

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

РГЗ

Выполнил:
Студент группы: ИП-712
Алексеев С.В.

Проверил: профессор кафедры ПМиК
Фионов А.Н.

Новосибирск – 2020

Оглавление

Задание	2
1	2
Вывод.....	3
2	4
Вывод.....	4

Задание

1. Сравните время запуска (создания) нити и время активизации с помощью импульса заранее созданной нити.
2. Установите, какой максимальный объем памяти может предоставить процессу система.

1

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
#include <sys/neutrino.h>
#include <sched.h>
#include <rdtsc.h>
#include <time.h>
#include <sys/signinfo.h>
#define BILLION 1000000000L;

int chid = ChannelCreate(0);
int numstation = 0, mark = 0, numStation = 0, tm = 1, point = 0, secStart = 0, nsecStart = 0, secStop = 0, ns
struct timespec start, start2, stop, stop2;
//int coid = ConnectAttach(0,0,chid,0,0);
void* startRoutine(void*){ printf("thread has been started\n");}
void* startRoutinePulse(void*){
    int coid = ConnectAttach(0,0,chid,0,0);
    setprio(0, 0);
    sched_yield();
    double accum;
    printf("\nwe're at point 3d\n");
    MsgSend(coid, NULL, sizeof(int), &mark, sizeof(int));
    startRoutine(NULL);
    if( clock_gettime(CLOCK_REALTIME, &stop2) == -1) {perror("clock_gettime");}
    secStop = stop2.tv_sec;
    nsecStop = stop2.tv_nsec;
}

void testThreads(){
    pthread_t thr;
    if( clock_gettime(CLOCK_REALTIME, &start) == -1) {perror("clock_gettime");} // №1
    pthread_create(NULL, NULL, startRoutine, NULL);
    if( clock_gettime(CLOCK_REALTIME, &stop) == -1) {perror("clock_gettime");}
    double accum = (stop.tv_sec - start.tv_sec) + (double) (stop.tv_nsec - start.tv_nsec) / (double)BILLION;
    printf("time 1: %lfseconds, including %ld nanoseconds\n", accum, (stop.tv_nsec - start.tv_nsec));
    printf("start.tv_sec: %d\nstop.tv_sec: %d", start.tv_sec, stop.tv_sec);
    printf("\nstart.tv_nsec: %d\nstop.tv_nsec: %d\n", start.tv_nsec, stop.tv_nsec);
    pthread_create(NULL, NULL, startRoutinePulse, NULL);// №2
    printf("\nwe're at point 2b\n");
    //int rcvid = MsgReceive(chid, &point, sizeof(int), 0);
    //printf("\nwe're at point 2c\n");
    int coid = ConnectAttach(0,0,chid,0,0);
    printf("\nwe're at point 2d\n");
    if( clock_gettime(CLOCK_REALTIME, &start2) == -1) {perror("clock_gettime");}

    //MsgReply(rcvid, 0, &mark, sizeof(int));//has to be pulse. But works
    MsgSendPulse(coid, 94, 5, 33);
    sleep(1);
    secStart = start2.tv_sec;
    nsecStart = start2.tv_nsec;
    accum = (secStop - secStart) + (double) (nsecStop - nsecStart) / (double)BILLION;
    printf("time 2: %lfseconds, including %ld nanoseconds\n", accum, (nsecStop - nsecStart));
    printf(" secStart = %d\n secStop = %d\n nsecStart = %d\n nsecStop = %d\n", secStart, secStop, nsecStart
}

int main() {
    printf("\n");
    testThreads();
}
```

В TestThreads просто объявляю нить, засекаю время, вычитаю из конечного времени начальное. Получаю время создания нити.

Для подсчета времени, нужного на вызов нити сначала создаю нить с MsgSend, ожидающим импульса, потом только засекаю время и шлю импульс, пробуждающий нить с легковесным заданием внутри.

```
$ ./a.out

thread has been started
time 1: 0.090986seconds, including 90986
077 nanoseconds
start.tv_sec: 1610379137
stop.tv_sec: 1610379137
start.tv_nsec: 537748372
stop.tv_nsec: 628734449

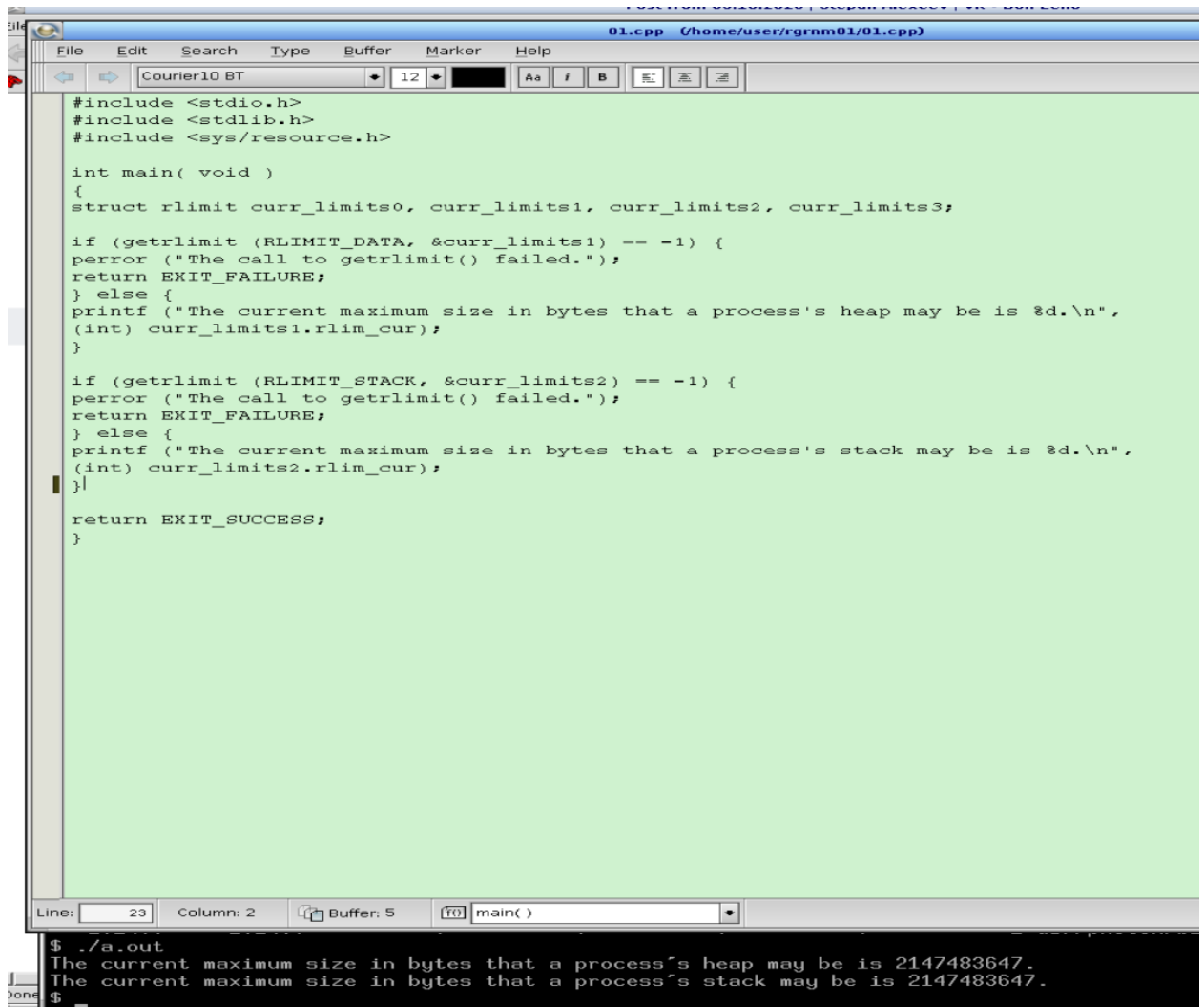
we're at point 2b

we're at point 2d

we're at point 3d
thread has been started
time 2: 0.003000seconds, including 29995
41 nanoseconds
secStart = 1610379137
secStop = 1610379137
nsecStart = 738717619
nsecStop = 741717160
$ ./a.out_
```

Вывод: запуск новой нити примерно в 30 раз медленнее активизации с помощью импульса заранее созданной нити.

2



The screenshot shows a C++ IDE window titled "01.cpp (/home/user/rgrnm01/01.cpp)". The code defines a `main` function that uses `getrlimit` to check the current limits for the heap (`RLIMIT_DATA`) and stack (`RLIMIT_STACK`). It prints the current maximum size in bytes for both. The status bar at the bottom indicates "Line: 23", "Column: 2", "Buffer: 5", and the current function is `main()`.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/resource.h>

int main( void )
{
    struct rlimit curr_limits0, curr_limits1, curr_limits2, curr_limits3;

    if (getrlimit (RLIMIT_DATA, &curr_limits1) == -1) {
        perror ("The call to getrlimit() failed.");
        return EXIT_FAILURE;
    } else {
        printf ("The current maximum size in bytes that a process's heap may be is %d.\n",
            (int) curr_limits1.rlim_cur);
    }

    if (getrlimit (RLIMIT_STACK, &curr_limits2) == -1) {
        perror ("The call to getrlimit() failed.");
        return EXIT_FAILURE;
    } else {
        printf ("The current maximum size in bytes that a process's stack may be is %d.\n",
            (int) curr_limits2.rlim_cur);
    }

    return EXIT_SUCCESS;
}
```

Terminal output:

```
$ ./a.out
The current maximum size in bytes that a process's heap may be is 2147483647.
The current maximum size in bytes that a process's stack may be is 2147483647.
$
```

Вывод

Максимальный объём кучи и стека для одного процесса: 2Гб.