INDUSTRIAL SHIELDS

OPEN MOTE B

Open Mote B User Guide:

Open Mote B User Guide

Revision August 2019

Preface

This User Guide is been implemented by Boot & Work, S.L. working under the name Industrial Shields.

Purpose of the manual

The information contained in this manual can be used as a reference to operating, to functions, and to the technical data of the signal modules, power supply modules and interface modules.

Intended Audience

This User Guide is intended for the following audience:

- Persons in charge of introducing automation devices.
- Persons who design automation systems.
- Persons who install or connect automation devices.
- Persons who manage working automation installation.

Warnings:

- Unused pins should not be connected. Ignoring the directive may damage the controller.
- Improper use of this product may severely damage the controller.
- Refer to the controller's User Guide regarding wiring considerations.
- Before using this product, it is the responsibility of the user to read the product's User Guide and all accompanying documentation.

Advertisements:

- Les broches non utilisées ne doivent pas être connectées. Ignorer la directive peut endommager le contrôleur.
- Une utilisation incorrecte de ce produit peut endommager gravement le contrôleur.
- Reportez-vous au Guide de l'utilisateur du contrôleur pour les considérations de câblage.
- Avant d'utiliser ce produit, il incombe à l'utilisateur de lire le Guide de l'utilisateur du produit et la documentation qui l'accompagne.

Application Considerations and Warranty

Read and Understand this Manual

Please read and understand this manual before using the product. Please consult your comments or questions to Industrial Shields before using the product.

Application Consideration

THE PRODUCTS CONTAINED IN THIS DOCUMENT ARE NOT SAFETY RATED. THEY SHOULD NOT BE RELIED UPON AS A SAFETY COMPONENT OR PROTECTIVE DEVICE FOR ENSURING SAFETY OF PERSONS, AS THEY ARE NOT RATED OR DESSIGNED FOR SUCH PURPOSES.

Please know and observe all prohibitions of use applicable to the products.

FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESSIGNED TO ADDRESS THE RISKS, NEVER USE THE INDUSTRIAL SHIELDS PRODUCTS.

NEVER USE THE INDUSTRIAL SHIELDS PRODUCTS BEFORE THEY ARE PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.

Industrial Shields shall not be responsible for conformity with any codes, regulations or standards that apply to the combination of products in the customer's application or use of the product.

The following are some examples of applications for which particular attention must be given. This is not intended to be an exhaustive list of all possible uses of the products, nor is it intended to imply that the uses may be suitable for the products:

- Systems, machines, and equipment that could present a risk to life or property.
- Nuclear energy control systems, combustion systems, railroad systems, aviation systems, medical equipment, amusement machines, vehicles, safety equipment, and installation subject to separate industry or government regulations.
- Outdoor use, uses involving potential chemical contamination or electrical interference, or conditions or uses not described in this document.

At the customer's request, INDUSTRIAL SHIELDS will provide applicable third party certification documents identifying ratings and limitations of use that apply to the products. This information by itself is not sufficient for a complete determination of the suitability of the products in combination with the system, machine, end product, or other application or use.

Disclaimers

Weights and Dimensions

Dimensions and weights are nominal and they are not used for manufacturing purposes, even when tolerances are shown.

Performance Data

The performance data given in this manual is provided as a guide for the user in determining suitability and does not constitute a warranty. It may represent the result of INDUSTRIAL SHIELDS's test conditions, and the users most correlate it to actual application requirements. Actual performance is subject to the INDUSTRIAL SHIELDS Warranty and Limitations of Liability.

Change in Specifications

Product specifications and accessories may be changed at any time based on improvements and other reasons.

It is our practice to change model numbers when features are changed, or published ratings or when significant construction changes are made. However, some specifications of the products may be changed without any notice. When in doubt, special numbers may be assigned to fix or stablish key specifications for your application on your request. Please consult with your INDUSTRIAL SHIELDS representative at any time to confirm actual specifications of purchased products.

Errors and Omissions

The information in this document has been carefully checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical, or proofreading errors, or omissions.

Warranty and Limitations of Liability

Warranty

Industrial Shields's exclusive warranty is that the products are free from defects in materials and workmanship for a period of one year (or other period if specified) from date of sale by Industrial Shields.

INDUSTRIAL SHIELDS MAKES NO REPRESENTATION OR WARRANTY, EXPRESSED OR IMPLIED, REGARDING MERCHANABILITY, NON-INFRINGEMENT, OR FITNESS FOR PARTICULAR PURPOSE OF THE PRODUCTS. ANY BUYER OR USER ACKNOWLEDGES THAT THE BUYER OR USER ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE. INDUSTRIAL SHIELDS DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED

Limitations of Liability

INDUSTRIAL SHIELDS SHALL NOT BE RESPONSIBLE FOR SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR COMERCIAL LOSS IN ANY WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED ON CONTRACT, WARRANTY, NEGLIGENCE, OR STRICT LIABILITY.

IN NO EVENT SHALL INDUSTRIAL SHIELDS BE RESPONISBLE FOR WARRANTY, REPAIR OR OTHER CLAIMS REGARDING THE PRODUCTS UNLESS INDUSTRIAL SHIELDS'S ANALYSIS CONFIRMS THAT THE PRODUCTS WERE PROPERLY HANDLED, STORED, INSTALLED, AND MAINTAINED AND NOT SUBJECT TO CONTAMINATION, ABUSE, MISUSE, OR INAPPROPIATE MODIFICATION OR REPAIR.

Table of Contents

1 General Description Open Mote B product 9
1.1 Industrial Internet of Things development platform 9
2 Technical Specifications 9
2.1 Micro-Controller (Texas Instruments, CC2538) 9
2.2 Transceiver 1 (Texas Instruments, CC2538) 10
2.3 Transceiver 2 (ATMEL, AT86RF215) 10
3 Platform Characteristics 10
4 System overview 11
4.1 Texas Instruments CC2538 11
5 Dimensions 14
6 Software 15
6.1 Preliminaries 15
6.2 Windows 10 15
6.2.1 Python 3.x 15
6.2.2 GCC Compiler for ARM Embedded Processors 19
6.2.3 GCC Compiler for ARM Embedded Processors 23
6.3 GNU/Linux 29
6.4 OpenMote 30
6.4.1 Windows 10 30
6.4.2 GNU/Linux 33
6.4.3 Blinking LED example 34
7 Revision Table 37

INDUSTRIAL SHIELDS

1 General Description Open Mote B product

1.1 Industrial Internet of Things development platform

The OpenMote B is the ultimate hardware development and prototyping platform for the Industrial Internet of Things (IIoT), specifically to researchers and developers working towards next-generation long-range and low-power wireless field area networks based on the IPv6 stack. It is built around the well-supported Texas Instruments CC2538 ARM-Cortex-M3 micro-controller, and it features simultaneous multi-band operation in the 2.4 GHz and 868/915MHz ISM bands with complete support for the latest IEEE 802.15.4 standards, including the MR-OFDM modulations of IEEE 802.15.4g-2012.

2 Technical Specifications

2.1 Micro-Controller (Texas Instruments, CC2538)

- **ARM Cortex-M3 with code pre-fetch**
  - Running at 16 MHz or 32 MHz
  - 32 Kbytes RAM
  - 512 Kbytes FLASH

- **On-chip peripherals:**
  - 4x general purpose, 1x sleep timer
  - 1x 12 bit ADC with 8 channels
  - 2x SPI, 2x UART, 1x I2C

- **Security hardware acceleration:**
  - AES-128/256/SHA2 encryption
  - ECC-128/256 secure key exchange

- **Low-power operation:**
  - Active mode: 7/13mA (16/32 MHz)
  - LPM1: 600uA (full retention, 4us wake-up)
  - LPM2: 1.3 uA (16 Kbyte RAM retention, 128us wake-up, wake-up from RTC)
  - LPM3: 0.4 uA (16 Kbyte RAM retention, 128us wake-up, wake-up from GPIO)

2.2 Transceiver 1 (Texas Instruments, CC2538)

- **Operates in the 2.4 GHz ISM band with support for IEEE 802.15.4-2006**
  - Modulation: OQPSK with DSSS
  - Data rate: 250 kbps
  - Receiver sensitivity: -97 dBm
  - Transmit power: 7 dBm
  - Transmit current: 24 mA at 0 dBm
  - Receive current: 20 mA

2.3 Transceiver 2 (ATMEL, AT86RF215)

- **Operates in the 868/915MHz and 2.4 GHz ISM bands with support for IEEE 802.15.4g-2012**
  - Modulation: MR-FSK/OFDM/O-QPSK
  - Data rate: 6.25 kbps to 2400 kbps
  - Receiver sensitivity: -123 dBm
  - Transmit power: 14.5dBm
  - Transmit current: 62 mA at 14 dBm
  - Receive current: 28 mA

3 Platform Characteristics

- **Human interfacing**: The main board includes 4 LEDs (Green, Yellow, Orange and Red) and a user button which are intended for debugging purposes. In addition, the main board also includes a reset button that allows to perform a hardware reset.

- **Serial communications:** The main board can communicate with a computer using a UART port on the CC2538. The solution is based on the FTDI FT2232H chip, a Serial-to-USB converter that allows to communicate with a computer using a standard UART port. In addition, the FTDI chip allows to program the CC2538 directly using the internal bootloader and the cc2538-bsl Python script.

- **Board expansion**: The main OpenMote-B board includes an expansion port (8 pins with 2.54 mm spacing) that can be used for debugging or to connect daughter boards, i.e., the OpenMote-B sensors board. The expansion board includes a VCC (2.5V) and a GND pin, as well as six configurable pins.

- **Extended security**: The main OpenMote-B board includes hardware-accelerated support for cryptographic functions using SHA2, AES-128/256, ECC-128/256 and RSA algorithms.

- **Antenna connectors:** The main board includes two SMA antenna connectors for the Sub-GHz and 2.4 GHz antennas. The Sub-GHz antenna connector is directly connected to the Sub-GHz radio on the AT86RF215. The 2.4 GHz antenna is multiplexed using an RF switch to the CC2558 and the AT86RF215 radio transceivers. The direction of the RF switch can be controlled using two CC2538 pins.

- **Power**: The main board can be powered from a USB port (5V) through an A male plug connector, or from two AA batteries (3V) located on the back of the board. The board features an auto-switching mechanism that selects the USB source whenever it is available and seamlessly transitions to the AA batteries when disconnected. Finally, the board includes an on/off button to disconnect the two AA batteries when not used.

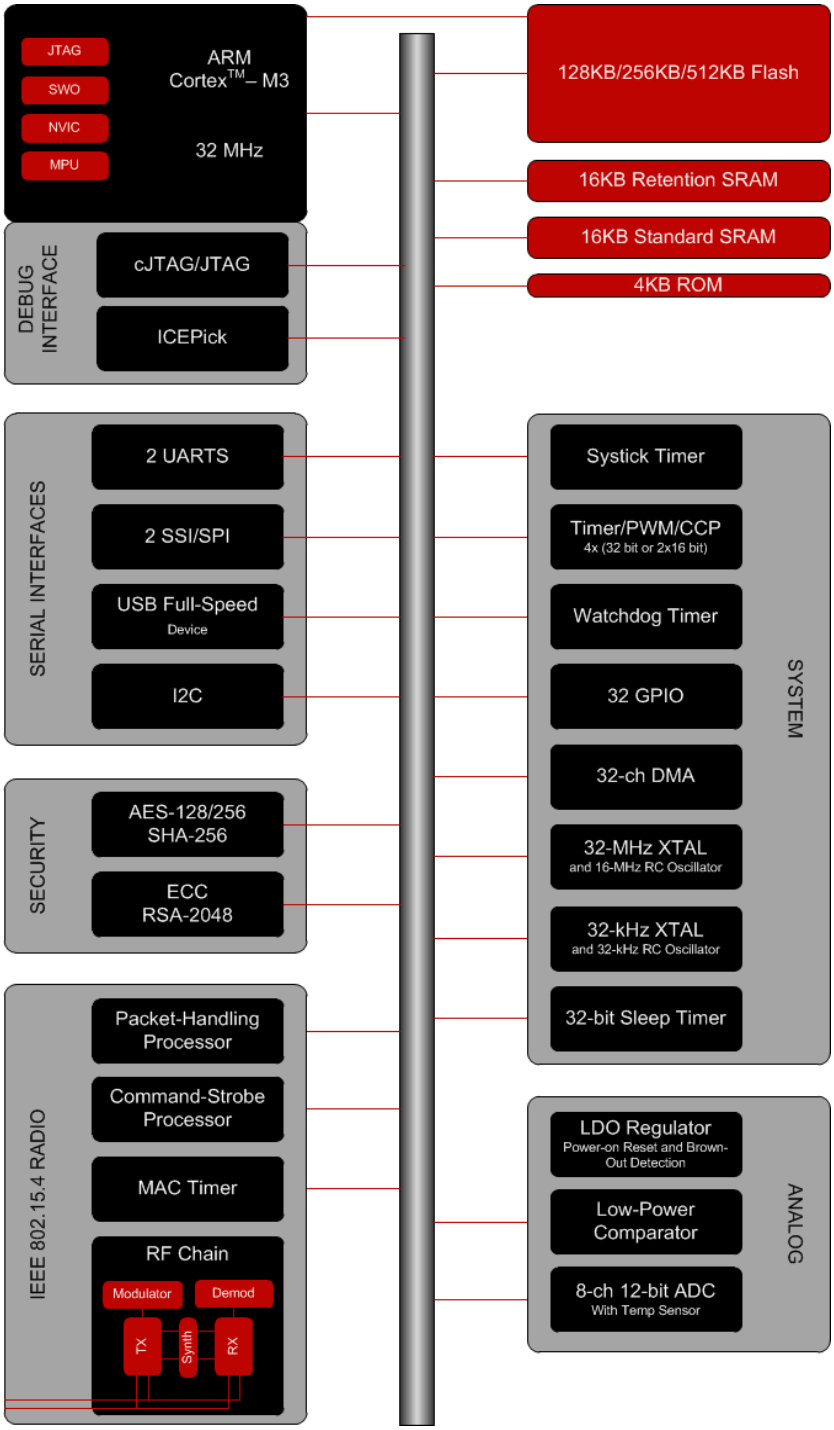
- **Current sensing:** The main board includes two ports (3 pins with 2.54 mm spacing) to measure current consumption of the whole system during operation. The first port measures the current consumption of the CC2538 chip, whereas the second port measures the current consumption of the AT86RF215 chip.

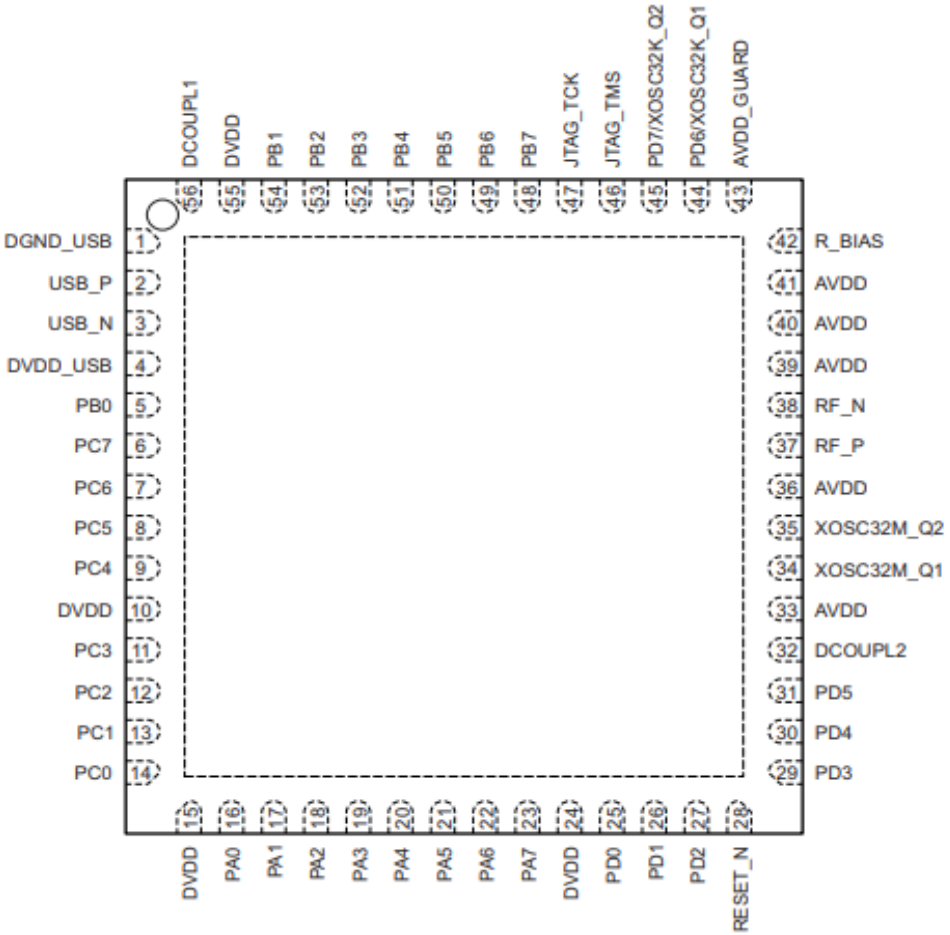
- **JTAG port:** The main board includes a 10-pin ARM connector that allows to load and debug code using an external JTAG probe. The interface is compatible with the main toolchains: Code Composer Studio, IAR Embedded Workbench and ARM.

4 System overview

4.1 Texas Instruments CC2538

The Texas Instruments CC2538 is a wireless micro-controller SoC (System on Chip) targeted at high-performance applications. It combines a powerful ARM Cortex-M3 running at 32 MHz, with 32 Kbytes of RAM and 512 Kbytes of Flash, with a robust 2.4 GHz radio transceiver compatible with the IEEE 802.15.4 standard. In addition, the CC2538 includes various peripherals to interface the processor with other systems: 2x SPI, 2x UART, 1x I2C.


JTAG
SWO
NVIC
MPU
ARM Cortex™ – M3
32 MHz
128KB/256KB/512KB Flash
16KB Retention SRAM
16KB Standard SRAM
4KB ROM
DEBUG INTERFACE
cJTAG/JTAG
ICEPick
SERIAL INTERFACES
2 UARTS
2 SSI/SPI
USB Full-Speed Device
I2C
SECURITY
AES-128/256 SHA-256
ECC RSA-2048
IEEE 802.15.4 RADIO
Packet-Handling Processor
Command-Strobe Processor
MAC Timer
RF Chain
Modulator
Demod
TX
Synth
RX
SYSTEM
Systick Timer
Timer/PWM/CCP 4x (32 bit or 2x16 bit)
Watchdog Timer
32 GPIO
32-ch DMA
32-MHz XTAL and 16-MHz RC Oscillator
32-kHz XTAL and 32-kHz RC Oscillator
32-bit Sleep Timer
ANALOG
LDO Regulator Power-on Reset and Brown-Out Detection
Low-Power Comparator
8-ch 12-bit ADC With Temp Sensor



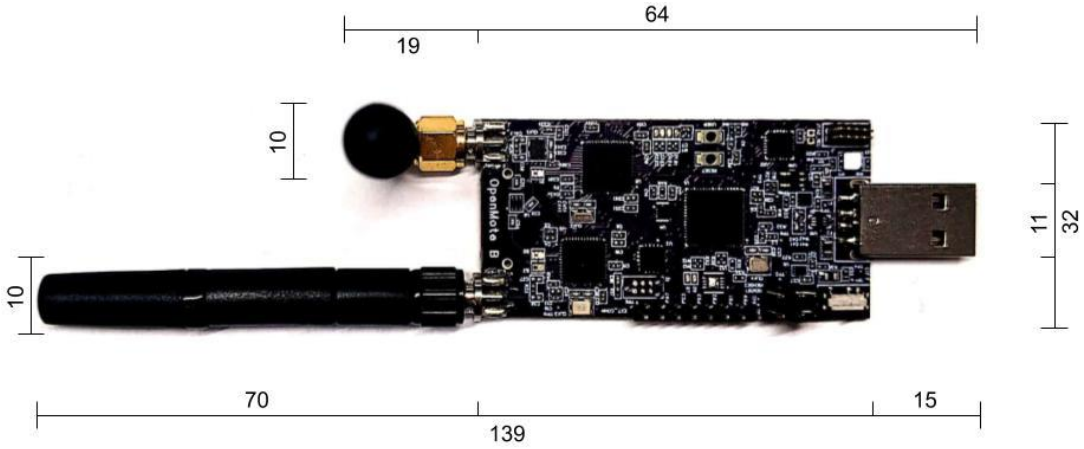
CC2538 Datasheet (http://www.ti.com/lit/gpn/cc2538)

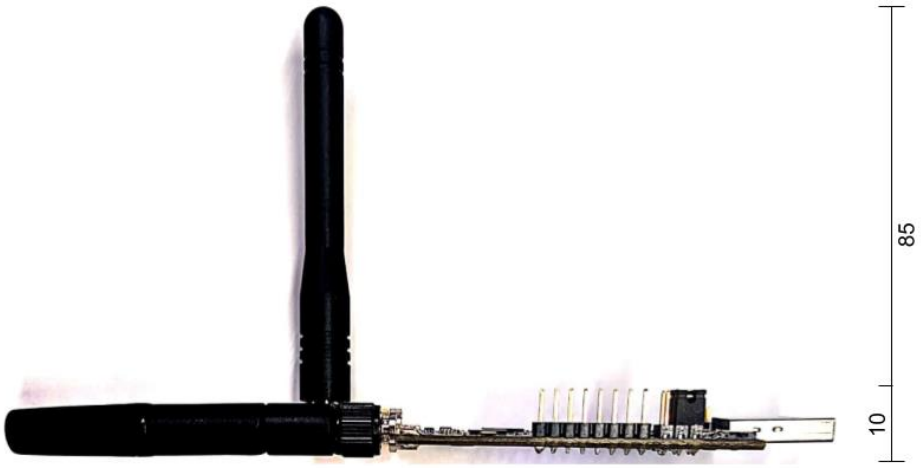
CC2538 User Manual (http://www.ti.com/lit/pdf/swru319)

CC2538 Errata (http://www.ti.com/lit/pdf/swrz045)

5 Dimensions

In mm





6 Software

This section is aimed at providing a starting point to run the main open-source software projects on the OpenMote-B hardware.

6.1 Preliminaries

To program the OpenMote-B boards with the various open source software projects it is necessary to have a working environment to build, upload, and debug the code for the ARM Cortex-M3 processor. It is possible to create a working environment with either Windows or Linux, as described next. Macintosh is known to work, but the former operating systems are preferred as they have been thoroughly tested.

Hence, the following steps show how to install the tools required to compile, run and debug embedded software on the OpenMote-B boards using either Windows 10 or Ubuntu 18.04 LTS. The process assumes that the user has a working installation of either operating system, either natively on their computer or in a virtual environment (i.e. VMWare Workstation or VirtualBox).

The tools that are required are:

- Python 3 with the SCons, PySerial and IntelHex packages
- GCC compiler for ARM Embedded Processors

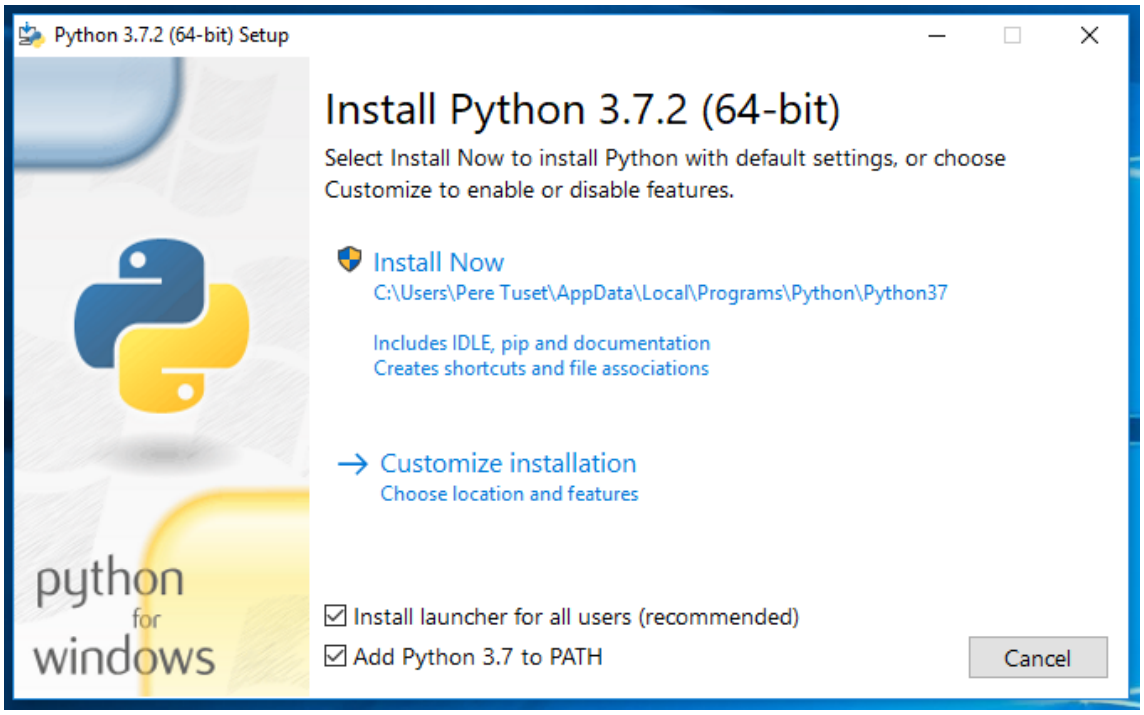
6.2 Windows 10

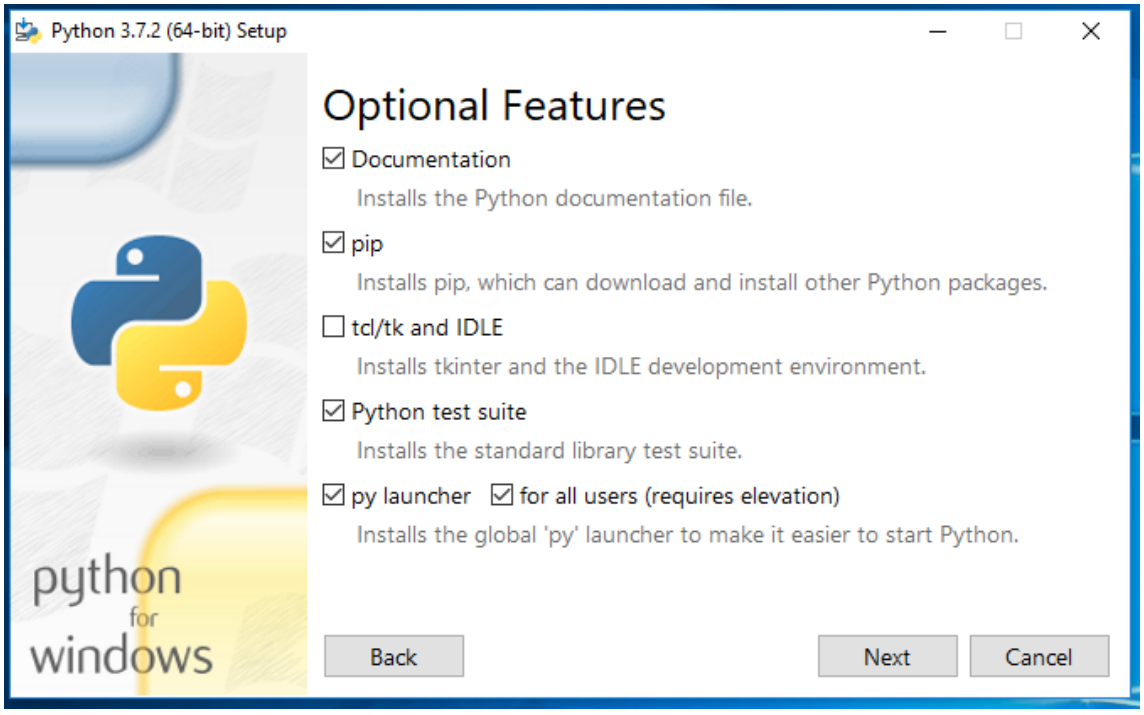
6.2.1 Python 3.x

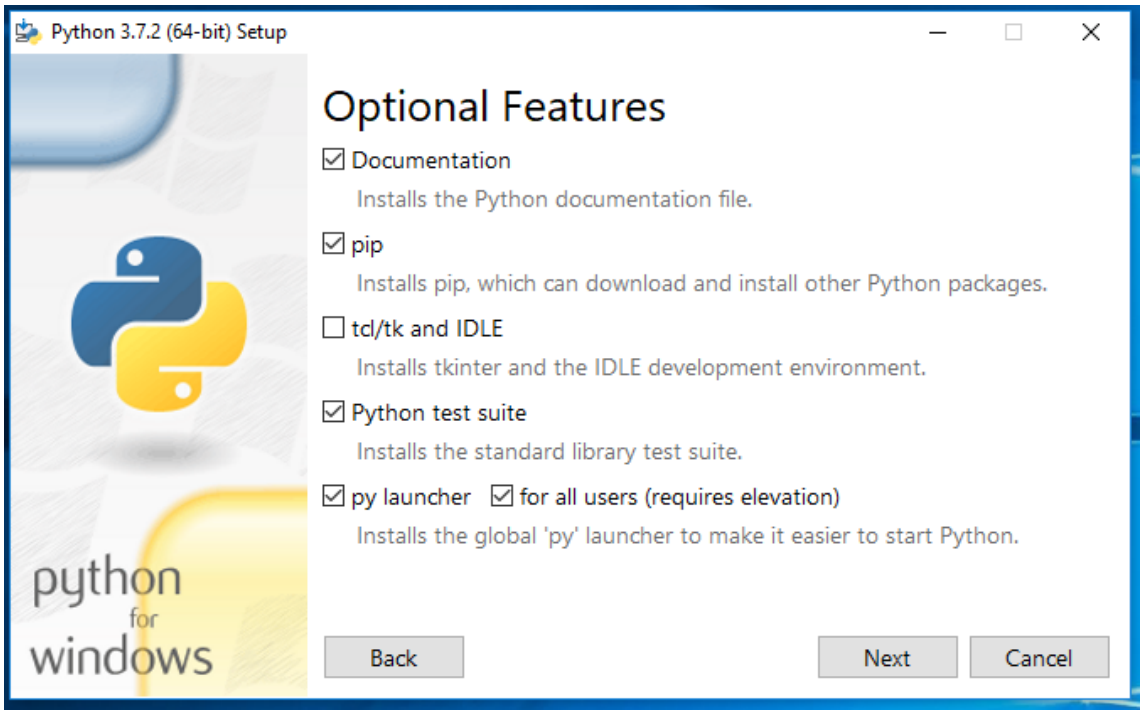
To install Python 3.x go to the official website and download the latest version[1].

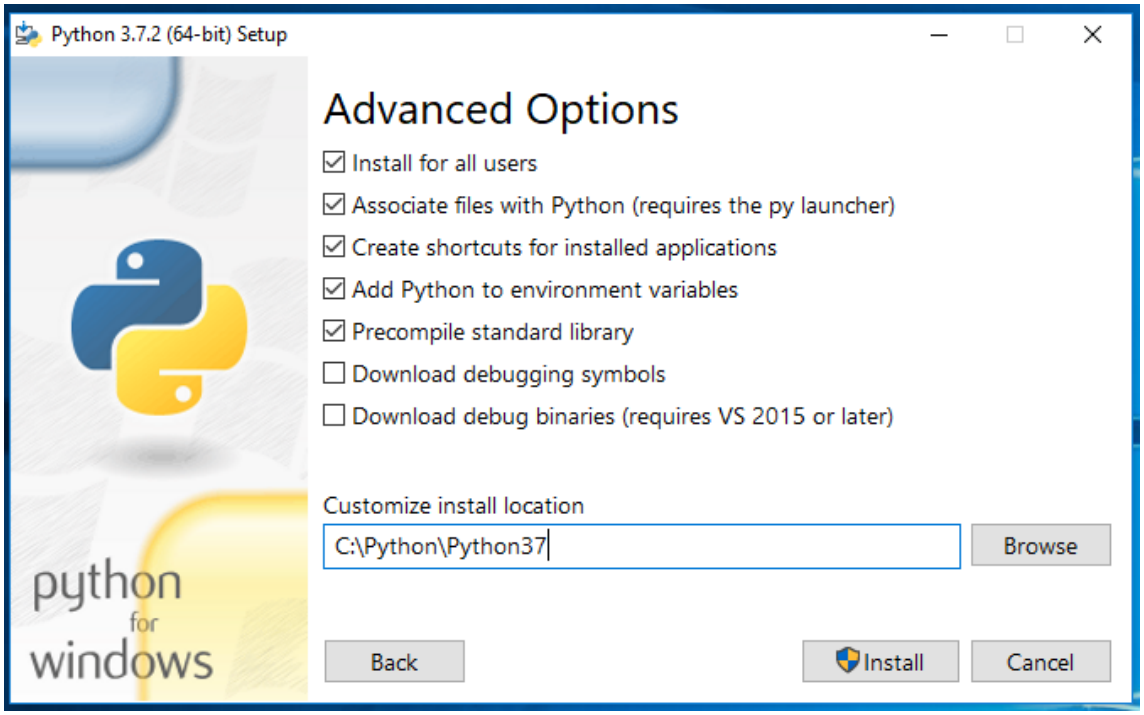
[1] https://www.python.org/downloads/windows/

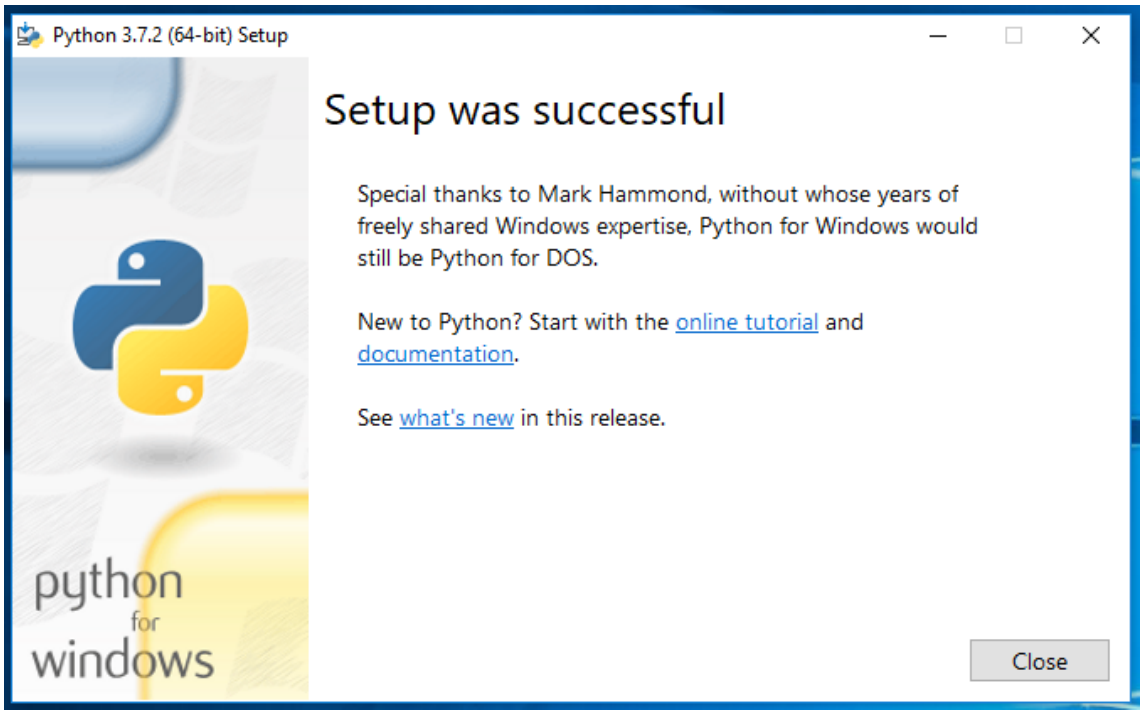
Once downloaded execute the installer and follow the steps depicted in the next Figures. It is important that you install Python for all users and also that you add the directory to the system PATH.



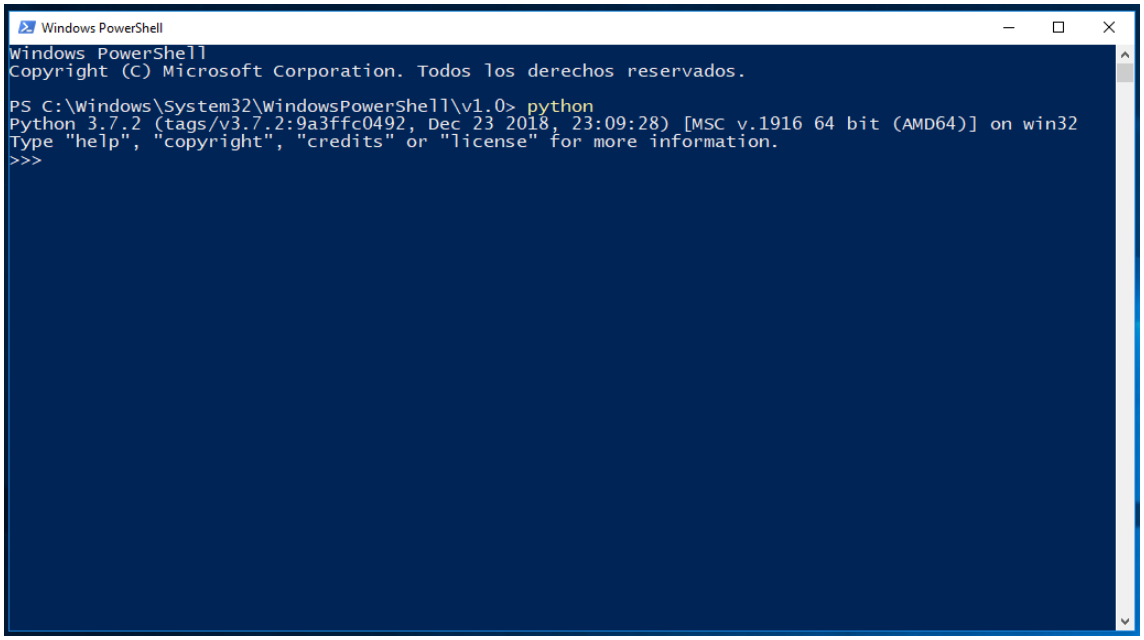



Python 3.7.2 (64-bit) Setup
Optional Features
Documentation
Installs the Python documentation file.
pip
Installs pip, which can download and install other Python packages.
tcl/tk and IDLE
Installs tkinter and the IDLE development environment.
Python test suite
Installs the standard library test suite.
py launcher
for all users (requires elevation)
Installs the global 'py' launcher to make it easier to start Python.
python
for
windows
Back
Next
Cancel


Python 3.7.2 (64-bit) Setup
Advanced Options
Install for all users
Associate files with Python (requires the py launcher)
Create shortcuts for installed applications
Add Python to environment variables
Precompile standard library
Download debugging symbols
Download debug binaries (requires VS 2015 or later)
Customize install location
C:\Python\Python37
Browse
python
for
windows
Back
Install
Cancel

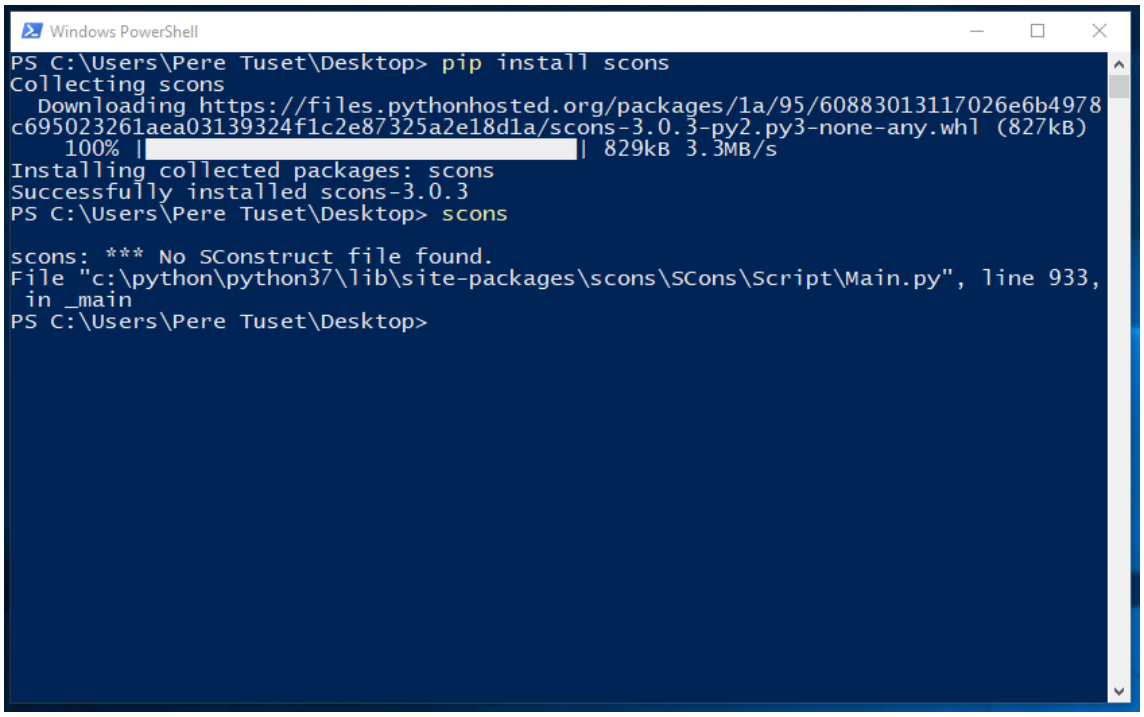


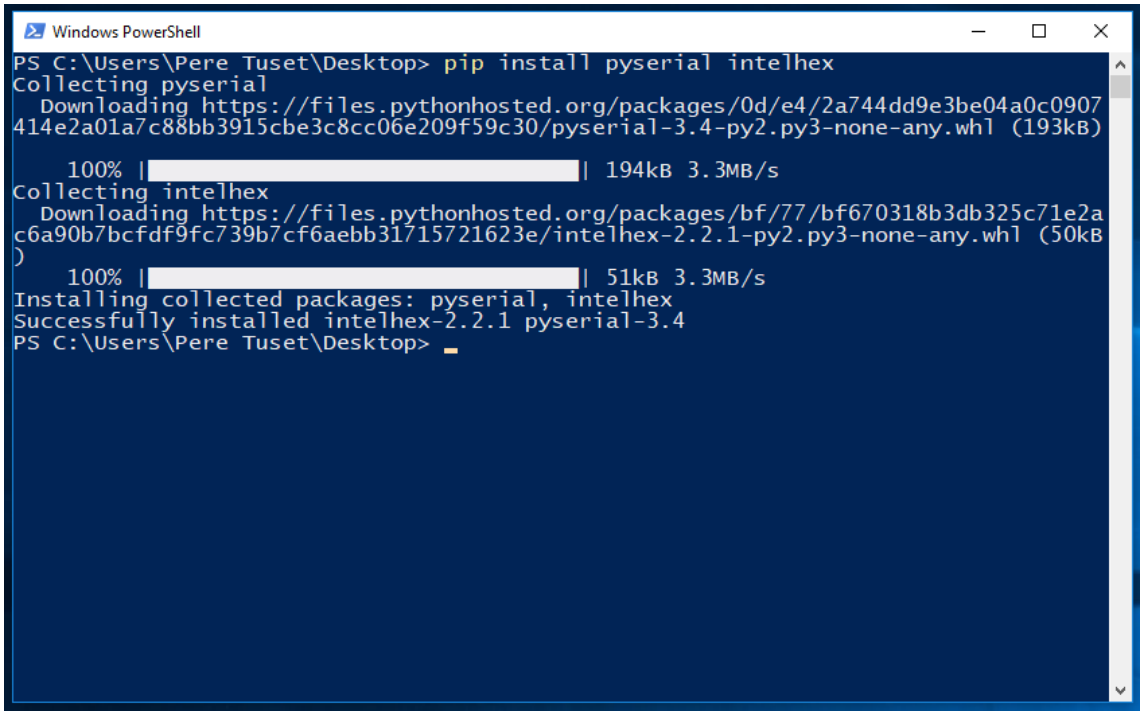
Once Python 3.x is installed in your system you should be able to execute it by running the python command from a command line, i.e. Windows PowerShell. If the installation was successful you should see a terminal like the next image.



Once Python 3.x is installed you need to install the additional packages that are required to compile, load and debug code. As mentioned earlier, these packages are SCons, PySerial and IntelHex and they can be installed using pip, as shown in the next images.

- Pip install scons
- Pip install pyserial
- Pip install intelhex

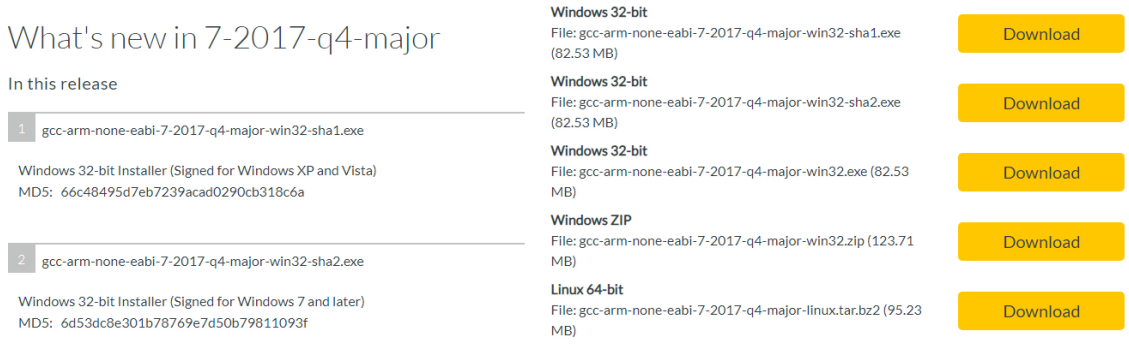




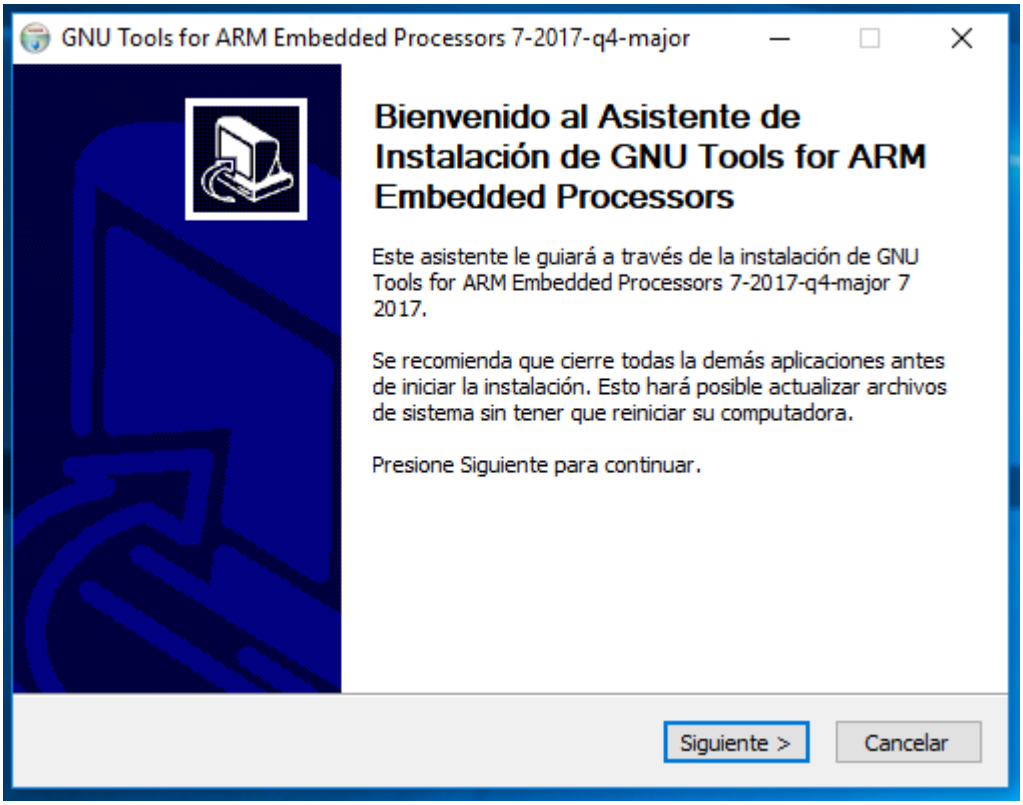
6.2.2 GCC Compiler for ARM Embedded Processors

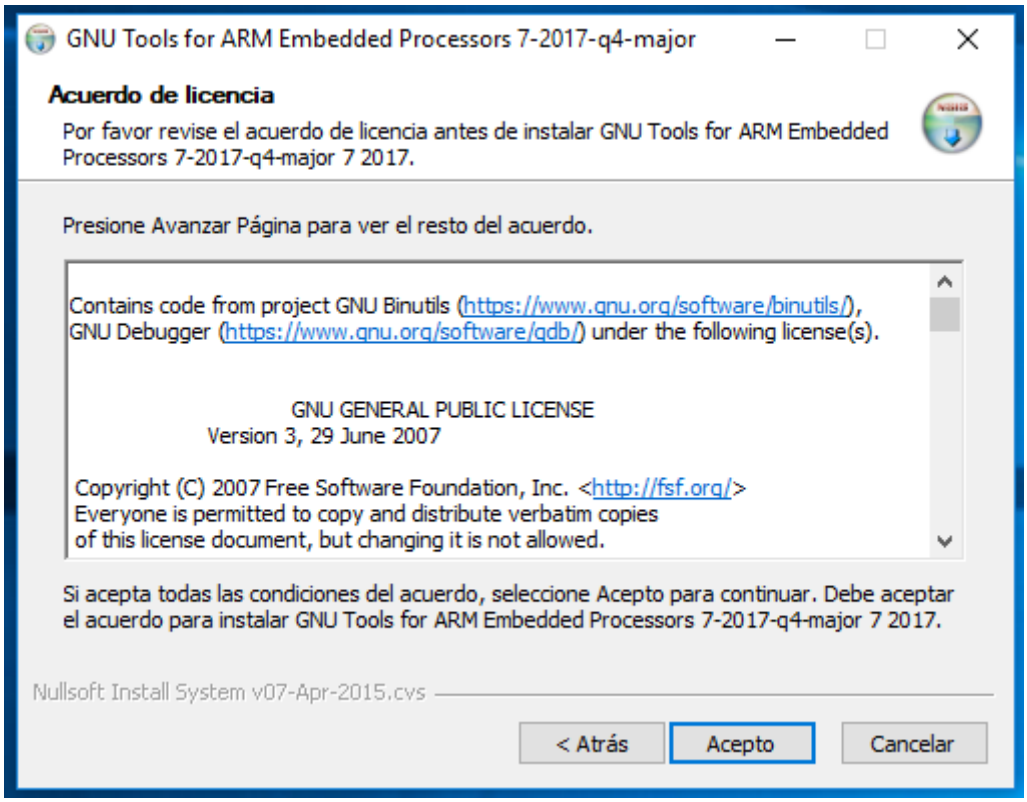
To install the GCC Compiler for ARM Embedded Processors go to the official website and download the latest version: 7-2017-q4-major[2].

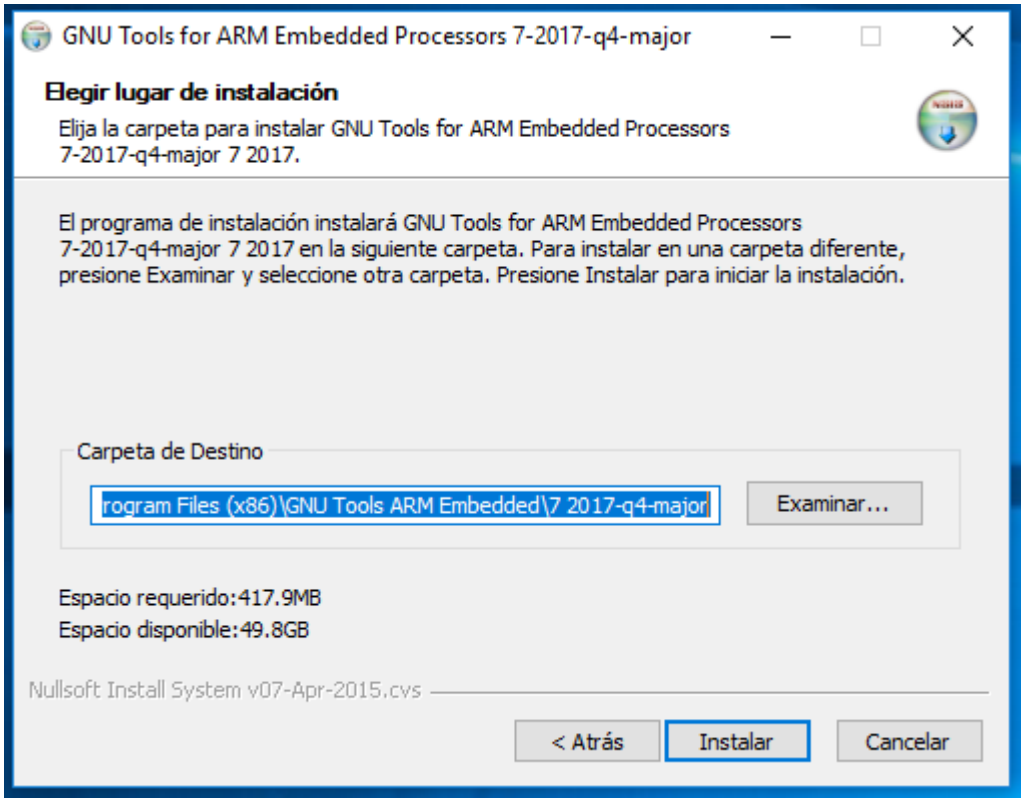
[2] https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads/7-2017-q4-major-1-1

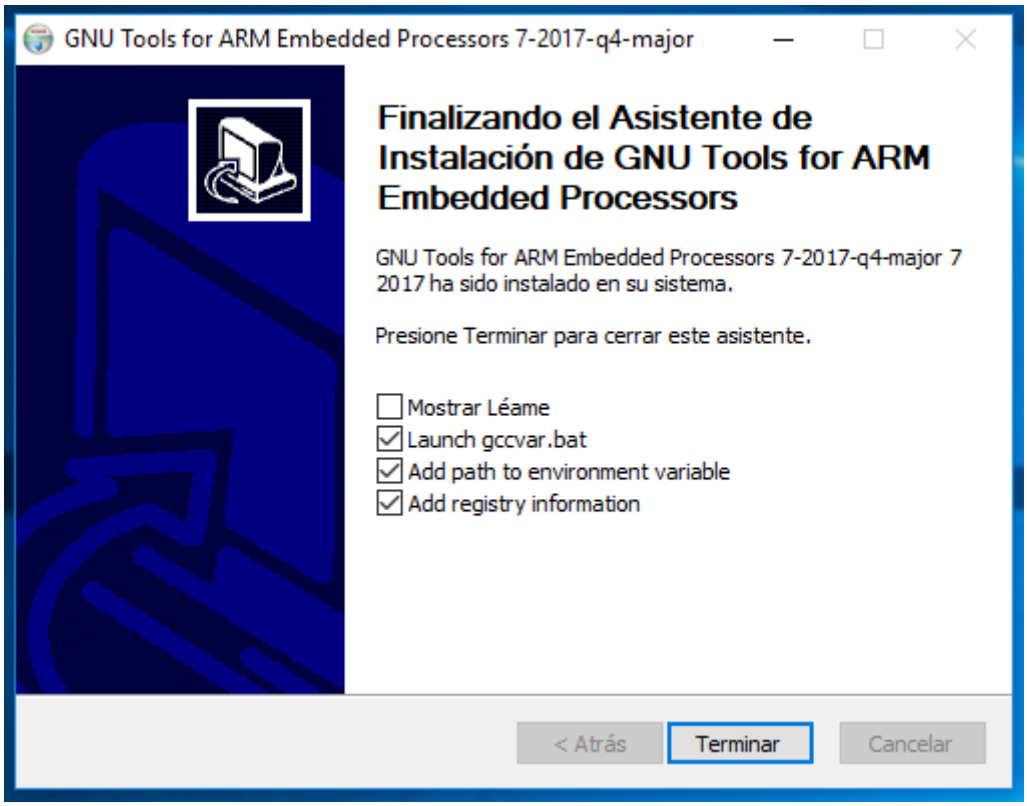


Once downloaded execute the installer and follow the steps depicted in the next Figures. It is important that you add the install directory in the path environment.

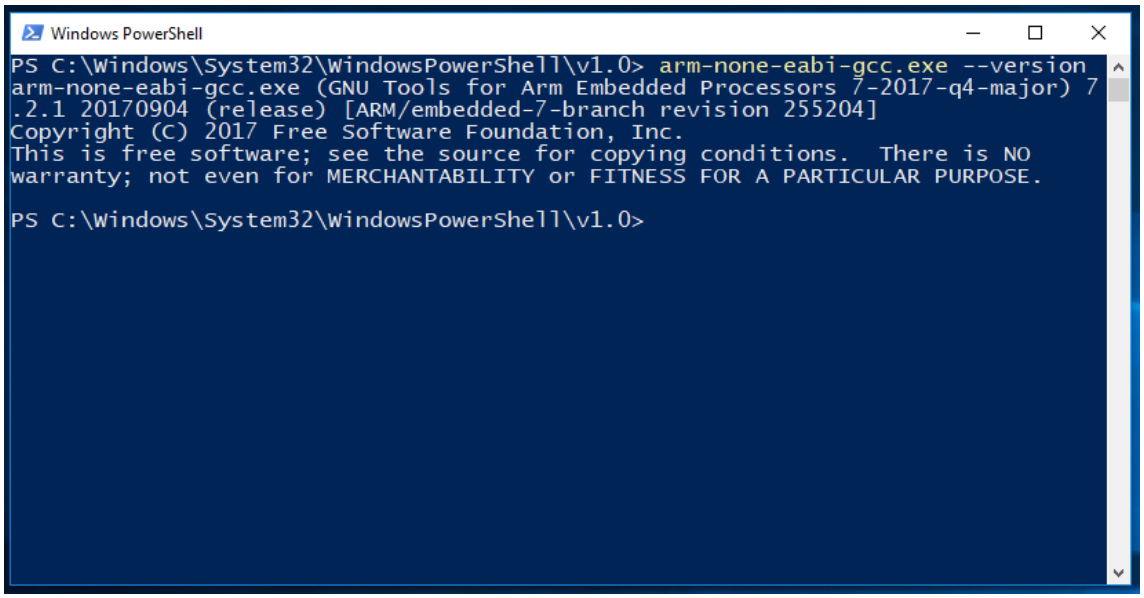



GNU Tools for ARM Embedded Processors 7-2017-q4-major
Acuerdo de licencia
Por favor revise el acuerdo de licencia antes de instalar GNU Tools for ARM Embedded Processors 7-2017-q4-major 7 2017.
Presione Avanzar Página para ver el resto del acuerdo.
Contains code from project GNU Binutils (https://www.gnu.org/software/binutils/), GNU Debugger (https://www.gnu.org/software/gdb/) under the following license(s).
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007
Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Si acepta todas las condiciones del acuerdo, seleccione Acepto para continuar. Debe aceptar el acuerdo para instalar GNU Tools for ARM Embedded Processors 7-2017-q4-major 7 2017.
Nullsoft Install System v07-Apr-2015.cvs
< Atrás
Acepto
Cancelar


GNU Tools for ARM Embedded Processors 7-2017-q4-major
Elegir lugar de instalación
Elija la carpeta para instalar GNU Tools for ARM Embedded Processors 7-2017-q4-major 7 2017.
El programa de instalación instalará GNU Tools for ARM Embedded Processors 7-2017-q4-major 7 2017 en la siguiente carpeta. Para instalar en una carpeta diferente, presione Examinar y seleccione otra carpeta. Presione Instalar para iniciar la instalación.
Carpeta de Destino
rogram Files (x86)\GNU Tools ARM Embedded\7 2017-q4-major
Examinar...
Espacio requerido:417.9MB
Espacio disponible:49.8GB
Nullsoft Install System v07-Apr-2015.cvs
< Atrás
Instalar
Cancelar

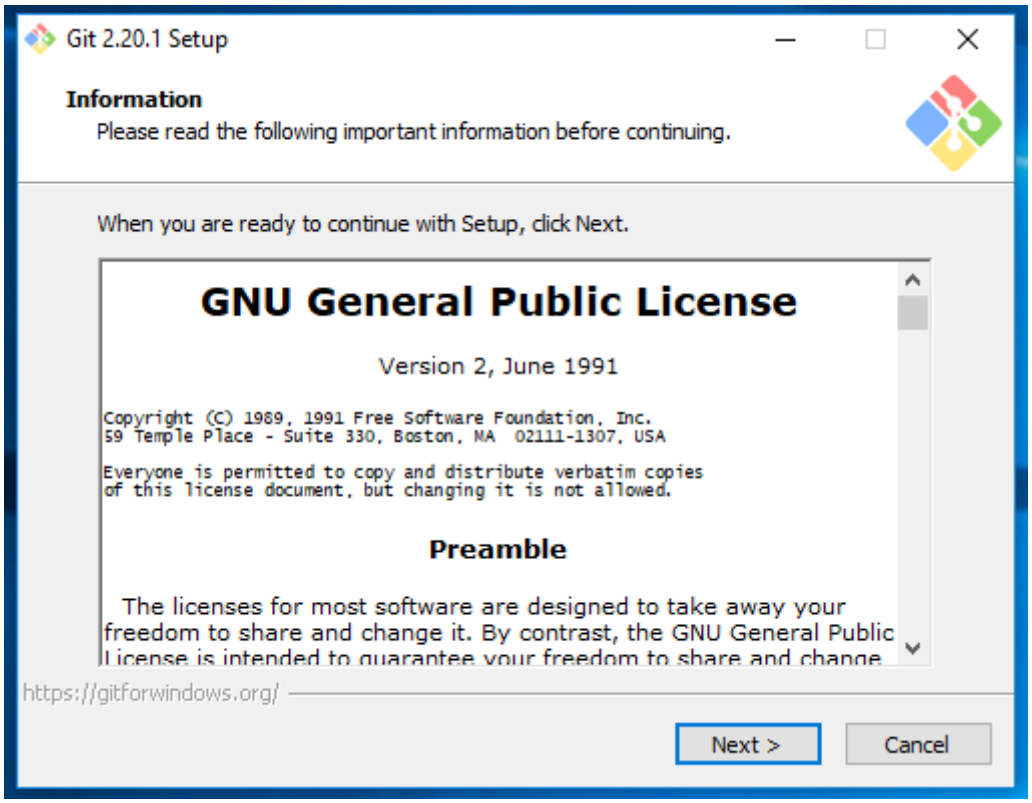


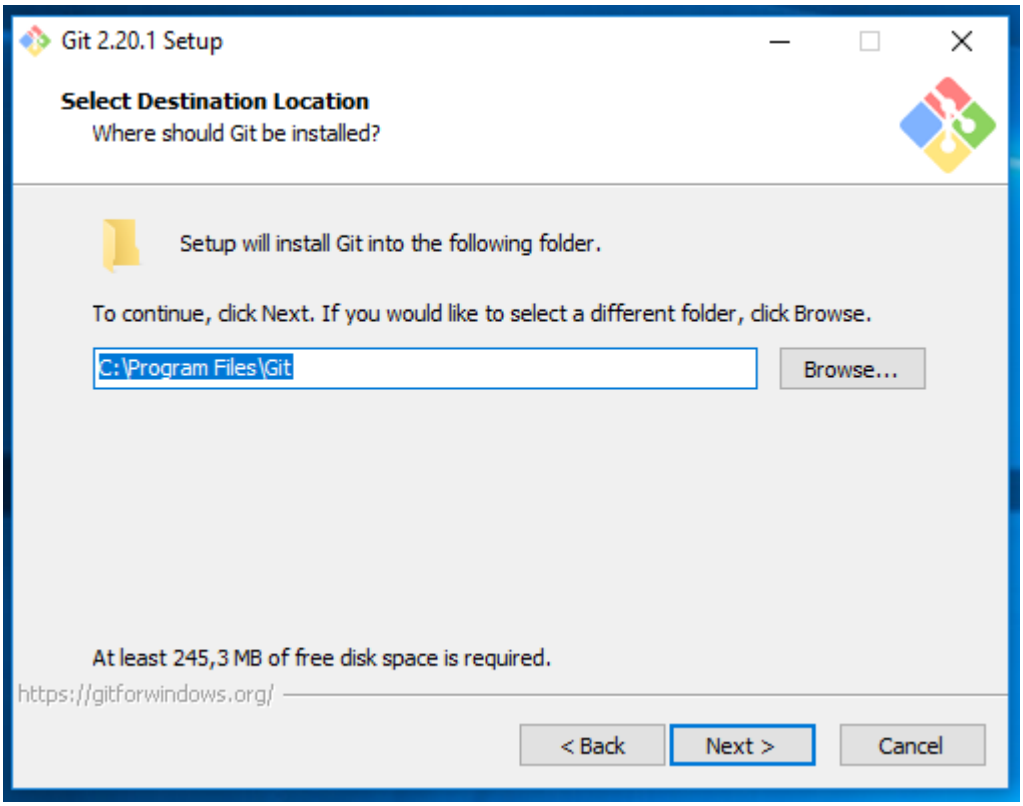
Once it is installed in your system you should be able to execute it by running the arm-none-eabi-gcc.exe --version command from a command line, i.e. Windows PowerShell. If the installation was successful you should see a terminal like the next image.

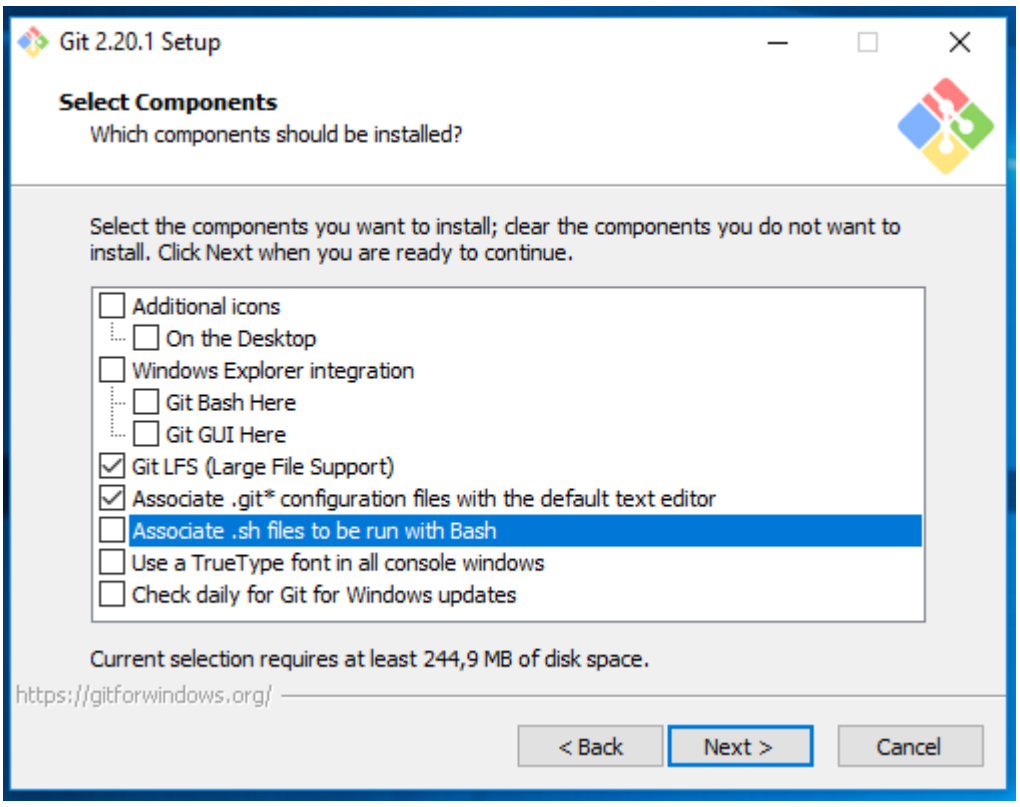


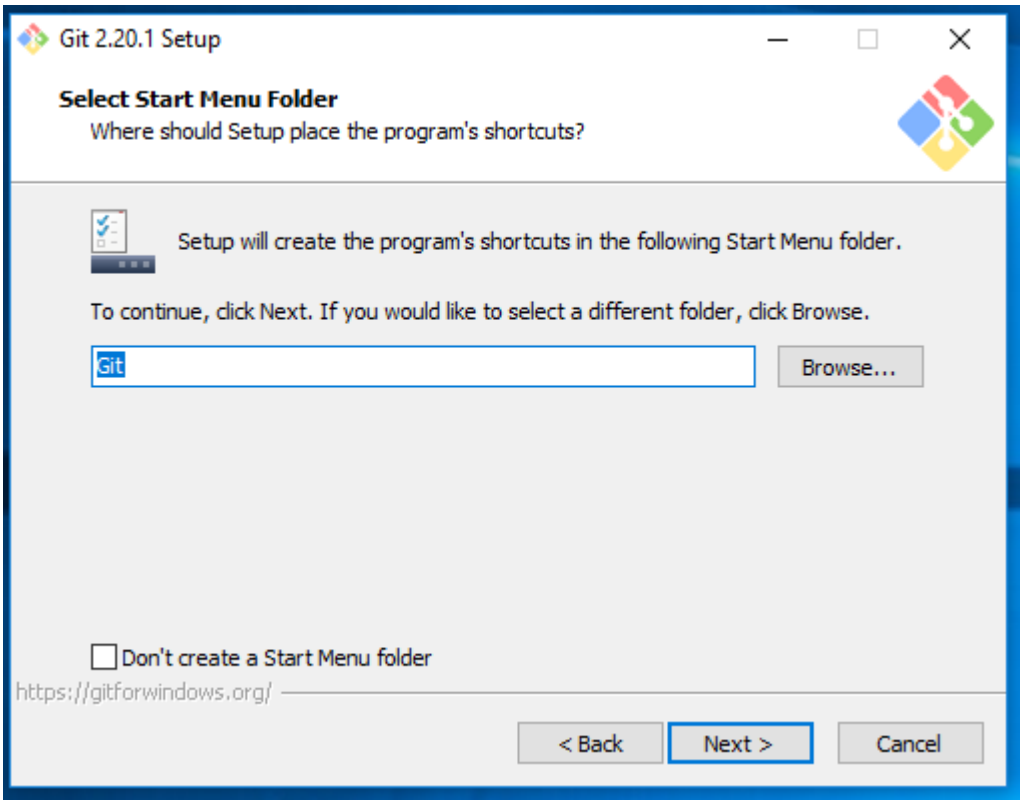
6.2.3 GCC Compiler for ARM Embedded Processors

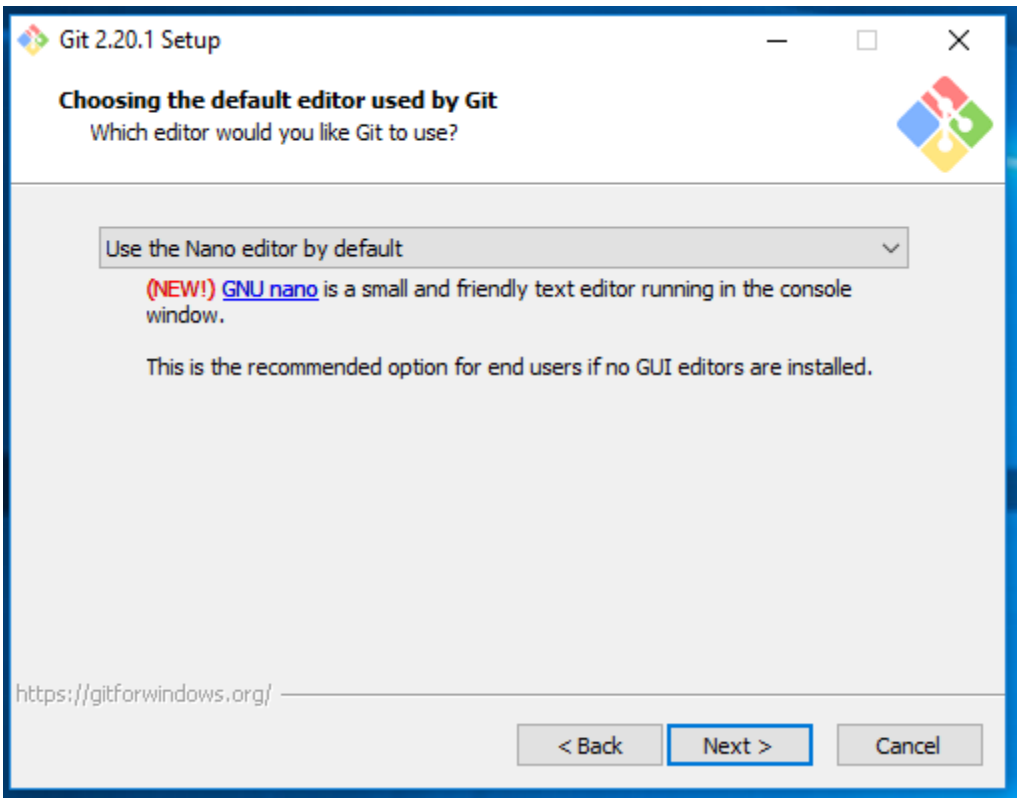
https://git-scm.com/download/win

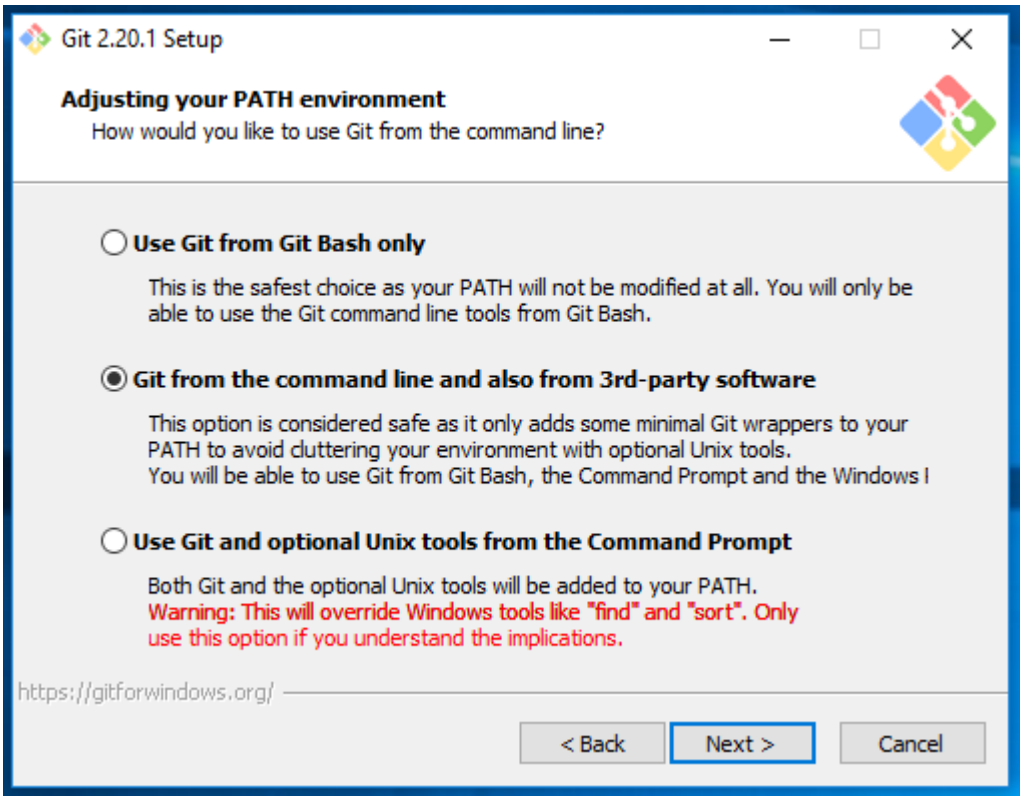


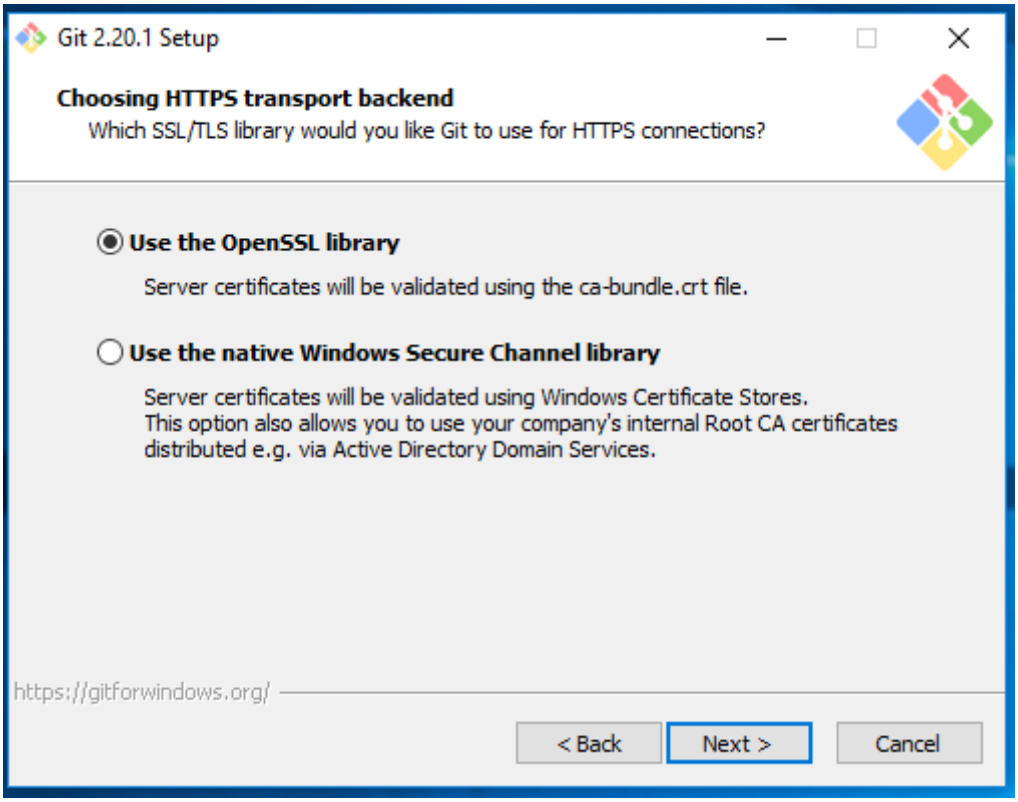

Git 2.20.1 Setup
Select Destination Location
Where should Git be installed?
Setup will install Git into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Program Files\Git
Browse...
At least 245,3 MB of free disk space is required.
https://gitforwindows.org/
< Back
Next >
Cancel

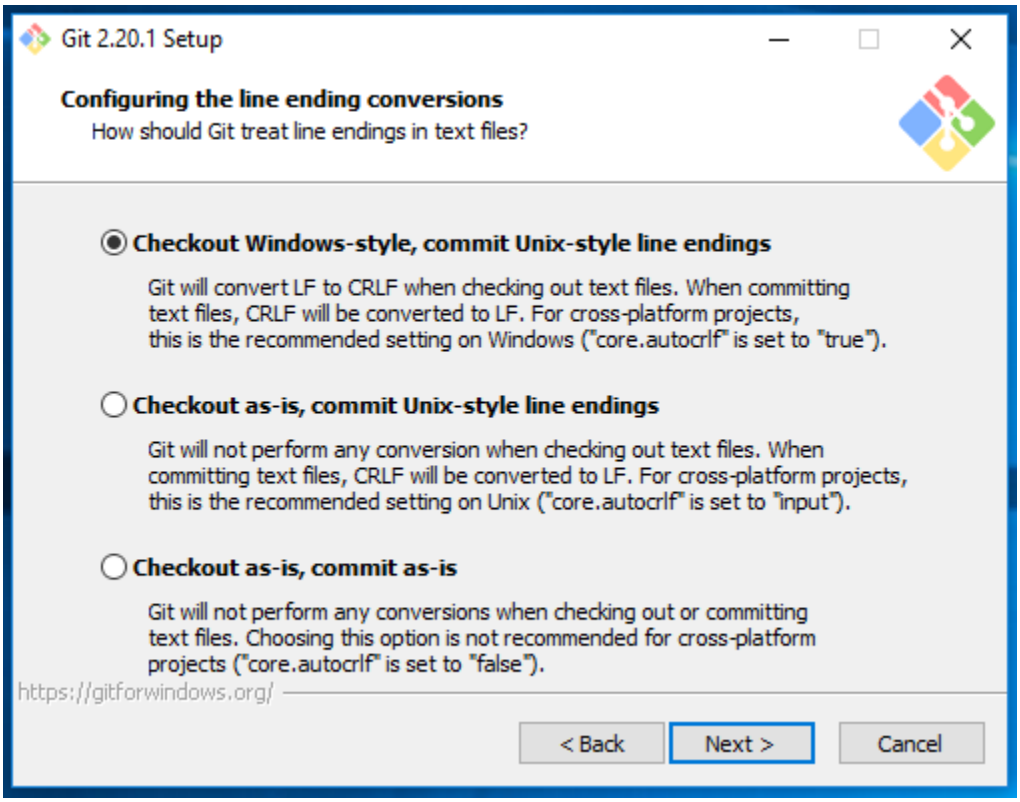

Git 2.20.1 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Git Bash Here
Git GUI Here
Git LFS (Large File Support)
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
Use a TrueType font in all console windows
Check daily for Git for Windows updates
Current selection requires at least 244,9 MB of disk space.
https://gitforwindows.org/
< Back
Next >
Cancel

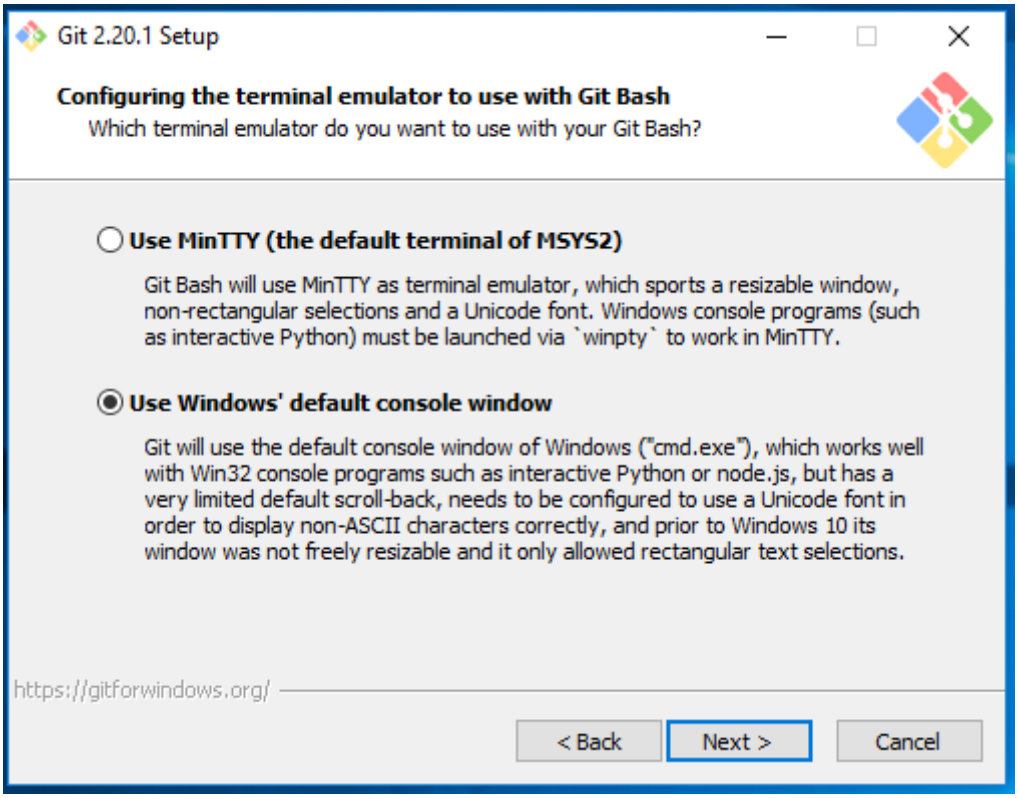

Git 2.20.1 Setup
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
Git
Browse...
Don't create a Start Menu folder
https://gitforwindows.org/
< Back
Next >
Cancel

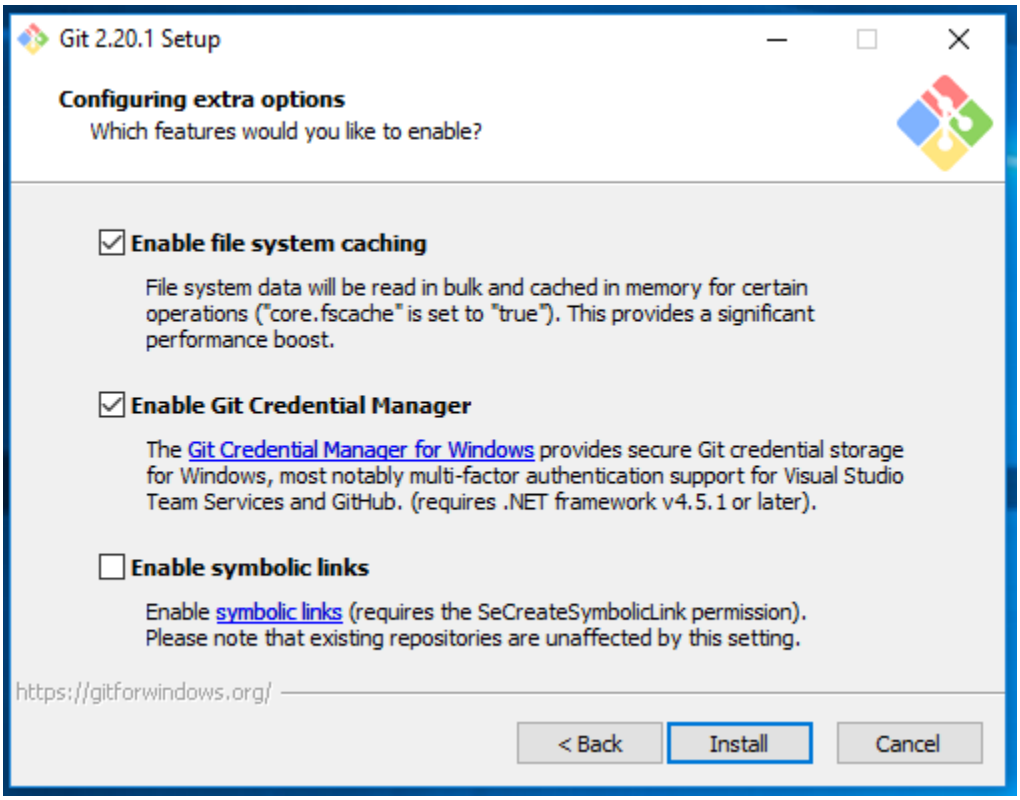

Git 2.20.1 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use the Nano editor by default
(NEW!) GNU nano is a small and friendly text editor running in the console window.
This is the recommended option for end users if no GUI editors are installed.
https://gitforwindows.org/
< Back
Next >
Cancel

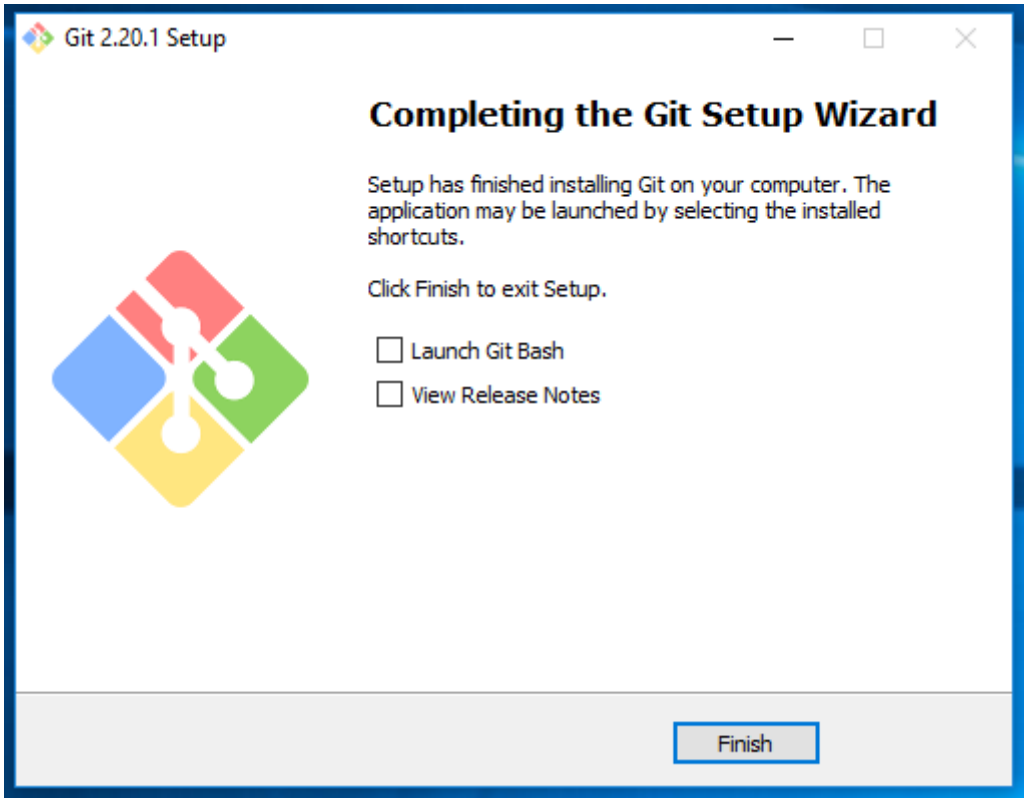

Git 2.20.1 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows I
Use Git and optional Unix tools from the Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
< Back
Next >
Cancel

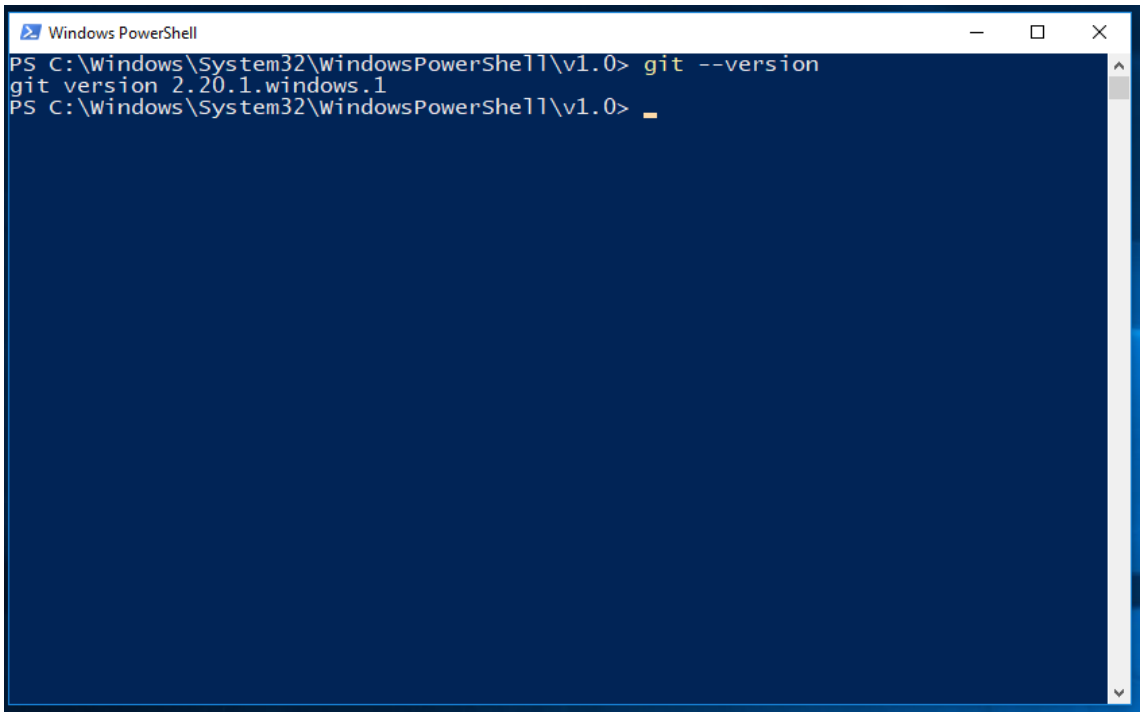

Git 2.20.1 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
< Back
Next >
Cancel


Git 2.20.1 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
< Back
Next >
Cancel


Git 2.20.1 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
< Back
Next >
Cancel


Git 2.20.1 Setup
Configuring extra options
Which features would you like to enable?
Enable file system caching
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.
Enable Git Credential Manager
The Git Credential Manager for Windows provides secure Git credential storage for Windows, most notably multi-factor authentication support for Visual Studio Team Services and GitHub. (requires .NET framework v4.5.1 or later).
Enable symbolic links
Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.
https://gitforwindows.org/
< Back
Install
Cancel


Git 2.20.1 Setup
Completing the Git Setup Wizard
Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.
Click Finish to exit Setup.
Launch Git Bash
View Release Notes
Finish



6.3 GNU/Linux

The first step is installing the toolchain to compile C/C++ code for the ARM Cortex-M3 processor that is used in the Texas Instruments CC2538 micro-controller.

---

sudo add-apt-repository ppa:team-gcc-arm-embedded/ppa

sudo apt-get install gcc-arm-embedded

Once the toolchain is installed, we will also need some additional tools to be able to work with the source code.

---

sudo apt-get install build-essential git scons python-pip

Finally, we will also need to install some Python modules that will be needed

---

sudo pip install intelhex pyserial

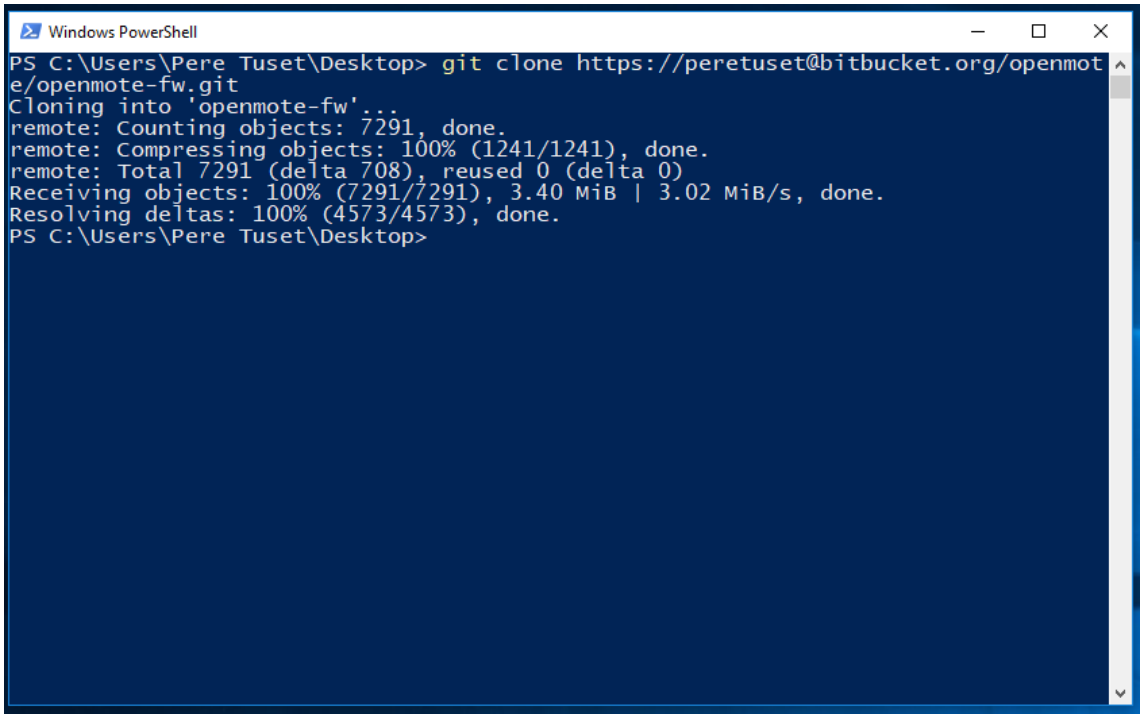
Once all the software has been properly installed in the system we can proceed to program the Open Mote B module.

6.4 OpenMote

The next sub-sections will show how to compile and execute the default OpenMote firmware projects step-by-step with Windows 10 and Ubuntu 18.04 LTS.

6.4.1 Windows 10

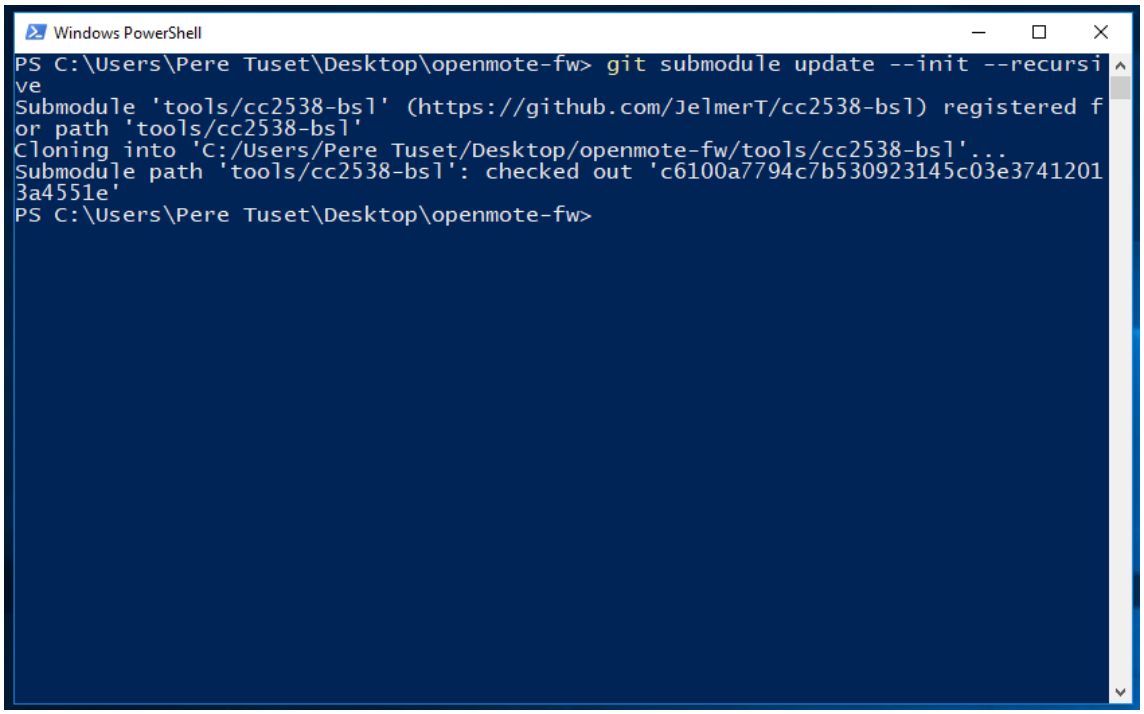
The first step is to clone the OpenMote repository using git. To do so, type the following command:



After the OpenMote repository has been cloned, you need to initialize the repository submodules by executing the following command:

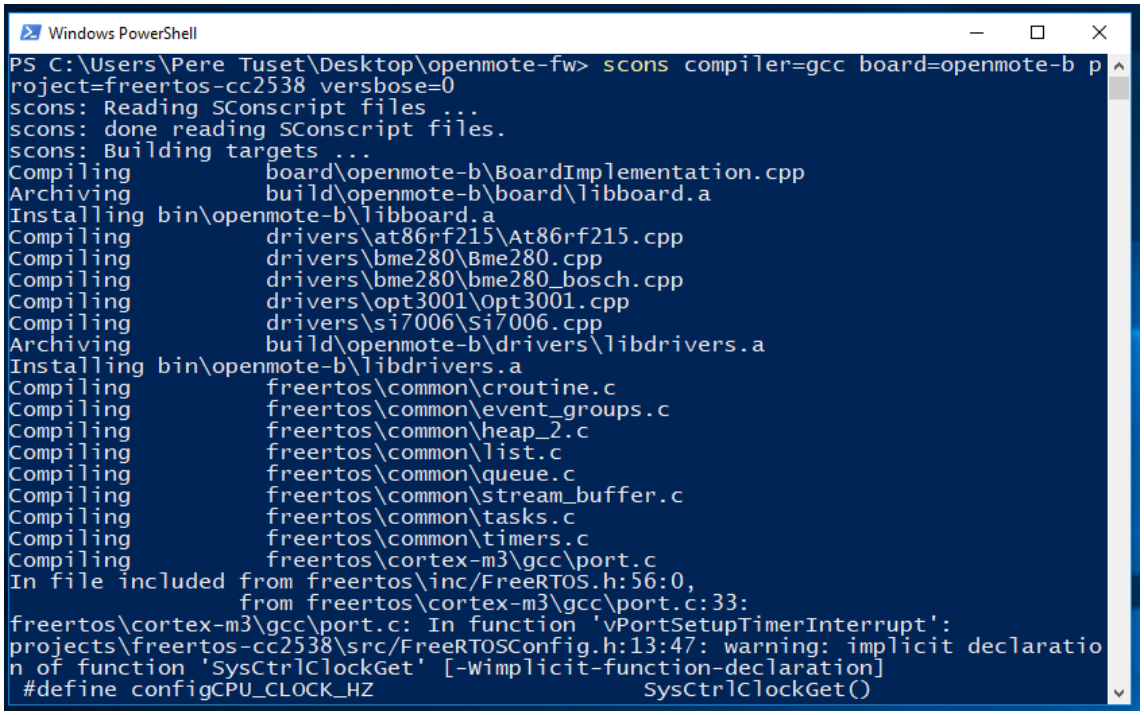
Git submodule update –init –recursive

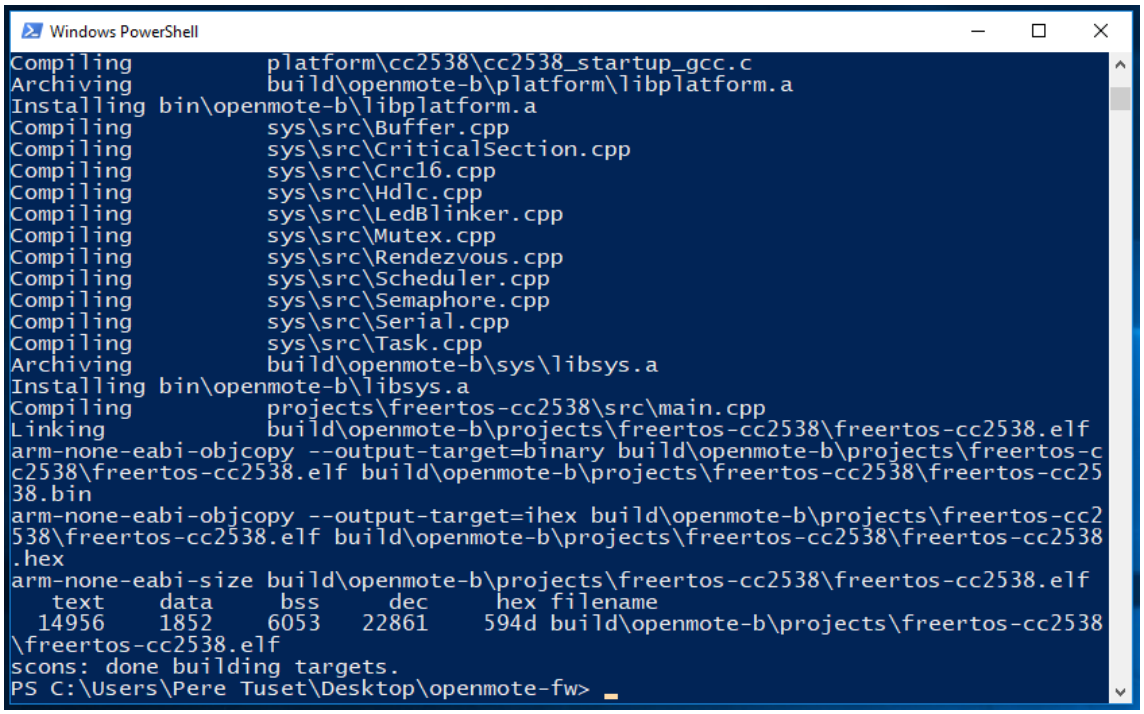
If the command is successful you should see a screen similar to the next one.

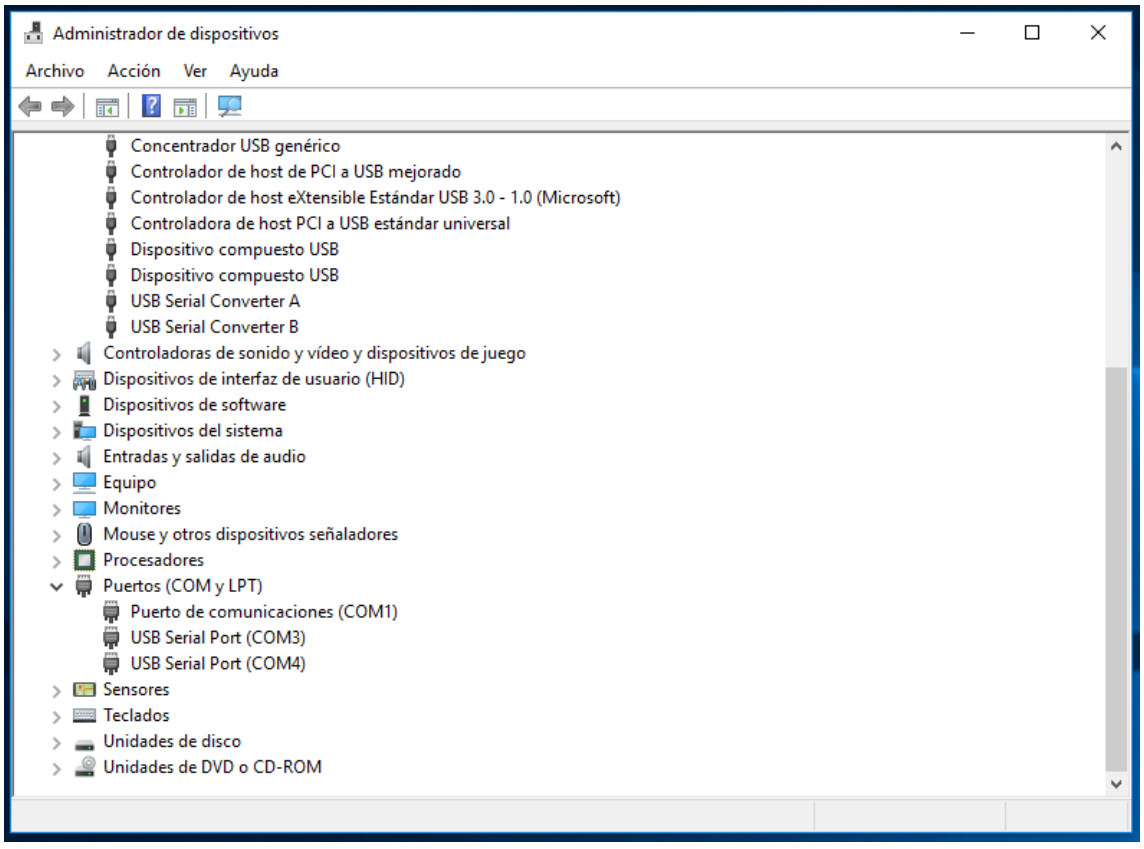


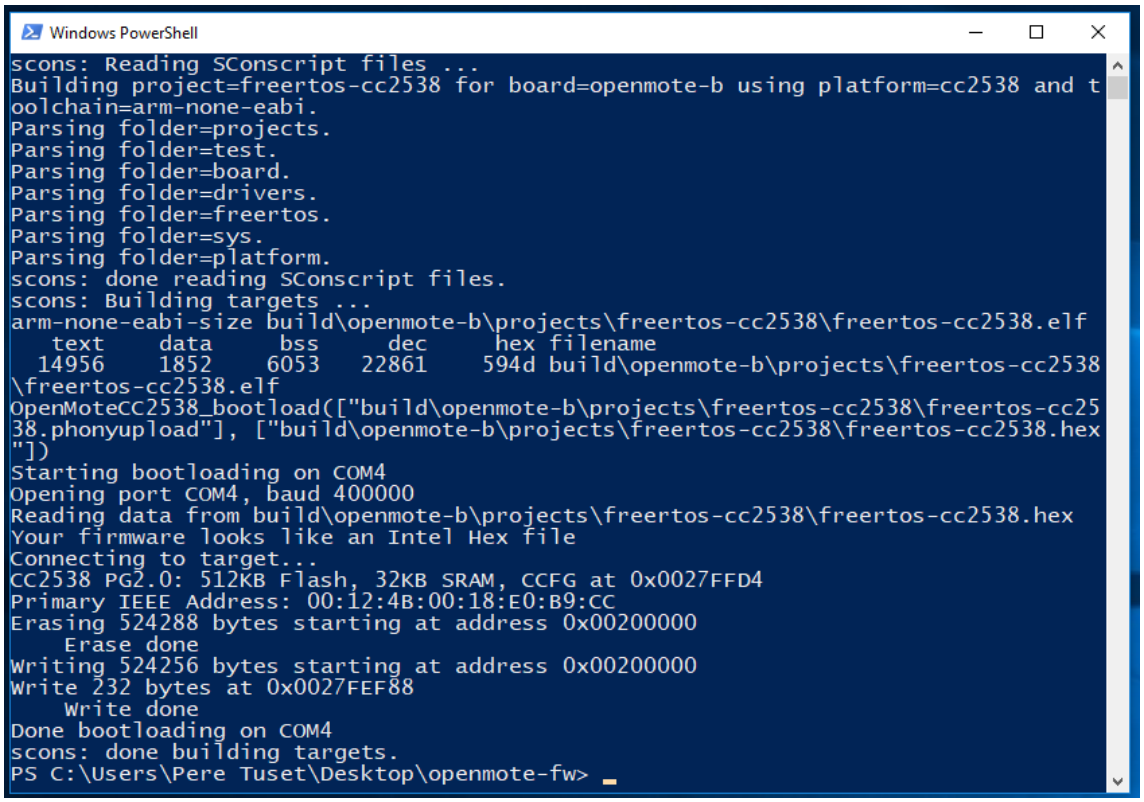
Once the initializations are made you can compile a project by the next command:

scons compiler=gcc board=openmote-b project=freertos-cc2538 versbose=0




Windows PowerShell
Compiling platform\cc2538\cc2538_startup_gcc.c
Archiving build\openmote-b\platform\libplatform.a
Installing bin\openmote-b\libplatform.a
Compiling sys\src\Buffer.cpp
Compiling sys\src\CriticalSection.cpp
Compiling sys\src\Crc16.cpp
Compiling sys\src\Hdlc.cpp
Compiling sys\src\LedBlinker.cpp
Compiling sys\src\Mutex.cpp
Compiling sys\src\Rendezvous.cpp
Compiling sys\src\Scheduler.cpp
Compiling sys\src\Semaphore.cpp
Compiling sys\src\Serial.cpp
Compiling sys\src\Task.cpp
Archiving build\openmote-b\sys\libsys.a
Installing bin\openmote-b\libsys.a
Compiling projects\freertos-cc2538\src\main.cpp
Linking build\openmote-b\projects\freertos-cc2538\freertos-cc2538.elf
arm-none-eabi-objcopy --output-target=binary build\openmote-b\projects\freertos-cc2538\freertos-cc2538.elf build\openmote-b\projects\freertos-cc2538\freertos-cc2538.bin
arm-none-eabi-objcopy --output-target=ihex build\openmote-b\projects\freertos-cc2538\freertos-cc2538.elf build\openmote-b\projects\freertos-cc2538\freertos-cc2538.hex
arm-none-eabi-size build\openmote-b\projects\freertos-cc2538\freertos-cc2538.elf
text data bss dec hex filename
14956 1852 6053 22861 594d build\openmote-b\projects\freertos-cc2538\freertos-cc2538.elf
scons: done building targets.
PS C:\Users\Pere Tuset\Desktop\openmote-fw>


Administrador de dispositivos
Archivo Acción Ver Ayuda
Concentrador USB genérico
Controlador de host de PCI a USB mejorado
Controlador de host eXtensible Estándar USB 3.0 - 1.0 (Microsoft)
Controladora de host PCI a USB estándar universal
Dispositivo compuesto USB
Dispositivo compuesto USB
USB Serial Converter A
USB Serial Converter B
Controladoras de sonido y vídeo y dispositivos de juego
Dispositivos de interfaz de usuario (HID)
Dispositivos de software
Dispositivos del sistema
Entradas y salidas de audio
Equipo
Monitores
Mouse y otros dispositivos señaladores
Procesadores
Puertos (COM y LPT)
Puerto de comunicaciones (COM1)
USB Serial Port (COM3)
USB Serial Port (COM4)
Sensores
Teclados
Unidades de disco
Unidades de DVD o CD-ROM



6.4.2 GNU/Linux

The first step is to clone the OpenMote firmware repository from BitBucket using the following command:

---

git clone https://peretuset@bitbucket.org/openmote/openmote-fw.git

---

```
openmote@openmote:~/Desktop$ git clone https://peretuset@bitbucket.org/openmote/openmote-fw.git
Cloning into 'openmote-fw'...
Password for 'https://peretuset@bitbucket.org':
remote: Counting objects: 6880, done.
remote: Compressing objects: 100% (847/847), done.
remote: Total 6880 (delta 454), reused 0 (delta 0)
Receiving objects: 100% (6880/6880), 2.91 MiB | 2.44 MiB/s, done.
Resolving deltas: 100% (4319/4319), done.
```

cd openmote-fw

git submodule update --init --recursive

```
openmote@openmote:~/Desktop/openmote-fw$ git submodule update --init --recursive
Submodule 'tools/cc2538-bsl' (https://github.com/JelmerT/cc2538-bsl) registered
for path 'tools/cc2538-bsl'
Cloning into '/home/openmote/Desktop/openmote-fw/tools/cc2538-bsl'...
Submodule path 'tools/cc2538-bsl': checked out 'c6100a7794c7b530923145c03e374120
13a4551e'
```

cd ti/cc2538/gcc

make

```
openmote@openmote:~/Desktop/openmote-fw/ti/cc2538/gcc$ make
Building 'libdriverlib' library...
Compiling ./../src/aes.c...
In file included from ./../src/aes.c:47:0:
./../src/aes.c: In function 'AESLoadKey':
./../src/aes.h:112:20: warning: implicit declaration of function 'asm' [-Wimplic
it-function-declaration]
 #define ASM_NOP    asm("NOP")
                    ^
./../src/aes.c:140:9: note: in expansion of macro 'ASM_NOP'
         ASM_NOP;
         ^~~~~~~
Compiling ./../src/systick.c...
Compiling ./../src/pka.c...
Compiling ./../src/interrupt.c...
Compiling ./../src/adc.c...
Compiling ./../src/sys_ctrl.c...
Compiling ./../src/ioc.c...
Compiling ./../src/cpu.c...
Compiling ./../src/ccm.c...
```

After generating the driverlib for the Texas Instruments CC2538 micro-controller, you can return to the root directory using the ‘cd –‘ command.

6.4.3 Blinking LED example

To test the build environment and the boards we will use the freertos-cc2538 project, which demonstrates using the FreeRTOS real-time operating system with the Texas Instruments CC2538 micro-controller on the OpenMote-B board. The code will be compiled with the gcc-arm-embedded toolchain. To compile the code issue the following command:

scons board=openmote-b project=freertos-cc2538 compiler=gcc verbose=0

The compile process for all the project modules will start, as displayed in the next figure.

```
openmote@openmote:~/Desktop/openmote-fw$ scons board=openmote-b project=freertos
-cc2538 compiler=gcc verbose=0
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Building targets ...
Compiling          board/openmote-b/BoardImplementation.cpp
Archiving          build/openmote-b/board/libboard.a
Indexing           build/openmote-b/board/libboard.a
Installing bin/openmote-b/libboard.a
Compiling          drivers/at86rf215/At86rf215.cpp
Compiling          drivers/bme280/Bme280.cpp
Compiling          drivers/bme280/bme280_bosch.cpp
Compiling          drivers/opt3001/Opt3001.cpp
Compiling          drivers/si7006/Si7006.cpp
Archiving          build/openmote-b/drivers/libdrivers.a
Indexing           build/openmote-b/drivers/libdrivers.a
Installing bin/openmote-b/libdrivers.a
```

If the process is correct the build system will display a successful message and will show the space occupied by the generated binary image, as depicted in the next figure.

```
Linking            build/openmote-b/projects/freertos-cc2538/freertos-cc2538
arm-none-eabi-objcopy --output-target=binary build/openmote-b/projects/freertos-
cc2538/freertos-cc2538 build/openmote-b/projects/freertos-cc2538/freertos-cc2538
.bin
arm-none-eabi-objcopy --output-target=ihex build/openmote-b/projects/freertos-cc
2538/freertos-cc2538 build/openmote-b/projects/freertos-cc2538/freertos-cc2538.h
ex
arm-none-eabi-size build/openmote-b/projects/freertos-cc2538/freertos-cc2538
   text    data     bss     dec     hex filename
  12664    2364    6129   21157    52a5 build/openmote-b/projects/freertos-cc253
8/freertos-cc2538
scons: done building targets.
```

Once the binary image has been generated, we can upload it to the OpenMote-B board using the bootloader and the cc2538-bsl Python script. To upload the firmware, issue the following command:

---

sudo scons board=openmote-b project=freertos-cc2538 compiler=gcc verbose=0 bootload=/dev/ttyUSB1

Notice that the command must be prepended with “sudo”, as accessing the serial port in Linux requires super-user permissions.

If the process is successful, the build system will display a success message, as depicted in the next image, and the OpenMote-B should start working: the green LED should blink briefly (100 ms) every second (1000 ms) and the user button should toggle the red LED.

```
openmote@openmote:~/Desktop/openmote-fw$ sudo scons board=openmote-b project=fre
ertos-cc2538 compiler=gcc verbose=0 bootload=/dev/ttyUSB1
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Building targets ...
arm-none-eabi-size build/openmote-b/projects/freertos-cc2538/freertos-cc2538
   text    data     bss     dec     hex filename
  12664    2364    6129   21157    52a5 build/openmote-b/projects/freertos-cc253
8/freertos-cc2538
OpenMoteCC2538_bootload(["build/openmote-b/projects/freertos-cc2538/freertos-cc2
538.phonyupload"], ["build/openmote-b/projects/freertos-cc2538/freertos-cc2538.h
ex"])
Starting bootloading on /dev/ttyUSB1
Done bootloading on /dev/ttyUSB1
scons: done building targets.
```

7 Revision Table

Revision Number	Date	Changes
0	22/08/2019	First implementation

About Industrial Shields:

Address:
Fàbrica del Pont, 1-11
Sant Fruitós de Bages, 08272 (Barcelona)
Spain

Telephone:
(+34) 938 760 191 / (+34) 635 693 611

Mail:
info@industrialshields.com