Philip Simpson

L'USINE NOUVELLE

LA CONCEPTION DE SYSTÈMES AVEC FPGA

Bonnes pratiques pour le développement collaboratif

Traduit de l'anglais par **Daniel Etiemble**

DUNOD

French edition updated and completed by Philip Simpson.

Translation from English language edition:
FPGA Design by Philip Simpson.

© Dunod, 2014 pour la version française

5 rue Laromiguière, 75005 Paris
www.dunod.com

ISBN 978-2-10-070792-8

Table des matières

Avant-propos

Les bonnes pratiques pour la conception avec FPGA

Ce livre décrit les bonnes pratiques pour la conception des FPGA. Il est le résultat de réunions avec des centaines de clients sur les problèmes auxquels sont confrontées leurs équipes de conception utilisant des FPGA.

En acquérant une compréhension de leurs environnements de conception, des processus, de ce qui fonctionne, de ce qui ne fonctionne pas, j'ai été en mesure d'identifier les différents types de problèmes dans les conceptions de systèmes. Plus important encore, cela m'a permis de définir une méthodologie à suivre qui fournit les conseils pour utiliser les bonnes pratiques pour surmonter les difficultés.

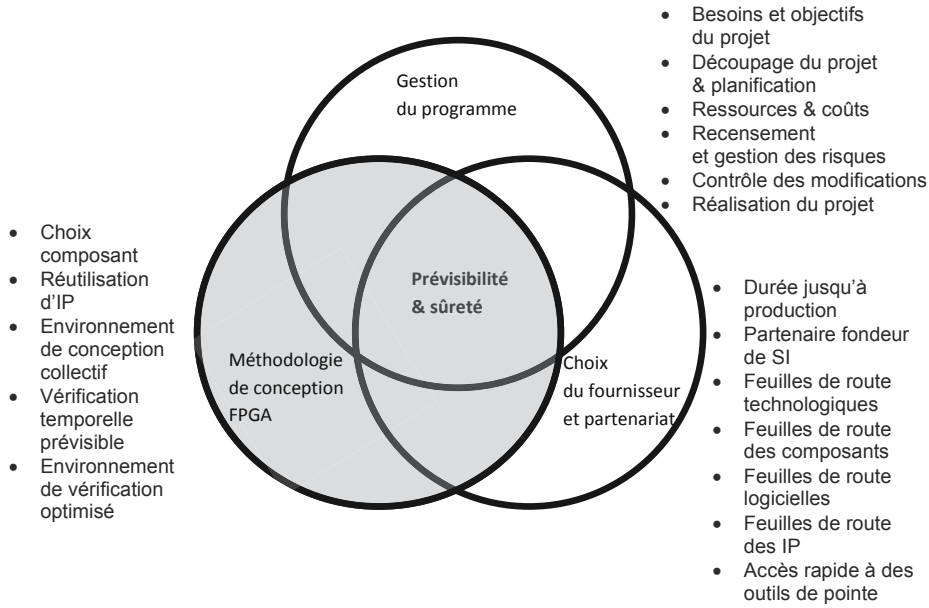
Ces données mettent l'accent sur les équipes de conception dispersées sur plusieurs sites. Le but étant d'augmenter la productivité des équipes de conception de FPGA en établissant une méthodologie commune entre elles, permettant l'échange de blocs de conception entre les équipes.

Ces bonnes pratiques ont permis d'établir une feuille de route pour avoir des résultats prévisibles lors des conceptions de systèmes dans un FPGA.

Les trois étapes pour des résultats prévisibles sont les suivantes :

1. Établir une bonne planification du projet et de son ampleur.
2. Choisir le bon composant FPGA pour être sûr que la technologie adéquate est disponible pour les projets d'aujourd'hui et de demain.
3. Suivre les bonnes pratiques dans le développement de la conception de FPGA pour raccourcir le cycle de conception et veiller à ce que les circuits soient terminés dans les délais prévus et que les blocs de circuit puissent être réutilisés dans de futurs projets avec un minimum d'effort.

Les éléments clé pour réussir une conception FPGA



Les trois étapes pour réussir un développement de FPGA.

Ces trois aspects doivent être bien coordonnés pour réussir à coup sûr une conception avec FPGA.

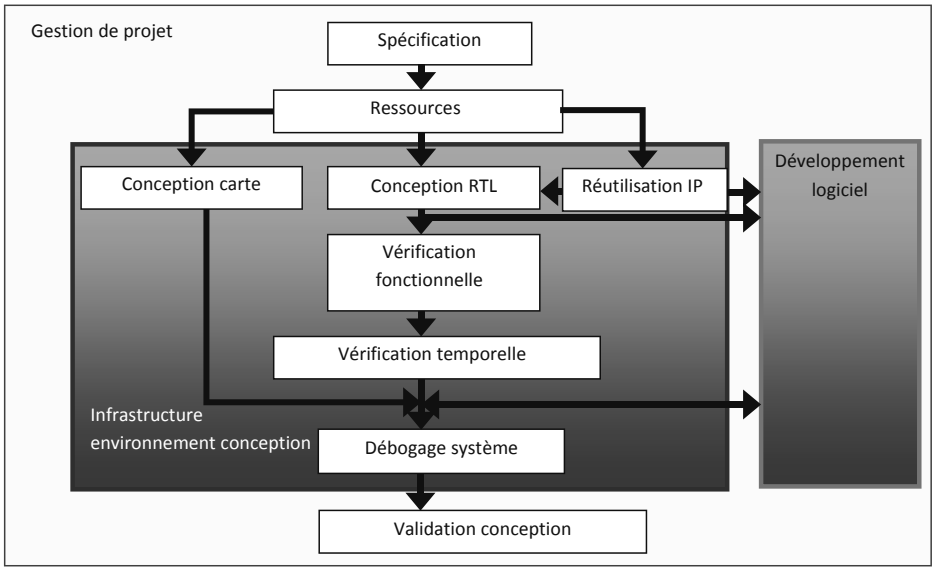
Le choix du fournisseur doit être un partenariat à long terme entre les deux sociétés. En partageant des feuilles de route et en gérant conjointement des projets existants, on peut être sûr que non seulement le projet soit un succès, mais qu'il fournira aussi des solutions adéquates à temps pour de futurs projets. C'est un processus d'adaptation fine fondée sur l'expérience du travail commun pour garantir le succès des projets.

Le troisième aspect est la méthodologie de conception de FPGA.

C'est l'objectif principal de la méthodologie des bonnes pratiques. Cela couvre le flot de conception FPGA complet des fondements aux techniques de pointe. Cette méthodologie est indépendante du fournisseur de FPGA dans la mesure où les sujets et les recommandations (les bonnes pratiques) s'appliquent à la conception de tous les FPGA. La plupart du contenu est générique, même s'il contient des références à des fonctionnalités des outils de conception d'Altera qui renforcent les bonnes pratiques recommandées.

Comment est structuré ce livre ?

Le schéma présenté dans la figure suivante montre les grandes lignes des bonnes pratiques pour la conception de FPGA.



Les bonnes pratiques pour réussir la conception d'un FPGA.

Chacun des blocs du schéma correspond à un chapitre de ce livre.

Un chapitre supplémentaire est consacré à l'énergie car elle recoupe de nombreux autres domaines de la méthodologie de conception. Les thèmes *Dessin de la carte*, *Conception RTL*, *Réutilisation des IP*, *Vérification fonctionnelle* et *Vérification temporelle* sont plutôt des thèmes pour lesquels les équipes de conception ont différentes méthodologies de conception et pour lesquels les ingénieurs ont besoin de conseils pour obtenir des résultats cohérents et raccourcir le cycle de conception.

Bon nombre des problèmes qui se posent dans la conception avec FPGA ne sont pas spécifiques à ce type de conception, mais sont des problèmes communs à toute conception. Les composants FPGA en eux-mêmes amènent des problèmes et des possibilités spécifiques par rapport aux conceptions ASIC. L'augmentation de la capacité des composants FPGA a conduit à des conceptions beaucoup plus complexes ciblant les FPGA et à une migration naturelle des concepteurs d'ASIC vers la conception de FPGA. De nombreuses équipes de conception ont ainsi été amenées à transférer les principes de conception ASIC vers les conceptions avec FPGA. En général, cela a été bénéfique au flot de conception FPGA. Mais on doit prendre en compte les avantages que les FPGA apportent au flot de conception.

La nature programmable des FPGA permet d'effectuer plus de vérifications du système. Correctement utilisée, elle peut grandement accélérer le cycle de vérification, mais les mauvaises utilisations peuvent aussi allonger le cycle de conception. La nature configurable des E/S pose des problèmes qui n'existent pas dans la conception ASIC. Les outils qui sont fournis par l'industrie de la CAO électronique sont également différents pour les FPGA et pour les ASIC, à la fois en termes de fonctionnalités et de coûts.

Ce livre vous aidera à adopter la bonne méthodologie de conception pour répondre à vos besoins.

Bien qu'il soit recommandé de lire le livre dans son intégralité, on peut se focaliser sur les différents chapitres qui concernent les domaines du flot de conception qui posent les plus grands problèmes à son équipe de conception.

Remerciements

Misha Burich, qui a fourni l'étincelle conduisant au concept des bonnes pratiques.

Brian Holley et Catizone Rich pour avoir fait vivre l'idée *via* leurs clients et avoir fourni une source constante de retours d'expériences.

Gregg Baeckler pour sa magie dans les optimisations RTL.

Les nombreux clients qui ont contribué au contenu en décrivant leurs environnements de conception et les problèmes auxquels ils sont confrontés dans la réalisation de leurs conceptions de système avec les composants FPGA.

Jean-Michel Vuillamy pour sa transmission continue des retours d'expérience sur la méthodologie de conception au cours des nombreuses années où nous avons travaillé ensemble.

Mon beau-père, James Leroy pour m'avoir motivé en permanence et son humour de cour d'école.

Ma femme Jill et ma fille Kayla, pour leur patience, leur soutien et leur amour.

Philip Simpson

1 Gestion de projet

Ce chapitre donne un bref aperçu sur la gestion de projet. Il expose rapidement les notions de base sur les phases d'un projet, l'estimation de leur durée et la gestion du planning global.

1.1 Le rôle de la gestion de projet

Le but de la gestion de projet est de fournir les bonnes caractéristiques à temps et en respectant le budget. Trois dimensions sont impliquées :

1. Les caractéristiques
2. Le temps de développement
3. Les ressources

Le chef de projet doit trouver le juste équilibre entre ces trois aspects pour répondre aux objectifs du projet.

1.2 Les phases de la gestion de projet

Chaque projet peut être décomposé en trois phases de gestion :

1. **La phase de planification** – Elle établit la liste des fonctionnalités, élabore le planning du projet et détermine l'ensemble des ressources et le budget.
2. **La phase de suivi** – Il s'agit de la tenue des examens mensuels d'avancement, les mises à jour du planning hebdomadaire, l'examen du budget et de l'encadrement nécessaire et l'examen des demandes de modifications d'ingénierie.
3. **La phase de synthèse** – Cela implique les rétrospectives de projets, l'examen des données et des améliorations de fonctionnement et le plan d'actions.

1.2.1 Estimer la durée du projet

Estimer la date de fin d'un projet est plus facile en suivant les étapes suivantes :

1. Sélectionner l'un des derniers grands projets terminés avec succès.
2. Créer un macro-modèle. Il s'agit d'identifier les principales phases du projet pour la spécification, la réalisation et la vérification. Extraire la durée exacte des phases et de tout chevauchement.
3. Définir l'objectif global d'amélioration des processus. Un exemple serait de vouloir réaliser un projet d'une complexité similaire 10 % plus vite.
4. Définir des métriques de complexité du projet telles que les caractéristiques de la conception et d'utilisation des ressources. Les caractéristiques de conception peuvent inclure le nombre de pages du cahier des charges, le nombre de ressources FPGA, le nombre de lignes de code RTL, la vitesse, la complexité technique.
5. Calculer le facteur d'échelle k par rapport au projet de référence du point 1.
6. Adapter le projet à venir en fonction du facteur d'échelle.
7. Évaluer le projet de manière adéquate et faire les ajustements appropriés.

1.2.2 Planning

Le planning du projet doit être mis à jour régulièrement. Il est recommandé qu'il soit mis à jour au moins une fois par semaine.

Toutes les réunions de mise à jour du planning doivent être concises et doivent seulement se concentrer sur la collecte des informations relatives à l'état du projet. Cela comprend les informations indiquant si une tâche a été commencée, si une activité est terminée, combien de temps il faudra pour terminer une tâche, et toute information sur la tâche d'un ingénieur qui détermine le niveau d'avancement d'une tâche.

Les réunions de mise à jour doivent également être utilisées pour estimer quand une tâche sera terminée. Le chef de projet doit respecter les estimations de la durée d'exécution d'une tâche découlant des ressources, mais doit questionner toutes les estimations qui semblent être complètement erronées.

Estimation du planning hebdomadaire

Le chef de projet doit analyser rigoureusement le planning du projet sur une base hebdomadaire. Il y a dix tâches principales impliquées dans ce processus :

1. Analyser et scruter les chemins critiques.
2. Passer en revue les tâches prévues pour la semaine à venir.
3. Discuter et s'accorder sur les priorités des tâches avec le reste de l'équipe d'examen du projet.

4. Identifier un plan visant à accélérer le chemin critique.
5. Identifier d'autres chemins à risques qui sont juste derrière le chemin critique.
6. Vérifier la charge sur les ressources affectées au chemin critique.
7. Confirmer la disponibilité des ressources avec les gestionnaires.
8. Déterminer la partie du plan de projet qui nécessite plus de travail.
9. Repérer les éléments d'action.
10. Exécuter les raffinements de tâches.

Il est essentiel que le gestionnaire de projet ne se laisse pas berner par le pourcentage de réalisation. Il s'agit d'une fonction non linéaire qui n'est pas utile pour estimer la durée du travail restant à accomplir

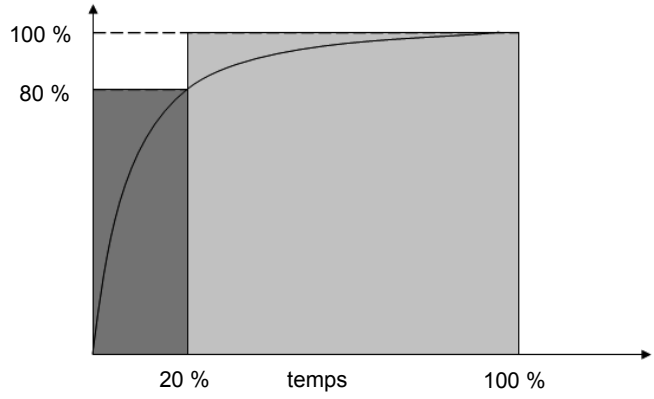


Figure 1.1 – Le dilemme du pourcentage de terminaison.

Gestion de projet proactive

Il faut un comportement extrêmement proactif pour terminer un projet à temps. Il faut être sûr de consacrer suffisamment de temps à la gestion du projet.

Parce que les circonstances évoluent, il faut une attention constante dans la gestion avec des mises à jour rigoureuses du planning du projet toutes les semaines.

La complexité du projet nécessite les bons outils pour faciliter le processus de prise de décision. L'identification et la gestion du chemin critique simplifient la définition des priorités.

2 Spécification de la conception

Ce chapitre explique l'intérêt de commencer tout projet par la rédaction précise de spécifications qui seront communiquées à toutes les équipes concernées. Il détaille la liste minimale des exigences qui doivent figurer dans la spécification fonctionnelle.

2.1 La communication est la clé du succès

Avoir une spécification complète et détaillée dans les débuts d'un projet permet d'éviter les faux départs et de réduire la probabilité de demandes de modifications d'ingénierie (ECO) tard dans le projet. Les modifications tardives de la spécification du circuit peuvent augmenter considérablement le coût d'un projet à la fois en termes de planning du projet et de coût du FPGA. Ce dernier point se produit lorsque des modifications importantes risquent d'entraîner la nécessité d'un composant FPGA plus gros.

Le but de la spécification est de communiquer clairement et précisément les informations.

Une autre façon de le dire est que les spécifications sont un moyen pour transmettre des informations entre les équipes et les personnes. Sans un cahier des charges complet, qui a été approuvé par toutes les parties concernées, un projet est sujet à des retards et à des modifications tardives dans les besoins, qui mènent toutes à des cycles de projet plus longs et des coûts de projet plus élevés. Un point clé de cette déclaration est l'« accord sur la spécification » qui implique qu'un processus soit en place pour l'examen de la spécification.

Un accord complet sur la spécification permet la coordination entre les différentes équipes travaillant sur le projet. Cela garantit que le produit livré est conforme aux spécifications fonctionnelles et répond aux besoins du client. Ceci facilite l'estimation précise des coûts de développement, des ressources et du planning du projet.

Une spécification solide permet le suivi cohérent des projets, qui conduira finalement à la livraison d'un produit de haute qualité. Elle sert aussi de référence pour la création de documents (et de leurs annexes) ou pour le support du produit. Toutes les spécifications doivent clairement identifier les modifications qui ont été apportées au cahier des charges. En outre, la spécification doit être intégrée dans le logiciel de contrôle de version.

2.1.1 Historique des révisions

Tableau 2.1. Exemple de page de contrôle des révisions

Version	Auteur	Date	Modification
0.9	psimpson	4-26-09	Examen initial
1.0	psimpson	5-11-09	Ajouts des informations temporelles du CODEC
1.1	aclarke	5-30-09	Modification de la carte des registres après examen avec les ingénieurs logiciels le 28 mai 2009
1.2	jjones	6-3-09	Ajout d'une section décrivant l'interface avec le processeur hôte.
1.3	psimpson	6-9-09	Mise à jour de l'interface avec le processeur hôte après une seconde réunion avec les ingénieurs logiciels le 4 juin

Les spécifications sont nécessaires à différents stades de la conception utilisant le FPGA, de la définition au processus de développement.

2.2 Spécifications fonctionnelles de haut niveau

La spécification fonctionnelle de haut niveau est créée et gérée par l'équipe d'ingénierie des systèmes. Ce document décrit les fonctionnalités de base du circuit implanté dans le FPGA, y compris l'interaction nécessaire avec l'interface logicielle et les interfaces entre le FPGA et d'autres composants de la carte. Ce document doit être examiné officiellement par le responsable de l'équipe de conception FPGA et le responsable de l'ingénierie du logiciel. Après l'examen, le document doit être mis à jour pour refléter les modifications demandées et répondre à toutes les questions soulevées au cours du processus d'examen. Ce processus est itératif jusqu'à ce que tous les problèmes aient été résolus et que l'équipe de conception du FPGA comprenne et accepte les besoins.

Un des problèmes de l'élaboration de la spécification fonctionnelle de haut niveau est de réussir à écrire les fonctionnalités dans un langage compréhensible. Soyons

honnêtes, la plupart des ingénieurs sont forts en mathématiques et en sciences, mais ne seront jamais des auteurs à succès.

Des spécifications exécutables aident à résoudre ce problème. Ce sont des modèles abstraits du système qui décrivent les fonctions du système final. Il s'agit essentiellement d'un prototype virtuel du système. La plupart des spécifications exécutables sont créées dans l'une des variantes de C (C, C++, SystemC). Ces langages sont suffisants pour la modélisation de la fonctionnalité désirée, mais ne couvrent pas des caractéristiques clés comme le fonctionnement temporel, la consommation et la taille du circuit. Celles-ci doivent être obtenues avec une spécification de haut niveau accompagnant la spécification exécutable. Le prototype virtuel à ce stade est le modèle du système et le banc de tests qui fait partie de la spécification exécutable. Cette spécification exécutable peut être utilisée tout au long du processus de développement afin de vérifier que l'implémentation détaillée satisfait aux besoins de la spécification exécutable. L'utilisation de SystemC, etc. comme moyen d'obtenir les spécifications pour la conception offre l'avantage que cette spécification peut être déduite des compromis de modélisation qui se produisent lors de l'exploration architecturale. Ceci est décrit plus en détail dans le chapitre 3 sur la modélisation du système.

Toutes les entreprises n'utilisent pas des spécifications exécutables durant le processus de conception avec FPGA, mais son utilisation est de plus en plus fréquente avec l'implémentation de systèmes plus complexes dans les composants FPGA.

2.3 Spécification fonctionnelle de la conception

L'équipe qui réalise le circuit utilisant le FPGA doit élaborer une spécification de conception détaillée qui représente les besoins de la spécification fonctionnelle de haut niveau. Cette spécification est de la responsabilité de l'équipe d'ingénierie de FPGA. Elle doit être examinée et approuvée par l'équipe de conception FPGA, les managers et les représentants des équipes d'ingénierie des systèmes et d'ingénierie logicielle. Ils doivent finaliser les spécifications concernant la fonctionnalité du circuit FPGA et détailler les interfaces avec le reste du système, logiciels compris.

Il est essentiel de s'entendre sur les détails des interfaces du FPGA avec les équipes de développement appropriées qui vont utiliser ces interfaces.

Prenez par exemple l'interface matériel/logiciel pour un circuit dans lequel un convertisseur A/D alimente le FPGA. Le FPGA à son tour fournit des données à un microprocesseur. La spécification du FPGA doit inclure l'interface vers le convertisseur A/D et être conçue pour éviter les pannes de fonctionnement, même dans des conditions particulières. Ne pas le faire peut entraîner des défaillances

fonctionnelles qui n'apparaîtront qu'au test final du système. Des tests de la carte peuvent montrer que le FPGA passe des données inutiles aux interfaces logicielles. Les ingénieurs logiciels ne sauront probablement pas comment interpréter ou déboguer ce problème. Cela peut augmenter la durée du test de la carte et dans le pire des cas conduire à reprendre le logiciel et/ou la conception du FPGA. Cela se traduira au final par un retard dans le planning.

2.4 Les grandes lignes de la spécification fonctionnelle

Dans cette section, nous allons détailler l'ensemble minimal d'exigences que doit contenir la spécification fonctionnelle :

1. **L'historique des modifications** – Un exemple de page de contrôle des modifications est montré en tableau 2.1. Il comprend la date des modifications, leurs auteurs et l'approbation des modifications.
2. **Le compte-rendu des réunions d'examen** – Cela doit inclure les détails sur toutes les réunions de modification des spécifications. Ces comptes-rendus doivent mentionner la date et le lieu de la réunion, les participants, le contenu de la réunion et les problèmes à résoudre pour obtenir l'approbation de la spécification.
3. **La table des matières**.
4. **La présentation des caractéristiques** – La vue d'ensemble des caractéristiques doit fournir le contexte d'utilisation du système dans lequel la caractéristique interviendra. Si la caractéristique est un sous-système du système final FPGA à concevoir, cette section doit décrire l'endroit où elle intervient dans le système global et son objectif, c'est-à-dire le problème qu'elle résout. La vue d'ensemble des caractéristiques doit également inclure un aperçu de haut niveau de la fonctionnalité requise.
5. **Les références sources** – Cette section doit décrire ce qui motive la demande de la fonctionnalité, par exemple, une spécification fonctionnelle de haut niveau, un besoin fonctionnel d'interface logicielle, etc.
6. **Le glossaire** – Le glossaire doit décrire tous les termes standards de l'industrie et les acronymes utilisés dans le document. Plus important encore, il doit le faire pour toute terminologie interne à l'entreprise utilisée dans le document. Il est étonnant de voir la quantité de temps perdu et la confusion due à l'utilisation d'une terminologie interne à l'entreprise. Beaucoup de nouveaux employés ou les employés d'autres groupes sont souvent embarrassés avant d'admettre qu'ils ne comprennent pas les mots « codés » dans les réunions de bilan, ce qui entraîne la confusion, des retards dans la prise de décision et entrave souvent la créativité.

7. **La description détaillée des caractéristiques** – C'est vraiment le cœur du document. Cette section doit comprendre une description de chacun des algorithmes utilisés, des détails sur l'architecture du circuit et de l'interface avec d'autres parties du circuit ou du système.
8. **Le plan de tests** – Le document doit se référer au plan de tests, ou au minimum établir la nécessité d'un plan de tests et être mis à jour lorsque le plan de tests existera.
9. **Les références** – Dans cette section, le document doit mentionner toutes les pièces justificatives à lire pour comprendre les spécifications fonctionnelles.

Après l'élaboration de la spécification détaillée de conception du FPGA, l'équipe d'ingénieurs créera un certain nombre de spécifications pour réexamen interne dans le département d'ingénierie. Elles comprennent notamment le plan de tests fonctionnels et le plan de tests questions-réponses. Les ingénieurs du projet élaboreront un plan d'ingénierie et un plan de tests fonctionnels pour la partie du circuit qu'ils implémenteront. Ceci devra être réexaminé dans le cadre du plan fonctionnel global. Cela garantit qu'il répond aux exigences globales de la conception du FPGA.

2.5 Les grandes lignes de la spécification des tests

1. **L'historique des modifications** – Un exemple de page de contrôle des modifications est montré en tableau 2.1. Il inclut la date des modifications, leurs auteurs et l'approbation des modifications.
2. **Le compte rendu des réunions de révision** – Cela doit inclure les détails sur toutes les réunions de modification des spécifications. Ces comptes-rendus doivent mentionner la date et le lieu de la réunion, les participants, le contenu de chaque réunion et les problèmes à résoudre pour obtenir l'approbation de la spécification.
3. **La table des matières**.
4. **La portée** – Elle donnera un aperçu des caractéristiques spécifiques que ce plan de tests couvrira. Si la couverture de tests a des parties communes avec le test de chaque sous-système, elle doit décrire en détail ce qui sera couvert dans ce plan de tests et citer les autres plans de tests.
5. **Les besoins du test** – Il doit détailler tout matériel spécifique, logiciel, outil de CAO électronique qui sont nécessaires pour terminer le test. Une partie doit comprendre les besoins particuliers pour l'initialisation du système.
6. **La stratégie de test** – Cela inclut les critères de réussite/d'échec.

   Est-ce que les résultats des tests nécessitent une vérification croisée avec d'autres sous-systèmes ?

   Est-ce que des tests existants seront réutilisés ou modifiés pour répondre aux besoins de ce plan de tests ?

Est-ce que les tests seront automatisés et si oui, comment le seront-ils ?

Comment les tests seront exécutés ? Un exemple serait un test de régression automatisé exécuté chaque nuit, ou un test manuel pour vérifier que les graphiques s'affichent correctement sur l'écran lorsqu'il est exécuté sur une carte de développement.

7. **Le plan d'automatisation** – Il est souhaitable d'automatiser autant que possible les tests. Cette section décrit comment y parvenir.
8. **L'exécution des tests** – Quelle est la durée prévue des tests ? Si les tests ne sont pas automatisés, quel est le délai prévu pour les tests à effectuer manuellement.
9. **La documentation du test** – Cette section doit inclure une description des cas de test. En pratique, l'infrastructure de test doit permettre d'isoler chaque test. Chaque cas de test doit donc avoir son propre répertoire de tests. La documentation doit détailler comment accéder aux résultats de la base de données des tests de régression. Cela suppose qu'un système de tests de régression a été établi. Ne pas établir un tel système signifie un projet voué à l'échec car il sera très difficile de contrôler la qualité du produit.

   La documentation du test doit également couvrir les procédures de test pour les cas où les sous-tests ne peuvent pas être automatisés. Dans ce cas, il est nécessaire de documenter la façon de tester manuellement la sous-fonction.

Au début des travaux de développement de la conception FPGA, le travail d'ingénierie doit comprendre des réunions régulières d'examen de la conception et de la vérification pour s'assurer qu'il n'y a aucune modification du plan. Ces examens seront l'occasion d'informer des modifications qui peuvent être nécessaires pour surmonter les problèmes d'implémentation et clarifier les ambiguïtés dans les spécifications. À la suite de ces réunions, les spécifications doivent être mises à jour et réexaminées. Si les modifications recommandées ont un impact sur la spécification fonctionnelle de haut niveau ou l'une des interfaces avec le FPGA, il doit y avoir des réunions de mise à jour formelles avec les personnels concernés pour valider les modifications.

En résumé, l'objectif principal de la spécification est de communiquer l'information entre les équipes afin que la conception réponde aux besoins et déterminer le personnel nécessaire pour obtenir les résultats voulus dans le temps imparti.

Les besoins de spécification fonctionnelle et de spécification du test seront déterminés par la politique de l'entreprise sur le respect des normes, par exemple, la conformité aux normes ISO 9001. Ce livre ne traite pas des détails sur la conformité ISO 9001. Une description détaillée de cette norme est disponible à l'adresse *www.iso.org*.

Lecture recommandée : *Writing Better Requirements* by Ian Alexander.

3 Modélisation système

Ce chapitre vous permet d'abord un rapide tour d'horizon des outils utilisés pour la modélisation système, avant de rentrer dans les détails de SystemC, largement utilisé pour la modélisation de systèmes mixtes matériels et logiciels.

3.1 Introduction

Les techniques utilisées pour la modélisation de systèmes varient d'un tableur Excel complexe à l'utilisation d'outils et de langages de modélisation système. Les langages les plus couramment utilisés sont C/C++, Lisa, UML, SystemC et Matlab. C/C++ permet de créer une spécification exécutable avec une exécution rapide. Les langages les plus largement utilisés dans la modélisation des conceptions de systèmes FPGA sont C / C++, SystemC et Matlab.

La modélisation est utilisée par tous les concepteurs. Dans la version la plus simple, les concepteurs effectuent une simulation RTL pour vérifier le fonctionnement de leur conception RTL au niveau des modules et de la conception complète. En 2013, un nombre croissant d'utilisateurs utilisent des techniques avancées de modélisation des systèmes dans le processus de conception de systèmes FPGA. Cela varie d'une modélisation complète du système en utilisant des modèles C ou le langage MathWorks de Matlab, à la modélisation système de parties clés de la conception, tels que le sous-système processeur pour écrire des pilotes logiciels et porter un système d'exploitation tel que Linux. Ce chapitre explique où les différentes classes de modèles peuvent être utilisées tout au long du processus de conception du système.

Matlab est très utilisé pour la modélisation des circuits de traitement du signal. Ceci est partiellement dû au fait que l'environnement de modélisation fourni par Mathworks est bien adapté aux applications ou systèmes de type traitement du signal. Les applications radar en sont un exemple. C'est aussi dû en partie au fait

qu'il y a un passage direct des modèles à l'implantation sur FPGA *via* les bibliothèques de modélisation Simulink. Mathworks fournit également une méthode de transfert de Matlab vers le HDL.

C/C++ est couramment utilisé pour la modélisation d'applications logicielles. Cependant, il est insuffisant pour les architectes qui doivent modéliser l'aspect matériel de la conception du système. C/C++ n'a pas de notion de temps. Les conceptions matérielles sont intrinsèquement concurrentes mais C/C++ n'a pas de moyen standard pour exprimer la concurrence. Il ne dispose pas non plus d'une façon d'exprimer les types de données du matériel comme le trois-états (*tri-state*). Ainsi, alors que C/C++ est bon pour la modélisation des systèmes logiciels, il ne peut fournir le niveau de modélisation nécessaire pour des conceptions matérielles ou des conceptions de systèmes mixtes matériel et logiciel.

SystemC a été conçu pour résoudre le problème de la modélisation de systèmes mixtes matériel et logiciels, tout en offrant les avantages de C/C++. Ceci est décrit plus en détail dans la section 3.3, « Qu'est-ce que SystemC ? »

Depuis la normalisation de SystemC, de nombreuses modélisations SystemC ont été créées par les concepteurs de systèmes avec FPGA. Il y a eu également de plus en plus d'offres d'outils de CAO électroniques utilisant SystemC. Compte tenu de cette situation, ce chapitre se focalise principalement sur la modélisation SystemC des systèmes à base de FPGA.

3.2 Définition de la modélisation système

La modélisation d'un système est généralement conçue comme étant une description exécutable du système permettant l'analyse et la mesure du comportement du système. Cette analyse peut se faire à différents niveaux d'abstraction. Le processus de modélisation du système permet le raffinement des modèles jusqu'à ce que l'implémentation finale de la conception soit réalisée et qu'il soit prouvé qu'elle répond aux exigences spécifiées dans le modèle de haut niveau.

La modélisation système permet à la fois la conception et l'analyse de l'architecture du système. Un exemple est l'interface matériel/logiciel. Un exemple de modélisation système d'une conception de système avec FPGA est le processus de partitionnement entre la conception du sous-système processeur et la logique FPGA. La modélisation système peut être utilisée pour déterminer ce qui doit être implanté dans le sous-système processeur et ce qui devrait être implanté avec les composants logiques du FPGA. Cette décision doit dépendre des goulots d'étranglement de performance déduits de la spécification. Cela nécessite la modélisation du processeur, des interconnexions, de la mémoire et des accélérateurs. Pour modéliser un tel système,