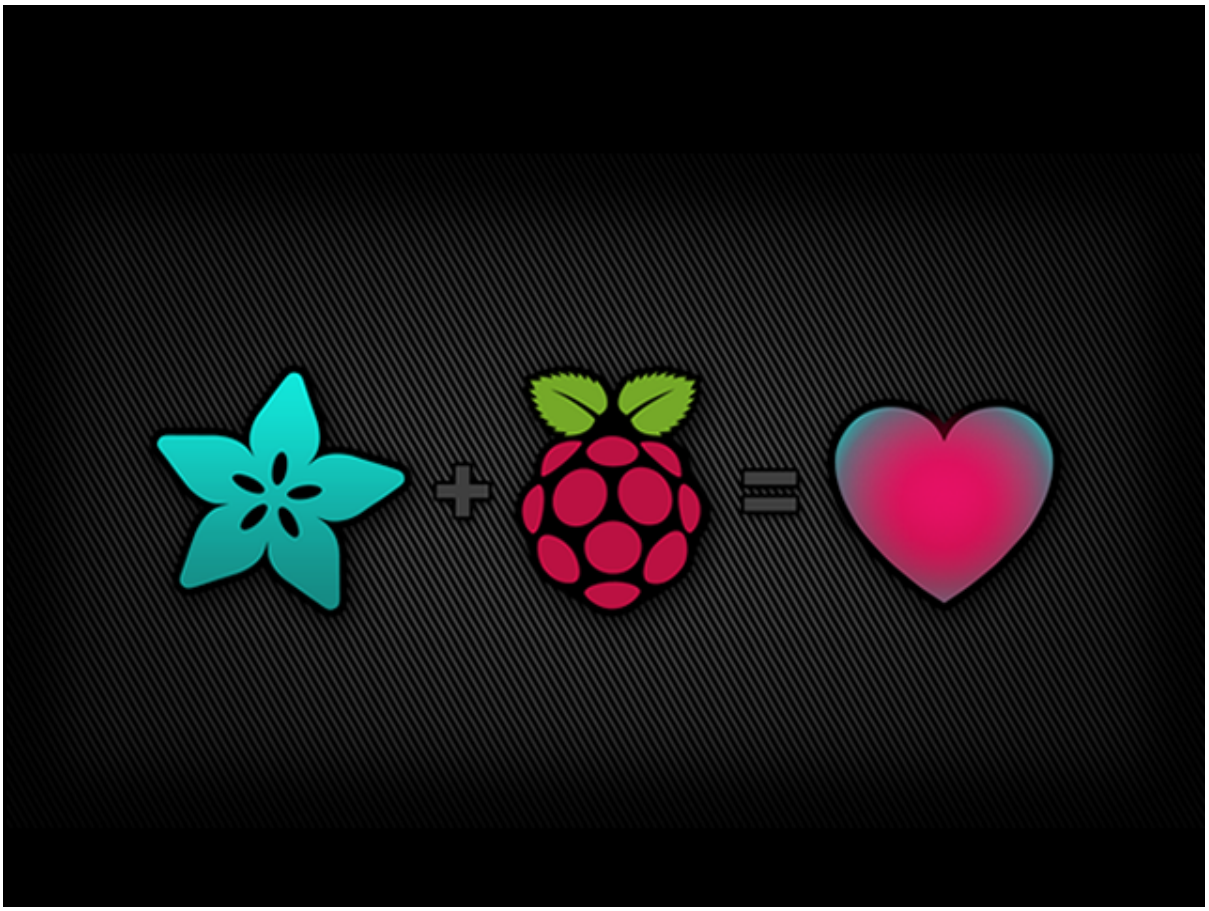
Adafruit Raspberry Pi Educational Linux Distro

Created by lady ada

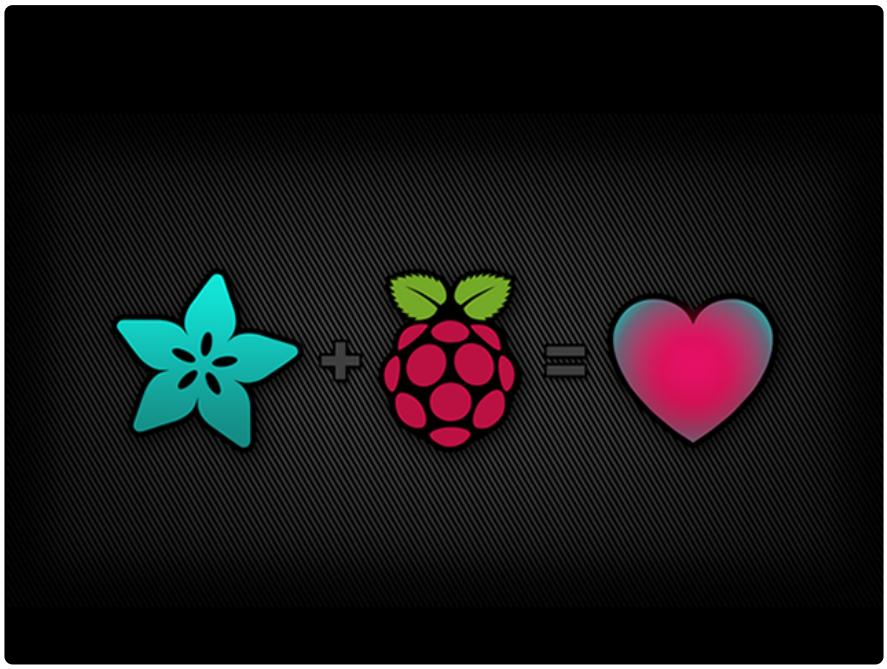


https://learn.adafruit.com/adafruit-raspberry-pi-educational-linux-distro

Last updated on 2023-11-01 03:48:46 PM EDT

Table of Contents

Overview



Its been a few years since we released Occidentalis. At the time, the standard Raspbian distribution was still very young and we found a lot of ways to add more hardware support.

However, nowadays the now named Raspberry Pi OS () is quite mature as software goes and has integrated a lot of the modifications!

So, instead of trying to keep up with continuous Raspbian releases, we created a Pi Bootstrapper, a program that you can run on your computer (or Pi) which will install a lot of our favorite tools and support for you! Check it out here ()

Occidentalis v0.3

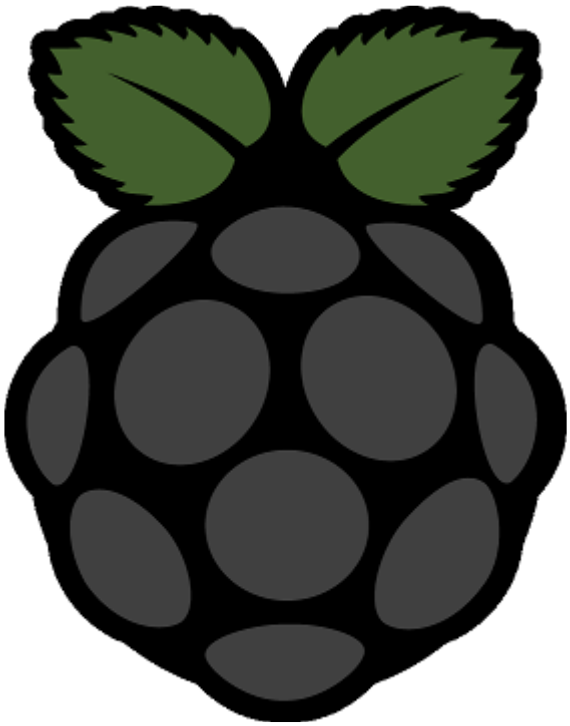
Its been a few years since we released Occidentalis. At the time, the standard Raspbian distribution was still very young and we found a lot of ways to add more hardware support.

However, nowadays Raspbian is quite mature as software goes and has integrated a lot of the modifications!

So, instead of trying to keep up with continuous Raspbian releases, we created a Pi Bootstrapper, a program that you can run on your computer (or Pi) which will install a lot of our favorite tools and support for you! Check it out here ()

Occidentalis v0.2

Occidentalis v0.2 is deprecated, please check out the v0.3 page for more details!



This is our second distro, Occidentalis v0.2. Rubus occidentalis is the black raspberry. It is derived from Raspbian Wheezy August 16 2012 ()

We have made a few key changes to make it more hardware-hacker friendly!

Version 0.2 updates (new!)

- Truncated image - only 2.6G now to fit on any 4G card
- raspi-config notice retained on boot
- Removed persistant wlan0 entry
- Password-change reminder on login
- Added RTC and lm-sensors kernel module
- Included kernel modules for: DS1307, AD525x I2C digipots, HMC6352, BMP085, ADS1015
- New! Adafruit's PWM/Servo kernel module for easy PWM/Servo control on GPIO#18

Version 0.1 updates (still included)

- Updated to Hexxeh firmware ()

- I2C () and hardware SPI () support
- I2C/SPI modules initialized on boot
- sshd on boot
- ssh keygen on first boot
- runs avahi daemon (Bonjour client) and is called raspberrypi.local
- Realtek RTL8188CUS wifi support (http://adafru.it/814)
- One wire support on GPIO #4 when loaded ()

Please keep in mind, we are not full time linux distro maintainers - we will try to fix any bugs we find but this distro is not for beginners or people who are new to linux!

This distribution is not compatible with Raspberry Pi's that have 'HYNIX' RAM Chips! We are working on a new distro that will be better than ever and also support HYNIX, no ETA at this time.

How to Install!

Click below to download the ZIP file:

- Adafruit Raspberry Pi Educational Distro - Occidentalis v0.2 () !900 Megs! (August 31, 2012)
  MD5 of the img itself (not the zip): 4256c0cdad82fa193c5e902143f1ca0e
  MD5 of the zip: 43456900352bb8bd8860902167195d83
- SHA1 of image: a609f588bca86694989ab7672badbce423aa89fd
- SHA1 of zip: 5f33ec07a183f336f973f82634f04108f690f5f3

and decompress it. Note that it is 2.6 GB large! You will need a 4GB card or larger. We suggest using our 4GB SD card which works great (http://adafru.it/102) After booting, run sudo raspi-config to auto-expand the file system to fit the card you've decided on

You will also need a SD or MicroSD card writer to burn the image on. We suggest using our speedy MicroSD card writer that works with any OS. (http://adafru.it/939)

Then follow the directions here (), except use the downloaded and uncompressed Occidentalis image instead of Wheezy

Features!

For details on the I2C, SPI, WiFi, Avahi, and 1-Wire modules please visit the v0.1 page (

)

New Features in v0.2!

New in v0.2, we have some great goodies!

Smaller Image

First up, we did not expand the filesystem beyond 2.6G, so the image itself is much smaller - only 2.6G instead of 4G. This will make writing the image faster, and it should also work better with a variety of 4G cards. There was no way to fit this in a 2G card, otherwise we would have done it.

Password and Configuration Reminders

Second, we retained the raspi-config notice on startup, just like the stock Wheezy distro. This will help people who wanted a reminder on how to set the timezone, disk size, password, keyboard, etc.

We also added a basic password reminder into ~/.profile - it will just check if the password hasn't been changed from the default. Change your password as soon as you boot, please!

Hardware RTC Support

The biggest news is we added a bunch of fun goodies to the kernel. We added RTC support so you can have an external RTC and use hwclock - we even have a tutorial about it here ().

Sensors Modules

We poked around the Kernel configuration file and added in module support for a few familiar sensors such as: AD525x Digipots, HMC6352 compass, BMP085 barometric/ temp sensor, ADS1015 I2C ADCs, etc.

Please Note: we didn't write or support these kernel modules, and we're not even sure if they all work, please experiment and read any kernel documentation about these modules as we do not have any tutorials or support for them at this time!

PWM and Servo Kernel Module

The most exciting addition is our custom-written kernel module specifically for handling the PWM/Servo capability of the Raspberry Pi's GPIO #18 pin. Unfortunately there is only one PWM pin available on the GPIO header and its shared with the Audio system. That means that you can't use PWM/Servo output and play audio through the 3.5mm jack at the same time. However, there might be a few situations where you just need a single servo or PWM and audio isn't a requirement.

The module was written by Sean Cross for Adafruit Industries, code is available at our github repository (see below)

This driver can be controlled through its sysfs entries. It will create the directory /sys/class/rpi-pwm/pwm0/ and populate it with the following files:

- active - Reports 1 if PWM is active. In delayed mode, write a 1 to this file to activate stored settings. Deactivate by writing a 0 to this file.
- delayed - If 0, any settings made will become active immediately. If 1, then settings are stored and won't take effect until a 1 is written into active.
- mode - The PWM mode. One of servo, pwm, or audio.
- servo - Moves the servo to this step. Range (0..servo_max) where 0 is a 0.5ms-long pulse and servo_max is a 2.5ms-long pulse.
- servo_max - The maximum number of servo steps, default of 32
- duty - Duty cycle percentage for PWM mode. Range (1..99) where 1 is the shortest positive pulse and 99 is the widest positive
- frequency - Desired frequency for PWM mode, write the value to this file
- real_frequency - The actual computed frequency, read the value from this file.
- mcf - A maximum common frequency (see Advanced below).

If you attempt to set a frequency or duty cycle that the Raspberry Pi does not support, you will get an error such as:
write error: Numerical result out of range
If this happens, the PWM will stop until you set values that are in range.

The mode file can be used to switch between pwm, servo, and audio mode:

- pwm - Drives a pulse with a frequency specified by the frequency file and a duty cycle of duty.
- servo - A special PWM mode that will drive a servo throughout its range of rotation, starting with 0.5ms wide pulse and ending with 2.5ms, some servos only respond to 1.0-2.0ms and some have a wider range, you will need to experiment to find the full range of your servo. Values are taken from the file servo, and range from 0 to servo_max (default 32 which is the max

resolution of 62.5us.) The PWM system does not seem to be able to handle a resolution better than 62.5us which is approximately 20 different servo positions or speeds. If you need better resolution, please check out our 16-channel servo driver tutorial which has 16 channels and 4us resolution ()

- audio - Echos the unfiltered contents of the right audio channel out the PWM port. Also enables delayed mode so that accidentally modifying PWM parameters won't cause the audio system to lock.

Using the PWM and playing audio at the same time is dicey at best. If you want to mirror audio out the PWM port, write audio into the mode file and leave it. When audio playback is done, you can switch back into pwm or servo mode. Then, either write 0 into the delayed file to get back into immediate mode, or set your parameters and write a 1 into the activate file.

Advanced settings

The default mcf is 16000 Hz. This is the frequency at which the PCM audio clock will run. The actual PWM output is derived based on this value, so it should be higher than the desired output frequency. For small duty cycles or for higher frequencies (e.g. above about 8 kHz), you may need to increase this value to get a more accurate real_ frequency. Due to rounding, it may not be possible to get your desired output rate. Compare the contents of the real_frequency file with that of the frequency file to determine accuracy.

Kernel Source

Want to compile your own modules? Or change the configuration of the kernel? Advanced users can find our kernel repo here ()

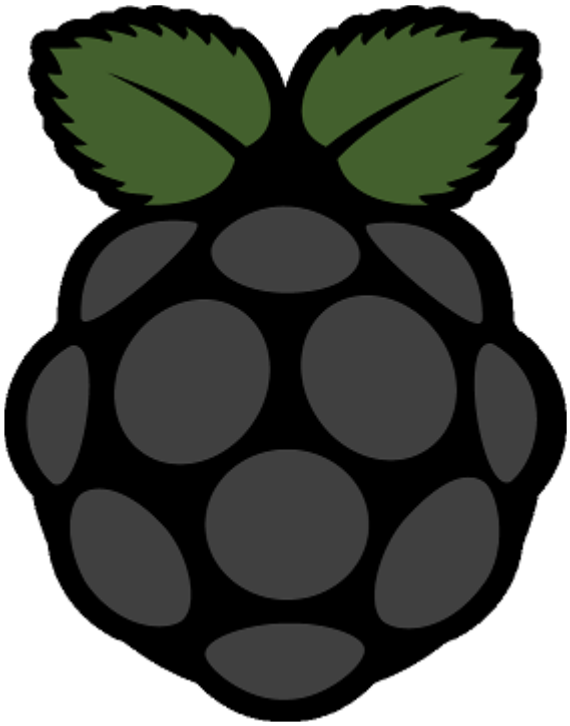
We also have a Kernel+Modules tgz file (), after you've copied this over to your pi, run the following commands

- tar -zxvf mykernel.tgz
- sudo cp tmp/kernel.img /boot/
- sudo cp -R tmp/modules/lib/* /lib/
- rm -rf tmp

We do not have any tutorials on how to download, compile or install the linux kernel.

Occidentalis v0.1

Occidentalis v0.1 is deprecated, please check out the v0.3 page for more details!



This is our first distro, Occidentalis v0.1. Rubus occidentalis is the black raspberry. It is derived from Raspbian Wheezy July 15

We have made a few key changes to make it more hardware-hacker friendly!

- Updated to Hexxeh firmware ()
- 4 Gig SD image (will not fit in 2 G cards!)
- I2C () and hardware SPI () support
- I2C/SPI modules initialized on boot
- sshd on boot
- ssh keygen on first boot
- runs avahi daemon (Bonjour client) and is called raspberrypi.local
- Realtek RTL8188CUS wifi support (http://adafru.it/814)
- One wire support on GPIO #4 when loaded ()

Please keep in mind, we are not full time linux distro maintainers - we will try to fix any bugs we find but this distro is not for beginners or people who are new to linux!

This distribution is not compatible with Raspberry Pi's that have 'HYNIX' RAM Chips! We are working on a new distro that will be better than ever and also support HYNIX, no ETA at this time.

How to Install!

Click below to download the ZIP file:

- Adafruit Raspberry Pi Educational Distro - Occidentalis v0.1 () !700 Megs! (August 2, 2012)
  MD5 of the img itself (not the zip): 34b5d3d511fcce0b82186816119d9881
  MD5 of the zip: cc3559cb6e7cb5f33b0e46118e16b748
  SHA1 of the img: e95dbb306bee8a9f77b486c729c7869923b7ee43
  SHA1 of the zip: 72eb71d316b8765d5594878b8662f2118dc4320a

and decompress it. Note that it is 4 GB large! You will need a 4GB card or larger. We suggest using our 4GB SD card which works great (http://adafru.it/102)

You will also need a SD or MicroSD card writer to burn the image on. We suggest using our speedy MicroSD card writer that works with any OS. (http://adafru.it/939)

Then follow the directions here (), except use the downloaded and uncompressed Occidentalis image instead of Wheezy

I2C Support

I2C support is on SDA and SCL pins. To test, connect any I2C device to power, ground, SDA and SCL. Then run i2cdetect -y 0 (as root) to detect which addresses are on the bus

For more ideas, check out this post () (by the most awesome cboot) and others on the Raspberry Pi forums

Our BMP085 (), MCP4725, () Servo Driver (), and 7-segment breakout () tutorials cover using I2C via Python on the Pi - so please check those out and read the code examples for I2C interfacing ideas!

SPI Support

SPI support is on the CLK/MOSI/MISO/CS0/CS1 pins. To test, connect your logic analyser/scope to the pins and run echo "testtext" > /dev/spidev0.0 to send some dummy data to the SPI port. You can simply read/write the /dev/spidev files to read/write from SPI

Our Light Painting () tutorial uses the hardware SPI system to write to digital LED strip, we also have a 'bitbanging' software SPI tutorial here () if you need such a thing

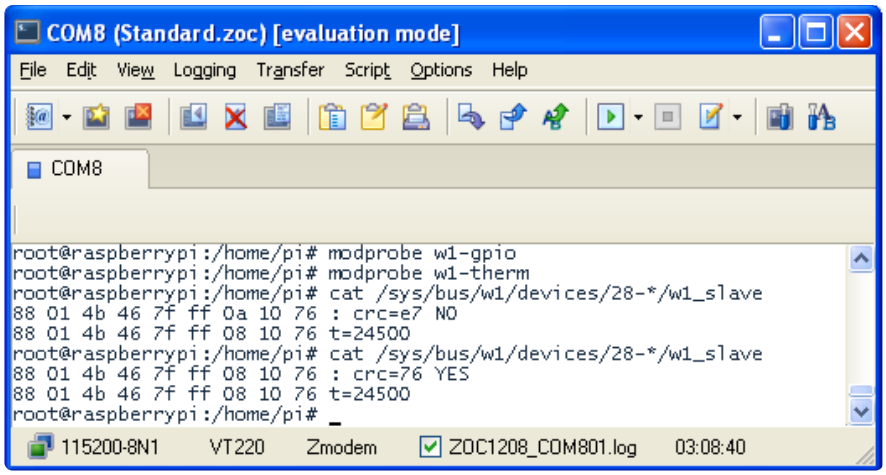
One Wire Support

One wire is most commonly used for DS18B20 temp sensors. The Pi does not have 'hardware' 1-wire support but it can bitbang it with some success. Connect a DS18B20 with VCC to 3V, ground to ground and Data to GPIO #4. Then connect a 4.7K resistor from Data to VCC.

Then run as root: modprobe w1-gpio and then modprobe w1-therm to attach the temperature submodule. Then you can run cat /sys/bus/w1/devices/28-*/w1_slave to read the temperature data from the bus

The first line has the CRC, if its "NO" then the data is corrupted. If you get a good CRC check, the second line has t=temperature in 1/1000 of a degree Centigrade. For example, below, the temperature is 24.5°C

Since 1-wire is bitbanged, its flakier than SPI or I2C. We have a short tutorial on using a DS18B20 sensor ()



WiFi support

We wanted to get our WiFI modules working (http://adafru.it/814), so we applied the RTL8192cu-based patches to the kernel. Please note that you almost certianly need a powered USB hub to run a wifi dongle.

Type ifconfig -a to verify that wlanN (wlan0, wlan1, etc) entry has been created.

You will have to edit /etc/network/interfaces with your SSID and password but after that, it should 'just work' - check iwconfig and iwscan if you're having problems

We have a tutorial on WiFi setup here ()

For Occidentalis v0.1 Users: Before plugging in the wifi adapter for the first time, please delete wlan0 from /etc/udev/rules.d/70-persistent-net.rules as we did not do this before taking the distro image. Otherwise, edit /etc/network/interfaces to refer to wlan1 instead of wlan0

Bonjour Support

Bonjour is what Apple uses to make it easier to find new devices on a LAN. Instead of having to look up the IP address, there's a local name. This distro uses raspberrypi.local by default. All Apple machines have Bonjour servers. If you have ever installed iTunes, it comes with it. Other Windows users can get it from here ()- its called the print server but its what you want

Test by trying to ping raspberrypi.local when the Pi is booted and connected to Ethernet (or WiFi once you have configured WiFi)

sshd on Boot

This image has sshd on boot - that means you can immediately ssh in using raspberry pi.local! The ssh keys are generated on boot but since the user/pass is simply pi/raspberry you should not put this on an accessible network until you've changed the password

Kernel Source

Want to compile your own modules? Or change the configuration of the kernel? Advanced users can find our kernel repo here ()

We do not have any tutorials on how to download, compile or install the linux kernel.