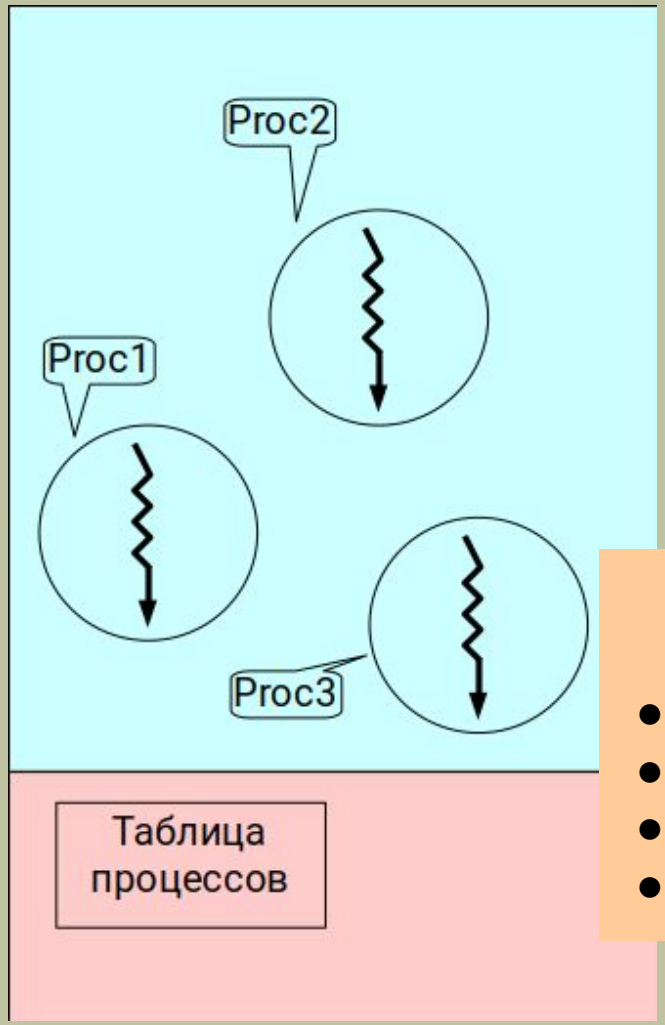
Лекция 8

- Потоки выполнения: применение - многопоточное программирование, модель потоков, объект ядра.
- Реализация потоков: POSIX, C++11.

Модель	Программные средства	Архитектура ВС
Общая память	POSIX (pthread), WinAPI(CreateThread), OpenMP...	MIMD, разделяемая память
Обмен сообщениями	MPI (Message Passing Interface): OpenMPI, MPICH, LAM (Local Area Multivomputer); PVM (Parallel Virtual Machine)...	MIMD, распределенная и разделяемая память
Параллелизм данных	Языки .NET, Python...	MIMD/SIMD

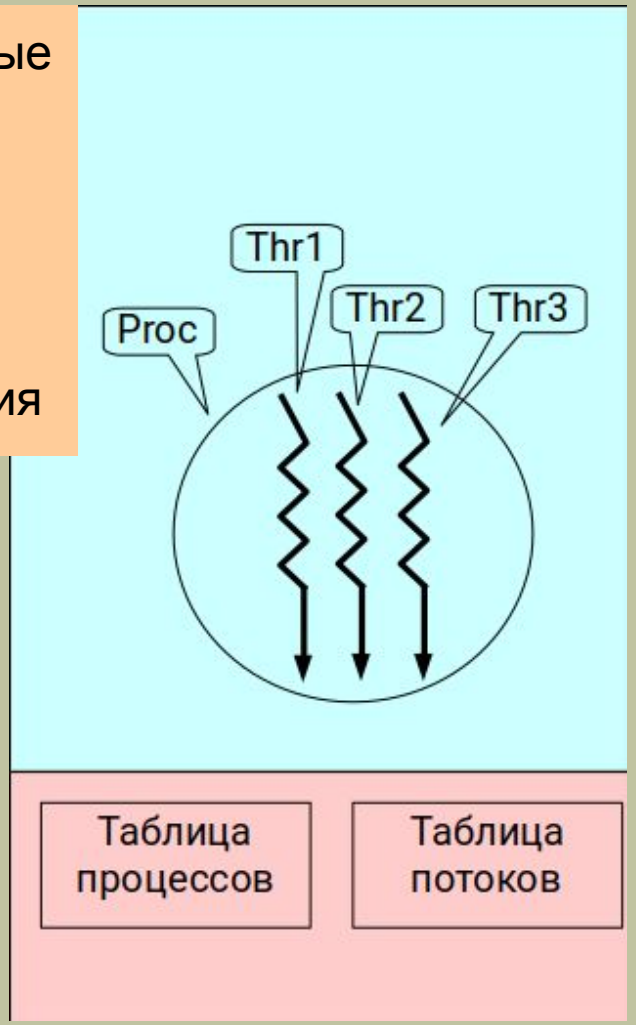
Совместно используемые ресурсы

- Адресное пр-во
- Глобальные переменные
- Открытые файлы
- Учетная информация

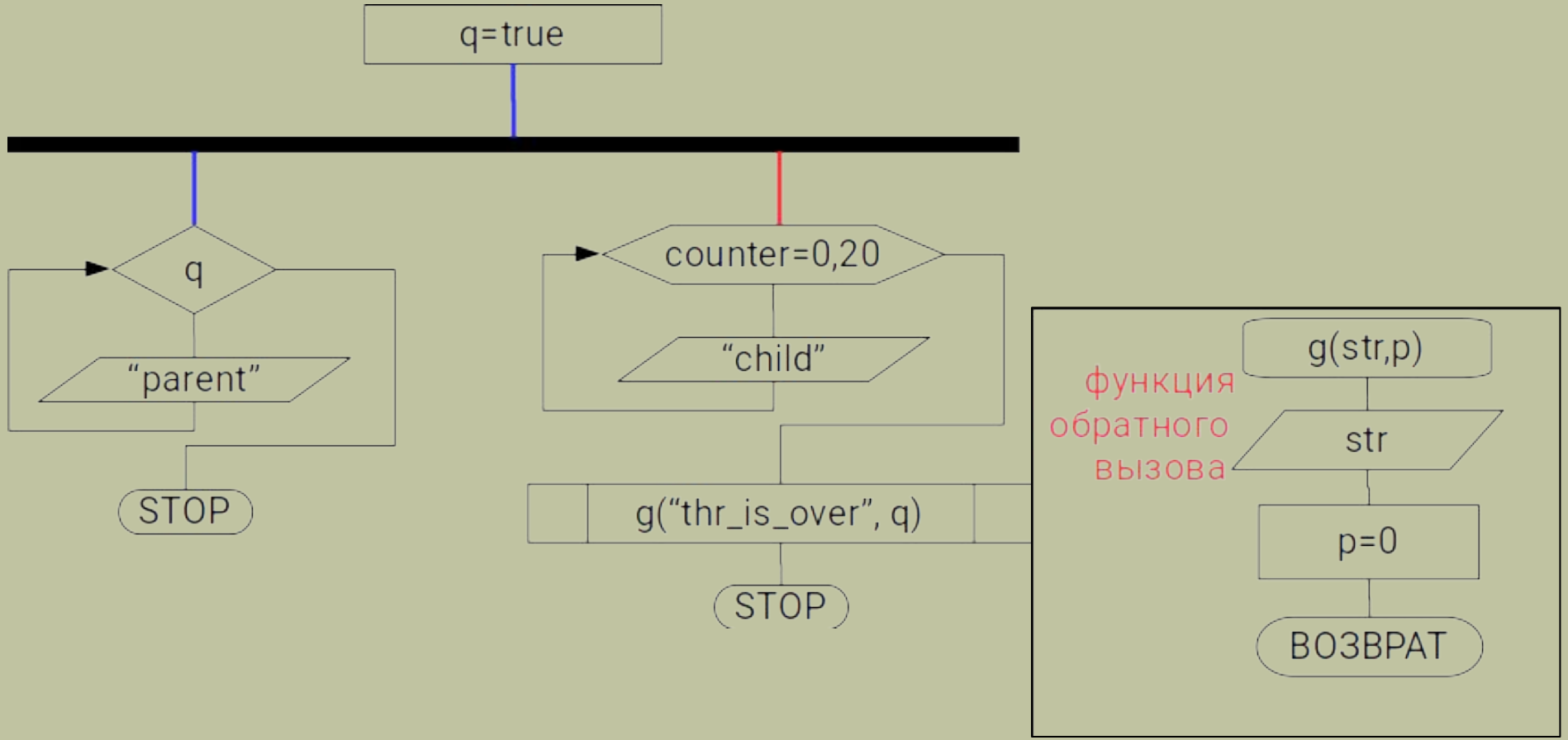


Индивидуальные элементы потоков

- Счётчик команд
- Регистры
- Стек
- Состояние



Многопоточная программа



Пример: интерфейс POSIX Threads (Pthreads)

```c
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
typedef int (*fun)(char*, int*);
int q=1;

int g(char* str, int* p){
*p=0;
printf("%s\t%d\n", str, q);
return 0;
}
```

```
void* my_thread(void*);

int main(){
pthread_t  th_id;

pthread_create(&th_id, NULL, &my_thread, (void*)g);
while(q){
 printf("parent\t%d\n",q);
 sleep(1);
}

return 0;
}
```

```
void* my_thread(void* pg){
int counter=0;
while(q){
 printf("child\n");
 sleep(1);
 if(counter++>5)
         break;
}
((fun)pg)("thread_is_over!\n", &q);
}
```

gcc lab8a.c -lpthread -o lab8a

Пример: интерфейс C++11 <thread>

```
#include <stdio.h>
#include <thread>
#include <unistd.h>
typedef int (*fun)(char*);

int g(char* str){
printf("%s\n", str);
return 0;
}
```

```
void my_thread(void*);

int main(){
std::thread  th(my_thread, (void*)g);
printf("thread's been created\n");

th.join();
printf("parent\n");

return 0;
}
```

```
void my_thread(void* pg){
int counter=0;
while(1){
 printf("child\n");
 sleep(1);
 if(++counter>5)
         break;
}
((fun)pg)((char*)"thread_is_over!\n");
}
```

Барьерная синхронизация

```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
int sh=0;
int flag[2]={1,1};
void*  my_thread0( ){
int i=0;
for(;i<5;i++, sh++)sleep(1);
flag[0]=0;
}
void*  my_thread1(){
int i=0;
for(;i<5;i++,sh+=2)sleep(1);
flag[1]=0;
}
```

```c
int main(){
 pthread_t th_id[2];

 pthread_create(&th_id[0], NULL, &my_thread0, NULL);
 pthread_create(&th_id[1], NULL, &my_thread1, NULL);

 while(flag[0] || flag[1]);

 printf("%i\n",sh);
 return 0;
}
```

```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
int sh=0;
pthread_barrier_t  barrier;
void*  my_thread0( ){
int i=0;
for(;i<5;i++, sh++)sleep(1);
pthread_barrier_wait(&barrier);
}
void*  my_thread1(){
int i=0;
for(;i<5;i++,sh+=2)sleep(1);
pthread_barrier_wait(&barrier);
}
```

```c
int main(){
 pthread_t th_id[2];
 pthread_barrier_init(&barrier, NULL, 3);
 pthread_create(&th_id[0], NULL, &my_thread0, NULL);
 pthread_create(&th_id[1], NULL, &my_thread1, NULL);

 pthread_barrier_wait(&barrier);

 printf("%i\n",sh);

 pthread_barrier_destroy(&barrier);
 return 0;
}
```