Vexriscv SoC with UART & SDRAM + AXI-RAM:

AXI-SDRAM application for Vexriscv based SoC.

Instructions:

Copy your demo folder from litex installation directory `litex/litex/soc/software/demo` and paste it inside your project directory. Use the main.c file provided in this example in the demo application and replace it with the main.c file located inside your newly copied demo folder in project directory.

1. Simulation:

We can simulate this application using litex_sim tool in litex.

Run the following command to generate your SoC:

```
litex_sim --cpu-type vexriscv --bus-standard axi-lite  --with-sdram --
multiaxiram --sim-debug --no-compile-gateware
```

Before running the simulation, you have to create the binary of your application code residing in demo. The python script below converts the application code to demo.bin, which is later loaded on to the SDRAM.

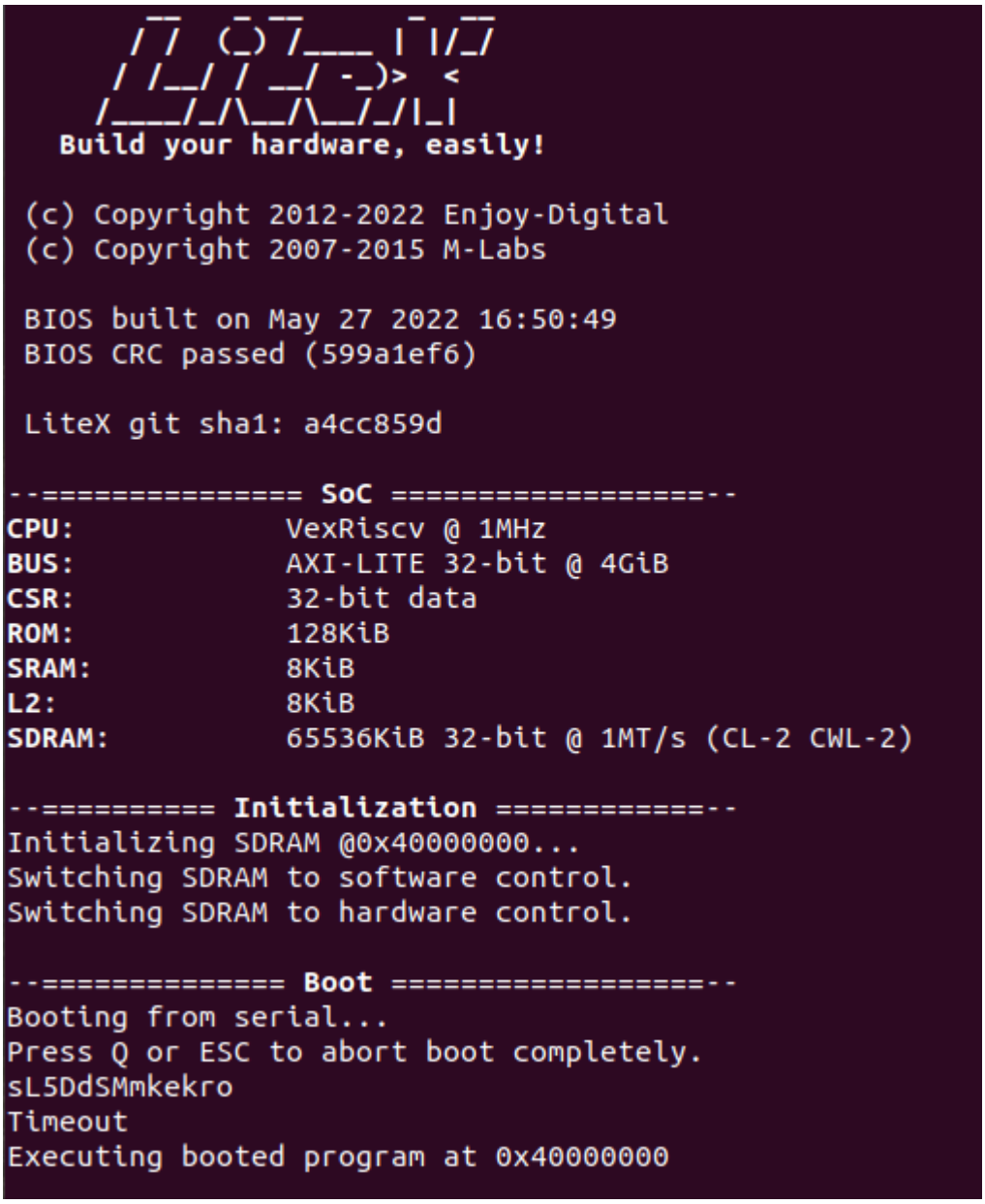
Run the following command to generate .bin file from .py file:

```
python3 ./demo/demo.py --build-path=build/sim
```

Run the following command to execute your application code onto the processor:

```
litex_sim --cpu-type vexriscv --bus-standard axi-lite  --with-sdram --
multiaxiram --sdram-init=demo.bin --sim-debug
```

Output:


Build your hardware, easily!
(c) Copyright 2012-2022 Enjoy-Digital
(c) Copyright 2007-2015 M-Labs
BIOS built on May 27 2022 16:50:49
BIOS CRC passed (599a1ef6)
LiteX git sha1: a4cc859d
--=============== SoC ==================--
CPU: VexRiscv @ 1MHz
BUS: AXI-LITE 32-bit @ 4GiB
CSR: 32-bit data
ROM: 128KiB
SRAM: 8KiB
L2: 8KiB
SDRAM: 65536KiB 32-bit @ 1MT/s (CL-2 CWL-2)
--========== Initialization ============--
Initializing SDRAM @0x40000000...
Switching SDRAM to software control.
Switching SDRAM to hardware control.
--============== Boot ==================--
Booting from serial...
Press Q or ESC to abort boot completely.
sL5DdSMmkekro
Timeout
Executing booted program at 0x40000000

```
======================
----TEST-7 SDRAM------
======================

Data written: 4008636142
Data read: 4008636142
DATA MATCHED


Data written: 4008636142
Data read: 4008636142
DATA MATCHED


Data written: 4008636142
Data read: 4008636142
DATA MATCHED


Data written: 4008636142
Data read: 4008636142
DATA MATCHED
```

```
======================
----TEST-8 AXIRAM-----
======================

Data written: 4294967295
Data read: 4294967295
DATA MATCHED


Data written: 4294967295
Data read: 4294967295
DATA MATCHED


Data written: 4294967295
Data read: 4294967295
DATA MATCHED


Data written: 4294967295
Data read: 4294967295
DATA MATCHED
```

```
======================
------TEST-RESULT-----
======================

TEST-1 SDRAM: PASSED

TEST-2 AXIRAM: PASSED

TEST-3 SDRAM: PASSED

TEST-4 AXIRAM: PASSED

TEST-5 SDRAM: PASSED

TEST-6 AXIRAM: PASSED

TEST-7 SDRAM: PASSED

TEST-8 AXIRAM: PASSED

======================
---------END----------
======================
```

2. Hardware:

Connect your Digilent Basys 3 board with your machine. We will be using the same design which we used in simulation to verify on the board. The following board file written in python creates the same SoC and later build and load it onto the Basys3 board.

Run the following command to generate your SoC:

```
./../../litex_installation/litex-
boards/litex_boards/targets/digilent_basys3.py --cpu-type vexriscv --bus-
standard axi-lite --axiram --load --build --uart-name=serial
```

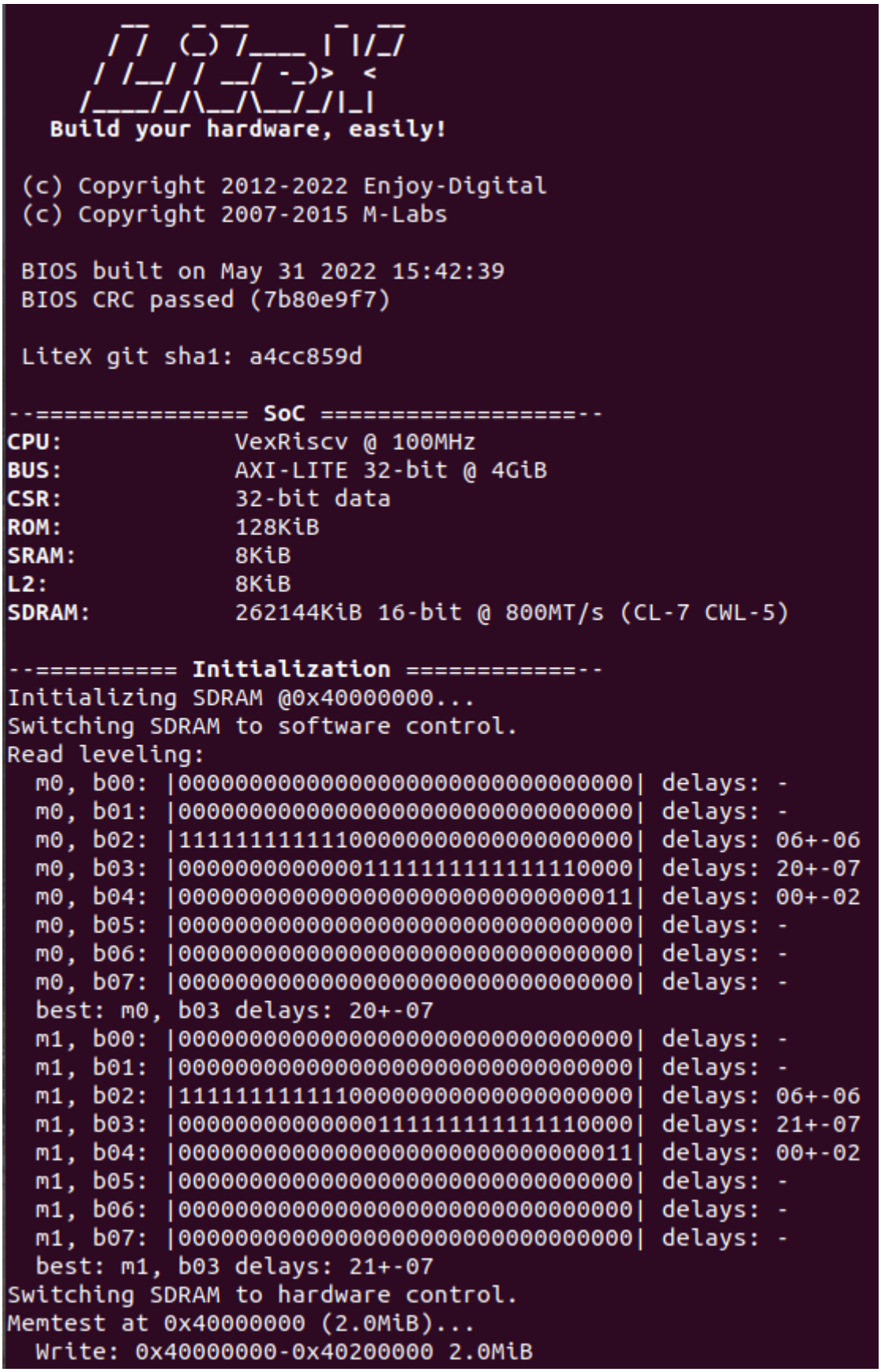
Run the following command to generate .bin file from .py file:

```
python3 ./demo/demo.py --build-path=build/digilent_basys3
```

The litex_term tool load the board with the application binary through the COM port.

```
litex_term /dev/ttyUSB2 --kernel=demo.bin
```

Output:


Build your hardware, easily!
(c) Copyright 2012-2022 Enjoy-Digital
(c) Copyright 2007-2015 M-Labs
BIOS built on May 31 2022 15:42:39
BIOS CRC passed (7b80e9f7)
LiteX git sha1: a4cc859d
SoC
CPU: VexRiscv @ 100MHz
BUS: AXI-LITE 32-bit @ 4GiB
CSR: 32-bit data
ROM: 128KiB
SRAM: 8KiB
L2: 8KiB
SDRAM: 262144KiB 16-bit @ 800MT/s (CL-7 CWL-5)
Initialization
Initializing SDRAM @0x40000000...
Switching SDRAM to software control.
Read leveling:
m0, b00: |00000000000000000000000000000000| delays: -
m0, b01: |00000000000000000000000000000000| delays: -
m0, b02: |11111111111100000000000000000000| delays: 06+-06
m0, b03: |00000000000011111111111111110000| delays: 20+-07
m0, b04: |00000000000000000000000000000011| delays: 00+-02
m0, b05: |00000000000000000000000000000000| delays: -
m0, b06: |00000000000000000000000000000000| delays: -
m0, b07: |00000000000000000000000000000000| delays: -
best: m0, b03 delays: 20+-07
m1, b00: |00000000000000000000000000000000| delays: -
m1, b01: |00000000000000000000000000000000| delays: -
m1, b02: |11111111111100000000000000000000| delays: 06+-06
m1, b03: |00000000000011111111111111110000| delays: 21+-07
m1, b04: |00000000000000000000000000000011| delays: 00+-02
m1, b05: |00000000000000000000000000000000| delays: -
m1, b06: |00000000000000000000000000000000| delays: -
m1, b07: |00000000000000000000000000000000| delays: -
best: m1, b03 delays: 21+-07
Switching SDRAM to hardware control.
Memtest at 0x40000000 (2.0MiB)...
Write: 0x40000000-0x40200000 2.0MiB

```
======================
----TEST-1 SDRAM------
======================

Data written: 0
Data read: 0
DATA MATCHED


Data written: 0
Data read: 0
DATA MATCHED


Data written: 0
Data read: 0
DATA MATCHED


Data written: 0
Data read: 0
DATA MATCHED
```

```
======================
----TEST-2 AXIRAM-----
======================

Data written: 1
Data read: 1
DATA MATCHED


Data written: 1
Data read: 1
DATA MATCHED


Data written: 1
Data read: 1
DATA MATCHED


Data written: 1
Data read: 1
DATA MATCHED
```

```
======================
------TEST-RESULT-----
======================

TEST-1 SDRAM: PASSED

TEST-2 AXIRAM: PASSED

TEST-3 SDRAM: PASSED

TEST-4 AXIRAM: PASSED

TEST-5 SDRAM: PASSED

TEST-6 AXIRAM: PASSED

TEST-7 SDRAM: PASSED

TEST-8 AXIRAM: PASSED

======================
---------END----------
======================
```

Application

In this application code, we write and read data to the SDRAM and the results are shown on UART console (LiteX Console).