Giaan Nguyen

Lab 8: Systolic Arrays on FPGAs

ECEN 689-600: FPGA Information Processing Systems
Wednesday, April 7, 2021

FPGA Outputs

Using Excel functions MMULT() and DEC2HEX(), we first calculate the matrix multiplication for the matrices provided in the simulation, as shown in Figure 1.

A			B			C=A*B	
0.5	1		1	2		3.5	5
1	0.5		3	4		2.5	4
16*A			16*B			16*C	
8	16		16	32		56	80
16	8		48	64		40	64
16*A (HEX)			16*B (HEX)			16*C (HEX)	
08	10		10	20		38	50
10	08		30	40		28	40

Figure 1. Excel calculation of the expected outputs of the simulation. The top matrices represent the real values of the matrices of interest. The middle matrices represent the top matrices multiplied by 16 (to accommodate for lack of a physical decimal point). The bottom matrices represent the middle matrices as hexadecimals.

Figures 2 and 3 show the results of simulating systolic arrays using solely combinational logic ("systolic1") and combined combinational and sequential logic ("systolic2"), respectively. In both figures, we can verify that the output sequences match the hexadecimal matrix (16*C) as shown in Figure 1.

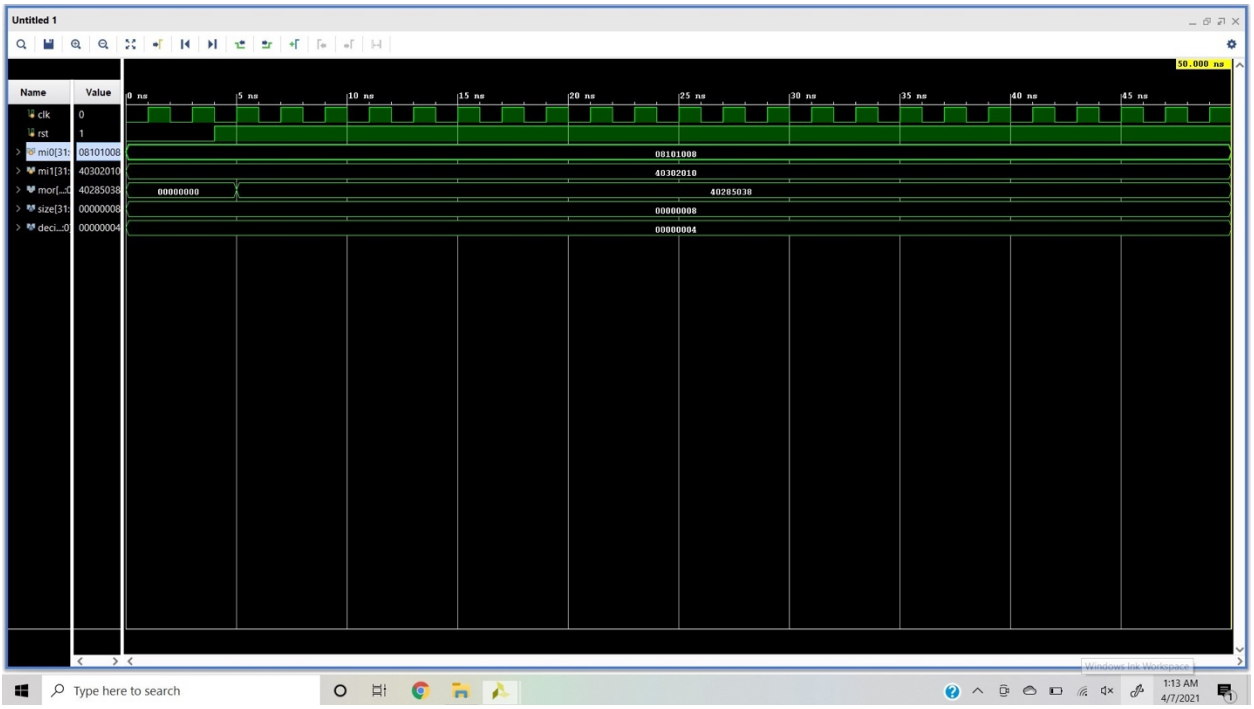


Figure 2. Simulation of "systolic1" design which uses solely combinational logic for its units.

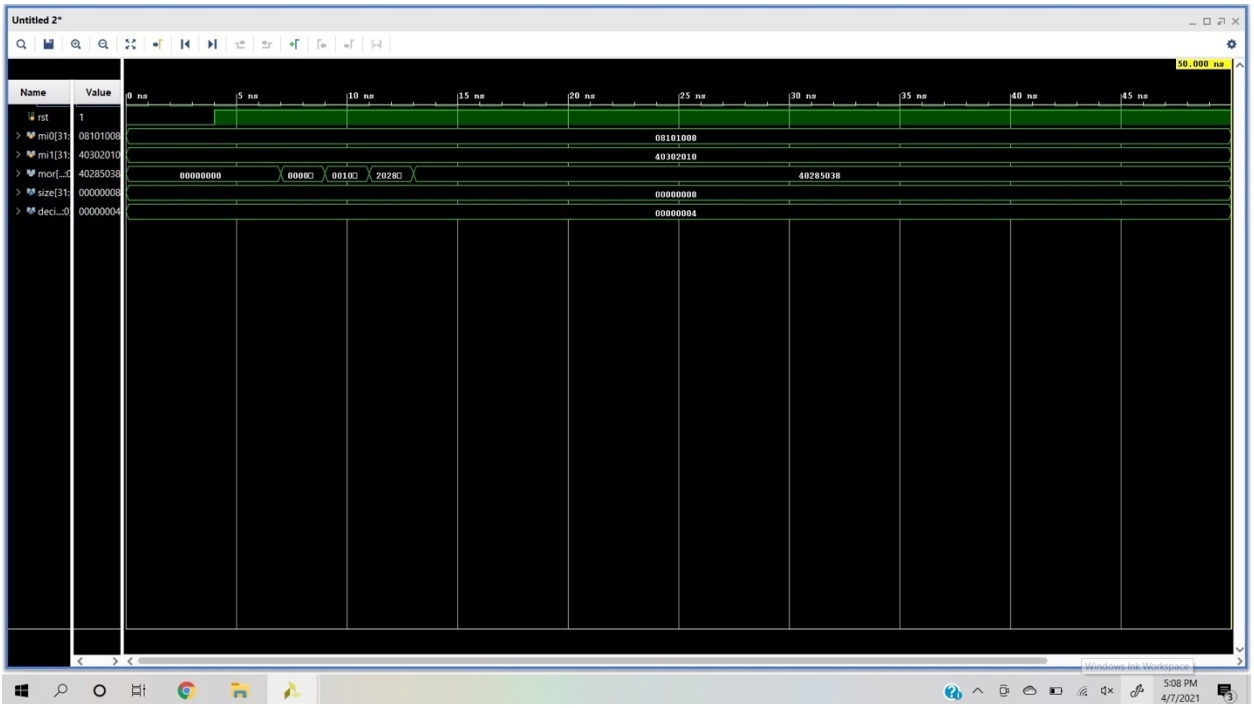


Figure 3. Simulation of "systolic2" design which uses both combinational and sequential logic in its units.

Terminal Outputs

Figure 4 shows the terminal outputs when implementing "systolic2". We can verify with a calculator that the generated matrix products are correct.

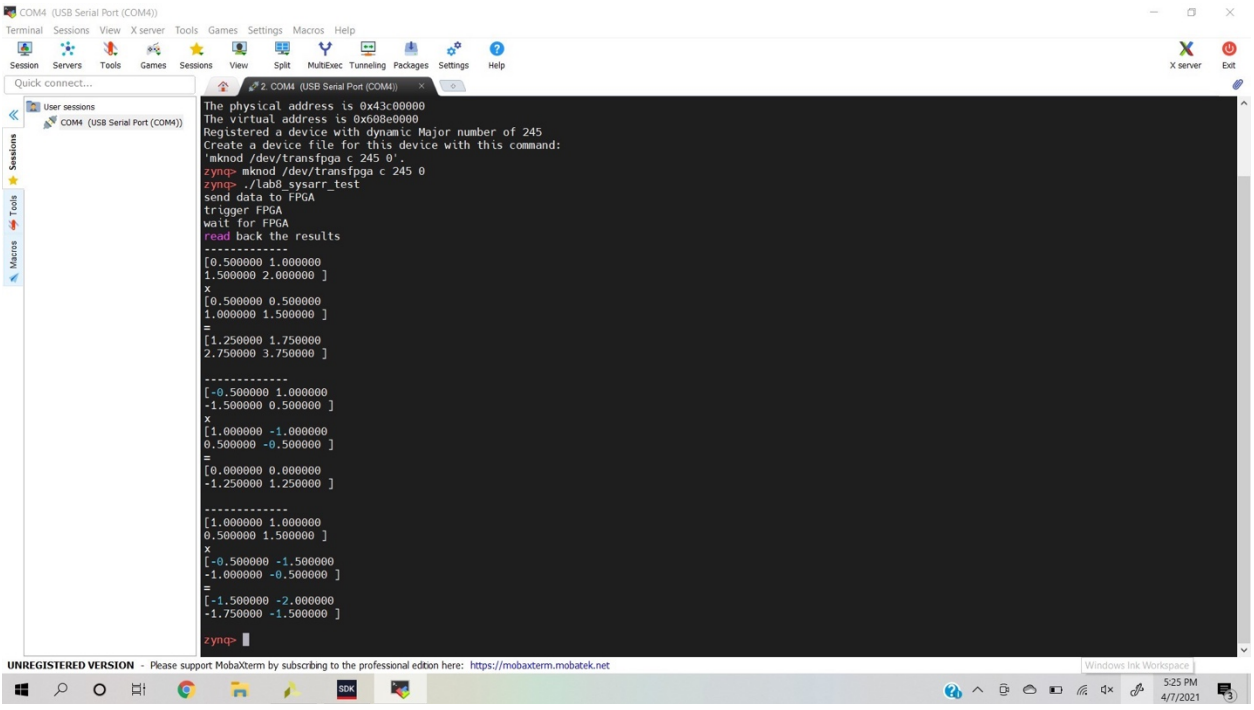
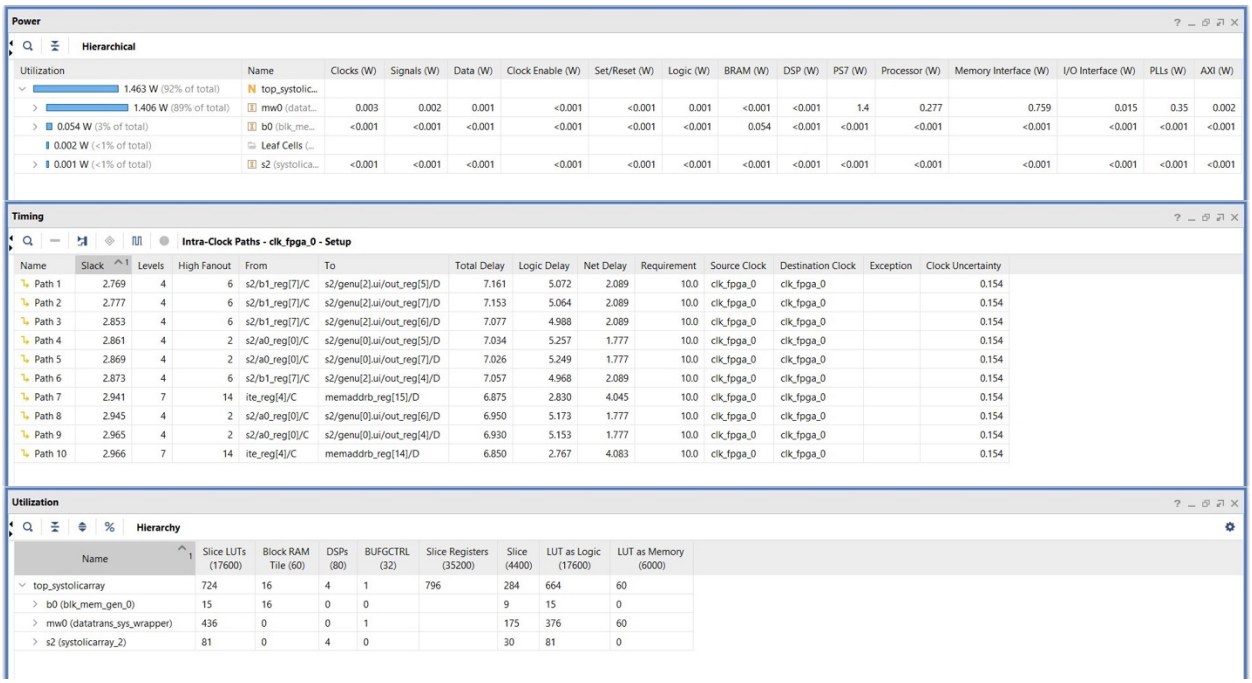


Figure 4. Terminal outputs when implementing "systolic2". Three different sets of matrices are tested and multiplied.

Implementation Summary

Figure 5 shows the power, timing, and utilization summaries for the implementation of systolic array “systolic2”.



Power

Hierarchical

Utilization	Name	Clocks (W)	Signals (W)	Data (W)	Clock Enable (W)	Set/Reset (W)	Logic (W)	BRAM (W)	DSP (W)	PS7 (W)	Processor (W)	Memory Interface (W)	I/O Interface (W)	PLLs (W)	AXI (W)
1.463 W (92% of total)	top_systolic...														
1.406 W (89% of total)	mw0 (datat...	0.003	0.002	0.001	<0.001	<0.001	0.001	<0.001	<0.001	1.4	0.277	0.759	0.015	0.35	0.002
0.054 W (3% of total)	b0 (blk_me...	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	0.054	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
0.002 W (<1% of total)	Leaf Cells (...														
0.001 W (<1% of total)	s2 (systolica...	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001

Timing

Intra-Clock Paths - clk_fpga_0 - Setup

Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	2.769	4	6	s2/b1_reg[7]/C	s2/genu[2].ui/out_reg[5]/D	7.161	5.072	2.089	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 2	2.777	4	6	s2/b1_reg[7]/C	s2/genu[2].ui/out_reg[7]/D	7.153	5.064	2.089	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 3	2.853	4	6	s2/b1_reg[7]/C	s2/genu[2].ui/out_reg[6]/D	7.077	4.988	2.089	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 4	2.861	4	2	s2/a0_reg[0]/C	s2/genu[0].ui/out_reg[5]/D	7.034	5.257	1.777	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 5	2.869	4	2	s2/a0_reg[0]/C	s2/genu[0].ui/out_reg[7]/D	7.026	5.249	1.777	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 6	2.873	4	6	s2/b1_reg[7]/C	s2/genu[2].ui/out_reg[4]/D	7.057	4.968	2.089	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 7	2.941	7	14	ite_reg[4]/C	memaddrb_reg[15]/D	6.875	2.830	4.045	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 8	2.945	4	2	s2/a0_reg[0]/C	s2/genu[0].ui/out_reg[6]/D	6.950	5.173	1.777	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 9	2.965	4	2	s2/a0_reg[0]/C	s2/genu[0].ui/out_reg[4]/D	6.930	5.153	1.777	10.0	clk_fpga_0	clk_fpga_0		0.154
Path 10	2.966	7	14	ite_reg[4]/C	memaddrb_reg[14]/D	6.850	2.767	4.083	10.0	clk_fpga_0	clk_fpga_0		0.154

Utilization

Hierarchy

Name	Slice LUTs (17600)	Block RAM Tile (60)	DSPs (80)	BUFGCTRL (32)	Slice Registers (35200)	Slice (4400)	LUT as Logic (17600)	LUT as Memory (6000)
top_systolicarray	724	16	4	1	796	284	664	60
b0 (blk_mem_gen_0)	15	16	0	0		9	15	0
mw0 (datatrans_sys_wrapper)	436	0	0	1		175	376	60
s2 (systolicarray_2)	81	0	4	0		30	81	0

Figure 5. Implementation summary for “systolic2” implementation: power (top), timing (middle), and utilization (bottom).

Questions

1. **What applications are systolic arrays used for?**

Generally, systolic arrays can be used to perform operations that rely on multiplication and accumulation, such as matrix multiplication, convolution, and correlation. The latter two operations are key concepts in image processing for which systolic arrays can be used for (i.e., spatial filters in real-time). Additionally, from the first two operations, systolic arrays can be used in hardware implementation of neural networks.