SDK Build Guide for

Raspberry Pi Using Cross Compiler

V1.0

**Table of Contents**

Contents

[1. Purpose 4](#_Toc468975885)

[2. References 4](#_Toc468975886)

[3. Assumptions 4](#_Toc468975887)

[4. RPI Development System 4](#_Toc468975888)

[5. Building ZLS38100 SDK 4](#_Toc468975889)

[Appendix 6](#_Toc468975898)

[A. Getting Sources and Toolchain 6](#_Toc468975899)

[Getting Toolchain 6](#_Toc468975900)

[Getting Sources 6](#_Toc468975901)

|  |  |
| --- | --- |
| **Document Owner** | Shally Verma |
| **Version** | 1.0 |
| **Date** | Dec 2016 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Change History** | | | |
| **Date** | **Version** | **Changed by** | **Change Description** |
|  |  |  |  |

**Abbreviations**:

|  |  |
| --- | --- |
| SDK | Software Development Kit |
| RPI | Raspberry Pi |
| Distro | Raspberry Pi bootable image. Also referred as “Raspberry Pi Distribution Package”. Current linux based distribution package is termed as Raspbian. Throughout this document terms Distro, Raspbian, Raspberry pi Distribution image will be used interchangeably to refer to linux based bootable image. |

**Table 1 Abbreviations**



# Purpose

This document specifically covers SDK build instructions for Raspberry Pi, if using cross compiler.

It should further be proceed with reference to ZLS38100\_SDK\_Build\_Guide\_Raspberry.pdf for SDK Load and Run instructions.

The SDK support all Linux and non-Linux platforms. However, the build instructions provided in this document is primarily for Linux-based Raspberry Pi distribution packages and compatible for other Linux-based host platforms.

# References

[1] <https://www.raspberrypi.org/documentation/>

[2] ZLS38100\_SDK\_Build\_Guide\_Raspberry.pdf

[3] https://www.raspberrypi.org/documentation/linux/kernel/building.md

# Assumptions

This document assumes that user is familiar to boot Raspberry pi using linux based distribution image and has up and running Raspberry Pi platform along with file sharing mechanism which enables user to copy libraries from Cross compiler machine to Pi.

# RPI Development System

Latest tested with following configuration

|  |  |
| --- | --- |
| Raspberry Distro | 2016-11-25-raspbian-jessie-lite |
| Raspberry Pi Board | Raspberry Pi 3 Model B Rev 1.2 |
| Toolchain | gcc-linaro-arm-linux-gnueabihf-raspbian (for cross compilation on 32-bit system) |

**Table 2 Development Platform**

# Building ZLS38100 SDK

* + 1. Compiling SDK Natively on Raspberry Pi

Please refer to adjoining ZLS38100\_SDK\_Build\_Guide\_Raspberry.pdf

* + 1. Compiling SDK for Raspberry Pi using Cross Compiler

In the following description, $(ROOT) refers to path where sdk is installed.

This section assumes user has raspbian distro compatible linux kernel sources and toolchain. If not, user can refer to [Appendix A](#_Appendix) to see help on retrieving linux sources and a toolchain.

Once you have kernel sources and toolchain, change following in Makefile.globals:

|  |  |
| --- | --- |
| PLATFORM | raspberry. should always be set to raspberry |
| RPI\_MODEL | 1 – raspberry pi 1 model B  2,3 – raspberry pi 2 and 3 respectively |
| KSRC | Path where kernel source directory is present. Example, if kernel source present at $(ROOT)/sdk/platform/raspberry/kernel/linux , then set KSRC=$(ROOT)/platform/raspberry/kernel/linux |
| TOOLSPATH | PATH to toolchain. Example, if toolchain present at $(ROOT)/platform/raspberry/tools, then set TOOLCHAIN=$(ROOT)/platform/raspberry/kernel/tools |
| CROSS-COMPILER | arm-linux-gnueabhihf- (This is very important) |
| ARCH | ARM |

* + - 1. Give command make hbilnx

This will build /platform, /hbi and /hbilnx components and save output hbi.ko in /libs directory

This will also build device tree overlays files .dtbo . User need to manually copy them to raspberry pi /boot/overlays directory to apply them during boot.

* + - 1. Give command make apps

This will build hbi\_test by default

OR

make apps TEST\_XXX=1

where xxx= test options ex. TEST\_DOA to build direction-of-arrival test for ZL38051

Test binaries are built inside ./apps directory

Refer to ZLS38100\_Apps.pdf inside /doc folder for more information.

* + - 1. make swig – This will build python c-wrapper module to test Microsemi vproc sdk ASR feature. Also can be tested through apps/swig/python/hbi\_test.py
      2. After successful build, copy libs over to Raspberry Pi board using some file sharing mechanism and following load and run instructions as in ZLS38100\_SDK\_Build\_Guide\_Raspberry.pdf
      3. Individual component make rules are also supported. Some of them listed below:
         1. make platform – builds complete /platform directory
         2. make ssl – build only SSL driver as ssl.o
         3. make hbi – build only hbi driver as hbi.o
         4. make hbilnx – build kernel loadable module hbi.ko
         5. make codec – build sound driver inside /platform directory

User can refer to master Makefile available at Root Directory for other supported target types.

Every rule is followed by clean with naming convention target\_clean. Example, to clean hbi.o use command make hbi\_clean. To clean hbi.ko, use make hbilnx\_clean.



# Appendix

# Getting Sources and Toolchain

For successful cross compiling, you should have compatible raspbian image and sources and toolchain. This section provides help information on various ways you can create compatible build environment.

# Getting Sources

* 1. Get the latest kernel sources and firmware
     1. If it is not a user requirement to stick to specific raspbian distro, it is recommended to start with latest raspbian distro. You can download latest raspbian from raspberry.org/download Or upgrade your existing raspbian distro to latest as described on the page here <https://www.raspberrypi.org/documentation/raspbian/updating.md> OR

Do

sudo apt-get install rpi-update

sudo rpi-update

Note: the above:

- pertains to a pi using the Raspian image.

- you need to reboot your pi after everything is complete

* + 1. Once distro is updated to latest one, you can get latest sources and toolchain through links provided in <https://www.raspberrypi.org/documentation/linux/kernel/building.md> unders section **CROSS COMPILING**. You can execute git checkout command directly on your cross-compiler machine. Please note you only need to get sources and tool chain and no other build instructions are needed.
  1. Getting specific kernel version

If you like a specific distro or have existing running pi and do not want to upgrade same, you need to find compatible sources from git repo. Following could be of help:

* + 1. Using rpi-source script

SDK provides a script rpi-source at $(ROOT)/sdk/platform/raspberry/tools. This script works only on raspberry pi and helpful if you have up and running raspberry. Copy over this script on your pi machine and execute it. It will download complete compatible kernel tar ball for you. A downloaded linux tarball then can be copied over to cross compiler. OR

* + 1. You can simply do an update to a specific kernel version by giving specific commit-id to rpi-update command. You can get commit-id from Pi firmware repository <https://github.com/Hexxeh/rpi-firmware/commits/master> and find the kernel you want to install.

commit-id is string next to **“<>”** symbol

Example, if you select Bump to 4.4.36, then you go to link https://github.com/Hexxeh/rpi-firmware/tree/b2de18efa673f3337dae0defd394306f2191437c

In example above, it would be

sudo rpi-update b2de18e

To get sources, look at file **git\_hash** it will show you commit-id string of a kernel (ideally would be same as above c6d86f7aa554854b04614ebb4d394766081fb41f) and then use git checkout using this commit-id(this can be executed over cross compiler machine)

git checkout c6d86f7

* + 1. Alternatively, you can also retrieve firmware and kernel sources directly through git

<https://github.com/raspberrypi/firmware/commits/master>

Once you decide which kernel and firmware you want, select the button "**<>**" to browse the code.

Example if you select kernel 4.4.36, once you click on that <> sign it will bring you to that link <https://github.com/raspberrypi/firmware/tree/6ce98b985998d7dfb779eff11bda6a2e297ca516>

That hash number 6ce98b985998d7dfb779eff11bda6a2e297ca516 is what you need to update your pi to that latest firmware

git checkout 6ce98b9

and the respective kernel sources **extra/git\_hash** shows c6d86f7aa554854b04614ebb4d394766081fb41f. get the kernel sources using

git checkout c6d86f

* + 1. More, You can refer to link

<https://www.raspberrypi.org/forums/viewtopic.php?f=66&t=82811&p=726802#p726802>

# Getting Toolchain

Checkout from git from link here:

<https://github.com/raspberrypi/tools/tree/master/arm-bcm2708>