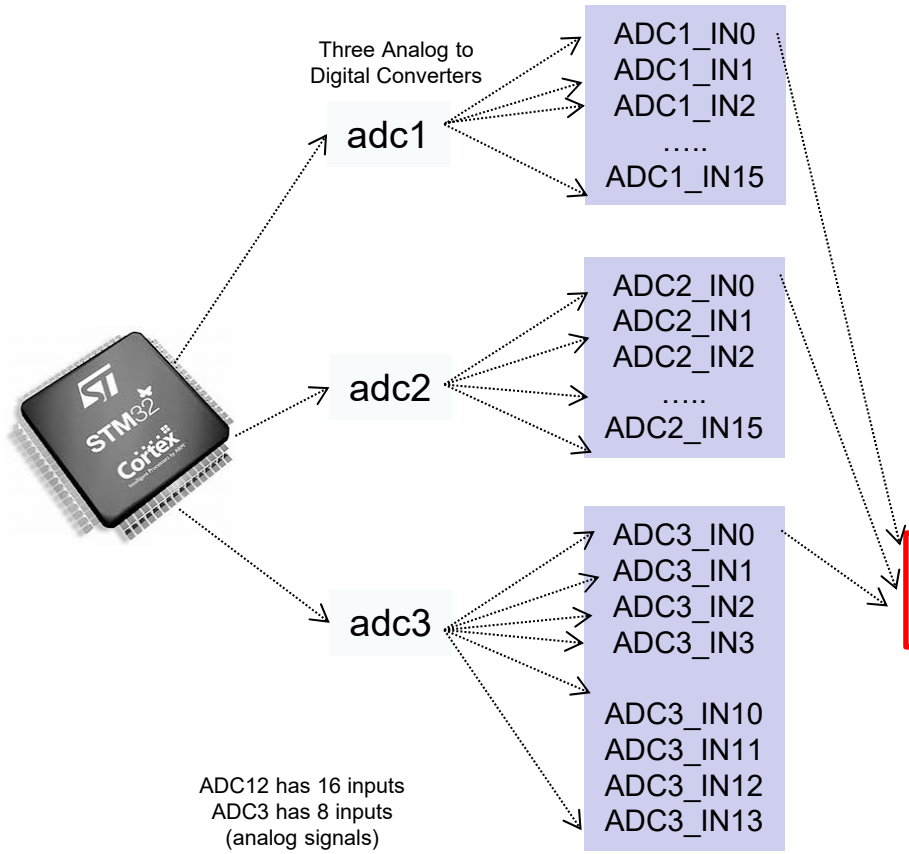


Table 5. High-density STM32F103xx pin definitions (continued)

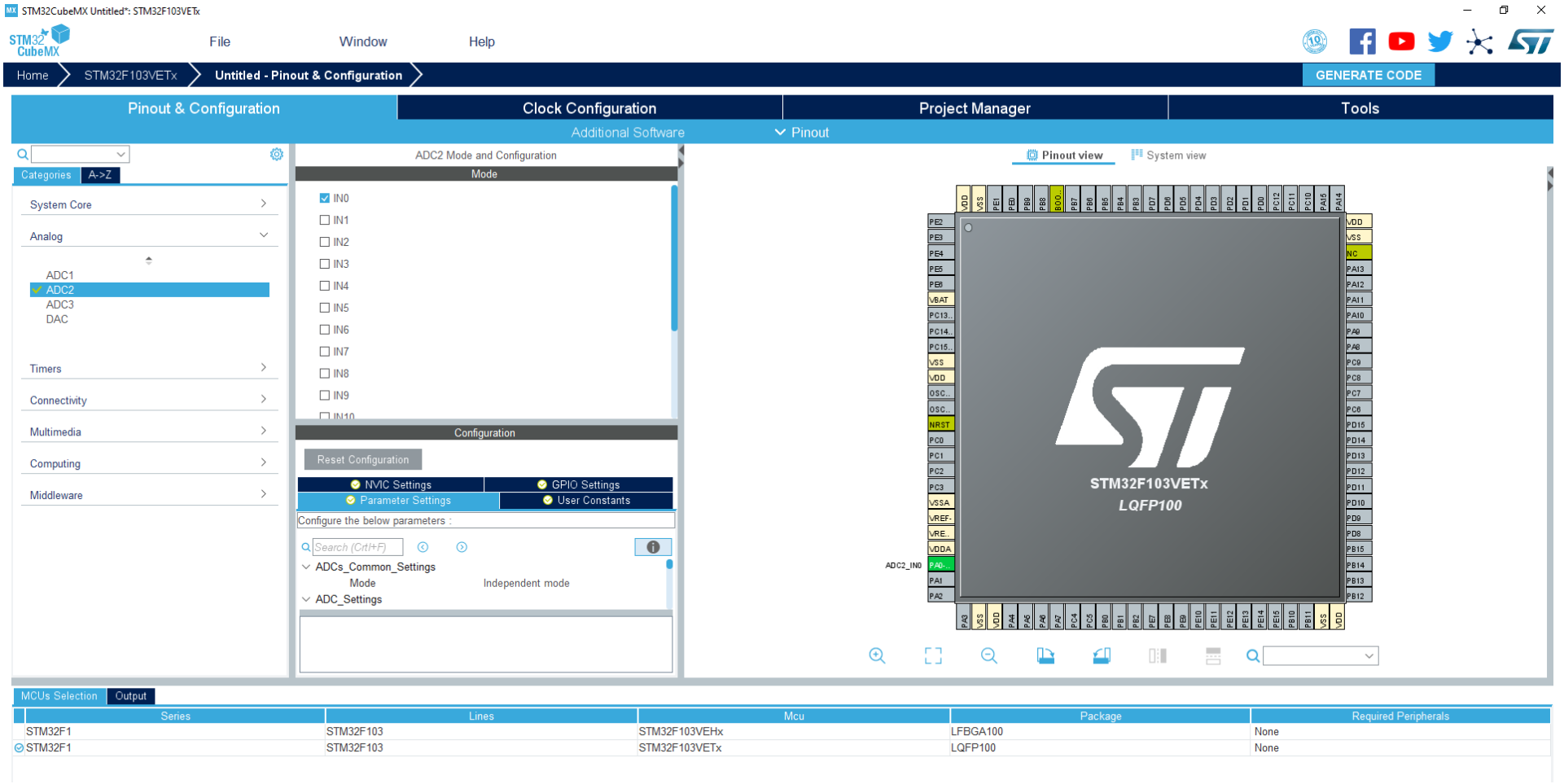
Pins						Pin name	Type(1)	I / O Level(2)	Main function(3) (after reset)	Alternate functions(4)	
BGA144	BGA100	WLCSP64	LQFP64	LQFP100	LQFP144					Default	Remap
K1	H1	-	-	20	31	V_{REF-}	S		V_{REF-}		
L1	J1	F7 (7)	-	21	32	V_{REF+}	S		V_{REF+}		
M1	K1	G8	13	22	33	V_{DDA}	S		V_{DDA}		
J2	G2	F6	14	23	34	PA0-WKUP	I/O		PA0	WKUP/USART2_CTS(8) ADC123_IN0 TIM2_CH1_ETR TIM5_CH1/TIM8_ETR	
K2	H2	E6	15	24	35	PA1	I/O		PA1	USART2_RTS(8) ADC123_IN1/ TIM5_CH2/TIM2_CH2(8)	

Notation: ADC123_IN0

CubeMX version 5.20

A graphical tool that allows a very easy configuration of STM32 microprocessors.

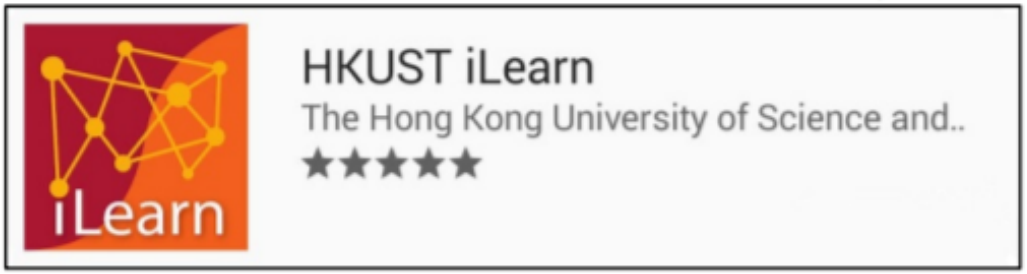
STM model: STM32F103VE, LQFP100



In class activity: Design architecture of an embedded system

In-class activities

For Android devices, search **HKUST iLearn** at Play Store.



For iOS devices, search **HKUST iLearn** at App Store.

Topic 4 – Question 1

Now, we have got a designed gadget.

How many switches are there?

Design architecture of an embedded system

- Let's go through 6 components in the design process

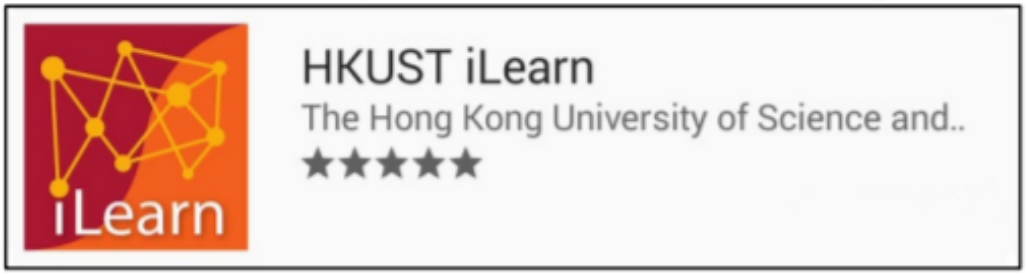
Products

Description	Choices in this course
Abstract idea of project (Define the functionality of the system)	Many
Data format / representation	Many
Programming Language	C-language
Communication Protocol	Many
Physical connection (Pins assignment)	Many
Hardware devices (Microcontroller, Peripherals)	Microcontroller: STM32 ARM Platform Peripherals: 7 switches, LCD, Buzzer, …

Components

In-class activities

For Android devices, search **HKUST iLearn** at Play Store.



For iOS devices, search **HKUST iLearn** at App Store.

Topic 4 – Questions 2-6

More about connecting a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

Physical Devices	Pin Assignment	Signal Type	Initialization (Configuration)	Signals at Physical connection
Micro Switch	General Purpose Input & Output	Input / ~~Output~~	General Purpose IO setting	On/Off

Data format

Initialization

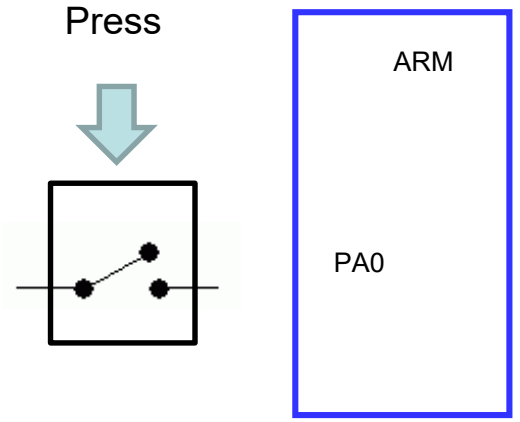
implementation

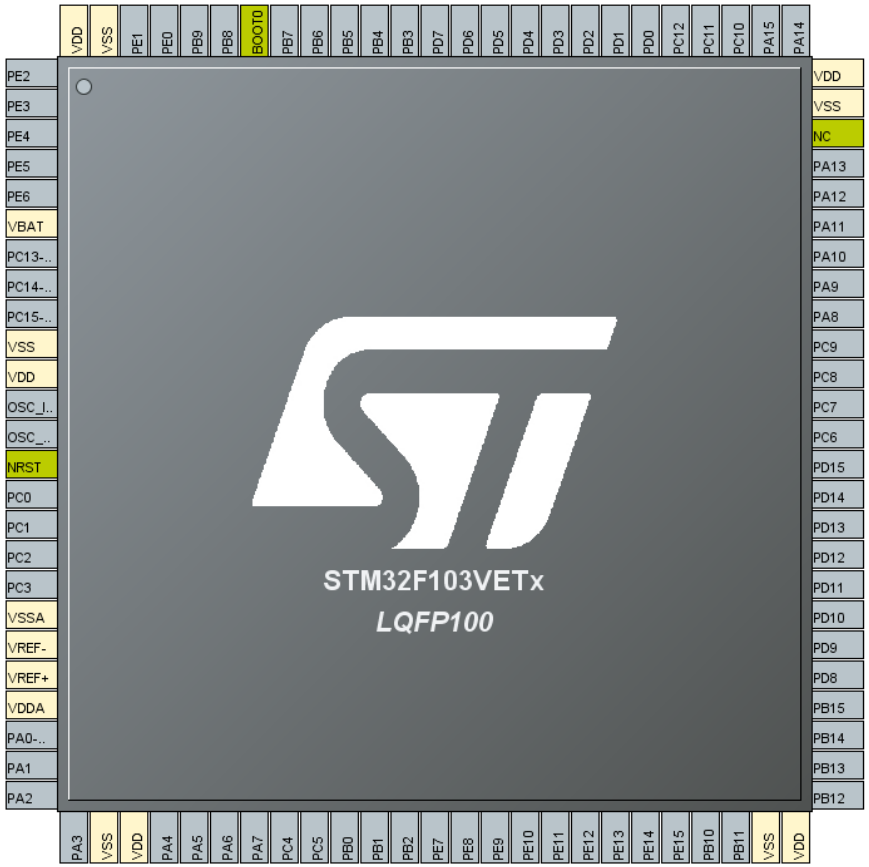
Programming Language

After assigning a pin for the switch, ……

Example: Connect a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)



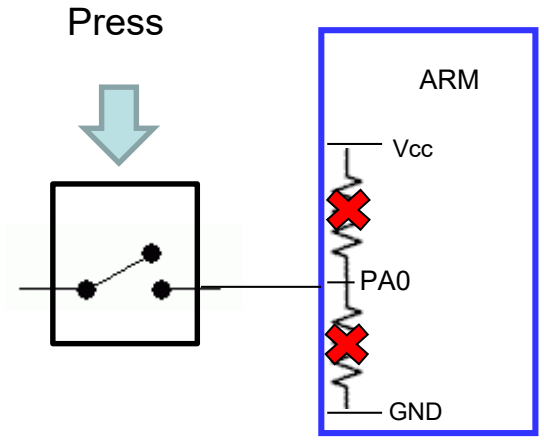


Example: Connect a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

What is the signal format of the digital signal?

Assume that PA0 is an open circuit.

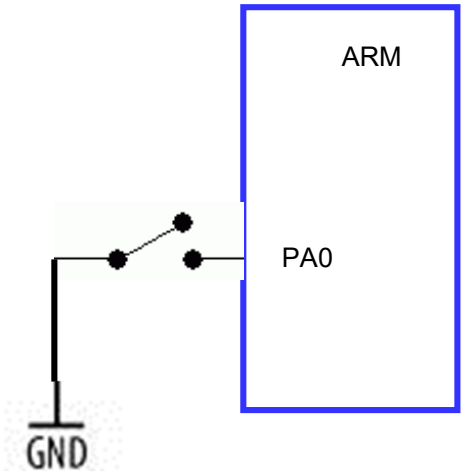


Status	Voltage at PA0	Digital Signal
Pressed (ON)		
Released (OFF)		

Example: Connect a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

Assume that PA0 is an open circuit.

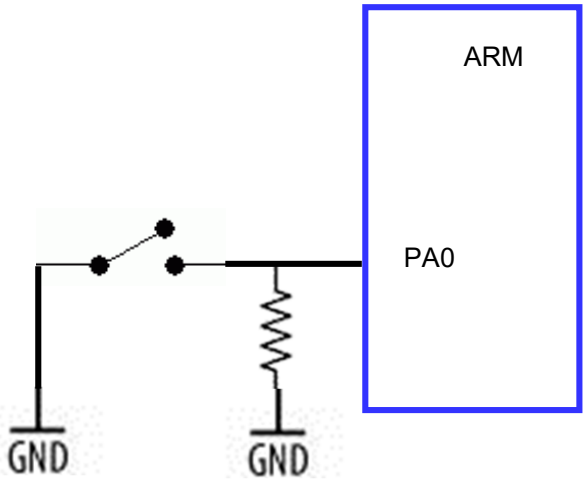


Status	Voltage at PA0	Digital Signal
Pressed (ON)		
Released (OFF)		

Example: Connect a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

Assume that PA0 is an open circuit.



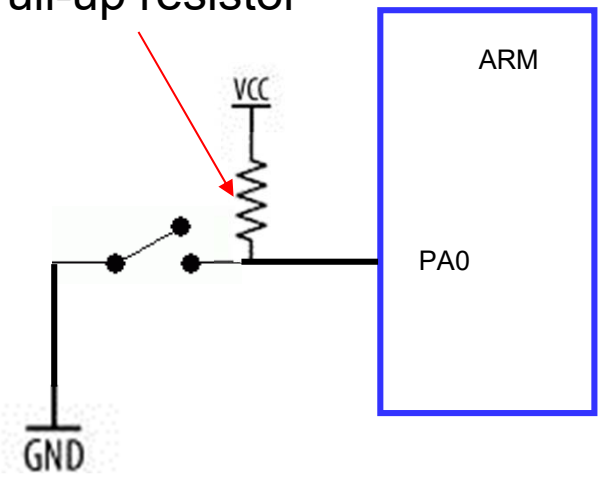
Status	Voltage at PA0	Digital Signal
Pressed (ON)		
Released (OFF)		

Example: Connect a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

Assume that PA0 is an open circuit.

Pull-up resistor

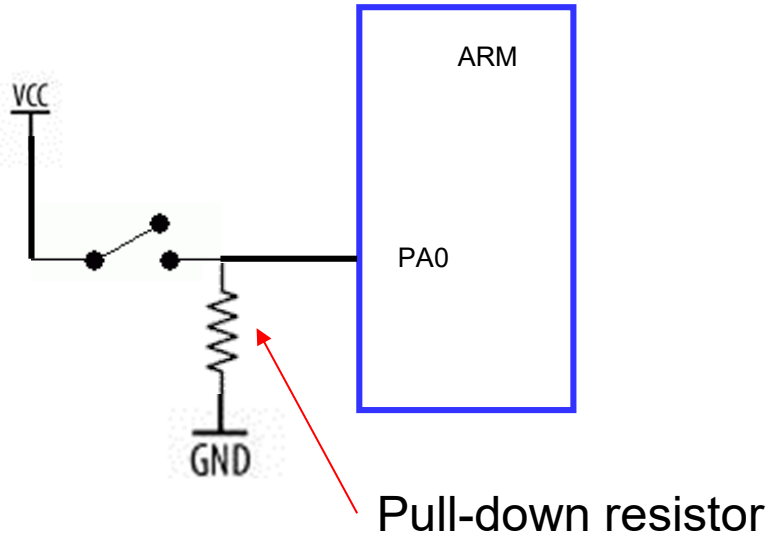


Status	Voltage at PA0	Digital Signal
Pressed (ON)		
Released (OFF)		

Example: Connect a switch to MCU board

Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

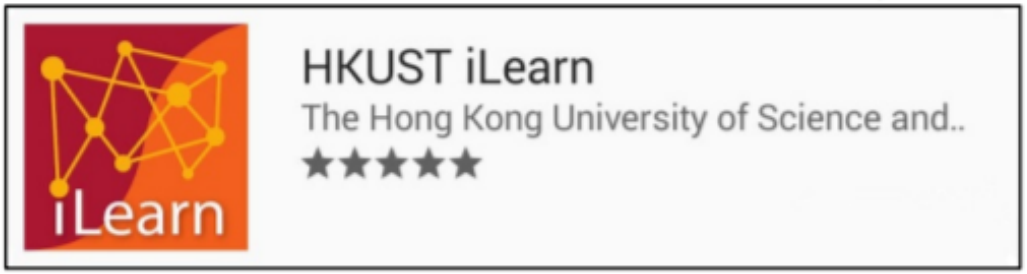
Assume that PA0 is an open circuit.



Status	Voltage at PA0	Digital Signal
Pressed (ON)		
Released (OFF)		

In-class activities

For Android devices, search **HKUST iLearn** at Play Store.



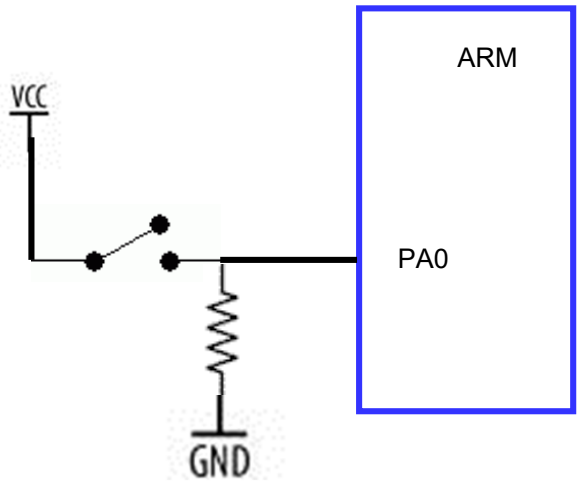
For iOS devices, search **HKUST iLearn** at App Store.

Topic 4 – Question 7

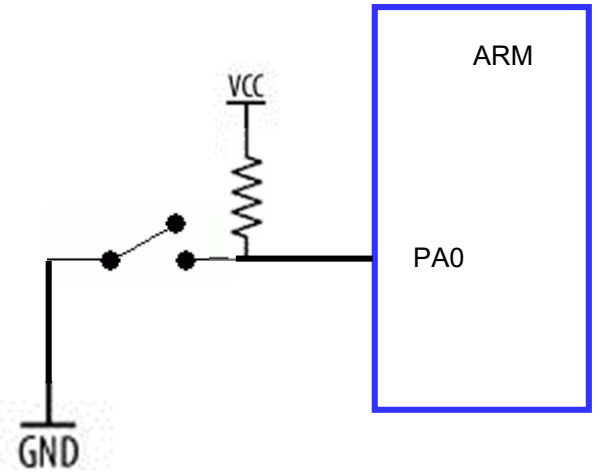
In-class activity – Question 9

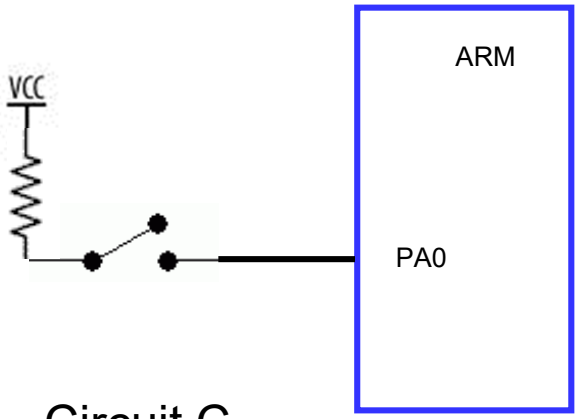
Assume that PA0 is an open circuit.

Circuit A

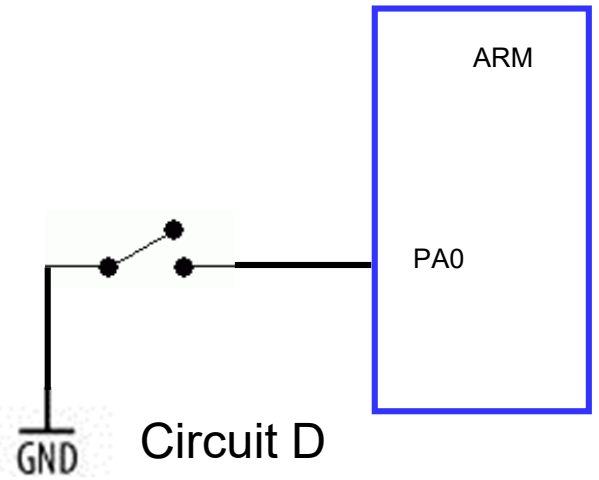


Circuit B





Circuit C

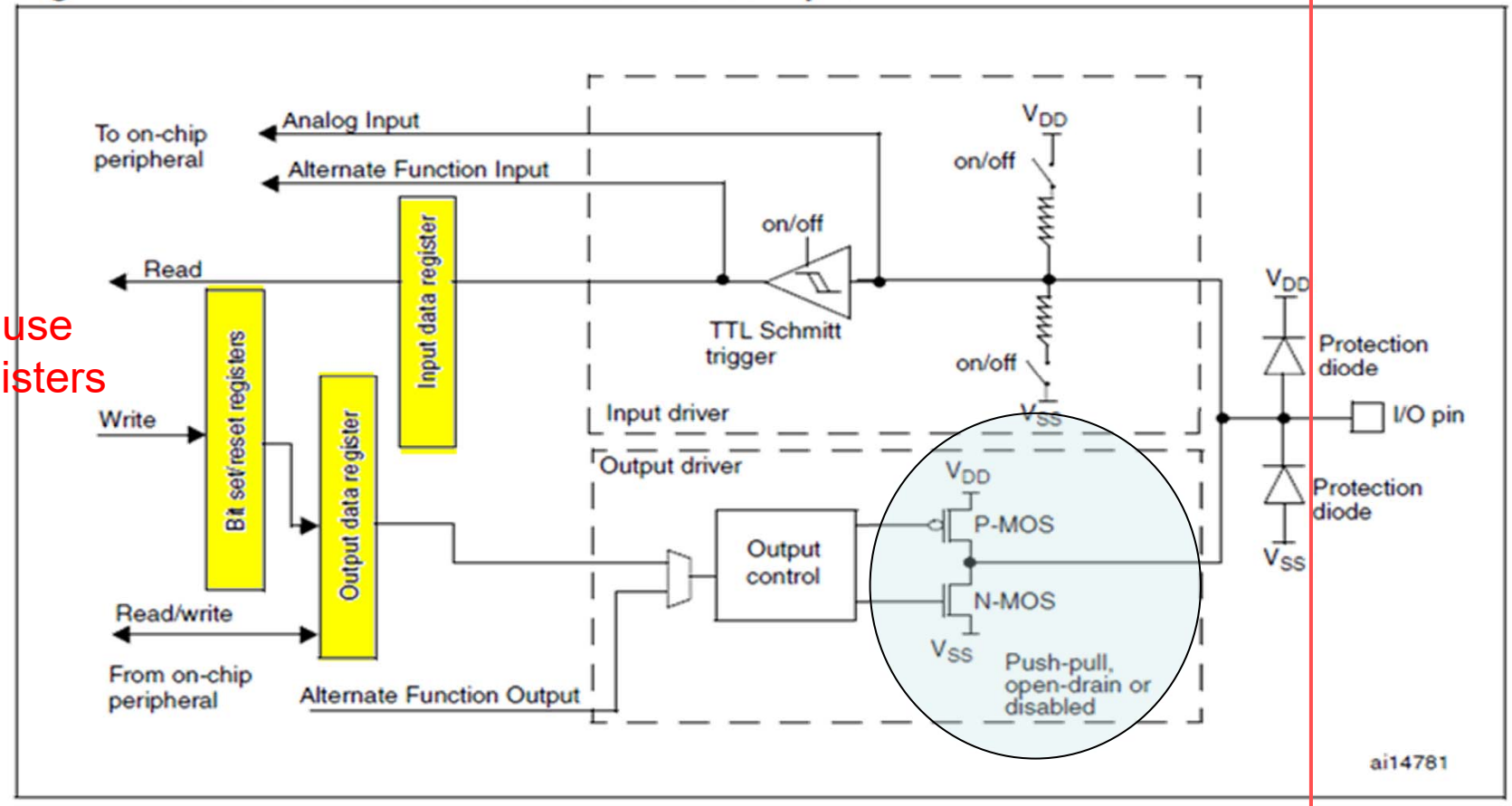


Circuit D

Example: General-purpose I/O (GPIO)

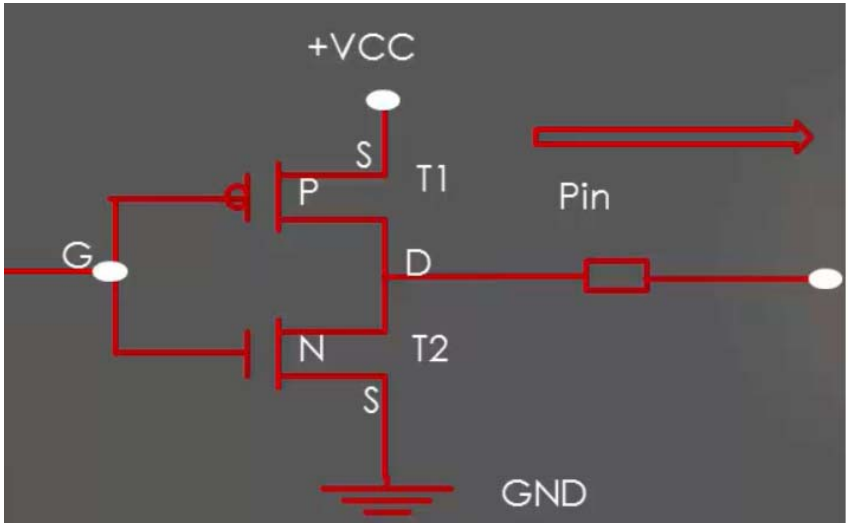
We will use these registers

Figure 13. Basic structure of a standard I/O port bit #



: STM32_Reference_Manual.pdf

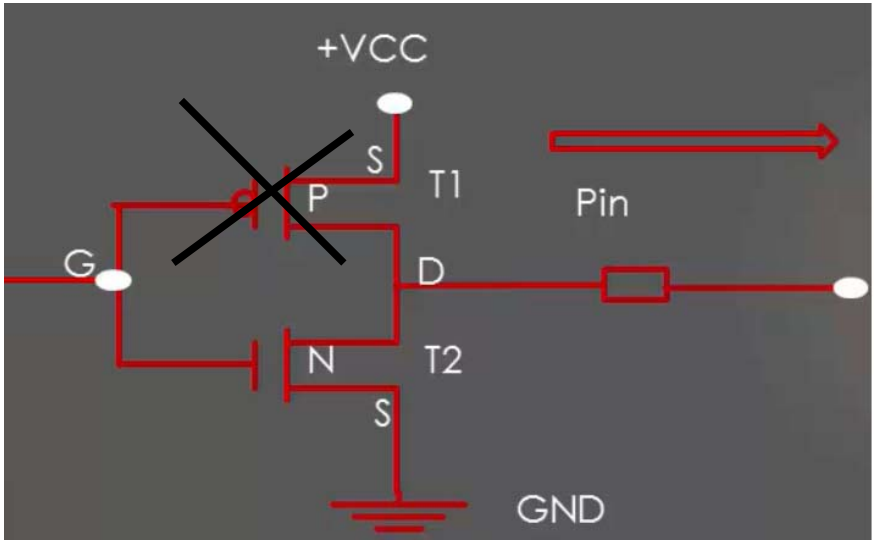
GPIO – Push-Pull Configuration



- When you enable GPIO port by default, its pin will be in input mode.
- If you set any pin as the output mode, then by default it will be in push-pull configuration.
- Push-pull means: Output will be pulled actively between low and high by using two transistors.

- The top transistor will be ON when the output has to be driven high.
- The bottom transistor will turn ON when the output has to go low.

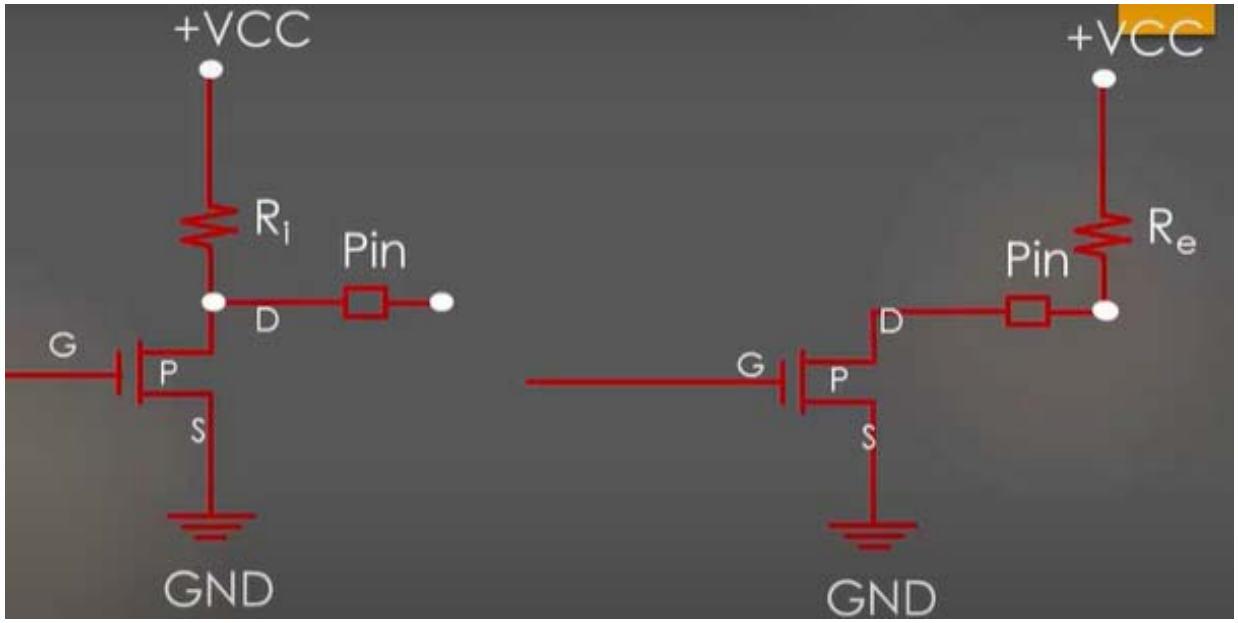
GPIO – Open Drain Configuration



- Output mode with Open drain means the top P-MOS transistor (T1) is deactivated.
- When T2 is ON, output pin will be pulled to ground (0 V).
- When T2 is OFF, drain D is floating, output pin is floating → open drain.

- Open drain configuration can only pull down the pin, but cannot pull it up.
- Hence there are only two states – either GROUND or FLOAT → Useless.
- Open drain can be made useful by activating pull up using pull up resistor internal/external.

GPIO – Open Drain Configuration with Pull-up Resistor



Internal pull up resistor

External pull up resistor

You have to do the GPIO configuration for activating the internal pull up resistor.

Example: Connect a switch to MCU board

- All GPIO pins have an internal weak pull-up and weak pull-down which can be activated or not when configured as input.

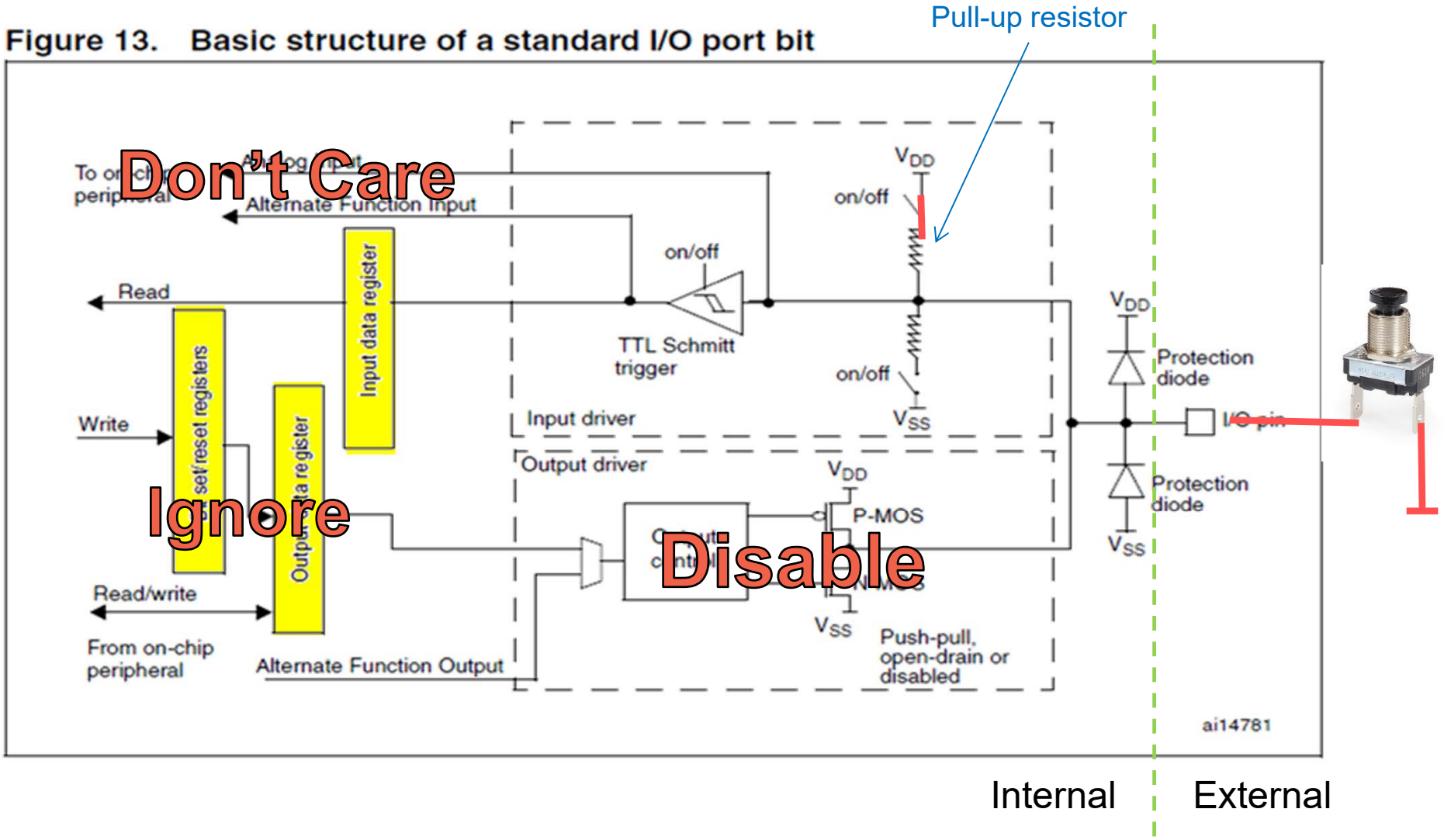
Table 20. Port bit configuration table #

Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-pull	0	0	01 10 11 see *Table 21*		0 or 1
	Open-drain		1			0 or 1
Alternate Function output	Push-pull	1	0			don't care
	Open-drain		1			don't care
Input	Analog	0	0	00		don't care
	Input floating		1			don't care
	Input pull-down	1	0			0
	Input pull-up					1

Initialization

: STM32_Reference_Manual.pdf

Example: General-purpose I/O (GPIO)



Example: Connect a switch to MCU board

- In the programming, we have two steps
  - Initialization for digital input
    - Set CNF1 = 1, CNF0 = 0, Mode1/0 = 00
    - Set PxODR = 0 or 1 (depends on the connection of switch)
  - Implementation
    - Write a routine for reading signal (Polling or Interrupt Driven)

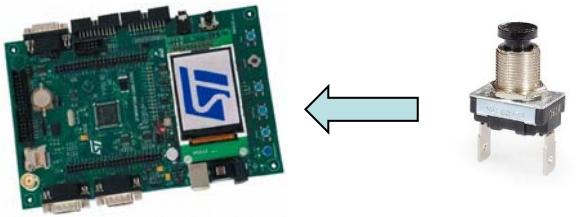
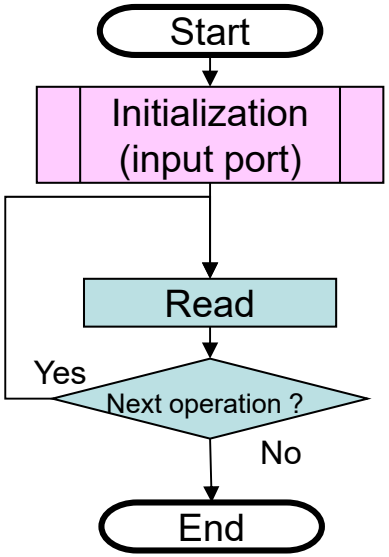


Table 21. Output MODE bits

MODE[1:0]	Meaning
00	Reserved
01	Max. output speed 10 MHz
10	Max. output speed 2 MHz
11	Max. output speed 50 MHz

Table 20. Port bit configuration table #

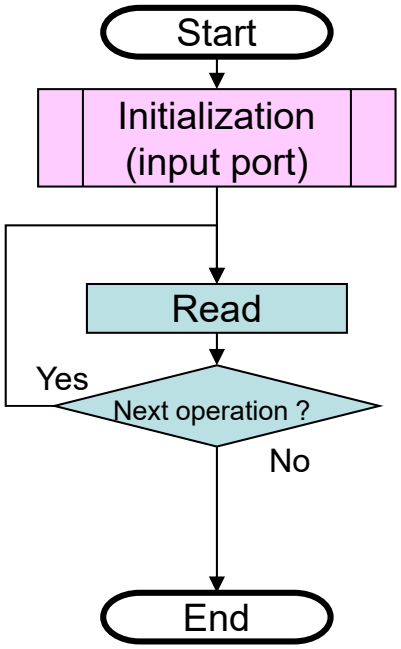
Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-pull	0	0	01 10 11 see *Table 21*		0 or 1
	Open-drain		1			0 or 1
Alternate Function output	Push-pull	1	0			don't care
	Open-drain		1			don't care
Input	Analog	0	0	00		don't care
	Input floating		1			don't care
	Input pull-down	1	0			0
	Input pull-up					1

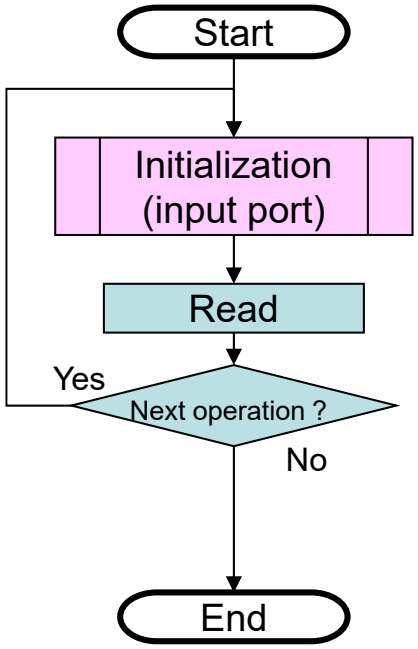


: STM32_Reference_Manual.pdf

Example: Connect a switch to MCU board

- Implementation
  - Write a routine for reading signal (Polling or Interrupt Driven)
  - Either polling or interrupt driven I/O, we can implement following flow charts.



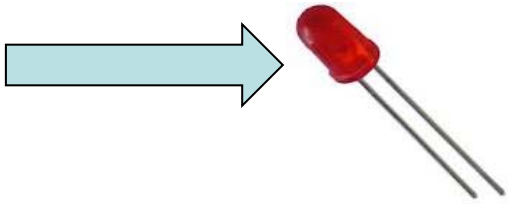


Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

Note:
The data is stored in Data Input Register.

Which one do you prefer?

Example: Connect a LED to MCU board



Description
Abstract idea of project (Define the functionality of the system)
Data format / representation
Programming Language
Communication Protocol
Physical connection (Pins assignment)
Hardware devices (Microcontroller, Peripherals)

Physical Devices	Pin Assignment	Signal Type	Initialization (Configuration)	Signals at Physical connection
LED	General Purpose Input & Output	~~Input~~ / Output	General Purpose IO setting	On/Off

Data format

Initialization

implementation

Programming Language

Example: Connect a LED to MCU board

- When configured as output, the value written to the Output Data register (GPIOx_ODR) is output on the I/O pin.
- It is possible to use the output driver in Push-Pull mode or Open-Drain mode (only the N-MOS is activated when outputting 0).

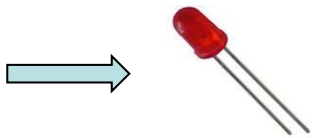


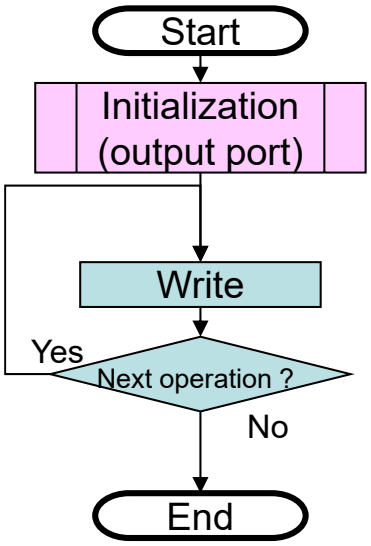
Table 21. Output MODE bits

MODE[1:0]	Meaning
00	Reserved
01	Max. output speed 10 MHz
10	Max. output speed 2 MHz
11	Max. output speed 50 MHz

#

Table 20. Port bit configuration table

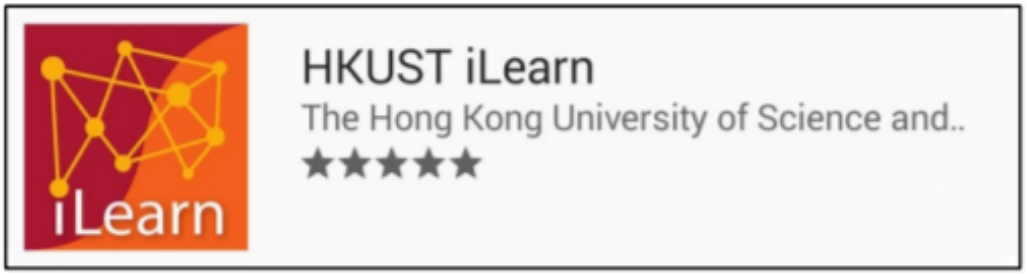
Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-pull	0	0	01 10 11 see *Table 21*		0 or 1
	Open-drain		1			0 or 1
Alternate Function output	Push-pull	1	0			don't care
	Open-drain		1			don't care
Input	Analog	0	0	00		don't care
	Input floating		1			don't care
	Input pull-down	1	0			0
	Input pull-up					1



: STM32_Reference_Manual.pdf

In-class activities

For Android devices, search **HKUST iLearn** at Play Store.

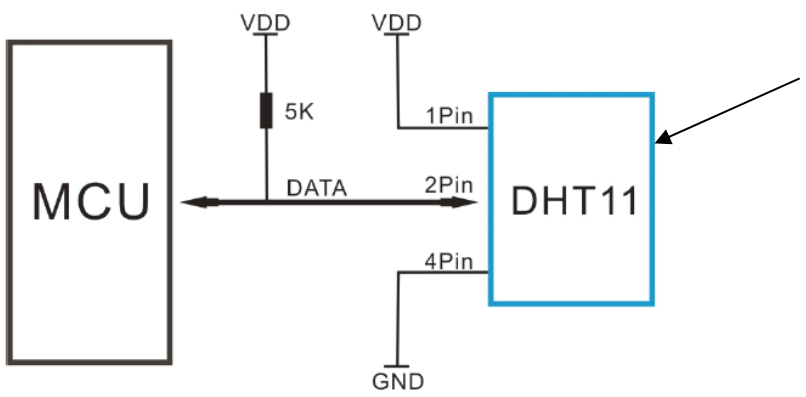


For iOS devices, search **HKUST iLearn** at App Store.

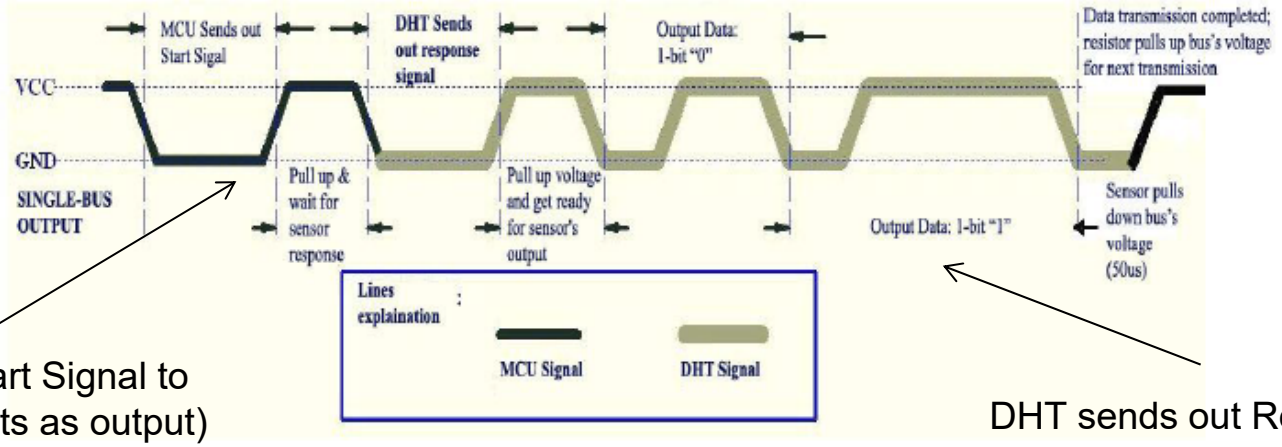
Topic 4 – Questions 8, 9

Example: Bi-directional I/O port

- How can we initialize a general purpose I/O pin as a bi-directional one?
- If yes, how does the flow chart look like?



Temperature & Humidity Sensor

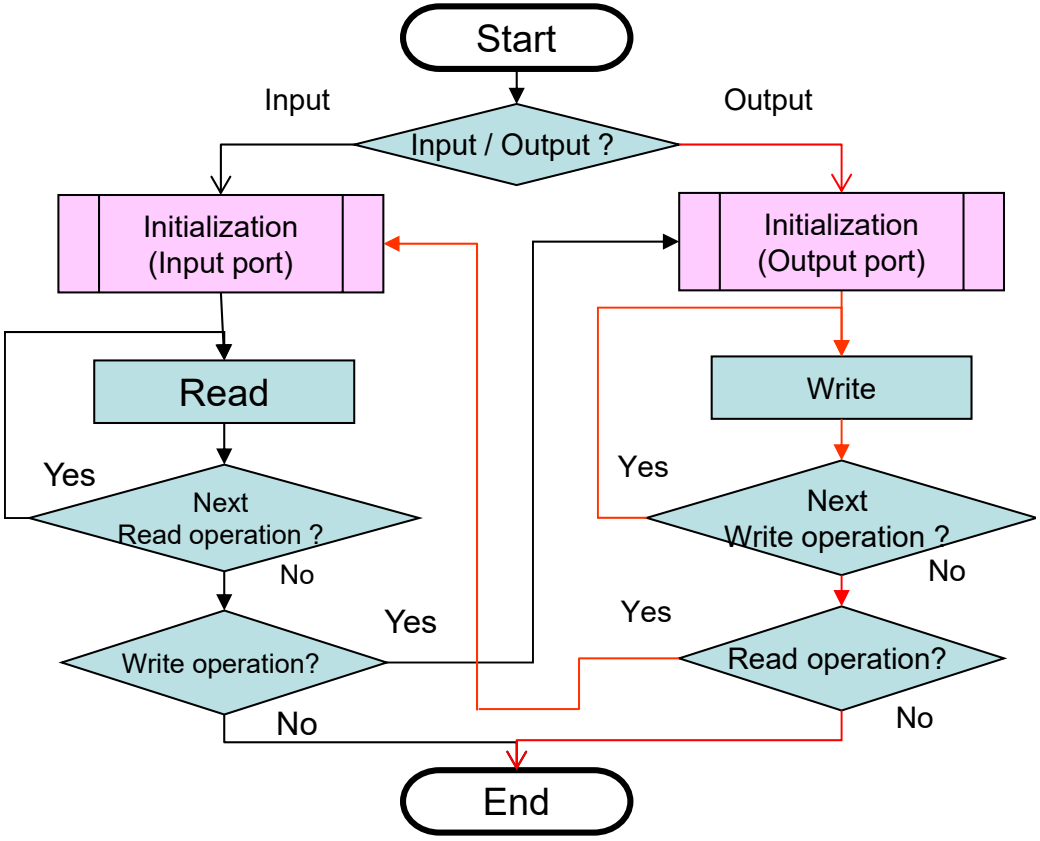


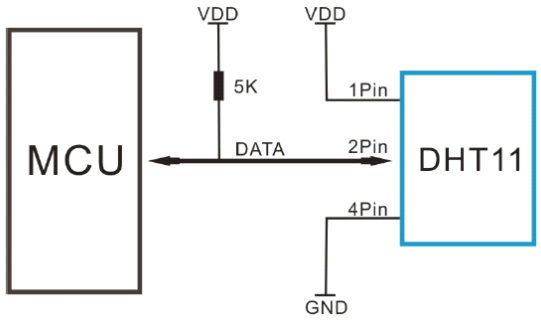
MCU sends out Start Signal to DHT11 (MCU pin acts as output)

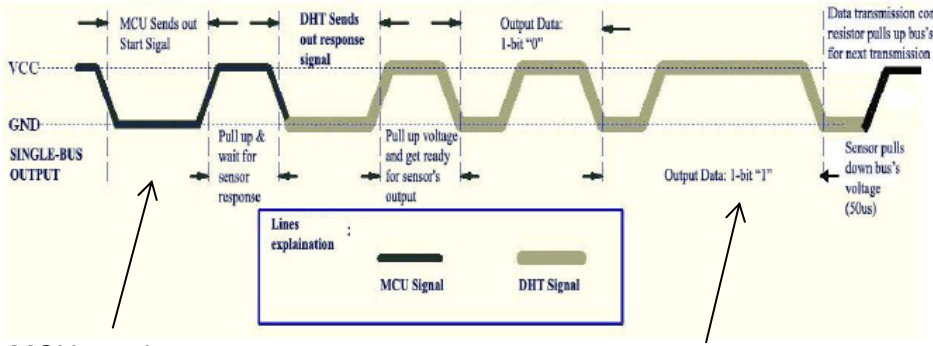
DHT sends out Response Signal to MCU (MCU pin acts as input)

Example: Bi-directional I/O port

- Can we initialize a general purpose I/O pin as a bi-directional one?
- If yes, how does the flow chart look like?





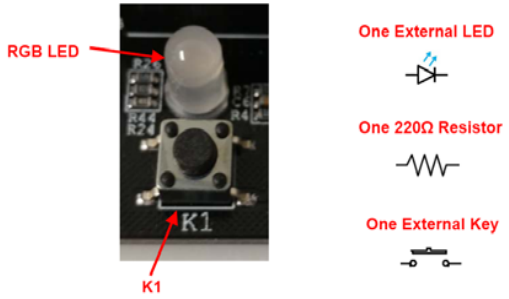


MCU sends out Start Signal to DHT11 (output)

DHT sends out Response Signal to MCU (input)

Review of laboratory experiments

- Can you fill in information in the following table?

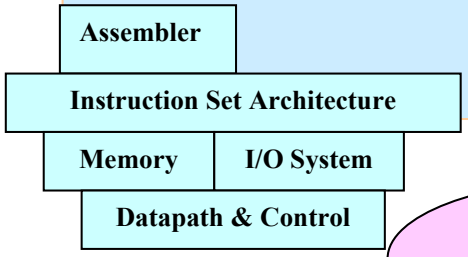


Description	Lab 2	Hints
Features		What are the objectives of this lab?
Data format		Is it digital or analog signal at the input port? Is it digital or analog signal at the output port? If it is digital signal, how many bits are there?
Programming Language		Refer to programming language of STM32
Communication Protocol		For the output signal, how do you handle it? For the input signal, which methods do you use for detecting the status of switches (pressed or released)?
Physical connection		Which type of physical pins do you use? (GPIO, ADC, etc ...) Which mode (pull-up or push-down) do you use?
Hardware devices (Microcontroller, Peripherals)		Which components do you have?

Reflection (self-evaluation)

- Do you
  - Describe the features of ARM micro-controller ?
  - Illustrate other examples of the pin definitions of multiple features?
  - Describe the bus architecture?
  - Understand the memory organization and its map?
  - Introduce the CRC calculation unit?
  - Configure General Purpose I/Os for reading 3 digital input signals and 4 digital output signals?
  - The difference between pull-up and push-down configurations?

Course Overview



Introduction to Embedded Systems

More about Embedded Systems

Basic Computer Structure

MCU Main Board

Digital and Analog Interfacing

USART, IEEE1394, USB, I2C and SPI

Microcontroller Structure

A/D Port

Serial Port

CPU

External Memory Port

External Interrupt Port

Interrupt Organization

Timer and Counter

External Timer Port

Simple I/O Port

Buffering and Direct Memory Access (DMA)

Memory, Interfacing to Memory, Memory Timing and applications

Interfacing LCD

Motor Interfacing

In this course, STM32 is used as a driving vehicle for delivering the concepts.

To be covered	In progress	Done