OpenCPI
FSK App Getting Started Guide

Version 1.4

WARNING: Run-time failures have been observed for several unit tests and applications on PCIe-based platforms. The work around requires modifying the **buffersize** attribute of the PL to PS (egress) boundary connection, as defined in the OAS. An example for modifying the OAS for executing on PCIe-based platforms is provided in a below section.

Revision History

Revision	Description of Change	Date
v1.1	Initial Release	3/2017
v1.2	Updated for OpenCPI Release 1.2	8/2017
v1.3	Updated for OpenCPI Release 1.3	1/2018
v1.4	Updated OCPI_LIBRARY_PATH paths	9/2018

Table of Contents

1 References

This document assumes a basic understanding of the Linux command line (or "shell") environment. The reference(s) in Table 1 can be used as an overview of OpenCPI and may prove useful.

Table 1: References

Title	Published By	Link
Getting Started	ANGRYVIPER Team	`Getting_Started.pdf`
Installation Guide	ANGRYVIPER Team	`RPM_Installation_Guide.pdf`
Acronyms and Definitions	ANGRYVIPER Team	`Acronyms_and_Definitions.pdf`
Overview	ANGRYVIPER Team	`http://opencpi.github.io/Overview.pdf`
FSK App[1]	OpenCPI	`FSK_app.pdf`

[1]Provides details of the "FSK App" reference application

2 Overview

This purpose of this document is to provide a compact set of instructions to build, run, and verify the OpenCPI FSK App reference application.

3 Prerequisites

This document assumes that the OpenCPI framework and the appropriate Xilinx Vivado tools have been installed. The application is supported on all of the OpenCPI platforms, but all of the examples shown here are for the Matchstiq-Z1 platform.

4 Build the *core* project

If the *core* project has not been created yet, follow the instructions in the Getting Started Guide. Once the *core* project has been created, the following ocpidev command can be used to build the primitives and workers required by the FSK app. Navigate to the *core* project directory and run the command:

```
ocpidev build --rcc --rcc-platform xilinx13_3 --hdl --hdl-platform matchstiq_z1 --no-assemblies
```

Note: The --no-assemblies argument excludes the creation of executable bitstreams.
This step takes approximately 20 minutes to complete.

5 Build the *assets* project

If the *assets* project has not been created yet, follow the instructions in the Getting Started Guide. Once the *assets* project has been created, the following ocpidev command can be used to build the primitives and workers required by the FSK app. Navigate to the *assets* project directory and run the command:

```
ocpidev build --rcc --rcc-platform xilinx13_3 --hdl --hdl-platform matchstiq_z1 --hdl-assembly fsk_modem
```

This step takes approximately 60 minutes to complete.

6 Build the FSK Application Executable

Next, the executable for the FSK Application must be built. Navigate to the `applications/FSK` of the *assets* project and run the command:

```
ocpidev build --rcc-platform xilinx13_3
```

If successful, a new directory name *target-xilinx13_3* will contain the FSK executable.

7 Running the Application

Prior to executing the application, ensure that the RX and TX ports of the platform are connected together via a coax cable, RF attenuation pads are not necessary for this test. The following steps describe setup and execution of the FSK in *txrx mode* on the Matchstiq-Z1. The *applications/FSK/Makefile* contains steps to execute the FSK in its other modes. To view these steps, open the Makefile in an editor or execute "*make show*" in the FSK application's directory.

7.1 Setting Up the Execution Platform

For embedded platforms (Matchstiq-Z1, Zedboard), connect to the platform via a serial port. Example syntax for establishing a connection can be seen below:

```
sudo screen /dev/ttyUSB0 115200
```

7.1.1 Networked mode

Networked mode is one of two modes typically used for running the application. It involves creating NFS mounts between the development platform and the execution platform to enable application deployment. A setup script is used to automate the required steps for this mode. Example syntax for running this script can be seen below:

```
source /mnt/card/opencpi/mynetsetup.sh <IP Address of Development Host>
```

7.1.2 Standalone mode

Standalone mode is the other mode used for running the application. It involves deploying the application using files only on local storage and not over a network connection. Similar to Networked mode, the setup for Standalone mode also involves running a setup script. Example syntax for running this script can be seen below:

```
source /mnt/card/opencpi/mysetup.sh
```

7.2 Setting OCPI_LIBRARY_PATH Environment Variable

The OCPI_LIBRARY_PATH environment variable is a colon-separated list of files/directories which is searched for executable artifacts during deployment of the application. For proper execution of the FSK App, all of the FSK App artifacts (detailed in the FSK App document) must be included in the `OCPI_LIBRARY_PATH` environment variable. Example syntax for setting the library path is provide in the below sections.

7.2.1 Networked Mode

In networked mode, artifact files can be accessed via NFS mounts. In the below export string, <RCC platform> must be replace with the appropriate OpenCPI's name for target RCC platform, i.e. xilinx13_3, xilinx13_4. This name can be obtained by examining the output of "ocpirun -C", when performed on the target platform.

```
$ export OCPI_LIBRARY_PATH=/mnt/net/cdk/<RCC platform>/artifacts:\
/mnt/ocpi_assets/applications/FSK/../../hdl/assemblies/fsk_modem:\
/mnt/ocpi_assets/applications/FSK/../../artifacts
```

At the time of this release, the <RCC platform> for the Matchstiq-Z1 limited to xilinx13_3, as shown in the export below. Used this export string to configured the environment on the Matchstiq-Z1 for execution of the FSK in txrx and bbloopback modes.

```
$ export OCPI_LIBRARY_PATH=/mnt/net/cdk/xilinx13_3/artifacts:\
/mnt/ocpi_assets/applications/FSK/../../hdl/assemblies/fsk_modem:\
/mnt/ocpi_assets/applications/FSK/../../artifacts
```

7.2.2 Standalone Mode

In standalone mode, artifact files can be copied to the SD card.

```
$ export OCPI_LIBRARY_PATH=/mnt/card/opencpi/artifacts
```

7.3 Running the Application

To run the application, navigate to the `applications/FSK` directory and run the executable in the target-* directory of your Execution Platform. Example syntax can be seen below.

```
$ ./target-xilinx13_3/FSK txrx
```

Reference the *FSK_ app.pdf* for executing the various modes (filerw, tx, rx, txrx, bbloopback) of the FSK application. Ensure that OCPI_LIBRARY_PATH is configured, as provided, for the respective mode and platform.

WARNING: Run-time failures have been observed for several unit tests and applications on PCIe-based platforms. The work around requires modifying the **buffersize** attribute of the PL to PS (egress) boundary connection, as defined in the OAS.
While the explicit PS to PL connection is explicitly define, the **buffersize** must be changed from 16352 to 8192, as shown in the example below:

```
<Connection>
  <Port Instance="rx_fir_real" Name="out"/>
  <Port Instance="baudTracking" Name="in" Buffersize="8192" Buffercount='4'/>
</Connection>
```

7.4 View the Results

After the application completes, the output can be found in the `applications/FSK/odata` directory. To view the results on the Development Host, navigate to the `applications/FSK` directory and execute the command:

```
$ eog odata/out_app_fsk_txrx.bin
```