Министерство образования и науки РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рыбинский государственный авиационный технический университет

имени П.А. Соловьева»

Факультет радиоэлектроники и информатики

Кафедра Математическое и программное обеспечение ЭВС

**ОТЧЕТ**

**по лабораторной работе №1**

**R**EADER 1

Студент группы ИПБ-15 Козин О.И .

*(Код) (Подпись, дата) (Фамилия И. О.)*

Рыбинск 2019

Оглавление

[Цель 2](#_Toc969503)

[Простейшие взаимодействия 2](#_Toc969504)

[Архитектура 2](#_Toc969505)

[Анализ работы программ 3](#_Toc969506)

[Управляющие автоматы для вариантов задачи Читатели – писатели 3](#_Toc969507)

[Приложение. Коды планировщиков. 5](#_Toc969508)

[Приложение. Интерпретация временных диаграмм 11](#_Toc969509)

# Цель

1. Разработать автоматную модель задачи читатели писатели
2. Научиться переводить автоматную модель в управляющую программу ADA
3. Запрограммировать решение задачи «Читатели – писатели» в соответствии с автоматной схемой.

# Простейшие взаимодействия

## Архитектура

На рис.1 представлена архитектура решения задачи «Читатели – писатели».

Точка входа

**start**

**start**

**start**

**start**

**Main**

**R**EADER 3

**S**hared **R**esource

**R**EADER 2

**W**RITER

**RqR**

**FIN**

**RqW**

**S**heduler

**R**EADER 1

Рис. 1. Архитектура программы решения задачи «Читатели – писатели»

## Анализ работы программ

Ниже представлен фрагмент временных диаграмм работы программы с соответственно автоматным схемам.

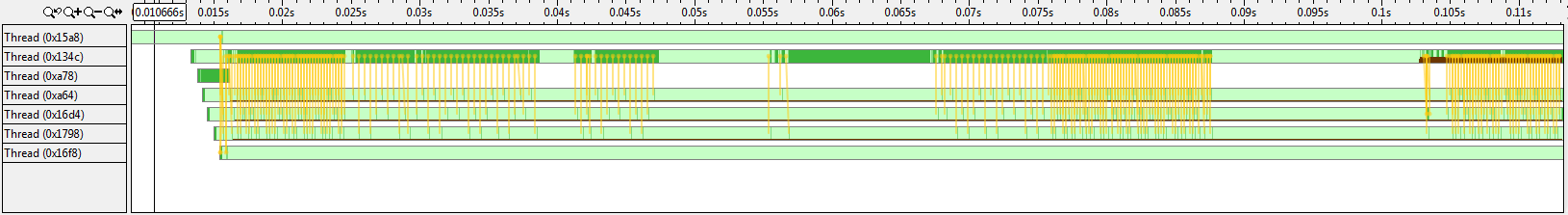


Рис. 2. Временные диаграммы работы программы читатели-писатели (без приоритета писателя)

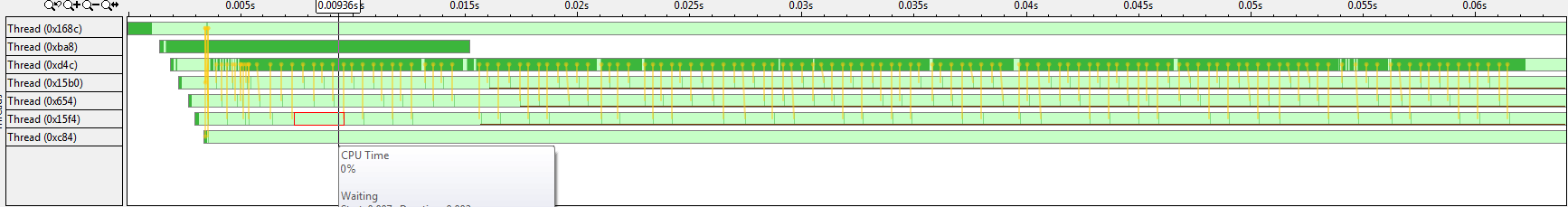


Рис. 3. Временные диаграммы начала работы программы читатели-писатели (с приоритетом писателя, через условие)

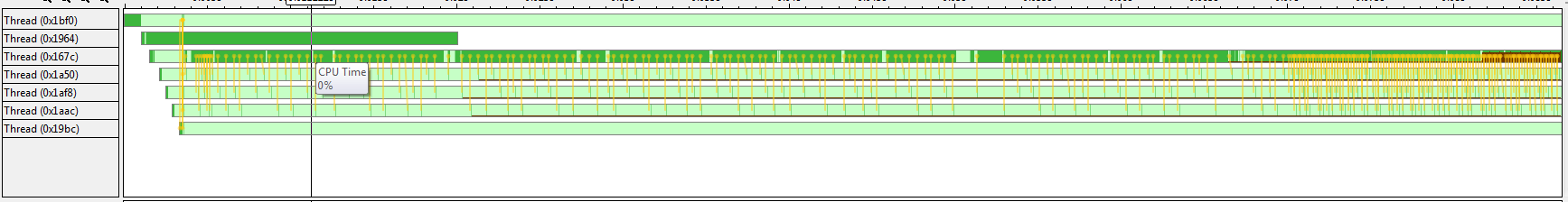


Рис. 4. Временные диаграммы середины работы программы читатели-писатели (с приоритетом писателя, через 5 состояний)

## Управляющие автоматы для вариантов задачи Читатели – писатели

w

Fin

Fin

Last Fin

r

r

Рис. 5. Автоматная схема задачи читатели-писатели (без приоритета писателя)

w

w

Fin

Fin

Fin

Last Fin

Last Fin

r

r

Рис. 6. Автоматная схема задачи читатели-писатели (с приоритетом писателя, через условие)

w

w

Fin

Fin

Fin

Last Fin

Last Fin

r

r

Рис. 7. Автоматная схема задачи читатели-писатели (с приоритетом писателя , через 5 состояний)

## Схема программы на языке c#

Write

Read

# Приложение. Коды планировщиков.

Вариант задачи 1.

Put\_Line (

"Controller start");

loop

case State is

when 0 =>

select

accept Read do State := 1;

Readers\_Count:=Readers\_Count+1;

Put\_Line (

"read start");

Put\_Line(Integer'Image(Readers\_Count));

end Read;

or

accept Write do State := 2;

Put\_Line (

"write start");

end Write;

end select;

when 1 =>

select

accept Read do State := 1;

Readers\_Count:=Readers\_Count+1;

Put\_Line (

"read start");

Put\_Line(Integer'Image(Readers\_Count));

end Read;

or

accept Fin do Readers\_Count := Readers\_Count - 1;

Put\_Line (

"Fin read");

Put\_Line(Integer'Image(Readers\_Count));

if (Readers\_Count = 0)

then

State := 0;

end if;

end Fin;

end select;

when 2 =>

select

accept Fin do State := 0;

Put\_Line (

"Fin write");

end Fin;

or

delay 0.1;

Put\_Line (

"Wait write Fin");

end select;

when others => -- error

null;

end case;

end loop;

Put\_Line (

"Controller End");

Вариант задачи 2.

Put\_Line (

"Controller start");

loop

case State is

when 0 =>

select

accept Read do State := 1;

Readers\_Count:=Readers\_Count+1;

Put\_Line (

"read start");

Put\_Line(Integer'Image(Readers\_Count));

end Read;

or

accept Write do State := 2;

Put\_Line (

"write start");

end Write;

end select;

when 1 =>

select

accept Read do State := 1;

Readers\_Count:=Readers\_Count+1;

Put\_Line (

"read start");

Put\_Line(Integer'Image(Readers\_Count));

end Read;

or

accept Fin do Readers\_Count := Readers\_Count - 1;

Put\_Line (

"Fin read");

Put\_Line(Integer'Image(Readers\_Count));

if (Readers\_Count = 0)

then

State := 0;

end if;

end Fin;

or

accept Write do while Readers\_Count > 0 loop accept Fin;

Readers\_Count := Readers\_Count - 1;

end loop;

Put\_Line (

"Start write");

State := 2;

end Write;

end select;

when 2 =>

select

accept Fin do State := 0;

Put\_Line (

"Fin write");

end Fin;

or

delay 0.1;

Put\_Line (

"Wait write Fin");

end select;

when others => -- error

null;

end case;

end loop;

Put\_Line (

"Controller End")

Вариант задачи 3.

Put\_Line (

"Controller start");

loop

case State is

when 0 =>

select

accept Read do State := 1;

Readers\_Count:=Readers\_Count+1;

Put\_Line (

"read start");

Put\_Line(Integer'Image(Readers\_Count));

end Read;

or

accept Write do State := 2;

Put\_Line (

"write start");

end Write;

end select;

when 1 =>

select

accept Read do State := 1;

Readers\_Count:=Readers\_Count+1;

Put\_Line (

"read start");

Put\_Line(Integer'Image(Readers\_Count));

end Read;

or

accept Fin do Readers\_Count := Readers\_Count - 1;

Put\_Line (

"Fin read");

Put\_Line(Integer'Image(Readers\_Count));

if (Readers\_Count = 0)

then

State := 0;

end if;

end Fin;

or

accept Write do State := 3;

Put\_Line (

"write req ");

end Write;

end select;

when 2 =>

select

accept Fin do State := 0;

Put\_Line (

"Fin write");

end Fin;

or

Put\_Line (

"Wait write Fin");

end select;

when 3 =>

select

accept Fin do Readers\_Count := Readers\_Count - 1;

Put\_Line (

"Stop read");

Put\_Line(Integer'Image(Readers\_Count));

if (Readers\_Count = 0)

then

State := 2;

Put\_Line (

"Start write");

end if;

end Fin;

end select;

when others => -- error

null;

end case;

end loop;

Put\_Line (

"Controller End");

;

Решение на c#

class rwt

{

static int buffer;

static int readCount = 0;

static int writersCount = 1;

static int readersCount = 3;

static Thread[] writers = new Thread[writersCount];

static Thread[] readers = new Thread[readersCount];

static Semaphore readCountAccess;

static Semaphore databaseAccess;

static void Main(string[] args)

{

readCountAccess = new Semaphore(1, 3);

databaseAccess = new Semaphore(1, 1);

for (int i = 0; i < writersCount; i++)

{

writers[i] = new Thread(new ParameterizedThreadStart(WriterThread));

writers[i].Priority = ThreadPriority.Lowest;

}

for (int i = 0; i < readersCount; i++)

{

readers[i] = new Thread(new ParameterizedThreadStart(ReaderThread));

readers[i].Priority = ThreadPriority.Normal;

}

for (int i = 0; i < writersCount; i++)

{

writers[i].Start(i);

}

for (int i = 0; i < readersCount; i++)

{

readers[i].Start(i);

}

for (int i = 0; i < writersCount; i++)

{

writers[i].Join();

}

for (int i = 0; i < readersCount; i++)

{

readers[i].Join();

}

}

static void WriterThread(object num)

{

int cnt = 0;

Random rand = new Random();

for (;;)

{

Thread.Sleep(rand.Next(100, 2000));

Console.WriteLine("\tWriter {0} trying to write -----------------==={1}", num, cnt);

databaseAccess.WaitOne();

buffer = rand.Next() % 20;

Console.WriteLine("\tWriter {0} is writing {1}", num, buffer);

Thread.Sleep(rand.Next(50, 500));

databaseAccess.Release();

cnt++;

}

}

static void ReaderThread(object num)

{

int cnt = 0;

Random rand = new Random();

for (;;)

{

Thread.Sleep(rand.Next(90, 1000));

Console.WriteLine("Reader {0} trying to read" + "----------------------------------{0}--{1}", num, cnt);

databaseAccess.WaitOne();

Console.WriteLine("Reader {0} is reading {1}", num, buffer);

Thread.Sleep(rand.Next(50, 500));

databaseAccess.Release();

cnt++;

}

}

}

# Приложение. Интерпретация временных диаграмм

