Class:	**CPE300L 1001**	Semester:	**Fall 2024**

Points		Document author:	**Alireza Bolourian**
		Author's email:	**bolouria@unlv.nevada.edu**
		Document topic:	**Postlab 10**

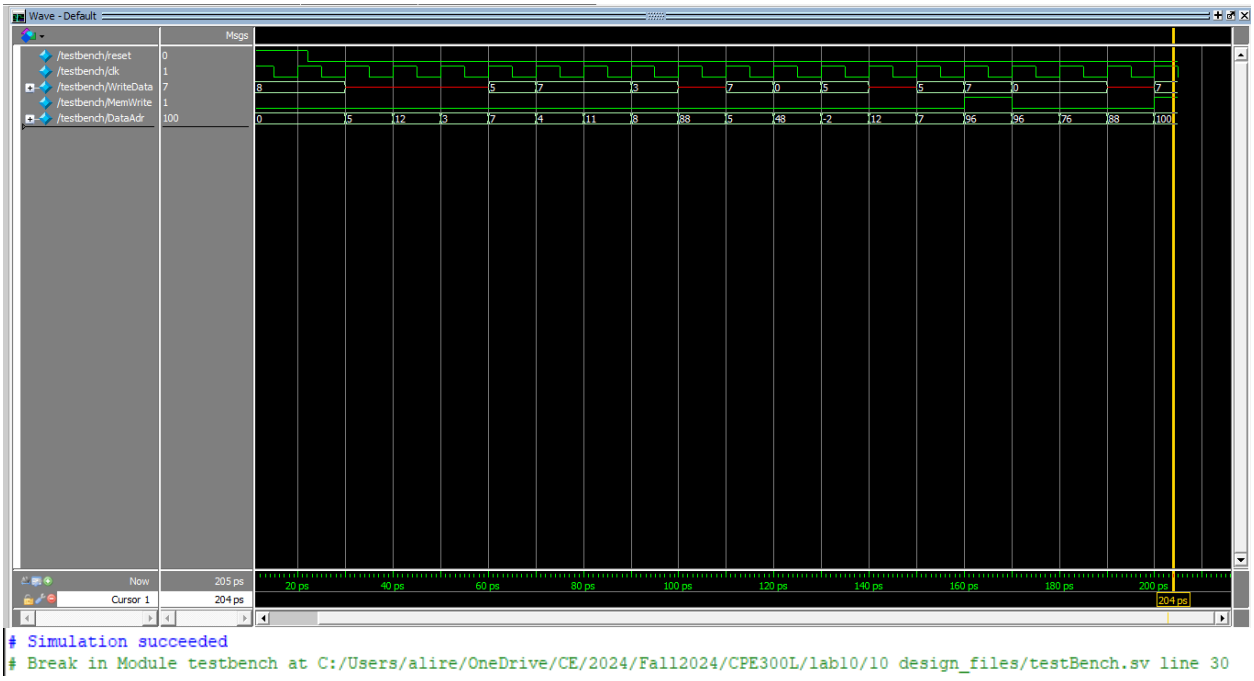
Instructor's comments:

1. Theory of Operation

In this lab, I learn about a different architecture called ARM. Even though the logistical architecture looks the same, there are differences in the details of the process such as being limited to only 16 general-purpose registers with R15 being the Program Counter register. ARM also includes flag signals inside the ALU which allows instructions to be conditionally executed without the need for branch instructions.

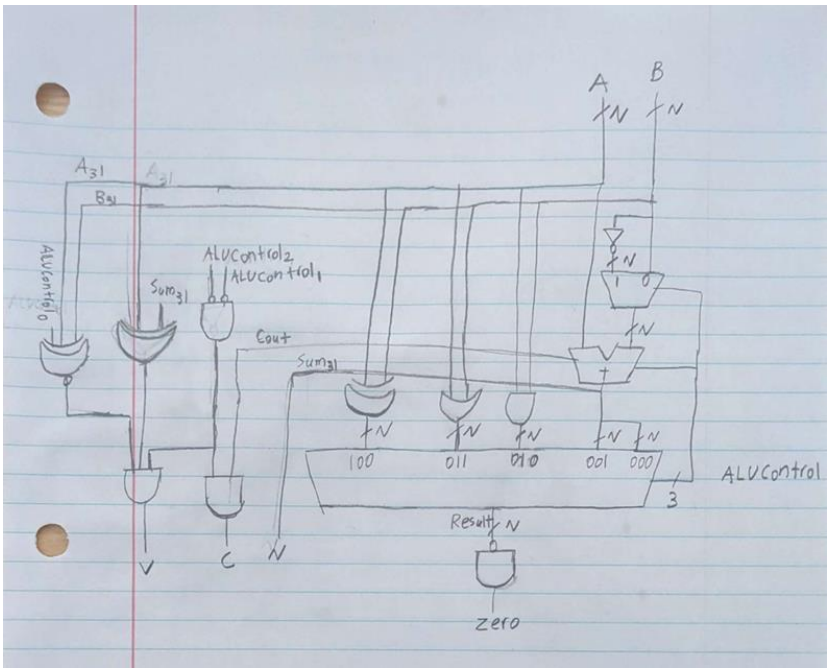
2. Experimental Results

1.



The waveform and the console output show that the value of 7 is in the address 100 as expected from the machine code.

2. EOR:



The multiplexer inside ARM ALU was expanded to select the XOR operation output.

Table 2. Extended functionality: ALU Decoder

ALUOp	***Funct*$_{4:1}$ *(cmd)***	***Funct*$_0$ *(S)***	**Notes**	***ALUControl*$_{2:0}$**	***FlagW*$_{1:0}$**
0	X	X	Not DP	000	00
1	0100	0	ADD	000	00
		1			11
	0010	0	SUB	001	00
		1			11
	0000	0	AND	010	00
		1			10
	1100	0	ORR	011	00
		1			10
1	0001	0	EOR	100	00
		1			10

One more instruction was added to indicate the XOR operation (Funct[4:1] = 0001) which decodes to ALUControl[2:0] = 100.

LDRB:

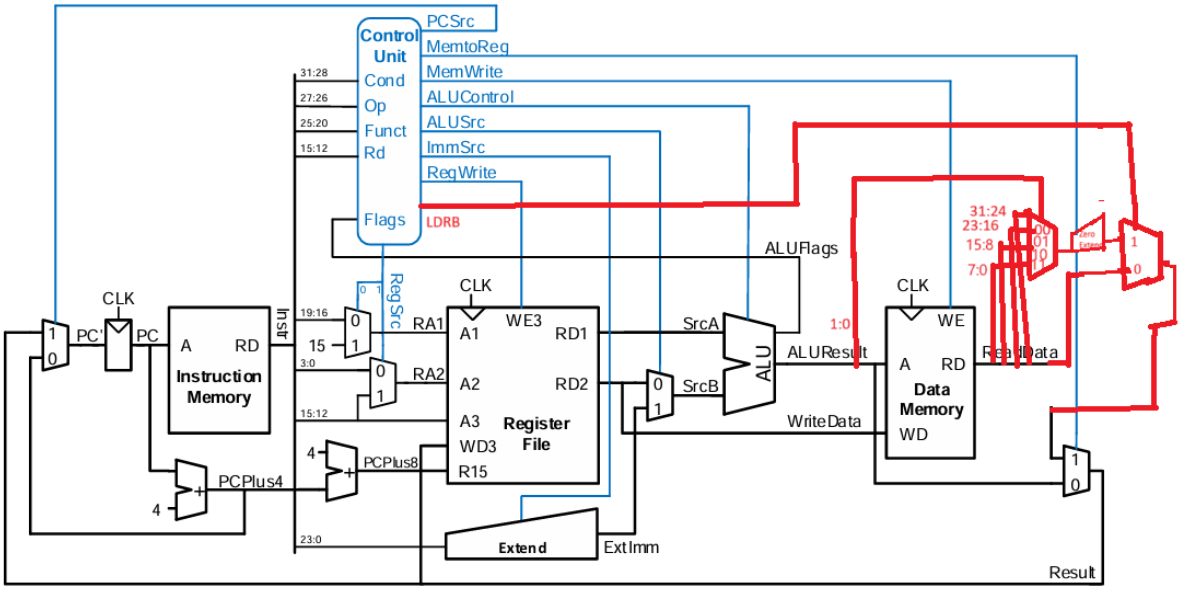


Figure 1. Single-cycle ARM processor

A multiplexer was used to select the desired byte indicated by the ALUResult[1:0]. It is then zero extended and can be selected from the full-length data if LDRB is asserted.

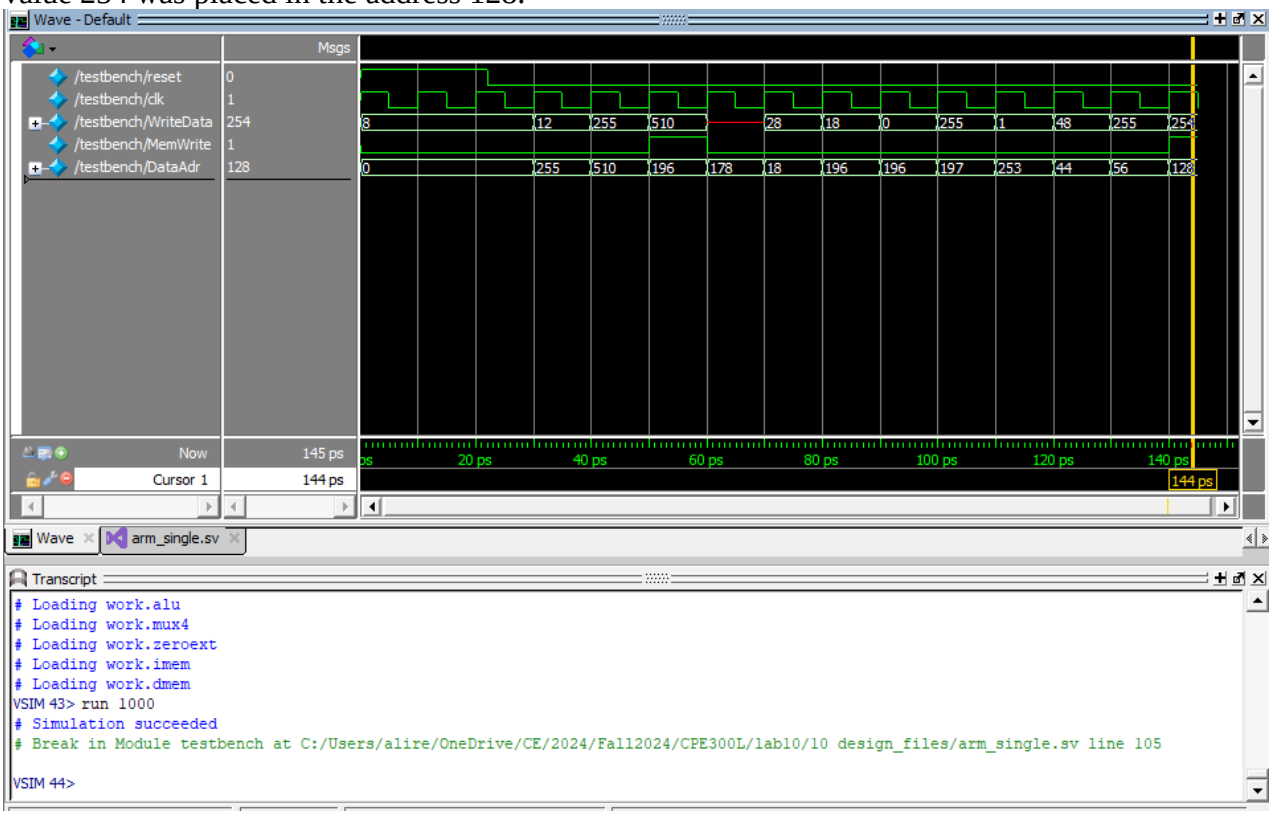
Table:

Cycle	reset	PC	Instr	SrcA	SrcB	Branch	AluResult	Flags$_{3:0}$ [NZCV]	CondEx	WriteData	MemWrite	ReadData
1	1	00	SUB R0, R15, R15 E04F000F	8	8	0	0	?	1	8	0	x
2	0	04	ADD R2, R0, #5 E2802005	0	5	0	5	?	1	x	0	x
3	0	08	ADD R3, R0, #12 E280300C	0	C	0	C	?	1	x	0	x
4	0	0c	SUB R7, R3, #9 E2437009	C	9	0	3	?	1	x	0	x
5	0	10	ORR R4, R7, R2 E1874002	3	5	0	7	?	1	x	0	x
6	0	14	AND R5, R3, R4 E0035004	C	7	0	4	?	1	x	0	x
7	0	18	ADD R5, R5, R4 E0855004	4	7	0	B	?	1	x	0	x
8	0	1c	SUBS R5, R5, #10 E255500A	B	A	0	1	0000	1	x	0	x
9	0	20	SUBSGT R5, R5, #2 C2555002	1	2	0	-1	1000	1	x	0	x
10	0	24	ADD R5, R5, #12 E285500C	-1	C	0	B	1000	1	x	0	x
11	0	28	SUBS R8, R5, R7 E0558007	B	3	0	8	0000	1	x	0	x
12	0	2c	BEQ END 0A00000C	34	30	1	64	0000	0	x	0	x
13	0	30	SUBS R8, R3, R4 E0538004	C	7	0	5	0000	1	x	0	x
14	0	34	BGE AROUND AA000000	3C	0	1	3C	0000	1	x	0	x
15	0	3c	SUBS R8, R7, R2 E0578002	3	5	0	-2	1000	1	x	0	x
16	0	40	ADDLT R7, R5, #1 B2857001	B	1	0	C	1000	1	x	0	x
17	0	44	SUB R7, R7, R2 E0477002	C	5	0	7	1000	1	x	0	x
18	0	48	STR R7, [R3, #224] E58370E0	C	E0	0	EC	1000	1	7	1	x
19	0	4c	LDR R2, [R0, #236] E59020EC	0	EC	0	EC	1000	1	x	0	7

The modified Verilog code corresponding to the changes has been uploaded separately.

3.

The code was converted into machine code using the ARM simulator (CPUlator ARMv7 System Simulator) and it was placed in the memfile2.dat file. The testbench was used to confirm that the value 254 was placed in the address 128.



3. Questions

1. What is specific to ARM architecture, compared to other ones?

Compared to MIPS, ARM ALU can set flags which allows the ARM architecture to conditionally execute instructions. This reduces the need for branch instructions which minimizes pipeline stalls. Compared to x86, ARM is a RISC architecture leading to lower power consumption.

2. List 3 advantages of ARM architecture

1. ARM architecture is a RISC architecture and so it is highly energy-efficient, making it ideal for battery-powered devices.
2. Conditional execution improves performance by minimizing pipeline stalls.
3. RISC architecture leads to faster execution of instructions and since ARM has Load/Store architecture, operations are performed on registers which simplifies instructions and improves performance.

4. Conclusions

In this lab, I became familiarized with the ARM architecture and increased the functionalities of the limited instructions set. I realize this architecture is harder to understand compared to MIPS, but it has interesting features to make it unique and especially when used along with pipelining. I understand why ARM is being used in smart phones and laptops to provide better battery life. Thumb and Thumb-2 further optimize ARM for embedded systems and mobile devices. Although efficient, ARM processors lack the performance of x86 processors in high performance and complex computational tasks.