TD 1 : Exercices simples de synthèse logique

TD – CE312

Exercice 1 :

On considère le programme ci-dessous :

1. **Compléter le chronogramme Fig 1.1.**
2. **Déduire de ce programme, par une construction méthodique, un schéma (bascules et portes logiques).**

```
entity transitm is
  port (hor, e : in  bit;
        s      : out bit);
end transitm;

architecture quasi_struct of transitm is
  signal qa, qb : bit;
begin

  s <= qa xor qb;

  schem : process(hor)
  begin
    if hor'event and hor = '1' then
      qa <= e;
      qb <= qa;
    end if;
  end process schem;
end quasi_struct;
```

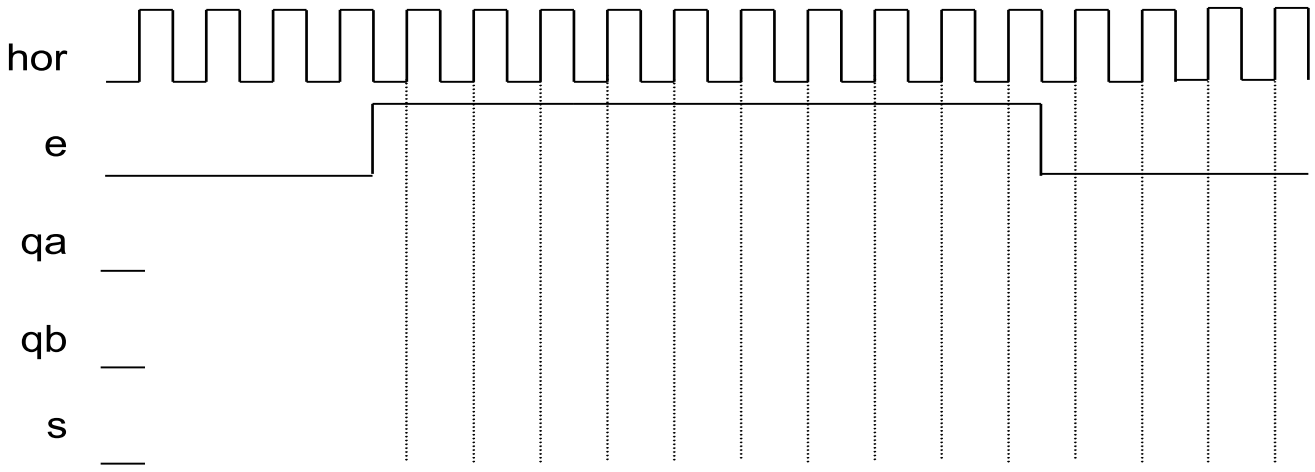

Fig 1.1 : Chronogramme à compléter

Exercice 2 :

On considère le programme ci-dessous :

1. **Ce code est il correct ? Si non, quelle(s) modification(s) apporter ?**
2. **Compléter le chronogramme Fig 1.3.**
3. **Déduire un schéma (bascules et portes logiques) de ce programme ;**

```
entity mise_feu is
  port (
    clk : in  bit;
    ina : in  bit;
    maf : out bit);
end mise_feu;

architecture detect of mise_feu is

begin

  sync : process (clk)
  begin
    if clk'event and clk = '1' then
      maf      <= ina and not maf     ;
    end if;
  end process sync;
end detect;
```

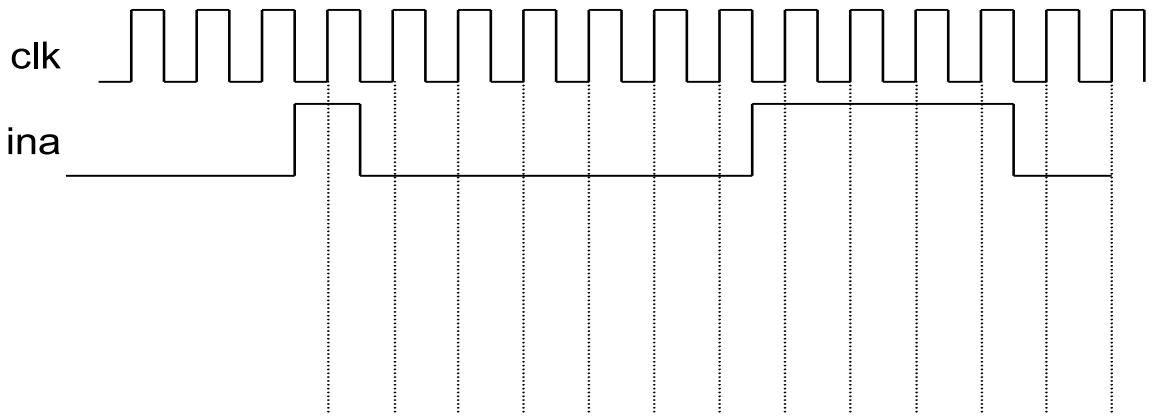

Fig 1.3 : Chronogramme à compléter

Exercice 3 :

On considère la fonction logique de la figure 2.1:

1. **Réaliser un chronogramme de test de cette fonction (compléter le chronogramme fig 2.2).**
2. **Compléter le programme pour réaliser cette fonction ;**

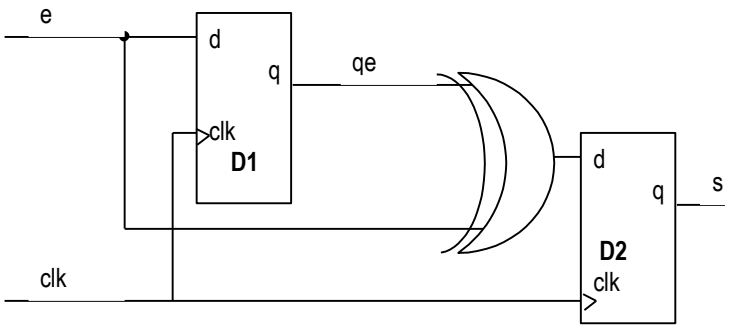

Fig 2.1 Bloc fonctionnel

```
entity detect is
  port (
    clk : in  bit;
    e   : in  bit;
    s   : out bit);
end detect;

architecture reg of detect is

begin
```

```
end reg;
```

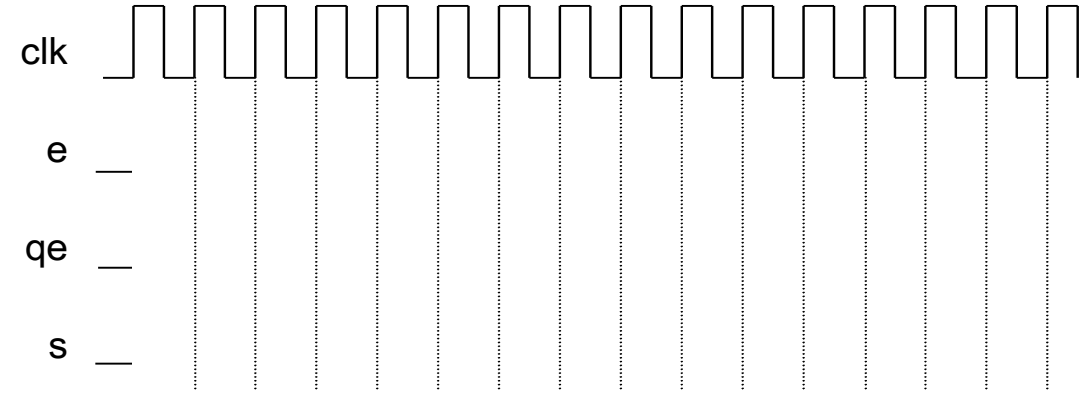

Fig 2.2 : Chronogramme à compléter

Exercice 4 :

On considère l'équation logique ci-dessous :

$$Q_{(t+1)} = /(init \cdot (T \oplus Q_{(t)}))$$

1. **Concevoir le schéma de réalisation de cette équation.**
2. **Concevoir le schéma de réalisation de cette équation avec uniquement des bascule(s) et multiplexeur(s). On pourra s'aider d'une table de vérité. Ce schéma correspond plus précisément à l'implémentation réelle sur un FPGA. Une cellule de FPGA est en effet composée essentiellement d'un réseau de multiplexeurs (LUT : Look-Up Table) suivi d'une bascule.**
3. **Compléter le code VHDL pour réaliser cette fonction. On utilisera une structure de type 'if…elsif'.**
4. **Le cahier des charges évolue : il faut transformer le signal init en une commande « actif bas » (si init='0'), de remise à zéro asynchrone.**
5. **Ajouter une commande de haute impédance « oen » active basse, qui pilote la sortie uniquement lorsque « oen » est actif.**

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.all;

entity basc is
  port (T, clk, init : in  std_logic;
        s            : out std_logic);
end basc;

architecture primitive of basc is
  signal q : std_logic;
begin









end primitive;
```