



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield





ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

Document Information

Title	ERF3000v2
Subtitle	LTE Cat. M1, Cat. NB1, EGPRS & GNSS Arduino Shield
Document type	User Guide
Document number	1
Revision and date	R1 12-06-2019

Product status

In production

this document applies to the following products:

Name	Type number	version	Product status
ERF3000	ERF3000	1.5	EOL
ERF3000v2	ERF3000	1.0	In production



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

Contents

Introduction	4
Set-up	4
1 Getting a Network connection	6
1.1 AT – commands	6
1.2 Examples	7
2 Firmware update	8
2.1 USB	8
2.2 DFOTA	10
3 Creating a Low-Power application	11
3.1 PSM	11
3.2 eDRX	11
3.3 Antenna design	11
3.3.1 Antenna design tuning	14
4 Arduino Programming	20
4.1 Setup	20
4.2 Examples	21
5 Measuring the current consumption	22
5.1 Preparations	22
5.2 Results	23



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

Introduction

This document functions as a Quick start guide / User guide. Several basic processes and features are described in the document to help speed up the development and evaluation time. Several of these basic processes are: Getting a network connection, performing a firmware update, creating an Arduino software example and explaining how to create a Low power Internet Of Things (IoT) application.

Set-up

To start evaluating the shield, please follow below steps:

Step 1: Connect the GSM antenna.

Step 2: Plug in a Nano SIM card.

Step 3: Plug in a USB mini cable and connect it to a pc/laptop.

Step 4: Plug in the power adapter supplied with the shield.

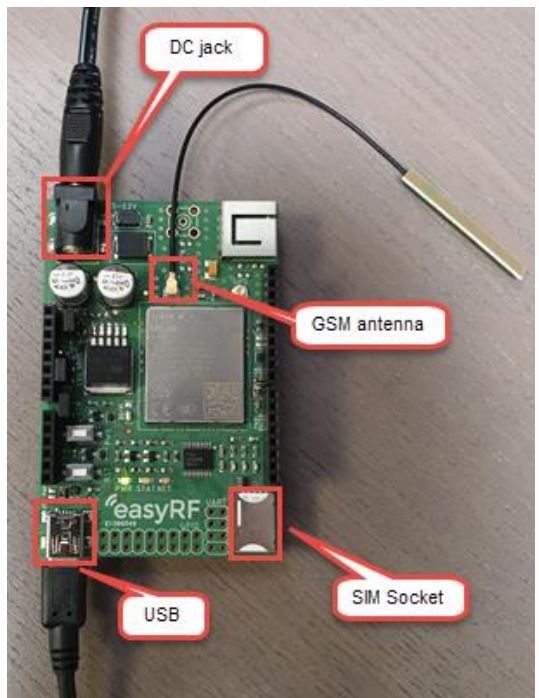


Please see chapter 3.1 Power of the [ERF3000 Datasheet](#) when using a different adapter.

Step 5: When power is supplied to the shield the green LED marked with PWR should light up.

Step 6: Press the PWR button to start the shield.

Step 7: After the module is done with starting up, the yellow LED marked with STAT should light up, and the red LED marked with NET will start blinking.



When above steps are followed, the shield is ready to receive AT-commands. Before a serial connection to the shield can be opened, the correct drivers need to be installed on your device.

Please download them from the [ERF3000v2 Github page](#). When the download is done, open and extract the .zip file.






ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

Inside the ERF3000v2-master folder navigate to the “Driver” folder, and open the Quectel_LTE_Windows_USB_Driver_V2.0.zip. inside this .zip folder you can find a setup.exe file, install this on you device.

When the installation is done, the PC will recognize the shield.

To verify this go to, device management and look for the following 3 ports.

-  Quectel USB AT Port (COM5)
-  Quectel USB DM Port (COM3)
-  Quectel USB NMEA Port (COM4)

Now any serial software like Putty, Realterm, etc.. can be used to send AT-commands to the BG96.

Quectel also developed special tools for their modules, these are called Qnavigator and QCOM. These programs can also be found in the ERF3000v2-master folder in the “Tool” folder.

- QCOM has the option to write scripts that will be send to the BG96 module.
- QNavigator is a dashboard environment with some examples.

Open either of the programs and set the COM Port settings to:

- 115200 baud
- Stopbits: 1
- Parity: none
- Bytesize: 8
- Flow control: no flow control

Set the COM Port to the USB AT Port, the number of the port can be found in your device management.

Now open the port and send AT, the module should respond with OK.





ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

1 Getting a Network connection

1.1 AT – commands

In order to establish a connection, the module must be initialized with AT-commands. Before attempting to connect to a network, the module must be initialized. Please note, most providers use different settings for all below commands. See chapter [1.2 Examples](#) for examples of the Dutch network providers.

Command	Description
AT+CFUN=0	Will disable all phone functionality of the module

When the phone functionality is disabled please set below settings.

Command	Description
AT+QCFG="band"	This command specifies the frequency bands allowed to be searched of UE
AT+QCFG="nwscanseq"	This command specifies the searching sequence of RATs
AT+QCFG="nwscanmode"	This command specifies the RAT(s) allowed to be searched
AT+QCFG="iotopmode"	This command specifies the network category to be searched under LTE RAT
AT+QCFG="nbsibscramble"	This command specifies the scramble function
AT+CGDCONT	This command specifies the PDP context parameters for a specific context

When the commands are set, the phone functionality can be activated again, and a connection to the network can be forced.



Before forcing a connection to a network, please check with your local provider if IMEI registration of the Quectel hardware is necessary. If you don't register the IMEI number with your provider and still connect to their network you may be put on a blacklist, and access to the network will be blocked. The IMEI number can be found on the metal casing of the Quectel module.

Command	Description
AT+CFUN=1,1	Will activate all phone functionality of the module and reset the module
AT+COPS	This command forces an attempt to select and register the network operator

When AT+COPS returns OK. A connection has been established. Information about the connection can be gained with the following commands:

Command	Description
AT+QNWINFO	This command indicates network information such as the access technology selected, the operator, and the selected band.
AT+QSPN	This command will display the name of the registered network
AT+CSQ	This command indicates the received signal strength (RSSI) and the channel bit error rate
AT+QCSQ	This command is used to query and report the signal strength of the current network service. Dependent on the network this will return: RSSI, RSRP, SINR & RSRQ
AT+QENG="neighbourcell"	This command will return the neighboring cell towers information
AT+QENG="servingcell"	This command will return the serving cell tower information



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

1.2 Examples

Below are examples of network settings for all 3 major Dutch network providers:

KPN NL
CAT-M Commands
AT+QCFG="band",1,80000,1,1
AT+QCFG="nwscanseq",020202,1
AT+QCFG="nwscanmode",3,1
AT+QCFG="iotopmode",2,1
AT+QCFG="nbsibscramble",0
AT+CGDCONT=1,"IP","item.webtrial.m2m"
AT+COPS=1,2,"20408",8

T-Mobile NL	
CAT-M Commands	NB-IoT Commands
AT+QCFG="band",1,4,80,1	AT+QCFG="band",1,4,80,1
AT+QCFG="nwscanseq",030303,1	AT+QCFG="nwscanseq",030303,1
AT+QCFG="nwscanmode",3,1	AT+QCFG="nwscanmode",3,1
AT+QCFG="iotopmode",2,1	AT+QCFG="iotopmode",2,1
AT+QCFG="nbsibscramble",0	AT+QCFG="nbsibscramble",0
AT+CGDCONT=1,"IP","smartsites.t-mobile"	AT+CGDCONT=1,"IP","cdp.iot.t-mobile.nl"
AT+COPS=1,2,"20416",8	AT+COPS=1,2,"20416"

VodafoneZiggo NL	
CAT-M Commands	NB-IoT Commands
AT+QCFG="band",1,80000,80000,1	AT+QCFG="band",1,80000,80000,1
AT+QCFG="nwscanseq",020202,1	AT+QCFG="nwscanseq",030303,1
AT+QCFG="nwscanmode",3,1	AT+QCFG="nwscanmode",3,1
AT+QCFG="iotopmode",2,1	AT+QCFG="iotopmode",2,1
AT+QCFG="nbsibscramble",0	AT+QCFG="nbsibscramble",0
AT+CGDCONT=1,"IP","live.vodafone.com"	AT+CGDCONT=1,"IP","nb.inetd.gdsp"
AT+COPS=1,2,"20404",8	AT+COPS=1,2,"20404",9



Before forcing a connection to a network, please check with your local provider if IMEI registration of the Quectel hardware is necessary. If you don't register the IMEI number with you provider and still connect to their network you may be put on a blacklist, and access to the network will be blocked. The IMEI number can be found on the metal casing of the Quectel module.

For more information please find the AT-Manuals in the ERF3000v2-master/Docs/AT manuals folder.



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

2 Firmware update

2.1 USB

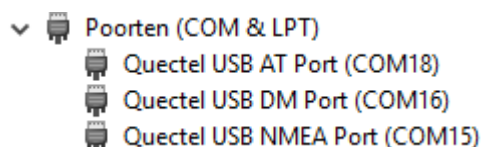
For the update through the USB interface the tool Qflash is needed, this tool can be found on the ERF3000v2 github page.

In order to get a firmware package please contact Quectel or TOP-electronics via: support@top-electronics.com.

Step 1: Turn on the ERF3000 shield by plugging in the adapter supplied with the shield and pressing the PWR button.

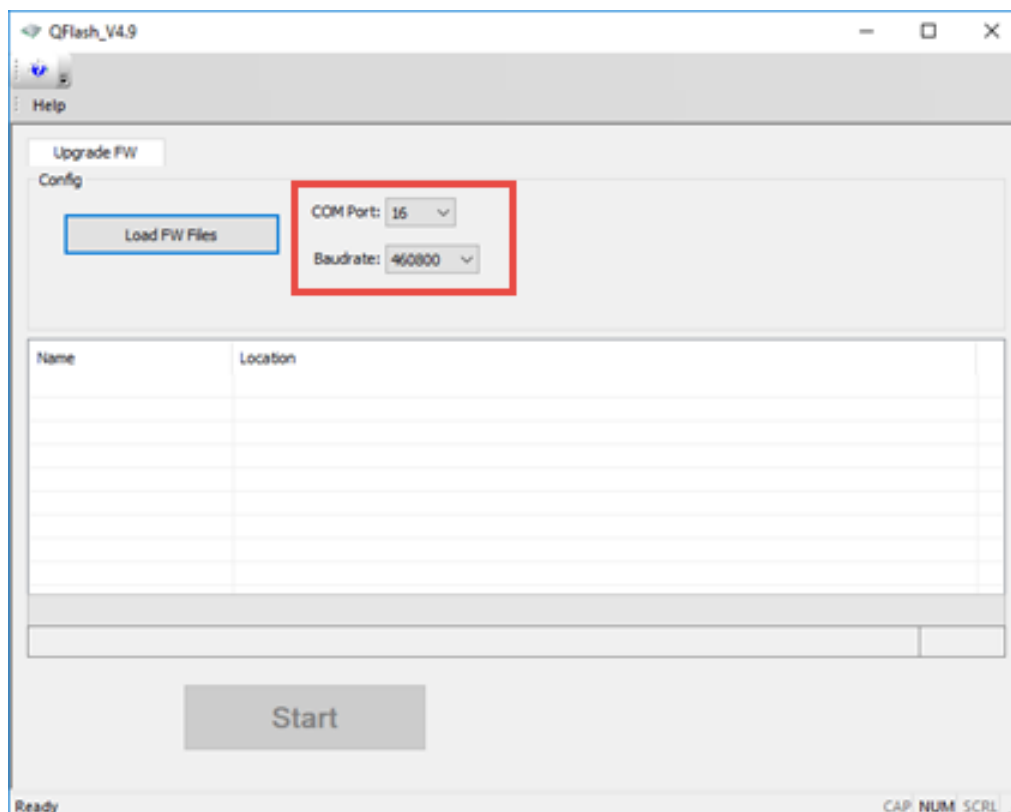
Step 2: Connect the ERF3000 to a laptop/PC through the USB interface.

Step 3: On your PC/Laptop look at your device management and find the "Quectel USB DM Port".



Step 4: Open the Qflash software

Step 5: Select the "Quectel USB DM Port" for the COM Port: and set the baudrate to 460800.

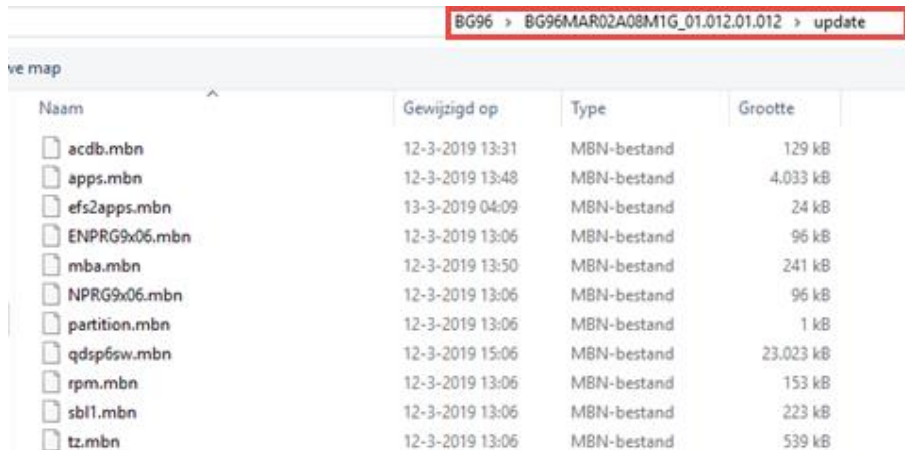




ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

Step 6: Select the Firmware with the “Load FW Files”, This will open an explorer tab.

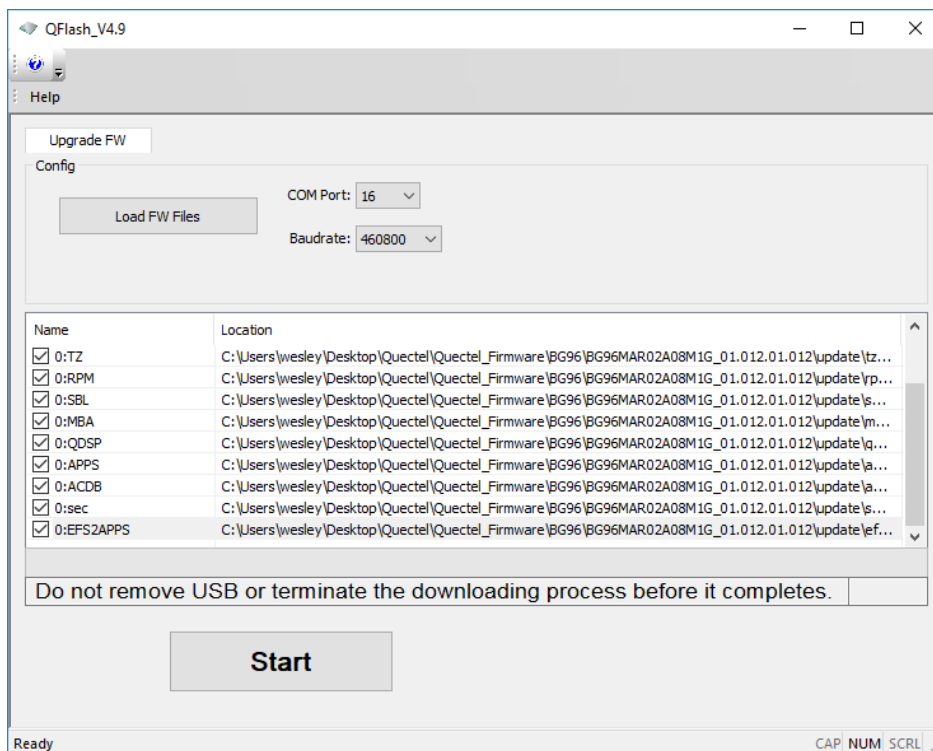
Navigate to the firmware folder and into the “update” folder.



Naam	Gewijzigd op	Type	Grootte
acdb.mbn	12-3-2019 13:31	MBN-bestand	129 kB
apps.mbn	12-3-2019 13:48	MBN-bestand	4.033 kB
efs2apps.mbn	13-3-2019 04:09	MBN-bestand	24 kB
ENPRG9x06.mbn	12-3-2019 13:06	MBN-bestand	96 kB
mba.mbn	12-3-2019 13:50	MBN-bestand	241 kB
NPRG9x06.mbn	12-3-2019 13:06	MBN-bestand	96 kB
partition.mbn	12-3-2019 13:06	MBN-bestand	1 kB
qdsp6sw.mbn	12-3-2019 15:06	MBN-bestand	23.023 kB
rpm.mbn	12-3-2019 13:06	MBN-bestand	153 kB
sbl1.mbn	12-3-2019 13:06	MBN-bestand	223 kB
tz.mbn	12-3-2019 13:06	MBN-bestand	539 kB

Select 1 of the “.mbn” files and click on “Open”. The software will automatically select all relevant files for the update.

Step 7: Click on the start button to begin the update.

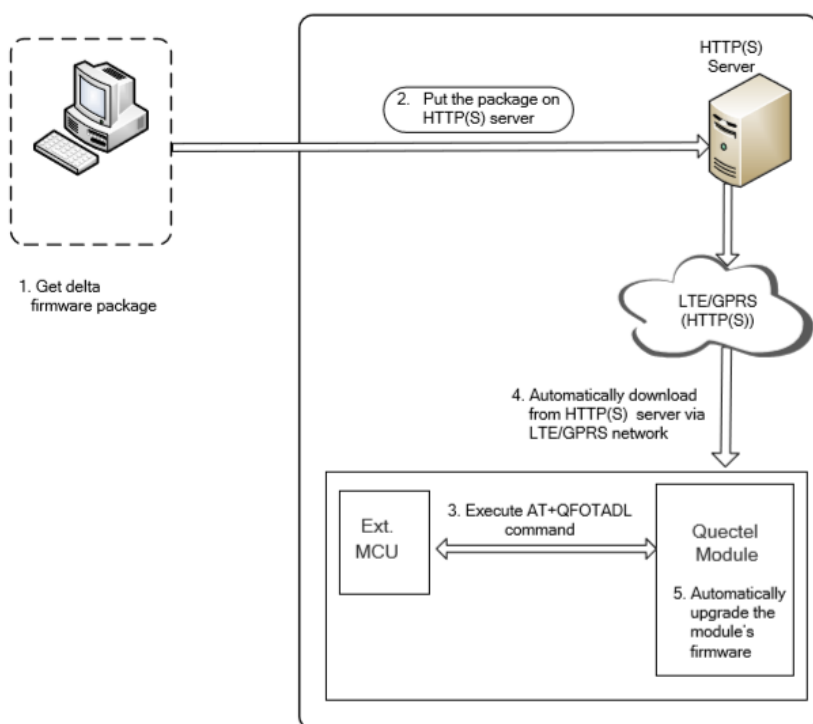


ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

2.2 DFOTA

In order to perform a DFOTA update, you will need to get a special DFOTA package. This can be obtained through contacting TOP-electronics via: support@top-electronics.com or Quectel. Please state your current firmware and specify what firmware version you would like to upgrade/downgrade to.

When the DFOTA package has been received please follow these steps:



Step 1: Get the delta firmware package.

Step 2: Put the delta firmware on a HTTP(s) server.

Step 3: Power on module and make sure a network connection is established.

Step 4: Execute AT+QFOTADL command. Then the module will automatically download the package from HTTP(S) server via LTE/GPRS network.

Command	Description
AT+QFOTADL=<httpURL>	This command will start the DOFTA process. (example AT+QFOTADL="https://www.quectel.com:100/update.zip")



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

3 Creating a Low-Power application

3.1 PSM

The Power saving mode (PSM) is the most important feature when creating a low-power application. This feature allows the module to enter a deep sleep mode for a specified time. When the specified time expires, the module will come back online. During the Power saving mode the current consumption drops to 10uA. The PSM can be configured with the following commands:

Command	Description
AT+QCFG="psm/enter",1	This command is used to trigger the module to enter into PSM mode immediately after the RRC released. When this function is enabled and RRC connection release is received, the module will skip active timer (T3324) and enter into PSM mode immediately
AT+QCFG="psm/urc",1	This command will allow the +QPSMTIMER URC message to be displayed, when the module receives the RRC release
AT+CPSMS=1,,,"TAU","active"	This command sets the TAU (T3312) and active (T3424) timers
AT+QPSMS=1,,,"TAU","active"	This command extends the previous set AT+CPSMS active and sleep timers

Please see chapter [5 Measuring the current consumption](#) for an example on how to measure the current consumption during this mode.

3.2 eDRX

The extended discontinuous reception (eDRX) is a feature to lower current consumption while connected to a network. Using the eDRX settings the interval between communication to the network can be manually set. The default value for M2M communication is 10.24 seconds. For IoT applications this time can be prolonged to 2,9 hours. In between the eDRX paging cycles, the module will turn off its radio and thus lowering the consumed current. Using the below commands the eDRX settings can be tuned.

Command	Description
AT+CEDRXS	This command will initialize the eDRX settings
AT+CEDRXRDP	This command will report the dynamic parameters of the extended eDRX settings

Please see chapter [5 Measuring the current consumption](#) for an example on how to measure the current consumption during this mode.

3.3 Antenna design

The antenna design also has great influence on the total power consumption of the module. The consumed power is in direct correlation with the RSSI value. The RSSI value of the connection can be retrieved with the following command:

Command	Description
AT+CSQ	This command will return the signal strength (RSSI) of the connection

The RSSI value is in its place in correlation with the Impedance and the return loss of the antenna (design).

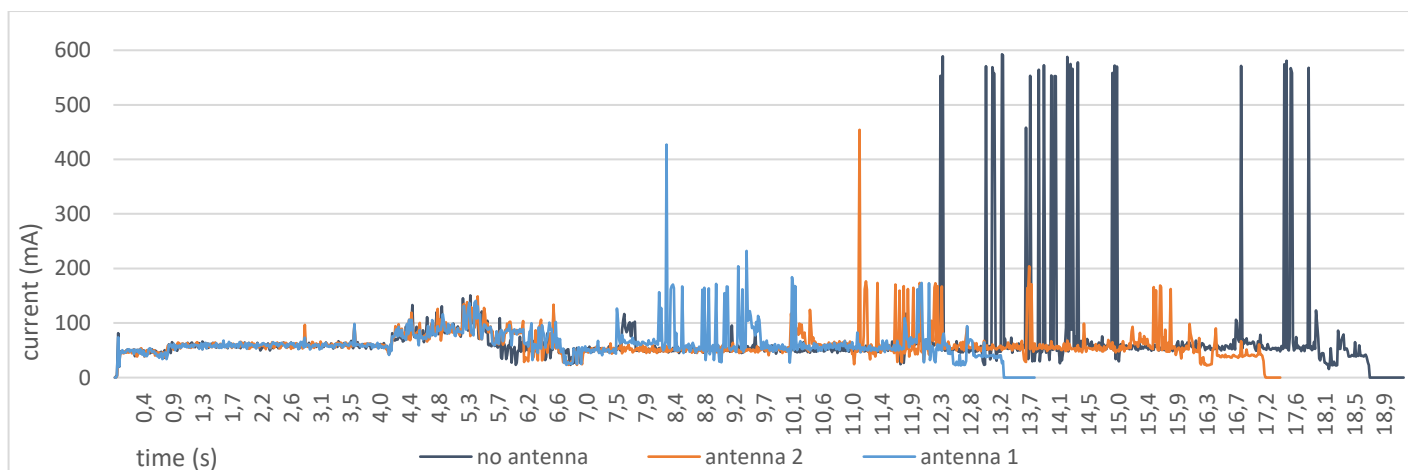


ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

In order to measure the return loss and impedance of an antenna (design), a Vector Network Analyzer (VNA) is needed. Using the VNA you can measure the impedance of your antenna design and get a representation of your antenna performance. To give an example of how important the antenna design is, see below test results:

The following tests are performed on the T-Mobile LTE Cat NB1 network. This network uses LTE B8 band with a center frequency of 900MHz.

Antenna	Return Loss @900 MHz	RSSI
Antenna 1 ERF4007	-16,14dB	-59dB
Antenna 2 ERF4041	-3,53dB	-72dB
No antenna	-0,22dB	-106dB



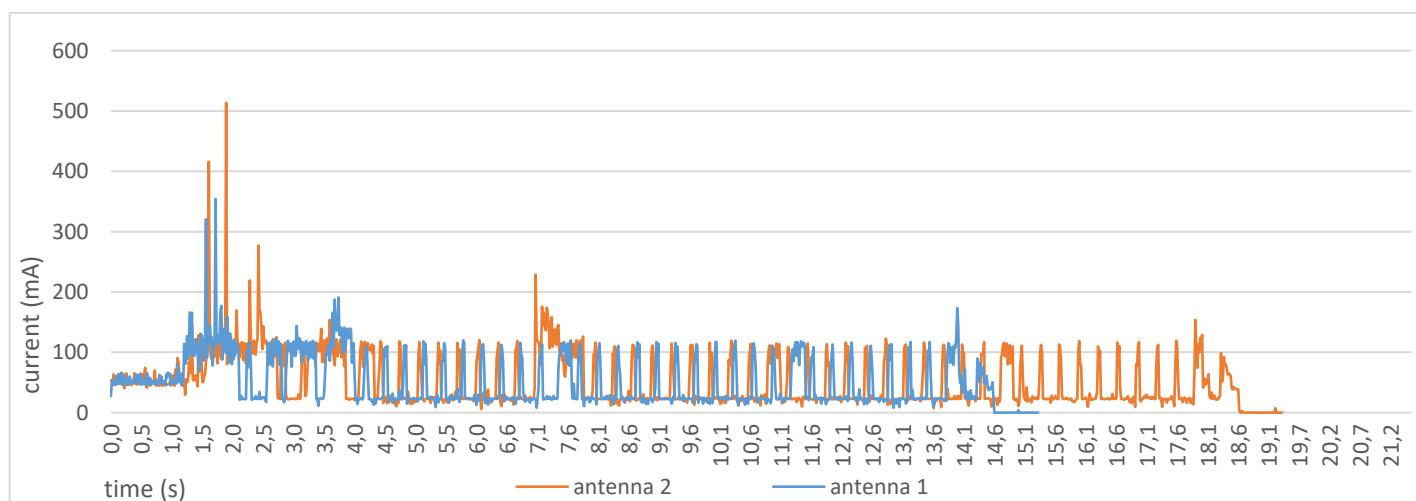
Results transmission	Avg current	Total time	Total	% increase in current
Antenna 1	62,26mA	13,75 sec	856,075mA	100%
Antenna 2	60,07mA	17,42 sec	1046,42mA	122%
No antenna	67,39mA	19,31 sec	1301,30mA	152%



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

The following tests are performed on the KPN LTE Cat M1 network. This network uses LTE B20 band with a center frequency of 800MHz.

Antenna type	Return Loss @800 MHz	RSSI
Antenna 1 ERF4041	-13,13dB	-62dB
Antenna 2 ERF4061	-3.5 dB	-77dB
No antenna	No connection possible	-



Results transmission	Avg current	Total time	Total	% increase in current consumption
Antenna 1	50,25mA	15,34 sec	770,84mA	100%
Antenna 2	50,22mA	19,39 sec	973,77mA	126%

As seen in the graphs the antenna design is a big part of creating an optimal low power application. In order to get an optimal antenna performance, measurements with a VNA are necessary.

An example of a suitable VNA supplied by our partner TOP-electronics can be found [here](#).

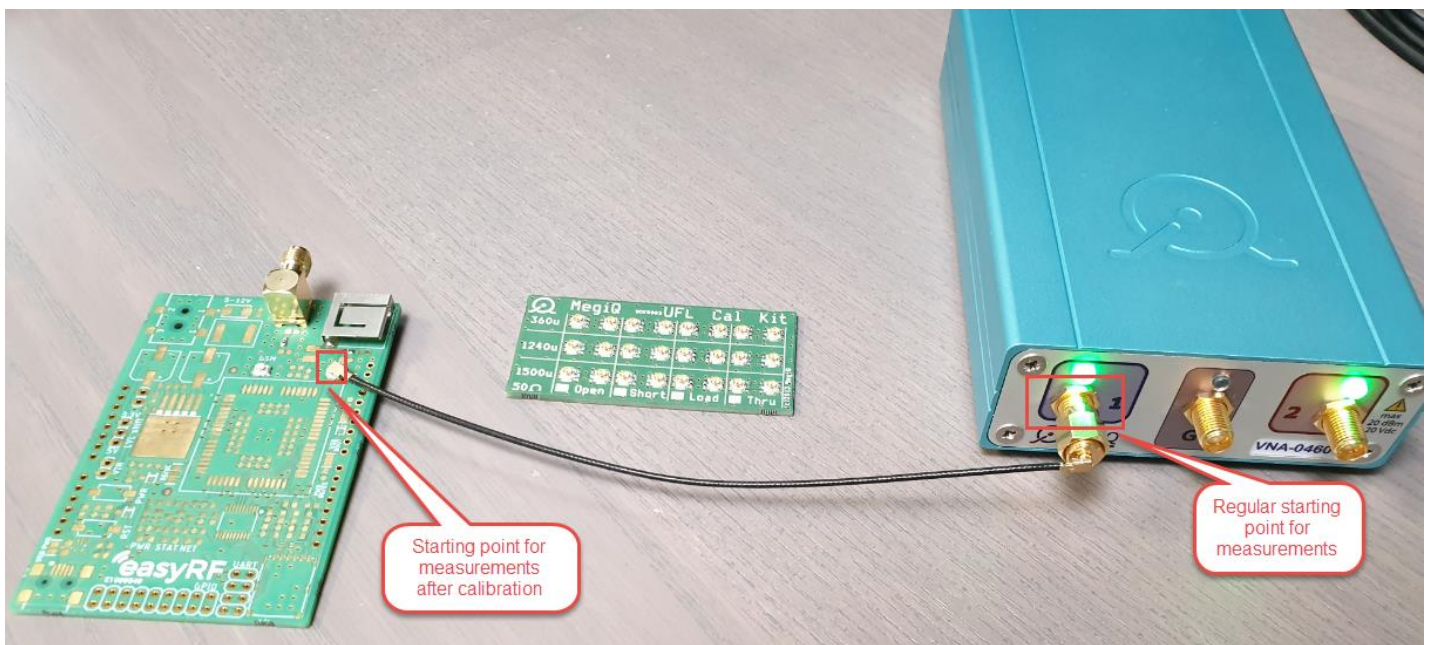
ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

3.3.1 Antenna design tuning

In this subchapter the process of tuning an antenna design will be described. For all antenna measurements a VNA supplied by our partner TOP-electronics is used. More info about the VNA can be found [here](#).

The measurements will be performed on the GNSS antenna of the ERF3000 shield. Please note, it is advised to perform the VNA measurements on an “Empty” PCB, this will result in the most accurate measurements.

The VNA has been calibrated with an Open, Short and Load PCB, supplied by the manufacturer of the VNA. By doing this calibration the “starting point” of the measurement has been moved to the U.FL connector on the ERF3000 PCB.



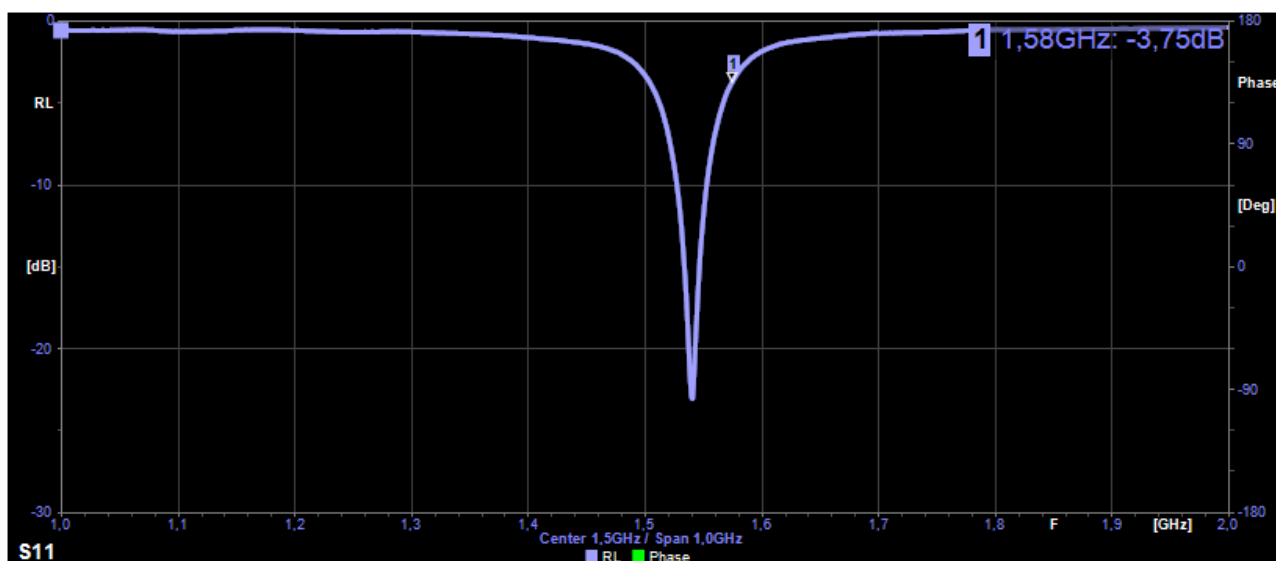
The VNA measurement will return 3 graphs, the Impedance, the return loss and the forward loss of the antenna circuit. These 3 graphs are used to interpret the total performance of the antenna circuit. Since the measured antenna is a GNSS antenna the best performance should be at 1575 MHz.

Keeping this in mind, the ideal Impedance should be: $50 \pm 0\Omega$, this will result in a return loss of $<-30\text{dB}$.

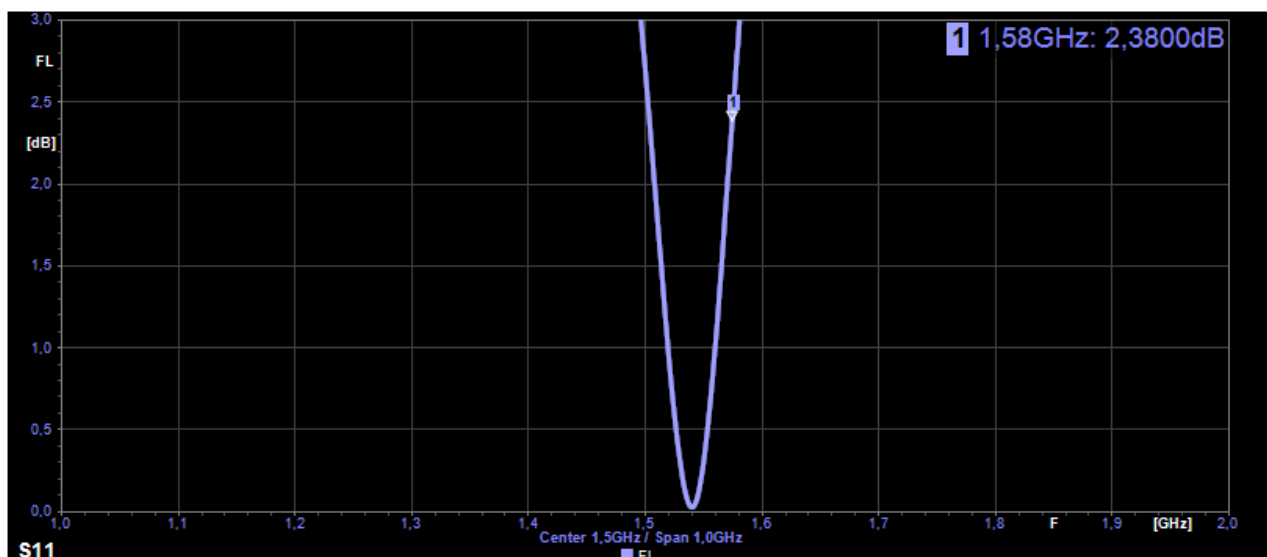


ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

As can be seen in the return loss graph, the return loss is currently -3,75dB. As a general rule of thumb, the -10dB line can be used as a reference point. Everything above the -10 dB line needs tuning, everything below doesn't need tuning but performance can be improved with tuning.



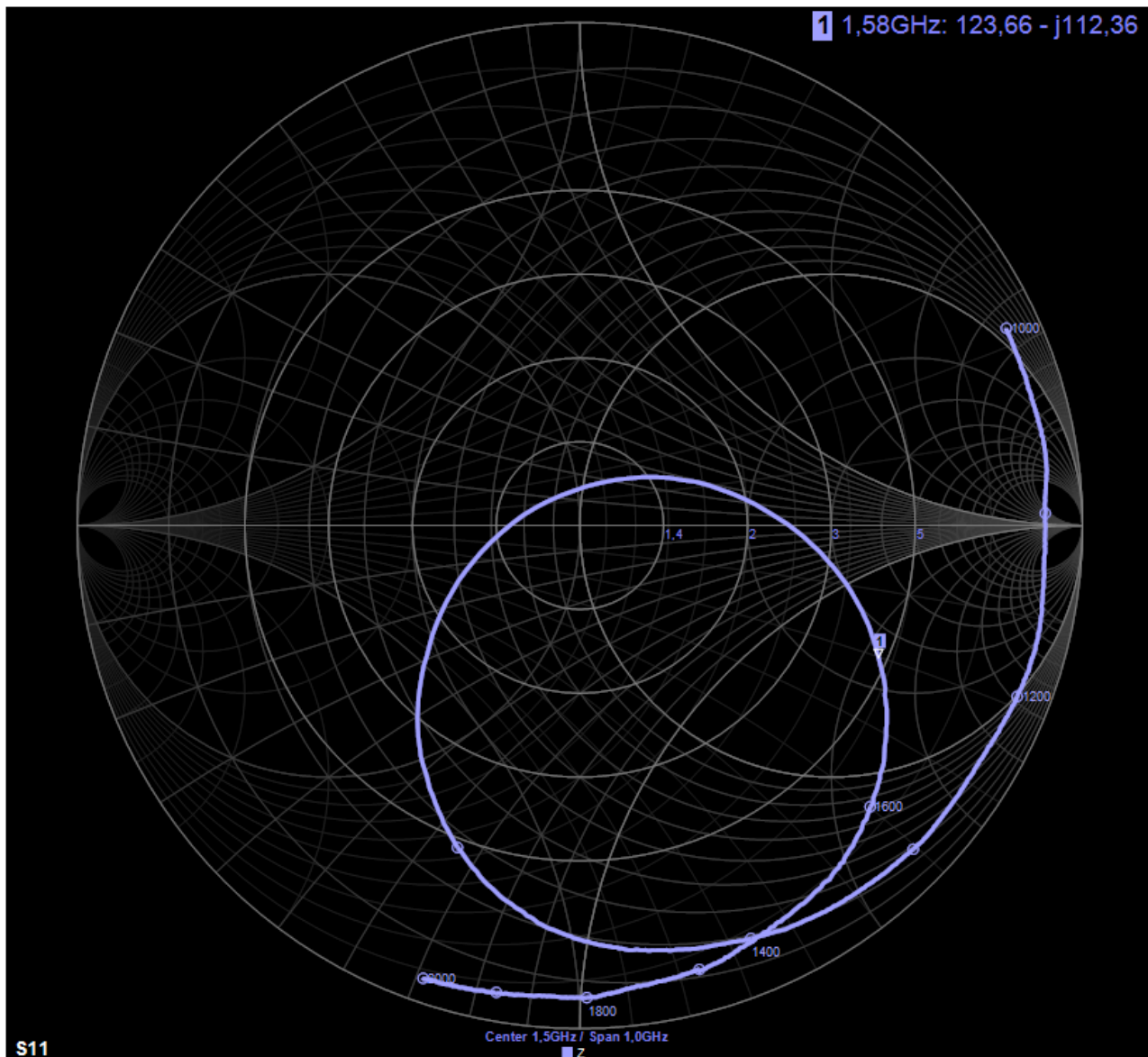
The Forward loss graph shows there is a loss of 2,38dB at 1575MHz. this is due to a mismatch in impedance. With a perfect match the forward loss will be 0dB.





ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

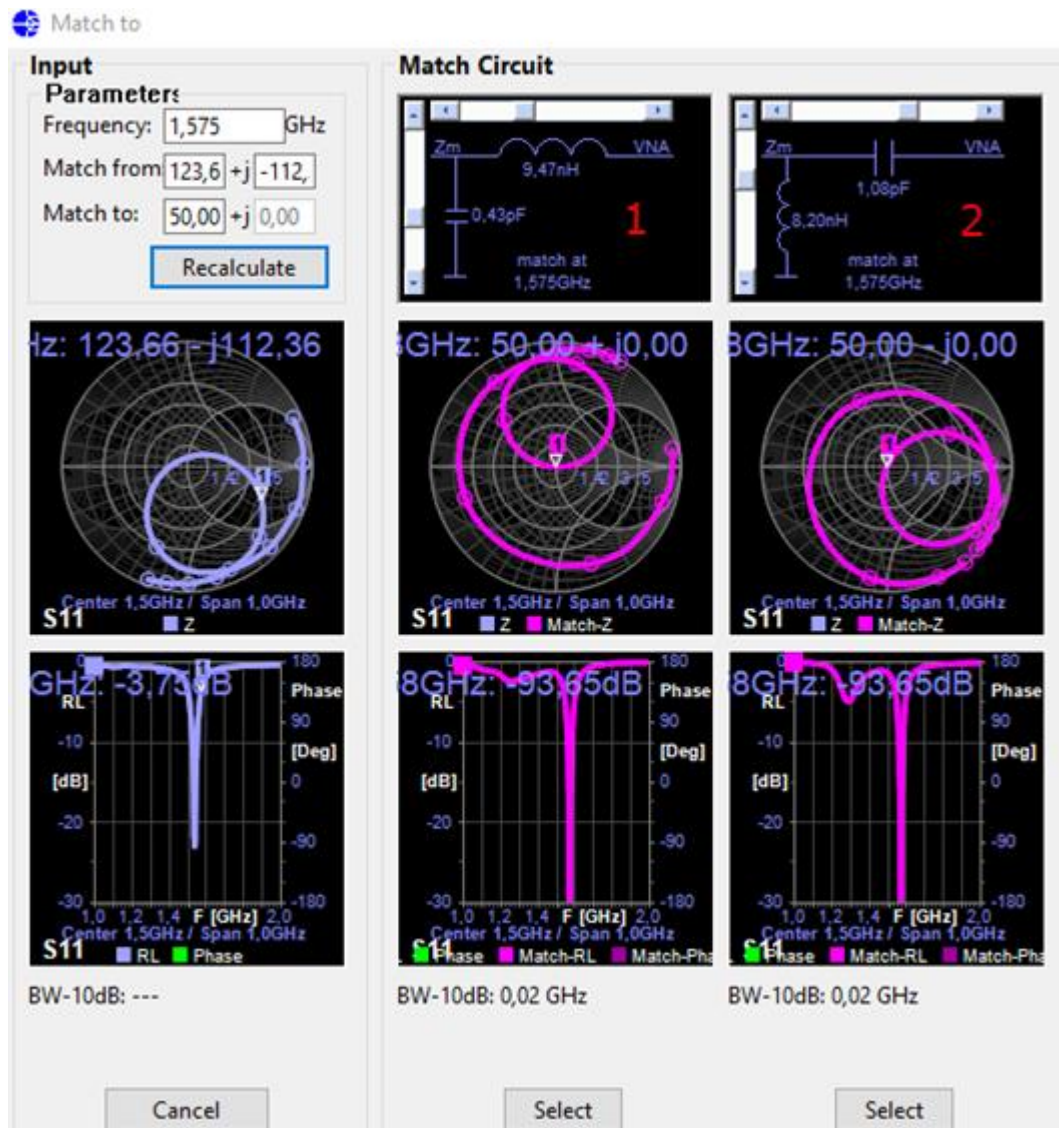
The Impedance of the antenna circuit is currently at $123,66 - j112,36\Omega$, instead of the ideal $50 \pm 0\Omega$. In order to match the impedance to $50 \pm 0\Omega$ a matching filter is required.



The matching filter can be calculated by hand, or the “Match Circuit” feature can be used. The Match circuit feature is a special tool made by the manufacturer and is integrated in the VNA software.

ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

When using the Match Circuit option is selected, the following pop-up will appear. There will be 2 or 4 options available to choose from:



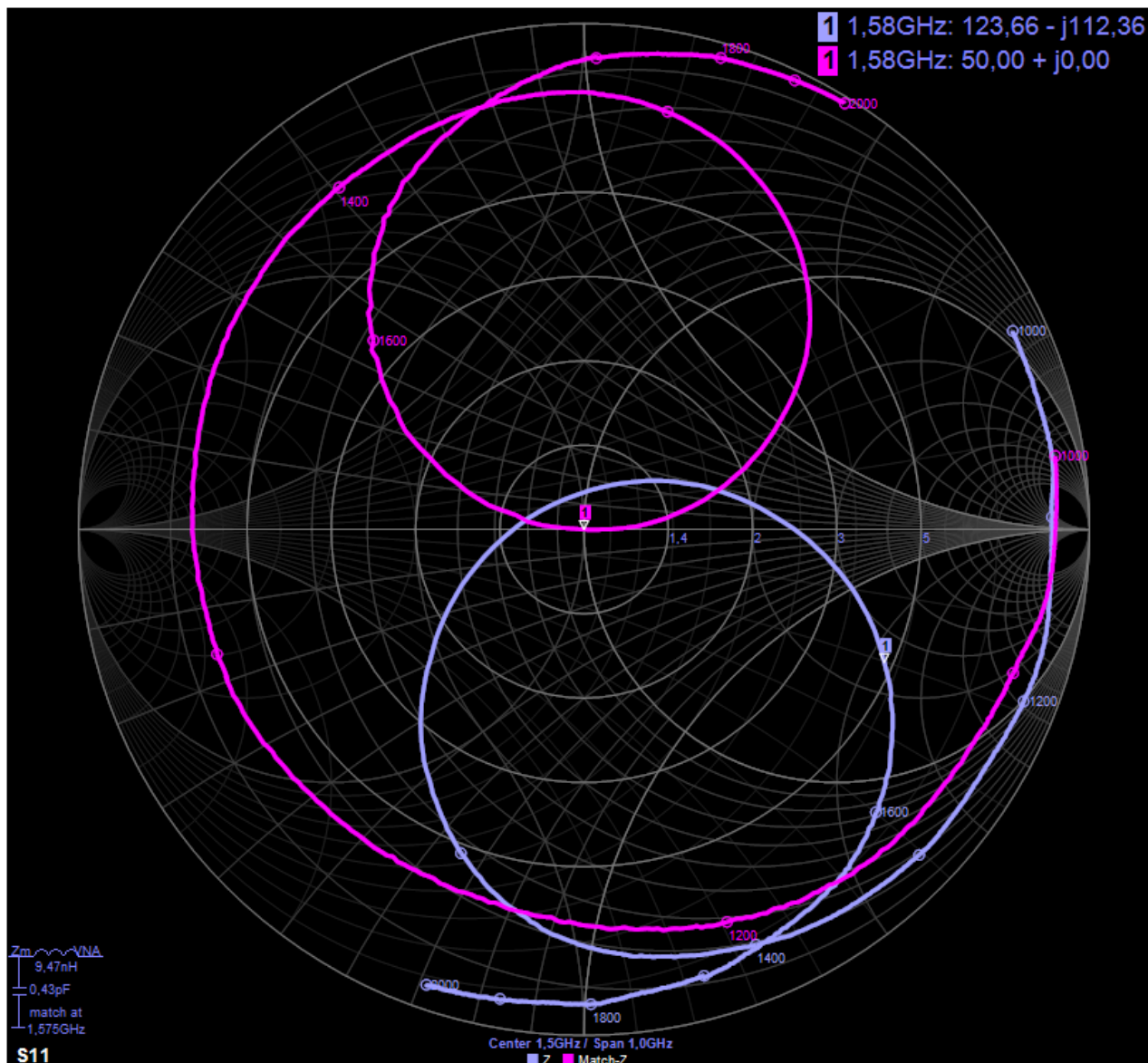
Matching filter 1 is a Low-pass LC filter, and filter 2 is a high pass LC filter.

The Low-pass filter has as benefit it will attenuate all frequencies higher than the cut-off frequency, including high frequency noise. So, this is in most cases the preferred filter to use.



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

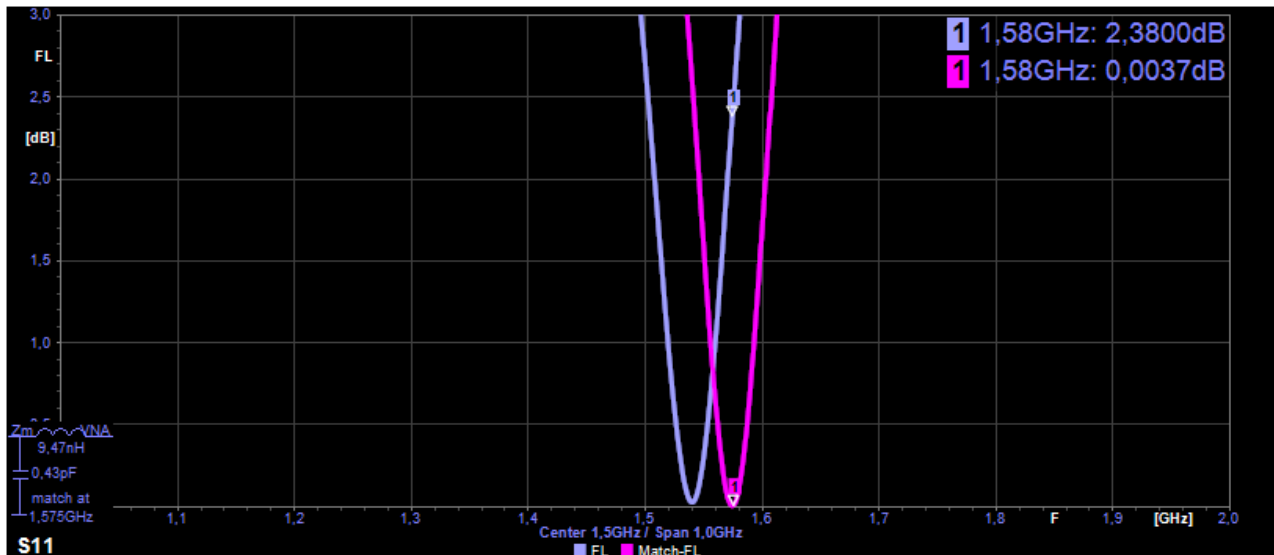
After selecting the matching filter, the matched graphs will be displayed. The Impedance is now matched to $50 \pm 0\Omega$.



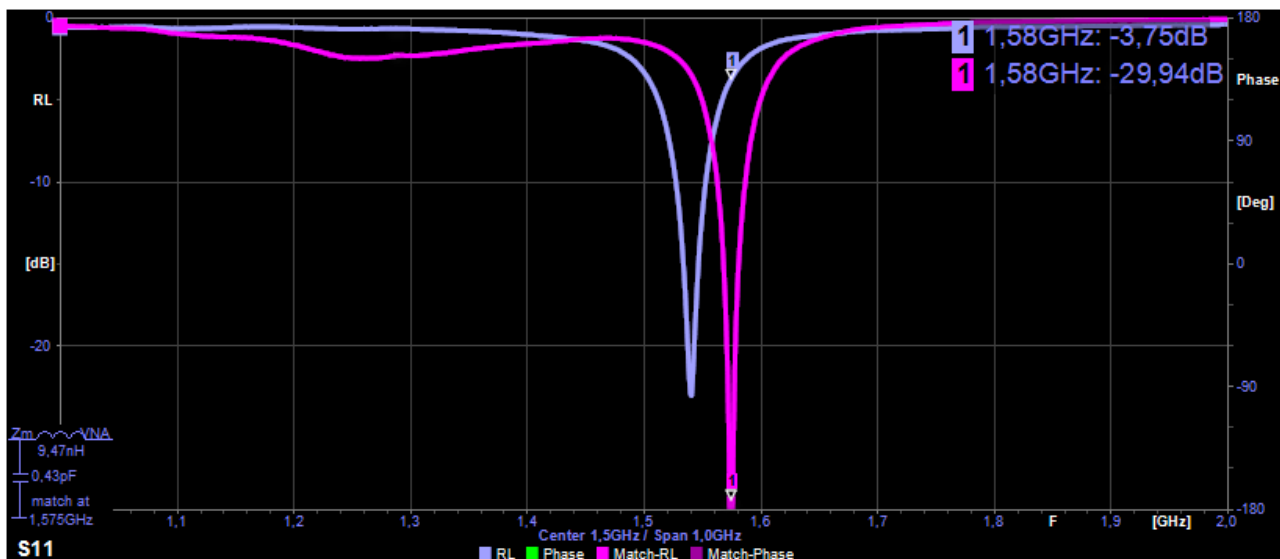


ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

The matched impedance results in a nearly ideal forward Loss of 0,0037dB.



And finally, a return loss of -29,94dB.



After matching the impedance the overall performance of the antenna circuit will be greatly improved.

These improvements includes reduced time to connection and power consumption, as can be seen in the measurements in [chapter 3.3](#).



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

4 Arduino Programming

4.1 Setup

To evaluate the shield with an Arduino board, the Arduino software is needed. This can be downloaded from the [Arduino website](#). After downloading the software, install it.

To communicate with the shield, a standard Arduino library called “SoftwareSerial” is needed. This is already installed with the Arduino IDE. Information about this library can be found [here](#).

When the Arduino software is installed, open the software.

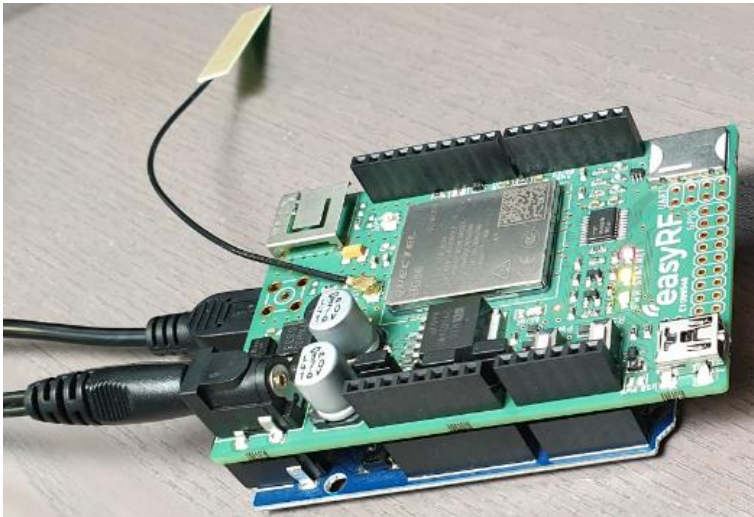
To speed up development and evaluation time, a few examples are provided on the [ERF3000 Github page](#).

In order to test the ERF3000 shield with an Arduino, place the ERF3000 on the Arduino headers, as can be seen in below image.

Now connect the following before inserting the USB cable in the Arduino:

- GSM antenna to the ERF3000 shield
- Insert SIM card
- Connect DC Power supply to the ERF3000 shield

When all above steps are followed, the USB connector can be inserted in the Arduino board.



Verify the Arduino board is recognized by the PC/Laptop, and upload an example of your choosing to the Arduino board. When opening the COM port in the Arduino software, make sure to put it on 9600 baud.



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

4.2 Examples

The Examples provided are:

Name	Special function	Description
BG96_KPN_NL_CAT-M_HTTP.ino	HTTPPOST	Connect to KPN CAT-M network, Post message via HTTP and enter PSM
BG96_KPN_NL_CAT-M_TCP.ino	TCP	Connect to KPN CAT-M network, Send message via TCP and enter PSM
BG96_T-Mobile_NL_NB-IoT_UDP.ino	UDP	Connect to T-Mobile NB-IoT network, Send message via UDP and enter PSM
BG96_VodafoneZiggo_NL_NB-IoT_UDP.ino	UDP	Connect to VodafoneZiggo NB-IoT network, Send message via UDP and enter PSM
BG96_GNSS.ino	GNSS	Start the GNSS service and get a fixed location with 1Hz interval. Please do note the GNSS doesn't work indoors, and it could take some time until a fixed location is retrieved!



ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

5 Measuring the current consumption

5.1 Preparations

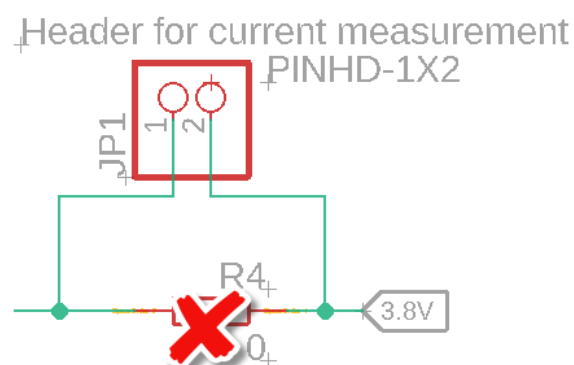
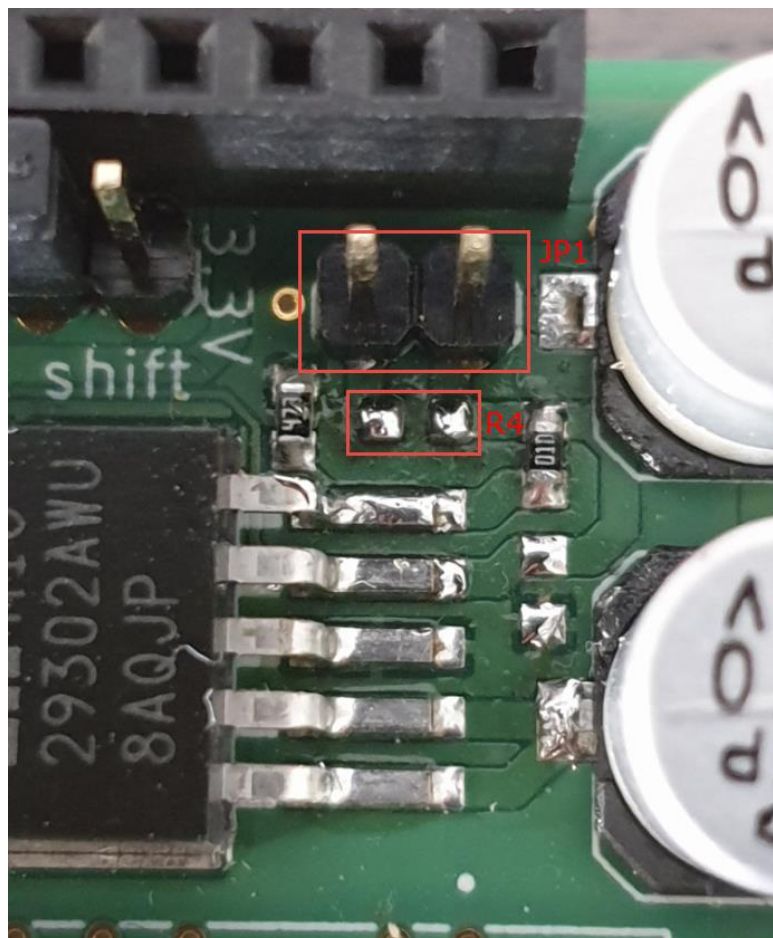
In order to perform an accurate current measurement over the ERF3000 shield, the following preparations need to be made:

- Desolder resistor R4
- Solder Header JP1

The locations of these parts can be found in the below image.

A current measurement can also be performed at the DC jack of the shield, but please take into account the loss of the LDO circuit, and the consumed current by the LED and Level shifter.

When the current is measured at header JP1, only the consumed current of the Quectel BG96 module is measured.

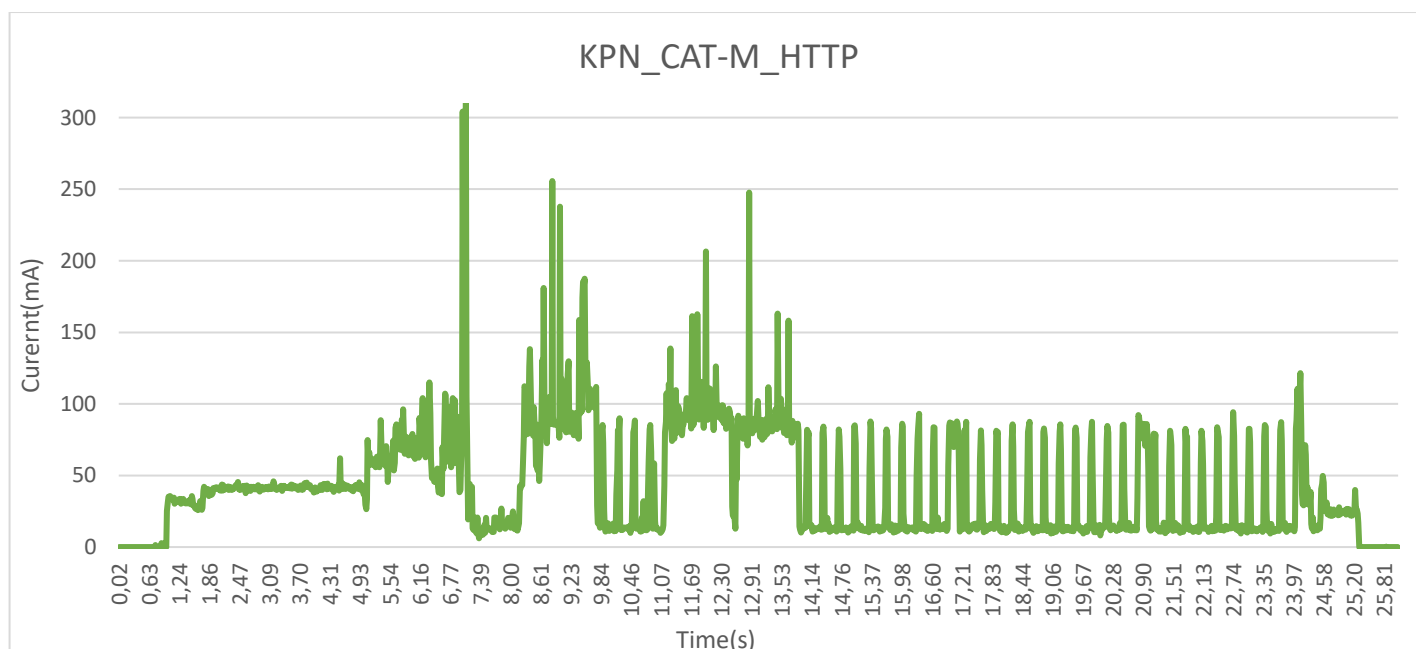
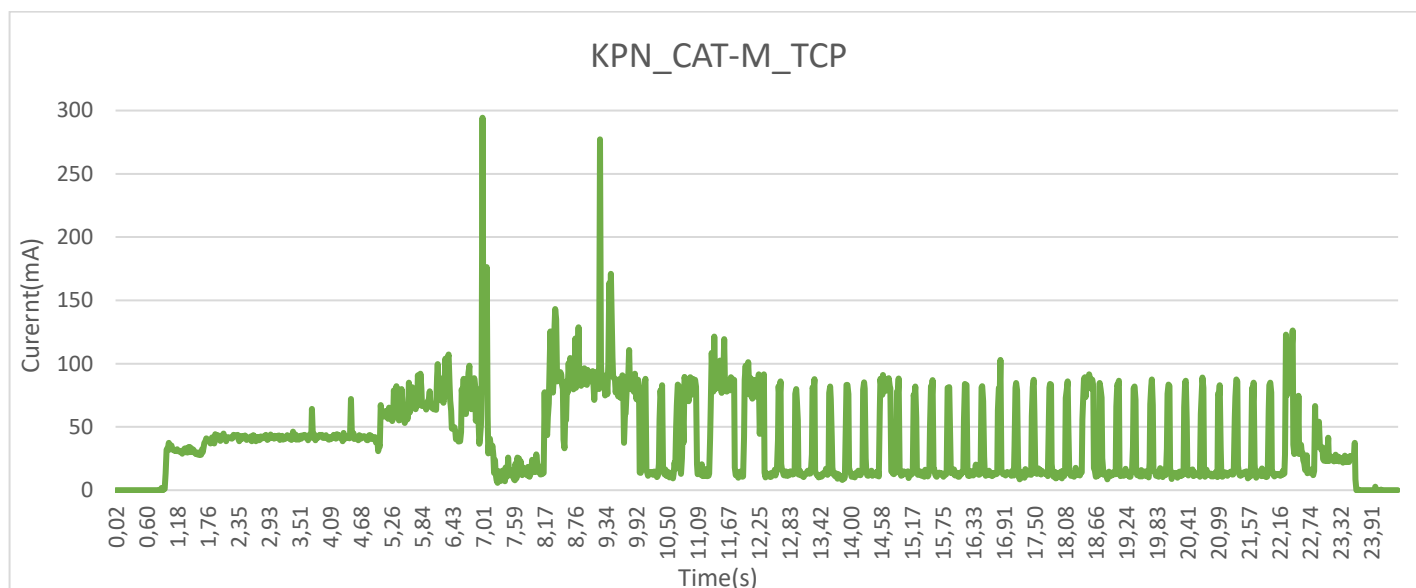




ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield

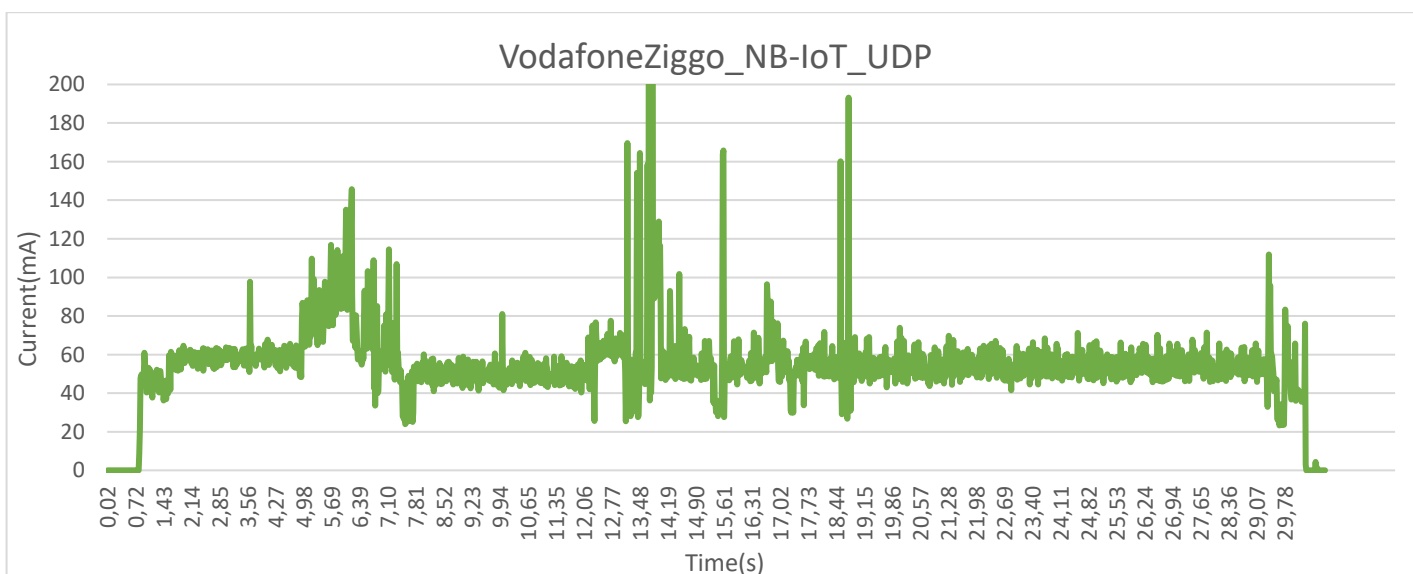
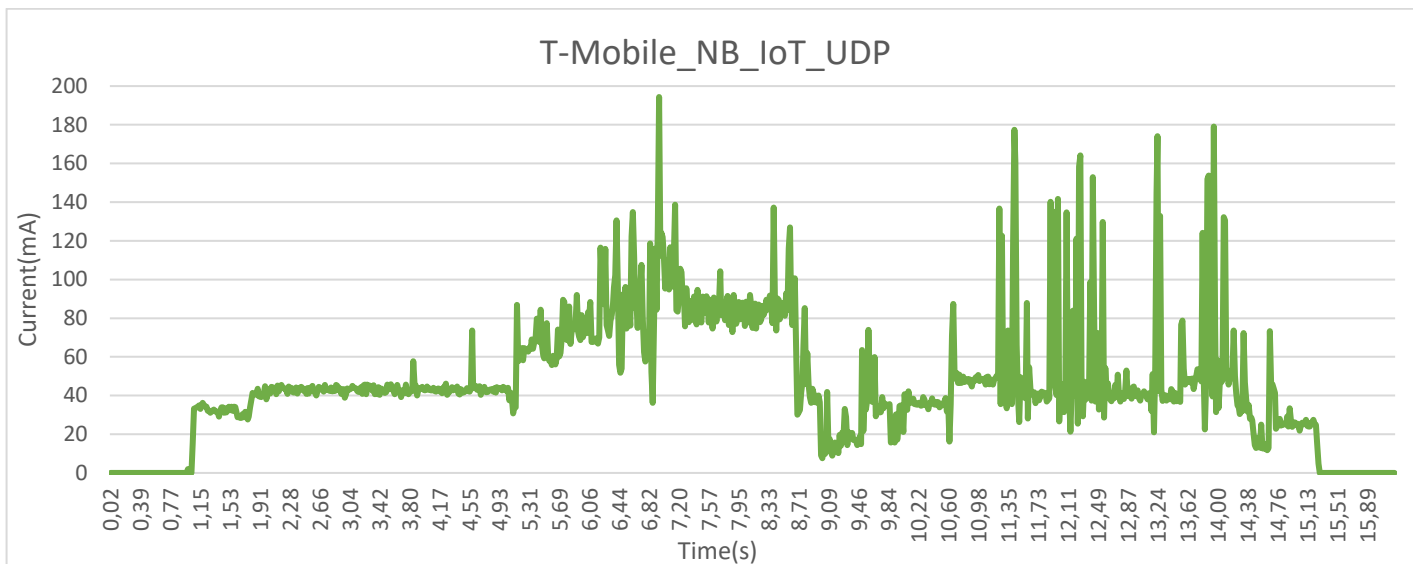
5.2 Results

Now following are a few test results of current measurements performed on the ERF3000v2 shield. The software used are the Arduino examples provided on the [ERF3000v2 Github page](#).





ERF3000v2, LTE Cat. M1, Cat.NB1, EGPRS & GNSS Arduino shield



Measurement	Total time (s)	Average current (mA)
KPN_CAT-M_TCP	24,41	37,76
KPN_CAT-M_HTTP	26,09	41,20
T-Mobile_NB_IoT_UDP	16,24	46,32
Vodafone_NB_IoT_UDP	30,8	53,85

Please note, your results may vary from above results, the consumed current is dependent on signal strength, providers, etc.. these measurements only function as example.