Ecole d'ingénieurs Sup Galilée

Spécialité Instrumentation

TP projet FPGA 3

Réalisation de prototype pour l'embarquer

Carte électronique moteur pas à pas

Programmation VHDL et Python

Connection SSH sur Raspberry pi - écran tactile gen4-uLCD-43DCT

Travaux pratiques

Fabrice Wiotte Laboratoire LPL

Sommaire

Présentation du TP RPPE
Réalisation d'un prototype pour l'embarquer :

L'ingénieur doit concevoir et optimiser un système complexe (conception électronique, logicielle, matériel embarqué, modélisation, simulation et prototypage)

L'objectif de ce TP est de développer sa propre conception électronique pour l'intégrer dans un ensemble plus complexe. Intégrer et développer des sous-ensembles électroniques qui pilotent un ou plusieurs moteurs pas à pas en logique numérique, interfacer en langage de haut niveau et communiquer à distance avec pour exemple une Raspberry Pi4 connectée en SSH.

Dans ce TP on ne fera pas l'étude pour implémenter un composant FPGA sur un PCB, le FPGA à souvent un boitier du type BGA (billes de connexions) et trop complexe à mettre en œuvre. Il est possible pour des experts en conception électronique CAO d'intégrer leur propre FPGA dans leur design. Dans le cadre des TP on restera sur des kits de développement FPGA commerciaux, sur lesquels on développera des cartes filles.

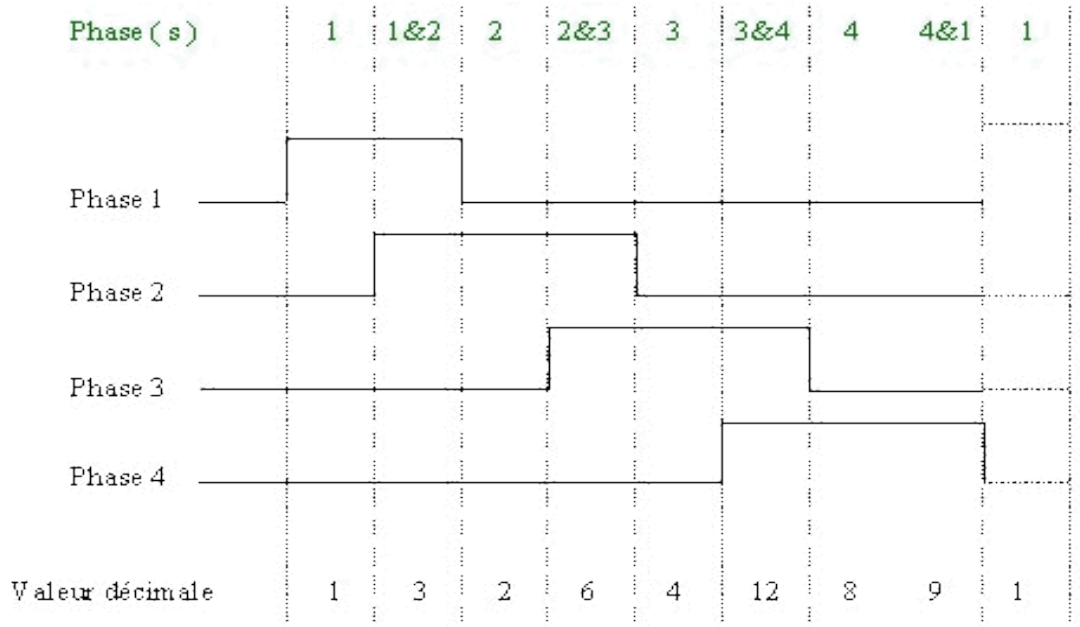
1.1 Objectif :

Développer des cartes modulaires qui s'adapterons sur un kit FPGA commercial pour contrôler un moteur pas à pas. On utilisera le moteur pas à pas et le programme VHDL étudié l'année précédente qui sera à adapté pour la nouvelle cible FPGA, voir description TP **FPGA 2**.

Rappel du TP FPGA2 :
Contrôler d'un moteur numérique du type pas à pas, sens de rotation, vitesse de rotation, contrôle du pas en mode local par potentiomètre numérique et/ou à distance avec une liaison USB-série avec le pont UART à disposition sur la carte FPGA.
Le moteur fonctionne en boucle ouverte non asservi, moteur utilisé : 28BYJ-48 5V moteur unipolaire à 4 enroulements avec réducteur, driver moteur ULN2003 et encodeur PEC11R 24 pas/tours.
Le moteur à une réduction de 1/64 soit 360 degrés/64 = 5.625 degré/pas.
En mode demi-pas qui correspond à 8 états et 4 phases la réduction finale est de 4096 (64x64).
Le moteur aura un pas unitaire angulaire de 5.625/64 = 0,0879 degrés !
Vitesse max du moteur = 255Hz

TP Stepper Motor

CW →



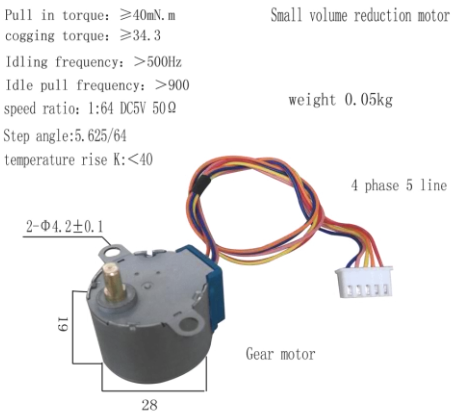
← CCW

Chronogramme de fonctionnement en mode demi pas du moteur

Choix du moteur :

28BYJ-48 5V moteur unipolaire

- Alimentation : 5 Vcc
- Résistance : 21 ohms
- Intensité : 25 mA
- Réduction : 1/64
- Nombre de pas par tour : 64 (réduction de 4096 en sortie d'axe)
- Entraxe de fixation : 35 mm
- Axe : Ø5 mm avec double méplat (épaisseur 3 mm)
- Longueur de l'axe : 12 mm



1.2 Outils et logiciels

Le composant numérique principal choisi pour cette carte pour piloter le moteur pas à pas est un FPGA de la famille Xilinx l'Artix 7-35T disposé sur **un kit Digilent, le kit CMODA7 35T**.

Carte Cmod A7 Digilent

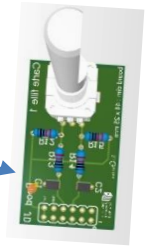
https://digilent.com/shop/cmod-a7-35t-breadboardable-artix-7-fpga-module/

La gamme Xilinx à découvrir :
https://www.xilinx.com/products/silicon-devices/fpga.html

Le code VHDL du **TP FPGA 2** est à adapter sur la cible CmodA7 sous Vivado.
Les cartes filles seront à développer sous ALTIUM. La carte CmodA7 est auto-alimentée par l'USB. Enfin on utilisera un Raspberry pi 4 en démonstration pour tester une communication distante (SSH) si le temps le permet.

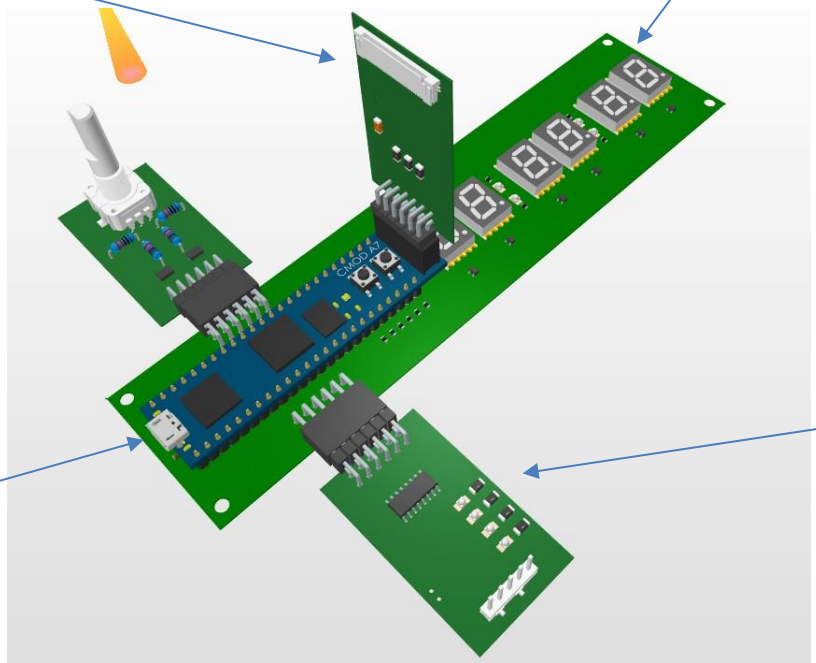
1.3 Cartes filles à développer

Encodeur
Carte fille 1

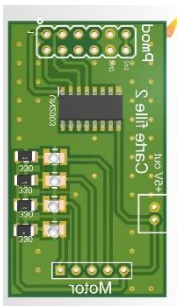


Carte Adaptateur CmodA7

Carte adaptation Ecran tactile (en démonstration)



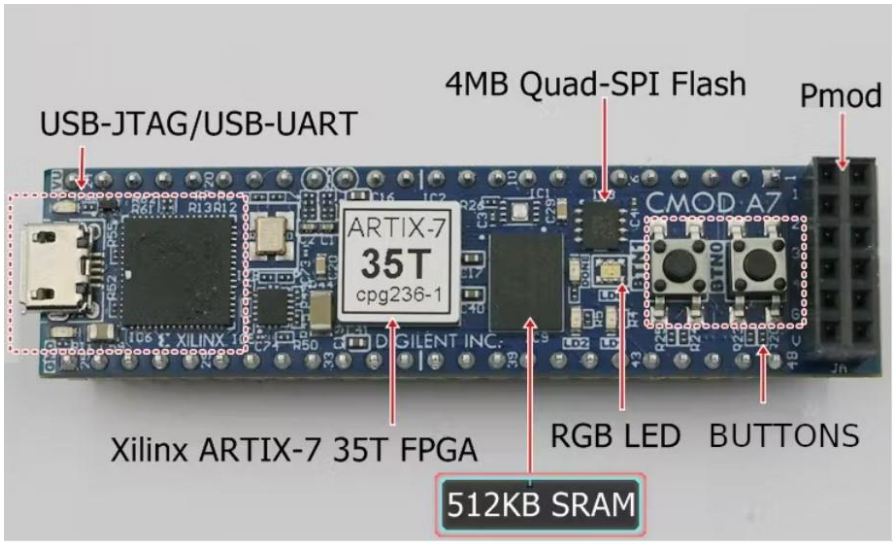
Power and
debug



Driver Moteur carte fille 2

Visuel cartes filles à réaliser 2 couches : composants disposés sur un côté

2. Présentation de la carte FPGA utilisée
CmodA7



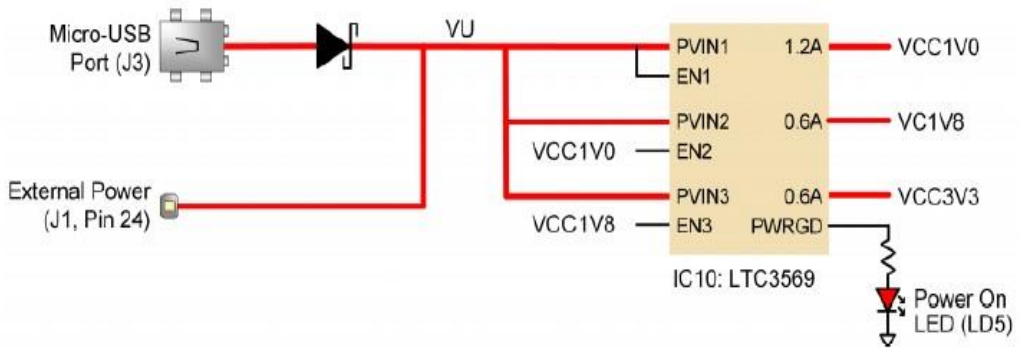
Site Web DIGILENT:
https://digilent.com/reference/programmable-logic/cmod-a7/start

Product Variant	Cmod A7-15T	Cmod A7-35T
FPGA Part	XC7A15T-1CPG236C	XC7A35T-1CPG236C
1 MSPS On-chip ADC	Yes	Yes
Programming options	Quad-SPI Flash/JTAG	Quad-SPI Flash/JTAG
Look-up Tables (LUTs)	10,400	20,800
Flip-Flops	20,800	41,600
Block RAM	112.5 KB	225 KB
Clock Management Tiles	5	5

https://www.xilinx.com/products/silicon-devices/fpga/artix-7.html#productTable

Présentation de la carte FPGA CmodA7

Alimentation et puissances

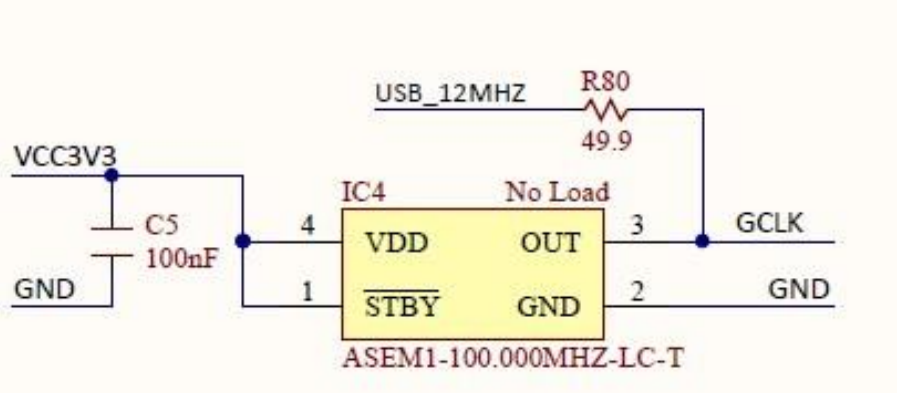


VCC1V0 → **FPGA Core and Block RAM**

VCC1V8 → **FPGA AUX and ADC, USB Core**

VCC3V3 → **SRAM, USB Controller, PMOD, LEDS, Buttons, FPGA USER I/O**

Les Horloges



La carte Cmod A7 comprend une seule entrée d'horloge de 12 MHz reliée à la broche L17
(L17 est une entrée MRCC sur la banque 14).
l'horloge par défaut est de 12 MHz, partagée avec l'USB

Présentation de la carte FPGA CmodA7

Quad SPI Flash

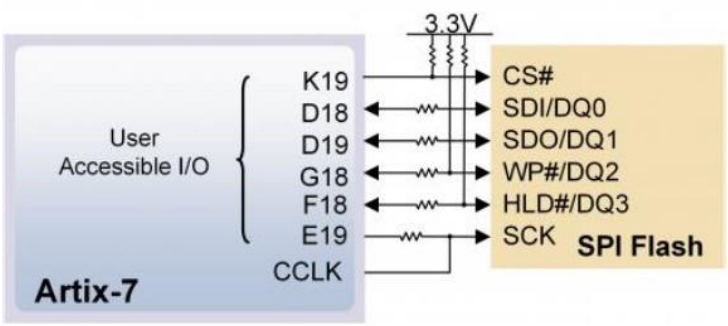
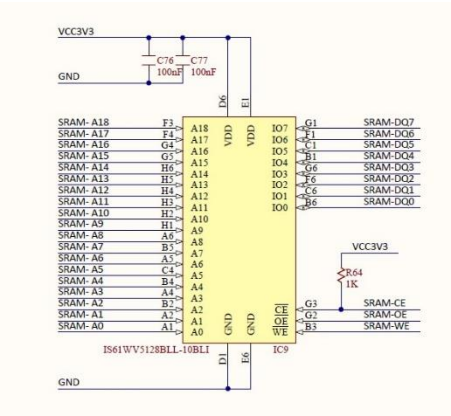


Figure 4.1. Cmod A7 Quad-SPI Flash.

La mémoire du FPGA sur le Cmod A7 étant volatile, elle s'appuie sur la mémoire flash Quad-SPI pour stocker la configuration entre les cycles d'alimentation.

SRAM 512KB



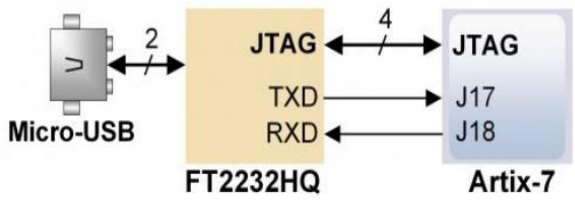
Le Cmod A7 comprend 512 Ko de mémoire statique à accès aléatoire (SRAM).

ISSI IS61WV5128BLL-10BLI

Cette mémoire possède une interface parallèle standard, facile à utiliser, avec 19 signaux d'adresse, 8 signaux de données bidirectionnels et 3 signaux de contrôle.

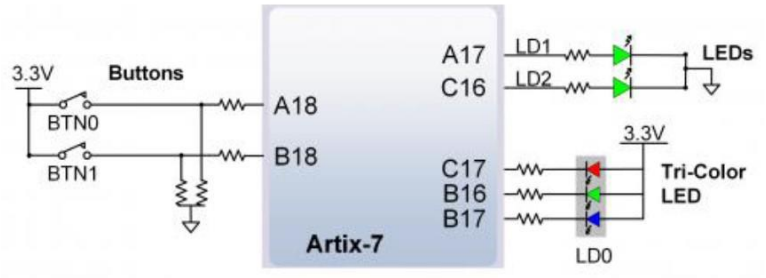
Présentation de la carte FPGA CmodA7

USB-UART Bridge



Le Cmod A7 comprend un pont USB-UART FTDI FT2232HQ (fixé au connecteur micro USB).
Convertir les paquets USB en données de port UART/série(Rx et TX)
Le FT2232HQ est également utilisé comme contrôleur pour le circuit Digilent USB-JTAG

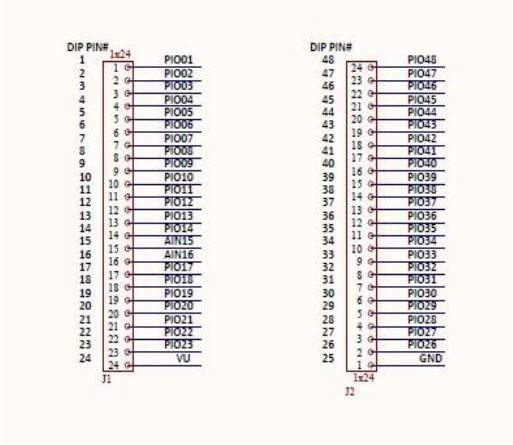
RGB LEDS



2 boutons poussoirs, 2 LED vertes, 1 LED RGB

Présentation de la carte FPGA CmodA7

Input /Output



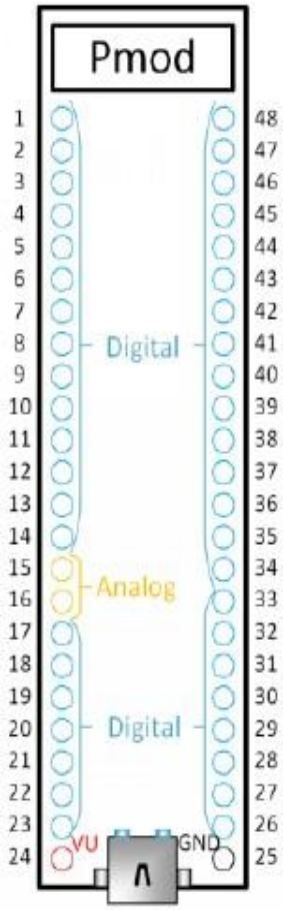


Figure 8.1. DIP pin diagram.

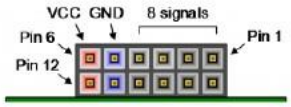


Figure 9.1. Pmod diagram.

	Pmod JA
Pmod Type	Standard
Pin 1	G17
Pin 2	G19
Pin 3	N18
Pin 4	L18
Pin 7	H17
Pin 8	H19
Pin 9	J19
Pin 10	K18

Table 9.1. Cmod A7 Pmod pinout.

Présentation de la carte FPGA CmodA7

XADC

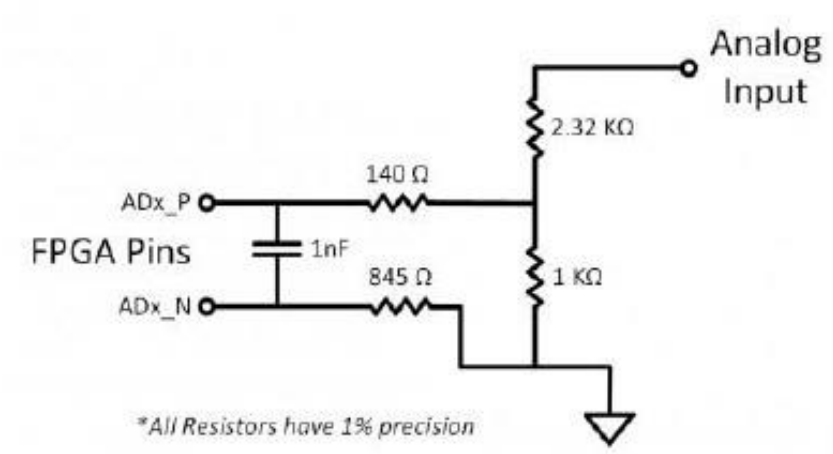


Figure 8.2.1. Analog input circuit.

Le noyau XADC de l'Artix-7 est un convertisseur analogique-numérique 12 bits à double canal, capable de fonctionner à 1 MSPS.

Il s'agit d'une macro matérielle, La plage d'entrée analogique nominale de l'ADC est de 0V à 1V. En mode unipolaire (par défaut), lorsque l'entrée est de 1V, l'entrée analogique de l'ADC génère un code pleine échelle de FFFh (12 bits).

Travail à réalisé

3.1 Partie CAO électronique

Il faudra développer, monter et souder les cartes filles électroniques.
On utilisera le logiciel de CAO électronique **ALTIUM vu en deuxième année et** disponible sur les machines des salles de TP G202.
Vous pouvez également télécharger le logiciel sur votre machine :
https://www.altium.com/solutions/academic-programs/student-licenses.
Licence gratuite pour 6 mois avec une adresse mail accréditée (EDU/Université).
Le schéma à éditer ci-après est fourni dans ce TP et vous permettra de réaliser le PCB (Print Circuit Board) avec les outils d'Altium.
Le circuit imprimé sera développé par un fabricant de circuits imprimés :
SAFE PCB.com.
Les composants seront fournis ainsi que la mise à disposition de poste à souder et de fils à souder. Une liste de matériel (BOM) est jointe au TP.
Les composants passifs et actifs sur les cartes filles seront à monter et à souder.
Il faudra tester électriquement les cartes filles avant d'utiliser la carte FPGA CmodA7.
Le schéma de la page 14 vous sera fourni au format adobe A3 pour plus de lisibilité.

3.2 Partie VHDL Xilinx VIVADO ML

Il faudra adapter les codes VHDL vu lors des TP FPGA 2 pour ce TP.
Ci-joint le lien pour les codes VHDL à implémenter et les infos du projet :
https://github.com/fabzz60/TP_FPGA3
Nous avions utilisé le logiciel Xilinx ISE WEB PACK 14 .7 durant les précédents TP. Ce logiciel étant obsolète nous allons utiliser les derniers outils de développement pour les cibles FPGA XILINX, la suite VIVADO ML. Une démonstration de prise en main de VIVADO sera proposée pour créer les premières étapes du TP côté programmation.

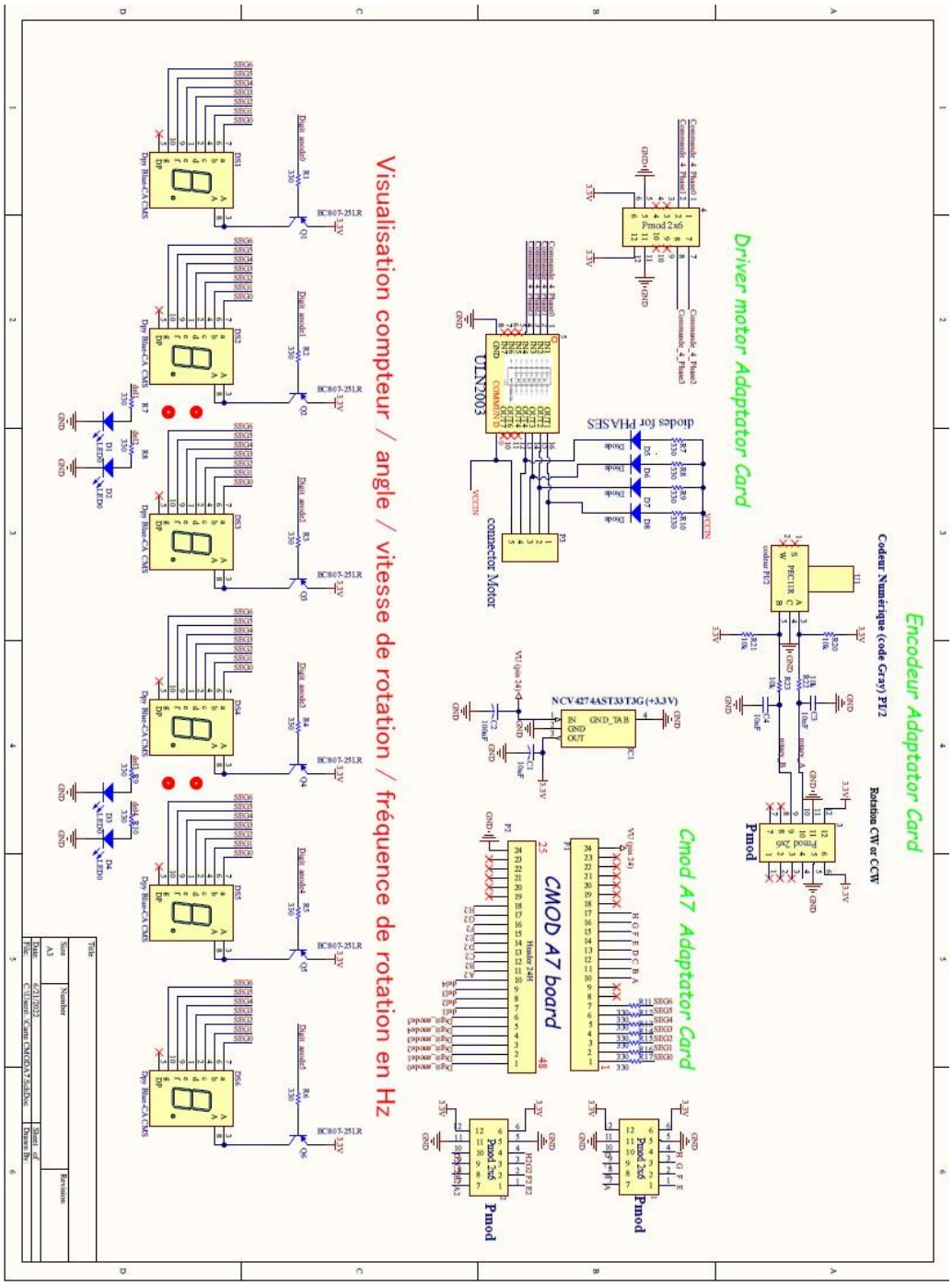
Le lien ci-dessous pour télécharger le logiciel Vivado :
https://www.xilinx.com/products/design-tools/vivado/vivado-ml.html
Créer un compte : login + MDP puis suivre la procédure d'installation.
Fichier à télécharger avant installation : une fois connecté récupérer le fichier Xilinx_Unified_2020.1_0602_1208_Win64.exe
Attention procedure web car downloading file = 41.9GB!!

Supports des TP : https://moodlelms.univ-paris13.fr/my/ ... FPGA2- INSTR2 mais aussi les supports De TP ELN de première année.

Partie CAO

Travail à réalisé

3.3 Schéma équivalent

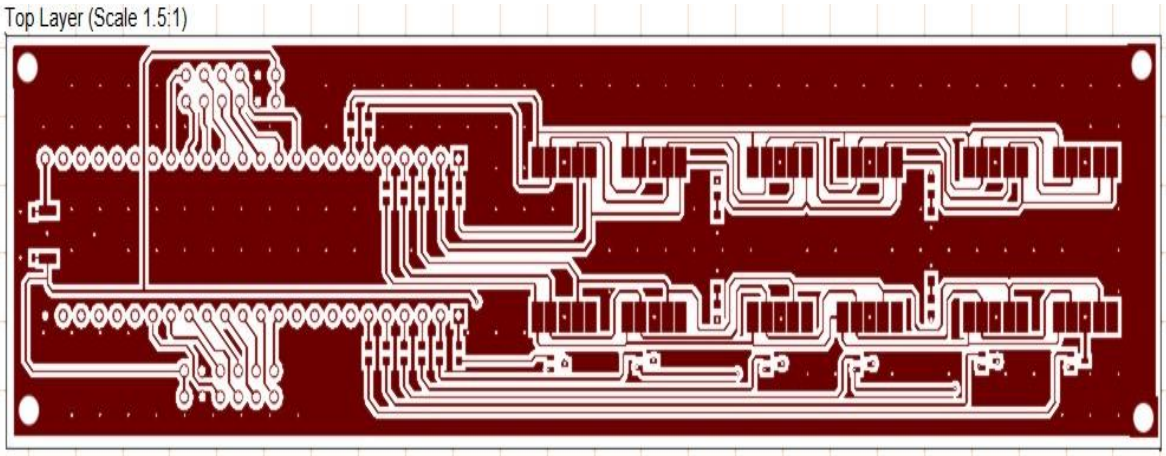


Partie CAO

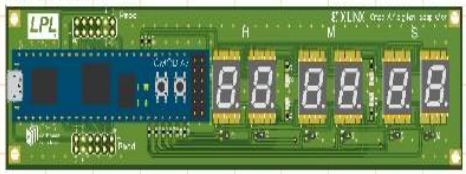
Travail à réalisé

3.4 PCB view

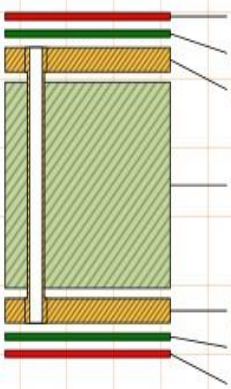
Top Layer (Scale 1.5:1)



Realistic View



Layer Stack Legend



Material	Layer	Thickness	Dielectric Material	Type	Gerber
	Top Overlay			Legend	GTO
Surface Material	Top Solder	0.01mm	Solder Resist	Solder Mask	GTS
Copper	**Top Layer**	**0.04mm**		**Signal**	**GTL**
		0.32mm	*FR-4*	*Dielectric*	
Copper	**Bottom Layer**	**0.04mm**		**Signal**	**GBL**
Surface Material	Bottom Solder	0.01mm	Solder Resist	Solder Mask	GBS
	Bottom Overlay			Legend	GBO

Total thickness: 0.41mm

View PCB assembly

Partie CAO

Travail à réalisé

Liste des composants BOM

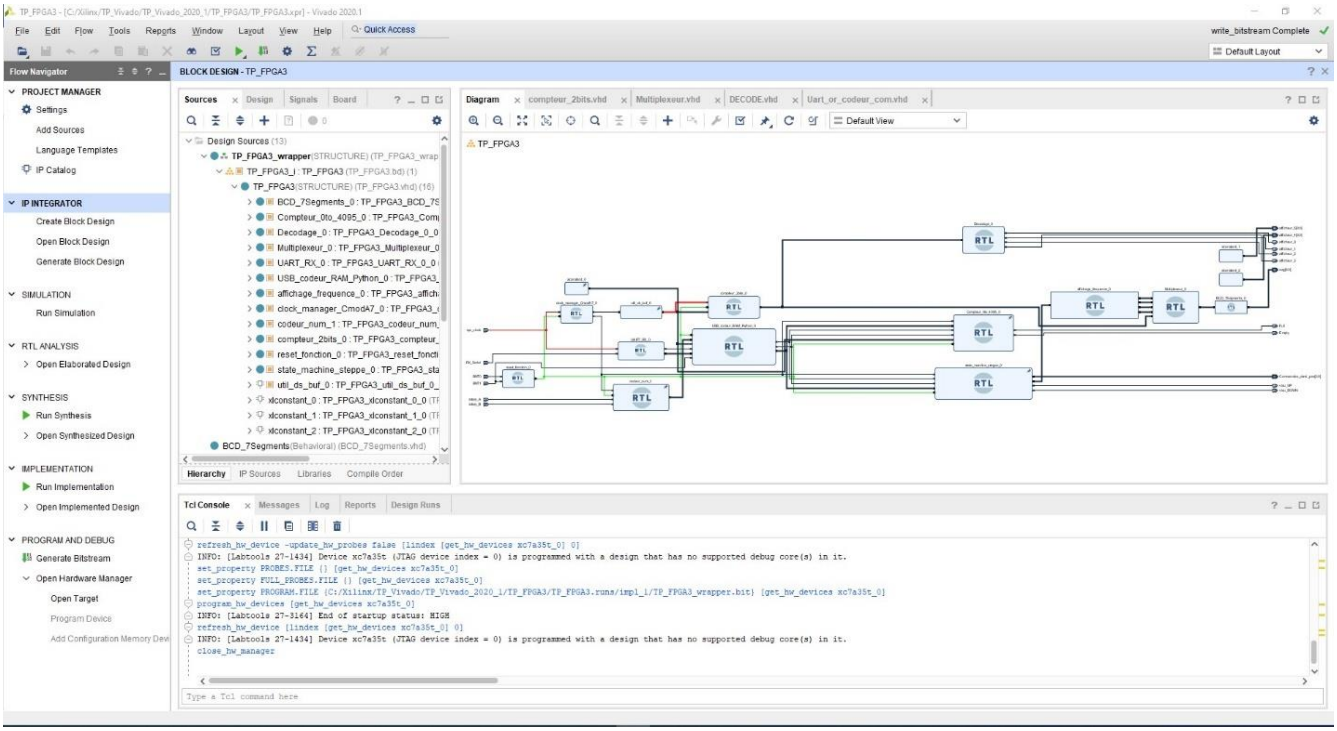
Name	Description	Designator	Quantity	manufacturer
Pmod mâle		3, 4	2	A trouver !
Pmod femelle	Header 2X6	1, 2	2	613012243121
ULN2003	driver motor	5	1	ULQ2003D1013TR
10uF	Capacitor	C1	1	0603ZD106MAT2A
10nF	Capacitor	C2, C3, C4	3	CC0603KRX7R9BB103
100nF	Capacitor	C2	1	MC0603B104K250CT
LED	Typical INFRARED GaAs LED	D1, D2, D3, D4	4	150060RS75000
LED	Typical INFRARED GaAs LED	D5, D6, D7, D8	8	150060RS75000
Dpy Blue-CA CMS	14.2 mm General Purpose Blue 7-Segment Display: CA, R	DS1, DS4, DS5, DS6	4	VDMG10C0
Dpy Blue-CA CMS	14.2 mm General Purpose Blue 7-Segment Display: CA, R	DS2, DS3	2	VDMG10C0
NCV4274AST33T3G (+3.3V)	Integrated Circuit	IC1	1	NCV4274AST33T3G
Header 24H	Header, 24-Pin, Right Angle	P1	1	2212S-24SG-85
Header 24H	Header, 24-Pin, Right Angle	P2	1	2212S-24SG-85
connector Motor	Header, 5-Pin, Right Angle	P3	2	825433-5
BC807-25LR	PNP General Purpose Amplifier	Q1, Q2, Q3, Q4, Q5, Q6	6	BC807-25LR
330	Resistor	R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15, R16, R17	17	ERJ3BWFR100V
10k	Resistor	R20, R21, R22, R23	4	MCMR06X1002FTL
codeur PI/2	12 mm Incremental Encodeur	U1	2	PEC11R-4225F-S0024

Partie VHDL

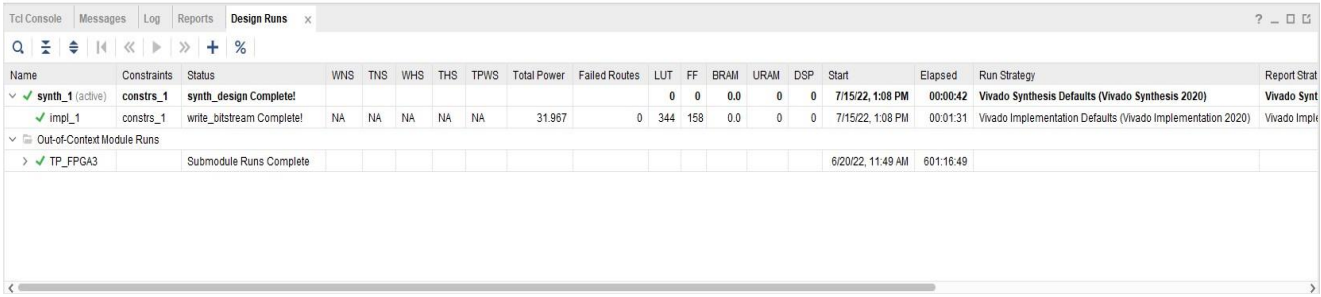
Environnement Vivado

Vue bloc design du code à développer avec les sous-blocs dans la CmodA7 (utiliser le code du TP FPGA 2 et l'adapter au FPGA3 et à Vivado)

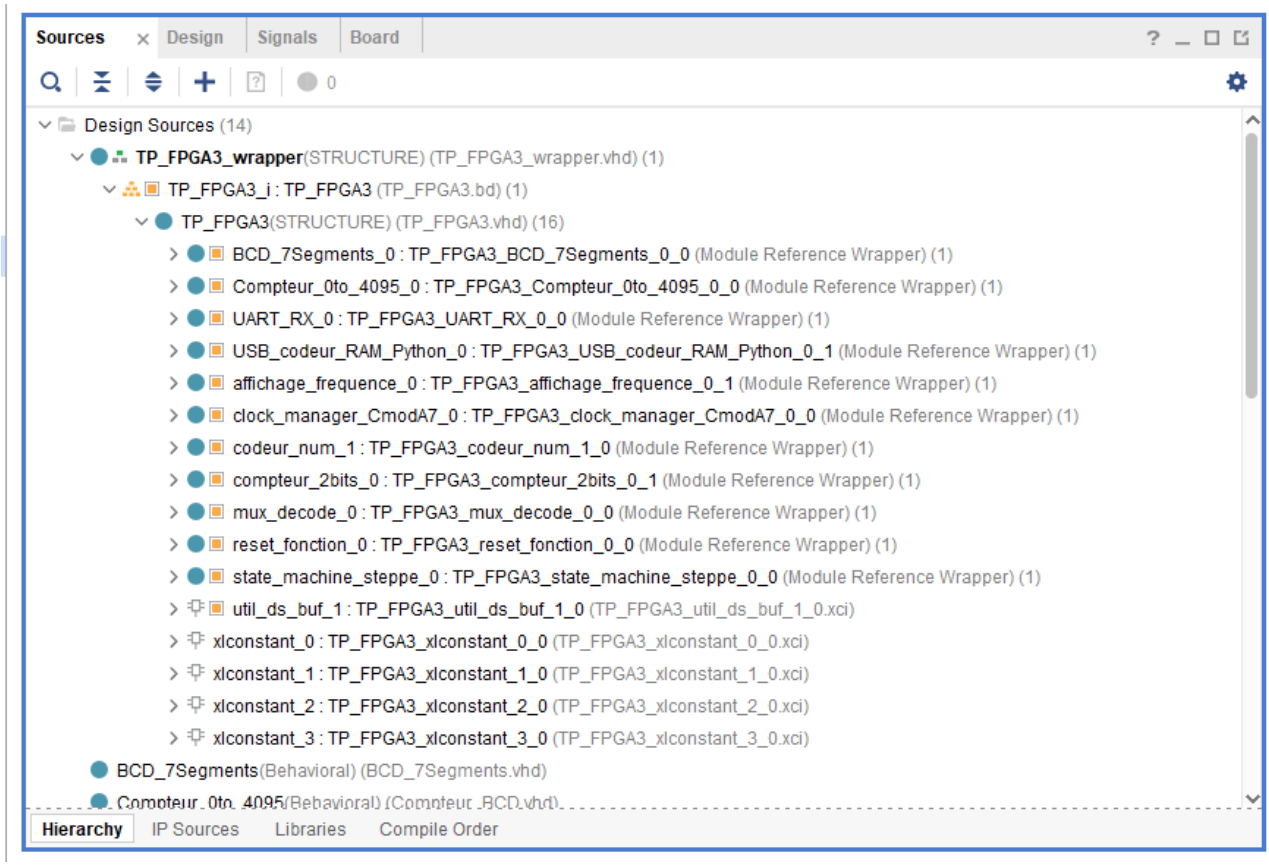
Vue projet



Console et compilation

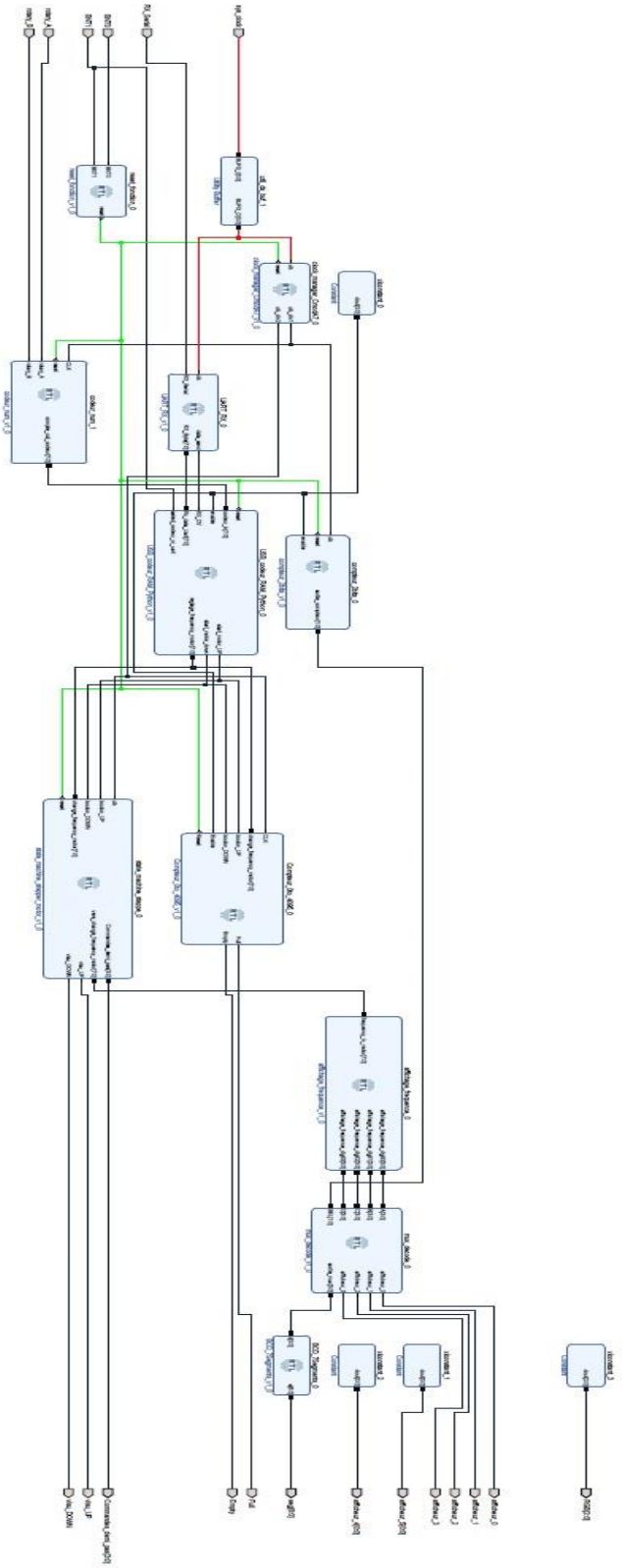


fichiers sources



Partie VHDL

Environnement Vivado



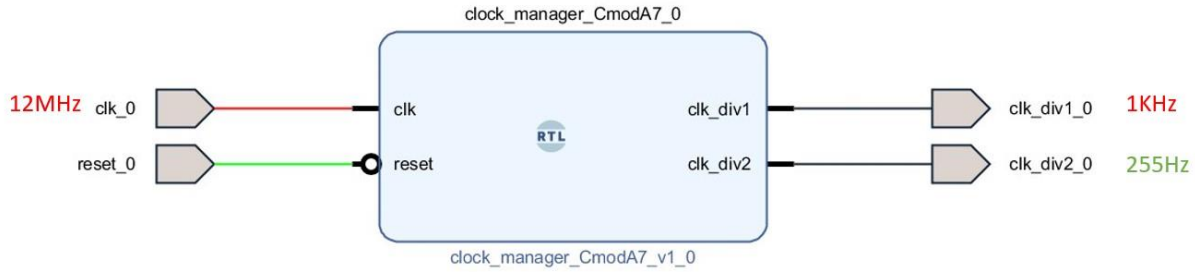
TP moteur pas à pas
contrôle PC (USB-sérial) encodeur numérique pour
commande moteur

Wiotte Fabrice Laboratoire de Physique des Lasers 07/2022

Partie VHDL

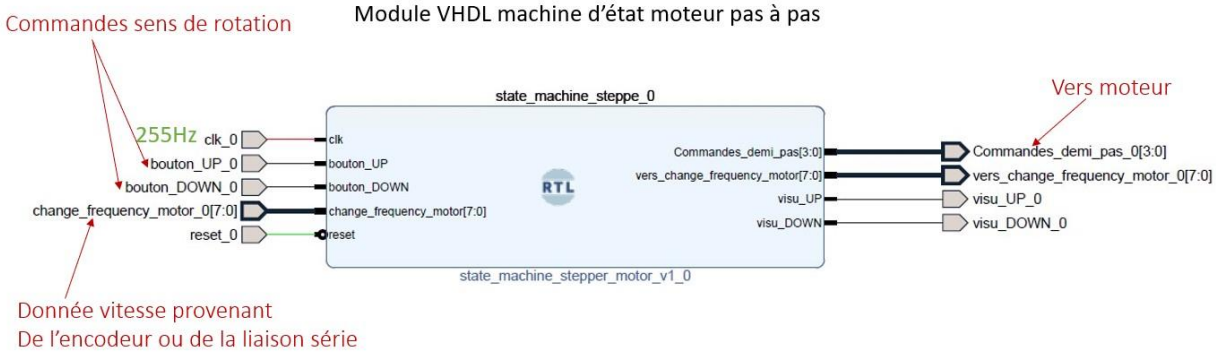
3.5 Environnement Vivado

Module VHDL gestion des horloges



https://github.com/fabzz60/TP_FPGA3

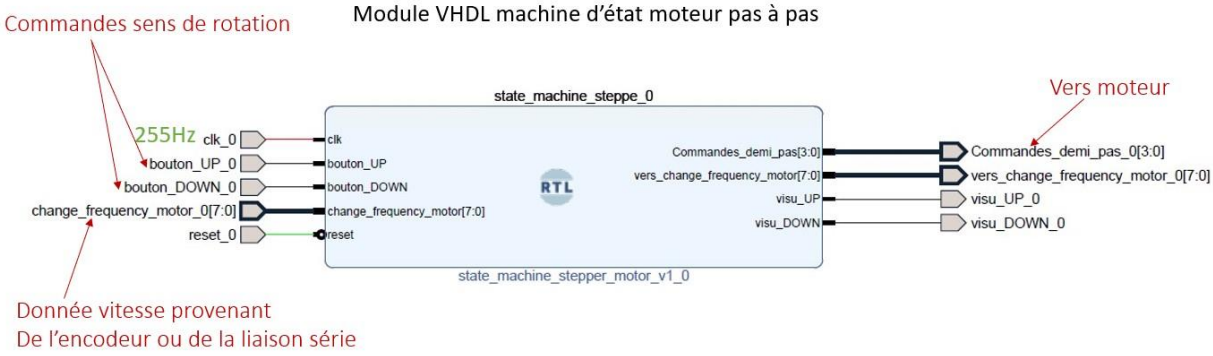
Module VHDL machine d'état moteur pas à pas

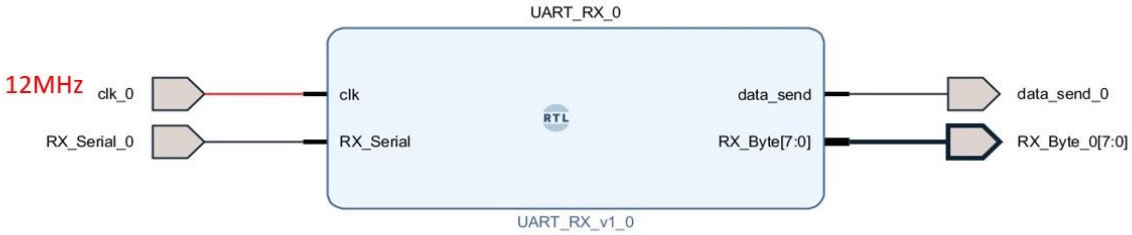


Partie VHDL

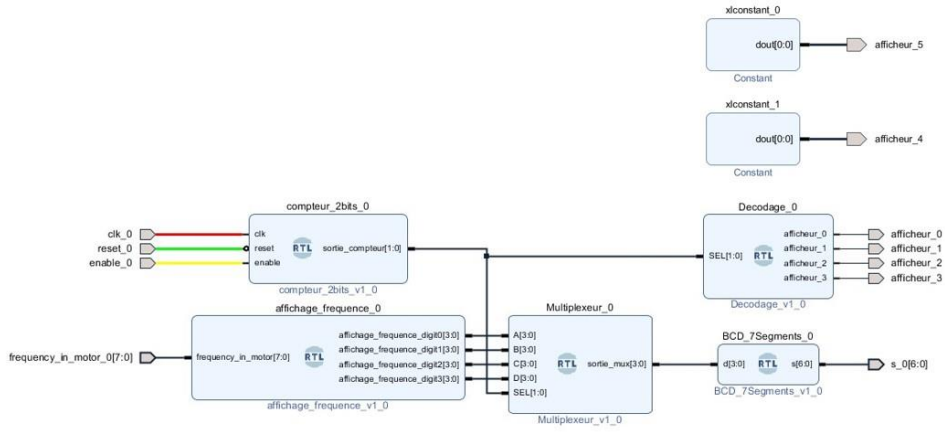
3.5 Environnement Vivado

Module VHDL machine d'état moteur pas à pas





Modules VHDL pour multiplexer et afficher la fréquence du moteur sur LED 7 segments



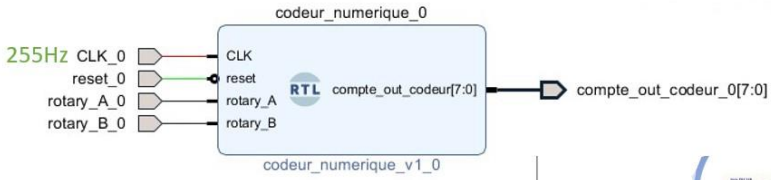
X 6

On n'utilisera que 4 afficheurs pour ce TP

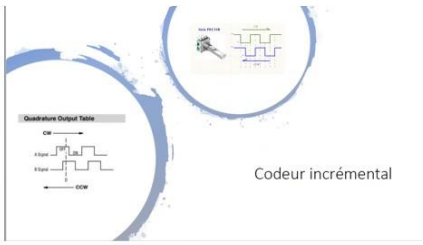
Codage décimal	Codage binaire naturel	Codage Gray ou binaire réfléchi
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100

Module VHDL encodeur numérique

Basé sur Le code de Gray, également appelé binaire réfléchi, permet de ne faire changer qu'un seul bit à la fois quand un nombre est incrémenté ou décrémenté d'une unité.



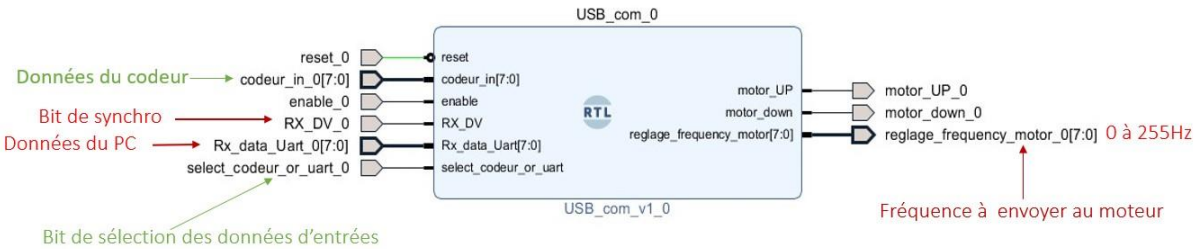
On incrémente localement la fréquence du moteur



13/07/2022

Fabrice Wiotte LPL

Module VHDL gestion des données d'entrées : UART-sérial ou Codeur numérique



3.6 Code Python pour lancer des profils de vitesse linéaire

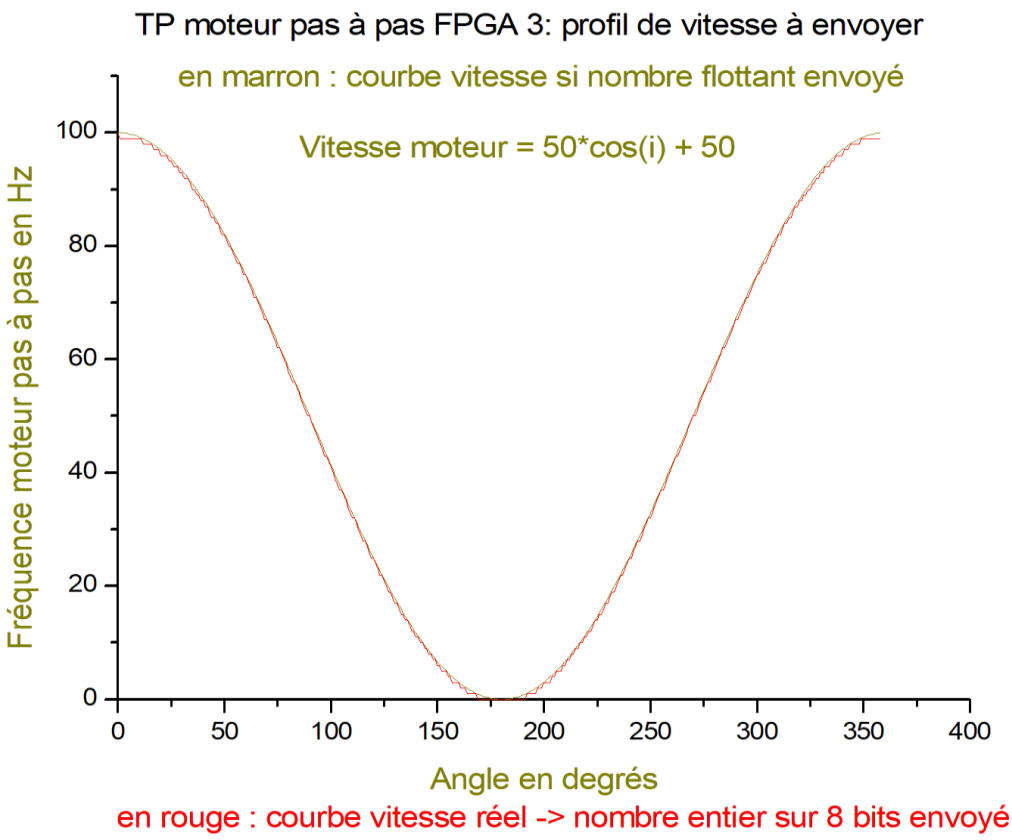
```
# -*- coding: utf-8 -*-
# On importe Tkinter
import serial
import time

ser =serial.Serial(
port ='COM5',
baudrate = 9600,
parity = serial.PARITY_NONE,
stopbits = serial.STOPBITS_ONE,
bytesize = serial.EIGHTBITS,
#timeout = 100
)
def compteur1():  # rampe up
    i = 1
    while i <= 255:

        time.sleep(0.02)
        values = bytearray([i, 2])  # write two byte value frequency and sense of
rotation
        ser.write(values)
        print(i)
        i = i + 1
        if i == 255:
            compteur2()

def compteur2():    # rampe down
    i = 255
    while i >= 0:
        time.sleep(0.05)
        values = bytearray([i, 1]) # write two byte value frequency and sense of
rotation
        ser.write(values)
        print(i)
        i = i - 1
        if i == 0:
            compteur1()
```

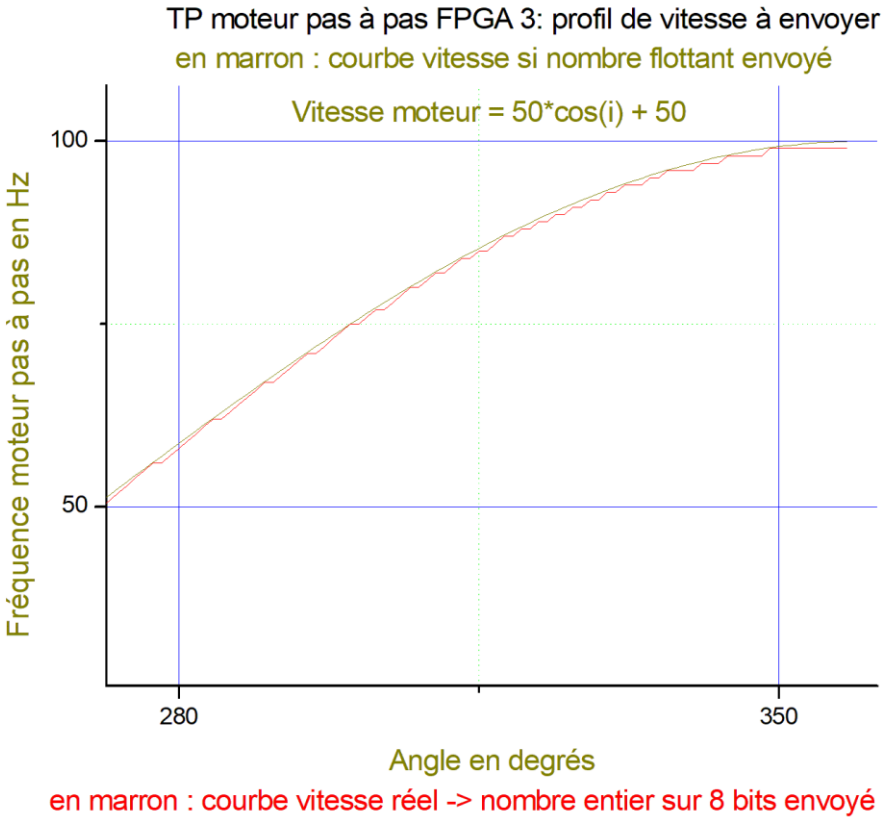
Code Python pour lancer des profils de vitesse linéaire

```
try:
    while True:
        compteur1()

except KeyboardInterrupt:  # ctrl+c
    ser.close()
    print("Fin du programme")
```

Code Python pour lancer des profils de vitesse non linéaire

```
# -*- coding: utf-8 -*-
# On importe Tkinter
import numpy as np
import matplotlib.pyplot as plt
from pylab import *
import serial
import struct
import time
import math  # importation du module
import sys

ser =serial.Serial(
port ='COM11',
baudrate = 9600,
parity = serial.PARITY_NONE,
stopbits = serial.STOPBITS_ONE,
bytesize = serial.EIGHTBITS,
timeout = 100
)
def courbe_cos():

    for z in range(0,360):
        i = math.radians(z)
        w = 50*cos(i) + 50
        time.sleep(0.05)
        #print(w)
        f= int(w)
        print(f)
        values = bytearray([f, 1])  # write two byte value frequency and sense of rotation
        ser.write(values)

try:
    while True:
        courbe_cos()

except KeyboardInterrupt:  # ctrl+c
    print("Fin du programme")
    print("ARRET MOTEUR")
    print("fermeture UART")
    ser.close()
```


TP moteur pas à pas FPGA 3: profil de vitesse à envoyer
en marron : courbe vitesse si nombre flottant envoyé
Vitesse moteur = 50*cos(i) + 50
Fréquence moteur pas à pas en Hz
100
80
60
40
20
0
0
50
100
150
200
250
300
350
400
Angle en degrés
en rouge : courbe vitesse réel -> nombre entier sur 8 bits envoyé


TP moteur pas à pas FPGA 3: profil de vitesse à envoyer
en marron : courbe vitesse si nombre flottant envoyé
Vitesse moteur = 50*cos(i) + 50
Fréquence moteur pas à pas en Hz
100
50
280
350
Angle en degrés
en marron : courbe vitesse réel -> nombre entier sur 8 bits envoyé

Connection à distance de la carte moteur pas à pas
Utilisation d'un Raspberry pi4
En mode SSH (Secure Shell)

3.7 Avant de se connecter en SSH, 2 actions sont nécessaires sur le Raspberry Pi :

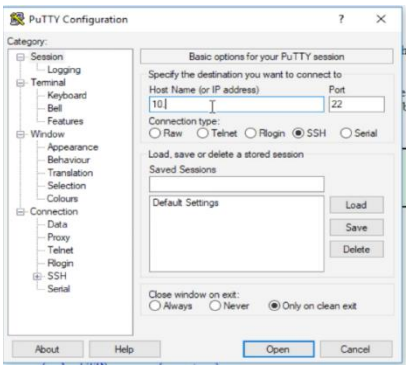
1. Activer le SSH dans le panneau de configuration (Menu démarrer > Préférences > Raspberry Pi Configuration > Interface > Activer SSH)

2. **Si vous avez installé la version complète Raspberry Pi OS (Rasbian)** avec le lien Ci-joint https://www.raspberrypi.org/software/ vous avez accès à l'interface graphique et au menu Paramètres Raspberry Pi, mais on peut aussi sans clavier et sans souris et écran juste avec son PC configurer et installer l'OS. Pour activer SSH sans écran connecté à la Raspberry -> Il vous suffit de créer un fichier nommé *ssh* dans **la partition** *boot* (le fichier n'attend aucune extension). Lors du premier démarrage de la Pi, celle-ci vérifie si le fichier existe et activera le SSH en conséquence -> Pour ceux qui sont habitués à linux et Debian.

3. Récupérer l'adresse IP du Raspberry Pi avec la ligne de commande -> ifconfig ou sur le haut à droite de l'écran sous Raspbian.

4. Pour vous connecter en SSH, le Raspberry Pi et votre ordinateur doivent être connecté au même réseau.**Connexion SSH au Raspberry Pi avec Windows**

Pour se connecter en SSH depuis un ordinateur sous Windows, le plus simple est d'installer "Putty". Vous pouvez le télécharger ici : https://www.putty.org/ Dans "Host Name", entrez l'adresse IP de votre Raspberry Pi et vérifiez que "SSH" est bien sélectionné.

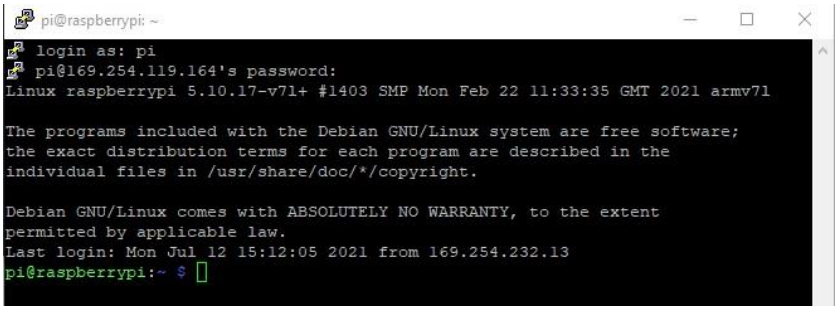


Cliquez ensuite sur "Open" et confirmez la connexion (Security Alert > Confirm). Ensuite, il va vous être demandé le nom de l'utilisateur et le mot de passe. Si vous n'avez pas modifié le nom d'utilisateur, ce sera "pi". Le mot de passe est celui que vous avez défini dans la configuration. Si vous n'en avez pas défini, par défaut, le mot de passe est "raspberry". Appuyez sur "Entrée" et vous voilà connecté au Raspberry Pi.

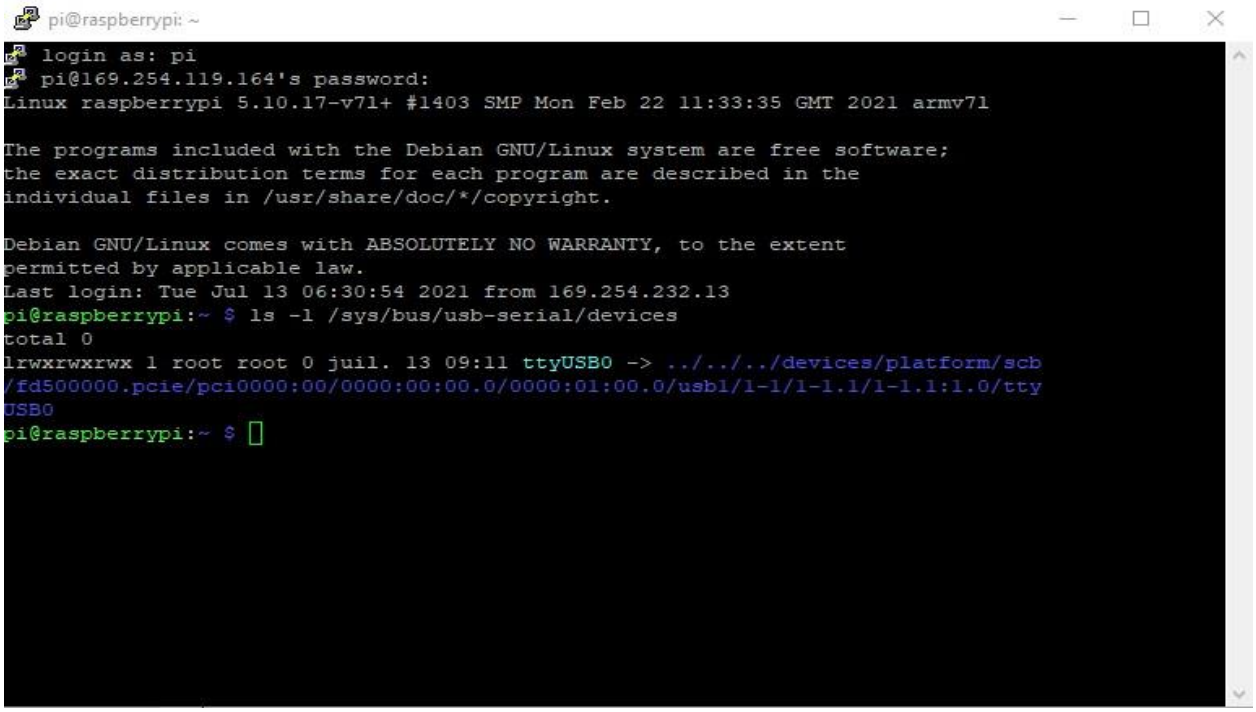
Vous pouvez exécuter des commandes directement depuis cette fenêtre.

Raspberry Pi 4 connectée sur la carte moteur pas à pas

Connection via un pc Windows et avec Putty

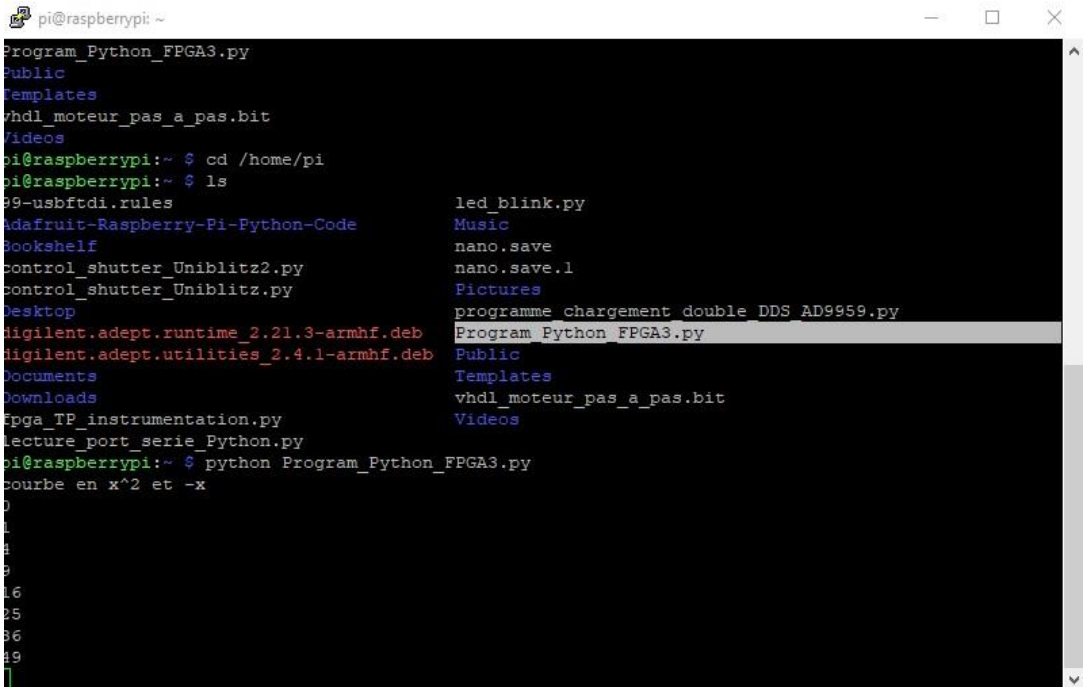


Port USB visible lors de la connection de la carte moteur pas à pas

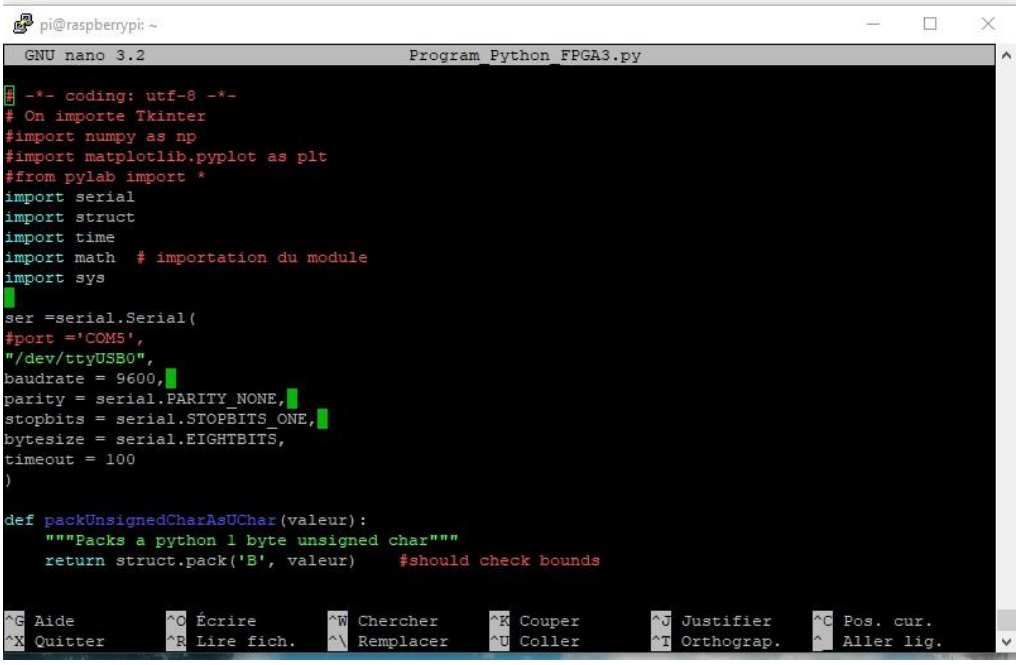


Raspberry Pi 4 connectée sur la carte moteur pas à pas

Lancement du programme

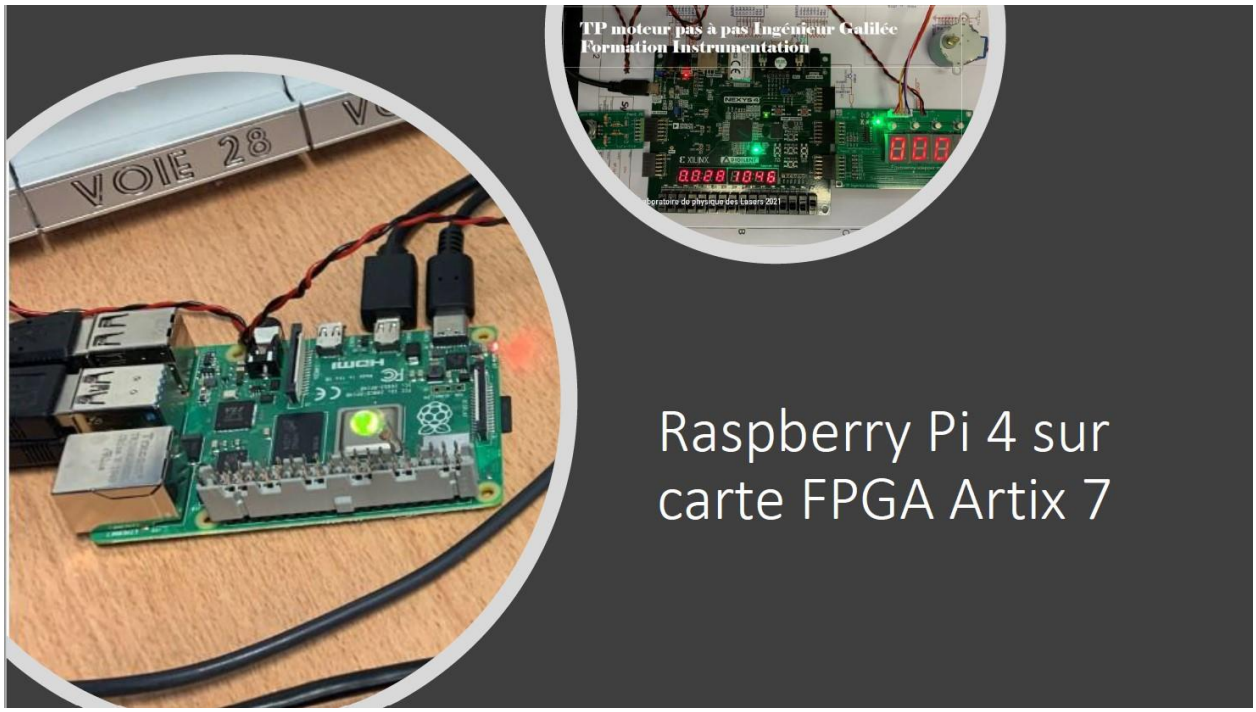


Modification du programme (ligne de commande) avec nano



Raspberry Pi 4 connectée sur la carte moteur pas à pas

TP FPGA instrumentation 2021



4D Systems

4. Module affichage tactile

GEN4-ULCD-43DT

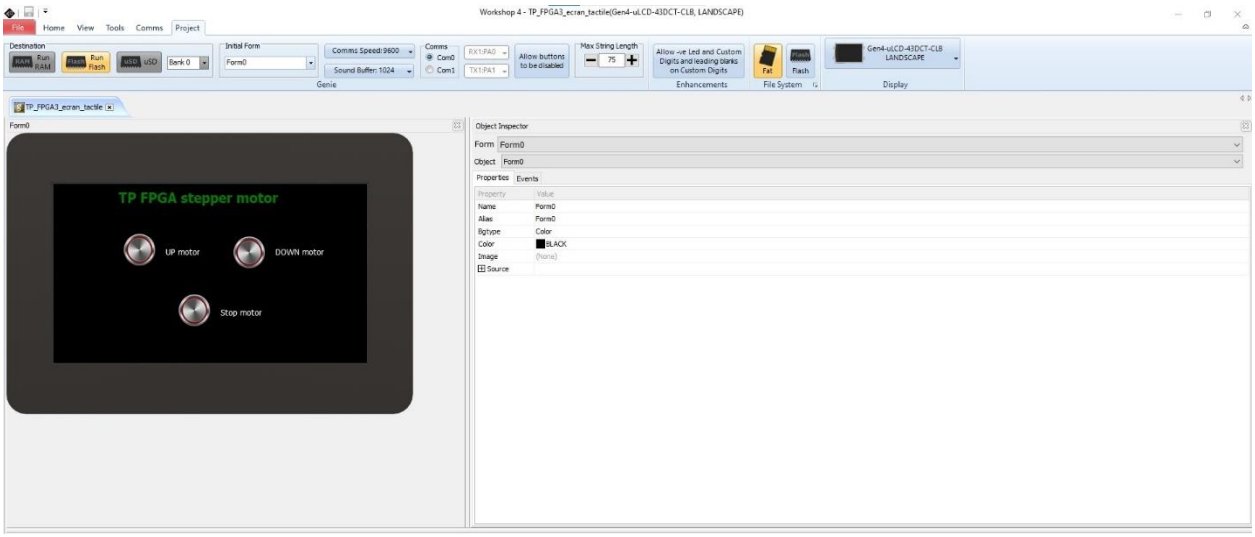
- Module d'affichage intelligent mince de 4,3 pouces, 480 x 272 pixels avec processeur DIABLO16 intégré et écran tactile résistif.
- Convient pour une intégration rapide et facile d'une IHM couleur avec écran tactile dans n'importe quelle application ou produit.
- Kits de démarrage disponibles pour les nouveaux utilisateurs.
- Bundles avec cartes d'interface matérielle pour Arduino et Raspberry Pi également disponibles.
- Voir la description du produit ci-dessous pour plus de détails.

https://4dsystems.com.au/
https://4dsystems.com.au/products/featured-products/gen4-ulcd-43dt
Logiciel: **Workshop 4 and VISI-Genie**
Getting start Workshop4 and VISI-Genie:
https://4dsystems.com.au/mwdownloads/download/link/id/26/

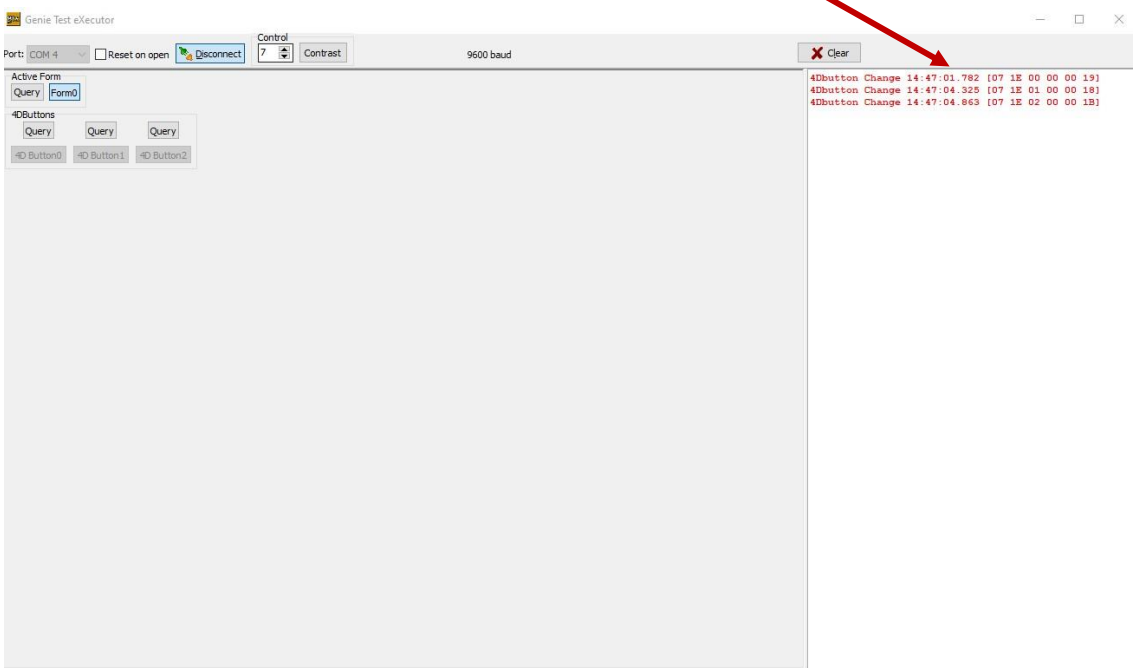
4D Systems

Module affichage tactile

Gen4-uLCD-43DT-CLB avec Workshop 4

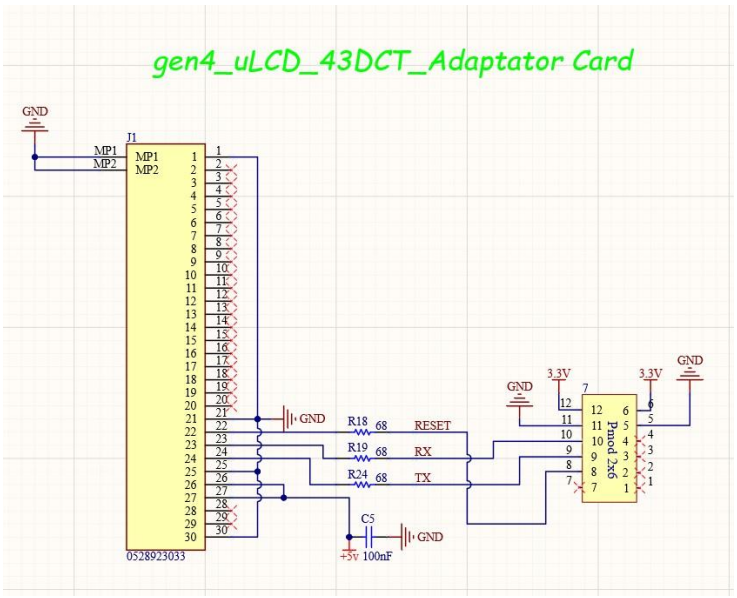


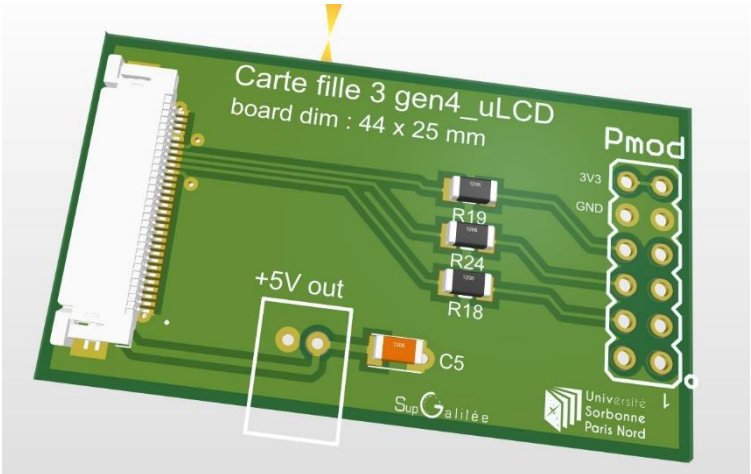
Communication RX-TX : trame de données (6 Octets)



4D Systems

Carte fille écran tactile pour Pmod
Sur carte FPGA CmodA7





4.3 Vue hiérarchique sous Vivado avec interface écran tactile