

psi_multi_stream_daq

Reference Design User Guide

Content

Table of Contents

1	Introduction.....	4
1.1	Overview.....	4
1.2	Requirements	4
2	Building the Vivado Project	5
3	Building the SDK Project.....	6
4	Running the Reference Design.....	7
4.1	Hardware	7
4.2	Program Device and Start Software.....	7
4.3	Operate the Reference Design	8
4.3.1	Software.....	8
4.3.2	Firmware.....	10

Figures

Figure 1: Reference Design Overview	4
---	---

1 Introduction

This document describes the reference design for the *psi_ms_daq_axi* Vivado IP-Core and how to use it.

The reference design is meant as help to get used to the IP-core quickly and also as a tool for having a reproducible test environment in case of bugs.

1.1 Overview

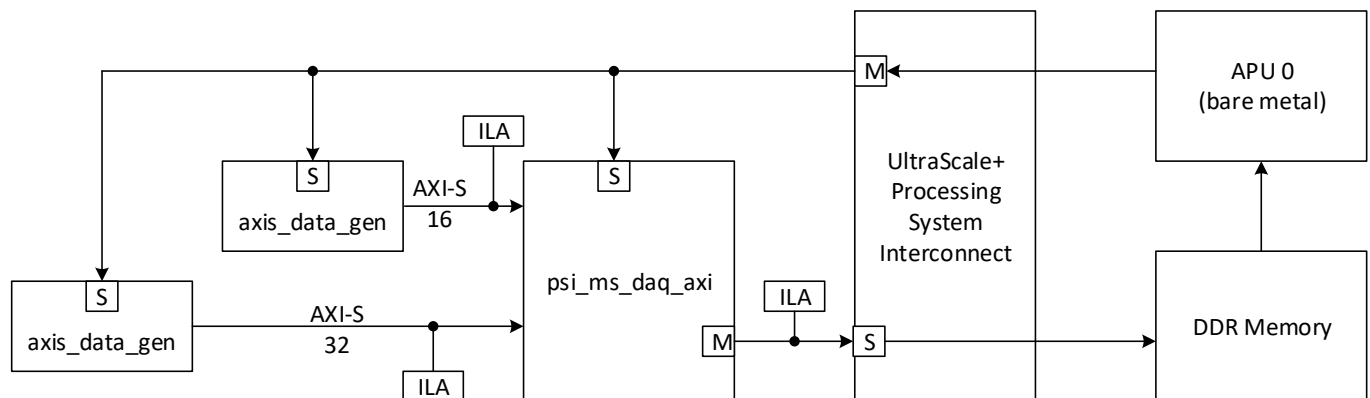


Figure 1: Reference Design Overview

In the reference design, the *psi_ms_daq_axi* component is configured to have two streams. Both streams are connected to AXI-S data generators (also a PSI component).

The data generators allow generating counter values with configurable data- and trigger-rate. Additionally the generation of triggers can be controlled at runtime. This allows easily setting up a performance- and functionality-test for different requirements.

The *psi_ms_daq_axi* component writes the data received over the AXI-S streams to the PS-DDR memory.

A bare metal application running on the APU0 configures the data generators as well as the *psi_ms_daq* component. This application uses the driver, so the software also works as reference design on the software side. The application allows the user to generate triggers via a simple command line interface.

ILAs (integrated logic analyzers) are connected to all major interface to allow for good visibility.

1.2 Requirements

To run the reference design, the following things are required:

- Xilinx ZCU102 Development Board
- Vivado 2019.1 (**Important: Don't use 2018.2, APU IRQs don't work there because of a bug**)
- SDK 2019.1 (**Important: Don't use 2018.2, APU IRQs don't work there because of a bug**)
- A terminal software (e.g. putty or tera-term)
- All dependencies of the *psi_ms_daq* repo according to the *Readme.md* file

2 Building the Vivado Project

Follow the steps below:

1. Open Vivado
2. In the TCL command line, navigate to **<PsiMsDaqRepo>/refdesign/ZCU102**
3. In the TCL command line, execute **source ./project.tcl**
4. Generate a bitstream by clicking **Flow > Generate Bitstream** in Vivado
5. Export the hardware by clicking **File > Export > Export Hardware** in Vivado.
You can choose any path to export the hardware to, just make sure you remember it for later.

If you are only interested in the software and you don't need the firmware part of the project, you can skip the whole section. A prebuilt bitstream is included in the SDK project.

3 Building the SDK Project

Follow the steps below:

1. Open SDK
2. Create a new, empty workspace.
3. Click **File > Import** to open the import dialog
 - a. Select **General > Existing Projects into Workspace**
 - b. Click **Next**
 - c. In **Select root directory** enter the path to **<PsiMsDaqRepo>/refdesign/ZCU102/Sdk**
 - d. Mark all projects for import (app, bsp and hw)
 - e. Click **Finish**
4. Because of a tool-bug, Vivado does not properly include the drivers during the *Export Hardware* step. As workaround for this, open the repositories dialog via **Xilinx > Repositories**
 - a. In the **Local Repositories** section, click **New**
 - b. Enter the path to **<LibraryPath>/Vivadopl**
5. Import the hardware specification created in step 5 of section 2 (only if section 2 was executed):
 - a. Right click on the *hw* project and select **Change Hardware Platform Specification**
 - b. In the warning dialog, click **Yes**
 - c. Select the hw export file from step 5 of section 2
 - d. Click **OK**
6. Right click on the *bsp* project and select **Board Support Package Settings**
 - a. In the **Overview > drivers** section, ensure that the drivers below are selected
 - b. *psi_ms_daq --> psi_ms_daq_axi*
 - c. *str0_tesgen_gen0 --> axis_data_gen*
 - d. *str1_tesgen_gen1 --> axis_data_gen*
 - e. click **OK**
7. Select the *app* project and clean/build it
 - a. **Project > Clean**
 - b. In the dialog popping up, click **OK**

4 Running the Reference Design

4.1 Hardware

Follow the steps below:

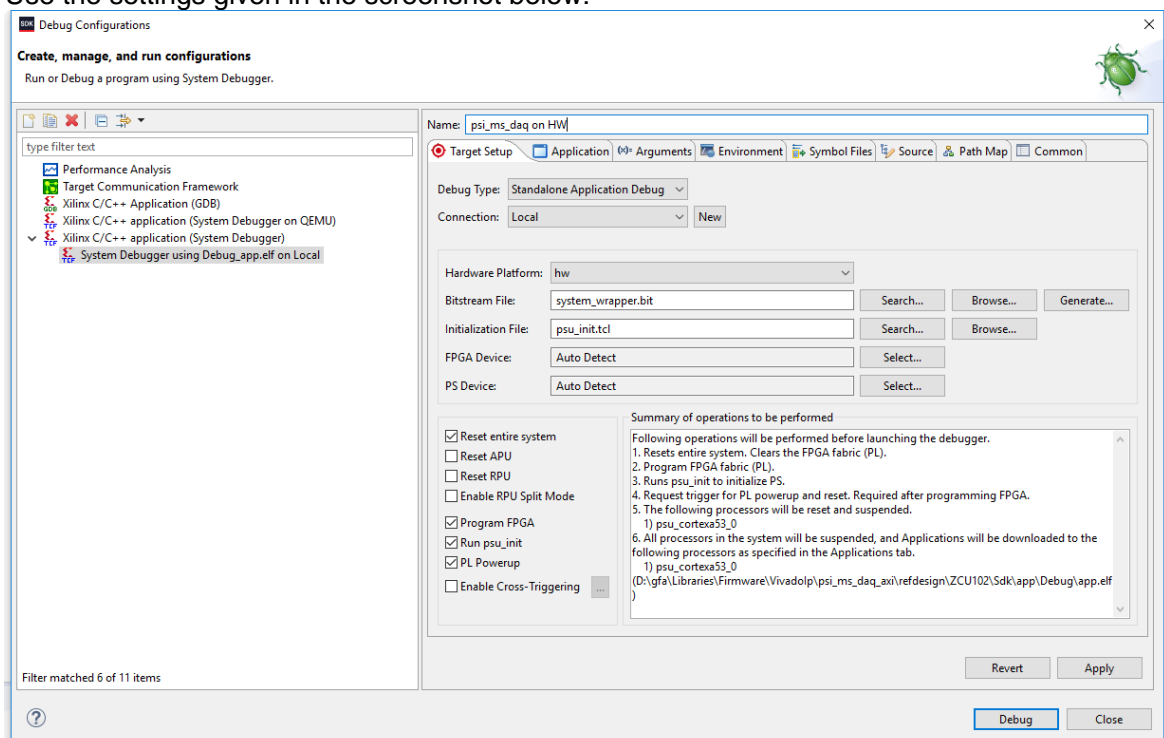
1. Power up the ZCU102 Board
2. Connect a Xilinx debugger or the USB debugging cable of the ZCU102
3. Connect an USB cable to the UART-USB
4. Open a terminal software for the ZCU102 PS UART0 (115200 baud)

Note that it is assumed that all drivers required for the ZCU102 UART are installed. For details about the drivers and the connectors to use, refer to the ZCU102 documentation.

4.2 Program Device and Start Software

Follow the steps below:

1. In SDK, click **Run > Debug Configurations**
2. Double-click to **Xilinx C/C++ application (System Debugger)**
3. Use the settings given in the screenshot below:



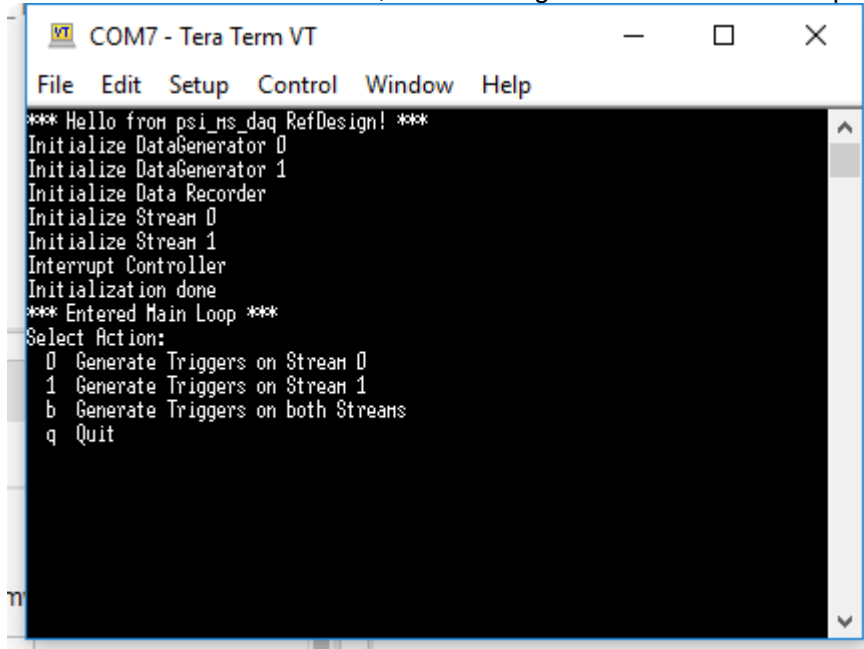
4. Click **Apply**
5. Click **Debug**
6. After all binaries are downloaded, the software automatically stops at the entry of main. Now press **Run > Resume** or **F8**

4.3 Operate the Reference Design

4.3.1 Software

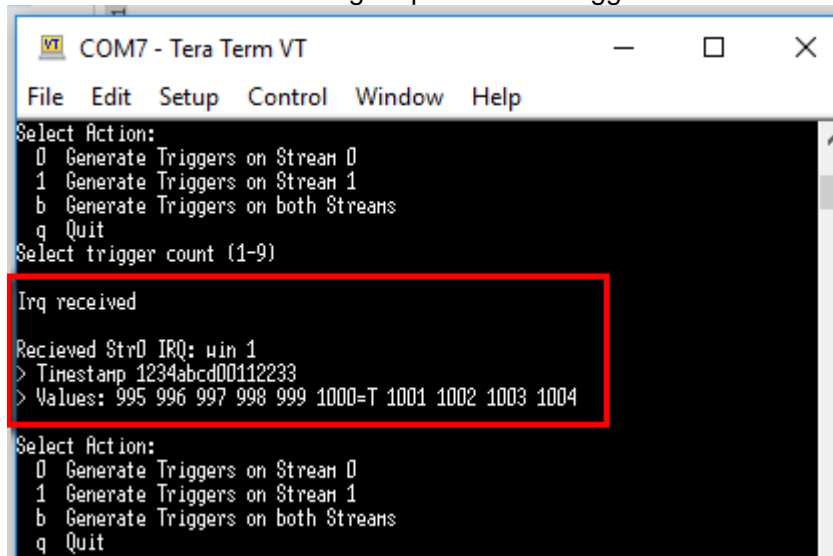
Follow the steps below:

1. After the software was started, the following information should be printed:



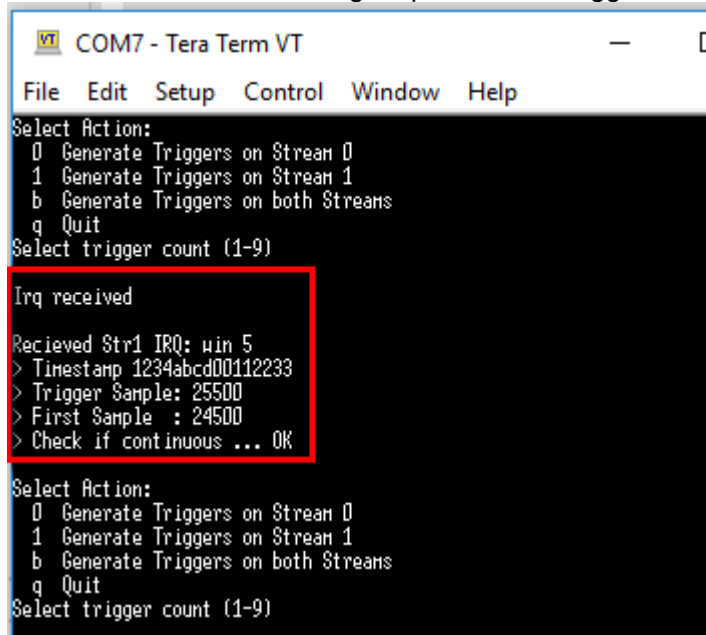
```
COM7 - Tera Term VT
File Edit Setup Control Window Help
*** Hello from psi_ms_daq RefDesign! ***
Initialize DataGenerator 0
Initialize DataGenerator 1
Initialize Data Recorder
Initialize Stream 0
Initialize Stream 1
Interrupt Controller
Initialization done
*** Entered Main Loop ***
Select Action:
0 Generate Triggers on Stream 0
1 Generate Triggers on Stream 1
b Generate Triggers on both Streams
q Quit
```

2. You can now generate triggers by following the command-line application. Always enter one character followed by *Enter*.
3. You should see the following output for each trigger on stream 0:



```
COM7 - Tera Term VT
File Edit Setup Control Window Help
Select Action:
0 Generate Triggers on Stream 0
1 Generate Triggers on Stream 1
b Generate Triggers on both Streams
q Quit
Select trigger count (1-9)
Irq received
Recieved Str0 IRQ: win 1
> Timestamp 1234abcd00112233
> Values: 995 996 997 998 999 1000=T 1001 1002 1003 1004
Select Action:
0 Generate Triggers on Stream 0
1 Generate Triggers on Stream 1
b Generate Triggers on both Streams
q Quit
```


4. You should see the following output for each trigger on stream 1:



```
COM7 - Tera Term VT
File Edit Setup Control Window Help
Select Action:
0 Generate Triggers on Stream 0
1 Generate Triggers on Stream 1
b Generate Triggers on both Streams
q Quit
Select trigger count (1-9)
Irq received
Recieved Str1 IRQ: win 5
> Timestamp 1234abcd00112233
> Trigger Sample: 25500
> First Sample : 24500
> Check if continuous ... OK
Select Action:
0 Generate Triggers on Stream 0
1 Generate Triggers on Stream 1
b Generate Triggers on both Streams
q Quit
Select trigger count (1-9)
```

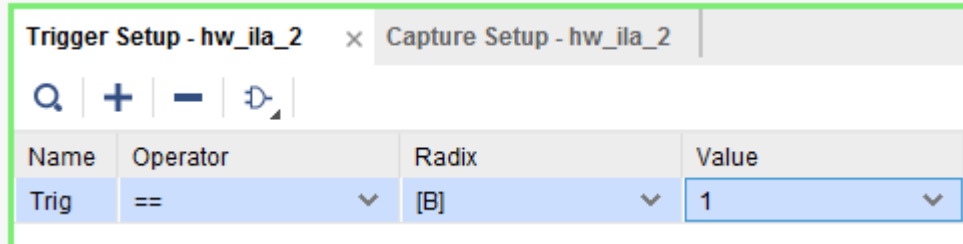
If you are interested in what happens in detail, please check the *main.c* file.


4.3.2 Firmware

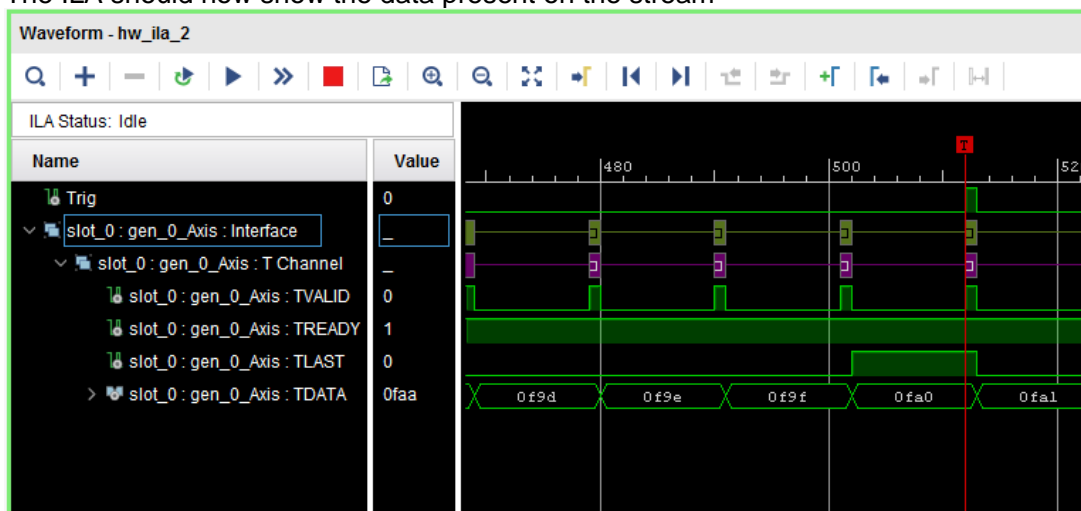
In firmware, ILAs are connected to both data streams and the AXI interface from the recorder to the memory.

As example, it is shown how to record data on the stream 0:

1. In the Vivado Project, click **Open Hardware Manager**
2. Click **Open Target** and select **Auto Connect**
3. Open the **hw_ila_2** conncted to stream0
4. Configure the ILA to trigger whenever the trigger of the data-stream is asserted



5. Arm the recorder by pressing the  button
6. Generate a trigger using the command-line application ass hown in the section above
7. The ILA should now show the data present on the stream



8. To change the data pattern, modify *main.c* (only if required)