Exam for "Systèmes Digitaux" course

Tuesday January 15, 2019

Abstract

This exam is made up of two problems. It is better to answer to some of them in depth than all of them superficially.

The exam duration is 4 hours. The maximum number of pages is 6. You cannot use class material.

1 CMOS logic gates: CMOS equality gate

We recall that a Negative MOS (NMOS — ) transistor is:

- *open* if the gate input is equal to one, and
- *closed* otherwise.

A Positive (PMOS — ) transistor behaves the opposite way.

Q1.1.

We recall that the "exclusive or" (also known as `XOR` function) is defined as:

a	b	$z = a \oplus b$ (`XOR` of a and b)
0	0	0
0	1	1
1	0	1
1	1	0

We denote by "gnd" the ground and by "vdd" the voltage supply.

Can the XOR function be implemented in one CMOS logic gate? If so, give a circuit implementation based on transistors. If not, explain how to obtain the XOR function using multiple CMOS gates?

Q1.2.

Derive a structure for the XOR gate using only `NAND` gates.

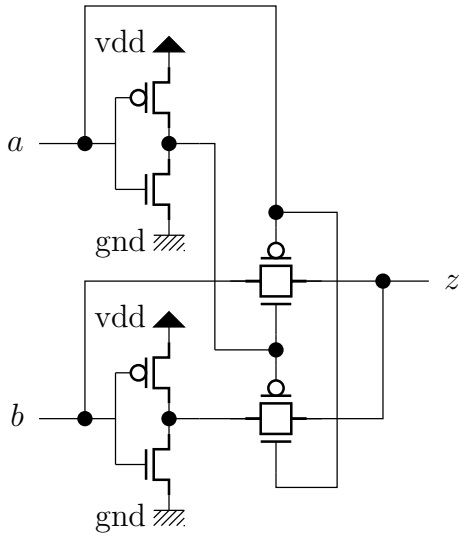

Figure 1: Structure in transistor of a digital cell

Q1.3.

Study the transistor-based circuit depicted in Fig. 1.

Explain how it works and what function it implements. Comment on its performance.

Q1.4.

"Wave Dynamic Differential Logic" (WDDL) is a way to realize circuits with constant activity. For this purpose, each bit a is represented as a pair of two bits, namely a_T ("a true") and a_F ("a false"). Computations in WDDL consist in the alternation of two phases, denoted "precharge" and "evaluation".

- In the precharge phase, all the nets are forced to zero, that is for net a, we have $a_T = a_F = 0$. This property holds for the inputs, but also for any intermediate net, until the output nets of any function. Hence the qualification of "wave" for this WDDL logic style.
- In the evaluation phase, either a_T is asserted, or (exclusively) a_F. We thus have $a_T \oplus a_F = 1$. Again, this property must hold for all the nets in the circuit.

This way, in a WDDL netlist of $2n$ wires, the number of bits set is 0 in the precharge phase and n in the evaluation phase. Similarly, the number of bits that toggle is constant, namely n (irrespective of the phase). The "Wave" property of WDDL logic implies that when inputs are in precharge (resp. evalua-

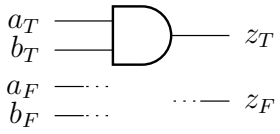

Figure 2: Structure of the WDDL AND gate (the false half is missing)

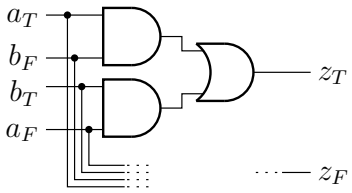

Figure 3: Structure of the WDDL XOR gate (the false half is missing)

tion), the output is in precharge (resp. evaluation). Notice that the alternation between the phases results from an external clock.

A positive gate is a CMOS gate that is equal to 0 when all its inputs are equal to zero, and that can only increase (i.e., the output can increase only when inputs increase).

Explain why the two-input AND is positive.

We aim to find the structure of a WDDL AND gate, whose inputs are a_T, a_F, b_T and b_F. As a hint, we provide half of the gate (namely, the logic which computes z_T) in Fig. 2. Complete this circuit.

Also, we seek the WDDL structure for the two-input XOR gate. Complete the second half of the gate represented in Fig. 3.

2 Arithmetic: modular operations

Consider a prime number p and an integer d. The ultimate aim is to compute $M^d \bmod p$, for a given $0 \leq M < p$. A typical application is for cryptography.

In a processor, an addition can be carried out in one clock cycle, whereas a multiplication usually requires multiple clock cycles.

Q2.1. Addition vs Multiplication

Give an explanation for this fact.

Q2.2. Consideration of signs

Consider an n-bit register, say with $n = 32$. An integer A can be:

- unsigned, meaning that $0 \leq A < 2^{32}$;
- signed, meaning that $-2^{31} \leq A < 2^{31}$.

Explain why it is possible to use the same hardware to perform the addition of two numbers modulo 2^{32}, but that it is not possible to reuse the same hardware to perform the multiplication of two numbers modulo 2^{32}. In practice, there is only one `add` mnemonic in assembly, which can serve for both unsigned and signed numbers. But there are two mnemonics for the product (namely `mul` for unsigned numbers, and `imul` for signed numbers).

Q2.3. Modular addition

We consider now a cryptographic algorithm such as RSA (Rivest-Shamir-Adelman). In this algorithm, operations modulo a prime number p are required. **For the sake of simplification, we assume that p fits in a 32-bit register. More precisely, we assume that p has its Most Significant Bit (MSB) set, that is $2^{31} \leq p < 2^{32} - 1$. In practical applications, p would be larger, e.g., be $1024 = 32 \times 32$ bit long. But for this exercise, we consider arithmetic on 32-bit registers.**

In the sequel, we focus on the computation of the multiplication of A and B in $\mathbb{F}_p$. Consider a processor operating on 32-bit words. The arithmetic operations are described below, considering unsigned numbers, i.e., $0 \leq A, B < 2^{32}$:

- `add`, which performs $A + B + c$, where c is the carry flag; the result is one register, and sets the carry if $A + B + c > 2^{32}$.
- `sub`, which performs $A - B - c$, where c is the carry flag, which acts as a borrow; the result is one register, and sets the c carry if $A - B - c$ is negative;

Explain how to compute $A+B \in \mathbb{Z}_p$, where $0 \leq A, B < p$, using instructions `add`, `sub` and (if required) conditional tests.

Q2.4. Modular multiplication

Give an algorithm to compute $AB \mod p$, where $0 \leq A, B < p < 2^{32}$, using the operation obtained previously for computing $A + B \in \mathbb{Z}_p$.

Q2.5. Exponentiation

Given an algorithm to compute $M^d \bmod p$ using addition and multiplication modulo p. This algorithm must be optimal, using at most $2\log_2(p)$ multiplications. Justify the complexity of your algorithm and, in particular, why it is possible to obtain a *worse case* complexity which is independent of the value of the exponent d.