LED display manager documentation

Clément Foucher (homepage)
Clement.Foucher@laas.fr

LAAS–CNRS
Laboratoire d'analyse et d'architecture des systèmes

Version 1.0

June 7, 2017

Contents

Document revisions

Revision number	Date	Changes
1.0	2017/06/07	Initial release.

Chapter 1

General information

1.1 Target hardware

The LED display manager IP is intended to command a DE-DP13212 LED display. As the DE-DP13212 is based on a HT1632C chip, this IP could possibly be adapted to other hardware based on the same chip.

The IP uses a AXI bus, so it should run on any Xilinx device with ability to command it from a software code running on an embedded processor. The IP can also be used without any software by using hardware direct connection or AXI interface.

Detail on how to use registers is provided in VHDL source code.

The hardware and software code provided have been tested using a ZedBoard.

1.2 License

LDM is distributed under the termes of the GNU GPL V2 license agreement. License agreement is available at http://www.gnu.org/licenses.

1.3 Download VHDL code

The source code of this IP can be downloaded using the following git command:

```
git clone git://git.renater.fr/leddisplaymana.git ldm
```

Chapter 2

Example of use on ZedBoard

This chapter presents an example of use on ZedBoard. Vivado 2017.1 was used to generate the hardware and compile the software code.

2.1 Create a Vivado IP

In Vivado, create a new project targeting ZedBoard.

Click on Settings, select *VHDL* as target language and type *led_manager_lib* as the default library.

Select Add sources › Add or create design sources. Click on Add Directories, and add the `vhdl/` folder.

Select Tools › Create and Package New IP... . Select Package your current project and choose a location to save the IP. Answer Yes when asked if you want to copy files. Finally, select Review and Package and click Package IP. You can close the project.

2.2 Create a Vivado design

In Vivado, create a new project targeting ZedBoard.

Select IP Catalog, right-click on the IP list and select Add Repository... , then choose the previously created folder containing the packaged IP.

Select Create Block Design, then Add IP, and select *ZYNQ7 Processing System*. Hit Run Block Automation › OK. Click Add IP again, and select *led_manager_axi_wrapper_v1_0*. Hit Run Connection Automation › OK.

In the left panel, select the three pins *data*, *cs* and *wr*, and press *ctrl* + *t* to make pins external. You can save and close the block design.

Right-click on the design on the source view and click Create HDL Wrapper... › Let Vivado manage wrapper and auto-update › OK.

Select Add sources › Add or create constaints, and choose the file `constraints/led_manager_zedboard.xdc`.

Your project is now ready to be generated.

First generate the output products: right-click on the design on the source view (it should now be listed as a child of the wrapper file), and select Generate Output Products... .

VERY IMPORTANT: Make sure the *Global* synthesis option is selected, as the design contains high impedence generation wich would be discarded if selecting other option.

Then, click on Generate Bitstream and wait for process completion.

When done, select File › Export › Export Hardware, check Include Bitstream and hit OK.

2.3 Compile embedded software

Still in Vivado project, select File › Launch SDK › OK.

In SDK, select File > New > Application Project, enter a project name and make sure OS is *standalone* and language is C. Click Next, choose a *Hello World* application and hit Finish.

Right-click on the newly created project, Build Configurations > Set Active > Release. Delete file `src/Helloworld.c` and folder `Debug`.

Finally, right-click on `src` folder and click Import. Select General > File System, and choose file `sample_sw/led_manager_example.c`.

2.4 Connect display to board and run

Connect the LED display to the ZedBoard using the following connection:

- cs => JA1,
- wr => JA2,
- data => JA3.

Do not forget to connect VCC and GND pins too.

Connect the ZedBoard to the computer, turn it on and select Xilinx Tools > Program FPGA. Right-click on the project, Run As > Run on Hardware.

That's all, folks!