

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №41

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

<u>Доцент, к. т. н.</u>	<u></u>	<u>О. А. Кононов</u>
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

УСЛОВНЫЕ ПЕРЕМЕННЫЕ

по дисциплине: 'СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ'

РАБОТУ ВЫПОЛНИЛ

<u>СТУДЕНТ ГР.</u>	<u>1742</u>	<u></u>	<u>Д. В. Коробков</u>
	номер группы	подпись, дата	инициалы, фамилия

Санкт-Петербург 2020

1. Постановка задачи

Ознакомиться с новым механизмом синхронизации работы потоков – условными переменными, применив его на практике.

2. Листинг программы

```
#include <stdio.h>
#include <time.h>
#include <sync.h>
#include <sys/neutrino.h>
#include <pthread.h>
int data_ready = 0;
int inf = 0;
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t condvar = PTHREAD_COND_INITIALIZER;
void *consumer (void * notused) {
    char buf [27];
    time_t now;
    time(&now);
    printf("Eto potrebite vremia starta %s \n", ctime_r (&now, buf));
    while(1){
        pthread_mutex_lock (&mutex);
        printf("W1\n");
        while (!data_ready){
            printf(" W2\n");
            pthread_cond_wait (&condvar, & mutex);
            printf(" W3\n");
        }
        time(&now);
        printf(" dannie ot proizv = %d %s \n",inf, ctime_r (&now, buf));
        data_ready=0;
        pthread_cond_signal(&condvar);
        pthread_mutex_unlock(&mutex);}}
void *producer (void * notused){
    char buf [27];
    time_t now;
    time(&now);
    printf("Eto proizvodite %s \n", ctime_r (&now, buf));
    while(1){
        sleep(2);
        time(&now);
        printf(" proizvoditel polychil dannie ot h/w = %d %s \n",inf, ctime_r (&now, buf));
        pthread_mutex_lock (&mutex);
        printf("Wp1\n");
        while (data_ready){
            printf("Wp2\n");
            pthread_cond_wait (&condvar, & mutex);
        }
        data_ready=1;
        inf++;
        printf("Wp3\n");
        pthread_cond_signal(&condvar);
        pthread_mutex_unlock(&mutex);}}
main(){
    time_t now;
    char buf [27];
    time(&now);
    printf("---Start--- %s \n", ctime_r (&now, buf));
    pthread_create(NULL,NULL, consumer,NULL);
    pthread_create(NULL,NULL, producer,NULL);
    sleep(10);
    time(&now);
    printf("---END--- %s \n", ctime_r (&now, buf));
}
```

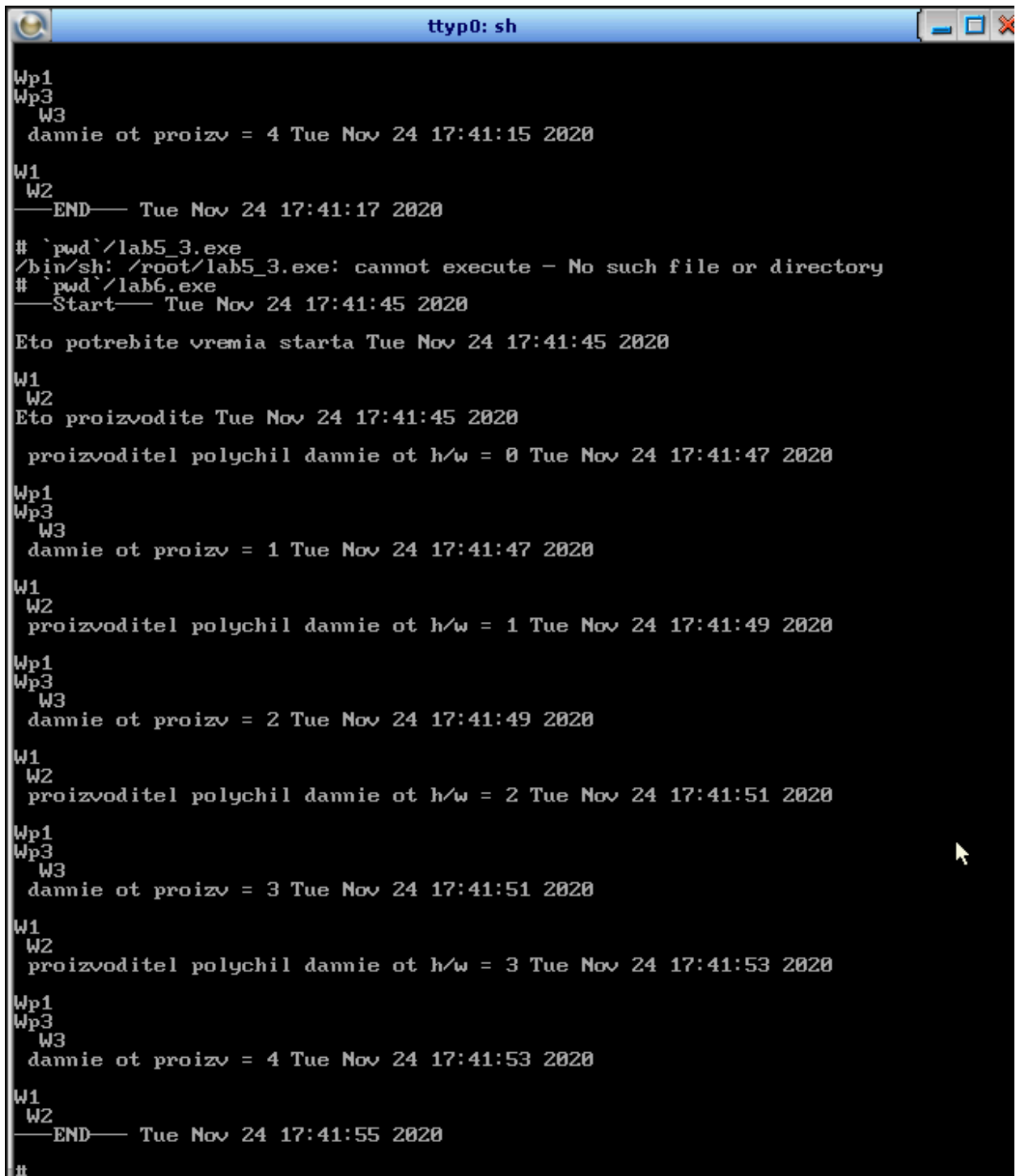
Рисунок 1 – Текст программы.

3. Краткие теоретические сведения

Условные переменные очень похожи на ждущие блокировки тем, так как вторые являются надстройкой для первой.

Функция `pthread_cond_wait()` освобождает мьютекс, а затем повторно блокирует мьютекс, аналогично функции `pthread_sleepon_wait()`.

4. Результаты работы программы



```
Wp1
Wp3
W3
dannie ot proizv = 4 Tue Nov 24 17:41:15 2020
W1
W2
---END--- Tue Nov 24 17:41:17 2020
# `pwd`/lab5_3.exe
/bin/sh: /root/lab5_3.exe: cannot execute - No such file or directory
# `pwd`/lab6.exe
---Start--- Tue Nov 24 17:41:45 2020
Eto potrebite vremia starta Tue Nov 24 17:41:45 2020
W1
W2
Eto proizvodite Tue Nov 24 17:41:45 2020
proizvoditel polychil dannie ot h/w = 0 Tue Nov 24 17:41:47 2020
Wp1
Wp3
W3
dannie ot proizv = 1 Tue Nov 24 17:41:47 2020
W1
W2
proizvoditel polychil dannie ot h/w = 1 Tue Nov 24 17:41:49 2020
Wp1
Wp3
W3
dannie ot proizv = 2 Tue Nov 24 17:41:49 2020
W1
W2
proizvoditel polychil dannie ot h/w = 2 Tue Nov 24 17:41:51 2020
Wp1
Wp3
W3
dannie ot proizv = 3 Tue Nov 24 17:41:51 2020
W1
W2
proizvoditel polychil dannie ot h/w = 3 Tue Nov 24 17:41:53 2020
Wp1
Wp3
W3
dannie ot proizv = 4 Tue Nov 24 17:41:53 2020
W1
W2
---END--- Tue Nov 24 17:41:55 2020
# _
```

Рисунок 2 – Результаты работы программы.

5. Временная диаграмма

На рисунке 3 изображена временная диаграмма для главного и двух созданных потоков. На диаграмме мы видим, что поток `main` в позициях 0, 2 и 4 сообщает нам о старте потока `main`, `consumer` и `producer`, соответственно. А после поток `main` блокируется на 10 секунд, дабы дать возможность взаимодействовать друг с другом двум потокам `consumer` и `producer`. В первом заходе в цикле видно, что значение переменной `inf` не меняется, так как изначально присутствует задержка `sleep(2)`, поэтому данные считываются до их изменения. Также дополнительно на диаграмме изображен порядок работы потоков и значение переменной `inf` для потоков `consumer` и `producer`.

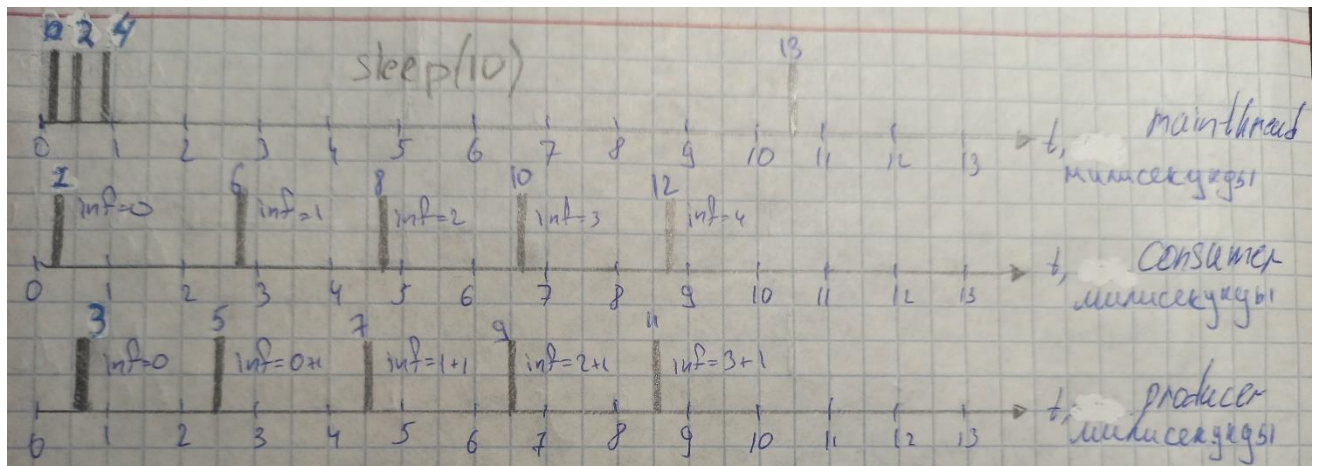


Рисунок 3 – Временная диаграмма работы программы.

6. Вывод

В ходе выполнения лабораторной работы была изучена программа с использованием мьютекса и условной переменной. Этот вид переменных позволяет контролировать блокировку мьютекса при работе с данными, когда к ним требуется доступ от нескольких потоков. Это делается для безопасности данных и избегания ошибочных ситуаций, когда один поток записывает значение в переменную, а другой поток считывает данные из нее в этот же момент. Тем самым считанные данные будут уже устаревшими. Благодаря мьютексам и условным переменным можно синхронизировать работу потоков, использующих общие ресурсы.