#### Mehmet Eymen Ünay

#### HW8

For the functional unit ALU and Shifter modules were designed. Ripple carry adder was modified to host the subtraction operation in addition. With this modification a single bit of funct7[6] can be used for carryin and operation selection.

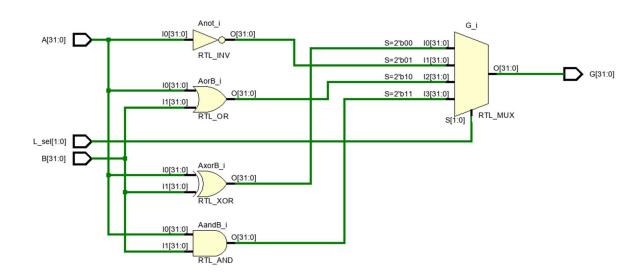
Flags are updated according to the result of ALU. Zero is detected by ORing the arithmetic result and NOTing. Overflow is detected by checking input sign bits, operation and the resulting sign bits.

Shifter is designed fully parametric with a barrel shift topology. The input and outputs are reversed to change the direction of the shift. 31st bit is used for arithmetic sign extension.

#### ALU simulation:

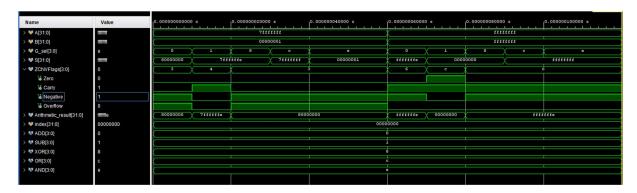


#### Logical unit RTL schematic:

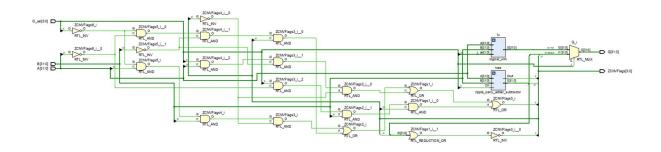


ALU simulation with Flags:

# Mehmet Eymen Ünay



## ALU RTL schematic:



SRA: Arithmetic shift can sign extend the number as can be seen below.

Name	Value		0.000000250000	00000250000 s		0.000000260000 s		g	0.000000280000 s		0.00000290000 s		0.000000300000 s	
> W shift(4:0)	05		00				1						03	
> W S[1:0]	3							3						
¼ IRL	1													
> W B[31:0]	mm6							11111116						
> W H_right[31:0]	111111111111111111111	111111111111111111111111111111111111111			11	1111111111111111	11111111111111	11	111111111111111111111111111111111111111				111111111111111111111111111111111111111	
> W H_shifter[31:0]	111111111111111111111111111111111111111	111111111111111111111111111111111111111			11111111111111111111111111111111111				111111111111111111111111111111111111111				11111111111111	111111
> W index[31:0]	00000006	00000000				0000	0001			00000003				

## SLL:

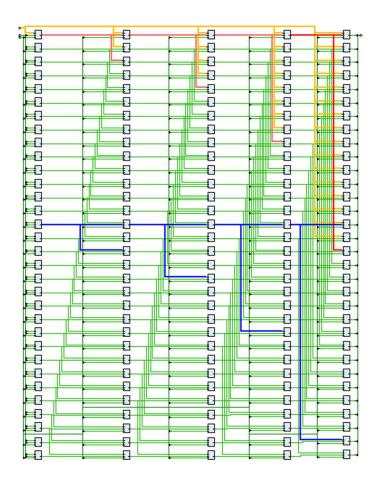
Name	Value	0.0	0.00000010000 =		0.000000020000 =		0.00000030000 =		0.000000040000 =		0.000000050000 =		0.000000060000 #		0.0000
> W shift[4:0]	00		00				1		<del> </del>		02		<del>}</del>	03	
> W S[1:0]	0								0						
¼ IRL	0														
> W B[31:0]	0000000a							000	0000a						
> W H_right[31:0]	0000000000000000000	00000000	00000000000	0000000001010	00	000000000000000	0000000000000001	01	00	00000000000000	000000000000000	10	0000000000000	0000000000000000	000001
> W H_left[31:0]	000000000000000000000000000000000000000	00000000	00000000000	0000000001010	00	000000000000000	000000000000101	00	00	00000000000000	000000000001010	100	0000000000000	000000000000000000000000000000000000000	10000
> W H_shifter[31:0]	0000000000000000000	00000000	00000000000	0000000001010	00	000000000000000	000000000000101	00	00	00000000000000	000000000001010	100	0000000000000	000000000000000000000000000000000000000	10000

## SLR:

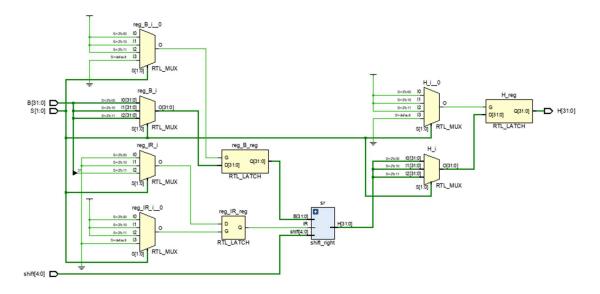
Name	Value	0.000000130	000 s	0.000000140000	*	0.000000150000	=	0.000000160000	0 s	0.000000170000	) =	0.000000180000	) =	0.0000
> W shift[4:0]	02		0	X		1				2		X .	03	
> W S[1:0]	0							2						
¼ IRL	0													
> W B[31:0]	0000000a						000	0000a						
> W H_right[31:0]	000000000000000000000000000000000000000	00000000000000000	00000000000001010	00	000000000000000	0000000000000001	01	0	0000000000000000	0000000000000000	10	0000000000000	000000000000000000000000000000000000000	000001
> W H_left[31:0]	000000000000000000000000000000000000000	00000000000000000	00000000000001010	000000000000000000000000000000000000000			000000000000000000000000000000000000000				000000000000000000000000000000000000000			
> W H_shifter[31:0]	000000000000000000000000000000000000000	0000000000000000	0000000000001010	000000000000000000000000000000000000000			•	000000000000000	000000000000000000000000000000000000000					

## Barrel Left Shifter RTL schematic:

Orange highlighted line is connected to the IR/IL bit which gets shifted into the number. Red and blue highlighted lines indicate the pattern of 2^n.

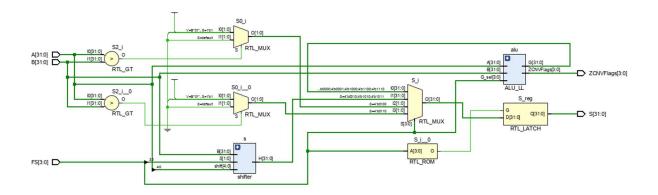


#### Shifter RTL schematic:



Functional Unit RTL schematic:

# Mehmet Eymen Ünay



## Functional Unit Simulation:

