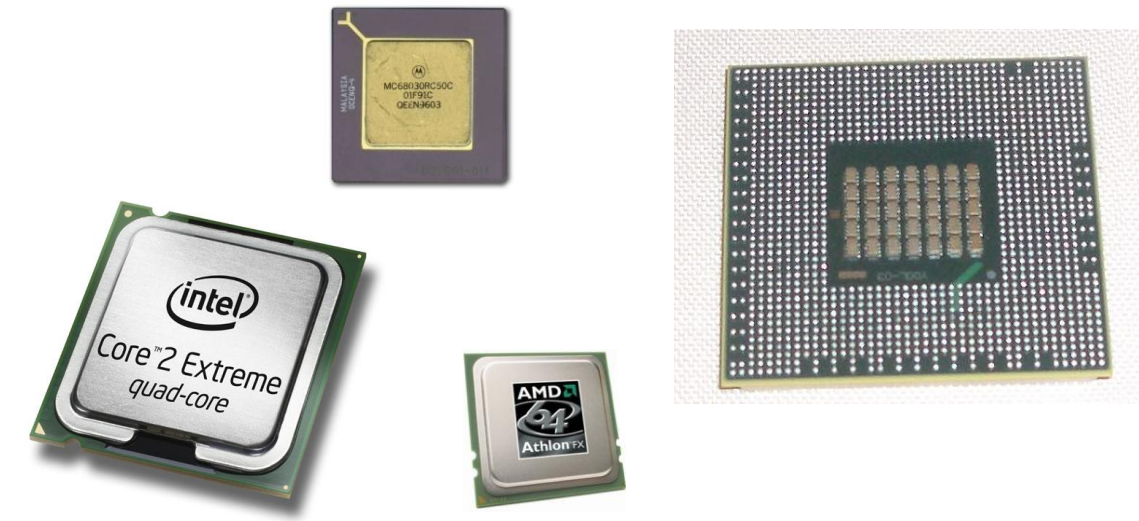
Architecture des systèmes à processeurs – IUT GEII (ISI-II2) - 5

Christophe BLANC
www.christophe-blanc.info
IUT de Montluçon
Département Génie Electrique et Informatique Industrielle



Architecture des systèmes à processeurs

LES INTERRUPTIONS

Sommaire

- Introduction – Notion d'interruptions
- Système d'interruptions hiérarchisées
- Les causes d'interruptions – Classification
- Mécanisme d'interruption
- Interruptions de la famille Renesas M32C80

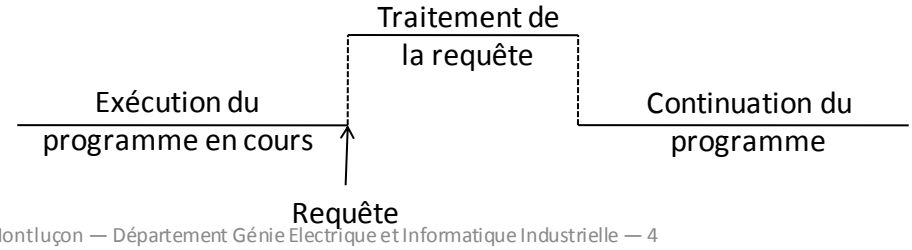
Introduction – Notion d'interruptions

Le microprocesseur est en permanence susceptible d'exécuter un programme.

Problème : à quel moment le microprocesseur va prendre en compte les **événements** extérieurs à la séquence d'instructions qu'il exécute?

- Requête d'un périphérique,
- Appui sur un bouton poussoir ,
- Passage d'un objet devant un capteur,
- …

A chacun de ces événements correspond une tâche à exécuter par le microprocesseur. Cette tâche est codée sous forme d'une procédure. Pour pouvoir exécuter cette procédure il faut que se produise une rupture de séquence. Cette rupture doit avoir lieu dans un délai assez court.

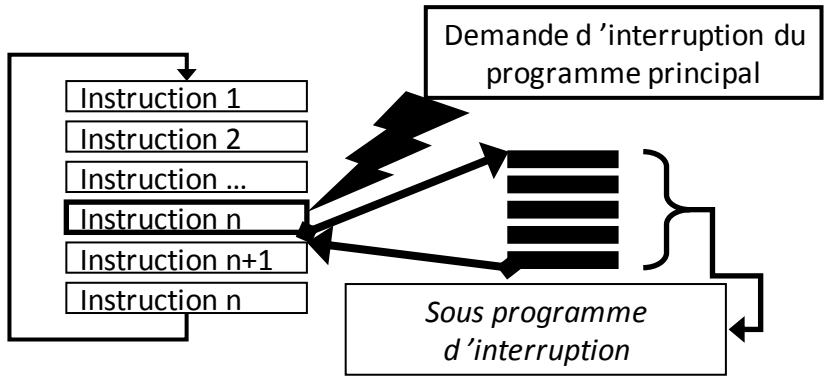


Introduction – Notion d'interruptions

Comment signaler au microprocesseur un événement asynchrone?

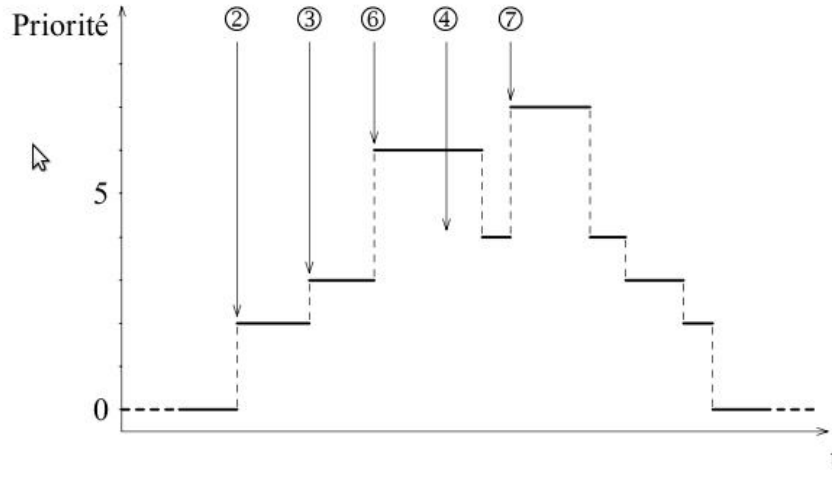
Solution : **système d'interruptions**

1. Interrompre « brutalement » l'exécution d'un programme en cours à la fin de l'instruction courante
2. Sauvegarder l'état présent du microprocesseur
3. Exécution d'un sous programme dépendant de la nature de l'interruption
4. Restitution de l'état du microprocesseur



Système d'interruptions hiérarchisées

A la notion d'interruption s'est très rapidement associée la notion de priorité, de manière à ce que certaines requêtes soient servies avant d'autres.
La figure représente un enchaînement de ruptures de séquence provoquées par des interruptions hiérarchisées.

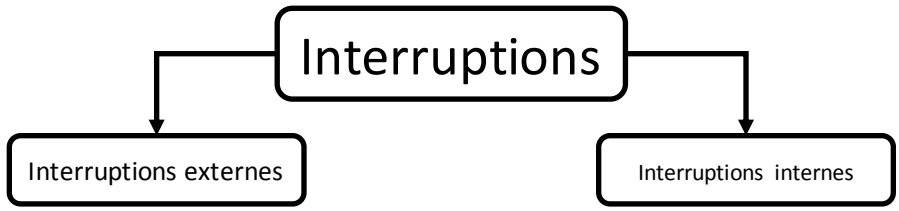


Système d'interruptions hiérarchisées

On trouve les systèmes d'interruptions les plus élaborés dans les ordinateurs orientés vers les applications "temps réel" de conduite de processus industriels ou d'acquisition de données. Dans certains cas chaque niveau est découpé en sous-niveaux de priorités différentes. Un bon système doit permettre au programmeur de pouvoir :

- activer/invalider le système d'interruption dans son ensemble;
- armer/désarmer chacune des interruptions individuellement : une interruption désarmée est ignorée;
- masquer/démasquer individuellement chaque interruption : une interruption masquée n'est pas ignorée, elle est mémorisée, mais elle n'est prise en compte que lorsqu'elle est démasquée;
- établir une hiérarchie entre les sources d'interruption avec plusieurs niveaux de priorité, si possible de façon dynamique;
- associer un programme spécifique à chaque interruption.

Les causes d'interruptions - Classification



- Matérielles : dues aux périphériques ou à des dispositifs extérieurs
- Logicielles : déclencher une interruption à l'aide d'une instruction spéciale

- Division par zéro
- Dépassement de capacité
- Erreur d'adressage
- etc

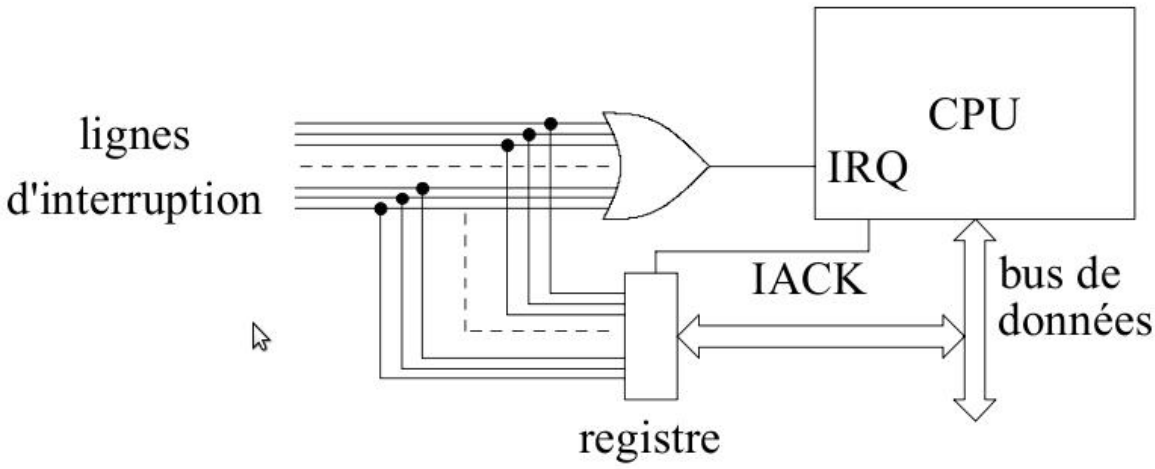
Mécanisme d'interruption

1. Identification de la source de l'interruption

Le microcontrôleur dispose d'une entrée spéciale, généralement appelée **IRQ** (Interrupt ReQuest), associée à un bit dit **bit d'interruption**. Avant de passer à l'instruction suivante, le microprocesseur teste l'état de ce bit. S'il est à 1 le microprocesseur est informé d'une demande d'interruption. Pour pouvoir la traiter il doit en déterminer l'origine. Il existe deux méthodes principales pour identifier la source d'une interruption : la **scrutation** (polling) et **l'identification directe**. Le choix entre ces deux méthodes est généralement le résultat d'un compromis coût/performance selon en particulier le nombre potentiel de périphériques qui peuvent être connectés. Etudions quelques mécanismes d'identification de la source d'une interruption. De très nombreuses architectures sont possibles.

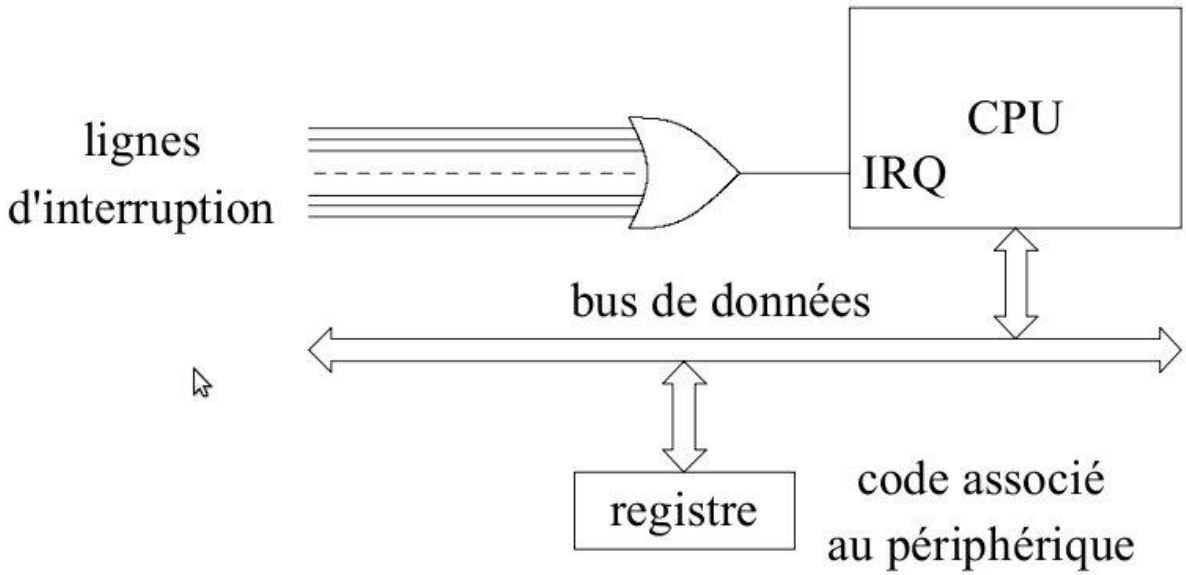
Mécanisme d'interruption

1. Identification de la source de l'interruption



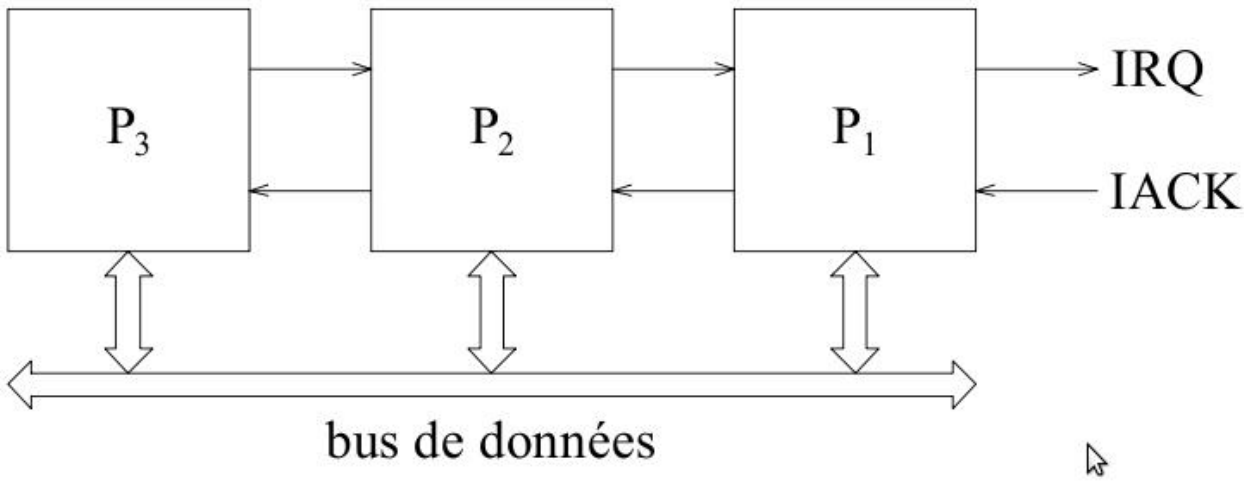
Mécanisme d'interruption

1. Identification de la source de l'interruption



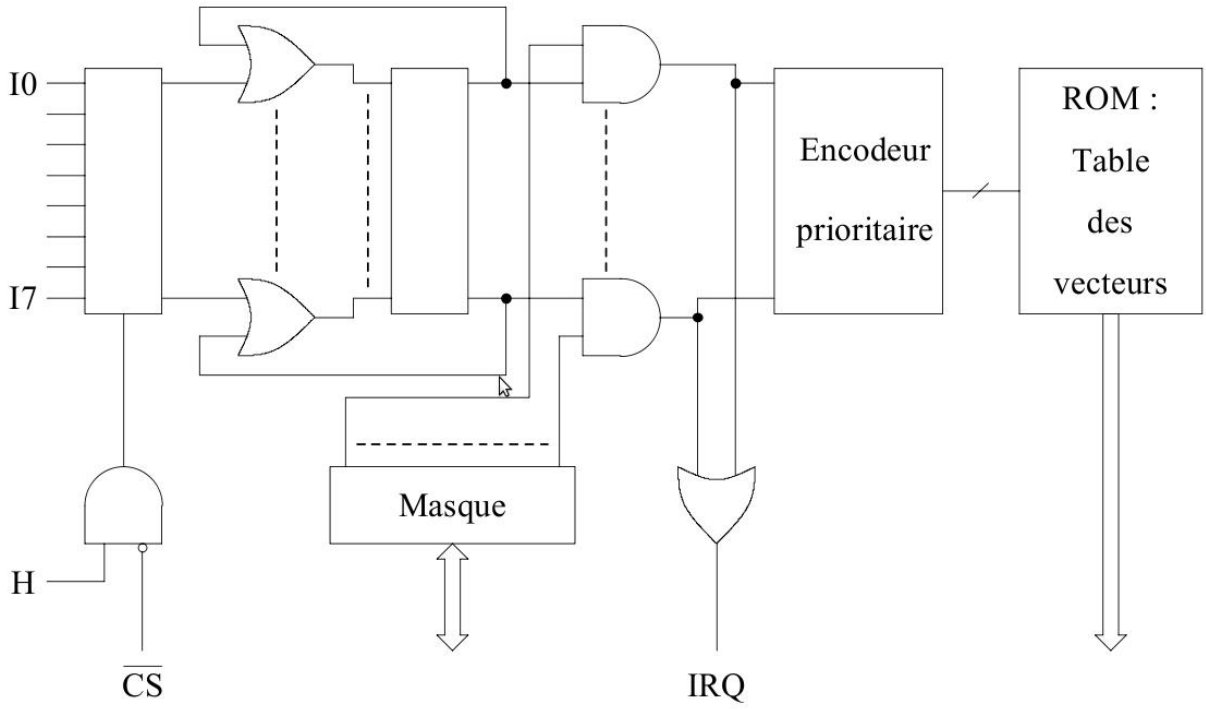
Mécanisme d'interruption

1. Identification de la source de l'interruption



Mécanisme d'interruption

1. Identification de la source de l'interruption



Mécanisme d'interruption

2. Algorithme de déroulement

1. Terminer l'instruction en cours
2. Sauvegarder l'état du processeur
3. Interdire les interruptions de niveau moins élevé
4. Accéder au sous programme d'exception
5. Exécuter le sous programme d'exception
6. Restituer l'état du processeur

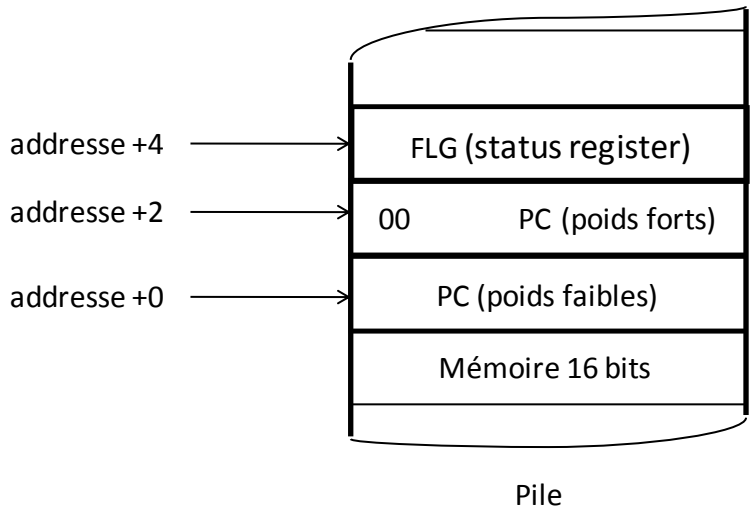
Mécanisme d'interruption

3. Exécuter le sous programme d'interruption

- Sauvegarder le contexte
  - Registres internes utilisés **dans le S.P.I.**
- **Exécuter l'action liée à l'interruption**
  - Rapidement (minimum d'instructions)
- Restituer le contexte
  - Registres internes utilisés dans le S.P.I.

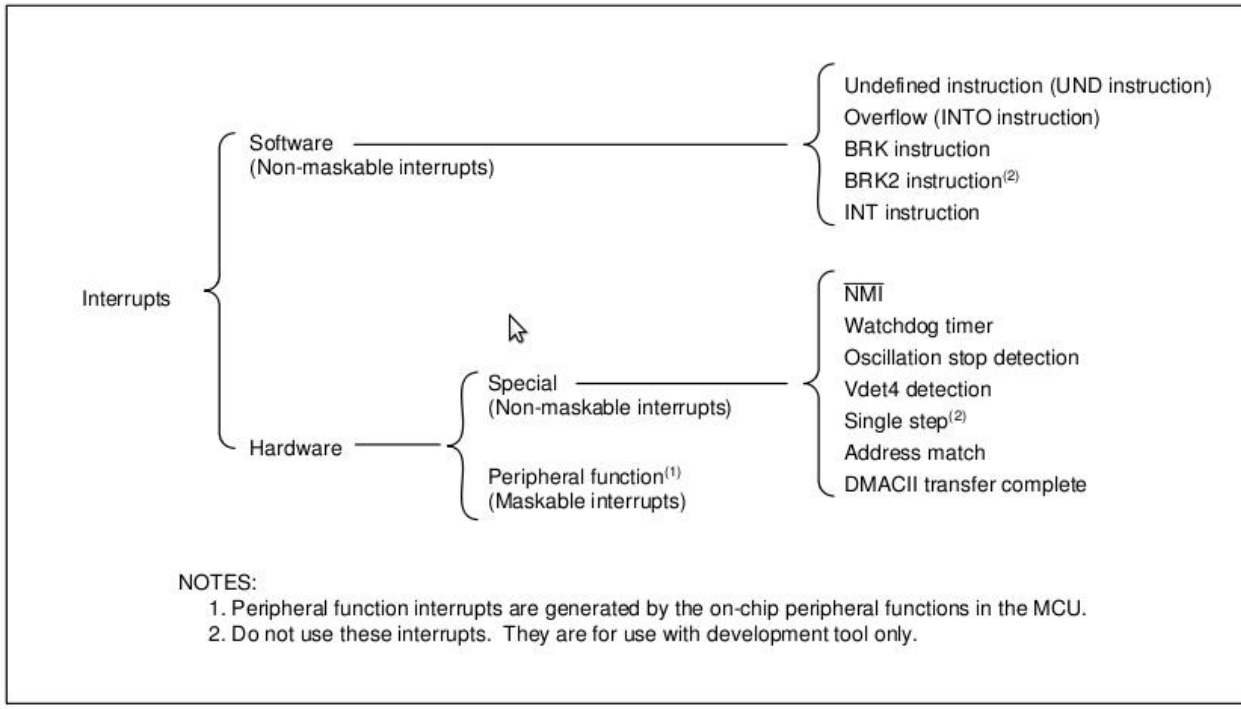
Mécanisme d'interruption

4. Sauvegarder l'état du processeur



Interruptions de la famille Renesas M32C80

1. Types d'interruptions

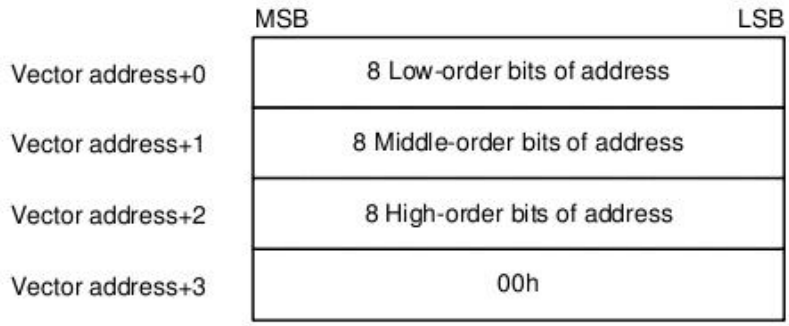


NOTES:
1. Peripheral function interrupts are generated by the on-chip peripheral functions in the MCU.
2. Do not use these interrupts. They are for use with development tool only.

Interruptions de la famille Renesas M32C80

2. Interruptions et vecteurs d'interruptions

- 4 octets dans chaque **vecteur d'interruption**
- Adresse d'une **fonction d'interruption** dans chaque **vecteur d'interruption**
- **Fonction d'interruption** exécutée à partir de l'adresse définie dans son **vecteur d'interruption**.



Un vecteur d'interruption

Interruptions de la famille Renesas M32C80

2. Interruptions et vecteurs d'interruptions

Table des vecteurs d'interruption figée (en mémoire)

Interrupt Source	Vector Addresses Address (L) to Address (H)	Remarks	Reference
Undefined instruction	FFFFDCh to FFFFDFh		M32C/80 series software manual
Overflow	FFFFE0h to FFFFE3h		
BRK instruction	FFFFE4h to FFFFE7h	If the content of the address FFFFE7h is FFh, the CPU executes from the address stored into software interrupt number 0 in the relocatable vector table.	
Address match	FFFFE8h to FFFFEBh		
–	FFFFECh to FFFFEFh	Reserved space	
Watchdog timer	FFFFF0h to FFFFF3h	These addresses are used for the watchdog timer interrupt, oscillation stop detection interrupt, and Vdet4 detection interrupt.	Reset, clock generation circuit, watchdog timer
–	FFFFF4h to FFFFF7h	Reserved space	
$\overline{\text{NMI}}$	FFFFF8h to FFFFFBh		
Reset	FFFFFCh to FFFFFFh		Reset

Interruptions de la famille Renesas M32C80

2. Interruptions et vecteurs d'interruptions

Table des vecteurs d'interruption modifiable (en mémoire) : l'adresse de début de la table des vecteurs est définie dans le registre **INTB**

256 octets à partir de l'adresse définie dans le registre INTB

Interrupt Source	Vector Table Address Address (L) to Address (H)(1)	Software Interrupt Number	Reference
BRK instruction(2)	+0 to +3 (0000h to 0003h)	0	M32C/80 Series Software Manual
Reserved space	+4 to +31 (0004h to 001Fh)	1 to 7	
DMA0	+32 to +35 (0020h to 0023h)	8	DMAC
DMA1	+36 to +39 (0024h to 0027h)	9	
DMA2	+40 to +43 (0028h to 002Bh)	10	
DMA3	+44 to +47 (002Ch to 002Fh)	11	
Timer A0	+48 to +51 (0030h to 0033h)	12	Timer A
Timer A1	+52 to +55 (0034h to 0037h)	13	
Timer A2	+56 to +59 (0038h to 003Bh)	14	
Timer A3	+60 to +63 (003Ch to 003Fh)	15	
Timer A4	+64 to +67 (0040h to 0043h)	16	
UART0 transmission, NACK(3)	+68 to +71 (0044h to 0047h)	17	Serial interfaces
UART0 reception, ACK(3)	+72 to +75 (0048h to 004Bh)	18	
UART1 transmission, NACK(3)	+76 to +79 (004Ch to 004Fh)	19	
UART1 reception, ACK(3)	+80 to +83 (0050h to 0053h)	20	

Interruptions de la famille Renesas M32C80

2. Interruptions et vecteurs d'interruptions

Table des vecteurs d'interruption modifiable (en mémoire) : l'adresse de début de la table des vecteurs est définie dans le registre **INTB**

Timer B0	+84 to +87 (0054h to 0057h)	21	Timer B
Timer B1	+88 to +91 (0058h to 005Bh)	22	
Timer B2	+92 to +95 (005Ch to 005Fh)	23	
Timer B3	+96 to +99 (0060h to 0063h)	24	
Timer B4	+100 to +103 (0064h to 0067h)	25	
$\overline{\text{INT5}}$	+104 to +107 (0068h to 006Bh)	26	Interrupts
$\overline{\text{INT4}}$	+108 to +111 (006Ch to 006Fh)	27	
$\overline{\text{INT3}}$	+112 to +115 (0070h to 0073h)	28	
$\overline{\text{INT2}}$	+116 to +119 (0074h to 0077h)	29	
$\overline{\text{INT1}}$	+120 to +123 (0078h to 007Bh)	30	
$\overline{\text{INT0}}$	+124 to +127 (007Ch to 007Fh)	31	
Timer B5	+128 to +131 (0080h to 0083h)	32	Timer B
UART2 transmission, NACK(3)	+132 to +135 (0084h to 0087h)	33	Serial interfaces
UART2 reception, ACK(3)	+136 to +139 (0088h to 008Bh)	34	
UART3 transmission, NACK(3)	+140 to +143 (008Ch to 008Fh)	35	
UART3 reception, ACK(3)	+144 to +147 (0090h to 0093h)	36	
UART4 transmission, NACK(3)	+148 to +151 (0094h to 0097h)	37	
UART4 reception, ACK(3)	+152 to +155 (0098h to 009Bh)	38	

Interruptions de la famille Renesas M32C80

2. Interruptions et vecteurs d'interruptions

Table des vecteurs d'interruption modifiable (en mémoire) : l'adresse de début de la table des vecteurs est définie dans le registre **INTB**

Bus conflict detection, Start condition detection/ Stop condition detection (UART2)(3)	+156 to +159 (009Ch to 009Fh)	39	Serial interfaces
Bus conflict detection, Start condition detection/ Stop condition detection (UART3 or UART0)(4)	+160 to +163 (00A0h to 00A3h)	40	
Bus conflict detection, Start condition detection/ Stop condition detection (UART4 or UART1)(4)	+164 to +167 (00A4h to 00A7h)	41	
A/D0	+168 to +171 (00A8h to 00ABh)	42	A/D converter
Key input	+172 to +175 (00ACh to 00AFh)	43	Interrupts
Intelligent I/O interrupt 0, CAN10(5), UART5 reception	+176 to +179 (00B0h to 00B3h)	44	Intelligent I/O, CAN, UART5, UART6, $\overline{\text{INT}}$
Intelligent I/O interrupt 1, CAN11(5), UART5 transmission	+180 to +183 (00B4h to 00B7h)	45	
Intelligent I/O interrupt 2	+184 to +187 (00B8h to 00BBh)	46	
Intelligent I/O interrupt 3	+188 to +191 (00BCh to 00BFh)	47	
Intelligent I/O interrupt 4	+192 to +195 (00C0h to 00C3h)	48	
Intelligent I/O interrupt 5, CAN12(5), CAN1 wake-up	+196 to +199 (00C4h to 00C7h)	49	

Interruptions de la famille Renesas M32C80

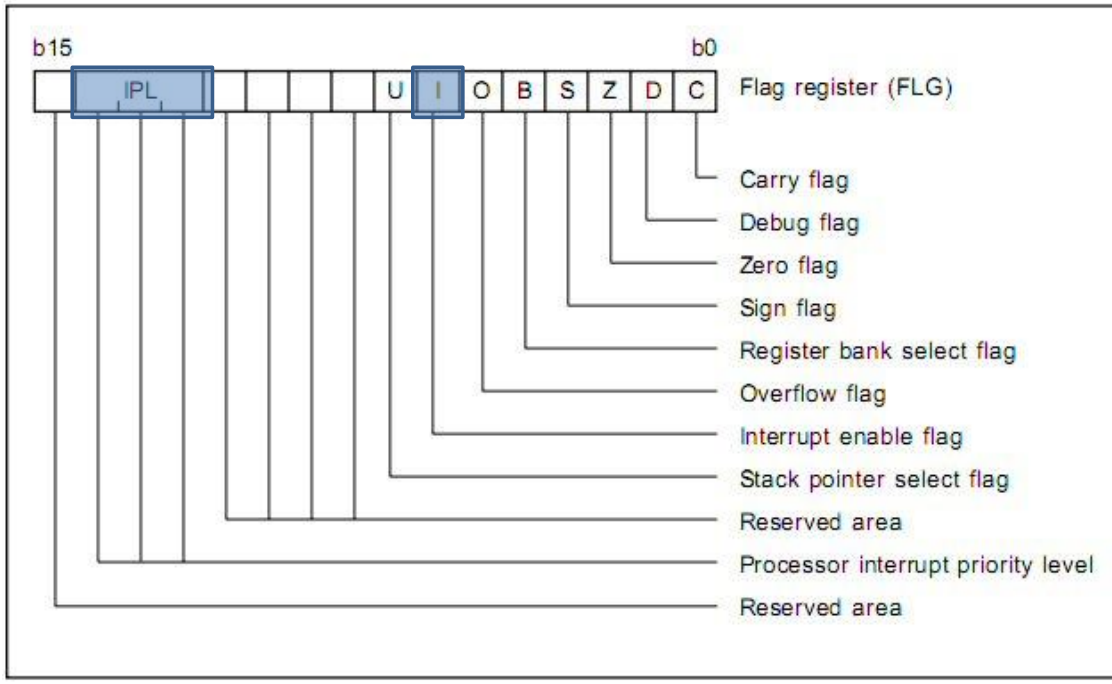
2. Interruptions et vecteurs d'interruptions

Table des vecteurs d'interruption modifiable (en mémoire) : l'adresse de début de la table des vecteurs est définie dans le registre **INTB**

Intelligent I/O interrupt 6	+200 to +203 (00C8h to 00CBh)	50	
Intelligent I/O interrupt 7	+204 to +207 (00CCh to 00CFh)	51	
Intelligent I/O interrupt 8	+208 to +211 (00D0h to 00D3h)	52	
Intelligent I/O interrupt 9, CAN00(5), UART6 reception, $\overline{INT6}$	+212 to +215 (00D4h to 00D7h)	53	
Intelligent I/O interrupt 10, CAN01(5), UART6 transmission, $\overline{INT7}$	+216 to +219 (00D8h to 00DBh)	54	
Reserved space	+220 to +227 (00DCh to 00E3h)	55, 56	–
Intelligent I/O interrupt 11, CAN02(5), $\overline{INT8}$	+228 to +231 (00E4h to 00E7h)	57	Intelligent I/O, CAN, $\overline{INT}$
Reserved space	+232 to +255 (00E8h to 00FFh)	58 to 63	–
INT instruction(2)	+0 to +3 (0000h to 0003h) to +252 to +255 (00FCh to 00FFh)	0 to 63	Interrupts

Interruptions de la famille Renesas M32C80

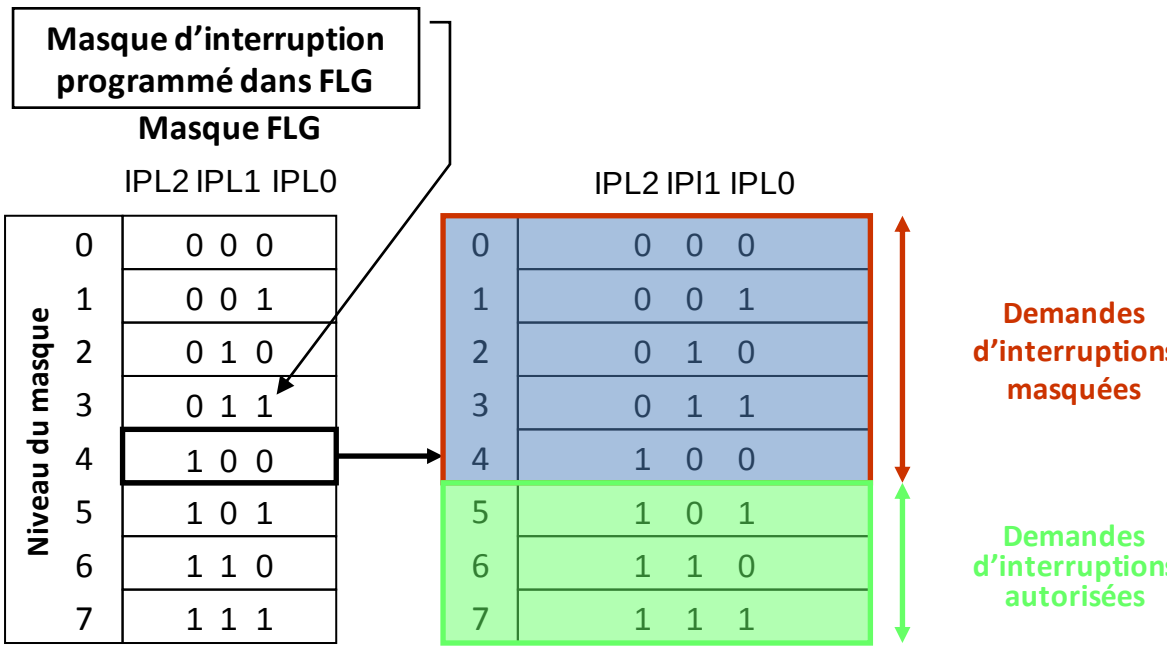
3. Registre d'état FLG



Quand le flag I est à 1 (enable), tous les interruptions masquables sont autorisées; quand I est à 0 (disable), elles sont bloquées.

Interruptions de la famille Renesas M32C80

4. masque d'interruptions



Interruptions de la famille Renesas M32C80

5. Intruction d'interruption INT

L'interruption spécifiée par l'instruction INT se produit lorsque l'instruction INT est exécutée. L'instruction INT permet de spécifier les interruptions logicielles (software interrupts) de 0 à 63. Les interruptions logicielles de 8 à 54 et 57 sont affectées à la table des vecteurs utilisés pour les fonctions d'interruptions liées aux périphériques. Cela signifie que le microcontroleur est en mesure d'exécuter les interruptions liées aux périphériques par l'instruction INT. Lorsque l'instruction INT est exécutée, le registre d'état FLG et le PC sont enregistrés dans la pile. Le vecteur de l'interruption logicielle spécifiée est stocké dans le PC.
La pile varie en fonction de l'interruption logicielle.