Практическая работа №6. Триггеры, автоматы и логические схемы с памятью.

Введение

Иногда в жизни и работе мы встречаем схемы, которые по определению невозможно описать в терминах т.н. «функционального» поведения: так, когда мы нажимаем на кнопку включения экрана смартфона, мы не можем точно сказать, что она сделает: включит экран или выключит, ведь это зависит от того, включен ли экран или нет. Точно так же, мы не можем сказать, включен экран смартфона или нет, если мы и не держим палец на кнопке — важнее **предыстория**, выраженная в виде сочетания: каково текущее состояние + каково внешнее воздействие.

Иными словами, срабатывание одного и того же входа (одной и той же логической переменной) не определяет значение выхода, а только влияет на то, каким будет **следующее** состояние и значение в контексте текущего состояния.

Определение

Конечный автомат — это математическая модель системы с памятью, поведение (выходной сигнал) которой определяется не только текущим входом, но и текущим состоянием; каждое из множеств возможных значений (входы, выходы, состояния) конечно, и сопоставляет выходным значениям состояние (или состояние + вход), а также обновляет состояние по входу и прошлому состоянию.

Формально конечный автомат задаётся кортежем

$$A = (S, I, O, \delta, \lambda, s_0),$$

где:

- S — конечное множество состояний;
- I — множество входных символов;
- O — множество выходных символов;
- $\delta : S \times I \to S$ — функция переходов, определяющая следующее состояние;
- λ — функция выходов;
- $s_0 \in S$ — начальное состояние, в котором автомат находится при «включении».

В зависимости от того, как определяется функция λ, различают два основных типа автоматов:

- **автомат Мили**: $\lambda : S \times I \to O$ — выход зависит от состояния и входа;
- **автомат Мура**: $\lambda : S \to O$ — выход зависит только от состояния.

Ключевое отличие от «функциональных» схем состоит в следующем: один и тот же вход $i \in I$ для автомата не определяет выход однозначно. Он лишь инициирует переход

$$s_{t+1} = \delta(s_t, i_t),$$

после чего выход определяется либо парой (s_t, i_t), либо новым состоянием s_{t+1}.

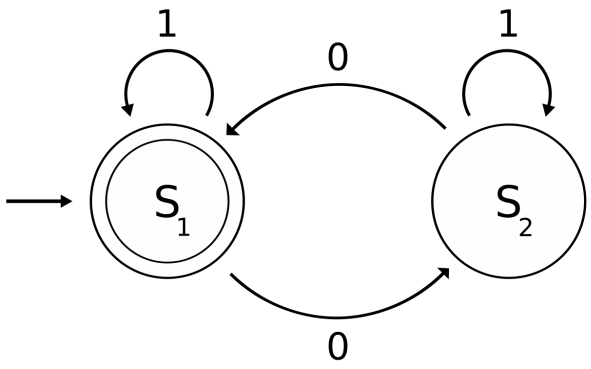
Таким образом, поведение системы описывается не функцией вида $I \to O$, а последовательностью состояний:

$$(s_0, i_0) \to s_1 \to (s_1, i_1) \to s_2 \to \ldots$$

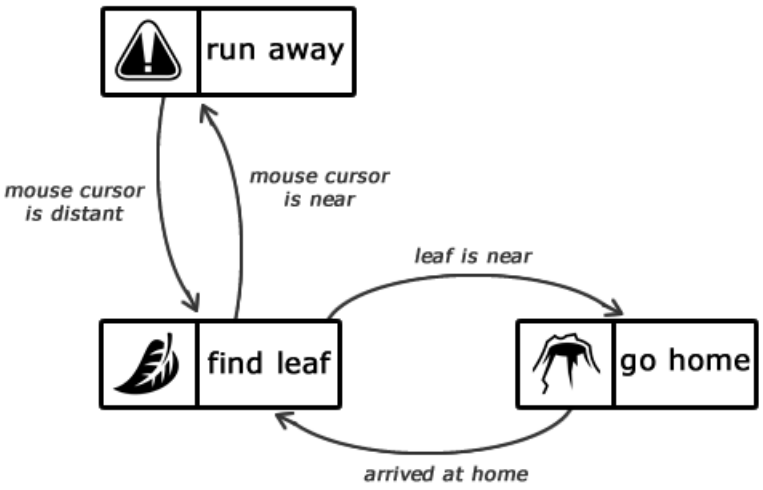
Входные, выходные сигналы, а также состояния нередко кодируются несколькими логическими переменными.

Граф состояний

Конечные автоматы можно задавать с помощью графа состояний. Например, на картинке изображен автомат в двумя состояниями s_1, s_2, и симметричными переходами по получению 0 или 1 на единственном входе.



Или так — да, с помощью конечных автоматов можно описывать поведение, скажем, игровых персонажей или объектов симуляции, например, муравья:



Использование конечных автоматов

Конечные автоматы широко используются в информационных технологиях. Finite State Machine (FSM) встречается в программировании операционных систем и интерпретаторов языков, а также при разборе текстов регулярными выражениями (где конечные автоматы строятся в ожидании той или иной последовательности символов) — то есть, конечные автоматы используются в любом текстовом редакторе для простейшей подсветки синтаксиса, а также в разработке игр для описания поведения игровых персонажей или внутриигровых механизмов.

В схемотехнике же, для реализации конечных автоматов и любых схем с памятью используются логические схемы, чьи выходы «замкнуты» обратно на часть входов, тем самым создавая обратную связь. В таблице истинности таких устройств фигурируют не только входы, но и текущее состояние выхода. Например, если у функции есть вход X и

выход Y и схема имеет память, таблица истинности схемы будет содержать по крайней мере столбцы X, Y(t) как входы, и также Y(t+1) как выход, где t и t+1 это обозначения времени в условных дискретных единицах.

Задания схемотехнические

Совет

Для выполнения заданий 1-5 рекомендуется программа Logisim, либо сайты simulator.io, либо https://logic.ly, либо https://simulator.io, допускаются другие по согласованию с преподавателем.

Задание 1. Реализация RS-триггера.

Реализуйте RS-триггер на элементах И-НЕ и ИЛИ-НЕ. Докажите наличие памяти у схемы, для этого продемонстрируйте, что при одинаковых входах возможны различные значения. Предоставьте таблицу истинности и граф переходов. Рекомендуется программа Logisim, либо сайты simulator.io, либо https://logic.ly, либо https://simulator.io, допускаются другие по согласованию с преподавателем.

1. Одинаково ли поведение RS-триггера на И-НЕ и ИЛИ-НЕ?
2. Все ли комбинации входов допустимы? Есть ли запрещенные?

Задание 2. Реализация JK-триггера.

Воспользуйтесь JK-триггером з Вашей среды моделирования либо реализуйте его сами. Докажите наличие памяти у схемы, для этого продемонстрируйте, что при одинаковых входах возможны различные значения. Предоставьте таблицу истинности и граф переходов. Если имеется что-то необычное при построении таблице истинности, обратите внимание на вопрос 2 данного задания.

1. Есть ли у него запрещенные комбинации?
2. Триггеры делят на статические и динамические по признаку управляемости по значению или его *изменению*. Каким является RS, каким является JK?

Задание 3. Реализация D-триггера.

Воспользуйтесь D-триггером из Вашей среды моделирования либо реализуйте его сами. Докажите наличие памяти у схемы, для этого продемонстрируйте, что при одинаковых входах возможны различные значения. Предоставьте таблицу истинности и граф переходов.

1. Почему этот триггер часто называют триггером копирования или триггером задержки сигнала?

Задание 4. Счетчик сигналов

На триггерах, простейших запоминающих схемах, можно строить счетчики.

Постройте вычитающий и суммирующий счетчик (ПР12 из СДО) и временную диаграмму.

Задание 5. Программируемый логический модуль

ПР14 из СДО.

Вопрос на подумать: почему мы используем счетчики, если по сути функция или преобразователи кодов это системы без памяти?

Задание 6. Описание сложного поведения

Для описания сложного поведения используется конечный автомат. Предложите конечный автомат (или несколько взаимосвязанных автоматов: сеть автоматов математически изоморфна одному автомату) и представьте в виде графа переходов.

- перечислите входные сигналы;
- перечислите выходные сигналы;
- перечислите состояния;
- задайте начальное состояние;
- постройте граф переходов.

Варианты:

1. Смартфонная кнопка включения экрана. (Указание: разделить длинное и короткое нажатие на кнопку как различные входные сигналы)
2. Светофор с кнопкой ЖДИТЕ для пешеходов
3. Игровой противник в компьютерном шутере
4. Автомат продажи напитков
5. Лифт на три этажа
6. Робот-пылесос без готовой карты и на аккумуляторе
7. Одна клетка игры «Жизнь» Конвея
8. Беспилотные колесный доставщик еды (продумать режимы работы так, чтобы он ни в кого не врезался)
9. Конечный автомат для распознавания десятичного числа: автомат принимает входную строку посимвольно и определяет, является ли она корректной записью десятичного числа: целого, числа с плавающей точкой или некорректной последовательностью символов.
10. Конечный автомат используется для управления абстрактным исполнителем Машина Тьюринга. Требуется реализовать конечный автомат для увеличения числа на 1 в одной конкретной системе счисления (например, десятичной). Предполагается, что считывающая головка уже стоит на последней цифре записи числа.