Using OpenOCD server and Eclipse CDT IDE

(a simple tutorial for Windows)

What will you need?

1. Download the OpenOCD package form Olimex. It contains drivers for all Olimex FTDI based JTAGs supported by OpenOCD, a tested compilation of the OpenOCD executable and sample configuration scripts - http://www.olimex/dev/
2. Download Eclipse IDE for C/C++ Developers - http://www.eclipse.org/downloads/
3. Download the latest version of YAGARTO GNU ARM toolchain - http://www.yagarto.de/
4. Download the latest version of YAGARTO Tools - http://www.yagarto.de/
5. If not installed yet you will need a Java Runtime Environment (JRE) by Sun/Oracle - http://www.java.com/en/download/index.jsp

Installing Device Drivers

Please follow the instructions in **README.txt** file in the drivers folder included in the OpenOCD package by Olimex. There is nothing specific about the installation process if you have installed Windows drivers before.

Installing Eclipse IDE (and JRE)

Again, if you haven't installed JRE yet please do so now.

Eclipse doesn't really need installation you should only unpack the .zip file somewhere in your file system.

Note: *A practically good idea is to keep your OpenOCD distribution in a separate folder. Something as simple as C:\OpenOCD\ is preferred, because there are known issues with folders that include spaces, or long paths.*

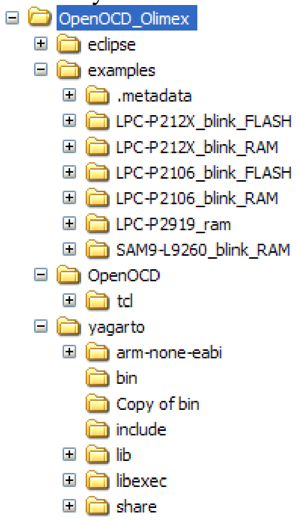
Installing YAGARTO GNU ARM toolchain and YAGARTO Tools

Just follow the instruction on the install packages. Remember to leave the checkbox to add an entry to the system path variable checked, otherwise Eclipse won't find your make and compiler executables.

Installing OpenOCD

Again nothing to install. Just copy the OpenOCD folder (part of Olimex's package) inside your distribution folder.

For the sample projects to work without modification you need the following directory structure:

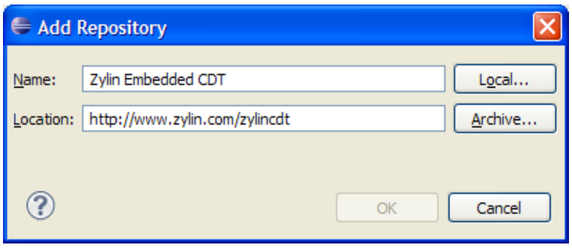


Adding Zylin Embedded CDT plugin

This is a general purpose GDB debug plug-in. It should be added to Eclipse in order to use OpenOCD server and be able to make debug configurations.

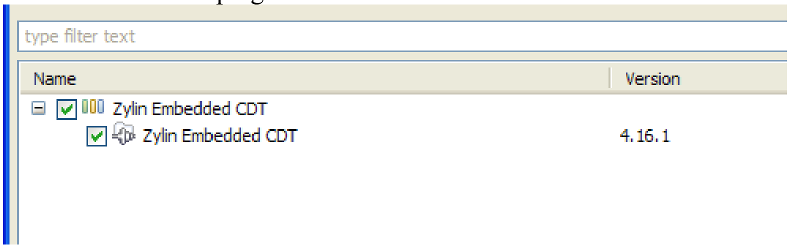
Open Eclipse, go to **Help>Install new software...**, then click on the **Add...** button and add the following location:

http://www.zylin.com/zylincdt



Name: Zylin Embedded CDT
Location: http://www.zylin.com/zylincdt

Continue by selecting the checkbox for the plug-in:



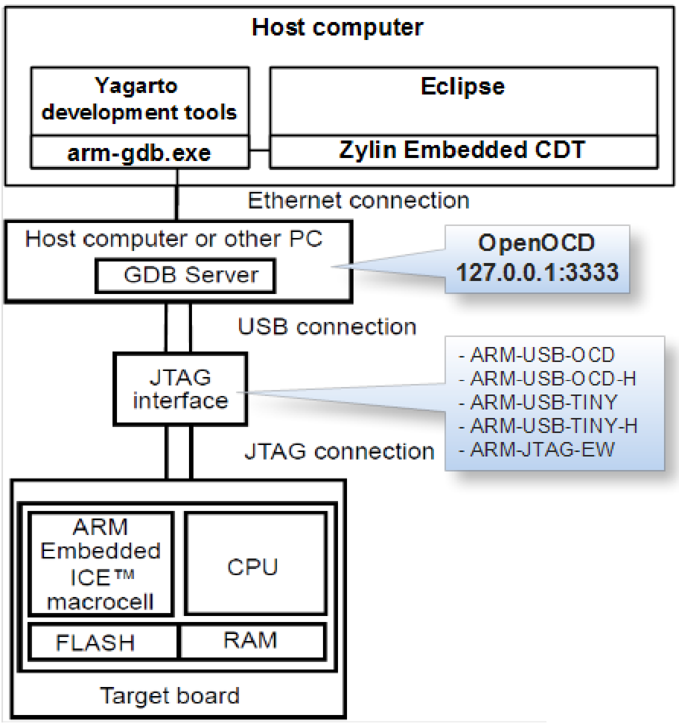
Name	Version
☑ Zylin Embedded CDT	
☑ Zylin Embedded CDT	4.16.1

Press **Next** a couple of times and you should end up with Zylin plugin downloaded and installed.

Opening a project with Eclipse

With everything set up already it is now time to get something done in fact. You should have a **target board** and **JTAG adapter** to continue. Also a **sample project** is required to actually perform target debugging. Sample projects should be available in Olimex's site. You can as well create a custom project from scratch, but using a sample project as reference is always an advantage.

This is an overview of your entire configuration:

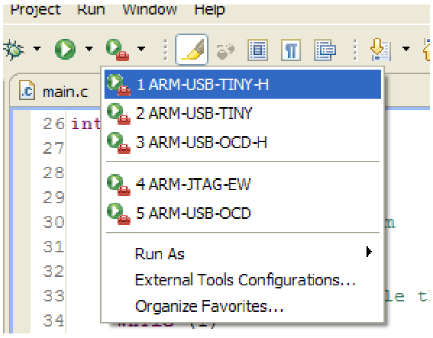


OpenOCD Server communication overview

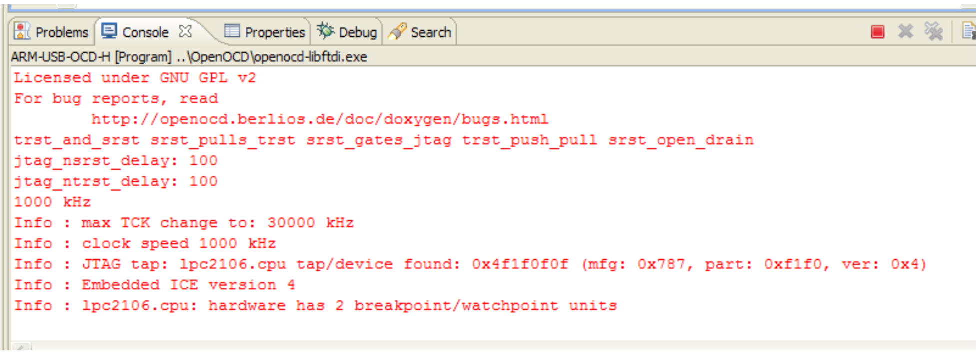
We start by opening the sample project. Start Eclipse and you'll be prompted to enter a directory that contains a workspace to use. Navigate to your project directory and continue.

With the cross compiler on your default paths system variable you should be able to compile and link the project and get an output file.

Connect your JTAG adapter to the PC. Connect the JTAG adapter to the target board. Power the board if needed. Now start an external tool configuration from the list (matching your adapter):



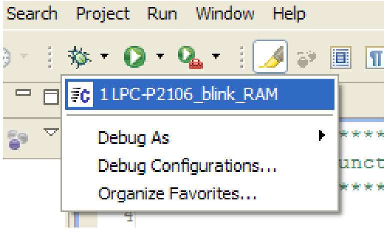
If everything goes well you should get an opened console like the following:



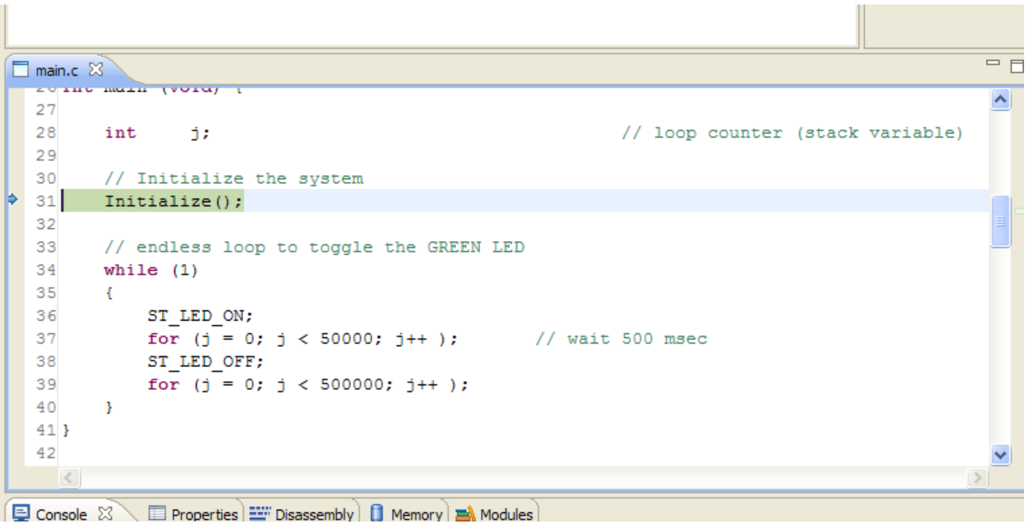
```
ARM-USB-OCD-H [Program] ..\OpenOCD\openocd-libftdi.exe
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.berlios.de/doc/doxygen/bugs.html
trst_and_srst srst_pulls_trst srst_gates_jtag trst_push_pull srst_open_drain
jtag_nsrst_delay: 100
jtag_ntrst_delay: 100
1000 kHz
Info : max TCK change to: 30000 kHz
Info : clock speed 1000 kHz
Info : JTAG tap: lpc2106.cpu tap/device found: 0x4f1f0f0f (mfg: 0x787, part: 0xf1f0, ver: 0x4)
Info : Embedded ICE version 4
Info : lpc2106.cpu: hardware has 2 breakpoint/watchpoint units
```

This means that your JTAG is connected to the target and is ready to execute commands.

Let's now start a debug configuration. Again choose from the list (if no configuration is available in this drop down menu, click on Debug Configurations..., find the elusive debug configuration and press the Debug button):



Your output file should now be loaded into the target. Eclipse switches automatically to Debug perspective and you should end up with a look like this:



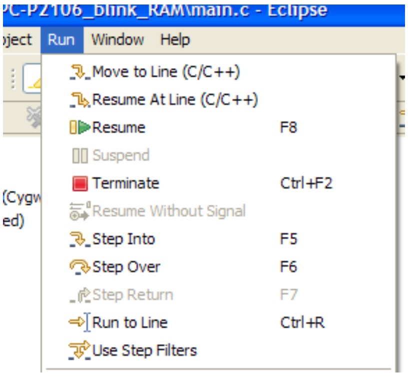
```
    int     j;                                  // loop counter (stack variable)

    // Initialize the system
    Initialize();

    // endless loop to toggle the GREEN LED
    while (1)
    {
        ST_LED_ON;
        for (j = 0; j < 50000; j++ );       // wait 500 msec
        ST_LED_OFF;
        for (j = 0; j < 500000; j++ );
    }
}

```

Using the Run menu you should be able to debug your code:



Conclusion

Please note that OpenOCD is sometimes pretty tricky to work with so be aware of configuration and runtime issues. Well, if things get very ugly you should stop the server, reconnect the JTAG adapter and start over.

If in need of more information check out any of the following:

- http://openocd.berlios.de/web/ - the official web page of the OpenOCD project.
- http://www.iar.com/ - home of the IAR development tools.
- http://www.olimex/dev/ - manufacturer of development boards and tools. That's us :).

Revision history

- 13 December 2010 – initial beta release.

THE END