Projet – *complètement décalé ?*

Telle est la question…

Ce projet nécessite de l'organisation, de la méthode, de l'analyse, de la réflexion, de la complétion, de la vérification, de l'explication, de la justification, …

Bref, du travail d'ingénieur !

Contexte

Une entreprise veut améliorer un système numérique de type CPU.
Votre équipe prend ce projet en cours et le gère, en concurrence avec les autres équipes.

___

Échéancier

Quatre séances encadrées sont planifiées, prévues pour *expérimenter*.
Votre équipe s'organise de manière adéquate, et par un partage équilibré des tâches, pour mener ce projet à bien.

___

Délivrables

- Votre équipe rendra un diagramme de Gantt prévisionnel détaillé en début de 2e séance : ce diagramme explicitera le travail de chacun dans l'équipe.
- Votre équipe déposera sous *moodle* deux archives Pj-24-CS3_Noms-equipiers_vx.*qar*, l'une en fin de dernière séance (x=s), l'autre mi-janvier (x=f ; la date limite DL sera précisée ultérieurement).
- Votre équipe déposera sous *moodle* un rapport Pj-24-CS3_Noms-équipiers.*pdf* (même DL).
  Ce rapport respectera la charte graphique de l'INSA et sera structuré, *i.e.*, constitué de sommaire, mots-clefs, glossaire si besoin, introduction, développement, conclusion et annexes.

Note : *le dernier paragraphe de l'introduction explicitera le plan et fil directeur du développement du rapport*.

Vous préciserez et justifierez les choix effectués pour vos réalisations ; des résultats de synthèse et de simulation, lisibles, pertinents et commentés, conforteront vos explications.

- Annexe 1 : vous y placerez vos diagrammes de Gantt prévisionnel et réel, en argumentant les différences constatées.
- Annexe 2 : vous y placerez vos fiches signalétiques établies des instructions afférentes à l'amélioration à apporter.

Note : *prévoir la rédaction du rapport qu'à partir de janvier n'est pas une bonne organisation …*

___

Observations

- Plus tôt un projet est implémenté, plus tranquille est l'équipe pour vérifier, tester, peaufiner, …
- Le papier-crayon est le support le plus efficace dans la réflexion …
- Étudiez, toujours, l'ensemble des spécifications, avant de réaliser, simuler, prototyper.
- Appliquez le cycle de développement en V dans votre démarche : développez et testez petite fonctionnalité par petite fonctionnalité, pour y gagner globalement en temps de débogage, et pour terminer une séance de travail avec un système propre, compréhensible, et compilable.
- Appliquer la méthode DMAIC dans l'amélioration de la CPU est un plus …

_____

Annexe A – *la* CPU

L'archive Pj-24-CS3.*qar* est une version opérationnelle de la CPU CS3.
Ces caractéristiques sont globalement définies dans les documents afférents.

CPU *originelle*

Le fichier CPU décrit la CPU en VHDL.

Note : *comprendre une description* VHDL *n'est absolument pas un objectif de ce projet*.

- init_stack initialise la pile logicielle interne à zéro.
- init_RAM initialise la RAM interne par des octets.
- init_PROM précharge la ROM interne : elle décrit le code programme à exécuter, instruction par instruction, octet par octet.

CPU_cfg configure élémentairement la CPU pour la simulation.
CPU_test est le script pour simuler CPU_cfg.
Le chronogramme obtenu correspond à l'exécution du code programme décrit.

CPU *améliorée*

Deux sous-ensembles de la CPU sont à améliorer.

- L'unité de traitement est enrichie d'une nouvelle unité d'exécution, *cf.* annexe B.
- Le diviseur de fréquence devient paramétrable, *cf.* annexe C.
- La CPU est à caractériser, *cf.* annexe D.

Annexe B – Pj_Unit ≅ k-*bit shifter*

Pj_Unit est l'unité d'exécution qui traite itérativement k décalage(s) de 1 bit sur un octet ou un mot 16-bit.

Décaleur ExU-2 *originel*

ExU-2 est l'unité d'exécution de la CPU qui traite un décalage de 1 bit sur un mot.

La CPU charge le mot à décaler dans le(s) registre(s) adéquat(s), ExU-2 génère le résultat du décalage combinatoire de 1 bit, la CPU le stocke en file de registres et sauvegarde les indicateurs d'état dans le registre de statut.

Le code programme teste les instructions relatives à cette unité : les sous-graphes d'état et chronogramme de simulation sont explicites sur leur fonctionnement.

Décaleur Pj_Unit *amélioré*

Analyse : la CPU traite k décalage(s) 1-bit en exécutant séquentiellement k instruction(s) identique(s), ce qui n'est guère performant.

Amélioration : la CPU doit traiter k décalage(s) 1-bit en une seule instruction avec le décaleur k-bit, k est ainsi codé dans l'instruction.

Les instructions afférentes sont déjà implémentées dans la CPU ; le code programme en teste.

Décaleur k-bit – *fonctionnement contraint*

La CPU charge le mot à décaler dans le décaleur, puis le mot k.
Pj_Unit génère le résultat, la CPU le stocke en file de registres et sauvegarde les indicateurs.

Ce décaleur traite itérativement un à quatre décalages, suivant k%8 (k *modulo* 8) :

- si k%8 est nulle, l'instruction effectue quand même un décalage (priorité dans l'utilisation de l'instruction sur la valeur nulle de k%8) ;
- si k%8 est supérieure à quatre, le décaleur ne traite que quatre décalages.

Décaleur k-bit – *partitionnement fonctionnel nivelé*

Pj_Unit fournie est une ébauche schématique à deux niveaux.

- *Niveau* 0 élaboration d'un décaleur 1-bit bidirectionnel sur octet
  + génération du débordement.

Note : *l'agencement des bistables diffère pour chaque équipe, à demander à l'équipe encadrante*.

- *Niveau* 1 instanciation de deux décaleurs *Niveau* 0, LSB et MSB
  + gestion des entrées série des deux décaleurs
  + gestion des débordements
  + gestion des décalages par une FSM de *Moore* locale.

Décaleur k-bit – *simulation / prototypage*

shifter_test est le script de simulation adapté au décaleur 1-bit sur octet shifter_8b.

Compiler, puis charger shifter_prototyping sur carte DE10-Lite, permet de prototyper shifter_8b.
Les afficheurs *hex_* sont pilotés par un transcodeur hexadécimal → 7 segment usuel.

PjU_test est le script de simulation adapté au décaleur k-bit Pj_Unit.

Décaleur k-bit – *contraintes de réalisation*

Pj_Unit est limitée à 25 bistables.

La FSM locale a une structure simple : registre d'état
+ fonction combinatoire d'évolution
+ fonction combinatoire de sortie.

Niveau 0 : dans ce logigramme, vous préciserez l'ordre imposé des bistables.

L'utilisation d'IP sélective combinatoire LPM_... du catalogue IP est acceptable, mais avec au plus une commande sur deux bits et au plus pour un format 4-bit.

Par exemple, au maximum : *multiplexeur* 4:1 ou *décodeur* 2:4, pour mots 4-bit.

Tout composant ajouté est adéquatement nommé (pas de « inst »).

Tout nouveau fichier est placé dans le sous-répertoire approprié et ordonné adéquatement dans la liste des fichiers constituant le projet.

Toute originalité efficace dans la réalisation de Pj_Unit est un plus …

___

Décaleur k-bit ↔ CPU

Pj_Unit est interfacée à la CPU.

Les diagrammes temporels sont explicites : l'exécution d'une instruction adaptée génère la fonction et les commandes adéquates, cycle par cycle.

PjU_unit active le signal done au cours du dernier décalage à traiter : done est la condition pour sortir la CU de la CPU de son état d'attente, afin qu'elle gère au cycle suivant le stockage du résultat et des indicateurs.

_____

Annexe C – *le* FDU

Le diviseur FDU interne à la CPU divise la fréquence du signal d'horloge externe pour générer un signal d'horloge interne cyclique propre et puissant, synchronisant l'ensemble des flip-flops de la CPU.

FDU *originel*

La configuration du flip-flop T divise par deux la fréquence du signal d'horloge externe.

___

FDU *amélioré*

Le diviseur devient paramétrable par la commande externe Sclk : cette commande sélectionne le signal d'horloge externe divisé par deux, ou par quatre, ou par huit, ou par seize.

La commande varie dans la simulation prévue de la CPU : la division par deux correspondra à la valeur de la commande la plus utilisée durant cette simulation.

La variation de la commande impactera de manière synchrone la CPU.

___

FDU – *contraintes de réalisation*

L'utilisation d'IP sélective combinatoire LPM_... du catalogue IP est acceptable, mais avec au plus une commande sur un bit.

Le FDU amélioré combinera au moins deux types de bistable, si besoin.

Tout composant ajouté est adéquatement nommé (pas de « inst »).

Tout nouveau fichier est placé dans le sous-répertoire approprié et ordonné adéquatement dans la liste des fichiers constituant le projet.

Toute originalité efficace dans la réalisation de l'unité FDU est un plus …

_____

Annexe D – *réflexions diverses*

- Avant complétion, puis après complétion, caractérisez shifter_8b, Pj_Unit, FDU et CPU, en ressources matérielles optimisées et paramètre f_{max} : commentez.
- Vérifiez l'implémentation RTL des quatre systèmes de mémorisation : ROM, RAM, pile, file de registres. Justifiez la caractéristique de 768 bits mémoire issue de la synthèse.
- *Quartus Prime* affecte par défaut le codage des états de la CU de la CPU.
  Cet encodage est lisible sous l'éditeur approprié, *via* l'onglet *Encoding*.

  Analysez l'impact du type d'encodage sur les caractéristiques de la CPU.
  Portez votre analyse sur les encodages *auto*, *Johnson*, *minimal bits*.

Note : *l'analyse globale des transitions entre états du graphe n'est pas demandée, excepté pour les instructions afférentes à* EXU-2 *et* Pj_Unit (fiches signalétiques à établir)

- .Que vous inspire le *warning* 332125 dans la synthèse de la CPU ?

Note : *pallier ce warning n'est pas demandé.*