

Présentation du logiciel de conception ALTERA :

Quartus II

Version 0.01 α

par J. WEISS

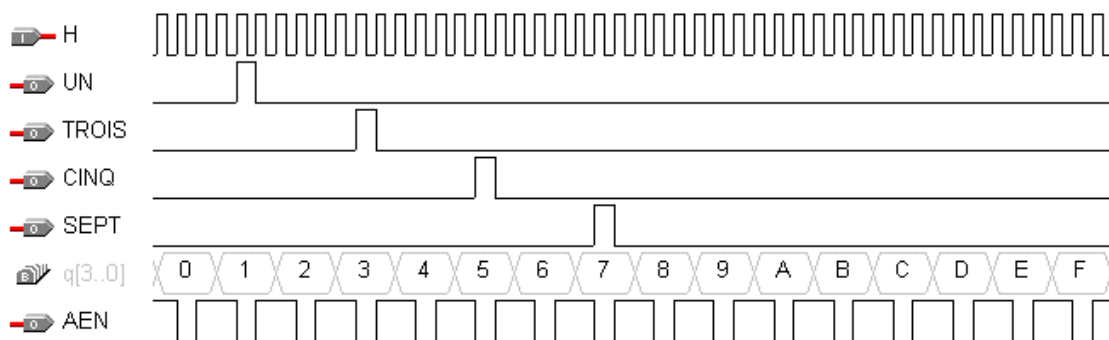
Projet étudié	1
Cahier des charges.....	1
Analyse du sujet.....	2
Conception	2
Définition du projet.....	2
Déroulement de la conception.....	2
Saisie	3
Compteur4 :.....	4
Decodeur (DECODE) :	5
Générateur de phases (GENE_3) :	5
Compilation fonctionnelle (simulation).....	6
Compilation Temporelle	8

Pour décrire le déroulement de la conception, prenons un exemple simple :

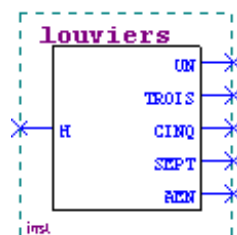
Projet étudié

Cahier des charges

On doit réaliser un module réalisant la fonction suivante :



Le symbole du module est le suivant :



Analyse du sujet

Notre module comprend donc 5 sorties et une entrée d'horloge; le cycle de la machine d'états étant de 16, il faut disposer, en interne d'un compteur 4 bits (il sera appelé **COMPTEUR4**).

Le compteur interne et la sortie AEN possèdent une période de 3 cycles d'horloge, il faudra donc concevoir un générateur de phases de même longueur (il sera appelé **GENE_3**).

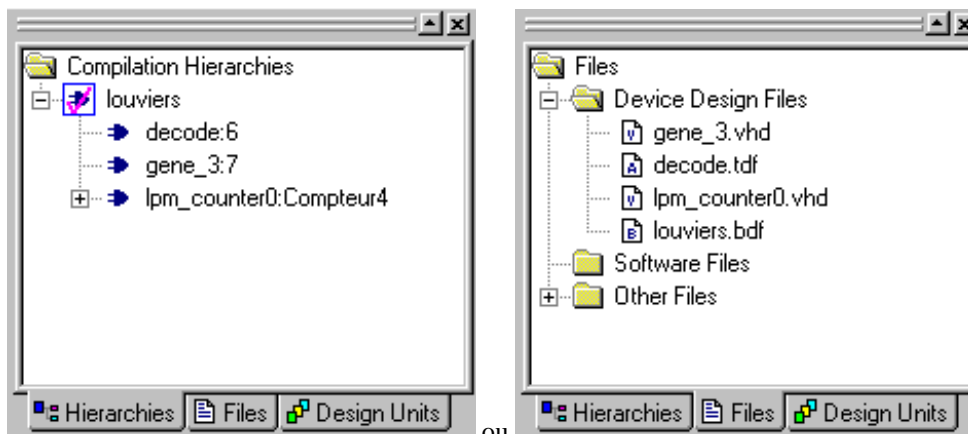
Les sorties UN, TROIS, CINQ et SEPT sont issues du décodage du compteur interne validé par AEN, on placera donc un décodeur, appelé **DECODEUR**.

Conception

Définition du projet

Quartus II gère des projets, c'est à dire des entités hiérarchiques, pour lesquels il effectue les opérations de CAO; pour illustrer cela, prenons notre module (nommé LOUVIERS), on va d'abord définir une feuille le décrivant et, pour cela, on va faire appel à des feuilles (de niveau hiérarchique inférieur) décrivant les sous-ensembles (COMPTEUR4, GENE_3 et DECODEUR).

La hiérarchie de notre projet est accessible par dans la fenêtre en haut à gauche (il y a 3 onglets sur cette fenêtre), le résultat est le suivant :

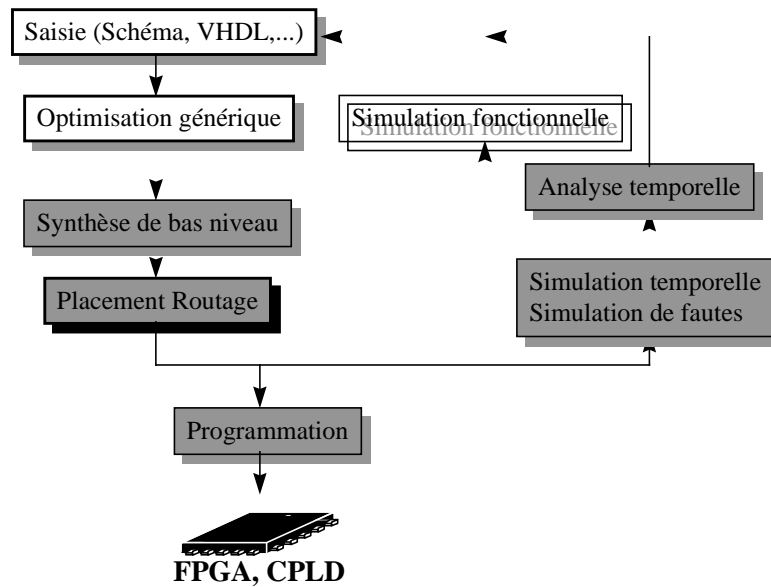


Affichage de la hiérarchie (suivant 2 onglets)

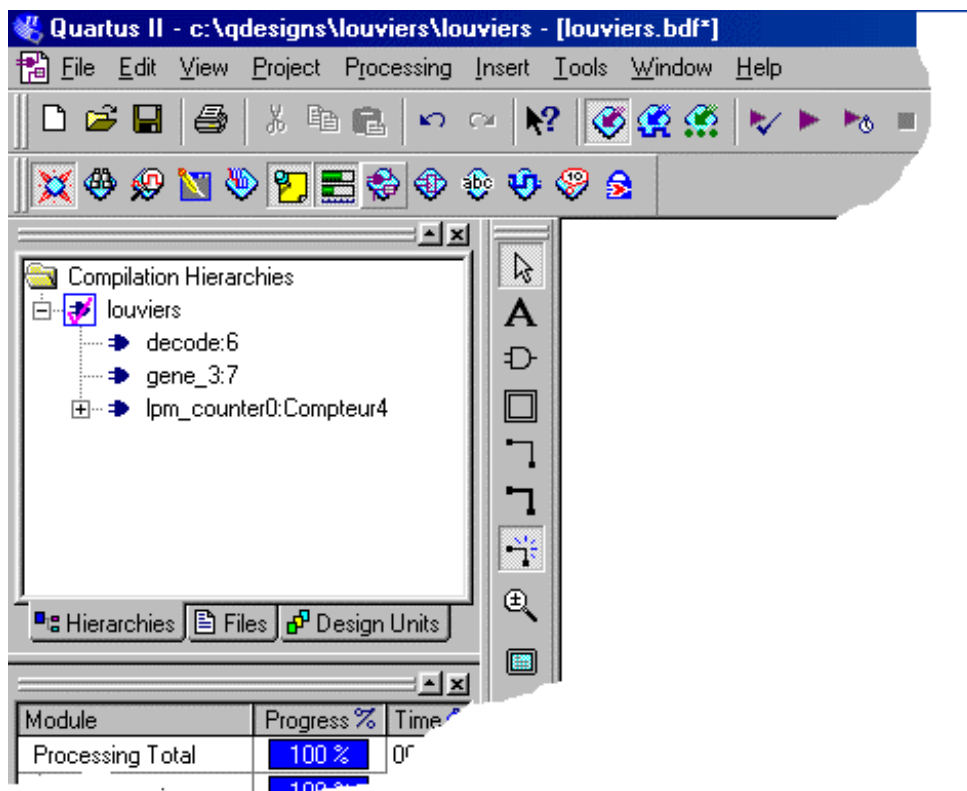
Déroulement de la conception

La figure suivante présente quelques étapes de conception, cela peut également être décrit de la manière suivante :

Saisie -> validation fonctionnelle -> validation temporelle -> programmation



La fenêtre de Quartus II est composée d'une barre de menus et de boutons (avec icônes) permettant un accès rapide aux fonctions ; le plan de travail est découpé en plusieurs fenêtres, sur la gauche, on trouve la hiérarchie du projet et l'état d'avancement des tâches, la fenêtre de droite sert de fenêtre de travail ; cette dernière dispose, sur sa gauche d'une colonne d'icônes 'contextuels'.



Saisie

Le logiciel permet plusieurs modes de saisie :

Schéma : mode de saisie graphique par association de symboles

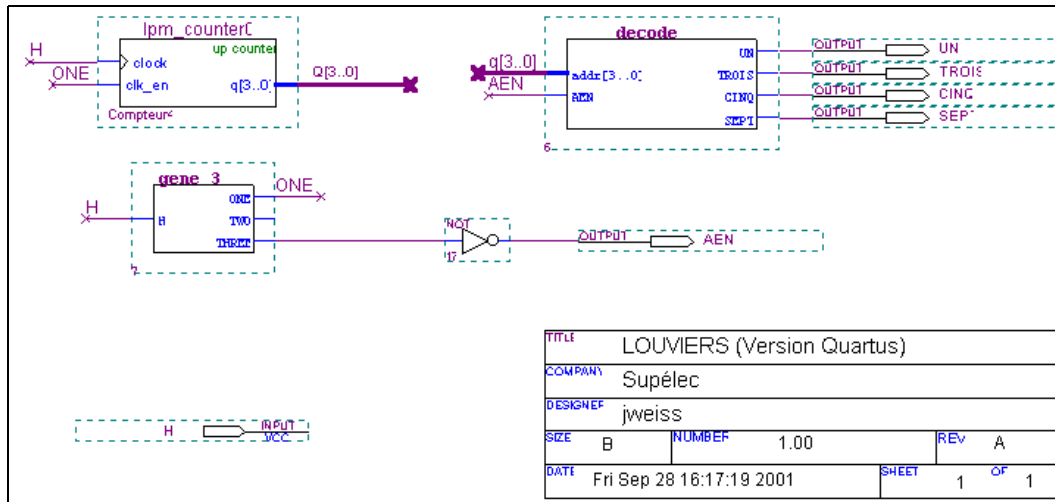
fichiers associés : *.GDF ou *.BDF

Textuel : AHDL et VHDL

fichiers associés : *.TDF et *.VHD

Pour notre exemple, nous allons choisir le schéma pour le projet, les megafonctions pour le compteur, la description textuelle (AHDL) pour le décodeur et la description textuelle (VHDL) pour le générateur de phases.

La feuille de description du projet est alors la suivante :



Feuille de description du projet (LOUVIERS.GDF)

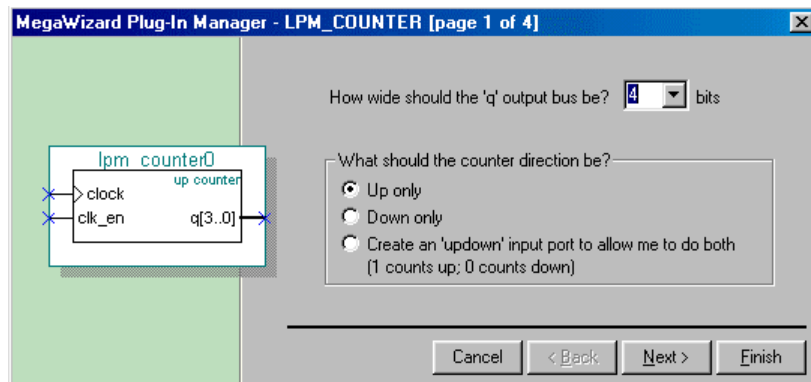
Les éléments de hiérarchie inférieure sont liés à la feuille du projet par leur symbole; pour y accéder, il suffit de "double-cliquer" sur le nom du symbole désiré dans la fenêtre de hiérarchie.

Les figures suivantes présentent la création de chaque sous-ensemble :

Compteur4 :

Nous utilisons les mégafonctions LPM (fonctions paramétrables) avec la personnalisation par le *MegaWizard Plug In Manager* :

Il faut chercher le composant (*Insert/symbol/ MegaWizard*) et se laisser guider pour le choix de la configuration de la fonction (dans notre cas : compteur 4 bits avec validation d'horloge) ; voici l'une des pages proposées :



Une des pages du *MegaWizard Plug In Manager*

Decodeur (DECODE) :

On utilise l'éditeur de texte pour composer le code, le fichier sera enregistré avec le type AHDL (*.TDF) :

```

%*****
*
*   Décodeur d'adresse
*
******%

TITLE "Décodeur d'adresse";

CONSTANT page1_addr    = H"1";
CONSTANT page3_addr    = H"3";
CONSTANT page5_addr    = H"5";
CONSTANT page7_addr    = H"7";

SUBDESIGN decode
(
  addr[3..0],
  AEN  : INPUT;

  UN,
  TROIS,
  CINQ,
  SEPT : OUTPUT;
)
BEGIN
  IF (!AEN) then
    UN      = (page1_addr == addr[]) ;
    TROIS   = (page3_addr == addr[]) ;
    CINQ    = (page5_addr == addr[]) ;
    SEPT    = (page7_addr == addr[]) ;
  END IF;
END;

```

Feuille de description du décodeur (DECODEUR.TDF)

Générateur de phases (GENE_3) :

On utilise l'éditeur de texte pour composer le code, le fichier sera enregistré avec le type VHDL (*.VHD) :

```

-- *****
-- *
-- *   Générateur à 3 phases séquentielles
-- *
-- *   J. WEISS, septembre 2001
-- *
-- *   Entrée : H : horloge
-- *   Sorties :
-- *     ONE : 1ère phase
-- *     TWO : 2ème phase
-- *     THREE : 3ème phase
-- *****

LIBRARY ieee;
USE ieee.std_logic_1164.all;
--USE ieee.std_logic_arith.all;

ENTITY GENE_3 IS
  PORT
  (
    H      : IN   STD_LOGIC;      -- Horloge
    ONE    : INOUT STD_LOGIC;     -- Phase
    TWO    : INOUT STD_LOGIC;     -- Phase
    THREE  : INOUT STD_LOGIC;     -- Phase
  );

```

```

END GENE_3;

ARCHITECTURE a OF GENE_3 IS
    SIGNAL ONE_Tmp      : STD_Logic;
BEGIN

    PROCESS (H)
    BEGIN
        ONE_Tmp <= TWO NOR THREE;
        IF (H'EVENT AND H = '1') THEN
            ONE <= THREE;
            TWO <= ONE_Tmp;
            THREE <= TWO;
        END IF;
    END PROCESS;
END a;

```

Feuille de description du générateur (GENE_3.VHD)

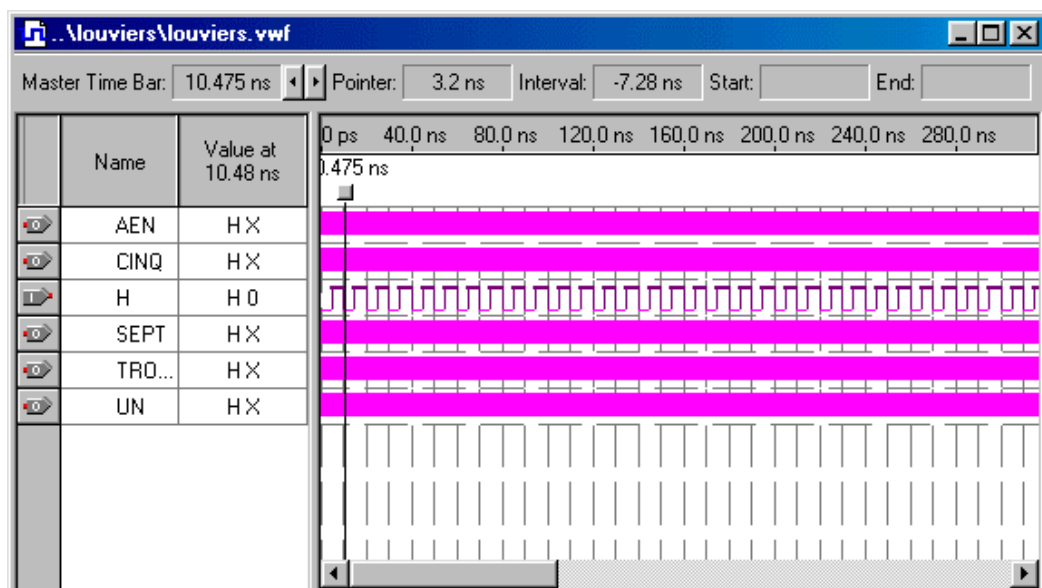
Une fois la saisie achevée, il va falloir compiler la conception.

Quartus dispose de plusieurs mode opératoire (compilation, simulation et logiciel (pour NIOS))

Compilation fonctionnelle (simulation)

Dans un premier temps, nous allons chercher à valider notre conception en fonctionnel, pour cela, il faut se placer en mode simulation (🔧) et configurer le simulateur en mode fonctionnel : *Processing/Simulator Settings/mode ...*


Il faut, en préalable à la simulation, définir les stimuli ; nous le faisons en graphique (📊) ; l'ajout de nœuds se fait par Double Clic ou par le menu *Insert/node* puis *Node Finder* (s'il n'y a pas de signaux proposés, il faut lancer une première simulation avant) ; dans notre cas, il faut définir le temps de simulation (*Time/End Time*) et le signal d'horloge (🕒):



Définition des stimuli (WaveForm Editor)


Il ne reste plus qu'à sauvegarder le fichier généré (*File/Save/LOUVIERS.WVF*)

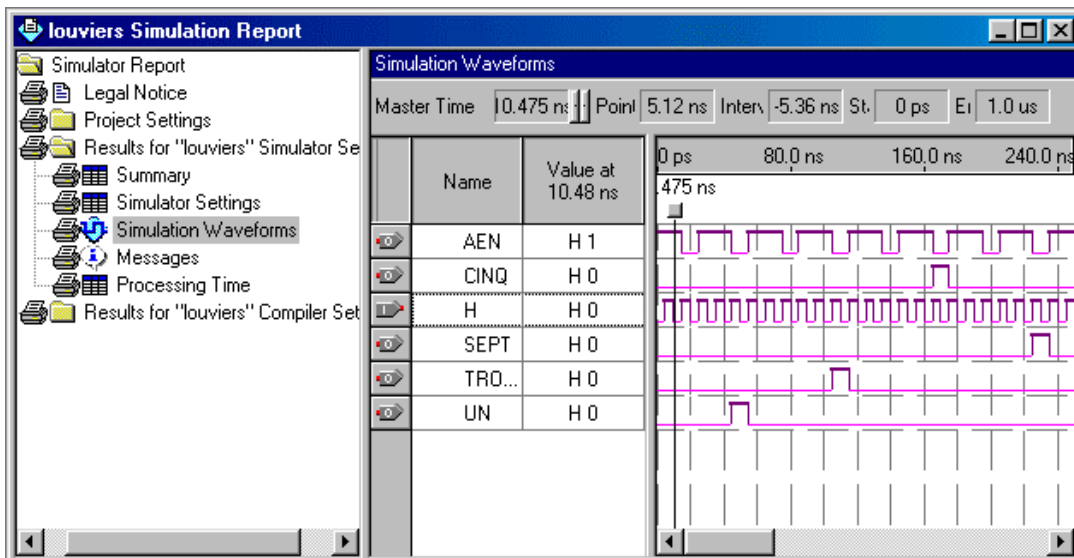
A ce niveau, on peut effectuer la simulation fonctionnelle de notre projet :

Le lancement de la simulation () fait apparaître les barres d'avancement du processus et les messages du simulateur.

A la fin de la simulation, Quartus affiche les chronogrammes résultant de la simulation.

On suppose qu'à ce stade de la conception le projet répond fonctionnellement au cahier des charges, on va donc pouvoir passer à la phase temporelle de la validation; pour cela, on reconfigure le simulateur en mode temporel (*Processing/Simulator Settings/mode ...*)


Le lancement de la simulation () fait apparaître les barres d'avancement du processus et les messages du simulateur. Le rapport de simulation est alors le suivant :




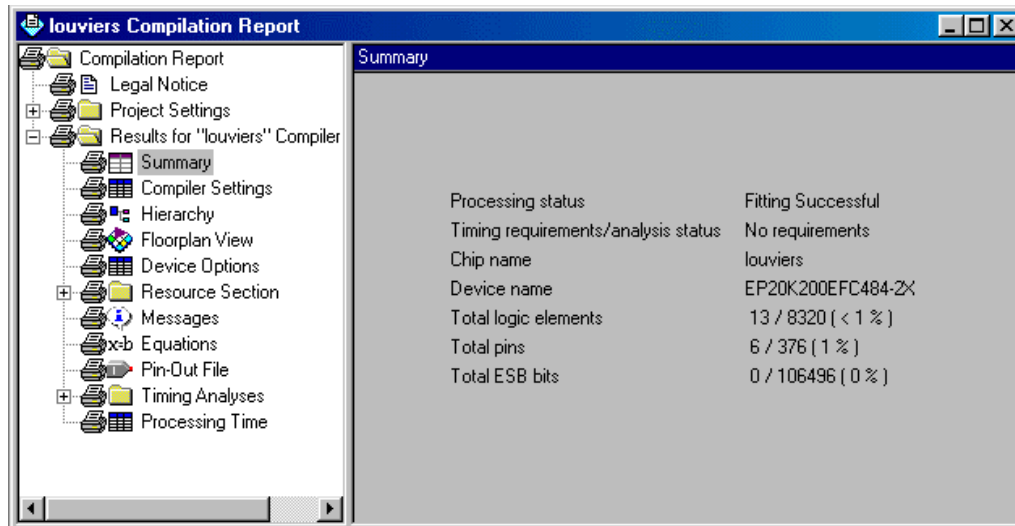
Rapport de simulation

A la rubrique "*Summary*", le simulateur indique le taux d'activité (*simulation coverage*)

Compilation Temporelle

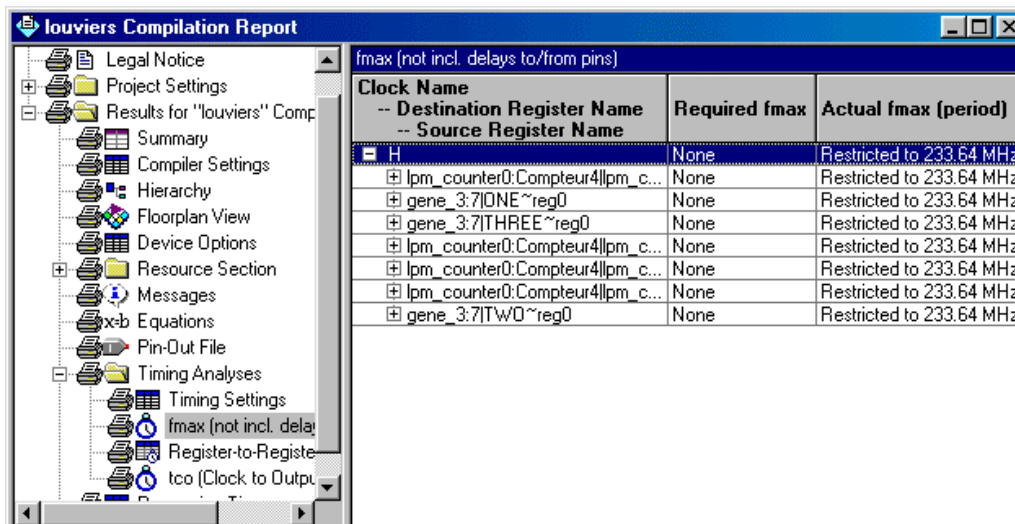
Le passage au mode de compilation se fait par l'icône  ; il faut configurer les paramètres de compilation et définir le composant utilisé (carte Excalibur : EP20K200EFC484 -2X) ; ceci se fait par les menus : *Processing/Compiler Settings/...*

Le lancement de la compilation () fait apparaître les barres d'avancement du processus et les messages du compilateur. Le rapport de compilation est alors le suivant :




Rapport de compilation (onglet : *Summary*)

On peut visualiser les résultats d'analyse temporelle à l'onglet "*Timing Analyses*" :



Rapport de compilation (onglet : *Timing Analyses*)

La définition du brochage se fait par le menu *Tools/Pin Assignment Organizer* ; pour des grosses conception, il est conseillé de clore le projet et d'éditer le fichier LOUVIERS.CSF et de remplir la rubrique [CHIP ...].

L'ultime étape consiste à programmer le composant, par le menu *MaxPlusII/Programmer* ou l'icône () .