ENS - Sysnum
TD 2

Building an Arithmetic-Logic Unit

10 nov. 2020

■ ■ ■

I - Warm up : Integer Arithmetic 101

Exercice 1. (Mechanical Turk)

1. Compute $4A_{16} + 21_{16}$ as well as $CE_{16} + D8_{16}$

2. Compute $69_{10} - 50_{10}$ as well as $73_{10} - 92_{10}$ using two's complement over 8 bits

3. Using the fact that $-B = \neg B + 1$, solve the previous question using only additions.

In what follows, we will build a 1-bit full adder, using two half-bit adders

Exercice 2. (1-bit Full-adder)

1. What equations should a half-adder follow ? Draw its corresponding circuit.

2. Compute the Karnaugh map for a full-adder.

3. How to make a full adder using two half adders ? Draw its corresponding circuit by abstracting the half adders.

4. What is the critical path of this circuit (in terms of gates and transistors) ? What can we say about the half-adders' carries ? How to exploit this property to redesign the carry computation ?

II - Time to work : Building an ALU

Exercice 3. (n-bit Adder)

1. Use a 1-bit full-adder to build a 4-bit full-adder.

2. What is the critical path of this circuit ?

3. Can we generalize this design to any n-bit adder ?

Exercice 4. (Carry-lookahead Adder) We can better handle the carry operation by splitting our 1-bit full adder's carry path into two, a *generate* signal indicating if a carry is generated from the inputs $G = A \cdot B$, and a *propagate* signal indicating if a carry is propagated from its inputs carry $P = A \oplus B$. Thus, we have $C_{out} = G + P \cdot C_{in}$.

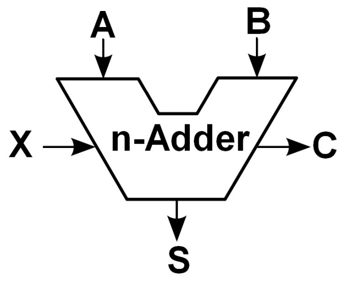
1. Draw the new 1-bit full-adder.

2. (Generate) For building a 2-bit carry-lookahead adder, we have to combine the signals P_0, G_0 and P_1, G_1 of two 1-bit adders into a signal P_{01}, G_{01}. With equations $P_{01} = P_1 \cdot P_0, G_{01} = G_1 + G_0 \cdot P_1$? How can we compute $G_{0(n-1)}$ for $n = 2^k$, with logarithmic critical path ?

3. (Carry-gen) Finally, we need to compute for each 1-bit adder its carry-in (except for the least significant bit), by chaining the equation $C_1 = G_0 + P_0 \cdot C_0$. Generalizing this last equation to 2-bits gives : $C_2 = G_{01} + P_{01} \cdot C_0$. How can we compute the input carries for every 1-bit adder $C_0, ..., C_{n-1}$ with logarithmic critical path ?

4. By combining both the Generate and Carry-gen operations into a single module, we may reduce the set of equations. What equations should the GC module follow ? How does a carry-lookahead n-adder look like ?

Exercice 5. (Making the ALU)



In what follow, we will abstract our n-bit adder as a black box having inputs two n-bit numbers A and B, a bit X (carry in), and outputting one n-bit number S, and one bit C (carry).

1. How can make this n-adder to perform both additions and subtractions depending on an input bit Sub ?

2. What happens when we have a signed overflow, add a new output bit V to signal these ?

3. How to modify this circuit to perform the NOT operation depending on an input bit Not ?

4. How to extend the circuit to perform AND, OR and XOR ?

5. Extend your circuit with a new flag N, equal to true if the output S is negative, and another Z signaling whether the output S is null.