RÉSUMÉ EXCEPTIONS

Une exception est une **exception au déroulement normal d'un programme**. Lorsqu'elle est déclenchée par le matériel, elle est appelée "**exception matérielle**", catégorie qui comprend l'**initialisation** (déclenchée par le bouton "RESET"), **les interruptions matérielles** (déclenchées par un périphérique) et les **trappes matérielles** (déclenchées par le CPU).

INITIALISATION

Par exemple, l'**initialisation** du CPU (déclenchée par le matériel, ici la **ligne RST** du bus connectée au bouton "RESET") est une **exception matérielle** qui arrête le programme en cours et lance (***sans retour***) un **programme d'initialisation** depuis une adresse de démarrage fixe (FFFAh ici). Certaines machines disposent d'une instruction qui l'initialise (par exemple **RST**).

INTERRUPTION MATÉRIELLE

Un périphérique (matériel) peut ***intempestivement*** requérir l'***interruption*** du programme en cours pour lancer un programme spécifique à son service (dit **programme de service** ou **gestionnaire d'interruption** ou **d'exception** "exception handler").
Cette exception *déclenchée par un périphérique* s'appelle **interruption matérielle**.
A la fin du programme de service d'exception, le programme en cours reprend là où il a été interrompu.
L'exception matérielle et donc le lancement de son programme de service sont inopinés, contrairement à l'appel à un sous-programme qui est prédictible et placé à un endroit où sa modification de SR ne gêne pas. Une différence essentielle entre un appel à un **programme de service d'exception** et un sous-programme est donc que **le contenu du registre d'état SR est aussi sauvé en pile** avant **l'adresse de retour,** afin de restituer SR en plus du PC au retour au moyen de l'instruction **RTI** au lieu de RTS.

DÉCODAGE ET ENREGISTREMENT DES REQUÊTES D'INTERRUPTION

• **Interruptions non décodées:** une ligne de contrôle vers le CPU ¬**ITRQST** ("InTerrupt ReQueST") commune (OU câblé) est à 1.
- Tout périphérique peut alors mettre ¬ITRQST=0 et couper aussi la ligne **ITACK** ("InTerrupt ACKnowledge") le traversant ;
- recevant ¬ITRQST=0, le CPU sort ITACK=1, reçu uniquement par le 1er périphérique sur ITACK ayant requis l'interruption ;
- ce périphérique place alors son n° d'exception n sur le bus de données DBUS vers le CPU qui déclenche alors l'exception n°n.

• **Interruptions décodées:** il y a normalement une **ligne de requête d'interruption** IRQ_i ("Interrupt ReQuest" n°i) par périphérique. Ces lignes font normalement partie du bus de contrôle (32 ici lignes ici, e.g. 15 dans l'ancien bus ISA du PC/AT).

• **Enregistrement:** Lorsqu'un signal (ici un front ↑) est envoyé par un périphérique sur sa ligne de requête d'interruption IRQ_i au **contrôleur d'exception**, la requête de **numéro d'exception** (INT sur PC) n=f(i) (ici n = i + 32) est enregistrée dans l'indicateur ERF_n ("**Exception Request Flag**" n°n) du registre **ERR** ("**Exception Request Register**") du contrôleur d'exception ($1 \rightarrow ERF_n$).

VECTEURS D'EXCEPTION

Pour une **exception vectorisée** (e.g. interruption), **l'adresse du programme de service d'exception** n°n se trouve dans le **vecteur d'exception** n°n de **la table des vecteurs d'exception**, placée en mémoire centrale à une adresse connue (ici 0000h).

La composition du **vecteur d'exception** dépend des machines. Ici il a deux mots: le premier est l'**adresse du programme de service**, le second est réservé. En résumé, pour une interruption matérielle décodée et vectorisée:

MASQUAGE ET PRIORITÉ DES EXCEPTIONS MATÉRIELLES VECTORISÉES

• **Masquage:** Un drapeau supplémentaire du registre d'état SR (**IF** "**Interrupt Flag**", ici n°4) permet d'inhiber (on dit **masquer) les exceptions** matérielles vectorisées masquables (ici lorsqu'il est à 0). Sur beaucoup de machines une interruption matérielle vectorisée mais non masquable est déclenchée par une ligne de requête spéciale appelée **NMI** ("*Non-Maskable Interrupt*").
Des instructions (ici **ENI** et **DSI**) permettent de mettre IF respectivement à 1 ou 0 et donc de respectivement valider ou inhiber les exceptions matérielles vectorisées masquables. On peut donc **protéger** des **sections critiques** de programme contre les interruptions.

Les exceptions matérielles vectorisées masquables sont ici **automatiquement inhibées** (IF est mis à 0) :
- à l'**initialisation** pour qu'aucune requête d'exception ne soit lancée tant que les vecteurs n'ont pas encore été configurés;
- au **lancement d'un programme de service** pour qu'il ne soit pas lui-même intempestivement interrompu par une requête plus tardive ou moins prioritaire. (Mais le programme de service peut les valider à nouveau après sa section critique avec ENI).

• **Priorité:** Si plusieurs requêtes sont enregistrées dans ERR, la plus prioritaire (ici celle de n° n le plus faible) est servie d'abord ; si IF=**1** son programme de service (le n°n) est lancé et sa requête effacée ($0 \rightarrow ERF_n$).

ATTENTE D'INTERRUPTION

L'instruction d'attente (ici HLT) met le CPU en attente d'interruption: il s'arrête alors après **HLT** en consommant moins de puissance électrique et donc en chauffant moins. Il répond aussi plus vite à toute requête d'interruption.
Il ne repart qu'à la prochaine interruption pour exécuter son programme de service, puis en revenir pour exécuter **l'instruction qui suit HLT et continuer**. (Ici, l'attente est explicitement indiquée par un 1 dans l'indicateur n°5 WF "Wait Flag" du registre d'état SR).

TRAPPE MATÉRIELLE

Une **exception déclenchée par le CPU** est appelée **trappe.**

Certaines trappes sont matérielles et donc inopinées, déclenchées par un problème inattendu dans le CPU comme ici :

INT	TYPE	SIGNIFICATION	EXPLICATION
0	Fault	Faute de bus	Pas de mémoire physique à l'adresse indiquée ; lance le programme de service puis ***ré-exécute*** l'instruction fautive. Utilisé pour la gestion de la **pagination** (cf. mémoire virtuelle)
1	Trap	Instruction illégale	Le code d'instruction n'est pas valide
2	Trap	Désalignement	Tentative d'accéder à un opérande de type Word à une adresse impaire
3	Trap	Protection	***Réservé*** *(e.g. le CPU accède illégalement à une page mémoire cf. mémoire virtuelle)*
4	Trap	Division par zéro	L'instruction DIVision a été utilisée avec un diviseur nul.

TRAPPE LOGICIELLE

On peut appeler un programme de service d'exception vectorisée de manière déterministe (comme un sous-programme) en prenant certaines précautions; ceci s'appelle une **trappe logicielle** ou interruption logicielle. L'instruction **TRP** n permet de lancer le programme de service d'exception n° n.
Ceci permet d'implémenter les **appels système** (i.e. appel à un programme dit "primitive système" du système opérateur "OS").
Chaque appel système aura toujours le même numéro n (cf. doc. de l'OS) même si au cours des diverses versions de l'OS l'adresse du programme change. Les paramètres sont en registre. Primitives système ici implantées sur le simulateur *(par K. Proch)*:

INT	NOM	ACTION	COMMENTAIRE
64	**EXIT**	retourne au "système opérateur"	le programme s'arrête, le "système opérateur" reprend la main
65	**READ**	lit une chaîne de caractères du clavier	R0 pointe sur le tampon mémoire où loger la chaîne ASCII lue
66	**WRITE**	écrit une chaîne de caractères à l'écran	R0 pointe sur le début de la chaîne ASCII terminée par NUL

INSTRUCTIONS DE GESTION DES EXCEPTIONS ET SOUS-PROGRAMMES

	COMMENTAIRE	INSTRUCTION	ACTIONS DU CPU
SOUS PROG	Appel au sous-programme ssp	**JSR *ssp***	adresse de retour → pile ; adresse de ssp → PC;
	Retour de sous-programme: ***SR non restitué***	**RTS**	pile → PC
EXCEPTION	initialisation logicielle *(n'existe pas ici)*	**RST**	FFFAh → PC, NOP→IR, 0000h→ SR (⇒ 0 → IF ...)
	Appel au programme de service d'exception vectorisée n° ***n***	**TRP *n***	SR → pile; PC → pile; adresse du prog. de service n℔ = M[4n] → PC;
	Retour de programme de service d'exception vectorisée: ***restitue aussi SR***	**RTI**	pile → PC; pile → SR; 0 → WF *(oublié dans le simulateur) ;*
	validation exceptions matérielles vectorisées	**ENI**	1 → IF
	inhibition exceptions matérielles vectorisées	**DSI**	0 → IF
	arrêt et mise en attente d'une interruption	**HLT**	1 → WF

EXCEPTIONS MATÉRIELLES

	COMMENTAIRE	ÉVÉNEMENT	ACTIONS DU CPU
ENREGISTREMENT DE REQUÊTE	enregistrement de requête d'**interruption matérielle** *(vectorisée, décodée et masquable)* par un **périphérique sur ligne n℔**	**IRQi ↑**	• **calcule le n°d'exception** n = INT = i+32 ; • **enregistre la requête** n℔ dans le drapeau n℔ ERF $_n$ du registre ERR du contrôleur d'exceptions : 1 → ERF_n
	enregistrement de requête de **trappe matérielle** *(vectorisée et masquable)* n℔ (le n°d'exception n est appelé INT sur le PC)	**pb interne CPU**	• calcule n℔d'exception = INT = n (cf. tableau des t rappes); • **enregistre la requête** n℔: 1 → ERF_n
LANCEMENT	lancement d'exception matérielle (vectorisée et masquable) n°n (dit INT sur PC)	**ERR≠0**	• le CPU termine l'instruction en cours ; • **si IF=1** le CPU **lance le prog de service** de la requête enregistrée la plus **prioritaire** (ici de n℔ le plus faible) i.e.: 0 → ERF_n; SR → pile ; PC → pile ; 0 → WF ; 0 → IF ; adresse du prog. de service n℔ = M[4n] → PC;
	initialisation *(ni vectorisée ni masquable)*	**RST=1**	FFFAh→PC, NOP→IR, 0→SR (0→IF, 0→WF), 0→ERR