MMCOMMANDER

What's this?

An interface for Medtronic CGM devices.

I've created this document to help you build and configure your MMCommander for your Medtronic device.

Note - currently MMCommander only supports the Medtronic MiniMed Veo pump with Guardian Enlite CGM system. Unfortunately the Medtronic MiniMed 640G pump with Guardian 2 CGM system is not supported yet.

MMCommander works by 'sniffing' the CGM readings while they are being sent wirelessly from the CGM transmitter to the pump. Then it sends that data to an Uploader App running on an Android phone, which in turn sends the data (either via Wi-Fi or the Mobile Phone Network) to a Nightscout website in the cloud.

Oops... I nearly forgot.

As you can imagine, Medtronic is not related to this project in any way and the author (me) assumes no responsibility for any damage arising from use of any of the information found in this repository.

USE IT AT YOUR OWN RISK.

Information obtained via this software MUST NEVER BE USED to take medical decisions.

What do I need?

Apart from the MMCommander, to use Nightscout you'll need an Android phone that supports USB OTG and a suitable USB-OTG cable (Micro-USB male plug to regular USB female socket) to connect to the MMCommander transceiver.

Check the Nightscout webpage http://www.nightscout.info/wiki/welcome/basic-requirements
and the "CGM in the Cloud" Facebook group https://www.facebook.com/groups/cgminthecloud/
for more info about what you need.

Bear in mind that Nightscout originated using the Dexcom CGM system, so lots of the information on the Nightscout Website relates to this.

The MMCommander is composed of three things:

1) CC1111 USB Dongle

http://www.ti.com/tool/cc1111emk868-915

This is a USB RF transceiver dongle made by Texas Instruments, and is used to 'snoop' on the data packets being sent wirelessly over-the-air between the Medtronic CGM and the pump.

2) CC Debugger

http://www.ti.com/tool/CC-DEBUGGER

The CC Debugger is a device which you'll only need in order to download software to the CC1111 (and after that you can share it with others if you want).

The cost of the whole MMCommander pack at October 2015 is:

- $75 for the CC1111 on TI's website (also available from other distributors)
- $49 for the CC Debugger on TI's website

3) MMCommander software

Lastly, you need the software found in this repository.

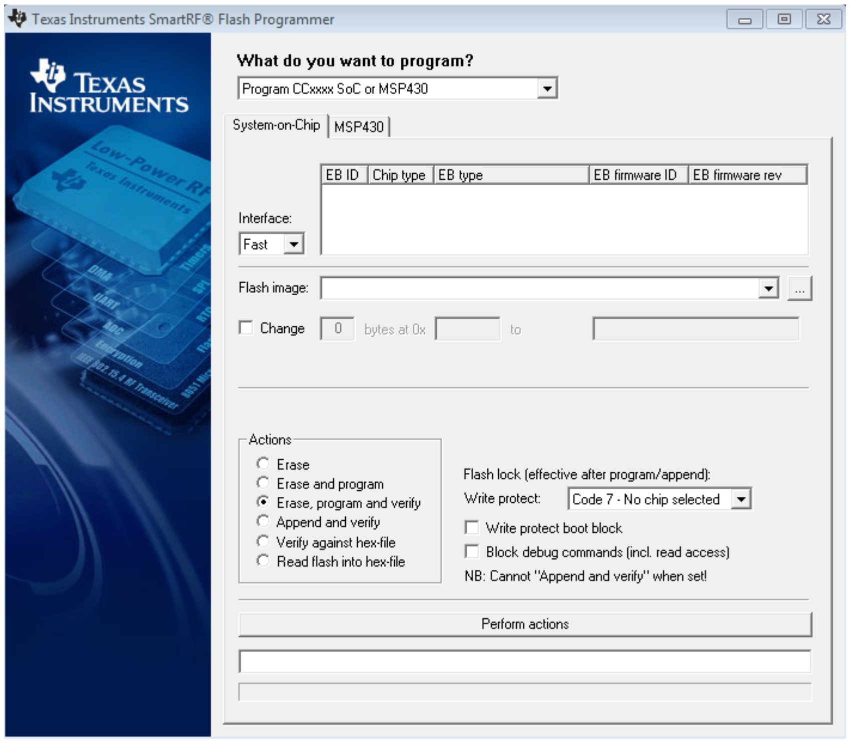
Now that I have the hardware... What do I do now?

Now that you have the hardware, you'll need to install (on a Windows computer) the software that Texas Instruments provides to program the CC1111.

It's called SmartRF Flash Programmer and can be downloaded from here: http://www.ti.com/tool/flash-programmer

Note - there are now two versions of this software. You need to choose the "SmartRF Flash Programmer" (which is for older 8051 based modules like the CC1111), and not "SmartRF Flash Programmer 2" (which is for newer ARM based modules).

It should look something like this.



What do you want to program? Program CCxxxx SoC or MSP430

Interface: Fast

Change 0 bytes at 0x

Actions: Erase, program and verify

Flash lock (effective after program/append):

Write protect: Code 7 - No chip selected

NB: Cannot "Append and verify" when set!

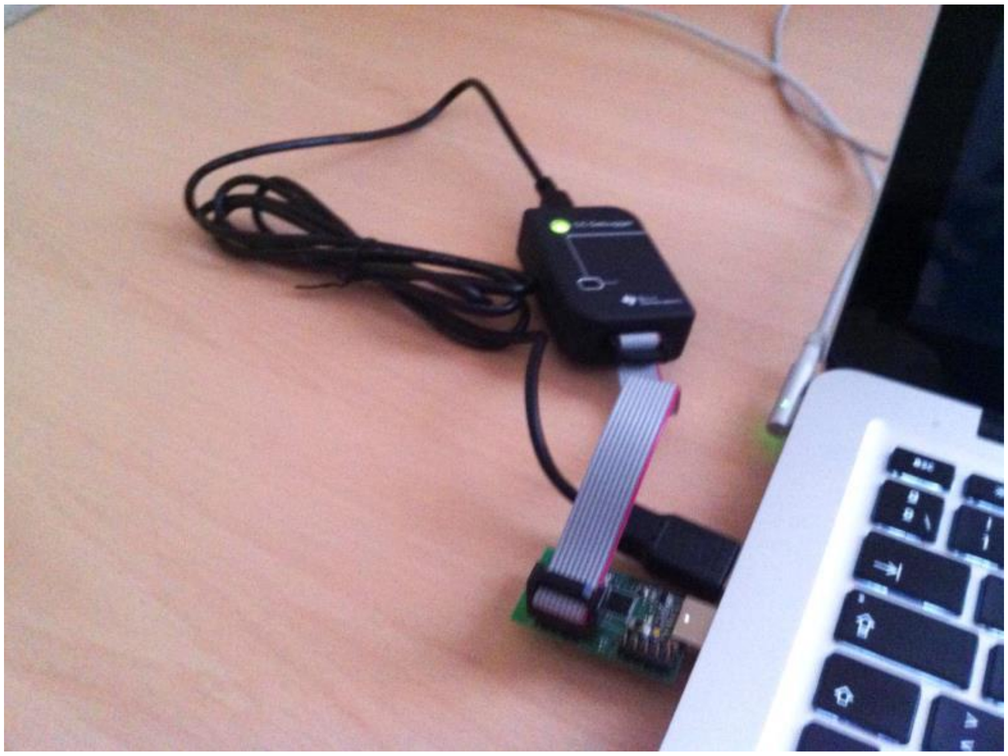
Connect the CC Debugger and CC1111

Connect the CC Debugger to the CC1111 board using the ribbon cable that comes with the Debugger (connecting to CC1111's "Debug" connector, not the "Test" connector).

Pay close attention to the red line on the ribbon cable marking pin 1: connect the cable as you can see in the following picture.

Note: some people have reported their cables were incorrectly assembled and they had to connect the ribbon cable with the red line facing the opposite side. Some people have even found the cable would not work whichever way it was connected (if that's you, see the Troubleshooting section for help). Don't be afraid to test, you won't break anything.

Plug both CC Debugger and CC1111 USB connectors into your computer.

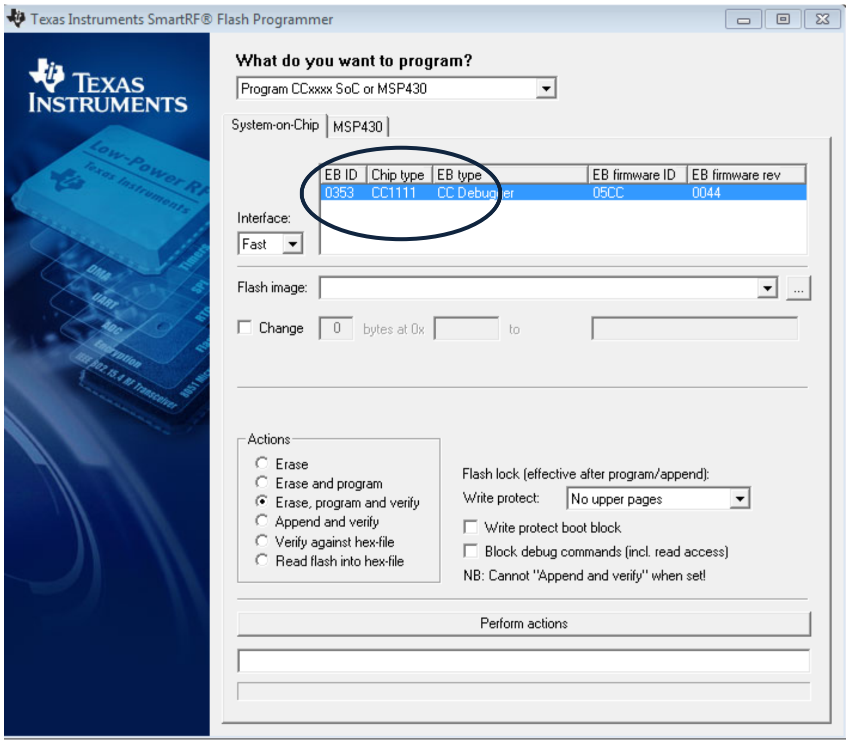


The CC Debugger should be detected by Windows and its USB driver should be installed automatically.

Note - Some CC Debuggers are shipped with an old Firmware version and the SmartRF Flash Programmer will ask you if you want to update it. You can do it if you want, but it's not mandatory. I would only do it if you have problems programming the CC1111. Up to you. (See the section near the end of this document for how to do this).

Now check the LED on the CC Debugger. If the LED is red, detection of the CC1111 failed - press the reset button on the CC Debugger and the LED should turn green.

If the LED is green, the Debugger has successfully detected the CC1111, and the CC1111 should appear in the devices list:



Texas Instruments SmartRF® Flash Programmer

What do you want to program?

Program CCxxxx SoC or MSP430

System-on-Chip | MSP430

EB ID	Chip type	EB type	EB firmware ID	EB firmware rev
0353	CC1111	CC Debugger	05CC	0044

Interface: Fast

Flash image:

☐ Change 0 bytes at 0x to

Actions
- ○ Erase
- ○ Erase and program
- ● Erase, program and verify
- ○ Append and verify
- ○ Verify against hex-file
- ○ Read flash into hex-file

Flash lock (effective after program/append):

Write protect: No upper pages

☐ Write protect boot block

☐ Block debug commands (incl. read access)

NB: Cannot "Append and verify" when set!

Perform actions

Note - The CC1111 only appears in the devices list if the CC Debugger is correctly connected with the ribbon cable (green LED). It does not appear here just because the CC1111 is plugged into a USB port, or when the CC Debugger LED is red.

If the CC Debugger LED is still red, something is not correctly connected. Check the ribbon cable, its orientation, and that both USB ports are properly connected to the computer (both CC Debugger and CC1111 need USB power). Take a look again at the earlier picture to check.

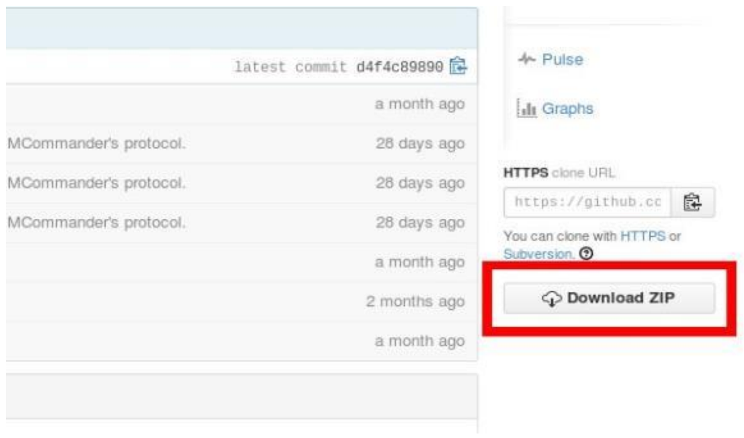
Note - if the LED is amber or flashing red, that means something is wrong with the Debugger itself. See TI's CC Debugger User Guide for help: http://www.ti.com/lit/pdf/swru197

Download the MMCommander software

Is it working? Well... let's download the MMCommander software (if you haven't already).

If you're not used to downloading info from GitHub I would recommend downloading the complete repository in a ZIP file.

Go to the repository's main page http://github.com/jberian/mmcommander and click on the “Download ZIP” button on the right side of the page.


latest commit d4f4c89890
a month ago
MCommander's protocol. 28 days ago
MCommander's protocol. 28 days ago
MCommander's protocol. 28 days ago
a month ago
2 months ago
a month ago
Pulse
Graphs
HTTPS clone URL
https://github.cc
You can clone with HTTPS or Subversion.
Download ZIP

Decompress the ZIP file and, in the “hex” folder, you'll find two HEX files:

- MMCommander_EUR_0.89_NoTx.hex
- MMCommander_US_0.89_NoTx.hex

These files are the result of compiling the sources for US and EUR frequencies WITHOUT transmission (Tx) enabled. This is the safest implementation possible and using these binaries you won't be able to affect in any way your pump or sensor. You'll be a passive listener and will be able to listen to your sensor and glucometer. When configuring the Android Uploader for Medtronic you'll see options like obtaining the calibration factor from the pump or history retrieving... However these options won't work since you won't be able to talk to your pump or Medtronic receiver, ok?

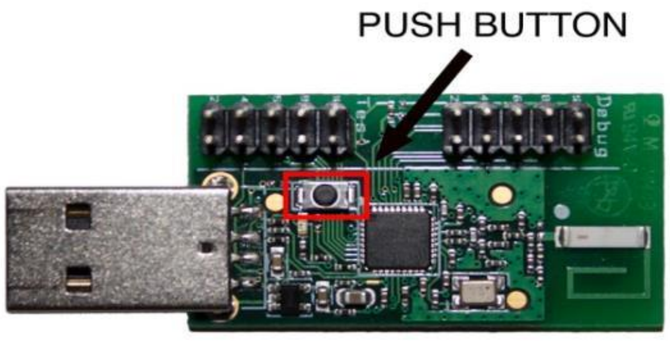
If you want to do more testing or enable other features for which you need to enable the transmission part, install the IAR Embedded Workbench for 8051, from here: https://www.iar.com/iar-embedded-workbench/downloads/

Ask for a 30 day evaluation license and compile the sources with the options you need enabled. Check the configuration header file for that. Be careful when doing tests.

The MMCommander, in any version with transmission enabled, is capable of receiving and TRANSMITTING any string. What would happen if someone hacks your smartphone/computer and sends commands to your pump?... for instance... a bolus command. Would be dangerous, right? That's why I implemented what I called the "transmission filter". This filter won't let you (or your hacker) send some certain types of commands so that its use is safer.

And what's the suspend support? Yes... It's a transmission filter that will allow you to send suspend commands. This will enable you/us to implement Medtronic's "Low Glucose Suspend" if you need it.

And what happens if I want to have the Transmission Filter but, in certain cases, I want to send "forbidden" commands? No problem. The CC1111 USB board has a push button and a green LED.



If you press the push button for a couple seconds the LED will turn on and then the filter will be disabled. Whenever you have the green LED on you'll be able to send whatever you want. After that, if you want the filter back, just press the push button again for a couple seconds until the LED turns off.

Then… Which one do I need? If you need to transmit I would recommend to use the one with suspend support for your zone, if not, the one without transmission. Again… at your own risk, remember?

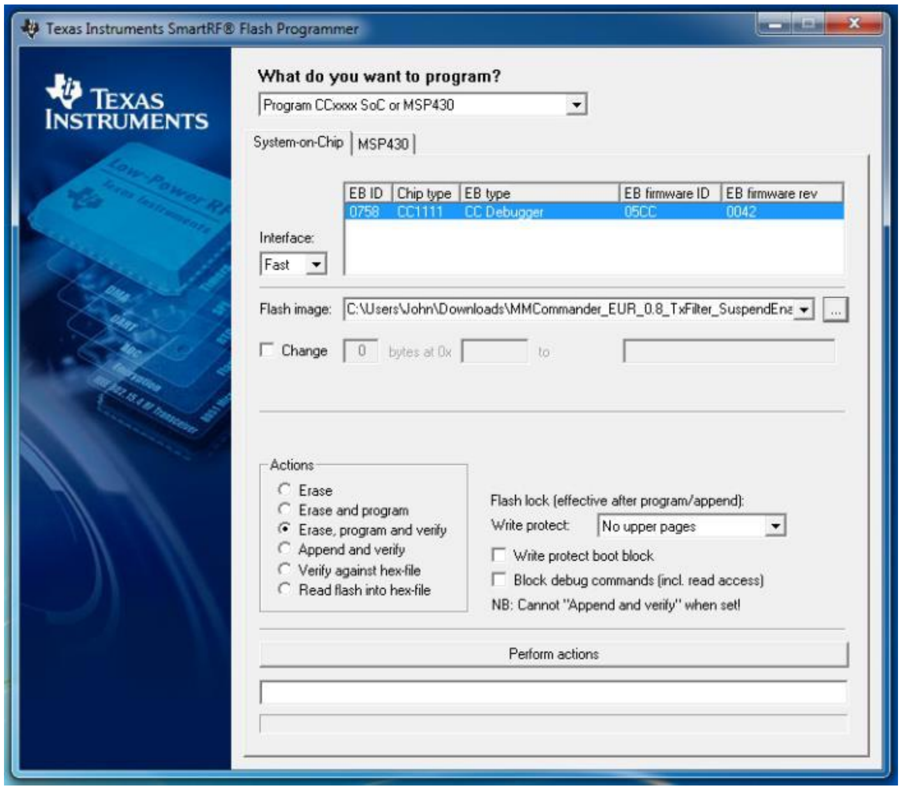
Now that you know which one is the one you need, get the HEX file and let's go back to the SmartRF Flash Programmer and program the CC1111.

Programming MMCommander into the CC1111

Now everything is connected, the LED on the CC Debugger is green... It's time to configure the SmartRF Flash Programmer to program the CC1111.

Check list:

- Select 'System-on-Chip' tab
- If everything is correctly connected you should see the CC1111 in the devices list
- In "Flash image" click the button with the 3 dots and select the MMCommander HEX file you downloaded
- In actions: choose Erase, program and verify



What do you want to program? Program CCxxxx SoC or MSP430

EB ID	Chip type	EB type	EB firmware ID	EB firmware rev
0758	CC1111	CC Debugger	05CC	0042

Interface: Fast

Flash image: C:\Users\John\Downloads\MMCommander_EUR_0.8_TxFilter_SuspendEna

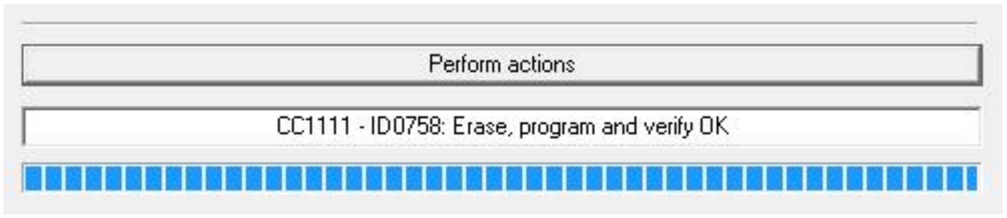
☐ Change 0 bytes at 0x to

Actions
- ○ Erase
- ○ Erase and program
- ● Erase, program and verify
- ○ Append and verify
- ○ Verify against hex-file
- ○ Read flash into hex-file

Flash lock (effective after program/append):
Write protect: No upper pages
☐ Write protect boot block
☐ Block debug commands (incl. read access)
NB: Cannot "Append and verify" when set!

Perform actions

After configuring the programmer, click on "Perform actions" and wait until you see this:



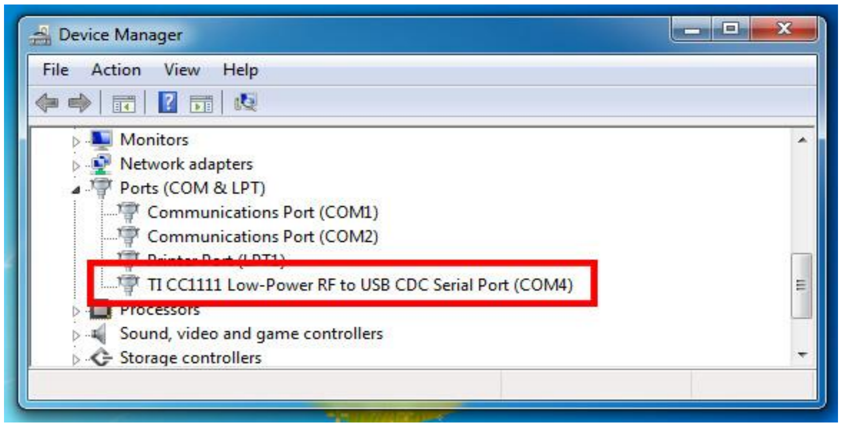
Perform actions

CC1111 - ID0758: Erase, program and verify OK

At this moment your computer should detect a new "MMCommander" device.

Install the MMCommander driver found in the “Windows_Driver” folder and check the Device Manager in your Control Panel.

You should see a new COM port like the one in the next image.


Device Manager
File
Action
View
Help
Monitors
Network adapters
Ports (COM & LPT)
Communications Port (COM1)
Communications Port (COM2)
Printer Port (LPT1)
TI CC1111 Low-Power RF to USB CDC Serial Port (COM4)
Processors
Sound, video and game controllers
Storage controllers

In Windows 8, you won't be able to install the driver directly because it's not signed. Don't worry. You can skip this step. It's only necessary if you plan to develop with it on a Windows PC.

To test, just plug the MMCommander into a USB port and hold the pushbutton for a few seconds. If the LED turns on and, after pressing again for another few seconds, it turns off then it's ready ;-)

Congratulations! You have a new MMCommander!

Nightscout setup

Now continue the process of configuring the data backend as described in the Nightscout website: http://www.nightscout.info/wiki/welcome/configuring-the-data-backend

Once the data backend is ready, install the Android Uploader for Medtronic (instead of the one for Dexcom mentioned in the Nightscout website) onto your phone: https://github.com/arbox0/MedtronicUploader

After that continue the process for the Azure website, again, as described in the Nightscout website: http://www.nightscout.info/wiki/welcome/monitor-cgm-web

REMEMBER, USE IT AT YOUR OWN RISK

Some notes on configuring the Android Uploader for Medtronic

You have set up the Mongo database and installed the Medtronic Uploader... how do I configure the uploader? Let's do a step by step configuration...

1. Close the uploader app if it's opened and connect the MMCommander with the USB OTG cable to the phone. You should see a pop-up asking you what to do with it. Select the Nightscout app and it should open automatically. If not, something is wrong with your setup.

2. The uploader is running. Open the preferences menu and configure the uploader:

- mmol/L -> Select the unit you want to use.
- Type -> Select Medtronic CGM
- Pump ID -> The ID on the back of your pump
- Glucometer ID -> The same on the glucometer
- Sensor ID -> The ID on the back of the Enlite transmitter. (Also configured in your pump)
- Calibration Type -> Choose “Manual”.
- Glucose Value Source -> Choose “Medtronic Sensor”
- Activate the “MongoDB uploading”
- MongoDB URI -> Put here the address you got when you created your database.
- Collection Name -> write “cgm”
- Glucometer Data Collection Name -> write “gcvalues”
- Wifi Hack -> Set it to OFF
- 2 Days at Startup -> Set it to ON
- Disclaimer -> You have to accept it and set it to YES. The uploader won't start the service until you accept it.

3. Go back to the main screen. You should see dashes, a green text saying “CGM Service Started” and another text saying “Uncalibrated”.

4. Each time the uploader receives data from the sensor you'll see a text line in the log (lower part of the screen) that says: “Medtronic CGM Message: sensor data value received”. Don't do anything else until you see that message for the first time. If you don't see that line in 10 minutes go back and repeat the configuration. The transmitter sends a message each 5 minutes (more or less). I've added a section on debugging the MMCommander at the end of this document in case you don't receive anything.

5. You have received the first data from the sensor.

6. Now it's time to calibrate... Select “Instant Calibration” and enter the BG value display on your pump. After that, on the main screen, you should see your actual BG and that it's calibrated.

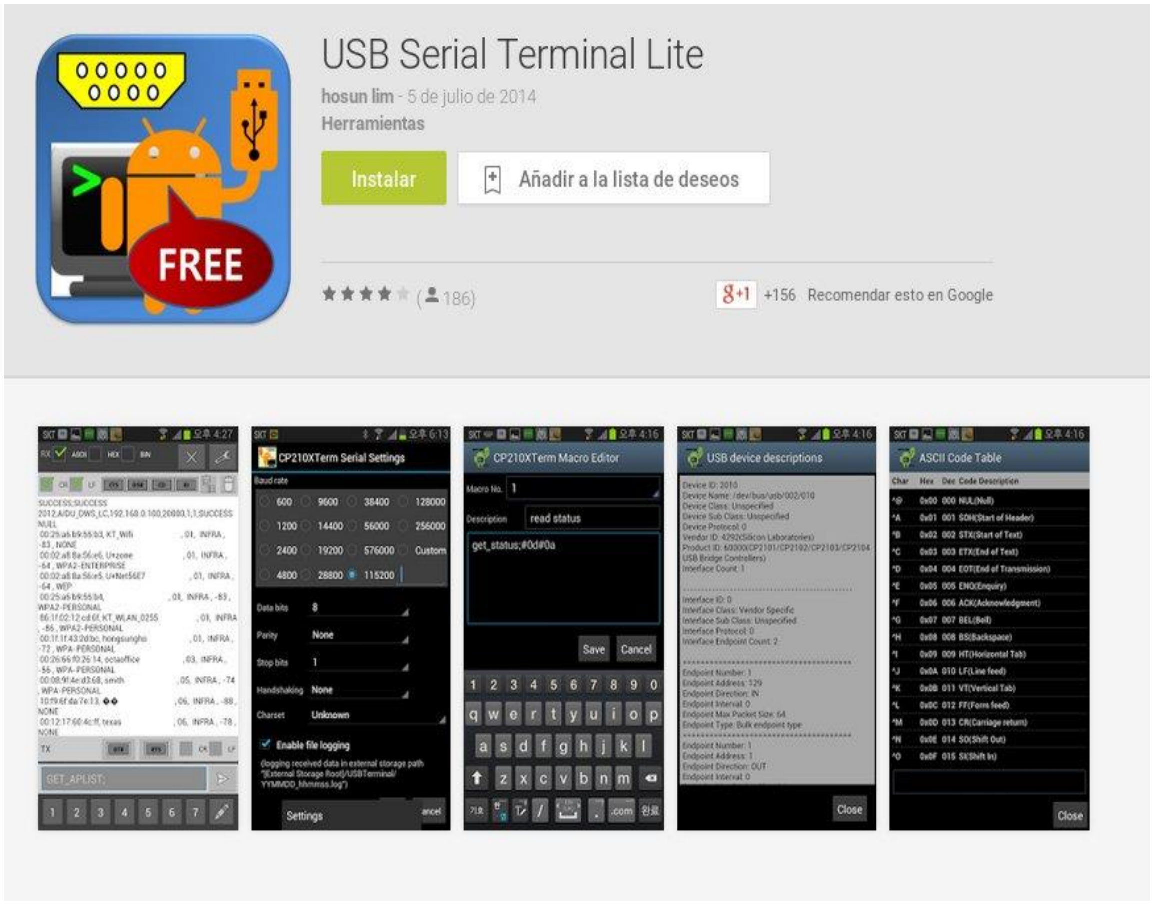
7. At this point, your Mongo database should start recording data from your uploader. If you want, you can check it looking at the number of documents stored in the “cgm” collection. Each time a sensor message is received the number of documents should increase (one unit).

8. As far as I know, you need to have at least 3 or 4 values before the Azure website starts displaying info.

How can I test if my phone and the MMCommander are receiving data from the Enlite transmitter?

If you're reading this it means you're probably having problems with the Medtronic Uploader and don't know how to continue... well... Let's see if your phone and MMCommander are capable of receiving data.

We are going to use an application named “USB Serial Terminal Lite”. Look for it in the Play Store and install it.


USB Serial Terminal Lite
hosun lim - 5 de julio de 2014
Herramientas
Instalar
Añadir a la lista de deseos
(186)
+156 Recomendar esto en Google
FREE

Once it's installed, connect the MMCommander with the USB-OTG cable to your phone and start the application. If the Medtronic Uploader launches itself automatically when you connect the MMCommander, press “Stop Uploading CGM Data” and go back to the Terminal.

1. Click on the configuration icon on the right top corner and configure the serial port.
2. Baudrate: 57600
3. Data bits: 8

4. Parity: None
5. Stop bits: 1
6. Handshaking: None
7. Charset: Unknown
8. Enable file logging: unchecked
9. Press “Save”.
10. After pressing “Save” you'll go back to the main screen.
11. Select “HEX” format on the top bar.
12. Press on the phone icon on the top right corner and it should transform into an “X” icon.
13. Select “RTS” at the bottom. They should highlight in green.

Now you should be able to see in hexadecimal format everything that is being received by the MMCommander.

First, let's check that the MMCommander was programmed correctly. Press the pushbutton until the green LED turns on. Once the LED is on, you should see in the terminal “13”. If you press the pushbutton again, the LED should turn off and you should see “03” in the terminal. If this happens, the phone, the OTG cable and the MMCommander are working.

Now let's check if the RF part of the MMCommander is working...

Take your pump and go to “Utilities” -> “Connect Devices” -> “Other Devices” -> “On” -> “Find Devices”.

You should be able to see some data on screen starting with “02 0B A2...” in periods of some seconds.

If you see these bytes, your MMCommander is working perfectly. As the Enlite transmitter is sending data each 5 minutes, it may be possible to see the info apart from the bytes from the pump. Anyway, if you see data from the Enlite transmitter (also starting with “02” or “82”) it's working.

If you can see the “03” and “13” bytes when turning on and off the LED, but don't receive anything else, try putting the MMCommander really really close to the pump. If you receive data when near, you have a RF problem and you have 2 options: resoldering the RF parts (ask in the group) or buy another CC1111 stick.

If it doesn't work with that, reprogram it. Maybe you selected the wrong zone by mistake.

Hope it helps ;-)

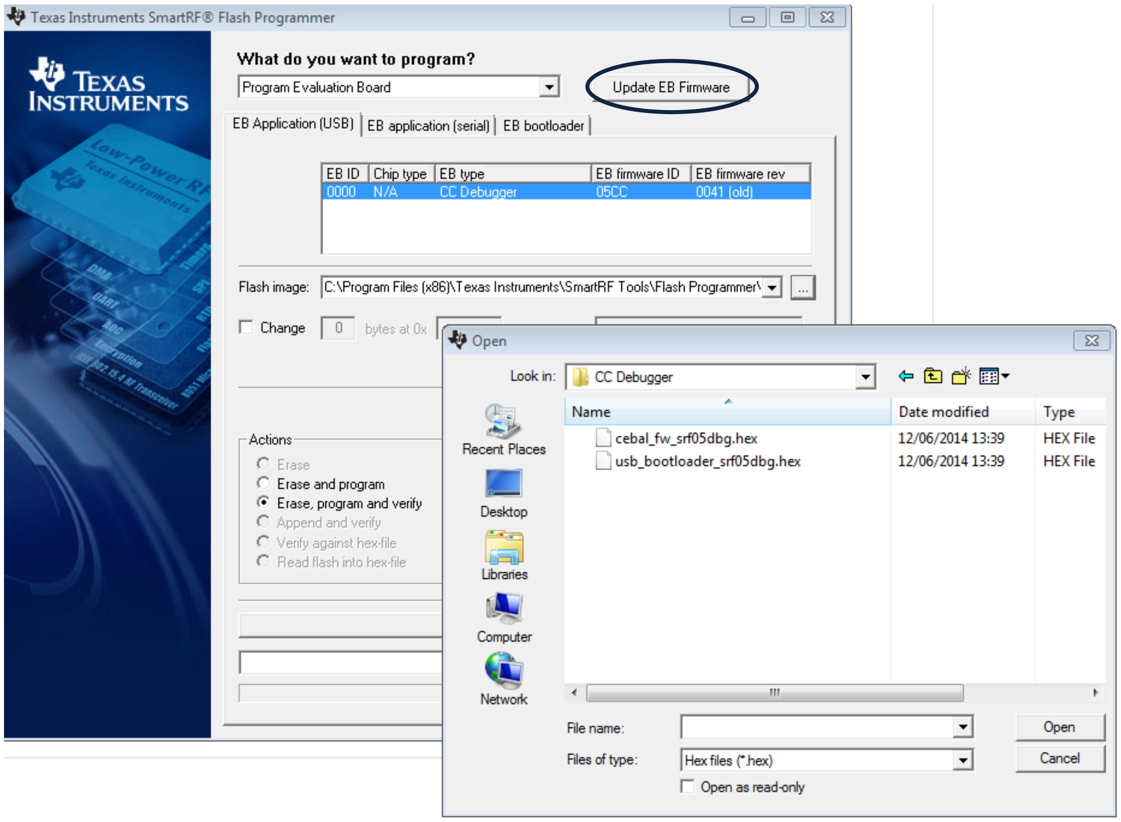
How to update CC Debugger firmware

First, just to be safe, disconnect the CC1111 from the CC Debugger (remove the ribbon cable).

In SmartRF Flash Programmer choose the 'Program Evaluation Board' pull-down, and click on "Update EB Firmware" (which is only enabled if the Debugger has older software), then find the Debugger firmware.

e.g. C:\Program Files (x86)\Texas Instruments\SmartRF Tools\Firmware\CC Debugger\cebal_fw_srf05dbg

(your version may be different)



Program Evaluation Board

EB ID	Chip type	EB type	EB firmware ID	EB firmware rev
0000	N/A	CC Debugger	05CC	0041 (old)

Flash image: C:\Program Files (x86)\Texas Instruments\SmartRF Tools\Flash Programmer\

Change 0 bytes at 0x

- Erase
- Erase and program
- Erase, program and verify [✓]
- Append and verify
- Verify against hex-file
- Read flash into hex-file

Look in: CC Debugger

Name	Date modified	Type
cebal_fw_srf05dbg.hex	12/06/2014 13:39	HEX File
usb_bootloader_srf05dbg.hex	12/06/2014 13:39	HEX File

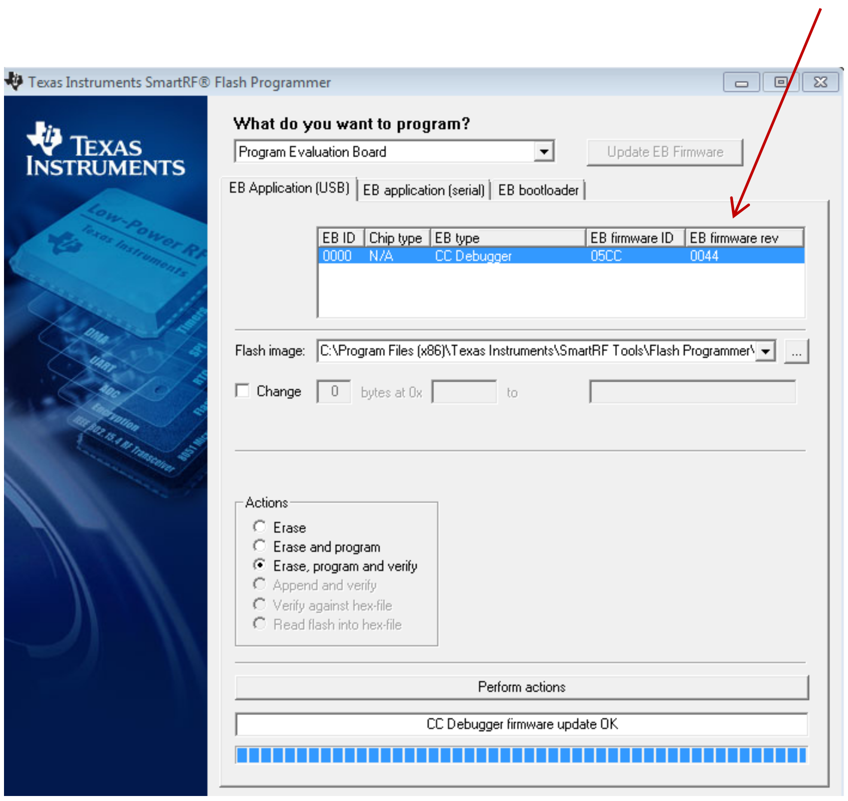
File name:

Files of type: Hex files (*.hex)

Open as read-only

Update the application (not the bootloader).

After a successful update, you should see an updated "EB firmware rev" and "…update OK"



Program Evaluation Board

EB ID	Chip type	EB type	EB firmware ID	EB firmware rev
0000	N/A	CC Debugger	05CC	0044

Flash image: C:\Program Files (x86)\Texas Instruments\SmartRF Tools\Flash Programmer\

Change 0 bytes at 0x to

Actions
- Erase
- Erase and program
- [✓ Erase, program and verify]
- Append and verify
- Verify against hex-file
- Read flash into hex-file

CC Debugger firmware update OK

Troubleshooting the CC Debugger - CC1111 ribbon cable connection

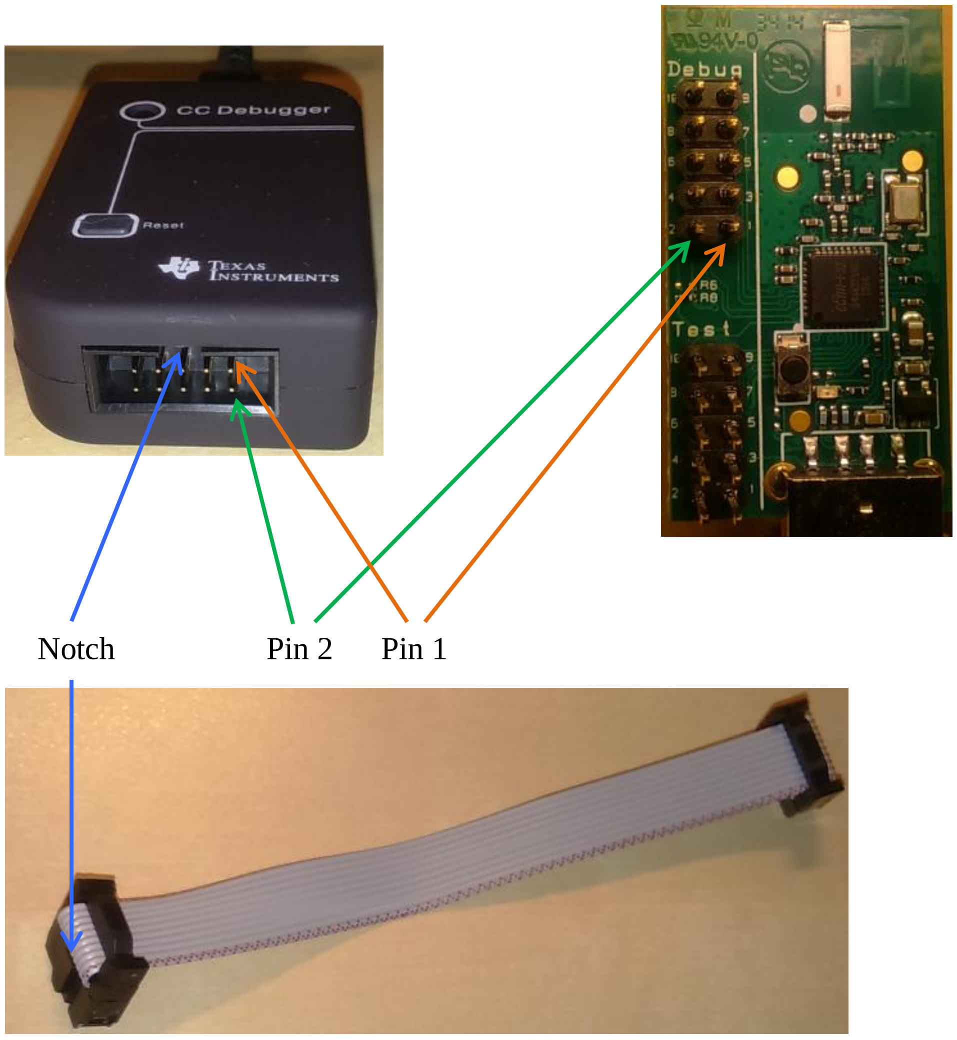
Simply, the first wire of the ribbon cable (often red or patterned to distinguish it from the others) needs to connect from Pin 1 of the CC Debugger to Pin 1 of the CC1111's debug connector.

Even if the cable has been assembled with the red wire going to Pin 10 instead, that doesn't matter provided it connects to Pin 10 at both ends. If that's happening, everything should work.

However, you may find your ribbon cable has been assembled incorrectly and the CC Debugger does not connect correctly to the CC1111 whichever way around you plug it in (the CC Debugger has a polarised notch, which means the cable can only be inserted one way round at the Debugger).

If you're feeling brave, it's possible to remove one of the cable socket connectors and re-fit it the other way round using nothing more than a flat blade screwdriver to push the wires back onto the pins of the connector. Alternatively a small workbench clamp can be used to push the cable back onto the pins.

Another solution is to remove the locating notch from one of the cable's socket connectors, e.g. by carefully using a modelling knife. This would allow insertion into the CC Debugger port in either orientation.



Glossary / Abbreviations

CGM	Continuous Glucose Monitor
Firmware	Software in electronic devices
LED	Light emitting diode
OTG	On-the-go (USB related)
RF	Radio Frequency
SoC	System on Chip
TI	Texas Instruments
Tx	Transmitter / transmission
USB	Universal Serial Bus