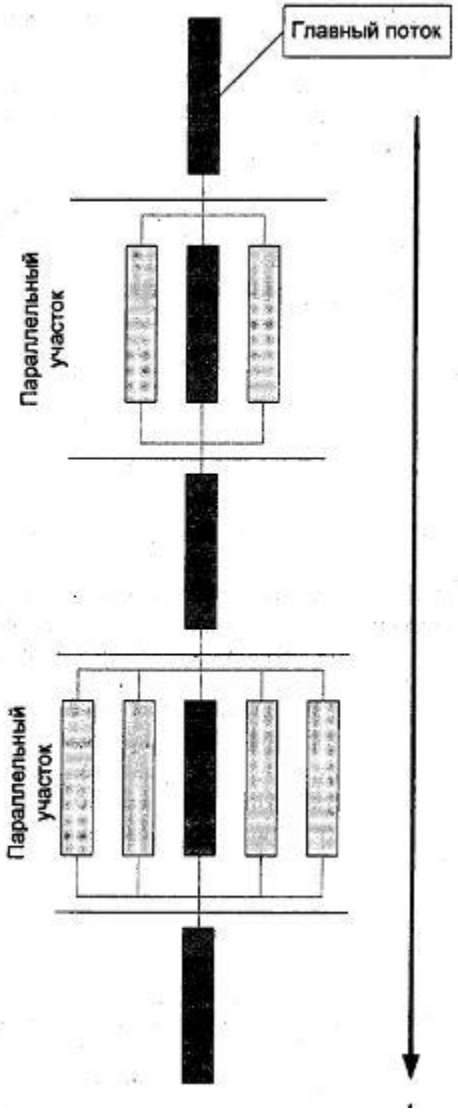
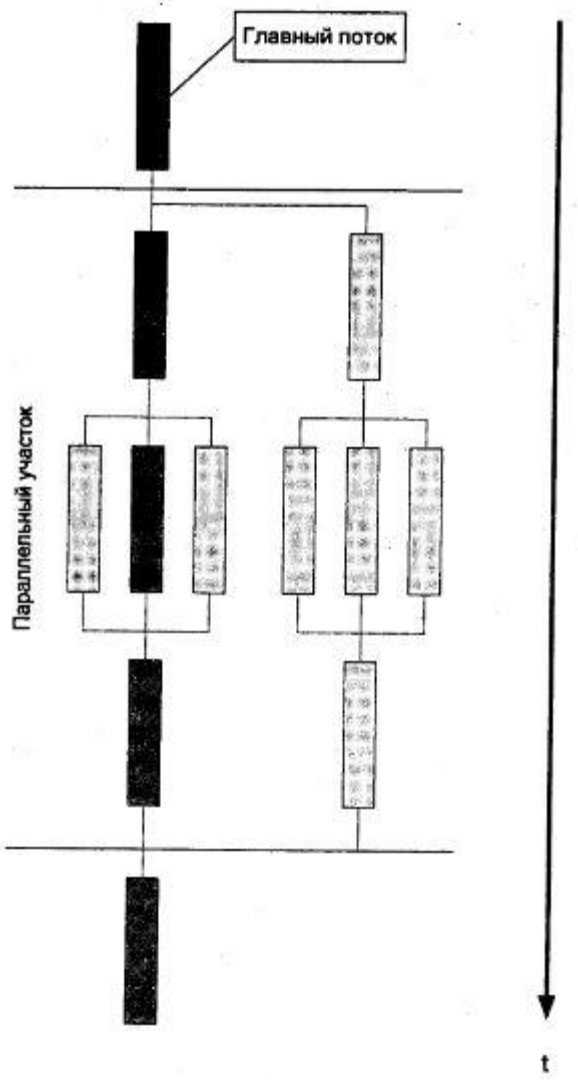
ЛЕКЦИЯ 1

Среда программирования
OpenMP


Главный поток
Параллельный участок
Параллельный участок


Главный поток
Параллельный участок
t

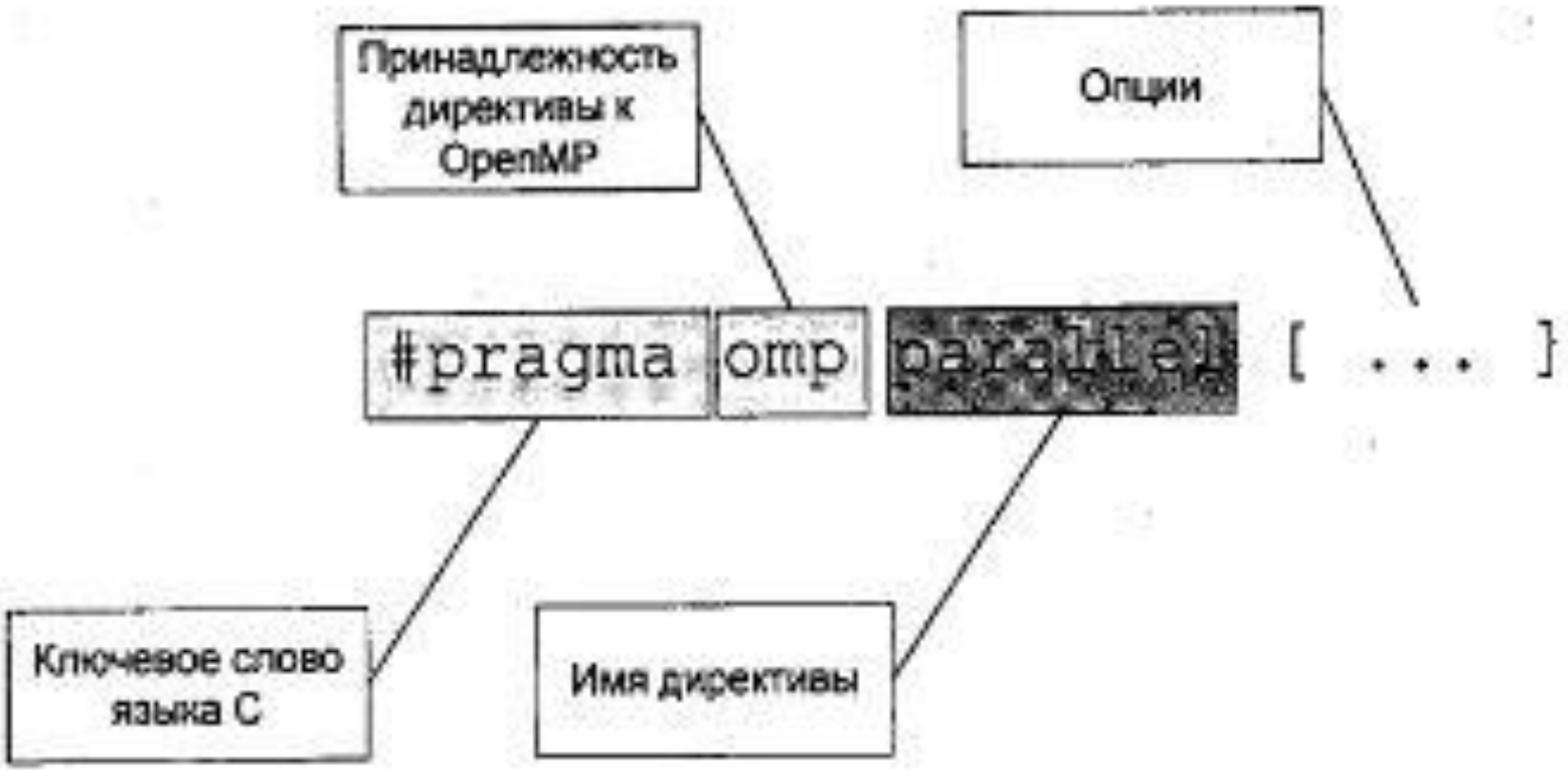
Модель памяти

Переменные в OpenMP-программе делятся на *общие* (shared) и *индивидуальные* (private). Индивидуальные переменные соответствуют некоторому потоку и могут считываться или записываться только им. Общие переменные доступны для чтения и записи нескольким потокам одной группы. Следует отметить, что если чтение и запись или повторная запись общей переменной производятся без синхронизации, то результирующее значение переменной считается неопределенным.

ПРОСТЕЙШАЯ ПРОГРАММА

```
1:  #include <stdio.h>
2:  #include <omp.h>
3:  main(){
4:  #pragma omp parallel
5:    printf("Hello World!\n");
6:  }
```

Синтаксис директивы



Изменение числа потоков

```
1:  main(int argc, char* argv[])
2:  {
3:    int n;
4:    n = atoi(argv[1]);
5:  #pragma omp parallel num_threads(n)
6:    printf("Hello World\n");
7:  }
```

Функции числа потоков

Получение количества процессоров, доступных для выполнения OpenMP-приложения в момент вызова:

```
int omp_get_num_procs(void)
```

Получение количества потоков в группе:

```
int omp_get_num_threads(void)
```

Получение номера потока в группе:

```
int omp get thread num(void)
```

```
1:  main()
2:  {
3:    int n;
4:    n = omp_get_num_procs();
5:    printf("%d processors available\n", n);
6:  #pragma omp parallel num_threads(n)
7:    printf("Hello. I'm thread %d from %d.\n",
       omp_get_thread_num(),
       omp_get_num_threads());
8:  }
```

Листинг 35. Пример изменения числа выполняемых потоков с помощью опции `num_threads`

На двухпроцессорной системе данная программа выведет на печать следующий текст:

```
2 processors available
Hello. I'm thread 0 from 2.
Hello. I'm thread 1 from 2.
```

```
1:  extern int A;
2:  void f(int c)
3:  {
4:    static double z;
5:    int x;
6:  }
7:
8:  main(){
9:    double y;
10:   #pragma omp parallel
11:   {
12:     int a;
13:     f(5);
14:   }
15: }
```

По отношению к параллельному участку следующие переменные являются общими:

A, z, y

остальные переменные:

a, c, x

являются индивидуальными.

Листинг 36. Правила по умолчанию для переменных в OpenMP

Опция `private` определяет список переменных, которые будут индивидуальными для потоков, выполняющих параллельный участок. При этом не задается, каким именно образом производится инициализация значения переменных на потоках. Также неопределенным будет значение переменной на главном потоке после завершения параллельного участка.

Опция `firstprivate` задает способ инициализации индивидуальных переменных: переменные, перечисленные в списке аргументов этой области, получают значение, равное значению переменной на главном потоке в момент входа в параллельный участок. Таким образом, опция `firstprivate` предоставляет всю функциональность опции `private`, добавляя к ней способ инициализации переменных.

Опция `reduction` определяет значение переменных, входящих в список ее аргументов, на главном потоке после завершения параллельного участка как результат выполнения редуктивной операции. На каждом из потоков, выполняющих параллельный участок, переменная получает значение, соответствующее редуктивной операции:

Опция редукции значений переменных на потоках:

```
reduction(operator:list)
```

`list` – список идентификаторов
`operator` – одна из следующих редуктивных операций:

Операция	Значение для инициализации
+	0
*	1
-	0
&	~0
|	0
^	0
&&	1
||	0

```
1:  #include <stdio.h>
2:  #include <omp.h>
3:  main(int argc, char* argv[])
4:  {
5:    int a, b, t;
6:    a = atoi(argv[1]);
7:    b = atoi(argv[2]);
8:    t = 0;
9:    #pragma omp parallel firstprivate(a)
10:     reduction(+ : t) num_threads(b)
11:   {
12:     t = a;
13:   }
14:   printf("a x b = %d\n", t);
15: }
```

Листинг 37. Применение опции `reduction`

ЛИТЕРАТУРА

Антонов А.С.
Параллельное программирование с использованием технологии OpenMP: Учебное пособие. – М.: Изд-во МГУ, 2009. – 77 с.

ISBN 978-5-211-05702-9

Учебное пособие предназначено для освоения практического курса параллельного программирования с использованием технологии OpenMP. В настоящее время технология OpenMP является основным средством программирования для компьютеров с общей памятью. Книга включает в себя описание большинства основных директив, функций и переменных окружения стандарта OpenMP 3.0 с примерами их применения, а также практические сведения, которые могут потребоваться при написании реальных программ. Некоторые детали описания стандарта опускаются для простоты изложения и восприятия материала. Описание ведётся с использованием вызовов процедур OpenMP из программ на языках Си и Фортран. Приводятся примеры небольших законченных параллельных программ, тексты которых доступны в сети Интернет на странице http://parallel.ru/tech/tech_dev/OpenMP/examples/. В конце разделов приводятся контрольные вопросы и задания, которые можно использовать в процессе обучения.

Для студентов, аспирантов и научных сотрудников, чья деятельность связана с параллельными вычислениями.