# Vexriscv SoC with UART & RAM

Axi Ram application code running on a Vexriscv. This design contains a Vexriscv processor, ON chip axi ram and UART.

### Instructions:

You can follow the below steps to generate the designs and simulate the application on Verilator.

## Generate Verilog for the LiteX design (No Simulation)

```
litex_sim --cpu-type vexriscv --axiram --no-compile-gateware
```

## Generate and Simulate the verilog for the LiteX design

Here we simulate the Axi Ram example using litex_sim_rs script provided in the example design directory.

### The following command generates your SoC:

```
~/litex_instll/litex_rs/raptor_example_designs/Vexriscv_axi_ram/litex_sim_r
s.py --integrated-main-ram-size=0x10000 --cpu-type=vexriscv --axiram --no-
compile-gateware --sim-debug
```

### Generate binary for the application code

Run the following command to generate .bin file:

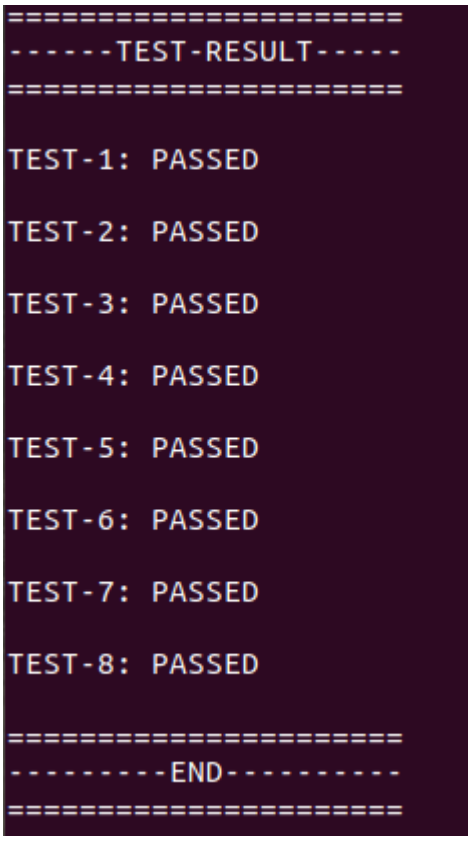
```
python3
~/litex_instll/litex_rs/raptor_example_designs/Vexriscv_axi_ram/test/demo/d
emo.py --build-path=build/sim
```

### Simulating the application using Verilator

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

```
~/litex_instll/litex_rs/raptor_example_designs/Vexriscv_axi_ram/litex_sim_r
s.py --integrated-main-ram-size=0x10000 --axiram --cpu-type vexriscv --ram-
init=demo.bin --sim-debug
```

### Output:




## Application

This application code does multiple tests onto the axiram, these tests do multiple write and read sequences on the ram to test the integration of the IP.

# Compile design for a Gemini FPGA Device

### Source Raptor for compilation

Raptor needs to be sourced before using Gemini.py

### Compiling a design on Raptor for Gemini device

```
~/litex_instll/litex_rs/Example_designs/Vexriscv_axi_gpio_led/gemini.py --
toolchain=Raptor --device=gemini --cpu-type=vexriscv --axiram --build
```