**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

по дисциплине «Операционные системы»

на тему: «Синхронизация процессов»

Выполнил: студент гр. ИТП-11

Ковтун А.В.

Принял: преподаватель

Карась О.В.

Гомель 2022

**Цель работы:** изучить основные алгоритмы планирования процессов. Научиться обрабатывать несколько процессов, и синхронизировать их.

**Задание 1:** выполнить алгоритм синхронизации двух процессов (Р0, Р1) «переменная–замок», использующих общие ресурсы, для данных приведенных в таблице 2.1. Алгоритм планирования процессов *Round Robin (RR),* величина кванта времени 3. Результаты оформить в виде таблицы иллюстрирующей работу процессов.

В алгоритме взаимодействия «переменная–замок» процесс может войти в критическую секцию только тогда, когда значение этой переменной-замка равно 0, одновременно изменяя ее значение на 1 – закрывая замок. При выходе из критической секции процесс сбрасывает ее значение в 0 – замок открывается.

Таблица данных

|  |  |  |  |
| --- | --- | --- | --- |
| Время возникновения входа в критическую секцию для P0 | Время возникновения входа в критическую секцию для P1 | Время выполнения критической секции Р0 | Время выполнения критической секции Р1 |
| 1-5-9-15-22-27-33 | 10-16-19-25-28-30 | 1-1-1-1-1-2 | 1-1-1-1-2-1 |

*Round Robin,* «переменная-замок»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| P0 | КС | И | И | Г | ГК | ГК | КС | И | И | Г | Г | Г | И | И | КС | Г | Г |
| P1 | Г | Г | Г | И | И | И | Г | Г | Г | КС | И | И | Г | Г | Г | КС | И |

*Round Robin,* «переменная-замок»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 |
| P0 | Г | И | И | И | ГК | ГК | ГК | И | И | КС | Г | Г | Г | И | И | КС | КС |
| P1 | И | Г | Г | Г | И | И | И | ГК | ГК | ГК | КС | КС | И | Г | Г | Г | Г |

**Задание 2:** выполнить алгоритм синхронизации двух процессов (Р0, Р1) «строгое – чередование», использующих общие ресурсы, для данных приведенных в таблице 2.1. Алгоритм планирования процессов *Round Robin (RR),* величина кванта времени 3. Результаты оформить в виде таблицы иллюстрирующей работу процессов.

В алгоритме «строгое чередование» очередной подход будет использовать общую переменную с начальным значением 0. Только теперь она будет играть не роль замка для критического участка, а явно указывать, кто может следующим войти в него.



Рисунок 1 – «Строгое чередование»

**Задание 3:** выполнить алгоритмы синхронизации процессов (Р0, Р1) «переменная – замок» и «строгое – чередование», использующих общие ресурсы, при наличии третьего процесса (Р2), не использующего ресурсы процессов Р0, Р1. Данные процессов (Р0, Р1) «приведенных в таблице 2.1, процесс Р2 появляется каждый 6 квант времени, длительность процесса равна 3 квантам. Алгоритм планирования процессов *Round Robin (RR),* величина кванта времени 3. Если процесс Р2 выполниться не успел, новый его экземпляр в очередь не ставится. Процесс Р2 не может прервать выполнение критической секции. Результаты оформить в виде таблиц иллюстрирующих работу процессов.

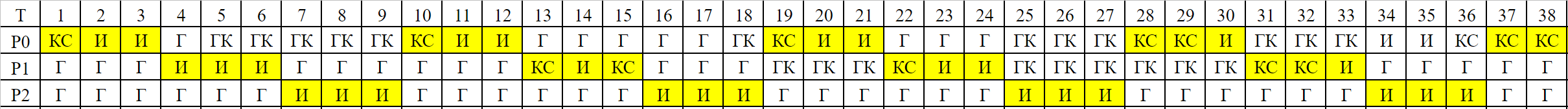


Рисунок 2 - «переменная – замок», три переменных

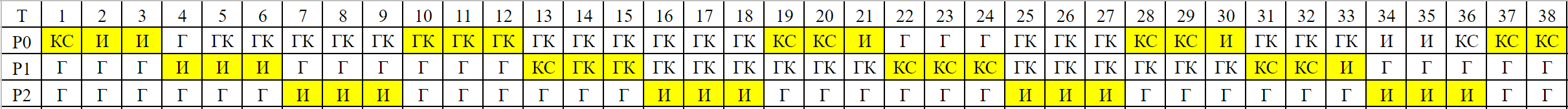


Рисунок 3 – «строгое чередование», три переменных

Задание 4: выполнить алгоритм синхронизации четырех процессов (Р0, Р1, Р2, Р3) «алгоритм булочной», использующих общие ресурсы. Процессы выбираются из таблицы 2.1, согласно таблице 2.2. При каждой постановке в очередь критической секции, вычисляется номер, присваиваемый процессу. Алгоритм планирования процессов *Round Robin,* величина кванта времени 3.

Алгоритм булочной: каждый вновь прибывающий процесс получает номер. Процесс с наименьшим номером обслуживается следующим.

Таблица данных

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P0 | Время возникновения | 2 | 4 | 16 | 22 | 29 | 36 |
| Продолжительность | 1 | 1 | 1 | 2 | 1 | 3 |
| P1 | Время возникновения | 8 | 13 | 19 | 23 | 27 | 32 |
| Продолжительность | 1 | 2 | 2 | 1 | 1 | 1 |
| P2 | Время возникновения | 2 | 5 | 12 | 17 | 23 | 32 |
| Продолжительность | 1 | 1 | 2 | 1 | 2 | 3 |
| P3 | Время возникновения | 10 | 14 | 18 | 25 | 30 | 33 |
| Продолжительность | 1 | 1 | 1 | 1 | 2 | 1 |

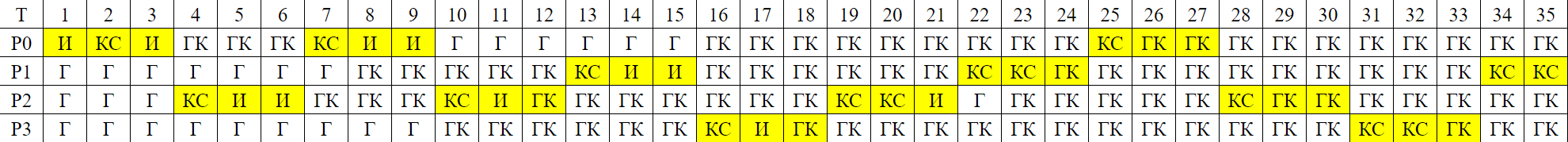


Рисунок 4.1 – Алгоритм булочной

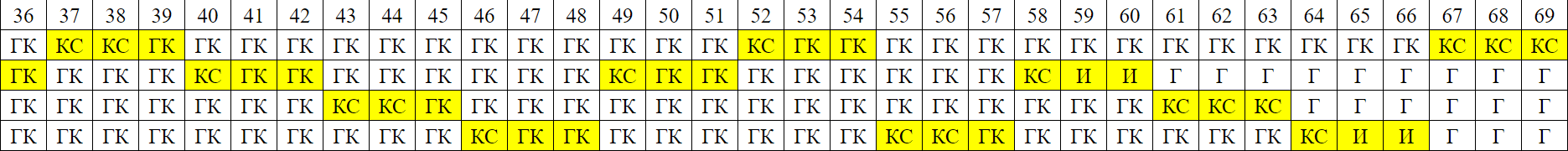


Рисунок 4.2 – Алгоритм булочной

**Вывод:** в лабораторной работе были изучены и протестированы различные алгоритмы планирования процессов. Научились обрабатывать несколько процессов, и синхронизировать их. Выяснили что, чем проще алгоритм, тем меньше времени простоя процессора, однако они также не являются идеальными.