

YBET déménagement
Rue Albert 1er, 7
B-6810 Pin - Chiny
Route Arlon - Florenville
(/fax: 061/32.00.15



Ciel
comptabilité
pour
Belgique et
Luxembourg



FORMATIONS

Le MAGASIN YBET

PRODUITS et SERVICES

[Formations à Pin](#)

[Activités et présentation](#)

[Support technique](#)

[COURS HARDWARE](#)

[Rayon d'action](#)

[Caisse enregistreuse, balance TEC](#)

[Définitions des termes techniques](#)

[Plan d'accès à Chiny](#)

[Logiciel de gestion CIEL et SAGE](#)

[YBET informatique](#)

[CONTACT](#)

[Forum informatique
technique](#)

[Achat en ligne](#)

3. Fonctionnement d'un système Microprocesseur, Z80



[3.1. Introduction](#) [3.2. Exemples de timing du Z80](#) [3.3. Décodage d'adresses](#) [3.4. Schémas d'une carte à base de microprocesseur Z80](#) [3.5. Systèmes Informatiques](#) [3.6. Types de mémoires](#)

3.1. Introduction.

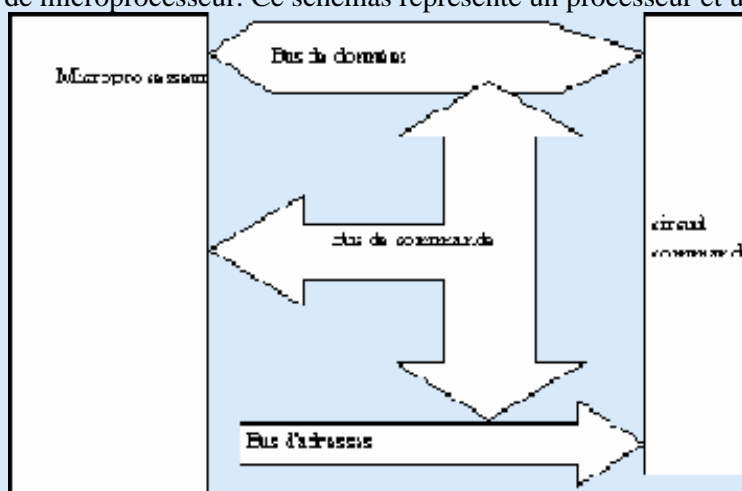
Commençons par examiner un petit montage électronique à base de microprocesseur. Ce schéma représente un processeur et un circuit interfacé.

Le **microprocesseur** (aussi appelé CPU pour Central Processor UNIT) est l'élément principal d'un ordinateur, il traite des instructions à la suite l'une de l'autre. Il ne prend aucune décision, seules des instructions conditionnelles sont influencées par des situations extérieures: clavier, demande de service d'un périphérique, ... Ces suites d'instructions sont communément appelées un programme. Chaque modèle de microprocesseur lit des instructions spécifique à sa conception sous forme d'un langage de base que l'on appelle **assembleur**. Ce langage de programmation est complexe à utiliser puisqu'il est spécifique "machine" et codé en hexadécimal (au même titre que les données ce qui complique encore la

programmation). Les logiciels que nous utilisons sont écrit dans des langages évolués (C, Visual Basic, ...) qui transposent les programmes en assembleur compréhensible par le processeur. Sauf quelques petites améliorations, tous les microprocesseurs des ordinateurs de la famille X86, du moins en 32 bits, (Pentium, Athlon, ...) comprennent le même langage assembleur.

Le circuit interfacé ci-dessus est divers: mémoire, port de sortie, ... Néanmoins, tous les montages électroniques à microprocesseurs incluent un programme de départ dans une mémoire ROM (le contenu n'est pas effacé en l'absence de tension d'alimentation du circuit). Ce programme permet au microprocesseur d'exécuter son initiation au démarrage (ce qu'il doit faire comme détecter le disque dur, tester la mémoire, ...). Un système à microprocesseur est donc constitué de plusieurs circuits interfacés, par exemple, mémoire ROM (obligatoire), mémoire RAM (mémoire de travail pour les résultats), port d'entrée (clavier), port de sortie (afficheur, écran), ... mis en parallèle.

Deux types de processeurs sont fabriqués, le **microprocesseur** et le **microcontrôleur**. Au niveau traitement des informations, les 2 sont pratiquement équivalents. La distinction vient des fonctionnalités internes. Un microcontrôleur est dédié aux traitements des entrées / sorties. De ce fait, des ports sont rajoutés (parallèle et / ou série suivant les modèles) qui vont permettre de recevoir ou d'envoyer des informations de périphériques lents. On pourrait utiliser un microprocesseur pour les mêmes fonctions mais ceci nécessiterais de rajouter des composants électroniques externes pour chaque port externe. Un microcontrôleur inclut souvent la programmation dans une mémoire ROM interne et même de la mémoire de travail de type RAM. Comme un microcontrôleur gère des périphériques lents, il n'est pas optimisé pour la vitesse de traitement

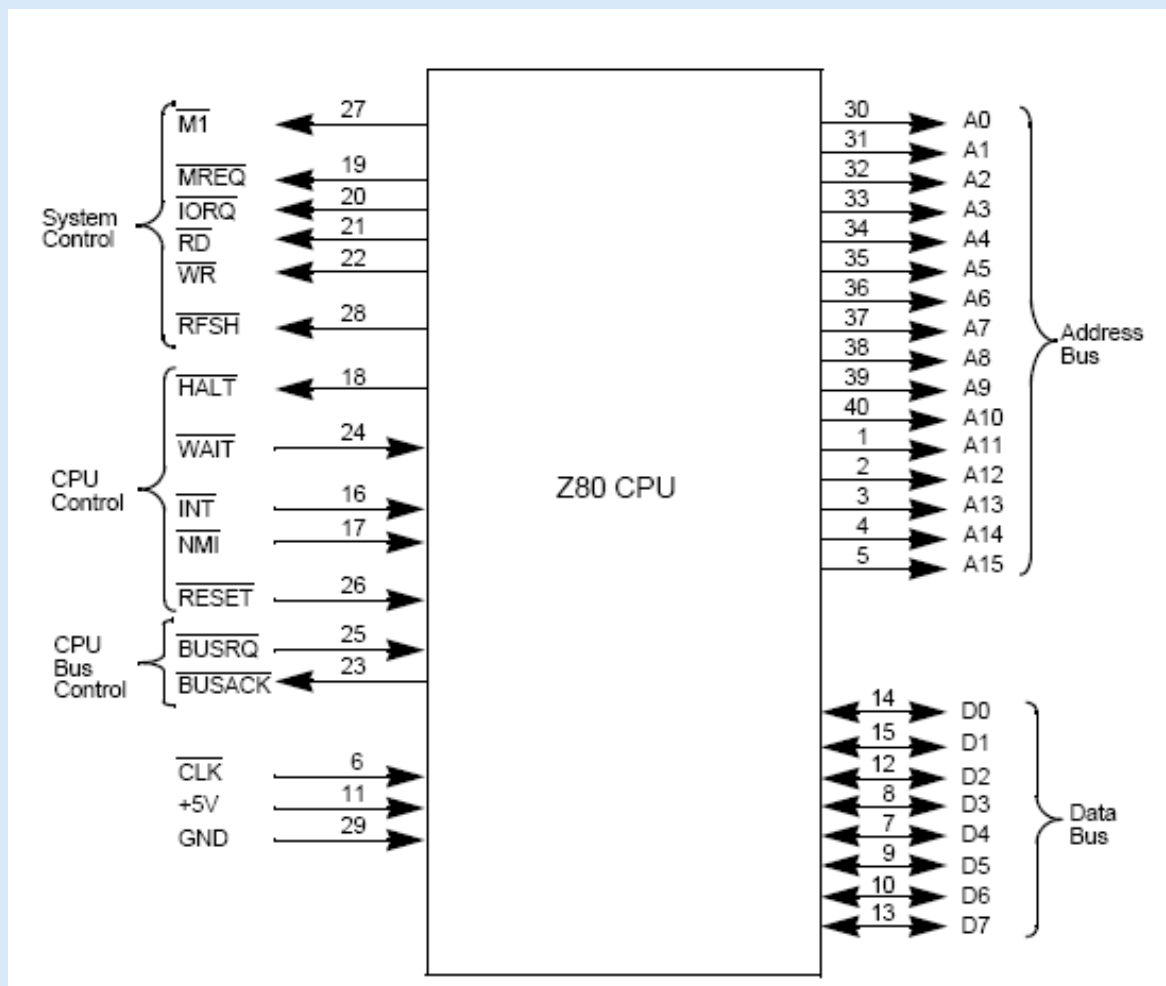


d'informations, ni même pour gérer des quantités de mémoire importantes.

L'interfaçage du processeur vers le circuit de commande nécessite 3 bus: un bus de donnée, un bus d'adresse et un bus de commande. Un **bus** est un ensemble de lignes de communication (matérialisés par des fils) qui relie 2 (ou plus) circuits digitaux entre-eux. Sur chaque fil, la tension électrique peut prendre 2 valeurs distinctes que nous désignerons généralement par 1 et 0 (tension présente ou absente).

Chaque emplacement mémoire ou périphérique interne est désigné par une adresse spécifique (parfois plusieurs à la suite de l'autre, ce que l'on appelle une plage d'adresse), similaire à l'adresse postale. Une adresse spécifique ne peut pas être partagée entre plusieurs circuits. Le **bus d'adresse** permet au processeur de communiquer avec le périphérique via son adresse (ou sa plage). Un bus d'adresse est constitué de plusieurs lignes. Un bus d'adresse 8 bit correspond à 8 lignes d'adresses et peut donc adresser 2^8 adresses, soit 256 adresses différentes, et ainsi de suite. Plus le nombre de lignes d'adresse est élevé, plus le processeur est capable de gérer de périphériques.

Une fois le périphérique contacté via le bus d'adresse, le **bus de donnée** permet de transférer des données binaires (y compris les lignes d'instruction en assembleur): en lecture (données transférées au processeurs) ou en écriture (envoi de données vers le périphérique). Le bus de donnée est constitué d'un certain nombre de lignes. Toutes les capacités des bus de donnée sont désignées sous 8 lignes de données (en Byte - octet), ou en multiple de 8 bits, les processeurs actuels utilisent 64 lignes de données par exemple. Donc 1 MB de mémoire signifie 1 MB sous 8 lignes, soit 8 Mb (Méga bits).



Brochage du microprocesseur Z80

Un **bus de commande** synchronise les transferts de données entre le processeur et les périphériques (mémoire, entrées / sorties). Il assure le dialogue nécessaire pour le transfert à (opération d'écriture) ou de (opération de lecture) l'adresse indiquée. Les signaux du bus de commande permettent également de gérer les interruptions, commandes spécifiques qui permettent à un circuit externe de signaler au processeur qu'il est prêt à recevoir des données de l'extérieur par exemple. Signal bas signifie qu'il est actif lorsque le signal est à 0 V (on le désigne par signal), non actif lorsque le signal est à l'état haut (typiquement 5V). Les séquences présentes sur le bus de commande sont également spécifiques au processeurs. C'est ce que l'on appelle le TIMING. Voici par exemple le brochage d'un processeur Z80 des années 80.

A0-A15: bus d'adresse, sortie 3 états, signal actif haut. Ceci permet donc 65536 adresses différentes (2^{16}). Pendant le rafraîchissement des mémoires, les 7 bits les plus bas contiennent l'adresse de validation du rafraîchissement. Le rafraîchissement sera expliqué avec les mémoires.

D0-D7: **bus de données**, entrée / sortie [3 états](#), état actif haut

MREQ: **Memory Request**, [sortie 3 états](#), signale que le bus d'adresse comporte une adresse valide pour un accès en mémoire (lecture ou écriture).

IORQ: **Input/ output request**, sortie 3 état, état actif bas, ce signal de demande d'entrée sortie indique que la moitié basse du bus contient une adresse d'entrée / sortie valide pour une opération d'entrée / sortie en lecture ou en écriture.

RD: **Memory Read**, sortie 3 état actif bas. Le signal indique que le microprocesseur veut lire une donnée dans la mémoire ou dans une entrée / sortie.

WR: **Memory Write**, Sortie 3 états, actif bas. Indique une demande d'écriture en mémoire ou en entrée / sortie.

RFSH: **refresh**, sortie, actif bas, indique une adresse de rafraîchissement sur les 7 bits inférieurs du bus d'adresse.

HALT: **Etat d'arrêts**, sortie, actif bas, indique que le CPU vient d'effectuer une instruction software HALT et attend une interruption masquable ou non masquable. Pendant l'arrêt, le CPU exécute des rafraîchissements de mémoires.

WAIT: **attente**, entrée, actif bas, indique au Z80 que le périphérique n'est pas prêt à envoyer des données, permet de synchroniser un périphérique plus lent.

INT: **interrupt request**, entrée, signal actif bas. Ce signal de demande d'interruption masquable (par instructions logicielles) signale au processeur qu'un périphérique demande une interruption. Quand le processeur accepte l'interruption, un signal accusé de réception IORQ est envoyé au début du cycle d'instruction suivant.

NMI: **Non-maskable interrupt**, entrée, déclenchée par flanc descendant. Cette demande d'interruption à la priorité sur int et est toujours prise en charge à la fin de l'instruction en cours. Dans le cas du Z80, le CPU redémarre automatiquement à l'adresse 0066 hex. Le contenu du compteur ordinal est automatiquement sauvegardé pour reprendre le programme après l'interruption.

Reset: réinitialisation, entrée, active bas, réinitialise le processeur dont initialisation de tous les compteurs, remise à 0 des interruptions. Le processeur redémarre au début de son programme d'initiation.

BUSRQ: Bus request, permet d'utiliser le DMA (direct memory access). Le CPU fait passer toutes ses entrées sortie trois états dans ce mode dès la fin du cycle machine en cours. Ceci permet aux périphériques de prendre entièrement le contrôle des bus et de transférer des données vers la mémoire sans passer par le processeur. Cette fonction est largement utilisée dans les systèmes actuels (disques durs [E-IDE](#) et SCSI, bus AGP et PCI, ...)

BUSAK: Bus acknowledge, sortie, actif bas, indique que le CPU accepte le BUSRQ et que les lignes sont en 3 états.

M1: cycle machine M1, cycle machine en cours est le cycle de recherche d'un code opératoire, ceci est spécifique au Z80.

Horloge: ce signal provenant de l'extérieur va cadencer l'ensemble des signaux du montage. C'est la fréquence de travail du processeur.

Peu de ces signaux n'interviennent réellement dans les notions courantes sur les systèmes PC (et autres). Cinq signaux sont cependant primordiaux pour une système informatique: **RESET**, **INT**, **NMI**, **BUSREQ** et **BUSAK**. Ils seront utilisés dans la suite du cours au niveau des PC.

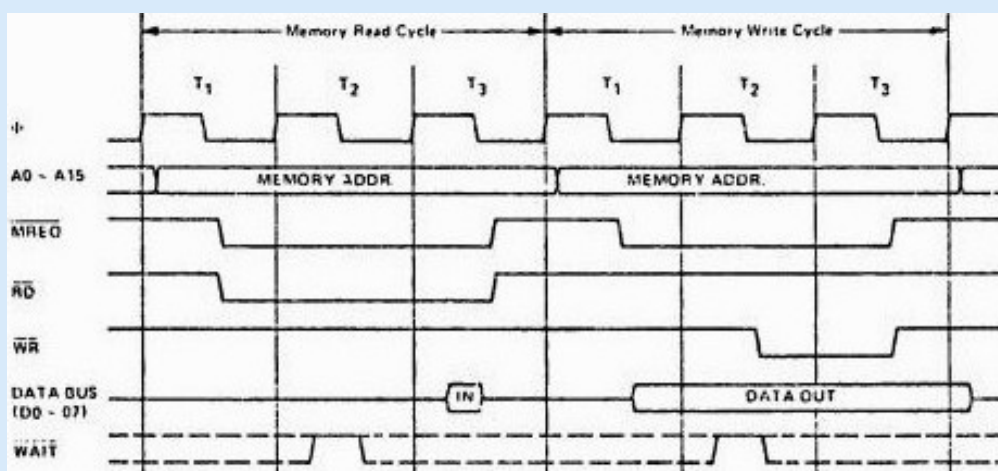
Pour la compréhension, faisons **une similitude** entre un système à microprocesseur et une salle de cours. Le professeur représente le microprocesseur, les étudiants des périphériques. Lors d'un cours normal, seul le processeur (professeur) dialogue avec les périphériques (étudiants). Un étudiant souhaite poser une question, il va utiliser une demande d'interruption (par

exemple lever le doigt). Pour repérer quel étudiant pose la question, le professeur va utiliser une table d'adresse (correspondance numéro de l'interruption - adresse étudiant) pour distinguer qui demande parmi les autres. Une fois l'étudiant clairement identifié, le bus de donnée va servir à transférer l'information (la question). La suite du dialogue va utiliser le bus d'adresse (distinction des interlocuteurs) et le bus de donnée. Supposons que le directeur de l'établissement rentre dans la salle. Ce directeur peut être vu de 2 manières, comme une interruption non mascable (identique à une interruption normale mais le processeur ne peut oublier de répondre, masquer la question). Voyons plutôt ce directeur comme une demande de bus (un Direct Memory Access). Le directeur (un périphérique avec de l'influence) demande la parole (un BUSRQ). Le professeur finit sa phrase en cours (l'instruction du processeur) et signale au périphérique (directeur) qu'il donne le contrôle par un BUSACK (qu'il libère les bus en se déconnectant) et le périphérique - directeur va pouvoir prendre la parole sur le montage électronique (la classe complète ou un élève - périphérique particulier).

... ça va me faire bizarre lorsqu'un étudiant posera une question (pardon une demande d'interruption).

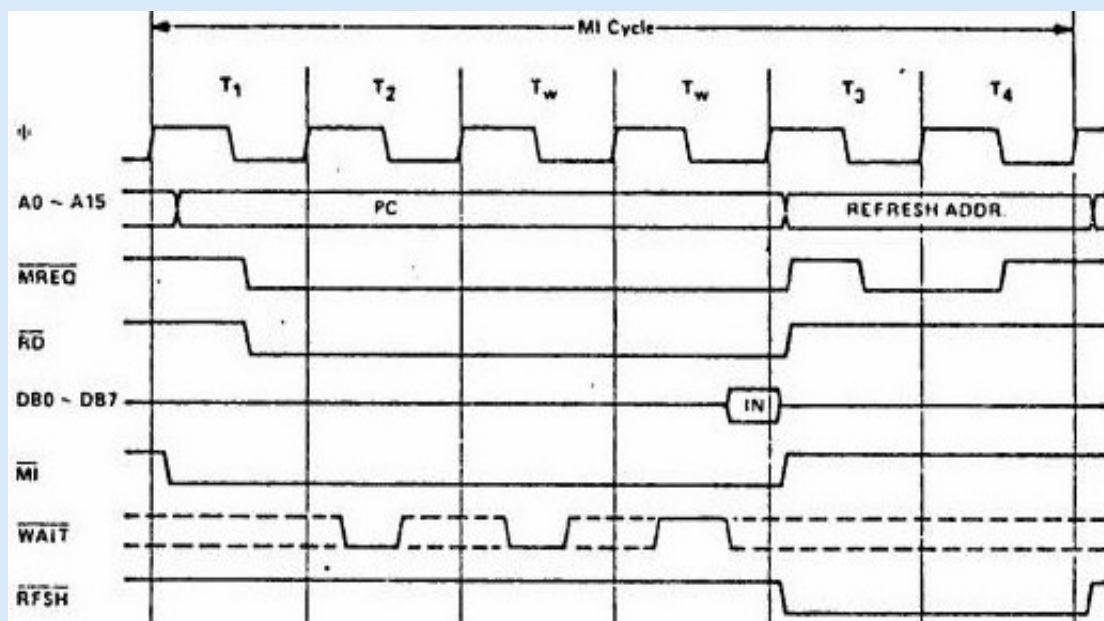
3.2. Exemples de timing microprocesseur Z80

Le but n'est pas de rentrer dans les détails des TIMING d'un Z80 (ni d'un autre d'ailleurs) mais d'expliquer ce qui se passe au coeur du montage informatique.



Lecture / écriture mémoire

Ceci représente le chargement d'une instruction ou d'une donnée par le processeur suivi d'une écriture. L'ensemble des signaux est cadencé par l'horloge. Le microprocesseur commence par mettre une adresse valide sur le bus d'adresse (celle où se trouve l'instruction). Une fois le signal stable, il émet un signal MREQ et un READ pour signaler à la mémoire qu'une adresse valide est présente sur le bus et qu'il va demander un accès mémoire en lecture. A la fin du cycle, la mémoire transfère sur le bus de données les informations (IN). Dans le cas d'une écriture vers la mémoire, le signal READ est remplacé par un signal WRITE. Un signal WAIT est renvoyé par la mémoire (ou plus généralement par un petit montage inséré dans les bus de commande) pour ralentir le montage.



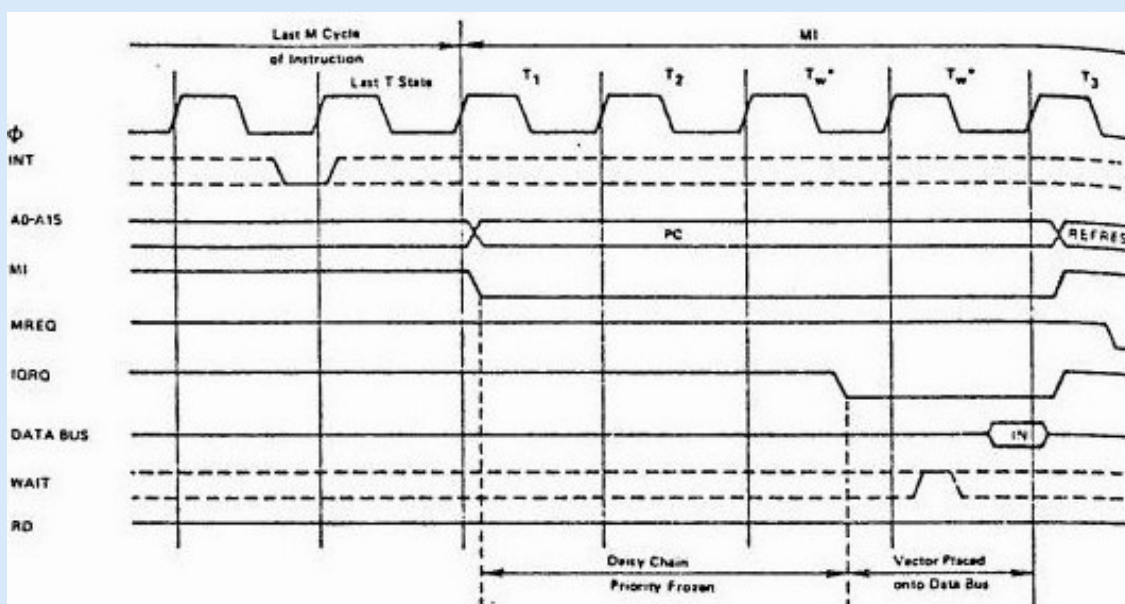
Instruction de code Fetch avec Wait States (instruction de chargement de donnée avec 1 temps d'attente).

Ce timing est identique à celui ci-dessus sauf que 2 temps d'attente (WAIT) sont insérés pour ralentir le montage (cas où la mémoire n'est pas assez rapide pour le processeur) et qu'un signal de rafraîchissement mémoire (REFRESH) est envoyé à la fin du chargement d'instruction.

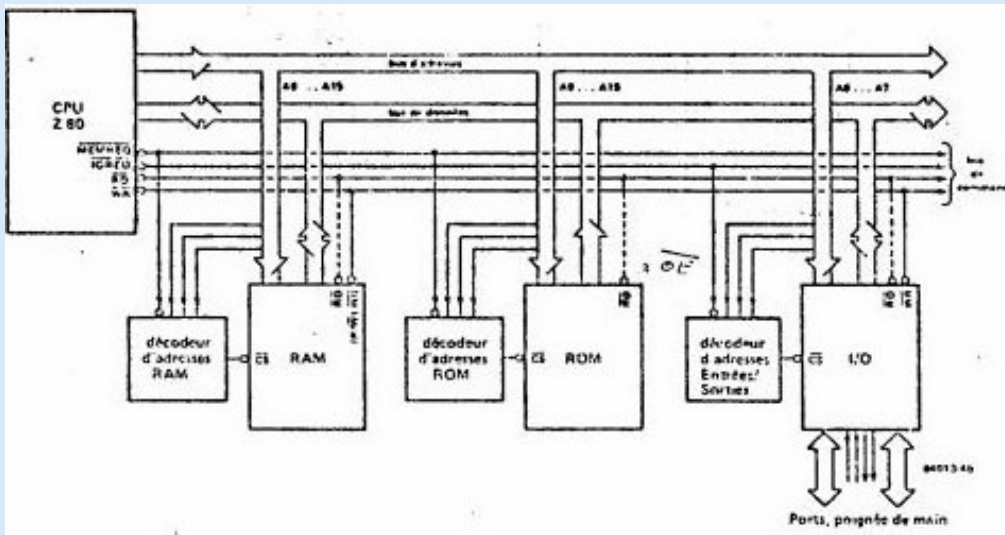
Requête d'interruption.

Une interruption permet à un périphérique de signaler au processeur qu'il doit s'occuper de lui (arrivée d'une donnée, ...). Le signal INT est échantillonné par le processeur sur le flanc montant du dernier cycle d'horloge de chaque instruction. Ce signal d'interruption n'est accepté que si le masque software d'interruption (une commande assembleur spécifique qui demande au processeur de ne pas tenir compte des interruptions masquables) n'est pas accepté ou si le signal BUSRQ n'est pas actif (demande de DMA).

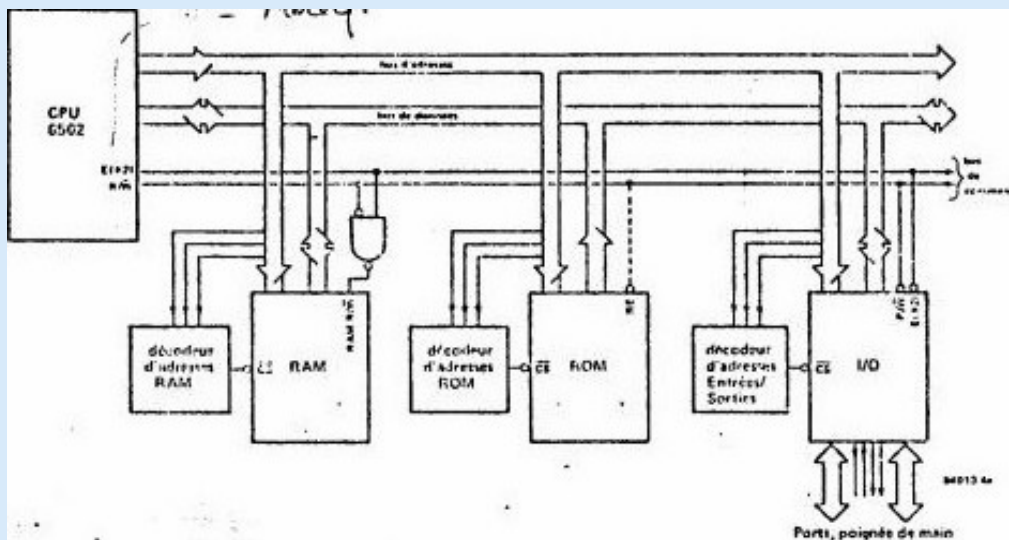
Quand le signal est accepté, un signal M1 est généré. Durant ce cycle spécial M1, le signal IORQ devient actif (à la place du signal mémoire MREQ), indiquant que le périphérique demandant une interruption peut placer une donnée sur 8 bit sur le bus de donnée. Dans le cas du Z80, 2 instructions d'attente sont d'office générées, ce qui évite un montage électronique pour ralentir la lecture des données sur des périphériques plus lents que la mémoire. Ceci permet au signal de se stabiliser.



3.3. Décodage d'adresses.



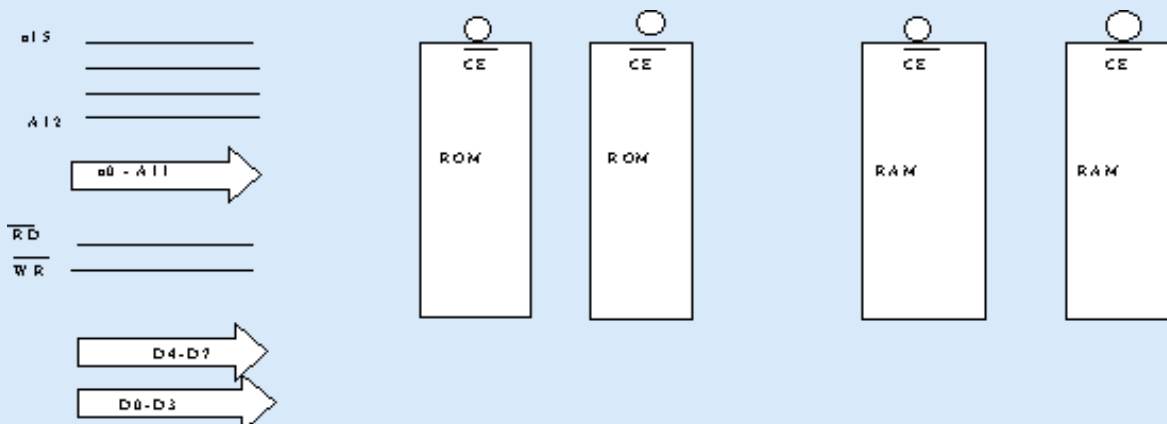
Décodage d'adresse d'un microprocesseur Z80



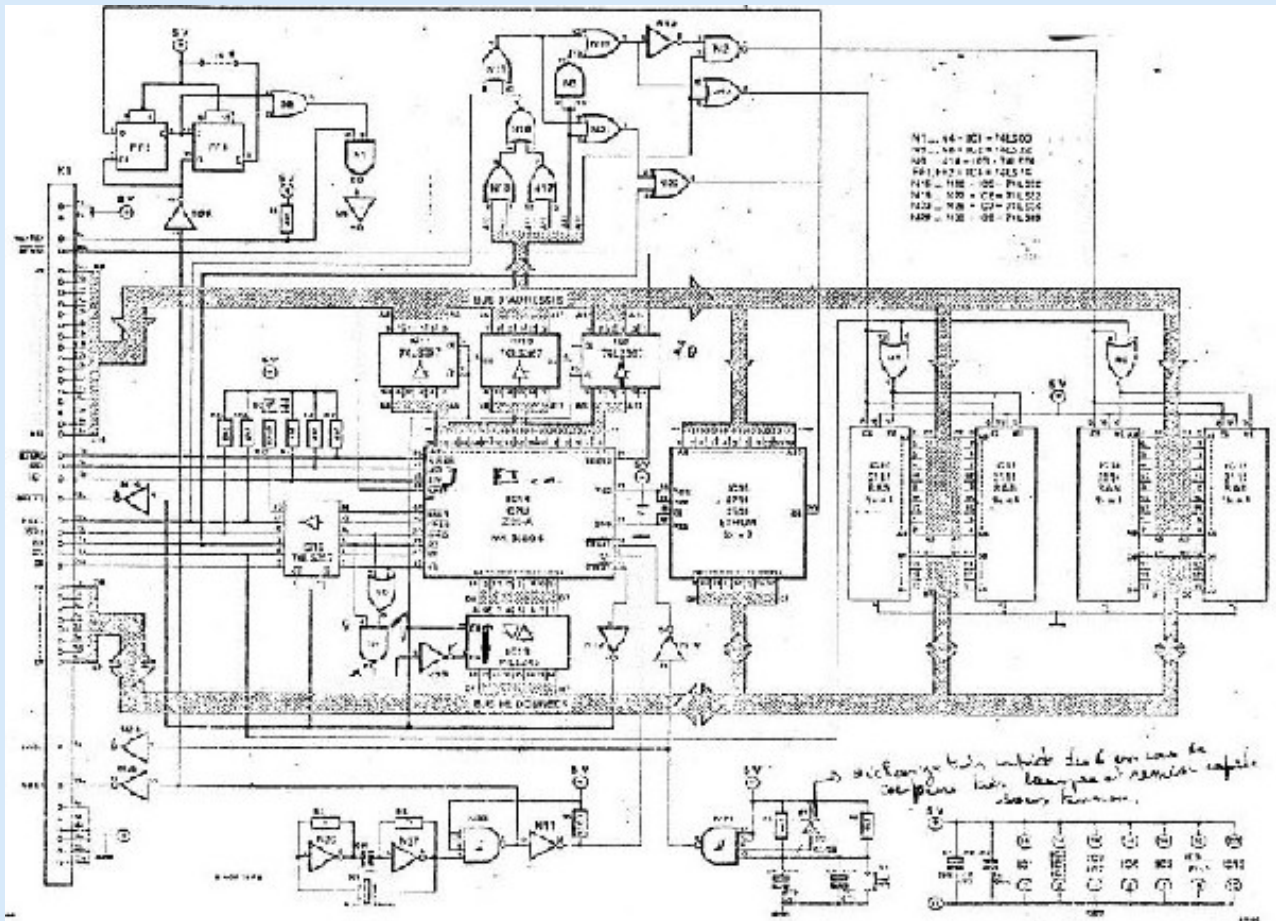
Décodage d'adresse d'un microprocesseur 6502

Ci-dessus le décodage d'adresse pour un Z80 et pour un 6502. Les signaux Read / Write sont décodés différemment. Le Z80 possède les broches READ et WRITE, à la différence du 6502 qui ne possède qu'une broche lecture / écriture. Nous connectons chaque fois une mémoire RAM (random Memory Access), une mémoire ROM (Read Only Memory) et un périphérique de sortie.

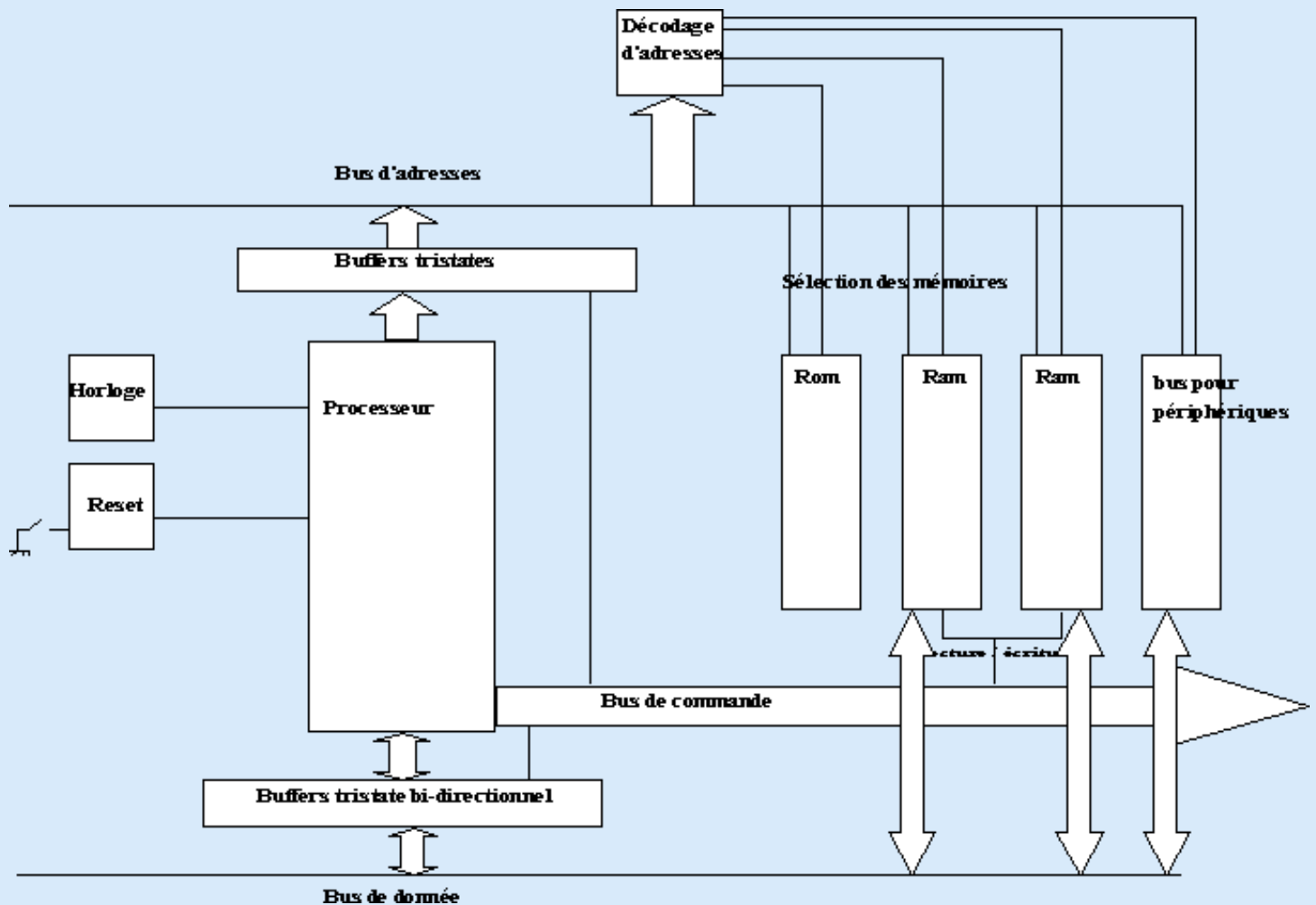
Exercice: supposons un système bus d'adresse 16 bits, bus de données 8 bits comportant 4 circuits mémoires comportant 2^{14} adresses valides chacune sur 4 bits de données. Dessinez les bus d'adresses et de données, ainsi que le décodage d'adresse sur la borne CE (Chip Enable) à base de portes NAND, OR ou NO.



3.4. Schémas d'une carte à base de Microprocesseur Z80



En schématique, le schémas électronique (sans les alimentations) devient comme ci-dessous.



Le microprocesseur z80 est connecté au bus d'adresse et au bus de donnée par des buffers [tri-state](#). Ces buffers ne sont pas obligatoires (inclus dans le processeur) mais sont souvent rajoutés par précaution. Ces buffers "trois états" permettent les fonctions de DMA (Direct Memory Access) du montage électronique: transferts directs des données de (ou vers) la mémoire à partir d'un circuit périphérique sans transit des données par le processeurs. Le circuit périphérique doit pour ce faire contrôler les trois bus.

Le bus d'adresse est relié en partie basse sur les mémoires et périphériques. La partie haute est réservée pour le décodage d'adresse.

Le bus de donnée est directement raccordé sur tous les périphériques. Chaque périphérique est également relié au bus de commande par les signaux lecture (READ) et écriture (WRITE), actifs bas.

Le signal RESET (le même que celui sur la face avant du PC) réinitialise le montage de manière hardware. Contrairement au RESET software (de type <ALT> + <CTRL> + <Suppr> dont le code est programmé dans le système d'exploitation), celui-ci fonctionne dans tous les cas de plantage, sauf si le montage électronique est en panne ce qui est rare (Microsoft contribue nettement mieux aux problèmes PC que INTEL, VIA, AMD et tous les autres réunis).

Le dernier signal est l'horloge qui synchronise la vitesse de transfert entre tous les circuits.

3.5. Montages informatiques

Le schéma électronique ci-dessus représente un système à microprocesseur de type industriel. Très efficace dans un processus "fermé", ce montage électronique est difficilement utilisable tel quel dans un système informatique courant. Tout le travail va être de remplacer les périphériques par des circuits plus ou moins spécialisés.

Dans le cas d'un ordinateur, il va falloir afficher le résultat du travail du système sur un écran ou un LCD. Dans le cas d'un ordinateur, on utilise un circuit dédié inséré sur une carte électronique que l'on appelle "[Carte graphique](#)".

Pour que nous, utilisateurs, puissions communiquer avec l'ordinateur, nous allons insérer un clavier et une souris. Ceux-ci sont également connectés via un circuit spécialisé.

En dernier, pour pouvoir interfacer des appareils externes divers, nous allons utiliser des "ports de communication". Ces ports peuvent être de type parallèle ou série.

Dans le cas d'un **port parallèle**, les lignes de données sont multiples de 8 (du moins dans un système informatique de type PC ou MAC). C'est en gros une continuation d'un bus de donnée vers l'extérieur via une interface dédiée qui utilise des signaux de contrôle plus spécifique aux appareils externes. La connexion d'une imprimante via un port Centronix est de ce type.

Dans le cas d'un port série, la transmission des informations se fait via un fil (en pratique 3 minimum: envoi, réception et masse). L'information sur 8 bit (un Byte) est découpée pour envoyer chaque bit un à la suite de l'autre. C'est le cas par exemple d'un modem (port série), d'un périphérique USB. Le fil de communication reçoit non seulement les données, mais également des signaux de contrôle. Nous verrons en détail ces liaisons plus tard.

En dernier, le système ci-dessus est appelé **système fermé**. En effet, il n'est pas possible de raccorder sur ce système des cartes supplémentaires. Dans le cas de **systèmes ouverts**, comme c'est le cas pour les PC, on installe sur la carte des ports internes. Ils n'ont rien à voir avec les ports de communication ci-dessus. Ce sont des connecteurs normalisés (dimension, brochage, timing des signaux, ...) qui permettent d'insérer des cartes électroniques supplémentaires pour des fonctions dédiées, notamment des ports de communication parallèles ou séries.

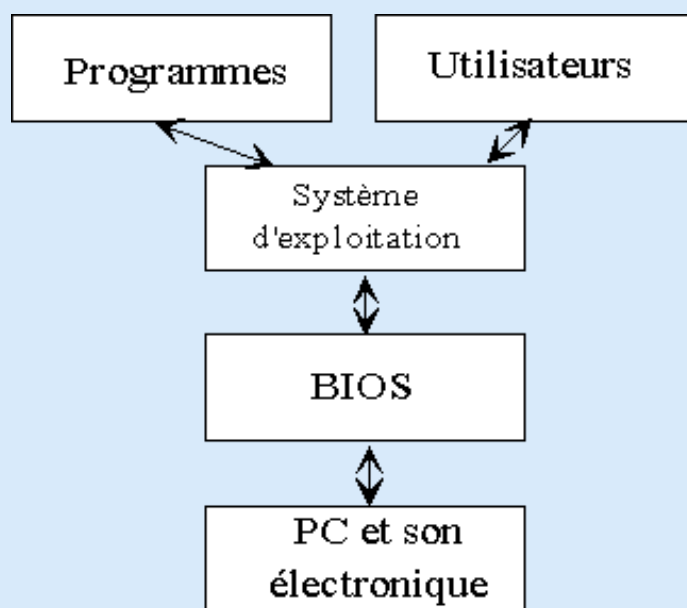
3.6. Types de Mémoires

On trouve 2 types de mémoires: les ROM et les Ram.

Une mémoire RAM permet l'écriture et la lecture. Néanmoins, elle nécessite d'être rafraîchies régulièrement. En effet, les données (1 ou 0) sont sauvegardées par effet capacitif dans un transistor. Les Ram sont raccordées par l'extérieur par le bus d'adresse et le bus de données. Les signaux de contrôle sont CE, WR et RD (oe). En plus, et c'est là leur principal problème, le contenu est effacé si vous cessez de les alimenter.

Une mémoire ROM est uniquement utilisable en lecture et programmée directement à la fabrication du circuit électronique. Le contenu (programme - donnée) n'est donc pas modifiable. De ce fait, d'autres circuits de type ROM modifiables sont utilisés: Eeprom (Electric Prom) programmables en 1 fois (mais peuvent être dans certaines versions effacés à la lumière UV), EEprom (Electric Erasable Prom), que l'on peut effacer complètement plusieurs fois et Rom Flash que l'on peut écrire directement par un signal électrique. Tous ces circuits gardent les informations si l'on coupe la tension d'alimentation.

Le BIOS, la rom électronique des PC comportant le programme de démarrage électronique, cette mémoire est de type Eeprom ou Flash Rom. Ce BIOS est apparu avec l'ordinateur à base de microprocesseur 286 d'IBM. Le Bios est une programmation spécifique au PC et permet certains paramétrages de l'utilisateur. Les paramétrages utilisateurs sont sauvegardés dans une mémoire. Cette mémoire de type RAM (le contenu est effacé si le circuit n'est pas alimenté) est alimentée en permanence par un pile.



On désigne les programmations spécifiques à l'électronique sous le nom de firmware. Cette dernière notion modifie notre schéma de base de l'informatique. Le flashage du BIOS permet donc de mettre dans la partie "flash Rom" une nouvelle version du [firmware](#) de votre ordinateur. Durant l'opération, la partie firmware est chargée en mémoire Ram, le temps de recharger en mémoire ROM la nouvelle version: s'il y a une coupure de courant durant cette opération, il est impossible de redémarrer la carte mère (et donc l'ordinateur), le BIOS (la partie ROM d'un système PC) étant absent ou incomplet.

[Hardware: microprocesseur PC](#)

Evolution, listes et caractéristiques des microprocesseurs utilisés dans les PC

[Réparation ordinateur portable](#)

Technologie, réparations, upgrade des ordinateurs portables: le cours hardware

[Ciel Compta 2006](#)

Probablement le meilleur rapport qualité - prix en comptabilité

[Les chipset PC](#)

Cours hardware: le circuit principal d'une carte mère, le chipset

[Les définitions techniques](#)

Dictionnaire technique en électronique, électricité, hardware, logiciels, Internet

[Les microprocesseurs serveurs](#)

Les processeurs spécifiques pour ordinateurs serveurs

[Cours: structure interne des microprocesseurs](#)

Architecture d'un microprocesseur moderne: RISC, pipeline, superpipeline, ..

La suite du cours hardware 1 > 4. [Augmenter les performances d'un système à Microprocesseur](#)

< 2. [Bases en électronique](#)

Mise à jour 10/07/2007

La [Formation hardware 1: PC et périphériques](#). La [Formation Hardware 2: réseaux, communications et serveurs](#)

Retrouver l'ensemble d'[YBET informatique à Florenville](#)

