MASTER ACSI

"Architecture Matérielle et Logicielle des Systèmes Intégrés"
Partie “Matériel”

Janvier 2007

On se propose d'étudier l'architecture interne d'un contrôleur d’interruptions (appelé dans la suite ICU, comme Interrupt Contrôleur Unit) respectant l’interface VCI.

La fonction principale d’un contrôleur d’interruptions est de concentrer 32 lignes d’interruptions provenant de N périphériques indépendants vers un seul signal d’interruption, qui est directement connecté au processeur. Il suffit qu’une seule ligne d’interruption entrante soit active (à l’état haut) pour que la ligne d’interruption sortante vers le processeur devienne active.

Une seconde fonctionnalité du contrôleur d’interruption est l’encodage des priorités entre les lignes d’interruption entrantes : les priorités sont fixes, et les lignes d’interruption connectées aux ports d’index le plus faible sont les plus prioritaires :

IRQIN(0) > IRQIN(1) > IRQIN(2) > ... IRQIN(31)

Lorsque processeur reçoit une requête d’interruption, le logiciel « gestionnaire d’interruption » interroge le composant ICU pour obtenir l’index de la ligne d’interruption active la plus prioritaire (les périphériques étant indépendants les uns des autres, il peut y avoir plusieurs requêtes actives simultanément). Le composant ICU renvoie alors l’index de la requête d’interruption ayant l’index le plus petit parmi toutes les requêtes actives au moment de cette interrogation. Cet index permet au gestionnaire d’interruption d’adresser un tableau (stocké en mémoire), contenant les adresses des différentes routines correspondant aux différentes lignes d’interruption. Ce tableau est appelé « vecteur d’interruption ».

Une troisième fonctionnalité du composant ICU est de permettre au logiciel de masquer individuellement chacune des 32 lignes d’interruption entrantes, au moyen d’un mot de 32 bits qui doit être chargé dans le composant ICU par le système d’exploitation.

Le composant ICU se comporte donc comme une cible VCI, et possède 1 registre de masque 32 bits, adressable en écriture pour définir le masquage, et un registre d’index, adressable en lecture, pour obtention de l’index de la ligne d’interruption active la plus prioritaire. Ces registres sont accédés en décodant le bit A2 de l’adresse VCI de la façon suivante :

A2 = 0 : Ecriture dans le registre de masque des interruptions
A2 = 1 : Lecture dans le registre d’index

L’automate contrôlant le composant ICU doit gérer deux types d’erreurs :

- Un paquet VCI de longueur supérieure à 1 déclenche une réponse erreur
- Une violation du mode d’accès (commande de lecture du registre de masque par exemple), doit déclencher une réponse erreur.

Q1) (1 point) Pourquoi les 32 adresses constituant le « vecteur d'interruption » ne sont-elles pas stockées directement dans le composant ICU ? L'avantage serait évidemment de permettre au composant ICU de renvoyer directement l'adresse de la routine à laquelle il faut se brancher.

Q2) (1 point) Le contrôleur d'interruption possède 32 lignes d'interruption entrantes (indexées de 0 à 31), mais la valeur de l'index renvoyée par le composant peut être comprise entre 0 et 32 inclus. Quelle est l'utilité de la valeur 32 ?

Q3) (1 point) Le composant ICU transmet les requêtes d'interruption des périphériques vers le processeur. En revanche, les signaux d'acquittement, permettant au gestionnaire d'interruption de désarmer les requêtes d'interruption, ne transitent pas par le composant ICU. Expliquer pourquoi.

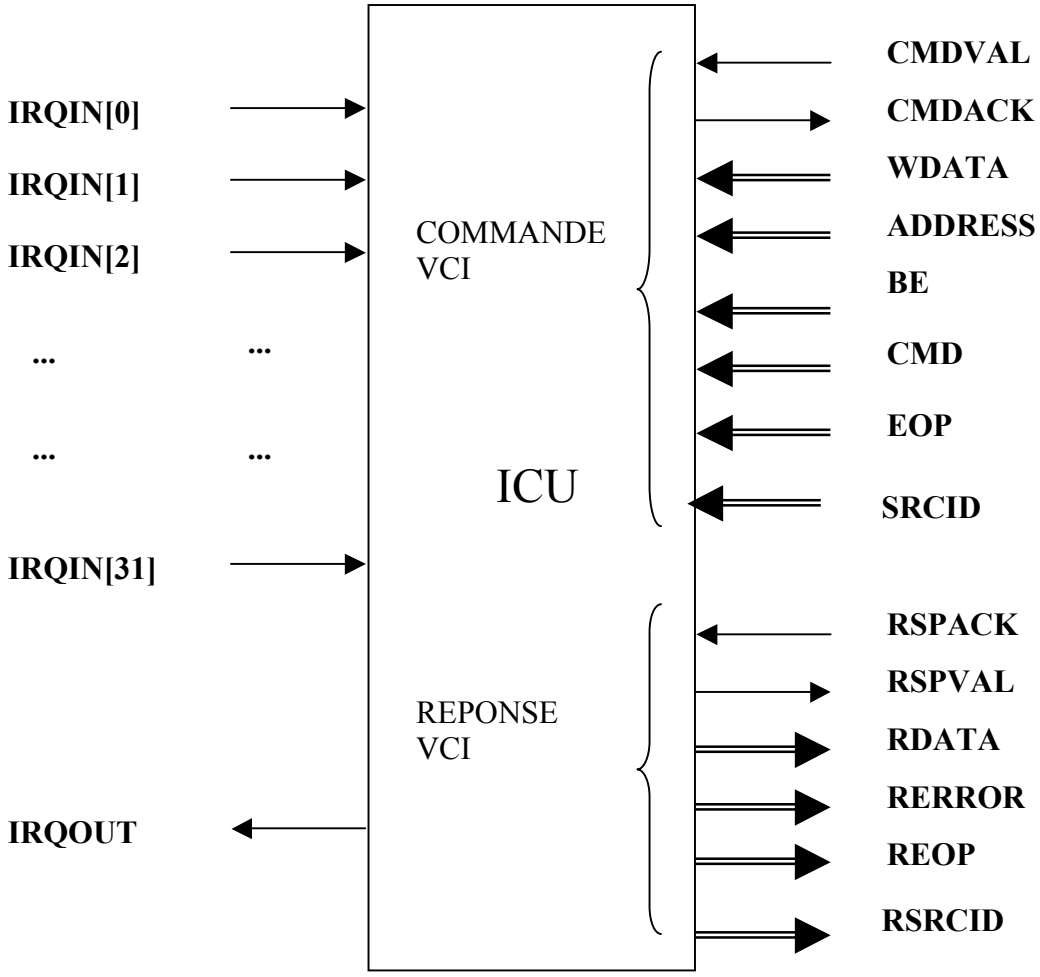
Q4) (1 point) Quelle est la taille minimale (en octets) du segment mémoire associé au contrôleur ICU. Expliquer pourquoi ce segment doit être aligné.

Q5) (1 point) Proposez un schéma de principe pour l'architecture interne du composant ICU, qui comporte quelques opérateurs combinatoires, un registre de masque, un registre d'index, et un unique automate de contrôle.

Q6) (2 points) Représenter graphiquement l'automate de Moore qui contrôle le composant ICU, en attachant à chaque transition de cet automate la condition de franchissement. Cet automate comporte 5 états : IDLE, W_MASK, R_INDEX, ERROR, et ERROR_LAST. Les signaux d'entrée de cet automate sont les signaux VCI CMDVAL, RSPACK, A2 (bits de poids 4 de l'adresse VCI), WRITE (signal résultant du décodage de la commande VCI et exprimant qu'il s'agit d'une écriture, EOP (signal VCI exprimant qu'il s'agit d'une fin de paquet).

Q7) (2 points) Remplir le tableau donnant les valeurs des signaux de sortie CMDACK, RSPVAL, RDATA, RERROR, REOP suivant l'état de l'automate.

Q8) (1 point) Comment peut-on traiter le cas d'un système comportant plus de 32 lignes d'interruption, sans modifier l'architecture interne du composant ICU, et en supposant que le processeur ne possède qu'un seul port d'interruption ?



Un mot d'un paquet commande VCI comporte 6 champs (en plus des 2 signaux de contrôle de flux CMDVAL et CMDACK) :

- ADDRESS (32 bits)
- WDATA (32 bits) : donnée à écrire, en cas d'écriture
- BE (4 bits) : indique quels octets doivent être écrits, en cas d'écriture
- CMD (2 bits) : définit le type de la requête : READ/WRITE/LINKED_READ/NOP
- EOP (1 bit) : marqueur de fin de paquet requête VCI.
- SCRID (8 bits) : numéro de l'initiateur source de la requête

Un mot d'un paquet réponse comporte 4 champs (en plus des 2 signaux de contrôle de flux RSPVAL et RSPACK) :

- RDATA (32 bits) : donnée demandée, en cas de lecture
- RERROR (1 bit) : signal d'erreur
- REOP (1 bit) : marqueur de fin de paquet réponse VCI.
- RSCRID (8 bits) : numéro de l'initiateur source de la requête