Laboratory 3

(Due date: **002**: March 1st, **003**: March 2nd)

OBJECTIVES

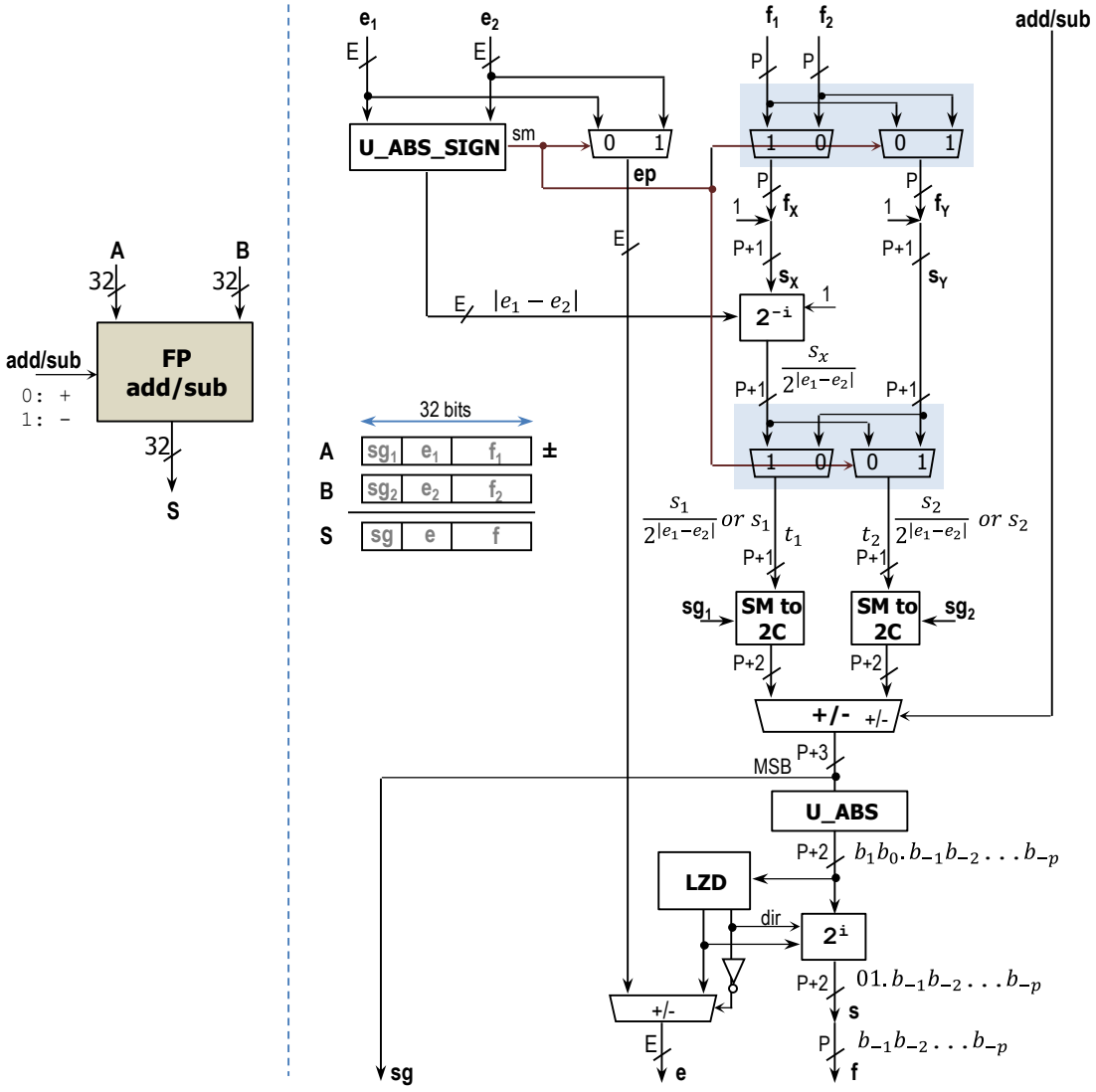
- ✓ Implement a large combinational circuit using the Structural Description in VHDL.
- ✓ Introduce floating point and fixed point representations for VHDL implementation and Vivado simulation.

VHDL CODING

- ✓ Refer to the Tutorial: VHDL for FPGAs for parametric code for: adder/subtractor and busmux.

FIRST ACTIVITY: FLOATING POINT ADDER/SUBTRACTOR (100/100)

- Implement the following single-precision (E=8, P=23), floating point adder/subtractor. The circuit only works for ordinary numbers generating ordinary numbers (e.g.: the cases $S = A \pm B = 0$, $A = 0$, or $B = 0$ are not considered by this circuit). The exponents in the circuit are biased exponents, so they always are positive numbers.



- **U_ABS_SIGN**: This circuit computes the absolute value of the difference of two unsigned numbers (e_1 and e_2). It also generates the signal sm.
  $e_1, e_2 \in [0, 2^E - 1] \rightarrow e_1 - e_2 \in [-(2^E - 1), 2^E - 1] \rightarrow |e_1 - e_2| \in [0, 2^E - 1]$.
  So, $|e_1 - e_2|$ only needs E bits as an unsigned number.
  $e_1 \geq e_2 \rightarrow sm = 0, ep = e_1, f_x = f_2, f_y = f_1$
  $e_1 < e_2 \rightarrow sm = 1, ep = e_2, f_x = f_1, f_y = f_2$

Instructor: Daniel Llamocca
TAs: Michael Lohrer, Jing Wu

sm	ep	s_x	s_y	t_1	t_2
0	e_1	$s_2 = 1.f_2$	$s_1 = 1.f_1$	s_1	$\frac{s_2}{2^{\lvert e_1 - e_2 \rvert}}$
1	e_2	$s_1 = 1.f_1$	$s_2 = 1.f_2$	$\frac{s_1}{2^{\lvert e_1 - e_2 \rvert}}$	s_2

s_x: operand that gets divided by $2^{|e_1-e_2|}$

- **Barrel shifter 2^{-i}**: This circuit performs alignment of s_x, where we always shift to the right by $|e_1 - e_2|$ bits. Use the VHDL code `mybarrelshifter.vhd` with `dir = '1'` and `mode = "LOGICAL"`.

- **SM to 2C**: This block converts a number represented as a sign and magnitude into a 2C number. If sign is '0', then we append a 0 to the MSB. If the sign is '1', we get the negative number in 2C representation. This increases the bit-width to $P + 2$ bits.

- **Main adder/subtractor**: This circuit operates in 2C arithmetic. Note that it is implied that we need to sign-extend the $(P + 2)$-bit operands to $P + 3$ bits.
  Input operands $\in [-2^{P+1} + 1, 2^{P+1} - 1]$, Output result $\in [-2^{P+2} + 2, 2^{P+2} - 2]$.

- **U_ABS block**: It takes the absolute value of a number represented in 2C arithmetic. The output is provided as an unsigned number. The absolute value $\in [0, 2^{P+2} - 2]$, this only requires $P + 2$ bits in unsigned representation.

- **Leading Zero Detector (LZD)**: This circuit outputs a number that indicates the amount of shifting required to normalize the result of the summation. It is also used to adjust the exponent. This circuit is commonly implemented using a priority encoder. `result` $\in [-1, p]$. The result is provided as sign and magnitude. Use the following code: `myLZD.vhd`.

`result`	`output`	`sign`	Actions
$[0, p]$	$sh \in [0, p]$	0	The barrel shifter needs to shift to the left by sh bits. Exponent adder/subtractor needs to subtract sh from the exponent ep.
-1	$sh = 1$	1	The barrel shifter needs to shift to the right by 1 bit. Exponent adder/subtractor needs to add 1 to the exponent ep.

- **Exponent adder/subtractor**: The figure is not detailed. This circuit operates in 2C arithmetic; as the input operands are unsigned, we zero-extend to $E + 1$ bits. Note that for ordinary numbers, $ep \in [1, 2^E - 2]$. The $(E + 1)$-bit result (biased exponent) cannot be negative: at most, we subtract P from ep, or add 1. Thus, we use the unsigned portion: E bits (LSBs).

- **Barrel shifter 2^{i}**: This performs normalization of the final summation. We shift to the left (from 0 to P bits) or to the right (1 bit). Use the VHDL code `mybarrelshifter.vhd` with `mode="LOGICAL"` (unsigned input data), `dir=sign(LZD)`.

- **VIVADO DESIGN FLOW FOR FPGAs:**
  - ✓ Create a new Vivado Project. Select the **XC7A100T-1CSG324 Artix-7 FPGA** device.
  - ✓ Write the VHDL code for the 32-bit floating point adder subtractor. Utilize the Structural Description: create a separate file for the components (2-to-1 bus mux, SM to 2C, U_ABS, LZD, Barrel shifter, adder/subtractor, U_ABS_SIGN), and the top file (where you will interconnect all the components).
  - ✓ Write the VHDL testbench to test the following cases:

```
60A10000 + C2F97000 = 60A10000
40B00000 + C2FA8000 = C2EF8000
42FA8000 + C0E00000 = 42EC8000
10DAD000 - 90FAD000 = 116AD000
3DE38866 - B300D959 = 3DE3886A
60A10000 - 60A1F000 = DCF00000
```

  - ✓ Perform Functional Simulation and Timing Simulation of your design. **Demonstrate this to your TA**.
    Note that when testing, it might be very useful to represent the inputs and output in single floating point precision. Or we might also want to represent the intermediate signals not only as integer numbers but also as fixed point numbers. You can use the `Radix → Real Settings` in Vivado simulator window to do so.

- Submit (as a .zip file) all the generated files: VHDL code files and VHDL testbench to Moodle (an assignment will be created). DO NOT submit the whole Vivado Project.

TA signature: _____ **Date:** _____

Instructor: Daniel Llamocca
TAs: Michael Lohrer, Jing Wu