УДК 681.5:656.257

М.Л. МАЛИНОВСКИЙ, И.А. ФУРМАН, А.Ю. АЛЛАШЕВ, С.Я. БОВЧАЛЮК

Харьковский национальный технический университет сельского хозяйства им. П. Василенко, Украина

МЕТОДЫ ПРОЕКТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ПЛИС-КОНТРОЛЛЕРОВ НА ТАБЛИЧНОМ ЯЗЫКЕ CYCLOGRAF

Рассмотрены элементы и основные конструкции технологического языка программирования ПЛИС-контроллеров CycloGraf, а также структурный и цикломатный методы описания на этом языке цифровых устройств.

технологическое визуальное программирование, табличный язык CycloGraf, цикломатное проектирование ПО, структурное проектирование ПО

Введение

Современные системы проектирования ПО для ПЛИС поддерживают несколько возможных вариантов описания цифровых устройств. Наиболее распространенным и, одновременно, универсальным инструментом для решения этих задач является применение текстовых языков описания аппаратуры: VHDL, AHDL, VerilogHDL. Также широко используются графические редакторы, позволяющие наглядно объединять отдельные функциональные узлы в единое цифровое устройство.

Кроме того, некоторые простые устройства могут быть реализованы в редакторе временных диаграмм. Нет никаких сомнений в том, что перечисленные инструменты позволяют реализовать сколь угодно сложные цифровые устройства, способные разместиться на выбранном кристалле ПЛИС. Вместе с тем, структурное и поведенческое описание проектируемых устройств перечисленными средствами ориентировано на пользователей с высокой квалификацией и требует значительных усилий даже при описании относительно несложных проектов. Таким образом, разработка программного обеспечения для ПЛИС-контроллеров превращается в сложный многоитерационный процесс с участием специалистов различного профиля. Наибольшую проблему при этом представляет взаимодействие технолога и программиста. Последние, как правило, не являются носителями единой терминологии, да и вообще обладают различными стилями мышления, что никак не может способствовать их взаимопониманию. Какого результата мы можем ожидать от совместной деятельности двух специалистов различного профиля, один из которых ничего не понимает в проблемах другого: технолог не в состоянии прочесть программный код, а программист не в состоянии постичь тонкости технологии производства? С этих позиций совершенно естественно выглядит соотношение затрат на внедрение систем автоматизации, где разработка ПО порой достигает 80%.

Для усовершенствования проектирования управляющих программ для промышленных систем логического управления авторами была предложена TVP-технология – технологическое визуальное программирование [1]. Данная технология позволяет непосредственно инженеру-технологу, как носителю экспертных знаний об объекте автоматизации, в естественной для него форме задать алгоритм управления на высоком уровне абстракции в виде технологической табличной модели. В рамках данной технологии по разработанной и отлаженной табличной модели управления технологическим оборудованием специализированный транслятор автоматически генерирует программное обеспече-

ние на выбранном языке программирования, например на языках описания аппаратуры.

Ключевой задачей, которая решалась при создании TVP-технологии являлась разработка эффективного языка программирования (в дальнейшем получившего название **CycloGraf**), ориентированного на пользователя, не имеющего навыков разработки программного обеспечения. При этом авторы поставили перед собой задачу «минимум»: обеспечить уровень наглядности программного кода, доступный технологу, не имеющему навыков программирования; и задачу «максимум»: вообще исключить программиста из технологического процесса подготовки управляющих программ и предоставить функции разработки ПО непосредственно технологу, снабдив его соответствующими инструментами. Задача «минимум» достигается в методе структурного проектирования ПО, задача «максимум» - в методе программирования на основе технологических циклограмм (цикломатного проектирования ПО). Оба метода поддерживаются разработанным языком **CycloGraf**.

Элементы и конструкции языка CycloGraf

Концепция разработки языка **CycloGraf** базируется на трех основных положениях, которые коротко можно обозначить следующим образом: простота и наглядность, психологическая естественность, минимум конструкций и элементов. Конструкций, которые включены в язык **CycloGraf**, всего три. Они представляют собой таблицы различной формы, позволяющие задать: 1) комбинационный автомат; 2) автомат с памятью; 3) цикл. Формы указанных таблиц будут рассмотрены далее при описании методов проектирования различных цифровых устройств.

Так же, как в универсальных высокоуровневых языках программирования и в языках описания аппаратуры, в набор основных элементов языка **CycloGraf** входят переменные (дискретные входные, выходные переменные, дискретные и целочисленные внутренние переменные и т.д.), таймеры, логические и арифметические операции и т.д. Разработанный табличный язык поддерживает все три базовые конструкции, характерные для языков высокого уровня: последовательное выполнение операций, ветвление и организация программного цикла.

Метод цикломатного проектирования ПО

Метод цикломатного проектирования ПО реализуется посредством комбинирования двух стилей программирования: сентенциального и «от состояний». В соответствии с [2] стилю программирования «от состояний» характерны глобальность действий и локальность условий. При этом в качестве математической модели используются конечные автоматы. Отсюда вытекает и используемая конструкция – таблица для задания конечных автоматов (таблица переходов). Сентенциональный стиль характеризуется глобальностью действий и условий. Так обстоит дело при описании циклов, поэтому вторая используемая конструкция – таблица микроциклов. Математическая модель, положенная в основу метода – канонический цикломат [3].

Рассмотрим типичный пример задания алгоритма управления технологическим объектом в виде циклограммы, при котором используются две взаимосвязанных таблицы: переходов и микроциклов. Каждая строка таблицы микроциклов (см. рис. 1) содержит:

- информацию о номере текущего микроцикла;
- номере шага (или выполняемой операции);
- командах (например, управления исполнительными механизмами, включения признака окончания микроцикла и т.д.);
- условиях переходов (или ожидаемых состояниях датчиков);
- адрес перехода, который указывает на № шага, к которому следует осуществить переход в

случае выполнения заданных условий, по умолчанию переход осуществляется к следующему (i+1) шагу микроцикла.

№ микроцикла	№ шага	Команды				Условия переходов			Адрес перехода
		c1	c2	c3	ENDmc	u1	u2	u3	
1	0	0	0	0	0	0	1	1	
	1	1	0	0	0	0	0	1	
	2	0	1	0	1				
2	0	0	1	1	0			1	
	1	0	0	1	0		1	0	
	2	1	0	0	1				

Рис. 1. Таблица микроциклов

Каждая строка таблицы переходов содержит (рис. 2): текущее состояние (№ микроцикла); условия переходов (состояние входных или внутренних переменных, при которых осуществляется переход к новому микроциклу, в том числе сигнала окончания микроцикла ENDmc); следующее состояние (микроцикл, к которому выполняется переход при выполнении заданных условий).

текущее состояние	**условия переходов**				**следующее состояние**
mc[1..0]	z1	z2	z3	ENDmc	mc[1..0]
0	1	0	0		1
1	0		0	1	2
2	1	0		1	0

Рис. 2. Таблица переходов между состояниями (микроциклами)

Таким образом, цикломатное программирование сводится к формированию пространства состояний объекта управления в процессе анализа его работы путем заполнения двух взаимосвязанных таблиц, одна из которых описывает последовательность выполняемых операций для каждого микроцикла, а вторая – условия переходов к этим микроциклам. Такое описание алгоритмов управления технологическими объектами является наглядным, первичным по своей природе и не требует навыков программирования, что соответствует решению задачи «максимум», поставленной в данном исследовании. Однако область эффективного использования цикломатного программирования ограничивается относительно несложными задачами управления объектами явно выраженного циклического действия.

Метод структурного проектирования ПО

Метод структурного проектирования ПО на языке **CycloGraf** предполагает программирование в структурном стиле, которому в соответствии с [2] присуща локальность условий и действий. Программа представляет собой набор таблиц, каждая из которых описывает отдельный функциональный блок (ФБ), реализующий ту или иную функцию. Совокупность ФБ образует иерархическую структуру проекта. Для обозначения входных, внутренних и выходных переменных используются уникальные имена, что дает возможность многократно использовать эти переменные при описании различных ФБ – так обеспечивается формирование линий связи между ними. Следует различать ФБ, настроенные на реализацию комбинационных схем, автоматов с памятью и циклов, для чего применяются соответствующие конструкции языка **CycloGraf**. Способ реализации комбинационных схем очевиден, поэтому ограничимся примером описания трехразрядного шифратора:

Значения аргументов	Значения функций
in[8..1]	code[3..1]
b”00000001”	0
b”00000010”	1
b”00000100”	2
b”00001000”	3
b”00010000”	4
b”00100000”	5
b”01000000”	6
b”10000000”	7

Рис. 3. Описание трехразрядного шифратора

Для описания автоматов с памятью используется таблица переходов. Пусть, например, автомат Мура задан графом (рис. 4).

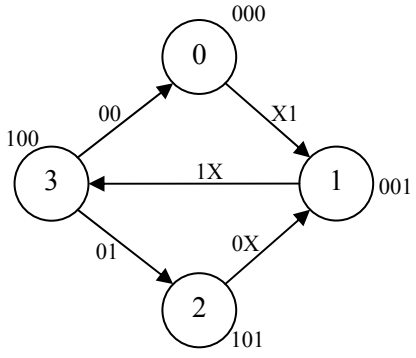

Рис. 4. Граф переходов, описывающий автомат Мура

Программа, описывающая данный граф, будет содержать две таблицы, причем первая из них описывает логические функции, которые реализуются во входном преобразователе, а вторая – в выходном преобразователе автомата Мура.

Текущее состояние	**Условия переходов**		**Следующее состояние**
State[1..0]	*p1*	*p2*	state[1..0]
0	x	1	1
1	1	x	3
2	0	x	1
3	0	1	2
3	0	0	0

Значения аргументов	**Значения функций**
state[1..0]	out[3..1]
0	b”000”
1	b”001”
2	b”101”
3	b”100”

Рис. 5. Описание автомата Мура на языке **CycloGraf**

В программе состояние автомата кодируется переменными state[1..0], входные переменные – *p1, p2*, выходные – out[3..1]. Отличие программы для задания автомата Мили состоит в том, что вторая таблица, описывающая выходной преобразователь автомата, дополняется значениями аргументов *p1*, *p2*, которые являются входными переменными и влияют на значения выходных сигналов out[3..1]. Следует отметить, что в таблице, описывающей условия переходов автомата, не требуется специально указывать условия, при которых автомат остается в предыдущем состоянии, поскольку эта функция реализуется по умолчанию.

Практический интерес представляет собой способ задания счетчиков. Рассмотрим счетчик, описанный при помощи таблицы переходов:

Текущее состояние	**Условия переходов**				**Следующее состояние**
state[3..0]	ena	reset	load	dir	state[3..0]
	1	0	0	0	state[3..0] + 1
	1	0	0	1	state[3..0] – 1
		0	1		0
		1			in[3..0]

Рис. 6. Описание счетчика на языке **CycloGraf**

В примере реализованы наиболее типичные функциональные возможности счетчика: сброса, загрузки, разрешения работы и реверсивного счета. Счетчик работает как суммирующий, если сигнал dir=0; как вычитающий, если dir=1; обнуляется при reset=1; устанавливается в состояние, соответствующее сигналам in[3..0] при load=1. Наивысшим приоритетом обладает сигнал reset, далее следует сигнал load, потом – ena, который разрешает инкрементирование или декрементирование содержания счетчика.

На всех незаданных значениях сигналов, описывающих условия переходов, счетчик не изменяет своего состояния. Данный пример интересен тем, что столбец «Текущее состояние» не заполнен, а зависимость следующего состояния счетчика от текущего отражается прямо в столбце «Следующее состояние» в виде арифметических операций «+1» или «-1».

Для описания цикла используется таблица микроциклов (рис. 7):

Неявное задание адреса перехода в цикле предполагает переход к следующей строке цикла. На 5-й строке реализовано ветвление в цикле, причем одно

из условий перехода (к 7-й строке) описано в виде логического уравнения. Следует отметить, что конструкции языка **CycloGraf** позволяют также наглядно описывать встроенные циклы, задавать непосредственно в теле цикла счетчики, запускать и опрашивать таймеры и т.д.

№ шага	Команды							Условия переходов										Адрес перехода
	t1e	t2e	t3e	t4e	t5e	t6e	razr	_1u	_2u	_3u	_4u	t1q	t2q	t3q	t4q	t5q	t6q	
0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	
1	1	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	
2	0	1	0	0	0	0	0	x	0	0	1	0	1	0	0	0	0	
3	0	0	1	0	0	0	0	x	0	0	0	0	0	0	0	0	0	
4	0	0	1	1	0	0	0	1	1	0	0	0	0	1	0	0	0	
5	0	1	0	1	1	0	1	1	1	1	0	0	1	0	0	0	0	
								_3u & ! t2q # ! _3u & t4q # ! _3u & t5q # ! _1u # ! _2u										7
6	0	0	0	0	0	0	1	! _4u & t6q # ! _1u # ! _2u # ! _3u										
7	0	0	0	0	0	0	0											7

Рис. 7. Описание цикла на языке **CycloGra**f

Метод структурного программирования на языке **CycloGraf,** обладая простой формой представления алгоритмов работы функциональных блоков, позволяет реализовать цифровые устройства практически любой сложности, но, одновременно, проигрывает в наглядности описания цикломатному проектированию ПО. В связи с этим, структурное программирование на языке **CycloGraf** эффективно при решении сложных и нестандартных задач и требует от пользователя определенных навыков построения цифровых устройств. При этом следует отметить, что описанные конструкции языка **CycloGraf** обладают явными преимуществами по сравнению с известными средствами проектирования ПО для ПЛИС, которые проявляются как при подготовке ПО, так и при его сопровождении, анализе и коррекции, а табличная форма представления алгоритмов делает их доступными для понимания технологами, не имеющими навыков программирования.

Выводы

1. Табличный язык CycloGraf позволяет на высоком уровне абстракции описывать технологи ческими терминами модель логического управления, на основе которой специализированный транслятор автоматически генерирует программный код на языке описания аппаратуры.

2. Разработанный метод структурного проектирования ПО за счет значительного сокращения количества и упрощения конструкций языка программирования обеспечивает уровень наглядности программ, доступный технологу, не имеющему навыков программирования; таким образом достигается существенное повышение эффективности взаимодействия программиста и технолога.

3. Разработанный метод цикломатного проектирования ПО обладает всеми свойствами технологического программирования, предполагающего возможность составления программ непосредственно технологом без участия программиста.

Литература

1. Фурман И.А., Малиновский М.Л., Бовчалюк С.Я., Аллашев А.Ю. Перспективная технология программирования промышленных ПЛИС-контроллеров // Мир техники и технологий. − 2007. − № 67. − С.60-62.

2. Непейвода Н.Н. Логики и стили программирования. – [Электронный ресурс]. – Режим доступа: http://logic.ru/ru/node/243.

3. Малиновський М.Л., Фурман І.О., Аллашев О.Ю., Тихонравов С.М. Синтез керуючих автоматів циклічної дії (цикломатів) // Проблеми енергозабезпечення та енергозбереження в АПК України: Вісник ХНТУСГ. − 2007. – Вип. 57, Т. 2. – С. 92-99.

Поступила в редакцию 12.02.2008

Рецензент: д-р техн. наук, проф. В.С. Харченко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.