В.Н. Гололобов

KTECHLAB, НОВЫЕ ПРИКЛЮЧЕНИЯ С ВЕРСИЕЙ 0.50.0

Москва - 2020

Оглавление

Предисловие

Все началось со сбоя в самый ответственный момент: при загрузке компьютера экран монитора оставался чёрным. Моему компьютеру лет, наверное, десять; когда-то он был куплен вместе с операционной системой, тогда новомодной, Vista. Я не знаю, была ли версия этой ОС 64-битовая или только 32-битовая. Последняя оказалась на моём компьютере.

Когда Microsoft'ом было предложено обновиться до Windows 10, я не стал противиться, в итоге на компьютере долгое время работала 32-битовая версия Win10. Мало-помалу компьютер стал долго загружаться, программы стали «слишком задумчивы», а я становился всё раздражительнее. Вдобавок, вас, думаю не удивит, в последнее время ряд программ стали появляться только в 64-битовом исполнении.

На материнской плате компьютера, что я выяснил при сбое, когда пришлось разобрать компьютер, выясняя причины сбоя (причину я так и не выяснил, только догадки), на материнской плате четыре гнезда для модулей памяти. А памяти было 4 Гб на двух модулях DDR2. «Ничтоже сумняшеся» я решил добавить ещё два модуля памяти, решив, что 8 Гб ускорят работу компьютера. Заказать необходимое на Алиэкспресс труда не составило. Но, добавив новое приобретение, я вызвал только полный отказ компьютера от работы.

Интернет – полезный справочник по борьбе с разными проблемами. Доверившись более опытным консультантам, которые советовали использовать модули памяти от одного производителя, я прикупил ещё два модуля памяти. Но к успеху это не привело, поскольку, когда я нашёл руководство пользователя к материнской плате, она была не предназначена к расширению памяти сверх 4 Гб, хотя процессор у меня 64-битовый.

Я не сторонник менять старые вещи на новые, если старые вещи прекрасно служат, но поломки с телевизорами и плитой, поломки на ровном месте, которые обошлись достаточно дорого, заставили меня задуматься, что можно предпринять с компьютером? Покупать новый компьютер имело смысл лет пять назад, но не сегодня. С другой стороны, я уже привык во многих случаях пользоваться компьютером, как при покупках на Алиэкспресс, так и в поиске ответов на возникающие вопросы.

Итогом этих размышлений стала покупка новой материнской платы и твердотельного ssd-диска, нового блока питания (прежний стал очень шуметь), установка Windows 10 в 64-битовой версии, покупка нового офисного пакета 2019, и сборка компьютера с новыми комплектующими.

Теперь компьютер загружается за несколько секунд, а текстовый процессор Word делает это почти мгновенно. Жаль только, что теперь мне нечего написать в Word, не то, что раньше. За время самоизоляции 2020 я не написал ни строчки – то ли желания не было, то ли давнее решение бросить сочинительство по причине того, что насочинял я много для радиолюбителей, но всё это про ток, напряжение, сопротивление, а более всего о компьютерных программах, за которые бывалые радиолюбители меня ругали нещадно. Ничего нового мне и сказать нечего.

Однако другого занятия, исключая обычные домашние дела, в которых я принимаю посильное участие, другого занятия я не нашёл. Вспомнив, что последние версии Windows 10 позволяют легко использовать Linux, я попытался это сделать, но столкнулся с тем, что мой процессор Intel Core 2 quad Q6600 не поддерживает всё необходимое для полной работы с WSL2. Тогда я вспомнил, что долгое время программа VirtualBox предлагала обновиться, но последняя версия есть только в 64-битовом исполнении. Теперь ничто не мешает установить программу, а в программе установить Linux. Осталось понять, зачем мне это нужно?

Я несколько раз рассказывал о программе KTechlab для Linux. Рассказывал, как дополнить программу для работы со всеми регистрами микроконтроллера PIC16F628A – выбор

микроконтроллера обусловлен в первую очередь относительной простотой и наличием руководства на русском языке – и рассказывал, что автор программы оставил её, программу подхватила группа энтузиастов. И рассказывал, как с большим трудом мне удалось скомпилировать программу из исходных кодов в дистрибутиве Fedora.

Стало интересно, что изменилось за прошедшее время.

Как выяснилось, в дистрибутиве Fedora по-прежнему несколько неудачная версия программы, не позволяющая моделировать схему с участием микроконтроллера. Но для некоторых дистрибутивов есть версия KTechlab-0.50.0. Решением стала установка openSUSE 15.2.

Первое, с чем пришлось при этом столкнуться – разрешение получилось только 800x600. То есть, при попытке изменить разрешение экрана появляются допустимые варианты, коих много, но они не работают. Решение проблемы оказалось простым (спасибо тому, кто подсказал это решение) – достаточно в настройках дисплея заменить значение по умолчанию другим, как показано на рис.1.1.

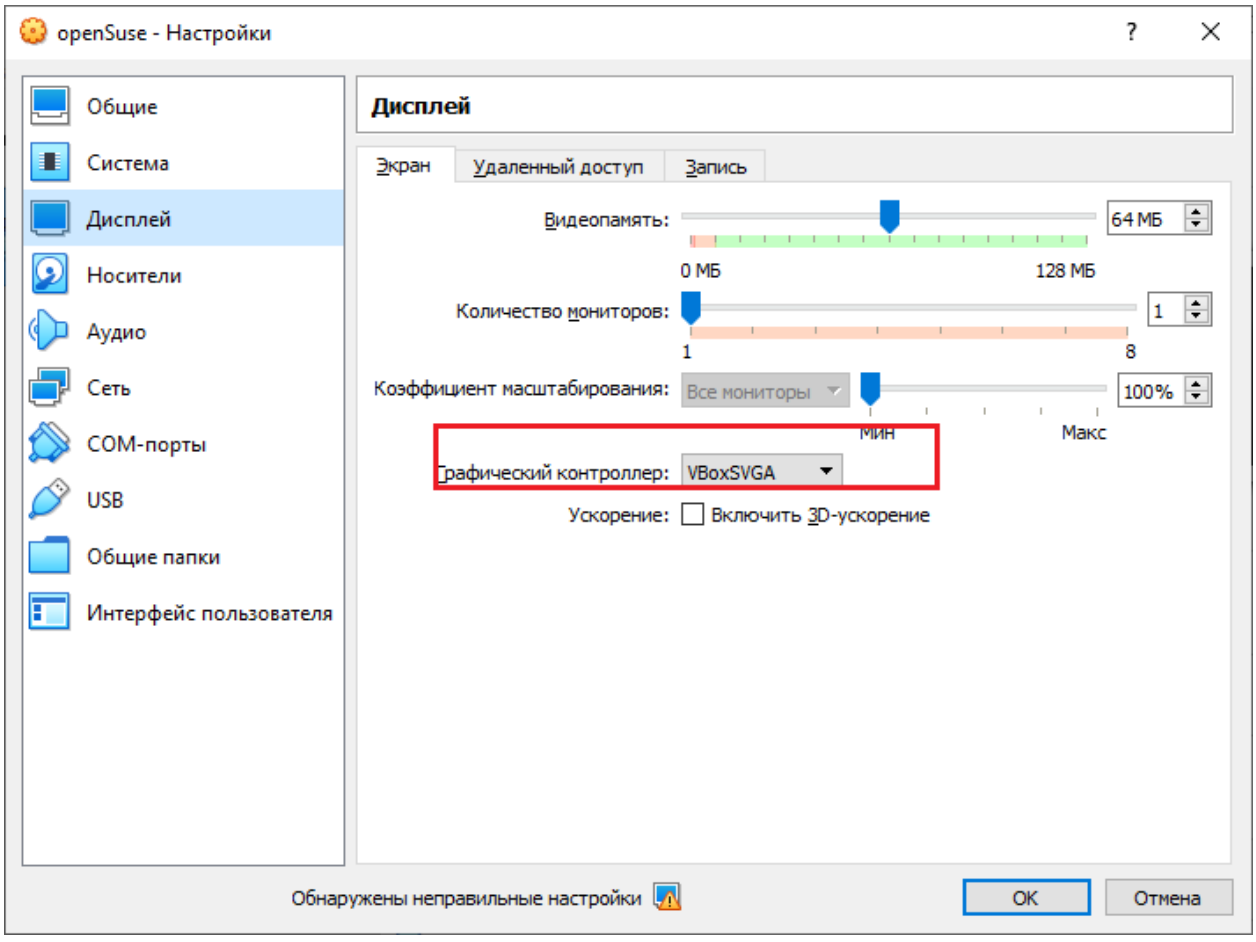

Рис. 1.1. Изменение настроек дисплея в VirtualBox 6.1

Можно, конечно, продолжить эксперименты с настройками, но пока это не существенно. При установке дистрибутива openSUSE я выбрал графический интерфейс KDE, программа KTechlab создавалась для этого интерфейса, хотя он считается «тяжеловатым». Но отведённых для работы Linux в «виртуальной коробке» 4 Гб оперативной памяти, мне кажется достаточно.

Почему openSUSE?

Для поиска готового к установке пакета KTechlab я использовал поисковик, рис.2.1.

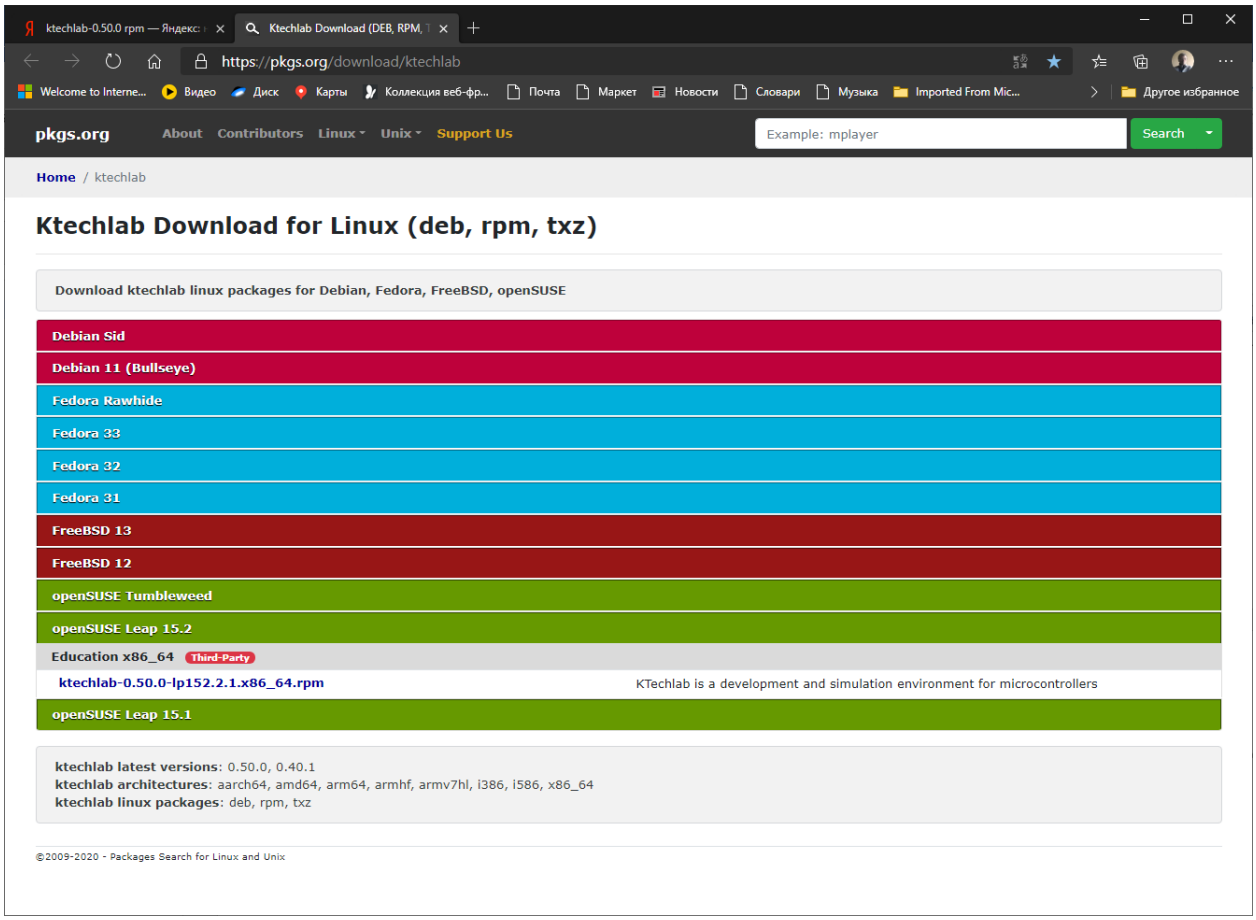

Рис. 2.1. Результат поиска пакета для установки KTechlab-0.50.0

Я предполагал, что установка программы будет сопровождаться установкой всего необходимого для полноценной работы, но, возможно, что-то пошло не так, пришлось установить ещё несколько программ после первой попытки. Так, если пытаться компилировать программу с языка Microbe на ассемблер, то появится сообщение, что не установлен пакет gputils. Попутно я установил программу gpsim – это симулятор для работы с микроконтроллерами. Нужно ли это? Не знаю, но когда-то было нужно.

Напомню, что многое в программе KTechlab работало давно, проблемы были с работой микроконтроллера: от попыток создать программу в среде Flowcode до попыток проверить работу программы с внешними компонентами.

Сейчас трудно вспомнить причины, из-за которых пришлось вносить изменения в права пользователя. Впрочем, проблемы возникли при обновлении. Любую ОС после установки следует сразу обновить. Попытка обновить систему оказалась неудачной. Как-то пришлось её обойти, а, пока не забыл, удобнее использовать обновление в консоли командой: sudo zypper update. Я не исключаю, что тогда и понадобилось добавить себя (любимого) в группу root. Хотя и не исключаю, что сделал попытку очистить папку временных файлов tmp, в чём мне было отказано, пока я не добавил себя в группу root.

Запомнилось последнее событие – попытка перенести нужный файл с флешки. Да, после подключения она определяется, но прав для работы с ней у пользователя нет. Не любят современные операционные системы пользователей; ой, не любят!

Для того, чтобы добавить себя дополнительно в какую-либо группу, следует осуществить несколько действий.

Вот, как выглядит этот процесс, рис. 2.2.

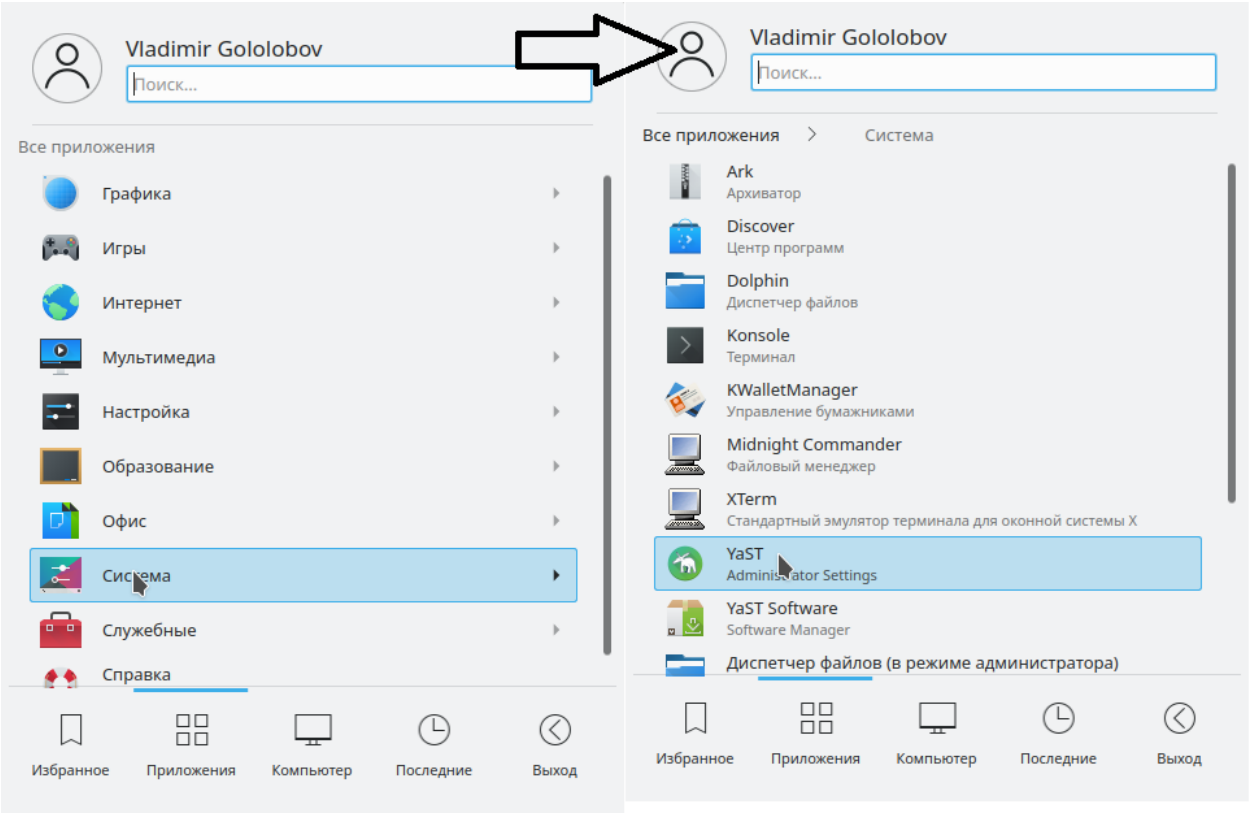

Рис. 2.2. Добавление пользователя в группы выбор приложения

Меню в openSUSE хорошо систематизировано. Так в разделе «Приложения» в разделе «Система» можно найти все важные для работы приложения. В данном случае понадобятся установки администрирования.

В этом приложении следует выбрать раздел «Управление пользователями и группами», далее двойным щелчком левой клавиши мышки выбрать себя (из списка, если пользователей несколько), рис. 2.3.

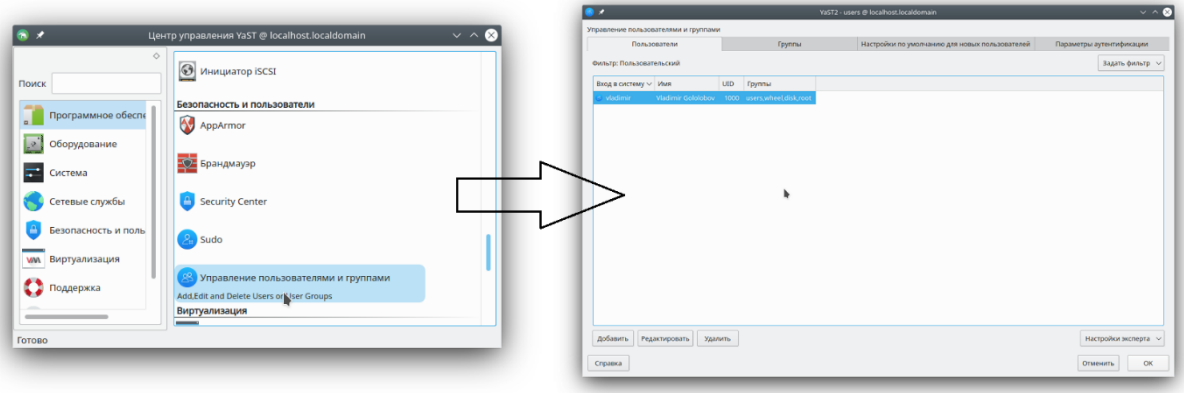

Рис. 2.3. Выбор пользователя

Теперь вы можете добавить себя в любые группы на ваше усмотрение, рис. 2.4. Я пробовал добавить себя в группу disk, пытаясь прочитать флешку, но после первого успеха повторилось всё с самого начала. Пришлось удалить себя из группы disk.

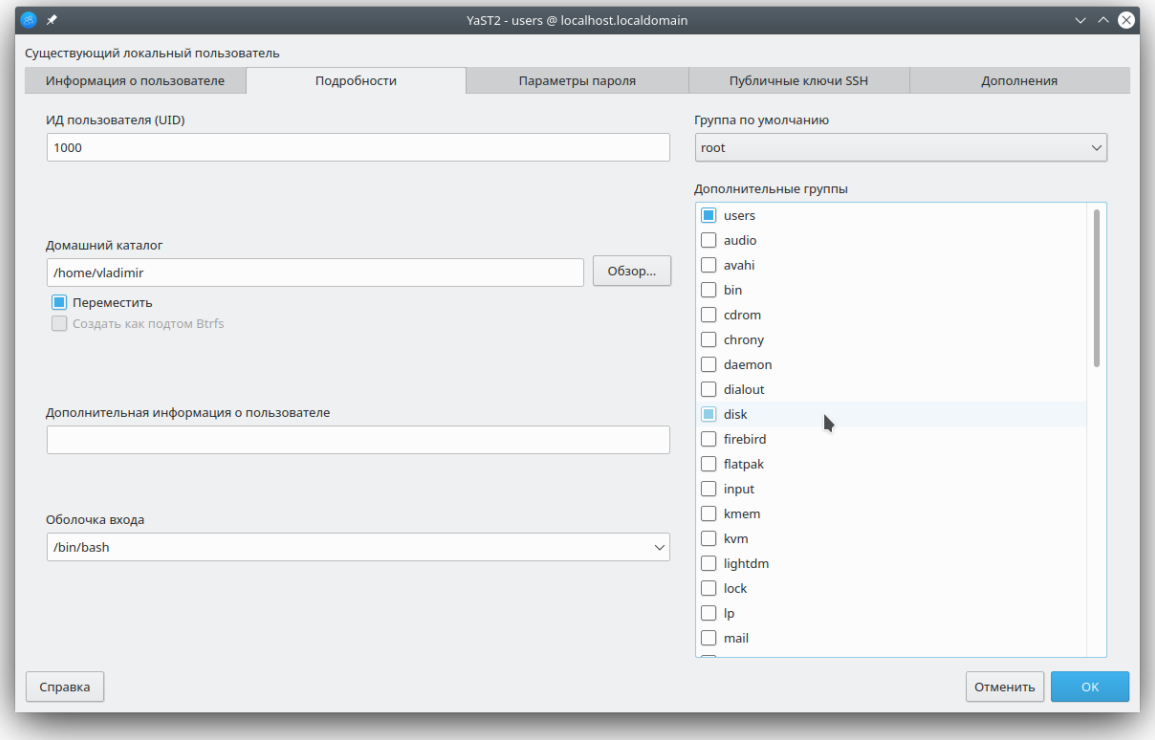

Рис. 2.4. Добавление пользователя в нужную ему группу

Я попробовал другой вариант: вставить флешку в порт USB, выйти из сеанса работы, зайти вновь. Подключить флешку (в нижней части, если вы в полноэкранном режиме работы, появляется меню), использовать раздел меню «Устройства», где есть USB устройства, останется найти свою флешку в списке. Посмотрим, будет ли это работать и дальше. Впрочем, когда спотыкаешься на каждом шагу, то, преодолев препятствия, забываешь о них. Проверим этот механизм работы для начала с флешкой.

И не напрасно. Всё оказалось проще, чем «показалось». При запуске openSUSE я не обратил внимание, что операционная система загружается, не требуя ввода пароля, что меня устраивает. Но, как оказалось, не устраивает флешку. При её подключении оказывается нет прав на работу с ней. Поэтому флешку отключаем. Завершаем сеанс. Начинаем новый. При этом требуется ввод пароля пользователя. Вставляем флешку, подключаем флешку. Можно с ней работать. Возможно, похожая ситуация повторилась и при установке обновлений. А ещё больше запутать могло то, что экран блокируется через некоторое время, а при разблокировании экрана вводится пароль пользователя, что может стать достаточным аргументом для подключения флешки, установки обновлений и т.п.

Первое знакомство с новой версией KTechlab

Как обычно, советую и вам, я начинаю знакомство с простейших программ. Таковой можно считать программу «помигать светодиодом». Можно, конечно, использовать и более простой вариант – включить один из выводов. К нему можно будет подключить светодиод, чтобы проверить работу программы. Но этот вариант может ввести в заблуждение по некоторым причинам от вас независящим. Итак, помигаем.

Начинаем знакомство с создания нового проекта, которому даём имя. Я начал с имени test. И, на всякий случай, папку с проектом создаём в директории /home/имя_пользователя (надеюсь, имя записано латиницей). После создания нового проекта выбираем тип проекта – Flowcode, рис. 3.1.

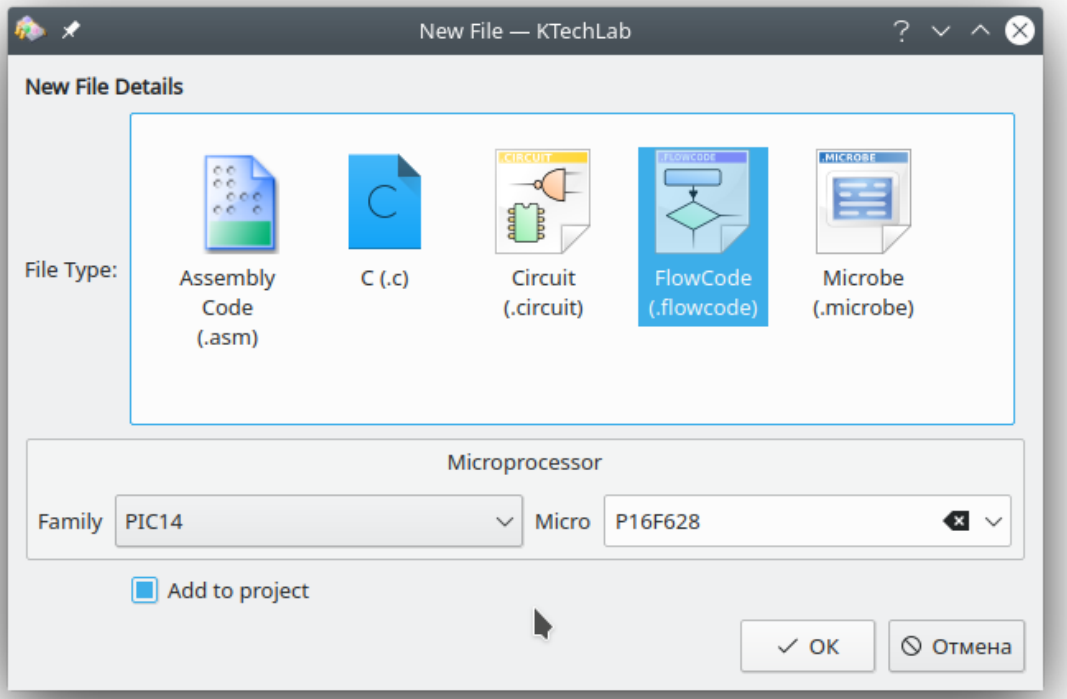

Рис. 3.1. Создание проекта с визуальным языком программирования

Я неоднократно говорил о том, что выбираю микроконтроллер PIC16F628 по причине достаточности его функций для первого знакомства, но, главное, по причине наличия описания на русском языке. В данном случае и выбор пока не велик. Выбрали микроконтроллер, выбрали тип проекта. В рабочем поле появился «сам микроконтроллер», у которого есть кнопка расширенных настроек «Advanced». Она открывает диалоговое окно, где можно задать выводы на вход или на выход, меняя единицу (работа на вход) соответствующего бита на ноль (работа на выход). Зададим работу вывода RA0 (нулевой порта А) на выход, рис. 3.2.

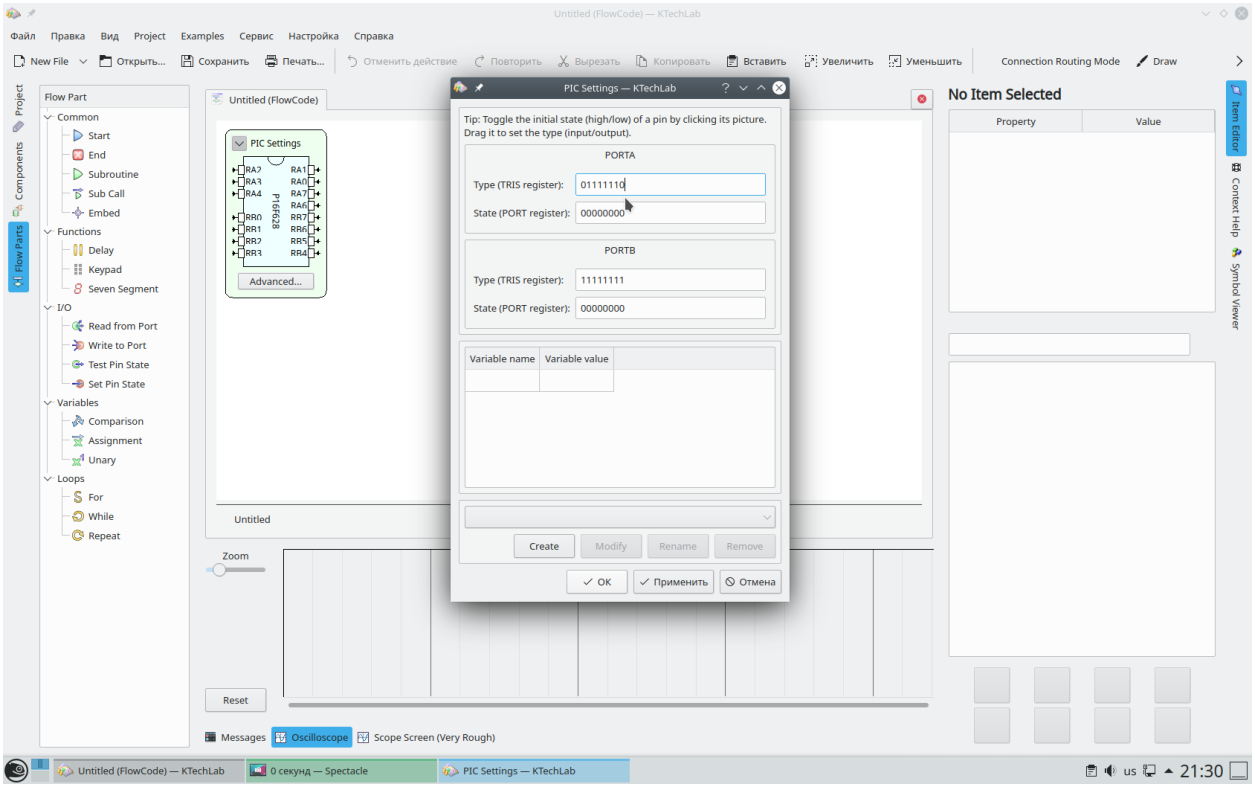

Рис. 3.2. Установка выводов микроконтроллера для работы

Теперь можно сохранить файл, дав ему имя. Я выбрал имя test. Позже я столкнулся с проблемой задаваемого имени. Но об этом позже. Сохранив файл, можно приступить к «сборке» программы, которая состоит из традиционного бесконечного цикла, в котором установим выход в высокий уровень, подождём немного, установим его в низкий уровень, подождём, и так пока не остановим работу микроконтроллера, рис. 3.3.

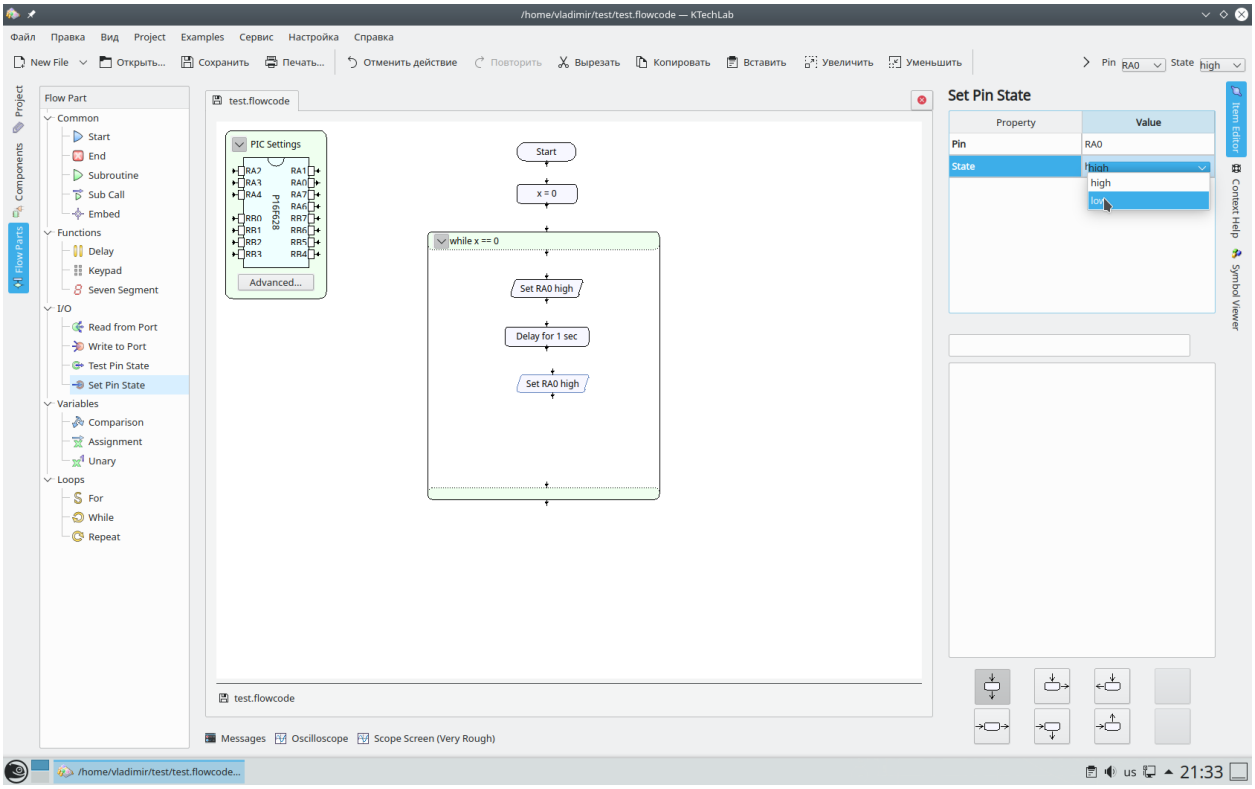

Рис. 3.3. Сборка программы

Компоненты берём с левой панели и мышкой перетаскиваем в рабочее поле. При необходимости изменить состояние щелчком выделяем его и меняем его состояние на правой панели свойств, как показано на рис. 3.3. Самый первый компонент, не входящий в цикл, это назначение переменной x значения ноль (таковым оно может быть и по умолчанию, но сделаем это на всякий случай). Два других компонента – это задание состояния вывода (Set Pin State) и пауза (Delay). Пауза задаётся в секундах. Попытка изменить это значение на меньшее у меня не получилась, хотя в свойствах компонента можно это сделать. При необходимости это можно сделать в следующем файле. Собранная программы выглядит так, рис. 3.4.

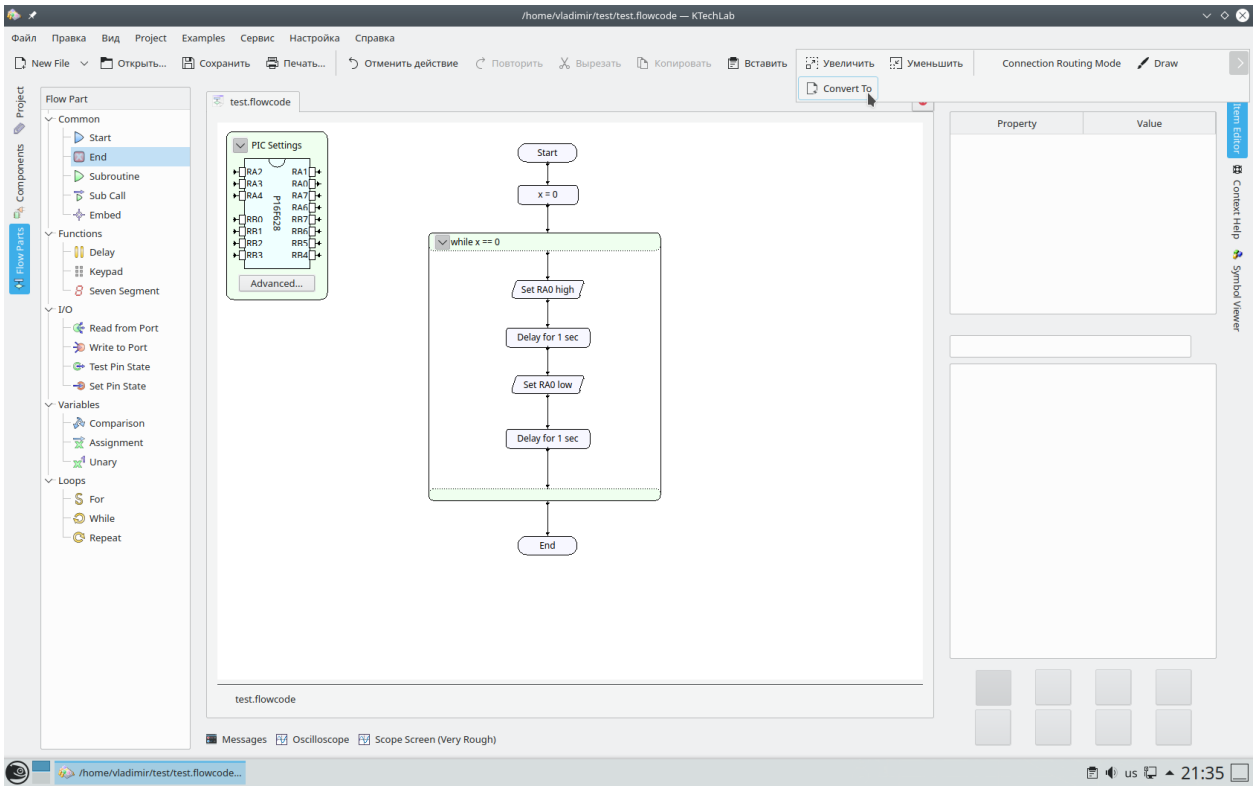

Рис. 3.4. Тестовая собранная программа

Я когда-то описывал порядок работы с программой, можно было бы не повторять, но я и сам что-то забываю, поэтому так подробно.

После сохранения собранной программы (всегда полезно сохранять сделанную работу, даже если она ещё не завершена) переместимся к основному меню, где строка меню завершается стрелкой, которая приводит нас к команде трансляции (Convert to). Нажав на этот пункт основного меню, мы получим возможность транслировать визуальный файл в файл языка Microbe, рис. 3.5.

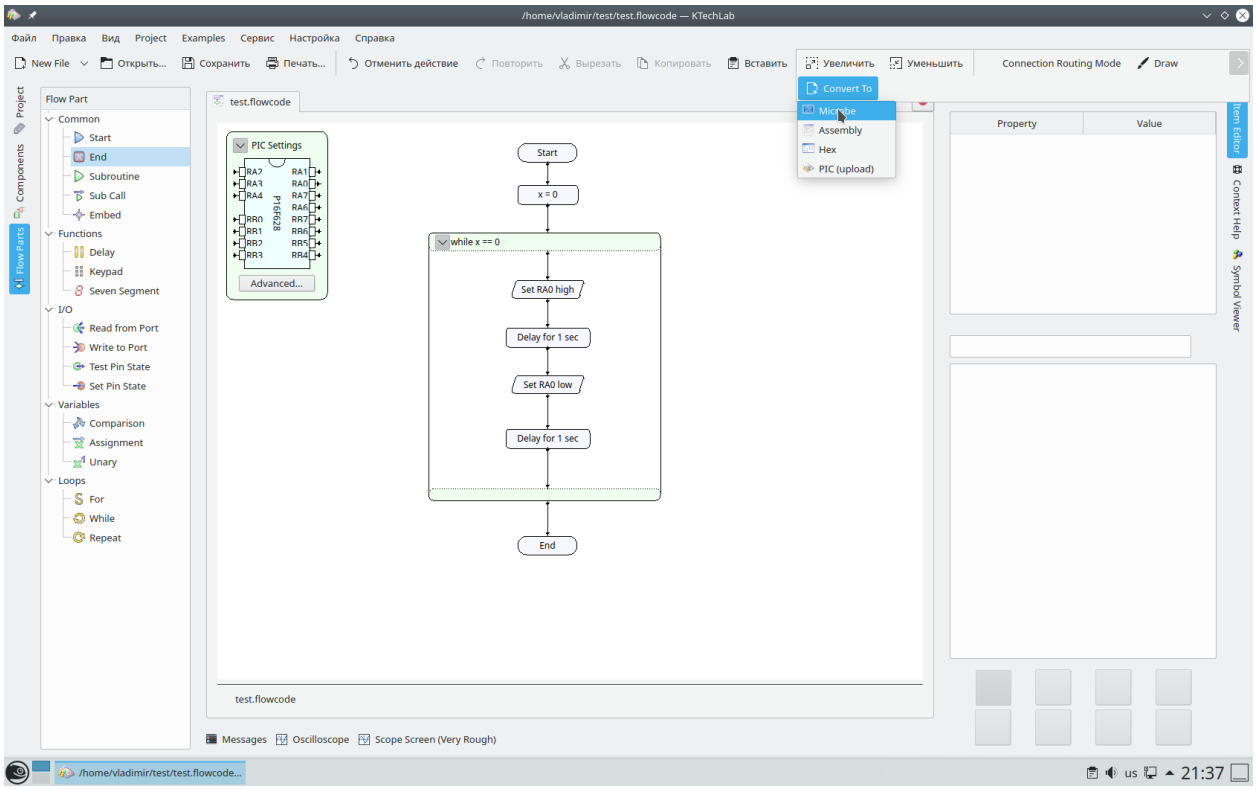

Рис. 3.5. Трансляция программы на язык Microbe

Если всё сделано правильно, то появится вторая закладка, где в рабочем поле будет запись программы на языке Microbe. Можно сохранить файл (под выбранным ранее именем), расширение файла microbe может появиться автоматически. В этом файле при необходимости можно уменьшить время паузы. Сохранив файл после переделок, переходим к трансляции на язык ассемблера, рис. 3.6.

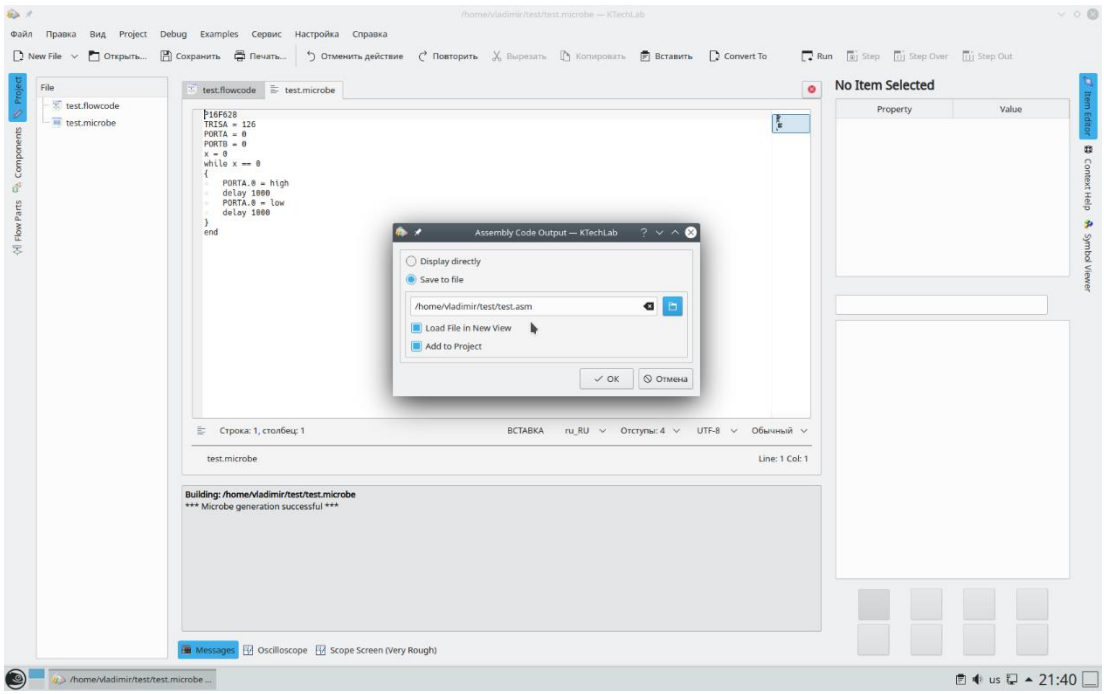

Рис. 3.6. Трансляция файла на язык ассемблера

При задании характера трансляции я выбираю сохранение файла, показ в новом окне и добавление файла к проекту (как и ранее).

Здесь есть ещё один момент. Я не знаю каким образом можно записать слово конфигурации при сборке проекта или на языке Microbe. Но это можно сделать в ассемблерном варианте, добавив после #include <p16f628.inc> ещё одну запись: __config 0x3f18 (два подчёркивания!).

Запись слова конфигурации определяет кроме прочих параметров тактовую частоту процессора. В данном случае используется внутренний генератор, тактовая частота 4 МГц. От этой частоты зависит и длительность пауз!

Закончив с файлом на языке ассемблера, сохранив файл, можно оттранслировать его в HEX-формат с помощью команды Convert to. При этом создаётся файл с расширением .cod, который используется для моделирования работы микроконтроллера. Для этой цели создаём файл из набора доступных с типом Circuit. Не забываем выбрать контроллер. Левая панель при этом предоставит набор компонентов электрической цепи на закладке «Components». Переносим компонент pic в рабочее поле. Здесь есть несколько особенностей (возможно, только у меня). Нельзя сохранить файл под именем, неактивны эти разделы. Но, если вернуться к предыдущей закладке, скажем, test.hex (на рисунке), затем перейти к схеме, то нужные разделы активируются.

По сценарию работы двойной щелчок по изображению микроконтроллера в рабочем поле должен открывать его свойства, в которых следует указать требуемый файл cod для работы. Но, когда всё работает, для этого приходится выйти из программы, запустить её вновь, в строке основного меню появляется подсказка: > Program с окошком для ввода пути к файлу (удобнее использовать крайнюю справа кнопку с изображением папки). Но…

Здесь меня подстерегала неприятность, с которой я долго возился, рис. 3.7.

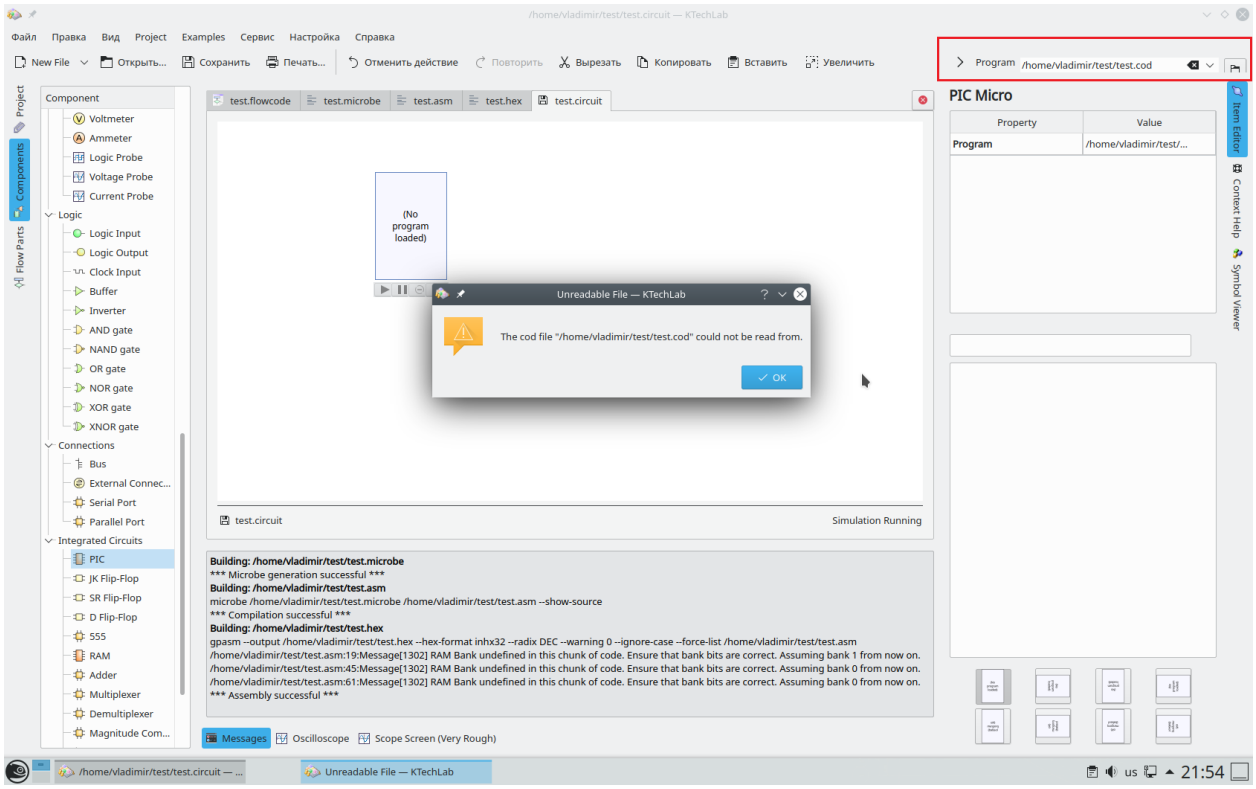

Рис. 3.7. Невозможность загрузить нужный файл

Я постарался проверить всё – от расположения до кодировки файла, но так и не понял, в чём проблема. Обошёл я её тем, что стал использовать в названии проекта и именах файлов три

буквенных символа (не исключаю, что и цифры тоже работают). После этого моделирование пошло вполне успешно. Но из-за задержки я, возможно, что-то упустил в своём рассказе, настолько неожиданным было очередное препятствие. Пока не забыл, вот строка добавления слова конфигурации, рис. 3.8.

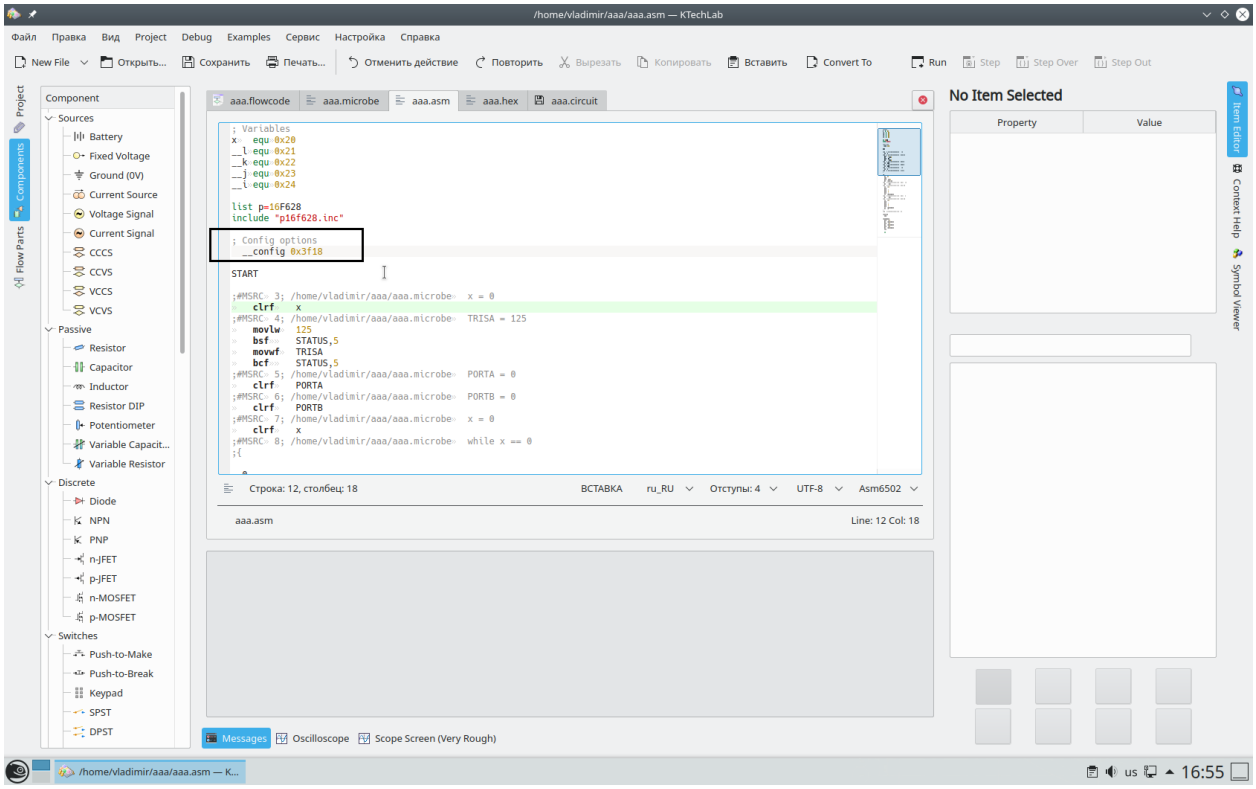

Рис. 3.8. Добавление слова конфигурации с нужными параметрами

Преодолев эту проблему, можно запустить моделирование: проверьте в строке состояния, работает ли симуляция (если нет, то запустите в разделе «Сервис»), запустите симуляцию, используя кнопку проигрывать на маленькой панели под микросхемой. Если в настройках KTeclab вы задали анимацию напряжения на выводах, то это будет отображаться. Можно добавить светодиод к выводу, но яркость цвета у меня получилась невысокая, невыразительная. Лучше использовать логический выход (Logic Output), рис. 3.9.

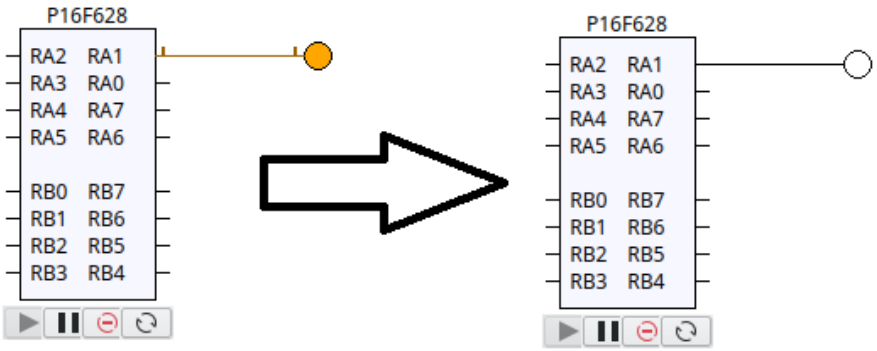

Рис. 3.9. Наблюдение за выходом

К выходу можно подключить осциллограф. Если скорость изменения сигнала достаточно высокая, вы можете видеть вполне внятную картинку, рис. 3.10.

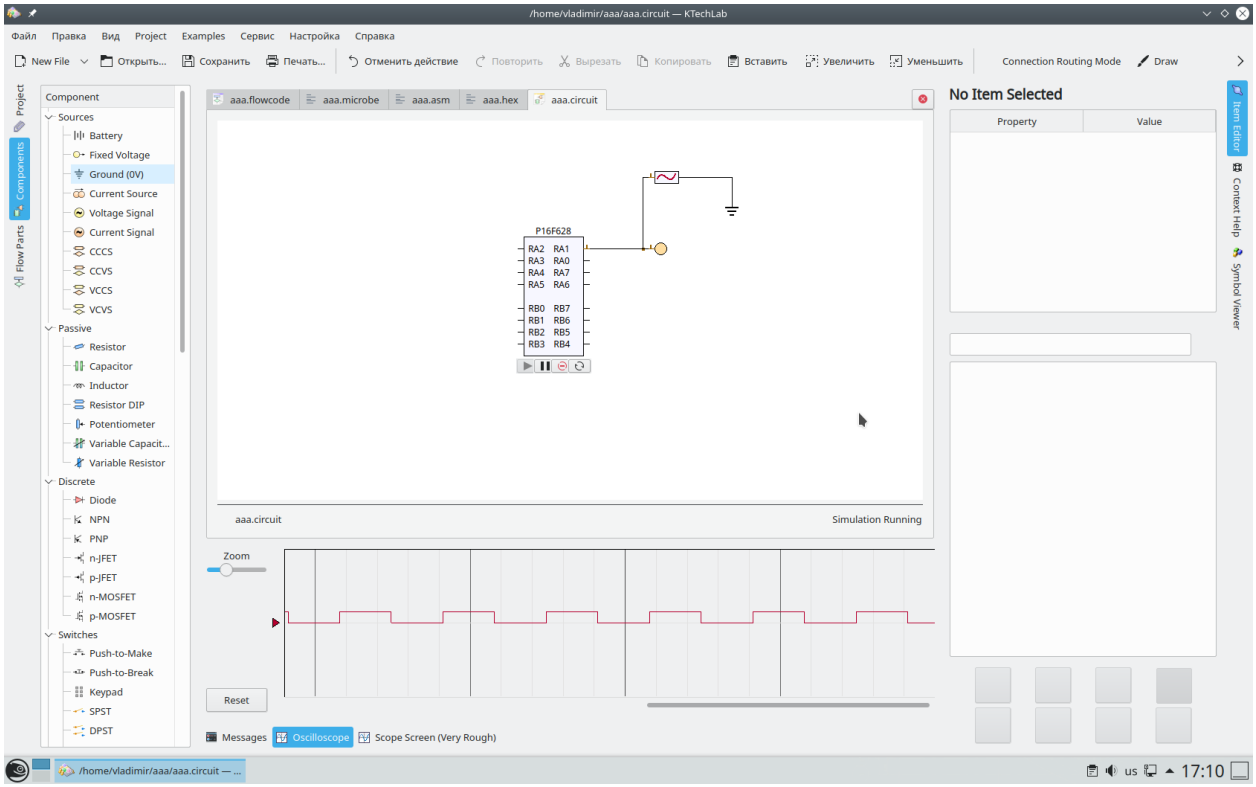

Рис. 3.10. Наблюдение за выходным сигналом

Выходной сигнал – это прямоугольные импульсы, их можно наблюдать и с помощь логического пробника (Logic Probe), рис. 3.11.

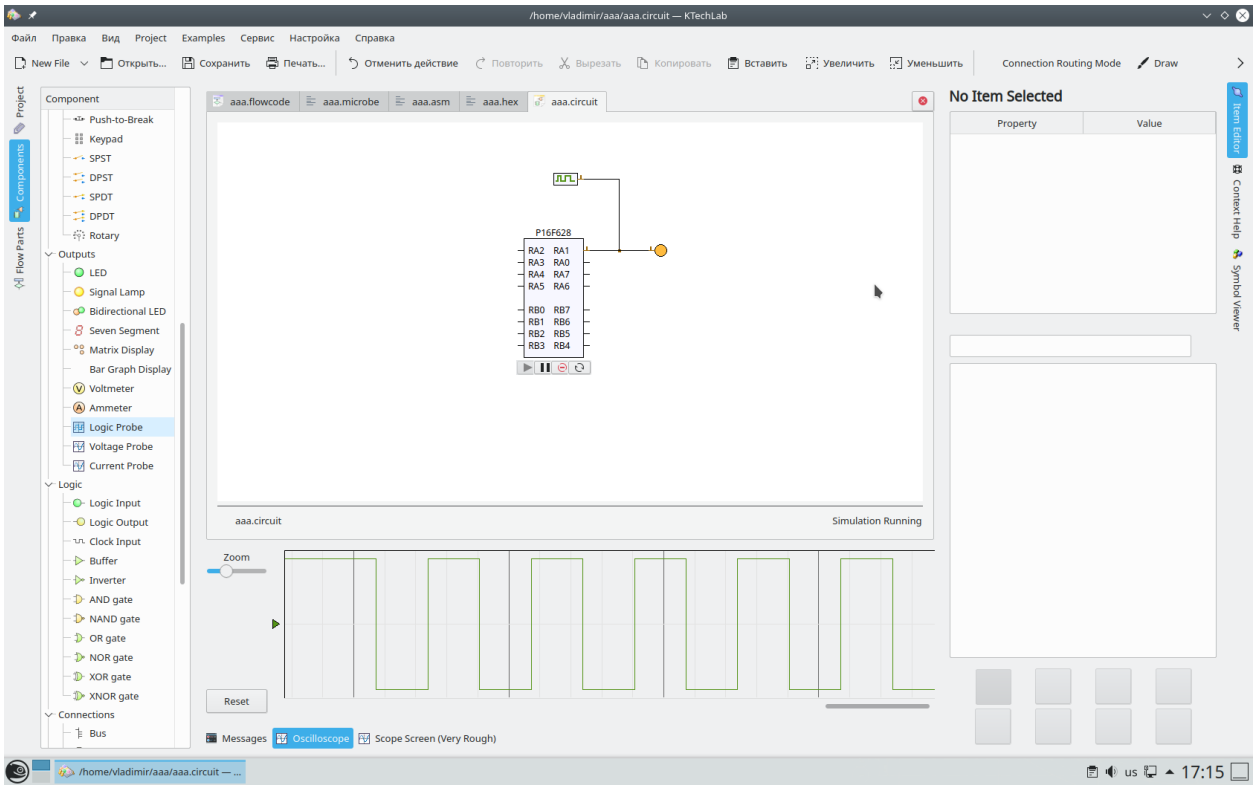

Рис. 3.11. Наблюдение за выходным сигналом с помощью логического пробника

Если визуального программирования недостаточно

Когда-то я сделал попытку изменить состав регистров микроконтроллера для языка Microbe. Это потребовало изменения в одном из файлов исходного кода программы KTechalab с последующей компиляции программы из исходных кодов.

Сегодня, этого нельзя исключить, подобная правка не получится. Надо пробовать.

Речь шла о попытке использовать встроенный в микроконтроллер модуль USART. Если вы хотите использовать все возможности микроконтроллера, то можно воспользоваться встроенной в программу возможностью работать с файлами на языке Си или ассемблере. Попробуем написать программу на языке Си, использующую модуль USART.

Создаем новый проект, я назвал его ace, но теперь для языка Си, рис. 3.1. После создания проекта появляется текстовый редактор, в котором предстоит ввести текст программы. Мне хотелось бы сделать всё «элегантно», но не получилось. Пришлось, как часто получается, делать всё по-простому. Вот текст программы:

```
#include <pic16regs.h>

__CONFIG (0x2007, 0x3f18);


void delay(int iterations)
{
      int i;
      for (i = 0; i < iterations; i++) {
            __asm nop __endasm;
      }
}

void main(void){
TRISA = 126;
CMCON = 0x7;
TRISB = 0xF6;
RCSTA = 0x90;
STATUS = 0x20;
SPBRG = 0x16;
STATUS = 0;
INTCON = 0;

while(1) {

    RA0 = 1;

    STATUS = 0x20;
     TXSTA = 0x20;
     STATUS = 0;
     PIR1 = 0;

    TXREG = 0x31;

     label_flag2:
     while (PIR1 != 0x10)
     {
           goto label_flag2;
     }
```

```
    delay(40);
        RA0 = 0;
}
}
```

Когда-то я решал проблему с компонентом delay для компилятора sdcc, создавая несколько заголовочных файлов, сейчас мне этим заниматься не хочется, поэтому я использую «срисованный» в Интернете вариант.

Текст программы можно запустить для трансляции на язык ассемблер, но…

Заглянем в настройки программы KTechlab, вначале на страницу того, о чём я говорил раньше, рис. 4.1.

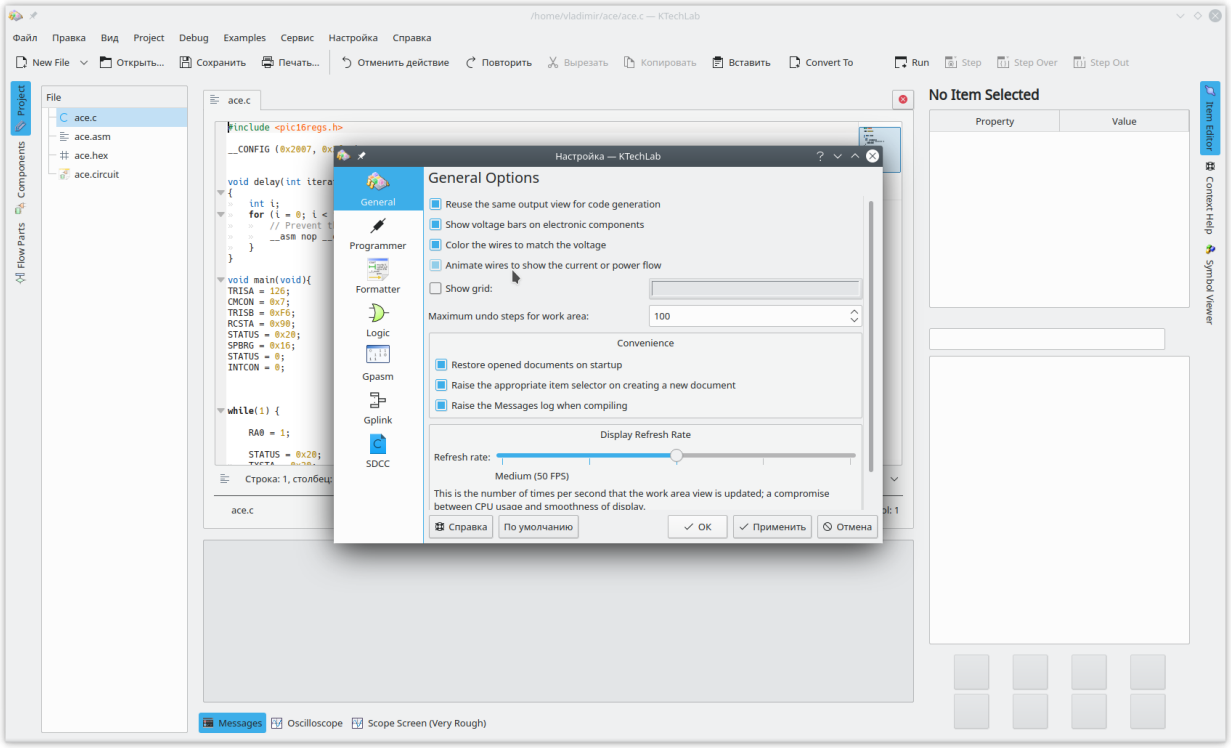

Рис. 4.1. Настройка отображения активности «проводов» (или выводов)

Затем перейдём к настройкам компилятора SDCC, рис. 4.2.

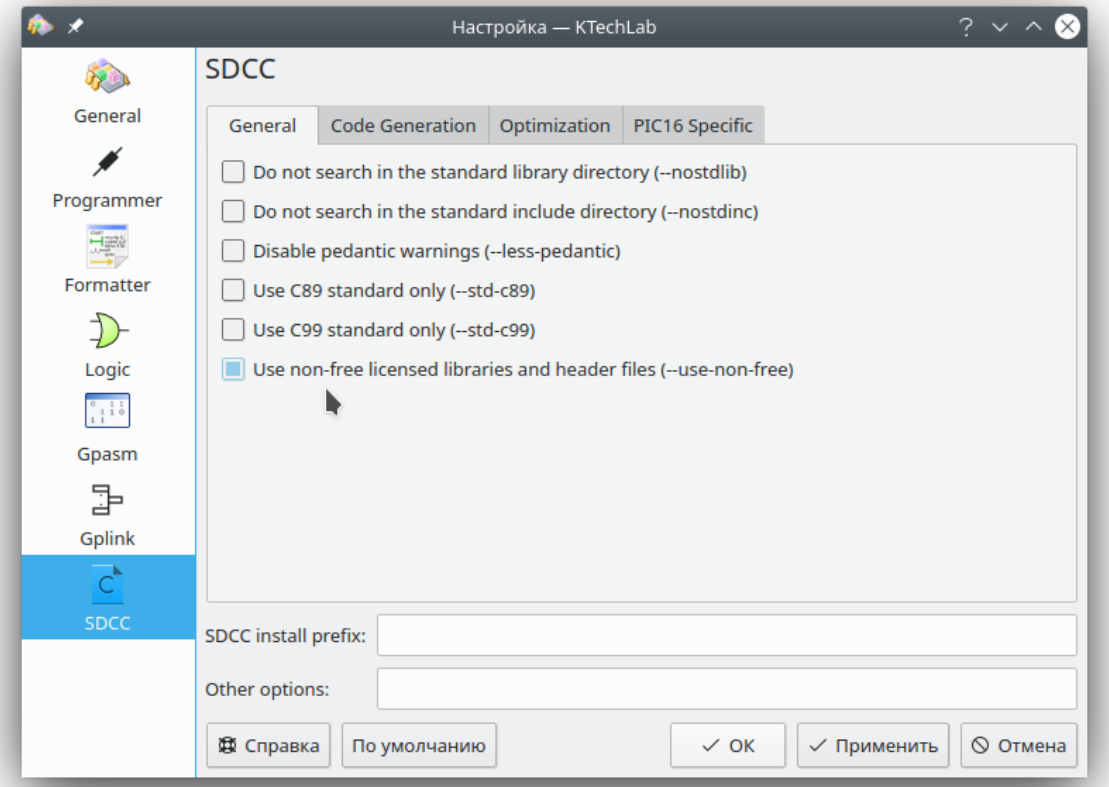

Рис. 4.2. Настройка компилятора sdcc

На этой странице есть много закладок и возможностей, которые ещё предстоит изучить, если вы намерены в полной мере использовать программу KTechlab. Но нас пока интересует тот факт, что используется none-free библиотека. При установке компилятора для Linux я не нашёл этой библиотеки. Не знаю, как это исправить в «общем виде», я вспомнил, что раньше использовал компилятор в Windows, откуда и «срисовал» нужную библиотеку, вставив её в то место, где программа пыталась найти библиотеку.

Теперь компиляция проходит. Для проверки работы программы при моделировании я добавил «мигание светодиодом» на выводе RA0. При моделировании этот сигнал есть, но сигнала на выходе USART я не вижу. В чём проблема я не знаю, но я и в прошлый раз получил такой же результат, рис. 4.3.

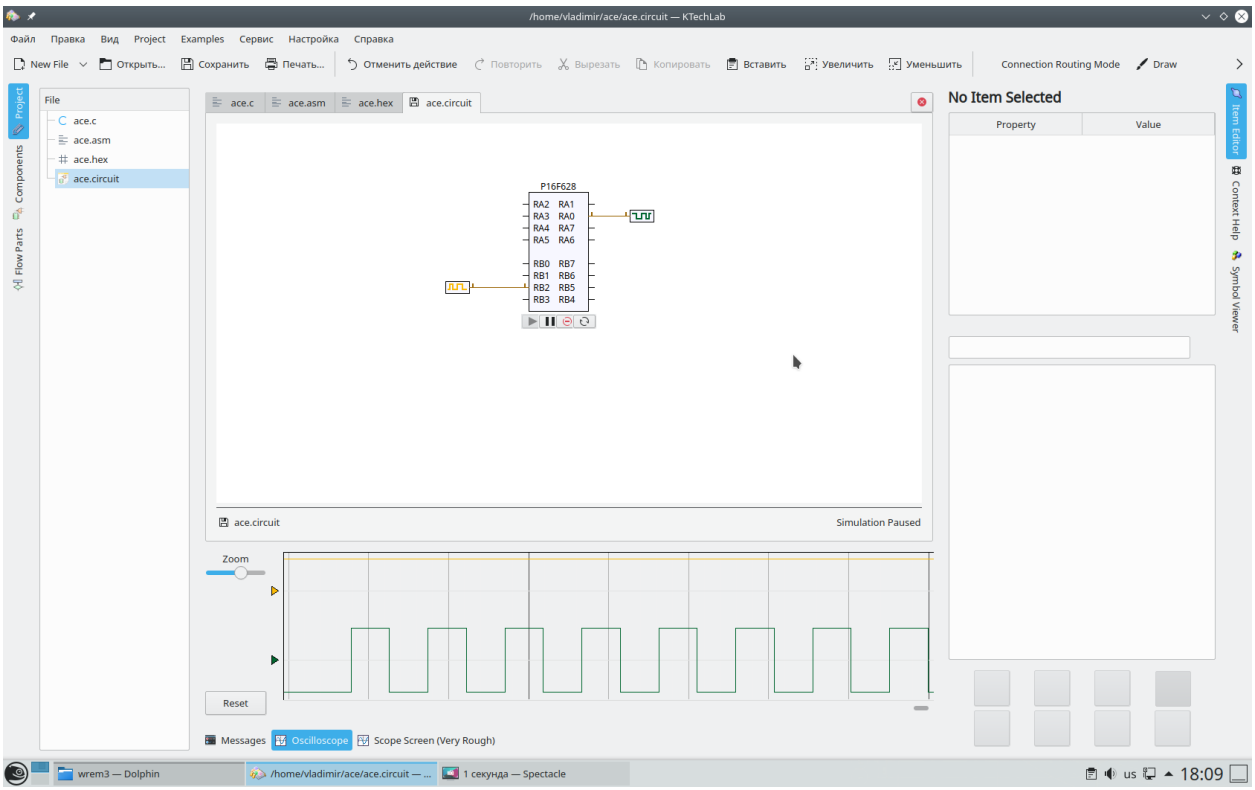

Рис. 4.3. Сигналы на выходах микроконтроллера

В прошлый раз, чтобы убедиться в наличии сигнала на выводе RB2, я использовал программу ISIS. Но её нет для версии Linux. Однако есть бесплатная программа SimulIDE. Я пробовал с ней работать в Windows, но есть и её версия для Linux. Чтобы найти исполняемую программу, можно воспользоваться поиском в любом web-проводнике. Мой браузер Edge нашёл программу достаточно быстро, а на странице загрузок можно увидеть все возможные варианты, рис. 4.4.

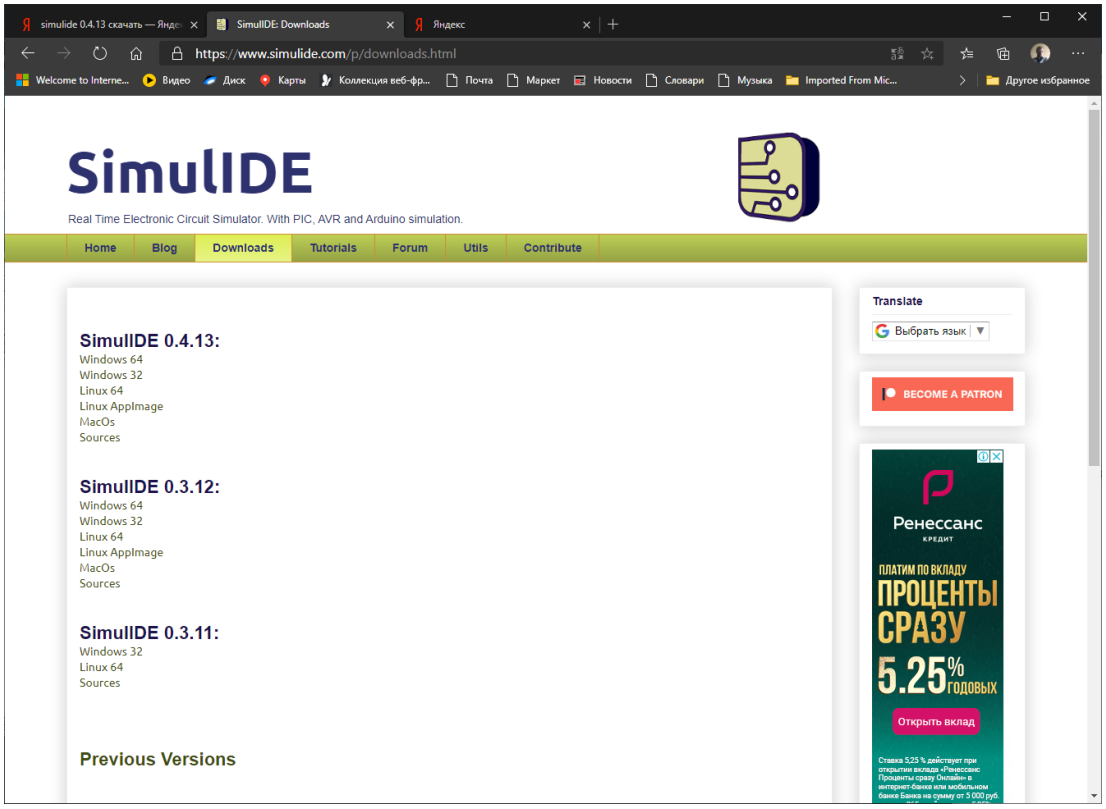

Рис. 4.4. Возможности загрузки исполняемого варианта программы SimulIDE

Действия после загрузки лаконичны: разархивировать файл, зайти в папку bin, щелкнуть по названию программы. Появляется предложение запустить программу, рис. 4.5.

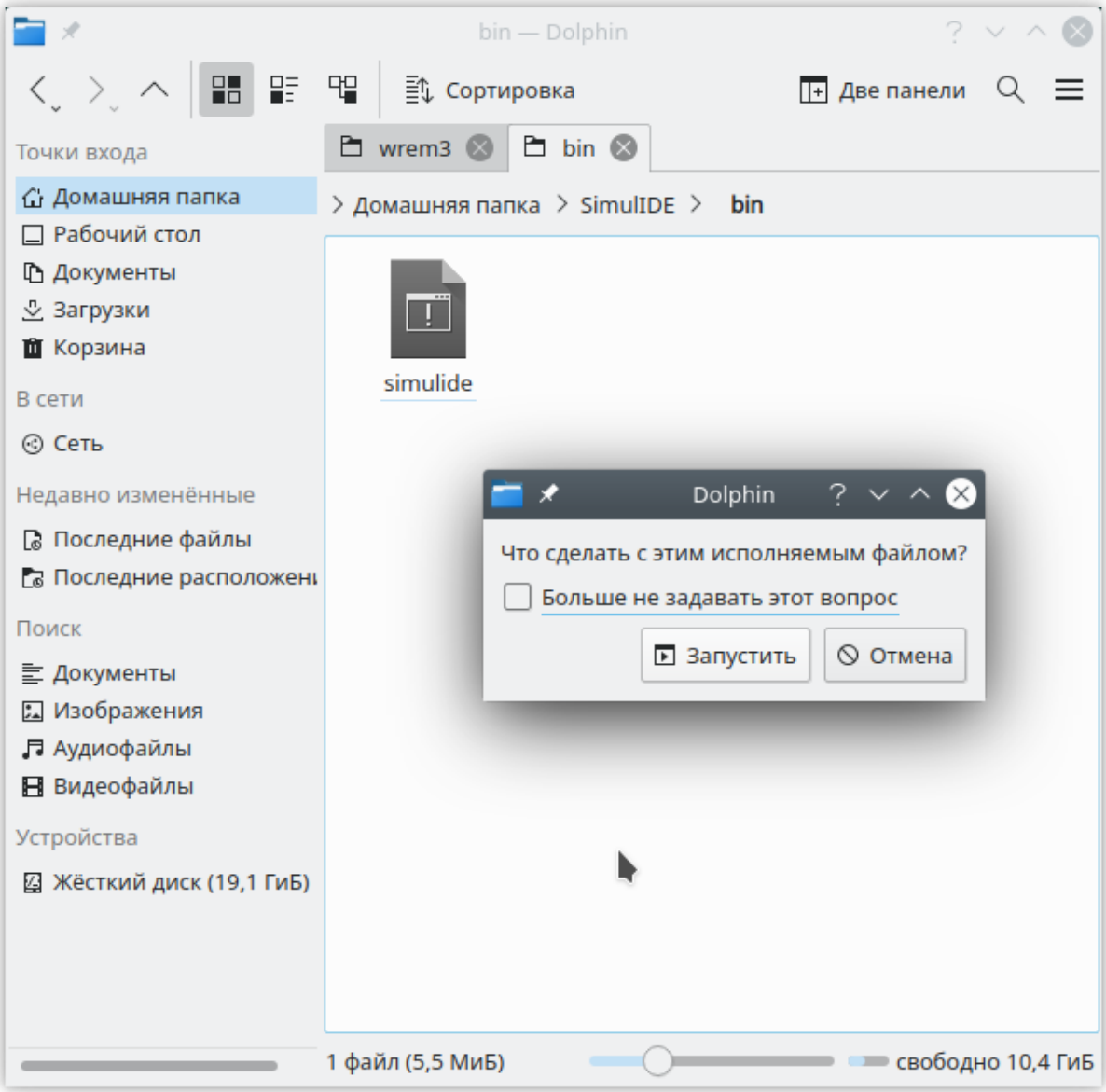

Рис. 4.5. Приглашение к запуску программы

Но не в этот раз. Сколько я не щёлкал, один раз или два, ничего не получалось. Я проверил, файл исполняемый. Но не в этот раз. И не важно, где папка с программой (я сразу исключил её расположение в директории с кириллицей). Для подобных случаев непонятного поведения исполняемой программы удобно воспользоваться консолью (терминалом), рис. 4.6.

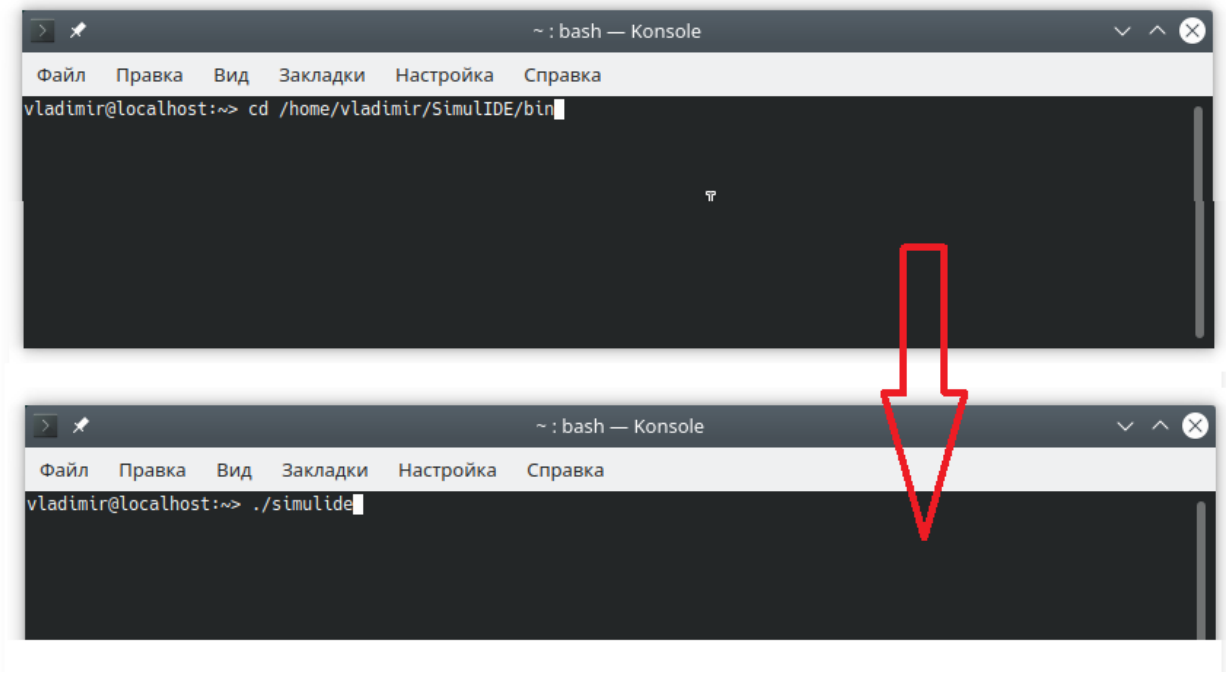

Рис. 4.6. Запуск программы в консоли

Запуск в консоли поясняет, что в операционной системе не хватает одного из системных файлов, которые требуются для работы программы. Этот файл удаётся найти в Интернете с помощью поисковика. После установки файла программа SimulIDE работает, а сигнал на выводе передатчика USART появляется, рис. 4.7.

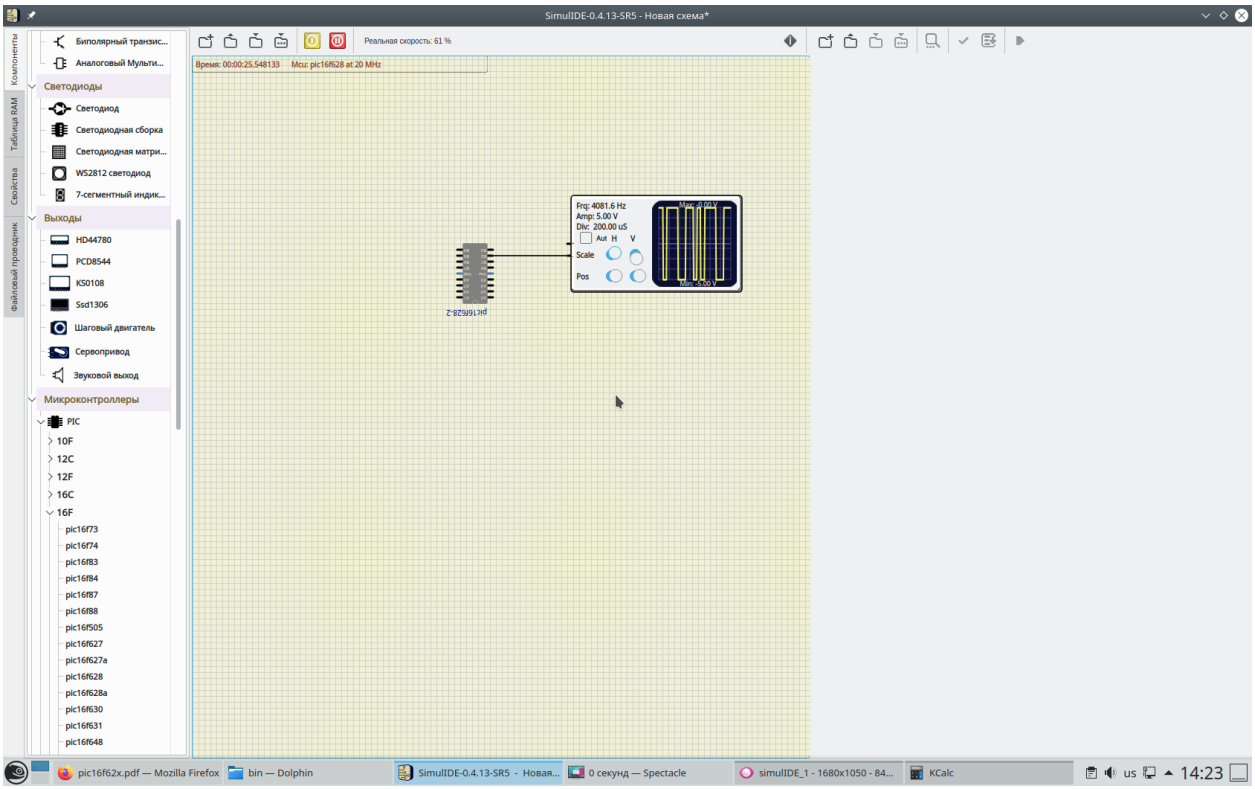

Рис. 4.7. Сигнал на выводе передатчика последовательного порта микроконтроллера

Не хочу утверждать с абсолютной уверенностью в своей правоте, но сигнал похож на символ единицы (0x31), который в двоичном виде выглядит как 00110001 (первым на экране осциллографа, напомню, появляется последний, нулевой бит).

Послесловие

Почему я столько времени уделил микроконтроллеру?

Всё просто – когда автор программы перестал её поддерживать, по возобновлению работы над проектом исчезла возможность работать с микроконтроллерами. И я очень рад, что эта возможность появилась.

Между тем, программа позволяет многое. Не знаю, годится ли она для использования в ВУЗах, но точно её можно использовать в колледжах, школах, в радиокружках. Когда за использование пиратских версий не наказывали, все охотно использовали более мощные программы. Но не сейчас. Сегодня многие ноутбуки имеют 64-битовые процессоры, 8 и более гигабайт памяти и в операционной системе Windows 10 можно использовать либо встроенную виртуальную машину для установки Linux, либо использовать, как это сделал я, VirtualBox. Linux работает вполне прилично. А у меня только 8 Гб оперативной памяти. На ноутбук можно установить и больше. Вдобавок память у меня достаточно старого типа, сегодня она работает со скоростью в два-три раза выше.

А программа KTechlab, загляните в раздел примеров, может рассказать о работе, например… рис. 5.1.

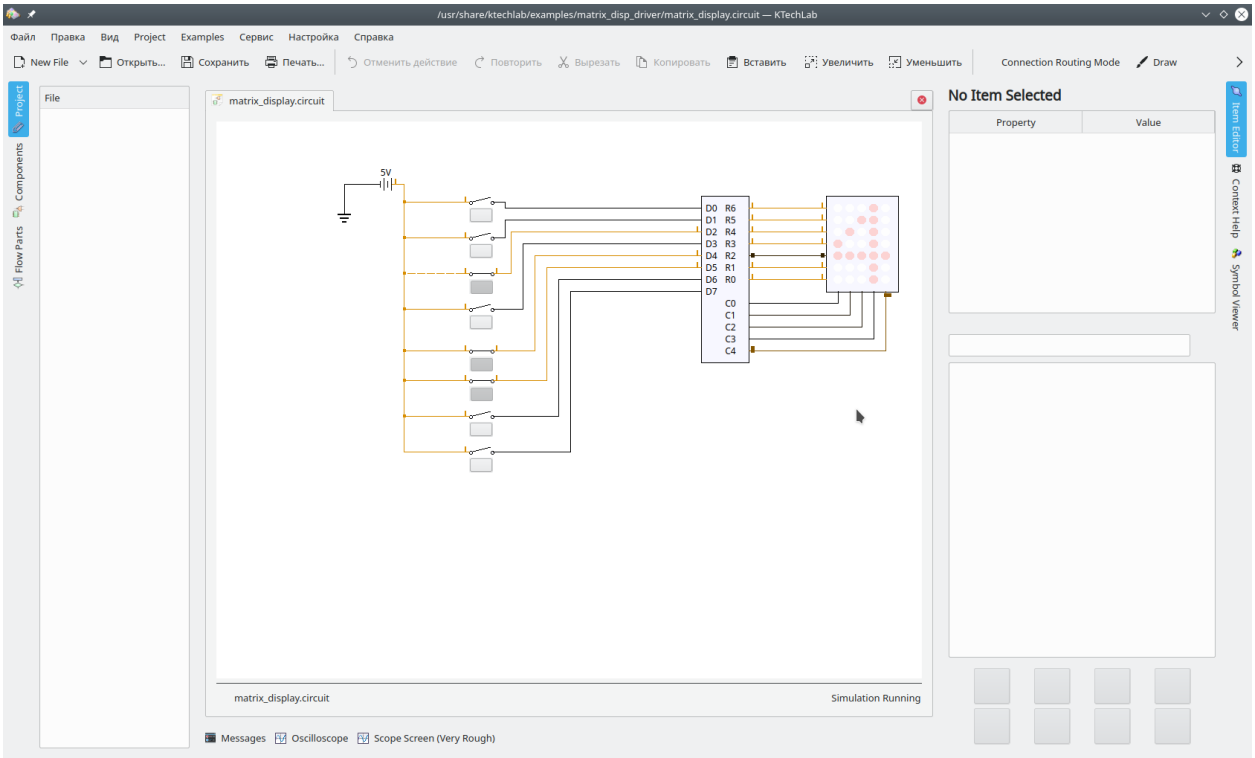

Рис. 5.1. Пример из набора, поставляемого с программой KTechlab

И в компонентах электрических схем есть операционные усилители, куда же без них, рис. 5.2.

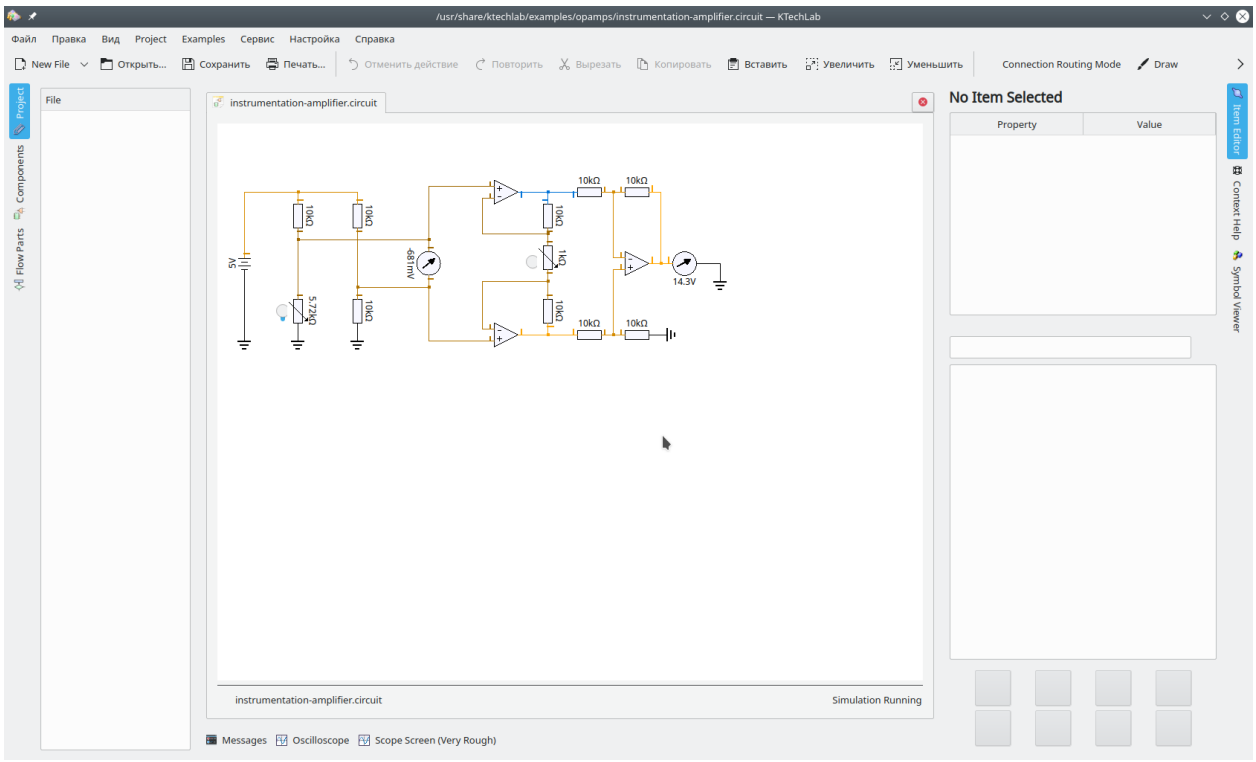

Рис. 5.2. Операционные усилители в программе KTechlab

Есть широко применяемый до сегодняшнего дня таймер 555, рис. 5.3.

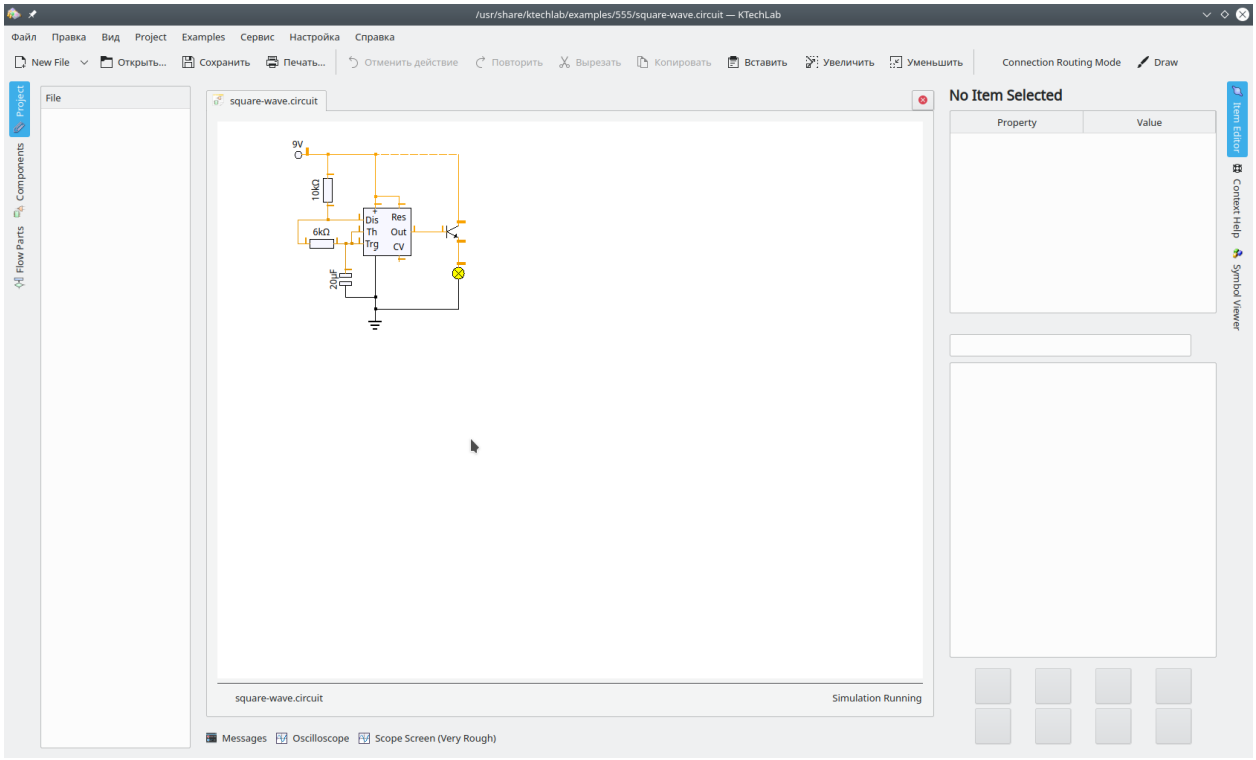

Рис. 5.3. Мультивибратор на микросхеме 555

И, что очень полезно, можно проверить работу схем на транзисторах, рис. 5.4.

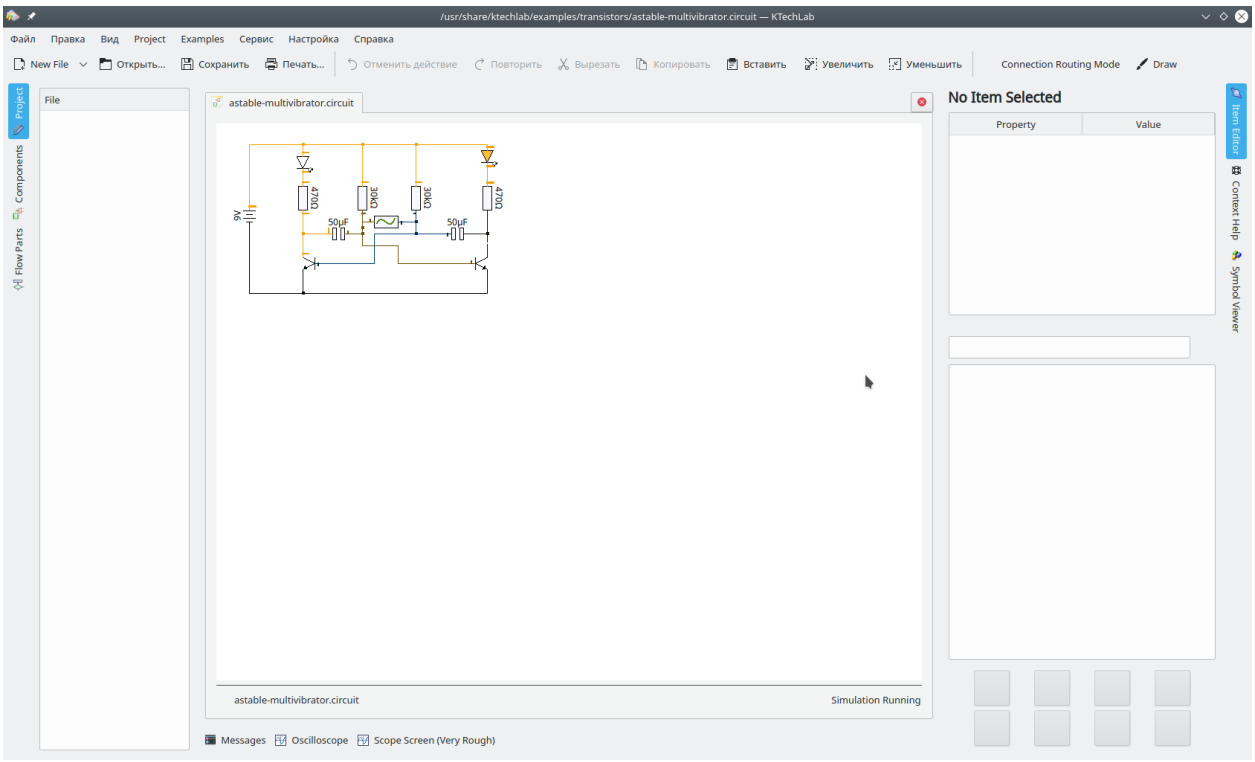

Рис. 5.4. Мультивибратор на транзисторах

И ещё есть много полезного в программе KTechlab. Программа бесплатная, доступная всем. А добавьте к ней программу SimulIDE, которая тоже многое позволяет, которая тоже доступна и бесплатна… Спасибо тем, кто создаёт эти программы!

Место для заметок