<u>Upon initial startup "raspi-config" will run:</u>

- * Manually → sudo raspi-config
- *info* Information about this tool
- *expand_rootfs* Expand root partition to fill SD-card
 Expand the *rootfs* to fill the balance of the SD-card
- overscan Change overscan
 - Select & configure this option if you are having display issues
- *configure_keyboard* Set keyboard layout
 - Select → "Desired Keyboard"
 - Select \rightarrow **Other**
 - Select → English (US)
 - Select → English (US)
 - Select → The default for the keyboard layout
 - Select → No compose key
 - For the Control+Alt+Backspace option: Select → No
- change_pass
 Change password for 'pi' user
 - \circ Always a good idea to perform this step
 - * Manually → passwd
- change_locale
- Set locale
- Deselect → en_GB.UTF-8 UTF-8 using the [Spacebar]
- Select → en_US.UTF-8 UTF-8 using the [Spacebar]
- Select \rightarrow **<Ok>** using the [Tab] \rightarrow [Enter]
- For Default locale for the system environment: \rightarrow en_US.UTF-8
- *change_timezone* Set timezone
 - Select \rightarrow US
 - Select \rightarrow Your current timezone {**Eastern** for my location}
- *memory_split* Change memory split
 - Return to this option later if necessary
- overclock
 Configure overclocking
 - \circ $\;$ Return to this option later if desired $\;$
- *ssh* Enable or disable ssh server
 - Select → <Enable>
- *boot_behaviour* Start desktop on boot?
 - Should we boot straight to desktop? \rightarrow **<No>** initially

- *update* Try to upgrade raspi-config
 o Select → [Enter] to update the "raspi-config" packages
- Select → <Finish > using [Tab] → [Enter]

At this point you are already logged in as *user* "pi" with the *password* you entered earlier. If you did not change the *password* the *default password* is "raspberry".

You will be presented with the "pi@raspberrypi" (user@host) prompt.

Next update the package manager by performing the following:

• sudo apt-get update

When the update is complete you will be returned to the "*username*@raspberrypi" (shell) prompt.

Finally, its time to start personalizing your Raspberry Pi. Start by changing the system's *hostname* by performing the following starting at the shell prompt.

• sudo nano /etc/hostname

Replace the "*raspberrypi*" with the name of your choice. Save your edits to */etc/hostname* and exit the *nano* editor.

- $\circ \quad \textit{WriteOut} \rightarrow \mathbf{^{0}}$
- Exit $\rightarrow \mathbf{X}$

This *hostname* change also needs to be reflected in the */etc/hosts* by performing the following starting at the shell prompt.

• sudo nano /etc/hosts

Replace the "127.0.1.1" value, "raspberrypi" at the bottom of the file with the name used in /etc/hostname above.

- $\circ \quad \textit{WriteOut} \rightarrow \mathbf{^{0}O}$
- Exit $\rightarrow \mathbf{X}$

At last it is time to perform the first **Reboot** of the system. Once again from the shell prompt enter:

• sudo reboot

Next setup the Raspberry Pi to work in a *headless* configuration. Since the ssh server has been previously enabled all that is required for *headless X-windows* support is to install and configure an *X-server*. In this case **"TightVNC Server"** will be used. From the shell prompt perform the following:

• sudo apt-get install tightvncserver

The first time you run the *tightvncserver* it will prompt you to set a password for each instance (the below example has two). This is the password that will be utilized when remotely connecting to the system.

• sudo /usr/bin/tightvncserver

Note: There is no need to create a "*view only"* password, unless there is a specific requirement for one.

**** Optional:** If you want the *TightVNC Server* to **automatically star**t when the system is rebooted perform the following from the shell prompt.

• sudo nano /etc/init.d/tightvncserver

Add the following to the /etc/init.d/tightvncserver file:

```
1. #! /bin/sh
 2. ### BEGIN INIT INFO
 3. # Provides:
                           tightvncserver
                           $local fs
 4. # Required-Start:
                           $local fs
 5. # Required-Stop:
                          2345
 6. # Default-Start:
                           016
 7. # Default-Stop:
 8. # Short-Description:
                           Start/stop tightvncserver
9. ### END INIT INFO
10.
11. # For more details see:
12. # http://www.penguintutor.com/linix/tightvnc
13.
14. ### Customize this entry
15. # Set the USER variable to the name of the user to
16. #
        start tightvncserver under
       This can be the default 'pi' or another user
17.#
18. export USER='pi'
19. #### End customization required
20.
21. eval cd ~$USER
22.
```

```
23. case "$1" in
24.
     start)
      echo""
25.
26.
       echo "Starting TightVNC Server :1 for $USER...."
27.
       su $USER -c '/usr/bin/tightvncserver :1 -geometry
1280x900 -depth 24'
28. # Multiple instances with different attributes can be started
29.
       echo "Starting TightVNC Server :2 for $USER...."
       su $USER -c '/usr/bin/tightvncserver :2 -geometry
30.
1024x600 -depth 24'
       echo""
31.
32.
       ;;
33. stop)
34.
       pkill Xtightvnc
       echo""
35.
       echo "TightVNC Server Stopped! "
36.
       echo""
37.
38.
       ;;
39.
     *)
       echo""
40.
       echo "Usage: /etc/init.d/tightvncserver {start|stop}"
41.
       echo""
42.
43.
       exit 1
44.
       ;;
45. esac
46. exit 0
```

Change the file owner to root (which is standard ownership for init files).

```
• sudo chown root:root /etc/init.d/tightvncserver
```

Change the file *modifier* (attributes) to allow execution.

```
• sudo chmod 755 /etc/init.d/tightvncserver
```

Add the script, "/etc/init.d/tightvncserver", to the default runlevels.

```
• sudo update-rc.d tightvncserver defaults
```

update-rc.d responds with \rightarrow update-rc.d: using dependency based boot sequencing

TightVNC Server will now automatically start after a reboot.

Reboot the system.

• sudo reboot

**** Optional:** If you have purchased the CODEC licenses from the Raspberry Pi store (<u>http://www.raspberrypi.com/</u>) To enable the CODECs on your device by add the following lines to the *config.txt* file in the FAT partition (*/boot*) of your SD card. Perform the following starting at the "*username*@raspberrypi" prompt.

• sudo nano /boot/config.txt

Add the following lines to the end of the config.txt file

- decode_MPG2=0×12345678
- decode_WVC1=0x12345678

To verify that the CODECs are now enabled, performing the following commands from the will report their respective status. Perform the following starting at the "*username*@raspberrypi" prompt.

- vcgencmd codec_enabled MPG2
- vcgencmd codec_enabled WVC1

**** Optional:** The Raspberry Pi CODEC status can be reported during system startup by creating the following script and including it in the initialization script sequence. Perform this from the shell prompt.

• sudo nano /etc/init.d/codec-status

Add the following to the */etc/init.d/codec-status* file:

```
1. #! /bin/sh
 2. ### BEGIN INIT INFO
 3. # Provides:
                                    Multimedia CODECs
                                 $local_fs
 4. # Required-Start:

    # Required-Stop:
    # Default-Start:
    # Default-Stop:

                                    $local fs
                                 2345
                                    016
 8. # Short-Description: Status of Multimedia CODECs
 9. ### END INIT INFO
10.
11. case "$1" in
12. start)
       echo""
13.

    echo "Raspberry Pi CODEC sta
15. echo ""
    # Report status of MPEG2 COI
17. vcgencmd codec_enabled MPG2
18. # Report status of VC-1 CODI

          echo "Raspberry Pi CODEC status: "
          # Report status of MPEG2 CODEC
          # Report status of VC-1 CODEC
```

19. vcgencmd codec_enabled WVC1
20. echo " "
21. ;;
22. esac
23. exit 0

Change the file owner to root (which is standard ownership for init files).

• sudo chown root:root /etc/init.d/codec-status

Change the file *modifier* (attributes) to allow execution.

• sudo chmod 755 /etc/init.d/codec-status

Add the script, "/etc/init.d/tightvncserver", to the default runlevels.

• sudo update-rc.d codec-status defaults update-rc.d responds with → update-rc.d: using dependency based boot sequencing

The Raspberry Pi CODEC status will now be automatically reported after a reboot.

****Optional:** Install "rsync" to aid in file copy/synchronization.

• sudo apt-get install rsync

****Optional:** Set a distinctive command line prompt. Edit the *~/.bashrc* file. Note that the "." file name preface indicates a 'hidden' file.

• nano ~/.bashrc → Edit the section marked in Red below.

```
if [ "$color_prompt" = yes ]; then
PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]
\u\[\033[01;33m\]@\[\033[01;32m\]\h\[\033[01;31m\] \[\
033[01;36m\]\w \[\033[01;33m\]!:>\[\033[00;37m\] `
else
PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ `
fi
```

To make the changes global perform the following:

• sudo cp /home/pi/.bashrc /root/.bashrc

See "[How-to] Set Terminal Command-Line Colors" for additional details.

****Optional:** Edit the ls directory color from Blue (very difficult to see) to Yellow by performing the following steps from the command prompt.

• dircolors -p > ~/.dircolorsrc

Edit the "~/.dircolorsrc" file. Change the "DIR" entry value marked in Red below.

DIR 01;33 # directory

• nano ~/.bashrc

```
eval "$(dircolors -b ~/.dircolorsrc)"
alias ls='ls -color=auto'
```

- nano ~/.dircolorsrc
- source ~/.bashrc

To make the changes global perform the following:

- sudo cp /home/pi/.dircolorsrc /root/.dircolorsrc
- sudo source /root/.bashrc

See "[How-to] Set Terminal Folder-File Colors" for additional details.

**** Optional:** To facilitate file backup or relocation install rsync.

• sudo apt-get install rsync

**** Optional:** For improved NTFS file system support, including read/write access install *ntfs-3g*.

• sudo apt-get install ntfs-3g

**** Optional:** Install *samba* to create and utilize Windows SMB shares.

- sudo apt-get install samba
- sudo apt-get install samba-common

• Note: Configuring samba and creating a new SMB network share detail can be found by 'Googling' "Samba Howto". If you just want to dive in and edit the /etc/samba/smb.conf perform the following at the "username@raspberrypi" prompt:

• sudo nano /etc/samba/smb.conf

This is covered in "[How-to] Mount Samba Shares in fstab using a Credential File".

**** Optional:** If desired you can create a *new user account* by performing the following starting at the "*username*@raspberrypi" prompt.

• sudo adduser *username*

• Follow the prompts

We will add the *new user* to the *sudoers* list by performing the following:

- sudo visudo
 - Under *#includedir /etc/sudoers.d* (which *visudo* implicitly edits)
 - Add an additional line after the "pi" entry
 - Username ALL=(ALL) NOPASSWD: ALL
 - $\circ \quad WriteOut \rightarrow ^{O}$
 - Exit \rightarrow **^**X

Now test the elevated privilege directive (*sudo*) by performing the following:

- Logout of the "pi" user account \rightarrow logout
- Login to the new *username* user account \rightarrow login

An easy way to test if your new *username* user account now has elevated privileges is to attempt to edit the *sudoers* file by performing the following:

• sudo visudo

If the above action was allowed the test is successful.

• Exit the editor by performing $\rightarrow X$

For the time being I would leave the "pi" account just in case...

** Optional: Install a Realtek RTL8188CU based Wireless LAN (WiFi) 802.11b/g/n network adapter.

- If the Raspberry Pi is powered and not in *halt* mode perform the following at the shell prompt.
- sudo shutdown -h now
- The system will shutdown and halt immediately.

Plug in the USB 2.0 WiFi adapter into the top USB 2.0 socket. I generally use the bottom USB 2.0 socket for my powered USB 2.0 hub.

Repower the Raspberry Pi. The USB 2.0 WiFi adapter should be automatically detected and configured using DHCP.

The easy way to do this is to **startx** and use the **WiFi Config** application link on the default desktop. The **wpa_gui** application window will be displayed with Adapter: \rightarrow wlan0 and Network: \rightarrow {blank}. The Current Status tab displays the Status: \rightarrow Inactive. Except for the Last Message: , all other fields are blank.

• Left-click the *Scan* button

The *Scan results* application window will additionally be displayed.

• Left-click the *Scan* button on this window also

The scan results will now be populated with WiFi networks your USB 2.0 adapter can 'see'.

• Double-click the line corresponding to the desired network SSID

Note: The WiFi network's SSID Broadcast, must be Enabled.

The top three fields should be automatically populated based upon the previously performed *scan*. Typically your WiFi network should be setup for *Authentication:* \rightarrow WPA2 Personal (PSK) and *Encryption:* \rightarrow CCMP.

- Enter the PSK: \rightarrow {your passphrase}
- Left-click the *Add* button to add the network to you list of networks
- Left-click the *Close* button to exit the *Scan results* window
- Left-click the *Connect* button to connect to the WiFi network

At this point the WiFi network you wish to connect to along with its encryption key have been defined.

Back at the *wpa_gui* window on the *Current Status* tab. Note that for *Adapter:* \rightarrow wlan0 and *Network:* \rightarrow your SSID all of the additional fields have been populated. Note that this tab supplies you DHCP assigned *IP address*. Now save the configuration.

Left-click <u>File</u> → Save Configuration Ctrl+S

You may now exit the *wpa_gui* window. *wpa_gui* will continue to run in background and be available in the tool tray.

←========= Base + Misc. + WiFi Configuration =======→

Note: This is a good place to create a SD-card image backup using *Win32 Disk Imager* program installed on you PC during the host PC setup. To perform this backup run *Win32DiskImager.exe* on your PC. Browse to the subdirectory where you whish to store your SD-card images, enter a new file name to use, and perform a *Read* operation to copy the contents of the SDcard to the image file you just created. Now you have a *Restore Point* to fall back to if necessary. If restoring an image to a SD-card perform a *Write* operation to copy the contents of an image file to the SD-card.

Note: A few words about my philosophy regarding networking. I prefer to utilize DHCP to assign all IP addresses with the exception of the *Router* (default Gateway) which I use the xxx.xxx.1 **statically assigned** IP address. Generally most home routers/wireless access points have the ability to perform as a *DHCP Server* for your network. I generally make the *scope* of the managed IP address pool from xxx.xxx.2 to xxx.xxx.254. If you want to assign a pseudo-static IP address to a host all that is necessary is to create a *DHCP reservation* for the host based upon the MAC address of host. In this way the same IP address will be assigned to that host each time a *DHCP address request* is made by the host. Additionally, for WiFi networks I like to also perform MAC address filtering of connected hosts. Even though this technique is not fool proof it provides an additional hurdle for the unwanted guest to overcome.

Note: To query the network interfaces on your Raspberry Pi perform the following from the shell prompt.

• ifconfig

Information about each of your network interfaces will be displayed.

eth0 Link encap:Ethernet HWaddr xx:xx:xx:xx:xx:xx inet addr:xxx.xxx.xxx Bcast:xxx.xxx.255 Mask:xxx.xxx.xxx.xxx UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:xx errors:0 dropped:0 overruns:0 frame:0
TX packets:xx errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:xxxx (x.x KiB) TX bytes:xxxxx (xx.x KiB)
Link encap:Local Loopback
inst addr:127 0.0 1 Marke255 0.0 0

- lo Link encap:Local Loopback inet addr:127.0.0.1 Mask:255.0.0.0 UP LOOPBACK RUNNING MTU:16436 Metric:1 RX packets:xx errors:0 dropped:0 overruns:0 frame:0 TX packets:xx errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:xxx (xxx.x B) TX bytes:xxx (xxx.x B)
- wlan0 Link encap:Ethernet HWaddr xx:xx:xx:xx:xx:xx inet addr:xxx.xxx.xxx Bcast:xxx.xxx.255 Mask:xxx.xxx.xxx.xxx UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:xx errors:0 dropped:0 overruns:0 frame:0 TX packets:xx errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:xxxx (x.x KiB) TX bytes:xxxxx (xx.x KiB)

Tip: The command "*sudo amixer cset numid=3 1*' will send sound to the headphone socket. Use "*sudo amixer cset numid=3 2*' to return it to HDMI. Use "*sudo amixer cget numid=3*" to fetch current setting.

Tip: To boot the Raspberry Pi in "**Safe Mode**" connect GPIO pins 5 & 6 with a shorting jumper.

Tip: To update the installed debian packages run "sudo apt-get update".

Tip: To *upgrade* the installed Raspbian operating system run "sudo aptget upgrade".

- **Tip:** To *update* the installed Raspberry Pi firmware perform the following: Install the "git-core" package if not already installed.
 - sudo apt-get install git-core

Install "rpi-update".

 sudo wget https://raw.github.com/Hexxeh/rpiupdate/master/rpi-update -0 /usr/bin/rpi-update && sudo chmod +x /usr/bin/rpi-update

Once installed run "sudo rpi-update".

v2.10 [How-to] Initial Startup Configuration

Note: To display the Paspberry Pi firmware version run "/opt/vc/bin/vcgencmd version".