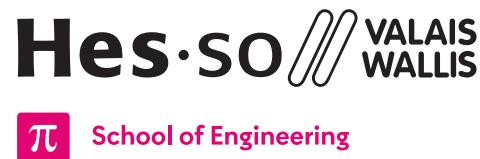
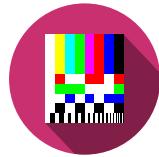


# Lecture Digital Design (DiD)

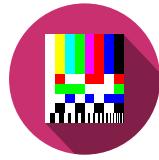


**Orientation:** Information and Communication Technology (ISC)  
**Specialisation:** Data Engineering (DE)  
**Course:** Digital Design (DiD)  
**Authors:** Silvan Zahno, Axel Amand  
**Date:** 04.12.2025  
**Version:** v2.5



# Contents

1	Introduction .....	3
2	Specification .....	4
2.1	Functions .....	4
2.2	Circuit .....	4
2.3	<b>Video Graphics Array (VGA)</b> Timing (example) .....	6
2.4	HDL-Designer Project .....	8
3	Components .....	9
3.1	<b>Field Programmable Gate Array (FPGA)</b> -Board .....	9
3.2	Buttons and <b>Light Emitting Diode (LED)</b> .....	9
3.3	<b>Peripheral Module (PMod)</b> - <b>Digital Visual Interface (DVI)</b> module .....	10
4	Evaluation .....	11
5	First steps .....	12
5.1	Procedure .....	12
5.2	Tips .....	13
	Glossary .....	14



# 1 | Introduction

The aim of the project is to apply the knowledge acquired at the end of the semester directly using a practical example. The goal is to create a controller which can via a [VGA](#) interface display a predefined image. This display system is represented in the illustration [Figure 1](#).

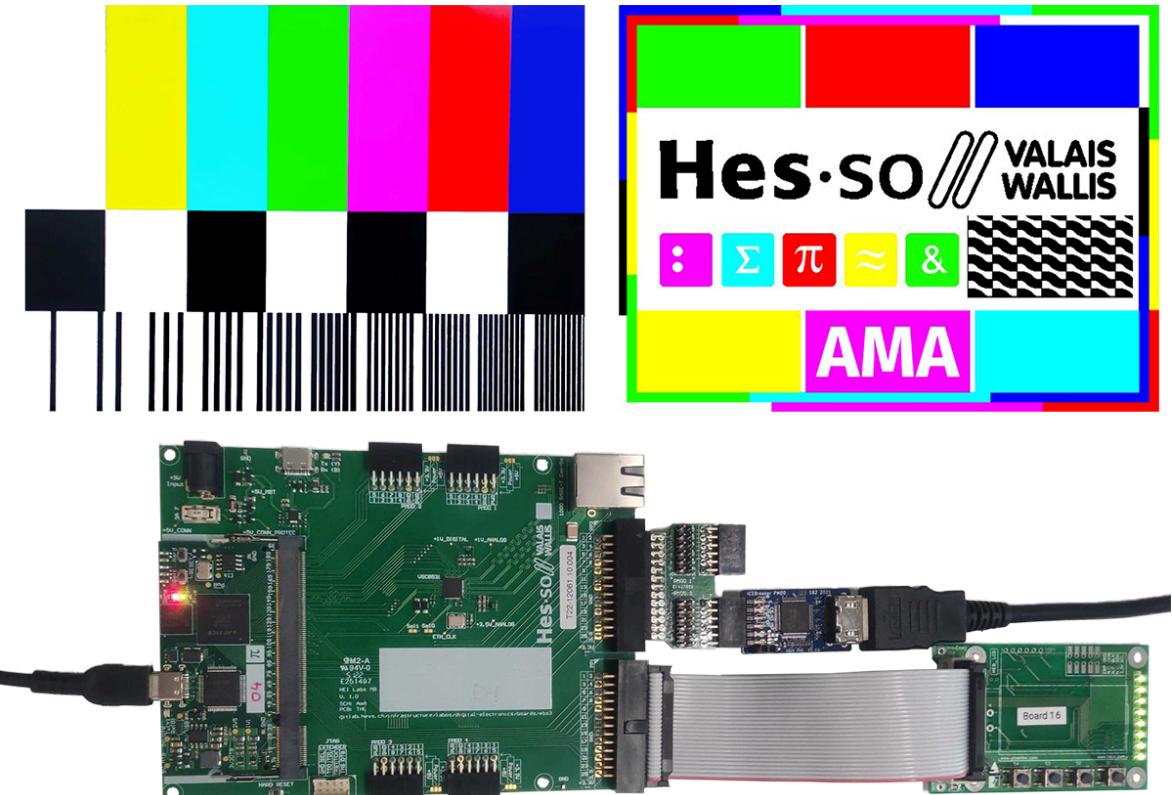
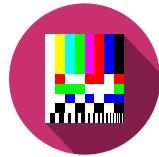


Figure 1 - Display setup (EBS3)

The task is to complete the minimal [specification 2](#), which can be optionally extended with additional functions, e.g. switching between a calculated and a pre-recorded image, animating the screen ...



Additional functions can earn some extra points.



## 2 | Specification

### 2.1 Functions

The basic functions are defined as follows:

- If the **start** button is pressed, a test image is displayed on the monitor.
- If the **stop** button is pressed, the test image is removed and the monitor turns black.
- The test image is generated by the circuit (*not pre-recorded*) and consists of all possible color combinations that can be combined with the **3bit per pixel (bpp)** DVI module. The illustration [Figure 2](#) shows a possible test image that can be displayed.
- The resolution must be 640px x 480px @ 60 Hz.



Figure 2 - Possible test image showing all color combinations

### 2.2 Circuit

The **FPGA** development board forms the core of the system. A [LED - chapter 3.2](#) board and a [DVI module - chapter 3.3](#) are connected to it. The screen is connected to the module output via **High Definition Multimedia Interface (HDMI)**. The entire system is schematically shown in the illustration [Figure 3](#).

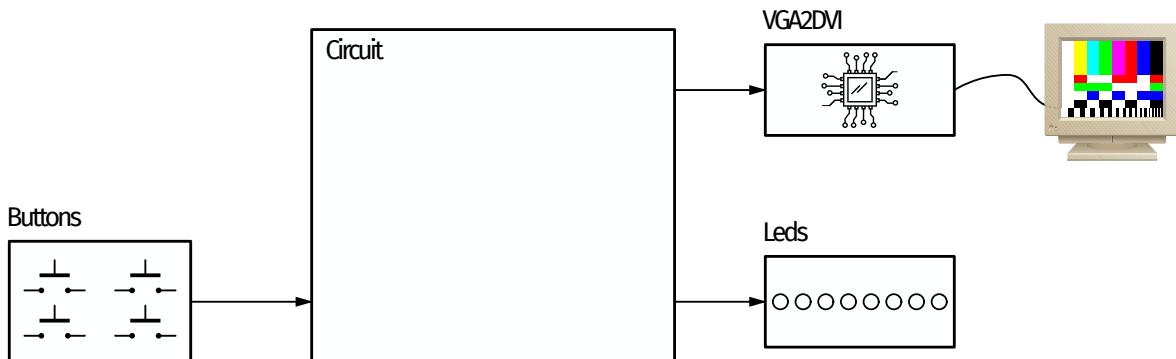
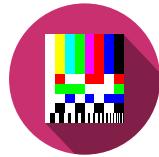


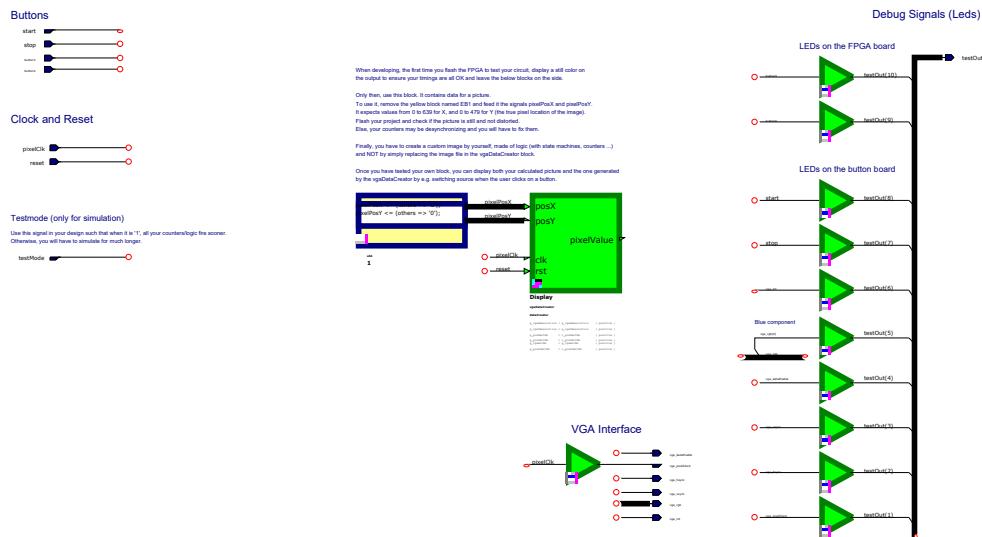
Figure 3 - Display circuit



The complete circuit works as follows:

- Four buttons are used to control the system: **start**, **stop** as well as two freely available buttons **button\_3** and **button\_4**. Those can be used for optional functions. A '1' on the signal means that the button is pressed.
- By pressing the **start** button, an image, calculated according to the pixel position (e.g. color bars), is continuously transmitted to the **PMod** module via the **VGA** interface.
- By pressing the **stop** button, a black image is transmitted.
- The **PMod** module for the previously mentioned **VGA** video interface has the following signals: **vga\_dataEnable**, **vga\_pixelClock**, **vga\_hsync**, **vga\_vsync** as well as **vga\_rgb[2:0]**.
  - The module converts **VGA** video signals to **Transition Minimized Differential Signaling (TMDS)** signals and sends them to the monitor via the **HDMI** interface.
- The **testOut** pins can be used to provide additional information about the system, for example for debugging purposes or to control **LED**.

The empty TopLevel circuit shows all signals connected to the **FPGA** board, see [Figure 4](#):



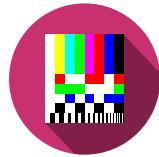
[Figure 4 - Empty Toplevel circuit](#)



The **testOut** signal allows to turn on and off the LEDs presented in the [Section 3.2](#).



The already existing **vgaDataCreator** block provides a pixel color according to the given X and Y positions (from 0 to 639 for X, 0 to 479 for Y) to display an image. It allows to roughly detect a synchronization error (i.e. the image looks distorted, wavy, cut off ...). **This block is provided for testing purposes and does not replace the primary request to display a custom image.**



## 2.3 VGA Timing (example)

Figure 6 and Figure 5 show the VGA timings for the resolution 640px x 480px @ 60Hz.

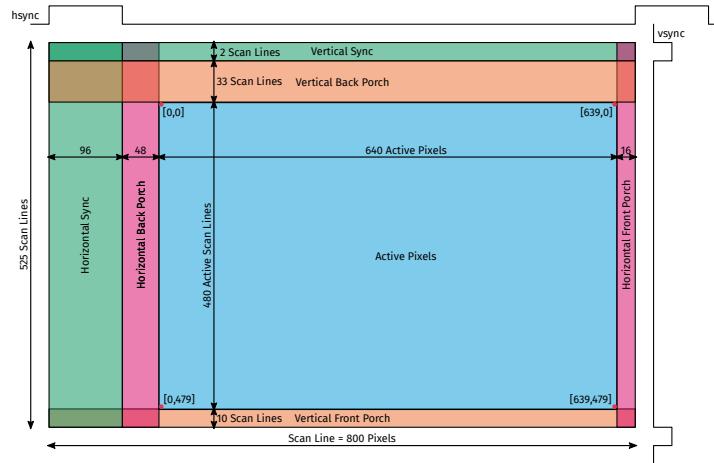


Figure 5 - VGA Video Decoding

The image is built line by line, starting from the upper left corner:

- **vga\_hsync** and **vga\_vsync** indicate if a new line (**hsync**) or a new page (**vsync**) starts and must be held for a specific time.
- The time for one pixel is one cycle of **vga\_pixelClock**, whose frequency is already set according to the 640px x 480px @ 60Hz standard.
- **vga\_dataEnable** indicates if the data signal is active and can be read.
- The **vga\_rgb** vector contains the data for a pixel, with 1 bit depth per color.
- **vga\_int** can be activated along, displaying brighter colors.



The timing conditions for the signals **vga\_hsync** and **vga\_vsync** must be perfectly understood and respected. More information can be found in the [Video Electronics Standards Association} \(VESA\) specification \[1\]](#) as well as on the [TinyVGA website \[2\]](#).

A temporal view of the **VGA** signals is shown in Figure 6:

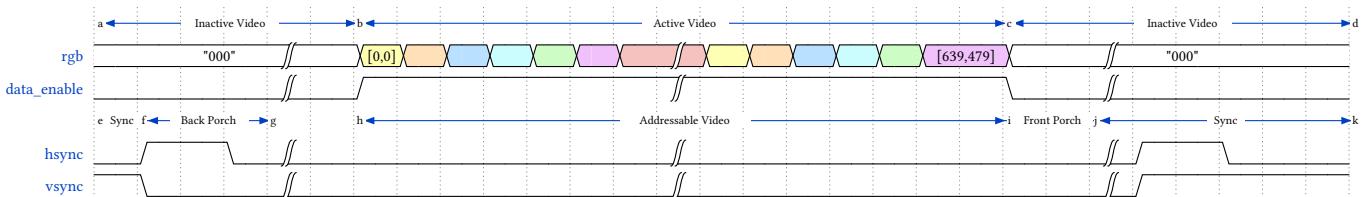
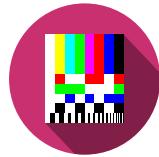


Figure 6 - Temporal sequence of the VGA signals (hsync and vsync mixed)



Outside the **Active Pixels** area, the RGB value must always be 0.



For an image of 640px x 480px, 800px x 525px are theoretically transmitted. The additional pixels are necessary for the vertical and horizontal **front porch** and **back porch**.

Those are intended to give an old CRT monitor enough time for the electron beam to reposition itself between lines and pages.

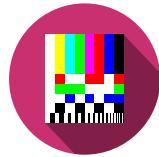
An example for the resolution 1920px x 1440px @ 60Hz is given in the listing [Table 1](#):

Name	1920x1440 @ 60Hz	
Aspect Ratio	4:3	
<hr/>		
Pixel Clock	234	MHz
Pixel Time	4.27	ns
Horizontal freq.	90	kHz
Line Time	11.11	μs
Vertical freq.	60	Hz
Frame Time	16.66	ms
<hr/>		
<b>Horizontal Timings</b>		
<hr/>		
Visible Area	1920	
Front Porch	128	
Sync Width	208	
Back Porch	344	
Total (blanks)	672	
Total (all)	2600	
Sync Polarity	neg	
<hr/>		
<b>Vertical Timings</b>		
<hr/>		
Visible Area	1440	
Front Porch	1	
Sync Width	3	
Back Porch	56	
Total (blanks)	60	
Total (all)	1500	
Sync Polarity	pos	
<hr/>		
<b>Active Pixels</b>		
<hr/>		
Visible Area	2,764,800	

Table 1 - [VGA](#) configuration for 1920px x 1440px @ 60Hz



The above figures are examples. The student must understand and adapt them to the required resolution.



## 2.4 HDL-Designer Project

A predefined HDL-Designer project can be downloaded from [Cyberlearn](#) or [Git](#). The file structure of the project is as follows:

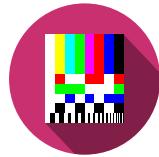
```
did_display
+-Board/           # Project and files for programming the FPGA
|   +-concat/      # Complete VHDL file including PIN-UCF file
|   +-ise/          # Xilinx ISE project
+-Display/         # Library for the components of the student solution
+-Display_test/    # Library for the simulation testbenches
+-doc/             # Folder with additional documents relevant to the project
|   +-Board/        # All schematics of the hardware boards
|   +-Components/   # All data sheets of hardware components
+-img/              # Pictures
+-Libs/             # External libraries which can be used e.g. gates, io, sequential
+-Prefs/            # HDL-Designer settings
+-Scripts/          # HDL-Designer scripts
+-Simulation/       # Modelsim simulation files
+-Tools/            # Specific tools, like a picture to BRAM translator
```



The project folder path must not contain spaces.



The project folder **doc/** contains many valuable information. Datasheets, project evaluation as well as help documents for HDL-Designer to name just a few.



# 3 | Components

The system consists of three different hardware boards, visible in the figure [Figure 1](#).

- A [FPGA](#) development board, see figure [Figure 7](#).
- A control board with 4 buttons and 8 [LED](#), see figure [Figure 8](#).
- A [PMOD to DVI](#) module for displaying the image on a screen via HDMI, see figure [Figure 9](#).

## 3.1 [FPGA](#) -Board

On the EBS3 board, the oscillator produces a clock signal ([clock](#)) with a frequency of  $f_{\text{clk}} = 100 \text{ MHz}$ . This clock is named [lcdClock](#) and is **only** intended for the dedicated LCD block of the [Buttons - LCD, Chapter 3.2](#) board (*currently unused*).

Also, this clock is reduced by PLL to  $f_{\text{clk}} = 25 \text{ MHz}$ , named [pixelClk](#). This is the one that should be used for the development of your circuit.



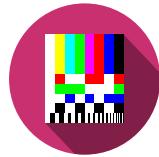
Figure 7 - EBS3 [FPGA](#) Board [3]

## 3.2 [Buttons and LED](#)

The button and [LED](#) board [4] is connected to the [FPGA](#) board. It has 4 buttons and 8 [LED](#) that can be used in the design. A second version accepts an [Liquid Crystal Display \(LCD\)](#) display [5], [6] (available on demand).



Figure 8 - Buttons-LED - LCD board [4]



### 3.3 PMod - DVI module

The PMod VGA to DVI module converts the [VGA](#) signals to [TMDS](#) signals. This allows to connect an [HDMI](#) monitor to the system.

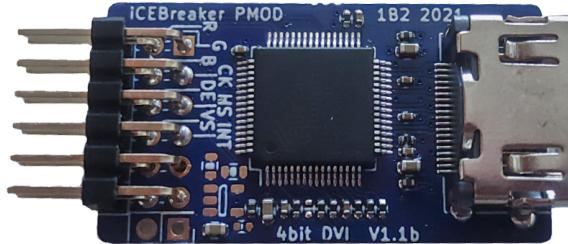


Figure 9 - PMod - DVI module

The block diagram of the Texas Instrument TFP410 chip [7] can be found in the diagram [Figure 10](#).



Study the datasheet [7] as well as the schema of the PMod module [8].

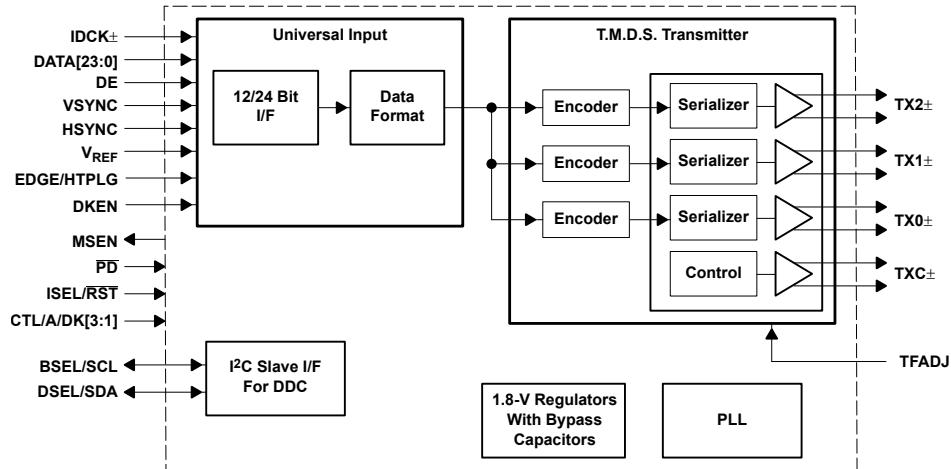
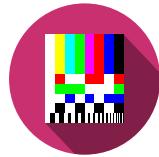


Figure 10 - Functional diagram of the PMod TI TFP410 chip [7]



## 4 | Evaluation

In the **doc/** folder, the file **evaluation-bewertung-display.pdf** shows the detailed evaluation scheme from [Table 2](#).

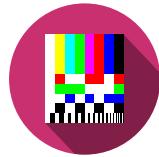
The final grade includes the report, the code as well as a presentation of the system.

<b>Evaluated aspects</b>	<b>Points</b>
<b>Report</b>	<b>55</b>
Introduction	3
Specification	5
Project	20
Verification and validation	10
Integration	9
Conclusion	3
Formal aspects of the report	5
<b>Functionality of the circuit</b>	<b>30</b>
<b>Quality of the solution</b>	<b>10</b>
<b>Presentation</b>	<b>10</b>
<b>Total</b>	<b>105</b>

Table 2 - Evaluation grid



The evaluation grid already gives indications about the structure of the report. For a good report, consult the document **How to write a project report? [9]**.



# 5 | First steps

To start the project on the right foot:

- Read the specifications and information presented carefully.
- Browse the documents in the **doc/** folder of your project.
- Analyze in detail the blocks that already exist.
- Develop a detailed functional diagram. You must be able to explain the signals and their functions.
- Implement and simulate the different blocks.
- Confirm the operation of the given hardware with the pre-installed program.
- Test the solution on the **FPGA** and find any errors 

## 5.1 Procedure

In order to minimize the number of bugs occurring at the same time, it is recommended to proceed as follows:

1. Develop a block diagram taking into account all your inputs, outputs and features you want to implement, without implementing said blocks (leave the signals at '**0**'). Read the following points before starting.
2. Implement the strict necessary to display a single color on the screen, without taking into account the buttons and/or the position of the displayed pixel. Remember, however, that the color should only be active in the **Active Pixels** area, otherwise left black ("**000**"). *Leave the **vgaDataCreator** and **eb1** blocks on the side.*  
This allows to check the timings of the signals **vga\_pixelClock**, **vga\_hsync**, **vga\_vsync** and **vga\_dataEnable**.
3. Now use the **vgaDataCreator** block to display a pre-recorded image on the screen. To do this, remove the yellow **eb1** block (*the latter holds the **pixelPosX** and **pixelPosY** signals at 0*). Feed it with the pixel position on the image at the right time, knowing that it needs a clock pulse to output the color data of the pointed pixel.
  - **pixelPosX** is an **unsigned** of **10 bits** accepting input values from 0 to 639.
  - **pixelPosY** is an **unsigned** of **9 bits** accepting input values from 0 to 479.
  - The reference image [Figure 11](#) is available under **doc/image\_in\_memory.bmp** to check if the display is correct (strange pixels due to JPEG compression and 1BPP conversion):

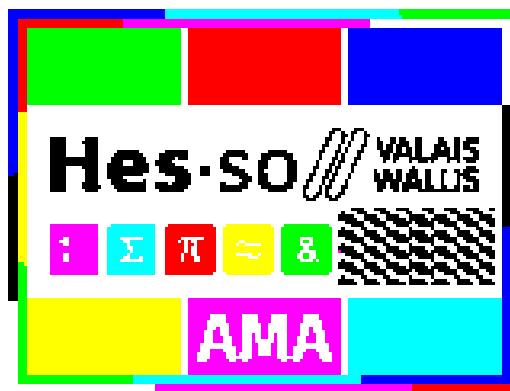
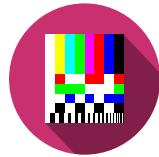


Figure 11 - Image displayed on the screen from vgaDatacreator



Displaying an image will help to detect desynchronization errors, drift, accuracy of position counters ... depending on whether the latter is distorted, wavy, cut, in movement ...

4. Then implement a block that calculates an image on the fly according to the pixel position on the screen. The image should contain all possible color combinations that can be displayed, see [Figure 2](#).
5. Finally, add the buttons and associated features. At a minimum, it should be possible to stop the display (black image) and restart it (test image).

At any time, it is possible to display certain signals on the [LED](#) to facilitate debugging. To do this, use the [testOut](#) signal linked to the [LED](#) module. It is possible to measure these with oscilloscopes or logic analyzers to help with error detection.

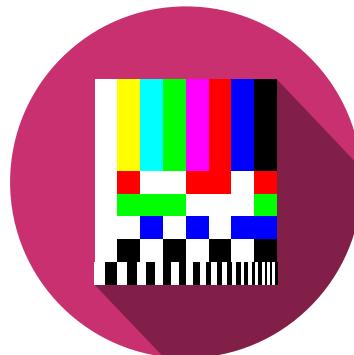
## 5.2 Tips

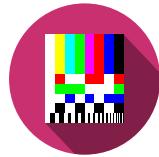
Here are some additional tips to avoid problems and time loss:

- Divide the problem into different blocks: use the empty Toplevel document. It is recommended to have a balanced mix between the number of components and the size/complexity of the components.
- Analyze the different input and output signals, their types, their sizes ... It is advisable to use the data sheets.
- Respect the DiD chapter “Methodology for the development of digital circuits (MET)” when creating the system. [\[10\]](#).
- Respect the proposed incremental procedure. Abuse tests as soon as possible.
- Save and document your intermediate steps. Architectures that have not worked, basic codes to test the architecture ... are all material that can be added to the report.



Don't forget to have fun.





# Glossary

***bpp*** – bit per pixel [4](#)

***DVI*** – Digital Visual Interface [2](#), [4](#), [9](#), [10](#)

***FPGA*** – Field Programmable Gate Array [2](#), [4](#), [5](#), [9](#), [12](#)

***HDMI*** – High Definition Multimedia Interface [4](#), [5](#), [10](#)

***LCD*** – Liquid Crystal Display [9](#)

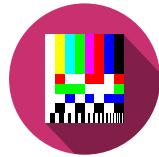
***LED*** – Light Emitting Diode [2](#), [5](#), [9](#), [13](#)

***PMod*** – Peripheral Module [2](#), [5](#), [9](#), [10](#)

***TMDS*** – Transition Minimized Differential Signaling [5](#), [10](#)

***VESA*** – Video Electronics Standards Association} [6](#)

***VGA*** – Video Graphics Array [2](#), [3](#), [5](#), [6](#), [7](#), [10](#)



# Bibliography

- [1] VESA, “VESA and Industry Standards and Guidelines for Computer Display Monitor Timing (DMT).” [Online]. Available: <https://glenwing.github.io/docs/VESA-DMT-1.13.pdf>
- [2] TinyVGA, “VGA Signal Timing.” Accessed: July 07, 2022. [Online]. Available: <http://www.tinyvga.com/vga-timing>
- [3] A. Amand and S. Zahno, “FPGA-EBS3 Electornic Technical Documentation.” 2022.
- [4] Silvan Zahno, “Schematic: Parallelport HEB LCD V2.” 2014.
- [5] Sitronix, “Datasheet Sitronix ST7565R 65x1232 Dot Matrix LCD Controller/Driver.” 2006.
- [6] Electronic Assembly, “Datasheet: DOGM Graphics Series 132x32 Dots.” 2005.
- [7] T. Instrument, “Datasheet Digital Transmitter Texas Insturment TFP410.” 2014.
- [8] E. Piotr, “Schematic: iCEBreaker PMOD 4bit DVI.” 2018.
- [9] Christophe Bianchi, François Corthay, and Silvan Zahno, “Comment Rédiger Un Rapport de Projet?” 2021.
- [10] François Corthay, Silvan Zahno, and Christophe Bianchi, “Méthodologie de Conception de Circuits Numériques.” 2021.