ACCESSIBILITY OF CHIP DESIGN TO THE NON-PROFESSIONAL

Alex Goldstein (Alex_Goldstein@outlook.com), Tim Edwards, Ph.D. (tim@opencircuitdesign.com)

Project Overview

01

Design a microchip that will fulfill a certain purpose and detail its functionality in Verilog Code

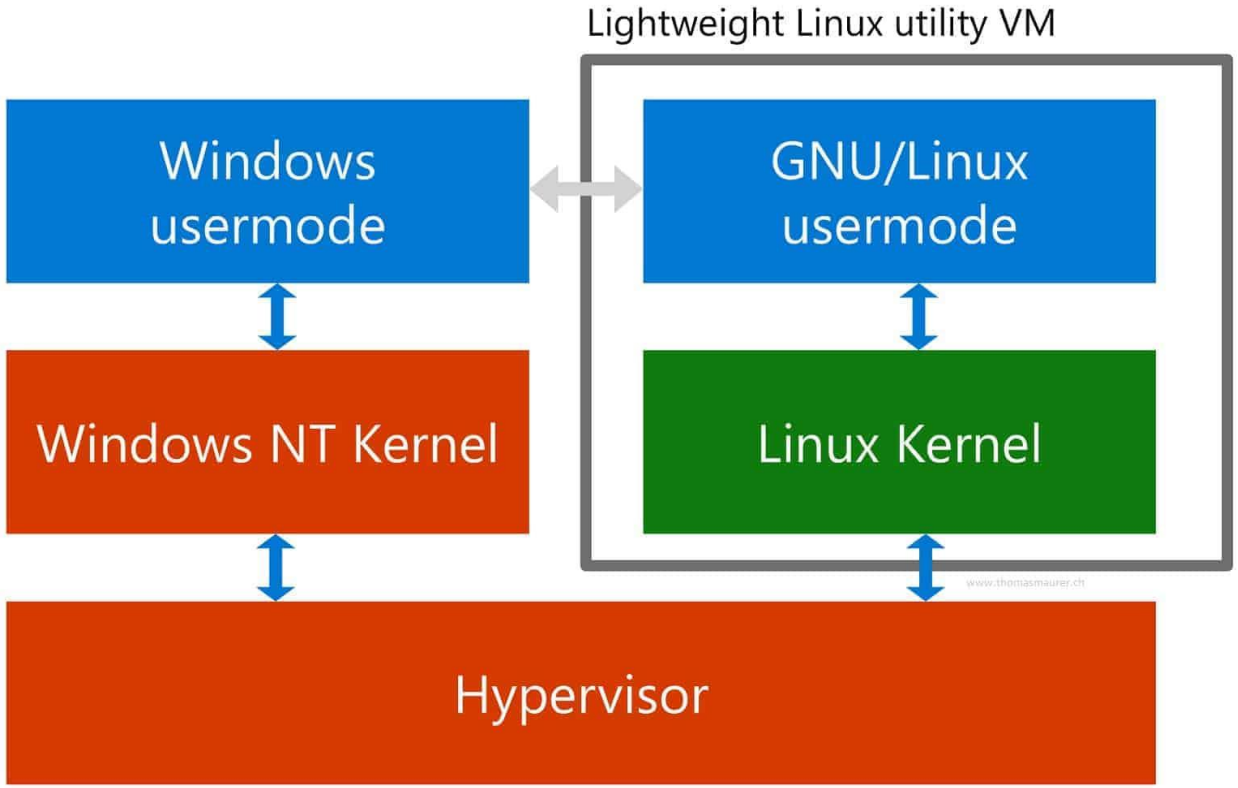
02

Create a potential chip layout to be fabricated using the OpenLANE software

03

Submit the chip layout to the Efabless Open MPW 7 Shuttle to be fabricated

WSL 2 architecture overview

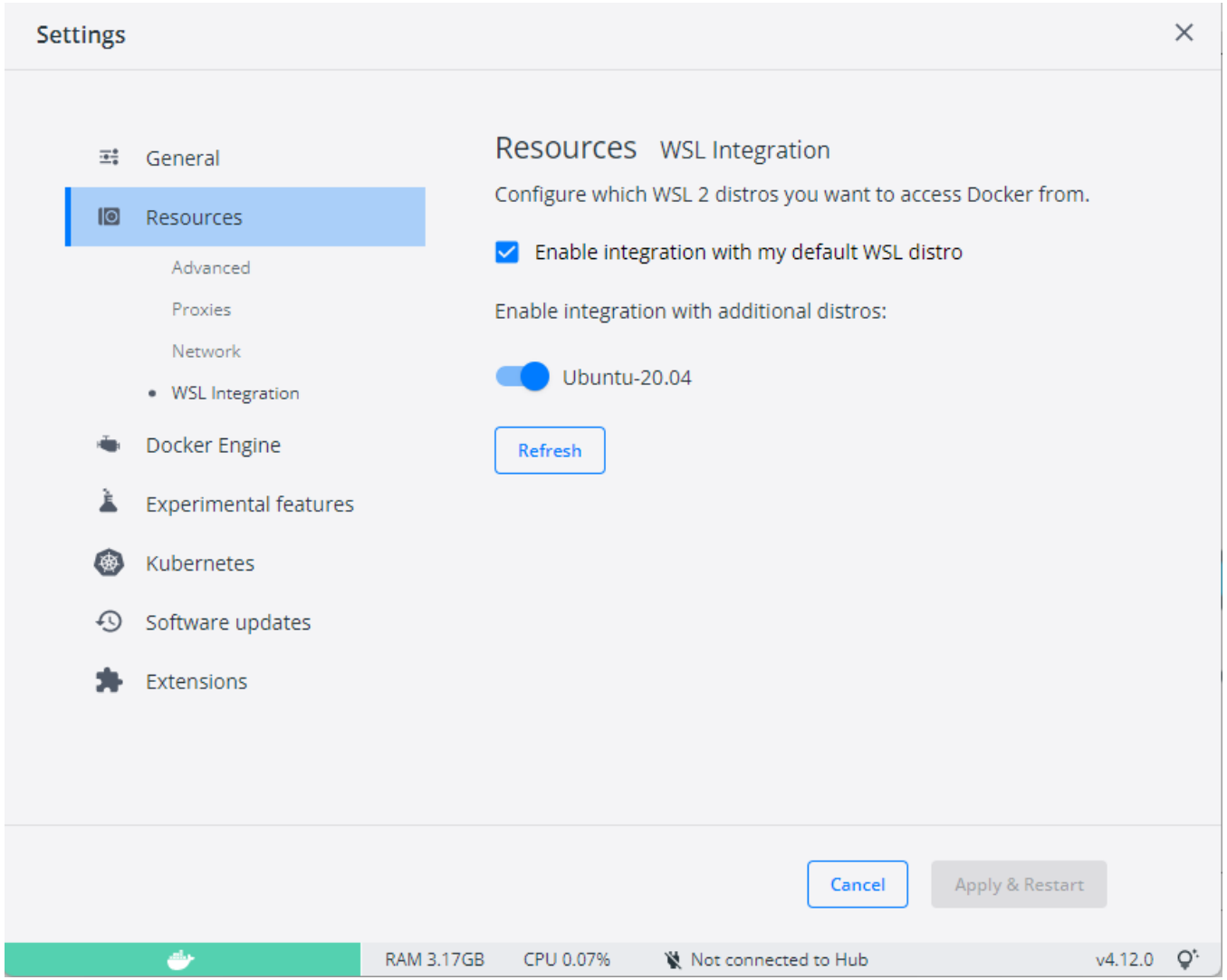


Setting Up the Synthesis Environment

- OpenLANE only runs in a Unix-based environment
- About 80% of home computers run Windows
- We can get around this by utilizing the “Windows Subsystem for Linux (WSL)”

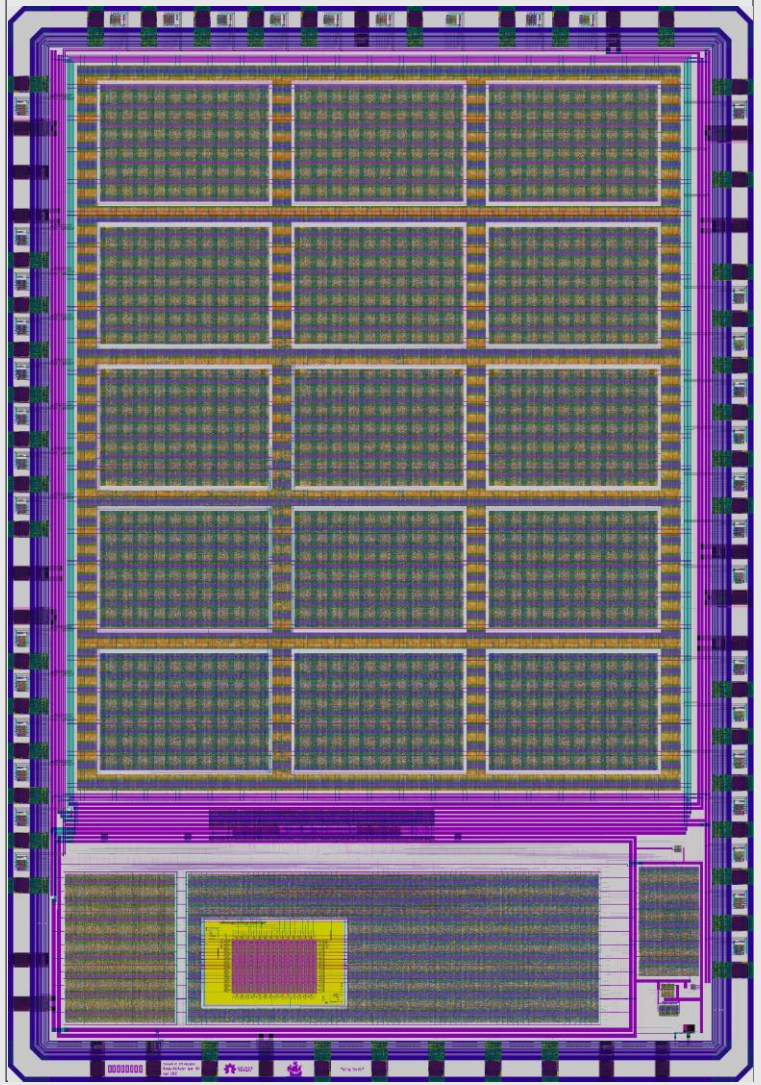
What About Dependencies?

- Almost all of the Dependencies can be installed as normal through the terminal
- The Docker Daemon cannot be run in the terminal on its own
- The Docker Desktop Application allows us to still use the daemon through the engine's WSL 2 Integration

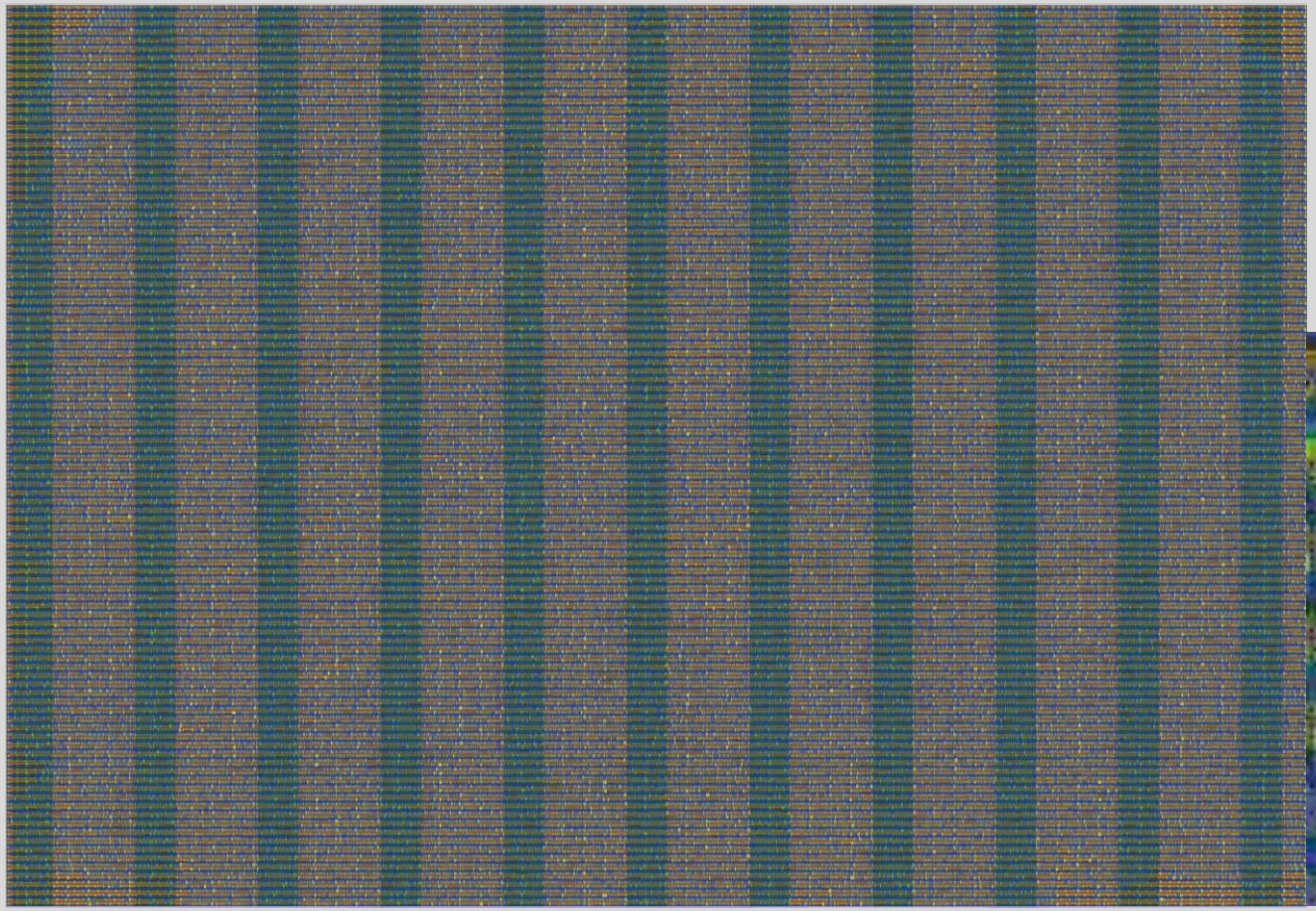


```
$ gh repo clone
AlexanderJGoldstein/chaos_automaton_Summer_2022 -- --
depth=1
$ cd chaos_automaton_Summer_2022
$ mkdir dependencies
$ export OPENLANE_ROOT=$PWD/dependencies/openlane_src
$ export PDK_ROOT=$PWD/dependencies/pdks
$ export PDK=sky130B
$ make setup
$ make chaos_subarray
$ make user_project_wrapper
$ make ship
```

Synthesis Process

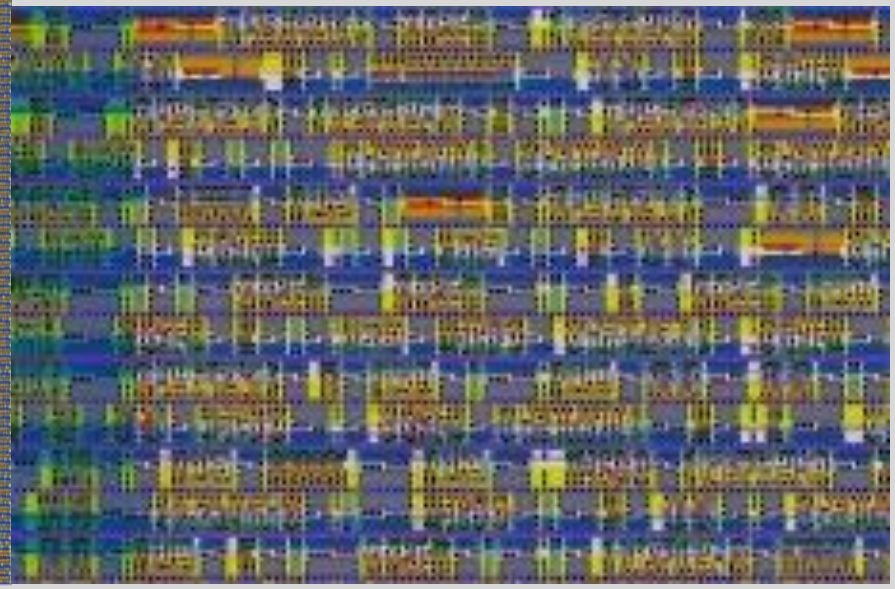


The Final Design



The final layout for a single completed subarray

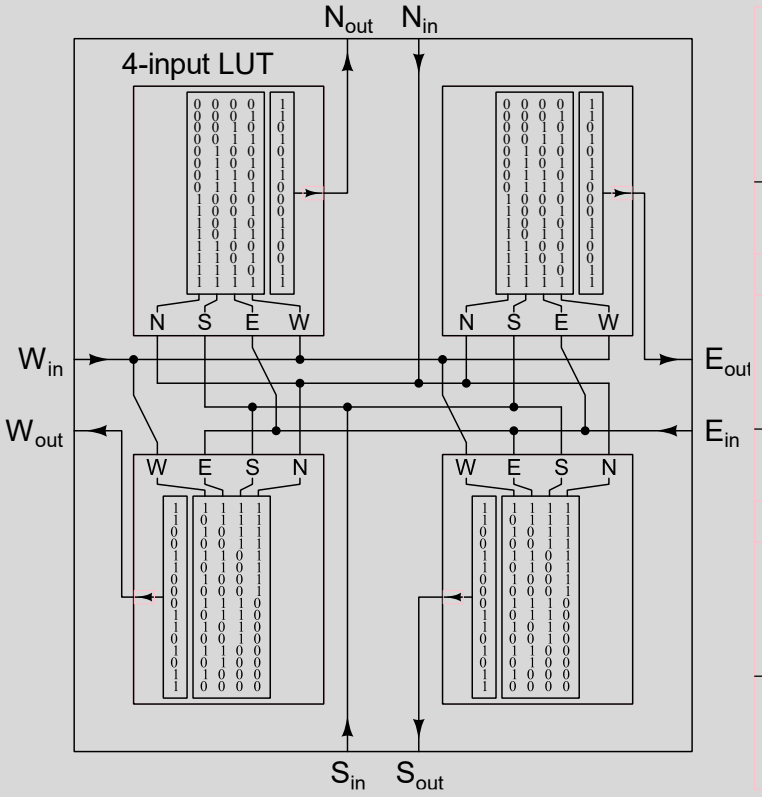
- 3 by 5 array of subarrays (15 total)
- Each subarray is a 10 x 6 array of chaos cells (60 per subarray)
- There are 900 chaos cells in total!
- 63.68% Utilization



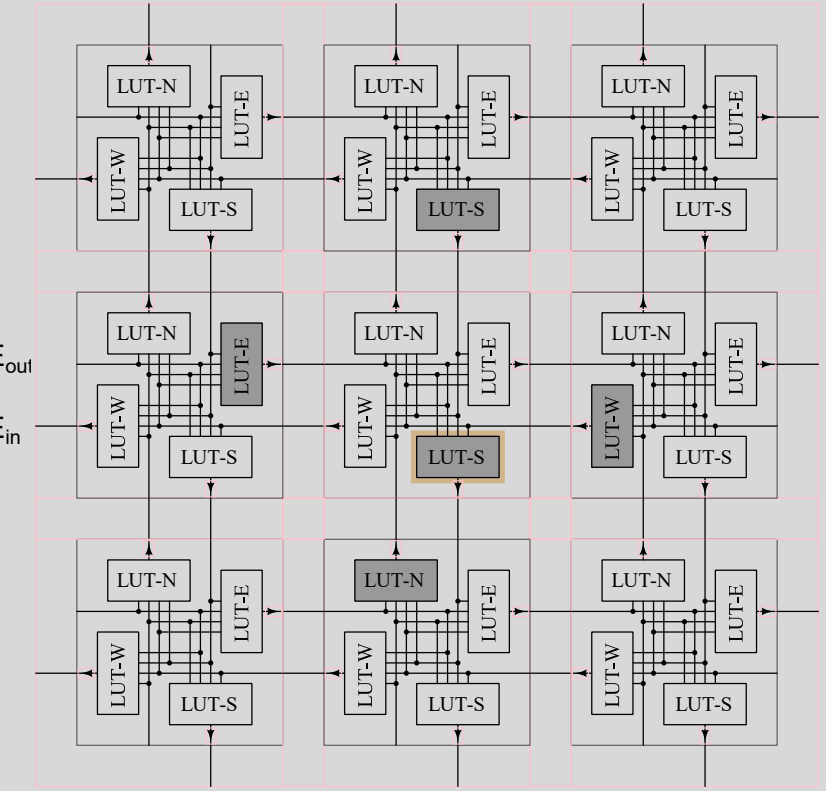
A close-up of 0.4% of the subarray (25,000% magnification)

What is this “Chaos Automaton?”

- A large array of interconnected LUT chips
- The array (apart from programming) is completely asynchronous



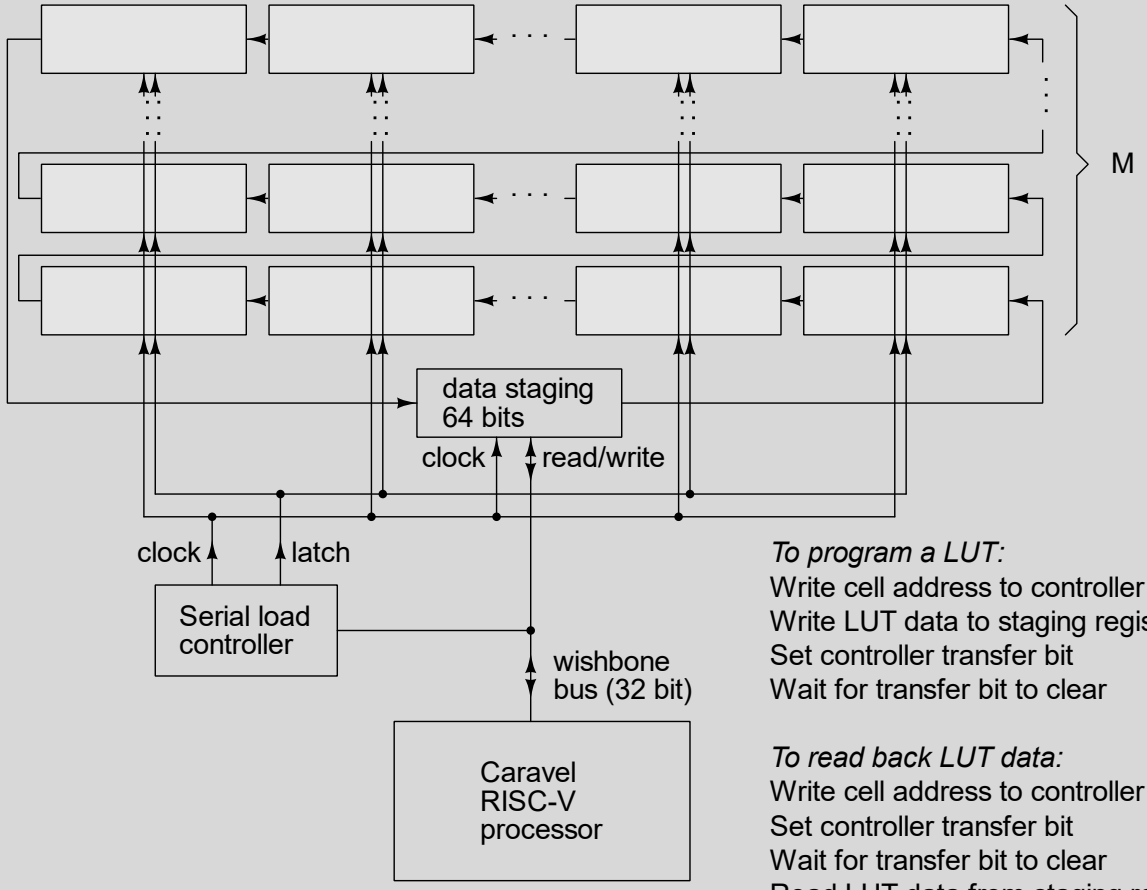
A single chaos cell



A 3 x 3 array of chaos cells

What is this “Chaos Automaton?”

- These LUTs are programmed individually by the Caravel Wrapper’s RISC V CPU



To program a LUT:
Write cell address to controller
Write LUT data to staging register
Set controller transfer bit
Wait for transfer bit to clear

To read back LUT data:
Write cell address to controller
Set controller transfer bit
Wait for transfer bit to clear
Read LUT data from staging register

WHY THIS DESIGN FOR A CELLULAR AUTOMATON?

The Chaos Automaton is an array-based (cellular) design. One separate goal was to see just how large of an array could fit within the user project area.

Custom Settings

- set ::env(ROUTING_CORES) 16
- set ::env(MACRO_PLACEMENT_CFG) $script_dir/macro_placement.cfg

Subarray:

- set ::env(DIE_AREA) "0 0 825 585“
- set ::env(PL_TARGET_DENSITY) 0.45

What Were the Obstacles?

01

Memory Consumption

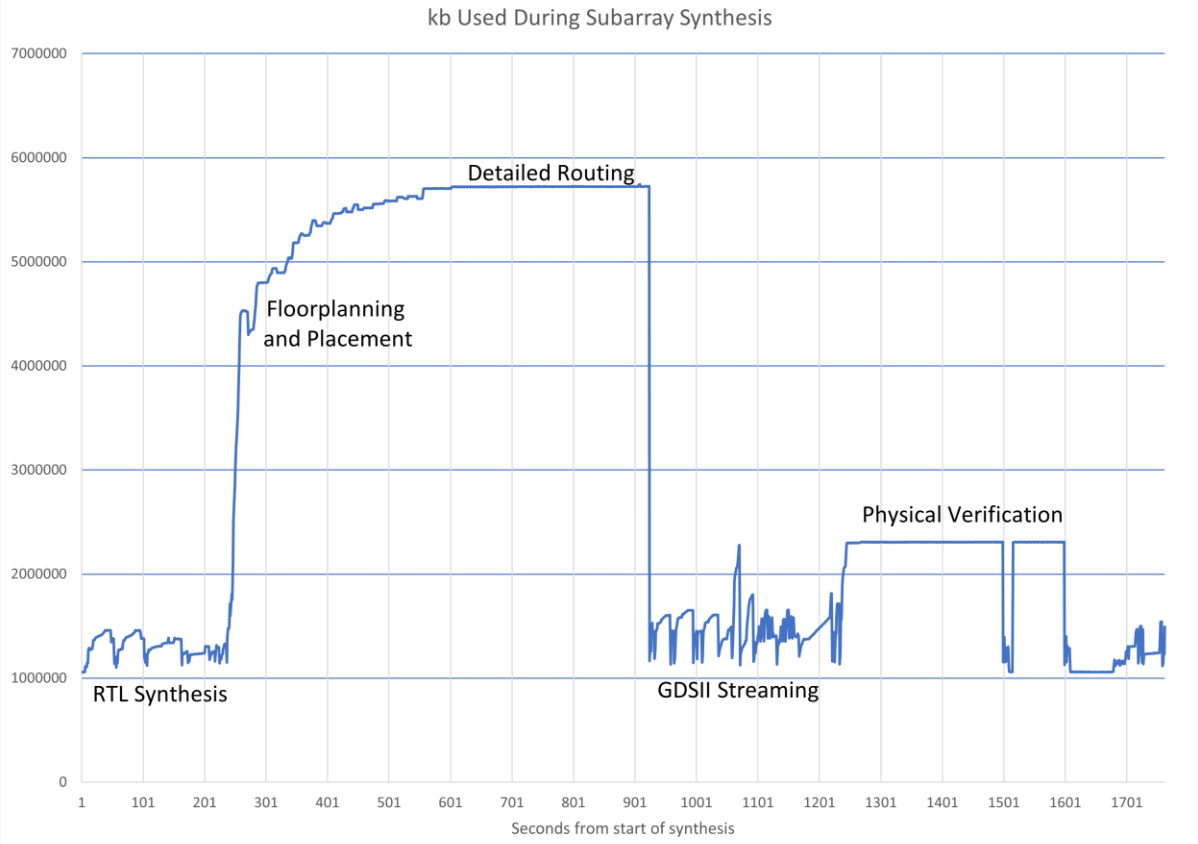
02

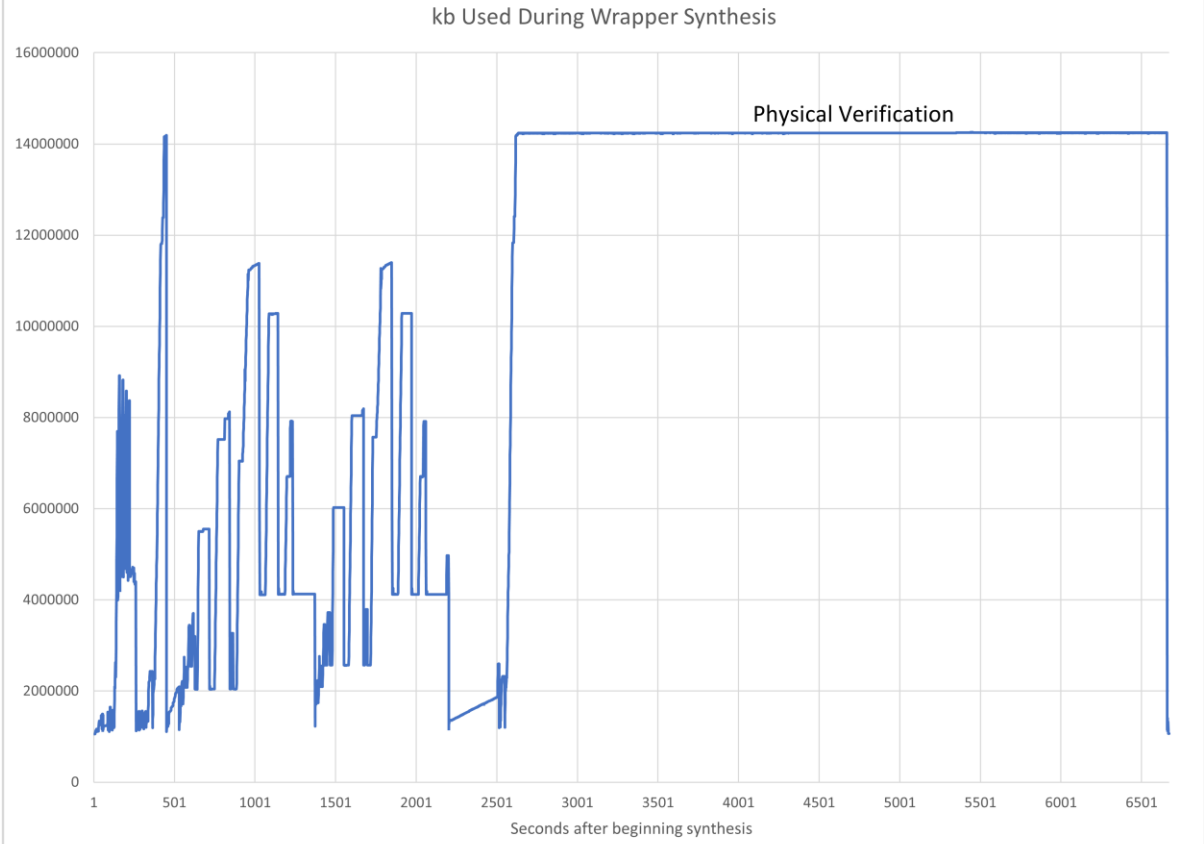
Documentation

03

Execution Time

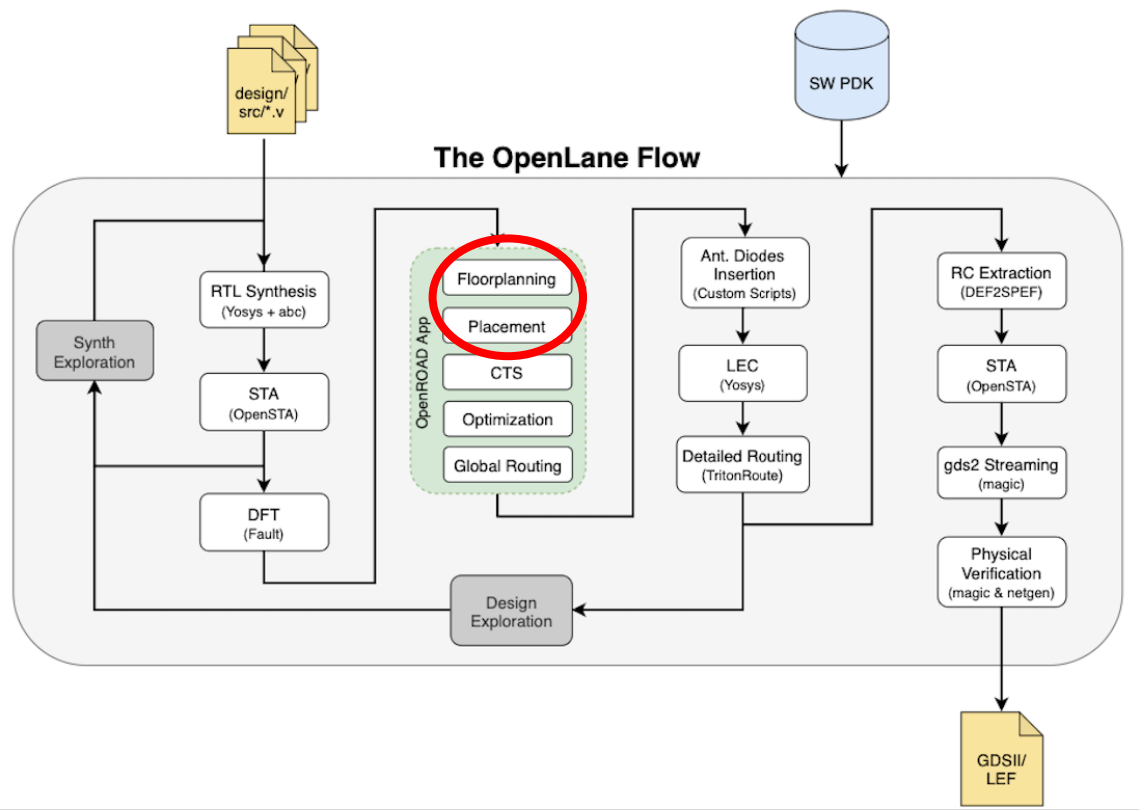
Obstacle 1: Memory Consumption





Without using subarrays, this would take even more memory!

Obstacle 2: Documentation



`FP_PDN_VOFFSET`	The offset of the vertical power stripes on the metal layer 4 in the power distribution network (Default: `16.32`)
`FP_PDN_VPITCH`	The pitch of the vertical power stripes on the metal layer 4 in the power distribution network (Default: `153.6`)
`FP_PDN_HOFFSET`	The offset of the horizontal power stripes on the metal layer 5 in the power distribution network (Default: `16.65`)
`FP_PDN_HPITCH`	The pitch of the horizontal power stripes on the metal layer 5 in the power distribution network (Default: `153.18`)
`FP_PDN_AUTO_ADJUST`	Decides whether or not the flow should attempt to re-adjust the power grid, in order for it to fit inside the core area of the design, if needed. 1=enabled, 0 =disabled (Default: `1`)

```
set ::env(VDD_NETS) [list {vccd1} {vccd2} {vdda1} {vdda2}]
set ::env(GND_NETS) [list {vssd1} {vssd2} {vssa1} {vssa2}]
set ::env(SYNTH_USE_PG_PINS_DEFINES) "USE_POWER_PINS"
set ::env(FP_PDN_MACRO_HOOKS) ".* vccd1 vssd1 vccd1 vssd1"
```

Obstacle 3: Execution Time

`gh repo clone`	`make setup`	`make chaos_subarray`	`make user_project_wrapper`
45 Seconds	1 Minute and 21 Seconds	30 Minutes	1 Hour and 45 Minutes

2 Hours and 15 Minutes per Synthesis

Conclusion

Overall, this research has proved that it is possible for a high school student to effectively use current EDA technologies, which acts as proof that more people are able to enter the Electronic Design space.

Key Links:

- Chaos Automaton Repository: https://github.com/AlexanderJGoldstein/chaos_automaton_Summer_2022
- Caravel Harness Repository: https://github.com/efabless/caravel
- Caravel User Project Repository: https://github.com/efabless/caravel_user_project
- OpenLANE Documentation: https://OpenLANE-docs.readthedocs.io/en/rtd-develop/
- WSL Documentation: https://learn.microsoft.com/en-us/windows/wsl/

Acknowledgements

- Poolesville High School Science, Math, and Computer Science Program