EE-334
Digital System Design

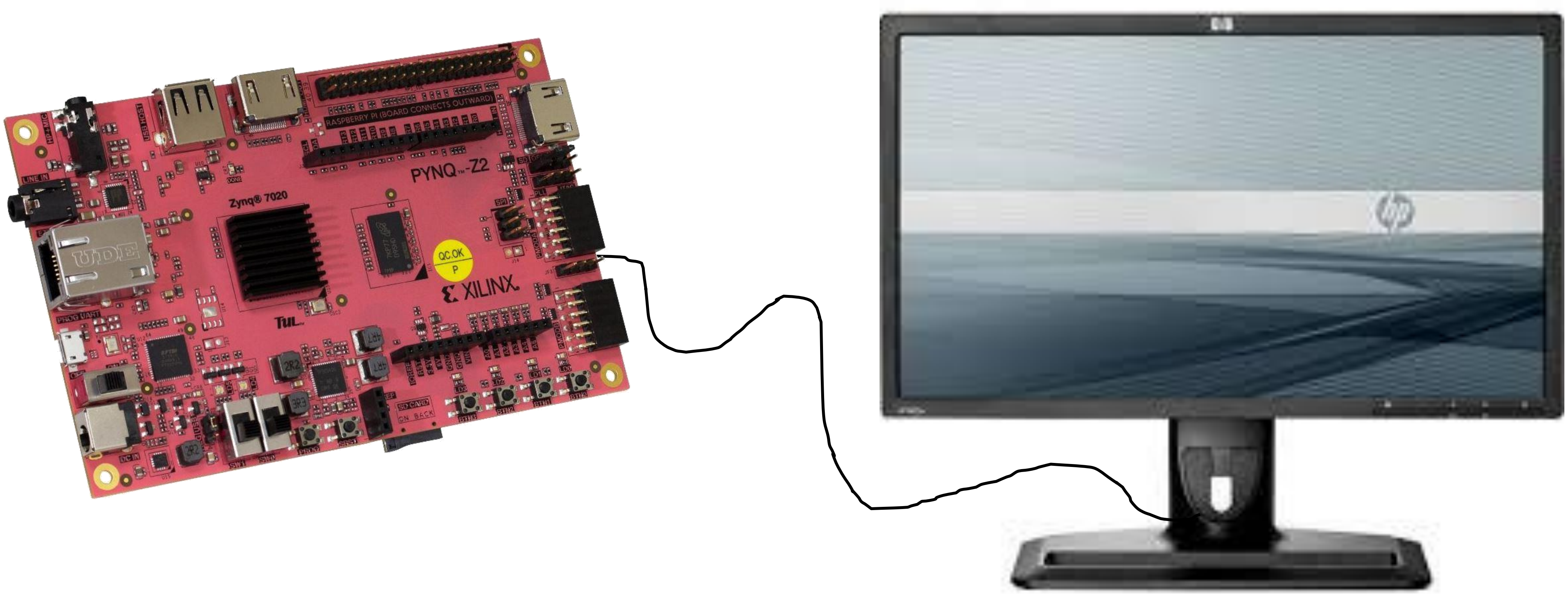
Custom Digital Circuits

Project Preparation: VGA Interface

Andreas Burg

An Initial Glimpse at the Project

- For the project, we will need a connection to a video screen



Video Resolutions and Data Rates

- **Sending video to a screen is** (and has been) a **demanding** task:
  - The need for higher resolutions (and initially also color) drives data rates up
  - **Data** transferred to screen **is uncompressed**

Resolution	Frame Size	Pixels/frame	Raw data rate
8k (Ultra HD)	7’680 x 4’320	33’177’600	> 30 Gbps
4k (Ultra HD)	3840 x 2’160	8’294’400	9-14 Gbps
1080p (Full HD)	1920 x 1000	2’073’600	4.455 Gbps
720p (HD)	1280 x 720	921’600	2.23 Gbps
480p (SD)	640 x 480	307’000	0.81 Gbps

- **Early standards were** based on **analog** signals
  - Driven by analog screens and insufficiently fast digital circuits
  - **Analog signal quality is insufficient for high resolution**

Video Interface Standards

- There are four major video standards (some at the end of their life)

VGA

Analog signalling

DVI

Digital signalling

HDMI

Serial digital signalling

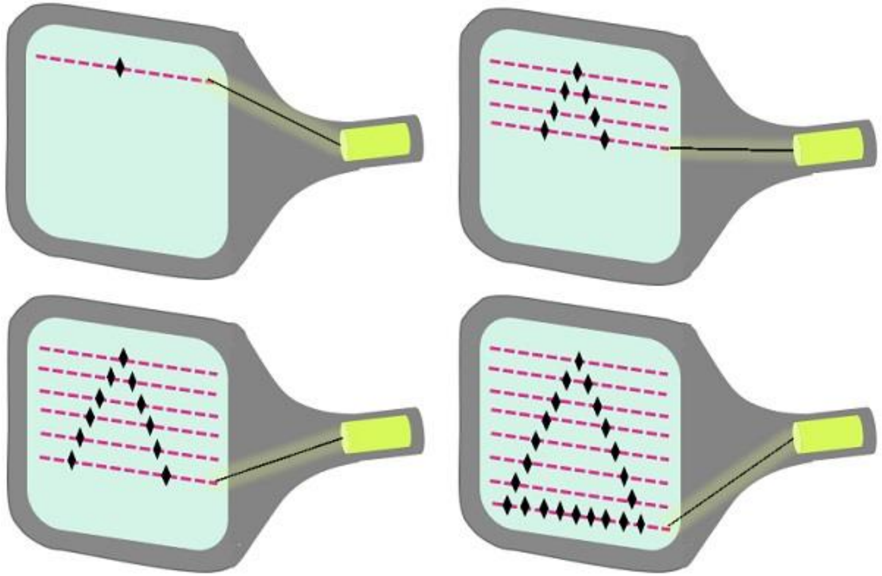
DisplyPort

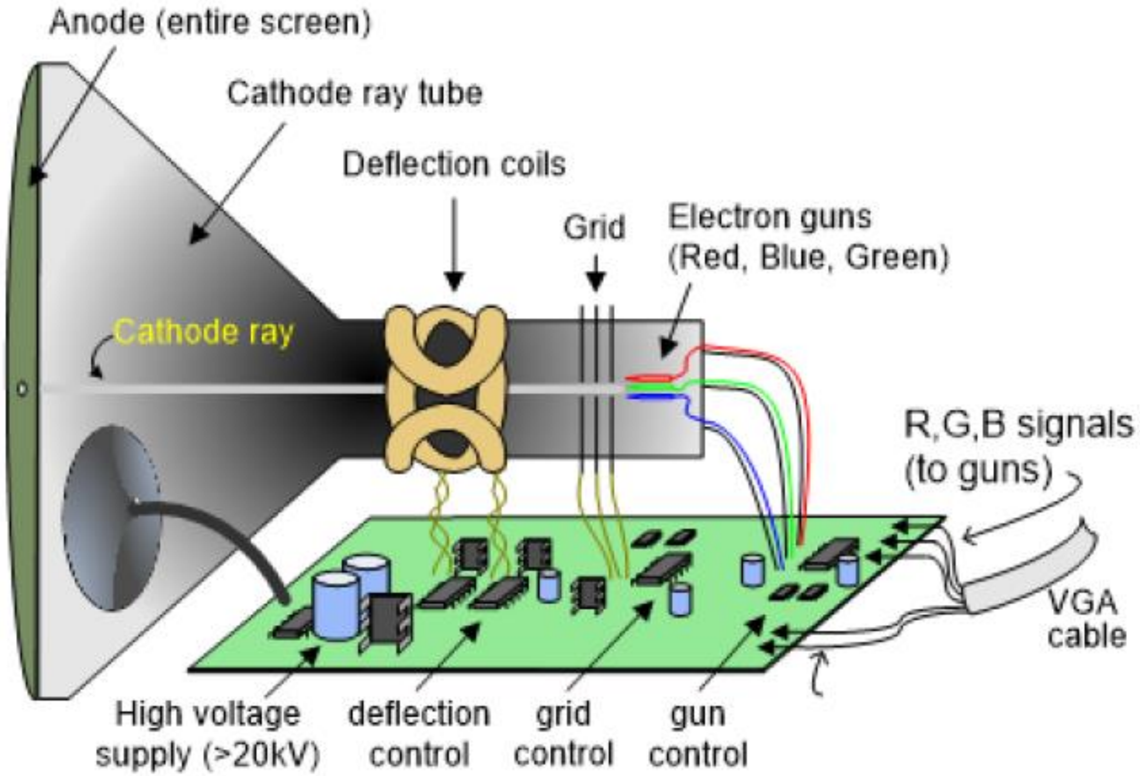
Serial digital signalling

- The technical foundations of these standards reflect the general **trend in digital systems interconnect: from analog to digital to high-speed serial interfaces**

The VGA System

- Signalling **motivated by** the needs of early days **analog cathode ray screens**
- **Image** is **drawn** rapidly **by a moving electron beam** on a fluorescent screen
  - **Beam scans the screen in lines**
  - Rapid **repetitions ensure** the **illusion of a steady image**

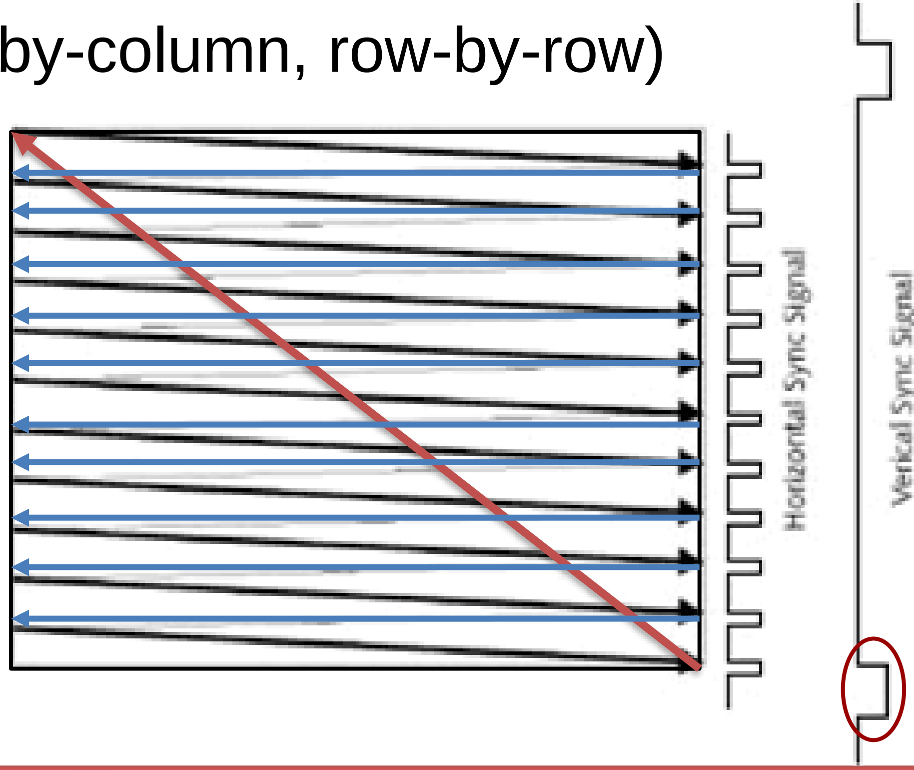




VGA Signal Control and Data Signals

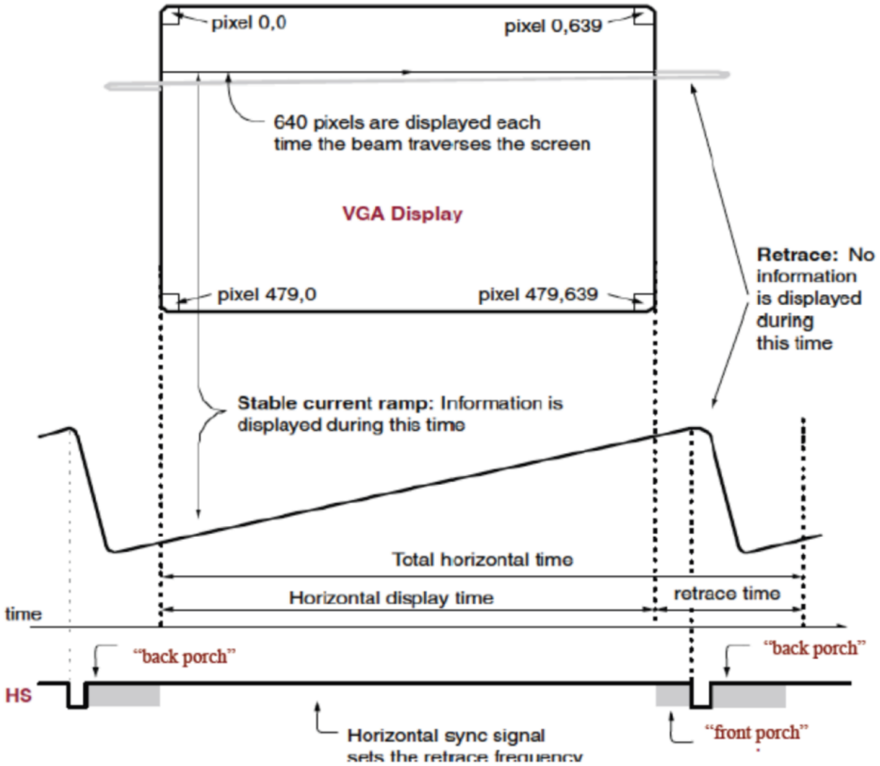
- The VGA interface contains the following signals:
  - **Colour: three analog signals with intensity information** for Red, Green, and Blue
  - **Control: two digital signals to control the electron beam** (to synchronize the colour information with the position currently drawn on the screen)

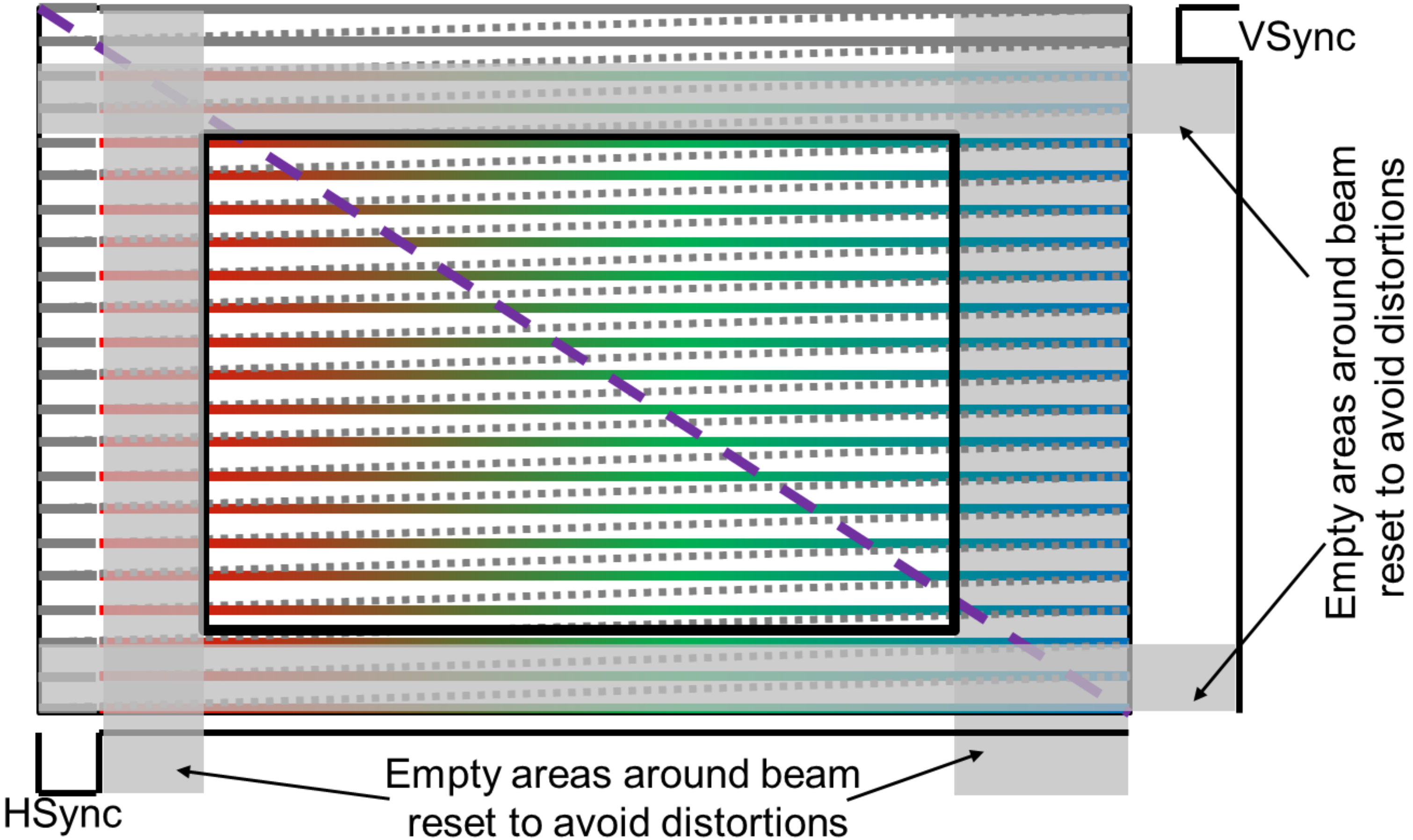
- **Raster scan**: top-left to bottom-right (column-by-column, row-by-row)
  - **Horizontal Sync (HS):** control the timing of a row (reset beam to start of a row)
  - **Vertical Sync (VS):** control the timing of a frame (reset beam to start of a frame)



Horizontal and Vertical Blank Intervals

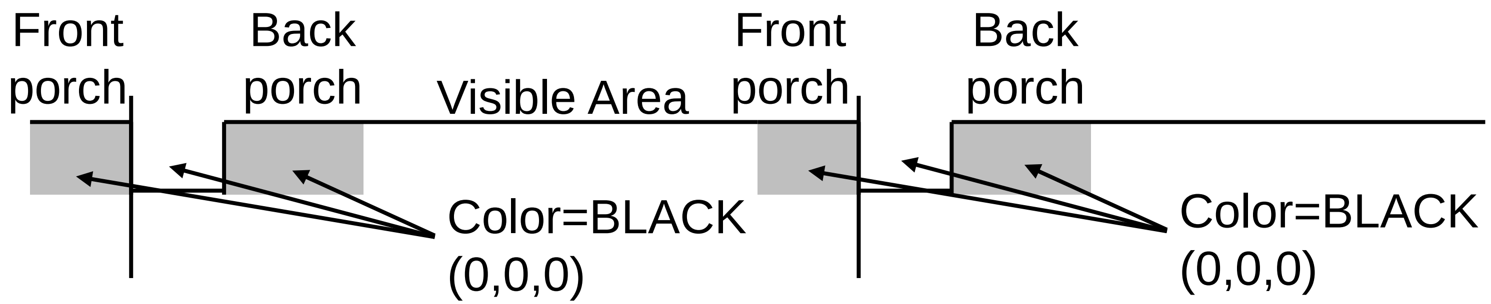
- Returning the beam in analog screens required time (return and stabilize)
  - **Empty (black, not visible) areas are added around the visible part of the screen**





VGA HS and VS Timing & Naming Convention

- **Empty areas before and after H/V-Sync pulses:**
  - Protective area = Porch
  - Front-porch and Back-porch relative to the Sync Pulse
  - Color during Front- Back-porch and Sync Pulse must be BLACK!



  - Example: VGA with 640x480

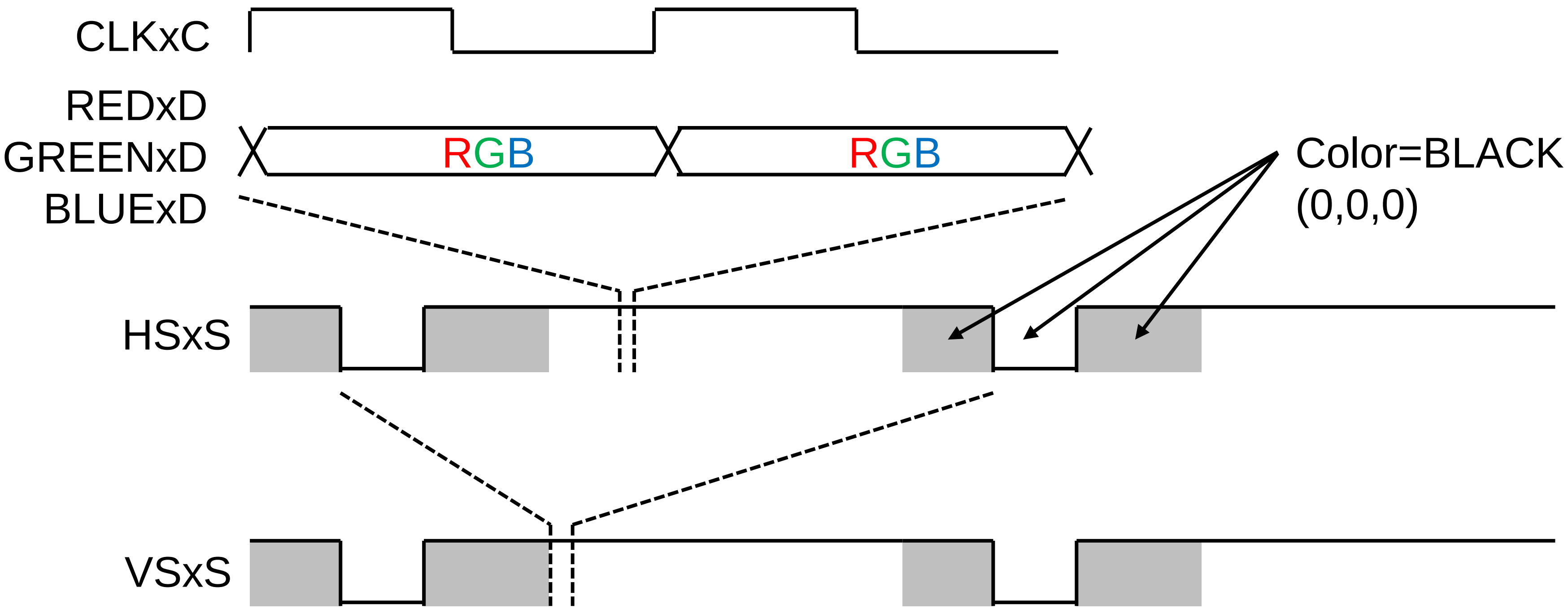
Symbol	Parameter	Vertical Sync			Horizontal Sync	
		Time	Pixels	Lines	Time	Pixels
T_S	Sync pulse time	16.7 ms	416,800	521	32 µs	800
T_{DISP}	Display time	15.36 ms	384,000	480	25.6 µs	640
T_{PW}	Pulse width	64 µs	1,600	2	3.84 µs	96
T_{FP}	Front porch	320 µs	8,000	10	640 ns	16
T_{BP}	Back porch	928 µs	23,200	29	1.92 µs	48

See: http://tinyvga.com/vga-timing

VGA HS and VS Timing

- **Timing depends on the video resolution and is specified as follows:**
  - HSync : Integer number of pixels
  - VSync : Integer number of lines
- **Pixel duration is often a multiple of a base clock (for low resolutions)**

See: http://tinyvga.com/vga-timing



VGA Controller Building Blocks

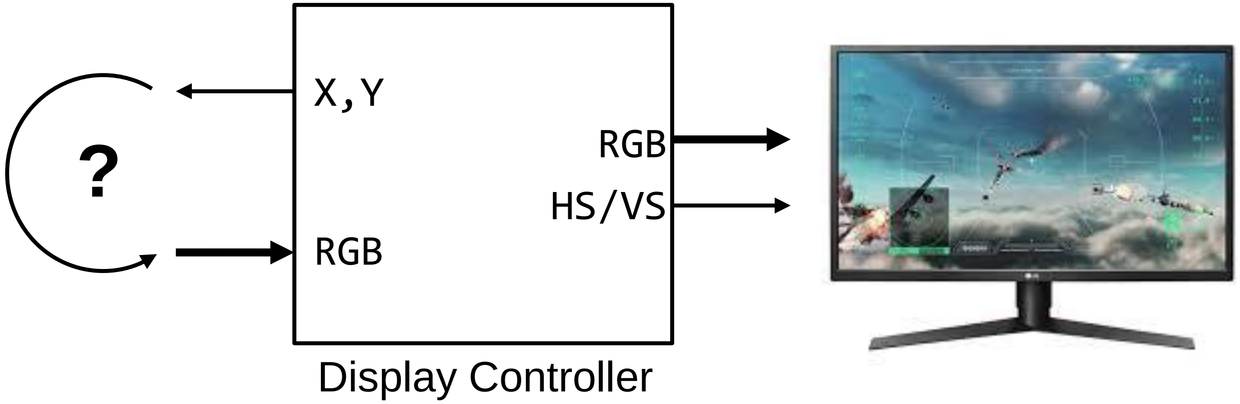
- *Registers and states:* What do you need
  - Keep track of the time *within* a frame (everything periodic with frames)
    - Pixel in each row (column)
    - Line (row) within a frame
  - Two options: FSM(s) or Counters … many states => counters
  - Hierarchical structure: interacting counters (can be updated in one combinational process)

- *Combinational Logic:*
  - Generate HS and VS signals (based on counters or FSM states)
    - Combinational logic
  - Logic to set the display color output (RGB) to black outside the visible field

Generating Graphics

- Content is sent pixel-by-pixel following a well defined repetitive raster scan pattern
  - **VGA controller generates control signals** and **provides coordinates (X,Y)** of current pixel
  - **Graphics hardware need to supply the** corresponding **colour (RGB)**

- **Considerations:**
  - High data rate for supplying pixel data (resolution dependent)
  - Some latency is acceptable when data can be pre-fetched

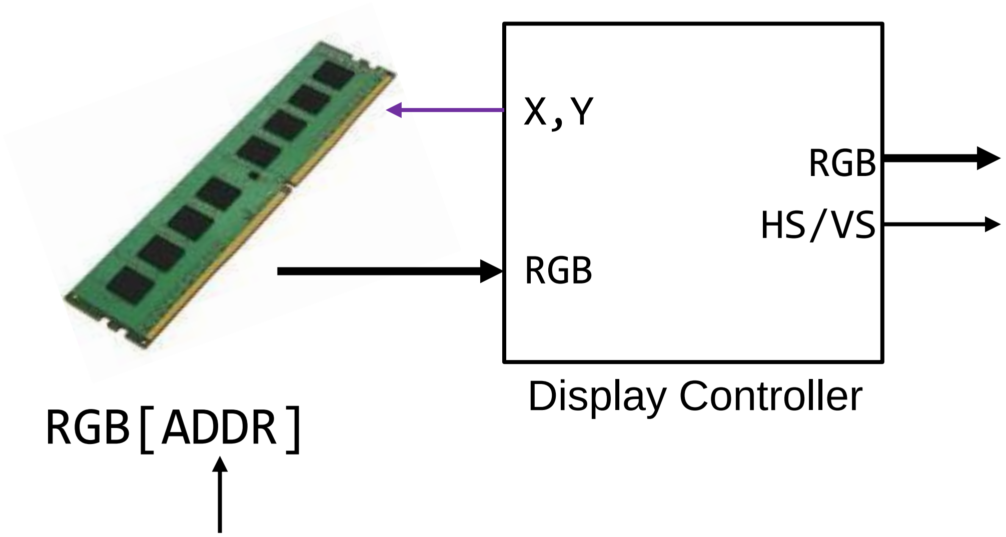


- **Two options for image generation:**
  1. **Frame buffer:** image stored in a memory
  2. **Logic:** generate the image on the fly

Generating Graphics

Frame Buffer

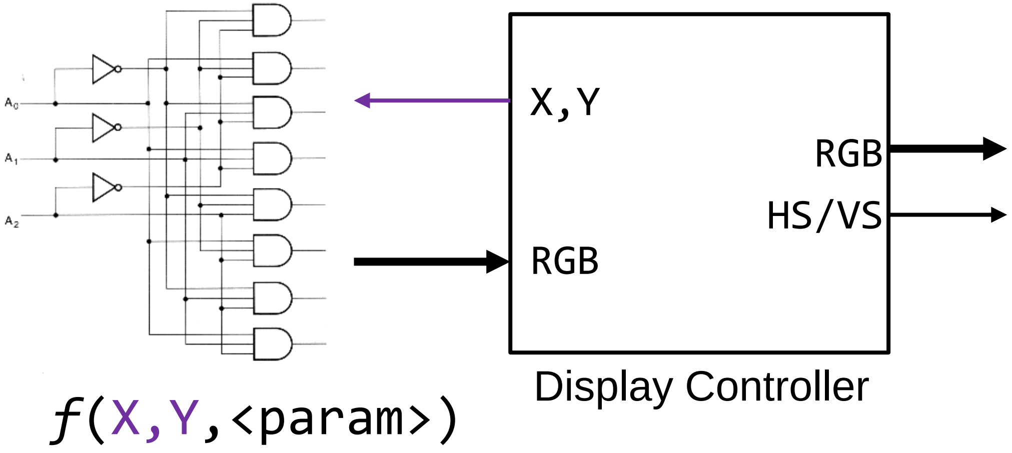
- **Image stored in a memory**



lin(X,Y): maps X,Y coordinates to a unique memory address

Online Generation

- **Generate the pixel colour on the fly**



Generating Graphics

Frame Buffer

- **Image stored in a memory**

- Advantages:
  - Arbitrary images (pre-loaded)
  - Complex scenes can be computed without much constraints on complexity
  - High pixel rate (resolution) is no problem
- Disadvantages:
  - Requires significant memory (need to store explicitly each pixel)
  - Updating the stored frame can be complex and time consuming

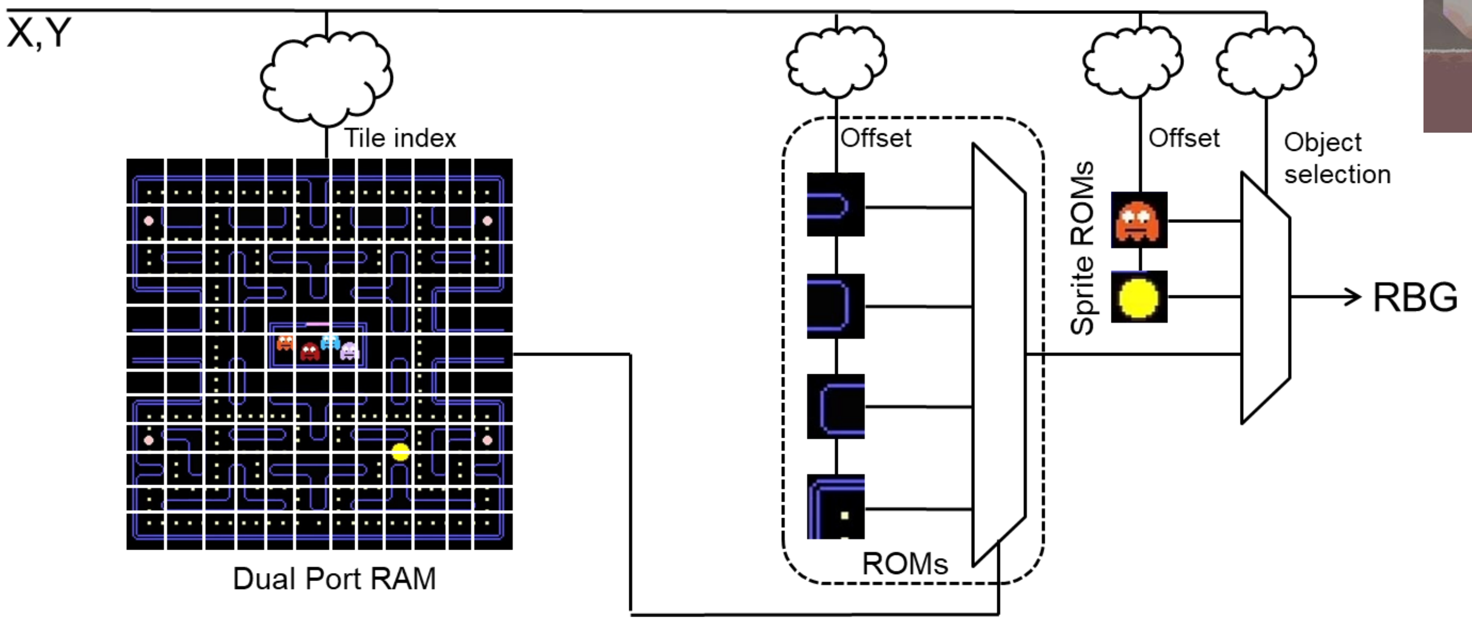
Online Generation

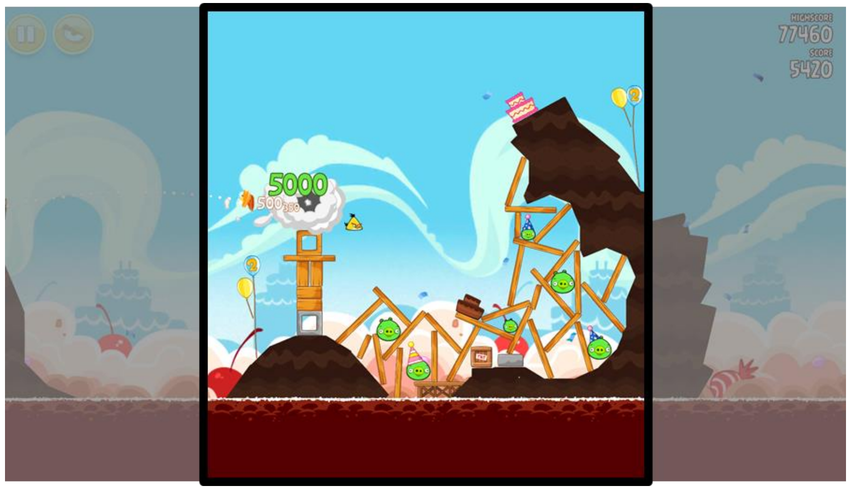
- **Generate the pixel colour on the fly**

- Advantages:
  - No memory required
  - Easy when colour can be derived from location on screen with simple logic
  - Some “scenes” require almost no logic at all
- Disadvantages:
  - High pixel rate allows only for low-complexity computations
  - Arbitrary images are not possible

Complex Scenes

- **Complex Scenes are often a mix of pre-computed graphics and logic**
  - Background: stored in a frame buffer
  - Sprites:
    - small memories or ROMs (LUTs) store small images
    - Logic controls what is displayed where on the screen

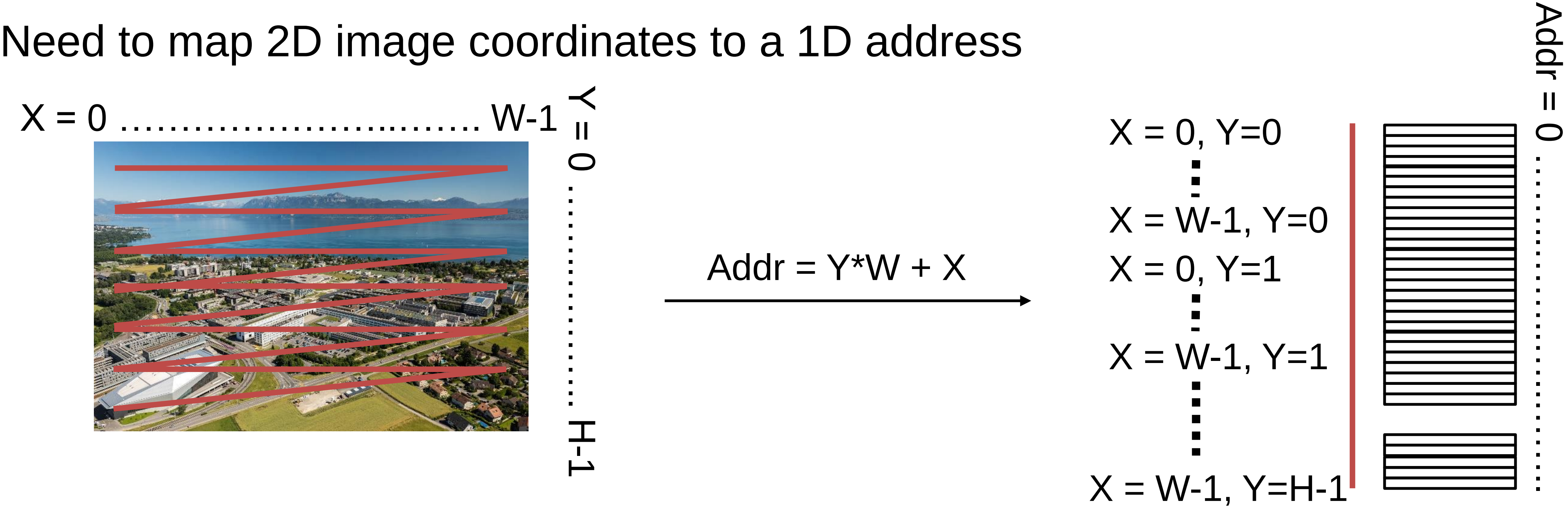




Frame Buffer Memory Map

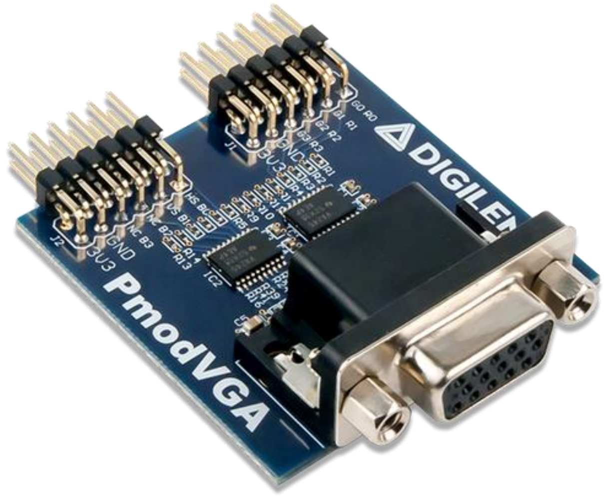
- In a frame buffer, the image on the screen is stored in a memory
  - The image is 2-dimensional (X=0,…,W-1, Y=0,…,H-1)
  - A memory is typically organized in only 1-dimension (Addr=0, …), where each address contains a data word of B bits

- Need to map 2D image coordinates to a 1D address



Generating an Analog VGA Signal

- The VGA standard carries the colour (RGB) components as analogue signals (only Hsync and Vsync are digital)
  - FPGA can only provide digital signals



- A digital-to-analog converter is required
  - We use the Digilent PmodVGA module for this purpose
  - Connection through two PMOD (GPIO) connectors of the FPGA board
  - 4+4+4 bit = 12 bit RGB colour representation
  - Simple R-2R resistor ladder based DAC up to 150 MHz pixel clock

- For more information: https://digilent.com/reference/_media/reference/pmod/pmodvga/pmodvga_rm.pdf