



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts,Customers Priority,Honest Operation,and Considerate Service",our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China





LPCXpresso v7 User Guide

Rev. 7.5 — 29 October, 2014

User guide












29 October, 2014

Copyright © 2013-2014 NXP Semiconductors

All rights reserved.

- 1. Introduction to LPCXpresso 1
 - 1.1. LPCXpresso IDE Overview of Features 1
 - 1.1.1. Summary of Features 1
 - 1.1.2. New functionality 2
 - 1.1.3. Supported debug probes 3
 - 1.2. LPCXpresso Development Boards (original version) 3
 - 1.3. LPC-Link 2 Debug Probe 4
 - 1.4. LPCXpresso V2 Boards 5
 - 1.5. LPCXpresso MAX Board 5
- 2. Installation and Licensing 7
 - 2.1. Host Computer Hardware Requirements 7
 - 2.2. Installation 7
 - 2.2.1. Windows 7
 - 2.2.2. Linux 8
 - 2.2.3. Mac OS X 10
 - 2.2.4. Running under virtual machines 10
 - 2.3. Licensing Overview 10
 - 2.3.1. Users of earlier versions of LPCXpresso IDE 11
 - 2.3.2. Users of Code Red Technologies Red Suite products 11
 - 2.4. Unregistered (UNREGISTERED) license 11
 - 2.5. Activating your product (LPCXpresso Free Edition) 11
 - 2.6. Activating your product (LPCXpresso Pro Edition) 12
 - 2.6.1. Multi-Seat Activations 12
 - 2.7. Further information on installation and licensing 12
- 3. LPCXpresso IDE Overview 13
 - 3.1. Documentation and Help 13
 - 3.2. Workspaces 13
 - 3.3. Perspectives and Views 13
 - 3.4. Major components of the Develop Perspective 15
- 4. Importing and Debugging example projects 17
 - 4.1. Software drivers and examples 17
 - 4.2. Importing an Example project 17
 - 4.2.1. Importing Examples for the LPCXpresso812 Development Board.... 19
 - 4.3. Building projects 20
 - 4.3.1. Build configurations 20
 - 4.4. Debugging a project 21
 - 4.4.1. Debug Emulator Selection Dialog 22
 - 4.4.2. Controlling execution 25
- 5. Creating Projects using the Wizards 27
 - 5.1. Creating a project using the wizard 27
 - 5.1.1. LPCOpen Library Project Selection 28
 - 5.1.2. CMSIS-CORE selection 29
 - 5.1.3. CMSIS DSP library selection 30
 - 5.1.4. Peripheral Driver selection 30
 - 5.1.5. Code Read Protect 30
 - 5.1.6. Enable use of floating point hardware 30
 - 5.1.7. Enable use of Romdivide library 31
 - 5.1.8. Disable Watchdog 31
 - 5.1.9. LPC1102 ISP Pin 31
 - 5.1.10. Redlib Printf options 31
 - 5.1.11. Project created 31
- 6. Memory Editor and User Loadable Flash Driver mechanism 33
 - 6.1. Introduction 33
 - 6.2. Memory Editor 33
 - 6.2.1. Editing a memory configuration 34
 - 6.2.2. Restoring a memory configuration 37
 - 6.2.3. Copying memory configurations 37

- 6.3. User loadable flash drivers 38
- 6.4. Modifying memory configurations within the New Project Wizards 38
- 7. Multicore projects 40
 - 7.1. LPC43xx Multicore projects 40
 - 7.2. LPC541xx Multicore projects 40
- 8. Red Trace Overview 41
 - 8.1. Serial Wire Viewer 41
 - 8.2. Instruction Trace 41
- 9. Red Trace : SWV Views 42
- 10. Red Trace : SWV Configuration 44
 - 10.1. Starting Red Trace 44
 - 10.1.1. Target Clock Speed 44
 - 10.1.2. Sample rate 45
 - 10.2. Start Trace 45
 - 10.3. Refresh  45
 - 10.4. Settings  45
 - 10.5. Reset Trace  45
 - 10.6. Save Trace  45
- 11. Red Trace : Profiling 46
 - 11.1. Overview 46
 - 11.2. Profile view  46
- 12. Red Trace : Interrupt tracing 48
 - 12.1. Overview 48
 - 12.2. Interrupt Statistics view  48
 - 12.3. Interrupt Trace view  48
- 13. Red Trace : Data Watch Trace 51
 - 13.1. Overview 51
 - 13.2. Data Watch view  51
 - 13.2.1. Item Display 53
 - 13.2.2. Trace Display 54
- 14. Red Trace : Host Strings (ITM) 55
 - 14.1. Overview 55
 - 14.2. Defining Host Strings 55
 - 14.3. Building the Host Strings macros 57
 - 14.4. Instrumenting your code 57
 - 14.5. Host Strings view  58
- 15. Red Trace : Instruction Trace 59
 - 15.1. Getting Started 59
 - 15.1.1. Configuring the Cortex-M0+ for Instruction Trace 59
 - 15.1.2. Trace the most recently executed instructions 60
 - 15.1.3. Stop trace when a variable is set 61
 - 15.2. Concepts 63
 - 15.2.1. Instruction Trace Overview 63
 - 15.2.2. MTB Concepts 63
 - 15.2.3. Embedded Trace Macrocell 65
 - 15.2.4. Embedded Trace Buffer 67
 - 15.2.5. Data Watchpoint and Trace 69
 - 15.3. Reference 71
 - 15.3.1. Instruction trace view 71
 - 15.3.2. Instruction Trace view Toolbar buttons 72
 - 15.3.3. Instruction Trace Config view for the MTB 75
 - 15.3.4. Instruction Trace Config view for the ETB 76
 - 15.3.5. Supported targets 78

15.4. Troubleshooting	78
15.4.1. General	78
15.4.2. MTB	79
15.4.3. ETB	79
16. Red State Overview	80
16.1. The NXP State Configurable Timer	80
16.2. Software State Machine	80
16.3. Integrating a state machine with your project	80
17. Red State : SCT state machine tutorial	81
17.1. Prerequisites	81
17.2. Creating a new project for the SCT	81
17.3. Adding a new SCT state machine to the project	81
17.4. The blinky state machine overview	83
17.5. Naming outputs and inputs	83
17.6. Matching the timer	84
17.7. The states	84
17.7.1. Special states	84
17.7.2. Deleting a state	84
17.7.3. Adding states	84
17.8. Adding transitions	86
17.8.1. Creating a new transition	86
17.8.2. Adding a signal to a transition	87
17.8.3. Adding action elements to transitions	88
17.8.4. Turning on <code>LED1</code>	89
17.8.5. Turning off <code>LED1</code>	90
17.8.6. Remaining transitions	90
17.9. Generating the configuration code	91
17.9.1. Files generated	91
17.9.2. Issues and warnings	92
17.10. Incorporating with your code	92
18. Red State : Software state machine tutorial	93
18.1. Software state machine tutorial overview	93
18.1.1. Building a traffic light example	93
18.2. Creating a new project	93
18.2.1. Importing the base project	93
18.3. Extending the LED Traffic base project	94
18.3.1. Add the state machine to the project	94
18.3.2. Adding states to the State Machine	96
18.3.3. Adding inputs	98
18.3.4. Adding outputs	99
18.3.5. The Initial State and the Reset signal	99
18.3.6. Adding a transition	100
18.3.7. Creating a signal	101
18.3.8. Adding a signal to a transition	102
18.3.9. Adding actions to a transition	102
18.3.10. Transition on button press	102
18.4. Integrating a state machine with existing code	104
18.4.1. Editing <code>main</code>	105
18.4.2. Generating the state machine code	105
18.4.3. Editing the actions C file	106
18.4.4. Accessing the outputs	107
18.4.5. Setting inputs in interrupt handlers	107
18.4.6. Running on the target	108
18.4.7. Other examples	108
19. Red State : New state machine Wizard	109
19.1. SCT Wizard options	109
19.2. Software State Machine Wizard options	110

- 20. Red State : The state machine editor 112
 - 20.1. Overview 112
 - 20.2. States 113
 - 20.2.1. Creating 113
 - 20.2.2. Naming 113
 - 20.2.3. Resizing 114
 - 20.2.4. Deleting 114
 - 20.2.5. Setting initial state 114
 - 20.3. Transitions 114
 - 20.3.1. Adding transitions 114
 - 20.3.2. Deleting transitions 115
 - 20.3.3. Adding signals to a transition 115
 - 20.3.4. Adding actions to a transition 115
 - 20.3.5. Appearance of transitions 115
 - 20.4. Signals 116
 - 20.5. Actions 117
 - 20.6. Inputs 117
 - 20.6.1. SCT inputs 117
 - 20.6.2. Software state machine inputs 118
 - 20.7. Outputs 118
 - 20.7.1. SCT outputs 118
 - 20.7.2. Software state machine outputs 119
 - 20.7.3. Preset values 119
- 21. Red State : Limitations 120
- 22. Red State : Frequently Asked Questions 121
 - 22.1. How do I migrate from a Red State project created in Red Suite /
LPCXpresso v4 to one created in this version 121
- 23. Appendix A – File Icons 124
- 24. Appendix B – Glossary of Terms 125

1. Introduction to LPCXpresso

LPCXpresso is a low-cost microcontroller (MCU) development platform ecosystem from NXP, which provides an end-to-end solution enabling embedded engineers to develop their applications from initial evaluation to final production.

The LPCXpresso platform ecosystem includes:

- The LPCXpresso IDE, a software development environment for creating applications for NXP's ARM based 'LPC' range of MCUs.
- The range of LPCXpresso development boards, which each include a built-in 'LPC-Link' or 'LPC-Link2' debug probe. These boards are developed in collaboration with Embedded Artists.
- The standalone 'LPC-Link 2' debug probe.

This guide is intended as an introduction to using LPCXpresso, with particular emphasis on the LPCXpresso IDE. It assumes that you have some knowledge of MCUs and software development for embedded systems.

1.1 LPCXpresso IDE Overview of Features

The LPCXpresso IDE is a fully featured software development environment for NXP's ARM-based MCUs, and includes all the tools necessary to develop high quality embedded software applications in a timely and cost effective fashion.

The LPCXpresso IDE is based on the Eclipse IDE and features many ease-of-use and MCU specific enhancements. The LPCXpresso IDE also includes the industry standard ARM GNU tools enabling professional quality tools at low cost. The fully featured debugger supports both SWD and JTAG debugging, and features direct download to on-chip flash.

1.1.1 Summary of Features

- Complete C/C++ integrated development environment
 - Latest Eclipse-based IDE with many ease-of-use enhancements
 - IDE can be further enhanced with Eclipse plugins
 - CVS source control built in; Subversion, TFS, Git, and others available for download
 - Command-line tools included for integration into build, test, and manufacturing systems
- Industry standard GNU toolchain, including
 - C and C++ compilers, assembler, and linker
 - Converters for SREC, HEX, and binary
- Fully featured debugger supporting JTAG and SWD
 - Built-in flash programming
 - High-level and instruction-level debug
 - Views of CPU registers and on-chip peripherals

- Support for multiple devices on JTAG scan-chain
- Library support
 - Redlib: a small-footprint embedded C library
 - Newlib: a complete C and C++ library
 - NewlibNano: a new small-footprint C and C++ library, based on Newlib
 - LPCOpen MCU software libraries
 - Cortex Microcontroller Software Interface Standard (CMSIS) libraries and source code
- Device-specific support for NXP's ARM-based MCUs (including Cortex-M, ARM7 and ARM9 based parts)
 - Automatic generation of linker scripts for correct placement of code and data into flash and RAM
 - Startup code and device initialization
 - No assembler required with Cortex-M based MCUs
- *Red Trace* [41]
 - Instruction trace via Embedded Trace Buffer (ETB) on certain Cortex-M3/M4 based MCUs or Micro Trace Buffer (MTB) on Cortex-M0+ based MCUs
 - Plus when debugging via Red Probe+ on Cortex-M3/M4 based MCUs, Serial Wire Viewer support providing:
 - Profile tracing
 - Interrupt trace and display
 - Datawatch trace
- *Red State* [80] state machine designer and code generator
 - Graphically design your state machines
 - Generates standard C code
 - Configures NXP State Configurable Timer (SCT) as well as supporting software state machines

1.1.2 New functionality

The following changes in functionality have been made in LPCXpresso IDE v7 compared to the previous release (v6).

- New release of the GNU compilers – v4.8.3.
 - Includes new 'general' optimization level, -Og. This new optimization level, aims at providing fast compilation, a superior debugging experience and reasonable runtime performance.
 - Adds Link Time Optimization (LTO), which is sometimes known as Whole Program Optimization. This allows all the different compilation units that make up a single executable to be optimized as a single module.

- Inclusion of a new small-footprint variant of the Newlib C and C++ library, known as NewlibNano. Use of this library can result in significantly smaller code size, especially of C++ applications.
- [Note that further details on the use of these new options can be found in the compiler documentation that is provided in the IDE help system.]
- Note that LPCXpresso IDE v7.3.0 upgrades the GNU Compilers to v4.8.4
- New release of the base Eclipse IDE – Luna (v4.4) and CDT (8.4)
- The Managed Linker script mechanism has been extended to support the features of new GNU compiler.

For the latest details on new features and functionality, visit <http://www.lpcware.com/content/forum/lpcxpresso-latest-release>

1.1.3 Supported debug probes

The following debug probes are supported by LPCXpresso IDE for general debug connections:

- LPC-Link (LPCXpresso board)
- LPC-Link 2 (with “Redlink” firmware) - either the standalone debug probe or the version built into LPCXpresso V2 boards
- CMSIS-DAP enabled debug probes, such as LPC800-MAX, Keil ULINK-ME etc.
- Red Probe / Red Probe+
- RDB1768 development board built-in debug connector (RDB-Link)
- RDB4078 development board built-in debug connector

Note that not all Red Trace functionality is supported by all debug probes. For more details on Red Trace, please see Chapter 8.

Support for GDB server based debug connections is provided. This feature enables support for 3rd party debug probes, such as Segger J-Link. When debugging with GDB server connections, some functionality may be disabled.

- For more information on using Segger J-Link with LPCXpresso, visit <http://www.segger.com/nxp-lpcxpresso.html>

1.2 LPCXpresso Development Boards (original version)

Since first introduced in 2009, the original LPCXpresso family of boards, along with the associated LPCXpresso IDE, has re-energised the whole MCU evaluation board market.

Developed in collaboration with Embedded Artists, each LPCXpresso board contains a JTAG/SWD debug probe called “LPC-Link” and a target MCU. LPC-Link is equipped with a 10-pin JTAG/SWD header and it seamlessly connects the LPCXpresso IDE to the target MCU via USB (the USB interface and other debug features are provided by NXP’s ARM9 based LPC3154 MCU). The target includes a small prototyping area and easily accessible connections for expansion. An LED is also fitted as standard, with some board variants having additional fittings such as an RGB LED, potentiometer or USB device connector.

An LPCXpresso board with its on-board target MCU can be used either on its own for software development and benchmarking or connected to an off-the-shelf baseboard, such as those available from Embedded Artists, for rapid proof-of-concepts.

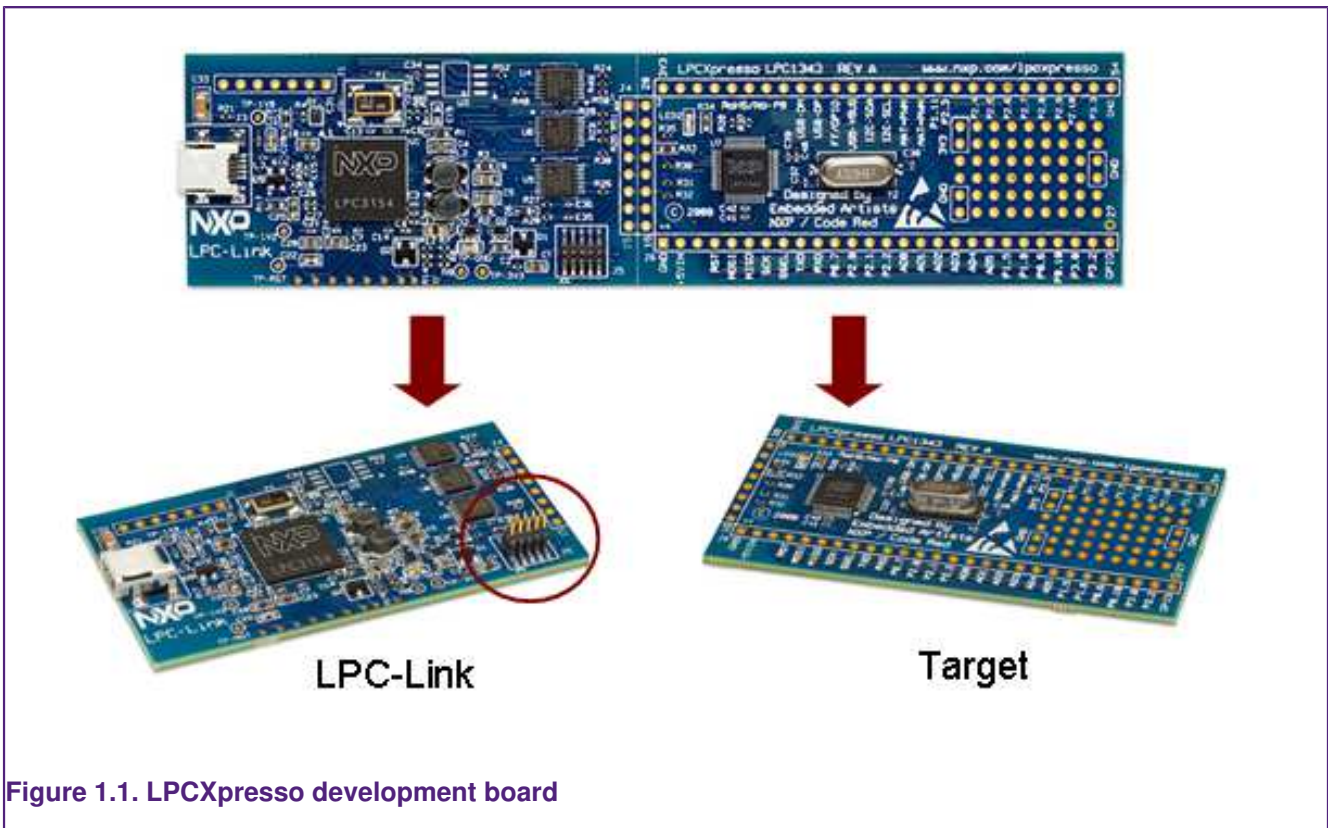


Figure 1.1. LPCXpresso development board

Cutting the tracks between the LPC-Link and the target will change the LPC-Link into a standalone JTAG/SWD debug probe. This enables the LPCXpresso platform to be connected to an external target, which may be an off-the-shelf commercial development board or a target board of your own design.

For more information, visit:

<http://www.lpcware.com/LPCXpressoBoards>

1.3 LPC-Link 2 Debug Probe

The LPC-Link 2 is a new generation of debug probe. It is powered by an NXP LPC4300 series MCU, and includes a standard 10-pin JTAG/SWD connector; a 20-pin JTAG/SWD/ETM connector; and analog, digital and serial expansion headers, making it a highly extensible platform. Unlike the original “LPC-Link” it does not contain a target MCU, but is rather designed to be used with a target mounted on an external board.

The LPCXpresso IDE works out of the box with LPC-Link 2 using “Redlink” firmware. In addition, several firmware images are available for LPC-Link 2 that make it compatible with other toolchains, including the popular SEGGER J-Link and the CMSIS-DAP debugger designed by ARM.

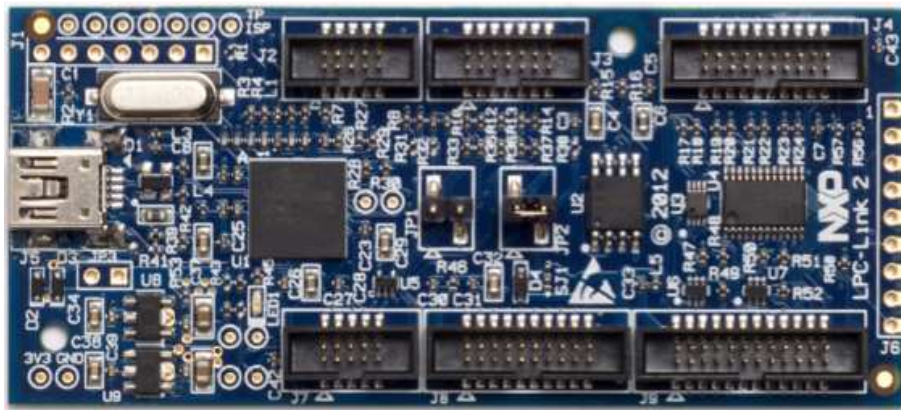


Figure 1.2. LPC-Link 2 Debug Probe

For more information, visit:

<http://www.lpcware.com/lpclink2>

1.4 LPCXpresso V2 Boards

The next generation LPCXpresso V2 boards build upon the original LPCXpresso board design, providing many new and exciting features. These include a new debug probe based on the standalone LPC-Link2, as well as enhanced expansion capabilities.



Figure 1.3. LPCXpresso V2 Board (LPCXpresso11U68)

For more information, visit:

<http://www.lpcware.com/LPCXpressoBoards>

1.5 LPCXpresso MAX Board

The LPCXpresso-MAX family of boards provide a powerful and flexible development system for some of NXP's low end MCUs, such as the LPC812 and LPC824. They can be

used out of the box with a range of development tools, including the LPCXpresso IDE, as well as with ARM’s mbed online toolchain.

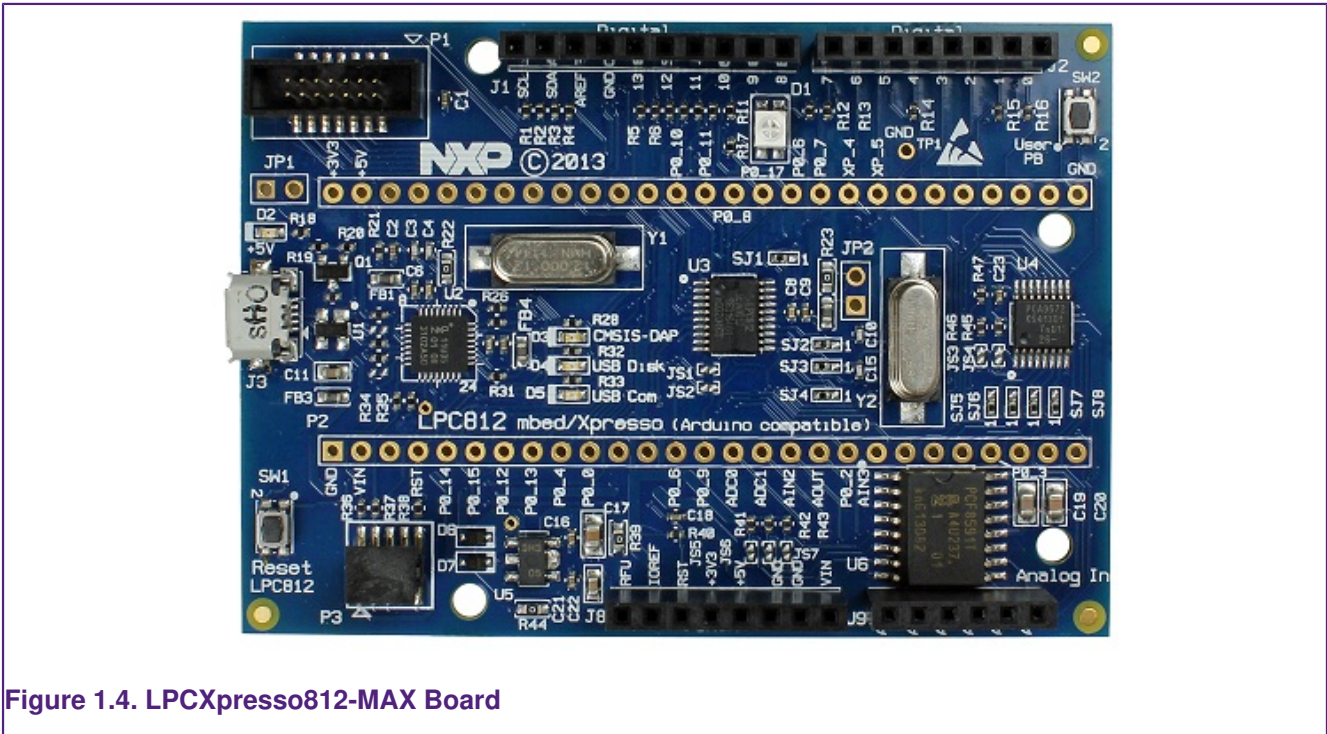


Figure 1.4. LPCXpresso812-MAX Board

For more information, visit:

<http://www.lpcware.com/LPCXpressoBoards>

2. Installation and Licensing

2.1 Host Computer Hardware Requirements

Before installation of the LPCXpresso IDE, you should make sure your development host computer meets the following requirements:

- A standard x86 PC with 2GB RAM minimum (4GB+ recommended) and 600MB+ of available disk space, running one of the following operating systems:
 - Microsoft® Windows Vista
 - Microsoft® Windows 7
 - Microsoft® Windows 8 and Windows 8.1
 - Linux – Ubuntu 12 and later
 - Linux – Fedora 18 and later
- An x86 Apple Macintosh with 2GB RAM minimum (4GB+ recommended) and 600MB+ of available disk space, running one of the following Mac OS X versions:
 - 10.8.5 (or later) / 10.9.4 (or later) / 10.10 (or later)

Additional host platform notes:

- Starting with LPCXpresso v7.1, and following the discontinuation of support by Microsoft, Windows XP is no longer an officially supported platform. LPCXpresso **may** continue to work on Windows XP but this can no longer be guaranteed. LPCXpresso is no longer tested on Windows XP.
- Starting with LPCXpresso v7.4, Mac OS X 10.7 (Lion) is no longer an officially supported platform. LPCXpresso **may** continue to work on Mac OS X 10.7 but this can no longer be guaranteed. LPCXpresso is no longer tested on Mac OS X 10.7.
- Both 32-bit and 64-bit Windows / Linux systems are supported.
- The LPCXpresso IDE may install and run on other Linux distributions. However, only the distributions listed above have been tested. We have no plans to officially support other distributions at this time.
- A screen resolution of 1024x768 minimum is recommended.
- An internet connection is **required** to request, install, and activate license keys. When using the product, an internet connection is required to update the product and to download new examples.

2.2 Installation

When installing, all components of the LPCXpresso IDE are installed, but some functionality may be restricted by the currently installed license activation.

2.2.1 Windows

The LPCXpresso IDE is installed into a single directory, of your choice. Unlike many software packages, the LPCXpresso IDE does not install or use any keys in the Windows Registry, or use or modify any environment variables (including PATH), resulting in a very clean installation that does not interfere with anything else on your PC. Should you wish

to use the command-line tools, a command file is provided to set up the path for the local command window.

2.2.2 Linux

Due to the huge variation in capabilities of different Linux distributions, LPCXpresso is only tested and supported on recent distributions of Ubuntu and Fedora. The LPCXpresso IDE **may** work on other distributions but we cannot provide support if it does not work.

The installer is supplied as an executable that installs the LPCXpresso IDE components. The installer requires root privileges, although, once it is installed, no special privileges are required to run the LPCXpresso IDE. The installer will request a super-user password when it is started. Once installation has completed, we strongly recommend that your system is restarted — if you do not do this then some areas of the tools may not function correctly. The installer should be started from the command line, but will switch to GUI mode once the super-user password has been entered.

Installation on 32-bit distributions

The LPCXpresso IDE is a 32-bit product that should run on most 32-bit Linux distributions. On some distributions some additional packages may be required for USB compatibility (see below for the latest information). Note that GLIBC v2.15 or greater is required.

Enabling the LPCXpresso IDE internal Web Browser

On most recent Linux distributions, an additional package must be installed to enable the IDE's internal Web Browser. If this is not installed, any web pages displayed by the IDE will be displayed in the system default Web Browser.

Ubuntu

Run the following command to install the webkit package that enables the internal Web Browser:

```
sudo apt-get install libwebkitgtk-1.0-0
```

Fedora

Run the following command to install the webkit package that enables the internal Web Browser:

```
sudo yum install webkitgtk.i686
```

Installation on 64-bit distributions

The LPCXpresso IDE is a 32-bit product. For 64-bit versions of Linux, the 32-bit compatibility components must be installed. Note that all of these components must be installed, otherwise the installation program will not run and the LPCXpresso IDE will not function correctly. To install these components from the command line, follow the instructions below for your distribution.

Ubuntu 13.10 or later:

Ubuntu 13.10 no longer provides a convenient method to install all 32-bit compatible libraries, so they must all be installed individually. Those that are required for LPCXpresso 7 are as follows:

```
sudo apt-get install libgtk2.0-0:i386 libxtst6:i386 libpangox-1.0-0:i386 \  
libpangoxft-1.0-0:i386 libidn11:i386 libglu1-mesa:i386 \  
libglu1-mesa:i386
```

```
libncurses5:i386 libudev1:i386 libusb-1.0:i386 libusb-0.1:i386 \
gtk2-engines-murrine:i386 libnss3-ld:i386 libwebkitgtk-1.0-0
```

Ubuntu 13.04 or earlier:

Ubuntu 13.04 and earlier provides a convenient method to install 32-bit compatible libraries. These can be installed using the follow command:

```
sudo apt-get install linux32 ia32-libs
```

Fedora

Install the following 32-bit libraries:

```
sudo yum install gtk2.i686 glibc.i686 glibc-devel.i686 libstdc++.i686 \
zlib-devel.i686 ncurses-devel.i686 libX11-devel.i686 libXrender.i686 \
libXrandr.i686 libusb.i686 libXtst.i686 nss.i686 libcanberra-gtk2.i686 \
PackageKit-gtk3-module.i686 webkitgtk.i686
```

Other Linux Distros

While not officially supported, the LPCXpresso IDE has been reported to work on many other Linux Distros, including Linux Mint, openSUSE and Debian. When attempting to run on these Distros, remember that LPCXpresso is a 32-bit application and so various 32-bit compatible libraries must be installed. These include:

```
libgtk2.0-0:i386 libxtst6:i386 libpangox-1.0-0:i386 libpangoxft-1.0-0:i386 \
libidn11:i386 libglu1-mesa:i386 libncurses5:i386 libudev1:i386 \
libusb-1.0:i386 libusb-0.1:i386 gtk2-engines-murrine:i386 \
libnss3:i386
```

Also note that that the glibc 2.15 shared library is required.

We are unable to provide assistance when installing on other Distros, but the LPCXpresso forum on <http://www.lpcware.com> is a good place to search for information or post questions.

Post-installation issues

dfu-util fails to run

On some Linux systems, when booting LPC-Link or LPC-Link2, the supplied version of dfu-util may fail to execute. To resolve this you may need to install an additional component:

```
sudo apt-get install libusb-0.1-4:i386 # Ubuntu
```

This should only be installed on systems where dfu-util fails to run.

Connection Refused error

When starting a debug session, on some Linux systems, a **"Connection refused"** error may be displayed. This happens because a critical system library is not installed where we expect to find it.

To resolve this issue, open a Terminal Window and execute the following commands (if running a 64-bit version of Linux, you must first install the 32-bit compatibility libraries as described above):

Note: The actual location of this library may depend on the distribution and version of Linux you are running.

Ubuntu

```
cd /lib/i386-linux-gnu
sudo ln -sf libudev.so.1 libudev.so.0
```

Fedora

```
cd /usr/lib/
sudo ln -sf libudev.so.1 libudev.so.0
```

Ubuntu-specific issues

When using the Unity interface, there may be an issue preventing some menu items from displaying in the LPCXpresso IDE (this does not affect the 'Classic' interface). To workaround this problem, create a shell script with the following content, and start the LPCXpresso IDE by running this script:

```
#!/bin/bash
export UBUNTU_MENUPROXY=0
/usr/local/<lpcxpresso_install_dir>/lpcxpresso/lpcxpresso
```

Fedora-specific issues

If SELINUX is used, it must be set to "permissive" mode to allow the LPCXpresso IDE to run.

2.2.3 Mac OS X

The LPCXpresso IDE installer is supplied as a Mac OS X `.pkg` installer file. Double click on the installer to install the LPCXpresso IDE into a subfolder of your Applications folder.

To start the LPCXpresso IDE, use the Mac OS X Launchpad. Alternatively click the **Open lpcxpresso** icon in the `/Applications/lpcxpresso_version` folder or run **lpcxpresso.app**, which can be found in the `lpcxpresso` subfolder of the main LPCXpresso IDE installation directory within `/Applications`.

2.2.4 Running under virtual machines

It is possible to install the LPCXpresso IDE within a virtual machine (VM) environment. Generally such installations cause few issues. However due to the nature of VMs the most likely problems relate to sharing of resources (USB, memory), and possible timeouts during debug operations.

In the unlikely event that you experience issues, we welcome reports but due to the nature of VM operation can offer no guarantee of resolution.

2.3 Licensing Overview

An LPCXpresso IDE installation will take one of the following three forms:

- **Unregistered** : the initial product state following a new customer installation, features are restricted, and code size is restricted to 8KB
- **Free Edition** : following installation of a Free Edition Activation code, maximum debug download size is limited to 256KB

- **Pro Edition** : following installation of Purchased Pro Edition Activation code, no restrictions

Note the current installation type can be identified by Help -> Display License Type

2.3.1 Users of earlier versions of LPCXpresso IDE

If you have previously been using LPCXpresso IDE v5.x or earlier, then note that your previous activation code is not compatible with this version. You will need to go through the activation process again in order to use this version.

If you have previously been using LPCXpresso IDE v6.x, then your existing activation code is compatible with LPCXpresso IDE v7.x. There is no need to reactivate.

2.3.2 Users of Code Red Technologies Red Suite products

Red Suite activation codes are not compatible with LPCXpresso IDE. You will need to obtain an LPCXpresso Free or Pro Edition activation code in order to use LPCXpresso IDE.

2.4 Unregistered (UNREGISTERED) license

Initially after installation LPCXpresso IDE will run with a default Unregistered (UNREGISTERED) license. Most features of the product may be used, although some functionality is restricted, including the size of applications that you can build and debug (limited to 8Kbytes), and Red Trace functionality is disabled. Activate your product with a Free Edition or Pro Edition license to remove these restrictions.

2.5 Activating your product (LPCXpresso Free Edition)

A Free Edition activation code may be obtained, **free of charge**, by registering the LPCXpresso IDE. This provides a license to use the complete development environment with a 256KB debug download size limit.

Note: You will need to have created an account and logged on to the LPCWare website to be able to obtain a Free Edition activation code.

To activate your installation with a Free Edition license, from within LPCXpresso IDE:

1. Go to the menu entry **Help->Activate->Create Serial number and register (Free Edition)...**
 - Your product's serial number will be displayed
 - Write down the serial number, or copy it into the clipboard.
2. Press **OK** and a web browser will be opened on the Activations page
 - If you are already logged in to the website, the serial number will be completed for you.
 - If you are not logged in, you will need to login, navigate to <http://www.lpcware.com/lpcxpresso/activate>, and enter the product's serial number.
3. Press the button to Register LPCXpresso
 - Your LPCXpresso Activation code will be generated and displayed.
4. Go to the menu entry **Help->Activate->Activate (Free Edition)...**
5. Enter your activation code and Press **OK**.

- This activates your product. The license type will be displayed and you will be able to use all the features of LPCXpresso, with a debug download limit of 256KB.

2.6 Activating your product (LPCXpresso Pro Edition)

A full, unrestricted activation code for the LPCXpresso IDE can be purchased via the menu entry **Help->Activate->Purchase from LPCXpresso webstore**. Once purchased, your activation code will be emailed to you.

When the activation code is received, follow the instructions below to activate your product. From within the LPCXpresso IDE:

1. Go to the menu entry **Help->Activate->Activate (Pro Edition)...**
2. Enter the Pro Edition activation code provided, and press **OK**
3. Enter your email address and provide a password of your choosing, and press **OK**.

Your product will now be activated. The license type will be displayed and you will be able to use all the features of LPCXpresso with no code size limits.

NOTES

1. The password should be kept safe, as it will be required should you wish reactivate your license (for example to move your license for LPCXpresso Pro Edition from one PC to another). Your email address will be used to send a reminder should you forget your password. We may also send occasional emails from which you may unsubscribe.
2. The first use of an activation code will also trigger an email from Softworkz (our licensing system partners) to the supplied email address. This email is important because it offers the option to create an account to centrally manage multiple seats of LPCXpresso. See <http://www.lpcware.com/content/faq/lpcxpresso/license-control-panel-email>

2.6.1 Multi-Seat Activations

When you purchase LPCXpresso Pro Edition, you can choose to purchase a single or multi seat license. If you purchase a multi seat license, then you will be provided with a single activation code that can be used to activate LPCXpresso Pro Edition on multiple machines.

The major benefit of a multi seat activation code over multiple single seat codes is that this enables a single activation code to be made available to users whilst offering the ability to monitor and administer usage centrally. This functionality is provided by a feature called the License Control Panel (LCP). Single seat licenses can be upgraded to multi-seat licenses by purchasing additional seats.

For additional information please visit:

<http://www.lpcware.com/content/faq/lpcxpresso/central-administration-activation-codes>

2.7 Further information on installation and licensing

Further information on LPCXpresso IDE installation, licensing and activation codes can be found in our FAQs at:

<http://www.lpcware.com/faq/activation-licensing>

3. LPCXpresso IDE Overview

3.1 Documentation and Help

The LPCXpresso IDE is based on the Eclipse IDE framework, and many of the core features are described well in generic Eclipse documentation and in the help files to be found on the LPCXpresso IDE's **Help -> Help Contents** menu. This also provides access to the LPCXpresso User Guide (this document), as well as the documentation for the compiler, linker, and other underlying tools.

To obtain assistance on using LPCXpresso visit

<http://lpcware.com/lpcxpresso/support>

3.2 Workspaces

When you first launch LPCXpresso IDE, you will be asked to select a Workspace, as shown in Figure 3.1.

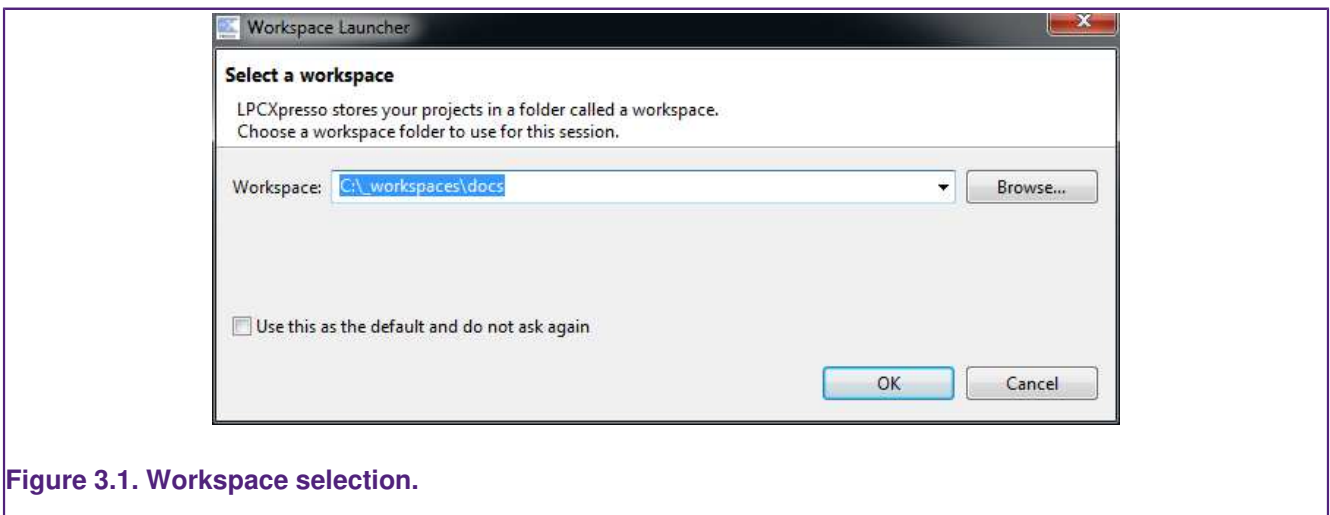


Figure 3.1. Workspace selection.

A workspace is simply a directory that is used to store the projects you are currently working on. Each workspace can contain multiple projects, and you can have multiple workspaces on your computer. The LPCXpresso IDE can only have a single workspace open at a time, although it is possible to run multiple instances in parallel — with each instance accessing a different workspace.

If you tick the **Use this as the default and do not ask again** option, then the LPCXpresso IDE will always start up with the chosen workspace opened. Otherwise you will always be prompted to choose a workspace.

It is also possible to change workspace whilst running the LPCXpresso IDE, using the **File -> Switch Workspace** option.

3.3 Perspectives and Views

The overall layout of the main LPCXpresso IDE window is known as a Perspective. Within each Perspective are many sub-windows, called Views. A View displays a particular set of data in the LPCXpresso environment. For example, this data might be source code,

hex dumps, disassembly, or memory contents. Views can be opened, moved, docked, and closed, and the layout of the currently displayed Views can be saved and restored.

Typically, the LPCXpresso IDE operates using the single **Develop Perspective**, under which both code development and debug sessions operate as shown in Figure 3.3. This single perspective simplifies the Eclipse environment, but at the cost of slightly reducing the amount of information displayed on screen.

Alternatively the LPCXpresso IDE can operate in a 'dual perspective' mode such that the **C/C++ Perspective** is used for developing and navigating around your code and the **Debug Perspective** is used when debugging your application.

You can manually switch between Perspectives using the Perspective icons in the top right of the LPCXpresso IDE window, as per Figure 3.2.

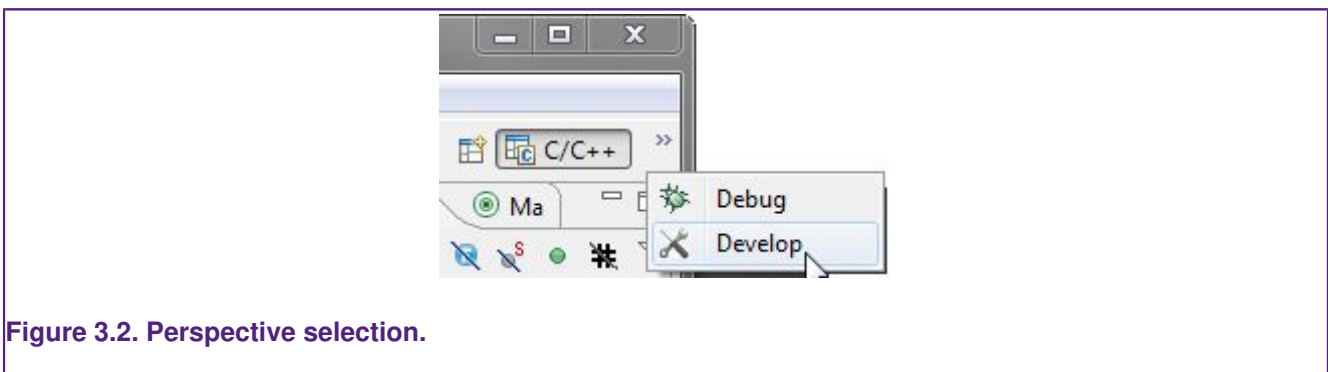


Figure 3.2. Perspective selection.

All Views in a Perspective can also be rearranged to match your specific requirements by dragging and dropping. If a View is accidentally closed, it can be restored by selecting it from the **Window -> Show View** dialog.

3.4 Major components of the Develop Perspective

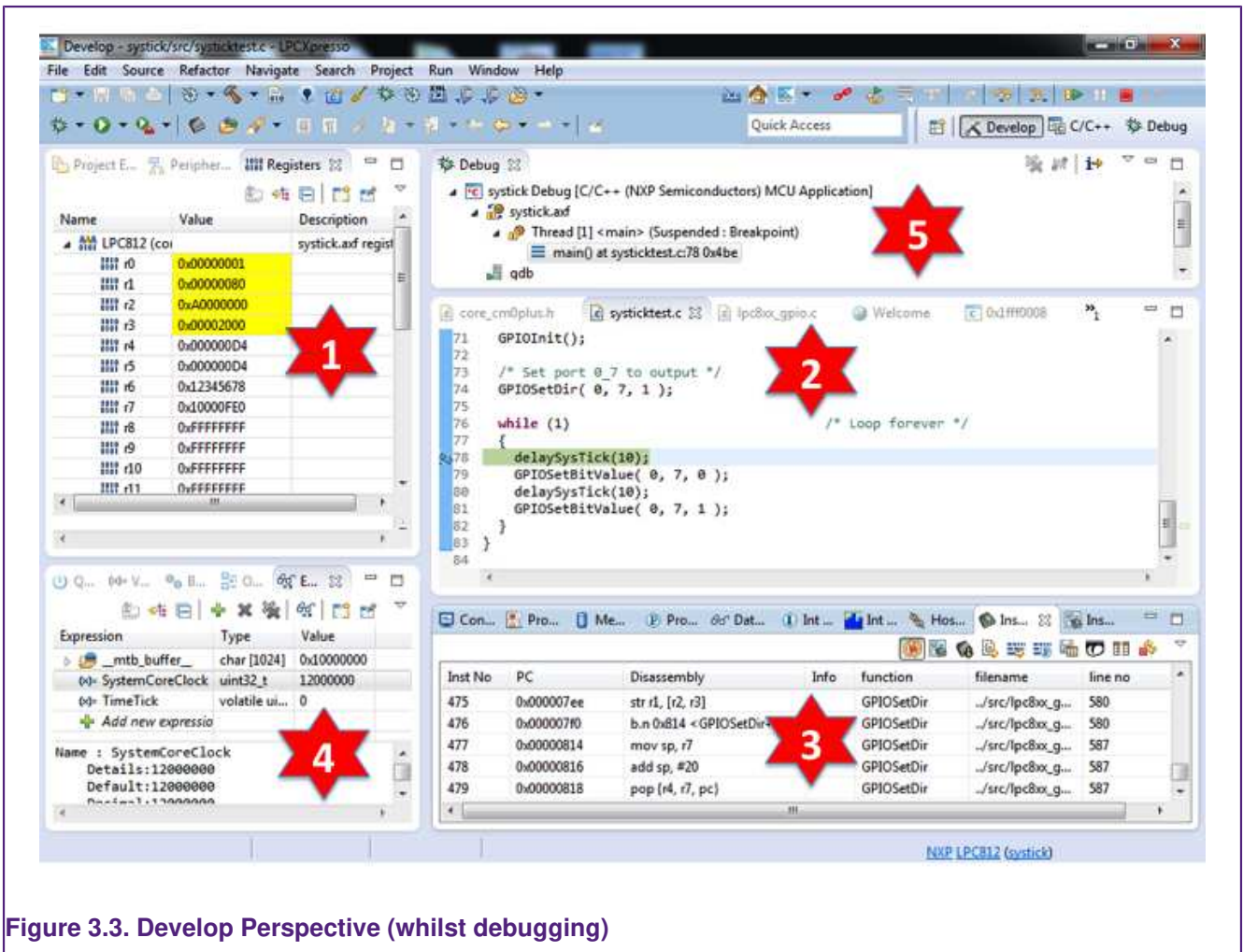


Figure 3.3. Develop Perspective (whilst debugging)

1. Project Explorer / Peripherals / Registers Views
 - The **Project Explorer** gives you a view of all the projects in your current 'Workspace'.
 - When debugging, the **Peripherals** view allows you to display the registers within Peripherals.
 - When debugging, the **Registers** view allows you to display the registers within the CPU of your MCU.
2. Editor
 - On the upper right is the editor, which allows modification and saving of source code. When debugging, it is here that you will see the code you are executing and can step from line to line. By pressing the 'i->' icon at the top of the Debug view, you can switch to stepping by assembly instruction. Clicking in the left margin will set and delete breakpoints.
3. Console / Problems / Red Trace Views
 - On the lower right are the Console and Problems Views. The Console View displays status information on compiling and debugging, as well as semihosted program output. The Problem View (available by changing tabs) shows all compiler errors and will allow easy navigation to the error location in the Editor View.

- Located in parallel with the Console View are the various views that make up the Red Trace functionality of LPCXpresso IDE. The Red Trace views allow you to gather and display runtime information using the SWV technology that is part of Cortex-M3/M4 based parts. In addition, for some MCUs, you can also view instruction trace data downloaded from the MCU's Embedded Trace Buffer (ETB) or Micro Trace Buffer (MTB). The example here shows instruction trace information downloaded from an LPC812's MTB. For more information on Red Trace functionality, please see Chapter 8.

4. Quick Start / Variables / Breakpoints / Expressions Views

- On the lower left of the window, the **Quickstart Panel** has fast links to commonly used features. This is the best place to go to find options such as Build, Debug, and Import.
- Sitting in parallel to the 'Quickstart' view, the **Variable** view allows you to see the values of local variables.
- Sitting in parallel to the 'Quickstart' view, the **Breakpoint** view allows you to see and modify currently set breakpoints.
- Sitting in parallel to the 'Quickstart' view, the **Expressions** view allows you to add global variables and other expressions so that you can see and modify their values.

5. Debug View

- The Debug view appears when you are debugging your application. This shows you the stack trace. In the 'stopped' state, you can click on any particular function and inspect its local variables in the Variables tab (which is located parallel to the **Quickstart Panel**).

4. Importing and Debugging example projects

4.1 Software drivers and examples

LPCOpen is now the preferred software platform for most NXP Cortex-M based MCUs, replacing the various CMSIS / Peripheral Driver Library / code bundle software packages made available in the past. LPCOpen has been designed to allow you to quickly and easily utilize an extensive array of software drivers and libraries in order to create and develop multifunctional products. Amongst the features of LPCOpen are:

- MCU peripheral device drivers with meaningful examples
- Common APIs across device families
- Thoroughly tested and maintained
- Commonly needed third party and open source software ports
- Support for Keil, IAR and LPCXpresso toolchains

The latest LPCOpen v2 now available provides:

- MCU family specific download package
- Support for USB ROM drivers
- Improved code organization and drivers (efficiency, features)
- Improved support for LPCXpresso IDE

CMSIS / Peripheral Driver Library / code bundle software packages are still available, both within your LPCXpresso IDE install directory `\lpcxpresso\Examples\NXP` and also downloadable from NXP LPCware website. But generally these should only be used for existing development work. When starting a new evaluation or product development, we would recommend the use of LPCOpen.

More information on LPCOpen together with package downloads can be found at:

<http://www.lpcware.com/lpcopen>

4.2 Importing an Example project

The **Quickstart Panel** provides rapid access to the most commonly used features of the LPCXpresso IDE. Using the **Quickstart Panel**, you can quickly import example projects, create new projects, build projects and debug projects.

On the **Quickstart Panel**, click on the 'Start Here' sub-panel, and click on Import project(s).

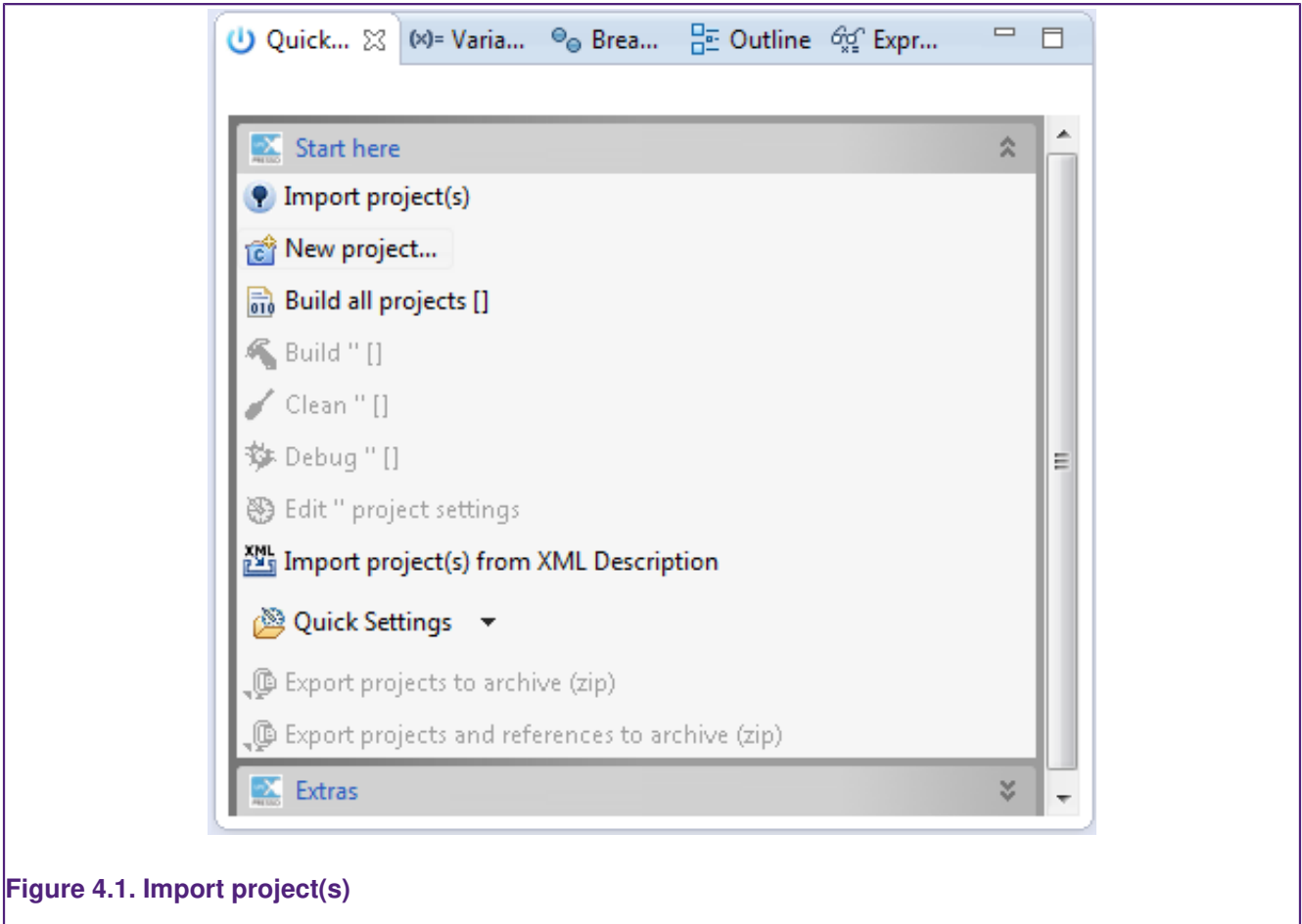


Figure 4.1. Import project(s)

As per Figure 4.2, from the first page of the wizard, you can

- Browse to locate Examples stored in zip archive files on your local system. These could be archives that you have previously downloaded (for example LPCOpen packages from the NXP LPCware website or the supplied, but deprecated, sample code bundles located within the Examples subdirectory of your LPCXpresso IDE installation).
- Browse to locate projects stored in directory form on your local system (for example, you can use this to import projects from a different workspace into the current workspace).
- Browse LPCOpen packages to visit LPCware and download appropriate LPCOpen package for your target MCU. This option will automatically open a web browser onto an appropriate links page on the NXP LPCware website.

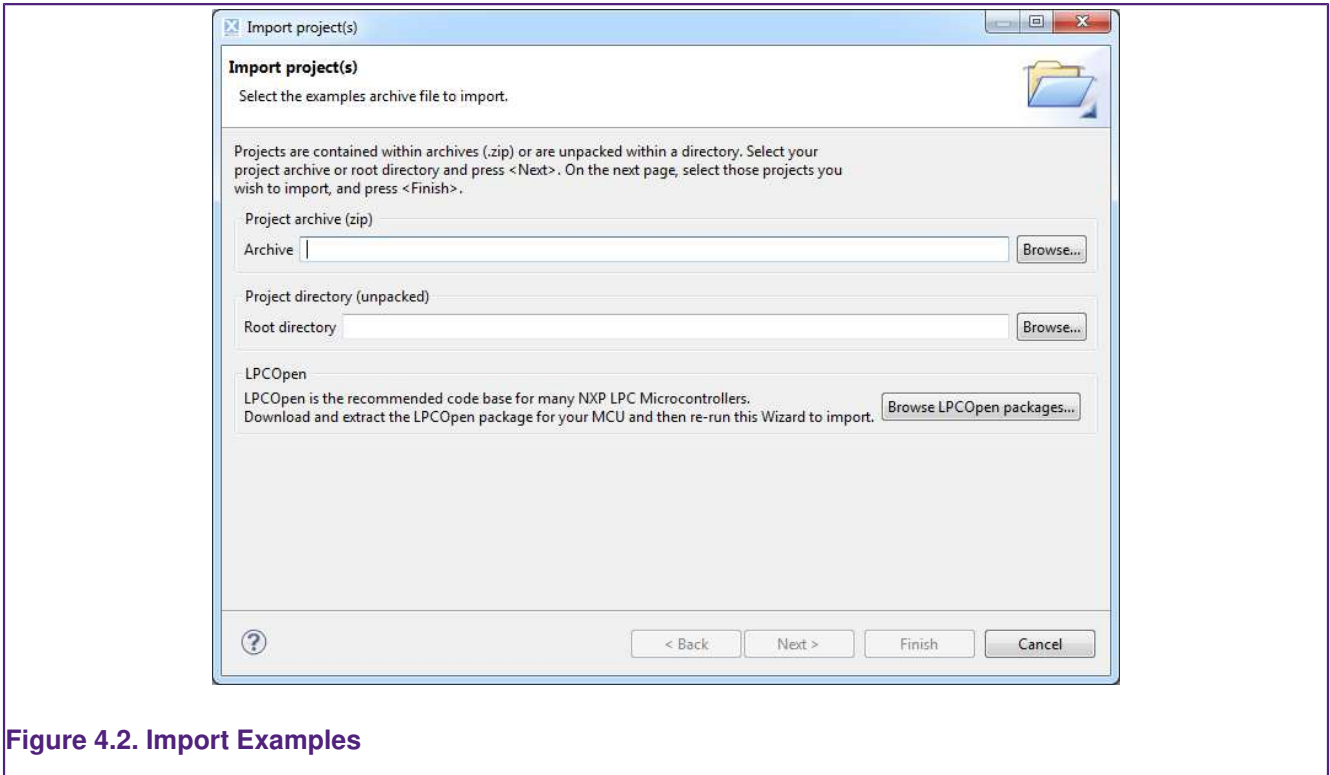


Figure 4.2. Import Examples

To demonstrate how to use the Import Project(s) functionality, we will now import the LPCOpen examples for the LPCXpresso812 development board.

4.2.1 Importing Examples for the LPCXpresso812 Development Board

First of all, assuming that you have not previously downloaded the appropriate LPCOpen package, click on the **Browse LPCOpen Packages**, which will open a web browser window. Click on **Download LPCOpen Packages** link, and then the link to the **LPCOpen v2.xx for LPC8xx family devices**, and then choose the download for the LPCXpresso812 board.

Once the package has downloaded, return to the Import Project(s) dialog and click on the **Browse** button next to **Project archive (zip)** and locate the LPCOpen LPCXpresso812 package archive previously downloaded. Select the archive, click **Open** and then click **Next**. You will then be presented with a list of projects within the archive, as shown in Figure 4.3.