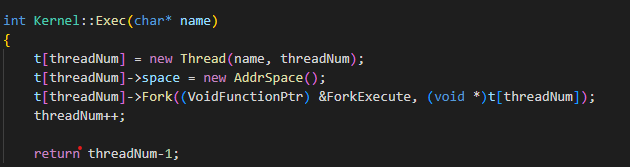
有兩個register set一個for user 一個for kernel

Sleep():

如果current thread完成了或被同步block住,放棄CPU使用權,如果沒有thread在ready queue中,CPU idle直到下一個I/O interrupt

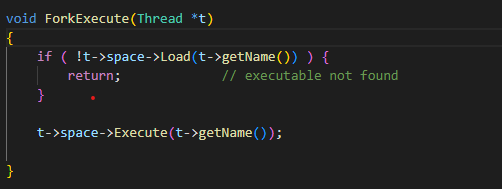
Kernel.cc的Exec()



New一個thrad,該thread的address space並且執行Fork跟ForkExecute

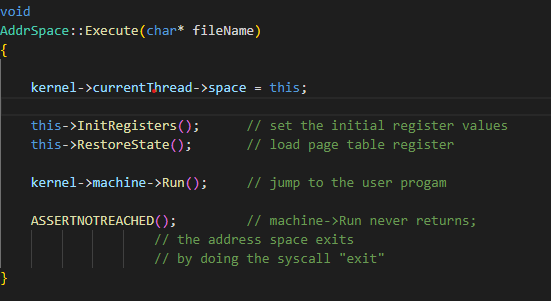
Kernel.cc的ForkExecute(),Load():

建造一條Thread,並且建立address space for user program



在User/History/qsa1.cc,使用Load()把user program從file載入到memory,Execute()表示如果program已經被load進memory就執行這個函式

Addrspace.cc的Execute():



Load進memory後要使用當前的Thread執行user program,初始化user register,如果要context switch必須保存好這個register的資訊(page table所在的位置),最後執行MIPS-RUN程式

Thread.cc的Fork():

同時執行caller跟callee,會先分配Stack,當scheduler切到某thread時可以確保執行該thread的程式碼,並且把此thread放進ready queue

Thread.cc StackAllocate():

分配以及宣告stack

Scheduler.cc的ReadToRun():

設定thread為ready state,並且放進ready list.

threads/kernel.cc

ExecAll就是要Main Thread(Kernel)依序去執行(Exec)所有要執行的程式(Thread)

Scheduler.cc Run():

必須先確保interrupt關掉,如果目前thread結束了,把現在的thread mark起來並刪除,如果這個thread是user program會儲存user register,下一個thread進來,並且設為Running state

在AddrSpace建構子加入:

bool AddrSpace::usedPhyPage[NumPhysPages]={false};

AddrSpace::Load()加入:

pageTable = new TranslationEntry[numPages];

    for(unsigned int i = 0, j = 0; i < numPages; i++) {

        pageTable[i].virtualPage = i;

        while(j < NumPhysPages && AddrSpace::usedPhyPage[j] == true)

            j++;

        AddrSpace::usedPhyPage[j] = true;

        pageTable[i].physicalPage = j;

        pageTable[i].valid = true;

        pageTable[i].use = false;

        pageTable[i].dirty = false;

        pageTable[i].readOnly = false;

    }

由於一開始initial physical address共用,使得兩個process access相同page.

增加了一個usedPhyPage紀錄已經被用過的page,並且使用virtual memory,到page table查詢該process轉譯到physical memory的frame.

1. How does Nachos allocate the memory space for a new thread(process)?

New一個address space並且分配Stack空間,initial register set

2. How does Nachos initialize the memory content of a thread(process), including loading the user binary code in the memory?

New一個thread,分配空間,分配Stack空間,關閉interrupt,把這個Thread放進ready queue裡,開啟interrupt,並且用scheduler控制正在執行的thread以及在ready state的thread.

3. How does Nachos create and manage the page table?

translate.h裡面會定義TranslationEntry, 可以操作pageTable做一些Virtual Memory相關的處理及轉譯

4. How does Nachos translate addresses?

Check page size有沒有超過,page有無合法  
5. How Nachos initializes the machine status (registers, etc) before running a thread(process)

在thread.cc建構子initialize

6. Which object in Nachos acts the role of process control block

Thread.cc裡有RestoreRegister()可以當作PCB

7. When and how does a thread get added into the ReadyToRun queue of Nachos CPU scheduler?

thread.c的Fork中，會呼叫scheduler->ReadyToRun(this),此行會將已經分配好資源的Thread放入Ready Queue，以供未來CPU排班執行