- ENSEIRB -

LE BUS INDUSTRIEL VME

Patrice KADIONIK kadionik@enseirb.fr
http://www.enseirb.fr/~kadionik

Bernard HUMBERT (IRES) bernard.humbert@ires.in2p3.fr

SOMMAIRE

1. QU'EST CE QU'UN BUS ?

Nous pouvons estimer qu'il y a deux notions importantes derrière le terme BUS :

- Il s'agit d'une part d'une structure d'interconnexion qui permet de relier par des lignes de communication, les uns aux autres différents éléments pour former un ensemble: un bus est une structure d'interconnexion constituée d'un ensemble de lignes (des liaisons électriques le plus souvent) sur lesquelles transitent les informations que s'échangent deux unités qui communiquent (exemple: l'échange d'informations entre un microprocesseur et sa mémoire centrale constituée par des boîtiers RAM).

- D'autre part, nous parlons de structure en bus à la différence des structures en anneau, en étoile, en réseau maillé etc... C'est une voie commune à plusieurs éléments dont le nombre peut varier sans changer la structure d'interconnexion. D'un point de vue électronique, la structure en bus impose l'utilisation de boîtiers à sorties trois états (tristate) (pour les données et les adresses) et/ou à sorties collecteur ouvert (pour les signaux de contrôle en OU câblé).

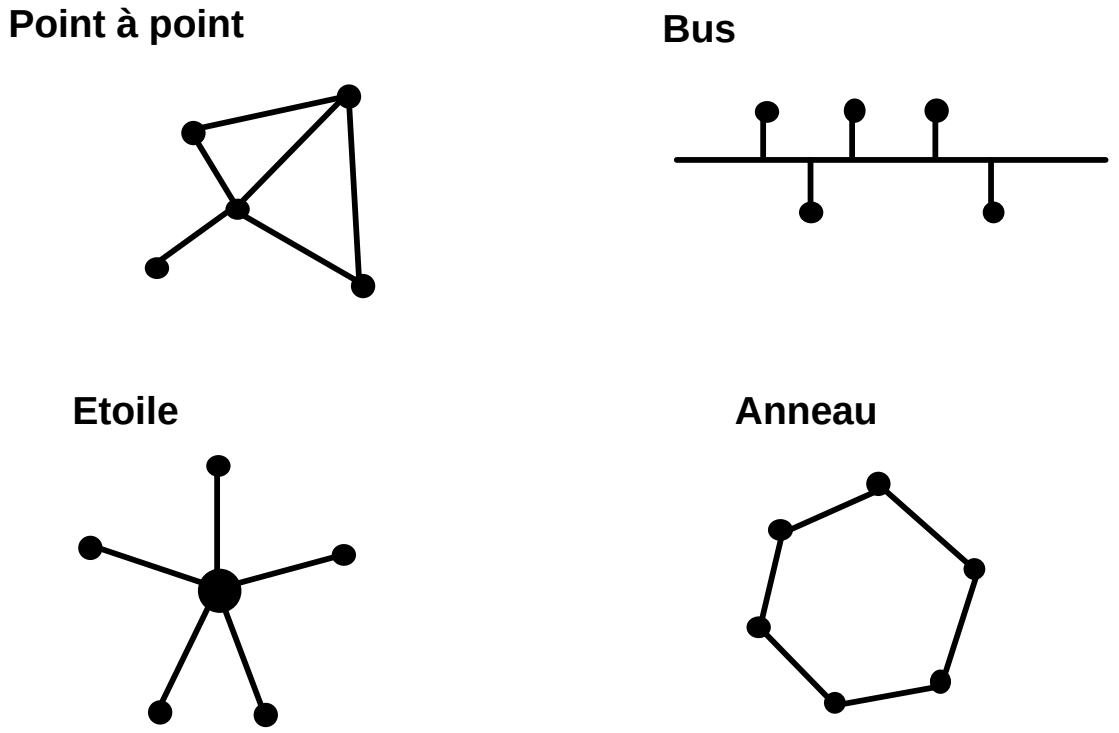


Figure 1 : Topologie des bus

2. ASPECT PHYSIQUE DU BUS VME

VME est un bus informatique. C'est un ensemble de liaisons électriques bien adaptées (réalisées par un circuit imprimé multicouche de qualité et des éléments de terminaison de lignes (terminaisons actives ou passives: ponts de résistances)) qui permettent de relier entre eux différents éléments d'un ordinateur. Ces différents éléments sont reliés au bus par des connecteurs dont l'ensemble constitue ce que nous appelons un fond de "panier" ("back plane" en anglais, littéralement plan arrière). Chaque emplacement ou logement ("slot") dans le fond de panier peut recevoir une carte électronique ("board"). Les cartes viennent s'enficher et se connecter dans les emplacements du fond de panier. Il y a des fils pour les adresses, les données, les signaux de contrôle et l'alimentation en courant électrique.

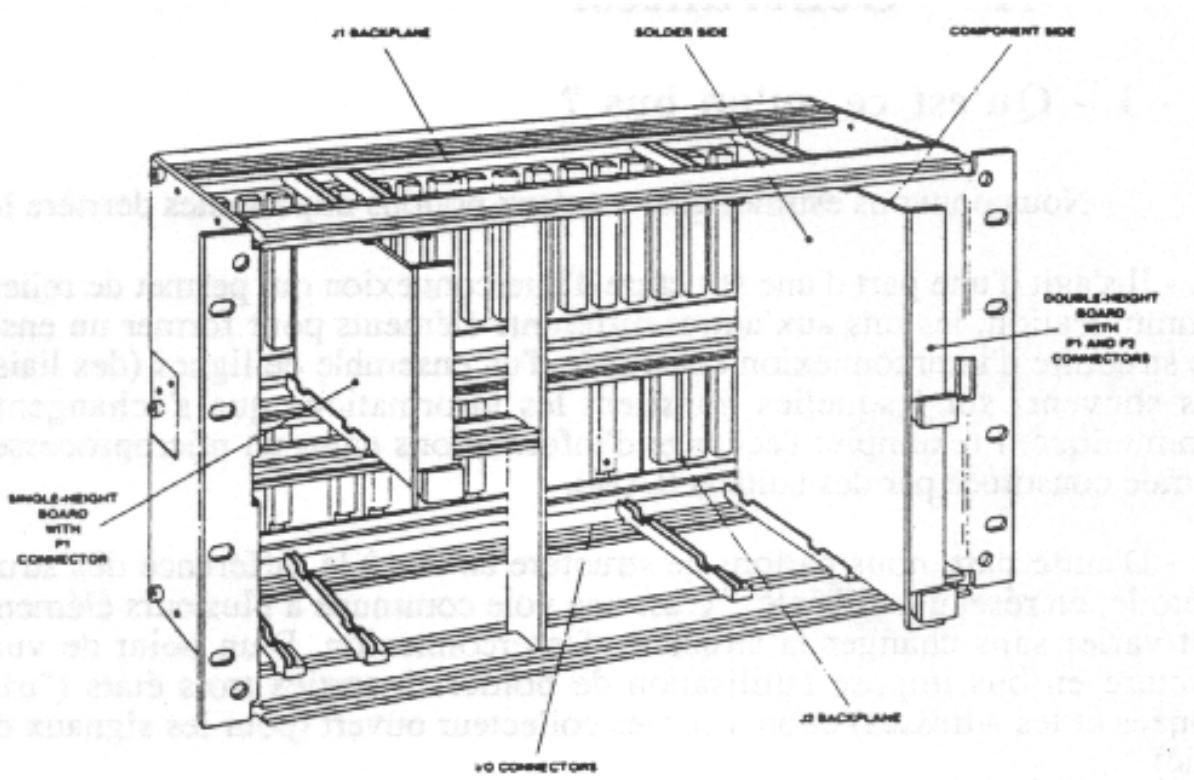


Figure 2 : Châssis VME

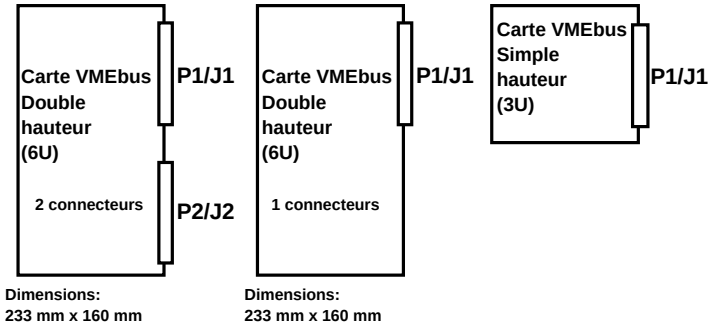


Figure 3 : Format des différentes cartes VME

3. QU'EST QUE VME ? HISTORIQUE

C'est la norme **IEEE P1014** ("Institute of Electrical and Electronics Engineers")**.**

Le bus **VME (Versa Module Eurocard)** révision A a été annoncé pour la première fois en octobre 1981 à Munich à l'occasion de "systems 81". Cette présentation a été faite conjointement par "feu" Mostek, Motorola et Philips/Signetics. Ce bus fut immédiatement adopté par Thomson/Efcis pour le développement de ses cartes 68000.

Aujourd'hui plus de **250 constructeurs** proposent plus de **5000 cartes**.

Une association regroupe les utilisateurs et les constructeurs VME: le **VITA** ("VME International Trade Association") qui compte actuellement plus de 150 membres.

Le bus VME a pour but d'offrir un support performant et fiable tout en étant compatible avec la technologie disponible et les besoins de ses utilisateurs. Ces deux critères ont amené le bus VME à évoluer régulièrement (voir fin du document dernières nouveautés).

Il est à noter que l'un des objectifs du bus VME est de laisser toute latitude de choix de composants et de méthodes au concepteur afin qu'il puisse optimiser les coûts ou les performances, ou les deux, sans restriction de compatibilités.

En 1982, la révision B a affiné les spécifications électriques des émetteurs et récepteurs de ligne, et a modifié les spécifications mécaniques afin de les rendre compatibles avec le format des cartes Europe IEC 297-3 ("International Electrotechnical Commission"). Tout comme la première révision, la révision B a été placée dans le domaine public.

En mars 1983, une standardisation IEEE a été programmée et le groupe de travail P1014 a été créé.

En collaboration avec l'IEC, un nouveau document, connu sous le nom de révision C1, a été publié. Les principales améliorations apportées sont une meilleure définition de certains chronogrammes, l'adjonction de nouveaux cycles ("Address

only", "Unaligned data transfers", ...), la redéfinition des accès en mode bloc, la création de nouveaux blocs fonctionnels ("Location monitor", par exemple).

Les raisons de cette évolution sont au nombre de quatre au minimum:
- amélioration des performances en terme de vitesse et de fiabilité,
- prise en compte de nouveaux processeurs (68020, 68030), voire de futurs processeurs,
- adjonction de nouvelles fonctionnalités,
- amélioration de la compatibilité entre cartes de différentes sources.

Suite à cette révision, le comité IEEE a rendu obligatoires, et non pas optionnelles, certaines caractéristiques afin de garantir la compatibilité entre cartes, et la norme 1014 a été approuvée le 12 mars 1987 par l'IEEE, puis le 11 septembre de la même année par l'ANSI ("American National Standards Institute").
Récemment la révision D est apparue. Elle apporte notamment les possibilités d'adressage sur 64 bits, le transfert de données sur 64 bits, le transfert de données en mode diffusé et le cycle de réessai.

4. CARACTERISTIQUE DU BUS VME

4.1. Bande passante du bus

Afin de déterminer la bande passante du bus, trois types de critères sont à prendre en compte:
- les chronogrammes de la norme qui prennent en compte les problèmes de propagation sur le bus,
- la réalisation pratique des modules d'interface du bus,
- les performances des processeurs ou des automates demandant le transfert, ainsi que le temps d'accès des unités acceptant le transfert.

Le calcul théorique du temps de cycle pour un transfert, sur le bus VME, hors phases d'arbitrage, est de 95 ns en lecture et 70 ns en écriture, ce qui correspond à une bande passante théorique de plus de 40 Méga octets par seconde, avec des transferts de mots longs.

En pratique, on obtient une bande passante inférieure pour trois raisons:
- la logique de commande du bus VME et de ses tampons introduit des retards (tampons 12 ns, décodage 30 ns, logique de commande 15 ns);
- le temps d'accès des unités esclaves (mémoire par exemple) peut varier de 45 à 120 ns, voire plus;
- le temps de prépositionnement et de maintien des adresses/données des processeurs, ainsi que les retards introduits par les logiques de décodage d'adresses de la carte maîtresse.

Ainsi avec un 68010 à 12.5 MHz, nous obtenons un temps de cycle lecture mémoire à travers le bus VME, de 400 ns, soit 5M octets par seconde. Avec un 68020 à 16 MHz, le temps de cycle est de 312 ns, soit 12.8 M octets par seconde. Enfin, certaines cartes équipées d'automates câblés et de mémoires tampon FIFO

("First In First Out") atteignent des débits pouvant aller jusqu'à 36 Méga octets par seconde. A titre de comparaison, le bus d'un PC AT débite au maximum à 1.2 Méga-octets/seconde, le bus d'un ordinateur VAX de Digital Equipement à 13.3 Méga-octets/seconde. Mais n'oublions pas que le bus VME d'un SUN3 est donné pour une vitesse de transfert de 6 Méga octets par seconde.

En accord avec les nouvelles spécifications VME64 (ou nouvelle révision D) du bus VME, plusieurs constructeurs proposent déjà des cartes affichant des débits maximums de l'ordre de 65 M octets par seconde

Avec des processeurs standards, les performances sont limitées par la vitesse du processeur lui-même. De plus, il faut ajouter, dans le cas d'une architecture multiprocesseur, le temps pris par les demandeurs et l'arbitre de bus (avec des boîtiers classiques environ 50 ns pour arbitrer, 80 ns pour prendre en compte la demande et pour indiquer que le bus est libre).

Pour améliorer les performances, les solutions utilisées sont, entre autres:

- Le remplacement des buffers (adresses, données, etc....) par des registres qui mémorisent les informations. Ainsi, le temps d'accès d'une carte mémoire peut passer de 170 ns en écriture à 40 ns. De même, une carte maîtresse peut terminer un accès en lecture dès la réception des données et les mémoriser tout le temps pendant lequel ces données seront nécessaire au processeur.
- L'augmentation de la fréquence des arbitres synchrones, ou l'utilisation des arbitres asynchrones avec des boîtiers très rapides afin de garantir le comportement métastable,
- Le "pipelining" qui consiste à terminer l'adressage d'un accès et à commencer le suivant alors que le cycle de transfert de données n'est pas encore terminé.

Toutes ces techniques nécessitent un nombre croissant de boîtiers, pour réaliser une interface VME. C'est pourquoi sont apparus des boîtiers contrôleurs de bus VME qui réalisent l'ensemble des fonctions (demande, arbitrage et commande de transfert) en un seul boîtier et avec des performances voisines des maxima théoriques du bus VME.

Citons entre autres les "VMEchip" de Motorola, le "FGA001/002" de Force Computers et surtout le "VIC068" conçu par un consortium de plus de trente utilisateurs du bus VME. et produit par Cypress (après rachat de la partie CMOS de l'activité VTC) et par la société canadienne Newbridge (représentée en France par Rep'France).

Pour la conception de cartes VME sans intelligence, il faut faire appel à des boîtiers beaucoup moins sophistiqués tels que :

PLX448 - boîtier programmable dédié aux applications d'interface de bus.
PLX464 - boîtier programmable dédié aux applications d'interface de bus.
VME1210/20 - VMEbus contrôleur et VMEbus maître.
VME2000 - VMEbus boîtier d'interface module esclave.
VME3000 - VMEbus générateur d'interruption.
VME4000 - VMEbus contrôleur d'interruption.

4.2. L'évolution des fonctionnalités

De nombreuses fonctionnalités sont apparues avec la révision C et sont ou

seront de plus en plus souvent mises en œuvre grâce aux boîtiers contrôleurs VME.

La notion de "location monitor" permet d'adresser plusieurs cartes simultanément et de les synchroniser ou de supporter des mécanismes tels que le passage de messages ("message passing") ou le "broadcasting". Les cycles d'adressage seuls, sans transfert de données, permettent soit de simplifier la conception des maîtres soit de réaliser des fonctions de synchronisation entre cartes par exemple.

La possibilité de réaliser plusieurs accès à des adresses consécutives, sans fournir les différentes adresses (mode bloc) permet de tirer partie de l'adressage en mode page des mémoires dynamiques ou des mémoires de type FIFO tout en diminuant le temps d'accès et le temps de prise du bus.

D'autres fonctionnalités apparaissent qui ne sont pas à proprement parler VME, au sens de la norme. Le "VIC068" permet, par exemple, l'écriture postée. Elle consiste, au niveau du maître, à mémoriser les données que le processeur veut écrire et à lui signifier immédiatement que l'accès est terminé, puis à réaliser quand cela est possible, l'accès physique sur le bus VME. Cette façon de procéder permet d'accéder sans cycle d'attente, à des périphériques même lents. Une autre possibilité du "VIC068" est l'utilisation d'instructions indivisibles à plusieurs adresses (CAS et CAS2 du 68020/68030). Lorsque le bus VSB ("Vme Subsystem Bus") est adjoint au VME, c'est pour supporter les très hauts débits , le bus VME se ramenant alors à un bus d'entrée-sortie et de synchronisation.

Avec l'évolution de la technologie, il est maintenant possible d'avoir à la fois de hauts débits sur le bus VME ainsi que des fonctionnalités de synchronisation évoluées. En tenant compte du fait qu'il est aisé d'avoir des mémoires doubles accès ou privées de grande capacité, le VME permet facilement l'implantation d'architectures multiprocesseurs, soit à exécution parallèle, soit en mode flot de données. Le bus VSB est néanmoins indispensable pour des extensions géographiques. La disponibilité de boîtiers contrôleurs de bus contribue non seulement à l'amélioration des performances, mais aussi à la garantie, pour les utilisateurs, d'une compatibilité totale pour un coût égal ou inférieur à celui d'architectures moins performantes.

Toutes ces caractéristiques tendent à positionner le bus VME comme un leader des années 1990, tant dans les applications industrielles qu'en télécommunications ou pour des stations de travail.

5. LE BUS VME

5.1. Introduction

Rappelons ce qui a conduit aux spécifications du bus VME dès le départ:

Ces spécifications ont pour but de définir un système d'interface capable d'interconnecter des unités de traitement de l'information, de stockage de données et de contrôle des périphériques dans une configuration à couplage étroit.

Le bus VME a été conçu par les industriels de l'informatique pour les industriels de l'informatique.

Ce système a été conçu avec les objectifs suivants:

- 1 Permettre à deux unités de communiquer entre elles sans gêner le

fonctionnement interne des autres unités connectées au VMEbus.

- 2 Créer un système dont les performances dépendent principalement des unités connectées au bus et non pas du bus lui-même.

- 3 Offrir au concepteur une grande latitude dans le choix des moyens qui lui permettent d'optimiser le coût et/ou les performances du système sans nuire à sa compatibilité.

Pour cela les caractéristiques électriques et mécaniques, les protocoles du système ont été spécifiés avec précision.

Les spécifications VME nous indiquent comment les choses doivent fonctionner et avec quelles caractéristiques, mais le concepteur garde une grande marge de manœuvre quant aux choix techniques.

5.2. Structure générale

Le VME est une structure de bus pour les systèmes 8, 16 et 32 bits. C'est un bus multiprocesseur permettant à plusieurs cartes maîtres de partager des ressources communes (mémoire, entrée-sortie) suivant une priorité d'accès hiérarchique ou tournante.

La structure du bus VME, qui dispose de quatre niveaux d'accès, est construite autour du concept "maître esclave»: le maître a le contrôle du bus, tandis que l'esclave après décodage de l'adresse le concernant, répond à la commande envoyée par le maître. Le cycle indivisible de lecture/modification/écriture du 68000 a été reporté sur le bus pour permettre l'utilisation de sémaphores pour se réserver, par exemple, une ressource commune dans un environnement multiprocesseur.

Sept niveaux d'interruption peuvent être contrôlés de façon centralisée ou répartie parmi les différents processeurs d'un même système.

Le fonctionnement en mode asynchrone du bus facilite l'installation dans un même système de processeurs, de mémoire et de dispositifs d'entrée-sortie ayant des caractéristiques de vitesses différentes.

5.3. Caractéristiques techniques

❖ Structure de base du VMEbus

Les lignes de signaux du VMEbus peuvent être classées en quatre catégories:

- a - Transfert de données:

Le transfert des données s'effectuent par l'intermédiaire d'un bus spécialisé: le DTB ("Data Transfert Bus"). Ce bus comprend le bus d'adresses (16, 24 ou 32 bits), le bus de données (8,16 ou 32 bits) et les signaux de commande associés.

- b - Arbitrage du DTB:

Il nous faut définir un peu de vocabulaire avant de pouvoir préciser ce qu'est l'arbitrage.

MODULE: C'est l'ensemble des composants électroniques remplissant un rôle

fonctionnel déterminé. Une carte peut contenir plusieurs modules.

DEMANDEUR: C'est le module qui est capable de demander le contrôle du bus de transmission de données (DTB).

MAITRE: C'est le module qui a la capacité de demander le contrôle du DTB par l'intermédiaire de son DEMANDEUR.

L'ARBITRE du DTB permet de transmettre le contrôle du DTB d'un maître à un autre et faire en sorte qu'un seul maître ait le contrôle du DTB à un instant donné.

- c - Interruption prioritaire:

Le système d'interruption prioritaire du VMEbus permet aux unités de demander l'interruption des activités normales du bus, ces unités sont alors prises en charge par un contrôleur d'interruptions.

Ces demandes d'interruption peuvent avoir sept niveaux de priorité.

Les modules fonctionnels associés sont appelés GENERATEURS et CONTROLEURS D'INTERRUPTIONS et utilisent l'ensemble des lignes de signaux appelé BUS d'interruption.

- d - Utilitaires:

Le bus des utilitaires comprend une ligne d'horloge, une ligne de remise à zéro du système, une ligne de défaut système et une ligne de défaut de l'alimentation alternative du secteur.

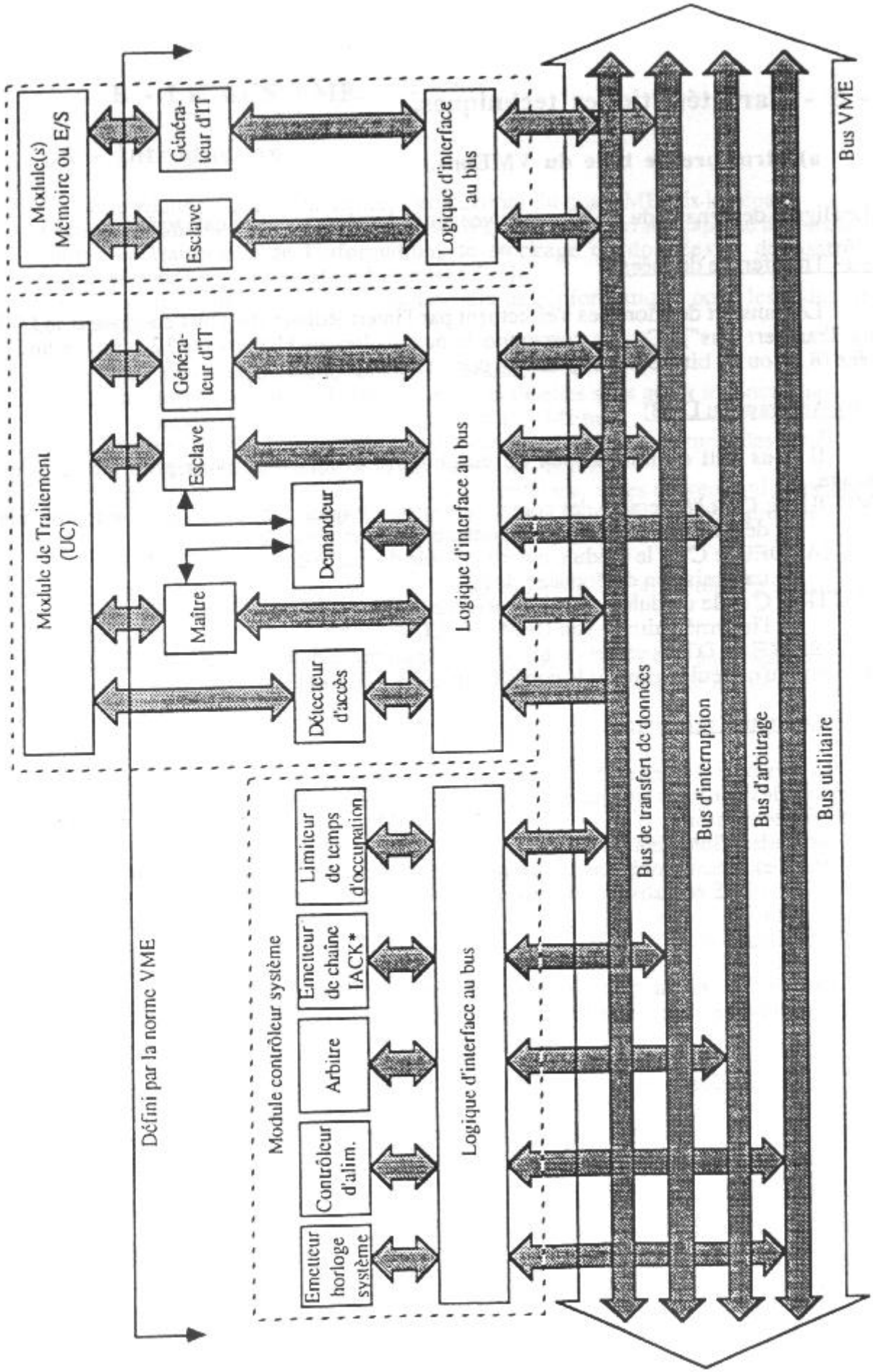

Module(s) Mémoire ou E/S
Généra-teur d'IT
Esclave
Logique d'interface au bus
Module de Traitement (UC)
Généra-teur d'IT
Esclave
Demandeur
Maître
Détecteur d'accès
Logique d'interface au bus
Module contrôleur système
Limiteur de temps d'occupation
Emetteur de chaine IACK*
Arbitre
Contrôleur d'alim.
Emetteur horloge système
Logique d'interface au bus
Défini par la norme VME
Bus de transfert de données
Bus d'interruption
Bus d'arbitrage
Bus utilitaire
Bus VME

Figure 4 : Diagramme fonctionnel du VME

❖ Transfert de données

a) Introduction

Le VMEbus comprend un bus parallèle non-multiplexé de transmission de données (DTB) rapide, fonctionnant en mode asynchrone. Le DTB permet à un processeur ou à une unité d'accès direct à la mémoire (DMA) de sélectionner le périphérique ou l'emplacement mémoire désiré et de procéder au transfert des données vers ou à partir de cet emplacement ou périphérique.

Le DTB possède l'ensemble d'options, suivant:

- D8, D16 ou D32 largeur du chemin de données (D "Data") et plus avec la révision D: D64.
- A16, A24 ou A32 largeur du chemin d'adresses (A "Address") et plus avec la révision D: A64.
- BTO (x) "Time-out" du bus ("Bus Time Out") x durée en microsecondes.
- BLT Transfert en bloc ("BLock Transfer") ou accès séquentiel: option SEQ de la révision B).
- RMW Transfert avec lecture-modification-écriture ("Read-Modify-Write").
- ADO Adressage seul sans transfert de données ("ADdress Only").
- UAT Transfert d'octets non alignés ("UnAligned data Transfer").

b) Fonctionnement du DTB ("Data Transfer Bus")

Chaque échange de données sur le DTB a lieu entre un MAITRE et un ESCLAVE.

ESCLAVE: Module capable de décoder les lignes d'adresses du VMEbus et de répondre à un maître en acceptant ou en rejetant la transmission des données par l'intermédiaire des lignes de réponse DTACK et BERR.

DTACK: Acquittement d'une transmission reconnue.

BERR: Transmission refusée, l'adresse présentée n'est pas à l'intérieur des limites d'adressage de l'esclave.

Le MAITRE est le module qui contrôle l'échange de données et l'ESCLAVE le module adressé qui répond. Nous pouvons remarquer que bien évidemment certaines cartes peuvent contenir à la fois un module maître et un module esclave (mémoire locale d'une carte processeur accessible par le VME).

Selon les options choisies, le DTB se compose des lignes suivantes:

- * - Contrôlées par le MAITRE :
  - 15, 23 ou 31 lignes d'adresses (A01 à A31).
  - 6 lignes de modificateurs d'adresses (AM0 à AM5).
  - Signal "strobe" d'adresse (AS*) (* signifie actif au niveau bas).
  - Signal "strobe" d'octet(s) de données (DS0*).
  - Signal "strobe" d'octet(s) de données (DS1*).
  - Sélection de MOT LONG (16/24/32 bits) (LWORD*).
  - Sélection de lecture/écriture (WRITE*).

- * - Contrôlées par le l'ESCLAVE :
  - Erreur bus (BERR*).
  - Acquittement de transmission de données (DTACK*).
  - 8, 16 ou 32 lignes de données (D00 à D31).

Remarque: Le signal BERR* n'est émis par l'esclave que lorsque l'esclave est adressé et que l'adresse à l'intérieur de l'esclave est incorrecte. Si l'adresse émise sur le bus ne correspond à aucun esclave alors, c'est le maître qui doit posséder un chien de garde pour indiquer à son demandeur que l'adresse émise est incorrecte lorsqu'il n'y a pas de réponse au bout d'un temps défini ou le bus dispose d'un bus timer.

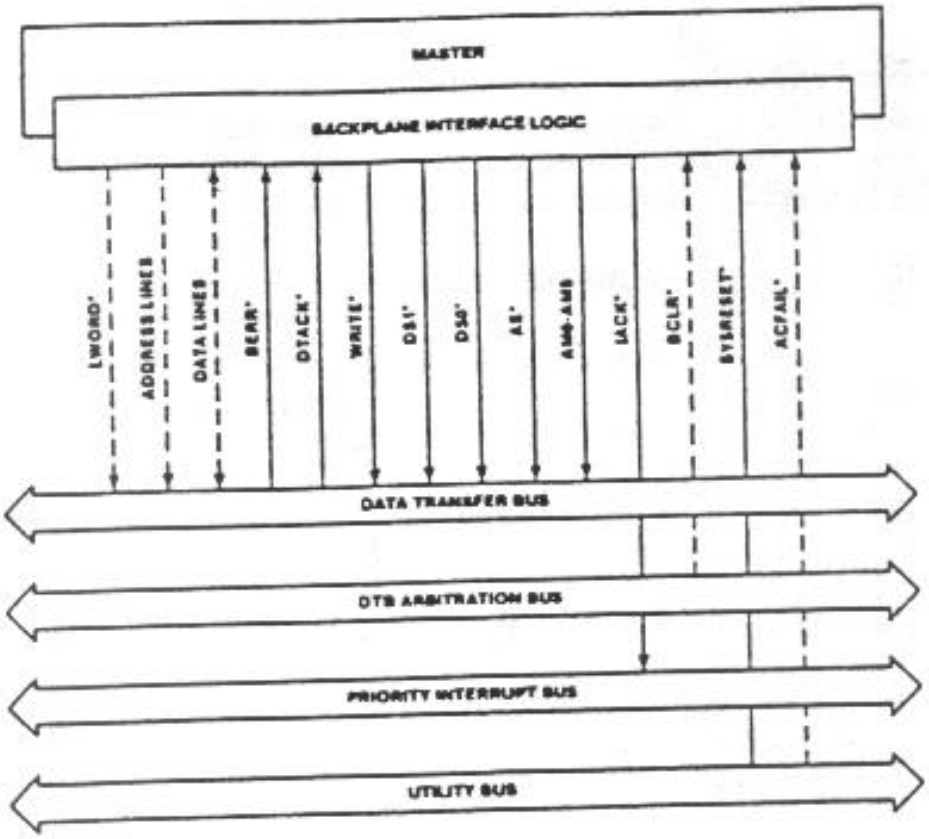


Fig 5
Block Diagram: Master

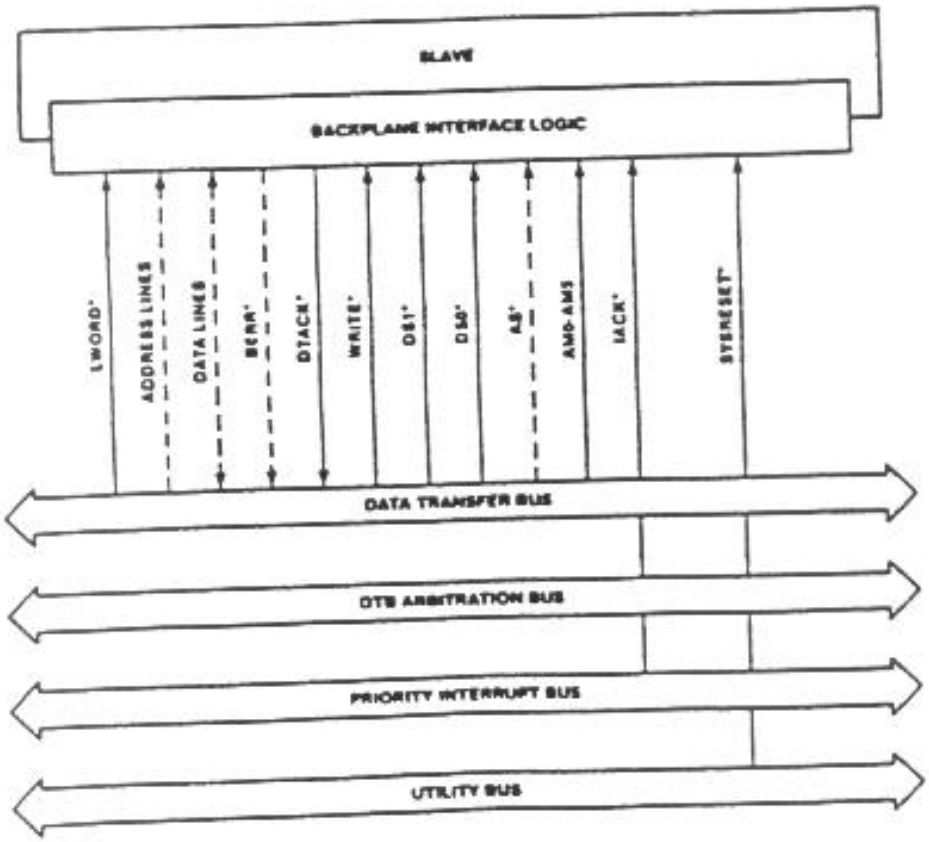


Fig 6
Block Diagram: Slave

Figure 5 : Diagramme bloc d'un maître / esclave VME

c) Lignes d'adresses (A01-A31)

Tout comme pour le 68000 la plus petite unité adressable est l'octet et la comptabilité d'adresses s'effectue toujours en octets. A chaque octet est associée une adresse binaire unique. De plus A00 n'existe pas, il est remplacé par DS0*, DS1*, A01 et LWORD*. En fait le maître indique sur les lignes A02-A31 le groupe de quatre octets qui doit être accéder et les quatre lignes DS0*, DS1*, A01 et LWORD* servent à la sélection de l'octet à l'intérieur de ce groupe.

d) Lignes des modificateurs d'adresse (AM0-AM5)

Avec ces lignes le maître peut envoyer des informations supplémentaires lors d'un transfert de données, informations qui précisent certaines fonctions:

- Découpage du système:

Les esclaves appartenant au système peuvent être configurables statiquement ou dynamiquement de manière à ne répondre qu'à un seul code de modification d'adresse. Par exemple, s'il y a plusieurs maîtres dans le système VMEbus chacun peut obtenir son propre code pour accéder aux esclaves. Ceci permet de découper le système et d'éviter qu'un seul maître en panne n'arrête tout le système (il y a des sous-ensembles maître/esclaves).

- Sélection de la topographie mémoire:

Les esclaves peuvent être conçus de manière à répondre à différentes adresses en fonction du modificateur d'adresse reçu. Ceci permet au maître qui utilise le bus de placer les ressources du système à des emplacements sélectionnés de la mémoire (ou de les éliminer de la topographie mémoire) en fournissant différents codes de modification d'adresse).

- Accès privilégiés:

Comme les esclaves peuvent ne répondre qu'à certains modificateurs d'adresses et ne pas réagir à d'autres, un grand nombre de niveaux de privilèges pourraient être établis. Chaque maître fournirait un modificateur d'adresse indiquant son niveau de privilège au moment d'accéder à un esclave. L'esclave ne répondrait que sur réception d'un code AM approprié.

- Types de cycle:

Les codes AM peuvent être utilisés pour spécifier un type spécial de cycle de transmission. Le VMEbus spécifie un type de cycle spécial. L'utilisateur peut utiliser d'autres codes pour spécifier d'autres types. Ce type de cycle spécial est un cycle d'accès séquentiel ascendant. Il y a quatre codes AM pour l'accès ascendant. Lorsque l'un de ces codes est placé sur le bus, l'esclave adressé verrouille l'adresse dans un compteur qui est incrémenté de 1,2 ou 4 après chaque transfert d'octet, de mot (2 octets) ou de long mot (4 octets).

- Gestion mémoire répartie:

La logique de gestion de la mémoire est utilisée dans beaucoup de systèmes pour assurer l'affectation et la translation dynamiques de segments de mémoire. Un ensemble de ces segments est affecté à chaque tâche active. Chaque fois que l'exécutif temps réel passe d'une tâche à une autre, il doit changer le contenu des registres de segments ou bien sélectionner un autre jeu de registres de segments (cette dernière solution est beaucoup plus rapide).

Les codes de modification d'adresse peuvent servir de sélecteurs de registres de segments. Dans ce cas, le maître place des codes AM sur le bus pour indiquer à sa logique de gestion de la mémoire sur les cartes esclaves quel jeu de registres de segments doit être utilisé.

- Domaine d'adressage:

Le VMEbus dispose de 31 lignes d'adresses permettant l'adressage direct de plus de quatre milliards d'octets. Cependant, pour la plupart des esclaves, la logique supplémentaire nécessaire au décodage des 31 lignes d'adresses est une dépense inutile.

C'est pourquoi, le VMEbus définit trois domaines d'adressage:

- l'adressage court 64 Kilo octets.
- l'adressage standard 16 Méga octets.
- l'adressage étendu 4 Giga octets.

Un groupe de codes de modification d'adresse est prévu pour chaque type d'adressage. Un esclave qui reçoit un code modificateur d'adresse courte ignore les lignes A16 à A31;Celui qui reçoit un AM d'adresse standard ignore les lignes A24 à A31. Les esclaves qui ne décodent pas les lignes d'adresses A24 à A31 ne doivent pas répondre aux codes AM étendus. Les esclaves ne décodant pas les lignes A16 à A31 ne doivent répondre ni aux codes AM standards, ni aux codes AM étendus.

Les spécifications VME nous donne un tableau des codes AM déjà définis (Table 3 page 42 "The VMEbus specification").

Pour permettre à l'utilisateur de configurer les codes nécessaires à son système sur un esclave, il suffit de disposer d'une PROM. Les lignes d'adresses de la PROM étant reliées aux AM.

Fréquemment les trois codes fonctions FC0-FC2 des 680x0 d'une carte maître sont reliés aux trois AM (AM0-AM2).

e) Lignes de données (D00-D31)

Des systèmes peuvent être construits à partir de fonds de panier 16 bits (D00-D15) ou 32 bits (D00-D31) de données. Lorsqu'il n'y a que 16 lignes de données un maître ne peut accéder qu'à deux octets simultanément, tandis qu'avec 32 lignes il est possible de transférer quatre octets d'un seul coup.

f) Lignes de contrôle des échanges de données

AS*, DS0*, DS1*, LWORD*, WRITE*, BERR* et DTACK*.

AS*: Cette ligne est le "strobe" d'adresse. Cette ligne signale à tous les modules ESCLAVES que l'adresse s'est stabilisée et peut être envoyée dans les registres de maintien. Ce signal reste également actif pendant toute la durée d'un transfert en mode bloc.

Les lignes **DS0*, DS1*, LWORD* et A1** permettent la sélection des octets (1,2,3 ou 4) à transférer (voir tables 1 page 39, 2 page 41 et 4 page 44 des spécifications VME).

DS0* et DS1*: En plus de leur fonction de sélection des données à transmettre, ces deux lignes servent de strobe. Dans le cas d'un cycle d'écriture le premier front descendant d'une de ces deux lignes indique que le maître vient de placer des données stabilisées. Dans le cas d'un cycle de lecture le premier front montant signale à l'esclave qu'il peut enlever ses données du bus de données.

LWORD*: Cette ligne indique que le transfert se fera sur deux, trois ou quatre octets (Associé aux lignes DS0,DS1 et A1 donne la sélection des octets).

WRITE*: Cette ligne permet au maître d'indiquer s'il s'agit d'un cycle d'écriture ou d'un cycle de lecture. Un niveau haut correspond à un opération de lecture et un niveau bas à une opération d'écriture.

DTACK*: Cette ligne est placée à l'état bas par l'esclave pour indiquer qu'il a pris en compte les données lors d'un cycle d'écriture. Lors d'un cycle de lecture, l'esclave force cette ligne à zéro pour signaler qu'il vient de placer des données stabilisées sur le bus de données**.**

BERR*: Cette ligne est forcée à l'état bas par l'esclave ou par le "timer" du bus pour signifier au maître que le transfert de données demandé ne peut se réaliser. Par exemple, quand un maître tente d'écrire à un emplacement mémoire qui est à lecture seule, l'esclave concerné fait passer cette ligne à l'état bas. Quand le maître essaye d'accéder à une adresse qui n'est supportée par aucun esclave, alors le "timer" du bus active la ligne BERR* après un temps défini. Le "timer" du bus peut être ou non implémenté: c'est un module fonctionnel optionnel.

Lors d'une demande de transfert, il y a soit DTACK*(acceptation), soit BERR*(refus).

6. ARBITRAGE

6.1. Prise de contrôle du bus de transmission de données

Avant de pouvoir transmettre des données sur le DTB, le maître doit obtenir la permission d'utiliser le bus, car il peut arriver que plusieurs maîtres désirent se servir simultanément du bus.

Le processus qui permet de déterminer quel maître a le droit d'utiliser le bus est

appelé arbitrage.

Bornons nous à signaler que l'arbitrage peut se faire de deux façons différentes:`

- L'arbitrage survient pendant le dernier transfert de données.
- L'arbitrage survient après le dernier transfert de données.

Tout maître doit attendre jusqu'à ce qu'il ait reçu l'autorisation d'utiliser le bus et jusqu'à ce que AS* soit au niveau haut avant de transmettre sur le bus.

6.2. Arbitrage du bus de transmission de données

❖ Introduction

Dans les systèmes multiprocesseurs, il est fréquent que différents processeurs se partagent des ressources communes. Dans le VMEbus la ressource essentielle est le DTB qui sert de moyen d'accès aux autres ressources. Par conséquent, le sous-système d'arbitrage du bus du système VMEbus répond à ce besoin. La caractéristique essentielle de l'arbitrage doit être la rapidité d'affectation du bus.

Le sous-système d'arbitrage du VMEbus est destiné à:

- éviter l'accès simultané de deux maîtres au bus.
- planifier les demandes simultanées émises par plusieurs maîtres pour optimiser l'utilisation des ressources.

C'est le module fonctionnel ARBITRE qui remplit cette fonction de trois façons possibles:

-option SGL ("SinGle Level")(option ONE de la révision B).
-option PRI ("PRIoritized").
-option RRS ("Round Robin Select").

L'arbitre est toujours placé dans la position la plus à gauche du châssis. Il fait partie du module contrôleur système.

Les demandes d'accès au bus se font avec une certaine priorité. Celle-ci dépend de la ligne à collecteur ouvert (OU câblé) de demande de bus (BR0-BR3, BR: "Bus Request") à laquelle le module est connecté et de sa position à l'intérieur du châssis. Plusieurs maîtres peuvent être connectés à la même ligne BRx* (OU câblé). A chaque ligne BR* correspond une ligne BGx* d'autorisation d'accès au bus qui relie en "daisy chain" tous les maîtres et accorde le bus par simple priorité sérielle, établissant ainsi un deuxième niveau de priorité suivant la position de la carte dans le châssis.

L'option arbitre SGL (un seul niveau de priorité) ne satisfait que les demandes transmisses sur BR3* et détermine les priorités suivant le principe de la connexion en guirlande ("daisy chain"). C'est la position du module dans le châssis qui donne la priorité.

L'option arbitre PRI attribue toujours le bus sur la base d'un système de priorité

fixe où chacune des quatre lignes de demande du bus possède une priorité fixe, de la priorité la plus élevé (BR3*) à la priorité la moins élevé (BR0*).

L'option arbitre RRS ("Round Robin Select") attribue le bus selon un système de priorité tournante. Si le maître qui est en possession du bus à le niveau de priorité "n", la plus haute priorité suivante sera celle qui a le niveau "n-1" et ainsi de suite.

Les lignes du bus d'arbitrage sont donc:

* Celles pouvant être activées par un demandeur du DTB:
  - une des lignes BR0*-BR3* de demande de bus ("Bus Request").
  - une des lignes BG0out*-BG3out* d'autorisation d'accès au bus ("Bus Grant").
  - la ligne d'occupation du bus BBSY* ("Bus buSY").

* Celles pouvant être activées par l'arbitre:
  - la ligne de remise à zéro du bus BCLR ("Bus CLeaR").
  - les quatre lignes d'entrée d'affectation du bus de l'emplacement A1 (BG0in*-BG3in*).

Remarque: Si une carte (ou un emplacement vide) n'utilise pas certaines lignes de demande du bus, les lignes d'entrée d'affectation du bus BGxin doivent être connectées (par cavaliers) aux lignes de sortie respectives BGxout.

Deux autres lignes sont liées au système d'arbitrage. Ce sont :
- la ligne de remise à zéro du système SYSRESET* (SYStème RESET").
- la ligne de défaut d'alimentation ACFAIL* ("Alternative Current FAIL").

❖ Lignes de demande et d'affectation du bus

Les lignes de demande du bus permettent à chaque DEMANDEUR de solliciter l'utilisation du bus de transmission de données (BR0*-BR3*). Les lignes d'affectation du bus (BG0*-BG3*) permettent à l'ARBITRE de satisfaire cette demande.

Cependant si nous explorons le mode "daisy chain» d'affectation du bus, nous pouvons arriver à un point où le niveau des signaux sur la "daisy chain» ne correspond plus à celui de la ligne commandée par l'arbitre. Ceci est dû au fait que chacune de ces lignes entre dans un emplacement déterminé sur ses broches BGxin*, mais quitte cet emplacement sur les broches BGxout*.

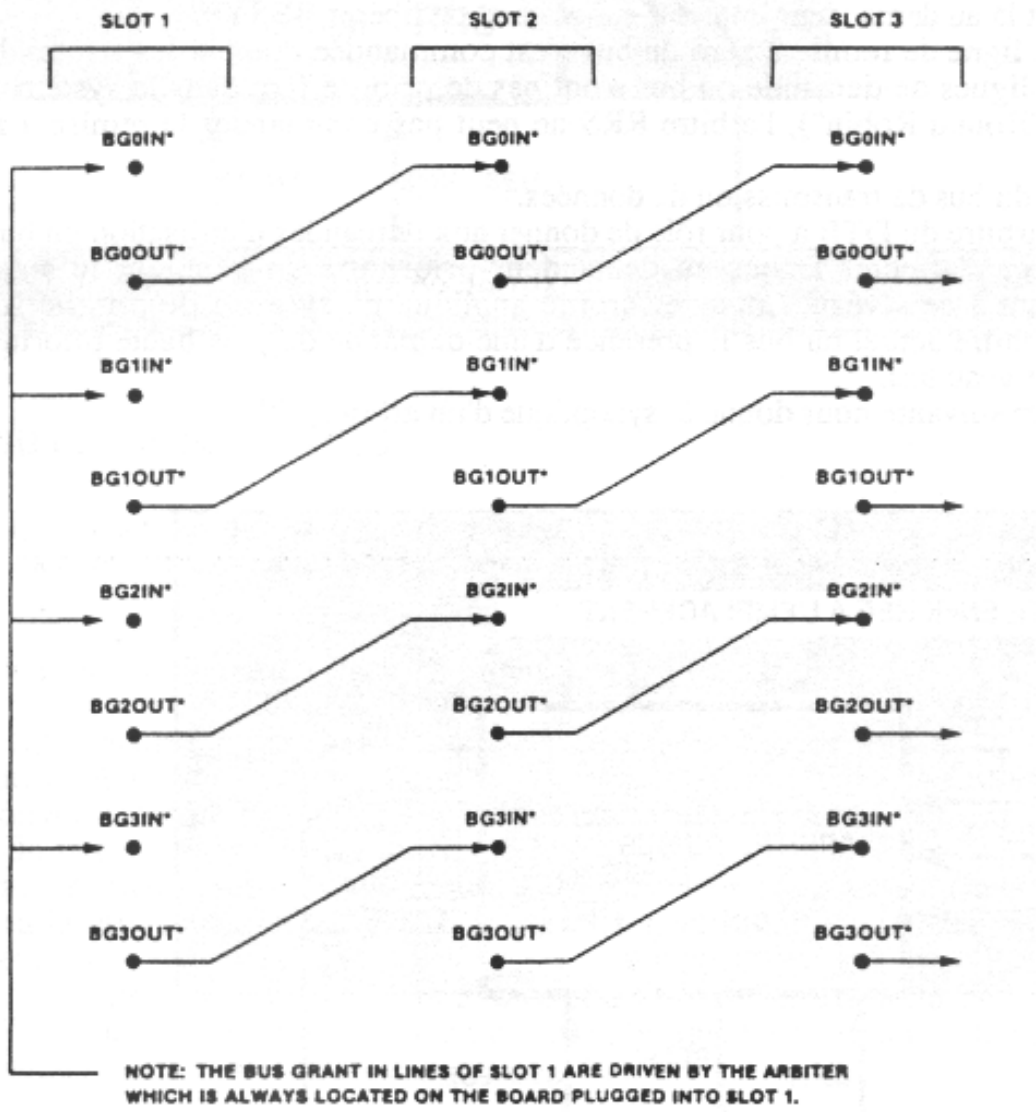


Figure 6 : Daisy chain BGIN*/BGOUT*

Rappelons que si une carte n'utilise jamais un niveau de demande ou d'affectation particulier, le signal est transmis par cavalier. Lorsque la carte utilise un niveau de demande ou d'affectation, le signal BGxin* est conditionné sur la carte. Si un demandeur vient d'émettre une demande d'utilisation du bus, il n'y a pas de niveau bas sur BGxout. S'il n'y a pas de demande en instance, un niveau bas apparaît sur BGxout après réception de BGxin* au niveau bas. Si un logement VME ne contient pas de carte, les signaux d'affectation doivent être transmis au moyen de cavaliers ("jumpers"). Les spécifications mécaniques du fond de panier indique les connexions cavalier à prévoir sur chaque logement.

Rappelons que l'arbitre doit se trouver à l'emplacement A1 (“slot” 1 le plus à gauche du châssis) et commande les broches BGxin* de sorte que tous les demandeurs de la carte qui se trouve à l'emplacement 1 peuvent suivre le même processus que sur une autre carte. Ceci permet d'uniformiser la structure de

l'interface du VMEbus sur chaque carte et de modulariser les fonctions de l'arbitre et du demandeur.

Ligne d'occupation du bus BBSY* ("Bus buSY"):

Lorsqu'un demandeur reçoit de la "daisy chain" le contrôle du DTB, il force BBSY* au niveau bas. Il garde le contrôle du bus jusqu'à la libération de BBSY*.

Ligne de remise à zéro du bus BCLR* ("Bus CLeaR"):

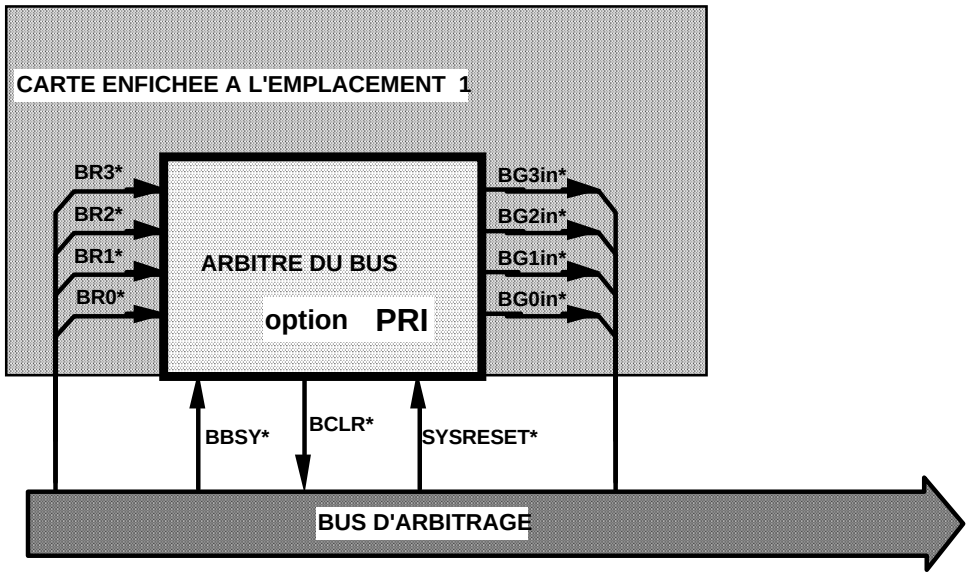
L'arbitre PRI emprunte la ligne de remise à zéro du bus pour signaler au maître qui a actuellement le contrôle du bus, qu'une demande de plus haute priorité est en attente. A ce moment le maître courant n'a pas de temps limite pour passer le contrôle mais ne peut effectuer plus de 256 transferts. En principe, il continue à transmettre jusqu'à ce qu'il rencontre un point d'interruption convenable, et permet à ce moment là au demandeur implanté sur sa carte de libérer BBSY*.

La ligne de remise à zéro du bus n'est commandée que par les arbitre du type PRI. Comme les lignes de demande du bus n'ont pas de priorité fixe dans le système d'arbitrage circulaire ("Round Robin"), l'arbitre RRS ne peut pas commander la remise à zéro du bus (BCLR*).

Arbitre du bus de transmission de données.

L'arbitre du DTB a pour rôle de donner aux demandes d'utilisation du bus un niveau de priorité et d'affecter le bus au demandeur prioritaire en générant le signal BGxin* correspondant à ce niveau. Lorsque l'arbitre applique un système de priorité fixe (PRI), il signale au maître actuel du bus la présence d'une demande de plus haute priorité en forçant BCLR* au niveau bas.

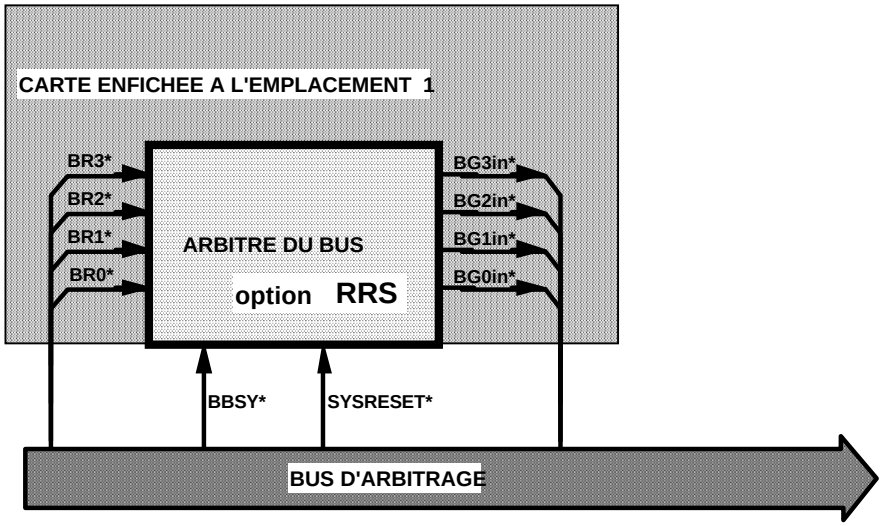
La figure suivante nous donne le synoptique d'un arbitre PRI



ARBITRE DU BUS, option PRI

Figure 7 : Arbitrage PRI

et la figure suivante celui d'un arbitre RRS, c'est à dire utilisant un système de priorité circulaire.



ARBITRE DU BUS, option RRS

Figure 8 : Arbitrage RRS

Notons qu'il n'active pas la ligne de remise à zéro du bus. Dans ce cas le maître qui occupe le bus a après libération du bus le niveau de priorité le plus bas.

Demandeur du bus de transmission de données.

Chaque maître possède une unité de demande d'accès au bus VME. Chaque demandeur du système doit:

- convertir la condition bus demandé par maître présente sur la carte en une demande d'utilisation du bus.
- prendre en compte le signal entrant d'affectation du bus pour le niveau concerné et, si le maître sur la carte ne désire pas le bus, émettre le signal de sortie d'affectation du bus pour le même niveau ou
- si le maître sur la carte désire le bus, convertir l'affectation du bus en un verrou interne qui fournit la réponse bus affecté au maître et le signal d'occupation du bus (BBSY*) tant que le maître maintient sa demande d'utiliser le bus.

Le demandeur peut libérer le bus de 2 façons:

- option RWD.
- option ROR.
- option FAIR.

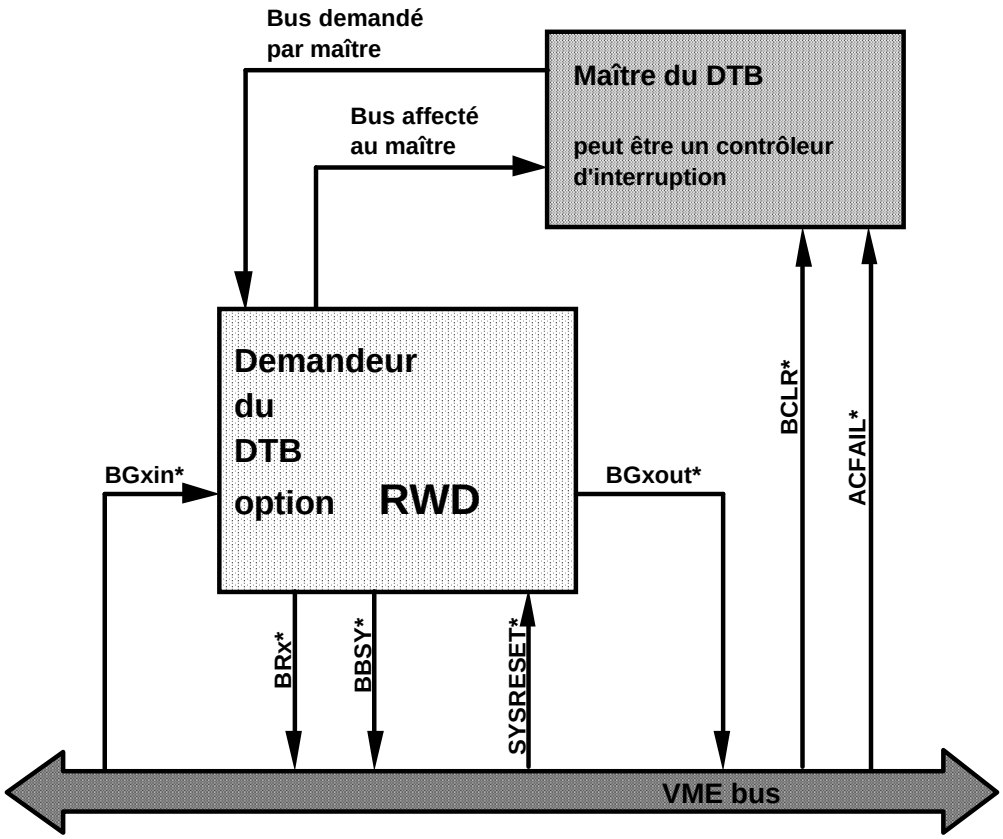
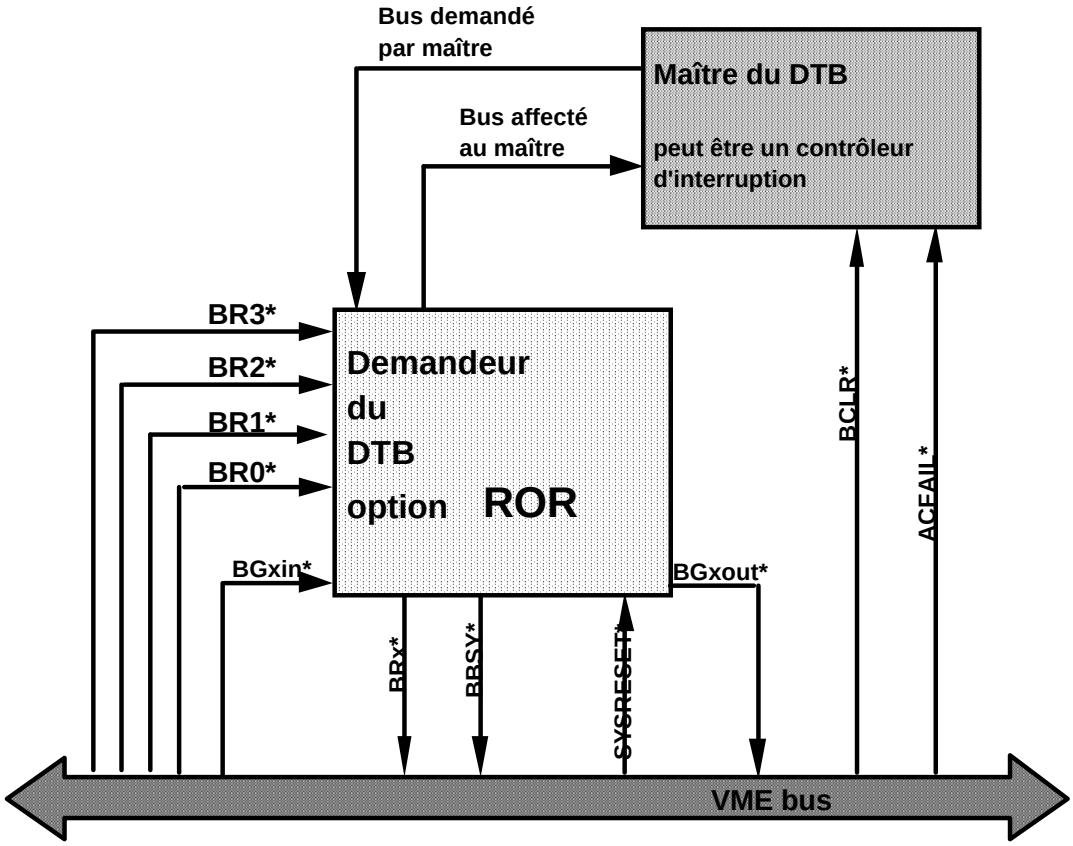


Figure 9 : Arbitrage RWD

Dans le cas du demandeur le plus simple (option RWD: "Release When Done" ou libération à la fin de l'utilisation) BBSY* est libéré quand le maître relâche sa demande d'utilisation du bus. Par contre, dans les systèmes qui nécessitent une vitesse de transmission maximale, un demandeur un peu plus complexe (option ROR "Release On Request ou libération sur demande) qui ne libère pas BBSY* quand le maître retire sa demande peut être mis en œuvre. Ce demandeur surveille les 4 lignes de demande du bus (BR0* à BR3*) et ne libère BBSY* que si une autre demande est en attente. L'utilisation de cette dernière option permet de réduire le nombre d'arbitrages effectués par un maître qui absorbe une grande partie du trafic du bus.



Demandeur, option ROR

Figure 10 : Arbitrage ROR

Nous pouvons remarquer sur les deux figures précédentes que chaque maître doit surveiller les deux signaux du VMEbus ACFAIL* et BCLR*, qui indiquent au maître la présence d'un besoin d'utilisation du bus supérieur au sien. Dans la cas de BCLR*, la conception du maître détermine combien de temps le maître peut garder le contrôle du bus. Par exemple, si un maître fournit un interface DMA à une unité très rapide telle qu'un disque dur, le maître ne peut abandonner le bus pendant un transfert de secteur sans risque de perte de données. Le maître doit garder le bus jusqu'à ce que le transfert de secteur soit terminé. Le signal ACFAIL* signale aux maîtres la détection d'un défaut d'alimentation courant alternatif et tous les problèmes que le maître peut rencontrer en abandonnant le bus sont insignifiants par rapport aux besoins du système global. Même dans ce cas le maître dispose de 200 microsecondes pour abandonner le DTB, ce qui devrait permettre de terminer normalement l'opération en cours.

Normalement chaque demandeur est associé à un maître particulier à qui il sert d'interface avec le bus d'arbitrage. Bien que le plus souvent la relation soit du type biunivoque, un maître qui nécessite une multiplicité de niveaux de demande du bus peut avoir une multiplicité de demandeurs. Dans ce cas, chaque demandeur demande le bus à un niveau différent. Les transmissions de données de niveau plus élevé peuvent alors être effectuées à un niveau de demande et celles de priorité moins élevé à un niveau inférieur. Ceci n'est réalisable qu'avec les arbitres PRI.

❖ Principes de fonctionnement

Arbitrage de deux niveaux différents de demande de bus.

Chaque maître A et B force simultanément sa ligne de demande du bus, le demandeur A commande BR1* et le demandeur B commande BR2*.

L'arbitre détecte en même temps un niveau bas sur BR1* et sur BR2*, mais il ne force que BG2in* de l'emplacement 1 à l'état bas parce que BR2* a la priorité la plus élevée. Une fois que le signal s'est propagé au travers du "daisy chain" jusqu'au demandeur B, celui-ci répond au niveau bas de BG2in* en forçant BBSY* à l'état bas. Puis il libère la ligne BR2* et signale à son maître B que le bus est disponible. Après avoir détecté un niveau bas sur BBSY*, l'arbitre force BG2in* haut. Quand le maître B a terminé sa (ses) transmission(s) de données, il libère BBSY* à condition que BG2in* soit remonté.

L'arbitre interprète la libération de BBSY* comme un signal pour arbitrer les demandes du bus. BR1* étant bas, l'arbitre accorde le bus au demandeur A en forçant BG1in* au niveau bas. Le demandeur A répond en forçant BBSY* au niveau bas. Quand le maître A a fini de transmettre, il libère BBSY* à condition que BG1in* soit remonté. A partir de ce moment comme aucune ligne BRx n'est active, l'arbitre est au repos jusqu'à l'envoi d'une nouvelle demande.

Notons que cette description est valable pour les deux options PRI et RSS, sauf si l'on considère un arbitre RRS avec une dernière demande active au niveau 2. Dans ce cas, l'arbitre traite d'abord la demande BR1*, puis la demande BR2*.

Arbitrage de deux demandes du bus sur la même ligne de demande du bus.

Un demandeur option ROR et un demandeur option RWD envoient simultanément des demandes à un arbitre sur une ligne de demande commune.

Dans cet exemple, l'arbitre et le demandeur option RWD sont placés sur la carte de l'emplacement 1, le demandeur option ROR est placé au deuxième emplacement. Chacun des demandeur active simultanément la ligne BR1*. L'arbitre qui se trouve à l'emplacement 1 détecte un niveau bas sur BR1* et comme BBSY* est au niveau haut, il force BG1in* au niveau bas en direction de son propre emplacement.

Lorsque le demandeur A qui se trouve à l'emplacement 1 détecte un niveau bas sur BG1in*, il répond en forçant BBSY* au niveau bas. En même temps il signale au maître A que le bus est disponible et libère également BR1*. Après avoir détecté la présence de BBSY* l'arbitre force BG1in* au niveau haut. Quand le maître A a fini de transmettre il force le signal bus demandé par maître au niveau bas qui reçu par le demandeur A provoque la libération du signal BBSY.

L'arbitre interprète la libération de BBSY* comme un signal pour arbitrer les demandes du bus. Comme la ligne BR1* est toujours au niveau bas, l'arbitre force à nouveau BG1in* au niveau bas. Quand le demandeur A détecte un niveau bas sur BG1in*, il force BG1out* au niveau bas puisqu'il n'a pas besoin du bus. Le demandeur B détecte alors un niveau bas sur son BG1in* qu'il ne transmet pas à l'emplacement suivant et force BBSY* au niveau bas.

Quand l'arbitre détecte un niveau bas sur BBSY*, il force BG1in* au niveau

haut, ce qui oblige à son tour le demandeur A à forcer son BG1out* au niveau haut. Après avoir détecté un niveau haut sur son BG1in*, lorsque le demandeur B reçoit un signal BUS demandé bas sur sa carte il sait que le maître a fini d'utiliser le bus mais ne libère pas BBSY* puisque c'est un demandeur du type ROR. Ainsi au cas où son maître désirerait à nouveau utiliser le bus, il ne serait pas nécessaire d'effectuer un nouvel arbitrage. Mais dans notre exemple le demandeur A à de nouveau forcé BR1in* au niveau bas pour indiquer qu'il voulait à nouveau utiliser le bus et le demandeur B du type ROR surveille les lignes de demande de bus et libère sur l'activation de BR1* la ligne BBSY*. Ainsi l'arbitre peut accorder le bus au demandeur A.

7. GESTION DES INTERRUPTIONS

Deux types de systèmes de gestion des interruptions peuvent être envisagés:

- les systèmes à contrôleur unique; ils possèdent un processeur de contrôle qui reçoit et traite toutes les interruptions du bus.
- les systèmes répartis; ils possèdent deux processeurs ou plus qui reçoivent et traitent les interruptions du bus.

Le bus de transmission de données, le bus d'arbitrage et le bus d'interruption interviennent tous dans la génération et le traitement des interruptions du bus.

7.1. Lignes du bus d'interruption

IRQ1* à IRQ7* lignes de demandes d'interruption.
IACK* ligne d'acquittement des interruptions.
IACKin*/IACKout* lignes de signaux de la "daisy chain" d'acquittement.

Chaque ligne de demande d'interruption peut être forcée à l'état bas par un **générateur** qui demande une interruption. Dans un système à contrôleur unique, ces lignes de demande d'interruption ont des niveaux de priorité hiérarchisés, IRQ7* ayant le niveau de priorité le plus élevé. La ligne IACK* suit toute la longueur du bus et est connecté à la broche IACKin* à l'emplacement A1. Lorsqu'elle est forcée à l'état bas, elle envoie un front descendant sur la ligne d'acquittement "INTERRUPT ACKNOWLEDGE DAISY CHAIN".

7.2. Lignes d'acquittement des interruptions IACK, IACKin*/IACKout*

Plusieurs générateurs pouvant émettre une interruption simultanément sur une même ligne IRQx, pour qu'un seul module soit acquitté il faut faire appel à une structure en "daisy chain". Cette ligne en "daisy chain" passe par chaque carte du VMEbus. Lorsqu'une interruption est acquittée par un signal IACK*, IACKin* est forcé au niveau bas à l'emplacement A1. Chaque module qui force une ligne de demande d'interruption au niveau bas doit attendre l'arrivée du niveau bas de IACKin* à son emplacement avant de prendre en compte l'acquittement. Le module qui prend en compte l'acquittement ne transmet pas le niveau bas en aval sur la "daisy chain" garantissant ainsi que seul un module soit acquitté.

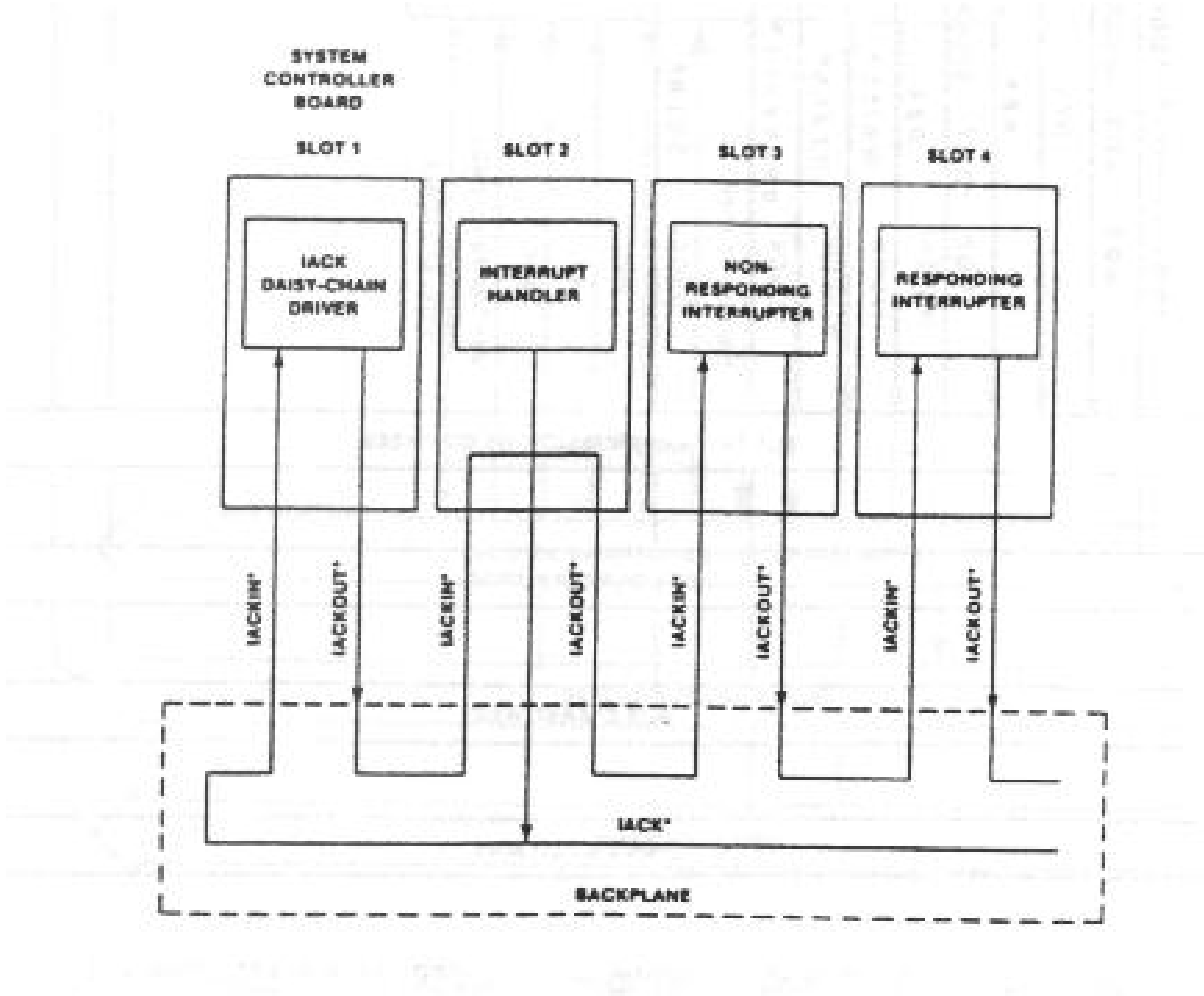


Figure 11 : Daisy IACKIN*/IACKOUT*

7.3. Contrôleur d'interruption

Le contrôleur d'interruption remplit plusieurs tâches:

- il accorde une priorité aux demandes d'interruption.
- il fait appel à son demandeur pour demander le bus et dès son obtention, il acquitte l'interruption.
- en fonction du mot d'état/identification ("status/ID") lu pendant l'acquittement, il lance la séquence de traitement de l'interruption.

Les contrôleurs d'interruption peuvent être identifiés dans un système par le nombre et par le rang des lignes de demandes d'interruption qu'ils desservent. La notation de l'option est IH (a-b). IH pour "Interrupt Handler"; a est la ligne du niveau le plus bas, et b celle du niveau le plus élevé. Une condition impérative du VMEbus est qu'un contrôleur d'interruption ne traite qu'une séquence contigu de

niveaux d'interruption.

La figure suivante représente un contrôleur d'interruption option IH (0-7) et les lignes de signaux qu'il utilise pour dialoguer avec les générateurs du VMEbus.

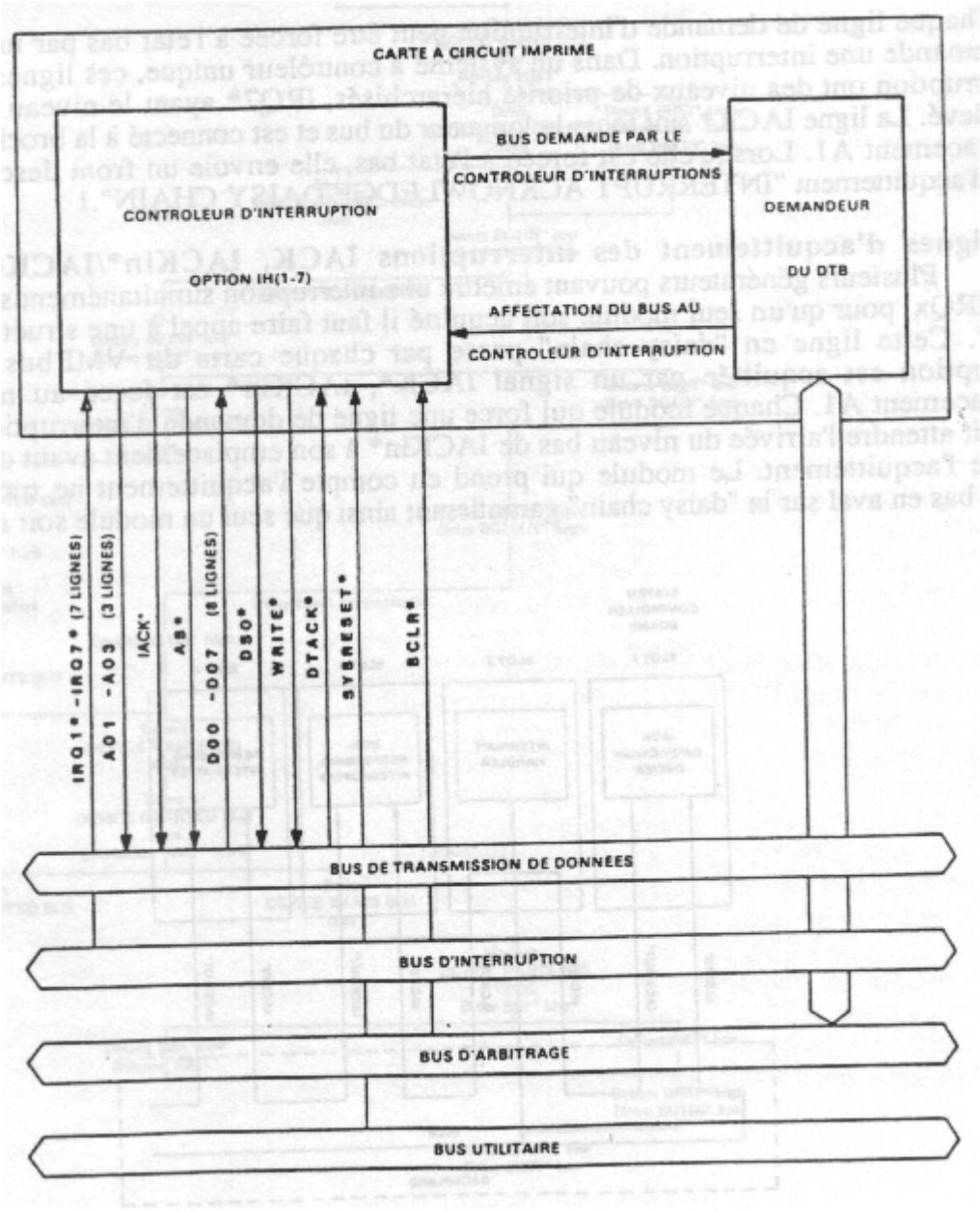


Figure 12 : Signaux utilisés par un contrôleur d'interruptions

Lorsqu'un contrôleur d'interruption reçoit une ou plusieurs demandes d'interruption des générateurs placés sur le bus, il ordonne au demandeur implanté sur sa carte de prendre le contrôle du bus de transmission de données. Il utilise ensuite le bus pour acquitter le générateur de plus haute priorité et lire l'"status/ID" envoyé par ce générateur.

7.4. Générateur d'interruption

Il remplit trois tâches:

- Il demande une interruption au contrôleur d'interruption qui surveille sa ligne de demande d'interruption IRQx* en activant celle-ci.
- Il fournit un "status/ID" (numéro de vecteur) au contrôleur d'interruption au moment de l'acquittement de sa demande d'interruption.
- Il doit transmettre un signal d'acquittement d'interruption sur la "daisy chain" s'il ne demande pas ce niveau d'interruption.

Si une carte interrompt sur plusieurs lignes elle doit posséder plusieurs générateurs, un générateur par ligne. Les générateurs sont identifiés par une option I (x) où x est le numéro de la ligne d'interruption utilisée.

La figure suivante représente un générateur option I (4) et les lignes de signaux qu'il utilise pour dialoguer avec son contrôleur d'interruption sur le VMEbus.

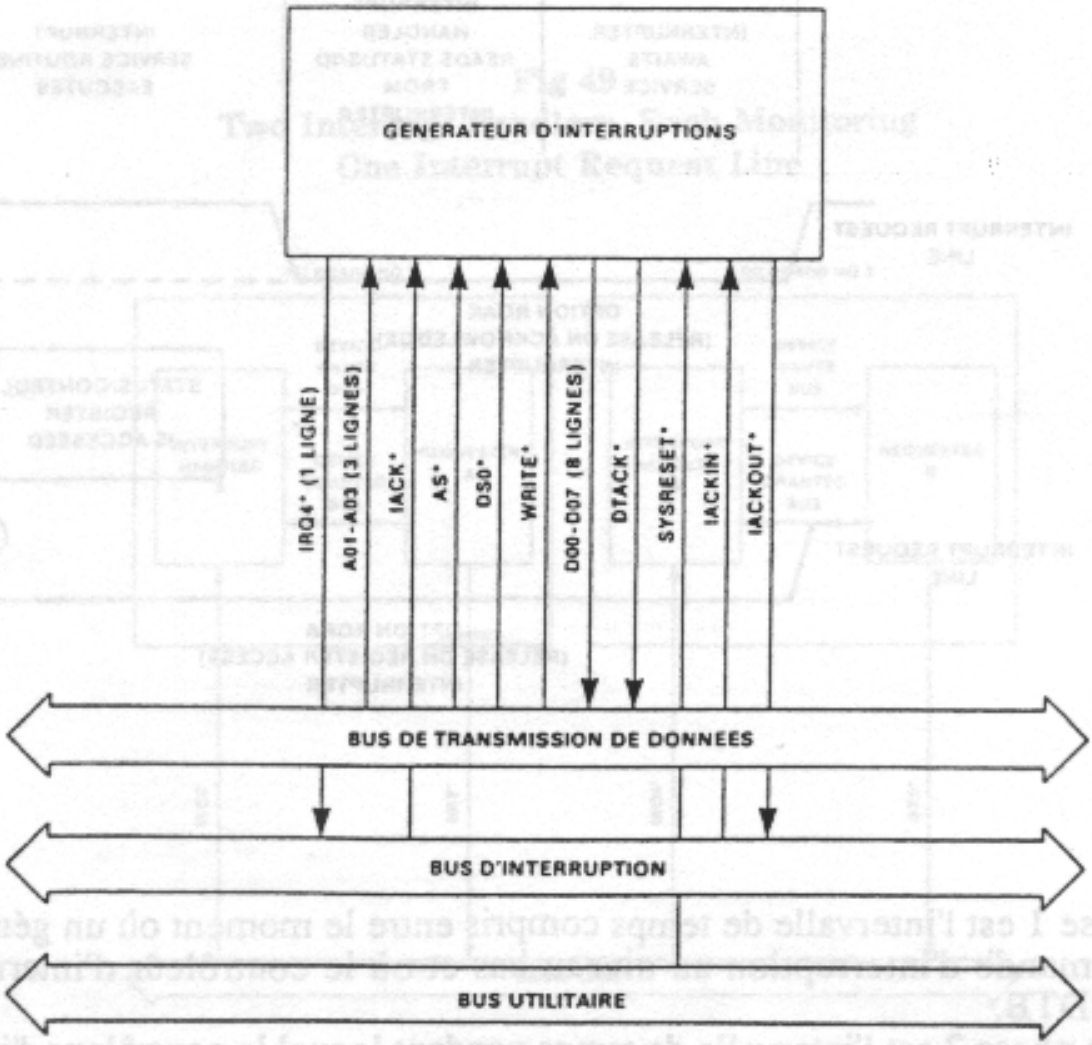


Figure 13 : Signaux utilisés par un générateur d'interruptions

Ce générateur utilise la ligne IRQ4*. il surveille le bus d'adresse du DTB, IACK* et la "daisy chain" IACKin*/IACKout* pour savoir quand son interruption est acquittée. Au moment de l'acquittement il place son "status/ID" (numéro de vecteur) sur les 8 lignes inférieures du bus de données et signale la validité de cette donnée au contrôleur d'interruption avec la ligne DTACK* (il faut qu'il ait le bus bien sûr).

Le contrôleur d'interruption indique sur les trois lignes d'adresses inférieures (A01 à A03) laquelle des 7 lignes de demande d'interruption est acquittée.

7.5. Principe de fonctionnement d'interruption sur le VMEbus

Une séquence d'interruption se divise en trois phases.

1 - La phase de demande d'interruption.
2 - La phase d'acquittement de l'interruption.
3 - La phase de traitement de l'interruption.

La figure suivante montre les relations entre les trois phases.

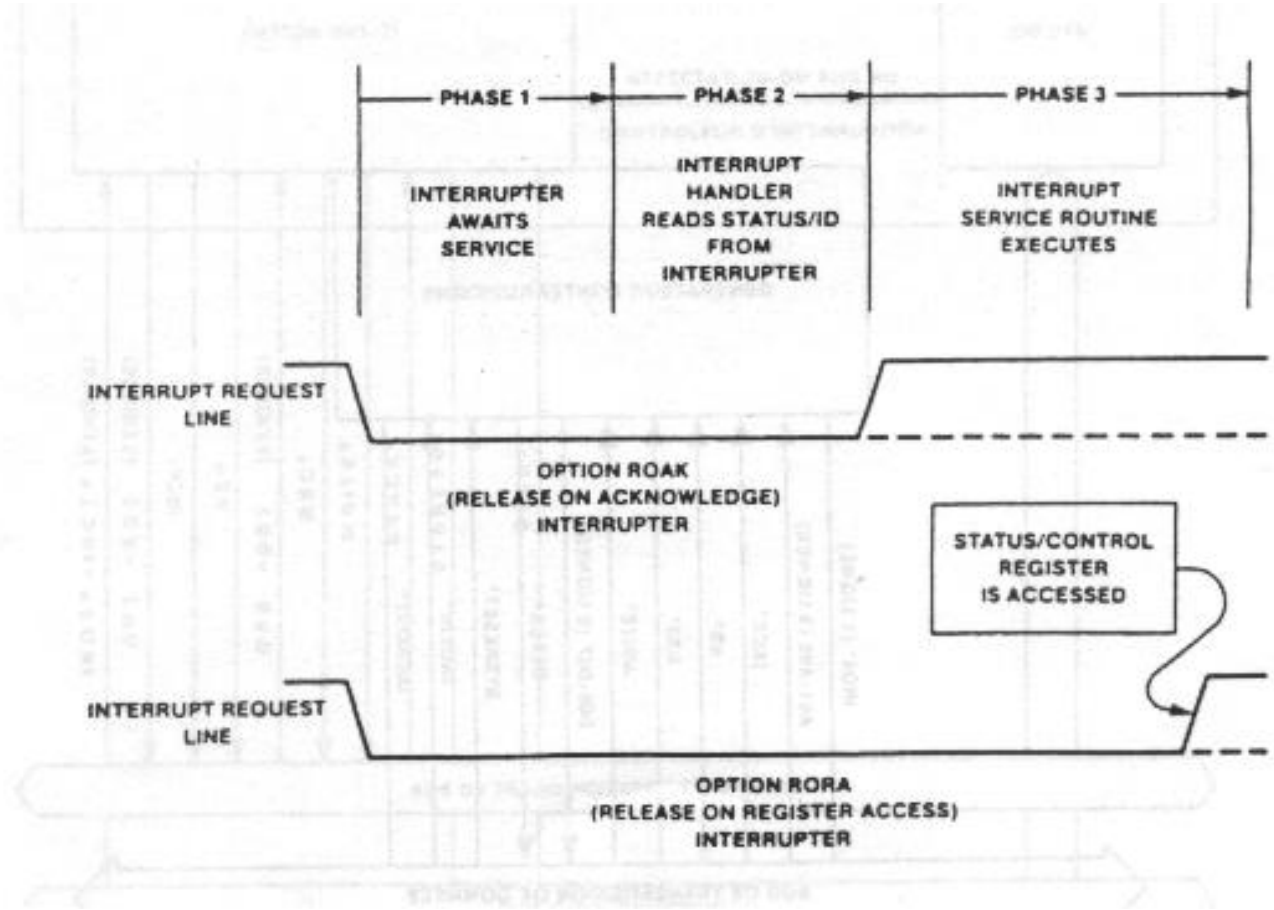


Figure 14 : Fonctionnement des interruptions

La phase 1 est l'intervalle de temps compris entre le moment où un générateur force une ligne de demande d'interruption au niveau bas et où le contrôleur d'interruption obtient le contrôle du DTB.

La phase 2 est l'intervalle de temps pendant lequel le contrôleur d'interruption utilise le DTB pour lire l'"status/ID" (numéro de vecteur) du demandeur.

Fonctionnement des systèmes à contrôle unique.

Dans les systèmes à contrôleur unique, les 7 lignes de demande d'interruption sont toutes surveillées par un seul contrôleur d'interruption. Dans ce cas, les lignes de demande d'interruption ont des priorités (IRQ7* correspondant à la priorité la plus élevée), et lorsque des demande simultanées sont détectées sur deux lignes de demande d'interruption, le "status/ID" de la demande la plus prioritaire est lu en premier.

Fonctionnement des systèmes d'interruption répartis.

Système d'interruption répartis ayant autant de contrôleurs qu'il y a de lignes d'interruption à surveiller.

Dans ce système à bus, chacune des lignes de demande d'interruption peut être surveillée par un contrôleur différent.

Chaque contrôleur d'interruption doit recevoir le contrôle du bus d'échange de données avant de pouvoir lire le "status/ID" transmis par un générateur qui active sa propre ligne de demande d'interruption. Lorsque 2 lignes de demande d'interruption sur le bus sont forcées simultanément à l'état bas, c'est l'arbitrage du bus qui donnera la priorité de lecture du "status/ID" au contrôleur ayant émis le plus fort BRx* ou pour des mêmes BRx* à celui le mieux placé dans la "daisy chain"

La figure suivante représente un système d'interruption réparti où le contrôleur A surveille IRQ2* et a un demandeur sur sa carte qui demande le DTB sur BR2*.

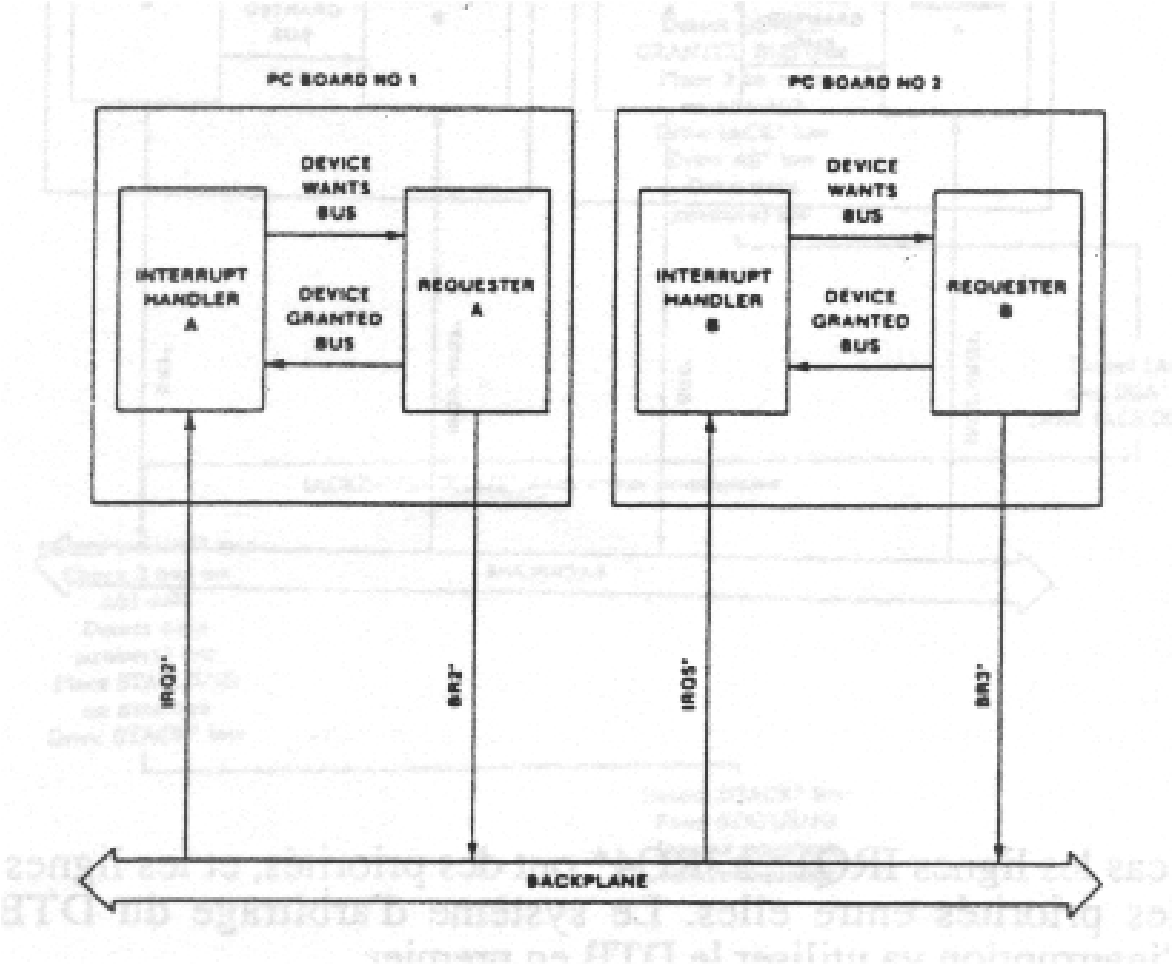


Figure 15 : Système d'interruptions réparti

Le contrôleur d'interruption B surveille IRQ5* et a un demandeur sur sa carte qui demande le DTB par BR3*.

Si 2 générateurs sur le bus d'interruption forcent simultanément IRQ2* et IRQ5* au niveau bas, les 2 contrôleurs d'interruption peuvent ordonner à leur demandeur de forcer BR2* et BR3* simultanément au niveau bas (Ceci n'est pas certain, puisque chaque contrôleur d'interruption peut attendre très longtemps avant de tenter de prendre en compte l'interruption). Si un arbitrage basé sur les priorités est utilisé et si les 2 demandes du bus sont au niveau bas au moment de l'arbitrage, l'arbitre accorde d'abord le contrôle du DTB au demandeur du contrôleur d'interruption B, et le contrôleur d'interruption A doit attendre que B ait fini d'utiliser le DTB. Si un arbitrage de type circulaire ("round robin") est utilisé, n'importe lequel des 2 contrôleurs peut recevoir le contrôle du bus en premier.

Systèmes d'interruption répartis ayant 2 à 6 contrôleurs d'interruption.

Dans ce système 2 lignes de demande d'interruption ou plus sont surveillées par un seul contrôleur d'interruption . La figure suivante représente un

système configuré avec deux contrôleurs d'interruption où le contrôleur A surveille IRQ1* à IRQ4* et le contrôleur B surveille IRQ5* et IRQ7*.

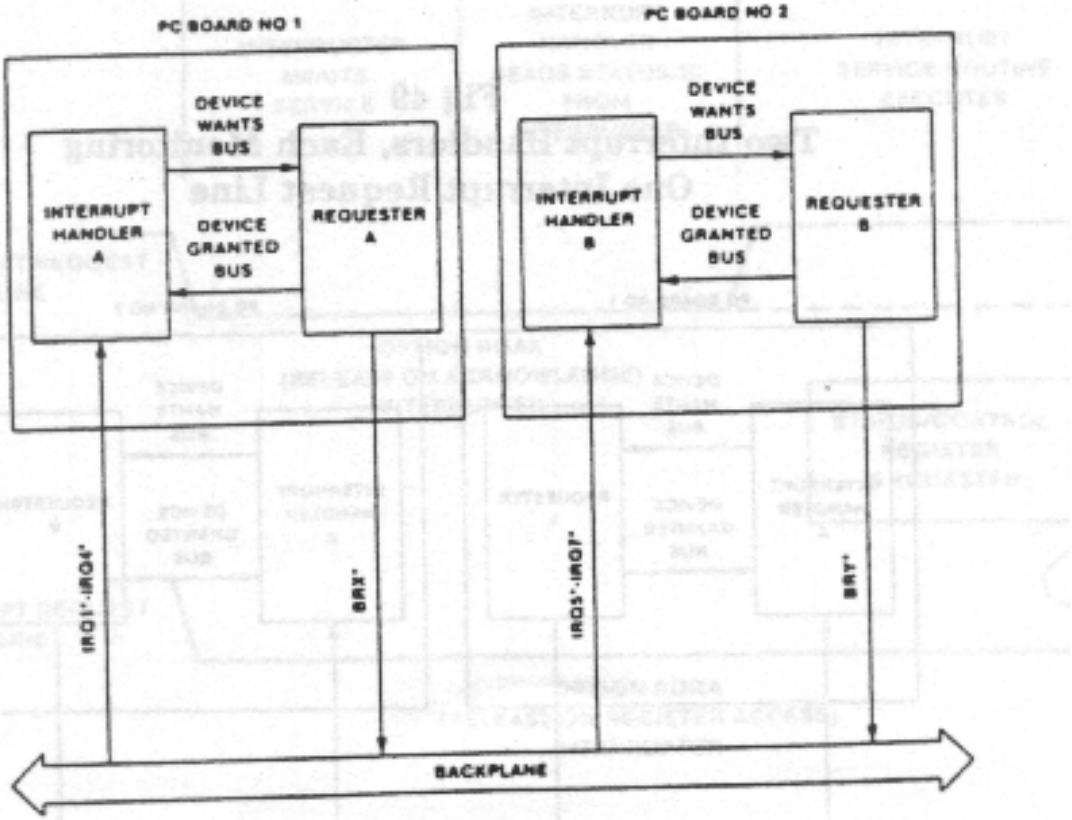


Figure 16 : Système d'interruptions à 2 contrôleurs

Dans ce cas les lignes IRQ1* à IRQ4* ont des priorités, et les lignes IRQ5* à IRQ7* ont également des priorités entre elles. Le système d'arbitrage du DTB décide lequel des contrôleurs d'interruption va utiliser le DTB en premier.

Principe de fonctionnement des systèmes d'interruption à contrôleur unique.

La figure suivante nous montre le fonctionnement d'un tel système dont le contrôleur surveille les 7 lignes d'interruption.

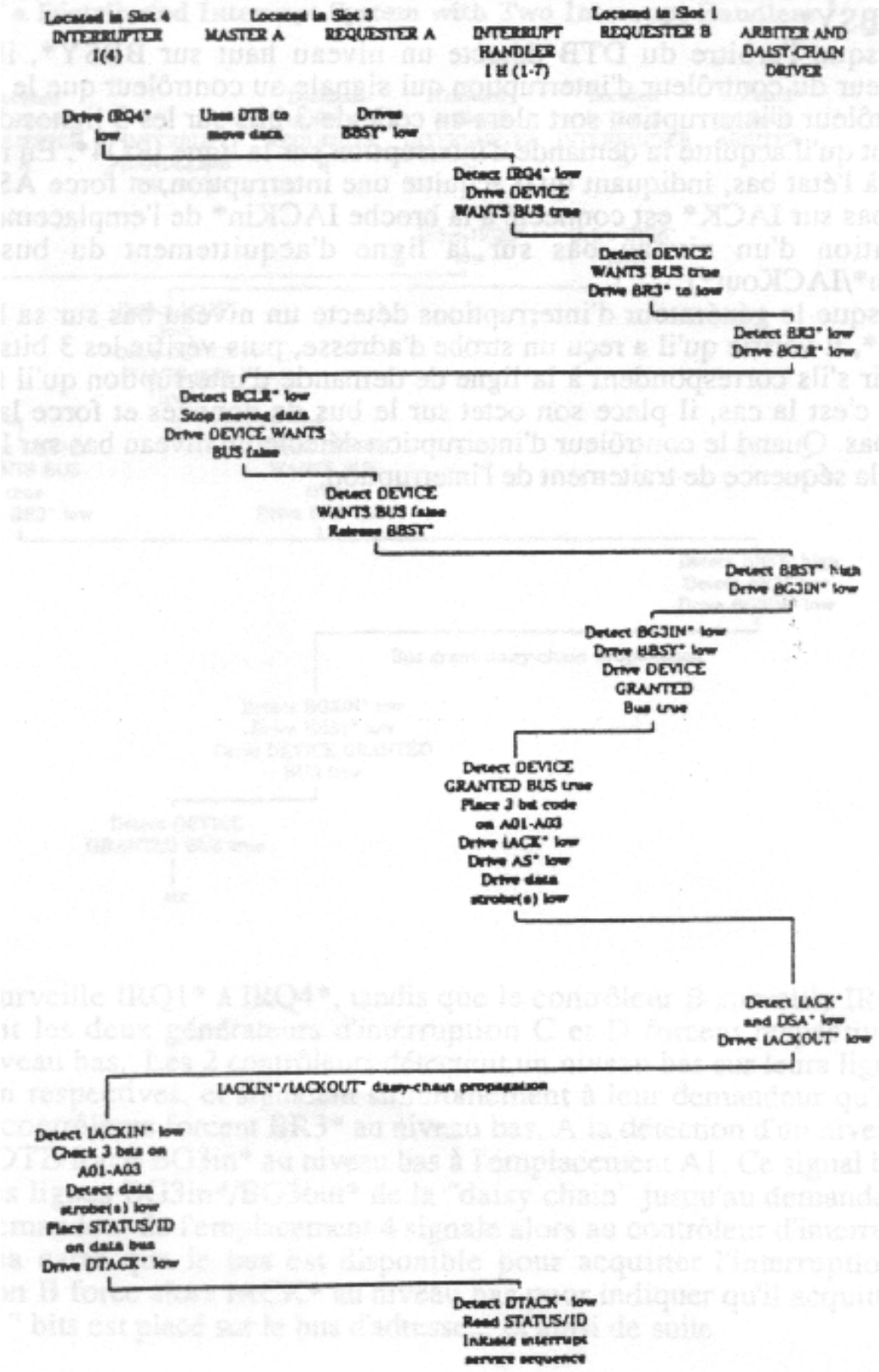


Figure 17 : Système à un contrôleur d'interruptions

Au moment où un générateur demande une interruption en forçant IRQ4* au niveau bas, un maître du DTB utilise le DTB pour transférer des données à l'intérieur du système au niveau de demande 2. Lorsque le contrôleur d'interruption détecte la présence d'un niveau bas sur IRQ4*, il envoie un signal à son demandeur, lui indiquant qu'il a besoin du bus. Ce demandeur force alors BR3* au niveau bas. A la détection de la demande du bus, l'arbitre du type PRI force BCLR* au niveau bas, indiquant qu'un demandeur plus prioritaire attend le DTB . Lorsque le maître du DTB détecte un niveau bas sur BLCR*, il arrête de transmettre des données et permet à son demandeur d'abandonner le contrôle du DTB: le demandeur pour cela libère BBSY*.

Lorsque l'arbitre du DTB détecte un niveau haut sur BBSY*, il donne le DTB au demandeur du contrôleur d'interruption qui signale au contrôleur que le DTB est disponible. Le contrôleur d'interruption sort alors un code de 3 bits sur les 3 lignes d'adresse inférieures, indiquant qu'il acquitte la demande d'interruption sur la ligne IRQ4*. En même temps, il force IACK* à l'état bas, indiquant qu'il acquitte une interruption, et force AS* au niveau bas. Le niveau bas sur IACK* est connecté à la broche IACKin* de l'emplacement A1 et entraîne la propagation d'un niveau bas sur la ligne d'acquittement du bus en "daisy chain" (IACKin*/IACKout*).

Lorsque le générateur d'interruptions détecte un niveau bas sur sa ligne d'acquittement IACKin*, il vérifie qu'il a reçu un strobe d'adresse, puis vérifie les 3 bits d'adresse inférieurs pour voir s'ils correspondent à la ligne de demande d'interruption qu'il force au niveau bas. Comme c'est la cas, il place son "status/ID" sur le bus de données et force la ligne DTACK* au niveau bas. Quand le contrôleur d'interruption détecte un niveau bas sur DTACK* il lit l'"status/ID" et lance la séquence de traitement de l'interruption.

Principe de fonctionnement des systèmes d'interruption répartis.

La figure suivante représente le fonctionnement d'un tel système réparti ayant deux contrôleur d'interruptions.

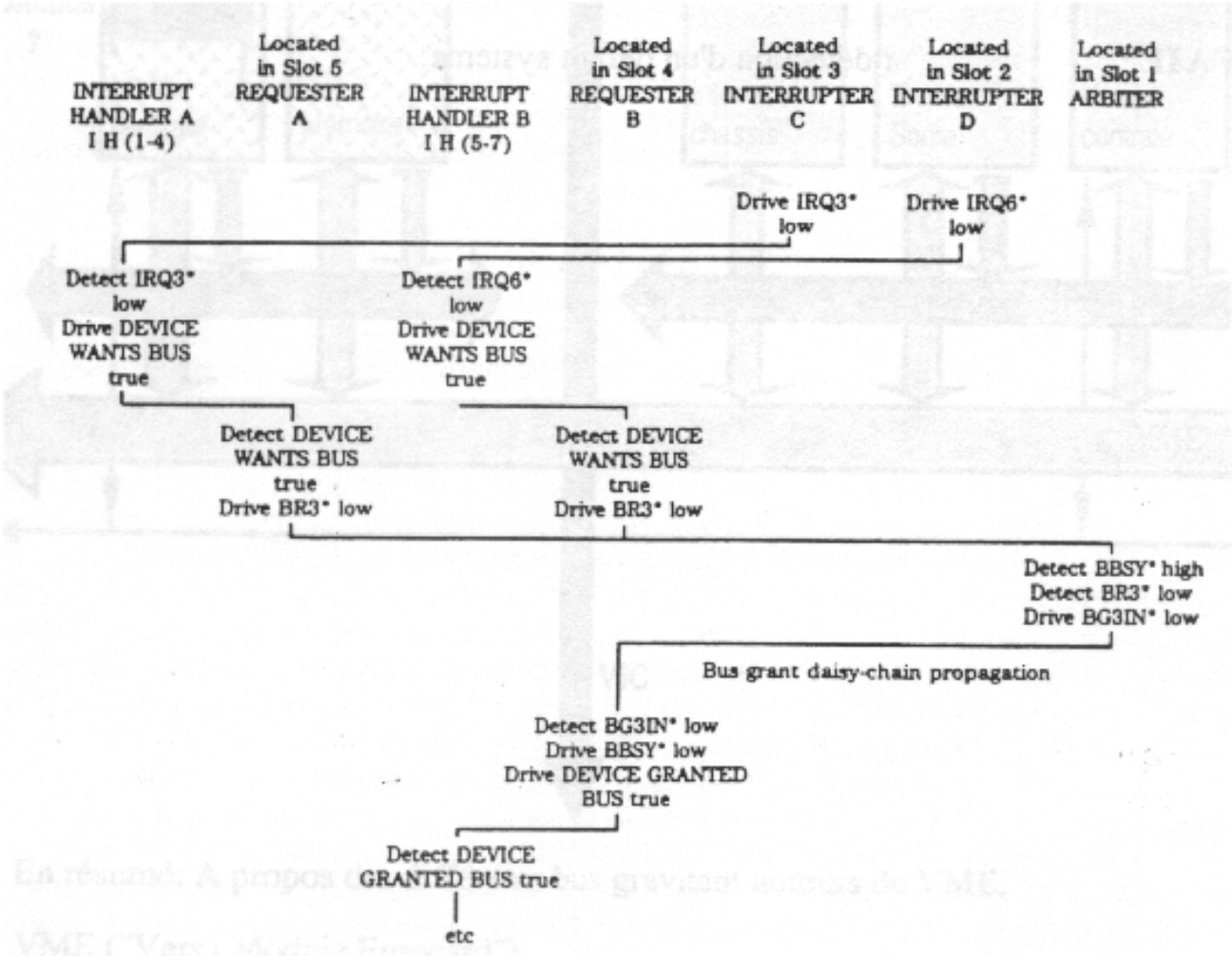


Figure 18 : Système à deux contrôleurs d'interruptions

Le contrôleur A surveille IRQ1* à IRQ4*, tandis que le contrôleur B surveille IRQ5* à IRQ7*. Simultanément les deux générateurs d'interruption C et D forcent respectivement IRQ3* et IRQ6* au niveau bas. Les 2 contrôleurs détectent un niveau bas sur leurs lignes de demande d'interruption respectives, et signalent simultanément à leur demandeur qu'ils ont besoin du DTB. Les 2 contrôleurs forcent BR3* au niveau bas. A la détection d'un niveau bas sur BR3*, l'arbitre du DTB force BG3in* au niveau bas à l'emplacement A1. Ce signal bas est transmis en aval sur les lignes BG3in*/BG3out* de la "daisy chain" jusqu'au demandeur de l'emplacement 4. Le demandeur de l'emplacement 4 signale alors au contrôleur d'interruption B qui se trouve sur sa carte que le bus est disponible pour acquitter l'interruption. Le contrôleur d'interruption B force alors IACK* au niveau bas pour indiquer qu'il acquitte une interruption et un code 3 bits est placé sur le bus d'adresse... et ainsi de suite.

8. UTILITAIRES

SYSCLK : horloge système à 16 MHz.

C'est une horloge système indépendante tournant en permanence à une fréquence de 16 Méga Hertz avec un cycle utile de 50%. Elle est fournie par le contrôleur système logé à l'emplacement 1 (le premier en partant de la gauche). C'est une base de temps fixe fort utile.

ACFAIL*: défaut d'alimentation en courant alternatif.

SYSRESET* **:** remise à zéro du système.

SYSFAIL* **:** détection d'un défaut système.

9. LE MONDE VME

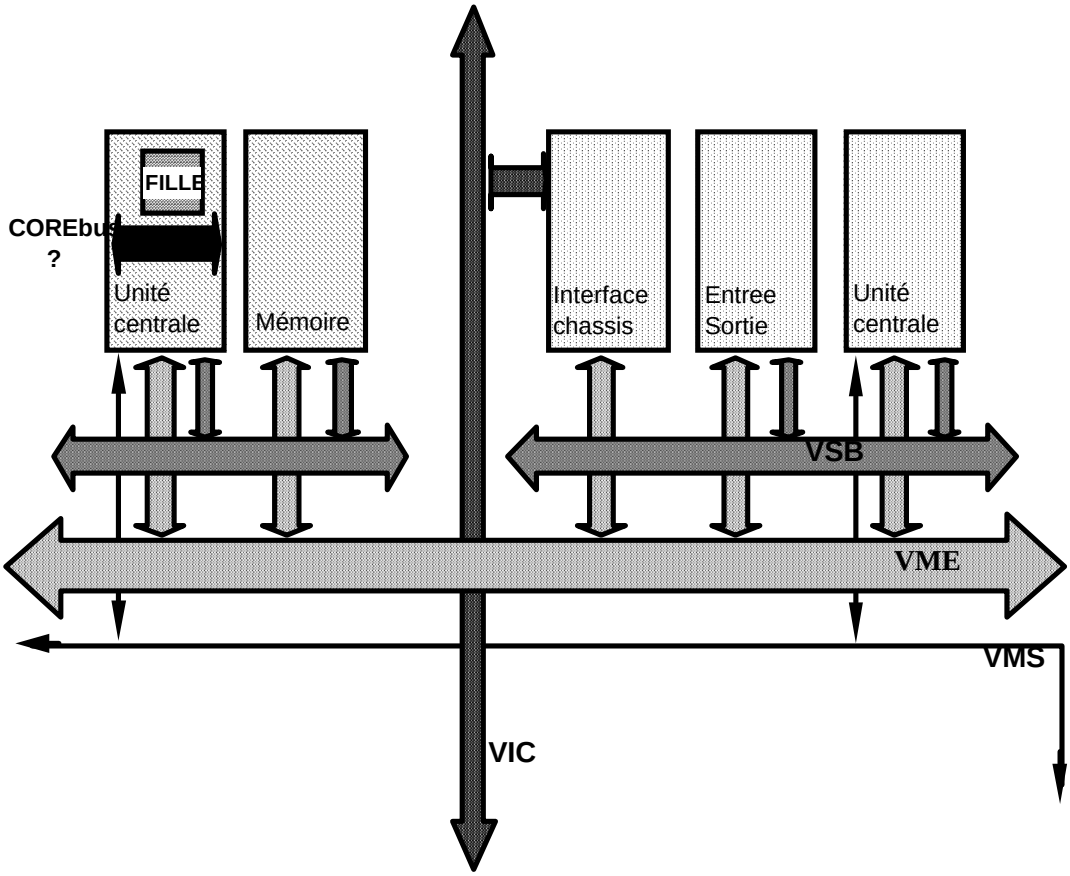


Figure 19 : Le monde VME

En résumé: A propos des différents bus gravitant autours de VME.

VME ("Versa Module Eurocard").

- asynchrone.
- non-multiplexé.
- capacité d'adressage maximale 4 Giga octets. (plus les modificateurs d'adresses).
- 8, 16 ou 32 bits de données (64 bits aujourd'hui en mode multiplexé révision D).
- 4 niveaux d'accès au bus.
- 7 niveaux d'interruption.
- vitesse de transfert maximale 40 Méga octets par seconde.
- nombre de slots indéfini (en pratique: 6, 9 ou 21 slots).

VSB ("Vme Subsystem Bus")

- asynchrone.
- multiplexé.
- capacité d'adressage 4 Giga octets.
- 32 bits de données.
- au maximum 6 cartes par bus VSB.
- 1 ou 2 maîtres par bus.
- vitesse de transfert théorique maximale 80 Méga octets par seconde.

VMS ("VMe Serial")

- protocole de transmission défini: technique de passage de jetons (jusqu'à 1024).
- vitesse de transfert maximale 3.2 Méga bits par seconde.

VIC ("Vme Inter-Crate bus")

Le VICbus ("Vme Inter-Crate bus") est un bus de 32 bits multiplexé. Ce bus multi-maître est un câble qui permet de relier entre eux plusieurs châssis VME. Il utilise une transmission différentielle sur ses 69 lignes (au total 138 fils). Nous pouvons ainsi relier jusqu'à 31 châssis sur un câble qui ne peut excéder 100 mètres. Les caractéristiques du VICbus permettent un fonctionnement transparent entre châssis, ainsi un transfert de données s'effectue de la même manière à l'intérieur du châssis qu'à l'extérieur. Le VICbus possède un seul niveau d'arbitration et 32 signaux d'interruption en mode multiplexé. Les spécifications de ce bus définies par le groupe de travail VMEbus de l'ESONE ("European Standards On Nuclear Electronics committee of European Laboratories") ont été adoptées par l'IEC pour une standardisation future (groupe de travail commun ISO/IEC (International Organization for Standards").

VXI : VME étendu à l’Instrumentation.

10. LE FUTUR DU BUS VME : VME 64 BITS REVISION D

L'arrivée entre autres des microprocesseurs RISC ("Reduced Instruction Set Computer") avec des vitesses d'exécution de plus en plus grande et l'usage intensif des mémoires cache entraînent l'évolution du bus VME et amènent de nouveaux bus plus performants encore. Ces nouvelles technologies sont à l'origine de la révision D du VMEbus et de NGA ("Next Generation Architecture") aujourd'hui appelé VFEA ("VME Futurebus+ Extended Architecture").

Le VMEbus utilise des lignes séparées pour les adresses et les données. Si nous utilisons ces lignes ensembles alors il est possible de les faire travailler de façon multiplexé et d'obtenir ainsi un bus 64 bits pour les adresses et pour les données.

L'adressage 64 bits:

L'adressage 64 bits est obtenu en utilisant un nouveau code de modification d'adresse qui indique qu'il s'agit d'une adresse 64 bits. Ainsi huit nouveaux codes de modification d'adresse ont été ajoutés. De même, les spécifications des protocoles et chronogrammes pour utiliser les lignes de données pour transporter une adresse ont été définis.

Le transfert de données sur 64 bits: option MBLT

Le transfert de données sur 64 bits est obtenu en utilisant quatre nouveaux codes de modification d'adresse qui indiquent qu'il s'agit d'un transfert de données sur 64 bits. Ce code est présenté en même temps que l'adresse et doit rester présent pendant tout le transfert du bloc de données. Ainsi quatre nouveaux codes de modification d'adresse ont été ajoutés. De même, les spécifications des protocoles et chronogrammes pour utiliser les lignes d'adresses pour transporter une données ont été définis. La vitesse théorique maximale de transfert passe ainsi à 80 Méga octets par seconde.

Les cycles de réessai en transfert de données:

A ces deux ajouts la révision D définit également les cycles de réessais en transfert de données ("Retrying Data Transfer Cycles"). La ligne VME jusqu'alors réservée est maintenant utilisée pour mettre en place deux mécanismes différents: - les réessais en transfert de données et les cycles de transfert de données en diffusion entièrement inter verrouillée ("Data broadcast fully interlocked").

Pour comprendre ce que sont ces cycles, il faut savoir ce qu'est la diffusion de données ("Data Broadcast"). La diffusion de données est l'écriture par un maître d'une donnée dans plusieurs esclaves simultanément. Comme le maître doit obtenir la réponse de tous les esclaves, il attend qu'une ligne en collecteur ouvert soit relâchée par tous les esclaves ("fully interlocked"). Tant que cette ligne (RETRY*) est active, le maître doit maintenir des données stabilisées sur le bus.

D'autre part, le réessai est une fonction qui permet d'éviter une étreinte mortelle ("dead lock") qui peut survenir. Cette étreinte mortelle intervient lorsqu'un processeur A essaie d'accéder à une donnée qui se trouve dans la mémoire locale d'un processeur B en même temps que le processeur B essaie d'accéder à une donnée qui réside dans la mémoire locale du processeur A. Du fait que les données que les deux processeurs tentent d'accéder réside dans la mémoire locale de l'autre, les deux doivent acquérir **à la fois** le contrôle de leur propre bus local et du VMEbus. Les

deux processeurs obtiendrons sans problème le contrôle de leur propre bus local sans se préoccuper l'un de l'autre, mais un seul des deux obtiendra le contrôle du bus VME. Et pour terminer son cycle VME le processeur doit obtenir l'accès de la mémoire locale de l'autre processeur qui est contrôlé par cet autre processeur. C'est fait le système est définitivement bloqué. Le cycle peut néanmoins être terminé par l'envoi d'un signal BERR* par le "bus timer". Maintenant la question est de savoir pourquoi il y a eu une erreur de bus (erreur d'adressage, erreur de parité mémoire, défaut de page mémoire ....). Pour résoudre ce problème la ligne réservée du bus VME est devenue le signal RETRY qui veut dire ce n'est pas une véritable erreur de bus alors réessayer.

Dernières propositions pour la révision D :

1) option SSBLT ("Source Synchronized Block Transfers")
2) définition de régions d'adressage pour des registres
   d'autoconfiguration: CSR (Control/Status Register")
3) annulation d'un signal DTACK*
4) sémaphores matériels
5) verrouillage de cycle VME

Arrêtons nous un instant sur un des aspects les plus intéressant de ces nouvelles propositions: l'option SSBLT. Tout comme cela est fait sur les bus EISA ou microchannel, l'idée est d'augmenter la bande passante du bus. Cette nouvelle option offre une extension du mode MBLT dont le débit passe de 80 Mégaoctets/s à 160 Mégaoctets/s. Dans le mode SSBLT l'accord accepté ("handshake") entre DS* et DTACK* est éliminé. Le signal DS0* devient le signal d'horloge pour les écritures et le signal DTACK* pour les lectures.

En plus de cela les deux fronts d'horloge sont utilisés pour échantillonner les données. Evidemment cette nouvelle proposition doit entraîner la définition de nouveaux codes de modification d'adresse.

11. REFERENCES

http://www.vita.com/
http://www.eg3.com/indc/indcxvme.htm
http://www.vmebus-systems.com/