УДК 004.94 DOI 10.18522/2311-3103-2024-4-101-110

А.В. Хлуденев, С.А. Сильвашко

СОПРЯЖЕННОЕ МОДЕЛИРОВАНИЕ БИС В САПР ПЕЧАТНЫХ УЗЛОВ

Виртуальное прототипирование выполняют в процессе разработки новых изделий с целью проверки проекта перед созданием физического прототипа, используя компьютерные модели. В САПР печатных узлов с этой целью используют SPICE симуляторы схем. Печатные узлы современных электронных устройств построены на одной или нескольких интегральных схем (ИС) высокой степени интеграции. Функционал больших интегральных схем (БИС) дополняют вспомогательные ИС и дискретные компоненты. В большинстве случаев требуемая эффективность обеспечивается при использовании БИС с процессорными ядрами. Поэтому симуляторы схем должны обеспечивать сопряженное моделирование аппаратных и программных средств. Приемлемыми по затратам вычислительных ресурсов являются модели БИС системного уровня. Основные достижения в области моделирования на системном уровне, включая сопряженное моделирование, связаны с разработкой самих БИС. В схемах печатных узлов они являются готовыми комплектующими изделиями. Эту специфику необходимо учитывать при реализации инструментов верификации схем печатных узлов. Модели БИС системного уровня должны встраиваться в модель всей схемы, быть экономичными и обеспечивать требуемую точность на границе внешних выводов. Модели цифровых БИС должны достоверно отображать задержки между изменениями уровней на выводах и диагностировать нарушения синхронизации. Модели БИС должны разрабатывать пользователи САПР печатных узлов, учитывая специфику проекта. Целью исследования является поиск решений для построения моделей БИС, включающих процессорные ядра, для прототипирования схем, используя OrCAD PCB Designer with PSpice. В статье рассматривается задача построения C/C++ модели для микроконтроллера семейства dsPIC, выполняющего обработку сигнала в реальном времени. Приведены пример построения модели, используя инструменты PSpice Model Editor, и результаты моделирования.

Печатный узел; виртуальное прототипирование; схемный симулятор; БИС; процессорное ядро; периферийный модуль; прямой доступ к памяти; модель системного уровня; сопряженное моделирование; симулятор системы команд; симуляция компилированного хостом программного обеспечения.

A.V. Khludenev, S.A. Silvashko

LSI COSIMULATION IN EDA FOR PCB DESIGN

Virtual prototyping is performed during product development to validate a design using a computer model before making a physical prototype. For this purpose, EDA for printed circuit board (PCB) design contain SPICE circuit simulator. Typically, modern PCB assemblies include one or more large-scale integrated circuits (LSI). The LSI functionality is complemented by auxiliary integrated circuits (IC) and discrete components. In most cases, the required efficiency is achieved by using LSIs that include processor cores. Therefore circuit simulators must provide a hardware/software co-simulation. System-level LSI models are acceptable in terms of computational resource costs. Major advances in system-level simulation, including co-simulation, come from the development of LSIs themselves. In PCB design, LSIs are fully fabricated components. This specificity must be taken into account when implementing tools for PCB design verifying. System-level LSI models must be integrated into the overall assembly circuit model. LSI models must provide the required accuracy only at the external pins. Models of digital LSIs must accurately reproduce delays between level changes at the pins and diagnose timing violations. EDA for PCB design users must develop LSI models tailored to the project specifics. The purpose of the research is to find solutions for building models of LSIs, containing processor cores, for prototyping circuits using OrCAD PCB Designer with PSpice. The article discusses the task of building a C/C++ model for the dsPIC33 microcontroller that performs signal processing in real time. An example of building a C/C++ model using the PSpice Model Editor tools and modeling results are given.

PCB assembly; virtual prototyping; circuit simulator; large-scale integrated circuit; processor core; peripheral module; direct memory access; system level model; co-simulation; instruction set simulator; host-compiled software simulation.

Введение. Стремление обеспечить расширенную функциональность и высокую надежность электронных устройств при снижении стоимости, потребляемой мощности, массы и габаритных размеров являются стимулом широкого применения в разработках БИС. В большинстве задач, некритичных по времени выполнения заданных функций, наилучшее соотношение цена – качество обеспечивается при реализации устройств на основе БИС микропроцессоров (МП) и микроконтроллеров (МК), далее МП БИС. Необходимая функциональность изделий обеспечивается выбором МП БИС с требуемыми характеристиками, а также в результате разработки и отладки встроенного программного кода. В конечных изделиях МП БИС вместе с другими компонентами устанавливаются в конструктивы на основе печатных плат. Разработку схем и конструкций таких изделий выполняют в среде САПР печатных узлов [1]. На этапе разработки электрической схемы печатного узла для анализа и верификации традиционно используют SPICE симуляторы и модели [2].

Симулятор PSpice A/D, входящий в состав САПР OrCAD PCB Designer, позволяет выполнять многоуровневый анализ схем, построенных на дискретных элементах, аналоговых и цифровых ИС. Макромодели цифровых ИС реализованы в виде подсхем, состоящих из базовых примитивов уровня логических вентилей [3]. В современных версиях PSpice A/D появилась возможность использовать алгоритмические C/C++ модели цифровых БИС системного уровня. В отличие от специализированных средств, например [4], в библиотеках OrCAD PCB Designer отсутствуют готовые модели БИС. Инструменты PSpice A/D ориентированы на ускоренный анализ схем печатных узлов, включающих от одной до нескольких БИС. Для этого следует использовать упрощенные проектно-ориентированные модели БИС, обеспечивающие требуемую точность на границе их внешнего интерфейса. Разработку таких моделей для своих проектов с учетом требований [5] должны выполнять пользователи. Исследование путей реализации PSpice C/C++ моделей цифровых БИС является актуальной задачей. В работе предлагается методика построения PSpice C/C++ моделей МП БИС, выполняющих целевую программу в режиме Bare Metal.

Постановка задачи. Для заданной МП БИС в составе схемы печатного узла построить математическую модель, отвечающую требованиям:

- должна встраиваться в PSpice модель схемы печатного узла;
- должна обеспечивать сопряженное моделирование аппаратных (HW) и программных (SW) средств;
- должна обеспечивать заданную точность моделирования сигналов на внешних выводах при минимальной сложности реализации и минимальных затратах вычислительных ресурсов.

Предлагаемая методика использует в качестве ресурса для выполнения последнего требования учет специфики выполняемых МП БИС задач и особенностей их технической реализации.

PSpice макромодели цифровых ИС. Основной задачей анализа схем цифровых устройств в PSpice A/D является обнаружение рассогласования сигналов во времени, рисков сбоя, нарушений условий синхронизации. Функционирование цифровых ИС в PSpice A/D описывается пятизначными моделями логических вентилей, триггеров и функциональных примитивов LOGICEXP, входящих в подсхему макромодели [3]. Свойства выводов описываются моделью входа - выхода, подсхемами интерфейса с аналоговой частью устройства и источниками питания. Динамические свойства описываются временной моделью, задающей задержки распространения сигналов с учетом разброса значений, и примитивом проверки временных ограничений CONSTRAINT.

Макромодель системного уровня в PSpice A/D также представляется подсхемой, в которую включен функциональный примитив LOGICEXP с квалификатором C_MODEL и ссылкой на dll файл исполняемого модуля модели, временной моделью типа ugate и моделью входов – выходов. Взаимодействие вычислительного ядра PSpice A/D с моделями устройств системного уровня обеспечивает Device Modeling Interface (DMI) [6]. Используя инструмент DMI Template code generator утилиты PSpice Model Editor, можно

автоматически сформировать файл PSpice макромодели и набор файлов проекта MS Visual Studio, включающий стандартные файлы DMI и файлы шаблона C/C++ модели БИС [7]. Исполняемый код модели БИС и DMI в виде dll файла создается в результате компиляции проекта в MS Visual Studio. Симулятор PSpice A/D загружает dll файл при выполнении Transient анализа.

Модели МП БИС системного уровня. Инструменты сопряженного HW/ SW моделирования появились и развивались для верификации проектов систем на кристалле, содержащих процессорные ядра [8–9]. Основные решения по разработке HW/SW симулятора связаны с реализацией модели процессорных ядер, выполняющих целевой программный код, симулятора аппаратных модулей МП БИС и синхронизации процессов в этих моделях. Модели более высокого уровня абстракции экономичнее по затратам вычислительных ресурсов на выполнение анализа, но менее точно учитывают время событий.

Модели процессорного ядра, способные имитировать его работу с точностью до машинного цикла, реализуют в виде симулятора набора команд (англ. instruction set simulator (ISS)) [10–11]. Код целевой программы формируют кросс компилятором. ISS интерпретирующего типа выполняет цикл, в котором симулируется выборка, декодирование и исполнение машинных команд целевым процессором. В ISS компилирующего типа инструкции целевого процессора преобразуют в эквивалентные макросы и затем компилируют в код хоста. В компилированной модели отсутствуют этапы выборки и декодирования инструкций, что приводит к ускорению моделирования [12]. Для сохранения точности учитывают затраченное время, используя механизм аннотации [13]. Скорость моделирования повышают, используя эффективные модели HW и механизмы синхронизации процессов [14]. Обзор коммерческих и академических инструментов сопряженного HW/SW моделирования приведен в [15].

Разработка моделей МП БИС на основе ISS оправдана при многократном применении в различных проектах. Для проектно ориентированных моделей эффективнее более простые решения. Точность модели на внешних выводах можно обеспечить при ее адекватности по времени при чтении и записи регистров портов и периферийных модулей. Время между другими событиями, протекающими в МП БИС, можно учитывать грубо. Часто в задачах реального времени процессорные ядра непосредственно не выполняют операции обмена с портами и регистрами данных периферийных модулей. В МП БИС с этой целью используют каналы прямого доступа к памяти (англ. Direct Memory Access (DMA)). В этом случае SW модель может быть "свободной по времени" и реализована путем компиляции адаптированного целевого С кода в исполняемый код хоста. Полагая, что целевая программа имеет модульную структуру и все изменения в регистрах и памяти происходят по фронту импульсов машинного цикла CLK, модель МП БИС можно построить, следуя правилам методики:

1) процессы в периферийных модулях протекают параллельно, поэтому после детектирования фронта CLK должна выполняться обработка моделей всех модулей (очередность обработки должна быть противоположной последовательности передачи сигналов между модулями, чтобы новые данные в регистрах действовали на входах приемников в следующем машинном цикле);

2) процессорные ядра выполняют операции обработки данных последовательно, поэтому после детектирования фронта CLK может выполняться моделирование только одного программного модуля (очередность обработки моделей программных модулей должна определяться алгоритмом программы);

3) вне очереди должно выполняться моделирование обработчиков прерываний при поступлении запросов на прерывания (при появлении нескольких запросов на прерывания должно выполняться моделирование обработчика для запроса с более высоким приоритетом).

Пример для построения модели. Экономичность реализации модели МП БИС зависит от учета особенностей реализации объекта моделирования. Рассмотрим в качестве примера фильтр с конечной импульсной характеристикой (КИХ) (рис. 1), реализованный

на МК dsPIC33FJ256GP506. МК семейства dsPIC33 содержат универсальное процессорное ядро MCU, ядро обработки сигналов DSP, контроллер DMA и набор периферийных модулей [16]. Процессорные ядра выполняют все инструкции (кроме команд передачи управления) за один машинный цикл, состоящий из двух машинных тактов. Процессорные ядра и периферийные модули тактируются от одного встроенного генератора с частотой импульсов F_{osc} = 80 МГц, частота машинного цикла F_{cy} = 40 МГц.

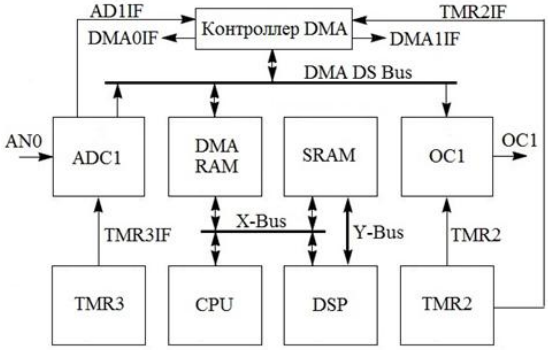

Рис. 1. Функциональная схема КИХ-фильтра

Для оцифровки входного сигнала с канала AN0 используется встроенный модуль аналого-цифрового преобразователя (АЦП) ADC1. АЦП последовательных приближений тактируется импульсами от делителя частоты с периодом $T_{AD} = 32/F_{cy}$. Запуск АЦП происходит при переполнении таймера TMR3 и установке флага TMR3IF, частота дискретизации F_s = 16 кГц. В течение интервала $3 \cdot T_{AD}$ происходит выборка входного напряжения, а затем в течение интервала $14 \cdot T_{AD}$ формируются 12 разрядов выходного кода [17]. При завершении преобразования устанавливается флаг AD1IF и инициирует передачу кода выборки по каналу DMA0, работающему в режиме ping-pong, в один их буферов обмена InBufferA или InBufferB, расположенных в области памяти данных DMA RAM [18]. При заполнении текущего буфера контроллер DMA устанавливает флаг прерывания DMA0IF и переходит к заполнению второго буфера.

Обработчик прерывания DMA0_IRQ инвертирует индикатор заполненного буфера InBufferIndicator и сбрасывает признак isReadBusy. Ядро MCU ожидает это событие и переходит к выполнению функции чтения заполненного буфера ADCChannelRead и копированию выборок в массив Samples в SRAM. При завершении копирования ADCChannelRead устанавливает признак isReadBusy. Схема алгоритма основного модуля целевой программы приведена на рис. 2.

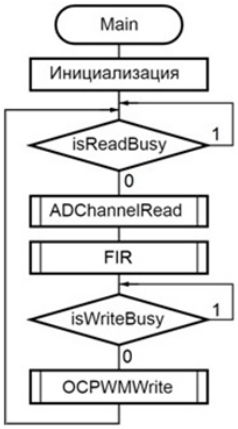

Рис. 2. Схема алгоритма целевой программы МК

Алгоритм КИХ-фильтрации выборок из массива Samples реализован в функции FIR. Основные операции выполняет ядро DSP при выборке данных из массивов линии задержки и коэффициентов. Коэффициенты были рассчитаны для формирования частотной характеристики полосового фильтра с полосой пропускания от 2 до 4 кГц. Подавление в диапазонах частот ниже 1 кГц и выше 5 кГц не менее 52 дБ. Обработанные выборки сохраняются в массиве Filterout в SRAM. Ядро MCU переходит к ожиданию сброса признака isWriteBusy.

Для восстановления аналогового сигнала используются широтно-импульсный модулятор (ШИМ), реализованный на встроенном модуле выходного компаратора OC1 [19], и внешний фильтр нижних частот (ФНЧ) [20]. Выходные выборки передаются из буфера OutBufferA или OutBufferB в памяти DMA RAM по каналу DMA1 в регистр выходного компаратора OS1RS по флагу TMR2IF при переполнении таймера TMR2. Модуль счета TMR2 определяет значение несущей частоты ШИМ сигнала F_{PWM} = 64 кГц. Канал DMA1 работает в одиночном блочном режиме без постинкремента адреса источника. Флаг прерывания DMA1IF устанавливается после четырехкратной передачи каждой выборки. Инкремент адреса источника и режим ping-pong реализованы программно в обработчике прерывания DMA1_IRQ. Когда все ячейки текущего буфера прочитаны, DMA1_IRQ инвертирует OutBufferIndicator, изменяет базовый адрес буфера и сбрасывает признак isWriteBusy. Ядро MCU переходит к выполнению функции OCPWMWrite для нормирования и записи элементов массива Filterout в свободный от чтения буфер обмена OutBufferA или OutBufferB. При завершении копирования OCPWMWrite устанавливает признак isWriteBusy. После выполнения функции OCPWMWrite ядро MCU переходит к ожиданию сброса признака isReadBusy.

Построение модели. Средства PSpice Model Editor формируют шаблон C/C++ модели только для БИС с полностью цифровым внешним интерфейсом. Поэтому для модуля АЦП была использована внешняя модель элемента ADC12break из библиотеки Breakout.OLB. Модель контроллера АЦП, формирующего сигнал ADC1IF, реализована в составе C/C++ модели. Для КИХ-фильтра были сформированы схемный символ для размещения на схеме (файл FIR_DMA.OLB), PSpice макромодель (файл FIR_DMA.LIB) и набор файлов проекта MS Visual Studio. В файле pspFIR_DMA.h проекта объявлены класс pspFIR_DMA и стандартные функции модели, идентификаторы переменных и констант (были дополнены функциями и идентификаторами пользователя). Файл проекта pspFIR_DMA.cpp содержит инсталлятор функций DMI installpspFIR_DMA. Шаблоны пользовательских функций модели приведены в файле FIR_DMA_user.cpp. Функция инициализации портов и переменных модели pspFIR_DMA::initialize вызывается PSpice A/D однократно при запуске Transient анализа. Функция pspFIR_DMA::evaluate вызывается PSpice A/D в каждом такте моделирования и выполняет операции:

- получает текущее время pTicks;
- получает PSpice уровни входных сигналов из pVectorStates;
- детектирует фронт синхросигнала CLK;
- преобразует PSpice уровни сигналов в C++ целочисленные значения;
- выполняет алгоритм C/C++ модели;
- преобразует C++ данные в PSpice уровни выходных сигналов;
- обновляет состояния выходов в pVectorStates.

Для данного примера наиболее простым решением будет построение модели, в которой в течение одного машинного цикла dsPIC33 хост полностью выполняет эквивалент одного модуля целевой программы. Схема алгоритма функции модели pspFIR_DMA::evaluate приведена на рис. 3. После обнаружения фронта CLK модель имитирует работу dsPIC33 в текущем машинном цикле. Получает уровни сигналов, проверяет флаги прерываний DMA0IF и DMA1IF. При поступлении запроса прерывания выполняет функцию обработчика DMA0_IRQ или DMA1_IRQ. При отсутствии запросов пре-

рывания анализирует значение переменной маркера (Marker). В каждом машинном цикле dsPIC33 хост выполняет только одну ветвь алгоритма модели с последующей передачей маркера. Операторы задержки при isReadBusy = 1 и isWriteBusy = 1 реализованы как пропуск передачи маркера.

После имитации работы процессорных ядер выполняется обработка моделей всех периферийных модулей в следующей последовательности:

1) канала DMA0 (обработка запроса AD1IF);
2) контроллера ADC1 (установка AD1IF через 17 периодов T_{AD} после запуска АЦП);
3) таймера TMR3 (установка TMR3IF при переполнении, запуск АЦП);
4) делителя частоты Clock Divider для тактирования ADC1;
5) канала DMA1 (обработка запроса TMR2IF);
6) выходного компаратора OC1 (сброс выхода OC1 при TMR2 = OC1RS);
7) таймера TMR2 (установка TMR2IF и выхода OC1 при переполнении).

В данном примере при использовании буферов DMA на 32 выборки один основной цикл целевой программы выполняется за 80000 машинных циклов. Из них при моделировании хост выполняет ветвь обработчика DMA1_IRQ восемь раз, остальные ветви (кроме ветвей пропуска) по одному разу.

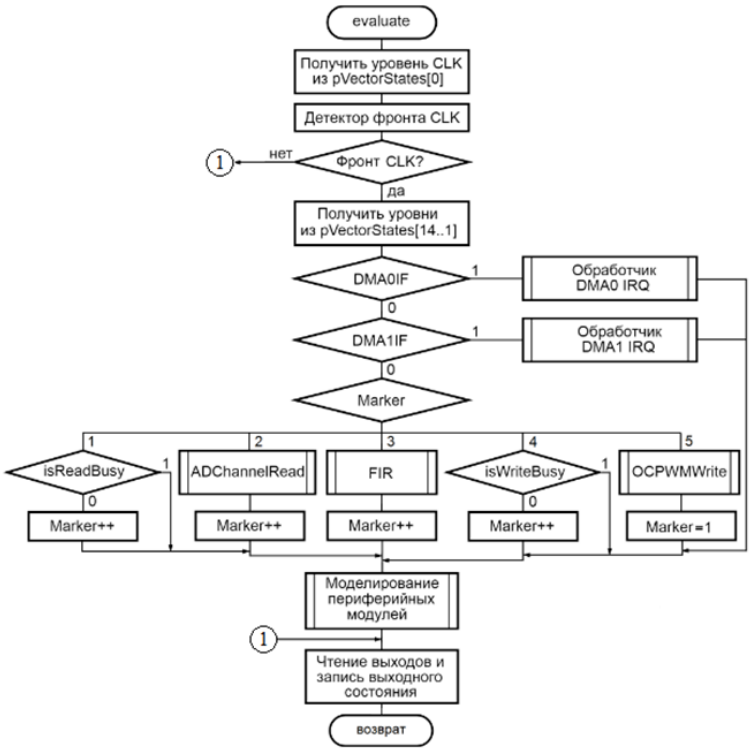

Рис. 3. Схема алгоритма функции модели pspFIR_DMA::evaluate

Результаты моделирования. На рис. 4 приведена схема для тестирования модели КИХ-фильтра в PSpice A/D. Параметры источников сигналов и аналоговых фильтров приведены на схеме. На рис. 5 приведен фрагмент результатов анализа реакции схемы на подачу двухтонального сигнала в течение 24 мс. Форма и параметры сигналов на выводах МП БИС соответствуют ожидаемым, подавляется гармоника с частотой 1 кГц. Грубые упрощения, допущенные при формировании модели SW компоненты, не оказывают влияние на полученные результаты. Параметры ШИМ сигнала на входе восстанавливающего фильтра (уровни высокого и низкого потенциалов и длительности фронтов) были учтены в PSpice модели входов - выходов КИХ-фильтра.

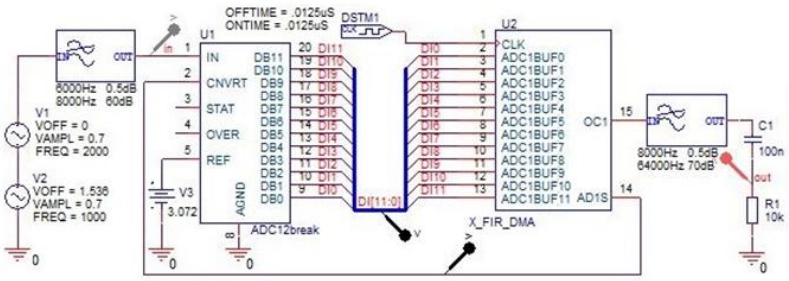

Рис. 4. Схема для тестирования модели КИХ-фильтра

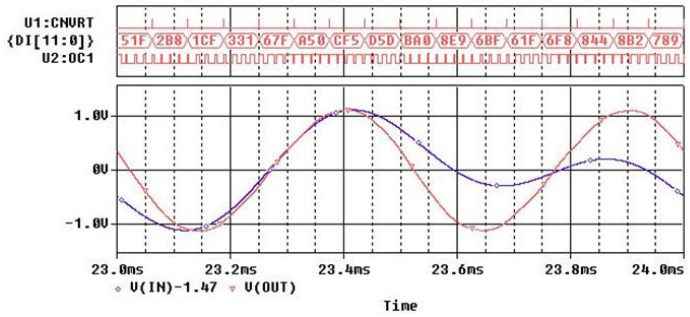

Рис. 5. Временные диаграммы

Модификации модели. Рассмотренный вариант модели будет давать идеализированные результаты работы МП БИС в случае, если процессорные ядра не будут успевать завершать обработку фреймов выборок до переключения буферов обмена контроллером DMA. Эти риски можно снизить, если выполнить доработку алгоритма модели МП БИС в одном из вариантов:

1) в течение одного машинного цикла хост полностью выполняет один модуль программы, при этом следующий программный модуль может быть выполнен после пропуска заданного количества машинных циклов;

2) если в модуле программы реализован циклический алгоритм, то в течение одного машинного цикла хост полностью выполняет один проход цикла программного модуля, следующий проход цикла программного модуля может быть выполнен после пропуска заданного количества машинных циклов.

Количество пропусков должно определяться временем выполнения программного модуля или одного прохода его цикла целевым процессором. В табл. 1 приведены значения времени выполнения программных модулей, определенные при выполнении кода, полученного кросс компилятором, с помощью инструментов симулятора – отладчика MPLAB IDE.

Таблица 1

Время выполнения модулей программы МК

Функция программы	Время, машинных циклов
DMA0_IRQ	20
DMA1_IRQ	22
ADCChannelRead	619
FIR	1048
OCPWMWrite	16388
Проверка isReadBusy	16
Проверка isWriteBusy	15

Заключение. Рассмотрена задача построения PSpice C/C++ моделей МП БИС для виртуального прототипирования схем печатных узлов. Учет особенностей технической реализации объекта моделирования позволяет снизить сложность реализации модели при заданной точности расчета сигналов на выводах МП БИС. Предложена методика построения модели МП БИС в форме алгоритма, управляющего вызовом эквивалентов модулей целевой программы и моделей периферийных модулей. При формировании эквивалентов на основе адаптированного С кода целевой программы МП БИС сокращаются трудоемкость и сроки разработки модели.

Приведен пример построения модели для МК семейства dsPIC33, выполняющего в реальном времени алгоритм КИХ-фильтрации при обмене кодами выборок входного и выходного сигналов с периферийными модулями через буферы DMA. При ограничениях на выполнение операций обмена с регистрами портов и периферийных модулей, точность модели на внешних выводах зависит от реализации моделей периферийных модулей (от машинного цикла до такта Transient анализа PSpice A/D). Такие модели целесообразно использовать при параллельном командном проектировании печатных узлов для верификации схем до завершения тестирования и отладки целевого программного кода. Предложены варианты модели МП БИС с учетом циклического характера алгоритмов и времени выполнения модулей целевой программы. В этих моделях необходимо учитывать время выполнения отлаженных модулей программы или одного прохода цикла.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ГОСТ Р 70607–2022. Системы автоматизированного проектирования электроники. Состав и структура системы автоматизированного проектирования печатных узлов. – М.: Рос. ин-т стандартизации, 2023. – 8 с.
2. ГОСТ Р 70884–2023. Системы автоматизированного проектирования электроники. Информационное обеспечение. Модели SPICE. Общие положения. – М.: Рос. ин-т стандартизации, 2023. – 8 с.
3. PSpice user guide. Product version 17.2–2016. – Cadence, 2016. – 900 p. – URL: https:// resources.pcb.cadence.com/i/1180526-pspice-user-guide/25 (дата обращения: 20.05.2024).
4. *Васьков А.С., Смирнов Б.И.* Использование пакета Proteus для проектирования виртуальных моделей микропроцессорных устройств // Известия вузов России. Радиоэлектроника. – 2013. – Вып. 2. – С. 38-43.
5. ГОСТ Р 70806–2023. Системы автоматизированного проектирования электроники. Информационное обеспечение. Порядок разработки моделей SPICE. Постановка задачи. – М.: Рос. ин-т стандартизации, 2023. – 11 с.
6. PSpice device modeling interface API reference. Product version 17.2–2016. – Cadence, 2016. – 93 p.
7. Virtual prototyping in PSpice: Application note, V1. – FlowCAD, 2016. – 72 p. – URL: https://www.flowcad.de/AN/FlowCAD-AN_Device-Modeling-Interface.pdf (дата обращения: 20.05.2024).
8. *Teich J.* Hardware/software codesign: the past, the present, and predicting the future // Proc. IEEE. – 2012. – Vol. 100. – P. 1411-1430. – DOI: 10.1109/JPROC.2011.2182009.
9. *Rowson J.A.* Hardware/software co-simulation // Conf. on Design Automation. – 1994. – P. 439-440. – DOI: 10.1109/DAC.1994.204143.
10. *Liu J., Lajolo M., Sangiovanni-Vincentelli A.* Software timing analysis using HW/SW cosimulation and instruction set simulator // Proc. Int. Workshop on hardware/software codesign. – 1998. – P. 65-69. – DOI: 10.1145/278241.278299.
11. *Formaggio L, Fummi F., Pravadell G.* A timing-accurate HW/SW co-simulation of an ISS with SystemC // Proc. IEEE/ACM/IFIP Int. Conf. on Hardware/Software Codesign and System Synthesis. – 2004. – P. 152-157. – DOI: 10.1145/1016720.1016759.
12. *Zivojnovic V., Meyr H.* Compiled HW/SW co-simulation // Proc. Design Automation Conf. – 1996. – P. 690-695. – DOI: 10.1145/240518.240649.
13. *Lajolo M., Lazarescu M., Sangiovanni-Vincentelli A*. A compilation-based software estimation scheme for hardware/software co-simulation // Int. Conf. on Hardware/Software Codesign. – 1999. – P. 85-89. – DOI: 10.1145/301177.301493.
14. *Bringmann O., Ecker W., Gerstlauer A., [et al.].* The next generation of virtual prototyping: Ultra-fast yet accurate simulation of HW/SW systems // Design, Automation & Test in Europe Conf. & Exhibition. – 2015. – P. 1698-1707. – DOI: 10.7873/DATE.2015.1105.

15. *Muttillo V., Pomante L., Santic M., Valente, G.* SystemC-based co-simulation / analysis for system-level hardware/software co-design // Computers and Electrical Engineering. – 2023. –Vol. 110. – 108803. – DOI: 10.1016/j.compeleceng.2023.108803.
16. dsPIC33FJXXXGPX06/X08/X10 high-performance, 16-bit digital signal controllers: Data sheet DS70286C. – Microchip Technology Inc., 2009. – 320 p. – URL: https://ww1.microchip.com/ downloads/en/DeviceDoc/70286C.pdf (дата обращения: 20.05.2024).
17. dsPIC33F/PIC24H family reference manual. Section 22. Direct memory access (DMA) DS70182A. – Microchip Technology Inc., 2006. – 60 p. – URL: http:// ww1.microchip.com/ downloads/en/ devicedoc/70183d.pdf (дата обращения: 20.05.2024).
18. dsPIC33F/PIC24H family reference manual. Section 16. Analog-to-digital converter (ADC) DS70183D. – Microchip Technology Inc., 2006-2012. – 88 p. – URL: https:// ww1.microchip.com/ downloads/en/DeviceDoc/70182A.pdf (дата обращения: 20.05.2024).
19. dsPIC33F family reference manual. Section 13. Output compare DS70209A. – Microchip Technology Inc., 2007. – 24 p. – URL: http://ww1.microchip.com/downloads/ en/DeviceDoc/70209A.pdf (дата обращения: 20.05.2024).
20. MPLAB starter kit for dsPIC® digital signal controllers: Users Guide DS51700A. – Microchip Technology Inc., 2008. – 42 p. – URL: https://ww1.microchip.com /downloads/en/DeviceDoc/51700B.pdf (дата обращения: 20.05.2024).

REFERENCES

1. GOST R 70607–2022. Sistemy avtomatizirovannogo proektirovaniya elektroniki. Sostav i struktura sistemy avtomatizirovannogo proektirovaniya pechatnykh uzlov [GOST R 70607–2022 Electronics automated design systems. Composition and structure of the computer-aided design of printed circuit assemblies]. Moscow: Ros. in-t standartizatsii, 2023, 8 p.
2. GOST R 70884–2023. Sistemy avtomatizirovannogo proektirovaniya elektroniki. Informatsionnoe obespechenie. Modeli SPICE. Obshchie polozheniya [GOST R 70884–2023 Electronics automated design systems. Information support. SPICE models. General provisions]. Moscow: Ros. in-t standartizatsii, 2023, 8 p.
3. PSpice user guide. Product version 17.2–2016. Cadence, 2016, 900 p. Available at: https:// resources.pcb.cadence.com/i/1180526-pspice-user-guide/25 (accessed 20 May 2024).
4. *Vas'kov A.S., Smirnov B.I.* Ispol'zovanie paketa Proteus dlya proektirovaniya virtual'nykh modeley mikroprotsessornykh ustroystv [Using the Proteus package for designing virtual models of microprocessor devices], *Izvestiya vuzov Rossii. Radioelektronika* [News of Russian universities. Radioelectronics], 2013, Issue 2, pp. 38-43.
5. GOST R 70806–2023. Sistemy avtomatizirovannogo proektirovaniya elektroniki. Informatsionnoe obespechenie. Poryadok razrabotki modeley SPICE. Postanovka zadachi [GOST R 70806–2023 Electronics automated design systems. Information support. Procedure for development of SPICE models. Formulation of the problem]. Moscow: Ros. in-t standartizatsii, 2023, 11 p.
6. PSpice device modeling interface API reference. Product version 17.2–2016. Cadence, 2016, 93 p.
7. Virtual prototyping in PSpice: Application note, V1. FlowCAD, 2016, 72 p. Available at: https://www.flowcad.de/AN/FlowCAD-AN_Device-Modeling-Interface.pdf (accessed 20 May 2024).
8. *Teich J.* Hardware/software codesign: the past, the present, and predicting the future, *Proc. IEEE,* 2012, Vol. 100, pp. 1411-1430. DOI: 10.1109/JPROC.2011.2182009.
9. *Rowson J.A.* Hardware/software co-simulation, *Conf. on Design Automation,* 1994, pp. 439-440. DOI: 10.1109/DAC.1994.204143.
10. *Liu J., Lajolo M., Sangiovanni-Vincentelli A.* Software timing analysis using HW/SW cosimulation and instruction set simulator, *Proc. Int. Workshop on hardware/software codesign,* 1998, pp. 65-69. DOI: 10.1145/278241.278299.
11. *Formaggio L, Fummi F., Pravadell G.* A timing-accurate HW/SW co-simulation of an ISS with SystemC, *Proc. IEEE/ACM/IFIP Int. Conf. on Hardware/Software Codesign and System Synthesis,* 2004, pp. 152-157. DOI: 10.1145/1016720.1016759.
12. *Zivojnovic V., Meyr H.* Compiled HW/SW co-simulation, *Proc. Design Automation Conf.,* 1996, pp. 690-695. DOI: 10.1145/240518.240649.
13. *Lajolo M., Lazarescu M., Sangiovanni-Vincentelli A*. A compilation-based software estimation scheme for hardware/software co-simulation, *Int. Conf. on Hardware/Software Codesign,* 1999, pp. 85-89. DOI: 10.1145/301177.301493.
14. *Bringmann O., Ecker W., Gerstlauer A., [et al.].* The next generation of virtual prototyping: Ultra-fast yet accurate simulation of HW/SW systems, *Design, Automation & Test in Europe Conf. & Exhibition,* 2015, pp. 1698-1707. DOI: 10.7873/DATE.2015.1105.

15. *Muttillo V., Pomante L., Santic M., Valente, G.* SystemC-based co-simulation / analysis for system-level hardware/software co-design, *Computers and Electrical Engineering,* 2023, Vol. 110, 108803. DOI: 10.1016/j.compeleceng.2023.108803.
16. dsPIC33FJXXXGPX06/X08/X10 high-performance, 16-bit digital signal controllers: Data sheet DS70286C. Microchip Technology Inc., 2009, 320 p. Available at: https://ww1.microchip.com/downloads/en/DeviceDoc/70286C.pdf (accessed 20 May 2024).
17. dsPIC33F/PIC24H family reference manual. Section 22. Direct memory access (DMA) DS70182A. Microchip Technology Inc., 2006, 60 p. Available at: http:// ww1.microchip.com/ downloads/en/devicedoc/70183d.pdf (accessed 20 May 2024).
18. dsPIC33F/PIC24H family reference manual. Section 16. Analog-to-digital converter (ADC) DS70183D. Microchip Technology Inc., 2006-2012, 88 p. Available at: https:// ww1.microchip.com/downloads/en/DeviceDoc/70182A.pdf (accessed 20 May 2024).
19. dsPIC33F family reference manual. Section 13. Output compare DS70209A. Microchip Technology Inc., 2007, 24 p. Available at: http://ww1.microchip.com/downloads/ en/DeviceDoc/70209A.pdf (accessed 20 May 2024).
20. MPLAB starter kit for dsPIC® digital signal controllers: Users Guide DS51700A. Microchip Technology Inc., 2008, 42 p. Available at: https://ww1.microchip.com /downloads/en/DeviceDoc/51700B.pdf (accessed 20 May 2024).

Статью рекомендовал к опубликованию д.т.н., профессор Ю.А. Кравченко.

Хлуденев Александр Владимирович – Оренбургский государственный университет; e-mail: avhludenev@yandex.ru; г. Оренбург, Россия, тел.: +73532372874; кафедра промышленной электроники и информационно-измерительной техники; к.т.н.; доцент.

Сильвашко Сергей Анатольевич – e-mail: sisean@yandex.ru; тел.: +73532372874; кафедра промышленной электроники и информационно-измерительной техники; к.т.н.; доцент.

Khludenev Alexander Vladimirovich – Orenburg State University; e-mail: avhludenev@yandex.ru; Orenburg, Russia; phone: +73532372874; the department of Industrial Electronics and Information Measuring Engineering; cand. of eng. sc.; associate professor.

Silvashko Sergey Anatolievich – e-mail: sisean @yandex.ru; phone: +73532372874; the department of Industrial Electronics and Information Measuring Engineering; cand. of eng. sc.; associate professor.

УДК 004.912 DOI 10.18522/2311-3103-2024-4-110-122

Ю.М. Вишняков, Р.Ю. Вишняков

ФОРМАЛИЗАЦИЯ РАСПОЗНАВАНИЯ И ИДЕНТИФИКАЦИИ СЕМАНТИЧЕСКИХ ОБЪЕКТОВ В ЕСТЕСТВЕННО-ЯЗЫКОВЫХ ТЕКСТОВЫХ ПОТОКАХ

Участившиеся случаи совершаемых в киберпространстве преступлений, в особенности, в социальных сетях и различного рода мессенджерах требуют создания адекватных и эффективных мер противодействия. Рост киберпреступлений настолько большой, что они уже могут нанести невосполнимый урон государству и обществу. Однако выявление подобного рода преступлений и преступных деяний, наталкивается на большие трудности, так как преступники присутствуют в социальных сетях виртуально и лингвистически, используют всячески их возможности и особенности для сокрытия следов своих преступлений. И, тем не менее, такими инструментами противодействия могли бы быть различного рода распознаватели и идентификаторы, способные автоматически обрабатывать естественный язык, выделять в нем специфические смысловые черты преступных деяний, распознавать и идентифицировать их. Поскольку по многим параметрам и обстоятельствам применительно к данным ситуациям использования нейросетевого подхода представляется нецелесообразным, в предлагаемой работе разрабатывается собственный формальный метод проектирования распознавателя для идентификации в текстовых потоках семантических объектов по их лингвистическим следам. Вводятся такие формальные понятия, как формальная модель семантического объекта, функция поведения, сценарий, лингвистический след, функция распознавания. Рассуждения строятся на теоретико-множественных положениях вычислительной теории семантической интерпретации и используют вычислительное представ-