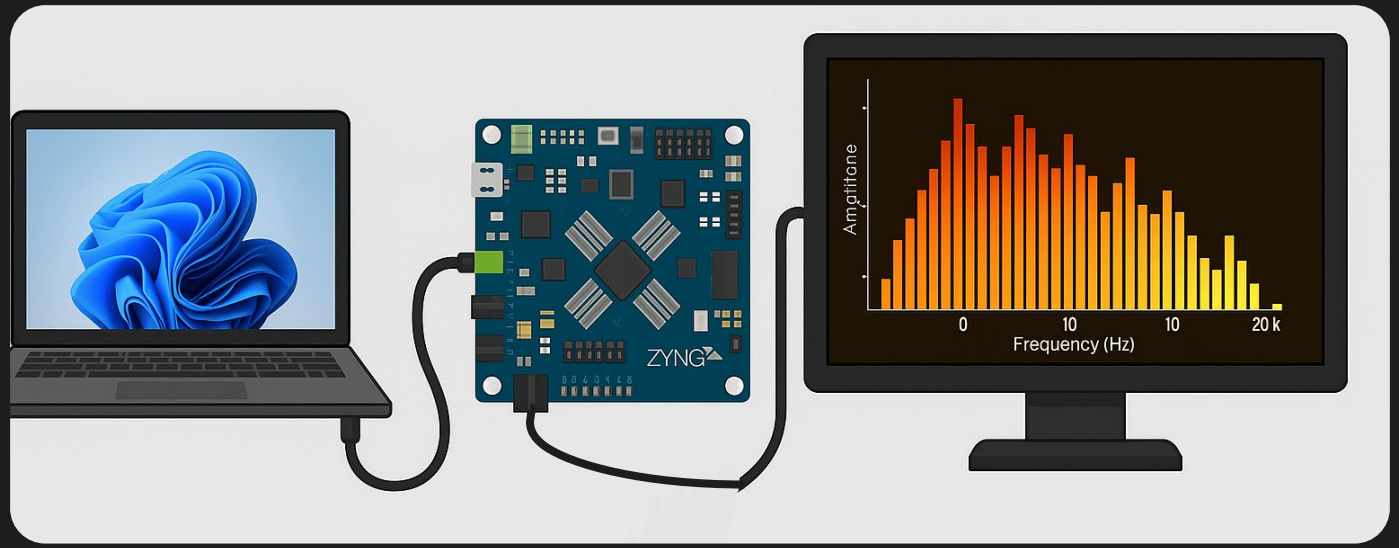
Projet Architecture et Codesign

Affichage VGA d’un signal audio et de sa FFT

Sommaire

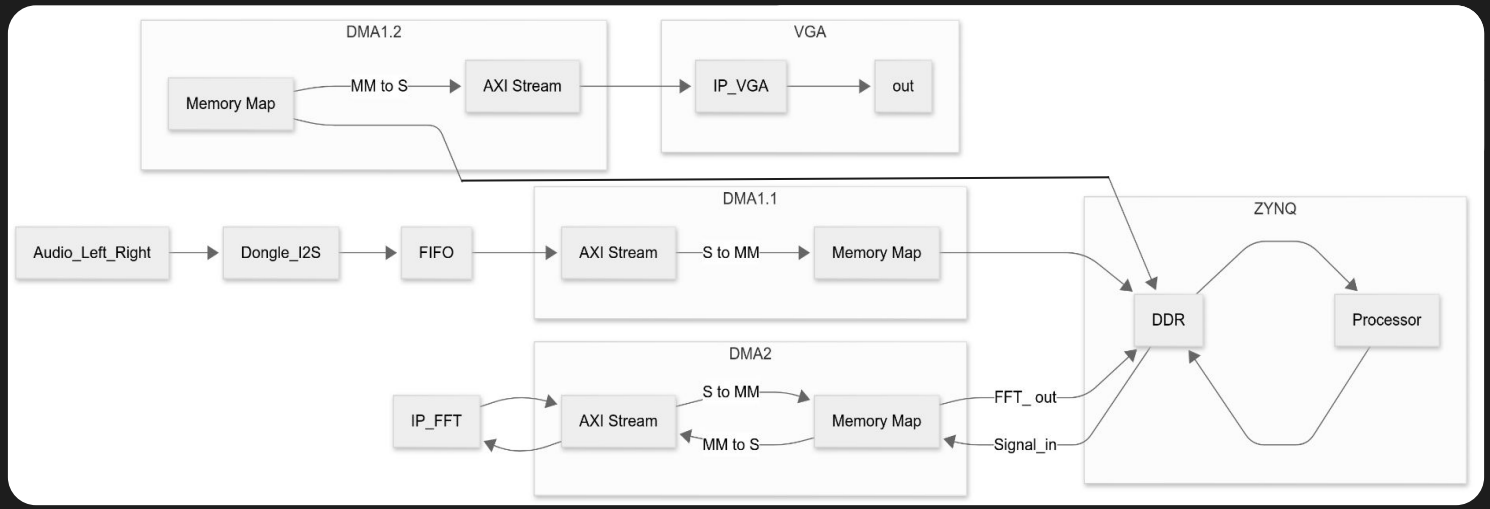
Rappel du sujet



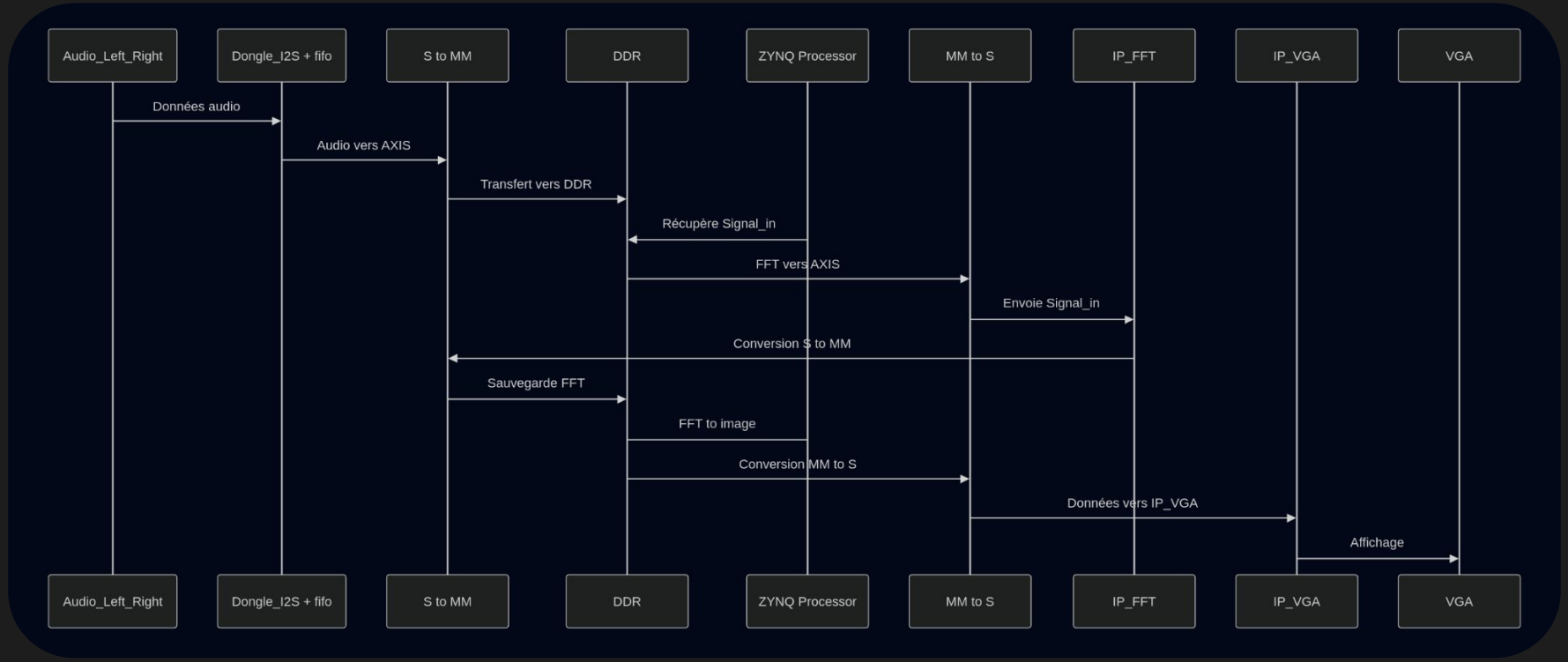
Introduction

Analyseur de spectre :

- IP I2S
- DDR
- DMA
- IP FFT
- IP VGA



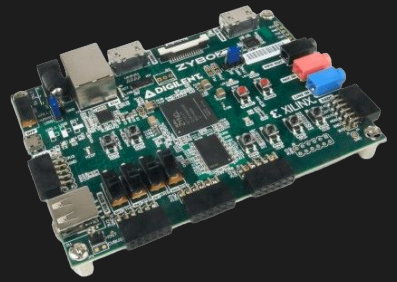
Fonctionnement prévu du projet



Introduction

Technologies et outils :

- FPGA Zynq-7000
- VHDL
- DMA, DDR et AXI-Stream
- FFT IP Core



Contexte et enjeux

- Puissance d’un FPGA
- Optimisation de la gestion des données
- Combiner le FPGA et l’ARM Cortex A9
- FFT IP Core

Défis :

- Assurer la précision et la rapidité de la FFT
- Transferts rapide vers la mémoire et le VGA sans utiliser le processeur

Gestion du projet

Répartition des tâches :

~ Une personne → Une tâche / Un composant

Fonctionnement :

Prise d’informations → Réalisation du composant → Testbench

Gestion du projet

Natao

FFT et formatage de l’image

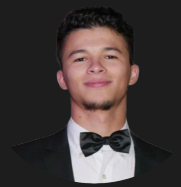
Gabriel Emmanuel

I2S et Dongle

Ali

VGA

Antoine



DMA

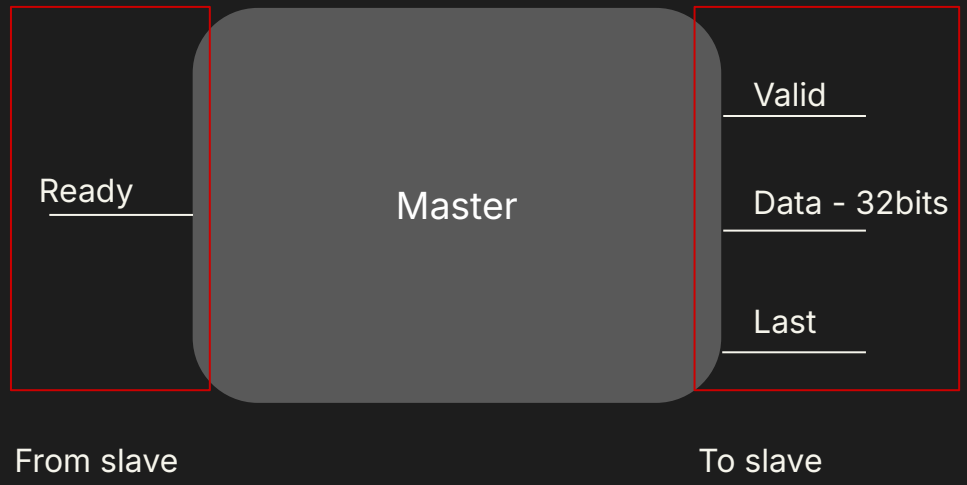
Présentation des composant

Réalisation et tests

AXI STREAM

4 signaux

- Ready
- Valid
- Data
- Last

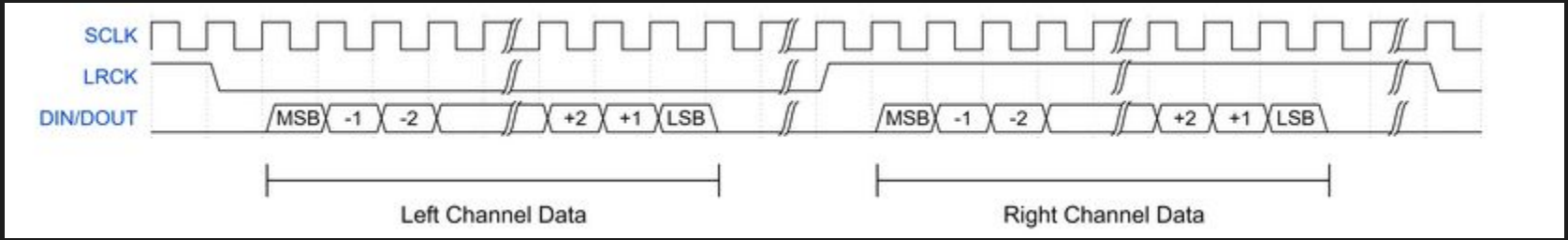


I2S - Protocole

Protocole avec 4 fils

- MCLK
- BCLK
- LRCLK
- SDIN

24 bits de données par canal



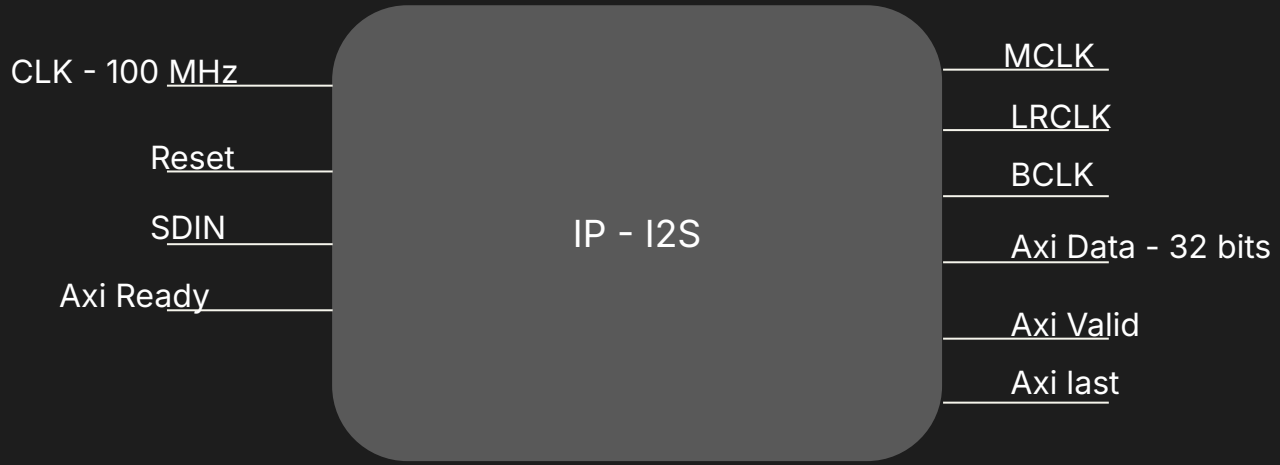
I2S - Implémentation

Choix de la valeurs des clocks

- fs = 44.1 kHz
- MCLK = 256 × fs
- BCLK = 64 × fs

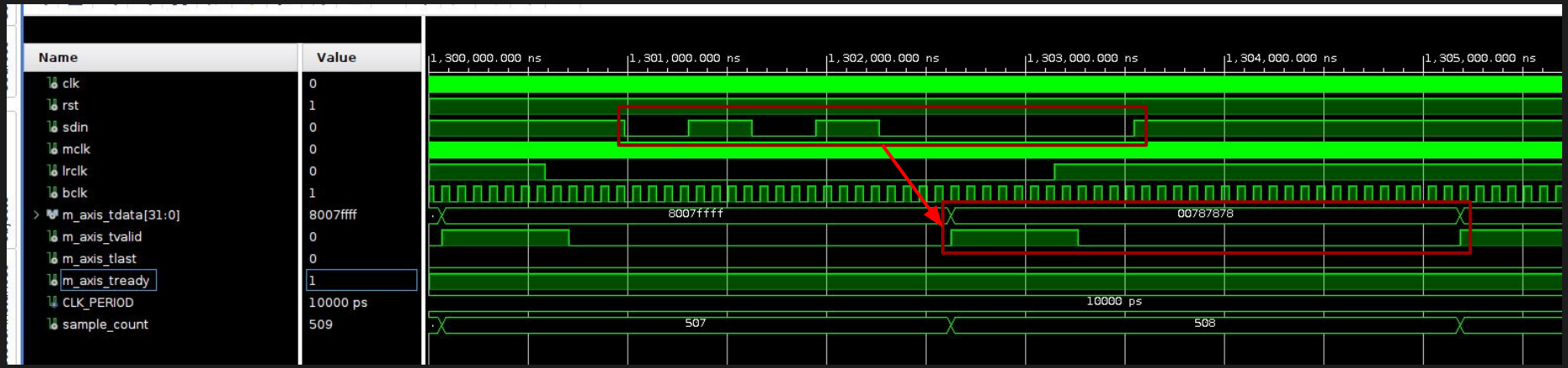
Speed Mode	MCLK/LRCK Ratio	SCLK/LRCK Ratio	Input Sample Rate Range (kHz)
Single-Speed Mode	256x	64	4-24, 43-54
	512x	64	43-54
	384x	64	4-24, 43-54
	784x	64	43-54
Double-Speed Mode	128x	64	86-108
	256x	64	86-108
	192x	64	86-108
	384x	64	86-108

I2S - Implémentation

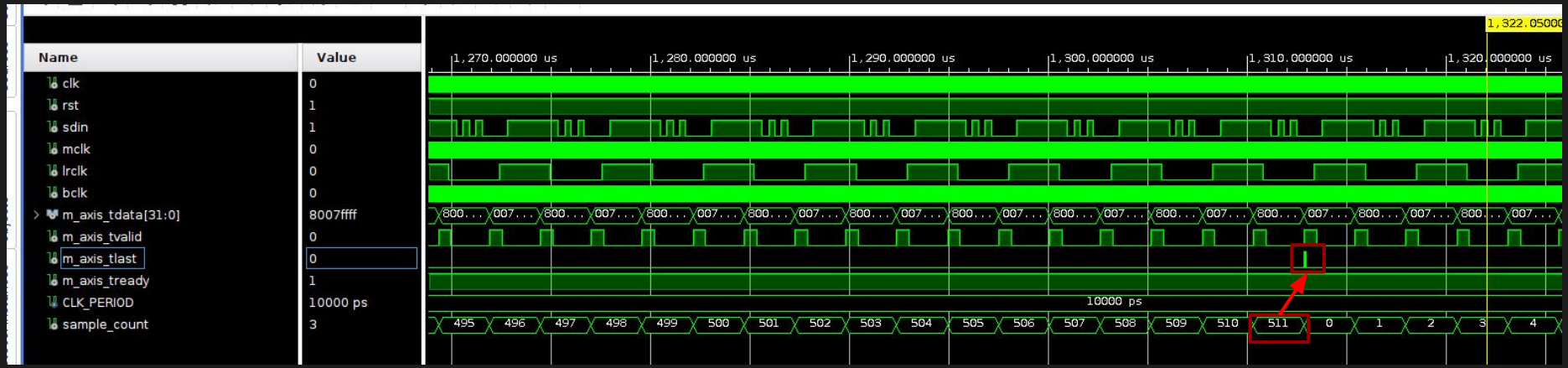


I2S - Implémentation

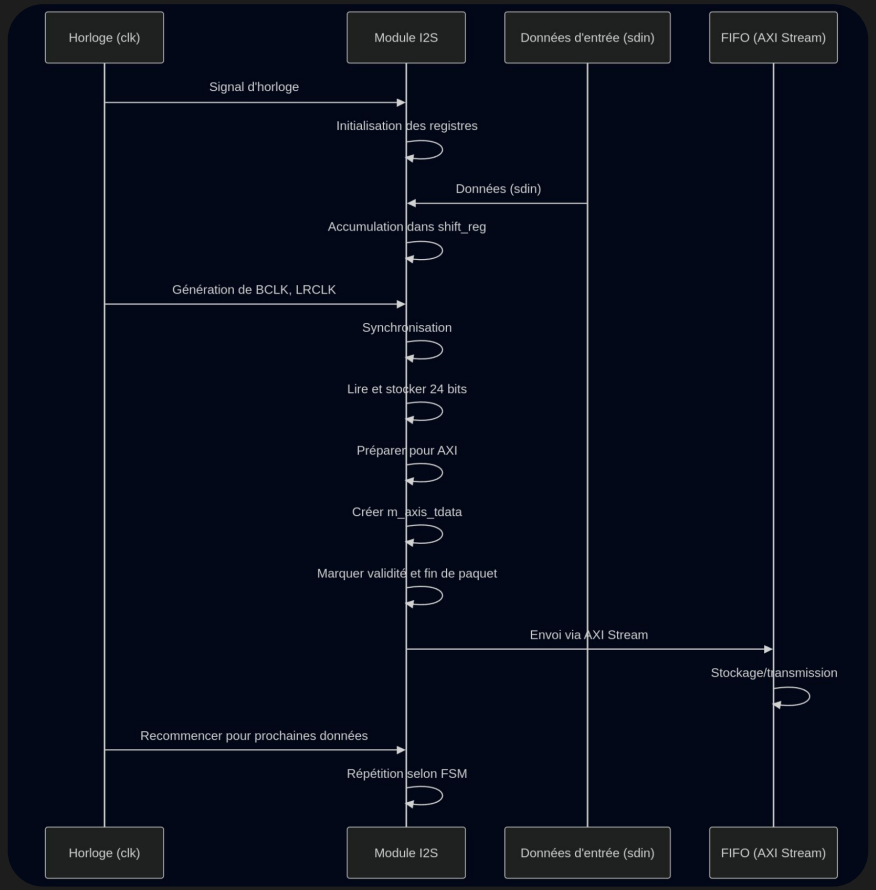
I2S - Simulation



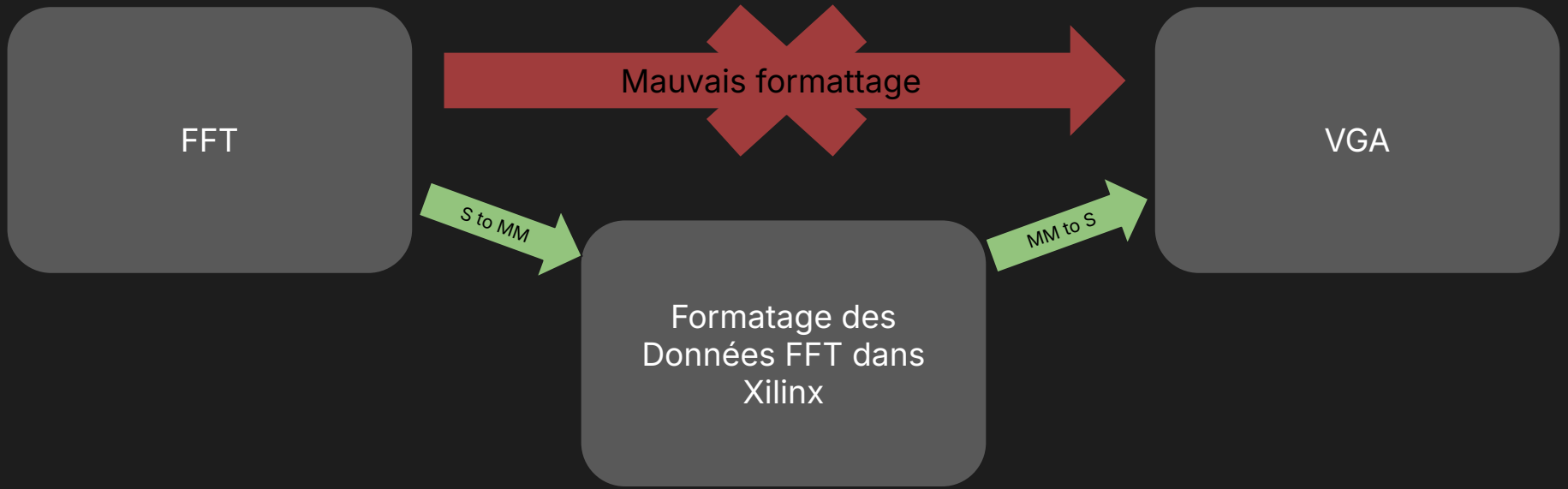
I2S - Simulation



I2S résumé architecture



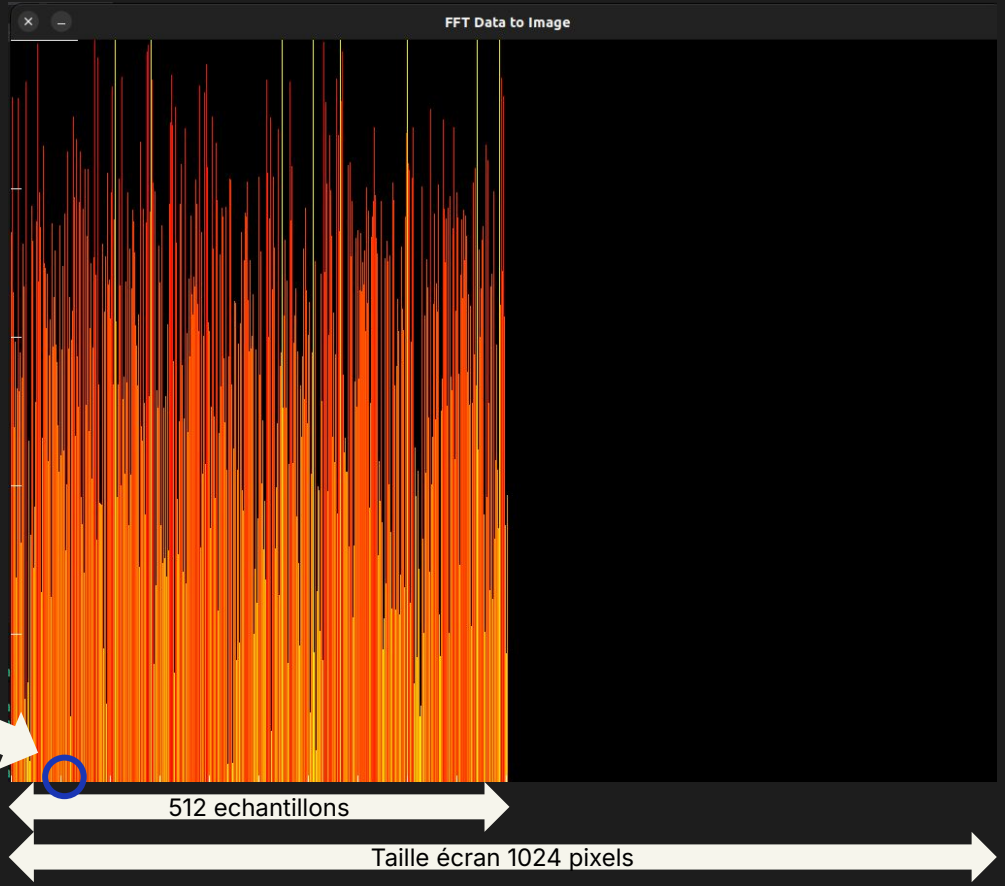
FFT : Contrainte



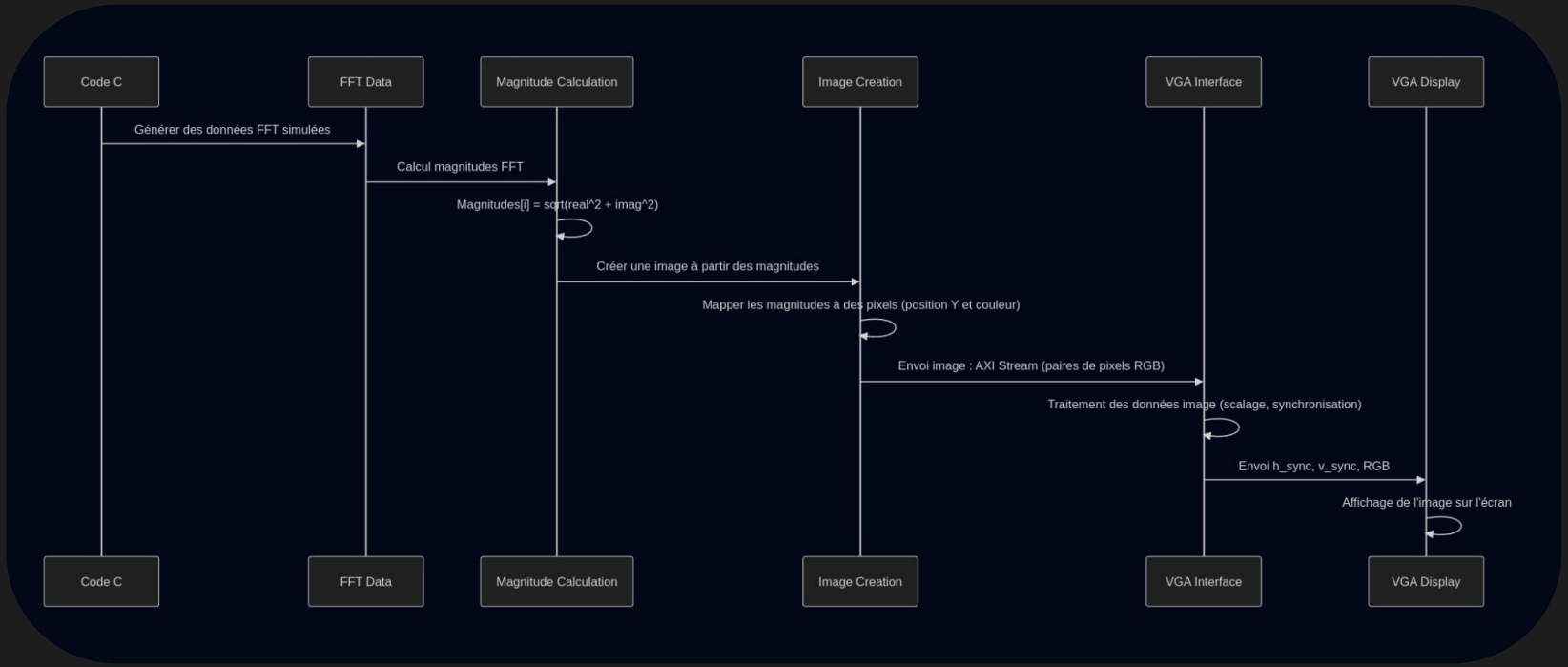
Formatage FFT

Graduations horizontales: 44.1HZ / 512

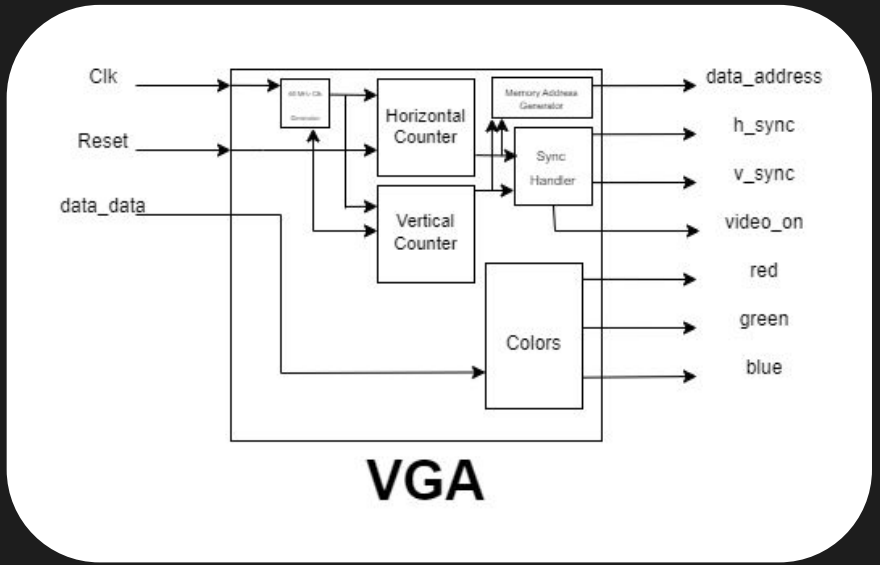
Step = Fe/taille FFT

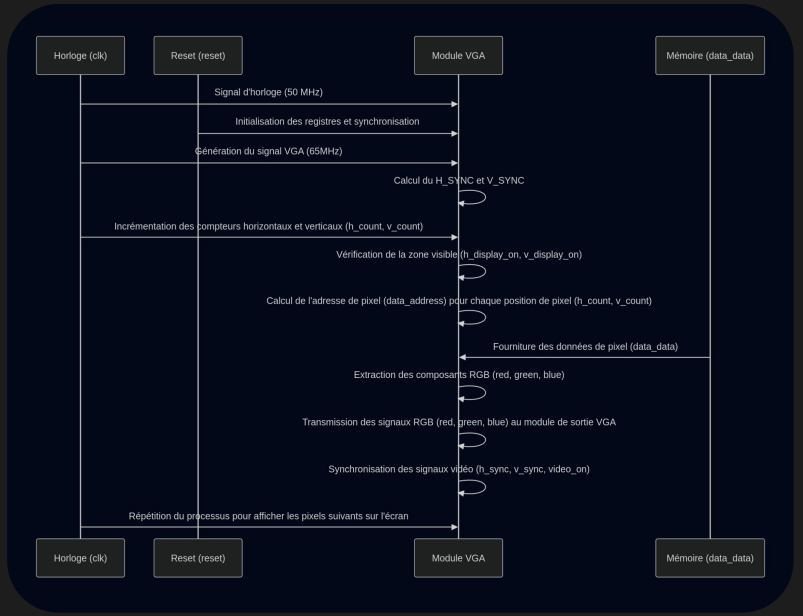


Lien FFT - VGA


Code C
FFT Data
Magnitude Calculation
Image Creation
VGA Interface
VGA Display
Générer des données FFT simulées
Calcul magnitudes FFT
Magnitudes[i] = sqrt(real^2 + imag^2)
Créer une image à partir des magnitudes
Mapper les magnitudes à des pixels (position Y et couleur)
Envoi image : AXI Stream (paires de pixels RGB)
Traitement des données image (scalage, synchronisation)
Envoi h_sync, v_sync, RGB
Affichage de l'image sur l'écran
Code C
FFT Data
Magnitude Calculation
Image Creation
VGA Interface
VGA Display

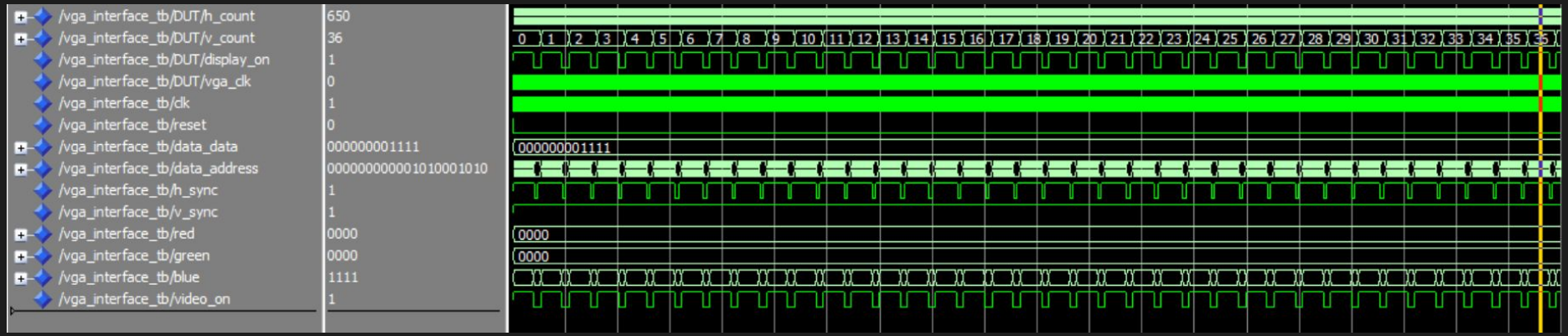
VGA



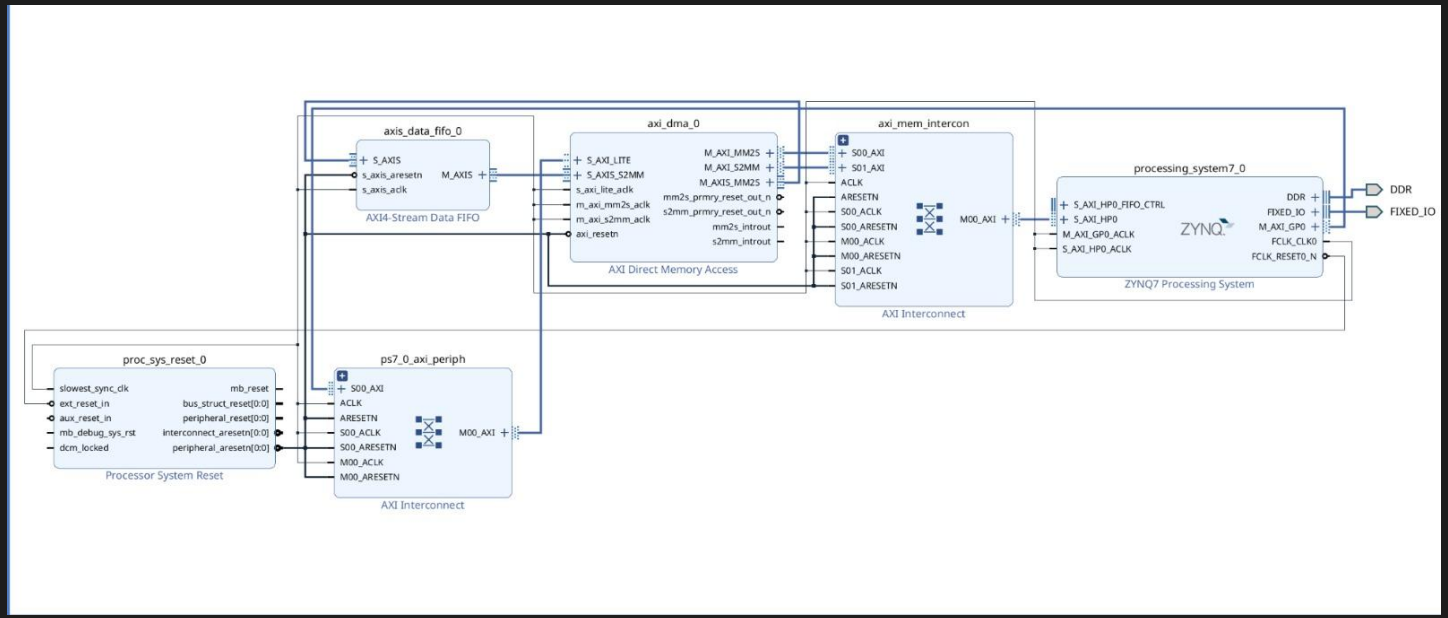


1024x768, 60Hz	65.000	1024	24	136	160	768	3	6	29

VGA


/vga_interface_tb/DUT/h_count
650
/vga_interface_tb/DUT/v_count
36
/vga_interface_tb/DUT/display_on
1
/vga_interface_tb/DUT/vga_clk
0
/vga_interface_tb/clk
1
/vga_interface_tb/reset
0
/vga_interface_tb/data_data
000000001111
/vga_interface_tb/data_address
00000000000101000101O
/vga_interface_tb/h_sync
1
/vga_interface_tb/v_sync
1
/vga_interface_tb/red
0000
/vga_interface_tb/green
0000
/vga_interface_tb/blue
1111
/vga_interface_tb/video_on
1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
000000001111
0000
0000

DMA


axis_data_fifo_0
S_AXIS
s_axis_aresetn
s_axis_aclk
M_AXIS
AXI4-Stream Data FIFO
axi_dma_0
S_AXI_LITE
S_AXIS_S2MM
s_axi_lite_aclk
m_axi_mm2s_aclk
m_axi_s2mm_aclk
axi_resetn
M_AXI_MM2S
M_AXI_S2MM
M_AXIS_MM2S
mm2s_prmry_reset_out_n
s2mm_prmry_reset_out_n
mm2s_introut
s2mm_introut
AXI Direct Memory Access
axi_mem_intercon
S00_AXI
S01_AXI
ACLK
ARESETN
S00_ACLK
S00_ARESETN
M00_ACLK
M00_ARESETN
S01_ACLK
S01_ARESETN
M00_AXI
AXI Interconnect
processing_system7_0
S_AXI_HP0_FIFO_CTRL
S_AXI_HP0
M_AXI_GP0_ACLK
S_AXI_HP0_ACLK
ZYNQ
DDR
FIXED_IO
M_AXI_GP0
FCLK_CLK0
FCLK_RESET0_N
ZYNQ7 Processing System
DDR
FIXED_IO
proc_sys_reset_0
slowest_sync_clk
ext_reset_in
aux_reset_in
mb_debug_sys_rst
dcm_locked
mb_reset
bus_struct_reset[0:0]
peripheral_reset[0:0]
interconnect_aresetn[0:0]
peripheral_aresetn[0:0]
Processor System Reset
ps7_0_axi_periph
S00_AXI
ACLK
ARESETN
S00_ACLK
S00_ARESETN
M00_ACLK
M00_ARESETN
M00_AXI
AXI Interconnect

DMA

[bare metal (C)]

↓

(AXI4-Lite → Registers)

↓

————MM2S ————

DDR ←DataMover (Read)

↓

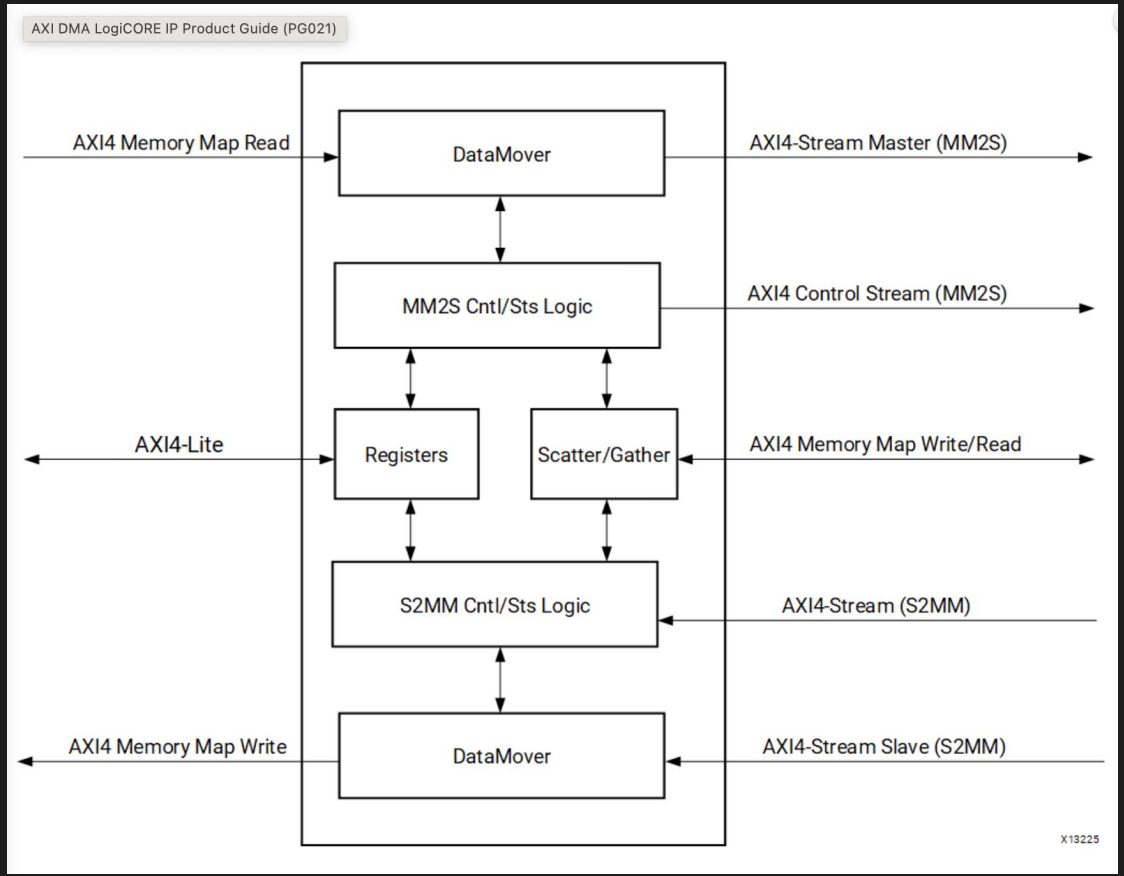
AXI4-Stream Master → FIFO

————S2MM ————

FIFO → AXI4-Stream Slave

↓

DataMover (Write) → DDR



DMA

// Offsets MM2S
#define DMA_MM2S_DMACR 0x00 // Contrôle du DMA MM2S (start, reset, etc.)
#define DMA_MM2S_DMASR 0x04 // Status du DMA MM2S (erreurs, fin de transfert...)
#define DMA_MM2S_SRC_ADDR 0x18 // Adresse source en mémoire (DDR)
#define DMA_MM2S_LENGTH 0x28 // Longueur du transfert en octets

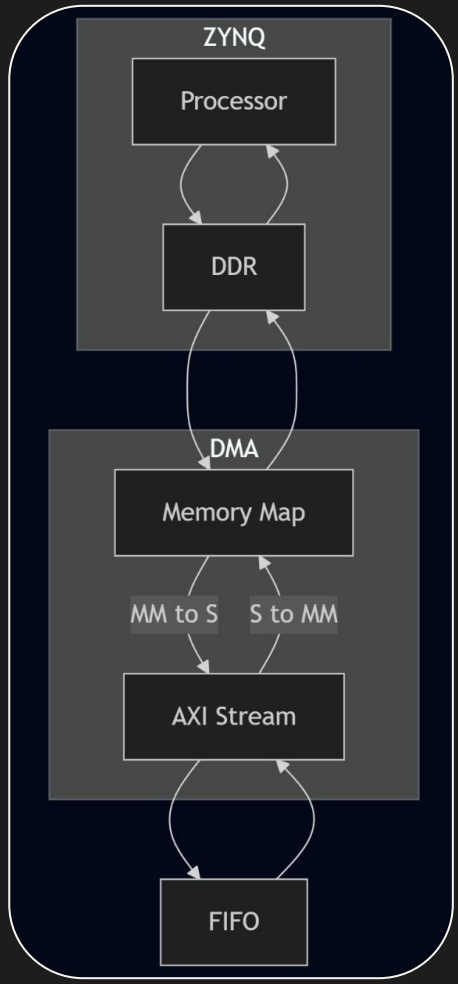
Address Space Offset [1]	Name	Description
00h	MM2S_DMACR	MM2SDMA Control register
04h	MM2S_DMASR	MM2SDMA Status register
08h	MM2S_CURDESC	MM2S Current Descriptor Pointer. Lower 32 bits of the address.
0Ch	MM2S_CURDESC_MSB	MM2S Current Descriptor Pointer. Upper 32 bits of address.
10h	MM2S_TAILDESC	MM2S Tail Descriptor Pointer. Lower 32 bits.
14h	MM2S_TAILDESC_MSB	MM2S Tail Descriptor Pointer. Upper 32 bits of address.
2Ch [2]	SG_CTL	Scatter/Gather User and Cache
30h	S2MM_DMACR	S2MM DMA Control register
34h	S2MM_DMASR	S2MM DMA Status register
38h	S2MM_CURDESC	S2MM Current Descriptor Pointer. Lower 32 address bits
3Ch	S2MM_CURDESC_MSB	S2MM Current Descriptor Pointer. Upper 32 address bits.
40h	S2MM_TAILDESC	S2MM Tail Descriptor Pointer. Lower 32 address bits.
44h	S2MM_TAILDESC_MSB	S2MM Tail Descriptor Pointer. Upper 32 address bits.

DMA

```
Xil_Out32(DMA_BASE_ADDR + DMA_S2MM_STRIDE, 0x0);
Xil_Out32(DMA_BASE_ADDR + DMA_S2MM_LENGTH, 128);

uint32_t table_to_send[200] = {0x11, 0x22, 0x33};
uint32_t received_table[200] = {0x00, 0x00, 0x00};

uint32_t received_table[200] = {0x11, 0x22, 0x33};
```

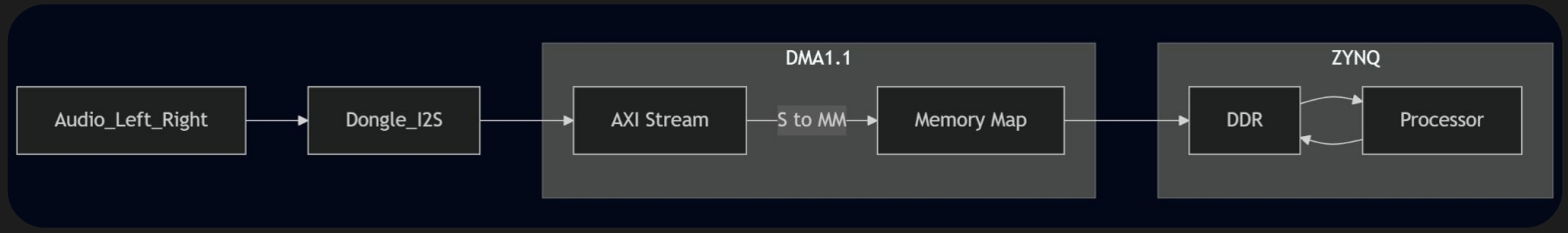


DMA

Fonctionnement attendu :

Réception des données par le dongle I2S à une adresseprécise.

❌ `uint32_t table_to_send[200] = {0x11, 0x22, 0x33};`
✅ `#define DMA_BASE_ADDR 0x40400000`



Conclusion

- Limites et difficultés
- Projet presque finalisé
- Perspectives

Merci pour votre attention