

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | Кафедра |
| информационных технологий | робототехники |
| и систем управления | и мехатроники |

**Курсовой проект**

**«Разработка программы для моделирования обработки на РТК»**

|  |  |
| --- | --- |
|  |  |
|  |  |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Игнатьев В.А. |
|  | (подпись) |
| Студент | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Березин А.И. |
|  | (подпись) |

Москва, 2020

Оглавление

[Введение 3](#_Toc54268438)

[Что такое конечный автомат? 3](#_Toc54268439)

[Детерминированный конечный автомат (deterministic state machine) 4](#_Toc54268440)

[Недетерминированный конечный автомат (nondeterministic state machine) 4](#_Toc54268441)

[Ход работы 5](#_Toc54268442)

[Полученное задание 5](#_Toc54268443)

# Введение

## Что такое конечный автомат?

Конечный автомат (state machine) это модель вычислений, основанная на гипотетической машине состояний. В один момент времени только одно состояние может быть активным. Следовательно, для выполнения каких-либо действий машина должна менять свое состояние.

Конечные автоматы обычно используются для организации и представления потока выполнения чего-либо. Это особенно полезно при реализации ИИ, приложений, игр.

Конечный автомат можно представить в виде графа, вершины которого являются состояниями, а ребра — переходы между ними. Каждое ребро имеет метку, информирующую о том, когда должен произойти переход.

Реализация конечного автомата начинается с выявления его состояний и переходов между ними. Представьте себе конечный автомат, описывающий действия муравья, несущего листья в муравейник:

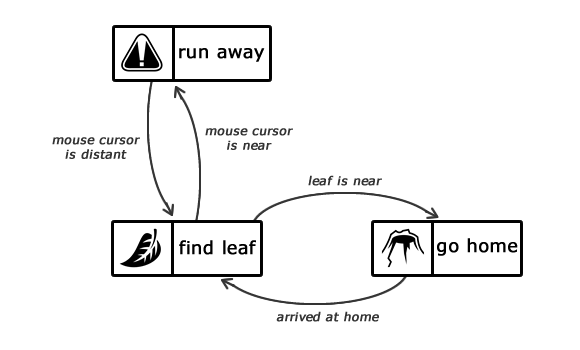


Рис. 1. SM муравья

Отправной точкой является состояние «find leaf», которое остается активным до тех пор, пока муравей не найдет лист. Когда это произойдет, то состояние сменится на «go home». Это же состояние останется активным, пока наш муравей не доберется до муравейника. После этого состояние вновь меняется на «find leaf».

## Детерминированный конечный автомат (deterministic state machine)

Простейший конечный автомат, который может принимать лишь одно состояние в текущий момент времени, обладает детерминированностью.

Детерминированность - для всех состояний имеется максимум и минимум одно правило для любого возможного входного символа, то есть например, для состояния 1 не может быть два перехода с одним и тем же входным символом.

## Недетерминированный конечный автомат (nondeterministic state machine)

Недетерминированный конечный автомат не является каким-то существенным улучшением ДКА, просто в нем добавлен синтаксический сахар, в виде свободных переходов, недетерминированности и множеств состояний. Реализовать можно как массив, состоящий из структур в которой хранится состояние, входной символ и следующее состояние.

# Ход работы

## Описание задачи

1. На основании описания моделируемой системы необходимо определить структуру модели: состав средств и очередей, их назначение, выполняемые операции. А также возможные дополнительные элементы модели – счётчики, переменные, списки.
2. Определить структуру потока заявок, особые свойства транзактов (их типы и приоритеты).
3. Составить список активностей моделируемого процесса с указанием их длительности.
4. Описать алгоритмы процедур событий, завершающих выполнение активностей.
5. Представить логическую структуру взаимодействия активностей и событий.
6. Указать, каким событием начинается моделируемый процесс, и условия завершения моделирования.

На РТК из двух станков и одного робота поступают детали одного типа без перерывов. Обработка детали на первом станке занимает 40 с и дает 4% брака. На втором станке - соответственно 60 с и 8% брака. Все бракованные детали возвращаются на второй станок для повторной обработки. Детали, попавшие в разряд бракованных дважды, считаются отходами и помещаются роботом в специальный контейнер за 3 с. Годные детали загружаются роботом в выходной накопитель за 2 с. Загрузка первого станка требует 5 с, второго - 6 с. Перемещение робота между накопителями - 3 с.

Смоделировать обработку 500 деталей. Определить процент отходов, сравнить с результатами расчета варианта, когда бракованные детали повторно обрабатываются на первом станке.

## Разработка

Первоначальная блок-схема процесса, для ориентировки, представлена на рис. 2.

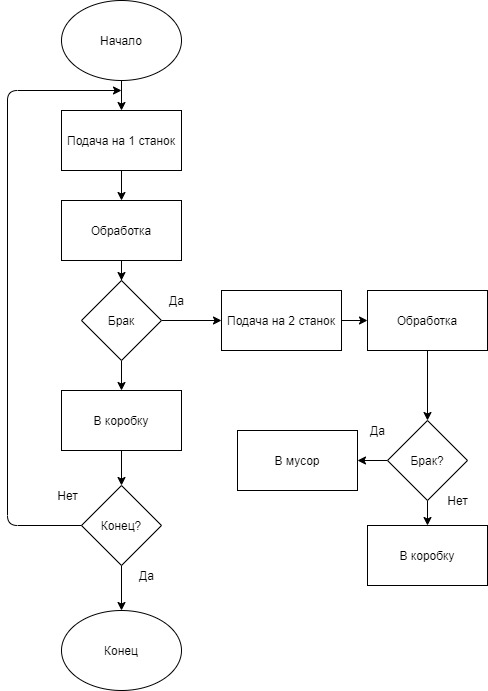


Рис. 2. Первоначальная блок-схема

Было принято решение написания программы на языке программирования G, в среде LabVIEW.

Графический язык программирования «G», используемый в LabVIEW, основан на архитектуре потоков данных. Последовательность выполнения операторов в таких языках определяется не порядком их следования (как в [императивных языках программирования](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), а наличием данных на входах этих операторов. Операторы, не связанные по данным, выполняются параллельно в произвольном порядке.

LabVIEW ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA)  Laboratory Virtual Instrumentation Engineering Workbench) — это [среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) и [платформа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0) для выполнения программ, созданных на [графическом языке программирования](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) «G» фирмы [National Instruments](https://ru.wikipedia.org/wiki/National_Instruments) ([США](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B5%D0%B4%D0%B8%D0%BD%D1%91%D0%BD%D0%BD%D1%8B%D0%B5_%D0%A8%D1%82%D0%B0%D1%82%D1%8B_%D0%90%D0%BC%D0%B5%D1%80%D0%B8%D0%BA%D0%B8)). Первая версия LabVIEW была выпущена в [1986 году](https://ru.wikipedia.org/wiki/1986_%D0%B3%D0%BE%D0%B4) для [Apple Macintosh](https://ru.wikipedia.org/wiki/Macintosh), в настоящее время существуют версии для [Unix](https://ru.wikipedia.org/wiki/Unix), [Linux](https://ru.wikipedia.org/wiki/Linux), [Mac OS](https://ru.wikipedia.org/wiki/Mac_OS) и [Microsoft Windows](https://ru.wikipedia.org/wiki/Windows).

LabVIEW используется в [системах сбора](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%81%D0%B1%D0%BE%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и обработки данных, а также для управления техническими объектами и технологическими процессами. Идеологически LabVIEW очень близка к [SCADA](https://ru.wikipedia.org/wiki/SCADA)-системам, но в отличие от них в большей степени ориентирована на решение задач не столько в области [АСУ ТП](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%BC_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D0%BC), сколько в области [АСНИ](https://ru.wikipedia.org/wiki/%D0%90%D0%A1%D0%9D%D0%98).

Разработка была начата с построения UI и выявления необходимых данных для информативности интерфейса. Последняя версия представлена на рис. 3.

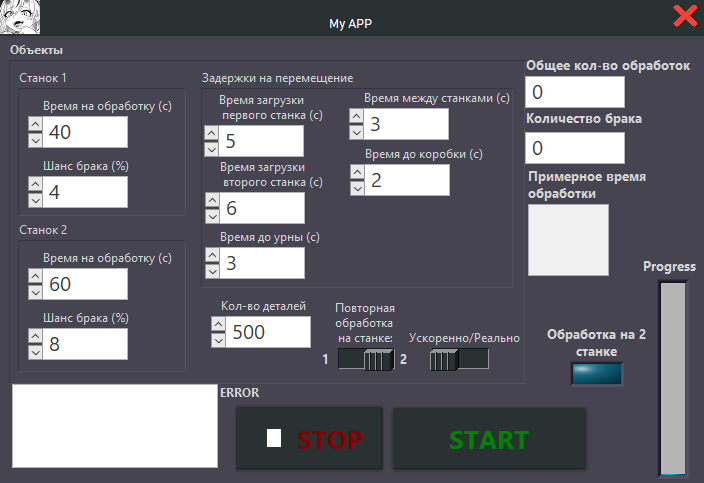


Рис.3 UI приложения MY\_Cuorse

Во время проектирования было принято решение, сделать поля ввода всех значений доступными для пользователя. Также добавлены кнопки состояния скорости исполнения программы (Ускоренно/Реально), где ускоренно покажет тот же результат, только за короткий промежуток, т.е. просто исключит временные задержки. Реальное исполнение будет отрабатывать все циклы и задержки, для удобство пользователя было добавлено поле с примерным временем обработки (значение рассчитывается не зависимо от значения кнопки «Ускоренно/Реально»), которое оперирует к закону распределения случайной величины и показывает затрачиваемое время с небольшой погрешностью. Еще элементами интерфейса являются прогресс бар показывающий прогресс (работа не совсем корректна), поле общее количество обработок показывает сумму обработок на первом и втором станке. Поле количество брака отражает детали, забракованные после повторной обработки. Также добавлен индикатор обработки на 2-ом станке, сделано для уведомления пользователя о том, что обработка на 2 станке идет работа.

В соответствии со второй частью задания была добавлена дополнительная кнопка переключения режима повторной обработки на 1 или 2 станке.

Также имеется окно ошибок для уведомления о возникших проблемах.

## Блок-схемы

Для параллельной обработки 2 задержек было принято решение использовать несколько циклов. Из-за размера блок-схема главной программы представлена в приложении 1. Далее на рисунках 4, 5, 6, 7, 8, 9 представлены используемые подпрограммы.

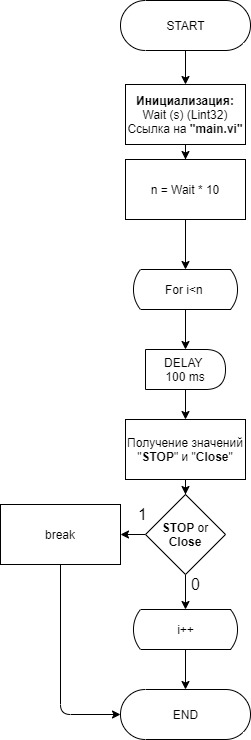


Рис. 4 Блок-схема подпрограммы Delay

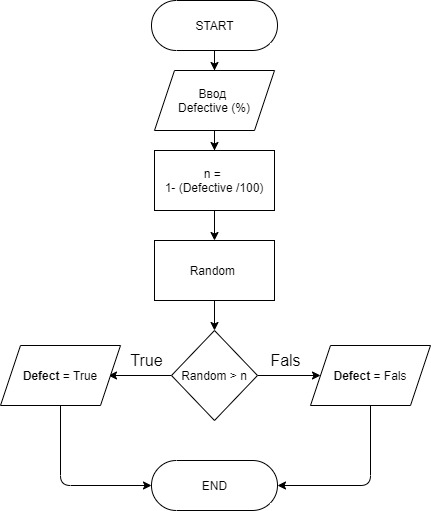


Рис. 5. Блок-схема подпрограммы Calculation chance

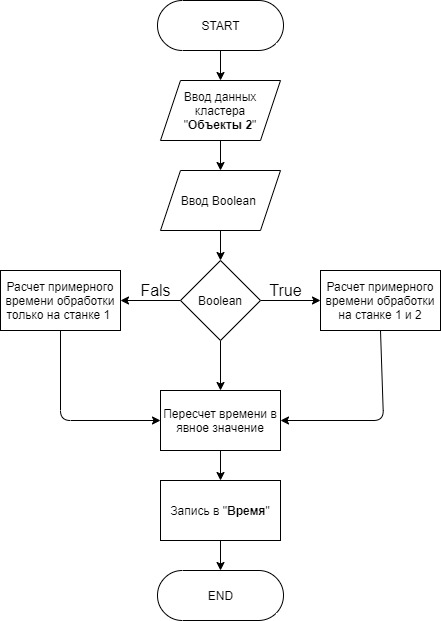


Рис. 6. Блок-схема подпрограммы estimated time

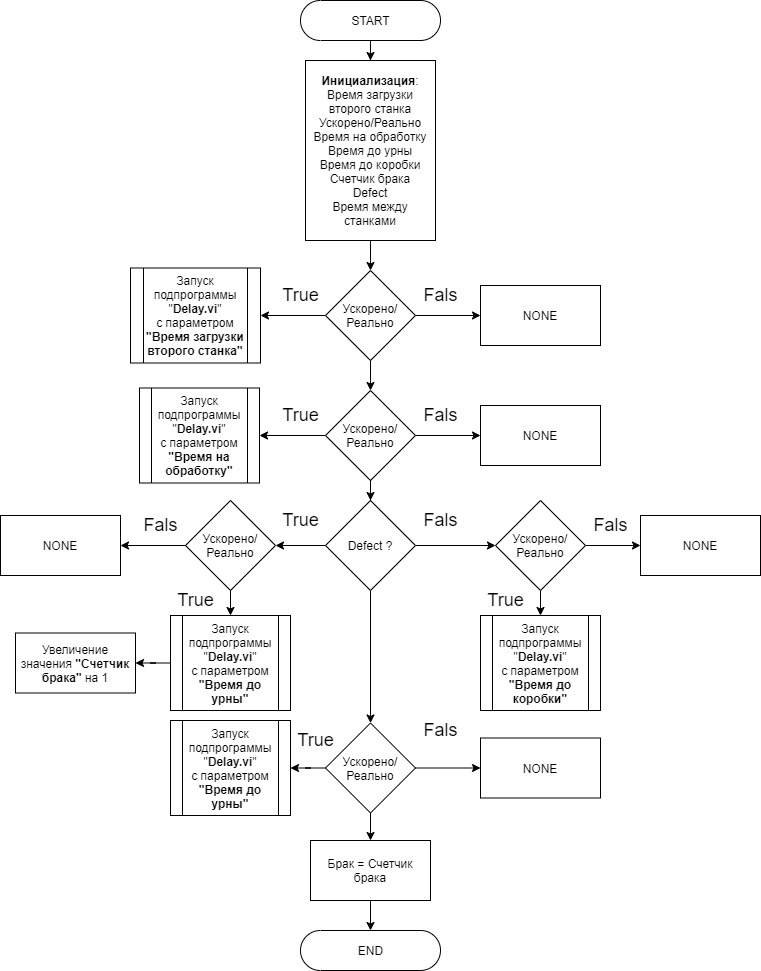


Рис. 7. Блок-схема подпрограммы Flat seque for 2nd machin

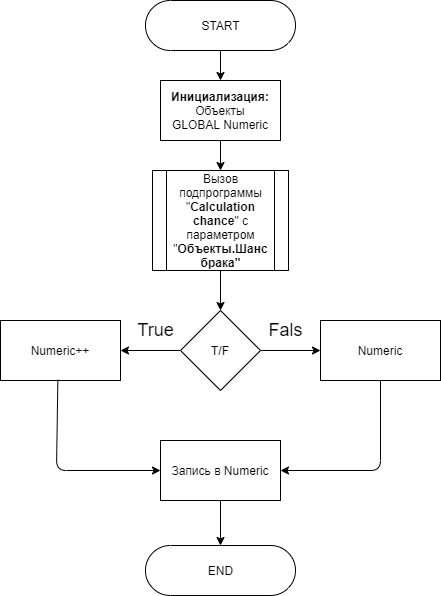


Рис .8. Блок-схема подпрограммы To work on 1

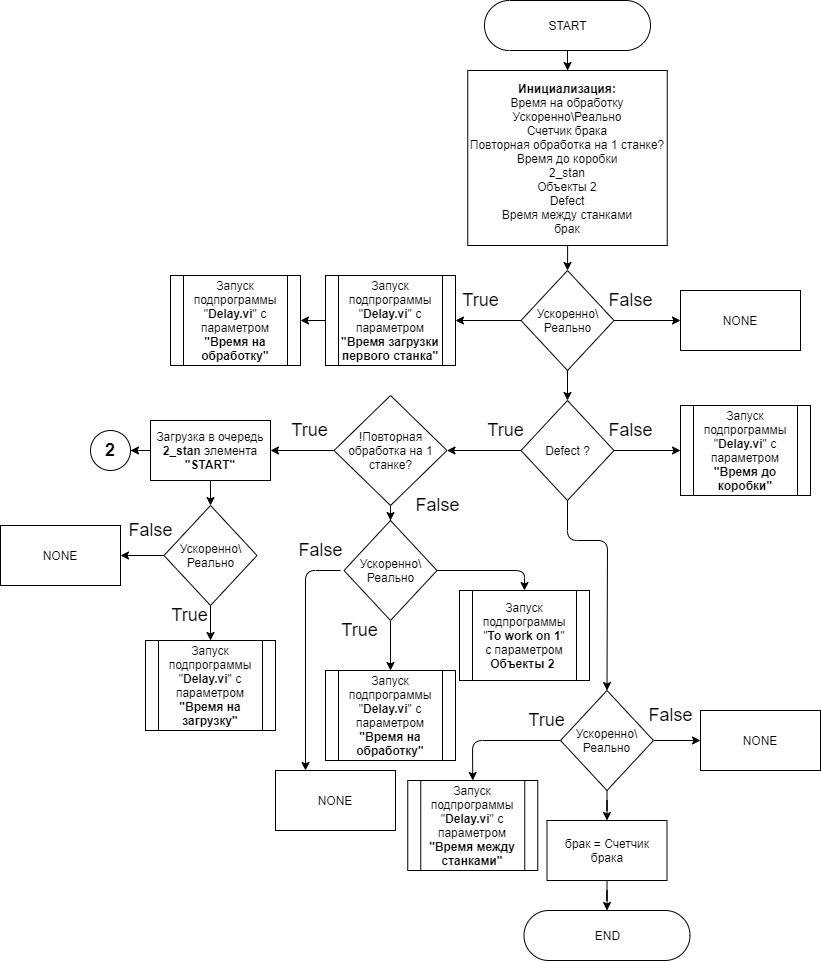


Рис. 9. Блок-схема подпрограммы Flat seque for 1st or 2nd machin

***Подпрограмма*** «**Delay»** отвечает за все задержки, использована для возможности остановки отработки ситуации без отключения конечного автомата (проблема стандартных Delay элементов).

***Подпрограмма «*To work on 1»** использована для расчета шанса брака при повторной обработке на 1 станке.

***Подпрограмма*** **«estimated time»** используется для прогнозирования времени обработки имеет 2 состояния: обработка на 2-х станках или на 1.

***Подпрограмма*** «**Calculation chance»** высчитывает шанс положительного исхода обработки, этот метод использован для упрощения работы.

***Подпрограмма* «Flat seque for 2nd machin»** является отображение этапов обработки на 2-м станке и имеет внутри все задержки, связанные со 2-м станком. Там же в ней реализован счетчик брака.

***Подпрограмма* «Flat seque for 1st or 2nd machin»** является отображение этапов обработки на 1-м станке и имеет внутри все задержки, связанные со 1-м станком. Дополнительно осуществляется проверка на повторную обработку на 1-м станке или отправка в очередь для обработки на 2-м станке.

Реализация кода из-за специфики языка представлена в приложении.

Реализация ПО представлена на рис. 10 обработка на 2-м станке и рис. 11 повторная обработка на 1-м станке.

