1. Use the dial switches on the development board for input, including 3 dial switches (x2,.. X0) for inputting test case numbers, 8 dial switches (sw7,.. Sw0) for inputting test data, and using LED lights or 7-segment tubes for output (If the number of LED lights is not enough, the output will be displayed alternately on the LED lights, with each part displayed for 2 seconds.) (Note that the high bit of the DIP switch and LED will be unified on the left, and the low bit will be unified on the right.)

NOTE 1: If there are multiple input data (such as a, b), please enter a first, press the confirm button, and then enter b

NOTE 2: Use LED to output the binary form of the operand, and use 7-segment tube (or VGA) to output the hexadecimal form of the operation result.

Scenario 1. Test case number	Test case description (in test case 3'b011-3'b111, both a and b are in binary complement form)	Examples
3‘b000	Input test number a, input test number b, and display the 8-bit binary format of a and b on the output device (LED)	Input 8’b1101_0011 as a, the state of leds should be 8’b1101_0011
3‘b001	Input the test number a, place it in a register by instruction lb, display the value of the 32-bit register in hexadecimal format on the output device (7-segment tubes or VGA), and save the number to memory (in the 3'b011-3'b111 test case, the value of a will be read from the memory unit using the ‘lw’ instruction for comparison)	Input 8’b1101_0011 as a， using instruction ‘lb’ to read it and store it to the register. The value in the register should be 32’hFFFF_FFD3, the value displayed on the 7-seg tubes(or VGA) should be FFFFFFD3

3‘b010	Input the test number b, place it in a register by instruction ‘lbu’, display the value of the 32-bit register in hexadecimal format on the output device (7-segment tubes or VGA), and save the number to memory (in the 3'b011-3'b111 test case, the value of b will be read from the memory unit using the ‘lw’ instruction for comparison)	Input 8’b1101_0011 as b，using instruction ‘lbu’ to read it and store it to the register. The value in the register should be 32’h0000_00D3, , the value displayed on the 7-seg tubes(or VGA) should be 000000D3
3‘b011	Compare test number a and test number b (from test case 3’b001 and test case 3’b010) using instruction ‘beq’. If the relationship is true, light up the LED, but if the relationship is not true, turn off the LED	While both a and b got from tc001 and tc010 are 8’b0101_0011(both a and b are stored in the registers then stored in the data memory), here a equal to b, so the led should be turned on； While both a and b got from tc001 and tc010 are 8’b1101_0011, (both a and b are stored in the registers then stored in the data memory), here a is not equal to b, so the led should be turned off； While a got from tc001 is 8’b0101_0011 and b got from tc010 is 8’b1101_0011 (both a and b are stored in the registers then stored in the data memory) , a is not equal to b , so the led should be turned off；
3‘b100	Compare test number a and test number b (from test case 3’b001 and test case 3’b010) using instruction ‘blt’. If the relationship is true, light up the LED, but if the relationship is not true, turn off the LED	While both a and b got from tc001 and tc010 are 8’b0101_0011, (both a and b are stored in the registers then stored in the data memory), a is not less than b, so the led should be turned off； While both a and b got from tc001 and tc010 are 8’b1101_001(both a and b are stored in the registers then stored in the data memory), the value of a turns to be 32’hFFFF_FFD3， the value of b turns to be

		32’h0000_00D3， a is less than b， so the led should be turned on； While a got from tc001 is 8’b1101_0011 and b got from tc010 is 8’b0101_0011 (both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is less than b, so the led should be turned on
3‘b101	Compare test number a and test number b (from test case 3’b001 and test case 3’b010) using instruction ‘bge’. If the relationship is true, light up the LED, but if the relationship is not true, turn off the LED	While both a and b got from tc001 and tc010 are 8’b0101_0011, (both a and b are stored in the registers then stored in the data memory), a is greater or equal to b, so the led should be turned on； While both a and b got from tc001 and tc010 are 8’b1101_0011, (both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is not greater or equal to b， so the led should be turned off； While a got from tc001 is 8’b1101_0011 and b got from tc010 is 8’b0101_0011 (both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is not greater or equal to b， so the led should be turned off.
3‘b110	Compare test number a and test number b (from test case 3’b001 and test case 3’b010) using instruction ‘bltu’. If the relationship is true, light up the LED, but if the relationship is not true, turn off	While both a and b got from tc001 and tc010 are 8’b0101_0011, (both a and b are stored in the registers then stored in the data memory), a is not less than b, so the led should be turned off； While both a and b got from tc001 and tc010 are 8’b1101_0011,

	the LED	(both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is not less than（unsigned）b， so the led should be turned off； While a got from tc001 is 8’b1101_0011 and b got from tc010 is 8’b0101_0011 (both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is not less than b(unsigned)， so the led should be turned off
3‘b111	Compare test number a and test number b (from test case 3’b001 and test case 3’b010) using instruction ‘bgeu’. If the relationship is true, light up the LED, but if the relationship is not true, turn off the LED	While both a and b got from tc001 and tc010 are 8’b0101_0011, (both a and b are stored in the registers then stored in the data memory), a is greater or equal（unsigned） to b，so the led should be turned on； While both a and b got from tc001 and tc010 are 8’b1101_0011, (both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is greater or equal（unsigned）to b， so the led should be turned on； While a got from tc001 is 8’b1101_0011 and b got from tc010 is 8’b0101_0011 (both a and b are stored in the registers then stored in the data memory), a turns to be 32’hFFFF_FFD3， b turns to be 32’h0000_00D3， a is greater or equal（unsigned）to b，so the led should be turned on

2. Use the dial switches on the development board for input, including 3 dial switches (x2,.. X0) for inputting test case numbers, 8 dial switches (sw7,.. Sw0) for inputting test data, and using LED lights or 7-segment tubes for output (If the number of LED lights is not enough, the output will be displayed alternately on the LED lights, with each part displayed for 2 seconds.) (Note that the high bit of the DIP switch and LED will be unified on the left, and the low bit will be unified on the right.)

NOTE 2: Use LED to output the binary form of the operand, and use 7-segment tube (or VGA) to output the hexadecimal form of the operation result

Scenario 2. Test case number	Test case description （The input data for test case 3 'b001-3' b011 is a signed number）	Example
3‘b000	Input an 8-bit number, calculate and output the number of leading zeros(the number of leading zeros of 8’ b00010000 is 3)	While input 8’b1101_0011, the value displayed on the output device should be 0 While input 8’b0001_0011, the value displayed on the output device should be 3 While input 8’b0000_0000, the value displayed on the output device should be 7
3‘b001	Input a 16 bit IEEE754 encoded half word floating-point number, round it up, and output the rounded result	While input 16’b0_10010_1001010000 (IEEE 754 encoded half word floating-point on 12.625), after rounding up, the value displayed on the output device should be D

3‘b010	Input a 16 bit IEEE754 encoded half word floating-point number, round it down, and output the rounded result	(IEEE 754 encoded half word floating-point on 12.625), after rounding up, the value displayed on the output device should be C
3‘b011	Input a 16 bit IEEE754 encoded half word floating-point number, rounding it, and output the rounded result	While input 16’b0_10010_1001010000 (IEEE 754 encoded half word floating-point on 12.625), after rounding up, the value displayed on the output device should be D
3‘b100	Input numbers a and b(each of them is 8-bit) , perform addition operations on a and b. If the sum exceeds 8 bits, remove the high bits and add them up to the sum. then inverting the sum, output the result	While input 8’B1111_1011 as a, input 8’B0111_1011 as b，calculate sum1 by using : sum1=a+b=9’b1_0111_0110, here the value of sum1[8] is 1， add sum1[8] to sum1 as sum2, sum2 = 1’b1+8’b0111_0110=8’b0111_0111; Invert sum2 8’b0111_0111, the final result is 8’b1000_1000，the value displayed on the output device should be 88
3‘b101	Input 12 bit data in little endian mode from the dial switch and present it on the output device in big endian mode	While input 12’b0011_1010_1100（here 8’b1011_1100（8’HAC）is byte0，8’b0000_0011(8’H03) is byte1）, output the data in big endial, the result displayed on the output device should be （from left to right） AC0 or C03（both AC0 and C03 are ok ） NOTE, in this test case, input 16bit data is also permitted, in this case, while input 16’b0101_0011_1010_1100（8’b1011_1100（8’HAC）is byte0，8’b0101_0011(8’H53) is byte1）, output the data in big endian, the result displayed on the output device should be（from left to right）AC53

3‘b110	Calculate the number of Fibonacci numbers which are smaller than the input data in a recursive manner, record the number of times the stack is pushed and pushed, and display the sum of the push and push times on the output device	Due to the different implementations, there is no unique answer to this question Whether the on board test results of the test case and the modified version of the test case (using ‘ecall’ to process input and output) running on RARAS are reasonable and consistent would be checked.
3‘b111	Calculate the number of Fibonacci numbers which are smaller than the input data in a recursive manner, record the push and pop data, display the push data on the output device(each push data display for 2-3 seconds (note that here not focus on the push and pop of the value of ‘ra’ register)	Due to the different implementations, there is no unique answer to this question Whether the on board test results of the test case and the modified version of the test case (using ‘ecall’ to process input and output, there is no need to hold each displayed number for 2-3 seconds) running on RARAS are reasonable and consistent would be checked.

三、Some tips about the test case on pipeline

Description	Instructions
1． How to handle a hazard(by using ALU-ALU forwarding)	Sub x2,x1,x3 And x12, x2,x5
2． How to handle a load-use hazard(by using MEM-ALU forwarding)	Lw x2, 8(x3) And x12, x2, x5
3． Two harzards	Sub x2,x1,x3 And x12, x2,x5 Or x13,x6, x2