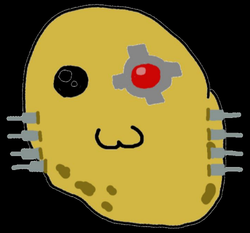


S.P.U.D.

Single-Core Processor Unit and Display

Mohamed Moussa
Computer Engineering

Frank Laterza
Computer Engineering

Gabriel Saborido
Computer Engineering

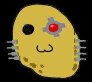
Valentina Terry
Computer Engineering

What Is Project S.P.U.D.?

An educational CPU system that transforms C programs into real-time visual output, allowing users to observe how software instructions directly translate into hardware behavior.

- Modern processors are fully optimized and difficult to inspect
- SPUD proposes a custom, modifiable, demonstrational, and accessible CPU for educational and entertainment purposes
    - can be customized in both the hardware level (Verilog) and software level (C programs)
- C programs can run directly on the FPGA-based CPU using a custom software library (spudkit) and SPUD toolchain
- CPU and System on Chip (SoC) are implemented in Verilog
- A custom matrix core drives the 64x64 RGB Matrix
- The arcade controller will be the main interface to control interactable demos

S.P.U.D. Parts



S.P.U.D. Design

64x64 LED Matrix

Arty S7 FPGA

Arcade Controller

Overall Functionality Demo

Engineering Specifications

Specifications	
Processor Performance	> 10 MIPS
Arcade Button Latency	< 30 ns
LED Matrix Refresh Rate	< 200 Hz
Functions	arithmetic, jump, and memory
FPGA Board Power Consumption	< 36 W (12V, 3A supply)
LED Matrix Power Consumption	< 20 W (5V, 4A supply)
Operational Conditions	Room Temperature: 68–77°F Humidity Level: 40-60%
Price Range	$300-$500

~ Demo 1 ~

Processor Performance (MIPS Calculation)

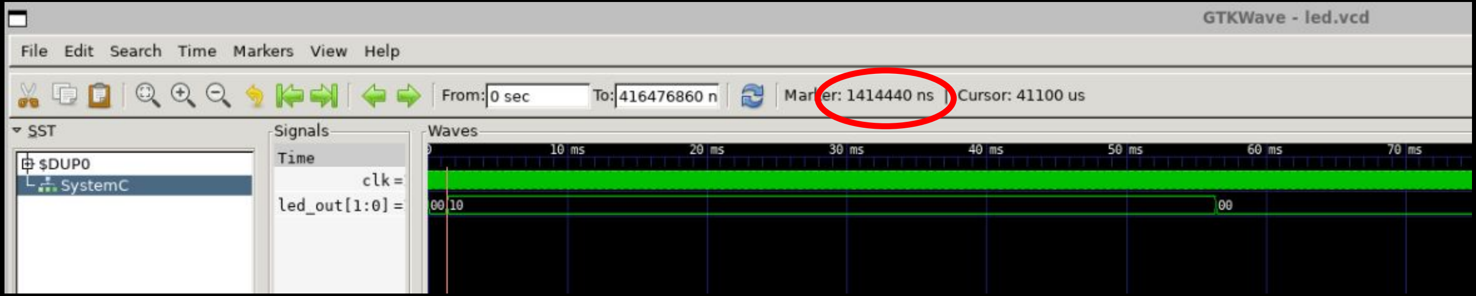
<u>Software Test</u>

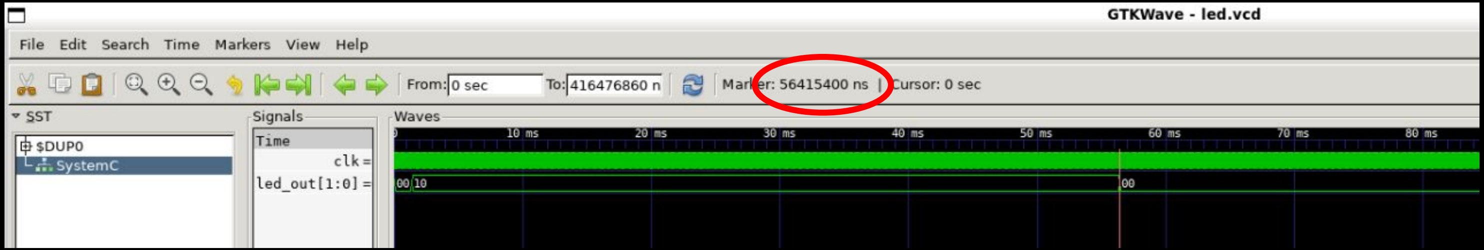
Simulation using sample c programs that toggle LED every N “nop” instructions (1M, 500k, 2M) using loop unrolling flag in compiler

<u>Hardware Test</u>

Oscilloscope connected to GPIO flag on FPGA programmed to be set anytime an LED toggles, using same sample c programs

MIPS Simulation Test - 1M instructions

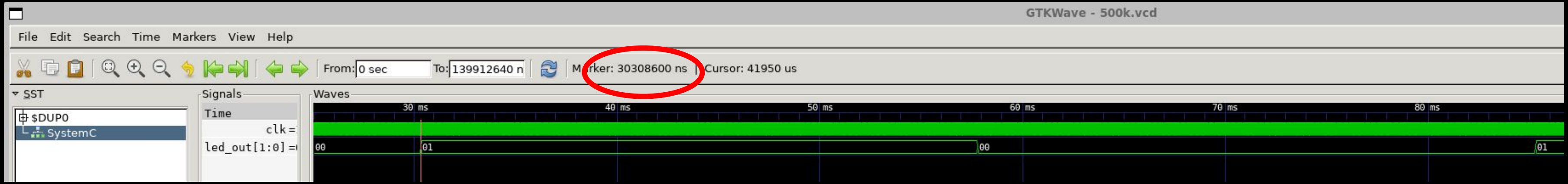


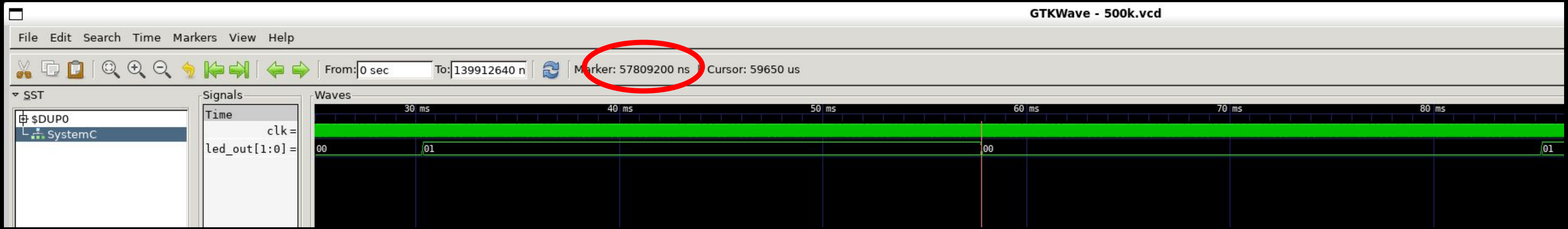


56,415,400 ns - 1,414,440 ns = 0.055 s

1M / 0.055s = 18.2 MIPS

MIPS Simulation Test - 500k Instructions

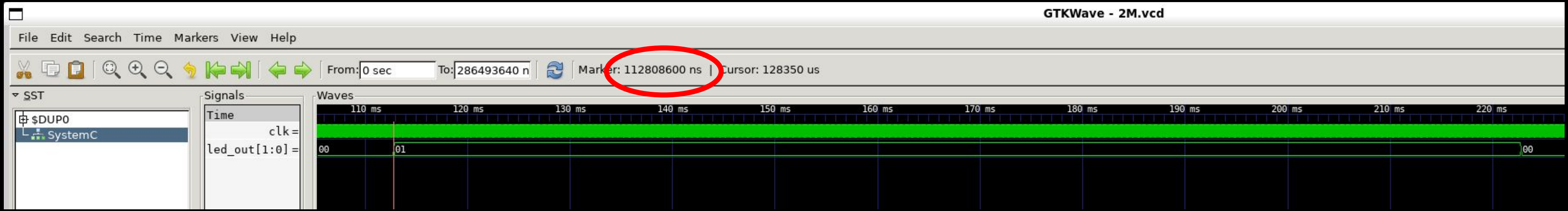


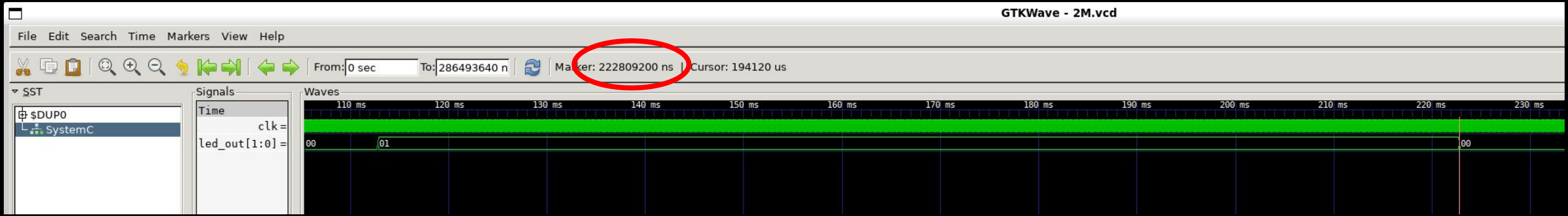


57,809,200 ns - 30,308,600 ns = 27.5 ms

500k / 27.5 ms = 18.18 = 18.2 MIPS

MIPS Simulation Test - 2M instructions





222,809,200 ns - 112,808,600 ns = 0.11 s

2M / 0.11s = 18.18 = 18.2 MIPS

MIPS Results

Test #	# of Instructions	Hardware Delay (s)	Hardware MIPS	Simulation Delay (s)	Simulation MIPS	Match
1	1M	0.055	18.2	0.055	18.2	Yes!
2	1M	0.055	18.2	0.055	18.2	Yes!
3	1M	0.055	18.2	0.055	18.2	Yes!
4	1M	0.055	18.2	0.055	18.2	Yes!
5	500k	0.028	18.2	0.028	18.2	Yes!
6	500k	0.028	18.2	0.028	18.2	Yes!
7	500k	0.028	18.2	0.028	18.2	Yes!
8	2M	0.11	18.2	0.11	18.2	Yes!
9	2M	0.11	18.2	0.11	18.2	Yes!
10	2M	0.11	18.2	0.11	18.2	Yes!

~ Demo 2 ~

LED Matrix Refresh Rate

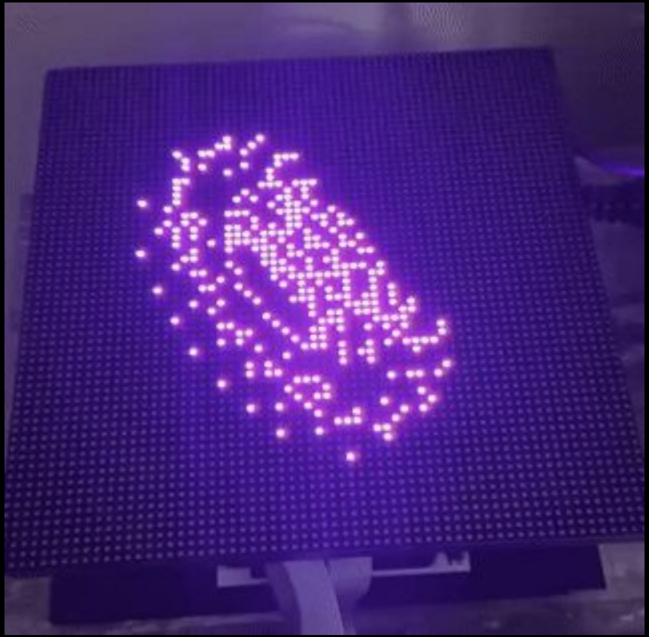
Software Test

Simulation using a C program to toggle a specified pin each full matrix display update (32 addresses cycle) and calculating the time between each toggle. We use the formula of inverse of time between each toggle (1/t) to calculate refresh rate.

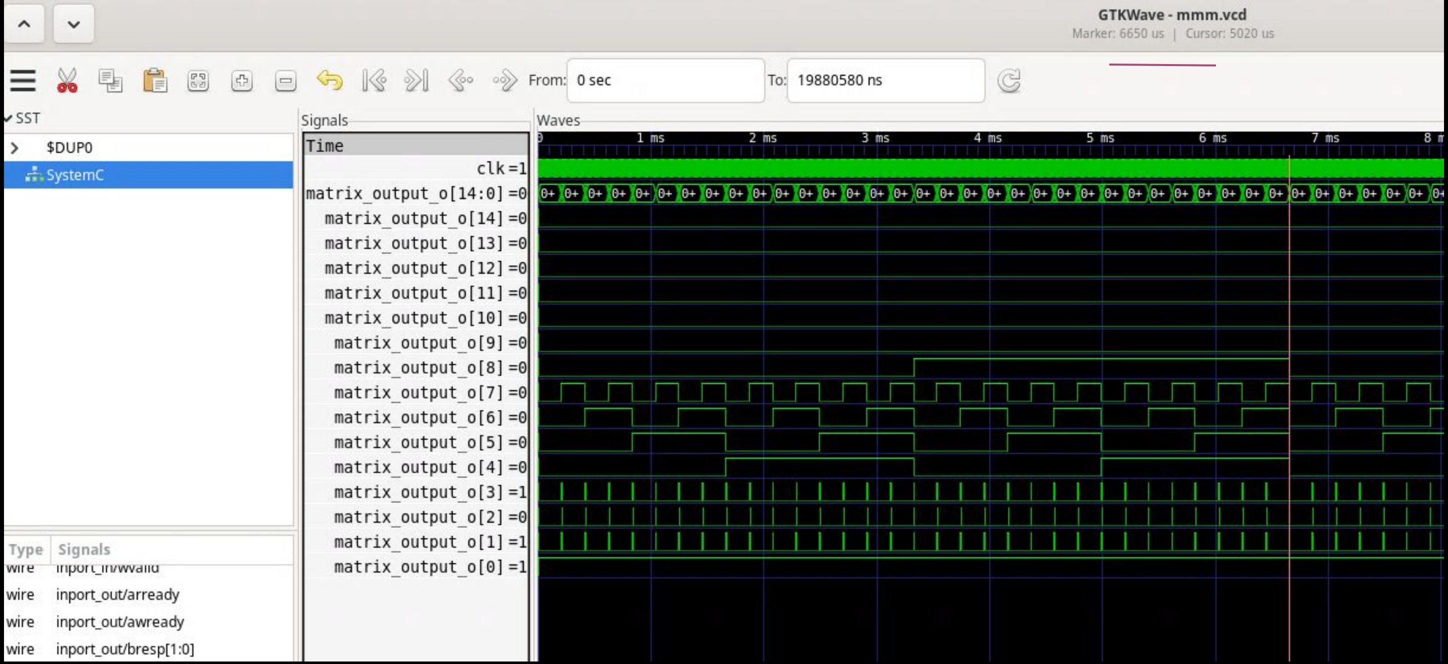
Hardware Test

Programing a similar pin on hardware and connecting an oscilloscope to said pin in order to measure the time between each toggle in real time.

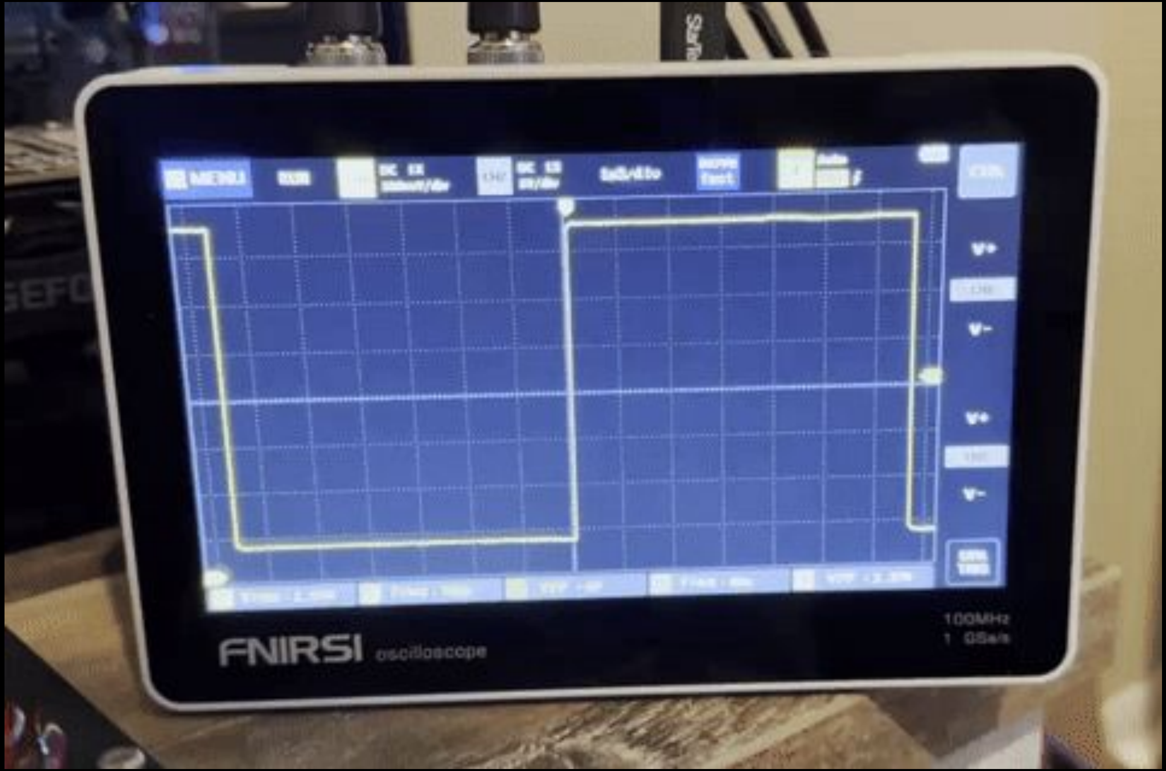
LED matrix display



Simulation Results



Oscilloscope Results



Refresh rate final results

Test #	Full display refresh rate (ms)	Simulation display refresh rate (ms)	Results matching
1	6.57	6.65	Yes
2	6.67	6.65	Yes
3	6.7	6.65	Yes
4	6.66	6.65	Yes
5	6.65	6.65	Yes
6	6.69	6.65	Yes
7	6.69	6.65	Yes
8	6.68	6.65	Yes
9	6.55	6.65	Yes
10	6.71	6.65	Yes

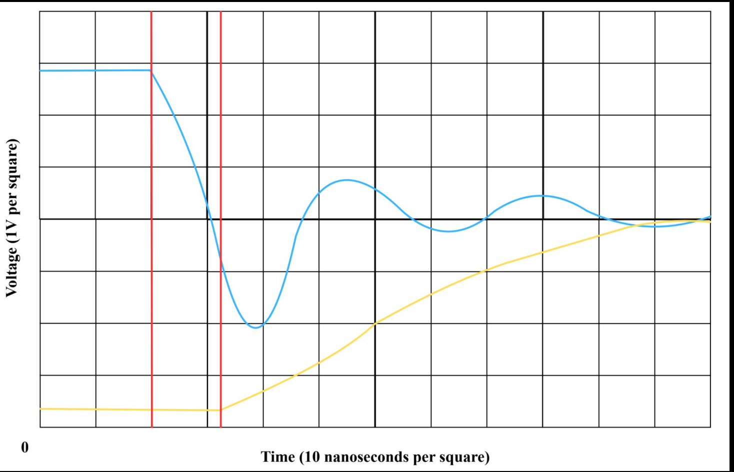
~ Demo 3 ~

Arcade Controller Button Latency

Hardware Test

Oscilloscope connected to GPIO flag on FPGA programmed to be set anytime an arcade button is pressed, and another probe connected to button input to FPGA

Button Latency Results



Blue: Button response

Yellow: FPGA response

Red: Measurement of latency

Test #	Latency (ns)	Match
1	10	Yes
2	13	Yes
3	10	Yes
4	11	Yes
5	10	Yes
6	20	Yes
7	13	Yes
8	14	Yes
9	11	Yes
10	12	Yes

Thank you!

~ The SPUD team 🥔