ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

LCIS
Laboratoire de Conception et d'Intégration des Systèmes

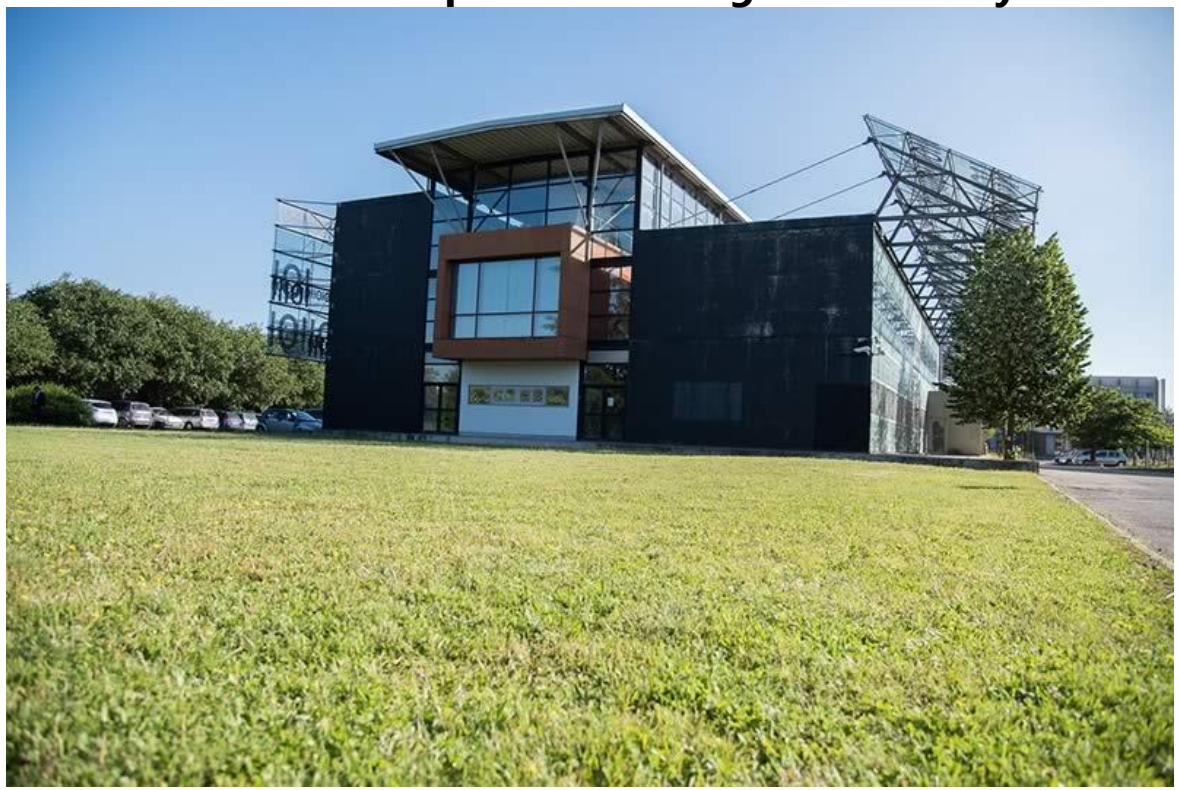
Protection des systèmes embarqués

STAGE 2A 2023 - 2024

Yassine HMIMOU

L'École des Ingénieurs Scientifiques

Laboratoire de Conception et d'Intégration des Systèmes

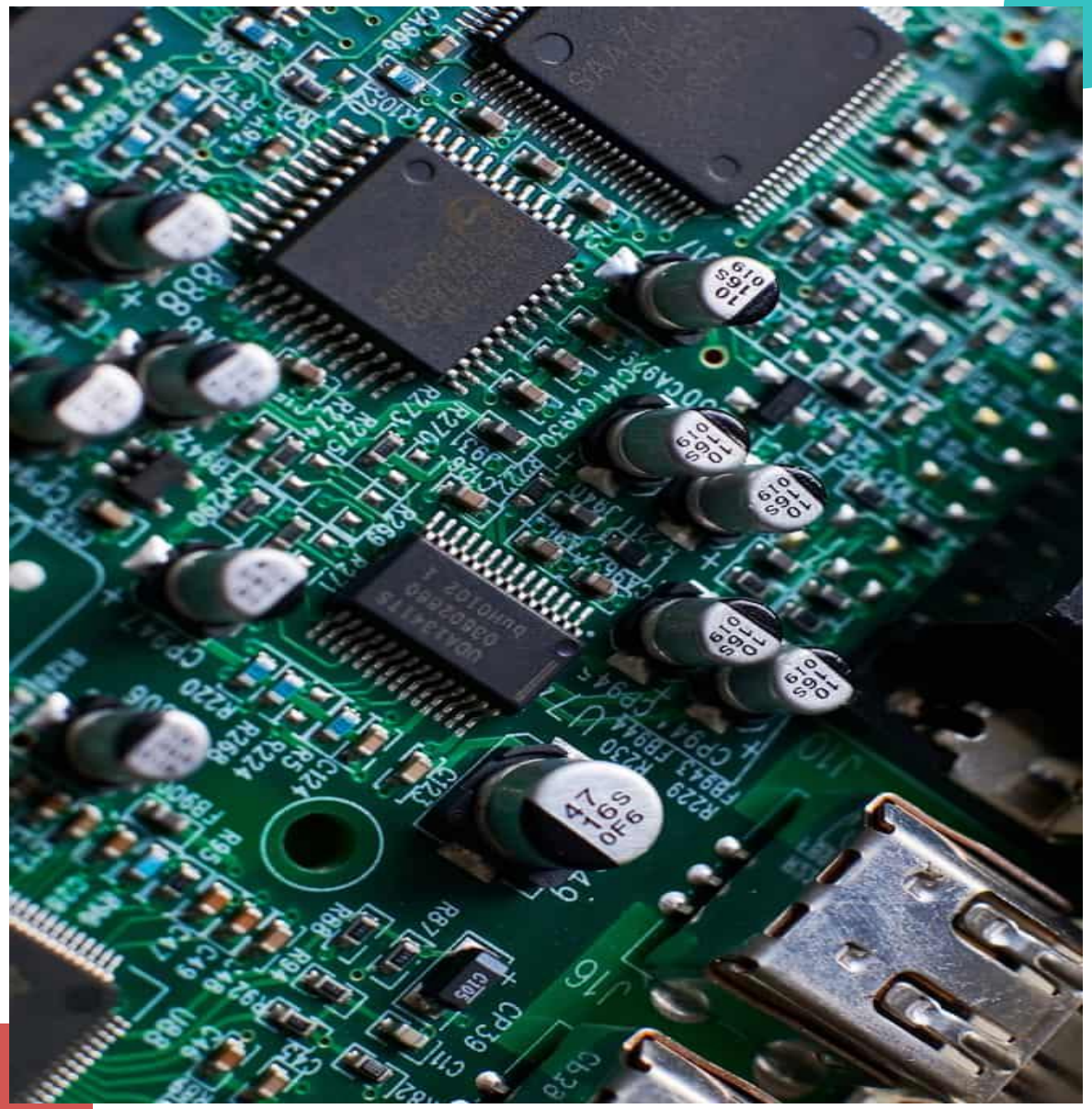


https://www.scattererid.eu/the-lcis-laboratory/

Présentation du laboratoire

- Université Grenoble Alpes
- Octobre 1996, Janvier 2003
  - Sûreté et sécurité des systèmes embarqués et distribués
  - Modélisation, analyse et supervision des systèmes complexes ouverts
  - Systèmes radiofréquences sans fil communicants

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

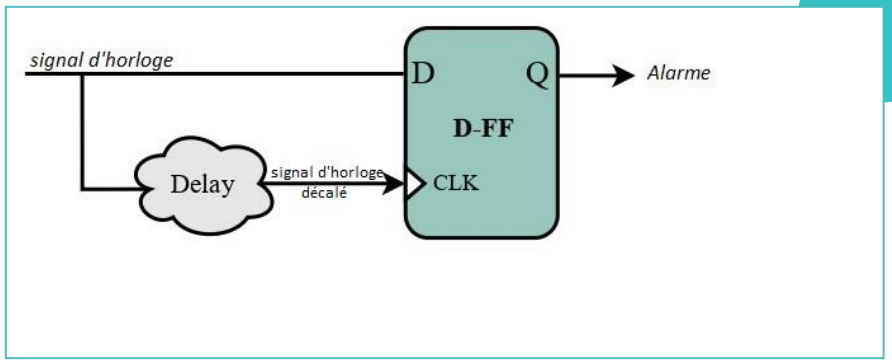


https://en.wikipedia.org/wiki/Embedded_system#:~:text=An%20embedded%20system%20is%20a,larger%20mechanical%20or%20electronic%20system.

INTRODUCTION

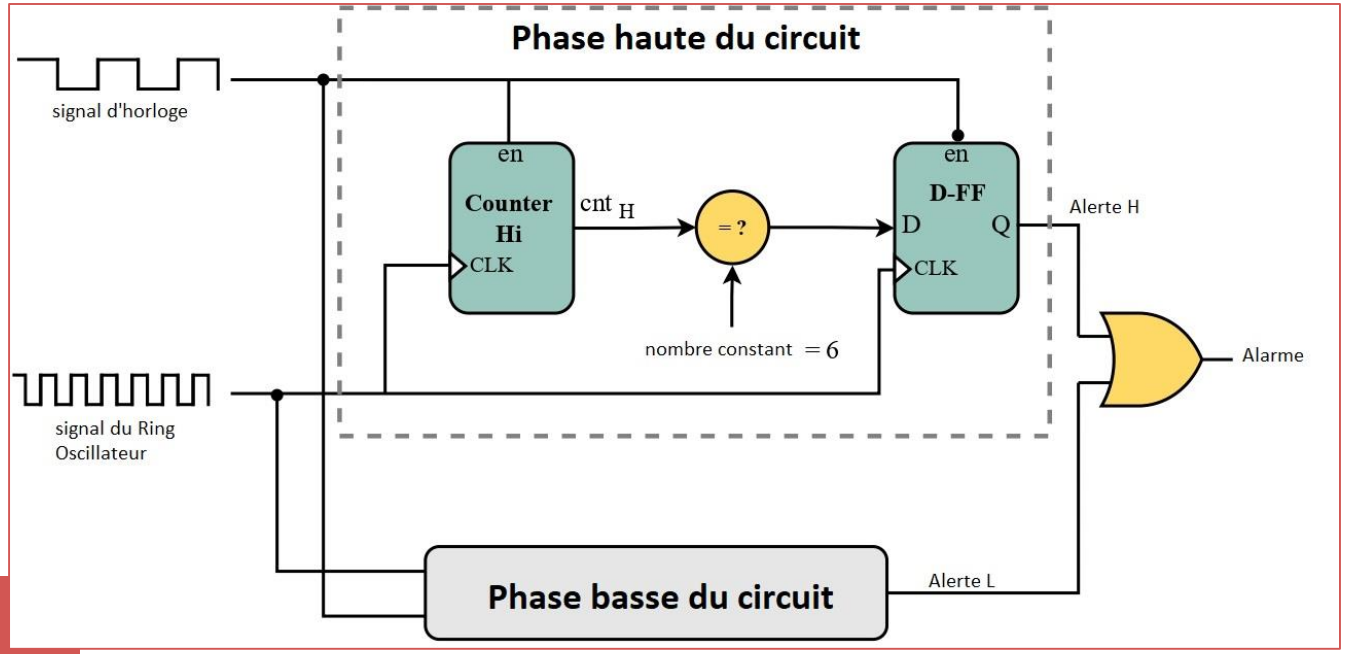
- Les systèmes embarqués
- Les attaques matérielles
  - ➢ Les attaques par canal auxiliaire
  - ➢ Les attaques par injection de faute

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE



Problématique

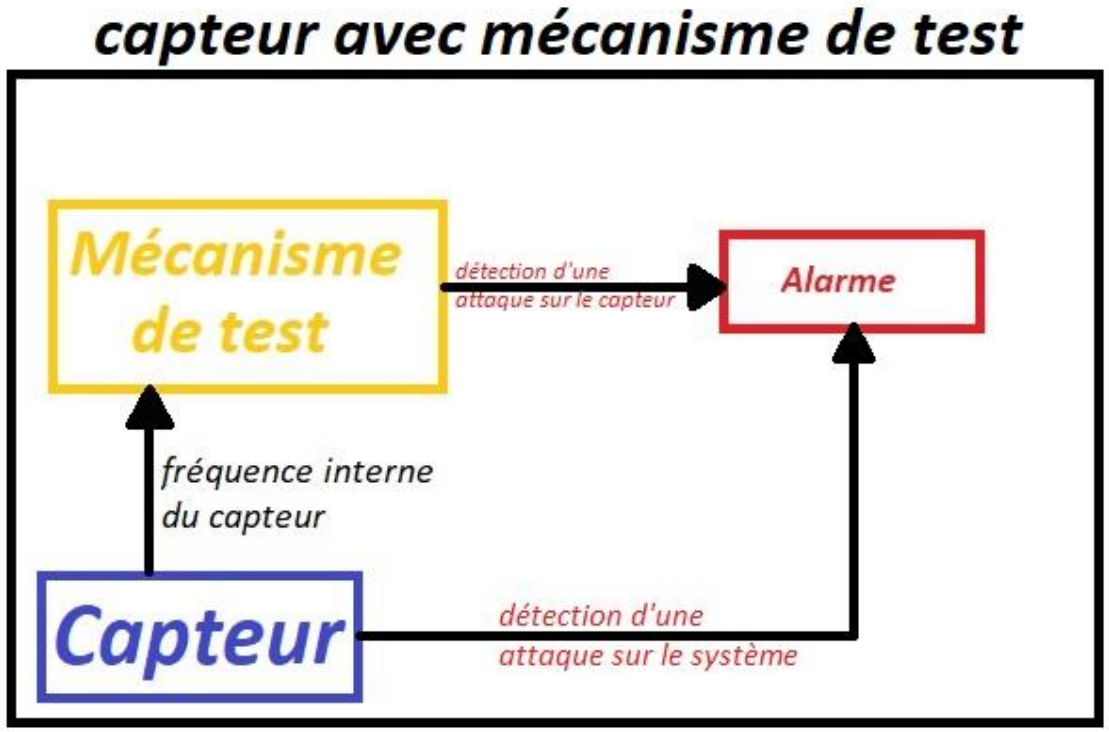
- Les détecteurs analogiques
- Les détecteurs numériques



- ➢ Le détecteur basé sur le retard
- ➢ Le détecteur basé sur l'oscillateur en anneau

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Contribution

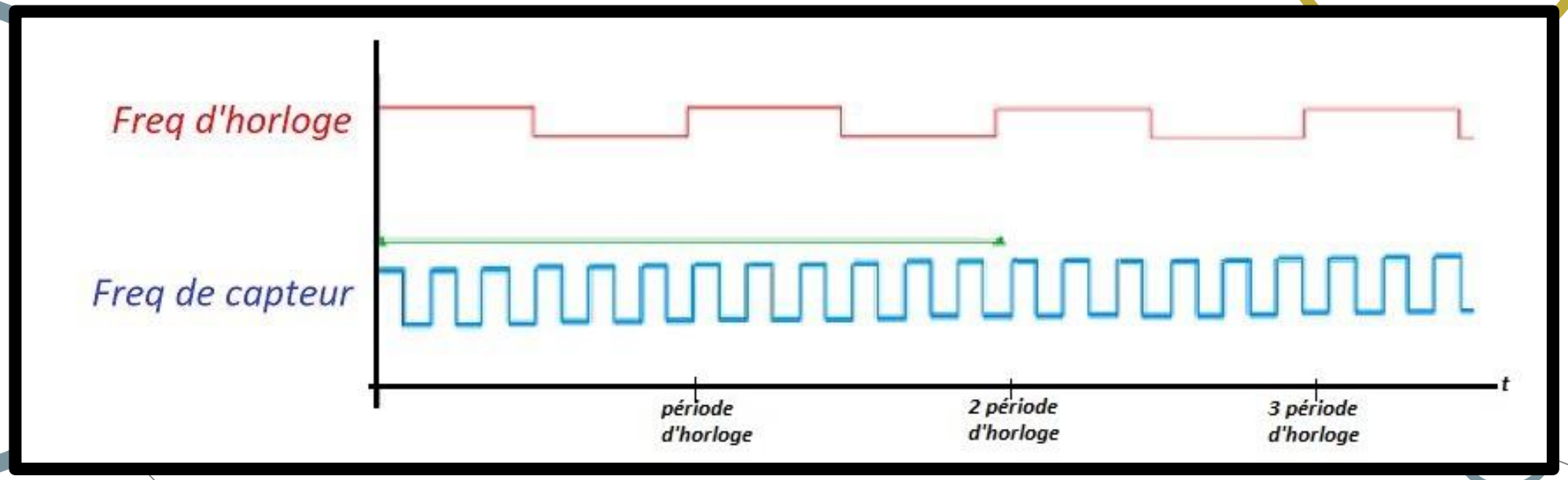


1. La redondance
2. Mécanisme de test robuste

- Détection locale des attaques
- Validation continue de la fréquence
- Vérification du bon fonctionnement

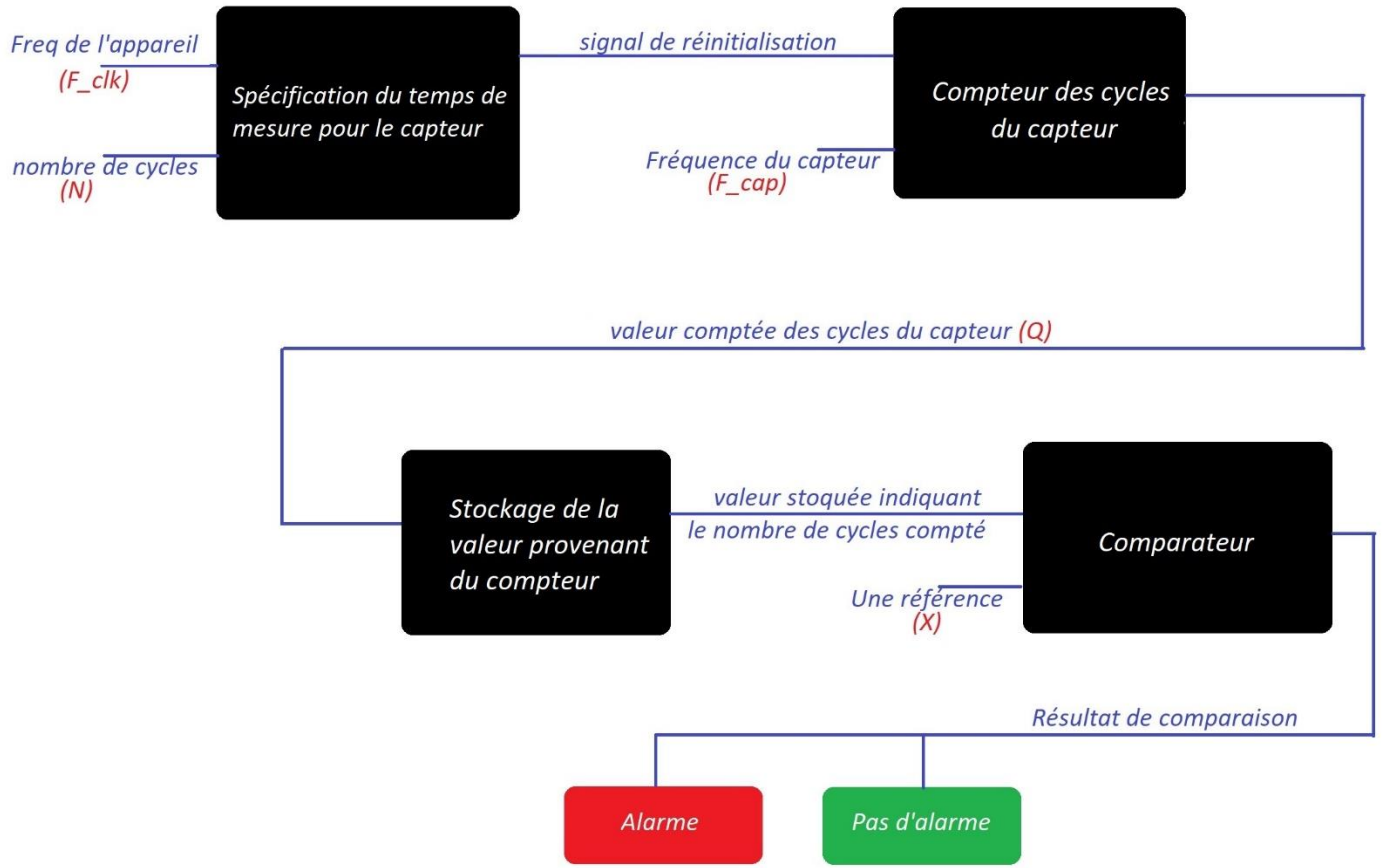
ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Réalisation



Chronogramme de fonctionnement du mécanisme de test

Schéma fonctionnel

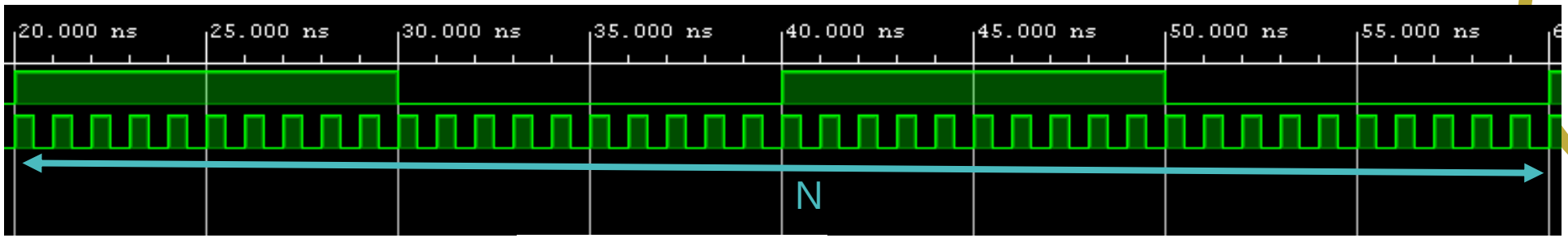


- Compter les cycles du capteur
- Spécifier le temps de mesure
- Récupération de la valeur de compte
- Comparaison

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

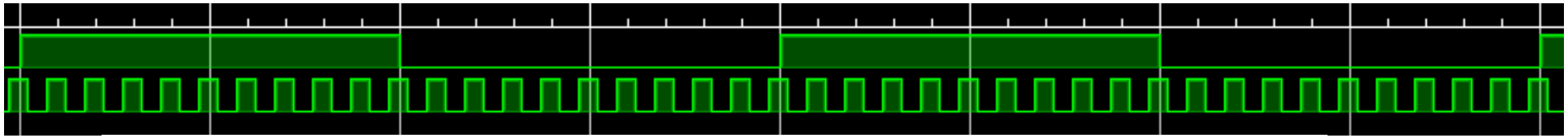
Méthodologie théorique (Choix de N)

Signal d'horloge
Signal du capteur



$$F_{cap} = \frac{Q * F_{clk}}{N}$$

Signal d'horloge
Signal du capteur



$$F_{cap} = \frac{Q * F_{clk}}{N} \quad \text{ou} \quad F_{cap} = \frac{(Q+1) * F_{clk}}{N}$$

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Méthodologie théorique (Choix de N)

L'erreur obtenue est la différence entre la fréquence calculée sur décalage et la vraie fréquence :

$$\Delta f = f_{obtenue\ sur\ décalage} - f_{obtenue\ sur\ normal}$$

$$\Delta f = \frac{(Q+1)*F_{clk}}{N} - \frac{Q*F_{clk}}{N} = \frac{F_{clk}}{N}$$

L'erreur donc est la fréquence de l'horloge divisée par le nombre de cycles choisi

$$N > \frac{F_{clk}}{\Delta f}$$

N peut être déterminée en fonction de la marge d'erreur acceptée

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Méthodologie théorique (Choix de X)

$$F_{cap} = \frac{Q * F_{clk}}{N}$$

Avec un changement de fréquence de k%

$$F_{cap} \pm k\% = \frac{Q_{new} * F_{clk}}{N}$$

Où Q_{new} est la valeur correspondant à la nouvelle fréquence.

$$X_{-k\%} = \left\lceil \frac{N*(F_{cap}-k\%)}{F_{clk}} \right\rceil - 1 \qquad \text{Et} \qquad X_{+k\%} = \left\lceil \frac{N*(F_{cap}+k\%)}{F_{clk}} \right\rceil + 1$$

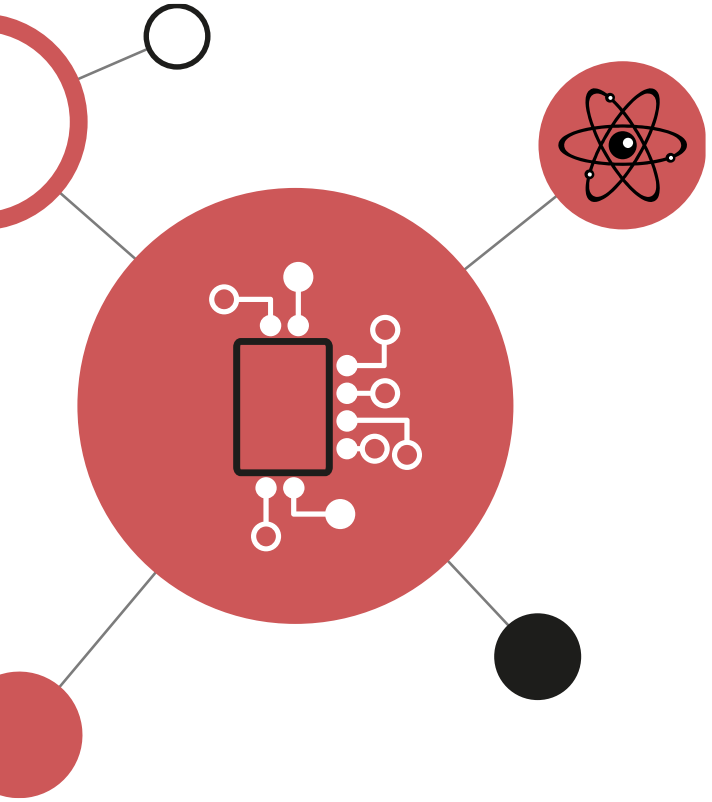
Les extrémités de l'intervalle de référence sont déterminés en fonction du changement de fréquence souhaitant détecter

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Validation expérimentale

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Validation de N pour une erreur de 10 Mhz



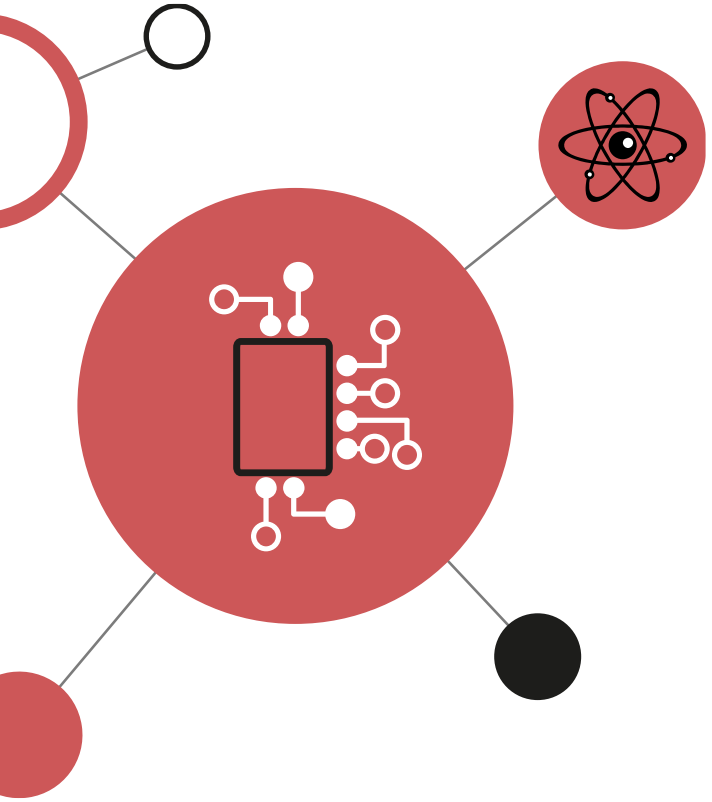
F_{clk} (Mhz)	F_{cap} (Mhz)	N	Fréquence calculée par le mécanisme (Mhz)	Δf (Mhz)	N_{min} calculée théoriquement
50	100	2	125	25	5
		4	112,5	12,5	
		7	107,143	7,143	
		10	105	5	
50	200	2	225	25	5
		4	212,5	12,5	
		7	207,143	7,143	
		10	205	5	
50	250	2	275	25	5
		4	262,5	12,5	
		7	257,143	7,143	
		10	255	5	
50	1000	2	1025	25	5
		4	1012,5	12,5	
		7	1007,143	7,143	
		10	1005	5	

Tant que N ne dépasse pas N théorique, l'erreur ne dépasse pas 10 Mhz

Validation expérimentale

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Validation de X pour une variation de 10%



F_{clk} (Mhz)	F_{cap} (Mhz)	types de mesure	N	Q sans variation	Q avec variation	$X_{k\%}$ mésurée théoriquement
50	250	Avec variation de - 10 %	2	10	10	9
			4	20	19	18
			7	35	33	31,5
			10	50	46	45
50	250	Avec variation de + 10 %	2	10	12	12
			4	20	23	23
			7	35	37	39,5
			10	50	56	56
50	1000	Avec variation de - 10 %	2	40	37	36
			4	80	73,5	72
			7	140	128	126
			10	200	182	180
50	1000	Avec variation de + 10 %	2	40	43	44
			4	80	87	88
			7	140	152	154
			10	200	220	220

$X_{-k\%}$

$X_{+k\%}$

$X_{-k\%}$

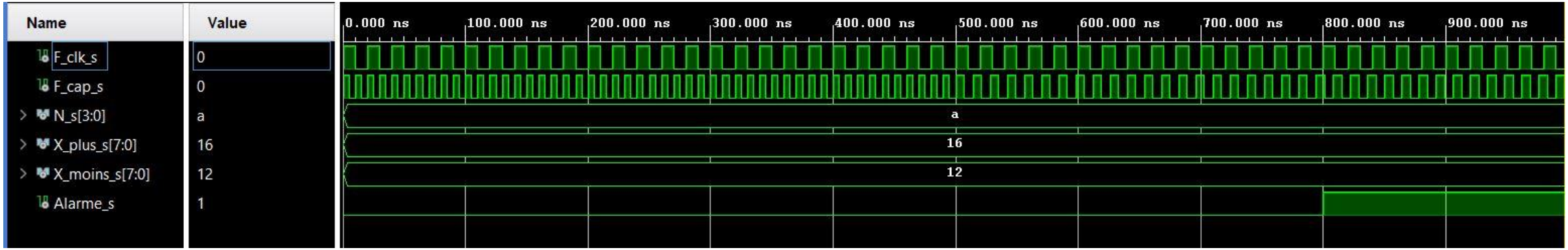
$X_{+k\%}$

Tant que la variation de fréquence ne dépasse pas 10%, la valeur de Q ne sort pas de l'intervalle

Validation expérimentale

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Validation du fonctionnement du système



- F_clk = 50 Mhz
- F_cap = 100 Mhz ➔ 70 Mhz
- N = 10
- $[X_{-k\%}, X_{+k\%}]$ = [18, 22] (pour une variation de 10%)

Conclusion

Un système :

- Capable de surveiller
- Robuste
- Flexible

L'École des Ingénieurs Scientifiques

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

LCIS

LCIS
Laboratoire de Conception et d'Intégration des Systèmes

MERCI

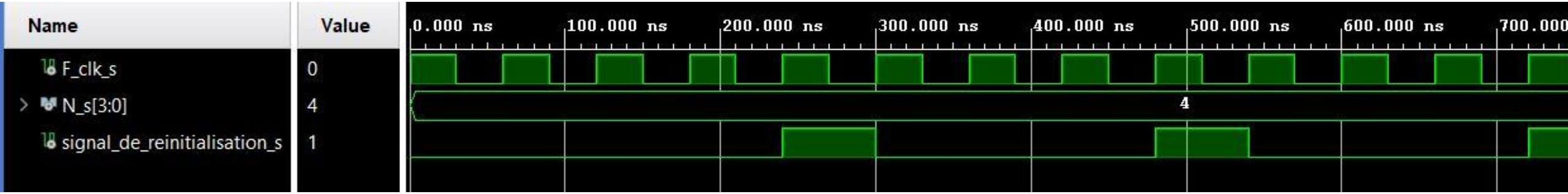
pour votre écoute

L'École des Ingénieurs Scientifiques

Annexe 1.1

```vhdl
-- Importation des bibliothèques nécessaires
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.numeric_std.all;
use IEEE.std_logic_unsigned.all;
-- Déclaration de l'entité et des ports entrées et sorties
entity specification_de_temps_de_mesure is Port (
        F_clk : in STD_LOGIC;                                   -- Horloge principale
        N : in STD_LOGIC_VECTOR (3 downto 0);                   -- Nombre de cycles d'horloge
        signal_de_reinitialisation : out STD_LOGIC := '0');     -- Signal de réinitialisation
end specification_de_temps_de_mesure;
-- Définition du fonctionnement
architecture Behavioral of specification_de_temps_de_mesure is
    signal cnt : STD_LOGIC_VECTOR (3 downto 0) := "0001";       -- Signal pour le compteur
begin
    process(F_clk)
    begin
        if rising_edge(F_clk) then
            if cnt = N then
                signal_de_reinitialisation <= '1';              -- Générer le signal de réinitialisation
                cnt <= "0001";                                  -- Réinitialiser le compteur
            else
                cnt <= cnt + 1;                                 -- Incrémenter le compteur
                signal_de_reinitialisation <= '0';              -- Maintenir le signal de réinitialisation
            end if;
        end if;
    end process;
end Behavioral;
```

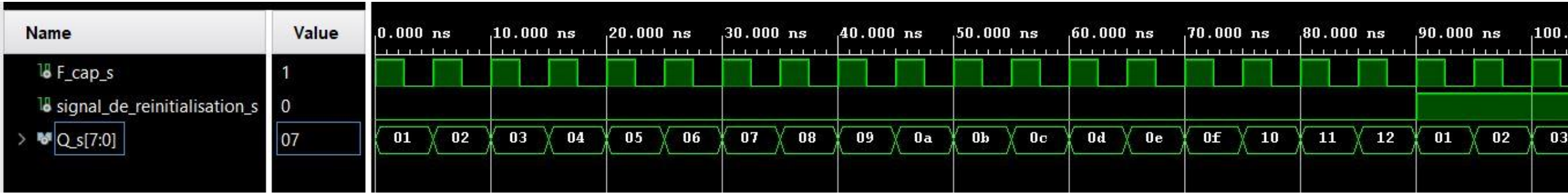
Annexe 1.2


Name
Value
F_clk_s
0
N_s[3:0]
4
signal_de_reinitialisation_s
1
0.000 ns
100.000 ns
200.000 ns
300.000 ns
400.000 ns
500.000 ns
600.000 ns
700.000
4

Annexe 2.1

```vhdl
-- Importation des bibliothèques nécessaires
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.numeric_std.all;
use IEEE.std_logic_unsigned.all;
-- Déclaration de l'entité et des ports entrées et sorties
entity compteur_des_cycles_du_capteur is
    Port ( F_cap : in STD_LOGIC;                                -- Fréquence du capteur
           signal_de_reinitialisation : in STD_LOGIC;           -- Signal de réinitialisation
           Q : out STD_LOGIC_VECTOR (7 downto 0));              -- Valeur comptée des cycles du capteur
end compteur_des_cycles_du_capteur;
-- Définition du fonctionnement
architecture Behavioral of compteur_des_cycles_du_capteur is
    signal cnt : STD_LOGIC_VECTOR (7 downto 0) := "00000001";   -- Signal pour le compteur
    signal signal_de_reinitialisation_dernier : STD_LOGIC := '0';
begin
    process(F_cap, signal_de_reinitialisation)
    begin
        if signal_de_reinitialisation = '1' and signal_de_reinitialisation_dernier ='0' then
            cnt <= "00000001";                                  -- Réinitialiser le compteur
        elsif rising_edge(F_cap) then
            cnt <= cnt + 1;                                     -- Incrémenter le compteur
        end if;
        signal_de_reinitialisation_dernier <= signal_de_reinitialisation ;
    end process;
    Q <= cnt;
end Behavioral;
```

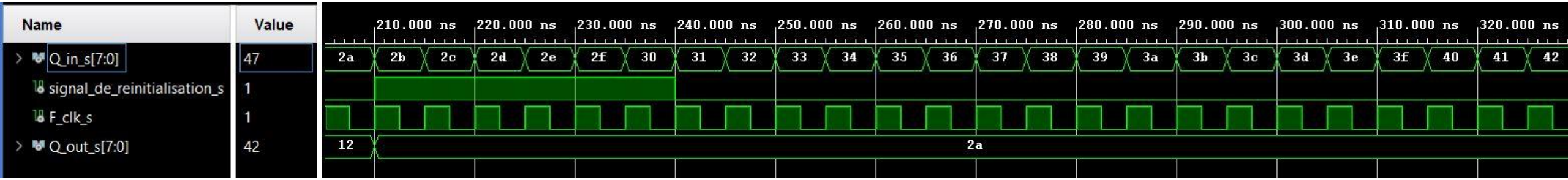
Annexe 2.2


Name
Value
F_cap_s
1
signal_de_reinitialisation_s
0
Q_s[7:0]
07
0.000 ns
10.000 ns
20.000 ns
30.000 ns
40.000 ns
50.000 ns
60.000 ns
70.000 ns
80.000 ns
90.000 ns
01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 01 02 03

Annexe 3.1

```vhdl
-- Importation des bibliothèques nécessaires
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- Déclaration de l'entité et des ports entrées et sorties
entity stockage_de_la_valeur_du_compteur is
    Port ( Q_in : in STD_LOGIC_VECTOR (7 downto 0);                    -- Valeur comptée des cycles du capteur
           F_clk : in STD_LOGIC;                                       -- Fréquence d'horloge
           signal_de_reinitialisation : in STD_LOGIC;                  -- Signal de réinitialisation
           Q_out : out STD_LOGIC_VECTOR (7 downto 0) :="00000000" );   -- Valeur stockée des cycles du capteur
end stockage_de_la_valeur_du_compteur;
-- Définition du fonctionnement
architecture Behavioral of stockage_de_la_valeur_du_compteur is
    signal Q_reg : STD_LOGIC_VECTOR (7 downto 0);
begin
    process(F_clk, signal_de_reinitialisation)
    begin
        if signal_de_reinitialisation = '1' then
            Q_out <= Q_reg ;
        elsif rising_edge(F_clk) then
            Q_reg <= Q_in;
        end if;
    end process;
end Behavioral;
```

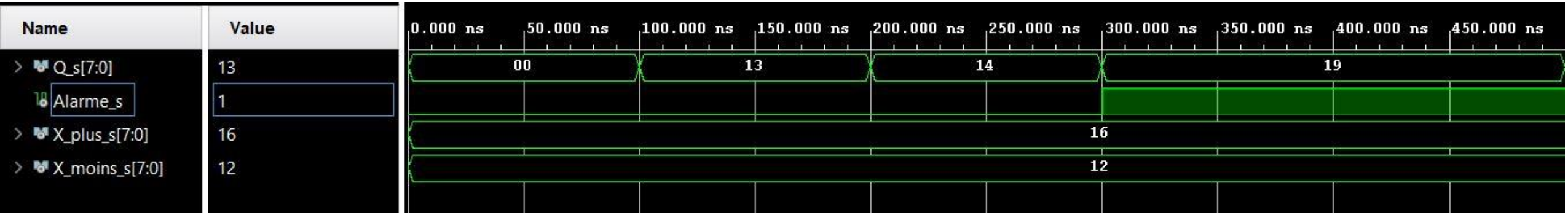
Annexe 3.2


Name
Value
Q_in_s[7:0]
47
signal_de_reinitialisation_s
1
F_clk_s
1
Q_out_s[7:0]
42
210.000 ns
220.000 ns
230.000 ns
240.000 ns
250.000 ns
260.000 ns
270.000 ns
280.000 ns
290.000 ns
300.000 ns
310.000 ns
320.000 ns
2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42
12
2a

Annexe 4.1

```vhdl
-- Importation des bibliothèques nécessaires
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- Déclaration de l'entité et des ports entrées et sorties
entity comparateur is
    Port ( Q : in STD_LOGIC_VECTOR (7 downto 0);                -- Valeur comptée
           Alarme : out STD_LOGIC :='0';                        -- Alarme signifiant que le capteur fonotionne mal
           X_plus : in STD_LOGIC_VECTOR (7 downto 0);           -- L'extrémité supérieure de l'intervalle de référence
           X_moins : in STD_LOGIC_VECTOR (7 downto 0));         -- L'extrémité inférieure de l'intervalle de référence
end comparateur;
-- Définition du fonotionnement
architecture Behavioral of comparateur is
begin
    process(Q)
    begin
        if Q = "00000000" then
            Alarme <= '0' ;
        elsif (Q < X_moins or Q > X_plus) then                  -- Vérification que Q appartient à l'intervalle
            Alarme <= '1';                                      -- Déclenoher l'alarme
        end if;
    end process;
end Behavioral;
```

Annexe 4.2

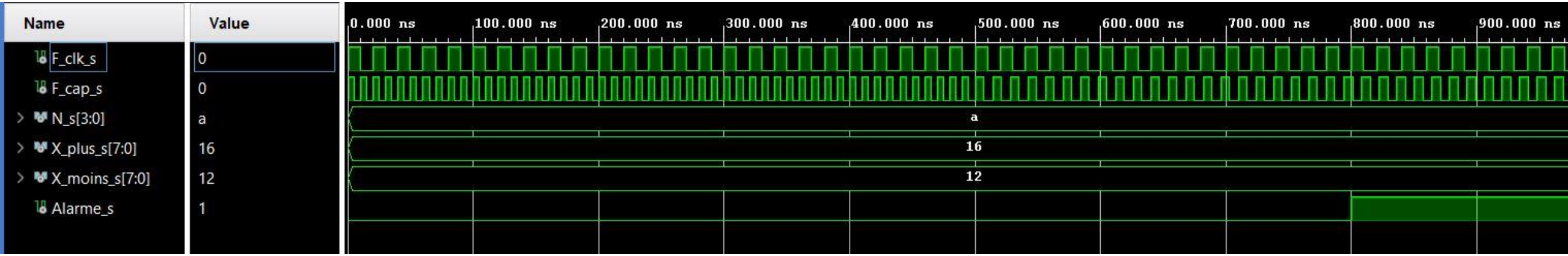

Name
Value
Q_s[7:0]
13
Alarme_s
1
X_plus_s[7:0]
16
X_moins_s[7:0]
12
0.000 ns
50.000 ns
100.000 ns
150.000 ns
200.000 ns
250.000 ns
300.000 ns
350.000 ns
400.000 ns
450.000 ns
00
13
14
19
16
12

Annexe 5.1

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mecanisme_de_test_robuste is Port(
       F_clk : in STD_LOGIC;                          -- Fréquence de l'horloge
       F_cap : in STD_LOGIC;                          -- Fréquence du capteur
       N : in STD_LOGIC_VECTOR (3 downto 0);          -- Nombre de cycles
       X_plus : in STD_LOGIC_VECTOR (7 downto 0);     -- Extrimité supérieure de l'intervalle de référence
       X_moins : in STD_LOGIC_VECTOR (7 downto 0);    -- Extrimité inférieure de l'intervalle de référence
       Alarme : out STD_LOGIC);                       -- Signal d'alarme
end mecanisme_de_test_robuste;

architecture Structural of mecanisme_de_test_robuste is
    signal signal_de_reinitialisation : STD_LOGIC;    -- Signal de réinitialisation
    signal Q_in : STD_LOGIC_VECTOR (7 downto 0);      -- Le nombre des cycles comptées
    signal Q_out : STD_LOGIC_VECTOR (7 downto 0);     -- La valeur stockée des cycles comptées

component specification_de_temps_de_mesure is Port (
        F_clk : in STD_LOGIC;                         -- Horloge principale
        N : in STD_LOGIC_VECTOR (3 downto 0);         -- Nombre de cycles d'horloge
        signal_de_reinitialisation : out STD_LOGIC := '0');  -- Signal de réinitialisation
end component;

component compteur_des_cycles_du_capteur is
    Port ( F_cap : in STD_LOGIC;                      -- Fréquence du capteur
           signal_de_reinitialisation : in STD_LOGIC; -- Signal de réinitialisation
           Q : out STD_LOGIC_VECTOR (7 downto 0));    -- Valeur comptée des cycles du capteur
end component;

component stockage_de_la_valeur_du_compteur is Port(
   Q_in : in STD_LOGIC_VECTOR (7 downto 0);           -- Valeur comptée des cycles du capteur
   F_clk : in STD_LOGIC;                              -- Fréquence d'horloge
   signal_de_reinitialisation : in STD_LOGIC;         -- Signal de réinitialisation
   Q_out : out STD_LOGIC_VECTOR (7 downto 0) :="00000000" );  -- Valeur stockée des cycles du capteur
end component;
```

```vhdl
component comparateur is
    Port ( Q : in STD_LOGIC_VECTOR (7 downto 0);          -- Valeur comptée
           Alarme : out STD_LOGIC :='0';                  -- Alarme signifiant que le capteur fonctionne mal
           X_plus : in STD_LOGIC_VECTOR (7 downto 0);     -- L'extrémité supérieure de l'intervalle de référence
           X_moins : in STD_LOGIC_VECTOR (7 downto 0));   -- L'extrémité inférieure de l'intervalle de référence
end component;

begin

inst_specification_de_temps_de_mesure : specification_de_temps_de_mesure PORT MAP(
    F_clk => F_clk,
    N => N,
    signal_de_reinitialisation => signal_de_reinitialisation);

inst_compteur_des_cycles_du_capteur : compteur_des_cycles_du_capteur PORT MAP(
    F_cap => F_cap,
    signal_de_reinitialisation => signal_de_reinitialisation,
    Q => Q_in);

inst_stockage_de_la_valeur_du_compteur : stockage_de_la_valeur_du_compteur PORT MAP(
    Q_in => Q_in,
    F_clk => F_clk,
    signal_de_reinitialisation => signal_de_reinitialisation,
    Q_out => Q_out);

inst_comparateur : comparateur PORT MAP(
    Q => Q_out,
    X_plus => X_plus,
    X_moins => X_moins,
    Alarme => Alarme);

end Structural;
```

Annexe 5.2


Name
Value
F_clk_s
0
F_cap_s
0
N_s[3:0]
a
X_plus_s[7:0]
16
X_moins_s[7:0]
12
Alarme_s
1
0.000 ns
100.000 ns
200.000 ns
300.000 ns
400.000 ns
500.000 ns
600.000 ns
700.000 ns
800.000 ns
900.000 ns
a
16
12