

******Draft******

Custom Raspbian Image

~~image_2018-02-07-RaspbianUltibo.zip~~

image_2018-02-14-RaspbianUltibo.zip

Created with pi-gen

Testing Lazarus/FPC

Comparing number files needed

for VideoCore Apps

Raspbian & Ultibo

02/14/18

******Draft******

Introduction:

The overall goal is to create a bootable micro sd card ready to start developing Ultibo applications using Lazarus IDE (Ultibo Edition).with Free Pascal Compiler FPC. This was made possible with creation of the “*ultiboinstaller.sh*” found at

<https://github.com/ultibohub/Tools/blob/master/Installer/Core/Linux/ultiboinstaller.sh> by Gary from Ultibo, “*pi-gen*” and the “*Stretch Raspbain release*” make this possible. Ultibo released an installer script “utiboinstaller.sh” that installs Lazarus and Free Pascal Compiler FPC on the Raspberry Pi. This marks a milestone in the Ultibo Bare Metal development providing an Intregrated Development Enviornment (IDE). The Strech Raspbian release still needs serveral dependencies to install Lazarus IDE (Ultibo Edition).with Free Pascal Compiler FPC. In addition, other tools such as arm-none-eabi tools, telnet, tftp, gitk, diffuse, hexedit are desired.

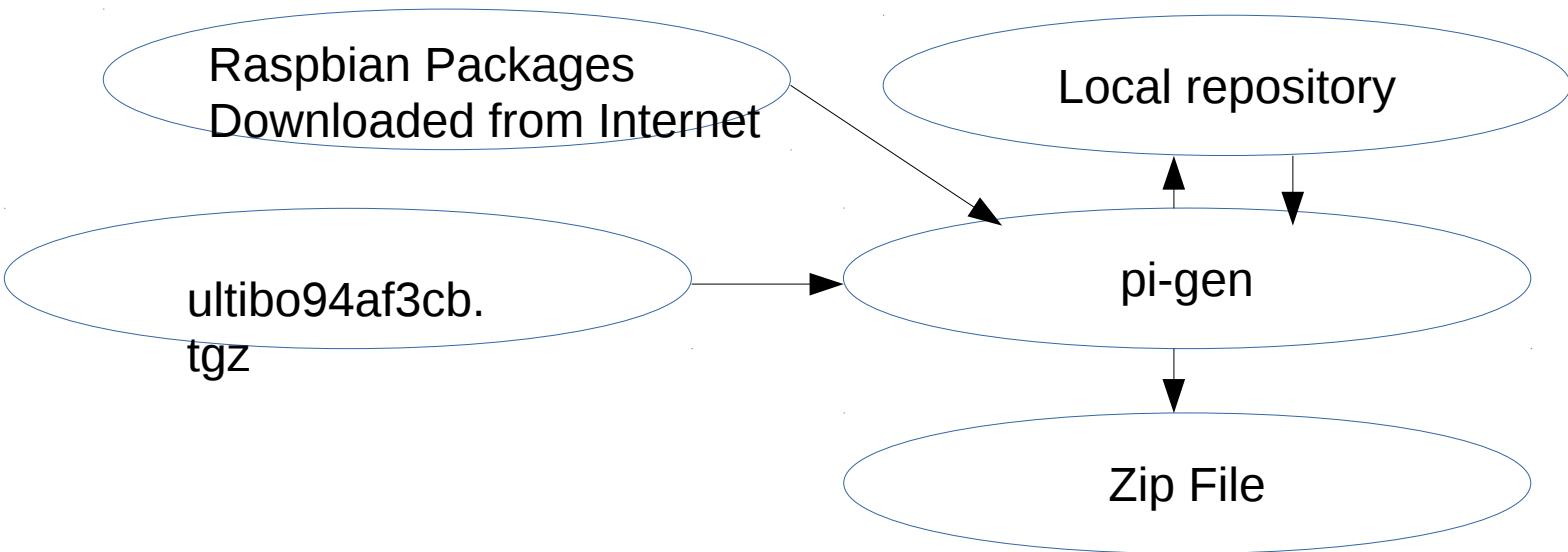
This is why pi-gen was used to get the system up quickly.

The target system is Linux based since the Linux firmware plus the kernel7.img or kernel.img are what make ultibo run on the RPi hardware.

We currently have a Lazarus/FPC executable that runs on Windows System.

This still requires the binutils and arm-none-eabi compiler.

Pi-gen installer



Creating the boot & rootfs in zip file that can be transferred to the micro sd card:
This required several steps found in [Appendix A: Notes during pi-gen testing](#)

Comparing number files needed for VideoCore Apps Raspbian & Ultibo.

Ultibo only requires a few files.

*bootcode.bin
start.elf
fixup.dat*

kernel.img RPi Zero or kernel7.img (RPi2B or RPi3B)

test.h264 needed for hello_videocube.bin

On Raspbian stretch requires

1313 packages

1400 packages with Ultibo dependencies

1420 packages with Ultibo & tools

Several thousand files are needed with an O/S

Below are the numbers of the main directories.

/usr/bin

1 directory, 1411 files

/usr/lib

3632 directories, 33202 files

sudo su

First need to compile the libraries

cd /opt/vc/src/hello_pi/libs/ilclient

make

cd /opt/vc/src/hello_pi/libs/vgfont

make

Next compile and test the apps

cd ../../hello_videocube/

make

./hello_videocube.bin

cd ../../hello_tiger/

make

./hello_tiger.bin

The following 4 images were complied on Rpi2B for a Rpi Zero. The first 2 are 2D image of Rotating Tiger and 3rd & 4th are of a Rotating cube.

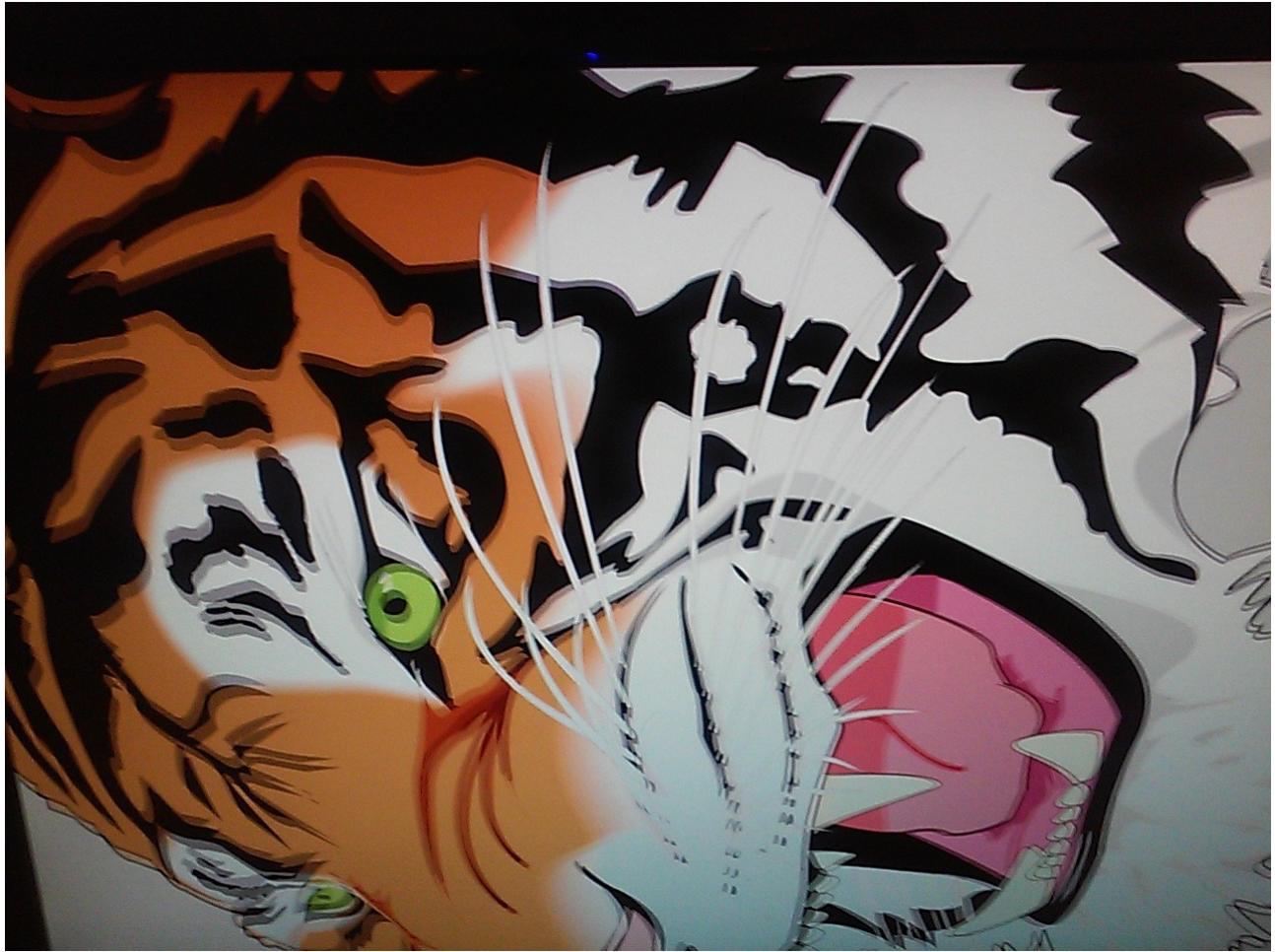


Image 1

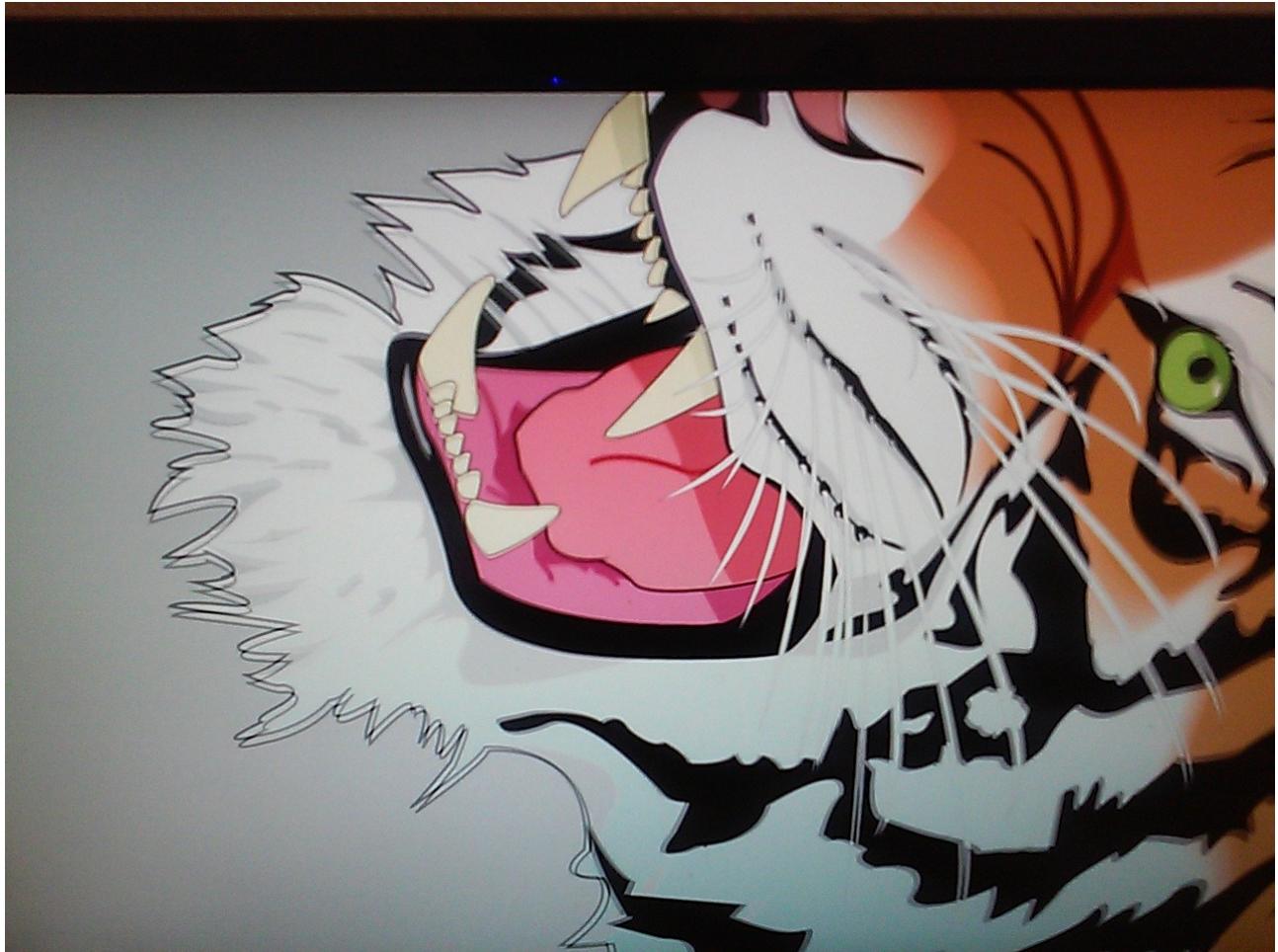


Image 2

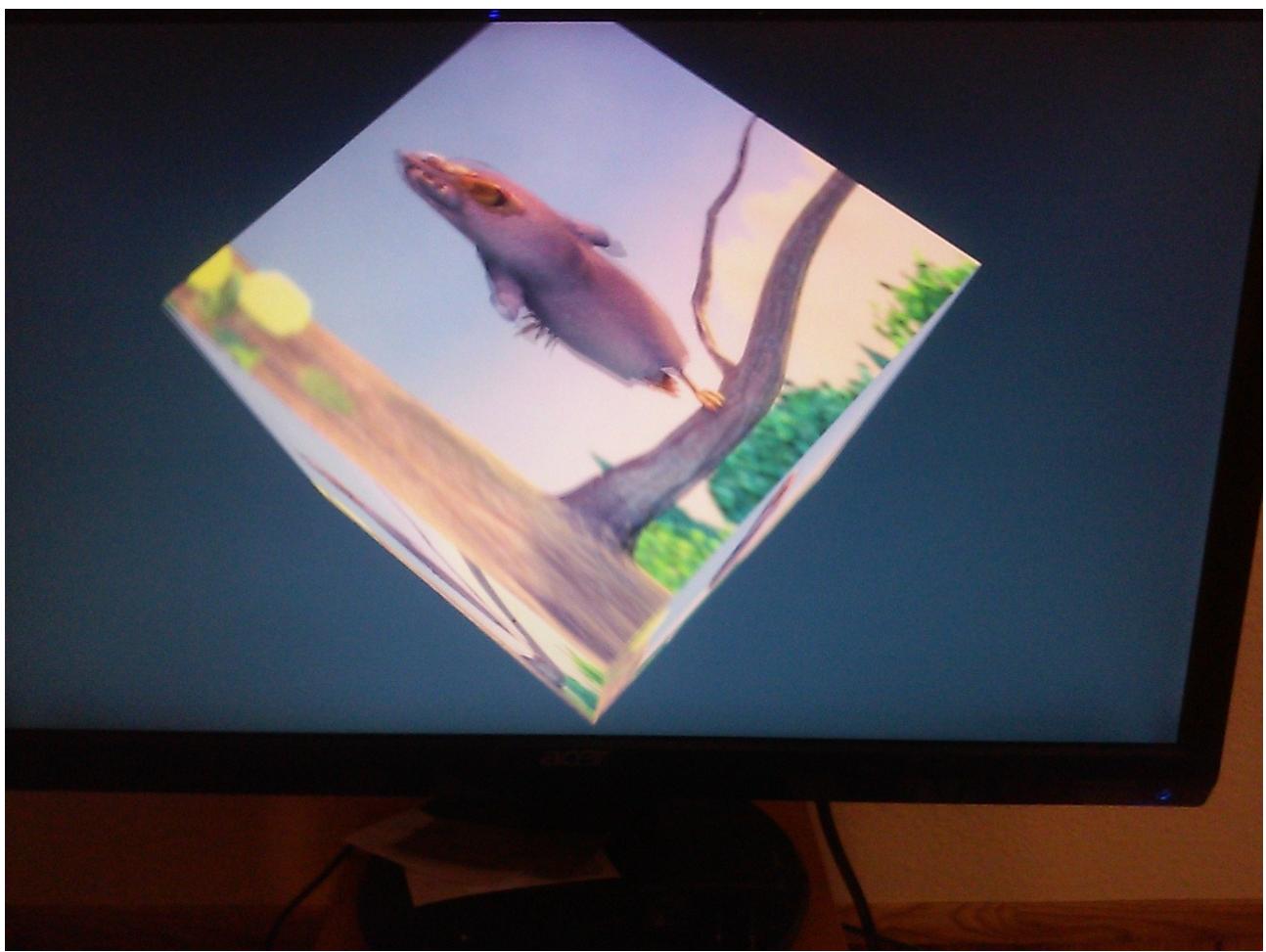


Image 3

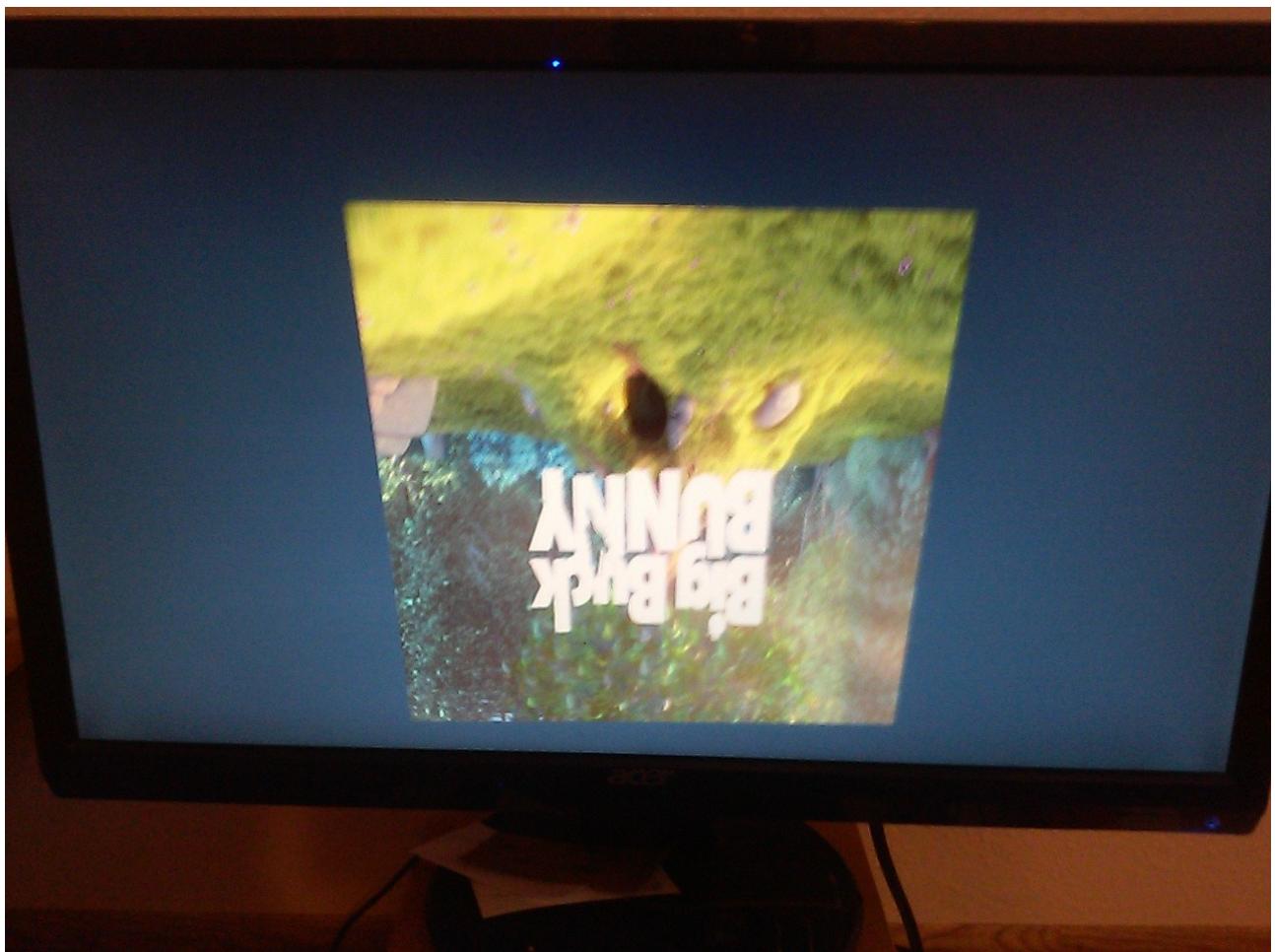


Image 4

Testing:

```
pi@raspberrypi:~ $ mkdir test_examples  
pi@raspberrypi:~ $ cd test_examples/  
pi@raspberrypi:~/test_examples $ git clone https://github.com/develone/Ultibo_PWM_TFTP.git
```

note

pi@raspberrypi:~ \$ **ultibo/core/startlazarus** was the command used to start the Lazarus IDE (Ultibo Edition).

This was required since a tar file **-rw-r--r-- 1 vidal vidal 329724698 Feb 7 17:19 ultibo.tgz** was used to install lazarus

```
pi@raspberrypi:~ $ time tar xfz ultibo.tgz
```

```
real 5m0.582s  
user 1m24.990s  
sys 0m44.640s  
ultibo  
2123 directories, 32499 files
```

From the Raspberry Pi/Programming/Lazarus IDE (Ultibo Edition)

If the PWM_TFTP_SERVO project was the current project Run/Compile

```
1 program PWM_TFTP_SERVO;  
.  
.  
{$mode objfpc}{$H+}  
.  
{ Advanced example - PWM_TFTP }  
.  
{ This example shows how to create webserver and TFTP server with remote shell }  
{ This version is for Raspberry Pi 2B and will also work on a 3B. }  
.  
{ After the first time that kernel7.img has been transferred to micro sd card }  
.  
{ tftp xx.xx.xx.xx < cmdstftp }  
.  
{ contents of cmdstftp }  
.  
{ binary }  
.  
{ put kernel7.img }  
.  
{ quit }  
.  
uses  
{InitUnit, {Include InitUnit to allow us to change the startup behaviour}  
RaspberryPi2, {Include RaspberryPi2 to make sure all standard functions are included}  
GlobalConfig,  
GlobalConst,  
GlobalTypes,  
Platform,  
Threads,  
Console,  
Classes,  
WebStatus,  
.  
.  
.  
uTFTP,  
Winsock2,  
{ needed to use ultibo-tftp }
```

1: 1 |INS |/home/pi/test_examples/Ultibo_PWM_TFTP/RPi2/PWM_TFTP_SERVO.lpr

Compile Project, OS: ultibo, Target: PWM_TFTP_SERVO: Success, Hints: 4

- ▶ PWM_TFTP_SERVO.lpr(44,2) Note: Local variable "MyPLoggingDevice" not used
- ▶ PWM_TFTP_SERVO.lpr(46,2) Note: Local variable "PWM1Device" not used
- ▶ PWM_TFTP_SERVO.lpr(49,2) Note: Local variable "Handle1" not used
- ▶ PWM_TFTP_SERVO.lpr(52,2) Note: Local variable "TCP" not used

Success Compile

If the compile works without errors a green bar wil appear as seen above.

Transferring the kernel7.img and firmware files displays the following when booting.

~~date > st.txt; ./ultiboinstaller.sh ; date > en.txt~~

~~pi@raspberrypi:~ \$ date > st.txt; ./ultiboinstaller.sh ; date > en.txt~~

~~Linux installer for Free Pascal and Lazarus (Ultibo edition)~~

This installation will download the sources for:

- ~~- Ultibo core~~
- ~~- Ultibo examples~~
- ~~- Free Pascal (Ultibo edition)~~
- ~~- Lazarus (Ultibo edition)~~

~~Then it will build all of the above, this will take several minutes to complete depending on the speed of your system.~~

~~The installation will not interfere with any existing development environments including other installations of Free Pascal and Lazarus.~~

Continue (y/n)? ~~y~~

~~^~~

~~Free Pascal and Lazarus (Ultibo edition) prerequisites~~

~~Installing and building Free Pascal requires several tools from the build essentials package including make, ld and as as well as the unzip utility.~~

~~These can be installed on Debian based distributions using:~~

~~sudo apt-get install build-essential unzip~~

~~Lazarus requires the GTK2 and X11 dev packages which can be installed on Debian based distributions by using:~~

~~sudo apt-get install libgtk2.0-dev libcairo2-dev \libpango1.0-dev libgdk-pixbuf2.0-dev libatk1.0-dev \libghe-x11-dev~~

~~Cross compiling Ultibo applications from Linux requires the arm-none-eabi build of the binutils package, this can be installed on Debian based distributions using:~~

~~sudo apt-get install binutils-arm-none-eabi~~

~~Press return to check for these prerequisites~~

```
make found
gdb found
unzip found
libgtk2.0-dev found
libcairo2-dev found
libpango1.0-dev found
libgdk-pixbuf2.0-dev found
libatk1.0-dev found
libghc-x11-dev found
arm-none-eabi-as found
arm-none-eabi-ld found
arm-none-eabi-objcopy found
```

~~Enter an installation folder or press return to accept the default install location~~

```
[/home/pi/ultibo/core]:  
The install folder will be:  
/home/pi/ultibo/core
```

~~Continue? (y/n): y~~

~~After install do you want a shortcut created in:
/home/pi/.local/share/applications (y/n)? y~~

After install do you want a shortcut created in:
/home/pi/.local/share/applications (y/n)? y
Downloading FPC minimal stable 3.0.2
Extracting FPC minimal stable 3.0.2
Downloading Ultibo core
Downloading Ultibo examples
Downloading Free Pascal (Ultibo edition)

~~This installer compiles many packages needed for Ultibo Edition.~~

~~Start: Thu 8 Feb 18:45:17 UTC 2018~~

~~Finish: Thu 8 Feb 20:53:29 UTC 2018~~

~~Takes twice as long on Rpi Zero as Rpi2B~~

~~Free Pascal and Lazarus (Ultibo edition) install complete~~

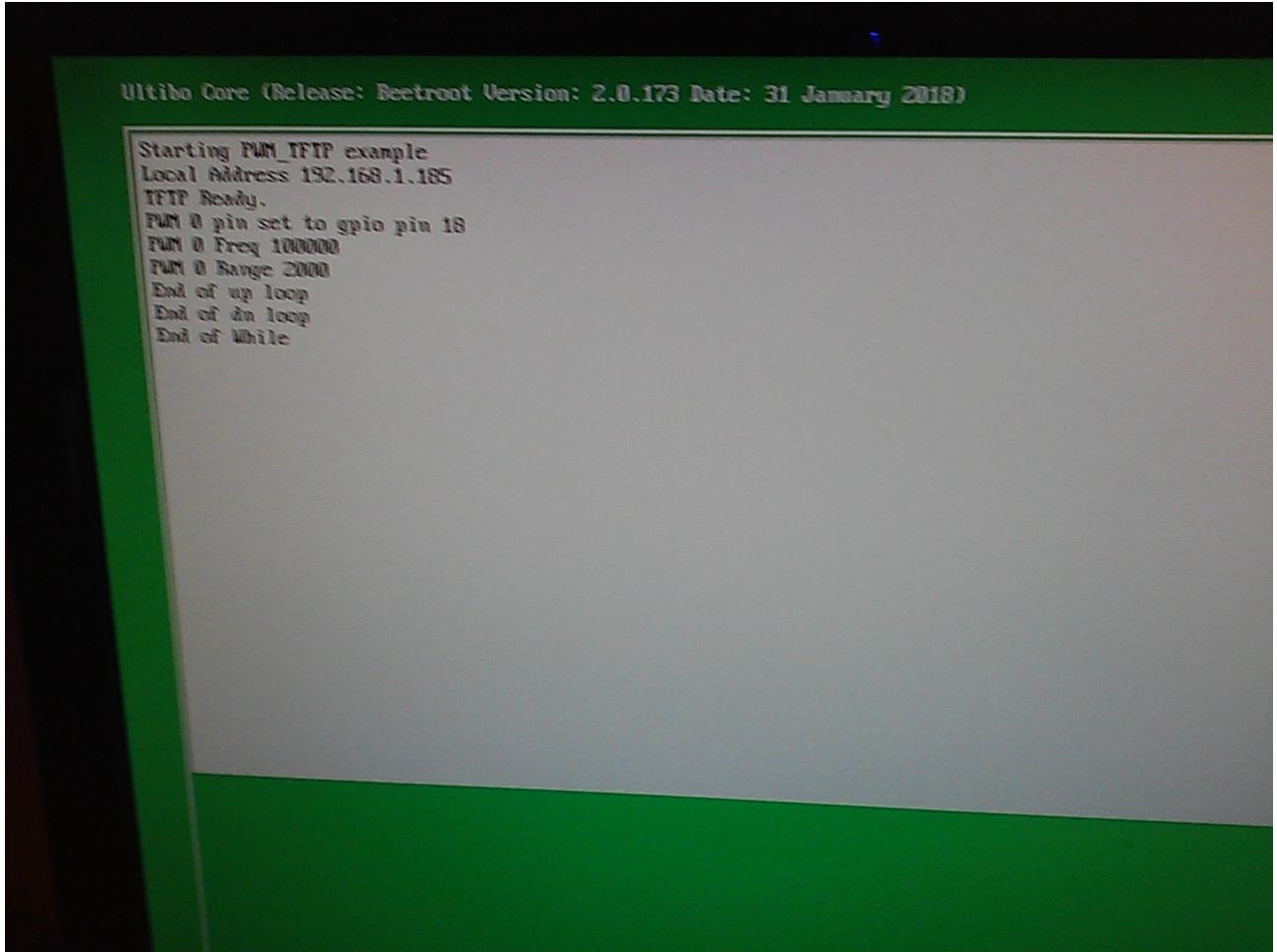
~~The Core was updated~~

~~ULTIBO_RELEASE_DATE = '31 January 2018';~~

~~ULTIBO_RELEASE_NAME = 'Beetroot';~~

~~ULTIBO_RELEASE_VERSION = '2.0.173';~~

~~The recently built target matches the values found in
ultibo/core/fpc/source/rtl/ultibo/core/globalconst.pas~~



pi@raspberrypi:~/test_examples/Ultibo_PWM_TFTP/RPi2 \$ cp kernel7.img /media/pi/6FCF-2BFD/

Rpi-Zero

prior to boot installed ultiboinstaller.sh and modified the dphys-swapfile

```
cd /media/vidal/rootfs/home/pi
cp /home/vidal/raspbian-pi-gen/ultiboinstaller.sh .
cd ../cd etc/
cp dphys-swapfile dphys-swapfile.orig
nano dphys-swapfile
diff dphys-swapfile dphys-swapfile.orig
16c16
< CONF_SWAPSIZE=1000
---
> CONF_SWAPSIZE=100
```

pi@raspberrypi:~ \$ dpkg -l | wc

1420 14441 203159

chg passwd for pi

setup wifi

128 GPU

enable ssh

Rpi-Zero running the ultiboinstaller.sh

```
date > st.txt; ./ultiboinstaller.sh ; date > en.txt  
pi@raspberrypi:~ $ date > st.txt; ./ultiboinstaller.sh ; date > en.txt
```

Linux installer for Free Pascal and Lazarus (Ultibo edition)

This installation will download the sources for:

- Ultibo core
- Ultibo examples
- Free Pascal (Ultibo edition)
- Lazarus (Ultibo edition)

Then it will build all of the above, this will take several minutes to complete depending on the speed of your system.

The installation will not interfere with any existing development environments including other installations of Free Pascal and Lazarus.

Continue (y/n)? y

+

Free Pascal and Lazarus (Ultibo edition) prerequisites

Installing and building Free Pascal requires several tools from the build-essentials package including make, ld and as as well as the unzip utility.

These can be installed on Debian based distributions using:

```
sudo apt-get install build-essential unzip
```

Lazarus requires the GTK2 and X11 dev packages which can be installed on Debian based distributions by using:

```
sudo apt-get install libgtk2.0-dev libcairo2-dev  
libpango1.0-dev libgdk-pixbuf2.0-dev libatk1.0-dev  
libghe-x11-dev
```

Cross compiling Ultibo applications from Linux requires the arm-none-eabi build of the binutils package, this can be installed on Debian based distributions using:

```
sudo apt-get install binutils-arm-none-eabi
```

Press return to check for these prerequisites

```
make found
gdb found
unzip found
libgtk2.0-dev found
libcairo2-dev found
libpango1.0-dev found
libgdk-pixbuf2.0-dev found
libatk1.0-dev found
libghc-x11-dev found
arm-none-eabi-as found
arm-none-eabi-ld found
arm-none-eabi-objcopy found
```

Enter an installation folder or press return to
accept the default install location

[/home/pi/ultibo/core]:
The install folder will be:
/home/pi/ultibo/core

Continue? (y,n): y

After install do you want a shortcut created in:
/home/pi/.local/share/applications (y/n)? y

After install do you want a shortcut created in:

```
/home/pi/.local/share/applications (y/n)? y  
Downloading FPC minimal stable 3.0.2  
Extracting FPC minimal stable 3.0.2  
Downloading Ultibo core  
Downloading Ultibo examples  
Downloading Free Pascal (Ultibo edition)
```

This installer compiles many packages needed for Ultibo Edition.

~~Start: Thu 8 Feb 18:45:17 UTC 2018~~

~~Finish: Thu 8 Feb 20:53:29 UTC 2018~~

~~Takes twice as long on Rpi Zero as Rpi2B~~

Free Pascal and Lazarus (Ultibo edition) install complete

~~Launch Lazarus from the application shortcut or by using~~

~~/home/pi/ultibo/core/lazarus.sh~~

~~from the command line~~

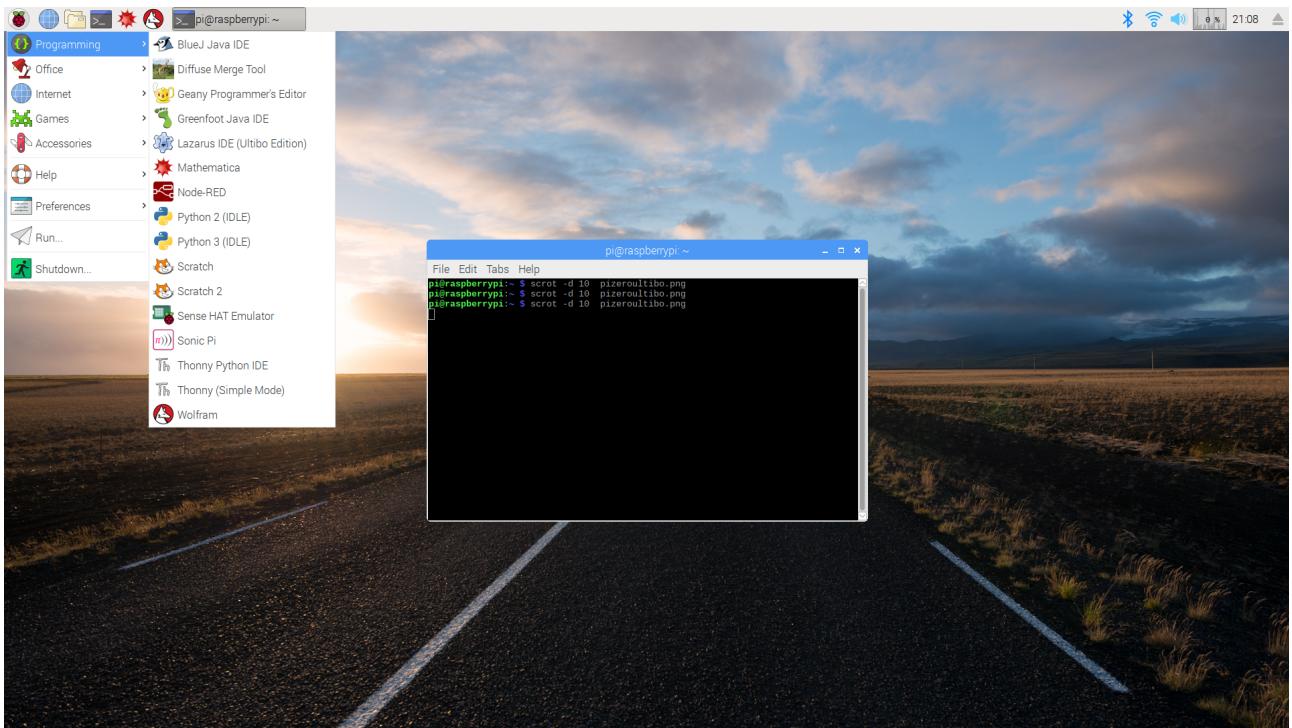
Thu 8 Feb 20:01:10 UTC 2018

```
[ 11%] Compiled package fel-xml  
Start compiling package chm for target arm-ultibo.  
--- Compiling chm/BuildUnit_chm.pp  
--- Compiling ./chm/src/chmbase.pas  
--- Compiling ./chm/src/chmtypes.pas  
--- Compiling ./chm/src/chmspecialfiles.pas  
--- Compiling ./chm/src/paslznonslide.pas  
--- Compiling ./chm/src/paslzxcomp.pas  
--- Compiling ./chm/src/fasthtmlparser.pas  
--- Compiling ./chm/src/htmlutil.pas  
--- Compiling ./chm/src/htmlindexer.pas  
--- Compiling ./chm/src/chmfiftimain.pas  
--- Compiling ./chm/src/chmwriter.pas  
--- Compiling ./chm/src/chmsitemap.pas  
--- Compiling ./chm/src/lzxcompressthread.pas  
--- Compiling ./chm/src/chmfilewriter.pas  
--- Compiling ./chm/src/paslzx.pas  
--- Compiling ./chm/src/chmreader.pas  
--- Compiling ./chm/src/itolitlstyles.pas  
--- Compiling ./chm/src/itsftransform.pas  
--- Compiling ./chm/src/itolitlsreader.pas  
[ 12%] Compiled package chm  
[ 12%] Skipped package cocoaint which has been disabled for target arm-ultibo  
[ 14%] Skipped package dbus which has been disabled for target arm-ultibo  
[ 15%] Skipped package dts which has been disabled for target arm-ultibo  
Start compiling package fastegi for target arm-ultibo.  
--- Compiling fastegi/src/fastegi.pp  
[ 15%] Compiled package fastegi
```

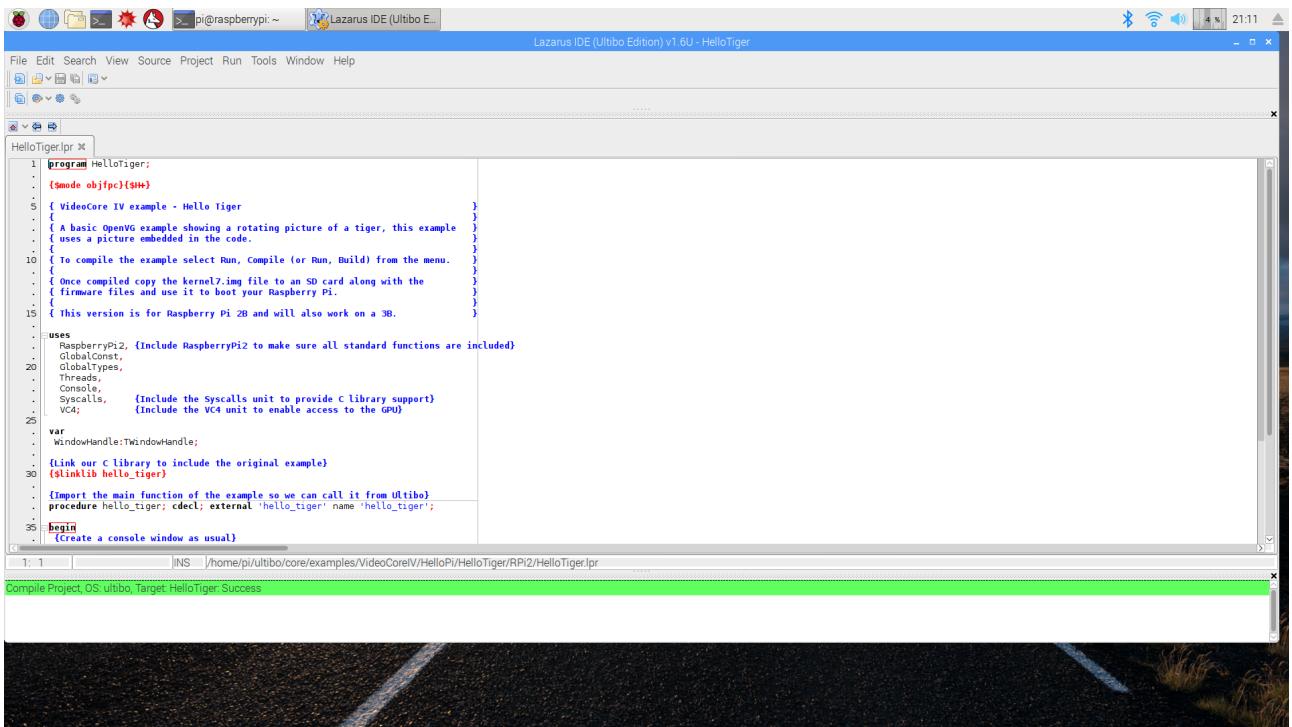
```
[ 16%] Skipped package fcl-async which has been disabled for target arm-ultibo
Start compiling package sqlite for target arm-ultibo.
--- Compiling sqlite/BuildUnit_sqlite.pp
--- Compiling ./sqlite/src/sqlite3.pp
--- Compiling ./sqlite/src/sqlite3db.pas
[ 17%] Compiled package sqlite
Start compiling package fcl-json for target arm-ultibo.
--- Compiling fcl-json/BuildUnit_fcl_json.pp
--- Compiling ./fcl-json/src/fjson.pp
--- Compiling ./fcl-json/src/jsonscanner.pp
--- Compiling ./fcl-json/src/jsonparser.pp
--- Compiling ./fcl-json/src/jsonconf.pp
--- Compiling ./fcl-json/src/fjsonrtti.pp
-
Thu 8 Feb 20:24:17 UTC 2018
[ 82%] Compiled package reactivex
[ 82%] Skipped package rexx which has been disabled for target arm-ultibo
Start compiling package rtl-unicode for target arm-ultibo.
--- Compiling rtl-unicode/BuildUnit rtl_unicode.pp
--- Compiling ./rtl-unicode/src/inc/unicodedetect.pas
--- Compiling ./rtl-unicode/src/collations/buildcollations.pas
--- Compiling ./rtl-unicode/src/collations/collation_de.pas
--- Compiling ./rtl-unicode/src/collations/collation_es.pas
--- Compiling ./rtl-unicode/src/collations/collation_fr_ca.pas
--- Compiling ./rtl-unicode/src/collations/collation_ja.pas
--- Compiling ./rtl-unicode/src/collations/collation_ko.pas
--- Compiling ./rtl-unicode/src/collations/collation_ru.pas
--- Compiling ./rtl-unicode/src/collations/collation_sv.pas
```

Rpi-Zero Starting Lazarus

From the Raspberry Pi/Programming/Lazarus IDE (Ultibo Edition)



Starting Lazarus



Compiling HelloTiger on Rpi-Zero

Success Compile

If the compile works without errors a green bar wil appear as seen above.

```
pi@raspberrypi:~/ultibo/core/examples/VideoCoreIV>HelloPi>HelloTiger/RPi2>HelloTiger.lpr
pi@raspberrypi:~/ultibo/core/examples/VideoCoreIV>HelloPi>HelloTiger/RPi2>HelloTiger.lpr$ cp kernel7.img
/media/pi/6FCF-2BFD/
```

The screenshot shows the Lazarus IDE interface with the title "Lazarus IDE (Ultibo Edition) v1.6U - HelloVideocube". The main window displays the source code for "HelloVideocube.lpr". The code includes comments explaining the compilation process and hardware requirements. Below the code editor, the status bar shows "1 1 INS /home/pi/ultibo/core/examples/VideoCoreIV>HelloPi/HelloVideocube/RPi2>HelloVideocube.lpr". A green bar at the bottom indicates a successful compilation: "Compile Project, OS: ultibo, Target: HelloVideocube: Success".

```

1 program HelloVideocube;
. . .
5 { VideoCore IV example - Hello Videocube
. . .
10 { Combining the 3D cube from hello triangle with video from the previous
. . .
15 { example, very nice demonstration.
. . .
20 { To compile the example select Run, Compile (or Run, Build) from the menu.
. . .
25 { Once compiled copy the kernel7.img file to an SD card along with the
{ Firmware files and use it to boot your Raspberry Pi.
. . .
30 { Make sure you also copy the test.h264 file from the Media folder.
. . .
35 { You also MUST create a config.txt file in the root directory of your SD card
{ with at least the following setting:
. . .
40 { gpu_mem=128
. . .
45 { this version is for Raspberry Pi 2B and will also work on a 3B.
. . .

uses
50 RaspberryPi2, {Include RaspberryPi2 to make sure all standard functions are included}
. . .
55 GlobalConst,
. . .
58 GlobalTypes,
. . .
60 Threads,
. . .
62 Console,
. . .
64 SysCalls,
. . .
66 SysCalls,
. . .
70 Vc4; {Include the Syscalls unit to provide C library support}
. . .
75 Vc4; {Include the VC4 unit to enable access to the GPU}

var
80 WindowHandle:TWindowHandle;
. . .

```

Compiling HelloVideocube on Rpi-Zero

Success Compile

If the compile works without errors a green bar wil appear as seen above.

```

pi@raspberrypi:~/ultibo/core/examples/VideoCoreIV>HelloPi/HelloVideocube/RPi2 $ cp
kernel7.img /media/pi/6FCF-2BFD/
pi@raspberrypi:~/ultibo/core/examples/VideoCoreIV>HelloPi/Media $ cp test.h264
/media/pi/6FCF-2BFD/

```

Both of examples created on the Rpi Zero ran correctly with on a few files running Ultibo.

Note:

Even on the RPI Zero the compile was very quick for the HelloTiger & HelloVideocube projects.

Appendix A: Notes during pi-gen testing

pi-gen Tool used to create the raspberrypi.org Raspbian images.

Forked pi-gen <https://github.com/RPi-Distro/pi-gen>
commit 066eb03d52868290661e813738a9a66eda263aa9
Author: Ben Pirt <ben@pirt.co.uk>
Date: Wed Jan 3 12:48:31 2018 +0000

Allow image building to be skipped for stages (#137)

pi-gen
GitHub - RPi-Distro/pi-gen: Tool used to create the raspberrypi.org ...
<https://github.com/RPi-Distro/pi-gen>

README.md. pi-gen. Tool used to create the raspberrypi.org Raspbian images.
Dependencies. pi-gen runs on Debian based operating systems. Currently it
is only supported on either Debian Stretch or Ubuntu Xenial and is known
to have issues building on earlier releases of these systems. To install
the required

On a Ubuntu 16.04 added pi-gen dependencies.

```
sudo apt-get install quilt parted realpath qemu-user-static debootstrap zerofree pxz zip dosfstools  
bsdtar libcap2-bin grep rsync xz-utils
```

Linux sim2 4.4.0-112-generic #135-Ubuntu SMP Fri Jan 19 11:48:36 UTC 2018 x86_64 x86_64
x86_64 GNU/Linux

AMD FX(tm)-4130 Quad-Core Processor
cpu MHz : 1800.000

Steps create the pi-gen/work/2018-02-07-RaspbianUltibo tree.

1.0)

```
git clone https://github.com/develone/pi-gen.git
```

2.0)

```
cd pi-gen/
```

3.0)

```
cp ~/raspbian-pi-gen/config .
```

Added a file pi-gen/config

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

```
export IMG_NAME=RaspbianUltibo
```

4.0)

```
cp ~/raspbian-pi-gen/stage4/00-install-packages/02-packages stage4/00-install-packages/
```

5.0)

```
cp ~/raspbian-pi-gen/ultibo_laz_fpc/ultibo94af3cb.tgz ~pi-gen/stage4/02-extras/files
```

Note: The file raspbian-pi-gen/ultibo_laz_fpc/ultibo94af3cb.tgz is not

in the raspbian-pi-gen repository due to the size of the tar file.

331297115 Feb 14 13:32 ultibo94af3cb.tgz

md5sum

e3b54b45689a14e67440f4ea428efc5a ultibo94af3cb.tgz

Adding the script in stage4/02-extras that un tar the ultibo tree which provides

lazarus. Uses the file ultibo94af3cb.tgz

Adding the short cut to start Lazarus

6.0)

```
cp ~/raspbian-pi-gen/stage4/02-extras/01-run.sh ~pi-gen/stage4/02-extras/
```

7.0)

```
date > t1.txt; sudo ./build.sh ; date > t2.txt
```

Wed Feb 14 11:35:50 MST 2018

Wed Feb 14 15:36:39 MST 2018

Note: The 3 hours and 1 min could be reduced with the use of local repository.

du -s local-apt-repo/

1513164 local-apt-repo/

The download of required packages takes considerable time.

8.0)

```
cd deploy/
```

9.0)

```
fdisk /dev/sdb
10.)
date > t1.txt; gzip -dc image_2018-02-14-RaspbianUltibo.zip | dd bs=16M of=/dev/sdb; date >>
t1.txt
0+165834 records in
0+165834 records out
7730102272 bytes (7.7 GB, 7.2 GiB) copied, 696.3 s, 11.1 MB/s
Wed Feb 14 15:43:22 MST 2018
Wed Feb 14 15:54:58 MST 2018
```

The stretch build has 1313 pkgs installed
The 2018-02-07-RaspbianUltibo build has 1420 pkgs installed

```
ssh -Y pi@192.168.1.185
uname -a
Linux raspberrypi 4.9.59-v7+ #1047 SMP Sun Oct 29 12:19:23 GMT 2017 armv7l GNU/Linux
```

Testing new build with the dependencies need to install the Lazarus/FPC

```
Added a file pi-gen/stage4/00-install-packages/02-packages
libgtk2.0-dev
libcairo2-dev
libpango1.0-dev
libgdk-pixbuf2.0-dev
libatk1.0-dev
libghc-x11-dev
binutils-arm-none-eabi
gcc-arm-none-eabi
```

```
./build.sh
Creates the pi-gen/work/2018-02-06-RaspbianUltibo tree.
vidal@sim2:~$ ls pi-gen/work/2018-02-07-
RaspbianUltibo/stage0/rootfs/var/cache/apt/archives/*.deb | wc
    220    220   15327
vidal@sim2:~$ ls pi-gen/work/2018-02-07-
RaspbianUltibo/stage1/rootfs/var/cache/apt/archives/*.deb | wc
      15     15   1613
vidal@sim2:~$ ls pi-gen/work/2018-02-07-
RaspbianUltibo/stage2/rootfs/var/cache/apt/archives/*.deb | wc
    214    214   15221
vidal@sim2:~$ ls pi-gen/work/2018-02-07-
RaspbianUltibo/stage3/rootfs/var/cache/apt/archives/*.deb | wc
    427    427   30630
vidal@sim2:~$ ls pi-gen/work/2018-02-07-
RaspbianUltibo/stage4/rootfs/var/cache/apt/archives/*.deb | wc
    536    536   38434
vidal@sim2:~$ ls pi-gen/work/2018-02-07-
RaspbianUltibo/stage5/rootfs/var/cache/apt/archives/*.deb | wc
      9      9    664
1421 packages are required
```

```
pi@raspberrypi:~ $ dpkg -l | wc  
    1420 14224 200233  
pi@raspberrypi:~ $ uname -a  
Linux raspberrypi 4.9.59-v7+ #1047 SMP Sun Oct 29 12:19:23 GMT 2017 armv7l GNU/Linux  
  
date > t1.txt; ./ultiboinstaller.sh ; date > t2.txt  
Tue 6 Feb 23:47:19 UTC 2018  
Wed 7 Feb 00:52:52 UTC 2018
```

Free Pascal and Lazarus (Ultibo edition) prerequisites

Installing and building Free Pascal requires several tools from the build essentials package including make, ld and as as well as the unzip utility.

These can be installed on Debian based distributions using:

```
sudo apt-get install build-essential unzip
```

Lazarus requires the GTK2 and X11 dev packages which can be installed on Debian based distributions by using:

```
sudo apt-get install libgtk2.0-dev libcairo2-dev \  
libpango1.0-dev libgdk-pixbuf2.0-dev libatk1.0-dev \  
libghc-x11-dev
```

Cross compiling Ultibo applications from Linux requires the arm-none-eabi build of the binutils package, this can be installed on Debian based distributions using:

```
sudo apt-get install binutils-arm-none-eabi
```

Press return to check for these prerequisites

```
make found  
gdb found  
unzip found  
libgtk2.0-dev found  
libcairo2-dev found  
libpango1.0-dev found  
libgdk-pixbuf2.0-dev found  
libatk1.0-dev found  
libghc-x11-dev found  
arm-none-eabi-as found  
arm-none-eabi-ld found  
arm-none-eabi-objcopy found
```

Enter an installation folder or press return to accept the default install location

[/home/pi/ultibo/core]:

The install folder will be:

/home/pi/ultibo/core

Continue? (y,n): y

After install do you want a shortcut created in:

/home/pi/.local/share/applications (y/n)? y

Downloading FPC minimal stable 3.0.2

Extracting FPC minimal stable 3.0.2

Downloading Ultibo core

Downloading Ultibo examples

Downloading Free Pascal (Ultibo edition)

Next step required

Need to create a ultibo-laz-fpc.deb file.

This will based on ultibo tree,