fpga_base
Data Sheet

Content

Table of Contents

Figures

1 Introduction

This component implements basic FPGA functionality for firmware and embedded software version, compilation date, control of eight GPIO and end LEDs.

1.1 Purpose

The document here describes this very generic firmware developed for all kinds of projects.

1.2 Scope

This document provides a detailed overview of the firmware interface and specifies the user interface.

1.3 Definitions, acronyms, and abbreviations

This document is based on the “IEEE Recommended Practice for Software Requirements Specifications” [1].

BSP	Board Support Package. Collection of software drivers adding functionality to easily accessing the components used in a System On Chip.
FPGA	Field Programmable Gate Array. Programmable logic device.

1.4 References

[1] IEEE Std 830-1998, Recommended Practice for Software Requirements Specifications.

1.5 Overview

Chapter 2 provides an overview and how the firmware is related to the other firmware used.
Chapter 3 contains all the detail information on the interfaces.
Chapter 4 is meant for developers working on the core.

2 Overall description

The fpga_base is loaded into a System On Chip (SOC) as an AXI slave component.

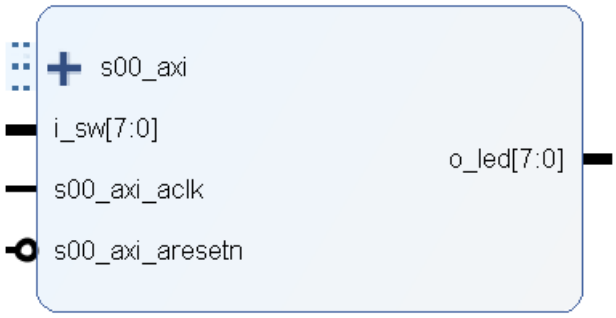

Figure 1: Component Overview

The version number is configurable through GUI, as well as device name and project name. The firmware compilation date is automatically updated by a pre TCL script during system assembly in Vivado. The software compilation date and time registers are updated by the embedded software itself in the initialization phase.

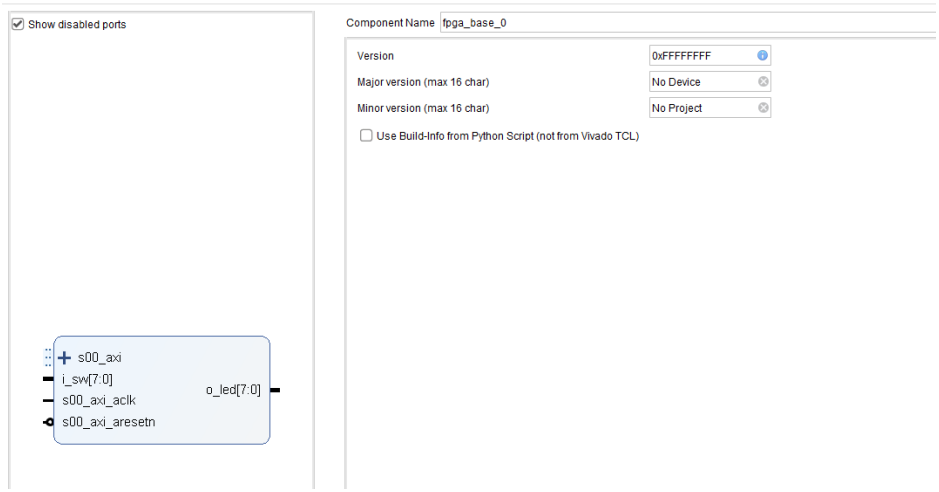

Figure 2: Component Configuration GUI

The checkbox “Use Build-info from Python Script (not from Vivado TCL)” is used to determine how the build information is updated (see sections below).

2.1 Vivado TCL based Update

To update the build date by a vivado TCL script, leave the checkbox unchecked. In this case, the user is responsible for registering the script *<root>/fpga_base.tcl* to be executed during the optimization phase. This script only updates the build date but does not change the build version (the value from the Vivado IPI GUI is used).

2.2 Python based Update

The update the build date from a python build script, set the checkbox. In this case, the user is responsible for calling the function *FpgaBaseUpdateVersion()* from a build script. This python function automatically updates the build date and stores the highest 8 digits of the git-hash in the version field. The *Version* value from the Vivado IPI GUI is ignored in this case.
The Python script also automatically checks if the repository is clean (i.e. there are not uncommitted changes) to avoid untrackable builds. If unclean, the script can still be executed but the user must confirm this interactively.

3 Request composition

3.1 Configuration

The following short notations for component parameters are used:

Address offset	R/W	Bit	Name	Description
0x00	R	31:0	VERSION	Firmware version set manually from GUI
0x04	R	31:0	FW_YEAR	FW compilation year
0x08	R	31:0	FW_MONTH	FW compilation month
0x0C	R	31:0	FW_DAY	FW compilation day
0x10	R	31:0	FW_HOUR	FW compilation hour
0x14	R	31:0	FW_MINUTE	FW compilation minute
0x18	R/W	31:0	SW_YEAR	SW compilation year
0x1C	R/W	31:0	SW_MONTH	SW compilation month
0x20	R/W	31:0	SW_DAY	SW compilation day
0x24	R/W	31:0	SW_HOUR	SW compilation hour
0x28	R/W	31:0	SW_MINUTE	SW compilation minute
0x40	R	char[16]	DEV_STRING	Device string characters
0x50	R	char[16]	PROJ_STRING	Project string characters
0x60	R/W	7:0	LED	LED control register
0x64	R	7:0	DIP	DIP switch read register

Table 1: Registers

4 Developer Information

4.1 Tools

To work on this core, the following tools are required:

- Vivado

4.2 Packaging

To simplify re-packaging of the IP-Core after changes and avoid trouble with the Vivado GUI, a packaging script was written. To re-package the IP-Core, follow the steps below:

1. Open Vivado
2. In the TCL console, navigate to the “scripts” directory (using “cd <path>”)
3. Execute “source ./package.tcl”