Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №2

Win32. Семафори, м’ютекси, події, критичні секції

Виконав:

студент групи ІП-22

Сочка О. О.

Перевірив:

Корочкін О. В.

Київ - 2015

**Лабораторна робота №1. АДА. Семафори**

**Мета роботи:** розробка програми для ПКС зі СП

**Мова програмування:** C++

**Засоби організації взаємодії процесів:** семафори, м'ютекси, події, критичні секції бібліотеки Win32

**Завдання:**

, MA

4

3

2

1

Спільна пам’ять

**Етап 1. Розробка паралельного математичного алгоритму**

Спільні ресурси:

**Етап 2. Розробка алгоритмів задач**

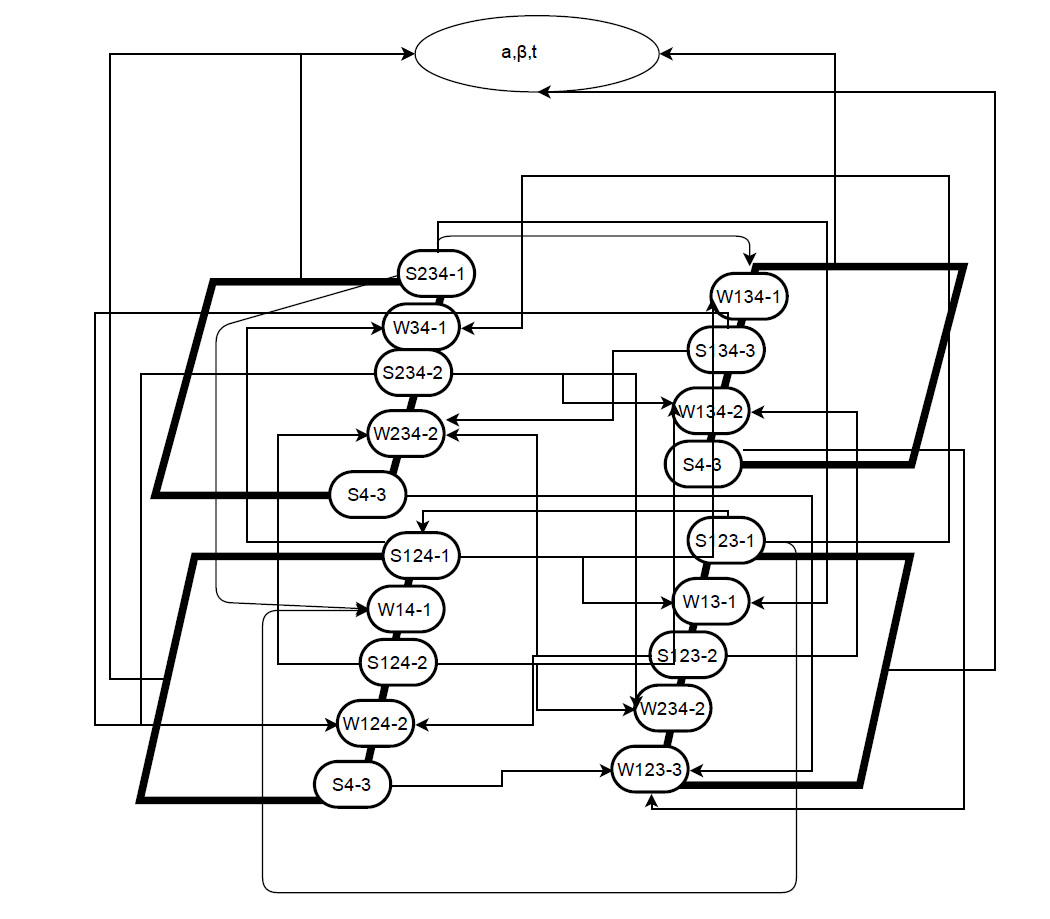
|  |  |  |
| --- | --- | --- |
|  | Т1 | ТС, КУ |
| 1 | Ввід |  |
| 2 | Сигнал T2, T3, T4 про ввід даних | S234-1 |
| 3 | Очікування сигналів від T3, T4 про ввід даних | W34-1 |
| 4 | Копія a1 = a, b1 = b | КУ |
| 5 | Рахунок |  |
| 6 | Сигнал про завершення підрахунку | S234-2 |
| 7 | Очікування сигналів від T2, T3, T4 про заверш. підр. | W234-2 |
| 8 | Підрахунок | КУ |
| 9 | Підрахунок |  |
| 10 | Сигнал T4 про завершення підрахунку | S4-3 |

|  |  |  |
| --- | --- | --- |
|  | T2 | ТС, КУ |
| 1 | Очікування сигналів від T1, T3, T4 про ввід даних | W134-1 |
| 2 | Копія a2 = a, b2 = b | КУ |
| 3 | Рахунок |  |
| 4 | Сигнал про завершення підрахунку | S134-3 |
| 5 | Очікування сигналів від T1, T3, T4 про заверш. підр. | W134-2 |
| 6 | Підрахунок | КУ |
| 7 | Підрахунок |  |
| 8 | Сигнал T4 про завершення підрахунку | S4-3 |

|  |  |  |
| --- | --- | --- |
|  | Т3 | ТС, КУ |
| 1 | Ввід |  |
| 2 | Сигнал T1, T2, T4 про ввід даних | S124-1 |
| 3 | Очікування сигналів від T1, T4 про ввід даних | W14-1 |
| 4 | Копія a3 = a, b3 = b | КУ |
| 5 | Рахунок |  |
| 6 | Сигнал про завершення підрахунку | S124-2 |
| 7 | Очікування сигналів від T1, T2, T4 про заверш. підр. | W124-2 |
| 8 | Підрахунок | КУ |
| 9 | Підрахунок |  |
| 10 | Сигнал T4 про завершення підрахунку | S4-3 |

|  |  |  |
| --- | --- | --- |
|  | Т4 | ТС, КУ |
| 1 | Ввід |  |
| 2 | Сигнал T1, T2, T3 про ввід даних | S123-1 |
| 3 | Очікування сигналів від T1, T3 про ввід даних | W13-1 |
| 4 | Копія a4 = a, b4 = b | КУ |
| 5 | Рахунок |  |
| 6 | Сигнал про завершення підрахунку | S123-2 |
| 7 | Очікування сигналів від T2, T3, T4 про заверш. підр. | W234-2 |
| 8 | Підрахунок | КУ |
| 9 | Підрахунок |  |
| 10 | Очікування сигналів від T1,T2,T3 про заверш. підр. | W123-3 |
| 11 | Вивід MA |  |

**Етап 3. Розробка схеми взаємодії задач**

****

**Етап 4. Розробка програми**

**Лістинг програми**

1 /\*  
2 ---------------------------------------------------------------------------------  
3 -- Лабораторна робота №2 з дисципліни ПП-2  
4 -- Lab2. Win32  
5 -- ПІБ Сочка Олександр Олександрович  
6 -- Група ІП-22  
7 -- Дата 26.02.2015р  
8 -- A = max(MO) \* MK \* MT \* a + b \* MK \* MR  
9 ---------------------------------------------------------------------------------  
10 \*/  
11 #define NOMINMAX  
12 #include <Windows.h>  
13 #include <iostream>  
14 #include <vector>  
15 #include <algorithm>  
16   
17 using namespace std;  
18   
19 typedef std::vector<int> Vector;  
20 typedef std::vector<Vector> Matrix;  
21   
22 const int N = 4;  
23 const int P = 4;  
24   
25 Matrix MA(N, Vector(N)), MO, MK, MT, MR;  
26 int alpha, beta;  
27   
28 HANDLE ev\_io1; // IO is finished in t1  
29 HANDLE ev\_io3; // IO is finished in t3  
30 HANDLE ev\_io4; // IO is finished in t4  
31 HANDLE ev\_t; // t is calculated  
32 HANDLE Sem1\_2;  
33 HANDLE Sem1\_3;  
34 HANDLE Sem1\_4;  
35 HANDLE Sem2\_2;  
36 HANDLE Sem2\_3;  
37 HANDLE Sem2\_4;  
38 CRITICAL\_SECTION CS1; // copy1 & copy2  
39 HANDLE Mtx; // copy2  
40   
41 int t1, t2, t3, t4, t;  
42   
43 void read\_matrix(Matrix& m) {  
44 m = Matrix(N, Vector(N, 1));  
45 }  
46   
47 void read\_num(int& value) {  
48 value = 1;  
49 }  
50   
51 void F1() {  
52 cout << "T1 started" << endl;  
53 read\_matrix(MO);  
54 read\_matrix(MK);  
55   
56 SetEvent(ev\_io1);  
57 WaitForSingleObject(ev\_io3, INFINITE);  
58 WaitForSingleObject(ev\_io4, INFINITE);  
59   
60 EnterCriticalSection(&CS1);  
61 int alpha\_c = alpha, beta\_c = beta;  
62 LeaveCriticalSection(&CS1);  
63   
64   
65 for (int r = 0; r < N / 4; ++r) {  
66 t1 = max(t1, \*max\_element(MO[r].begin(), MO[r].end()));  
67 }  
68   
69 WaitForSingleObject(Sem1\_2, INFINITE);  
70 WaitForSingleObject(Sem1\_3, INFINITE);  
71 WaitForSingleObject(Sem1\_4, INFINITE);  
72   
73 t = max(max(t1, t2), max(t3, t4));  
74   
75 SetEvent(ev\_t);  
76   
77 WaitForSingleObject(Mtx, INFINITE);  
78 int t\_c = t;  
79 ReleaseMutex(Mtx);  
80   
81 // main calculation  
82 for (int i = 0; i < N / 2; ++i) {  
83 for (int j = 0; j < N; ++j) {  
84 for (int k = 0; k < N / 2; ++k) {  
85 MA[i][k] += t\_c \* MK[i][j] \* MT[j][k] \* alpha\_c + beta\_c \* MK[i][j] \* MR[j][k];  
86 }  
87 }  
88 }  
89   
90 // end of main calculation  
91   
92 WaitForSingleObject(Sem2\_2, INFINITE);  
93 WaitForSingleObject(Sem2\_3, INFINITE);  
94 WaitForSingleObject(Sem2\_4, INFINITE);  
95   
96 if (N <= 10) {  
97 for (int i = 0; i < N; ++i) {  
98 for (int j = 0; j < N; ++j) {  
99 cout << MA[i][j] << ' ';  
100 }  
101 cout << endl;  
102 }  
103 }  
104   
105 cout << "T1 finished" << endl;  
106 }  
107   
108 void F2() {  
109 cout << "T2 started" << endl;  
110   
111 WaitForSingleObject(ev\_io1, INFINITE);  
112 WaitForSingleObject(ev\_io3, INFINITE);  
113 WaitForSingleObject(ev\_io4, INFINITE);  
114   
115 EnterCriticalSection(&CS1);  
116 int alpha\_c = alpha, beta\_c = beta;  
117 LeaveCriticalSection(&CS1);  
118   
119 for (int r = N / 4; r < N / 2; ++r) {  
120 t2 = max(t1, \*max\_element(MO[r].begin(), MO[r].end()));  
121 }  
122 ReleaseSemaphore(Sem1\_2, 1, NULL);  
123   
124 WaitForSingleObject(ev\_t, INFINITE);  
125   
126 WaitForSingleObject(Mtx, INFINITE);  
127 int t\_c = t;  
128 ReleaseMutex(Mtx);  
129   
130 // main calculation  
131 for (int i = N / 2; i < N; ++i) {  
132 for (int j = 0; j < N; ++j) {  
133 for (int k = 0; k < N / 2; ++k) {  
134 MA[i][k] += t\_c \* MK[i][j] \* MT[j][k] \* alpha\_c + beta\_c \* MK[i][j] \* MR[j][k];  
135 }  
136 }  
137 }  
138   
139 // end of main calculation  
140 ReleaseSemaphore(Sem2\_2, 1, NULL);  
141 cout << "T2 finished" << endl;  
142 }  
143   
144 void F3() {  
145 cout << "T3 started" << endl;  
146 read\_matrix(MT);  
147 read\_num(alpha);  
148   
149 SetEvent(ev\_io3);  
150   
151 WaitForSingleObject(ev\_io1, INFINITE);  
152 WaitForSingleObject(ev\_io4, INFINITE);  
153   
154 EnterCriticalSection(&CS1);  
155 int alpha\_c = alpha, beta\_c = beta;  
156 LeaveCriticalSection(&CS1);  
157   
158 for (int r = N / 2; r < N / 4 \* 3; ++r) {  
159 t3 = max(t1, \*max\_element(MO[r].begin(), MO[r].end()));  
160 }  
161 ReleaseSemaphore(Sem1\_3, 1, NULL);  
162   
163 WaitForSingleObject(ev\_t, INFINITE);  
164   
165 WaitForSingleObject(Mtx, INFINITE);  
166 int t\_c = t;  
167 ReleaseMutex(Mtx);  
168   
169 // main calculation  
170 for (int i = 0; i < N / 2; ++i) {  
171 for (int j = 0; j < N; ++j) {  
172 for (int k = N / 2; k < N; ++k) {  
173 MA[i][k] += t\_c \* MK[i][j] \* MT[j][k] \* alpha\_c + beta\_c \* MK[i][j] \* MR[j][k];  
174 }  
175 }  
176 }  
177   
178 // end of main calculation  
179   
180 ReleaseSemaphore(Sem2\_3, 1, NULL);  
181 cout << "T3 finished" << endl;  
182 }  
183   
184 void F4() {  
185 cout << "T4 started" << endl;  
186 read\_matrix(MR);  
187 read\_num(beta);  
188   
189 SetEvent(ev\_io4);  
190 WaitForSingleObject(ev\_io1, INFINITE);  
191 WaitForSingleObject(ev\_io3, INFINITE);  
192   
193 EnterCriticalSection(&CS1);  
194 int alpha\_c = alpha, beta\_c = beta;  
195 LeaveCriticalSection(&CS1);  
196   
197 for (int r = N / 4 \* 3; r < N; ++r) {  
198 t4 = max(t1, \*max\_element(MO[r].begin(), MO[r].end()));  
199 }  
200 ReleaseSemaphore(Sem1\_4, 1, NULL);  
201   
202 WaitForSingleObject(ev\_t, INFINITE);  
203   
204 WaitForSingleObject(Mtx, INFINITE);  
205 int t\_c = t;  
206 ReleaseMutex(Mtx);  
207   
208 // main calculation  
209 for (int i = N / 2; i < N; ++i) {  
210 for (int j = 0; j < N; ++j) {  
211 for (int k = N / 2; k < N; ++k) {  
212 MA[i][k] += t\_c \* MK[i][j] \* MT[j][k] \* alpha\_c + beta\_c \* MK[i][j] \* MR[j][k];  
213 }  
214 }  
215 }  
216   
217 // end of main calculation  
218 ReleaseSemaphore(Sem2\_4, 1, NULL);  
219 cout << "T4 finished" << endl;  
220 }  
221   
222 int main() {  
223   
224 InitializeCriticalSection(&CS1);  
225 Mtx = CreateMutex(NULL, FALSE, NULL);  
226 ev\_io1 = CreateEvent(NULL, TRUE, FALSE, NULL);  
227 ev\_io3 = CreateEvent(NULL, TRUE, FALSE, NULL);  
228 ev\_io4 = CreateEvent(NULL, TRUE, FALSE, NULL);  
229 ev\_t = CreateEvent(NULL, TRUE, FALSE, NULL);  
230   
231 Sem1\_2 = CreateSemaphore(NULL, 0, 1, NULL);  
232 Sem1\_3 = CreateSemaphore(NULL, 0, 1, NULL);  
233 Sem1\_4 = CreateSemaphore(NULL, 0, 1, NULL);  
234 Sem2\_2 = CreateSemaphore(NULL, 0, 1, NULL);  
235 Sem2\_3 = CreateSemaphore(NULL, 0, 1, NULL);  
236 Sem2\_4 = CreateSemaphore(NULL, 0, 1, NULL);  
237   
238 DWORD thread\_id;  
239 HANDLE T1 = CreateThread(NULL, 0, (LPTHREAD\_START\_ROUTINE)F1, NULL, 0, &thread\_id);  
240 HANDLE T2 = CreateThread(NULL, 0, (LPTHREAD\_START\_ROUTINE)F2, NULL, 0, &thread\_id);  
241 HANDLE T3 = CreateThread(NULL, 0, (LPTHREAD\_START\_ROUTINE)F3, NULL, 0, &thread\_id);  
242 HANDLE T4 = CreateThread(NULL, 0, (LPTHREAD\_START\_ROUTINE)F4, NULL, 0, &thread\_id);  
243   
244 WaitForSingleObject(T1, INFINITE);  
245 WaitForSingleObject(T2, INFINITE);  
246 WaitForSingleObject(T3, INFINITE);  
247 WaitForSingleObject(T4, INFINITE);  
248 CloseHandle(T1);  
249 CloseHandle(T2);  
250 CloseHandle(T3);  
251 CloseHandle(T4);  
252 CloseHandle(ev\_io1);  
253 CloseHandle(ev\_io3);  
254 CloseHandle(ev\_io4);  
255 CloseHandle(ev\_t);  
256 CloseHandle(Mtx);  
257 CloseHandle(Sem1\_2);  
258 CloseHandle(Sem1\_3);  
259 CloseHandle(Sem1\_4);  
260 CloseHandle(Sem2\_2);  
261 CloseHandle(Sem2\_3);  
262 CloseHandle(Sem2\_4);  
263 DeleteCriticalSection(&CS1);  
264 }  
265