Contents

Software Overview

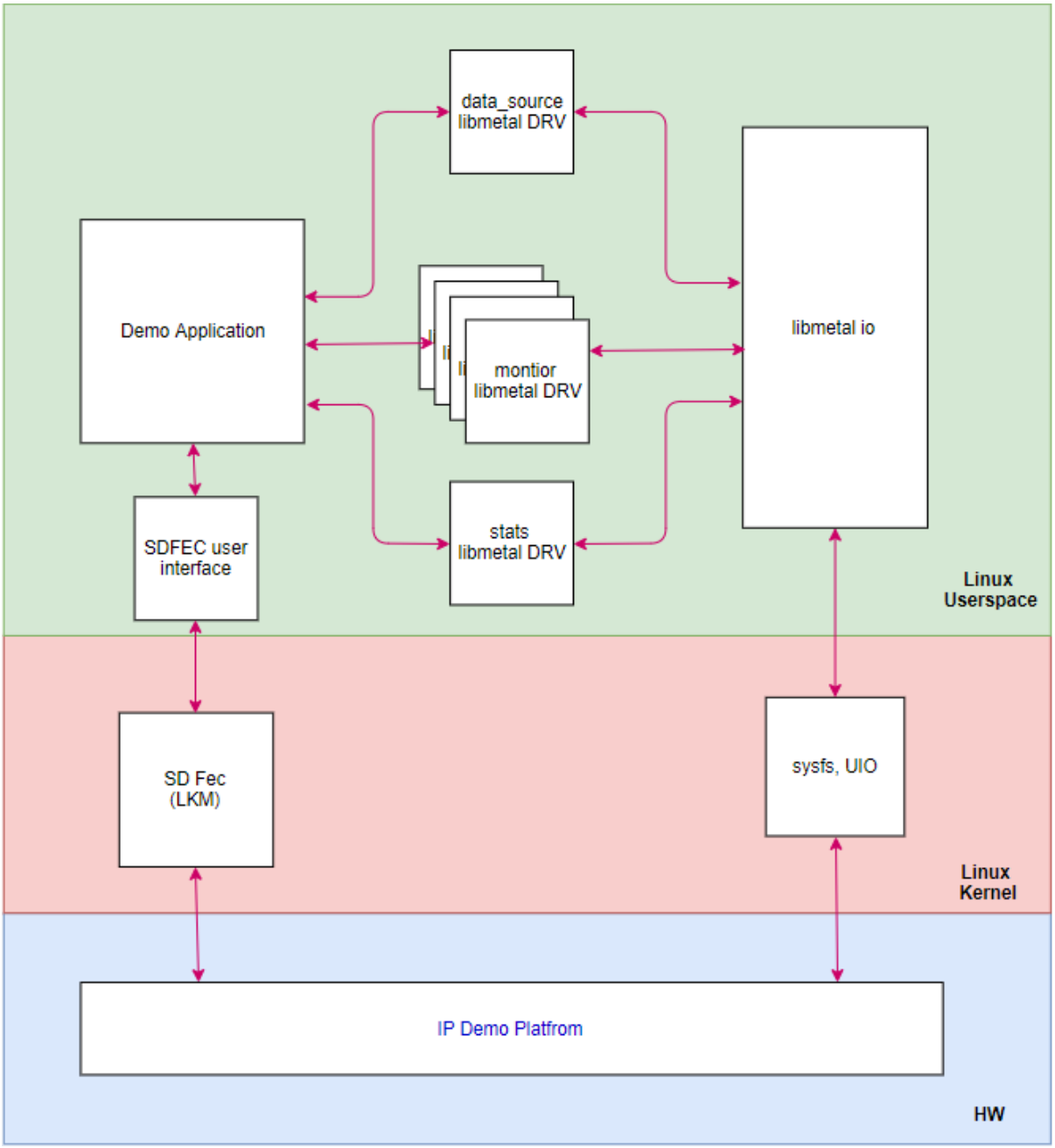

Figure 1 - Linux Demo Environment

The SD-FEC Linux example application (illustrated in Figure 1) is used to setup and control the BER test hardware and demonstrates how to configure the SD-FEC IP instance(s) using the SD-FEC Linux Kernel Module [Ref 3].

The purpose of the demo application is to show the user an example usage of the SD-FEC Linux Kernel Module to load LDPC codes. The sdfec-demo accesses the Linux Kernel module via the SD-FEC user interface library. The Linux userspace drivers are also used in the demo application to control the data source, monitor and statistics blocks (i.e. the BER test hardware). The Linux userspace drivers apply common user APIs, provided by OpenAMP "libmetal" library, to access the devices and request memory across Linux userspace [Ref 4].

A visual guidance to the user is provided by two LEDs through the GPIO sysfs interface.

The remainder of this document provides information on how to load and run the SD-FEC Linux Demo using PetaLinux Tools. Prior to starting the user must ensure that the PetaLinux Tools are installed, refer to Chapter 2: Setting up your Environment [Ref 2].

- Project Setup & BSP installation
- Building PetaLinux Image
- Loading Image to Board

The final section shows how to execute the demo application and an application flow

Project Setup & BSP Installation

The PetaLinux project structure is created using a ZCU111 BSP. The sdfec-demo application and sdfecusrintf library are added to the project as recipes using the build script provided.

Steps:

1. First Ensure that PetaLinux Tools are installed, refer to Chapter 2: Setting up your Environment [Ref 2].
2. Setup Petalinux project for working: use the following command

```
$ create_sdfec_demo.sh setup -b <path-to-bsp>
```

The user must specify the location of the ZCU111 BSP.
Where the setup implements the following

- Creates a petalinux project based on the given bsp
- Adds the demo application and the SD-FEC user interface library to the PetaLinux project
- Edit PetaLinux configuration to ensure the SD-FEC application is included in the build

Building PetaLinux Image

Steps:

1. A full build is required before loading onto the ZCU111. To build Petalinux system image, run the following command:

```
$ create_sdfec_demo.sh build
```

This step generates a device tree DTB file, a first stage bootloader (if selected), U-Boot, the Linux kernel, and a root filesystem image. Finally, it generates the necessary boot images.

2. The compilation progress shows on the console. Wait until the compilation finishes. The full compilation log "build.log" is stored in the build subdirectory of your PetaLinux project.

3. Check that the Petalinux image is built.

```
plnx_sdfec-zcu111-zu28-es1-2018.2>>ls ./images/linux/
bl31.bin    rootfs.cpio            rootfs.ext3.bz2  rootfs.manifest
system.dtb        zynqmp_fsbl.elf
bl31.elf    rootfs.cpio.bz2        rootfs.ext4      rootfs.tar.bz2
System.map.linux
Image       rootfs.cpio.gz         rootfs.ext4.gz   rootfs.tar.gz
u-boot.bin
```

```
image.ub    rootfs.cpio.gz.u-boot  rootfs.its
rootfs.testdata.json  u-boot.elf
rootfs.bin  rootfs.ext3           rootfs.jffs2     system.bit
vmlinux
```

Image Loading on Board

The resultant image can be loaded using methods described in Chapter 5: Booting and Packaging [Ref 2].

For a more detailed guide on PetaLinux tools setup and installation please refer to PetaLinux Tools Documentation – Reference Guide [Ref 2].

Application Summary

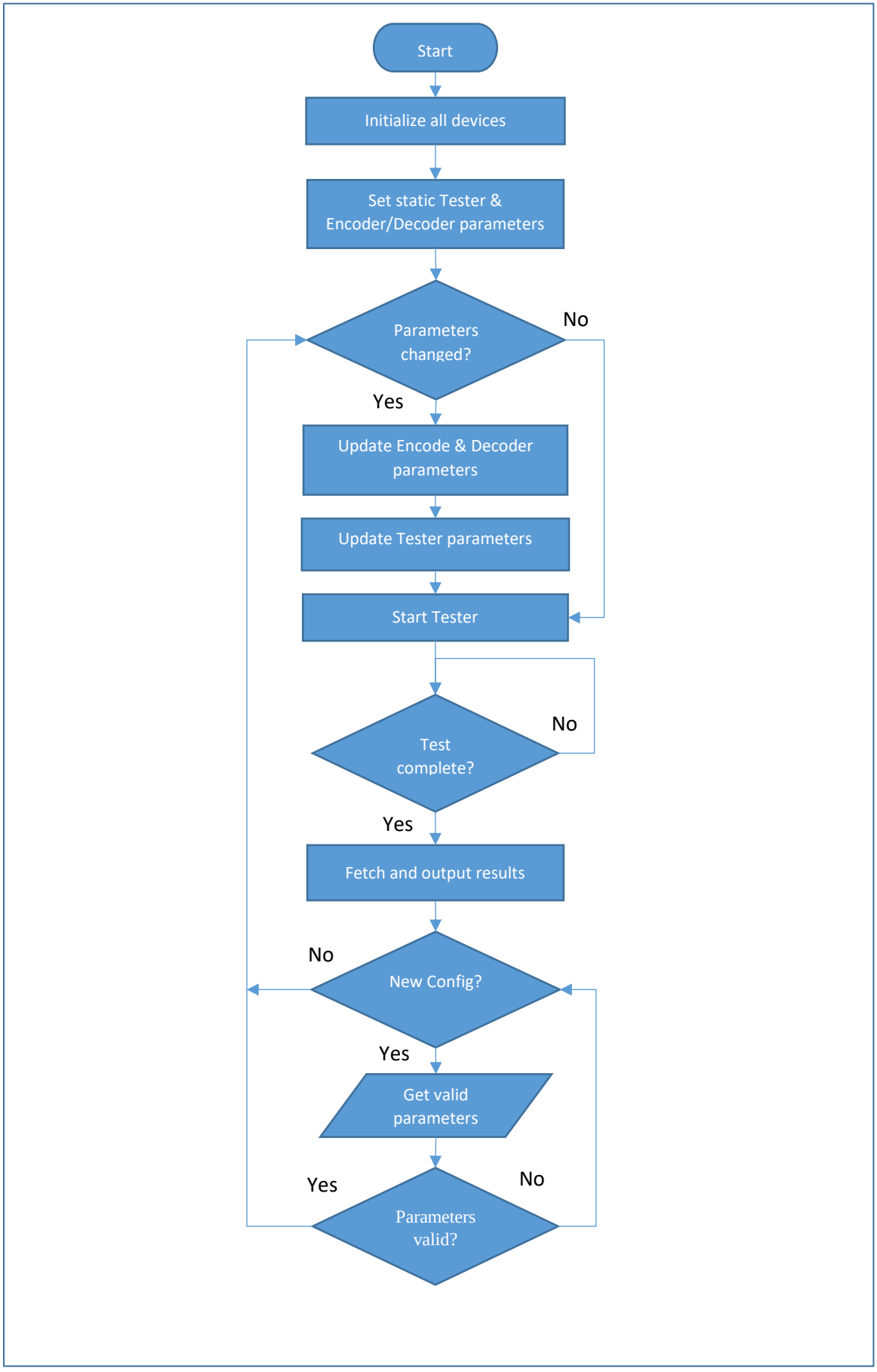

Figure 2 – Application Flow Chart

Figure 2 illustrates the application behaviour. The user can select BER Test Hardware configurations and three different LDPC codes (short, medium and long DOCSIS parameters). The results of the application vary depending on the configuration specified.

The interactive functionality enables the design parameters, defined in Table 6-1 and Table 6-2 [Ref 1] to be updated via the console. The demo application supports Non-5G Control LDPC only. A menu system is implemented where the parameters can be updated and repeated tests run. Input validation is also provided to prevent improperly formed data from entering the system. To exit the demo application, the user must terminate the process on the console (ctrl-C).

```
root@xilinx-zcu111-zu28-es1-2018_2:~# sdfec-demo
Initialize all devices
metal: info:      metal_uio_dev_open: No IRQ for device a0010000.data_source_top.
metal: info:      metal_uio_dev_open: No IRQ for device a0060000.stats_top.
metal: info:      metal_uio_dev_open: No IRQ for device a0020000.monitor.
metal: info:      metal_uio_dev_open: No IRQ for device a0030000.monitor.
metal: info:      metal_uio_dev_open: No IRQ for device a0020000.monitor.
metal: info:      metal_uio_dev_open: No IRQ for device a0040000.monitor.
Open FECs
Setup LDPC code for decoder
Setup LDPC code for encoder
Start decoder FEC
Start encoder FEC
Setup data source, stats and monitors
Wait for finish
Collect stats

---- RESULTS ----
        Stats:
                Decode iteration count = 800
                Channel bit error count = 21180
                Channel block error count = 100
                Uncorrected bit error count after Decode = 21569
                Uncorrected block error count after Decode = 100
        Encoder Monitors:
                In Monitor First Timestamp = 494936602
                In Monitor Last Timestamp = 494950858
                In Monitor Num Stalled Clk Cycles = 3028
                Out Monitor First Timestamp = 494937032
                Out Monitor Last Timestamp =  494951772
                Out Monitor Num Stalled Clk Cycles = 865
        Decoder Monitors:
                In Monitor First Timestamp = 494936602
                In Monitor Last Timestamp = 494950858
                In Monitor Num Stalled Clk Cycles = 3028
                Out Monitor First Timestamp = 494936381
                Out Monitor Last Timestamp =  494938039
                Out Monitor Num Stalled Clk Cycles = 965
Stop encoder FEC
Stop decoder FEC
Current config:
code        : 0
num_blocks  : 100
snr         : 6.000000
max_iter    : 8
term_on_pass: 0
mod_type    : 2
zero_data   : 0
skip_chan   : 0
New config? (y/n):
```

Figure 3 – SDFEC Linux Demo Application running on ZCU111 board

References

[1] PG256 – Soft-Decision FEC Integrated Block v1.1 Product Guide

[2] UG1144 - PetaLinux Tools Documentation: Reference Guide (2018.2)

[3] SDFEC Driver

[4] libmetal GIT repository