# TP : A L'INTÉRIEUR D'UN PROCESSEUR

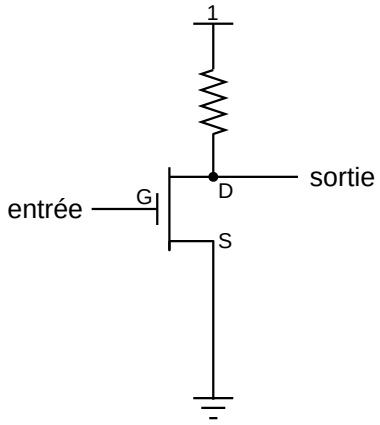
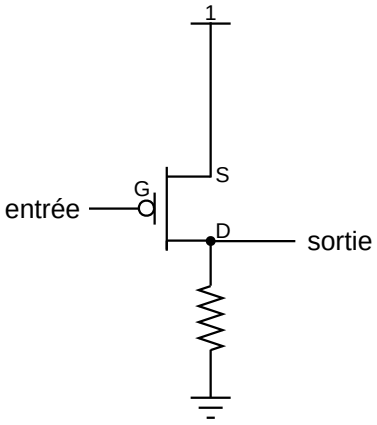
Un processeur est constitué d'un très grand nombre de transistors interconnectés (environ un milliard pour les processeurs actuels). Chacun de ces transistors fonctionne comme un interrupteur commandable électriquement. Or il est difficile de concevoir comment, en assemblant de simples interrupteurs, on peut obtenir quelque chose d'aussi puissant et compliqué qu'un processeur !
Le but de ce TP est donc d'approcher un tout petit peu ce grand mystère en réalisant une « Unité Arithmétique et Logique » très rudimentaire.

## *I) La brique de base : le transistor MOSFET*

Ces transistors ont 3 broches, dont l'une (la grille G) permet de commander le passage du courant entre les deux autres (le drain D et la source S).
La grille est isolée du reste du transistor et n'est donc traversée par aucun courant.

Il y a deux sortes de transistors MOSFET, ceux de type N et ceux de type P, dont la structure interne et le fonctionnement sont symétriques.

Comme on cherche ici à implémenter des fonctions booléennes, on ne considérera que deux niveaux de tension : le niveau bas noté 0 (0V) et le niveau haut noté 1 (en général 5V).

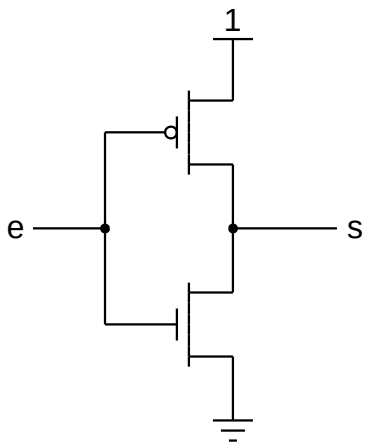
| Transistor de type N | Transistor de type P (petit rond sur la grille) |
|---|---|
|  |  |
| Si l'entrée est à 0, le transistor est bloqué<br>Si l'entrée est à 1, il est passant | Si l'entrée est à 0, le transistor est passant<br>Si l'entrée est à 1, il est bloqué |

**A faire :**
- Afficher la simulation ci-contre : Transistors MOSFET de type N et P
- Bien observer le comportement des deux types de MOSFET en modifiant le curseur « Niveau d'entrée » (tout à fait à droite de l'écran).
- Dans les deux cas, quelle porte logique très simple obtient-on ?

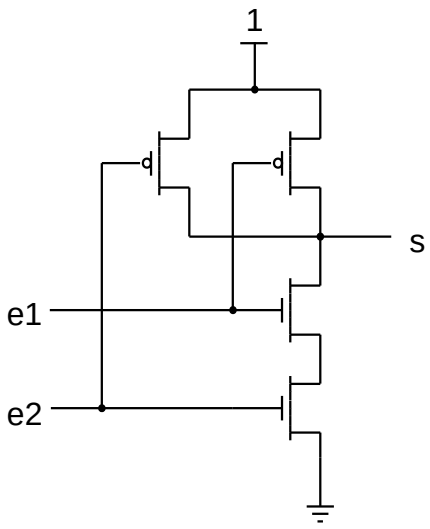
# *II) Faire une porte logique en assemblant des transistors*

Les 5 montages ci-dessous correspondent à des portes logiques différentes.

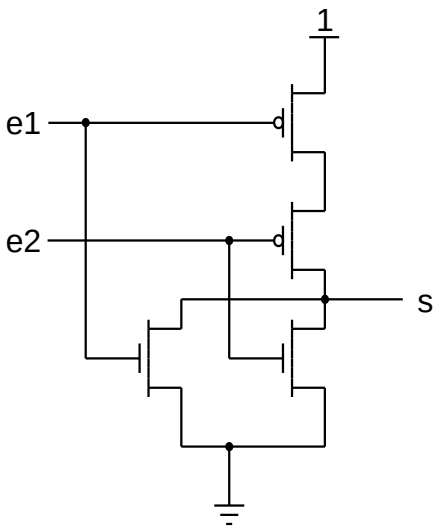
Montage 1



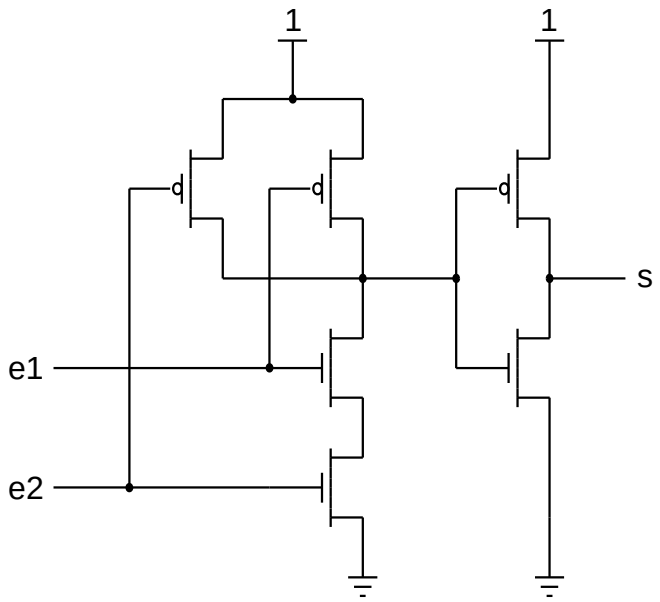

Montage 2



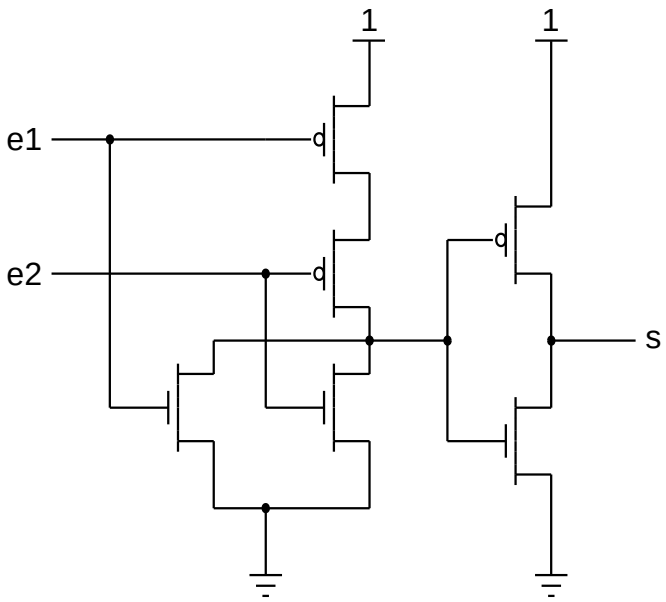

Montage 3




Montage 4




Montage 5




**A faire pour chacun des 5 montages :**

- Pour chaque combinaison d'entrées du montage, essayer de prévoir l'état de chaque transistor (bloqué ou passant) et en déduire la sortie du montage.
- Vérifier vos hypothèses grâce aux simulations ci-dessous (cliquer sur les entrées des montages) :
  Montage 1 Montage 2 Montage 3 Montage 4 Montage 5
- Quelles portes logiques ces montages permettent-ils d'obtenir ?

**Remarque :**

- Dans les deux montages de la page précédente, on a utilisé une résistance. Malheureusement, ces résistances chauffent et dissipent de l'énergie.
- Dans les montages ci-dessus, on a réussi à éviter l'usage de résistances en associant de façon complémentaire des transistors de type N et P. On parle de technologie CMOS (C pour Complementary). Sur une « verticale » de l'un de ces montages, soit ce sont les transistors N qui sont bloqués, soit ce sont les transistors P, mais dans tous les cas, le courant qui traverse est très faible.

# *III) Faire un additionneur en assemblant des portes logiques*

## 1) Demi additionneur 1 bit

- Compléter le tableau ci-dessous :
  Attention le + désigne ici l'addition et non l'opérateur booléen « ou »

| e1 | e2 | e1 + e2 |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

- On remarque que la valeur de e1 + e2 s'écrit sur 2 bits.
  Le bit des unités s'appellera dans la suite « s » (comme « sortie ») et le bit des « deuzaines » s'appellera « cout » (comme « carry out » c'est à dire « retenue sortante »).
  A l'aide de 2 portes logiques, proposer un montage permettant de calculer e1 + e2.
  Ce montage aura deux entrées (e1 et e2) et deux sorties (s et cout).

  Vérifiez votre proposition : simulation d'un demi additionneur 1 bit

- Compléter l'addition binaire ci-dessous en observant la propagation de la retenue :

```
  1   0   1   1
          1   1
 ---------------
```

  On remarque que pour ajouter deux nombres de n bits, on effectue n petites additions à 3 entrées : 1 bit du premier nombre + 1 bit du second + 1 bit de la retenue de l'addition précédente.
  Le demi additionneur proposé ci-dessus n'a que 2 entrées et ne permet pas de faire ces petites additions. Il va donc falloir lui ajouter une 3ème entrée.

## 2) Additionneur complet 1 bit

- Compléter la table de vérité ci-dessous : (« cin » est la retenue qui vient de l'addition précédente et « cout » est la retenue à reporter dans l'addition suivante)

| cin | e1 | e2 | cout | s |
|---|---|---|---|---|
| 0 | 0 | 0 | | |
| 0 | 0 | 1 | | |
| 0 | 1 | 0 | | |
| 0 | 1 | 1 | | |
| 1 | 0 | 0 | | |
| 1 | 0 | 1 | | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

- En observant la table de vérité ci-dessus, compléter :

  s = (e1 $\oplus$ e2) …… cin

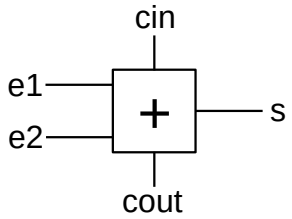
  cout = (e1 …… e2) + ( (e1 $\oplus$ e2) …… cin)

- A l'aide de 5 portes logiques, vous appuyer sur les deux expressions ci-dessus pour proposer un montage permettant de calculer e1 + e2 + cin.

  Vérifiez votre proposition : simulation d'un additionneur complet 1 bit

## 3) Additionneur 4 bits

- On décide de schématiser l'additionneur complet 1 bit ainsi :




Proposer un montage permettant d'additionner 2 nombres de 4 bits en enchaînant 4 additionneurs complets 1 bit. (Prenez un brouillon, posez des additions avec des nombres de 4 bits, observez comment les retenues se propagent de bit en bit… et ne regardez pas la solution ci-dessous tant que vous n'avez pas trouvé ;-)

Vérifiez votre proposition : simulation d'un additionneur 4 bits

## 4) Deux autres exemples

- Proposer un montage faisant le complément à 2 d'un nombre de 4 bits pour obtenir son opposé. On utilisera pour cela des inverseurs et des demi-additionneurs 1 bits.

  Vérifiez votre proposition : simulation de l'opposé d'un nombre 4 bits

- Pour réaliser un soustracteur 4 bits, on peut observer que : $a - b = a + (-b) = a + \overline{b} + 1$
  (ou $\overline{b}$ désigne le nombre b dont on a inversé tous les bits).
  On peut donc réutiliser un additionneur 4 bits en inversant au préalable tous les bits de b et en reliant le cin du premier étage à 1.

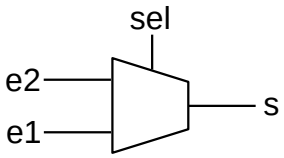
  Vérification : simulation d'un soustracteur 4 bits

# *IV) Faire un multiplexeur*

Les multiplexeurs sont des circuits très utilisés qui ont plusieurs entrées mais une seule sortie, et qui permettent de sélectionner laquelle des entrées doit être transmise à la sortie (comme un aiguillage).

## 1) Multiplexeur 1 bit – 2 entrées vers 1 sortie

- Schéma :




Fonctionnement :

Si **sel** est à 0, la sortie **s** renvoie la valeur de **e1**.
Si **sel** est à 1, la sortie **s** renvoie la valeur de **e2**.

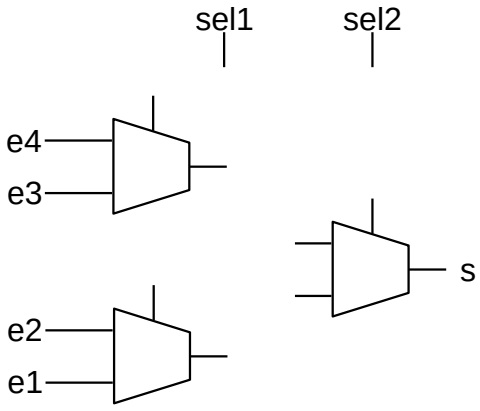
| sel | s |
|---|---|
| 0 | e1 |
| 1 | e2 |

- A l'aide de 2 portes « ET », 1 porte « OU » et 1 inverseur, proposer un montage permettant de réaliser le multiplexeur ci-dessus.

  Vérifiez votre proposition : simulation d'un multiplexeur 1 bit - 2 entrées vers 1 sortie

## 2) Multiplexeur 1 bit – 4 entrées vers 1 sortie

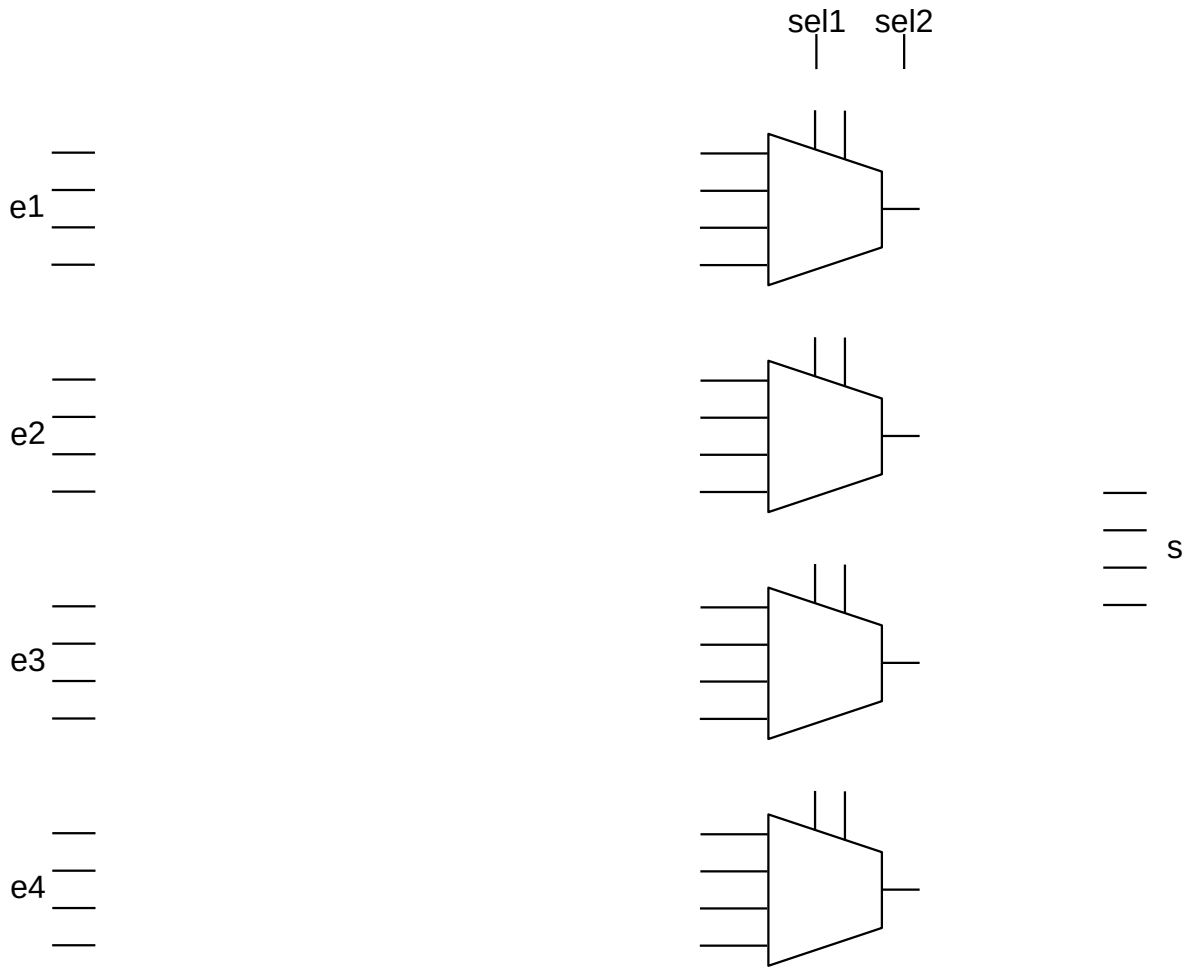
- Dans le schéma précédent, il y avait seulement deux entrées. Un bit de sélection suffisait donc pour préciser l'entrée que l'on devait retrouver en sortie. Si l'on veut maintenant pouvoir sélectionner une entrée parmi quatre, combien nous faut-il de bits de sélection ?
- Compléter le schéma ci-dessous permettant de réaliser un multiplexeur « 1 bit – 4 entrées vers une sortie » en utilisant 3 multiplexeurs « 1 bit – 2 entrées vers une sortie »




Vérifiez votre proposition : simulation d'un multiplexeur 1 bit - 4 entrées vers 1 sortie

## 3) Multiplexeur 4 bits – 4 entrées vers 1 sortie

- Compléter le schéma ci-dessous permettant de réaliser 1 multiplexeurs « 4 bits – 4 entrées vers une sortie » en utilisant 4 multiplexeurs « 1 bit – 4 entrées vers une sortie »




Vérifiez votre proposition : [simulation d'un multiplexeur 4 bits - 4 entrées vers 1 sortie]

## *V) Exemple d'UAL très simplifiée !*