

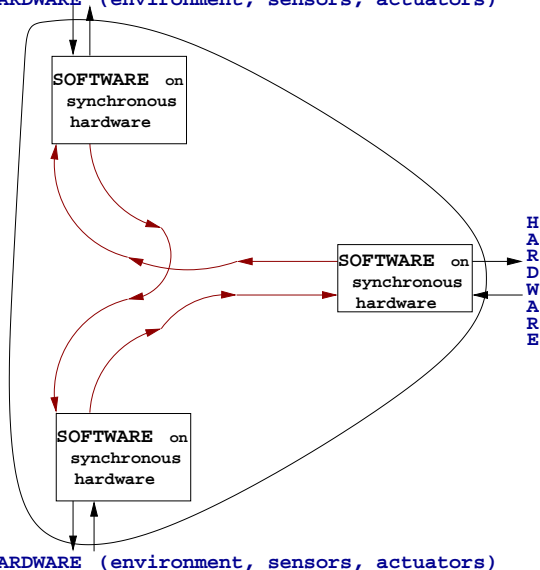
# Certification de la résistance des réseaux de programmes synchrones aux erreurs matérielles

Julien Bertrane  
bertrane@di.ens.fr

22 mars 2007

# Système type à analyser : systèmes embarqués

HARDWARE (environment, sensors, actuators)



HARDWARE (environment, sensors, actuators)

# Programme synchrone

- ▶ Initialize(S)
  - ▶ while true do
    - ★ (O, S) := Compute (S, I)
    - ★ wait for clock
  - ▶ od

où I : entrées, S : variables d'état, O : sortie

# Objectifs

- Analyser des Systèmes composés de **plusieurs programmes synchrones communicants** (chacun avec sa propre horloge). Pourquoi ?
  - ▶ certains systèmes embarqués sont **trop grands** pour une unique horloge : la transmission de l'information serait **trop lente**
  - ▶ les systèmes critiques sont **redondants** pour résister à la panne d'une unité ( $\Rightarrow$  plusieurs horloges)

# Objectifs

- Analyser des Systèmes composés de **plusieurs programmes synchrones communicants** (chacun avec sa propre horloge). Pourquoi ?
  - ▶ certains systèmes embarqués sont **trop grands** pour une unique horloge : la transmission de l'information serait **trop lente**
  - ▶ les systèmes critiques sont **redondants** pour résister à la panne d'une unité ( $\Rightarrow$  plusieurs horloges)
- la preuve doit être automatique, rapide, robuste aux changements mineurs dans le code. Basé sur la théorie de l'interprétation abstraite

# Spécifications à vérifier

- Spécifications de **sécurité**
  - ▶ **Pour tout comportement  $s$ , à tout instant  $t$ ,  $s(t) \neq false$**

# Spécifications à vérifier

- Spécifications de **sécurité**
  - ▶ Pour tout comportement  $s$ , à tout instant  $t$ ,  $s(t) \neq false$
- Spécifications **temporelles**
  - ▶ Pour tout comportement  $s$ , à aucun instant  $t$  on n'a :

pour tout  $t' \in [t, t + \alpha]$ ,  $s(t') = true$

# Spécifications à vérifier

- Spécifications de **sécurité**
  - ▶ **Pour tout comportement  $s$ , à tout instant  $t$ ,  $s(t) \neq false$**
- Spécifications **temporelles**
  - ▶ **Pour tout comportement  $s$ , à aucun instant  $t$  on n'a :**

pour tout  $t' \in [t, t + \alpha], s(t') = true$

- Spécifications **quantitatives**
  - ▶ les **sorties** de 2 systèmes redondants sont **égales au moins 50% du temps** durant chaque intervalle de temps de largeur minimale  $\delta$ .

# Modélisation

# Contraintes sur le système étudié

- Imperfections matérielles inévitables :
  - ▶ Les horloges des unités de commande se **désynchronisent**

# Contraintes sur le système étudié

- Imperfections matérielles inévitables :
  - ▶ Les horloges des unités de commande se **désynchronisent**
  - ▶ Les communications ne sont **pas instantanées**

# Contraintes sur le système étudié

- Imperfections matérielles inévitables :
  - ▶ Les horloges des unités de commande se **désynchronisent**
  - ▶ Les communications ne sont **pas instantanées**
  - ▶ Le temps de communication n'est **pas constant**

# Hypothèses pour ce modèle

- **Quasi-synchronie** :
  - ▶ désynchronisation : la durée de chaque cycle (période entre deux **ticks**) est dans  $[\alpha, \beta]$ ,  $\alpha > 0$ .
  - ▶ différent du quasi-synchronisme introduit par P. Caspi

# Hypothèses pour ce modèle

- **Quasi-synchronie** :
  - ▶ désynchronisation : la durée de chaque cycle (période entre deux **ticks**) est dans  $[\alpha, \beta]$ ,  $\alpha > 0$ .
  - ▶ différent du quasi-synchronisme introduit par P. Caspi
- Transmission en **série** entre 2 systèmes synchrones

# Hypothèses pour ce modèle

- **Quasi-synchronie** :
  - ▶ désynchronisation : la durée de chaque cycle (période entre deux **ticks**) est dans  $[\alpha, \beta]$ ,  $\alpha > 0$ .
  - ▶ différent du quasi-synchronisme introduit par P. Caspi
- Transmission en **série** entre 2 systèmes synchrones
- **Blackboard** à l'entrée de chaque système

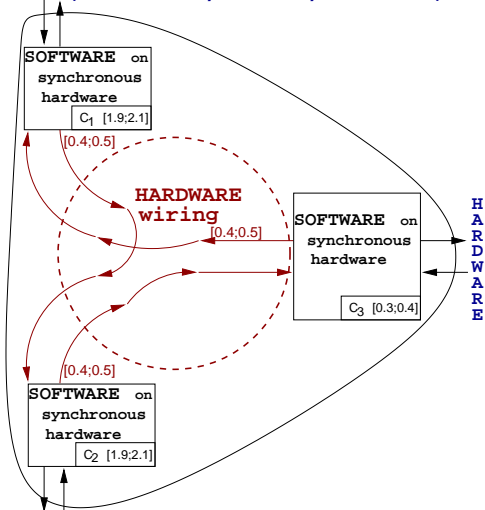
# Hypothèses pour ce modèle

- **Quasi-synchronie** :
  - ▶ désynchronisation : la durée de chaque cycle (période entre deux **ticks**) est dans  $[\alpha, \beta]$ ,  $\alpha > 0$ .
  - ▶ différent du quasi-synchronisme introduit par P. Caspi
- Transmission en **série** entre 2 systèmes synchrones
- **Blackboard** à l'entrée de chaque système
- **À l'initialisation** toutes les variables contiennent 0 ou *false*

# Système type à analyser

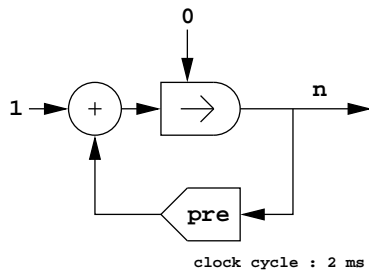
Détails quantifiés des imperfections matérielles

**HARDWARE (environment, sensors, actuators)**



**HARDWARE (environment, sensors, actuators)**

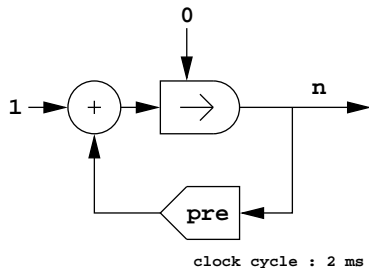
# Quelles sémantiques ?



- $n = 0 \rightarrow (1 + pre\ n)$
- $a_i = pre(b_i) \Leftrightarrow \begin{matrix} a_0 \text{ undefined} \\ a_{i+1} = b_i \end{matrix}$
- $a_i = b_i \rightarrow c_i \Leftrightarrow \begin{matrix} a_0 = b_0 \\ a_{i+1} = c_{i+1} \end{matrix}$

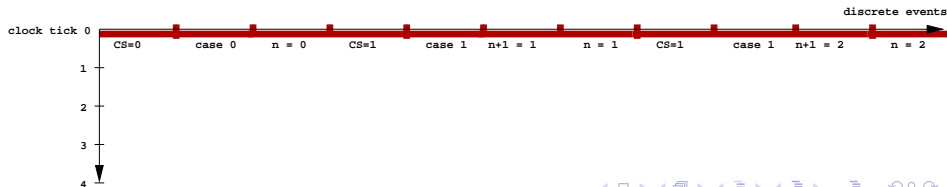
# Quelles sémantiques ?

La sémantique classique d'un programme est + ou - celle du C

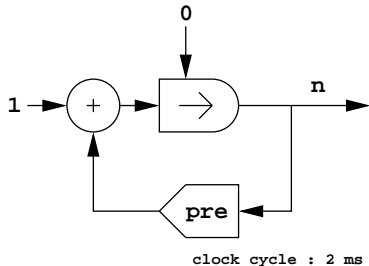


```
switch (global_state->current_state){
case 0 :
global_state->n = 0 ;
global_state->current_state = 1 ; break ;
break ;

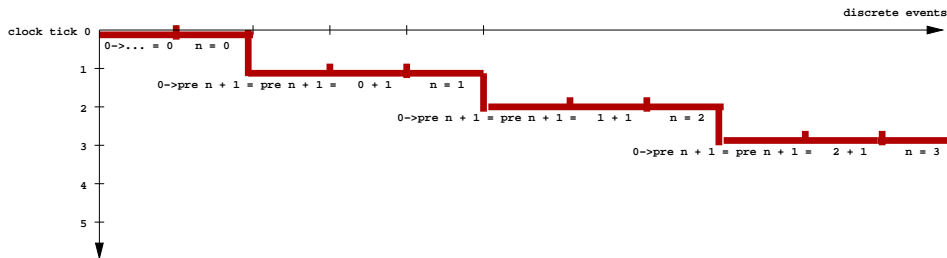
case 1 :
global_state->n = (global_state->n)+ 1 ;
global_state->current_state = 1 ; break ;
break ;}
```



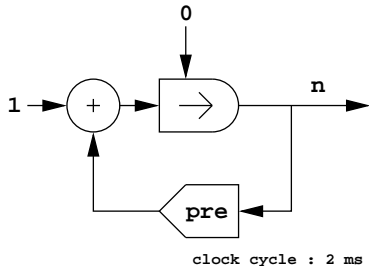
# Quelles sémantiques ?



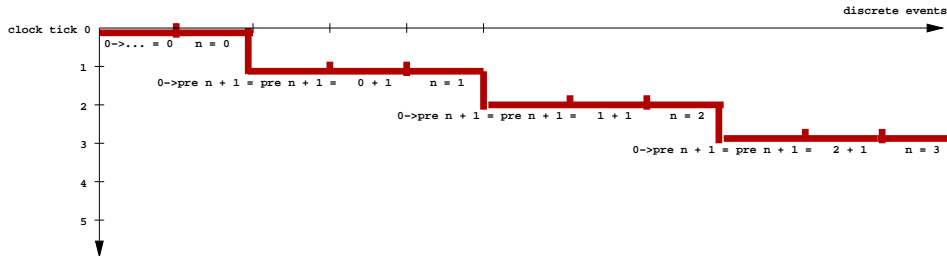
- Pour soft embarqué, il important de respecter des timings



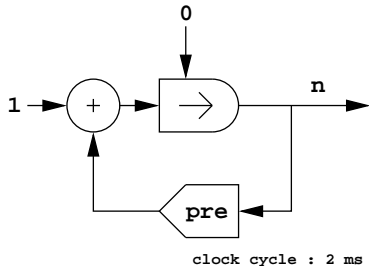
# Quelles sémantiques ?



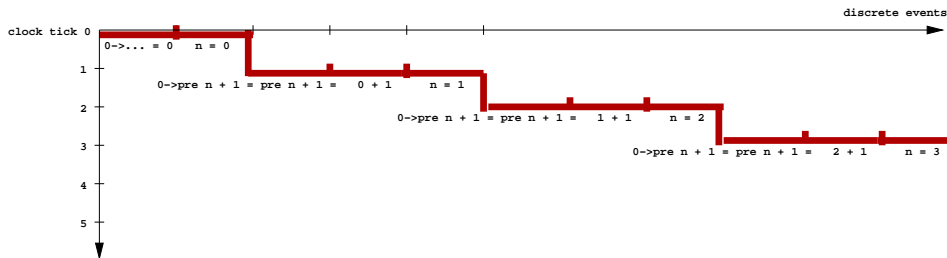
- Pour soft embarqué, il est important de respecter des timings
- en C, on a aucune garantie sur ces timings. Ils dépendent
  - ▶ de l'ordinateur exécutant le code
  - ▶ du compilateur,
  - ▶ des optimisations éventuelles ...



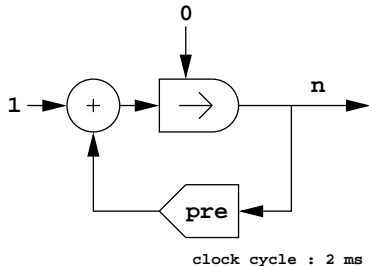
# Quelles sémantiques ?



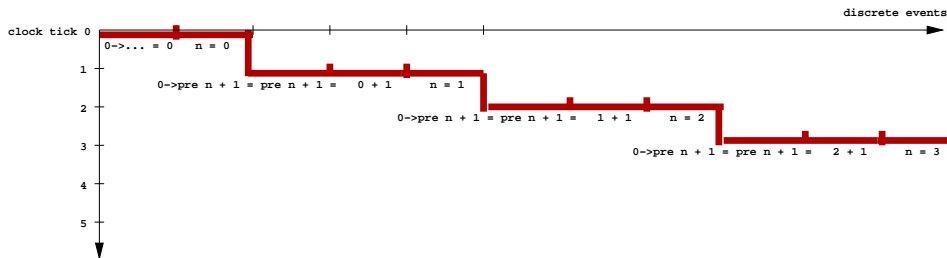
- On peut définir des cycles d'horloge rythmant l'exécution



## Quelles sémantiques ?



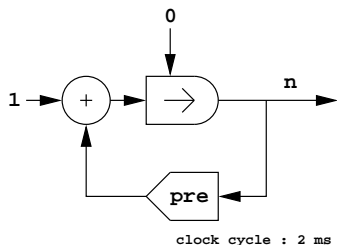
- On peut définir des cycles d'horloge rythmant l'exécution
- Question : l'ensemble des instructions réparties sur un niveau a-t-il assez de temps pour s'exécuter entre deux tick consécutifs d'horloge (WCET) ?



# Avec quelles sémantiques doit-on travailler

On est donc conduit à définir une autre sémantique : celle du Lustre

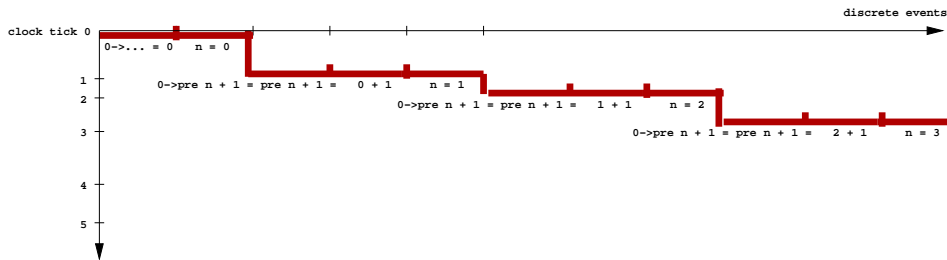
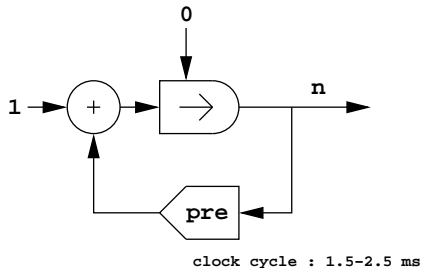
$$\llbracket S \rrbracket : P \rightarrow (\mathbb{N} \rightarrow (\mathbb{N} \cup \{\text{undefined}\}))$$



	$n$	$\text{pre } n$	$1 + \text{pre } n$	$0 \rightarrow 1 + \text{pre } n$
clock tick 0	0	?	?	0
1	1	0	1	1
2	2	1	2	2
3	3	2	3	3
4	4	3	4	4
5	5	4	5	5

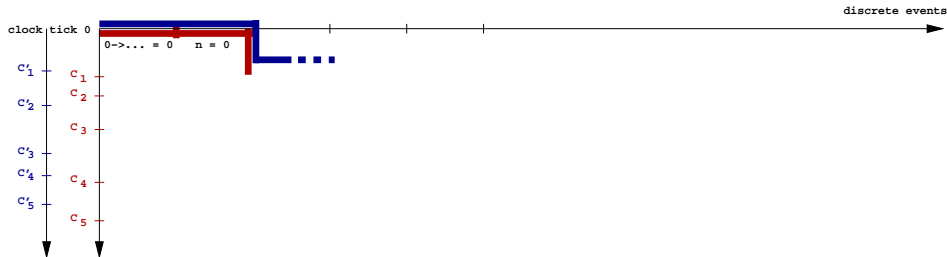
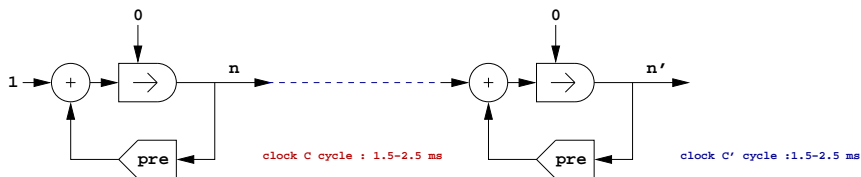
# Avec quelles sémantiques doit-on travailler

- Et si l'horloge est imprécise ?



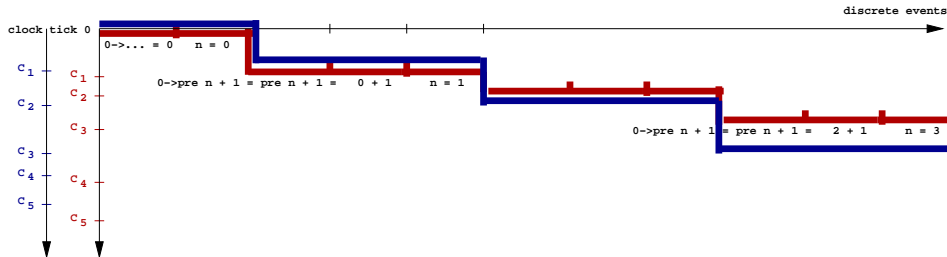
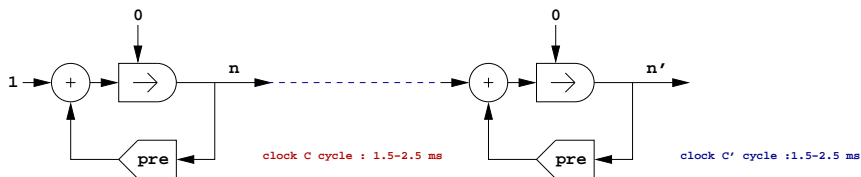
# Avec quelles sémantiques doit-on travailler

Cas de plusieurs systèmes, chacun ayant son horloge imprécise :



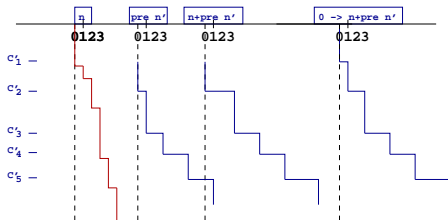
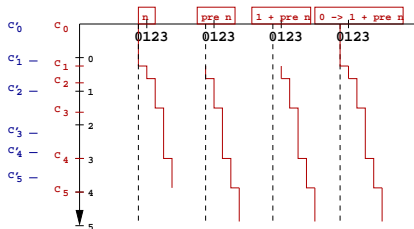
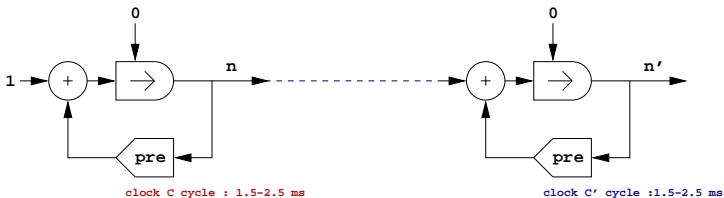
# Avec quelles sémantiques doit-on travailler

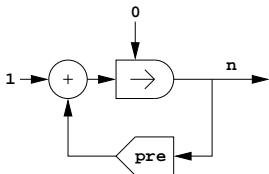
Cas de plusieurs systèmes, chacun ayant son horloge imprécise :



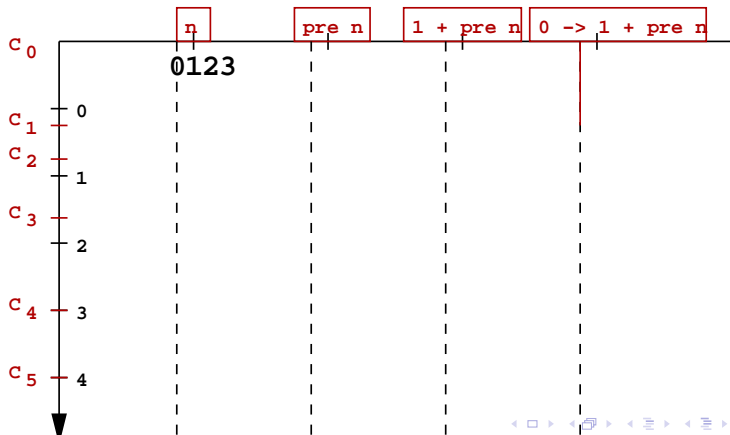
# Avec quelles sémantiques doit-on travailler

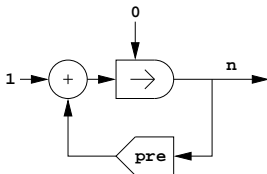
Une sémantique totalement continue permet d'étudier la communication de plusieurs systèmes



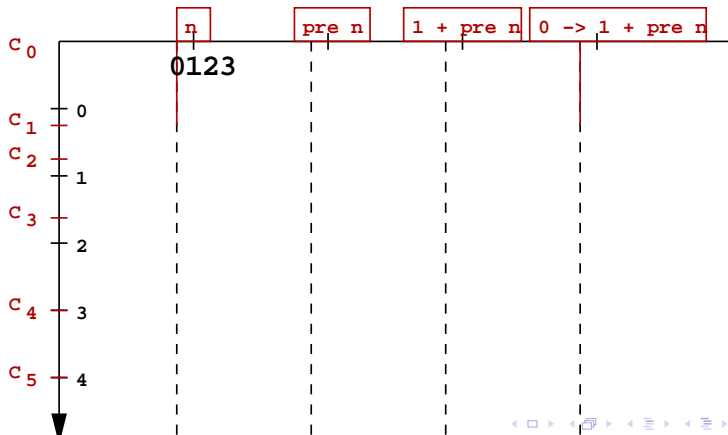


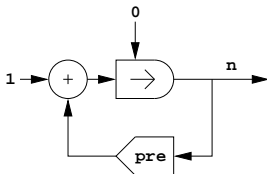
clock cycle : 1.5-2.5 ms



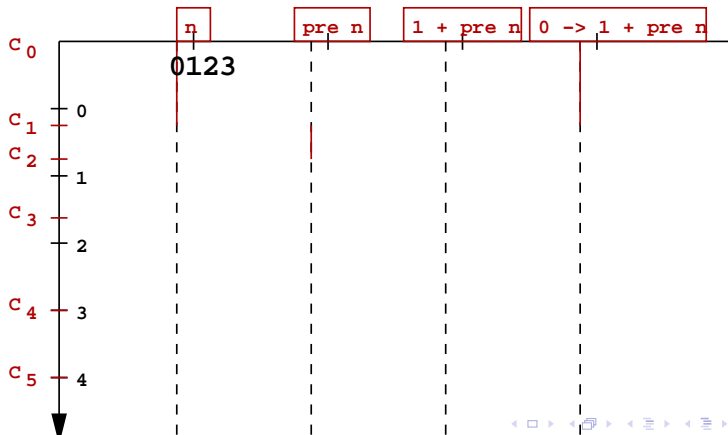


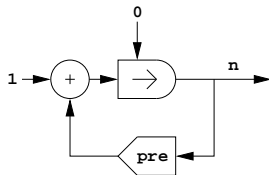
clock cycle : 1.5-2.5 ms



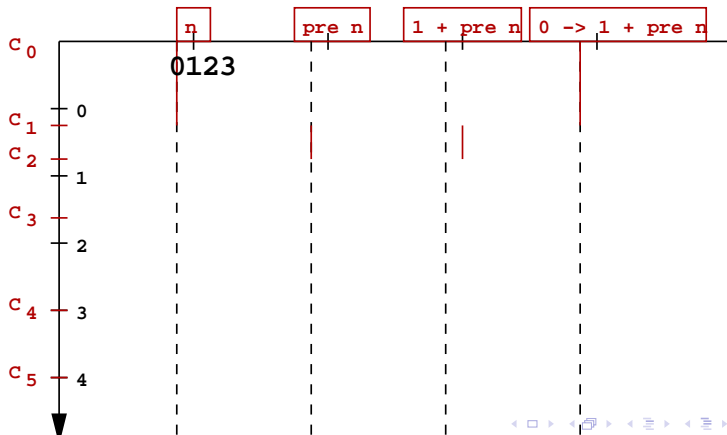


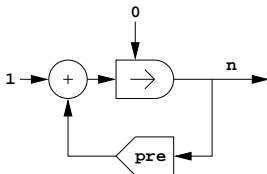
clock cycle : 1.5-2.5 ms



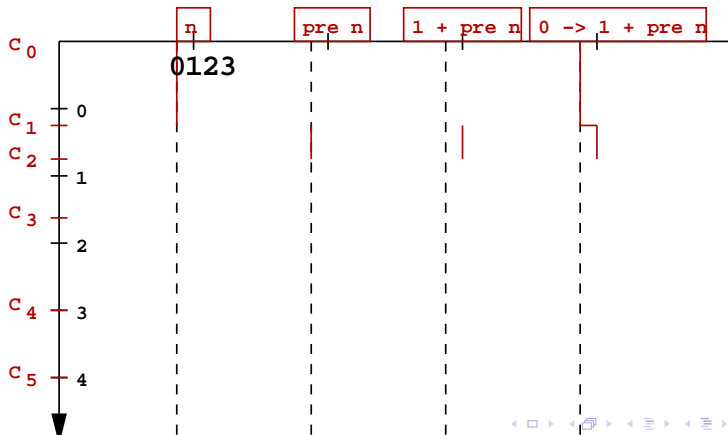


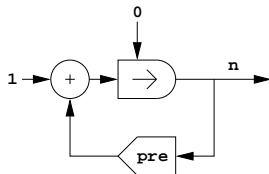
clock cycle : 1.5-2.5 ms



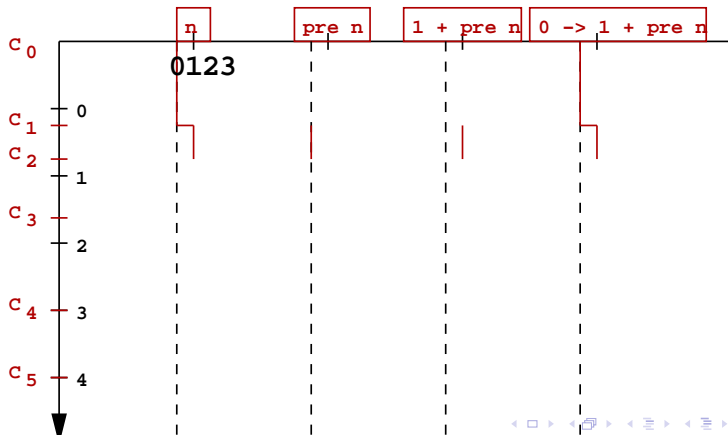


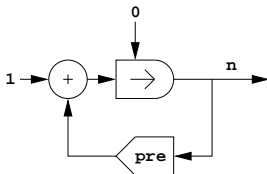
clock cycle : 1.5-2.5 ms



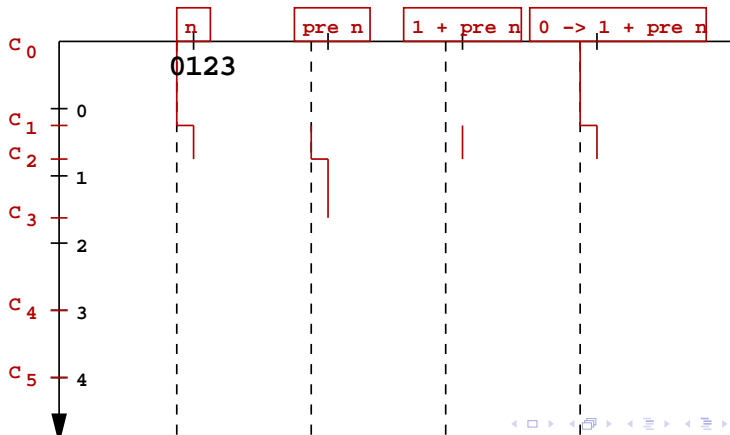


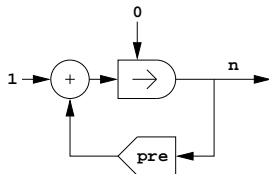
clock cycle : 1.5-2.5 ms



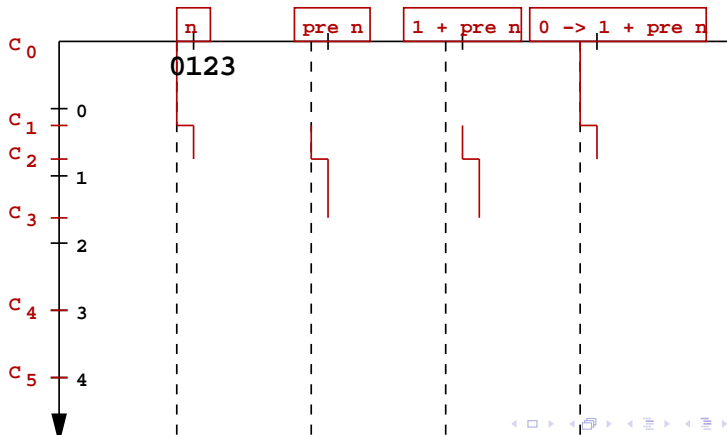


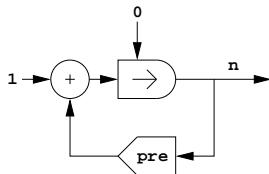
clock cycle : 1.5-2.5 ms



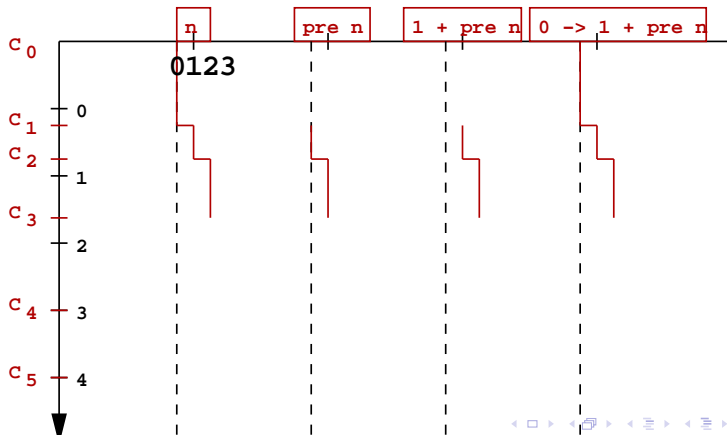


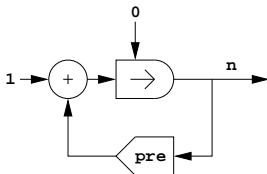
clock cycle : 1.5-2.5 ms



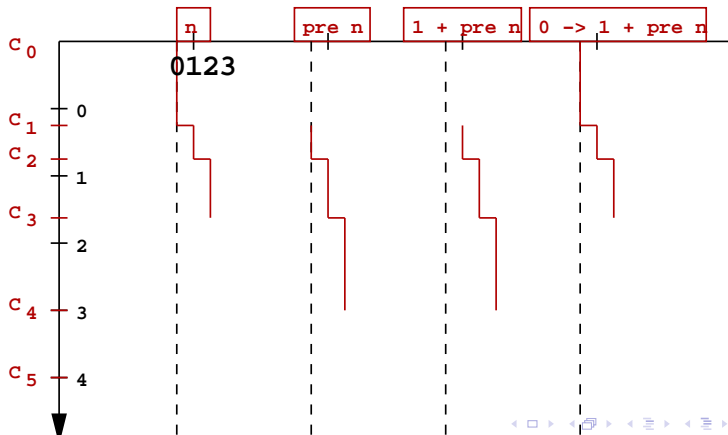


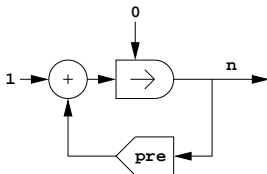
clock cycle : 1.5-2.5 ms



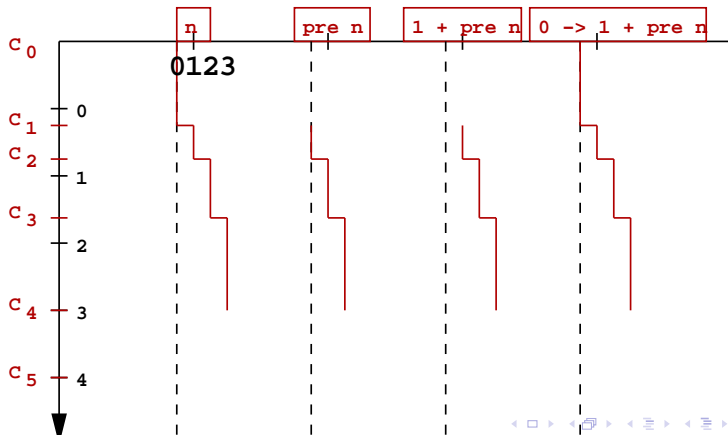


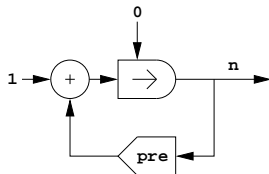
clock cycle : 1.5-2.5 ms



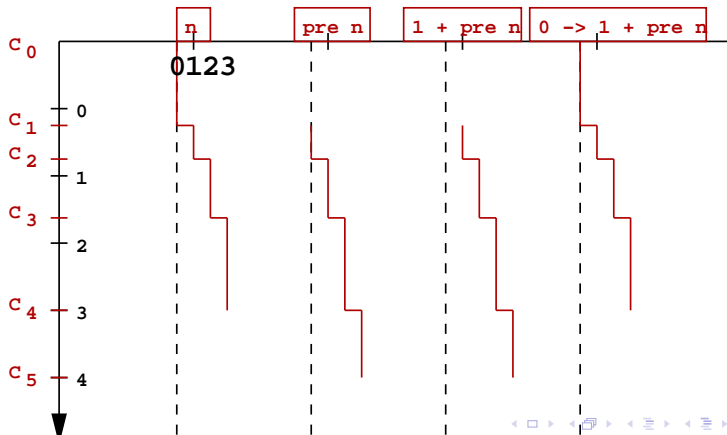


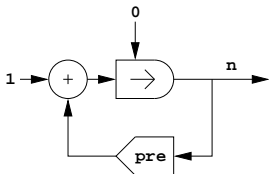
clock cycle : 1.5-2.5 ms



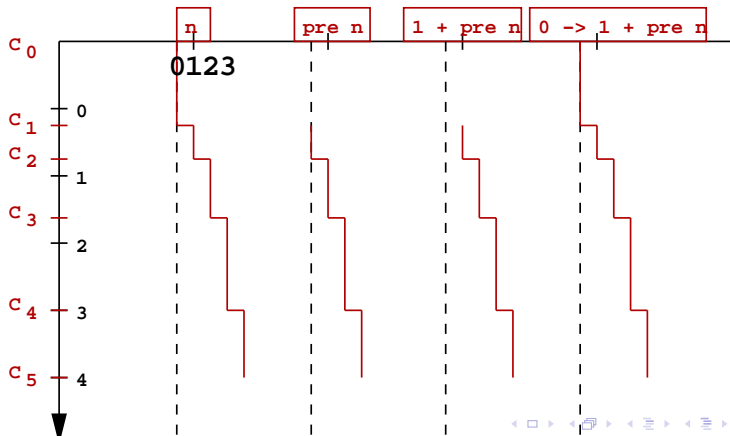


clock cycle : 1.5-2.5 ms

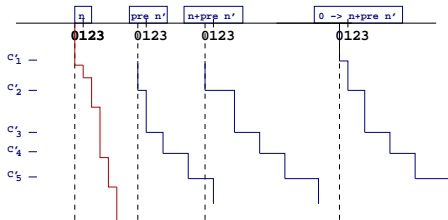
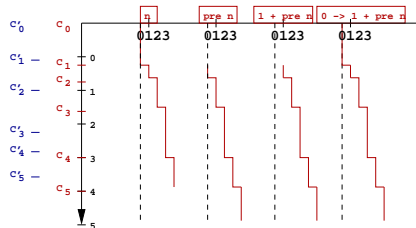
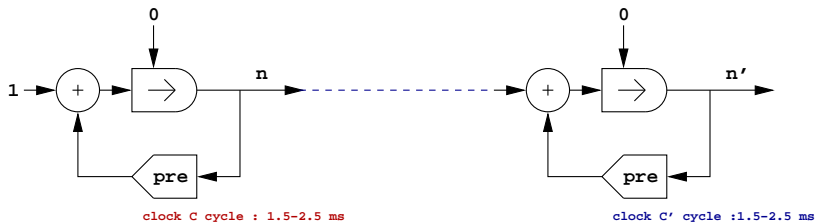




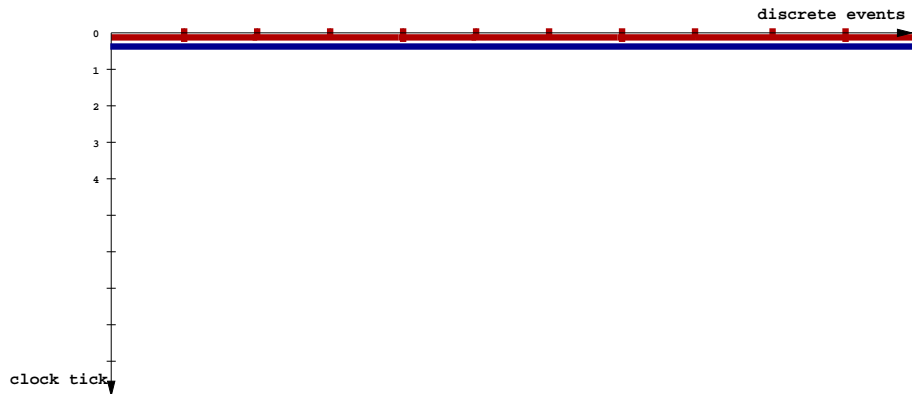
clock cycle : 1.5-2.5 ms



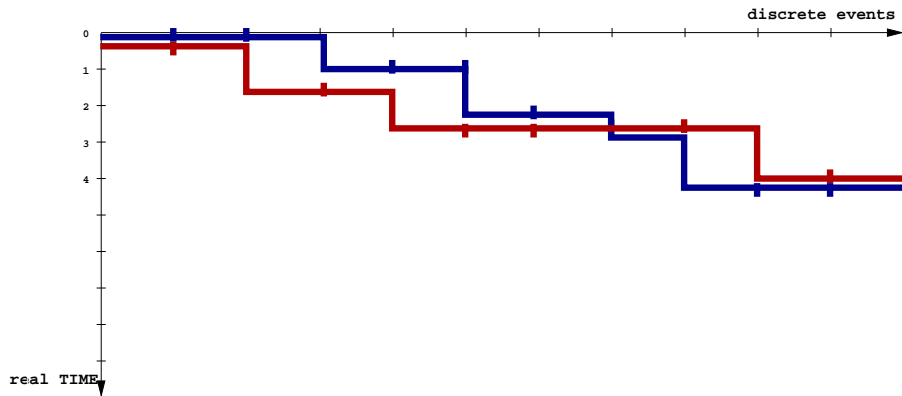
# Avec quelles sémantiques doit-on travailler



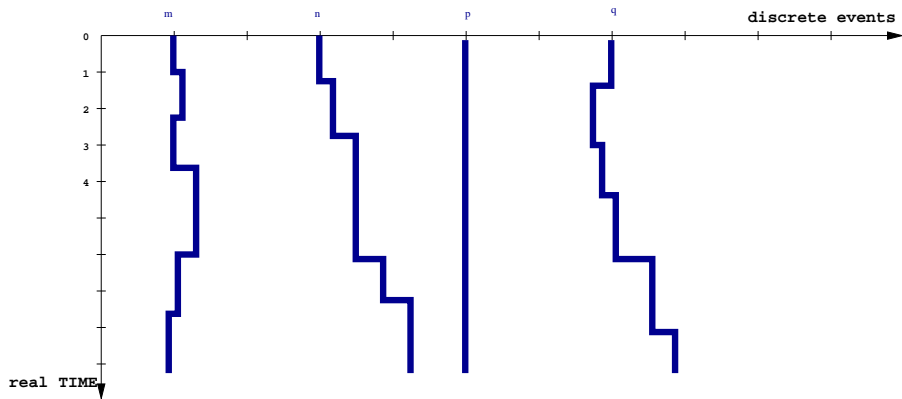
# Résumé des modifications de la sémantique



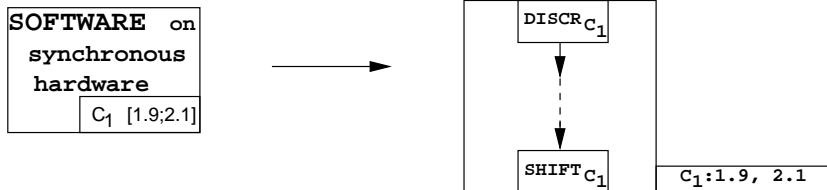
# Résumé des modifications de la sémantique



# Résumé des modifications de la sémantique

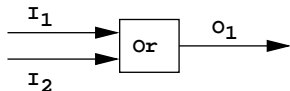


# Comportement d'un système synchrone



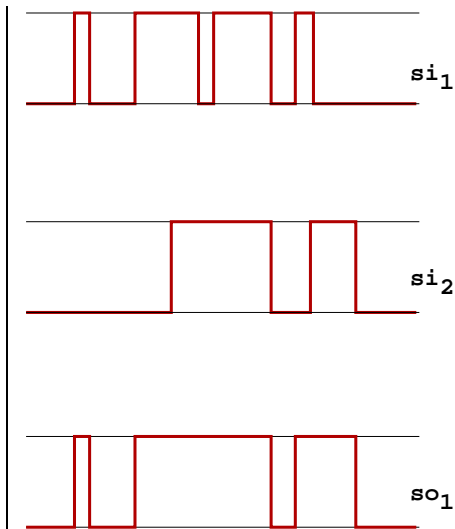
- une horloge est une fonction  $:\mathbb{N} \rightarrow \mathbb{R}^+$
- paramétrée par  $:\ [\alpha, \beta]$ , avec  $\alpha, \beta \in \mathbb{R}^+$  et  $0 < \alpha \leq \beta$
- une horloge  $c$  satisfait  $[\alpha, \beta]$  ssi  $c_{n+1} - c_n \in [\alpha, \beta]$
- $DISCR_{C_1}$  modélise la lecture periodique du blackboard à l'entrée du système
- $SHIFT_{C_1}$  modélise l'attente du prochain tick d'horloge, et l'émission du résultat lors de ce tick

## Sémantique des opérateurs atemporels

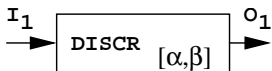


$$so_1(t) = \begin{cases} \bullet \textit{true} & \text{if } si_1(t) = \textit{true} \\ & \text{or } si_2(t) = \textit{true} \\ \bullet \textit{false} & \text{else} \end{cases}$$

$$so_1 \triangleq \Psi_{OR}(si_1, si_2)$$



# Sémantique des opérateurs temporels

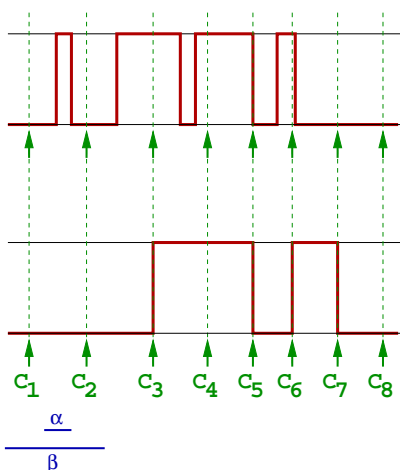


Il existe une horloge  $C$  de paramètre  $[\alpha, \beta]$

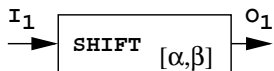
i.e. :  $c_{n+1} - c_n \in [\alpha, \beta]$  telle que

$$so_1(t) = \begin{cases} \bullet \text{ false} & \text{if } t < c(0) \\ \bullet \text{ si}_1(c_n) & \text{if } t \in [c_n, c_{n+1}) \end{cases}$$

$$so_1 \triangleq \Psi_{\text{DISCR}_{[\alpha, \beta]}}(si_1)$$



## Sémantique des opérateurs temporels

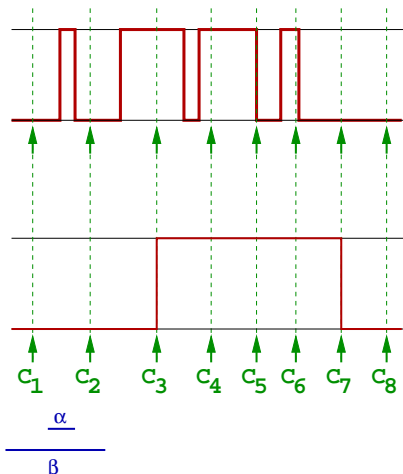


Il existe une horloge  $C$  de  
paramètre  $[\alpha, \beta]$

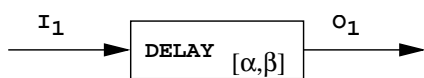
i.e. :  $c_{n+1} - c_n \in [\alpha, \beta]$  telle que

$$s_{O_1}(t) = \begin{cases} \bullet 0 & \text{if } t < c(0) \\ \bullet \lim_{\substack{t \rightarrow c_n \\ t < c_n}} s_{I_1}(t) & \text{if } t \in [c_n, c_{n+1}) \end{cases}$$

$$s_{O_1} \triangleq \Psi_{\text{SHIFT}_{[\alpha, \beta]}}(s_{I_1})$$



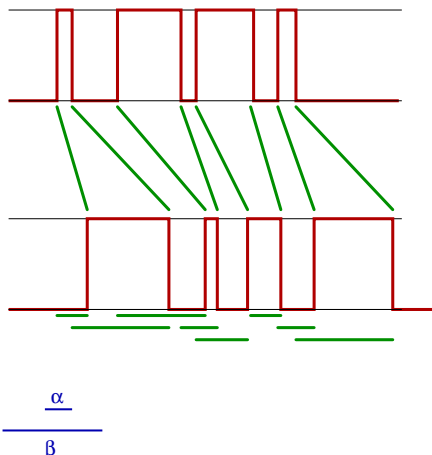
# Sémantique des opérateurs temporels



Il existe une fonction  $\delta$   
 bijection croissante de  $\mathbb{R} \rightarrow \mathbb{R}$   
 de paramètre  $[\alpha, \beta]$  i.e. :  
 $\forall t \in \mathbb{R}, \delta(t) - t \in [\alpha, \beta]$  telle que

$$so_1(t) = si_1(\delta(t))$$

$$so_1 \triangleq \Psi_{\text{DELAY}_{[\alpha, \beta]}}(si_1)$$



## semantique concrète

- un point de contrôle  $\triangle$  equivalent graphique d'une variable
- L'ensemble des points de contrôle est noté  $P$ .

## semantique concrète

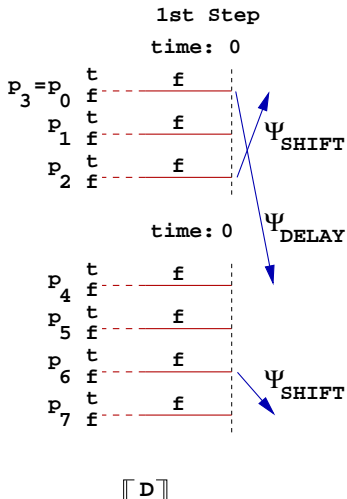
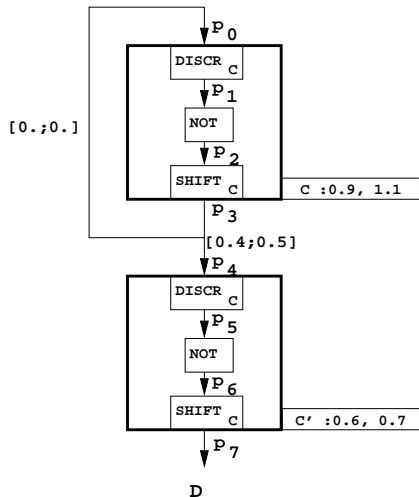
- un point de contrôle  $\triangle$  equivalent graphique d'une variable
- L'ensemble des points de contrôle est noté  $P$ .
- $D$  : ensemble de systèmes synchrones,  $[[D]] \subseteq P \rightarrow (\mathbb{R}^+ \rightarrow \mathbb{B})$

# semantique concrète

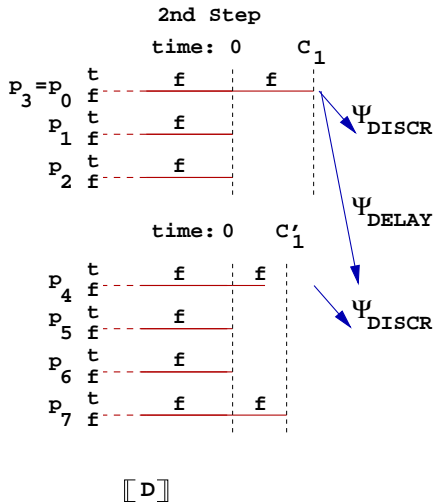
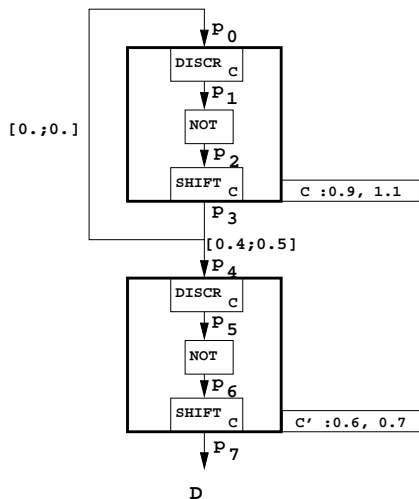
- un point de contrôle  $\triangleq$  équivalent graphique d'une variable
- L'ensemble des points de contrôle est noté  $P$ .
- $D$  : ensemble de systèmes synchrones,  $\llbracket D \rrbracket \subseteq P \rightarrow (\mathbb{R}^+ \rightarrow \mathbb{B})$

- $\left( \begin{array}{ccc} p_1 & \mapsto & s_1 \\ & \vdots & \\ p_{\#P-1} & \mapsto & s_{\#P-1} \end{array} \right) \in \llbracket D \rrbracket$  ssi il satisfait les équations de toutes les portes de  $D$ .

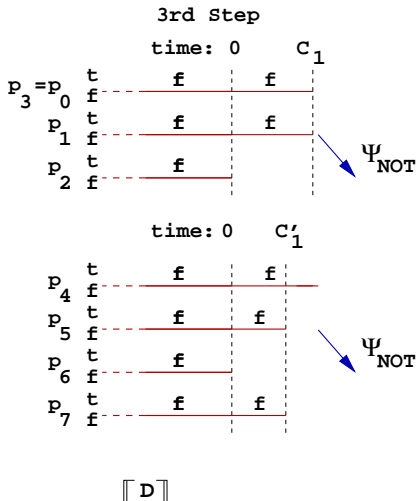
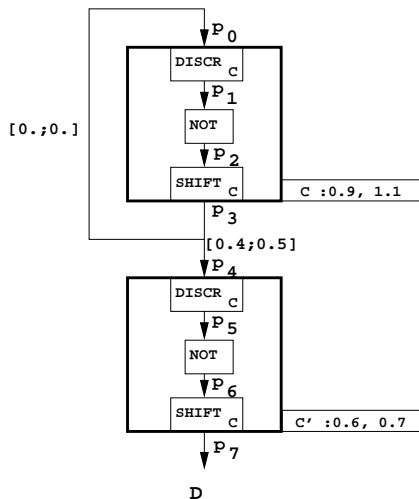
# Syntaxe et sémantique



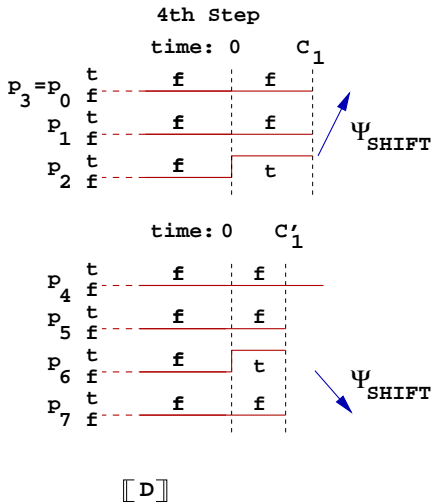
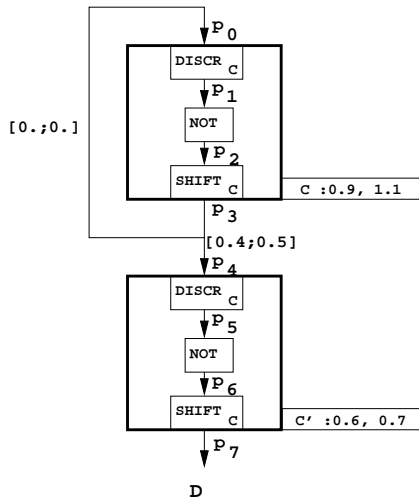
# Syntaxe et sémantique



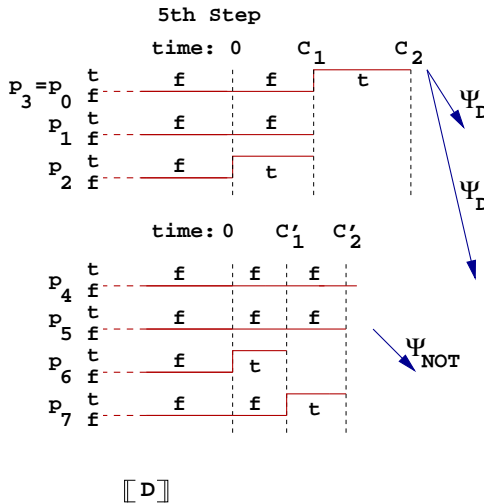
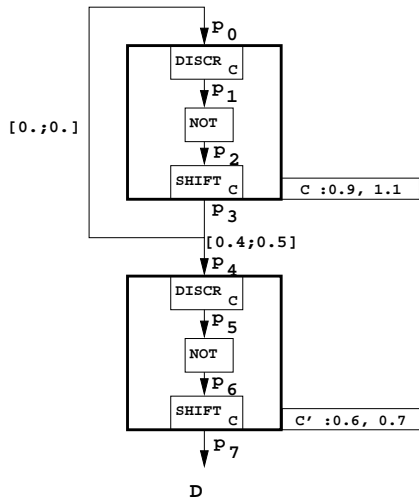
# Syntaxe et sémantique



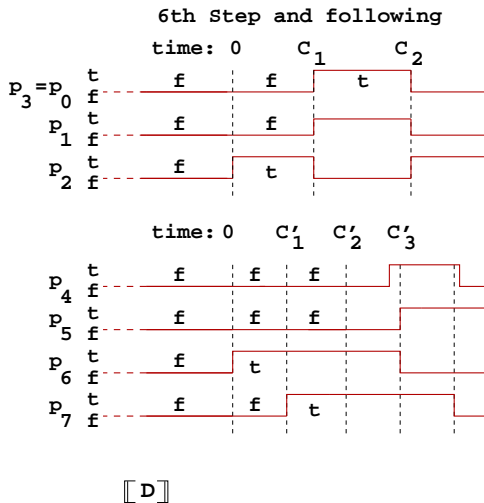
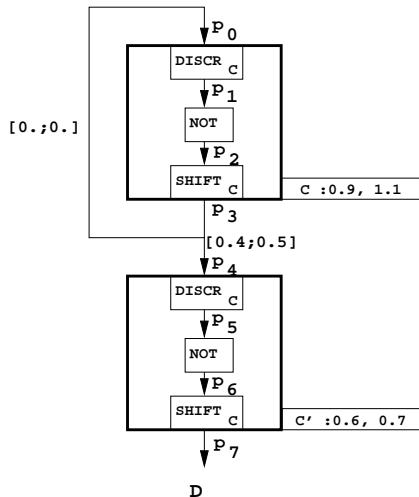
# Syntaxe et sémantique



# Syntaxe et sémantique

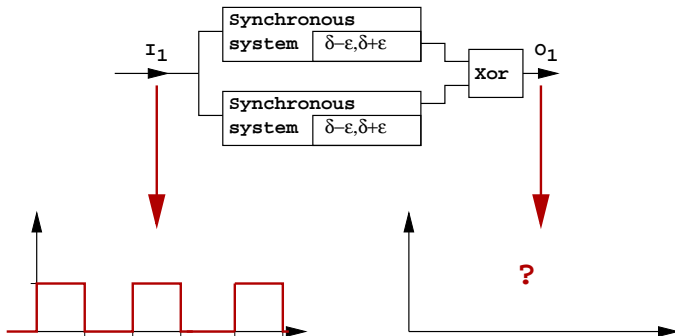


# Syntaxe et sémantique

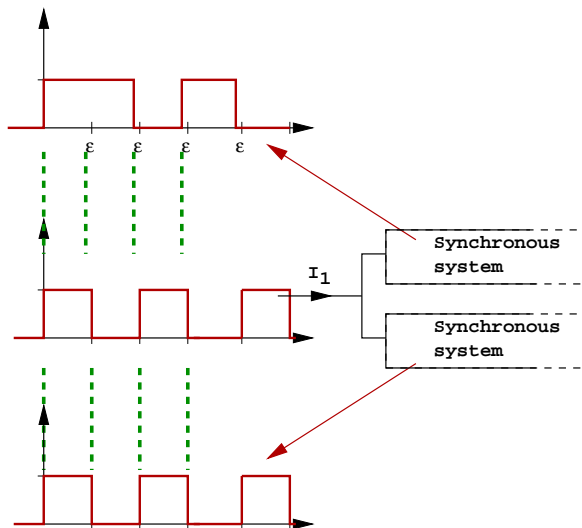


## Difficultés résultantes

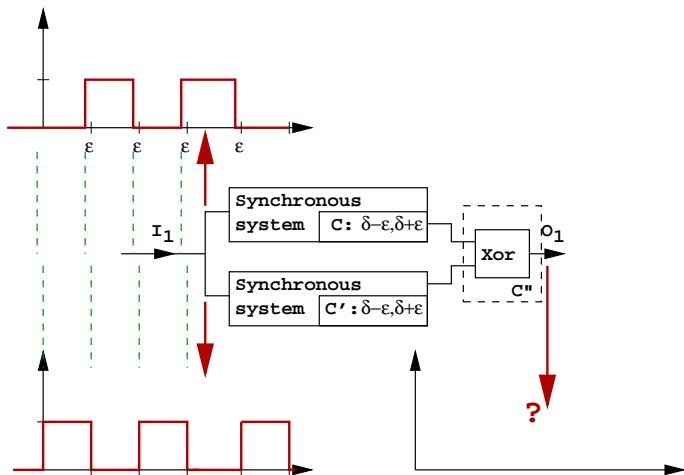
- imprécision de l'horloge (c-à-d *quasi-synchronous* à la place de *synchronous*)  $\Rightarrow$  nombre de comportements **non dénombrable**



# Difficultés résultantes

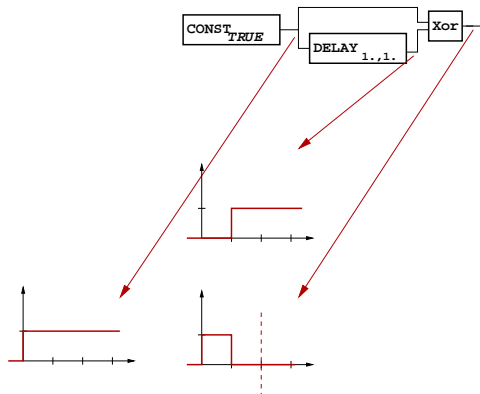


# Difficultés résultantes

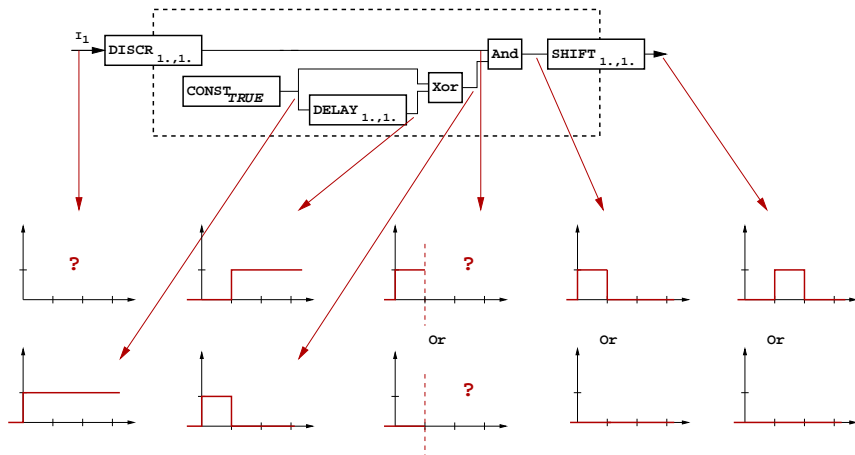


# Analyse abstraite

# Analyse par regroupement ?

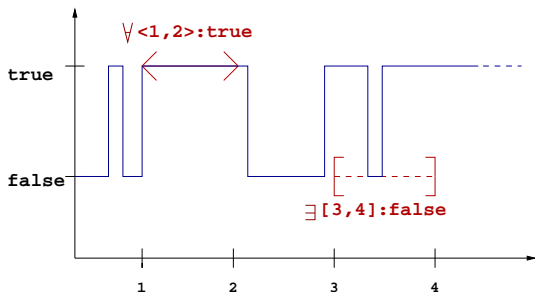


# Analyse par regroupement ?



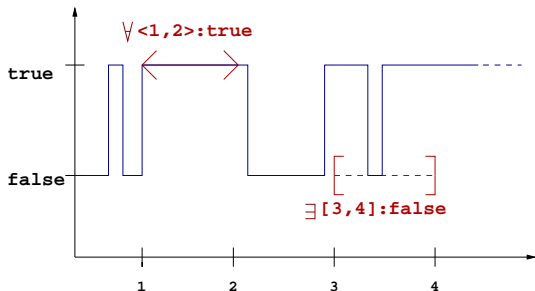
- Manipulation possible de données inconnues ou “infinies”

# 1er domaine abstrait : les contraintes



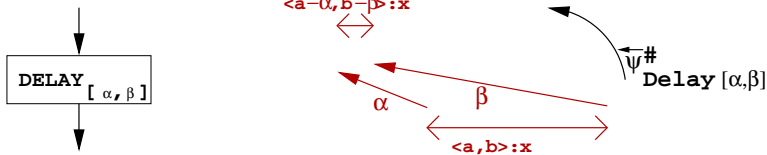
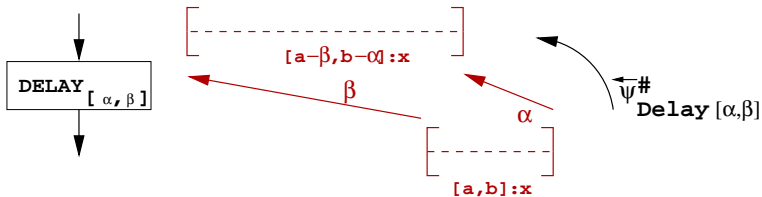
- La constraint  $\exists[a; b] : x$  garantit que les signaux prennent la valeur  $x$  **au moins une fois** pendant  $[a; b]$ .
- La constraint  $\forall \langle a; b \rangle : x$  garantit que les signaux prennent la valeur  $x$  **durant tout l'intervalle**  $[a; b]$ .

# 1er domaine abstrait : les contraintes



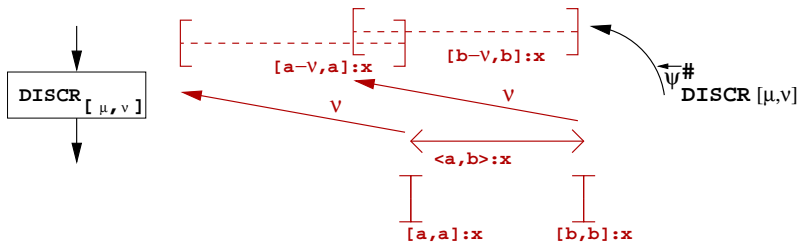
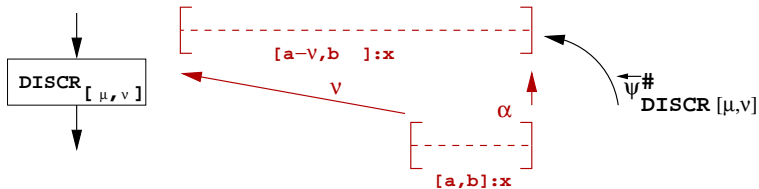
- La constraint  $\exists[a; b] : x$  garantit que les signaux prennent la valeur  $x$  **au moins une fois** pendant  $[a; b]$ .
- La constraint  $\forall \langle a; b \rangle : x$  garantit que les signaux prennent la valeur  $x$  **durant tout l'intervalle**  $[a; b]$ .
- Permet d'exprimer de nombreuses **propriétés temporelles**

## Opérateurs Abstraits et Contraintes : un exemple



- $\overleftarrow{\Psi}^{\#}_{\text{DELAY}[\alpha, \beta]}(\exists \langle a; b \rangle : x) \triangleq \exists \langle a - \beta; b - \alpha \rangle : x$
- $\overleftarrow{\Psi}^{\#}_{\text{DELAY}[\alpha, \beta]}(\forall \langle a; b \rangle : x) \triangleq \forall \langle a - \alpha; b - \beta \rangle : x$

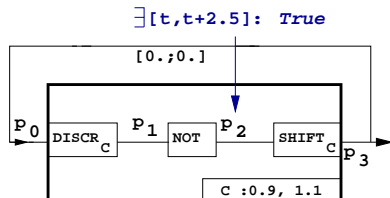
## Opérateurs Abstraits et Contraintes : exemple



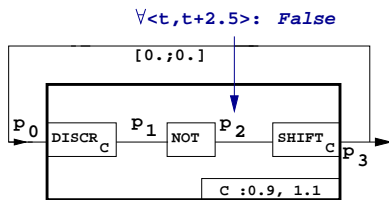
- $\overleftarrow{\Psi}^{\#}_{\text{DISCR}[\mu, \nu]}(\exists[a; b] : x) \triangleq \exists[a - \nu; b] : x$
- $\overleftarrow{\Psi}^{\#}_{\text{DISCR}[\mu, \nu]}(\forall\langle a; b \rangle : x) \triangleq \bigwedge_{u \in [a, b]} \exists[u - \nu; u] : x$

# Analyse dans le domaine abstrait des **Contraintes**

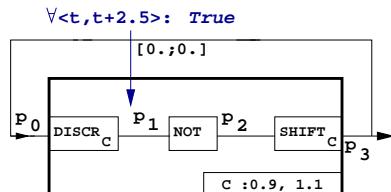
- Prouver la propriété abstraite suivante.

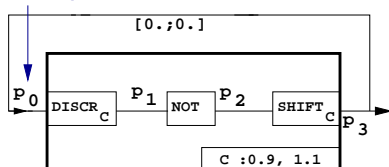


# Analyse dans le domaine abstrait des **Contraintes**

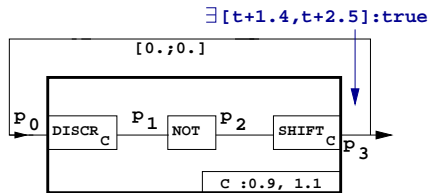


# Analyse dans le domaine abstrait des **Contraintes**

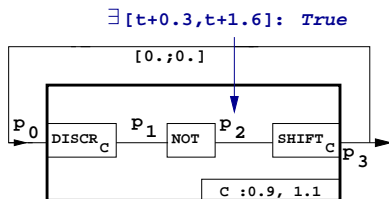


Analyse dans le domaine abstrait des **Contraintes**
 $\exists [t+1.4, t+2.5]: \text{True}$ 


# Analyse dans le domaine abstrait des **Contraintes**



# Analyse dans le domaine abstrait des **Contraintes**



Au point  $p_2$  : 2 contraintes doivent être satisfaites :

- $\forall \langle t; t + 2.5 \rangle : \text{False}$
- $\exists [t + 0.2; t + 1.6] : \text{True}$

ce qui est impossible et invalide donc l'hypothèse initiale

$\forall \langle t; t + 2.5 \rangle : \text{False}$ .

- Donc  $\exists [t; t + 2.5] : \text{True}$  est certifié.

## Analyse par interprétation abstraite

- $\llbracket D \rrbracket$  : sémantique de l'ensemble de systèmes  $D$ .
- Pour toute propriété  $P$ ,  $[P]$  est l'ensemble de comportements satisfaisant  $P$ .
- **Ancien objectif** : Prouver que  $\llbracket D \rrbracket \subseteq [P]$ .

# Analyse par interprétation abstraite

- $\llbracket D \rrbracket$  : sémantique de l'ensemble de systèmes  $D$ .
- Pour toute propriété  $P$ ,  $[P]$  est l'ensemble de comportements satisfaisant  $P$ .
- **Ancien objectif** : Prouver que  $\llbracket D \rrbracket \subseteq [P]$ .
- **Alternative** : Prouver que  $\llbracket D \rrbracket \cap [\neg P] = \emptyset$ .

# Analyse par interprétation abstraite

- $\llbracket D \rrbracket$  : sémantique de l'ensemble de systèmes  $D$ .
- Pour toute propriété  $P$ ,  $[P]$  est l'ensemble de comportements satisfaisant  $P$ .
- **Ancien objectif** : Prouver que  $\llbracket D \rrbracket \subseteq [P]$ .
- **Alternative** : Prouver que  $\llbracket D \rrbracket \cap [\neg P] = \emptyset$ .
- **Or** :

$$\llbracket D \rrbracket \cap [\neg P] \subseteq \text{gfp}_{[\neg P]}(\Psi \cap Id)$$

# Analyse par interprétation abstraite

- $\llbracket D \rrbracket$  : sémantique de l'ensemble de systèmes  $D$ .
- Pour toute propriété  $P$ ,  $\llbracket P \rrbracket$  est l'ensemble de comportements satisfaisant  $P$ .
- **Ancien objectif** : Prouver que  $\llbracket D \rrbracket \subseteq \llbracket P \rrbracket$ .
- **Alternative** : Prouver que  $\llbracket D \rrbracket \cap \llbracket \neg P \rrbracket = \emptyset$ .

- **Or** :

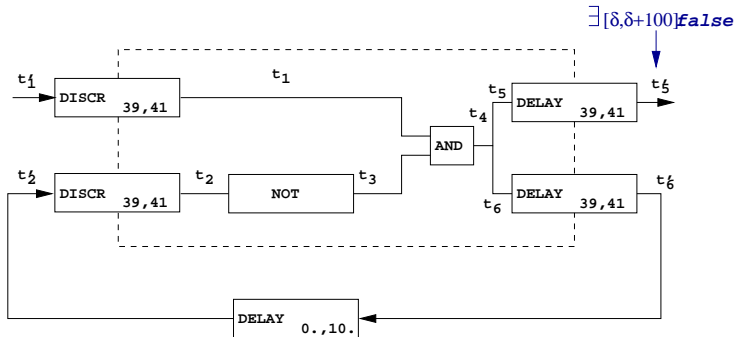
$$\llbracket D \rrbracket \cap \llbracket \neg P \rrbracket \subseteq \text{gfp}_{\llbracket \neg P \rrbracket}(\Psi \cap Id)$$

- **Nouvel objectif** :

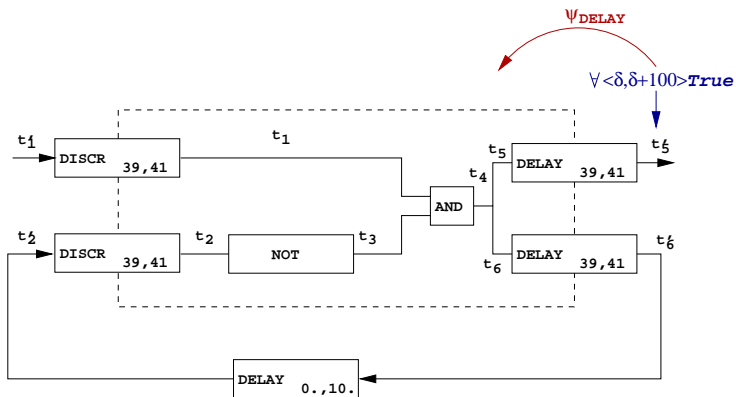
$$\text{gfp}_{\llbracket \neg P \rrbracket}(\Psi \cap Id) \subseteq^? \emptyset$$

# Exemple d'analyse : Itération jusqu'au point fixe vide

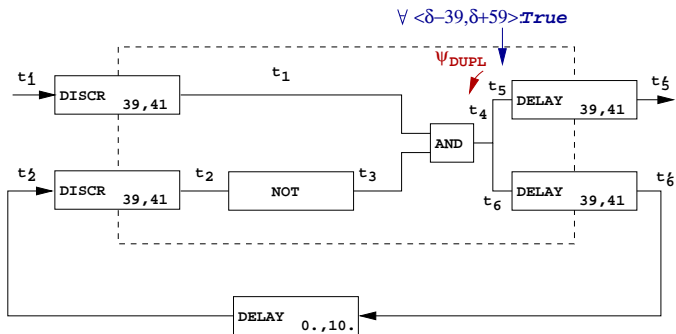
- prouver la propriété abstraite suivante



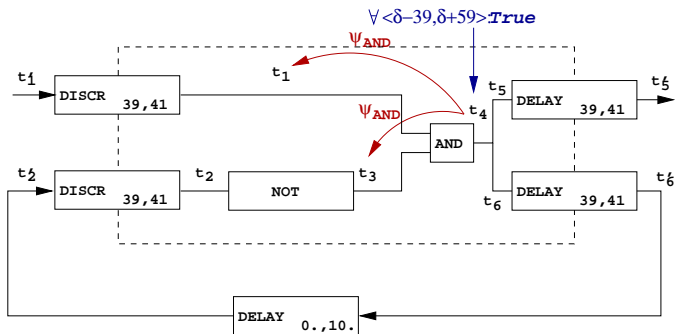
## Exemple : Itération jusqu'au point fixe (vide ?)



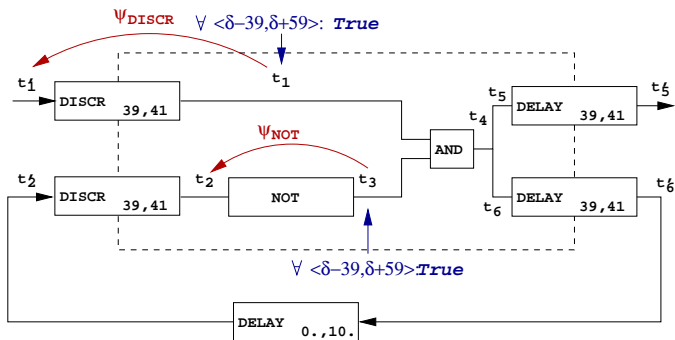
## Exemple : Itération jusqu'au point fixe (vide ?)



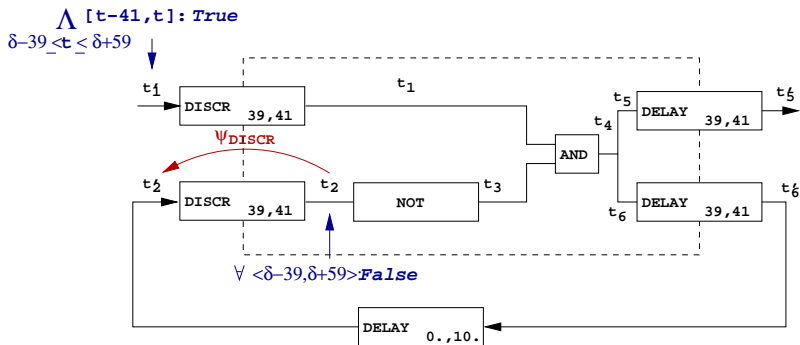
## Exemple : Itération jusqu'au point fixe (vide ?)



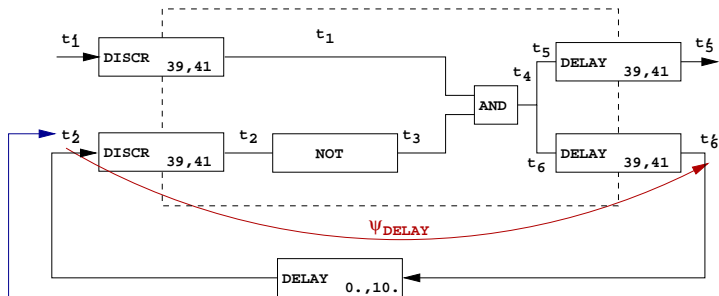
## Exemple : Itération jusqu'au point fixe (vide ?)



## Exemple : Itération jusqu'au point fixe (vide ?)

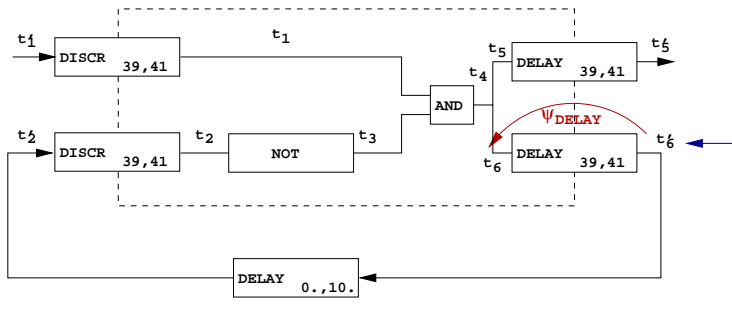


## Exemple : Itération jusqu'au point fixe (vide ?)



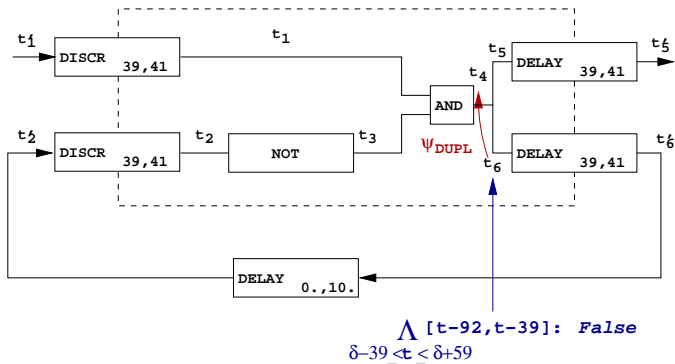
$\bigwedge [t-41, t]: \text{False}$   
 $\delta-39 \leq t \leq \delta+59$

## Exemple : Itération jusqu'au point fixe (vide ?)

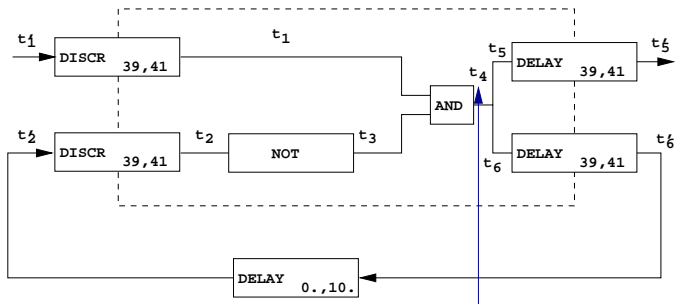


$\bigwedge [t-51, t]: \text{False}$   
 $\delta-39 \leq t \leq \delta+59$

## Exemple : Itération jusqu'au point fixe (vide ?)



## Exemple : Itération jusqu'au point fixe (vide ?)



$$\bigwedge [t-92, t-39]: \text{False}$$

$$\delta-39 \leq t \leq \delta+59$$

## Exemple : Itération jusqu'à un point fixe vide

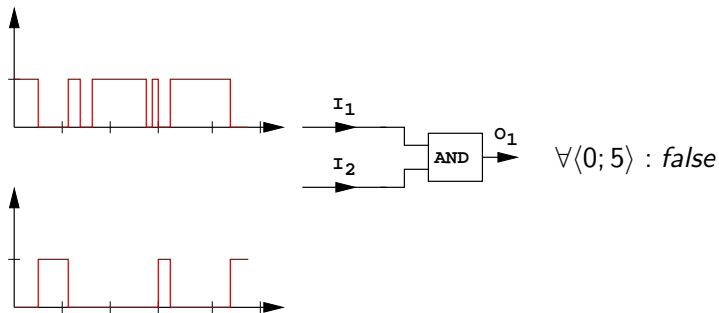
Les signaux doivent donc satisfaire au point de contrôle  $t_4$  :

$$\langle \delta - 39, \delta + 59 \rangle : \textit{True} \textbf{ and } \bigwedge_{\delta - 39 \leq t \leq \delta + 59} ([t - 92, t - 39] : \textit{False})$$

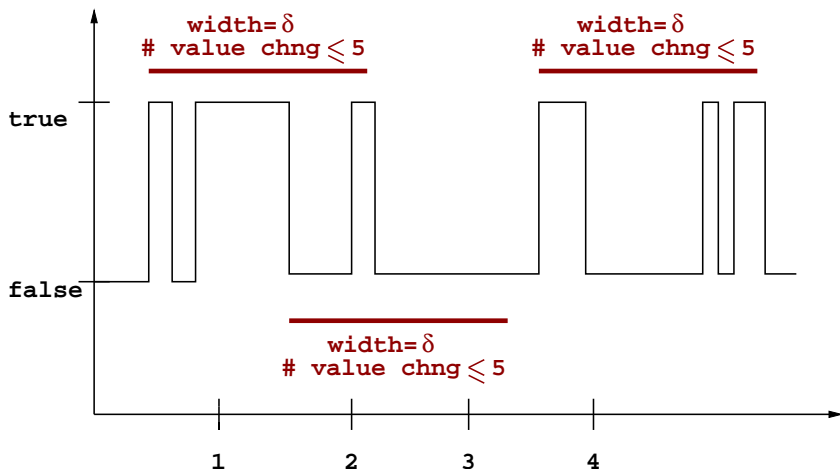
ce qui implique  $[\delta - 33 = \delta + 59 - 92, \delta + 20 = \delta + 59 - 39] : \textit{False}$

# Faiblesses du domaine des contraintes

- Domaine précis dans le cas de : DELAY, DISCR, SHIFT, NOT,
- Grande perte de précision dans le cas de : AND, OR, XOR

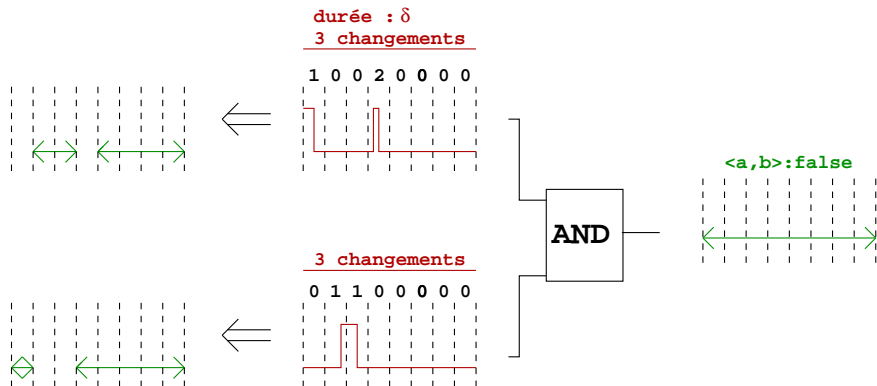


## 2ème Domaine Abstrait : Domaine du comptage des Changements de valeurs

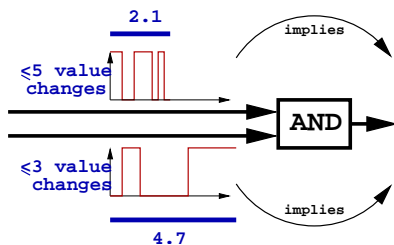


## 2ème Domaine Abstrait : Domaine du comptage des Changements de valeurs

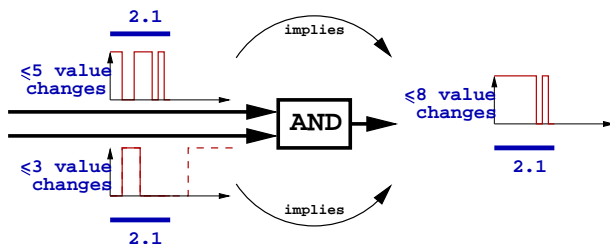
### Utilisation



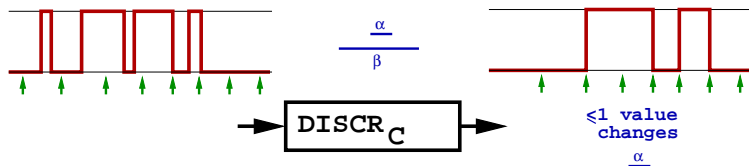
un opérateur abstrait **indépendant** du temps dans le domaine du comptage des Changements de valeurs



un opérateur abstrait **indépendant** du temps dans le domaine du comptage des Changements de valeurs



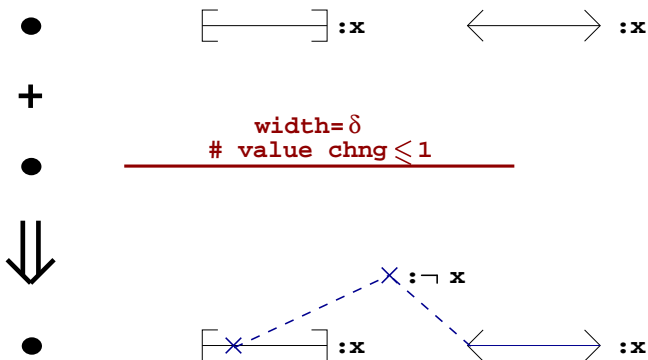
# un opérateur abstrait **dépendant** du temps dans le domaine du comptage des Changements de valeurs



- $[\alpha, \beta]$  paramètre de l'horloge  $C$
- $\vec{\Psi}_{\text{DISCR}_{[\alpha, \beta]}}^{\#}(\text{any input}) \triangleq (1, \alpha)$

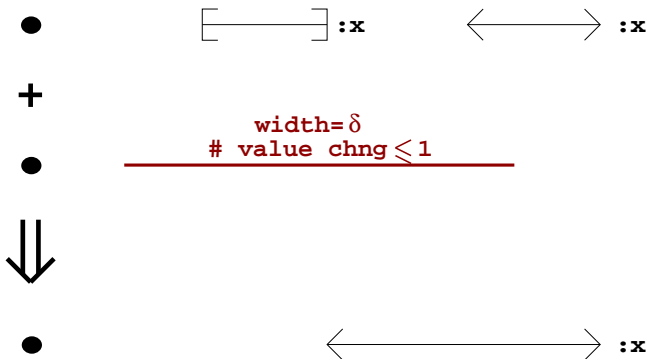
# Produit Réduit

## Contraintes-nombre de changements de valeur

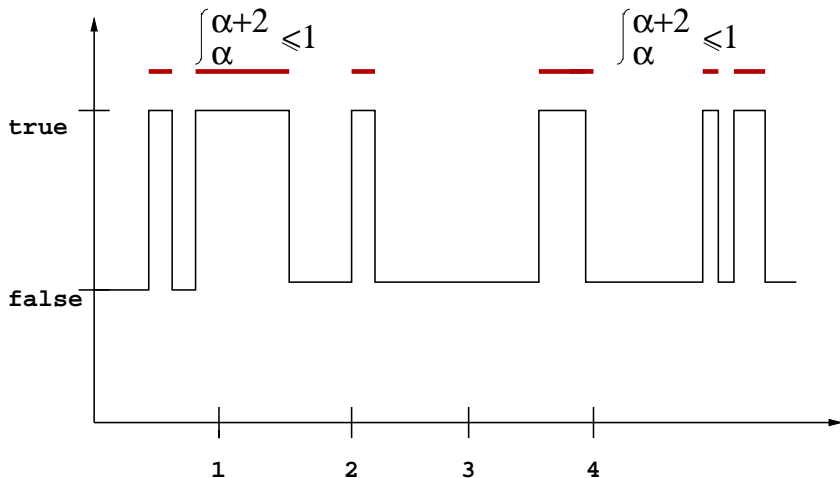


# Produit Réduit

## Contraintes-nombre de changements de valeur



## 3ème domaine abstrait : encadrement des intégrales



- Permet d'exprimer les spécifications quantitatives

# Coopération entre domaines

## encadrements des intégrales et domaine des contraintes

- Supposons à un point de contrôle les informations abstraite suivantes :
  - ▶  $0 \leq \int_x^{x+4} v(t) dt \leq 3.$
  - ▶  $\forall \langle x; x + 1 \rangle : \text{True}$  **et**  $\forall \langle x + 3; x + 4 \rangle : \text{True}$

# Coopération entre domaines

## encadrements des intégrales et domaine des contraintes

- Supposons à un point de contrôle les informations abstraites suivantes :

- ▶  $0 \leq \int_x^{x+4} v(t) dt \leq 3.$

- ▶  $\forall \langle x; x + 1 \rangle : \text{True}$  **et**  $\forall \langle x + 3; x + 4 \rangle : \text{True}$



$$\begin{aligned} \int_x^{x+4} v(t) dt &= \int_x^{x+1} v(t) dt + \int_{x+1}^{x+3} v(t) dt + \int_{x+3}^{x+4} v(t) dt \\ &= 2 + \int_{x+1}^{x+3} v(t) dt \end{aligned}$$

# Coopération entre domaines

## encadrements des intégrales et domaine des contraintes

- Supposons à un point de contrôle les informations abstraites suivantes :

- ▶  $0 \leq \int_x^{x+4} v(t) dt \leq 3.$

- ▶  $\forall \langle x; x + 1 \rangle : \text{True}$  **et**  $\forall \langle x + 3; x + 4 \rangle : \text{True}$



$$\begin{aligned} \int_x^{x+4} v(t) dt &= \int_x^{x+1} v(t) dt + \int_{x+1}^{x+3} v(t) dt + \int_{x+3}^{x+4} v(t) dt \\ &= 2 + \int_{x+1}^{x+3} v(t) dt \end{aligned}$$

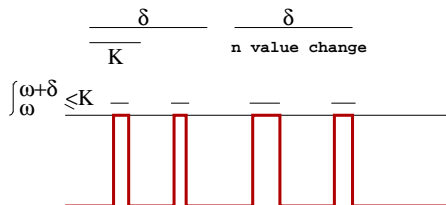
- $0 \leq \int_{x+1}^{x+3} v(t) dt \leq 1.$

# Coopération entre domaines : optimisation des $\Psi^\#$

encadrements des intégrales et comptage des changements de valeur

- $\forall \omega, \int_{\omega}^{\omega+\delta} s_i(t) dt \leq K$

- $(n, \delta)_{\mathcal{N}}$

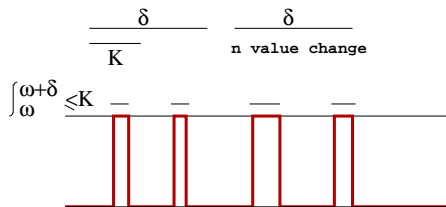
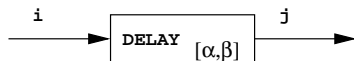


# Coopération entre domaines : optimisation des $\Psi^\#$

encadrements des intégrales et comptage des changements de valeur

- $\forall \omega, \int_{\omega}^{\omega+\delta} s_i(t) dt \leq K$

- $(n, \delta)_{\mathcal{N}}$

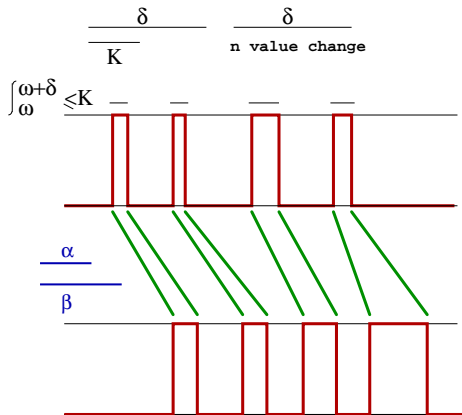
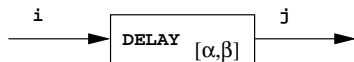


# Coopération entre domaines : optimisation des $\Psi^\#$

encadrements des intégrales et comptage des changements de valeur

- $\forall \omega, \int_{\omega}^{\omega+\delta} s_i(t) dt \leq K$

- $(n, \delta)_{\mathcal{N}}$

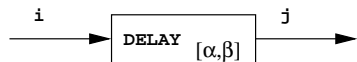


# Coopération entre domaines : optimisation des $\Psi^\#$

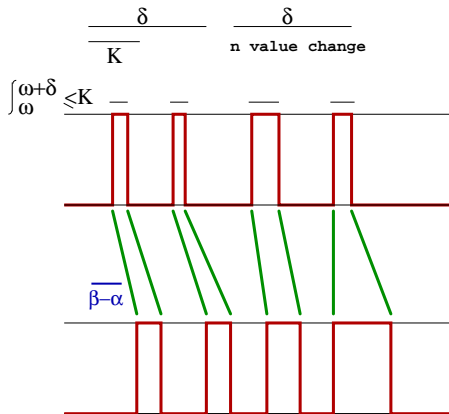
encadrements des intégrales et comptage des changements de valeur

- $\forall \omega, \int_{\omega}^{\omega+\delta} s_i(t) dt \leq K$

- $(n, \delta) \mathcal{N}$



- $\forall \omega, \int_{\omega}^{\omega+\delta-\beta+\alpha} |s_i(t+\alpha) - s_j(t)| dt$

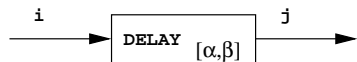


# Coopération entre domaines : optimisation des $\Psi^\#$

encadrements des intégrales et comptage des changements de valeur

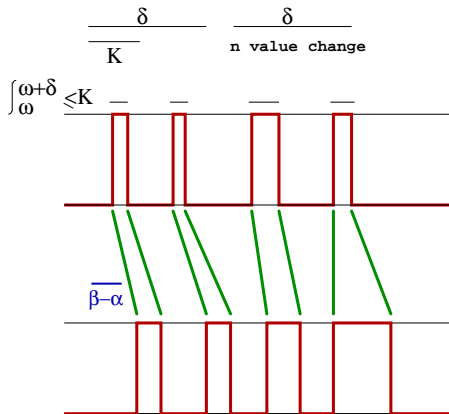
- $\forall \omega, \int_{\omega}^{\omega+\delta} s_i(t) dt \leq K$

- $(n, \delta)_{\mathcal{N}}$



- $\forall \omega, \int_{\omega}^{\omega+\delta-\beta+\alpha} |s_i(t+\alpha) - s_j(t)| dt$

$$\leq n \times (\beta - \alpha)$$

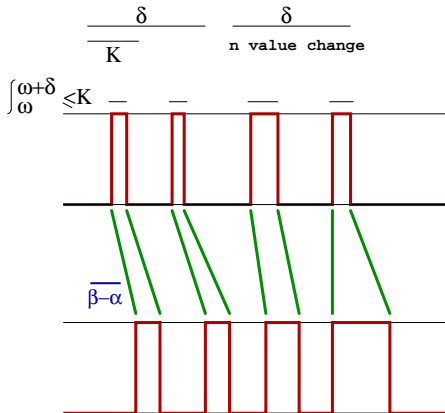
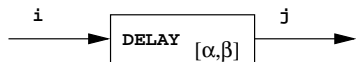


# Coopération entre domaines : optimisation des $\Psi^\#$

encadrements des intégrales et comptage des changements de valeur

- $\forall \omega, \int_{\omega}^{\omega+\delta} s_i(t) dt \leq K$

- $(n, \delta)_{\mathcal{N}}$



$$\int_{\omega}^{\omega+\delta} \leq K$$

- $\forall \omega, \int_{\omega}^{\omega+\delta-\beta+\alpha} |s_i(t+\alpha) - s_j(t)| dt$

$$\leq n \times (\beta - \alpha)$$

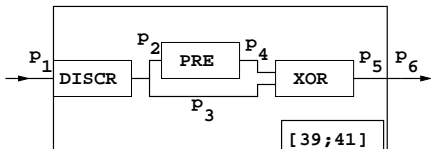
donc si  $\delta - \beta + \alpha \geq 0$  :

$$\forall \omega, \int_{\omega}^{\omega+\delta-\beta+\alpha} s_j(t) dt \leq K + n(\beta - \alpha)$$

- $\vec{\Psi}^\# (\int_{\Delta=\delta} \leq K) \triangleq \int_{\Delta=\delta-\beta+\alpha} \leq K + n(\beta - \alpha)$

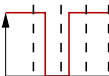
Exemple de **Coopération** entre les 3 domaines abstraits

width=100  
# value chng  $\leq 1$



100

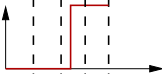
$\leq 2$  value  
changes



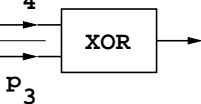
# chng : 0 1 1 0

# chng : 0 0 1 0

$\leq 2$  value  
changes



100



$$95 \leq \int_{\alpha}^{\alpha+100} |s_5(t)| dt \leq 100$$

# Analyse de code

# Analyse de la redondance et du vote

- Plusieurs unités de calcul redondantes
- Périodiquement, un résultat est choisi, parmi les sorties de ces unités (ou une moyenne éventuellement pondérée si ce sont des flottants)
- Comment choisir ? Quand choisir ?

## voter2/2

P. Caspi and R. Salem, *Threshold and Bounded-Delay Voting in Critical Control Systems*, September 2000.

```
voter2/2(x1,x2,nmax) = x
```

```
where (x, n) =  
  if x1=x2  
  then (x1, 0)  
  else if 0 -> pre n < nmax-1  
        then (x0->pre x, 0 -> (pre n) +1  
        else alarm
```

# Exécution du voter

```
##### STEP 1 #####
x1 (integer) ? 1
x2 (integer) ? 1
x = 1, alarm = false
##### STEP 2 #####
x1 (integer) ? 1
x2 (integer) ? 2
x = 1, alarm = false
##### STEP 3 #####
x1 (integer) ? 2
x2 (integer) ? 1
x = 0, alarm = true
##### STEP 4 #####
x1 (integer) ? 2
x2 (integer) ? 1
x = 0, alarm = false
##### STEP 5 #####
x1 (integer) ? 2
x2 (integer) ? 2
x = 2, alarm = false
##### STEP 6 #####
x1 (integer) ? 1
x2 (integer) ? 1
x = 1, alarm = false
```

```
##### STEP 7 #####
x1 (integer) ? 1
x2 (integer) ? 2
x = 1, alarm = false
##### STEP 8 #####
x1 (integer) ? 2
x2 (integer) ? 2
x = 2, alarm = false
```

# Problèmes du voter

- Si les signaux d'entrée sont instables et décalés, `alarm` est toujours *true* !
- Il faut donc des signaux stabilisés en entrée, mais qui continuent à refléter les valeurs des senseurs

## Imposer la stabilisation d'un signal

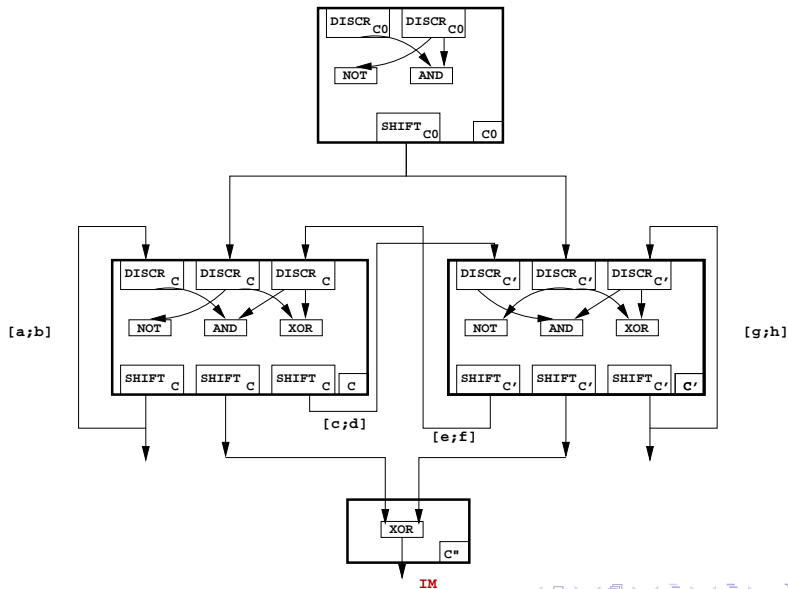
```
node confirm (xp:int) returns (xs: int);
var n, nmax, xo : int;
let
  xo = 314;
  nmax=4;
  xs,n = if (xp=( xo-> pre xp))
    then
      (if ((0 -> pre n) < nmax - 1)
        then
          xo -> pre xs, 0 -> pre (n+1)
        else
          xp, 0 -> pre n
        )
    else
      (xo -> pre xs), 0;
tel;
```

# Imposer la stabilisation d'un signal

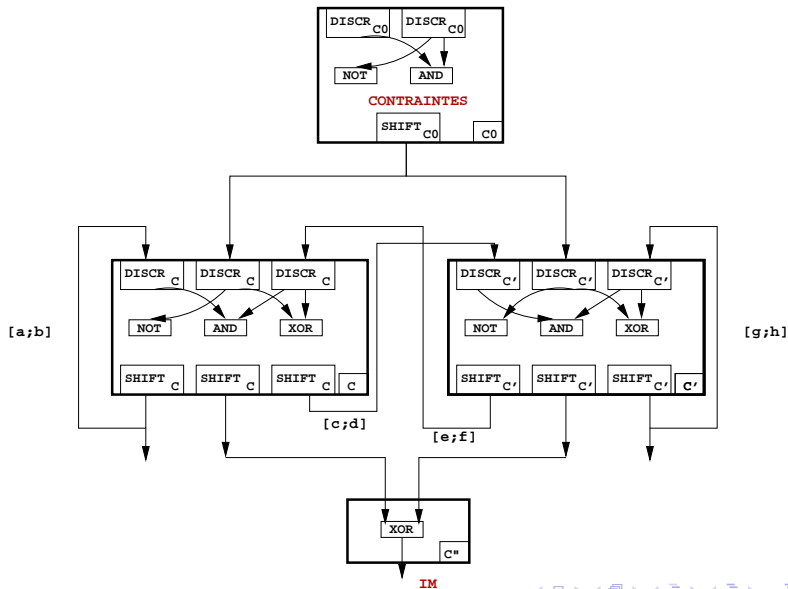
```
##### STEP 1 #####
xp (integer) ? 1
xs = 314
##### STEP 2 #####
xp (integer) ? 2
xs = 314
##### STEP 3 #####
xp (integer) ? 3
xs = 314
##### STEP 4 #####
xp (integer) ? 1
xs = 314
##### STEP 5 #####
xp (integer) ? 1
xs = 314
##### STEP 6 #####
xp (integer) ? 1
xs = 1
##### STEP 7 #####
xp (integer) ? 2
xs = 1
##### STEP 8 #####
xp (integer) ? 2
xs = 1
```

```
##### STEP 9 #####
xp (integer) ? 3
xs = 1
##### STEP 10 #####
xp (integer) ? 3
xs = 1
##### STEP 11 #####
xp (integer) ? 4
xs = 1
##### STEP 12 #####
xp (integer) ? 4
xs = 1
##### STEP 13 #####
xp (integer) ? 4
xs = 4
```

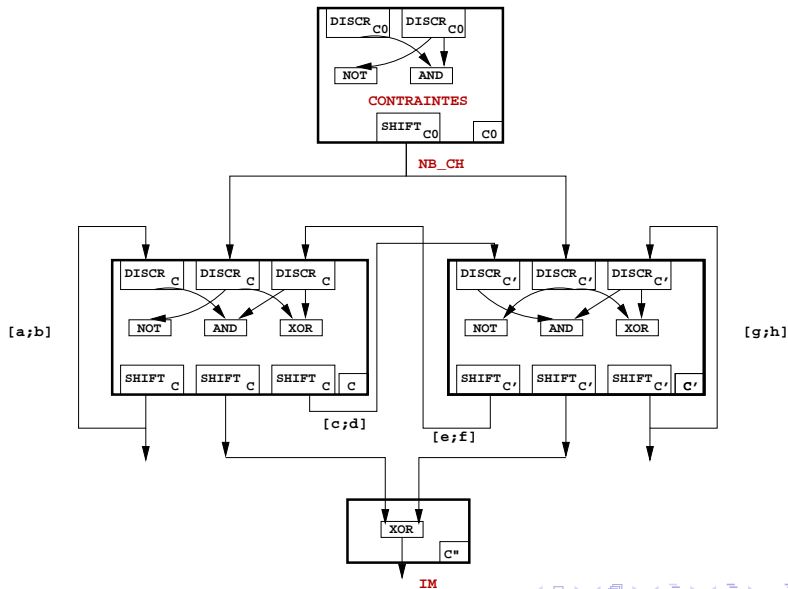
# Propagation efficace de l'information abstraite



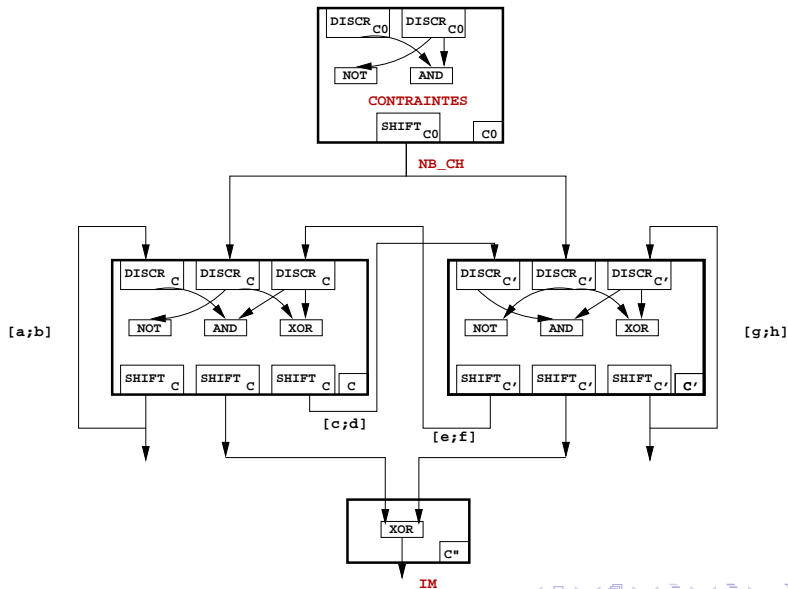
# Propagation efficace de l'information abstraite



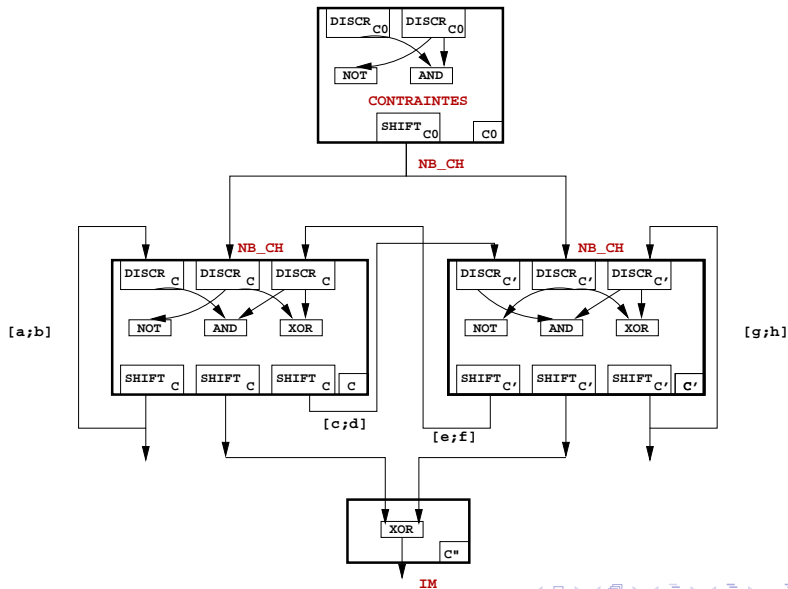
# Propagation efficace de l'information abstraite



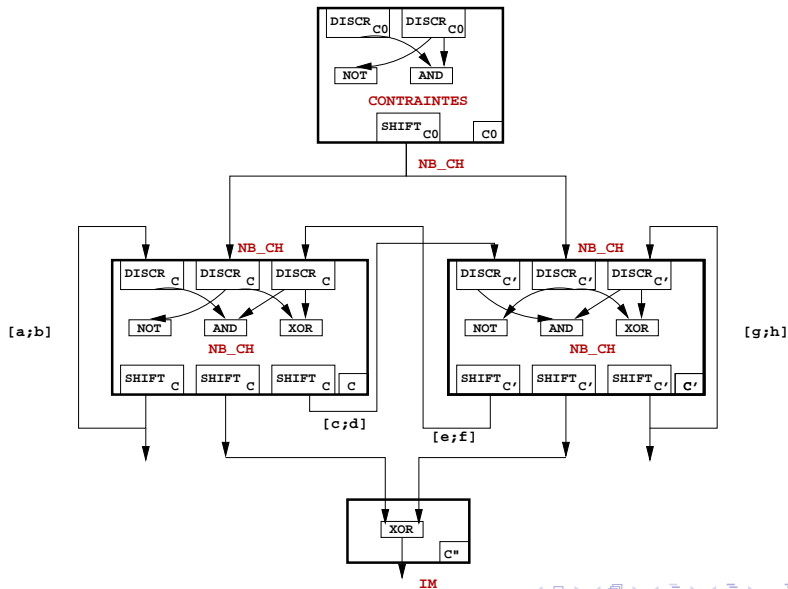
# Propagation efficace de l'information abstraite



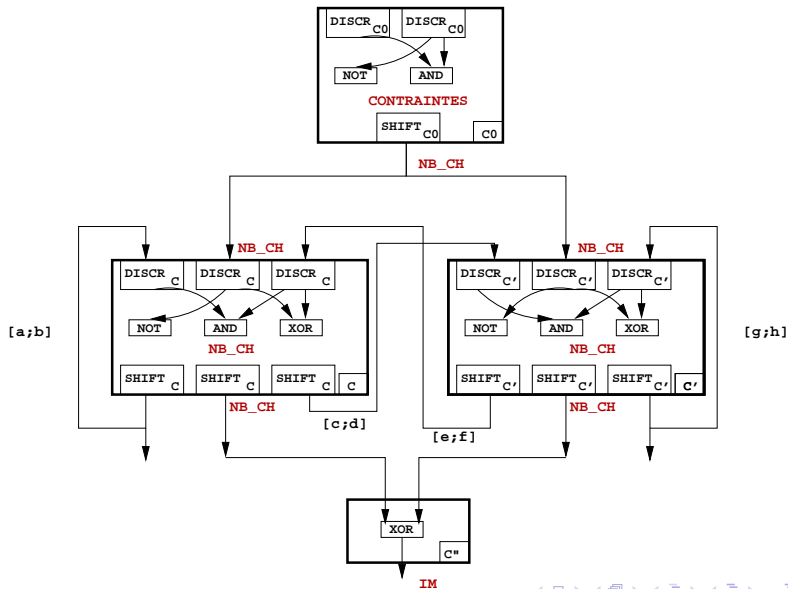
# Propagation efficace de l'information abstraite



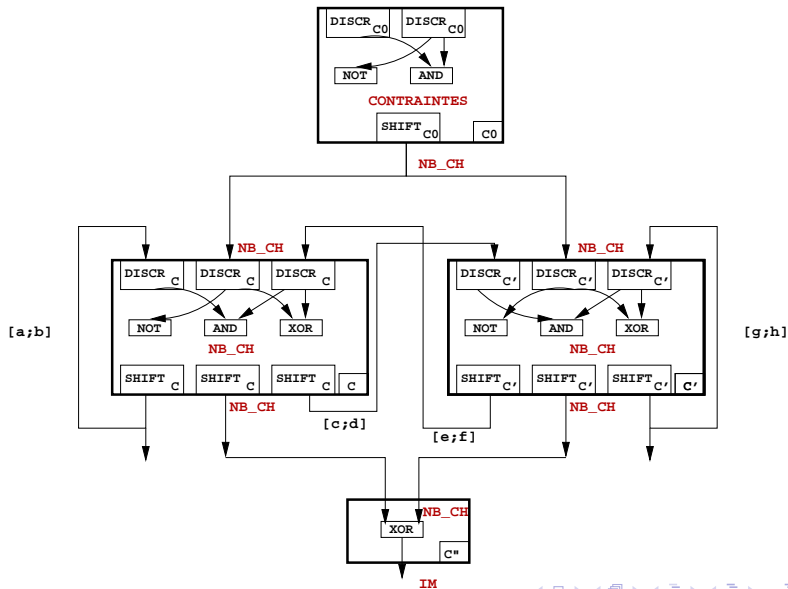
# Propagation efficace de l'information abstraite



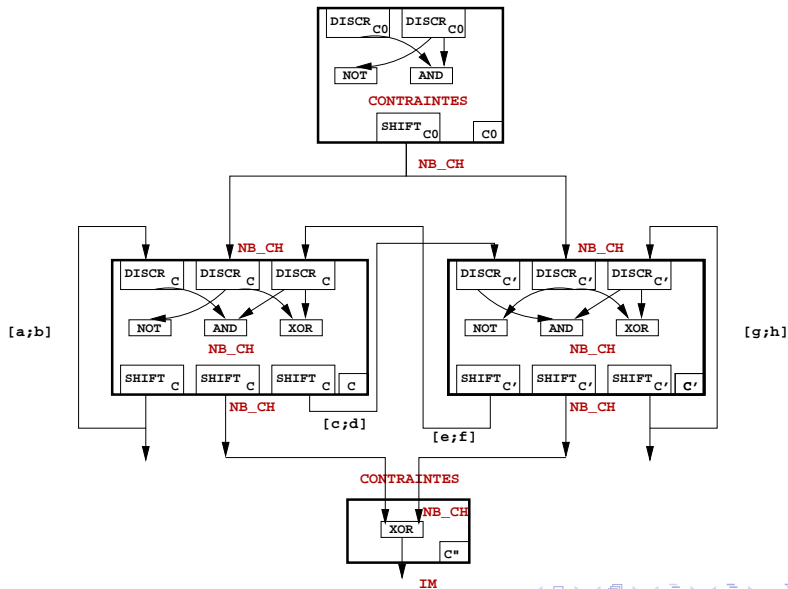
# Propagation efficace de l'information abstraite



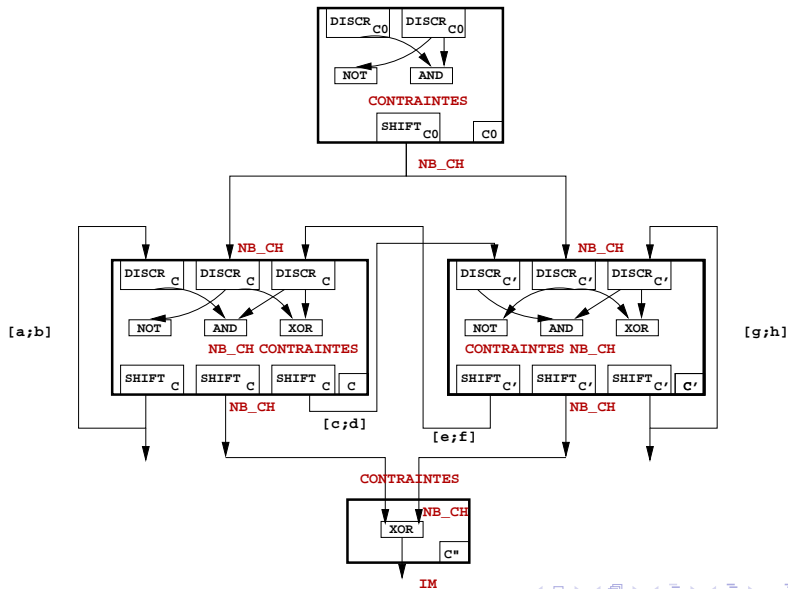
# Propagation efficace de l'information abstraite



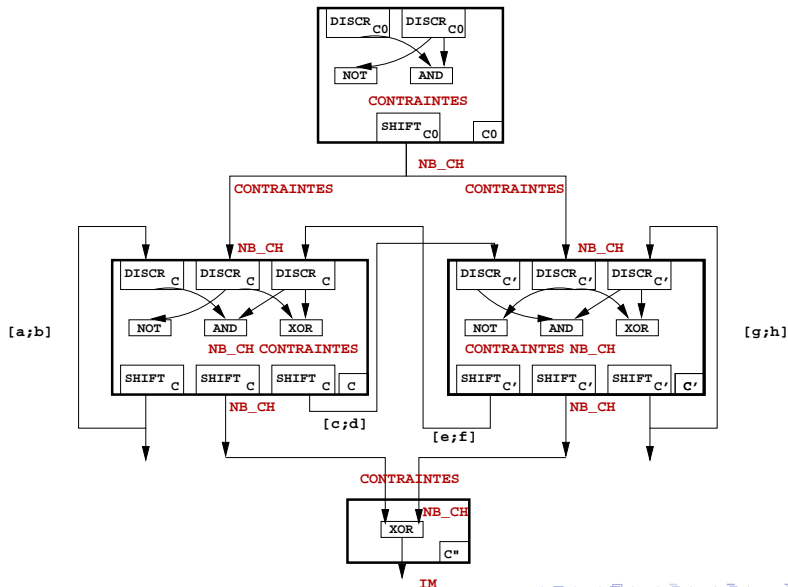
# Propagation efficace de l'information abstraite



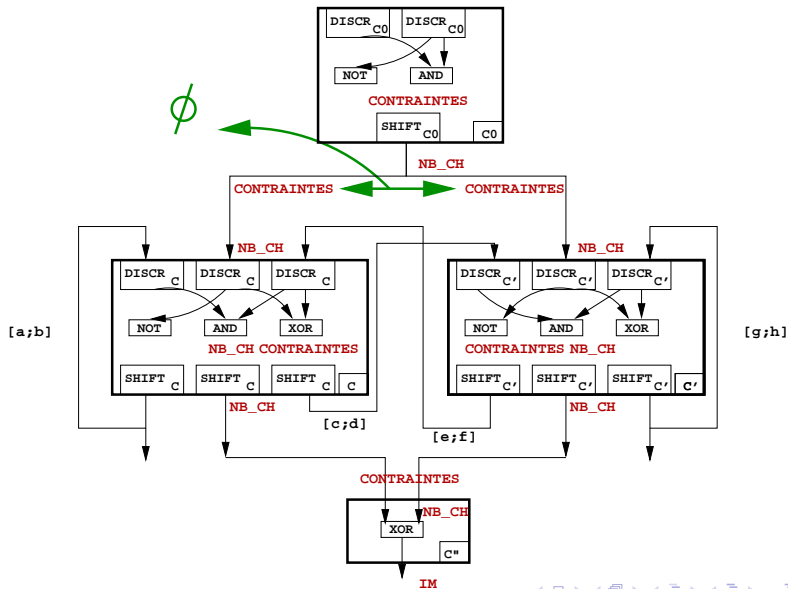
# Propagation efficace de l'information abstraite



# Propagation efficace de l'information abstraite



# Propagation efficace de l'information abstraite



# Résultats

- Sous certaines hypothèses, certaines des techniques proposées par P. Caspi sont validées
- Analyse possible quand divergence de l'horloge reste mesurée
- Stabilité des signaux à l'entrée très importante (les contre-exemples sont souvent des signaux alternants)

# Conclusion

- Modèle réaliste de l'exécution des systèmes synchrones communicant à horloge imparfaite
- La syntaxe permet des annotations quantifiant les imperfections matérielles
- La sémantique est à "temps continu" ( $\neq$  des sémantiques classiques)
- Modularité de l'analyse vis à vis des domaines abstraits