Министерство науки и высшего образования

Российской Федерации

Федеральное государственное бюджетное образовательное   
учреждение высшего образования

«Новосибирский государственный технический университет»



Кафедра теоретической и прикладной информатики

### Лабораторная работа № 5 по дисциплине «Проектирование Систем Реального Времени»

### Синхронизация потоков

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Факультет: | ПМИ |  |  |
| Группа: | ПМИМ-01 |  |  |
| Студенты: | Ершов П. К.  Грициенко И. Г. |  |  |
| Бригада: | 7 |  |  |
| Преподаватель: | Кобылянский В. Г. |  |  |

Новосибирск

2021

1. **Цель работы**

Цель работы - изучение основных принципов использования QNX Momentics IDE.

1. **Задание на лабораторную**
2. С помощью средств перспективы QNX System Information определить следующую информацию о целевой платформе:

* архитектуру и частоту процессора;
* общий объем оперативной памяти и объем занятой памяти;
* версию операционной системы QNX;
* количество запущенных процессов;
* количество потоков у процесса **pipe** и их состояние.

1. Создать QNX C Project с кодом приложения согласно варианту. Передать проект под управление CVS.
2. Скомпилировать проект, исправить синтаксические ошибки, если такие имеются. Занести исправления в репозиторий с описанием изменений.
3. Создать конфигурацию для запуска отладки и выполнить пошаговую отладку приложения. Внесенные исправления также заносить в репозиторий с описанием.
4. Создать конфигурацию для обычного запуска без отладочной информации и выполнить запуск приложения. Сравнить размеры исполняемых файлов с отладочной информацией и без нее (пункт Properties в контекстном меню нужного файла в представлении Project Explorer).
5. Выполнить профилирование методами “Function Instrumentation profiling” и “Sampling and Call Count instrumentation profiling”. Из результатов получить время выполнения функций программы для обоих методов. Занести данные по времени в таблицу:

|  |  |
| --- | --- |
| Функция | Затраченное время |
| main |  |
| sorting\_str |  |
| print\_str |  |

1. **Ход работы.**
   1. Определение информации о целевой платформе с помощью перспективы QNX System Information

* архитектура и частота процессора: x86, 3.186 Ghz



Рисунок 1.

* общий объем оперативной памяти и объем занятой памяти 2 ГБт



Рисунок 2.

* версия операционной системы QNX 6.5.9

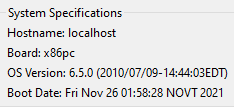


Рисунок 3.

* количество запущенных процессов - 36

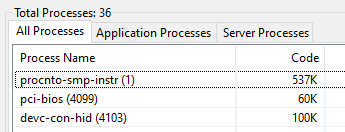
****

Рисунок 4.

* количество потоков у процесса **pipe** и их состояние

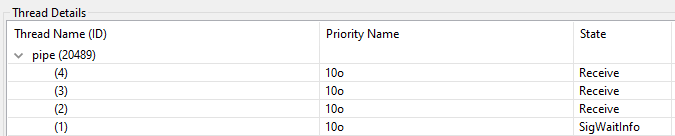
****

Рисунок 5.

* 1. Создать QNX C Project с кодом приложения согласно варианту. Передать проект под управление CVS

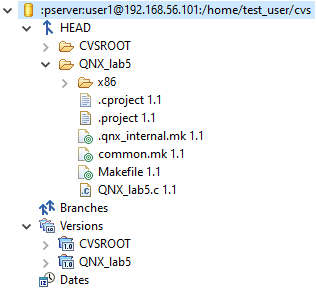
****

Рисунок 6. Проект создан и передан под управление CVS

* 1. Cкомпилировать проект, исправить синтаксические ошибки, если такие имеются. Занести исправления в репозиторий с описанием изменений.

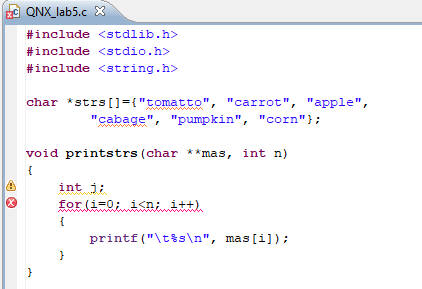


Рисунок 7. Ошибка в программе.

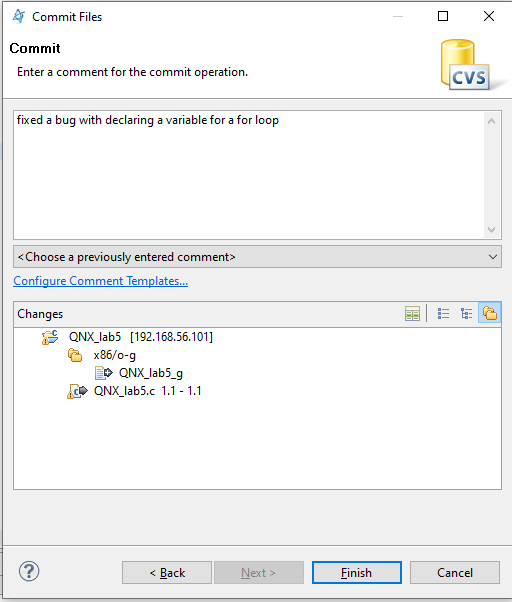


Рисунок 8. Создание коммита

* 1. Создать конфигурацию для запуска отладки и выполнить пошаговую отладку приложения. Внесенные исправления также заносить в репозиторий с описанием

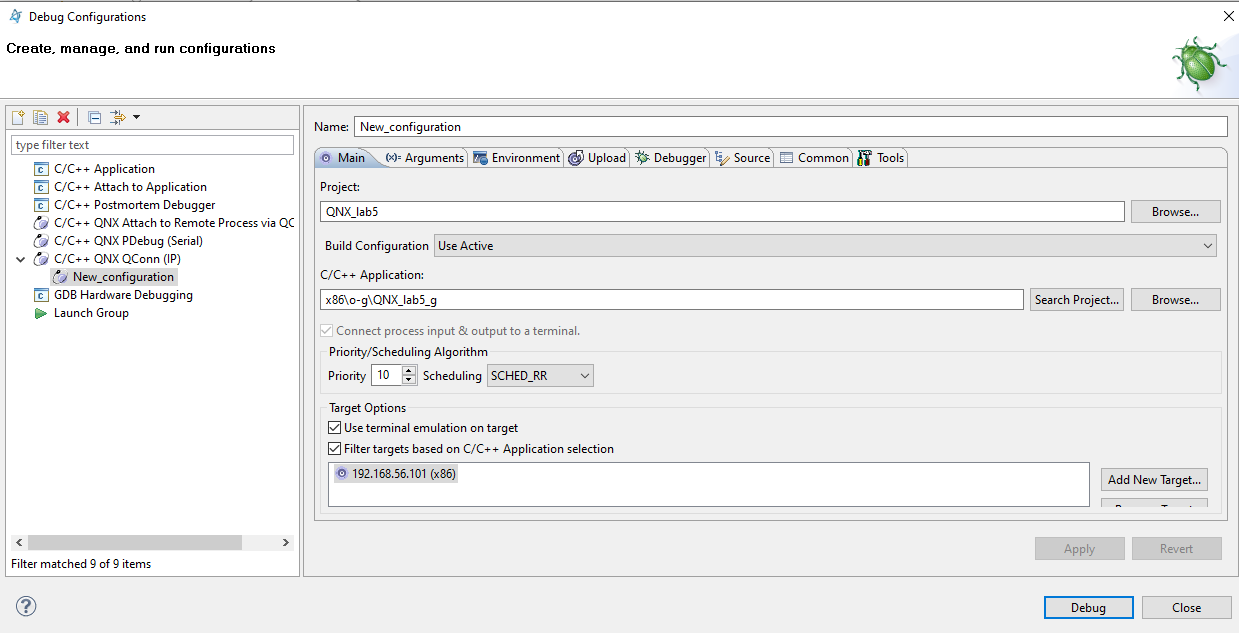


Рисунок 9. Конфигурация отладки

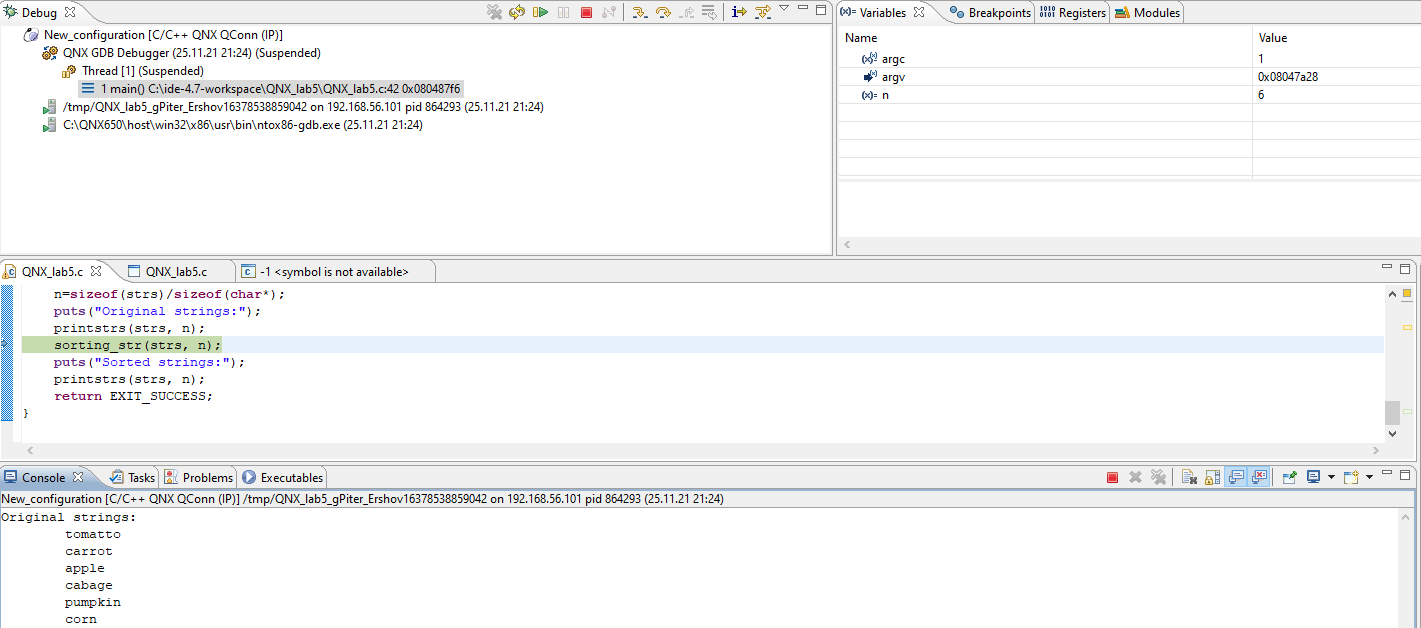


Рисунок 10. Окно отладки

* 1. Создать конфигурацию для обычного запуска без отладочной информации и выполнить запуск приложения. Сравнить размеры исполняемых файлов с отладочной информацией и без нее (пункт Properties в контекстном меню нужного файла в представлении Project Explorer

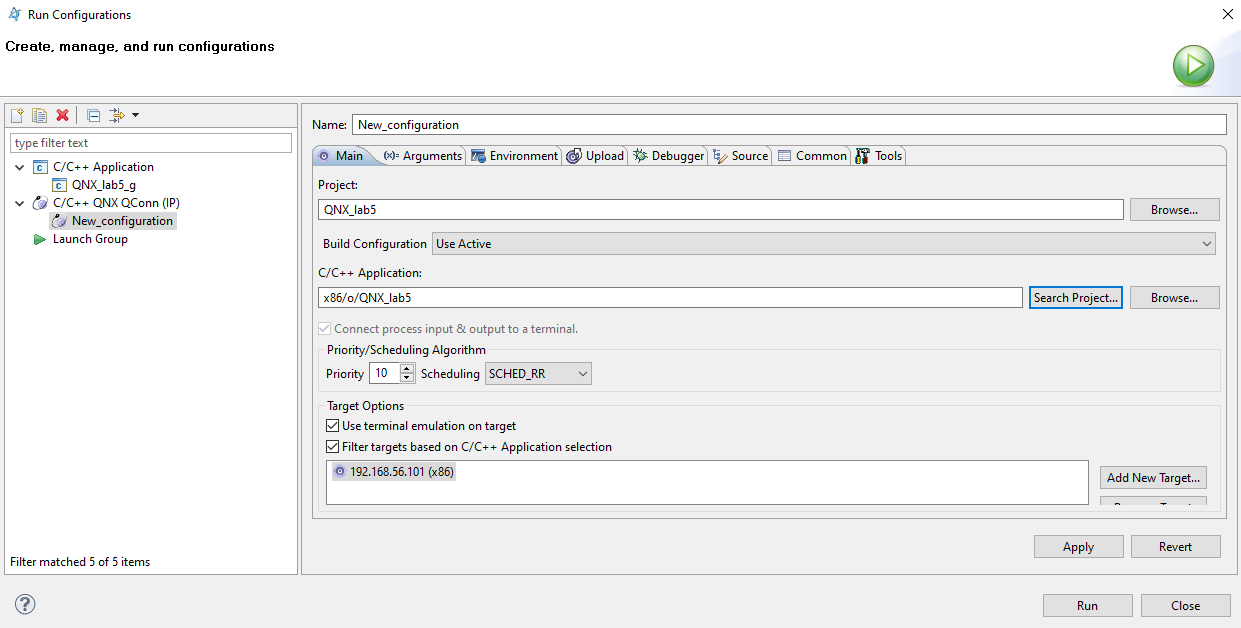


Рисунок 11. Конфигурация простого запуска (без отладки)

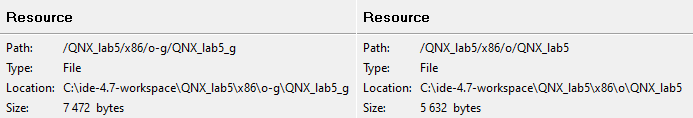


Рисунок 12. Сравнение размеров исполняемого файла отладки (слева) и обычного запуска (справа)

Как хорошо видно на рисунке 12, размер исполняемого файла отладки заметно больше.

* 1. Выполнить профилирование методами “Function Instrumentation profiling” и “Sampling and Call Count instrumentation profiling”. Из результатов получить время выполнения функций программы для обоих методов. Занести данные по времени в таблицу

|  |  |  |
| --- | --- | --- |
| Метод профилирования | Function Instrumentation profiling | Sampling and Call Count instrumentation profiling |
| Функция | Затраченное время | |
| main | 0,068 мс |  |
| sorting\_str | 0,001 мс |  |
| print\_str | 0,987 мс, 1,621 мс |  |

1. **Код программы**

**#include** <stdlib.h>

**#include** <stdio.h>

**#include** <string.h>

**char** \*strs[]={"tomatto", "carrot", "apple", "cabage", "pumpkin", "corn"};

**void** **printstrs**(**char** \*\*mas, **int** n)

{

**int** i;

**for**(i=0; i<n; i++)

**printf**("\t%s\n", mas[i]);

}

**void** **sorting\_str**(**char** \*\*mas, **int** n)

{

**int** i, j, min;

**char** \*temp;

**for**(i=0; i<n-1; i++)

{

min=i;

**for**(j=0; j<n; j++)

{

**if**(**strcmp**(mas[j], mas[min])<0)

min=j;

}

temp=mas[i];

mas[i]=mas[min];

mas[min]=temp;

}

}

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

{

**int** n;

n=**sizeof**(strs)/**sizeof**(**char**\*);

**puts**("Original strings:");

printstrs(strs, n);

sorting\_str(strs, n);

**puts**("Sorted strings:");

printstrs(strs, n);

**return** EXIT\_SUCCESS;

}