KTechlab Handbook

David Saxton & Daniel Clarke

Оглавление

Общее

KTechlab это IDE для микроконтроллеров и электроники.

Часть 1 Обзор

1.1 Введение

KTechlab это IDE (Integrated Development Environment — интегрированная среда разработки) для электронных цепей и микроконтроллеров. Она может выполнять симуляцию разнообразных компонент (логических, интегральных, линейных, нелинейных и реактивных), симуляцию и отладку PIC микроконтроллеров с помощью gpsim, и приходит с ее собственными связанными и взаимодополняющими языками высокого уровня: FlowCode и Microbe.

KTechlab была разработана так, чтобы быть легкой в использовании и насколько возможно ненавязчивой; все компоненты и Flow-элементы имеют контекстно зависимую подсказку, а симуляция электроники столь же проста, как перетаскивание компонент в рабочую область и создание разъемов, автоматически соединяющих свои выводы. FlowCode позволяет пользователям, не знакомым с PIC контроллерами, сразу создать их собственные программы, тогда как симуляция электроники позволяет пошаговый проход по ассемблерной программе PIC внутри схемы.

1.2 Документы

Чтобы начать работать с KTechlab, вам нужно создать новый документ, тип которого определится вашей задачей:

- FlowCode документ – сконструирует PIC программу с помощью flowcharting (блок-схема).
- Circuit документ – симулирует электрические схемы и микроконтроллеры.
- Microbe документ – язык высокого уровня для PIC контроллеров, также используемый FlowCode для генерации ассемблера.
- Assembly документ – начало написания PIC ассемблерной программы.

KTechlab использует Document-View модель, в которой логика документа полностью отделена от открытого вида документа. Это позволяет иметь несколько видов одного и того же файла.

При создании нового документ его вид создается на отдельной вкладке. Каждая из них может поддерживать несколько видов, сложенных в любой произвольный шаблон. Это позволяет, например, симулировать PIC программу в схеме, пошагово проходя ее в ассемблерном

документе на той же вкладке.

Содержимое вкладок может дублироваться перетаскиванием вкладки в пустую область вкладок. Они могут вставляться в существующие вкладки перетаскиванием их на эту вкладку.

Детальные инструкции по всем выше приведенным документам могут быть найдены в соответствующих разделах.

1.3 Черчение

В документах Circuit и FlowCode есть несколько инструментов рисования, включая инструмент создания текста. Они доступны после щелчка по иконке кисточки на инструментальной панели. Для черчения подведите мышку либо к фигуре, либо к линии, соответственно нужному инструменту рисования.

Когда выбор сделан, рисунок может масштабироваться вручную растягиванием. Если при этом удерживать клавишу shift, то осуществляется привязка к подсвеченной сетке. Каждый инструмент имеет базовые опции доступные через инструментальную панель, такие как цвета. Есть также множество расширенных опций, таких как стиль линии или шрифта, в разделах боковой панели.

Часть 2 PIC программы

2.1 Манипуляции с программой

Когда вы создаете документ FlowCode или Text, вы заметите на инструментальной панели иконку с изображением ракеты. С помощью меню вы можете манипулировать с вашей PIC программой, преобразовывая ее к разным формам:

- Convert to Microbe – используется только для документов FlowCode. Пояснения можно найти дальше в части 4.
- Convert to Assembly – может использоваться в четырех случаях. Когда открыт документ FlowCode, это выведет FlowCode, как ассемблерные инструкции. Когда открыт документ Microbe, это вызовет программу microbe, распространяемую с KTechlab, для компиляции программы. Аналогично, если открыта программа на C, это приведет к попытке компиляции ее с помощью SDCC. Когда открыт текстовый документ, содержащий PIC hex (формат для программирования), будет вызвана программа gpdasm для дизассемблирования hex.
- Convert to Hex – может также использоваться в четырех случаях. Как и Convert to Assembly, использоваться с FlowCode, Microbe и C документами. Также, когда открыт ассемблерный документ, можно ассемблировать с помощью gpasm.
- Upload to PIC – этим ассемблируется редактируемая в настоящий момент PIC программа и загружается с помощью программатора, который выбрал пользователь.

Ни одна из этих акций не требует, чтобы документ был сохранен, что очень полезно для тех моментов, когда нужно быстрое программирование. Для целей вне PIC программирования вызывается диалог вывода щелчком по одному из этих разделов и возможен либо вывод результатов (всегда текст во всех вышеприведенных случаях) для обновления документа, либо вывод в файл. Если вывод сохраняется в файл, также предоставляются опции для загрузки файла после создания, и добавления вновь созданного файла к открытому проекту (если он открыт).

Заметьте, что вы можете сделать так, что KTechlab всегда будет использовать тотже вид отображения содержимого вывода, если вы выберете соответствующую опцию в General Settings.

2.2 Загрузка

KTechlab использует программаторы сторонних производителей для загрузки программ в PIC контроллеры. Предопределено множество общих программаторов. Другие же могут добавляться через диалог Settings. Загляните на web-сайт KTechlab, если нужна дополнительная информация.

Список портов получен сканированием последовательных и параллельных портов, из которых можно читать и в которые можно записывать. Последовательные порты:

- /dev/ttyS[0..7]
- /dev/tts/[0..7]
- /dev/ttyUSB[0..7]
- /dev/usb/tts/[0..7]

Параллельные порты:

- /dev/usb/parport[0..7]
- /dev/usb/parports/[0..7]

Часть 3 Схемы

3.1 Размещение компонент

Слева вы найдете вкладку компонент.

Перетаскивание компонента с боковой панели в цепь разместит его под курсором мышки. В качестве альтернативы вы можете двойным щелчком по изображению компонента на левой панели повторно добавлять его в схему. В этом режиме копия выбранного компонента будет повторно размещаться по левому щелчку мышки до тех пора, пока не будет нажата клавиша Escape, либо правая клавиша мышки.

Для перемещения компонента щелкните по нему левой клавишей мышки и перетащите его. Вы увидите, что он привяжется к подсвеченной сетке. Если при перетаскивании компонента вы выйдите за правый или нижний край рабочей области, она, приспосабливаясь, изменит свой размер.

Все компоненты имеют представление об ориентации 0, 90, 180 или 270 градусов. Те, что не симметричны относительно осей, могут также отражаться. Для поворота выбранных компонент либо щелкните правой клавишей мышки и выберите нужное из всплывающего меню, или щелкните по кнопке вращения на инструментальной панели. Последние могут быть также доступны с помощью клавиш "[" и "]" (хорошо известных пользователям Inkscape). Боковая панель (справа) предоставляет хороший метод задания ориентации с помощью предварительного просмотра компонент. Отражение компонент тоже доступно только через разделы боковой панели.

3.2 Соединение компонент

Есть два режима создания соединений (wires): автоматический (automatic) и ручной (manual).

Эти режимы выбираются через выпадающее меню "Connection Routing Mode" на инструментальной панели. Поэкспериментируйте с обоими, автоматическая трассировка часто лучше для маленьких схем, тогда как более сложные могут потребовать ручного соединения.

В автоматическом режиме создаются соединения либо перемещением от вывода компонента, либо от существующего соединения с последующим отпусканием клавиши мышки на выбранном выводе или соединении. Вы увидите пунктирную линию, которая становится оранжевой, когда вы отпустите клавишу. Если линия, которую вы рисуете, черная, это означает, что либо под курсором мышки ничего нет, либо, что вы соединили вместе два элемента, которые уже соединены. При создании блок-схем критерий правильного соединения более сложный, но к этому мы вернемся позже.

Лучший способ почувствовать ручное соединение это поэкспериментировать с ним. Щелкните по начальному выводу или соединению, а затем расширяйте протосоединение движением мышки от этого места. Для создания угла щелкните левой клавишей. Для прекращения вычерчивания соединения либо нажмите на клавиатуре клавишу Escape, либо щелкните правой клавишей мышки.

KTechlab пытается наилучшим образом разместить ваше соединение. Однако, если перетаскивание компонента приводит к тому, что конечные точки соединения перемещаются относительно друг друга, KTechlab будет форсировать перерисовку соединения, используя автоматический режим (auto-routing). Перед перемещением для нового соединения они по щелчку становятся серыми.

Для удаления существующего соединения выберите его с помощью прорисовки маленького прямоугольника выбора на части соединения, а затем нажмите клавишу удаления (delete).

3.3 Атрибуты компонент

Большинство компонент будут иметь редактируемые атрибуты, такие как сопротивление для резисторов. По умолчанию вы можете редактировать простые атрибуты с помощью инструментальной панели, когда выбрана группа однотипных компонент. Если ваш выбор содержит смесь разных типов компонент (как, например, резисторы и конденсаторы), тогда атрибуты не будут отображены для редактирования.

Некоторые компоненты имеют более развитые атрибуты, которые не доступны через инструментальную панель. Они могут быть найдены в разделах боковой панели справа. Так, например, диод имеет несколько поведенческих характеристик, которые вы можете редактировать здесь.

Если ваш выбор компонент имеет разные значения атрибутов (как разное сопротивление для резисторов), образец на боковой панели будет иметь серый вывод для несогласованных атрибутов. Вы можете включить это щелчком по клавише "Merge properties".

Клавиша "Defaults" сбросит атрибуты компонента к значению, которое он имел при создании.

Есть один тип атрибутов, который не может быть редактируемым ни с помощью инструментальной панели, ни с помощью боковой панели multilinetext (многострочный текст). Двойной щелчок по нему откроет диалоговое окно, где может быть введен текст.

3.4 Симуляция

По умолчанию симуляция будет запущена, когда вы создадите новую цепь. Состояние симуляции отображается в нижнем правом углу вида цепи, и может быть изменено через меню Tools. Вначале небольшое объяснение, как работает симуляция. Это позволит вам получить

более полное представление о процессе.

Когда создается или модифицируется цепь, затронутые области разбиваются на группы выводов и соединений, которые могут рассматриваться независимо. Каждая группа затем симулируется, как отдельная сущность (хотя и взаимодействует через компоненты), с симуляцией предоставляется зависимость от сложности групп. Сложные группы, такие как те, что содержат нелинейные компоненты подобные LED (светодиоды), замедляют симуляцию. Группы содержащие только логические выводы, на которых контролируется только значение этого вывода, самые быстрые в симуляции.

Результаты симуляции проходят через несколько графических средств.

Выводы на компонентах будут отображать окошки с напряжением. Они раскрашены оранжевые для положительных значений, синие для отрицательных. Их длина зависит от уровня напряжения и величины тока, протекающего через вывод. Отображение можно выключить на вкладке WorkArea диалога Configuration.

Установка мышки над выводом или соединением отобразит маленький указатель напряжения и тока в этой точке цепи. Несколько компонент предоставляют также графическую анимацию, например, LED и вольтметры или амперметры.

Наконец, есть осциллограф, обсуждаемый в следующем разделе.

3.5 Осциллограф

Осциллограф может записывать логические данные, данные напряжения и тока. Логический пробник оптимален для запоминания булевых значений, так что должен использоваться вместо пробника напряжения при логических измерениях.

Для сбора данных создайте новый компонент пробника и установите его в надлежащую точку схемы. Вы увидите вывод немедленно, нарисованный на экране осциллографа. Добавление большего количества пробников приведет к наложению выводов одного на другой. Вы можете перетаскивать их с помощью стрелок слева от экрана осциллографа, и менять их цвет через атрибуты пробников.

Для пробников напряжения и тока диапазон входных значений может быть задан с помощью "Item Editor" на правой боковой панели.

Масштабирование управляется ползунком. Шкала логарифмическая; для каждых нескольких пикселей, на которые смещается ползунок, zoom-фактор будет умножаться на константу. KTechlab симулирует логику с максимальной точностью в 1 микросекунду, а на максимальном zoom-уровне, одна микросекунда представлена 8 пикселями.

Когда полоса прокрутки перетаскивается к концу, она остается там, пока записываются новые данные. Иначе позиция полосы прокрутки остается фиксированной во времени. Экран осциллографа может также перемещаться вперед и назад после щелчка левой клавиши мышки и перетаскивания. Благодаря ограничениям, лежащим в основе системы виджетов, прокрутка будет очень «зернистой» при максимальном зуммировании.

Щелчок правой клавиши мышки по экрану осциллографа отрывает меню, где вы можете управлять периодом времени обновления окна осциллографа. Что позволяет либо сгладить изображение или ослабить использование CPU.

3.6 Подсхемы

Подсхемы открывают возможность повторного и аккуратного использования цепей, когда вы интересуетесь только внешними соединениями схемы. Подсхемы создаются как IC (интегральные схемы) с выводами, работающими во взаимодействии с внутренней схемой.

Первое, используемая схема как шаблон для создания подсхемы, в основном должна быть сконструирована. Точки соединения определяются через "External Connection"
компоненты. Они должны быть соединены и позиционированы там, где вы предполагаете иметь выводы подсхемы IC.

Далее, выбранную группу компонент и внешних соединителей следует поместить в подсхему, выбрать "Create Subcircuit" из меню, вызванного правым щелчком мышки. Вам будет предложено ввести имя для подсхемы. После ее создания имя будет подменю селектора компонент под выбором Subcircuits. Она может обрабатываться, как обычный компонент с дополнительной опцией удаления ее правым щелчком мышки по разделу и выбором Remove.

Часть 4 FlowCode

4.1 Введение

FlowCode позволяет очень быстрое и легкое конструирование PIC программ. После того, как пользователь создал блок-схему из доступных программных частей, KTechlab может конвертировать блок-схему в несколько форматов. Для вывода в hex формат, например, следующая цепочка преобразований может иметь место:

1. FlowCode конвертируется в Microbe, язык высокого уровня, компилятор которого распространяется с KTechlab.
2. Исполняемый microbe затем компилирует Microbe файл в PIC ассемблер.
3. И, наконец, gpasm берет файл PIC ассемблера и выдает hex для программы.

Конечно, если у вас не установлены gputils, с которыми распространяется gpasm, тогда последний шаг не может быть выполнен.

4.2 Создание программы

Каждая программма FlowCode нуждается в уникальной точке старта это место, где ваша программа будет запускаться с запуском PIC. Для определения этой точки, откройте боковую панель FlowParts слева и перетащите Start part. KTechlab только позволит вам использовать одно из следующего.

Вы затем сможете сконструировать вашу программу, используя предопределенные элементы слева, или вставляя собственный код (на ассемблере или в формате Microbe) с помощью Embed part. Ход программы контролируется через связь с
FlowParts. Раздел 3.2 более детально описывает соединения.

FlowCode вводит ограничения в дополнительно к тем, что в Circuits, на то, что может быть соединено. Например, каждая FlowPart может иметь только одно выходное соединение. Дополнительные ограничения описаны в разделе 4.4.

4.3 PIC установки

Когда вы создаете новый FlowCode документ, вы заметите картинку PIC-контроллера, используемую вами, в верхнем левом углу рабочей области. Так представлены начальные установки PIC-контроллера.

Каждый вывод показанный на картинке PIC-контроллера отображает начальный тип вывода (ввод-input или вывод-output) и его начальное состояние (высокое-high или низкое-low).Вы можете менять их, перетаскивая вывод для установки типа и щелкая по нему для переключения состояния.

Диалог Settings, вызываемый щелчком по клавише Settings, также позволяет вам редактировать начальные тип вывода и состояние, в этом случае через редактирование двоичного значения, записанного в регистры PORT и TRIS. Аналогично установкам выводов диалог позволяет редактировать начальные значения переменных в PIC программе.

Внизу есть список в данный момент определенных карт выводов (pin maps), а также кнопки для манипуляции ими. Карта выводов используется для спецификации того, как семи-сегментный индикатор или клавиатура подключены к PIC-контроллеру. Для использования SevenSegment или Keypad FlowCode элементов вам необходимо вначале определить здесь карту выводов.

4.4 Вложенный FlowCode

Множество FlowParts, таких как подпрограммы и циклы (subroutines и loops), могут содержать их собственный код. После создания такого контейнера FlowParts могут добавляться либо перетаскиванием, либо сбросом их в контейнер. Контейнер будет подсвечен для индикации, что он становится новым родителем FlowPart.

Контейнер берет на себя ответственность за FlowParts, вложенные внутри. Если клавиша expand (расширить) отжата, все содержащиеся FlowParts будут скрыты, и наоборот, содержимое будет показано, если клавиша expand нажата вновь. Соединенения не могут быть сделаны между FlowParts в разных контейнерах, и содержимое контейнера будет перемещаться вместе с контейнером.

Часть 5 Microbe

5.1 Введение и основной синтаксис

Microbe компилирует программу, написанную на привычном языке для PIC, как вспомогательная программа для KTechlab. Синтаксис был разработан для согласования с FlowCode программами. Синтаксис для запуска microbe из командной строки такой:

```
microbe [options] [input.microbe] [output.asm]
```

где опции:

--show-source – добавить каждую строку исходного кода Microbe, как комментарий в ассемблерный вывод перед собственно ассемблерной инструкцией на этой строке.
--no-optimize – предотвращает оптимизацию инструкций, генерируемых из исходного кода. Оптимизация обычно безопасна, и таким образом опция в основном используется для отладки.

Входной файл .microbe должен идентифицировать целевой PIC с помощью вставки имени PIC в верхней части .microbe файла, то есть, имя PIC16F84 это "P16F84".

Пример 5.1 Простая полная программа Microbe

```
P16F84

      a=0
      repeat
      {
            PORTA = a
            a=a+1
      }
      until a== 5

end
```

5.1.1 Соглашения об именах

Следующие правила относятся к именам переменных и меток:

• Они могут содержать только буквенно-цифровые символы [a..z][A..Z][0..9] и подчеркивание "_".
• Они зависимы от регистра (case-sensitive).
• Они не могут начинаться с цифры.
• Они не должны начинаться с "__" (двойное подчеркивание), поскольку это зарезервировано для использования компилятором.

5.1.2 Соглашения о скобках

Фигурные скобки, {}, показывают начало и конец блока кода. Они могут появиться в любом месте до начала и после завершения блока кода. Примеры приемлемых блоков кода:

```
statement1 {
some code

}

statement2 {
other code }

statement3

{
other code

}

statement5 {
code block

} statement6
```

5.1.3 Комментарии

Комментарии похожи на те, что в C. // комментирует все следующее в строке. /* и */ обозначает многострочный комментарий.

```
// Это комментарий

x=2

/* А здесь

многострочный комментарий */
```

5.1.4 Структура программы

PIC id (идентификатор) должен быть вставлен вверху программы. Конец основной прогрммы (main program) отмечается с помощью "end". Подпрограмма должна помещаться после "end".

5.1.5 Подпрограммы

Подпрограмма может быть вызвана из любого места в коде. Синтаксис:

```
sub SubName

{
// Code...
```

```
}
```

Подпрограмма вызвана с помощью "call SubName".

5.2 Microbe ссылки языка

5.2.1 if

Условный переход. Синтаксис:

```
if [expression] then [statement]
```

или

```
if [expression] then
{
      [statement block]
}
```

Аналогично для else:

```
else [statement]
```

или

```
else

{
      [statement block]
}
```

Пример 5.2 if

```
if porta.0 is high then
{
      delay 200
}
else
{
      delay 300
}
```

5.2.2 alias (псевдоним)

Подменяет одну строку на другую. Синтаксис:

```
      alias [from] [to]
```

5.2.3 repeat

Выполняет заданный блок многократно пока выражение не станет истинным. Становление

выражения происходит после выполнения блока, так что заданный блок будет выполнен хотя бы один раз. Синтаксис:

```
repeat
{
      [statement block]
}
until [expression]
```

5.2.4 while

Аналогично repeat, этим многократно выполняется заданный блок. Однако выражение оценивается до выполнения, а не после. Так что, если выражение оценивается как ложное на первом проходе, блок не будет выполняться. Синтаксис:

```
      while [expression]
      {
            [statement block]
      }
```

5.2.5 goto

Это приводит к тому, что продолжение выполнения кода будет со следующего после заданной метки кода. Goto синтаксис:

```
      goto [labelname]
```

Синтаксис метки (Label):

```
      labelname:
```

Часто считается хорошей практикой программирования избегать использования goto. Использование управления и подпрограмм приводит к лучшей «читабельности» программы.

Пример 5.3 goto

```
goto MyLabel
...
[MyLabel]:
// Выполнение кода продолжится с этого места
```

5.2.6 call

Вызывает подпрограмму. Синтаксис:

```
      call [SubName]
```

где SubName имя вызываемой подпрограммы.

5.2.7 delay

Приводит к остановке выполнения кода на заданный период времени. Интервал в миллисекундах. Синтаксис:

```
delay [interval]
```

ПРИМЕЧАНИЕ
В настоящее время Microbe подразумевает, что PIC работает на частоте 4Mhz, то есть, каждая инструкция занимает 1 микросекунду. Если это не так, следует пропорционально изменить интервал.

5.2.8 sevenseg

Используется для определения карты выводов для (с общим катодом) семи-сегментного индикатора, соединенного с PIC. Синтаксис:

```
sevenseg [name] [a] [b] [c] [d] [e] [f] [g]
```

где [a]...[g] выводы PIC, к которым подключены соответствующие сегменты семи-сегментного дисплея. Выводы могут записываться либо, как PORTX.N или
RXN.

Для отображения цифры на семи сегментах, карта выводов обрабатывается, как переменная только для записи.

Пример 5.4 Определение и вывод на семи-сегментный индикатор

```
sevenseg seg1 RB0 RB1 RB2 RB3 RB4 RB5 RB6
seg1=x + 2
```

5.2.9 keypad

Используется для определения карты выводов для клавиатуры, соединенной с PIC. Синтаксис:

```
keypad [name] [row 1] ... [row 4] [column 1] ... [column n]
```

где [row 1] ... [row 4] и [column 1] ... [column n] это выводы PIC, к которым соответствующие выводы строк и столбцов клавиатуры присоединены (в настоящее время число строк нельзя изменить). Посмотрите раздел 5.2.8 (выше), там больше информации по картам выводов.

Столбцы клавиатуры должны быть заземлены через резистор 100 кОм. Выводы строк должны быть сконфигурированы, как выходы, а выводы столбцов, как входы. Когда клавиатура определена, она обрабатывается как переменная только для чтения.

Пример 5.5 Определение и чтение с клавиатуры

```
keypad keypad1 RB0 RB1 RB2 RB3 RB4 RB5 RB6
x = keypad1
```

По умолчанию значения, возвращаемые для клавиатуры, это:

- Значение цифры, если это цифровая клавиша (от 1 к 3 строке сверху; шестнадцатеричные от A до D вниз к четвертому столбцу, и дальше для каждого дополнительного столбца).
- 253 для клавиши в строке 4, столбце 1.
- 254 для клавиши в строке 4, столбце 3.

Эти значения могут быть переопределены использованием команд псевдонимов (alias), где имя клавиши в строке x, столбце y (строки и столбцы начинаются с 1) это Keypad_x_y.
Например, чтобы клавиша звездочки на клавиатуре 4x3 имела значение ноль, следующий псевдоним (alias) будет использован:

Пример 5.6 Присваивание псевдониму клавиши значения

```
alias Keypad_4_1 0
```

5.3 PIC I/O

5.3.1 Направленность порта

Направленность порта устанавливается присваиванием значения TRIS*, где * это буква порта:
Например:

Пример 5.7 Установка направленности порта

```
TRISB = b'01111001'
```

Выше устанавливаются выводы RB1, RB2 и RB7 порта PORTB на вывод, а остальные выводы порта PORTB на ввод. В этом примере b'01111001' это двоичное представление типа вывода. 1 справа представляет вывод на RB0, а 0 слева ввод на RB7.

5.3.2 Порт I/O (ввод/вывод)

Порт может обрабатываться, как переменная. Например:

Пример 5.8 Запись в порт

```
x = PORTA
```

Выше переменной x присваивается значение PORTA.

5.3.3 Вывод I/O

Каждый вывод порта достигается через префикс номера вывода за именем порта; то есть, вывод 2 (начиная с вывода 0) на PORTA получается, как PORTA.0. Синтаксис для установки состояния вывода:

```
PORTX.N = STATE
```

где STATE может быть высоким или низким. Синтаксис для проверки состояния вывода:

```
if PORTX.N is STATE then
```

Комбинируя эти примеры, мы получим:

Пример 5.9 Установка и проверка состояния вывода

```
TRISA = 0
TRISB = 255
if PORTA.3 is high then
      {
            PORTB.5 = low
      }
      else
      {
            PORTB = PORTA + 15
      }
```

5.4 Переменные

Все переменные это 8-битовые беззнаковые целые, получающие значения от 0 до 255. Microbe поддерживает типовые унарные операции (работающие на одной переменной) и бинарные операции (работающие на двух переменных), которые поддерживаются PIC-контроллером. Дополнительно Microbe также поддерживает деление и умножение.

5.4.1 Унарные операции

- rotateleft x – сдвигает переменную x влево через перенос.
- rotateright x – сдвигает переменную x вправо через перенос.
- increment x – увеличивает (инкремент) переменную x. Если x имеет значение 255, тогда x возвращается вновь к 0.
- decrement x – уменьшает (декремент) переменную x. Если x имеет значение 0, тогда x возвращается вновь к 255.

5.4.2 Арифметика

Поддерживаемые операции:

- Сложение: x+y
- Вычитание: x-y
- Умножение: x*y
- Деление: x/y
- Двоичное XOR: x XOR y
- Двоичное AND: x AND y
- Двоичное OR: x OR y

5.4.3 Сравнение

Поддерживаемые операции:

- Эквивалентность: x == y
- Не эквивалентность: x != y
- Больше, чем: x>y
- Меньше, чем: x<y
- Больше, чем или равно: x >=y
- Меньше, чем или равно: x <=y

Например:

Пример 5.10 Сравнение

```
if PORTA >= 5 then
{
...
}
```

Часть 6 Отладка

6.1 Запуск отладчика

Поддержка отладки осуществляется для ассемблера, SDCC и Microbe, когда они открыты как текстовые документы. С этого места действия управляются через меню Debug. Есть два метода запуска отладчика.

Если PIC программа уже запущена в схеме, тогда дважды щелкните по PIC компоненту, что откроет программу. Для ассемблерной PIC программы отладчик текстового документа будет привязан к PIC компоненту. В этом случае меню Debug не может остановить PIC программу, поскольку она принадлежит PIC компоненту.

Если ассемблерный файл уже открыт, тогда отладчик может быть запущен через меню Debug. После компиляции программы отладчик будет готов, а PIC программа приостановлена на первой инструкции. Заметьте, что при отладке языков высокого уровня текущая точка выполнения не будет показана, если нет строки, относящейся к первой выполняемой ассемблерной инструкции. В этом случае щелчок по next перенесет точку выполнения к первой строке в программе.

6.2 Управление отладкой

Отладчик может находится в одном из двух режимов: выполнения и пошаговом. В режиме выполнения PIC программа будет симулироваться в реальном времени. Чтобы перейти к пошаговому режиму, PIC программа должна быть приостановлена, либо щелчком по Interrupt в меню Debug, либо щелчком по клавише pause на PIC компоненте.

В пошаговом режиме зеленая стрелка в поле текстового документа показывает строку текста, которая будет выполняться (знакомо для пользователей Kdevelop). Может быть полезно включить рамку иконки через меню View (можно постоянно включить в диалоге Editor Settings).

Есть три типа шагов:

- Step into – этим выполняется текущая инструкция. Зеленая стрелка перемещается на следующую строку, которая должна выполняться.
- Step over – если следующая инструкция, которая должна выполняться, вызывается или что-то похожее, этим будет сделан шаг над "step over" вызовом, возвращаясь к пошаговому режиму, как только вызов возвращается. Другими словами, шаг над инструкцией ведет себя идентично шагу. Технически это так, начальный уровень стека записывается, а выполнение программы приостанавливается до момента, когда уровень стека возвращается к начальному состоянию.

• Step out – если текущее выполнение внутри вызова или нечто похожее, тогда этим реализуется ожидание возвращение из вызова. Похожее на шаг над, это эквивалентно ожиданию, когда уровень стека возвращается к состоянию предшествующему начальному состоянию стека, если начальное состояние больше, чем ноль.
• Breakpoints точки останова позволяют приостановить выполнение, когда PIC программа достигает заданной инструкции. Для переключения точки останова на строке, содержащей курсор, либо используйте меню Debug, либо щелкните по рамке иконки текстового документа.

Боковая панель "SymbolViewer" справа показывает значения Special Function Registers (регистры специальных функций). Чтобы найти значение переменной в General Purpose Registers (регистры общего назначения), вы можете поместить мышку над именем переменной в инструкции, которая оперирует с данным регистром. Заметьте, что выбор системы счисления в SymbolViewer также управляет тем, как отображаются значения, когда курсор мышки проходит над переменной.

Часть 7 FAQ (часто задаваемые вопросы)

Configure не может найти gpsim

Configure попытается компилировать несколько файлов, чтобы проверить наличие gpsim; проверяются версии 0.21.4, 0.21.11 и 0.21.12. Эти файлы используют часть gpsim
API, которая также используется KTechlab, и была приведена к соответствующей версии gpsim. Если компиляция этой программы обрывается, появляется предупреждение пользователю, а поддержка симуляции PIC не будет компилирована в KTechlab. Есть несколько возможных причин, почему не работает gpsim:

• gpsim >= 0.21.4 не установлен. KTechlab не поддерживает gpsim-0.21.2 или gpsim-0.20.14 (или любую раннюю версию). Последующие версии могут работать, но это не может быть проверено до их появления. Последние версии gpsim могут быть получен на web-сайте gpsim.
• Gpsim установлен, но файлы заголовков не могут быть найдены. Если gpsim установлен не в стандартные директории, вам может понадобиться обозначить местонахождение файлов заголовков с помощью "./configure--with-extra-includes=DIR". И похоже, вам понадобится указать местонахождение библиотеки с помощью опции configure "--withextra-lib", если includes не в стандартной директории.
• Gpsim установлен и заголовки где-то, где configure может найти их, но обнаруживается какая-то другая сумасшедшая glib.

Configure генерирует файл config.log, который содержит детали всего, что пошло не так. Найдите gpsim в этом файле, что поможет определить причины произошедшего. В качестве примера вот часть файла config.log, где пропущен один из файлов заголовка gpsim:

Пример 7.1 Возможный вывод config.log, где обнаружение gpsim прервано

```
configure:30956: checking for gpsim 0.21.4 availability
configure:31009: g++ -c -I/usr/include/glib-2.0 -I/usr/lib/glib-2.0/include
-DQT_THREAD_SUPPORT -O3 -march=athlon-xp -mcpu=athlon-xp -D_REENTRANT
conftest.cc >&5
conftest.cc:48:35: gpsim/gpsim_interface.h: No such file or directory
```

Если config.log не помог, пожалуйста, свяжитесь с разработчиками KTechlab и не забудьте прикрепить соответствующую часть config.log.

KTechlab «рухнул» при запуске PIC программы

Версия 0.21.11 gpsim имеет ошибку, которая может стать причиной крушения KTechlab при втором запуске PIC программы. Эта ошибка уже зафиксирована в gpsim.
Вы может либо вернуться к более ранней gpsim версии 0.21.4(прекомпилировав Ktechlab), или обновить gpsim (к моменту написания этого gpsim > version 0.21.11 не была выпущена, но CVS версия работает). Обратите внимание, что Debian gpsim-0.21.11 сборка, представленная на web-сайте KTechlab, была исправлена в отношении этой ошибки. Если вы уже используете сборку Debian или не используете версию 0.21.11, тогда, пожалуйста, свяжитесь с разработчиками KTechlab.

High Level Language (HLL – язык высокого уровня) отладчик не работает

Версия 0.21.11 gpsim имеет ошибку, препятствующую работе HLL отладчика. Как и в случае, описанном выше, вы можете либо использовать предыдущую версию, либо обновить gpsim. Сборка Debian gpsim-0.21.11, представленная на web-сайте KTechlab была пропатчена для устранения ошибки. Если вы уже используете сборку Debian или не используете версию 0.21.11, тогда, пожалуйста, свяжитесь с разработчиками KTechlab.

KTechlab перегружает CPU

Есть несколько возможных причин. Симуляция цепей, содержащих и реактивные и нелинейные компоненты (такие как конденсаторы и транзисторы), требует много времени CPU для выполнения симуляции. Вы можете приостановить и возобновить симуляцию с помощью меню Tools. Черчение в рабочей области (в частности перерисовка многочисленных быстро обновляющихся напряжений на выводах) также интенсивно использует CPU. Вы можете уменьшить время обновления или выключить отображение напряжения с помощью диалога Settings. Время обновления Oscilloscope можно также уменьшить правым щелчком мышки на его дисплее. Заметьте, что следующий выпуск KTechlab будет значительно быстрее в части отображения и рабочей области, и реактивных и нелинейных компонент.