Краткий обзор отладочной среды Code Composer Studio 4

Интегральная среда проектирования Code Composer Studio версии 4 (CCSv4) предназначена для разработки программного обеспечения ко всем процессорам, выпускаемым компанией Texas Instruments Incorporated (TI). Среда CCSv4 включает такие инструменты, как редактор исходных текстов, компилятор, линковщик, отладчик, симуляторы для всех типов процессоров от TI, средства визуализации и целый ряд других вспомогательных инструментов. Это единый графический пользовательский интерфейс, позволяющий производить пошаговую разработку и отладку программного кода. Интуитивно понятные инструменты позволяют разработчикам быстро начать свою собственную разработку, постепенно осваивая новые функциональные возможности и улучшая производительность своих решений.

Игорь ГУК
gii@scanti.ru

Средство разработки CCSv4 базируется на программном обеспечении с открытым исходным кодом Eclipse. Использование именно этого продукта обосновано тем, что Eclipse обеспечивает превосходную структуру программного обеспечения (ПО), которую поддерживают многие продавцы встраиваемых решений. В CCSv4 совмещаются преимущества Eclipse и лучшие решения от TI, что позволяет реализовать удобную и многофункциональную среду проектирования. На рис. 1 показано главное окно CCSv4.

Встроенный отладчик адаптируется к конкретному типу DSP-процессора, а также умеет выставлять многофункциональные контрольные точки и работать с ними, что значительно облегчает отладку программного кода. Контрольные точки можно привязать к строкам С-кода, локальным переменным или регистрам. Окно просмотра памяти позволяет увидеть каждый ее уровень, что упрощает отладку сложных участков кода, особенно с использованием кэша. Среда CCSv4 поддерживает возможность работы с многопроцессорными и многоядерными системами. Глобальные контрольные точки и синхронизация операций обеспечивают полный контроль над такими системами.

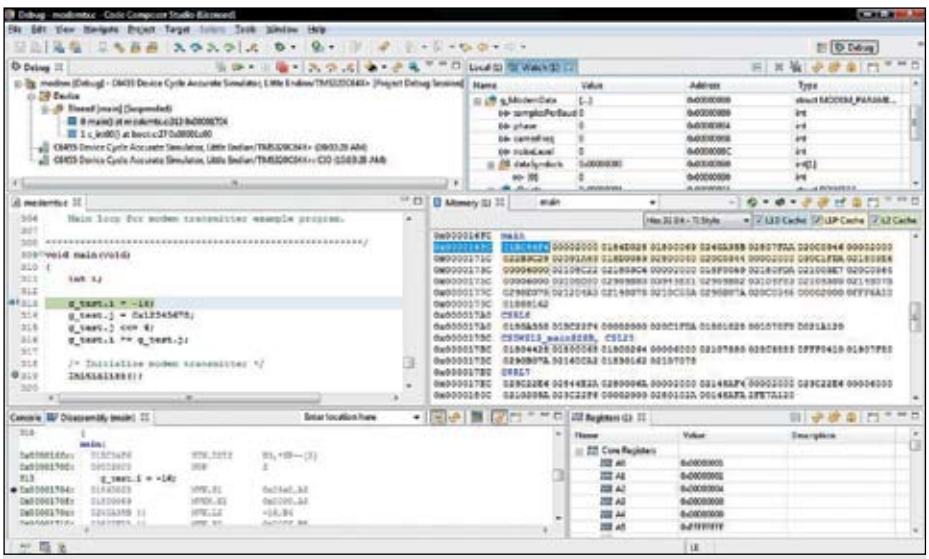

Рис. 1. Основное окно среды разработки Code Composer Studio

Встроенный интерактивный профилировщик помогает быстро определить наиболее ресурсоемкие места в коде программы и правильно распределить производительность DSP. Он позволяет легко тестировать производительность любых C/C++ функций в приложении при их циклическом использовании, или в других случаях, например, при использовании кэша, конвейера, ветвлений. При проведении профилирования можно выделять диапазоны кода, чтобы получить более полную информацию о задействованных ресурсах во время оптимизации и отладить код. Для увеличения эффективности работы возможности профилирования доступны на любом этапе разработки.

Некоторые задачи, такие как многочасовое или даже многодневное тестирование, могут выполняться без участия разработчика. Для выполнения данных операций среда разработки должна иметь возможность их автоматизации. У CCSv4 есть полноценная скриптовая среда, помогающая создавать и выполнять сценарии для таких задач, как сопоставительный анализ и тестирование производительности. Отдельное окно позволяет вводить одиночные скриптовые команды или выполнять сценарии в пределах среды разработки.

Среда CCSv4 позволяет проводить анализ изображений и графическую визуализацию обрабатываемых данных, а также отображать данные на мониторе и автоматически их обновлять во время выполнения программного кода. Кроме того, CCSv4 может показывать видео в различных форматах. На рис. 2 представлено окно визуализации.

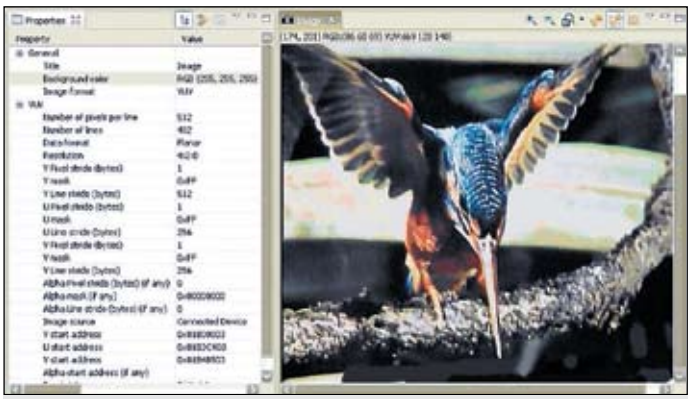

Рис. 2. Окно визуализации CCSv4

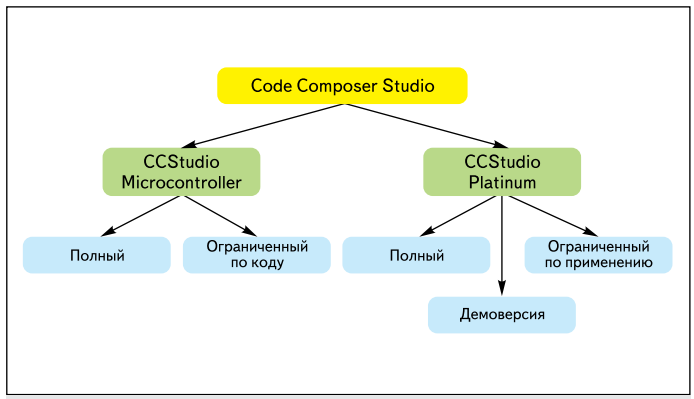

Рис. 3. Варианты поставки CCSv4

Компиляторы C/C++ от компании TI оптимизируются под конкретный тип процессора таким образом, чтобы максимизировать производительность кода. Эти компиляторы используют как классические методы оптимизации, так и методы, определяемые выбранной архитектурой DSP-процессора. В частности, оптимизация включает:

- упрощение выражений;
- программный конвейер;
- сокращение выражений;
- автоинкрементацию адресов;
- распределение регистров;
- аппаратные циклы;
- “in-lining”-функции;
- векторизацию циклов.

Компиляторы TI также могут выполнять такую оптимизацию программы, при которой оценивается эффективность кода на прикладном уровне. Программная оптимизация позволяет оценить издержки при взаимодействии разработанного ПО с той системой, куда оно встраивается. Это во многих случаях позволяет увеличить производительность системы в целом.

Компиляторы C/C++ для ARM-ядра и микроконтроллеров оптимизированы с целью уменьшения размера кода и повышения его эффективности. Они являются одним из лучших решений по производительности и совместимости кода для промышленных решений.

Встроенный симулятор позволяет начать разработку без наличия реальной отладочной платы. Кроме того, симулятор обеспечивает широкие возможности по визуализации процесса выполнения программного кода. Предлагается несколько вариантов симуляторов для того, чтобы можно было выбрать между точностью оценки кода, скоростью анализа и возможностями по моделированию работы с периферией; одни симуляторы являются идеальными для сопоставительного анализа алгоритма, другие — для детального моделирования системы.

Во все процессоры TI встроена передовая система эмуляции. Она включает:

- JTAG-интерфейс (1149.1) и периферийное сканирование;
- доступ к регистрам и памяти;
- режим реального времени, предусматривающий отладку кода с прерываниями, которые должны быть разрешены (этот режим позволяет приостанавливать выполнение второстепенного кода, продолжая выполнять критические прерывания);
- многоядерные операции, такие как синхронный запуск, отладка по шагам и остановка;
- межъядерный переключатель, который обеспечивает запуск одного ядра и остановку другого.

Функциональный менеджер событий (Advanced Event Triggering, AET) позволяет останавливать центральный процессор или вызывать другие события, исходя из некоторых, определенных заранее, условий — например, появление недействительных данных или запрос доступа к памяти. Это позволяет более точно определить эксплуатационные качества программного кода и подсчитывать определенные системные события (например, обращение к кэшу).

Среда CCSv4 поддерживает режим трассировки (Processor Trace), что помогает разработчикам находить «невидимые» до этого и трудные для поиска другими методами ошибки — состязание событий, «дрожание» сигнала в реальном времени, переполнение стека, бесконтрольный код и ложные прерывания без остановки процессора. Трассировка — это метод для отладки DSP-процессора без вмешательства и без изменений его поведения в реальном времени. Данный режим помогает получить максимальную производительность от программного кода и провести оптимизацию использования кэша для многоканальных приложений. Трассировка поддерживает экспорт программного кода и данных, тактирование и выбор процессора, а также позволяет обрабатывать системные события и прерывания. Режим трассировки поддерживается любым внешним JTAG-эмулятором XDS560, а на отдельных чипах может сохранять данные во встроенном трассировочном буфере (Embedded Trace Buffer, ETB).

Операционная система DSP/BIOS 6.x, поставляемая в составе новой версии CCSv4, является современной расширяемой операционной системой реального времени (ОСРВ), которая поддерживает такие ядра, как ARM926, ARM Cortex M3, C674x, C64x+, C672x и C28x. DSP/BIOS 6.x обеспечивает поддержку гораздо более широкого — относительно DSP/BIOS 5.x — списка ядер, гарантируя при этом более быстрое и гибкое управление памятью, событиями и приоритетами. Новая версия ОСРВ обеспечивает совместимость с предыдущим поколением операционной системы, что помогает легко адаптировать прикладное ПО к новой платформе. Операционная система DSP/BIOS 6.x является мощным инструментом, который позволяет разрабатывать производительный код в короткое время. Изучение даже основных возможностей этой системы выходит за рамки данной статьи. Получить более детальную информацию можно на сайте компании TI [1]. В меню ***Products*** выберите раздел ***Processors***, а там — пункт ***Digital Signal Processing***. На новой страничке найдите пункт ***DSP Design Support*** (горизонтальное меню вверху страницы). Затем в разделе ***Tools and Software*** выберите пункт ***Operating Systems (OS/RTOS)*** и перейдите на закладку ***DSP/BIOS***.

Данная версия CCSv4 уже доступна. Существуют несколько вариантов ее поставки. Они представлены на рис. 3.

Выделяются два направления развития CCSv4: только для микроконтроллеров и для всех процессоров в целом. К микроконтроллерам относятся семейства MSP430, C2000 и Stellaris. Следует отметить, что микроконтроллеры Stellaris построены на базе ядра Cortex M3, и их поддержка будет включена в CCSv4 в четвертом квартале этого года или в первом следующего.

Ограниченная по коду версия для микроконтроллеров позволяет разрабатывать прикладное ПО размером не более 32 кбайт для процессоров на базе ядра C28x и 16 кбайт для MSP430. Кроме того, можно скачать полнофункциональную версию, ограниченную по применению. Эта версия работает только

Таблица. Стоимость различных вариантов поставки CCS

№ п/п	Код продукта	Наименование продукта	Стоимость, USD	Примечание
Поддержка всех семейств процессоров и микроконтроллеров компании TI				
1	TMDSCCSALL-1	Code Composer Studio v3.3	1995	С годовой поддержкой
2	TMDSSUBALL	Годовая поддержка Code Composer Studio™	600	
3	TMDSCCS-ALLN01	Code Composer Studio IDE v4.0	1995	Одиночная лицензия (N01)
4	TMDSCCS-ALLN01D	Code Composer Studio IDE v4.0	1895	Одиночная лицензия (N01D) — без DVD
5	TMDSCCS-ALLF01	Code Composer Studio IDE v4.0	2995	Плавающая лицензия для 1 пользователя (F01)
6	TMDSCCS-ALLF03	Code Composer Studio IDE v4.0	7995	Плавающая лицензия для 3 пользователей (F03)
7	TMDSCCS-ALLF05	Code Composer Studio IDE v4.0	11 495	Плавающая лицензия для 5 пользователей (F05)
8	TMDSCCS-ALLF10	Code Composer Studio IDE v4.0	20 995	Плавающая лицензия для 10 пользователей (F10)
9	TMDSCCS-ALLF25	Code Composer Studio IDE v4.0	44 995	Плавающая лицензия для 25 пользователей (F25)
Поддержка только микроконтроллеров компании TI				
1	TMDSCCS2000-1	C2000 Code Composer Studio™	495	С годовой поддержкой
2	TMDSSUB2000	Годовая поддержка Code Composer Studio™ C2000	495	
3	TMDSCCS-MCUN01D	Code Composer Studio IDE v4.0 — MCU	445	Одиночная лицензия (N01)
4	TMDSCCS-MCUN01	Code Composer Studio IDE v4.0 — MCU	495	Одиночная лицензия (N01D) — без DVD
5	TMDSCCS-MCUF01	Code Composer Studio IDE v4.0 — MCU	795	Плавающая лицензия для 1 пользователя (F01)
6	TMDSCCS-MCUF03	Code Composer Studio IDE v4.0 — MCU	1995	Плавающая лицензия для 3 пользователей (F03)
7	TMDSCCS-MCUF05	Code Composer Studio IDE v4.0 — MCU	2995	Плавающая лицензия для 5 пользователей (F05)
8	TMDSCCS-MCUF10	Code Composer Studio IDE v4.0 — MCU	5495	Плавающая лицензия для 10 пользователей (F10)
9	TMDSCCS-MCUF25	Code Composer Studio IDE v4.0 — MCU	11 995	Плавающая лицензия для 25 пользователей (F25)

с эмулятором класса XDS100 и не со всеми типами процессоров. Демонстрационная версия имеет ограничение по сроку использования (на данный момент — 120 дней).

Следует отметить, что версия 4 имеет и серверный вариант инсталляции. Предусмотрена возможность приобретения различного количества так называемых «плавающих» лицензий, что может быть удобно для коллективов программистов.

Стоимость различных вариантов CCS на момент написания статьи представлена в таблице. Данные взяты с сайта производителя, поэтому необходимо их уточнение у официальных дистрибьюторов.

Для того чтобы загрузить один из бесплатных вариантов CCSv4, необходимо зарегистрироваться на сайте TI. Иметь регистрацию в любом случае полезно: можно подписаться на рассылку новостей, а также скачивать различную документацию и заказывать бесплатные учебные материалы.

Затем необходимо установить отладочную среду. При установке будет предложен ряд эмуляторов от различных производителей. Рекомендуется ограничиться минимальным их количеством, так как если указанного эмулятора не будет в системе, может произойти «зависание» компьютера. Примерный набор эмуляторов, не вызывающий «зависания» при установке CCSv4 даже при их отсутствии в системе, приведен на рис. 4.

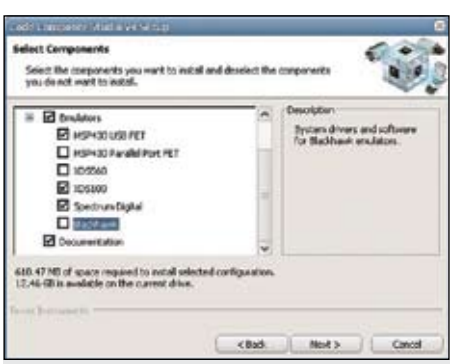

Рис. 4. Минимально необходимый набор компонентов при установке CCSv4

При первом запуске отладочной среды появится информационное окно системы (рис. 5).

В этом окне, переходя от пункта к пункту, можно ознакомиться с основными возможностями отладочной среды и правилами построения проекта. Чтобы попасть в рабочее пространство CCSv4, необходимо закрыть информационное окно, щелкнув мышкой на крестике в левом верхнем углу (отметка красным кружком на рис. 5).

Теперь рассмотрим небольшой пример использования CCSv4. Для его реализации понадобятся:

1. Бесплатная версия среды разработки с ограничением на максимальный размер кода для микроконтроллеров — TMDFCCS-MCULTD.
2. Отладочный комплект TMS320F28335 Experimenter Kit. Его описание дано в [2]. Комплект состоит из базовой платы и модуля расширения, который вставляется

Рис. 5. Окно CCSv4 при первом запуске системы

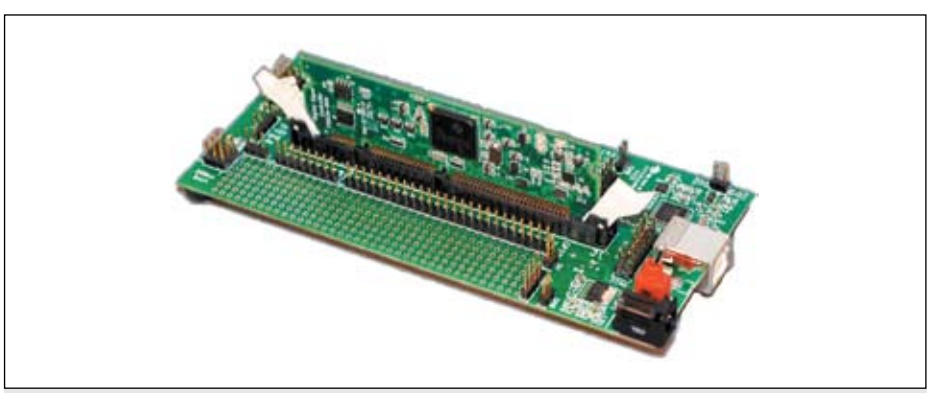

Рис. 6. Отладочный комплект TMS320F28335 Experimenter Kit

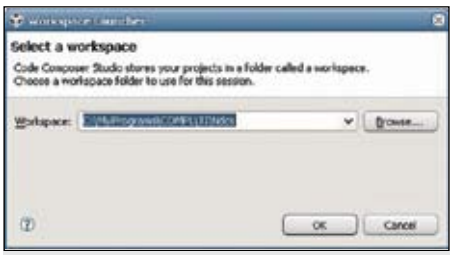

Рис. 7. Определение местоположения рабочей области

в DIMM-разъем. Причем можно приобретать дополнительные модули расширения отдельно, за меньшую стоимость и для различных типов микроконтроллеров семейства C2000, то есть, приобретя один такой комплект, в дальнейшем при необходимости можно освоить новый тип процессоров данного семейства, докупая только модули расширения. На рис. 6 показана отладочная плата с установленным модулем расширения.

Питание отладочного комплекта осуществляется через USB-интерфейс. Кроме того, в базовую плату встроен эмулятор класса XDS100. Отметим, что имеется возможность подключать как внешний блок питания, так и дискретный эмулятор. Для этого на базовой плате предусмотрены специальные разъемы. Данная функция может понадобиться при макетировании собственных устройств на основе стандартного набора. Для макетирования предназначено несколько рядов металлизированных сквозных отверстий и штыревых разъемов (хорошо видных на рис. 6), куда подводятся выводы микроконтроллера.

3. Программный пакет с примерами кода C2833x/C2823x C/C++ Header Files and Peripheral Examples. Скачать его (без регистрации) можно по адресу [3]. Загрузите, распакуйте и установите его на свой ПК.

При запуске CCSv4 появится диалоговое окно с предложением выбрать рабочую область (рис. 7). В этой папке будут сохраняться все персональные настройки для CCSv4. Также — после выключения CCSv4 — в выбранную папку по умолчанию будут сохраняться все настройки разрабатываемых проектов. Например, если до выключения CCSv4 происходила работа с несколькими проектами и при этом были открыты окна памяти, графики, переменных, то после закрытия все настройки сохранятся, а при повторном запуске CCSv4 все восстановится в том виде, который был перед закрытием.

Рабочая область автоматически сохраняется при закрытии CCSv4. Существует возможность работать с несколькими рабочими областями: создавать новые и переключаться между уже существующими можно через главное меню ***File->Switch Workspace***. Рабочая область в рассматриваемой среде не является портативной, так как она включает специфические настройки компьютерной системы. Поэтому нельзя сохранить рабочую область, заархивировать ее и передать другому пользователю.

Для первого знакомства с возможностями CCSv4 будем использовать проект ***cpu_timer***, входящий в пакет примеров "*C2833x/2823x C/C++ Header Files and Peripheral Examples*", который к этому моменту должен быть уже установлен.

Вначале выберем в главном меню CCSv4 пункт ***Project->Import Existing CCS/CCE Eclipse Project*** (рис. 8).

Затем в появившемся окне ***Import*** (рис. 9) необходимо определить папку проекта — ***Select root directory***: нажав кнопку ***Browse*** и выбрав в стандартном диалоговом окне путь ***<папка установки>/DSP2833x_examples_ccsv4/cpu_timer***, подтвердите выбор кнопкой ***OK***. В области ***Projects*** окна ***Import*** галочкой необходимо отметить проект ***Example_2833xCpuTimer***. НЕ ОТМЕЧАТЬ пункт ***Copy projects into workspace***. Нажать кнопку ***Finish***.

В CCSv4 в области ***C/C++ Projects*** (рис. 10) должны отобразиться все файлы проекта:

- Раздел ***Binaries*** — после компиляции в этой папке будет находиться бинарный исполняемый файл ****.out***.
- Раздел ***Includes*** — здесь перечислены все директории, ссылки на которые включаются в опциях компиляции. Дополняется список всеми файлами ****.h*** и ****.inc***, которые упоминаются в проекте.
- Раздел ***Debug*** — содержит все файлы компиляции, включая ****.map*** и ****.obj***.
- Далее перечисляются все исходные файлы. Если файлы с исходным кодом содержат функции, включают другие файлы или структуры, то можно, нажав крестик напротив имени файла, увидеть список всех входящих в него объектов. Двойным щелчком мыши на имени объекта осуществляется переход к тому месту в программном коде, где размещены соответствующая функция, подключаемый файл или структура.

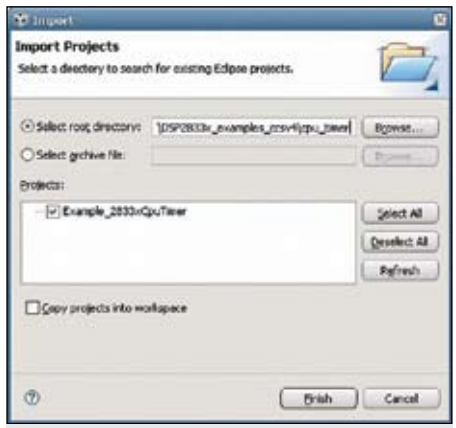

Рис. 9. Выбор папки проекта

Следует отметить, что разделы ***Binaries*** и ***Debug*** появляются только после первой успешной компиляции проекта.

Теперь пришло время настроить параметры подключения эмулятора к отладочной плате. Для этого необходимо выполнить следующее:

- Подключить отладочную плату к компьютеру при помощи USB-кабеля из комплекта поставки. При первом подключении платы должно произойти ее автоматическое распознавание операционной системой Windows, так как драйверы для нее установились при инсталляции CCSv4.

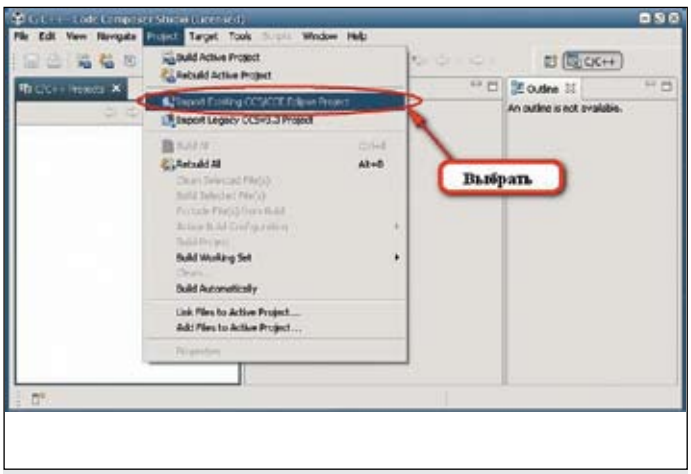

Рис. 8. Выбор проекта

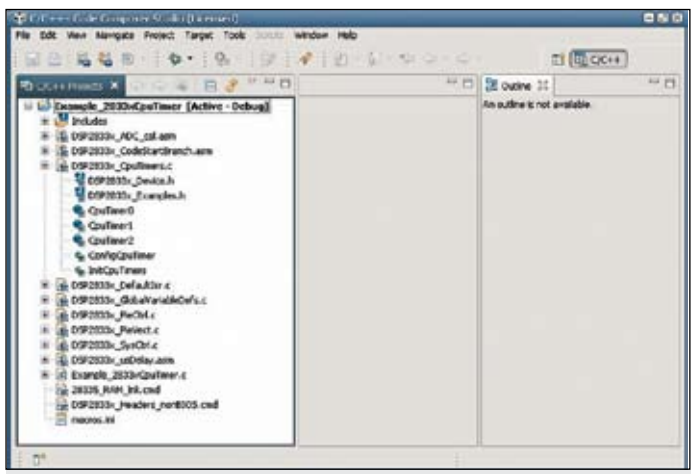

Рис. 10. Окно C/C++ Projects

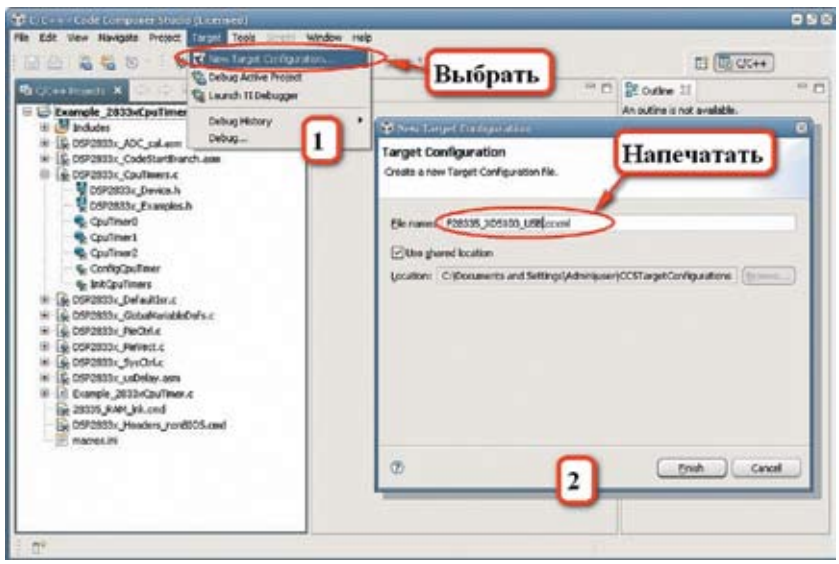

Рис. 11. Создание новой конфигурации подключения отладочной платы

- Открыть окно конфигурации подключения через главное меню CCSv4, выбрав пункт ***Target->New Target Configuration*** (рис. 11, сноска 1).

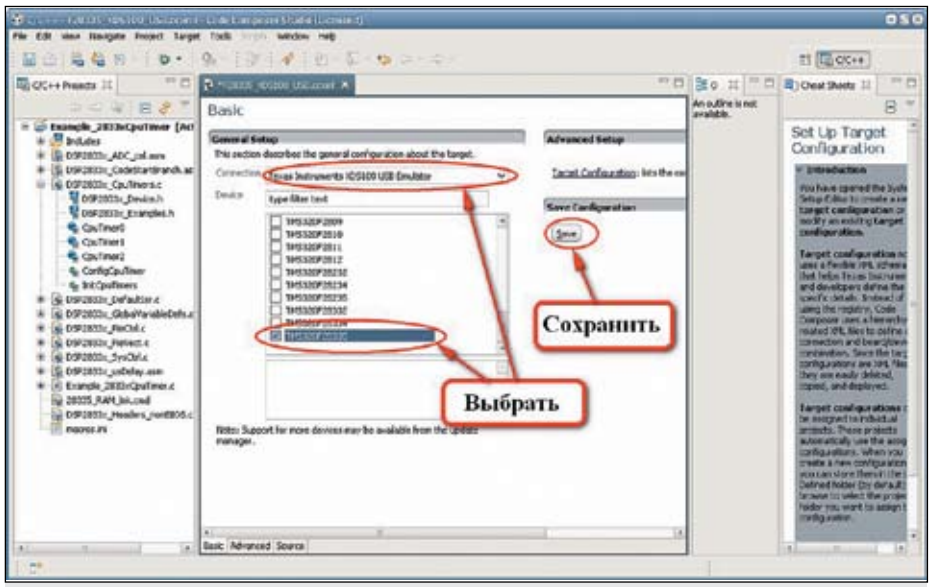

Рис. 12. Определение типов эмулятора и микроконтроллера

- Ввести имя файла конфигурации (рис. 11, сноска 2). Имя должно иметь суффикс .ccxml. Поставить галочку в боксе ***Use default location*** и нажать ***Finish***.
- В появившемся окне выбрать требуемый тип эмулятора из выпадающего списка ***Connection***. В списке ***Device*** отметить нужный тип процессора. В рассматриваемом примере выбраны ***Texas Instruments XDS100 USB Emulator*** и ***TMS320F28335*** (рис. 12).
- Сохранить настройки подключения, нажав кнопку ***Save*** в разделе ***Save Configuration*** (рис. 12). Воспользуйтесь полосой прокрутки внизу окна, если раздел ***Save Configuration*** виден не целиком.
- Посмотреть настройки подключения можно в меню ***View->Target Configuration***.
- В разделе ***User-Defined*** (рис. 13) показан новый конфигурационный файл подключения ****.ccxml***. Необходимо щелкнуть на нем правой кнопкой мыши, а затем выбрать пункт ***Link File to Project->Example_2833xCpuTimer*** (для того чтобы всегда использовать эту конфигурацию для данного проекта) или пункт ***Set as Default*** (если необходимо использовать этот эмулятор и GEL-файл в качестве настройки подключения по умолчанию для всех открываемых в будущем проектов, не имеющих ассоциируемых с ними файлов конфигурации).

Теперь можно скомпилировать и загрузить проект в подключенную отладочную плату, для чего следует нажать зеленую кнопку с изображением «жучка» (***Debug***) на панели быстрых кнопок (рис. 14). При этом произойдут три операции: компиляция проекта, подключение к отладочной плате и загрузка в нее программного кода.

Операции можно выполнить и поочередно. Компиляция проекта запускается через главное меню путем выбора пункта ***Project->Build Active Project*** (рис. 15).

Затем выбираем в подменю кнопку с зеленым «жучком» на панели быстрых кнопок ***Launch TI Debugger*** (рис. 16). После этого отладочная среда перейдет в режим ***Debug***, и появится окно дизассемблера.

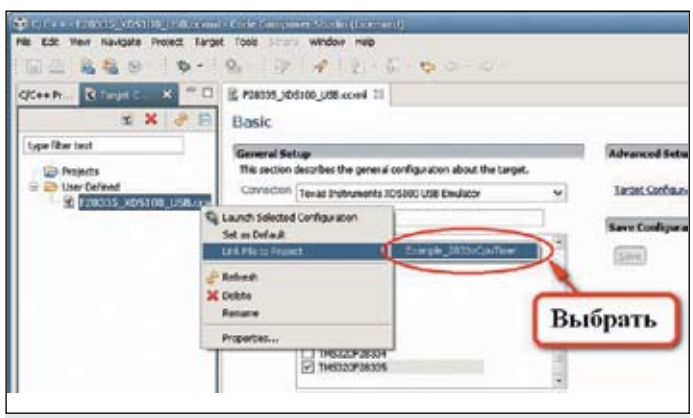

Рис. 13. Подключение файла конфигурации к проекту

Рис. 14. Компиляция, подключение и загрузка проекта

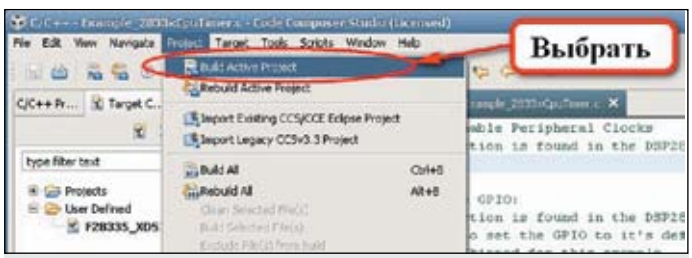

Рис. 15. Компиляция проекта без автоматических подключения и загрузки

Рис. 16. Перевод отладочной среды CCSv4 в режим отладки

Рис. 17. Подключение к отладочной плате

Следующий шаг — это подключение CCSv4 к отладочной плате, для чего необходимо выбрать в главном меню пункт ***Target->Connect Target*** (рис. 17).

И, наконец, загружаем бинарный код в отладочную плату, выбрав в главном меню пункт ***Target->Load Program*** и указав в появившемся окне файл ***Example_2833xCpuTimer.out***, который находится в директории ***/cpu_timer/Debug***.

Для повторного запуска проекта требуется произвести сброс процессора, выбрав в главном меню пункт ***Target->Reset->Reset CPU***, и, перегрузив программный код, выбрать ***Target->Reload Program***.

Отладочная среда CCSv4 позволяет добавлять переменные в окно наблюдения (***Watch Window***) и отслеживать их изменения при работе программы в реальном времени. Для того чтобы добавить переменные в окно наблюдения, необходимо:

- Нажать значок ***C/C++*** в верхнем правом углу экрана для перехода в режим ***C/C++*** (рис. 18, сноска 1), перейти на закладку с отображением файлов проекта (рис. 18, сноска 2) и открыть файл с исходным кодом (рис. 18, сноска 3).
- В файле ***Example_2833xCpuTimer.c*** в комментариях найти блок ***Watch Variables.*** Выделить ***CpuTimer0.InterruptCount***, щелкнуть правой кнопкой мыши и во всплывающем меню выбрать ***Add Watch Expressio*** (рис. 19). Затем повторить эти действия для ***CpuTimer1.InterruptCount*** и ***CpuTimer2.InterruptCoun***. Эти переменные появятся в окне просмотра (***Watch Window***).
- Переключиться в режим ***Debug***, нажав значок ***Debug*** с зеленым «жучком» (рис. 18, сноска 1).

Запускаем проект на выполнение, выбрав пункт ***Target->Run*** (рис. 20, сноска 1). Остановить выполнение проекта можно через пункт ***Target->Halt*** (рис. 20, сноска 2). К сожалению, эмулятор XDS100 обладает ограниченными возможностями по отладке в реальном времени: он не поддерживает функции RTDX, поэтому наблюдать изменение переменных можно только запуская и останавливая выполнение программного кода.

Если использовать внешний эмулятор более высокого класса, например, XSD510, то можно реализовать режим отладки в реальном времени. Для этого, во-первых, необходимо разрешить этот режим, выбрав в главном меню пункт ***Tools->Debugger Options***. В открывшемся окне спуститься вниз, до бокса ***Realtime Option***s, и отметить пункт ***Enable silicon real-time mode (service critical interrupts when halted, allow debugger accesses while running)*** (рис. 21). Затем в появившемся окне выбрать "Yes" для подтверждения переключения в режим реального времени.

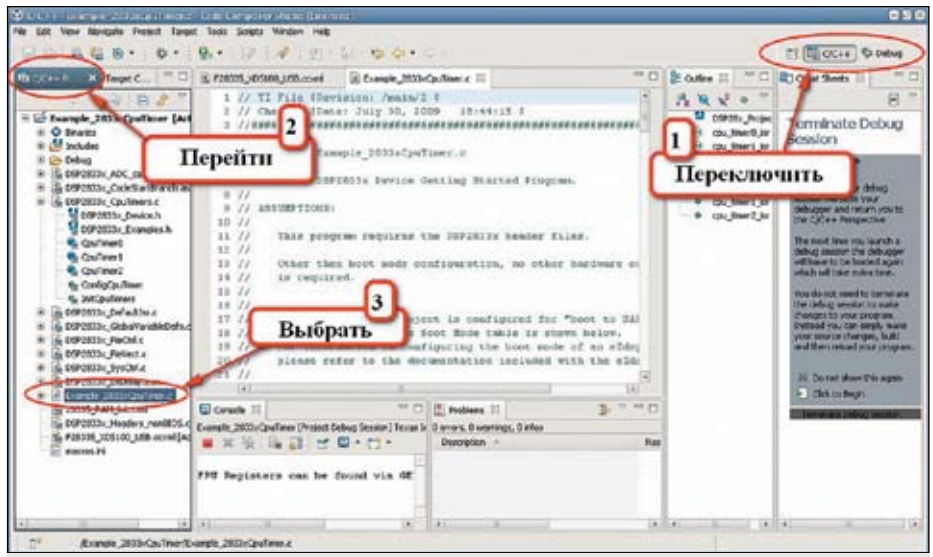

Рис. 18. Переключение CCSv4 между режимами "Debug" и "C/C++"

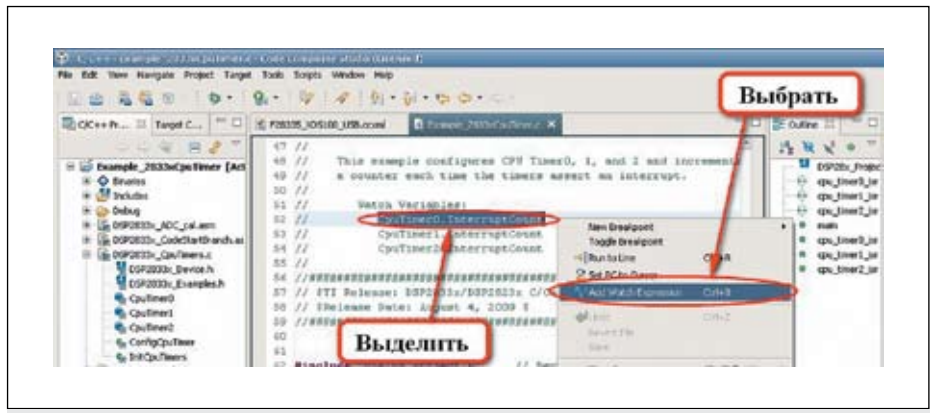

Рис. 19. Добавление переменных в окно наблюдения

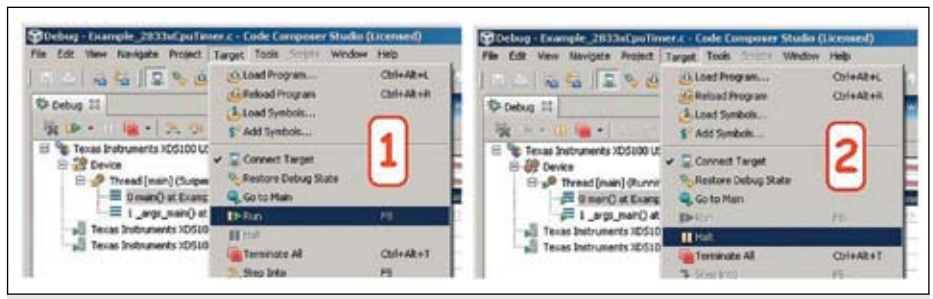

Рис. 20. Запуск и остановка выполнения программного кода

Рис. 21. Разрешение режима реального времени для мониторинга переменных

Рис. 24. Инкрементация значения переменных в окне просмотра

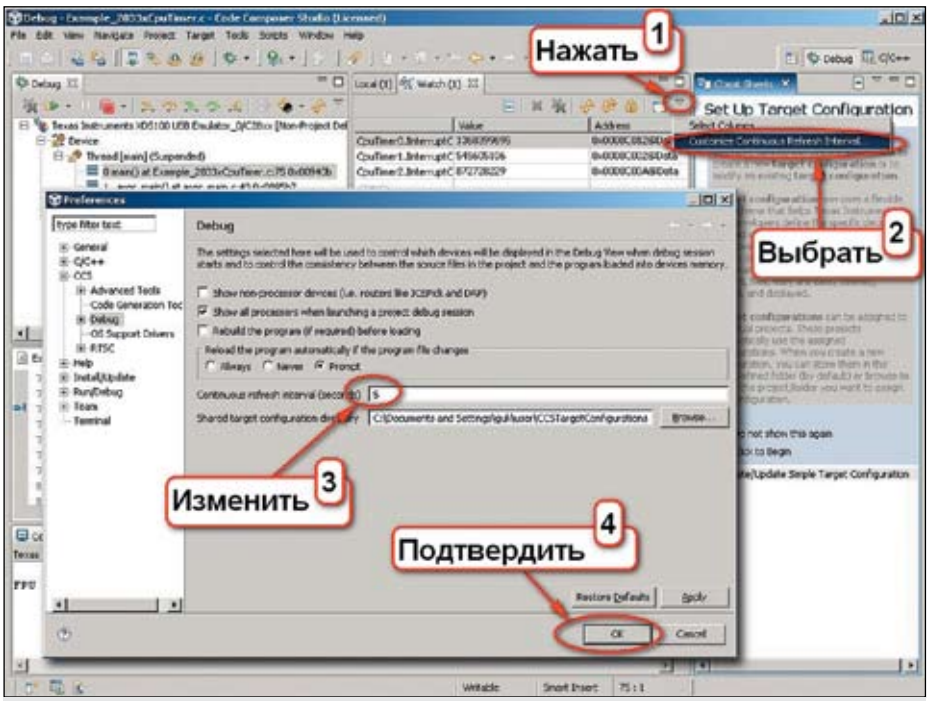

Рис. 22. Изменение интервала мониторинга переменных

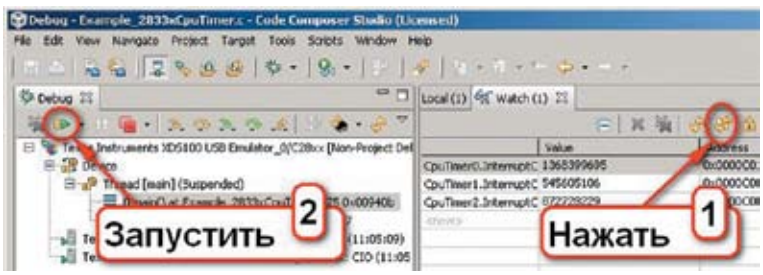

Рис. 23. Запуск проекта в режиме отладки в непрерывном времени

Во-вторых, в верхнем правом углу окна просмотра (***Watch Window***) нажать белую стрелку и выбрать пункт ***Customize Continuous Refresh Interval...***. В появившемся окне изменить значение интервала обновления на «1» (секунду) вместо «5» (секунд) по умолчанию. Процесс показан на рис. 22.

В-третьих, для того чтобы разрешить режим непрерывного обновления, в верхнем правом углу окна просмотра переменных необходимо нажать на желтую иконку с изображением вращающихся вокруг знака паузы стрелок (рис. 23, сноска 1): наблюдаемые переменные будут постоянно обновляться в реальном времени до тех пор, пока выполняется программа.

И последнее: запуск программы. Для этого выберите иконку в виде зеленого треугольника на панели быстрых кнопок (рис. 23, сноска 2), или в главном меню пункт ***Target->Run***, как это уже было показано на рис. 20, сноска 1.

Наблюдаемые переменные в окне просмотра будут инкрементироваться один раз в секунду в соответствии с прерываниями таймера ЦПУ (рис. 24).

На этом краткий обзор новой отладочной среды компании Texas Instruments закончен.

Литература

1. www.ti.com
2. http://focus.ti.com/docs/toolsw/folders/print/tmdsdock28335.html
3. http://focus.ti.com/docs/toolsw/folders/print/sprc530.html
4. http://focus.ti.com/dsp/docs/dspsupportatn.tsp?familyId=44&sectionId=3&tabId=415&toolTypeId=30
5. http://tiexpressdsp.com/index.php/C2000_Getting_Started_with_Code_Composer_Studio_v4