

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»

Институт Информатики и кибернетики   
Кафедра Программных систем

Дисциплина Технологии промышленного программирования

**ОТЧЁТ**  
  
к лабораторной работе

«Запуск и организация взаимодействия параллельных процессов»

Обучающийся группы 6231-020302D \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гижевская В.Д.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Баландин А.В.

Самара 2023

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#_Toc147860313)

[2 Результаты работы 4](#_Toc147860314)

[ПРИЛОЖЕНИЕ А Листинг модуля M1 5](#_Toc147860315)

[ПРИЛОЖЕНИЕ B Листинг модуля M2 7](#_Toc147860316)

[ПРИЛОЖЕНИЕ С Листинг модуля M3 9](#_Toc147860317)

1. Постановка задачи

Разработать приложение, состоящее из трех взаимодействующих процессов. Требуется написать три программных модуля – М1, М2, М3. На базе модуля М1 из shell запускается стартовый процесс Р1(М1).

Процесс Р1 создает канал и, используя функцию семейства spawn\*(), запускает процесс Р2(М2), передавая ему в качестве параметра chid созданного канала, затем переходит в состояние приема сообщений по созданному каналу.

Процесс Р2 создает свой канал, и, используя функцию семейства spawn\*(), запускает процесс Р3(М3), передавая ему в качестве параметра chid созданного канала, затем устанавливает соединение с каналом процесса Р1, передает ему chid своего канала, затем переходит в состояние приема сообщений по своему каналу. Процесс Р3 устанавливает соединение с каналом процесса Р2 и посылает ему запрос на получение pid процесса Р1 и chid его канала. Получив от Р2 ответ (pid и chid), устанавливает соединение с каналом процесса Р1 и посылает ему свой pid и chid своего канала, после чего переходит в состояние приёма сообщений по своему каналу.

Процесс Р1 устанавливает соединение с каналом процесса Р2 и передает ему сообщение "Р1 send message to Р2", получив ответ, выводит его на терминал, затем устанавливает соединение с каналом процесса Р3 и передает ему сообщение "Р1 send message to Р3", получает ответ, выводит его на экран и терминируется.

Процесс Р2, получив сообщение от процесса Р1, выводит его на терминал, посылает ответ "Р2 executed " и терминируется. Процесс Р3, получив сообщение от процесса Р1, выводит его на экран, посылает ответ "Р3 executed " и терминируется.

1. Результаты работы

Результаты работы представлены в виде вывода на консоль сообщений во время выполнения программы.

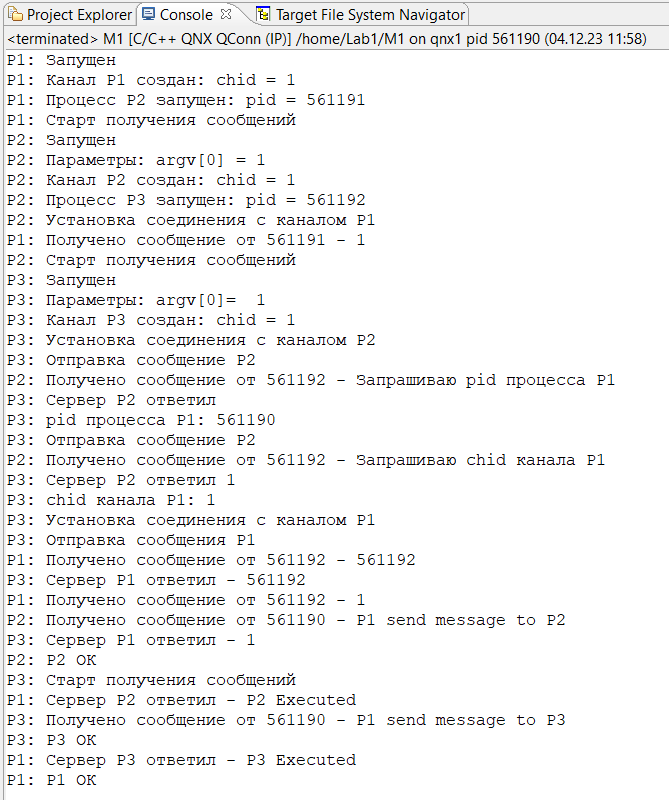


Рисунок 1 – Результат выполнения программы в консоли

ПРИЛОЖЕНИЕ А  
Листинг модуля M1

**#include** <cstdlib>

**#include** <iostream>

**#include** <stdlib.h>

**#include** <sys/neutrino.h>

**#include** <unistd.h>

**#include** <process.h>

**#include** <string.h>

**#include** <unistd.h>

**#define** GetCurrentDir getcwd

**int** **main**(**int** argc, **char** \*argv[]) {

std::cout << "P1: Запущен" << std::endl;

**int** chid = **ChannelCreate**(\_NTO\_CHF\_SENDER\_LEN);

std::cout << "P1: Канал P1 создан: chid = " << chid << std::endl;

**char** buffer[20];

**const** **char** \*chidStr = **itoa**(chid, buffer, 10);

**int** pidP2 = **spawnl**(P\_NOWAIT, "/home/host/Lab1/M2/x86/o/M2", chidStr, NULL);

**if** (pidP2 < 0){

std::cout << "P1: Ошибка запуска процесса P2" << pidP2 << std::endl;

exit(EXIT\_FAILURE);

}

std::cout << "P1: Процесс P2 запущен: pid = " << pidP2 << std::endl;

**int** pidP3 = {};

**int** chidP3 = {};

std::cout << "P1: Старт получения сообщений" << std::endl;

**int** countMsg = 0;

**while**(countMsg < 3){

**char** msg[200] = {};

\_msg\_info info;

**int** rcvid;

rcvid = **MsgReceive**(chid, msg, **sizeof**(msg), &info);

**if**(rcvid == -1){

std::cout << "P1: Ошибка MsgReceive" << std::endl;

}

std::cout << "P1: Получено сообщение от " << info.pid << " - " << msg << std::endl;

**if** (countMsg == 0){

**MsgReply**(rcvid, NULL, msg, **sizeof**(msg));

}

**if** (countMsg == 1) {

pidP3 = atoi(msg);

**MsgReply**(rcvid, NULL, msg, **sizeof**(msg));

}

**if** (countMsg == 2) {

chidP3 = atoi(msg);

**MsgReply**(rcvid, NULL, msg, **sizeof**(msg));

}

countMsg++;

}

**char** rmsg[200] = {};

**int** coidP2 = **ConnectAttach**(0, pidP2, chid, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP2 == -1){

std::cout << "P1: Ошибка соединения с каналом P2" << std::endl;

exit(EXIT\_FAILURE);

}

**char** \*msg1 = (**char** \*)"P1 send message to P2";

**if**(MsgSend(coidP2, msg1, strlen(msg1) + 1, rmsg, **sizeof**(rmsg)) == -1){

std::cout << "P1: Ошибка MsgSend" << std::endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg) > 0){

std::cout << "P1: Сервер P2 ответил - " << rmsg << std::endl;

}

**int** coidP3 = **ConnectAttach**(0, pidP3, chidP3, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP3 == -1){

std::cout << "P1: Ошибка соединения с каналом P3" << std::endl;

exit(EXIT\_FAILURE);

}

**char** \*msg2 = (**char** \*)"P1 send message to P3";

**if**(MsgSend(coidP3, msg2, strlen(msg2) + 1, rmsg, **sizeof**(rmsg)) == -1){

std::cout << "P1: Ошибка MsgSend при отправки в P3" << std::endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg) > 0){

std::cout << "P1: Сервер P3 ответил - " << rmsg << std::endl;

}

std::cout << "P1: Р1 ОК" << std::endl;

**return** EXIT\_SUCCESS;

}

ПРИЛОЖЕНИЕ B  
Листинг модуля M2

**#include** <cstdlib>

**#include** <iostream>

**#include** <stdlib.h>

**#include** <sys/neutrino.h>

**#include** <unistd.h>

**#include** <process.h>

**#include** <string.h>

**#include** <unistd.h>

**#define** GetCurrentDir getcwd

**int** **main**(**int** argc, **char** \*argv[]) {

std::cout << "P2: Запущен" << std::endl;

std::cout << "P2: Параметры: " << "argv[0] = " << argv[0] << std::endl;

**int** pChid = atoi(argv[0]);

**int** chid = **ChannelCreate**(\_NTO\_CHF\_SENDER\_LEN);

std::cout << "P2: Канал P2 создан: chid = " << chid << std::endl;

**char** buffer[20];

**const** **char** \*chidStr = **itoa**(chid, buffer, 10);

**int** pidP3 = **spawnl**(P\_NOWAIT, "/home/host/Lab1/M3/x86/o/M3", chidStr, NULL);

**if** (pidP3 < 0){

std::cout << "P2: Ошибка запуска процесса P3" << pidP3 << std::endl;

exit(EXIT\_FAILURE);

}

std::cout << "P2: Процесс P3 запущен: pid = " << pidP3 << std::endl;

std::cout << "P2: Установка соединения с каналом P1" << std::endl;

**int** coidP1 = **ConnectAttach**(0, **getppid**(), pChid, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP1 == -1){

std::cout << "P2: Ошибка соединения с каналом P1" << std::endl;

exit(EXIT\_FAILURE);

}

**char** rmsg1[200] = {};

**if**(MsgSend(coidP1, chidStr, strlen(chidStr) + 1, rmsg1, **sizeof**(rmsg1)) == -1){

std::cout << "P2: Ошибка MsgSend" << std::endl;

exit(EXIT\_FAILURE);

}

std::cout << "P2: Старт получения сообщений" << std::endl;

**int** countMsg = 0;

**while**(countMsg < 3){

**char** msg[200] = {};

\_msg\_info info;

**int** rcvid;

rcvid = **MsgReceive**(chid, msg, **sizeof**(msg), &info);

**if**(rcvid == -1){

std::cout << "P2: Ошибка MsgReceive" << std::endl;

}

std::cout << "P2: Получено сообщение от " << info.pid << " - " << msg << std::endl;

**if**(countMsg == 0){

**int** pidP1 = **getppid**();

**char** pidP1Buffer[50];

**char** **const** \*pidP3Str = **itoa**(pidP1, pidP1Buffer, 10);

MsgReply(rcvid, NULL, pidP3Str, strlen(pidP3Str));

}

**if**(countMsg == 1){

**char** \*pChidStr = argv[0];

**MsgReply**(rcvid, NULL, pChidStr, **sizeof**(pChidStr));

}

**if**(countMsg == 2){

**char** \*msgEx = (**char** \*)"P2 Executed";

MsgReply(rcvid, NULL, msgEx, strlen(msgEx) + 1);

}

countMsg ++;

}

std::cout << "P2: Р2 ОК" << std::endl;

**return** EXIT\_SUCCESS;

}

ПРИЛОЖЕНИЕ С  
Листинг модуля M3

**#include** <cstdlib>

**#include** <iostream>

**#include** <stdlib.h>

**#include** <sys/neutrino.h>

**#include** <unistd.h>

**#include** <process.h>

**#include** <string.h>

**#include** <unistd.h>

**#define** GetCurrentDir getcwd

**int** **main**(**int** argc, **char** \*argv[]) {

std::cout << "P3: Запущен" << std::endl;

std::cout << "P3: Параметры: " << "argv[0]= " << argv[0] << std::endl;

**int** chid = **ChannelCreate**(\_NTO\_CHF\_SENDER\_LEN);

std::cout << "P3: Канал P3 создан: chid = " << chid << std::endl;

**int** pChid = atoi(argv[0]);

**int** pidP1;

**int** pChidP1;

std::cout << "P3: Установка соединения с каналом P2" << std::endl;

**int** coidP2 = **ConnectAttach**(0, **getppid**(), pChid, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP2 == -1){

std::cout << "P3: Ошибка соединения с каналом P2" << std::endl;

exit(EXIT\_FAILURE);

}

**char** rmsg[200] = {};

std::cout << "Р3: Отправка сообщение Р2" << rmsg << std::endl;

**char** \*smsg1 = (**char** \*)"Запрашиваю pid процесса Р1";

**if**(MsgSend(coidP2, smsg1, strlen(smsg1) + 1, rmsg, **sizeof**(rmsg)) == -1){

std::cout << "P3: Ошибка MsgSend" << std::endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg) > 0){

std::cout << "P3: Сервер P2 ответил " << std::endl;

pidP1 = atoi(rmsg);

std::cout << "P3: pid процесса Р1: " << pidP1 << std::endl;

}

std::cout << "Р3: Отправка сообщение Р2" << std::endl;

**char** \*smsg2 = (**char** \*)"Запрашиваю chid канала Р1";

**if**(MsgSend(coidP2, smsg2, strlen(smsg2) + 1, rmsg, **sizeof**(rmsg)) == -1){

std::cout << "P3: Ошибка MsgSend" << std::endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg) > 0){

std::cout << "P3: Сервер P2 ответил " << rmsg << std::endl;

pChidP1 = atoi(rmsg);

std::cout << "P3: chid канала Р1: " << pChidP1 << std::endl;

}

std::cout << "P3: Установка соединения с каналом P1" << std::endl;

**int** coidP1 = **ConnectAttach**(0, pidP1, pChidP1, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP1 == -1){

std::cout << "P3: Ошибка соединения с каналом P1" << std::endl;

exit(EXIT\_FAILURE);

}

**char** rmsg2[200] = {};

std::cout << "Р3: Отправка сообщения Р1" << std::endl;

**int** pid = **getpid**();

**char** pidBuffer[50];

**char** **const** \*pidStr = **itoa**(pid, pidBuffer, 10);

**char** buffer[20];

**const** **char** \*chidStr = **itoa**(chid, buffer, 10);

**if**(MsgSend(coidP1, pidStr, strlen(pidStr) + 1, rmsg2, **sizeof**(rmsg2)) == -1){

std::cout << "P3: Ошибка MsgSend" << std::endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg2) > 0){

std::cout << "P3: Сервер P1 ответил - " << rmsg2 << std::endl;

}

**if**(MsgSend(coidP1, chidStr, strlen(chidStr) + 1, rmsg2, **sizeof**(rmsg2)) == -1){

std::cout << "P3: Ошибка MsgSend" << std::endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg2) > 0){

std::cout << "P3: Сервер P1 ответил - " << rmsg2 << std::endl;

}

std::cout << "P3: Старт получения сообщений" << std::endl;

**int** countMsg = 0;

**while**(countMsg < 1){

**char** msg[200] = {};

\_msg\_info info;

**int** rcvid;

rcvid = **MsgReceive**(chid, msg, **sizeof**(msg), &info);

**if**(rcvid == -1){

std::cout << "P3: Ошибка MsgReceive" << std::endl;

}

**char** \*msgEx = (**char** \*)"P3 Executed";

std::cout << "P3: Получено сообщение от " << info.pid << " - " << msg << std::endl;

MsgReply(rcvid, NULL, msgEx, strlen(msgEx) + 1);

countMsg++;

}

std::cout << "P3: Р3 ОК" << std::endl;

**return** EXIT\_SUCCESS;

}