CITROBITS

HYPERRAM™ Memory Controller RISC-V™ reference design

User guide

Date	version	Description
11.04.2025	24.2	Initial Release

CITROBITS

Contents

CITROBITS

Introduction

This reference-design showcases the citrobits HyperRAM memory controller implemented on a CRUVI Certus-NX Base Board (CR00103).

Quick facts

Table 1 Summary of the Reference Design

General	Target Device	Certus NX LFD2NX-40
	Source Code Format	C, RTL
Software Requirements	Software Tool and Version	Lattice Propel SDK 2024.2 Lattice Propel Builder 2024.2 Lattice Radiant Software Version 2024.2 Lattice Radiant Programmer Version 2024.2
	IP Version	RISC-V MC Version 2.7.0 System Memory Version 2.3.0 AHB Lite Interconnect Version 1.3.2 AHB Lite to APB Bridge Version 1.1.2 GPIO Version 1.7.0 AHB-Lite to AXI4 Bridge 1.2.0 Citrobits HyperRAM 2.0.1
Hardware Requirements	Board	CRUVI Certus-NX Base Board (CR00103)
	Cable	Micro USB cable
Profile		ref_design_riscv_cruvi_top

Features

The RISC-V CPU is programmed to perform reads and writes to the HyperRAM with different data-widths. If an error is detected the red LED is switched on. When all memory accesses are done, the green LED is turned on.

Directory structure

Table 2 File List

Attribute	Description
hyper_ref_design_riscv_cruvi.rdf	Radiant project file
constraints/cruvi.pdf	Constraints file
sources/rtl/	Directory with plain RTL-files
sources/ip/	Directory with generated IPs
sources/bd/system/system	Directory with propel-builder design
sources/bd/system/sw/mem_test	Directory with propel workspace

Functional description

The top-module of the reference designs consists of some basic modules (pll, reset-generator and oscillator) and the propel-design which is the main part of the design:

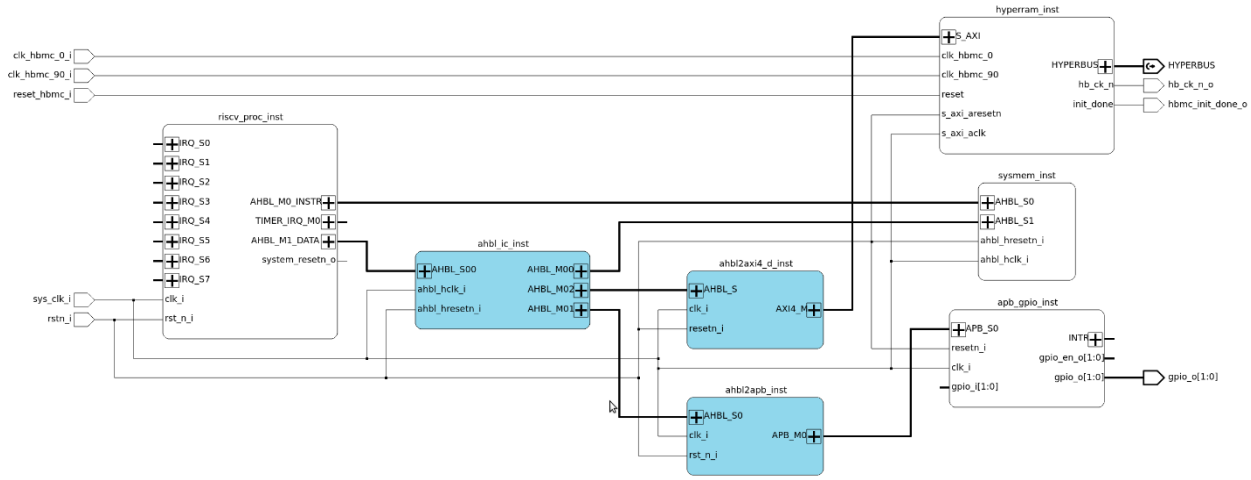

Figure 1 propel design

Besides some bus-infrastructure modules, the design consist of the following:

- riscv_proc_ist: CPU where the software is running
- sysmem_inst: Memory for the CPU
- apb_gpio_inst: Controller for LEDs
- hyperram_inst: Citrobits HyperRAM controller

The software running on the RISC-V CPU consists of mostly of the following functions:

Table 3 Software description

Function	Action
write_u32_test	Write to memory in chunks of 32-bit
read_u32_test	Read chunks of 32-bit from memory and compare with expected value
write_u16_test	Write to memory in chunks of 16-bit
read_u16_test	Read chunks of 16-bit from memory and compare with expected value
write_u8_test	Write to memory in chunks of 8-bit
read_u8_test	Read chunks of 8-bit from memory and compare with expected value

Building the reference design

Running Propel SDK Project

1. Start Propel SDK
2. Open workspace located in “sources/bd/system/sw/mem_test”
3. Build release configuration
4. Mem-file of RISC-V program is created in “sources/bd/system/sw/mem_test/Release”

Running Propel Builder Design

1. Start Propel Builder
2. Open design sources/bd/system/system/system.sbx
3. Configure hyperram_inst to use the mem-file generated in the previous section
4. Generate the design

Running Radiant Project

1. Start Radiant
2. Open the Radiant Project located in the root-directory
3. Generate the bitstream
4. Start Radiant Programmer
5. Flash the bitstream

After flashing, the green LED should turn on after a couple of seconds. If the red LED turns on, an error occurred.