ÉCOLE POLYTECHNIQU
FÉDÉRALE DE LAUSANN

REGISTRE D'EXTENSION SÉ

PIERRE-YVES ROCHAT, I

RÉV 2015/09/18

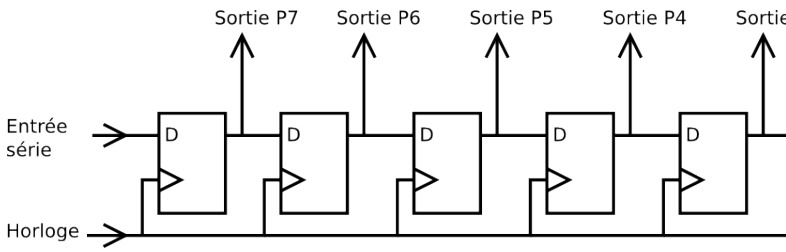
BESOIN DE BROCHES

Les enseignes et surtout les afficheurs à LED nécessitent beaucoup de broches dont on peut comma
d'entrées-sorties pour faire face à ce besoin. Plusieurs circuits logiques classiques offrent la fonctior
adressables, qui sont très pratiques. Le circuit 74HC259 est un latch adressable à 8 sorties.

Mais le composant le plus souvent utilisé dans le domaine des afficheurs à LED est le *registre série-p*

REGISTRE SÉRIE

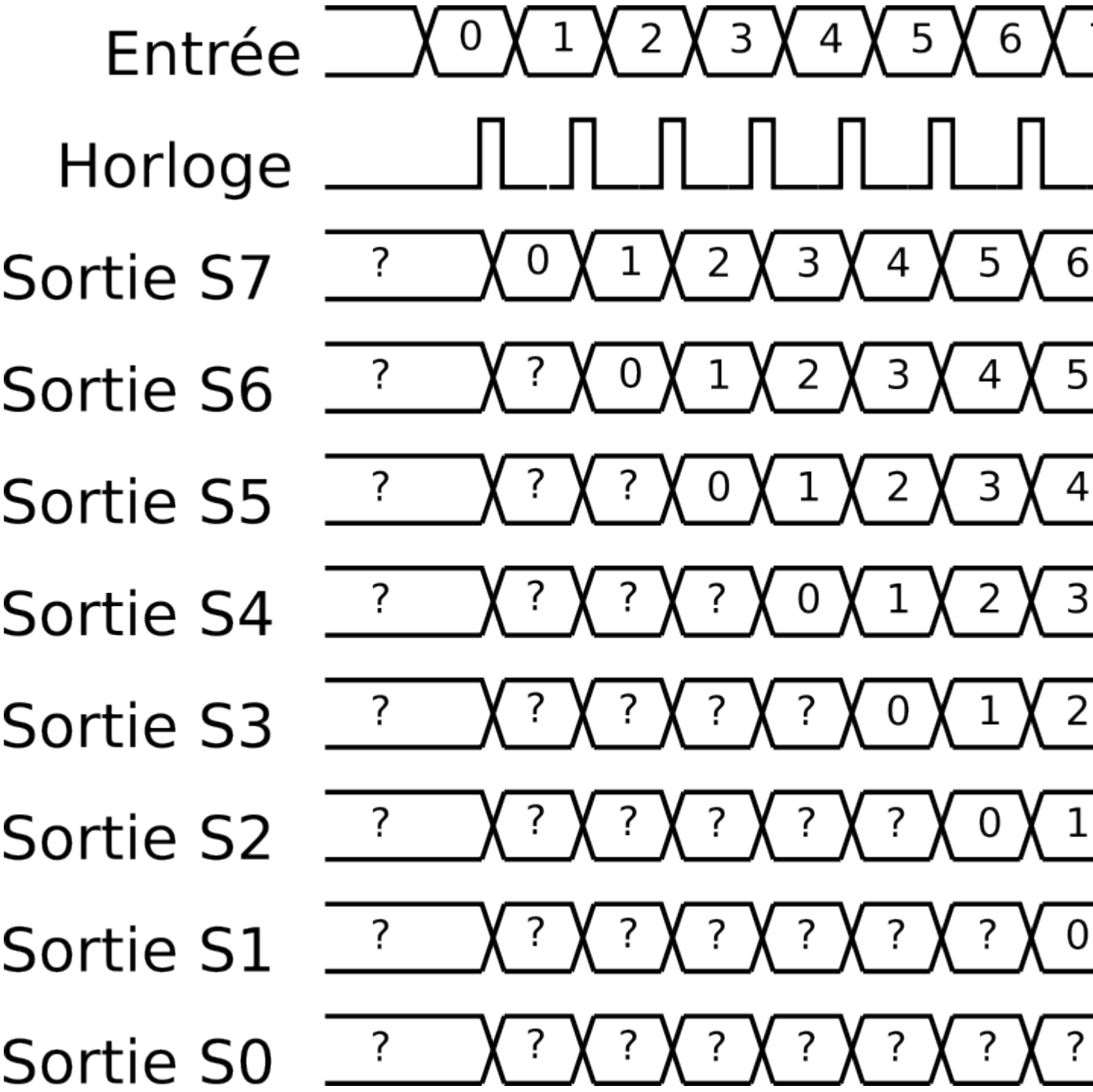
Les registres à décalage sont réalisés avec des bascules. Voici le schéma d'un registre série 8 bits :



Registre série

On y trouve 8 bascules D. Les horloges des 8 bascules sont reliées ensemble. L'entrée D de la premiè
seconde bascule et ainsi de suite. Le système a 8 sorties.

Voici un diagramme des temps qui permet de comprendre comment fonctionne ce registre. Pour ch
flanc montant de l'horloge.

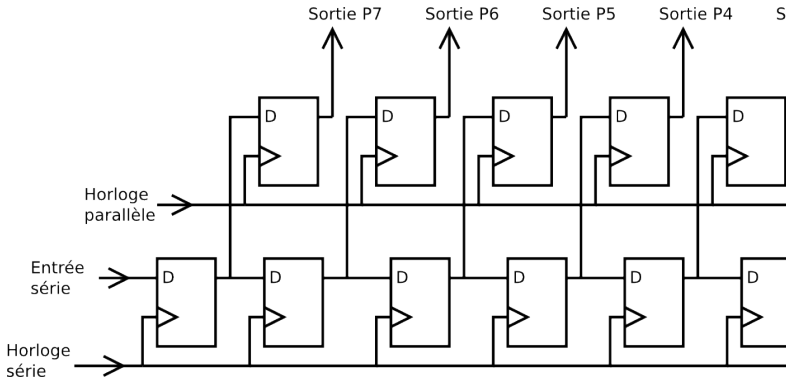


Les valeurs notées 0,1,2..7 correspondent à des valeurs binaires 0 ou 1 placées successivement sur l'
coup d'horloge. Après 8 coups d'horloge, les 8 valeurs envoyées en série vont se retrouver sur les so
les 8 sorties.

Ce dispositif permet donc de disposer 8 sorties, tout en ne monopolisant que 2 broches du microcon
valeurs non désirées vont apparaître sur les sorties. Dans certains cas particuliers, ce n'est pas grave
est très sensible et des artefacts sur les LED deviennent vite gênants.

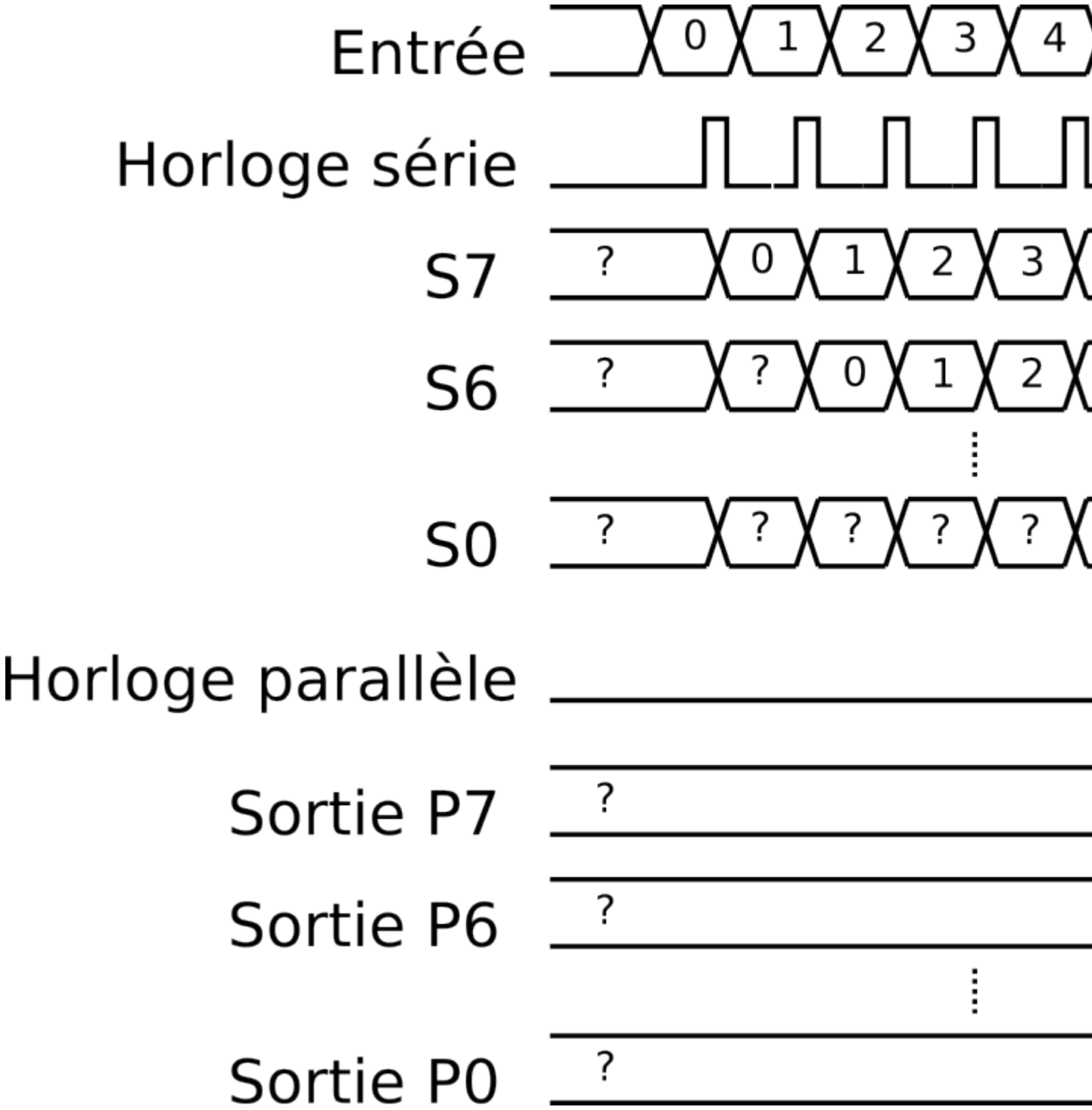
Registre série-parallèle

Ajoutons 8 bascules D supplémentaires à notre montage. Ces 8 bascules forment cette fois un regi
horloges des 8 bascules sont reliées ensemble.



Registre série-parallèle

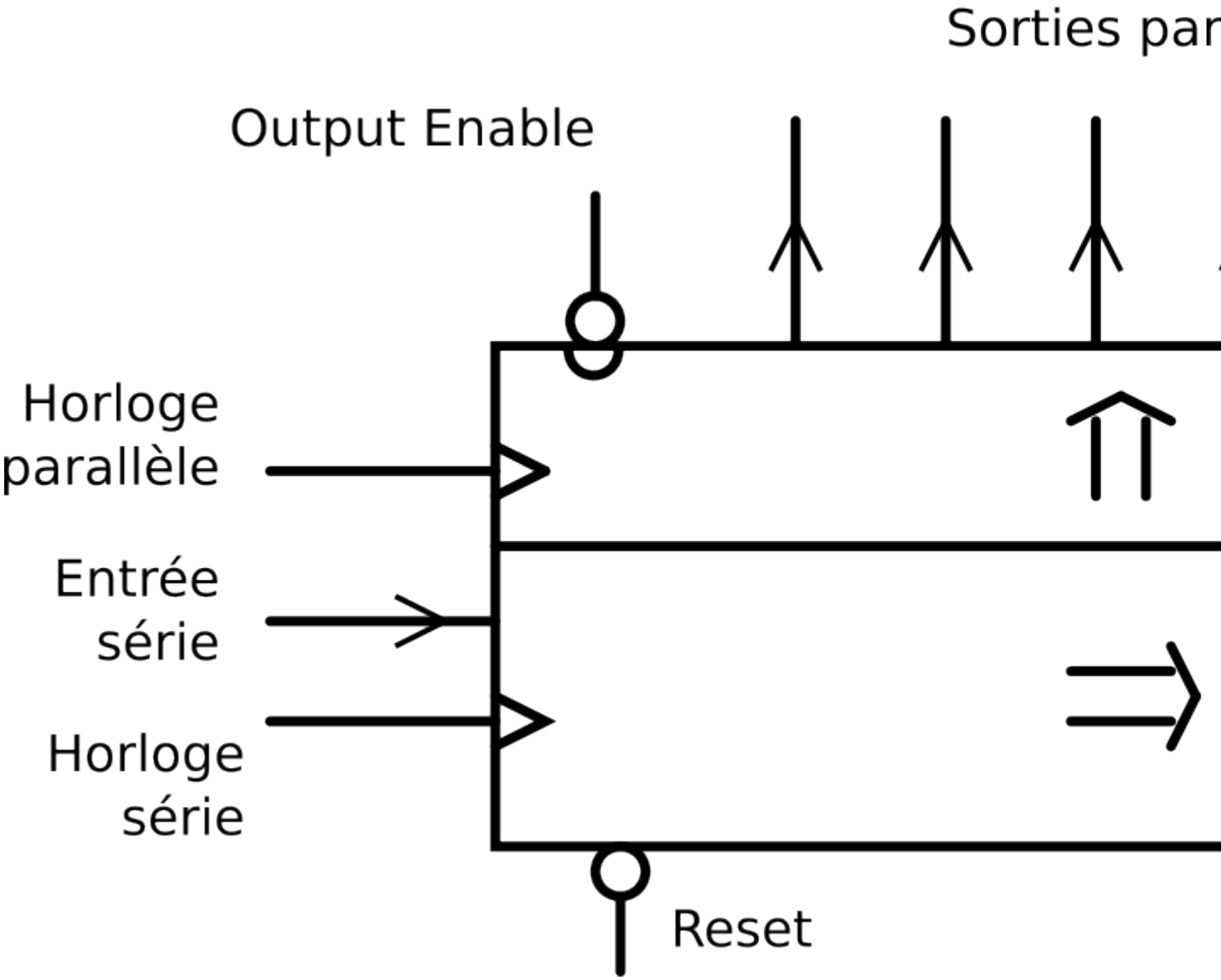
Sur ce diagramme des temps, on voit que les données transmises en série sont ensuite copiées sur l



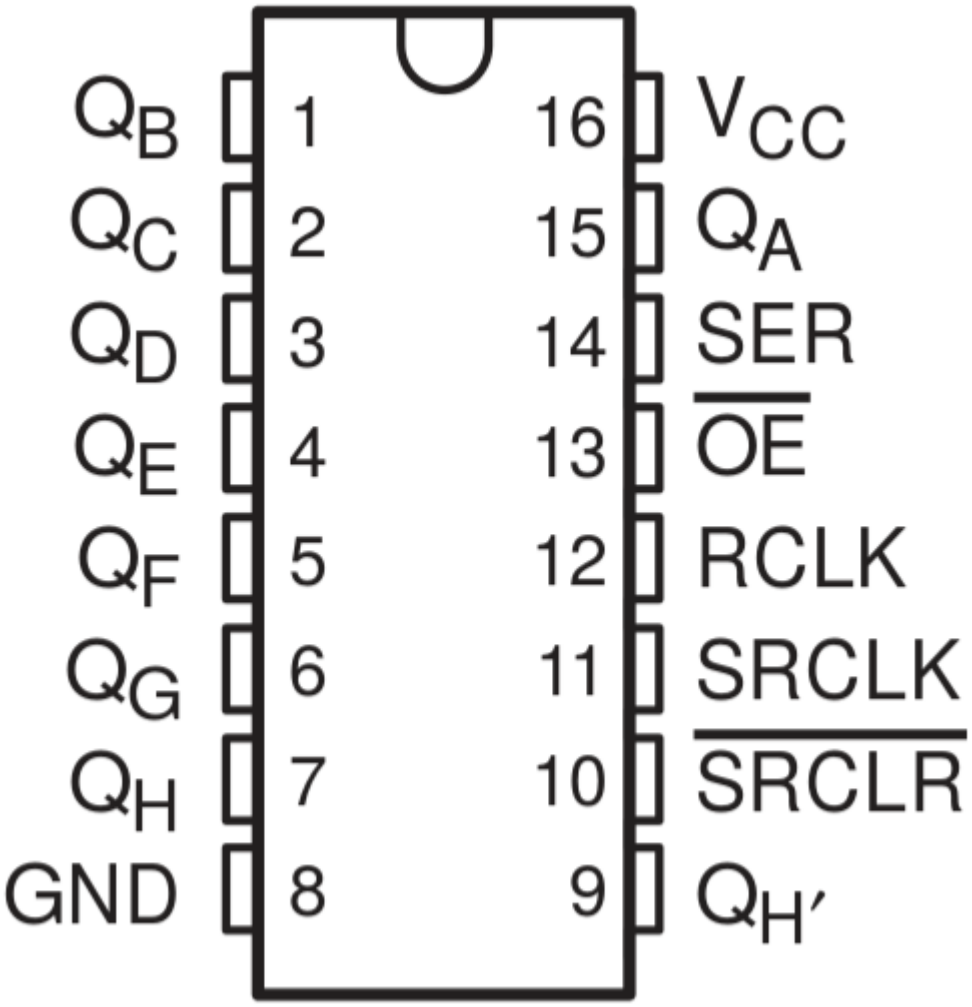
Les 8 valeurs arrivent en même temps sur les sorties du registre parallèle. Les anciennes valeurs so
jour en même temps. Il n'y a donc pas de risque d'artefacts.

Le circuit 74HC595

Le circuit 74HC595 est très souvent présent dans des afficheurs à LED. C'est un circuit C-MOS, de la
sont à *trois états*, commandées par le signal Output Enable. Une entrée Reset permet de forcer les v

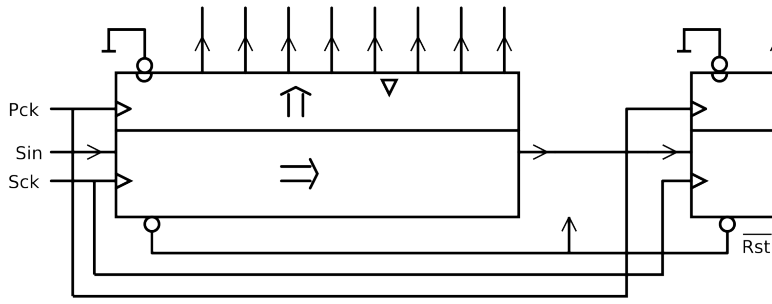


{ width=10cm }



{ width=4cm }

Sa sortie série permet de cascader les circuits, c'est-à-dire les placer les uns à la suites des autres, co



Registre série 16 bits utilisant 2 registres 74H

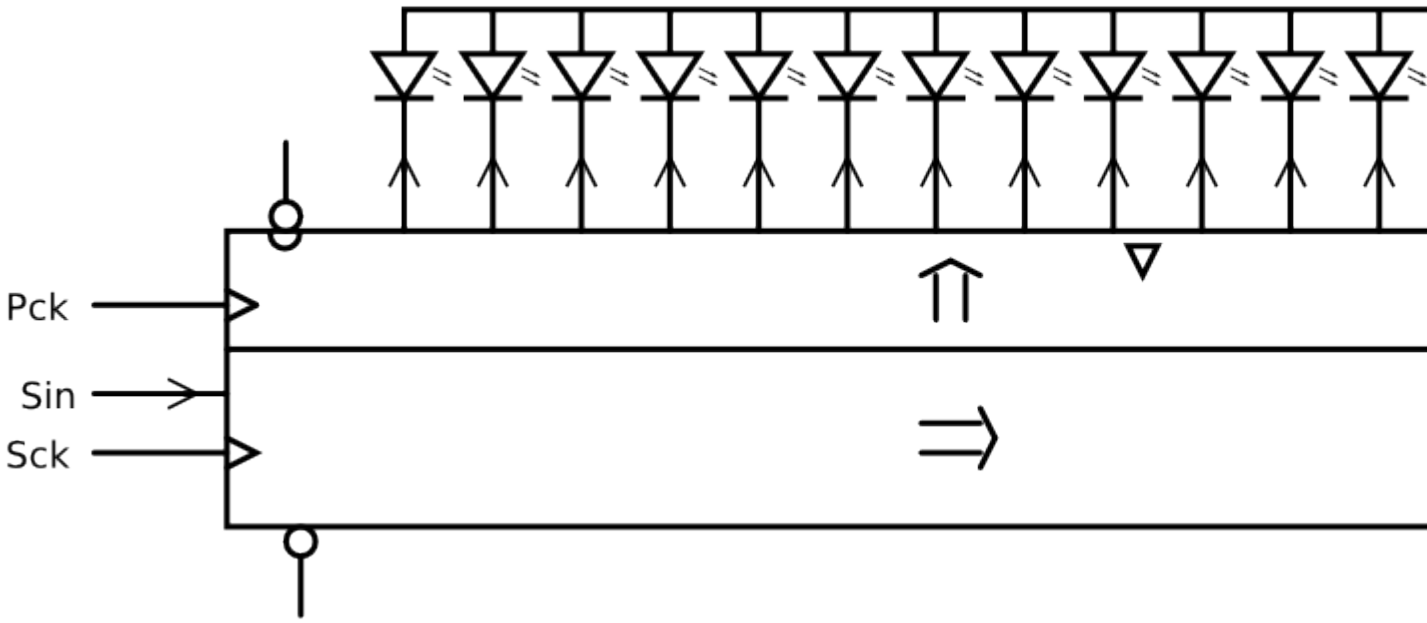
Pour ce faire, les horloges doivent être communes à tous les registres : l'horloge série, notée ***Sck*** l'entrée du deuxième et ainsi de suite. Il est possible de créer de très longs registres. Quelle que soit

saires pour le commander. Si le nombre de registre est vraiment important, il sera judicieux de pla
suivant. Dans beaucoup d'afficheurs à LED, on trouve des 74HC245 qui jouent ce rôle. Il s'agit de pas

Registres à sorties à courant constant

Bien entendu, les sorties du 74HC595 sont des sorties C-MOS normales. Pour les utiliser pour com
résistance pour limiter le courant.

Il existe plusieurs circuits registres série-parallèles dont les sorties incorporent des sources de cour
simplifie le schéma :



La valeur du courant dans toutes les sorties est fixé par une seule résistance, notée R sur le schéma.

Plusieurs fabricants proposent différents modèles de registres série-parallèle avec sources de coura
différents, mais qui ont en commun leur brochage. Toshiba propose le TB62701, Texas Instrument l
dont je n'ai trouvé la documentation... qu'en chinois !

Programmation

La procédure pour envoyer 8 bits dans notre registre série parallèle est constituée d'une boucle *fo*
transmettre est d'abord placée sur une sortie, puis un flanc montant est produit sur l'horloge. L'hor
Tout à la fin, une impulsion est envoyée sur l'horloge parallèle.

```
#define SortieSerieOn P1OUT |= (1<<0)
#define SortieSerieOff P1OUT &=~(1<<0)

#define ClockSerHaut P1OUT |= (1<<1)
#define ClockSerBas P1OUT &=~(1<<1)

#define ClockParHaut P1OUT |= (1<<2)
#define ClockParBas P1OUT &=~(1<<2)

void Envoie8bitsSerie (uint8_t valeur) {
  uint16_t i;
  for (i=0; i<8; i++) {
    if (valeur & (1<<i)) {
      SortieSerieOn;
    } else {
      SortieSerieOff;
    }
    ClockSerHaut; ClockSerBas;
  }
  ClockParHaut; ClockParBas;
}
```

Cette procédure devient souvent la procédure centrale dans un programme qui gère un afficheur à l
même jusqu'à regarder comment le compilateur l'a traduite en instructions assembleur, pour cherch

Voici par exemple une version plus optimisée de l'envoi des bits :

```
...
  for (i=0; i<8; i++) {
    if (valeur & (1<<0)) {
      SortieSerieOn;
    } else {
      SortieSerieOff;
    }
    ClockSerHaut; ClockSerBas;
    valeur = valeur >> 1;
...
```

Cette version supprime l'opération `valeur & (1<<i)`, qui prend davantage de cycles d'horloge du mi

Ce n'est qu'un petit exemple d'optimisation. Une autre technique sera décrite plus loin dans ce co
Dans ce cas, ce n'est plus le microcontrôleur qui va effectuer le travail d'envoi des bits, mais un contr