Для демонстрации данного алгоритма предоставлен следующий код.

Файл WIRE.cpp без функций соединения с другими процессами.

while (TRUE)

{

while (cbRead1 == 0 && cbRead2 == 0)

{

ReadFile(hNamedPipe1, szBuf1, BUFSIZ, &cbRead1, NULL);

ReadFile(hNamedPipe2, szBuf2, BUFSIZ, &cbRead2, NULL);

}

int timeSleep = 1000;

if (cbRead1 != 0)

{

cout << "open collision window (" << (double)timeSleep / 1000 << ") sec:" << endl;

while (cbRead2 == 0 && timeSleep > 0)

{

if (timeSleep % 500 == 0)

{

cout << (double)timeSleep / 1000 << " sec..." << endl;

}

ReadFile(hNamedPipe2, szBuf2, BUFSIZ, &cbRead2, NULL);

timeSleep--;

Sleep(1);

}

cout << "close collision window" << endl;

if (cbRead2 == 0)

{

WriteFile(hNamedPipe2, szBuf1, BUFSIZ, &cbWritten2, NULL);

WriteFile(hNamedPipe1, szBuf1, BUFSIZ, &cbWritten1, NULL);

ReleaseSemaphore(hSemaphore1, 1, NULL);

ReleaseSemaphore(hSemaphore2, 1, NULL);

}

else

{

strcpy\_s(szBuf1, "collision");

WriteFile(hNamedPipe1, szBuf1, BUFSIZ, &cbWritten1, NULL);

strcpy\_s(szBuf2, "collision");

WriteFile(hNamedPipe2, szBuf2, BUFSIZ, &cbWritten2, NULL);

ReleaseSemaphore(hSemaphore1, 1, NULL);

ReleaseSemaphore(hSemaphore2, 1, NULL);

}

cout << "otpravil" << endl;

}

else

{

cout << "open collision window (" << (double)timeSleep / 1000 << ") sec:" << endl;

while (cbRead1 == 0 && timeSleep > 0)

{

if (timeSleep % 500 == 0)

{

cout << (double)timeSleep / 1000 << " sec..." << endl;

}

ReadFile(hNamedPipe1, szBuf1, BUFSIZ, &cbRead1, NULL);

timeSleep--;

Sleep(1);

}

cout << "close collision window" << endl;

if (cbRead1 == 0)

{

WriteFile(hNamedPipe2, szBuf2, BUFSIZ, &cbWritten2, NULL);

WriteFile(hNamedPipe1, szBuf2, BUFSIZ, &cbWritten1, NULL);

ReleaseSemaphore(hSemaphore1, 1, NULL);

ReleaseSemaphore(hSemaphore2, 1, NULL);

}

else

{

strcpy\_s(szBuf1, "collision");

WriteFile(hNamedPipe1, szBuf1, BUFSIZ, &cbWritten1, NULL);

strcpy\_s(szBuf2, "collision");

WriteFile(hNamedPipe2, szBuf2, BUFSIZ, &cbWritten2, NULL);

ReleaseSemaphore(hSemaphore1, 1, NULL);

ReleaseSemaphore(hSemaphore2, 1, NULL);

}

cout << "otpravil" << endl;

}

cbRead1 = 0;

cbRead2 = 0;

}

Файл COMP1.cpp без функций соединения с другими процессами.

cout << "PRESS KEY TO START THE STATION 1!" << endl;

char\* startMessage = "MESSAGE FROM FIRST STATION";

strcpy\_s(szBuf, startMessage);

rewind(stdin);

\_getch();

for (int attempt = 0; attempt < 10; attempt++)

{

cout << endl;

Sleep(1);

strcpy\_s(szBuf, startMessage);

int timeDelay = delay(100, 100 \* pow(2, attempt));

cout << timeDelay<<endl;

Sleep(timeDelay);

WriteFile(hNamedPipe1, szBuf, BUFSIZ, &cbWritten, NULL);

WaitForSingleObject(hSemaphore1, INFINITE);

while (cbRead == 0)

{

ReadFile(hNamedPipe1, szBuf, BUFSIZ, &cbRead, NULL);

}

if (strcmp(szBuf, "collision") == 0)

{

cout << "collision" << endl;

cbRead = 0;

continue;

}

else

{

cbRead = 0;

if (strcmp(startMessage, szBuf) == 0)

{

cout << "Our message is sended. Wait for another message:"<<endl;

while (cbRead == 0)

{

ReadFile(hNamedPipe1, szBuf, BUFSIZ, &cbRead, NULL);

}

cout << szBuf << endl;

system("pause");

return 0;

}else

{

cout << "Find a message:" << endl;

cout << szBuf << endl;

ReadFile(hNamedPipe1, szBuf, BUFSIZ, &cbRead, NULL);

cbRead = 0;

if (strcmp(startMessage, szBuf) == 0)

{

cout << "Our message is sended." << endl;

system("pause");

return 0;

}

attempt--;

continue;

}

}

cbRead = 0;

}

cout << endl << "All attempts were used. Can't send the message." << endl;

Файл COMP2.cpp идентичен COMP1.cpp за исключением названий соответствующих переменных и передаваемого сообщения.

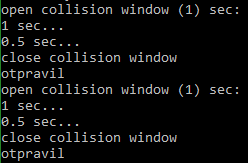
В данном коде присутствуют 3 процесса, 2 из которых представляют взаимодействующие станции (COMP1 и COMP2) и канал их взаимодействия (WIRE). При передаче сообщения от одной станции другой в первую очередь оно попадает в WIRE, где открывается окно коллизий – период, во время которого может произойти коллизия.

Если коллизия не произошла, то станция, передав сообщение, просто выжидает ответного сообщения от другой станции.

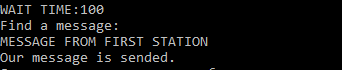
Процесс COMP1



Процесс WIRE

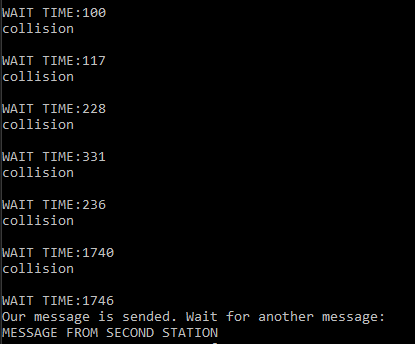


Процесс COMP2

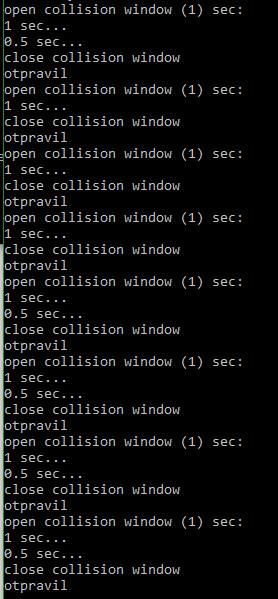


Если коллизия всё-таки происходит, то станциям возвращается сообщение об этом, и они выжидают случайное время для предотвращения, которое отображается как WAIT TIME перед сообщением collision в процессах. Это происходит до того момента пока не закончатся попытки либо сообщения не будут переданы.

Процесс COMP1



Процесс WIRE



Процесс COMP2

