Architecture des Ordinateurs
Partie II : Microprocesseur

3. Interruptions et DMA (suite)

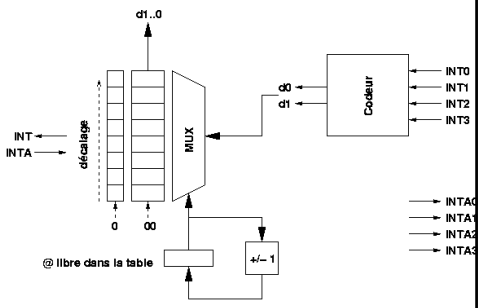
David Simplot
simplot@fil.univ-lille1.fr

Au sommaire...

Objectifs

- Comment implémenter un gestionnaire d'interruption ?
- Transition vers le chapitre 4
  - Microprogrammation
- Unité de contrôle plus simple que celle d'un microprocesseur :
  - Automates,
  - Premier pas vers la microprogrammation…

Exemple d'implémentation (1/15)

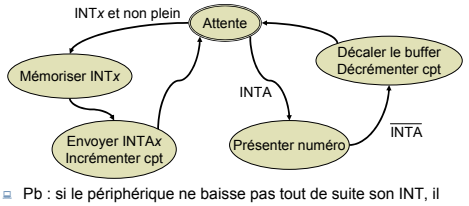


Exemple d'implémentation (2/15)
(1) déterminer le comportement

- Que doit faire le gestionnaire d'interruption ?
  - Sur réception d'un signal INT*x* : Mémoriser dans le buffer
    - Y a-t-il une place libre ?
      - Si oui, mémoriser puis incrémenter le compteur il faut également envoyer le signal INTA*x*
      - Si non, rien à faire… attendre qu'une place se libère…
  - Sur réception d'un signal INTA : Présenter le numéro d'interruption
    - Attendre que INTA repasse à zéro
    - Décaler les registres du buffer et décrémenter le compteur

Exemple d'implémentation (3/15)
(1) déterminer le comportement (suite)

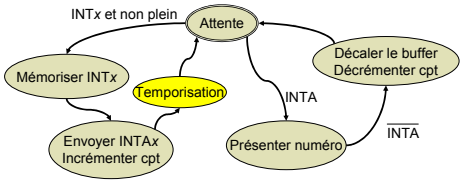
- On utilise un automate :



- Pb : si le périphérique ne baisse pas tout de suite son INT, il sera mémorisé plusieurs fois…
  - On ajoute un état de temporisation…

Exemple d'implémentation (4/15)
(1) déterminer le comportement (suite)

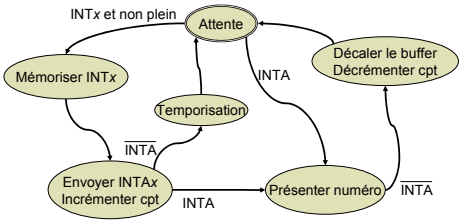
- Nouvel automate…



- Est-il utile de perdre un cycle d'horloge à chaque mémorisation?
  - Si il y a un INTA, on peut passer directement à son traitement…

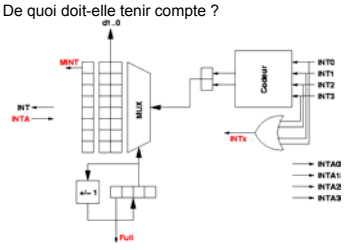
Exemple d'implémentation (5/15)
(1) déterminer le comportement (suite)

- On court-circuite la temporisation :



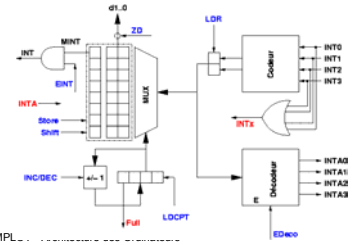
Exemple d'implémentation (6/15)
(2) déterminer les entrées/sorties

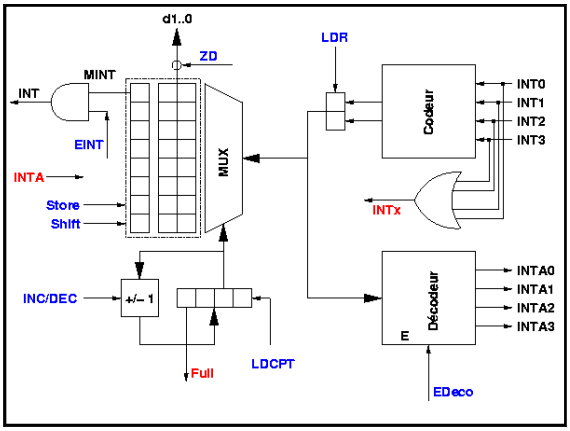
- Déterminer les entrées de l'unité de contrôle
  - De quoi doit-elle tenir compte ?



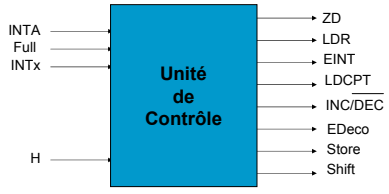
Exemple d'implémentation (7/15)
(2) déterminer les entrées/sorties (suite)

- Déterminer les signaux de sortie de l'UC
  - Quels signaux doit-elle générer ?



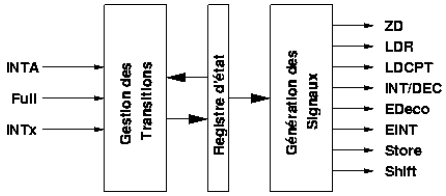


Exemple d'implémentation (8/15)
(2) déterminer les entrées/sorties



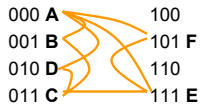
Exemple d'implémentation (9/15)
(3) choix de la technique d'implémentation

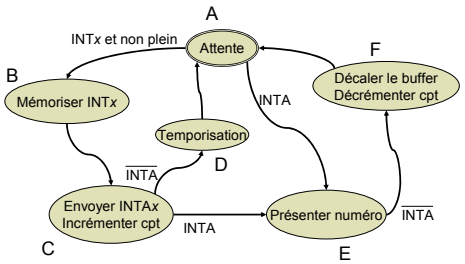
- Implémentation « câblée » :



Exemple d'implémentation (10/15)
(4) Codage des états

- Dans notre automate, on a 6 états
  - ⇒ au minimum 3 bits dans le registre d'états
  - Il faut attribuer une configuration de ces 3 bits pour chacun des états
    - Critère : minimiser la distance (en nombre de bits différents) entre deux états reliés entre eux…

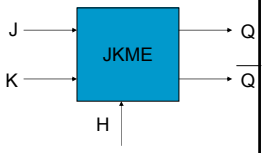




Exemple d'implémentation (11/15)
(5) Gestion des transitions

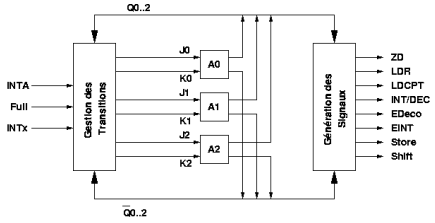
- Pour stocker le registre d'états, on utilise des bascules JKME
  - Vues en TD
- Pour mémoire :

J	K	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	$\overline{Qn}$



Mémorisation sur un front montant (ou descendant au choix)

Exemple d'implémentation (12/15)
(5) Gestion des transitions (suite)



- On doit donc juste générer les J0..2 et K0..2 en fonction de Q0..2 et des entrées…

Exemple d'implémentation (13/15)
(5) Gestion des transitions (suite)

Etat	Q2..0	INTA	Full	INTx	état suivant
A	000	*	0	1	B 001
		1	1	*	E 111
		1	0	0	E 111
		0	1	*	A 000
		0	0	0	A 000
B	001	*	*	*	C 011
C	011	0	*	*	D 010
		1	*	*	E 111
D	010	*	*	*	A 000
E	111	0	*	*	F 101
		1	*	*	E 111
F	101	*	*	*	A 000

Exemple d'implémentation (14/15)
(5) Gestion des transitions (suite)

Etat	Q2..0	INTA	Full	INTx	état suivant	JK2	JK1	JK0
A	000	*	0	1	B 001	0*	0*	1*
		1	1	*	E 111	1*	1*	1*
		1	0	0	E 111	1*	1*	1*
		0	1	*	A 000	0*	0*	0*
		0	0	0	A 000	0*	0*	0*
B	001	*	*	*	C 011	0*	1*	*0
C	011	0	*	*	D 010	0*	*0	*1
		1	*	*	E 111	1*	*0	*0
D	010	*	*	*	A 000	0*	*1	0*
E	111	0	*	*	F 101	*0	*1	*0
		1	*	*	E 111	*0	*0	*0
F	101	*	*	*	A 000	*1	0*	*1

Exemple d'implémentation (15/15)
(6) Génération des signaux

- Pour chaque état, on regarde les signaux générés

Etat	Q2..0	ZD	LDR	EINT	LDCPT	INT/!IDEC	EDeco	Store	Shift
A	000	0	*	1	0	*	0	0	0
B	001	0	1	1	0	1	0	1	0
C	011	0	0	1	1	1	1	0	0
D	010	0	*	1	0	*	0	0	0
E	111	1	*	1	0	0	0	0	0
F	101	0	*	0	1	0	0	0	1

Conclusion

- La méthode d'implémentation « câblée » des automates est propre aux µP de type RISC :
  - RISC = Reduced Instruction Set Computer
    - *E.g.* PowerPC, MIPS, …
  - Le format des instructions est uniforme et on peut facilement « tirer » des câbles à partir du registre IR
  - La plupart des instructions se font en un cycle et sont donc facilement représentable sous forme d'automates
- Pour les CISC…
  - Complex Instruction Set Computer
    - *E.g.* Pentium x86, Motorolla 68xxx, …
  - Automate monstrueux ⇒ il faut une autre méthode…