7. Stateflow. Конечный автомат

StateFlow – последовательность состояний. **Statechart** – диаграмма состояний. Основные неграфические компоненты таких диаграмм – это **событие** и **действие**, основные графические компоненты – **состояние** и **переход**.

Событие – нечто, происходящее вне рассматриваемой системы, возможно требуя некоторых ответных действий. *События* считаются мгновенными (для выбранного уровня абстрагирования).

Действия – это реакции моделируемой системы на события. Подобно событиям, *действия* принято считать мгновенными.

Состояние – условия, в которых моделируемая система пребывает некоторое время, в течение которого она ведет себя одинаковым образом. Как правило, состояние соответствует промежутку времени между двумя событиями. В диаграмме переходов *состояния* представлены прямоугольными полями со скругленными углами.

Переход – изменение состояния, обычно вызываемое некоторым значительным событием. *Переходы* показываются в диаграммах переходов линиями со стрелками, указывающими направление перехода.

Конечный автомат (**finite state machine** (**FSM**)) – вариант управляемой событиями (реактивной) системы. Управляемая событиями система переходит из одного состояния (режима) в другое предписанное состояние в том случае, если условие, определяющее изменение, истинно.

7.1. Постановка задачи

Построить модель конечного автомата FA={**Q**, $\mathbf{q_0}$, **A**, **Σ**, **δ**}, используя ***Simulink*** + ***Stateflow***. Определить какие входные последовательности являются *правильными* (*допустимыми*) и проверить это на построенной модели. Входная последовательность считается правильной, если модель останавливается в допускающем состоянии. Научиться создавать и отлаживать *Stateflow*-диаграммы.

- **Q:={Zero;First}** – множество из двух состояний;
- $\mathbf{q_0}$**:=Zero** – начальное состояние: $q_0 \in Q$;
- **A={First}** – допускающее состояние;
- **Σ={1;2}** – входной алфавит;
- **δ:Q×Σ→Q** – функция перехода по правилу:

Q\Σ	Zero	First
1	First	Zero
2	Zero	Zero

Входная последовательность из **1** и **2** задается при помощи автоматического вызова **m-**файла **inpdata.m**. Результат – сообщение о корректном или недопустимом останове модели – выводится при помощи автоматического вызова **m-**файла **result.m**.

7.1.1. Создание Simulink-модели

Шаг 1. Построим дискретную *Simulink*-модель **`L0701.mdl`**.

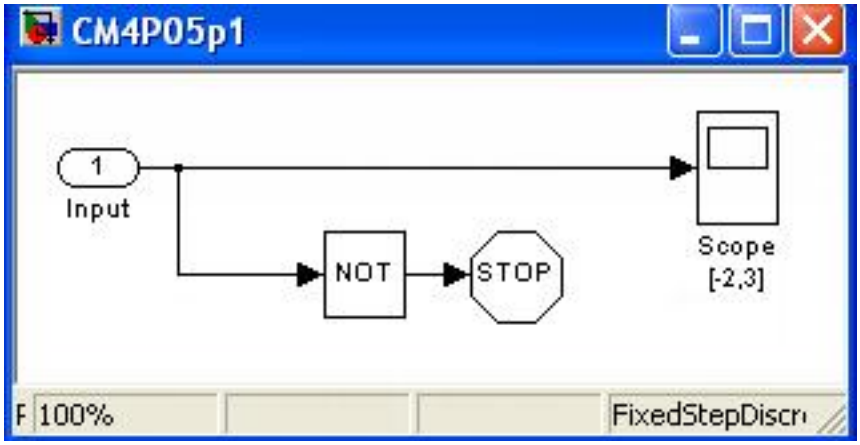

Рис.7. 1. Simulink-модель

Блок **`Sinks\Stop Simulation`** используем для досрочного завершения моделирования по концу входной последовательности (блок **`Stop Simulation`** останавливает процесс, когда на его вход подается ненулевой сигнал).

Шаг 2. Для осей блока осциллографа **`Sinks\Scope`** выберем интервал отображения **`[-2;3]`**. В локальном меню блока выберем команду **`Block Properties\Block Annotation`** и укажем этот интервал в качестве текста аннотации блока: **`[%<YMin>,%<YMax>]`**.

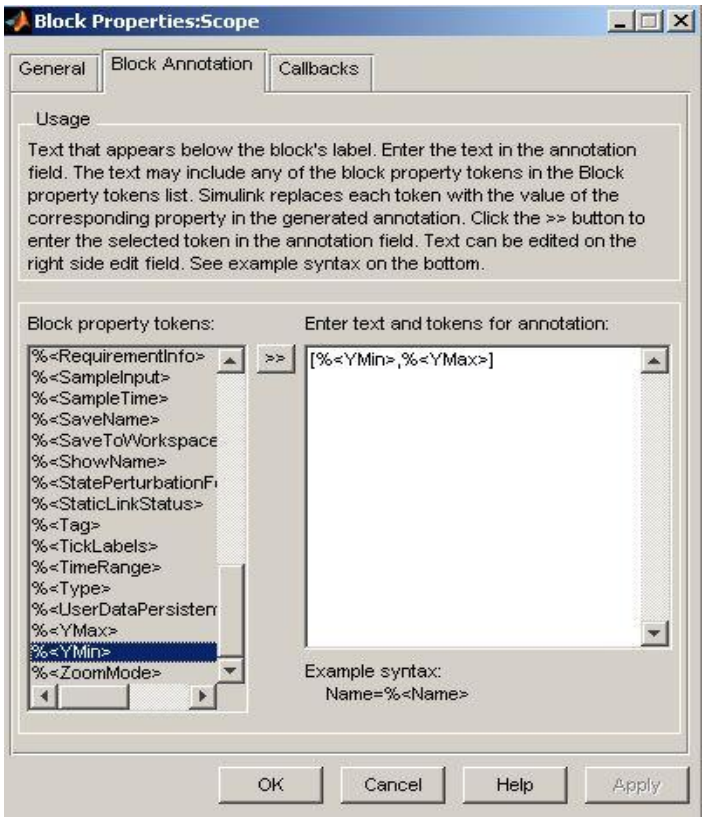

Рис.7. 2. Настройка блока Scope

Шаг 3. В блоке **`Input`** входных параметров (**`Sources\In1`**) укажем шаг по времени **`Sample time=1`**.

Шаг 4. Настроим параметры моделирования: команда главного меню **`Simulation \ Configuration Parameters`**..., закладки **`Solver`** и **`Data Import/Export`**:

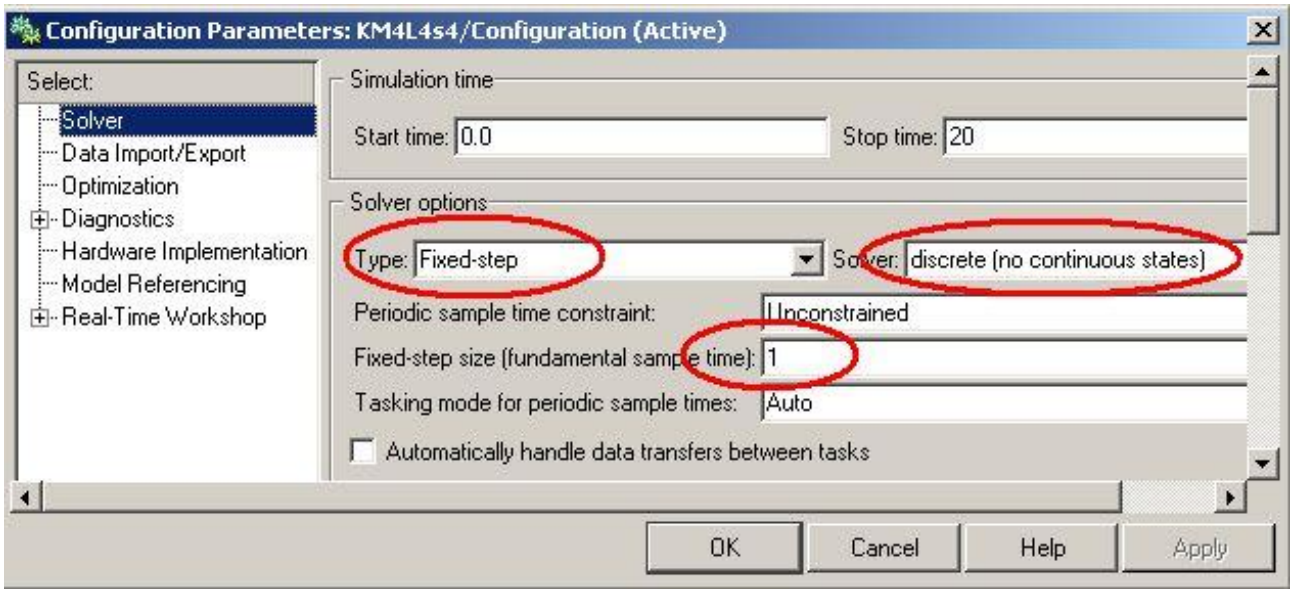

Рис.7.3. Параметры конфигурации Solver

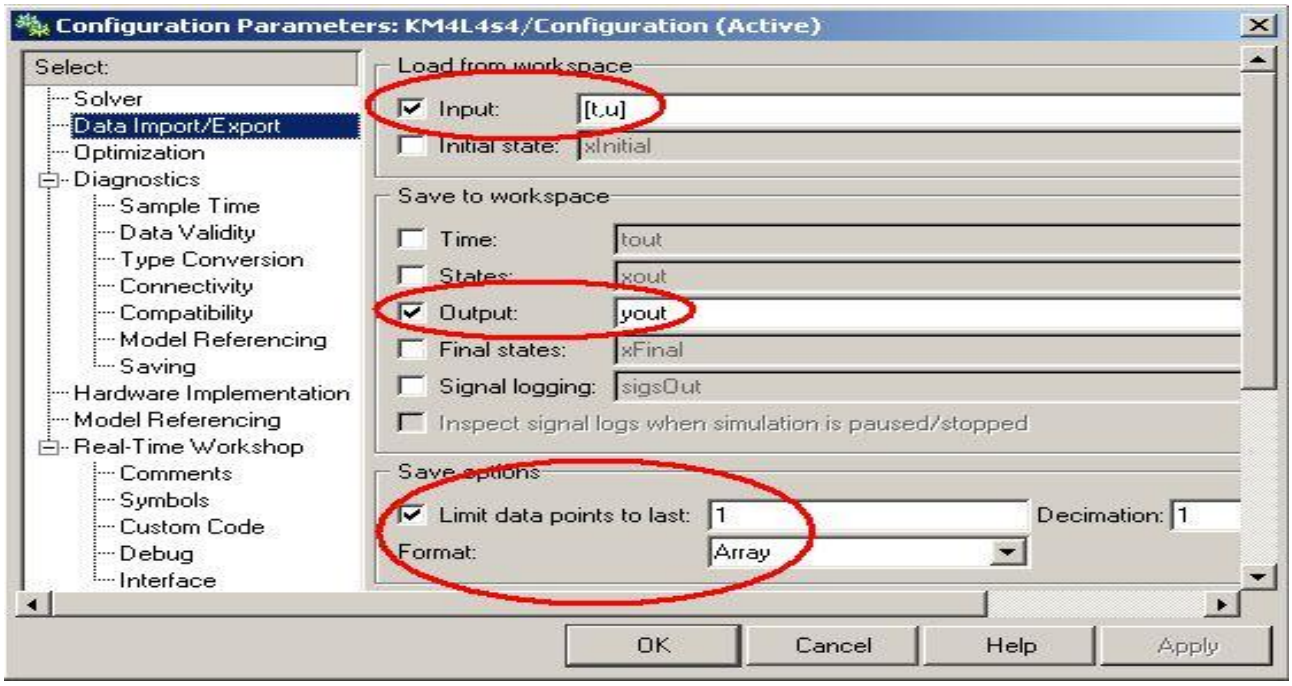

Рис.7.4. Параметры конфигурации Data Import/Export

7.1.2. Интерфейс пользователя

Шаг 5. Перед запуском модели необходимо в рабочей области **`Workspace MatLab`** подготовить входные данные для моделирования, которые будут находиться в переменных **`t`** (***отсчеты времени***) и **`u`** (***входная последовательность*** – массив, элементами которого яв-

ляются числа 1 и 2). Для нужд реализации добавим в исходную последовательность, содержащую **n** чисел **1** и **2**:

- нулевой элемент, равный **9**, для инициализации диаграммы состояний;
- последний, **n+1** элемент, равный **0**, для указания конца моделирования (для блока **Stop Simulation**).

Код оформим в виде m-файла **inpdata.m**, который следует разместить в той же папке, где находится **Simulink**-модель.

```
>> u=inputdlg('Enter a sequence of[1,2]','Input String');
>> u=[9, double(u{1})-double('0'), 0]';
>> t=[0:length(u)-1]';
```

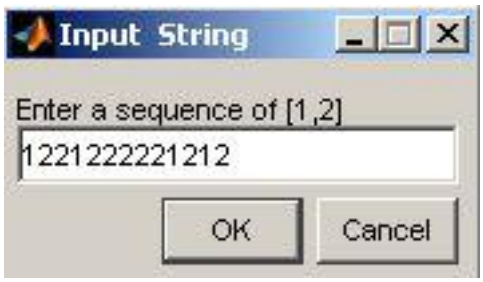

Рис.7.5. Диалоговое окно для задания входной (управляющей) последовательности

Шаг 6. **Задание для самостоятельной работы**: Самостоятельно подключите m-файл **inpdata.m** к модели таким образом, чтобы он автоматически вызывался в момент запуска моделирования.

7.1.3. Диаграмма состояний

Шаг 7. Добавим в модель блок диаграммы состояний **Stateflow\Chart** и переименуем ее в **Finite Automata**. Откроем редактор **Stateflow**-диаграмм, дважды щелкнув мышкой по данному блоку. Создание диаграммы начнем с определения её интерфейса: входных и выходных данных. Для этого откроем *Проводник* (команда главного меню **Tools\Explore** графического редактора **Stateflow**-диаграмм).

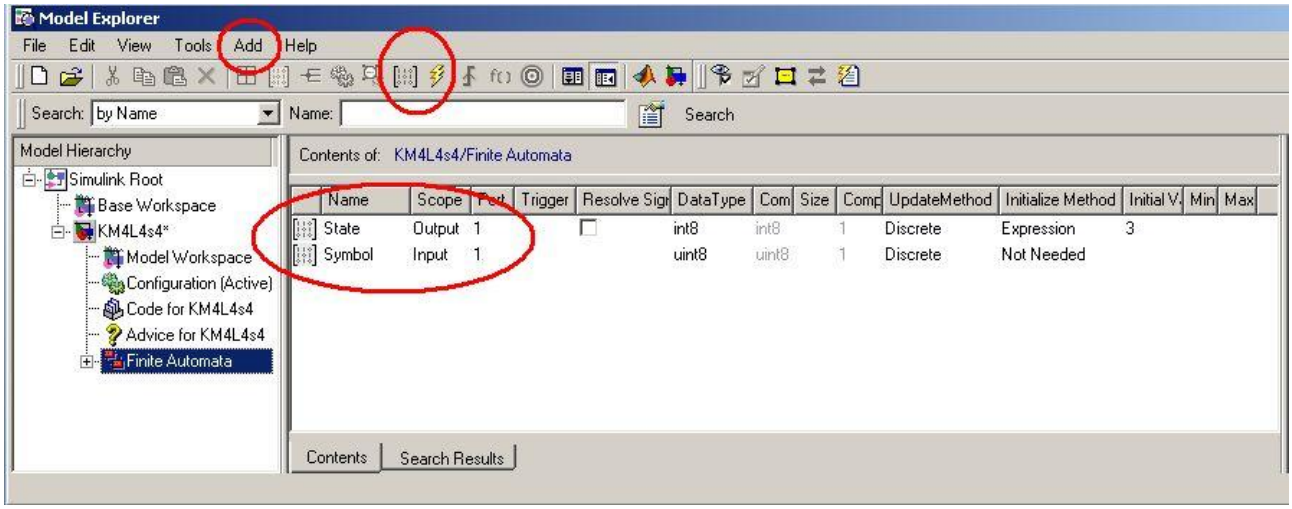

Рис.7.6. Stateflow Explorer. Добавление переменных

Выберем в дереве **`Model Hierarchy`** диаграмму **`Finite Automata`**, и командой главного меню **`Add\Data`** добавим одну входную переменную **`Symbol`** и одну выходную – **`State`**. Входная переменная **`Symbol`** будет содержать очередной символ входной последовательности, а выходная переменная **`State`** будет возвращать в **`Simulink`** номер состояния, в котором в данный момент времени находится конечный автомат. Дважды щелкнув мышкой на пиктограмму данного (левый край строки), настроим параметры обеих переменных:

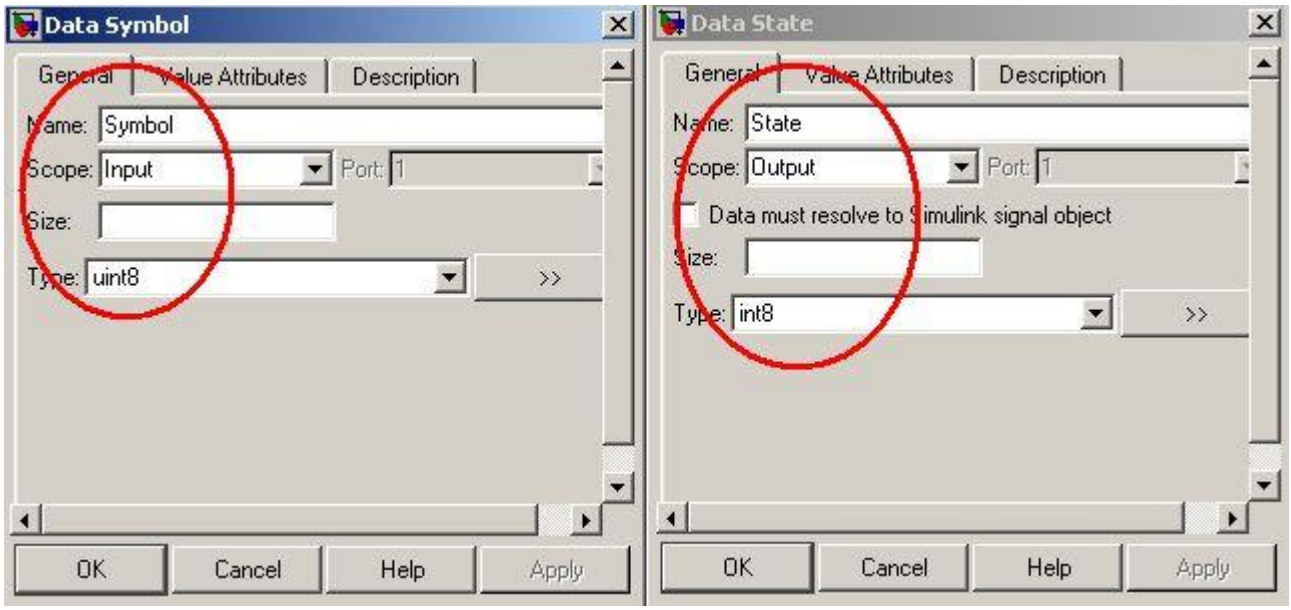

Рис.7.7. Настройка свойств данных Symbol и State

Шаг 8. Вернемся в редактор диаграмм и создадим диаграмму состояний, соответствующую заданному в условии задачи конечному автомату. Для наглядности отображения сигнала и состояния в общем графике мы будем указывать номер состояния неположительными числом.

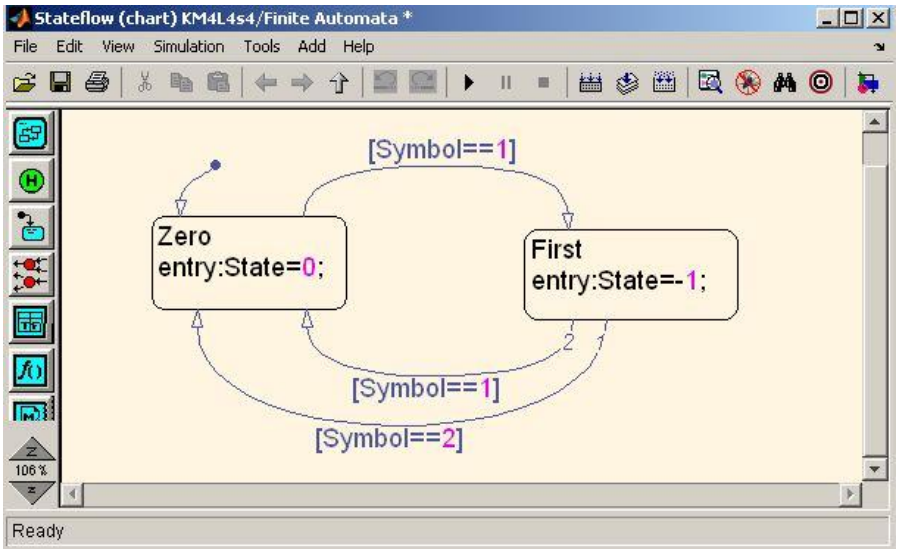

Рис.7.8. Диаграмма состояний и переходов модели

Шаг 9. Вернувшись к **Simulink**-модели, добавим в нее один выходной порт **Result** (блок **Sinks\Out1**) для передачи выходного сигнала в рабочее пространство **Workspace MATLAB**.

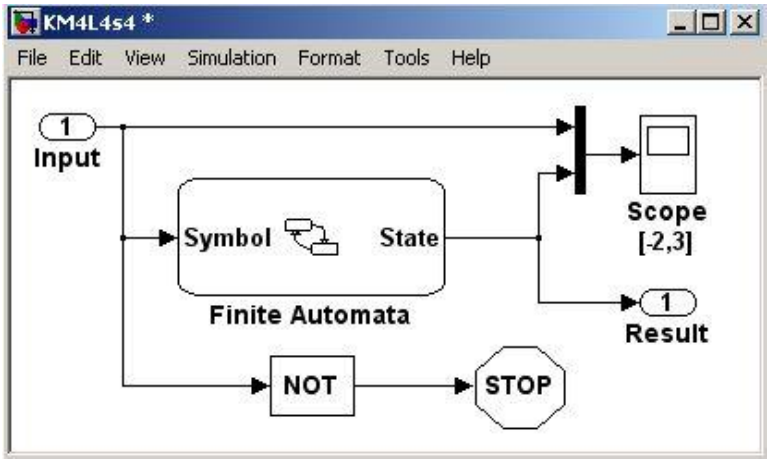

Рис.7.9. Модель

7.1.4. Интерфейс пользователя (продолжение)

Шаг 10. Учитывая сделанную в **Шаге 4** настройку параметров моделирования, **Simulink** будет сохранять последнее состояние модели в **MatLab**-переменной **yout**. Следующий фрагмент **MatLab**-кода позволяет вывести сообщение о результатах моделирования:

```
>> switch yout, …
>>     case 0, …
>>         msgbox('Not accepted','Result','error','modal'), …
>>     case -1, …
>>         msgbox('Correct','Result','none','modal'), …
>> end
```

Рис.7.10. Сообщение о результате моделирования

Задание для самостоятельной работы: Оформите этот код в виде **m**-файла **result.m** и подключите его к модели таким образом, чтобы он автоматически вызывался в момент останова моделирования.

7.2. Нахождение НОД двух чисел

Модель **L0702.mdl**

Самостоятельно реализуйте в виде **Simulink-Stateflow-**модели алгоритм по нахождению наибольшего общего делителя двух натуральных чисел, представленный в виде блок-схемы на Рис.7.11.

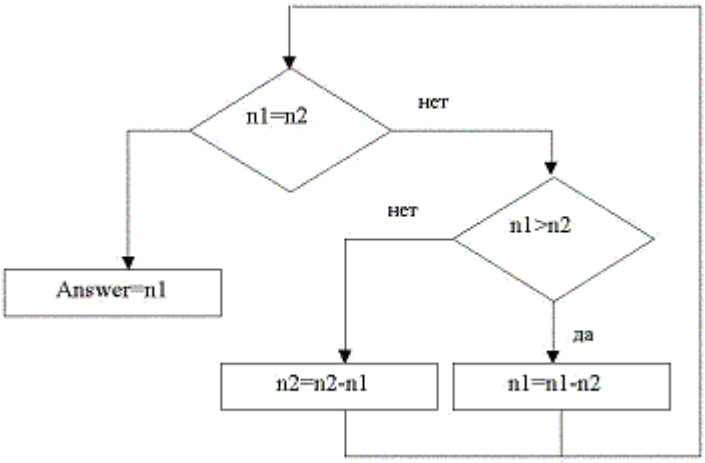

Рис.7.11. Алгоритм нахождения НОД

Simulink-моделью работы этого алгоритма может служить показанная на следующем рисунке модель.

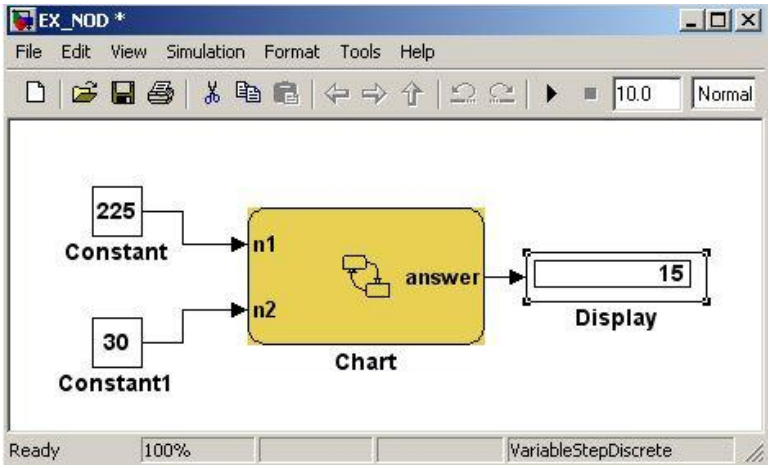

Рис.7.12. Simulink-модель нахождения НОД двух натуральных чисел