Summary of https://gnu-mcu-eclipse.github.io/install/ below:

- Overview here: https://gnu-mcu-eclipse.github.io/install/
- Java from here https://www.java.com/en/download/win10.jsp
  - If you run 64 bit windows make sure to install 64 bit JRE from here https://java.com/en/download/manual.jsp
  - Install and test
- Gnu Arm toolchain from here, Win32 exe https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads
  - Execute downloaded file. Tick “add to path”.
  - Installs in C:\Program Files (x86)\GNU Tools ARM Embedded\7 2017-q4-major
  - See https://gnu-mcu-eclipse.github.io/toolchain/arm/install/
- Download Gnu Arm Windows build tools from here https://github.com/gnu-mcu-eclipse/windows-build-tools/releases
  - Copy all .exe files from “bin” directory from downloaded .zip to C:\Program Files (x86)\GNU Tools ARM Embedded\7 2017-q4-major\bin
  - See https://gnu-mcu-eclipse.github.io/windows-build-tools/install/
- Install J-Link from here http://www.segger.com/jlink-software.html (If you have J-Link..)
  - See https://gnu-mcu-eclipse.github.io/debug/jlink/install/
  - Download from here https://www.segger.com/downloads/jlink/
  - Will install in C:\Program Files (x86)\SEGGER\JLink_V630d
- Install Git for Windows from here https://git-scm.com/download/win
- Install OpenOCD as explained here https://gnu-mcu-eclipse.github.io/blog/2018/01/23/openocd-v0-10-0-7-20180123-released/
  - Install node.js “TLS” from here https://nodejs.org/en/
  - Open Command line window in win10. Enter

```
npm install xpm --global
```

This installs xpm package manager executable

- Open Command line window in win 10. Enter

```
xpm install @gnu-mcu-eclipse/openocd --global
```

to install Gnu MC Eclipse openOCD

- Install Quemu as explained here https://gnu-mcu-eclipse.github.io/qemu/install/
  - Download latest Windows exe from here https://github.com/gnu-mcu-eclipse/qemu/releases
  - Will install in C:\Program Files\GNU ARM Eclipse\QEMU\2.8.0-201612271623-dev
- Get latest MCU Eclipse from here https://github.com/gnu-mcu-eclipse/org.eclipse.epp.packages/releases/
  - Extract downloaded zip file to c:\ (otherwise path names may become too long)
- Run GNU MCU Eclipse
  - Follow work space preferences “global tool chain path” in https://gnu-mcu-eclipse.github.io/eclipse/workspace/preferences/
  - Follow tool chain path management in https://gnu-mcu-eclipse.github.io/toolchain/path/

Install PAckages CMSIS

- See https://gnu-mcu-eclipse.github.io/plugins/packs-manager/

- In Eclipse Help→Install new software, Work with: “GNU MCU Eclipse plug-ins”, Click “What is already installed”
    - Check that “GNU MCU C/C++ Packs (Experimental)” are installed, or install them
    - In Eclipse goto C/C++ packs perspective (hover over toolbar icons to find the icon with two boxes in orange yellow)
    - In this perspectiv, above middle window, click on the icon with the two yellow arrows, to update the packages definitions from all repositories
    - If any warnings that certain packages cannot be downloaded please click “ignore” to continue, as long as it does not concern the STMicro packages we are interested in - STM32F7 and STM32H7
    - When finished, in left window, choose STMMicroelectronics
        - Install STM32F7 pack
            - Ín left window, click once on STM32F7 series.
            - Then select package in middle window
            - then above middle window click yellow box icon to install a local copy of the selected package
            - You will see status message “Install Packs” in bottom right of Eclipse window
        - Install STM32H7 in the same way

Set active project, check MCU set correctly

- See https://gnu-mcu-eclipse.github.io/eclipse/project/assign-device/
    - Set/check that active project is correct
    - Set/check that device is correct:
        - For OVI40 with F7 MCU: Project → Properties → C/C++ Build → Settings → Devices → STM32F767ZI
        - For OVI40 with H7 MCU: Project → Properties → C/C++ Build → Settings → Devices → STM32H743ZI

Using Git with GNU-MCU-Eclipse

Option: Github Desktop

- Download Github desktop from https://desktop.github.com/
- Start Github Desktop, enter your Githu username and password, clone repository “https://github.com/df8oe/UHSDR” * Optional: create local branch of cloned repo for tests * In Eclipse, create new workspace pointing to local clone of cloned Github UHSDR repository * Check Eclipse settings for * active build config * MCU device used * …

Option: git commandline

- create directory where UHSDR repository should reside, e.g. d:\uhsdr

```
git clone https://github.com/df8oe/UHSDR
```

Import git UHSDR project into Eclipse installation

- In Eclipse, choose “File→Import”.
- .. then choose “General→Existing Projects into Workspace”.
- Then choose the folder “mchf-eclipse in the path where you just downloaded the zip file as “root-directory”
- The project is now recognized and can be imported
- Next: check Eclipse settings

From:
https://amateurfunk-sulingen.de/wiki/ - **Afu - Wiki des DARC OV Sulingen I40**

Permanent link:
https://amateurfunk-sulingen.de/wiki/doku.php?id=en:uhsdr_dev:win10_toolchain&rev=1519553436

Last update: **25.02.2018 10:10**