lab5

PB21020552 张易

思考题

在上下文切换的现场维护中，pushf和popf对应，pusha和popa对应，call和ret对应，但是为什么 CTS SW 函数 中只有ret而没有call呢?

CTS_SW函数是一个子例程，它被其他代码调用。在调用该函数之前，已经使用call指令将控制流转移到该函数，并在函数执行完毕后使用ret指令将控制流返回到调用者。因此，在CTS_SW函数内部，不需要再使用call指令。

对 stack init 函数的理解

stack_init函数是一个用于初始化任务栈的函数。它将函数指针task和一些特定的值存储到给定的栈地址stk指向的位置上。栈的底部地址由stk指向，而栈向下增长。首先存入任务的返回地址,表示任务返回的时候,会从这里读取数值然后返回相关的地址.,然后存入任务的入口地址,当任务第一次开始进行时,开始执行的位置就是其实地址.,然后在存入标志寄存器的初始值表示允许中断.通过调用这个函数,可以把栈初始化成一个特定的状态,准备执行特定的任务.

myTCB结构体定义中的stack[STACK SIZE]的作用是什么?BspContextBase[STACK SIZE]的作用又是什么?

stack[stack size]

一个栈,可以存入当前这个任务处理当前指令之后的对应的寄存器的值,以及stack_init中保留的一部分值.同时stack size还具有保护作用,防止外溢.

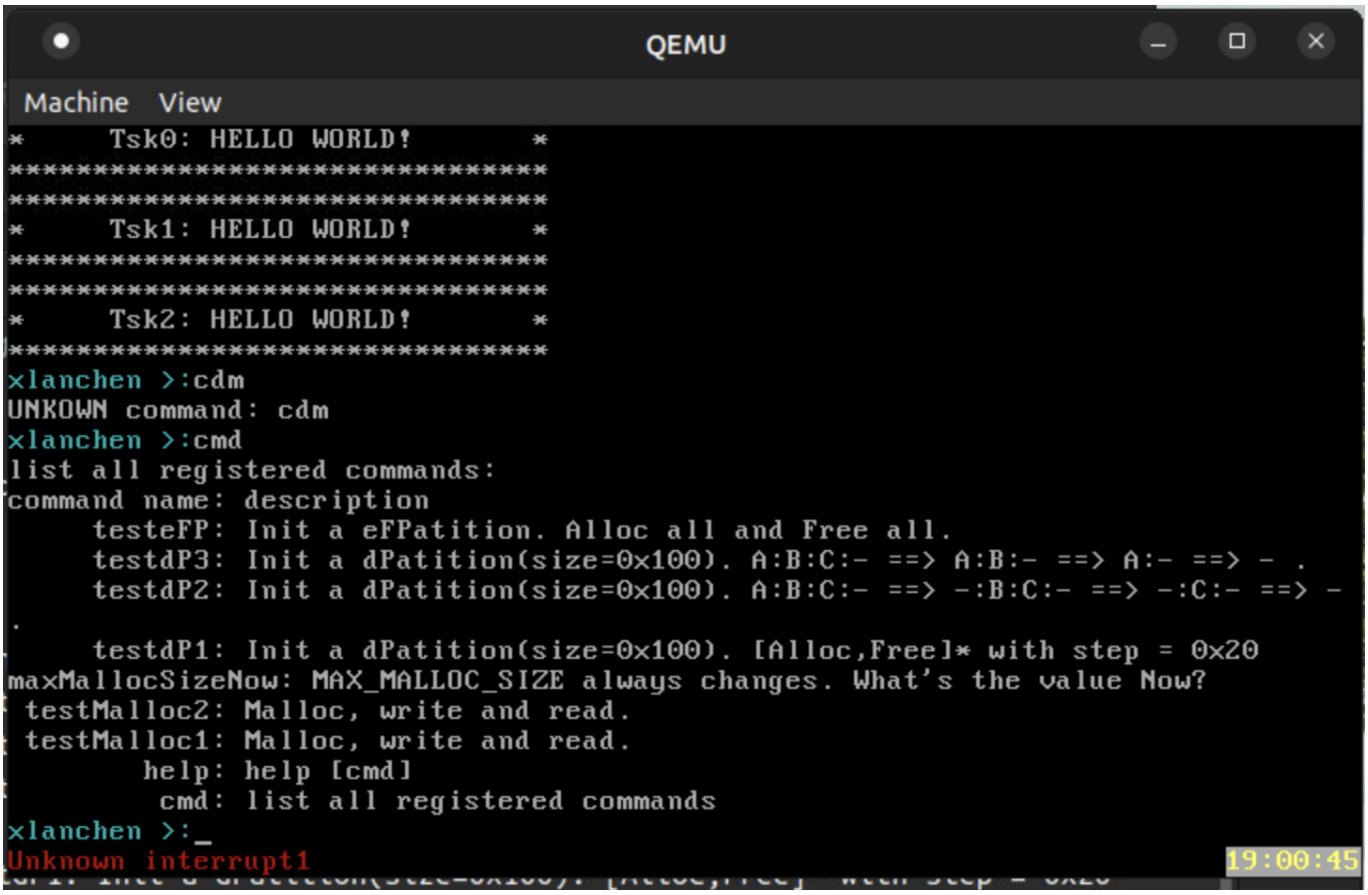
BspContextBase[STACK SIZE]

这个是系统栈,作用与上文的stack [stack size]相似.

prevTSK StackPtr是一级指针还是二级指针?为什么?

是一级指针,作为指向指针的指针,它指向了prevTsk->stkTop的地址，即指向了一个指针的指针,用于在上下文切换时保存前一个任务的栈指针。

实验结果


QEMU
Machine View
*   Tsk0: HELLO WORLD!   *
****
****
*   Tsk1: HELLO WORLD!   *
****
****
*   Tsk2: HELLO WORLD!   *
****
xlanchen >:cdm
UNKOWN command: cdm
xlanchen >:cmd
list all registered commands:
command name: description
testeFP: Init a eFPatition. Alloc all and Free all.
testdP3: Init a dPatition(size=0x100). A:B:C:- ==> A:B:- ==> A:- ==> - .
testdP2: Init a dPatition(size=0x100). A:B:C:- ==> -:B:C:- ==> -:C:- ==> -
.
testdP1: Init a dPatition(size=0x100). [Alloc,Free]* with step = 0x20
maxMallocSizeNow: MAX_MALLOC_SIZE always changes. What's the value Now?
testMalloc2: Malloc, write and read.
testMalloc1: Malloc, write and read.
help: help [cmd]
cmd: list all registered commands
xlanchen >:_
Unknown interrupt1
19:00:45