簡介

概論

「工作」（task）代表一個執行的程式。它可能是一個應用程式、一個作業系統的服務、或是一個中斷 / 例外處理程序、或是作業系統的核心。即使是最簡單的系統，至少也會有一個工作。而複雜的系統可能會支援多工處理的能力，即同時有多個工作同時執行。
在保護模式中，一個工作是由「工作執行空間（task execution space）」和「工作狀態分段（task-state segment，TSS）」所組成的。工作執行空間就是一個工作的活動空間，由一個程式碼 segment、一個堆疊 segment、和一個或多個資料 segment 組成。如果作業系統有使用保護機制的話，每個工作還要為各個特權等級分別準備一個分開的堆疊 segment。而 TSS 則是指出工作執行空間的位址，並存放工作的狀態。如果系統中有多個工作的話，它還負責把各個工作連結起來。
TSS 是一個系統 segment，因此它有自己的 segment descriptor。在系統中，就是利用它的 segment selector 來表式一個工作 TSS。在載入一個工作執行時，TSS 的 segment selector、基底位址、邊界值、和一些 segment descriptor 的屬性，都會載入到工作暫存器（task register，TR）中。同時，若有使用分頁功能，則在工作切換時，還會把該工作的分頁目錄的基底位址載入到 CR3 中。

工作狀態

在 TSS 中，存放著工作的狀態。工作的狀態由下列幾項來決定：

- 分段暫存器（CS、DS、ES、FS、GS、SS）的值，這些暫存器決定工作的執行空間位址。
- 通用暫存器的值。
- EFLAGS 的值。
- EIP 的值。
- CR3 的值。
- 工作暫存器（TR）的值。
- LDTR 的值。
- I/O 對映的基底位址和 I/O 對映表。
- 高權限（ring 0 到 ring 2）堆疊的堆疊指標。
- 到上一個執行的工作的連結。

在執行工作之前，TSS 中包含上面所有的資料（除了工作暫存器之外）。此外，在 TSS 中的 LDTR 的值，只包括 segment selector 的部分，而不包括其它部分。

執行工作

執行一個工作，就和呼叫一個程序相當類似。下面的五種方法，都可以執行一個工作：

- 直接用 CALL 指令呼叫工作。
- 直接用 JMP 指令跳躍到工作中。
- 因為發生中斷，而跳到中斷處理工作中。
- 因為發生例外，而跳到例外處理工作中。
- 在 IRET 指令返回時，在 EFLAGS 中的 NT 旗標是設為 1。

這些方法中，可以直接指向一個 TSS，或是經由 task-gate 來指向 TSS，指出目標工作。在利用 CALL 或 JMP 指令時，可以直接指向 TSS 或 task-gate；但是在中斷或例外時，在 IDT 的 entry 中，一定要經由 task-gate 來指向 TSS，而不能直接指向 TSS。
在執行另一個工作時，會導致工作切換。處理器會把目前工作的狀態存到 TSS 中，並暫停目前工作的執行，然後載入新工作的 TSS，開始執行新的工作。如果新的工作是第一次執行，EIP 會指向該工作的第一個程式碼；反之，EIP 會指向上次離開時，執行的指令的下一個指令。同時，處理器會在 TSS 中存放上一個工作的位址。在所有 Intel Architecture 的處理器中，工作都是不可遞迴的，也就是說，工作不可以呼叫自己。
中斷和例外也可以用一個分開的工作來處理，處理器會在處理程序完成後，自動切換回原來的工作。如此一來，可以使中斷或例外的處理程序有一個分開的工作空間。這種方式，可以使在中斷處理程序中，仍然可以處理中斷的發生。
在工作切換時，處理器也會重新載入 LDTR 的值。這可以使各個工作定義自己的分段方式。如果有使用分頁功能，則在工作切換時，還會重新載入 CR3 的值。這個以讓各個工作有自己的分頁方式。這對保護機制非常重要，因為若不將各個工作的 LDT 和分頁方式獨立的話，各個工作就可以互相干擾，破壞對方的定址空間了。即使利用特權等級的機制來保護系統，但是各個同等級的工作仍然可以互相存取對方的資料。所以，一個安全的系統，應該要為各個工作指定獨立的 LDT 和分頁表。