# COL216 Assignment 2

## Stage 1

Rishabh Dhiman 2020CS10837

9 February 2022

# 1 Objective

Construct an ALU, a register file, a primary memory unit and a data memory unit for an ARM processor in VHDL.

## 2 Technical Details

- The VHDL code was compiled and simulated using GHDL 1.0.0.
- The waveform viewer used is GTKWave Analyzer v3.3.104.

#### 3 Documentation

The submission contains four VHDL files defining the various units,

- types.vhdl,
- alu.vhdl,
- reg\_file.vhdl, and
- memory.vhdl.

Along with four testbenches for testing these units

- alu\_testbench.vhdl,
- reg\_file\_testbench.vhdl,
- primary\_memory\_testbench.vhdl, and

#### • data\_memory\_testbench.vhdl.

Along with an output folder, containing the waveforms on simulating the testbenches, in the form of .ghw and .pdf files.

#### types.vhdl

This files defines some custom types for words, half-words and bytes along with an enumerated type for opcodes. It's taken from Piazza post 163, with minor modifications.

#### alu.vhdl

It defines the ALU satisfying the specifications given in the problem statement.

#### reg\_file.vhdl

It defines the register file satisfying the specifications given in the problem statement.

#### memory.vhdl

It defines the primary memory and data memory satisfying the specifications given in the problem statement.

Since the contents primary memory had to be defined in the declaration itself, I decided to define it such that the entire memory contains 0, barring the index 1 which contains 0x0000FFFF.

## 4 Tests and Results

It was simulated using ghdl, the waveform was generated only till at most time 2500ns.

The output waveform of the testbench results can be viewed in .ghw and .pdf files.

The .pdf files don't contain the entire waveform due to a lack of space, however, the .ghw files can be opened in any waveform analyzer like GTKWave to view the output waveform. The pdf files are also attached at the end of this report.

Timo	1					10 113					N Is				1	O is				40 tis				Į.	l nis			
Time carry_in=0															т ,													
instr=andop	andop	Yenr	Ysuh	Yrsh	<b>Y</b> add	Yado	Yshr	Yrsc	Ytst	<b>V</b> ten	Yomp	Yemn	Yorr	mov	Ynic	Ymvn Ya	ndon Yeo	or Ye	ub <b>(</b> rsb	Yadd	Yado	Vshc	Yrsc	Ytst	Yten	Yemp	Yemn Yor	r <b>Y</b> mov
carry_out=0	anaop	VCOT		Vran	Λααα	Nauc	Vonc	VESC	Vene	Necd	Citio	Voint	Vott	VIIIO V	Vote	V::: Vo	maop Neo	01 <b>V</b> 0	m Vran	Nauu	Mauc	Vonc	Vrac	Vene	Vecd	/cmp	Voints Vor	T VIIIO
	PFFFFFF	,																										
	FFFFFFF																											
result[31:0]=FFFFFFFF			00		FFFFFFF	E	FFFFFF	PPP		(00000000		FFFFFFF	FFFFFFFF		(00000000	(F	PFFFFFF (00	0000000		PPPPPPP	PE (PPPPPP	FF <b>(</b> 0000000	00	FFFFFFFF	000000	100	FFFFFFF FF	FFFFFF
																1												

Me d_addr[5:0]				2040 rs					2150 ns	1/00	V02	V0.4	Voc	2060 ns	Von	Var
d_addr[5:0] te_addr[5:0]	<b>X</b> 39	<b>X</b> 3A	<b>X</b> 3B	<b>X</b> 3C	<b>X</b> 3D	<b>X</b> 3E	(3F		(01	<b>X</b> 02	<b>X</b> 03	<b>X</b> 04	<b>X</b> 05	<b>X</b> 06	<b>X</b> 07	<b>X</b> 08
te_enable[3:0]	K <sub>23</sub>	Voh		,sc	<b>Т</b> эл		/or	<b>X</b> 0								
a_in[31:0]	X00003C06	<b>X</b> 00003C05	X00003C04	X00003C03	<b>X</b> 00003C02	X00003C01	<b>X</b> 00003C00									
a_out[31:0]	00003C3F	Λ							X00003C3E	<b>X</b> 00003C3D	X00003C3C	X00003C3B	<b>X</b> 00003C3A	X00003C39	X00003C38	<b>X</b> 000
ck																

Timo	1	1 ts	2 tis	3 ds 4	l is	5 ns (05	f us	7 tsl 8	IS	9 ns   10	XOA	1 ns 12 n (0 B		3 is 14	ns   15 ns	16 us
Time read_addr[5:0]	00	1 ts 01	<b>Y</b> 02	3 tis 4	<b>Y</b> 04	<b>Y</b> 05	<b>Y</b> 06	1 Is 8 <b>X</b> 07	<b>Y</b> 08	<b>Y</b> 09	YOA	1 ns 12 n <b>X</b> OB	Ync	YOD	YOR YO	li is
data_out[31:0]	00000000	0000FFFF	00000000		N° 1					No.2			,oc		Von Vo	
.aca_oac[31.0]	0000000	100001111	100000000	-	-	-	-	-	-							

m. m.o	1						10 ts					20 lts					1) is		
Time read_addr1[3:0]				10	10	10		100	- Va	10	1		16	1/-	1	15			
read_addr1[3:0]	0			X2	X3	X4	<b>X</b> 5	X6	X/	X8		XA.	ДВ	_XC	_Xp	XE	F	In the state of th	
read_addr2[3:0]	F	Œ		XP	XC	<u>χ</u> β	XA.	X9	X8		X6	<b>X</b> 5	χ4	<b>X</b> 3	χ2	X1	10		
write_addr[3:0]	0	1		χ2	<b>X</b> 3	<b>X</b> 4	<b>X</b> 5	<b>X</b> 6	<b>X</b> 7	χ8	<b>X</b> 9	<b>X</b> A	XΒ	Xc Xc	<b>X</b> D	XΕ	F		
data_out1[31:0]	uu+ 0000	000F <b>(</b> uu+	0000000E	uu+ (0000000D	uu+ (00000000	uu+ (0000000B	uu+ (0000000A	uu+ <b>(</b> 00000009	uu+ (00000008	uu+ (00000007	uu+ <b>(</b> 00000006	uu+ (00000005	uu+ 00000004	uu+ 00000003	uu+ (00000002	uu+ 00000001	uu+ <b>(</b> 00000000	0000000F	
data_out2[31:0]	uuuuuuu		,							00000008	<b>X</b> 00000009	X0000000A	X0000000B	X0000000C	X0000000D	X0000000E	X0000000F		X0000000E
clock			ПГ													1			
write_enable							+												
wilte_enable	-																		
1																			
1																			
1																			
1																			
1																			
1																			
1																			
1																			