Ecotron

Assembly Manual

Table of Contents

Pin mapping.....2
- SIM808 Module.....2
- Temperature Sensor.....2
- Ultrasonic Rangefinder Sensors.....3
- Built-in connections.....3

Assembly.....4
- Seeeduino Stalker.....4
- SIM808.....4
- HC-SR04 sensors.....5
- Sensor Shield.....5
- Power supply.....6

Programming.....7
- Connecting programmer.....7
- Arduino IDE setup.....7
- Firmware changes.....7
- Uploading firmware.....7

Testing.....8
- Debug mode.....8
- Manual mode.....11
- Connection to server.....11

Pin mapping

SIM808 Module

SIM808	**Arduino**
TX	8
RX	10
DTR	11
RI	12
PWR	13
RST	Not used
VIO	Vcc
GND	Any GND
VBAT	Battery

Notes:
- RX/TX use SoftwareSerial library
- avoid using pin 13 as input
- SIM808 should not be powered via Seeeduino Stalker Vcc, which is regulated, for it may not be able to give enough power. Direct power supply from battery give better overall results.

Temperature Sensor

Temperature sensor	**Arduino**
Vcc	Any Vcc
GND	Any GND
Signal	A0

Ultrasonic Rangefinder Sensors

Can number	**HC-SR04 module**	**Arduino**
Far right (can 1)	Trigger	0
	Echo	1
Far left (can 5)	Trigger	4
	Echo	5
Middle left (can 4)	Trigger	6
	Echo	7
Close right (can 2)	Trigger	A1
	Echo	A2
Close left (can 3)	Trigger	3 (digital!)
	Echo	A3

Notes:
- Vcc and GND required for each sensor can be connected to any Vcc and GND
- positions refer to the actual container with 5 separate bins left and right as seen by device from within the container (not outside viewer!)
- from the outside, cans are conveniently ordered 1-5 from left to right

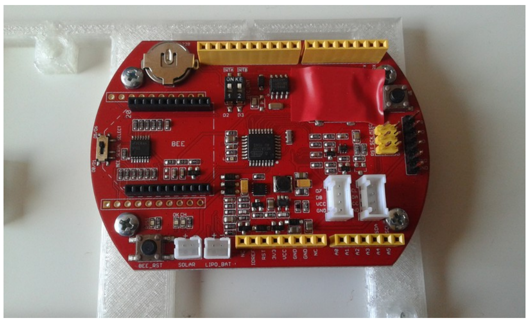
Built-in connections

Sensor power switch – high on	9
RTC Hardware interrupt	2
RTC I2C SDA	A5
RTC I2C SCL	A4

Notes:
- Seeeduino Stalker 3.1 has a built-in FET that powers Vcc, controlled by pin 9 – use it to supply power to sensors
- RTC interrupt pin can be changed – don't

Assembly

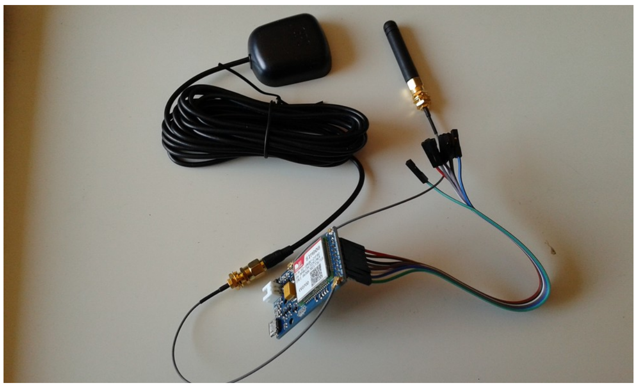
Seeeduino Stalker

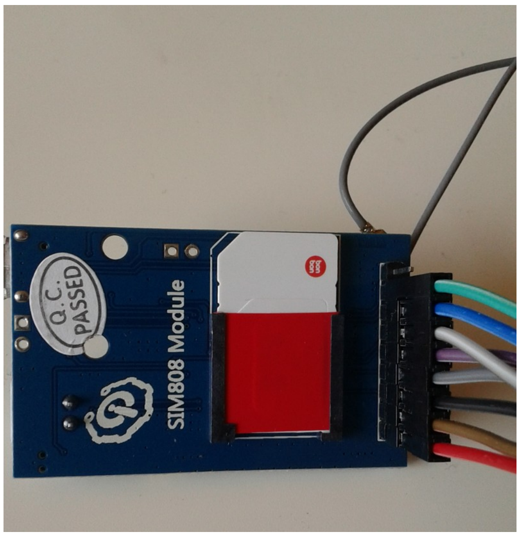


SD Card socket does not fit tight, fix it using some duct tape.

SIM808

All pins of SIM808 except RST are used, antennae are connected via uFL sockets.

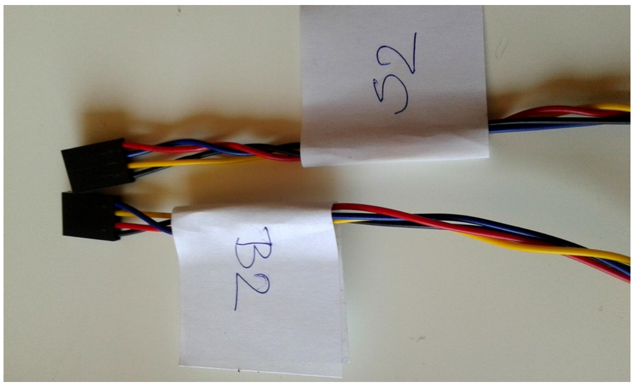


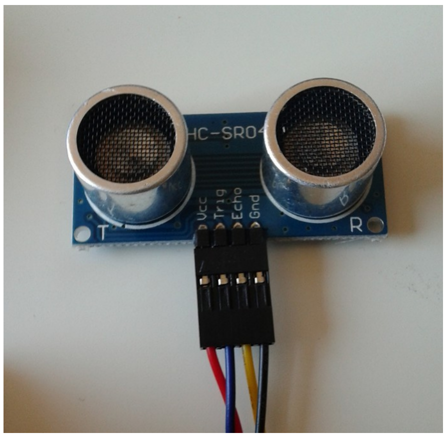


SIM card socket does not fit tight. To keep SIM in place, use a bit of duct tape.

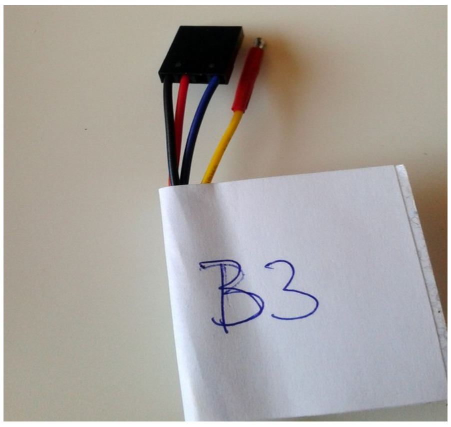
HC-SR04 sensors

Note that wires are crossed. Labelling is handy.





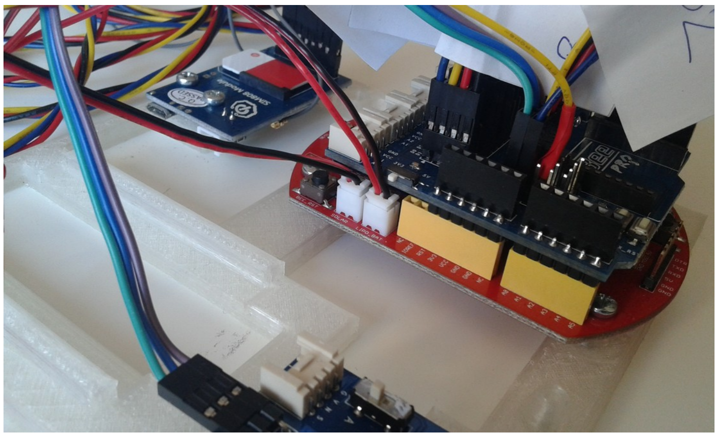
Sensor 3 is connected to one digital and one analog pin!



Sensor Shield

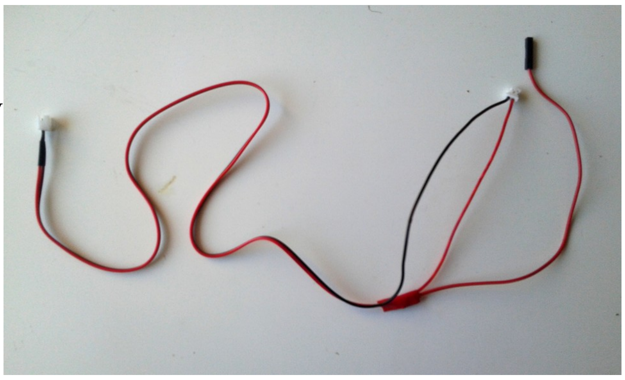
Note that battery and solar panel cannot be plugged in while sensor shield is in place – this is design flaw of Seeeduino Stalker.

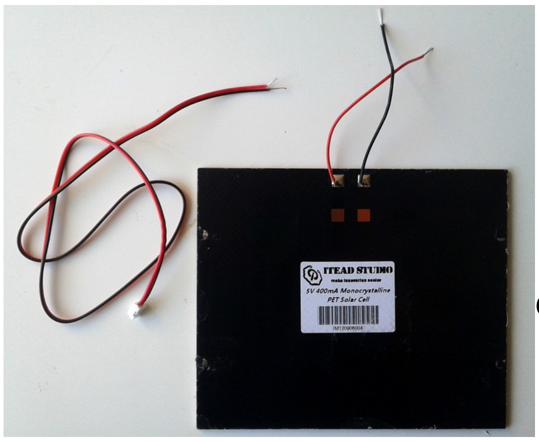
However, plugging sensor cables is easier when shield is in place.



Power supply

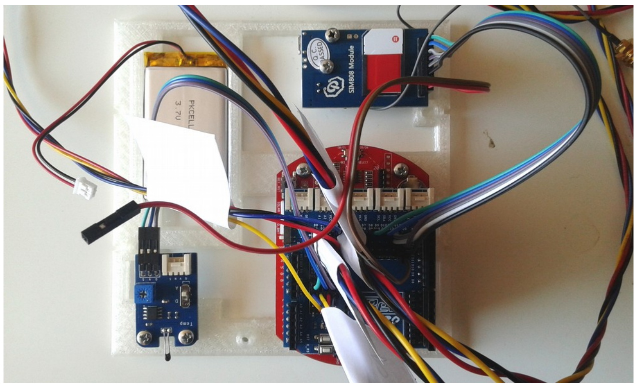
Prepare the cable first: JST-PH goes to battery slot of Stalker, and another wire to VBAT of SIM808 module.





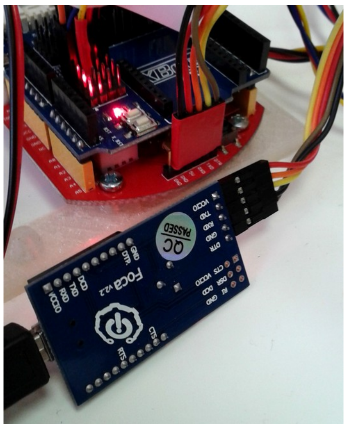
Guess what, solar panel cables need to be soldered.

... then plug them into Stalker, add sensor shield on top of it.



Programming

Connecting programmer



Connect the thing to programmer – we use Foca by Itead.

Plug programmer's USB cable into PC.

Arduino IDE setup

In Arduino IDE, choose *Arduino Pro or Pro mini* under *Tools* -> *Board*.
Under *Tools* -> *Processor*, select *Atmega 328 (3.3V, 8MHz)*.
Select port board is connected to in *Tools* -> *Port*.

Firmware changes

Each SIM cart has it's own PIN. In addition, depending on mobile operator, APN may need to be changed.
Look up PIN and APN in SIM808.ino, and set to your values, i.e.

```
// carrier data for GSM network
String PIN="8215";
String APN="mobileinternet.tele2.hr";
```

Server URL also needs to be changed, at least Station ID:

```
// server data
String SERVER="http://www.mikrotron.hr/ecotronserver/upload?stationId=gama";
```

Uploading firmware

USB connection requires digital ports 0 and 1, used to read the data from ultrasonic rangefinder #1. It may interfere with firmware upload, so disconnect the sensor before uploading.

Upload the firmware by clicking upload button, or pressing CTRL+U.

Testing

Sim808.ino contains flags intented for testing. Two most important are *DEBUG* and *MANUAL*.

```
// runtime options
#define DEBUG true      // run in debug mode
#define MANUAL true    // manual control mode
```

They both need to be *false* in regular operation.

For testing first set *DEBUG* to *true*.

Debug mode

Upload the firmware, and the device will start measuring heights, one by one sensor.
Note that sensor one will not measure anything when serial connection is used, and is best left disconnected.

Here's an example of debug output, with comments.

Firmware first restarts SIM808 module:

```
restarting sim808...OFF
ON
restarting sim808...OFF
ON
```

After the SIM808 has been reset, measurement starts:

```
can 1
3.57, 1: -1
can 2
187.94, 191.07, 190.62, 190.38, 191.55, 2: 190
can 3
188.87, 190.65, 191.34, 190.69, 190.93, 3: 190
can 4
188.18, 191.75, 191.07, 191.27, 190.48, 4: 191
can 5
5: -1
```

In the above example, *can 1* is disconnected intentionally, wiring for *can 5* needs to be fixed.
Device then connects to the network, by issuing a number of *AT commands* to SIM808:

```
init...
AT flushing 21 bytes
9,at
OK

OK
echo flushing 0 bytes
11,ate0
OK

OK
flushing 0 bytes
22,
+CPIN: READY
```

```
OK

SIM RDY
sim started!
net reg flushing 0 bytes
20,
+CREG: 0,1

OK

OK
gprs attach flushing 0 bytes
19,
+CGATT: 1

OK

OK
IP shut flushing 0 bytes
11,
SHUT OK

OK
IP stat flushing 0 bytes
27,
OK

STATE: IP INITIAL

OK
mux off flushing 0 bytes
6,
OK

OK
network flushing 0 bytes
6,
OK

OK
gprs link flushing 0 bytes
23,
OK

+CMTI: "SM",1

OK
IP flushing 0 bytes
17,
10.203.84.114

ERR
gprs started!
contype flushing 0 bytes
20,
OK

+CMTI: "SM",
OK
set apn flushing 3 bytes
6,
OK

OK
gprs ON flushing 0 bytes
6,
OK

OK
settings flushing 0 bytes
54,
+SAPBR: 1,1,"10.203.84.114"

OK

+CMTI: "SM",3

OK
gprs configured!
HTTP start flushing 0 bytes
6,
OK

OK
CID flushing 0 bytes
6,
OK

OK
REDIR flushing 0 bytes
6,
OK

OK
GPS on flushing 0 bytes
6,
OK
```

Once it's connected, it starts reading GPS coordinates:

```
OK
GPS INFO GPS parse flushing 0 bytes
60,
+CGNSINF: 1,0,19800105235943.000,,,,0.00,0.0,0,,,,,,0,0,,
10,,,

OK


+CGNSINF: 1,0,19800105235943.000,,,,0.00,0.0,0,,,,,,0,0,,,,,

OK
```

GPS may need some time to lock on to available satellites. Messages like this reappear for a while, until all GPS values are successfully read, or a time period expires.

```
OK
got 1,0,20170309131859.000,,,,0.28,179.6,0,,,,,,4,3,,,32,,
bot 0
GPS parse flushing 0 bytes
60,
+CGNSINF: 1,1,20170309131904.000,45.789780,15.928943,96.30
46,0,3.32,76.6,1,,3.8,3.9,1.0,,5,4,,,30,,

OK
```

Once GPS values are read, all values are sent to the server:

```
+CGNSINF: 1,1,20170309131904.000,45.789780,15.928943,96.300,3.32,76.6,1,,3.8,3.9,1.0,,5,4,,,30,,

OK

OK
got 1,1,20170309131904.000,45.789780,15.928943,96.300,3.32,76.6,1,,3.8,3.9,1.0,,5,4,,,30,,
bot 0
GPS off flushing 0 bytes
6,
OK

OK
batstat flushing 0 bytes
6,
OK

OK
GET flushing 0 bytes
31,
OK

+HTTPACTION: 0,200,26

OK
HTTP stop flushing 0 bytes
6,
OK

OK
4:8
OFF
INIT SLEEP
```

... and, device initiates sleep sequence.

Manual mode

Should any off above AT commands fail, it can be tested manually. To do so, turn on *MANUAL* flag in Sim808.ino. (*DEBUG* flag also has to be *true*) Upload the firmware, and start serial monitor (ctrl+shift+M). Any AT command typed to serial console is routed directly to SIM808 module, and every response from SIM808 is written to serial console.

First command issued should probably be plain AT – if module does not respond, check RX/TX connection. Good idea is to instruct SIM808 to report verbose errors:
AT+CMEE=2

Two errors in assembly are typical:
1) SIM808 does not respond to any commands – check RX/TX connections
2) SIM card does not respond to PIN – instead of `+CPIN: READY` an error message appears. If that happens, check
    1) PIN in SIM808.ino
    2) make sure that SIM card is properly inserted and fixed in SIM808

Connection to server

Fastest way to check if device sent data to the server is opening station's URL in web browser, i.e.

http://www.mikrotron.hr/ecotronserver/last?stationId=epsilon
The server should respond with received data in JSON format, i.e.

```
{"stationId":"epsilon","time":"2017-03-09 14:37:23.163","can1":-
1.0,"can2":190.0,"can3":190.0,"can4":191.0,"can5":11.0,"temp":9.0,"gpsInfo":"1,1,20170309133713.000,
45.789935,15.929025,107.600,0.07,234.1,1,,3.5,3.7,1.0,,7,3,,,31,,","gpsTime":"2017-03-09
13:37:13.0","gpsLatitude":45.789936,"gpsLongitude":15.929025,"gpsAltitude":107.6,"gpsSpeed":0.07,"gp
sCourse":234.1,"gpsHdop":3.5,"gpsPdop":1.0,"gpsVdop":3.7,"batInfo":"0,88,4064"}
```