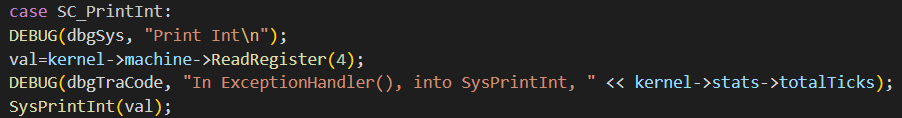
(c) PC\_PrintInt:

1. exception.cc

exceptionHandler ( ) : 先進到system call exception的case printInt，再從reg[4]讀進來要print的value，再將此value當作參數去call ksycall.h中的SysPrint。



1. ksycall.h

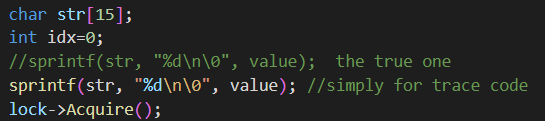
SysPrintInt ( ) : 這個function主要負責呼叫synchConsoleOut中的PutInt而已，將上一層的value往下一層傳。

一張含有 文字 的圖片

自動產生的描述

1. synchconsoleout.cc

SynchConsoleOut::PutInt ( ) : 先用一個字串str去記傳進來的value，接著啟動lock->Acquire()。



Locking 出鎖機制主要在避免同時呼叫I/O，acquire會主動等到lock變成free狀態時再將其設為busy，release則會將lock設成free並喚醒在等待的thread。

接著，進入do-while迴圈，在str[idx]不為空字元的情況下持續呼叫consoleOutput的PutChar。

同時開啟waitFor->P ( )。

一張含有 文字 的圖片

自動產生的描述

waitFor機制分成P ( )和V ( )兩種，P ( ) 會在值 > 0時持續遞減，V ( ) 則會先增加值並call一個在P ( ) 裡面等待的thread。

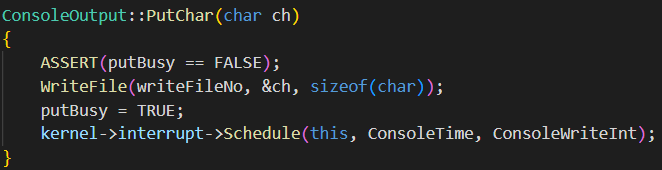
SynchConsoleOut::PutChar ( ) : 同樣概念，只是差在不用用string去記且不用用do-while迴圈去ca持續呼叫consoleOutput的PutChar。

一張含有 文字 的圖片

自動產生的描述

1. console.cc

ConsoleOutput : : PutChar ( ) : 將要output的char傳給WriteFile display，並將設為busy狀態且最後schedule一個interrupt。



1. interrupt.cc

Interrupt : : Schedule ( ) : 先計算此interrupt之後排到的時間，也就是目前的時間 ( totalTicks ) + 從現在往後數的時間 ( fromNow )。接著將這個interrupt insert進pending的interrupt序列中。

其中的CallBackObj是在等等提到的callback function中會用到的。

一張含有 文字, 螢幕擷取畫面, 螢幕, 靠近 的圖片

自動產生的描述

1. mipssim.cc

Machine : : Run ( ) : 在無限迴圈中跑到OneTick ( )時會檢查pending的interrupt。

一張含有 文字 的圖片

自動產生的描述

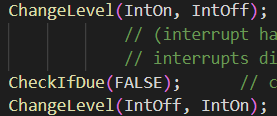
1. interrupt.cc

Machine : : OneTick ( ) : 首先將kernel的stats讀進來，再將ticks 根據user or system mode做refresh。

一張含有 文字 的圖片

自動產生的描述

再來透過level的changing來turn on / off interrupt，才能用CheckIfDue去檢查有沒有在pending的interrupt。



最後就是看有沒有要產生yield，如果yield的話會擱置目前的thread並執行下一個thread，如果沒有下一個thread的話就繼續run現在這個。

一張含有 文字 的圖片

自動產生的描述

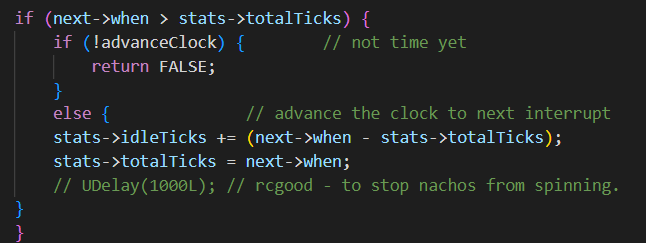
1. interrupt.cc

Interrupt : : CheckIfDue ( ) : 一樣將kernel的stats讀進來，如果no pending interrupt就return False。如果有的話用變數next去記pending的第一個。

一張含有 文字 的圖片

自動產生的描述

接著檢查目前system的totalTicks跟next的when大小，如果還沒到next->when，就將totalTicks指到next->when。



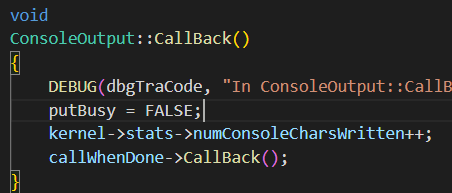
最後在do-while迴圈中呼叫callback function，並將做完的delete掉。

一張含有 文字 的圖片

自動產生的描述

1. console.cc

ConsoleOutput : : CallBack ( ) : 這個function主要將busy的狀態設回free，且更新kernel裡面的stats ( numConsoleCharsWritten )，最後呼叫SynConsoleOutput的callback。



1. synchconsole.cc

SynConsoleOutput : : CallBack ( ) : 這裡是對應到waitFor關係中的V ( )，詳情見第3點的最後。

一張含有 文字 的圖片

自動產生的描述