Instrumentation Temps Réel

Développement FPGA et Python avec l'Écosystème PYNQ

Fabrice Wiotte

Laboratoire de Physique des Lasers (LPL)

CNRS - Université Sorbonne Paris Nord

AG FIRST-TF 2026

Workshop Électronique numérique et IA pour le temps fréquence

Qu’est-ce que l'écosystème PYNQ ?

PYNQ est un projet open source de **AMD** - **Xilinx** qui permet de programmer et d'utiliser « facilement » des puces FPGA en utilisant le langage Python et des **Notebook** Jupyter directement depuis un **navigateur web**.



Abstraction Python

Accès direct AXI/DMA en Python
Élimine les drivers C/C++



Overlay Hardware

FPGA = Accélérateur dynamique
Chargement à la volée



Jupyter Interactive

Widgets de visualisation RF
Prise en main et débogage simplifiés

Le FPGA piloté par Python : PYNQ



Base Linux Embarquée

Ubuntu sur cœurs ARM : gestion native du réseau, des fichiers et des bus AXI.



Développement Interactif

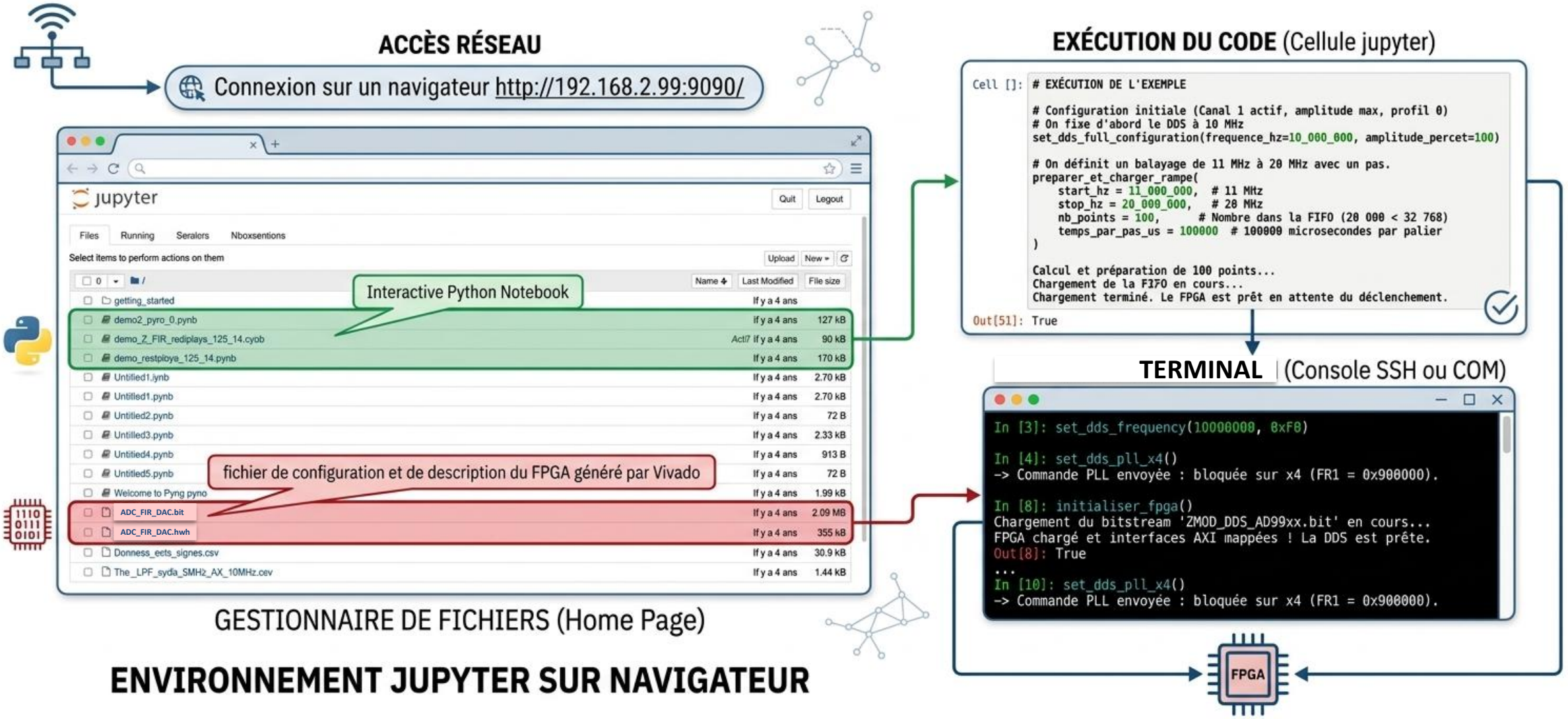
Codez, contrôlez et analysez vos données sans aucune installation, directement depuis votre navigateur.



Accélération Dynamique

Fini la "boîte noire" : chargez vos configurations FPGA à la volée via de simples appels Python.

Interface Jupyter Notebook



Pilotage avancé du FPGA

Accès Console & SSH (classique)

- **Ligne de commande :** Connexion directe via SSH ou port COM.
- **IPython interactif :** Chargement des bitstreams sans interface web.

```
xilinx@pynq:~$ sudo -E ipython
In [1]: from dds_control import *
In [2]: initialiser_fpga()
Chargement du bitstream
'ZMOD_DDS_AD99xx.bit' en cours...
 FPGA chargé et interfaces AXI mappées ! La
DDS est prête. Out[2]: True
```

Performances C/C++

- **Compilation GCC :** Appel direct du compilateur natif Linux.
- **Module Cython :** Écriture et exécution C/C++ à la volée.

```
In [10]: !gcc chargement_freq_canal.c -o
chargement_freq_canal

In [11]: !sudo ./chargement_freq_canal
18000000 0xF0
>> Succès : Canal selectionne [0xF0] |
Frequence envoyee : 18000000 Hz
In [12]: _
```

Ressources

Site Officiel PYNQ : **www.pynq.io**

Support Officiel

PYNQ-Z1 (Digilent)
PYNQ-Z2 (TUL)
ZCU104 / ZCU111/ ZCU208
Kria KV260 / KR260
RFSoC 4x2 / 2x2 …

Communauté & Custom

Eclypse Z7 (via image custom)
Redpitaya 125_14 (via image custom)
Ultra96-V2 (Avnet)
Zybo Z7 / Cora Z7
Architecture Zynq-7000
Min. 512MB RAM & Network

Lien & Utilisation

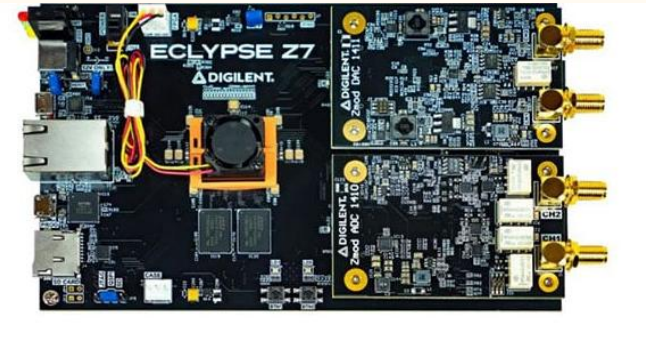
https://www.pynq.io/boards.html
Documentation & Tutoriels :
pynq.readthedocs.io
Accès Jupyter par défaut :
http://pynq:9090
https://pynq.readthedocs.io/en/latest/getting_started.html

Official boards : PYNQ-Z2 | PYNQ-Z1 | ZCU104 | AUP-ZU3 | PYNQ-ZU | ZCU208 | ZCU111 | RFSoC 2x2 | RFSoC 4x2

Communauté : https://www.pynq.io/community.html

Un écosystème matériel hétérogène

PYNQ v2.7 (custom)



Digilent Eclypse Z7

Base Zynq-7000 SoC. Modulaire et polyvalente avec ports haut débit SYZYGY (Zmod AWG/DAC 1411-125 et Scope/ADC 1410-125).

PYNQ v3.0.1 (custom)



Red Pitaya 125-14

Laboratoire intégré. 2 ADC / 2 DAC à 125 MSPS (14-bit). Carte compacte idéale pour l'instrumentation de paillasse.

PYNQ v3.x.x (official)



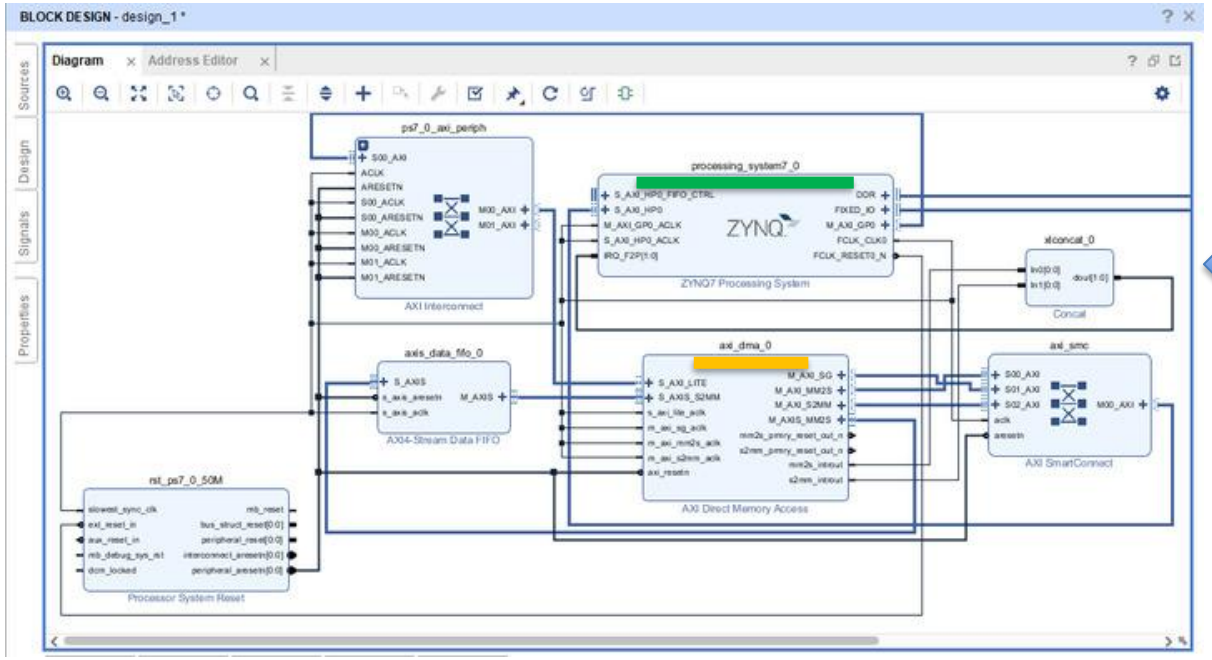
RealDigital RFSoC 4x2

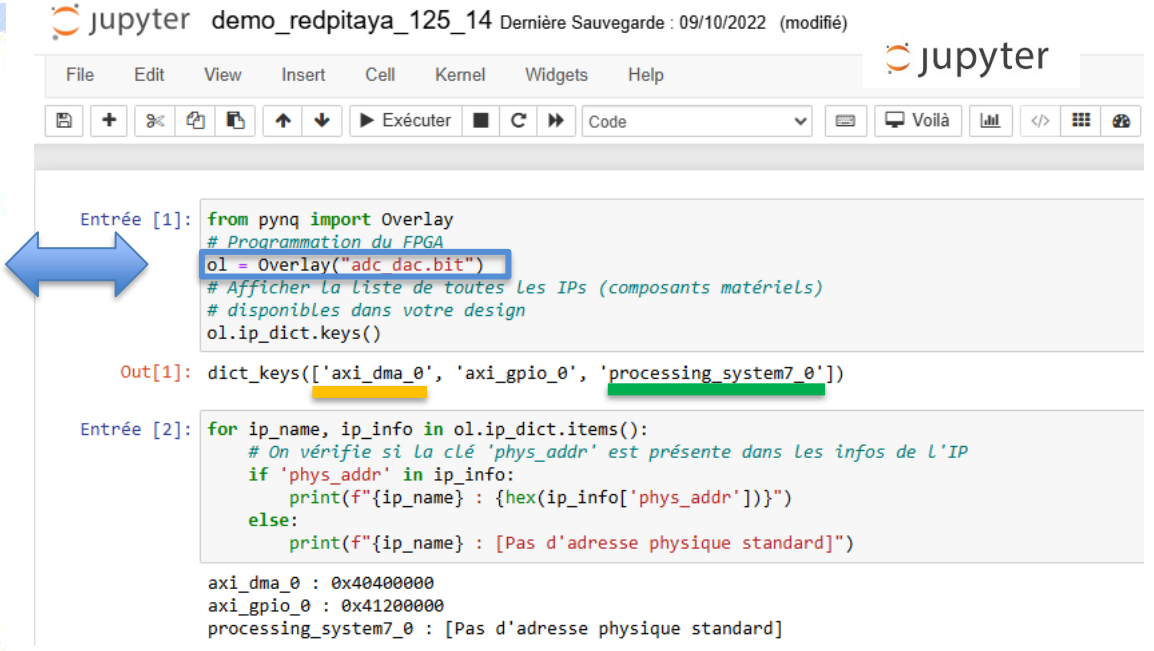
Conversion RF directe ultra-rapide. ADC 5 GSPS & DAC 9.85 GSPS. Prototypage avancé et traitement temps réel massif.

Le concept d'Overlay avec PYNQ !

Un Overlay est un circuit FPGA précompilé vu comme une application matérielle.

⚠ La première chose à faire dans un Notebook Jupyter c'est de charger le programme FPGA et d'analyser le fichier .hwh pour créer les liens vers les composants matériels.





Hardware as a Library !

Partie Hardware ⇔ VIVADO ⇔ « Overlay »

Choix d'Architecture (1/2) : Bare-Metal vs PYNQ

Bare-Metal Traditionnel

Déterminisme : Latence prédictible < 1 µs.

Cas d'usage : Asservissements ultra-rapides.

Contrainte : Développement bas niveau complexe (C/C++ pur) et pas d'OS.

Écosystème PYNQ (Linux)

Productivité : Programmation Python interactive (x10).

Cas d'usage : R&D, prototypage rapide et acquisition massive.

Compromis : Overhead lié au système d'exploitation (~10-100 µs).

Choix d'Architecture (2/2) : Linux Embarqué vs PYNQ

Linux Embarqué (Yocto / Petalinux)

Contrôle : Personnalisation profonde et optimisation du système d'exploitation.

Cas d'usage : Déploiement industriel, systèmes figés et produits commerciaux finis.

Contrainte : Cycle d'itération très lourd (recompilation fastidieuse du noyau Linux à chaque modification matérielle sous Vivado).

Écosystème PYNQ (Linux)

Agilité : Environnement pré-packagé avec serveur web et Jupyter embarqués.

Cas d'usage : Instrumentation de laboratoire agile et reconfigurable à la volée.

Avantage décisif : Chargement dynamique du matériel (Overlay) depuis un script Python, sans jamais redémarrer la carte ni recompiler l'OS.

Développement FPGA : choix de la carte Eclypse Z7

Intérêt principal : Prototypage rapide, acquisition haut débit, ports Zmod dédiés aux ADC DAC, schémas et documentation matérielle accessibles.

Exemples de projets LPL :

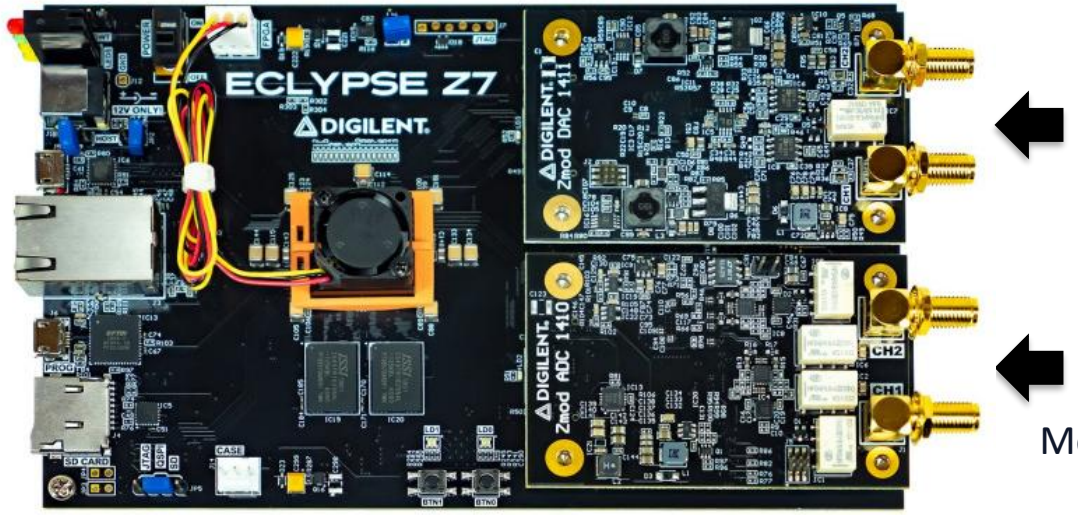
- *Réjection de RAM LPL ANR BRIOCHE (F. Wiotte)*
- *Lock Cavité Ultra Stable REFIMEVE (O. Lopez)*

Norme SYZYGY : Transfert de données ultra-rapide – cartes modulaires interchangeables.

Coût : 450€ la carte FPGA seule.







Modules Zmod **Digilent** au format SYZYGY

Extension Matérielle sur Eclypse Z7 : Le standard SYZYGY

- **Le standard ouvert :** Créé par Opal Kelly, SYZYGY est une connectique haute performance offrant un compromis entre la lenteur du standard Pmod et le coût élevé du format FMC.

- **Intégration sur Eclypse Z7 :** Exploitée nativement par les cartes Digilent (via la nomenclature Zmod), cette interface permet des transferts de données à haut débit.

- **Développement LPL :** Le service électronique du LPL développe ses propres cartes filles (DDS) sur ce format pour s'adapter précisément aux besoins expérimentaux.

INNOVATION SYZYGY :
PLATEFORMES DE CONCEPTION FLEXIBLES



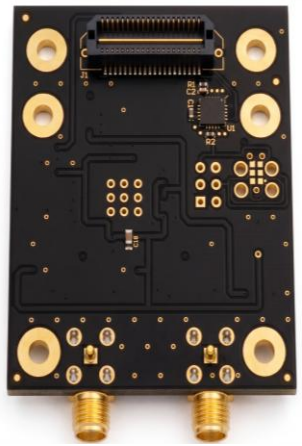
SZG-ADC-LTC2264
OPAL KELLY
dual 40 MSPS 12-bit ADC



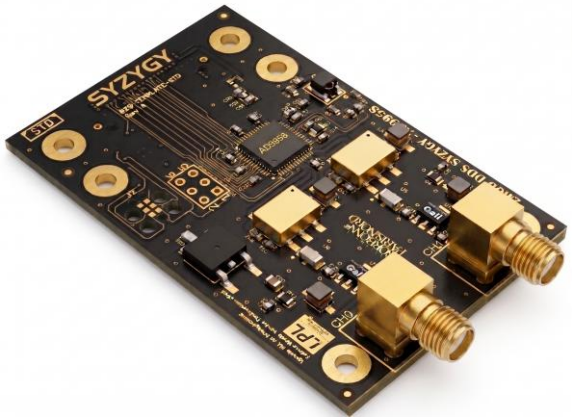
ZMOD SCOPE DIGILENT
ADC - AD9648
dual 125 MSPS 14-bit ADC

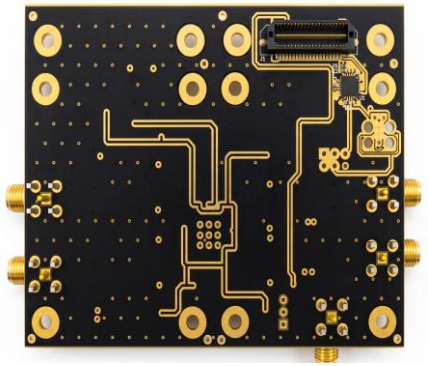
Modules commerciaux

Développements LPL : modules DDS AD99xx

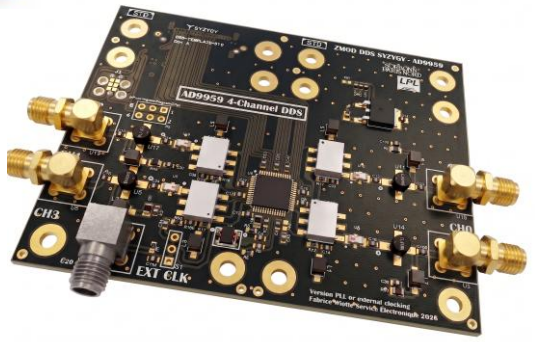


Module SYZYGY **LPL**
DDS 2 Canaux
AD9958 500MSPS
70 x 45 mm



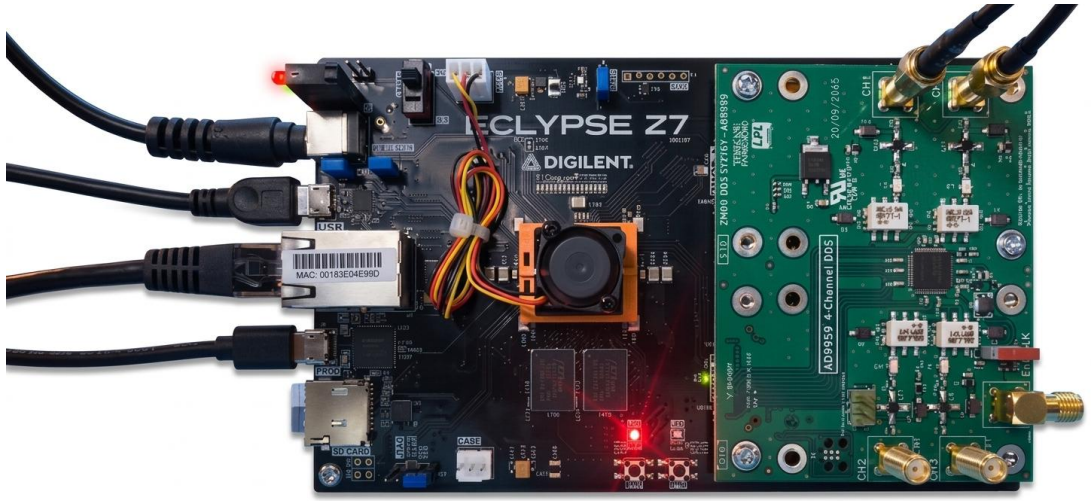


Module SYZYGY **LPL**
DDS 4 Canaux
AD9959 500MSPS
70 x 90 mm



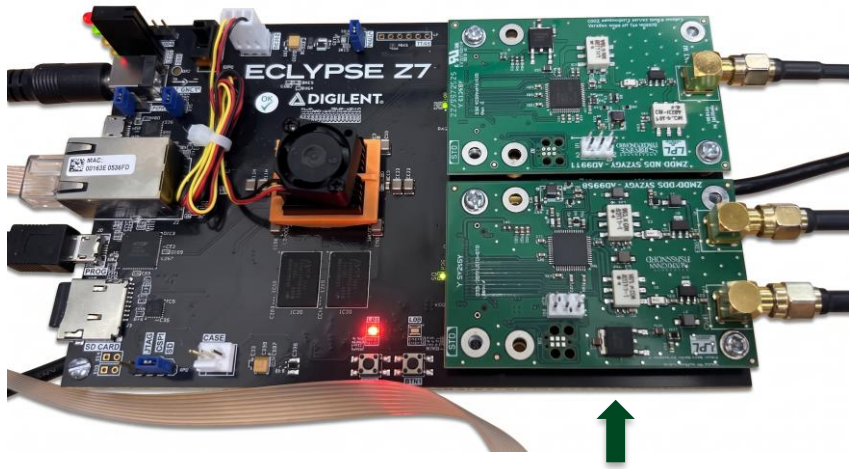
Module SYZYGY DDS 2 ou 4 Canaux

Module 2 ou 4 Canaux AD99xx sur carte FPGA Eclypse z7



Module SYZYGY LPL
DDS 4 Canaux **AD9959 500MSPS**
Fout de 50KHz à 200MHz

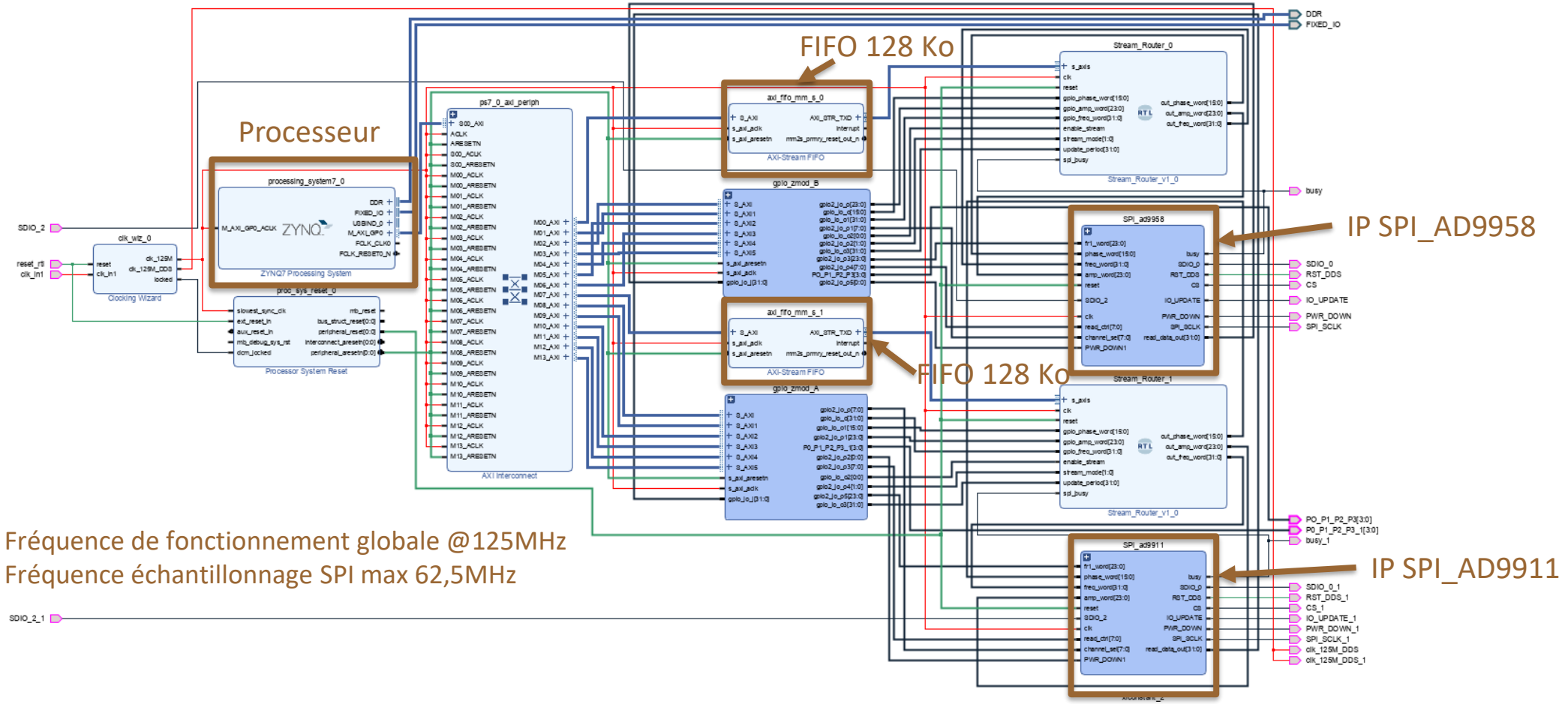




Module SYZYGY LPL
DDS 2 Canaux **AD9958 500MSPS**
Fout de 50KHz à 200MHz

Architecture matérielle développée : SYZYGY AD99xx

- **Conception FPGA sur mesure :** Exemple de développement VHDL sous Vivado.



Conclusion

Choix Stratégique au Laboratoire

L'écosystème PYNQ et la modularité des FPGA rendent l'accélération matérielle « transparente ». Cela nous permet de concevoir une instrumentation agile, compacte et reconfigurable à la volée, sans alourdir le développement.

Merci de votre attention.

Fabrice Wiotte | LPL, CNRS - Université Sorbonne Paris Nord