Application Note

AN_325

FT90x Toolchain Installation Guide

Version 1.03

Issue Date: 2016-09-19

This guide documents the tools and methods required for building, programming and debugging the FT90x series devices from FTDI.

Future Technology Devices International Limited (FTDI)
Unit 1, 2 Seaward Place, Glasgow G41 1HH, United Kingdom
Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758
Web Site: http://ftdichip.com

Table of Contents

1 FT90x Toolchain Introduction

The free FT90x toolchain is a port from the popular GNU toolchain which includes the following components:

- GCC based compiler
- GNU Binary Utilities (binutils) based tools, most notably:
  - as - the assembler
  - ld - the linker
  - and some other useful tools such as objdump, ar, ranlib, addr2line, etc.
- GDB based debugger
- In addition, a plugin for the Eclipse IDE is also provided. This plugin allows the FT90x toolchain to integrate seamlessly into Eclipse and as a result, greatly simplifies the development works for the FT90x MCUs.

1.1 Compiler: ft32-elf-gcc

The FT90x compiler is used similarly to standard GCC. It supports most GCC options such as -Wall, -O1, -O2…

Example: To compile a C file into an object file:

```
ft32-elf-gcc -c -o file.o file.c
```

1.2 Assembler: ft32-elf-as

The FT90x assembler functions in the same way as the standard GNU assembler (GAS). The assembly files should be written using the GAS general syntax.

Example: To compile an assembly file into an object file:

```
ft32-elf-as -o file.o file.s
```

1.3 Linker: ft32-elf-ld

Typically running behind ft32-elf-gcc, the FT90x linker performs two tasks. It first links all object files and libraries into a.out and then convert's a.out into an executable file for FT90x. Similar to the FT90x compiler and assembler, the FT90x linker supports most standard GNU linker options.

Example:

- To link various object files / libraries into an .elf file:

  ```
  ft32-elf-gcc -nostartfiles file1.o file2.o -L <libfolder> -l lib1 -l lib2 -o file.elf
  ```
- To convert file.elf into a FT90x binary file, which can be programmed into the chips:

  ```
  ft32-elf-ld --oformat binary -o file.bin file.elf
  ```

1.4 Debugger: ft32-elf-gdb

The FTDI programmer/debugger module is needed for the communication between ft32-elf-gdb and the chip. The communication follows the GDB remote protocol. In addition to the debugger module, two software components are needed:

- GDB Bridge: for converting GDB commands into the debugger module commands
- Bootloader: for receiving & executing the debugger module commands

More information on how to use the FT90x debugger can be found in section 5.1.3 of this document.

1.5 A useful utility: ft32-elf-objdump

ft32-elf-obj dump displays various information about object files. Its usage is the same as standard GNU objdump.

Example: To disassemble file.elf into a text file

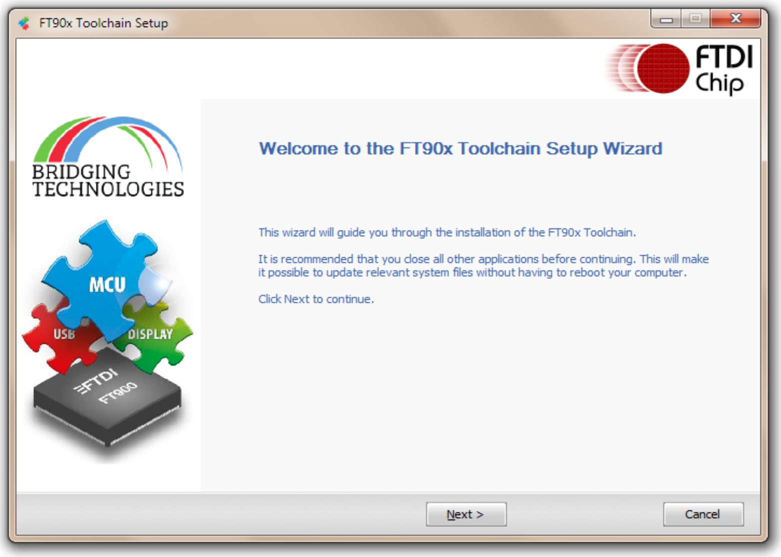
```
ft32-elf-objdump -d file.elf > disassembly.txt
```

2 Setting up the FT90x Toolchain

2.1 Installing the toolchain

The toolchain can be installed by running the setup wizard "FT90x Toolchain Setup_*version*.exe", which can be downloaded from the FTDI website. Please follow the steps in the wizard to complete the installation process. It is recommended to use the default settings for simplicity.

Note: all applications should be closed before the installation or a restart may be required.



Welcome to the FT90x Toolchain Setup Wizard

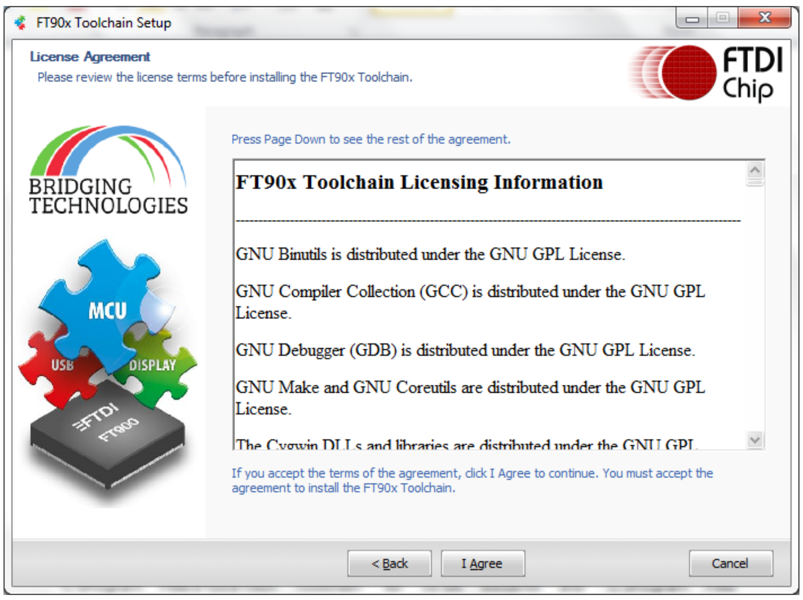
This wizard will guide you through the installation of the FT90x Toolchain.

It is recommended that you close all other applications before continuing. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 1 Toolchain Setup Wizard Dialog box

In the License Agreement dialog box, click **I Agree**.



License Agreement
Please review the license terms before installing the FT90x Toolchain.

Press Page Down to see the rest of the agreement.

FT90x Toolchain Licensing Information

GNU Binutils is distributed under the GNU GPL License.

GNU Compiler Collection (GCC) is distributed under the GNU GPL License.

GNU Debugger (GDB) is distributed under the GNU GPL License.

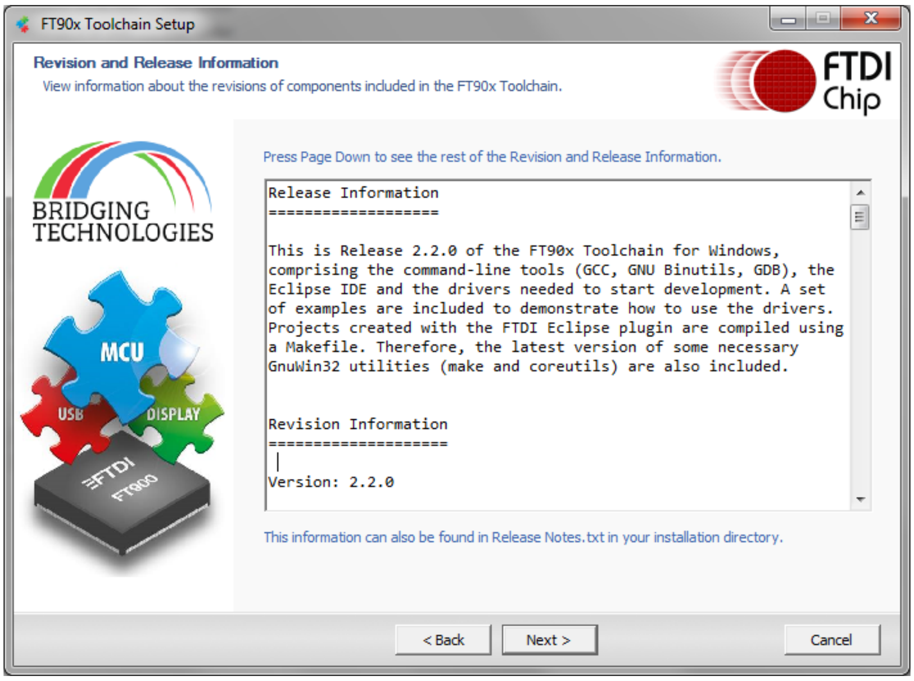
GNU Make and GNU Coreutils are distributed under the GNU GPL License.

The Cygwin DLLs and libraries are distributed under the GNU GPL

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install the FT90x Toolchain.

Figure 2 License Agreement Dialog box

Go through the Revision and Release information and click **Next**.



Revision and Release Information
View information about the revisions of components included in the FT90x Toolchain.

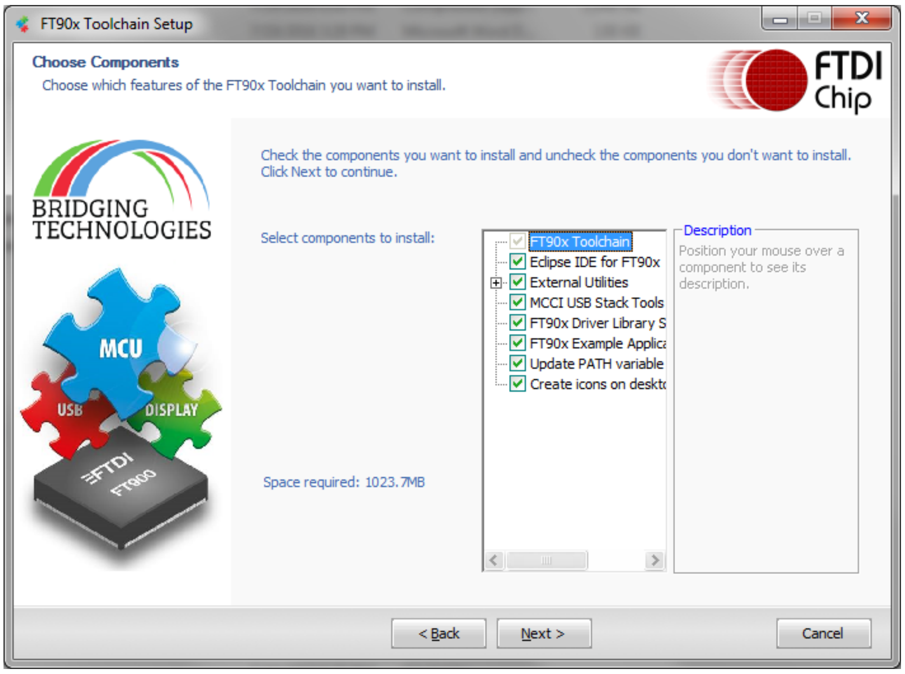
Press Page Down to see the rest of the Revision and Release Information.

```
Release Information
===================

This is Release 2.2.0 of the FT90x Toolchain for Windows,
comprising the command-line tools (GCC, GNU Binutils, GDB), the
Eclipse IDE and the drivers needed to start development. A set
of examples are included to demonstrate how to use the drivers.
Projects created with the FTDI Eclipse plugin are compiled using
a Makefile. Therefore, the latest version of some necessary
GnuWin32 utilities (make and coreutils) are also included.


Revision Information
====================

Version: 2.2.0
```

This information can also be found in Release Notes.txt in your installation directory.

Figure 3 Revision and Release Information Dialog box

Select the Components and click **Next**.



Choose Components
Choose which features of the FT90x Toolchain you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

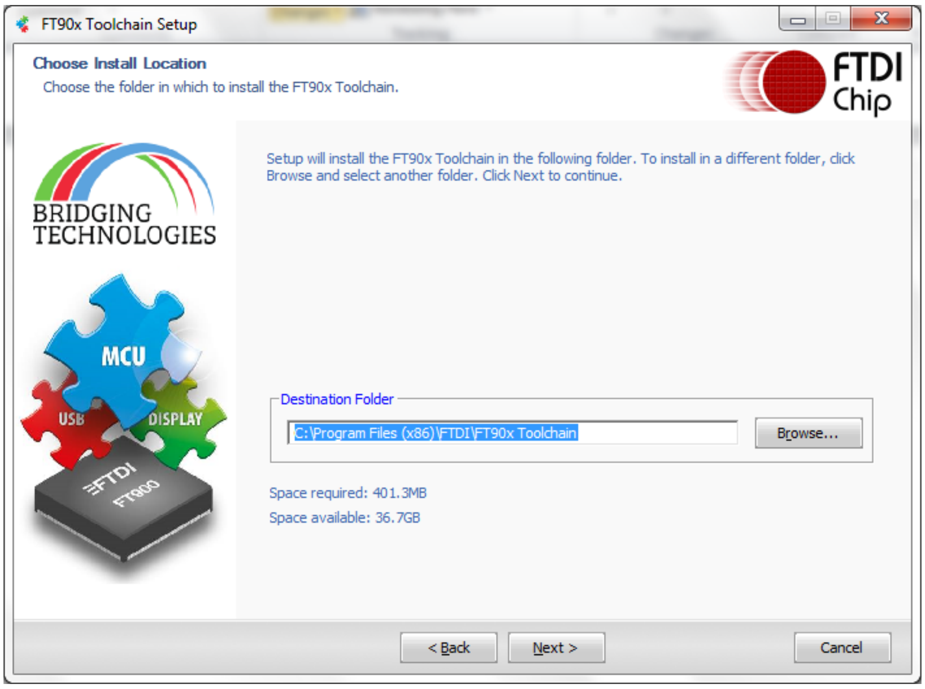
- [✓] FT90x Toolchain
- [✓] Eclipse IDE for FT90x
- [✓] External Utilities
- [✓] MCCI USB Stack Tools
- [✓] FT90x Driver Library S
- [✓] FT90x Example Applica
- [✓] Update PATH variable
- [✓] Create icons on deskt

Description
Position your mouse over a component to see its description.

Space required: 1023.7MB

Figure 4 Components Dialog box

Click **Browse** and select a different file path for FT90x Toolchain installation. Alternately, continue installing in the specified folder by clicking **Next**.


FT90x Toolchain Setup

Choose Install Location

Choose the folder in which to install the FT90x Toolchain.

Setup will install the FT90x Toolchain in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

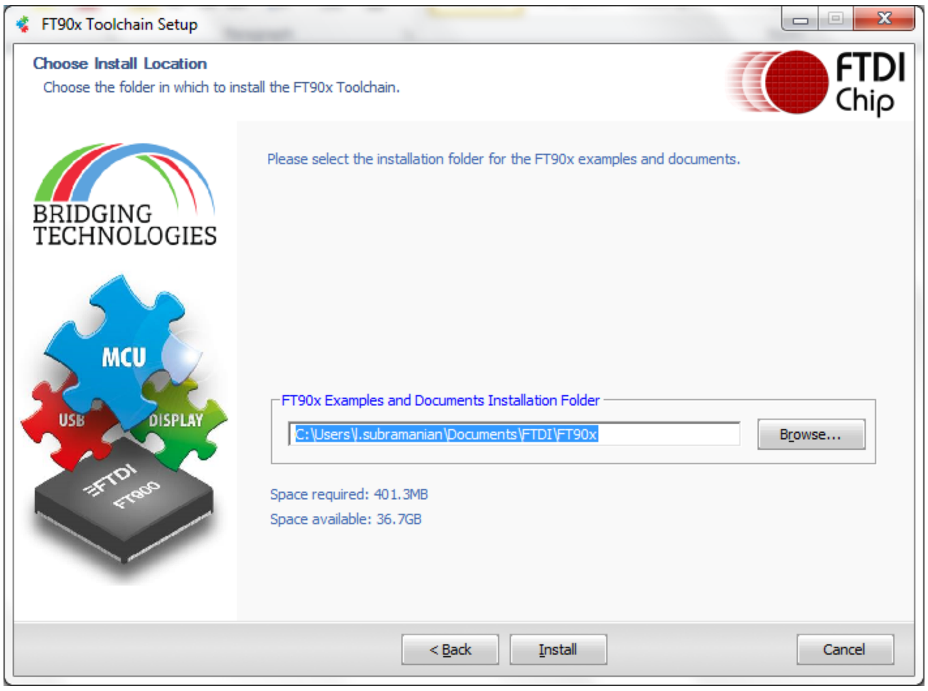
Destination Folder: C:\Program Files (x86)\FTDI\FT90x Toolchain

Space required: 401.3MB

Space available: 36.7GB

Figure 5 FT90x Toolchain Install Location Dialog box

Click **Browse** and select a different file path for installing FT90x examples and documents. Alternately, continue installing in the specified folder, by clicking **Install**.


FT90x Toolchain Setup

Choose Install Location

Choose the folder in which to install the FT90x Toolchain.

Please select the installation folder for the FT90x examples and documents.

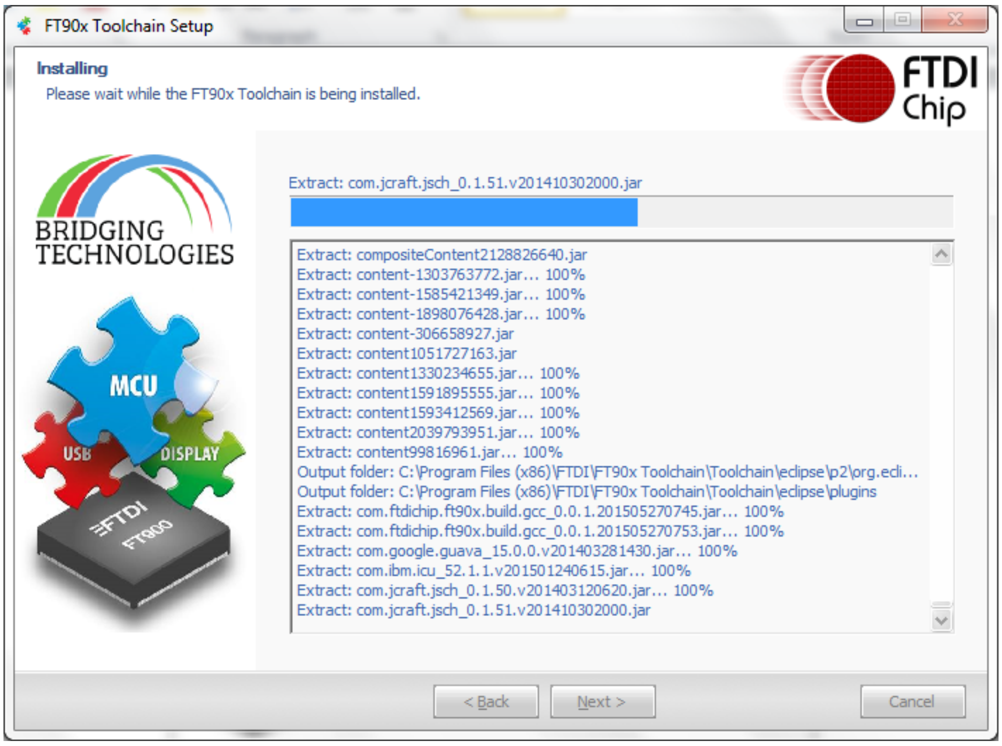
FT90x Examples and Documents Installation Folder: C:\Users\l.subramanian\Documents\FTDI\FT90x

Space required: 401.3MB

Space available: 36.7GB

Figure 6 FT90x Toolchain-Examples & Documents Install Location Dialog box

The FT90x Toolchain installation progress bar is displayed.

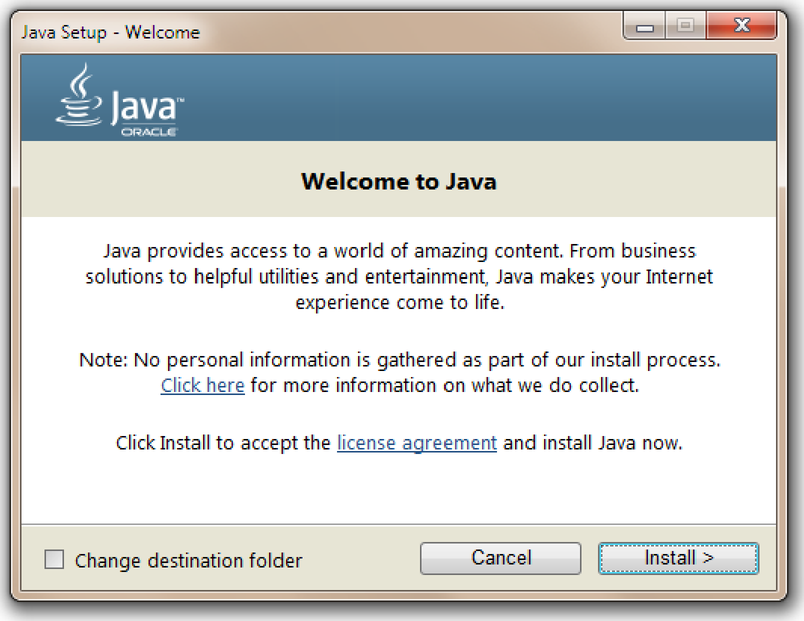


```
Extract: com.jcraft.jsch_0.1.51.v201410302000.jar
```

```
Extract: compositeContent2128826640.jar
Extract: content-1303763772.jar... 100%
Extract: content-1585421349.jar... 100%
Extract: content-1898076428.jar... 100%
Extract: content-306658927.jar
Extract: content1051727163.jar
Extract: content1330234655.jar... 100%
Extract: content1591895555.jar... 100%
Extract: content1593412569.jar... 100%
Extract: content2039793951.jar... 100%
Extract: content99816961.jar... 100%
Output folder: C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\eclipse\p2\org.ecli...
Output folder: C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\eclipse\plugins
Extract: com.ftdichip.ft90x.build.gcc_0.0.1.201505270745.jar... 100%
Extract: com.ftdichip.ft90x.build.gcc_0.0.1.201505270753.jar... 100%
Extract: com.google.guava_15.0.0.v201403281430.jar... 100%
Extract: com.ibm.icu_52.1.1.v201501240615.jar... 100%
Extract: com.jcraft.jsch_0.1.50.v201403120620.jar... 100%
Extract: com.jcraft.jsch_0.1.51.v201410302000.jar
```

Figure 7 **FT90x Toolchain - Installation Progress Window**

If Java is selected for installation, the following window is displayed.



Welcome to Java

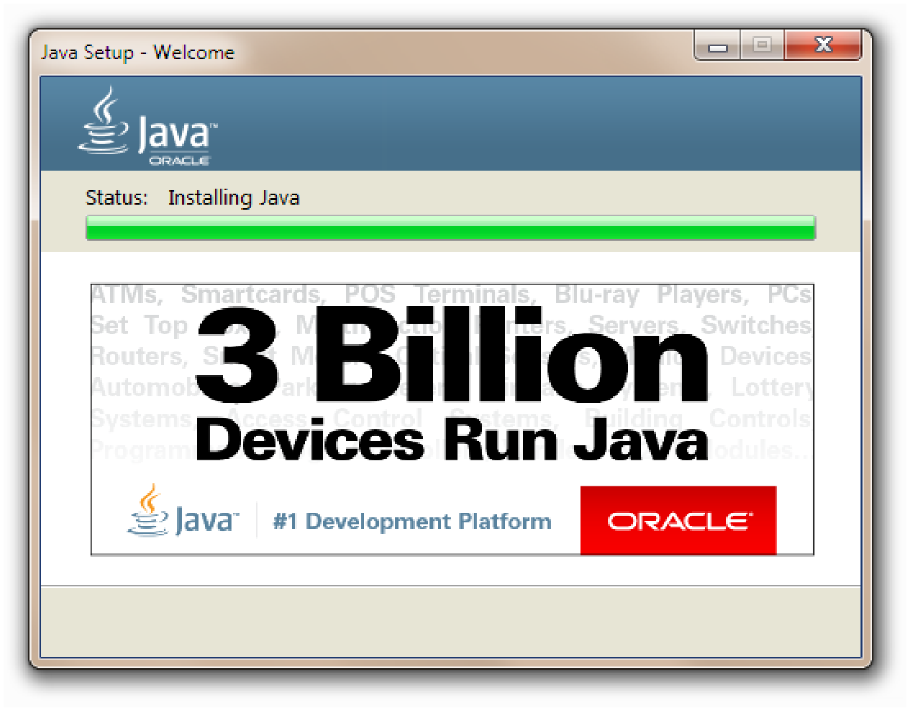
Java provides access to a world of amazing content. From business solutions to helpful utilities and entertainment, Java makes your Internet experience come to life.

Note: No personal information is gathered as part of our install process. Click here for more information on what we do collect.

Click Install to accept the license agreement and install Java now.

Figure 8 Java Setup Window

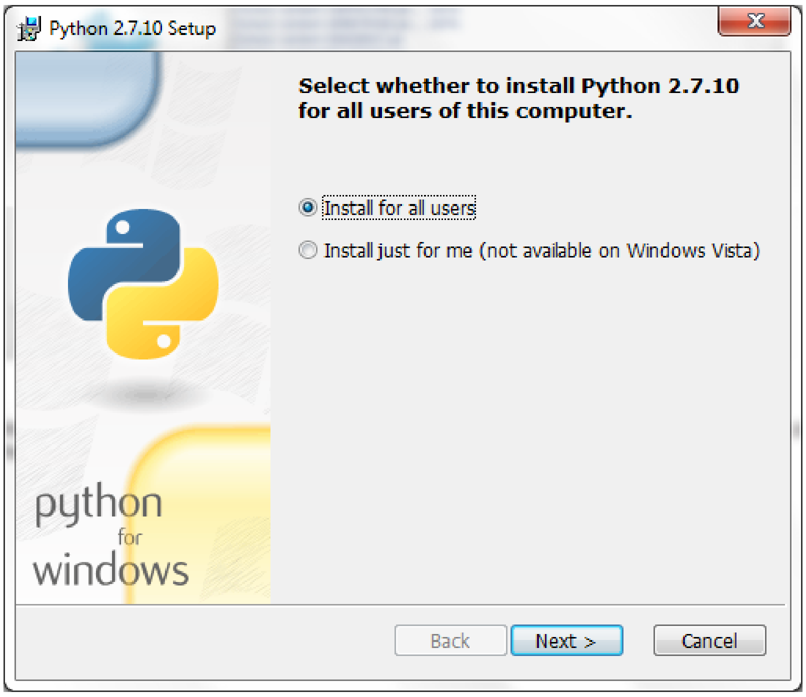
Click **Install** and follow the instructions to install Java on the machine.



Status: Installing Java

Figure 9 Java Setup Progress Window

During installation, if a **Python 2.7.10 Setup** dialog box is displayed, select the appropriate option as required and click **Next**.

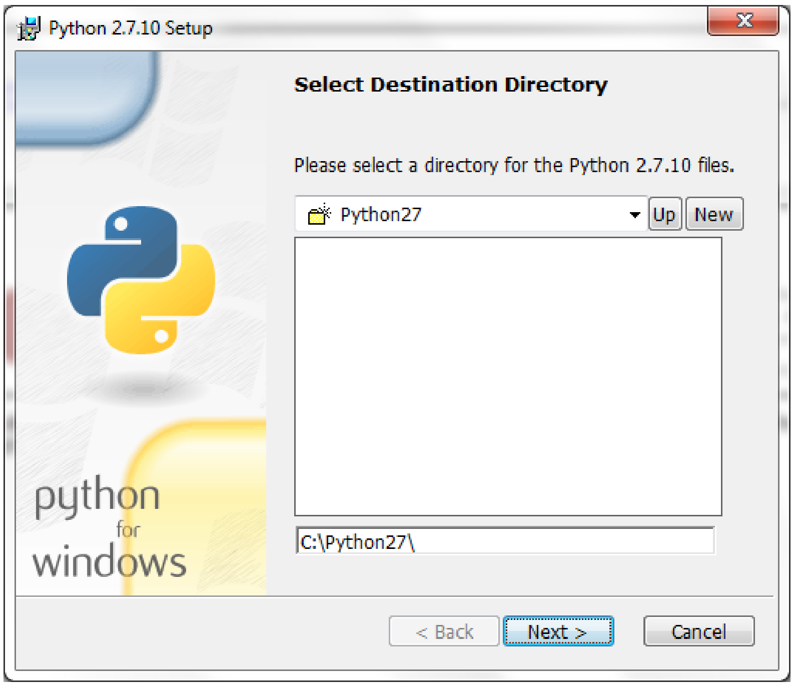


Select whether to install Python 2.7.10 for all users of this computer.

(●) Install for all users
( ) Install just for me (not available on Windows Vista)

Figure 10 Python Setup Dialog box

Select a different Destination Directory to setup Python. Alternately, continue installing in the specified folder by clicking **Next**.


Python 2.7.10 Setup

Select Destination Directory

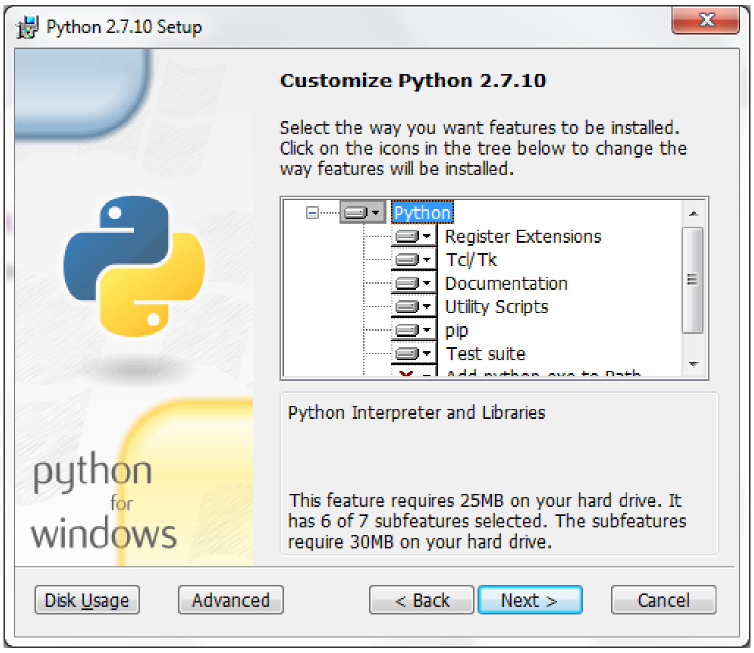
Please select a directory for the Python 2.7.10 files.

Python27

C:\Python27\

Figure 11 Destination Directory Selection Dialog box for Python Setup

Select the Python features to be installed or continue with the default features and click **Next**.


Python 2.7.10 Setup

Customize Python 2.7.10

Select the way you want features to be installed.
Click on the icons in the tree below to change the way features will be installed.

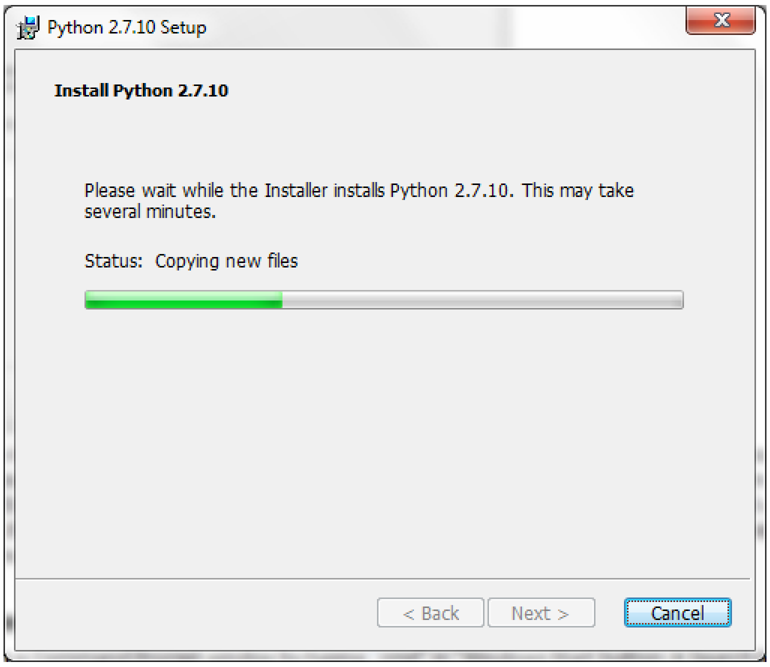
- Python
  - Register Extensions
  - Tcl/Tk
  - Documentation
  - Utility Scripts
  - pip
  - Test suite
  - Add python.exe to Path

Python Interpreter and Libraries

This feature requires 25MB on your hard drive. It has 6 of 7 subfeatures selected. The subfeatures require 30MB on your hard drive.

Figure 12 Python Features Customization Dialog box

Python installation progress bar is displayed.



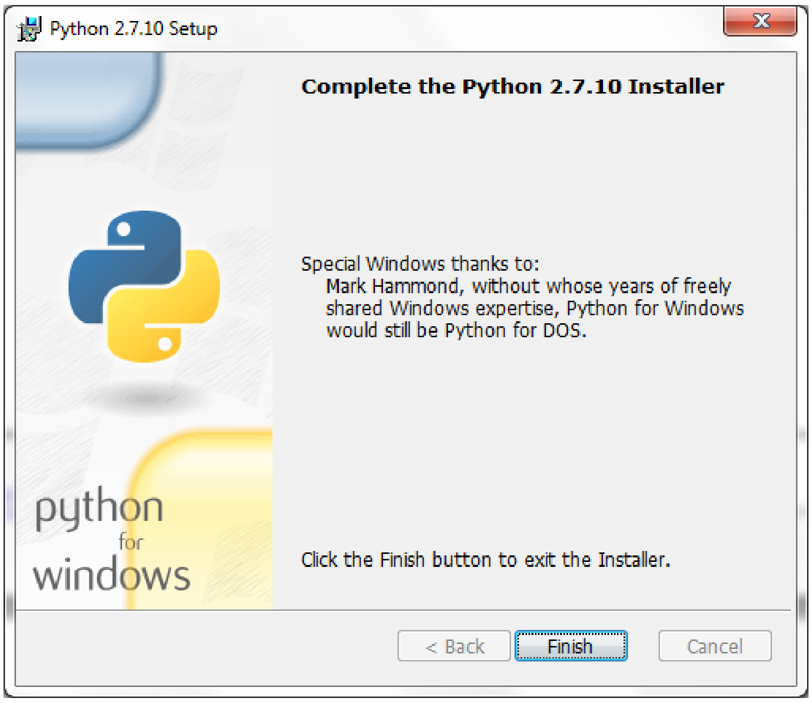
Install Python 2.7.10

Please wait while the Installer installs Python 2.7.10. This may take several minutes.

Status: Copying new files

Figure 13 Python Installation Progress Dialog box

Click **Finish** to complete the Python installation.



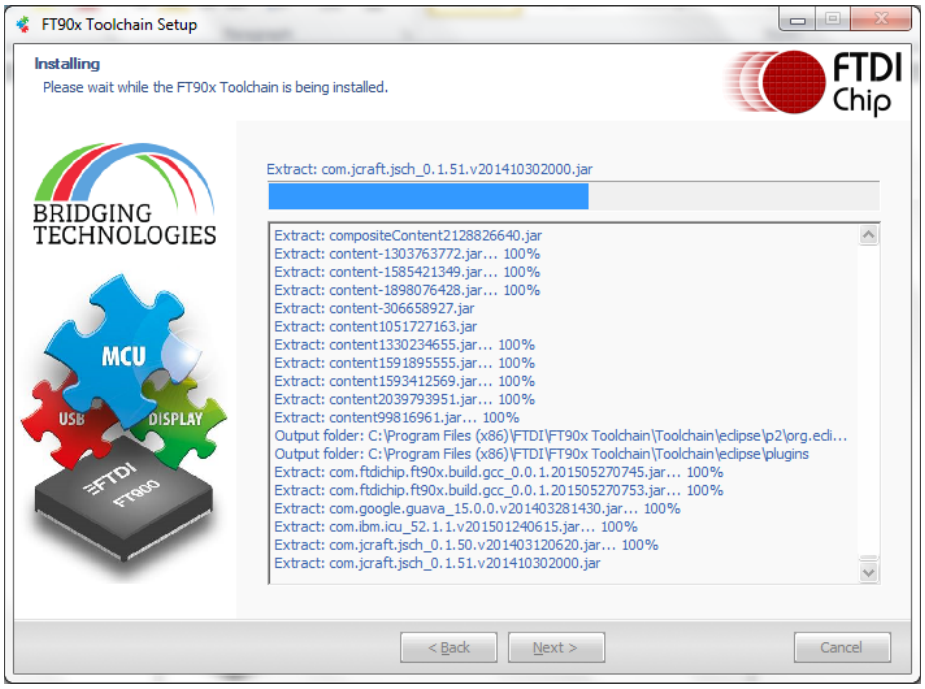
Complete the Python 2.7.10 Installer

Special Windows thanks to:
Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

Click the Finish button to exit the Installer.

Figure 14 Python Installation Completion Dialog box

The FT90x Toolchain installation is continued.

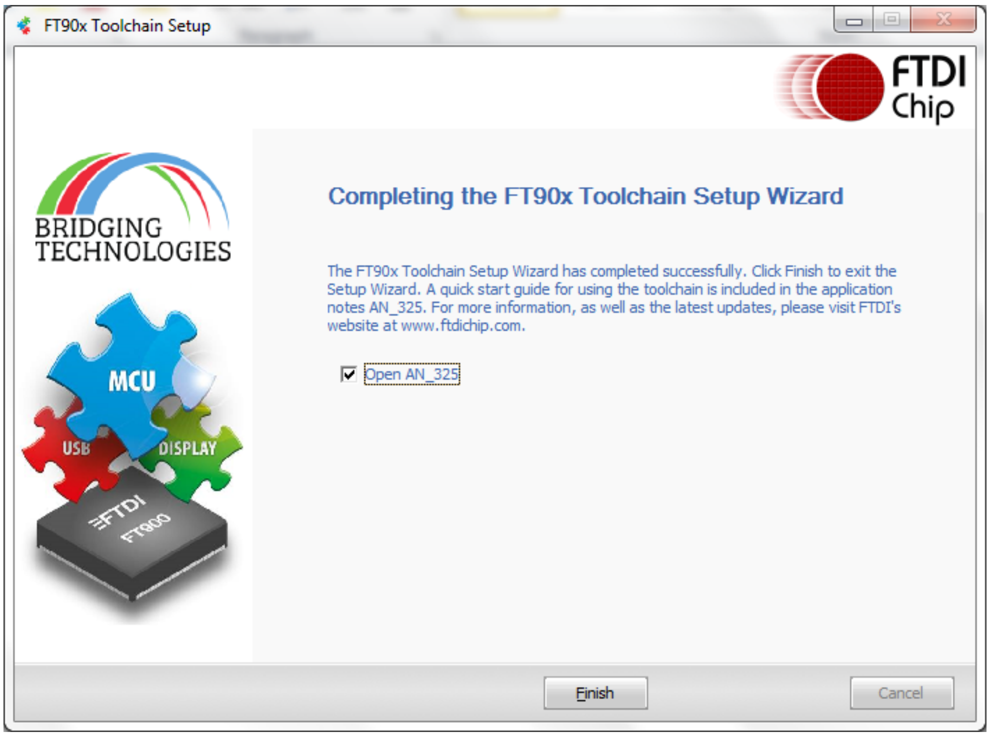


Extract: com.jcraft.jsch_0.1.51.v201410302000.jar

```
Extract: compositeContent2128826640.jar
Extract: content-1303763772.jar... 100%
Extract: content-1585421349.jar... 100%
Extract: content-1898076428.jar... 100%
Extract: content-306658927.jar
Extract: content1051727163.jar
Extract: content1330234655.jar... 100%
Extract: content1591895555.jar... 100%
Extract: content1593412569.jar... 100%
Extract: content2039793951.jar... 100%
Extract: content99816961.jar... 100%
Output folder: C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\eclipse\p2\org.ecli...
Output folder: C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\eclipse\plugins
Extract: com.ftdichip.ft90x.build.gcc_0.0.1.201505270745.jar... 100%
Extract: com.ftdichip.ft90x.build.gcc_0.0.1.201505270753.jar... 100%
Extract: com.google.guava_15.0.0.v201403281430.jar... 100%
Extract: com.ibm.icu_52.1.1.v201501240615.jar... 100%
Extract: com.jcraft.jsch_0.1.50.v201403120620.jar... 100%
Extract: com.jcraft.jsch_0.1.51.v201410302000.jar
```

Figure 15 **FT90x Toolchain - Installation Progress Window**

Select the **Open AN_325** checkbox to start immediately after closing the Setup Wizard. Else leave it unchecked. Click **Finish** to complete the FT90x Toolchain Setup.



Completing the FT90x Toolchain Setup Wizard

The FT90x Toolchain Setup Wizard has completed successfully. Click Finish to exit the Setup Wizard. A quick start guide for using the toolchain is included in the application notes AN_325. For more information, as well as the latest updates, please visit FTDI's website at www.ftdichip.com.

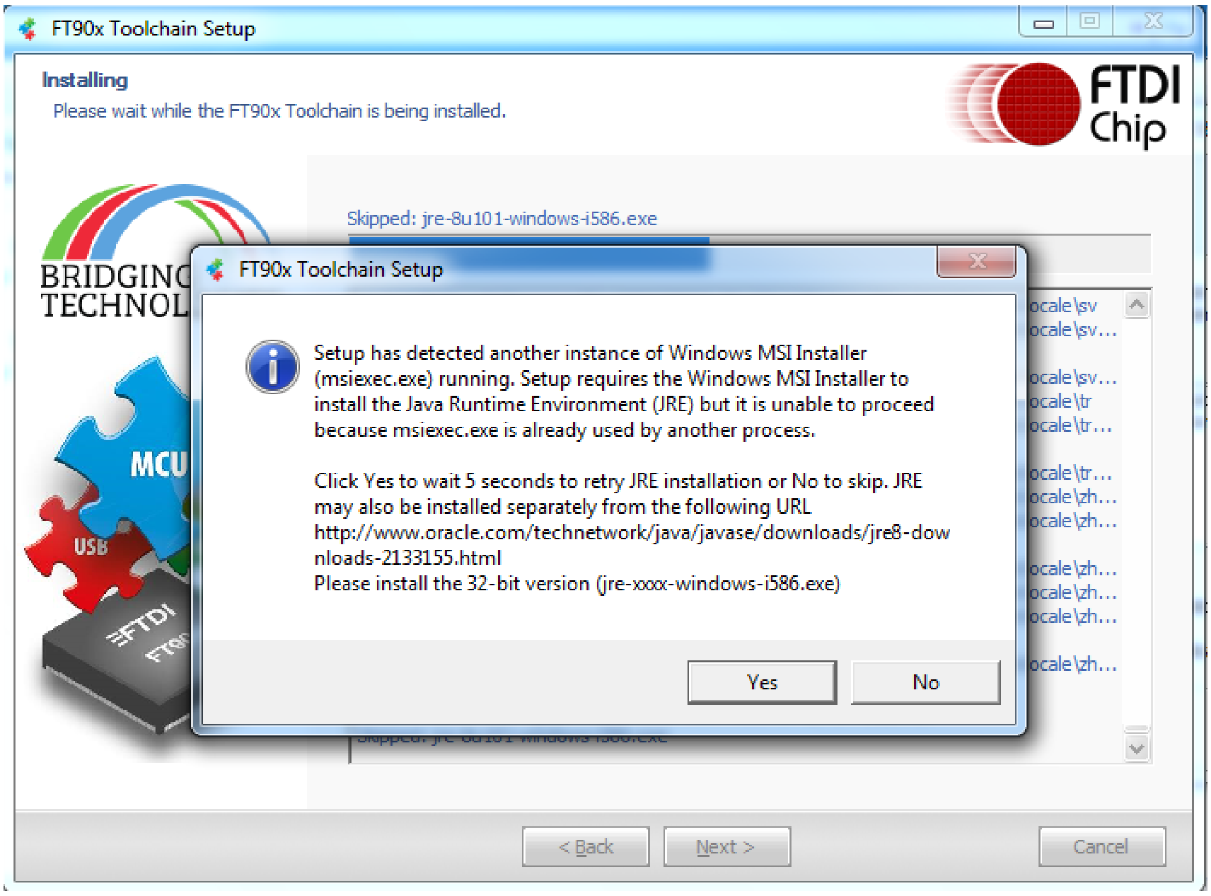
☑ Open AN_325

Figure 16 FT90x Toolchain Setup Completion Dialog box

After the installation, the toolchain can be found in the installation directory. The default location is "C:\Program Files\FTDI\FT90X Toolchain" for 32-bit Windows and "C:\Program Files (x86)\FTDI\FT90X Toolchain" for 64-bit Windows. This directory also contains the external utilities needed. The FT90x drivers, sample applications and documents (if selected for installation) can be found in "My Documents\FTDI\FT90x".

2.1.1 Installing Java Runtime Environment Manually

The toolchain requires the Windows MSI Installer (msiexec.exe) while installing Java Runtime Environment (JRE). The MSI installer can only process one installation at a time. Under some conditions, msiexec.exe may have already been started by another Windows process during automatic Windows Update for example. If the installer detects another instance of msiexec.exe running in the background, the user will be prompted to either wait for the background MSI Installer to complete and retry after 5 seconds or to skip the JRE installation entirely. This is shown in Figure 17


FT90x Toolchain Setup

Installing
Please wait while the FT90x Toolchain is being installed.

Skipped: jre-8u101-windows-i586.exe

FT90x Toolchain Setup

Setup has detected another instance of Windows MSI Installer (msiexec.exe) running. Setup requires the Windows MSI Installer to install the Java Runtime Environment (JRE) but it is unable to proceed because msiexec.exe is already used by another process.

Click Yes to wait 5 seconds to retry JRE installation or No to skip. JRE may also be installed separately from the following URL
http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html
Please install the 32-bit version (jre-xxxx-windows-i586.exe)

Figure 17: MSI Installer is busy

If the user skips JRE installation, JRE can be installed manually from the Oracle Website (http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html) or by re-running the FT90x Toolchain Installer later. Please ensure to install the 32-bit version of JRE (jre-xxx-windows-i586.exe) as the Eclipse installed as part of the FT90x Toolchain install is 32-bit.

2.2 Verifying the installation

1. Open a Command Prompt window by typing "cmd" in "Windows Start button → Search box".
2. Type "ft32-elf-gcc --version" in the command prompt. It should give the following message:

```
ft32-elf-gcc (GCC) 7.0.0 20160708 (experimental)
Copyright (C) 2016 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

If this message appears, then the toolchain has been successfully setup.

3 Quick Start Guide: From creating to getting your application to run on the FT90x MCUs

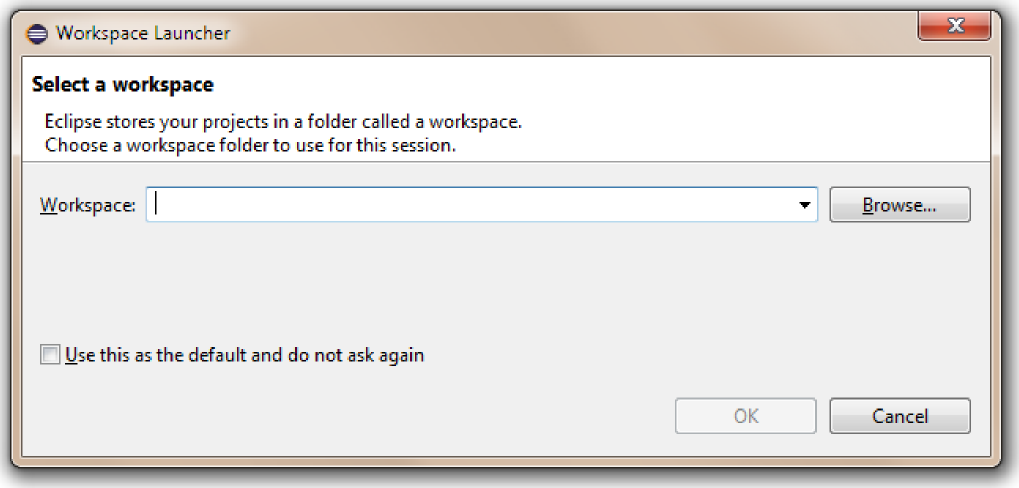
This chapter guides you through the steps to create a new application, compile and program it into the chip. To debug your application, please refer to chapter 4 - "Setting up Eclipse for Debugging". For more information about the tools, as well as the advanced features, refer to chapter 5 - "Advanced Topics".

3.1 Creating a new project

Double click on the icon "Eclipse for FT90x" to launch the Eclipse IDE.

Figure 18 Eclipse for FT90x Icon

When you run Eclipse for the first time, it will ask you for the location of the workspace. Eclipse will create some files within this directory to manage the projects. Specify a folder of your choice and click OK.



Workspace Launcher

Select a workspace

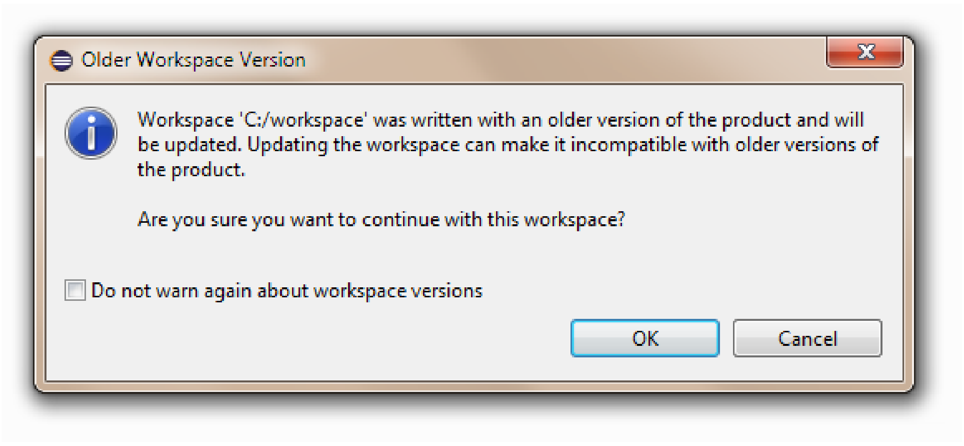
Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace:

☐ Use this as the default and do not ask again

Figure 19 Eclipse Workspace Selection

Note: *The following message will be displayed if an existing workspace, which was created by an older version of Eclipse, is specified. As there may be some configurational changes in files related to workspace in the newer version of Eclipse which may cause issues, it is recommended to create a new workspace and imported the existing projects there.*


Older Workspace Version

Workspace 'C:/workspace' was written with an older version of the product and will be updated. Updating the workspace can make it incompatible with older versions of the product.

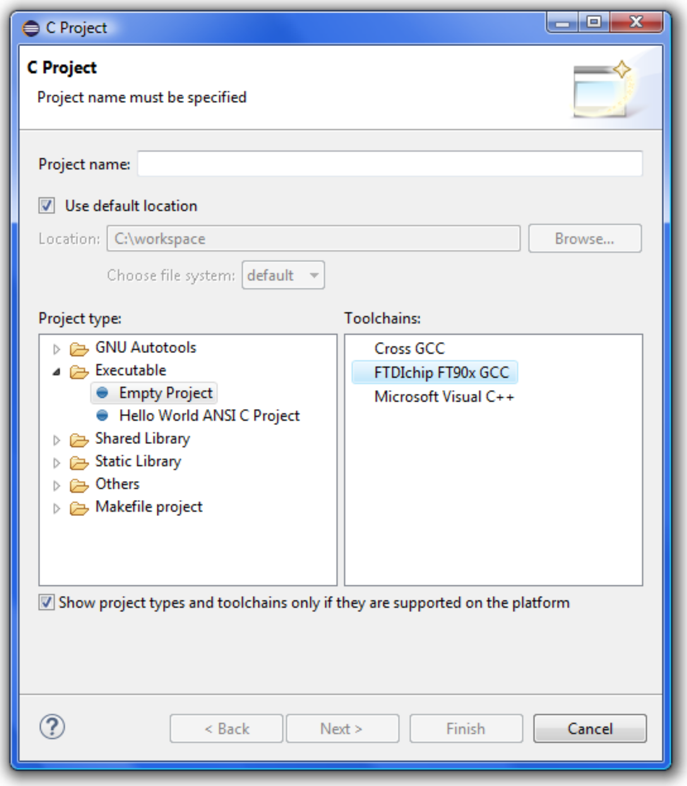
Are you sure you want to continue with this workspace?

☐ Do not warn again about workspace versions

Figure 20 Eclipse Workspace Update

To create a new C project in Eclipse, on the menu bar click "File → New → C Project". The C Project wizard will open.

Give a name to the project, for example "Hello World". By default, the new project will be created inside the workspace you have chosen. If you want to change it, uncheck the box "Use default location" and specify another location. Choose **"Empty Project"** for the project type and **"FTDIchip FT90x GCC"** for the toolchain. This ensures all the relevant FT90x include files are part of the project. Click Next.


C Project

C Project
Project name must be specified

Project name:
☑ Use default location
Location: C:\workspace
Choose file system: default

Project type:
- GNU Autotools
- Executable
  - Empty Project
  - Hello World ANSI C Project
- Shared Library
- Static Library
- Others
- Makefile project

Toolchains:
- Cross GCC
- FTDIchip FT90x GCC
- Microsoft Visual C++

☑ Show project types and toolchains only if they are supported on the platform

Figure 21 C Project Wizard

In the next window, select both Debug and Release for the configuration and click **Next**.

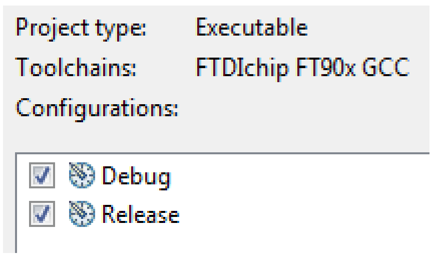

Project type: Executable
Toolchains: FTDIchip FT90x GCC
Configurations:
☑ Debug
☑ Release

Figure 22 Project Wizard - Build Configurations Selection

The last window is for the toolchain prefix and location. By default, the values will be prefilled as follows.

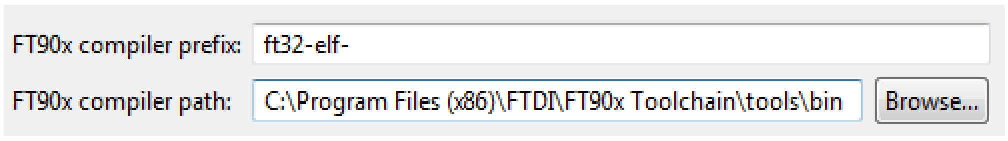

FT90x compiler prefix: ft32-elf-
FT90x compiler path: C:\Program Files (x86)\FTDI\FT90x Toolchain\tools\bin

Figure 23 C Project Wizard - Toolchain Details

Click Finish to complete the New Project Wizard. A new FT90x project will be created in Eclipse.

3.2 Building the project

After the wizard completes, some folders and an empty source file (main.c) will be created.

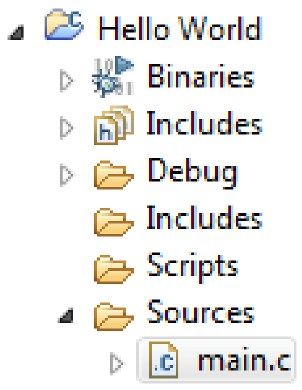

```
Hello World
    Binaries
    Includes
    Debug
    Includes
    Scripts
    Sources
        main.c
```

Figure 24 New empty project structure

Let's give main.c some content, for example:

```c
#include <stdio.h>
#include <ft900.h>

int main(void)
{
        /* Enable UART0 */
        sys_enable(sys_device_uart0);
        /* Make GPIO48 function as UART0_TXD and GPIO49 as UART0_RXD */
        gpio_function(48, pad_uart0_txd);
        gpio_function(49, pad_uart0_rxd);
        /* Open UART0 */
        uart_open(UART0,
                  1,
                  UART_DIVIDER_115200_BAUD,
                  uart_data_bits_8,
                  uart_parity_none,
                  uart_stop_bits_1);
        /* Print out a welcome message */
        uart_puts(UART0, "Hello World!\r\n");
        /* Now keep looping */
        while (1);

        return 0;

}
```

Save the file. Now the project can be built by clicking on the menu **Project → Build Project**, but note that there are a few options like right-click on the project → Build Project and the icon.

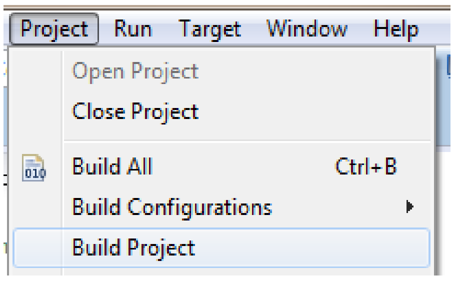
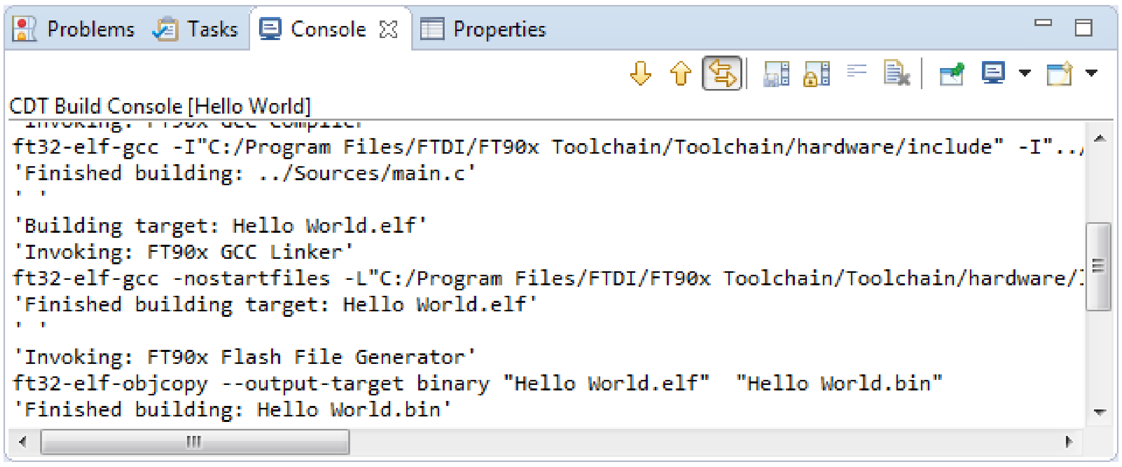


Figure 25 Building the Project

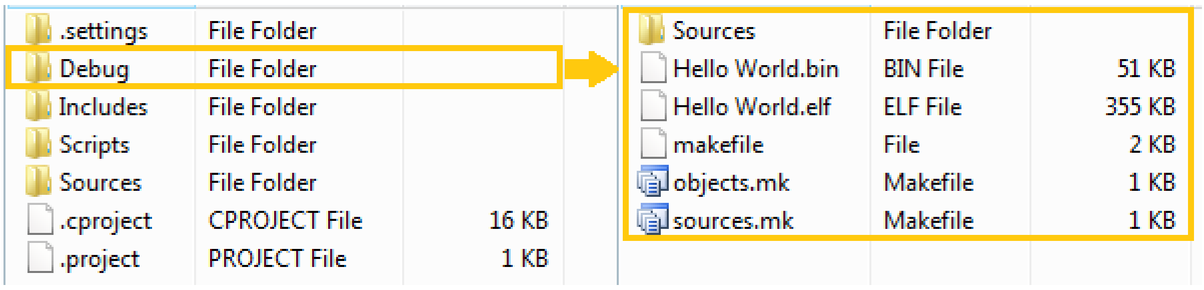
The console window at the bottom of the IDE shows the build status. If the build completes successfully, two files will be created - "Hello World.elf" and "Hello World.bin". The file to be programmed into the chip is "Hello World.bin". The .elf file is used for the debugger, as detailed in the next chapter.



CDT Build Console [Hello World]

```
ft32-elf-gcc -I"C:/Program Files/FTDI/FT90x Toolchain/Toolchain/hardware/include" -I"../
'Finished building: ../Sources/main.c'
' '
'Building target: Hello World.elf'
'Invoking: FT90x GCC Linker'
ft32-elf-gcc -nostartfiles -L"C:/Program Files/FTDI/FT90x Toolchain/Toolchain/hardware/l
'Finished building target: Hello World.elf'
' '
'Invoking: FT90x Flash File Generator'
ft32-elf-objcopy --output-target binary "Hello World.elf"  "Hello World.bin"
'Finished building: Hello World.bin'
```

Figure 26 Build Status



.settings	File Folder	
Debug	File Folder	
Includes	File Folder	
Scripts	File Folder	
Sources	File Folder	
.cproject	CPROJECT File	16 KB
.project	PROJECT File	1 KB

Sources	File Folder	
Hello World.bin	BIN File	51 KB
Hello World.elf	ELF File	355 KB
makefile	File	2 KB
objects.mk	Makefile	1 KB
sources.mk	Makefile	1 KB

Figure 27 List of Files after building

3.3 Programming the binary file into the chip

The FT90x Programmer is provided together with the toolchain. There are a couple of options available.

3.3.1 GUI Version

To run it, double click on the icon "FT900 Programming Utility" created on your desktop, if selected during install, otherwise it can be found in:

C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\programmer\dist

Figure 28 FT900 Programming Utility Icon

You can also open the programming utility from Eclipse by selecting it in the FTDI Utilities menu or the toolbar icon as highlighted in Figure 29:

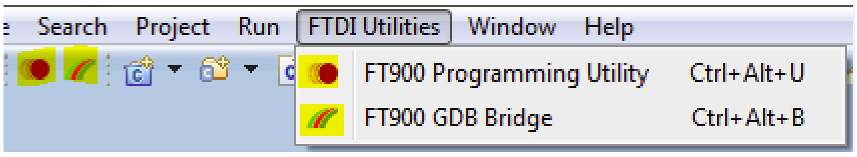

Search
Project
Run
FTDI Utilities
Window
Help
FT900 Programming Utility Ctrl+Alt+U
FT900 GDB Bridge Ctrl+Alt+B

Figure 29 FTDI Utilities Menu

After the splash message the following screen will appear.

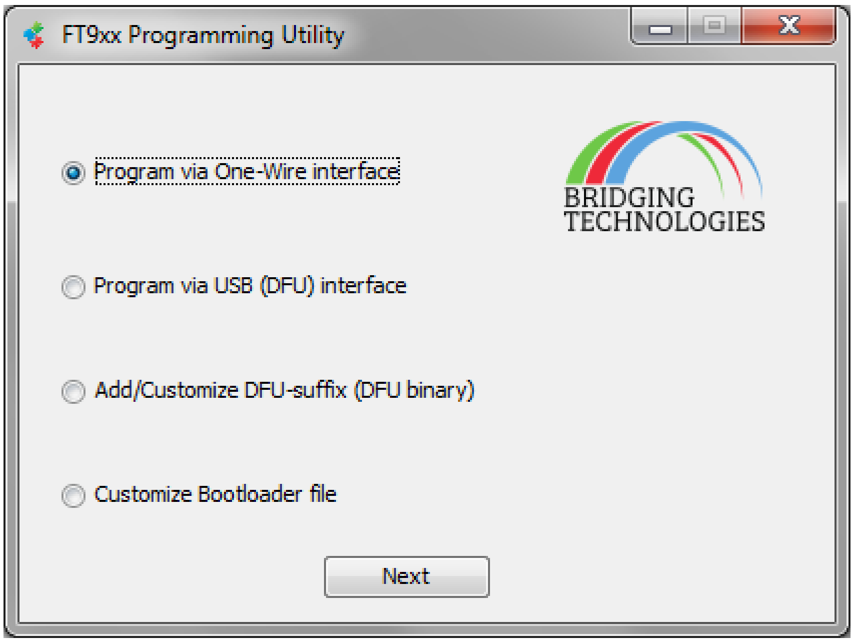
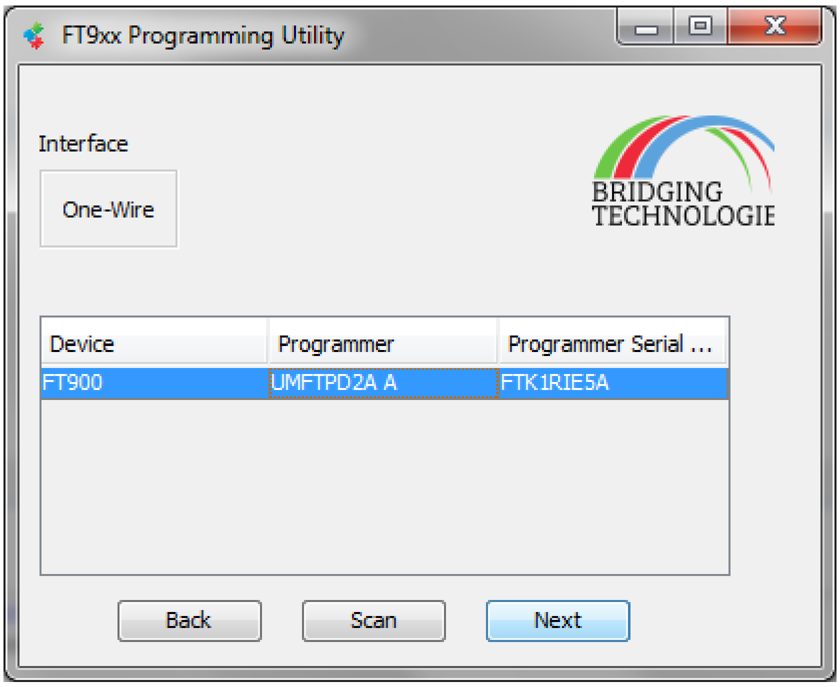

FT9xx Programming Utility
Program via One-Wire interface
BRIDGING TECHNOLOGIES
Program via USB (DFU) interface
Add/Customize DFU-suffix (DFU binary)
Customize Bootloader file
Next

Figure 30: FT90x Programmer - Work with One-Wire

Select the "Work with One-Wire" option and click Next. The next screen shows a list of supported devices that you might wish to program.

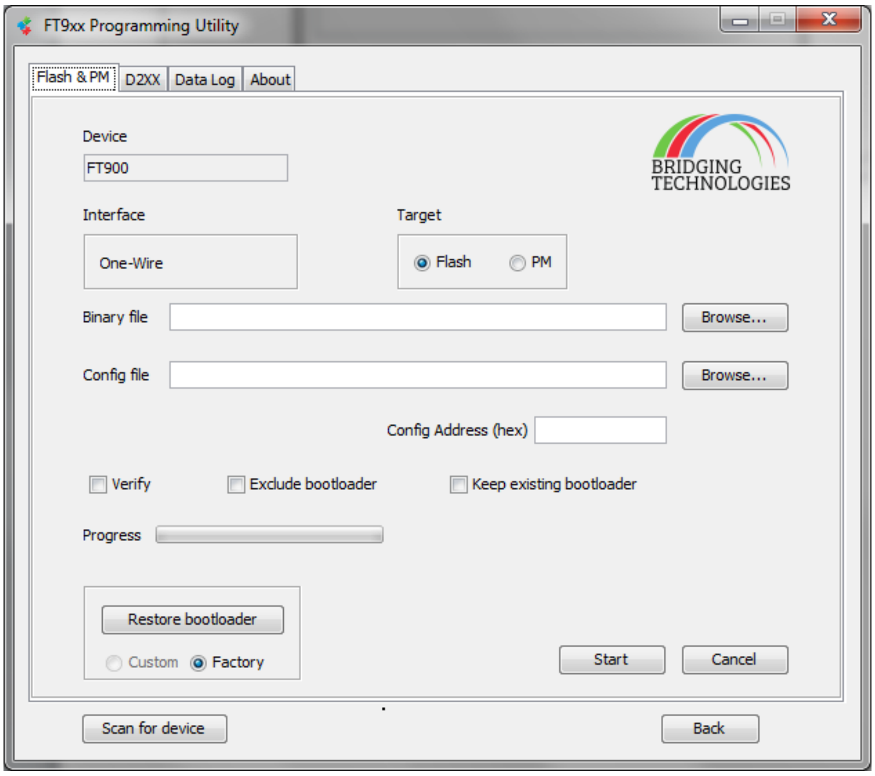
When a valid FT900 and Programmer module are detected, the information will be displayed in the list. Select the device you wish to program and click Next to launch the programmer window.



Device	Programmer	Programmer Serial ...
FT900	UMFTPD2A A	FTK1RIE5A

Figure 31: FT90x Programmer - Device Selection

In the programmer window, leave everything as default. Specify the location of the binary file and click Start. If the Verify check box is selected, an icon will show up next to the status bar to indicate whether the flash memory has been properly programmer.



Device: FT900
Interface: One-Wire
Target: [✓ Flash] PM
Binary file:
Config file:
Config Address (hex):
☐ Verify ☐ Exclude bootloader ☐ Keep existing bootloader
Custom [✓ Factory]

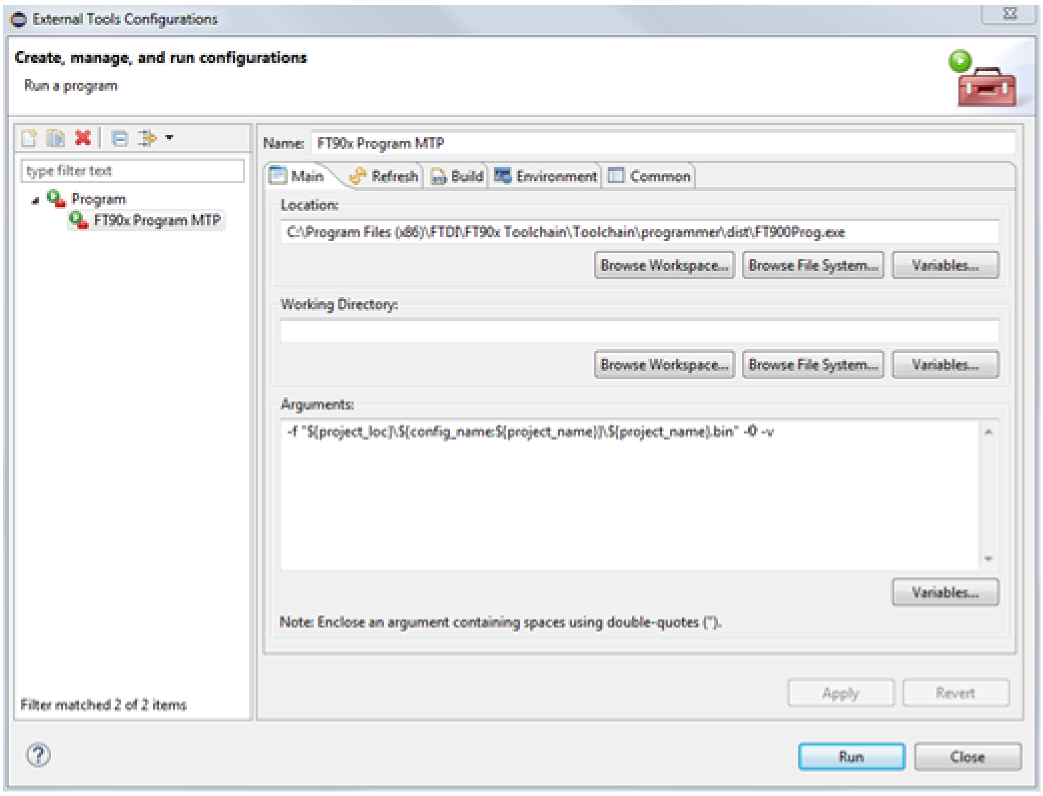
Figure 32 **FT90x Programmer – Flash and PM Screen**

More information on the utility can be found in the 'About' tab, then click on Help.

3.3.2 Command Line Version

FT900Prog.exe is available to run at a command prompt. Enter FT900Prog.exe to see the options available. See section 5.1.2 for more details.

This can also be run within Eclipse as an External Tool. See Figure 33 for settings found in Run → External Tools → External Tools Configurations.



Name: FT90x Program MTP

Location:
C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\programmer\dist\FT900Prog.exe

Working Directory:

Arguments:
```
-f "${project_loc}\${config_name:${project_name}}\${project_name}.bin" -0 -v
```

Note: Enclose an argument containing spaces using double-quotes (").

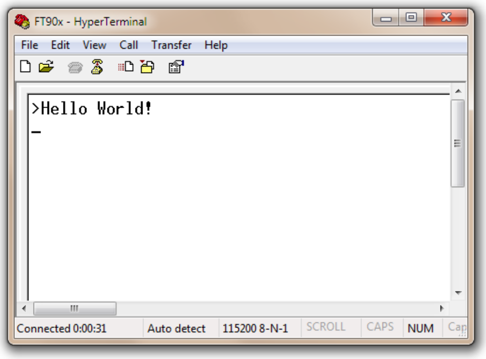
Figure 33 FT90x Programmer in Eclipse

3.4 "Hello World" in action, and more...

The "Hello World" example above will send a message to a serial terminal via the FT90x UART0 port. Open a terminal on your computer, for example Tera Term or HyperTerminal. Apply the following settings:

• Baud Rate:	115200
• Parity Bit:	None
• Data Bit:	8
• Stop Bit:	1
• Flow Control:	None

Now when you reset the MCU, the message will be printed to the terminal.


FT90x - HyperTerminal
File Edit View Call Transfer Help
Connected 0:00:31 Auto detect 115200 8-N-1 SCROLL CAPS NUM Cap

```
>Hello World!
_
```

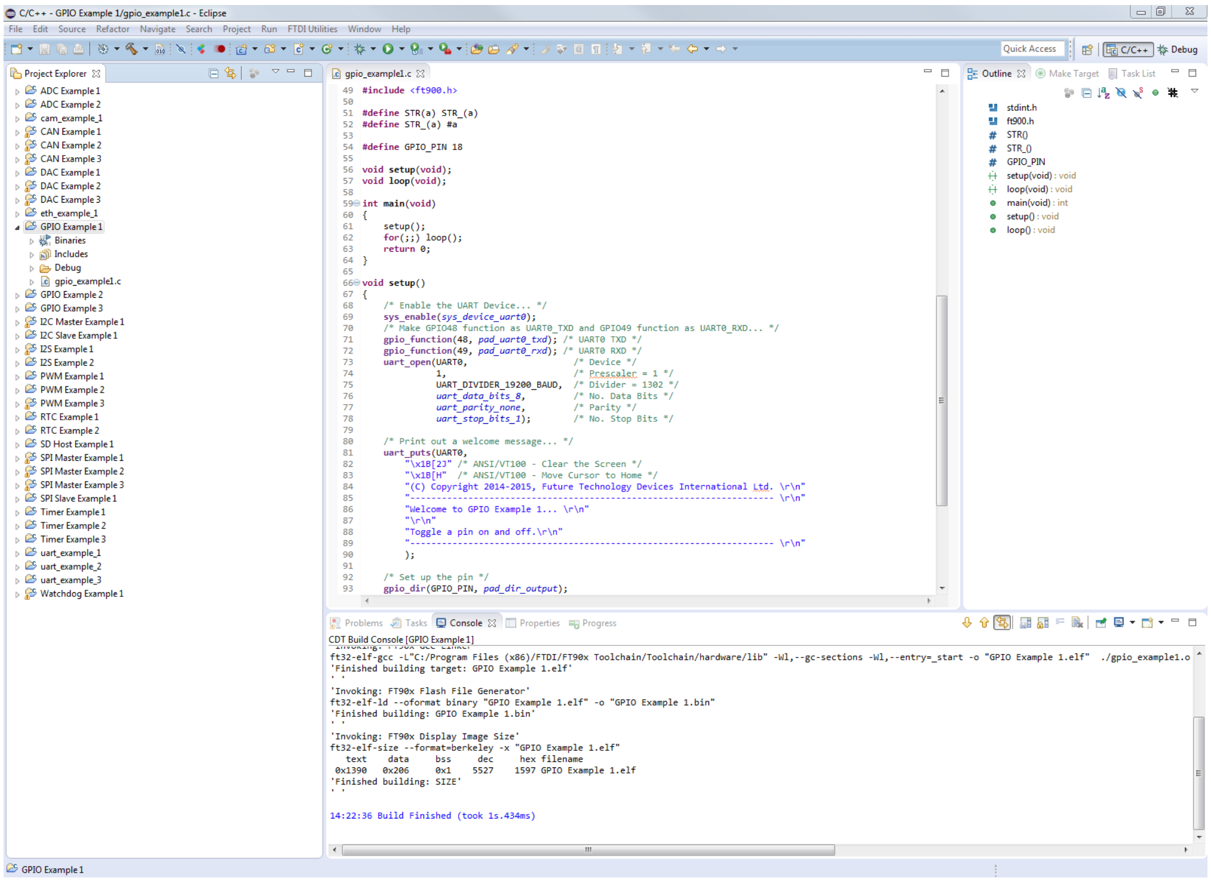
Figure 34 Hello World

Congratulations! You have just completed your first project for FT90x. The FT90x toolchain comes with plenty of examples, which demonstrate a variety of features. If you have selected to install them in the Toolchain Installation Wizard, by default they can be found in:

"My Documents\FTDI\FT90x\version\Examples"

The Eclipse project has already been setup for these examples, as suggested by the presence of two files - ".cproject" and ".project". Instead of creating a new project, you can simply import these projects into the workspace. To do this:

1. On the Menu bar, choose "File → Import"
2. In the Import window, choose "General → Existing Projects into Workspace" and click "Next".
3. In the next window, set the root directory to "My Documents\FTDI\FT90x\version\Examples". The projects will be detected by Eclipse.
4. Select which projects you wish to import and click Finish to complete the importing process. This is an example of how Eclipse would look like with the sample applications. Please refer to AN_360 (which is also included with the toolchain installation) for more details about these applications.



```c
#include <ft900.h>

#define STR(a) STR_(a)
#define STR_(a) #a

#define GPIO_PIN 18

void setup(void);
void loop(void);

int main(void)
{
    setup();
    for(;;) loop();
    return 0;
}

void setup()
{
    /* Enable the UART Device... */
    sys_enable(sys_device_uart0);
    /* Make GPIO48 function as UART0_TXD and GPIO49 function as UART0_RXD... */
    gpio_function(48, pad_uart0_txd); /* UART0 TXD */
    gpio_function(49, pad_uart0_rxd); /* UART0 RXD */
    uart_open(UART0,                    /* Device */
              1,                        /* Prescaler = 1 */
              UART_DIVIDER_19200_BAUD,  /* Divider = 1302 */
              uart_data_bits_8,         /* No. Data Bits */
              uart_parity_none,         /* Parity */
              uart_stop_bits_1);        /* No. Stop Bits */

    /* Print out a welcome message... */
    uart_puts(UART0,
        "\x1B[2J" /* ANSI/VT100 - Clear the Screen */
        "\x1B[H"  /* ANSI/VT100 - Move Cursor to Home */
        "(C) Copyright 2014-2015, Future Technology Devices International Ltd. \r\n"
        "-------------------------------------------------------------------- \r\n"
        "Welcome to GPIO Example 1... \r\n"
        "\r\n"
        "Toggle a pin on and off.\r\n"
        "-------------------------------------------------------------------- \r\n"
        );

    /* Set up the pin */
    gpio_dir(GPIO_PIN, pad_dir_output);
```

```
ft32-elf-gcc -L"C:/Program Files (x86)/FTDI/FT90x Toolchain/Toolchain/hardware/lib" -Wl,--gc-sections -Wl,--entry=_start -o "GPIO Example 1.elf"  ./gpio_example1.o
'Finished building target: GPIO Example 1.elf'
' '
'Invoking: FT90x Flash File Generator'
ft32-elf-ld --oformat binary "GPIO Example 1.elf" -o "GPIO Example 1.bin"
'Finished building: GPIO Example 1.bin'
' '
'Invoking: FT90x Display Image Size'
ft32-elf-size --format=berkeley -x "GPIO Example 1.elf"
   text    data     bss     dec     hex filename
 0x1390   0x206     0x1    5527    1597 GPIO Example 1.elf
'Finished building: SIZE'
' '

14:22:36 Build Finished (took 1s.434ms)
```

Figure 35 FT90x Examples

4 Setting up Eclipse for Debugging

Eclipse comes with an intuitive GUI for debugging applications. To enable this feature, eclipse requires additional information about our debugger. The steps are presented below.

4.1 Build the application using the Debug configuration

The application should be built using the Debug configuration so that the debug information is available. It is the default build configuration but can be verified in the Project menu.

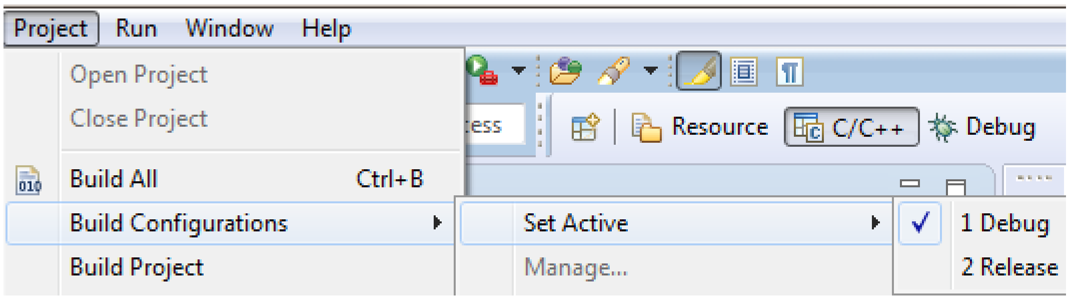


Figure 36 Build Configuration

4.2 Create a new debug configuration

A debug configuration is used by Eclipse to launch the debug GUI and only needs to be created once for the FT90x Debugger. To create it:

1. On the Menu bar, select **Run → Debug Configurations…**
2. In the Debug Configurations window, double click on "C/C++ Remote Application"
3. Press the 'New' button to create a new debug configuration
4. In the next window, a Debugging Launcher will need to be specified. Click on "Select others…" at the bottom of the window.

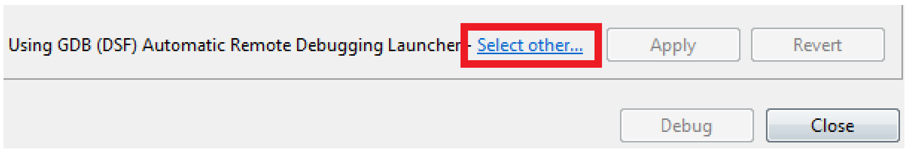
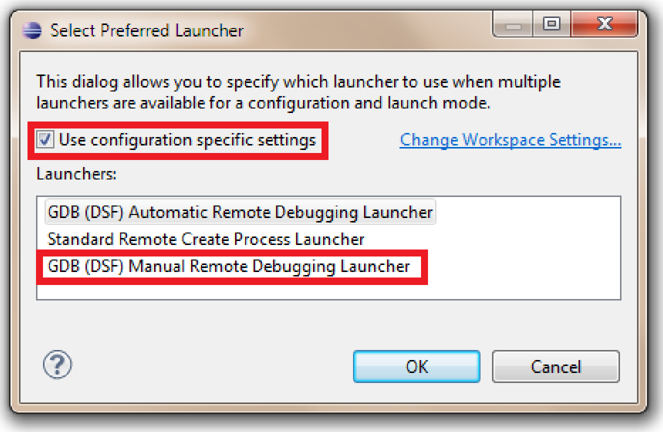

Using GDB (DSF) Automatic Remote Debugging Launcher - Select other...

Figure 37 Choosing a Debugging Launcher (1)

5. In the "Select Preferred Launcher" window, check "Use configuration specific settings". Then choose "GDB (DSF) **Manual** Remote Debugging Launcher". Click OK.


Select Preferred Launcher

This dialog allows you to specify which launcher to use when multiple launchers are available for a configuration and launch mode.

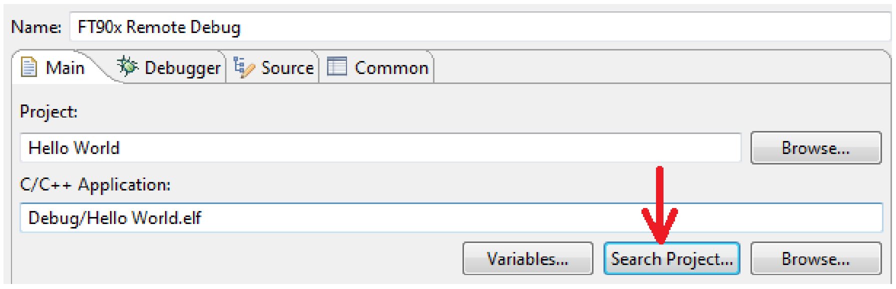
☑ Use configuration specific settings   Change Workspace Settings...

Launchers:

GDB (DSF) Automatic Remote Debugging Launcher
Standard Remote Create Process Launcher
GDB (DSF) Manual Remote Debugging Launcher

Figure 38 Choosing a Debugging Launcher (2)

6. Now provide the details for the configuration. Specify the **name** of the Debug configuration, for example "FT90x Remote Debug". Use this configuration to debug FT90x projects from now on.
7. Under Main tab, specify the **project** and the **.elf file** for **Application**. The "Browse…" button next to the project field will list all active projects. The .elf file can be found easily after the project has been selected, by clicking on **"Search Project..."** button.

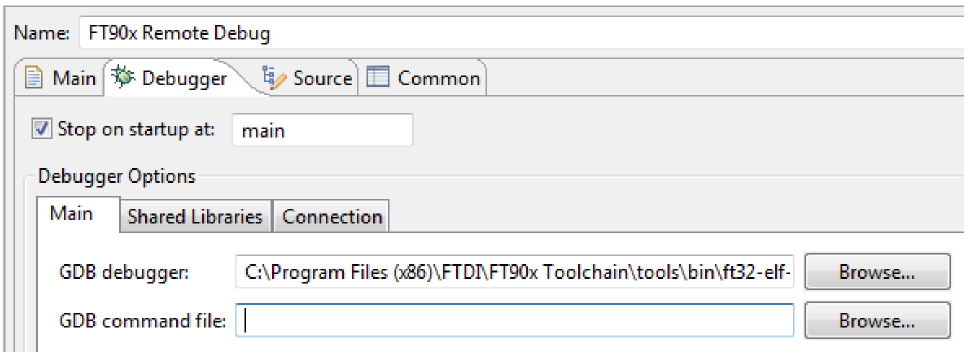

Name: FT90x Remote Debug

Project: Hello World

C/C++ Application: Debug/Hello World.elf

Figure 39 Eclipse Debugging Application Settings

8. Under the Debugger tab, the user needs to provide some Debugger Options. Specify the path to ft32-elf-gdb.exe (or simply "ft32-elf-gdb.exe") in the **Main sub-tab** and make sure the **"GDB command file"** field is **empty**. ft32-elf-gdb.exe can be located in the toolchain installation folder, under "tools\bin".


Name: FT90x Remote Debug

☑ Stop on startup at: main

Debugger Options

GDB debugger: C:\Program Files (x86)\FTDI\FT90x Toolchain\tools\bin\ft32-elf-

GDB command file:

Figure 40 Eclipse GDB Settings

9. Under the Connection sub-tab, choose the connection type to be **TCP**. Enter **"localhost"** for the **host name** and **9998** for the **port number**.

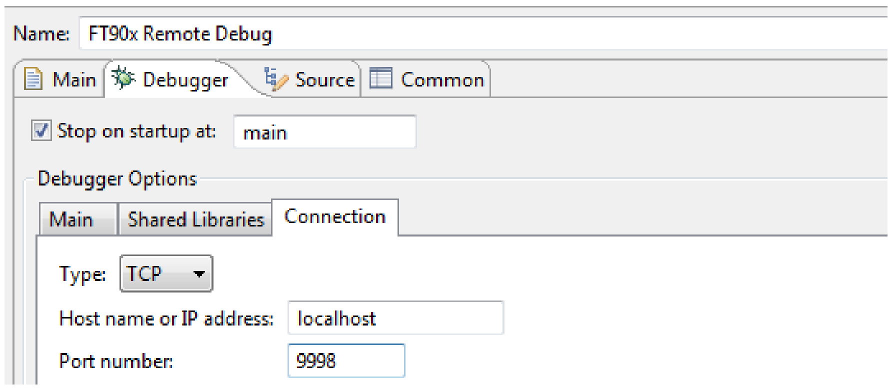


Figure 41 Eclipse TCP Port Settings

10. Click "Apply" and close the window.

A debug configuration for FT90x has now been created. To use the same configuration for other projects, simply open it and select the right project and application, as presented in step 6 above.

4.3 Running the GDB Bridge

The GDB Bridge is needed for ft32-elf-gdb to talk to the MCU. To run it, simply double click on the desktop icon "GDB Bridge".

Figure 42 GDB Bridge Icon

You can also launch the GDB Bridge from Eclipse by selecting it in the FTDI Utilities menu or the toolbar icon as highlighted in **Figure 43** FTDI Utilities Menu

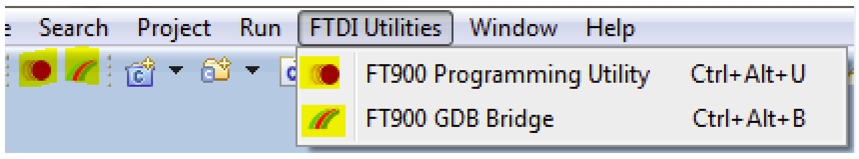
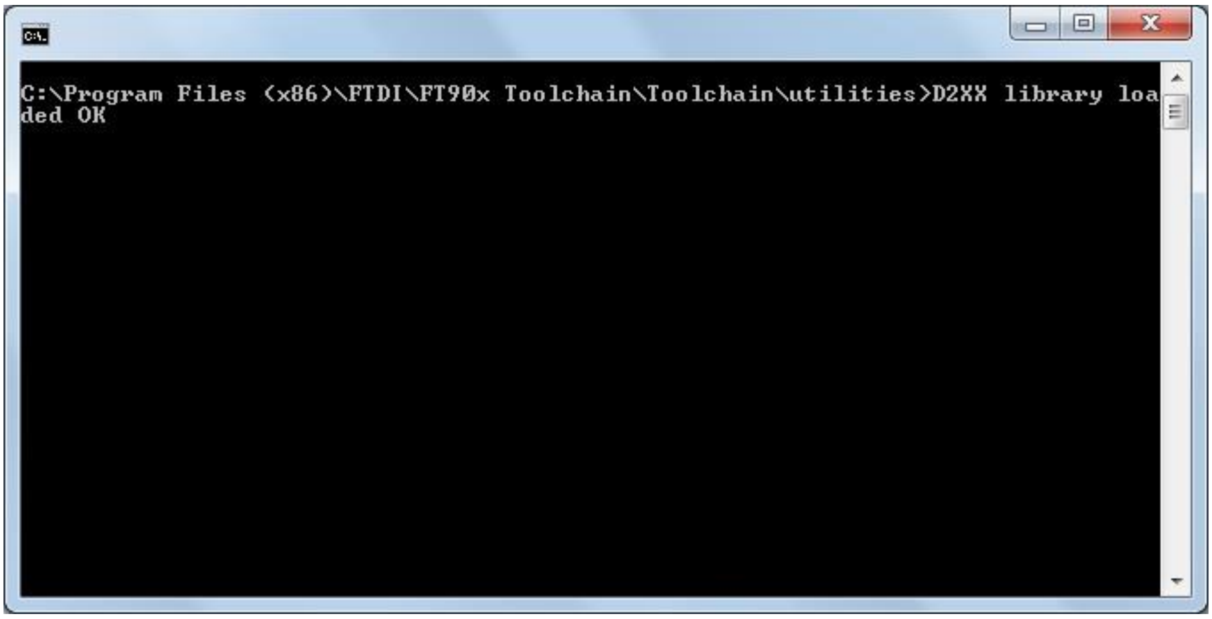

Search
Project
Run
FTDI Utilities
Window
Help
FT900 Programming Utility Ctrl+Alt+U
FT900 GDB Bridge Ctrl+Alt+B

Figure 43 FTDI Utilities Menu

The following window should appear:



```
C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\utilities>D2XX library loa
ded OK
```

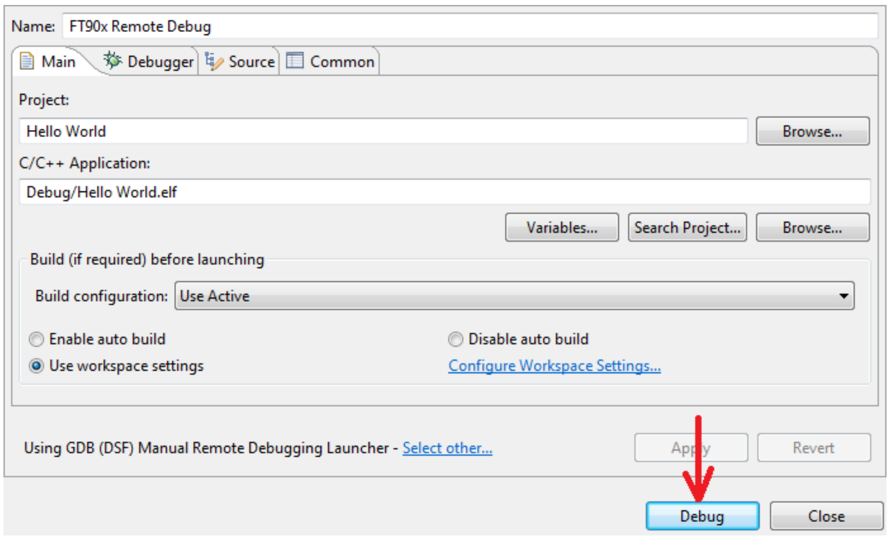
Figure 44 GDB Bridge in action

Now the tools are ready to debug the application in Eclipse.

***Note**: the user must close this debug GDB script when debugging is finished, otherwise it may not be possible to program the device for example.*

4.4 Debugging the application in Eclipse

1. Open the debug configuration that was created (FT90x Remote Debug) and click Debug. Note that this will appear in the Debug button on the toolbar after running once.



Name: FT90x Remote Debug

Project: Hello World

C/C++ Application: Debug/Hello World.elf

Build (if required) before launching

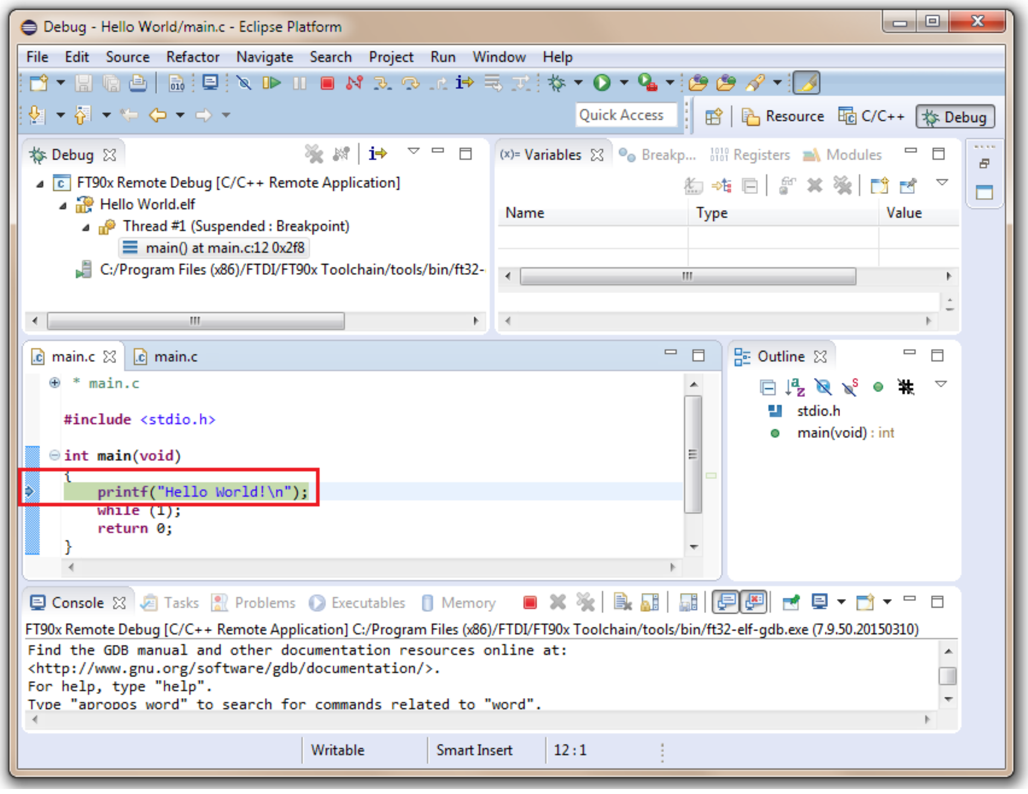
Build configuration: Use Active

( ) Enable auto build ( ) Disable auto build
(●) Use workspace settings Configure Workspace Settings...

Using GDB (DSF) Manual Remote Debugging Launcher - Select other...

Figure 45 Eclipse Run Remote Debugging

2. The Debug perspective will be opened. The execution will stop at the first line in main(), as shown below. Various debug commands (step into/over, resume, halt, stop, etc.) can now be accessed from the toolbar via buttons. Function variables, setting breakpoints and viewing physical memory in the memory tab, along with some other debug features are also available now.



```
 * main.c

#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");
    while (1);
    return 0;
}
```

```
FT90x Remote Debug [C/C++ Remote Application] C:/Program Files (x86)/FTDI/FT90x Toolchain/tools/bin/ft32-elf-gdb.exe (7.9.50.20150310)
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
```

Figure 46 Eclipse Debug Environment

Note: *If there is an error message about missing source file as below, locate the source file that contains the main() function using the "Locate file…" button.*

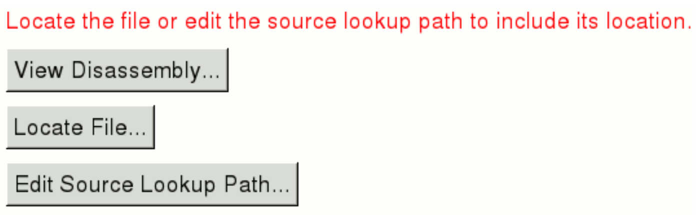
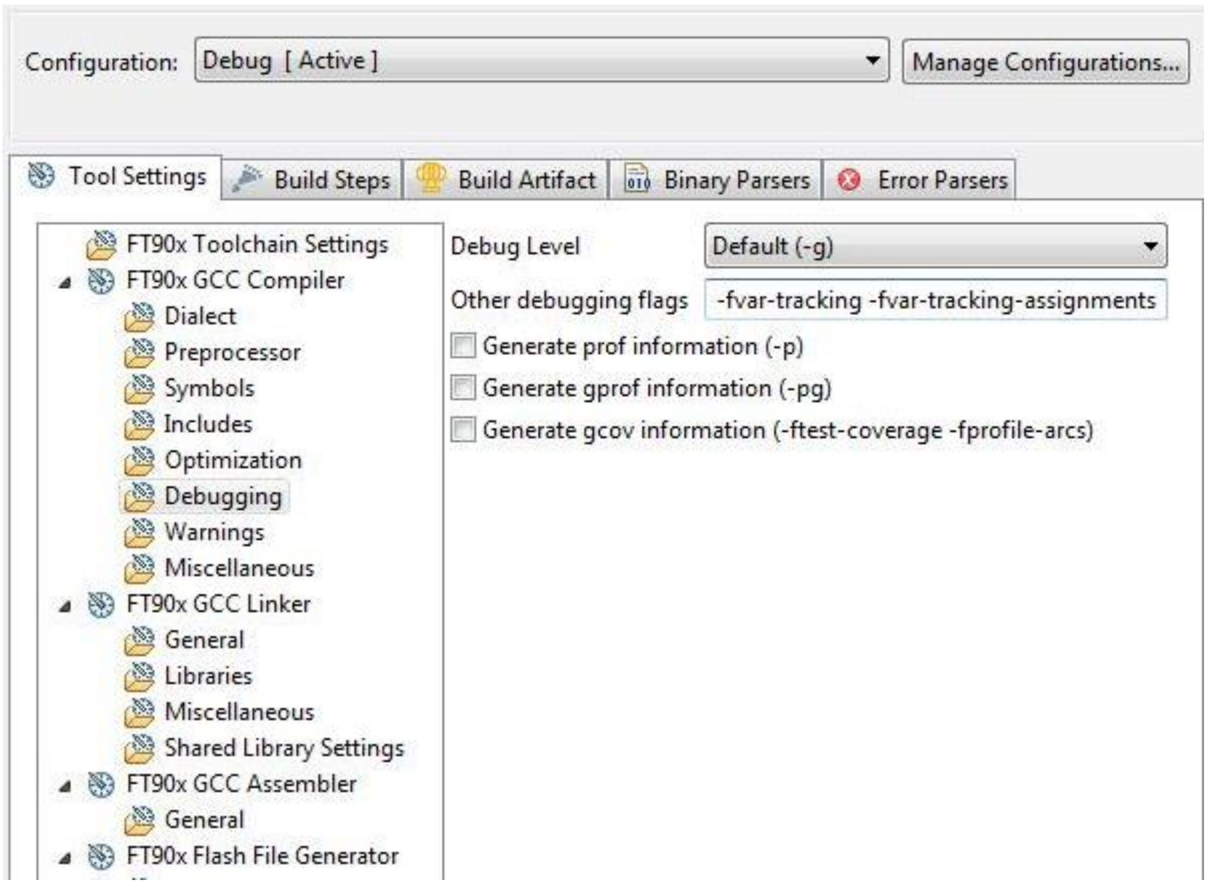

Locate the file or edit the source lookup path to include its location.

Figure 47 Eclipse Missing Source File

4.4.1 Watch variables in Eclipse debug perspective

If the watch variables fail to update or display incorrect values, check that the following flags exist for the debug build (they are present by default in all projects created with FTDI eclipse plugin)

-fvar-tracking -fvar tracking-assignments



Configuration: Debug [Active]

Debug Level: Default (-g)

Other debugging flags: -fvar-tracking -fvar-tracking-assignments

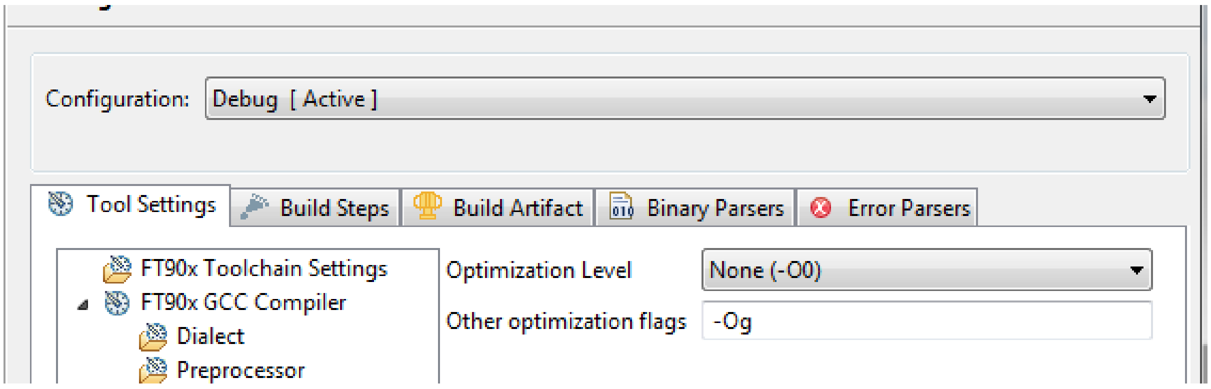
- [ ] Generate prof information (-p)
- [ ] Generate gprof information (-pg)
- [ ] Generate gcov information (-ftest-coverage -fprofile-arcs)

Figure 48 Debug flags

4.4.2 Og compiler option when debugging

When compiling a project with no optimization (or –O0) some useful debugging information may not be generated at all, leading to possible unexpected results while debugging. To avoid this, it is recommended to turn on –Og option when no other optimization flags are used. The FTDI eclipse plugin does this automatically.

Note that if multiple optimization options are used, only the last option will be effective.



Configuration: Debug [Active]

Optimization Level: None (-O0)

Other optimization flags: -Og

Figure 49 Og compiler optimization option

More information can be found in the GCC documentation - https://gcc.gnu.org/onlinedocs/gcc/Debugging-Options.html and https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html

4.5 Eclipse features supported by ft32-elf-gdb

At the moment, not all features of the Eclipse debug perspective are supported by ft32-elf-gdb. The current list of supported features is:

- Breakpoint creation.
- Single stepping/stepping in/stepping out of functions
- Watch variables
- Assembly instruction stepping

5 FTDI Projects

Besides the empty project used as the example in Chapter 3, there are several project types specific to FTDI. They can be found under "Others" in the project type selection window. Currently, there are two project types:

- D2XX Project
- Data Log (DLOG) Project

For more details about these project types, refer to "AN_360 FT900 Example Applications", which is included in the toolchain installation.

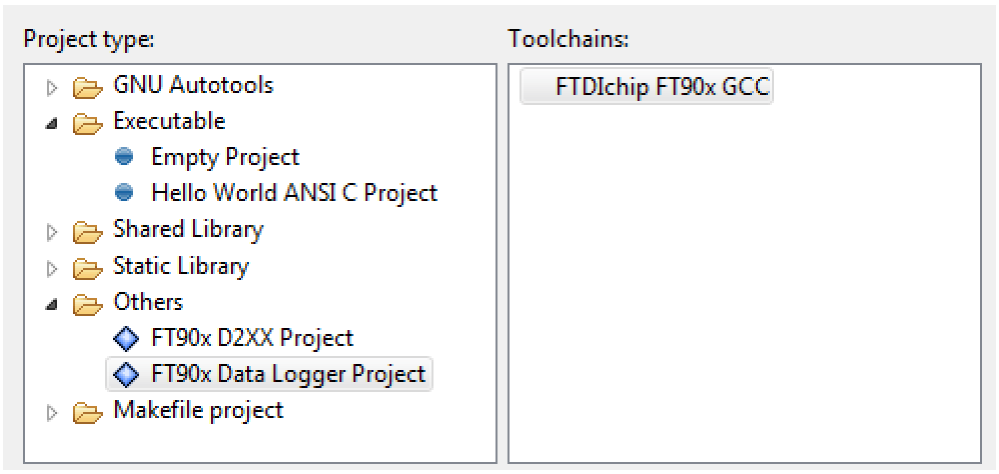
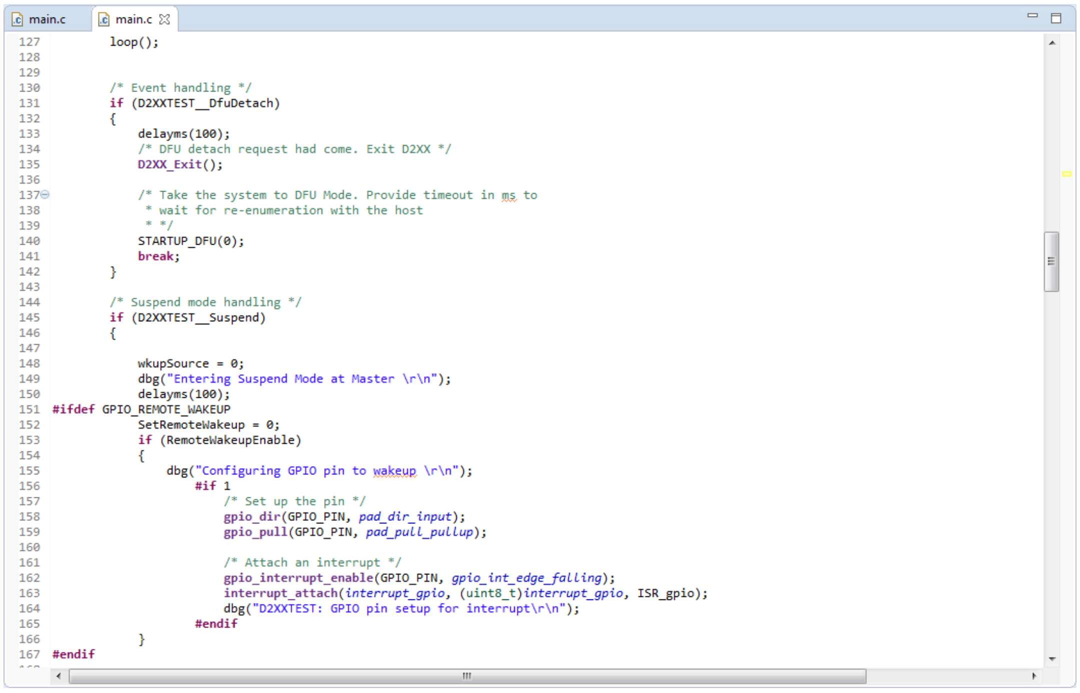


Figure 50 FTDI Project Types

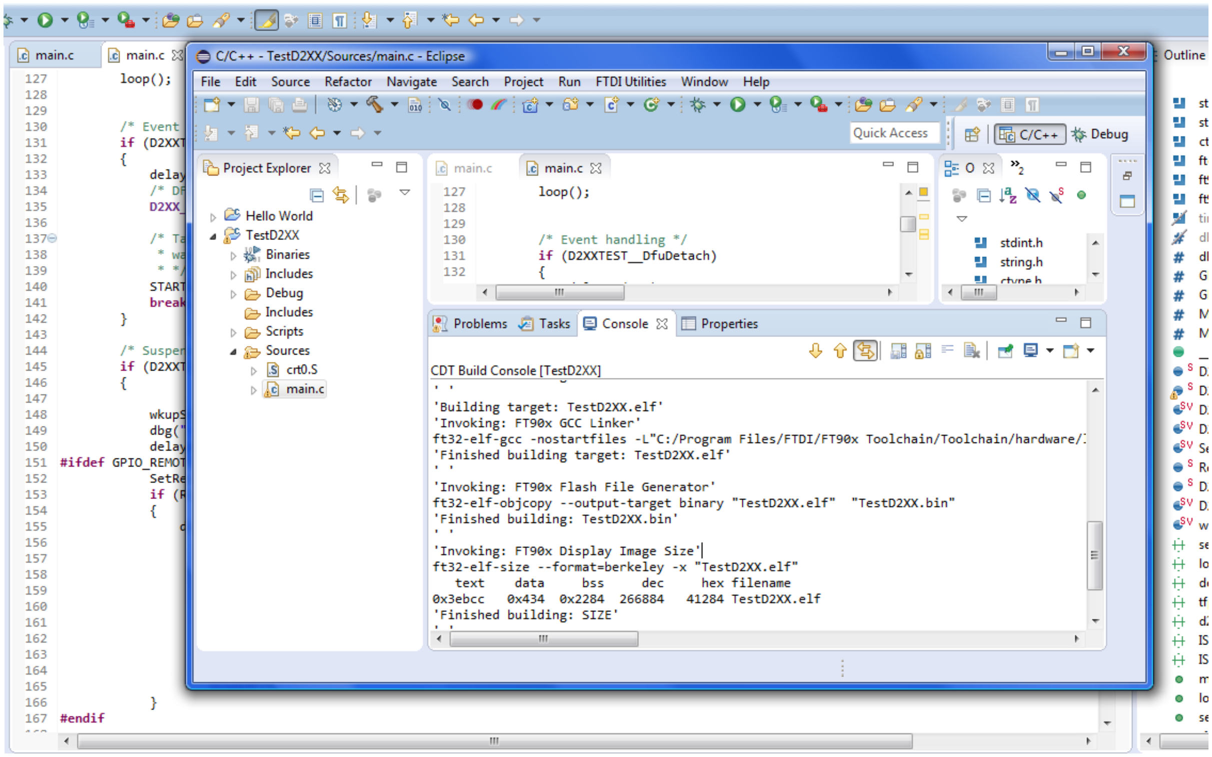
The procedure to create a new project is similar to the empty project. When the wizard completes, a template source file will be generated. Below is the template generated for the D2XX project.



```
        loop();


        /* Event handling */
        if (D2XXTEST__DfuDetach)
        {
            delayms(100);
            /* DFU detach request had come. Exit D2XX */
            D2XX_Exit();

            /* Take the system to DFU Mode. Provide timeout in ms to
             * wait for re-enumeration with the host
             * */
            STARTUP_DFU(0);
            break;
        }

        /* Suspend mode handling */
        if (D2XXTEST__Suspend)
        {

            wkupSource = 0;
            dbg("Entering Suspend Mode at Master \r\n");
            delayms(100);
#ifdef GPIO_REMOTE_WAKEUP
            SetRemoteWakeup = 0;
            if (RemoteWakeupEnable)
            {
                dbg("Configuring GPIO pin to wakeup \r\n");
                    #if 1
                        /* Set up the pin */
                        gpio_dir(GPIO_PIN, pad_dir_input);
                        gpio_pull(GPIO_PIN, pad_pull_pullup);

                        /* Attach an interrupt */
                        gpio_interrupt_enable(GPIO_PIN, gpio_int_edge_falling);
                        interrupt_attach(interrupt_gpio, (uint8_t)interrupt_gpio, ISR_gpio);
                        dbg("D2XXTEST: GPIO pin setup for interrupt\r\n");
                    #endif
            }
#endif
```

Figure 51 D2XX Project Template

The template can be compiled as it is but additional code is needed to customize it according to the user’s need.



```
CDT Build Console [TestD2XX]
' '
'Building target: TestD2XX.elf'
'Invoking: FT90x GCC Linker'
ft32-elf-gcc -nostartfiles -L"C:/Program Files/FTDI/FT90x Toolchain/Toolchain/hardware/
'Finished building target: TestD2XX.elf'
' '
'Invoking: FT90x Flash File Generator'
ft32-elf-objcopy --output-target binary "TestD2XX.elf"  "TestD2XX.bin"
'Finished building: TestD2XX.bin'
' '
'Invoking: FT90x Display Image Size'
ft32-elf-size --format=berkeley -x "TestD2XX.elf"
   text    data     bss     dec     hex filename
0x3ebcc   0x434  0x2284  266884   41284 TestD2XX.elf
'Finished building: SIZE'
```

Figure 52 Compiling D2XX Template Code

6 Advanced Topics

6.1 Running the toolchain from the command prompt

6.1.1 Compiling the sample applications using a Makefile

The FT90x GNU toolchain can be used to compile source code from a command prompt in the same way the official GNU Toolchain is used, often with the help of a Makefile or a batch file.

The sample applications are available in "My Documents\FTDI\FT90x\Examples\ if you have installed them using the installation wizard.

NOTE: makefiles are not included with the toolchain installer.

6.1.2 Programming a binary file into the chip

The programmer can be found in the folder "programmer\dist" in the program installation directory (C:\Program Files (x86)\FTDI\FT90x Toolchain). The command line programmer is FT900Prog.exe. The toolchain is provided with a default bootloader. The bootloader is located at the top 4 KB of the flash memory (address 0x3F000 to 0x3FFFF). At boot, the FT90x resets and executes instruction at 0x00000, jumping into the bootloader. The bootloader then performs the initializations needed and jumps to location 0x8c, which is the start of the user program. The bootloader is also needed to support debugging with the FT90x port of GDB.

1. Run the tool FT900Prog.exe without any arguments, the options and usage will be printed. They will also be printed if the specified options are not valid. The most common usage is programming a binary file through the one-wire interface with the supplied bootloader. To do this, the command is:

   ```
   FT900Prog.exe -f <.bin file with path if needed> -O
   ```
   in which the options are:

   -f: programming the binary file into the **f**lash. The path to the binary file must follow.
   -O: using the **o**ne-wire interface.

   If you want to **v**erify the content of the flash memory after programming, specify "-v" in the command:

   ```
   FT900Prog.exe -f <.bin file with path if needed> -O –v
   ```

2. If the bootloader is not required, option "-x" can be specified, in which case the program will start executing from address zero and the command is:

   ```
   FT900Prog.exe -f <.bin file with path if needed> -O -x
   ```

   The supports for GDB debugging will not be available however.

6.1.3 Debugging the sample applications with ft32-elf-gdb

1. The applications must to be compiled with -g option (i.e. ft32-elf-gcc -g …). An .elf file will be created which includes the debug information, for example GPIO/gpio_example1.elf. Note that this file is not used for programming the chip.

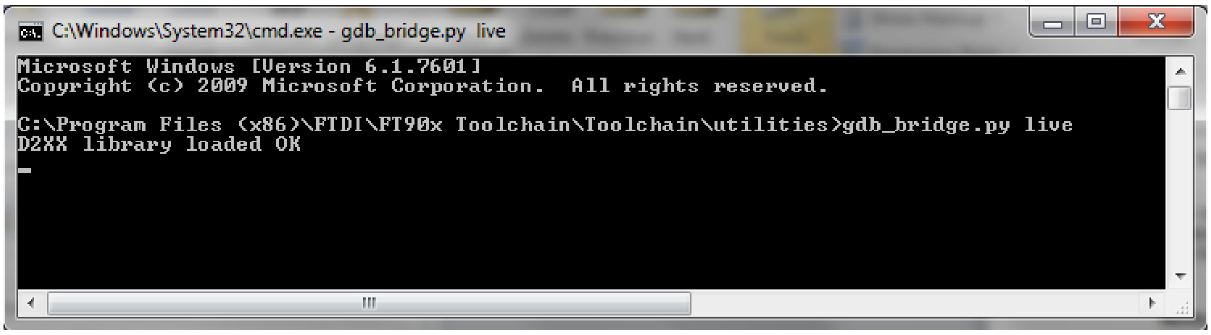
   Note: *If the output file name for the linker is not specified in the Makefile (i.e. option -o is missing), a.out will be created instead of an .elf file. They are the same and these steps can be applied to a.out as well.*

2. Flash the .bin file into the chip. Refer to section 5.1.2 above.

3. Open a command line window, run:
   "python <Installation directory>\Toolchain\utilities\gdb_bridge.py live"

   Note: *An alternative is to double click on the shortcut "GDB Bridge" created after the installation.*

   The correct response should be:



```
C:\Windows\System32\cmd.exe - gdb_bridge.py  live
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Program Files (x86)\FTDI\FT90x Toolchain\Toolchain\utilities>gdb_bridge.py live
D2XX library loaded OK
_
```

Figure 53 FT90x Debugging Status

   Note 1: *If there is an error message about permission being denied, the command line window may need to be opened with administrator rights by right-clicking and selecting 'Run as administrator'.*

   Note 2: *It is also possible to run the GDB Bridge using the shortcut created after the installation.*

   Note 3: *If the path to gdb_bridge.py contains spaces, enclose it with double quotes ("").*

4. Open another command line window, go to the folder that includes the .elf file, run "ft32-elf-gdb <.elf file>", for example "ft32-elf-gdb gpio_example1.elf".

5. After ft32-elf-gdb starts, type in "target remote localhost:9998" to establish a connection to the MCU.

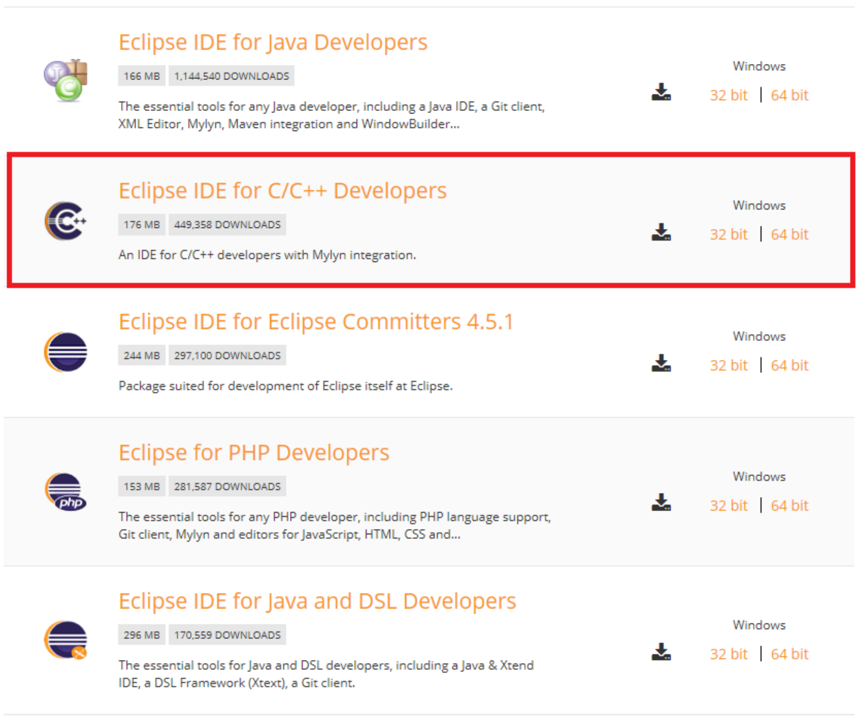
6. Use standard GDB commands to debug the program. Note that the command to start execution should be "continue", not "run".

6.2 Installing Eclipse and the FT90x plugin manually

When running the installer, it is possible to choose not to install Eclipse as part of the installation. This might be useful if the user have already installed Eclipse for other purposes. This section details how to set it up for use with the FT90x.

6.2.1 Eclipse Install

1. Go to Eclipse website, download "Eclipse IDE for C/C++ Developers". At the time of this writing, Eclipse Mars is the latest release and is the recommended version.



Eclipse IDE for Java Developers
166 MB | 1,144,540 DOWNLOADS
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven integration and WindowBuilder...
Windows 32 bit | 64 bit

Eclipse IDE for C/C++ Developers
176 MB | 449,358 DOWNLOADS
An IDE for C/C++ developers with Mylyn integration.
Windows 32 bit | 64 bit

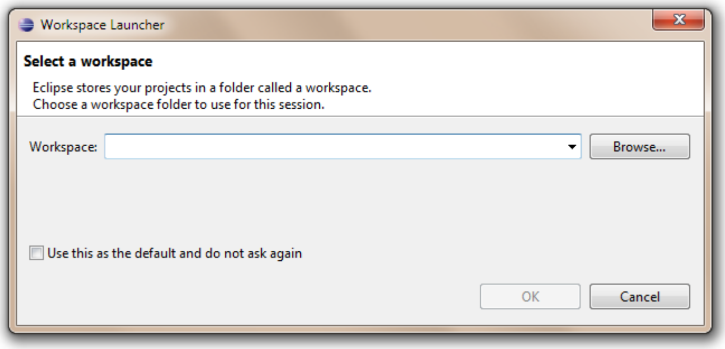
Eclipse IDE for Eclipse Committers 4.5.1
244 MB | 297,100 DOWNLOADS
Package suited for development of Eclipse itself at Eclipse.
Windows 32 bit | 64 bit

Eclipse for PHP Developers
153 MB | 281,587 DOWNLOADS
The essential tools for any PHP developer, including PHP language support, Git client, Mylyn and editors for JavaScript, HTML, CSS and...
Windows 32 bit | 64 bit

Eclipse IDE for Java and DSL Developers
296 MB | 170,559 DOWNLOADS
The essential tools for Java and DSL developers, including a Java & Xtend IDE, a DSL Framework (Xtext), a Git client.
Windows 32 bit | 64 bit

Figure 54 Eclipse Versions

2. When Eclipse is run for the first time, it will ask for the workspace location.



Workspace Launcher

Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace:

☐ Use this as the default and do not ask again

Figure 55 Eclipse Workspace Location

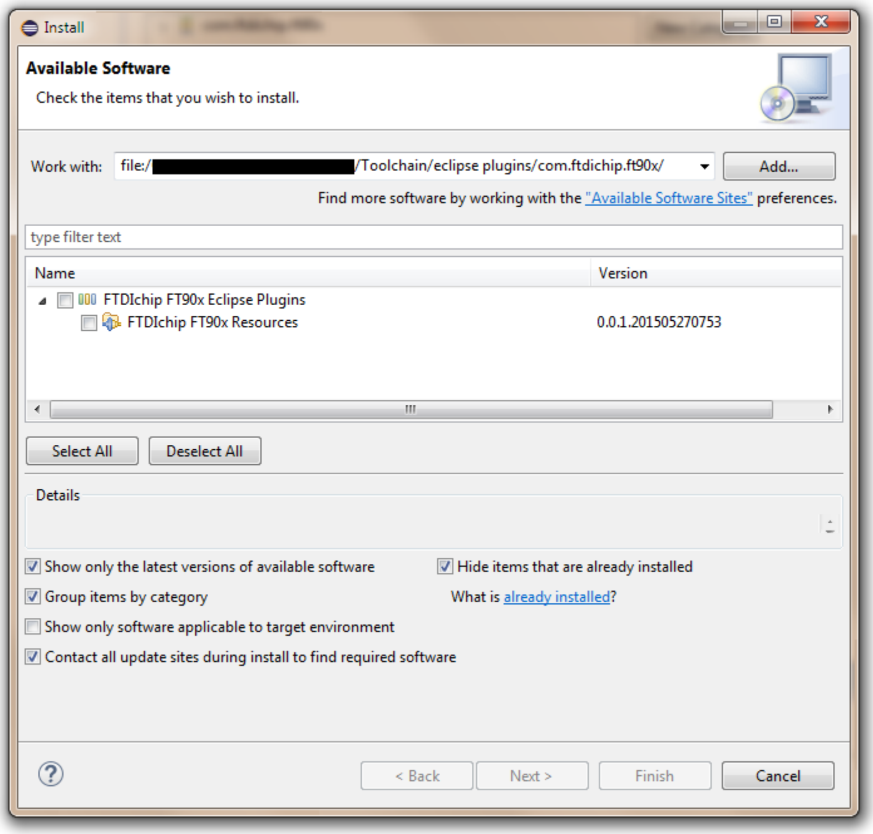
A workspace is a directory on the hard drive where Eclipse stores the projects defined to it. More specifically, a workspace is a logical collection of projects. When you specify this directory name to Eclipse, Eclipse will create some files within this directory to manage the projects. The projects controlled by this workspace may or may not reside in this directory. Specify a directory name and click OK.

Note: *To run Eclipse, it is required to download and install the Java Run Time Environment (JRE) or Java Developer Kit (JDK). Eclipse should display a warning if this is not installed. Oracle provides these tools for free.*

6.2.2 FT900 Eclipse Plugin Installation

To assist with completing the configuration of Eclipse for FT900 coding an extra plug-in is provided as part of the download. To install the plug-in the following steps are required:

1. From the Eclipse toolbar select Help -> Install New Software which will pop up the window as below.



Install

Available Software

Check the items that you wish to install.

Work with: file:/[redacted]/Toolchain/eclipse plugins/com.ftdichip.ft90x/ Add...

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
☐ FTDIchip FT90x Eclipse Plugins	
☐ FTDIchip FT90x Resources	0.0.1.201505270753

Select All Deselect All

Details

☑ Show only the latest versions of available software
☑ Hide items that are already installed
☑ Group items by category
What is already installed?
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software

< Back Next > Finish Cancel

Figure 56 Eclipse Plugin Setup Wizard

2. Select the ADD button, and browse to the LOCAL location of the folder 'com.ftdichip.ft90x' which can be found in "Toolchain\eclipse plugins" in the toolchain installation directory.

3. Press "SELECT ALL" followed by NEXT to install the plugin

4. Close the window when complete.

6.3 Common project settings in Eclipse

6.3.1 Include paths

Eclipse uses its built-in indexer to resolve dependencies between files. In order for the indexer to work correctly, paths that contain the header files in the project need to be added as follows:

1. Right-click on the project and select **Properties**

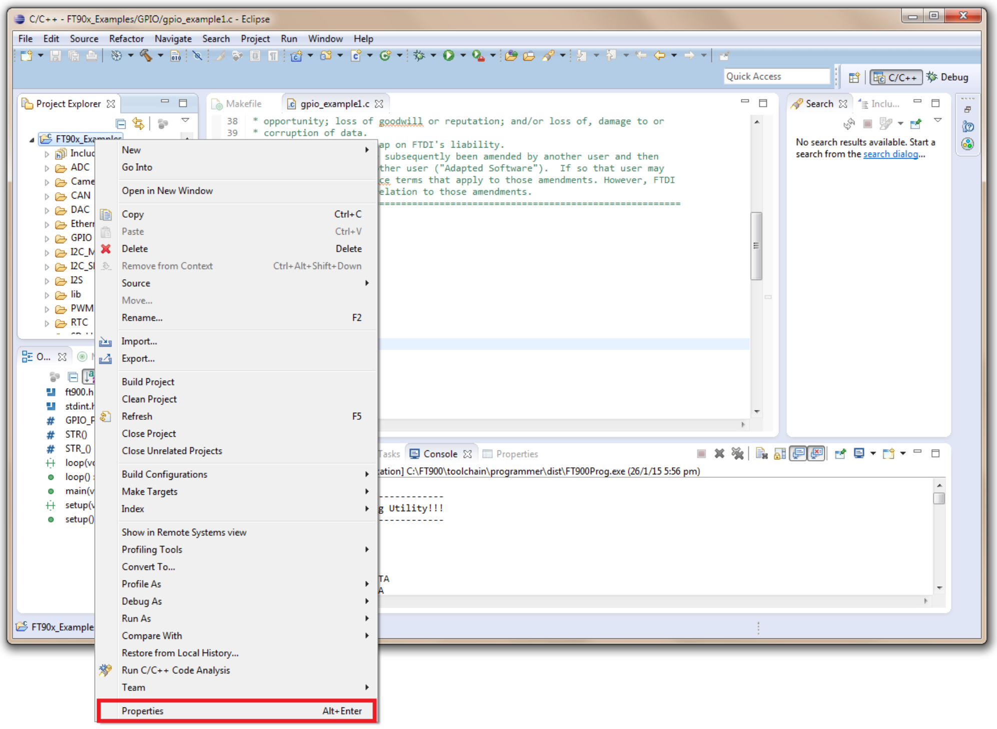


Figure 57 Eclipse Project Properties

2. In the Properties window, **select C/C++ General > Paths and Symbols**

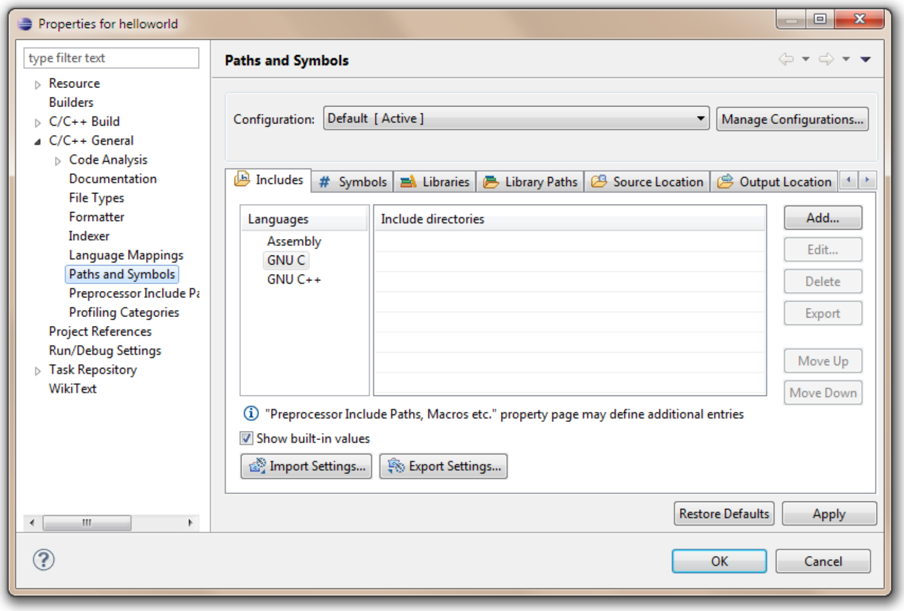


Figure 58 Eclipse Paths and Symbols

3. Under the **Includes** tab, choose **"GNU C"** under **Languages**, then click **"Add…"** on the right side of the window

4. In the **"Add directory path"** window, specify the path to the folder that contains the header files. If the same path is used for some C++ files, check the box **"Add to all languages"**, then click **OK**.

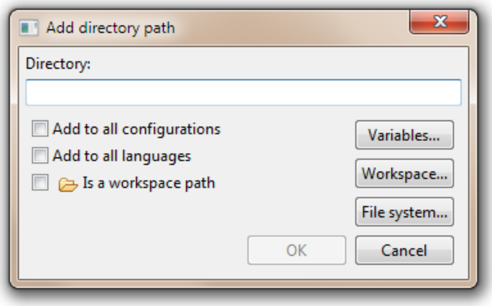


Figure 59 Eclipse Add Directory Path

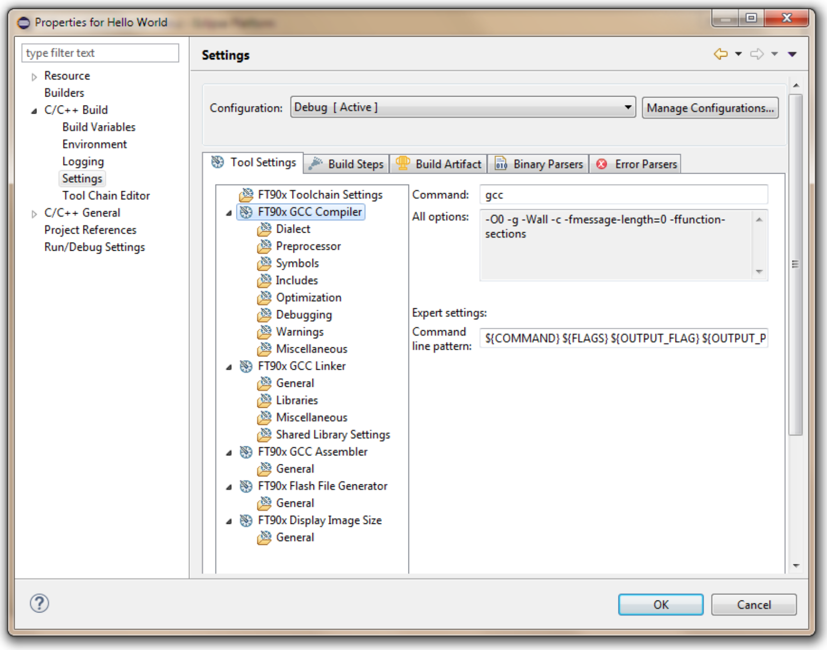
Note: *The paths should be added one at a time. The use of semicolon is not supported.*

6.3.2 Toolchain settings

The FT90x toolchain supports most GNU toolchain options. To specify an option that is not included by default, for example to create a map file, do it as follows:

1. Right click on the project and select Properties

2. In the Properties window, select C/C++ Build > Settings. The toolchain settings can be adjusted in the Settings window.



Configuration: Debug [Active]

Command: gcc

All options: -O0 -g -Wall -c -fmessage-length=0 -ffunction-sections

Expert settings:

Command line pattern: ${COMMAND} ${FLAGS} ${OUTPUT_FLAG} ${OUTPUT_P

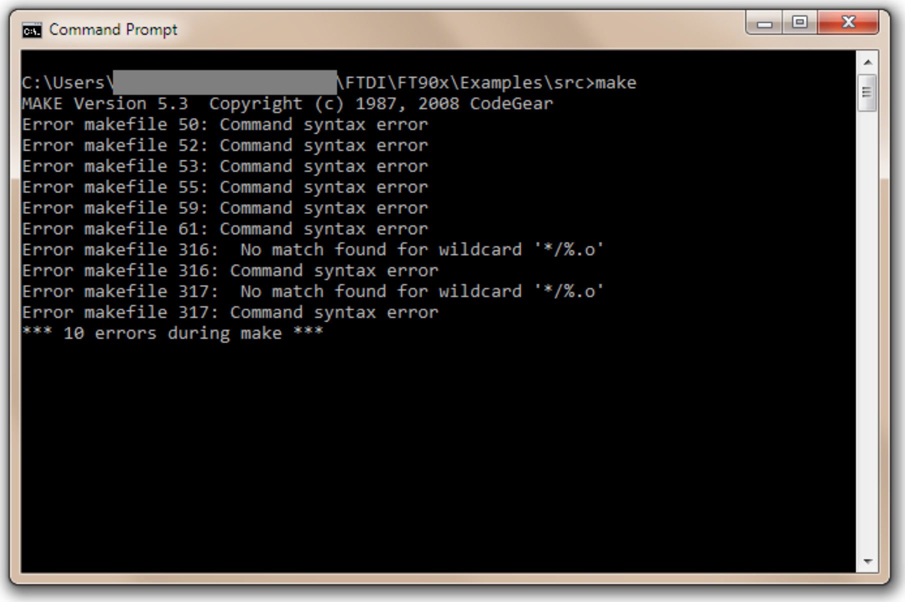
Figure 60 Eclipse Toolchain Settings

7 Troubleshooting

This section documents the problems you may encounter when using the FT90x toolchain.

7.1 Makefile error

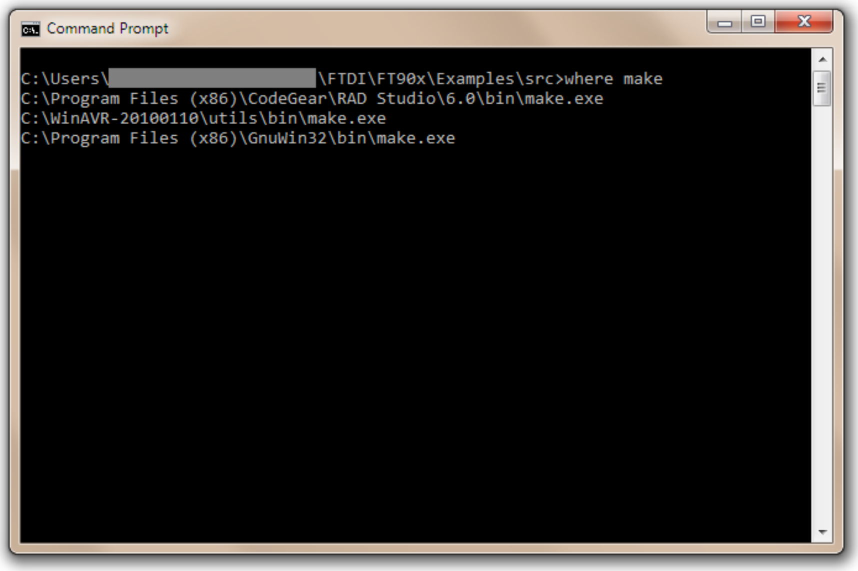
If using a makefile to build an application, some makefile errors may be reported, for example:



```
C:\Users\[redacted]\FTDI\FT90x\Examples\src>make
MAKE Version 5.3  Copyright (c) 1987, 2008 CodeGear
Error makefile 50: Command syntax error
Error makefile 52: Command syntax error
Error makefile 53: Command syntax error
Error makefile 55: Command syntax error
Error makefile 59: Command syntax error
Error makefile 61: Command syntax error
Error makefile 316:  No match found for wildcard '*/%.o'
Error makefile 316: Command syntax error
Error makefile 317:  No match found for wildcard '*/%.o'
Error makefile 317: Command syntax error
*** 10 errors during make ***
```

Figure 61: Makefile Error

This is usually because some existing toolchain on the system may be using its own "make" utility which is also referred to in the PATH variable. The FT90x examples need to be built by the GnuWin32 "make" utility, which can be installed during the toolchain installation. To solve this problem, adjust the PATH variable so that the correct "make" utility is called by the toolchain. Note that it may be necessary to adjust PATH again for the other toolchain. Type "where make" in a command prompt to find out which "make" utilities are present on the system.



```
C:\Users\[redacted]\FTDI\FT90x\Examples\src>where make
C:\Program Files (x86)\CodeGear\RAD Studio\6.0\bin\make.exe
C:\WinAVR-20100110\utils\bin\make.exe
C:\Program Files (x86)\GnuWin32\bin\make.exe
```

Figure 62 "make**" Locations**

8 Contact Information

Head Office – Glasgow, UK

Future Technology Devices International Limited
Unit 1, 2 Seaward Place, Centurion Business Park
Glasgow G41 1HH
United Kingdom
Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758

E-mail (Sales)	sales1@ftdichip.com
E-mail (Support)	support1@ftdichip.com
E-mail (General Enquiries)	admin1@ftdichip.com

Branch Office – Tigard, Oregon, USA

Future Technology Devices International Limited
(USA)
7130 SW Fir Loop
Tigard, OR 97223-8160
USA
Tel: +1 (503) 547 0988
Fax: +1 (503) 547 0987

E-Mail (Sales)	us.sales@ftdichip.com
E-Mail (Support)	us.support@ftdichip.com
E-Mail (General Enquiries)	us.admin@ftdichip.com

Branch Office – Taipei, Taiwan

Future Technology Devices International Limited
(Taiwan)
2F, No. 516, Sec. 1, NeiHu Road
Taipei 114
Taiwan , R.O.C.
Tel: +886 (0) 2 8797 1330
Fax: +886 (0) 2 8751 9737

E-mail (Sales)	tw.sales1@ftdichip.com
E-mail (Support)	tw.support1@ftdichip.com
E-mail (General Enquiries)	tw.admin1@ftdichip.com

Branch Office – Shanghai, China

Future Technology Devices International Limited
(China)
Room 1103, No. 666 West Huaihai Road,
Shanghai, 200052
China
Tel: +86 21 62351596
Fax: +86 21 62351595

E-mail (Sales)	cn.sales@ftdichip.com
E-mail (Support)	cn.support@ftdichip.com
E-mail (General Enquiries)	cn.admin@ftdichip.com

Web Site

http://ftdichip.com

Distributor and Sales Representatives

Please visit the Sales Network page of the FTDI Web site for the contact details of our distributor(s) and sales representative(s) in your country.

Appendix A – References

Document References

http://www.ftdichip.com/Products/ICs/FT90x.html

TN_160 Eclipse Projects

Acronyms and Abbreviations

Terms	Description
CMD	Command-line interface
DLL	Dynamic-link Library
DLOG	Data Log (Project)
GAS	GNU Assembler
GCC	GNU Compiler Collection
GDB	GNU Project Debugger
GNU	GNU (Gnu's Not Unix) Operating System
GUI	Graphical User Interface
IDE	Integrated Development Environment
JDK	Java Development Kit
JRE	Java Runtime Environment
MCU	Microcontroller Unit
PATH	PATH Environment Variable
TCP	Transmission Control Protocol

Appendix B – List of Tables & Figures

List of Figures

Appendix C – Revision History

Document Title:	AN_325 FT90x Toolchain Installation Guide
Document Reference No.:	FT_001041
Clearance No.:	FTDI# 452
Product Page:	http://www.ftdichip.com/FTProducts.htm
Document Feedback:	Send Feedback

Revision	Changes	Date
1.0	Initial release	2014-05-02
1.01	Expanded screenshots of Installation Wizard in Section 2	2015-08-21
1.02	Updated Version for Toolchain 2.1.0	2016-02-22
1.03	Added section 2.1.1 to document the handling of another instance of MSI installer running in the background while installing JRE (results in JRE install error 1618) Debugger related information moved to section 4.4. from Troubleshooting Updated screenshots for programmer	2016-09-19