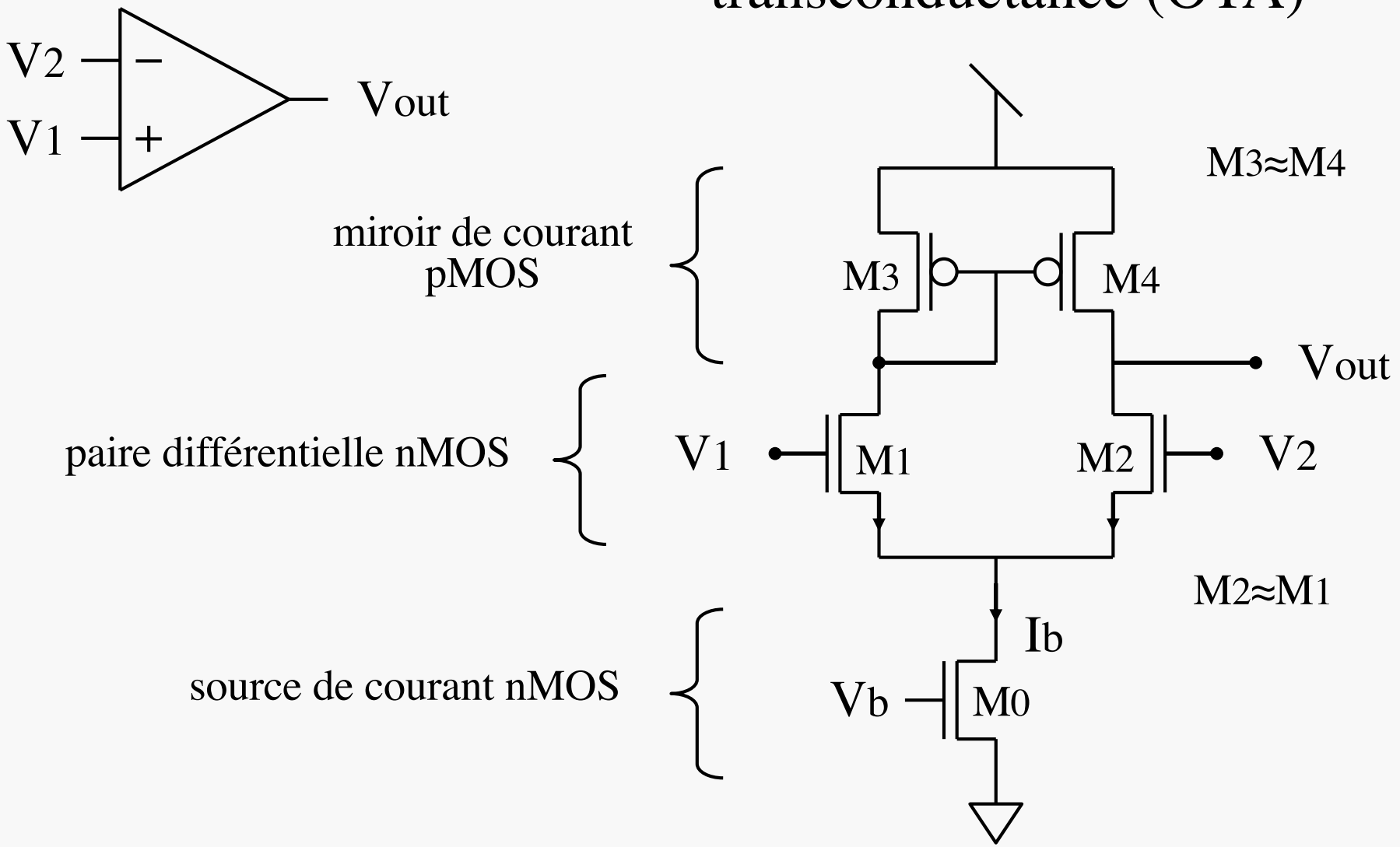
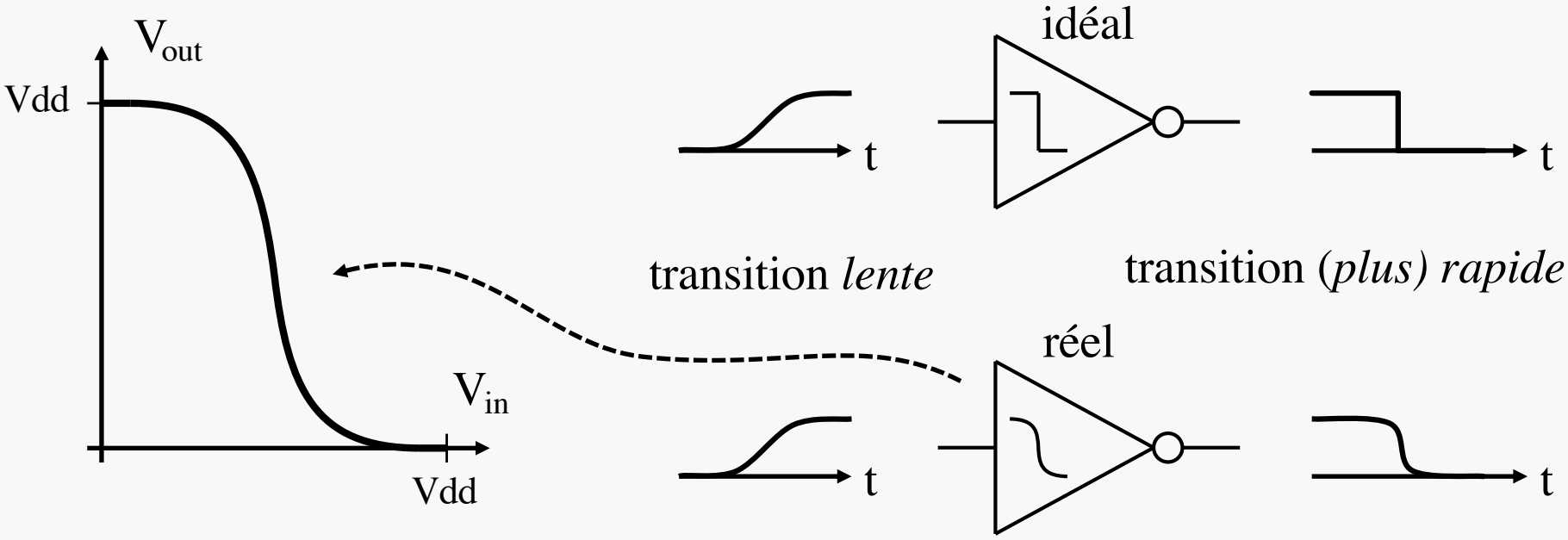
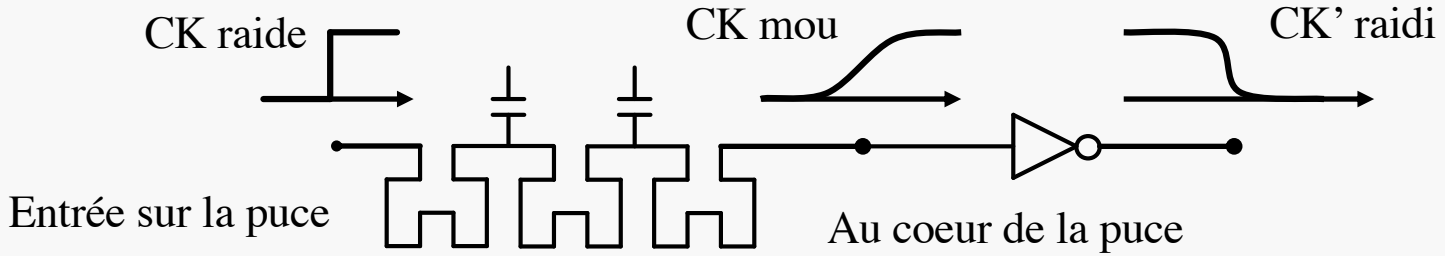
ENCORE UN PEU D'ANALOGIQUE ? ☺

A portée de main : l'amplificateur opérationnel à transconductance (OTA)



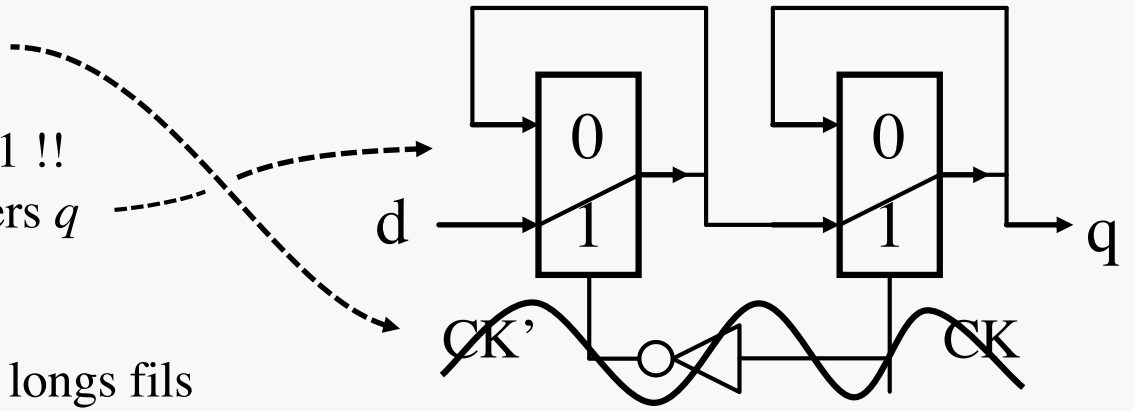
INVERSEUR CMOS : ACCÉLÉRATEUR REDRESSEUR DE TRANSITIONS



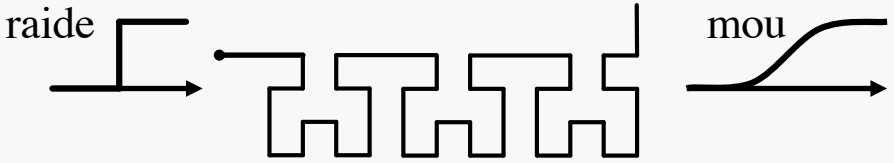


DANGERS INTRA-BASCULE D

CK↑ ⇒ CK'↓
mais pas instantanément
⇒ transitoirement, CK=CK'=1 !!
⇒ brève liaison directe de *d* vers *q*
trop brève pour affecter q



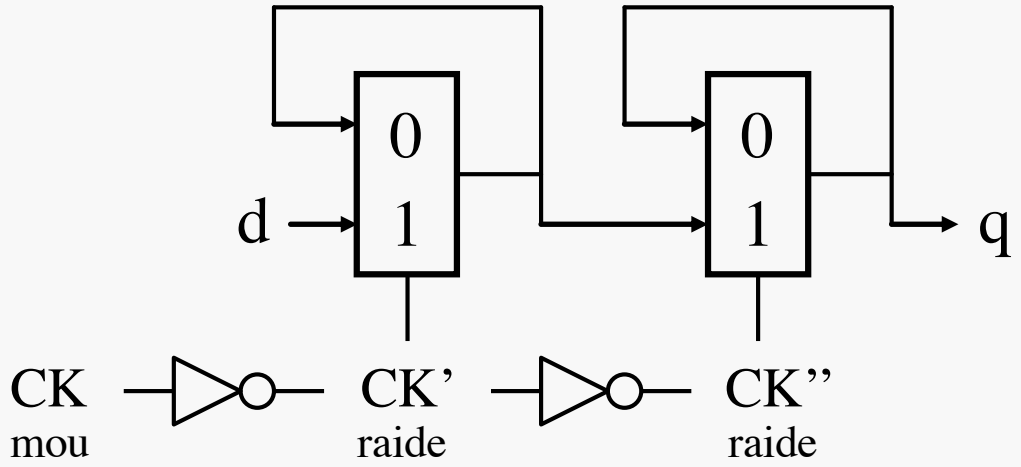
Mais filtrage passe-bas de CK par longs fils de routage résisto-capacitifs → CK mou
⇒ retard/avance de commutation des MUX
⇒ possible liaison plus durable de *d* vers *q*
Situation inacceptable !



Solution :

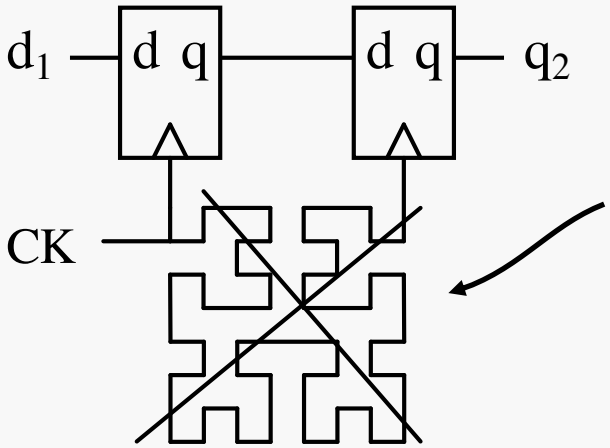
• utiliser l'inverseur CMOS comme *raidisseur* local de fronts lents/mous

• adopter l'organisation ci-contre, qui garantit des fronts raides et proches sur CK' et CK'' (désormais propres à chaque bascule)

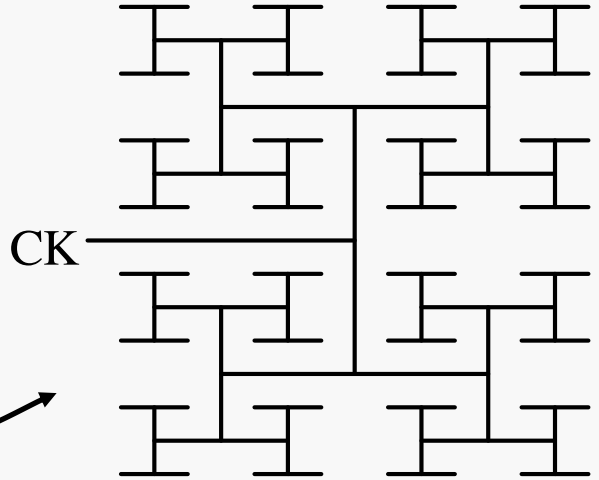


→ 24 transistors pour une bascule D fiable

DANGERS INTER-BASCULES D



Si la deuxième bascule reçoit le top d'horloge significativement plus tard que la première, d_1 peut sauter directement en q_2 !

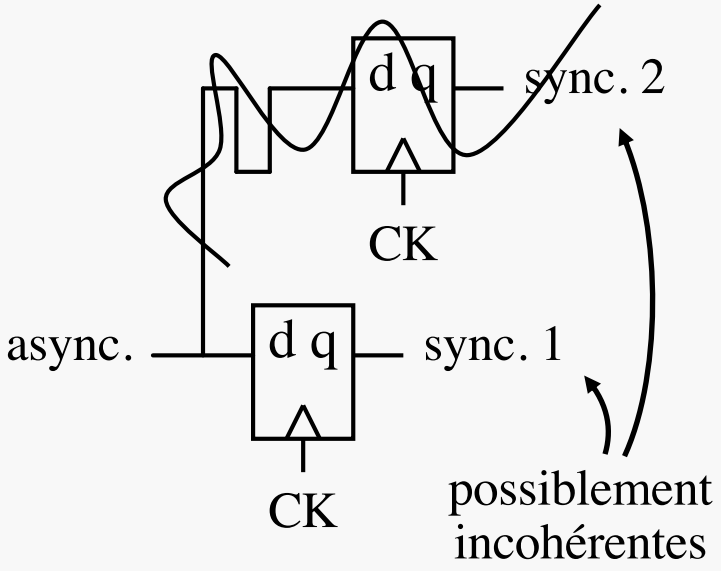


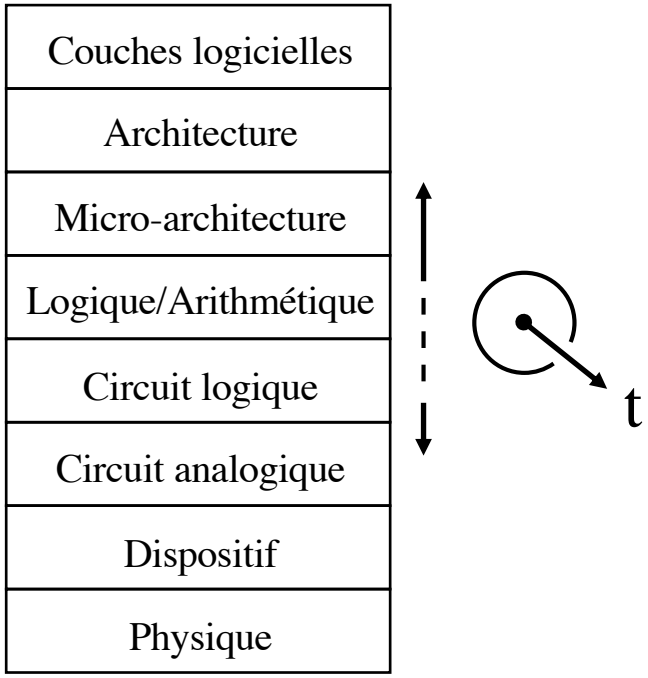
Structure isochrone de distribution de l'horloge à bord d'une puce

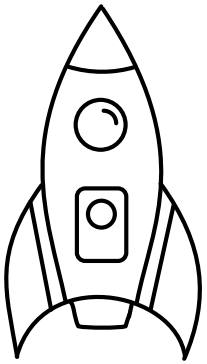
Il arrive qu'une entrée soit purement asynchrone (voire synchrone avec une autre horloge…)

Comment l'exploiter ?

En l'appliquant en entrée d'une – et une seule – bascule D cadencée par CK. La sortie sera une version synchronisée (avec CK), généralement exploitable malgré les aléas d'échantillonnage





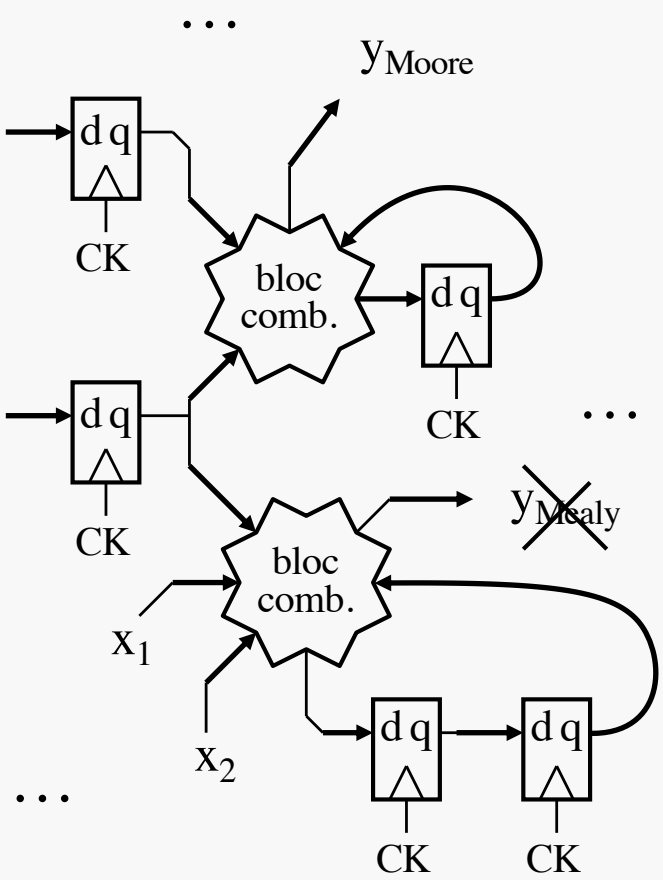


LOGIQUE SÉQUENTIELLE ET SYSTÈMES DYNAMIQUES DISCRETS

ES102 / CM6

CIRCUIT SÉQUENTIEL SYNCHRONE

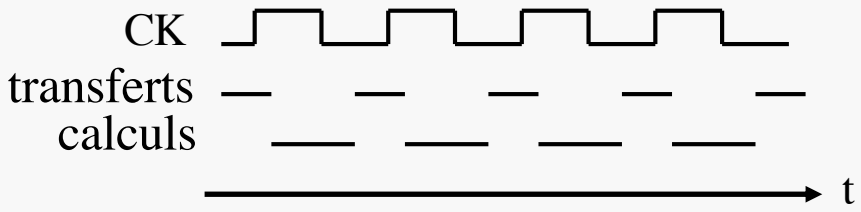
bis CM5



= ensemble de bascules D

- toutes commandées par la même horloge CK
- interconnectées par des blocs combinatoires
- avec des entrées (t.q. x_1) et sorties (t.q. y_{Moore})

- horloge = chef d'orchestre
- toutes les bascules transfèrent simultanément
- temps découpé en une alternance transferts/calculs



- les entrées doivent être des signaux synchrones
(sans oublier les délais combinatoires avant bascules D)

- sortie du circuit = probable entrée d'un autre circuit séquentiel cadencé par CK
  - certifiable synchrone si ne dépend combinatoirement que de sorties de bascules D
  - par précaution, on se limitera à de telles sorties, dites de type *Moore*
  - évitant celles dépendant combinatoirement d'une entrée, dites de type *Mealy*

BITS REGROUPÉS EN VECTEURS

- Soit un circuit séquentiel synchrone comportant *n* bascules D
- Les entrées et sorties de ces bascules sont les d_i et q_i, $1 \leq i \leq n$

→ Soit les *vecteurs*/n-uplets $\mathbf{d}=(d_1, d_2, \cdots, d_n)$ et $\mathbf{q}=(q_1, q_2, \cdots, q_n)$

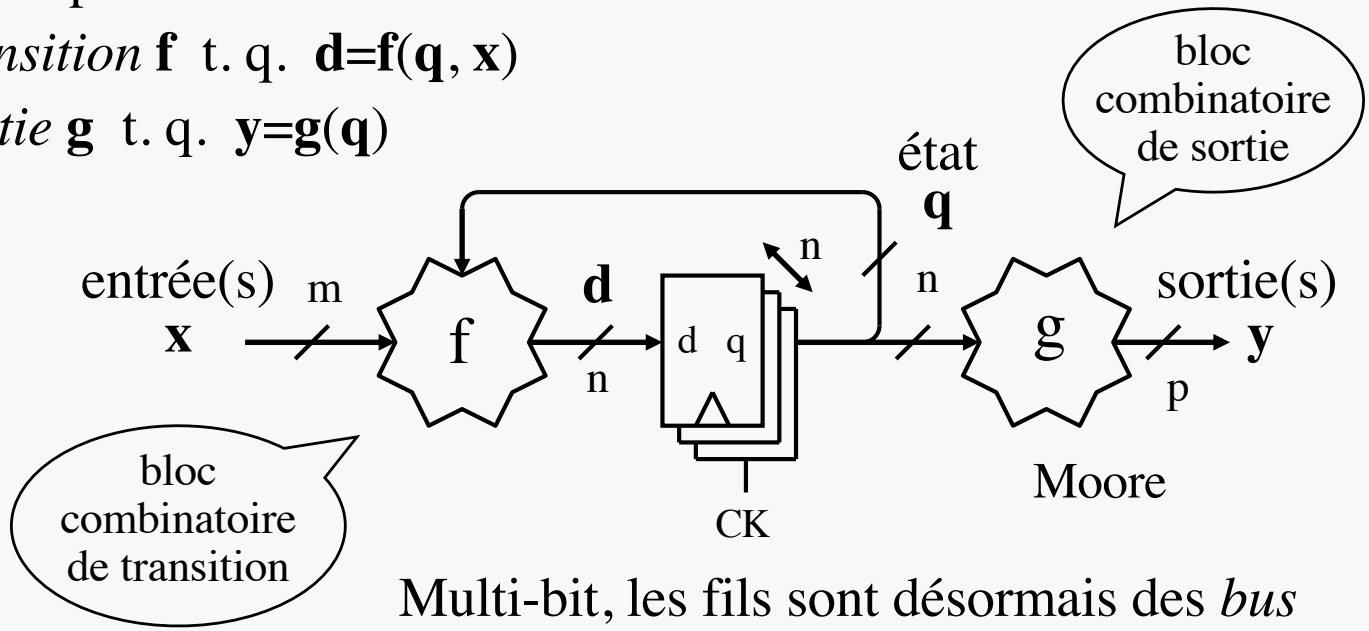
vecteurs en caractères gras

- **q** est appelé *l'état* du circuit (système) séquentiel
  - les q_i sont appelés « bits d'état » (même sens que « variables d'état »)
- Entrées et sorties binaires du circuit séquentiel également mises sous forme de vecteurs : **x** (m-uplet) et **y** (p-uplet)

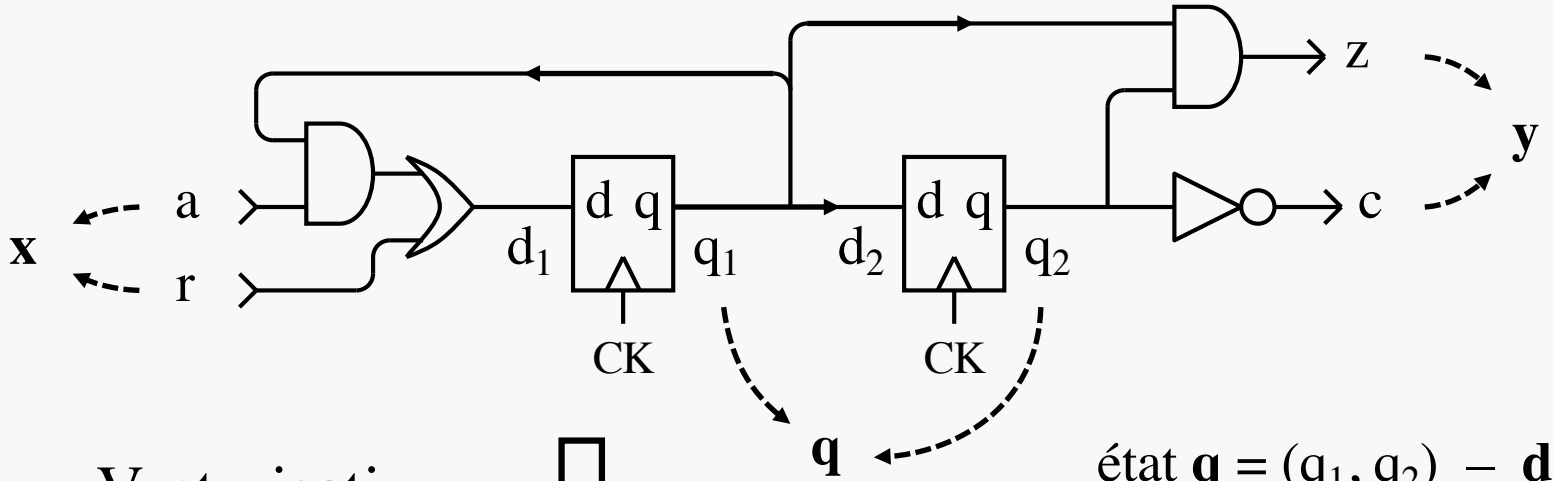
f. b. vectorielle : n ou p-uplet de f. b.

→ Circuit séquentiel décrit par 2 fonctions booléennes vectorielles :

- la *fonction de transition* **f** t. q. **d=f(q, x)**
- la *fonction de sortie* **g** t. q. **y=g(q)**
  - car Moore
  - **y=g(q, x)** si Mealy
- en contexte linéaire (cf. AO102), f et g seraient représentées par des matrices *n*×-

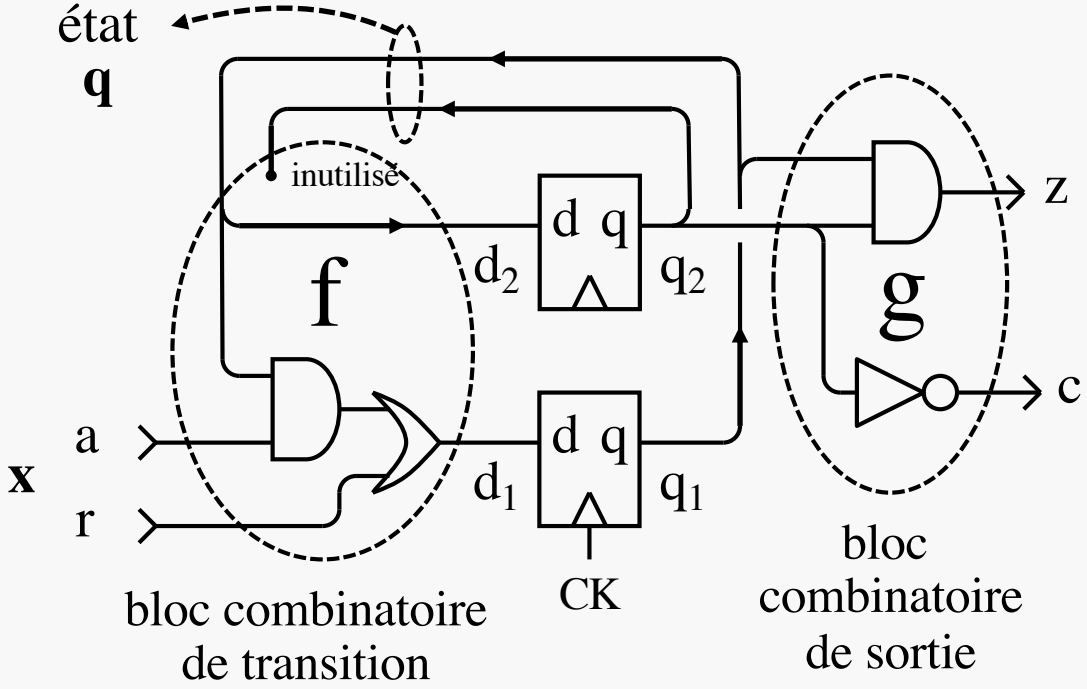


Multi-bit, les fils sont désormais des *bus*



« Vectorisation »

état **q** = (q_1, q_2) – **d** = (d_1, d_2)
signaux d'entrée : a et r → **x** = (a, r)
signaux de sortie : z et c → **y** = (z, c)



EXEMPLE
avec n=m=p=2

d = (d_1, d_2) = ($r+aq_1$, q_1) = f(**q**, **x**)

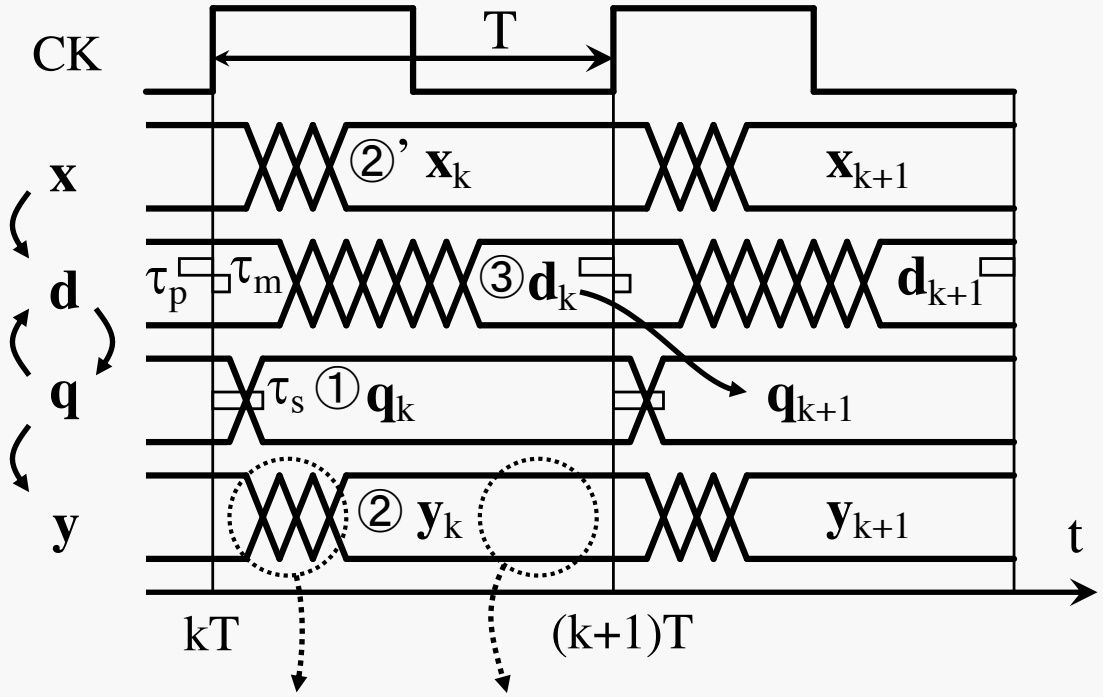
y = (z, c) = (q_1q_2, q_2') = g(**q**)

DERNIER REGARD EN TEMPS CONTINU : DÉROULEMENT DE LA PÉRIODE [kT, (k+1)T]

avec top à chaque extrémité

k : indice temporel

Toutes grandeurs vectorielles
(pas d'indice pour les composantes ici)



Certains bits de **y** peuvent changer

Aucun bit ne change

Notations pour chronogramme de bus

① un petit délai τ_s après le top d'horloge, nouvelle valeur $\mathbf{q}_k$ en sortie des bascules D

② calcul combinatoire de $g(\mathbf{q}_k)$, qui fournit bientôt la sortie $\mathbf{y}_k$

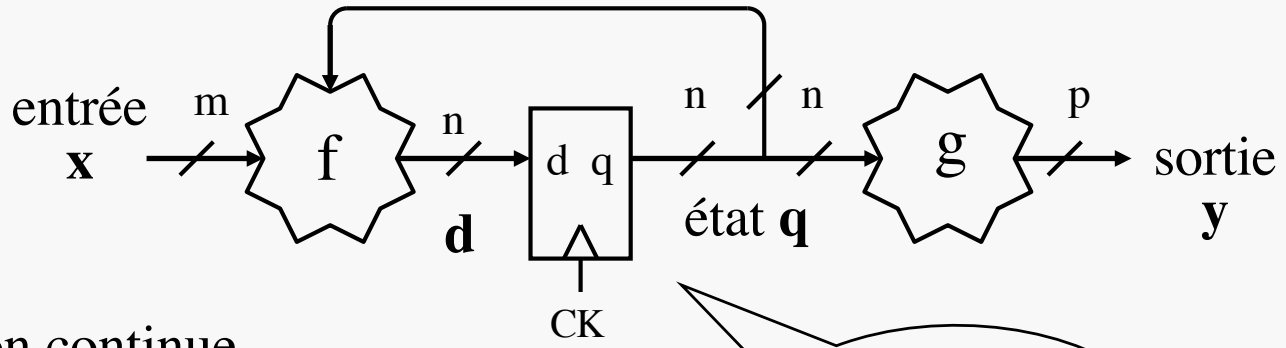
②' parallèlement, l'entrée s'établit, à la valeur $\mathbf{x}_k$ (provenant typiquement de sorties d'autres circuits séquentiels cadencés par CK également, d'où le parallélisme avec ②)

③ $\mathbf{q}_k$ et $\mathbf{x}_k$ étant établis, calcul combinatoire de $f(\mathbf{q}_k, \mathbf{x}_k)$, devant fournir $\mathbf{d}_k$ au plus tard à $(k+1)T-\tau_p$

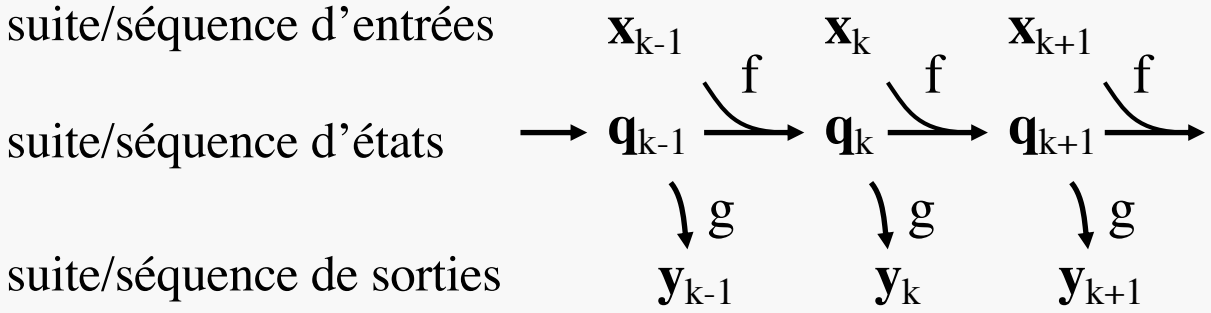
Au prochain top d'horloge, $\mathbf{d}_k$ sera transféré en sortie des bascules D, pour y devenir $\mathbf{q}_{k+1}$

① …

ABSTRACTION EN TEMPS DISCRET



- Abandon de la vision continue
- **q**, **x** et **y** désormais considérées comme de simples suites (temporelles) de vecteurs binaires
  - avec $\forall k \in Z$, $\mathbf{q}_{k+1} = f(\mathbf{q}_k, \mathbf{x}_k)$ et $\mathbf{y}_k = g(\mathbf{q}_k)$
  - **q** : sorte de suite récurrente



- Relation de cause à effet entre séquences **x** et **y** = *comportement* du circuit

SYSTÈME DYNAMIQUE DISCRET $\rightarrow q^+$

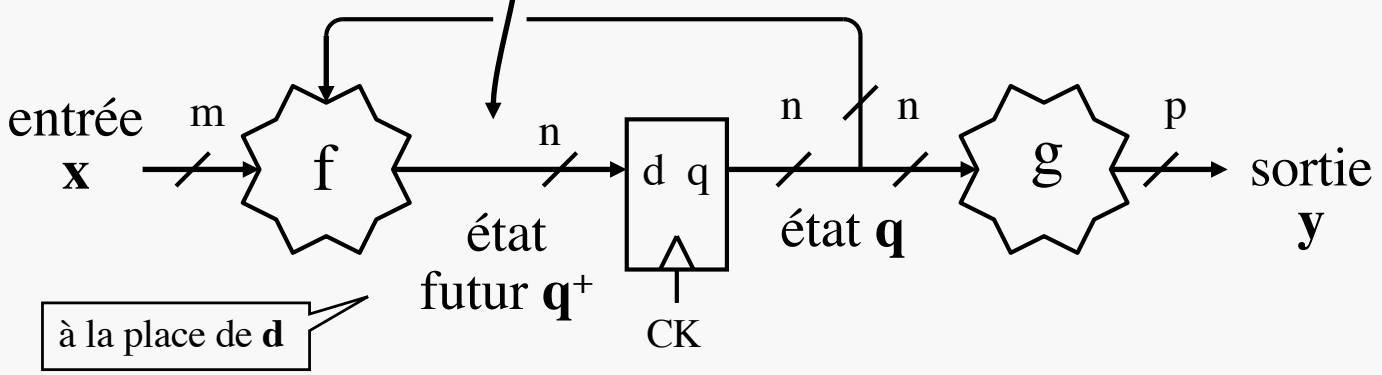
- suite $(\mathbf{q}_k)$ = fonction $\mathbf{q}$ de N vers $\mathbb{B}^n$: $(\forall k \in N)\ \mathbf{q}(k) = \mathbf{q}_k$
- Soit $\mathbf{q}^+$ la suite « successeur de $\mathbf{q}$ » : $(\forall k \in N)\ \mathbf{q}^+(k) = \mathbf{q}_{k+1}$
- Or, $(\forall k \in N)\ \mathbf{q}_{k+1} = f(\mathbf{q}_k, \mathbf{x}_k)$ et $\mathbf{y}_k = f(\mathbf{q}_k)$
- D'où $\mathbf{q}^+ = f(\mathbf{q}, \mathbf{x})$ et $\mathbf{y} = g(\mathbf{q})$
  loi d'évolution (ou de transition) — loi de sortie
  - équations sur des suites temporelles
  - *représentation d'état* d'un système
  - interprétable à l'instant (au *pas*) courant :
    $\mathbf{q}^+$ est le prochain état, successeur de $\mathbf{q}$ sous l'entrée $\mathbf{x}$,
    - présent en entrée de bascule(s) lorsque la période courante est assez avancée…

    alias *l'état futur* de q

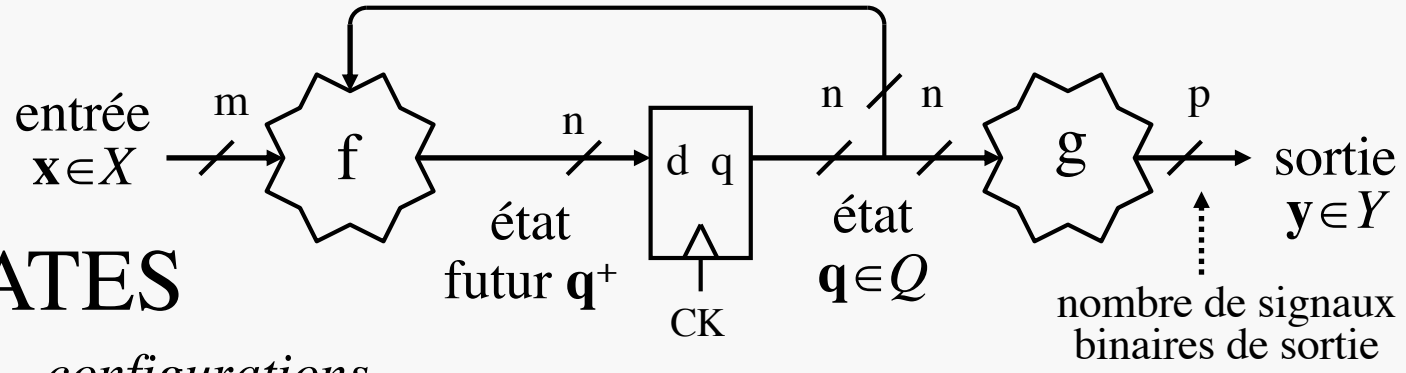
Rôle analogue à celui de $\dot{q} = f(q, x)$ et $y = g(q)$ pour modéliser un système dynamique continu en temps et en valeur des variables

mais objectifs (et notations) différents de ceux de l'Automatique

résurgence du continu

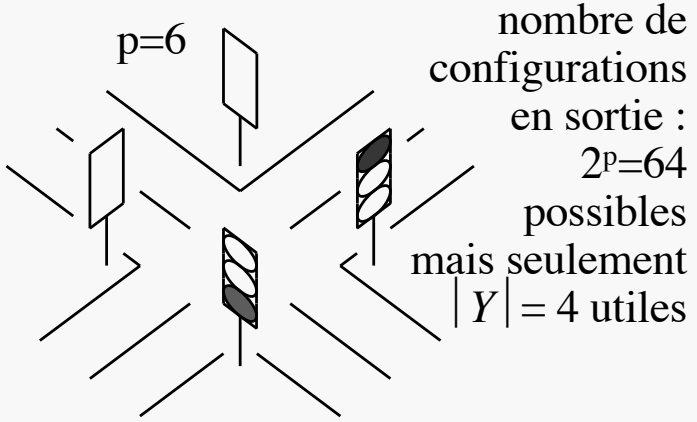


AUTOMATES



configurations

- X, Q et Y ensembles des *valeurs* possibles de **x**, **q** et **y**
  - a priori, $X=\mathbb{B}^m$, $Q=\mathbb{B}^n$ et $Y=\mathbb{B}^p$
  - mais, souvent, ∃ états ou sorties sans objet
  → *on restreint Q et Y à leurs seuls éléments utiles*



- Ignorant leur réalité numérique, on se permet aussi de désigner les états ($\in Q$) par des *symboles* :

  des mots porteurs de sens (« marche », « arrêt ») ou de simples lettres (A, B, C, ...)

  indispensable pour débuter la synthèse d'un circuit séquentiel, partant de son comportement

- fonction de transition f : $Q\times X \rightarrow Q$
- fonction de sortie g : $Q \rightarrow Y$

- Le quintuplet (Q, X, f, Y, g) constitue un *modèle mathématique comportemental*, appelé *automate*, du circuit séquentiel synchrone
  - en anglais : *Finite State Machine* (machine à nombre fini d'états) → FSM
  - utile aussi en informatique théorique, mais en plus spécifique (notion d'état initial/final)

DIAGRAMME D'ÉTAT

Représentation graphique de f et g

$f : Q \times X \to Q$ est représentable par un *graphe* orienté à nœuds et arêtes valués

- nœud $\leftrightarrow$ élément de Q = état (symbolique)
  placé dans un cercle
- arête orientée $\leftrightarrow$ *sous-ensemble* de X
  arc, flèche, *transition*
  en fait son indicatrice $1_{\{\cdots\}}$
  = condition sur les signaux d'entrées, sous laquelle l'arc est emprunté
  (*au top d'horloge*)

Un arc non valué représente une transition inconditionnelle (toujours vraie)

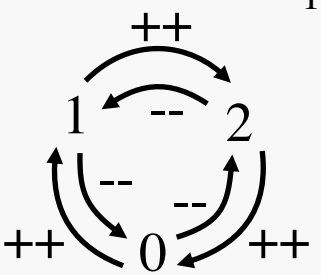
$g : Q \to Y$ (Moore)

chaque état E (encerclé) est présenté avec sa configuration en sortie g(E)

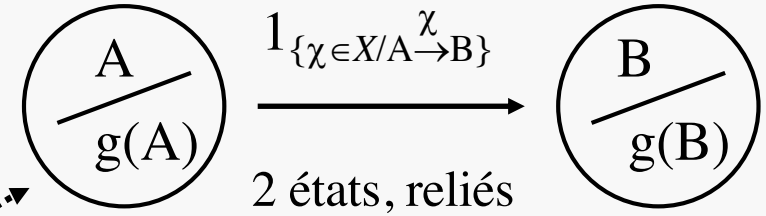
exemple

$Q=\mathbb{Z}/3\mathbb{Z}$

$X=\{--,++\}$



sous-ensemble plutôt qu'élément, car plusieurs éléments de X = { config. en entrée } peuvent faire transiter entre 2 états donnés



2 états, reliés par une transition conditionnelle

brique de base d'un diagramme d'état

ÉTUDE D'UN EXEMPLE (ANALYSE)

comportement à
horizon lointain
(global en temps)

diagramme
d'états
comportement
immédiat
(local en temps)

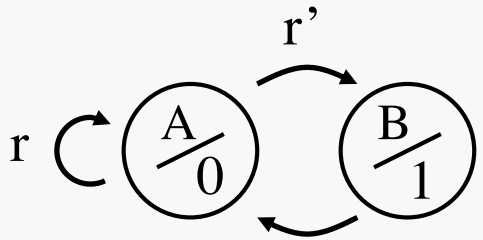
symboliques

numériques

lois d'évolution f
et de sortie g

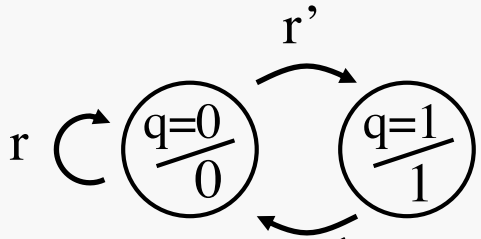
circuit séquentiel
synchrone

Comportement (relation entrée-sortie) :
la sortie est remise à 0 par r=1,
sinon elle oscille entre 0 et 1.



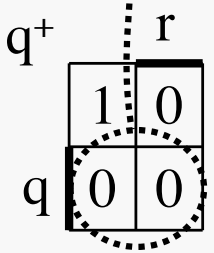
états
symboliques

états numériques
(et même binaires)

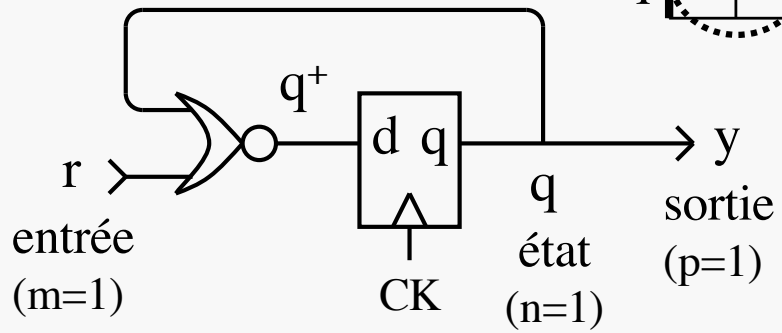


loi
d'évolution
$q^+ = q'r'$
$(q_{k+1} = q_k' r_k')$

loi de
sortie
$y = q$
$(y_k = q_k)$



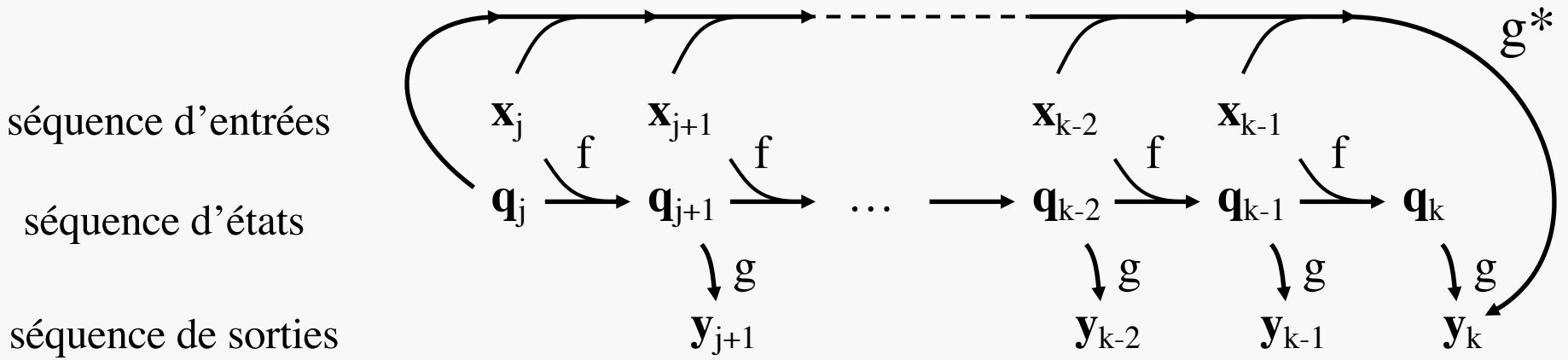
$\forall \chi \in X$ ...
transition
incondi-
tionnelle



COMPORTEMENT À HORIZON LOINTAIN

- Partant d'un état q_j à l'instant *j* pour le système, quelle sera sa sortie y_k après avoir subi une séquence d'entrées $\xi=(x_j, x_{j+1}, \ldots, x_{k-2}, x_{k-1})$?

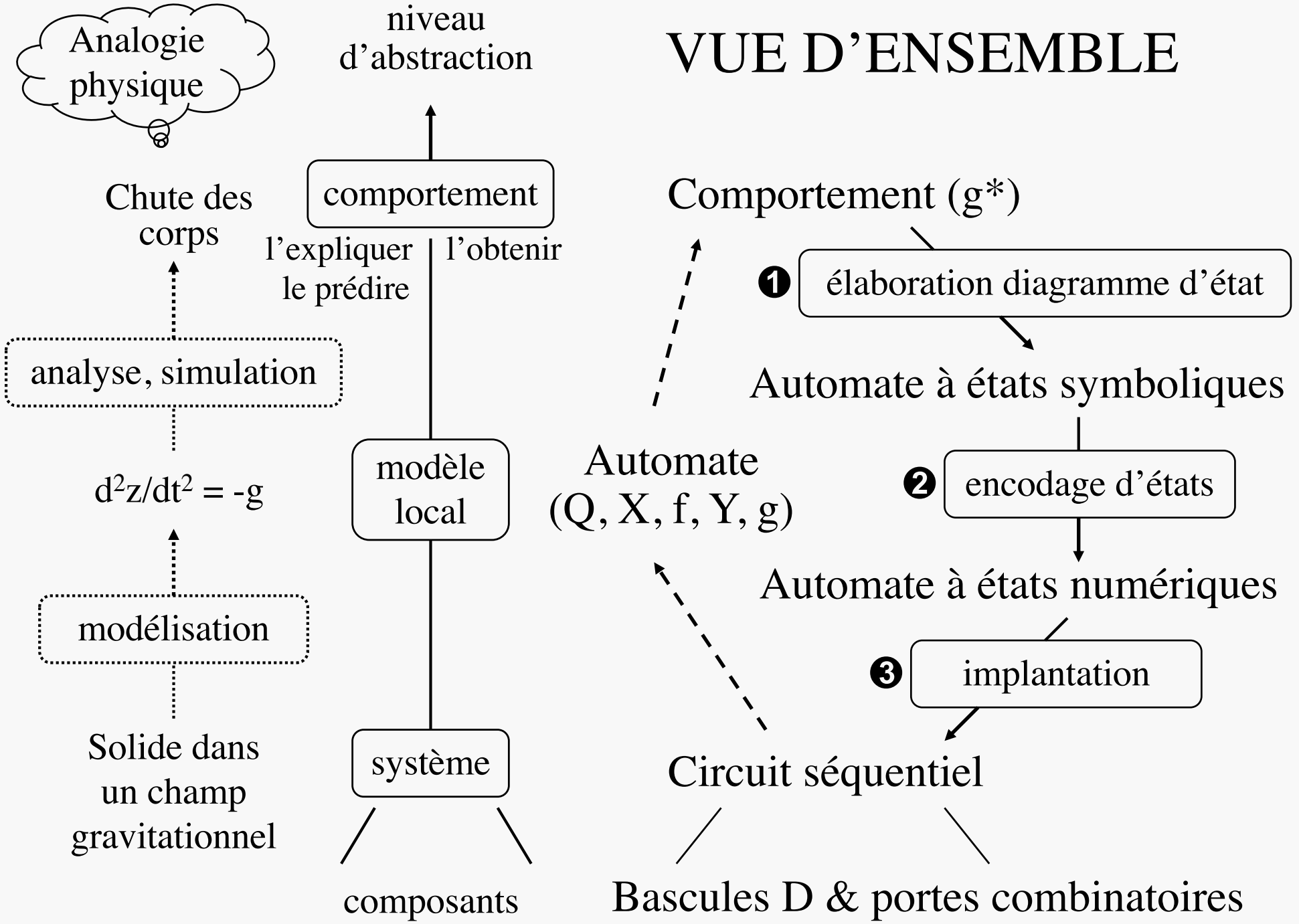
  → une certaine valeur $g^*(q_j, \xi)$
  où g*, dite *fonction de sortie généralisée*,
  résulte de compositions multiples de f, puis g



- g* décrit/*spécifie* le comportement global du système
- en pratique, g* est donc connue *avant* f et g !
  - mais souvent seulement sur un état *naturel* q_{nat} du système

typiquement l'état après réinitialisation

VUE D'ENSEMBLE



SYNTHÈSE DE CIRCUITS SÉQUENTIELS

Spécifications comportementales

⇩

❶ *Elaborer un diagramme d'état*

entrées X - sorties Y

états naturels, t.q. q_{init}

spécifications t.q. $g^*(q_{init}, \cdot)$

Mobiliser les états nécessaires au comportement voulu

Trouver Q, f et g répondant aux spécifications

⇩

❷ *Encoder les états* — changement de variables

q symbolique

⇩

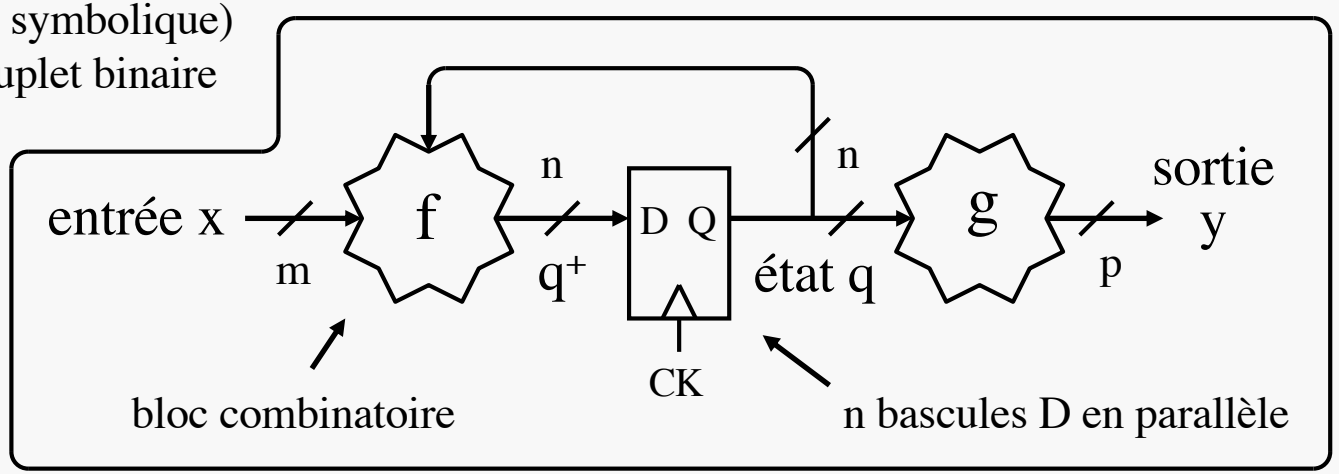
q numérique = γ(q symbolique)
n-uplet binaire

$|Q|$ désormais connu

Choisir $n \geq \log_2(|Q|)$ et γ injectif : $Q \rightarrow \mathbb{B}^n$

⇩

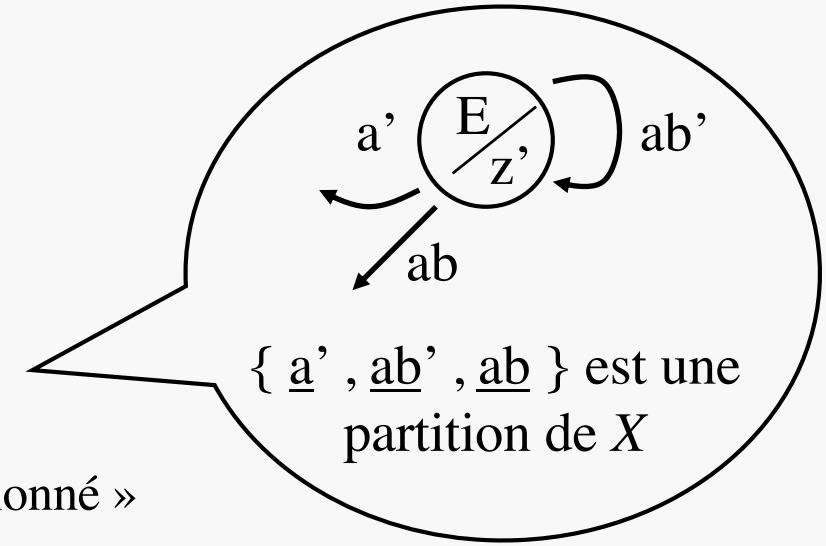
❸ *Implanter*



→ objet de plusieurs PC

❶ ÉLABORER UN DIAGRAMME D'ÉTAT

- Tâche souvent délicate :
  - spécifications initiales souvent en langage naturel
    - souvent incomplètes, et même incohérentes
  - pas de méthodologie systématique
- Propriété fondamentale :
  - pour chaque état et chaque valeur d'entrée(s), un et un seul successeur
    - ⇒ les transitions sortant d'un état doivent toujours partitionner *X*
    - on dira alors que « le D.E. est bien conditionné »



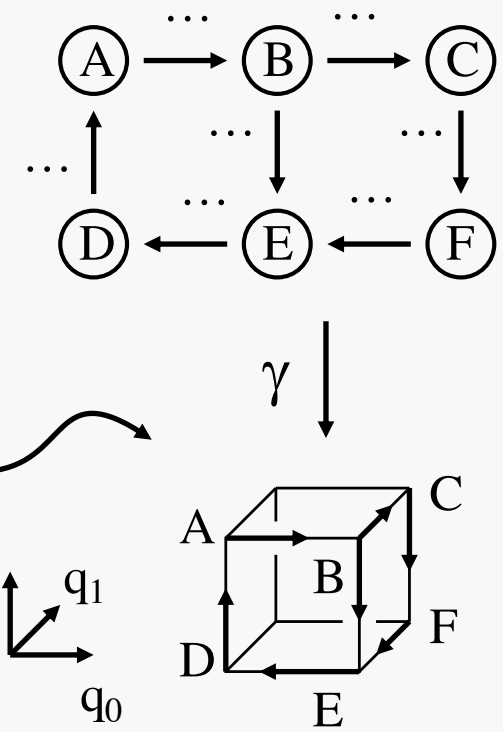
- Représentation :
  - sous forme graphique : diagramme d'état
  - sous forme de programme :
    - langages de description matérielle (HDL), langages dits synchrones
  - souvent hiérarchique/modulaire : interaction entre plusieurs automates

❷ ENCODER LES ÉTATS

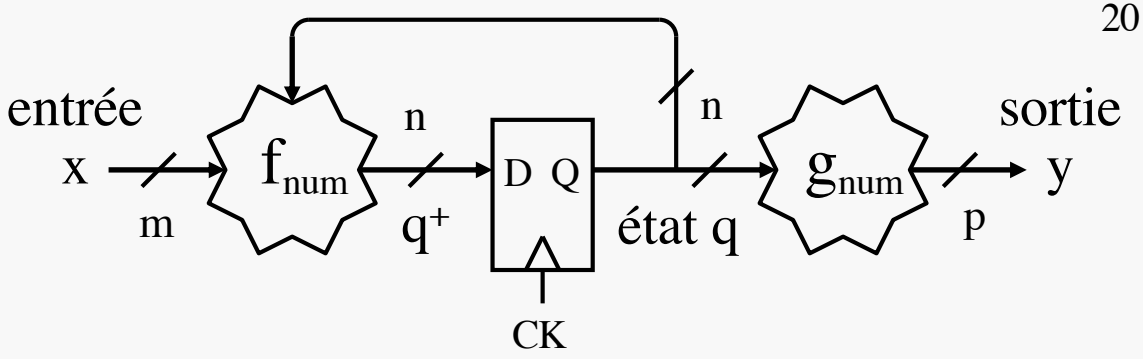
cardinal de Q

- Choisir $n \geq \log_2(|Q|)$, puis un codage injectif $\gamma : Q \rightarrow \mathbb{B}^n$
  - c'est un changement de variable : des états symboliques aux bits q_i

  → mobilise *n* bascules D pour porter les *n* variables d'état q_i
- Stratégies d'encodage usuelles :
  - « adjacence » : usuellement avec $n = \lceil \log_2(|Q|) \rceil$
    - codes proches pour états successeurs l'un de l'autre
      - car tend à simplifier la fonction de transition f
    - idéalement : un seul bit de *différence*
      - exige que les transitions suivent les arêtes du n-cube
      - souvent impossible topologiquement, sauf coup de chance
      - à défaut, on veille à minimiser les *différences*
  - « one-hot » : $n = |Q|$
    - pour chaque état, un des q_i vaut 1, les autres 0
    - coûteux en bascules D (une par état), mais logique simplifiée
      - rapide, bien adapté aux FPGA du commerce
  - codage optimal : problème algorithmique (très) difficile



❸ IMPLANTER



- Diagramme d'état (❶) et encodage γ des états (❷) déterminent numériquement les fonctions f et g :
  - $f_{num} = \gamma \circ f_{symb} \circ \gamma^{-1}$ & $g_{num} = g_{symb} \circ \gamma^{-1}$

  effets classiques d'un changement de variables
  - reste à les implanter sous forme de logique combinatoire en se connectant en entrée et en sortie des bascules D
- Exemple générique avec états, entrées et sorties chacun sur 2 bits :

  $\mathbf{q} = (q_1, q_0)$, $\mathbf{x} = (x_1, x_0)$, $\mathbf{y} = (y_1, y_0)$ état futur : $\mathbf{q}^+ = (q_1^+, q_0^+)$

  → fonctions booléennes à exprimer et implanter :
  - Logique de transition $\mathbf{q}^+ = f(\mathbf{q},\mathbf{x})$

    $$\begin{cases} q_0^+ = f_0(q_1, q_0, x_1, x_0) \\ q_1^+ = f_1(q_1, q_0, x_1, x_0) \end{cases}$$

    f_{num}

    Avec des entrées sur *m* bits et des états sur *n* bits, une fonction de transition f s'exprime par *n* fonctions booléennes de *m*+*n* variables…
  - Logique de sortie $\mathbf{y} = g(\mathbf{q})$

    $$\begin{cases} y_1 = g_1(q_1, q_0) \\ y_0 = g_0(q_1, q_0) \end{cases}$$

    g_{num}