FLUENCE
A Siemens and AES Company

Controller Installing and Testing Procedure

CONTROLLER & CONNECTED COMPONENTS

[23rd September 2021]

Document Control Number: 00TE-PRO-001-R2.71

Revision Number: 2.71

Expiration Date:

Author: Boris Kajganić

Approval: _____

Revision History

Date	Revision Number	Pages Affected
21st July 2020	1.0	Initial release
26th August 2020	1.1	EATON UPS Added DL-10 Added
31st August 2020	1.2	improved documentation
8th September 2020	1.21	
27th October 2020	1.3	OsensaView workaround, added 7.5 kW Chiller
2nd December 2020	1.4	Added Modbus-Script Testing
10th December 2020	1.41	Osensa sensor as optional
16th December 2020	1.5	Chapter “Booting from eMMC”
2nd December 2020	1.52	Correction in Modbus-Script and eMMC
20th January 2021	1.53	Add UPS part in Modbus Server
26th January 2021	1.54	Added Flow Meter
28th January 2021	2.0	Added OCTE
10th February 2021	2.1	Added Document Control Number and OCTE UPS/HVAC
28th April 2021	2.2	Cube UPS power cycling
2nd August 2021	2.5	Banana PI Erase EMMC Modberry 500 installation
20th August 2021	2.6	Installation scripts
27th August 2021	2.61	Script controllerfos.sh
29th August 2021	2.7	Self-Configuration
23rd September 2021	2.71	Added MAC for ModBerry

Table of Contents

1. Terms and definitions

Banana Pi R2	- Open-Source Wireless Router
Controllino MEGA	- Programmable logic controller
PLC	- Programmable logic controller
SBC	- Single Board Computer
UART	- Universal Asynchronous Receiver/Transmitter
HVAC	- Heating, ventilation, and air conditioning device
Chiller	- liquid base HVAC
TX-812-PWR+	- Osensa Fiber Optic Temperature device
DS18B20	- Temperature sensor
DL-10	- Temperature and humidity sensor
UPS	- Uninterruptible power supply
Nut	- Network UPS Tools
AX-GS-CM-V-65	- Carbon Monoxide Sensor
OCTE	- Outdoor Core Talco Rack
CTE	- Core Talco Rack
CUBE	- Fluence Battery Storage

Controllers type in Fluence environment:

Short Name	Name
ATR FMC (ATRMC)	Array Telco Rack Controller (reserved name for the future)
CTR FMC (CTRMC)	Core Telco Rack Controller
Array FMC (AMC)	Array Controller
DAS FMC (DMC)	DAS Controller
Core FMC (CMC)	Core Controller
Node FMC (NMC)	Node Controller
Cube FMC (CuMC)	Cube Controller

FMC	Fluence Multipurpose Controller

2. Start

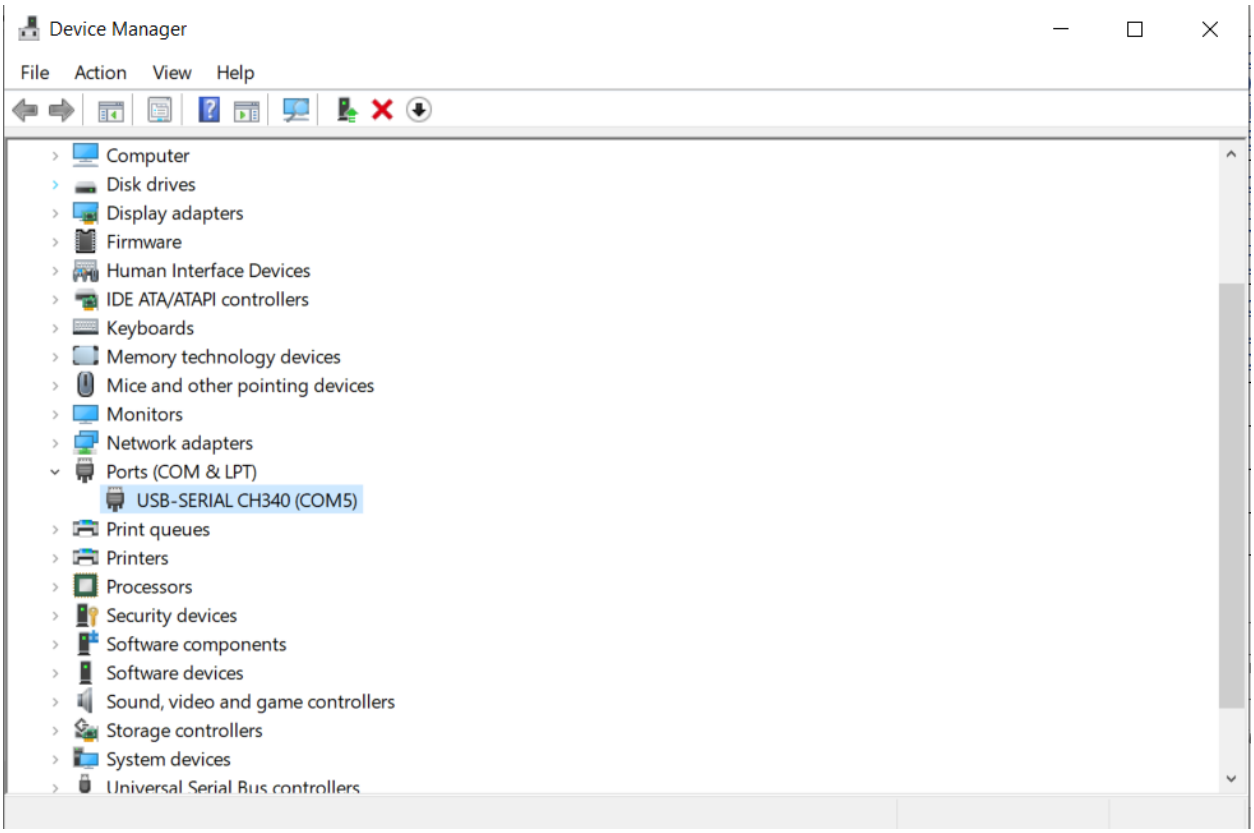
The Installation and Test Procedure cover both Cube and Core Talco Rack (CTE).

It is strongly recommended to read complete test procedure first. At first, the reader will get overview what should be tested and how and the experience user will find what is different from previous version.

3. Required Hardware

Hardware		
Laptop (with Ethernet port)		
Ethernet cable 2 to 3 m		
Ethernet crossover cable 2 to 3 m	Optional	
USB to ethernet adapter	Optional	https://www.reichelt.de/logilink-usb-3-0-zu-gigabit-ethernet-adapter-logilink-ua0184a-p187303.html?&trstct=pol_7&nbc=1
SD Card 32G	Optional	https://www.reichelt.de/microsdhc-card-32gb-samsung-evo-plus-sams-mb-mc32ga-p207608.html?&trstct=pol_0&nbc=1
USB RS485		https://www.reichelt.de/raspberry-pi-usb-rs485-interface-ch340c-rpi-usb-rs485-p242783.html?r=1 Windows driver: Link under “Datasheets” -> Windows-Treiber
Multimeter	Optional	Check voltages on terminal block

You will need appropriate driver for USB to RS485. Properly installed driver is located under Ports of the **Device Manager**.



4. Required Software

The following software is required for testing of Controllers inside Cube or Core Talco Rack.

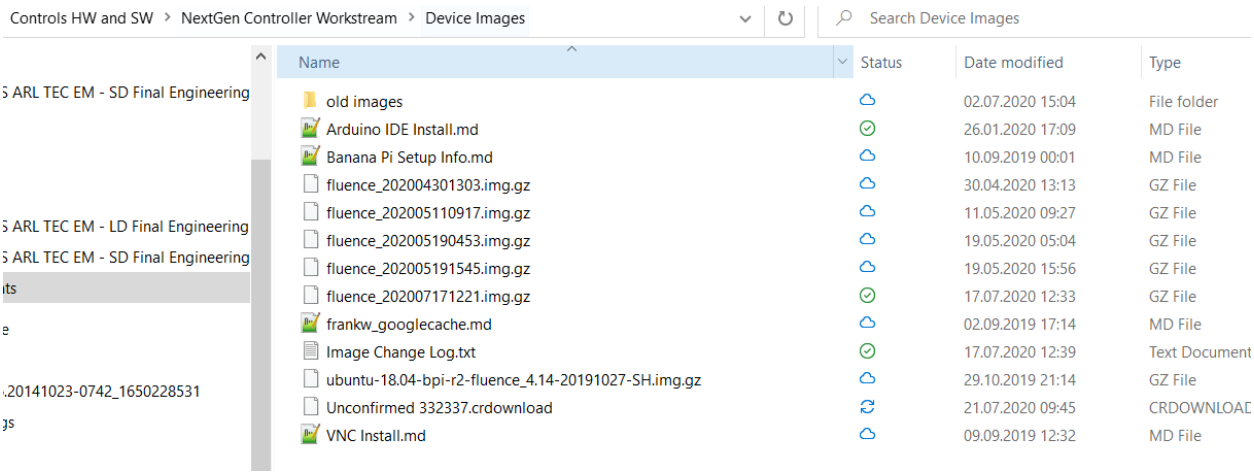
Software		
Putty	PuTTY is an SSH and telnet client	https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
QModMaster	QModMaster is a free Qt-based implementation of a MODBUS master application. A graphical user interface allows easy communication with MODBUS RTU and TCP slaves. QModMaster also includes a bus monitor for examining all traffic on the bus.	https://sourceforge.net/projects/qmodmaster/
BalenaEtcher	Suggested program to flash SD card	https://www.balena.io/etcher/?ref=etcher_footer

5. Flash SD Card

Copy the last Fluence image from Fluence SharePoint:

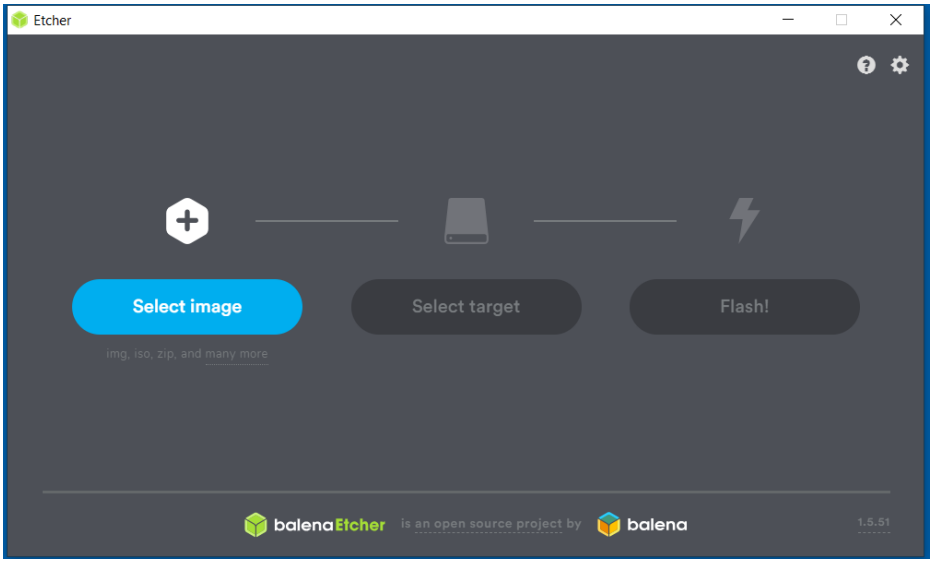
https://fluenceenergy.sharepoint.com/:f:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Device%20Images?csf=1&web=1&e=lMdQuT

All images start with “fluence_%timestamp%.img.gz”



Use the youngest of the fluence images.

Flash the image to SD card with help of BalenaEtcher program. The program is available for Windows, Linux and Mac OS.

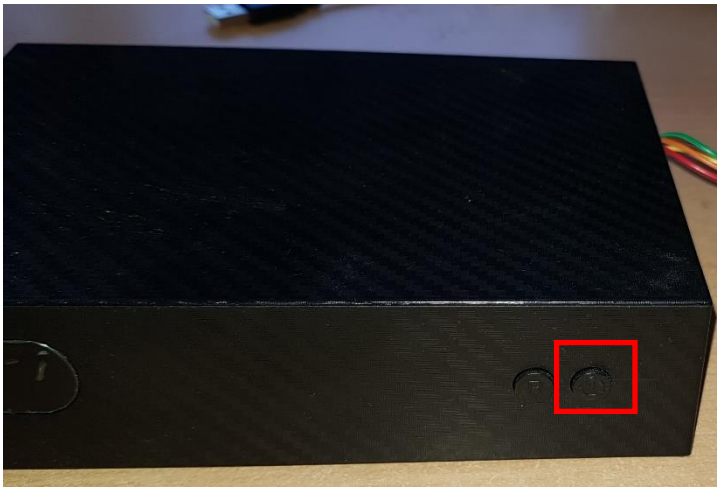


6. Start Banana Pi

Banana Pi must be power off before the SD card is inserted into the slot of the Banana Pi. The best is to pull out power cable (12 V) of Banana Pi.

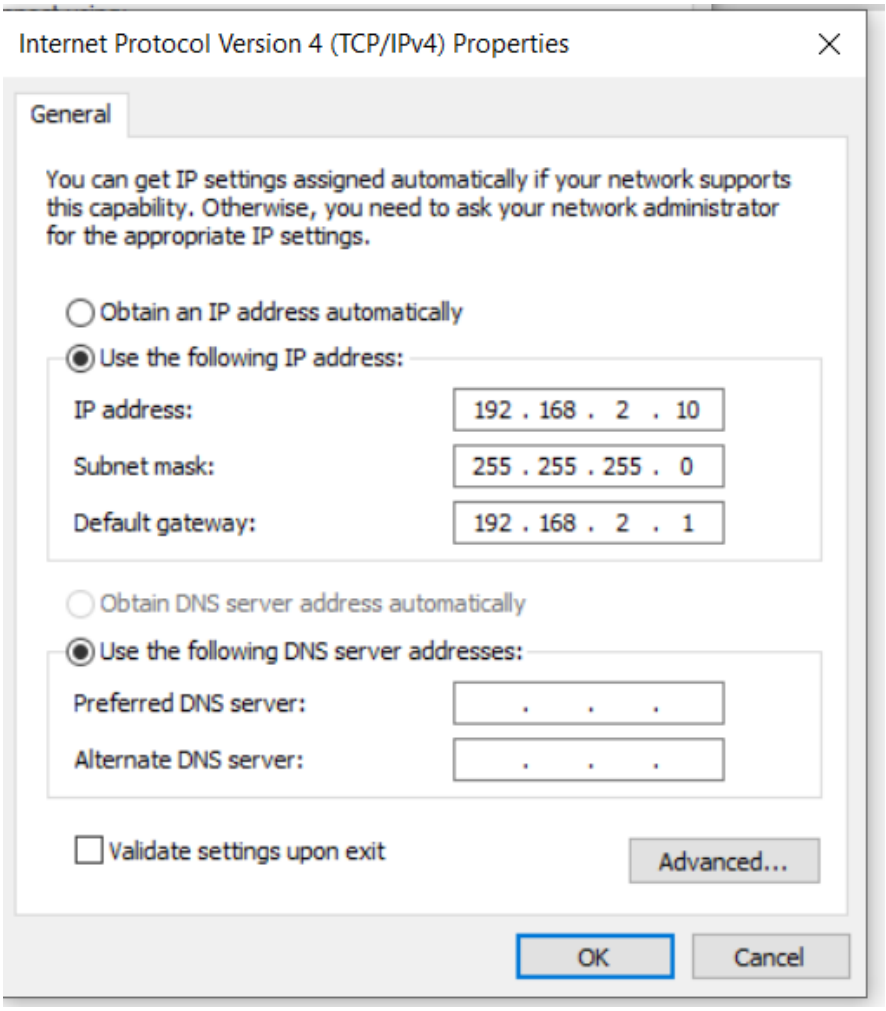
Insert SD Cart.

Connect back power cable. If USB OTG port (left from power) is not supplied with 5 V. you will need to press power button for 10 second to start the Banana Pi.



7. Ethernet

Add IP address “192.168.2.10” to ethernet adapter of your laptop.



On Banana Pi, first ethernet port (from left) is WAN port. It connects previous cube. The last port LAN 4 connects next cube. The controllers inside OCTE/CTE are connected to the switch directly and only WAN port is used.
Expected network will be **172.XXX.XXX.XXX** or equivalent network. Current Banana Pi will be determined by DHCP server of during the Self Configuration process on the site network.

Internal 3 ports of Banana PI, are for internal network of the cube. IP addresses are **192.168.2.XX**.

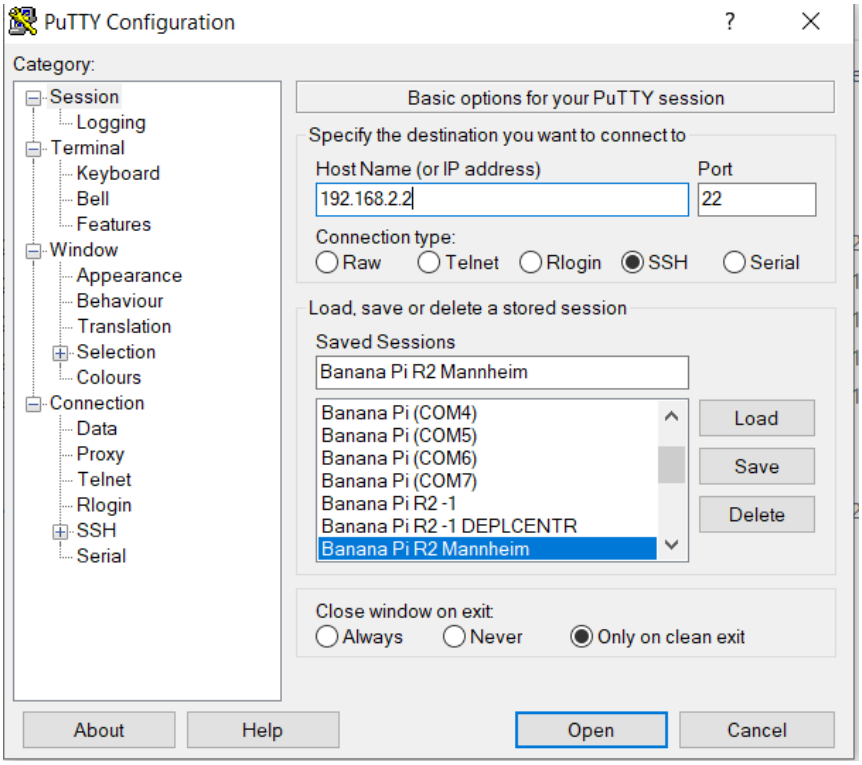
Banana Pi has IP address of 192.168.2.2. Controllino is connected to one of tree internal ports and has IP address **192.168.2.3**. Connect you testing laptop to one of the internal free ports:

Test equipment:

- Laptop with Ethernet port or converter (USB to Ethernet)

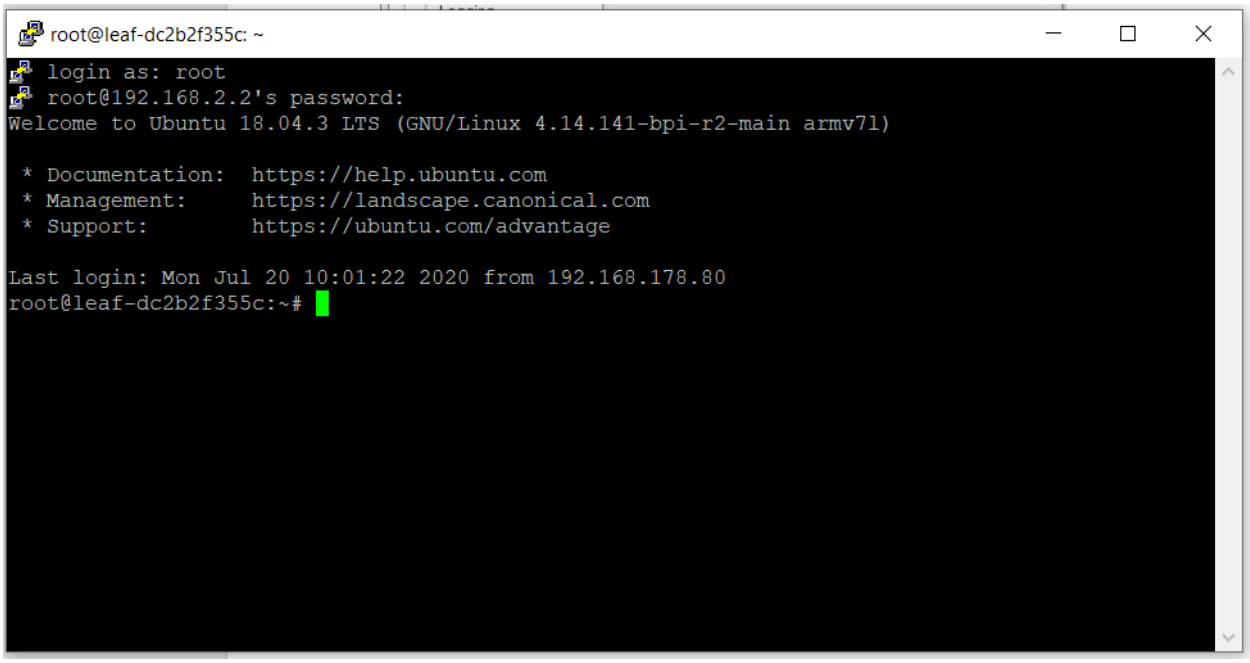
7.1.1 Testing with Putty

Test connection with Putty:



User: **root**

Password: **root**



If you use access to Banana Pi from external network, please execute **fluence_iptables.sh** after first login. The script will setup correct Network address translation (NAT) from external to internal cube ethernet network.

8. Modbus RTU Devices

Current Fluence implementation of the cube has following Modbus RTU devices:

- Osensa Fiber Optic Temperature Device (optional)
- Chiller 2.5 kW (2x) or Chiller 7kW (1x) in long duration cube and (2x) in short duration
- HVAC (1x or 2x)
- DL-10 Temperature and humidity sensor

Current Fluence implementation of the CTR has following Modbus RTU devices:

- DL-10 Temperature and humidity sensor

All devices are connected over Modbus RTU protocol based on RS485. For each device we need to ensure same baud rate 19200 8N1 and unique Modbus slave ID. ID **42** is reserved of Controllino MEGA and cannot be used.

Default setting of listed devices are unknown. You can assume the more than one device will have the same parameters. **Ensure to connect only one device with your Laptop, the other devices must be disconnected from Modbus RTU bus or the devices must be switch off.**

After all devices are correctly setup, you can connect them back to same RT485 bus.

Test equipment:

- Laptop with USB to RS485 converter

8.1 Osensa Fiber Optic Temperature Device (optional)

Test	Cube	CTR
Applicable	**X**	

All Osensa documents are in SherePoint:

https://fluenceenergy.sharepoint.com/:f:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Vendor%20Info/Sensors/Osensa?csf=1&web=1&e=QDfdqG

Install Osensa software.

OSENSAView_64bit_V4.30_Setup.exe

The default administrator password is “0” (number zero).

If you are not able to use 64 bits version of OsensaView you can install 32 bits version (OSENSAView_64bit_V4.30_Setup.exe).

If you have trouble tom start OSENSAView it is consequence of incorrect java settings.

Osensa is Installed in “C:\Program Files\OsensaView” for 64 bits version or in “C:\Program Files (x86)\OsensaView” for 32 bits version.

In the installation folder there is the file OsensaViewExecute.bat

```
java -jar OsensaView.jar
```

The file must be adapted to use correct Java version

```
"C:\Program Files (x86)\Java\jre1.8.0_212\bin\java" -jar OsensaView.jar
```

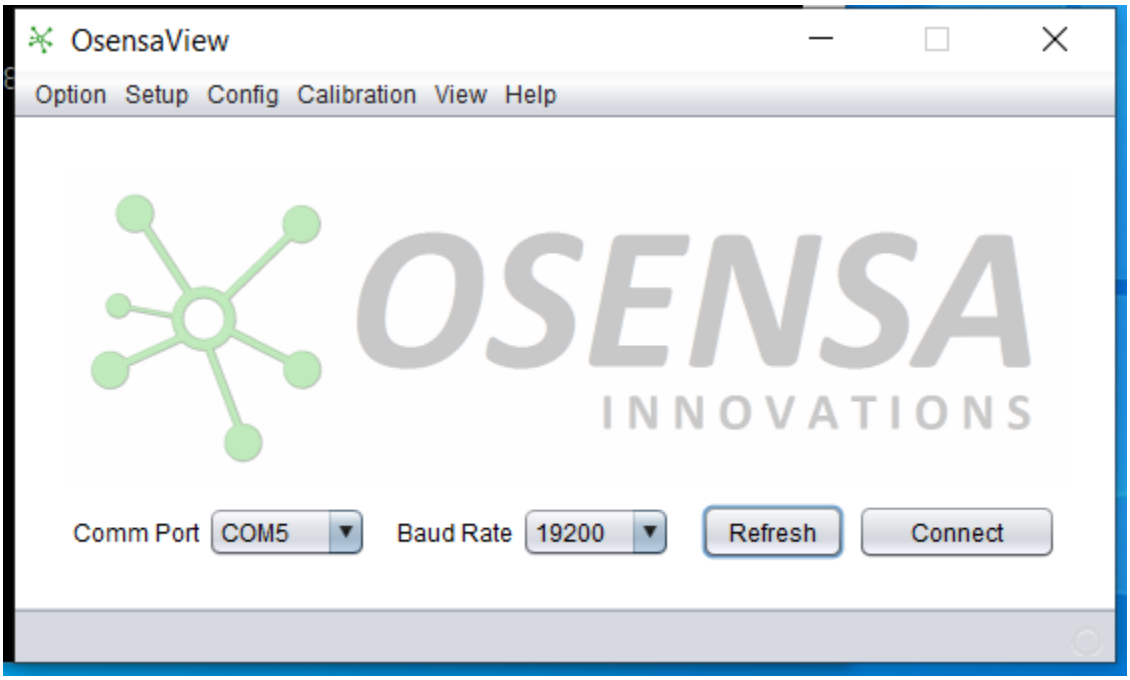
Or

```
"C:\Program Files\Java\jre1.8.0_251\bin\java" -jar OsensaView.jar
```

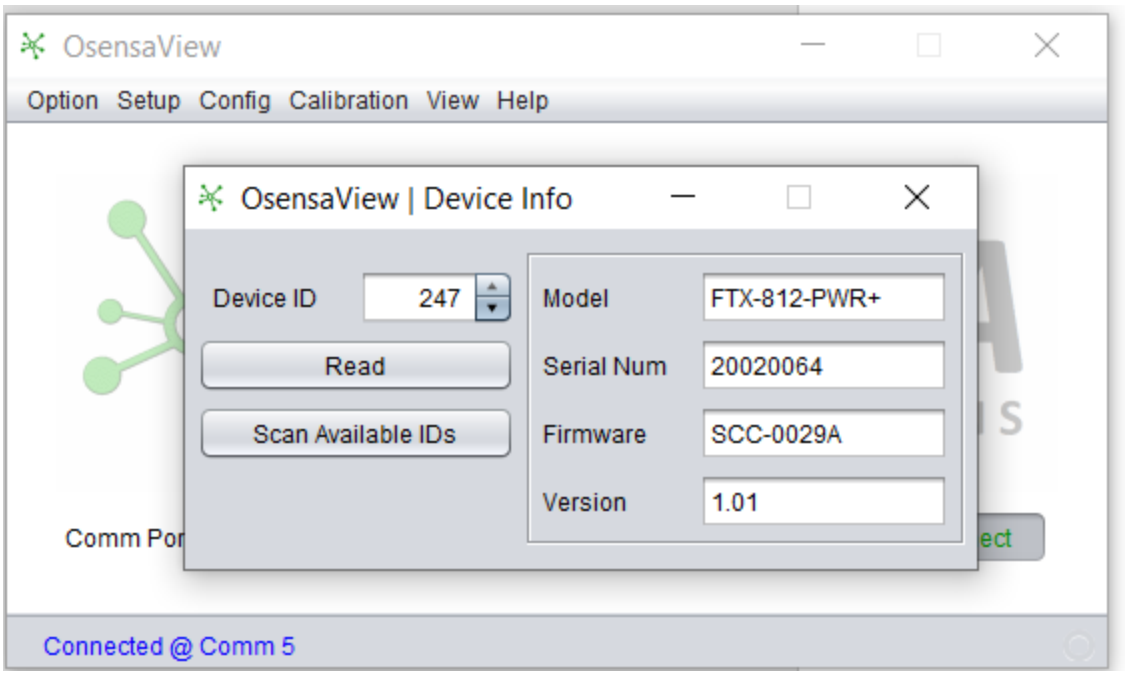
Check correct version of java in C:\Program Files\Java” for 64 bits version or in “C:\Program Files (x86)\Java” for 32 bits version.

8.1.1 Setup the device

The default device parameters are 9600 8N1 and Modbus slave address 247. Our test example was somehow buggy, I could change 9600 8N1 to 19200 8N1, but not slave ID to 10. After reset, the ID was again 247. Fluence ordered devices to be factory selected to 19200 8N1, Modbus ID = 10.

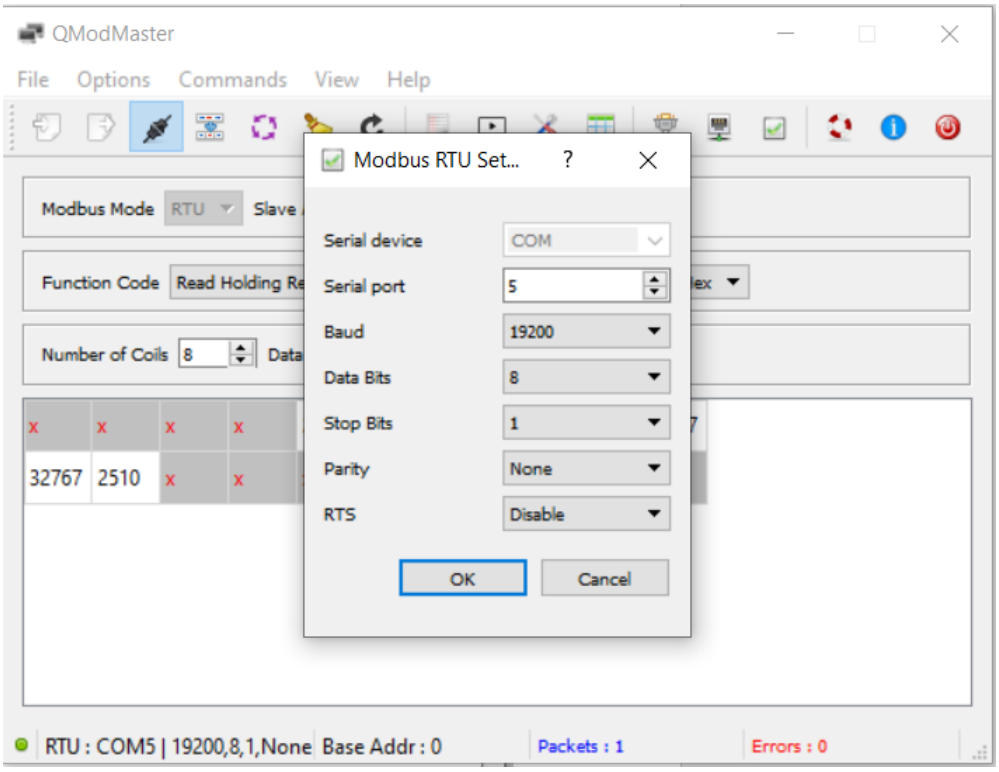


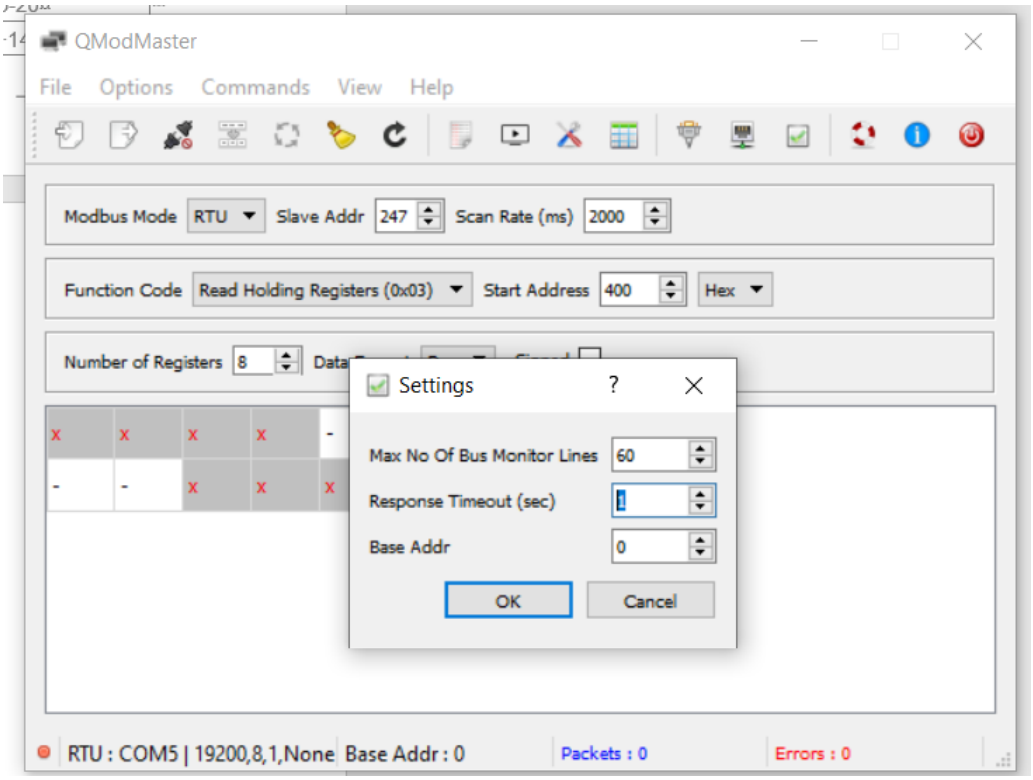
Test response for the device.



8.1.2 Testing with QModMaster

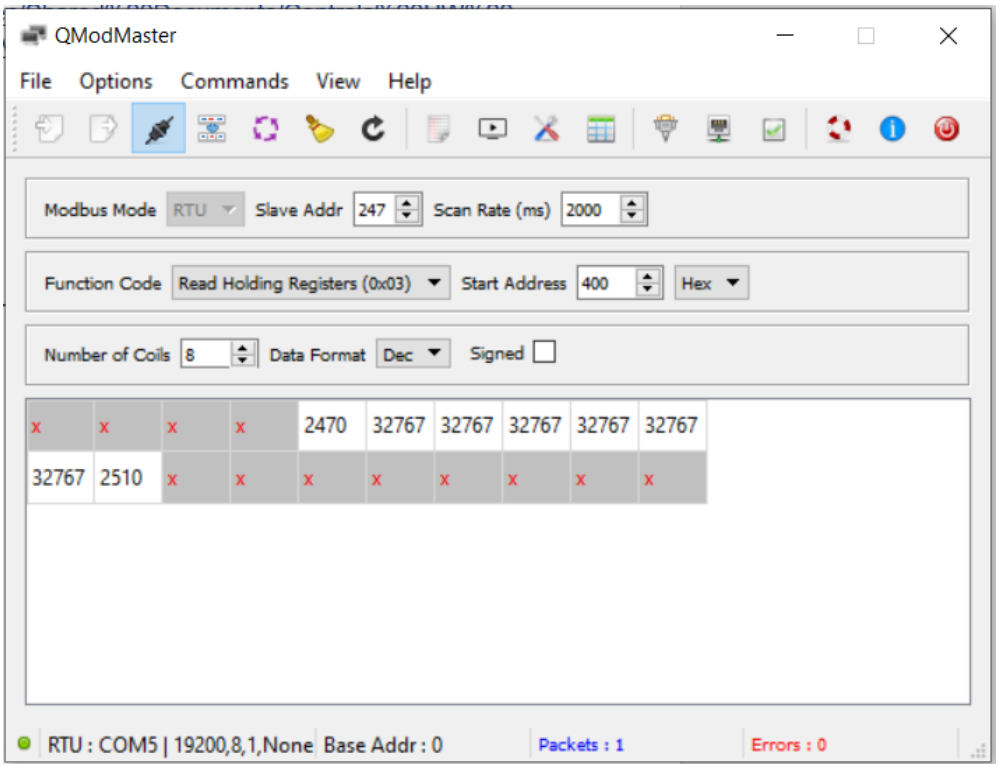
Parameters for testing:





Ensure Response Timeout is **min 1** second. The scan rate will be double value of timeout value. Communication port, communication parameter and timeout are common for all testing.

Osensa has 8 sensors starting from address 0x400 (1024 dec). In example only the first and the last sensors were connected. The measured temperature is 100 bigger, divide with 100 for temperature in C.



8.2 DL-10 Temperature and humidity sensor

Test	Cube	CTR
Applicable	**X**	**X**

All DL-10 documents are in SherePoint:

https://fluenceenergy.sharepoint.com/:f:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Vendor%20Info/Sensors/ICP-DAS/DL-10?csf=1&web=1&e=7AZfM7

Unpack on local disk the file “dcon_utility_pro_pc_2019_1217.zip”

Start DL-10 configuration software.

DCON_Utility_Pro.exe

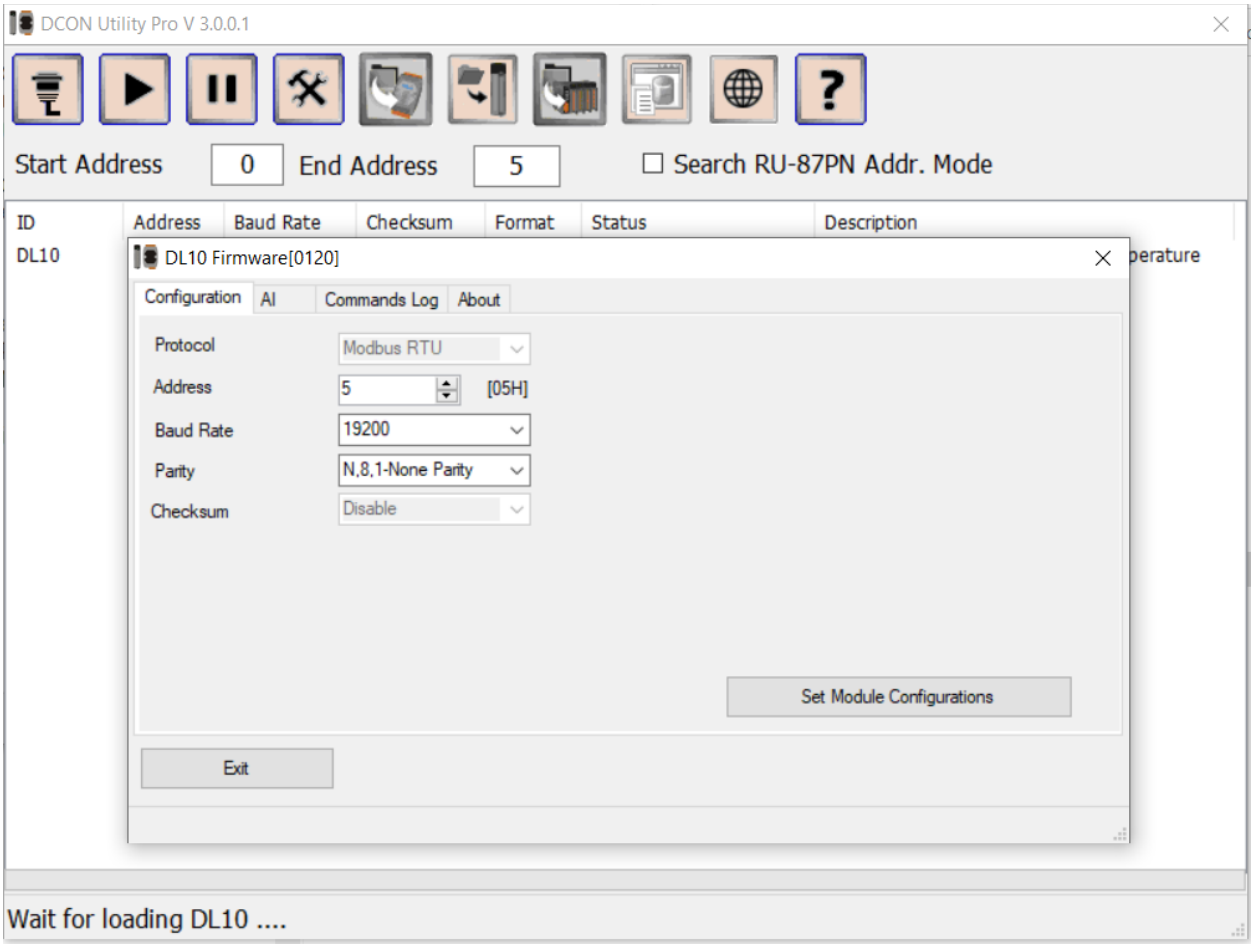
8.2.1 Setup the device

The default device parameters are 115200 8N1 and Modbus slave address 1. Change the setting to 19200 8N1 and Modbus slave address 5.

Follow the instruction (3.3 Search Module) from the manual:

https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Vendor%20Info/Sensors/ICP-DAS/DL-10/dl-10_user_manual_v13_en.pdf?csf=1&web=1&e=J4tRUm

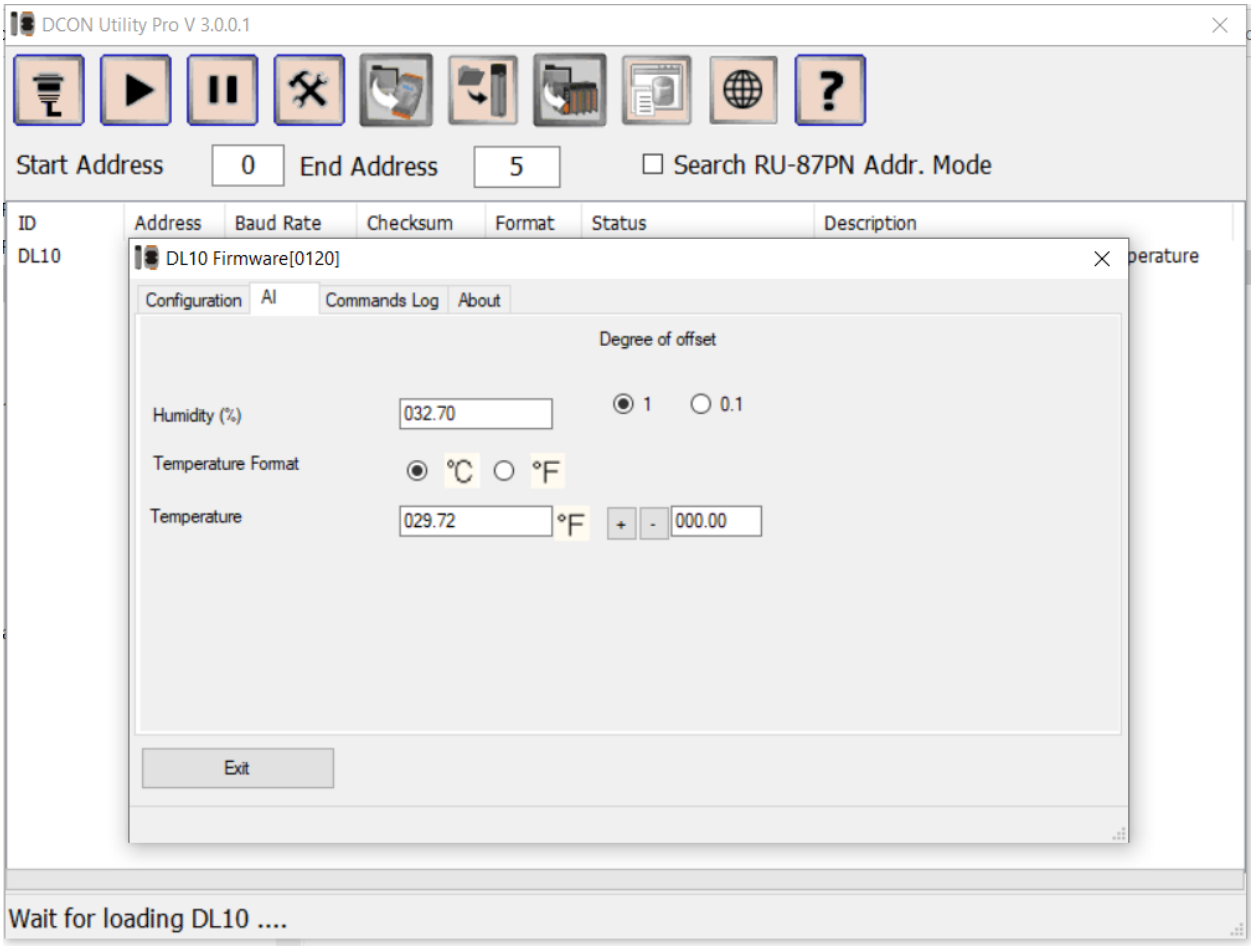
Change device setting to new values.



Important: Power OFF the sensor. Wait for 30 seconds. Switch on the sensor.

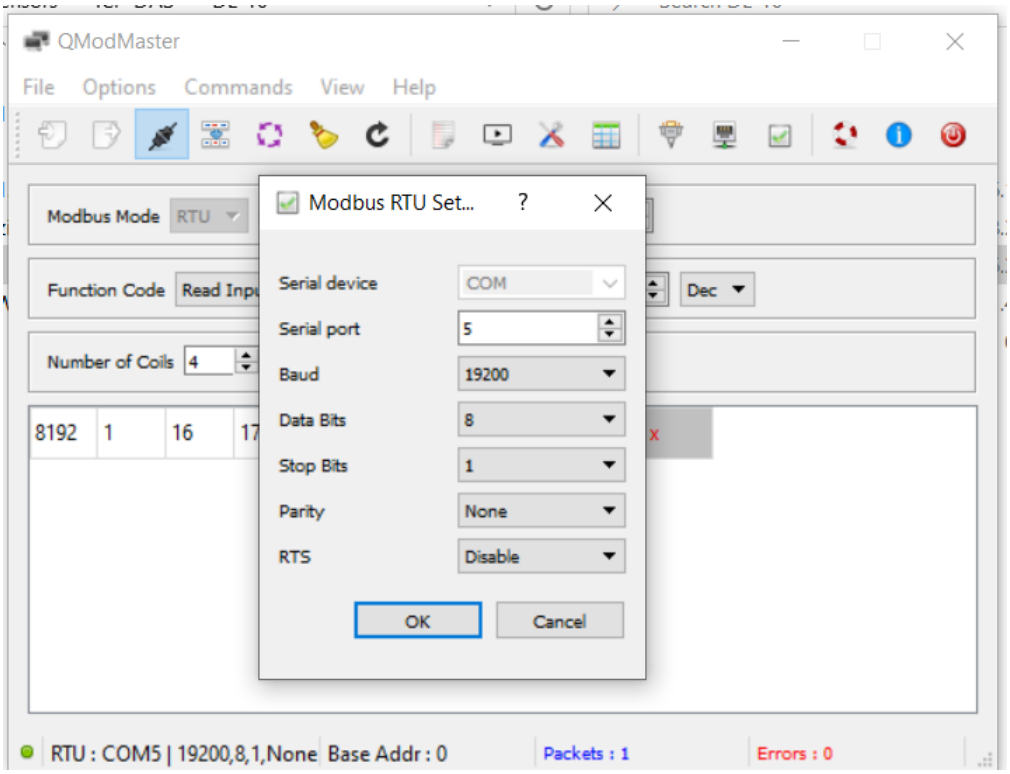
I had the problem; the output capacitor of my power supply is really big and even I switch off the power supply the sensor was powered for next 20 seconds. Without power cycle the new value will by not taken in account.

Received values:



8.2.2 Testing with QModMaster

Parameters for testing:

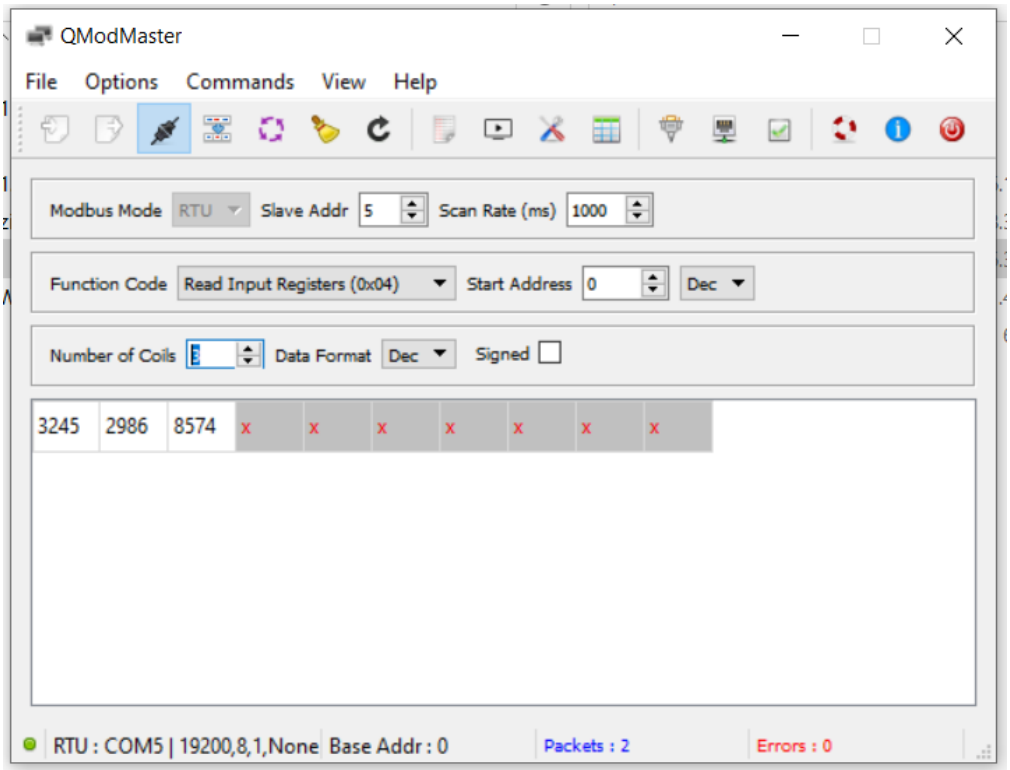


Set timeout to min 1 second.

DL-10 has 2 sensors starting from address 0x000. The measured temperature is 100 bigger, divide with 100 for temperature in C or F. Humidity is 100 bigger integer value.

Input Register (3xxxx)

Register		Points	Description	Data Format	Attribute
DEC	HEX				
30001	000	1	Read the humidity value (unit: 0.0.1 %)	0 ~ 10000	R
30002	001	1	Read the temperature value in degrees Celsius (unit: 0.01℃)	-32767 ~ 32768	R
30003	002	1	Read the temperature value in degrees Fahrenheit (unit: 0.01℉)	-32767 ~ 32768	R



8.3 Envicool Chiller

Test	Cube	CTR
Applicable	**X**	

Evicool Chiller 2.5 kW or 7 kW are interesting devices. I am not able to find in the documentation where to change Modbus baud rate and Modbus Slave ID.

User Manual for 2.5 kW Chiller:

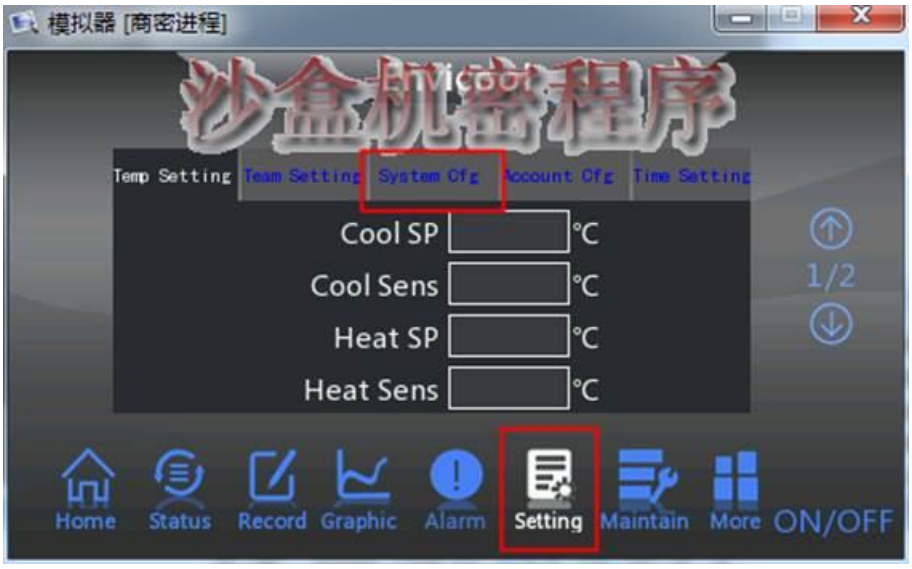
https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Cooling%20Concept/Chiller/User%20Manual%20EMW25HDNC1A%20(2.5kW%20Chiller)%2020200528.pdf?csf=1&web=1&e=YZjeld

8.3.1 Setup the chiller

The instruction and screenshots are from Envicool email.

Chiller display change route: (ignore the Chinese words, it's Envicool confidential software)

Default password: **0001**



Change the setting to 19200 8N1 and Modbus slave address 1 for the first chiller and 2 for the second device.

If is not possible to change the Modbus parameter on display, I assume default setting are 9600 8N1, Modbus slave address 1.

Evicool 2.5 kW Chiller Modbus Map:

https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Cooling%20Concept/Chiller/EMW25HDNC1A%20Modbus%20protocol%20-%2020200721.pdf?csf=1&web=1&e=qAkdex

Evicool 7.5 kW Chiller Modbus Map:

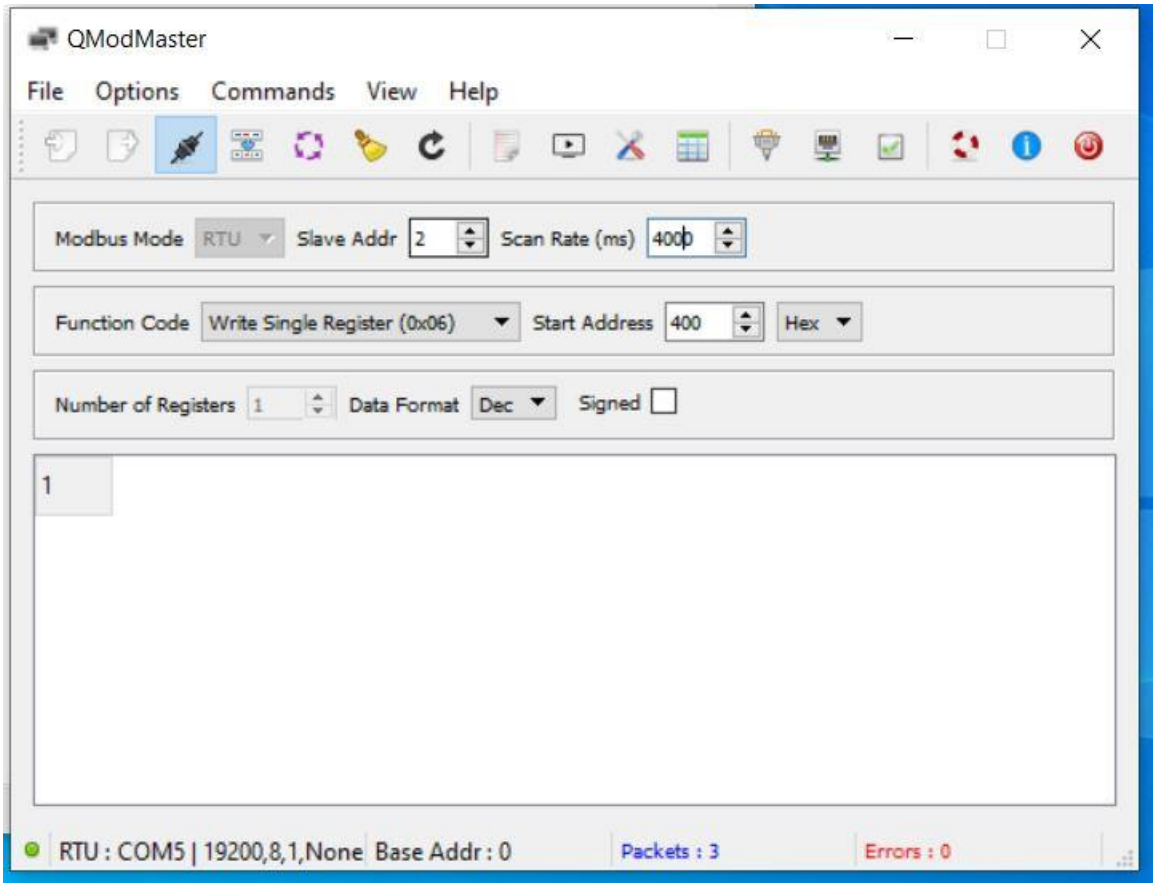
https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Cooling%20Concept/Chiller%207.5%20kW/EMW75HDNC1A%20Modbus%20protocol%20-%2020201011.pdf?csf=1&web=1&e=a4wkim

NO.	Item	unit	Register Address	Attribute	Ratio	Remark
On/off status						
1	System ON/OFF		0x0400	Read/Write	x1	0 OFF 1 ON

8.3.2 Testing with QModMaster

Try to read current status of the chiller. Try to write start or stop command.

Example of Send stop command to the chiller 2 of the SD cube (Modbus Slave id = 2)



8.4 Envicool HVAC

Test	Cube	CTR
Applicable	**X**	

Apply the same testing procedure as for Evicool Chiller.

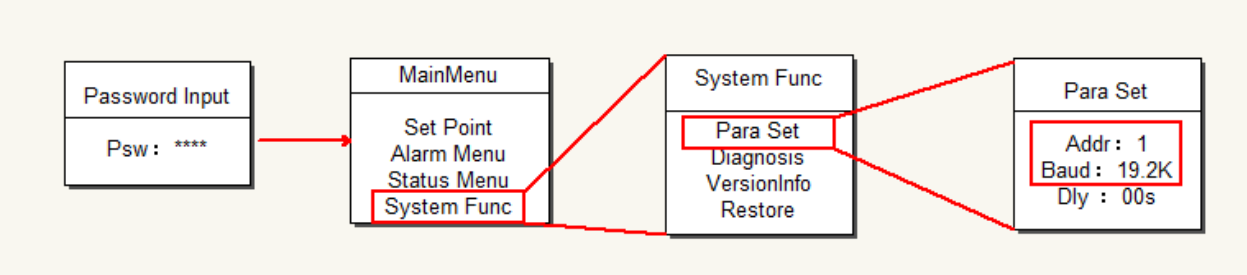
User Manual:

https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Cooling%20Concept/HVAC/EC06HDNC1U%20-%20User%20Manual%20-%2020150527.pdf?csf=1&web=1&e=8SCZRM

From one Evicool email I got following screen shot. The setting is required to be changed on the HVAC device.

Change the setting to 19200 8N1 and Modbus slave address 3 for the first HVAC and 4 for the second HVAC if the second HVAC is installed in the system.

EC06HDNC1U display change route, password: 0001



Evicool HVAC Modbus Map:

https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Cooling%20Concept/HVAC/EC06HDNC1U-Protocol%20(new).pdf?csf=1&web=1&e=Co9y1R

and

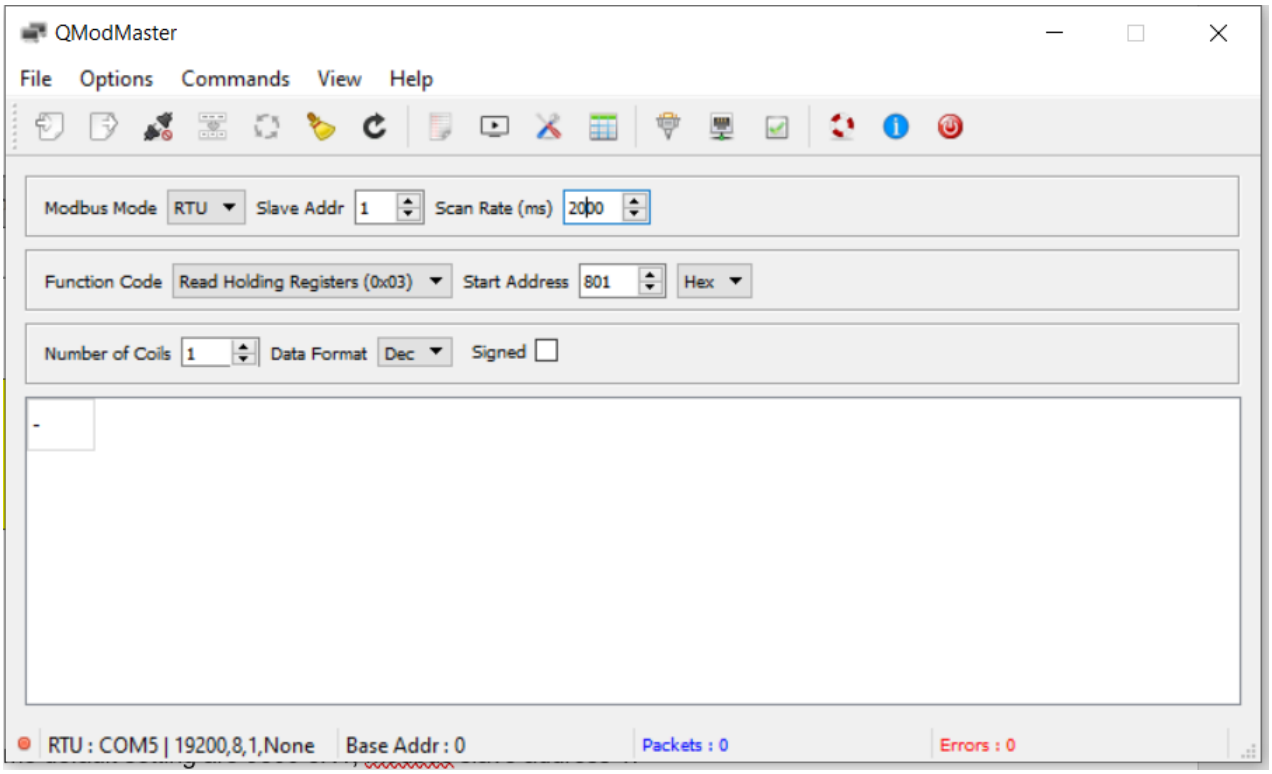
https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Cooling%20Concept/HVAC/EC03U3-modbus-20200721%20(EC06HDNC1U).pdf?csf=1&web=1&e=fAqQqY

From not complete Modbus mapping there is possibilities to start stop chiller

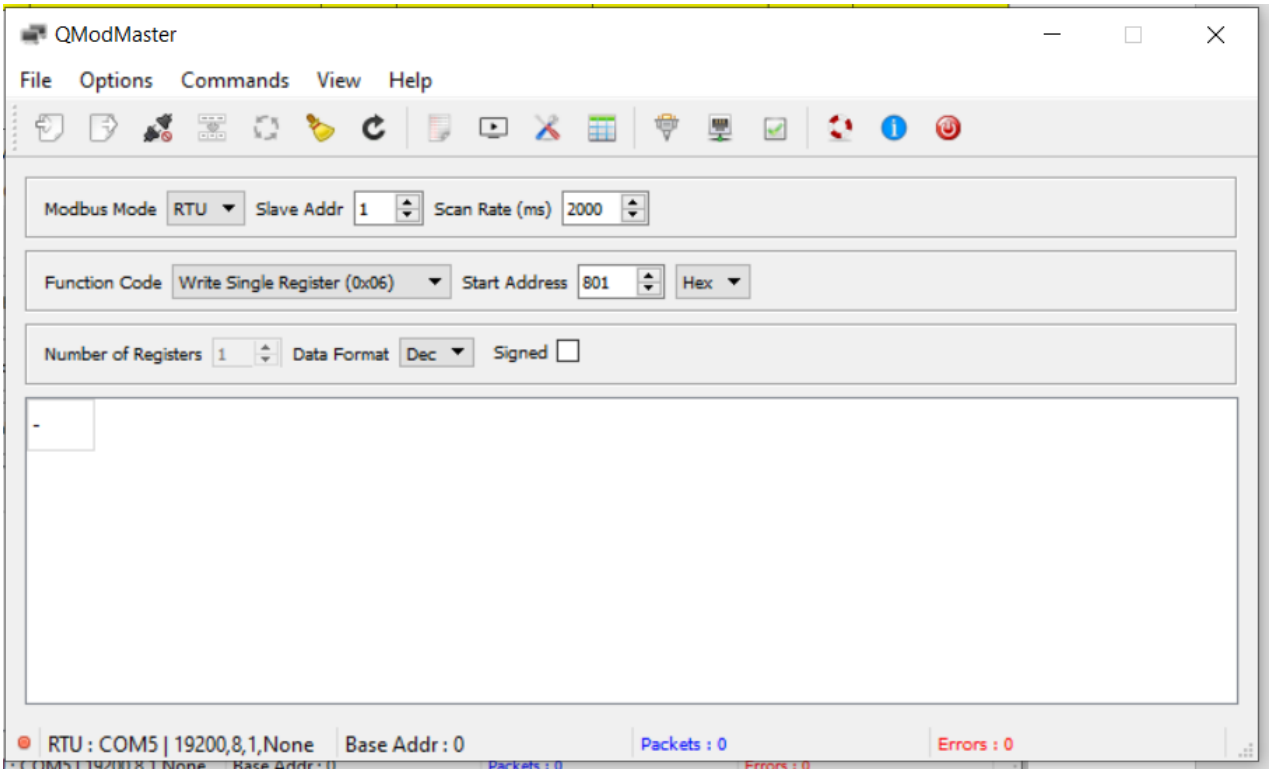
Remote Control Parameter						
1	Reserve		0x0800			
2	Remote control		0x0801	Read/Write		1: Open, 0: Close
3	Baudrate		0x0766	Read/Write		0: 9600, 1: 14400, 2: 19200

8.4.1 Testing with QModMaster

Try to read current status of the HVAC.



Try to write start or stop command.



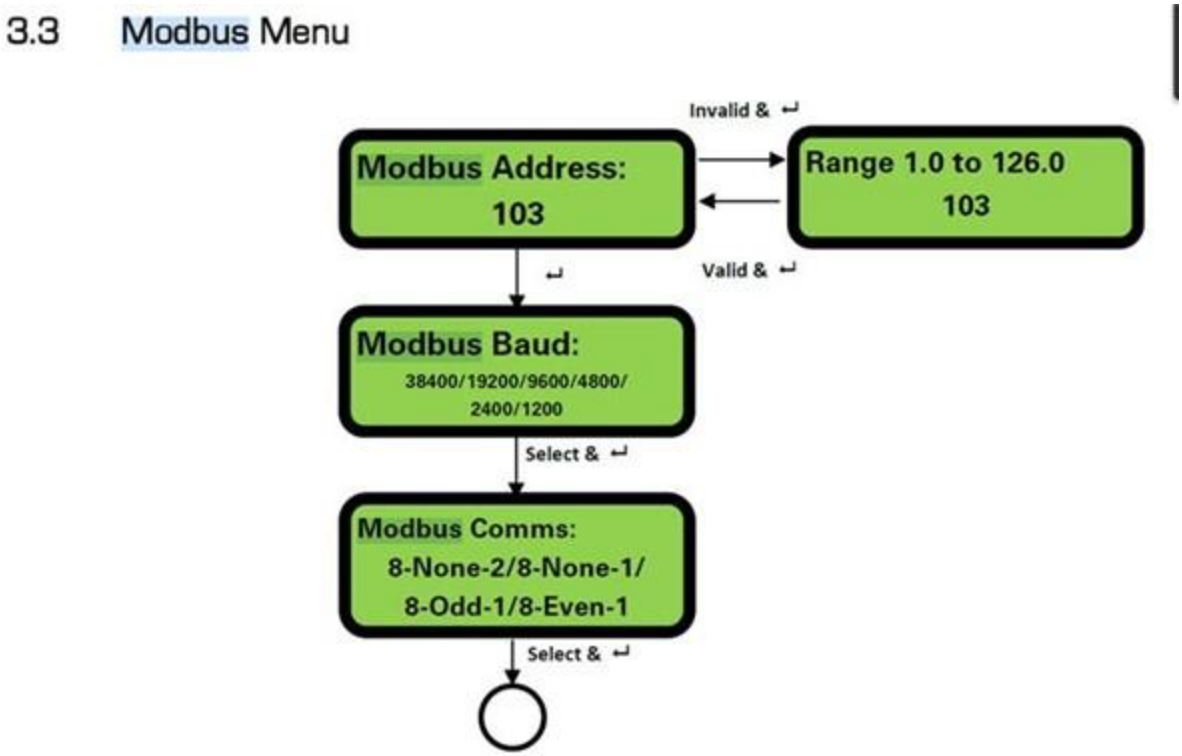
8.5 U1000MKII-FM: Clamp-on Ultrasonic Flow Meter (optional)

Test	Cube	CTR
Applicable	**X**	

User Manual:
https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Vendor%20Info/U1000%20Inexpensive%20Clamp%20on%20Flow%20Meter/U1000MKII-combined-Issue-3.3b.pdf?csf=1&web=1&e=q8sIxc

8.5.1 Setup the Flow Meter

Change the setting to 19200 8N1 and Modbus slave address 20 for the first Flow Meter and 4 for the second Flow Meter if the second Flow Meter is installed in the cube.



8.5.2 Testing with QModMaster

Flow Meter use the holding registers starting from address 0x000. For the Modbus Map use pages 32-34 from the Flow Meter manual.

Try to read current status of the Flow Meter.

9. Modbus TCP Devices

Current Fluence implementation of the CTR has the following Modbus TCP devices:

- nVent Spectracool Air Conditioner (HVAC)
- Quint4 UPS

9.1 nVent Spectracool Air Conditioner (HVAC)

Test	Cube	CTR
Applicable		**X**

All HVAC documents are in SharePoint:

https://fluenceenergy.sharepoint.com/:f:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Telco%20Rack/Components/OCTE%20HVAC?csf=1&web=1&e=sXBajC

For Modbus communication, the HVAC has to be used together with the Remote Access Control Panel. Documentation and Software downloads can be found here:

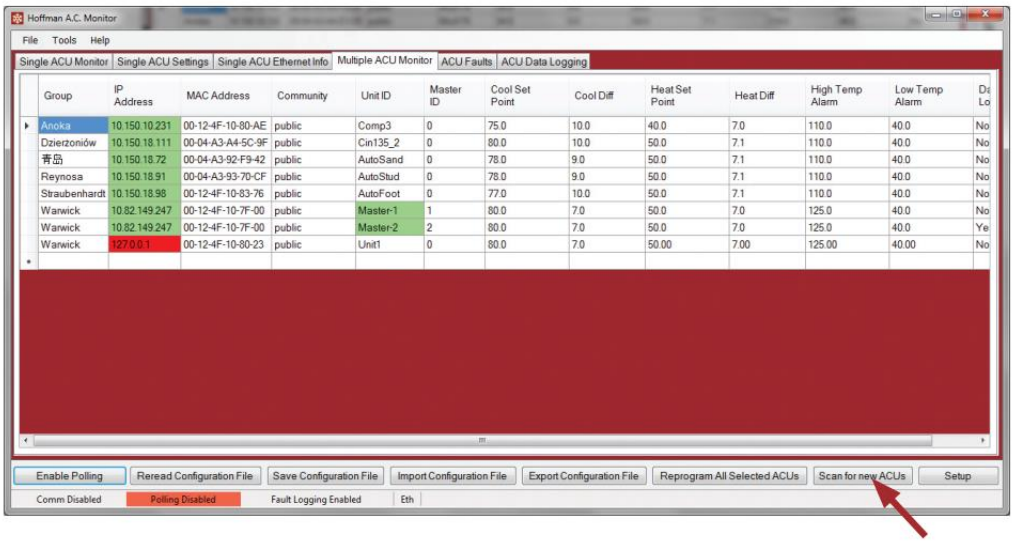
https://hoffman.nvent.com/en-us/remote-access-control

From there, download the Remote Access Control program for Windows.

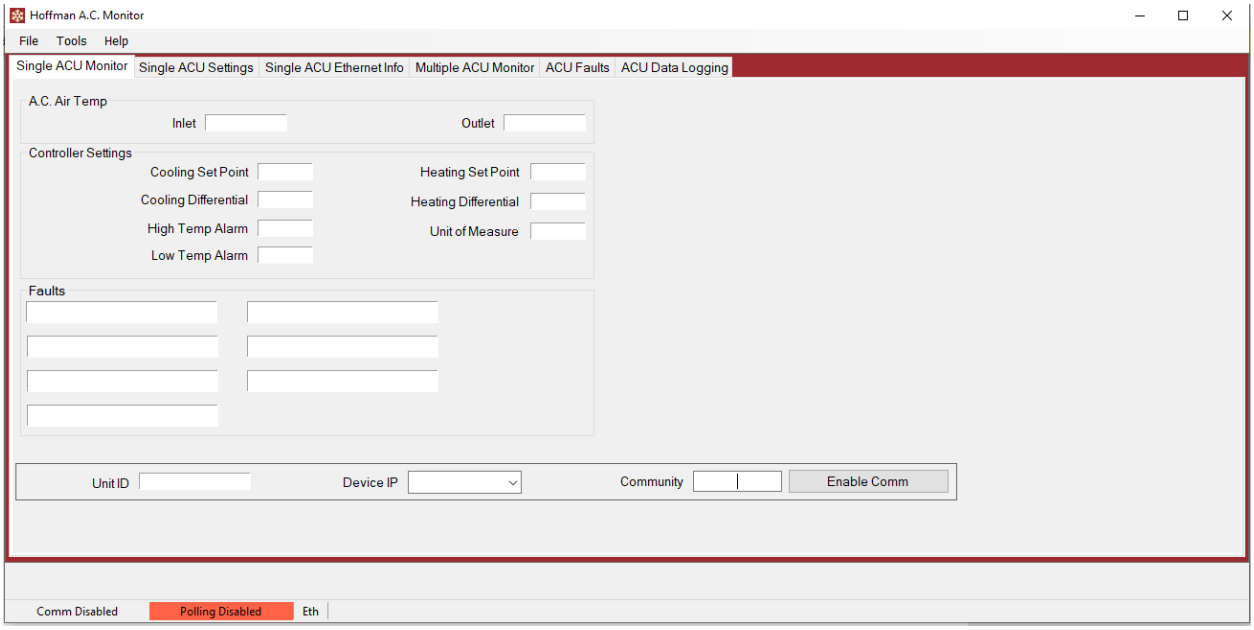
9.1.1 Setup the HVAC

The default IP-Address is 192.168.1.2.
I could not find out the default port, but I would assume it is 502.

If you don´t know the IP-Address, the Remote Access Control program can scan the network for HVACs. To use the scanning, press the following Button on the “Multiple ACU Monitor” tab.



If you know the IP-Address, you can enter IP-Address and Port as well as Community String (public or private) and press “Enable Com”.



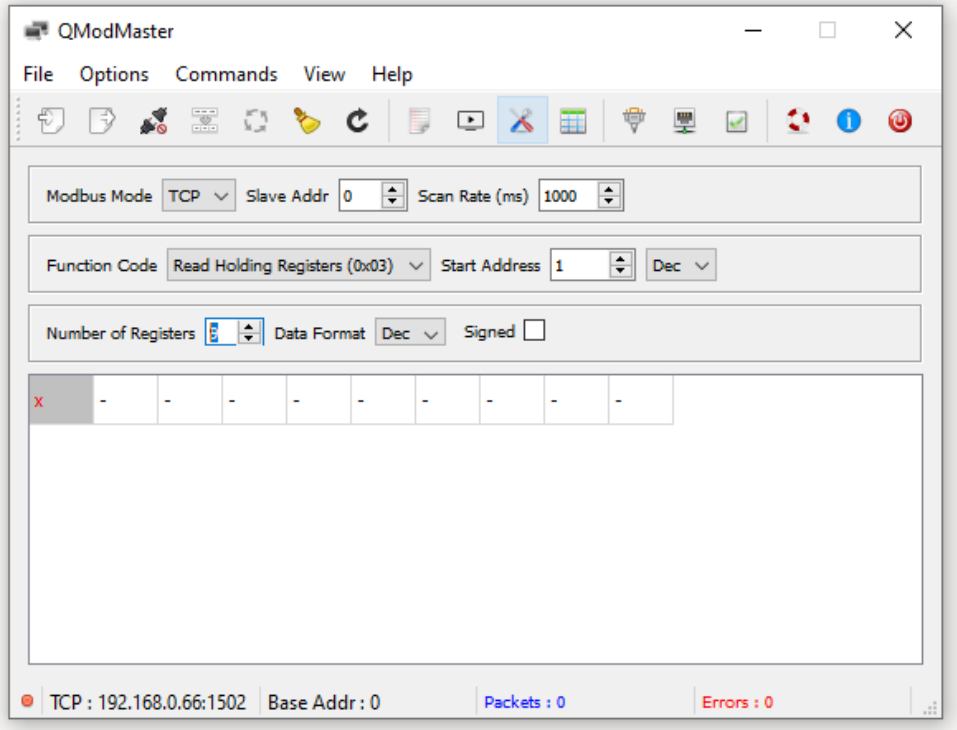
In the tab „Single ACU Settings”, all settings can be set as desired.

HVAC Modbus Map:

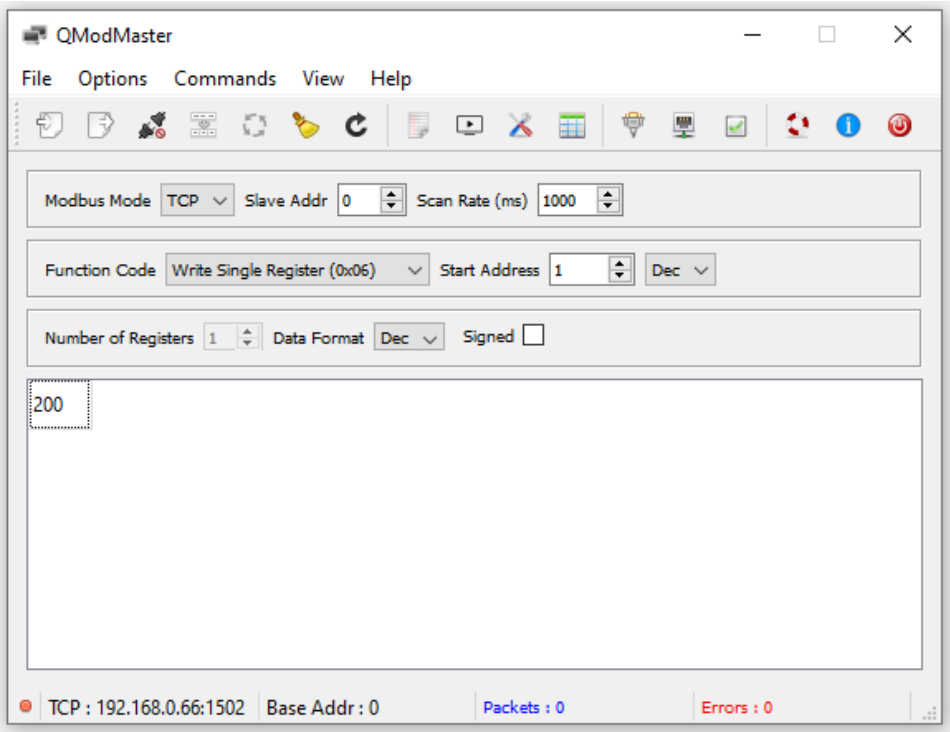
https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Telco%20Rack/Components/OCTE%20HVAC/ModbusTCPRegistersandCoils.pdf?csf=1&web=1&e=tcAeNH

9.1.2 Testing with QModMaster

Try to read current configuration of the chiller.



Try to write the cooling setpoint.



9.2 Quint4 UPS

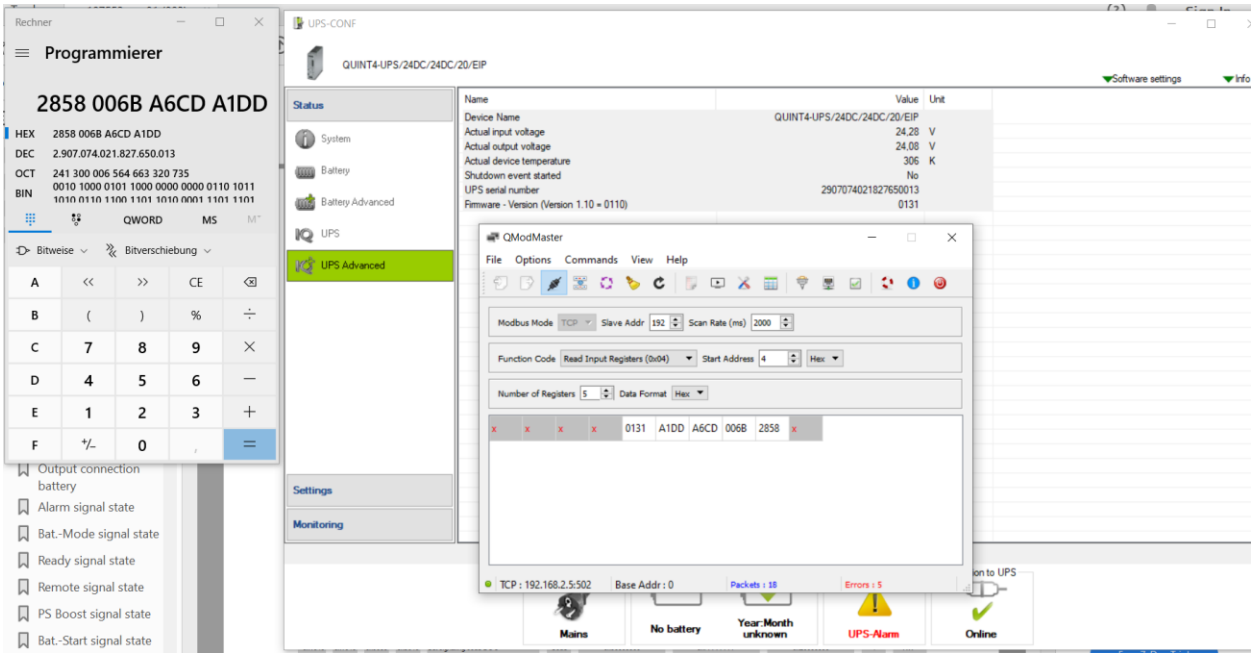
Test	Cube	CTR
Applicable		**X**

All UPS documents are in SharePoint:

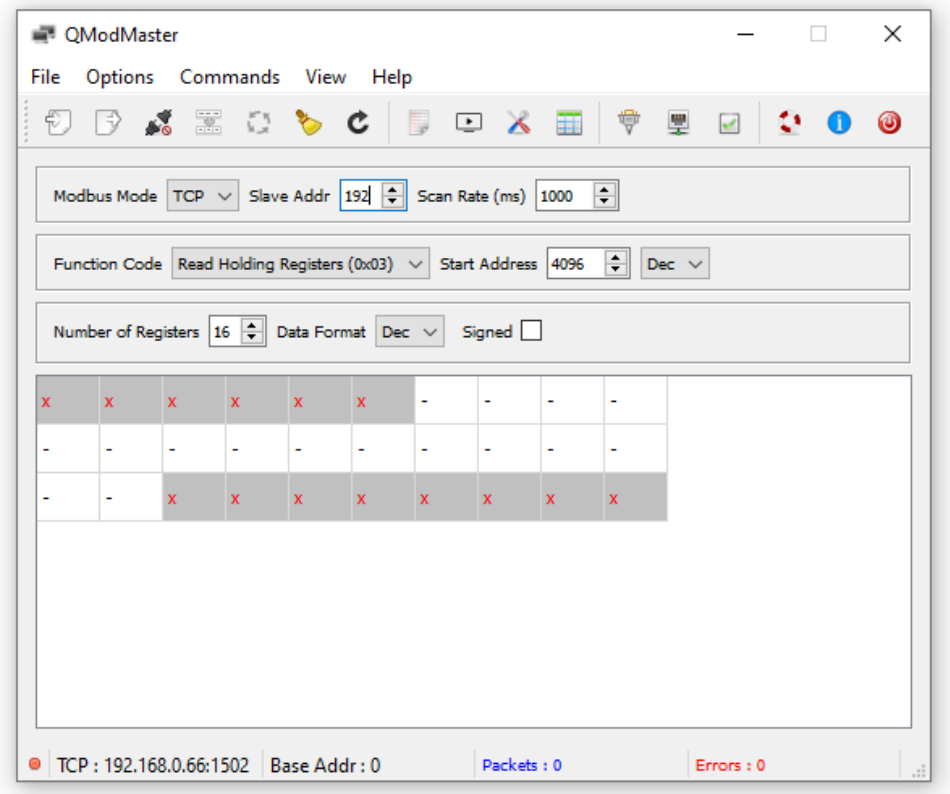
https://fluenceenergy.sharepoint.com/:f:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Documentation/Telco%20Rack/Components/OCTE%20UPS?csf=1&web=1&e=GChZIr

BOOTP is supported protocol. Port 502 is used. The Slave address for Modbus TCP is not mentioned in the manual, for Modbus RTU, it is 192. The device accept any Modbus slave ID, we internally use 192 for Modbus/TCP.

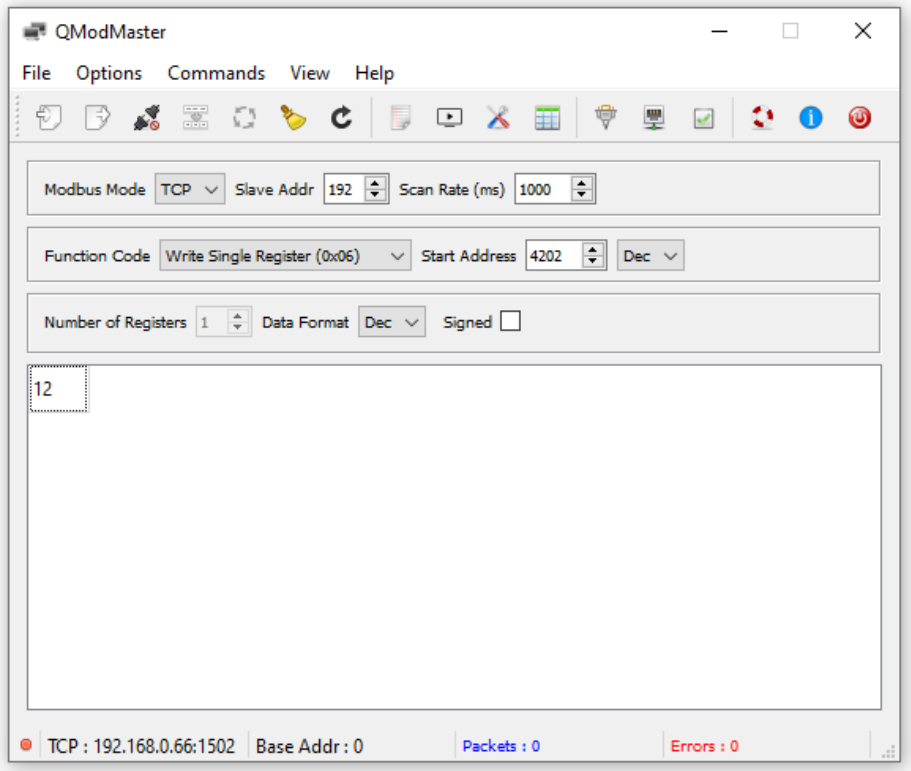
9.2.1 Testing with QModMaster



Try to read the User Device Name.



Try to write the battery alarm SOC voltage:



10. Controllino MEGA

Test	Cube	CTR
Applicable	**X**	**X**

Controllino MEGA is not directly connected to the site network, it is connected to the Banana Pi internal network (3 ethernet ports in the middle). Controllino IP is defined as **192.168.2.3**.

Controllino Mega has Web server on port 80 and Modbus server on port 502. If the testing is done on **192.168.2.XX** network, e.g. laptop directly connected to the cube internal network, user can access Controllino with web browser http:/192.168.2.3 or Modbus server on the same address with the help of QModMaster.

If Banana Pi is connected to the site or factory network, it will get IP address from the DHCP server.

Let's assume Banana Pi address will be 172.16.1.50. Banana Pi makes port forwarding 8080 to Controllino 80, and 1602 to Controllino 502. To access Controllino from outside use http:/ 172.16.1.50:8080 and 172.16.1.50:1602 for Modbus Server in Controllino MEGA.

It is possible to connect Laptop and Controllino direct over Ethernet crossover cable. In this case, complete developing environment must be installed on the laptop and the testing is cover only by developer and it is not part of the document.

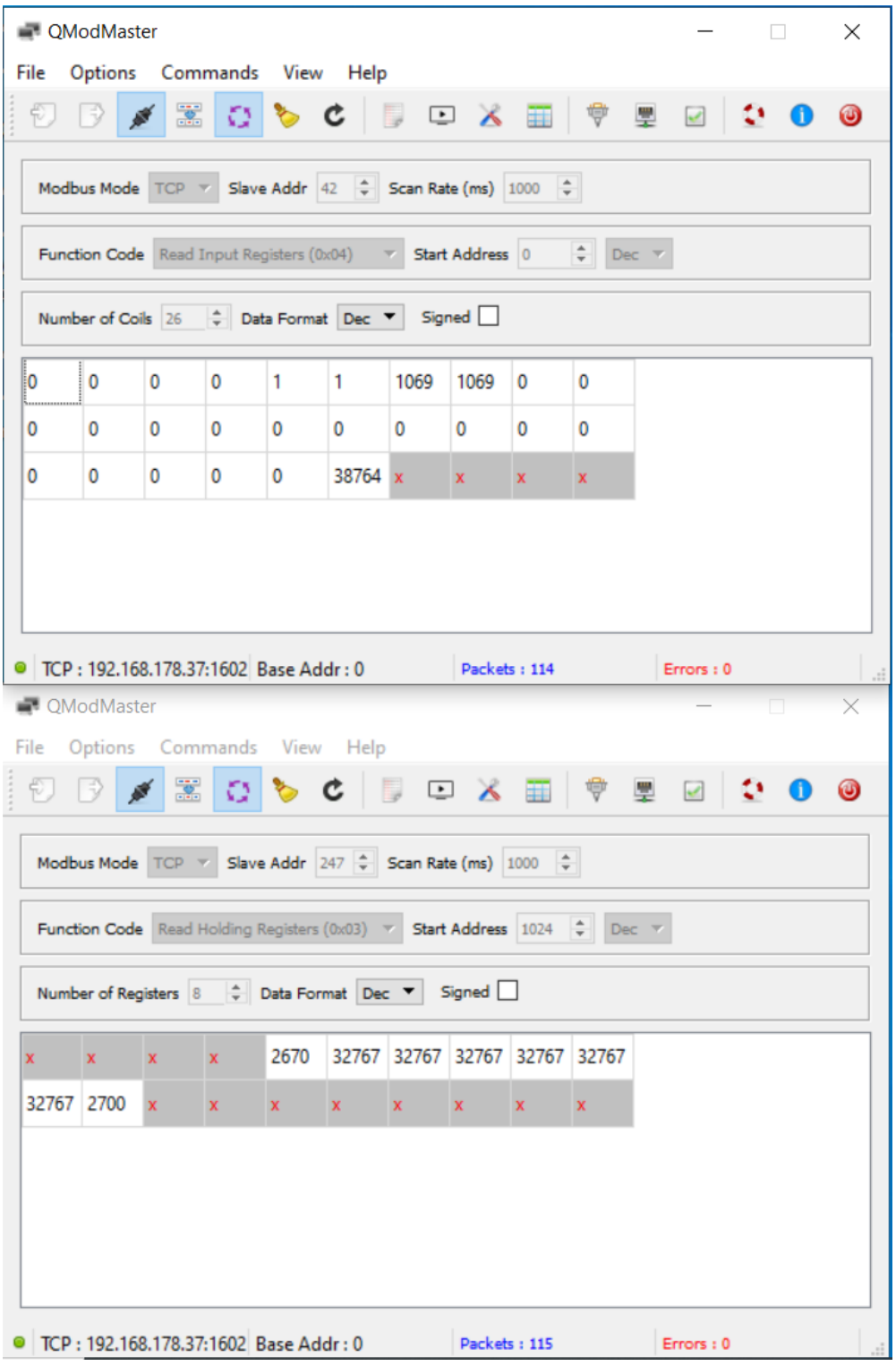
The main program for Controllino will be loaded with the last test. With correct response of the last test Controllino is programed and ready to be used.

Test equipment:

- Laptop with Ethernet port or converter (USB to Ethernet)

Example of read data over Modbus protocol

It the site network is down or not exists; the laptop can be connected to one of three internal ports. Use 192.168.2.3 and the port 502 for the communication setting inside QModMaster.



10.1 Controllino images

Controllino relevant images are located under /root/Controllino on Banana Pi.

```
root@leaf-dc2b2f355c:/ # cd /root/Controllino

root@leaf-dc2b2f355c:/root/Controllino# ls -la
total 344
drwxr-xr-x  2 root root   4096 Jul 22 06:43 .
drwx------ 10 root root   4096 Jul 22 05:58 ..
-rw-r--r--  1 root root   4665 Apr 24 11:04 Arduino_Blink.hex
-rw-rw-r--  1 root root  22792 Jul 20 05:44 Arduino_TempSensor_DS18B20.hex
-rw-rw-r--  1 root root  11149 Jul 20 07:12 Controllino_AnalogRead.hex
-rw-r--r--  1 root root   8674 Apr 24 11:15 Controllino_Blink.hex
-rw-rw-r--  1 root root   9034 Jul 20 06:46 Controllino_COSensor_AX_GS_CM_VR_100_65.hex
-rw-rw-r--  1 root root   9234 Jul 20 07:12 Controllino_DoorSwitch.hex
-rw-rw-r--  1 root root   9234 Jul 20 06:44 Controllino_EStop.hex
-rw-rw-r--  1 root root   8948 Jul 20 07:23 Controllino_Leakagesensor.hex
-rw-rw-r--  1 root root 133885 Jul 20 17:23 LeafController.hex    <- main Controllino img
-rw-r--r--  1 root root 100961 Jul 16 18:08 LeafController_ModbusConverter.hex
```

Data manipulation with Controllino is done with the help of small scripts. Controllino MEGA must be connected over USB cable to Banana Pi.

10.2 Upload Sketch to Controllino

Use script *controllino_upload.sh* to upload the appropriate Controllino image to the Controllino MEGA.

Usage:

```
controllino_upload.sh Arduino_Blink.hex
```

Response:

The blinking leds on Controllino.

10.3 Reset Controllino

Use script *controllino_reset.sh* to make soft reset of the Controllino MEGA. For hard reset use the reset button located on Controllino MEGA.

Usage:

```
controllino_reset.sh
```

Response:

The power and reset led on the Controllino will change status for small period.

The power Led is ON.

10.4 Serial monitor to Controllino (optional)

The set of the tests used from software developer. Could be avoid for production and site acceptance testing.

Use script controllino_stty.sh

Usage

```
controllino_stty.sh 115200
```
or
```
controllino_stty.sh
```

default setting is 9600. The script opens the serial communication with Controllino and prints Controllino outputs to the console.

10.4.1 Receive data from Controllino

Enter the command in the console:

```
cat /dev/ttyACM0
```

To see the hex data codes coming from the device, use the hexdump command.

```
cat /dev/ttyACM0 | hexdump -C
```

To output data from the device to the screen and to a text file, need to use tee:

```
cat /dev/ttyACM0 | tee output.txt
```

10.4.2 Send Commands to Controllino

Enter in another console:

```
echo -n "Command" > /dev/ttyACM0
```

Currently this is not used as testing case.

10.5 Controllino connected devices

The list of the sensors and devices connected to Controllino MEGA

- DS18B20 Temperature sensors (Cube only)
- Leak sensor (Cube only)
- Door Switch sensor (Cube only)
- Cube F-Stop (Cube only)

- CO Sensor (optional, Cube only)
- Osensa fiber optic temperature device (optional, Cube only)
- Envicool Chillers (Cube only)
- Envicool HVAC (Cube only)
- DL-10

For each or group of devices the appropriate Controllino test program is created.

10.5.1 DS18B20 Temperature sensors

Test	Cube	CTR
Applicable	**X**	

Usage:

```
controllino_upload.sh Arduino_TempSensor_DS18B20.hex
```

Test the output of Controllino with help of

```
controllino_stty.sh 115200
```

10.5.2 Leak sensor

Test	Cube	CTR
Applicable	**X**	

Usage:

```
controllino_upload.sh Controllino_Leakagesensor.hex
```

Test the output of Controllino with help of

```
controllino_stty.sh 115200
```

10.5.3 Door Switch sensor

Test	Cube	CTR
Applicable	**X**	**X**

Usage:

```
controllino_upload.sh Controllino_DoorSwitch.hex
```

Test the output of Controllino with help of

```
controllino_stty.sh 115200
```

Expected results:

1 - Door closed

0 - Door opened or sensor does not exist. The case of the cube with only one door

10.5.4 Cube F-Stop

Test	Cube	CTR
Applicable	**X**	

Usage:

```
controllino_upload.sh Controllino_EStop.hex
```

Test the output of Controllino with help of

```
controllino_stty.sh 115200
```

Expected results:

1 – Normal state

0 – Alarm state

10.5.5 CO Sensor (optional)

Test	Cube	CTR
Applicable	**X**	

CO Sensor AX-GS-CM-V-65 (or similar) is an optional device and it is not included in every cube design.

Usage:

```
controllino_upload.sh Controllino_COSensor_AX_GS_CM_VR_100_65.hex
```

Test the output of Controllino with help of

```
controllino_stty.sh 115200
```

10.5.6 Modbus RTU devices

Modbus RTU connected devices over RS485:

- Osensa fiber optic temperature device (Cube only)
- Envicool Chillers (Cube only)
- Envicool HVAC (Cube only)
- DL-10

Ensure every Modbus device must have unique ID and baud rate **19200 8N1**. **ID 42** is reserved of Controllino MEGA.

Test	Cube	CTR
Applicable	**X**	**X**

Usage:

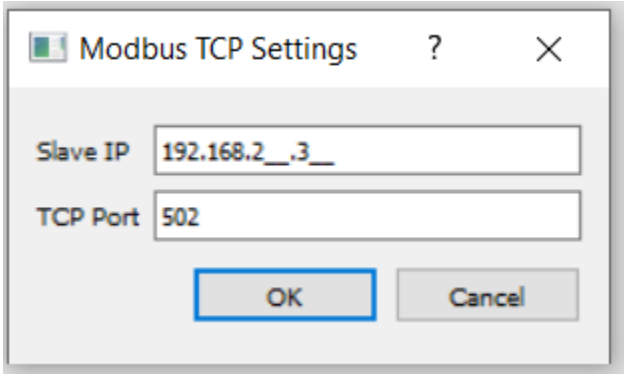
```
controllino_upload.sh LeafController.hex
```

LeafController.hex is the main and the last program for testing. User could collect data from all connected sensors or connected Modbus RTU devices.

Test each Modbus RTU device with help of QModMaster. Controllino MEGA work as Modbus Gateway. If the required Modbus ID is 42, this is Controllino MEGA itself, all other ID will be rerouted to Modbus RTU over RS 485.

Repeat the same tests as in Chapter: *Modbus RTU Devices*, but this time use Modbus TCP protocol.

Modbus settings for QModMaster:



Do not forget to change Modbus Mode to TCP when you repeat the tests.

10.5.7 Controllino as Modbus TCP server

Test	Cube	CTR
Applicable	X	X

Controllino MEGA has Modbus ID 42.

Modbus object types:

Object type	Access	Size
Coil	Read-write	1 bit
Discrete input	Read	1 bit
Input register	Read	16 bits
Holding registers	Read-write	16 bits

Normally coils a are used to write digital values to an output. Discrete inputs are used to read digital inputs. Registers are used to communicate data between the devices and usually used for analog I/Os.

Modbus functions:

Modbus	Command	Controllino function
Read Coils	0x01	read Relays
Read Discrete Inputs	0x02	not used
Read Holding Registers	0x03	read Digital Outputs
Read Input Registers	0x04	read Analog/Digital Inputs + six Temperature sensors + Heartbeat status
Write Single Coil	0x05	write Relay
Write Single Register	0x06	write DO
Write Multiple Coils	0x0f	write Relays
Write Multiple Resisters	0x10	write DOs

Test Controllino IO with help of QModMaster.

Current Cube design use only "Input Registers", where the electrical signals are connected. Relays and DO are working but nothing are connected to the Controllino.

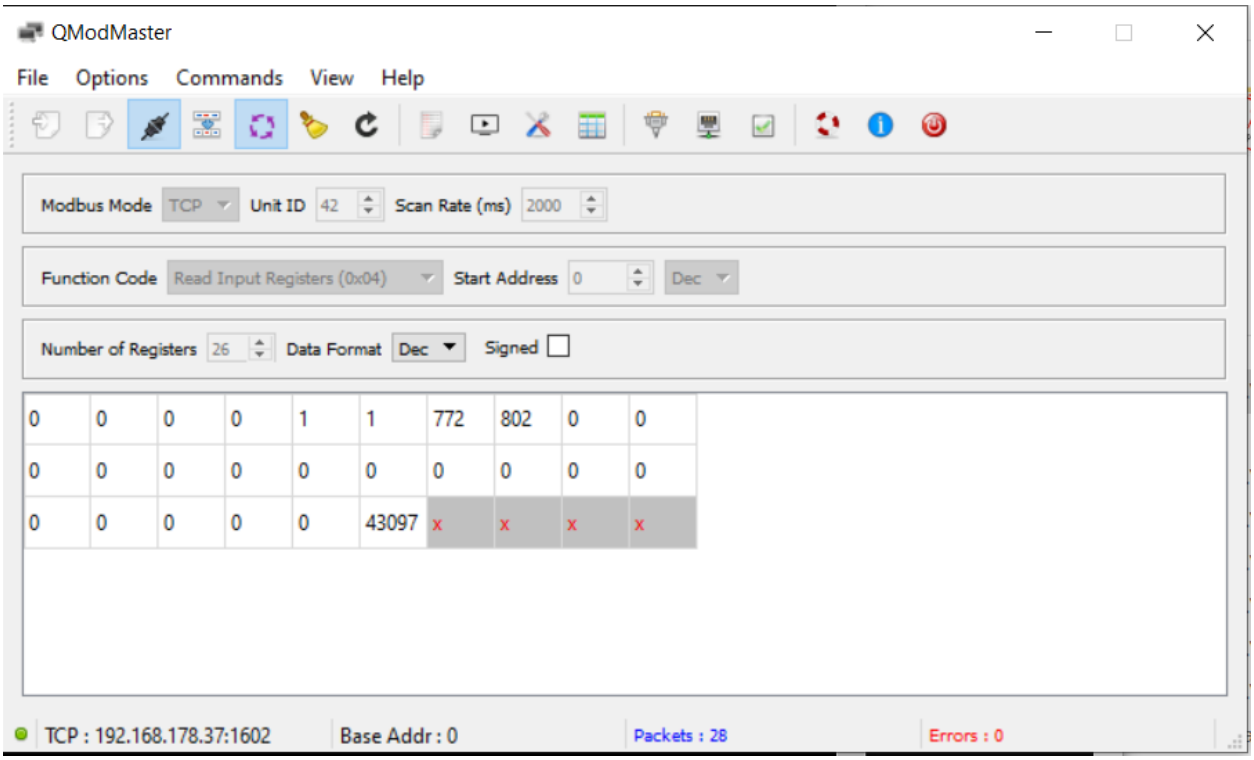
Read Input Registers consist of Controllino Analog/Digital Inputs and additionally six Temperature sensors + Heartbeat status

Controllino MEGA	**Modbus ID**	**42**	*If Controllino receives any other slave id, it is working as Modbus TPC/RTU gateway*
		Reserved	Used in different cube combination
		Spare	Not used, allocated for future expansion
		Size 2 Bytes	

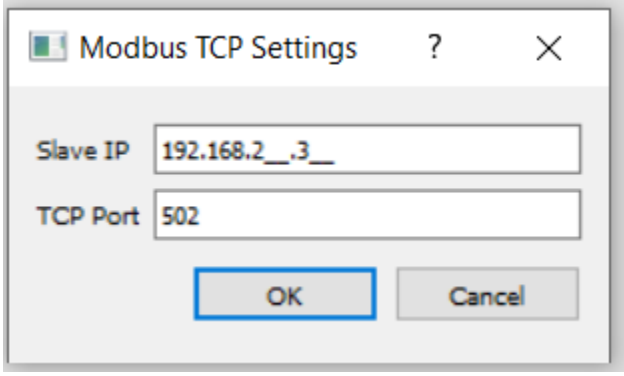
Input Registers	Signal	allowable values	Description
0	Door Contact 1	0 1	0 - Open 1 - Closed
1	Door Contact 2 Only in SD Cube	0 1	0 - Open 1 - Closed
2	Reserved		
3	Reserved		
4	Reserved		
5	Reserved		
6	Reserved		
7	Reserved		
8	F-Stop K1	0 1	1 Default 0 F-Stop K1 is activated
9	F-Stop push button	0 1	1 Default 0 F-Stop push button is activated
10	Leakage sensor	0 1	0 - no leakage 1 - Leakage detected (could be swapped, approval in progress)
11	Power Supply (Wago 787-734)	0 1	1 - OK 0 - Error
12	Reserved		
13	Reserved		
14	Reserved		
15	Reserved		
16	Spare		
17	Spare		
18	Spare		

19	Temp sensor 1	10 * Temp in C or F	Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
20	Temp sensor 2	10 * Temp in C or F	
21	Temp sensor 3	10 * Temp in C or F	
22	Reserved		
23	Reserved		
24	Reserved		
25	Heartbeat		incremental countnter

Example of Input Registers:



Modbus settings:



10.5.8 Web Tests (optional)

Test	Cube	CTR
Applicable	**X**	**X**

Test working with assumed IP (replace with proper IP address if changed). All GET works in Web browser!

Testing is possible:

- Web Browser (all get examples), e.g. copy `http://192.168.2.3/all` as address
- Windows 10 has build in cUrl program, open cmd prompt and copy the example
- Use Putty and connect to Banana Pi (192.168.2.2), from console use examples

HTTP identify

```
curl -i -X GET "http://192.168.2.3"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Show status of all relays and digital inputs and outputs

```
curl -i -X GET "http://192.168.2.3/all"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 813
User-Connection: close
```

```
Relay 00 = off
Relay 01 = off
Relay 02 = off
Relay 03 = off
Relay 04 = off
Relay 05 = off
Relay 06 = off
Relay 07 = off
Relay 08 = off
Relay 09 = off
Relay 10 = off
Relay 11 = off
Relay 12 = off
Relay 13 = off
Relay 14 = off
Relay 15 = off
DO 00 = off
DO 01 = off
DO 02 = off
DO 03 = off
DO 04 = off
DO 05 = off
DO 06 = off
DO 07 = off
DO 08 = off
DO 09 = off
DO 10 = off
DO 11 = off
DO 12 = off
DO 13 = off
DO 14 = off
DO 15 = off
DO 16 = off
DO 17 = off
DO 18 = off
DO 19 = off
DO 20 = off
DO 21 = off
DO 22 = off
DO 23 = off
DI 00 = off
DI 01 = off
DI 02 = off
DI 03 = off
DI 04 = on
DI 05 = on
DI 06 = off
DI 07 = off
DI 08 = off
DI 09 = off
DI 10 = off
DI 11 = off
```

```
DI 12 = off
DI 13 = off
DI 14 = off
DI 15 = off
DI 16 = off
DI 17 = off
DI 18 = off
```

Show status of all relays

```
curl -i -X GET "http://192.168.2.3/relays"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 423
User-Connection: close

[{"relay":0,"state":"off"},{"relay":1,"state":"off"},{"relay":2,"state":"off"},{"rela
y":3,"state":"off"},{"relay":4,"state":"off"},{"relay":5,"state":"off"},{"relay":6,"s
tate":"off"},{"relay":7,"state":"off"},{"relay":8,"state":"off"},{"relay":9,"state":"
off"},{"relay":10,"state":"off"},{"relay":11,"state":"off"},{"relay":12,"state":"off"
},{"relay":13,"state":"off"},{"relay":14,"state":"off"},{"relay":15,"state":"off"}]
```

Show relays with status off

```
curl -i -X GET "http://192.168.2.3/relays?state=off"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 423
User-Connection: close

[{"relay":0,"state":"off"},{"relay":1,"state":"off"},{"relay":2,"state":"off"},{"rela
y":3,"state":"off"},{"relay":4,"state":"off"},{"relay":5,"state":"off"},{"relay":6,"s
tate":"off"},{"relay":7,"state":"off"},{"relay":8,"state":"off"},{"relay":9,"state":"
off"},{"relay":10,"state":"off"},{"relay":11,"state":"off"},{"relay":12,"state":"off"
},{"relay":13,"state":"off"},{"relay":14,"state":"off"},{"relay":15,"state":"off"}]
```

Show status of relay 3

```
curl -i -X GET "http://192.168.2.3/relays/3"
```

Response:

```
HTTP/1.1 200 OK
```

```
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 27
User-Connection: close

[{"relay":3,"state":"off"}]
```

Show (invalid relay)

```
curl -i -X GET "http://192.168.2.3/relays/44"
```

Response:

Switch all relays on

```
curl -i -X PUT "http://192.168.2.3/relays?state=on"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Switch all relays off

```
curl -i -X PUT "http://192.168.2.3/relays?state=off"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Switch relay 3 on

```
curl -i -X PUT "http://192.168.2.3/relays/3?state=on"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Switch relay 3 off

```
curl -i -X PUT "http://192.168.2.3/relays/3?state=off"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Error (invalid value)

```
curl -i -X PUT "http://192.168.2.3/relays/3?state=blink"
```

Response:

Show status of all digital outputs

```
curl -i -X GET "http://192.168.2.3/digitaloutputs"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 567
User-Connection: close

[{"DO":0,"state":"off"},{"DO":1,"state":"off"},{"DO":2,"state":"off"},{"DO":3,"state"
:"off"},{"DO":4,"state":"off"},{"DO":5,"state":"off"},{"DO":6,"state":"off"},{"DO":7,
"state":"off"},{"DO":8,"state":"off"},{"DO":9,"state":"off"},{"DO":10,"state":"off"},
{"DO":11,"state":"off"},{"DO":12,"state":"off"},{"DO":13,"state":"off"},{"DO":14,"sta
te":"off"},{"DO":15,"state":"off"},{"DO":16,"state":"off"},{"DO":17,"state":"off"},{"
DO":18,"state":"off"},{"DO":19,"state":"off"},{"DO":20,"state":"off"},{"DO":21,"state
":"off"},{"DO":22,"state":"off"},{"DO":23,"state":"off"}]
```

Show digital outputs with status on

```
curl -i -X GET "http://192.168.2.3/digitaloutputs?state=on"
```

Response:

```
HTTP/1.1 200 OK

User-Agent: Arduino-ethernet

Content-Type: text/plain

Content-Length: 2
```

```
User-Connection: close
```

Show status of digital output 3

```
curl -i -X GET "http://192.168.2.3/digitaloutputs/3"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 24
User-Connection: close

[{"DI":3,"state":"off"}
```

Show (invalid digital outputs)

```
curl -i -X GET "http://192.168.2.3/digitaloutputs/44"
```

Response:

Switch all digital outputs on

```
curl -i -X PUT "http://192.168.2.3/digitaloutputs?state=on"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Switch all digital outputs off

```
curl -i -X PUT "http://192.168.2.3/digitaloutputs?state=off"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Switch digital output 3 on

```
curl -i -X PUT "http://192.168.2.3/digitaloutputs/3?state=on"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Switch digital output 3 off

```
curl -i -X PUT "http://192.168.2.3/digitaloutputs/3?state=off"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/html
User-Connection: close
```

Error (invalid value)

```
curl -i -X PUT "http://192.168.2.3/digitaloutputs/3?state=blink"
```

Response:

Show status of all digital inputs

```
curl -i -X GET "http://192.168.2.3/digitalinputs"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 445
User-Connection: close

[{"DI":0,"state":"off"},{"DI":1,"state":"off"},{"DI":2,"state":"off"},{"DI":3,"state"
:"off"},{"DI":4,"state":"on"},{"DI":5,"state":"on"},{"DI":6,"state":"off"},{"DI":7,"s
tate":"off"},{"DI":8,"state":"off"},{"DI":9,"state":"off"},{"DI":10,"state":"off"},{"
DI":11,"state":"off"},{"DI":12,"state":"off"},{"DI":13,"state":"off"},{"DI":14,"state
":"off"},{"DI":15,"state":"off"},{"DI":16,"state":"off"},{"DI":17,"state":"off"},{"DI
":18,"state":"off"}]
```

Show digital outputs with status on

```
curl -i -X GET "http://192.168.2.3/digitalinputs?state=on"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 45
User-Connection: close

[{"DI":4,"state":"on"},{"DI":5,"state":"on"}]
```

Show status of digital output 3

```
curl -i -X GET "http://192.168.2.3/digitalinputs/3"
```

Response:

```
HTTP/1.1 200 OK
User-Agent: Arduino-ethernet
Content-Type: text/plain
Content-Length: 24
User-Connection: close

[{"DI":3,"state":"off"}]
```

Show (invalid digital outputs)

```
curl -i -X GET "http://192.168.2.3/digitalinputs/44"
```

Response:

11. Eaton 5P850iR UPS

Test	Cube	CTR
Applicable	**X**	

Datasheet:

https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Vendor%20Info/EATON/Eaton_5P850iR_Datasheet.pdf?csf=1&web=1&e=auwhFo

Installation & User Manual 5P

https://fluenceenergy.sharepoint.com/:b:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Vendor%20Info/EATON/Eaton%20Installation%20%26%20User%20Manual_5P_Eng.pdf?csf=1&web=1&e=H5CTtA

EATON UPS is connected with Banana Pi over USB cable. Check if both part of the cable are connected. On Banana Pi is installed Network UPS Tools (*nut*). The package is use for the communication with EATON UPS.

Testing is done with the help of Laptop connected to one of the ports on Banana Pi. Required software: Putty.

11.1 Start

Normally *nut* will start automatically after power on of the Banana Pi and if the Eaton UPS is connected to the SBC. Manually start could be done with following command.

```
upsdrvctl start
```

11.2 Stop

Manually stop of the services.

```
upsdrvctl stop
```

11.3 UPS Status

11.3.1 Find all connected UPS

```
upsc -l
```

Response:

```
Init SSL without certificate database
eaton5p
```

11.3.2 Request of All UPS Stati

The command lists supported stati with current values from EATON UPS.

```
upsc eaton5p
```

Response:

```
Init SSL without certificate database
battery.capacity: 7.20
battery.charge: 100
battery.charge.low: 20
battery.charge.restart: 0
battery.charger.status: floating
battery.energysave: no
battery.energysave.delay: 300
battery.energysave.load: 5
battery.protection: yes
battery.runtime: 7400
battery.type: PbAc
battery.voltage: 27.1
battery.voltage.nominal: 24
device.mfr: EATON
device.model: 5P 850
device.serial: G115K44007
device.type: ups
driver.name: usbhid-ups
driver.parameter.pollfreq: 30
driver.parameter.pollinterval: 2
driver.parameter.port: auto
driver.parameter.synchronous: no
driver.version: 2.7.4
driver.version.data: MGE HID 1.39
driver.version.internal: 0.41
input.current: 0.10
input.frequency: 49.9
input.frequency.extended: no
input.frequency.nominal: 50
input.sensitivity: normal
input.transfer.boost.low: 184
input.transfer.high: 281
input.transfer.low: 160
input.transfer.trim.high: 253
input.voltage: 227.3
input.voltage.extended: no
input.voltage.nominal: 230
outlet.1.autoswitch.charge.low: 0
outlet.1.delay.shutdown: 65535
outlet.1.delay.start: 3
outlet.1.desc: PowerShare Outlet 1
outlet.1.id: 1
outlet.1.status: on
outlet.1.switchable: yes
outlet.2.autoswitch.charge.low: 0
outlet.2.delay.shutdown: 65535
outlet.2.delay.start: 6
outlet.2.desc: PowerShare Outlet 2
outlet.2.id: 2
outlet.2.status: on
```

```
outlet.2.switchable: yes
outlet.desc: Main Outlet
outlet.id: 0
outlet.switchable: no
output.current: 0.00
output.frequency: 49.9
output.frequency.nominal: 50
output.powerfactor: 0.00
output.voltage: 227.3
output.voltage.nominal: 220
ups.beeper.status: enabled
ups.delay.shutdown: 20
ups.delay.start: 30
ups.efficiency: 0
ups.firmware: 02.14.0026
ups.load: 0
ups.load.high: 105
ups.mfr: EATON
ups.model: 5P 850
ups.power: 0
ups.power.nominal: 850
ups.productid: ffff
ups.realpower: 0
ups.realpower.nominal: 600
ups.serial: G115K44007
ups.shutdown: enabled
ups.start.auto: yes
ups.start.battery: yes
ups.start.reboot: yes
ups.status: OL CHRG
ups.test.interval: 2592000
ups.test.result: Done and passed
ups.timer.shutdown: -1
ups.timer.start: -1
ups.type: offline / line interactive
ups.vendorid: 0463
```

11.3.3 Request of single UPS Status

11.3.3.1 UPS Battery Capacity

```
upsc eaton5p battery.capacity
```

Response:

```
Init SSL without certificate database
7.20
```

11.3.3.2 UPS Battery Charge

```
upsc eaton5p battery.charge
```

Response:

```
Init SSL without certificate database
100
```

11.3.3.3 UPS Input Voltage

```
upsc eaton5p input.voltage
```

Response:

```
Init SSL without certificate database
227.3
```

11.3.3.4 UPS Input Frequency

```
upsc eaton5p input.frequency
```

Response:

```
Init SSL without certificate database
49.9
```

11.4 Supported command for EATON UPS

The command lists supported commands for EATON UPS.

```
upscmd -l eaton5p
```

Response:

```
Instant commands supported on UPS [eaton5p]:

beeper.disable - Disable the UPS beeper
beeper.enable - Enable the UPS beeper
beeper.mute - Temporarily mute the UPS beeper
beeper.off - Obsolete (use beeper.disable or beeper.mute)
beeper.on - Obsolete (use beeper.enable)
load.off - Turn off the load immediately
load.off.delay - Turn off the load with a delay (seconds)
```

```
load.on - Turn on the load immediately
load.on.delay - Turn on the load with a delay (seconds)
outlet.1.load.off - Turn off the load on outlet 1 immediately
outlet.1.load.on - Turn on the load on outlet 1 immediately
outlet.2.load.off - Turn off the load on outlet 2 immediately
outlet.2.load.on - Turn on the load on outlet 2 immediately
shutdown.return - Turn off the load and return when power is back
shutdown.stayoff - Turn off the load and remain off
shutdown.stop - Stop a shutdown in progress
test.battery.start.deep - Start a deep battery test
test.battery.start.quick - Start a quick battery test
test.battery.stop - Stop the battery test
```

Be careful with the following command. Switching off USP loads, or group of loads could lead power off Banana Pi and the other controls. Check schematics and prove how the devices are connected. It is recommended to have independent power supply for Banana Pi during UPS tests.

11.4.1 EATON UPS all loads off

```
upscmd -u upsuser -p ups eaton5p load.off
```

Response:

```
OK
```

Check if all connected devices to Eaton UPS are switched off.

11.4.2 EATON UPS all loads on

```
upscmd -u upsuser -p ups eaton5p load.on
```

Response:

```
OK
```

Check if all connected devices to Eaton UPS are switched on.

11.4.3 EATON UPS Outlet 1 load off

```
upscmd -u upsuser -p ups eaton5p outlet.1.load.off
```

Response:

```
OK
```

Check if all connected devices to Eaton UPS outlet 1 are switched off.

11.4.4 EATON UPS Outlet 1 load on

```
upscmd -u upsuser -p ups eaton5p outlet.1.load.on
```

Response:

```
OK
```

Check if all connected devices to Eaton UPS outlet 1 are switched on.

11.4.5 EATON UPS Outlet 2 load off

```
upscmd -u upsuser -p ups eaton5p outlet.2.load.off
```

Response:

```
OK
```

Check if all connected devices to Eaton UPS outlet 2 are switched off.

11.4.6 EATON UPS Outlet 2 load on

```
upscmd -u upsuser -p ups eaton5p outlet.2.load.on
```

Response:

```
OK
```

Check if all connected devices to Eaton UPS outlet 2 are switched on.

11.5 UPS Power cycling

The script `fluence_cube-power-cycling.sh` encapsulates soft and hard power cycling of the Eaton UPS. Soft power cycling sends the commands to the UPS over USB cable. Hard power cycling is manipulating with additional time relay over Controllino relay No 0.

Available command:

```
root@leaf-a7d9b205f9:/usr/local/bin# fluence_cube-power-cycling.sh help
Usage:
  fluence_cube-power-cycling.sh [soft]
    Software reset (power cycling) of EATON UPS, with shutdown of the controller.
With time delay of 15 seconds for switch off
  fluence_cube-power-cycling.sh hard
    Immediately hard reset (power cycling) of EATON UPS, without shutdown of the
controller
  fluence_cube-power-cycling.sh on
    Immediately All Load On of EATON UPS
  fluence_cube-power-cycling.sh off
    Immediately All Load Off of EATON UPS
  fluence_cube-power-cycling.sh outlet1on
    Immediately Outlet 1 Load On of EATON UPS
  fluence_cube-power-cycling.sh outlet1off
    Immediately Outlet 1 Load Off of EATON UPS
  fluence_cube-power-cycling.sh outlet2on
    Immediately Outlet 2 Load On of EATON UPS
  fluence_cube-power-cycling.sh outlet2off
    Immediately Outlet 2 Load Off of EATON UPS
```

11.5.1 UPS Soft power cycling

Soft power cycling will send command to switch off the cube with delay of 15 seconds. This is used to shutdown the controller in the cube. 10 seconds after power is off, the UPS will which switch back the power.

Usage:

```
root@leaf-a7d9b205f9:/usr/local/bin# fluence_cube-power-cycling.sh soft
```

or

```
root@leaf-a7d9b205f9:/usr/local/bin# fluence_cube-power-cycling.sh
```

11.5.2 UPS Hard power cycling

Hard power cycling immediately shut down the power in the cube. Time relay will switch the power back after delay of 10 seconds. *In the normal operation of the cube, hard power cycling should be used if soft power cycling is not working for some reason (e.g. communication problem over USB)*

Usage:

```
root@leaf-a7d9b205f9:/usr/local/bin# fluence_cube-power-cycling.sh hard
```

12. Modbus-Script

12.1 Introduction

Test	Cube	CTR
Applicable	**X**	**X**

The Modbus script is based on the Python Library PyModbus. When running on the Cube Controller or Core Telco Rack Controller, it is capable of reading connected Modbus-Devices, like the Chillers, HVACs, Sensors, etc. It is also capable to act as a server, so the values read from the clients will be buffered and can be passed on immediately when requested, as the server and the client-functionality run asynchronously.

The script is located at the following path:

```
/user/local/bin/controllino_modbus.py
```

Before testing, make sure the script is not already running by calling:

```
ps -aux | grep controllino_modbus.py
```

or if the script is started form the service:

```
systemctl status fluence-modbus
```

If the script is running, it was probably started by the service. To disable the service, use the following command:

```
systemctl stop fluence-modbus
```

Check again if the script is running by using the ps-aux command. If it is still running, note the process ID and kill it. Example:

```
root       417  0.0  0.5  28752 11688 ?        Ssl  Aug26   0:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-startup-triggers
root      5955 16.4  2.1  60968 43784 pts/0    T    23:11   0:03 python3 controllino_modbus.py 192.168.2.3 502 notserver notcyclic
root      6040  0.0  0.0   4168  1344 pts/0    S+   23:11   0:00 grep --color=auto python
```

```
kill 5955
```

If the normal kill command does not work, try:

```
sudo kill -9 5955
```

12.2 Testing

12.2.1 Test Client-Functionality

Afterwards the script can be tested. It is recommended to test the clients first one by one, and afterwards the server-function. The following pages describe how to test the devices one by one. When comparing the results with example results, especially watch out for 0xFF (255) values. The dataframes in the script are initialized with those values and if they are still 0xFF that means they probably could not be read. This is expected in some cases (for example HVAC Baudrate, Heartrate and ID, as those values are not yet implemented in the HVAC).

For your information: When starting the script, the settings will be displayed at first. Example settings can be seen below:

```
#==============Settings==============#
The Following Settings have been applied

MODE: Manual
clientIpAddress: 192.168.0.66
clientPort: 502
serverIpAddress: 192.168.0.66
serverPort: 1502
Server Active: False
Clients Active: True
cyclicRead: False
controllinoRead: True
chiller1Read: False
chiller2Read: False
hvac1Read: False
hvac2Read: False
osensaRead: False
dl10Read: False
flowmeterRead: False
upsRead: False
virtualSlaveRead: False
sleepTime: 2
controllinoUnitId: 42
chiller1UnitId: 1
chiller2UnitId: 2
hvac1UnitId: 3
hvac2UnitId: 4
osensaUnitId: 10
dl10UnitId: 5
flowmeterUnitId: 20
upsUnitId: 21
virtualSlaveUnitId: 100
Press Enter to continue.
```

12.2.1.1 Controllino

Test	Cube	CTR
Applicable	**X**	**X**

Use the following command to test the Controllino:

```
controllino_modbus.py controllino
```

Example Output

```
# ============ New Data ============ #


# ============ Controllino (Input Registers) ============ #
                         address  value writeaccess
signal_name
door1                          0      0          ro
door2                          1      0          ro
reserved1                      2      0          ro
CoreIMD                        3      0          ro
fStopButton                    4      0          ro
bmsFstopButton                 5      0          ro
preFirePanelSignalPoint        6      0          ro
mainFstop                      7      0          ro
fstopK1Relay                   8      0          ro
fstopPushButton                9      1          ro
leakageSens                   10      1          ro
powerSupp                     11      1          ro
spf                           12      0          ro
ups                           13      0          ro
hvac                          14      0          ro
mvTransPress                  15      0          ro
mvTransOilLevel               16      0          ro
mvTransOilTemp                17      0          ro
mvTransWindingTemp            18      0          ro
tempSens1                     19    190          ro
tempSens2                     20    196          ro
tempSens3                     21    190          ro
reserved11                    22      0          ro
reserved12                    23      0          ro
reserved13                    24      0          ro
heartbeat                     25  46938          ro


# ============ Controllino (Coils) ============ #
             address  value writeaccess
signal_name
reserved1          0  False          ro
reserved2          1  False          ro

Shutdown of Client Readout Thread and Server
```

12.2.1.2 Envicool Chiller

Test	Cube	CTR
Applicable	**X**	

Use the following command to test the Chiller 1 (Chiller 2 should also be tested the same way):

```
controllino_modbus.py chiller1
```

Example Output

```
# ============ New Data ============ #


# ============ Chiller 1 ============ #
                        address  value writeaccess
signal_name
systemOnOff                1024      1          rw
modeSelection              1025      2          rw
waterTempSet              35596    180          rw
hysteresisSet             35598     30          rw
flowRateSelection         38921      0          rw
supplyWaterTemp           40960    189          ro
returnWaterTemp           40962    193          ro
environmentTemp           40973  32767          ro
outletHighWaterTemp       45312      0          ro
outletLowWaterTemp        45313      0          ro
outletWaterTempSensFail   45314      0          ro
returnWaterTempSensFail   45316      0          ro
heatingFail               45323     32          ro
pumpFail                  45329     32          ro
inverterComFail           45333      0          ro
highSystemPressAlarm      45340      0          ro
highOutletPressAlarm      45360      0          ro
WaterReplenishmentAlarm   45362      0          ro
sysHighVoltageLock        45367      0          ro
sysLowVoltageLock         45368      0          ro
exhaustGasHighTempLock    45369      0          ro
inverterOverCurrentLock   45370      0          ro
inverterOverTempLock      45371      0          ro
inverterOverVoltLock      45372      0          ro
inverterUnterVoltLock     45373      0          ro
inverterPhaseLossLock     45374      0          ro
inverterOtherFaultLock    45375      0          ro
heatingFaultLock          45378     32          ro
pumpCommandSpeed          41473    790          ro
heartbeat                 32772    255          ro
id                        36864    255          rw
baudrate                  35337    255          rw

Shutdown of Client Readout Thread and Server
```

12.2.1.3 Envicool HVAC

Test	Cube	CTR
Applicable	**X**	

Use the following command to test the HVAC 1 (HVAC 2 should also be tested the same way):

```
controllino_modbus.py hvac1
```

Example Output

```
# ============ New Data ============ #


# ============ Hvac 1 ============ #
                      address  value writeaccess
signal_name
softwareVersion             0    255          ro
unitRunningStatus         256      1          ro
internalFanStatus         257      1          ro
externalFanStatus         258      0          ro
compressorStatus          259      0          ro
heaterStatus              260      0          ro
emergencyFanStatus        261      2          ro
evapTemp                 1280    193          ro
outdoorTemp              1281   2000          ro
condenserTemp            1282    188          ro
indoorTemp               1283    198          ro
humidity                 1284    120          ro
dischargeTemp            1285   2000          ro
acRunningCurrent         1286  32767          ro
acInputVoltage           1287      0          ro
dcInputVoltage           1288      0          ro
highTempAlarm            1536      0          ro
lowTempAlarm             1537      0          ro
highHumidAlarm           1538      0          ro
lowHumidAlarm            1539      0          ro
coilFreezeProtection     1540      0          ro
highExhaustTempAlarm     1541      0          ro
evapTempSensFail         1542      0          ro
outdoorTempSensFail      1543      0          ro
condensTempSensFail      1544      0          ro
indoorTempSensFail       1545      0          ro
exhaustTempSensFail      1546      0          ro
humidSensFail            1547      0          ro
internalFanFailAlarm     1548      0          ro
externalFanFailAlarm     1549      0          ro
compressorFailAlarm      1550      0          ro
heaterFailAlarm          1551      0          ro
emergencyFanFailAlarm    1552      0          ro
hpAlarm                  1553      0          ro
lpAlarm                  1554      0          ro
waterAlarm               1555      0          ro
```

```
fireAlarm                1556      0          ro
gatingAlarm              1557      0          ro
hpLock                   1558      0          ro
lpLock                   1559      0          ro
highExhaustTempLock      1560      0          ro
acOverVoltageAlarm       1561      0          ro
acUnderVoltageAlarm      1562      0          ro
acPowerSupplyFail        1563      0          ro
losePhaseAlarm           1564      0          ro
freqFault                1565      0          ro
antiPhaseAlarm           1566      0          ro
dcOverVoltageAlarm       1567      0          ro
dcUnderVoltageAlarm      1568      0          ro
refrigStopPoint          1792     25          rw
refrigBand               1793      3          rw
heatingStopPoint         1794     17          rw
heatingBand              1795      5          rw
reserve                  1796     60          ro
reserve1                 1797     10          ro
highTempPoint            1798     45          rw
lowTempPoint             1799      5          rw
highHumPoint             1800     90          rw
internFanStopPoint       1802  65516          rw
Shutdown of Client Readout Thread and Server
```

12.2.1.4 Osensa Temperature Sensor (Optional)

Test	Cube	CTR
Applicable	**X**	

Use the following command to test the Osensa Temperature Sensor:

```
controllino_modbus.py osensa
```

Example Output

```
# ============ New Data ============ #


# ============ Osensa ============ #
             address  value writeaccess
signal_name
osensaTemp1     1024   1750          ro
osensaTemp2     1025   1790          ro
osensaTemp3     1026   1760          ro
osensaTemp4     1027   1760          ro
osensaTemp5     1028   1840          ro
osensaTemp6     1029   1890          ro
osensaTemp7     1030   1940          ro
osensaTemp8     1031  32767          ro

Shutdown of Client Readout Thread and Server
```

12.2.1.5 DL-10 Temperature Sensor

Test	Cube	CTR
Applicable	**X**	**X**

Use the following command to test the DL-10 Temperature Sensor:

```
controllino_modbus.py dl10
```

Example Output

```
# ============ New Data ============ #


# ============ Dl10 ============ #
             address  value writeaccess
signal_name
dl10Humid          0   5212          ro
dl10TempC          1   2253          ro
dl10TempF          2   7255          ro
Shutdown of Client Readout Thread and Server
```

12.2.1.6 Flowmeter (Optional)

Test	Cube	CTR
Applicable	**X**	

Use the following command to test the Flowmeter:

```
controllino_modbus.py flowmeter
```

Example Output

```
# ============ New Data ============ #


# ============ Flow Meter ============ #
                                    address     value writeaccess
signal_name
FlowMeterDeviceID                         0       172          ro
FlowMeterStatus                           1         0          ro
FlowMeterSystem Type                      2         4          ro
FlowMeterSerialIdentifier                 3         0          ro
FlowMeterSerialIdentifier2                4     21269          ro
FlowMeterSerialIdentifier3                5     28672          ro
FlowMeterMeasuredVelocity                 6  0.676044          ro
FlowMeterMeasuredVelocity2                7         0          ro
FlowMeterMeasuredFlow                     8   2.77471          ro
FlowMeterMeasuredFlow2                    9         0          ro
FlowMeterCalculatedPower                 10         0          ro
FlowMeterCalculatedPower2                11         0          ro
FlowMeterCalculatedEnergy                12         0          ro
FlowMeterCalculatedEnergy2               13         0          ro
FlowMeterMeasuredTemperatureHot          14         0          ro
FlowMeterMeasuredTemperatureHot2         15         0          ro
FlowMeterMeasuredTemperatureCold         16         0          ro
FlowMeterMeasuredTemperatureCold2        17         0          ro
FlowMeterMeasuredTemperatureDifference   18         0          ro
FlowMeterMeasuredTemperatureDifference2  19         0          ro
FlowMeterMeasuredVolumeTotal             20   1942.17          ro
FlowMeterMeasuredVolumeTotal2            21         0          ro
FlowMeterInstrument Units                22         0          ro
FlowMeterInstrumentGain                  23        47          ro
FlowMeterInstrumentSNR                   24        43          ro
FlowMeterInstrumentSignal                25        88          ro
FlowMeterMeasuredDeltaTimeDifference     26   47.8337          ro
FlowMeterMeasuredDeltaTimeDifference2    27         0          ro
FlowMeterInstrumentETA                   28   161.496          ro
FlowMeterInstrumentETA2                  29         0          ro
FlowMeterInstrumentATA                   30       157          ro
FlowMeterInstrumentATA2                  31         0          ro

Shutdown of Client Readout Thread and Server
```

12.2.1.7 Eaton UPS

Test	Cube	CTR
Applicable	**X**	

Use the following command to test the UPS:

```
controllino_modbus.py ups
```

Example Output

```
# ============ UPS ============ #
                              address  value writeaccess
signal_name
battery_capacity                    0     72          ro
battery_charge                      1    100          ro
battery_charge_low                  2     20          ro
battery_charge_restart              3      0          ro
battery_charger_status              4    255          ro
battery_energysave                  5      0          ro
battery_energysave_delay            6    300          ro
battery_energysave_load             7      5          ro
battery_protection                  8      1          ro
battery_runtime                     9   6200          ro
battery_type                       10      0          ro
battery_voltage                    11    263          ro
battery_voltage_nominal            12     24          ro
device_mfr                         13      0          ro
device_model                       14      0          ro
device_serial                      15    255          ro
device_type                        16      0          ro
driver_name                        17      0          ro
driver_parameter_pollfreq          18     30          ro
driver_parameter_pollinterval      19      2          ro
driver_parameter_port              20      0          ro
driver_parameter_synchronous       21      0          ro
driver_version                     22    274          ro
driver_version_data                23      0          ro
driver_version_internal            24      4          ro
input_current                      25      4          ro
input_frequency                    26    599          ro
input_frequency_extended           27      0          ro
input_frequency_nominal            28     60          ro
input_sensitivity                  29      0          ro
input_transfer_boost_low           30    184          ro
input_transfer_high                31    281          ro
input_transfer_low                 32    160          ro
input_transfer_trim_high           33    253          ro
input_voltage                      34   2346          ro
input_voltage_extended             35      0          ro
input_voltage_nominal              36    230          ro
outlet_1_autoswitch_charge_low     37      0          ro
outlet_1_delay_shutdown            38  65535          ro
```

outlet_1_delay_start	39	3	ro
outlet_1_desc	40	0	ro
outlet_1_id	41	1	ro
outlet_1_status	42	1	ro
outlet_1_switchable	43	1	ro
outlet_2_autoswitch_charge_low	44	0	ro
outlet_2_delay_shutdown	45	65535	ro
outlet_2_delay_start	46	6	ro
outlet_2_desc	47	0	ro
outlet_2_id	48	2	ro
outlet_2_status	49	1	ro
outlet_2_switchable	50	1	ro
outlet_desc	51	0	ro
outlet_id	52	0	ro
outlet_switchable	53	0	ro
output_current	54	6	ro
output_frequency	55	599	ro
output_frequency_nominal	56	60	ro
output_powerfactor	57	1	ro
output_voltage	58	2346	ro
output_voltage_nominal	59	220	ro
ups_beeper_status	60	1	ro
ups_delay_shutdown	61	20	ro
ups_delay_start	62	30	ro
ups_efficiency	63	40	ro
ups_firmware	64	0	ro
ups_load	65	16	ro
ups_load_high	66	105	ro
ups_mfr	67	0	ro
ups_model	68	0	ro
ups_power	69	141	ro
ups_power_nominal	70	850	ro
ups_productid	71	65535	ro
ups_realpower	72	18	ro
ups_realpower_nominal	73	600	ro
ups_serial	74	255	ro
ups_shutdown	75	1	ro
ups_start_auto	76	1	ro
ups_start_battery	77	1	ro
ups_start_reboot	78	1	ro
ups_status	79	255	ro
ups_test_interval	80	65535	ro
ups_test_result	81	0	ro
ups_timer_shutdown	82	-1	ro
ups_timer_start	83	-1	ro
ups_type	84	0	ro
ups_vendorid	85	463	ro
readTimestamp0	86	0	ro
readTimestamp1	87	0	ro
readTimestamp2	88	24546	ro
readTimestamp3	89	49652	ro
readAlarm	96	0	ro
heartbeat	97	1	ro

```
Shutdown of Client Readout Thread and Server
```

12.2.2 Test Server-Functionality

Test	Cube	CTR
Applicable	X	X

Open the Config file with the following command:

```
sudo nano /usr/local/share/fluence/controllino_modbus.conf
```

A config file similar to the one below should appear:

```
# Config File for the controllino_modbus Script

# The mode determines the behaviour of the script:
# 'Testing' will simulate a modbus connection by using random values, if connection
is impossible
# 'Manual' will use input values from the user.
# 'Production' won't request any input from the user and will not print out any
values, only errors
mode = Production

# IP-Adresses and Ports
clientIpAddress = 192.168.2.3
clientPort: 502
#serverIpAddress: 172.16.1.111 #Usually the script will choose the correct Ip-
Address.
serverPort: 1502

#Enable/disable the server functionality of the script.
ServerActive: True

#If set to false, no client will be read, if set to true all enabled clients will be
read.
ClientsActive: True

# If set to false, every client will be read only once.
cyclicRead: True

#enable/disable single devices
controllinoRead: True
chiller1Read: True
chiller2Read: True
hvac1Read: True
hvac2Read: True
osensaRead: True
dl10Read: True
flowmeterRead: True
upsRead: True
virtualSlaveRead: True

# Time between read cycles
sleepTime: 2
```

```
#Modbus IDs
controllinoUnitId = 42
chiller1UnitId = 1
chiller2UnitId = 2
hvac1UnitId = 3
hvac2UnitId = 4
osensaUnitId = 10
dl10UnitId = 5
flowmeterUnitId = 20
virtualSlaveUnitId = 100
```

Read the Server IP-Address and the Server-Port from the Settings and remember it. Close the config file without changing anything by pressing ctrl+x.

Afterwards start the script without arguments, so the values from the config file will be not be ignored:

```
controllino_modbus.py
```

Now open QModMaster on a laptop, which is connected to the network and enter the Server IP-Address and the Server-Port in the ModbusTCP Settings. QModMaster should be able to read all values from the Modbus Script. As an example, enter Slave Addr 42 (Controllino) and try to read the first 10 Input-Registers by pressing the “Scan”-Button. The values should match with the values that were previously printed out in the script.

12.2.3 Test Telco-Rack-specific devices

12.2.3.1 HVAC (Telco Rack)

Test	Cube	CTR
Applicable		**X**

Use the following command to test the HVAC located in the Telco Rack:

```
controllino_modbus.py 192.168.1.2 502 hvacTelco
(use correct IP-Address and port as first and second argument)
```

Example Output

```
IMPORTANT: THIS IS ONLY AN EXAMPLE OUTPUT. NORMALLY THE VALUES SHOULD NOT BE 255.
VALUES OF 255 MEAN THE HVAC COULD NOT BE READ.


# ============ New Data ============ #
# ============ HVAC (Telco) ============ #
# ------------ Coils ------------ #
address  value writeaccess
signal_name
unitOfMeasure             1    255          ro
frostAlarm                2    255          ro
highTempAlarm             3    255          ro
low TempAlarm             4    255          ro
inletSensorAlarm          5    255          ro
outletSensorAlarm         6    255          ro
doorOrSmokeAlarm          7    255          ro
highPressureAlarm         8    255          ro
controllerCommFail        9    255          ro
readAlarm                10      1          ro
# ------------ Holding Registers ------------ #
address  value writeaccess
signal_name
coolSp                    1    255          rw
coolDif                   2    255          rw
heatSp                    3    255          rw
heatDif                   4    255          rw
hiTempAlarm               5    255          rw
loTempAlarm               6    255          rw
unitId1                   7    255          rw
unitId2                   8    255          rw
unitId3                   9    255          rw
unitId4                  10    255          rw
inletTemp                31    255          ro
outletTemp               32    255          ro
coolMin                  33    255          ro
coolMax                  34    255          ro
heatMin                  35    255          ro
```

```
heatMax              36    255          ro
commVers             37    255          ro
carelVers            38    255          ro
cbSerial1            39    255          ro
cbSerial2            40    255          ro
cbSerial3            41    255          ro
unitSerial1          42    255          ro
unitSerial2          43    255          ro
unitSerial3          44    255          ro
unitSerial4          45    255          ro
unitSerial5          46    255          ro
unitSerial6          47    255          ro
unitSerial7          48    255          ro
unitSerial8          49    255          ro
unitSerial9          50    255          ro
unitModel1           51    255          ro
unitModel2           52    255          ro
unitModel3           53    255          ro
unitModel4           54    255          ro
unitModel5           55    255          ro
unitModel6           56    255          ro
unitModel7           57    255          ro
unitModel8           58    255          ro
readAlarm            64      1          ro
readTimestamp0       65    255          ro
readTimestamp1       66    255          ro
readTimestamp2       67    255          ro
readTimestamp3       68    255          ro
heartbeat            69      1          ro
```

12.2.3.2 UPS (Telco Rack)

Test	Cube	CTR
Applicable		**X**

Use the following command to test the UPS that is located in the Telco Rack.

```
controllino_modbus.py 192.168.1.2 502 upsTelco
(use correct IP-Address and port as first and second argument)
```

Example Output

```
IMPORTANT: THIS IS ONLY AN EXAMPLE OUTPUT. NORMALLY THE VALUES SHOULD NOT BE 255.
VALUES OF 255 MEAN THE HVAC COULD NOT BE READ.


# ============ New Data ============ #

# ============ UPS (Telco) ============ #
# ------------ Holding Registers ------------ #
address  value writeaccess
signal_name
fwVersion                                4    255          ro
serialNumber0                         1025    255          ro
serialNumber1                         1026    255          ro
serialNumber2                         1027    255          ro
serialNumber3                         1028    255          ro
userDeviceName0                       4096    255          ro
userDeviceName1                       4097    255          ro
userDeviceName2                       4098    255          ro
userDeviceName3                       4099    255          ro
userDeviceName4                       4100    255          ro
userDeviceName5                       4101    255          ro
userDeviceName6                       4102    255          ro
userDeviceName7                       4103    255          ro
userDeviceName8                       4104    255          ro
userDeviceName9                       4105    255          ro
userDeviceName10                      4106    255          ro
userDeviceName11                      4107    255          ro
userDeviceName12                      4108    255          ro
userDeviceName13                      4109    255          ro
userDeviceName14                      4110    255          ro
userDeviceName15                      4111    255          ro
userSystemName0                       4128    255          ro
userSystemName1                       4129    255          ro
userSystemName2                       4130    255          ro
userSystemName3                       4131    255          ro
userSystemName4                       4132    255          ro
userSystemName5                       4133    255          ro
userSystemName6                       4134    255          ro
userSystemName7                       4135    255          ro
userSystemName8                       4136    255          ro
userSystemName9                       4137    255          ro
```

userSystemName10	4138	255	ro
userSystemName11	4139	255	ro
userSystemName12	4140	255	ro
userSystemName13	4141	255	ro
userSystemName14	4142	255	ro
userSystemName15	4143	255	ro
setParameters	4160	255	rw
setParameters_REG2	4161	255	rw
setSignalingCodeDO1	4162	255	rw
setSignalingCodeDO1_REG2	4163	255	rw
setSignalingCodeDO2	4164	255	rw
setSignalingCodeDO2_REG2	4165	255	rw
setSignalingCodeDO3	4166	255	rw
setSignalingCodeDO3_REG2	4167	255	rw
setFunctionCodeDI1	4170	255	rw
setFunctionCodeDI2	4171	255	rw
setChargeCurrentUPS	4173	255	rw
setChargeAbsorbtionVoltage	4174	255	rw
setChargeEndVoltage	4175	255	rw
setBatteryTempCoefficient	4176	255	rw
setDischargeBatteryEndvoltage	4177	255	rw
setSwitchingThresholdInpVoltageMin	4182	255	rw
setSwitchingThresholdInpVoltageMax	4183	255	rw
setBatmodeReturnToMainsTime	4184	255	rw
setCustomBuffertime	4185	255	rw
setBatmodeDelaytime1PcShutdown	4186	255	rw
setBatmodeDelayTime2	4187	255	rw
setBatmodeDelayTime3	4188	255	rw
setPcModeShutdownTime	4189	255	rw
setPcModeResetTime	4190	255	rw
setSignalingTimeAfterSwitchOff	4191	255	rw
setThresholdBufferReady	4193	255	rw
setBatmodeUsableCapacity	4194	255	rw
setUserInstalledPeripherie	4195	255	rw
SetBatUserInstalledCapacityNominal	4196	255	rw
SetTestIntervalBatConductance	4199	255	rw
SetBatAlarmUserReplaceTime	4200	255	rw
SetBatAlarmSOCVoltage	4201	255	rw
SetBatAlarmSOCPercent	4202	255	rw
SetBatAlarmSOCTime	4203	255	rw
SetBatAlarmSOHPercent	4204	255	rw
SetBatAlarmSOHTime	4205	255	rw
SetBatWarningSocVoltage	4206	255	rw
SetBatWarningSocPercent	4207	255	rw
SetBatWarningSocTime	4208	255	rw
SetBatWarningSohPercent	4209	255	rw
SetBatWarningSohTime	4210	255	rw
SetBatteryWarningDeltaTemperature	4211	255	rw
SetModeSelectorSwitch	4212	255	rw
SetEnableDisableFunction	4214	255	rw
SetEnableDisableFunction_REG2	4215	255	rw
SetBatteryInternalResistorMax	4216	255	rw
SetResistorBetweenUpsAndBattery	4217	255	rw

StatusFunctions	8192	255	ro
StatusFunctions_REG2	8193	255	ro
StatusInterface	8194	255	ro
StatusInterface_REG2	8195	255	ro
StatusActualInputVoltage	8196	255	ro
StatusActualInputCurrent	8197	255	ro
StatusActualOutputVoltage1	8198	255	ro
StatusActualOutputCurrent1	8199	255	ro
StatusBatteryActualVoltage	8202	255	ro
StatusBatteryChargeCurrent	8203	255	ro
StatusBatteryTemperature	8205	255	ro
StatusDeviceTemperature	8206	255	ro
StatusSoc	8207	255	ro
StatusSocRemainingTime	8208	255	ro
StatusSocRemaningTimePcsh	8210	255	ro
StatusSoh	8211	255	ro
StatusSohRemainingLifetime	8212	255	ro
StatusInstalledPeripherie	8213	255	ro
CountOperationTime	8216	255	ro
CountUserOperationTime	8218	255	rw
CountUserOperationTime_REG2	8219	255	rw
CountSystemStart	8220	255	ro
CountSystemStart_REG2	8221	255	ro
CountUserSystemStart	8222	255	rw
CountUserSystemStart_REG2	8223	255	rw
CountBatteryModeEvent	8224	255	ro
CountBatteryModeEvent_REG2	8225	255	ro
CountUserBatteryModeEvent	8226	255	rw
CountUserBatteryModeEvent_REG2	8227	255	rw
CountBatteryModeTime	8228	255	ro
CountBatteryModeTime_REG2	8229	255	ro
CountUserBatteryTime	8230	255	rw
CountUserBatteryTime_REG2	8231	255	rw
CountActualBatteryModeTime	8232	255	ro
CountActualBatteryModeTime_REG2	8233	255	ro
CountDischargeBatteryEndvoltage	8234	255	ro
CountDischargeBatteryEndvoltage_REG2	8235	255	ro
CountAlarmDeviceTemperature	8236	255	ro
CountAlarmDeviceTemperature_REG2	8237	255	ro
CountAlarmBatteryTemperature	8238	255	ro
CountAlarmBatteryTemperature_REG2	8239	255	ro
CountWarningBatteryTemperature	8240	255	ro
CountWarningBatteryTemperature_REG2	8241	255	ro
CountAlarmOverload	8242	255	ro
CountAlarmOverload_REG2	8243	255	ro
CountAlarmService	8244	255	ro
CountAlarmService_REG2	8245	255	ro
CountTimeAfterSohExpired	8246	255	ro
CountTimeAfterSohExpired_REG2	8247	255	ro
StatusAnalogInput	8248	255	ro
StatusBatteryInternalResistor	8249	255	ro
StatusActualAlarm	12288	255	ro
StatusActualAlarm_REG2	12289	255	ro

StatusAlarmMinus1	12290	255	ro
StatusAlarmMinus1_REG2	12291	255	ro
StatusAlarmMinus2	12292	255	ro
StatusAlarmMinus2_REG2	12293	255	ro
StatusAlarmMinus3	12294	255	ro
StatusAlarmMinus3_REG2	12295	255	ro
StatusAlarmMinus4	12296	255	ro
StatusAlarmMinus4_REG2	12297	255	ro
StatusAlarmMinus5	12298	255	ro
StatusAlarmMinus5_REG2	12299	255	ro
StatusAlarmMinus6	12300	255	ro
StatusAlarmMinus6_REG2	12301	255	ro
StatusAlarmMinus7	12302	255	ro
StatusAlarmMinus7_REG2	12303	255	ro
StatusAlarmMinus8	12304	255	ro
StatusAlarmMinus8_REG2	12305	255	ro
StatusActualWarning	12306	255	ro
StatusActualWarning_REG2	12307	255	ro
StatusWarningMinus1	12308	255	ro
StatusWarningMinus1_REG2	12309	255	ro
StatusWarningMinus2	12310	255	ro
StatusWarningMinus2_REG2	12311	255	ro
StatusWarningMinus3	12312	255	ro
StatusWarningMinus3_REG2	12313	255	ro
StatusWarningMinus4	12314	255	ro
StatusWarningMinus4_REG2	12315	255	ro
StatusWarningMinus5	12316	255	ro
StatusWarningMinus5_REG2	12317	255	ro
StatusWarningMinus6	12318	255	ro
StatusWarningMinus6_REG2	12319	255	ro
StatusWarningMinus7	12320	255	ro
StatusWarningMinus7_REG2	12321	255	ro
StatusWarningMinus8	12322	255	ro
StatusWarningMinus8_REG2	12323	255	ro
LogActualInputVoltage	12324	255	ro
LogActualInputCurrent_REG2	12325	255	ro
LogActualOutputVoltage1	12326	255	ro
LogActualOutputCurrent	12327	255	ro
LogActualBatteryVoltage	12330	255	ro
LogActualBatteryChargeCurrent	12331	255	ro
LogActualBatteryTemperature	12333	255	ro
LogStatusSoc	12334	255	ro
LogStatusSoh	12335	255	ro
LogCountOperationTime	12336	255	ro
LogCountOperationTime_REG2	12337	255	ro
Battery1FwVersion	16898	255	ro
Battery1SerialnumberLSB	16899	255	ro
Battery1Serialnumber1	16900	255	ro
Battery1Serialnumber2	16901	255	ro
Battery1SerialnumberMSB	16902	255	ro
Battery1BatteryType	16903	255	ro
Battery1InstalledCapacity	16913	255	ro
Battery1BatteryResistorNominal	16914	255	ro

Battery1TemperatureAlarmMax	16917	255	ro
Battery1TemperatureAlarmMin	16918	255	ro
Battery1ChargeCharacteristicType	16919	255	ro
Battery1LifetimeNominal	16920	255	ro
Battery1ChargeCurrentMax	16921	255	ro
Battery1ChargeAbsorbtionVoltage	16922	255	ro
Battery1ChargeEndvoltage	16923	255	ro
Battery1ChargeTemperatureCoefficient	16924	255	ro
Battery1DischargeEndvoltage	16925	255	ro
Battery1DischargeCurrentMax	16926	255	ro
Battery1TemperatureWarningMax	17024	255	ro
Battery1TemperatureWarningMin	17025	255	ro
Battery1DischargeEndvoltageLowCurrent	17029	255	ro
Battery1StatusSOC	17031	255	ro
Battery1StatusSOH	17032	255	ro
Battery1StatusActualTemperature	17059	255	ro
Battery1StatusFuse	17061	255	ro
Battery1StatusActualInternalVoltage	17063	255	ro
Battery1StatusActualBlockVoltage	17064	255	ro
Battery2FwVersion	17154	255	ro
Battery2SerialnumberLSB	17155	255	ro
Battery2Serialnumber1	17156	255	ro
Battery2Serialnumber2	17157	255	ro
Battery2SerialnumberMSB	17158	255	ro
Battery2BatteryType	17159	255	ro
Battery2InstalledCapacity	17169	255	ro
Battery2BatteryResistorNominal	17170	255	ro
Battery2TemperatureAlarmMax	17173	255	ro
Battery2TemperatureAlarmMin	17174	255	ro
Battery2ChargeCharacteristicType	17175	255	ro
Battery2LifetimeNominal	17176	255	ro
Battery2ChargeCurrentMax	17177	255	ro
Battery2ChargeAbsorbtionVoltage	17178	255	ro
Battery2ChargeEndvoltage	17179	255	ro
Battery2ChargeTemperatureCoefficient	17180	255	ro
Battery2DischargeEndvoltage	17181	255	ro
Battery2DischargeCurrentMax	17182	255	ro
Battery2TemperatureWarningMax	17280	255	ro
Battery2TemperatureWarningMin	17281	255	ro
Battery2DischargeEndvoltageLowCurrent	17285	255	ro
Battery2StatusSOC	17287	255	ro
Battery2StatusSOH	17288	255	ro
Battery2StatusActualTemperature	17315	255	ro
Battery2StatusFuse	17317	255	ro
Battery2StatusActualInternalVoltage	17319	255	ro
Battery2StatusActualBlockVoltage	17320	255	ro
Battery3FwVersion	17410	255	ro
Battery3SerialnumberLSB	17411	255	ro
Battery3Serialnumber1	17412	255	ro
Battery3Serialnumber2	17413	255	ro
Battery3SerialnumberMSB	17414	255	ro
Battery3BatteryType	17415	255	ro
Battery3InstalledCapacity	17425	255	ro

Battery3BatteryResistorNominal	17426	255	ro
Battery3TemperatureAlarmMax	17429	255	ro
Battery3TemperatureAlarmMin	17430	255	ro
Battery3ChargeCharacteristicType	17431	255	ro
Battery3LifetimeNominal	17432	255	ro
Battery3ChargeCurrentMax	17433	255	ro
Battery3ChargeAbsorbtionVoltage	17434	255	ro
Battery3ChargeEndvoltage	17435	255	ro
Battery3ChargeTemperatureCoefficient	17436	255	ro
Battery3DischargeEndvoltage	17437	255	ro
Battery3DischargeCurrentMax	17438	255	ro
Battery3TemperatureWarningMax	17536	255	ro
Battery3TemperatureWarningMin	17537	255	ro
Battery3DischargeEndvoltageLowCurrent	17541	255	ro
Battery3StatusSOC	17543	255	ro
Battery3StatusSOH	17544	255	ro
Battery3StatusActualTemperature	17571	255	ro
Battery3StatusFuse	17573	255	ro
Battery3StatusActualInternalVoltage	17575	255	ro
Battery3StatusActualBlockVoltage	17576	255	ro
Battery4FwVersion	17666	255	ro
Battery4SerialnumberLSB	17667	255	ro
Battery4Serialnumber1	17668	255	ro
Battery4Serialnumber2	17669	255	ro
Battery4SerialnumberMSB	17670	255	ro
Battery4BatteryType	17671	255	ro
Battery4InstalledCapacity	17681	255	ro
Battery4BatteryResistorNominal	17682	255	ro
Battery4TemperatureAlarmMax	17685	255	ro
Battery4TemperatureAlarmMin	17686	255	ro
Battery4ChargeCharacteristicType	17687	255	ro
Battery4LifetimeNominal	17688	255	ro
Battery4ChargeCurrentMax	17689	255	ro
Battery4ChargeAbsorbtionVoltage	17690	255	ro
Battery4ChargeEndvoltage	17691	255	ro
Battery4ChargeTemperatureCoefficient	17692	255	ro
Battery4DischargeEndvoltage	17693	255	ro
Battery4DischargeCurrentMax	17694	255	ro
Battery4TemperatureWarningMax	17792	255	ro
Battery4TemperatureWarningMin	17793	255	ro
Battery4DischargeEndvoltageLowCurrent	17797	255	ro
Battery4StatusSOC	17799	255	ro
Battery4StatusSOH	17800	255	ro
Battery4StatusActualTemperature	17827	255	ro
Battery4StatusFuse	17829	255	ro
Battery4StatusActualInternalVoltage	17831	255	ro
Battery4StatusActualBlockVoltage	17832	255	ro
Battery5FwVersion	17922	255	ro
Battery5SerialnumberLSB	17923	255	ro
Battery5Serialnumber1	17924	255	ro
Battery5Serialnumber2	17925	255	ro
Battery5SerialnumberMSB	17926	255	ro
Battery5BatteryType	17927	255	ro

```
Battery5InstalledCapacity                17937    255          ro
Battery5BatteryResistorNominal           17938    255          ro
Battery5TemperatureAlarmMax              17941    255          ro
Battery5TemperatureAlarmMin              17942    255          ro
Battery5ChargeCharacteristicType         17943    255          ro
Battery5LifetimeNominal                  17944    255          ro
Battery5ChargeCurrentMax                 17945    255          ro
Battery5ChargeAbsorbtionVoltage          17946    255          ro
Battery5ChargeEndvoltage                 17947    255          ro
Battery5ChargeTemperatureCoefficient     17948    255          ro
Battery5DischargeEndvoltage              17949    255          ro
Battery5DischargeCurrentMax              17950    255          ro
Battery5TemperatureWarningMax            18048    255          ro
Battery5TemperatureWarningMin            18049    255          ro
Battery5DischargeEndvoltageLowCurrent    18053    255          ro
Battery5StatusSOC                        18055    255          ro
Battery5StatusSOH                        18056    255          ro
Battery5StatusActualTemperature          18083    255          ro
Battery5StatusFuse                       18085    255          ro
Battery5StatusActualInternalVoltage      18087    255          ro
Battery5StatusActualBlockVoltage         18088    255          ro
readAlarm                                18176      1          ro
readTimestamp0                           18177    255          ro
readTimestamp1                           18178    255          ro
readTimestamp2                           18179    255          ro
readTimestamp3                           18180    255          ro
heartbeat                                18181      1          ro
```

12.3 Finish Testing

Test	Cube	CTR
Applicable	**X**	**X**

After finishing the tests successfully enable the modbus-service again. This can be done with the following command:

```
systemctl start fluence-modbus
```

The current status of the command could be verified with:

```
systemctl status fluence-modbus
```

13. Installing Banana Pi R2

Test	Cube	CTR
Applicable	**X**	**X**

13.1 Booting from eMMC

Test	Cube	CTR
Applicable	**X**	**X**

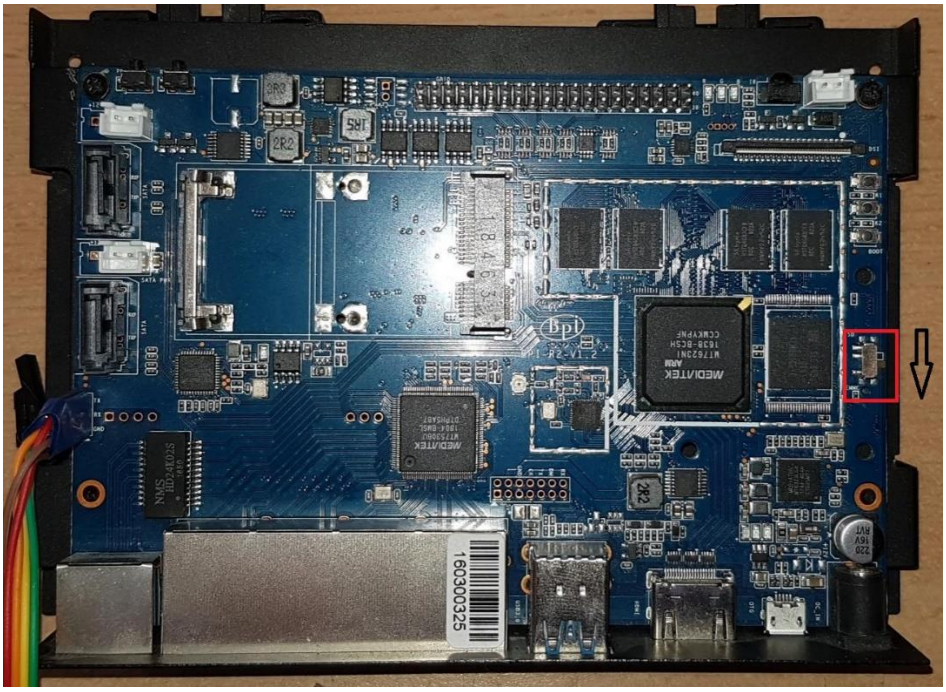
Booting from an SD-card is useful for testing purposes but makes the system vulnerable. Therefore, the BananaPi should be set to boot from the eMMC. The following chapter describes the necessary steps.

Important: The following commands can be used to reboot or shutdown the Banana Pi:

```
reboot
shutdown -h now
```

The following steps must be executed on a host computer:

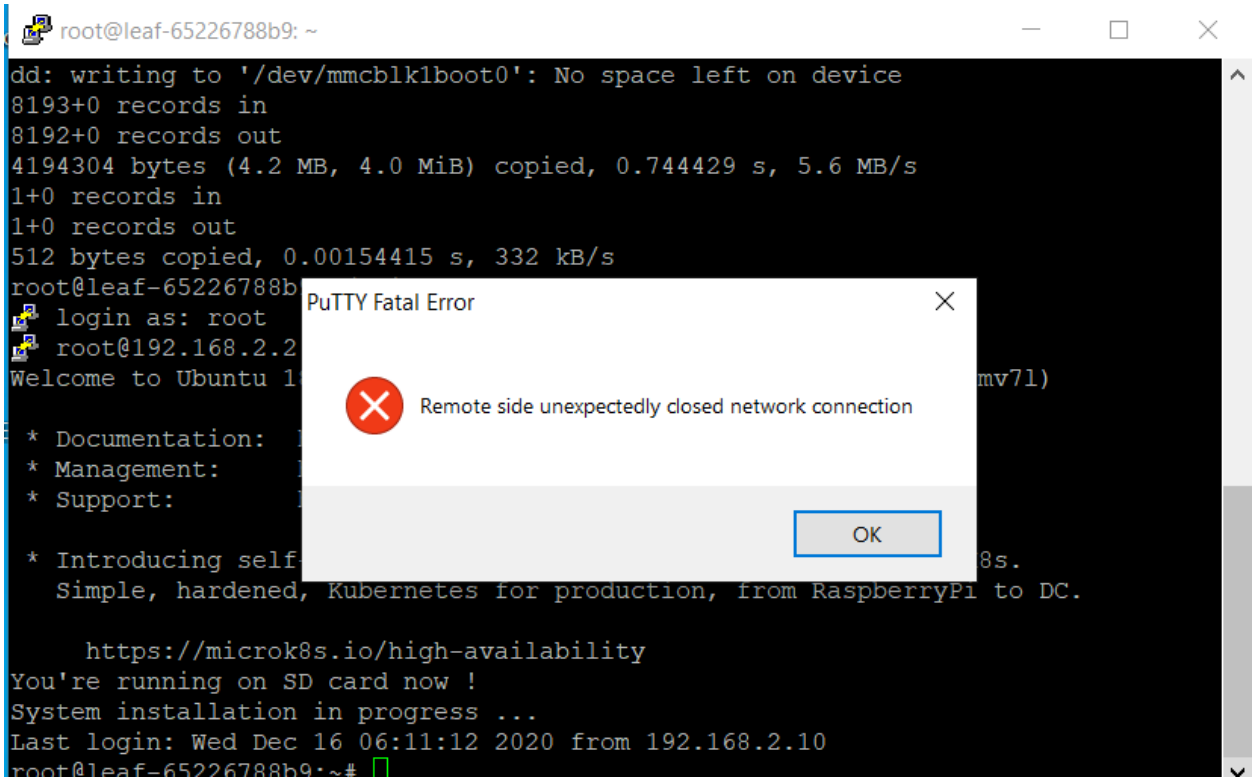
1. Download the latest master image from Sharepoint. The image is named: fluence_<timestamp>_master.img.gz https://fluenceenergy.sharepoint.com/:f:/r/sites/nextgen/Shared%20Documents/Controls%20HW%20and%20SW/NextGen%20Controller%20Workstream/Device%20Images?csf=1&web=1&e=96fH4R
2. Use a Software (e.g. BalenaEtcher) to flash the image to an SD-Card.
3. Make sure the Banana Pi is unplugged. Now, move the eMMC-switch to the eMMC-Position. This switch is located on the Banana Pi next to the SD-Card port (See Picture below).



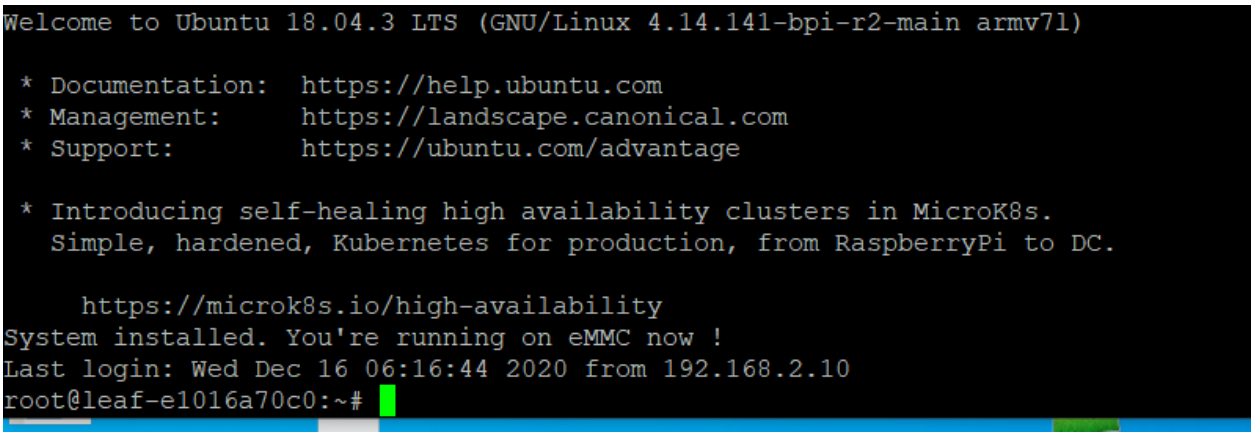
4. Insert the SD Card into the Banana Pi and start it. On the first boot the system will be transferred into eMMC. Afterwards, the Banana Pi will reboot. If logging in to the Banana Pi before the System reboots automatically, the last sentence of the header will be “You are running on SD-Card now System installation in progress”.

```
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.14.141-bpi-r2-main armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.

     https://microk8s.io/high-availability
You're running on SD card now !
System installation in progress ...
Last login: Wed Dec 16 06:11:12 2020 from 192.168.2.10
root@leaf-65226788b9:~#
```

You will be disconnected when the Banana Pi automatically reboots.



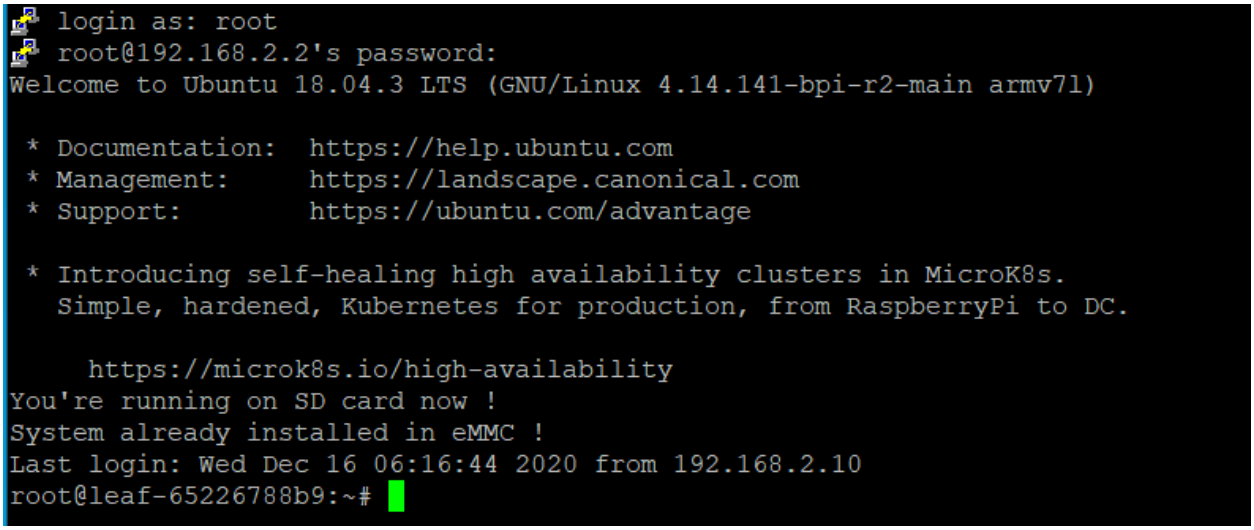
If you log in again, the last sentence of the header should be “System installed. You´re running on eMMC now”.



The little switch should force a boot from the eMMC, even if an SD-Card is plugged in. To test this, leave the SD-Card in the Banana Pi and restart it again. The last sentence of the header should be “System installed. You´re running on eMMC now !”.

Test if the booting also works without the SD-Card. For that, shut down the Banana Pi, unplug it, remove the SD-Card and start it again. You should be able to log in with putty and read the header below.

If the little switch is **not** in the position “boot from the eMMC” you will boot from CD card and message will be as in the picture below.



Removing of SD card will boot the system from eMMC.

13.2 Force next boot from SD card

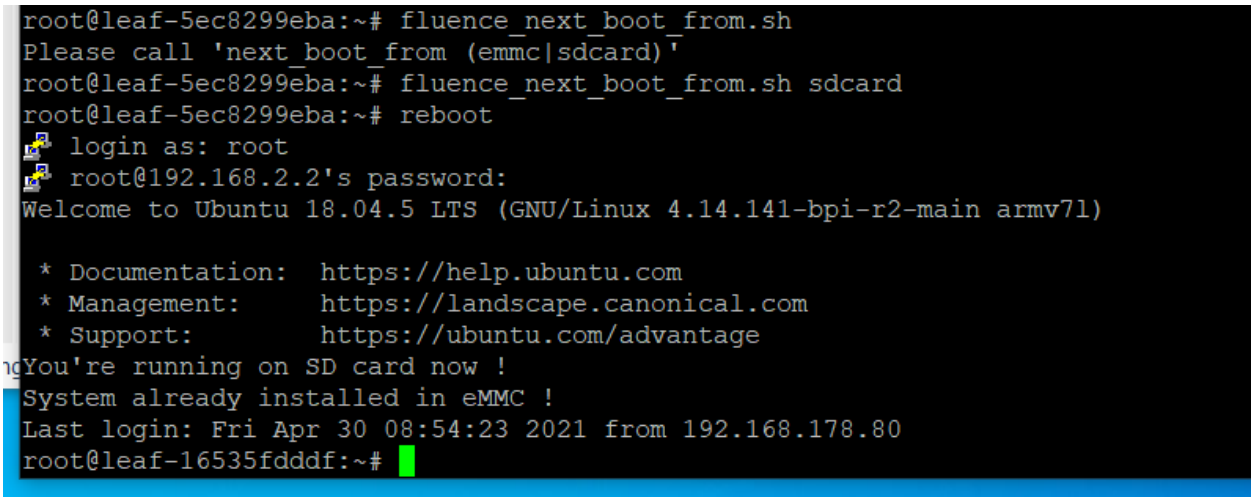
Test	Cube	CTR
Applicable	**X**	**X**

If we would like to boot the system from SD Card instead eMMC, we have two possibilities:

- Move internal switch into SD position (upper).
- Use internal program to force booting from SD card.

```
fluence_next_boot_from.sh sdcard
```

Next reboot the Banana PI will start form SD card it the Master SD card is inserted into the slot.

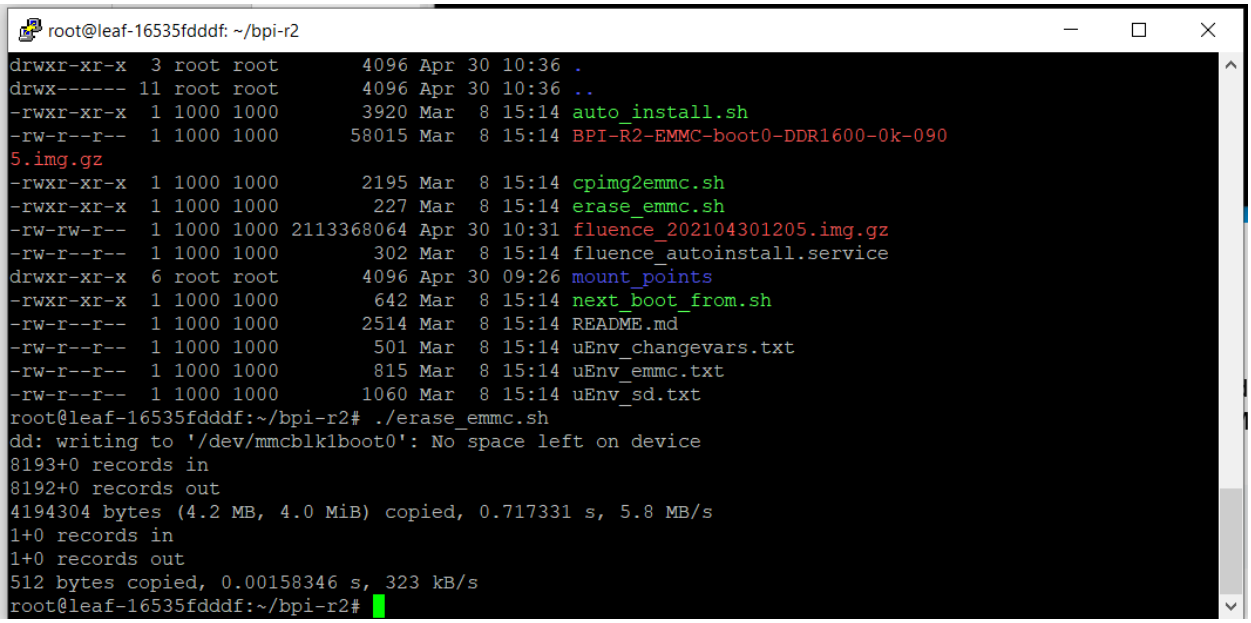


13.3 Erase eMMC

Test	Cube	CTR
Applicable	**X**	**X**

If the Banana Pi is booted from Master SD card and the Master SD card is detect already installed system on eMMC, *the automatic install procedure will not be triggered*. eMMC is possible to erase with the help of the small utility.

```
cd
cd bpi-r2
./erase_emmc.sh
```



Reboot Banana Pi and the automatic installation will be triggered.

13.4 Something is wrong, no start from eMMC

Test	Cube	CTR
Applicable	**X**	**X**

If the switch is correct position (eMMC) and Banana Pi is not starts, move the switch to the SD position (upper position) and start the controller once more from Master SD card. Small utility located under /root/bpi-r2/ will help to erase eMMC memory.

```
cd
cd bpi-r2
./erase_emmc.sh
```

Reboot the controller and the installation process will be started once more.

14. Installing Modberry 500 M3 (Clean installation)

Test	Cube	CTR
Applicable		**X**

The next steps are procedure to install clean OS on Modberrry 500 M3. The image will not conssist Fluence software.

You will need following:

1) RPIBOOT Setup program (rpiboot_setup.exe)

2) Balena Etcher (https://www.balena.io/etcher/)

3) USB to Micro USB cable

4) Modbery PI 500 image (CM3_4.19_06.05.2020_cm3+.img)

Steps:

1) Install RPI Boot on your computer

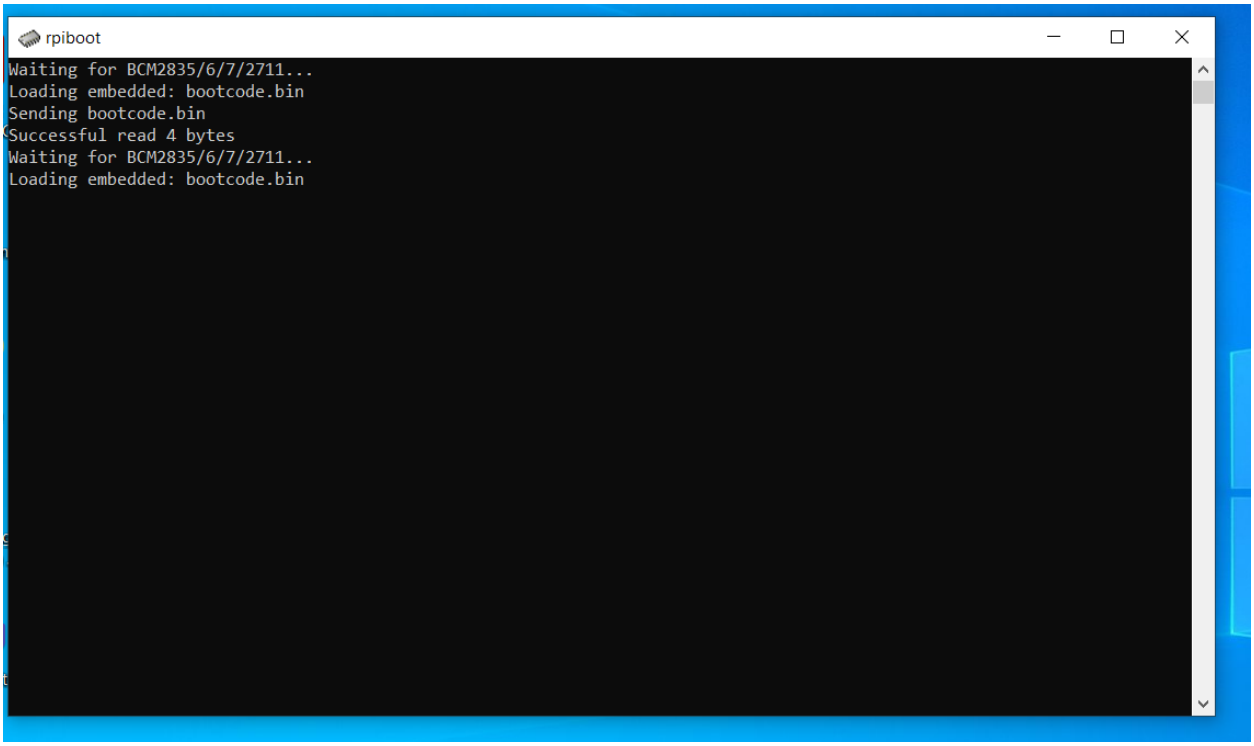
2) Install Balena Etcher

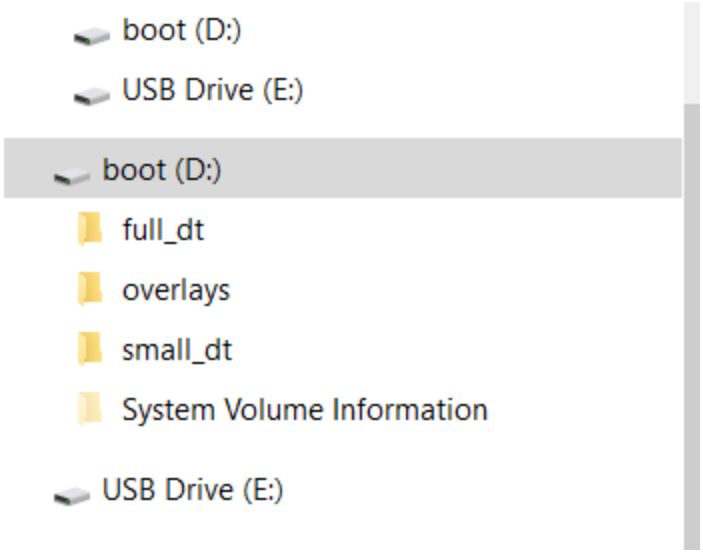
3) Power Off Modberry 500 and disconnect all network and USB Cables

4) Connect Modberry with your Laptop, use micro USB port on Modberry!!!!!

5) Start RPIBoot program

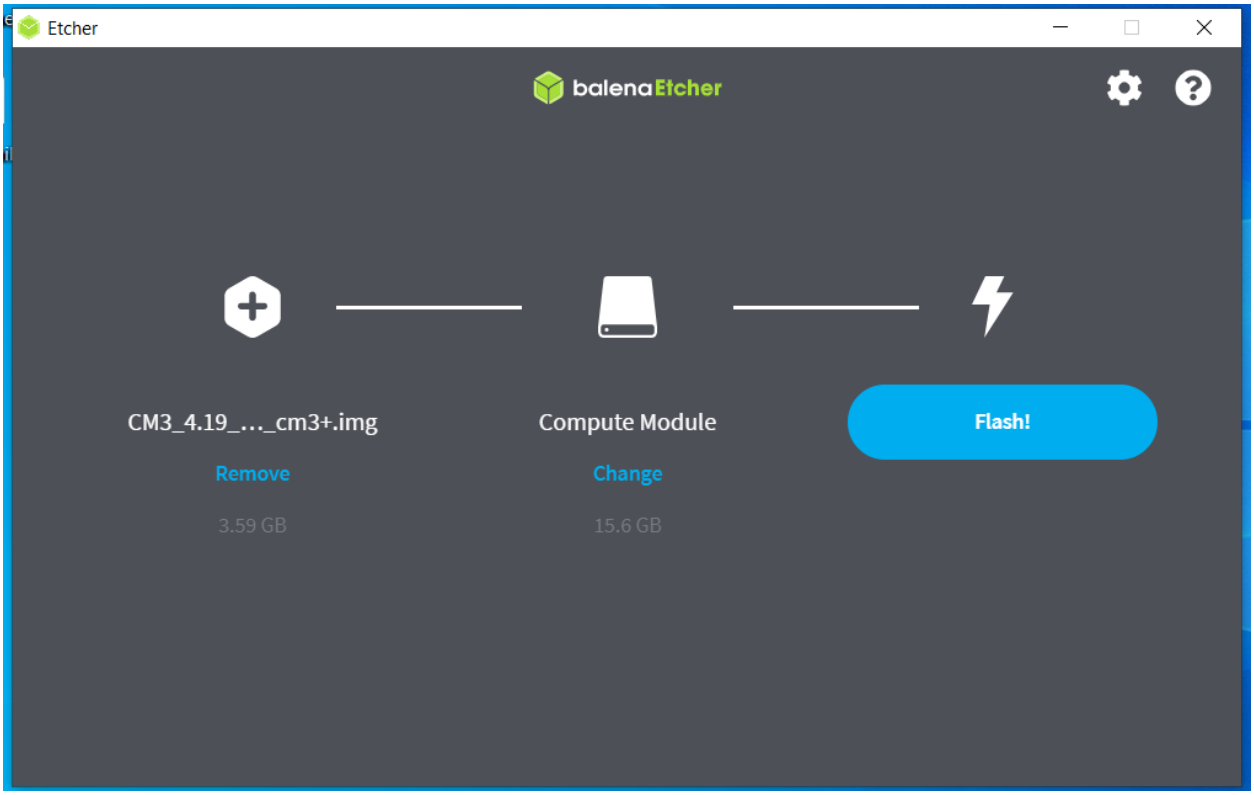
6) Power Modberry 500, it must be recognized as external drive in your computer





Two new disks: one is recognised as MSDOS disk, for the second Windows offers to format it. **Do not format the disk!!**

7) with help of Balena Etcher install image on Modberry



8) Power Off ModBerry and disconnect USB cable

9) Connect ModBerry network and Controllino back

10) Power on ModBerry

11) Login and execute script. The script will expand file system to the maximum size.

```
raspi-config --expand-rootfs
```

12) Reboot ModBerry to use new expanded file system

ModBerry default MAC address is `00:00:00:00:01`. If you need more ModBerries on the network, please assign the unique MAC address to each of them. Read the MAC address from the sticker, change the example MAC with the one from the sticker.

```
cd /root/scripts
./set_mac_in_uboot.sh 18:83:C4:04:51:BC
```

and

```
./set_mac.sh 18:83:C4:04:51:BC
```

And reboot. This will assign different MAC to each fresh installed ModBerry.

It is strait forward method, I done it many times. Flashing process need cca 13 minutes on my test computer.

15. Installing or Updating FOS for Controllers

Test	Cube	CTR
Applicable	**X**	**X**

You will need account on array server if the installation/updating is performing on Gen6 site or any another computer in the same network whare is the target controller.

Two different option are prapared for the installation:

- Direct from the last sources (GitHub)
- With help of installation archive

15.1 Prepare installation process from GitHub

The following steps are required to successfully install the project if you are member of the https://github.com/FluenceEnergy/controller_software

15.1.1 Prerequired condition

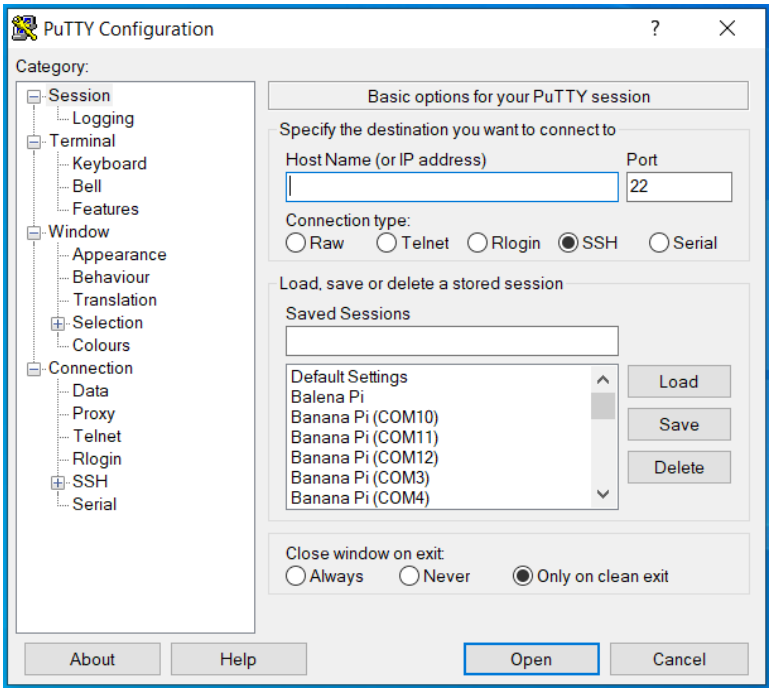
- To have GitHub account
- To have access to https://github.com/FluenceEnergy/controller_software
- Local Linux or Windows computer
- Windows Computer need installed Putty
- VPN access to “Fluence Support VPN”

15.1.2 Steps

- Ask Shakti Mori to create the account for you on the array controller. Each project has one array controller.

- Logon on array controller and change password (command passwd)
  Example Linux:
  ```
  ssh bkajganic.admin@10.194.141.3
  ```

  Example Windows:



- (Linux only) Generate public key on **local** computer **only** if is not exist

```
ssh-keygen -t ed25519
```

- (Linux only) Copy public key from **local** computer to **array** server

```
ssh-copy-id -i ~/.ssh/id_ed25519.pub user@array_controller
```

- (Linux only) Test ssh without passwd. Coping the public key saves typing of the password each time.

- Generate public key on **array** controller **only** if is not exist

```
ssh-keygen -t ed25519
```

- Goto your GitHub User Settings (https://github.com/settings/keys or GitHub account -> User -> Settings -> SSH and GPG keys), create new ssh key and upload ed25519.pub from array controller

- Test ssh from array controller to GitHub

```
ssh -vT git@github.com
```

- If is not working you will need to setup ssh over https as is described here

- Create local copy of the project

```
cd ~
mkdir bpi-r2
cd bpi-r2
git clone git@github.com:FluenceEnergy/controller_software.git
```

- Add user and email address

```
cd controller_software
git config --local user.email "name.familyname@flueneceenergy.com"
git config --local user.name "GitUserName"
```

- Get update from repository

```
git pull origin
```

- Fetch remote branch (first argument) `Gen6` as local branch `Gen6` (second argument) and set tracking information for this branch

```
git fetch origin Gen6:Gen6 && git checkout Gen6
git branch --set-upstream-to=origin/Gen6 Gen6
```

Gen6 is liquid cooling long duration Cubes by default (LD cubes). If you need SD cubes replace `Gen6` with `RaP` branch in the previous step.

15.1.3 Prepare installation of the controllers

Next steps will create local deploy scripts and data required for the further installation. The data will be copied to each controller and execute on the controller.

```
cd ~
cd bpi-r2/controller_software/LeafController/Deploy/
./clean.sh
./Modberry500M3_deploy.sh fos copy pack
./BananaPiR2_deploy.sh fos copy pack
```

If the project is based on IEC version of the OCTE the step `./Modberry500M3_deploy.sh fos copy pack` could be avoided. In IEC project all controllers are Banana Pi R2.

15.1.4 Prepare archive file

If you need to prepare archive file with all need scripts and data for your colleague, the following steps are required.

```
cd ~
cd bpi-r2/controller_software/LeafController/Deploy/
./clean.sh
./Modberry500M3_deploy.sh fos copy pack
./BananaPiR2_deploy.sh fos copy pack
./packall.sh
```

If the project is based on IEC version of the OCTE the step `./Modberry500M3_deploy.sh fos copy pack` could be avoided. In IEC project all controllers are Banana Pi R2.

The result archive file is `ControllerAllData.tgz`. Copy or email to your colleague.

15.2 Prepare installation process from archive file

If you don't have access to the GitHub project described in the previous chapter, somebody need to prepare archive file (`ControllerAllData.tgz`) where all require data are packed.

15.2.1 Prerequired condition

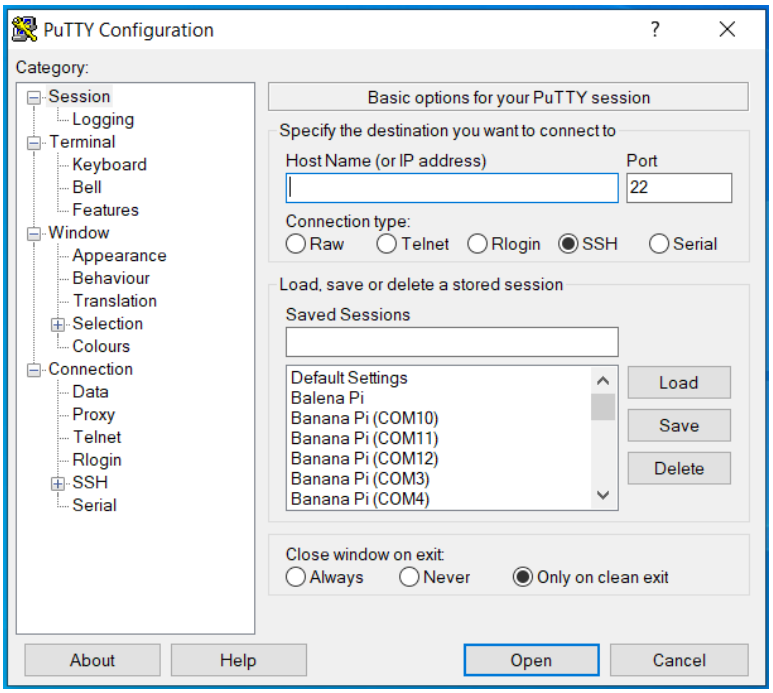
- To have prepared archive file `ControllerAllData.tgz`
- Local Linux or Windows computer
- Windows Computer need installed Putty
- VPN access to "Fluence Support VPN"

15.2.2 Steps

- Ask Shakti Mori to create the account for you on the array controller. Each project has one array controller.

- Logon on array controller and change password (command passwd)
  Example Linux:
  ```
  ssh bkajganic.admin@10.194.141.3
  ```

  Example Windows:



- (Linux only) Generate public key on **local** computer **only** if is not exist

```
ssh-keygen -t ed25519
```

- (Linux only) Copy public key from **local** computer to **array** server

```
ssh-copy-id -i ~/.ssh/id_ed25519.pub user@array_controller
```

- (Linux only) Test ssh without passwd. Coping the public key saves typing of the password each time.

- Copy and unpack the installation file

```
cd ~
mkdir installation
cd installation
copy or upload ControllerAllData.tgz to the ~/insallation
tar -xzvf ControllerAllData.tgz
```

15.3 Installation process

Prerequired software:

- sshpass
- fping

Check if the software is installed on the array controller.

```
ls_power_diablo enst89m01c00n00s01 Deploy : sshpass -v
Usage: sshpass [-f|-d|-p|-e] [-hV] command parameters
   -f filename   Take password to use from file
   -d number     Use number as file descriptor for getting password
   -p password   Provide password as argument (security unwise)
   -e            Password is passed as env-var "SSHPASS"
   With no parameters - password will be taken from stdin

   -P prompt     Which string should sshpass search for to detect a password prompt
   -v            Be verbose about what you're doing
   -h            Show help (this screen)
   -V            Print version information
At most one of -f, -d, -p or -e should be used

ls_power_diablo enst89m01c00n00s01 Deploy : fping -v
fping: Version 4.0
fping: comments to david@schweikert.ch
```

If `sshpass` and/or `fping` is not installed, you can install the packages:

```
sudo apt-get install sshpass
sudo apt-get install fping
```

If you are not part of sudo group, you will need ask somebody to install the packages for you.

Supported hardware:

- Banana Pi R2
- Modberry 500 M3

Single Board Computer	Script Name
Banana Pi R2	BananaPiR2_deploy.sh
Modberry 500 M3	Modberry500M3_deploy.sh

Both scripts have the same functionality.

```
boris@ubuntu-dell$ ./BananaPiR2_deploy.sh
Usage:
  BananaPiR2_deploy.sh clear
    Remove BananaPiR2_transfer and BananaPiR2_data.tgz
  BananaPiR2_deploy.sh copy
    Collect necessary data in transfer folder
  BananaPiR2_deploy.sh pack
    pack transfer folder
  BananaPiR2_deploy.sh cpack
    copy and pack transfer folder
  BananaPiR2_deploy.sh fos
    create BananaPiR2_fos.txt version file
  BananaPiR2_deploy.sh remote_IP rootpwd copy pack
  BananaPiR2_deploy.sh remote_IP rootpwd
  BananaPiR2_deploy.sh file_with_remote_IPs rootpwd
    copy packed data and execute deploy_script.sh on remote computer

always 'copy' before 'pack'
```

Example:

```
./BananaPiR2_deploy.sh 192.168.178.30 root
./Modberry500M3_deploy.sh 192.168.178.31 techbase
./Modberry500M3_deploy.sh Core1octe.txt techbase
```

First argument is Controller IP address or the file with IP addresses and the second is root password.

Example of the file with IP addresses:

```
ls_power_diablo enst89m01c00n00s01 Deploy : less Core1octe.txt
10.11.8.57
10.11.8.58
10.11.8.59
10.11.8.60
10.11.8.61
10.11.8.62
```

15.3.1 How to detect dynamic IP address range?

Each project has associate Excel list where the IP addresses are assigned to each controller. This is the end status of the installation and will be valid after the controller is installer and self-configuration process find form the network topology whare the controllers belong. In the first moment the controller will get only dynamic address from DHCP server what is the array controller in each project.

As Example:

f011-C02 (Core 02)	01	24	F011-C02	10.11.16.3
f011-C02-CTR-FMC (Core 02)	01	23	F011-C02	10.11.16.161
f011-C02-N01-PCS01 (Core 02)	01	09	F011-C02	10.11.16.151
f011-C02-UPS (Core 02)	01	13	F011-C02	10.11.16.171
f011-C02-HVAC (Core 02)	01	18	F011-C02	10.11.16.181
f011-C02-N01 (Core 02)	01	14	F011-C02	10.11.17.3
f011-C02-N01-BMS01 (Core 02)	01	19	F011-C02	10.11.17.121
f011-C02-N01-R01-CU01 (Core 02)	01	01	F011-C02	10.11.17.101
f011-C02-N01-R01-CU02 (Core 02)			F011-C02	10.11.17.102
f011-C02-N01-R01-CU03 (Core 02)			F011-C02	10.11.17.103
f011-C02-N01-R01-CU04 (Core 02)			F011-C02	10.11.17.104
f011-C02-N01-R01-CU05 (Core 02)			F011-C02	10.11.17.105
f011-C02-N01-R01-CU06 (Core 02)			F011-C02	10.11.17.106
f011-C02-N01-R01-CU07 (Core 02)			F011-C02	10.11.17.107
f011-C02-N01-R01-CU08 (Core 02)	01	02	F011-C02	10.11.17.108

For dynamic IP range of each core, we need to know only Core controller IP address as the starting point. IN the example the Core IP address is 10.11.16.3. The dhcp range for all connected devices in the Core 2 is for 10.11.16.10 to 10.11.16.120. The first 3 octets must be the same.

Find connected devices to the Core 1:

```
ls_power_diablo enst89m01c00n00s01 Deploy : fping -ag 10.11.8.10 10.11.8.120
10.11.8.18
10.11.8.19
10.11.8.21
10.11.8.24
10.11.8.28
10.11.8.30
10.11.8.31
10.11.8.34
10.11.8.38
10.11.8.39
10.11.8.40
10.11.8.41
10.11.8.42
10.11.8.43
10.11.8.57
10.11.8.58
10.11.8.59
10.11.8.60
10.11.8.61
10.11.8.62
10.11.8.64
10.11.8.67
10.11.8.68
10.11.8.69
10.11.8.70
10.11.8.71
10.11.8.72
```

Response will be from MBMUs, Banana PIs and Modberries. The IEC version of OCTE is based of Banana Pi what make work al little bit easier.

Execute the command once more

```
fping -ag 10.11.8.10 10.11.8.120 >Core1.txt
```

The output will be written into the file Core1.txt

The file could be used with next command `testconnection.sh`. With help of the script, we can test connection to one or list of the controllers

Example from Luna project:

```
spower_luna enst95m01c00n00s01 Deploy : ./testconnection.sh 10.23.8.10 root
10.23.8.10  Connection OK
```

```
spower_luna enst95m01c00n00s01 Deploy : ./testconnection.sh Core28.txt root
10.23.8.10  Connection OK
10.23.8.11  Connection OK
10.23.8.15  Connection OK
```

```
10.23.8.17  Connection OK
10.23.8.18  Connection OK
10.23.8.20  Connection OK
10.23.8.21  Connection OK
10.23.8.22  Connection OK
10.23.8.23  Connection OK
10.23.8.27  Connection OK
10.23.8.28  Connection OK
10.23.8.29  Connection OK
10.23.8.30  Connection OK
10.23.8.31  Connection OK
10.23.8.32  Connection OK
10.23.8.33  Connection OK
10.23.8.34  Connection OK
10.23.8.35  Connection OK
Permission denied, please try again.
10.23.8.48
Permission denied, please try again.
10.23.8.49
Permission denied, please try again.
10.23.8.51
Permission denied, please try again.
10.23.8.52
Permission denied, please try again.
10.23.8.56
```

If the response is `Permission denied, please try again.` , the controller is ModBerry because we try to use “root” as the password. If the response is `Connection refused`, the device is MBMU.

My workflow is to remove MBMU from the file and additionally split the file into 2 additional files.

Core1.txt consists of all controllers.

Core1octe.txt consists only the controller inside OCTE (Core, Node and Telco Rack controller) Max 6 controllers.

Core1cubes.txt consists of all cube controllers.

It is recommended to start the installation with OCTE controllers.

```
./Modberry500M3_deploy.sh IPaddress techbase
```

Or for IEC version:

```
./BananaPiR2_deploy.sh IPaddress root
```

Modberry500M3_deploy.sh will execute remote script with

```
nohup bash ~/deploy_script.sh ${CONTROLLER} > ~/deploy_TIMESTAMP.txt
```

What leads to immediately return to the user console. The installation process is not finished and could take up to 10 minutes.

On the remote computer in folder /root, the workflow could be monitor

```
tail -f ~/deploy_TIMESTAMP.txt
```

Hint: After some time, tail will stop working because, the network configuration and **dynamic IP is changed**. Wait until the computer is back on the network. It could be checked with help of fping. The controller is installed if the name is like **leaf-5ec8299eba**.

Banana Pi are already preinstalled and using of BananaPiR2_deploy.sh will update the software on the controller. The installation process does not require `nohup` and the output of the installation is visible to the user.

At the end, independent if the output is in the file or on the console it could be two cases.

```
****** Installation finished ******
```

```
****** System is NOT correctly installed. Restart the installation process. ******
```

Hint: The dynamic IP address assigned to the controller will be the same after **two** reboots of the controller. This is consequence of changing the controller MAC address.

Script	Description
BananaPiR2_deploy.sh	Main script to deploy Banana Pi software
Modberry500M3_deploy.sh	Main script to deploy Modberry 500 M3 software
controllerreboot.sh	Reboot remote controller Example: boris@ubuntu-dell: ./controllerreboot.sh Target IP address or file with remote IPs is missing Usage: controllerreboot.sh remote_IP rootpwd controllerreboot.sh file_with_remote_IPs rootpwd Reboot remote computer
controllertype.sh	The script returns the type of the remote controller depends on the port where is the controller connect to the OCTE switch Example: boris@ubuntu-dell: $ ./controllertype.sh -h Usage: controllertype.sh Get controller type on local computer: Node Controller | Core Controller | Cube Controller | TR Controller | Unknown Controller controllertype.sh remote_IP rootpwd

	controllertype.sh file_with_remote_IPs rootpwd Get controller type on remote computer: Node Controller | Core Controller | Cube Controller | TR Controller | Unknown Controller controllertype.sh help|--help|-h print help
controllerinfo.sh	The script returns the controller info. It is extended version of controllertype.sh script Example: boris@ubuntu-dell: ./controllerinfo.sh -h Usage: controllerinfo.sh Get controller info on local computer: Node Controller | Core Controller | Cube Controller | TR Controller | Unknown Controller controllerinfo.sh remote_IP rootpwd controllerinfo file_with_remote_IPs rootpwd Get controller info on remote computer: Node Controller | Core Controller | Cube Controller | TR Controller | Unknown Controller controllerinfo.sh help|--help|-h print help
controllerfos.sh	The script returns the controller fos version. FOS will be written only if the controller is correctly installed Example: boris@ubuntu-dell: ./controllerfos.sh -h Usage: controllerfos.sh Get controller fos version on local computer controllerfos.sh remote_IP rootpwd controllerfos.sh file_with_remote_IPs rootpwd Get controller fos version on remote computer controllerfos.sh help|--help|-h print help
testconnection.sh	Test connection to the remote controller.
setdate.sh	Set data & time on remote controller.
setip.sh	Force reapplies network setting on remote controller
deploy_script.sh	Internally used from BananaPiR2_deploy.sh or Modberry500M3_deploy.sh
utils_deploy.sh	Internally used from BananaPiR2_deploy.sh or Modberry500M3_deploy.sh

If you logon the controller and the installation is started and not finished, you will see the message.

```
ls_power_diablo enst89m01c00n00s01 Deploy : ssh root@10.11.32.62
root@10.11.32.62's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.14.141-bpi-r2-main armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
System installation in progress ...  NO Reboot!!!!
```

```
Last login: Fri Apr 30 09:26:42 2021 from 192.168.2.10
root@leaf-e5eaff97e2:~#
```

If the controller is correctly installed the message will be:

```
ls_power_diablo enst89m01c00n00s01 Deploy : ssh root@10.11.32.62
root@10.11.32.62's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.14.141-bpi-r2-main armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
System installed. You're running on eMMC now !
Last login: Fri Aug 13 10:36:22 2021 from 10.0.0.3
root@leaf-e5eaff97e2:~#
```

For Modberries, the message is only "**`System installed.`**"

If the controller is **not** correctly installed the message will be:

```
ls_power_diablo enst89m01c00n00s01 Deploy : ssh root@10.11.32.62
root@10.11.32.62's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.14.141-bpi-r2-main armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
System is NOT correctly installed. Restart the installation process.
Last login: Fri Aug 13 10:36:22 2021 from 10.0.0.3
root@leaf-e5eaff97e2:~#
```

15.3.2 Known problems

Early software version of delivery cubes has Banana Pi with small error in routing table, one gateway is wrong setup. As consequence the cube controller is not visible from array controller if the user collects the controllers IP address with fping.

The cube controllers are visible form the any of the controllers inside OCTE.

Proposed workflow.

- Install all controllers inside OCTE
- Copy the installation (`ControllerAllData.tgz`) to the Core Telco Rack controller and start the installation from there.

15.3.3 Manual Work

Small manual work is still required, but hopefully will be removed in coming releases. This is related only for Core Telco rack controller inside OCTE. To find which controller Telco rack controller use the script `controllerinfo.sh` or `controllertype.sh`

Logon to the Telco Rack controller.

Stop fluence service

```
systemctl stop fluence-modbus
cd /usr/local/share/fluence
```

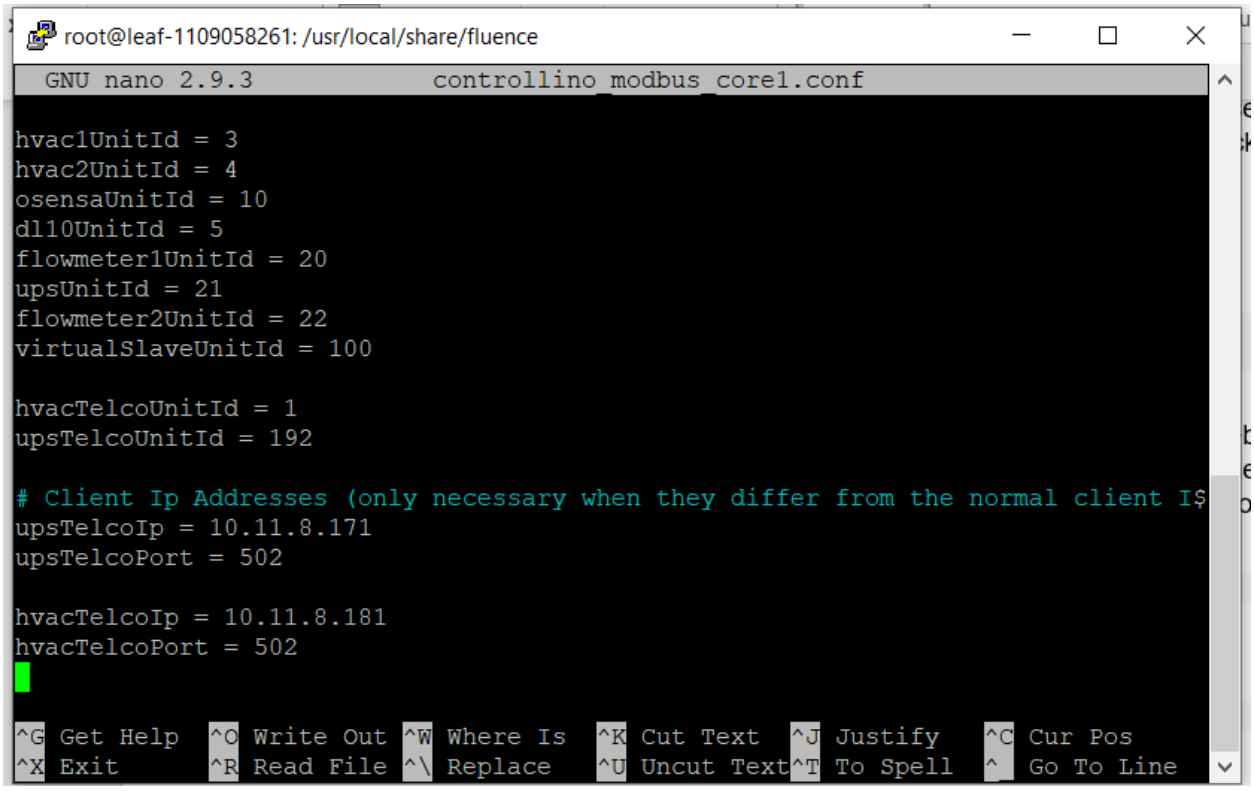
Default configuration file (controllino_modbus.conf) is for the cube, it is needed to be changed with controllino_modbus_octe.conf. Copy prototype controllino_modbus_octe.conf to new file where X is replaced with core number. Additional file will prevent to be overwritten once more of the update process is started again.

```
cp controllino_modbus_octe.conf controllino_modbus_coreX.conf
```

Edit controllino_modbus_coreX.conf

```
nano controllino_modbus_coreX.conf
```

Find the lines at the end of the file



Client Ip Addresses (only necessary when they differ from the normal client I$

```
upsTelcoIp = 10.11.8.171
upsTelcoPort = 502

hvacTelcoIp = 10.11.8.181
hvacTelcoPort = 502
```

Change IP addresses of the upsTelcoIp and hvacTelcoIp to fit with the core number. The first 3 octets must be the same as for the core controller.

Save the file (CTRL+O and press Enter) and exit (CTRL+X)

Copy controllino_modbus_coreX.conf as default file

```
cp controllino_modbus_coreX.conf controllino_modbus.conf
```

Start the service once more.

```
systemctl start fluence-modbus
```

16. Installing of Self-Configuration on Array Controller

The installation of the Self-Configuration on the array controller consists of two parts. First one is the installation of the config server itself and its small configuration, and second is the configuration of the site in the Self-Configuration server.

16.1 Installation and configuration of the Self-Configuration server

Note: In controller_software/ArrayController/Deploy/create_selfconfig_tgz.sh is a script which creates an actual package that can be installed on the array controller. The script creates a package selfconfig-<date>.tgz, which can then be committed to controller_software/ArrayController/Deploy folder. The script has to be executed in folder controller software like this: controller_software/ArrayController/Deploy/create_selfconfig_tgz.sh. The resulting tgz will be placed in the upper folder.

The install package can be found in the git repository in:

controller_software/ArrayController/Deploy/selfconfig-<date>.tgz

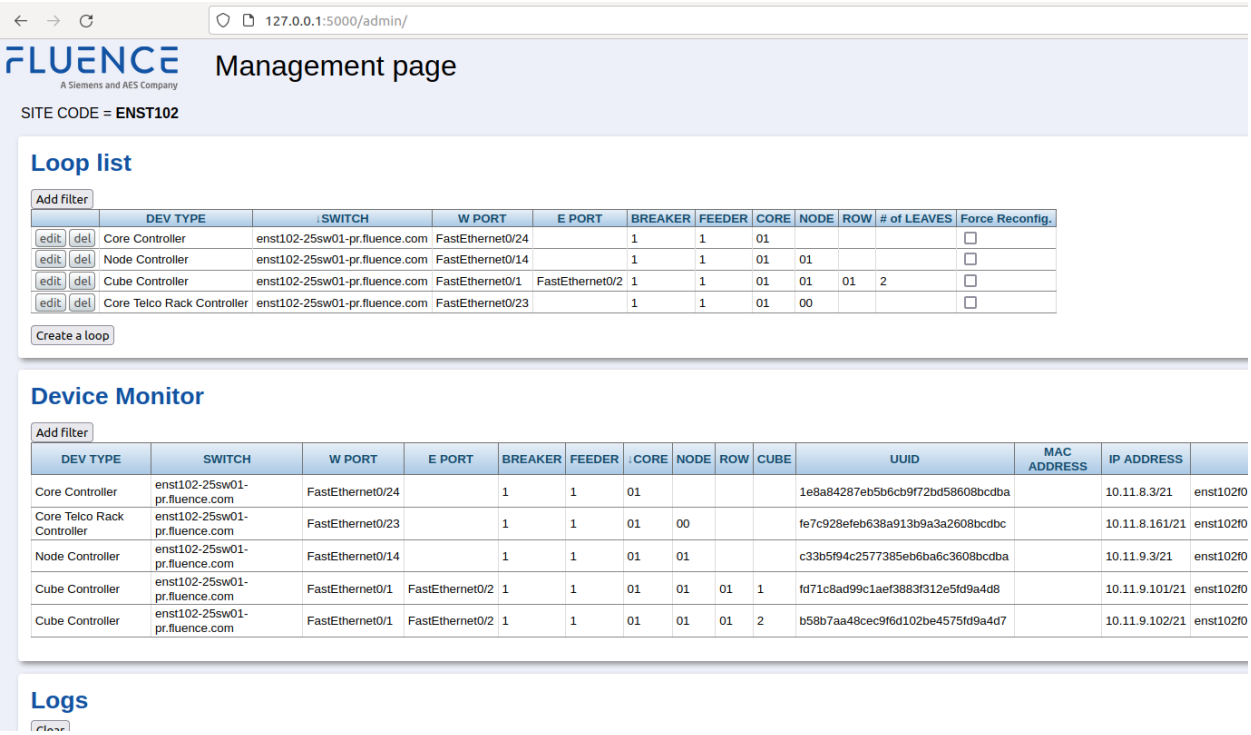
Steps:

1. Copy the selfconfig-<date>.tgz file in home directory of the root user on the array controller
2. Unpack the file with the command “tar xzvf selfconfig-<date>.tgz”
3. Edit the config.ini file, for example with nano editor
   “nano ./LeafController/SelfConfiguration/config.ini”
   and change the line with the site_code according to the site code of the project, i.e.
   site_code = enst00 → site_code = enst99
4. Change into the directory with “cd controller_software/ArrayController/Deploy”
5. Execute the installation script “bash ./deploy_script_selfconfiguration-server.sh”

That’s it. You should now be able to access the config frontend in a browser with:

http://<IP of array controller>:5000/admin

You should see something similar to this:



In the area of the Loop List, delete all entries by clicking the “del” button.

16.2 Configuring of the site in the Self-Configuration Server

What you need to configure the site in the server is:

1. The IP Excel sheet of the site
2. An understanding of the site, breaker, feeder, row etc.

By clicking “Add loop”, you can add each loop for the site (the cubes) as well as Core Telco Rack controller, Node Controller, Core controller.

The following cabling diagram will help you to configure them:

ETHERNET PORTS SCHEDULE			
PORT	DESIGNATION	DESCRIPTION	COLORING CODE
1X	1st CUBE ON ROW #1	SET OUT CAT 6 CABLE	GRAY
2X	LAST CUBE ON ROW #1	RETURN CAT 6 CABLE	
3X	1st CUBE ON ROW #2	SET OUT CAT 6 CABLE	SEA BLUE
4X	LAST CUBE ON ROW #2	RETURN CAT 6 CABLE	
5X	1st CUBE ON ROW #3	SET OUT CAT 6 CABLE	YELLOW
6X	LAST CUBE ON ROW #3	RETURN CAT 6 CABLE	
7X	1st CUBE ON ROW #4	SET OUT CAT 6 CABLE	YELLOW BLUE
8X	LAST CUBE ON ROW #4	RETURN CAT 6 CABLE	
9X	INVERTER 1	ETHERNET IP - CAT 6 CABLE	ORANGE
10X	INVERTER 2	ETHERNET IP - CAT 6 CABLE	
11X	INVERTER 3	ETHERNET IP - CAT 6 CABLE	
12X	INVERTER 4	ETHERNET IP - CAT 6 CABLE	
13X	UPS	ETHERNET IP - CAT 6 CABLE	GREEN
14X	NODE CONTROLLER #1	MODBUS TCP -IP - CAT 6 CABLE	RED
15X	NODE CONTROLLER #2	MODBUS TCP -IP - CAT 6 CABLE	
16X	NODE CONTROLLER #3	MODBUS TCP -IP - CAT 6 CABLE	
17X	NODE CONTROLLER #4	MODBUS TCP -IP - CAT 6 CABLE	
18X	HVAC	ETHERNET IP - CAT 6 CABLE	BLACK
19X	MBMU1	ETHERNET IP - CAT 6 CABLE	BROWN
20X	MBMU2	ETHERNET IP - CAT 6 CABLE	
21X	MBMU3	ETHERNET IP - CAT 6 CABLE	
22X	MBMU4	ETHERNET IP - CAT 6 CABLE	
23X	LEAF CONTROLLER	ETHERNET IP - CAT 6 CABLE	VIOLET
24X	CORE CONTROLLER	ETHERNET IP - CAT 6 CABLE	WHITE

Each device contacts the server periodically to get the configuration, you see this in the Web-frontend in the area “Logs”, which will present also swich name and Switch Port/s.

Once a device got a configuration, it will present in the Web-Frontend area “Device Monitor”.