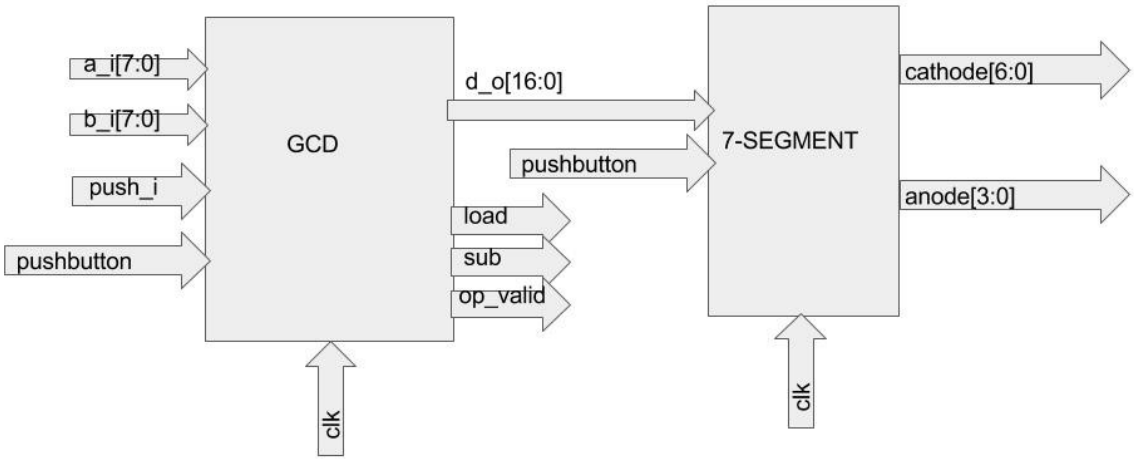
Lab Exercise 5: GCD calculator

Design and implement a circuit for calculating GCD of two 2-digit numbers

The aim of this exercise is to learn how to sequence operations with a clock and to use a previously designed module as a component in the VHDL design by instantiating it.

Exercise description:

The algorithm to be used here to compute GCD is repeated subtraction of the smaller number from the larger number, till the two become equal. Both the numbers are continuously being displayed on 7 segment display in integer format. The numbers are taken in BCD format from the slide switches. In BCD representation of a decimal number, each decimal digit is independently encoded as a 4 bit binary number. Thus a 2-digit decimal number can be represented using 8 bits and two such numbers for GCD computation can be input using 16 slide switches.



The computation should proceed as follows. Check for validity of operands (each digit should be in the range 0-9 and no operand should be zero). If both the operands are valid, Load operands in two 8-bit registers when a push button is pressed. After loading, while the numbers are unequal, repeatedly replace the larger number by the difference of the two. Continuously display the contents of the registers.

Check for the validity of operands asynchronously purely based on combinational circuit from the input operands. In case of invalid operands, the event of pressing push button should **NOT** trigger GCD calculation. This implies that LOAD and SUB should be “zero”. For case of valid operands, set LOAD and reset SUB asynchronously when the push button is pressed. (“LOAD” and “SUB” are two flip-flops used to decide what needs to be done at a given point

of time). After this initial asynchronous assignment, comes the sequential operation. At every clock do the following. Use a slow clock, say 0.5 Hz, for convenience of observation.

If LOAD = 1:

- Load operands from slide switches into registers
- Set SUB
- Reset LOAD

If SUB = 1:

- If operands are unequal, replace the larger number by the difference of the two
- Otherwise reset SUB

Display LOAD, SUB and operand valid condition on LEDs.

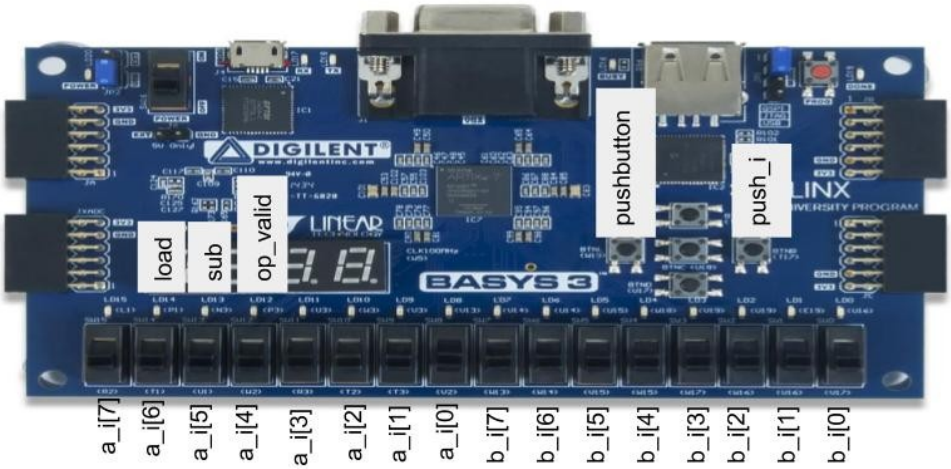
Comparison and subtraction logic is as follows. Let the two operands be A = a_1 a_0 and B = b_1 b_0, and the difference of the two be D = d_1 d_0, where the subscript 0 refers to the lower significant digit and the subscript 1 refers to the higher significant digit. Clearly,

$a_1 = b_1$ AND $a_0 = b_0$ means A = B
$a_1 > b_1$ OR ($a_1 = b_1$ AND $a_0 > b_0$) means A > B
$a_1 < b_1$ OR ($a_1 = b_1$ AND $a_0 < b_0$) means A < B

Suppose A > B. Then

if $a_0 \geq b_0$ then $d_0 = a_0 - b_0$ and $d_1 = a_1 - b_1$
otherwise $d_0 = 10 + a_0 - b_0$ and $d_1 = a_1 - b_1 - 1$

Do the design using VHDL.



pushbutton is used to select clock rate
pushbutton = 1 selects the fast clock mode. For Simulation

pushbutton = 0 selects the slow clock mode. For Synthesis
For verifying the synthesized circuit board, do not press pushbutton

You need to fill in the constraints **YOURSELF** in the sample constraints file provided.

Pin Name	**Description**	**Purpose**
clk	Input(from Basys3 internal clock	clocking
a_i[7:0]	Input	BCD input from switches
b_i[7:0]	Input	BCD input from switches
push_i	Input	Controls loading of operands
pushbutton	Input	Selects clock rate
load	Output	Indicates the loading of operands in registers
sub	Output	Indicates GCD computation in progress
op_valid	Output	Indicates the validity of operands
cathode[6:0]	Output	To display the integer digit
anode[3:0]	Output	To enable the digit on display
Top Level module name	lab5_gcd	

Note: When using pushbutton=1 for **simulation**, the Seven Segment Display counter runs at normal clock and in 4 cycles, each digit is displayed once. That is, refresh rate of seven segment display is one fourth of the normal clock frequency. Thus, here the GCD circuit clock frequency should be same as the refresh rate. This is necessary so that you are able to see each step of GCD calculation in simulation.