УДК 681.324
И.О. Турега, аспирант
Севастопольский национальный технический университет
Университетская, 33, г Севастополь, Украина, 99053
E-mail: kvt@sevgtu.sebastopol.ua

ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ РАБОТЫ УЗЛОВ МИКРОКОНТРОЛЛЕРНЫХ СЕТЕЙ

Проведен обзор подходов к функциональному моделированию работы сетевых узлов на базе микроконтроллерных систем. В качестве примера использована модель сетевого узла, являющегося частью однонаправленного кольца.

Сетевые структуры стали одними из наиболее распространенных технических систем в современном мире. Данные системы используют управляющие устройства на базе универсальных или специализированных процессоров для высокоскоростных, и микроконтроллеры – для встраиваемых решений.

В процессе разработки систем управления и автоматизации на базе микропроцессорных и микроконтроллерных сетей встает задача моделирования процесса информационного обмена – протокола передачи данных. Моделирование протоколов возможно с помощью построения функциональных моделей. Функциональные модели для микропроцессорных сетей могут основываться на различных подходах как к построению моделей, так и инструментах и методах описания объектов.

Целью данной статьи является рассмотрение подходов и программных средств, используемых для построения функциональных моделей, а также автоматизации процесса моделирования работы микроконтроллерных узлов и сетей.

Процесс информационного обмена представляет собой совокупность состояний сетевых узлов, сетевых данных, протокольных событий и переходов между состояниями. Под состоянием узла микроконтроллерной сети будем понимать совокупность значений аппаратных регистров и буферов, а также программных структур данных, отвечающих за сетевой обмен. Состояние может рассматриваться как для конкретного интерфейса передачи данных, так и для узла в целом. Рассмотрение общего состояния узла возможно, так как микроконтроллер функционирует в однозадачном режиме с прерываниями абсолютного, либо относительного приоритетов. Следовательно, в определенный момент времени состояние микроконтроллера может быть определено однозначно.

Сетевые данные включают в себя служебную и пользовательскую информацию на различных уровнях протоколов. Служебные данные включают в себя адреса, поля, контролирующие целостность пользовательских данных и в целом, специализированные команды интерфейсам.

Под протокольными событиями понимаются изменения в управляющих данных и регистрах сетевых интерфейсов, произошедшие вследствие работы системы прерываний, либо действий пользователя системы. Совокупность событий определяют переходы между состояниями сетевых узлов.

Рассмотрим ряд функциональных моделей микроконтроллерных сетей, использующих различные математические аппараты и программные средства. В качестве примера такой сети используем однонаправленное кольцо с контролем целостности данных. Приведем словесное описание алгоритма работы узла сети (первичную спецификацию).

1. Ожидается поступление пакета из сети, либо данных от пользователя для передачи.
2. По прибытию пакета анализируется его целостность:
   - 2.1. Для целого пакета анализируется адрес получателя:
     - 2.1.1. Если адрес совпал с адресом узла, то извлекаются данные и передаются пользователю.
     - 2.1.2. Иначе анализируется адрес отправителя.
       - 2.1.2.1. Если адрес отправителя не совпал с адресом узла – пакет в неизменном виде передается далее (так как он еще не достиг получателя, и не вернулся к отправителю).
       - 2.1.2.2. Иначе пакет игнорируется (не был принят адресатом, но обошел все кольцо и должен быть уничтожен).
   - 2.2. Для поврежденного пакета анализируется адрес отправителя.
     - 2.2.1. Если адрес отправителя совпал с адресом узла, то выполняется повторное формирование ранее переданного пакета, и он отправляется повторно.
     - 2.2.2. Если адрес отправителя не совпал с адресом узла, то пакет в неизменном виде передается далее.
3. По команде пользователя формируется пакет, устанавливаются адреса, вычисляется контрольное поле. Далее пакет отправляется в сеть.

Наиболее простым и распространенным способом описания принципа работы сетевого узла является модель конечного автомата, представленная с помощью графа. В этой модели состояния сопоставляются вершинам графа, переходы – дугам. Дуги взвешиваются логическими выражениями, активирующими переходы. Данные выражения определяются комбинациями протокольных событий.

Для создания функциональных моделей, отражающих структурированное изображение функций системы или среды, а также информации и объектов, связывающих эти функции, может применяться методология стандарта IDEF0 (Integration Definition For Function Modeling). Основной концептуальный принцип IDEF0 состоит в представлении любой анализируемой системы в виде набора взаимосвязанных и взаимодействующих графических блоков, соответствующих функциям объекта моделирования. Связи и взаимодействия между блоками на диаграмме IDEF0 описываются дугами, отражающими входы функций (стрелки слева от блоков), выходы (справа), управляющие сигналы (сверху) и механизмы выполнения функций (снизу). Основной принцип, используемый при построении моделей – принцип декомпозиции (детализации). Данный принцип позволяет пошагово раскрывать содержание функций, начиная с единого блока – диаграммы уровня A-0, представляющей весь моделируемый объект, и заканчивая необходимым уровнем детализации для каждой из минимально разделимых функций подсистем. При этом дуги могут быть общими для различных уровней детализации [1].

IDEF0 диаграммы используются для концептуальной проработки функционального наполнения системы, подход интересен в основном как механизм описания системы. В процессе моделирования координируется работа экспертов, авторов, читателей и тех, кто принимает окончательную версию модели системы. Процесс моделирования включает в себя этапы выбора цели моделирования, точки зрения, сбора информации, непосредственного построения диаграмм, их согласования, утверждения и использования [1].

Данная методология поддерживается несколькими программными средами, однако эти продукты коммерческие и не имеют ознакомительных версий. Основной причиной такого положения дел является область использования диаграмм – построение моделей автоматизированных систем управления предприятиями и производственными процессами.

На рисунке 1 изображен вариант описания системы передачи данных с помощью спецификации IDEF0. Данная диаграмма представляет собой уровень детализации, обобщающий функции сетевого обмена узла микроконтроллерной сети. На диаграмме выделены функции приема и передачи сетевых данных, определяемые механизмами сетевых интерфейсов и протоколами передачи данных. Протоколы обработки информации являются базовыми механизмами для функций анализа информации и обработки команд пользователя.

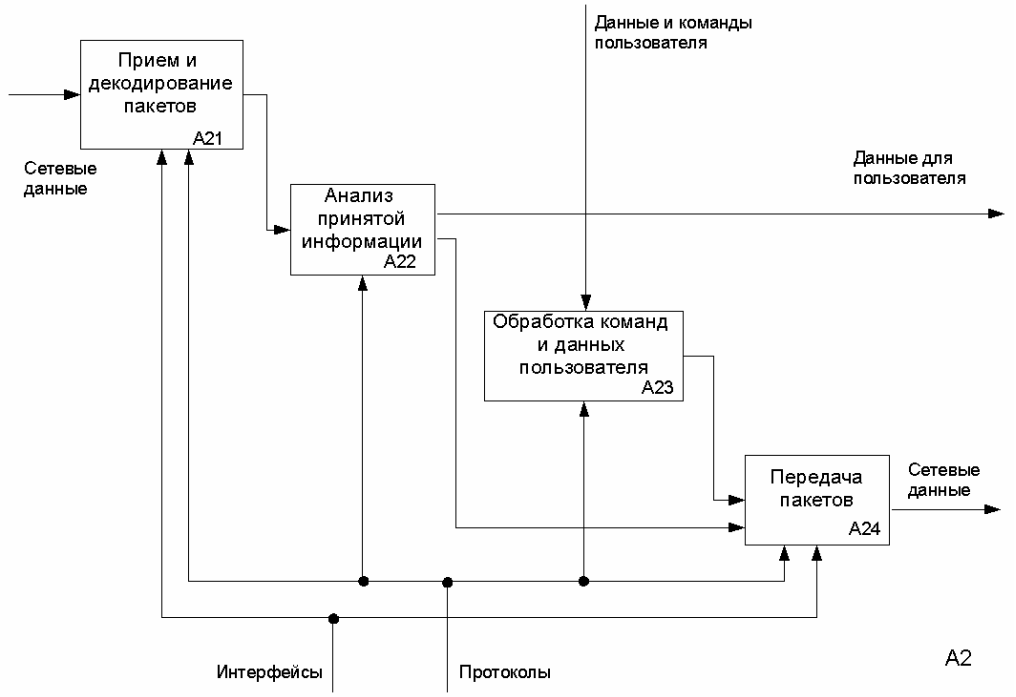

Рисунок 1 – Диаграмма сетевых функций

Для разработки событийно управляемых моделей может быть использована среда Stateflow – интегрированный с MATLAB инструмент проектирования и моделирования сложных систем. Stateflow позволяет интерактивно разрабатывать модели систем, используя представление с помощью диаграмм

состояния. Диаграммы Stateflow отражают графическое представление конечного автомата. В качестве базовых конструктивных блоков моделируемой системы используются состояния и переходы. Кроме того, в рамках модели могут внедряться события, данные, иерархические конструкции с последовательным и параллельным выполнением. Переходы разделяются на условные и безусловные, могут определяться историей работы модели, а также быть выполняемыми по умолчанию [2].

Stateflow предоставляет возможности настройки временных параметров моделирования, установки источников входных и назначения выходных сигналов, анимационной подсветки активных переходов и состояний при моделировании диаграмм. Благодаря этому, среда становится удобным инструментом в разработке и проверке моделей сложных систем с множеством состояний, переходов и событий.

Кроме того, используя возможности Simulink, диаграмма Stateflow может быть интегрирована в любую сложную систему обработки сигналов. При этом могут использоваться источники сигналов Simulink для входов диаграмм, средства отображения и анализа значений выходов, средства преобразования сигналов и другие многочисленные инструменты.

На рисунке 2 показана модель рассматриваемого сетевого узла в нотации Stateflow. Диаграмма имеет переход по умолчанию в начальное состояние Init. Базовым для модели является состояние ожидания Wait_, соответствующее пункту 1 описания системы. В состояниях выполняется изменение внутренних переменных модели, позволяющих подсчитать количество принятых, отправленных и маршрутизированных пакетов.

Непомеченные переходы являются безусловными. Остальные переходы выполняются по логическим условиям, которые определяются внешними переменными. Данные переменные в реальной системе будут определяться содержимым принятых пакетов и фактами наступления протокольных событий. В моделируемой системе переменные используют источники случайных сигналов.

В диаграмме присутствую два составных состояния: Packet_analyze и Prepare_packet, отражающие этапы подготовки пакетов и их анализа соответственно. Состояние подготовки пакетов содержит последовательно сменяемые состояния, для них определен переход по умолчанию в начальное состояние. Состояние анализа пакета содержит параллельно наступающие состояния анализа составных частей пакета, поэтому перехода по умолчанию они не требуют. Модель не имеет конечного состояния, время ее работы ограничивается временем моделирования.

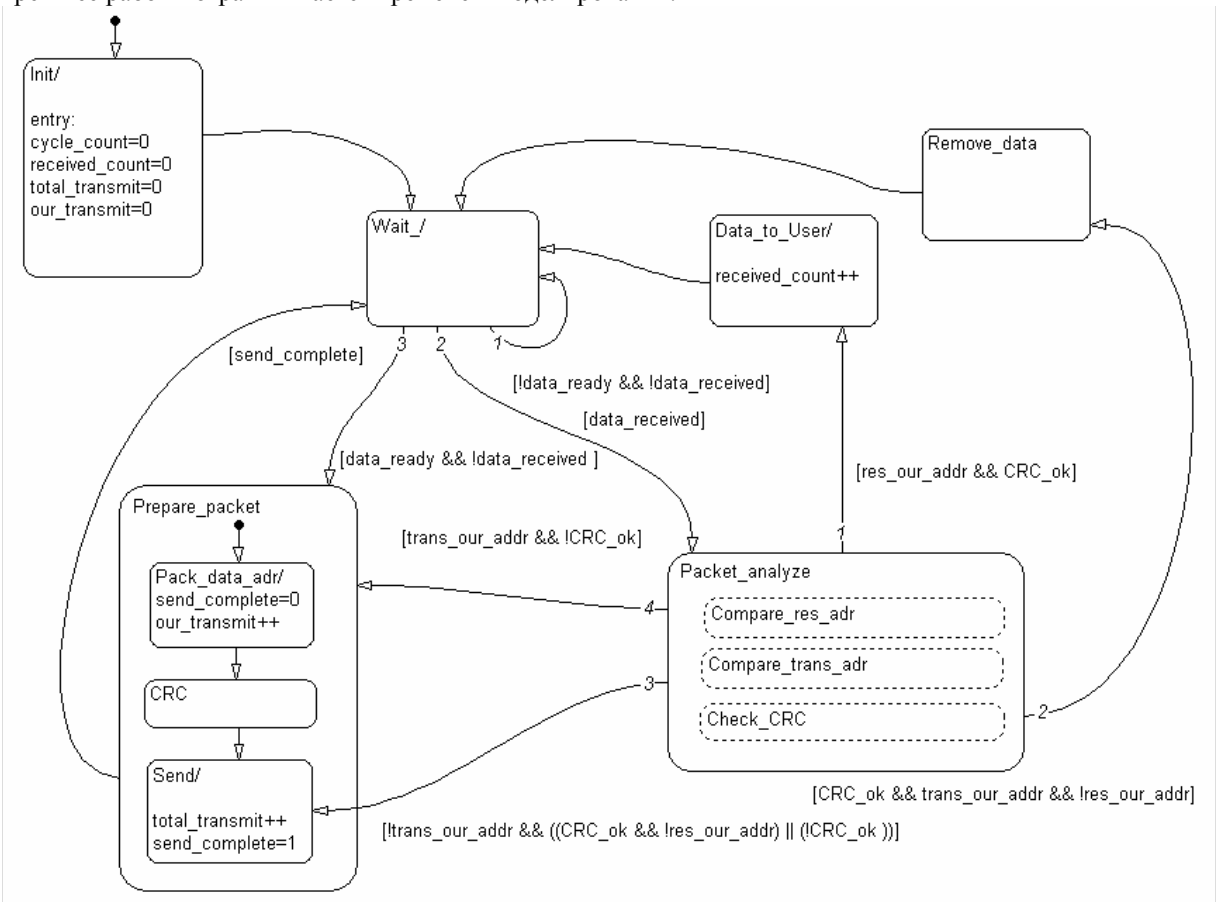

Рисунок 2 – Диаграмма состояний сетевого узла

Все предложенные способы представления функциональных моделей являются графическими. Для сложных, структурированных систем описанные диаграммы объемны, либо обладают высокой степенью вложенности, что вызывает затруднения их отображении и восприятии. Альтернативным методом может выступать представление совокупности событий, действий и данных в табличном виде. Такой метод использован при описании функционирования сетевых узлов с помощью таблиц протокольных событий [3].

Представление с помощью таблиц предполагает выделение непересекающегося множества наборов событий. Событиям сопоставляются строки верхней части таблицы, наборам данных событий – столбцы. Если событие учитывается в данном наборе как необходимое, оно отмечается единицей, если событие не должно происходить – нулем. Также событие может игнорироваться. Под строками, соответствующими событиям расположены строки действий, выполняемых при истинности наборов событий. В столбцах, соответствующих наборам событий, проставляются в строках числа, отражающие порядок выполнения действий. Аналогичным образом могут быть описаны используемые структуры данных. Действия могут быть сложными. Такие действия представляются с помощью подчиненных таблиц аналогичной структуры.

В таблицах 1 и 2 представлены множество событий и действия, соответствующие рассматриваемому примеру. В таблице 1 выполняется анализ событий приема и передачи пакетов (пункт 1 описания системы). Прием пакета (e_1) является более приоритетным событием. При получении команды и данных для отправки в сеть выполняются действия, соответствующие пункту 3 описания. В случае приема пакета выполняется сложное действие a_1, раскрытое в таблице 2. Здесь анализируются события, соответствующие пункту 2 описания системы – проверяются поля пакета и на основании полученных данных выполняются последовательности действий из множества [a_{11}: a_{16}].

Таблица 1 – Основной цикл

	События		
e_1	получен пакет	1	0
e_2	поступили данные для отправки	X	1
	Действия		
a_1	анализировать пакет (таблица 2)	1	
a_2	упаковать данные и адреса		1
a_3	вычислить и упаковать контрольную сумму		2
a_4	отправить пакет		3
a_5	возврат	2	4

Таблица 2 – Анализ пакета

	События						
e_{11}	адрес получателя совпал с адресом узла	1	1	0	0	x	x
e_{12}	адрес отправителя совпал с адресом узла	1	0	0	1	1	0
e_{13}	результат проверки целостности	1	1	1	1	0	0
	Действия						
a_{11}	извлечь данные и передать пользователю	1	1				
a_{12}	уничтожить (игнорировать) пакет	2			1		
a_{13}	упаковать данные и адреса					1	
a_{14}	вычислить и упаковать контрольную сумму					2	
a_{15}	отправить пакет			1		3	1
a_{16}	Возврат	3	2	2	2	4	2

Таким образом, предложен ряд подходов к построению функциональных моделей для узлов сетей передачи данных. Каждый подход имеет свои преимущества и недостатки. В частности использование методологии IDEF0 позволяет выполнять концептуальное проектирование систем, однако отсутствие доступных систем моделирования перекладывает на проектировщика полную ответственность за интерпретацию моделей.

Использование Stateflow обладает широкими возможностями в плане моделирования событийных систем, благодаря наличию мощного инструмента, позволяющего автоматизировать процесс моделирования. Однако данный инструмент коммерческий, кроме того, довольно сложные модели затруднительны для восприятия вследствие изобилия структурных единиц и их типов.

Подход с использованием таблиц протокольных событий позволяет описывать событийно ориентированные системы любой сложности в доступной для восприятия форме. Однако отсутствие

инструментария для построения и анализа моделей не позволяет в полной мере использовать данный подход.

В рамках дальнейших исследований ставится задача детального анализа использования таблиц протокольных событий и возможном проектировании соответствующего инструмента, автоматизирующего процесс построения моделей. Кроме того, предполагается рассмотрение подходов и технических средств создания структурных моделей микроконтроллерных сетей автоматизированного управления.

Библиографический список

1. Функциональное моделирование на базе стандарта IDEF0. Учебный курс. — Минск: Software development & bisiness consulting, 2002. — 35 с.

2. Нури А.Ж. Моделирование, анализ и оптимизация протокола TCP / А.Ж. Нури, Д.С. Ндуесо, Б.Б. Моэз // Сборник работ магистрантов Донецкого национального технического университета. — Донецк: ДонНТУ, 2002. — Вып. 1. — С. 927–937.

3. Апраксин Ю.К. Спецификация сетевых протоколов средствами языка таблиц событий / Ю.К. Апраксин // Вестник СевГТУ. Сер. Информатика, электроника, связь: сб. науч. тр. — Севастополь, 2001. — Вып. 31. — С. 4–8.

Поступила в редакцию 1.07.2008 г.