Resources for PaPiRus ePaper eInk display https://www.pi-supply.com/product/pap...

README.md

PaPiRus

Resources for PaPiRus ePaper eInk displays. This repository is based on, and makes use of, the rePaper/gratis GitHub repository.

Setup PaPiRus

Auto Installation

Just run the following script in a terminal window and PaPiRus will be automatically setup.

```
# Run this line and PaPiRus will be setup and installed
curl -sSL https://pisupp.ly/papiruscode | sudo bash
```

Manual Installation

If you have any troubles with the auto installation or if for some reason you prefer to install PaPiRus manually, then follow

the steps below.

Enabling SPI and I2C interfaces on Raspberry Pi

Before using PaPiRus, do not forget to enable the SPI and the I2C interfaces. You can enable the SPI by typing `sudo raspi-config` at the command line and then selecting `Interfacing options > SPI` and then selecting Enable. Without exiting the tool still in `Interfacing options > I2C` and then selecting Enable.

Install Python API (best to run all of these commands as root using sudo)

```
# Install dependencies
apt-get install git -y
apt-get install python-imaging -y
apt-get install python-smbus -y
apt-get install bc i2c-tools -y

git clone https://github.com/PiSupply/PaPiRus.git
cd PaPiRus
sudo python setup.py install    # Install PaPirRus python library
```

Install Driver (Option 1) (best to run all of these commands as root using sudo)

```
sudo papirus-setup    # This will auto install the driver
```

Install Driver (Option 2) (best to run all of these commands as root using sudo)

```
# Install fuse driver
sudo apt-get install libfuse-dev -y

mkdir /tmp/papirus
cd /tmp/papirus
git clone https://github.com/repaper/gratis.git

cd /tmp/papirus/gratis
make rpi EPD_IO=epd_io.h PANEL_VERSION='V231_G2'
make rpi-install EPD_IO=epd_io.h PANEL_VERSION='V231_G2'
systemctl enable epd-fuse.service
systemctl start epd-fuse
```

Select your screen size

```
sudo papirus-set [1.44 | 1.9 | 2.0 | 2.6 | 2.7 ]
or
sudo papirus-config
```

Python API

The Basic API

```
from papirus import Papirus
```

```
# The epaper screen object.
# Optional rotation argument: rot = 0, 90, 180 or 270
screen = Papirus([rotation = rot])

# Write a bitmap to the epaper screen
screen.display('./path/to/bmp/image')

# Perform a full update to the screen (slower)
screen.update()

# Update only the changed pixels (faster)
screen.partial_update()

# Disable automatic use of LM75B temperature sensor
screen.use_lm75b = False

# Change screen size
# SCREEN SIZES 1_44INCH | 1_9INCH | 2_0INCH | 2_6INCH | 2_7INCH
screen.set_size(papirus.2_7INCH) (coming soon)
```

The Text API

```
from papirus import PapirusText

text = PapirusText([rotation = rot])

# Write text to the screen
# text.write(text)
text.write("hello world")
```

The Positional Text API (example 1)

```
from papirus import PapirusTextPos

# Same as calling "PapirusTextPos(True [,rotation = rot])"
text = PapirusTextPos([rotation = rot])

# Write text to the screen at selected point, with an Id
# "hello world" will appear on the screen at (10, 10), font size 20, straight away
text.AddText("hello world", 10, 10, Id="Start" )

# Add another line of text, at the default location
# "Another line" will appear on screen at (0, 0), font size 20, straight away
text.AddText("Another line", Id="Top")

# Update the first line
# "hello world" will disappear and "New Text" will be displayed straight away
text.UpdateText("Start", "New Text")

# Remove The second line of text
# "Another line" will be removed from the screen straight away
text.RemoveText("Top")

# Clear all text from the screen
# This does a full update so is a little slower than just removing the text.
text.Clear()
```

The Positional Text API (example 2)

```
from papirus import PapirusTextPos

# Calling PapirusTextPos this way will mean nothing is written to the screen be default
text = PapirusTextPos(False [,rotation = rot])

# Write text to the screen at selected point, with an Id
# Nothing will show on the screen
text.AddText("hello world", 10, 10, Id="Start" )

# Add another line of text, at the default location
# Nothing will show on the screen
text.AddText("Another line", Id="Top")

# Now display BOTH lines on the scrren
text.WriteAll()

# Update the first line
# No change will happen on the screen
text.UpdateText("Start", "New Text")

# Remove The second line of text
# The text won't be removed just yet from the screen
text.RemoveText("Top")

# Now update the screen to show the changes
text.WriteAll()
```

Notes

PaPiRusTextPos will take in to account \n as a line break (or multiple line breaks) Meaning text will be aligned to the X position given, it will not return to x=0 for the start of the next line.

Unicode Support in the Text API

```
from papirus import PapirusText

text = PapirusText()

# Write text to the screen, in this case forty stars alternating black and white
# note the use of u"" syntax to specify unicode
text.write(u"\u2605 \u2606 \u2605 \u2606 \u2605 \u2606 \u2605 \u2606 \u2605 \u2606 \u2605 \u2606 \u2605
```

Note

The default font, FreeMono, has limited unicode support, so you may want to specify an alternate font to use a fuller range characters.

The Image API

```
from papirus import PapirusImage

image = PapirusImage([rotation = rot])
```

```
# easy write image to screen
# image.write(path)
image.write('/path/to/image')

# write image to the screen with size and position
# image.write(path, width, (x,y))
image.write('/path/to/image', 20, (10, 10) ) # This is not confirmed to work correctly yet!!
```

The composite API (Text and image)

```
from papirus import PapirusComposite

# Calling PapirusComposite this way will mean nothing is written to the screen until WriteAll is called
textNImg = PapirusComposite(False)

# Write text to the screen at selected point, with an Id
# Nothing will show on the screen
textNImg.AddText("hello world", 10, 10, Id="Start" )

# Add image
# Nothing will show on the screen
# textNImg.AddImg(path, posX,posY,(w,h),id)
textNImg.AddImg("/path/to/image",20,20,(25,25), Id="BigImg")

# Add image to the default place and size
# Nothing will show on the screen
textNImg.AddImg("/path/to/image", Id="Top")

# Now display all elements on the scrren
textNImg.WriteAll()

# Update the first line
# No change will happen on the screen
textNImg.UpdateText("Start", "New Text")

# Update the BigImg
# No change will happen on the screen
textNImg.UpdateImg("BigImg", "/path/to/new/images")

# Remove top image
# The images won't be removed just yet from the screen
textNImg.RemoveImg("Top")

# Now update the screen to show the changes
textNImg.WriteAll()
```

Font family

PaPiRusText and PaPiRusTextPos are using the font *FreeMono.ttf* by default. It is possible to specify the argument `font_path` in `PapirusText.write`, `PapirusTextPos.AddText`, `PapirusTextPos.UpdateText` and `PapirusTextPos.addToImageText` to change the *font family*. The argument must be a string containing the path to the *.ttf* file.

```
# Change font family
from papirus import PapirusText
```

```
text = PapirusText()
text.write("Hello World", font_path='/path/to/ttf')
```

Command Line

```
# Set the screen size you are using
papirus-set [1.44 | 1.9 | 2.0 | 2.6 | 2.7 ]

# Write data to the screen
papirus-write "Some text to write" [-x ] [-y ] [-fsize ] [-rot]

# Draw image on the screen
papirus-draw /path/to/image -t [resize | crop] -r [0 | 90 | 180 | 270]

# Clear the screen
papirus-clear
```

Demos

All demos can be seen by running the following commands. Code can be found in the repo for the python bin directory.

```
# Show clock
papirus-clock [rotation]

# Run game of life
papirus-gol

# Show system information
papirus-system (coming soon)

# Push framebuffer to screen
papirus-framepush (coming soon)

# Demo of using the buttons
papirus-buttons [rotation]

# Demo of getting temperature from LM75
papirus-temp

# Demo showing depdency of update rate on temperature
papirus-radar

# Display text filling the width of the display
papirus-textfill 'Some text' [rotation]

# Snakes game
papirus-snakes (coming soon)

# Display Twitter feeds
papirus-twitter

# Composite text and graphics
papirus-composite-write
```

Tips for using images

The PaPiRus can only display Bitmap images (.BMP) in black and white (1 bit colour). If you pass an image to PaPiRus that is not a 1 Bit Bitmap, it will automatically be converted to this by the software. However, for best results and higher image quality we would recommend that you convert the image to a 1 Bit Bitmap before pushing to the PaPiRus screen using GIMP or Photoshop or similar photo editing tools like the rePaper companion to resize images and convert them to XBM format or WIF (the WyoLum Image Format).

Screen Resolutions

The screens have the following screen resolutions:

```
1.44"     128 x 96
1.9"      144 x 128
2.0"      200 x 96
2.6"      232 x 128
2.7"      264 x 176
```

Full and Partial Updates

Also try using the method partial_update() instead of the update() one if you want to refresh the screen faster and maybe want to create some simple animations. Remember though that the partial method cannot be used indefinitely and you will have to refresh the screen every once in a while. You should ideally do a full refresh of the screen every few minutes and it is also recommended to completely power down the screen every few hours.

Hardware tips

In case you have problems assembling the board please check this article on our website on which you can find:

- Connect the screen to the PaPiRus board
- Connect the GPIO adapter
- Install the pogo pin connector
- Install the push buttons Not all the sections apply to both the PaPiRus HAT and the PaPiRus Zero.

Datasheets, connectivity, pinout, jumpers and further information

For additional information follow the links below:

- PaPiRus HAT
- PaPiRus Zero
- Pinout.xyz resources

Third party software libraries

It is safe to say we have an awesome and growing community of people using epaper with PaPiRus and beyond and we get a huge amount of contributions of code, some of which we can easily integrate here and others which we can't (we are only a small team). However we want to make sure that any contributions are easy to find, for anyone looking. So here is a list of other software libraries that might be useful to you:

- Go software library for driving PaPiRus

- RISC OS software library for driving PaPiRus
- PaPiRus HAT working with resin.io