Національний технічний університет України
«Київський політехнічний інститут»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №1

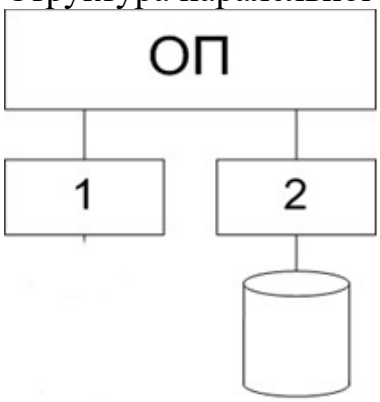
З предмету: «Паралельне програмування - 2»

Виконав: студент групи ІП-32
Ковальчук Олександр Миронович

Київ 2016р.

Технічне завдання

1. Структура паралельної комп’ютерної системи з спільною пам’яттю:



2. Задача: $A = B(MO \times MK) + \alpha \cdot E$
3. Мова програмування:
   Ада.
4. Засіб взаємодії задач:
   семафори.

Виконання завдання.

Етап 1. Побудова паралельного алгоритму

$$A_H = B(MO \times MK_H) + \alpha \cdot E_H$$

Спільні ресурси: α, МО, В.

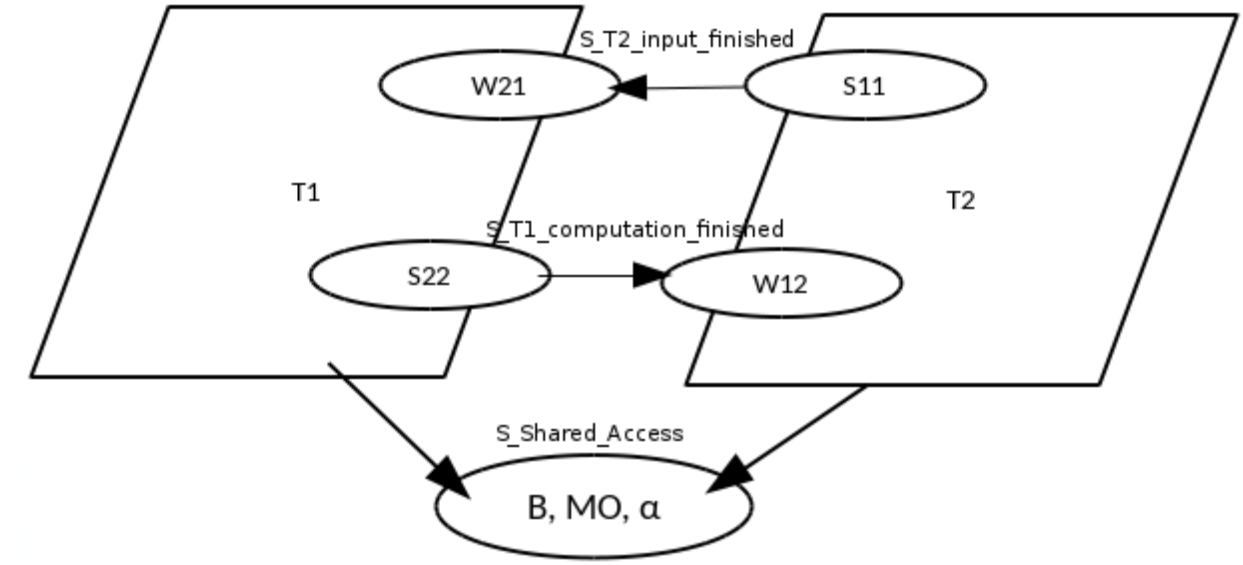
Етап 2. Розробка алгоритму процесів (задач)

№	Задача Т1	ТС / КД
1	Очікувати сингнал від Т2 про введення даних	W_{21}
2	Копіювати: • $MO_1 := MO$ • $\alpha_1 := \alpha$ • $B_1 := B$	КД
3	Обчислити $A_H = B(MO \times MK_H) + \alpha \cdot E_H$	
4	Сигнал Т2 про завершення обчислень	S_{22}

№	Задача Т2	ТС / КД
1	Введення В, МО, МК, α, Е	
2	Надіслати сигнал Т1 про закінчення вводу	S_{12}

	даних	
3	Копіювати: • MO_2:= MO • α_2:= α • B_2:= B	КД
4	Обчислення: $A_H = B(MO \times MK_H) + \alpha \cdot E_H$	
5	Чекати сигнал від T1 про завершення обчислень	W_{22}
6	Вивести A	

Етап 3. Розроблення структурної взаємодії задач



Етап 4. Розроблення програми

```
GNAT 5.3.1 20151207 (Red Hat 5.3.1-2)
Copyright 1992-2015, Free Software Foundation, Inc.

Compiling: lab1.adb
Source file time stamp: 2016-03-02 00:46:13
Compiled at: 2016-03-02 09:34:22

     1. -------------------------------------
     2. --      Parallel programming      --
     3. --              Lab 1             --
     4. --         Ada. Semaphores        --
     5. --                                --
     6. -- Task: A = B(MO*MK) + alphaE    --
     7. -- @author Olexandr Kovalchuk     --
     8. -- @group IP 32                   --
     9. --                                --
    10. -- @date 2016-02-24               --
    11. -------------------------------------
    12. with Data; use Data;
```

```
13. with Ada.Text_IO; use Ada.Text_IO;
14. with Ada.Synchronous_Task_Control; use Ada.Synchronous_Task_Control;
15.
16.
17. procedure Lab1 is
18.     A, B, E: Vector;
19.     MO, MK: Matrix;
20.     alpha: Integer;
21.
22.     -- These semaphores are used to signalize about events
23.     S_T2_input_finished: Suspension_Object;
24.     S_T1_computation_finished: Suspension_Object;
25.
26.     -- Semaphores to controll access to shared variables
27.     S_Shared_Access: Suspension_Object;
28.
29.
30.     procedure RunTasks is
31.
32.         task T1 is
33.             pragma Storage_Size (300_000_000);
34.         end T1;
35.         task body T1 is
36.             -- Copies of shared data
37.             MO_1: Matrix;
38.             B_1: Vector;
39.             alpha_1: Integer;
40.         begin
41.             Put_Line("Task T1 started");
42.
43.             -- Wait until T2 inputs data
44.             Suspend_Until_True(S_T2_input_finished);
45.
46.             -- Copy shared objects
47.             Suspend_Until_True(S_Shared_Access);
48.             alpha_1 := alpha;
49.             B_1 := B;
50.             MO_1 := MO;
51.             Set_True(S_Shared_Access);
52.
53.             -- Compute partial result
54.             for i in 1 .. H loop
55.                 A(i) := 0;
56.                 for j in 1 .. N loop
57.                     for k in 1 .. N loop
58.                         A(i) := A(i) + B_1(j) * MO_1(j,k) * MK(k,i);
59.                     end loop;
60.                 end loop;
61.                 A(i) := A(i) + alpha_1 * E(i);
62.             end loop;
63.
64.             -- Signal about T1 finished its computations
65.             Set_True(S_T1_computation_finished);
66.
67.             Put_Line("Task T1 finished");
68.         end T1;
69.
70.         task T2 is
71.             pragma Storage_Size (300_000_000);
72.         end T2;
73.         task body T2 is
74.             MO_2: Matrix;
75.             B_2: Vector;
76.             alpha_2: Integer;
77.         begin
78.             Put_Line("Task T2 started");
79.
80.             -- Input data
81.             GetVector(B);
82.             GetVector(E);
83.             GetMatrix(MO);
```

```
    84.             GetMatrix(MK);
    85.             alpha := 1;
    86.
    87.             -- Signal about data input has finished
    88.             Set_True(S_T2_input_finished);
    89.
    90.             -- Copy shared objects
    91.             Suspend_Until_True(S_Shared_Access);
    92.             alpha_2 := alpha;
    93.             B_2 := B;
    94.             MO_2 := MO;
    95.             Set_True(S_Shared_Access);
    96.
    97.             -- Compute partial result
    98.             for i in H + 1 .. N loop
    99.                 A(i) := 0;
   100.                 for j in 1 .. N loop
   101.                     for k in 1 .. N loop
   102.                         A(i) := A(i) + B_2(j) * MO_2(j,k) * MK(k,i);
   103.                     end loop;
   104.                 end loop;
   105.                 A(i) := A(i) + alpha_2 * E(i);
   106.             end loop;
   107.
   108.             -- Wait until T1 finishes computations
   109.             Suspend_Until_True(S_T1_computation_finished);
   110.
   111.             -- Output result
   112.             if (N < 8) then
   113.                 New_Line;
   114.                 Put("Result: ");
   115.                 Put(A);
   116.                 New_Line;
   117.                 New_Line;
   118.             end if;
   119.
   120.             Put_Line("Task T2 finished");
   121.         end T2;
   122.
   123.     begin
   124.         null;
   125.     end RunTasks;
   126. begin
   127.     Put_Line("Lab1 started");
   128.
   129.     -- Prepare semaphores:
   130.     Set_True(S_Shared_Access);
   131.
   132.     RunTasks;
   133.     Put_Line("Lab1 finished");
   134. end Lab1;
   135.

 135 lines: No errors

GNAT 5.3.1 20151207 (Red Hat 5.3.1-2)
Copyright 1992-2015, Free Software Foundation, Inc.

Compiling: data.adb
Source file time stamp: 2016-03-01 21:58:59
Compiled at: 2016-03-02 09:40:04

     1. with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;
     2. with Ada.Text_IO; use Ada.Text_IO;
     3.
     4. package body Data is
     5.
     6.   procedure Fill(filler: in Integer; mtrx: out Matrix) is
     7.   begin
     8.     for r in mtrx'range(1) loop
     9.       for c in mtrx'range(2) loop
```

```
    10.         mtrx(r, c) := filler;
    11.       end loop;
    12.     end loop;
    13.   end;
    14.
    15.   procedure Fill(filler: in Integer; vec: out Vector) is
    16.   begin
    17.     for i in vec'range(1) loop
    18.       vec(i) := filler;
    19.     end loop;
    20.   end;
    21.
    22.   procedure GetMatrix(mtrx: out Matrix) is
    23.   begin
    24.     Fill(1, mtrx);
    25.   end;
    26.
    27.   procedure GetVector(vec: out Vector) is
    28.   begin
    29.     Fill(1, vec);
    30.   end;
    31.
    32.   procedure Put(value: in Matrix) is
    33.   begin
    34.     for row in value'range(1) loop
    35.       for col in value'range(2) loop
    36.         Put(value(row, col), 4);
    37.         Put(',');
    38.       end loop;
    39.       New_Line;
    40.     end loop;
    41.   end;
    42.
    43.   procedure Put(value: in Vector) is
    44.   begin
    45.     for i in value'range(1) loop
    46.       Put(value(i), 4);
    47.       Put(',');
    48.     end loop;
    49.   end;
    50.
    51. end Data;
    52.

Compiling: data.ads
Source file time stamp: 2016-03-02 00:44:40
Compiled at: 2016-03-02 09:40:04

     1. package Data is
     2.
     3.     -- Size of vector or matrix dimension
     4.     N : Integer := 4;
     5.     -- Size of half
     6.     H : Integer := N / 2;
     7.
     8.     type Vector is array (1 .. N) of Integer;
     9.     type Matrix is array (1 .. N, 1 .. N) of Integer;
    10.
    11.     procedure GetMatrix(mtrx: out Matrix);
    12.     procedure GetVector(vec: out Vector);
    13.
    14.     procedure Put(value: in Matrix);
    15.     procedure Put(value: in Vector);
    16. end Data;
    17.

 52 lines: No errors
```