



M3 Sensor & Digi Gateway Setup Procedure for MODBUS

Procedure

To convert an application for MODBUS operation, follow the steps shown below. This python code will not require knowledge of the Sensor's MAC address as it will automatically be detected (as long as it associates with the gateway) and then be assigned a device ID.

1) Retain all previous python files in the current gateway if you plan on restoring it later.

2) Load the following python files that contain the MODBUS python files and verify that the files sizes match what is show here to the right.

- dia.yml
- dia.zip
- dia.py
- python.zip
- zigbee.py

Note: The mbus_map.txt file shown here will only be displayed after you have booted the gateway with these files. Also, the dia.yml file will have sleep set to 2 minutes. See instructions later in this document to set these to your application.

Python Configuration

▼ Python Files

Upload Files

Upload Python programs

Upload File:

Warning: If you modify the Python files (archives or scripts), it is strongly recommended that you reboot your Digi device server for the modified files to take effect. Unpredictable behaviors may result if you do not reboot, depending on what has been modified.

Manage Files

Action	File Name	Size
<input type="checkbox"/>	mbus_map.txt	120 bytes
<input type="checkbox"/>	python.zip	144321 bytes
<input type="checkbox"/>	zigbee.py	1147 bytes
<input type="checkbox"/>	dia.zip	432404 bytes
<input type="checkbox"/>	dia.py	12484 bytes
<input type="checkbox"/>	dia.yml	1225 bytes

► Auto-start Settings

3) Verify the gateway's date/time is set to the current date & time.

4) Verify the gateway's XBee registers as: SC=0x1ffe, SN=2880, SP=1000.

5) Set the Auto-start command line to "dia.py dia.yml" and enable this line only. Don't enable "Action on Exit" at this time. Apply settings and then reboot the gateway.

6) After reboot, the python file "mbus_map.txt" will appear (see above). This will contain this one line:

```
# Dia Modbus Server unit_id mapping as of 2013-05-06 09:14:32
```

7) Next power the sensor and wait for it to associate to the gateway (red LED blinking every 2 seconds). If your sensor's sleep setting is zero, then it will require the sensor to be manually commissioned to the gateway. Do this by pressing the sensor's commission button once after it is blinking every 2 seconds which will broadcast to the gateway and program the sensor's settings per dia.yml settings. The "mbus_map.txt" file will be updated as shown here in this example. The sensor is now programmed to sleep and awake every 2 minutes and report new range. If you associate another sensor, a new line will be created with a new device ID #.

```
# Dia Modbus Server unit_id mapping as of 2013-05-06 09:14:32
1,'Massa M3/150','MassaM3','[00:13:a2:00:40:a8:20:48]!'
```

8) Every time the sensor wakes up, it will update the Modbus registers that are mapped out below.

Modbus	Offset	Mode	Description
4x00001	0	Read-Only	Magic Number, always 0xE801 for this register map
4x00002	1	Read-Only	Device Status as 16 bits
4x00004	3	Read-Only	Range as inches, fixed point * 100, so 1423 = 14.23 inches
4x00005	4	Read-Only	Temperature as Deg C, fixed point * 100, so 2315 = 23.15 DegC
4x00006	5	Read-Only	Target Strength, fixed point * 100, so 7500 = 75%
4x00007	6	Read-Only	Latest Event counter as word, so 1 to 65535
4x00013	12	Read-Only	Battery Voltage, fixed Point * 100, so 420 = 4.20v
4x00014	13	Read-Only	Low Word of UNIX style date/time of last data reading
4x00015	14	Read-Only	High Word of UNIX style date/time of last data reading

9) The dia.yml supplied has the sleep settings set for 2 minutes. Adjust these 2 settings below with the same setting your application requires. Reload the dia.yml into the gateway and reboot the gateway. This will update the sensor with the new settings. This example sets the sleep and data acquisition rate to 1 hour.

```
sample_rate_sec: 3600
sleep_rate_sec: 3600
```

Confirming MODBUS operation

This Connections Management page indicates verifies the python code is functioning with by line items “Modbus/TCP Listener”.

The “Modbus/TCP via TCP” and “Modbus/TCP via UDP” is the protocol of a Modbus scanner shown below.

Connections Management

Virtual Private Network (VPN) Connections

Action	Description	Remote Address	Local Address	Status
No VPN connections available				
<button>Refresh</button>	<button>Disable</button>			

Active System Connections

Action	Connected From	Connected To	Protocol	Sessions
<input type="checkbox"/>		Python: dia.py		0
<input type="checkbox"/>		Python thread		0
<input type="checkbox"/>		Python thread		0
<input type="checkbox"/>		Python thread		0
<input type="checkbox"/>		Python thread		0
<input type="checkbox"/>	TCP 502	Modbus/TCP Listener		0
<input type="checkbox"/>	UDP 502	Modbus/TCP Listener		0
<input type="checkbox"/>		Python thread		0
<input type="checkbox"/>		Python thread		0
<input type="checkbox"/>	169.254.195.17	Modbus/TCP via TCP		0
<input type="checkbox"/>		127.0.0.1	Modbus/TCP via UDP	0

RefreshDisconnect

A MODBUS listener application program seen here verifies the sensor status data is addressed to the locations as defined in the table above. Here, the target distance is reporting at 11.95”.

ModScan1	
Address: 0001	Device Id: 1
Length: 15	MODBUS Point Type
	03: HOLDING REGISTER
<pre> 40001: <59393> 40002: <12800> 40003: <00000> 40004: <01195> 40005: <02162> 40006: <07500> 40007: <00010> 40008: <00000> 40009: <00000> 40010: <00000> 40011: <00000> 40012: <00000> 40013: <00452> 40014: <35491> 40015: <20871> </pre>	

A Presentations page can be viewed at: http://.../idigi_dia.html showing the sensor status.

iDigi Dia	
Massa M3/150	
FWa_version:	031
FWb_version:	021
battery:	4.525 V
distance:	11.953125 in
event:	10
gain:	Low
sensor_model:	M3-150
serial_number:	0000
strength:	Very Strong
target_strength:	75 %
temperature:	21.62437 C