Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №1**

**з дисципліни «Програмування паралельних комп’ютерних систем»**

**на тему «Ада. Семафори»**

Виконав:

студент ІІI курсу

групи ІО-53

Лісовий В. О.

Перевірив:

доц. Корочкін О. В.

Київ – 2018

**Тема**

«Програмування для комп’ютерних систем зі спільною пам’яттю. Ада. Семафори»

**Технічне завдання**

Розробити програму для розв’язання ПКС із СП (структура на рис. 1) математичної задачі: A = (B + C \* MO) \* d + Z \* (MT \* MH).

Мова програмування: Ада.

Засоби взаємодії: семафори.

ОП

1

2

1

2

B, C, MO, d, Z, MT, MH

Рис. 1 Структурна схема ПКС

**Виконання роботи**

**Етап 1. Побудова паралельного алгоритму**

A­H = (BH + C \* MOH) \* d + Z \* (MT \* MHH).

Спільний ресурс: C, d, Z, MT.

**Етап 2. Розроблення алгоритмів роботи кожного процесу**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Задача Т1 |  | Задача Т2 |  |
| 1 | Чекати завершення вводу даних в Т2 | W2,1 | Ввід даних (B, C, MO, d, Z, MT, MH) |  |
| 2 | Копіювати d1=d, C1=C, Z1=Z, MT1=MT | КД | Сигнал Т1 про завершення вводу даних | S1,1 |
| 3 | Обчислення  A­H = (BH + C1 \* MOH) \* d1 + Z1 \* (MT1 \* MHH). |  | Копіювати d2=d, C2=C, Z2=Z, MT2=MT | КД |
| 4 | Чекати завершення обчислень в Т2 | W2,2 | Обчислення  A­H = (BH + C2 \* MOH) \* d2 + Z2 \* (MT2 \* MHH). |  |
| 5 | Виведення результату |  | Сигнал Т1 про завершення обчислень | S1,2 |

**Етап 3. Розроблення структурної схеми взаємодії задач**

На структурній схемі взаємодії задач уведено такі семафори

* Skd1 – для керування доступом до спільних ресурсів
* S1 – синхронізація з завершенням введення в Т2
* S2 – синхронізація з завершенням обчислень в T2

ЕT1

T2

Skd1

S1

S2

Рис. 2 Структурна схема взаємодії задач

**Етап 4. Розробка програми**

1.

2. -------------------------------------

3. --Parallel and distributed computing.

4. --Lab 1. Ada Semafore

5. --Lisovyi Volodymyr

6. --IO-53

7. --13.03.18

8. --A = (B + C \* MO) \* d + Z \* (MT \* MH)

9. ---------------------------------------

10.

11. with Ada.Text\_IO, Ada.Integer\_Text\_IO, System.Multiprocessors;

12. use Ada.Text\_IO, Ada.Integer\_Text\_IO, System.Multiprocessors;

13. with Ada.Text\_IO, Ada.Integer\_Text\_IO, Ada.Synchronous\_Task\_Control;

14. use Ada.Text\_IO, Ada.Integer\_Text\_IO, Ada.Synchronous\_Task\_Control;

15. with Data;

16.

17. procedure Main is

18.

19. CPU1 : CPU\_Range := 0;

20. CPU2 : CPU\_Range := 1;

21. N : Integer := 4;

22.

23. package new\_data is new data(Size => N);

24. use new\_data;

25.

26. d : Integer;

27. A, B, C, Z : Vector;

28. MO, MT, MH : Matrix;

29. S2, S3, Skd1: Suspension\_Object;

30.

31. procedure Start is

32.

33. --Specification of Task 1 for F1

34. task Task1 is

35. pragma Task\_Name("Task 1");

36. --pragma Priority(1);

37. pragma CPU(CPU1);

38. pragma Storage\_Size(65535);

39. end Task1;

40.

41. --Specification of Task 2 for F2

42. task Task2 is

43. pragma Task\_Name("Task 2");

44. --pragma Priority(5);

45. pragma CPU(CPU2);

46. pragma Storage\_Size(65535);

47. end Task2;

48.

49. task body Task1 is

50. C1, Z1 : Vector;

51. d1 : Integer;

52. MT1 : Matrix;

53. begin

54. Put\_Line("Task 1 start");

55.

56.

57. --Task 1 start waiting Task 2 input"

58. Suspend\_Until\_True(S2);

59.

60. --Task 1 wait to enter CS

61. Suspend\_Until\_True(Skd1);

62. --Task 1 start copying data

63. C1 := C;

64. Z1 := Z;

65. d1 := d;

66. MT1 := MT;

67. --Task 1 finish copying data

68. Set\_True(Skd1);

69.

70. --Task 1 start computing

71. F(A, B, C1, Z1, d1, MO, MT1, MH, 1, N/2);

72. --Task 1 finish computing

73.

74. --Task 1 wait for Task 2

75. Suspend\_Until\_True(S3);

76.

77. if (N < 7) then

78. Put("A = ");

79. Vector\_Output(A);

80. end if;

81.

82. Put\_Line("Task 1 finish");

83. end Task1;

84.

85. task body Task2 is

86. C2, Z2 : Vector;

87. d2 : Integer;

88. MT2 : Matrix;

89. begin

90. Put\_Line("Task 2 start");

91.

92. --Task 2 start input

93. d := 1;

94. Fill\_Vector\_With\_Num(B, 1);

95. Fill\_Vector\_With\_Num(C, 1);

96. Fill\_Vector\_With\_Num(Z, 1);

97. Fill\_Matrix\_With\_Num(MO, 1);

98. Fill\_Matrix\_With\_Num(MT, 1);

99. Fill\_Matrix\_With\_Num(MH, 1);

100.

101. --Task 2 send signal to Task 1

102. Set\_True(S2);

103. --Task 2 finish input about finishing entering data

104.

105. --Task 2 wait to enter CS

106. Suspend\_Until\_True(Skd1);

107. --Task 2 start copying data

108. C2 := C;

109. Z2 := Z;

110. d2 := d;

111. MT2 := MT;

112. --Task 2 finish copying data

113. Set\_True(Skd1);

114.

115. --Task 2 start computing

116. F(A, B, C2, Z2, d2, MO, MT2, MH, N/2 + 1, N);

117. --Task 2 finish computing

118.

119. --Task 2 send signal to Task 1 about finishing computing

120. Set\_True(S3);

121.

122. Put\_Line("Task 2 finish");

123. end Task2;

124. begin

125. null;

126. end Start;

127. begin

128. Put\_Line("Main start");

129. Set\_True(Skd1);

130. Start;

131. Put\_Line("Main finish");

132. end Main;

1. generic

2. Size : Positive;

3. package Data is

4.

5. --Type declaration

6. type Vector is private;

7. type Matrix is private;

8.

9.

10. --Procedure declaration

11. procedure Vector\_Input (v : out Vector);

12. procedure Vector\_Output(v : in Vector);

13. procedure Matrix\_Input (m : out Matrix);

14. procedure Matrix\_Output(m : in Matrix);

15. procedure Fill\_Vector\_With\_Num(v : in out Vector; num : in Integer);

16. procedure Fill\_Matrix\_With\_Num(m : in out Matrix; num : in Integer);

17.

18. procedure F(A : out Vector; B, C, Z: Vector; d: in Integer; MO, MT, MH: in Matrix; from, to: in Integer);

19.

20. --Type determination

21. private

22. type Vector is array(1..Size) of Integer;

23. type Matrix is array(1..Size, 1..Size) of Integer;

24.

25. end Data;

1. with Ada.Text\_IO, Ada.Integer\_Text\_IO;

2. use Ada.Text\_IO, Ada.Integer\_Text\_IO;

3.

4. package body Data is

5.

6. procedure Vector\_Input (v : out Vector) is

7. begin

8. for i in 1..Size loop

9. Get(v(i));

10. end loop;

11. end Vector\_Input;

12.

13. procedure Vector\_Output(v : in Vector) is

14. begin

15. Put("(");

16. for i in 1..Size loop

17. Put(v(i));

18. Put(" ");

19. end loop;

20. Put(")");

21. New\_Line;

22. end Vector\_Output;

23.

24. procedure Matrix\_Input (m : out Matrix) is

25. begin

26. for i in 1..Size loop

27. for j in 1..Size loop

28. Get(m(i, j));

29. end loop;

30. end loop;

31. end Matrix\_Input;

32.

33. procedure Matrix\_Output(m : in Matrix) is

34. begin

35. for i in 1..Size loop

36. for j in 1..Size loop

37. Put(m(i, j));

38. Put(" ");

39. end loop;

40. New\_Line;

41. end loop;

42. New\_Line;

43. end Matrix\_Output;

44.

45. procedure Fill\_Vector\_With\_Num(v : in out Vector; num : in Integer) is

46. begin

47. for i in 1..Size loop

48. v(i) := num;

49. end loop;

50. end Fill\_Vector\_With\_Num;

51.

52. procedure Fill\_Matrix\_With\_Num(m : in out Matrix; num : in Integer) is

53. begin

54. for i in 1..Size loop

55. for j in 1..Size loop

56. m(i, j) := num;

57. end loop;

58. end loop;

59. end Fill\_Matrix\_With\_Num;

60.

61.

62. --Multiplication of vector and matrix

63. procedure Vector\_Matrix\_Multiplication(A : in Vector; B: in Matrix; C : in out Vector; from , to: in Integer) is

64. buf: Integer;

65. begin

66. for i in from..to loop

67. buf := 0;

68. for j in 1..Size loop

69. buf := buf + A(j) \* B(j, i);

70. end loop;

71. C(i) := buf;

72. end loop;

73. end Vector\_Matrix\_Multiplication;

74.

75. -- Multiplication of matrices

76. procedure Matrix\_Multiplication(A, B: in Matrix; C : in out Matrix; From, To: in Integer) is

77. buf: Integer;

78. begin

79. for i in 1..Size loop

80. for j in from..to loop

81. buf := 0;

82. for k in 1..Size loop

83. buf := buf + A(i, k)\*B(k, j);

84. end loop;

85. C(i, j) := Buf;

86. end loop;

87. end loop;

88. end Matrix\_Multiplication;

89.

90. procedure F(A : out Vector; B, C, Z: Vector; d: in Integer; MO, MT, MH: in Matrix; from, to: in Integer) is

91. Mbuf: Matrix;

92. Vbuf: Vector;

93. begin

94. Matrix\_Multiplication(MT, MH, MBuf, from, to);

95. Vector\_Matrix\_Multiplication(C, MO, Vbuf, from, to);

96. Vector\_Matrix\_Multiplication(Z, MBUF, A, from, to);

97. for i in from..to loop

98. A(i) := A(i) + (B(i) + VBuf(i)) \* d;

99. end loop;

100. end F;

101.

102. end Data;