Chapitre 1

Electronique Numérique & Logique : Introduction

Electronique à niveaux discrets : 0 et 1 (0V et 5V)

L'électronique numérique et logique manipule les mêmes signaux que l'électronique analogique (tensions, courants), mais ces signaux sont discrets. Par signaux discrets, on entend des signaux qui ne peuvent prendre que des valeurs disjointes clairement séparées, par opposition aux signaux continus qui peuvent prendre des valeurs quelconques arbitrairement voisines les unes des autres, un peu comme les entiers sont discrets par opposition aux nombres réels qui sont continus.
A priori on s'intéresse à l'électronique binaire : les signaux ne peuvent prendre que deux valeurs, 0 ou 1. Dans les circuits physiques, ces deux valeurs sont représentées par des niveaux de tension bien séparés, par exemple 0V et 5V. Le système binaire présente de nombreux avantages. Les 4 opérations (+, -, x, /) y sont simples, on peut cabler un système binaire à l'aide de simples relais ouverts ou fermés (c'est d'ailleurs sous cette forme mécanique que les premiers ordinateurs ont été réalisés), et l'existence de 2 niveaux seulement minimise le nombre de "frontières", donc le nombre d'erreurs possibles, ce qui est impératif pour pouvoir disposer de calculateurs permettant de réaliser des calculs arbitrairement longs.

Les circuits électroniques numériques et logiques utilisent des composants identiques à ceux de l'électronique analogique, mais en régime *non linéaire* (commutation). Ces composants sont des transistors (d'où l'appellation parfois rencontrée de "Total Transistor Logic") et permettent de réaliser des interrupteurs, des diodes, des résistances, etc.
L'éléments binaire de base s'appelle le Binary Digit (BIT) ; il peut prendre deux valeurs : 0 ou 1. Il peut être interprété en tant que nombre binaire (écrit en base 2, contenant uniquement des 0 et des 1), et cela donne lieu à l'application numérique de cette électronique (Exemple : fonction addition).
Il peut être interprété en tant qu'élément de choix : soit l'un (0) soit l'autre (1), et cela donne lieu à l'application logique de cette électronique. Les circuits de base (comportant un nombre réduit de transistors) de l'électronique numérique et logique réalisent des opérations logiques simples : ET (AND), OU (OR), NON (NOT), XOR, etc. A partir de ces briques de base de la logique, on peut ensuite réaliser toutes les fonctions logiques complexes. Ces opérations de base sont définies par leur table de vérité (un peu comme les opérations de calcul de base sont définies par les tables d'addition et de multiplication). Pour revenir à l'exemple de l'addition, il n'existe pas de circuit de base qui réalise l'addition mais on peut écrire l'opération "addition de deux bits" en fonction d'opérations logiques simples. On écrit en fait la table de vérité de l'addition. On aboutit ainsi à combiner les 2 approches et à considérer une Electronique Numérique & Logique, ou digitale.
"a bit", en anglais veut dire "un peu", et il faut effectivement de nombreux bits pour coder la moindre information. On manipule alors des "mots" binaires de plusieurs bits (8, 16, 32, 64, etc.).

Electronique Numérique

L'électronique numérique traite des nombres binaires (en base 2 : 1100011 = $2^0+2^1+...+2^5+2^6=99$). Si $A_1B_1C_1D_1$ et $A_2B_2C_2D_2$ représentent des nombres (de 4 bits : de 0 à 15), on peut calculer leur somme, leur différence, leur produit, etc. Les calculatrices, les ordinateurs, etc., sont basés sur l'utilisation de ces opérations.
Exemples de circuits numériques simples : le 74LS83 permet de réaliser l'addition de 2 nombres de 4 bits, le 74LS85 permet de comparer 2 nombres de 4 bits, (>, <, =), etc.

Electronique Logique

L'information binaire sert à représenter des Grandeurs binaires : Vrai ou Faux, Blanc ou Noir, Oui ou Non.... 1 ou 0. Cela permet par exemple le contrôle de processus à partir de capteurs qui donnent des informations sur l'état du système. Un exemple (très) simple : un plafonnier de voiture s'allume en fonction de l'information fournie par deux variables D et G qui représentent l'état d'ouverture ou de fermeture des portes. Si P est la variable qui détermine l'allumage de la lampe au plafond, l'équation logique donnant P est :
P = D ou G. A partir de cette équation, on représente le schéma symbolique de la fonction "plafonnier", puis son schéma électronique. Quelques exemples de fonctions logiques simples : boîtier de 4 portes ET à 2 entrées : 74LS08 ; boîtier de 4 portes OU à 2 entrées : 74LS32, etc.

De l'Analogique au Numérique/Logique

Le monde analogique et le monde digital

Les signaux du monde "réel" sont le plus souvent analogiques, c'est à dire qu'ils peuvent prendre n'importe quelle valeur (ce n'est plus vrai au niveau microscopique où les effets quantiques se font sentir). On dit aussi qu'ils sont continus, et en l'absence de bruit, entre 2 valeurs, on peut toujours en intercaler une troisième. A l'opposé, le monde digital manipule des signaux qui ne peuvent pas prendre n'importe quelle valeur, un peu comme l'ensemble des nombres entiers où l'on ne trouve pas de "signal" entre 2 et 3.

Ce qui peut sembler une limitation ("blocage" du signal digital) apporte aussi un avantage énorme pour la transmission et le traitement des données : si le signal ne peut pas prendre toutes les valeurs possibles mais seulement certaines bien définies au départ, alors on pourra plus facilement savoir quand une erreur est intervenue, si le signal ne "tombe" pas dans les bornes autorisées. Dans la série télé culte "Le prisonnier", le héros se rebelle souvent en criant "je ne suis pas un numéro, je suis un homme libre !" ; les signaux digitaux sont souvent comme des numéros, et n'ont pas la liberté de varier, contrairement aux signaux analogiques.

En fait, si on y regarde de plus près, la "liberté" des signaux analogiques est une illusion. La présence de bruit superposé au signal empêche l'utilisateur de déterminer si le signal qu'il utilise a telle valeur précise. En électronique comme en physique, on détermine une mesure avec une certaine incertitude. Il est alors illusoire de chercher a distinguer deux valeurs qui ne sont pas suffisamment séparées l'une de l'autre car le bruit les "mélange", de même qu'il est impossible de chercher ce qu'il y a entre 2 et 3 dans le monde des entiers. Ce flou omniprésent dans le monde analogique permet d'établir un "pont" entre l'analogique et le numérique, et de "modéliser" le monde analogique par sa représentation numérique.

De l'Analogique au Digital : les 2 critères

Le traitement des signaux digitaux se fera toujours sur des signaux en tension. Si le signal initial n'est pas une tension, on utilise un capteur qui transforme de manière continue le signal mesuré (pression, température, vitesse, etc.) en tension. Le passage d'une tension analogique (continue) à sa version digitalisée (numérisée) se fait par l'intermédiaire d'un circuit appelé Convertisseur Analogique Digital (ADC en anglais). La réalisation de cette opération se fait en deux temps : échantillonnage puis numérisation, et doit respecter deux critères.

Critère temporel - Cadence d'échantillonnage - Théorème de Shannon

Le signal numérique doit être estimé suffisamment souvent : si la tension d'origine varie entre deux échantillons, on perd de l'information. Inversement, si la tension d'origine ne varie pas, il est inutile de prendre de nouveaux échantillons. En se plaçant dans l'option la plus simple où on ne fait pas varier la cadence d'échantillonnage, on voit qu'on est limité par les parties du signal qui varient le plus vite (à plus haute fréquence) et qu'il faut choisir une cadence d'échantillonnage qui permette de "passer" les parties HF du signal. On s'appuie pour cela sur un théorème mathématique qui stipule qu'on ne perd pas d'information en remplaçant une fonction continue par une série d'échantillons régulièrement espacés d'un intervalle τ_E où $\tau_E < 1/2.F_{max}$, F_{max} étant la fréquence maximum dans le spectre de la fonction.

Critère en tension - Dynamique - Ajustement du rapport Signal/Bruit

On suppose que l'échantillonnage de la tension initiale est réalisée à une cadence suffisante, chacun des échantillons permet de garder la tension à une valeur fixe pendant que l'ADC en fournit une représentation numérique. Le passage de la tension analogique à sa représentation digitale se fera sans perte d'information si le niveau minimum entre deux valeurs significatives est le même des deux cotés du circuit. Du coté digital, il s'agit du LSB, de l'écart entre un nombre et son plus proche voisin ; du coté analogique, il s'agit de la valeur rms du bruit superposé au signal. De même qu'il n'y a pas d'information entre deux nombres entiers, il n'y a pas d'information entre deux valeurs de tension analogique qui ne sont pas espacées d'une quantité supérieure au bruit (on ne peut pas les distinguer). Si ΔV est l'amplitude du signal analogique d'entrée, avec un bruit σ, et Nmax le nombre

maximum de la valeur numérique en sortie ($N_{max}=2N_B$, où N_B est le nombre de bits du convertisseur), on doit prendre $N_{max}>\Delta V/\sigma$.

Conversion Analogique/Digitale (CAD, ADC en anglais)

Un ADC est un circuit spécifique qui comprend une entrée analogique et une sortie numérique sur N_B bits, selon les critères exprimés ci-dessus. Généralement, N_B est une puissance de 2 ; à chaque valeur de tension d'entrée Vi, le convertisseur fait correspondre un nombre entier binaire compris entre 0 et N_{max}.
Ce montage a une conséquence sur l'architecture des circuits numériques. Si le signal analogique n'a généralement besoin que de 2 fils pour se propager, le signal numérique (sur N_B bits) utilise autant de fils que de bits, pour transporter la même information que le signal analogique. On verra dans la suite de ce cours que cette apparente complexité supplémentaire offre de nombreux avantages. D'une part les signaux numériques sont quasiment insensibles au parasites, d'autre part, le signal numérisé peut être stocké, et peut être soumis à des calculs.

Chapitre 2

Circuits logiques - Caractéristiques électriques

Constitution des circuits logiques - Commutation

Les circuits numériques et logiques véhiculent des signaux électriques binaires qui ne peuvent prendre que 2 états qu'on appelle états Haut (H) et Bas (L). Ces états sont codés par des niveaux de tension distincts ; en logique dite positive, l'état Haut (H) correspond au niveau de tension le plus élevé : 5 V, et l'état Bas (L) au niveau de tension le plus bas : 0 V.
NB: en logique négative, c'est l'inverse (état H = 0 V / état L = 5 V).
Note: les contraintes de miniaturisation et de consommation des circuits numériques actuels poussent à utiliser des niveaux de tension de moins en moins élevés (3.3 V, et encore moins, jusqu'à 1.8V), mais sauf exception, on continuera à travailler ici en 0/5 V. De toute façon, tant qu'on garde la notation H/L, on effectue les opérations numériques et logiques sans se préoccuper des niveaux de tension utilisés.

Les circuits logiques sont constitués de transistors, qui fonctionnent en commutation. L'électronique logique utilise exclusivement les deux états que l'électronique analogique d'efforce d'éviter : le blocage et la saturation. Les circuits logiques sont en fait constitués d'interrupteurs.
Plusieurs familles technologiques sont disponibles. A base de transistors bipolaires, qui fonctionnent en régime saturé/bloqué, on est dans le domaine de la TTL (Total Transistor Logic). A base de transistors à effet de champ MOS-n et MOS-p, on utilise des montages complémentaires “n-p”, et on est dans le domaine de la logique CMOS.

Le circuit logique le plus simple que l'on puisse concevoir est le circuit inverseur ; on peut le réaliser avec un seul transistor. On peut le cabler très simplement sur une plaquette d'essai. Quand la tension d'entrée vaut 5V ("1" logique) la tension de sortie vaut 0V ("0" logique). En fait, le plus simple des circuits logiques existant dans le commerce comporte déjà un nombre respectable de transistors. La porte logique universelle est la porte NAND car elle permet de réaliser toutes les autres fonctions logiques.

Transmission du signal digital

Marge de bruit statique
Un des intérêt de l'utilisation du signal binaire consiste en la grande différence de tension existant entre le niveau bas et le niveau haut. Le signal est alors protégé contre la plupart des parasites. De plus, pour que la transmission du signal digital soit toujours interprétée sans erreur, il existe une convention qui établit les marges de valeur des tensions d'entrée et de sortie affectées au niveaux L et H. Par exemple, en logique TTL (0-5V), le constructeur d'un circuit logique garantit qu'en sortie de ses circuits, un niveau L sera codé par une tension V_{OL} comprise entre 0 et 0.4V ; en

même temps, il est garanti que le signal sera interprété en entrée du circuit suivant comme un L si la tension correspondante V_{IL} est située entre -0.4 et 0.8V. La marge de bruit statique correspondante est donc de 0.4V. Le même genre de convention est adopté pour le niveau H : en sortie, la tension V_{OH} est garantie d'être comprise entre 2.4V et 5V, alors qu'un signal compris entre 2 et 5.5V sera interprété en entrée (V_{IH}) comme un H. La marge de bruit est également de 0.4V.
Les conventions diffèrent selon la technologie du circuit (TTL ou CMOS) mais l'idée reste la même : la gamme des valeur de tensions affectée à un niveau en sortie est incluse dans la gamme des valeurs de tensions interprétées en entrée du circuit suivant pour ce niveau.

Marge de bruit dynamique
Il peut arriver que le signal présente des parasites d'amplitude supérieure à la marge de bruit de 0.4V. Cependant, si ce parasite dure suffisamment peu de temps, il se peut que le circuit n'ait pas le temps de réagir (à cause de capacités parasites essentiellement). On obtient alors ce qu'on nomme une marge de bruit dynamique qui peut être supérieure à la marge de bruit statique.

Caractéristiques électriques des circuits logiques

Les circuits logiques sont trop souvent considérés comme des "boites noires" sans prendre en compte leurs caractéristiques électriques. De nombreux problèmes peuvent être évités si on respecte ces caractéristiques.
La sortie d'un circuit logique TTL est quasiment toujours constituée d'un étage dit Totem-pole, constitué de deux transistors en montage push-pull. Au niveau H, la sortie débite un courant I_{OH} dans l'entrée du circuit suivant qui doit le consommer (I_{IH}). A l'état bas, c'est l'inverse : la sortie du circuit précédent consomme un courant I_{OL} qui doit être fourni par le circuit suivant (I_{IL}).
Une conséquence de la structure comparée des étages d'entrée et de sortie des circuits logique est qu'une entrée non connectée (en l'air) est interprétée comme étant à l'état H. Il est donc vivement conseillé de toujours cabler explicitement les entrées non utilisées à 5V (de préférence via une résistance de 1kΩ) ou à la masse selon le niveau logique que l'on veut y fixer. Une entrée en l'air sera a priori interprétée à H, mais elle sera également très sensible aux parasites, surtout dans les circuits séquentiels où intervient une horloge, dans les compteurs, processeurs, etc.
Dans les familles de circuits utilisant la technologie CMOS, les courants d'entrée sont différents, du fait de la très grande impédance des grilles de transistor MOS.
La fonction de transfert d'un circuit logique donne la caractéristique $U_S(U_E)$ de la porte concernée. A partir de 0V, si U_E augmente, jusqu'à 0.8V il ne se passe rien (l'entrée suivante "voit" un état L), puis survient une zone instable où la sortie est indéterminée. Cette zone est "interdite" car la sortie peut alors présenter des oscillations (passage 0/1/0/1...). Enfin, lorsque le signal U_E dépasse 2V, l'étage suivant "bascule" et "voit" un niveau H. Il est donc dangereux en logique d'utiliser des signaux qui varient trop lentement en fonction du temps. Pour se protéger contre cet effet, on utilise des circuits particuliers appelés *trigger de Schmidt* qui possèdent un effet d'hystérésis et ne basculent pas sur le même niveau selon que le signal d'entrée augmente ou diminue.

Lorsqu'une les sorties de deux circuits logiques sont reliées entre elles, si elles ne prennent pas les mêmes niveaux à tout instant, il peut arriver qu'un niveau H (5V) soit relié à un niveau L (0V), créant un court-circuit. Un court-circuit est toléré mais pendant un temps réduit (une seule sortie à la fois et pendant moins d'une seconde). La conséquence de court-circuits trop fréquents dans un montage mal conçu par exemple, sera de faire vieillir prématurément le montage, voire d'entraîner sa destruction !
La consommation d'un circuit va déterminer le dégagement de chaleur en fonctionnement. Elle s'obtient par le produit de la tension d'alimentation (toujours la même, fixée à 5V, 3.3V, etc.) par le courant d'alimentation I_{CC}. Elle varie selon la famille technologique du circuit et est le plus souvent de l'ordre de quelque dizaines de mW par circuit. La consommation d'un circuit logique n'est pas la même dans l'état H et dans l'état L. Elle peut atteindre et dépasser le Watt pour des circuits spéciaux comme les microprocesseurs. La consommation TTL est indépendante de la fréquence de fonctionnement jusqu'à un certain point où les capacités parasites se font sentir ($1/C\omega$ devient trop faible). Par contre dans les circuits CMOS, la consommation dépend très vite de la fréquence car cette famille de circuits comporte des capacités parasites omniprésentes (l'entrée d'un transistor MOS, sur une grille isolée, est une capacité !).
Lorsqu'un circuit logique commute, sa sortie change brusquement d'état, et peut être amenée à consommer brutalement un courant important. Pour éviter que cet appel de courant fasse chuter la tension d'alimentation au niveau du circuit, il est recommandé de monter en parallèle des entrées d'alimentation 0/5V un condensateur de découplage qui fournira le courant en cas de transition à haute fréquence ; ce condensateur joue le rôle de filtre passe bas sur l'entrée d'alimentation.
Lorsqu'on veut protéger un point d'un montage (entrée ou sortie) contre des valeurs de tensions susceptibles d'endommager des circuits, on peut installer en parallèle des diodes de protection au 5V ou à la masse.

Entrance et Sortance (fan in / fan out)

Lorsqu'on branche plusieurs circuits en sortie d'un même premier, la loi de conservation du courant entraîne des règles sur le nombre maximum de circuits qu'on peut brancher en parallèle en sortie d'un premier. On définit ainsi la sortance à l'état H : I_{OH}/I_{IH}, à l'état bas : I_{IL}/I_{OL}. Pour les circuits TTL, elle est typiquement de l'ordre de 10. Par exemple, pour le circuit 74LS00 : I_{OH} = -0.4 mA / I_{IH} = 20 µA -> N_H = 80. I_{OL} = 8 mA / I_{IL} = -0.4 mA -> N_L = 20. La sortance effective du LS00 est de 20

Nomenclature des circuits logiques

Selon la technologie utilisée dans la réalisation du circuit, on le repère par une série de lettres. De même, dans une famille donnée, les circuits peuvent posséder des caractéristiques différentes (optimisation en vitesse, en faible consommation, etc.).

Famille technologique	Type
TTL	F, LS, ALS

ECL
CMOS 4000, HC, HCT

Dans une famille et un type donné, les fonctions logiques réalisées sont ensuite repérées par des numéros (00, 01, 02, etc.)
Avant la mention du type dans la famille, on trouve en TTL un numéro supplémentaire qui indique si le circuit répond aux normes civiles (74) ou militaires (54).
On peut également trouver une lettre supplémentaire qui indique le type de boitier du circuit : boitier plastique (N) ou céramique (J, qui possède des propriétés thermiques différentes), boitier miniaturisé (D). Au total, on pourra trouver une référence telle que : SN 74 ALS 00A N

SN : Famille standard
74 : circuit civil (0 à +70°C) / 54 : gamme militaire (-55 à +125°C)
ALS : sous famille de TTL
00 : Numéro d'immatriculation de la fonction (voir répertoire)
00-A : Advanced = plus rapide que le ALS 00 normal
N : type de boitier (ici : plastique) ; existe aussi en céramique (J), etc.

On distingue également les circuits par leur densité d'intégration, c'est à dire le nombre de transistor que l'on peut implanter par mm^2 de silicium : SSI : quelques portes / circuit ; MSI : 10 - 100 portes / boitier ; LSI : 100 - 1000 portes / boitier ; VLSI : > 1000 portes / boitier ; Mémoires, processeurs : > 10^6 cellules / boitier

Paramètres temporels

La rapidité d'un circuit est déterminée par son temps de transfert. Ce temps peut être différent selon que la commutation a lieu de H vers L (t_{PHL}) ou de L vers H (t_{PLH}). La relation consommation/rapidité détermine l'usage du circuit ; le circuit idéal bascule vite sans consommer de courant. On peut ainsi comparer les caractéristiques des différentes familles de circuits.
Le temps de transfert de toute une chaine de circuits s'obtient par la somme de tous les temps de transferts des circuits de la chaine. lorsqu'on a affaire à deux chaines fonctionnant en parallèle, il est important d'étudier les temps de propagation respectifs des deux chaines pour repérer d'éventuels aléas de fonctionnement.

Lecture d'un schéma de circuit logique

La documentation de chaque circuit logique comporte un certain nombre d'informations standardisées qu'il est essentiel de savoir rapidement décoder. On trouve notamment le schéma de cablage du circuit, avec la liste des connexions indiquant les entrées et les sorties, et la sortance correspondante. Les entrées/sorties indiquées de manière inversée (/E ou /Z) sont actives à l'état bas (L). L'action des entrées et les valeurs des sorties correspondantes est indiquée dans une table de vérité. Un schéma logique à base de portes élémentaires complète cette information et permet de comprendre comment fonctionne le circuit (même si le schéma réel du circuit est plus complexe). Enfin, la documentation donne les caractéristiques électriques du circuit qui sont essentielles pour pouvoir réaliser le montage physique sur table.

Conception d'un circuit logique

La conception d'un montage logique doit respecter un certain nombre de règles et contraintes visant à assurer le bon fonctionnement du montage dans tous les cas possibles. Autrement dit, il faut toujours prévoir le pire (règle du "worst case design"), et faire en sorte que le circuit réalisé fonctionne, même si les composants utilisés ont des caractéristiques à la limite de leur gamme de variation garantie. Par exemple, les composants seront choisis en fonction de l'opération logique souhaitée, de la simplicité du circuit (la simplicité implique moins de pannes et une plus grande facilité de réparation), de la consommation des circuits (qui va déterminer le calibre de l'alimentation nécessaire, le diamètre des fils, la quantité de chaleur dégagée donc la nécessité éventuelle de prévoir un ventilateur de refroidissement, etc.), de leur rapidité et ...de leur coût !

Chapitre 3

Collecteur ouvert - Haute impédance - Bus de données -

Etage de sortie à collecteur ouvert

Schéma électrique de sortie de porte TTL
Quelle que soit la porte logique considérée (ET, OU, NAND, etc), le schéma électrique de l'étage de sortie est toujours le même : un étage "totem-pole" comportant essentiellement deux transistors qui ne sont jamais passant en même temps. Il existe un autre type de montage de l'étage de sortie où seul le transistor cablé à la masse (transistor "du bas") est présent. Le signal de sortie est à prendre sur le collecteur de ce transistor, d'où l'appellation de *collecteur ouvert* pour un tel montage. Ce transistor se comporte comme un interrupteur à la masse, ouvert ou passant, correspondant respectivement aux états de sortie H ou L, dans le montage totem-pole. Pour retrouver un signal logique 5V/0V en sortie du montage à collecteur ouvert, il faut rajouter une résistance de tirage à 5V. Il existe quelques circuits logiques possédant une version avec sortie à collecteur ouvert (CO), la fonction restant la même mais la sortie est modifiée. Ces versions CO sont repérées par un astérisque '*'. Ainsi le circuit -00 est un boitier de portes NAND, et le circuit -01* est la version CO du même boitier.

Applications "analogiques" du montage à collecteur ouvert
Il est fréquent de vouloir visualiser le niveau de signal logique en un point du circuit par l'utilisation d'une diode électroluminescente (LED). Par contre il est déconseillé de brancher une telle diode directement en sortie d'une porte logique, surtout si celle-ci attaque une autre porte. Le niveau de courant qui passe dans une LED est très différent de celui pour lequel les portes logiques sont prévus, et le fonctionnement du montage pourrait s'en trouver perturbé. Une solution peut être de placer un montage à collecteur ouvert pour piloter la LED. On placera une résistance de protection d'environ 300Ω en série avec la diode pour la protéger. Il existe des boitiers contenant 6 inverseurs avec sortie en collecteur ouvert (74LS05) qui permettent de piloter plusieurs LED en même temps (pour visualiser un mot de 4 bits, par exemple). Cependant, on peut aussi bien utiliser un simple transistor npn 2N2222, si on doit piloter une seule charge.

Le circuit à collecteur ouvert peut être utilisé pour réaliser une conversion de niveau de tension affectée au niveau H. Par exemple, pour passer d'un circuit TTL (0-5V) à un circuit CMOS (0-12V), on utilise la commande du transistor de sortie du montage CO pour ouvrir ou fermer un circuit à la masse et faire passer le point de collecteur de 0 (+/- la tension résiduelle de saturation) à 12V.
Si la base du transistor de sortie est commandée par des signaux carrés d'amplitude 5V, on retrouvera sur le collecteur des signaux carrés de même période mais d'amplitude différente, selon la tension d'alimentation en sortie. Les signaux carrés

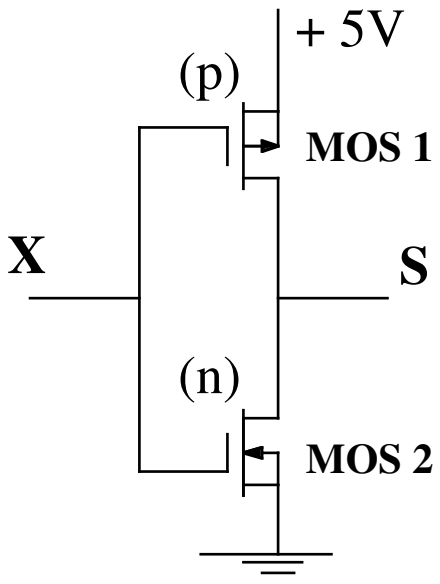
de sortie peuvent être déformés de manière dissymétrique à la montée et à la descente à cause de la capacité parasite que voit le collecteur au niveau H.

Bus de données à base de portes à collecteur ouvert

Dans un montage logique classique, on ne peut connecter entre elle deux sorties de circuits différents. En effet, lorsque des niveaux H et L seront présents simultanément, il se produira un court circuit qui peut détruire le montage. L'utilisation d'un point commun de sortie par l'intermédiaire de CO permet d'éviter cet inconvénient. Si on relie deux montages à CO en un même point, avec une résistance de tirage au 5V, ce point commun restera à l'état H tant que les deux CO seront bloqués. Si le premier des deux CO seulement devient passant, il court-circuitera la résistance de tirage à la masse, imposant par la même un niveau L en sortie. Le deuxième CO restant bloqué, cela n'aura pas d'influence sur cette partie du montage. On peut donc réaliser ainsi une mise en commun d'une ligne pour le signal de sortie de plusieurs montages. Dès que l'un des circuits passe à 0, la sortie est à 0. On nomme un tel montage un circuit un "ET cablé". C'est le principe du bus qui utilise un seul fil pour véhiculer le signal en provenance de différents circuits, pourvu que ceux-ci ne transmettent pas leur information simultanément. Le revers de l'utilisation du bus est l'obligation de séquencer le transfert d'information.

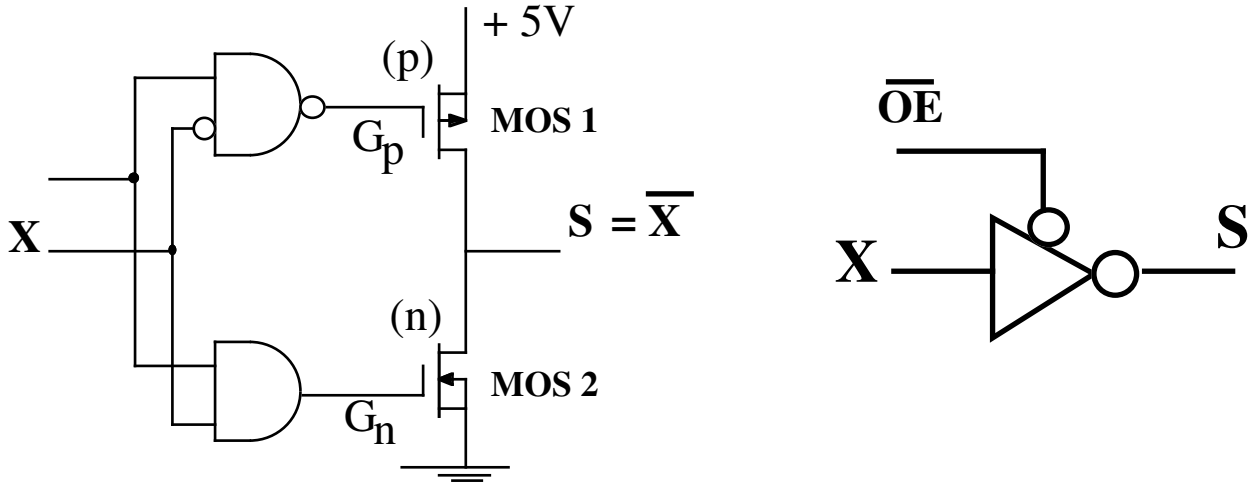
Une application informatique du ET cablé est le circuit de demande d'interruption (*interrupt request*) pour traiter une tache asynchrone urgente. Le microprocesseur surveille une ligne et une seule ; la détection d'un état L lui permet de savoir qu'un des circuits du montage demande une interruption.

Schéma électrique de sortie de porte logique : montage totem-pole



Que ce soit en technologie TTL ou CMOS, l'étage de sortie d'une porte logique est toujours le même : deux transistors (passant/bloqués en opposition) en totem-pole qui permettent de relier la sortie à 0V ou à 5V selon lequel des 2 transistors est passant, définissant ainsi deux états en sortie, comme dans la figure ci-contre.
L'état qui correspondrait au cas où les deux transistors seraient passant en même temps est évidemment impossible car il entrainerait un court-circuit 5V-0V. Par contre, on peut imaginer un troisième état où les deux transistors sont bloqués simultanément. Dans un tel état, le point de sortie se trouve à un potentiel flottant, comme s'il était déconnecté à la fois du 0V et du 5V. On

appelle un tel état *Haute impédance* car les deux transistors du totem présentent une très haute impédance entre la sortie et la masse et entre la sortie et le point 5V. Ce troisième état est noté 'Z'.
Le blocage simultané des deux transistors de sortie est commandé par une entrée supplémentaire souvent notée /E ou /OE (Output Enable), qu'on représente par une connexion supplémentaire sur le côté des triangles figurant les buffers (voir figure ci-dessous).



Sortie haute impédance et sortie verrouillée

Attention à ne pas confondre l'état haute impédance avec un état intermédiaire qui serait situé à 2.5V par exemple ! De même, on ne confondra pas l'état haute impédance avec l'état verrouillé (à 0 ou à 1, mais à un potentiel bien défini) commandé par l'entrée /E.

Etat électrique et paramètres temporels

Lorsqu'une porte se trouve en état haute impédance, l'état électrique de sa sortie n'est pas déterminé par la porte elle-même, mais par le ou les points auxquels la sortie est connectée.
L'existence d'un troisième état entraîne la définition de paramètres temporels supplémentaires : t_{PLZ} et t_{PZL}, t_{PHZ} et t_{PZH}, qui définissent les temps de passage des états définis (1 ou 0) à l'état Z et inversement.

Bus de données à base de portes 3 états

La possibilité d'isoler une sortie du fil sur lequel elle se trouve branchée lors de son passage en haute impédance, permet de faire travailler les circuits logiques à l'aide de bus. Dans un micro ordinateur, par exemple, le processeur partage un bus 32 bits (32 fils) avec la mémoire, les périphériques, etc. Lorsque la mémoire n'est pas sollicitée, elle se met en haute impédance et se trouve en quelque sorte déconnectée du bus, qui peut alors être utilisé par d'autres circuits.

Théoriquement, un seul circuit à la fois adresse le bus ; cependant, il peut arriver que deux circuits se retrouvent branchés sur le bus, parce que le second est déja actif tandis que le premier n'est pas encore arrivé en état Z (cela dépend des $t_{PL/HZ}$ et $t_{PZL/H}$ relatifs des deux circuits). Si les deux signaux de sortie sont contradictoires (0/1), on obtient alors un conflit de bus, qui déclenche une erreur (bus error).

Circuits de gestion de bus à 3 états

Dans un micro processeur, les différentes informations circulent sur différents bus. Les données manipulées circulent sur le bus de données qui peut comporter 32, 64 bits, et plus. Les données sont repérées par leurs adresses qui circulent sur le bus d'adresse, qui permet d'adresser 2Go par exemple si il comporte 32 bits. Enfin, l'ensemble du circuit est géré par un bus de contrôle qui détermine l'état des différents périphériques, qui transporte les demandes d'interruption, etc.

L'interface d'un circuit avec un bus se fait par l'intermédiaire de circuits "portail", comme le 74LS245. Il s'agit d'un circuit bidirectionnel, qui permet de faire transiter les données dans deux directions, en contrôlant la mise en troisième état de chacune des branches.

Trigger de Schmidt

La fonction de transfert d'une porte logique comporte une "zone interdite" où le signal de sortie est indéterminé. Si le signal d'entrée ne varie pas suffisamment franchement de 0 a 1 par exemple, le signal de sortie peut "hésiter" et présenter des oscillations. Pour éviter ce genre de problème dans le cas où les signaux risquent de varier trop "lentement" (charge de condensateur par exemple), on peut utiliser des circuits (inverseurs ou buffers) comportant un effet d'hystérésis, c'est à dire des seuils de déclenchement S_0 et S_1 ($S_1>S_0$) différents selon qu'ils sont franchis à la montée ou à la descente. Le signal doit franchir S_1 pour déclencher le passage L->H en sortie mais ensuite, il devra repasser en dessous de S_0 pour redéclencher le passage H->L. Un tel circuit à hystérésis se nomme *trigger de Schmidt* et permet de réduire les problèmes de bruit et d'oscillation dans certains montages critiques.

Chapitre 4

Fonctions logiques - Opérations - Circuits associés

Variables et fonctions logiques

Le signal manipulé par l'électronique digitale ne prend que 2 valeurs que l'on va interpréter dans ce chapitre du point de vue logique (V/F, O/N, etc.). Du point de vue électrique, il s'agit de valeurs de tensions (0/5 V), et du point de vue représentation, on adoptera la notation 0/1 par souci de simplicité. Dans un circuit logique, l'information consiste en signaux qu'on appelle des variables logiques, utilisées en entrée de fonctions logiques variées pour obtenir des résultats. L'action d'une fonction logique est définie par sa *table de vérité*, qui donne le résultat en sortie pour toutes les combinaisons possibles des variables d'entrée. Le terme "fonction logique" ne doit pas prêter à confusion, ni faire croire qu'on ne traitera dans ce chapitre aucun problème lié au numérique. Ainsi l'addition de deux bits peut être envisagé du point de vue logique, via sa table de vérité.

La logique combinatoire concerne des opérations logiques où la sortie dépend directement de l'état des entrées. Une table de vérité suffit à déterminer le fonctionnement d'une fonction logique combinatoire.
La logique séquentielle concerne des opérations logiques où la sortie dépend non seulement de l'état des entrées à un instant donné mais aussi de l'histoire de ces entrées, c'est à dire de la séquence des évènements qui se sont déroulés à l'entrée de la fonction logique considérée. C'est la logique séquentielle qui prévaut dans le fonctionnement des microprocesseurs, par exemple.

Fonctions logiques combinatoires élémentaires - Circuits associés

Inverseur

Entrée	Sortie
0	1
1	0

La plus simple des fonctions logiques (à part l'identité qui réalise Sortie=Entrée à l'aide d'un simple bout de fil) est la fonction inverseur, dont la table de vérité est donnée dans le tableau ci-contre.

L'inversion se note : S = NOT(E), ou S=/E.

ET logique (AND)

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

La fonction ET est une fonction de deux variables que l'on note : S = A.B, comme le produit en mathématiques, car la table de vérité (ci-contre) donne un résultat analogue. La fonction NOT(ET), notée /(AB) est le NAND.

OU logique (OR)

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

La fonction OU est une fonction de deux variables que l'on note : S = A+B, comme la somme en mathématiques, car la table de vérité (ci-contre) donne un résultat analogue, **sauf lorsque A=B=1**.
La fonction NOT(OR), notée /(A+B) est le NOR.

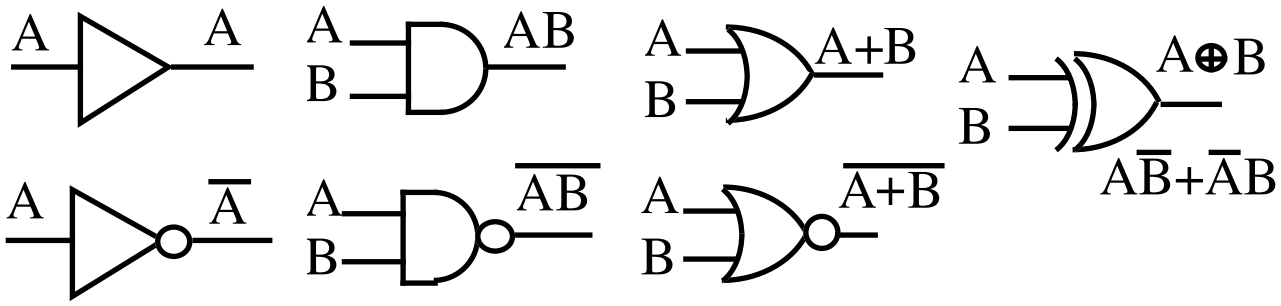
OU exclusif (XOR)

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

La fonction XOR est une fonction de deux variables que l'on note: S = A xor B. Sa table de vérité montre que le XOR vaut 1 lors de l'inégalité des deux variables d'entrée. La fonction NOT(XOR) donne 1 pour deux entrées identiques.

A partir de l'analyse de la table de vérité du XOR, on peut montrer que cette opération s'écrit : A xor B = A/B+/AB. Chacun des termes produits A/B et /AB est un *minterme* de la fonction XOR. Le résultat A xor B vaut 1 lorsque A vaut 1 ET B vaut 0 (c'est à dire /B vaut 1) OU lorsque A vaut 0 (c'est à dire /A vaut 0) ET B vaut 1, ce qui permet d'écrire l'identité ci-dessus.

Dans la plupart des documentations de circuits logiques, les fonctions logiques de base sont représentées à l'aide des symboles ci-dessous :



Les portes logiques de base sont disponibles dans les familles de circuits intégrés les plus courants : 00 : NAND, 04 : inverseur, 08 : ET, 32 : OU, 86 : XOR, etc.

Algèbre de Boole

En mathématique, il existe une algèbre pour calculer avec des variables prenant deux valeurs seulement (0/1), mise au point par le mathématicien Boole. Il y a quelques règles permettant de calculer plus simplement (identités remarquables) et si une incertitude subsiste, il est toujours possible de recourir à la table de vérité. Ainsi on a notamment :
1.X = X ; 0.X = 0 ; 1+X = 1 ; 0+X= X ; X+/X=1 ; X./X =0 ; X+X=X ; XA+X/A=X(A+/A)=X ; X+X/Y=X(1+/Y)=X, etc.

A noter particulièrement les égalités suivantes, qui constituent le théorème de Morgan : /(X.Y) = /X+/Y ; /(X+Y) = /X./Y ; on vérifie que le passage de la logique positive à la logique négative change le ET en NOR et le OU en NAND.

Une porte universelle : le NAND
Une des conséquences du théorème de Morgan est qu'on peut trouver une fonction de deux variables permettant de réaliser toutes les fonctions logiques imaginables. Cette fonction doit contenir l'inversion (irremplaçable par aucune autre fonction) et au choix, le ET ou le OU puisque ces deux opérations sont interchangeables via l'inversion et le théorème de Morgan. Techniquement, c'est le NAND qui est la porte universelle de base en logique positive. Pouvoir réaliser n'importe quelle fonction à base de NAND permet de n'avoir à garder en stock qu'une seule sorte de circuit et de pouvoir concevoir un petit circuit rapidement. Cependant, les circuits logiques sont aujourd'hui disponibles en grand nombre dans diverses sortes, ce qui peut permettre de gagner du temps de conception : au lieu d'imaginer comment réaliser un montage avec des NAND, on utilise directement le circuit tout fait.

Fonctions logiques complexes

Les circuits logique utilisent des fonctions standards qui se répètent souvent, même dans des montages ayant des finalités très différentes. Ainsi, on retrouve toujours des (dé)multiplexeurs, des (dé)codeurs, des compteurs, etc. Pour ces fonctions complexes là, il existe des circuits tout prêts dans le commerce, qu'on étudiera plus en détail plus loin dans le cours.
Cependant, il est nécessaire de savoir concevoir rapidement une fonction de 3 ou 4 variables à base de portes logiques simples, sans pour autant être obligé de chercher dans le catalogue un circuit tout fait (qui n'existe peut être pas d'ailleurs !).
La réalisation d'une fonction logique passe toujours par l'établissement de sa table de vérité qui permet de prendre en compte tous les cas possibles. On repère ensuite les cas où la sortie prend la valeur 1, et on identifie les associations des variables d'entrée qui correspondent à ces cas (les *mintermes*). On écrit enfin la fonction comme le OU (la somme logique) des mintermes ainsi identifiés, ce qui donne une première expression pas forcément la plus simple. On simplifie enfin la fonction grâce aux identités remarquables étudiées précédemment. Par exemple : ABC+AB/C=AB(C+/C)=AB.
Dans l'exemple ci-dessus, on a pu simplifier la variable C parce qu'elle apparaissait dans deux termes identiques à l'exception de C qui s'inversait d'un terme à l'autre. Il existe une méthode "graphique" permettant de repérer dans une fonction quelconque tous les termes donnant lieu à de telles simplifications. Elle est présentée dans le paragraphe suivant.

Table de Karnaugh

La méthode de la table de Karnaugh consiste à écrire la table de vérité de la fonction étudiée sous la forme d'un tableau à double entrée. Les entrées de ce tableau sont détaillées en prenant en compte tous les cas possibles des variables d'entrée en faisant en sorte que d'une case du tableau à sa voisine, une seule variable change (s'inverse) à la fois.

A\BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Le tableau ci-contre est la table de Karnaugh du circuit de vote majoritaire. On notera l'ordre dans lequel les cas **BC** sont listés.

Le repérage des simplifications dans la table de Karnaugh se fait par repérage des cases voisines contenant des 1. Dans le tableau précédent il y a 3 cas de regroupement, donc de simplification (cases grisées). On obtient : ABC+AB/C=AB ; ABC+/ABC=BC ; ABC+A/BC=AC. Et la fonction "vote" s'écrit R=AB+AC+BC : dès que deux personnes au moins se mettent d'accord, leur vote l'emporte. La dernière étape de la réalisation de la fonction consiste en son cablage qu'on représentera sous la forme la plus claire possible, avec des connexions à angle droit.

Fonctions logiques programmées :

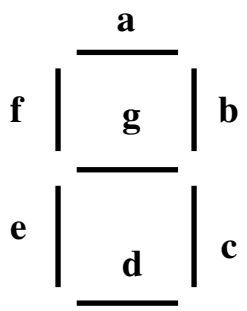
Multiplexeur/décodeur 1 parmi 8 : 74LS151
Ce circuit possède 8 entrées de données I_0-I_7 et 3 entrées d'adresse (ou sélection) $S_0S_1S_2$ qui permettent de choisir une donnée parmi 8 (3 bits permettent de coder 8 valeurs) et de la recopier de l'"aiguiller" vers la sortie Z. Par un choix des valeurs placées sur les entrées I_0-I_7, le circuit 74LS151 permet de réaliser le codage de n'importe quelle fonction de 3 variables. On notera également la présence de 'X' dans la table de vérité, qui signifient que le résultat ne dépend pas de la variable considérée. De plus, le circuit comporte une entrée *Enable* (notée /E) qui est active à l'état bas : lorsque E=H, la sortie du circuit est bloquée à la valeur L, quelles que soient les états des variables d'entrée ('X' partout).

(P)ROM
Une PROM est une mémoire programmable qu'on écrit une fois et qu'on ne peut plus effacer par la suite. Les entrées d'adresse de la prom permettent de choisir une case mémoire parmi plusieurs et ainsi de programmer une fonction d'autant de variables qu'il y a de bits d'adresse.

PAL, PLA, PLD
Les PAL/PLA/PLD sont des circuits constitués de matrices de portes ET et OU dont les interconnexions sont laissées à la disposition de l'utilisateur. Ils permettent de cabler des circuits à la demande.

Un exemple de circuit complexe : Décodeur BCD/7 segments (74LS47)

L'interface homme/machine passe en particulier par l'affichage de nombres, résultats de calculs, comptage, etc. L'afficheur 7 segments représente une solution d'affichage simple. Il s'agit de 7 diodes électroluminescentes placées en forme de '8' selon la convention ci-dessous. Chacun des segments a,b,c,d,e,f,g s'allume selon qu'il apparaît ou non dans le chiffre correspondant au nombre binaire de 4 bits ABCD servant à coder les chiffres de 0 à 9.



Le décodeur 74LS47 fournit la fonction de correspondance entre tous les cas possibles ABCD et les 7 variables de commande des 7 segments a,b,c,d,e,f,g. Il existe deux sortes d'afficheurs, à anode commune (reliée au 5V) ou à cathode commune (reliée à la masse). Dans le premier cas les variables de commande sont actives (allument le segment) au niveau bas (cas du 74LS47), et dans le second, elles sont actives au niveau haut (cas du 74LS48).

Nombre BCD	Nombre décimal	**a**
0000	0	1
0001	1	0
0010	2	1
0011	3	1
0100	4	0
0101	5	1
0110	6	0
0111	7	1
1000	8	1
1001	9	1

La table de vérité ci-contre est celle du segment '**a**' dans l'afficheur. Si on devait cabler les 7 segments, il faudrait tracer 7 tables de vérité analogues et simplifier toutes les expressions logiques correspondan-tes. Les circuits -47 et -48 comportent toutes ces tables de vérité précablées en interne.

Chapitre 5

Nombres binaires - Opérations - Circuits associés

Numérotation binaire

L'utilisation d'un signal binaire permet entre autre de coder des nombres en base 2. Ainsi le nombre 1101 en base 2 vaut $1x2^3+1x2^2+0x2^1+1x2^0=13$ en base 10. Le bit le plus à droite du nombre binaire est le bit de poids faible (Less Significant Bit, ou LSB) ; il représente "uniquement" l'information de parité du nombre. Le bit le plus à gauche est le bit de poids fort (Most Significant Bit, ou MSB) ; il représente plus de la moitié de la valeur du nombre.
Le codage en base 2 consomme plus de chiffres qu'en base dix : $2^3<10$, donc il y a en moyenne un peu plus de 3 fois plus de bits que de chiffres décimaux pour coder le même nombre. Avec N bits, on peut coder 2^N valeurs ; ainsi un octet permet de coder $2^8=256$ valeurs. On peut convertir un nombre de la base 10 à la base 2 par la méthode des divisions successives par 2 ; ainsi $37_{10}=100101_2$.
Pour condenser la notation binaire des nombres, on utilise l'hexadécimal (codage en base 16). Cette base possède 10 chiffres identiques au système décimal auxquels on ajoute 6 chiffres nouveaux qu'on représente simplement par les 6 premières lettres majuscules de l'alphabet : A, B, C, D, E et F.
Il faut 4 bits pour coder un chiffre en hexadécimal. Pour distinguer un nombre en hexadécimal d'un nombre en décimal, on représente parfois le premier précédé d'un '$'. Ainsi on a : $10=16 ; $F=15 ; $100=256 ; $1FF=511 ; $400=1024, etc.
Les multiples les plus utilisés sont le Kilo, le Méga, le Giga, etc. binaires : $1K=2^{10}=1024$ (#1000) ; $1M=2^{20}=1K^2=1048576$; $1G=2^{30}=1K^3=1073741824$.

Nombres entier positifs

Avec N bits, on code 2^N entiers positifs. Par exemple, sur 4 bits, on code 16 valeurs : 0000=0, 0001=1, 0010=2, 0011=3, 0100=4, 0101=5, 0110=6, 0111=7, 1000=8, 1001=9, 1010=10, 1011=11, 1100=12, 1101=13, 1110=14, 1111=15.

Code BCD

Il s'agit d'un code "binaire codé décimal" où l'on utilise 4 bits pour coder les nombres de 0 à 9. ce codage gaspille un grand nombre de bits (ainsi 10 s'écrit 0001 0000) mais il peut sembler pratique justement parce qu'il ne manipule pas de chiffre hexadécimal qu'il faudrait convertir avant de l'afficher.

Code *GRAY*

Le code Gray permet de compter en ne changeant qu'un seul bit lorsqu'on passe d'un nombre au nombre immédiatement supérieur. On obtient le codage suivant en cherchant le nombre le plus proche qui ne diffère que d'un bit du précédent.

Ce code s'appelle aussi code réfléchi car le codage des nombres comportant un bit de poids fort supplémentaire est le symétrique des codages des nombres précédents. Ce codage est utilisé pour les capteurs de position de manière à ce qu'il n'y ait pas de valeur intermédiaire fausse si tous les bits du code ne changent pas en même temps : en codage "normal", le passage de 3 à 4, c'est à dire de 0011 à 0100, peut faire apparaitre les valeurs 0111, ou 0010 ou 0000, et encore d'autres si les bits ne commutent pas tous exactement en même temps.

0000
0001
0011
0010
0110
0111
0101
0100
etc.

Nombres signés

Pour coder des nombres de -N à +N, il faut deux fois plus de bits que pour coder les nombres de 0 à N, c'est à dire qu'il faut un bit d'information supplémentaire. Ce bit est le bit de signe, il indique si le nombre est positif ou négatif. Comme les nombres positifs considérés jusqu'à présent comportaient un bit de poids fort égal à 0, il est légitime d'adopter un bit de signe égal à 1 pour les nombres négatifs.

+3	0011
Inversion des bits	1100
Addition 1	1100 +0001
-3	1101

Le codage adopté par les calculateurs pour les nombres signés est obtenu par la méthode du complément à 2 : on obtient l'opposé d'un nombre en inversant tous ses bits et en lui ajoutant 1. L'opération de calcul du nombre -3 est détaillée dans le tableau ci-contre.

On peut vérifier quelques avantages de cette méthode : le nombre +0 est codé de la même manière que -0 ; si on additionne +3 et -3, on obtient bien 0 (à condition de ne garder que le même nombre de bits qu'au départ pour coder le résultat).
Comme +/-0 prend une place du côté des nombres positifs, on peut coder un entier positif de moins que de nombre négatif ; ainsi, avec 16 bits, on code des entiers de -32768 à +32767, c'est la gamme de variation des entiers standards (codés sur 2 octets) dans un ordinateur 16 bits. Sur un ordinateur 32 bits, les entiers standards sont codés de $-2\ 10^9$ à $+2\ 10^9$ environ.

Nombres réels

Pour coder les nombres réels, on utilise la notation mantisse/exposant : chaque nombre est représenté comme $+/-M\ 2^E$, où l'exposant E est un entier signé.
Dans le code IEEE, un "real*4" est un nombre réel codé sur 4 octets selon la convention suivante : R = [S][... 8 bits...][... 23 bits ...]

[S] [... 8 bits...] [... 23 bits ...]

Signe +/- : 0/1 Exposant : +/-127 Mantisse < $8.4\ 10^6$

On notera la gamme de valeurs accessibles dans ce codage : $2^{127} \approx 1.7\ 10^{38}$, de sorte que l'on peut coder des nombres de 10^{-38} à 10^{+38} environ,, et $2^{23} \approx 8.4\ 10^{6}$, de sorte que la précision est de l'ordre de $+/-10^{-7}$.

Dans la même convention, un réel*8 sera codé avec 11 bits d'exposants et 52 bits de mantisse, ce qui permet de coder des nombres de 10^{-308} à 10^{+308} avec une précision d'environ 10^{-16}.
Le codage adopté pour représenter les réels dépend de l'architecture de la machine de calcul. Les différents constructeurs (IBM, DEC, Apple, etc.) ont des conventions de codage des réels différentes et il faut faire attention lors du passage de l'une à l'autre.

Arithmétique binaire

Addition
L'addition de deux nombres de 1 bit nécessite déjà un circuit à 3 entrées et 2 sorties pour gérer correctement la propagation de la retenue. On trouve dans le commerce des circuits additionneurs cablés sur 4 bits, comme le 74LS283, par exemple.

Multiplication
Le décalage d'un bit vers la gauche correspond à une multiplication par 2 (4:0100 -> 1000:8) ; le décalage vers la droite correspond à la division par 2 (4:0100 -> 0010:2). Cette méthode est utilisée par les algorithmes de multiplication. Un exemple de circuit multiplieur est le -558, qui multiplie deux nombres de 8 bits pour fournir un nombre de 16 bits en sortie.

Comparaison
La série des circuits -682, 684, 688 effectue la comparaison de deux nombres de 8 bits et fournit en sortie les variables P=Q, et P>Q (le résultat P<Q est obtenu à partir des deux premiers).

Unité arithmétique et logique

Ce circuit (ALU en anglais) se trouve au cœur de tout micro processeur. Un microprocesseur manipule des données au sens large du terme : il les stocke, les transfère, les compare, effectue des opérations arithmétiques et logiques, etc. l'ALU comporte deux entrées X et Y pour deux nombres de N bits (par exemple le circuit -181 travaille sur des nombre de 4 bits) et une sortie qui fournit le résultat de l'opération. Cette opération est sélectionnée parmi 16 possibles à l'aide de 4 bits de sélection S_0-S_3.

Codage des caractères

Pour coder l'ensemble des caractères utilisés dans toutes les écritures du monde, il faut un grand nombre de bit. Cependant, pour des raisons historiques, le plus célèbre des codes de caractères, le code ASCII ne permet de coder que 127 caractères. ASCII signifie *American* Standard Code for Information Interchange,

ce qui explique peut-être cela... Le code ascii permet de coder les caractères imprimables sur une télétype (ancêtre de l'imprimante), ainsi que divers caractères de contrôle (retour à la ligne, saut de page, etc.).
Il existe plusieurs extensions du code ascii qui permettent en particulier de coder les lettres avec accent, et divers autres caractères, mais ces extensions ne sont pas forcément toutes compatibles entre elles.

Transmission de données

Contrôle de parité
La transmission de données binaires est a priori plus fiable que la transmission de signal analogique, du fait de l'existence d'une marge de bruit. Cependant, quel que soit le mode de transmission adopté, on peut rendre celle-ci plus sure en adoptant un contrôle entre la donnée émise et la donnée reçue. Le contrôle de parité consiste à rajouter au mot émis un bit supplémentaire de tel sorte que le nombre de bits égaux à 1 dans le mot résultant soit pair ("even parity") ou impair ("odd parity"). Autrement dit, la somme de tous les bits du mot final doit être paire ou impaire selon la parité adoptée. Ainsi en parité impaire, le caractère 'H', codé $48=1001000 (qui comporte un nombre pair de bits à 1) en ascii se verra transformé en $C8=11001000 (dont la somme de tous les bits, égale à 3, est paire). En parité paire, le mot $48 restera codé $48. Le bit supplémentaire est appelé bit de parité. Avant un transfert de données, les deux systèmes qui communiquent (généralement deux ordinateurs) se mettent d'accord sur le type de parité adopté pour la transmission ; à chaque envoi d'un mot binaire, l'émetteur rajoute un bit de parité, et le récepteur vérifie que chaque mot reçu respecte la parité convenue. Si la parité n'est pas respectée, cela signifie qu'un bit du mot global a changé de valeur au cours du transfert, et le récepteur demande une ré-émission du mot.
Le circuit -280 détermine la parité d'un mot binaire : il possède 9 entrées et deux sorties, *odd* et *even*, donnant respectivement 1/0 selon que le mot d'entrée comporte un nombre pair ou impair de bits à 1.
Le contrôle de parité ne permet pas de savoir quel bit a été modifié pendant la transmission et il peut ne pas détecter une erreur si deux bits ont changé de valeur. D'autres codes, comme le *code de Hamming*, qui ajoute plusieurs bits de contrôle à chaque mot envoyé, peuvent permettre, à la réception, de déterminer quel bit est en erreur, et de le corriger.

Liaison RS232
La transmission en parallèle (un fil par bit) de mots binaires comportant un grand nombre de bits d'une machine à l'autre réclamerait trop de fils. De plus, si on définit le cable de transmission pour une taille de mot donnée (par exemple 16 bits), le format devient rapidement obsolète avec l'évolution technologique. Une solution consiste à convertir les mots binaires de parallèle en série pour pouvoir les transmettre sur deux fils seulement (un fil de référence et un fil de signal, en différentiel). L'utilisation de paires (de fils) torsadées est très répandue dans la transmission de données. Elle a l'avantage de rendre la transmission insensible aux parasites par induction électromagnétique car la tension parasite induite changeant de signe à chaque boucle (torsade), le signal résultant sur la longueur du fil est minime.

Une fois la conversion parallèle-série effectuée, on utilise un UART (Universal Asynchronous Receiver Transmitter) pour la transmission. La tension reste à un niveau <-3V pendant l'attente (MARK bit), puis un START bit est émis à une tension >+3V, suivi des 8 bits de signal à 0 ou à 1, terminé ensuite par un ou plusieurs STOP bits à -3V. Le récepteur est activé à la réception du changement de niveau de mark à start, puis il vérifie que le signal est toujours à start 1/2 période plus tard (afin d'éliminer un possible parasite) puis il pointe les valeurs des bits suivants toutes les périodes. La fin d'envoi est validée par la présence du stop bit à la fin du mot. L'envoi est dit asynchrone car la fréquence d'envoi des bits successifs n'a besoin d'être précise que pendant une dizaine de périodes, le temps d'envoi d'un mot de 8 bits.
D'autres standards existent, les RS 423, 422, 485.

Chapitre 6

Bascules, Compteurs, Machines à états & Séquenceurs

Etat logique et mémoire: introduction

Dans un circuit logique combinatoire, comme tous ceux rencontrés jusque là, l'état des entrées détermine pleinement l'état des sorties, par le biais de la table de vérité (si on néglige les états transitoires de quelques ns qui correspondent aux temps de propagation dans les circuits et dans les fils). On peut cependant concevoir des circuits avec rétroaction qui se comportent de manière un peu plus subtile. Ainsi, si on réalise un circuit avec deux portes NOR dont les sorties sont rebouclées sur les entrées, on obtient un circuit à deux entrées A et B et deux sorties correspondantes X et Y. Dans l'état AB=00, les deux états XY=01 et XY=10 sont également possibles. On a toujours Y=/X, mais il n'est pas possible de prévoir X et Y à partir de A et B, et ce n'est pas un effet de temps de transfert.

Toujours avec des entrées AB=00, si on décide de partir d'un état de sortie bien défini, comme XY=01, et que l'on fait passer B à 1, alors XY passe dans l'état 10. Si B repasse à 0, rien ne change plus, XY reste dans l'état 10. Et si on part de l'état XY=10, le passage de B à 1 (puis à 0 si on le souhaite) ne change rien. On vérifie que l'impulsion B : 0-1-0 force le circuit dans l'état stable XY=10. Comme on a toujours Y=/X, on peut caractériser la sortie par Y seulement (qu'on note alors Q) et on vérifie que l'entrée B est une entrée de "remise à 0" (entrée **R**eset, dont le passage à 1 entraine le passage de Q à 0).
Inversement, quel que soit l'état de départ de XY, une impulsion A : 0-1-0 fera basculer le système dans l'état XY=01 (état Q=1), l'entrée A est une entrée de "remise à 1" (**S**et).

On peut imaginer des systèmes bouclés à base de portes NAND mais le principe reste toujours le même : l'état de la sortie XY (/QQ) ne dépend pas directement de l'état des entrées R et S, mais de la séquence des valeurs qu'elles ont prises. Dans le cas d'un système bouclé à base de NAND, ce sont les passages A/B : 1-0-1 qui déclenchent le changement d'état en sortie. Dans tous les cas, ce ne sont pas les états des variables d'entrées qui font changer la sortie mais le passage de l'une ou l'autre des entrées de 0 à 1 ou de 1 à 0. On voit apparaître la notion de commande d'un circuit logique par un signal dépendant du temps et dont les "fronts" (*edges*, de passage de 0 à 1 ou de 1 à 0) sont actifs. On appelle un front de montée *leading edge*, et de descente, *trailing edge*.

NB. la prise en compte des temps de transfert reste d'actualité car aucun changement ne s'effectue instantanément, en électronique logique comme en physique, mais ces temps de transfert ne génèrent pas ici d'ambiguité, quelle que soit leur valeur, le comportement du circuit reste identique.

Bascule Maitre-esclave / synchronisée sur un front

Pour éviter d'éventuels problèmes si A et B devaient changer de valeur en même temps, on ajoute un étage au montage en faisant intervenir un signal supplémentaire. Par l'intermédiaire de portes ET, ce nouveau signal H, en passant à 1, détermine les instants où le circuit bouclé doit prendre en compte les nouvelles valeurs de A et B (R et S). Il reste cependant un risque que R et S changent pendant l'intervalle de temps (court mais pas infiniment court), où H=1. On définit alors un montage "maitre-esclave" constitué de deux bascules en série qui basculent sur des niveaux différents du front de montée de H. La première bascule prend en compte les valeurs de R et S pendant un instant très court et ces valeurs sont alors verrouillées pour la deuxième partie de la bascule.

Bascule D

Si on relie les entrées S et R d'une bascule par un inverseur, on obtient un circuit où la sortie recopie l'entrée au front d'horloge. Une telle bascule est nommée bascule **D** (comme Donnée, ou Data, que la sortie recopie sur commande). Le circuit 74LS109 comprend deux bascules D. La sortie est disponible sous sa forme directe (Q) et sous sa forme inversée (/Q).
Une bascule D possède des temps de transfert $t_{PLH/HL}$, ainsi que deux paramètres temporels supplémentaires, le temps de "setup" t_S pendant lequel la donnée doit être fixe avant le front d'horloge, et le temps de "hold" t_H, pendant lequel le signal D doit rester fixe après le front d'horloge (la plupart des bascules ont maintenant un t_H=0).
Une bascule D possède également deux entrées de mise forcée à 0 ou à 1 : Reset et Set, et ce de manière asynchrone (indépendamment du front d'horloge, c'est à dire que ces entrées sont différentes, dans leur action, des entrées R et S vues au début de ce chapitre, qui étaient elles, synchrones). Les bascules D sont utilisées dans de nombreuses applications dont les compteurs, ainsi que dans des circuits dits de verrouillage ("Latches") car ils permettent d'isoler l'entrée et la sortie à l'aide du front d'horloge. Une autre application est le registre à décalage utilisé par exemple pour effectuer des opérations de division/multiplication par 2.

Bascule JK

La bascule JK possède deux entrées (J et K !) dont l'état au moment du front d'horloge détermine l'état de la sortie Qn+1 après le front, selon la table de vérité ci-dessous.

J	K	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	/Qn

Les deux entrées J et K jouent un peu le rôle d'entrée Set et Reset "synchrones" puisque K=1 force la sortie à 0 au front d'horloge suivant et J=1 force Q à 1. Il existe deux états supplémentaires (00 et 11) qui donnent lieu soit à une recopie de l'état précédent (Qn+1=Qn), soit à une inversion (Qn+1=/Qn, mode "toggle"). Le circuit 74LS112 comprend deux bascules JK. On peut obtenir une bascule D à partir d'une bascule JK en reliant J à K par le biais d'un inverseur.

Les bascules JK possèdent également deux entrées Set et Reset asynchrones, et la sortie est disponible sous forme Q et /Q.

La première application des bascules D et JK est le compteur, qui sera dit asynchrone si les différentes bascules qui le composent n'ont pas la même horloge et synchrone dans le cas contraire.

Compteur asynchrone

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

La réalisation d'un compteur asynchrone est basée sur le principe suivant : dans le déroulement de la séquence de nombres entiers de 0 à N (voir tableau ci-contre), les bits qui constituent le nombre binaire basculent régulièrement de 0 à 1 et de 1 à 0 selon des rythmes différents selon leur place dans le nombre. Ainsi le LSB bascule à chaque front d'horloge, le bit suivant bascule tous les deux fronts d'horloge, le suivant tous les 4 fronts, et ainsi de suite.
A partir de bascules JK, on place toutes les bascules du compteur (une par bit) en mode "toggle" (JK=11) et on utilise la sortie de chaque bascule comme entrée de la bascule suivante, la première bascule du compteur (LSB) étant reliée à l'horloge. On peut également placer une bascule D en mode toggle en reliant la sortie /Q à l'entrée D. Dans les deux cas, on peut prévoir un mode comptage ou décomptage selon que l'on relie Q ou /Q à l'entrée d'horloge de la bascule suivante.

On peut programmer la remise à 0 d'un compteur asynchrone en agissant sur l'entrées Reset de chaque bascule à l'aide d'un circuit qui décode un nombre binaire défini au préalable et qui fournit en sortie un 0 (les entrées Set et Reset, notée /S et /R, sont la plupart du temps actives à l'état bas). Le chronogramme de la remise à 0 d'un compteur asynchrone peut présenter des états transitoires indésirables. Une façon de limiter ce genre d'inconvénient est d'utiliser un compteur synchrone.

Compteur synchrone

Dans un compteur synchrone, il y a toujours une bascule par bit et toutes les bascules sont pilotées par la même horloge. Toutes les bascules changent donc d'état simultanément, l'état futur étant déterminé par les valeurs des entrées avant le front d'horloge. On peut réaliser un compteur synchrone à base de bascules D ou JK.

No d'état	Sorties $Q_3Q_2Q_1$	Entrées $D_3D_2D_1$
		000
a	000	001
b	001	010
c	010	011
d	011	100
e	100	000
a	000	

Dans le cas de bascules D, la méthode consiste à déterminer la liste des états que l'on souhaite voir décrire par le compteur (y compris une éventuelle remise à 0, voire même remise à un nombre non nul), ce qui fournit la colonne des sorties, puis d'en déduire la liste des entrées nécessaires pour passer de l'état présent à l'état futur lors du front d'horloge suivant. On détermine ensuite les fonctions logiques qui permettent de calculer les entrées (pour l'état futur) en fonction des sorties (de l'état présent), à l'aide d'une table de Karnaugh par variable Di si nécessaire. Le tableau ci-contre montre la liste d'états d'un compteur sur 3 bits de 0 (état a) à 4 (état e).
On trouve dans le commerce des compteurs programmables sur 4, 8 et plus de bits. La série de circuits 74LS160-3 compte sur 4 bits, de 0 à 9 ou de 0 à 16. Ces compteurs peuvent être mis en cascade pour obtenir un compteur global sur un plus grand nombre de bits (permettant de compter de plus grands nombres).

On peut réaliser un compteur programmé (de 0 à N) en utilisant des bascules JK. La méthode est la même : dans l'état présent, on prépare l'état futur. Pour cela, on utilise la table de vérité de la bascule JK, sachant que J=K=1 entrainera une transition (mode toggle : 0->1 ou 1->0), et J=K=0 entraînera un gel de la sortie (Qn+1=Qn : 0->0 ou 1->1).

Q1	Q2	φ1	φ2
0	0	0	1
0	1	1	1
1	0	0	1
1	1	1	1

On a donc toujours J=K (=φ), et on détermine la fonction φ pour chaque bascule en fonction du graphe d'états souhaité. le tableau ci-contre donne les fonctions φ1 et φ2 pour un compteur de 0 à 3.

Quelle que soit la méthode utilisée, il est plus sûr de prévoir également les sauts à effectuer à partir d'états théoriquement jamais atteints. En effet, à la mise sous tension ou du fait de parasites, le système peut très bien se retrouver dans un des états non prévus, et s'y trouver "bloqué" si rien n'est prévu pour l'en faire sortir.

Systèmes à états

L'utilisation de compteurs synchrones permet de réaliser des circuits capables de traiter des situations variables dans le temps ; on nomme ces circuits des *machines à état*. Ces circuits peuvent fonctionner de manière autonome, d'autres, plus complexes, tiennent compte non seulement des différents états dans lesquels ils passent mais aussi de variables supplémentaires extérieures qui interviennent alors dans la séquence d'états suivie.

On réalise une machine à états en suivant une série d'étapes bien définies. On commence par déterminer la liste des états souhaités, qu'on repère par des lettres plutôt que par des nombres (pour ne pas risquer d'ambiguité quant au premier état "1" ou "0" par exemple). Il peut être utile de tracer le graphe d'états du système

visé, de manière à bien visualiser les transitions souhaitées. A partir du nombre de ces états, on obtient le nombre de bits nécessaires pour coder ce nombre, donc le nombre de bascules du circuit. On trace le tableau des "états présents" (sorties Qi) dont on déduit les "états futurs" (entrées Di), puis on détermine les tables de Karnaugh des entrées Di en fonction des sorties Qi, avant de tracer le plan de cablage correspondant. Pour réaliser physiquement le circuit souhaité, on peut utiliser un circuit programmable PAL. Par exemple, le circuit PAL16R4 comporte 4 Registres, et permet de réaliser des machines à états comportant jusqu'à 16 états.
Un système à états autonome permet de gérer des situations répétitives : gestion de feux rouges à un carrefour, division de fréquence par un nombre arbitraire, etc.

Lorsque l'on veut gérer des situations répétitives en tenant quand même compte de variables d'entrée (par exemple un feu rouge avec une voie non prioritaire qui ne déclenche le feu vert que si une voiture est détectée), on peut soit synchroniser ces variables de manière à ce que quel que soit leur instant de variation, elles ne soient prises en compte qu'au front d'horloge suivant (machine de Moore), ou bien faire intervenir directement ces variables au niveau du calcul des variables de sortie (machine de Mealy).

Séquenceur

Un séquenceur est un système comprenant un compteur programmable, un circuit mémoire dans lequel est stocké un "programme" sous forme de série d'instructions qui sont fournies (lorsque le compteur adresse la case mémoire correspondante) à un circuit décodeur qui analyse l'action à effectuer et l'état futur souhaité. A chaque étape, le décodeur décide si le compteur doit augmenter d'un cran ou si un saut à une adresse différente est nécessaire. Un séquenceur peut être vu comme un microprocesseur rudimentaire.

Chapitre 7

CIRCUITS MÉMOIRES

Introduction

Le traitement de l'information numérique nécessite souvent de pouvoir stocker cette information (données, programmes, etc.). Pour cela on utilise des circuits mémoires. La bascule D rencontrée précédemment est un circuit mémoire rudimentaire sur un bit (la sortie Q recopie, c'est à dire mémorise l'entrée D pendant une période d'horloge). Un registre à décalage permet de stocker un mot binaire entier.
Il existe aujourd'hui des circuits permettant de stocker beaucoup plus qu'un simple mot (l'entreprise NEC a récemment fabriqué un circuit mémoire de 4Giga-octets, mais donner des chiffres dans un cours sur le numérique est risqué car ils sont sans cesse dépassés). La capacité d'une mémoire se compte en octets.

RAM & ROM

Il existe deux types de circuit mémoire, celles qui gardent l'information lorsque le courant est coupé, appelées initialement mémoires mortes parce qu'on ne peut ré-écrire par la suite, et celles qui sont volatiles, appelées mémoires vives car on peut y écrire de l'information plusieurs fois. Les mémoires mortes et vives sont repérées par des acronymes qui ne sont pas opposés, ROM (Read Only Memory) pour mémoire morte et RAM (Random access Memory) pour mémoire vive. Par exemple, un disque compact (laser) contenant des données est un type particulier de mémoire morte (CD-ROM). Cependant, on ne considèrera dans ce chapitre que les mémoires constituées de circuits électroniques.

Capacité, Format, Brochage

Quel que soit le support physique utilisé pour constituer la mémoire, l'information disponible dans la mémoire est disponible sous forme structurée. Vue de l'extérieur, l'information binaire est stockée dans des *cases* mémoires repérées par une *adresse*. L'adresse est un mot binaire fourni en entrée (par l'utilisateur) et la donnée est un mot binaire fourni en sortie (par le circuit mémoire).
La taille des cases peut varier : 1 bit, 8 bits, 16 bits, etc., c'est le *format* de la mémoire. Le nombre de cases disponibles dans un même circuit peut varier : 256, 512, 1024, etc., c'est la *capacité* de la mémoire. La capacité de la mémoire détermine le nombre de bits d'adresse nécessaires (8 bits pour adresser 256 valeurs, par exemple), et le format de la mémoire détermine le nombre de bits fournits en sortie. Aux connexions d'adresse et de données, s'ajoutent des bits de contrôle (généralement au nombre de 3 ou 4) qui sont utilisés pour déterminer si la mémoire est sélectionnée ou pas (broche CS), si elle est utilisée en écriture ou en lecture (broche R/W), si la sortie est en haute impédance ou pas (broche OE), etc.

On cherchera toujours à disposer de la taille mémoire maximum pour un encombrement minimum, d'où l'emploi d'astuces de cablage. Par exemple, une mémoire d'un Méga-Octet nécéssite 20 bits donc 20 broches d'adresse. On organise alors la mémoire (ne serait-ce que d'un point de vue de représentation) en (1024) lignes et en (1024) colonnes et on fournit l'adresse en deux fois : d'abord celle de la ligne (sur 10 bits) indiquée par un signal supplémentaire RAS (Raw Adress Strobe), puis celle de colonne, sur 10 bits également en utilisant les mêmes broches, et le signal CAS (Column Adress Strobe). On "échange" ainsi 10 connexions contre 2.
Une autre méthode consiste à utiliser les mêmes broches pour l'entrée (signaux en écriture) et la sortie (signaux en lecture), grâce à l'utilisation de buffers 3 états.

Mémoires RAM

Les mémoires RAM sont des mémoires rapides dans lesquelles on peut lire et écrire. Les temps d'accès sont de l'ordre de 10 à 50 ns. On distingue les mémoires DRAM (dynamic ram) où l'information est stockée dans des condensateurs constitués de capacités parasites de grilles de transistors MOS, donc très faibles. De telles mémoires, comme leur nom l'indique, sont très rapides (10-20 ns). Elles ont l'inconvénient de devoir être "rafraîchies" (c'est à dire rechargées) régulièrement, typiquement toutes les quelques milli-secondes.
Il existe également des mémoires dites SRAM (static ram) constituées à base de bascules D, qui gardent mieux l'information mais sont plus volumineuses.

Mémoires ROM

La structure d'une ROM est la même que celle d'une RAM, mais l'information y est stockée à demeure, grace au claquage de micro-fusibles dans la structure même du circuit . Pour programmer une ROM, on utilise les entrées/sorties d'adresse et de données, mais en appliquant une tension plus élevée que lors d'une utilisation normale (21V au lieu de 5V par exemple). Les fusibles une fois grillés, la ré-écriture n'est plus possible mais la lecture peut être effectuée un grand nombre de fois. Les temps d'accès sont plutot plus lents que pour une RAM, mais restent excellents devant le temps d'accès de l'information stockée sur disque dur par exemple (quelques dizaines a quelques centaines de nanosecondes). De tels circuits mémoires permettent de stocker par exemple les premières instructions de démarrage d'un système informatique, des formes d'ondes de signaux, des tables de transcodage (pour une table de Karnaugh qui aurait un trop grand nombre de cases pour être simplifiée facilement, comme le codage des pilotes d'afficheurs 7 segments), etc. Ces ROM sont généralement programmées par le fournisseur du système utilisé et leur contenu est parfois gardé secret.

Certaines ROM peuvent être programmées par l'utilisateur à l'aide d'une interface appropriée ; on parle alors de PROM (Programmable ROM). D'autres ROM peuvent encore être effacées pour pouvoir être ré-utilisées ; on parle alors d'EPROM (Erasable PROM). L'effacement se fait souvent par passage des circuits dans un "bain" de rayons ultra-violets, qui induisent l'ionisation du matériau semiconducteur constituant le circuit mémoire, et la reconstitution des circuits fusibles détruits lors de la programmation précédente. Une telle opération

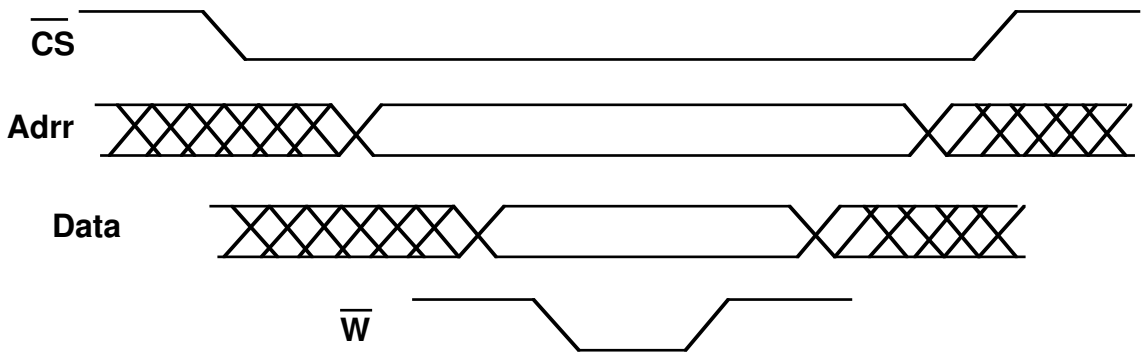
d'effacement peut prendre plusieurs minutes. On peut également effacer les PROM de manière électrique, on parle alors d'EEPROM (Electrically EPROM).

Paramètres électriques et temporels

Comme tous les circuits numériques connus, les mémoires possèdent des paramètres de fonctionnement, électriques et temporels. Il y a même un plus grand nombre de paramètres temporels que pour un circuit normal, car les opérations d'écriture et de lecture doivent être effectuées de manière très précises. On distingue notamment les états suivants :
Etat de veille : l'information est conservée sous tension d'alimentation réduite. Pas de possibilité de lire ou d'écrire. *Etat passif* (standby) : boitier hors veille mais non sélectionné : la consommation est réduite. *Etat actif* : circuit mémoire sélectionné (CE ou CS actif) : la mémoire peut être lue/écrite.
La consommation d'une mémoire peur être importante et varie selon le "rythme de travail" (temps de cycle). Une mémoire aura par ailleurs les paramètres temporels suivants :
Temps d'accès : Δt entre ordre donné et exécution (terme général) *Temps d'écriture* : Δt entre l'instant où toute l'information nécessaire à l'écriture est transmise à la mémoire et l'instant où l'info est effectivement écrite. *Temps de lecture* : Idem / lecture (data présente en sortie). *Temps de cycle* : Δt entre deux opérations consécutives. *Durée de validité* (des données en sortie) : temps pendant lequel les données sont disponibles en sortie. *Temps de maintien* : Δt pendant lequel un signal doit être maintenu pour être "compris" (exemple : temps de présence de l'adresse en entrée, etc.). *Temps de récupération* (recovery time) : Δt nécessaire pour passer d'une opération à la suivante.

Exemple de cycle d'écriture

Dans un cycle d'écriture moyen, il faut tout d'abord sélectionner le circuit concerné, puis présenter l'adresse, puis la donnée et enfin envoyer un ordre /W d'écriture. Après écriture, les signaux (actifs à 0) "remontent" à 1 dans l'ordre inverse.



Extension de format et de capacité

Lorsqu'on ne dispose pas de circuits ayant la capacité ou le format nécessaire, il est toujours possible d'étendre l'une ou l'autre ou les deux caractéristiques. On augmente le format en faisant travailler plusieurs circuits avec le même bus d'adresse et des sorties en parallèle (deux fois 8 bits sur un bus 16 bits en sortie). On augmente la capacité en plaçant les sorties sur un bus de même taille et en attaquant les boitiers mémoire successivement, les premiers bits (de poids faible) étant utilisés pour les adresses et les bits de poids fort étant utilisés pour sélectionner le circuit actif (entrée /OE).