

Setting Up RISC-V Development Environment for RV32M1-VEGA

1. Introduction

This guide describes the setup for the RISC-V development environment of RV32M1-VEGA board. The RV32M1-VEGA development board is a small, low-power, and cost-effective evaluation and development board for application prototyping and demonstration of the RV32M1 device. The RV32M1 device integrates two RISC-V cores: a RI5CY core and a ZERO_RISCV core. There's a JTAG port used to debug two RISC-V cores. For detailed information on the RV32M1 device, see RV32M1RM document.

The setup for RISC-V development environment of RV32M1 includes installing GNU GCC toolchain, OpenOCD debugger, Eclipse and its configurations for RISC-V, and Software Development Kit (SDK) for RV32M1.

The development system of RV32M1 RISC-V supports Windows, Linux, and MacOS operating system. In each operating system, you can set up a GUI-based development system with the Eclipse IDE or use command line simply if you prefer it.

To set up the development system, you can follow one of the following guides according to your host operation system.

- Windows
- Linux
- macOS

Contents

1.	Introduction.....	1
2.	Development Environment Setup on Windows	2
2.1	Install Requirements and Dependencies.....	2
2.2	Install Eclipse IDE for C/C++ Developers.....	2
2.3	Install Windows Build Tools	2
2.4	Install RV32M1 GNU GCC Toolchain.....	3
2.5	Install OpenOCD	4
2.6	Install RV32M1 SDK.....	6
2.7	Install CMake for Command Line Mode	6
2.8	Install MinGW for Command Line Mode.....	7
3.	Development Environment Setup on Linux.....	8
3.1	Install RV32M1 GNU GCC Toolchain and CMake	8
3.2	Install OpenOCD	8
3.3	Install Eclipse IDE for C/C++ Developers.....	9
4.	Development Environment Setup on MacOS	9
4.1	Install RV32M1 GNU GCC Toolchain and CMake	9
4.2	Install OpenOCD	10
5	References.....	10
6	Revision history	10

2. Development Environment Setup on Windows

This section describes how to set up the RISC-V development environment for RV32M1 in a Windows environment.

2.1 Install Requirements and Dependencies

RV32M1 uses the OpenOCD debugger that can work on the connection from Telnet Client to debug the application. The Windows Telnet Client needs to be installed if you use OpenOCD and telnet to debug/run an application.

To install Telnet Client, run following command using the `cmd.exe` command prompt:

```
pkgmgr /iu:"TelnetClient"
```

2.2 Install Eclipse IDE for C/C++ Developers

To install Eclipse IDE for C/C++ developers, follow these steps:

1. Download the GNU MCU Eclipse IDE for C/C++ from [GitHub releases](#).
 - Download the latest available version
 - Select the `-win32.win32.x86` file for x32 Windows and `-win32.win32.x86_64` for x64 Windows
2. Unzip the downloaded package at a place of your choice, for example, root directory of D.

2.3 Install Windows Build Tools

This step is optional. Skip this section if you never use Eclipse external builder, otherwise you need to GNU MCU Eclipse Windows Build Tools. Because `make` and `rm` executable binaries are required by the external builder.

Follow these steps to install Windows Build Tools on your Windows host system.

1. Download the GNU MCU windows Build Tools from the [GitHub Releases](#).
 - Download the latest available version
 - Select the `-win32` file for x32 Windows and `-win64` for x64 Windows
2. Unzip the package at the *Eclipse* folder created in the [step 2.1](#).
 - Although possible to install the build tools in any folder, it's recommended to install it under Eclipse unzipped folder for clean and tidy purpose.
3. Go to the *Eclipse* menu → **Windows** → **Preferences** to set *Global Build Tools Path* to its installed location as showed in the [Figure 1](#).

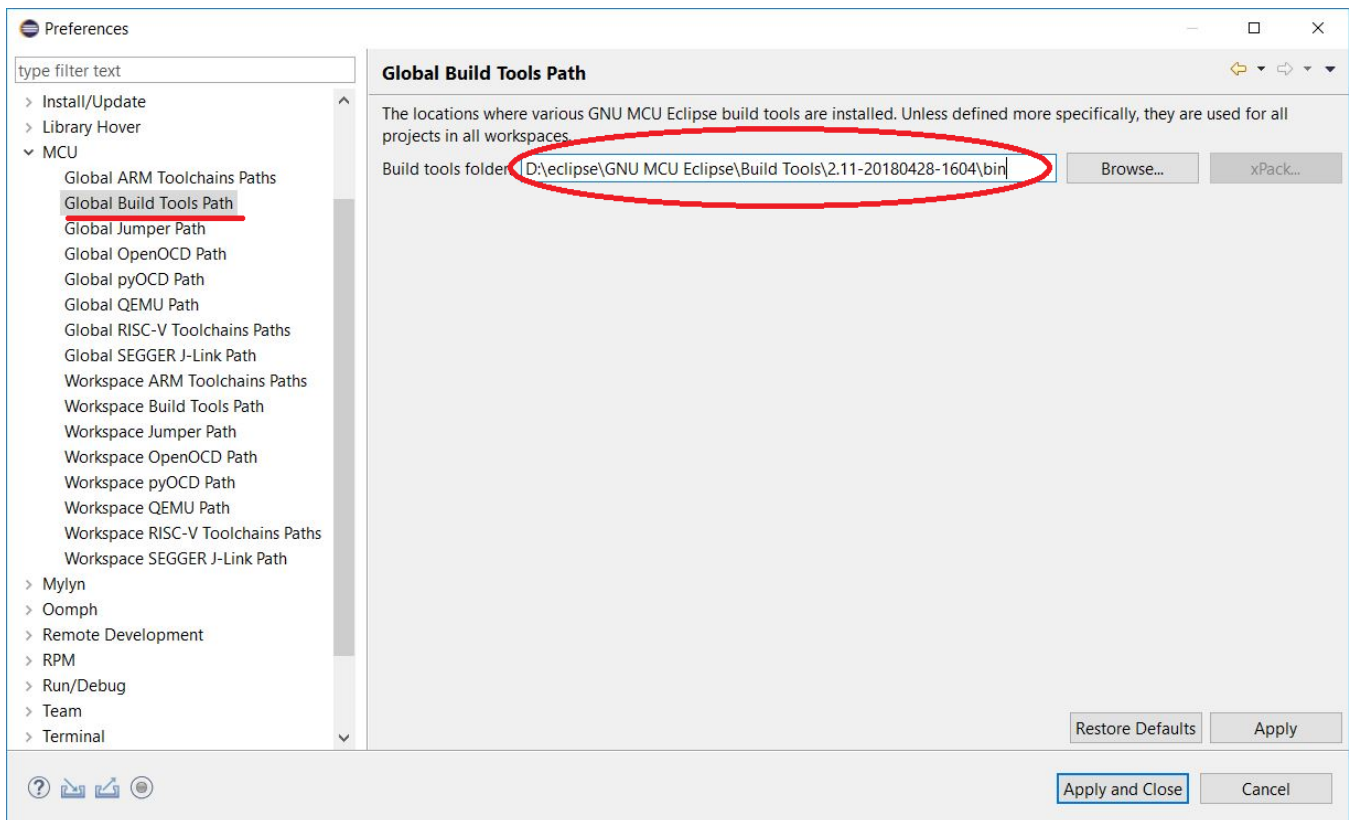


Figure 1. set Global Build Tools Path to the installed location

2.4 Install RV32M1 GNU GCC Toolchain

www.open-isa.org provides the pre-built GNU GCC toolchain for RV32M1 device. It's the RISC-V C and C++ cross-compiler and supports the Newlib library for use on the RV32M1-based embedded system. Follow the steps below to set it up:

1. Download the pre-built RV32M1 GNU Compiler toolchain from the www.open-isa.org website.
2. Unzip the downloaded package at the *GNU MCU Eclipse* folder created in the [step 2.2](#).
 - a) Although possible to install the toolchain in any folder, it's recommended to install it under *D:\clipse\GNU MCU Eclipse* folder for clean and tidy purpose.
3. Go to the *Eclipse* menu **Windows** → **Preferences** to set *Global RISC-V Toolchains Paths* to its installed location as showed in the [Figure 2](#).
4. Create an environment variable `RISCV32GCC_DIR` and set it to GNU GCC Toolchain folder, for example, *D:\eclipse\GNU MCU Eclipse\riscv32-unknown-elf-gcc*.
5. Add GNU GCC Toolchain executable files folder, *D:\eclipse\GNU MCU Eclipse\riscv32-unknown-elf-gcc/bin*, to the user or system `PATH` variable.

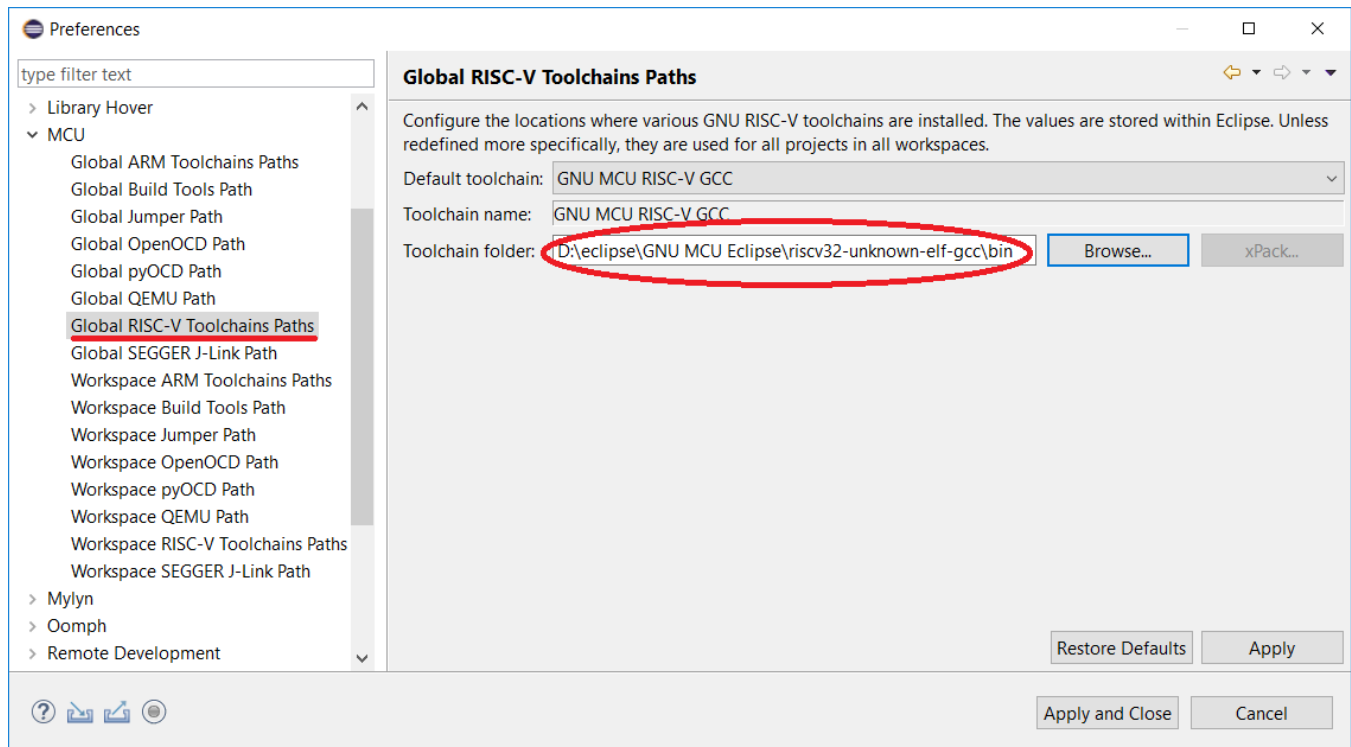


Figure 2. set Global RISC-V Toolchains Paths to the installed location

2.5 Install OpenOCD

www.open-isa.org provides the pre-built OpenOCD binary file and the driver to support J-link adapter hardware. Follow the steps below to install OpenOCD debugger:

1. Download the pre-build OpenOCD and J-Link driver from the www.open-isa.org website.
2. Unzip the downloaded package at the *GNU MCU Eclipse* folder created in the [step 2.2](#).
 - a) Although possible to install the toolchain in any folder, it's recommended to install it under *D:\clipse\GNU MCU Eclipse* folder for clean and tidy purpose.
3. Go to the *Eclipse* menu **Windows** → **Preferences** to set *Global OpenOCD Path* to its installed location as showed in the [Figure 3](#).
4. Add the path of OpenOCD.exe, for example, *D:\eclipse\GNU MCU Eclipse\openocd\bin* to the environment variable PATH.
5. Plug in the J-Link hardware on Windows PC
6. Run *D:\eclipse\GNU MCU Eclipse\openocd\drivers\drivers\UsbDriverTool.exe*
7. Right click the "J-link driver", and select "Install WinUSB" to install USB driver as showed in [Figure 4](#)
 - a) After the USB driver installed, the J-Link could not be recognized by SEGGER software anymore. For recovery, run *UsbDriverTool.exe* and click "Restore default driver" as showed in the [Figure 5](#).

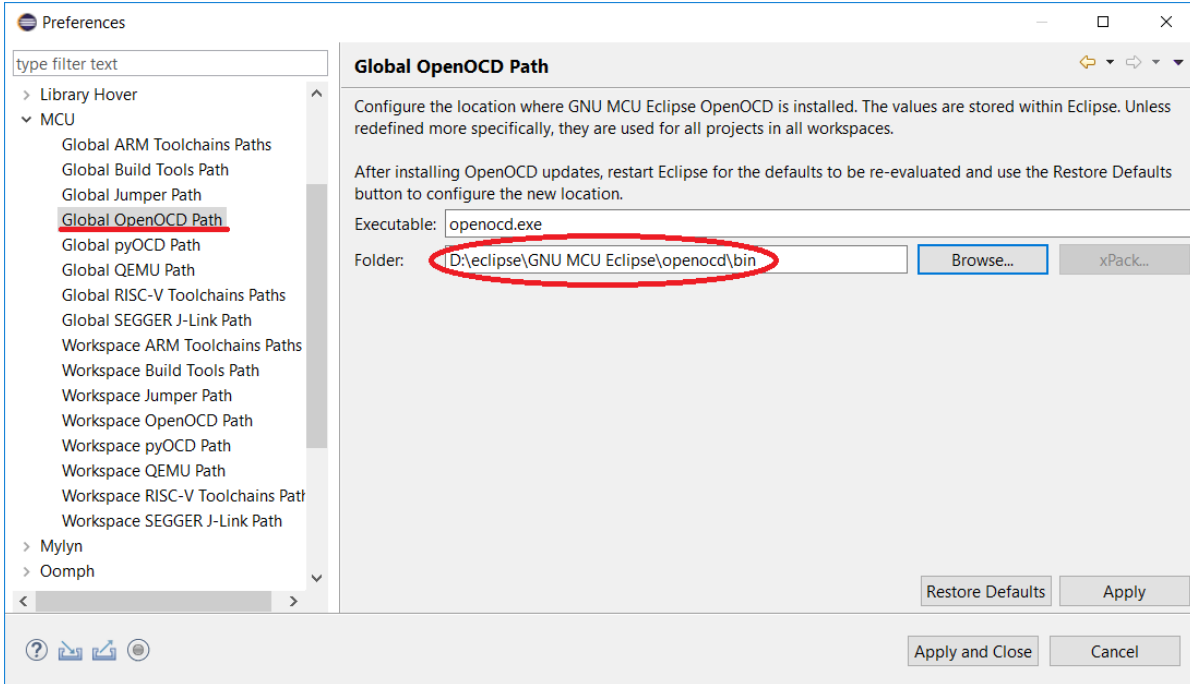


Figure 3. set Global OpenOCD Path to the installed location

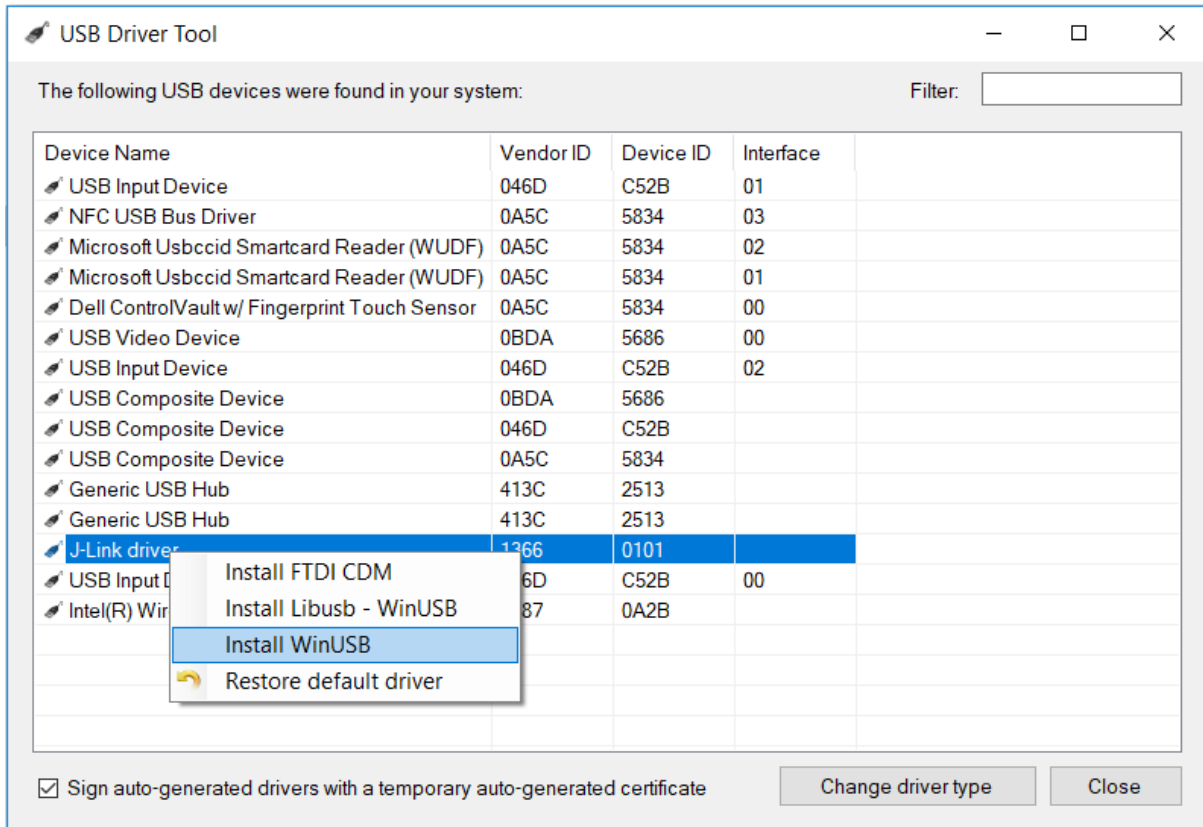


Figure 4. Install J-Link USB driver

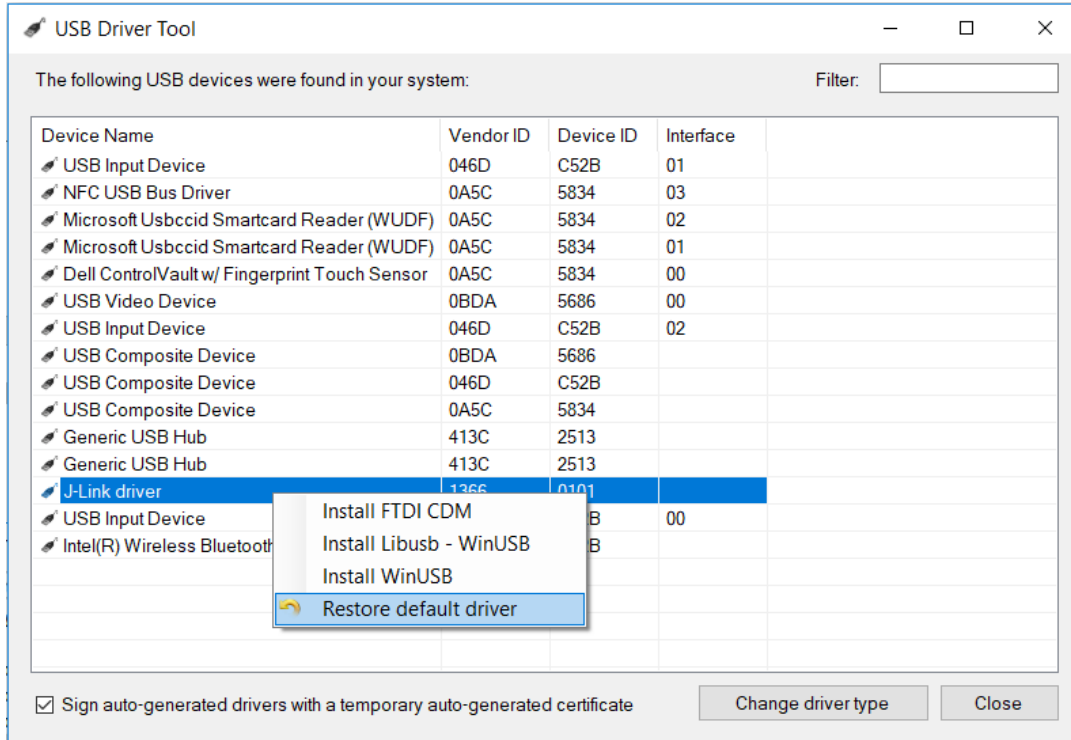


Figure 5. Restore J-Link USB driver

2.6 Install RV32M1 SDK

Follow these steps to install the RV32M1 SDK on your Windows host system.

1. Download the latest SDK from the www.open-isa.org website
2. Unzip the downloaded package at a place of your choice.

Now you can use Eclipse IDE to import the RV32M1 SDK example, build, debug, program, and run the application.

If you prefer to use command line to build, debug, and run an application, additional software tools are needed. The following sub-sections describe how to install additional software tools for command line build environment.

2.7 Install CMake for Command Line Mode

Download the latest CMake from <https://cmake.org/download/>. Select the `-win32-x86` or `-win64-x64` version according to your Windows system.

Suggest using Windows installer file to install CMake and ensure that the option "Add CMake to system PATH" is selected when installing. The user chooses to select whether it is installed into the PATH for all users or just the current user.

If downloading zip file and installing CMake manually, you need to add it to the PATH variable after installation.

2.8 Install MinGW for Command Line Mode

The Minimalist GNU for Windows (MinGW) development tools provide a set of tools that are not dependent on third party C-Runtime DLLs (such as Cygwin). The build environment used by the SDK does not utilize the MinGW build tools but does leverage the base install of both MinGW and MSYS. MSYS provides a basic shell with a Unix-like interface and tools.

1. Download the latest MinGW mingw-get-setup installer from sourceforge.net/projects/mingw/files/Installer/.
2. Run the installer. The recommended installation path is C:\MinGW, however, you may install to any location. Note: **the installation path cannot contain any spaces**.
3. Ensure that the “mingw32-base” and “msys-base” are selected under Basic Setup [Figure 6](#).

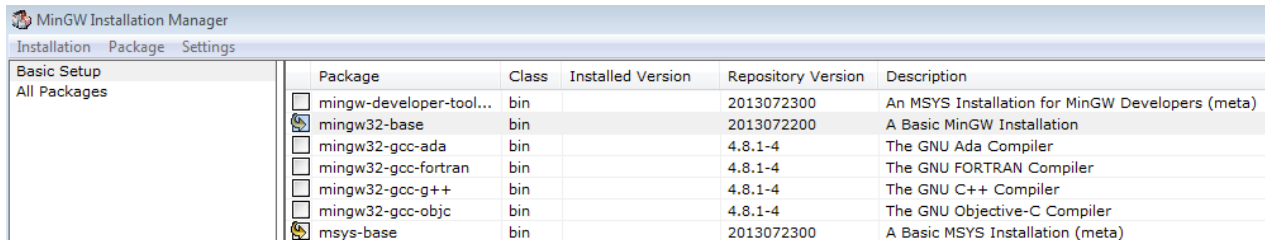


Figure 6. Setup MinGW and MSYS

4. Click “Apply Changes” in the “Installation” menu and follow the remaining instructions to complete the installation as showed in the [Figure 7](#).

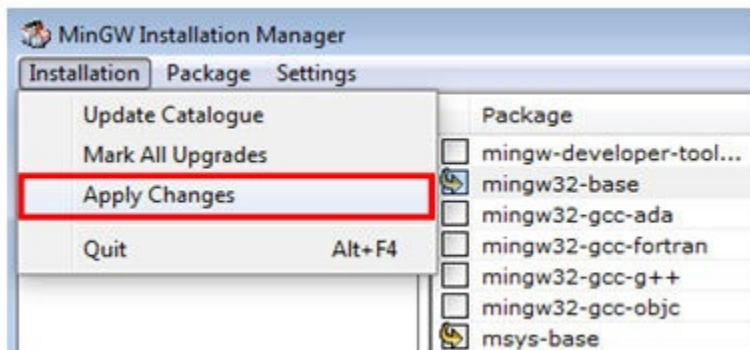


Figure 7. Complete MinGW and MSYS installation

5. Add `<mingw_install_dir>\bin` to PATH environment variable.

3. Development Environment Setup on Linux

The section describes the steps to install the necessary software tools required to build and run a RV32M1 SDK demo application on Linux operating system. All steps are verified on Ubuntu 18.04.

3.1 Install RV32M1 GNU GCC Toolchain and CMake

1. Install requirements and Dependencies

```
sudo apt-get install autoconf automake autotools-dev curl libmpc-dev libmpfr-dev
libgmp-dev gawk build-essential bison flex texinfo gperf libtool patchutils bc
zlib1g-dev libusb-1.0-0-dev libudev libudev1 libudev-dev g++
export LD_LIBRARY_PATH=/usr/local/lib
```

2. Clone and build the RV32M1 GNU Compiler toolchain.

```
git clone -recursive https://github.com/pulp-platform/pulp-riscv-gnu-toolchain
cd pulp-riscv-gnu-toolchain
./configure --prefix=/opt/pulp --with-arch=rv32imc --with-cmodel=medlow --enable-
multilib
sudo make
```

The toolchain is installed to /opt/pulp. However, you may install to any location and update the system variables accordingly described below.

www.open-isa.org also provides the pre-built RV32M1 GCC Toolchain. User can download the package and extract and install the toolchain.

3. Install CMake

```
sudo apt-get install cmake
```

4. Update system variables

```
export PATH=$PATH:/opt/pulp/bin
export RISC32GCC_DIR="/opt/pulp"
```

3.2 Install OpenOCD

1. Install requirements and Dependencies

```
sudo apt-get install autoconf automake autotools-dev curl libmpc-dev libmpfr-dev
libgmp-dev gawk build-essential bison flex texinfo gperf libtool patchutils bc
zlib1g-dev libusb-1.0-0-dev libudev libudev1 libudev-dev g++
```


2. www.open-isa.org also provides the pre-built OpenOCD binary for RV32M1. Download it and extract the file. Ensure the install location is added in the PATH system variable.

3.3 Install Eclipse IDE for C/C++ Developers

To install Eclipse IDE for C/C++ developers, follow these steps:

1. Download the GNU MCU Eclipse IDE for C/C++ from [GitHub releases](https://github.com/mcu-eclipse/mcu-eclipse/releases).
 - Download the latest available version
 - Select the `-linux.gtk.x86` file for x32 Linux and `-linux.gtk.x86_64` for x64 Linux
2. Unpack the archive and move the Eclipse folder at a place of your choice.
3. Set the *Global RISC-V Toolchains Paths* and *Global OpenOCD Path* to the correct folders used in the 3.1 and 3.2 subsections.

4. Development Environment Setup on MacOS

The section describes the steps to install the necessary software tools required to build and run a RV32M1 SDK demo application on Linux operating system. All steps are verified on Ubuntu 18.04.

4.1 Install RV32M1 GNU GCC Toolchain and CMake

1. Install requirements and Dependencies

```
brew install gawk gnu-sed gmp mpfr libmpc isl zlib
```

2. Clone and build the RV32M1 GNU Compiler toolchain.

```
git clone -recursive https://github.com/pulp-platform/pulp-riscv-gnu-toolchain
cd pulp-riscv-gnu-toolchain
./configure --prefix=/opt/riscv --with-arch=rv32imc --with-cmodel=medlow --
enable-multilib
make
```

The toolchain is installed to `/opt/pulp`. However, you may install to any location and update the system variables accordingly described below.

www.open-isa.org also provides the pre-built RV32M1 GCC Toolchain. User can download the package and extract and install the toolchain.

3. Install CMake

```
brew install cmake
```

4. Update system variables

```
export PATH=$PATH:/opt/pulp/bin
export RISC32GCC_DIR="/opt/pulp"
```

4.2 Install OpenOCD

1. Install requirements and Dependencies

```
sudo apt-get install autoconf automake autotools-dev curl libmpc-dev libmpfr-dev
libgmp-dev gawk build-essential bison flex texinfo gperf libtool patchutils bc
zlib1g-dev libusb-1.0-0-dev libudev libudev1 libudev-dev g++
```

2. www.open-isa.org also provides the pre-built OpenOCD binary for RV32M1. Download it and extract the file. Ensure the install location is added in the PATH system variable.

5 References

Following references are available on www.open-isa.org:

- *RV32M1-VEGA-SCH: Schematics*
- *RV32M1-VEGA-LAYOUT: Layout*
- *RV32M1_VEGA_Board_User_Guide*
- *RV32M1RM: Reference Manual*
- *RV32M1DS: Datasheet*

6 Revision history

Rev.	Date	Substantive change(s)
0	11/2018	Initial release



VEGÅ*