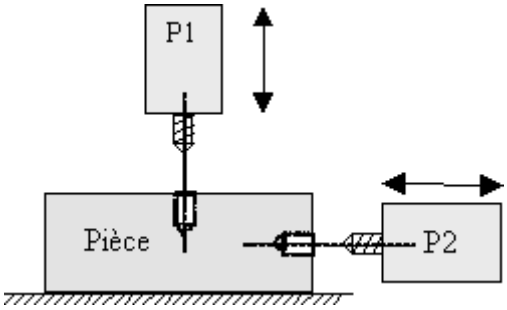
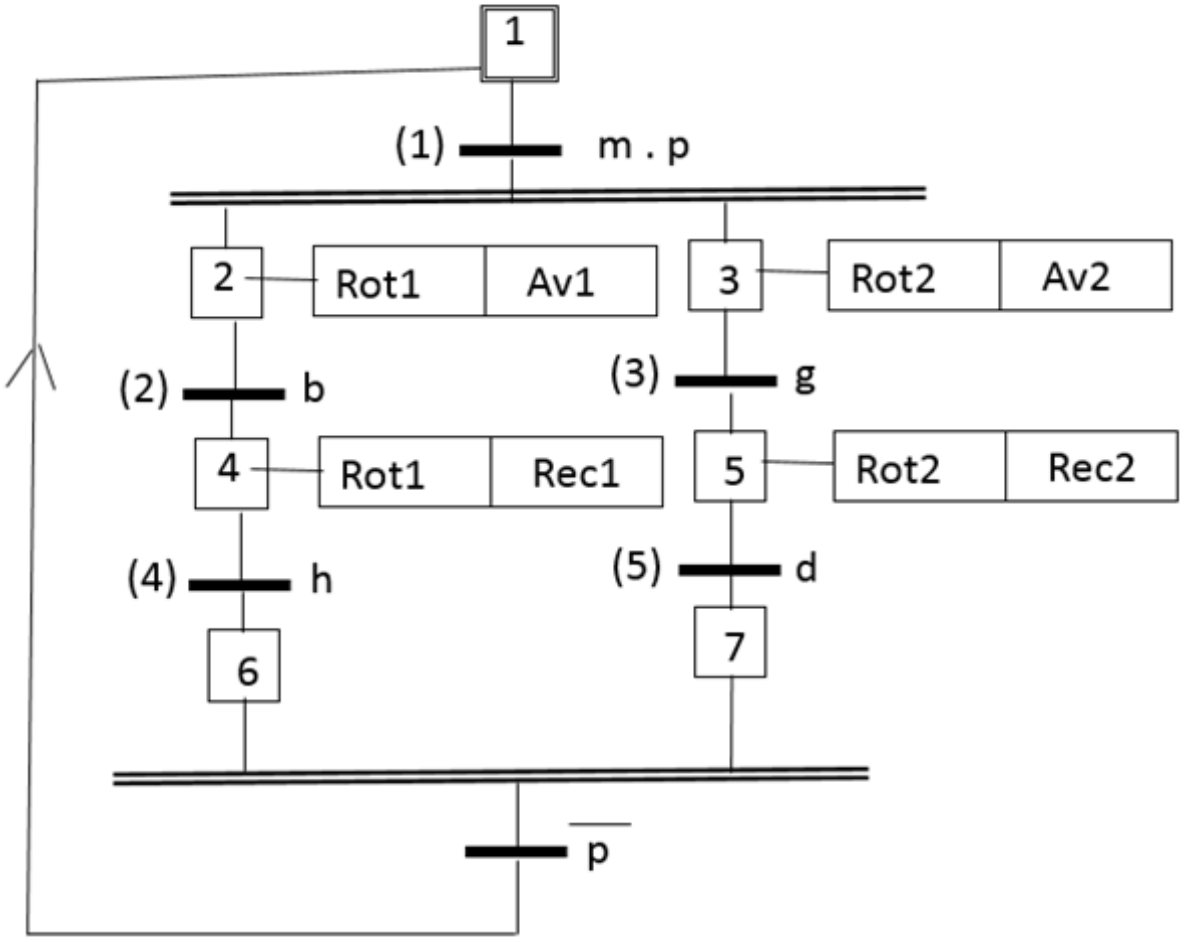
# TP2

# Implémentation des GRAFCETs en utilisant le langage VHDL

**CHIRAZ TRABELSI**
**Et**
**ALEXANDRE BRIERE**


Soit le contrôleur d'un système de perçage à deux perceuses vu en TD.




1. Donner les équations des bascules qui implémentent ce contrôleur en utilisant le codage One-hot. Chaque équation contient une partie transition et une partie maintien tel que le montre l'équation de D1 pour la bascule qui implémente l'étape 1 :

   D1= (Q6.Q7.$\overline{p}$) + (Q1. ($\overline{m.p}$))

2. Pour implémenter le GRAFCET, on va créer un fichier VHDL. Pour cela, aller dans « File → New → VHDL File »
   Copier le code suivant dans le fichier :

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

LIBRARY work;

ENTITY GRAFCET_2PERCEUSES IS
	PORT
	(
		m :  IN  STD_LOGIC;
		p :  IN  STD_LOGIC;
		h :  IN  STD_LOGIC;
		b :  IN  STD_LOGIC;
		d :  IN  STD_LOGIC;
		g :  IN  STD_LOGIC;
		Clock :  IN  STD_LOGIC;
		Reset :  IN  STD_LOGIC;
		Rot1:OUT STD_LOGIC;
		Av1:OUT STD_LOGIC;
		Rec1:OUT STD_LOGIC;
		Rot2:OUT STD_LOGIC;
		Av2:OUT STD_LOGIC;
		Rec2:OUT STD_LOGIC
	);
END GRAFCET_2PERCEUSES;
```

L'entité (entity) VHDL à créer s'appelle GRAFCET_2PERCEUSES. Cette entité a comme entrée : m, p, h, b, d, g, Clock et Reset qui sont toutes des booléens (STD_LOGIC). Les sorties sont Rot1, Av1, Rec1, Rot2, Av2 et Rec2.

Toujours dans le même fichier copier ce code à la suite du premier :

```
ARCHITECTURE bdf_type OF GRAFCET_2PERCEUSES IS

SIGNAL	Q1 :  STD_LOGIC:='0';
SIGNAL	Q2 :  STD_LOGIC:='0';
SIGNAL	Q3 :  STD_LOGIC:='0';
SIGNAL	Q4 :  STD_LOGIC:='0';
SIGNAL	Q5 :  STD_LOGIC:='0';
SIGNAL	Q6 :  STD_LOGIC:='0';
SIGNAL	Q7 :  STD_LOGIC:='0';

SIGNAL	D1 :  STD_LOGIC:='0';
SIGNAL	D2 :  STD_LOGIC:='0';
SIGNAL	D3 :  STD_LOGIC:='0';
SIGNAL	D4 :  STD_LOGIC:='0';
SIGNAL	D5 :  STD_LOGIC:='0';
SIGNAL	D6 :  STD_LOGIC:='0';
SIGNAL	D7 :  STD_LOGIC:='0';
```

Ce code permet de déclarer les signaux internes qu'on va utiliser. Ces signaux seront traduits automatiquement en bascules par l'outil dans la phase de compilation. Les signaux de Q1 à Q7 représentent les étapes (sorties des bascules) et les signaux D1 à D7 les entrées des bascules.

Si vous avez remarquez, l'entité contient une entrée Reset. Cette entrée permet de donner la ou les étapes initiales. Sur le front montant de l'horloge, les étapes se mettent à jour (les sorties Q des bascules prennent les valeurs des entrées D). Pour implémenter l'initialisation et la mise à jour des étapes, copiez le code suivant :

```
BEGIN

PROCESS(Reset,Clock)
BEGIN

IF (Reset='1') THEN
        Q1<='1';
        Q2<='0';
        Q3<='0';
        Q4<='0';
        Q5<='0';
        Q6<='0';
        Q7<='0';
ELSE
IF (RISING_EDGE(Clock)) THEN
        Q1<=D1;
        Q2<=D2;
        Q3<=D3;
        Q4<=D4;
        Q5<=D5;
        Q6<=D6;
        Q7<=D7;
END IF;
END IF;
END PROCESS;
```

Ce code montre que seule la sortie Q1 sera active à l'initialisation. Sur front montant (RISING_EDGE) de l'horloge, les sorties Q des bascules prennent les valeurs des entrées D.

Il reste de spécifier les équations des entrées D des bascules et celles des sorties. Pour ceci, copier ce code et compléter les équations manquantes :

```
D1<=(Q6 and Q7 and (not(p))) or (Q1 and (not(m and p)));
D2<=
D3<=
D4<=
```

```
D5<=
D6<= (Q4 and h) or (Q6 and (not(Q7) or p));
D7<=

Rot1 <= Q2 or Q4;
Av1<=
Rec1<=
Rot2 <=
Av2<=
Rec2<=

END bdf_type;
```

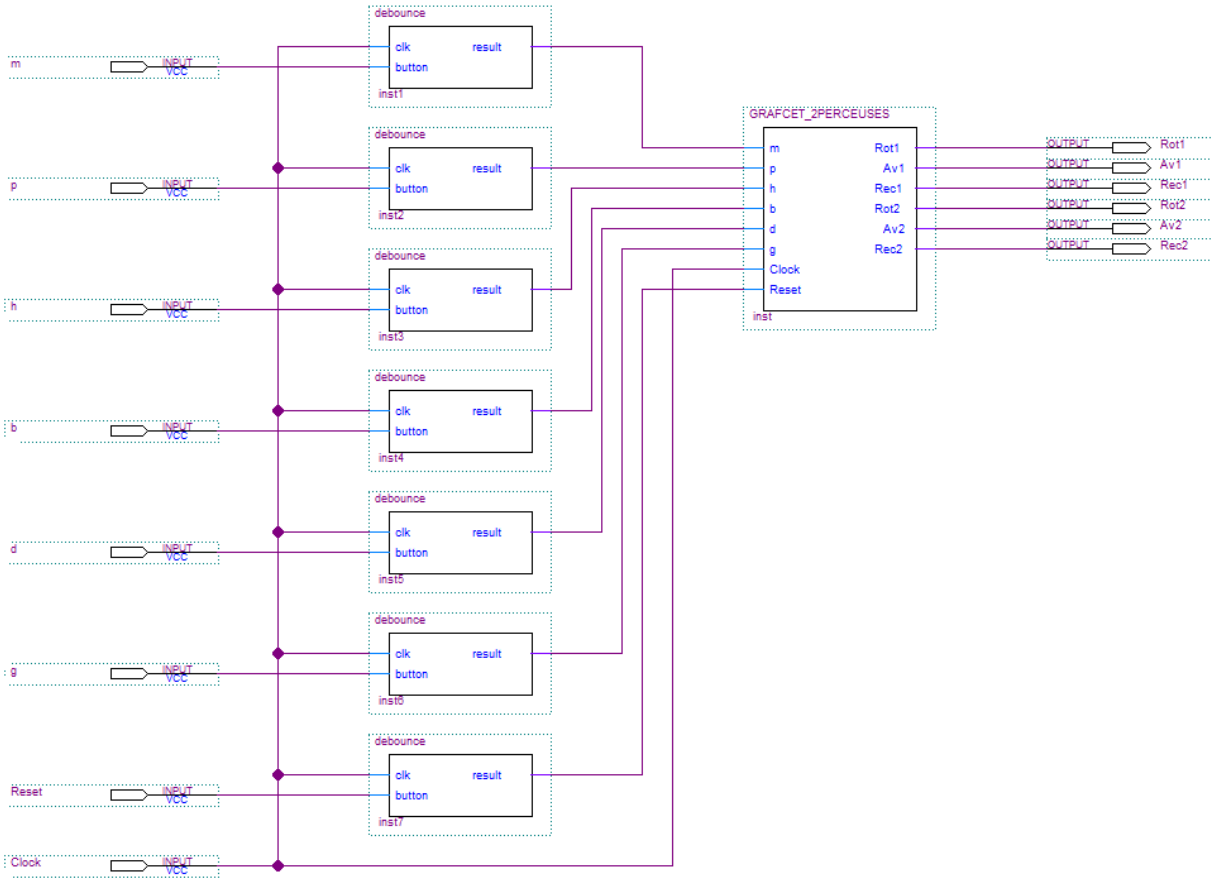
Sauvegarder le fichier VHDL avec le nom GRAFCET_2PERCEUSES.vhd. En effet, **il faut avoir le même nom que l'entité déclarée dans le fichier**, donc si vous voulez sauvegarder le fichier avec le nom TP2.vhd par exemple, il faut remplacer le terme GRAFCET_2PERCEUSES par TP2 (il est utilisé à trois endroits) dans fichier VHDL.

Désactiver l'optimisation de placement et compiler le projet.

**Appeler l'encadrant pour valider les équations.**

Générer un symbole à partir de ce fichier et ajouter le symbole dans un nouveau schéma.

Nous allons utiliser des switches pour simuler les entrées du Grafcet. Les switches disponibles sur la carte font des rebonds (à chaque fois qu'on change de niveau, il y a des oscillations et donc on risque d'interpréter un changement de niveau comme plusieurs changements successifs, vous les avez peut-être remarqués dans le TP précédent). Pour cette raison, on va utiliser des modules « debounce » qui vont supprimer les rebonds (oscillations). Pour ce faire, ajouter le fichier « debounce.vhd » que vous trouvez dans Moodle dans votre projet. Ouvrez-le dans le projet et générez un symbole. Dans le fichier schematic qui contient le symbole du Grafcet que vous venez de créer ajouter des instances du module « debounce » pour obtenir le schéma suivant :




Affecter les entrées/sorties aux broches suivantes en utilisant Assignment → Pin Planner. Suivez le tableau suivant et déterminer les codes des PINs à partir du manuel de la carte. Pour le signal Clock, entrez directement le code PIN_AF14 (c'est un emplacement dédié une horloge de 50 MHz sur la carte).

| Entrée/sortie | Pin |
|---|---|
| m | SW[4] |
| p | SW[3] |
| h | SW[0] |
| b | SW[1] |
| d | SW[8] |
| g | SW[9] |
| Clock | PIN_AF14 |
| Reset | SW[6] |
| Rot1 | LEDR[0] |
| Av1 | LEDR[1] |
| Rec1 | LEDR[2] |
| Rot2 | LEDR[7] |
| Av2 | LEDR[8] |
| Rec2 | LEDR[9] |

Compiler le projet et tester sur la carte. Pour l'initialisation, mettre le switch 6 (Reset) à 1 pour quelques temps, puis à 0 pour le reste de l'exécution. Initialement les

switches correspondant à h et d sont à 1. Pour placer une pièce, mettre le switch correspondant à 1. Pour simuler le bouton m, mettre le switch correspondant à 1 pour quelque temps puis à 0.

**Préparer un scénario de simulation qui permet de suivre le fonctionnement décrit dans le GRAFCET. Tester ce scénario et appeler l'encadrant pour validation.**