Unité : CSN & SysLog2

Description en VHDL d'éléments mémoires

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Etienne Messerli

Octobre 2021

Description d'éléments mémoires

- Pas de déclaration explicite en VHDL
- Description **doit indiquer** au synthétiseur que le signal correspond à un élément mémoire **!**

Dès lors:

- utilisez les descriptions standards
- pas de fantaisies !!!!!
- les synthétiseurs refusent les boucles asynchrones, sauf s'ils reconnaissent un élément mémoire (bascule)

Comportement par défaut du VHDL

Lors de l'utilisation d'instructions séquentielles :

Cas non traité $\Rightarrow$ VHDL maintien l'état du signal

- Cette fonctionnalité servira de base pour écrire les éléments mémoires
- Cette fonctionnalité sera aussi un piège par l'obtention de *latchs* non désirés

Avertissement

Avec le langage VHDL :

- Il est possible d'obtenir un élément mémoire lorsque vous ne le voulez pas !

INVERSEMENT

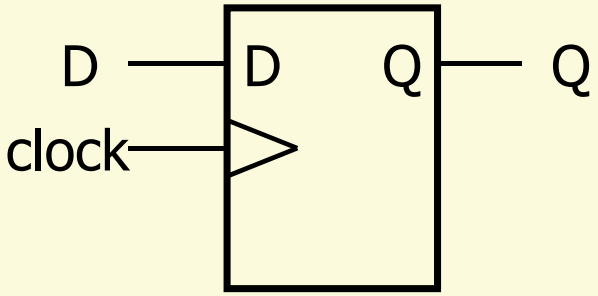
- Il est possible de ne pas avoir d'élément mémoire lorsque vous le voulez !

Description d'un flip-flop

```
Library ieee;
use ieee.std_logic_1164.all;

entity  flip_flop_D is
   port (D       : in  std_logic;
         clock   : in  std_logic;
         Q       : out std_logic);
end flip_flop_D;

architecture comport of flip_flop_D is
begin
  process(clock)
  begin
    if rising_edge(clock) then
      Q <= D;
    end if;
  end process;
end comport;
```

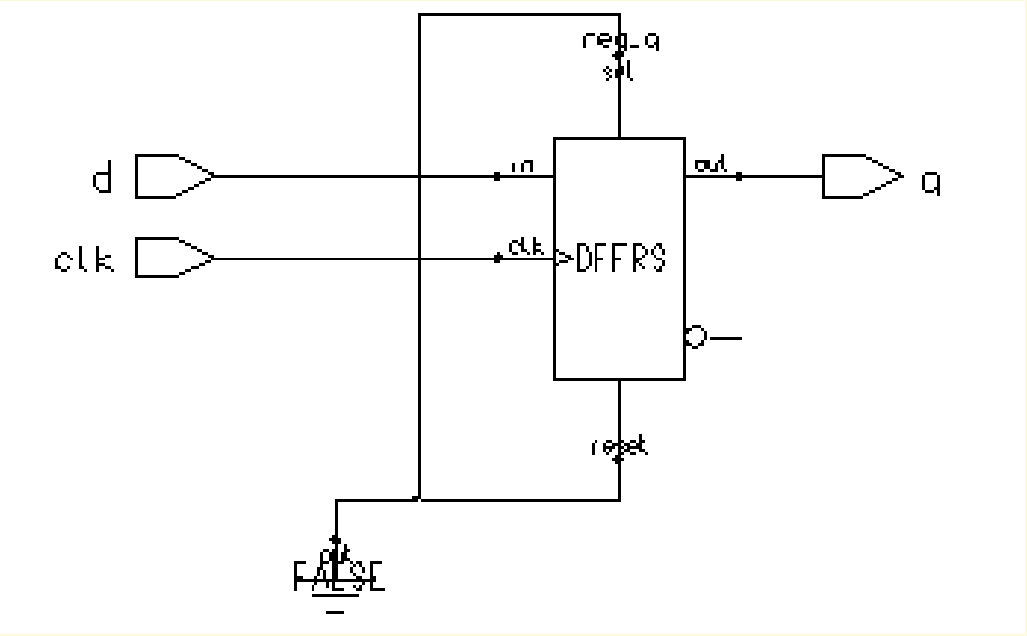


Synthèse du flip-flop

- Cette description se synthétise parfaitement sur tous les synthétiseurs actuels
- Deux informations confirment la notion de flip-flop :
  - le processus ne réagit que sur l'horloge *clock*
  - la condition du test spécifie clairement une action dynamique (rising_edge)

Synthèse DFF avec Leonardo

- Vue RTL

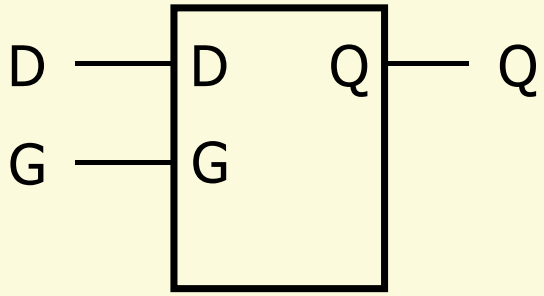


Description d'un latch

```
Library ieee;
use ieee.std_logic_1164.all;

entity  latch is
   port (D      : in  std_logic;
         G      : in  std_logic;
         Q      : out std_logic);
end latch;

architecture comport of latch is
begin
  process(G, D)
  begin
    if G = '1' then
      Q <= D;
    end if;
  end process;
end comport;
```

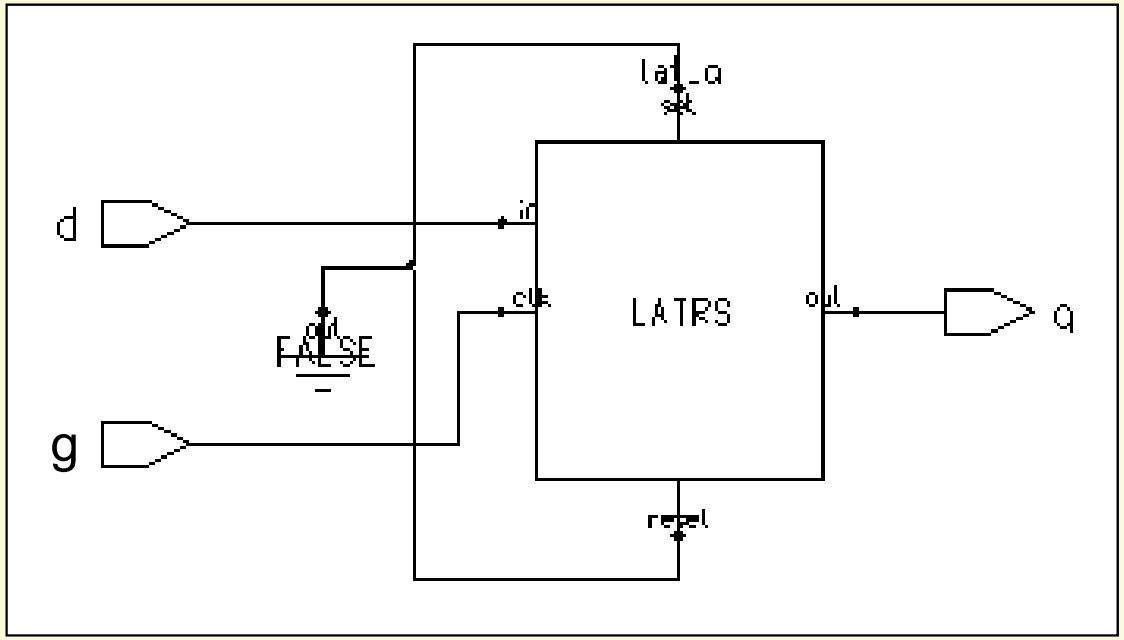


Synthèse du latch

- Cette description se synthétise parfaitement sur tous les synthétiseurs actuels
- Deux informations confirment la notion de latch :
  - le processus réagit sur les deux signaux G et D
  - la condition du test spécifie clairement une action sur un niveau, soit "="

Synthèse latch avec Leonardo

- Vue RTL

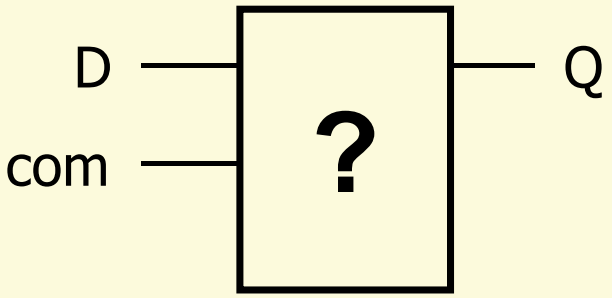


Type de bascule ?

```
Library ieee;
use ieee.std_logic_1164.all;

entity  bascule is
   port (D   : in  std_logic;
         com : in  std_logic;
         Q   : out Std_logic);
end bascule;

architecture comport of bascule is
begin
  process(com)
  begin
    if com = '1' then
      Q <= D;
    end if;
  end process;
end comport;
```



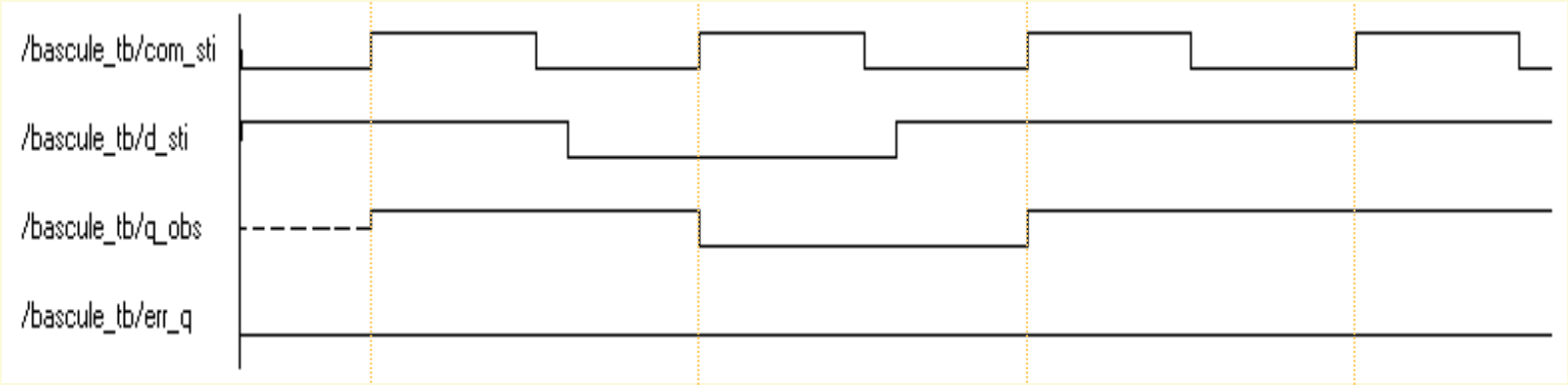
Synthèse de cette bascule

- Cette description est **ambiguë**.
- Deux informations se contredisent :
  - le processus se déclenche uniquement sur *com*
    => action dynamique => flip-flop
  - la condition du test spécifie un niveau (=)
    => action sur un niveau => latch

Type de bascule ?

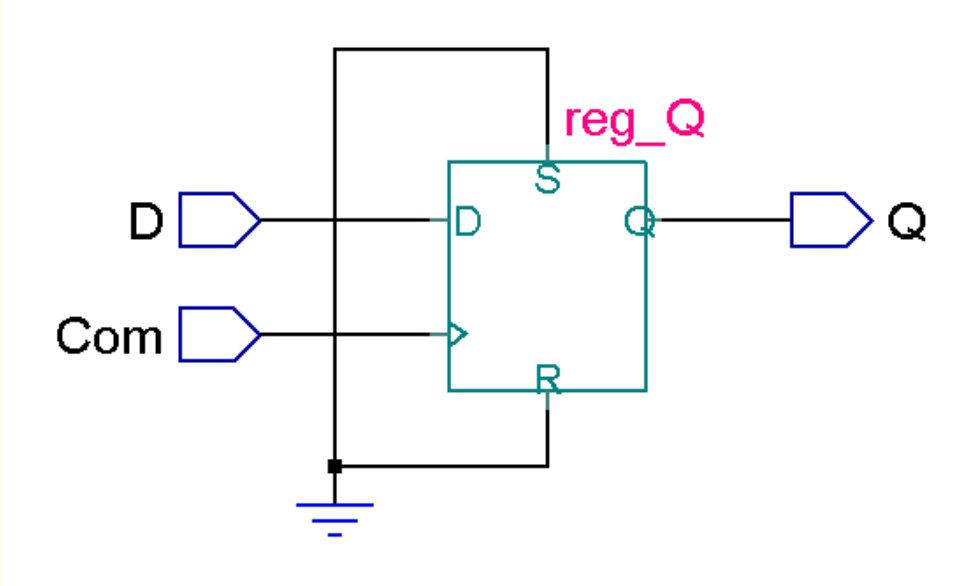
- Compréhension par le simulateur, interprétation stricte de la description VHDL:
  - une seule réponse : .....
- Compréhension par le synthétiseur qui va traduire la description VHDL en logique:
  - priorité à la liste de sensibilité : .....
  - priorité à la condition de test : .....

Synthèse avec Synplify : 3 WARNING



Le comportement correspond a celui d'un flip-flop D

Synthèse avec Léonardo

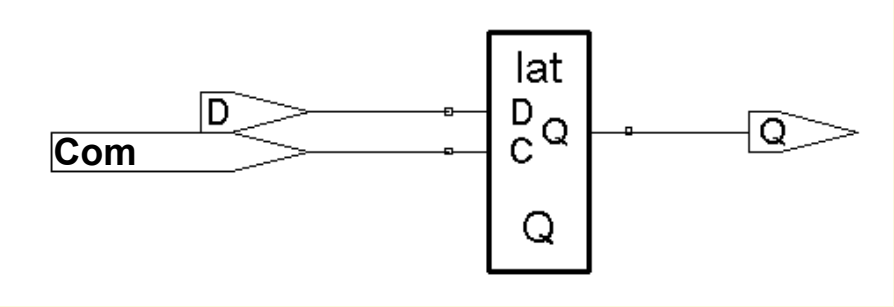


Le résultat de la synthèse est un flip-flop

Synthèse avec Synplify : 3 WARNING

```
Synthesizing work.bascule.comport
@W:"f:\en_cours\synplify\bascule.vhd":30:2:30:8
  Incomplete sensitivity list - assuming
  completeness
@W:"f:\en_cours\synplify\bascule.vhd":33:10:33:10
  Referenced variable d is not in sensitivity list
Post processing for work.bascule.comport
@W:"f:\en_cours\synplify\bascule.vhd":32:4:32:5
  Latch generated from process for signal q,
  probably caused by a missing assignment in an if
  or case stmt
@END
```

Synthèse avec Synplify



Le résultat de la synthèse est un LATCH

Incohérent avec la simulation !!!

Conclusion sur cette description

- Le comportement en simulation peut-être différent du matériel obtenu
- Matériel obtenu dépend du synthétiseur, résultat pas identique

Cette description est **"NON SYNTHETISABLE"**

Interdit par la norme IEEE 1076.6-1999

Etat initial d'une bascule

- A l'instant 0 ns :
  Tous les signaux sont à l'état 'U'
- Une initialisation est indispensable
- Comment forcer l'état initial d'une bascule en VHDL ?

Proposition I

```
Library ieee;
use ieee.std_logic_1164.all;

entity flip_flop_d is
   port (D     : in  std_logic;
         clock : in  std_logic;
         Q     : out std_logic := '0' );
end flip_flop_d;

architecture comport of flip_flop_d is
begin
  process(clock)
  begin
    if rising_edge(clock) then
      Q <= D;
    end if;
  end process;
end comport;
```

Proposition II

```
Library ieee;
use ieee.std_logic_1164.all;

entity flip_flop_d is
   port (D      : in  std_logic;
         clock  : in  std_logic;
         reset  : in  std_logic;
         Q      : out std_logic);
end flip_flop_d;

architecture comport of flip_flop_d is
begin
  process(clock, reset)
  begin
    if reset = '1' then
      Q <= '0';
    elsif rising_edge(clock) then
      Q <= D;
    end if;
  end process;
end comport;
```

Conclusion initialisation bascule

- Proposition I
  - fonctionne en simulation VHDL
  - ne sera pas synthétisé : "soft"
- Proposition II
  - fonctionne en simulation VHDL
  - sera correctement synthétisé : "hard"
  - fonctionne en simulation après synthèse

Flip-flop avec actions asynchrones

La norme IEEE-1076.6 "Standard for VHDL Register Transfer Level (RTL) Synthesis" définit le sous-ensemble utilisable en synthèse.

- Voici la syntaxe de l'instruction *process* pour un flip-flop avec une action asynchrone

```
process(clock, action_asynchrone)
begin
  if action_asynchrone = '1' then
    Q <= ETAT_INITIAL;
  elsif rising_edge(clock) then
    Q <= D;
  end if;
end process;
```

Flip-flop D avec reset asynchrone

```
Library ieee;
use ieee.std_logic_1164.all;

entity flip_flop_d is
   port (D      : in  std_logic;
         clock  : in  std_logic;
         reset  : in  std_logic;
         Q      : out std_logic);
end flip_flop_d;

architecture comport of flip_flop_d is
begin
  process(clock, reset)
  begin
    if reset = '1' then
      Q <= '0';
    elsif rising_edge(clock) then
      Q <= D;
    end if;
  end process;
end comport;
```

Description VHDL d'éléments mémoires

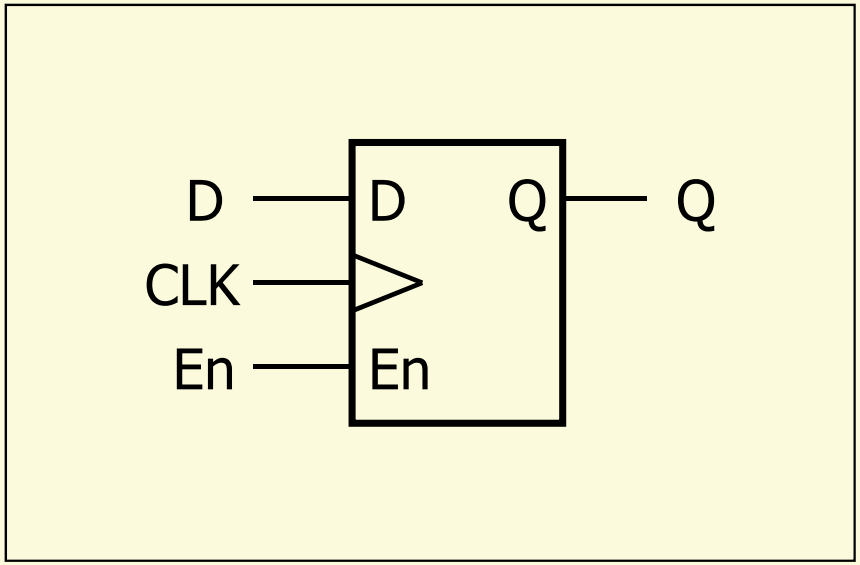
- Voir présentation séparée

A traiter si pas fait exercices

- Si les exercices sur les bascules ne sont pas traités, il est important de voir le cas de la bascule DFFE.
- La solution intuitive (en pensant soft) n'est pas synthétisable

Description DFFE

- Description d'un flip-flop D avec maintien (Enable)
- Symbole DFFE :



Description DFFE, proposition I

```
Library ieee;
use ieee.std_logic_1164.all;

entity  dff_I is
   port (clock   : in std_logic;
         D, en   : in std_logic;
         Q       : out std_logic);
end dff_I;

architecture comport of dff_I  is
begin
  process(clock)
  begin
    if rising_edge(clock) and en = '1' then
      Q <= D;
    end if;
  end process;
end comport;
```

Description DFFE, proposition II

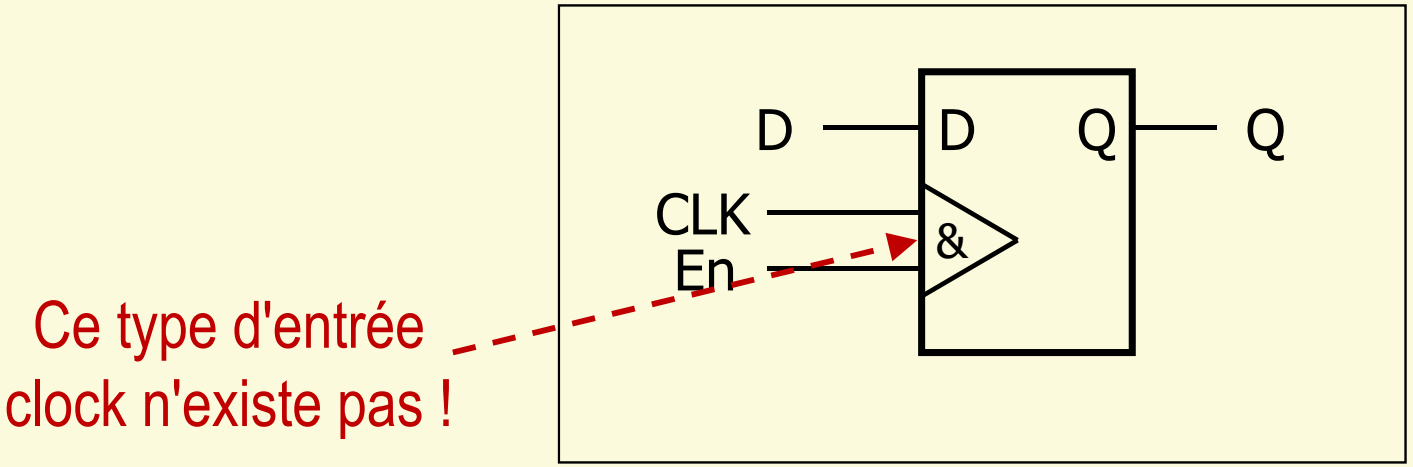
```
Library ieee;
use ieee.std_logic_1164.all;

entity  dff_II is
   port (clock   : in std_logic;
         D, en   : in std_logic;
         Q       : out std_logic);
end dff_II;

architecture comport of dff_II is
begin
  process(clock)
  begin
    if rising_edge(clock) then
      if en = '1' then
        Q <= D;
      end if;
    end if;
  end process;
end comport;
```

Description DFFE, proposition I

- Représentation matériel directe:

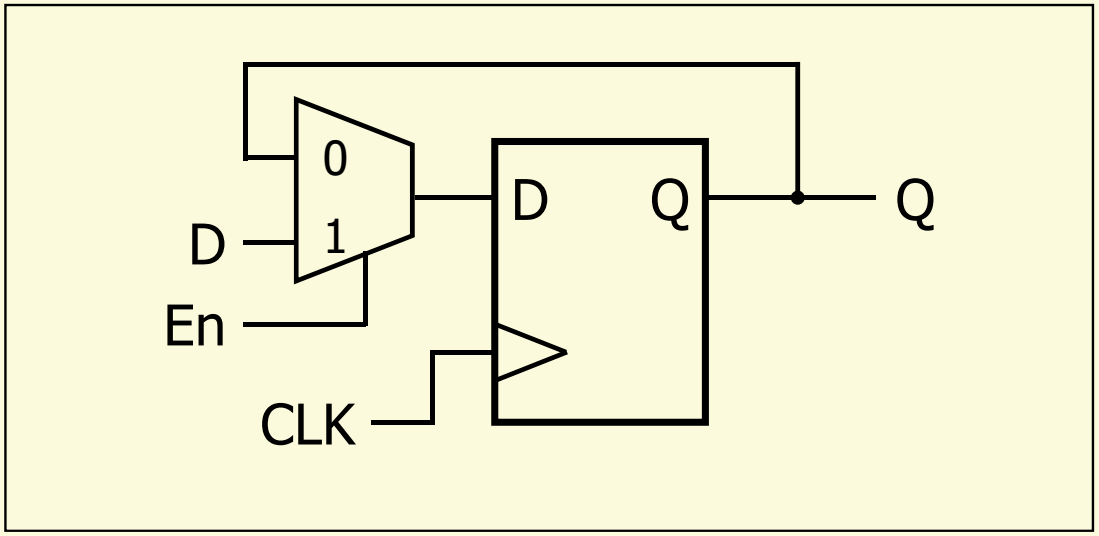


La norme IEEE 1076.6 -2004 spécifie qu'un signal testé avec un clock a une action synchrone

Je ne recommande pas cette synthaxe

Description DFFE, proposition II

- Représentation matériel :



Les composants utilisés existent

Cette description est lisible et synthétisable

Questions ?