Lab. Practice 4 (Extra part)

Optimized sequential 4x4-bit multiplier

Objective

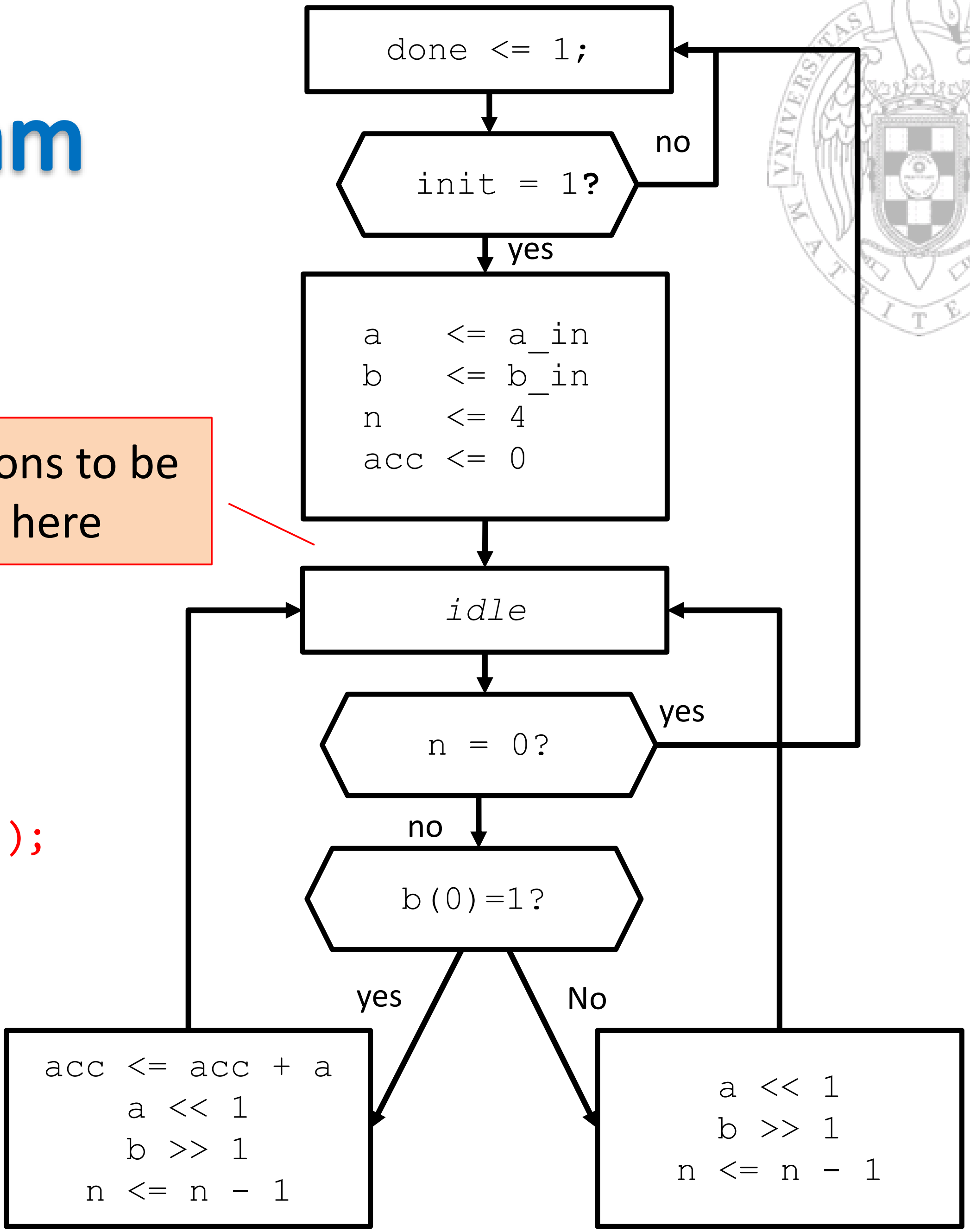
- To reduce the number of iterations needed to calculate the multiplication based on additions.
  - Both operands will be compared and the one with lowest length will be used as index to determine the number of iterations.
  1. If b is the operand with the lowest length, no changes are needed.
  2. If a is the operand with the lowest length, registers a and b need to switch their contents.
  - If case 2 happened, an additional LED (*Switched*) will be activated.

ASM diagram

- ASM multiplier:

```
a    = a_in;
b    = b_in;
~~n    = 4;~~
acc = 0;
Switched = 0;
if (length(a) < length(b)){
  a ⇔ b;   // atomically
  Switched = 1;
}
n = MIN(length(a), length(b));
while( n > 0 ){
  if( b(0) == 1 )
    acc = acc + a;
  a << 1;
  b >> 1;
  n --;
}
```

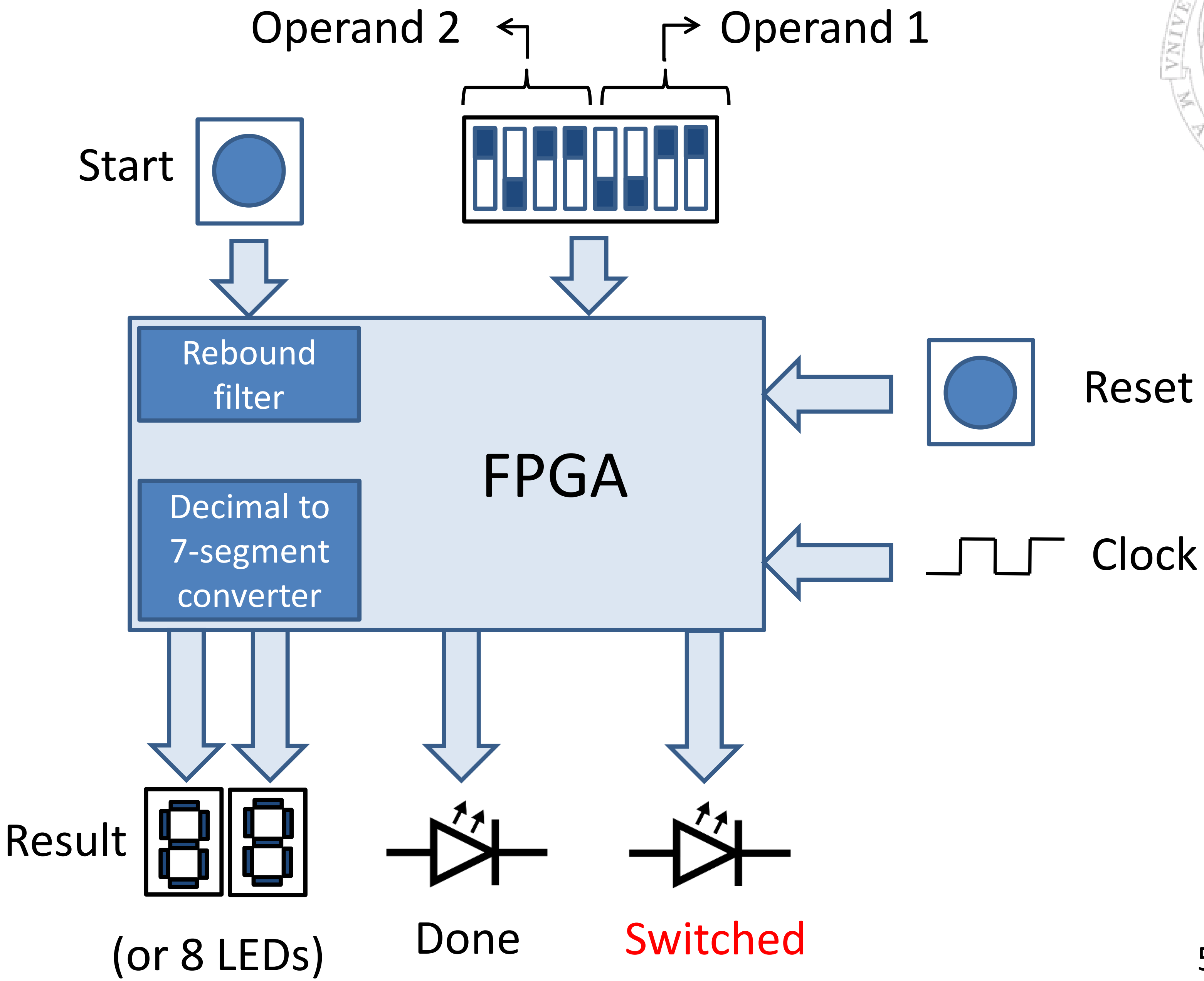
Modifications to be done from here



Modifications in datapath

- Two combinatorial modules at the output of numbers a and b to know the effective length of the operands. For instance:
  - a = “0110” → length = 3 bits
  - b = “1011” → length = 4 bits
- A comparator to check if length(a)<length(b).
- You can also do this with 1 module whose inputs are a and b, and whose output (1 bit) compares the length of both.
- A combinatorial module that calculates the minimum of 2 numbers (a and b).
- To multiplex inputs of registers a and b to make possible the switching of values (a⇔b).
- A flip-flop whose output is connected to the “*Switched*” LED.

TOC



Operand 2

Operand 1

Start

Rebound filter

Decimal to 7-segment converter

FPGA

Reset

Clock

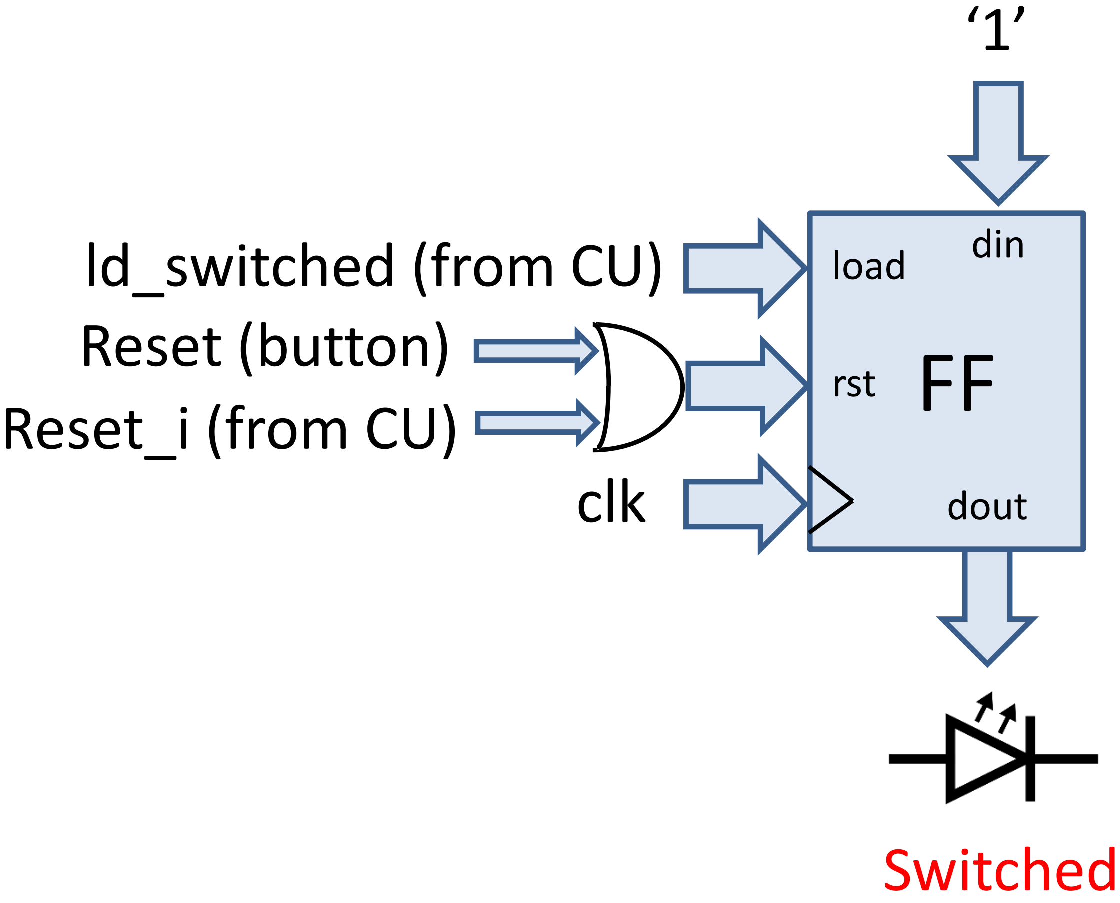
Result

(or 8 LEDs)

Done

Switched

Flip-flop for *Switched*


'1'
ld_switched (from CU)
Reset (button)
Reset_i (from CU)
clk
load
din
rst
FF
dout
Switched