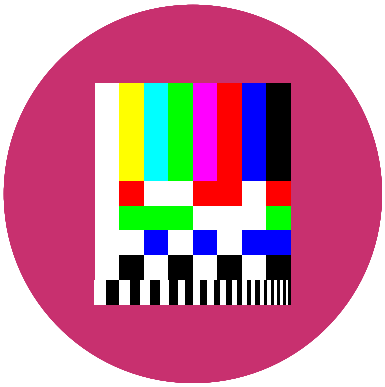
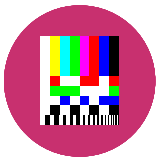


Course Digital Design (CNum)

Hes·so VALAIS WALLIS

Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

Orientation: Systèmes industriels (Synd)
Spécialisation: Infotronics (IT)
Cours: Digital Design (CNum)
Auteurs: Silvan Zahno, Axel Amand
Date: 11.03.2025
Version: v2.4



Contenu

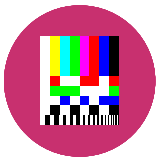
1 Introduction

L'objectif du projet est d'appliquer directement les connaissances acquises à la fin du semestre à l'aide d'un exemple pratique. Il s'agit de piloter un écran via une interface VGA afin d'afficher une image prédéfinie. Ce système d'affichage est représenté dans l'illustration Figure 1.

Figure 1 - Équipement du display (EBS3)

Le but est de réaliser les Spécifications minimales - Section 2. Les étudiants peuvent, en option, ajouter des fonctions supplémentaires, par exemple basculer entre une image calculée et une image pré-enregistrée, animer l'écran …

> Les fonctions supplémentaires permettent d'obtenir des points supplémentaires.



2 Spécifications

2.1 Fonctions

Les fonctions de base sont définies comme suit :

- Si la touche **`start`** est appuyée, une image test s'affiche sur le moniteur.
- Si la touche **`stop`** est appuyée, l'image test est supprimée et le moniteur devient noir.
- L'image test est générée par le circuit *(non pas pré-enregistrée)* et composée de toutes les combinaisons de couleurs possibles qui peuvent être combinées avec le 3**bit** **p**er **p**ixel (bpp) DVI module. L'illustration Figure 2 montre une image test possible qui peut être affichée.
- La résolution utilisée est de 640px x 480px @ 60 Hz.

Figure 2 - Mire de test montrant toutes les combinaisons de couleurs

2.2 Circuit

La carte de développement FPGA constitue le cœur du système. On y connecte une carte LED - chapitre 3.2 ainsi qu'un module DVI - chapitre 3.3. L'écran est connecté à la sortie du module par **H**igh **D**efinition **M**ultimedia **I**nterface (HDMI). L'ensemble du système est schématisé dans l'illustration Figure 3.

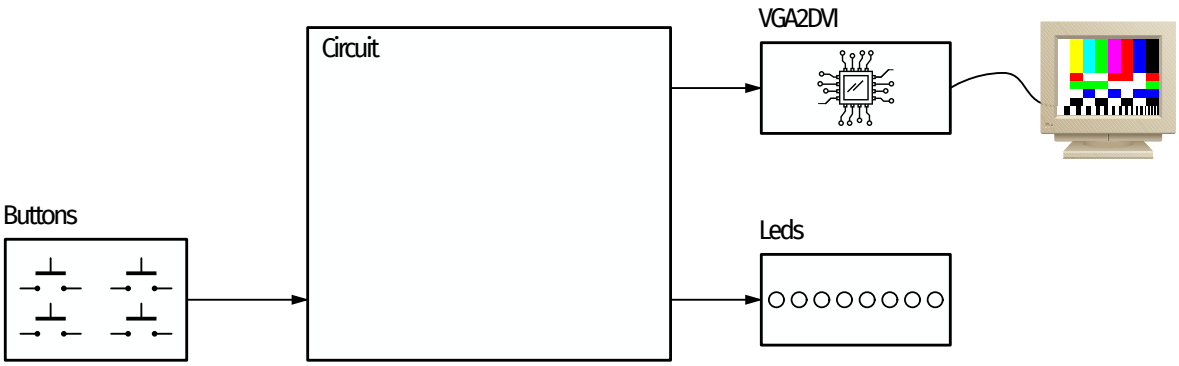
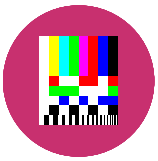


Figure 3 - Circuit d'affichage



Le circuit complet fonctionne comme suit :

- Quatre boutons sont utilisés pour contrôler le système : **`start`**, **`stop`** ainsi que deux boutons librement disponibles **`button_3`** et **`button_4`**. Ceux-ci peuvent être utilisés pour des fonctions optionnelles.
- En appuyant sur le bouton **`start`**, une image, calculée selon la position des pixels (ex. barres de couleur), est transmise en continu au module PMod via l'interface VGA.
- En appuyant sur le bouton **`stop`**, une image noire est transmise.
- Le module PMod possède des broches de configuration ainsi que l'interface vidéo VGA mentionnée précédemment pour recevoir les données d'image. Les signaux suivants sont utilisés à cet effet : **`vga_dataEnable`**, **`vga_pixelClock`**, **`vga_hsync`**, **`vga_vsync`** ainsi que **`vga_rgb[2:0]`**.
- Le module convertit les signaux vidéo VGA en **T**ransition **M**inimized **D**ifferential **S**ignaling (TMDS) et les envoie au moniteur via l'interface HDMI.
- Les broches **`testOut`** peuvent être utilisées pour fournir des informations supplémentaires sur le système, par exemple pour le débogage ou pour contrôler les LED.

Le circuit TopLevel vide montre tous les signaux connectés à la platine FPGA, voir Figure 4:

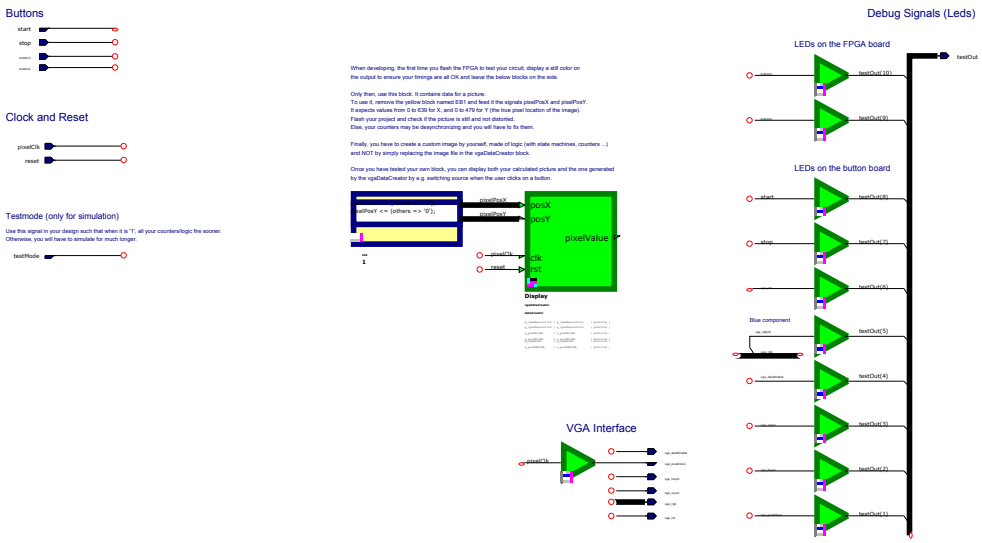
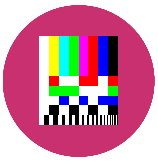


Figure 4 - Circuit Toplevel vide

Le signal **testOut** permet d'allumer et éteindre les leds présentes sur la plaque Chapitre 3.2.

Le bloc **vgaDataCreator** déjà présent permet de fournir une couleur de pixel selon les positions X et Y données (de 0 à 639 pour X, 0 à 479 pour Y) afin d'afficher une image. Il permet de détecter grossièrement une erreur de synchronisation (c.-à-d. l'image semble déformée, ondulée, coupée ...). **Ce bloc est fourni à des fins de test et ne remplace pas la demande première qui est d'afficher une image personnalisée.**



2.3 VGA Timing (exemple)

Les figures Figure 5 et Figure 6 montrent le timing VGA pour la résolution 640px x 480px @ 60Hz.

L'image est construite ligne par ligne, en commençant par le coin supérieur gauche. Les signaux **`vga_hsync`** et **`vga_vsync`** indiquent si une nouvelle ligne (**`hsync`**) ou une nouvelle page (**`vsync`**) commence. **`vga_dataEnable`** indique si le signal de données est actif (c.-à-d. dans la zone **`Active Pixels`**). Le signal **`vga_rgb`** contient les données d'un pixel, transmis à chaque cycle de **`vga_pixelClock`** lorsque nécessaire. **`vga_int`** peut être activé avec les signaux RGB, permettant d'afficher des couleurs plus claires.

Les conditions temporelles pour les signaux **`vga_hsync`** et **`vga_vsync`** doivent être parfaitement comprises et respectées. Plus d'informations peuvent être trouvées dans la spécification **V**ideo **E**lectronics **S**tandards **A**ssociation} (VESA) [1] ainsi que sur le site web de TinyVGA [2]

.

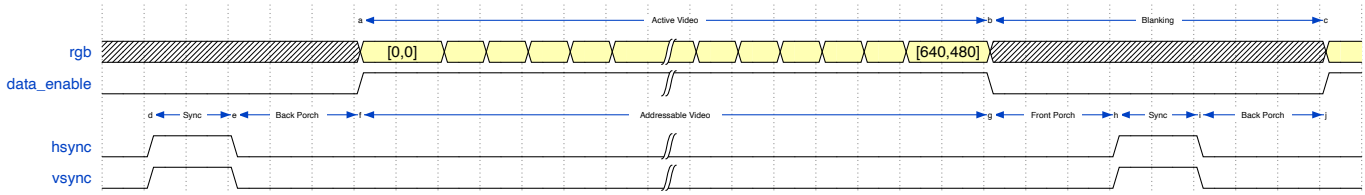


Figure 5 - Séquence temporelle des signaux VGA (**`hsync`** et **`vsync`** mélangés)

Hors de la zone **Active Pixels**, la valeur du RGB doit en tout temps être 0.

Pour une image de 640px x 480px, 800px x 525px sont théoriquement transmis, les pixels supplémentaires étant nécessaires pour le **`front porch`** et le **`back porch`** verticaux et horizontaux. Ceux-ci sont destinés à donner à un vieux moniteur CRT suffisamment de temps pour que le faisceau d'électrons puisse se repositionner entre les lignes et les pages.

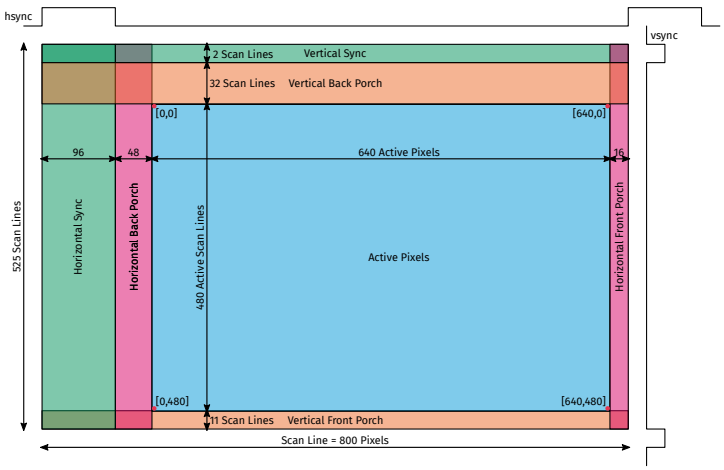
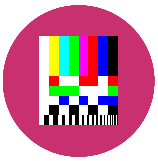


Figure 6 - Codage VGA de l'affichage



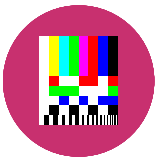
Les timings des signaux **`vga_hsync`** et **`vga_vsync`** sont prédéfinis avec précision.

Un exemple pour la résolution 1920px x 1440px {@} 60Hz est donné dans le listing Table 1:

```
Name
1920x1440 @ 60Hz
Aspect Ratio         4:3

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


Horizontal Timings

Visible Area         1920
Front Porch           128
Sync Width            208
Back Porch            344
Total (blanks)        672
Total (all)          2600
Sync Polarity         neg


Vertical Timings

Visible Area         1440
Front Porch             1
Sync Width              3
Back Porch             56
Total (blanks)         60
Total (all)          1500
Sync Polarity         pos


Active Pixels
Visible Area   2,764,800
```

Table 1 - VGA configuration pour 1920px x 1440px @ 60Hz

Les figures ci-dessus sont des exemples. L'étudiant se doit de les comprendre et les adapter à la résolution demandée.



2.4 Projet HDL-Designer

Un projet HDL-Designer prédéfini peut être téléchargé par Cyberlearn ou Git. La structure de fichier du projet se présente comme suit:

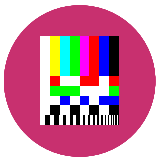
```
did_display
+--Board/           # Project and files for programming the FPGA
|   +--concat/      # Complete VHDL file including PIN-UCF file
|   +--ise/         # Xilinx ISE project
+--Display/         # Library for the components of the student solution
+--Display_test/    # Library for the simulation 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
```

Le chemin d'accès au dossier du projet ne doit pas contenir d'espaces

.

Le dossier de projet **`doc/`** contient de nombreauses informations précieuses : fiches techniques, évaluation de projet et documents d'aide pour HDL-Designer, pour n'en citer que quelques-uns

.



3 Composants

Le système se compose de 3 platines différentes, visibles dans la figure Figure 1.

- Une carte de développement FPGA, voir figure Figure 7.
- Une carte de contrôle à 4 boutons et 8 LED, voir figure Figure 8.
- Un module PMod vers DVI pour l'affichage de l'image sur un écran par HDMI, voir figure Figure 9.

3.1 Carte FPGA

Sur la carte EBS3, l'oscillateur utilisé produit un signal d'horloge (**`clock`**) avec une fréquence de $f_{\text{clk}} = 100$ MHz. Cette horloge est nommée **`lcdClock`** et n'est prévue **que** pour le bloc LCD dédié de la platine Boutons - LCD, Chapitre 3.2 *(actuellement inutilisé)*.

Aussi, cette horloge est réduite par PLL à $f_{\text{clk}} = 25$ MHz, nommée **`pixelClk`**. C'est cette dernière qui doit être utilisée pour le développement de votre circuit.

Figure 7 - Carte électronique FPGA EBS3 [3]

3.2 Boutons et LED

La platine boutons et les LED [4] est connectée à la platine FPGA. Elle possède 4 boutons et 8 LED qui peuvent être utilisés dans le design, ainsi qu'un affichage **L**iquid **C**rystal **D**isplay (LCD) [5], [6].

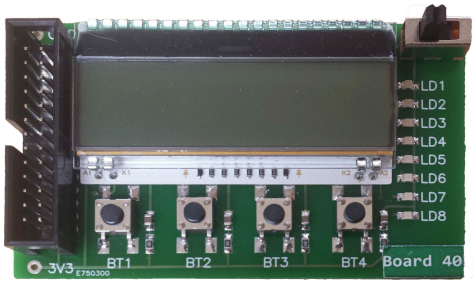


Figure 8 - Carte électronique boutons-LED - LCD [4]

3.3 Module PMod - DVI

Le module PMod VGA vers DVI convertit les signaux VGA en signaux TMDS. Cela permet de connecter un moniteur HDMI au système.

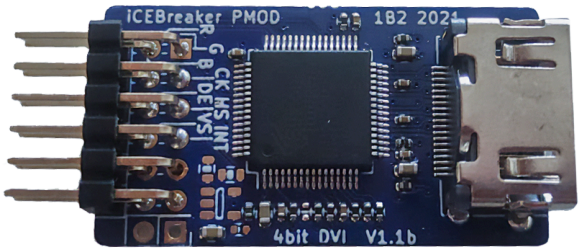
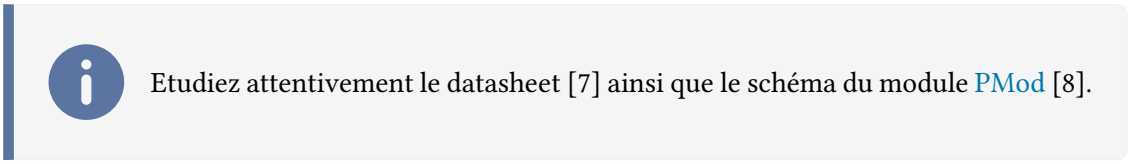


Figure 9 - Module PMod - DVI

Le schéma fonctionnel de la puce Texas Instrument TFP410 [7] peut être consulté dans le diagramme Figure 10.



Etudiez attentivement le datasheet [7] ainsi que le schéma du module PMod [8].

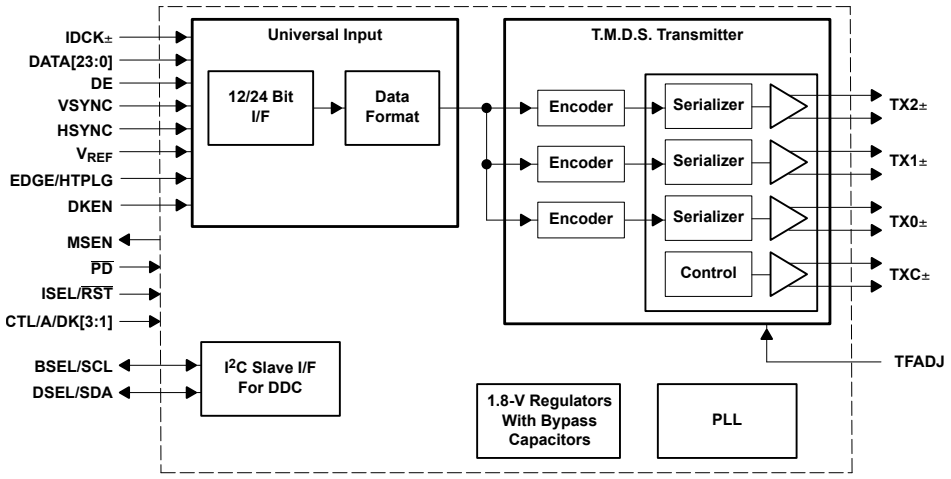
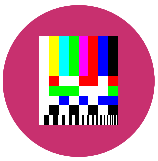


Figure 10 - Schéma fonctionnel de la puce du PMod TI TFP410 [7]



4 Evaluation

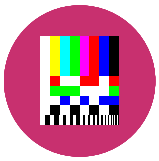
Dans le dossier **doc/**, le fichier **evaluation-bewertung-display.pdf** montre le schéma d'évaluation détaillé du Table 2.

La note finale contient le rapport, le code ainsi qu'une présentation du système.

Aspects évalués	Points
Rapport	**55**
Introduction	3
Spécification	5
Projet	20
Vérification et validation	10
Intégration	9
Conclusion	3
Aspects formels du rapport	5
Fonctionnalité du circuit	**30**
Qualité de la solution	**10**
Présentation	**10**
Total	**105**

Table 2 - Grille d'évaluation

La grille d'évaluation donne des indications sur la structure du rapport. Pour un bon rapport, consultez le document **Comment rédiger un rapport de projet** [9].



5 Premières étapes

Pour bien démarrer le projet :

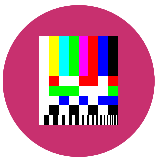
- Lisez attentivement les spécifications et les informations présentées.
- Parcourez les documents dans le dossier **`doc/`** de votre projet.
- Analysez en détail les blocs qui existent déjà.
- Développez un schéma fonctionnel détaillé. Vous devez pouvoir expliquer les signaux et leurs fonctions.
- Implémentez et simulez les différents blocs.
- Confirmez le fonctionnement du matériel donné grâce au programme préinstallé.
- Testez la solution sur la FPGA et trouvez les éventuelles erreurs 🕷

5.1 Marche à suivre

Afin de minimiser le nombre de bugs se présentant au même instant, il est recommandé de procéder comme suit:

1. Développez un schéma de principe prenant en compte toutes vos entrées, sorties et fonctionnalités que vous souhaitez implémenter, sans pour autant implémenter lesdits blocs (laissez les signaux à **`'0'`**). Lisez les points suivants avant de démarrer.
2. Implémentez le strict nécessaire afin d'afficher une couleur unique sur l'écran, sans pour le moment prendre en compte les boutons et/ou la position du pixel affiché.
   Rappelez vous toutefois que la couleur ne devrait être active que dans la zone **Active Pixels**, sinon laissée noire (**`"000"`**).
   *Laissez les blocs **vgaDataCreator** et **eb1** sur le côté.*
   - Cela permet de vérifier les timings des signaux **vga_pixelClock**, **vga_hsync**, **vga_vsync** et **vga_dataEnable**.
3. Utilisez maintenant le bloc **vgaDataCreator** afin d'afficher une image pré-enregistrée sur l'écran. Pour ce faire, supprimez le bloc jaune **eb1** *(ce dernier tient les signaux **pixelPosX** et **pixelPosY** à 0)*. Fournissez-lui au bon moment la position du pixel sur l'image, sachant qu'il a besoin d'un coup de clock pour sortir la donnée de couleur du pixel pointé.
   - **pixelPosX** est un **unsigned** de **10 bits** acceptant en entrée des valeurs de 0 à 639.
   - **pixelPosY** est un **unsigned** de **9 bits** acceptant en entrée des valeurs de 0 à 479.
   - Afficher une image permettra de détecter des erreurs de désynchronisation, de drift, d'exactitude des compteurs de positions … selon si cette dernière est déformée, ondulée, coupée, mobile …
   - L'image de référence Figure 11 est disponible sous **`doc/image_in_memory.bmp`** afin de vérifier si l'affichage est correct (pixels étranges dûs à la compression JPEG et à la conversion 1BPP):

Figure 11 - Image affichée sur l'écran à partir de vgaDatacreator



4. Ensuite, implémentez vous-même un bloc permettant de calculer une image calculée à la volée selon la position du pixel sur l'écran. L'image doit contenir toutes les combinaisons de couleur possibles pouvant être affichées, voir Figure 2.
5. Finalement, ajoutez les boutons et les fonctionnalités associées. Au minimum, il doit être possible d'arrêter l'affichage (image noire) et de le relancer (image de test).

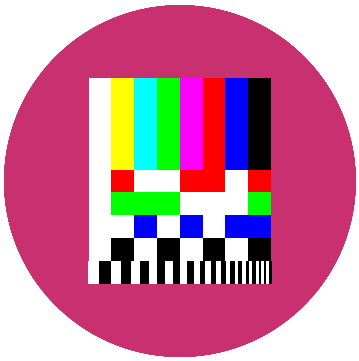
A tout moment, il est possible d'afficher certains signaux sur les LED pour faciliter le débogage. Pour ce faire, utilisez le signal **`testOut`** lié au module LED. Il est possible de mesurer ces dernières grâce à des oscilloscopes ou des analyseurs logiques afin d'aider à la détection des erreurs.

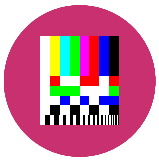
5.2 Tips

Ci-joint quelques conseils supplémentaires pour éviter les problèmes et les pertes de temps:

- Divisez le problème en différents blocs : utilisez pour cela le document Toplevel vide. Il est recommandé d'avoir un mélange équilibré entre le nombre de composants et la taille/complexité de ces derniers.
- Analysez les différents signaux d'entrée et de sortie, leurs types, leurs tailles … Il est conseillé d'utiliser en partie les fiches techniques.
- Respectez le chapitre DiD « Méthodologie de conception de cicuits numériques (MET) » lors de la création du système. [10].
- Respectez la marche à suivre incrémentale proposée. Abusez des tests dès que possible.
- Sauvegardez et documentez vos étapes intermédiaires. Les architectures n'ayant pas fonctionnées, les codes basiques pour tester l'architecture … sont autant de matière à ajouter au rapport.

N'oubliez pas de vous amuser.





Glossaire

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

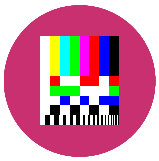
LED **– Light Emitting Diode** 2, 5, 9, 13

PMod **– Peripheral Module** 2, 5, 9, 10

TMDS **– Transition Minimized Differential Signaling** 5, 10

VESA **– Video Electronics Standards Asociation}** 6

VGA **– Video Graphics Array** 2, 3, 5, 6, 7, 10



Bibliographie

[1] VESA, « VESA and Industry Standards and Guidelines for Computer Display Monitor Timing (DMT) ». [En ligne]. Disponible sur: https://glenwing.github.io/docs/VESA-DMT-1.13.pdf

[2] TinyVGA, « VGA Signal Timing ». Consulté le: 7 juillet 2022. [En ligne]. Disponible sur: http://www.tinyvga.com/vga-timing

[3] A. Amand et 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, et Silvan Zahno, « Comment Rédiger Un Rapport de Projet? ». 2021.

[10] François Corthay, Silvan Zahno, et Christophe Bianchi, « Méthodologie de Conception de Cicuits Numériques ». 2021.