TDS: archi

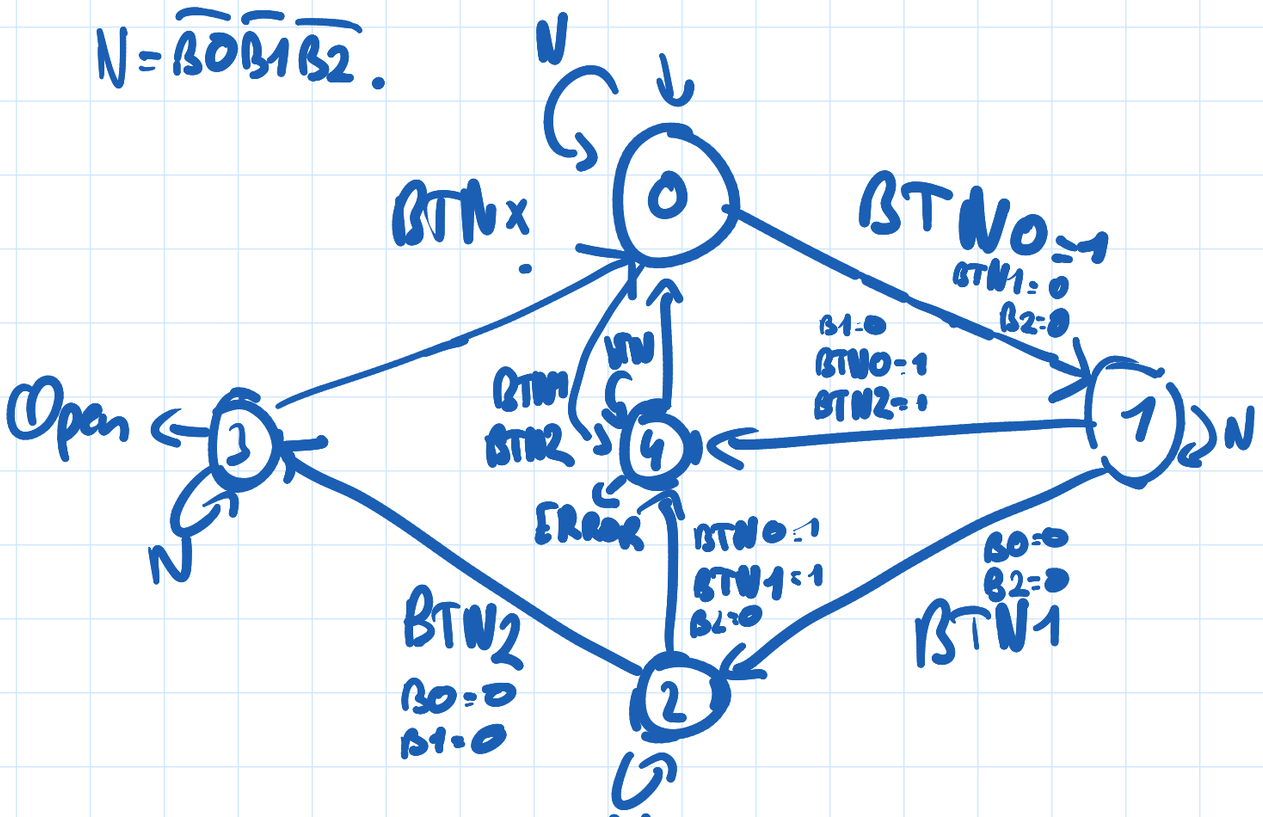
Cette séance de TD est préparatoire au TP 6 qui sera à réaliser sans directive .

Vous allez construire un détecteur de séquence synchrone. Ce système permet de détecter une séquence entrée par l'usager à l'aide de trois boutons poussoir.

- Les boutons doivent être activés à tour de rôle dans le bon ordre pour générer la bonne séquence.
- Lorsque le système détermine que la bonne séquence a été entrée, il ouvre la porte et il active une diode électroluminescente (Led open).
- Les boutons doivent être activés dans l'ordre suivant pour générer la bonne séquence : BTN0, BTN1, BTN2.
- Un quatrième bouton BTN3 sert comme Reset(RAZ) asynchrone de votre système. En cas de reset la saisie du code reprend au début à tout moment du processus. Son effet matériel est un reset asynchrone sur les bascules D du registre d'état.
- En cas de faux code saisi une seconde LED s'allume et il faudra alors faire un RESET pour reprendre la saisie et éteindre cette LEDerror . La sortie associée à cette led sera nommée ERROR
- A la différence des TP où il faudra réaliser un circuit spécial, on considère ici que les boutons, lorsqu'ils sont appuyés produisent une impulsion qui dure un cycle d'horloge.
- Un signal OPEN en sortie permet d'ouvrir la port quand il vaut 1 et allume la led open. Pour que l'automate revienne ensuite dans l'état initial pressez un des boutons ou le RESET.

L'utilisation d'une FSM est tout indiquée pour détecter une séquence d'évènements.

1. Dessinez l'automate synchrone de cette FSM. Pour chaque état précisez les transitions qui en partent vers d'autres états et la valeur produite sur les sorties OPEN et ERROR.

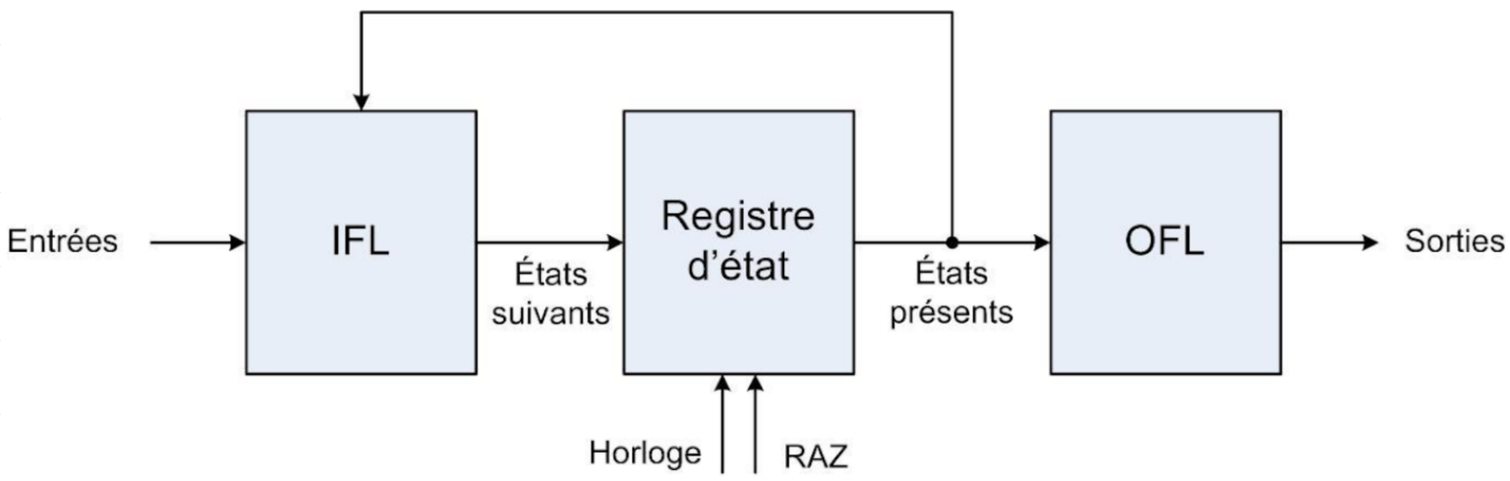


N

1 = 000 ; 2 = 001 ; 3 = 010 ; 4 = 011 ; 5 = 100

On peut observer qu'une FSM est composée

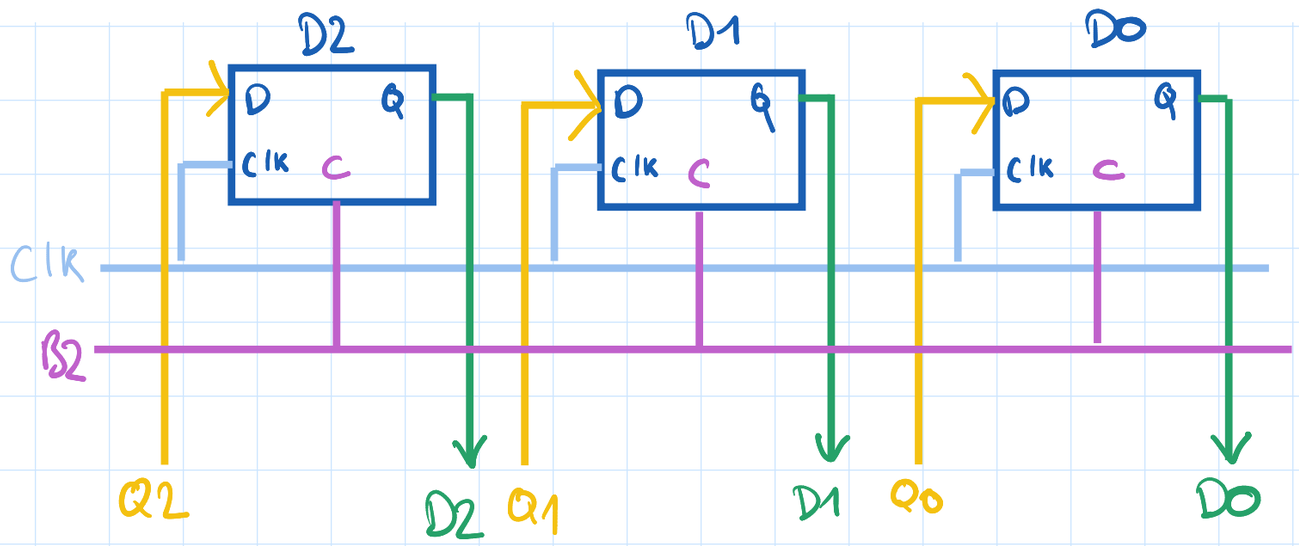
- d'un circuit combinatoire IFL (input forming logic) ou *next_state* pour traiter les entrées et générer les états suivants,
- d'un registre d'état ou *state* pour maintenir la valeur de l'état courant pendant toute la durée d'une période d'horloge et
- d'un circuit combinatoire OFL (output forming logic) ou *sortie* pour activer les sorties de la FSM



3 Réalisez le registre d'état *state* avec des bascules, choisissez les bonnes bascules!

4 Pour chaque état du registre donnez la table de transition *next_state* en fonction des boutons.

5 Pour chaque état du registre donnez la table de vérité des sorties OPEN et ERROR



E	Q2	Q1	Q0	B0	B1	B2	E_N	D2	D1	D0	L0	Lc
E0	0	0	0	0	0	0	E0	0	0	0	0	0
E0	0	0	0	1	0	0	E1	0	0	1	0	0
E0	0	0	0	x	1	x	E4	1	0	0	0	0
E0	0	0	0	x	x	1	E4	1	0	0	0	0
E1	0	0	1	0	0	0	E1	0	0	1	0	0
E1	0	0	1	0	1	0	E2	0	1	0	0	0
E1	0	0	1	1	x	x	E4	1	0	0	0	0
E1	0	0	1	x	x	1	E4	1	0	0	0	0
E2	0	1	0	0	0	0	E2	0	1	0	0	0
E2	0	1	0	0	0	1	E3	0	1	1	0	0
E2	0	1	0	1	x	x	E4	1	0	0	0	0
E2	0	1	0	x	1	x	E4	1	0	0	0	0
E3	0	1	1	0	0	0	E3	0	1	1	1	0
E3	0	1	1	0	0	1	E0	0	0	0	1	0
E3	0	1	1	0	1	0	E0	0	0	0	1	0
E3	0	1	1	1	0	0	E0	0	0	0	1	0
E4	1	0	0	x	x	x	E4	1	0	0	0	1

D0 =