IoT Lab 1. - syllabus

The goal of the measurement is to practice a typical IoT scenario when a hardware sensor is realized in the physical world, and its virtual pair exists in an IoT platform, somewhere in a cloud. The sensor measurements are collected and displayed in the IoT platform, as well as actions, taken in the platform, have a real physical effect on the sensor (actuator in this case).

Students have to build a sensor, which can communicate to the internet. They have to create its virtual pair in an IoT platform. Sensor data and actuator actions should be transmitted between the sensor and the platform. The sensor is an ESP8266 based microcontroller + Wifi radio combo with an NTC temperature sensor, a push button, and a led. The IoT platform is the myDevices.com Cayenne platform. The communication protocol is based on MQTT.

1. Preparation

The students get the following items before the measurement:

- Wemos D1 mini device (microcontroller + WiFi radio)
- NTC thermistor
- LED
- Resistors, wires
- Breadboard

The students should boot up the “IoT” image in the computer lab, where all the necessary tools are installed:

- Arduino IDE
- MQTT-spy and MQTT.FX

The student should register on the Cayenne site: https://mydevices.com/cayenne/signup/

Before the measurement, there is a small exam with 5 questions. The students have to answer at least 4 questions successfully in order to begin the measurements. The topics they should prepare from are:

- Analog, digital connection
- USART, SPI, I2C
- MQTT

Suggested materials (ignore them if they are too basic):

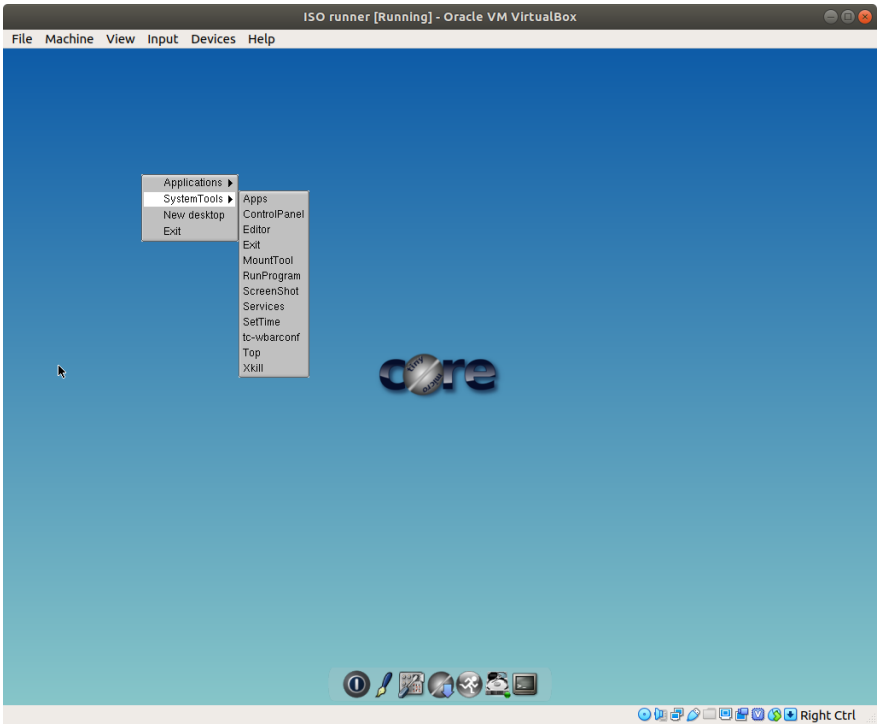
- https://learn.sparkfun.com/tutorials/what-is-a-circuit
- https://learn.sparkfun.com/tutorials/voltage-dividers
- https://learn.sparkfun.com/tutorials/analog-to-digital-conversion
- https://learn.sparkfun.com/tutorials/serial-communication
- https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi
- https://learn.sparkfun.com/tutorials/i2c
- https://randomnerdtutorials.com/what-is-mqtt-and-how-it-works/

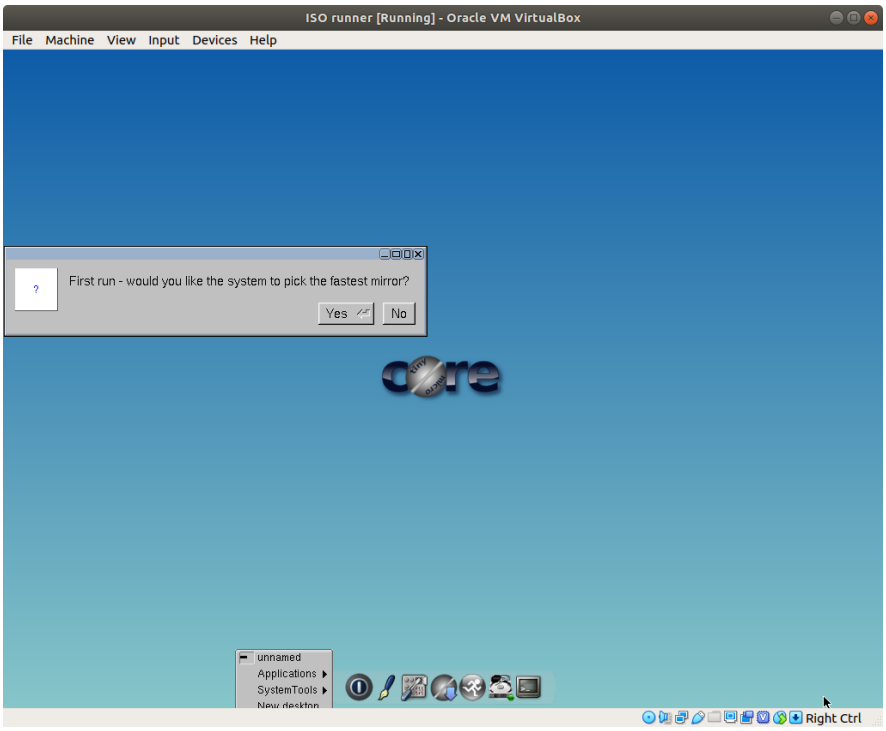
The student has to register into the lab's Google classroom in order to fill out the exam form and in order to submit the measurement report. The classroom code is provided at the beginning of the laboratory exercise.

2. Preparation with VirtualBox

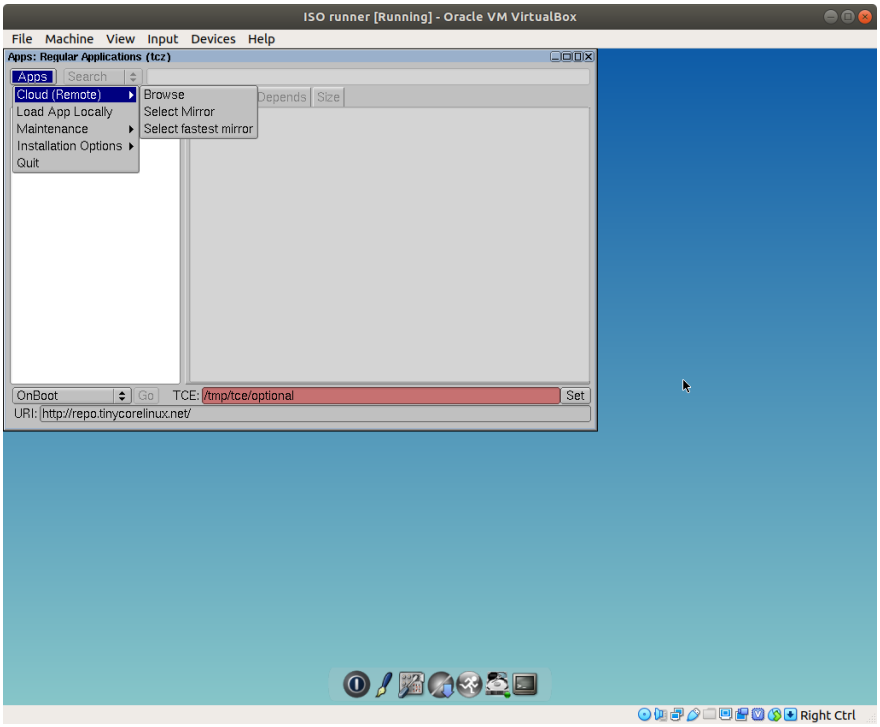
When the laboratory machines are powered up, the "IoT" image should be selected from the boot list. In this case, the Tiny Core Linux starts and brings up VirtualBox with the IoT measurement image. Unfortunately, this virtual machine is not ready to use the serial port where the sensor device will be programmed. In order to have access to the sensor, the following steps should be taken:

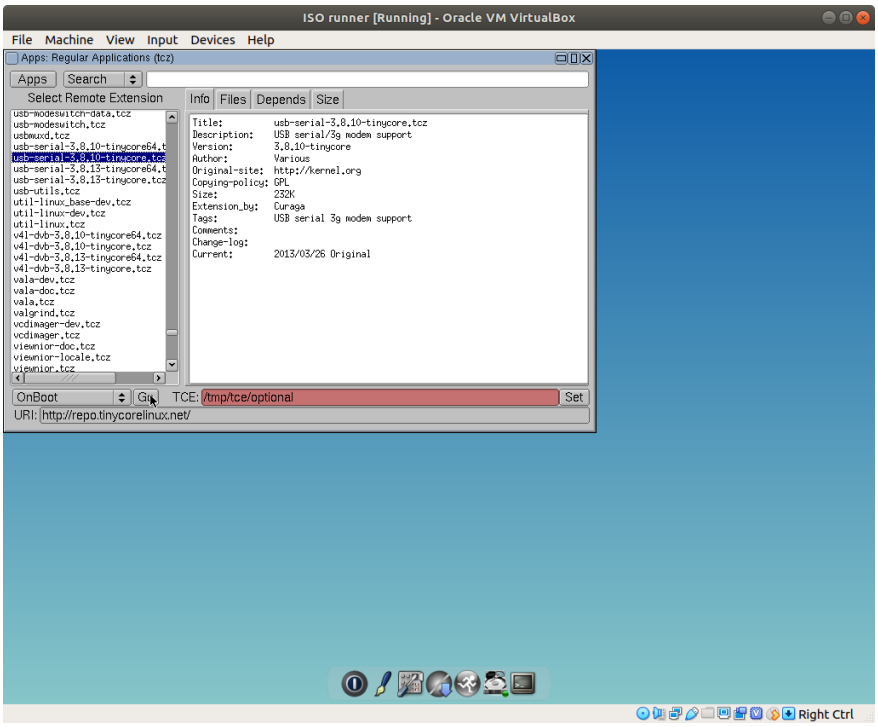
1. The serial driver for Linux should be installed.
    a. Clicking on the background, SystemTools/Apps should be selected. Apps starts and ask about to pick the fastest mirror. Click NO! The required file is already on the disk.





b. Select Apps / Cloud (Remote) / Browse. The list on the left side will be populated with apps possible to install. The list is in alphabetical order. Select "*usb-serial-3.8.13-tinycore64.tcz*" item. There are more from this, select the latest version. Then click on "Go" button..



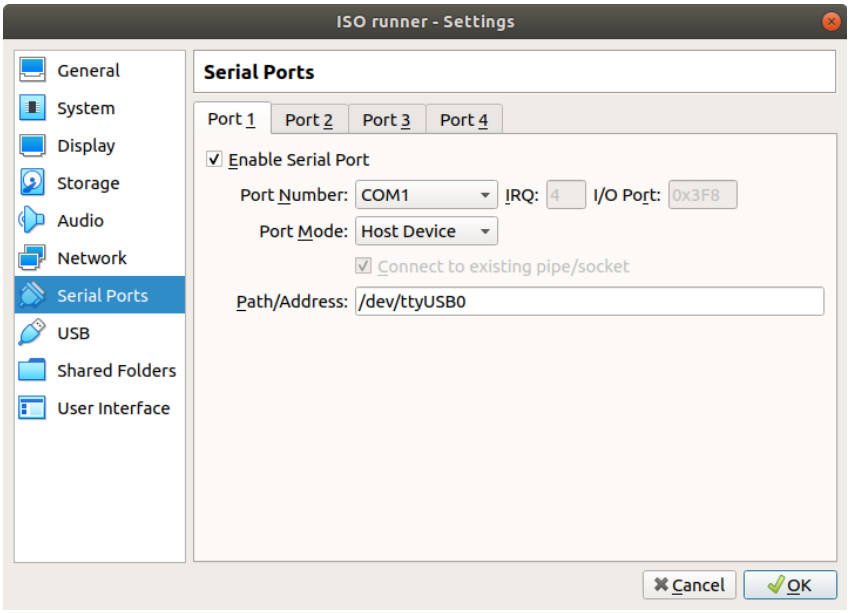


c. The file is downloaded from the Internet and installed. There should be no error message.

2. VirtualBox application should be restarted.
    a. The existing application should be closed. Clicking on the background there is an option to start a terminal program. Applications / Terminal
    b. Run the terminal program by clicking on it,

c. Type “*virtualbox*” into the terminal window and press Enter

3. The VirtualBox Linux image should be set up to attach the serial port.
   a. Go to the settings of the VirtualBox image, by clicking on the gear icon.
   b. Select the Serial Ports menu item



   c. Enable Serial Port by clicking on the box. Adjust port mode to “Host Device” and set the input field *Path* to “/dev/ttyUSB0”
4. Start the Linux image in VirtualBox

Since the virtual machine on this hardware host is somewhat limited, it is not possible to use the serial port with the highest speed that Arduino IDE will offer. Once the Arduino IDE is running, set the upload speed to 115200 baud at Tools / Upload Speed and also set the port to /dev/ttyS0 at Tools / Port.

3. Documentation

The student should create a measurement report, describing all the major steps of the measurement. The report should be illustrated with screenshots and/or pictures. The report should contain an appendix with the code of the sensor program. The student can use Google Docs in order to create the report. LibreOffice is also installed on the measurement machine. The report should be submitted to the classroom at the end of the measurement.

4. Building the sensor

SAFETY FIRST: Do NOT use the 5V pin of the Wemos device. It can kill the device itself. Avoid short circuits! NEVER use the led without current limiting resistor! The maximum current the led can tolerate is 20mA.

The students have to build the temperature sensor with a button that sends its measurement data to the IoT Platform and receives actions regarding the led lighting. First, the sensor should just display the measurement value and button press on the host machine's serial monitor.

All the necessary information to the Wemos device is available here:
https://escapequotes.net/esp8266-wemos-d1-mini-pins-and-diagram/

The thermistor is an electrical part that can change its resistance according to its temperature. The change in the resistance is not linear with the temperature change! There are some approximations how to calculate the temperature from the resistance value. The student should make the calculation in her/his code. The simplest approach is probably the Steinhart–Hart equation. The thermistor type is MF52 103 3950. Look up the corresponding values in the datasheet: http://www.sinochip.net/eng/mf52.pdf

The Wemos device has an A0 pin. This is the pin for analog measurements. Sample code for analog measurements can be found in the Arduino IDE: File -> Examples -> 01.Basics -> AnalogReadSerial. The code also serves as an example of how to display text on the host computer.
The resistance value cannot be measured directly as the Wemos can measure voltage level only. The student has to create a voltage divider in order to get the resistance value.

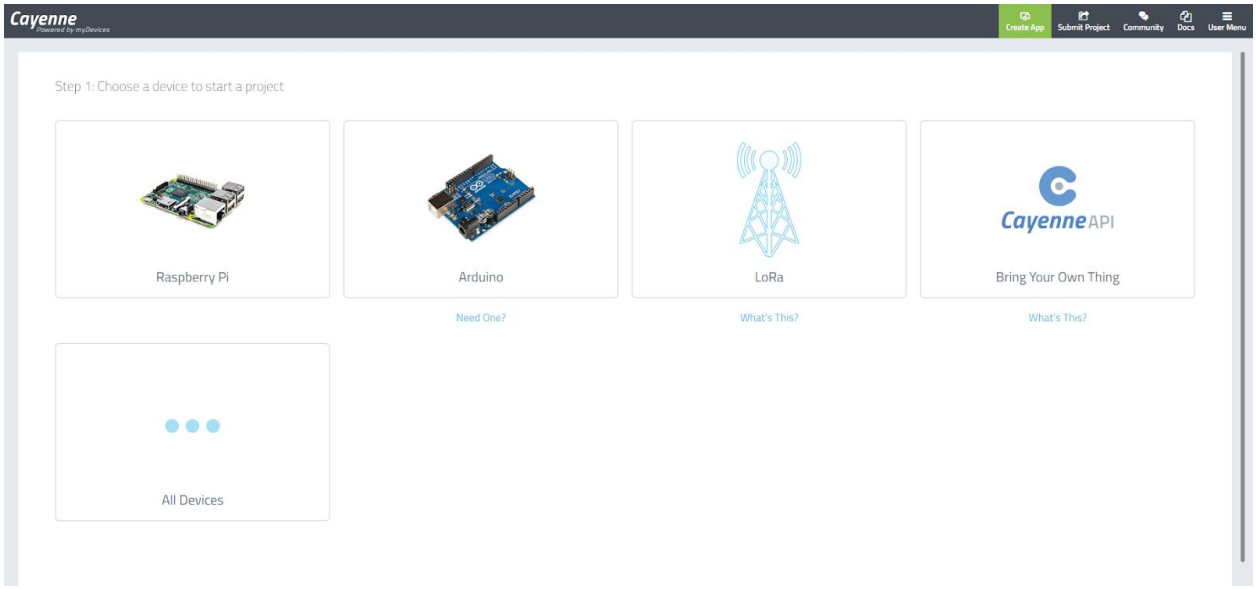
The button press is a digital input. There is no button provided, but it is really easy to create it from a single wire. Sample code can be found at: File -> Examples -> 01.Basics -> DigitalReadSerial. Some pins on the device are already pulled up or down, choose carefully!

The student has to prepare the sensor for the led control as well. The Wemos device has already a led, but an additional led is also provided. Sample code for the led control can be found at: File -> Examples -> 01.Basics -> Blink.

Create pictures with your mobile phone about the device, and include them to the report!

5. Virtual device setup

The next step of the measurement is to set up the virtual device. Already registered users should go to the https://cayenne.mydevices.com page to log in.



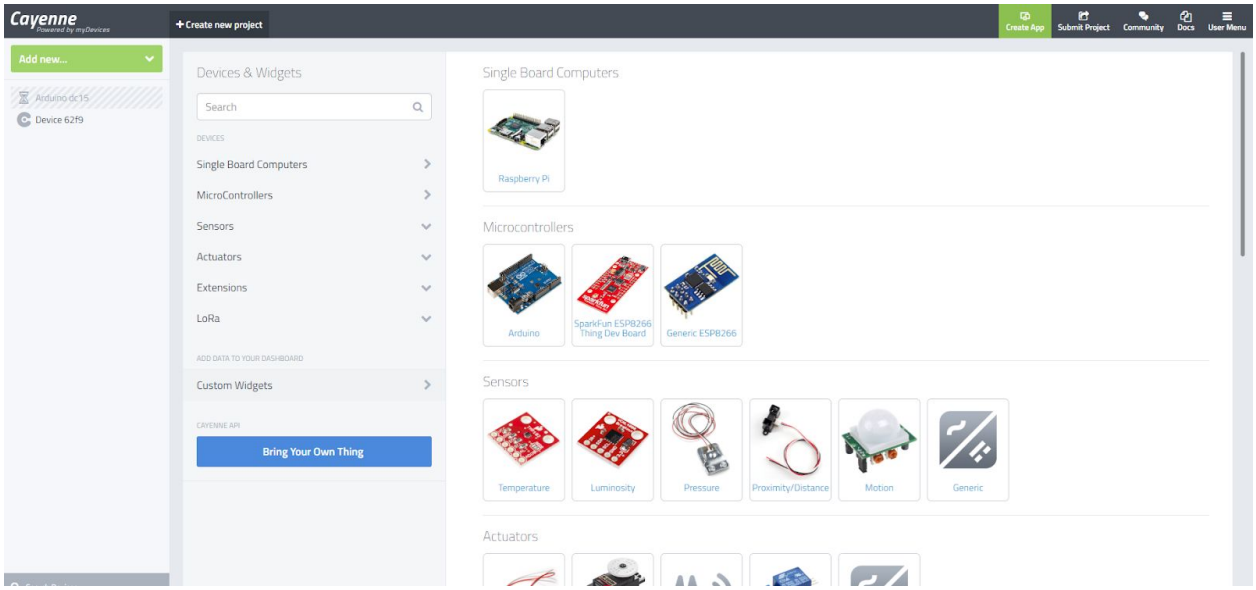
The initial screen of the webpage should be similar to the picture above. Cayenne support Arduino microcontrollers, however despite the Arduino IDE the Wemos device is not the member of the Arduino family. Choose “Bring Your Own Thing”! The Wemos will use the Cayenne MQTT API.

First of all test Cayenne with an emulated device using manually constructed MQTT messages. Follow the tutorial of Cayenne about MQTT messages:
https://mydevices.com/link/bring-your-own-thing-api-manually-sending-verifying-data

The Cayenne screen will not proceed over “Waiting for board to connect…”, unless an MQTT client is connected with a given username and password. Use MQTT.fx for the connection! Complete the tutorial by sending and receiving messages through MQTT. (The end of the relevant tutorial section is at the sentence: “Congrats! You can now handle actuator messages sent by the Cayenne Cloud!”)

6. Connecting physical and virtual device

After completing the MQTT tutorial, now students should connect their own physical device to the Cayenne IoT platform. The device should be added as a new device. There is a green button close to the upper left corner to add a new device.



Creating the new virtual device will start with the “Bring Your Own Thing” button, the same as before. The webpage will wait for the connection with a new id and credentials.

At this point, the student should start Arduino IDE and integrate her/his own device into the Cayenne IoT platform. In order to connect to the platform, the CayenneMQTT library should be used. This is already installed to the Arduino IDE. Look at the example at File -> Examples -> CayenneMQTT -> Connections -> ESP8266. Replace the WiFi AP’s name and password with the one that is provided by the supervisor! Replace the client ID and user credentials according to the page in Cayenne! Read the example code, upload it and test it!

The previous code with the temperature reading and button should be integrated into the Cayenne example code. Integrate the led control code also! Create widgets for the temperature sensor, button, and the led control!

Make a screenshot on the dashboard displaying the widgets and put it into the report! Display the graph of the temperature values and put that into the report as well!

7. Submitting the report

All the major steps and their results should be documented in a report. All students, who participated in the laboratory exercise, should create and submit her/his own report. The submission happens on the Google Classroom platform.