Operating System: Project 2 Document

Yao Class 18, Qianlan Yang, 2018011294, Maolong Yang, 2018011414

November 19, 2020

All of these tests should be executed under sh.coff.

1 Test1: Simple file system test

This test is set to check whether the project can open and close file correctly, and the maximum number of files it can open.

In this test, we try to open 20 files, and write something to it, then close them.

The result is that we can open 14 files simultaneously, since there are maximum 16 files and *stdin*, *stdout*, the result is correct.

2 Test 2: Simple file system test

This test aims to test the correctness of open and close files and the time of it.

In this test, we try to enumerate 1000 times, each time we choose a file, if it is opened, we will close it, while if it is closed, we will open it.

The result is what we expected, the opened file system is in cycle.

3 Test 3: Execute test

This test is to execute itself.

In $F(n)$, it will execute a $F(n-1)$ and let itself become $F(n-1)$ when $n > 0$.

In $n = 2$ case, it create 4 threads in total, one $F(2)$(origin), one $F(1)$ and two $F(0)$. But we can't run fully when $n = 3$ since we have no enough physical memory.

4 Test 4: Segment fault test

This test is to test when memory access is not in the correct position, what will happen.

We allocate an array of 20 integers, and we will access from -10 to 30.

In this test, we find that, if we try to write something, it will lead to crash, if we only try to read from them, there won't be error. This is because that virtual memory system cannot distinguish stack and section memory, so we will write on stack which leads to error.

5 Test 5: Unlink test

This test is to test create, close and unlink.

We create 14 files, then close and unlink them. We will repeat this some times.

The result is what we expected, all operations are successfully finished.

6 Test 6: Divide zero test

This test aims to test divide zero. The result is overflow error, I think that we cannot handle it.

7 Test 7: Large virtual address test

In this test we try to access a extremely huge virtual memory. The result is that, if the address exceeds the memory size, it will lead to page fault, and if the address fit in physical memory, our project will ignore it.

8 Test 8: Process test

Test 8 and test 9 is set to test process actions between actions.

In test 8, we have three programs, and we only need to use test8.coff to execute it. In the first program, it will exec the second process, and then join it. In the second process, it will first print something, then exec the third process, but it won't join the third process. The third process has a $while(1)$ so it will not end.

We run this program and get result that, it finishes process 2, then back to the shell controller. But if we print something in the $while(1)$ loop, we will find that process 3 is still running. This means, when we finished process 2, it left process 3 still running with the main shell controller at the same time, which follows the requirements.

9 Test 9: Process test 2

Test 9 is almost the same as test 8. At the end, we will join Process 3 instead end Process 1. But only the process's father can join it, so the behavior of this test should be the same as test 8.

10 Github repo

https://github.com/YMDragon/OperatingSystem