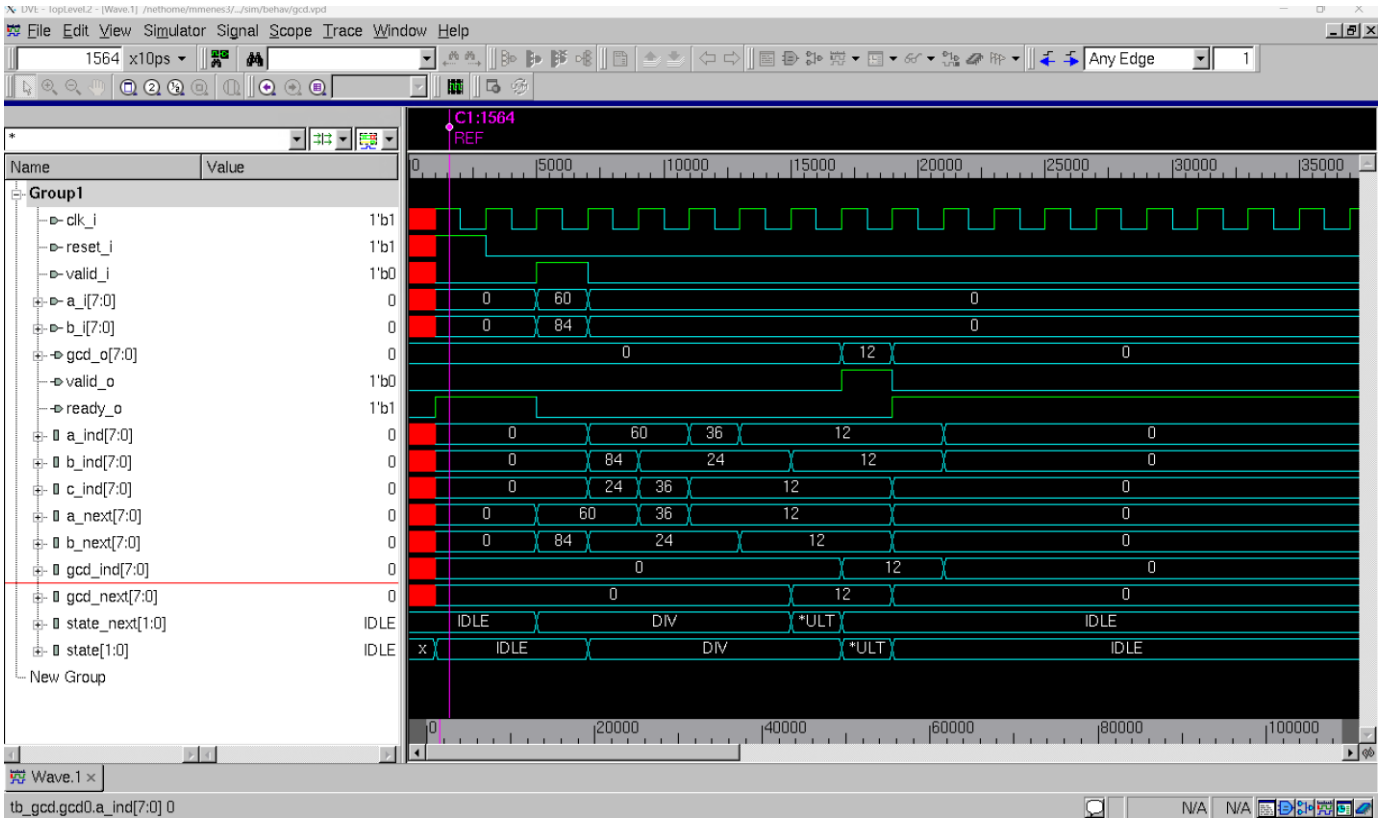
GCD Module

Given two inputs A and B the circuit successfully identifies the greatest common divisor between the 2:

```
Contains Synopsys proprietary information.
Compiler version W-2024.09_Full64; Runtime version W-2024.09_Full64;  Feb  2 21:
44 2025
Message: From $vcdpluson at time 0 in file ./tb_gcd.sv line 32: [VCD+-SVFN]:
Setting VPD File by "+vpdfile+" switch to ./gcd.vpd.

VCD+ Writer W-2024.09_Full64 Copyright (c) 1991-2024 by Synopsys Inc.

--------------Beginning Simulation!--------------

--------------Initializing Signals---------------

GCD of 60 and 84
Result:  12


-------------Finished Simulation!----------------

$finish called from file "./tb_gcd.sv", line 51.
$finish at simulation time              109000
           V C S   S i m u l a t i o n   R e p o r t
Time: 1090000 ps
CPU Time:      0.230 seconds;       Data structure size:   0.0Mb
Sun Feb  2 21:44:56 2025
```

This testbench log shows the display prompts of the testbench. The result of having 60 and 84 as inputs is 12 accordingly.

Wave Diagram:



The functionality of the circuit will be explained by going through each clock period and explaining how my outputs match what is expected from the behavior of the circuit:

P1: The clock starts on high, and the reset input is high. Given that reset is active, the reset logic of the sequential part is activated and resets the values (in this case this is done for setting the values) while also setting the current state as IDLE (the circuit still would've done it since IDLE is the default state). The IDLE logic sets next state as IDLE given that we have asserted our input.

P2: Now that reset is low the current state is not forced to IDLE. However the next state logic sets the state to IDLE and the IDLE logic starts. Given that there hasn't been any valid input.

P3: Now that valid_i is 1 the input from a_i and b_i is set and thus ready_o is set to 0 while loading a_next and b_next with those values to be assigned to the internal signals of a and b (since we can't operate using our inputs). Next state is set do Divide

P4: Now that we've reached the state of divide the values on a_ind and b_ind have been set and a rest is performed. C contains the result of the rest, which is 24 and as we can see the logic made b_next to be set to 24 given that it had the largest value. The algorithm works successfully and a_next remains unchanged.

P5: After the former clock period a_ind and b_ind are updated with their next values and the operation is performed, the difference is 36 and this time a_next is equal to c_ind given that it had the largest value.

P6: The new values are loaded into a_ind and b_ind and the result is 12, it gets loaded into A given that it is the largest value still. B_next remains unchanged.

P7: Now b_next is 12

P8: 12 is loaded into b_ind and the 4th case of B happens as the difference of both a_ind and b_ind is 0. Therefore gcd_next is loaded with the value of a_ind (12) and the next state will be set to Result, given that we've reached the greatest common divisor.

P9: The signal valid output is asserted true given that the gcd is greater than 0 (every case must be handled) and the former gcd_next value has been loaded into gcd_ind which is assigned to gcd_o (it wouldn't have been assigned if the output would've been invalid). The waveform now displays the correct result of GCD and the next state is set to IDLE which eventually resets the values of all internal signals, triggering ready_o to be 1 indefinitely (until a new input is written).