Vexriscv SoC with UART & Hello World:

Hello World 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 the hello world example using litex_sim tool in litex.

Run the following command to generate your SoC:

```
litex_sim --integrated-main-ram-size=0x10000 --cpu-type=vexriscv --no-compile-gateware --sim-debug
```

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 RAM.

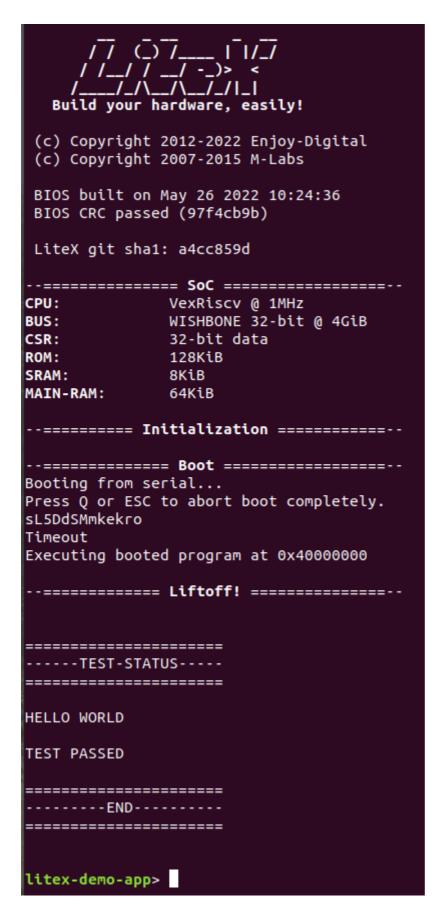
Run the following command to generate .bin file from .py file:

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

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

```
litex_sim --integrated-main-ram-size=0x10000 --cpu-type vexriscv --ram-init=demo.bin --sim-debug
```

Output:



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 Basys board.

Note: Before using this GPIO design on hardware, we need to replace the board files with the necessary changes provided in the board files directory.

Run the following command to generate your SoC:

```
./../../litex_installation/litex-
boards/litex_boards/targets/digilent_basys3.py --integrated-main-ram-
size=0x10000 --cpu-type=vexriscv --build --load --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/ttyUSB1 --kernel=demo.bin
```

Output:

```
__/_/|_|
  Build your hardware, easily!
 (c) Copyright 2012-2022 Enjoy-Digital
 (c) Copyright 2007-2015 M-Labs
BIOS built on May 17 2022 15:24:23
BIOS CRC passed (9ee2154e)
LiteX git sha1: a4cc859d
CPU:
              VexRiscv @ 100MHz
              WISHBONE 32-bit @ 4GiB
BUS:
CSR:
              32-bit data
ROM:
              128KiB
SRAM:
              8KiB
MAIN-RAM:
              64KiB
Memtest at 0x40000000 (64.0KiB)...
 Write: 0x40000000-0x40010000 64.0KiB
  Read: 0x40000000-0x40010000 64.0KiB
Memtest OK
Memspeed at 0x40000000 (Sequential, 64.0KiB)...
 Write speed: 166.5MiB/s
  Read speed: 87.4MiB/s
--========== Boot ===========---
Booting from serial...
Press () or ESC to abort boot completely.
sL5DdSMmkekro
[LITEX-TERM] Received firmware download request from the device.
[LITEX-TERM] Uploading demo.bin to 0x40000000 (4660 bytes)...
[LITEX-TERM] Upload calibration... (inter-frame: 10.00us, length: 64)
[LITEX-TERM] Upload complete (9.7KB/s).
[LITEX-TERM] Booting the device.
[LITEX-TERM] Done.
Executing booted program at 0x40000000
--============ Liftoff! ==========---
```

Application

In this application code, we print a Hello world to the UART console(LiteX Console).