

PDK/Maitaining Custom Changes Over TI Software Release

This guide will give you some tips on how you can maintain your custom changes over the TI Software release. These changes can be specific to the custom board, usecase specific updates etc. The key things we cover as part of this tutorial are

1. How to maintain custom changes over a TI software release
2. If in case you need to migrate to a different version of the same TI Software
3. How to take in Bug fixes from the latest and greatest TI Software release without really having to migrate to the latest version. (Note: This is available from the PDK 01.09.XX.XX version)

Contents

Introduction

One Time Setup

Migrating to a New Release

Applying Bug Fixes to older releases

Introduction

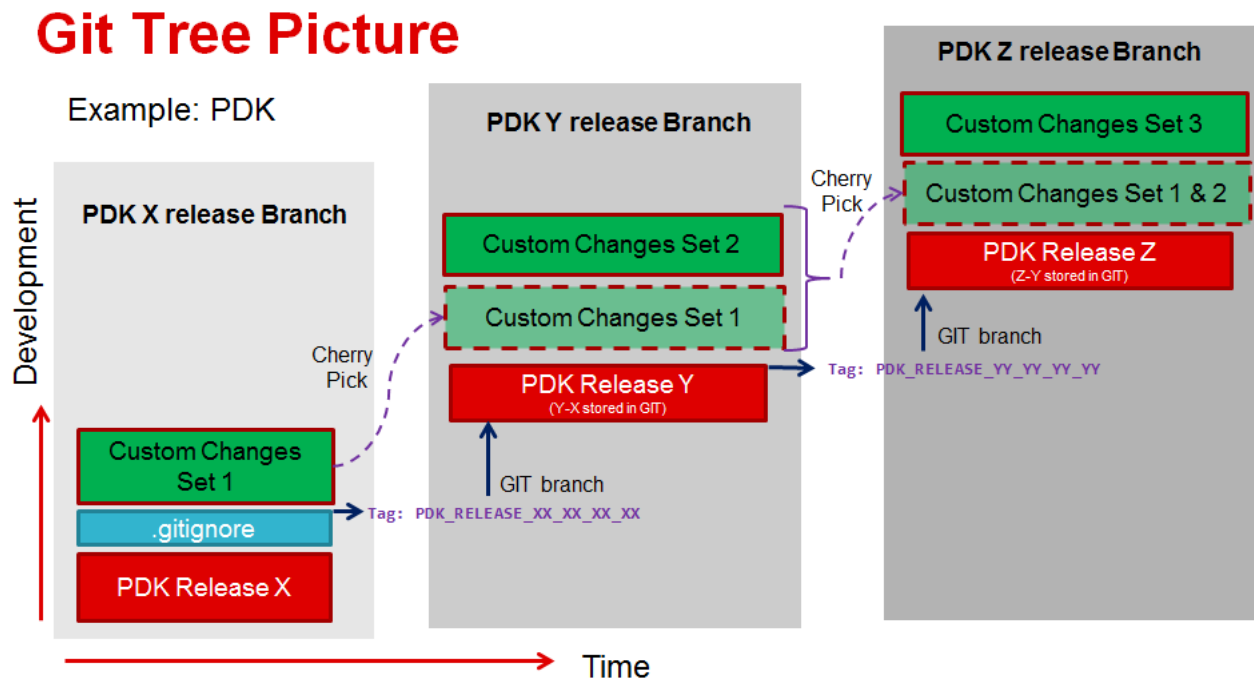
The basic idea is to use GIT (<https://git-scm.com/downloads>) or any other version control mechanism to maintain changes on TI Software. The techniques mentioned below are applicable for PDK and Processor SDK. The top-level steps include:

- Download the first base version of TI Software
- Use Instructions in [One Time Setup](#) to setup GIT on the TI Software release from which the development will start. (Lets say version XX.XX.XX.XX)
- Make changes to this version. Commit the changes to maintain change history with respect to the TI Software version.
- For any new release create a different branch from the base version TI Software commit.
- Cherry-pick custom changes from the previous branch. Resolve merge conflicts if any.

The advantage of following this methodology is:

- You have a full history of custom changes.
- You can easily see changes between TI Software versions.
- You can easily switch between TI Software versions and custom change versions.
- It is easy to pick up bug fixes from TI.

The following picture helps understand this idea better. The Custom changes highlighted below are the changes that you make over the TI releases.



One Time Setup

The first step is to download the TI Software release version that you would like to start with from <http://www.ti.com/tool/processor-sdk-vision> or <http://www.ti.com/tool/processor-sdk-radar>. Let this be version XX.XX.

With PDK as the example next follow below steps:

```
> mv pdk_xx_xx_xx_xx/ pdk_
> cd pdk_
> git init
> git add -A
> git commit -a -m "PDK Release xx_xx_xx_xx"
```

From PDK release 01.09.XX.XX you would find the git ignore files as a part of the folders. If you do have the gitignore files you can jump to **STEP 4**. Note: git ignore files are needed to ignore the generated files like library files which should not be committed to version by mistake.

STEP 1: If you are on an older PDK release, you can copy the gitignore files from [File:Pdk gitignore.zip](#) to the PDK release folder.

STEP 2: You would find below in the created PDK repository:

```
> git status
On branch master
Untracked files:
(use "git add <file>..." to include in what will be committed)
packages/ti/.gitignore
packages/ti/boot/sbl_auto/.gitignore
packages/ti/csl/.gitignore
packages/ti/diag/.gitignore
packages/ti/drv/bsp_lld/.gitignore
packages/ti/drv/fw_l3l4/.gitignore
packages/ti/drv/ipc_lite/.gitignore
packages/ti/drv/pm/.gitignore
packages/ti/drv/stw_lld/.gitignore
packages/ti/drv/vps/.gitignore
```


STEP 3: Once the .gitignore files have been added you can commit these changes as below:

```
> git add -A
> git commit -a -m "Adding .gitignore files for libraries and binary"
> git log --pretty=oneline
<Commit ID> Adding .gitignore files for libraries and binary
<Commit ID> PDK Release xx_xx_xx_xx
```

STEP 4: Apply a tag to keep a record of base TI version. For example:

```
> git tag -a PDK_RELEASE_XX_XX_XX_XX -m "Release version pdk_XX_XX_XX_XX"
> git tag
PDK_RELEASE_XX_XX_XX_XX
> git branch -m master PDK_RELEASE_XX_XX_XX_XX_BRANCH
```

Migrating to a New Release

 **Note:** It is **not mandatory** to migrate and follow every TI software release for your development. You can stick to a given release and continue your work. In case, you decide to migrate to the latest TI software release version this section would give some guidelines

Lets say you have made some changes over PDK in the release PDK_RELEASE_XX_XX_XX_XX and your git log is as below:

```
> git log --pretty=oneline
<Commit ID Last Change> Test Change 2
<Commit ID First Change> Test Change 1
9899209a3c7319b1a35b04bfff4844549e872bef Adding the .gitignore
b7c1fa3f355c8fa37f69d64db68203a4843a65b1 Release 01_08_01_06
```

```
> git tag
PDK_RELEASE_XX_XX_XX_XX
```

```
> git branch
PDK_RELEASE_XX_XX_XX_XX_BRANCH
```

In order to migrate the latest PDK release from this point,

STEP 1: Create a new branch for holding the new PDK

```
> git checkout -b PDK_RELEASE_YY_YY_YY_YY_BRANCH PDK_RELEASE_XX_XX_XX_XX
> git branch
PDK_RELEASE_XX_XX_XX_XX_BRANCH
* PDK_RELEASE_YY_YY_YY_YY_BRANCH
```

STEP 2: Copy and replace the all the files from the new release to the new branch. Note: Some files would have been removed from latest release. So delete the entire directory (except the .git folder) and then copy the new package.

STEP 3: Add all files to the git. This will make a baseline of the new release.

```
> git add -A
> git commit -a -m "PDK Release yy_yy_yy_yy"
> git tag -a PDK_RELEASE_YY_YY_YY_YY -m "Release version pdk_YY_YY_YY_YY"
> git tag
PDK_RELEASE_XX_XX_XX_XX
PDK_RELEASE_YY_YY_YY_YY
```

STEP 4: Copy the changes from the previous branch:

```
> git cherry-pick <Commit ID First Change>^.. <Commit ID Last Change>
[PDK_RELEASE_YY_YY_YY_YY_BRANCH <ID>] Test Change 1
1 file changed, 2 insertions(+)
[PDK_RELEASE_YY_YY_YY_YY_BRANCH <ID>] Test Change 2
1 file changed, 2 insertions(+)
```

STEP 5:Merge conflicts if any should be fixed while cherry-picking.

Applying Bug Fixes to older releases

From PDK release version 01.09.XX.XX, TI will provide the .patch file along with PDK under docs/patches folder. Each bug will be in a folder for example PDK-1777 with multiple patch files (for fixes spanning across multiple components) and a readme.txt file. This will give the following details

- 1. The folders which are impacted by the bug.
- 2. The files which are impacted by the bug.
- 3. The order of patches that needs to be applied.

Keystone=

1. switchcategory:MultiCore=

■ For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum

■ For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum

Please post only comments related to the article **PDK/Maitaining Custom Changes Over TI Software Release** here.

■ For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum

■ For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum

Please post only comments related to the article **PDK/Maitaining Custom Changes Over TI Software Release** here.

C2000=For technical support on the C2000 please post your questions on The C2000 Forum. Please post only comments about the article **PDK/Maitaining Custom Changes Over TI Software Release** here.


DaVinci=For technical support on the DaVinciplease post your questions on The DaVinci Forum. Please post only comments about the article **PDK/Maitaining Custom Changes Over TI Software Release** here.

MSP430=For technical support on MSP430 please post your questions on The MSP430 Forum. Please post only comments about the article **PDK/Maitaining Custom Changes Over TI Software Release** here.

OMAP35x=For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article **PDK/Maitaining Custom Changes Over TI Software Release** here.

OMAPL1=For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article **PDK/Maitaining Custom Changes Over TI Software Release** here.

MAVRK=For technical support on MAVRK please post your questions on The MAVRK Toolbox Forum. Please post only comments about the article **PDK/Maitaining Custom Changes Over TI Software Release** here.



Amplifiers & Linear Audio

Broadband RF/IF & Digital Radio

Clocks & Timers

Data Converters

DLP & MEMS High-Reliability Interface

Logic

Power Management

Processors

■ ARM Processors

■ Digital Signal Processors (DSP)

■ Microcontrollers (MCU)

■ OMAP Applications Processors

Switches & Multiplexers

Temperature Sensors & Control ICs

Wireless Connectivity

Retrieved from "https://processors.wiki.ti.com/index.php?title=PDK/Maitaining_Custom_Changes_Over_TI_Software_Release&oldid=233548"

This page was last edited on 9 March 2018, at 22:04.

Content is available under [Creative Commons Attribution-ShareAlike](#) unless otherwise noted.

https://processors.wiki.ti.com/index.php/PDK/Maitaining_Custom_Changes_Over_TI_Software_Release

3/3