МЕТОД ЛОГИКО-ВРЕМЕННОГО АНАЛИЗА ЦИФРОВЫХ СИСТЕМ С ОГРАНИЧЕНИЕМ НА ВРЕМЯ ОТКЛИКА

Черемисинов Д. И
Объединённый институт проблем информатики Национальной академии наук Беларуси
Минск, Республика Беларусь
E-mail: cher@newman.bas-net.by

Предлагается метод анализа цифровых систем по алгоритмическому описанию на языке параллельных алгоритмов логического управления. Метод основан на построении множества структурных состояний переходов параллельного алгоритма и расширенного графа достижимости состояний на множестве меток и логических переменных алгоритма, и является модификацией метода символической проверки на модели.

Введение

Интегральные схемы в настоящее время проектируется в результате выполнения ряда шагов, каждый из которых переводит абстрактную спецификацию, уточняя ее, во все более конкретное воплощение. Процесс проектирования начинается с «поведенческой модели», например, с программы, которая описывает архитектуру процессора на уровне набора команд. Результатом проектирования является описание фактической топологии транзисторов и проводов на чипе. Каждый шаг проектирования должен давать описание, релевантное исходному поведению абстрактной модели. К сожалению, методы выполнения проверки соответствия спецификации и реализации становятся все более дорогими и трудными по мере роста сложности проекта. Для многих проектов размер группы проектировщиков, выполняющих проверку (верификацию) проекта, теперь превышает число участников группы проектирования.

Формальная проверка обеспечивает новый подход к подтверждению правильного поведения описания СБИС. В моделировании, традиционном способе отладки описания СБИС, подтверждение правильности текущего описания СБИС – это результат прогонов моделирования с большим количеством испытательных ситуаций. Формальная проверка (верификация), напротив, использует математические методы, чтобы исследовать полное пространство состояний модели для подтверждения соответствия наперед заданному поведению. последние годы инструменты для символической верификации находят существенное и растущее применение, они рутинно используются в индустриальных САПР [1].

Ключевой вопрос при использовании инструмента для символической верификации – это формулировка свойств - assertion на языке спецификации временных условий, который является одним из определяющих интерфейсов программы. (Другой важный интерфейс – это язык моделирования, в этом качестве обычно используется язык описания аппаратных средств ЭВМ, применяемый проектировщиками СБИС). Языки спецификации временных условий имеют в основе формализм временных логик [2, 3].

Алгоритмы верификации на символической модели могут проверять большой и важный класс свойств цифровых систем. Тем не менее, существует важный класс свойств, который нельзя адекватно обрабатывать с помощью этого метода. Этот класс состоит из свойств, которые ограничивают время отклика. Во временных логике CTL [3] можно указать, что какое-то событие произойдет в будущем, но свойство, что какое-то событие произойдет не более чем через x единиц временных не может быть выражено напрямую.

В настоящей работе метод символической верификации, модель проверяемого устройства задается на языке ПРАЛУ [4], и метод дает минимальный и максимальный временной интервал для перехода из известного состояния запуска ПРАЛУ алгоритма в заданное.

I. Предлагаемый метод

Проверяемая система задается на языке ПРАЛУ и затем компилируется в граф переходов состояний. алгоритм на ПРАЛУ служит точным описанием системы, которое может выявить тонкие двусмысленности и может использоваться для целей документирования. Формализация понятия процесса в языке ПРАЛУ использует в качестве базы представление об операции. Временное свойство всех операции одно: протяженности во времени – операция это интервальное событие, т.е. временной интервал конечной, но не нулевой длины Предполагается, что все элементарные операции действия имеют одинаковую длительность, т.е. параллельные процессы выполняются в режиме блокировки (lock-step – шагать в ногу тесным строем).

Метод позволяет получить количественную информации о времени отклика, т.е. минимальные и максимальные задержки между запросом и соответствующим ответом. Например, метод позволяет получить границы интервала времени между запросом на доступ к шине и соответству-

ющим предоставлением доступа к шине. Кроме того, можно вычислить количество случаев, когда третье событие происходит в течение такого интервала, например, количество других регистровых пересылок, происходящих между запросом на шину и соответствующим разрешением на доступ.

Основным ограничением нашего подхода является неотъемлемая сложность проблемы проверки на модели. Граф переходов имеет экспоненциальную асимптотическую сложность в количестве вершин, и нет никаких гарантий того, что алгоритм анализа сможет выполнить проверку в любом практическом примере. В большинстве случаев проверка выполняется в считанные минуты, даже для сложных систем реального мира. Следует, однако, сказать, что эти проблемы присущи любому формальному методу логико-временного анализа.

II. Скелет α-сети

Далее для задания поведения систем управления используется параллельные алгоритмы логического управления на языке ПРАЛУ в стандартном виде модели параллельного автомата [4]. Алгоритмы в таком виде представляют подкласс раскрашенных сетей Петри – расширенные сети свободного выбора. Алгоритм представляет собой совокупность стандартных цепочек вида: $\tau_i = (\mu_i \rightarrow \nu_i)/(k_i^1 \rightarrow k_i^2)$, где метки τ_i и ν_i трактуются как подмножества частичных состояний, а элементарные конъюнкции k_i^1 и k_i^2 – как условие перехода и выходные сигналы, сопровождающие переход. Переход срабатывает, когда текущая маркировка N_t на множестве состояний M = 1,2,. . . ,m включает все состояния из μ_i и переменные принимают значения, обращающие k_i^1 в 1. После срабатывания перехода переменным из k_i^2 присваиваются значения, обращающие k_i^2 в 1, а маркировка N_t заменяется на $(N_t \backslash \mu_i) \cup \nu_i$. Формальная верификация описания алгоритма на языке ПРАЛУ основана на анализе его скелета, α-сети, задаваемого множеством переходов $\tau_i = (\mu_i \rightarrow \nu_i)$ и соответствующего ординарной сети Петри.

Основной структурой, лежащей в основе методов анализа поведенческих свойств алгоритма на языке ПРАЛУ, является его граф достижимости, который строится на основе α–сети алгоритма управления. Вершинам графа достижимости соответствуют все возможные разметки N_t, а дуга, помеченная символами одного или нескольких переходов, соединяет разметки, такие, что сеть переходит от первой разметки ко второй при срабатывании этих переходов. Анализируя α–сеть и построенный на ее основе граф достижимости, можно исследовать корректность алгоритма на языке ПРАЛУ [5], но это исследование будет неполным. Для полноты необходимо учесть в нем логические характеристики: достижимые структурные состояния на множестве его переменных, которые обуславливают и временные характеристики выполнения алгоритма управления. Соответственно дуги графа достижимости необходимо помечать не только переходами, но и парой: входное условие срабатывания перехода и значения изменяемых переменных. Метод построения такого расширенного графа достижимых частичных состояний и соответствующих достижимых состояний логических переменных основан на получении множества структурных состояний переходов параллельного алгоритма – состояний его логических переменных на момент переходов.

III. Количественный анализ, время отклика

Проверка на модели (Model checking) – это метод автоматического анализа, который исследует все возможные состояния анализируемой системы, чтобы проверить, удовлетворяет ли система формально заданному свойству. Свойства записываются в виде формул в пропозициональной временной логике. В предлагаемом методе свойства задаются неявно указанием пары разметок графа достижимости для вычисления количественной временной информации, такой как точные минимальные и максимальные задержки на время между запросом и соответствующим ответом.

Свойства, необходимые для количественного временного анализа, не могут быть выражены в традиционной пропозициональной временной логике. Можно обеспечить задание и проверку ограниченных по времени свойств введением границ в временные операторы временной логики. Расширенная логика называется RTCTL [6]. Предлагаемый метод задания информации для количественного анализа поведения не так универсален как RTCTL, но позволяет упростить алгоритм проверки на модели для типично проверяемых свойств.

1. J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and J. Hwang. Symbolic model checking: 1020 states and beyond. Information and Computation ,Volume 98, Issue 2, June 1992, Pages 142-170.
2. L. Lamport. Sometimes is sometimes “not never” – on the temporal logic of programs. In Proc. 7th ACM Symp. on Principles of Programming Languages, pages 174–185, January 1980.
3. F. Wang. Timing behavior analysis for real-time systems. Proceedings of Tenth Annual IEEE Symposium on Logic in Computer Science, San Deigo, CA, USA, 1995, pp. 112-122.
4. Черемисинова, Л.Д. Реализация параллельных алгоритмов логического управления – Минск: Ин-т техн.кибернетики НАН Беларуси, 2002. – 246 с.
5. Закревский, А.Д. Параллельные алгоритмы логического управления – Минск: Ин-т техн.кибернетики НАН Беларуси, 1999. – 202 с.
6. E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. In Lecture Notes in Computer Science, Computer-Aided Verification. Springer- Verlag, 1990.