
TEST INTEGRE

BIST : “Built-In Self-Test”

Mounir BENABDENBI
Mounir.Benabdenbi@lip6.fr

Laboratoire d’Informatique de Paris 6 (LIP6)



Pourquoi du test intégré ?

- prix élevé et toujours croissant des testeurs,
- génération des séquences de test longue et difficile
- temps d 'application des séquences élevé
- inefficacité grandissante des testeurs

Test intégré

Un passage obligé

Carte de route SIA (aspect technologie)

| | 1995 | 2001 | 2010 |
|--|-------------|-------------|-------------|
| Technologie (micron) | 0,35 | 0,18 | 0,07 |
| Nombre de transistors (millions) | 1 | 10 | 100 |
| Fréquence d'horloge (MHz) | 300 | 600 | 1100 |
| Nombre d'E/S | 900 | 2000 | 4800 |
| Taille de la puce (mm²) | 450 | 750 | 1400 |
| Pourcentage de circuits avec DFT/BIST (%) | 25 | 50 | 90 |

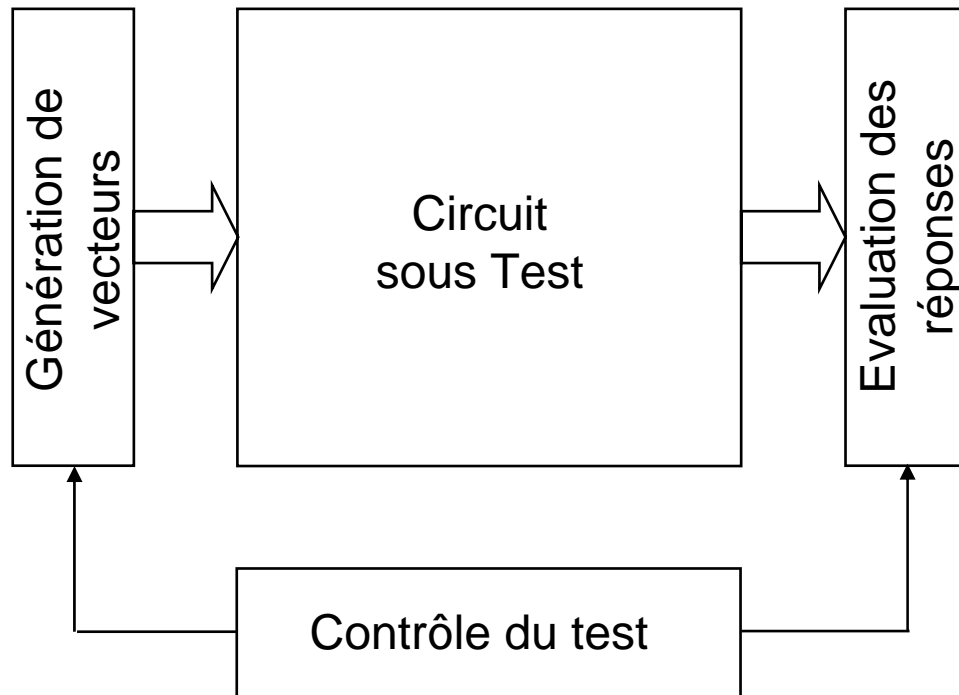
Test intégré

Un passage obligé

Carte de route SIA (aspect test)

| | 1995 | 2001 | 2010 |
|--|------|------|------|
| Vitesse du test (M B/s) | 125 | 225 | 550 |
| Longueur de la séquence de test (millions) | 4 | 8 | 2 |
| Standardisation des logiciels de test (%) | 0 | 25 | 90 |
| Testeurs avec option de test analogique (%) | 20 | 100 | 100 |
| Coût du test en caractérisation (k\$/pin) | 14 | 5 | 1,5 |
| Coût du test en production (k\$/pin) | 3,3 | 1,3 | 0,4 |

Principe du test intégré



Avantages du test intégré

- Suppression de la nécessité de testeur coûteux
- possibilité de test à vitesse nominale
- taux de couverture bon et « modulable »
- temps de test court (vitesse + hiérarchisation)
- possibilité de test en fonctionnement (temps de dormance)

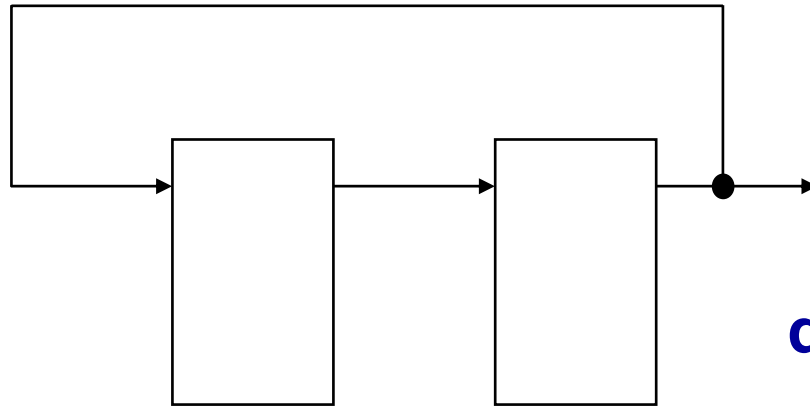
Plan du chapitre

- Génération intégrée de vecteurs de test
- Analyse intégrée des réponses
- Structures pour le test intégré
- Planification et contrôle du test intégré

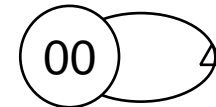
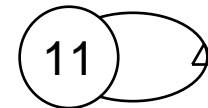
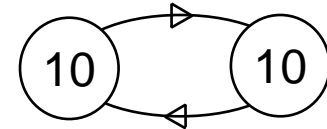
Différentes méthodes de génération

- Test aléatoire (pseudo-aléatoire)
 - pas besoin d'ATPG
 - test long (TC fonction de la longueur de la séquence)
 - pseudo-aléatoire : même caractéristique qu'aléatoire mais appliqué de manière déterministe
- Test déterministe
 - utilisation d'ATPG
 - vecteurs fixés et optimaux
- Test exhaustif (pseudo-exhaustif)
 - pas besoin d'ATPG
 - pseudo-exhaustif : même caractéristique qu'exhaustif mais séquence moins longue

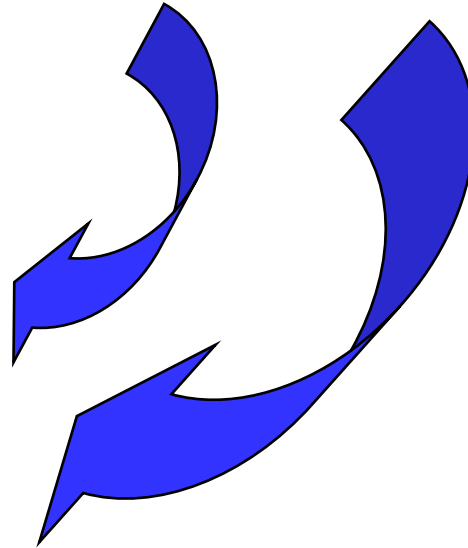
Registre à décalage rebouclé et autonome



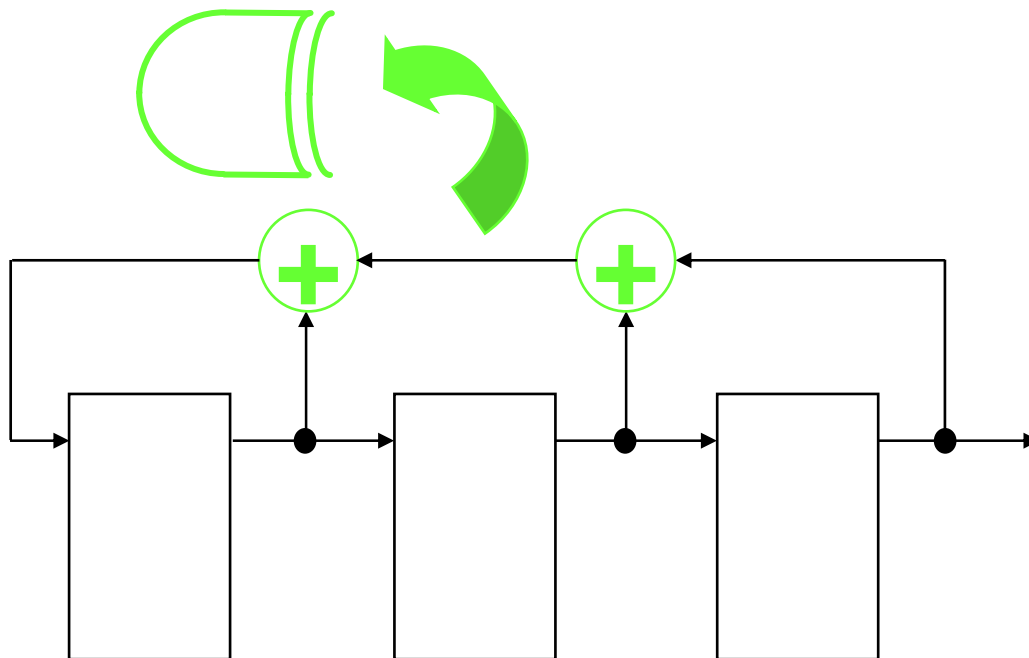
La séquence d'états parcourue dépend de l'état initial



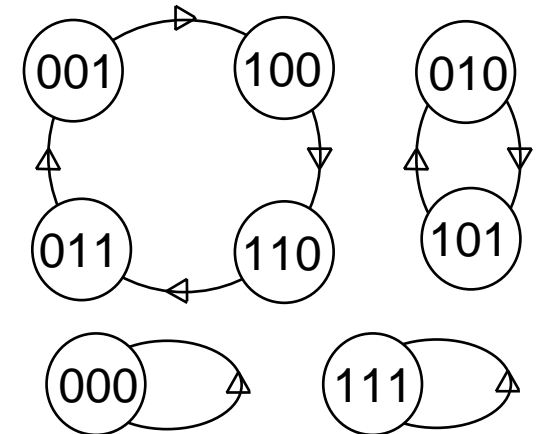
BASCULES D



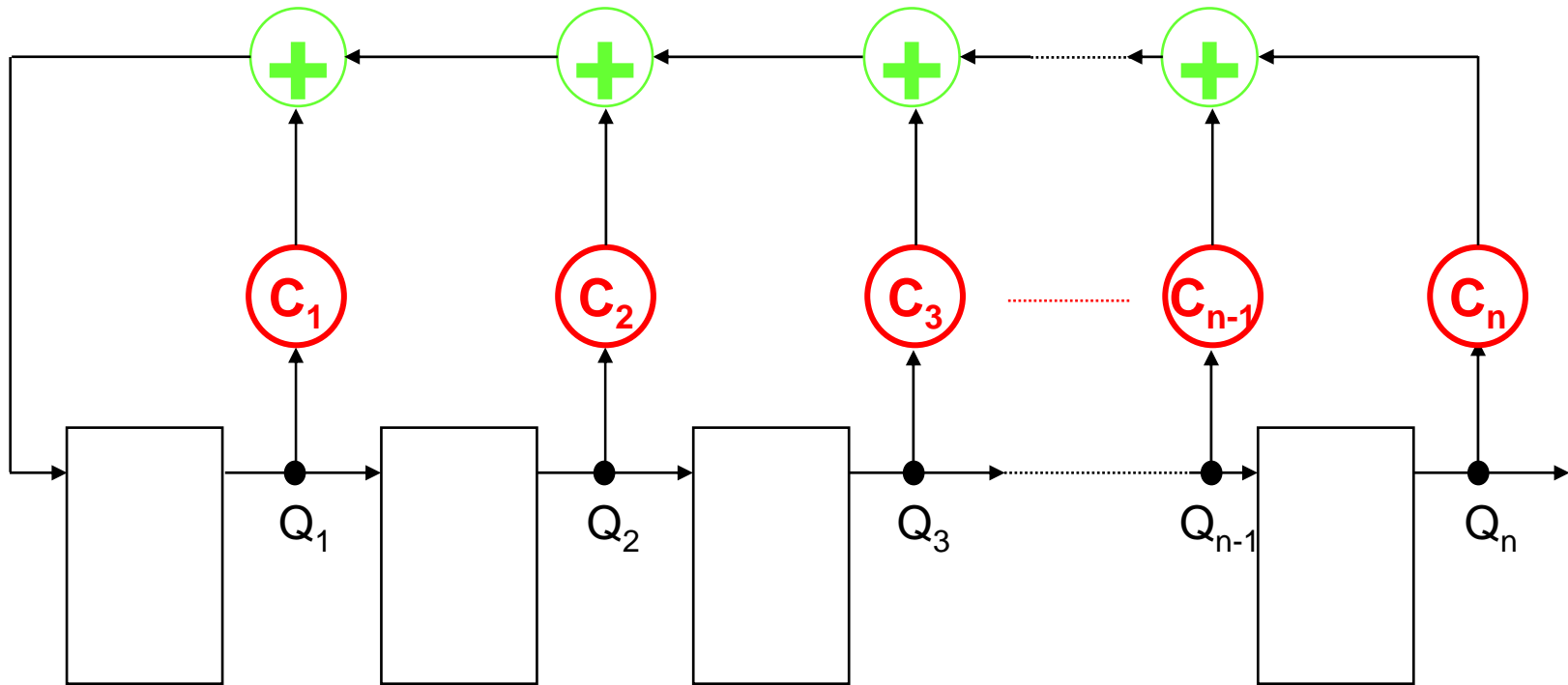
Registre à décalage à rebouclage linéaire (LFSR)



La séquence d'états parcourue dépend toujours de l'état initial



Registre à décalage à rebouclage linéaire (généralisation)

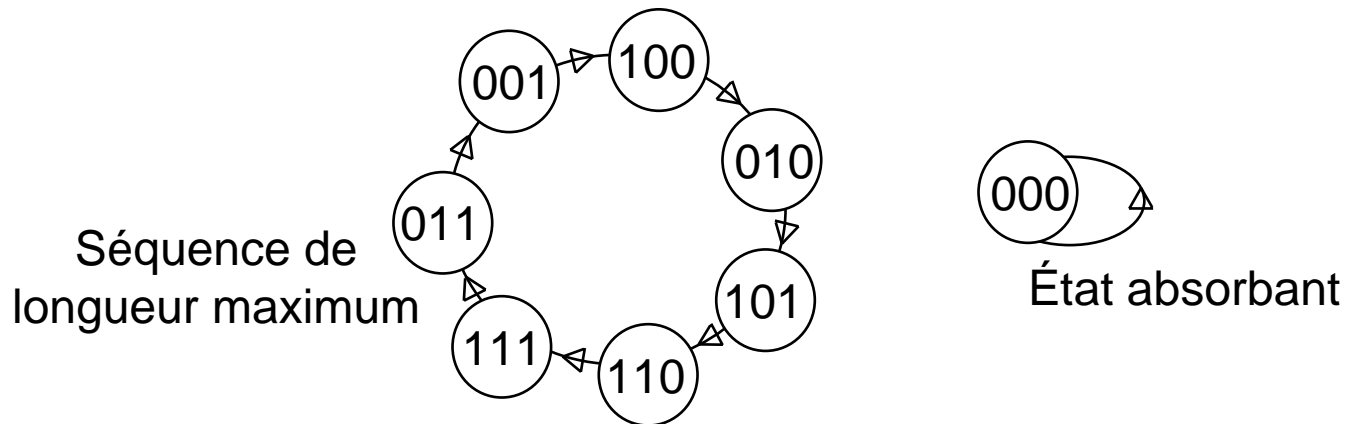
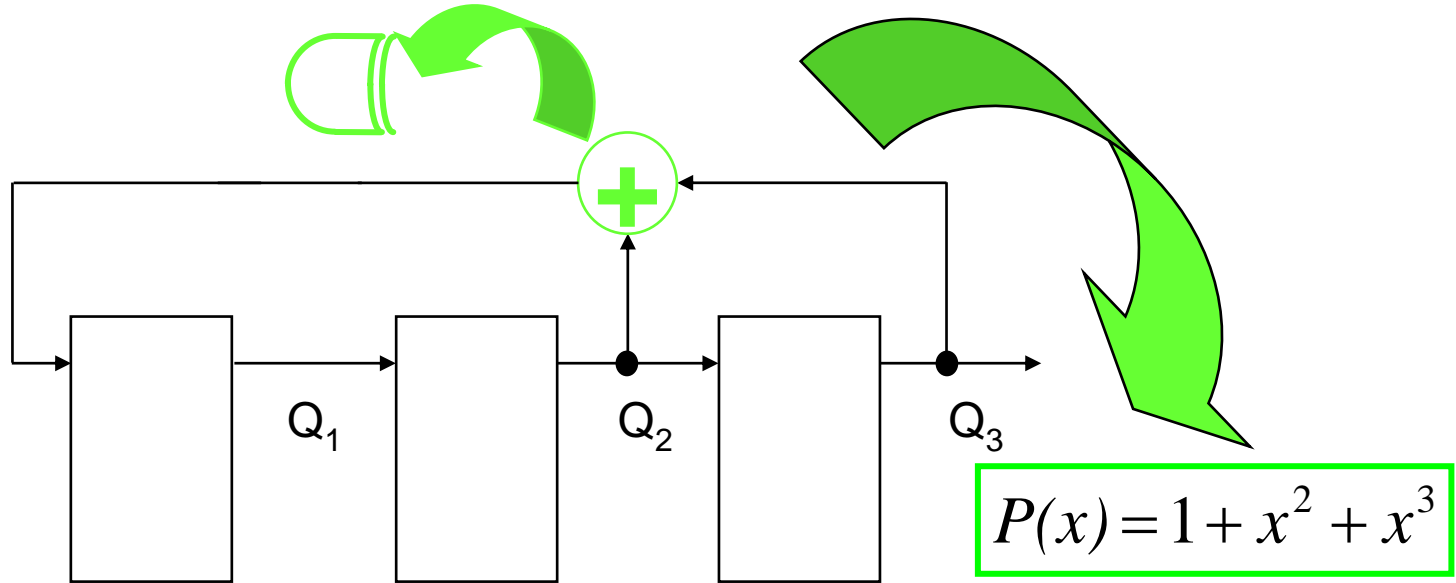


$c_i=1$ si la connexion existe sinon $c_i = 0$

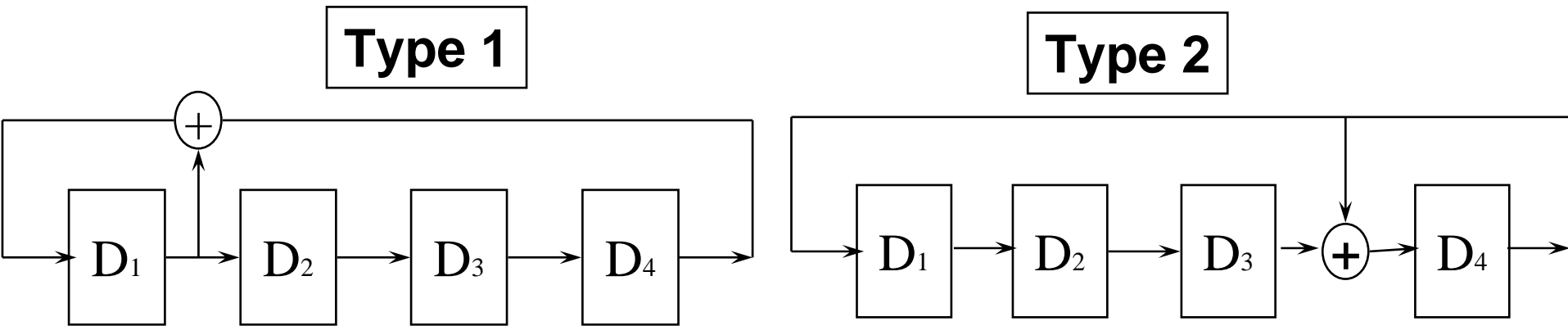
Registre à décalage à rebouclage linéaire (généralisation)

- pour un LFSR de n bascules, une séquence de longueur $2^n - 1$ est dite de longueur maximum
- le polynôme caractéristique d'un LFSR à longueur maximum est dit **polynôme primitif**
- il existe des polynômes primitifs pour toutes valeurs de n
- en pratique on privilégie les polynômes primitifs avec peu de termes (surface plus faible)

LFSR de longueur maximum (n=3)

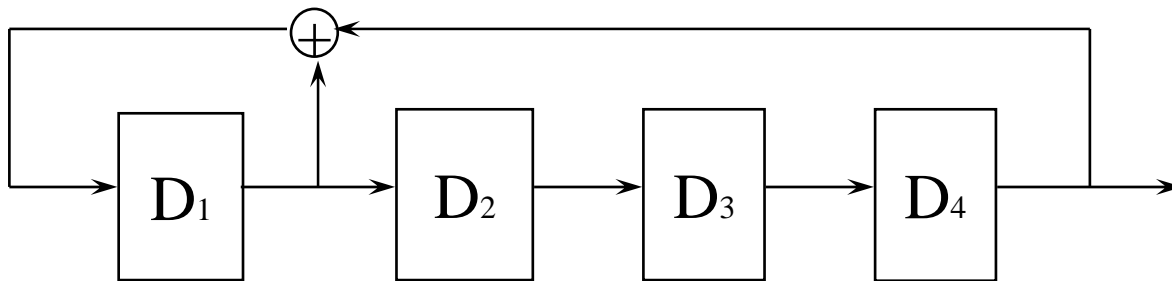


Registre à décalage à rebouclage linéaire LFSR



Type 1 plus facile à générer avec des outils de CAO

Registre à décalage à rebouclage linéaire LFSR type 1



$$g(x) = x^4 + x^1 + 1$$

$a_{-n} = 1 \Rightarrow$

| D_4 | D_3 | D_2 | D_1 |
|-------|-------|-------|-------|
| 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |

LFSR de longueur maximum

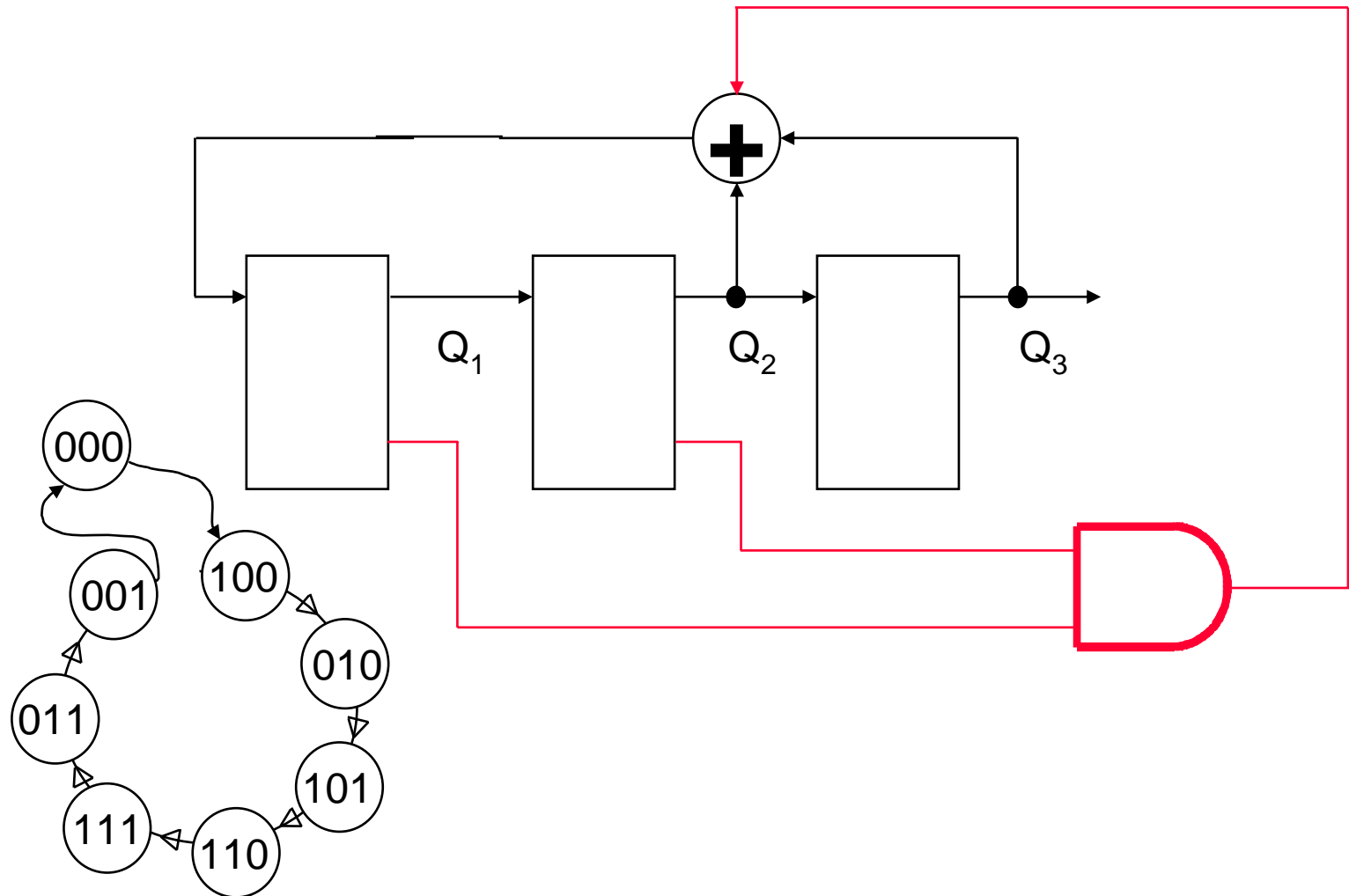
Quelques polynômes primitifs pour $1 \leq n \leq 36$

| degré | | | | | degré | | | | | degré | | | | |
|-------|---|---|---|---|-------|----|----|---|---|-------|----|----|---|---|
| 1 | 0 | | | | 13 | 4 | 3 | 1 | 0 | 25 | 3 | 0 | | |
| 2 | 1 | 0 | | | 14 | 12 | 11 | 1 | 0 | 26 | 8 | 7 | 1 | 0 |
| 3 | 1 | 0 | | | 15 | 1 | 0 | | | 27 | 8 | 7 | 1 | 0 |
| 4 | 1 | 0 | | | 16 | 5 | 3 | 2 | 0 | 28 | 3 | 0 | | |
| 5 | 2 | 0 | | | 17 | 3 | 0 | | | 29 | 2 | 0 | | |
| 6 | 1 | 0 | | | 18 | 7 | 0 | | | 30 | 16 | 15 | 1 | 0 |
| 7 | 1 | 0 | | | 19 | 6 | 5 | 1 | 0 | 31 | 3 | 0 | | |
| 8 | 6 | 5 | 1 | 0 | 20 | 3 | 0 | | | 32 | 28 | 27 | 1 | 0 |
| 9 | 4 | 0 | | | 21 | 2 | 0 | | | 33 | 13 | 0 | | |
| 10 | 3 | 0 | | | 22 | 1 | 0 | | | 34 | 15 | 14 | 1 | 0 |
| 11 | 2 | 0 | | | 23 | 5 | 0 | | | 35 | 2 | 0 | | |
| 12 | 7 | 4 | 3 | 0 | 24 | 4 | 3 | 1 | 0 | 36 | 11 | 0 | | |

$$P(x) = x^{34} + x^{15} + x^{14} + x + 1$$



Génération exhaustive



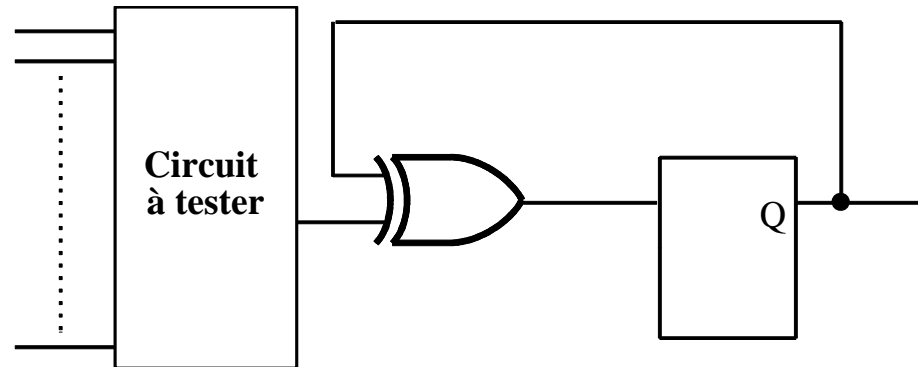
Génération déterministe

- Génération d'une séquence de test par ATPG (ou autre moyen)
- génération de cette séquence par une structure matérielle :
 - * ROM : simple mais coûteux
 - * MEF : toujours simple et toujours coûteux
 - * recherches en cours

Analyse intégrée des réponses

- Vérification de la parité
- Compaction de la réponse par comptage :
 - * comptage des 1 (0)
 - * comptage des transitions
- Compaction de la réponse par LFSR

Vérification de la parité

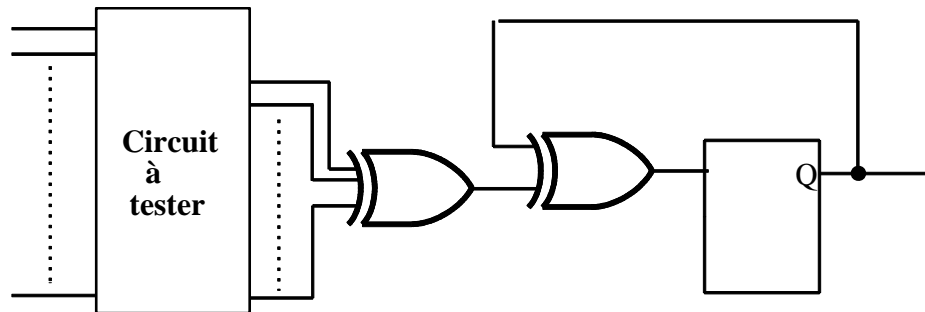


- Initialisation de la bascule
- Somme sur x cycles(mod 2) de la sortie
- Détection des fautes simples sur un bit et d'erreurs en nombre impair sur une chaîne de m bits

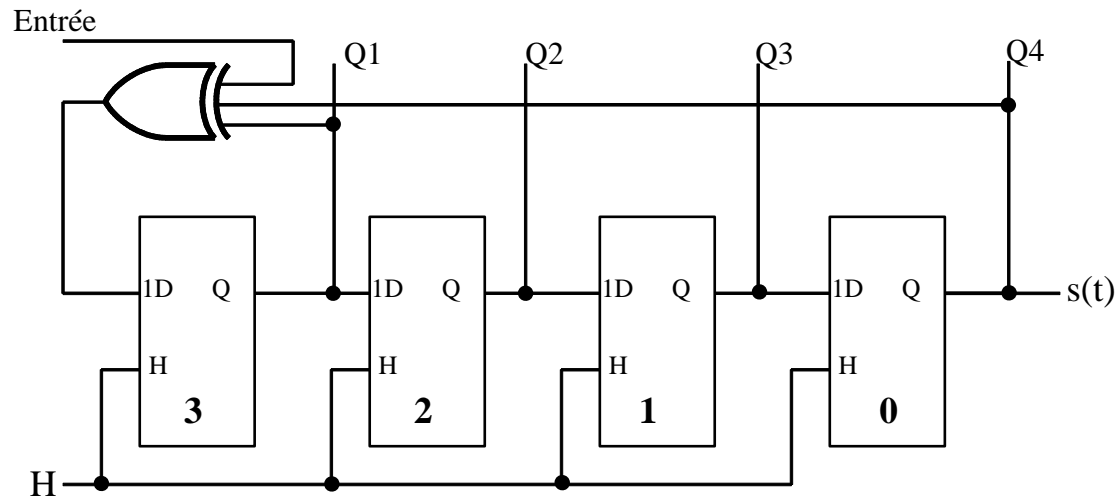
- Probabilité de masquage $P_m = \frac{2^m - 1}{2^m - 1}$

Vérification de la parité (plusieurs sorties)

- Associer un vérificateur de parité à chaque sortie (coût élevé)
- « groupage » des sorties avant compression (taux de masquage plus élevé)

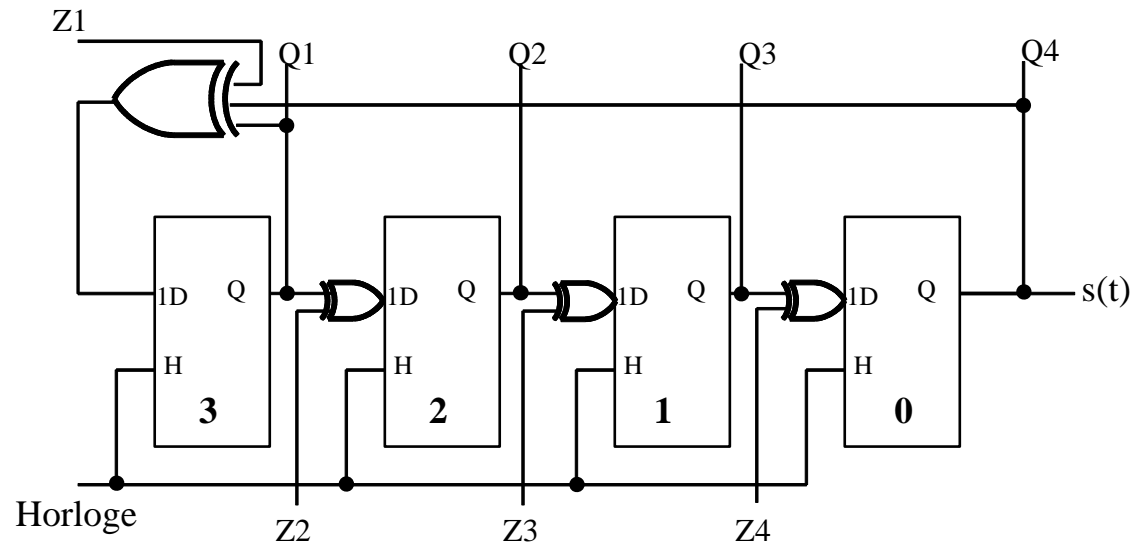


Compaction par utilisation de LFSR (une seule sortie)



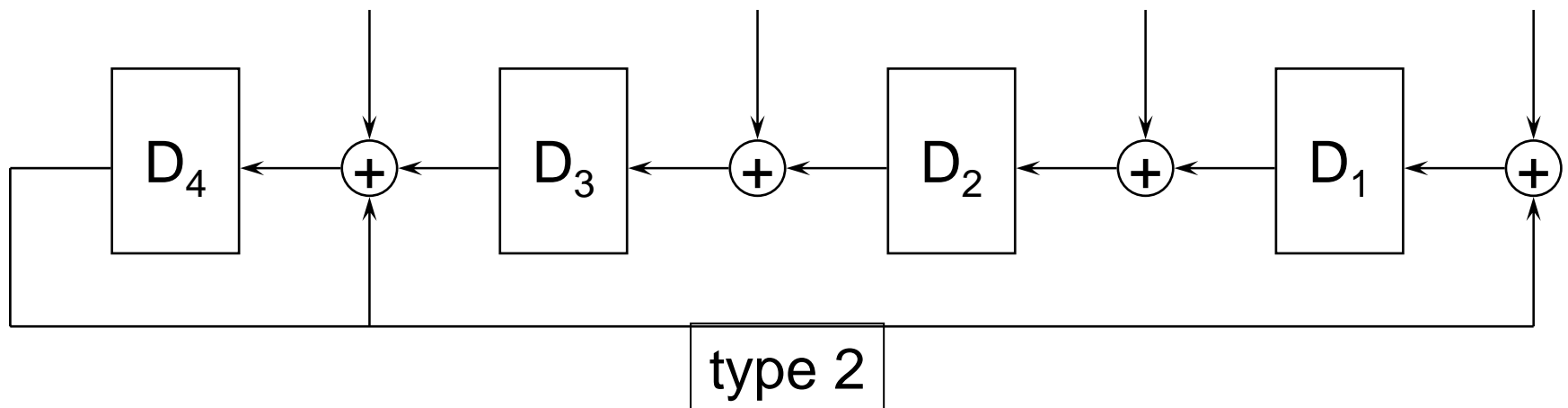
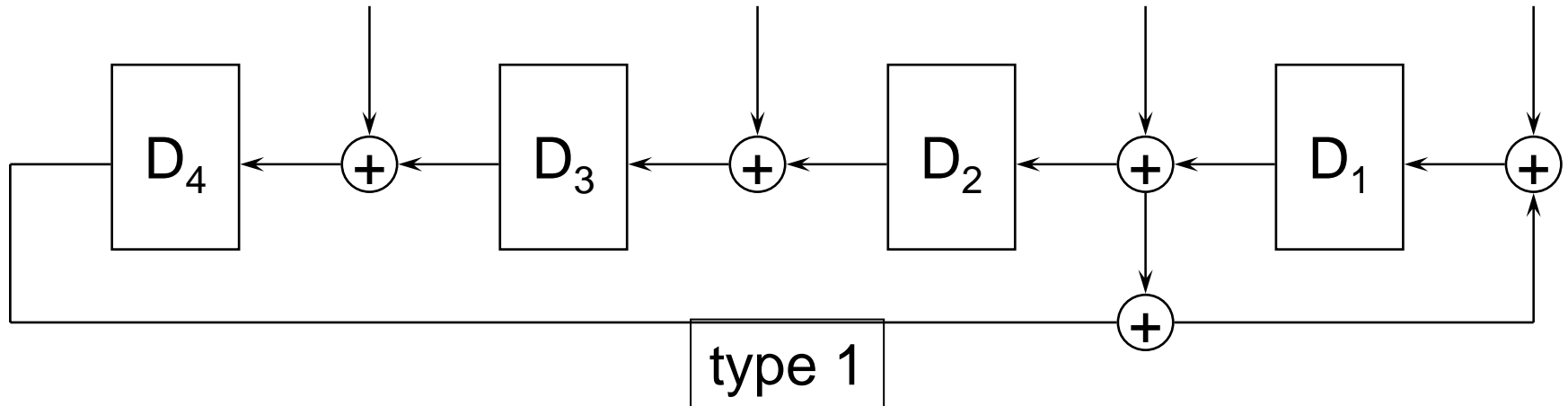
- Taux de masquage faible et surtout « modulable »
- de l'ordre de $Pm = \frac{1}{2^n}$ si n est le nombre de bascules

Compaction par utilisation de MISR (plusieurs sorties)

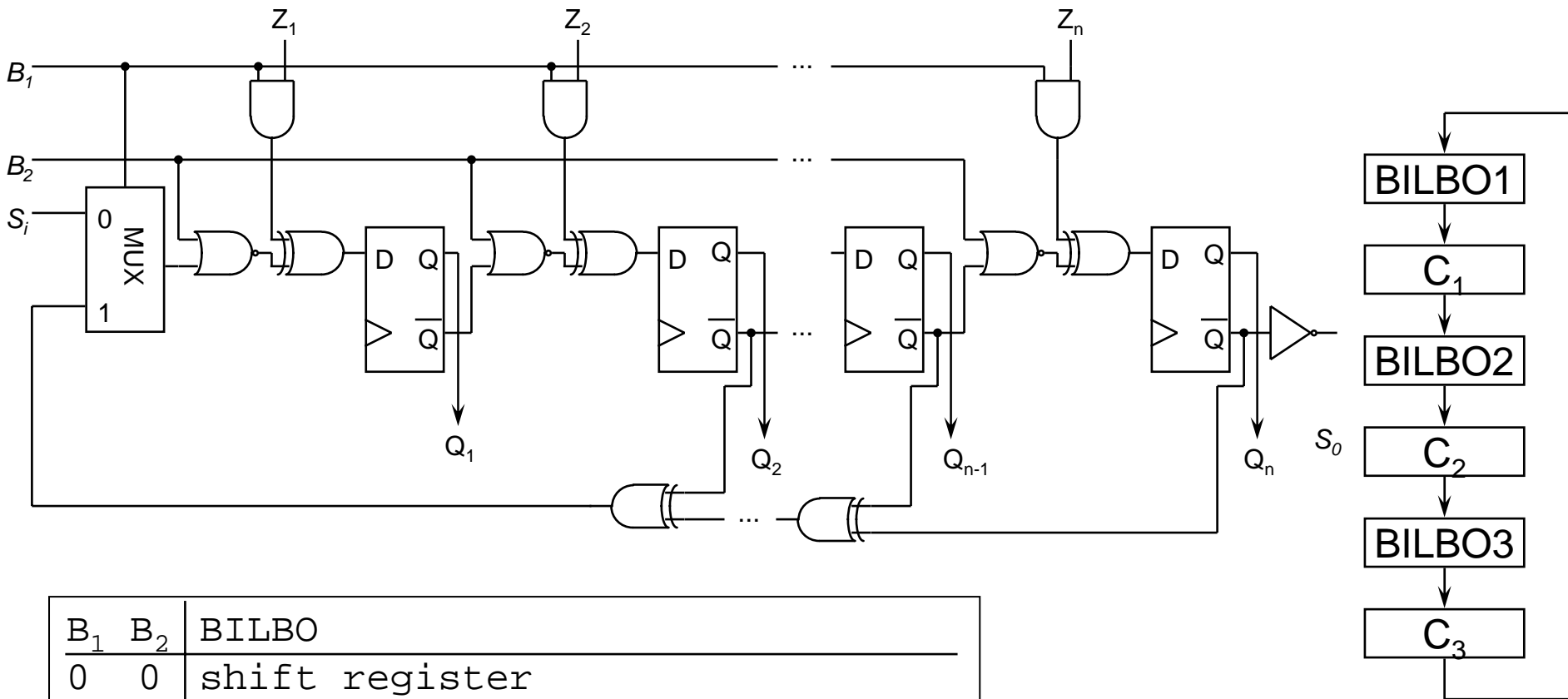


- Taux de masquage faible et « modulable »
- Toujours de l'ordre de $P_m = \frac{1}{2^n}$ si n est le nombre de bascules

Compaction par utilisation de MISR (plusieurs sorties)



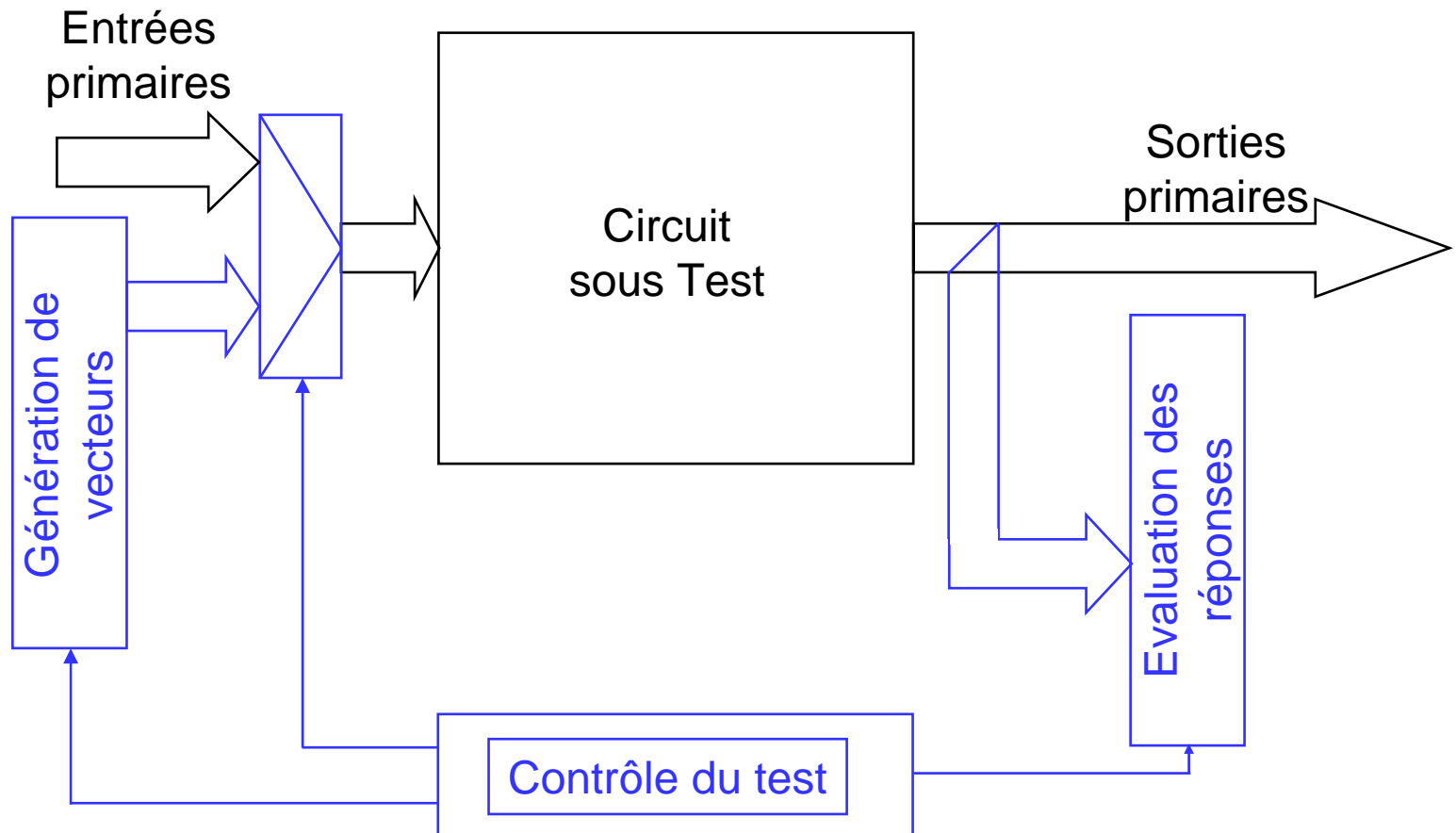
Built-in Logic Block Observation [Koenemann 79.80]



| B_1 | B_2 | BILBO |
|-------|-------|----------------------------------|
| 0 | 0 | shift register |
| 0 | 1 | reset |
| 1 | 0 | MISR (input * constant * LFSR) |
| 1 | 1 | parallel load (normal operation) |

Mise en œuvre

Test parallèle : « Test per clock » »



« Test per clock »

POUR et CONTRE

POUR

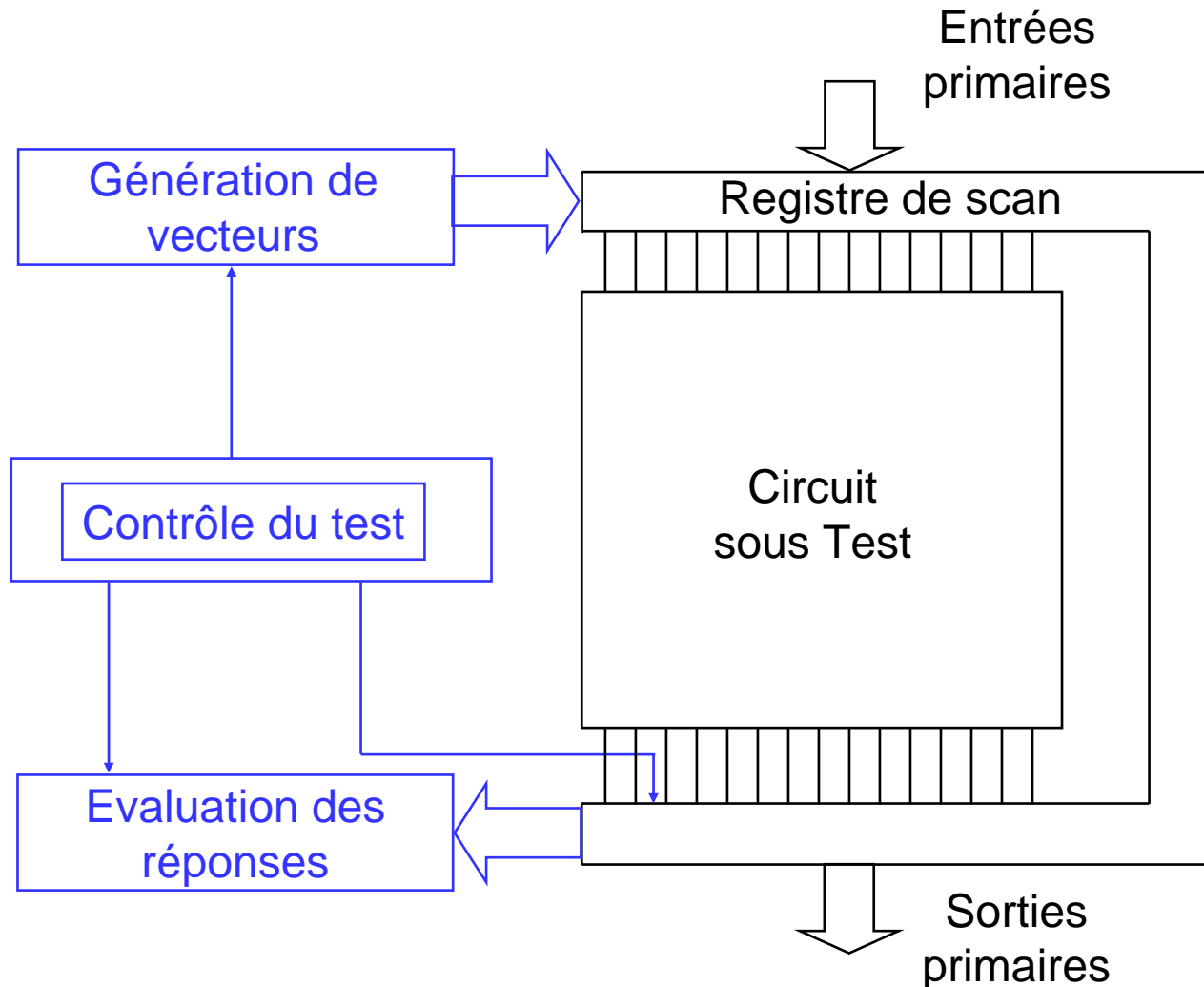
- temps de test réduit
- le test à la vitesse nominale est possible
- les tests utilisant deux vecteurs consécutifs sont envisageables moyennant l'utilisation de registres de test appropriés

CONTRE

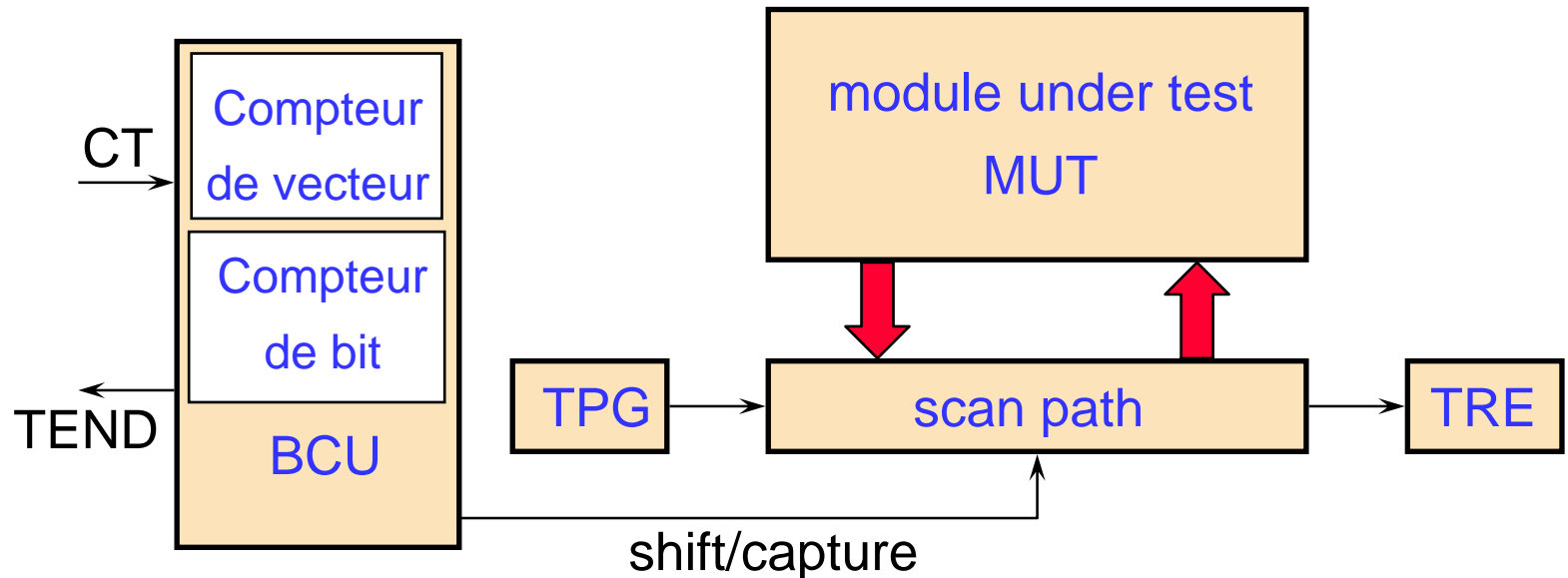
- le surcoût matériel est élevé car les registres (BILBO) prennent plus de place qu'un scan avec générateur série
- la partie contrôle est plus compliquée que dans le test per scan
- l'insertion des registres de test a un impact sur les performances du système

Mise en œuvre

Test série : « Test per scan » »



Partie contrôle du test série : « Test per scan »



- 1) TPG génère une séquence de bits et remplit le scan
- 2) Sur l'horloge de "capture" :
 - appliquer le contenu du scan path au MUT
 - charger la réponse du MUT dans le scan path
- 3) Décaler une nouvelle séquence de bits et compacter la réponse (TRE)

« Test per scan »

POUR et CONTRE

POUR

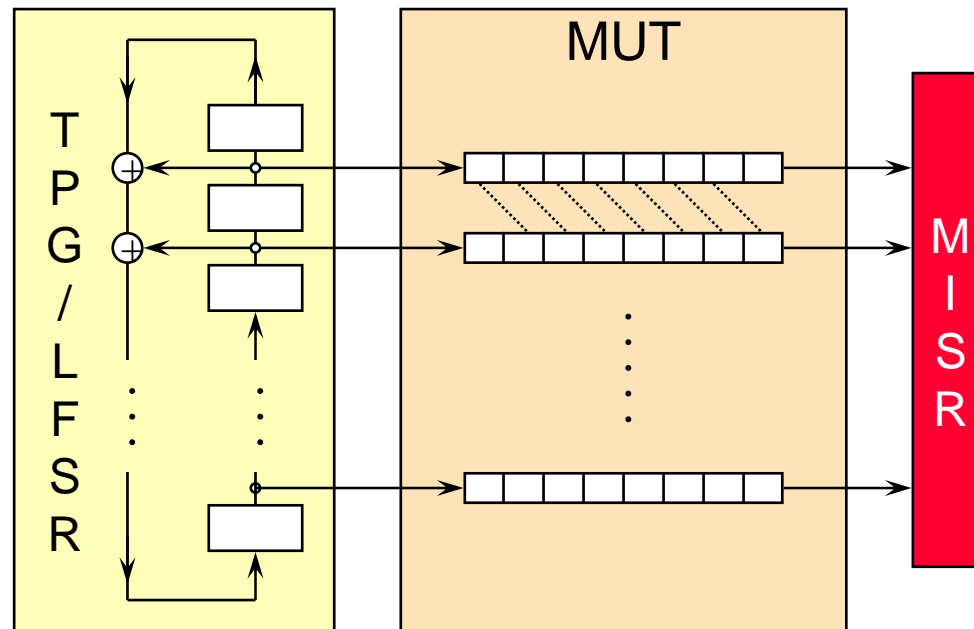
- convient à tout flot de conception commercial supportant le scan
- le matériel pour le BIST est extérieur au MUT d'où un faible impact sur les performances du système
- le contrôle du BIST est simple
- l'approche peut être facilement étendu au scan partiel et au scan multiple
- en général, le surcoût matériel est plus faible que dans le schéma de "test per clock"

CONTRE

- temps de test long (entrée série)
- de nombreuses fautes nécessitent deux vecteurs de test et ne peuvent pas être détectés par scan
- le module sous test n'est pas testé à vitesse nominale

Mise en œuvre

Test série : « multiple scan »

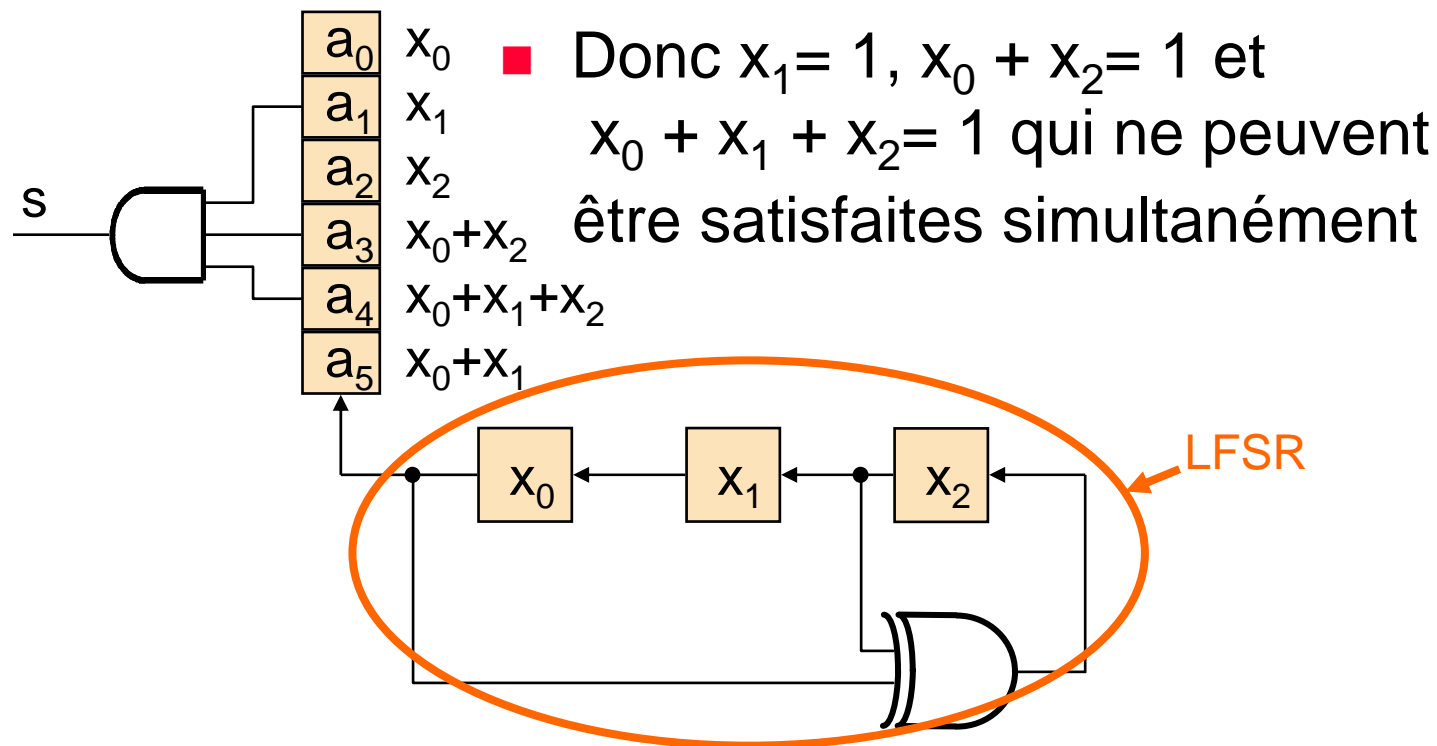


- Bardell and McAnney, 1984 ("STUMPS" à IBM)
- Réduction du temps de test
- Corrélation (dépendance linéaire) entre les signaux d'entrée (insertion de "Phase shifter")

Dépendance linéaire

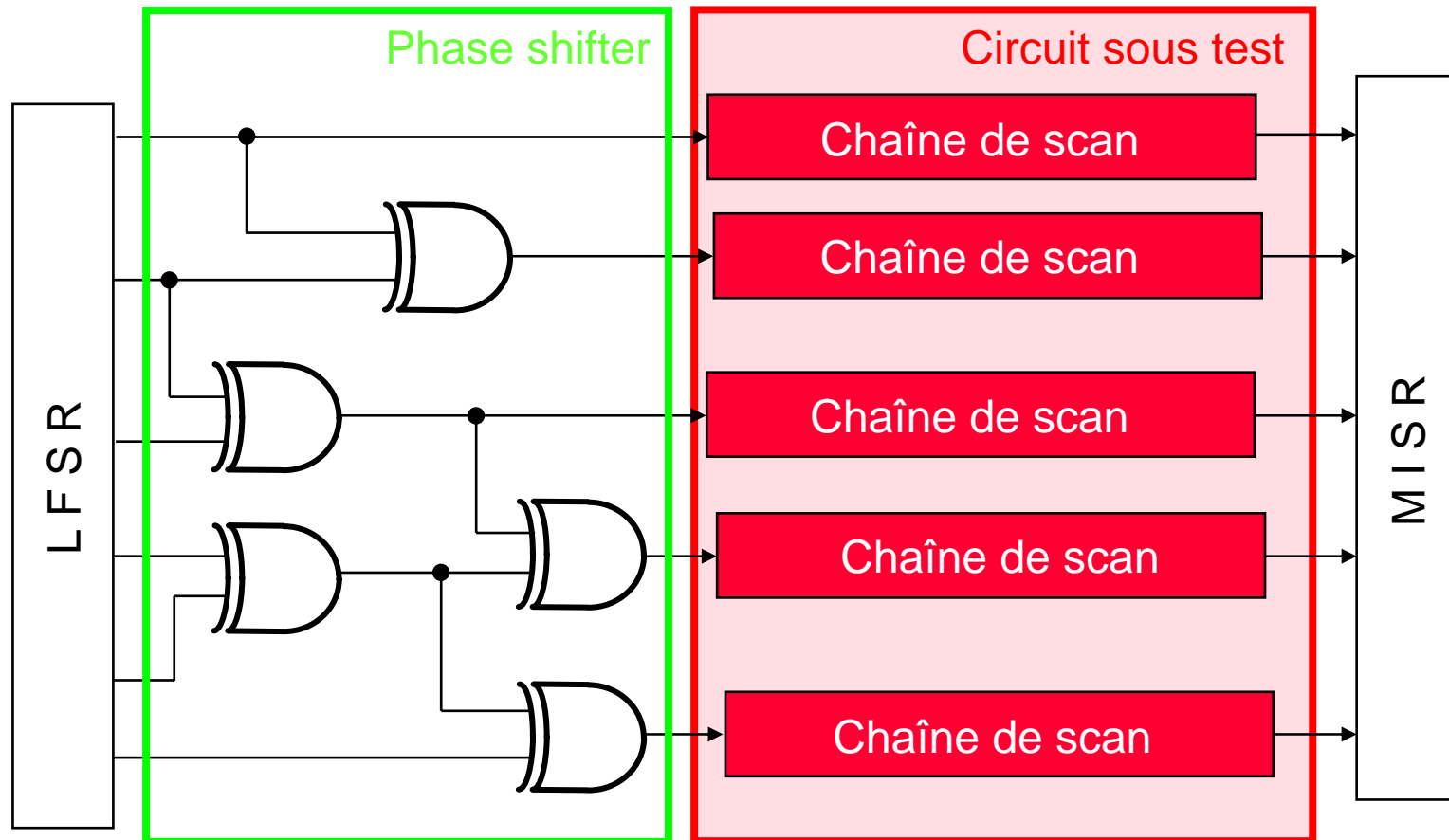


- Par construction les bits fournis par un LFSR sont linéairement dépendants
- par exemple le test du collage à zéro de s nécessite $a_1 = a_3 = a_4 = 1$



Insertion de "Phase Shifter"

[Rajski, Tyszer] VLSI test Symposium1998



- Bonne séparation des chaînes avec 1 à 3 portes OU/ET par chaîne