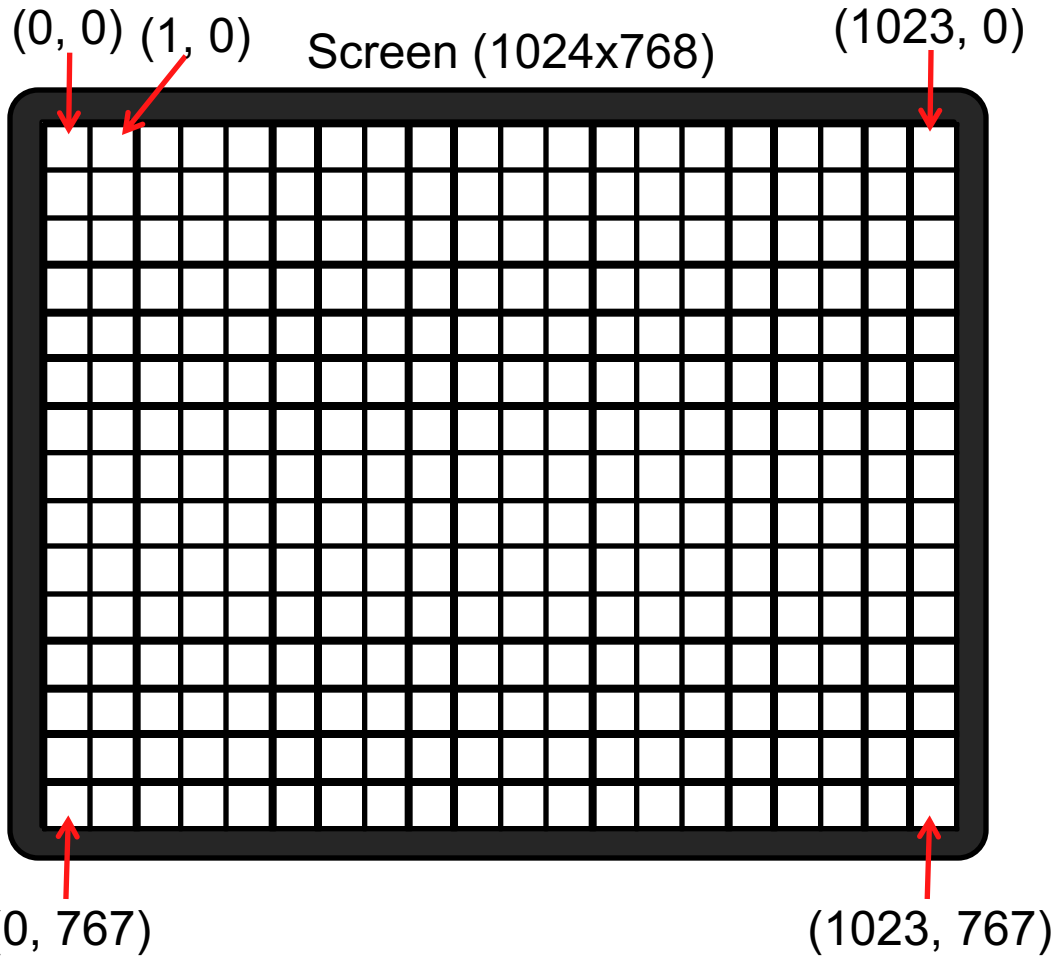
CPSC 359 – Tutorial #7

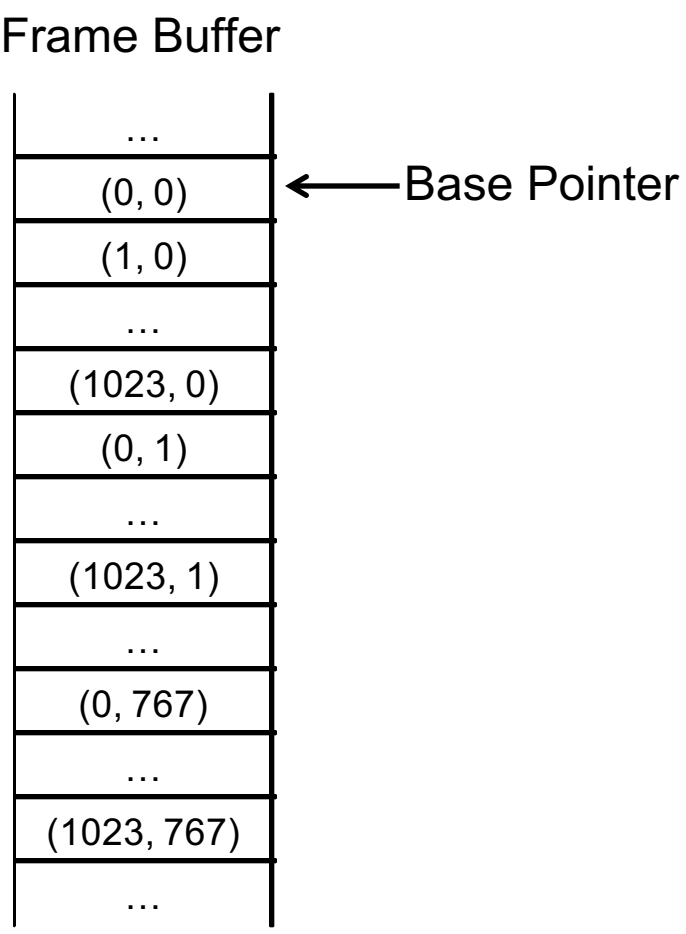
Video Interface

Modified from Andrew Kuipers

Updated for RPi3 / Winter 2018

Video Interface



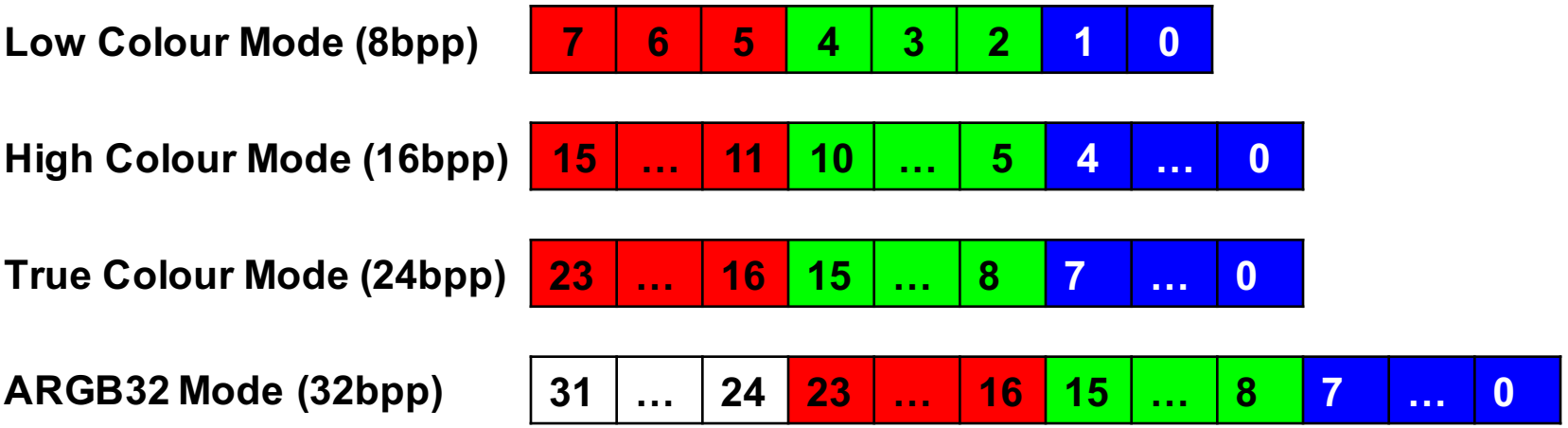


Video Interface

- Draw pixels by writing colour values to the Frame Buffer

  addr(x, y) = base pointer + ((y * width) + x) * (bpp / 8)

- Colour value is split into Red, Green and Blue colour channels
  - Higher values in a channel mean more of that colour



Video Interface

- Before we can draw pixels, we need to:
  1. Set the Resolution (width & height in pixels) of the display
  2. Set the Bit Depth (bits per pixel) of the display
  3. Get a pointer to the Frame Buffer
- Need to interface with the GPU to accomplish this
  - Raspberry Pi uses a Mailbox interface to talk to the GPU

Video Interface

Initialize Frame Buffer via Mailbox Interface

1. Create a data structure containing initialization information
2. Wait until Mailbox can accept a message
3. Write address of init. struct to Mailbox Frame Buffer Channel
4. Wait for response from Mailbox
5. Wait for Frame Buffer pointer in init. struct to be set

Get Frame Buffer information

- Use `initFbInfo` function to get the frame buffer information.
- Argument: frame buffer information structure in r0.

```
ldr    r0, =frameBufferInfo
bl     initFbInfo
```


```
.data
frameBufferInfo:
.int   0              @ frame buffer pointer
.int   0              @ screen width
.int   0              @ screen height
```

Image Bitmap

- Check on D2L the “ImagetoASCII” Java application for converting an Image to ASCII bitmap structure.
- Save it in the `.data` section as ASCII structure.
- Create a function that loads 32-bit color values [***words***] and stores into the frame buffer.
- The ASCII bitmap structure created is a 1-D array that contains 32-bit color values in row-major order.
- Use it for picking 32-bit ***hex*** color code as well.

Challenge

- Download a 16x16 pixels image.
- Convert using “ImagetoASCII”.
- Write a function to draw your image on the screen:
  - Arguments:
    - Address of the image data.
    - X & Y coordinate to place your image on the screen.