2021 春季学期 计算机组成原理

32 位 ALU

设

计

报

告

Hollow Man

ALU 模块设计报告

一、项目简述

设计语言: Verilog

硬件描述语言

仿真环境: Vivado

最大延迟: 6.426ns

二、实现功能

算术逻辑运算单元(ALU)是数字计算机中执行加、减等算术运算, 执行与、或、非等逻辑运算,以及执行比较、移位等操作的功能部件, 本设计报告为 MIPS 单周期处理器的 ALU 模块的功能、接口、时序及 其实现。该 ALU 在 Vivado 软件环境下进行功能仿真。

实现功能:

and	0000	10010 0	and \$1,\$2,\$3	\$1=\$2 & \$3	rd <- rs & rt ; 其中 rs=\$2, rt=\$3, rd=\$1
or	0000	10010 1	or \$1,\$2,\$3	\$1=\$2 \$3	rd <- rs rt ; 其中 rs=\$2, rt=\$3, rd=\$1
xor	0000	10011 0	xor \$1,\$2,\$3	\$1=\$2 ^ \$3	rd <- rs xor rt ; 其中 rs=\$2, rt=\$3, rd=\$1(异或)
nor	0000	10011 1	nor \$1,\$2,\$3	\$1=~(\$2 \$3)	rd <- not(rs rt) ; 其中 rs=\$2, rt=\$3, rd=\$1(或非)

slt	0000	10101 0	slt \$1,\$2,\$3	if(\$2<\$3) \$1=1 else \$1=0	if (rs < rt) rd=1 else rd=0 ; 其中 rs =\$2, rt=\$3, rd=\$1
sltu	0000	10101 1	sltu \$1,\$2,\$3	if(\$2<\$3) \$1=1 else \$1=0	if (rs < rt) rd=1 else rd=0 ; 其中 rs =\$2, rt=\$3, rd=\$1 (无符号数)
sll	0000	00000	sll \$1,\$2,10	\$1=\$2<<10	rd <- rt << shamt ; shamt 存放移 位的位数, 也就是指令中的立即数,其中 rt=\$2, rd=\$1
srl		00001 0	srl \$1,\$2,10	\$1=\$2>>10	rd <- rt >> shamt ; (logical) ,其中 rt=\$2, rd=\$1
sra	0000	00001 1	sra \$1,\$2,10	\$1=\$2>>10	rd <- rt >> shamt ; (arithmetic) 注 意符号位保留 其中 rt=\$2, rd=\$1
sllv		00010 0	sllv \$1,\$2,\$3	\$1=\$2<<\$3	rd <- rt << rs ; 其中 rs=\$3, rt=\$2, rd=\$1
srlv	0000	00011 0	srlv \$1,\$2,\$3	\$1=\$2>>\$3	rd <- rt >> rs ; (logical)其中 rs= \$3,rt=\$2, rd=\$1
srav	0000	00011 1	srav \$1,\$2,\$3	\$1=\$2>>\$3	rd <- rt >> rs ; (arithmetic) 注意 符号位保留 其中 rs=\$3, rt=\$2, rd=\$1

三、设计方案

注:文档仅对实现原理和部分结构进行说明,具体实现请查看项目代码。

本设计共有1个主模块,5个子模块,1个测试模块

主模块:

算数逻辑运算单元 ALU (加、减等算术运算,执行与、或、非等逻辑运算,以及执行比较、移位)

子模块 (供 ALU 调用):

- 8位加法器(按位实现)
- 32 位加法器 (调用 8 位超前进位加法器实现)
- 32 位减法器(被减数取补码,调用加法器实现)
- 1位移位器(按位实现)
- 2位移位器(按位实现)

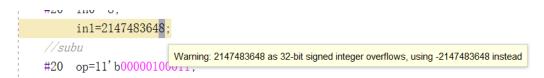
测试模块:

仿真测试,供ALU测试用

基本原理

(一) 关于输入输出的说明

vivado 中,有符号数是采用补码的方式进行输入输出的(如下图,输入为 2³¹ 时已经溢出,为 2³¹⁻¹ 时则会正常),因此不需要单独进行原码和补码之间的转换,为了便于观察,最终算术运算和比较结果将采用有符号十进制(Signed Decimal)和无符号十进制(Signed Decimal)展示,移位和逻辑运算采用二进制(Binary)展示。

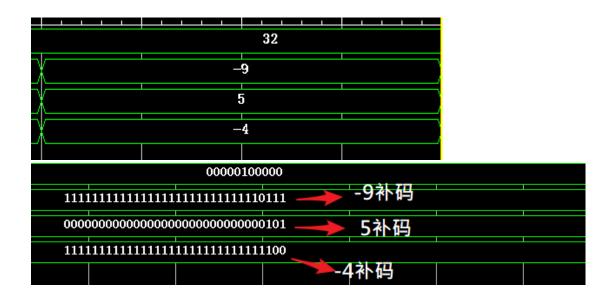


以有符号加法的两组输入为例进行解释:

第一组,采用二进制输入时,viavdo 会将其识别为有符号数的 3和-1,并输出 2的补码,使用有符号十进制查看即可得到 3+(-1)=2



第二组,采用十进制输入时,(-9)+5=(-4),使用二进制查看 发现 vivado 会自动采用其补码形式进行计算,并输出-4的补码



(二) 算术运算

加法器:

加数 A	加数 B	本位 C	进位 D
0	0	0	0
0	1	1	0
1	1	0	1
1	0	1	0

由基本加法进位原理的真值表可知,本位 C 的值是加数 A 和加数 B 做异或得到的,而进位 D 的值是加数 A 与加数 B 做与得到的,故当两个数相加时,本位用异或门实现,而进位使用与门实现。 使用超前进位加法器:

$$C_{i+1} = (A_i \cdot B_i) + (A_i \cdot C_i) + (B_i \cdot C_i)$$
$$= (A_i \cdot B_i) + (A_i + B_i) \cdot C_i$$

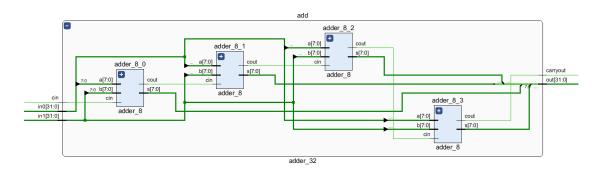
设:

。生成 (Generate) 信号: Gi=Ai·Bi

。传播 (Propagate) 信号: P_i=A_i+B_i

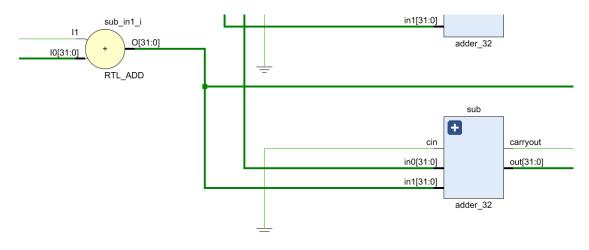
则: $C_{i+1}=G_i+P_i\cdot C_i$

由上式可以看出直接实现 32 位超前进位加法器会非常复杂,所以首先实现 8 位超前进位加法器,并通过 4 个 8 位超前进位加法器组成 32 位超前进位加法器。



减法器:

减法器原理与加法器原理相同,只需要将对应被减数取补码即可。



addu, subu:

addu: 最高有效位向高位有进位,产生进位 carryout

subu: 当被减数小于减数时,最高有效位向高位有借位,产生借位,

将加法进位标志 carryout 取反即可。

add sub:

[x] * + [y] * + [x+y] * +

 $[x-y] \hat{N} = [x] \hat{N} - [y] \hat{N} = [x] \hat{N} + [-y] \hat{N}$

 $[-y \ \hat{\uparrow}] = [y] \hat{\uparrow} + 1$

add 溢出: 两个正数相加产生负数 0111(7)+0110(6)=1101

两个负数相加产生正数 1000(-8)+1001(-7)=0001

一个正数与一个负数相加不产生溢出

sub 溢出: 正数 - 正数 (不产生溢出)

正数 - 负数,结果为负数(上溢)

负数 - 负数 (不产生溢出)

负数 - 正数,结果为正数(下溢)

(三)逻辑运算

~: 按位取反

&(and): 按位与操作

(or): 按位或操作

^(xor): 按位异或操作

~(|) (nor): 按位或非操作

指令对标志位影响:指令执行后,CF和OF置O,ZF根据结果是否为O设置

(四) 比较运算

有符号数比较 slt

指令对标志位影响: in0 小于 in1 置 CF 为 1

in0为正数, in1为负数, out为0

in0 为负数, in1 为正数, out 为 1

in0 和 in1 为负数, alu 进行补码运算, 1111 (-1)、1110 (-2), 直接进行数值比较的结果与其代表的有符号数比较结果相同

in0和 in1为正数,直接进行比较

通过分析,后两种情况可以合并

无符号数比较 sltu

可直接进行比较

指令对标志位影响: in0 小于 in1 置 0F 为 1

(五)移位运算

逻辑左移 sll, sllv

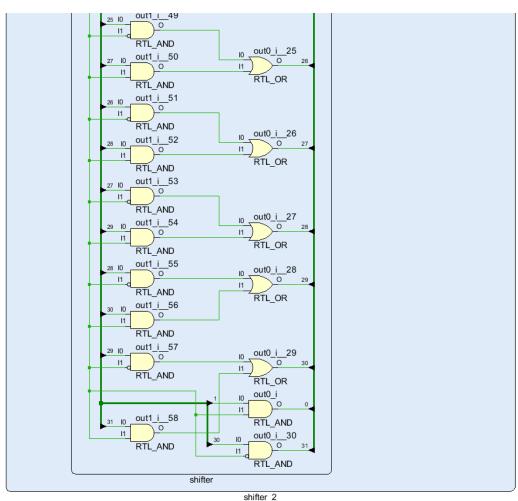
将数据向左移动,最低位用0补充

逻辑右移 srl, srlv

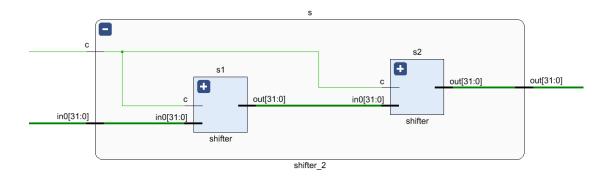
将数据向右移动,最高位用0补充

算术右移 sra, srav

将各位依次右移指定位数,然后在左侧用原符号位补齐 指令对标志的影响:将最后移出的移位写入 CF 移位器 (1位,部分):

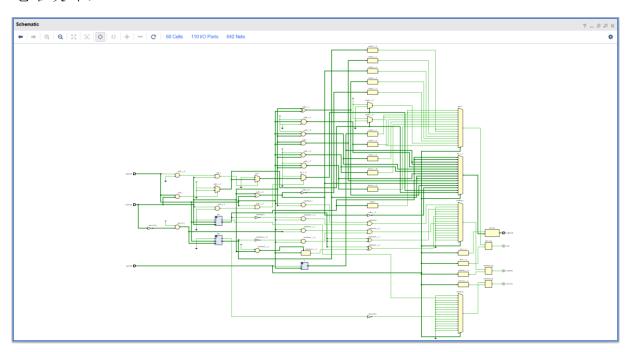


移位器 (2位):



运行结果以及延迟

电路设计:



线路延迟:

Name	Slack ^1	Levels	Routes	High Fanout	From	То	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
3 Path 1	∞	13	14	114	in1[1]	zero_reg/D	6.426	1.919	4.508	00	input port clock
3 Path 2	∞	13	14	114	in1[1]	out_reg[29]/D	5.933	1.922	4.012	00	input port clock
3 Path 3	∞	13	14	114	in1[1]	out_reg[25]/D	5.930	1.919	4.012	∞	input port clock
3 Path 4	∞	13	14	114	in1[1]	out_reg[27]/D	5.930	1.919	4.012	00	input port clock
3 Path 5	∞	13	14	114	in1[1]	out_reg[26]/D	5.922	1.922	4.001	∞	input port clock
→ Path 6	∞	13	14	114	in1[1]	out_reg[31]/D	5.906	2.108	3.799	00	input port clock
→ Path 7	∞	13	14	114	in1[1]	out_reg[28]/D	5.754	2.108	3.647	∞	input port clock
→ Path 8	∞	13	14	114	in1[1]	out_reg[30]/D	5.754	2.108	3.647	00	input port clock
→ Path 9	∞	12	13	114	in1[1]	overflow_reg/D	5.665	1.866	3.800	∞	input port clock
▶ Path 10	∞	12	13	114	in1[1]	out_reg[23]/D	5.587	2.055	3.533	00	input port clock

运行结果:

add

Name	Value	0.000 ns	20.000 ns	40.000 ns	60.000 ns	80.000 ns
> W op[10:0]	34			32		
> W in0[31:0]	-2	3	-9	2147483647	-2147483647	-2147483648
> w in1[31:0]	2147483647	-1	5	1	_	-1
> 1 out[31:0]	2147483647	2	-4	-2147	483648	2147483647
⅓ overflow	1					
₩ zero	0					
¹⊌ carryout	0					

addu



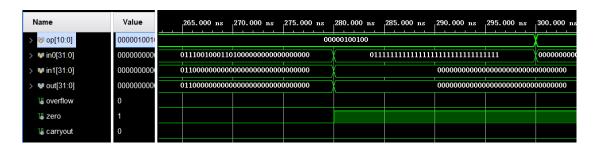
sub

Name	Value	140.000 ns 150.000 ns	160.000 ns 170.000 ns	180.000 ns 190.000 ns	200.000 ns 210.000 ns 2
> W op[10:0]	35	(3	}4	·
> 😻 in0[31:0]	-5	-5	5	-2	2147483647
> 😻 in1[31:0]	8	8	-2	2147483647	-1
> W out[31:0]	-13	-13	7	2147483647	-2147483648
U overflow	0				
1⊌ zero	0				
¼ carryout	0				

subu

Name	Value	220.000 ns 230.000 ns	240.000 ns 250.000 ns 2
> 😻 op[10:0]	36		35
> 💗 in0[31:0]	1916010496	4294967291	8
> 😻 in1[31:0]	1610612736	8	9
> 😼 out[31:0]	1610612736	4294967283	4294967295
₩ overflow	0		
₩ zero	0		
¹⊌ carryout	0		

and



or

Name	Value	30	5.000 ns	310.000 ns	315.000 ns	320.000 ns	325.000 ns	330.000 ns	335.000 ns 3
> 1 op[10:0]	000001001				000001	100101			Х
> 😻 in0[31:0]	101000000	000000	0000000000	0000000000000	0000	0111	1111111111111	11111111111111	1111
> W in1[31:0]	010100000	000000	0000000000	000000000000	0000	1111	.0000000000000	000000000000000000000000000000000000000	0001
> W out[31:0]	111100000	000000	0000000000	000000000000	0000	1111	11111111111111	111111111111111	111
¹⊌ overflow	0								
1⊌ zero	0								
¹⊌ carryout	0								

xor

Name	Value		345.000 ns	350.000 ns	355.000 ns	360.000 ns	365.000 ns	370.000 ns	375.000 ns
> 🐶 op[10:0]	000001001				00000	100110			
> 😻 in0[31:0]	000100100	1010	000000000000	0000000000000	0000	011	11111111111111	1111111111111	1111
> 😻 in1[31:0]	011000000	0101	000000000000	0000000000000	0000	111	10000000000000	000000000000000000000000000000000000000	0001
> W out[31:0]	100011011	1111	000000000000	0000000000000	0000	1000	111111111111	1111111111111	1110
⅓ overflow	0								
₩ zero	0								
	0								

nor

Name	Value		385.000 ns	390.000 ns	395.000 ns	400.000 ns	405.000 ns	410.000 ns	415.000 ns
> 💗 op[10:0]	000001010				00000	100111			·
> 😻 in0[31:0]	111111111	000	1001000110100	010100011111	1111	011	111111111111111	1111111111111	1111
> 😻 in1[31:0]	000000000	0110	000000000000000000000000000000000000000	000000000000000000000000000000000000000	0000	111	10000000000000	00000000000000	0001
> W out[31:0]	000000000	100	0110111001011	101011100000	0000	000	000000000000000000000000000000000000000	000000000000000000000000000000000000000	0000
	0								
¼ zero	0								
l carryout	0								

slt

Name	Value		425.000 ns	430.000 ns	435.000 ns	440.000 ns	445.000 ns	450.000 ns	455.000 ns	460.000 ns	465.000 ns	470.000 ns	475.000 ns	
> W op[10:0]	000001010						00000101010							
> w in0[31:0]	111111111	11	11111111111111	11111111111111	11001	0000	000000000000000000000000000000000000000							
> w in1[31:0]	000000000	000000000000000000000000000000000000000			X	111111111111111111111111111111111111111								
> W out[31:0]	000000000	00	000000000000	000000000000000000000000000000000000000	000001	000000000000000000000000000000000000000								
18 overflow	0													
¹₀ zero	1													
18 carryout	0													

sltu

Name	Value		485.000 ns	490.000 ns	495.000 ns	500.000 ns	505.000 ns	510.000 ns	515.000 ns	520.000 ns	525.000 ns	530.000 ns	535.000 ns 54	
> 🕶 op[10:0]	000000000						0000	0101011						
> 🐶 in0[31:0]	000000000	11	1111111111111	11111111111111	11001	0000	0000000000000	000000000000	0010	111:	111111111111111	11111111111111	111	
> W in1[31:0]	111111111	000000000000000000000000000000000000000					111111111111111111111111111111111111111							
> W out[31:0]	000000000	000000000000000000000000000000000000000				0000	000000000000000000000000000000000000000	000000000000	0001	000000000000000000000000000000000000000				
¹⊌ overflow	0													
¼ zero	0													
1 carryout	0													

s11

Name	Value		545.000 ns	550.000 ns	555.000 ns	560.000 ns	565.000 ns	570.000 ns	575.000 ns	580.000 ns	585.000 ns	590.000 ns	595.000 ns 6
> N op[10:0]	000000000		000000000										
> 💆 in0[31:0]	000000000	00	0000000000000	00000000000000	000101	1111	11111111111111	1111111111111	1000	111:	11111111111111	1111111111111	1111
> 💆 in1[31:0]	111111111												
> W out[31:0]	000000000	00	0000000000000	000000000000000000000000000000000000000	10100	111:	1111111111111	1111111111110	0000	11111111111111111111111111111100			
overflow	0												
18 zero	0												
1 carryout	0												

srl

Name	Value		605.000 ns	610.000 ns	615.000 ns	620.000 ns	625.000 ns	630.000 ns	635.000 ns	640.000 ns	645.000 ns	650.000 ns	655.000 ns		
> W op[10:0]	000000000						0000	0000010							
> 😻 in0[31:0]	000000000	00	000000000000000000000000000000000000000	000000000000000000000000000000000000000	00101	111	11111111111111	1111111111111	1000	1111	11111111111111	11111111111111	111		
> w in1[31:0]	111111111														
> W out[31:0]	000000000	00	000000000000000000000000000000000000000	000000000000000000000000000000000000000	00001	001	11111111111111	11111111111111	1110	0011111111111111111111111111111					
¼ overflow	0														
¼ zero	0														
¹⊌ carryout	0														

sra

Name	Value	66	5.000 ns	670.000 ns	675.000 ns	680.000 ns	685.000 ns	690.000 ns	695.000 ns	700.000 ns	705.000 ns	710.000 ns	715.000 ns 7	
> w op[10:0]	000000001						0000	0000011					X.	
> W in0[31:0]	000000000	0000	00000000000	000000000000000000000000000000000000000	000101	1111	11111111111111	11111111111111	1000	111	111111111111111	11111111111111	.111	
> w in1[31:0]	000000000		miniminimi											
> W out[31:0]	000000000	0000	00000000000	000000000000000000000000000000000000000	000001	1111	11111111111111	11111111111111	1110	111111111111111111111111111111111111111				
¹⊌ overflow	0													
1⊌ zero	0													
1 carryout	0													

sllv

Name	Value		725.000 ns	730.000 ns	735.000 ns	740.000 ns	745.000 ns	750.000 ns	755.000 ns	760.000 ns	765.000 ns	770.000 ns	775.000 ns	
> W op[10:0]	000000001						0000	0000100		X				
> w in0[31:0]	000000000	00	0000000000000	000000000000000000000000000000000000000	00101	111	11111111111111	11111111111111	1000	111111111111111111111111111111111111111				
> W in1[31:0]	000000000	00	0000000000000	000000000000000000000000000000000000000	00011	000	000000000000000000000000000000000000000	0000000000000	0101	000000000000000000000000000000000000000				
> W out[31:0]	000000000	00	0000000000000	000000000000000	01000	111	11111111111111	1111111110000	0000	111	1111111111111111111111111110000			
¹⊌ overflow	0													
¼ zero	1													
18 carryout	0													

srlv

Name	Value		785.000 ns	790.000 ns	795.000 ns	800.000 ns	805.000 ns	810.000 ns	815.000 ns	820.000 ns	825.000 ns	830.000 ns	835.000 ns 8		
> 🐶 op[10:0]	00000001						0000	0000110			· · · · · · · · · · · · · · · · · · ·				
> w in0[31:0]	000000000	00	000000000000	0000000000000	000101	111	1111111111111	1111111111111	1000	111111111111111111111111111111111111111					
> 😻 in1[31:0]	000000000	00	0000000000000	000000000000000000000000000000000000000	000011	000	00000000000000	0000000000000	0101	000000000000000000000000000000000000000					
> W out[31:0]	000000000	00	000000000000	000000000000000000000000000000000000000	000000	000	0011111111111	1111111111111	1111	000	000011111111111111111111111111				
overflow	0														
¹⊌ zero	1														
18 carryout	0														

srav

Name	Value		840.000 ns	845.000 ns	850.000 ns	855.000 ns	860.000 ns	865.000 ns	870.000 ns	875.000 ns	880.000 ns	885.000 ns	890.000 ns	895.000 ns	
> W op[10:0]	00000001	<u> </u>	1					00000	000111						
> w in0[31:0]	111111111		0000000000000000000000000000000000						1111111111011	0000	111111111111111111111111111111111111111				
> w in1[31:0]	000000000	<u> </u>	0000000000000000000000000000011					000000000000000000000000000000000000000	0000000000000	0101	000000000000000000000000000000000000000				
> W out[31:0]	111111111)	000	000000000000000000000000000000000000000	000000000000000000000000000000000000000	0000	111	111111111111	11111111111111	1101	111111111111111111111111111111111111111				
16 overflow	0														
¹⊌ zero	0														
18 carryout	0														