OSTeam19\_Report

Team\_member：鍾昀諠，陳麒懋

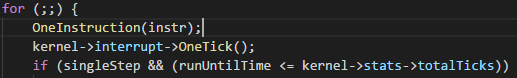
* **Trace system calls**
* **machine.h**

****

在ExceptionType裡有定義一些instruction的類別。而這次作業所碰到的則是SyscallException，當instruction指令有用到Syscall的話，它會被傳入RaiseException()裡。

* **mipssim.cc**

首先，先把程式碼透過Compiler轉換成MIPS，然後將它load到machine來處理。其中在Machine::Run()裡有OneInstruction()來處理MIPS。

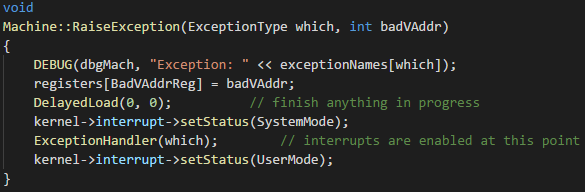


在OneInstruction()中，假如讀到SystemCall指令則會進入case OP\_SYSCALL，然後將SyscallException傳入RaiseException()當中。



* **machine.cc**

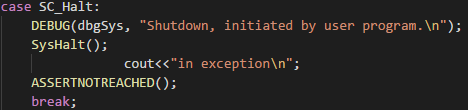
之後，產生interrupts將status由UserMode改為SystemMode。接著將ExceptionType傳進ExceptionHandler()去做處理。做完之後再將status變回UserMode。

****

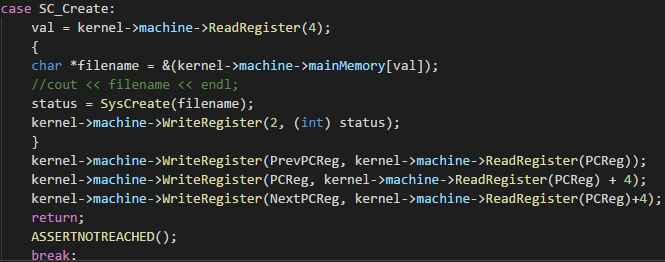
* **exception.cc**

ExceptionHandler()在此檔實作。

**Halt()**



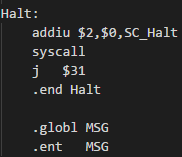
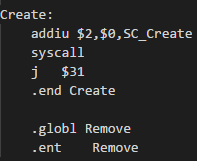
**Create()**



為了讓kernel知道哪個system call被呼叫，必須先在system call的interface也就是syscall.h中去定義SC\_Halt,SC\_Create和函數Halt(),

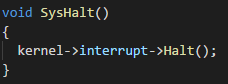
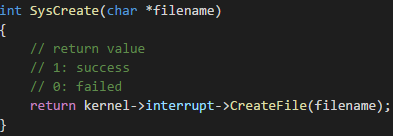
Create()，以方便在exception.cc和start.S中用到。

* **Start.S**

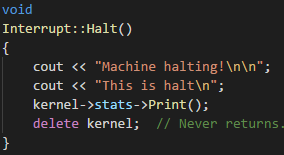
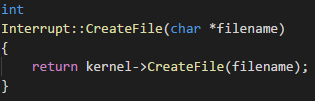
* **ksyscall.h**

ExceptionHandler()之後則是進入SysHalt()和SysCreate()。

* **interrupt.h/interrupt.cc**

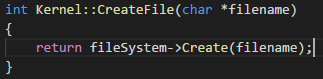
之後到interrupt::Halt()和interrupt::CreateFile()。

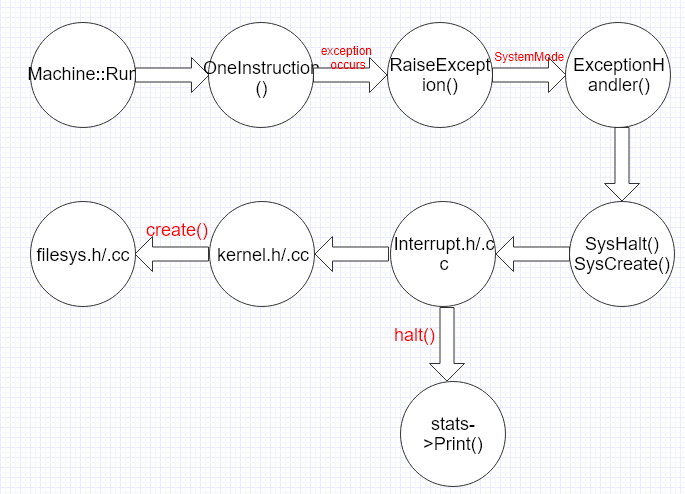
Halt()最後會到stats.cc做Print()的動作。

* **kernel.h/kernel.cc**

進入Kernel:: CreateFile()然後到filesys.h/filesys.cc中實作。

****

* **Diagram**

****