TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

ZYNQ PetaLinux

2018.07.25 ~ 2018.07.31

강사 : Innova Lee(이상훈)

강사 메일 : gcccompil3r@gmail.com

학생 : 문지희

학생 메일 : mjh8127@naver.com

PetaLinux Tools Documentation

PetaLinux Command Line Reference

UG1157 (v2015.4) December 7, 2015

Introduction

PetaLinux 는 Xilnix AP SoC 및 FPGA 에 임베디드 리눅스를 성공적으로 부팅하는 데 필요한 작업을 자동화하는 개발구축 환경이다. 이 문서는 PetaLinux 환경을 구성하는 도구에 대한 자세한 정보를 설명한다.

PetaLinux Tools Overview

PetaLinux 디자인 플로우를 구성하는 6 개의 툴이 있다

- petalinux-boot
- petalinux-build
- petalinux-config
- petalinux-create
- petalinux-package
- petalinux-util.

대부분 Petalinux 툴은 유연하게 툴에 전달 된 특정 옵션을 사용자에게 동일한 툴에 대한 다른 옵션과 비교해 사용 모델을 제시한다. 이러한 사용 모델을 “workflows”라 한다.

이 문서에서 workflows 의 수식어로 동작하는 명령어를 “option”이라 한다. 옵션에서 사용자 지정 값을 승일할 수 있는 경우 기울임 꼴로 표시된다. 경우에 따라 사용자가 지정한 값을 생략하면 기본 defalt 값이 동작한다. 관련 기본값에 대한 자세한 내용은 표의 “ Defalt Value”를 참조한다.

도구가 올바르게 작동하도록 하는 옵션이 필수적이면 “REQUIRED”로 표시된다. 옵션이 특정 워크플로에 대해 선택적이면 “OPTIONAL”로 표시된다.

Petalinux workflos 마다 사용 예가 제공된다.

Design Flow Overview

일반적으로 Petalinux 툴은 순차 workflows 모델을 따른다. 아래 표는 작업을 완료해야 하는 순서와 해당 작업에 대한 도구와 예제 디자인 워크플로를 제공한다.

Design Flow Step	Tool / Workflow
Hardware Platform Creation	Vivado
Create PetaLinux Project	petalinux-create -t project
Initialize PetaLinux Project	petalinux-config --get-hw-description
Configure System-Level Options	petalinux-config
Create User Components	petalinux-create -t COMPONENT
Configure the Linux Kernel	petalinux-config -c kernel
Configure the Root Filesystem	petalinux-config -c rootfs
Build the System	petalinux-build
Test the System	petalinux-boot --qemu
Deploy the System	petalinux-package

The petalinux-boot Tool

petalinux-boot 도구는 지정된 Linux 시스템 이미지 파일을 부팅합니다. 이 도구는 두 가지 워크 플로를 제공합니다. petalinux-boot --jtag 작업 과정에서 시스템 이미지 파일은 JTAG 케이블 연결을 통해 물리적 보드에 다운로드 및 부팅된다. petalinux-boot --qemu 워크 플로에서 시스템 이미지 파일은 QEMU 소프트웨어 에뮬레이터를 통해 로드되고 부팅됩니다. petalinux-boot 도구에 대해 --jtag 또는 --qemu 옵션을 필수적으로 지정해야 한다.

기본적으로 petalinux-boot 도구는 <PROJECT> / images / linux / 디렉토리에서 파일을 로드한다.

아래 표는 petalinux-boot 워크 플로우에 공통적인 명령 옵션을 설명한다.

Option	Functional Description	Value Range	Default Value
--jtag	REQUIRED. JTAG 워크 플로를 사용해라. QEMU 워크 플로우와 상호 배타적이다.	none	none
--qemu	REQUIRED. QEMU 워크 플로를 사용해라. JTAG 워크 플로우와 상호 배타적이다.	none	none
--prebuilt	OPTIONAL. Boot a prebuilt image.	1 (bitstream/FSBL) 2 (U-Boot) 3 (Linux Kernel)	none
--boot-addr BOOTADDR	OPTIONAL. Boot address.	user-specified	none
-i,--image IMAGEPATH	OPTIONAL. Linux kernel image.	user-specified	zImage(Zynq) Image(Zynq UltraScale+ MPSoC) image.elf(MicroBlaze)
--uboot	OPTIONAL. Specify U-Boot elf binary.	user-specified	u-boot.elf
--kernel	OPTIONAL. Specify Linux kernel binary	user-specified	zImage(Zynq) Image(Zynq UltraScale+ MPSoC) image.elf(MicroBlaze)
-v,--verbose	OPTIONAL. Displays additional output messages.	none	none
-h,--help	OPTIONAL. Displays tool usage information.	none	none

NOTE : --prebuilt 1 은 QEMU 워크 플로에 유효하지 않음

Details for the petalinux-boot --jtag Workflow

이 워크 플로우는 JTAG 연결을 통해 PetaLinux 이미지로 MicroBlaze / Zynq / Zynq UltraScale + MPSoC 시스템을 부팅한다.

다음 표에는 JTAG 부팅 워크플로와 관련된 옵션의 세부 정보가 나와 있다.

Option	Functional Description	Value Range	Default Value
--xsdb-conn COMMAND	OPTIONAL. Customised XSDB connection command to run prior to boot.	user-specified	none
--load-addr LOADADDR	OPTIONAL. Address to load the image.	user-specified	none
--hw_server-url URL	OPTIONAL. URL of the hw_server to connect to.	user-specified	none
--tcl OUTPUTFILE	OPTIONAL. Log JTAG Tcl commands used for boot	user-specified	none
--fpga	OPTIONAL. Program pre-built bitstream. Assumes the presence of the pre-built directory. See petalinux-package for more details.	user-specified	
--bitstream BITSTREAM	OPTIONAL. Program specified bitstream.	user-specified	none

NOTE : 이 워크 플로는 가상 컴퓨터에서 실행될 때 예상대로 작동하지 않을 수 있습니다.

NOTE : --fpga 옵션은 기본적으로 <PROJECT> / pre-built / linux / implementation 에서 download.bit 를 찾습니다.

Example Usage of the petalinux-boot --jtag Workflow

다음 예제는 petalinux-boot --jtag 워크 플로우의 올바른 사용법을 보여준다.

- 사전 빌드 된 비트 스트림 (및 Zynq / Zynq UltraScale + MPSoC 용 FSBL)을 JTAG 을 통해 실제 보드에 다운로드하고 부팅하십시오.

  $ petalinux-boot --jtag --prebuilt 1

- 사전 빌드 된 U-Boot elf 를 JTAG 을 통해 실제 보드에 다운로드하고 부팅하십시오.

  $ petalinux-boot --jtag --prebuilt 2

- JTAG 을 통해 빌드 된 커널 이미지를 다운로드하여 물리적 보드로 부팅합니다.

  $ petalinux-boot --jtag --kernel

  MicroBlaze 의 경우 image.elf 를 다운로드하고, Zynq 의 경우 zImage 및 system.dtb 를 다운로드한다. Zynq UltraScale + MPSoC 의 경우 ATF, Image 및 system.dtb 를 다운로드한다.

Details for the petalinux-boot --qemu Workflow

이 워크 플로우는 QEMU 에뮬레이터를 통해 PetaLinux 이미지로 MicroBlaze / Zynq / Zynq UltraScale + MPSoC 시스템을 부팅한다. 많은 QEMU 옵션은 제대로 작동하려면 수퍼유저(root) 액세스가 필요하다다. --root 옵션은 ROOT MODE 를 활성화하고 사용자에게 sudo 자격 증명을 요구한다.

다음 표에는 QEMU 부팅 워크플로와 관련된 옵션의 세부 정보가 나와 있다.

Option	Functional Description	Value Range	Default Value
--dtb DTBFILE	OPTIONAL. Use a specified device tree file	user-specified	system.dtb

Example Usage of the petalinux-boot --qemu Workflow

다음 예제는 petalinux-boot --qemu 워크 플로우의 올바른 사용법을 보여준다.

- QEMU 를 통해 빌드 된 U-Boot elf 를 로드하고 부팅한다.

  $ petalinux-boot --qemu --prebuilt 2

- 루트 모드에서 QEMU 를 통해 빌드 됐던 U-Boot elf 를 로드하고 부팅한다.

  $ petalinux-boot --qemu --root --prebuilt 2

The petalinux-build Tool

petalinux-build 도구는 전체 임베디드 Linux 시스템 또는 지정된 Linux 시스템 구성 요소를 빌드한다. 이 도구는 단일 워크플로를 제공하지만 petalinux-build -c 및 petalinux-build -x 옵션을 통해 작업의 세부 사항을 지정할 수 있다.

아래 표는 petalinux-build 툴에의 옵션에 대한 표이다.

Option	Functional Description	Value Range	Default Value
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path.	user-specified	current directory
-c, --component COMPONENT	OPTIONAL. Build specified component. "all" is the implied default.	all bootloader kernel u-boot rootfs	all
-x, --execute MAKE-TARGET	OPTIONAL. Execute specified GNU Makefile command.	all bootloader kernel u-boot rootfs	all
--makeenv ENVARS	OPTIONAL. Additional GNU make environment variables.	none	none
-v,--verbose	OPTIONAL. Displays additional output messages.	none	none
-h,--help	OPTIONAL. Displays tool usage information.	none	none

Details for the -c Option

-c 옵션은 임베디드 시스템의 지정된 구성 요소를 빌드한다. 구성 요소를 지정하지 않으면 petalinux-build 도구가 전체 프로젝트에서 작동한다. 루트 파일 시스템을 위해 사용자가 만든 구성 요소

이름으로 해당 구성 요소를 타겟팅 하여 만들 수 있습니다 (예 : -c rootfs / <COMPONENT-NAME>).

아래 목록은 워크 플로우를 대상으로 사용할 수 있는 구성 요소를 요약 한 것이다.

- all - 프로젝트의 모든 구성 요소를 빌드하고 <PROJECT> / images /에 복사한다. 이것은 옵션이 없는 기본 동작이다.

- bootloader - 부트 로더 elf 이미지만 빌드하고 <PROJECT> / images /에 복사한다. Zynq 및 Zynq UltraScale + MPSoC 장치의 경우 FSBL, MicroBlaze CPU 의 경우 FS-BOOT 이다.
- device-tree - 장치 트리 DTB 파일만 빌드하고 <PROJECT> / images /에 복사한다.
- ATF - ATF 이미지만 만들고 <PROJECT> / images /에 복사한다.
- kernel - Linux 커널 이미지만 빌드하고 <PROJECT> / images /에 복사한다.
- rootfs - 루트 파일 시스템 이미지만 빌드하고 <PROJECT> / images /에 복사한다.
- u-boot - U-Boot elf 이미지만 빌드하고 <PROJECT> / images /에 복사한다.

Details for the -x Option

-x 옵션을 사용하면 petalinux-build 도구에 표준 GNU Makefile 옵션을 지정하여 지정된 구성 요소의 조작 방법을 제어 할 수 있다.

아래 목록은 이 옵션과 함께 사용할 수있는 Makefile 명령을 요약 한 것이다.

- clean - 대상 구성 요소의 빌드 데이터를 정리한다. -c 옵션과 함께 사용해야한다. -c all 과 함께 유효하지 않는다.
- distclean - 빌드 영역을 정리한다. 그러면 <PROJECT> / build / 디렉토리가 제거된다.
- all - 전체 PetaLinux 프로젝트를 만들고 출력을 <PROJECT> / images /로 복사한다. 이것은 -c all 과 동일하다.
- build - 전체 PetaLinux 프로젝트를 빌드하지만 <PROJECT> / images /로 복사하지 않는다.
- install - 필요한 경우 device-tree, U-Boot 및 Linux 커널 바이너리를 <PROJECT> / build /의 준비된 위치에 복사한다.
- package - 빌드 영역의 현재 내용에서 FIT 이미지 image.ub 를 생성하고 <PROJECT> / images /로 복사한다.

Example Usage of the petalinux-build Workflow

다음 예제는 petalinux-build 워크 플로우의 올바른 사용법을 보여준다.

- BSP 또는 revision control 용으로 보관할 PetaLinux 프로젝트의 빌드 영역을 지워라. 이 예제는 프로젝트의 images 디렉토리를 유지한다.

  $ petalinux-build -x distclean

- PetaLinux 프로젝트의 U-Boot 구성 요소에서 모든 빌드 담보를 지 웁니다.

  $ petalinux-build -c u-boot -x clean

- 빌드 영역의 현재 내용에서 업데이트 된 FIT 이미지를 만듭니다.

  $ petalinux-build -x package

- 전체 PetaLinux 프로젝트를 빌드하십시오.

  $ petalinux-build -c all

The petalinux-config Tool

petalinux-config 툴을 사용하여 지정된 프로젝트를 사용자 정의 할 수 있다. 이 도구는 두 개의 별도 워크플로를 제공한다. **petalinux-config --get-hw-description** 워크 플로우에서 프로젝트는 지정된 하드웨어 구성을 반영하도록 초기화되거나 업데이트된다. **petalinux-config -c COMPONENT** 워크 플로우에서 지정된 구성 요소는 menuconfig 인터페이스를 사용하여 사용자 정의된다.

아래 표는 petalinux-config 도구에 사용할 수있는 옵션을 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
--get-hw-description PATH	REQUIRED. Initialize or update the hardware configuration for the PetaLinux project. Mutually exclusive with -c.	user-specified	none
-c,--component COMPONENT	REQUIRED. Configured the specified system component. Mutually exclusive with --get-hw-description.	none kernel rootfs	none
--searchpath	OPTIONAL. Modify the search environment used by PetaLinux.	--prepend --append --replace --print --delete	none
--defconfig DEFCONFIG	OPTIONAL. Linux kernel only. Use the specified defconfig file to initialize the Linux kernel configuration.	user-specified	none
--oldconfig	OPTIONAL. Restore the configuration for the specified component to a prior version. Without -c, restores system-level configuration.	none	none
-v,--verbose	OPTIONAL. Displays additional output messages.	none	none
-h,--help	OPTIONAL. Displays tool usage information.	none	none

Details for the --searchpath Option

petalinux-config --searchpath 옵션을 사용하면 PetaLinux 가 참조 구성 요소에 사용하는 SEARCHPATH 환경을 제어 할 수 있다. 이 옵션은 아래에 설명 된 수식어 중 하나와 함께 사용해야 하고 여러 수식어를 동시에 사용할 수 있다. 아래 표는 PetaLinux 에서 사용되는 SEARCHPATH 에 사용 가능한 수식어와 기능을 자세히 설명한다. <PETALINUX-INSTALL-DIR> / components 는 항상 가장 낮은 우선 순위인 반면 <PROJECT> / components 경로는 항상 최우선 순위이다.

- --prepend PATH - 프로젝트 외부 검색 경로에 PATH 를 추가한다.
- --append PATH - PATH 를 추가하여 외부 검색 경로를 프로젝트에 추가한다.
- --replace PATH - 프로젝트 외부 검색 경로 (있는 경우)를 PATH 로 바꾼다.
- --print - 전체 프로젝트 검색 경로를 보여준다.
- --delete - 프로젝트 외부 검색 경로 삭제

Details for the petalinux-config --get-hw-description Workflow

petalinux-config -get-hw-description 워크플로우를 사용하여 지정된 Vivado 하드웨어 프로젝트의 하드웨어 관련 정보로 PetaLinux 프로젝트를 초기화하거나 업데이트 할 수 있다. 이 프로세스의 영향을 받는 구성 요소에는 FSBL 구성, U-Boot 옵션, Linux 커널 옵션 및 Linux 장치 트리 구성이 포함될 수 있다. 이 워크플로우는 PetaLinux 프로젝트의 실수나 의도하지 않은 하드웨어 구성 변경을 방지하기 위해 신중하게 사용해야한다. 이 워크 플로와 함께 사용되는 경로는 HDF 파일 자체의 전체 경로가 아닌 HDF 파일이 들어있는 디렉터리이다. 이 전체 옵션은 HDF 파일이 들어있는 디렉토리에서 실행하는 경우 생략 할 수 있다.

Example Usage of the petalinux-config --get-hw-description Workflow

다음 예제는 petalinux-config --get-hw-description 워크 플로의 올바른 사용법을 보여준다.

- 외부에서 받은 장치 트리 생성 도구를 사용하여 PetaLinux 프로젝트를 초기화해라.

  $ petalinux-config --get-hw-description <PATH-TO-HDF> --searchpath --prepend <PATH-TO-HDF>

- HDF 가 포함된 디렉토리에서 PetaLinux 프로젝트를 초기화한다.

  $ petalinux-config --get-hw-description -p <PATH-TO-PETALINUX-PROJECT>

- 중립적인 위치에서 PetaLinux 프로젝트를 초기화한다.

  $ petalinux-config --get-hw-descriptioni <PATH-TO-HDF> -p <PATH-TO-PETALINUXPROJECT>

Details for the petalinux-config -c COMPONENT Workflows

petalinux-config -c COMPONENT 워크플로를 사용하면 표준 menuconfig 인터페이스를 사용하여 임베디드 Linux 시스템을 제어 할 수 있다. petalinux-config 가 다른 옵션없이 실행될 때, system-level 또는 "generic"menuconfig 를 시작한다. 이 인터페이스를 통해 사용자는 원하는 부팅 장치 또는 기본 호스트 이름과 같은 시스템에 대한 메타 데이터와 같은 정보를 지정할 수 있다. **petalinux-config -c kernel** 과 **petalinux-config -c rootfs** 워크 플로는 각각 Linux 커널과 루트 파일 시스템을 사용자 정의하기위한 menuconfig 인터페이스를 시작한다.

--oldconfig 옵션을 사용하면 이전 구성을 복원 할 수 있다. 이전 구성은 지정된 구성 요소가 들어있는 디렉토리 내에 CONFIG.old 파일로 저장되어있다.

NOTE : Xilnix 기술 지원은 일반 Linux 커널 구성보다는 Linux 커널에서 Xilnix 특정 옵션 및 / 또는 사용자 정의를 지원한다.

Example Usage of the petalinux-config -c COMPONENT Workflow

다음 예제는 petalinux-config -c COMPONENT 워크 플로우의 올바른 사용법을 보여준다.

- 시스템 수준 구성의 menuconfig 를 시작한다.

  $ petalinux-config

- 루트 파일 시스템에 대한 이전 구성을 로드한다.

  $ petalinux-config -c rootfs --oldconfig

- 특정 기본 구성으로 Linux 커널 구성을 로드한다.

  $ petalinux-config -c kernel --defconfig xilinx_zynq_base_trd_defconfig

The petalinux-create Tool

petalinux-create 도구는 PetaLinux 프로젝트의 일부인 객체를 생성한다. 이 도구는 두 개의 별도 워크 플로우를 제공한다. **petalinux-create -t** 프로젝트 워크플로에서 이 도구는 새로운 PetaLinux 프로젝트 디렉토리 구조를 만든다. **petalinux-create COMPONENT** 워크플로에서 툴은 지정된 프로젝트 내에 구성 요소를 작성한다. 이러한 워크 플로우는 petalinux-create -t project 또는 petalinux-create -t COMPONENT 로 각각 실행됩니다.

아래 표는 모든 petalinux-create 워크 플로우에 공통적인 명령 옵션을 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
-t,--type TYPE	REQUIRED. Specify the TYPE of object to create.	project apps libs modules generic	none
-n,--name NAME	REQUIRED. Create object with the specified NAME. This is optional when creating a project from a BSP source.	user-specified	none
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path	user-specified	current directory
--force	OPTIONAL. Overwrite existing files on disk.	none	none
-h,--help	OPTIONAL. Display usage information.	none	none

Details for the petalinux-create -t project Workflow

petalinux-create -t 프로젝트 워크 플로우는 설정한 이름으로 지정된 위치에 새로운 PetaLinux 프로젝트를 생성한다. 기본적으로 이 워크 플로우는 현재 작업 디렉토리에 새로운 Zynq AP SoC 기반 프로젝트를 작성한다. 이러한 기본값을 덮어 쓰려면 아래의 옵션을 사용해야한다. 기본적으로 이 명령으로 생성 된 디렉토리는 최소한의 구조이며 petalinux-config --get-hw-description 워크플로를 사용하여 초기화 될 때까지 전체 시스템을 빌드하는 데는 유용하지 않다. 소스로 BSP 파일을 사용하여 생성 된 프로젝트는 즉시 빌드하기에 적합하다.

다음 표는 프로젝트를 만들때 특별히 사용되는 옵션에 대해 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
--template TEMPLATE	OPTIONAL. Assume specified CPU architecture. Required when "-s|-source" is not provided.	zynq microblaze	zynq
-s,--source SOURCE	OPTIONAL. Create project based on specified BSP file. SOURCE is the full path on disk to the BSP file.	user-specified	none
--out OUTPUTDIR	OPTIONAL. Create a project in the specified directory.	none	current directory

Example Usage of the petalinux-create -t project Workflow

다음 예제는 petalinux-create -t 프로젝트 워크 플로우의 올바른 사용법을 보여줍니다.

- 참조 BSP 파일에서 새 프로젝트를 만든다.

  $ petalinux-create -t project -s <PATH-TO-BSP>

- MicroBlaze 템플릿을 기반으로 새 프로젝트를 생성힌다.

  $ petalinux-create -t project -n <NAME> --template microblaze

- 중립적 인 위치에서 새 프로젝트를 만든다.

  $ petalinux-create -t project -n <NAME> --template zynq -out <PATH-TO-CREATE>

Details for the petalinux-create -t COMPONENT Workflows

petalinux-create-COMPONENT 워크플로를 사용하여 사용자는 지정된 PetaLinux 프로젝트 내에서 다양한 구성 요소를 작성할 수 있다. 이 구성 요소는 petalinux-config -c rootfs 워크 플로우를 사용하여 토글링하여 선택적으로 최종 시스템에 포함되거나 제외 할 수 있다. petalinux-create -t generic 또는 petalinux-create -t 모듈 워크플로에는 구성 요소별 옵션이 없다.

petalinux-create -t apps 워크 플로우는 사용자가 작성하는 동안 어플리케이션 컴포넌트가 초기화되는 방법을 사용자 정의 할 수 있게한다. 다음 표는 PetaLinux 프로젝트 내에서 어플리케이션을 생성 할 때 공통적으로 나타나는 옵션을 설명한다.

Option	Functional Description	Value Range	Default Value
-s,--source SOURCE	OPTIONAL. Create the component from pre-existing content on disk. Valid formats are .tar.gz, .tar.bz2, .tar, .zip, and source directory (uncompressed).	user-specified	none
--template TEMPLATE	OPTIONAL. Create the component using a pre-defined project template.	c c++ autoconfig install	c
--enable	OPTIONAL. Upon creating the component, automatically enable it in the project's root filesystem. Else, enable using the petalinux-config -c rootfs	none	disabled

petalinux-create -t libs 워크플로를 사용하면 사용자가 작성 중에 라이브러리 구성 요소를 초기화하는 방법을 사용자 정의 할 수 있다. 이 옵션을 사용하면 생성 된 라이브러리가 컴파일되어 최종 루트 파일 시스템에 제대로 포함되도록 할 수 있다. 다음 표는 PetaLinux 프로젝트 내의 라이브러리 구성 요소와 관련된 옵션을 설명한다.

Option	Functional Description	Value Range	Default Value
--priority	OPTIONAL. Specify the build priority for the library. Denoted by an integer suffix on the Kconfig file in the library directory. e.g., Kconfig.n	1 - 11	7

Example Usage of the petalinux-create -t COMPONENT Workflow

다음 예제는 petalinux-create -t COMPONENT 워크 플로우의 올바른 사용법을 보여준다.

- 루트 파일 시스템에서 활성화 된 응용 프로그램 구성 요소를 만든다.

  $ petalinux-create -t apps -n --enable

- 컴파일 우선 순위가 1 인 새 라이브러리 구성 요소를 만든다.

  $ petalinux-create -t libs -n --priority 1

- 새로운 설치 전용 응용 프로그램 구성 요소를 만든다. 이 흐름에서 아무 것도 컴파일되지 않는다.

  $ petalinux-create -t apps -n --template install

The petalinux-package Tool

petalinux- 패키지 도구는 PetaLinux 프로젝트를 배포에 적합한 형식으로 패키지한다. 이 도구는 대상 패키지 형식에 따라 작동 방식이 다른 여러 가지 워크플로를 제공한다. 지원되는 형식/워크플로는 부팅, bsp, 펌웨어 및 사전 빌드다.

petalinux-package 도구는 패키지 유형 이름을 사용하여 **petalinux-package --PACKAGETYPE** 형식의 특정 워크 플로우를 지정하여 실행된다.

- 부트 패키지 유형은 대상 장치를 부팅 할 수 있는 파일 (.BIN 또는 .MCS)을 만든다.
- bsp 패키지 유형은 대상 PetaLinux 프로젝트의 전체 내용을 포함하는 .bsp 파일을 생성한다.
- 펌웨어 패키지 유형은 이미 구성된 보드의 PROM 장치를 업데이트하는 데 필요한 파일을 포함하는 .tar.gz 파일을 만든다. 이 패키지 형식은 업그레이드 펌웨어 PetaLinux 데모 응용 프로그램에서만 호환된다.

- The pre-built package type creates a new directory within the target PetaLinux project called "pre-built" and contains pre-built content that is useful for booting directly /on a physical board.

기본적으로 petalinux-package 도구는 <PROJECT> / images / linux / 디렉토리에서 기본 파일을 로드한다.

아래 표는 모든 petalinux-package 워크 플로우에 공통적인 옵션을 설명한다.

Option	Functional Description	Value Range	Default Value
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path.	user-specified	current directory
-h,--help	OPTIONAL. Display usage information.	none	none

Details for the petalinux-package --boot Workflow

petalinux-package --boot 워크 플로우는 Zynq 제품군 디바이스 (Zynq 및 Zynq UltraScale + MPSoC 포함) 또는 MicroBlaze 기반 FPGA 디자인과 직접 사용할 수있는 부팅 가능한 이미지를 생성한다. Zynq 제품군 디바이스의 경우 부팅 가능한 포맷은 SD 카드에서 부팅 할 수있는 BOOT.BIN 이다. MicroBlaze 기반 설계의 경우 기본 형식은 Vivado 또는 다른 PROM 프로그래머를 통한 프로그래밍에 적합한 MCS PROM 파일이다.

Zynq AP SoC 장치의 경우이 워크 플로는 Xilinx SDK 와 함께 제공되는 bootgen 유틸리티를 둘러싼 wrapper 이다. MicroBlaze 기반 FPGA 디자인의 경우이 워크 플로우는 해당 Vivado Tcl 명령에 대한 wrapper 이며 MCS 형식의 프로그래밍 파일을 생성한다. 이 MCS 파일은 대상 보드에 직접 프로그래밍 한 다음 부팅 할 수 있다.

아래 표는 petalinuxpackage - boot 워크 플로우로 부팅 가능한 이미지를 생성 할 때 유효한 옵션에 대해 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
--format FORMAT	REQUIRED. Image file format to generate.	BIN MCS	BIN
--fsbl FSBL	REQUIRED. Path on disk to FSBL elf binary.	user-specified	zynq_fsbl.elf(Zynq) fs-boot.elf(MicroBlaze)
--force	OPTIONAL. Overwrite existing files on disk.	none	none
--fpga BITSTREAM	OPTIONAL. Path on disk to bitstream file.	user-specified	none
--atf ATF-IMG	OPTIONAL. Path on disk to ARM trusted firmware elf binary.	user-specified	bl31.elf
--uboot UBOOT-IMG	OPTIONAL. Path on disk to U-Boot elf binary.	user-specified	u-boot.elf
--kernel KERNEL-IMG	OPTIONAL. Path on disk to Linux Kernel image. Used to include FIT image in BOOT.BIN.	user-specified	image.ub
--add DATAFILE	OPTIONAL. Path on disk to arbitrary data to include.	user-specified	none
--offset OFFSET	OPTIONAL. Offset at which to load the prior data file.	user-specified	none
--bmm BMMFILE	OPTIONAL. Valid for MicroBlaze only.	user-specified	BMM in directory with FPGA bitstream
--flash-size SIZE	OPTIONAL. Must be a power-of-2. Valid for MicroBlaze only. Not needed for parallel flash types.	user-specified	16MB (SPI)
--flash-intf INTERFACE	OPTIONAL. Valid for MicroBlaze only.	SERIALx1 SPIx1 SPIx2 SPIx4 BPIx8 BPIx16 SMAPx8 SMAPx16 SMAPx32	auto-detect
-o, --output OUTPUTFILE	OPTIONAL. Path on disk to write output image.	user-specified	current directory
--cpu DESTINATE CPU	OPTIONAL. ZynqMP only.	a53-0	none

	destination CPU of the data file		
--file-attribute DATA File ATTR	OPTIONAL. Zynq/ZynqMP only. data file file-attribute	user-specified	none
--bif-attribute-value VALUE	OPTIONAL. Zynq/ZynqMP only. value of the attribute specified by -attribute argument	user-specified	none
--bif BIF FILE	OPTIONAL. Zynq/ZynqMP only. BIF file. It overrides all other settings: -fsbl, -fpga, -u-boot, -add, -fsblconfig, -file-attribute, -bif-attribute, -bif-attribute-value.	user-specified	none
--boot-device BOOT-DEV O	OPTIONAL. Zynq/ZynqMP only. sd, flash default will be the one selected from system select menu of boot image settings	user-specified	none

Example Usage of the petalinux-package --boot Workflow

다음 예제는 petalinux-package - boot 워크플로우의 올바른 사용법을 보여준다.

• Zynq 제품군 디바이스 (Zynq 및 Zynq UltraScale + MPSoC 포함) nq 및 Zynq UltraScale + MPSoC 용 BOOT.BIN 파일을 작성한다.

$ petalinux-package --boot --format BIN --fsbl --u-boot -o 〈PATH-TO-OUTPUT〉

• PL 비트 스트림과 FIT 이미지가 포함 된 Zynq 제품군 장치 용 BOOT.BIN 파일을 만든다.

$ petalinux-package --boot --format BIN --fsbl --u-boot ₩ --fpga 〈PATH-To-BITSTREAM〉 --kernel -o 〈PATH-TO-OUTPUT〉

• PMU 펌웨어가 포함 된 Zynq UltraScale + MPSoC 장치 용 BOOT.BIN 파일을 만든다.

$ petalinux-package --boot --u-boot --kernel ₩ --add 〈PATH-TO-PMU-FW-ELF〉 -file-attribute destination_device=pmufw

Details for the petalinux-package --bsp Workflow

petalinux-package --bsp workflow 는 지정된 PetaLinux 프로젝트 디렉토리의 모든 내용을 제공된 파일 이름으로 BSP 파일로 컴파일한다. 이 .bsp 파일은 배포되어 나중에 새로운 PetaLinux 프로젝트를 생성하기 위한 소스로 사용될 수 있다. 이 워크플로는 일반적으로 다른 사용자에게 배포 할 수 있는 프로젝트 이미지를 제작하는 마지막 단계로 사용됩니다. PetaLinux 를위한 모든 Xilnix 레퍼런스 BSP 는이 워크 플로우를 사용하여 패키징된다.

아래 테이블은 petalinuxpackage --bsp 워크 플로우로 PetaLinux BSP 파일을 패키징 할 때 유효한 옵션에 대해 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
-o, --output BSPNAME	REQUIRED. Path on disk to store the BSP file. File name will be of the form BSPNAME.bsp.	user-specified	current directory
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path. In the BSP context, multiple project areas can be referenced and included in the output BSP file.	user-specified	current directory
--force	OPTIONAL. Overwrite existing files on disk.	none	none
--clean	OPTIONAL. Clean the hardware implementation results to reduce package size	none	none
--hwsource HWPROJECT	OPTIONAL. Path to a Vivado project to include in the BSP file.	none	none
--no-extern	OPTIONAL. Exclude components external to the project referenced using the --searchpath option. This may prevent the BSP from building for other users.	none	none
--no-local	OPTIONAL. Exclude components referenced in the local PetaLinux project. This may prevent the BSP from building for other users.	none	none

Example Usage of the petalinux-package --bsp Workflow

다음 예제는 petalinux-package --bsp 워크 플로우의 올바른 사용법을 보여준다.

- 프로젝트를 지우고 BSP 이미지를 만든다.

  $ petalinux-package --bsp --clean -o <PATH-TO-BSP>

- 참조 하드웨어 정의가 포함된 BSP 이미지를 작성한다.

  $ petalinux-package --bsp --hwsource <PATH-TO-HW-EXPORT> -o <PATH-TO-BSP>

- 중립적인 위치에서 BSP 이미지를 만든다.

  $ petalinux-package --bsp -p <PATH-TO-PROJECT> -o <PATH-TO-BSP>

Details for the petalinux-package --firmware Workflow

petalinux-package --firmware 워크플로는 지정된 PetaLinux 프로젝트를 기반으로 한 펌웨어 업데이트 패키지를 생성한다. 펌웨어 패키지를 통해 사용자는 배포 된 시스템의 구성 요소를 선택적으로 업데이트 할 수 있다. 이 패키지에는 U-Boot, Linux 커널, Linux device-tree 또는 Linux 루트 파일 시스템과 같은 구성 요소가 포함될 수 있다.

아래 표는 petalinux-package --firmware 워크 플로우로 PetaLinux 펌웨어 이미지를 패키징 할 때 유효한 옵션에 대해 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
-o, --output PACKAGENAME	OPTIONAL. Full path and name on disk to store the firmware image. Default location is current directory	user-specified	firmware.tar.gz
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path.	user-specified	current directory
--linux UBIMAGE	OPTIONAL. Update the Linux kernel partition with the specified UBIMAGE.	none	image.ub
--dtb DTBFILE	OPTIONAL. Update the device tree DTB partition with the specified DTBFILE.	none	system.dtb
--fpga BITSTREAM	OPTIONAL. Update the FPGA bitstream partition with the specified BITSTREAM.	none	none
--u-boot UBOOT-S	OPTIONAL. Update the U-Boot binary partition with the specified UBOOT-S binary.	none	u-boot-s.bin
--bootbin BOOT.BIN	OPTIONAL. Update the boot partition with the specified BOOT.BIN binary. Zynq only.	none	BOOT.bin
--jffs2 JFFS2IMAGE	OPTIONAL. Update the user's JFFS2 partition with the specified JFFS2IMAGE image.	none	jffs2.img
-a,--add dev:file	OPTIONAL. Update the flash partition named dev with the file specified by file. This option can be repeated multiple times.	user-specified	none
--flash FLASHTYPE	OPTIONAL. Specify the type of flash device with which the image is compatible.	spi parallel	parallel
--little-endian	OPTIONAL. Specify that the target system is little endian. AXI systems are little-endian.	none	none
--big-endian	OPTIONAL. Specify that the target system is big endian. PLB systems are big-endian.	none	none
--data-width SIZE	OPTIONAL. Specify the bit width of the data bus for the target flash device.	8 16 32	none
--product STRING	OPTIONAL. Specify a product compatible string used to validate firmware image.	user-specified	none
--pre SCRIPT	OPTIONAL. Specify a SCRIPT that should be run on the target platform prior to	user-specified	none

	updating the flash partitions.		
-v,--verbose	OPTIONAL. Displays additional output messages.	none	none

--image, --dtb, --uboot 및 --jffs2 옵션을 사용하면 --add 옵션의 partition : file 구문을 사용하여 기본 파일 이름과 파티션을 무시할 수 있다.

Example Usage of the petalinux-package --firmware Workflow

다음 예제는 petalinux-package --firmware 워크 플로우의 올바른 사용법을 보여준다.

- FIT 이미지 (image.ub)를 safe-image 라는 플래시 파티션에 설치한다.

  $ petalinux-package --firmware -a / dev / flash / safe-image : <PATH-TO-FIT-IMAGE>

- MicroBlaze 용 비트스트림, U-Boot 및 Linux 커널로 펌웨어 이미지 패키징한다.

  $ petalinux-package --firmware --fpga <BITSTREAM> --u-boot --linux -o <FILETO-CREATE>

Details for the petalinux-package --prebuilt Workflow

petalinux-package --prebuit 워크플로는 지정된 PetaLinux 프로젝트의 디렉토리 계층 내에 "pre-built"라는 새로운 디렉토리를 생성한다다. 이 디렉토리에는 프로젝트를 완전히 재구성하지 않고 즉시 보드를 부팅하는 데 필요한 파일이 들어있다. 이 워크 플로우는 나중에 petalinux-package --bsp 워크 플로우를 사용하여 배포 할 PetaLinux BSP 파일을 작성하는 사용자를 위한 것이다. 모든 Xilnix 레퍼런스 PetaLinux BSP 에는 빌드 된 디렉토리가 있다.

아래 표는 petalinuxpackage --prebuilt 워크 플로우를 사용하여 프로젝트에 빌드 된 데이터를 포함시킬 때 유효한 옵션에 대해 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path.	user-specified	current directory
--force	OPTIONAL. Overwrite existing files on disk.	none	none
--clean	OPTIONAL. Remove all files from the /prebuilt directory.	none	none
--fpga BITSTREAM	OPTIONAL. Include the BITSTREAM file in the prebuilt directory.	user-specified	none
-a,--add src:dest	OPTIONAL. Add the file/directory specifed by src to the directory specifed by by dest in the pre-built directory	user-specified	none

Example Usage of the petalinux-package --prebuilt Workflow

다음 예제는 petalinux-package --prebuilt 워크 플로우의 올바른 사용법을 보여준다.

- 빌드된 영역에 특정 비트 스트림을 포함시킨다.

  $ petalinux-package --prebuilt --fpga <BITSTREAM>

- 빌드된 영역에 특정 데이터 파일을 포함시킨다.

  $ petalinux-package --prebuilt -a <APP>:images/<APP>

The petalinux-util Tool

petalinux-util 툴은 다른 PetaLinux 워크플로에 다양한 서비스를 지원한다. 필요한 기능에 따라 여러 워크 플로를 제공한다.

Details for the petalinux-util --gdb Workflow

petalinux-util --gdb 워크 플로우는 표준 GNU GDB 디버거의 wrapper 이며 현재 터미널에서 GDB 디버거를 시작한다. 터미널 프롬프트에서 petalinux-util --gdb --help 를 실행하면 사용할 수있 는 자세한 GDB 옵션을 제공한다.

Example Usage of the petalinux-util --gdb Workflow

다음 예제는 petalinux-util --gdb 워크 플로의 올바른 사용법을 보여준다.

- GNU GDB debugger 를 실행한다.

  $ petalinux-util --gdb

Details for the petalinux-util --xsdb-connect Workflow

이 워크 플로우는 QEMU 에 XSDB 연결을 제공한다. 이것은 Zynq 및 Zynq UltraScale + MPSoC 전용이다. 아래 표는 petalinux-util --jtag-logbuf 작업 흐름을 사용할 때 유효한 옵션을 설명한다.

Option	Functional Description	Value Range	Default Value
--xsdb-connect HOST:PORT	REQUIRED. Host and the port XSDB should connect to. This should be the host and port that QEMU has opened for GDB connections. It can be found in the QEMU command line arguements from: --gdb tcp: :	user-specified	none

Details for the petalinux-util --jtag-logbuf Workflow

이 워크플로는 JTAG 을 통해 Linux 커널 이미지를 부팅 할 때 발생하는 Linux 커널 printk 출력 버퍼를 기록한다. 이 워크플로는 검토 및 디버그를 위해 Linux 커널을 디버깅 하기위한 것이다. 이 워크플로는 Linux 커널이 serial 터미널을 통해 출력하지 않을 때 유용하다. JTAG 을 통해 시스템을 부팅하는 방법에 대한 자세한 내용은 petalinux-boot --jtag 작업 과정을 참조한다. MicroBlaze 의 경우 이미지는 <PROJECT> /image/linux/image.elf 이고, ARM 의 경우 이미지는 <PROJECT> / image / linux / vmlinux 이다.

아래 표는 petalinux-util --jtag-logbuf 작업 흐름을 사용할 때 유효한 옵션을 자세히 설명한다.

Option	Functional Description	Value Range	Default Value
-i,--image IMAGEPATH	REQUIRED. Linux kernel ELF image.	user-specified	none
--hw_server-url URL	OPTIONAL. URL of the hw_server to connect to.	user-specified	none
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path.	user-specified	current directory
--noless	OPTIONAL. Do not pipe output to the less command.	none	none
-v,--verbose	OPTIONAL. Displays additional output messages.	none	none
-h,--help	OPTIONAL. Displays tool usage information.	none	none

Example Usage of the petalinux-util --jtag-logbuf Workflow

다음 예제는 petalinux-util --jtag-logbuf 워크플로우의 올바른 사용법을 보여준다.

- 특정 Linux 커널 이미지를 실행한다.

  $ petalinux-util --jtag-logbuf -i <PATH-TO-IMAGE>

- 중립 위치에서 JTAG logge 를 시작한다. 이 워크 플로우는 Zynq 디바이스 전용이다.

  $ petalinux-util --jtag-logbuf -i <PATH-TO-IMAGE> -p <PATH-TO-PROJECT>

Details for the petalinux-util --update-sdcard Workflow

이 워크플로는 SD 카드에 있는 실제 파티션에 루트 파일 시스템을 로드하는 작업을 자동화한다. 이는 Zynq AP SoC 장치에만 유효히다. 이 워크플로는 VFAT 파티션 (BOOT.BIN 의 경우), Linux ext 파티션 (루트 파일 시스템의 경우) 또는 둘 모두 로드 할 수 있다. **petalinux-util --update-sdcard** 워크플로는 BOOT.BIN 파일을 <PROJECT> / images / linux 에서 boot :로 지정된 파티션으로 복사한다. 이 파일이 이 위치에 없으면 실패한다.

: rootfs 요소가 생략되면 도구는 <PROJECT> /images/linux/image.ub 파일을 VFAT 파티션에 복사한다. : rootfs 구성 요소가 있으면 도구는 <PROJECT> / build / linux / rootfs / targetroot /의 내용을 : rootfs 에 지정된 위치로 복사한다. boot 및 rootfs 의 대상 디렉토리가 적절한 액세스 권한으로 마운트되지 않은 경우 성공적으로 완료하려면 수퍼 유저 액세스가 필요하다. 아래 표는 petalinux-util-update-sdcard 워크플로를 사용할 때 유효한 옵션을 설명한다.

Option	Functional Description	Value Range	Default Value
-d,--dir boot:rootfs	REQUIRED. Copy files to the SD card.	user-specified	none
-p,--project PROJECT	OPTIONAL. PetaLinux project directory path.	user-specified	current directory
-h,--help	OPTIONAL. Display usage information.	none	none

Example Usage of the petalinux-util --update-sdcard Workflow

다음 예제는 petalinux-util --update-sdcard 워크플로의 올바른 사용법을 보여준다.

- Zynq AP SoC 장치의 BOOT.BIN 및 image.ub 파일을 SD 카드의 VFAT 파티션에 복사한다.

권한이 있는 사용자 또는 지정된 경로에 대한 액세스 권한이 필요.

```
$ petalinux-util --update-sdcard --dir <VFAT-PATH>
```

- BOOT.BIN (및 image.ub)과 루트 파일 시스템의 내용을 SD 카드의 ext 파일 시스템에 복사한다

권한이 있는 사용자 또는 지정된 경로에 대한 액세스 권한이 필요.

```
$ petalinux-util --update-sdcard --dir <VFAT-PATH>:<EXT-PATH>
```

Details for the petalinux-util --webtalk Workflow

이 워크 플로는 Xilinx WebTalk 기능을 끄고 킨다. Xilinx WebTalk 는 다양한 PetaLinux 툴에 대한 사용 데이터를 Xilinx 에 제공한다. 이 기능을 사용하려면 인터넷에 연결되어 있어야한다.

Option	Functional Description	Value Range	Default Value
--webtalk	REQUIRED. Toggle WebTalk.	on off	on
-h,--help	OPTIONAL. Display usage nformation.	none	none

Example Usage of the petalinux-util --webtalk Workflow

다음 예제는 petalinux-util --webtalk 워크 플로우의 올바른 사용법을 보여준다.

- Toggle the WebTalk feature off.

  $ petalinux-util --webtalk off

- Toggle the WebTalk feature on.

  $ petalinux-util --webtalk on