Examen – Juin 2009

Durée approximative : 45 minutes. – Les documents sont autorisés.

 *Cette partie est à rédiger sur une copie séparée.* 

Exercice 1 Modélisation d'une imprimante

1. Modéliser avec un automate temporisé une imprimante ayant les caractéristiques suivantes :

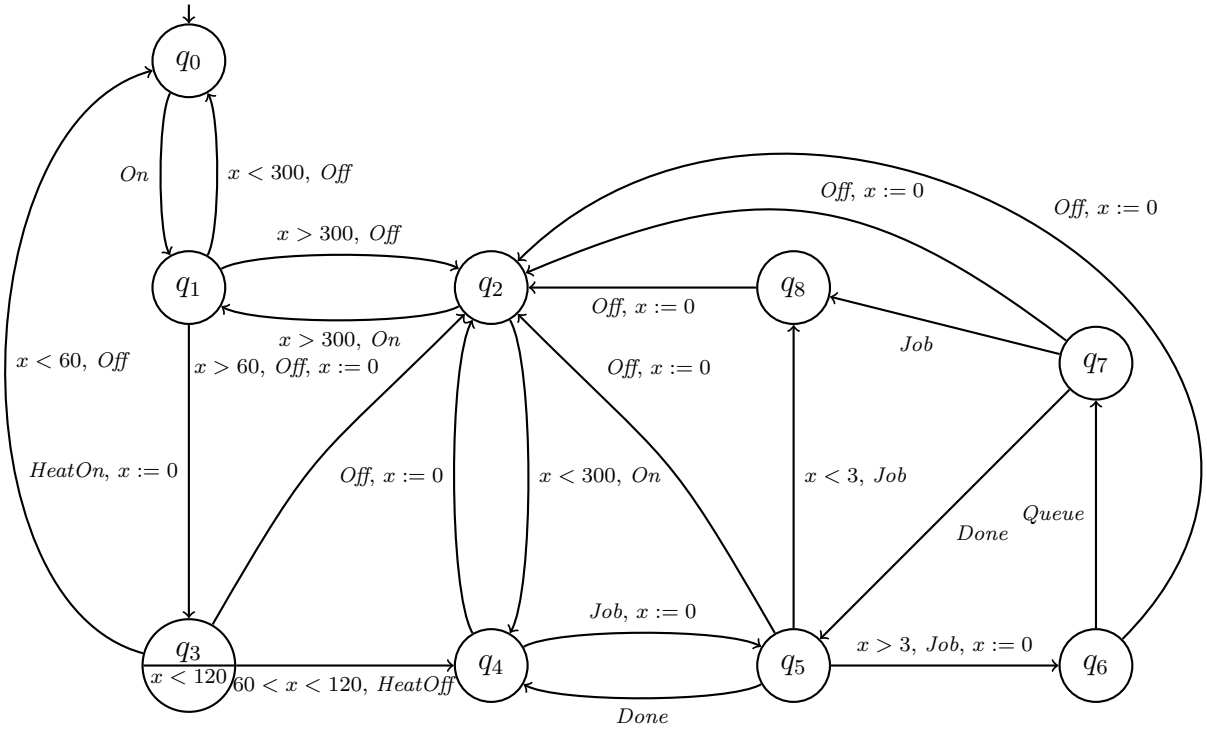
   – L'imprimante est initialement éteinte.
   – Lorsque l'imprimante est mise sous tension et qu'elle a été éteinte froide ou éteinte depuis plus de 5 minutes, elle doit chauffer pendant 1 à 2 minutes.
   – Une fois chaude, elle peut recevoir des tâches d'impression.
   – Lorsqu'une tâche est reçue, l'imprimante la traite ; elle émet un signal lorsque l'impression est terminée.
   – Si une seconde tâche arrive alors elle est mise en attente.
   – Si deux tâches d'impression sont reçues en moins de 3 secondes d'intervalle, l'imprimante va dans un état d'erreur.
   – L'imprimante ne peut mettre qu'une seule tâche en attente ; si une troisième tâche arrive, l'imprimante va dans un état d'erreur.
   – On ne peut quitter l'état d'erreur que en éteignant et rallumant l'imprimante.
   – On peut éteindre l'imprimante à chaque instant. On perd alors les tâches en cours.

   L'automate pourra s'exécuter sur l'alphabet suivant : {*On* (allumage de l'imprimante), *Off* (extinction de l'imprimante), *HeatOn* (début du chauffage), *HeatOff* (fin du chauffage), *Job* (arrivée d'une tâche), *Queue* (mise en attente d'une tâche), *Done* (fin de l'impression d'une tâche)}.

2. Traduire l'automate précédent dans la syntaxe de HyTech.

Solution de l'Exercice 1

1. On est obligés de considérer à part le cas initial où l'imprimante n'a jamais été allumée, et donc n'est pas chaude, même si l'on est dans cet état depuis moins de 5 minutes. On doit ensuite gérer les cas selon si le chauffage a été effectué avant que l'imprimante ne soit éteinte.



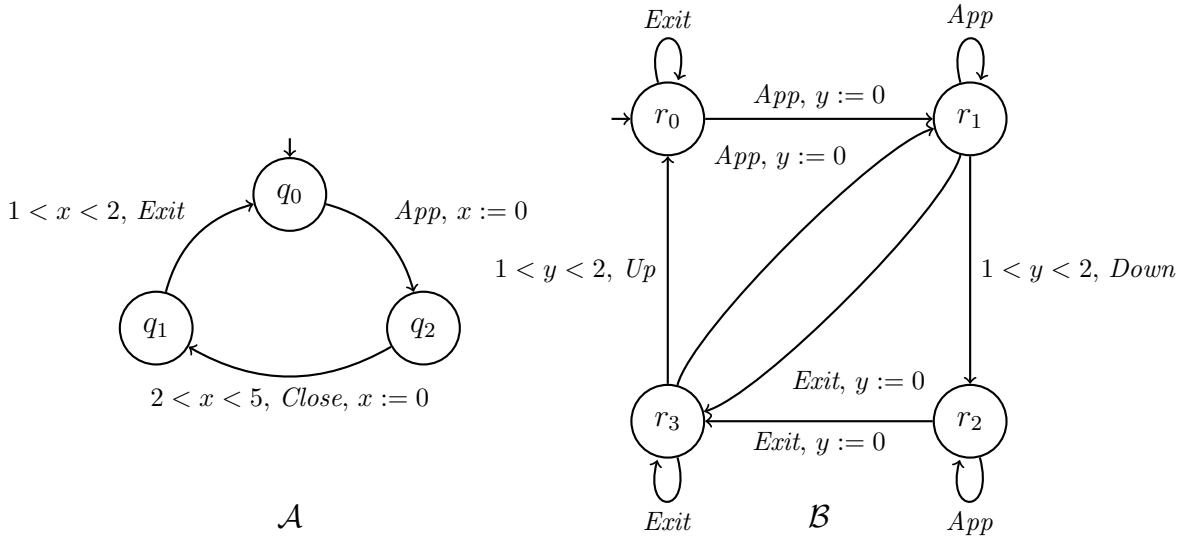
2.

```
-- Déclaration des horloges
var x:clock;

-- Déclaration de l'automate
automaton printer
synclabs: On,Off,HeatOn,HeatOff,Job,Queue,Done;

initially init_off & x=0;

-- État q_0
loc init_off: while True wait {}
    when True sync On goto cold;

-- État q_1
loc cold: while True wait {}
    when x<300 sync Off goto init_off;
    when x>300 sync Off goto off;
    when True sync HeatOn do {x'=0} goto heating;

-- État q_2
loc off: while True wait {}
    when x<300 sync On goto idle;
    when x>300 sync On goto heating;

-- État q_3
loc heating: while x<120 wait {}
    when x<60 sync Off goto init_off;
    when x>60 sync Off do {x'=0} goto off;
```

```
    when x>60 & x<120 sync HeatOff goto idle;

-- État q_4
loc idle: while True wait {}
    when True sync Off do {x'=0} goto off;
    when True sync Job do {x'=0} goto printing;

-- État q_5
loc printing: while True wait {}
    when True sync Off do {x'=0} goto off;
    when x<3 sync Job goto error;
    when x>3 sync Job do {x'=0} goto do_queue;
    when True sync Done goto idle;

-- État q_6
loc do_queue: while True wait {}
    when True sync Off do {x'=0} goto off;
    when True sync Queue goto two_jobs;

-- État q_7
loc two_jobs: while True wait {}
    when True sync Off do {x'=0} goto off;
    when True sync Done goto printing;
    when True sync Job goto error;

-- État q_8
loc error: while True wait {}
    when True sync Off do {x'=0} goto off;

end
```

Exercice 2 Composition d'automates temporisés

On considère les automates temporisés suivants :



1. Décrire de façon formelle ou informelle l'ensemble des mots acceptés par l'automate

temporisé $\mathcal{B}$.

2. Composer les automates temporisés $\mathcal{A}$ et $\mathcal{B}$ en les synchronisant sur les actions *App* et *Exit*.

Solution de l'Exercice 2

1. De manière informelle, l'automate $\mathcal{B}$ accepte les mots tels que lorsqu'un premier signal *App* est reçu, un signal *Down* est envoyé entre 1 et 2 unités de temps après, à moins qu'un signal *Exit* soit reçu. De même, dès qu'un premier signal *Exit* est reçu, un signal *Up* est envoyé entre 1 et 2 unités de temps après, à moins qu'un signal *App* soit reçu. On considère qu'un signal *App* (resp. *Exit*) est "le premier" quand c'est le premier après un signal *Exit* (resp. *App*). Plus formellement, les mots acceptés sont les mots $w = (w_1, \tau_1) \cdot (w_2, \tau_2) \cdot \cdots \cdot (w_n, \tau_n)$ de $(\{App, Exit, Up, Down\} \times \mathbb{R}^*$ tels que si $w_i = Down$, alors il existe $j < i$ tel que $w_{j-1} \neq App$, $1 < \tau_i - \tau_j < 2$ et pour tout $k \in \{j, \ldots, i-1\}$, $w_k = App$. Notons qu'ici on note un mot comme une suite de lettres avec le temps absolu associé. Si l'on voulait noter un mot comme une suite de lettres avec le temps écoulé depuis la lettre précédente, la condition de temps serait $1 < \sum_{k=j+1}^{i} \tau_k < 2$.
2. Les états entourés de pointillées ne sont pas accessibles et donc peuvent être enlevés.