



PetaLinux SDK Migration Guide

v2.1 April 4th, 2011



Table of Contents

Table of Contents	1
About This Guide	2
Related Documents	2
PetaLinux SDK vs PetaLinux-v4.0	2
Overview.....	2
Platform Configuration.....	3
<i>Flat Device Trees</i>	3
New Features in PetaLinux SDK.....	6
Migrating a PetaLinux platform	8
Overview.....	8
Update Your Hardware Project to Support PetaLinux-SDK.....	8
Create A New PetaLinux Platform for the Project.....	10
Merge Customised Kernel Modification from PetaLinux-v0.40 to PetaLinux SDK.....	10
Migrating Custom User Applications to PetaLinux SDK.....	11
Update Your New Platform Default Configuration Files.....	12
Revision History	12

About This Guide

This document discusses some of the important ways that PetaLinux SDK 1.1 differs from earlier releases, and provides advice on how to migrate existing PetaLinux projects to the new SDK tools. Familiarity with PetaLinux-v0.40 (or earlier), and the Xilinx EDK tools, is assumed.

Related Documents

The following other documents exist to help you to make the most of your PetaLinux experience:

- PetaLinux SDK Installation Guide
- Getting Started with PetaLinux SDK
- Guide to QEMU System Simulation in PetaLinux SDK
- Guide to Create and Debug User Applications

You can also find PetaLinux documentation online at

- http://www.petalogix.com/resources/documentation/petalinux_sdk

PetaLinux SDK vs PetaLinux-v4.0

Overview

PetaLinux SDK is based on the tools and techniques used successfully by thousands of developers with earlier releases, and as such the overall flow will be familiar to users transitioning from these older tool versions.

However, PetaLinux SDK is a significant upgrade, with some powerful new features and important differences particularly in how the hardware/software platform synchronisation mechanism works.

Before going into details, the following table gives an overview of the major differences.

	PetaLinux SDK	PetaLinux-v4.0
Linux kernel version	2.6.31 – MMU only PetaLinux SDK only supports MMU-enabled MicroBlaze systems. This gives a much more stable runtime platform and makes software porting much easier.	2.6.20 no-MMU and MMU
GCC toolchain	gcc-4.1.2/gdb-6.5/glibc-2.3.6	gcc-3.4.1/gdb-5.3/glibc-2.3.4/uClibc-0.9.28
Xilinx EDK version	Tested with Xilinx EDK 11.4, with commitment to support each new release from Xilinx	Development ceased with Xilinx 10.3 tools
Platform configuration	Hardware platform is described using 'flat device tree' (DTS) files	Hardware platform is described in Kconfig.auto file.

Platform Configuration

This section gives an overview of the changes in platform configuration between PetaLinux SDK and older releases

Flat Device Trees

The modern MicroBlaze Linux kernel uses a technology called Flat Device Trees, or DTS files, to describe the hardware platform, including address map, peripheral configurations, memory layout and processor features. This technology is also used in the PowerPC architecture, and is gaining wider use in the Linux kernel generally.

One benefit of DTS files is that they are processed dynamically at kernel boot time, which means that a single kernel image can be used to boot against a range of different hardware platforms, simply by modifying the device tree. This is in contrast to earlier MicroBlaze kernels (and PetaLinux releases), where the hardware platform information was compiled into each kernel image.

	PetaLinux SDK	PetaLinux-v0.40
Linux Kernel configuration file	<p>“petalinux-dist/linux-2.6.x/arch/microblaze/boot/<VENDOR>-<PLATFORM>.dts”:</p> <p>Device tree file to describe the hardware platform. <i>Automatically copied into place by petalinux-copy-autoconfig</i></p>	<p>“petalinux-dist/linux-2.6.x/arch/microblaze/platform/<VENDOR>-<PLATFORM>/kconfig.auto”:</p> <p>Hardware platform description file.</p>
U-boot configuration	<p>“petalinux-dist/u-boot/petalogix/microblaze-auto/xparameters.h”:</p> <p>Processor, timer, memory, FLASH and network interface configuration definition file. <i>Automatically copied into place by petalinux-copy-autoconfig</i></p>	Hardware configuration definition file.
	<p>“petalinux-dist/u-boot/petalogix/microblaze-auto/config.mk”:</p> <p>Processor features and u-boot start address in memory specification file. <i>Automatically copied into place by petalinux-copy-autoconfig</i></p>	
Software platform configuration “petalinux-dist/vendors/<VENDOR>/<PLATFORM>” directory.	<p>“<VENDOR>-<PLATFORM>.dts”:</p> <p>Device tree file - to describe the hardware platform.</p>	
	<p>“config.vendor”:</p> <p>User land libraries and applications and system settings configuration file.</p>	<p>“config.vendor”:</p> <p>User land libraries and applications and system settings configuration file.</p>
	<p>“xparameters.h”:</p> <p>Processor, timer, memory, FLASH and network interface configuration definition file for u-boot.</p>	

	PetaLinux SDK	PetaLinux-v0.40
	<p>“<code>config.mk</code>”:</p> <p>Processor features and u-boot start address in memory specification file for u-boot.</p>	

PetaLinux SDK includes a new version of the PetaLinux BSP tools which plugin to the Xilinx EDK tools. These tools automatically generate DTS files for your MicroBlaze system, so platform configuration remains almost entirely automatic.

Another important change is that new platforms no longer require their own subdirectory in `arch/microblaze/platforms` – a single generic platform is sufficient.

The `petalinux-copy-autoconfig` tool has been completely re-written to work with the new platform configuration technology, however the user interface has not changed.

New Features in PetaLinux SDK

Here are some of the important new features of PetaLinux SDK:

- QEMU full system simulator

PetaLinux now includes a QEMU-based MicroBlaze system simulation. You can test your MicroBlaze Linux system with QEMU without hardware, which decouples the software development from the hardware development.

The `petalinux-boot-prebuilt` command can optionally boot any prebuilt reference design in QEMU, instead of via JTAG onto the real hardware.

See the PetaLinux SDK “Guide to QEMU System Simulation” for more details.

- Improved support for custom user applications and modules

The `petalinux-new-app` and `-new-module` commands are still used to create new template applications and kernel modules, however these applications and modules are now integrated into the overall system configuration build procedure.

Custom applications automatically appear in their own submenu in the User/Vendor configuration menu, and can be enabled or disabled individually. Additional menu configuration options can easily be added for each application, and the settings are saved as part of the overall system configuration.

See the PetaLinux SDK “Application Development and Debug Guide” for more details.

- Multiple root filesystem models

PetaLinux SDK supports the automatic generation and configuration for three different root filesystem types

- INITRAMFS (default)
- JFFS2 (Flash based root file system)
- NFS (network file system useful for development and debugging)

The choice of root filesystem type is made in the top level System Configuration menu. PetaLinux SDK will automatically ensure the necessary kernel configuration settings are made, and generate the correct type of filesystem image as required.

- Installable BSPs and reference designs

PetaLinux BSPs are now packaged and distributed separately, and can be installed using the `petalinux-install-bsp` command. BSPs can contain multiple reference designs, for example a 'lite' and 'full' platform targeting the same board, demonstrating both high performance and low resource usage systems for the same board target.

- Automatic kernel configuration checking

PetaLinux SDK includes tools to validate your kernel configuration, maximising the chances of first-boot success in new board bringup. In a software system as large and complex as the Linux kernel it is not possible to check and validate all settings, however common configuration errors that result in a “dead board” are checked and rectified.

- Shortcut commands to configure the Linux kernel and application/vendor settings separately:

`make menuconfig` in the `petalinux-dist` directory allows you to configure both the kernel and user land settings.

`petalinux-config-kernel` opens Linux kernel configuration.

`petalinux-config-apps` opens userland and vendor configuration settings.

`petalinux-platform-config` can be used to save or restore default settings for the currently configured platform.

Migrating a PetaLinux platform

This section outlines the procedure to migrate an existing project into PetaLinux SDK from an older release.

IMPORTANT

Due to the significant differences in kernel version and platform configuration method, it is not recommended to simply copy the vendors./XXX/YYY folder from your PetaLinux 0.40 system into PetaLinux SDK.

Overview

An overview of the platform migration process is as follows

1. Copy your hardware project folder into the hardware/user-platforms directory
2. Update your hardware project files to use new the PetaLinux BSP, and ensure the MicroBlaze CPU is configured correctly
3. Regenerate the libraries of the hardware project
4. Create a new PetaLinux software platform
5. Run petalinux-copy-autoconfig to copy configuration files to new PetaLinux platform
6. Build PetaLinux and test it.
7. Merge your Linux kernel configuration changes to PetaLinux-SDK
8. Migrate any custom applications and modules
9. Build PetaLinux and test it.
10. Update the default configuration files of the PetaLinux platform for your project.

The following sections detail these processes.

Update Your Hardware Project to Support PetaLinux-SDK

PetaLinux-SDK has been tested with Xilinx EDK 11.3 and 11.4. You are strongly recommended version 11.x Xilinx tools. If you project files are generated with an older Xilinx EDK version, you will need to perform a 'revup' step on your hardware project first. Please consult the Xilinx EDK documentation for information on this procedure.

PetaLinux SDK requires the MicroBlaze MMU be enabled. Here are the recommended MMU settings of MicroBlaze, to be applied to the system MHS file:

```
PARAMETER C_USE_MMU = 3
PARAMETER C_MMU_ZONES= 2
PARAMETER C_MMU_DTLB_SIZE = 4
PARAMETER C_MMU_ITLB_SIZE = 2
PARAMETER C_MMU_TLB_ACCESS = 3
```

IMPORTANT

The MicroBlaze parameter “C_AREA_OPTIMIZED” must be set to zero or omitted completely when using the MMU.

It is then necessary to update your EDK project to use the correct version of the PetaLinux BSP generator. This is achieved by modifying the MSS file, which can be done either in the XPS GUI or in a text editor.

1. Open the hardware project with the XPS GUI
2. Double click the “Project Options” from the “Project” tab to open the “Project Options” dialog
3. Make sure the “Project Peripheral Repository Search Path” is set to the “<PETALINUX-SDK_TOP_DIR>/hardware/edk_user_repository”. A relative path like “../hardware/edk_user_repository” is also fine, and is more portable if you later move the SDK tree to a different place in your filesystem.
4. Click “OK” button to accept the change and close the the dialog.
5. Select “Software Platform Settings” from the “Software” menu to open the “Software Platform Settings” dialog.
6. In the “Software Platform” tab, select “petalinux” version “2.00a” as the OS.
7. Open the “OS and Lib Configuration” tab, and then expand “petalinux” configuration. Set the stdout, stdin, main_memory, flash_memory, lmb_memory, iic, sysace, ethernet and timer peripherals as appropriate.
8. Click “OK” button to accept the change and close the dialog.

Next, rebuild the MicroBlaze system libraries by selecting “Clean libraries” and then “Generate Libraries and BSPs” from the “Software” menu on the XPS GUI.

Finally, check the “microblaze_N/libsrc” directory, you should see “petalinux_vX_YY_Z”, “device-tree_vX_YY_Z” and “uboot_vX_YY_Z” there.

Create A New PetaLinux Platform for the Project

Because PetaLinux-SDK uses 2.6.31Linux kernel while PetaLinux-v0.40 uses an much older kernel, the configuration for the old kernel will not work for the new kernel. Thus, we create a new PetaLinux Platform for the Project. Please follow these instructions:

1. Run

```
$ petalinux-new-platform -p <PLATFORM> -v <VENDOR>
```

to create a new platform for the project.

2. Change to your hardware project and run:

```
$ petalinux-copy-autoconfig
```

to copy the hardware configuration files to PetaLinux. This operation will update the:

- “petalinux-dist/linux-2.6.x/arch/microblaze/boot/<VENDOR>-<PLATFORM>.dts”
 - “petalinux-dist/linux-2.6.x/.config”
- for the Kernel;
- “petalinux-dist/u-boot/board/petalogix/microblaze-auto/config.mk”
 - “petalinux-dist/u-boot/board/petalogix/microblaze-auto/xparameters.h”

for the u-boot.

3. Build PetaLinux by running “make” in the petalinux-dist directory and test it.

Merge Customised Kernel Modification from PetaLinux-v0.40 to PetaLinux SDK

If you have previously made kernel source code changes, you will need to manually merge and update these changes into the newer 2.6.31 PetaLinux kernel.

If you have your own device drivers, you will need to port them,. You are strongly recommended to make your driver compatible with the DTS configuration system - You can refer to xemaclite_of_probe() function of “drivers/net/xilinx_emaclite.c” for an example of how this works.

Because many kernel configurations options have changed between the 2.6.20 and 2.6.31 kernel versions, please do not copy the old kernel configuration file.

Instead, run “petalinux-config-kernel” in the PetaLinux-SDK tree to configure the kernel with your project's requirements.

Migrating Custom User Applications to PetaLinux SDK

Your application source code can be expected to port across unmodified, however some changes in the user application build infrastructure will modification of your application Makefile at least. The simplest way to do this is usually to use petalinux-new-app to create the build support, transfer your source code files into the newly created folder, then edit the Makefile to complete your build process.

1. Run

```
$ petalinux-new-app <APPLICATION>
```

to create you user application. The new user application can be found in “<PETALINUX SDK TOP DIR>software/user-apps” directory.

2. Copy the source files from your application in the old PetaLinux-v0.40 to the user application in the PetaLinux SDK.
3. Merge any changes from the toplevel Makefile of your user application into the newly created Makefile in your application in the PetaLinux SDK.
4. Run

```
$ petalinux-config-apps
```

to select your application. Your application should be in “Custom User Applications” sub-menu in the “PetaLinux Configuration” main menu.

5. Run “make” in the petalinux-dist directory to build the system.

Update Your New Platform Default Configuration Files

If you have tested your core platform and everything is working fine, it is a good idea to save the new default configuration as follows:

```
$ petalinux-platform-config --update
```

This operation will update the following files in the “petalinux-

dist/vendors/<VENDOR>/<PLATFORM>” directory:

- “<VENDOR>-<PLATFORM>- .dts” from the one in the “petalinux-dist/linux-2.6.x/arch/microblaze/boot/dts/” directory.

Please note, the petalinux-copy-autoconfig doesn't update the DTS file in the “petalinux-dist/vendors/<VENDOR>/<PLATFORM>” directory.

- “config.linux-2.6.x” from “petalinux-dist/linux-2.6.x/.config”.
- “config.vendor” from “petalinux-dist/config/.config”.
- “xparameters.h” from “petalinux-dist/u-boot/petalogix/microblaze-auto/xparameters.h”.
- “config.mk” from “petalinux-dist/u-boot/petalogix/microblaze-auto/config.mk”.

Revision History

Date	Version	Notes
3/08/2010	.90	Draft version for SDK 1.1 release
3/09/2010	1.00	Initial public release
04/04/11	2.1	Updated for PetaLinux SDK 2.1 release