ПРИЛОЖЕНИЕ 12

ПРИМЕР ПОСТРОЕНИЮ ГРАФА ПЕРЕХОДОВ УПРАВЛЯЮЩЕГО АВТОМАТА, РЕАЛИЗУЕМОГО ПРОГРАММНО

Методику, изложенную в разд. 4.4.9, проиллюстрируем примером построения графа переходов, реализующего алгоритм управления и контроля трехпозиционным клапаном (Кл) с памятью с помощью трех кнопок без памяти. Наличие памяти в исполнительных механизмах объекта управления (клапана) позволяет снимать с них управляющие сигналы после того, как клапан откроется или закроется, сохраняя это положение.

Приведем словесное описание алгоритма управления клапаном.

1. При нажатии кнопки «Откр.» клапан начинает открываться.
2. После его открытия срабатывает сигнализатор открытого положения, зажигается лампа «Откр.» и управляющий сигнал с клапана снимается.
3. При нажатии кнопки «Закр.» клапан начинает закрываться.
4. После его закрытия срабатывает сигнализатор закрытого положения, зажигается лампа «Закр.» и управляющий сигнал с клапана снимается.
5. Если в течение 3 с клапан не откроется или не закроется, то управляющий сигнал с клапана снимается и зажигается лампа контроля «Неисправность».
6. Сброс сигнала контроля осуществляется нажатием кнопки «Разблок.».

Построение графа переходов ироводится в несколько этапов

1. Строится схема связей «источник информации—управляющий автомат—средства представления информации—исполнительные механизмы» (рис. 1).
2. Определяются устойчивые и неустойчивые состояния объекта управления, формируя его пространство состояний. Для клапана устойчивыми будут: состояния «Закрыт» и «Открыт», а неустойчивыми — «Открывается» и «Закрывается» (рис. 2).
3. Каждому состоянию клапана сопоставляется вершина в графе объекта, которая помечается десятичным номером (рис. 3).
4. Каждая вершина графа объекта через дробь с кодирующей ее цифрой помечается кортежем значений двоичных переменных, являющихся подмножеством множества X, которые формируются сигнализаторами объекта в этом состоянии (рис. 4).
5. Определяются все допустимые переходы между состояниями объекта, что отражается введением соответствующих дуг в граф объекта (рис. 5).
6. Каждая дуга и петля в графе объекта помечаются конъюнкциями неременных или их инверсий из подмножества множества Z, которые соответствуют значениям переменных, подаваемых на входы исполнительных механизмов объекта и средств представления информации (рис. 6).
7. По графу объекта строится граф переходов функционирования модели объекта (граф модели) (рис. 7).
8. Выполняется анализ технического задания с целью выявления необходимости использования функциональных элементов задержки в алгоритме управления. При их наличии корректируется схема связей, построенная на этапе / (рис. 8).
9. Каждому состоянию объекта сопоставляется состояние автомата, управляющего объектом (рис. 9).
10. Для каждой вершины графа переходов автомата (граф автомата) определяется кортеж значений всех его выходных переменных, образующих множество Z, который обеспечивает пребывание объекта в состоянии (устойчивом или неустойчивом), соответствующем рассматриваемому состоянию автомата (рис. 10).

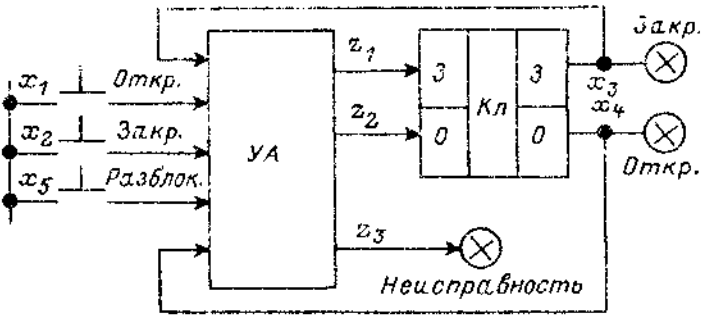

Рис. 1

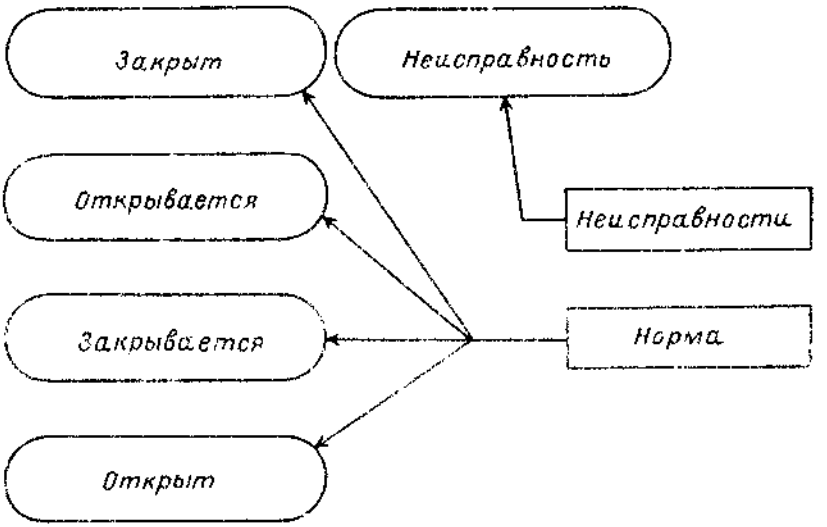

Рис. 2

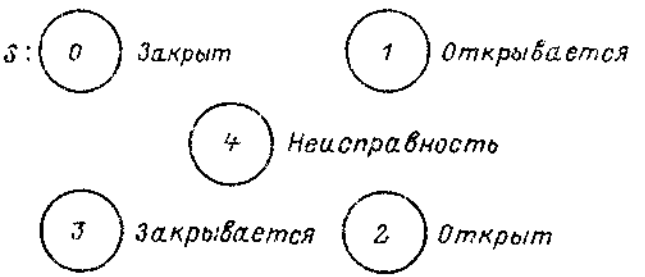

Рис. 3

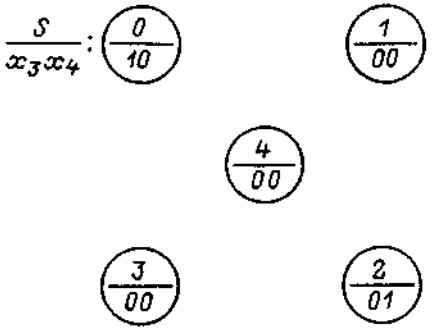

Рис. 4

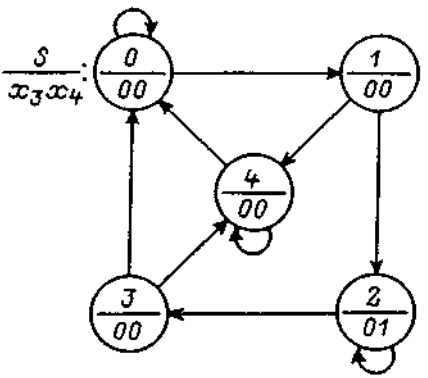

Рис. 5

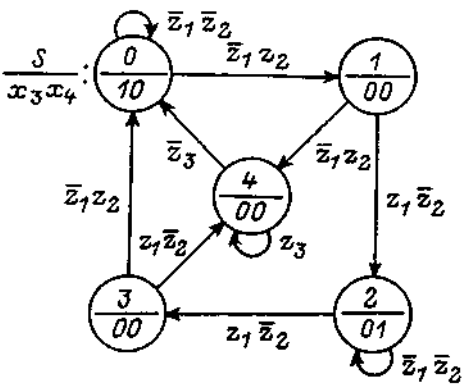

Рис. 6

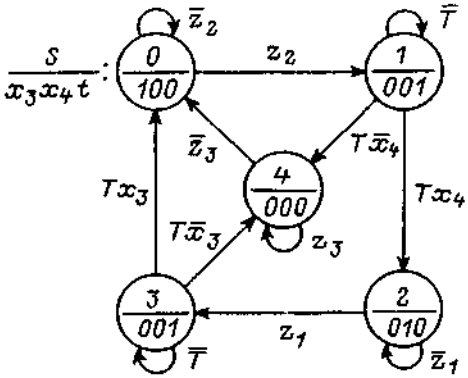

Рис. 7

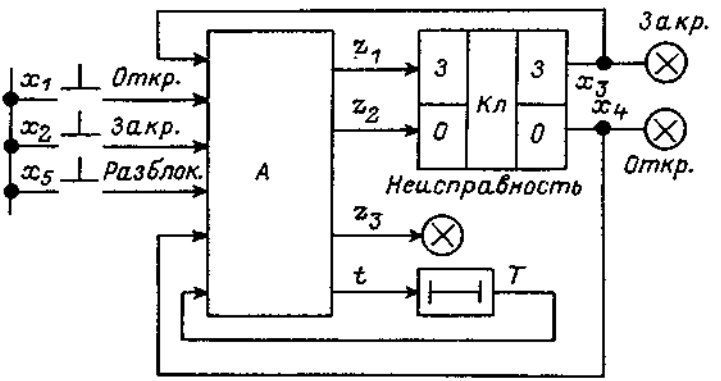

Рис. 8

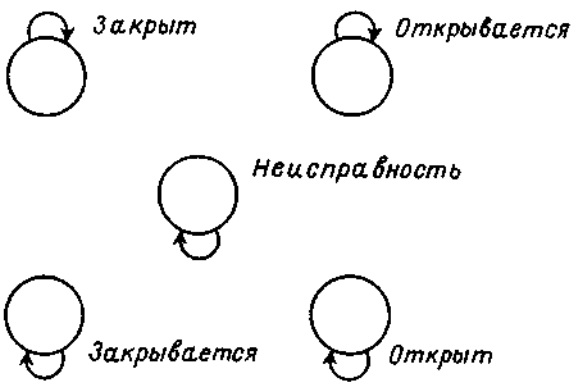

Рис. 9

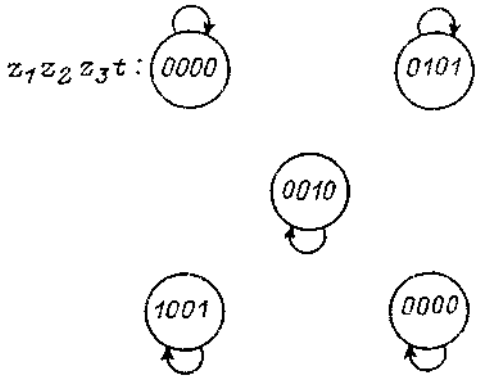

Рис. 10

11. Вершины графа автомата соединяются дугами в соответствии с соединением вершин в графе объекта (рис. 11).

!2 Каждая дуга графа автомата помечается булевой формулой, составленной из переменных множества *X* и (или) множества *T,* равенство единице которой инициирует соответствующий переход в автомате (рис. 12).

13. В граф автомата вводится дополнительная дуга для устранения возможного несоответствия начальных состояний автомата и объекта (рис. 13).

14. Дуга, дополнительно введенная в граф автомата, помечается булевой формулой, равенство единице которой инициирует переход между вершинами, соединенными этой дугой (рис. 14).

15. Для каждой вершины графа автомата ортогонализацией обеспечивается непротиворечивость переходов в другие вершины (рис. 15)

16. Осуществляется пометка петель вершин графа автомата за счет обеспечения полноты переходов из каждой вершины (рис. !6)

17. Ортогонализацией устраняется генерирующий контур, состоящий из трех вершин (рис. 17).

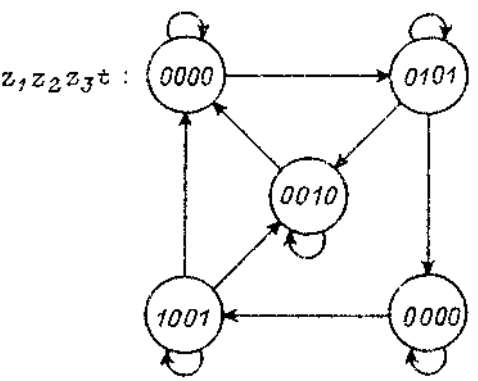

Рис. 11

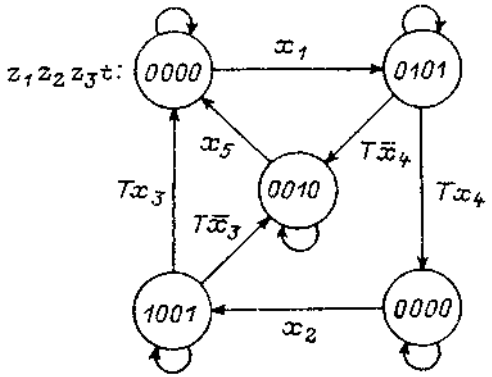

Рис. 12

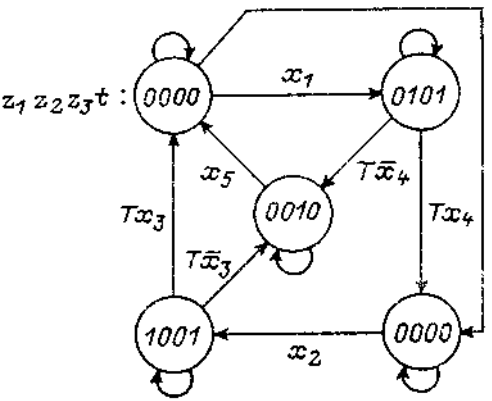

Рис. 13

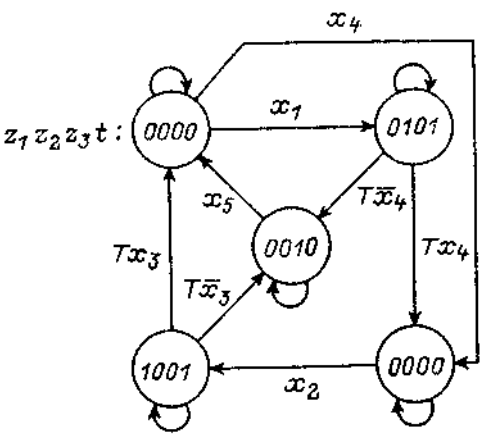

Рис. 14

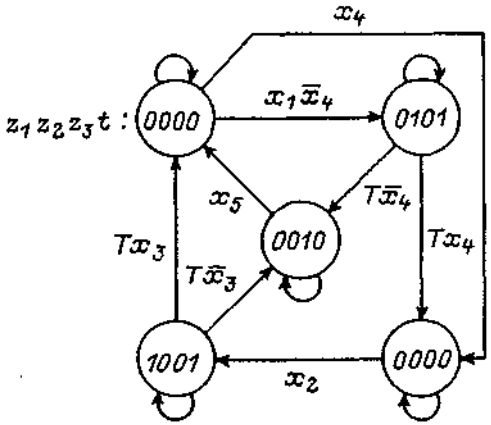

Рис. 15

В графах переходов, рассматриваемых на следующих этапах, предполагается, что выполненный анализ позволяет сделать вывод о том, что контуры в них не являются генерирующими.

18. Выполняется многозначное кодирование вершин графа автомата (рис. 18).

19. Из рассмотрения рис. 18 следует, что ГП вершины с номерами «0» и «2» могут быть совмещены (рис. 19).

В полученном графе в нулевой вершине, которая соответствует состояниям объекта «Открыт» и «Закрыт», имеет место противоречие $X_1 = X_2 = 1$. Устранение этого противоречия может привести по желанию Разработчика к построению одного из трех ГП, в каждом из которых выполнена перенумерация вершины «4» в ГП (рис. 19): граф с приоритетом сигнала закрытия (рис. 20), граф с приоритетом сигнала открытия (рис. 21), граф без приоритетов (рис. 22).

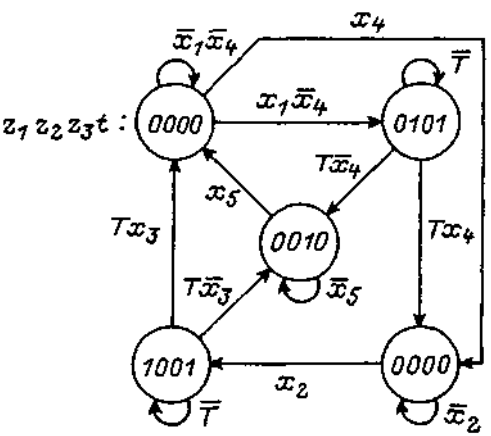

Рис. 16

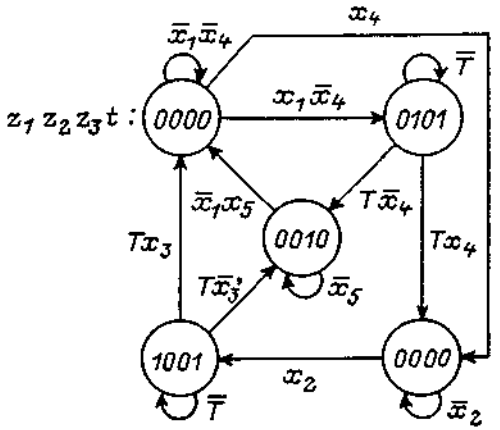

Рис. 17

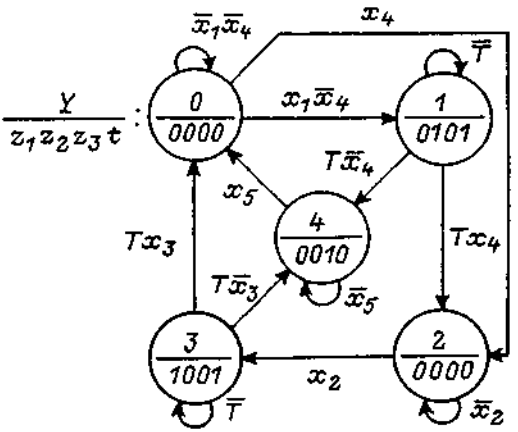

Рис. 18

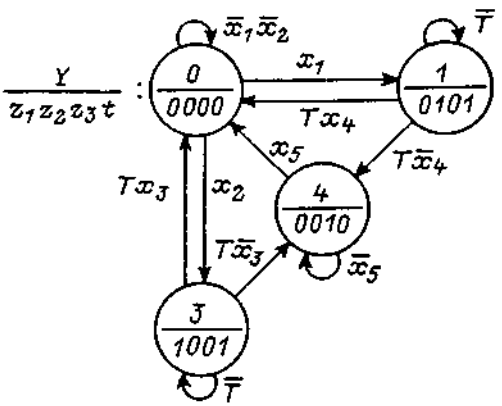

Рис. 19

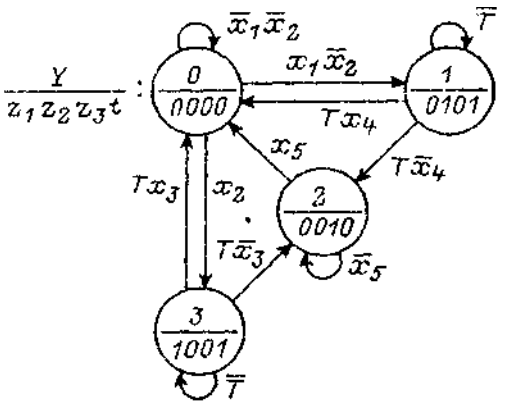

Рис. 20

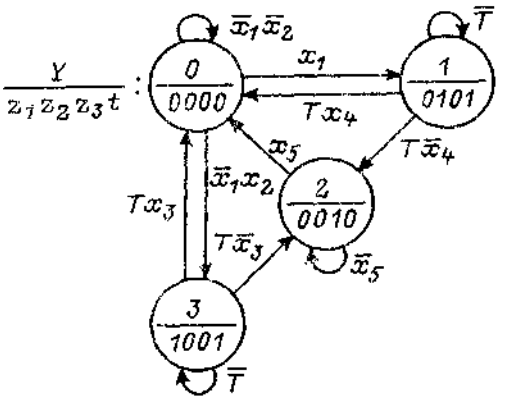

Рис. 21

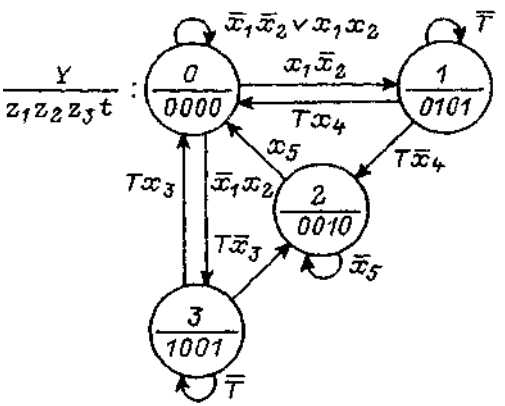

Рис. 22

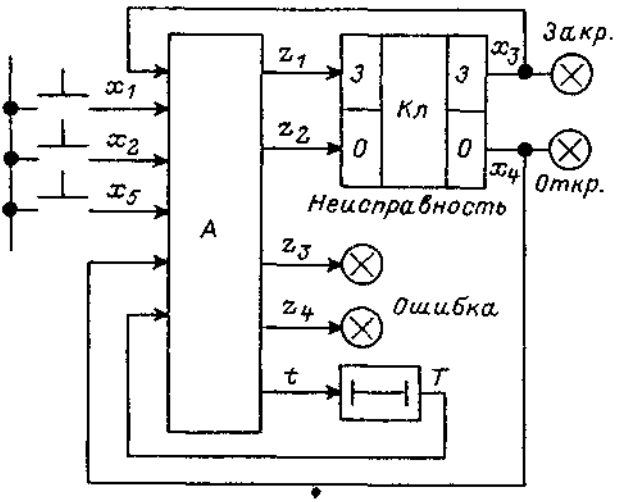

Рис. 23

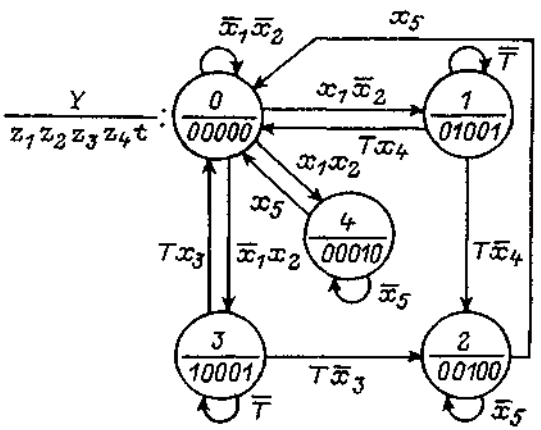

Рис. 24

20. Выполняется корректировка схемы связей, построенной на этапе *8,* с целью обеспечения возможности отражения неправильных действий Оператора.

На рис. 23 приведена откорректированная схема связей, в которую включена лампа сигнализации «Ошибка», предназначенная для фиксации такого события, как одновременное нажатие двух управляющих кнопок X_1 и X_2. Сброс сигнализации осуществляется кнопкой X_5.

Граф переходов, фиксирующий в том числе и указанное событие, приведен на рис. 24.

В заключение раздела на простом примере еще раз продемонстрируем преимущества предлагаемой технологии.

Пусть для заданного словесного описания построен ГП, который формально реализован конструкцией switch, изоморфной ГП.

Пусть это описание реализовано также и эвристически построенной функциональной схемой.

Если в исходное описание требуется внести изменение, например соответствующее введению дополнительной дуги в ГП (дуга *4 на рис. 18), то построенная указанным образом программа корректируется введением в соответствующую метку case одной строки с оператором if, в то время как эвристически построенная схема должна перепроектироваться.

Предположим теперь, что имеются откорректированная программа, изоморфная, например, ГП (рис. 18), и откорректированная функциональная схема. Рассмотрим вопрос об их проверке.

Построенная указанным образом программа является «абсолютно белым ящиком», и ее проверка может выполняться следующим образом:

— проверяется (по номерам состояний) реализация каждого перехода в ГП;

— в каждом состоянии проверяется кортеж значений выходных переменных;

— проводится сверка текста программы с ГП, которая позволяет доказать, что программа не реализует ничего лишнего.

Это исключает необходимость проведения анализа входо-выходных последовательностей.

Для эвристически построенной функциональной схемы, рассматриваемой как «относительно белый ящик», в котором состояния не определены, из-за наличия одинаковых кортежей значений выходных переменных приходится проводить весьма трудоемкий анализ входо-выходных последовательностей, который, однако, не позволяет определить: не делает ли схема чего-либо лишнего.

Из изложенного следует, что без использования технологии алгоритмизации и программирования, направленной на создание качественных программ, невозможно решить проблему полноты их проверки, так же как нельзя обеспечить контролепригодность аппаратуры, если это свойство не закладывать при ее проектировании.

Предлагаемый подход позволяет заменить тестирование программ анализом их функциональных возможностей.