IOT ONLINE COURSE

Fundamentals of IoT

F-IOT-2c: Introduction to IoT hardware

Prof. Congduc Pham http://www.univ-pau.fr/~cpham Université de Pau, France













http://di	y.waziup.io		<u>IOT COURSES</u>	
Introduction Arduino LoRa IoT × +		WAZIUP IoT Courses		
← → C (a cpham.perso.ur	iv-pau.fr/LORA/HUBIQUITOUS/solution-lab/arduino-lora-tutorial/introdu	For users who wants to gain knowledge on IoT in a step-by-step lecture mode, we have defined the following curriculum with materials from both existing sources and specific materials produced by WAZIUP/WAZIHUB project.		
Introduction • Microcontrollers •	INTRODUCTION	F-IOT-1a: What is loT ?	F-IOT-1b: Introduction to Basic Elec	ctronics
■Arduino IDE>♥Sensors>■Advanced boards>#MQTT♦Node-RED>♥LoRa communication>■WaziDev board>■Solution Lab>	This online tutorial on Arduino, Sensors, IoT and LoRa technol by University of Pau, France, in the context of the WAZIUP® funded by the European Union in the H2020 research progra this online tutorial is to provide comprehensive and guided th in training, hackathons, bootcamps, entrepreneur's days th WAZIUP/WAZIHUB across Africa. The main contributors are Muhammad Ehsan and Congduc Pham [®] . Our main current re networks and IoT but this tutorial first start with basic of Arc programming to understand sensing systems that are the fou Internet-of-Things (IoT) concepts. Then in a second step, we protocols and technologies with a focus on LoRa radio techno- low-cost, long-range and energy-efficient IoT devices.	 HAZI I IoT and Big Data Platfi Intel IoT What Does The Inte Edureka Internet of Things (I Geospatial IoT IoT- What is Ii IBM Think Academy How It V F-IOT-1b: Introduction to I HAZI I Introduction To Basic Electroni Basic Electronics - Instructable HAZI I Introducing physical set 	HAZI → Introduction To Basic Electronics Introduction To Basic Electronics - Makers Basic Electronics - Instructables HAZI → Introducing physical sensors, part HAZI → Introducing physical sensors, part HAZI → F-IOT-2: IoT ecosystem and hardwa HAZI → F-IOT-2a: Wireless Communicatio HAZI → F-IOT-2b: Understanding IoT Dev HAZI → F-IOT-2c: Introduction to IoT hard F-IOT-3: Introduction to Arduino ID	Spaces 1 2 I re on Essentials ices, Architecture & Ecosystem dware
✓ Collapse sidebar		LIOT 9. LIAT ADAGAMATINA AND Sist=PI OH32runalvirula OlfP5kAQQ/RaO9k	 Introduction to Arduino IDE - YouTube ₩AZI Presentation of the Arduino IDE ₩AZI Setting up the Arduino IDE F-IOT-4: WAZIUP IoT ecosystem ₩AZI F-IOT-4: WAZIUP Open Technologies 	gies for Low-cost IoT







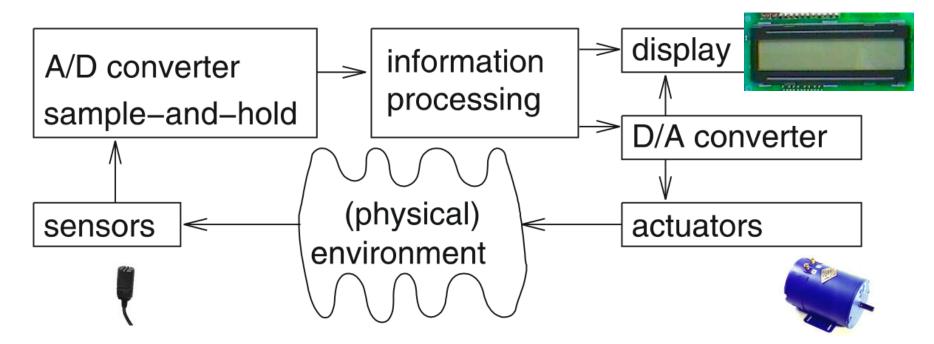
Prof. Congduc Pharr http://www.univ-pau.fr/~cpha

3





 IoT device can be viewed as a simple Embedded System Hardware which is frequently used in a loop ("hardware in a loop"):



From "An Embedded System Overview" by Dr. Eng. Amr T. Abdel-Hamid

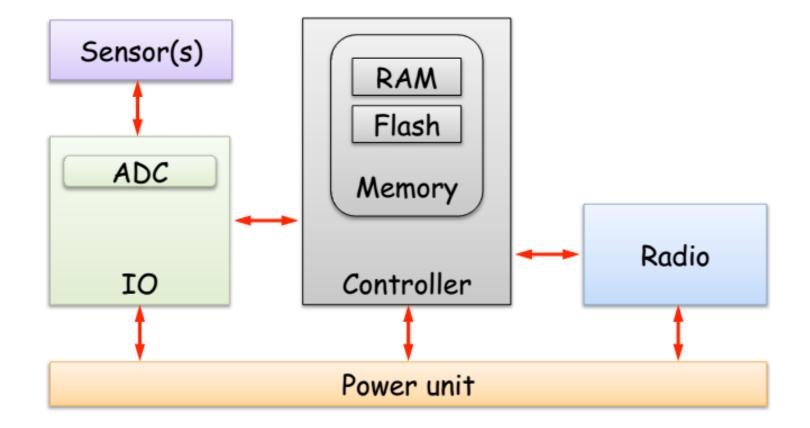




- Dependability
- Single-functioned (dedicated System)
 - Executes a single program, repeatedly
- Tightly-constrained (Efficient)
 - Low cost, low power, small, fast, etc.
- Reactive and real-time
 - Continually reacts to changes in the system's environment
 - Must compute certain results in real-time without delay
- Dependability and real-time are not mandatory for IoT

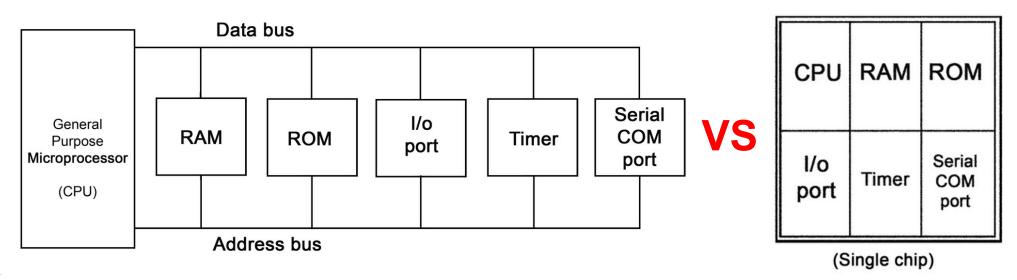






Microprocessors & Microcontrollers (Waziup)

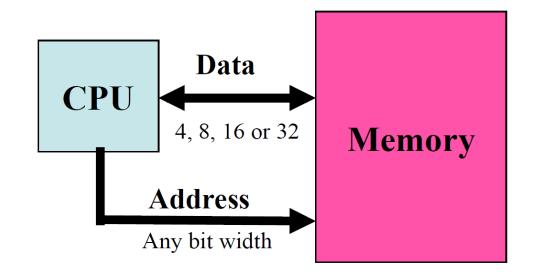
- A microprocessor unit (MPU) is a processor on one silicon chip
- A microcontroller unit (MCU) is a microprocessor with some added circuitry on one silicon chip
- Microcontrollers are used in embedded computing and most IoT devices are based on microcontrollers







• The number of bits describing the data path defines the Microcontroller Bit Definition



Common Memory Types in MCUs (WARZiupo)

• RAM

- Most microcontrollers have little amount of internal RAM. Generally, 1 Kbytes ("embedded" in MCUs) is a common amount, although some more powerfull microcontrollers can have more
- RAM is typically used to store variables (global, local, stack,...) and data of your program
- So use the small amount of RAM wisely!
- EEPROM electrically-erasable-and-programmable ROM
 - Internally, they are similar to EPROMs, but the erase operation is accomplished electrically, rather than by exposure to ultraviolet light
 - higher cost and write cycles are also significantly longer than RAM
 - EEPROM are typically used to store permanent data
- Flash memory





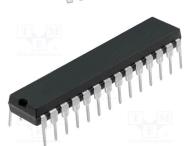
- Flash memory devices are
 - high density,
 - low cost,
 - nonvolatile,
 - fast (to read, but not to write), and
 - electrically reprogrammable.
 - Write/Erase large blocks of bytes
- EEPROM is similar to flash memory but the principal difference is that
 - EEPROM requires data to be written or erased one byte at a time whereas flash memory allows data to be written or erased in blocks.
- Flash memory is typically used in MCUs for storing program code





The high-performance Microchip picoPower 8-bit AVR RISC-based microcontroller combines 32KB ISP flash memory with read-while-write capabilities, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts.





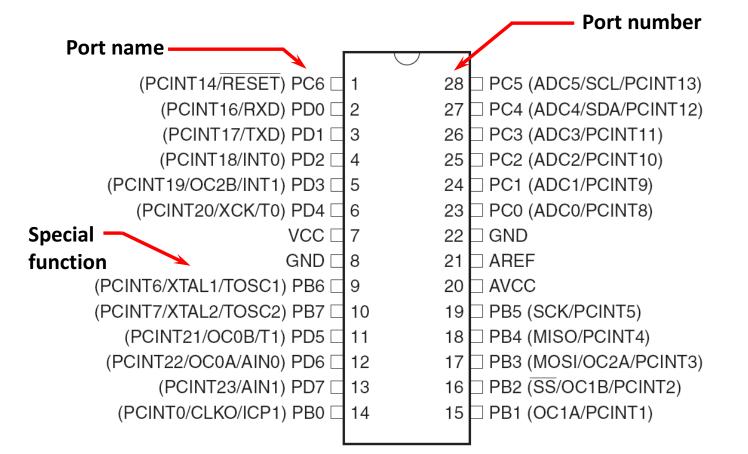
Program Memory Type	Flash		
Program Memory Size (KB)	32	Timers	2 x 8-bit, 1 x 16-bit
CPU Speed (MIPS/DMIPS)	20	Number of Comparators	1
SRAM Bytes	2,048	Temperature Range (C)	-40 to 85
Data EEPROM/HEF (bytes)	1024	Operating Voltage Range (V)	1.8 to 5.5
Digital Communication Periphe	1-UART, 2-SPI, 1-I2C	Pin Count	32
Capture/Compare/PWM Periph	1 Input Capture, 1 CCP, 6PWM	Low Power	Yes

https://www.microchip.com/wwwproducts/en/ATmega328p





 Ports/pins are communication channels through which information flows into or out of the microcontroller



Prof. Congduc Pham http://www.univ-pau.fr/~cphan





Input

 When you want to take information from the external world (sensors) into the MCU

- Output
 - When you want to change the state of something outside the MCU (turn a led/motor on or off, etc.)
- Pins default to input direction on power-up or reset
- Your program can set or change the directionality of a pin at any time



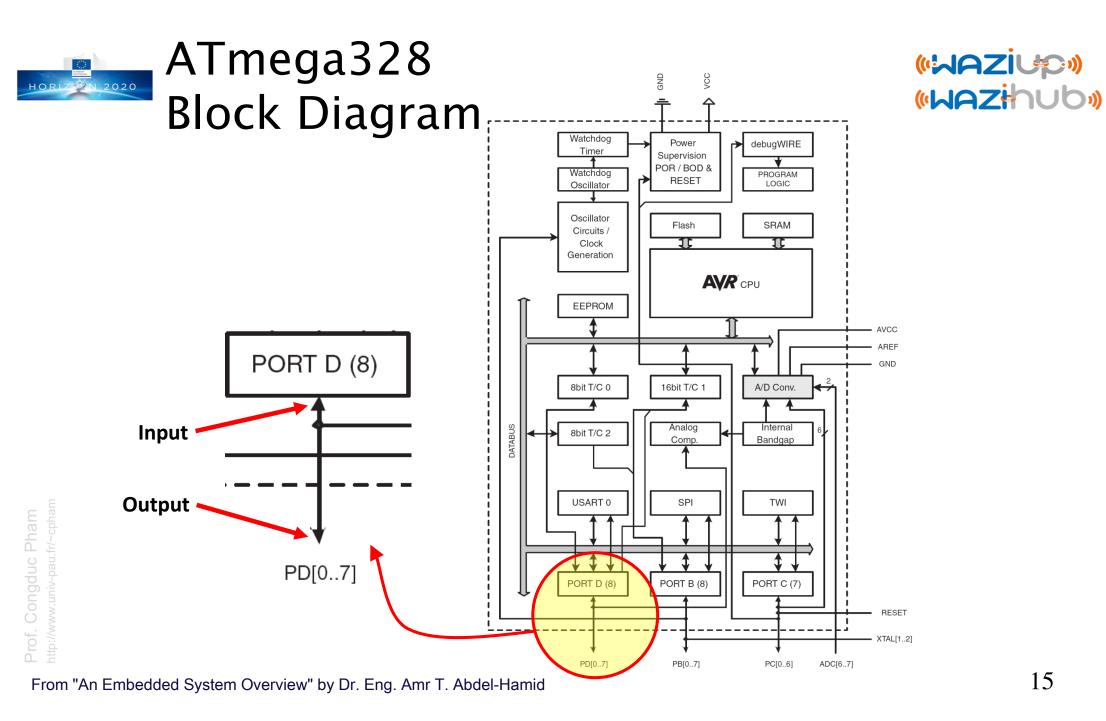
Pin Voltages



• Microcontrollers are fundamentally *digital* devices.

For digital IO pins, information is 'coded' in two discrete states:

- HIGH or LOW (logic: 1 or 0)
- Voltages
 - ⊙TTL
 - 5 V (for HIGH)
 - \odot 0 V (for LOW)
 - 3.3 V CMOS
 - 3.3 V (for HIGH)
 - \odot 0 V (for LOW)



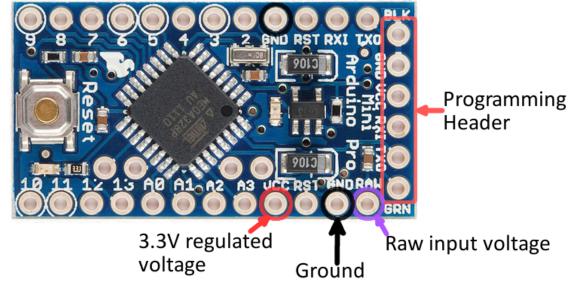
From µcontroller to µcontroller bo



- Although a µcontroller can be standalone, it is usually mounted on a board with additional electronics parts
 Leds

 - Voltage regulators
 Reset button

 - Serial-USB
- Ex: Arduino Pro Mini
 - See next slide on Arduino boards
- You can use an existing voltage Ground microcontroller board to build your IoT device or make your own board by integrating
 - yourself the microcontroller





µcontroller pins to µcontroller board pins



- Mapping of μcontroller pins to μcontroller board pins is not always obvious, nor standardized
- So need to look at the pin-out schematic

	Name Power CND Control	ADC PWM Serial Ext Inter PC Inter					Progr		no F				EV-1	111	.4)
	Arduino	PC Interi	rupt												
P	Port	Misc					GND CTS VCC TX To FTDI break FTDI Hea]						
							GND GND VCC RX	ard]						
							žo o o o) O Og							
		PCINT17	TXD	PD1	DI	тхо	GND VCC RX		RAW	RAW					
		PCINT16	RXD	PD0	DO	RXI			GND	GND					
			PCINT14	PC6	Reset	RST		• 📓 🔊 🔘	RST	Reset	PC6	PCINT14			
					GND	GND			VCC	VCC					
		PCINT18	INTO	PD2	D2	2				A3/D17		ADC3	PCINTII		
OC2B	PCINT19	INT1	8-bit	PD3	D3	3	Onthe	1,80	A2	A2/D16	PC2	ADC2	PCINTIO		
	ХСК	то	PCINT20	PD4	D4	4	Q ⊤)	C CEO		A1/D15		ADC1	PCINT9		
F1	OC0B		8-bit	PD5	D5	5		11 CO		A0/D14	PC0	ADC0	PCINT8		
AINO	OC0A	PCINT22	8-bit	PD6	D6	6		ri O	13	D13	PB5	SCK	PCINT5	LED	
		IN1	PCINT23	PD7	D7	7	Reset		12	D12	PB4	MISO	PCINT4		
	CLKO	ICP1	PCINTO	PB0	D8	8		7. 70		D11	PB3	8-bit	MOSI	PCINT3	OC2A
	OC1A	PCINTI	8-bit	PB1	D9	9		4 4 20	10	D10	PCB2	8-bit	SS	PCINT2	OC1B
							Reset Bu	utton							
								A5	A5/D19	PC5	ADC5	SCL	PCINT13		
								A4	A4/D18	PC4	ADC4	SDA	PCINT12		
								Δ7	Δ7						







WHAT IS ARDUINO?

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects.



ARDUINO BOARD

Arduino senses the environment by receiving inputs from many sensors, and affects its surroundings by controlling lights, motors, and other actuators.

_	_	_	
\sim		_	

void	<pre>setup() {</pre>
}	
void	<pre>loop() {</pre>
}	

ARDUINO SOFTWARE

You can tell your Arduino what to do by writing code in the Arduino programming language and using the Arduino development environment.













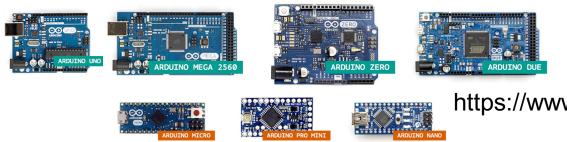








- The Arduino project was started at the *Ivrea Interaction Design Institute* in Italy and targets students, tech enthousiasts and hobbyists
- It is now one of the de facto standard and other boards usually offers Arduino compatibility if they want to attract users
- There are also a lot of Arduino clones from mostly China manufacturers and they are working just fine
- The basic Arduino boards (Uno, MEGA, Due, Mini, Nano) are now completed with a lot of other boards

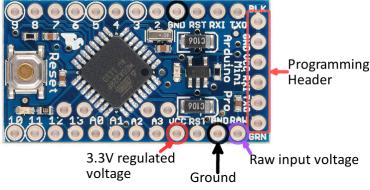


https://www.arduino.cc/en/Main/Products

Which Arduino board for IoT?



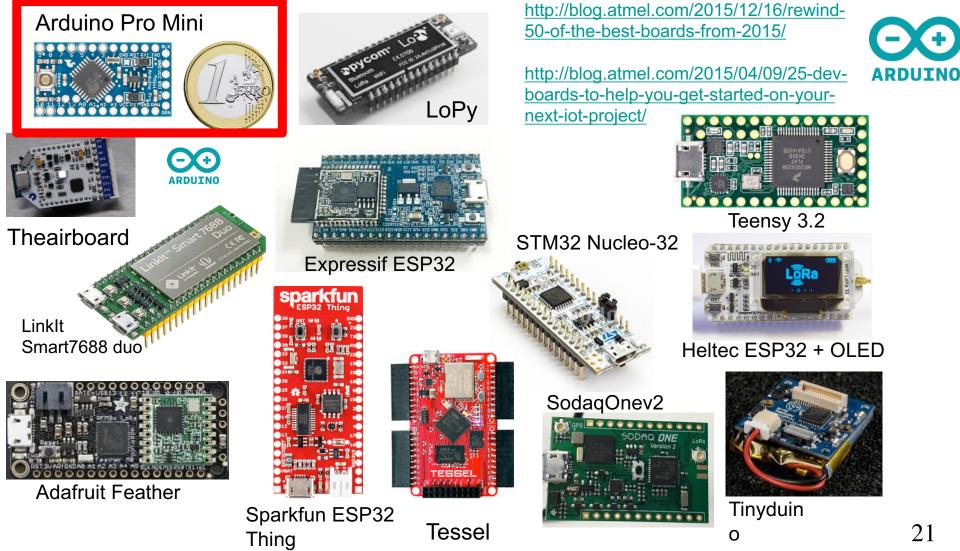
- Depending on the application constraint, any board can be considered suitable for IoT provided that it can have radio capabilities (natively or with additional hardware)
- However, in many IoT applications, size and energy consumption are very important issues so large boards and boards with energy-hungry radio such as WiFi are not really suitable



• The previous Arduino Pro Mini which exists in 3.3V and 8MHz version is a great board for early integration phase



Large ecosystem, still growing... «Waziup»



21





- Open source IDE to program Arduino boards
- Arduino boards can be programmed in C++ -like language
- Additional functions are introduced to manage I/O and other features such as interrupts, delays

External Interrunts

	External interrupts		
Time	attachInterrupt()		
delay()	detachInterrupt()		
delayMicroseconds()			
micros()	Interrupts		
millis()	interrupts()		
	noInterrupts()		
analogRead()			
)	Communication		
	Serial		
eference/en/	Stream		
	delay() delayMicroseconds() micros() millis()		

Structure of an Arduino Program



• An arduino program == 'sketch'

- Must have:
 - setup()
 - ⊙loop()
- setup()
 - \odot configures pin modes and registers
- Ioop()
 - runs the main body of the program forever
 - \odot like while(1) {...}
- Where is main()?
 - Arduino simplifies things
 - \odot Does things for you

/* Blink - turns on an LED for DELAY_ON msec, then off for DELAY_OFF msec, and repeats BJ Furman rev. 1.1 Last rev: 22JAN2011 */

#define LED_PIN 13 // LED on digital pin 13 #define DELAY_ON 1000 #define DELAY_OFF 1000

void setup()

// initialize the digital pin as an output: pinMode(LED_PIN, OUTPUT);

// loop() method runs forever,
// as long as the Arduino has power

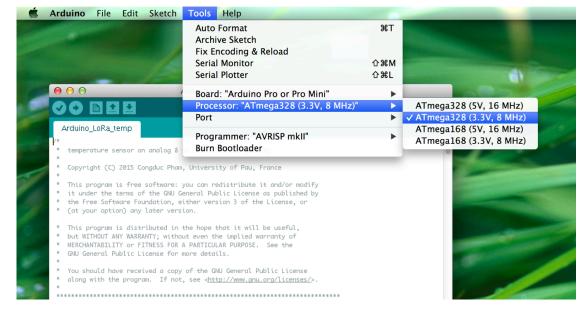
void loop()

digitalWrite(LED_PIN, HIGH); // set the LED on delay(DELAY_ON); // wait for DELAY_ON msec digitalWrite(LED_PIN, LOW); // set the LED off delay(DELAY_OFF); // wait for DELAY_OFF msec



«WAZİUD» «WAZihub»

	temperature sensor on analog 8 to test the LoRa gateway
	Copyright (C) 2015 Congduc Pham, University of Pau, France
* * * *	This program is free software: you can redistribute it and/or modify it under the terms of the GUU General Public Licanse as published by the Free Software foundation, either version 3 of the License, or (at your option) and later version.
* * *	This program is distributed in the hope that it will be useful, but WITHOUT ANY MARANATY, without even the implied morranty of MERCHMINELITY or FITHISS FOR A PARTICULAR PURPOSE. See the GNU General Public Liensee for more details.
*	You should have received a copy of the GNU General Public License along with the program. If not, see <u><<u>http://www.anu.org/licenses/</u>>.</u>
111.	NHPORTANT Jease uncomment only 1 choice
11 11	it seems that both MopeRF and Modtronix board use the PA_BOOST pin and not the RFO. Therefore, for the poards we set the initial power to 'x' and not 'M'. This is the purpose of the define statement
	uncomment if your radio is an HopeRF REM92W or REM95W
11	fine RADIO_RFM2_95 nocoment if your notio is a Modtronix inAir98 (the one with +2048m features), if inAir9, leave comme Jefine RADIO_INAIR98
	INDAPTANT



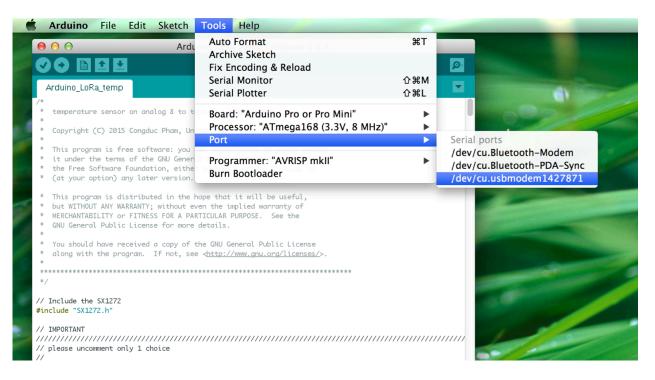
Select the Arduino board you have, here Arduino Pro Mini 3.3V 8MHz version

Then, click on the « verify » button





(«WAZİUP») («WAZihub»)



Prof. Congduc Pham

Connect the USB end to your computer and the USB port should be detected in the Arduino IDE. Select the serial port for your device. It may have another name than what is shown in the example. Then click on the « upload » button

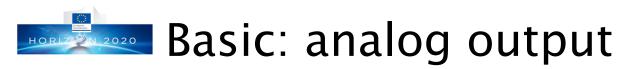




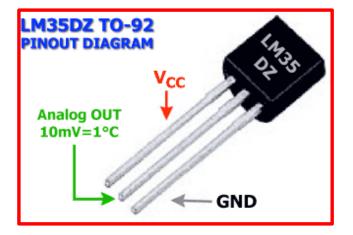


• Main feature of IoT!









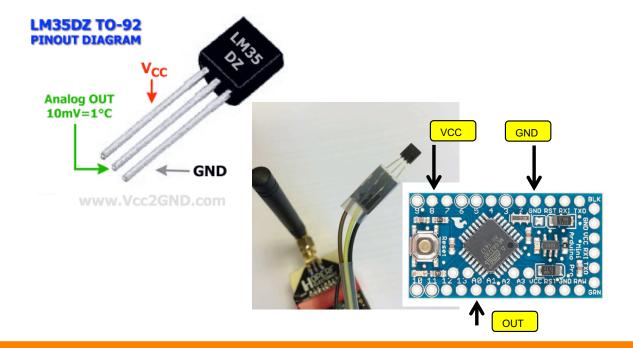
Vcc is typically 5V or 3.3V, assuming 3.3V here

If 0 means 0V and 1024 means 3300mV (10-bit resolution) then 3300mV/1024=3.22mV is the granularity of the measure

A digital value of 100 means 100*3.22mV=322mV If the sensor output is 10mV/1°C then the physical temperature is 322mV/10mV=32.2°C



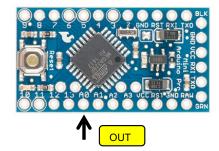




GND should be connected to one of the board's GND, VCC can be connected to the output of digital pin 8 for instance (to get power) and the OUT pin can be connected to the analog A0 pin.





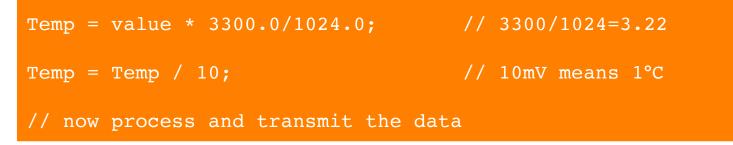


// sensor output connected to A0 analog pin

```
value = analogRead(A0);
```

// now need to convert to Celcius degree

And converting into Celcius







- Sensors with a digital output usually provide the sensed data in some advanced digital format, not only an analog value.
- In that case, additional libraries are often needed to read the sensor's data depending on the interface/protocol/bus
 - Serial UART (traditional serial line)
 - 1-Wire
 - I2C: Inter-Integrated Circuit, or TWI: Two-Wire Interface
 - SPI: Serial Peripheral Interface
- In many cases, there are both library and example provided with the sensor to read the data, so do not start from scratch!
- Example with the DHT22 sensor follows. You can search for "DHT22 Arduino" on the web





 The DHT22 is a temperature and humidity sensor with a digital output on pin 2



- There are various libraries available (search Arduino+DHT22 on the web)
- We use the library developed by Ben Adams: https://github.com/nethoncho/Arduino-DHT22





- Many sensors need a little preparation before you can safely connect them to a microcontroller board
- Usually, you may need to add some resistor and/or condensator to some pins or wires
- Consult the sensor datasheet or search the web on the required schematic for a safe and proper operation of the sensor





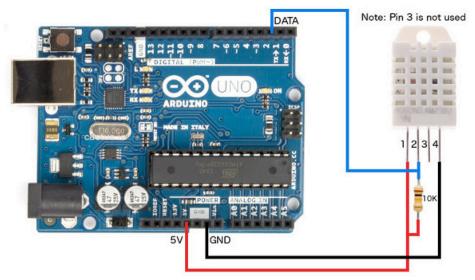
```
// include Ben Adams's DHT22 library
#include "DHT22.h"
```

```
DHT22* dht = NULL;
```

```
// provide the digital pin to which the
// DHT22 data pin is connected to. Here
// digital pin 2 on the Arduino
dht = new DHT22(2);
```

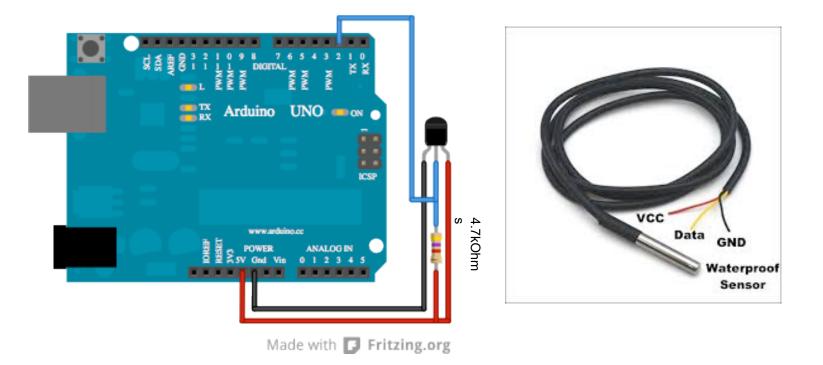
```
// start reading
dht->readData();
```

// then get both temperature and humidity
double temp=(double)dht->getTemperatureC();
double hum=(double)dht->getHumidity();







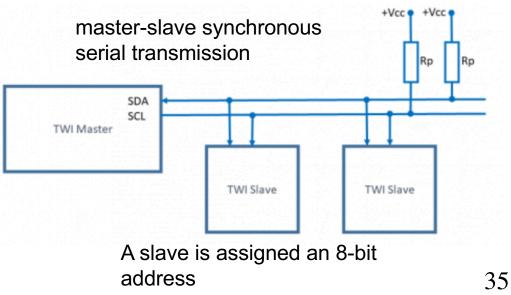






- I2C (Inter-Integrated Circuit) is one of the most popular communication protocol used in embedded systems. It has been designed by Philips for simple audio-video appliances controlled by the microprocessor
- There are many chips that can be connected to the processor with this interface which uses SDA (data) and SCL (clock)
 - EEPROM memory chips
 - RAM memory chips
 - AD/DA converters
 - Real-time clocks
 - Sensors (temperature, pressure, gas, air pollution)

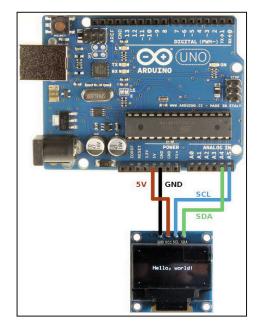
From iot-open.eu IoT course book



I2C example: connecting OLED



- Small OLED screens usually use I2C communication interface with SDA and SCL wire
- Just connect the corresponding wires to their matching pins on the Arduino. Usually SDA=A4 and SCL=A5 (100kHz-400kHz)



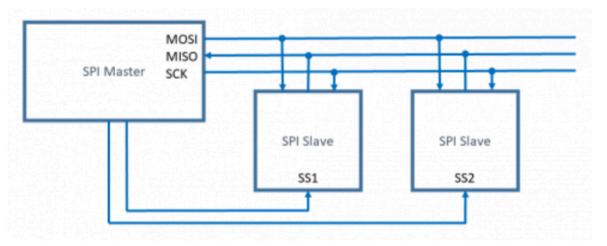
SHT Sensor Pins	Arduino Pins		
VCC	5V or 3.3V		
GND	GND		U日·U日 ^{菜单 2014-10-01} QQ
SDA	A4		TT GG
SCL	A5		
(PC	NT14/RESET) PC	6 🗌 1	28 🗆 PC5 (ADC5/SCL/PCINT13)
(PCINT16/RXD) PD	0 🗆 2	27 C PC4 (ADC4/SDA/PCINT12)
((PCINT17/TXD) PD	1 🗆 3	26 🗆 PC3 (ADC3/PCINT11)
(PCINT18/INT0) PD	2 🗌 4	25 🗆 PC2 (ADC2/PCINT10)
		·	. 36





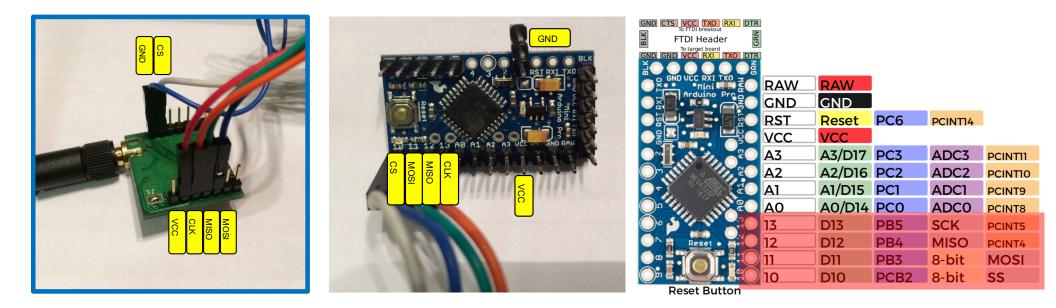
- SPI (Serial Peripheral Interface) is a synchronous serial interface and protocol that can transmit data with speed up to 20Mbps
- To communicate SPI uses three lines common to all of the connected devices, and one enabling line for every slave element

Line	Description	Direction
MISO	Master In Slave Out	$\text{peripheral} \rightarrow \text{uC}$
MOSI	Master Out Slave In	$uC \rightarrow peripheral$
SCK	Serial Clock	$uC \rightarrow peripheral$
SS	Slave Select	$uC \rightarrow peripheral$



SPI example: connecting radio mo

- Many radio modules use SPI interface to communicate with the host microcontroller
- Example with a LoRa radio module







- Most microcontrollers do not have display
- When connected to a computer, output can be displayed by the computer, mostly for debugging purposes
- In most cases, the connection is realized using serial interface (Universal Asynchronous Receiver Transmitter, UART)
- With the Arduino IDE, use
 - Serial.begin(38400);
 - Serial.print("The value is: ");
 - Serial.println(my_value);
- There are libraries or turnaround to have C-like printf function
- Or use sprintf then Serial.print





Arduino_LoRa_temp Arduino 1.6.6	and the second second second second second second second second second second second second second second second		
	Serial Monitor 👂		
rduino_LoRa_temp	000 /dev/ci	u.usbmodem1427871	
temperature sensor on analog 8 to test the LoRa gateway			Send
Copyright (C) 2015 Congduc Pham, University of Pau, France	Simple LoRa temperature sensor		
	Teensy31/32 detected SX1276 detected, starting		
This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by	SX1276 LF/HF calibration		
the Free Software Foundation, either version 3 of the License, or	 Get back previous sx1272 config		
(at your option) any later version.	Using packet sequence number of 40		
This program is distributed in the hope that it will be useful,	Setting Mode: state 0 Setting Channel: state 0		
but WITHOUT ANY WARRANTY; without even the implied warranty of	Setting Channel: state 0 Setting Power: state 0		
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.	Setting node addr: state 0		
and deneral Public Eldense for more details.	SX1272 successfully configured Reading 83		
You should have received a copy of the GNU General Public License	(Temp is 26.75		
along with the program. If not, see < <u>http://www.gnu.org/licenses/</u> >.	Sending \!#4#TC/26.74		
	Real payload size is 13		
	0K1		
nclude the SX1272	> waiting for 1 CAD = 62 > CAD duration 547		
lude "SX1272.h"	0K2		
MPORTANT	> RSSI -126 LoRa pkt seg 40		
	LoRa Sent in 1569		
lease uncomment only 1 choice	LoRa Sent w/CAD in 2116		
t seems that both HopeRF and Modtronix board use the PA_BOOST pin and not the	Packet sent, state 0 Reading 88		
oards we set the initial power to 'x' and not 'M'. This is the purpose of the	de(Temp is 28.36		
ncomment if your radio is an HopeRF RFM92W or RFM95W	Sending \!#4#TC/28.35 Real payload size is 13		
ine RADIO_RFM92_95	> CAD duration 547		
<pre>ncomment if your radio is a Modtronix inAir9B (the one with +20dBm features), efine RADIO_INAIR9B</pre>	OK1		
///////////////////////////////////////	> waiting for 6 CAD = 372 > CAD duration 547		
MPORTANT	0K2		
neoriani ///////////////////////////////////	> RSSI -125		
lease uncomment only 1 choice ine BAND868	✓ Autoscroll	No line ending \$	38400 baud 🗘

Prof. Congduc Pham http://www.univ-pau.fr/~cphan

You can see the output from the sensor if it is connected to your computer. Use the Arduino IDE « serial monitor » to get such output, just to verify that the sensor is running fine, or to debug new code. Use same baud rate (e.g. 38400 baud) that the one fixed by the program.





- Most of microcontroller boards have very basic capabilities
- Additional hardware implementing advanced features can be added by using shields
 - Ethernet, Bluetooth, 2G/3G, WiFi,...
 - Real-Time Clock
 - SD card
 - Audio, Display
 - High power relay
 - GPS
 - ...
- Most of the addons use I2C or SPI buses



Prof. Congduc Pham http://www.univ-pau.fr/~cphar



(«WAZİUP») («WAZİHUb»)

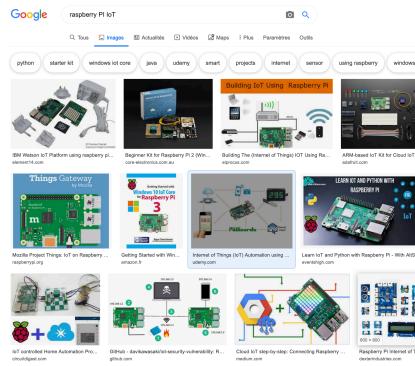
- ESP32
 - 32 bit
 - WiFi
 - Heltec ESP32 has embedded OLED screen
- Teensy3X
 - 32 bit, ARM Cortex
 - Lots of RAM (e.g. 96KB of RAM)
 - <u>https://www.pjrc.com/teensy/</u>

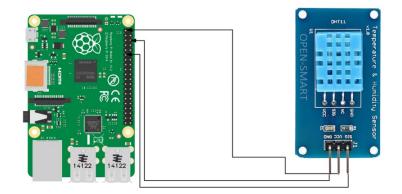


- Many boards are compatible with the Arduino IDE by installing dedicated plug-ins
- It is always possible to use another IDE (Eclipse) but sometime it is very difficult to make the setup

What about single-board computers about single-board computers about single-board computers about which we have a second

- Single-board computers such as the well-known Raspberry PI are oftenly advertised as IoT devices
- However, as they are very powefull, their high energy consumption make them not suitable in many IoT applications





Prof. Congduc Pham http://www.univ-pau.fr/~cphar





- Most of these single-board computers can run a complete Linux-based OS with various connectivity features
- They are definitely suitable as low-cost, flexible, versatile IoT gateways



Additional materials/tutorials (1) (WAZiupo)

- IoT course book from iot-open.eu
 - <u>http://iot-open.eu/download/io1-introduction-to-the-iot/</u>
- Slides
 - <u>https://github.com/CongducPham/tutorials/blob/master/Low-cost-LoRa-loT-step-by-step.pdf</u>
 - <u>https://github.com/CongducPham/tutorials/blob/master/Low-cost-LoRa-loT-outdoor-step-by-step.pdf</u>
- Videos
 - Build your low-cost, long-range IoT device
 - <u>https://www.youtube.com/watch?v=YsKbJeeav_M</u>
 - Extreme low-cost & low-power LoRa IoT for real-world deployment
 - <u>https://www.youtube.com/watch?v=2_VQpcCwdd8</u>
- Online tutorial/course
 - <u>http://cpham.perso.univ-pau.fr/LORA/WAZIUP/tuto/index.html</u>

Additional materials/tutorials (2) (WAZiupo)

- Resources from Arduino Community
 - <u>https://www.arduino.cc/en/Tutorial/HomePage</u>
- Resources from Adafruit
 - <u>https://learn.adafruit.com/</u>
- Resources from Instructables.com
 - <u>https://www.instructables.com/circuits/arduino/projects/</u>
- Resources from hackster.io
 - <u>https://www.hackster.io/arduino/projects</u>
- Resources from makerspaces.com
 - https://www.makerspaces.com/we-love-maker-educators/
- Resources from circuit digest
 - <u>https://circuitdigest.com/arduino-projects</u>
- Resources from Electronics Hub
 - <u>https://www.electronicshub.org/arduino-project-ideas/</u>
- And much more, just crawl the web!

IOT ONLINE COURSE

Fundamentals of IoT

Continue with F-IOT-3: Introduction to Arduino IDE F-IOT-4: Low-cost & Open-source Technologies for Low-Cost IoT

