簡介

中斷和例外

中斷（interrupts）有兩種來源：由硬體產生的和由軟體產生的。通常，在硬體週邊設備需要 CPU 時（例如，要跟 CPU 要資料，或是有資料要給 CPU），會對 CPU 發出中斷要求。軟體也可以利用 INT n 指令來對 CPU 發出中斷要求。
例外（exceptions）分為三種：程式錯誤例外、軟體產生例外、和機器錯誤例外。程式錯誤例外是在處理器偵測到程式有不合法的行為，或是作業系統發生某些錯誤時，由處理器自動發出的例外。這種例外又可分為三種：faults、traps、和 aborts。
軟體產生的例外，是由 INTO、INT3、和 BOUND 指令產生的。例如，INT3 命令會發出一個「程式中斷點」的例外。用 INT n 指令也可以「模擬」例外，但是，例外發生時，處理器會把錯誤碼推入堆疊中，而 INT n 指令則不會。所以，如果直接使用 INT n 指令呼叫例外處理程序，則該程序會從堆疊中取出錯誤碼，而在返回時，會回到錯誤的位址。
在 Pentium 以後的處理器，可以檢查其內部是否有錯誤。在處理器有錯誤時，會產生機器錯誤例外。

中斷向量

中斷和例外是利用一個數字來區分不同的中斷或例外，這個數字稱為「中斷向量」（interrupt vector）。中斷向量是一個由 00H 到 FFH 的數字。其中，00H 到 1FH 的中斷向量是保留作系統用途的，不可任意使用；而其它的中斷向量則可以自由使用。保留的中斷向量如下表所示：

向量編號	助憶碼	說明	型態	錯誤碼	來源
00H	#DE	除法錯誤	Fault	無	DIV 和 IDIV 指令。
01H	#DB	除錯	Fault/Trap	無	任何對程式或資料的參考、或是 INT 1 指令。
02H	-	NMI 中斷	Interrupt	無	不可遮罩的外部中斷。
03H	#BP	中斷點	Trap	無	INT3 指令。
04H	#OF	溢出	Trap	無	INTO 指令。
05H	#BR	超出 BOUND 範圍	Fault	無	BOUND 指令。
06H	#UD	非法的指令	Fault	無	UD2 或未定義的指令碼。
07H	#NM	沒有 FPU	Fault	無	浮點運算指令或 WAIT/FWAIT 指令。
08H	#DF	雙重錯誤	Fault	有	任何會產生例外的指令。
09H	-	保留	Fault	無	386 以後的處理器不產生此例外。
0AH	#TS	不合法的 TSS	Fault	有	工作切換、或存取 TSS。
0BH	#NP	分段不存在	Fault	有	載入或存取分段。
0CH	#SS	堆疊分段錯誤	Fault	有	載入 SS 載存器或存取堆疊。
0DH	#GP	一般性錯誤	Fault	有	存取記憶體或進行其它保護檢查。
0EH	#PF	分頁錯誤	Fault	有	存取記憶體。
0FH	-	保留			
10H	#MF	浮點運算錯誤	Fault	無	浮點運算指令或 WAIT/FWAIT 指令。
11H	#AC	對齊檢查	Fault	有	存取記憶體。
12H	#MC	機器檢查	Abort	無	和機器型號有關。
13H ~ 1FH	-	保留			
20H ~ FFH	-	自由使用			

例外的型態

程式錯誤例外有三種：

- Fault：這種例外通常是可以更正的，而且在更正後，程式應該可以繼續無誤地進行。在發生 fault 時，處理器會回到造成 fault 指令之前的狀況。在堆疊中存放的返回位址，存放的是造成 fault 的指令的位址。所以，在例外處理完，返回原程式時，會重新執行原來造成 fault 的指令。正常的話，應該不會再產生 fault。例如，最常見的 fault 應該是分頁錯誤（Page-fault），分頁錯誤的處理程序應該要讀入所需要的分頁，再返回程式中，讓程式可以繼續執行。
- Trap：這種例外是在被 trap 的指令執行完後，立刻產生的。因此，它的返回位址是被 trap 的指令的下一個指令。在例外返回後，程式可以正確無誤地斷續執行。如果被 trap 的是一個跳躍指令（例如，JMP 指令）也不會有問題，它的返回位址會是 JMP 指令的目標。
- Abort：這種例外是非常嚴重的錯誤，沒有辦法返回原程式繼續執行。通常，只有在機器錯誤、或是在系統表格中有不一致或不合法的數值時，才會出現這種例外。這類例外的處理程序，應該是把錯誤發生時的狀態存下，並儘可能安全地關閉應用程式和系統。

因為在中斷或（有些）例外發生時，原程式必須能繼續執行，因此，中斷和例外一定是在指令和指令中間發生，而不會在指令進行的途中發生。