## 0 Présentation de la guidance.

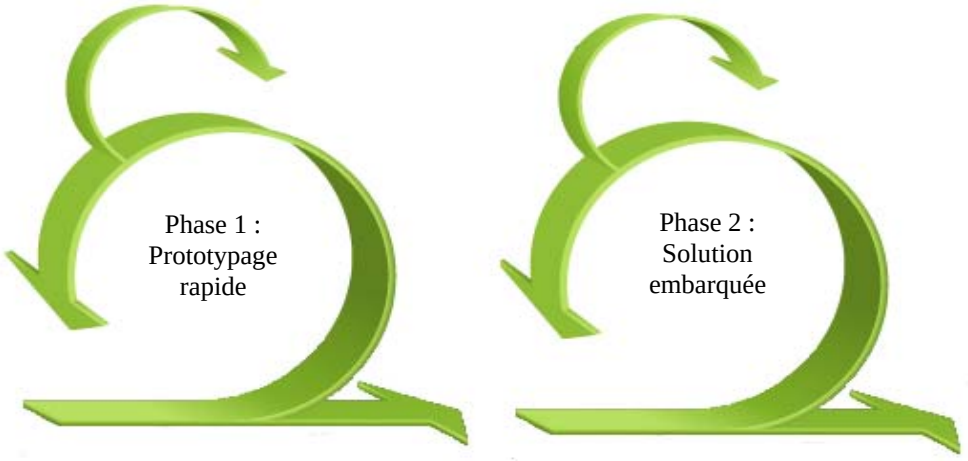
Pré-requis : ⇒ Avoir suivi les TP1, TP2, TP4 et un mini projet sous Altium.
⇒ Connaissance du langage C ANSI.

Durée estimée : ⇒ 2 heures.

Objectif : ⇒ Cette guidance à pour finalité de rendre autonome les étudiants lors de projets mettant en œuvre un FPGA en STS SN.

⇒ Phase 1: **Prototypage rapide**, développer et valider une solution **SOC** dans un **FPGA** à l'aide de la **Nanoboard 3000AL** sous **Altium**.

⇒ Phase 2 : Transférer la solution testée par le prototypage rapide dans un **FPGA** sur une carte fille **DE0Nano**. La carte fille étant implantée sur une carte mère développée par l'étudiant.

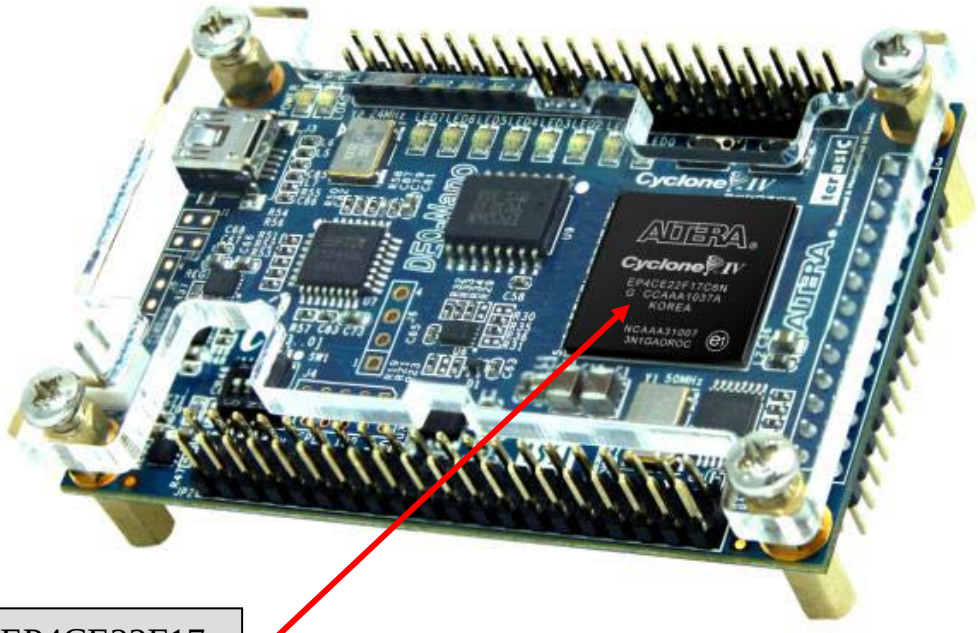



Organisation du projet de fin d'étude en deux temps :

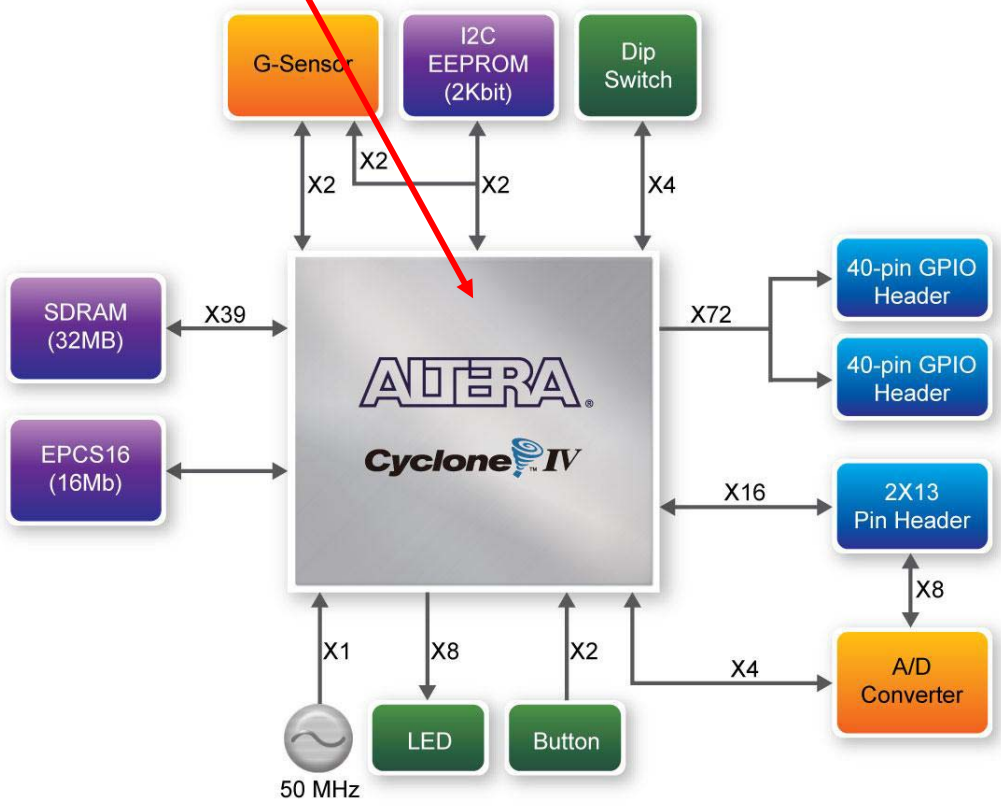
| | Phase 1 | Phase 2 |
|---|---|---|
| Prototypage rapide sur la Nanoboard 3000AL | | |
| Transfert de la solution du prototypage rapide vers la carte fille DE0nano. | | |

Remarque : Dans le cadre d'un projet commun EC-IR le développement logiciel pourra se limiter au test fonctionnel des solutions matérielles développées pour l'option EC.
Le développement logiciel sera complet pour l'option IR.

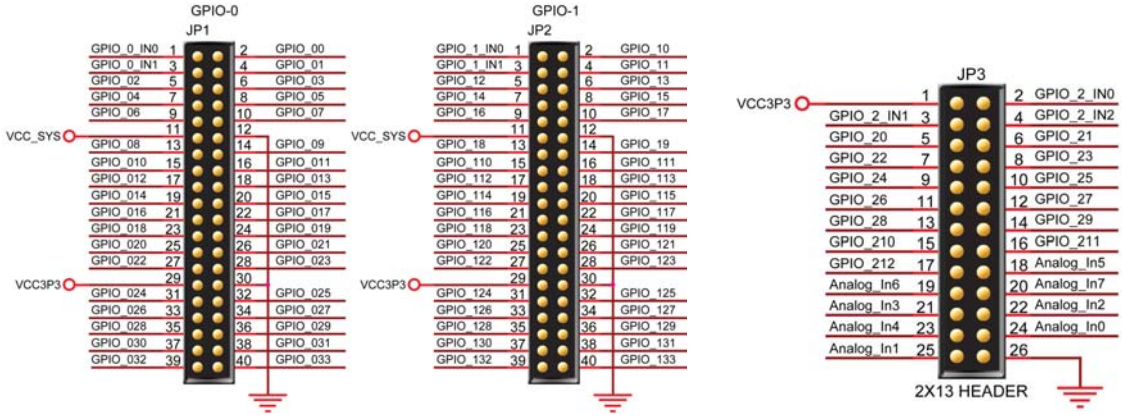
## Présentation matérielle de la carte DE0 Nano



| FPGA : EP4CE22F17 |
|---|
| Cellules logiques : 47<br>Boîtier : BGA<br>Broches : 256<br>Alimentation : 1.27V |



La carte DE0 Nano est équipée de trois connecteurs J1, J2 J3 qui permettent de ressortir les signaux d'entrée-sortie du FPGA.




| CONNECTEUR J1 | | |
|---|---|---|
| N° de broche J1 | Nom | Broche FPGA |
| 1 | GPIO_0_IN0 | A8 |
| 2 | GPIO_00 | D3 |
| 3 | GPIO_0_IN1 | B8 |
| 4 | GPIO_01 | C3 |
| 5 | GPIO_02 | A2 |
| 6 | GPIO_03 | A3 |
| 7 | GPIO_04 | B3 |
| 8 | GPIO_05 | B4 |
| 9 | GPIO_06 | A4 |
| 10 | GPIO_07 | B5 |
| 13 | GPIO_08 | A5 |
| 14 | GPIO_09 | D5 |
| 15 | GPIO_010 | B6 |
| 16 | GPIO_011 | A6 |
| 17 | GPIO_012 | D7 |
| 18 | GPIO_013 | D6 |
| 19 | GPIO_014 | A7 |
| 20 | GPIO_015 | C6 |
| 21 | GPIO_016 | C8 |
| 22 | GPIO_017 | E6 |
| 23 | GPIO_018 | E7 |
| 24 | GPIO_019 | D8 |
| 25 | GPIO_020 | E8 |
| 26 | GPIO_021 | F8 |
| 27 | GPIO_022 | F9 |
| 28 | GPIO_023 | E9 |
| 31 | GPIO_024 | C9 |
| 32 | GPIO_025 | D9 |
| 33 | GPIO_026 | E11 |
| 34 | GPIO_027 | E10 |
| 35 | GPIO_028 | C11 |
| 36 | GPIO_029 | B11 |
| 37 | GPIO_030 | A12 |
| 38 | GPIO_031 | D11 |
| 39 | GPIO_032 | D12 |
| 40 | GPIO_033 | B12 |

| CONNECTEUR J2 | | |
|---|---|---|
| N° de broche J2 | Nom | Broche FPGA |
| 1 | GPIO_1_IN0 | T9 |
| 2 | GPIO_10 | R13 |
| 3 | GPIO_1_IN1 | R9 |
| 4 | GPIO_11 | T15 |
| 5 | GPIO_12 | T14 |
| 6 | GPIO_13 | T13 |
| 7 | GPIO_14 | R13 |
| 8 | GPIO_15 | T12 |
| 9 | GPIO_16 | R12 |
| 10 | GPIO_17 | T11 |
| 13 | GPIO_18 | T10 |
| 14 | GPIO_19 | R11 |
| 15 | GPIO_110 | P11 |
| 16 | GPIO_111 | R10 |
| 17 | GPIO_112 | N12 |
| 18 | GPIO_113 | P9 |
| 19 | GPIO_114 | N9 |
| 20 | GPIO_115 | N11 |
| 21 | GPIO_116 | L16 |
| 22 | GPIO_117 | K16 |
| 23 | GPIO_118 | R16 |
| 24 | GPIO_119 | L15 |
| 25 | GPIO_120 | P15 |
| 26 | GPIO_121 | P16 |
| 27 | GPIO_122 | R14 |
| 28 | GPIO_123 | N16 |
| 31 | GPIO_124 | N15 |
| 32 | GPIO_125 | P14 |
| 33 | GPIO_126 | L14 |
| 34 | GPIO_127 | N14 |
| 35 | GPIO_128 | M10 |
| 36 | GPIO_129 | L13 |
| 37 | GPIO_130 | J16 |
| 38 | GPIO_131 | K15 |
| 39 | GPIO_132 | J13 |
| 40 | GPIO_133 | J14 |

| CONNECTEUR J3 | | |
|---|---|---|
| N° de broche J3 | Nom | Broche FPGA |
| 2 | GPIO_ 2IN0 | E15 |
| 3 | GPIO_ 2IN1 | E16 |
| 4 | GPIO_ 2IN2 | M16 |
| 5 | GPIO_20 | A14 |
| 6 | GPIO_21 | B16 |
| 7 | GPIO_22 | C14 |
| 8 | GPIO_23 | C16 |
| 9 | GPIO_24 | C15 |
| 10 | GPIO_25 | D16 |
| 11 | GPIO_26 | D15 |
| 12 | GPIO_27 | D14 |
| 13 | GPIO_28 | F15 |
| 14 | GPIO_29 | F16 |
| 15 | GPIO_210 | F14 |
| 16 | GPIO_211 | G16 |
| 17 | GPIO_212 | G15 |

**Mode opératoire : déroulement du projet.**

**Partie 1 : Prototypage rapide sous la Nanoboard 3000AL**

1.1 Créer un nouveau projet dans un nouveau dossier

1.2 Editer le fichier open bus

1.3 Créer le schéma top

1.4 Créer les fichiers contraintes pour la Nanoboard 3000AL

1.5 mise en place des API : SWPLATFORM

1.6 Mettre en place et écrire le projet embarqué

1.7 Allocation d'espace pour la pile et pour les registres

1.8 Compiler, Synthétiser, Build, Programmer la Nanoboard

1.9 Tester par mesure votre programme

Procédure présentée durant le TP4

**Partie 2 : Implantation sur la carte DE0 Nano**

2.1 créer un nouveau répertoire pour un nouveau projet

2.2 Importer dans votre fichier OPEN BUS sous le nouveau répertoire

2.3 Placer le fichier contrainte DE0_Nano .Constraint dans le projet

2.4 Recréer un nouveau schéma à partir de l'OPENBUS.

2.5 Créer un projet embarqué et l'associer au projet hard

2.6 Créer un fichier swplatform

2.7 Associer le fichier *.c développer précédemment pour la Nanoboard

2.8 Réserver l'espace mémoire de la pile et l'espace de la mémoire dynamique

2.9 choisir le FPGA à programmer

2.10 Compiler, synthétiser, build

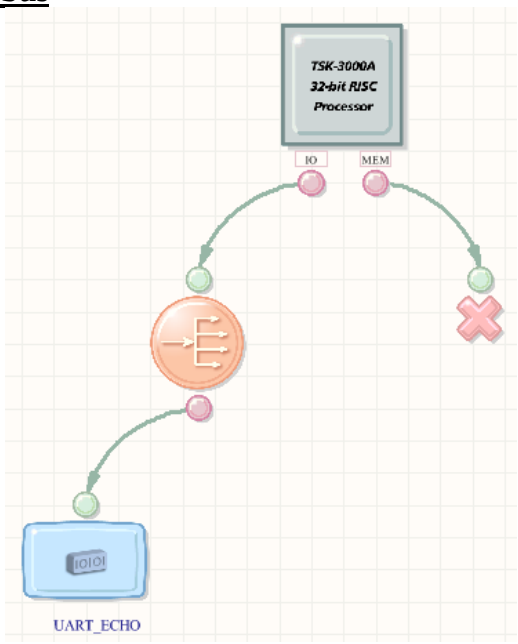
2.11 Ouvrir Quartus, A partir des fichiers *.hex et *.sof construction d'un fichier *.jic

2.12 Programmer la DE0 depuis Quartus

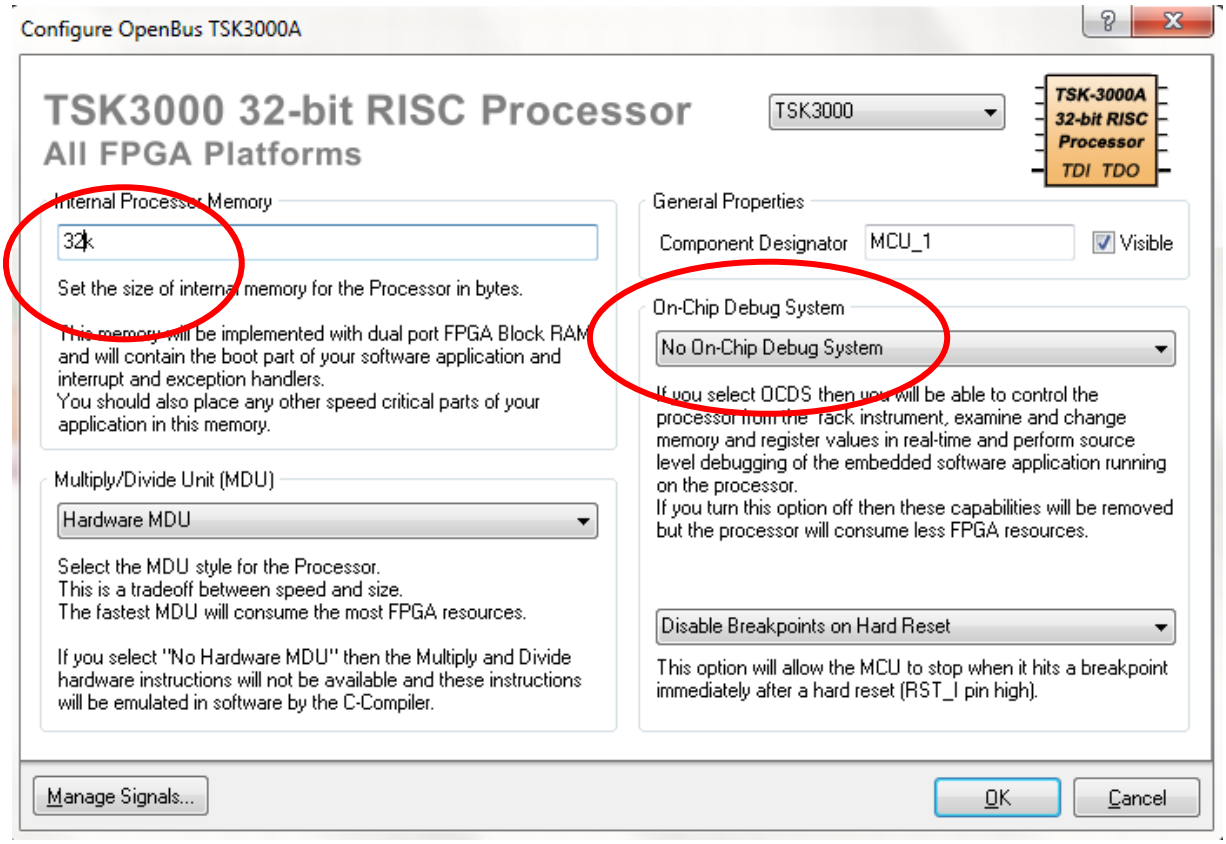
# **Partie 1** : Prototypage rapide sur la Nanoboard 3000AL sous ALTIUM

## 1.1 créer un nouveau projet dans un nouveau dossier

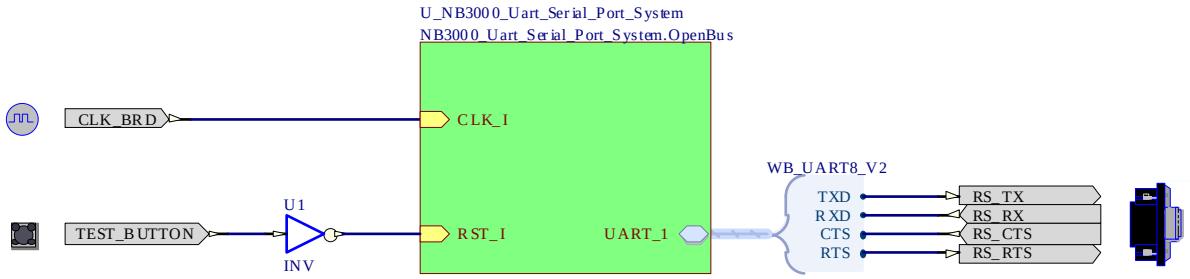
## 1.2 Editer le fichier open bus



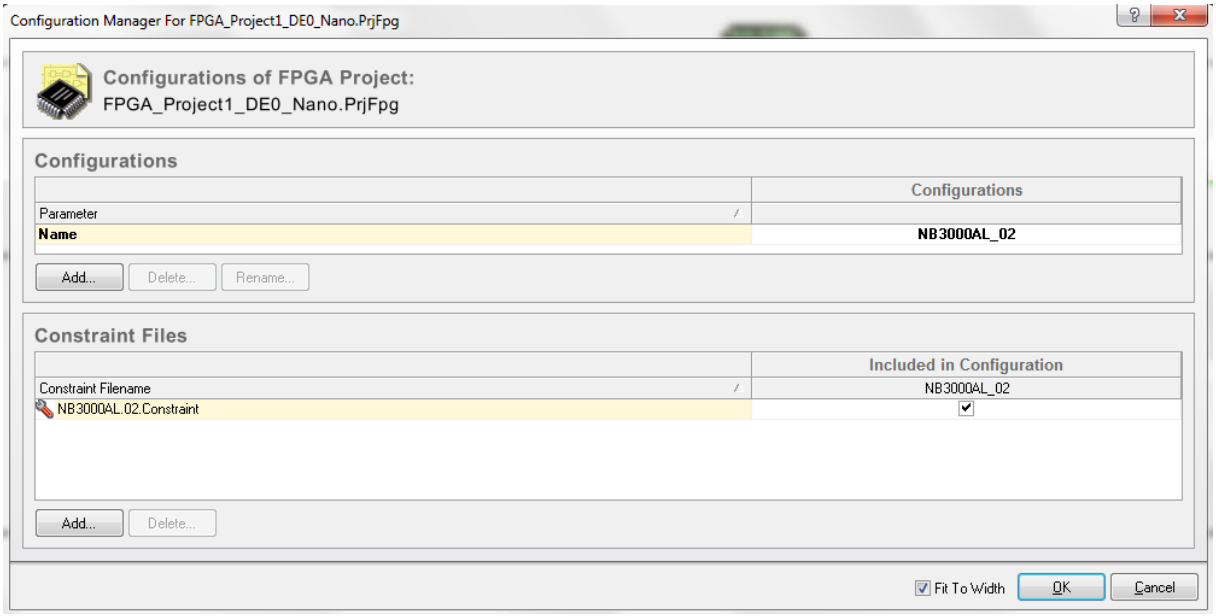

⇒ Paramétrez le processeur comme ci-dessous :



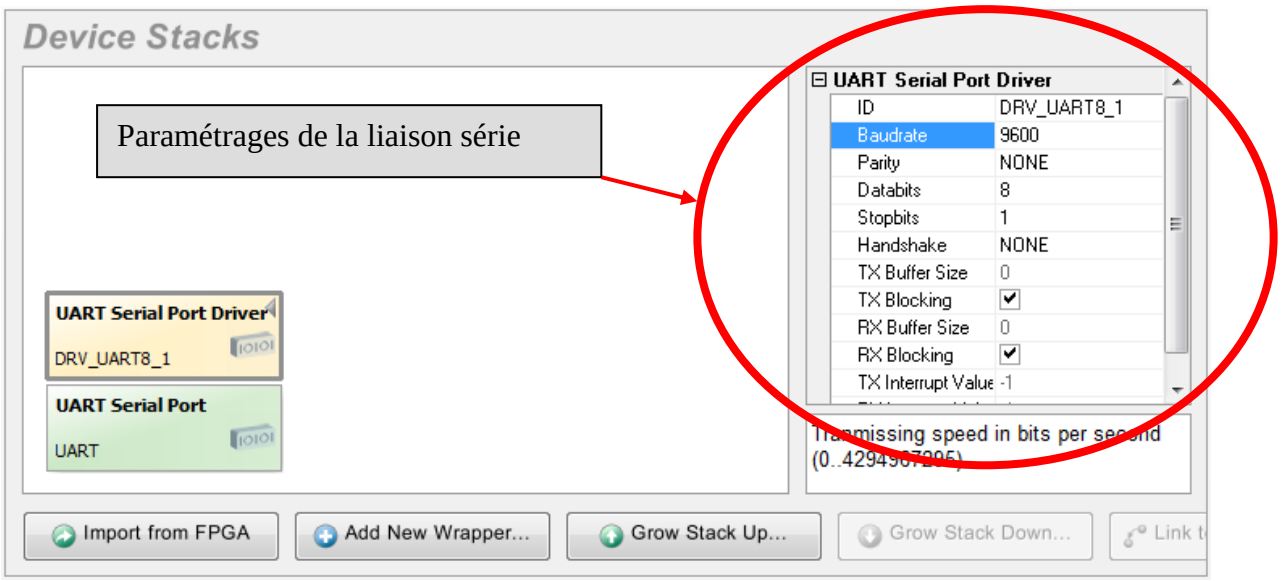
## 1.3 créer le schéma top



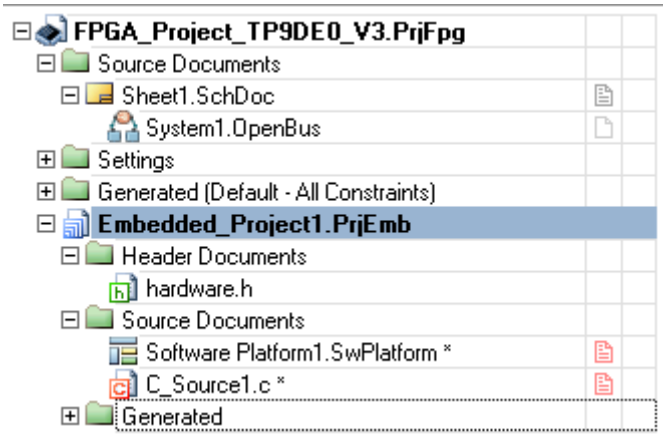

## 1.4 Associer les fichiers contraintes pour la Nanoboard 3000AL




## 1.5 mise en place des API : SWPLATFORM



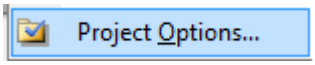
## 1.6 Mettre en place et écrire le  projet embarqué



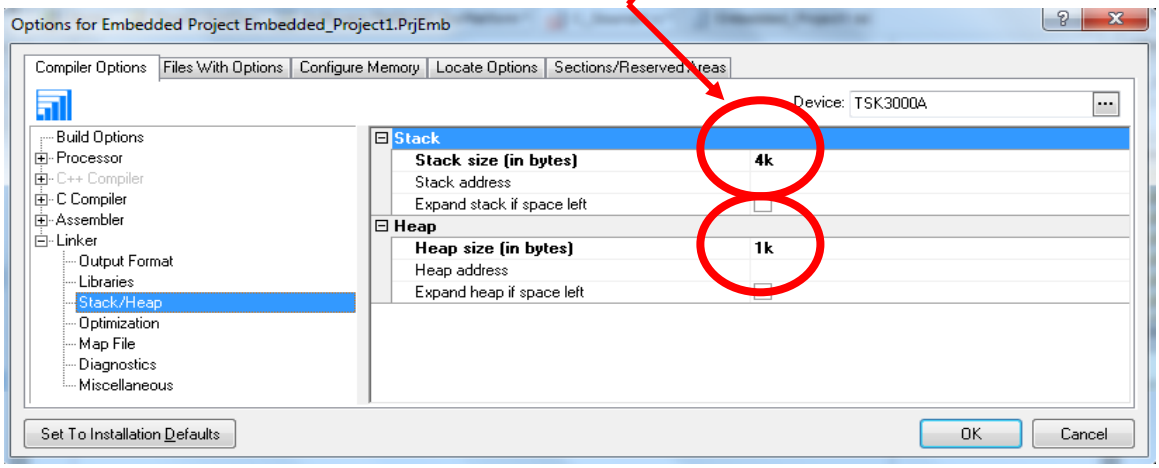

```
// programme emission trame AT

#include "swplatform.h"

void main(void)
{
    int tab_AT[20]={'A','T','+','C','G','S','N'};  //Commande AT N° de série
    int i;
    swplatform_init_stacks();

    for (;;)
    {
     for (i=0;i<7;i++)uart8_putchar( drv_uart8_1, tab_AT[i] );
     delay_ms (100);
    }
}
```

## 1.7 Allocation d'espace pour la pile et pour les registres

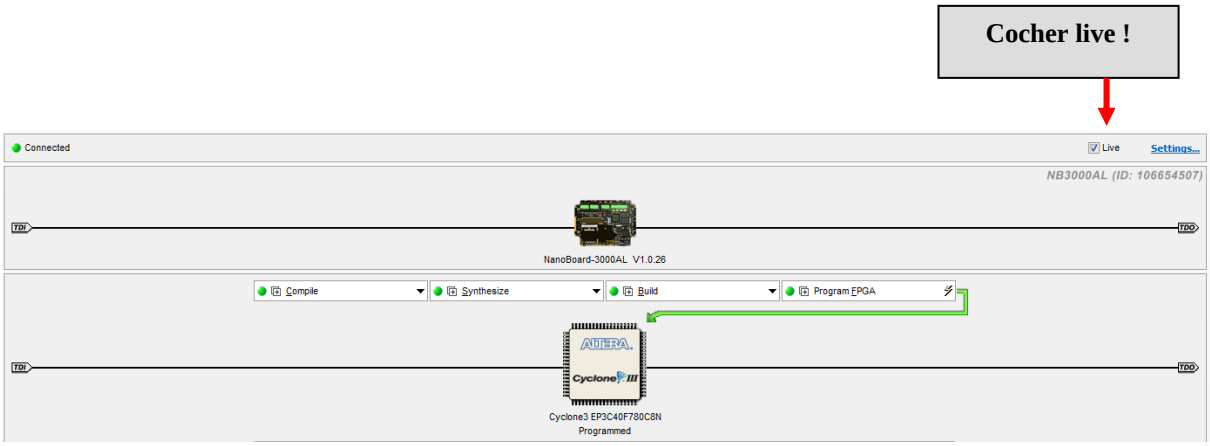

Cliquer bouton de droite sur **Embedded Project**




La partie hard ne comporte pas de mémoire externe. Il faut donc dédier une partie de la mémoire interne du processeur :
⇒ à la pile
⇒ à la gestion de la mémoire dynamique ( le tas )

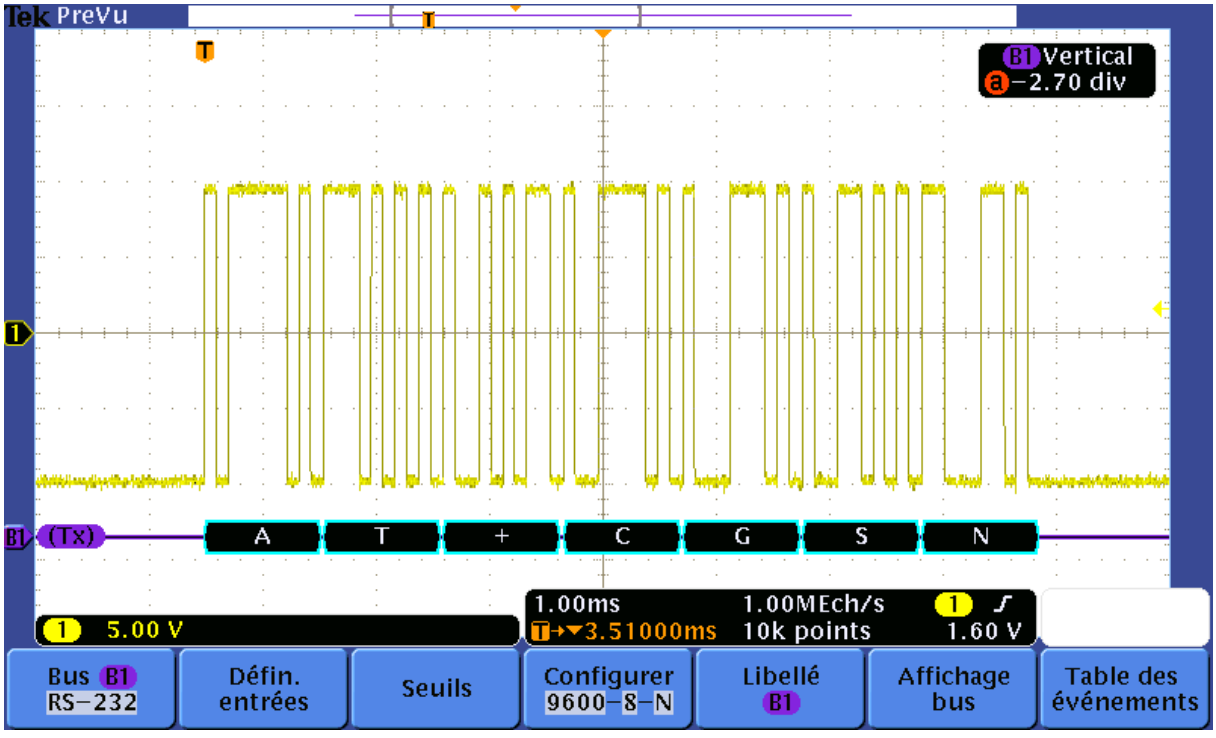


## 1.8 Compiler, Synthétiser, Build, Programmer la Nanoboard




## 1.9 Tester par mesure votre programme

⇒ Réponse attendue :



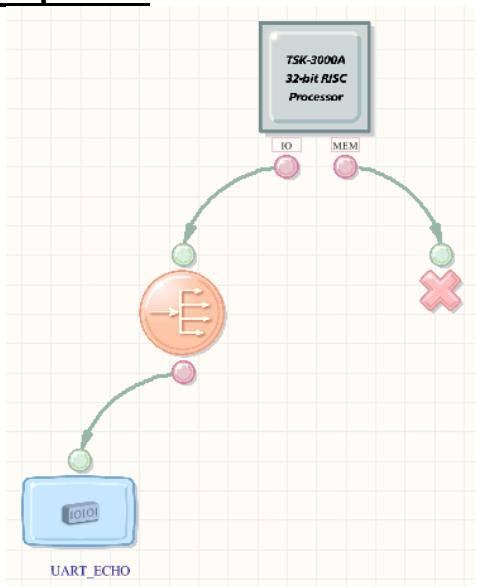
# **Partie 2** : Implantation de la solution sur la carte fille DE0 Nano

## 2.1 créer un nouveau répertoire et un nouveau projet : FPGA_Prject_DE0nano.prjFpg

**⇒ ERREUR A NE PAS COMMETTRE : PLACER LES DEUX PROJETS DANS LE MEME REPERTOIRE.**

## 2.2 Importer votre fichier OPEN BUS sous le nouveau répertoire

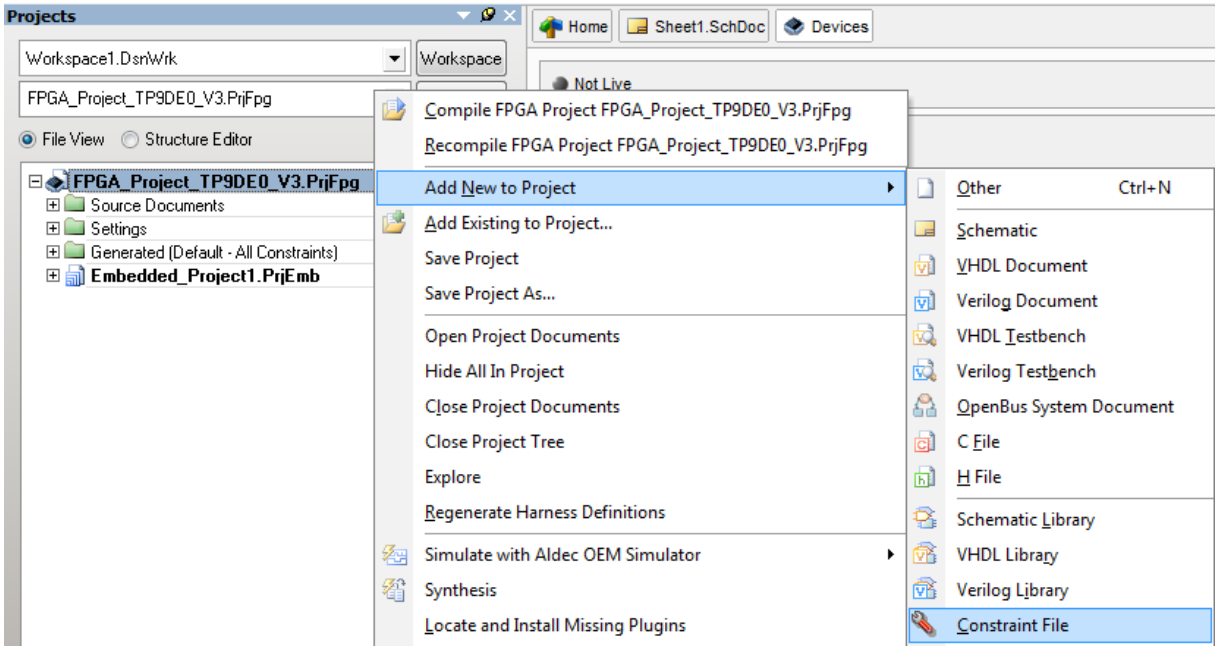
Le schéma OPENBUS reste le même



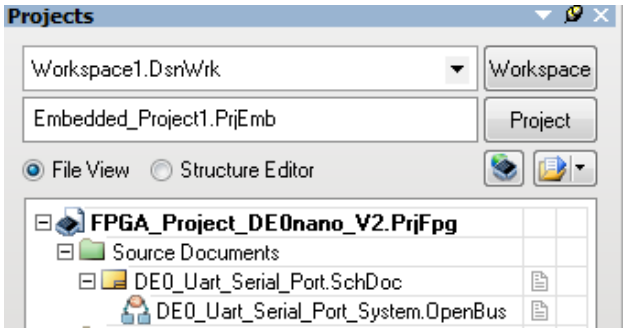

## 2.3 Placer le fichier contrainte DE0_Nano .Constraint dans le projet

⇒ Recopier ce dossier DE0_Nano .Constraint dans votre répertoire de travail courant.

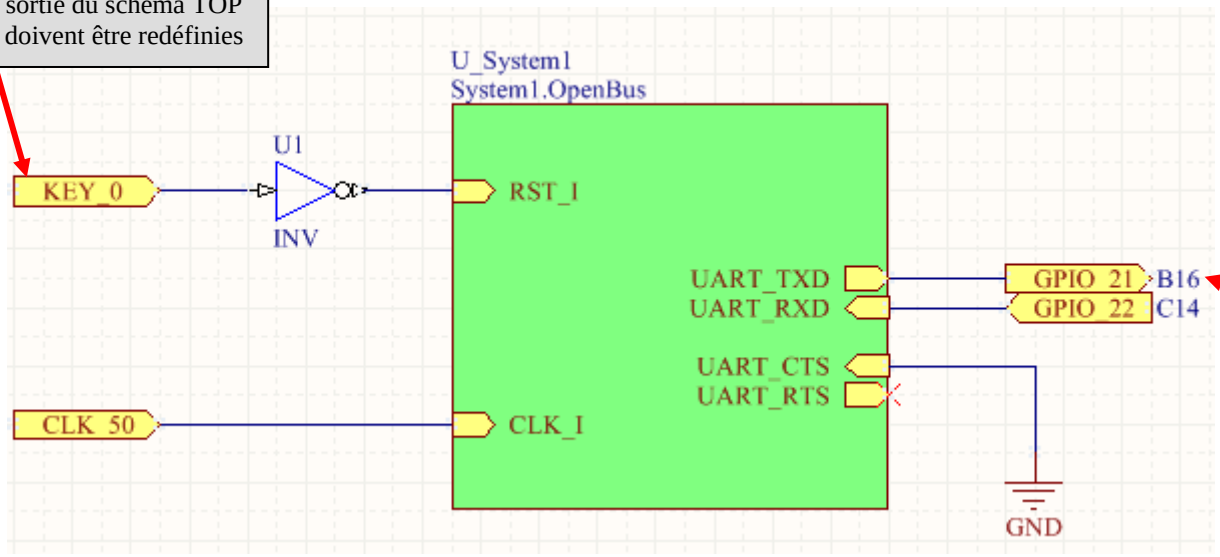
⇒ Cliquer le bouton de droite sur le projet ⇒ Add New to Project



## 2.4 Recréer un nouveau schéma à partir de l'OPENBUS.




Les broches d'entrées sortie du schéma TOP doivent être redéfinies




⇒ Associer les broches du schémas structurel aux noms définis dans le fichier contrainte.

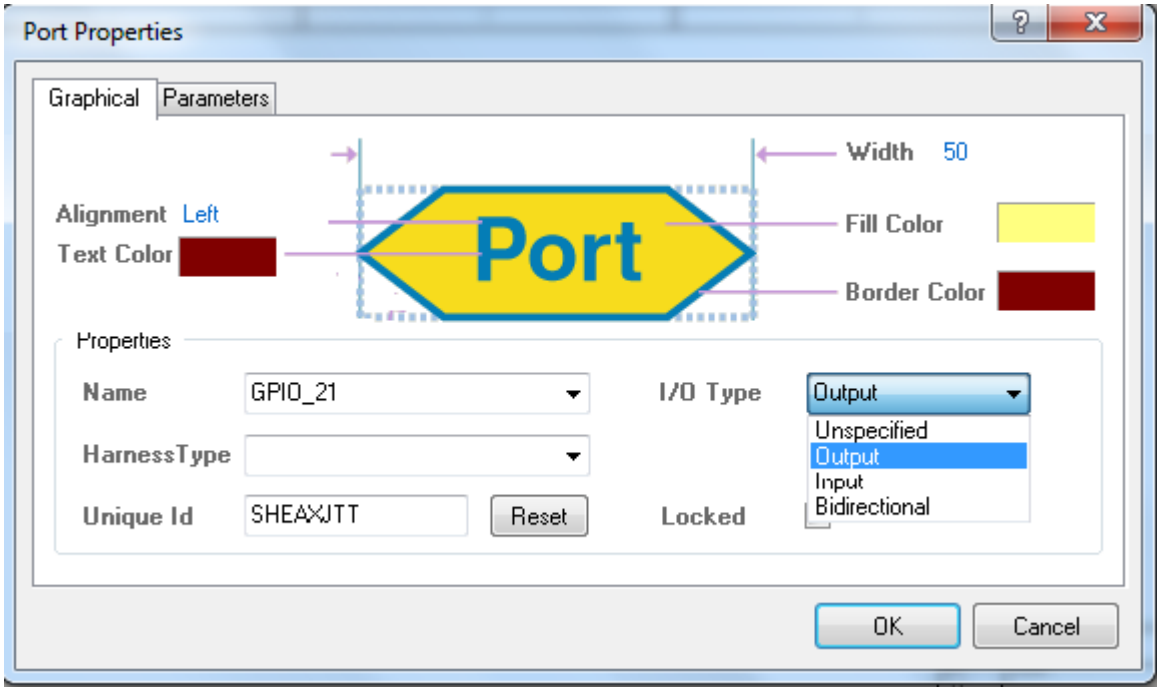

**Exemple** : le signal **TXD** est connecté à la broche **B16** du FPGA et à la broche **GPIO_21** du connecteur GPIO-2 ( **JP3**)

Extraits du fichier contrainte :

```
;------------------------------------------------------------------------
; Clocks
;------------------------------------------------------------------------
Record=Constraint| TargetKind=Port| TargetId=CLK_50 | FPGA_PINNUM=R8
;
;------------------------------------------------------------------------
; Keys
;------------------------------------------------------------------------
Record=Constraint| TargetKind=Port| TargetId=KEY_1 | FPGA_PINNUM=E1
Record=Constraint| TargetKind=Port| TargetId=KEY_0 | FPGA_PINNUM=J15
;-------------------------------------------------------------------------
; GPIO-2 Connector JP3
;-------------------------------------------------------------------------
Record=Constraint| TargetKind=Port| TargetId=GPIO_20|FPGA_PINNUM=A14
Record=Constraint| TargetKind=Port| TargetId=GPIO_21|FPGA_PINNUM=B16
```

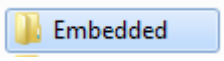
```
Record=Constraint| TargetKind=Port| TargetId=GPIO_22|FPGA_PINNUM=C14
```

⇒ Chaque broche doit être définie en entrée ou en sortie :
⇒ Cliquer droit sur le port de la broche → GPIO_21 B1 6
⇒ Exemple la broche GPIO

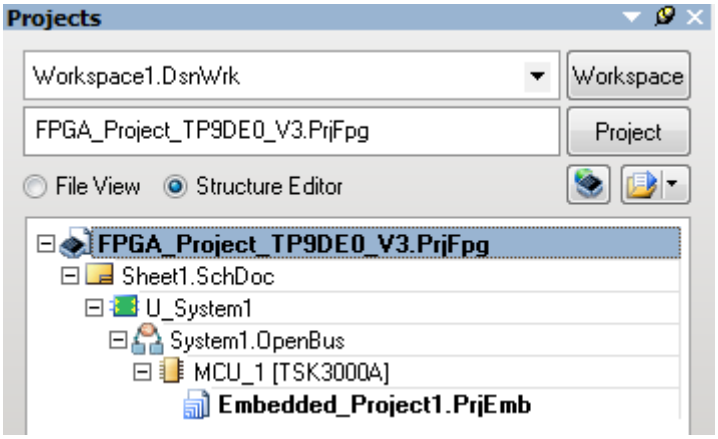



Compiler le projet hard pour le vérifier avant de passer à la suite !

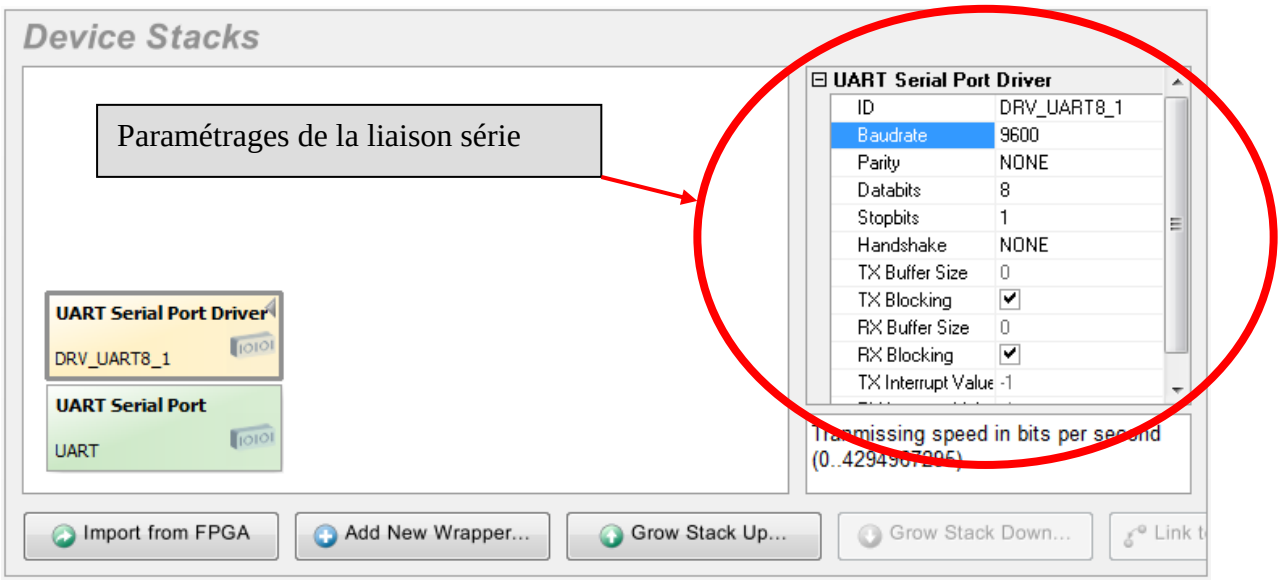
## 2.5 Créer un NOUVEAU projet embarqué et l'associer au projet hard




⇒ Sous le répertoire du projet courant créer un sous répertoire Embbeded.
⇒ Sous Altium créer un projet embarqué Embedded_Project_1.PrjEmb
⇒ associer ce projet embarqué au projet FPGA ( voir méthode dans TP4 )



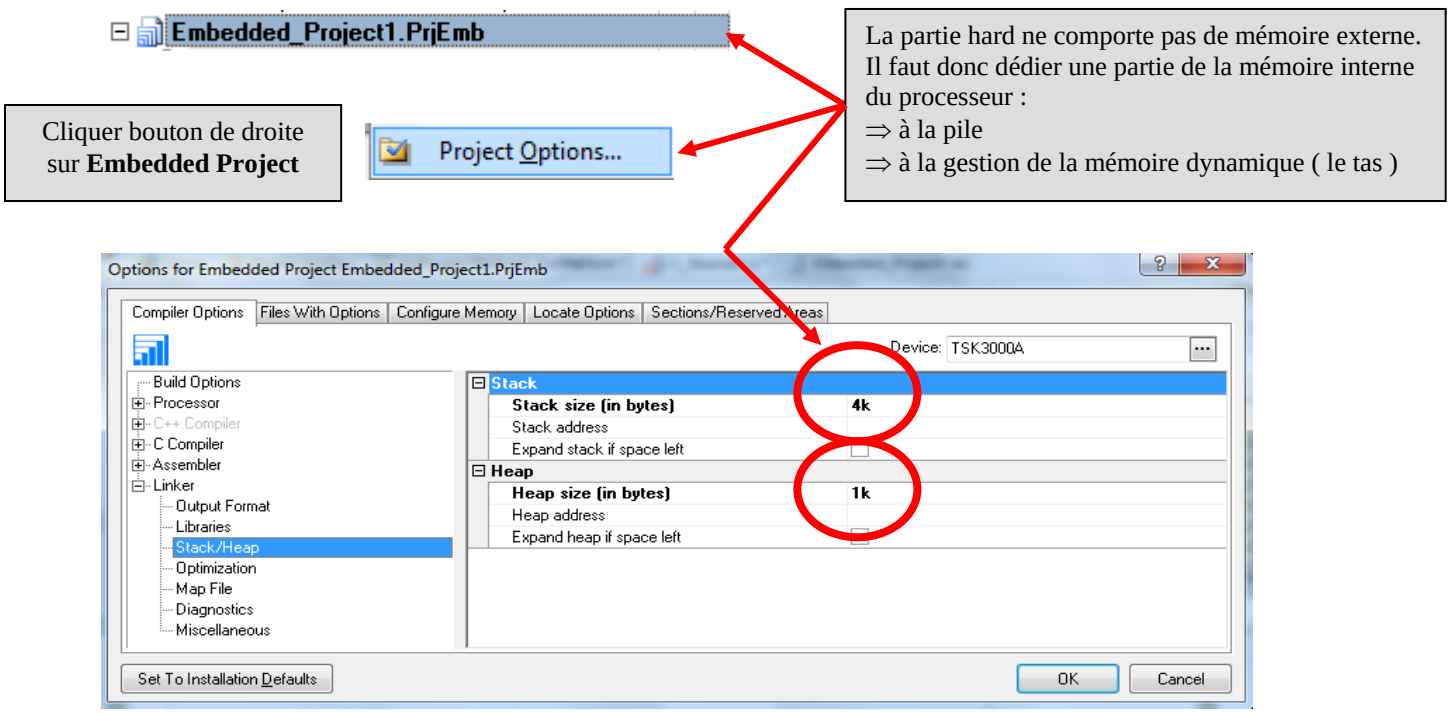
## 2.6 Créer un fichier SWWPLATFORM




## 2.7 Associer le fichier *.c développer précédemment pour la Nanoboard

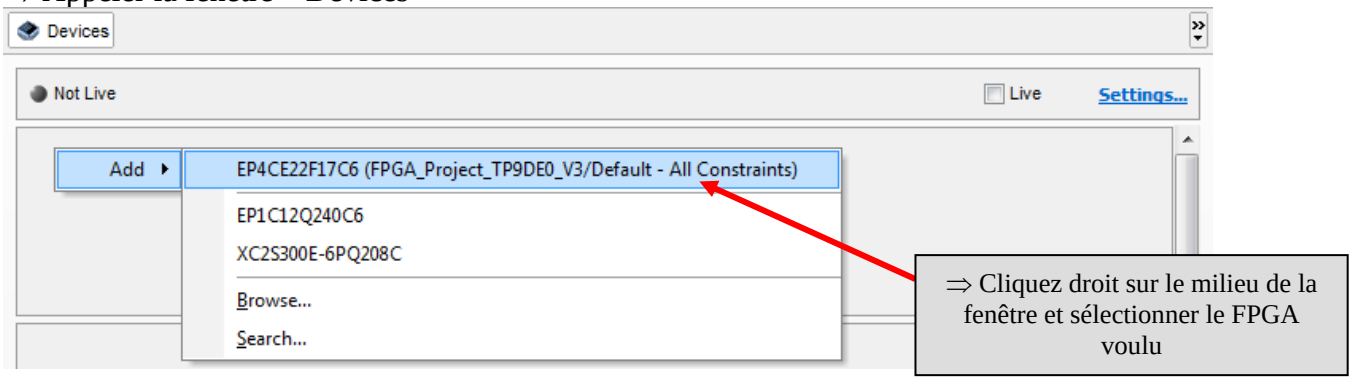
⇒ Recopier dans le sous répertoire Embbeded le fichier C et associé le au projet DE0Nano.

## 2.8 Réserver l'espace mémoire de la pile et l'espace de la mémoire dynamique

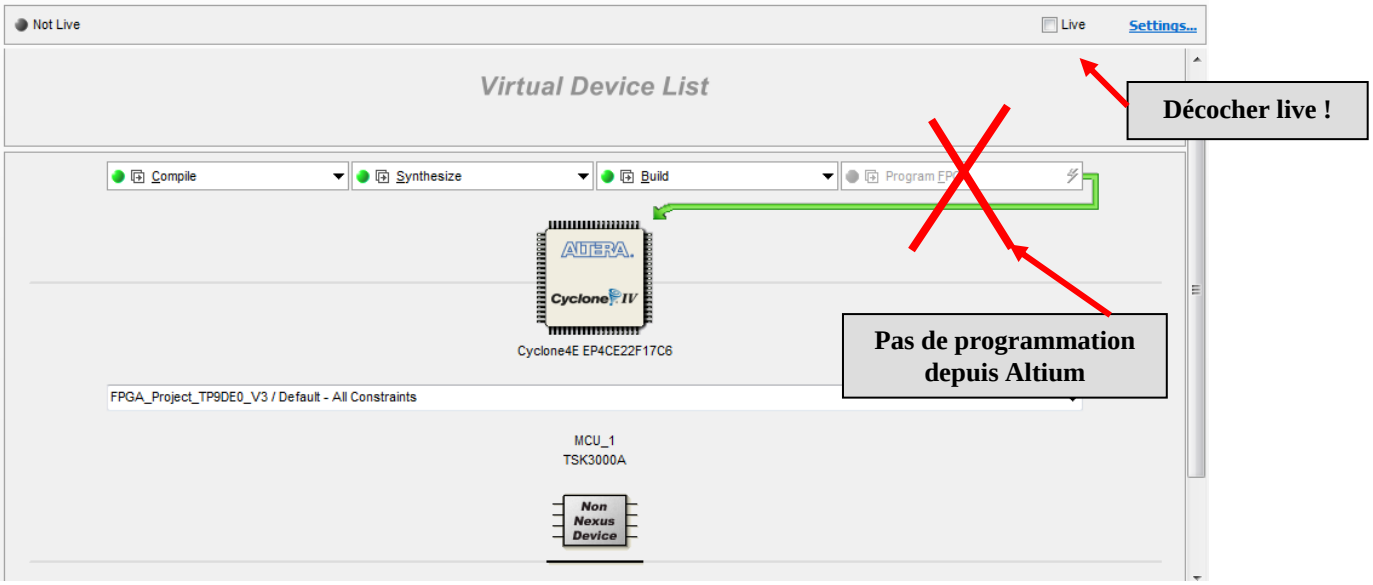


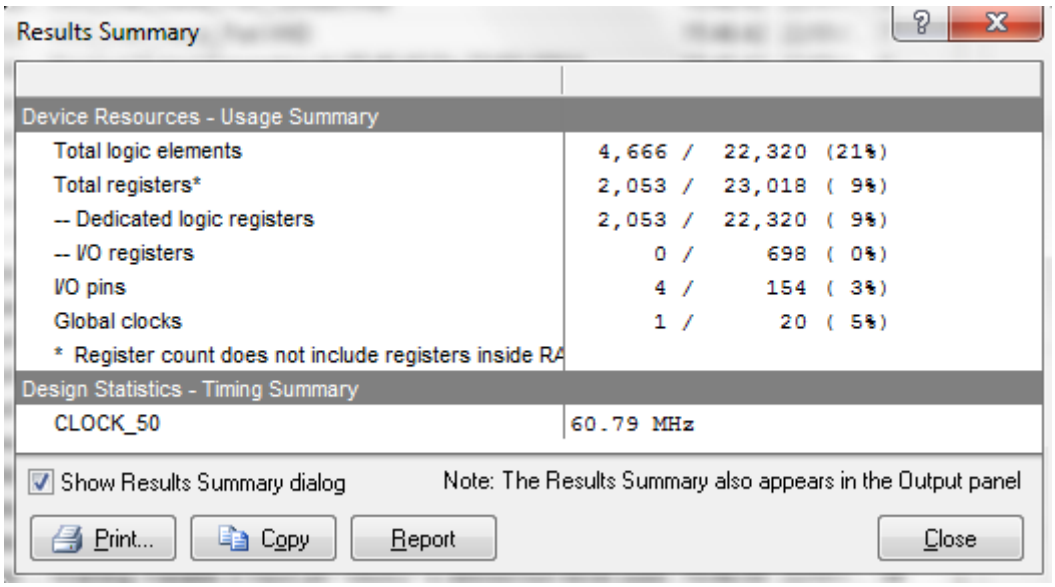
## 2.9 choisir le FPGA à programmer

⇒ Appeler la fenêtre « Devices »




## 2.10 Compiler, synthétiser, build :



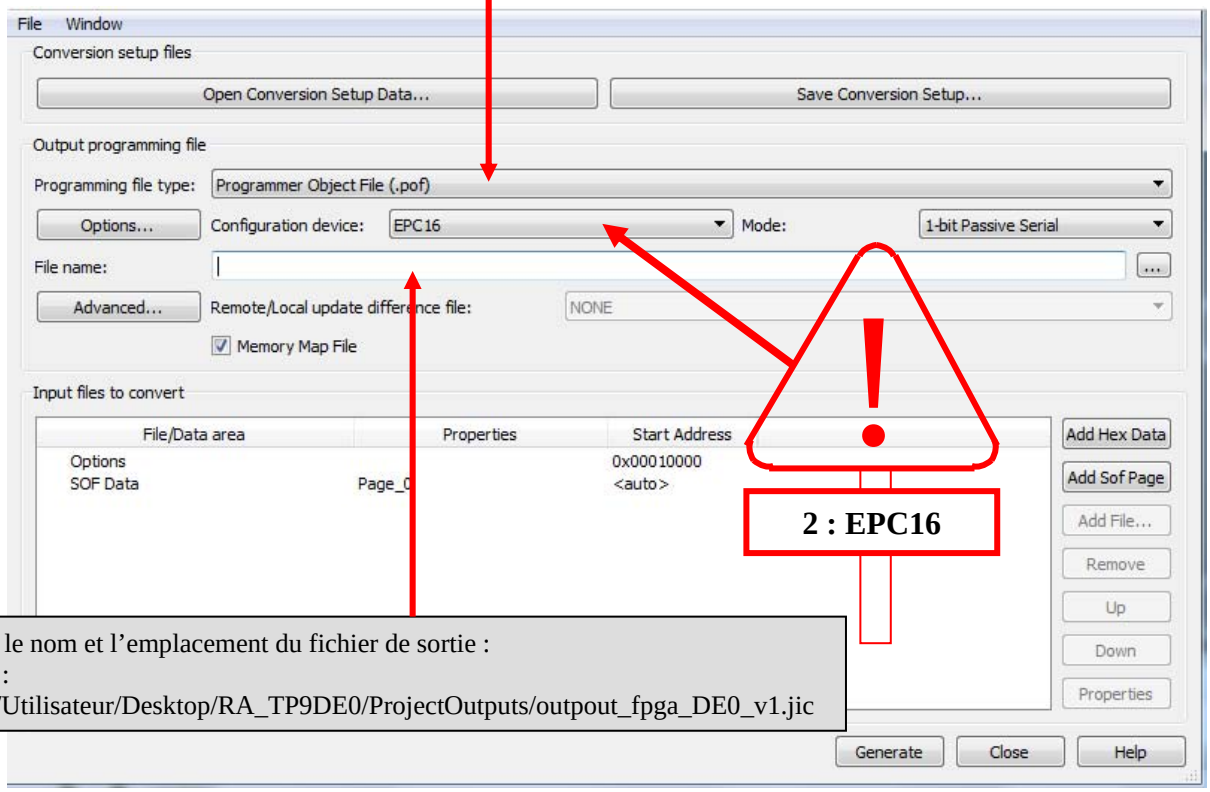

**A ce stade les fichiers *.hex et *.sof ont été générés dans les répertoires OUTPUT du projet**

## 2.11 Conversion des fichiers.

A partir des fichiers *.hex et *.sof  générés sous ALTIUM nous allons construire un fichier *.jic qui servira à programmer la DE0Nano

2.11.1 Ouvrir Quartus ⇒ Menu « file » ⇒ « Convert Programming file »

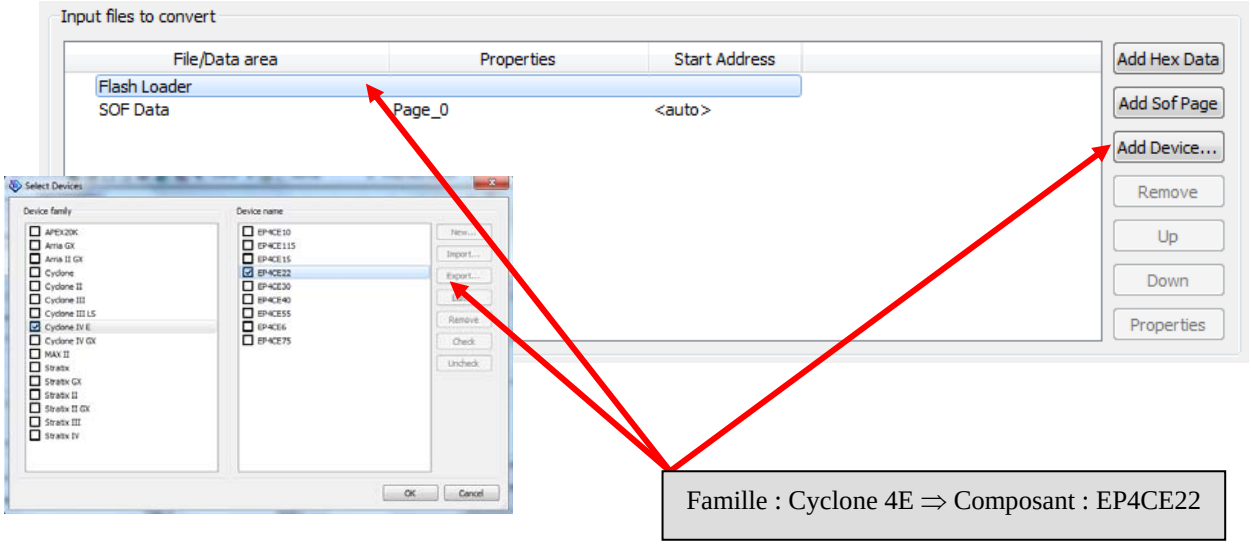
1 : Définir le type de fichier de sortie JTAG Indirect Configuration File  ***.JIC**




2 : EPC16

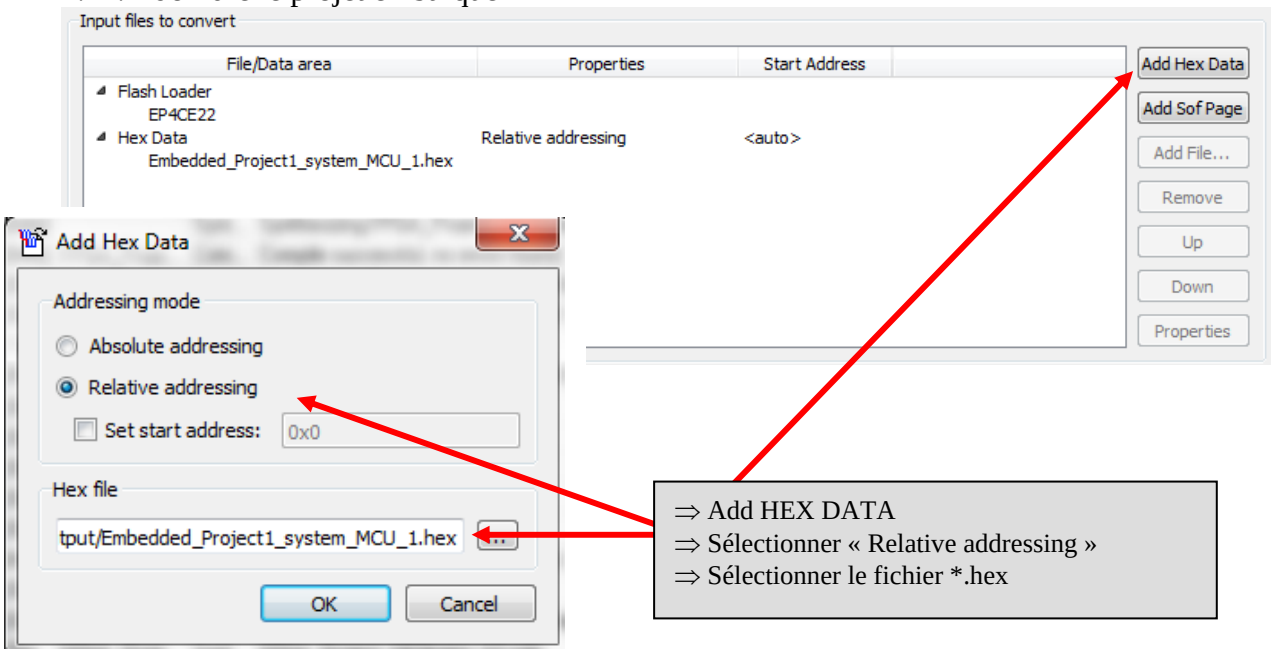
**3** Définir le nom et l'emplacement du fichier de sortie :
Exemple :
C:/Users/Utilisateur/Desktop/RA_TP9DE0/ProjectOutputs/outpout_fpga_DE0_v1.jic

⇒ Sélectionner le FPGA à programmer :

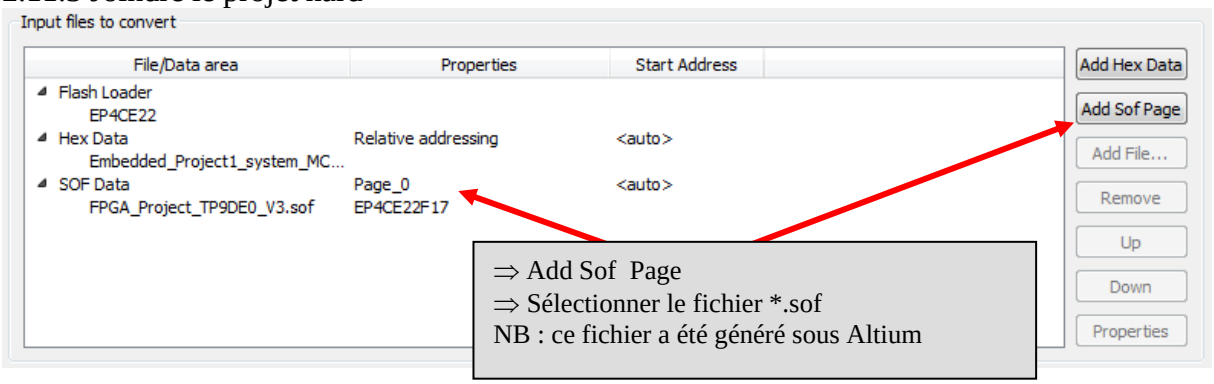



Famille : Cyclone 4E ⇒ Composant : EP4CE22

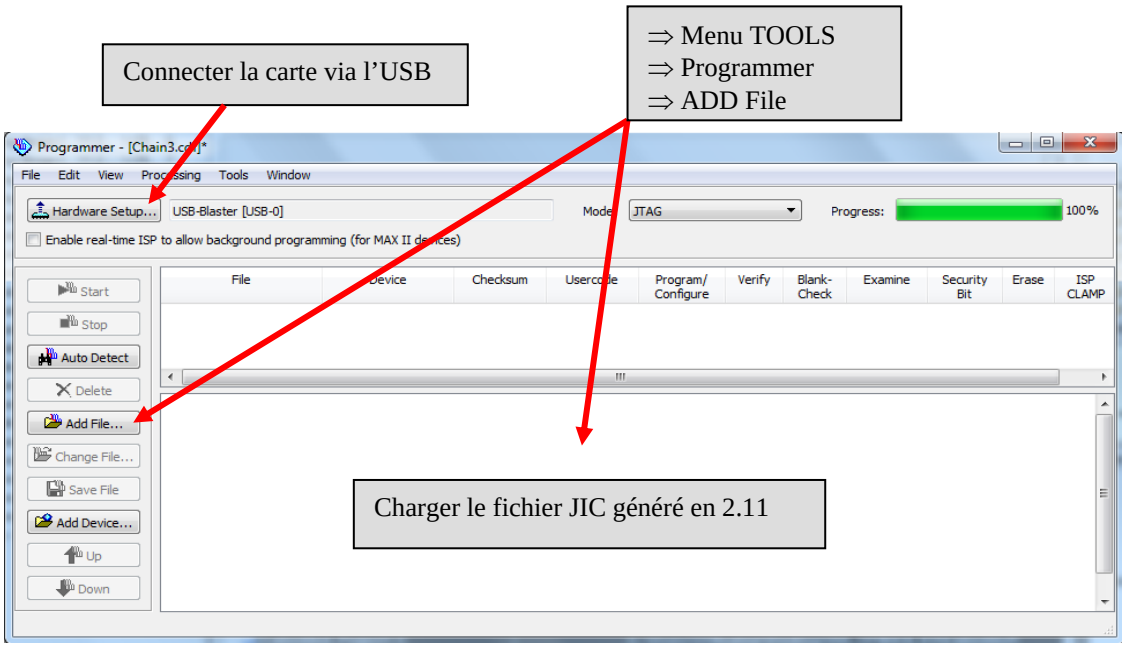
### 2.11.2 Joindre le projet embarqué

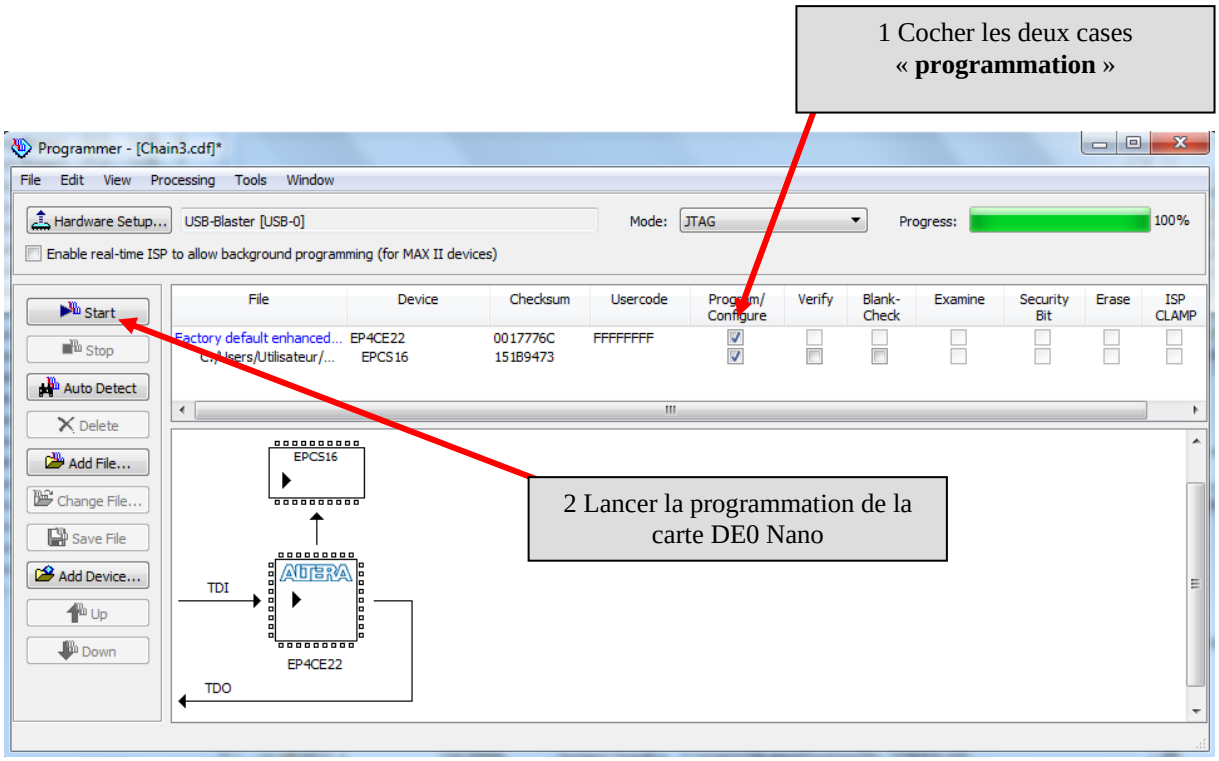



### 2.11.3 Joindre le projet hard



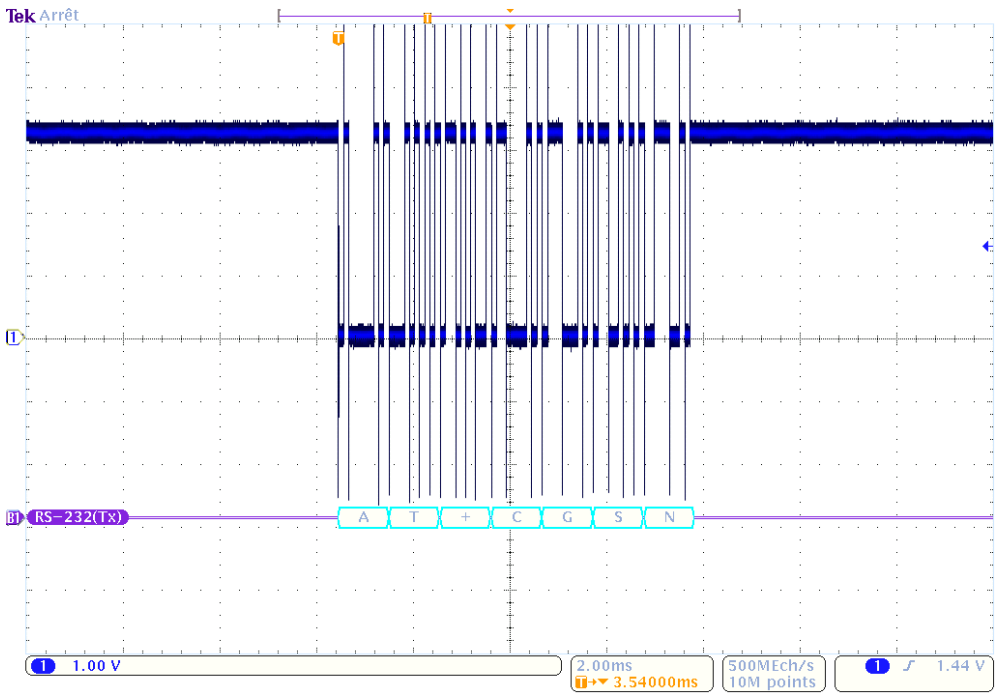

## 2.12 Programmer la DE0 depuis Quartus






Débrancher, rebrancher la liaison USB pour transférer le programme de l'EPCS16 vers le FPGA

## 2.13 Tester par mesure votre programme




********** Fin de la guidance de projet **********