Motivation

Why did I choose PyRTL?

PyRTL is **embedded in Python**, inheriting the power and ease of use of the popular Python.

Personal **interest**

Recommended by **professionals**

Papers that I read about PyRTL (PyRTL, PyRTLMatrix, Wire Sorts)

The introduction **video by Dr. Tim Sherwood** on Youtube

Fantastic **demo by Dr. Michael Christensen** in our class

What is it good for?

Highly **abstracted constructs with implicit clock and reset** signals which prevents generation of unintended latches.

Great for learning a high level environment capable of creating complicated circuits

If one knows **Python** S/He will learn PyRTL with not much problem

Inherits the **interactive environment in Python** making learning much faster compared to Verilog

Rapid development

Complicated circuits can be implemented with **small amount of code**

Simplicity is preferred in its philosophy over flexibility according to the authors of PyRTL

Many tools good for production like built in **Waveform Viewer, Simulator, Graph Generator**, **Analysis** and **optimization** tools

Philosophy

What I aim for in my assignments:

Apart from:

Learning PyRTL by doing

Considering the **Learning Outcomes**

The goal is:

Appreciation of the **power of abstraction** offered by PyRTL. (LSFR behavioral)

To get students comfortable with the **structures inherited from Python**. (Bit arrangement of **List** objects in PyRTL felt odd until getting used to it)

To get those familiar with Verilog get used to the concept of Elaboration by Execution (**sometimes order matters**)

To get students comfortable with **tools offered in PyRTL** specially the built in waveform viewer

Links:

Github repository:

https://github.com/MikeAnj/CSE293

Lab 1 and 2 Google Docs

Lab 1:

https://docs.google.com/document/d/1LYDXyFrN1tXEWb9VitemztiZ0RRkEPExi4FMquxpndY/edit

Lab 2:

https://docs.google.com/document/d/14Us2uT1Qr19BGQNmYdjr1ASBFaCkbOs9XP1L1ZtDAbg/edit

Summary of Lab 1

Brief description of Parts and Sub-parts

Part 1: Implementation of **LFSR in combinatorial logic**

Part 2: Implementation of **LFSR in behavioral manner**

Learning Outcome

LFSR can be a **good candidate** for a useful circuit simple enough to implement as an educational subject.

The combinatorial part is intended to make the student think and work on **creating structures like Modules in Verilog** but implemented as Python/PyRTL functions.

Student will come out of this assignment with an appreciation of **abstraction offered by PyRTL** and yet ease and safety of implementation.

Student will feel comfortable with the fantastic **Waveform viewer**.

Summary of Lab 2

Brief description of Parts and Sub-parts

Assignment 2:

Part 1: Implementation of a **Digital Clock**

Part 2: Using the **extra tools offered in PyRTL**

Learning Outcome

This assignment makes students **think of structures that count numbers and update other modules**.

This assignment will make students **think about decision making in circuits** and **conditional assignments and nested context manager "with"**.

In Part 2, students will start using **extra powerful tools** provided by PyRTL