Интегрированная среда разработки и отладки программ MC Studio. Пользовательский интерфейс

Руководство оператора

Листов 85

Порядок использования данного документа

Настоящая документация охраняется действующим законодательством Российской Федерации об авторском праве и смежных правах, в частности, законом Российской Федерации «Об авторском праве и смежных правах». ГУП НПЦ «ЭЛВИС» является единственным правообладателем исключительных авторских прав на настоящую документацию.

Настоящую документацию, не иначе как по предварительному согласию ГУП НПЦ «ЭЛВИС», запрещается:

- воспроизводить, т.е. изготавливать один или более экземпляров настоящей
- документации, ее части, в любой форме, любым способом;
- сдавать в прокат;
- публично показывать, исполнять или сообщать для всеобщего сведения,
- переводить;
- переделывать или другим образом перерабатывать (дорабатывать).

ГУП НПЦ «ЭЛВИС» оставляет за собой право в любой момент вносить изменения (дополнения) в настоящую документацию без предварительного уведомления о таком изменении (дополнении).

ГУП НПЦ «ЭЛВИС» не несет ответственности за вред, причиненный при использовании настоящей документации.

Передача настоящей документации не означает передачи каких-либо авторских прав ГУП НПЦ «ЭЛВИС» на нее.

Возникновение каких-либо прав на материальный носитель, на котором передается настоящая документация, не влечет передачи каких-либо авторских прав на данную документацию.

Все указанные в настоящей документации товарные знаки принадлежат их владельцам.

Оглавление

1. О системе

Система **MultiCore Studio** (MCS) представляет собой интегрированную среду разработки и отладки программного обеспечения для изделий, построенных на базе ИМС **MultiCore**.

Система функционирует на инструментальном компьютере типа IBM-PC под операционной системой **Windows 98/XP**.

2. Назначение и условия применения

2.1. Назначение

Среда разработки программ **MultiCore Studio** (MCS) предназначена для разработки и отладки программ для устройств, построенных на базе ИМС MultiCore.

2.2. Функции

Среда разработки обеспечивает

- создание проекта программы;
- ввод и редактирование текстов программы;
- компиляцию файлов и компоновку программы;
- диагностику и визуальную локализацию синтаксических ошибок;
- подготовку образа памяти для загрузки в целевое устройство.

Среда отладки обеспечивает:

- загрузку программ в модель памяти симулятора MultiCore;
- задание точек останова программы по адресу в программе или на строке программы;
- запуск программы;
- исполнение программы до точки останова или по шагам, с заходом в вызываемую функцию или с пропуском вызываемых функций;
- получение сообщений об остановах и завершении программ;
- чтение данных из памяти симулятора по адресу или символическому имени переменной при остановах программы;
- чтение данных и запись данных с регистров симулятора MultiCore и запись данных в память и регистры.

2.3. Требования к инструментальной машине

MCS является кросс системой и функционирует на инструментальном компьютере типа IBM-PC под операционными системами **Windows 98/XP**.

3. Пользовательский интерфейс системы

3.1. Состав пользовательского интерфейса

Пользовательский интерфейс системы включает:

- главное окно системы;
- дочерние окна;
- плавающие окна;
- диалоги.

Пользователь взаимодействует с системой посредством меню и инструментальных кнопок главного окна, всплывающих меню, дочерних и плавающих окон и кнопок диалогов.

Результаты отображаются на рабочем поле дочерних и плавающих окон, на панелях диалогов и в строке состояния главного окна и дочерних окон.

Дочерние окна системы располагаются внутри рабочей области главного окна и служат:

- для редактирования текста программы (окно редактора);
- отображения текста дизассемблированного кода программы (окно дизассемблера).

Положение дочерних окон внутри главного окна устанавливается с помощью команд меню «Window». Это меню содержит также список открытых дочерних окон.

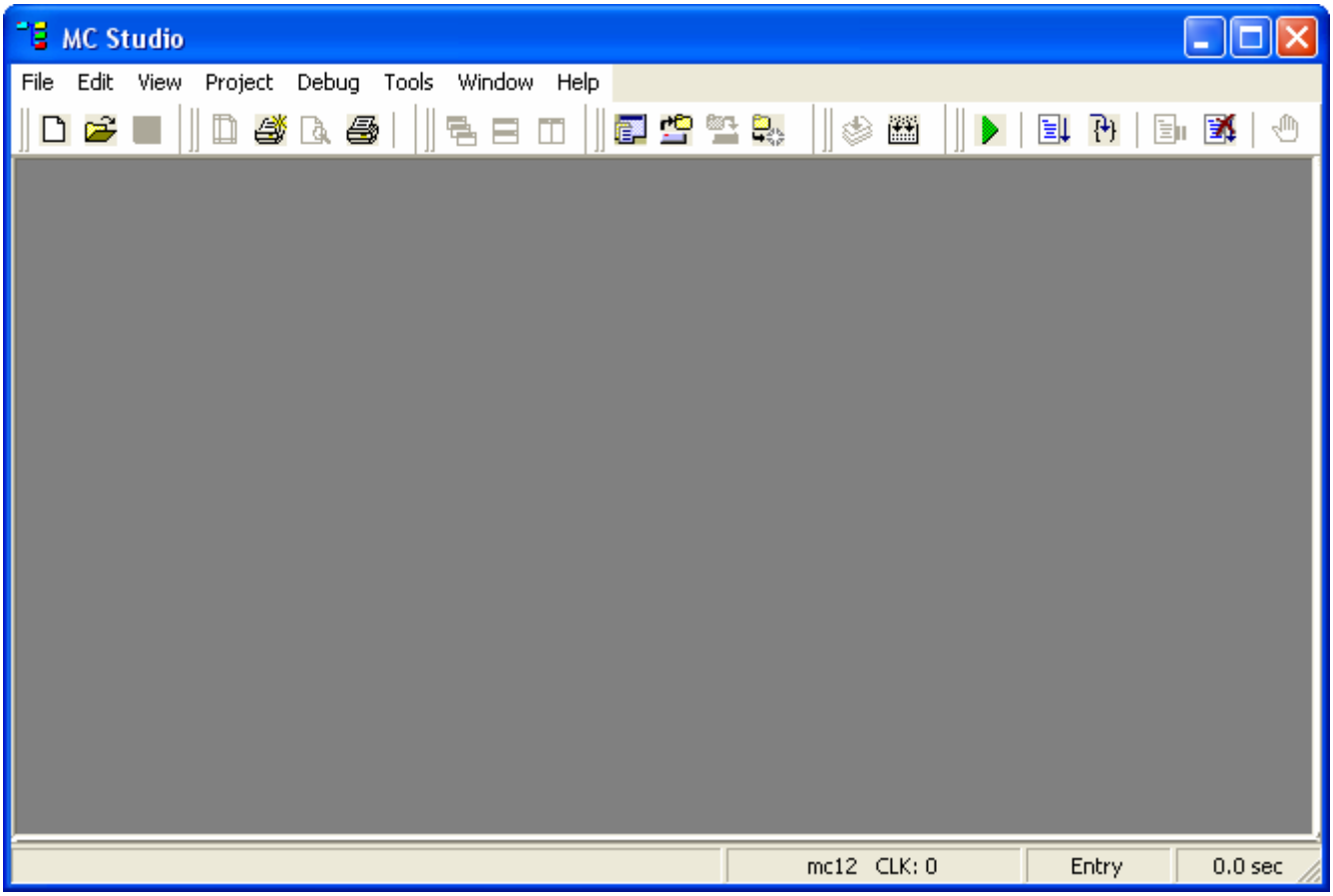
Плавающие окна:

- окно проекта (*Project*);
- окно сообщений (*Message Window*);
- окно поиска (*Search Window*);
- окно точек наблюдения (*Watches*);
- окно локальных переменных (*Locals*);
- окно стека вызовов (*Call Stack*);
- окно профилирования (*Profiler*);
- окно системных регистров (*System*);
- окно регистров RISC-ядра (*RISC*);
- окно регистров DSP-ядра (*DSP*);
- окно памяти (*Memory*);
- окно кэша программ (*Cache*);
- окно TLB (*TLB*);
- окно видеотерминала (*Video*).

Плавающие окна располагаются либо свободно поверх всех окон в любом месте экрана, либо около одного из причалов главного окна.

3.2. Главное окно

При запуске MCS открывается **главное окно** интегрированной среды с заголовком *MC Studio*.



В интегрированной среде все предусмотрено для разработки и отладки программ для устройств, построенных на базе ИМС MultiCore.

В заголовке окна размещается название системы, системное меню и системные кнопки. Это стандартные элементы Windows.

В верхней части окна интегрированной среды расположена полоса главного меню системы.

Ниже главного меню системы расположена полоса инструментальных панелей. На каждой панели расположена группа кнопок, соответствующих командам меню.

В основном поле окна при выполнении соответствующих операций появляются окна редактора, окно проекта, окно сообщений и окна отладчика.

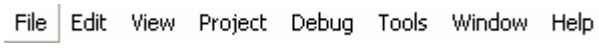
Строка состояния

Строка состояния располагается в нижней части **главного окна**. В строке состояния подсвечивается выполняемое действие при выборе пунктов меню, а также количество отсчетов таймера **CLK**, состояние программы и время работы для режима отладки.

3.3. Меню и кнопки

3.3.1. Главное меню

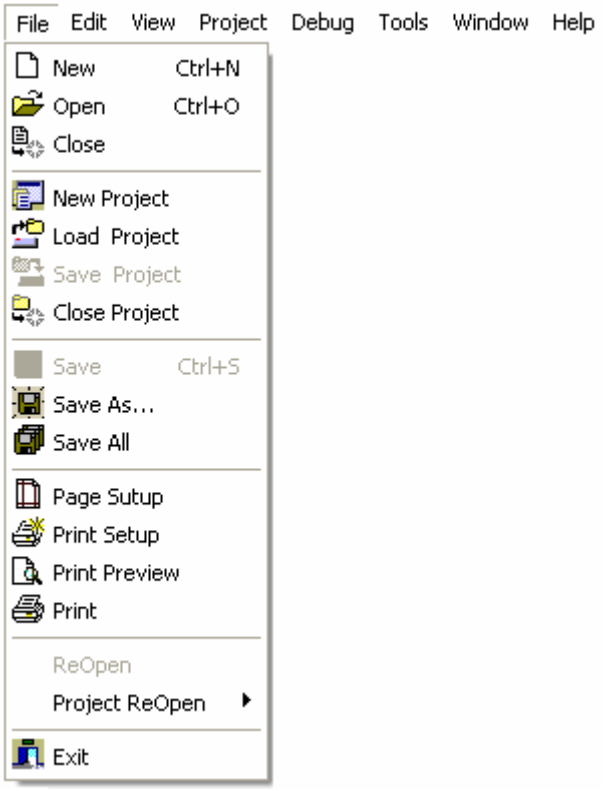
Главное меню системы расположено в верхней части главного окна MCS.



- Раздел меню File позволяет создать новый проект или файл, открыть ранее созданный проект или файл, сохранить проект или файл.
- Раздел меню Edit появляется после открытия файла и позволяет выполнять обычные редакторские действия.
- Раздел меню View позволяет открывать окна сообщений, поиска, отладчика.
- Раздел меню Project позволяет добавлять в проект модули (*units*) RISC – ядра и DSP–ядра и файлы в них, компилировать файлы и компоновать проект, задавать параметры компоновки проекта.
- Раздел меню Debug дает возможность выполнять проект в отладочном режиме в выбранной среде исполнения.
- Раздел меню Tools позволяет осуществлять настройку инструментария для конкретного проекта.
- Раздел меню Window предоставляет возможность управлять размещением окон редактора в главном окне.
- Меню Help содержит строку вызова справочной информации о MCS, инструментах RISC и DSP.

3.3.2. Меню File

Меню **File** позволяет создать новый проект или файл, открыть ранее созданный проект или файл, сохранить проект или файл.



В меню входят следующие пункты:

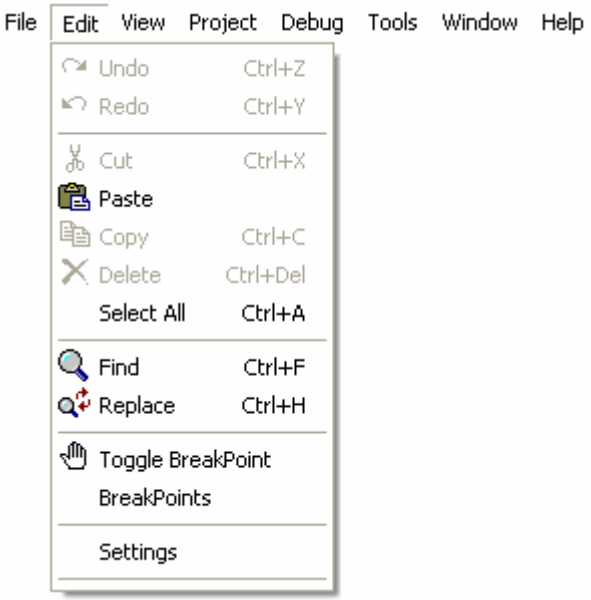
- **New** – создать новый файл. Для нового файла откроется пустое окно редактора.
- **Open** – открыть существующий файл. Выбранный файл загружается в окно редактора кода.
- **Close** – закрыть файл, связанный с активным окном редактора. Если файл был изменен, но не сохранен, появится диалоговое окно с предложением сохранить изменения.
- **New Project** – создать новый проект. При выборе этого пункта меню открывается диалоговое окно создания проекта.
- **Load Project** – открыть существующий проект. При выборе этого пункта меню открывается диалоговое окно открытия проекта.
- **Save Project** – сохранить текущий проект.
- **Close Project** – закрыть текущий проект. Если проект был изменен, система предложит сохранить проект перед закрытием.
- **Save** – сохранить текущий файл, с которым в данный момент выполнялась работа в окне редактора.

- **Save As** – сохранить текущий файл, с которым в данный момент выполнялась работа в окне редактора, под новым именем.
- **Save All** – сохранить все открытые файлы.
- **Page Setup**, **Print Setup**, **Print Preview**, **Print** – изменить параметры страницы печати, настройки принтера, просмотреть изображение печатаемого текста и напечатать текст из текущего окна редактора.
- **ReOpen** – перезагрузить файл, ранее открытый при помощи пунка меню **Open**.
- **Project Reopen** – перезагрузить открытый проект.
- **Exit** – выйти из MCS. Если проект был изменен, но не сохранен, то будет задан вопрос «Сохранить проект, или нет?».

3.3.3. Меню Edit

Меню **Edit** появляется после открытия файла и позволяет выполнять стандартные процедуры редактирования.

Все действия относятся к *активному* окну редактора.

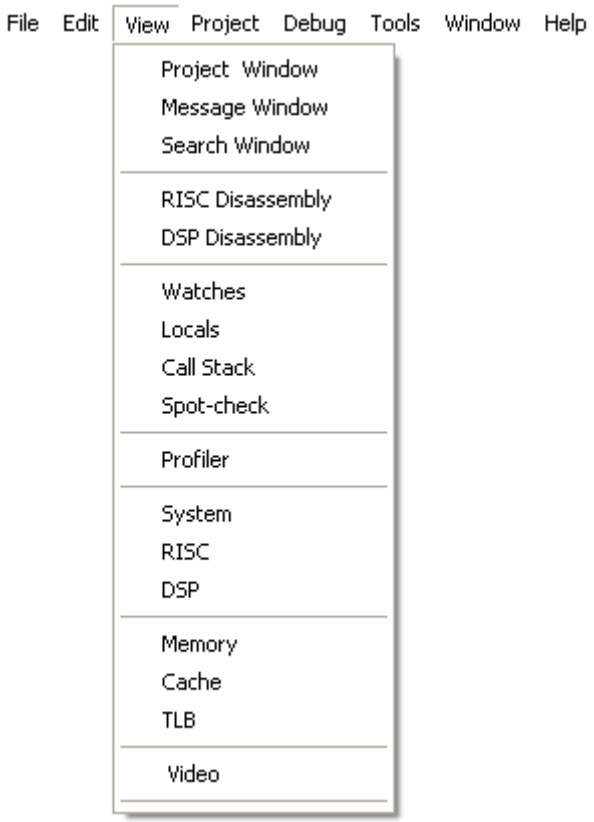


- **Undo** – отменить предыдущее действие.
- **Redo** – отменить предыдущую команду **Undo**.
- **Cut** – копировать в буфер обмена выделенный фрагмент текста. Выделенный фрагмент удаляется из текста.
- **Paste** – копировать содержимое буфера обмена в текст.
- **Copy** – копировать в буфер обмена выделенный фрагмент текста. Выделенный фрагмент не удаляется из текста.

- **Delete** – удалить выделенный текст. Ошибочное действие может быть отменено с помощью команды **Undo**.
- **Select All** – выделить весь текст.
- **Find** – открыть диалоговое окно поиска текста.
- **Replace** – открыть диалоговое окно замены текста.
- **Toggle Break Point** – установить/удалить точку останова в соответствии с положением курсора в окне редактора.
- **BreakPoints** – открыть диалоговое окно редактирования точек останова.
- **Settings** – открыть диалоговое окно настройки редактора кода .

3.3.4. Меню View

Меню **View** позволяет управлять отображением на экране окон для наблюдения в процессе отладки.



Пункты меню **View**:

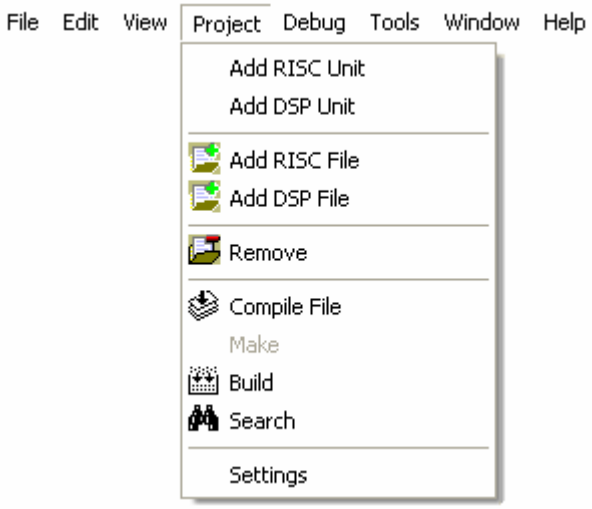
- **Project Window** – окно проекта.
- **Message Window** – окно сообщений.
- **Search Window** – окно поиска.
- **RISC Disassembly** – окно RISC-дизассемблера.
- **DSP Disassembly** – окно DSP-дизассемблера.

- **Watches** – окно точек наблюдения.
- **Locals** – окно локальных переменных.
- **Call Stack** – окно стека вызовов.
- **Spot-check** – окно выборочного просмотра.
- **Profiler** – окно профилирования.
- **System** – окно системных регистров.
- **RISC** – окно регистров RISC-ядра.
- **DSP** – окно регистров DSP-ядра.
- **Memory** – окно памяти.
- **Cache** – окно кэша программ.
- **TLB** – окно TLB.
- **Video** – окно видеотерминала.

Окна, открытые в рабочей области MCS, отмечены в меню галочкой (), кроме окон проекта, сообщений, поиска и дизассемблеров.

3.3.5. Меню Project

Меню **Project** предназначено для построения и сборки проекта. Проект строится из модулей (*units*), которые состоят из файлов (*files*).



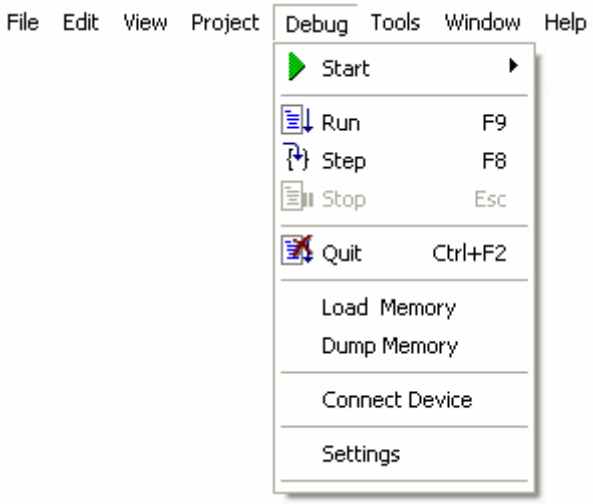
Пункты меню **Project**:

- **Add RISC Unit** – добавить RISC-модуль.
- **Add DSP Unit** – добавить DSP-модуль.
-  **Add RISC File** – добавить RISC-файл (с расширением *.s,.c,* или *.h*) к выбранному RISC-модулю проекта.
-  **Add DSP File** – добавить DSP-файл (с расширением *.s*) к выбранному DSP-модулю проекта.

-  **Remove** – удалить выбранный в окне проекта элемент (файл, модуль с входящими в него файлами, весь проект).
-  **Compile File** – компилировать файл, открытый в **активном** окне редактора кода. Компиляция производится с учетом типа файла и расширения файла. Время, затраченное на компиляцию, отображается в диалоге процесса компиляции файла. Если компиляция прошла успешно, создается обьектный файл с именем компилируемого файла. Сообщения о выполнении компиляции появляются в окне сообщений.
-  **Build** – выполнить сборку проекта. Время, затраченное на сборку, отображается в диалоге процесса сборки проекта. Если компиляция и компоновка прошли успешно, создается обьектный файл проекта и образ памяти для загрузки в симулятор. Сообщения о выполнении операций в процессе компиляции и компоновки проекта появляются в окне сообщений.
-  **Search** – искать текст в файлах проекта. Время, затраченное на поиск, отображается в диалоге процесса поиска. Результаты поиска отображаются в окне поиска.
- **Settings** – открыть диалоговое окно настройки проекта.

3.3.6. Меню Debug

Меню **Debug** предназначено для отладки проекта.



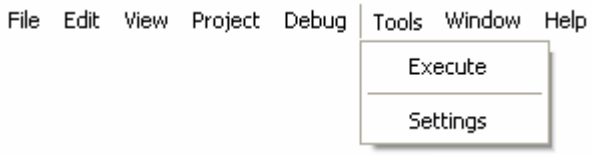
Пункты меню **Debug**:

-  **Start** – начать отладку проекта. Отладка может проходить в одном из двух режимов - в режиме программной (**Simulator**) или в режиме аппаратной (**Emulator**) симуляции. После начала отладки выполняется загрузка образа памяти открытого проекта в память процессора и пункты меню **Debug** становятся доступными.

- **Run** – запустить программу на исполнение до следующей точки останова (BreakPoint) или до выполнения пункта **Stop** меню **Debug**.
- **Step** – запустить программу на исполнение следующей строки кода.
- **Stop** – остановить исполнение программы. В **активном** окне редактора с текстом программы или дизассемблера на исполняемой в момент останова строке будет отображаться символ .
- **Quit** – выйти из режима отладки.
- **Load Memory** – открыть диалог загрузки образа памяти из двоичного файла.
- **Dump Memory** – открыть диалог записи образа памяти в двоичный файл.
- **Connect Device** – открыть диалог подключения внешнего устройства.
- **Settings** – открыть диалог настройки отладчика.

3.3.7. Меню Tools

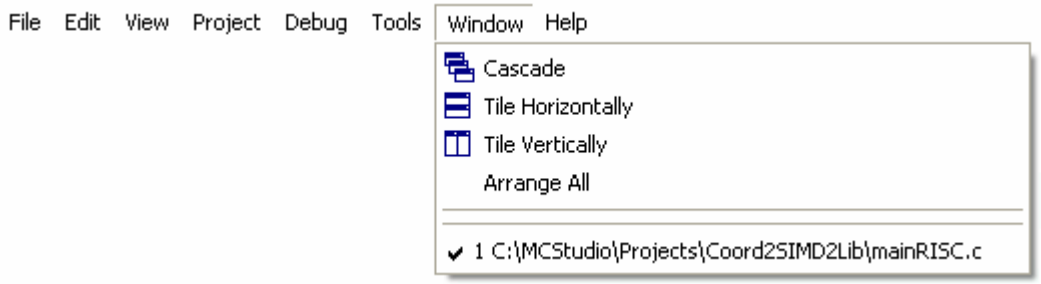
Меню **Tools** предназначено для настройки параметров сборки проекта.



- **Execute** – открыть диалог исполнения командного файла.
- **Settings** – открыть диалог настройки набора инструментов.

3.3.8. Меню Window

Меню **Window** позволяет изменить расположение окон редактора кода в рабочей части главного окна MCS.



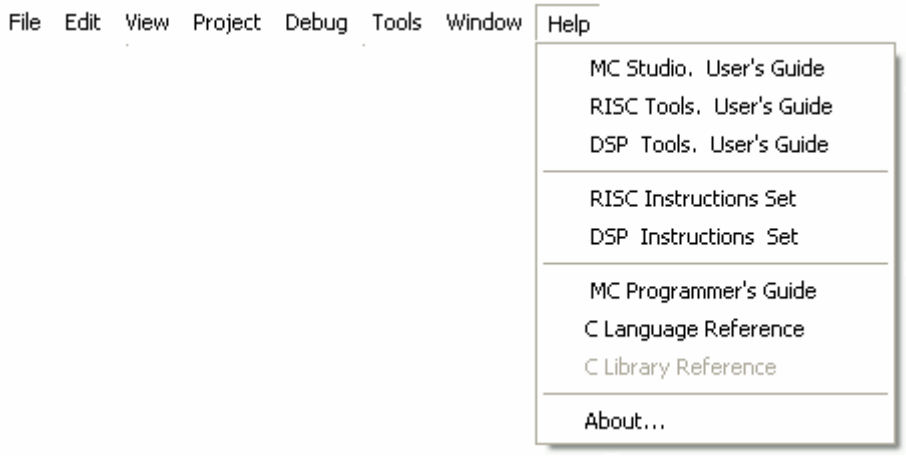
- **Cascade** – расположить окна редактора кода каскадом.
- **Tile Horisontally** – расположить окна редактора кода горизонтально.
- **Tile Vertically** – расположить окна редактора кода вертикально.

- **Arrange All** – упорядочить расположение окон открытых файлов.

После разделительной полосы перечислены все открытые в окнах редактора кода файлы. Галочкой ✔ обозначено активное окно.

3.3.9. Меню Help

Меню **Help** позволяет открыть файлы справки MC Studio.



- **MC Studio** – файл справки по работе с MC Studio.
- **RISC Tools** – файл справки по инструментам RISC.
- **DSP Tools** – файл справки по инструментам DSP.
- **RISC Instructions** – файл справки по инструкциям RISC-ядра.
- **DSP Instructions** – файл справки по инструкциям DSP-ядра.
- **MC Programmer's Giude** – файл справки по программированию в среде MC Studio;
- **C Language Reference** – файл справки по языку программирования *Cu*;
- **C Library Reference** – файл справки по библиотеке *LIBELCORE*;
- **About** – открывает окно "О системе".

3.3.10. Инструментальные кнопки и «горячие» клавиши

Под полосой главного меню расположена **инструментальная панель**, содержащая быстрые кнопки, дублирующие наиболее часто используемые команды меню.

-  – команды меню File: **New**, **Open**, **Save**.
-  – команды меню File: **Page Setup**, **Print Setup**, **Print Preview**, **Print**.
-  – команды меню Window: **Cascade**, **Tile Horizontally**, **Tile Vertically**.
-  – команды меню File: **New Project**, **Load Project**, **Save Project**, **Close Project**.
-  – команды меню Project: **Compile File**, **Build**.
-  – команды меню Debug: **Start**, **Run**, **Step**, **Stop**, **Quit**, а также команда меню Edit **Toggle BreakPoint** (точка останова устанавливается в соответствии с положением курсора в активном окне редактора кода).

«Горячие» клавиши:

- <Ctrl>+<N> – команда **File -- New**.
- <Ctrl>+<O> – команда **File -- Open**.
- <Ctrl>+<S> – команда **File -- Save**.

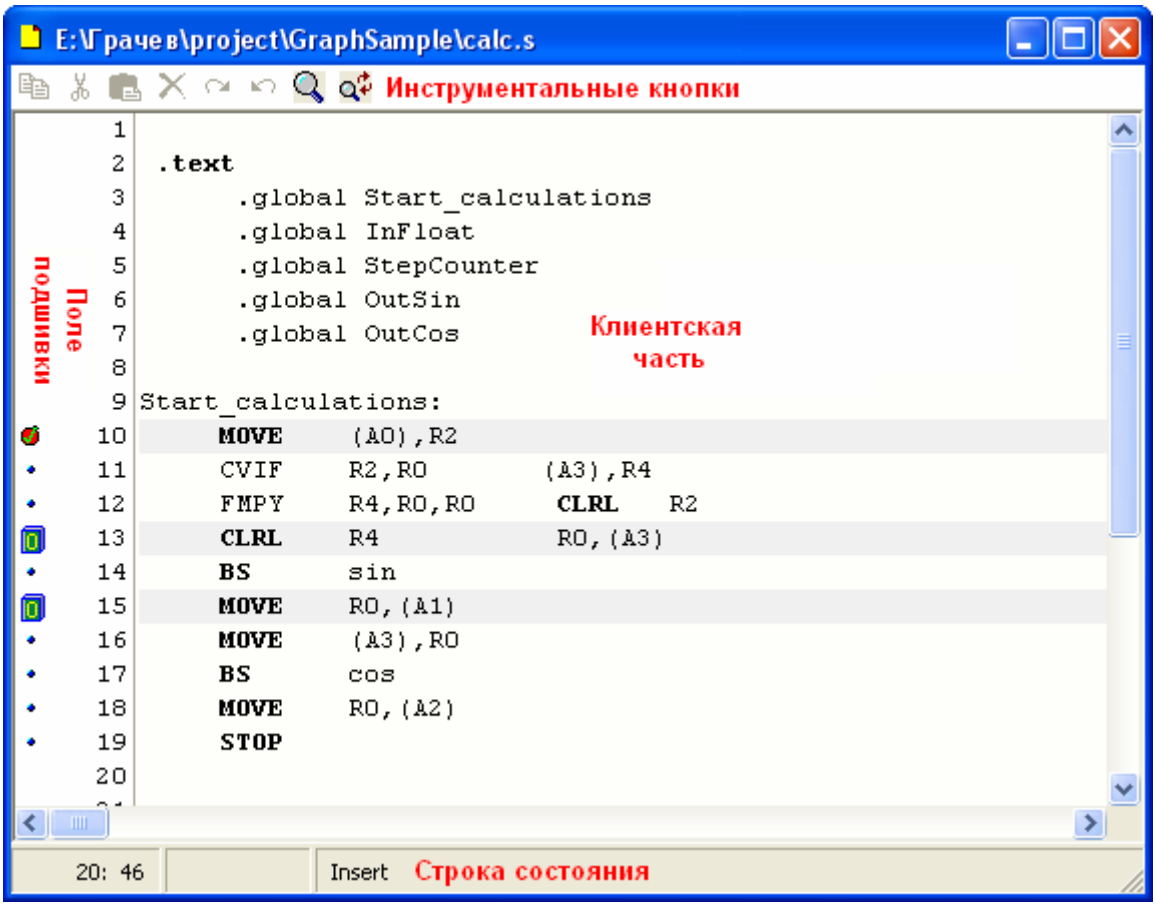
- <Ctrl>+<Z> – команда **Edit -- Undo**.
- <Ctrl>+<Y> – команда **Edit -- Redo**.
- <Ctrl>+<X> – команда **Edit -- Cut**.
- <Ctrl>+<C> – команда **Edit -- Copy**.
- <Ctrl>+<V> – команда **Edit -- Paste**.
- <Ctrl>+<Del> – команда **Edit -- Delete**.
- <Ctrl>+<A> – команда **Edit -- Select All**.
- <Ctrl>+<F> – команда **Edit -- Find**.
- <Ctrl>+<H> – команда **Edit -- Replace**.

- <F9> – команда **Debug -- Run**.
- <F8> – команда Debug **-- Step**.
- <Esc> – команда Debug **-- Stop**.
- <Ctrl>+<F2> – команда Debug **-- Quit**.

3.4. Окна

3.4.1. Окно редактора

Окна редактора – основные рабочие окна MC Studio. В них отображаются все открытые файлы с кодом и производятся изменения над ними. При создании(загрузке) файла с кодом ассемблера или C, создается новое **окно редактора**.



Инструментальные кнопки

- – команда Edit -- **Copy**.
- – команда Edit -- **Cut**.
- – команда Edit -- **Paste**.
- – команда Edit -- **Delete**.
- – команда Edit -- **Undo**.
- – команда Edit -- **Redo**.
- – команда Edit -- **Find**.
- – команда Edit -- **Replace**.

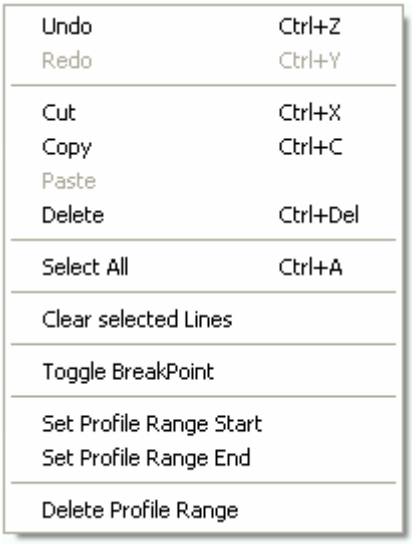
Все перечисленные команды равносильны пунктам меню Edit, примененным к активному **окну редактора кода**.

Клиентская часть окна редактора кода

В клиентской части **окна редактора** отображается и изменяется открытый файл. Слева от клиентской части расположено **поле подшивки**. **Поле подшивки** содержит нумерацию строк кода и специальные символы. Символ  указывает на строку кода, которая будет исполнена следующей, а символами  обозначены точки останова (BreakPoints). Символами  обозначены точки, в которых останов не может быть осуществлен. Символами  обозначены начало и конец блока профилирования, причем цифра в центре символа указывает на номер блока.

Символами  обозначены строки кода, содержащие хотя бы одну инструкцию и в которых может быть поставлена точка останова (BreakPoint).

При щелчке правой кнопкой мыши на **окне редактора кода**, появится выпадающее меню:



Пункты выпадающего меню:

- **Undo** – команда **Edit -- Undo**.
- **Redo** – команда **Edit -- Redo**.
- **Cut** – команда **Edit -- Cut**.
- **Copy** – команда **Edit -- Copy**.
- **Paste** – команда **Edit -- Paste**.
- **Delete** – команда **Edit -- Delete**.
- **Select All** – команда **Edit -- Select All**.
- **Clear selected Lines** – очистить все подсвеченные линии **клиентской части** окна редактора кода.
- **Toggle BreakPoint** – установить/удалить точку останова.
- **Set Profile Range Start** – установить начало блока профилирования.
- **Set Profile Range End** – установить конец блока профилирования.
- **Delete Profile Range** – удалить блок профилирования.

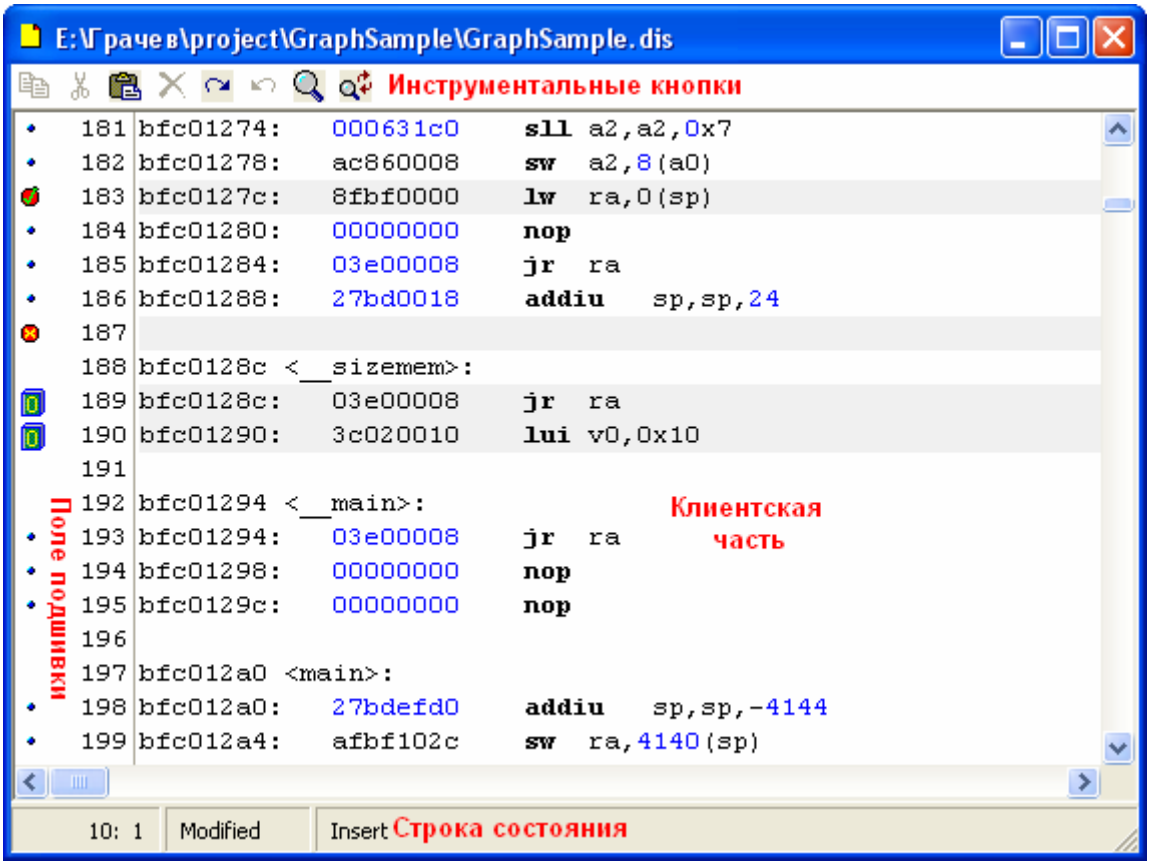
Строка состояния

В строке состояния **окна редактора** отображается положение курсора (столбец:строка), состояние файла (изменен/не изменен) и режим набора текста (*Insert*/*Overwrite*).

Пользователь может настроить внешний вид окна редактора кода по своему усмотрению.

3.4.2. Окно дизассемблера

В **окне дизассемблера** отображается дизассемблированный код для RISC или DSP. Окно открывается выбором пункта **RISC Disassembly** или **DSP Disassembly** в меню View.



Инструментальные кнопки

-  – команда **Edit -- Copy**.
-  – команда **Edit -- Cut**.
-  – команда **Edit -- Paste**.
-  – команда **Edit -- Delete**.
-  – команда **Edit -- Undo**.
-  – команда **Edit -- Redo**.
-  – команда **Edit -- Find**.
-  – команда **Edit -- Replace**.

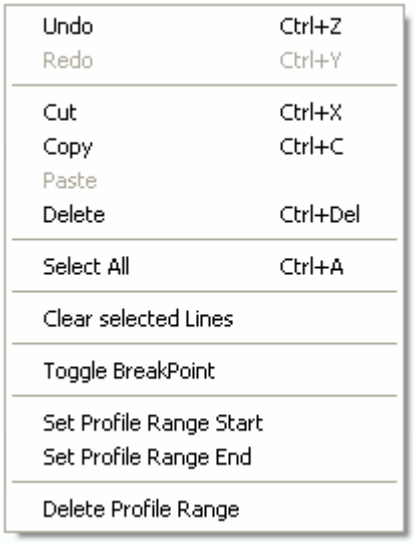
Все перечисленные команды равносильны пунктам меню Edit, примененным к активному **окну дизассемблера**.

Клиентская часть окна дизассемблера

В клиентской части **окна дизассемблера** отображается и изменяется файл с дизассемблированным кодом. Слева от клиентской части расположена панель нумерации строк кода. Символ  указывает на строку кода, которая будет исполнена следующей, а символами  обозначены точки останова (BreakPoints). Символами  обозначены точки, в которых останов не может быть осуществлен. Символами  обозначены начало и конец блока профилирования, причем цифра в центре символа указывает на номер блока.

Символами  обозначены строки кода, содержащие хотя бы одну инструкцию и в которых может быть поставлена точка останова (BreakPoint).

При щелчке правой кнопкой мыши на **окне дизассемблера**, появится выпадающее меню:



Пункты выпадающего меню:

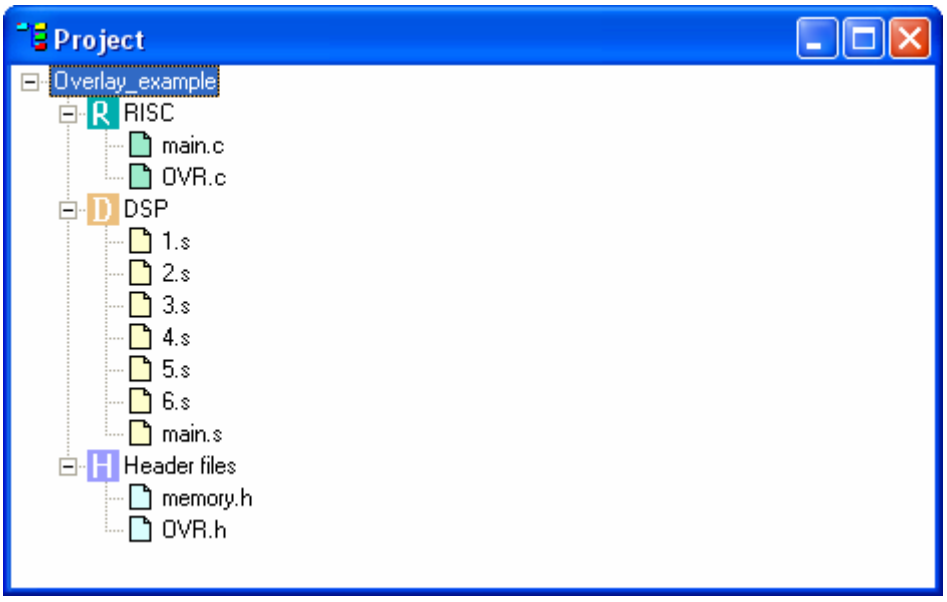
- **Undo** – команда **Edit -- Undo**.
- **Redo** – команда **Edit -- Redo**.
- **Cut** – команда **Edit -- Cut**.
- **Copy** – команда **Edit -- Copy**.
- **Paste** – команда **Edit -- Paste**.
- **Delete** – команда **Edit -- Delete**.
- **Select All** – команда **Edit -- Select All**.
- **Clear selected Lines** – очистить все подсвеченные линии **клиентской части** окна дизассемблера.
- **Toggle BreakPoint** – установить/удалить точку останова.
- **Set Profile Range Start** – установить начало блока профилирования.
- **Set Profile Range End** – установить конец блока профилирования.
- **Delete Profile Range** – удалить блок профилирования.

Строка состояния

В строке состояния **окна дизассемблера** отображается положение курсора (столбец:строка), состояние файла (изменен/не изменен) и режим набора текста (Insert/Overwrite).

3.4.3. Окно проекта

Окно проекта выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно открывается при загрузке проекта, при создании нового проекта, а также при выборе пункта **Project Window** меню View.



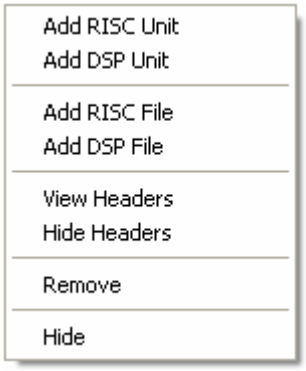
В **окне проекта** отображается структура открытого проекта: его название, модули(*units*), присоединенные к проекту, файлы, присоединенные к модулям. Символами R обозначены модули RISC, символами D - модули DSP, символом H - список файлов заголовков (.*h*).

- файлы модулей RISC, - файлы модулей DSP, - файлы заголовков.

Следует учитывать, что при сборке проекта файлы будут обрабатываться компилятором именно в той последовательности, в которой они приведены в окне проекта. Перемещение файлы в окне проекта осуществляется по принципу *Drag'n'Drop*.

При двойном щелчке на любом файле проекта, окно редактора кода, содержащее этот файл, откроется и станет **активным**.

При щелчке на **окне проекта** правой кнопкой мыши появится выпадающее меню:

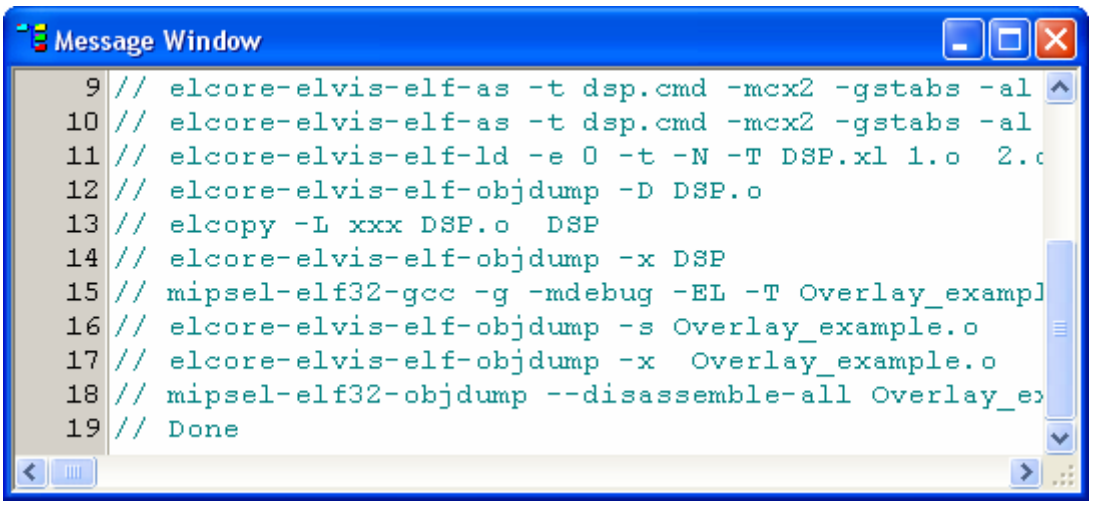


Пункты выпадающего меню:

- **Add RISC Unit** – команда **Project -- Add RISC Unit**.
- **Add DSP Unit** – команда **Project -- Add DSP Unit**.
- **Add RISC File** – команда **Project -- Add RISC File**.
- **Add DSP File** – команда **Project -- Add DSP File**.
- **View Headers** – показать список файлов заголовков.
- **Hide Headers** – скрыть список файлов заголовков.
- **Remove** – команда **Project -- Remove**.
- **Hide** – скрыть **окно проекта**.

3.4.4. Окно сообщений

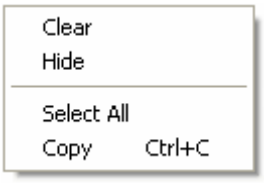
Окно сообщений выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется автоматически при сборке проекта, а также при выборе пункта **Message Window** меню View.



В **окне сообщений** отображаются сообщения, генерируемые в процессе компиляции файла или сборки проекта. Все сообщения пронумерованы. Если в окне появилось сообщение об ошибке, то при щелчке мышью на номер сообщения, будет осуществлен переход к строке, содержащей эту ошибку.

Также в окне сообщений могут быть отображены файлы, генерируемые в процессе сборки. Файлы, содержимое которых следует отобразить в окне сообщений, выбираются в диалоге настройки набора инструментов.

При щелчке на **окне сообщений** правой кнопкой мыши появится выпадающее меню:



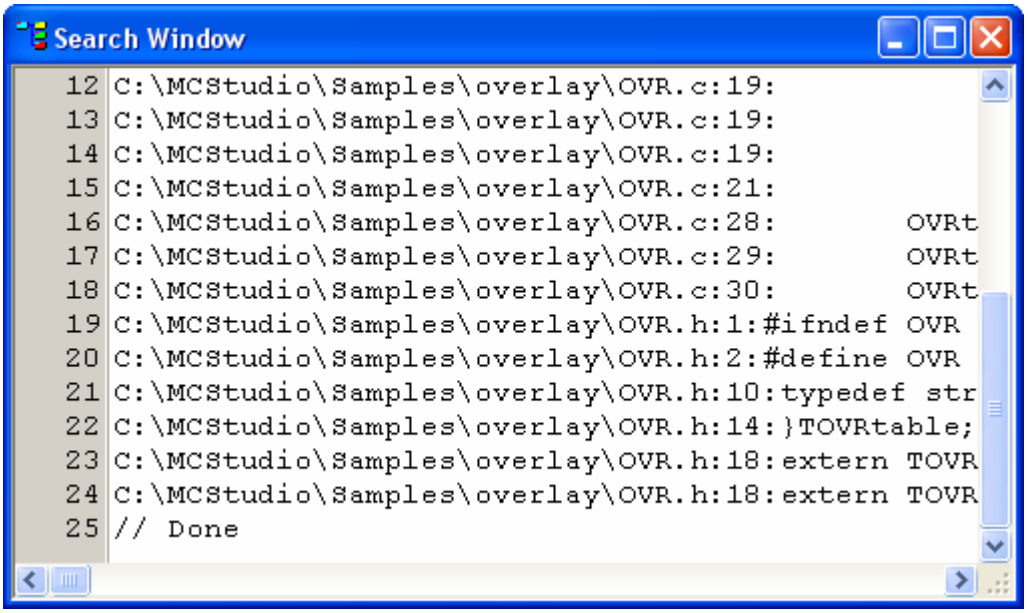
Пункты выпадающего меню:

- **Clear** – очистить **окно сообщений**.
- **Hide** – скрыть **окно сообщений**.
- **Select All** – выделить все содержимое окна.
- **Copy** – скопировать выделенный текст в буфер.

3.4.5. Окно поиска

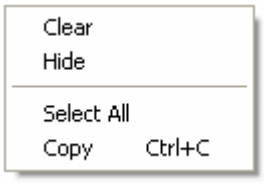
Окно поиска выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя.

Окно появляется автоматически при исполнении поиска по проекту, или при выборе пункта **Search Window** меню View.



В **окне поиска** отображаются результаты последнего поиска текста по всем файлам проекта. Каждая строка окна содержит имя файла, где был найден текст. Все найденные строки пронумерованы. При щелчке на номер, окно редактора кода, содержащее указанную строку, станет активным, а сама строка будет выделена.

При щелчке на **окне поиска** правой кнопкой мыши появится выпадающее меню:

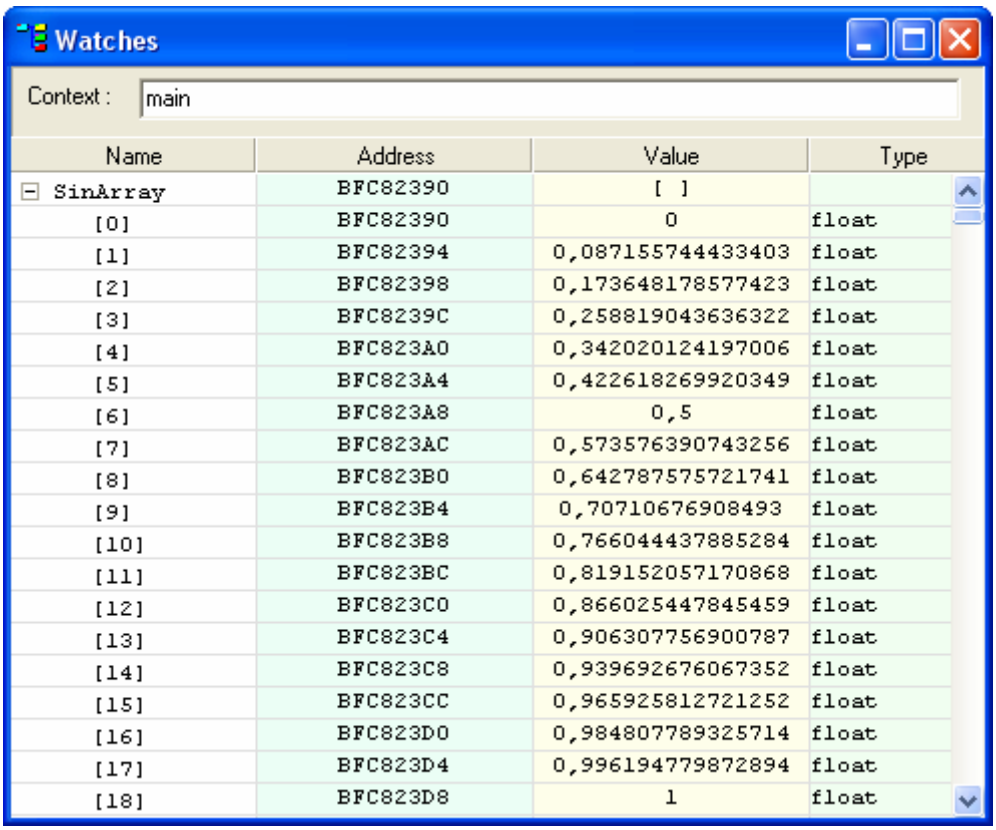


Пункты выпадающего меню:

- **Clear** – очистить **окно поиска**.
- **Hide** – скрыть **окно поиска**.
- **Select All** – выделить все содержимое окна.
- **Copy** – скопировать выделенный текст в буфер.

3.4.6. Окно точек наблюдения

Окно точек наблюдения выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Watches** меню View.



Name	Address	Value	Type
SinArray	BFC82390	[ ]	
[0]	BFC82390	0	float
[1]	BFC82394	0,087155744433403	float
[2]	BFC82398	0,173648178577423	float
[3]	BFC8239C	0,258819043636322	float
[4]	BFC823A0	0,342020124197006	float
[5]	BFC823A4	0,422618269920349	float
[6]	BFC823A8	0,5	float
[7]	BFC823AC	0,573576390743256	float
[8]	BFC823B0	0,642787575721741	float
[9]	BFC823B4	0,70710676908493	float
[10]	BFC823B8	0,766044437885284	float
[11]	BFC823BC	0,819152057170868	float
[12]	BFC823C0	0,866025447845459	float
[13]	BFC823C4	0,906307756900787	float
[14]	BFC823C8	0,939692676067352	float
[15]	BFC823CC	0,965925812721252	float
[16]	BFC823D0	0,984807789325714	float
[17]	BFC823D4	0,996194779872894	float
[18]	BFC823D8	1	float

В **окне точек наблюдения** отображены все точки наблюдения (Watches) за переменными и выражениями в процессе отладки, установленные пользователем.

Окно точек наблюдения содержит таблицу из четырех столбцов. В первом столбце (**Name**) перечислены имена точек наблюдения. В столбце **Address** указан адрес в памяти для каждого выражения. Столбец **Value** содержит значения выражений в шестнадцатеричном или десятичном формате (или знаки "?", если выражение не определено). Столбец **Type** - тип каждого выражения (или знак "?", если тип не определен).

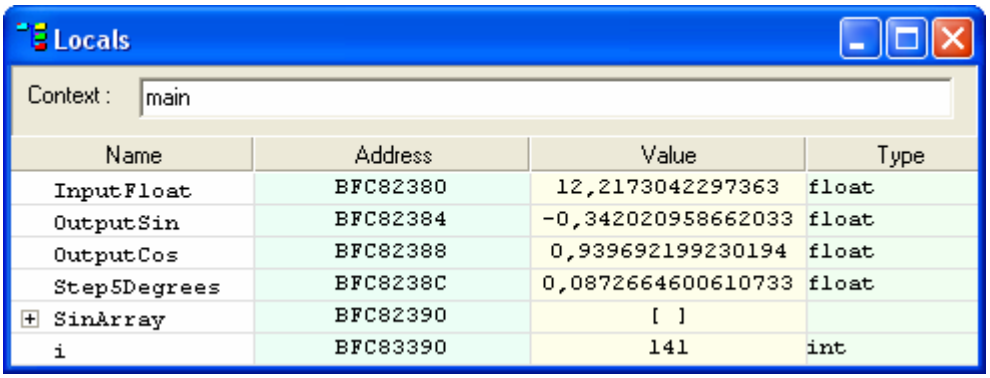
В поле **Context** приводится имя функции, в контексте которой осуществляется наблюдение за выражениями.

При щелчке правой кнопкой мыши на **окне точек наблюдения**, появляется выпадающее меню:

- **Add Watch** – добавить точку наблюдения. При выборе этого пункта меню система предложит ввести выражение для наблюдения. После нажатия кнопки <*OK*> выражение появится в **окне точек наблюдения**.
- **Edit Watch** – редактировать точки наблюдения. При выборе этого пункта появится окно, содержащее список всех точек наблюдения. В окне точек наблюдения будут появляться только выражения, помеченные галочкой (☑). Чтобы удалить точку наблюдения из списка, щелкните на нее мышью и нажмите кнопку <*Remove*>.
- **Hexadecimal Display** – этот пункт меню позволяет пользователю изменить формат столбца **Value** в окне точек наблюдения с десятичного на шестнадцатеричный и обратно.

3.4.7. Окно локальных переменных

Окно локальных переменных выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Locals** меню View.



В **окне локальных переменных** отображаются значения всех локальных переменных той функции, которая исполняется в данный момент. Имя этой функции приводится в поле **Context**.

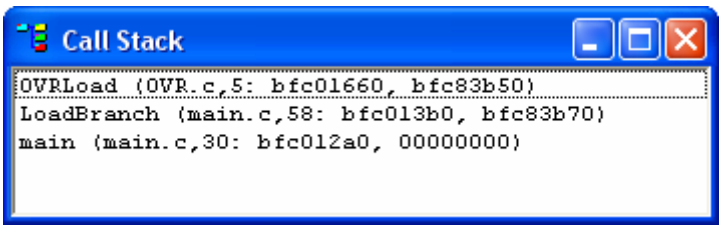
Окно локальных переменных содержит таблицу из четырех столбцов. В первом столбце (**Name**) перечислены имена локальных переменных. В столбце **Address** указан адрес в памяти для каждой переменной. Столбец **Value** содержит значения переменных в шестнадцатеричном или десятичном формате (или знаки "?", если выражение не определено). Столбец **Type** - тип каждой переменной (или знак "?", если тип не определен).

При щелчке правой кнопкой мыши на **окне локальных переменных**, появляется выпадающее меню с пунктом **Hexadecimal Display**. Этот пункт позволяет пользователю изменить формат столбца **Value** окна локальных переменных с десятичного на шестнадцатеричный и обратно.

Содержимое **окна локальных переменных** доступно только в режиме отладки проекта.

3.4.8. Окно стека вызовов

Окно стека вызовов выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Call Stack** меню View.

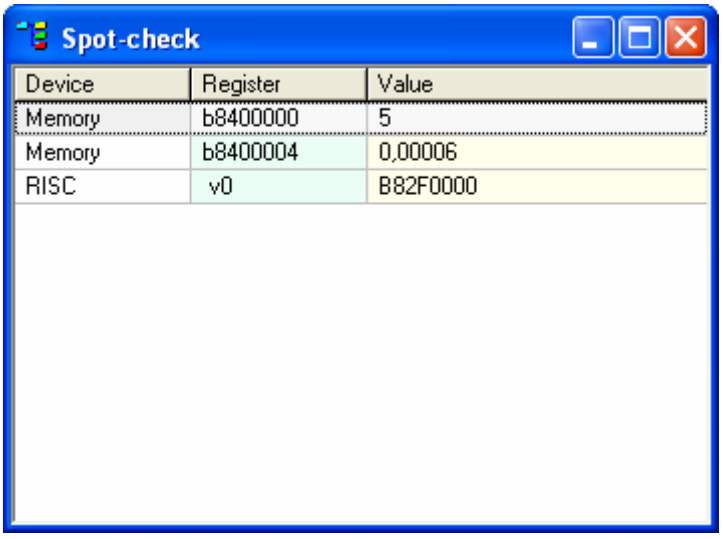


В окне отображено содержимое стека вызовов функций. Для каждой вызванной функции в окне указаны (слева направо):

- имя функции;
- имя файла с определением функции;
- номер строки, с которой начинается определение функции;
- адрес размещения функции в памяти RISC;
- адрес фрейма функции (содержимое регистра **s8**).

3.4.9. Окно выборочного просмотра

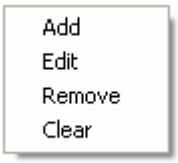
Окно выборочного просмотра выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Spot-check** меню View.



Окно выполнено в виде таблицы и позволяет выборочно просматривать содержимое тех или иных ячеек памяти и регистров *MultiCore* в выбранном формате. Поле **Device** содержит тип устройства, содержимое элемента которого отображается. Поле **Register** содержит адрес отображаемой ячейки памяти или имя регистра. Поле

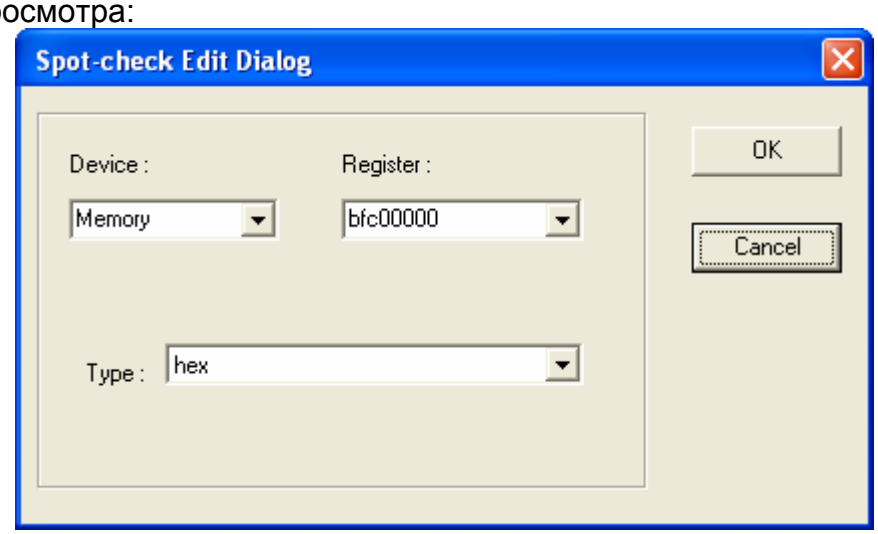
Value отображает значение, содержащееся в регистре/ячейке памяти в выбранном в формате.

При щелчке правой кнопкой мыши в пределах **окна выборочного просмотра** появится выпадающее меню:



Пункт **Add** позволяет добавить в окно выборочного просмотра ячейку памяти или регистр. Пункт **Edit** позволяет изменить настройки просмотра выбранной ячейки памяти/регистра. Пункт **Remove** удаляет выбранную ячейку пямяти/регистр из окна выборочного просмотра. Пункт **Clear** полностью очищает окно выборочного просмотра.

Выбор пунктов **Add** или **Edit** откроет диалог настроек окна выборочного просмотра:

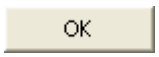


Селектор **Device** позволяет пользователю выбрать тип устройства, содержимое ячейки памяти или регистра которого необходимо просмотреть. Доступные устройства: **Memory** (отображение ячейки памяти *MultiCore*) и **RISC** (отображение регистров RISC-ядра).

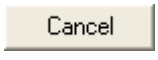
В поле **Register** необходимо идентифицировать просматриваемую ячейку/регистр. Если для просмотра выбрано устройство **Memory**, в поле **Register** следует указывать адрес нужной ячейки памяти (виртуальный, байтовый). Если для просмотра выбрано устройство **RISC**, необходимо выбрать один из регистров RISC-ядра. Для выбора доступны регистры общего назначения RISC-ядра, регистры **HI**, **LO**, **PC**, а также регистры управляющего сопроцессора CP0.

В поле **Type** следует выбрать тип отображаемых данных. Для выбора доступны следующие типы:

- **hex** - отображает содержимое выбранной ячейки памяти/регистра в шестнадцатеричном виде. Отображается слово длиной 4 байта (первый байт - младший);
- **int** - отображает слово длиной 4 байта в десятичном виде, формат - целочисленный знаковый;
- **unsigned int** - отображает слово длиной 4 байта в десятичном виде, формат - целочисленный беззнаковый;
- **float** - отображает содержимое ячейки памяти/регистра в формате числа с плавающей точкой (24E8, IEEE-754);
- **.dw #s** - отображает 2 младших байта в десятичном виде, формат - целочисленный беззнаковый;
- **.dl #S** - отображает слово длиной 4 байта в десятичном виде, формат - целочисленный беззнаковый;
- **.dl #[Re,Im]** - отображает слово длиной 4 байта в десятичном виде как комплексное число, где 2 старших байта - действительная часть, а 2 младших байта - мнимая часть. Формат - целочисленный беззнаковый;
- **.dl #[[@Re1,Re0],[@Im1,Im0]]** - отображает слово длиной 4 байта в десятичном виде как два комплексных числа. 2 старших байта являются действительными частями комплексных чисел, 2 младших байта - мнимыми. Формат - целочисленный беззнаковый;
- **.fr #s** - отображает 2 младших байта в десятичном виде, формат - дробный (*fractional*) знаковый;
- **.frl #S** - отображает слово длиной 4 байта в десятичном виде, формат - дробный (*fractional*) знаковый;
- **.frl #[Re,Im]** - отображает слово длиной 4 байта в десятичном виде как комплексное число, где 2 старших байта - действительная часть, а 2 младших байта - мнимая часть. Формат - дробный (*fractional*) знаковый;
- **.frl #[[@Re1,Re0],[@Im1,Im0]]** - отображает слово длиной 4 байта в десятичном виде как два комплексных числа. 2 старших байта являются действительными частями комплексных чисел, два младших байта - мнимыми. Формат - дробный (*fractional*) знаковый;
- **.real** - отображает содержимое ячейки памяти/регистра в формате числа с плавающей точкой (24E8, IEEE-754).

Нажатие кнопки 

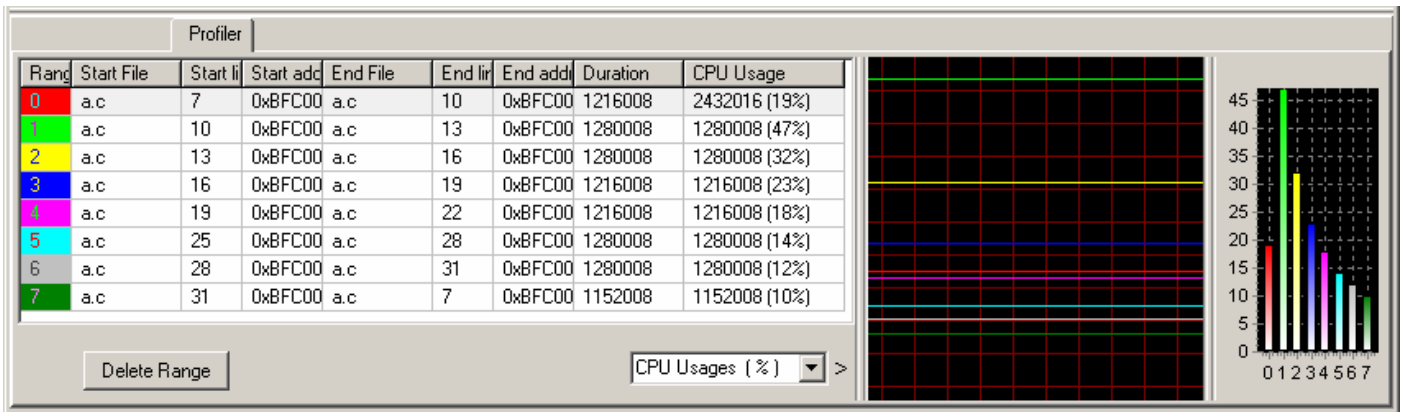
подтверждает добавление нового (изменение текущего) элемента и закрывает диалог.

Кнопка 

закрывает диалог настроек окна выборочного просмотра без подтверждения изменений.

3.4.10. Окно профилирования

Окно профилирования выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Profiler** меню View.



Окно профилирования предназначено для получения параметров при выполнении блоков профилирования программы - таких параметров, как Количество тактов (**Range duration**) и Использование процессора (**CPU Usage**).

Окно содержит таблицу и две панели для отображения графиков изменения параметров в процессе отладки проекта.

Столбец **Range** указывает номер блока профилирования (всего до восьми блоков). Столбцы **Start File** и **End File** содержат имена файлов, в которых соответственно начинается и заканчивается блок. Столбцы **Start line** и **End line** отображают номера начальной и конечной строк блока. Столбцы **Start Address** и **Stop Address** содержат адреса первой и последней инструкции блока в памяти. Столбец **Duration** отображает Количество тактов, за которое выполнен блок, а столбец **CPU Usage** - Использование ЦПУ за время выполнения блока.

Нажатие кнопки 

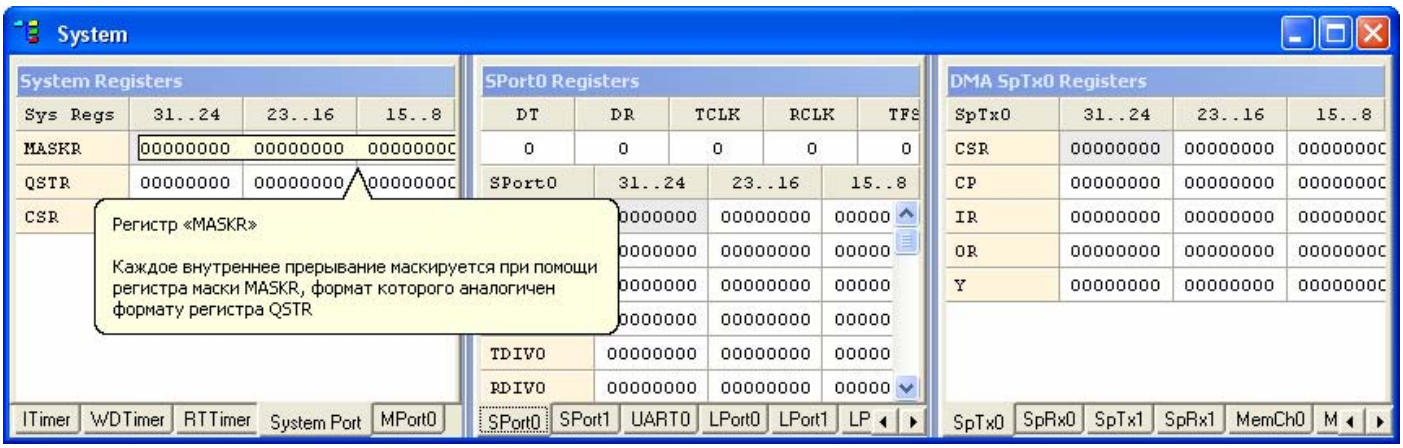
удаляет выбранный блок профилирования.

Селектор параметров 

позволяет выбирать, какой из параметров будет иллюстрироваться графиками в момент исполнения блока профилирования.

3.4.11. Окно системных регистров

Окно системных регистров выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **System** меню View.



Окно содержит три панели.

Первая панель отображает содержимое регистров таймеров, регистров системного порта и регистров порта памяти.

Вторая панель отображает содержимое регистров последовательных портов, регистров параллельных портов и регистров порта UART.

На третьей панели отображено содержимое регистров каналов DMA.

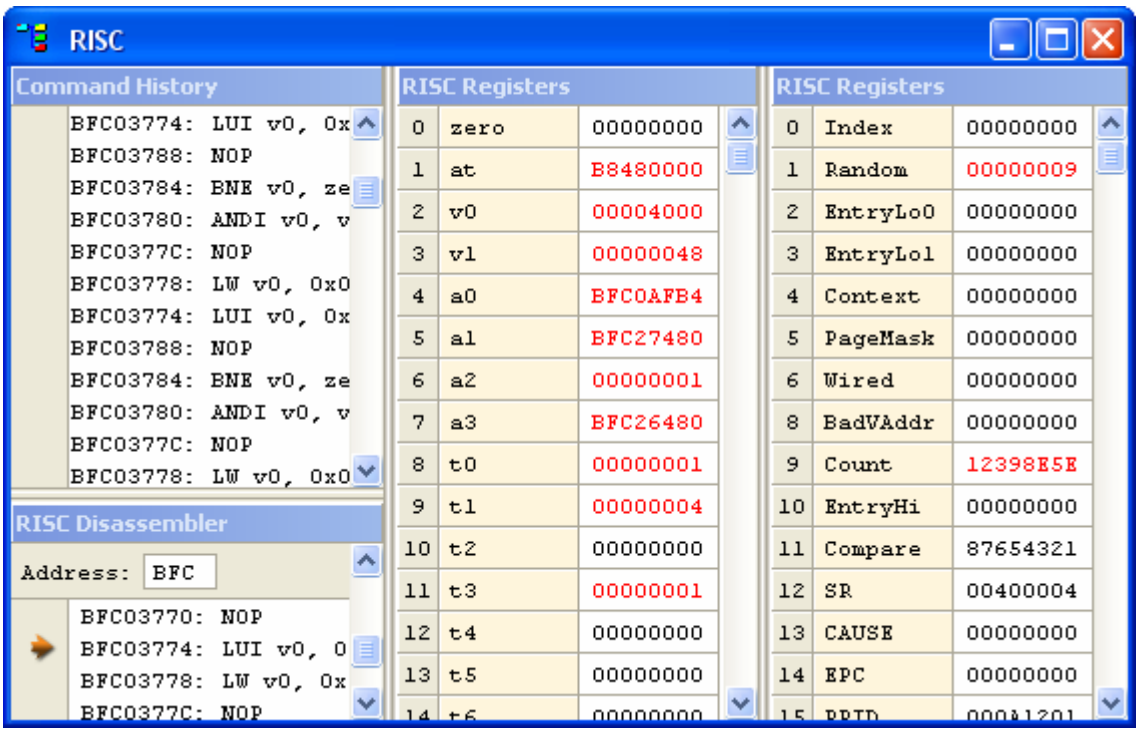
Перемещение между закладками каждой из трех панелей осуществляется кнопками  .

Если задержать курсор мыши на содержимом одного из регистров, появится всплывающее окно с кратким описанием регистра как показано на рисунке.

Содержимое **окна системных регистров** доступно только в режиме отладки проекта.

3.4.12. Окно регистров RISC-ядра

Окно регистров RISC-ядра выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **RISC** меню View.



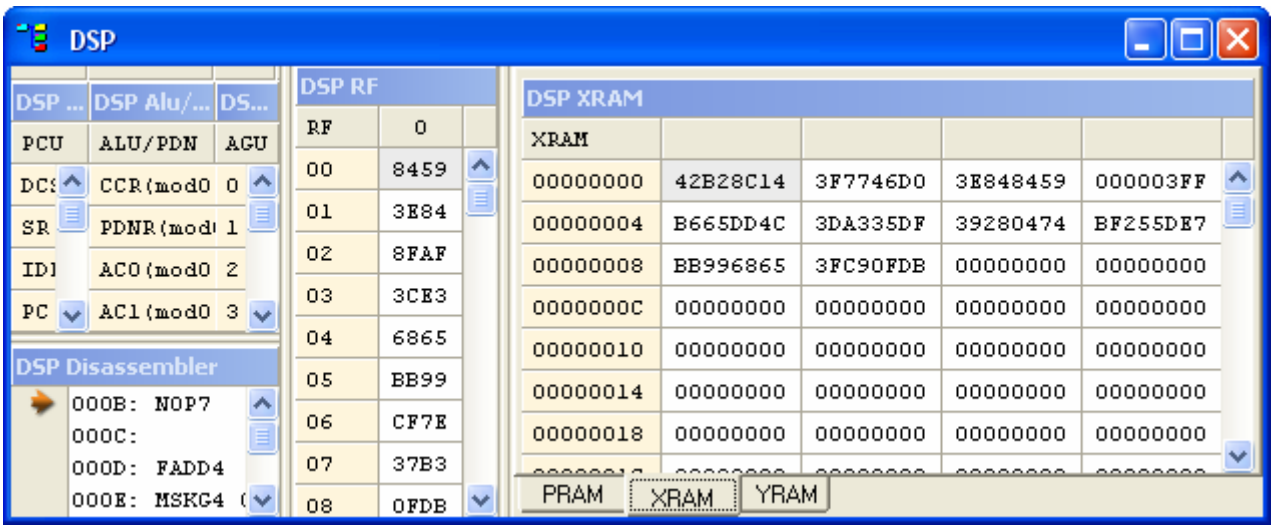
В окне регистров RISC-ядра отображены:

- регистры RISC-ядра (**RISC Registers**);
- история выполненных инструкций (**Command History**);
- дизассемблированный код программы (**RISC Disassembler**) с указателем на текущую инструкцию ( ).

Содержимое окна доступно только в режиме отладки проекта.

3.4.13. Окно регистров DSP-ядра

Окно регистров DSP-ядра выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **DSP** меню View.



В окне **регистров DSP-ядра** отображено:

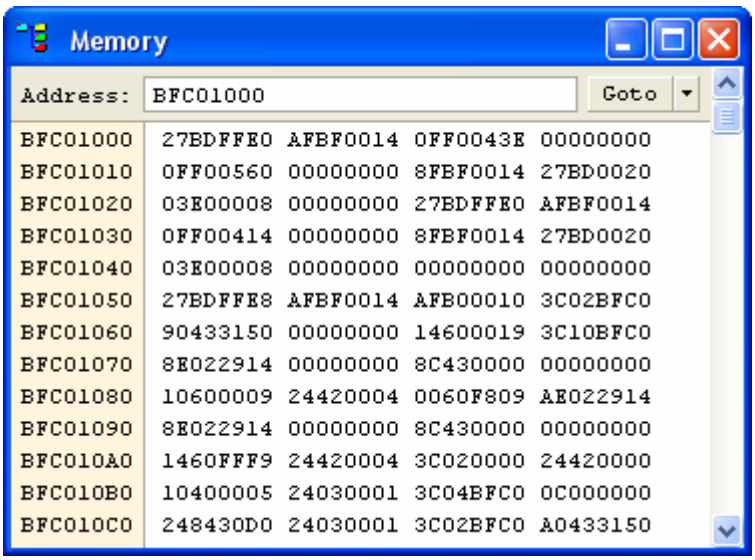
- содержимое регистров **PCU**;
- содержимое регистров **ALU** и **PDN**;
- содержимое адресных регистров **AGU** и **AGU-Y**;
- содержимое регистров регистрового файла **RF**;
- содержимое ячеек памяти (**PRAM**, **XRAM**, **YRAM**);
- дизассемблированный код программы с указателем на текущую инструкцию (➧).

Окно доступно только в режиме отладки проекта.

Примечание: содержимое **окна регистров DSP-ядра** может варьироваться в зависимости от типа DSP-ядра выбранной модели ИМС "*МУЛЬТИКОР*", для которой создается проект.

3.4.14. Окно памяти

Окно памяти выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Memory** меню View.



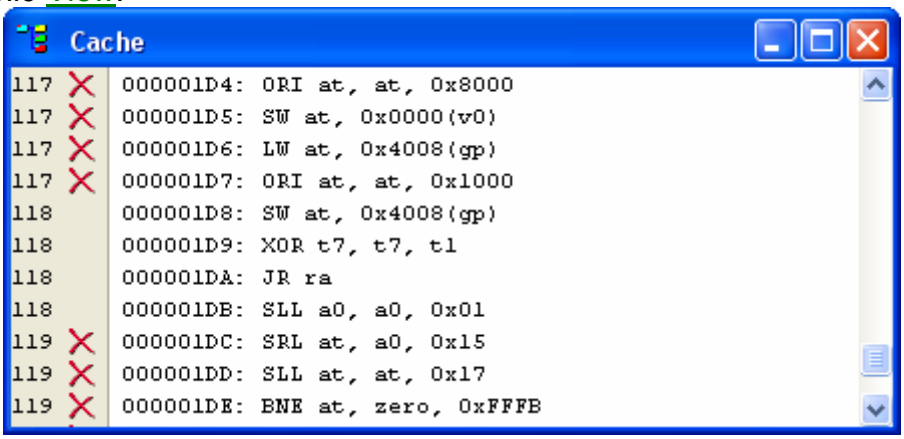
В **окне памяти** отображено содержимое ячеек памяти процессора MultiCore. В поле **Address** указывается адрес области памяти (виртуальный, байтовый, в шестнадцатеричной записи). Кнопка Goto позволяет перейти к введенному адресу. Нажатие стрелки рядом с кнопкой **Goto** открывает выпадающее меню, содержащее список регистров RISC-ядра. Меню позволяет перейти к адресу, содержащемуся в выбранном регистре.

Под полем Address отображается выбранная область памяти. Отображение происходит по четыре слова на строке в шестнадцатеричной записи, младший байт - справа. Слева от каждой строки указывается адрес (виртуальный, байтовый), соответствующий младшему байту первого слова строки.

Содержимое окна доступно только в режиме отладки проекта.

3.4.15. Окно кэша

Окно кэша выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Cache** меню View.

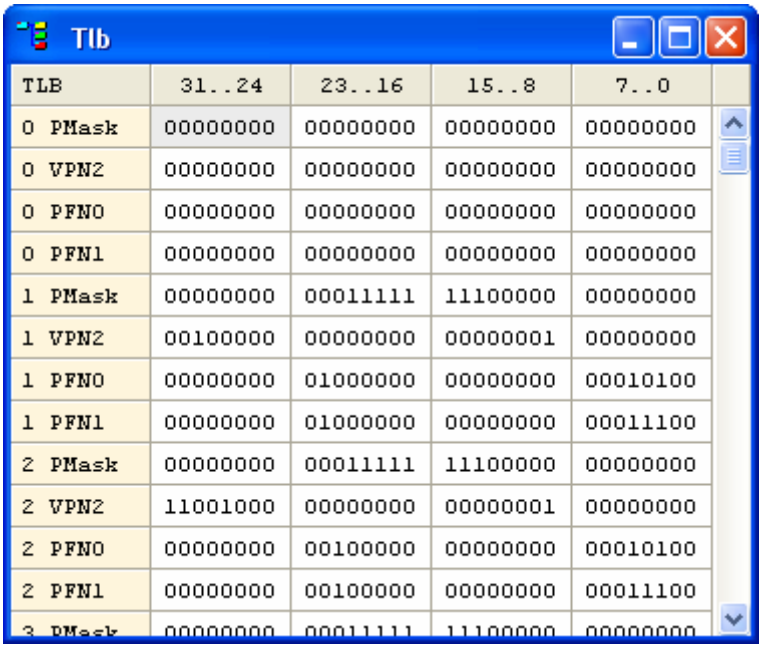


Окно кэша отображает содержимое кэша программ. На каждой строке окна изображено содержимое одного слова строки кэша (одна строка кэша содержит 4 слова). По левому краю окна для каждого слова приведен номер строки кэша.

Содержимое окна доступно только в режиме отладки проекта.

3.4.16. Окно TLB

Окно TLB выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **TLB** меню View.



Окно TLB отображает содержимое буфера быстрого преобразования адреса (*Translation Lookaside Buffer*). Для каждого элемента **TLB** в окне приводится:

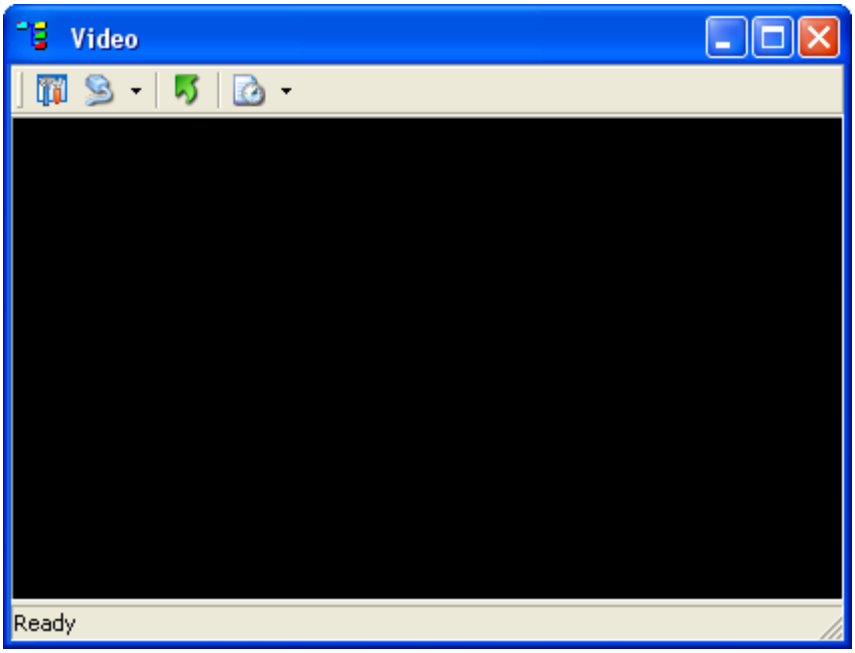
- **PMask** – маска размера страницы (*Page Mask*);
- **VPN2** – виртуальный номер страницы, поделенный на 2 (*Virtual Page Number*);
- **PFN0**, **PFN1** – номер фрейма страницы (*Page Frame Number*), используется либо **PFN0**, либо **PFN1**, в зависимости от установленного бита четности.

По левому краю окна для каждой строки приведен номер элемента **TLB**.

Содержимое окна доступно только в режиме отладки проекта.

3.4.17. Окно видеотерминала

Окно видеотерминала выполнено по методу причаливающего окна и может быть расположено по усмотрению пользователя. Окно появляется при выборе пункта **Video** меню View.



Окно видеотерминала предназначено для вывода изображений в процессе отладки.

Кнопка  открывает диалог настройки выбранного плагина (если плагин имеет диалог настройки).

Кнопка  открывает стандартный диалог открытия файла для загрузки плагина.

Нажатие стрелки рядом с кнопкой позволяет пользователю выбрать плагины из списка доступных:

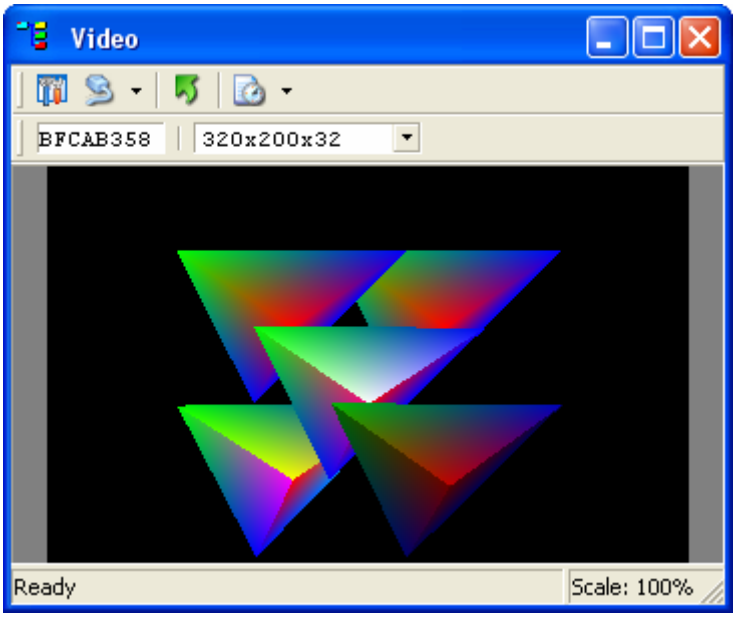
- **Console**;
- **Picture Viewer**;
- **Array Visualizer**.

Нажатие кнопки  осуществляет обновление содержимого экрана видеотерминала.

Кнопка  позволяет осуществлять автоматическое обновление экрана видеотерминала. Стрелка рядом с кнопкой позволяет установить необходимую частоту обновления. Автоматическое обновление экрана осуществляется только при *нажатой* кнопке обновления -  .

Плагин **Console** позволяет во время отладки проекта выводить на экран содержимое буфера вывода *StdOut*.

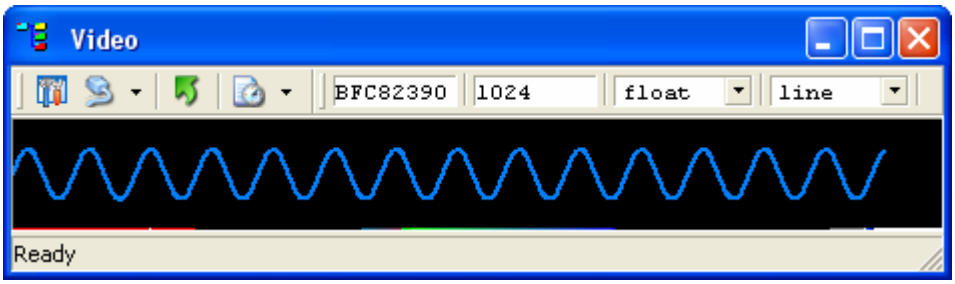
Плагин **Picture Viewer** позволяет отображать область памяти в виде изображения.



Пользователю необходимо ввести адрес видеобуфера и выбрать необходимое разрешение изображения. На рисунке, адрес видеобуфера равен `0xBFCAB358`, разрешение равно `320x200x32`. На строке состояния справа указан масштаб изображения (*Scale: 100%*). Для изменения масштаба следует изменить размер окна видеотерминала.

*Изображение получено при помощи проекта *Pgl*, расположенного в директории *\MCStudio\Samples\Pgl*.

Плагин **Array Visualizer** позволяет выводить содержимое массива в виде линий или точек. Этот плагин применяется для построения графиков.



Пользователю необходимо ввести адрес начала массива, число и тип его элементов, а также тип вывода (**line** или **point**). При выборе **line** каждая точка будет соединяться линией со следующей. При выборе типа вывода **point** точки соединяться не будут. Для вывода допустимы массивы элементов следующих типов: **char**, **short**, **int**, **float**, **double**.

На рисунке изображена синусоида. Точки графика размещены в массиве из 1024 элементов типа **float**, начиная с адреса `0xBFC82390`, каждая точка соединена линией со следующей.

Окно видеотерминала доступно только в режиме отладки проекта.

3.5. Диалоги

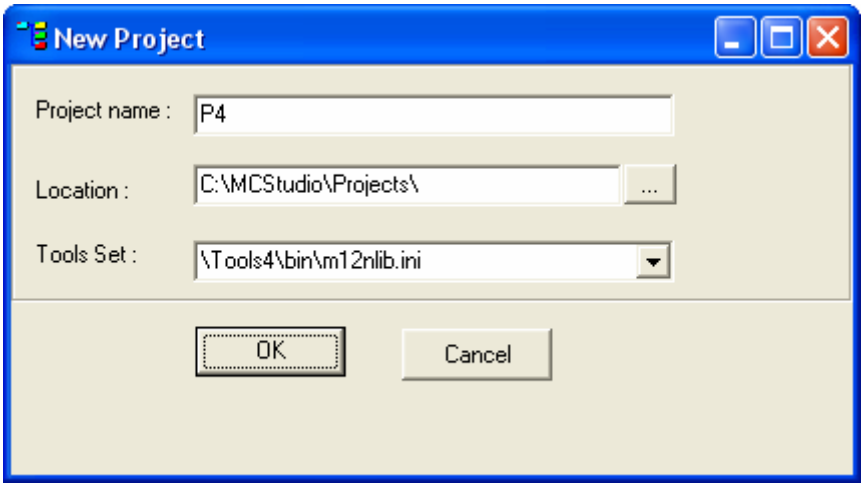
3.5.1. Диалог о системе

Диалог "**О системе**" появляется при запуске среды MCS, а также при выборе пункта **About...** меню Help.

Диалог содержит информацию о разработчике среды MultiCore Studio и дату выхода.

3.5.2. Диалог создания проекта

Диалог создания проекта появляется при выборе пункта **New Project** меню File.



Диалог позволяет пользователю создать новый проект. В поле **Project** name следует ввести имя проекта, в поле **Location** - путь к директории, где будет расположен проект (по умолчанию это *\MCStudio\Projects*). Селектор **Tools Set** позволяет выбрать набор инструментов для работы с проектом.

При этом, если создаваемый проект будет включать в себя текст программы, написанный на языке *C*, следует использовать **только** наборы инструментов *\Tools4\bin\m12nlib.ini* и *\Tools4\bin\m24nlib.ini*. Эти наборы инструментов автоматически подключают библиотеку *Lib*, необходимую для компиляции *C*-программ. Префиксы *m12*, *m24* и *m44* в наименовании набора инструментов означают тип ИМС "*МУЛЬТИКОР*" - МС-12, МС-24 и МС-44 соответственно.

В том случае, когда программа для RISC будет написана только на языке *Ассемблера*, рекомендуется использовать наборы инструментов *\Tools4\bin\m12.ini* и *\Tools4\bin\m24.ini* без библиотеки *Lib*. Если же Вам по тем или иным причинам необходимо подключение этой библиотеки, программу RISC следует писать с соблюдением всех правил связей и вызовов функций языка *C*.

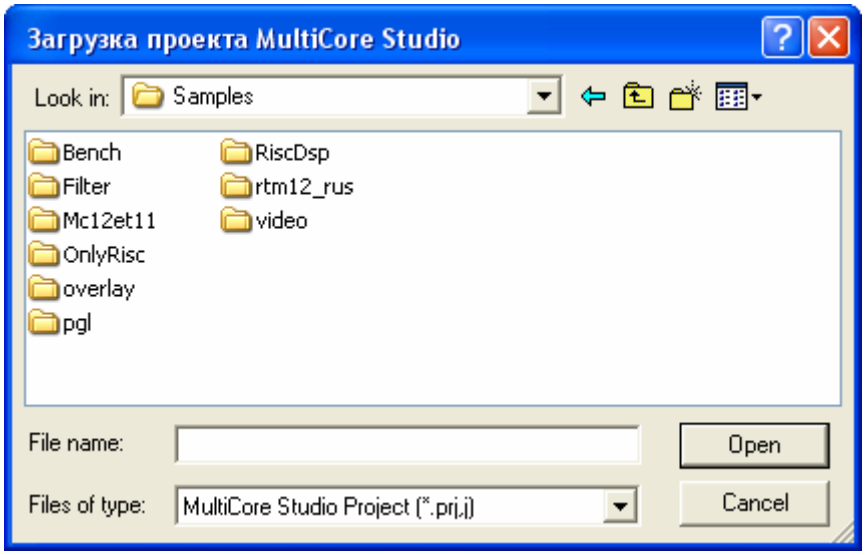
Подробнее о написании программ для процессора MultiCore и для его RISC-ядра в частности см. книгу "**MC Programmer's Guide**".

Нажатие кнопки OK подтверждает создание проекта. При этом в директории, указанной в поле **Location**, создается папка с именем, указанным в поле **Name** и открывается окно проекта, в котором появляется вновь созданный проект.

Нажатие кнопки Cancel отменяет создание проекта и закрывает диалог.

3.5.3. Диалог открытия проекта

Диалог открытия проекта появляется при выборе пункта **Load Project** меню File.

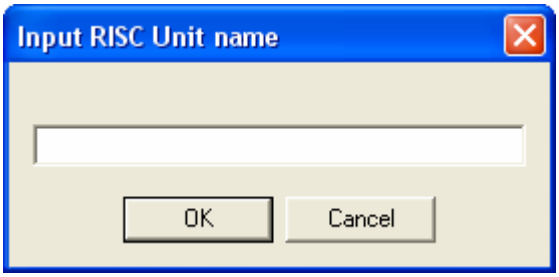


Диалог позволяет пользователю открыть проект среды MC Studio. Для этого необходимо выбрать нужный файл проекта (*.prj,j*) и нажать кнопку Open. После этого появится окно проекта, содержащее вновь открытый проект со всеми файлами и модулями.

Нажатие кнопки Cancel отменяет открытие проекта и закрывает диалог.

3.5.4. Диалог "добавить RISC-модуль"

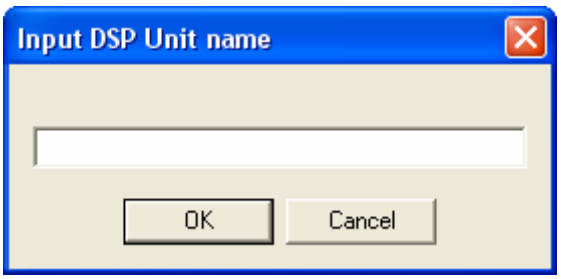
Диалог "**добавить RISC-модуль**" появляется при выборе пункта **Add RISC Unit** меню Project или выпадающего меню окна проекта.



Чтобы добавить модуль RISC, следует ввести имя нового модуля и нажать кнопку OK. Добавленный модуль появится в окне проекта. Чтобы отменить создание модуля, нужно нажать кнопку Cancel.

3.5.5. Диалог "добавить DSP-модуль"

Диалог "**добавить DSP-модуль**" появляется при выборе пункта **Add DSP Unit** меню Project или выпадающего меню окна проекта.

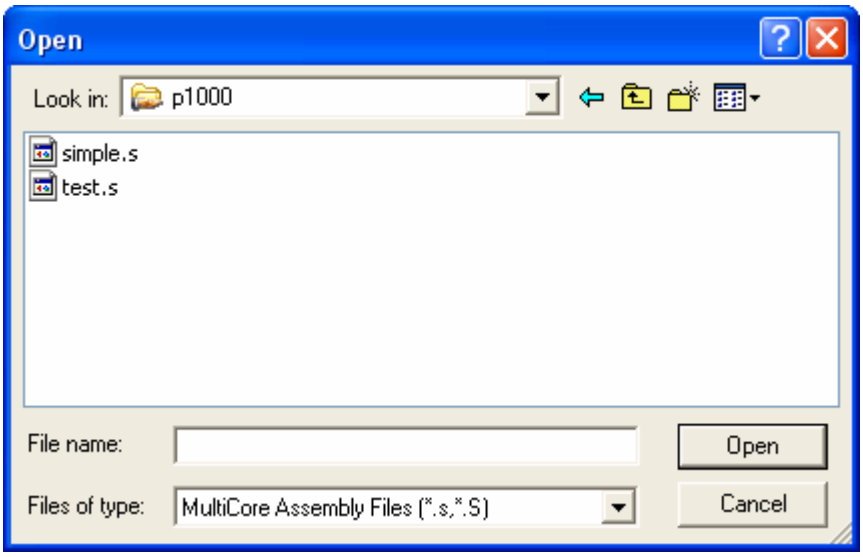


Чтобы добавить модуль DSP, следует ввести имя нового модуля и нажать кнопку OK. Добавленный модуль появится в окне проекта. Чтобы отменить создание модуля, нужно нажать кнопку Cancel.

3.5.6. Диалог открытия файла

Диалог открытия файла является стандартным диалогом ОС Windows.

Диалог появляется при выборе пункта **Open** меню File или при выборе пункта **Add RISC File** (**Add DSP File**) меню Project или выпадающего меню окна проекта.

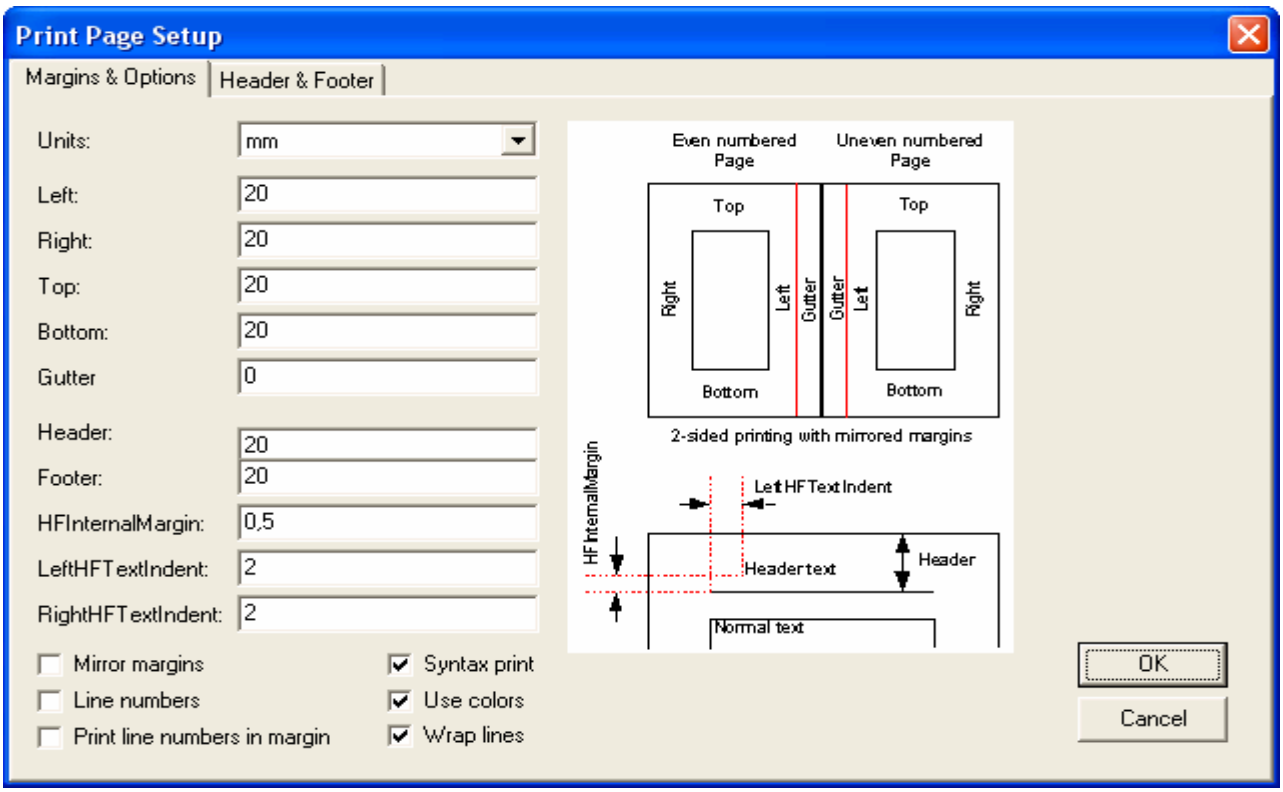


Диалог позволяет пользователю открыть файл для работы с ним в среде MCS. В поле **Имя файла** вводится имя открываемого файла, в селекторе **Тип файлов** - тип файлов, с которыми можно работать в среде MCS. Для подтверждения открытия файла нужно нажать кнопку Open, а для отмены - кнопку Cancel.

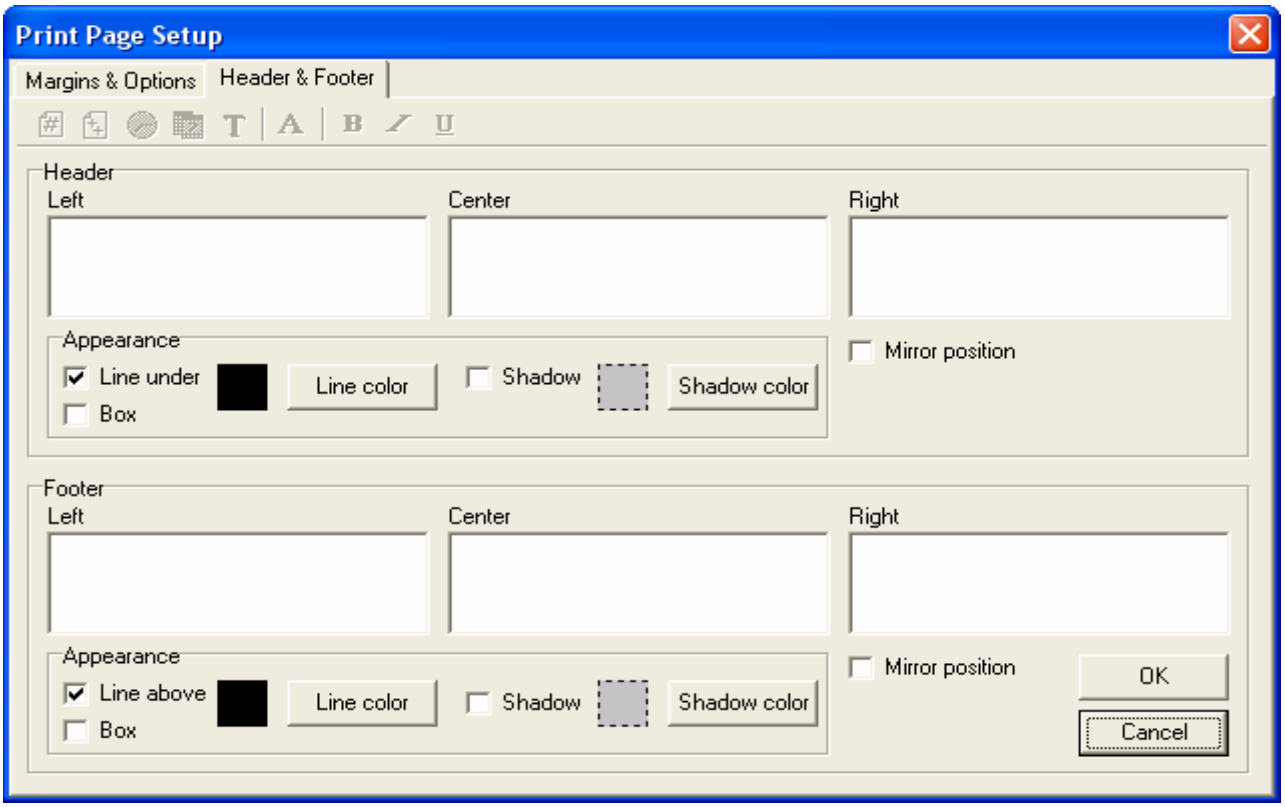
3.5.7. Диалог установки параметров страницы печати

Диалог установки параметров страницы печати появляется при выборе пункта **Print Page Setup** меню File.

Диалог содержит две закладки.



Закладка **Margins & Options** позволяет пользователю изменить отступы на странице.



Закладка **Header & Footer** позволяет вставить в заголовки страницы стандартные данные, такие как дата, время и номер страницы.

Нажатие кнопки OK подтверждает изменение параметров страницы.

Нажатие кнопки Cancel закрывает диалог без изменения параметров страницы.

3.5.8. Диалог предварительного просмотра

Диалог предварительного просмотра появляется при выборе пункта **Print Preview** меню File.

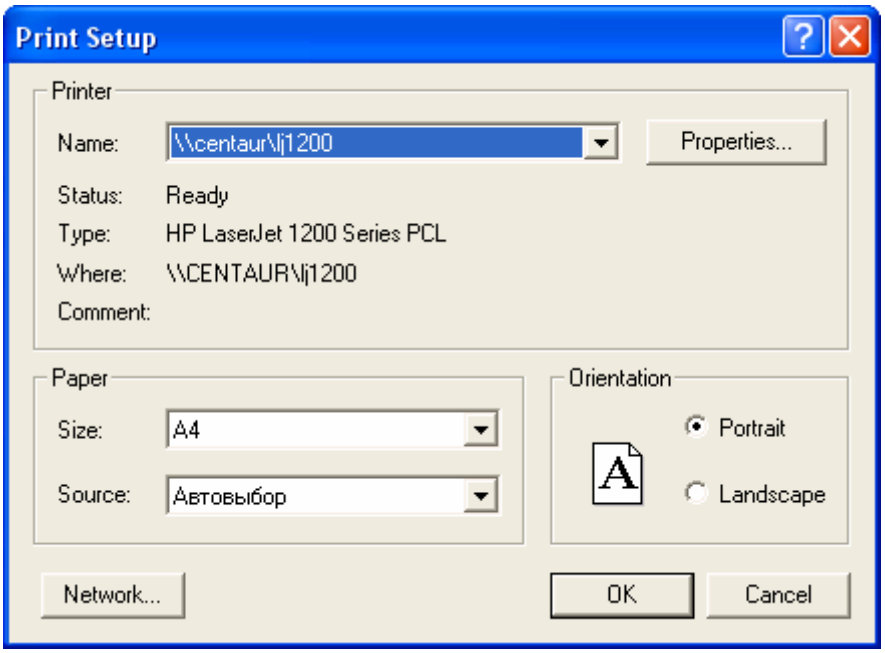
Диалог демонстрирует пользователю разбитый на страницы текст для печати так, как он бы выглядел после распечатки.

Инструментальные кнопки:

-  – позволяет перемещаться между страницами.
-  – увеличить/уменьшить масштаб отображения страницы.
-  – начать распечатку текста.
-  – закрыть **диалог предварительного просмотра**.

3.5.9. Диалог настройки принтера

Диалог настройки принтера является стандартным диалогом ОС Windows. Диалог появляется при выборе пункта **Print Setup** меню File.



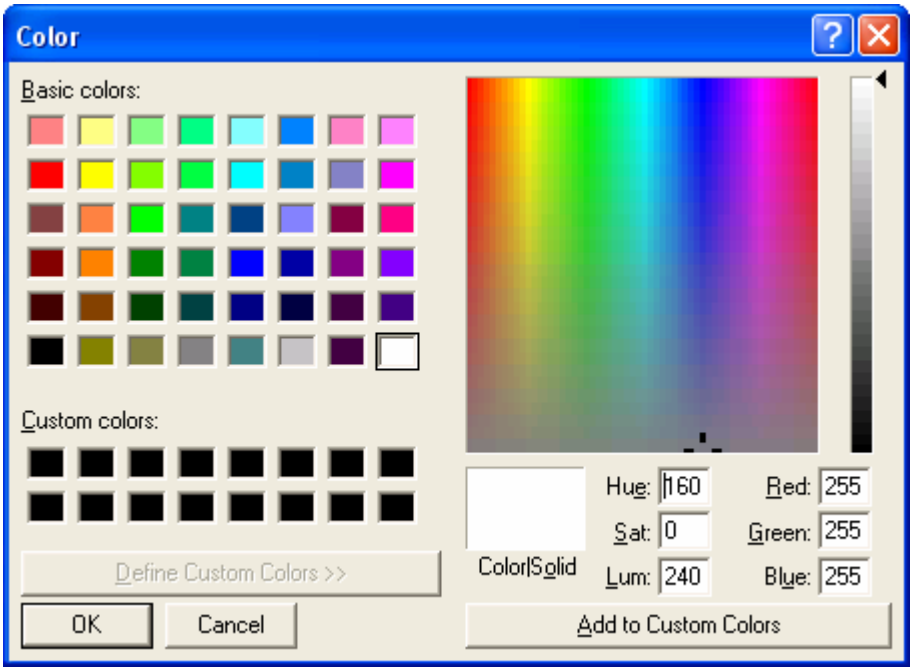
Диалог позволяет пользователю изменить настройки печати, такие как размер и ориентация бумаги, тип подачи бумаги и свойства принтера.

Нажатие кнопки OK подтверждает изменение параметров печати.

Нажатие кнопки Cancel закрывает диалог без изменения параметров печати.

3.5.10. Диалог выбора цвета

Диалог выбора цвета является стандартным диалогом ОС Windows.



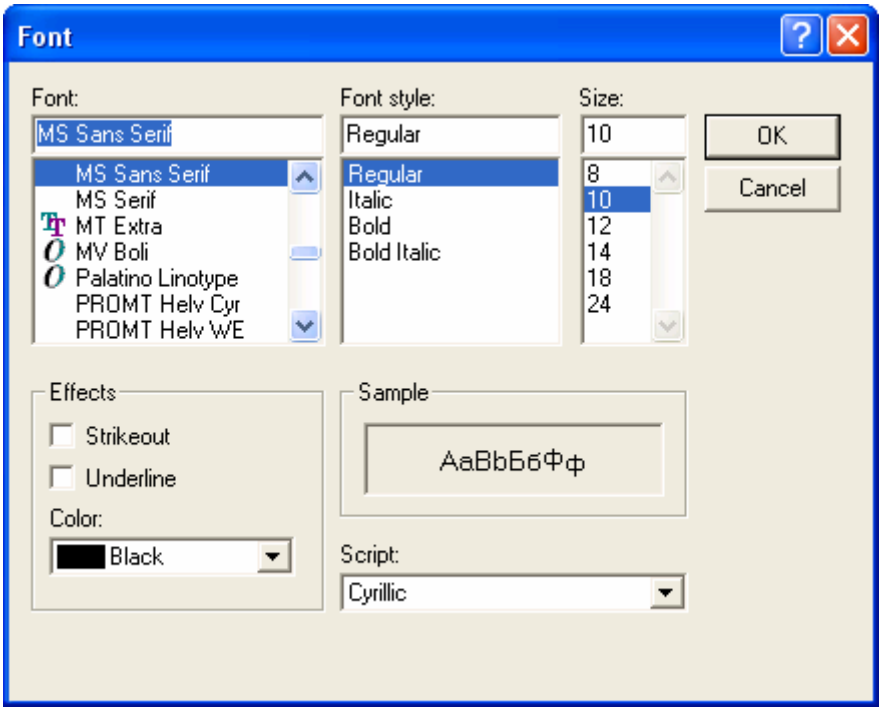
Диалог появляется при нажатии кнопок **Right Edge Color**, **Color**, **Selected Foreground**, **Selected Background** и **Gutter Color** диалога настройки редактора. В зависимости от нажатой кнопки, в диалоге пользователь может выбрать цвет для той или иной области окна редактора. Выбор осуществляется путем щелчка мышью в основной палитре по нужному цвету и нажатия кнопки OK.

Если в основной палитре нет необходимого цвета, его следует выбрать нажатием кнопки Define Custom Colors >>.

Нажатие кнопки Cancel отменяет выбор цвета и закрывает диалог.

3.5.11. Диалог выбора шрифта

Диалог выбора шрифта является стандартным диалогом ОС Windows.



Диалог появляется при выборе пункта **Font** диалога настройки редактора и позволяет настроить шрифт.

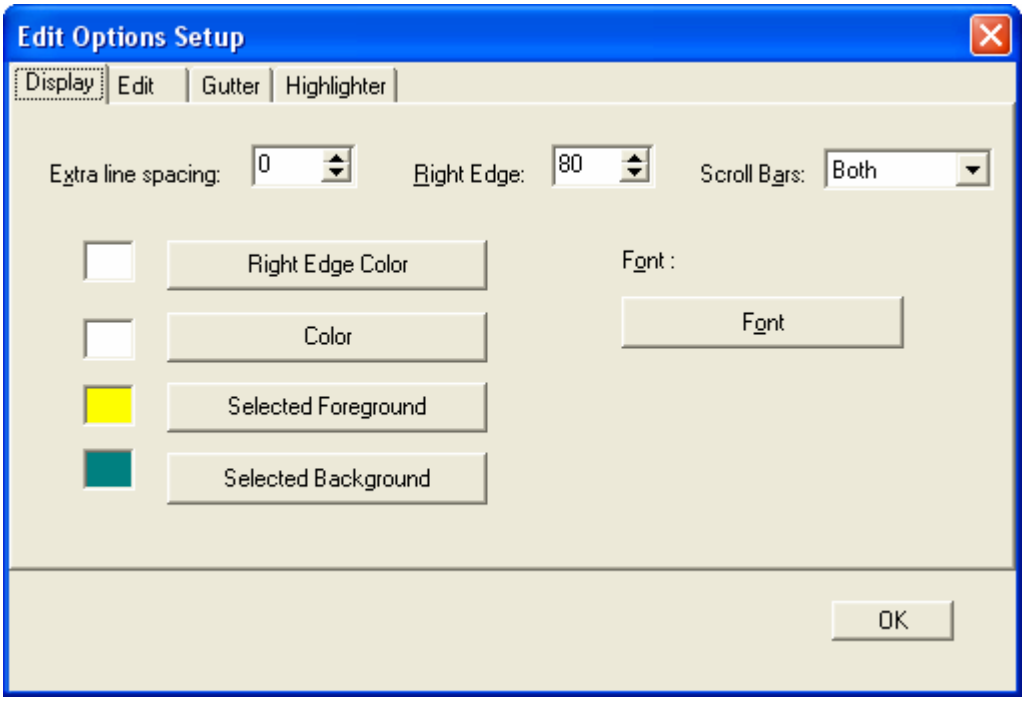
Нажатие кнопки OK подтверждает изменение настроек шрифта и закрывает диалог.

Нажатие кнопки Cancel закрывает диалог без изменений настройки шрифта.

3.5.12. Диалог настройки редактора

Диалог настройки редактора кода появляется при выборе пункта **Settings** меню Edit.

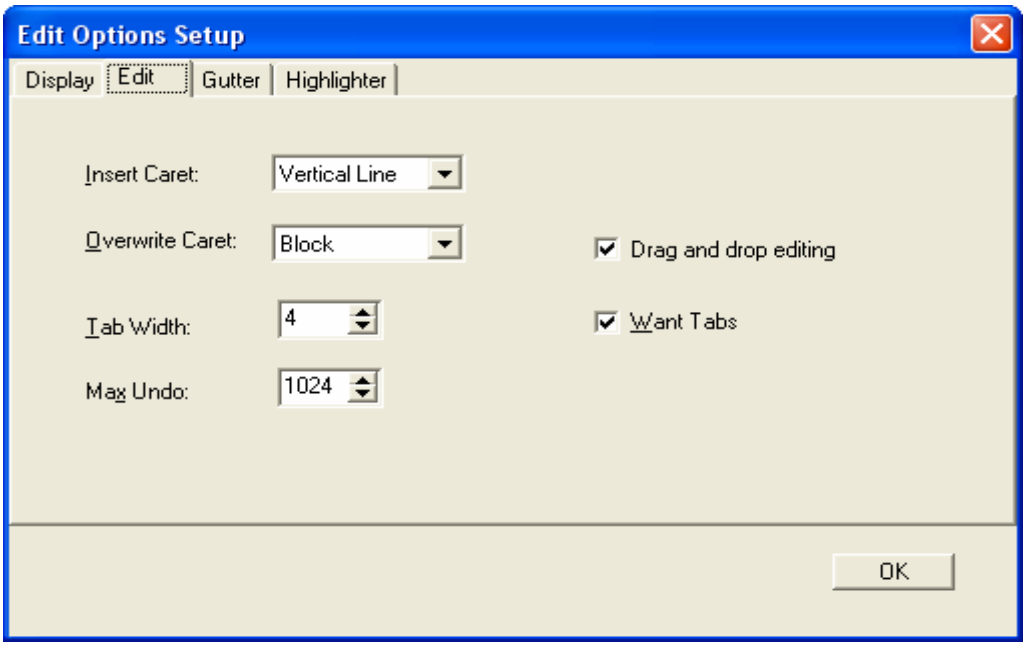
Диалог состоит из пяти закладок и позволяет пользователю изменить настройки окна редактора кода.



Закладка **Display** позволяет пользователю:

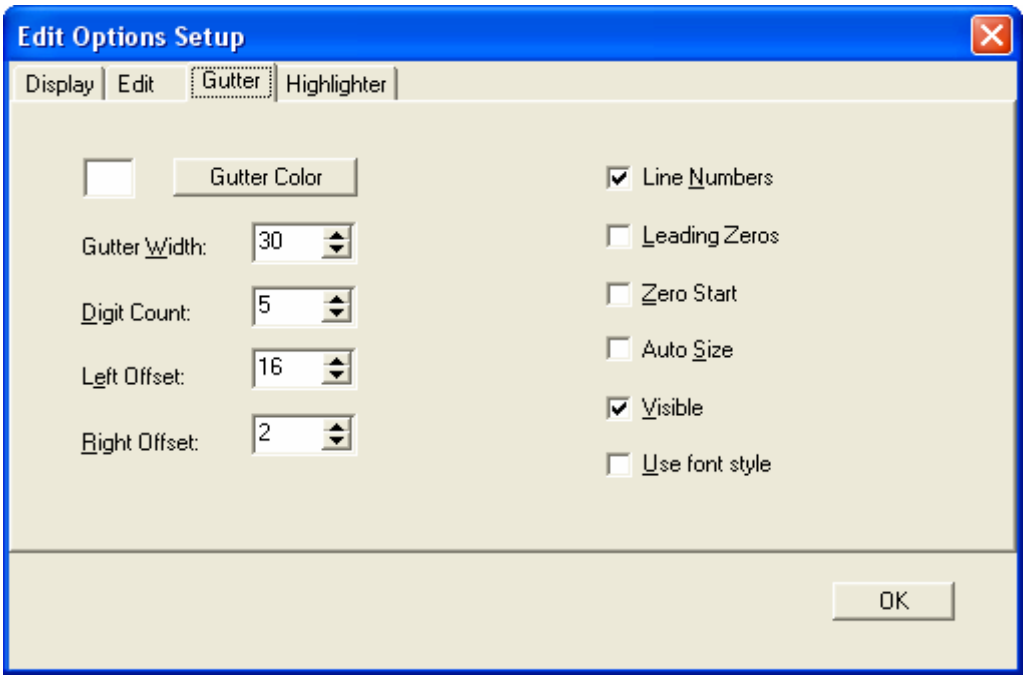
- изменить расстояние между строками (**Extra line spacing**).
- изменить расстояние до правого края клиентской части редактора кода (**Right Edge**).
- включить/выключить полосы вертикальной и горизонтальной прокрутки (**Scroll Bars**).
- выбрать шрифт для окна редактора (**Font**).
- изменить цвет ограничителя правого края (**Right Edge Color**).
- изменить цвет клиентской части окна редактора кода (**Color**).
- выбрать цвет выделенного текста в активном окне (**Selected Foreground**).
- выбрать цвет выделенного текста в неактивном окне (**Selected Background**).

Выборка цвета осуществляется посредством диалога выбора цвета.



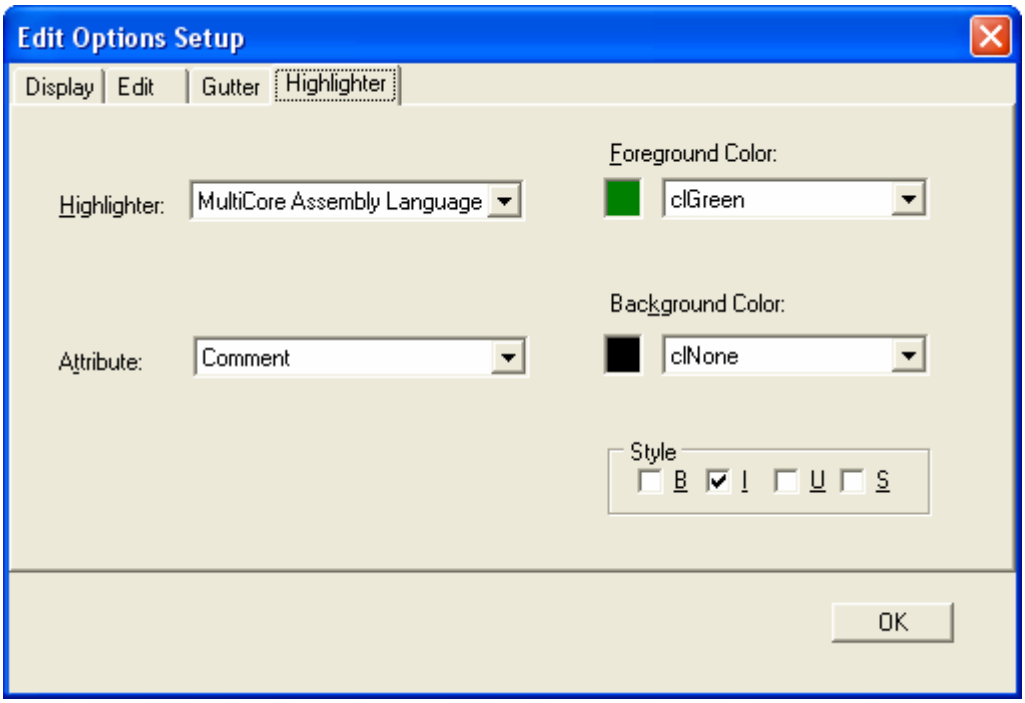
Закладка **Edit** позволяет:

- выбрать вид курсора в режиме Insert (**Insert Caret**). Курсор может иметь вид вертикальной линии, горизонтальной линии, блока и половины блока.
- выбрать вид курсора в режиме Overwrite (**Overwrite Caret**). Курсор может иметь вид вертикальной линии, горизонтальной линии, блока и половины блока.
- изменить ширину символа табуляции (**Tab Width**).
- изменить максимальное количество операций **Undo** меню Edit (**Max Undo**).
- включить/выключить режим редактирования Drag'n'Drop (**Drag and drop editing**).
- включить/выключить символ табуляции (**Want Tabs**).



Закладка **Gutter** позволяет:

- выбрать цвет подшивки окна редактора (**Gutter Color**). Выборка цвета осуществляется посредством диалога выбора цвета.
- изменить ширину подшивки (**Gutter Width**).
- изменить количество знаков для номера строки (**Digit Count**).
- изменить смещение подшивки слева от номеров строк (**Left Offset**).
- изменить смещение подшивки справа от номеров строк (**Right Offset**).
- включить/выключить нумерацию строк на подшивке (**Line Numbers**).
- включить/выключить нули перед значащими цифрами номера (**Leading Zeros**).
- определить номер первой строки кода - 0 или 1 (**Zero Start**).
- включить/выключить автоматическое изменение размеров поля подшивки (**Auto Size**).
- показать/скрыть поле подшивки (**Visible**).
- включить/выключить выборку шрифта для поля подшивки (**Use font style**).



Закладка **Highlighter** позволяет:

- выбрать, какой раздел кода будет подсвечен - код ассемблера MultiCore, код C++, код Batch-файлов, код ini-файлов, или код ассемблера RISC (**Highlighter**).
- выбрать цвет и стиль выделения атрибутов кода - комментариев, идентификаторов, чисел, зарезервированных слов, пробелов, строк и символов (**Attribute**). Цвет выбирается в селекторе **Foreground Color** для текста и в **Background Color** для фона. Стиль выбирается на панели **Style**.

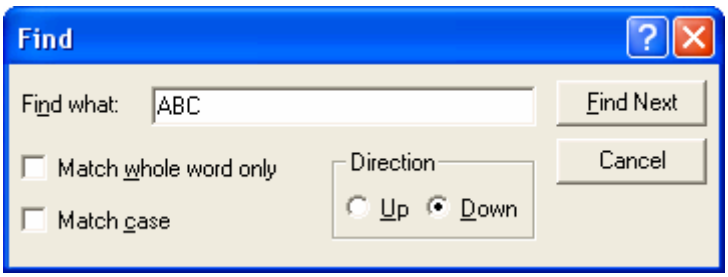
Все изменения, сделанные в **диалоге настройки редактора**, автоматически отображаются на экране в окнах редактора кода.

Нажатие кнопки OK закрывает диалог.

3.5.13. Диалог поиска

Диалог поиска является стандартным диалогом ОС Windows.

Диалог появляется при выборе пункта **Find** меню Edit или при нажатии клавиш <Ctrl>+<F> в активном окне редактора кода.



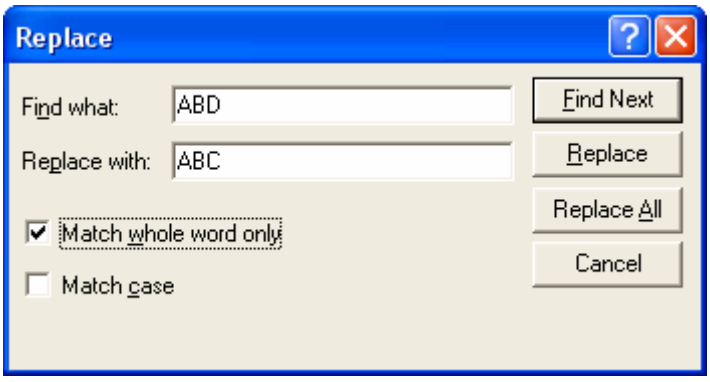
Диалог осуществляет поиск текста Вверх или Вниз по активному окну редактора. Пользователь должен ввести текст для поиска и выбрать параметры поиска. Поиск осуществляется по нажатию кнопки Find Next. Найденный текст выделяется в окне редактора. Кнопка Cancel закрывает **диалог поиска**.

Примечание: Диалог поиска работает только с одним окном редактора - с тем, которое было активно в момент открытия диалога. Для поиска в другом окне необходимо сделать это окно активным и открыть новый диалог поиска. Старый диалог при этом можно не закрывать.

3.5.14. Диалог замены

Диалог замены является стандартным диалогом ОС Windows.

Диалог появляется при выборе пункта **Replace** меню Edit или при нажатии клавиш <Ctrl>+<H> в активном окне редактора кода.



Диалог позволяет искать текст в активном окне редактора и автоматически заменять его введенным текстом.

Пользователю следует ввести искомый текст, новый текст и параметры поиска и замены.

Поиск по тексту осуществляется кнопкой Find Next.

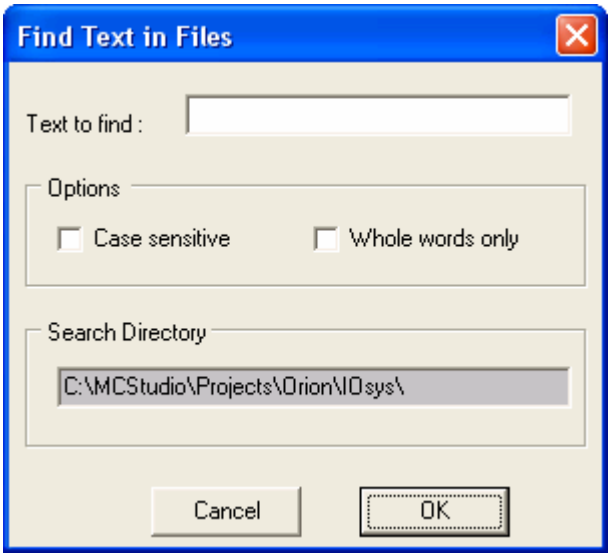
Кнопка Replace осуществляет замену вновь найденного текста, а кнопка Replace All - заменяет сразу все экземпляры текста автоматически.

Кнопка Cancel закрывает **диалог замены**.

Примечание: Диалог замены работает только с одним окном редактора - с тем, которое было активно в момент открытия диалога. Для работы с другим окном необходимо сделать это окно активным и открыть новый диалог поиска. Старый диалог при этом можно не закрывать.

3.5.15. Диалог поиска по всем файлам проекта

Диалог поиска по всем файлам проекта появляется при выборе пункта **Search** меню Project.



Диалог позволяет осуществлять поиск текста по всем файлам проекта. Результаты отображаются в окне поиска.

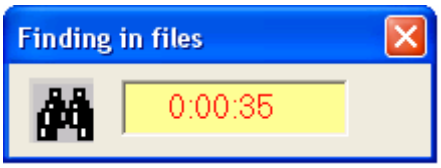
Пользователю следует ввести искомый текст и определить параметры поиска.

Нажатие клавиши OK открывает диалог процесса поиска по всем файлам проекта и начинает поиск.

Кнопка Cancel закрывает **диалог поиска по всем файлам проекта**.

3.5.16. Диалог процесса поиска по всем файлам проекта

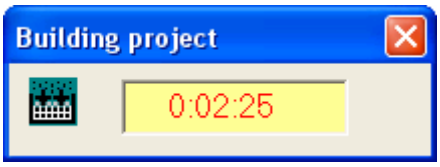
Диалог процесса поиска по всем файлам проекта отображает время, затрачиваемое на поиск.



После окончания поиска диалог автоматически закрывается.

3.5.17. Диалог процесса сборки проекта

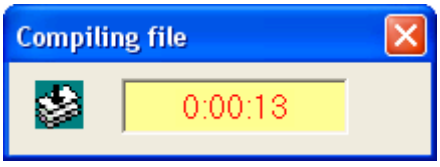
Диалог процесса сборки проекта отображает время, затрачиваемое на сборку проекта.



После окончания сборки диалог автоматически закрывается.

3.5.18. Диалог процесса компиляции файла

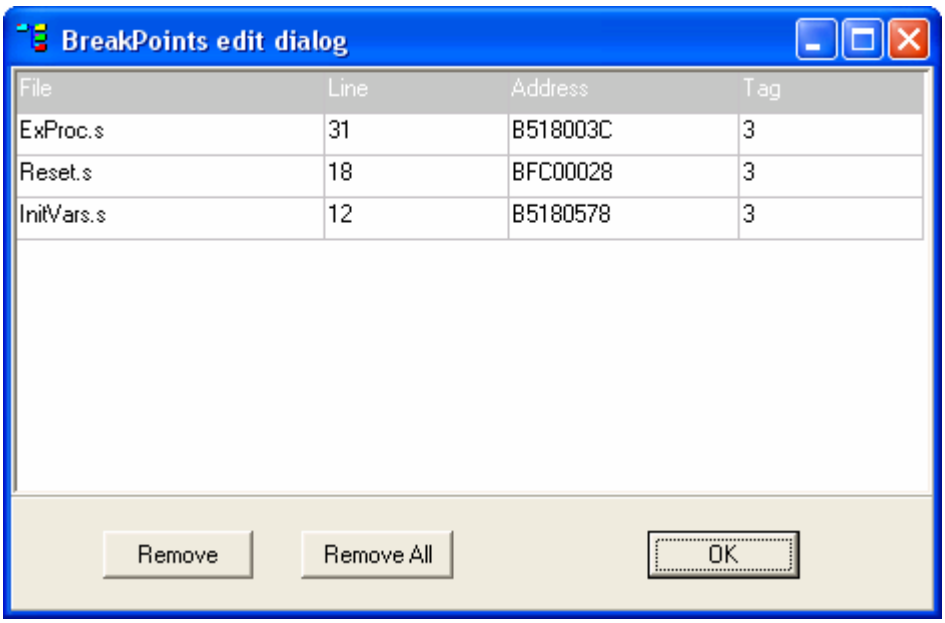
Диалог процесса компиляции файла отображает время, затрачиваемое на компиляцию файла.



После окончания компиляции диалог автоматически закрывается.

3.5.19. Диалог редактирования точек останова

Диалог **редактирования точек останова** появляется при выборе пункта **BreakPoints** меню Edit.



В диалоге перечислены все точки останова (BreakPoints), заданные пользователем. В таблице для каждой точки указано имя файла (**File**), содержащего точку останова, номер строки с точкой останова (**Line**), адрес инструкции в памяти (**Address**) и число привязки (**Tag**).

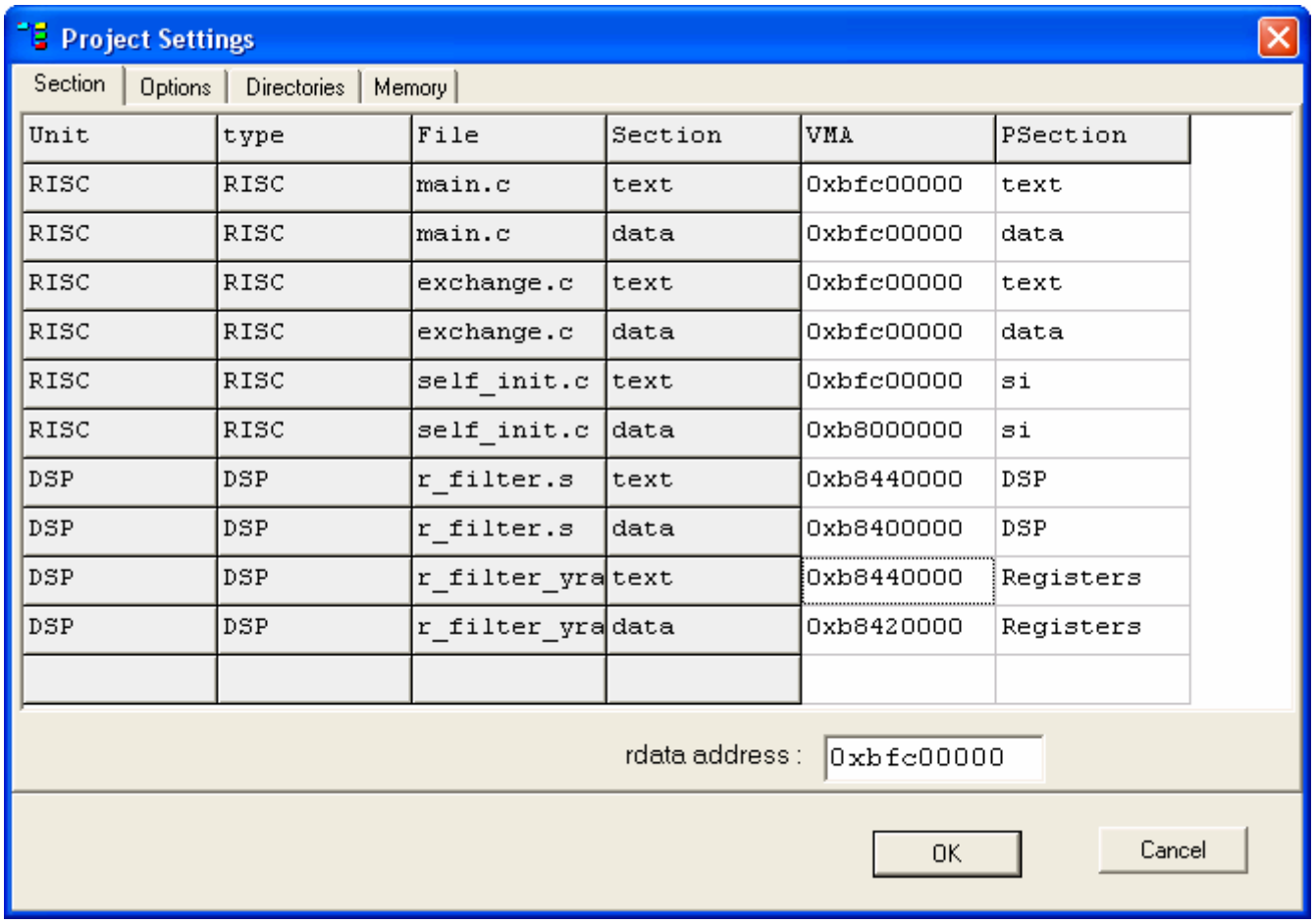
Точки останова используются в режиме отладки проекта. Точки останова отмечены в окне редактора (окне дизассемблера) символом  .

В диалоге **редактирования точек останова**:

- Remove – удалить выбранную в списке точку останова.
- Remove All – удалить все точки останова.
- OK – закрыть диалог редактирования точек останова.

3.5.20. Диалог настроек проекта

Диалог настроек проекта появляется при выборе пункта **Settings** меню Project.



Закладка **Section** выполнена в виде таблицы, содержащей для каждой секции модулей проекта:

- имя модуля (**Unit**) - в этом столбце указано имя модуля;
- тип модуля (**Type**) - в этом столбце указывается тип модуля (RISC или DSP);
- имя файла (**File**) - указывает имя файла с расширением;
- тип секции (**Section**) - в данном столбце указан тип секции (**text** или **data**);
- адрес размещения секции (**VMA**) - в этом поле следует указать адрес, по которому секция будет размещена в памяти **MultiCore**;
- имя выходной секции (**PSection**) - в этом поле необходимо ввести имя выходной секции.

Поле **Set rdata address** позволяет задать виртуальный адрес размещения следующий секций данных: *rdata, gp, lit4, sdata, rodata, edata, fbss, sbss, bss, stack, heap*. Данные секции будут размещены друг за другом сразу после последней секции данных, размещенной в данной области.

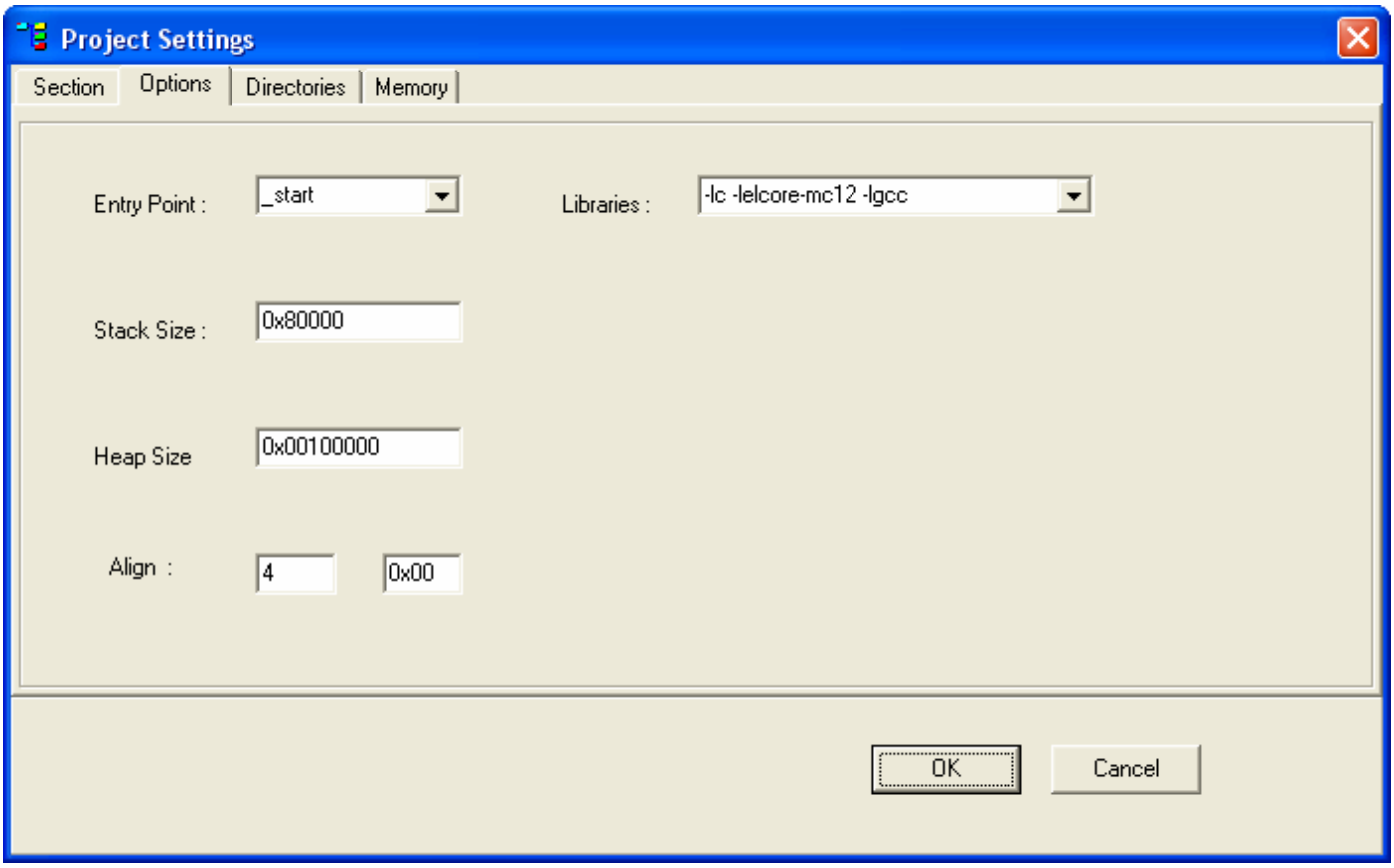
Пользователю предлагается изменить по своему усмотрению имена выходных секций и адреса областей памяти, в которых должны быть размещены секции текста и данных программы в процессе сборки проекта.

Размещать секции по тем или иным адресам необходимо в соответствии со следующими правилами:

- Секции одного типа, имеющие одинаковые имена и адреса **VMA**, будут скомпонованы в памяти одна за другой, начиная с указанного адреса **VMA**, в порядке их отображения в таблице;
- Секция, имеющая уникальное имя будет размещена непосредственно по указанному адресу **VMA**, поэтому при размещении разноименных секций необходимо задавать адреса так, чтобы секции не перекрывали друг друга. Исключениями являются секции DSP.
- При сборке к имени секции будет добавлен суффикс типа секции. Пример: секция *Registers* (см. рисунок) типа **data** будет называться *Registers_data*. Исключениями являются секции с именами *text* и *data* - к ним при сборке суффиксов добавлено не будет.

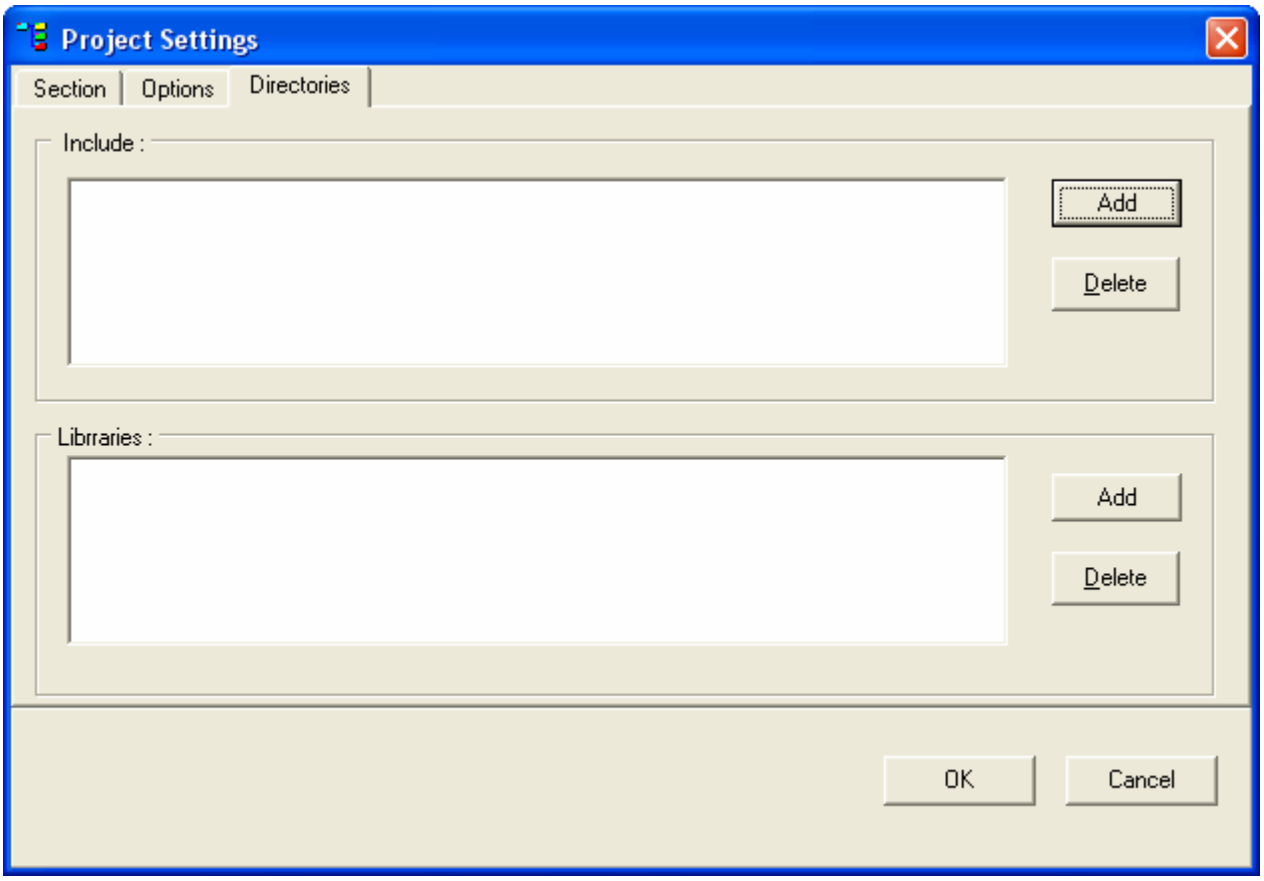
При размещении секций модулей DSP необходимо учитывать следующие правила:

- Младшие 18 разрядов адреса **VMA** представляют байтовое смещение секции относительно начала памяти DSP (**XRAM** для секций типа **data**, **PRAM** для секций типа **text**);
- В старших 14 разрядах адреса **VMA** указывается область памяти для размещения секции DSP. Если эти разряды равны нулю, то секция будет размещена во внешней памяти вслед за секциями RISC (при этом компоновщиком DSP будут использованы младшие 18 разрядов для сборки внутри модуля). Если старшие 14 разрядов не равны нулю, секция будет размещена непосредственно по адресу **VMA**;
- Секции DSP, имеющие разные имена, но одинаковые адреса **VMA**, при сборке будут размещены последовательно, начиная с указанного адреса **VMA**.

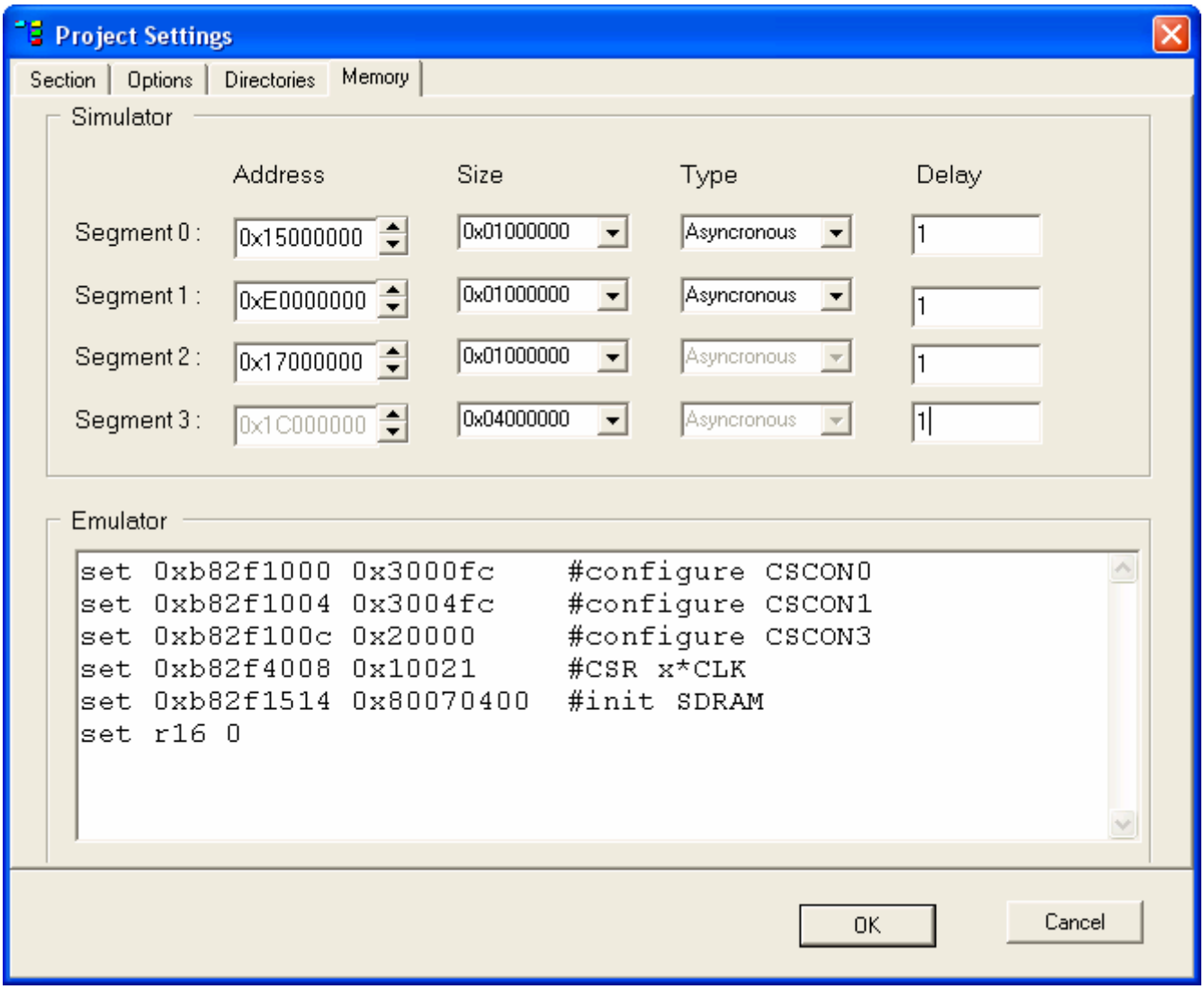


Закладка **Options** позволяет пользователю ввести:

- **Entry Point** - точка входа в программу. Возможны варианты входа с метки **_start** или с метки **_reset**;
- **Stack Size** - размер стека;
- **Heap Size** - размер кучи;
- **Align** - способ выравнивания данных в памяти;
- **Libraries** - подключенные к проекту библиотеки.



Закладка **Directories** позволяет добавить путь для поиска файлов, указанных в директивах `.include` и `#include` текста программы (список путей расположен на панели **Include**), а также добавить пути к используемым библиотекам (панель **Libraries**). Добавление осуществляется нажатием кнопки Add. Чтобы удалить выделенный элемент списка, следует нажать кнопку Delete.



Закладка **Memory** позволяет пользователю настроить адреса *сегментов памяти* (**Address**), их размер (**Size**) и тип (**Type**). Размер задается в диапазоне от `0x04000000` байт до `0x80000000` байт. Тип может быть задан как синхронный (**Syncronous**) или асинхронный (**Asyncronous**). Поле **Delay** задает задержку обращения для каждого сегмента памяти. В случае, если используется асинхронная память, общая задержка на обращение к памяти вычисляется как сумма значения в поле **Delay** с полем задержки в соответствующем регистре **CSCON**. В случае синхронной памяти поле **Delay** не используется.

Примечание: под *сегментами памяти* подразумеваются сегменты 0, 1, 2 и 3, конфигурируемые регистрами **CSCON0**, **CSCON1**, **CSCON2** и **CSCON3**.

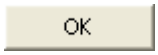
Настройки памяти вступают в действие только после перезапуска симулятора.

Поле **Emulator** задает настройки выполнения проекта в режиме эмулятора. Команда `set <address> <value>` устанавливает при загрузке эмулятора ячейку

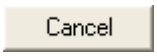
памяти или регистр с адресом <address> в значение <value>. На рисунке в окне эмулятора настраиваются:

- регистры **CSCON0**, **CSCON1** и **CSCON3** - регистры настраиваются на работу с сегментами памяти на плате эмулятора;
- регистр **CSR** - биты 4..7 регистра **CSR** задают множитель частоты. Например, на рисунке эти биты задают значение 2, то есть результирующая частота работы ИМС "*МУЛЬТИКОР*" будет равна 2*CLK, где **CLK** - частота, подаваемая на ИМС от адаптера связи (см. документацию по отладочному модулю);
- инициализация **SDRAM** - данная строка инициализирует динамическую память (настраивает регистр **SDRCON**);
- `set r16 0` - устанавливает содержимое регистра общего назначения №16 равным нулю.

Настройку **CSCON0..3** и **SDRCON** необходимо осуществлять в соответствии с характеристиками модулей памяти, установленных на плате эмулятора. Эти характеристики приведены в документации по отладочному модулю. Описание полей регистров **CSCON0..3** и **SDRCON** рассматривается в руководстве по эксплуатации ИМС "*МУЛЬТИКОР*".

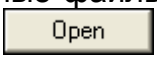
Нажатие кнопки 

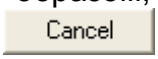
подтверждает изменение **настроек проекта** и закрывает диалог.

Кнопка 

закрывает диалог без изменения настроек.

3.5.21. Диалог исполнения командного файла

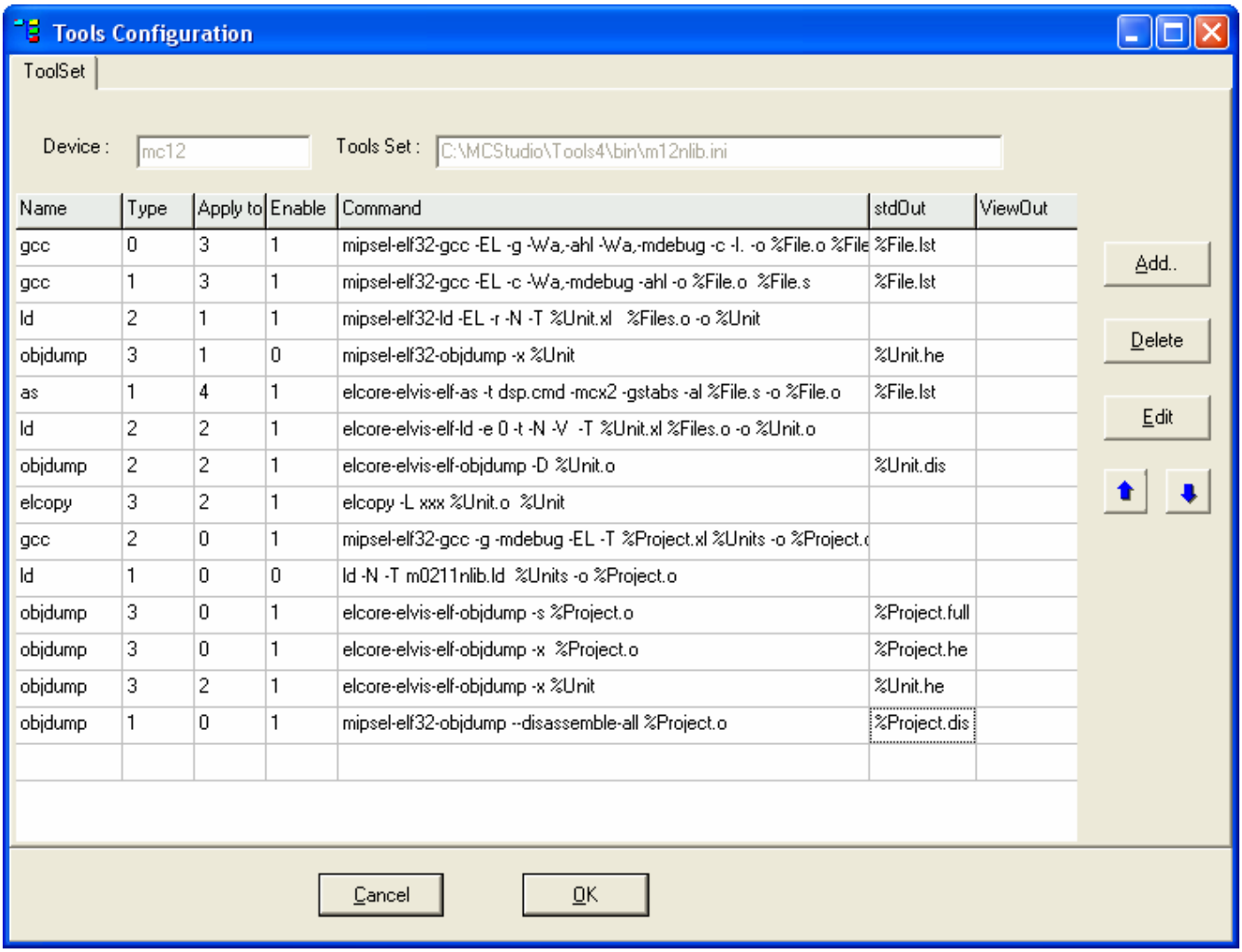
Диалог исполнения командного файла появляется при выборе пункта **Execute** меню Tools. Диалог представляет собой обычный диалог открытия файла и может открывать командные файлы (*.exe,.com,.bat*). После выбора нужного командного файла и нажатия кнопки 

командный файл исполняется. Таким образом, например, можно выполнить командную строку компилятора. Нажатие кнопки 

закрывает диалог без исполнения файла.

3.5.22. Диалог настройки набора инструментов

Диалог настройки набора инструментов появляется при выборе пункта **Settings** меню Tools.



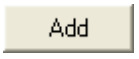
Диалог позволяет пользователю настраивать набор инструментов, используемых при сборке проекта.

В полях **Device** и **Tools Set** указаны соответственно наименование устройства, используемого в проекте, и набор инструментов для сборки. В таблице для каждого инструмента указаны:

- название инструмента (**Name**).
- тип инструмента (**Type**).
- объект применения инструмента (**Apply to**).
- использование инструмента (**Enable**).
- командная строка инструмента (**Command**).

- выходные файлы, формируемые инструментом (**stdOut**).
- показывать ли содержимое файла после сборки в окне сообщений (**viewOut**). Если **viewOut**=1, файл будет показан.

Если необходимый инструмент отсутствует в списке, его следует добавить.

Чтобы добавить инструмент в набор, нужно нажать кнопку 

и в появившемся диалоге открытия файла выбрать нужный инструмент.

Чтобы удалить инструмент из набора, нужно нажать кнопку 

.

Кнопки  и  позволяют передвигаться по списку инструментов.

По умолчанию, проект имеет настройки инструментов, показанные на рисунке выше. Рассмотрим каждую строку этой таблицы:

1. **gcc** – компилятор файлов RISC (**Apply To**=3), написанных на языке С (**type**=0). Командная строка компилятора содержит следующие опции:
- **-EL** – указывает компилировать код в режиме little-endian.
- **-g** – указывает произвести компиляцию с добавлением информации для отладчика.
- **-Wa,-ahl** – **-Wa** передает ассемблеру опцию **-ahl**. Опция **-ahl** указывает добавить к файлам листинга код программы на языках С и Ассемблера
- **-Wa,-mdebug** – **-Wa** передает ассемблеру опцию **-mdebug**. Опция **-mdebug** создает отладочную информацию вида *mdebug*.
- **-c** – указывает, что результатом компиляции должен быть объектный файл.
- **-I.** – указывает искать файлы, подключенные директивами `include`, в текущей директории.
- **-o %File.o %File.c** – опция указывает, что для каждого файла проекта **File.c** выходным файлом будет **File.o**.

Результатом работы компилятора для каждого фала **File.c** будет файл **File.lst**.

2. **gcc** – компилятор файлов RISC (**Apply To**=3), написанных на языке Ассемблера (**type**=1). Командная строка компилятора содержит опции:
- **-EL** – указывает компилировать код в режиме little-endian.
- **-c** – указывает, что результатом компиляции должен быть объектный файл.
- **-Wa,-mdebug** – **-Wa** передает ассемблеру опцию **-mdebug**. Опция **-mdebug** создает отладочную информацию вида *mdebug*.
- **-ahl** – передает ассемблеру опцию **-ahl**. Опция **-ahl** указывает добавить к файлам листинга код программы на языках С и Ассемблера.
- **-o %File.o %File.s** – опция указывает, что для каждого файла проекта **File.s** выходным файлом будет **File.o**.

Результатом работы компилятора для каждого файла **File.s** будет файл **File.lst**.

3. **ld** – компоновщик (**type**=2) модулей RISC (**Apply To**=1). Командная строка компоновщика включает следующие опции:
- **-EL** – указывает связывать объектные файлы в режиме little-endian.
- **-r** – указывает создать перемещаемый выходной файл, то есть файл, который впоследствии может быть использован в качестве входного файла компоновщика **ld**.

- **-N** – указывает установить секции текста и данных доступными для чтения и записи. Также указывает не выравнивать сегмент данных по странице.
- **-T %Unit.xl** – указывает, что для каждого модуля **Unit** директивы компоновщика расположены в файле **Unit.xl**.
- **%Files.o** – указывает, что входными файлами для компоновки каждого модуля являются объектные файлы модуля (**.o*).
- **-o %Unit** – указывает, что выходным файлом для каждого модуля **Unit** будет файл **Unit**.

4. **objdump** – программа отключена (**Enable**=0).
5. **as** – компилятор файлов DSP (**Apply To**=4), написанных на языке Ассемблера (**type**=1). Командная строка включает следующие опции:

- **-t dsp.cmd** – подключает файл **dsp.cmd**, в котором можно описать операции, не включенные в Ассемблер.
- **-mcx2** – опция разрешает использование всех операций, регистров и способов адресации mc02.
- **-gstabs** – указывает добавлять к результирующим файлам отладочную информацию.
- **-al** – указывает добавить к файлам листинга ассемблерный код.
- **%File.s** – указывает, что входными файлами являются все файлы модулей DSP с расширением ***.s***.
- **-o %File.o** – указывает, что для каждого входного файла **File.s** выходным файлом будет **File.o**.

Результатом работы компилятора для каждого файла **File.s** будет файл **File.lst**.

6. **ld** – компоновщик (**type**=2) модулей DSP (**Apply To**=2). Командная строка компоновщика содержит опции:

- **-e 0** – устанавливает начальный адрес в 0.
- **-t** – указывает выводить имена всех входных файлов по мере их обработки.
- **-N** – указывает установить секции текста и данных доступными для чтения и записи. Также указывает не выравнивать сегмент данных по странице.
- **-V** – выводит версию компоновщика и информацию об эмуляции.
- **-T %Unit.xl** – указывает, что для каждого модуля **Unit** директивы компоновщика расположены в файле **Unit.xl**.
- **%Files.o** – указывает, что входными файлами для компоновки каждого модуля являются объектные файлы модуля (**.o*).
- **-o %Unit.o** – указывает, что выходным файлом для каждого модуля **Unit** будет файл **Unit.o**.

7. **objdump** – программа предназначена для проверки, анализа и обработки объектных и выполняемых файлов. В данном случае, программа работает с модулями **DSP** (**Apply To**=2). Командная строка программы включает в себя следующие опции:

- **-D** – дизассемблирование всех секций.
- **%Unit.o** – указывает, что входными файлами для программы будут скомпонованные объектные файлы всех модулей **DSP**.

Результатом работы программы для каждого файла **Unit.o** будет файл с дизассемблированным кодом **Unit.dis**.

8. **elcopy** – программа преобразует объектные файлы модулей DSP в формат RISC (глобальные адреса преобразуются из словного формата в байтовый). Это необходимо

для того, чтобы модули DSP можно было скомпоновать с модулями RISC. Опция **-L xxx** указывает пометить все символы обрабатываемых объектных файлов как локальные.

9. **gcc** – на данном этапе **gcc** компонует все модули проекта в один файл. Командная строка имеет следующие опции:

- **-g** – добавляет информацию для отладчика.
- **-mdebug** – Опция **-mdebug** создает отладочную информацию вида *mdebug*.
- **-EL** – устанавливает режим компоновки little-endian.
- **-T %Project.xl** – указывает, что директивы компоновщика расположены в файле **Project.xl**.
- **%Units** – указывает, что входными файлами для компоновки являются все файлы модулей **Unit**.
- **-o %Project.o** – указывает, что выходным файлом будет файл **Project.o**.

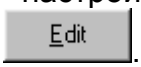
10. **ld** – на данном этапе компоновщик отключен (**Enable**=0).

11. **objdump** – программа обрабатывает объектные файлы. В данном случае, это файл проекта **Project.o**. Опция **-s** указывает включить в результирующий файл полное содержимое всех секций объектного файла. Результирующим файлом будет файл **Project.full**.

12. **objdump** – программа обрабатывает объектные файлы. Здесь обрабатывается файл **Project.o**. Опция **-x** указывает включить в результирующий файл содержимое всех заголовков. Результирующим файлом будет файл **Project.he**.

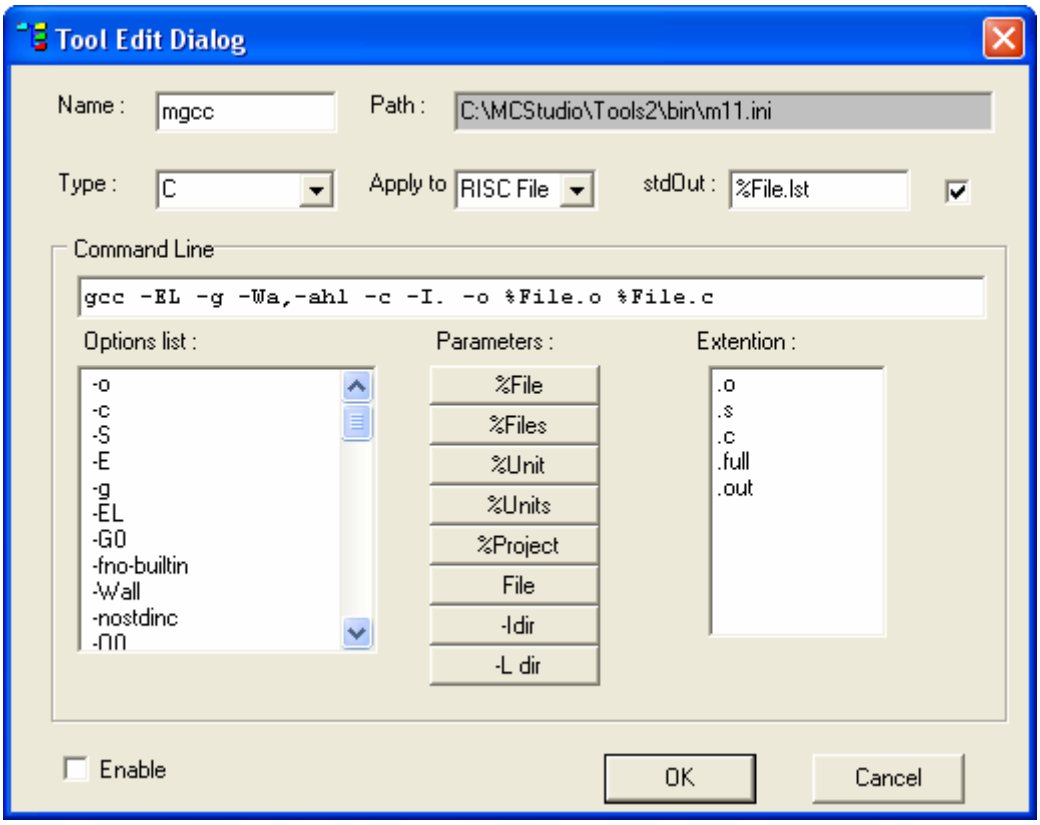
13. **objdump** – программа обрабатывает объектные файлы. Обработке подвергаются файлы модулей DSP проекта (**Unit**). Опция **-x** указывает включить в результирующий файл содержимое всех заголовков. Результирующими файлами будут файлы **Unit.he**.

14. **objdump** – на последнем этапе обрабатывается объектный файл проекта **Project.o**. Опция **--disassemble-all** указывает включить в результирующий файл содержимое всех секций в ассемблерном виде. Результирующим файлом будет файл **Project.dis**.

Чтобы изменить настройки инструмента, необходимо выделить инструмент мышью и нажать кнопку 

.

Появится **диалог настройки инструмента**:



Диалог настройки инструмента позволяет:

- выбрать тип инструмента - инструмент С, инструмент ассемблера, линковщик, или другой тип (**Type**). В таблице типы инструментов обозначаются цифрами от 0 (**инструмент С**) до 3 (**Other**).
- выбрать объект применения инструмента - проект, модуль RISC, модуль DSP, файл RISC, файл DSP (**Apply to**). В таблице объекты применения обозначаются цифрами от 0 (**Project**) до 4 (**DSP file**).
- изменить тип выходных данных инструмента (**stdOut**).
- изменить командную строку (**Command Line**) инструмента - добавить ключи (**Options List**) и параметры (**Parameters**).
- включить/выключить инструмент в/из процесса сборки проекта (**Enable**).

Галочка, установленная справа от имени выходного файла, означает, что после сборки файл будет выведен в окне сообщений. При установке галочки, параметр **viewOut** строки настроек инструмента установится в единицу.

Рассмотрим возможные формальные параметры панели **Parameters**:

- **%File** - вместо этого параметра будет подставлено фактическое имя файла. Например, вместо параметра **%File.s** будет подставлено реальное имя файла с расширением **.s**. Набор файлов определяется селектором **Apply To**.

- **%Files** - вместо этого параметра будут подставлены фактические имена всех файлов. Например, вместо **%Files.o** будут подставлены имена всех объектных файлов. Набор файлов также определяется селектором **Apply To**.
- **%Unit** - вместо этого параметра будет подставлено фактическое имя модуля. Это может быть модуль RISC или DSP, в зависимости от селектора **Apply To**. Например, вместо **%Unit.o** будет подставлено имя объектного файла, совпадающее с именем модуля.
- **%Units** - вместо этого параметра будут подставлены имена всех модулей RISC или DSP, в зависимости от селектора **Apply To**. Например, вместо **%Units.o** будут подставлены имена всех объектных файлов, совпадающие с именами модулей.
- **%Project** - вместо параметра **%Project** будет подставлено имя проекта. Например, вместо **%Project.o** будет подставлен объектный файл с именем проекта.
- кнопка **File** открывает диалог открытия файла. Полный путь к выбранному в диалоге файлу будет включен в командную строку.
- кнопка **-Idir** открывает диалог выбора директории. Путь к выбранной директории будет добавлен в командную строку с ключом **-I**. Директория будет использоваться для поиска файлов, указанных в директивах `include`.
- кнопка **-L** dir открывает диалог выбора директории. Путь к выбранной директории будет добавлен в командную строку с ключом **-L**. Директория будет использоваться для поиска подключенных библиотек.

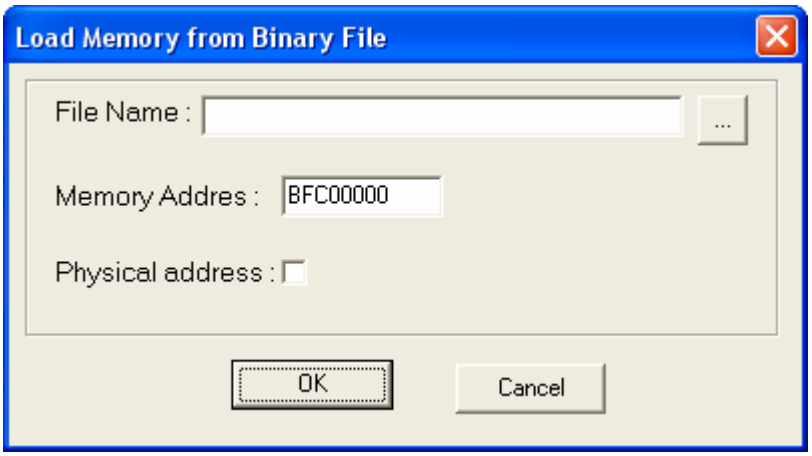
Нажатие кнопки OK подтверждает изменение **настроек инструмента** и закрывает диалог.

Кнопка Cancel закрывает диалог без изменения настроек.

Подробнее об инструментах RISC и DSP см. книги "**Инструменты RISC**" и "**Инструменты DSP**".

3.5.23. Диалог загрузки образа памяти

Диалог загрузки образа памяти появляется в режиме отладки проекта при выборе пункта **Load** меню Debug.



Диалог позволяет пользователю загрузить данные из бинарного файла по адресу, указанному в **Memory Address**. Адрес может быть физическим или виртуальным, это определено в поле **Physical Address**.

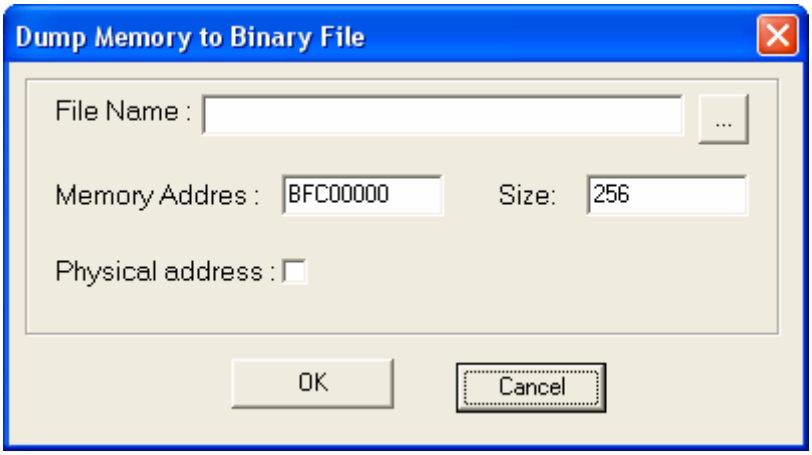
Нажатие кнопки ... открывает стандартный диалог открытия файла для выбора нужного бинарного файла.

Нажатие кнопки OK подтверждает **загрузку образа памяти** и закрывает диалог.

Кнопка Cancel отменяет загрузку.

3.5.24. Диалог записи образа памяти

Диалог записи образа памяти появляется в режиме отладки проекта при выборе пункта **Dump** меню Debug.



Диалог позволяет пользователю записать образ памяти размера **Size** (в байтах), начиная с адреса **Memory Address**, в бинарный файл, указанный в поле **File Name**. Адрес может быть физическим или виртуальным, это определено в поле **Physical Address**.

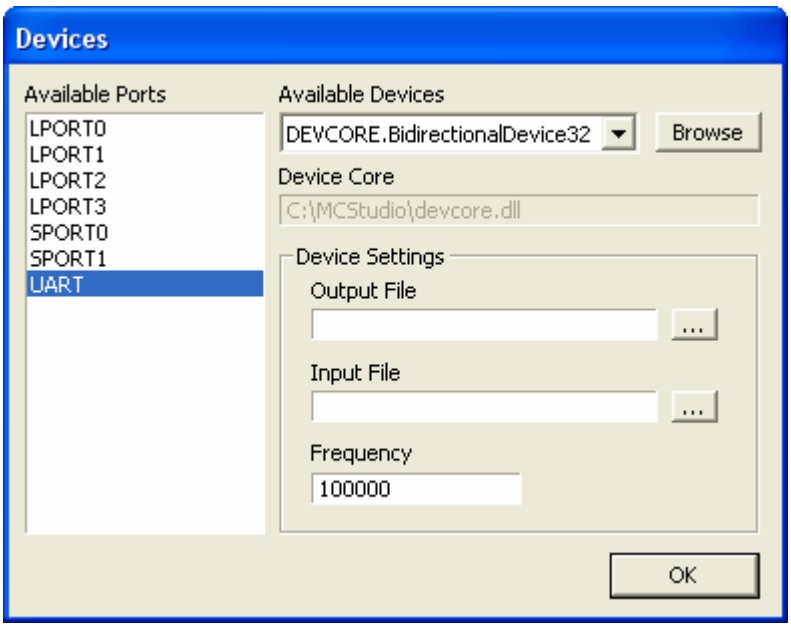
Нажатие кнопки ... открывает стандартный диалог открытия файла для выбора нужного бинарного файла.

Нажатие кнопки OK подтверждает **запись образа памяти** и закрывает диалог.

Кнопка Cancel отменяет запись.

3.5.25. Диалог подключения внешнего устройства

Диалог подключения внешнего устройства появляется в режиме отладки проекта при выборе пункта **Connect Device** меню Debug.



Диалог позволяет пользователю подключить библиотеку внешнего устройства к симулятору для отладки взаимодействия внешнего устройства с процессором MultiCore. По нажатию кнопки Browse появляется диалог открытия файла. В этом диалоге пользователю следует выбрать DLL-файл необходимого устройства. Селектор **Available Devices** позволяет выбрать устройство из списка доступных. Порт, к которому следует подключить устройство, нужно выбирать из списка **Available Ports**. Панель **Device Settings** содержит диалог настроек устройства, экспортируемый из библиотеки DLL устройства. По нажатию кнопки OK устройство подключается к симулятору.

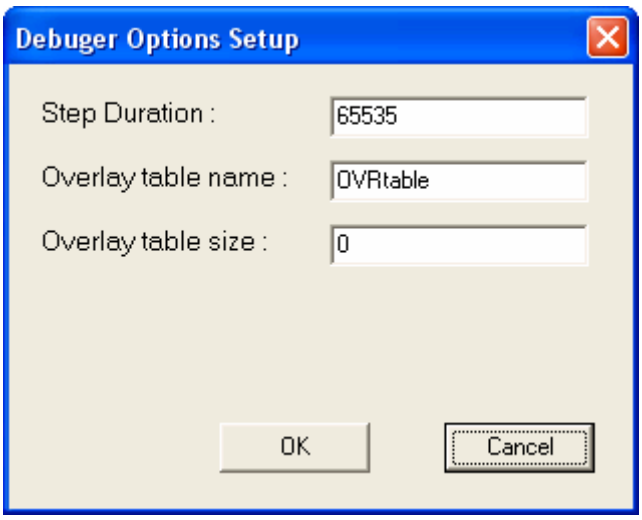
При подключении устройства, в папке проекта автоматически создаются ini-файлы с информацией об этом устройстве.

Среда MC Studio поставляется с несколькими готовыми внешними устройствами. Они рассмотрены в главе "Работа с внешними устройствами".

Процесс создания собственных внешних устройств рассмотрен в книге "**MC Programmer's Guide**".

3.5.26. Диалог настройки отладчика

Диалог настройки отладчика появляется при выборе пункта **Settings** меню Debug.



Диалог содержит следующие поля:

- **Step Duration** - продолжительность шага отладчика в тактах *МУЛЬТИКОР*;
- **Overlay table name** - имя отладочной таблицы оверлейных секций проекта;
- **Overlay table size** - размер отладочной таблицы оверлейных секций проекта (число записей).

4. Работа с проектом

4.1. Введение

Работа с проектом программы для ИМС "*МУЛЬТИКОР*" состоит из следующих этапов:

- Создание проекта;
- Добавление модулей к проекту;
- Добавление файлов к модулям проекта;
- Настройка проекта;
- Сборка проекта;
- Отладка проекта.

Примечание: прежде, чем загружать проект в отладочный модуль (эмулятор), необходимо правильно задать настройки регистров **MPort** и множитель тактовой частоты.

4.2. Создание проекта

Для того чтобы **создать новый проект**, нужно выбрать пункт **New Project** меню File. Откроется диалоговое окно создания проекта. В нем предлагается дать проекту **имя**, указать его **расположение** и выбрать **набор инструментов**, после чего вновь созданный проект появится в окне проекта, а в директории проекта будет записан файл с информацией о проекте (*.prj*).

В проект среды MCS могут входить три типа файлов:

- файл программы на **ассемблере** для DSP, имеет расширение *.s*,
- файл программы на **ассемблере** для RISC, имеет расширение *.s*,
- файл программы на **C** для RISC, имеет расширение *.c*.

В проекте файлы группируются в **модули** (*units)*.

Различают два типа модулей – **модуль** RISC и **модуль** DSP.

Критерии для обьединение файлов в RISC-модули выбираются пользователем, разбиение файлов на DSP-модули определяется созданием оверлейной структры в памяти программ DSP-ядра, то есть DSP-модуль представляет оверлейную секцию, загружаемую в память программ DSP-ядра.

Все **модули** проекта и файлы, присоединенные к ним, отображаются в окне проекта. Символами  обозначены модули RISC, символами  - модули DSP.  - файлы модулей RISC,  - файлы модулей DSP. Все файлы, относящиеся к проекту размещаются в одной директории.

4.3. Добавление модуля к проекту

Для того, чтобы **добавить RISC-модуль** (*unit*) к проекту, нужно выбрать пункт **Add RISC Unit** меню Project или выпадающего меню окна проекта, после чего в диалоге "добавить RISC-модуль" ввести имя нового **модуля**. Новый **модуль** появится в окне проекта с символом  .

Для того чтобы **добавить DSP-модуль** (*unit*) к проекту, нужно выбрать пункт **Add DSP Unit** меню Project или выпадающего меню окна проекта, после чего в диалоге "добавить DSP-модуль" ввести имя нового **модуля**. Новый **модуль** появится в окне проекта с символом  .

4.4. Добавление файлов к проекту

Чтобы **добавить файл к проекту** необходимо:

- выбрать в окне проекта **модуль** (*unit*), к которому следует добавить файл;
- выбрать пункт **Add RISC File** (или **Add DSP File**) в меню Project или в выпадающем меню окна проекта;
- появится диалог открытия файла. В этом диалоге следует выбрать нужный файл. Если этот файл расположен вне директории проекта, он будет автоматически скопирован, так как все файлы проекта должны храниться в одной директории. Если нужно создать новый файл, в диалоге следует ввести имя для создаваемого файла.

После этого вновь открытый (созданный) файл добавится к выбранному модулю в окне проекта с символом  для файла RISC, с символом  для файла DSP и с символом  для файла заголовков.

Примечание: имя создаваемого файла проекта не должно совпадать с именем модуля проекта или с именем другого файла в любом из модулей, так как это приведет к ошибкам при сборке проекта.

4.5. Удаление файлов и модулей из проекта

Чтобы **удалить** какой-либо элемент (**файл** или **модуль**) из **проекта**, необходимо выделить его в окне проекта и выбрать пункт **Remove** меню Project или выпадающего меню окна проекта. Элемент будет удален из проекта и исчезнет из окна проекта. При удалении **модуля** удаляются все файлы, присоединенные к нему. При удалении **проекта**, удаляются все **модули** и файлы проекта.

4.6. Открытие проекта

Для того, чтобы **открыть проект**, необходимо выбрать пункт **Load Project** меню File. В появившемся диалоге открытия проекта следует выбрать нужный проект. После этого проект будет открыт и появится в окне проекта вместе со всеми **модулями** и файлами, присоединенными к нему.

4.7. Сохранение проекта

Чтобы **сохранить проект**, нужно выбрать пункт **Save Project** меню File. После этого проект будет сохранен в директории, указанной при создании проекта.

При выходе из среды MCS, если проект был изменен, но не сохранен, система автоматически предложит **сохранить** его.

4.8. Настройки

4.8.1. Настройка редактора

Настройка окна редактора кода позволяет пользователю изменить внешний вид редактора кода по своему усмотрению. Также она позволяет настроить подсветку частей кода и выделение отдельных инструкций.

Чтобы настроить окно редактора кода, нужно выбрать пункт **Settings** меню Edit. Все изменения производятся в появившемся диалоговом окне настройки редактора.

4.8.2. Настройка проекта

Перед сборкой и отладкой, проект необходимо настроить:

- Установить адреса размещения секций текста и данных проекта;
- Настроить точку входа в программу, размер стека и кучи, а также, при необходимости, подключить дополнительные библиотеки;
- Настроить сегменты памяти (адреса, размер, тип, задержку) для симулятора;
- Настроить регистры **MPort** и множитель тактовой частоты для отладочного модуля.

Для осуществления настроек проекта нужно выбрать пункт **Settings** меню Project. Все изменения проводятся в появившемся диалоговом окне настройки проекта.

4.8.3. Подключение инструментов

Подключение инструментов позволяет пользователю выбирать и настраивать набор инструментов, используемый
при сборке проекта. Это необходимо для правильной компиляции, сборки и отладки проекта.

Весь инструментарий при установке среды MCS записывается в директорию *…/MCS/Tools4/bin* и все инструменты автоматически подключаются для обработки соответствующих файлов и модулей в необходимой последовательности. Однако, пользователь может изменить **настройки инструментов** в диалоговом окне настроки набора инструментов, появляющемся при выборе пункта **Settings** меню Tools.

Подробнее об инструментах и их настройке см. книги "**Инструменты RISC**" и "**Инструменты DSP**".

4.9. Сборка проекта

Для того, чтобы выполнить **сборку проекта**, необходимо выбрать пункт **Build** меню Project.

В процессе **сборки проекта** для каждого **модуля** выполняется компиляция файлов, затем **линковка** файлов, входящих в **модуль**. После обработки всех модулей выполняется **линковка** всего проекта. Если **линковка** была успешной, то выполняется преобразование обьектного файла в загрузочный файл.

В результате **сборки проекта** в директорию проекта записываются следующие файлы:

- Файлы с расширением *.txt* и именем памяти. Например, *bank3.txt* – образы памяти для загрузки в **симулятор**;
- Файлы с расширением *.bin* – файлы с двоичным кодом секции;
- Файлы с расширением *.o* или без расширения – обьектные файлы;
- Файлы с расширением *.lst* – листинги откомпилированных файлов;
- Файлы с расширением *.dis* – дизассемблированные обьектные файлы. Их можно загружать в окна дизассемблера;
- Файл с расширением *.ldr* – перечень файлов с загрузочным кодом для загрузки проекта в **симулятор**;
- Файлы с расширением *.xl* – автоматичеси формируемые MCS-скрипты для **линковки**.
- Файл с расширением *.full* отображает содержимое всех секций проекта.
- Файлы с расширением *.he* отображают карту памяти и таблицу символов всего проекта и модуля DSP (отдельно).
- Файл *StdErr.txt* – протокол сообщений об ошибках, обнаруженных в процессе обработки проекта.
- Файл *StdOut.txt* – протокол выходного потока *StdOut*.

Сообщения о результатах сборки проекта появляются в окне сообщений.

Инструменты, используемые при сборке проекта, настраиваются в диалоге настройки набора инструментов.

Адреса размещения секций проекта в памяти указываются в диалоге настроек проекта.

Примечание: файлы *.bin*, *.txt* и *.ldr* появляются после запуска симулятора.

Подробнее о процессах компиляции и линковки см. книги "**Инструменты RISC**" и "**Инструменты DSP**".

4.10. Отладка проекта

4.10.1. Запуск и останов отладчика

Запуск отладчика осуществляется выбором пункта **Start** меню Debug. Пользователю следует выбрать один из вариантов отладки - программная отладка (**Simulator**) или аппаратная отладка (**Emulator**). После выбора одного из вариантов, программа переходит в режим отладки и пункты меню Debug становятся доступными.

В **процессе отладки** возможно осуществлять следующие действия:
- запускать программу на исполнение до следующей точки останова (или до точки начала/конца блока профилирования) (**Run**).
- запускать программу на исполнение текущей строки инструкций (**Step**).
- приостанавливать программу (**Stop**).
- подключать к симулятору внешнее устройство (Connect Device).

4.10.2. Отладка проекта на симуляторе

В режиме программной отладки проекта, проект загружается и исполняется в **симуляторе** (специальной программе для имитации функций ИМС "МУЛЬТИКОР" на персональном компьютере).

В **процессе программной отладки** возможно осуществлять следующие действия:
- запускать программу на исполнение до следующей точки останова (или до точки начала/конца блока профилирования) (**Run**).
- запускать программу на исполнение текущей строки инструкций (**Step**).
- приостанавливать программу (**Stop**).
- подключать к симулятору внешнее устройство (Connect Device).

Для облегчения **отладки проекта**, пользователю доступны следующие окна наблюдения за исполнением программы:
- окно дизассемблера;
- окно точек наблюдения;
- окно локальных переменных;
- окно стека вызовов;
- окно профилирования;
- окно системных регистров;
- окно регистров RISC-ядра;
- окно регистров DSP-ядра;
- окно памяти;
- окно кэша программ;
- окно TLB;
- окно видеотерминала.

Запуск симулятора осуществляется выбором пункта **Start** (меню Debug). После выбора варианта **Simulator**, программа переходит в режим программной отладки и пункты меню Debug становятся доступными.

4.10.3. Отладка проекта на эмуляторе

В режиме аппаратной отладки проекта, проект загружается и исполняется на плате **эмулятора**.

Запуск отладки в режиме эмулятора осуществляется выбором пункта **Start** (меню Debug). После выбора варианта **Emulator**, программа переходит в режим аппаратной отладки и пункты меню Debug становятся доступными.

Процесс аппаратной отладки в целом проходит так же, как процесс программной отладки, за исключением следующих ограничений:

Ограничение 1: в режиме аппаратной отладки (**Emulator**) в программе DSP-ядра допустимо вводить только одну точку останова.

Ограничение 2: в режиме аппаратной отладки (**Emulator**) просмотр содержимого некоторых регистров средствами MCS (например, в окне System) может привести к некорректному исполнению программы. Это связано с тем, что чтение некоторых регистров влечет сброс битов состояния. Например, чтение приемного буфера линкового порта приведет к сбросу поля LSTAT в соответствующем регистре **LCSR**.

Ограничение 3: в режиме аппаратной отладки диалог подключения внешнего устройства Connect Device не доступен.

4.10.4. Ввод и удаление точек останова

Точки останова используются в режиме отладки проекта следующим образом: при выборе пункта **Run** меню Debug программа исполняется до следующей **точки останова**, после чего переходит в состояние **BreakPoint**. В состоянии **BreakPoint** пользователь может снять показания со всех **окон наблюдения** за отладкой проекта, после чего снова запустить программу на исполнение командами **Run** или **Step**.

Чтобы добавить/удалить **точку останова** нужно:
а) выбрать пункт **Toggle BreakPoint** меню Edit или в выпадающем меню окна редактора (окна дизассемблера), или

б) щелкнуть мышью на кнопке  инструментальной панели.

Точка останова устанавливается в *активном* окне редактора кода на той строке, где находится курсор.

Точки останова обозначены символами  . Символами  обозначены точки, в которых останов не может быть осуществлен. На поле подшивки окна редактора кода строки, в которых возможны точки останова, обозначены символом  .

Не допускается ставить точку останова на первой исполняемой инструкции программы DSP или RISC.

Установленные точки останова сохраняются в проекте и работают при последующих запусках проекта.

Чтобы просмотреть и отредактировать список установленных **точек останова**, нужно выбрать пункт **BreakPoints** меню Edit. Изменения проводятся в появившемся диалоге редактирования точек останова.

4.10.5. Блоки профилирования

Блоки профилирования служат для получения параметров исполнения частей программы - таких параметров, как Количество тактов и Использование ЦПУ. Чтобы получить эти параметры, необходимо сделать нужную часть программы **блоком профилирования** и отслеживать изменения параметров в режиме отладки в окне профилирования.

Чтобы открыть окно профилирования, необходимо выбрать пункт **Profiler** меню View.

Блоки профилирования действуют так же, как и точки останова: при прохождении программы через точку начала или конца **блока профилирования**, программа останавливается и переходит в режим **BreakPoint**. В состоянии **BreakPoint** пользователь может снять показания со всех **окон наблюдения** за отладкой проекта, после чего снова запустить программу на исполнение командами **Run** или **Step**.

Чтобы задать **блок профилирования** необходимо осуществить следующие действия:

1. В окне профилирования выделить номер блока, который будет задан (всего можно задать до восьми блоков);
2. В окне редактора кода щелкнуть правой кнопкой мыши на строке, с которой должен начинаться блок;
3. В появившемся выпадающем меню выбрать пункт **Set Profile Range Start**. Рядом с номером строки появится символ с номером **блока профилирования**;
4. В окне редактора кода щелкнуть правой кнопкой мыши на строке, которой должен оканчиваться блок;
5. В появившемся выпадающем меню выбрать пункт **Set Profile Range End**. Рядом с номером строки появится символ с номером **блока профилирования**.

Теперь при переходе в режим отладки проекта можно будет получить параметры Количество тактов и Использование ЦПУ при исполнении вновь созданного блока профилирования.

Во время исполнения программы параметры профилирования вычисляются следующим образом:

- При остановке программы на точке останова или по нажатии кнопки останова внутри блока профилирования вычисляется только Использование ЦПУ;

- При прохождении всего блока от начала до конца, включая вложенные функции, вычисляется и Количество тактов, и Использование ЦПУ;
- При прохождении блока в режиме **Step** ни один из параметров не вычисляется.

Начало и конец **блока профилирования** на **полях подшивки** окна редактора кода обозначены символами  , причем цифра в центре символа указывает на номер блока.

4.10.6. Загрузка программы

В процессе сборки проекта формируется образ памяти для загрузки в **симулятор**.
Описание структуры загружаемой информации хранится в файле с расширением *.ldr* в директории проекта.
В зависимости от подключенного конвертора для сборки (**elconv** или **elconvb**) образы памяти хранятся в файлах с именами обьектов памяти с расширением *.txt*, или с именами секций с расширением *.bin*.

Чтобы подключить конвертор для сборки, необходимо выбрать пункт **Settings** меню Tools. Появится диалог настройки набора инструментов, в котором следует подключить конвертор. Для этого нужно нажать кнопку 

и выбрать конвертор в появившемся списке.

Выбор образа для загрузки выполняется в зависимости от содержимого файла *.ldr.*

4.11. Завершение работы с проектом

Для завершения работы с проектом следует выбрать пункт **Close Project** меню File. После этого будут закрыты все окна редактора с файлами проекта. Окно проекта будет очищено.

Если проект был изменен, система предложит сохранить изменения до завершения работы с проектом.

5. Работа с файлами

5.1. Файлы

В среде MCS поддерживаются следующие операции над **файлами** с программным кодом:

- **Создание файла** – пункт **New** меню File;
- **Открытие файла** – пункт **Open** меню File;
- **Закрытие файла** – пункт **Close** меню File;
- **Сохранение файла** – пункт **Save** меню File. После внесения изменений, перед компиляцией файл необходимо **сохранить**, выполнив данный пункт меню;
- **Сохранение файла под новым именем** – пункт **Save As** меню File;
- **Редактирование файла** – изменение содержимого файла в окне редактора кода;
- **Выделение фрагмента в файле** – выделение осуществляется перемещением курсора по окну редактора кода при нажатой левой кнопке мыши. Выделение всего текста осуществляется пунктом **Select All** меню Edit;
- **Копирование в буфер** выделенного в окне редактора кода фрагмента – пункт **Copy** меню Edit;
- **Вставка текста из буфера** в окно редактора кода – пункт **Paste** меню Edit;
- **Поиск и замена текста в файле** – пункты **Find** и **Replace** меню Edit;
- **Поиск текста в файлах проекта** – пункт **Search** меню Project;
- **Вызов файла по ссылке** в окне поиска – после выполнения пункта **Search** в меню Project появится окно поиска с результатами. Щелчок по ссылке делает **активным** окно редактора, содержащее нужный файл;
- **Вызов файла по ссылке** в окне сообщений об ошибке – после выполнения компиляции файла или сборки проекта появится окно сообщений с результатами компиляции(сборки). Щелчок по ссылке делает **активным** окно редактора кода, содержащее файл с ошибкой, на которую ссылается сообщение;
- Компиляция файла – пункт **Compile File** меню Project;
- Удаление файла из проекта – пункт **Remove** меню Project.

Все операции редактирования проводятся над файлом, содержащимся в **активном** окне редактора.

5.2. Компиляция файла

Для того, чтобы **откомпилировать файл**, открытый в **активном** окне редактора кода, необходимо выбрать пункт **Compile File** меню Project.

Для компиляции файла вызывается соответствующий типу этого файла **компилятор**:

- для компилирования текстов на ассемблере DSP – компилятор **elas**;
- для компилирования текстов на ассемблере RISC и языке C – компилятор **mgcc**.

Сообщения о результатах **компиляции** выдаются в окно сообщений.

Подробнее о компиляторах файлов RISC и DSP см. книги "**Инструменты RISC**" и "**Инструменты DSP**".

6. Работа с внешними устройствами

Среда MC Studio поставляется с несколькими готовыми внешними устройствами, а именно:

- внешними устройствами библиотеки Devcore.dll;
- устройствами заглушки портов ввода-вывода.

Процесс подключения внешнего устройства к симулятору описан на странице "Диалог подключения внешнего устройства".

6.1. Устройства библиотеки Devcore

Библиотека **devcore.dll** расположена в директории установки MC Studio. Библиотека включает в себя три устройства:

- устройство чтения ReadDevice32;
- устройство записи WriteDevice32;
- устройство чтения и записи BidirectionalDevice32.

Все эти устройства для чтения/записи данных используют файлы. Формат файлов описан на странице "Формат файлов Devcore".

6.1.1. ReadDevice32

Устройство **ReadDevice32** предназначено для чтения данных с подключенного порта. Подключение осуществляется к портам *Lport0*, *Lport1*, *Lport2*, *Lport3*, *Sport0*, *Sport1* и *UART*.

При подключении устройства необходимо указать файл для вывода принятых данных (*Output File*). В режиме отладки, устройство **ReadDevice32** примет передаваемые выбранным портом данные и выведет их в указанный файл. Формат этого файла описан здесь.

Процесс подключения внешнего устройства к симулятору описан на странице "Диалог подключения внешнего устройства".

6.1.2. WriteDevice32

Устройство **WriteDevice32** предназначено для отправки данных в подключенный порт. Подключение осуществляется к портам *Lport0*, *Lport1*, *Lport2*, *Lport3*, *Sport0*, *Sport1* и *UART*.

При подключении устройства необходимо указать имя файла с данными, подлежащими отправке (*Input File*), а также частоту передачи данных устройства (*Frequency*). Формат файла, содержащего входные данные устройства **WriteDevice32**, описан на странице "Формат файлов Devcore".

В режиме отладки устройство **WriteDevice32** будет отсылать данные из файла *Input File* в подключенный порт с частотой *Frequency*. Например, если *Frequency*=100, то каждые 100 тактов симулятора в порт будет отсылаться одно значение.

Устройство начинает передачу данных *сразу* после подключения.

Следует учитывать, что устройство **WriteDevice32** не осуществляет приема данных. То есть все данные, поступающие в буфер выхода подключенного порта, при передаче будут потеряны (для порта UART).

Процесс подключения внешнего устройства к симулятору описан на странице "Диалог подключения внешнего устройства".

6.1.3. BidirectionalDevice32

Устройство **BidirectionalDevice32** осуществляет и чтение, и запись данных. Подключение устройства осуществляется к портам *Lport0*, *Lport1*, *Lport2*, *Lport3*, *Sport0*, *Sport1* и *UART*.

При подключении устройства необходимо указать имя файла с данными, подлежащими отправке (*Input File*), имя файла для вывода принятых данных (*Output File*), а также частоту передачи данных устройства (*Frequency*). Формат файла, содержащего входные данные устройства **BidirectionalDevice32**, описан на странице "Формат файлов Devcore".

В режиме отладки устройство **BidirectionalDevice32** будет отсылать данные из файла *Input File* в подключенный порт с частотой *Frequency*. Например, если *Frequency*=100, то каждые 100 тактов симулятора в порт будет отсылаться одно значение.

Данные, отправляемые портом, будут приняты и выведены в файл *Output File*.

Устройство начинает передачу данных *сразу* после подключения.

Процесс подключения внешнего устройства к симулятору описан на странице "Диалог подключения внешнего устройства".

6.1.4. Формат файлов Devcore

Устройства библиотеки **Devcore** используют для приема/отправки данных файлы. Формат этих файлов таков:

```
00000001
00000002
...
```

, то есть на одной строке размещается одно 32-разрядное число в шестнадцатеричном представлении, записанное символами ASCII-кода.

При этом, если порт передает и принимает меньше 32-х бит данных, незначащие биты заполняются нулями.

6.2. Устройства заглушки портов ввода-вывода

Устройства заглушки портов ввода-вывода применяются для замыкания внешних выводов портов на себя. То есть, при подключении устройства заглушки к порту, все данные, поступившие в выходной буфер, будут переданы во входной буфер порта.

Устройства заглушки портов ввода-вывода встроены в симулятор MultiCore и не требуют загрузки библиотек DLL.

Чтобы подключить **устройство заглушки порта ввода-вывода** необходимо:

- в списке доступных портов выбрать порт для заглушки;
- в качестве подключаемого устройства (в селекторе доступных устройств) указать имя выбранного порта.

Подробнее процесс подключения внешнего устройства к симулятору описан на странице "Диалог подключения внешнего устройства".

7. Запуск MC Studio

7.1. Запуск и завершение

Запуск среды MultiCore Studio осуществляется двойным щелчком по пиктограмме MCS на рабочем столе или щелчком по пиктограмме MCS в меню *Пуск -> Программы ->MC Studio->MC Studio*.

Завершение работы с MultiCore Studio осуществляется выбором пункта **Exit** меню File.

7.2. Инсталляция

Инсталляция среды MultiCore Studio осуществляется запуском файла *setup.exe*. Для установки MCS требуется **не менее 100 Мб** свободного пространства на жестком диске. Инсталлировать MCS рекомендуется в отдельную директорию. При инсталляции на рабочем столе автоматически создается пиктограмма MCS, а в меню Пуск->Программы – раздел с инструментами MCS и файлами справки.

8. Предметный указатель