

Dr. Bob Davidov

Компьютерные средства систем управления. Raspberry Pi.

Цель работы: Ознакомиться со средствами одноплатного компьютера Raspberry Pi для решения задач построения систем управления.

Задача работы: Построить несколько вариантов связи компьютера Raspberry Pi с внешними устройствами.

Приборы и принадлежности: Персональный компьютер с интерфейсом для записи SD карт, компьютер Raspberry Pi (версия B) с периферией и 8 ГБ микро SD картой.

ВВЕДЕНИЕ

Raspberry Pi - одноплатный компьютер, размером с кредитную карту, разработанный в Великобритании. В разработке самого компьютера и программного обеспечения к нему участвуют тысячи энтузиастов и разработчиков встраиваемых систем со всего мира. Одним из серьезных преимуществ компьютера Raspberry Pi является рекордно низкая цена - 35 долларов. За полтора года после запуска в производство продано более полутора миллионов компьютеров.

На базе компьютера строятся медиacentры, системы охраны и обработки данных. Для систем управления важным вопросом является обеспечение связи между вычислительными средствами и объектами. Внешние устройства к Raspberry Pi можно подключать через цифровой ввод/вывод и целый ряд последовательных интерфейсов порта GPIO. В этой работе рассматриваются средства и варианты подключения таких устройств ко второй ("B") версии компьютера.

ОБЩИЕ СВЕДЕНИЯ

Одноплатный компьютер Raspberry Pi «B» имеет следующие характеристики

- Процессор 700 MHz **ARM1176JZF-S** core
- ОЗУ 512 Мб
- Видео:
- Видео выходы: композитный RCA (RGB analog) или HDMI (RGB digital, высокой четкости, 1080 p) для подключения телевизора или дисплея через HDMI-DVI-D адаптер (видео доступно только через один выход)
- Два USB 2.0 порта для подключения клавиатуры, мыши, или USB хаб (HUB)
- Сетевое подключение: 10/100 Ethernet RJ45
- Аудио выход (стерео): TRC 3.5 мм разъём; или через HDMI адаптер
- Аудио вход: отсутствует, возможен через USB микрофон или звуковую карту
- Приемник SD карты (4 .. 32 ГБ с Linux операционной средой)

- **Разъем GPIO** (Рис. 2) конфигурируется для работы в режиме цифрового ввода – вывода, аппаратного ШИМ (один вывод) или для реализации следующих интерфейсов:
 - Serial Peripheral Interface Bus (SPI),
 - I²C,
 - I²S, (на нераспаянных контактах)
 - Universal Asynchronous Receiver / Transmitter (UART)

Примечание: Высокий уровень вывода порта GPIO: 3.3 В. **Подключение устройств с уровнем 5 В недопустимо!**

Рядом с основным разъёмом GPIO на плате разведены (но не распаяны) ещё четыре пары GPIO выводов, дополнительно дающие I²C и I²S интерфейсы.

Частота аппаратного ШИМ зависит от скважности: максимальная частота 300 кГц наблюдается при скважности 512/1024.

Максимальная частота устойчивого переключения цифрового вывода GPIO: ~27 кГц под управлением программы на Python (интерпретируемый язык программирования) и 5 МГц под управлением программы на C (язык компилируемого типа).

Контакты разъема GPIO подключены напрямую к процессору ARM и весьма чувствительны к статическому электричеству. **Используйте заземление, перед тем, как касаться выводов GPIO или оборудования, подключенного к GPIO.**

В результате кратковременного замыкания двух контактов разъема GPIO между собой или замыкания контакта питания на землю могут сработать защитные предохранители которые выключат Raspberry Pi. Предохранителям требуется время (до нескольких часов) на восстановление.

- CSI разъем для подключения видеокамеры (5 MPixel: 2592 x 1944)
- Последовательный интерфейс JTAG для тестирования и отладки [5]
- Часы отсчетов реального времени: отсутствуют (время считывается из Интернет)
- Сигнальные светодиоды платы: три для Ethernet, один (красный) показывает наличие питания +5В и ещё один (зеленый) сигнализирует о работе SD-карты.

Питание (через разъем микро-USB тип B или GPIO разъем) 5В / 700 .. 1200 мА. При токе потребления больше 1А необходимо использовать соответствующий адаптер USB 5В / 220В. Линия питания USB кабеля должна иметь низкое сопротивление (< 0.5 Ом), поскольку при падении питающего напряжения на компьютере ниже 4 В отдельные устройства (например, USB мышь) перестают работать.

- Операционная система на SD карте одна из более чем 30 LINUX подобных сред: Debian Squeeze, Raspbian, Pidora, RISC OS, и др.
- Габаритные размеры 85 x 56 x 17 мм
- Вес 40 г
- Стоимость компьютера US \$35

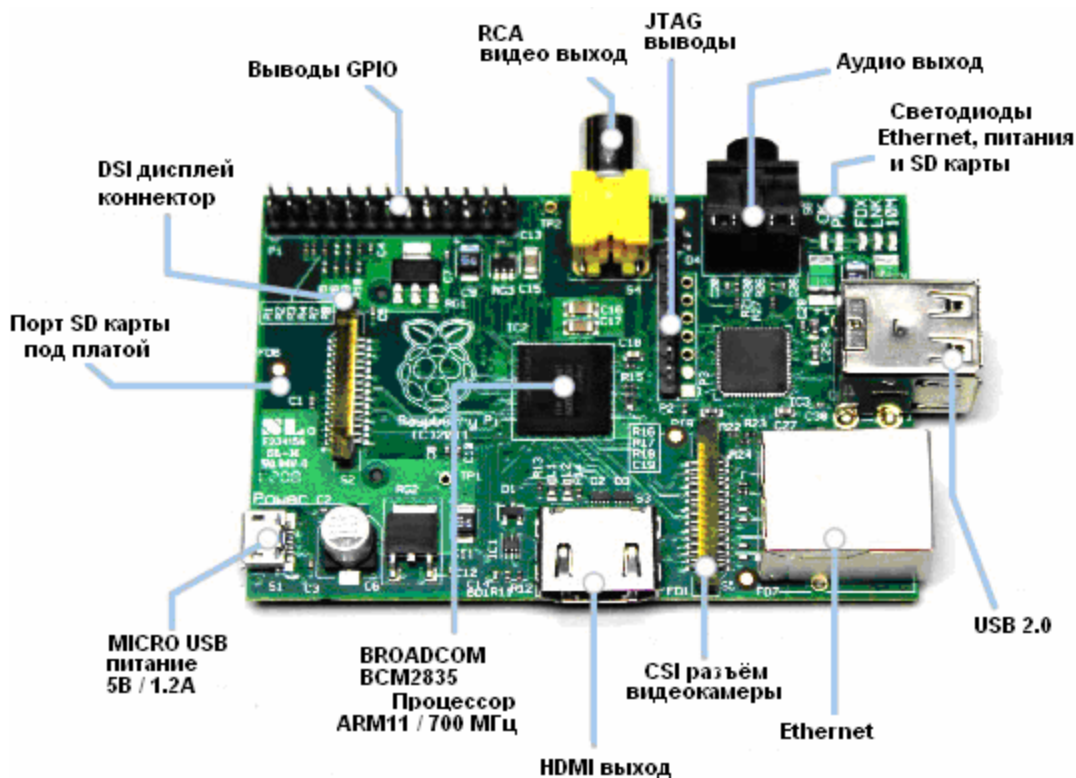


Рис. 1. Одноплатный компьютер Raspberry Pi.

Разъем GPIO

| | | | | | |
|------------|---------|----|----|---------|----------------|
| | 3.3 В | 1 | 2 | 5 В | |
| I2C (SDA) | GPIO 2 | 3 | 4 | 5 В | |
| I2C (SCL) | GPIO 3 | 5 | 6 | Ground | |
| | GPIO 4 | 7 | 8 | GPIO 14 | UART (TXD) |
| | Ground | 9 | 10 | GPIO 15 | UART (RXD) |
| | GPIO 17 | 11 | 12 | GPIO 18 | PWM аппаратный |
| | GPIO 27 | 13 | 14 | Ground | |
| | GPIO 22 | 15 | 16 | GPIO 23 | |
| | 3.3 В | 17 | 18 | GPIO 24 | |
| SPI (MOSI) | GPIO 10 | 19 | 20 | Ground | |
| SPI (MISO) | GPIO 09 | 21 | 22 | GPIO 25 | |
| SPI (SCLK) | GPIO 11 | 23 | 24 | GPIO 08 | SPI (CE0) |
| | Ground | 25 | 26 | GPIO 07 | SPI (CE1) |

Рис. 2. Назначение выводов разъёма GPIO [6] для подключения внешних устройств (ревизия 2). Все сигнальные выводы GPIO можно сконфигурировать для обычного цифрового ввода/вывода. Отдельные выводы могут быть переконфигурированы для организации последовательных интерфейсов. Суммарное питание 3.3 В (через выводы 1 и 17) не должно превышать 50 мА. UART интерфейс программируется с использованием выводов 8 (TXD) и 10

(RXD), I²C интерфейс использует выводы 3 (SDA) и 5 (SCL), SPI интерфейс использует выводы 24 (CE0_N), 26 (CE1_N), 19(MOSI), 21(MISO) и 23 (SCLK) Вывод 18 можно сконфигурировать на формирование сигнала ШИМ (PWM) аппаратными средствами. Через UART интерфейс, например, можно подключить, например, дисплей 3.2 inch TFT-LCD Touch Intelligent Display Module 240 x RGB x 320, µLCD-32PT(SGC).

Примечание. Уровни напряжения GPIO 0 / 3,3 В.

Входной сигнал на любом из цифровых выводов GPIO может служить источником внешнего прерывания.



Рис. 3. Примеры плат [4] расширения для Raspberry Pi подключаемые через разъем GPIO. Слева – АЦП (ADC) с переключаемой разрядностью и частотой 3.75 преобразований в секунду для 17 разрядного режима, 15 пр./сек для 15 разрядов, 60 для 13 и 240 преобразований в секунду для 11 разрядной конфигурацией. Справа - часы реального времени с дополнительным I²C интерфейсом. Продолжение контактов GPIO на платах расширения позволяет подключать несколько плат к компьютеру одновременно.

Краткая характеристика операционных сред для Raspberry Pi

Существует более 30 дистрибутивов операционных систем Linux для компьютера Raspberry Pi, среди них [2]:

- **Raspbian**

Raspberry Pi + Debian = Raspbian. Проект по созданию порта Debian Wheezy (7.x) armhf с поддержкой математического сопроцессора для Raspberry Pi. Цель проекта в том, чтобы предоставить пользователям Raspberry Pi доступ к более, чем 10000 бинарных пакетов Debian оптимизированных для наилучшей совместимости с Raspberry Pi. В настоящее время, усилия направлены на то, чтобы сделать Raspbian самым простым, самым стабильным и наиболее оптимизированным дистрибутивом Linux для Raspberry Pi. Дополнительная информация находится на [1].

- **RISC OS**, это быстрая и легкая операционная система, разработанная в Кембридже компанией Acorn. Выпущенная впервые в 1987-м году, она уходит корнями непосредственно к команде разработчиков микропроцессора ARM.
- **OpenELEC** создана для поддержки XBMC (медиацентр с открытым исходным кодом).

- **Arch.** Arch Linux для Raspberry Pi поставляется без графического интерфейса, который, тем не менее, легко установить самостоятельно. Дистрибутив Arch не предназначен для начинающих.
- **RaspBMC** - это минимальный дистрибутив Linux, основанный на Debian, для проигрывания медиа файлов в структуре недорого домашнего медицентра.









Создание загрузочной SD карты

Для работы компьютера требуется операционная среда, **img** файл которой можно загрузить, например, с сайта [2]. В компьютер Raspberry Pi операционная среда загружается с SD карты. Сделать загрузочной SD карту в Windows 7/8/Vista/X/2000/NT помогает программа Win32DiskImager.exe [7]

Создание загрузочной SD карты, например, для пакета операционных сред **NOOBS** (включает ОС: Raspbian, Pidora, RISC OS, RaspBMC, Arch и OpenELEC) [2] или отдельной ОС, например, Raspbian “wheezy” выполняется в следующей последовательности.


1. При необходимости отформатируйте SD карту, например, программой SDFormater или утилитой операционной среды Windows: устройство > ПКМ > Format.
2. Загрузите пакет NOOBS или Raspbian с сайта [2]
3. Скопируйте **img** файл операционной среды из **zip** файла в рабочий каталог.

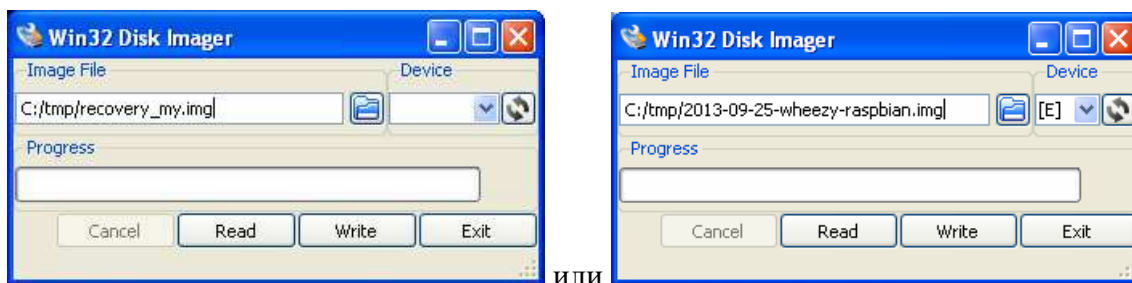
4.  [..] <DIR>
 2013-09-25-wheezy-raspbian **img** 2,962,227,200 Загрузите и распакуйте Win32DiskImager


| Name | Ext | Size |
|---|-----|------------|
|  [..] | | <DIR> |
|  README | txt | 1,434 |
|  Win32DiskImager | exe | 94,720 |
|  LGPL-2 | 1 | 26,434 |
|  GPL-2 | | 17,987 |
|  QtCore4 | dll | 2,741,248 |
|  QtGui4 | dll | 11,448,320 |
|  mingwm10 | dll | 15,964 |

5. Запустите **Win32DiskImager.exe** на правах администратора: ПКМ > run as >



6. Нажав на значок  выберите рабочий каталог с img файлом. Для NOOBS в поле File Name введите свое имя img файла, например, recovery_my.img. Для Raspbian введите оригинальное имя скопированное из zip файла, например, 2013-09-25-wheezy-raspbian.img. Убедитесь, что устройство SD карты (например, USB card reader Device: E) выбрано правильно.



7. В случае Raspbian перейдите к следующему пункту. Для установки NOOBS пакета нажмите Read и дождитесь создания recovery_my.img файла в рабочем каталоге.
8. Не меняя имени файла в поле **Image File** окна **Win32 Disk Imager** нажмите клавишу Write и, затем, **OK**. Распаковка img файла и построение загрузочной SD карты занимает несколько минут (см. состояние прогресс индикатора).
9. Убедитесь, что Windows показывает SD карту как загрузочное устройство  **boot**. Для Raspbian установка ОС выполнена, SD карта содержит, например, следующие файлы.

| Name | Ext | Size |
|------------------|--------|-----------|
| issue | txt | 137 |
| LICENSE | oracle | 18,974 |
| cmdline | txt | 142 |
| config | txt | 1,180 |
| bootcode | bin | 17,816 |
| fixup | dat | 5,746 |
| fixup_cd | dat | 2,037 |
| fixup_x | dat | 8,779 |
| kernel | img | 2,824,280 |
| kernel_emergency | img | 9,614,632 |
| start | elf | 2,497,684 |
| start_cd | elf | 471,256 |
| start_x | elf | 3,476,100 |

10. Для полной установки NOOBS необходимо скопировать средствами Windows все файлы и папки из соответствующего zip файла на SD карту, которая в результате будет содержать, например,

| Name | Ext | Size |
|----------------------------|---------|------------|
| [.] | <DIR> | |
| [defaults] | <DIR> | |
| [os] | <DIR> | |
| bootcode | bin | 17,824 |
| BUILD-DATA | | 268 |
| INSTRUCTIONS-README | txt | 2,249 |
| recovery | cmdline | 71 |
| recovery | elf | 471,736 |
| recovery | img | 2,093,824 |
| recovery | rfs | 20,579,519 |
| RECOVERY_FILES_DO_NOT_EDIT | | 0 |
| riscos-boot | bin | 9,728 |

Настройка ОС

Команда консоли `sudo raspi-config` вызывает следующие установки ОС.

- Expand Filesystem - увеличение root размера ОС до полного размера SD карты
- Change User Password - смена пароля "raspberrry" пользователя "pi".
- Enable Boot to Desktop/Scratch – выбор загрузки в режиме консоли или визуальной оболочки.
- Internationalisation Options
 - I1 Change Locate - установка языка системы. Выбор русской кодировки UTF-8 (en_GB.UTF-8 UTF-8) осуществляется пробелом.
 - I2 Change Timezone – выбор часового пояса. В Raspberry Pi нет своих часов, поэтому время берется из Интернета.
 - I3 Change Keyboard Layout- выбор драйвера клавиатуры
- Enable Camera – подключение драйвера видеокамеры
- Add to Rastrack – подключение к <http://rastrack.co.uk>

- Overclock - разгон процессора Raspberry Pi. Следует учитывать, что при увеличении частоты процессора возрастает и потребление энергии.
- Advanced Options
 - A1 Overscan – удаление черной рамки на TV (не на дисплее)
 - A2 Hostname – установка сетевого имени для Raspberry Pi
 - A3 Memory Split - выделение памяти для графического процессора. При работе в режиме консоли достаточно 16 Мб, для видео следует выделить 64 или 128 Мб из ряда 16, 32, 64, 128 или 256.
 - A4 SSH - включение / выключение SSH сервера для удаленного управления.
 - A5 SPI – автоматическая загрузка SPI kernel (необходима, например, для PiFace)
 - A6 Update
- About raspi-config

Таблица 1. Список полезных команд ОС Linux для работы Raspberry Pi в режиме консоли.

| Команда консоли | Описание |
|-------------------|--|
| cal | Вывод календаря на текущий месяц |
| cd | Переход в папку, например, cd /home/pi |
| cp | Копирование файла, например, cp /tmp/file.txt /newfile.txt |
| Ctrl+C | Выход из открытой консольной программы |
| Ctrl+Ins | Копирование выделенного текста из консоли |
| date | Вывод времени и даты |
| df -Bm или df -BM | Вывод распределения пространства SD карты |
| dir | Вывод содержимого текущей папки |
| help | Вывод списка команд консоли |
| ls | Вывод содержимого текущей папки и наличия исполняемых файлов |
| mv | Переименование файла, например, mv old-file-name new-file-name |
| ping schem.net | Проверка подключения к Интернет |
| pwd | Вывод пути текущего каталога |
| rm | Удаление файла, например rm abc.txt |
| rmdir | Удаление пустой директории, например, rmdir mydirectory |
| Shift+Ins | Вставка текста в консоль |

| | |
|------------------------------|---|
| sudo | Ставится перед командой для ее выполнения с правами пользователя root |
| apt-get update | Обновляет список пакетов с репозитории |
| apt-get upgrade | Обновляет установленные пакеты (+544 кб) |
| apt-get install [имя пакета] | Устанавливает нужный пакет, например, sudo apt-get install mc |
| halt | Выключает компьютер |
| mkdir | Создание каталога |
| nano | Вызов текстового редактора |
| python | Запуск интерпретатора Python |
| raspi-config | Вызов настройки Raspberry Pi |
| reboot | Перезагрузка компьютера |
| startx | Загрузка визуальной оболочки |
| wget | Копирование файла в текущую директорию, например, wget http://mysite.com/myfile.deb |

Выход в Ethernet/Интернет

По умолчанию, операционная среда Raspbian [1] настроена на DHCP - сетевой протокол, позволяющий компьютерам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP. Если маршрутизатор сети работает в этом режиме, то Raspberry Pi автоматически получит свой IP адрес и будет работать в сети. Проверить подключение можно командой `ping cxem.net`

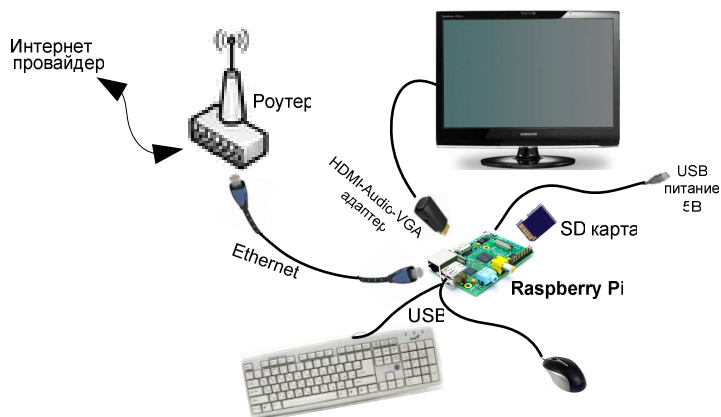


Рис. 4. Вариант подключения Raspberry Pi к Интернет через роутер

Для подключения Raspberry к Интернет через статический IP адрес необходимо отредактировать файл сетевых интерфейсов, и, при необходимости, конфигурационный файл и MAC адрес сетевого устройства.

Файл сетевых интерфейсов открывается через консоль командой

```
sudo nano /etc/network/interfaces.
```

В файле вместо строки

```
iface eth0 inet dhcp
```

следует указать используемые сетевые параметры, например,

```
iface eth0 inet static  
address 10.222.0.182  
netmask 255.255.252.0  
gateway 10.222.3.254
```

Конфигурационный файл имеет путь `/etc/resolv.conf`. Файл `resolv.conf` обычно содержит IP адреса серверов имён (DNS) которые переводят имена в адрес для любого узла доступного в сети.

В файле `resolv.conf` вместо строки

```
nameserver 8.8.8.8
```

следует написать адреса DNS сервера, например,

```
nameserver 83.243.64.2  
nameserver 83.243.65.2
```

Если провайдер обеспечивает доступ к Интернет только через одно сетевое устройство пользователя физический адрес (MAC адрес) которого известен провайдеру, то при подключения к сети провайдера напрямую нового устройства, например, Raspberry Pi с собственным Ethernet адаптером необходимо, чтобы провайдер изменил в своем списке старый адрес устройства на адрес нового устройства - адаптера Raspberry Pi. В этом случае первое (старое) устройство потеряет выход в Интернет, поскольку его адрес исключили из списка провайдера. Чтобы этого не происходило, пользователь должен самостоятельно одному из устройств присвоить физический адрес другого устройства, тогда любое из двух (или более) устройств, подключенное напрямую к сети провайдера, может выходить в Интернет без перенастроек со стороны провайдера.

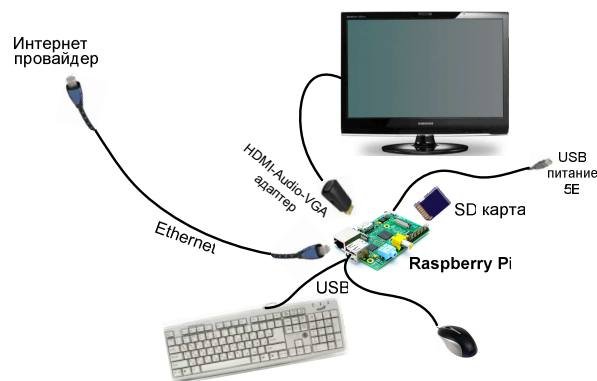


Рис. 5. Вариант подключения Raspberry Pi к сети провайдера напрямую (без роутера).

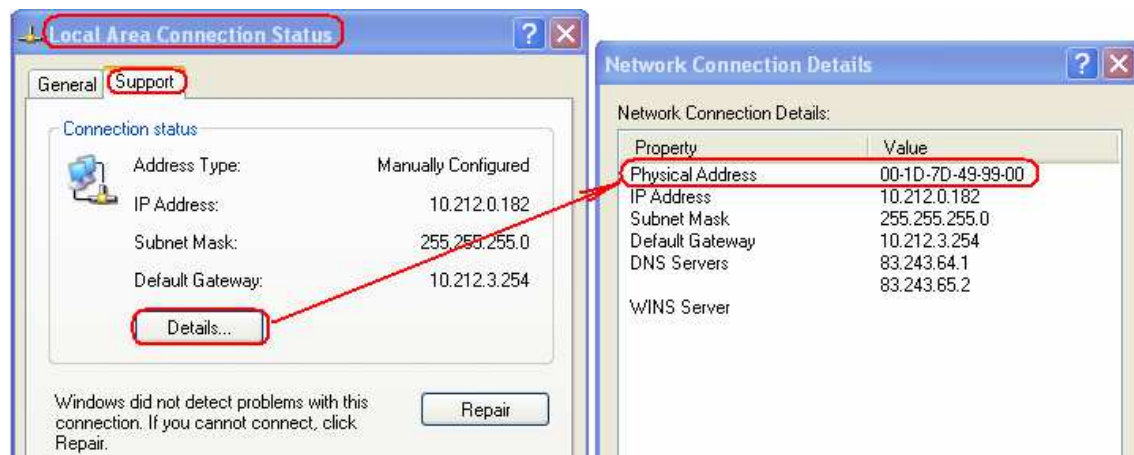
MAC адрес Ethernet адаптера Raspberry Pi находится в файле `/sys/class/net/eth0/address`. Его можно считать командой

`cat /sys/class/net/eth0/address` или командой `ifconfig`

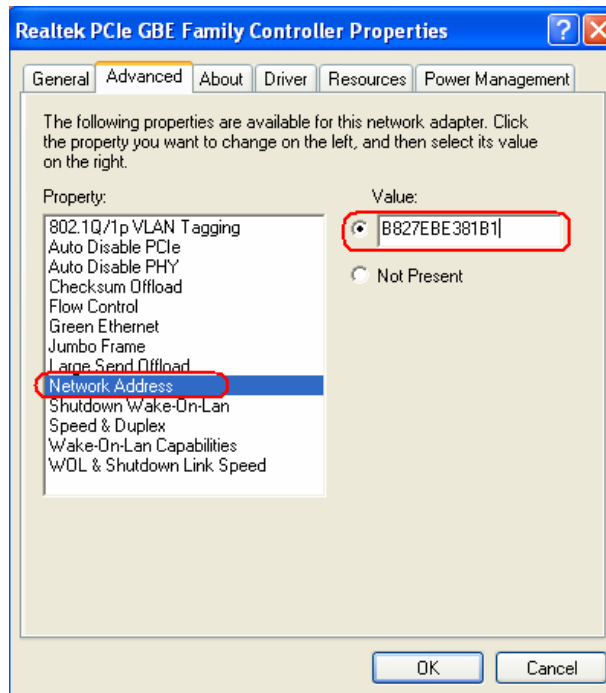
MAC адрес выглядит, например, так `b8:27:eb:e3:81:b1`. При необходимости MAC адрес можно изменить редактором `nano`.

Изменить MAC адрес Ethernet адаптера в Windows XP на адрес Raspberry Pi можно, например, в следующей последовательности.

1. Считайте и сохраните физический адрес (Physical Address) устройства: Start > My Network Places > Правая Клавша Мыши (ПКМ) > Properties > Network Connection > Local Area Connection > (ПКМ) > Status > вкладка Support > Details > Physical Address

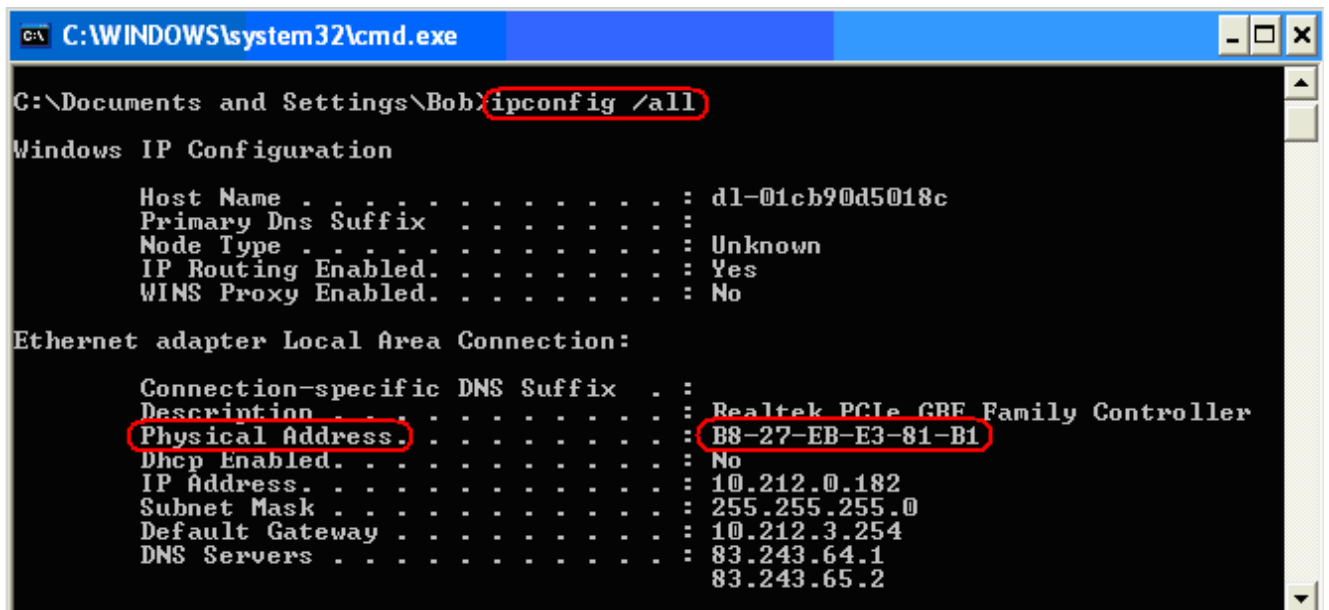


2. Измените физический адрес: Start > My Network Places > Правая Клавша Мыши (ПКМ) > Properties > Network Connection > Local Area Connection > (ПКМ) > Properties > General > Configure > Advanced > Properties > Network Address > Value > новый адрес (адрес другого адаптера)



3. Перегрузите систему.

4. Проверьте новый физический адрес сетевого адаптера: Start > Run > cmd > `ipconfig /all`



Новый адрес установлен. Если произошел сбой в установке адреса, повторите второй пункт.

В Raspberry Pi можно работать используя ресурсы другого компьютера. В этом случае минимальное подключение к Raspberry Pi составляет: питание через USB кабель, SD карту с ОС и сеть Ethernet (Рис. 6).



Рис. 6. Вариант работы с Raspberry Pi как с консолью сетевого компьютера.

В графической оболочке ОС Raspbian можно выйти в Интернет через браузер **Midory**. Оболочка из режима консоли загружается по команде **sudo startx**.

Формирование шины I²C на GPIO разъеме.

Периферийные устройства можно подключать к Raspberry Pi через последовательную шину данных I²C состоящую из двух двунаправленных линий SDA (для передачи данных) и SCL (частота для тактирования). Интерфейс I²C обеспечивает 10-битную адресацию, максимальную скорость выше 400 кбит/с (до 3,4 Мбит/с). Максимально допустимое количество микросхем, подсоединенных к одной шине, ограничивается максимальной емкостью шины в 400 пФ.

Подключение к I²C шине 8(16)-разрядных портов, например, PCF8574AN позволяет увеличить количество цифровых вводов/выводов Raspberry Pi..

В операционной системе Raspbian шина I²C изначально (по дефолту) отключена. Чтобы выходы 3 и 5 разъема GPIO пропускали сигналы SDA и SCL шины I²C (Рис. 2), необходимо выполнить следующее.

- Командой **sudo nano /etc/modprobe.d/raspi-blacklist.conf** откройте файл с записями

```
blacklist spi-bcm2708  
blacklist i2c-bcm2708
```

- Закомментируйте строку **blacklist i2c-bcm2708** установкой **#** в начале строки.

```
# blacklist spi-bcm2708.
```

Примечание. Таким же образом можно можно обеспечить доступ к интерфейсу SPI:

- Сохраните изменения (**Ctrl-x**) и перезагрузите систему командой консоли **sudo reboot**

Теперь, после каждого включения компьютера для активизации интерфейса I²C (0-го и/или 1-го) необходимо вводить через консоль следующие команды.

```
sudo modprobe i2c-dev  
sudo chmod o+rw /dev/i2c-0  
sudo chmod o+rw /dev/i2c-1
```

- Примечание. 1. Эти команды можно записать в файл `/etc/rc.local` для автоматического активирования I²C интерфейса при загрузке компьютера.
2. Проверить установку I²C можно командой `i2cdetect -y`.

В результате должна появиться таблица

```
  0 1 2 3 4 5 6 7 8 9 a b c d e f
00      - - - - -
10 - - - - -
20 - - - - -
...
60 - - - - -
70 - - - - -
```

Для установки модуля I²C шины: `'smbus'` библиотеки `python` через Интернет введите

```
sudo apt-get install python-smbus.
```

Скорость шины, например, 400000 Гц можно задать командами

```
cd dev
sudo modprobe -r i2c_bcm2708 && sudo modprobe i2c_bcm2708 baudrate=400000
```

которая удаляет старый драйвер шины и устанавливает новый, с заданной частотой.

Частоту драйвера можно вывести на экран командой `dmeshg`

Текстовый редактор Nano

Запуск текстового редактора Nano для создания и/или редактирования файла выполняется из командной строки.

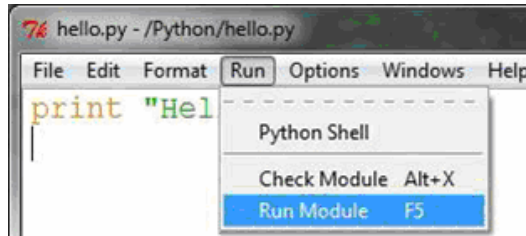
```
sudo nano filename.txt или nano filename.txt или sudo nano filename.py или sudo nano
```

Ранее созданный текстовый файл `filename.txt` (`filename.py`) можно загрузить сделав соответствующее указание в команде вызова редактора и непосредственно из редактора. Основные команды редактора отображаются в нижней части окна. Полный список команд можно раскрыть командой `^G` `<Ctrl G>`. Выход из редактора с сохранением или изменением имени файла выполняется командой `<Ctrl X>`.

Программирование порта GPIO на языке Python

Интерпретатор языка Python (версии 1 и 3) входит в пакет операционной среды Raspbian которая может быть установлена самостоятельно или в составе пакета NOOBS операционных сред Linux [2].

Взаимодействовать с интерпретатором Python можно через консоль: `sudo python` (Таблица 2, Таблица 3) или графическую оболочку `startx > ярлык IDLE` в которую программа загружается (или набирается через клавиатуру) в окно интерпретатора. Запуск программы выполняется через меню `> Run > Run Module F5`.



Результаты программы выводятся в окно Shell интерпретатора. Остановка программы выполняется командой `<Ctrl C>`.

Для работы с портом GPIO необходимо установить модуль RPi.GPIO:

```
$sudo apt-get update
$sudo apt-get install python-dev
$sudo apt-get install python-rpi.gpio
```

В Raspbian, как в дистрибутиве Linux, любой объект является файлом, что позволяет выводить и считывать сигналы с GPIO обычными командами оболочки bash прямо в терминале (см. команды в Таблица 4), например, вывод логической единицы выполняется командой записи "1" в файл, соответствующий нужному выводу.

Для выполнения команд терминала пакетом необходимо создать файл, например, build.sh и в первой строке сделать обращение к компилятору, например,

```
#!/bin/bash
```

Для запуска программы на языке Python (например, `my-script.py`) через консоль первой строчкой в программе надо прописать путь к интерпретатору, например, `#!/usr/bin/python` и Компиляция выполняется командой `chmod`:

```
$ chmod u+x my-script.py.py или
$ chmod 755 my-script.py
```

Запуск программы, например, `gpio-tst.py` выполняется командой

```
$ sudo python ./gpio-tst.py
```

Таблица 2. Команды Python для работы с GPIO в режиме цифрового ввода/вывода

| | |
|--|--|
| <code>sudo apt-get install python-rpi.gpio</code> (или <code>python3-rpi.gpio</code> для 3-й версии Питона) | Подключение библиотеки RPi.GPIO с репозиторияев (в новом дистрибутиве Raspbian [2] она уже установлена) |
| <code>sudo python</code> | Запуск Python для исполнения программы в консоли |
| <code>import RPi.GPIO as GPIO</code> | Импортирование библиотеки для работы с GPIO |
| <code>GPIO.setmode(GPIO.BOARD)</code> или <code>GPIO.setmode(GPIO.BCM)</code> | Установка способа нумерации выводов GPIO. BCM – для использования нумерации GPIO (см. Рис. 2); |

| | |
|--|--|
| | BOARD – для использования нумерации физических контактов (1 .. 26) разъема. |
| GPIO.setup(18, GPIO.OUT) GPIO.setup(16, GPIO.IN) | Конфигурация выводов на вывод / ввод |
| GPIO.setup(18, GPIO.OUT, initial=GPIO.LOW) | Конфигурация вывода на вывод с начальной установкой |
| GPIO.output(18, GPIO.HIGH) GPIO.output(18, True) GPIO.output(18, GPIO.LOW) GPIO.output(18, False) | Установка выходного напряжения. HIGH или True – логическая "1" (3.3 В) LOW или False – логический "0" (0 В). |
| GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_UP) | Настройка "подтяжка" вывода к питанию или к земле, где pull_up_down - PUD_OFF, PUD_UP или PUD_DOWN, PUD_OFF – по дефолту. |
| input_value = GPIO.input(16) | Считывание сигнала с GPIO 16 в переменную input_value. HIGH вход==1, LOW==0 |
| GPIO.add_event_detect(16, GPIO.RISING) | Разрешение обнаружения переключения входа: RISING (подъём), FALLING (спад) или BOTH (оба) |
| if GPIO.event_detected(16): print('Rising edge has occurred!') | Проверка, было ли переключение с момента последнего опроса |
| def my_event_callback_function(): print('Callback function called!') GPIO.add_event_callback(16, my_event_callback_function) | Включить в список вызов функции по переводу на входе вывода. Примечание. Для этого вызова необходимо сначала разрешить обнаружение функцией add_event_detect() |
| GPIO.remove_event_detect(16) | Снять разрешение обнаружения переключения входа. |
| def my_event_callback_function(): print('Callback function called!') GPIO.add_event_detect(16, GPIO.RISING, callback=my_event_callback_function) | Другой вариант вызова подпрограммы по обнаружению перепада входного сигнала (обработки прерывания). |

| | |
|-------------------------------------|---|
| GPIO.wait_for_edge(16, GPIO.RISING) | Ожидать нажатия кнопки. Не требует опроса. Используется минимальное время процессора. |
| GPIO.cleanup() | Завершение работы с GPIO. Выводы возвращаются в первоначальное состояние (высвобождаются для повторного использования). |

Таблица 3. Команды Python для работы GPIO в режиме программного Широтно-Импульсного Модулятора (ШИМ), PWM

| | |
|---|---|
| import RPi.GPIO as GPIO | Импортирование библиотеки для работы с GPIO |
| GPIO.setmode(GPIO.BOARD) или GPIO.setmode(GPIO.BCM) | Установка способа нумерации выводов GPIO. BCM – для использования нумерации GPIO (см. Рис. 2); BOARD – для использования нумерации контактов (1 .. 26) разъёма. |
| GPIO.setup(18, GPIO.OUT) | Конфигурация вывода на вывод |
| p = GPIO.PWM(channel, frequency) p = GPIO.PWM(18, 0.5) | Создать экземпляр ШИМ (PWM), где channel – номер вывода, freq – частота ШИМ в Гц . |
| p.start(dc) | Старт ШИМ, где dc = 0.0 .. 100.0, $dc/100$ – скважность цикла длительностью 1/freq секунд. |
| p.ChangeFrequency(freq) | Изменить частоту $freq$ в Гц |
| p.ChangeDutyCycle(dc) | Изменить скважность ($dc/100$), dc = 0.0 .. 100.0 |
| p.stop() | Остановить ШИМ |
| GPIO.cleanup() | Завершение работы с GPIO. |

Таблица 4. Ручное переключение выводов GPIO через файловый доступ в режиме терминала

В Unix GPIO выходы Raspberry Pi являются файлами. В консольном окне через файлы можно устанавливать уровни напряжения на выводах GPIO.

| | |
|-----------------------------------|--|
| \$ sudo -s | Запуск оболочки пользователя, равносильно команде $sudo \$SHELL$ |
| #echo 24 > /sys/class/gpio/export | Создание файлов доступа к GPIO выводам |

| | |
|--|--|
| #echo "24" > /sys/class/gpio/export | |
| #echo out > /sys/class/gpio/gpio24/direction | Конфигурация вывода на вывод (Out) |
| #echo 1 > /sys/class/gpio/gpio24/value #echo 0 > /sys/class/gpio/gpio24/value | Запись логической единицы / нуля на вывод GPIO |
| #echo in > /sys/class/gpio/gpio24/direction | Конфигурация вывода на ввод (In) |
| #cat /sys/class/gpio/gpio24/value | Чтение ввода GPIO |
| #echo 24 > /sys/class/gpio/unexport | Удаление созданного файла |
| exit | Выход из оболочки |

Аппаратный ШИМ (PWM)

Raspberry Pi имеет один GPIO вывод (номер 12 в нотации GPIO, и 18 - в нотации BCM) который может быть сконфигурирован на формирование сигнала ШИМ (PWM) встроенным аппаратным генератором доступ к которому программными средствами осуществляется через соответствующие библиотеки, например, [wiringpi](#) [8].

Примечание: Частота ШИМ зависит от скважности: максимальная частота 300 кГц наблюдается при скважности 512/1024, а 150 кГц – при 255/1024.

Программирование GPIO на C

Для программного управления на языке C портами GPIO необходима библиотека, например, [bcm2835-1.17](#) [9]. Библиотека устанавливается через окно терминала в следующей последовательности.

1. Скачивание библиотеки.

```
sudo wget http://www.open.com.au/mikem/bcm2835/bcm2835-1.17.tar.gz
```

В текущем каталоге появляется файл bcm2835-1.17.tar.gz

2. Распаковка архива.

```
sudo tar zxvf bcm2835-1.17.tar.gz
```

В текущем каталоге появляется каталог bcm2835-1.17

3. Переход в директорию библиотеки.

```
cd bcm2835-1.17
```

4. Компиляция и установки библиотеки:

```
./configure make
sudo make check
sudo make install
```

Устанавливать компилятор C на Raspberry Pi не требуется поскольку в Linux ОС есть компилятор `gcc` программ на C. Для компиляции исходного кода, например, `gpio-test.c` в исполняемый файл с тем же именем и с использованием библиотеки `bcm2835` необходимо ввести через окно консоли

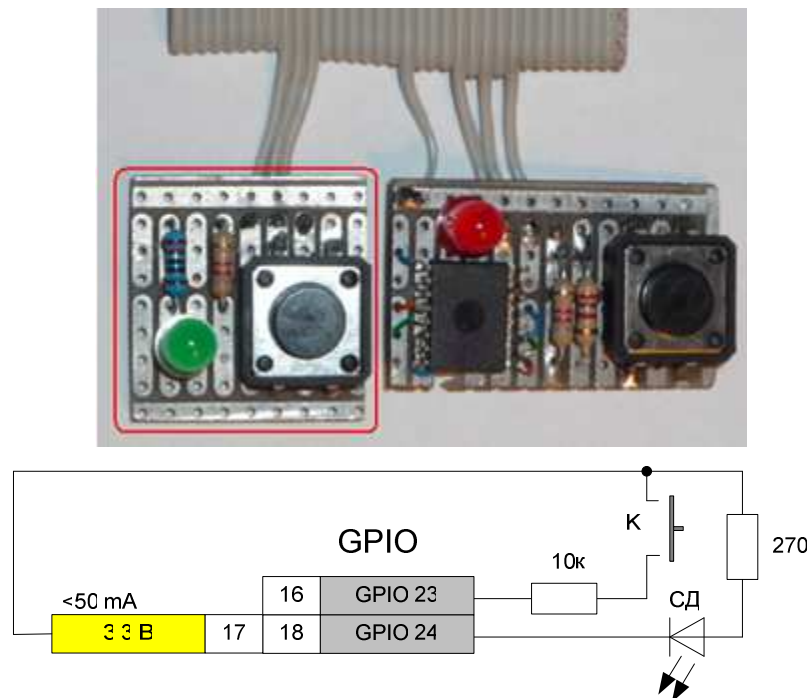
```
gcc -o gpio-test gpio-test.c -lrt -lbcm2835
```

Примечание: После успешной компиляции в каталоге появится новый исполняемый файл `gpio-test`, отмеченный зелёным цветом. Это можно увидеть введя команду консоли `ls`.

ПРИМЕРЫ ПОЛУЧЕНИЯ ПРОВЕРЕННЫХ РЕЗУЛЬТАТОВ И ВАРИАНТЫ ДЛЯ САМОКОНТРОЛЯ

Задание 1. Ручное переключение выводов GPIO в режиме терминала.

1. При выключенном компьютере подключите к выводам разъёма GPIO (Рис. 2) светодиод и механический замыкатель контактов (кнопка К) как показано на Рис. 7.



Примечание. Не забывайте, что у выводы GPIO не защищены от короткого замыкания и высокого ($> 3.3 \text{ V}$) напряжения.

Рис. 7. Электрическая схема подключения внешних устройств к разъему GPIO сконфигурированного на прием / передачу цифровых сигналов. Кнопка “К” специально подключена так, чтобы в отключенном состоянии на входе GPIO 23 был бы не определенный потенциал. Для обозначения этого состояния логическим нулем используется команда `GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)`. Другое подключение кнопки можно увидеть на Рис. 8. В этом примере периферия подключена к разъему GPIO через стандартный шлейф-кабель для FDD (34 жилы), показан фрагмент.

2. Загрузите ОС в режиме терминала. Введите логин и пароль.
`pi raspberry`
3. Перейдите в домашнюю директорию.
`cd имя_директории`
4. Создайте рабочий каталог, например,
`mkdir myprog`
5. Перейдите в рабочий каталог.
`cd myprog`
6. Загрузите интерпретатор Python
`$ sudo python`
7. Импортируйте библиотеку для работы с GPIO
`>>> import RPi.GPIO as GPIO`
8. Установите способ нумерации выводов GPIO (BCM или BOARD – номера физических контактов разъема)
`>>> GPIO.setmode(GPIO.BOARD)`
9. Сконфигурируйте выводы для работы в режиме передачи (OUT) и приёма (IN)
`>>> GPIO.setup(18, GPIO.OUT) или`
`>>> GPIO.setup(18, GPIO.OUT, initial=GPIO.LOW)`
`>>> GPIO.setup(16, GPIO.IN) или`
`>>> GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_UP) или`
`>>> GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)`
10. Запишите “1”, а затем “0” в выходной порт. Наблюдайте за состоянием светодиода.
`>>> GPIO.output(18, HIGH)`
`>>> GPIO.output(18, LOW)`
11. Считайте состояние кнопки
`>>> GPIO.input(16)`
12. Завершите работу возвращением всех выводов GPIO в исходное состояние
`>>> GPIO.cleanup()`
13. Выйдите из интерпретатора в консоль: `exit()` или `<Ctrl + D>`.

Задание 2. Ручная установка выводов GPIO для работы в режиме программного ШИМ под управлением интерпретатора Python.

1. Загрузите интерпретатор Python

```
$ sudo python
```

2. Импортируйте библиотеку для работы с GPIO

```
>>> import RPi.GPIO as GPIO
```

3. Установка способ нумерации выводов GPIO, например, BOARD – нумерация по физическим контактам разъёма GPIO.

```
>>>GPIO.setmode(GPIO.BOARD)
```

4. Сконфигурируйте вывод GPIO на вывод – формирование выходного напряжения на контакте

```
>>>GPIO.setup(18, GPIO.OUT)
```

5. Создайте экземпляр ШИМ (PWM) для работы с частотой 0,5 Гц

```
>>>p = GPIO.PWM(18, 0.5)
```

6. Запустите ШИМ со скважностью 50%. Наблюдайте ШИМ по миганию светодиода.

```
>>>p.start(50)
```

7. Измените частоту ШИМ freq

```
>>>p.ChangeFrequency(freq)
```

8. Измените скважность ШИМ: dc = 0.0 .. 100.0

```
>>>p.ChangeDutyCycle(dc)
```

9. Остановите ШИМ

```
>>>p.stop()
```

10. Завершите работу с GPIO.

```
>>>GPIO.cleanup()
```

Задание 3. Ручное переключение выводов GPIO через файловый доступ в режиме терминала.

1. В режиме терминала запустите оболочку SHELL

```
$ sudo -s
```

2. Создайте файл доступа к выводу GPIO (с BCM нумерацией).

```
echo 24 > /sys/class/gpio/export
```

3. Настройте вывод GPIO на формирование выходного напряжения.

```
echo out > /sys/class/gpio/gpio24/direction
```

4. Записывая логическую единицу / ноль на вывод GPIO наблюдайте за состоянием светодиода, подключенного к 24-му выводу.

```
echo 1 > /sys/class/gpio/gpio24/value  
echo 0 > /sys/class/gpio/gpio24/value
```

5. Удалите созданный файл

```
echo 24 > /sys/class/gpio/unexport
```

6. Завершите задание выходом из оболочки

```
exit
```

Задание 4. Пакетное выполнение команд оболочки bash.

1. С помощью редактора nano создайте файл, например, tsk_4.sh.

```
nano tsk_4.sh
```

2. Напишите программу включения и выключения светодиода подключенного к GPIO порту.

```
#!/bin/bash  
echo 24 > /sys/class/gpio/export  
echo out > /sys/class/gpio/gpio24/direction  
for i in {1..5}  
do  
    echo 0 > /sys/class/gpio/gpio24/value  
    echo "LED ON, loop $i"  
    sleep 1  
    echo 1 > /sys/class/gpio/gpio24/value  
    echo "LED OFF, loop $i"  
    sleep 1  
done  
echo 24 > /sys/class/gpio/unexport
```

3. Проверьте работу программы.

```
sudo bash tsk_4.sh
```

Задание 5. Запуск программы Python через консоль.

1. Для создания кода программы, например, tsk_5.py - бесконечного переключения светодиода откройте текстовый редактор nano.

```
nano tsk_5a.py
```

2. Напишите программу на языке Python которая управляет миганием светодиода СД подключенного к 18-му контакту разъёма GPIO (см. Рис. 7).

```
#!/usr/bin/ python
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setup(18, GPIO.OUT)
GPIO.output(18, True)
LEDState = 0
while 1:
    if LEDState == 0:
        GPIO.output(18, False)
        print ('1')
        time.sleep(1)
        LEDState = 1
    else:
        GPIO.output(18, True)
        print ("0")
        time.sleep(0.5)
        LEDState = 0
```

3. Запустите файл на исполнение. Проверьте работу программы по миганию светодиода.

```
sudo python tsk_5a.py
```

4. Остановите выполнение программы

```
<Ctrl C>.
```

5. Напишите и проверьте работу программы которая определяет состояние кнопки К (Рис. 7) и отображает его при помощи светодиода СД.

```
# Import the required module.
import RPi.GPIO as GPIO

# Set the mode of numbering the pins.
GPIO.setmode(GPIO.BOARD)

# GPIO pin 18 is the output.
GPIO.setup(18, GPIO.OUT)

# GPIO pin 16 is the input.
GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.DOWN)

# Initialise GPIO18 to high (true) so that the LED is off.
GPIO.output(18, True)

while 1:
    if GPIO.input(16):
```

```
GPIO.output( 18, False)
```

else:

```
# When the button switch is not pressed, turn off the LED.
```

```
GPIO.output( 18, True)
```

6. Проверьте работу программы по реакции светодиода на нажатие кнопки.
7. Остановите выполнение программы клавишами <Ctrl C>.

Задание 6. Работа в графической среде IDLE интерпретатора Python.

1. Перейдите из консоли в графическую оболочку

```
sudo startx
```

2. Загрузите интерпретатор Python дважды щелкнув по значку

```
IDLE 3
```

3. Напишите программу которая в бесконечном цикле выводит на экран 1 и 0.

```
import time
LEDState = 0
while 1
    if LEDState == 0:
        LEDState = 1
        print ('1')
        time.sleep (1)
    else:
        LEDState = 0
        print ('0')
        time.sleep (0.5)
```

4. Запустите программу нажатием клавиши F5.
5. Остановите программу клавишами <Ctrl C>.
6. Перейдите из GUI в консольный режим.

```
Logout
```


Задание 7. Управление выводами GPIO программой на языке C.

1. В консольном режиме напишите в текстовом редакторе **nano** C программу включения светодиода на 1 секунду. Имя программы, например, `tsk_7a`.

```
// tsk_7a.c
// Программа включает на 1 секунду светодиод,
// подключённый к порту P1_18, где 18 – физический номер вывода GPIO
// Компиляция выполняется командой gcc -o tsk_7a tsk_7a.c -lrt -lbcm2835
// Запуск программы: sudo tsk_7a

#include <bcm2835.h> // Заголовочный файл с описаниями функций выводов GPIO
#define PIN RPI_V2_GPIO_P1_18

int main()
{
    if (!bcm2835_init())           // Инициализация GPIO
        return 1;                 // Завершение программы, если инициализация не
    удавалась

    bcm2835_gpio_fsel(PIN, BCM2835_GPIO_FSEL_OUTP); // Установка направления
вывода
    int i;
    for (i=5; i!=0; i--)
    {
        bcm2835_gpio_write(PIN, LOW);           // Запись в порт логического нуля (~0В)
        bcm2835_delay(1000);                     // Задержка на 1 секунду
        bcm2835_gpio_write(PIN, HIGH);          // Запись в порт логической единицы (~3.3В)
        bcm2835_delay(1000);                     // Задержка
    }
    return 0;                                     // Выход из программы
}
```

2. Скомпилируйте программу.

```
gcc -o tsk_7a tsk_7a.c -lrt -lbcm2835,
где gcc- имя компилятора; bcm2835 - библиотека.
```

Примечание. Если компилятор не выдал никаких сообщений, то компиляция прошла успешно.

3. Проверьте появление исполняемого файла (отмеченного зеленым цветом) в текущем каталоге командой

```
ls
```

4. Запустите исполняемый файл. Проверьте работу программы по миганию светодиода.

```
sudo ./tsk_7a
```

5. Напишите и проверьте работу программы плавного изменения яркости светодиода

```

// tsk_7b.c
// program smoothly changes the brightness of the LED
// LED is connected to the GPIO port P1_18
// compilation: gcc -o tsk_7b tsk_7b.c -lrt -lbcm2835
#include <bcm2835.h>
#define PIN RPI_GPIO_P1_18
int main()
{
    if (!bcm2835_init())
        return 1;

    bcm2835_gpio_fsel(PIN, BCM2835_GPIO_FSEL_OUTP); // pin mode is output
    int i, t_on, t_off, period=2000; // duty cycle = t_on/period, PWM period in ms
    int a=5; // loop number
    while (a)
    {
        for (i=period; i!=0; i--)
        {
            t_on = i;
            t_off = period - i;
            bcm2835_gpio_write(PIN, LOW);
            delayMicroseconds(t_on);
            bcm2835_gpio_write(PIN, HIGH);
            delayMicroseconds(t_off);
        }
        a--;
    }
    return (!bcm2835_close ()); // Exit
}

```

Задание 8. Двусторонняя связь Raspberry Pi с внешней средой через I²C интерфейс организованный с использованием выводов интерфейса GPIO.

1. При выключенном компьютере подключите к выводам разъёма GPIO (Рис. 2) светодиод и механический замыкатель контактов (кнопка К) через I²C цифровой 8-разрядный порт **PCF8574A** как показано на Рис. 8.

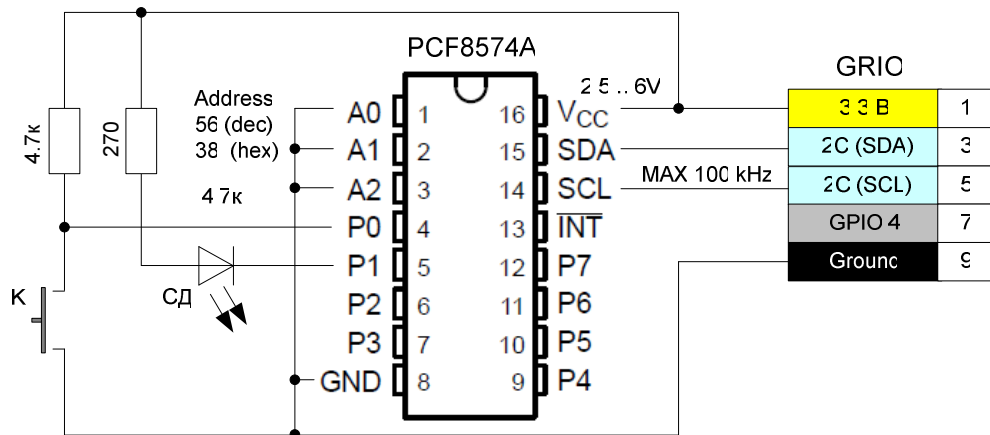
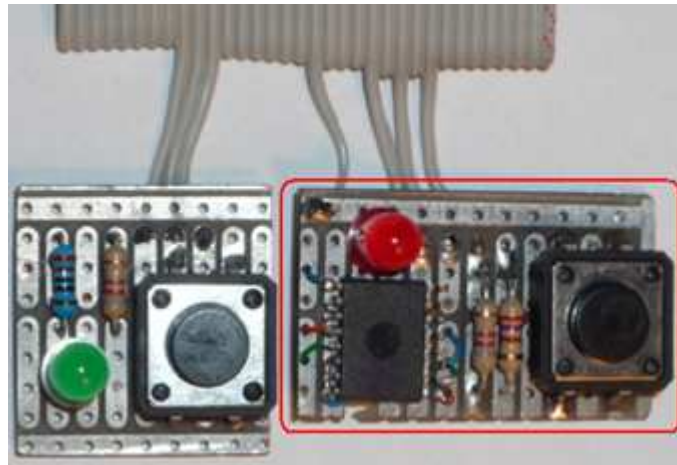


Рис. 8. Расширение цифровых входов/выходов через разъем GPIO сконфигурированного для работы в режиме последовательного I²C интерфейса.

2. Активируйте I²C интерфейс.

```
sudo modprobe i2c-dev
sudo chmod o+rw /dev/i2c-0
```

3. Проверьте частоту I²C канала

```
dmesg
```

4. Установите частоту 100 кГц (если она отличается),

```
sudo modprobe -r i2c_bcm2708 && sudo modprobe i2c_bcm2708 baudrate=100000
```

5. Напишите на языке Python программу, например, `tsk_8a.py` отображения состояния 8-ми входов I²C порта

```
import smbus
import time
```

```
bus = smbus.SMBus(1)
```

```
# перевести все выходы порта в состояние логической единицы: 3,3 В.
```

```
bus.write_byte(56, 255)
```

```
while 1:  
    a = bus.read_byte(56)  
    print (a)  
    time.sleep(0.5)
```

6. Проверьте программу по отображению состояния кнопки, подключенной к порту.

7. Остановите программу

<Ctrl C>

8. Напишите программу (tsk_8b.py) отображения светодиода состоянием кнопки.

```
import smbus  
# доступ к шине i2c  
bus = smbus.SMBus(1)  
# перевод порта в высокое состояние с выключением светодиода  
bus.write_byte(56, 255)  
while 1:  
    # если кнопка нажата, первый вход порта будет равен нулю, на входе порта будет  
    # 111111102 или 25410.  
    if bus.read_byte(56) in (254):  
        # запись 00000000 для включения светодиода.  
        bus.write_byte(56, 0)  
    else:  
        # запись 11111111 для выключения светодиода  
        bus.write_byte(56, 255)
```

9. Запустите программу

```
sudo python tsk_8b.py
```

10. Проверьте работу программы по включению светодиода при замыкании кнопки.

11. Выйдите из программы

<Ctrl C>

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Сравните компьютер Raspberry Pi с контроллером Arduino UNO.
2. Какие интерфейсы поддерживает Raspberry Pi?
3. Можно ли построить канал цифрового ввода/вывода и I²C интерфейс на одних и тех же выводах разъёма GPIO компьютера Raspberry Pi?

4. Как обеспечить подключение Raspberry Pi к Интернет?
5. Назовите варианты подключения Raspberry Pi к контроллеру Arduino.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. OS Raspbian <http://www.raspbian.org>
2. OS Raspbian downloads <http://www.raspberrypi.org/downloads>
3. RTC Pi Real time Clock Module <http://www.abelectronics.co.uk/products/3/Raspberry-Pi/15/RTC-Pi-Real-time-Clock-Module>
4. Connectors, ADC, and add-on boards and interfaces which are compatible with the Raspberry Pi Linux computer boards. <http://www.abelectronics.co.uk/products/3/Raspberry-Pi>
5. Последовательный интерфейс JTAG <http://habrahabr.ru/post/190012/>
6. General Purpose Input/Output (GPIO) http://elinux.org/Rpi_Low-level_peripherals#General_Purpose_Input.2FOutput_.28GPIO.29
7. Win32 Disk Imager <http://sourceforge.net/projects/win32diskimager/>
8. Библиотека WiringPi для аппаратного PWM и инструкция по её установке <https://github.com/WiringPi/WiringPi-Python>
9. C library for Broadcom BCM 2835 as used in Raspberry Pi <http://www.airspayce.com/mikem/bcm2835/index.html>
10. Raspberry Pi and Arduino Connected Using I2C. <http://blog.oscarliang.net/raspberry-pi-arduino-connected-i2c/>
11. Dr. Bob Davidov. Компьютерные технологии управления в технических системах <http://portalnp.ru/author/bobdavidov>