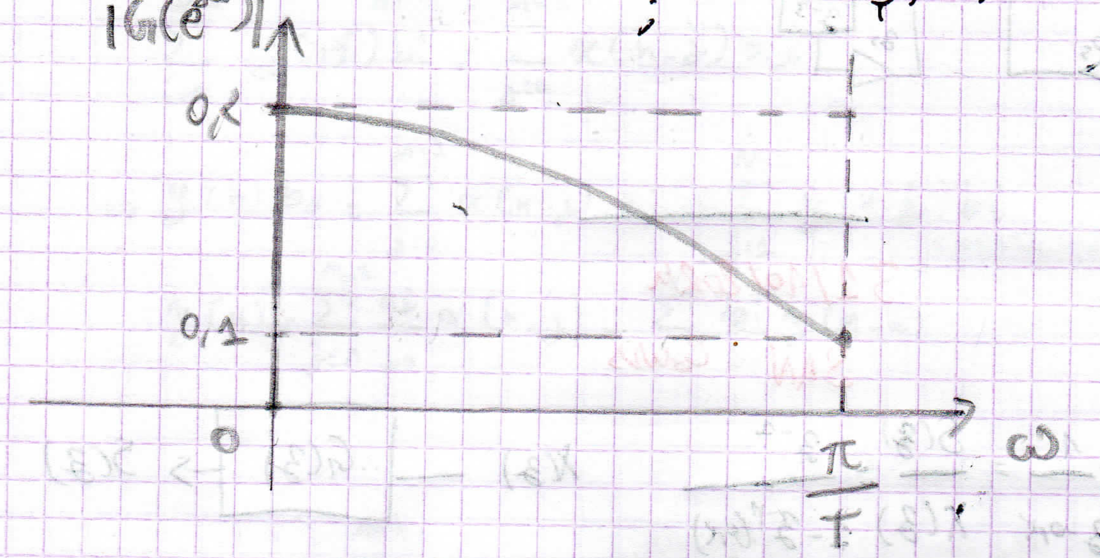
$= 0{,}5\,(\cos(\omega T) - j\sin(\omega T)) + 0{,}2$

$= [0{,}5\cos(\omega T) + 0{,}2] - j[\sin(\omega T)]$

$|G(e^{j\omega T})| = \sqrt{[0{,}5\cos(\omega T)+0{,}2]^2 + \sin^2(\omega T)} = \sqrt{0{,}13 + 0{,}12\cos(\omega T)}$

on travaille dans le domain $[0, \frac{f_e}{2}] \equiv [0, \frac{\pi}{T}]$

$G(0) = 0{,}5$ $\qquad$ $G(\frac{\pi}{T}) = 0{,}1$



5-11-2024

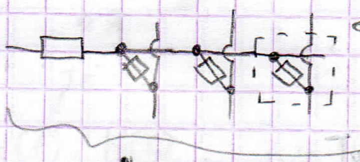
~~SAMC~~ FPGA cours

C.I

- → ASIC (application specific)
  - ← programable-logic
  - ← gate array
  - ← full costum IC
- → standard logic
  - → porte logic
  - → mux / demux
  - → HA / FA
  - → diviseur
  - → subtracteur

programable-logic:

- → pal programable array logic
- → CpLD complex programable logice device
- → FPGA

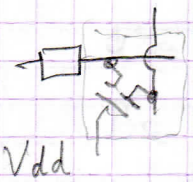


one time programming architecture

dans la programmation on envoi une tention très grand dans la fisible de (0/1), donc les 2 lien se deconnectea

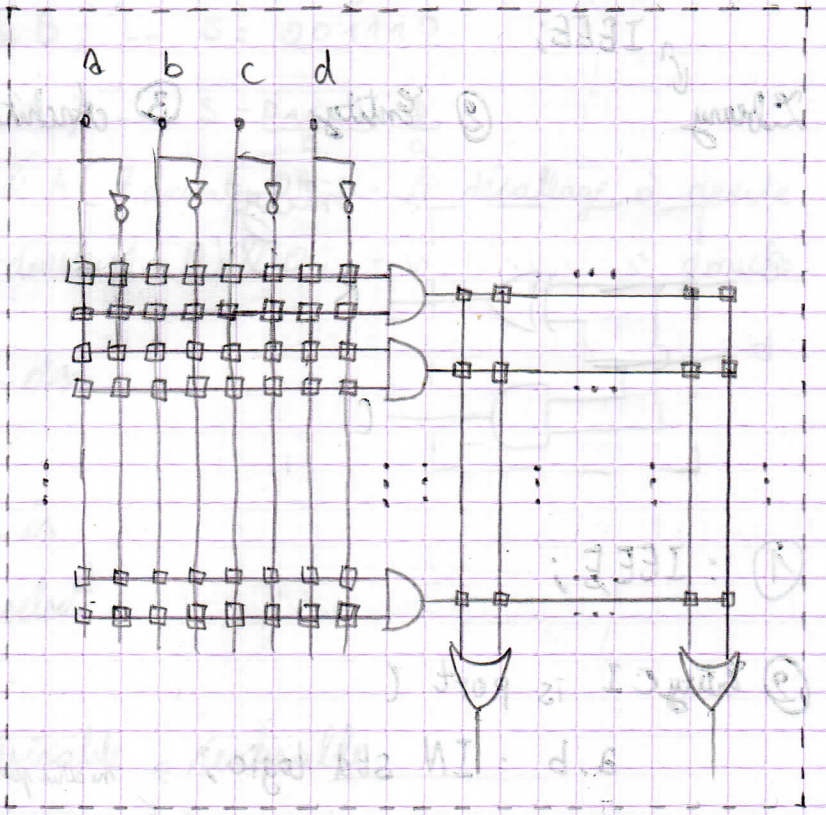
حرق الفيزيبل بـ تيار عالي

γ

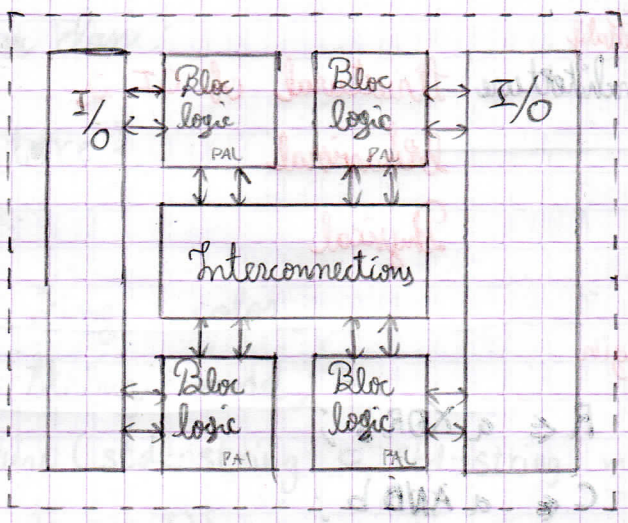


interconnection

cette architecture permet de programmer le CI plusieurs fois



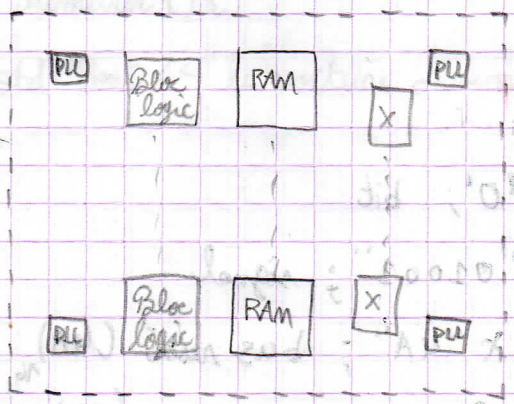
PAL



CPLD

Example

Réf : 5M160ZE64ASN

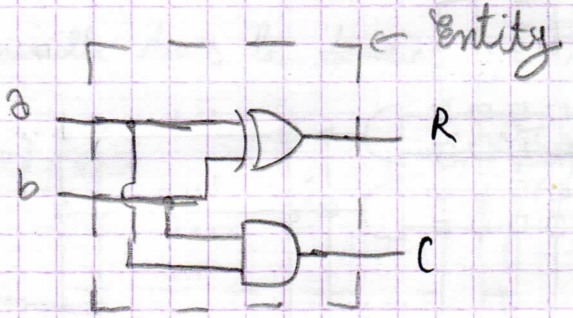


FPGA

VHDL :

→ Commentair : '-- c'est une commentair'

① Library (IEEE;) ② Entity ③ Architecture



① : IEEE ;

② Entity CI is port (

a, b : IN std logic ;

C, R : OUT std logic ← n'écrire pas la ; (à cause de la ")")

);

end ; CI (facultatif)

③ Architecture Structural of CI is
Behavioral
Physical

Begin

R ⇐ a XOR b ;

C ⇐ a AND b ;

end Arcb (facultatif)

a ⇐ '0' ; bit

a ⇐ "01001" ; signal

a ⇐ X "4A" ; bus resoi $(4A)_{16}$

a ⇐ 24 ; " " $(24)_{10}$

a <= '0' ;

B <= "01110";

S <= a & B ; -- S = 00 1110 (a = 0, B = 01110)

S <= B & a ; -- S = 01110 0 (B = 01110, a = 0)

S <= 0 & A(7 downto 0); -- A décallage à droite

S <= A(7 downto 0) & 0; -- " " " gauche

When - else

if_then

case - is

with - select

synthèsisable ≠ réalisable

POO - cours

création d'une classe :

```
class voiture {
    public :
        std:: string color;
        std:: String model;
        voiture (std:: string c, std:: string m):color(c), model(m) {}//I
        void demarer (){
            std:: cout << "la voiture démarre !" << std:: endl;
        }
};
```

Création d'un objet

```
voiture ma_voiture ("Rouge", "Toyota");
ma_voiture.démarer(); // appeler la méthod/fonction démarer
std::cout << "couleur : " << ma_voiture.color << std::endl;
```

On continue la Class //I

Constructeur :

```
voiture (std::string c, std::string m) : color(c), model(m){
    std::cout << "objet voiture créer !" << std::endl;
}
```

destructeur :

```
~ Voiture (){
    std::cout << "objet voiture détruit !" << std::endl;
}
```

6-11-2024

TAS // SAP

- Random variable

(ensemble of X_1 and X_2 respectively)

1. Independence principle : Considring $A=\{a_1, a_2, \dots a_N\}$ & $B=\{b_1, b_2, \dots, B_M\}$

It stipulates that, if 2 random variables X_1 & X_2 are independent, then the joint propability for X_1 to take the value a_k noted $P_r(X_1=a_k)$ and $P_r(X_2=b_i)$ is $\boxed{P_r(X_1=a_k, X_2=b_i) = P_r(X_1=a_k) \times P_r(X_2=b_i)}$

2. The exclusion Principle : If the 2 variables X_1 & X_2 cannot take ~~simult~~ simultaneously the value $X_1=a_k$ & $X_2=b_2$ then the probability is

$P_r(X_1=a_k, X_2=b_i) = P_r(X_1=a_k) + P_r(X_2=b_i)$

→ Probability

1) True probability : $P_r(X = a_k) = \lim_{N_t \to \infty} \frac{n_{ak}}{N_t}$

2) Estimated ≈ : $Pr(X = a_k) = \frac{n_{ak}}{N_e}$

→ The average of a random variable

1) The true average: assume that $A = \{a_1, a_2, \dots, a_N\}$ ensemble of N possible values, the average is :

$$m_X = \left. \frac{\sum_{k=1}^{N} a_k \cdot n_{ak}}{N_t} \right|_{N_t \to +\infty} = \sum_{k=1}^{N} \left[a_k \lim_{N_t \to +\infty} \left(\frac{n_{ak}}{N_t} \right) \right] = \sum_{k=1}^{N} a_k P_r(a_k) = E[X]$$

espérance de X

$$= P_r(1) a_1 + P_r(2) a_2 + \dots + P_r(N) a_N$$