Multicycle datapath in Logisim

General structure



Sub-components

The "Datapath" circuit is made of various sub-components.

You see the various components of the datapath using the menu on the left.

If you click on them you can see the circuit.

🖻 MIPS_MultiCycle
– 🔂 Datapath
- 🔂 ALU
- 🔂 ALUControl
– 🔂 RegisterFile
- 🔂 Memory
- 🖬 SignExtend
– 🖬 ShiftL2_32bit
- 🖬 ShiftLeft26BitTo28
– 🔂 ControlUnit
- 🖸 OverflowDetect
– 🖸 NextStateControlUnit
– 🛄 OutputControlUnit

Load a program (1)

Go in the "Datapath" circuit, right click on "Memory" -> "View Memory" Right click on RAM -> "Load image.." and select the .hex file to store in the memory





Load a program (2)

You can also manually insert instructions and data in hexadecimal.

WARNING: the addresses in the memory are word aligned, you cannot address by bytes

ł	4	0.)	RAM 1	6M x 32		
ł	L	22 A	1677721	5		
•		M2 [Write en	able]	0.000	
-		CI				
-		<u> </u>			<u> </u>	
		A;1;2	000000	00000000	A,1	Me
1		A;1,2	000001	00000000	· A;1	f1
	-	A;1;2	000002	00000000	· · · A;1	4
	-	A;1;2	000003	00000000	· A;1	3
	-	A.1.2	000004	00000000	· A 1	•
		A-1-2-	000005	00000000		2
	· e	A,1,2	000006	00000000	A,1	6
	. 7	A;1;2	000007	00000000	· A;1	,
1		A;1,2	000008	00000000	A,1	
	÷	A;1,2	000009	00000000	A,1	H
1	-	A;1,2	00000a	00000000	· A,1	2
	10	A:1.2	000006	00000000		1.6
1	11	4.3.2	00000c	Delete		
	12	A,1,2	00000d	Show A	Attrib	utes
	13	A;1,2	00000e	E 10 C		
1	-	A;1;2	00000f	Edit Co	nten	ts
1	14	A;1;2	000010	Clear C	onte	nts
	15	A,1,2	000011	Loadin	anen	
	16	A1.2	000012	LUau III	naye	
	17	A-1-2-	000013	Save In	nage	
	18	A,1,2	000014	00000000		18 10 10 10
1	1.	A;1,2	000015	00000000	A,1	
1		A;1,2	000016	00000000	A;1	1
	20	A;1;2	000017	00000000	· A,1	-
1	21	A;1;2	000018	00000000	· A;1	23
	22	A:1.2	000019	000000000	A:1	2
	23		ouuula	000000000		
- 4	_	A-1-2 ···	000034	00000000		23
	24	A;1,2	00001b	00000000	A;1	2)
A COLUMN A	.24 .25	A;1;2 A;1;2	00001b	00000000	A,1	23
A REAL PROPERTY OF	1 2 2 2	A;1,2 A;1,2 A;1,2	00001b 00001c 00001d		A,1 A,1 A,1	2) 23
A DESCRIPTION OF A DESC	, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,	A:1,2 A:1,2 A:1,2 A:1,2	00001b 00001c 00001d 00001e	000000000000000000000000000000000000000	A,1 A,1 A,1 A,1	23 23 23
A REAL PROPERTY OF A REAL PROPER	A R R R	A,1,2 A,1,2 A,1,2 A,1,2 A,1,2	00001b 00001c 00001d 00001e 00001f	000000000000000000000000000000000000000	A,1 A,1 A,1 A,1	23 23 25
		A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2	00001b 00001c 00001d 00001e 00001f 000020	000000000000000000000000000000000000000	A,1 A,1 A,1 A,1 A,1 A,1	23 23 23 23
And a first of the start of the start of the start of the start	, *, *, *, *, *, *, *, *, *, *, *, *, *,	A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2	00001b 00001c 00001d 00001e 00001f 000020 000021	000000000000000000000000000000000000000	A,1 A,1 A,1 A,1 A,1 A,1	23 23 23 23 23
and the second state state and a second state of the] X] X] X] X] X] X] X]	A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2 A,1,2	00001b 00001c 00001d 00001e 00001f 000020 000021 000022	000000000000000000000000000000000000000	A,1 A,1 A,1 A,1 A,1 A,1 A,1	23 23 23 23 23 23 23 23 23 23 23
and the second se		A(1,2 A(1,2 A(1,2 A(1,2 A(1,2 A(1,2 A(1,2 A(1,2 A(1,2) A(1,2)	00001b 00001c 00001e 00001f 000020 000021 000022 000023		A(1) A(1) A(1) A(1) A(1) A(1) A(1) A(1)	23 23 23 23 23 23 23 23 23 23 23 23 23 2
The state of the s		A(1,2) A(1,2) A(1,2) A(1,2) A(1,2) A(1,2) A(1,2) A(1,2) A(1,2) A(1,2) A(1,2)	00001b 00001c 00001e 00001f 000020 000021 000022 000023 000024		A,1 A,1 A,1 A,1 A,1 A,1 A,1 A,1 A,1 A,1	23 23 23 23 23 23 23 23 23 23

Window Help

00002 100000 0 0000000 0000000 0000000 0000000 0 0x0 00000000 00000 0000400 000050 000060 00000000 0

Execute a program

Verify that "Run simulation" is ticked.

Click on "Tick Full Cycle" (or press F9) to do one clock cycle.

You can verify the datapath steps by clicking on the components in the "simulate" menu

	Simulate	FPGA	Window	Help										
e	Run Sin	nulator		Ctrl-E										
1	Step Si	Ctri-I												
	Reset S	Ctrl-R												
	VHDL Simulation Enabled													
	Restart	VHDL :	simulator											
	Go Out	To Sta	te	•										
	Go In T	o State		•										
_	Tick Ha	If Cycle		Ctrl-T										
1	Tick Fu	ll Cycle		F2										
	Ticks E	nabled		Ctrl-K										
	Tick Fre	equenc	у	•										
	Chrono	gram												
	Test Ve	ector												
	Assem	oly viev	ver											



Watching registers values

You can watch the current values of the registers in the "state" tab, in the bottom left corner of logisim.

Properties State			
RegisterFile	r20	00000000	
RegisterFile	r21	00000000	
RegisterFile	r22	00000000	
RegisterFile	r23	00000000	
RegisterFile	r24	00000000	
RegisterFile	r25	00000000	
RegisterFile	r26	00000000	
RegisterFile	r27	00000000	
RegisterFile	r28	00000000	
RegisterFile	r29	00000000	
RegisterFile	r30	00000000	
RegisterFile	r31	00000000	
main	A	000000f	
main	ALUOut	00000010	
main	В	00000004	_
main	Cause	00000000	
main	EPC	00000000	
main	InstructionRegister	000000f	
main	MemoryDataRegister	000000f	
main	PC	0000010	-

Timing diagrams

You can also watch the evolution of the various signals and registers clock cycle by clock cycle.

In the "simulate" menu click con "Timing Diagram"

Select CLK as the clock source, then select what signals you want to watch.

After this steps you can tick the clock and watch the signals evolve

SignalName	SignalValue	0 ns	20.0 ps	40.0 µs	60.0 µs	80.0 µs	100.0 µs	120.0 µs	140.0 µs	160.0 µs	180.0	8495
⊡ CLK	0	0 1	0 1 0 1 0	1 0 1 0	1 0 1 0	1 0 1 0	1 0 1 0	1 0 1 0	1 0 1 0	1 0 1 0	1 0	
f RegFile/r8[310]	15	0		15								
🛱 RegFile/r9[310]	4	0					4					
🗗 RegFile/r10[310]	19	0							19			

Tips & Tricks for modifying the datapath

Control Unit (1)

The control unit is made from 2 combinational circuits and a register.

The first combinational block outputs the next state based on the current state, the opcode and the overflow signal.

The next state gets written to the StateRegister on every rising edge.

The output of the StateRegister is also used by the second combinational block to decide the outputs signals.



Control Unit (2)

The 2 combinational blocks are generated by a Logisim Evolution tool, that allows us to create a circuit based on its truth table

You can import truth tables from the .txt files in the repo (the same is true for the ALU Control)

1	Window	Help								
	Minimiz	ze	Ctrl-M							
_	Maximi	ze								
	Close Ctrl-W									
	• Combin	• Combinational Analysis								
	• Prefer	ences	15							
	. MIPS M	IultiCycle								



Control Unit (3)

In the "Table" tab you can see the truth table of the circuits.

The bit indicated by "-" are "don't cares".

CurrentState[30]	Op[50]	Overflow
0000		

Control Unit (4)

You have to modify both the "NextStateControlUnit" and the "OutputControlUnit" truth tables and generate the new circuits.

Import the .txt, modify the truth table as needed and then click "Build Circuit". Enter the same name and overwrite the previous circuit. You can also save the modified truth table by clicking on "Export Table".

ALUSrcA[1..0]

Modifying the width of a signal (1)

You might need to modify the numbers of bit of a signal.

Go in the "Inputs & Outputs" tab, double click on the signal you want to modify and select the number of bits.

If you are adding bits remember to insert "0" where there are "-" in the new bits.

PCWriteCond	
PCWrite	
lorD	
MemRead	
MemWrite	
MemtoReg	
IRWrite	
CauseWrite	
IntCause	
EPCWrite	
PCSource[10]	
ALUOp[10]	
ALUSrcA	2 bits
ALUSrcB[10]	
RegWrite	
RegDst	
NextState[30]	
Click to add a new variable	

Modifying the width of a signal (2)

When you modify the width of the signals you also need to modify the components that use the signals.

Verilog								
Verilog								
East								
Wide								
Top/Right								
2								
32								
Zero								
No								
	East Wide Top/Right 2 32 Zero No							

Selectio	n: Tunnel "ALUSrcA"						
VHDL	Verilo	g					
Facing	South						
Data Bits	2	-					
Label	ALUSICA						
Label Font	SansSerif Bold 16						

Add control signals (1)

To add a new control signals go in "Inputs & Outputs", click "Add a new variable", write the name of the signal and select the number of bits.

Output Variables	
PCWriteCond	-
PCWrite	
lorD	
MemRead	
MemWrite	
MemtoReg	
IRWrite	
CauseWrite	
IntCause	
EPCWrite	
PCSource[10]	
ALUOp[10]	
ALUSrcA[10]	
ALUSrcB[10]	
RegWrite	
RegDst	
NextState[30]	
Click to add a new variable	

Add control signals (2)

After building the new circuits you can find the signal in the "Datapath" circuit.

Add a "Tunnel" to use the signal inside the datapath.

· · · · · · · · · · · · · · · · · · ·						L													•	82		~	•									
			· Testsia						Testsia				Tostsia		Tostsia				atgia				9	TestSig						2	-	
Ì	C	oI	nt	r	0	1	U	n	it	TC		· L		ĽŸ		•			34 32		•		9			- - -		<u>.</u>			Ļ	
																·	·	•	•	·	•	٠	•		•	•	8			•		•
			5		2		5		2		3	÷	2	•	8		23		3		2		3		2		5		20		12	

You can find everything on my github: <u>https://github.com/fdila/MIPS-multicycle-datapath</u>

Open a issue if you find something wrong, and Pull Requests are also welcome!