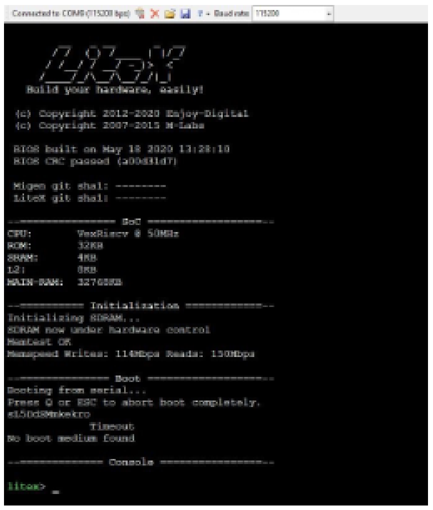
Demo - Litex Development Enviroment

Litex is an alternative and open-source development enviroment for FPGA designs written in Python. It offers Migen, a python like Hardware Description Language.

For every board supported there is a demo within the Litex installation.

Description of the demo

The demo for the **TEI009 / C10LP RefKit development board** consists of a design, describing a RISC-V processor so that it can run C code to display a BIOS promt via **UART** (**115200 / 8N1**) , furthermore is makes the LEDs **D13** to **D17** blink.
The demo is non persistent and is delete by a power down or a board reset (Switch **S2**) . Also a soft reset is available (Switch **S7**) .



```
    Build your hardware, easily!

 (c) Copyright 2012-2020 Enjoy-Digital
 (c) Copyright 2007-2015 M-Labs

 BIOS built on May 18 2020 13:28:10
 BIOS CRC passed (a00d31d7)

 Migen git sha1: --------
 LiteX git sha1: --------

--================== SoC ====================--
CPU:        VexRiscv @ 50MHz
ROM:        32KB
SRAM:       4KB
L2:         8KB
MAIN-RAM:   32768KB

--============ Initialization ==============--
Initializing SDRAM...
SDRAM now under hardware control
Memtest OK
Memspeed Writes: 114Mbps Reads: 130Mbps

--================== Boot ===================--
Booting from serial...
Press Q or ESC to abort boot completely.
sL5DdSMmkekro
                Timeout
No boot medium found

--================= Console =================--

litex> _
```

Tested on

Windows 10 - Version 1909 / build 18363
Python 3.82 at least 3.6 is required
Intel Quartus Prime Lite 19.1

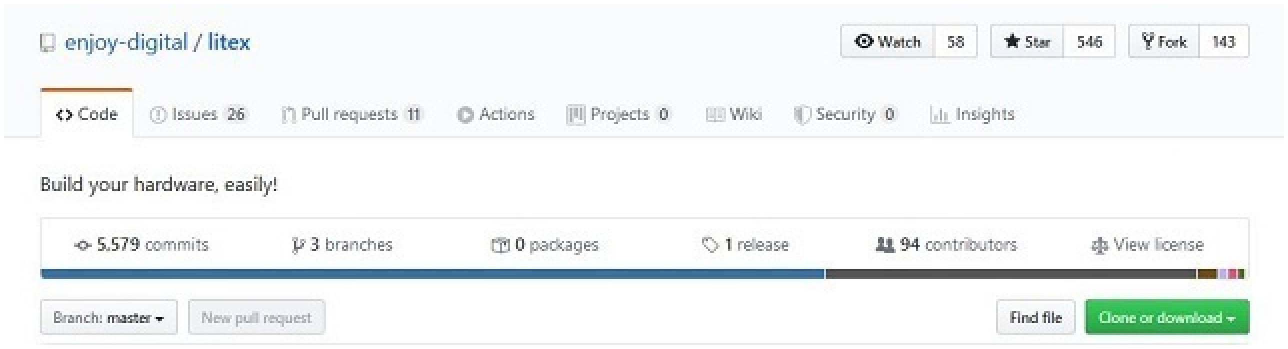
Prequesitions

Intel Quartus Prime Lite:
Download and install it. If necessary, add device support for Cyclone 10 LP. The device support can be installed via download of the file from Intel's webpage and running the program Device Installer.

Python 3.6+:
Download and install it.

In the Python installation folder is the python.exe, on windows it is necessary to have a copy of the python.exe in the same folder under the name python3.exe . Open the python folder and copy the python.exe into the same folder and rename it to python3.exe .

Litex setup-script:
The script can be sourced at the projects git web page: https://github.com/enjoy-digital/litex.git as part of the sources.



Download the source litex-master.zip archive via the green button "Clone or download" and extract it.

Diskspace, at least 1.5 GB.

RISC-V Toolcain:
This can easily be installed via the Litex setup script.

Make:
Included as part of the NIOS2 package within the Intel Quartus Lite installation.

Path variables

Path variables point the litex scripts to the necessary executables for running various tasks. They need to be present before running a script. In windows they can be edited and stored permanently or non permanently.

This manual edits the path variable non permanently, the changes made effect only the CMD console in which they are made and will be lost by closing the console.

Necessary editions to the path variable are described where needed. The Litex scripts needs to know the folder of the python3.exe . The necessary change to the path variable should be conducted by the Python installation.

Installation of Litex

Make a folder for your litex-installation. It is highly recommand to use an installation-folder inside your user directory.

Copy from the litex-master.zip archive the script file litex_setup.py into the installation-directory.

Open a CMD console and browse to this directory and run the installation command for Litex:
python llitex_setup.py init install - All CMD console commands are in *italic letters*.

This downloades and installs Litex along the necessary Python modules into the installation directory.

Error handling:
No access to / missing files Delete files or folder in question and restart the installation
Use the console to start the setup of the package in question manually
Separate the command *python llitex_setup.py init install* into *python llitex_setup.py init* and *python llitex_setup.py install*

Installation of a RISC-V toolchain

In the same CMD console run the command:
python litex_setup.py gcc

This downloads and extracts the toolchain (folder riscv64-unknown-elf-gcc-8.3.0-2019.08.0-x86_64-w64-mingw32).

Setup the path environmental variables

Edit the path variables via copying these commands.

Intel Quartus Prime Lite:
set PATH=%PATH%;C:\IntelFPGA_lite\19.1\quartus\bin64

RISC-V Toolcain
set PATH=%PATH%;C:\users\<your litex installtion folder>\riscv64-unknown-elf-gcc-8.3.0-2019.08.0-x86_64-w64-mingw32\bin

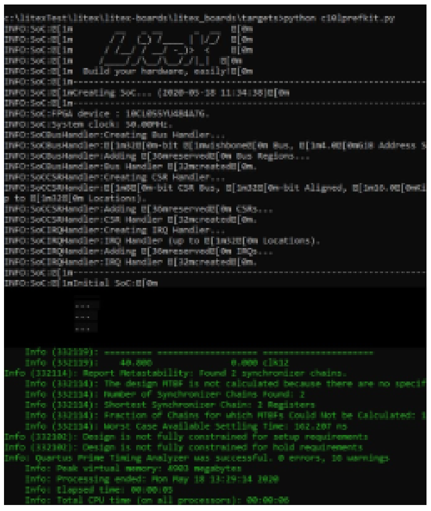
Make:
set PATH=%PATH%;c:\IntelFPGA_lite\19.1\nios2eds\bin\gnu\H-x86_64-mingw32\bin

Error handling:
In case of the already existing path variable leading to errors during the build, one can override the PATH variable completely. The following command should be executed, which overrides the PATH variable temporarily and adds Python to it.
set PATH=c:\<Path to your python installation folder>
Issue the path commands above for Intel Quartus Prime Lite, RISC-V Toolcain and Make.

Generate the Litex BIOS demo

To generate the demo, navigate to the folder
C:\users\<your litex installtion folder>\litex-boards\litex_boards\targets
and execute the build command with the --build option:
python c10lprefkit.py --build

These starts the build beginning with the software compilation followed by the hardware design generation.



```
c:\litexTest\litex\litex-boards\litex_boards\targets>python c10lprefkit.py
INFO:SoC:[1m      [illegible]      [0m
INFO:SoC:[1m      [illegible]      [0m
INFO:SoC:[1m      [illegible]      [0m
INFO:SoC:[1m      [illegible]      [0m
INFO:SoC:[1m  Build your hardware, easily![0m
INFO:SoC:[1m--------------------------------------------------------------------
INFO:SoC:[1mCreating SoC... (2020-05-18 11:34:38)[0m
INFO:SoC:[1m--------------------------------------------------------------------
INFO:SoC:FPGA device : 10CL055YU484A7G.
INFO:SoC:System clock: 50.00MHz.
INFO:SoCBusHandler:Creating Bus Handler...
INFO:SoCBusHandler:[1m32[0m-bit [1mwishbone[0m Bus, [1m4.0G[0miB Address S
INFO:SoCBusHandler:Adding [36mreserved[0m Bus Regions...
INFO:SoCBusHandler:Bus Handler [32mcreated[0m.
INFO:SoCCSRHandler:Creating CSR Handler...
INFO:SoCCSRHandler:[1m8[0m-bit CSR Bus, [1m32[0m-bit Aligned, [1m16.0[0mKi
p to [1m32[0m Locations).
INFO:SoCCSRHandler:Adding [36mreserved[0m CSRs...
INFO:SoCCSRHandler:CSR Handler [32mcreated[0m.
INFO:SoCIRQHandler:Creating IRQ Handler...
INFO:SoCIRQHandler:IRQ Handler (up to [1m32[0m Locations).
INFO:SoCIRQHandler:Adding [36mreserved[0m IRQs...
INFO:SoCIRQHandler:IRQ Handler [32mcreated[0m.
INFO:SoC:[1m--------------------------------------------------------------------
INFO:SoC:[1mInitial SoC:[0m

        ...
        ...
        ...

    Info (332119): ========= =================== ====================
    Info (332119):    40.000                   0.000 clk12
Info (332114): Report Metastability: Found 2 synchronizer chains.
    Info (332114): The design MTBF is not calculated because there are no specif
    Info (332114): Number of Synchronizer Chains Found: 2
    Info (332114): Shortest Synchronizer Chain: 2 Registers
    Info (332114): Fraction of Chains for which MTBFs Could Not be Calculated: 1
    Info (332114): Worst Case Available Settling Time: 102.207 ns
Info (332102): Design is not fully constrained for setup requirements
Info (332102): Design is not fully constrained for hold requirements
Info: Quartus Prime Timing Analyzer was successful. 0 errors, 10 warnings
    Info: Peak virtual memory: 4903 megabytes
    Info: Processing ended: Mon May 18 11:29:34 2020
    Info: Elapsed time: 00:00:05
    Info: Total CPU time (on all processors): 00:00:06
```

The build can end with an error message which is of no concern to the successful generation of the file.

The resulting files are stored inside the same folder in **soc_basesoc_c10lprefkit**, the demo file **top.sof** is stored in **targets\soc_basesoc_c10lprefkit\gateware.**

Program that file to the board and the demo runs.

Options:
To generate only the FPGA design, containing the RISC-V processor and blinking LEDs, without the software part, the Litex UART BIOS promt.
python c10lprefkit.py --no-compile-software

Generate only the software sources.
python c10lprefkit.py --no-compile-gateware