Экономичная реализация графического интерфейса пользователя на базе одноплатного компьютера Raspberry Pi

А.В. Ескин, В.А. Жмудь, В.Г. Трубин
ФГБОУ ВПО НГТУ, Новосибирск, Россия

Аннотация: **Освещаются вопросы касающиеся применения одноплатного компьютера Raspberry Pi в качестве дешёвой платформы для создания устройств с графическим интерфейсом сравнимым с персональным компьютером.**

***Ключевые слова*: Одноплатный компьютер Raspberry Pi, ARM, SoC, графический интерфейс пользователя, Linux, Window среда разработки Qt, кросс-компиляция, среда разработки программ Eclipse, удалённый доступ, SSH.**

ВВЕДЕНИЕ

Бурным развитием микроэлектроники ни кого уже не удивишь. То, что было новинкой в прошлом году, уже в году нынешнем считается устаревшим. Примеров этому множество. Проникновение электроники в жизнь простого человека несведущего в электроники стало настолько сильным, что такое устройство как сотовый телефон превратился в показатель престижа и достатка. Внедрение в повседневную жизнь новых устройств, таких как КПК, планшетный компьютер, ноутбук и других, привело к массовому потреблению электронной продукции, что, несомненно, сказывается на её цене.

С бурным ростом портативной цифровой техники, подстёгиваемым рекламой, стали доступны по низкой цене весьма производительные процессоры, которые приближаются, или даже в чем то, являются мощнее, процессоров персональных компьютеров. Это обстоятельство, в настоящее время, даже привело к сокращению рынка персональных компьютеров.

Массовое потребление приводит не только к снижению цены, но и даёт благодатную почву для развития технологий. Устройства становятся компактнее, производительнее, быстрее и с меньшим уровнем тепловых потерь. Всё это вместе позволяет делать то, что раньше казалось немыслимым: реализовать полноценный компьютер в виде одной, небольшой, печатной платы. Конечно эта идея родилась не только что, а известна уже давно. Так, например, появились микроконтроллеры (одно кристальные микро ЭВМ). Но за годы развития компьютеры ушли далеко вперёд. Появились операционные системы с графическим интерфейсом, целый набор различных сложных коммуникационных интерфейсов, устройства ввода вывода (клавиатура, мышь, дисплей и т. п.), которые были просто не по плечу обычному микроконтроллеру из-за технологических ограничений и поэтому не возможно было реализовать идею «однокристального компьютера» в полной мере.

Так продолжалось до недавнего времени. Шагом на пути к этой цели стали одноплатные компьютеры. Они появились благодаря появлению целого класса микроэлектронных устройств – систем на кристалле (*System on a chip* сокращенно – *SoC*).

Первым, и пожалуй наиболее удачным представителем таких устройств стал одноплатный компьютер *Raspberry Pi* созданный сообществом *Raspberry Pi Foundation* [1]. За два года своего существования (на момент 2014 год) создал рынок подобных устройств и сделал революцию в жизни простого радиолюбителя. Позиционируясь как учебная платформа для детей [2], приобщил большую массу людей к компьютерам и связанных с этим технологий на более продвинутом уровне, чем обычный Пользователь. Тем самым показав, что компьютеры это просто.

Raspberry Pi обладает возможностями сравнимыми с современными компьютерами, но в разы дешевле и компактнее. Нас эта плата будет интересовать как база для построения устройств с графическим интерфейсом пользователя аналогичным компьютерному. То есть подразумевает наличие клавиатуры, мыши и компьютерного монитора. Графический интерфейс должен представлять собой привычный в компьютерах оконный интерфейс.

ОПИСАНИЕ RASPBERRY PI

Компьютер представляет собой шестислойную печатную плату размером чуть больше чем банковская карта (85 х 56 мм). Бывает двух модификация «*A*» и «*B*». Модель «*B*», по сравнению с «*A*» имеет 512 МБ оперативной памяти (против 256 МБ в модели «*A*»), дополнительный *USB2.0* порт (в сумме 2 порта у модели «*B*») и порт *Ethernet* (в модели «*A*» отсутствует). Внешний вид компьютера модели «*B*» представлен на *Рис*. 1 и 2.

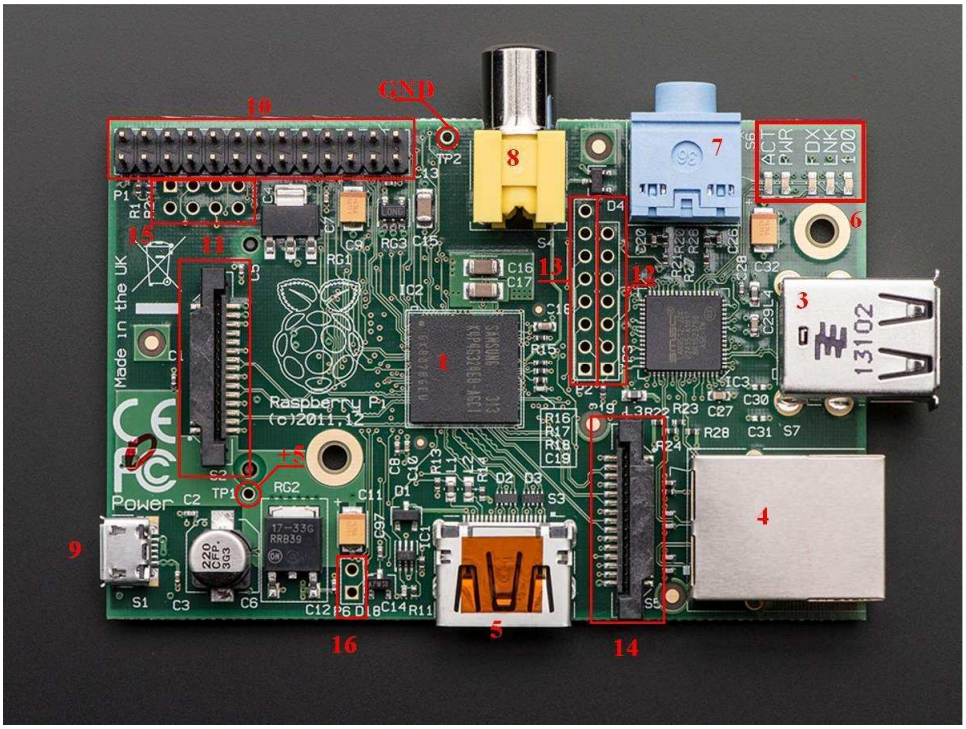

Рис. 1. Внешний вид одноплатного компьютера *Raspberry Pi* (вид сверху)

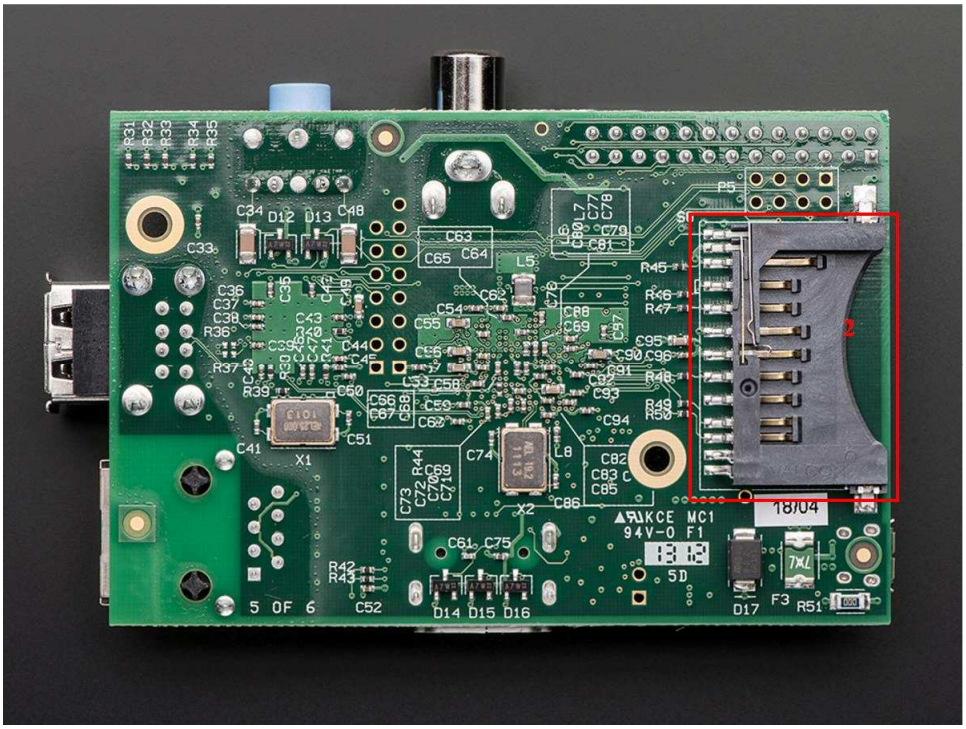

Рис. 2 - Внешний вид одноплатного компьютера *Raspberry Pi* (вид снизу)

На плате имеются:

- микросхема *BCM2835* (*SoC*) английской фирмы Broadcom и микросхема памяти размером 256 МБ или 512 МБ (в зависимости от модели) соединенные вместе по технологии *Package on package* (*PoP*) [3] (одна над другой, по типу бутерброда, при этом микросхема памяти расположена сверху), (обозначена «*1*» на *Рис*. 1);
- разъём для карт *Secure Digital* (*SD*), с позиционным обозначением на плате *S8*, предназначен для вставки карт с развернутой операционной системой («*2*» на *Рис*. 2);
- порт(ы) *USB 2.0*, имеющие позиционное обозначение *S7*, для подключения клавиатуры и / или мыши («*3*» на *Рис*. 1);

- порт *Ethernet* (наличие в зависимости от модели), имеющий позиционное обозначение *P4*, для подключения к локальной сети и доступу к *Internet* («*4*» на рисунке 1);
- разъём *HDMI*, имеющий позиционное обозначение *S3*, для подключения соответствующего компьютерного монитора («*5*» на *Рис*. 1);
- 5 индикаторных светодиода (зелёный *ACT (*Светится при обращении к *SD* карте*)*, красный *PWR (*светится при наличии питания 3,3 Вольта*)*, зелёный *FDX (*Светится если сетевой адаптер работает в полном дуплексном режиме*)*, зелёный *LNK (*Отображает активность сети*)*, жёлтый *100 (*Горит если сеть работает на скорости 100 *Мбит* в секунду*)*) («*6*» на *Рис*. 1);
- аналоговый аудио выход, имеющий позиционное обозначение *S6*, для подключения динамиков («*7*» на *Рис*. 1);
- композитный видеовыход (*RCA «тюльпан»*), обозначенный на плате *S4*, для подключения, например, к телевизору («*8*» на *Рис*. 1);
- разъём *micro USB*, обозначенный на плате *S1*, используемый в качестве разъёма питания (ВНИМАНИЕ! Использование этого разъёма как *USB* не предусмотрено, также недопустимо подключать эту плату к *USB* персонального компьютера, так как ток потребления может превысить допустимые 500 мА) («*9*» на *Рис*. 1);
- разъём основных портов ввода / вывода (*General Purpose I/O*) микросхемы *SoC* («*10*» на *Рис*. 1, обозначение *P1* на плате);
- разъём для подключения внешнего *LCD* дисплея, обозначенный на плате *S2*, по протоколу *DSI* [4] (11 на *Рис*. 1);
- разъём интерфейса *JTAG* для отладки работы микросхемы *LAN9512 (Ethernet* интерфейс*)*, обозначенный на плате *P3* («*12*» на *Рис*. 1);
- разъём интерфейса *JTAG* для отладки работы процессора (*CPU*) *ARM*, обозначенный на плате *P2* («*13*» на *Рис*. 1);
- разъём для подключения внешней камеры, обозначенный на плате *S5* («*14*» на *Рис*. 1);
- разъём расширение для разъёма *P1*, обозначенный на плате *P5* с нижней стороны платы, содержит выводы интерфейса I^2C и сигналы *Handshake* модуля *UART* (выводы *TX* и *RX*, расположены на разъёме *P1*) («*15*» на *Рис*. 1);
- разъём для подключения кнопки сброса, обозначенный на плате *P6*, («*16*» на *Рис*. 1).

На *Рис*. 1 также обозначены тестовые точки для измерения уровней сигнала +5 *В* и земли.

Все указанные выше обозначения соответствуют принципиальным схемам на компьютер [5], [6]. Платы также могут быть двух версий: *rev 1.0* и *rev 2.0*. Эти платы можно отличить друг от друга сравнив плату с фотографиями той или иной версии, приведенных в [7], так что будьте внимательны.

Микросхема *BCM2835* содержит в своем составе ядро *ARM1176JZF-S* [8] работающее на частоте 700 МГц и имеет расширение *Vector Floating Point* (*VFP*) (вектора чисел с плавающей запятой) [9]. Так как *BCM2835* является *SoC*, то помимо ядра *ARM*, имеется периферия [10]:

- Таймеры;
- Контроллер прерываний;
- *GPIO;*
- *USB*;
- *PCM / I2S*;
- *DMA* контроллер;
- *I2C* ведущий;
- *I2C / SPI* подчиненный;
- *SPI0*, *SPI1*, *SPI2*;
- *PWM*;
- *UART0*, *UART1*.

Дополнительно к центральном ядру добавлено *GPU* (*Graphics Processing Unit*) ядро *Broadcom VideoCore IV GPU* [11]. Оно поддерживает *OpenGL ES 1.1*, *OpenGL ES 2.0*, аппаратно ускоренный *OpenVG 1.1*, *Open EGL*, *OpenMAX* и *1080p30 H.264* высоко эффективное декодирование.

Также есть *DSP* (*Digital Signal Processing*) ядро, но информация по нему отсутствует в общем доступе.

Плата *Raspberry Pi* должна быть подключена к внешнему источнику питания 5 *В* (предпочтительно использовать зарядное устройство для смартфонов) с разъёмом *micro USB type B*. Потребляемая мощность от источника питания составляет 3,5 *Вт* (ток до 700 *мА*) для модели «*B*» и 1,5 *Вт* (ток до 300 *мА*) для модели «*A*». При подключении дополнительной периферии к плате нагрузка может возрасти, поэтому нужен запас по мощности источника питания.

Также для минимальной работы с компьютером понадобиться либо монитор *HDMI*, либо телевизор с композитным видеовходом *RCA «тюльпан»*, а также *USB* клавиатура для персональных компьютеров.

Подробную информацию по вопросам связанным с этим компьютером, на английском языке, можно найти на *Wiki* страничке *eLinux* [12], а также в других местах.

ПОДГОТОВКА ОБРАЗА ОПЕРАЦИОННОЙ СИСТЕМЫ

Любой компьютер не сможет работать без операционной системы. Сообщество *Raspberry Pi*

Foundation предоставляет набор подготовленных образов операционных систем на базе известных дистрибутивов *Linix* [13].

Большое распространение среди пользователей получил образ *Raspbian Debian Wheezy* (основан на компактной версии дистрибутива *Debian*). Для создания операционной системы этого образа потребуется загрузить его с сайта.

Образ предназначен для записи на *SD* карту. Процесс записи образа на карту хорошо описан на сайте сообщества [14]. Для записи Вам потребуется персональный компьютер с установленной одной из операционных систем *Linux*, *Windows* или *Mac OS*, а также картридер для *SD* карт.

Минимальный размер *SD* карты, который может быть использован для образа системы составляет 4 *ГБ*. Многие типы карт уже проверялись различными людьми на предмет совместимости с *Raspberry Pi* сводная таблица на этот счет находится по адресу [15]. Оптимальными являются карты 6, 8 или 10 класса.

После вставки карты в картридер необходимо убедиться, что она определилась как логический диск и видна системой. Вам может потребоваться форматирование. Это можно сделать при помощи стандартных механизмов Вашей операционной системы, при этом должна быть выбрана файловая система FAT32.

В качестве примера опишем копирование образа на карту в *ОС Windows*. Потребуется программа *Win32DiskImager* [16], при открытии которой (требуются права администратора) необходимо указать местоположение файла образа операционной системы на Вашем компьютере и логический диск Вашей *SD* карты. В итоге должно получиться так, как показано на *Рис*. 3.

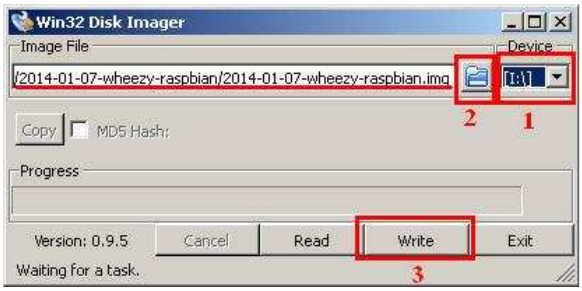

Рис. 3. Подготовка к записи образа ОС в программе *Win32 Disk Imager*

Процесс записи начнётся после нажатия кнопки «*Write*». Данный процесс достаточно длителен и может занять до получаса на среднем по производительности компьютере. После записи следует закрыть программу и извлечь карту из картридера.

ПЕРВОЕ ВКЛЮЧЕНИЕ И НАСТРОЙКА RASPBERRY PI

После того как *SD* карта готова, можно приступить к включению *Raspberry Pi*. Для этого Вам потребуется вставить карту в соответствующий разъем одноплатного компьютера, подключить клавиатуру и возможно мышь, подключить монитор (*RCA* или *HDMI* разъемы на выбор) и только затем подсоединить к источнику питания.

Raspberry Pi начнет загрузку. На экране будут мелькать командные строки. Автоматически (только при первом включении, при последующих включениях автоматический запуск отключен) должен запуститься скрипт *raspi-config* (*Рис*. 4).

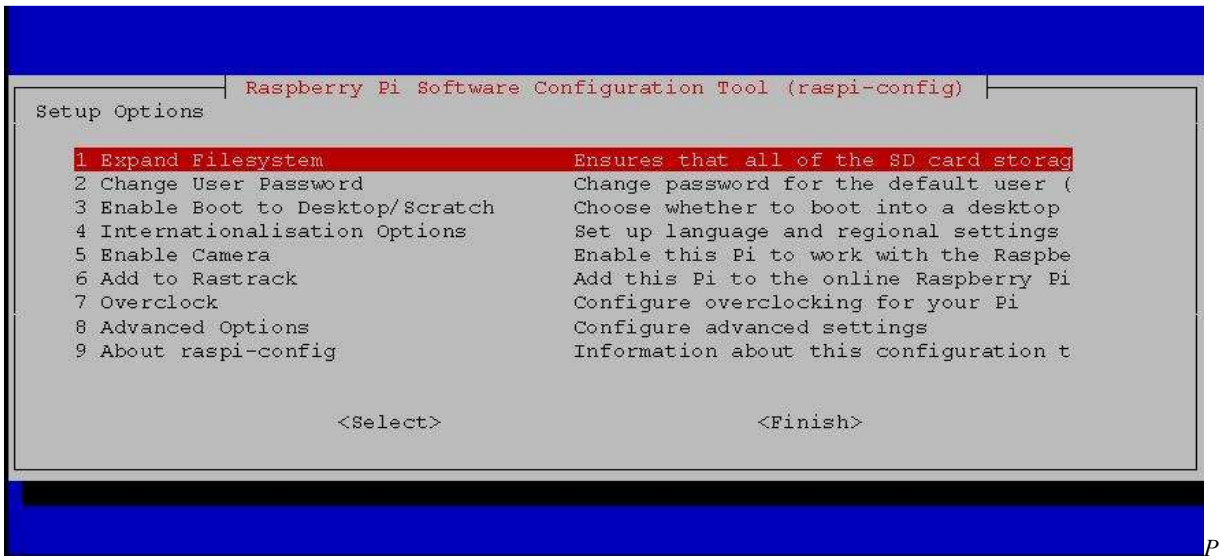

Рис. 4. Автоматический запуск скрипта *raspi-config*

Если запуск скрипта не произошел, то в командной строке приглашения следует набрать «*sudo saspi-config*». Если система спрашивает «*raspberry login*», а затем «*Password*», то необходимо ввести «*pi*» и «*rasspberry*» (соответственно является логином и паролем по

умолчанию).

Данный скрипт является обычным меню, навигация по которому осуществляется при помощи кнопок управления курсором клавиатуры. Опустим рассмотрение каждого пункта меню и сосредоточимся на основных.

Пункт «*Expand Filesystem*» позволяет расширить корневой раздел файловой системы на весь размер *SD* карты. Настоятельно рекомендуется выполнить данных пункт (выделить его и нажать кнопку *Enter*).

Пункт «*Change User Password*» позволяет изменить пароль по умолчанию «*raspberry*» на свой собственный. Потребуется ввести новый пароль два раза.

Пункт «*Enable Boot to Desctop/Scratch*» позволяет выбрать поведение при загрузки компьютера. При выборе этого пункта появляется новое меню с тремя вариантами:

- *Console Text console, requiring login (default);*
- *Desctop Log in as user 'pi' at the graphical desctop;*
- *Scratch Start the Scratch programming environment upon boot.*

Выбор первого пункта приведет к тому, что после загрузки система останется в режиме консоли и будет запрашивать логин и пароль пользователя. Выбор второго пункта приведет к тому, что система сама загрузит графическую оболочку (рабочий стол) со входом под пользователем «pi». Третий пункт позволяет загрузить среду программирования на языке *Scratch* вместо рабочего стола. Выберите наиболее подходящий Вам пункт.

Следующий пункт меню «*Internationalization Options*» распадается на три ветки, при его выборе:

- *Change Locale;*
- *Change Timezone;*
- *Change Keyboard Layout.*

Провести настройки нужно по всем трём веткам. Первый пункт (переводится как изменить локаль) необходим для того, чтобы в системе установить поддержку русского языка. В списке в окне на *Рис*. 5 показано, как следует поступить. Используя клавишу пробел на клавиатуре установить символ «*» напротив строки «*ru_RU.UTF8 UTF-8*».

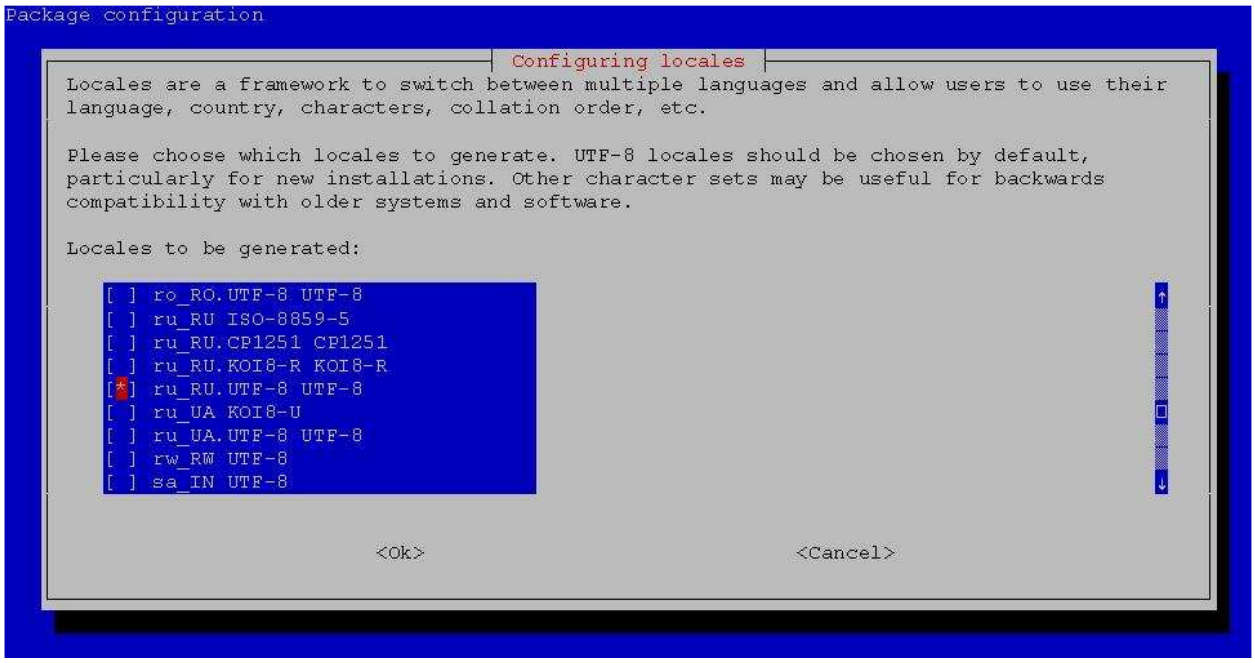

Рис. 5. Установка российского локаля на *Rasspberry Pi*

В списке можно заметить, что локаль «*en_GB.UTF-8 UTF-8*» для английского языка уже установлен. В итоге получится, что в списке будет выбрано два локаля «*ru_RU.UTF8 UTF-8*» и «*en_GB.UTF-8 UTF-8*». После нажатия на кнопку «*<Ok>*» появится новое меню, в котором необходимо выбрать локаль по умолчанию, все зависит от Ваших предпочтений. После выбора локоля и нажатии кнопки *Enter* система создаст новые локали и настроит их.

После настройки локалей переходим к настройке временной зоны (пункт «*Change Timezone*»). В первом окне выбираем географическую зону, в которой находится Ваш город, для России справедливо всего два варианта: «*Europa*» и «*Asia*». После нажатия кнопки *Enter*, выберите Ваш город из списка. В этом случае компьютер запросит из интернета текущее время. У *Raspberry Pi* нет часов реального времени, информация берется из интернета.

И, наконец, последний пункт «*Change Keyboard Layout*». На первом экране позволяет из списка выбрать драйвер для используемой Вами клавиатуры. На втором: раскладку клавиатуры (обычно раскладка соответствует «*Russian*»). На

третьем предлагается выбрать из списка метод переключения (сочетание клавиш) между национальной раскладкой и латиницей. Затем в следующих трех экранах меню рекомендуется выбрать пункты по умолчанию. В последнем пункте, на вопрос «*Use Control+Alt+Backspace to terminate the X server*» следует ответить «*<No>*».

Следующий интересующий нас пункт это - «*Advanced Options*» данный пункт распадается на 7 подменю:

- *A1 Overscan;*
- *A2 Hostname;*
- *A3 Memory Split;*
- *A4 SSH;*
- *A5 SPI;*
- *A6 Audio;*
- *A7 Update*.

Но нас будет интересовать всего два пункта: «*A3 Memory Split*» и «*A4 SSH*». Первый позволяет перераспределить оперативную память между главным процессором и видео ядром. При выборе этого пункта следует указать размер памяти отдаваемой видео ядру. Допустимое числа: *16, 32, 64, 128* и *256* МБ. Пункт *SSH* позволяет включить *SSH* сервер для удалённого управления операционной системой (удалённый доступ к консоли операционной системы). Понадобиться в дальнейшем для настройки кросс-платформенной среды разработки.

После всех манипуляций появится главное меню скрипта *raspi-config* в котором необходимо выбрать пункт «*<Finish>*», для принятия всех внесённых изменений. На вопрос «*Would you like to reboot now?*» следует ответить положительно. Затем одноплатный компьютер перезагрузится. Если Вы, в чем то ошиблись, то исправить это можно всегда вводом команды «*sudo saspi-config*» в консоли ОС или программе терминале.

После перезагрузки система загрузится по тому варианту, который был указан в пункте «*Enable Boot to Desctop/Scratch*». Если Вы выбрали пункт «*Console Text console, requiring login (default)*» (является пунктом по умолчанию), то загрузить графический рабочий стол можно из командной строки вводом команды «*startx*».

В режиме консоли можно использовать известные в *Linux* команды. Для того, чтобы перезагрузить компьютер нужно ввести команду «*sudo shutdown -r now*», выключить - «*sudo shutdown -h now*», обновить список репозитариев дистрибутивов установленного ПО - «*sudo apt-get update*», обновить всё установленное ПО до последних версий - «*sudo apt-get upgrade*». Для того, чтобы установить какое-либо новое *ПО* необходимо ввести «*sudo apt-get install*» с указанием имени интересующего *ПО*. Так как *ОС Linux* изначально развивался как консольная операционная система, то используемый перечень всех команд весьма обширен. Описание всех команд выходит за пределы данной статьи. Здесь лишь приведен очень краткий перечень из них. Если Вы планируете использовать консоль в работе компьютером, то рекомендуется прочитать соответствующую литературу по консольным командам *Linux*.

В режиме графического рабочего стола всё является более менее понятным, чем с консолью. Среди прочих ярлыков, на рабочем столе, хочется отметить «*LXTerminal*» (доступ к командной строке *Linux*), «*Shutdown*» (выключение *Raspberry Pi*) и «*Справочник по Debian*» (полезная информация по операционной системе). Доступ к другим установленным программ организуется из меню, по аналогии с *ОС Windows* нажатием на иконку в левом нижнем углу экрана как показано на *Рис*. 6.

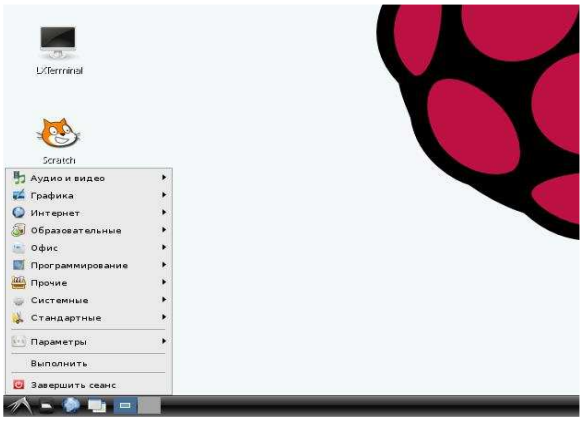

Рис. 6. Аналог меню «Пуск» в *Raspberry Pi*

Все программы сгруппированы по категориям: «*Аудио и видео*», «*Графика*» «*Интернет*» и т.д.

УСТАНОВКА БИБЛИОТЕКИ QT НА RASPBERRY PI

В *ОС Linux* весь графический интерфейс базируется на системе *X Window System* [17]. Для неё разработаны библиотеки: *GTK+* [18], *Qt* [19] *wxWidgets* [20], *Juce* [21], *FLTK* [22], которые содержат все необходимые классы для написания графического интерфейса пользователя. В настоящее время широкое распространение получили две библиотеки: *GTK+* и *Qt*. *GTK+* в большей степени ориентирована на системы *Linux*, в то время как *Qt* поддерживает другие *ОС*. С *Qt*, при переходе на иную операционную систему, нет необходимости изменять код, достаточно просто откомпилировать его под соответствующую систему.

Для того, чтобы использовать библиотеку *Qt* на *Raspberry Pi* потребуется установить саму библиотеку *Qt*, а также среду разработки, где будут писаться программы для неё.

Установка библиотеки *Qt* на *Raspberry Pi* выполняется из командной строки (в режиме графического рабочего стола необходимо запустить программу терминал – *LXTerminal*) [23]. Для этого в приглашении командной строки следует ввести «*sudo apt-get install qt4-dev-tools*» и нажать клавишу *Enter*. В ходе установки,

возможно, потребуется дополнительное подтверждение по установки (трудностей возникнуть не должно). По окончании установки, в новом приглашении командной строки, следует ввести новую команду «*sudo apt-get install qtcreator*», которая установит среду разработки *Qt Creator* (рекомендованная разработчиками библиотеки *Qt*). Ход установки аналогичен предыдущему шагу и не вызывает затруднений.

После всех установок, в главном меню графического рабочего стола, в пункте «*Программирование*», должен появится ряд значков связанных с установленной библиотекой *Qt* и средой *Qt Creator*, *Рис*. 7.

Рис. 7. Запуск *Qt Creator*

Выбор пункта «*Qt Creator*» приведёт к запуску среды разработки, внешний вид которой приведен на *Рис*. 8.

По умолчанию, среда настроена на компиляцию кода для удалённого встраиваемого устройства, а не для программирования для данного компьютера. Этот вариант не наш случай, поэтому нужно произвести изменения в настройках.

Сначала необходимо указать набор компиляции (*toolchain*), для этого вызываем пункт главного меню *Qt Creator – Tools → Options...*, выбираем справа раздел «*Build & Run*» и затем выбираем вкладку «*Tool Chains*». Нажимаем кнопку *Add* (справа) и далее отмечаем пункт *GCC*. В списке, в разделе «*Manual*» должен появится новый пункт «*GCC*». Затем нужно указать: местоположение набора компиляции, («*/usr/bin/arm-linux-gnueabihf-gcc-4.6*»), местоположение отладчика («*/usr/bin/gdb*») и тип *Mkspec*, («*default*»). В результате всех изменений должно быть так, как приведено на *Рис*. 9.

Вслед за настройкой *toolchain* необходимо отключить компиляцию кода для удаленных встраиваемых систем, и перевести в режим компиляции для текущей платформы. Для этого, сначала, необходимо выполнить пункт главного меню *Help → About Plugins...*, где необходимо отключить подпункт «*RemoteLinux*» пункта «*Device Support*». Затем нужно перезапустить *Qt Creator*. После того, как откроется окно программы необходимо выполнить пункт главного меню *Tools → Options...* Затем, убедившись, что справа выбран раздел «*Build & Run*», перейти на вкладку «*Qt Versions*». Кнопкой *Add*, справа от списка, добавить новое расположение утилиты *qmake*, в данном случае должен быть задан путь «*/usr/bin/qmake-qt4*». Желательно удалить другие пути из ветки *Manual*, чтобы избежать конфликтов.

После этого, среда готова к компиляции кода для данного компьютера. Покажем, на примере компиляции тестового проекта, как работать со средой.

Создадим тестовый проект. На стартовой странице, в разделе «*Welcome*», необходимо выбрать пункт «*Develop*» (смотрите *Рис*. 8) и далее пункт «*Create Project*». Откроется окно выбора типа нового проекта. В правом окне *Projects* следует выбрать пункт *Applications*, а в левом: *Qt Gui Aplication*. Дальше запуститься мастер создания проекта выбранного типа. На первом шаге задается имя проекта (например, *BasicQtProject*) и его местоположения (проблем не должно возникнуть). На следующем шаге должна быть отмечена конфигурация целевой платформы: галочка напротив «*Desktop*», «*Create build configurations:*» - «*For One Qt Version One Debug And One Release*», отметить оба пункта *Qt 4.8.2 (System) Release* и *Qt 4.8.2 (System) Debug*. Затем, следует нажать два раза *Next* и один раз *Finish*. Будет создан тестовый проект.

Для его компиляции и запуска, в *Release* версии, необходимо выполнить пункт главного меню «*Build*» → «*Run*» или нажать сочетания клавиш *Ctrl + R*. В результате этого произойдет запуск пустого окна программы.

Для компиляции и запуска программы в *Debug* версии необходимо выполнить пункт главного меню «*Debug*» → «*Start Debugging*» → «*Start Debugging*» или нажать кнопку *F5* на клавиатуре. Если требуется поставить точку остановка (*Breakpoint*), то достаточно щелкнуть два раза на поле справа от интересующего места.

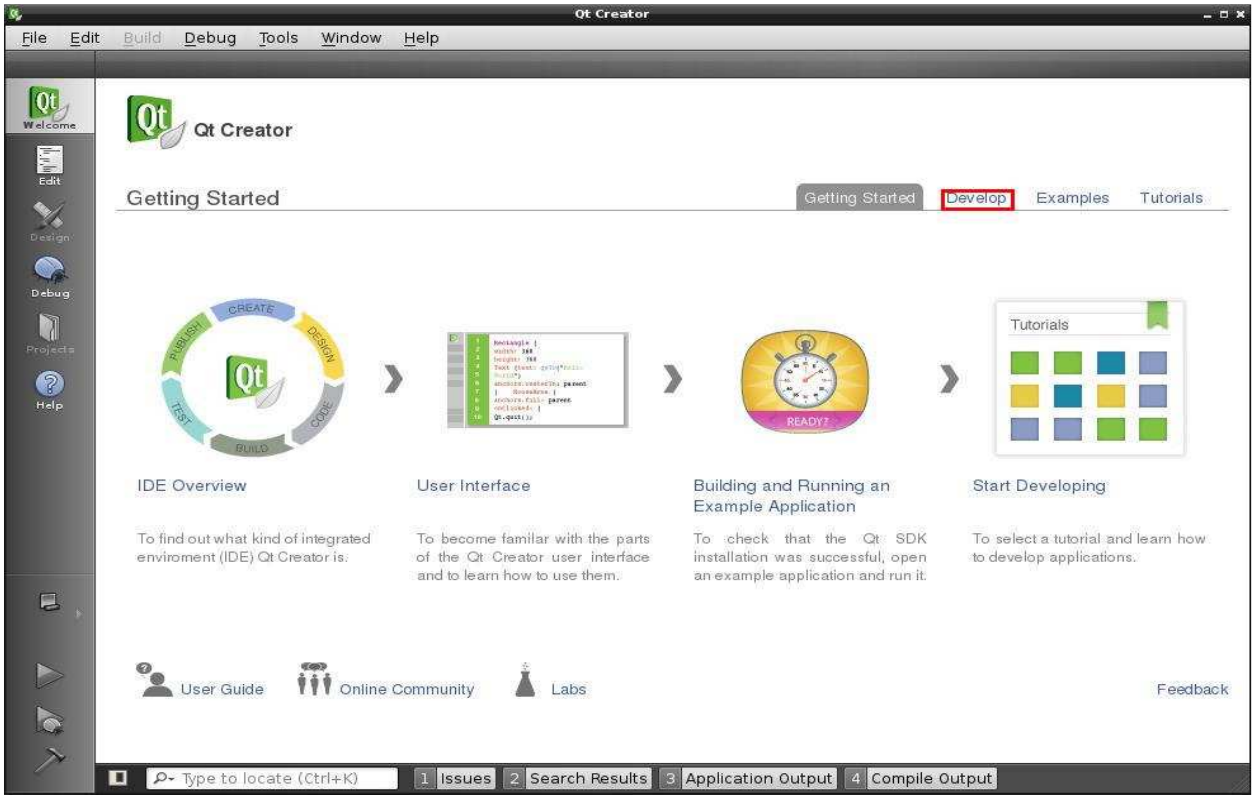

Рис. 8. Внешний вид среды разработки *Qt Creator*

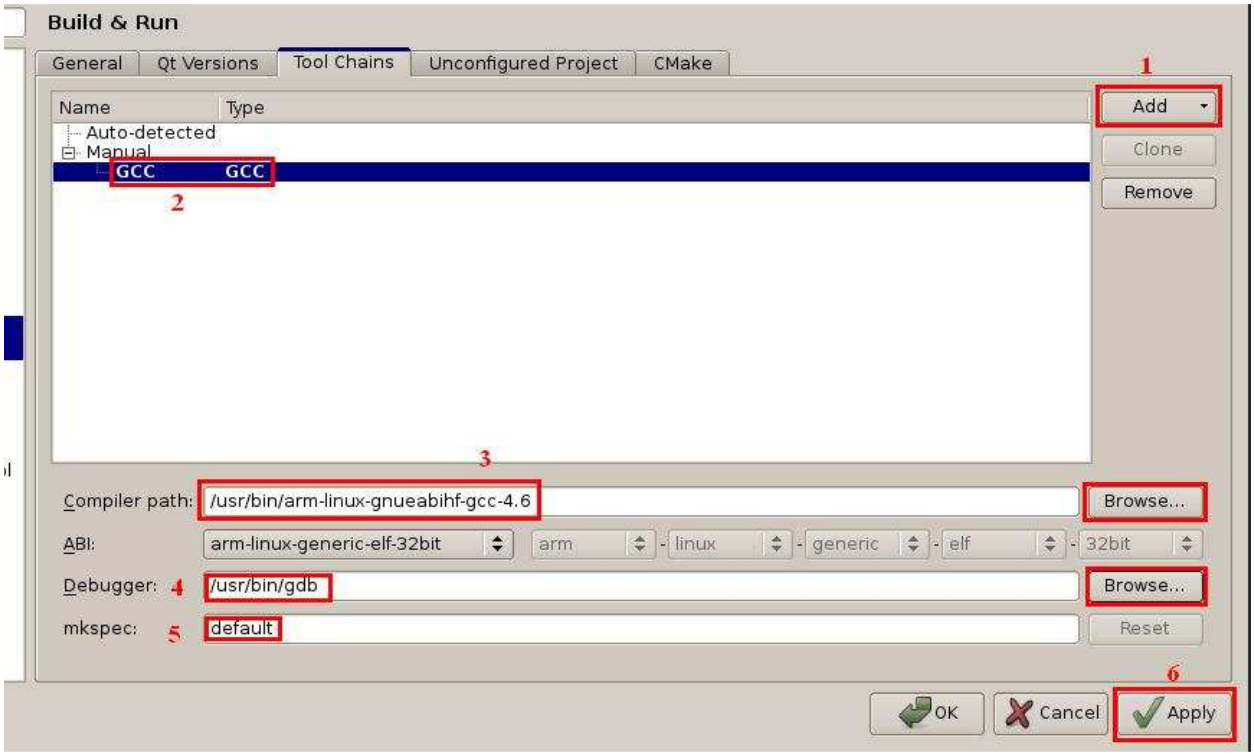

Рис. 9. Настройка набора компиляции в *Qt Creator*

КРОСС КОМПИЛЯЦИЯ ПРОЕКТОВ QT ДЛЯ RASPBERRY PI

Рассмотренный выше способ компиляции и отладки имеет один недостаток: длительное выполнение компиляции громоздких проектов.

Решением этой проблемы может стать кросс-компиляция. Этот способ предполагает, что вся работа по компиляции и написанию кода происходит на другом, более мощном и быстром компьютере. Но данный способ на порядок сложнее в реализации.

Введем несколько терминов для простоты изложения последующего материала: «*Целевая*

платформа» - устройство, на котором будет выполняться подготовленная программа, в данном случае подразумевается одноплатный компьютер *Raspberry Pi*, «*Host компьютер*» - *IBM PC* совместимый персональный компьютер под управлением *ОС Windows*.

В глобальной паутине, по вопросу кросс-компиляции для *ОС Linux* имеется множество решений и они достаточно хорошо описаны [24] [25], но в случае установленной *ОС Windows* на *host компьютере* информация практически отсутствует. Данная статья, среди прочего, посвящена этому вопросу.

Не смотря на скудность информации, всё же можно найти более или менее подходящие варианты. Так, например, в [26] и [27] приведена методика настройки *ПО* с использованием для написания кода интегрированных сред разработки (*IDE*), а в [28] приведена методика без использования *IDE*. В результате объединения этих трех вариантов получился метод настройки *ПО*, который будет описан далее. Этот спсоб лишен недостатков каждого из трех и сокращает длительность процесса установки.

Перед сборкой необходимо, чтобы на *Raspberry pi* уже была установлена библиотека *Qt* (смотрите предыдущий пункт). Также необходимо определить версию этой библиотеки. В нашем случае версия – 4.8.2.

Сборку *ПО* начнём с установки кросс компилятора. На сайте компании *Sysprogs UG* [29] имеется уже собранный набор кросс-компилятора для *ОС Windows*. Лучше выбрать универсальную версию *toolchain*, без *sysroot*. После запуска, загруженного с сайта файла, достаточно всего лишь, установить галочку «*I accept the terms of the license agreement*», как на *Рис*. 10.

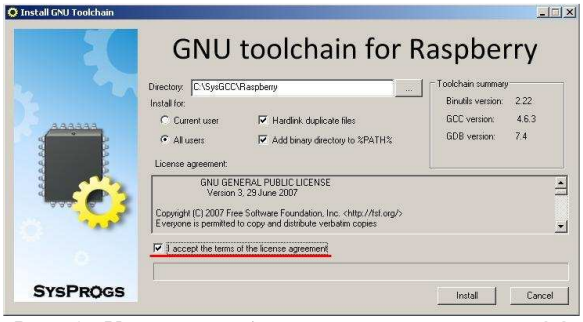

Рис. 10 - Установка набора кросс компилятора для ОС *Windows*

Папку установки, предложенную по умолчанию, можно поменять на другую. В дальнейшем будем исходить из того, что *toolchain* установлен в папку по умолчания. В случае успеха установки, появится соответствующее сообщение.

Если бы на *host компьютере* был установлен *Linux*, то набор компиляции для *Raspberry pi* можно было бы получить самостоятельно, воспользовавшись, например, такими инструментами как *crosstool-ng* [30], *buildroot* [31] и т. п. Этот способ потребовал бы обширных знаний, терпения и упорства.

В случае установленной *ОС Windows*, есть ещё два способа получения кросс компилятора [32] [33]. Оба способа сложны и заключаются в самостоятельной сборке и компиляции.

Для того, чтобы можно было компилировать код для отличной от используемой Вами операционной системы, компилятору нужны все те библиотеки, которые имеются на *целевой платформе*. В *ОС Linux* все эти библиотеки расположены в файловой системе по фиксированным путям и все они вместе называются *sysroot*. Компилятору необходимо чтобы *sysroot Raspberry pi* был скопирован на *host компьютер* в определенное место. Разработчики из компании *Sysprogs UG* предусмотрели это подготовив файл *C:\SysGCC\Raspberry\TOOLS\UpdateSysroot.bat* Запустив этот файл можно скопировать весь *sysroot*.

Окно «*Synchronize sysroot*» (*Рис*. 11) позволяет произвести необходимые настройки для копирования.

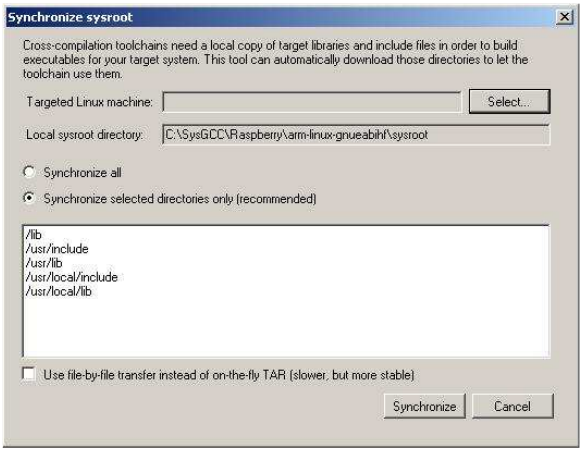

Рис. 11. Окно *Synchronize sysroot*

Необходимо, чтобы *целевая платформа* и *host компьютер* были подключены к одной локальной сети. В поле «*Target Linux machine:*» следует указать *целевую платформу* (*Raspberry pi*). Кнопка «*Select...*» позволяет задать необходимый *IP* адрес *Raspberry pi* (понадобиться включенный на *Raspberry pi*, при помощи скрипта *raspi-config*, *SSH* сервер, а также знание *IP* адреса *Raspberry pi* (ввод в консоли *целевой платформы* команды «*ifconfig*»)).

Нажатие этой кнопки приведёт к созданию нового соединения по протоколу *SSH* [34] как приведено на *Рис*. 12.

Необходимо задать: установленный *IP* адрес одноплатного компьютера (в данном случае *192.168.0.11*), имя пользователя – *pi* и используемый пароль. Кнопка *Connect* попытается соединиться с *Raspberry pi*. В случае успеха Вам предложат сохранить ключ к этому соединению для упрощения дальнейшей работы. Рекомендуется сохранить его на *host компьютере*.

Рис. 12.Установка соединения с целевой платформой по *SSH* протоколу

После этого программа вернется к окну *Рис*. 11, где теперь можно нажать кнопку «*Synchronize*». В результате чего с целевой платформы будет скопирован весь *sysroot*.

Кроме *sysroot* для создания графического интерфейса потребуются некоторые утилиты из состава *Qt* для *Windows*, но проще установить библиотеку целиком.

На сайте разработчиков *Qt* можно найти, в свободной загрузке, откомпилированную библиотеку необходимой нам версии. Важно чтобы версия *Qt* на *целевой платформе* совпадала с установленной на *host компьютере*. Архив версий библиотеки *Qt* можно найти по ссылке [35]. Каждая версия поддерживает различные платформы. Нам необходима версия для *MinGW*.

MinGW - это компилятор (портированный на *ОС Windows GCC*) позволяющий работать теми же методами как и на *ОС Linux* [36]. Перед тем как устанавливать библиотеку *Qt* необходимо загрузить *MinGW*, пройдя по ссылке [37]. Далее необходимо установить его на операционную систему следуя инструкциям мастера. После установки необходимо убедиться, что в переменой среды *PATH ОС Windows* установлен путь до папки *bin* каталога с *MinGW*. По умолчанию: *C;\MinGW\bin*.

Затем можно приступать к установке *Qt* .При установке необходимо указать путь до директории с *MinGW*. Также может возникнуть предупреждение аналогичное изображенному на *Рис*. 13. На него не следует обращать внимание (выбрать «*Yes*»).

Рис. 13. Предупреждение о несовпадении версий *MinGW*

Более подробно процесс установки библиотеки *Qt* описан в [38].

Дальнейшую сборку проведем по описанию [28]. На этой страничке предлагается программное средство *QT Configurator*, работа которого сводится к обновлению уже установленного *sysroot*, а также копированию дополнительных четырёх файлов, необходимых для интеграции компилятора с библиотекой *Qt*. После загрузки *QT Configurator*, по рекомендуемой ссылке, необходимо указать местоположение утилиты *qmake*, из состава библиотеки *Qt*, местоположение установленного на предыдущем шаге кросс компилятора *GCC* и подключиться к целевой платформе известным способом. Затем программа выполнит обновление установленного *sysroot* (дополнительно будет скопирована папка *mkspec* в директорию «*C:\SysGCC\Raspberry\arm-linux-gnueabihf\sysroot\usr\share\qt4*»).

В эту директорию будет добавлена папка *arm-linux-gnueabihf* с думя файлами: *qmake.conf* и *qplatformdefs.h*. Этих файлов нет на *целевой платформе*. Оба файла являются текстовыми. После работы *QT Configurator* необходимо исправить файл *qmake.conf*. В нём, в строчке «*QMAKE_LIBDIR_QT = C:\SysGCC\Raspberry\arm-linux-gnueabihf\sysroot\usr\lib*», необходимо поменять символы «\» на «/».

В конце работы *QT Configurator* выдаст сообщение с рекомендациями как компилировать проекты при помощи консоли *Windows*. На страничке дана ссылка на архив с тестовым проектом. Можно попробовать откомпилировать проект и даже выполнить его на целевой платформе.

Такой способ не совсем удобен для повседневной работы с программой, так как нет удобного редактора, подсветки синтаксиса и прочих удобных вещей современных интегрированных сред разработки (*IDE*).

IDE ДЛЯ ЗАДАЧ КРОСС КОМПИЛЯЦИЯ ПРОЕКТОВ QT

Рассмотренный ранее *Qt Creator* хоть и является рекомендуемой средой разработки, но в старых версиях поставляется отдельно от библиотеки. Это позволяет использовать отличную, от предлагаемой, среду разработки.

Более удобной и с большими возможностями, по мнению автора, является среда *Eclipse* [39].

Описание [27] содержит полный цикл установки среды на компьютер с *ОС Windows*, но описание сборки кросс-компилятора на порядок сложнее в реализации чем приведенный выше способ. Справедливости ради надо сказать, что настройка *Eclipse* в этом описании удалась простой. Далее будет кратко описана методика настройки *Eclipse* по аналогии с этим способом.

Пропустим установку самого *Eclipse* и *Java Runtime Environment*, она подробно описана в рассматриваемом описании. Дальнейшее описание будет посвящена версии *Eclipse Helios* (версия 3.6).

Рекомендуется создать папку проектов (*workspace Eclipse*) и ярлык на рабочем столе для запуска *Eclipse*. В *workspace* необходимо распаковать рекомендуемый в [28] архив тестового проекта (все файлы проекта должны быть размещены в одной папке с названием проекта). При запуске *IDE* следует указать созданную папку *workspace*. Далее необходимо выполнить пункт главного меню «*Help → Install New Software...*». Выбрать из выпадающего списка «*Work with:*» пункт «*--All available sites--*». Прокрутив вниз содержимое списка, расположенного ниже, следует раскрыть пункт «*Mobile and Device Development*» (значок «+» рядом с название). В раскрывшемся списке следует поставить галочки напротив: «*C/C++ Remote Launch*», «*Remote System Explorer End-User Runtime*», «*Remote System Explorer User Actions*» и «*C/C++ GCC Cross Compiler Support*». . Последний пункт следует выбрать в пункте «*GNU ARM C/C++ Cross Development Tools*».

Далее необходимо установить выбранные плагины — процесс достаточно простой. После установки необходимо перезапустить *Eclipse*.

Дальнейшим шагом станет установка соединения с *Raspberry pi* по протоколу *SSH*. Для этого следует запустить утилиту *SmarTTY.exe* в каталоге установленного компилятора (путь по умолчанию: *C:\SysGCC\Raspberry\TOOLS\PortableSmartty*). После подключения к консоли *Linux* ввести поочередно команды:

- *mkdir remote-debugging;*
- *cd remote-debugging;*
- *touch .gdbinit.*

Таким образом, в каталоге */home/pi целевой платформы* будет создана папка *remote-debugging*, в которой будет помещен скрытый файл *.gdbinit.*

Не закрывая окно утилиты *SmarTTY.exe*, ввести в командную строку ещё две строчки: «*cp /usr/bin/gdbserver ./*» и «*chmod +x gdbserver*». Тем самым копируется файл *gdbserver* в созданную на предыдущем пункте папку и задается этому файлу атрибут исполняемого файла. Теперь утилита *SmarTTY.exe* больше нам не понадобиться — можно её закрыть или ввести в консоли команду «exit».

Вернемся к открытому окну *IDE Eclipse*. Выполним пункт главного окна *File → New → Makefile Project with Existing Code*. В открывшемся окне «*Import Existing Code*» в поле «*Project Name*» необходимо ввести имя распакованной в *workspace* папки тестового проекта. В нашем случае это *BasicQtProject*. В поле «*Existing Code Location*», при помощи кнопки «*Browse...*», указать местоположение папки проекта. В списке «*Toolchain for Indexer Setting*» оставить пункт «*<none>*». Должно получиться как на *Рис*. 14.

После этих манипуляция в окне *Eclipse* «*Project Explorer*» появится каталог не настроенного, пока, тестового проекта. Приступим к его настройке.

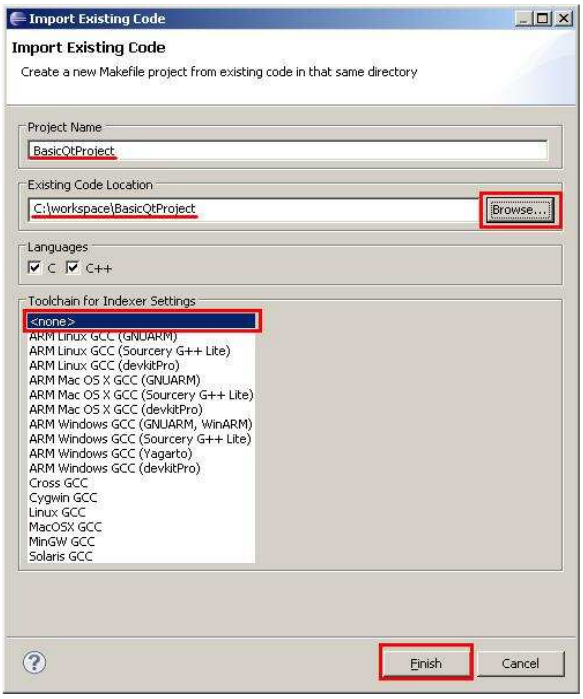

Рис. 14. Создание тестового проекта *в IDE Eclipse*

Первым делом, сообщим проекту местоположение заголовочных файлов библиотеки *Qt* скопированных с целевой платформы (часть *sysroot*). Для этого выделим папку проекта в *Project Explorer* и затем выполним пункт главного меню *Project → Properties*. В появившемся окне диалога нужно перейти к разделу «*C/C++ General*» списка слева. На вкладке «*Includes*», в списке «*Languages*» выбрать пункт «*GNU C++*» и добавить (при помощи кнопки «*Add...*» справа) указанные на рисунке 15 пути. Пути приведены для установки кросс компилятора в папку по умолчанию.

Для принятия всех изменений нажмите кнопку «*Apply*».

Затем создадим точку вызова утилиты *qmake.exe* из состава кросс-компилятора. Для этого, не закрывая окно свойств проекта, перейдем к пункту *Builders*, как приведено на *Рис*. 15. Кнопка *New...* (справа) позволяет создать точку автоматического запуска консольных программ, во время построения проекта. Воспользовавшись ею и нажав *OK* в следующем окне, переходим к окну «*Edit Configuration*». На вкладке *Main* следует заполнить все так, как показано на *Рис*. 16. На рисунке приведена конфигурация для отладочной версии программ. Если стоит задача подготовить *Release* версию программы, то в поле «*Arguments:*» следует опустить написание строчек вида «*CONFIG+=...*». Для ускорения написания строчек можно воспользоваться кнопкой «*Variables...*», которая позволяет вставить символы вид «*${...}*». Данная настройка полагается на то, что проект уже имеет в своем составе файл с расширение «*.pro*».

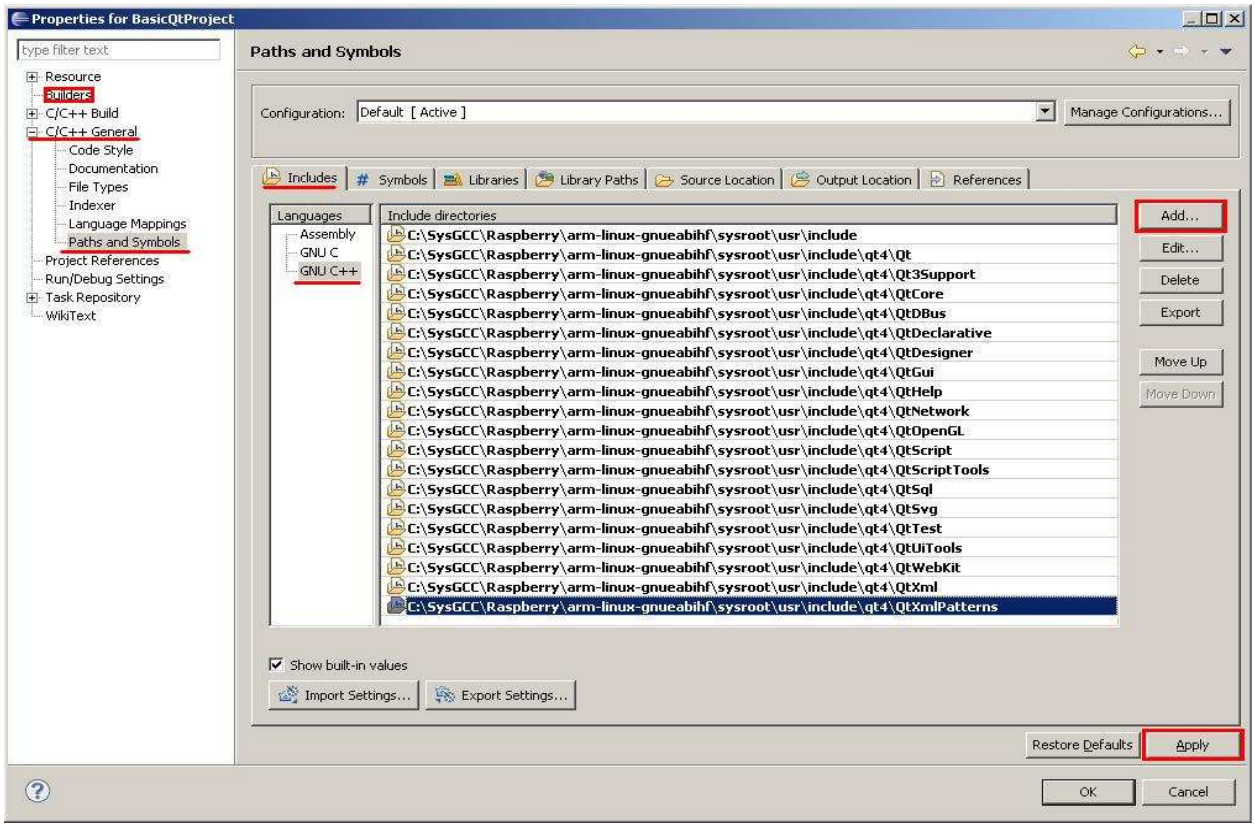

Рис. 15. Задание путей к заголовочным файлам библиотеки *Qt*

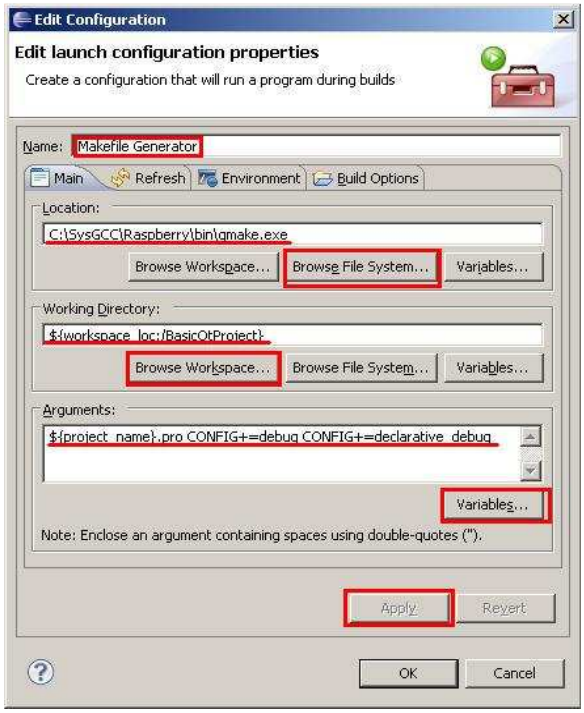

Рис. 16. Создание точки вызова утилиты *qmake*. Вкладка *Main*

Следующая вкладка «*Environment*» позволяет создать переменную окружения, для корректной работы библиотеки *Qt*. Необходимо воспользоваться кнопкой «*New...*» для создания новой переменной.

Имя этой переменной должно быть «*QMAKESPEC*», а значение «*C:/SysGCC/Raspberry/arm-linux-gnueabihf/sysroot/usr/share/qt4/mkspecs/arm-linux-gnueabihf*» для установки кросс-компилятора в папку по умолчанию. Результат приведен на *Рис*. 17. На этом конфигурация точки вызова утилиты *qmake.exe* закончена, можно нажать кнопку *OK*.

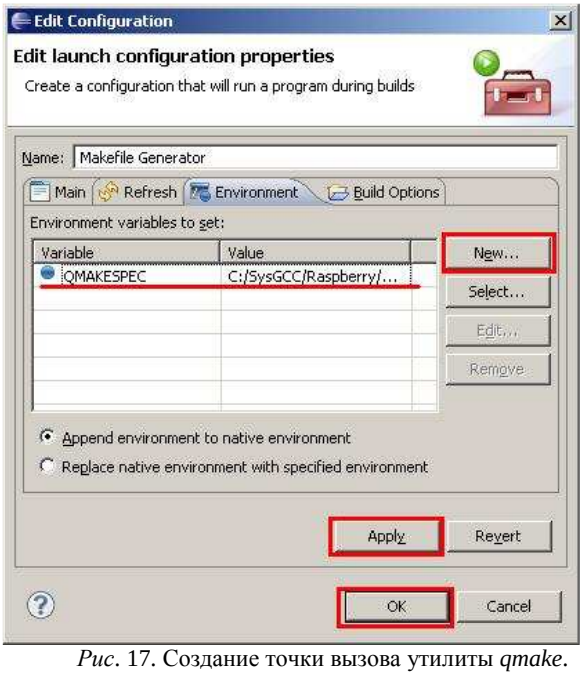

Рис. 17. Создание точки вызова утилиты *qmake*. Вкладка *Environment*

В списке «*Builders*» окна свойств проекта созданная точка вызова «*Makefile Generator*» должна быть самой первой. Осуществить это можно при помощи кнопки «*Up*», результат приведен на *Рис*. 18.

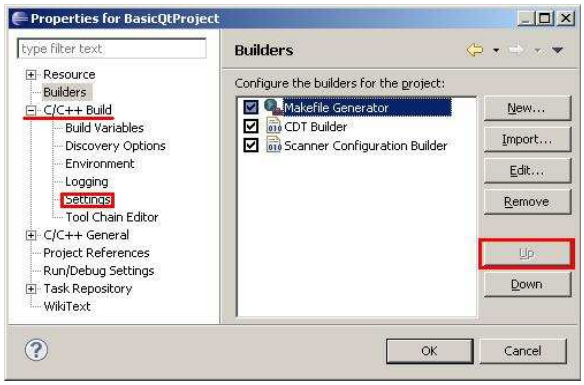

*Рис.*18. Список *Builders*

Следующим объектом настройки свойств проекта станет раздел «*C/C++ Build*». При его выделении, в правой части окна, следует выбрать вкладку «*Builder Settings*». В поле «*Build command:*» следует вписать «*C:\SysGCC\Raspberry\bin\make.exe*», в случае установки кросс-компилятора в папку по умолчанию.

Далее в разделе «*C/C++ Build*» будет необходимо подкорректировать пункт «*Settings*». После выбора этого пункта следует поставить галочку напротив «*Elf Parser*» в списке «*Binary Parser*» на одноименной вкладке. На этом настройка свойств проекта закончена, можно нажать кнопку *OK*.

Теперь доступна компиляция проекта. Выполнение пункта «*Build Project*» контекстного меню папки проекта в «*Project Explorer*» позволяет выполнить её. При удачном исходе, на вкладке «*Concole*» можно будет наблюдать ход процесса компиляции. Если ошибок нет, то можно приступить к дальнейшей настройке.

Организуем автоматизацию подключения по *SSH* протоколу к целевой платформе. Необходимо, чтобы в *Eclipse* были установлены описанные выше плагины. Всё управление этим подключением организуется через специальную перспективу. Чтобы её открыть выполните пункт главного меню «*Window*» → «*Open Perspective*» → «*Other...*». В открывшемся списке следует выбрать пункт «*Remote System Explorer*». Откроется новая перспектива.

Создадим файл подключения по протоколу *SSH*. Нужно выполнить пункт главного меню «*File*» → «*New*» → «*Other...*». В новом окне следует раскрыть раздел «*Remote System Explorer*» и выбрать там пункт «*Connection*». Затем нажать на кнопке «*Next*» внизу окна и выбрать в следующем окне операционную систему *Linux*. В следующем окне (нажать кнопку «*Next*») необходимо ввести *IP* адрес *целевой платформы*, так как показано на рисунке 19.(*IP* адрес нужно ввести свой).

Далее нужно поступить так, как показано на *Рис*. 20 и 21.

Далее нужно проверить правильность имени пользователя (*Login*) для доступа по *SSH*.

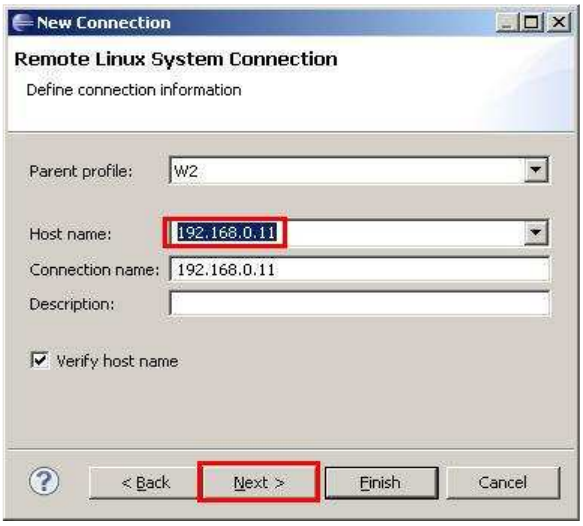

Рис. 19. Задание *IP* адреса целевой платформы

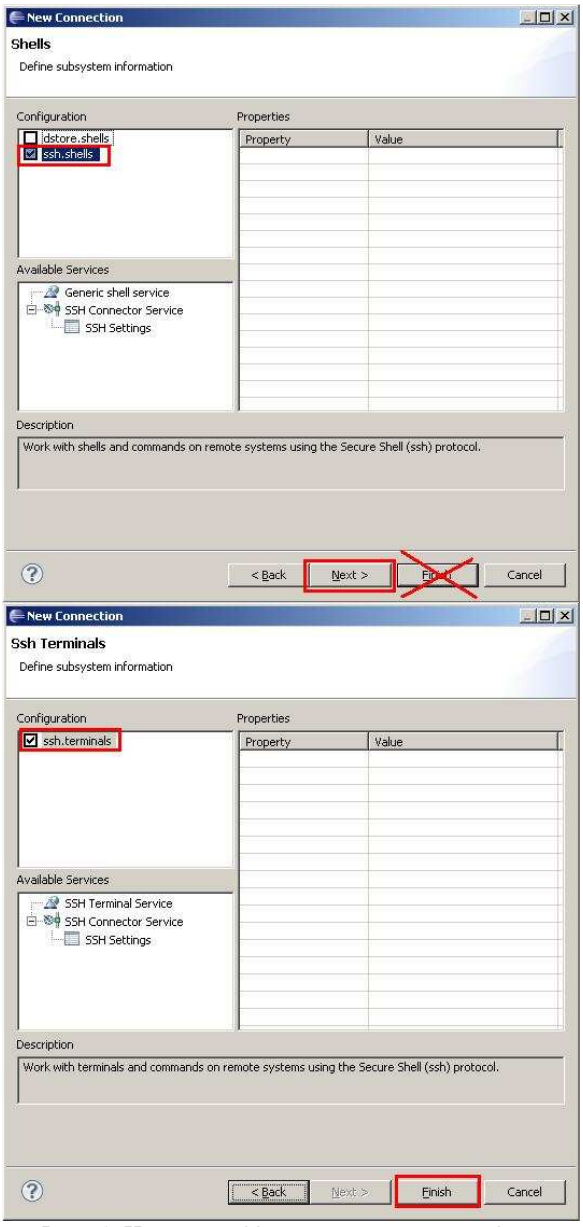

Рис. 20. Настройка *SSH* подключения, шаги 3 и 4

Возвратившись в перспективу «*Remote System Explorer*», на окне «*Remote Systems*» следует

выбрать появившийся раздел с именем подключения эквивалентном *IP* адресу *целевой платформы*. В его контекстном меню выбрать пункт «*Properties*» и в открывшемся диалоге «*Properties for ...*» выбрать в списке слева раздел «*Host*». Изменить настройки в соответствии с *Рис*. 22.

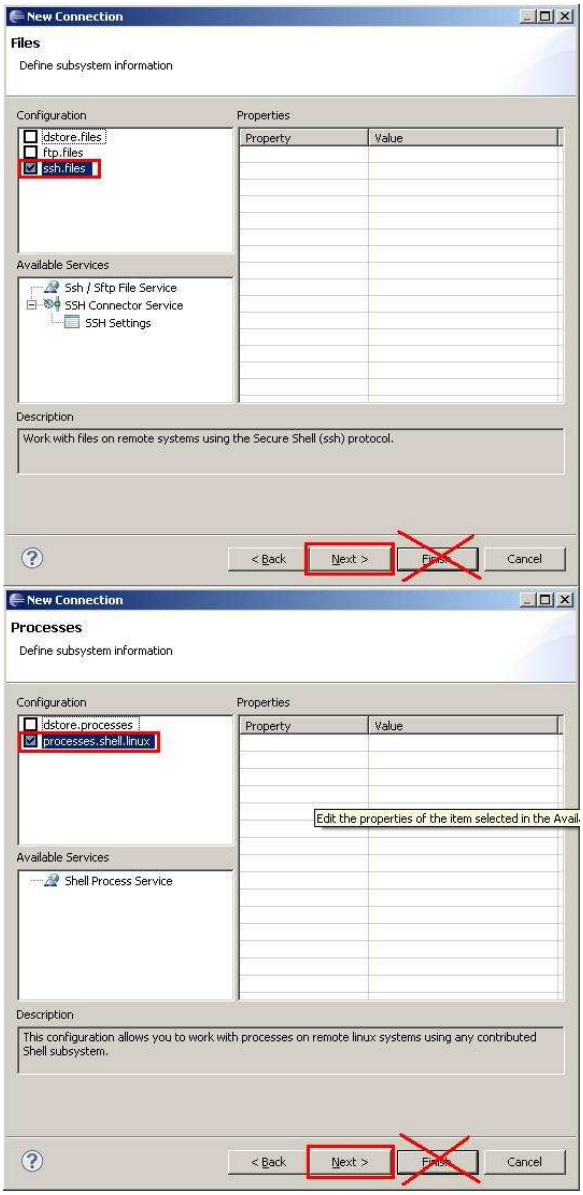

Рис. 21. Настройка *SSH* подключения, шаги 1 и 2

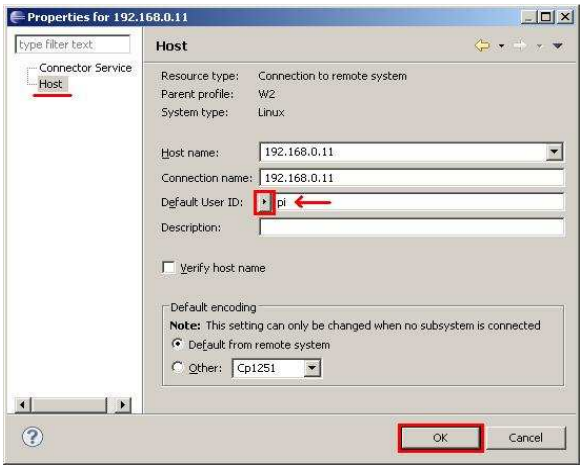

Рис. 22. Задание логина подключения *SSH*

Возвратимся в перспективу «*C/C++*». Создадим пустой текстовый файл, для чего вызовем контекстное меню папки проекта в «*Project Explorer*». В контекстном меню нужно выбрать пункт «*New*» → «*Other...*». В списке появившегося диалога «*New*» раскрыть раздел «*General*», выбрать там «*Untitled Text File*» и нажать на кнопке «*Finish*». В пустом открывшемся файле ввести строку «*set sysroot C:\SysGCC\Raspberry\arm-linux-gnueabihf\sysroot*». И затем сохранить этот файл с именем «*.gdbinit*» в каталоге проекта.

Создадим отладочную конфигурацию для проекта. В главном меню нужно выполнить пункт «*Run*» → «*Debug Configurations*». В появившемся диалоге «*Debug Configurations*» выделить, в списке слева, пункт «*C/C++ Remote Application*» и затем, в верхней части списка, нажимаем на иконку добавления конфигурации (изображение чистого листа с плюсом). Далее производим настройку в соответствии с *Рис*. 23.

На шаге 5 (*Рис*. 23) указывается абсолютный путь куда будет скопирован исполняемый файл подготовленный на *host компьютере*, а на шаге 6 подаётся дополнительная команда для установки расширенных прав доступа и атрибута исполняемого файла.

Далее на вкладке «*Arguments*» в поле «*Program Arguments:*» следует набрать «*-display :0.0*». Эта команда позволяет использовать в качестве дисплея для отображения графики *целевую платформу*. Затем настала очередь вкладки «*Debugger*». Интерес представляет область «*Debugger Options*». Настроенная вкладка «*Main*» приведена на *Рис*. 24, вкладка «*Shared Libraries*» на *Рис*. 25, а вкладка «*Gdbserver Settings*» на *Рис*. 26.

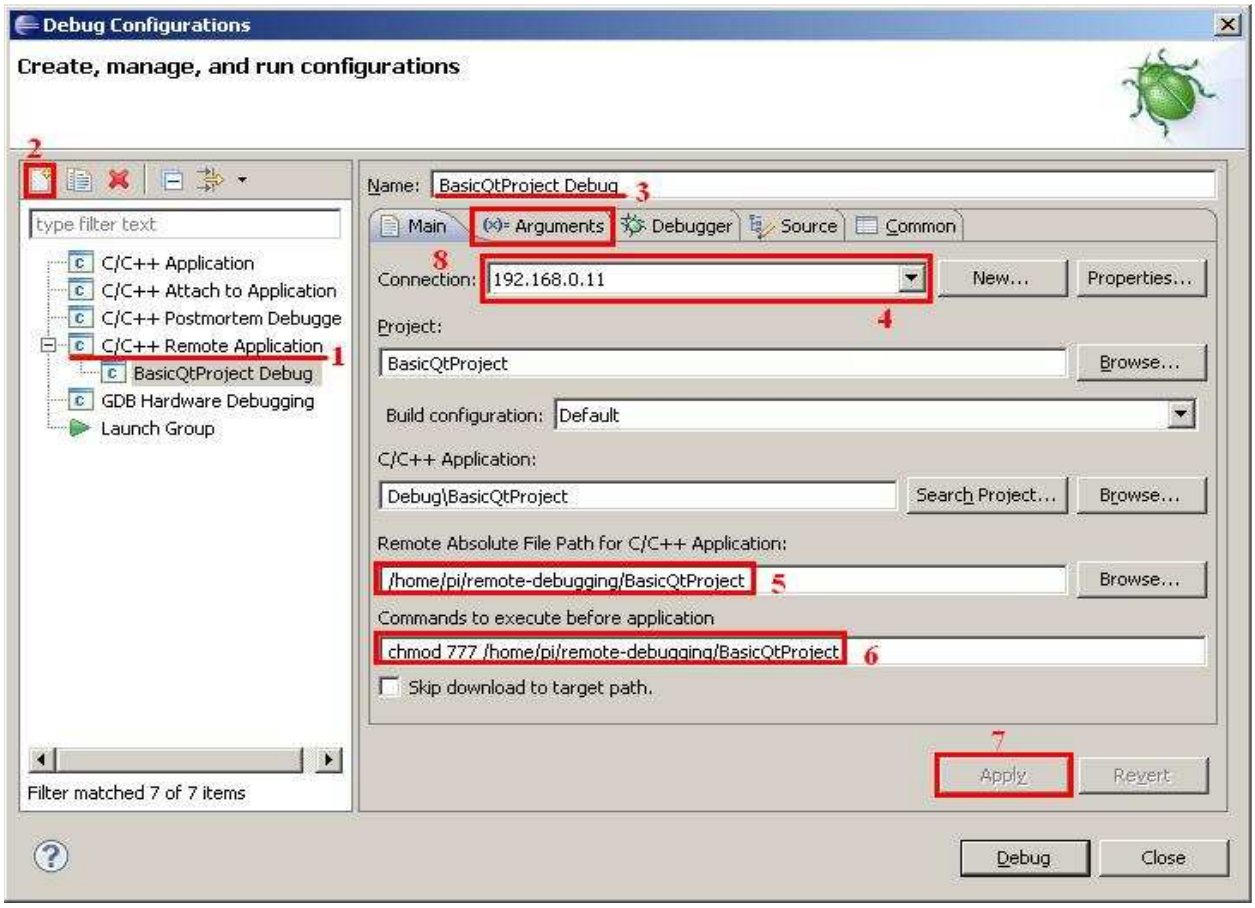

Рис. 23. Настройка отладочной конфигурации. Вкладка *Main*

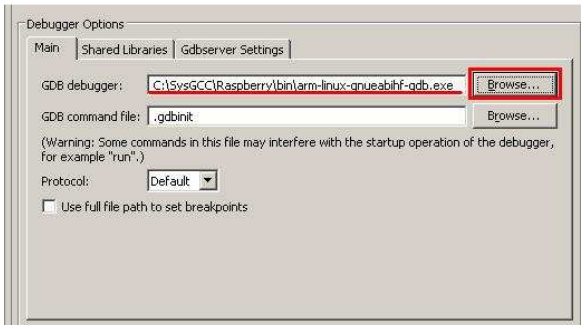

Рис. 24. Debugger Options. Вкладка *Main*

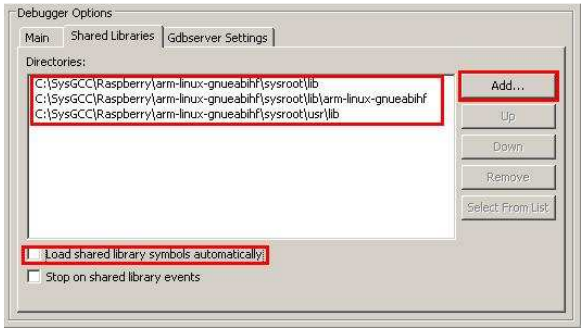

Рис. 25. *Debugger Options*. Вкладка *Shared Libraries*

В первом и третьем случаях указывается местоположение *GDB клиента* на *host компьютере* и местоположение *GDB сервера* на *целевой платформе*. Во втором указываются дополнительные библиотеки необходимы для нормальной работы отладки. Важно снять галочку «*Load shared library symbols automatically*». Эта галочка справедлива для старых версий *gdbserver*.

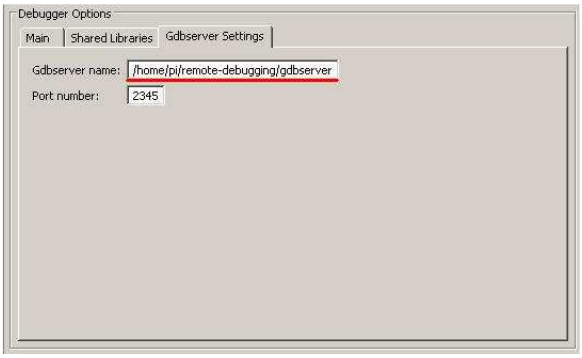

Рис. 26. *Debugger Options*. Вкладка *Gdbserver Settings*

На этом настройка отладочной конфигурации закончена. Нажатие на клавише «*Debug*» приведет к запуску компиляции и установки соединения с *целевой платформой* по *SSH*. Появится окно подключения, где необходимо ввести пароль для доступа к *Raspberry pi*. *Eclipse* предложит переключится на перспективу «*Debug*». После этого программа запустит отладку и остановится на входе в функцию *main()*. После запуска на выполнение, на рабочем столе *Raspberry pi*, высветится окно программы.

Повторно запустить отладку можно через пункт главного меню *Eclipse* «*Run*» → «*Debug*» или нажав клавишу «*F11*».

Для того, чтобы просто, дистанционно, запустить скомпилированное приложение на *целевой платформе* необходимо выполнить пункт главного меню «*Run*» → «*Run*» или нажать сочетание клавиш «*Ctrl + F11*».

Для уменьшения размера исполняемого файла можно компилировать программу в *Release* версии, для чего в строке аргументов точки вызова утилиты *qmake.exe* опустить строки «*CONFIG+=debug*» и «*CONFIG+=declarative_dedug*» (*Рис*. 16).

Созданный проект можно сохранить на диски и использовать его как шаблон для разработки собственных проектов в будущем.

Для удобной компоновки форм (будущих окон графического интерфейса) можно применить установленную на *host компьютере* программу *Qt Designer* из состава библиотеки *Qt*.

ВЫВОДЫ

1. Одноплатный компьютер *Raspberry pi* предоставляет разработчикам широки возможности, сравнимые со средним персональным компьютером, но при этом выполнен в очень экономичном решении.

2. Существуют способы построения сред разработки для *Raspberry pi*, при помощи которых становится возможным создание программ с графическим интерфейсом сравнимым с тем, что предлагают современные персональные компьютеры.

3. Рассмотренные в статье сборки сред предлагают разработчику широкие возможности по написанию и отладки собственного кода, как на самом одноплатном компьютер, так и дистанционно, на более мощном, современном персональном компьютере.

ЛИТЕРАТУРА

[1] Raspberry Pi. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/Raspberry_Pi (дата обращения 23.05.14).

[2] Raspberry Pi Foundation. URL: http://www.raspberrypi.org/ (дата обращения 23.05.14).

[3] Package on package. From Wikipedia, the free encyclopedia. URL: http://en.wikipedia.org/wiki/Package_on_package (дата обращения 23.05.14).

[4] MIPI Alliance. URL: http://mipi.org/specifications/display-interface (дата обращения 23.05.14).

[5] Official Rev 1.0 schematics PDF. URL: www.raspberrypi.org/wpcontent/uploads/2012/04/Raspberry-Pi-Schema tics-R1.0.pdf (дата обращения 23.05.14).

[6] Official Rev 2.0 schematics PDF. URL: www.raspberrypi.org/wp-content/uploads/2012/10/Raspberry-Pi-R2.0-Schematics-Issue2.2_027.pdf (дата обращения 23.05.14).

[7] RPi Hardware. URL: http://elinux.org/RPi Hardware (дата обращения 23.05.14).

[8] ARM1176JZF-S Technical Reference Manual URL: http://infocenter.arm.com/help/topic/com.arm.doc.ddi0301h/DDI0301H_arm1176jzfs_r0p7_trm.pdf (дата обращения 23.05.14).

[9] ARM (архитектура). Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/ARM_%28%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%29 (дата обращения 23.05.14).

[10] BCM2835 ARM Peripherals. URL: www.raspberrypi.org/wp-content/uploads/2012/02/BCM2835-ARM-Peripherals.pdf (дата обращения 23.05.14).

[11] VideoCore. From Wikipedia, the free encyclopedia. URL: www.raspberrypi.org/ wp-content/uploads/2012/02/BCM2835-ARM-Peripherals.pdf (дата обращения 23.05.14).

[12] Raspberry Pi Wiki. RPi Hub. URL:http://elinux.org/RPi_Hub (дата обращения 23.05.14).

[13] Raspberry Pi Foundation. Downloads. . URL:http://www.raspberrypi.org/downloads/ (дата обращения 23.05.14).

[14] Raspberry Pi Foundation. Installing Operating System Images. URL: http://www.raspberrypi.org/documentation/installation/installing-images/README.md (дата обращения 23.05.14).

[15] Raspberry Pi Wiki. Rpi SD cards. URL: http://elinux.org/RPi_SD_cards (дата обращения 23.05.14).

[16] SourceForge. Win32 Disk Imager. This tool is used for writing images to USB sticks or SD/CF cards URL: http://sourceforge.net/projects/win32_diskimager/ (дата обращения 23.05.14).

[17] X Window System. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/X_Window_System (дата обращения 23.05.14).

[18] GTK+. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/GTK%2B (дата обращения 23.05.14).

[19] Qt. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/Qt (дата обращения 23.05.14).

[20] wxWidgets. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/WxWidgets (дата обращения 23.05.14).

[21] Juce. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/Juce (дата обращения 23.05.14).

[22] FLTK. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/FLTK (дата обращения 23.05.14).

[23] Qt Project. Tutorial: apt-get install Qt4 on the Rasperry Pi. URL: http://qt-project.org/wiki/apt-get_Qt4_on_the_Raspberry_Pi (дата обращения 23.05.14).

[24] Cross Compiling and Cross Debugging C++ with Eclipse from Debian Squeeze x64 to Debian Squeeze ARM (Raspberry Pi). URL: http://linuxtortures.blogspot.it/2012/06/cross-compiling-and-cross-debugging-c.html (дата обращения 23.05.14).

[25] Cross-Compiling for Raspberry Pi. URL: http://www.kitware.com/blog/home/post/426 (дата обращения 23.05.14).

[26] Syntax Error. Developping for Qt 4.8 Embedded on Windows. URL: http://c2143.blogspot.ru/?view=classic (дата обращения 23.05.14).
[27] GuruCoding.com. Articles and tools for software development professionals. URL: http://www.gurucoding.com/en/pc_cross_compiler/index.php (дата обращения 23.05.14).
[28] QT Configurator for Windows cross-toolchains. URL: http://visualgdb.com/tools/QtCrossTool/ (дата обращения 23.05.14).
[29] Windows toolchain for Raspberry PI. URL: http://gnutoolchains.com/raspberry/ (дата обращения 23.05.14).
[30] Crosstool-NG.org. URL: http://crosstool-ng.org/ (дата обращения 23.05.14).
[31] Buildroot Making Embedded Linux Easy. URL: http://buildroot.uclibc.org/ (дата обращения 23.05.14).
[32] Remote cross debugging: Windows to Raspberry-Pi. URL: http://www.a2p.it/wordpress/tech-stuff/development/remote-debugging-ras pberrypi/ (дата обращения 23.05.14).
[33] Raspberry Pi Resources. Setting Up Windows Eclipse Programming of the Rpi. URL:http://www.raspberry-projects.com/pi/prog ramming-in-c/compilers-and-ides/eclipse/prog ramming-the-rpi-in-windows-using-eclipse (дата обращения 23.05.14).
[34] SSH. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/SSH (дата обращения 23.05.14).
[35] Qt Project. Qt Downloads. Index of archive qt. URL: http://download.qt-project.org/archive/qt/ (дата обращения 23.05.14).
[36] MinGW. Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/MinGW (дата обращения 23.05.14).
[37] SourceForge. MinGW - Minimalist GNU for Windows. A native Windows port of the GNU Compiler Collection (GCC). URL: http://sourceforge.net/projects/mingw/files/ (дата обращения 23.05.14).
[38] Хабрахабр. Eclipse + QT: установка и настройка. URL: http://habrahabr.ru/post/30636/ (дата обращения 23.05.14).
[39] Eclipse (среда разработки). Материал из Википедии — свободной энциклопедии. URL: http://ru.wikipedia.org/wiki/Eclipse_%28%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8%29 (дата обращения 23.05.14).

Алексей Викторович Ескин - ведущий инженер ООО «КБ Автоматика»,
E-mail: kba-elma@bk.ru

Вадим Аркадьевич Жмудь – заведующий кафедрой Автоматики НГТУ, профессор, доктор технических наук, автор более 200 научных статей, включая 10 патентов и 6 учебных пособий. Область научных интересов и компетенций – теория автоматического управления, электроника, лазерные системы, оптимизация, измерительная техника.
E-mail: oao_nips@bk.ru

Виталий Геннадьевич Трубин - зав. лаб. кафедры Автоматики НГТУ, директор ООО «КБ Автоматика». Автор 18 научных статей. Область интересов – разработка специализированной электроники.
E-mail: trubin@ngs.ru

Cheap Realization of Graphical Interface of User on the Base of Single-Plate Computer Raspberry Pi

A.V. ESKIN, V.A. ZHMUD, V.G. TRUBIN

Abstract: The paper discusses the questions involved with single-board computer *Raspberry Pi* as cheap platform for creating of devices with graphical interface and comparable with *PC*.

Key words – single-board computer *Raspberry Pi*, ARM, SoC, graphical interface, Linux, Window development environment Qt, cross compilation, development environment Eclipse, remote access, SSH.