## TEST CASES ALU

ALU								
ADDER								
NAME	TEST_A	TEST_B	TEST_ALUFN	TEST_C	TEST_Z	TEST_V	TEST_N	Description
ADD1	0000000000000001	0000000000000001	000000	0000000000000010	0	0	0	Basic addition of 2 positive numbers: 1 + 1 = 2
ADD2	1111111111111111	1111111111111111	000000	1111111111111111	0	0	1	Addition of 2 negative numbers: (-1) + (-1) = (-2)
ADD3	000000100000010	1111111011010101	000000	11111111110101111	0	0	1	Addition of positive and negative number: 258 + (-299) = -41
ADD4	1000000000000000	10000000000000000	000000	000000000000000000000000000000000000000	1	1	0	Addition of 2 negative numbers, exceeds -32768: (-32768) + (-32768) = (-65536)
ADD5	0100000000000000	01000000000000000	000000	10000000000000000	0	1	1	Addition of 2 positive numbers, exceeds 32767: 16384 + 16384 = 32768
SUB1	0000000000000001	00000000000000001	000001	000000000000000000000000000000000000000	1	0	0	Basic subtraction of 2 positive numbers: $1 - 1 = 0$
SUB2	1111111111111111	1111111111111111	000001	000000000000000000000000000000000000000	1	0	0	Subtraction of 2 negative numbers: (-1) - (-1) = 0
SUB3	0000110110101100	1111011000101000	000001	0001011110000100	0	0	0	Positive number subtract negative number: 3500 – (-2520) = 6020
SUB4	0100111000100000		000001	1000100010111000	0	1	1	Positive number subtract negative number, exceeds 32767: 20000 – (-15000) = 35000
SUB5	1000101011010000		000001	0111011101001000	0	1	0	Negative number subtract positive number, exceeds -32768: (-30000) – 5000 = (-35000)
NNUL	0000010000011010		000010	0000101100100000	0	0	0	Multiplication of 2 16-bit positive numbers, lowest 16-bits: 1050 * 2000 = 2100000
COMPAR			000020	000000000000000000000000000000000000000				
NAME	TEST A	TEST B	TEST ALUFN	TEST C	TEST Z	TEST V	TEST N	Description
EQ1	0000000000001011	0000000000011111	110011	000000000000000000000000000000000000000	0	0	1	Compare equals 2 different numbers: 11 == 31 = FALSE
		0000000000011111	110011		0	0	0	Compare equals 2 different numbers, opposite: 31 == 11 = FALSE
EQ2	0000000000011111			000000000000000000000000000000000000000			0	
EQ3	000000000000000000000000000000000000000		110011	000000000000000000000000000000000000000	1	0	0	Compare equals 2 zeros: 0 == 0 = TRUE
EQ4		1111111111111111	110011	000000000000000000001	1	0	0	Compare equals the same negative number: (-1) == (-1) = TRUE
EQ5	000000000001011	000000000001011	110011	00000000000000000001	1	0	0	Compare equals the same positive number: 11 == 11 = TRUE
EQ6	0000000000000001	00000000000000001	110011	00000000000000001	1	0	0	Compare equals 2 ones: 1 == 1 = TRUE
LT1	000000000001011	000000000011111	110101	000000000000000000001	0	0	1	Compare less than 2 different numbers: 11 < 31 = TRUE
LT2	000000000011111	0000000000001011	110101	000000000000000000000000000000000000000	0	0	0	Compare less than 2 different numbers, opposite: 31 < 11 = FALSE
LT3	00000000000000000	000000000000000000000000000000000000000	110101	000000000000000000000000000000000000000	1	0	0	Compare less than 2 zeros: 0 < 0 = FALSE
LT4	11111111111111111	1111111111111111	110101	000000000000000000000000000000000000000	1	0	0	Compare less than the same negative number: (-1) < (-1) = FALSE
LT5	000000000001011	000000000001011	110101	000000000000000000000000000000000000000	1	0	0	Compare less than the same positive number: 11 < 11 = FALSE
LE1	000000000001011	000000000011111	110111	00000000000000001	0	0	1	Compare less than or equal to 2 different numbers: 11 <= 31 = TRUE
LE2	000000000011111	000000000001011	110111	000000000000000000000000000000000000000	0	0	0	Compare less than or equal to 2 different numbers, opposite: 31 <= 11 = FALSE
LE3	00000000000000000	000000000000000000000000000000000000000	110111	00000000000000001	1	0	0	Compare less than or equal to 2 zeros: 0 <= 0 = TRUE
LE4	11111111111111111	11111111111111111	110111	00000000000000001	1	0	0	Compare less than or equal to the same negative number: (-1) <= (-1) = TRUE
1.55	000000000000000000000000000000000000000							
LE5	0000000000001011	0000000000001011	110111	00000000000000001	1	0	0	Compare less than or equal to the same positive number: 11 <= 11 = TRUE
BOOLEA		0000000000001011	110111	00000000000000000001	1	0	0	
BOOLEA	N				1 TEST Z			Compare less than or equal to the same positive number: 11 <= 11 = TRUE
NAME	N TEST_A	TEST_B	TEST_ALUFN	TEST_C		TEST_V		Compare less than or equal to the same positive number: 11 <= 11 = TRUE  Description
NAME AND1	N TEST_A 0100011110001001	TEST_B 0001101000001110	TEST_ALUFN 011000	TEST_C 0000001000001000	0	TEST_V 0	TEST_N 0	Compare less than or equal to the same positive number: 11 <= 11 = TRUE  Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520
NAME AND1 AND2	TEST_A 0100011110001001 11111111111111111	TEST_B 0001101000001110 0000000000001010	TEST_ALUFN 011000 011000	TEST_C 000000100001000 0000000000001010	0	TEST_V	TEST_N	Compare less than or equal to the same positive number: 11 <= 11 = TRUE  Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10
NAME AND1 AND2 OR1	TEST_A 0100011110001001 1111111111111 100010001000000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0	TEST_V 0 0	TEST_N 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369
NAME AND1 AND2 OR1 OR2	TEST_A 0100011110001001 1111111111111 100010001000000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110 011110	TEST_C 000000100001000 0000000000001010 100110011100100	0 0 0 0	TEST_V 0 0 1	TEST_N 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535
NAME AND1 AND2 OR1 OR2 XOR1	TEST_A 0100011110001001 111111111111 10001000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0	TEST_V 0 0 1 0 0	TEST_N 0 0 0 1 1	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702
NAME AND1 AND2 OR1 OR2 XOR1 XOR2	TEST_A 0100011110001001 1111111111111 100010001000000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 010110	TEST_C 000000100001000 000000000001010 100110011100100	0 0 0 0 0	TEST_V 0 0 1 0 1 1 1 1	TEST_N 0 0	Compare less than or equal to the same positive number: 11 <= 11 = TRUE  Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 34944   39241 = 39369  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1	TEST_A 0100011110001001 111111111111 10001000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 010110 011010	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 1 0 0	TEST_N 0 0 0 1 1 1 0	Compare less than or equal to the same positive number: 11 <= 11 = TRUE  Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 34944   39241 = 39369  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2	TEST_A 0100011110001001 111111111111 10001000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 010110 011010 011010	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0	TEST_N 0 0 0 1 1 0 1 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B	TEST_A 0100011110001001 1111111111111 100010001000000	TEST_B 0001101000001110 0000000000001010 1001100101001 001000000	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011100	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0 1 0 0	TEST_N 0 0 0 1 1 0 1 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR	TEST_A 0100011110001001 111111111111 10001000	TEST_B 0001101000001110 0000000000001010 10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 010110 011010 011010 011010 01100	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 0 1 0 0 0 0	Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 34944   39241 = 39369  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321  Asel of 2 different numbers: Asel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 29591  Bitwise NOR of 2 different numbers: ~(21796   10) = 35537
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND	TEST_A 0100011110001001 111111111111 10001000	TEST_B 0001101000001110 0000000000001010 1001100101001 001000000	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011100	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0 1 0 0	TEST_N 0 0 0 1 1 0 1 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER	TEST_A 0100011110001001 1111111111111 100010001000000	TEST_B  0001101000001110  0000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011100 0111100 010111	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0	Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 34944   39241 = 39369  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321  Asel of 2 different numbers: Asel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 29591  Bitwise NOR of 2 different numbers: ~(21796   10) = 35537  Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 TEST_ALUFN	TEST_C 0000001000001000 0000000000001010 100110011100100	0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0 1 TEST_V	TEST_N  0  0  0  1  1  0  0  0  TEST_N	Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 34944   39241 = 39369  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321  Asel of 2 different numbers: Asel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 29591  Bitwise NOR of 2 different numbers: ~(21796   10) = 35537  Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1	TEST_A  0100011110001001  1111111111111  100010001000000	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 TEST_ALUFN 100000	TEST_C 000000100001000 0000000000001010 100110011100100	0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 0 0 TEST_V	TEST_N  0  0  0  1  1  0  0  0  TEST_N	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2
BOOLEAI NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 TEST_ALUFN 100000 100000	TEST_C 000000100001000 0000000000001010 100110011100101 111111	0 0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0	TEST_V 0 0 1 0 1 0 0 1 0 0 0 TEST_V	TEST_N  0  0  1  1  0  0  0  TEST_N  0  0  0  0  0  0  0  0  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 15: 1 << 15 = 32768
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3	TEST_A  0100011110001001  1111111111111  100010001000000	TEST_B  0001101000001110  0000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 0111100 TEST_ALUFN 100000 100000 100000	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0 0	TEST_V 0 0 1 0 0 1 0 0 0 0 0 TEST_V	TEST_N  0  0  0  1  1  0  0  0  TEST_N	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512
BOOLEAI NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 011010 01001 010011 TEST_ALUFN 100000 100000 100000	TEST_C 000000100001000 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0 0	TEST_V  0  0  1  0  1  0  0  0  1  0  0  TEST_V  0  0  0  0  0  TEST_V  0  0  0	TEST_N  0  0  1  1  0  0  0  TEST_N  0  0  0  0  0  0  0  0  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512 Shift left by 8, exceeds 65535: 39321 << 8 = 39168
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5	TEST_A  01000100010000000001  11111111111111	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 0111100 TEST_ALUFN 100000 100000 100000	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0 0 0	TEST_V 0 0 1 0 0 1 0 0 0 0 0 TEST_V	TEST_N  0  0  1  1  0  0  0  TEST_N  0  0  0  0  0  0  0  0  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512 Shift left by 8, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 34944 << 9 = 0
BOOLEAI NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 011010 01001 010011 TEST_ALUFN 100000 100000 100000	TEST_C 000000100001000 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0 0	TEST_V  0  0  1  0  1  0  0  0  1  0  0  TEST_V  0  0  0  0  0  TEST_V  0  0  0	TEST_N  0  0  1  1  0  0  0  TEST_N  0  0  0  0  0  0  0  0  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512 Shift left by 8, exceeds 65535: 39321 << 8 = 39168
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5	TEST_A  01000100010000000001  11111111111111	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 011010 01001 010011 TEST_ALUFN 100000 100000 100000 100000	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0 0 0	TEST_V 0 0 1 0 0 1 0 0 0 0 0 0 TEST_V 0 0 0 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512 Shift left by 8, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 34944 << 9 = 0
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001110  00000000000001010  10011001010010	TEST_ALUFN 011000 011000 011110 011110 010110 010110 011010 011010 011010 011100 01001 010111  TEST_ALUFN 100000 100000 100000 100000 100000 100000	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 TEST_Z 0 0 0	TEST_V 0 0 1 0 1 0 0 1 0 0 0 TEST_V 0 0 0 1 1 0 1 1 0 1 0 1 1 1 0 1 1 1 1	TEST_N  0  0  1  1  0  0  1  0  0  TEST_N  0  0  1  1  0  1  1  0  1  1  0  0  1  1	Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 34944   39241 = 39369  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321  Asel of 2 different numbers: Asel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 29591  Bitwise NOR of 2 different numbers: ~(21796   10) = 35537  Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description  Shift left by 1: 1 << 1 = 2  Shift left by 10, exceeds 65535: 65535 << 10 = 64512  Shift left by 9, exceeds 65535: 34944 << 9 = 0  Shift right by 1: 32768 >> 1 = 16384
BOOLEAN NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2	TEST_A  010001001000000000000000000000000000	TEST_B  0001101000001110  1001100101001001  100100	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 01100 011010 011010 01001 01001 01001 100000 100000 100000 100000 100000 100000 100000 100000 100000	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 1 0 0 0 0 TEST_N 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512 Shift left by 8, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 34944 << 9 = 0 Shift right by 1: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 15 = 1
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001110  1001100101001001  100100	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 011100 01001 010111  TEST_ALUFN 100000 100000 100000 100000 100000 100000 100000 100000 100000 100000 100000 1000001	TEST_C 000000100001010 100110011101111 0001101011001110 100110011001110 0010101011001101 100110011001101 100110011010111 100010111111	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 1 0 0 0 0 TEST_N 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise OR of 2 different numbers: 39494   39241 = 39369  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321  Asel of 2 different numbers: Asel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 29591  Bitwise NOR of 2 different numbers: ~(21796   10) = 35537  Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description  Shift left by 1: 1 << 1 = 2  Shift left by 10, exceeds 65535: 65535 << 10 = 64512  Shift left by 9, exceeds 65535: 39321 << 8 = 39168  Shift left by 9, exceeds 65535: 34944 << 9 = 0  Shift right by 1: 32768 >> 1 = 16384  Shift right by 15: 32768 >> 15 = 1  Shift right by 10: 65535 >> 10 = 63  Shift right by 8: 39321 >> 8 = 153
BOOLEAN NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3 SHR4	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001010  1001100101001001  100100	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 01100 011010 011100 01001 010111  TEST_ALUFN 100000 100000 100000 100000 100000 100000 100001 100001 100001	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	TEST_N  0  0  1  1  0  0  1  0  0  1  0  0  1  0  0	Description  Bitwise AND of 2 different numbers: 18313 & 6670 = 520  Bitwise AND of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 65535 & 10 = 10  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise OR of 2 different numbers: 65535 & 10 = 65535  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702  Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947  Asel of 2 different numbers: Asel(39321, 8) = 39321  Asel of 2 different numbers: Asel(257, 29591) = 257  Bsel of 2 different numbers: Bsel(257, 29591) = 257  Bsel of 2 different numbers: "(21796   10) = 35537  Bitwise NOR of 2 different numbers: "(1 & 3) = 65534  Description  Shift left by 1: 1 << 1 = 2  Shift left by 15: 1 << 15 = 32768  Shift left by 10, exceeds 65535: 39321 << 8 = 39168  Shift left by 9, exceeds 65535: 34944 << 9 = 0  Shift right by 1: 32768 >> 1 = 16384  Shift right by 15: 32768 >> 15 = 1  Shift right by 8: 39321 >> 8 = 153  Shift right by 9: 273 >> 9 = 0
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3 SHR4 SHR5 SRA1	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001010  1001100101001001  100100	TEST_ALUFN 011000 011110 011110 010110 010110 011010 011010 011010 011010 011010 01001 010011 TEST_ALUFN 100000 100000 100000 100000 100000 100001 100001 100001 100001 100001 100001 100001	TEST_C 000000100001010 100110011101111 0001101011001110 100110011001110 0010101011001110 100110011001101 100110011011	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 1 0 0 0 0 1 1 0 0 0 1 1 0 0 1 1 0 0 1 1 1 0 0 1 1 0 0 0 1 1 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 257 Bsel of 2 different numbers: 8sel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(21796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 34944 << 9 = 0 Shift right by 1: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 10 = 63 Shift right by 8: 39321 >> 8 = 153 Shift right by 9: 273 >> 9 = 0 Shift right arithmetic by 1 with signed bit 1: -32768 >>> 1 = -16384
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3 SHR4 SHR5 SRA1 SRA2	TEST_A  0100011110001001  11111111111111  10001001	TEST_B  0001101000001010  1001100101001001  100100	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 011110  TEST_ALUFN 100000 100000 100000 100000 100001 100001 100001 100001 100001 100001 100001 100001 100001	TEST_C 000000100001010 100110011100101 1111111	TEST_Z  0  0  0  0  0  0  0  0  0  0  0  0  0	TEST_V 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	TEST_N  0  0  0  1  1  0  0  0  0  1  0  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 257 Bsel of 2 different numbers: 8sel(257, 29591) = 29591 Bitwise NOR of 2 different numbers: ~(12796   10) = 35537 Bitwise NAND of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 10, exceeds 65535: 65535 << 10 = 64512 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 34944 << 9 = 0 Shift right by 1: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 10 = 63 Shift right by 18: 39321 >> 8 = 153 Shift right by 19: 3273 >> 9 = 0 Shift right arithmetic by 1 with signed bit 1: -32768 >>> 1 = -16384 Shift right arithmetic by 1 with signed bit 0: 16384 >>> 14 = 1
BOOLEAN NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3 SHR4 SHR5 SRA1 SRA2 SRA3	TEST_A  0100011110001001  11111111111111  100010001000000	TEST_B  0001101000001010  1001100101001001  100100	TEST_ALUFN 011000 011110 011110 010110 010110 011010 011010 011010 011010 011010 011010 01001 010011 TEST_ALUFN 100000 100000 100000 100000 100000 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001	TEST_C 000000100001010 100110011101111 0001101011001110 0010101011001110 001010110011001 100110011001101 100110011011	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	TEST_N  0  0  1  1  0  0  0  1  0  0  0  1  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 257 Bsel of 2 different numbers: ~(21796   10) = 35537 Bitwise NOR of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 15: 1 << 1 = 2 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift right by 1: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 1 = 64512 Shift right by 9: 273 >> 9 = 0 Shift right by 9: 273 >> 9 = 0 Shift right arithmetic by 1 with signed bit 1: -32768 >>> 1 = -16384 Shift right arithmetic by 1 with signed bit 0: 16384 >>> 14 = 1 Shift right arithmetic by 10 with signed bit 1: -1 >>> 10 = -1
NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3 SHR4 SHR5 SRA1 SRA2 SRA3 SRA4	TEST_A 0100011111111111111111111111111111111	TEST_B  0001101000001010  1001100101001001  100100	TEST_ALUFN 011000 011000 011110 011110 010110 011010 011010 011010 011010 011010 011110  TEST_ALUFN 100000 100000 100000 100000 100000 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001	TEST_C 000000100001010 100110011100101 1111111	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_N 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 34944   39241 = 39369 Bitwise OR of 2 different numbers: 93462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65535 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 257 Bsel of 2 different numbers: ~(21796   10) = 35537 Bitwise NOR of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 15: 1 << 1 = 2 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift right by 1: 32768 >> 1 = 16384 Shift right by 1: 32768 >> 1 = 16384 Shift right by 10: 65535 >> 10 = 63 Shift right by 9: 273 >> 9 = 0 Shift right arithmetic by 1 with signed bit 1: -32768 >>> 1 = -16384 Shift right arithmetic by 14 with signed bit 0: 16384 >>> 14 = 1 Shift right arithmetic by 10 with signed bit 1: -1>>> 10 = -1 Shift right arithmetic by 8 with signed bit 1: -26215 >>> 8 = -103
BOOLEAN NAME AND1 AND2 OR1 OR2 XOR1 XOR2 A1 A2 B NOR NAND SHIFTER NAME SHL1 SHL2 SHL3 SHL4 SHL5 SHR1 SHR2 SHR3 SHR4 SHR5 SRA1 SRA2 SRA3	TEST_A 0100011110001001 11111111111111111111	TEST_B  0001101000001010  1001100101001001  100100	TEST_ALUFN 011000 011110 011110 010110 010110 011010 011010 011010 011010 011010 011010 01001 010011 TEST_ALUFN 100000 100000 100000 100000 100000 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001 100001	TEST_C 000000100001010 100110011101111 0001101011001110 0010101011001110 001010110011001 100110011001101 100110011011	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	TEST_V 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	TEST_N  0  0  1  1  0  0  0  1  0  0  0  1  0  0	Description Bitwise AND of 2 different numbers: 18313 & 6670 = 520 Bitwise AND of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 10 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise OR of 2 different numbers: 65535 & 10 = 65535 Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 Bitwise XOR of 2 different numbers: 54564 ^ 65511 = 10947 Asel of 2 different numbers: Asel(39321, 8) = 39321 Asel of 2 different numbers: Asel(257, 29591) = 257 Bsel of 2 different numbers: Bsel(257, 29591) = 257 Bsel of 2 different numbers: ~(21796   10) = 35537 Bitwise NOR of 2 different numbers: ~(1 & 3) = 65534  Description Shift left by 1: 1 << 1 = 2 Shift left by 15: 1 << 1 = 2 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift left by 9, exceeds 65535: 39321 << 8 = 39168 Shift right by 1: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 1 = 16384 Shift right by 15: 32768 >> 1 = 64512 Shift right by 9: 273 >> 9 = 0 Shift right by 9: 273 >> 9 = 0 Shift right arithmetic by 1 with signed bit 1: -32768 >>> 1 = -16384 Shift right arithmetic by 1 with signed bit 0: 16384 >>> 14 = 1 Shift right arithmetic by 10 with signed bit 1: -1 >>> 10 = -1

## FAIL CASES

ALU								
ADDER								
NAME	TEST_A	TEST_B	TEST_ALUFN	TEST_C	TEST_Z	TEST_V	TEST_N	Description
ADD1	0000000000000001	0000000000000001	0 <mark>1</mark> 0000	00000000000000000010	0	0	0	Basic addition of 2 positive numbers: 1 + 1 = 2 (Invalid ALUFN)
ADD2	1111111111111111	11111111111111111	000000	1111111111111111	0	0	<mark>0</mark>	Addition of 2 negative numbers: (-1) + (-1) = (-2) (Invalid N)
SUB1	0000000000000001	00000000000000001	000001	000000000000000000000000000000000000000	1	<mark>1</mark>	0	Basic subtraction of 2 positive numbers: $1 - 1 = 0$ (Invalid V)
SUB2	1111111111111111	11111111111111111	000001	000000000000000000000000000000000000000	0	0	0	Subtraction of 2 negative numbers: (-1) - (-1) = 0 (Invalid Z)
NNUL	0000010000011010	0000011111010000	000010	000010110010000 <mark>1</mark>	0	0	0	Multiplication of 2 16-bit positive numbers: 1050 * 2000 = 2100000 (Invalid C)
COMPAR	RE							
NAME	TEST_A	TEST_B	TEST_ALUFN	TEST_C	TEST_Z	TEST_V	TEST_N	Description
EQ1	000000000001011	000000000011111	11001 <mark>0</mark>	000000000000000000000000000000000000000	0	0	1	Compare equals 2 different numbers: 11 == 31 = FALSE (Invalid ALUFN)
EQ2	000000000011111	000000000001011	110011	000000000000000000000000000000000000000	0	0	<mark>1</mark>	Compare equals 2 different numbers, opposite: 31 == 11 = FALSE (Invalid N)
LT1	000000000001011	000000000011111	110101	00000000000000000001	0	<mark>1</mark>	1	Compare less than 2 different numbers: 11 < 31 = TRUE (Invalid V)
LT2	000000000011111	000000000001011	110101	000000000000000000000000000000000000000	<mark>1</mark>	0	0	Compare less than 2 different numbers, opposite: 31 < 11 = FALSE (Invalid Z)
LE1	000000000001011	000000000011111	110111	00000000000000000000000000000000000000	0	0	1	Compare less than or equal to 2 different numbers: 11 <= 31 = TRUE (Invalid C)
BOOLEA	N							
NAME	TEST_A	TEST_B	TEST_ALUFN	TEST_C	TEST_Z	TEST_V	TEST_N	Description
AND1	0100011110001001	0001101000001110	01100 <mark>1</mark>	0000001000001000	0	0	0	Bitwise AND of 2 different numbers: 18313 & 6670 = 520 (Invalid ALUFN)
OR1	1000100010000000	1001100101001001	011110	1001100111001001	0	1	<mark>1</mark>	Bitwise OR of 2 different numbers: 34944   39241 = 39369 (Invalid N)
OR2	1111111111111111	0010000000001010	011110	1111111111111111	0	<mark>1</mark>	1	Bitwise OR of 2 different numbers: 65535 & 10 = 65535 (Invalid V)
XOR2	1101010100100100	1111111111100111	010110	0010101011000011	1	1	0	Bitwise XOR of 2 different numbers: 39462 ^ 32776 = 6702 (Invalid Z)
A2	000000100000001	0111001110010111	011010	000000010000000 <mark>0</mark>	0	0	0	Asel of 2 different numbers: Asel(257, 29591) = 257 (Invalid C)
SHIFTER								
NAME	TEST_A	TEST_B	TEST_ALUFN	TEST_C	TEST_Z	TEST_V	TEST_N	Description
SHL1	0000000000000001	00000000000000001	1000 <mark>1</mark> 0	00000000000000000010	0	0	0	Shift left by 1: 1 << 1 = 2 (Invalid ALUFN)
SHL2	0000000000000001	000000000001111	100000	1000000000000000	0	0	<mark>1</mark>	Shift left by 15: 1 << 15 = 32768 (Invalid N)
SHR1	1000000000000000	00000000000000001	100001	0100000000000000	0	0	0	Shift right by 1: 32768 >> 1 = 16384 (Invalid V)
SHR2	1000000000000000	000000000001111	100001	00000000000000001	1	1	0	Shift right by 15: 32768 >> 15 = 1 (Invalid Z)
SRA1	1000000000000000	0000000000000001	100011	1100000000000000 <mark>1</mark>	1	1	0	Shift right arithmetic by 1 with signed bit 1: -32768 >>> 1 = -16384 (Invalid C)

## **ERROR MODES**

EKI	KUK IVIC	<u>/DES</u>	
A	JF1 (Au	tomatic Fail 1)	
	NAME	Description	
	FAIL	The 16-bit output from the ALU module does not correspond with the expected value of the output from the test case	
A	JF2 (Au	tomatic Fail 2)	
	NAME	Description	
	ERR1	The 16-bit output from the ALU module does not correspond with the expected value of the output from the test case	
	ERR2	The 1-bit output for zero condition Z from the ALU/adder unit does not correspond with the expected value of Z from the test case	
	ERR3	The 1-bit output for overflow condition V from the ALU/adder unit does not correspond with the expected value of V from the test case	
	ERR4	The 1-bit output for the most significant bit from the ALU/adder unit N does not correspond with the expected value of the most significant bit from the test case	
	FRR5	The 6-bit ALLIEN input to the ALLI module is not a valid opcode (no operation for that opcode has been implemented in the ALLI)	