Conception et modélisation d'un émulateur de réseaux de capteurs sans fils

Nadim Nasreddine

Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National des Sciences Appliquées de Toulouse (INSA Toulouse)

Discipline ou spécialité :

Conception des Circuits Microélectroniques et Microsystèmes

Présentée et soutenue par :
Nasreddine Nadim

le : 11/07/2012

Titre :

Conception et modélisation d'un émulateur de réseaux de capteurs sans fils

---

Ecole doctorale :
Génie Electrique, Electronique et Télécommunications (GEET)

Unité de recherche :
LAAS-CNRS

Directeur(s) de Thèse :
FOURNIOLS Jean Yves : professeur à l'Institut National des Sciences Appliquées de Toulouse
BOIZARD Jean louis : maître de conférences à l'université de Toulouse I

Rapporteurs :

OUSTEN Yves, Professeur IMS Bordeaux — Rapporteur
LATORRE Laurent, HDR Université de Montpellier — Rapporteur

Membre(s) du jury :

CAMPS Thierry, Professeur Université Paul Sabatier, Toulouse — Examinateur
SOTO-ROMERO Georges, MDCF Université de Franche-Comté, Besancon — Examinateur

Résumé

La conception des systèmes embarqués devient un processus de plus en plus complexe et multidisciplinaire. Elle nécessite que se développent des méthodes et des outils de conception, vérification et de réutilisation assurant un travail coopératif sans faute. Cette exigence est présente dans le développement en cours des « réseaux de capteurs » destinés à la surveillance et au diagnostic dans des domaines très divers (Systèmes de « Health monitoring » & « ambiant intelligence ») qui nous concernent ici.

Les travaux que nous présentons dans ce mémoire appliquent les approches descendantes classiques de la conception des systèmes. Au terme de l'étape de conception, avant de procéder au choix des technologies et à la fabrication du prototype matériel, une phase de simulation et de vérification est nécessaire : c'est une première contribution de notre travail de proposer une démarche de modélisation logico-temporelle multi niveaux. Vue la particularité et la complexité des systèmes de télécommunications et de contrôle, un processus spécifique de simulation et de validation doit être envisagé.

La simulation d'un système global complexe peut s'appuyer sur un ensemble de plusieurs simulations séparées correspondant à chaque composition et chaque élément du système dans des environnements divers. Afin d'améliorer sa fiabilité on essaye d'avoir une simulation regroupant l'ensemble des simulations partielles. Cette simulation globale, idéalement faite dans un environnement unique, devrait couvrir les différents domaines et tous les éléments composant le système. C'est une procédure performante mais lourde et consommatrice de temps et c'est un des facteurs les plus exigeants dans la détermination du « time to market ».

Afin d'accélérer ce processus de conception, un environnement de simulation rapide et performant peut s'avérer indispensable pour le concepteur; une simulation rapide et performante met en œuvre des modèles fonctionnels simplifiés, comportementaux qui représentent les fonctions considérées de haut niveau, sans entrer dans les détails des équations physiques de chaque élément de cette fonction, d'où sa dénomination : « fonctionnelle comportementale ». Pour la rendre performante, c.à.d. capable d'apporter des éléments décisifs dans les choix architecturaux et technologiques, les modèles comportementaux des composants élémentaires du système doivent être capables de remplacer les éléments réels dans leurs influences et réponses à tous les phénomènes influents: perturbations, affaiblissements, retards...

Nos travaux de thèse visent à contribuer à cette approche méthodologique : ils traitent, du point de vue applicatif, le développement d'un émulateur de **réseau de capteurs autonomes communicant sans fil d'instrumentation**. Pour ce faire deux types de simulateurs ont été étudiés (développés) basés sur l'étude de la **qualité de communication** évaluée par la possibilité d'introduire des erreurs dans la communication représentée par le **taux d'erreurs par bit (BER)**:

- le premier est un simulateur « software » basé sur la création de modèles comportementaux, décrits en langage VHDL-AMS, représentant chaque élément du réseau. Ces modèles sont ensuite interconnectés entre eux dans l'environnement "SystemVision" de "MentorGraphics" suivant une architecture à tester pour la simuler.
- le deuxième est un simulateur hardware basé sur la création d'un modèle logique comportemental, décrit en langage VHDL synthétisable, de chaque élément du système. Une fois l'architecture du réseau définie (câblage des éléments entre eux), celle-ci est compilée puis synthétisée dans l'environnement approprié et enfin le fichier de programmation obtenu est téléchargé sur un composant FPGA cible dans lequel s'effectuera la simulation d'où la dénomination de « hardware ». L'environnement de développement utilisé dans nos travaux est la suite "Quartus" de la société "Altera". Une fois téléchargée sur un FPGA, des modifications peuvent être faites sur l'architecture de manière dynamique via l'outil « In System Memory Content Editor », ceci afin de pouvoir simuler plusieurs configurations possibles. Ce même outil est également utilisé pour récupérer les résultats des simulations.

Nous montrons, en mettant en œuvre ces simulateurs sur des exemples d'applications (miniaturisation du projet SACER de surveillance de structures d'avion), comment nous pouvons assurer les fonctions de vérification des choix architecturaux et dynamiques temporels et guider l'étape essentielle des choix technologiques, dans un environnement à la fois fonctionnel et matériel. Ce guidage s'opère par des optimisations calculées à partir de « fiches techniques » associées aux modèles fonctionnels.

Abstract

Embedded systems design becomes a complex and multidisciplinary process. This requires developing methods and tools for design, verification and reuse ensuring a cooperative work. These requirements are present in the development of "sensor networks" for monitoring and diagnosis in very different fields ("Health monitoring" & "ambient intelligence" systems) that concern us here.

The work presented in this thesis applies traditional top-down approach in system design. At the end of the design stage, before the technology selection and fabrication of prototype, a phase of simulation and verification is needed: this is a first contribution of our work; to propose a logical modeling approach Multi-temporal levels; View the particularity and complexity of telecommunications systems and control, a specific process simulation and validation should be considered.

The simulation of a complex system can be based on a set of several separate simulations for each composition and each element of the system in various environments. To improve reliability we try to have a simulation containing all the partial simulations. This global simulation, ideally made in a single environment, should cover the different areas and all the system components. This procedure is a powerful but cumbersome and time-consuming and it is critical to determine the "time to market."

To accelerate the design process, a fast and efficient simulation environment may be essential for the designer; a fast and efficient simulation implements simplified functional models, which represent the high level behavioral functions, without entering into details of the physical equations of each element. To make it efficient, i.e. capable of providing crucial elements in the architectural and technological choices, the behavioral patterns of the elementary components of the system must be able to replace the real elements in their influences and responses to all the influential phenomena: disruptions, impairments, delays...

Our thesis work aims to contribute to this methodological approach: they handle the development of an emulator for **wireless autonomous sensor network for instrumentation use.** To do, two types of simulators have been studied (developed) based on the study of the **communication quality** evaluated by the possibility of introducing errors in the communication represented by the **bit error rate (BER)**:

- The first one is a software simulation based on the creation of behavioral models, described in VHDL-AMS, representing each network element. These models are then interconnected in "SystemVision" environment of "MentorGraphics" corresponding to a tested architecture for simulation.
- The second is a hardware simulator based on the creation of a behavioral logic model, described in synthesizable VHDL, representing each system element. Once the network architecture defined (wiring between the elements), it is compiled and synthesized in the proper environment and finally got the programming file, it will be downloaded to an FPGA target in which the simulation is executed, hence the name of "hardware". The development environment used in our work is "Quartus" from "Altera". Once downloaded to an FPGA, changes can be made on the architecture dynamically via the tool "in System Memory Content Editor", in order to simulate several possible configurations. This same tool is also used to retrieve the results of simulations.

We show, by using these simulators on sample applications (miniaturization of SACER project monitoring aircraft structures), how we can ensure the verification functions of architectural choices and temporal dynamics and guide the essential step of technological choices. This is achieved by guiding optimizations calculated from "sheets" associated to the functional models.

Remerciements

Cette thèse vient couronner 26 ans passés sur les bancs des études. Il parait difficile de résumer en quelques lignes mes sentiments et mes souvenirs qui viennent gambader dans ma tête mais ce qui est revenu le plus tout au long de ces 26 années c'est l'impatience avec laquelle j'attendais le jour où je terminerais mes études et le stress en attendant les examens et leurs résultats. Enfin ce jour a eu lieu…

Ce jour ne serait jamais arrivé sans l'intervention et l'aide de plusieurs personnes que je promets de ne jamais oublier.

Je remercie M. ARLAT Jean, SANCHEZ Jean Louis, directeurs du LAAS qui m'ont permis d'effectuer ma thèse au sein du LAAS.

Messieurs FOURNIOLS Jean Yves et BOIZARD Jean Louis, mes directeurs de thèse, vous m'avez fait l'honneur de m'accueillir dans votre équipe avec la plus grande bienveillance et me témoigner votre confiance. Il m'a été permis d'apprécier vos grandes qualités tant humaines que professionnelles qui ont largement contribué à finir mon travail. En témoignage de ma profonde reconnaissance, veuillez, trouver mes remerciements les plus sincères.

J'exprime aussi ces remerciements à tous ceux qui m'ont aidé durant la thèse surtout l'équipe du projet SACER pour leur patience et pour leurs réponses à toutes mes interrogations.

Sans les oublier, je remercie également tous mes collègues du laboratoire, surtout les membres du groupe N2IS, et mes collègues dans tous mes étapes d'apprentissage ainsi que tous mes enseignants.

Un grand merci aux rapporteurs, qui ont consacré de leur temps pour lire ma thèse et l'évaluer, ainsi qu'aux membres du jury de ma soutenance.

Mais tout ce paragraphe n'a aucune signification sans quatre personnes. Ces quatre personnes sont ma mère, mon père, mon frère et ma sœur, qui m'ont accompagné durant toute ma vie et dans toutes les situations, et que le remerciement est très peu pour ce qu'ils ont fait.

Sommaire

Introduction générale

Dans sa plus grande généralité, le terme « système » désigne un ensemble d'unités réelles ou abstraites qui interagissent afin de servir un objectif commun. L'environnement système peut être élargi pour contenir tout ce qui est en relation avec le système lui même c'est-à-dire les interactions du système avec l'environnement.

La modélisation sert à créer des modèles de chaque élément de l'environnement système dans le but d'interpréter une expérimentation et de prédire comment le système évolue et comment il se comporte dans une situation donnée. Cette modélisation permet de valider un fonctionnement par simulation et d'affiner le cahier de charges.[1]

A partir des années 40, où les outils de calcul numérique commencent à apparaître et se développer, et durant les années 60 où les circuits numériques ont vu le jour, les études des systèmes sont devenues plus faciles et plus complexes à la fois. C'est en profitant de ces outils que le calcul numérique devient un fondement de l'étape de conception et de la mise en œuvre, sur ordinateurs ou, d'une façon plus large, sur processeurs numériques intégrés.

Pour garantir au concepteur la fonctionnalité, les performances, une qualité de service et une sécurité de fonctionnement, les recherches dans le but de concevoir et disposer d'outils grâce auxquels on pourra concevoir un système, commencent à demander des efforts considérables tant théoriques qu'à caractère appliqué.

De même on assiste à une croissance sans précédent de la Modélisation et de la Simulation (M&S) dans les différents secteurs de l'ingénierie industrielle pour la conception des nouveaux systèmes afin d'étudier le plus largement possible leurs caractéristiques, ce qui encourage de plus en plus le renforcement des méthodes de développement des outils de (M&S).

Pour notre part, nous traitons ici du développement de systèmes de capteurs autonomes communicants sans fils dédiés à l'instrumentation ce qui suppose de garantir une transmission sans perte d'informations. Dans ce contexte, un système est un équipement physique : son flot de conception fait appel à un ensemble de processus relatifs à la conception, la production, la vérification, l'exploitation…

guidés par des recommandations générales, en l'occurrence dans notre travail à celles de IEEE/EIA-632. Nous appliquons ces recommandations à la modélisation/ simulation des réseaux de capteurs, en nous appuyant sur le cas d'un exemple pratique : le projet SACER de surveillance de structures en aéronautique. Le développement rapide du concept d'intelligence ambiante est caractérisé par le déploiement et l'interconnexion de réseaux de communications numériques au service d'applications nombreuses telles que sur les Personnes, l'Habitat, les Transports, l'Energie et l'Environnement… Etc. Un des moteurs de cette évolution est la conception-déploiement de microsystèmes autonomes communicants sans fil : Ils sont dans le prolongement des dispositifs microélectroniques, composés de capteurs, d'électronique de traitement de données et sont reliés aux réseaux par des interfaces de communication.

Le LAAS aborde ces questions de plusieurs angles de vue : les aspects systèmes [2], les aspects microsystèmes et les aspects réseaux, en y intégrant des perspectives applicatives : La surveillance des systèmes est une des applications probables du développement d'une intelligence ambiante ; **l'objectif est de créer un niveau de supervision des systèmes complexes dont les missions seraient :**

- de détecter des comportements systèmes anormaux et de créer des alarmes qui pourront être mises à profit pour des interventions préventives ou réparatrices,
- de faire un diagnostic de l'origine probable du défaut détecté,
- de contribuer par la collecte des données de test, autotest, diagnostic,.., à une mise au point accélérée des systèmes de plus en plus complexes.

Nous nous intéressons, dans ce mémoire, aux **systèmes de surveillance** s'appuyant sur des réseaux de capteurs repartis dans le système à surveiller : les données sont collectées sur un ordinateur central dans lequel opèrent des logiciels de surveillance du bon fonctionnement de l'installation et réalisent des procédures de diagnostic : détection de panne et identification des causes. C'est une classe très importante de systèmes qui est amenée à se généraliser comme une option utile dans la gestion de la sureté de fonctionnement des grands systèmes complexes : le LAAS a

une expérience déjà ancienne de ces surveillances de systèmes notamment sur des ensembles mettant en cause l'opérateur humain. C'est ainsi que des travaux ont été conduits sur la « détection de l'hypovigilance du conducteur automobile » [3] et sur la « surveillance des personnes âgées » [4]. **Notre intérêt porte plus précisément dans ce mémoire, sur la surveillance de structures mécaniques (Structural Health Monitoring), dans des applications liées à l'Aéronautique.**

Chacun connaît l'intérêt qu'il y a à alléger les structures avion et à rechercher des matériaux métalliques ou composites légers, rigides, fiables… Ces matériaux nouveaux, quels que soient leurs rôles, seront soumis à des contraintes répétées et donc à des effets de fatigue dont il faut :

- caractériser les effets pour pouvoir les qualifier,
- suivre l'évolution de leurs propriétés durant le « cycle de vie » pour éventuellement effectuer des travaux de maintenance.

Pour réaliser ces objectifs, différents types de systèmes de surveillance devront être conçus et réalisés : si l'on considère le point de vue du système de communications, il peut s'agir de surveillance long terme ne nécessitant que quelques mesures journalières sans grandes exigences de communications ; il peut s'agir, à l'opposé, de mesures nombreuses à gérer en temps réel : c'est le cas du projet SACER qui apporte une réponse d'instrumentation distribuée par un réseau de capteurs.

Le projet SACER suppose de déployer des centaines de capteurs sur une aile d'avion pour en étudier le comportement dynamique en temps réel… On peut imaginer que cette ambition : vitesse, débit, complexité, plages d'utilisation (sur sol, durant les vols,)… pose des problèmes très difficiles à résoudre à tous les niveaux d'un système de surveillance :

- les nœuds de réseaux constitués de microsystèmes intégrés, miniaturisés et énergétiquement autonomes.
- le réseau d'échange des données,
- les traitements algorithmiques.

Ce projet est un projet coopératif entre plusieurs équipes académiques et industrielles. Notre tâche a été essentiellement d'apporter au concepteur un outil qui lui permette de définir l'architecture d'un réseau, d'en valider le choix et d'en faire l'optimisation par simulation logico-temporelle. Au delà de la recherche d'une solution ponctuelle au problème particulier posé par le projet SACER, nous nous sommes efforcés d'une part, d'ouvrir cet outil de simulation à d'autres applications de RCSF, et d'autre part de tirer des enseignements généraux de notre expérience sur les méthodes et les outils de la conception-vérification de tels systèmes. De ce dernier point de vue, notre travail se situe dans le cadre structuré de l'ingénierie des systèmes électroniques : il suppose tout d'abord un travail de spécification puis de formalisation de ces spécifications sur lequel nous ne reviendrons que peu dans la présentation de nos résultats. Par contre, nous nous attacherons à montrer l'importance de la simulation logico-temporelle : d'abord pour valider les choix architecturaux puis pour faire des évaluations plus approfondies sur le fonctionnement prévisionnel du système, enfin pour construire une approche simple mais efficace de la transition « meet in the middle » lorsque l'on veut passer du stade de la logique fonctionnelle au prototypage matériel.

Notre manuscrit est organisé en trois chapitres :

Le premier chapitre décrit les réseaux de capteurs avec une vision historique sur leurs développements, quelques définitions de la syntaxe rencontrée dans ce domaine ; il introduit le projet SACER qui nous a servi de guide et de démonstrateur ; il précise notre plan de travail et met l'accent sur le rôle majeur de la simulation ; il propose une analyse de l'état de l'art sur les simulateurs existants; il montre les insuffisances des outils existants dès qu'il s'agit de considérer les performances du réseau de transmission et leur optimisation.

Le deuxième chapitre est consacré à la description des simulateurs que nous avons développés, le premier est basé sur le langage de description VHDL-AMS, le deuxième est décrit en langage VHDL synthétisable. Il présente quelques caractéristiques concernant la description des modèles, la mise en œuvre et les performances obtenues.

Le troisième chapitre présente une application des simulateurs sur un réseau de capteurs pour converger vers une solution en utilisant les 2 simulateurs et en

comparant leurs résultats, leurs coûts de fonctionnement et les aspects temporels qui sont un des verrous de toute simulation. Puis on présente les résultats de simulation du réseau SACER avant de généraliser la simulation sur d'autres possibilités et de conclure sur l'extrapolation de la méthode à d'autres applications.

Ces différents chapitres sont analysés et comparés aux plans méthodologiques et applicatifs : Ils seront suivis d'une conclusion générale sur nos travaux et ouvriront quelques pistes nouvelles de développements.

Références

[1] Laurent LATORRE, Evaluation des techniques microélectroniques contribuant à la réalisation de microsystèmes : application à la mesure du champ magnétique, thèse, UNIVERSITE MONTPELLIER II, Juin 1999.
[2] J.C.HAMON, Méthodes et outils de la conception amont pour les systèmes et les microsystèmes, Doctorat, Institut National Polytechnique, Toulouse, 1er Février 2005.
[3] A.SANTANA-DIAZ, Conception d'un système de détection de la baisse de vigilance du conducteur automobile par l'utilisation des ondelettes et l'apprentissage statistique, Doctorat, Université Paul Sabatier, Toulouse, 6 Janvier 2003.
[4] Youssouf ZATOUT, Conception et évaluation de performances d'un réseau de capteurs sans fil hétérogène pour une application domotique, Doctorat, Université Toulouse 2 Le Mirail, 2011.

Chapitre 1 : Conception des systèmes de surveillance et simulation des réseaux de capteurs

Chapitre 1. Conception des systèmes de surveillance et simulation des réseaux de capteurs

Sans nous en rendre compte et compte tenu de l'évolution progressive des techniques, nous utilisons actuellement de nombreux systèmes répartis : ces systèmes électroniques ont été introduits dans de nombreux domaines d'applications tels que l'automobile, l'avionique, les systèmes multimédia, les appareils électroménagers ou bien les terminaux de communication sans fil. Parmi les applications de ces systèmes on trouve des applications dans le domaine de la surveillance et de la détection que nous allons définir dans les prochains paragraphes et qui font partie de notre périmètre de travail.

1.1 Conception et simulation

1.1.1 Besoin : systèmes et systèmes de surveillance

La complexité croissante des systèmes multiplie les problèmes de fiabilité et de sureté de fonctionnement. Différentes techniques peuvent être mises en œuvre pour répondre à ces problèmes dès la conception même des systèmes : soit par la mise en œuvre de redondances, soit par le choix des technologies appropriées, … **Une approche qui tend à se généraliser est d'associer au système opérationnel un système de surveillance** agissant comme un superviseur, capable de détecter des comportements anormaux, de les signaler de manière à automatiser le lancement de fonctions redondantes, d'anticiper les opérations de réparations et d'accélérer les procédures de maintenance et d'entretien.

Ainsi un processus de surveillance consiste à :

- prélever des mesures réparties dans le système à surveiller
- collecter les données correspondantes dans une base de données à laquelle peuvent être associés des algorithmes de traitement et de diagnostic
- générer éventuellement des alarmes ou initier des interventions de corrections du fonctionnement du système.

Les systèmes multi-capteurs sont actuellement en train de devenir un domaine de recherche d'intérêt croissant avec des perspectives ouvertes grâce à la mise en œuvre de « réseaux sans fil » : ils permettent une installation rapide de systèmes de surveillance sous la forme de microsystèmes bientôt énergétiquement autonomes, communicants et répartis sur tous les points sensibles du système à surveiller ; ils sont une composante importante de ce que l'on appelle l' « **intelligence ambiante** ».

Beaucoup de secteurs économiques sont concernés : l'Habitat, les Transports, la Santé, l'Environnement, ... et ceux-ci pourront énormément tirer parti de ces dispositifs innovants. Pour ce qui nous concerne, nous traiterons ici d'une classe particulière de systèmes de surveillance : la surveillance des structures avionotiques, que nous illustrerons par le projet SACER.

1.1.2 La problématique de la conception

La conception d'un système de microsystèmes répartis chargé de la surveillance de structures aéronautiques et considérée dans sa généralité, entre dans le domaine de la conception des systèmes complexes, hétérogènes et répartis :

- le système doit être conçu en relation avec toutes les autres contraintes du « système avion », qu'il s'agisse d'un réseau de surveillance utilisé dans la mise au point du prototype ou d'un réseau qui, probablement, s'imposera à terme, parmi les techniques de surveillance et de maintenance des appareils.
- il associe des technologies matérielles très diverses: électronique, énergie, assemblages.., et logicielles dans le traitement des données et dans les étapes de diagnostic (détection de pannes),
- il doit répondre à des exigences particulièrement strictes du point de vue sécuritaire, de la gestion d'énergie, de la miniaturisation, …

L'approche de conception relève donc de **l'ingénierie système, laquelle propose des méthodes et des outils** dont la mise en œuvre doit :

- D'une part, assurer le concepteur de la conformité de ses propositions par rapport au cahier des charges,

- D'autre part, faciliter la coordination entre toutes les contributions de conception dans le développement d'un système complexe.

Notre contribution se situe volontairement dans cette logique de conception, en s'appuyant sur les normes IEEE-EIA-632 de l'ingénierie système [1]: nous nous situons dans le versant descendant du cycle en V : spécifications, architecture, vérifications. Nous reviendrons sur ces différentes étapes mais nous pouvons anticiper que notre contribution majeure se situe sur l'étape de **vérification par simulation logico-temporelle** et sur la manière d'aborder l'étape suivante de choix des technologies basée sur une approche « meet in the middle ».

La modélisation et la simulation

La modélisation d'un processus, d'un concept ou d'un système du monde réel, consiste à créer une association de fonctions et d'équations, appelée modèle, qui sert comme une approximation, une représentation ou une idéalisation de certains des aspects de sa structure, de son comportement, de sa fonction, ou de ses caractéristiques. [2]

Cette modélisation peut être utilisée pour représenter :

– le système en cours de développement.

– l'environnement dans lequel le système opère.

La modélisation est motivée par la volonté d'avoir des informations à propos du système et de son comportement, avant que des ressources conséquentes ne soient engagées dans la conception, la construction ou le test du système. La simulation est le fait de chercher une solution analytique du modèle, mais cela n'est pas toujours faisable à cause des limitations du formalisme de modélisation, de la complexité du système modélisé ou du système à simuler. Dans ce cas, on peut exécuter le modèle dans le temps et ainsi tirer des conclusions à propos du système réel en observant le comportement dynamique du modèle. Dans ce contexte, « simulation » signifie « expérimentation » avec un modèle.

Une simulation est aussi une technique pour le test, l'analyse ou l'entrainement dans laquelle des systèmes réels ou des modèles qui représentent ces systèmes réels

sont utilisés. Dans [3], les auteurs définissent une simulation comme un modèle qui se comporte comme un système donné lorsqu'il est soumis à un ensemble de stimuli. [4]

Rôle du simulateur

Nous verrons que le périmètre de notre étude concerne le développement d'un outil de simulation permettant de prévoir les performances d'un réseau de capteur sans fil, valider son organisation topologique et vérifier son fonctionnement dans son environnement opérationnel. Ceci est obtenu par :

- La comparaison entre les concepts/solutions candidats répondant aux problématiques posées et les concepts et options qui offrent le meilleur fonctionnement.
- Le développement de ces concepts pour consolidation avant l'engagement du développement du système.
- La prédiction des performances en identifiant les exigences qui ne sont pas satisfaites ou qui seront difficiles à satisfaire et le risque associé à la non-satisfaction d'une exigence.
- La validation des tests en permettant de les planifier, d'identifier des problèmes a priori et de minimiser les surprises.

L'objectif du simulateur est donc la modélisation de l'environnement opérationnel dans lequel chaque élément qui compose le réseau va fonctionner. Cela revient à modéliser d'une part les phénomènes internes aux composants ainsi que ceux liés à l'environnement et qui influencent le fonctionnement global du système. Cette simulation permet également de réduire les tests de validation sur maquettes expérimentales et favorise l'exploration architecturale.

1.2 Intelligence ambiante et réseaux de capteurs

1.2.1 Les réseaux de capteurs sans fils et la problématique de l'intelligence ambiante

Comme il a été vu dans le paragraphe précédent, la simulation d'un réseau de capteurs prenant en compte son mode de transmission est l'élément clef de la conception d'un tel système si l'on veut le dimensionner correctement et guider l'étape des choix technologiques.

Du point de vue des technologies, nous introduirons succinctement le rôle des technologies microsystèmes qui apportent la miniaturisation souhaitable aux installations et dans une certaine mesure participent à l'autonomie énergétique.

Un réseau de capteur est un système complexe multidisciplinaire dont l'élément de base est le nœud, que l'on peut considérer dans les approches technologiques actuelles, comme une association d'au moins quatre microsystèmes : celui de captage, celui de traitement, de transmission et de contrôle de l'énergie. Selon le domaine d'application, il peut également contenir des microsystèmes supplémentaires tels que des dispositifs de localisation (GPS) ou générateurs/récupérateurs d'énergie embarqués (cellule solaire)…

Les capteurs intègrent plusieurs sous-systèmes : le récepteur (reconnaissant l'analyte) et le transducteur (convertissant le signal du récepteur en signal électrique). Ce signal électrique analogique est ensuite converti par le sous-système de conversion Analogique/Numérique. Ce dernier transforme ces signaux en un signal numérique compréhensible par le microsystème de traitement embarqué.

Le traitement s'appui sur un processeur généralement associé à une petite unité de stockage. Il fonctionne à l'aide d'un système d'exploitation spécialement conçu pour les micro-capteurs. Il exécute des opérations de traitement des données et des opérations pour les protocoles de communications, lesquels sont nécessaires pour la communication du nœud avec les autres éléments du réseau. Il peut aussi analyser les données captées pour alléger la tâche du concentrateur.

La transmission gère toutes les émissions et réceptions des données sur un *medium « sans-fils »*. IL comprend des circuits de modulation, démodulation, filtrage et multiplexage.

Le contrôle de l'énergie : Le nœud de réseau dispose d'une ressource énergétique (généralement une batterie). Cet aspect fait souvent de l'énergie la ressource la plus précieuse d'un réseau de capteurs, car elle influe directement sur la durée de vie des micro-capteurs et donc du réseau entier.

Le contrôle de l'énergie constitue donc une fonction essentielle. IL doit répartir l'énergie disponible aux autres modules de manière optimale (par exemple en réduisant les dépenses inutiles et en mettant en veille les composants inactifs). Ce microsystème peut aussi gérer des systèmes de rechargement d'énergie à partir de l'environnement via des cellules photovoltaïques par exemple. [5, 6, 7]

1.2.2 L'état des pratiques dans la mise en œuvre des réseaux de capteurs :

Dans la vie courante, l'utilisation des capteurs sans fil est de plus en plus exigée pour la supervision et la sécurité. Les industriels proposent alors des capteurs sans fil qui peuvent renseigner l'utilisateur sur l'évolution de différentes grandeurs physiques. Ces capteurs peuvent être embarqués et reliés ensemble pour former un réseau sans fil, se basant sur des protocoles pour communiquer et pouvant disposer de programmes de diagnostic.

Un réseau de capteurs sans fil, est constitué d'un grand nombre de nœuds, comprenant un ou plusieurs capteurs autonomes qui permettent la récolte et la transmission des données environnementales à un récepteur final appelé concentrateur, c'est un réseau ad hoc[1]. La répartition géographique de ces nœuds n'est pas obligatoirement prédéterminée : ils peuvent être aléatoirement dispersés dans une zone de mesure correspondant au terrain d'intérêt pour le phénomène surveillé. [8]

[1] Les **réseaux ad hoc** (en latin : « qui va vers ce vers quoi il doit aller », c'est-à-dire « formé dans un but précis », telle qu'une *commission ad-hoc*, formée pour régler un problème particulier) sont des réseaux sans fil capables de s'organiser sans infrastructure définie préalablement.(wikipedia)

Les réseaux de communication sans fil ont connu un succès croissant et un développement important au sein des communautés scientifiques et industrielles. Grâce à ses divers avantages, cette technologie a pu s'imposer comme un acteur incontournable dans les architectures de réseaux actuelles. La communication radio offre en effet des propriétés uniques, qui peuvent être résumées en trois points : la facilité du déploiement, la simultanéité de l'information et le coût réduit d'installation. Au cours de son évolution, les applications du domaine sans fils se sont répandues sur plusieurs domaines, telles que : les réseaux cellulaires, les réseaux locaux et autres. Durant cette dernière décennie, un nouveau domaine d'application a commencé à occuper ce domaine : les réseaux de capteurs sans fils. Ce type de réseaux résulte d'une fusion de deux pôles de l'informatique moderne : les systèmes embarqués et les communications sans fils.

Un réseau de capteurs sans fils (RCSF), ou "Wireless Sensor Network" (WSN), se base sur des capteurs regroupés et associés à des unités de traitements embarquées pour constituer des "nœuds", lesquels communiquent les mesures effectuées et reçoivent les configurations via des liens sans fil. Le but général d'un RCSF est la collecte d'un ensemble de paramètres de l'environnement entourant les nœuds, telles que la température, la pression, les mouvements, les chutes, les oscillations, la déformation…, afin de les acheminer vers des points de traitement, d'analyse et d'affichage. Les RCSF sont souvent considérés comme étant les successeurs des réseaux ad hoc. En effet, les RCSF partagent avec les MANET (Mobile Ad hoc NETworks) plusieurs propriétés en commun, telles que l'absence d'infrastructure et les communications sans fil. La différence clé la plus importante entre les deux architectures est le domaine d'application. Vu leur réalisme, leur apport concret et les bénéfices qu'il est possible de cumuler, les RCSF ont commencé à avoir un grand succès après avoir attiré un nombre croissant d'industriels, contrairement aux réseaux MANET, qui sont restés dans un domaine d'application très limité. En effet, dans divers domaines industriels et activités de la société, on peut avoir besoin de suivre de façon continue l'évolution d'un environnement donné, et l'absence de fils de connexion facilite la répartition et l'installation. Les applications des RCSFs couvrent presque toutes les domaines de la vie quotidienne : les processus industriels, les applications militaires de suivi, le monitoring d'habitat, le contrôle des structures, ainsi que l'agriculture de précision ne sont que des exemples d'applications possibles.

Grâce à ce potentiel riche en applications, les RCSF on su attirer de grandes entreprises à travers le monde, telles que IBM, Sun, Intel et Philips. Malheureusement, les RCSF ne sont pas parfaits ! A cause de leur faible coût et leur déploiement dans des zones parfois hostiles, les nœuds sont assez fragiles et vulnérables à diverses formes de défaillances : cassure, faible énergie, ... etc. Ces problèmes rendent les RCSF des systèmes à fragilité innée, ce qui doit être considéré comme une propriété normale du réseau. [9]

1.2.3 Les réseaux de capteurs sans fils (RCSFs) : les principes

Les débuts des réseaux de capteurs sans fils reviennent aux années 1990. C'est en profitant des progrès des techniques « sans-fils » (Wireless), que certaines applications dans les domaines de l'environnement et de l'industrie, on vu le jour. Avant cette date seules quelques balises radio, pour acheminer les données d'un capteur au *contrôleur central,* n'utilisaient pas un câblage coûteux et encombrant. Aujourd'hui, les réseaux de capteurs sans-fils sont employés pour récupérer ces données intéressantes presque dans tous les domaines.

Les données captées par les nœuds sont envoyées au destinataire "point de collecte" et appelé concentrateur (nœud-puits ou *sink*). Ce dernier peut être connecté à l'utilisateur du réseau via un système de surveillance placé à proximité ou situé à distance via un moyen de communication type internet, transmission radio, ou satellite... Le concentrateur peut envoyer des requêtes ou des configurations au réseau ou à une partie, précisant par exemple la nature et les paramètres des données requises.

Les progrès conjoints de la microélectronique, microtechnique, des technologies de transmission sans fil et des applications logicielles ont permis de produire à coût raisonnable des micro-capteurs de quelques millimètres cubes de volume ainsi que des nœuds flexibles et planaires non encombrants susceptibles de fonctionner en réseaux. Ils intègrent :

- une unité de captage chargée de mesurer des grandeurs physiques (chaleur, humidité, vibrations, rayonnement...) et de les transformer en grandeurs numériques.
- une unité de traitement informatique et de stockage de données.

- un module de transmission sans fil (Wireless).
- des modules de gestion de l'énergie, convertisseur DC-DC, chargeurs (cellules photovoltaïques, micro turbine, condensateurs piézoélectrique).

Ces nœuds sont donc de véritables systèmes embarqués. Le déploiement de plusieurs d'entre eux en vue de collecter et transmettre des données environnementales vers un ou plusieurs points de collecte, d'une manière autonome, forme un **réseau de capteurs sans fil** (RCSF).

Un RCSF est composé d'un ensemble de nœuds capteurs. Ces nœuds capteurs sont organisés en champs « sensor fields », figure 1.1. Chacun de ces nœuds a la capacité de collecter des données et de les transférer au concentrateur par l'intermédiaire d'une architecture multi-sauts. Le concentrateur transmet ensuite ces données à la supervision «Gestionnaire de tâches» pour analyser ces données et prendre des décisions.

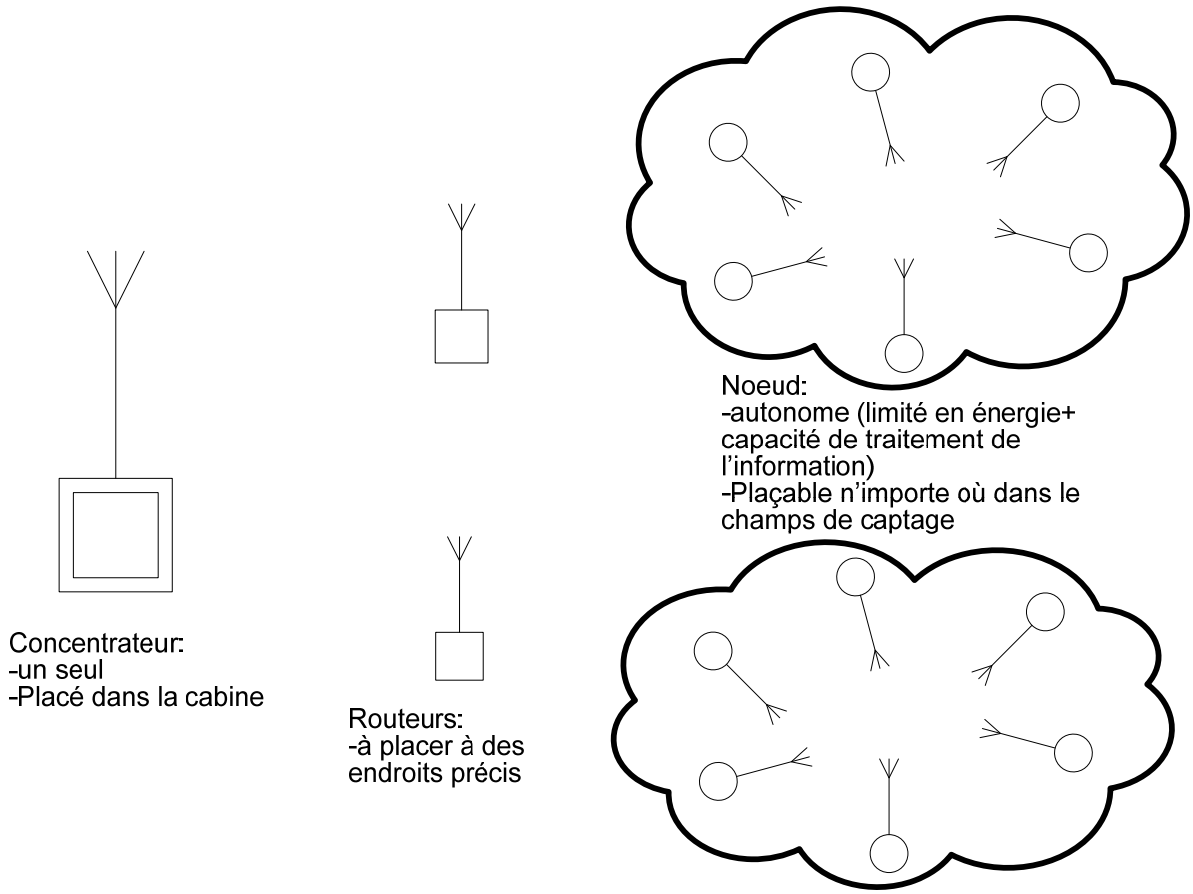


Figure 1. 1 Architecture des RCSFs

1.2.4 Les champs d'applications des réseaux de capteurs sans fils (RCSFs) :

La diminution de la taille et du coût des micro-capteurs, l'élargissement de la gamme des types de capteurs disponibles (thermique, pression, optique, vibrations,...) et l'évolution des supports de communication sans fil, ont élargi le champ d'application des réseaux de capteurs. Ils s'insèrent notamment dans d'autres systèmes tels que le contrôle et l'automatisation des chaînes de montage. Ils permettent de collecter et de traiter des informations complexes provenant de l'environnement (météorologie, étude des courants, de l'acidification des océans, de la dispersion de polluants, de propagules, etc.…).

Certains prospectivistes pensent que les réseaux de capteurs pourraient révolutionner la manière même de comprendre et de construire les systèmes physiques complexes, notamment dans les domaines militaire, environnemental, domestique, sanitaire, de la sécurité, etc. [10-29]

Applications environnementales

Pour améliorer la connaissance de l'environnement et surtout signaler d'éventuels problèmes y afférent (incendie, pollution, épidémies, aléa météorologique...), il est possible de répartir, dans la zone considérée, des thermo-capteurs formant un RCSF et offrant ainsi la possibilité d'une surveillance et une meilleure efficacité des moyens de lutte. De la même manière, des capteurs appropriés peuvent être dispersés au-dessus d'emplacements industriels, de centrales nucléaires, puits pétroliers, …, pour détecter et contrôler des fuites de produits toxiques (gaz, produits chimiques, éléments radioactifs, pétrole, etc.). Ces installations permettraient d'une part, de donner l'alerte en un temps record et d'autre part de suivre l'évolution de la catastrophe pour permettre une intervention plus efficace.

Des RCSF peuvent également être répandus ou fixés sur le corps des animaux afin de surveiller les forêts, la variation des habitats naturels, les déplacements, l'activité et la santé des groupes animaliers, ce qui permet d'observer la biodiversité sans aucune intervention locale et aucune perturbation de l'environnement naturel des espèces.

Les éventuelles conséquences de la dispersion en masse des micro-capteurs dans l'environnement ont soulevé plusieurs inquiétudes, notamment parce que ceux-ci sont généralement dotés d'une micro-batterie contenant des métaux nocifs. Néanmoins, le déploiement d'un million de capteurs de 1 millimètre cube chacun ne représente qu'un volume total d'un litre. Même si tout ce volume était constitué de batteries, cela n'aurait pas de répercussions désastreuses sur l'environnement. [10, 11, 12]

Applications à la sécurité

Les RCSF peuvent être utilisés pour assurer la sécurité des voyageurs empruntant des avions, navires, automobiles, métros, etc. en suivant en temps réel l'état de leurs structures, de même que l'état des réseaux de circulation ou de distribution de l'énergie. Des RCSF peuvent être aussi intégrés dans les murs ou dans le béton, sans alimentation électrique ni connexions filaires, pour surveiller la dégradation de la structure d'un bâtiment, d'une route, d'un quai, d'une voie ferrée, ou d'un pont. Dans ce contexte des mesures discontinues et périodiques permettent des durées de vie des RCSF importantes.

Selon leurs promoteurs, ces réseaux de capteurs pourraient diminuer certaines failles de systèmes de sécurité et mécanismes de sécurisation, tout en diminuant leur coût. D'autres craignent aussi des dérives sécuritaires ou totalitaires si l'usage de ces réseaux n'est pas assujetti à des garanties éthiques sérieuses. [13, 14, 15]

En plaçant à différents points stratégiques des capteurs, on peut ainsi prévenir des cambriolages ou des passages de gibier sur une voie de chemin de fer (par exemple) sans avoir à recourir à de coûteux dispositifs de surveillance vidéo.

Applications militaires

Un réseau de capteurs déployé sur un secteur stratégique ou difficile d'accès, permet par exemple d'y surveiller tous les mouvements (amis ou ennemis) ou d'analyser le terrain avant d'y envoyer des troupes (détection d'agents chimiques, biologiques ou de radiations). Ces utilisations militaires, comme dans le cas de beaucoup de technologies, ont été un moteur initial pour leur développement du fait de leur déploiement rapide, leur coût réduit, leur auto-organisation et leur tolérance aux pannes. [16]

Applications médicales et vétérinaires

Des applications importantes dans la surveillance des personnes souffrant de certaines maladies (Alzheimer, perte de mémoire,…) sont aussi un moteur important pour le développement de cette technologie. Les applications médicales des RCSFs ne se limitent pas à la surveillance des personnes mais elles peuvent aussi couvrir la surveillance des fonctions vitales d'un organisme vivant en implantant les micro-capteurs dans le corps ou en les faisant pénétrer par des moyens normaux (avalés). Des micro-caméras permettant, par exemple, de transmettre des images de l'intérieur d'un corps humain après avoir été avalées existent déjà. D'autres applications peuvent être envisagées pour remédier à certains organes défectueux ou les remplacer (analyseur de taux de sucre dans le sang, rétine artificielle pour corriger la vue, détection précoce de cancers).

Des réseaux de capteurs permettraient théoriquement une surveillance permanente des patients et une possibilité de collecter des informations physiologiques de meilleure qualité, facilitant ainsi le diagnostic de quelques maladies. [17, 18, 19, 20, 21]

Applications commerciales

On pourrait imaginer devoir stocker des subsistances nécessitant un certain taux d'humidité et une certaine température (min ou max). Dans ce type d'application, le réseau doit pouvoir collecter ces différentes informations et alerter en temps réel si les seuils critiques sont dépassés, ce qui permet un meilleur processus de stockage et de livraison (pour garantir la chaîne du froid en particulier). En plus des applications visant à surveiller la température et l'humidité, les RCSF permettent d'autres applications telles que le suivi des paquets, leur orientation, …). Grâce aux réseaux de capteurs, les entreprises pourraient offrir une meilleure qualité de service tout en réduisant leurs coûts. Les produits en fin de vie pourraient être mieux démontés et recyclés ou réutilisés si les micro-capteurs en garantissent le bon état. Des applications des RCSFs peuvent avoir un intérêt économique, en optimisant les systèmes de chauffage et de climatisation, en régulant l'éclairage. Utilisées à grande échelle, de telles applications permettraient de réduire la demande mondiale en énergie, la consommation des matières premières et diminuer la pollution. Le monde

économique pourrait ainsi diminuer ses impacts environnementaux sur le climat. [22, 23, 24]

Applications au contrôle de structures

Il est possible d'inclure sur les parois des barrages des capteurs qui permettent de calculer en temps réel la pression exercée par l'eau. Il est donc possible d'en réguler le niveau si les limites sont atteintes. On peut aussi imaginer inclure des capteurs entre les sacs de sables formant une digue de fortune. La détection rapide d'infiltration d'eau peut servir à renforcer le barrage en conséquence. Cette technique peut aussi être utilisée pour d'autres constructions tels que ponts, routes de montagnes, bâtiments et autres ouvrages d'art. [25, 26, 27, 28, 29]

Dans l'aéronautique, la connaissance de la répartition de la pression ainsi que la température sur les ailes des avions durant les vols est indispensable autant dans le choix des matériaux de fabrication que pour la certification d'un appareil avant sa commercialisation. Un tel test demande de répartir plus de 200 capteurs par plan d'ailes, à une fréquence d'échantillonnage importante, ce qui demande des kilomètres de câbles posant alors des problèmes d'aérodynamique et rendant l'utilisation des RCSFs quasiment incontournable.

1.2.5 Les caractéristiques techniques

Protocoles de communication industriels

Vue la diversité des domaines d'utilisation des RCSFs, les technologies pour leur mise en œuvre sont nombreuses. Pour les protocoles de communication, Bluetooth, Zigbee, Wibree, sont les premiers à apparaitre dans les RCSFs, d'autres sont plus récents comme Wireless USB, WiFi, Z-Wave, SimpliciTI, MiWi et TinyOS... [30] [31]

Certains des protocoles cités ci-dessus comme SimpliciTI, Synkro, MiWi, TinyOS, et Zigbee, se basent sur la norme IEEE 802.15.4 dédiée à la création de réseaux locaux à bas débit, petites distances et faible consommation.

La norme *Bluetooth*, dont Ericsson a initié le projet en 1994, a pour but la création et le maintien de réseaux à portée personnelle, PAN (Personal Area Network). Un tel réseau est utilisé pour le transfert de données à bas débit à faible

distance entre appareils compatibles. Malheureusement, le grand défaut de cette technique est sa trop grande consommation d'énergie et ne peut donc pas être utilisée par des capteurs qui sont alimentés par une batterie et qui idéalement devraient fonctionner pour des durées importantes, en plus de l'incapacité à évoluer jusqu'à des systèmes comportant des milliers de nœuds. [32, 33]

Le *ZigBee* combiné avec IEEE 802.15.4 offre des caractéristiques répondant aux besoins des réseaux de capteurs en termes d'économies d'énergie. ZigBee offre des débits de données moindres, mais il consomme également nettement moins que Bluetooth. Un faible débit de données n'handicape pas pour autant un réseau de capteurs où les fréquences de transmission sont faibles. [34, 35, 36]

Wibree est une technologie nouvelle, semblable à Bluetooth, qui est conçu pour coexister avec Bluetooth dans des applications similaires. L'importante différence est qu'il est conçu pour avoir une consommation électrique extrêmement faible. Comme Bluetooth, Wibree est conçu pour les réseaux personnels (PAN) et il n'est pas capable de former de grands réseaux. [31]

Wireless USB est une technologie peu coûteuse, principalement ciblé pour les périphériques du PC. Bien que l'espace des capteurs ne soit pas son objectif principal, Wireless USB pourrait être adapté à cette fin.

WiFi est une technologie largement utilisée qui pourrait être adaptée pour former les réseaux de capteurs. Cependant, il n'y a pas de norme commune disponible pour réseaux de capteurs avec le WiFi. Les principaux inconvénients de WiFi sont le prix relativement élevé, la consommation d'énergie excessive et la grande complexité.

Z-Wave est un protocole (et technologie) exclusive RCSF, sous-1GHz, de la société Zensys de Sigma Designs. Il est destiné spécialement aux applications domotiques. Z-Wave n'est pas une norme ou une technologie ouverte, mais il a plusieurs adoptants dans l'industrie domotique [31]. Z-Wave est assez similaire à Zigbee, dans le sens ou les deux, Zigbee et Z-Wave, sont conçus pour une utilisation WSN de faible consommation [37].

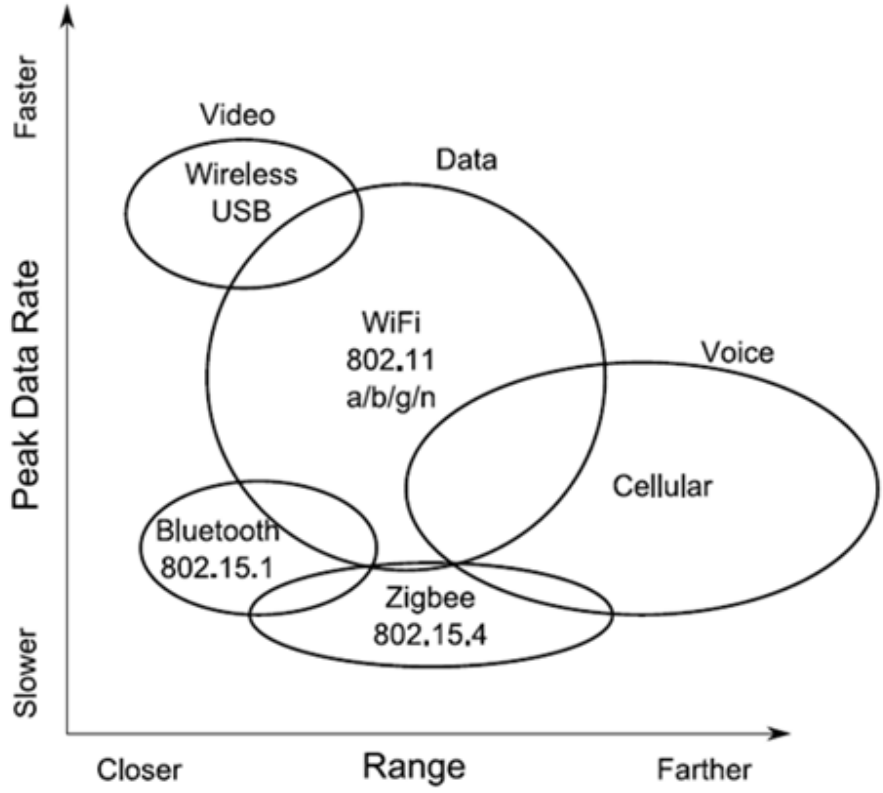
SimpliciTI, de Texas Instrument Corporation (TI), est un protocole à simple répétition destiné pour une utilisation avec TI 802.15.4 radios. SimpliciTI est un protocole simple de faible consommation, visant des petits réseaux RF. Ces réseaux contiennent généralement des appareils qui nécessitent une batterie longue durée, ont

un faible débit de données, un faible taux d'utilisation et un nombre limité de nœuds interconnectés entre eux ou à un routeur central (réseau en étoile). Le réseau en étoile n'est pas obligatoire, mais des fonctionnalités supplémentaires peuvent être implémentées comme, par exemple, la sauvegarde et le transfert des messages. SimpliciTI a été conçu pour une mise en œuvre facile et un déploiement sur plusieurs plates-formes RF TI tels que la famille MSP430 de microcontrôleurs à faible puissance et les émetteurs-récepteurs et CC1XXX/CC25XX/CC430 SoCs. [38]

MiWi, de Microchip Corporation, est un protocole qui utilise IEEE 802.15.4 dans ses couches inférieures. Ce protocole prend en charge les applications à courte portée et à faible consommation des réseaux sans fil. MiWi est destiné pour une utilisation avec les cartes radios de Microchip. [39]

TinyOS est un système d'exploitation open-source, conçu pour réseaux de capteurs sans fil. Il utilise un langage de programmation appelé le nesC [40]. TinyOS a été adapté pour divers projets de RCSF.

Il existe beaucoup plus de protocoles et de technologies disponibles pour les réseaux de capteurs sans fil que ce que l'on a cité ici. Les recherches se basent à chercher les protocoles à garantir une meilleur configuration des réseaux et une économie d'énergie de communication [41]. Les constructeurs tendent à employer des *« techniques propriétaires »* ayant l'avantage d'être spécifiquement optimisées pour une utilisation précise, mais avec l'inconvénient de ne pas être compatibles entre elles. Il reste à indiquer, que selon l'application voulue et les contraintes imposées sur la consommation, le débit et la portée, un protocole ou un autre, une technique ou l'autre peuvent être choisis. La figure 1.2 illustre une comparaison sur des exemples de protocole abordés dans ce chapitre.



Standard	Compagnie	Bande de fréquence	Débit	Range	description
Wibree	Nokia	2.4G	1M	<10m	Contraintes sur la faible puissance Utilisé en parallèle avec Bluetooth
Z-Wave	Zensys	868.42M 908.42M	40k	100m	Coute la moitié de ZigBee
SimpliciTI	TI	1G/2.4G	Moins de 100 nœuds, complémentaire de ZigBee Moins de coût et moins de puissance		
MIWI	Microchip	2.4G	Basé sur IEEE802.15.4, faible puissance, faible débit		
SMAC & Synkro	Freescale	2.4G	SMAC : basé sur 802.15.4 PHY, topologie étoile Synkro : control de RF basé sur 802.15.4, communication en double sens		

Figure 1. 2 Comparaison des protocoles sans fils [31]

1.2.6 Les verrous matériels et logiciels

Les recherches actuelles en matière de RCSF se focalisent d'une part sur l'augmentation du débit des échanges d'information et d'autres parts sur la réduction de la consommation énergétique et la possibilité de pénétration à travers des obstacles. Pour ces différents aspects, l'UWB[2] (Ultra Wide Band) se présente comme

[2] L'**ultra wideband**, ou **UWB** est une technique de modulation radio qui est basée sur la transmission d'impulsions de très courte durée, souvent inférieure à la nanoseconde. Ainsi, la bande passante atteint de très grandes valeurs. On utilise principalement deux méthodes de modulation des signaux : modulation en position d'impulsions, soit en modulation temporelle, soit en modulation biphase.

une voie intéressante car c'est une technique de transmission permettant des consommations extrêmement basses grâce à sa simplicité matérielle. De plus, l'atténuation du signal engendré par des obstacles est moindre qu'avec les systèmes radio à bande étroite conventionnels. [42]

Les applications des RCSFs étant promises à un essor considérable, cela pousse à créer des systèmes d'exploitation de taille extrêmement réduite en termes de mémoire et des protocoles de communications allégés, lesquels devant répondre aux besoins de miniaturisation et de prolongation de la durée d'utilisation sans gaspiller d'énergie et de ressources physiques.

1.2.7 Architectures des réseaux de capteurs

Parmi les topologies de réseaux à communication radio, on peut citer celles qui peuvent être applicables aux RCSFs.

La Topologie en étoile

La topologie en étoile, comme son nom l'indique, se base sur un centre de communication qui envoie et reçoit les messages de tous les nœuds du réseau. Toute communication ne se fait qu'au travers de ce centre. Cette topologie est caractérisée par sa simplicité, l'économie d'énergie et la communication directe qui minimise les retards dans la transmission des données. Figure 1.3

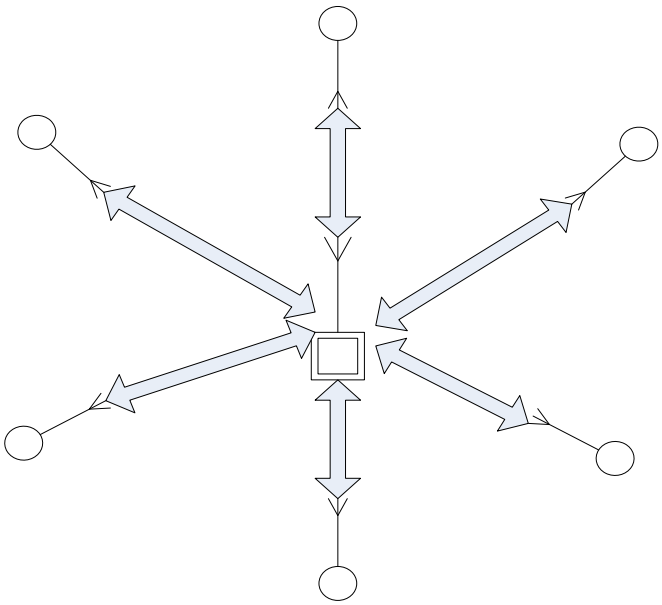


Figure 1. 3 Topologie en étoile

La topologie « en toile » ou « en grille » (Mesh Network)

Contrairement à la topologie précédente, celle-ci peut être décrite par des liaisons dans tous les sens : chaque nœud peut échanger avec n'importe quel nœud du réseau qui est à sa portée de transmission. En conséquence la transmission des données vers le concentrateur peut passer par plusieurs nœuds intermédiaires, d'où la dénomination de « multi-saut ». Cette topologie, qui se caractérise par sa capacité à grandir en ajoutant des nœuds et à avoir une autoréparation en cas de pannes, demande une importante consommation d'énergie et entraîne une augmentation des retards dans la chaine de communication. Figure 1.4

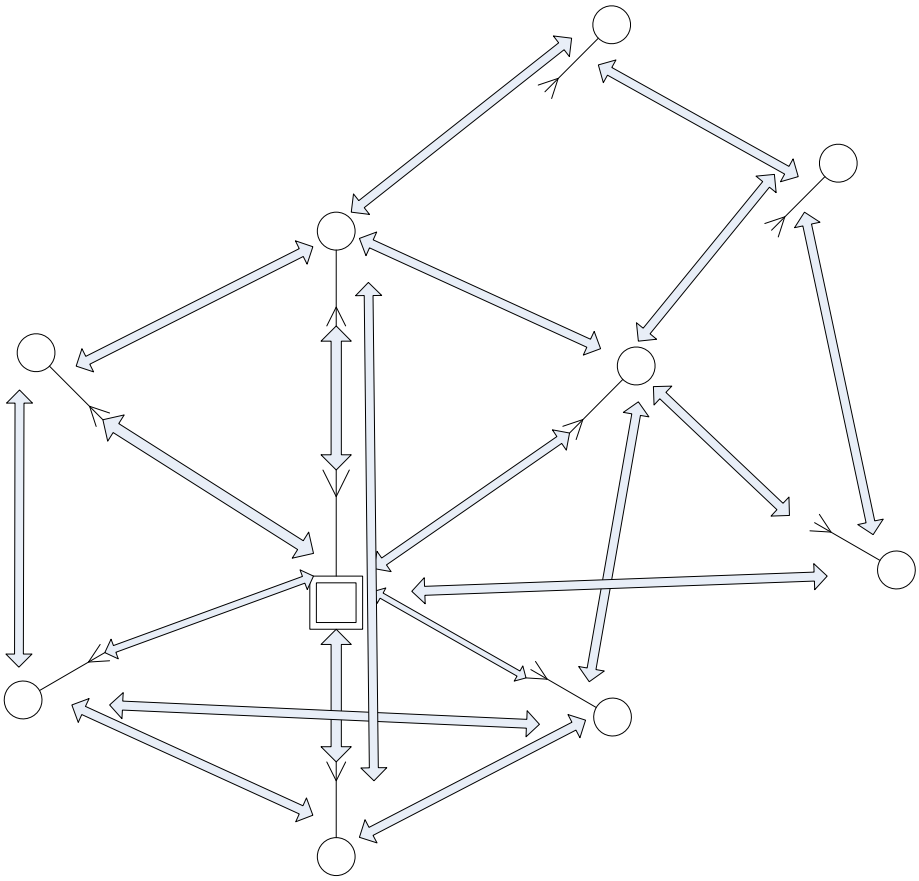


Figure 1. 4 Topologie en toile ou en grille

La topologie « en grappes »

La topologie en grappes est un assemblage des deux topologies en étoile et en grille. C'est une topologie qui essaie de regrouper la minimisation de la consommation d'énergie avec la robustesse et la possibilité d'élargissement. Dans cette topologie les nœuds ne communiquent pas entre eux mais sont reliés à d'autres éléments qu'on appelle routeurs. Ces derniers effectuent le routage et peuvent avoir

une source d'énergie externe. Le réseau, dans ce cas, prend la forme d'une grappe. Figure 1.5 [11]

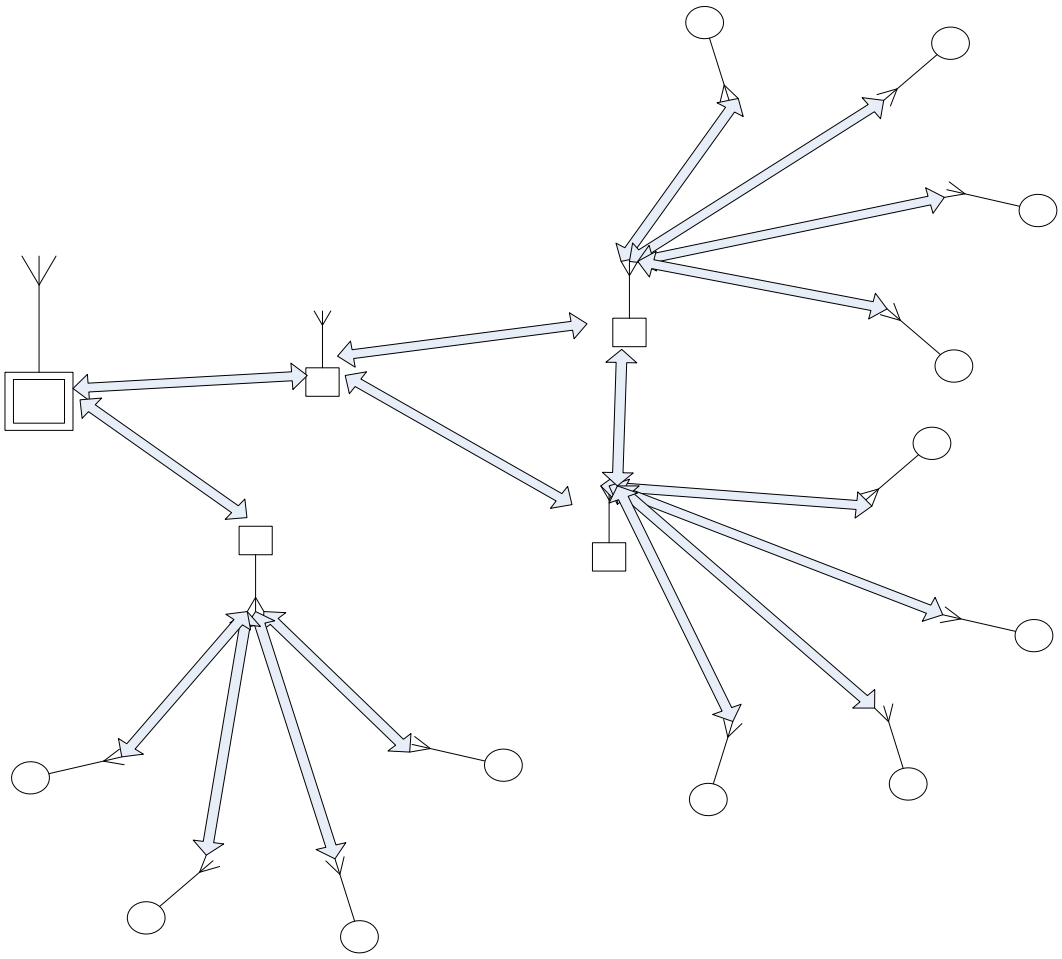


Figure 1. 5 Topologie en grappes

1.3 La problématique : simulation du projet SACER

1.3.1 Types de RCSFs

Les domaines d'application des réseaux de capteurs sans fils sont très divers, de même les finalités recherchées au sein d'un même domaine peuvent changer. Ainsi l'architecture des réseaux et les éléments constituants dépendent de ces domaines d'utilisation et de l'objectif pour lequel le réseau est installé.

Certains réseaux peuvent être installés pour des utilisations à longue durée, d'autres pour une période de test définie, certaines applications utilisent un grand nombre de nœuds, d'autres non. Des applications demandent des mesures en continu de tous les nœuds à des fréquences importantes, donc haut débit, d'autres n'agissent

qu'à des instants précis et à des endroits limités, donc bas débit. La présence de communication latérale entre les nœuds peut paraître indispensable dans certaines applications dans d'autres non. La possibilité d'une extension ultérieure du réseau peut se poser également.

Dans certaines applications toutes les mesures doivent être envoyées au concentrateur pour être traitées au niveau de ce dernier, dans d'autres, une analyse est faite au niveau du nœud pour prendre la décision d'envoyer ou non les données recueillies. Cependant même dans ce dernier cas, un réseau intelligent (avec un pouvoir de traitement au niveau des nœuds) ne peut être construit sans passer par une étape intermédiaire de collecte de toutes les données, lesquelles serviront à définir l'algorithmique qui pourra être implantée localement.

En résumé une application doit se caractériser par les termes suivants avant d'être développée :

- Durée d'utilisation.
- Taille du réseau.
- Consommation énergétique.
- Débit des données.
- Nature des communications (latérales, ascendantes, descendantes).
- Possibilité d'extension.
- Réseau intelligent ou non.

C'est en se basant sur ces différentes caractéristiques que l'on doit choisir les solutions techniques du réseau : nature des ressources énergétiques, architecture, protocole de communication, redondance, topologie…

Ainsi les critères d'évaluation des RCSFs, lors de l'étape de conception, dépendent de leur type et de leur mode d'emploi : par exemple si la caractéristique d'élargissement du réseau est la plus importante, l'étude doit se focaliser sur la capacité du protocole de communication à s'auto-configurer et à assurer la communication d'un nouveau nœud. Dans le cas d'un réseau intelligent où le débit d'échange est faible mais dans lequel une donnée doit impérativement être envoyée

du fait de sa criticité, la question de la redondance prend toute sa place et l'étude du réseau prendra en compte la possibilité d'enregistrer temporairement les données dans l'attente d'un accusé de réception de la part du concentrateur.

Dans les réseaux d'instrumentation dont le projet SACER fait partie, les exigences sont différentes : c'est un réseau dont la répartition des nœuds est généralement prédéfinie, qui fait des mesures en continu et achemine celles-ci, à haut débit, vers le concentrateur. Dans ce contexte, bien que le temps devenant un facteur crucial excluant la mise en œuvre de mécanismes de redondances, il faut malgré tout garantir la qualité des données transmises. Un critère important dans la conception du réseau sera donc **l'évaluation de la qualité du canal de transmission.** Cette évaluation pourra se faire au travers du **taux d'erreurs par bits échangés (BER)** car ce critère peut donner une idée sur la qualité du canal de transmission et sur la possibilité d'avoir des données corrompues lors de la communication. Un autre facteur tel le nombre d'erreurs par trame échangée, peut être pris en compte. Ce facteur est important du fait de la possibilité d'intégrer dans la trame un code correcteur qui permet de vérifier l'intégrité des bits échangés voire de les rectifier via un algorithme approprié. Il est à noter que ce code correcteur, selon sa performance, influe sur la taille des trames et le temps calcul qu'il nécessite.

1.3.2 Le projet SACER de surveillance des structures aéronautiques :

Description du projet

Le but du projet SACER (**S**ystème **A**utonome **C**ommunicant **E**n **R**éseau) est l'introduction de communications sans fil dans des réseaux constitués de centaines de capteurs embarqués afin, d'une part, de faciliter leur mise en œuvre et leur exploitation et d'autre part, limiter l'usage de kilomètres de câbles qui par leur poids et leur encombrement modifient les comportements des structures à surveiller.

Ce projet a été initié par le pôle de compétitivité Midi-Pyrénéen « Aerospace valley » pour des besoins aéronautiques et spatiaux. Il fédère quatre PME régionales, le LAAS-CNRS en tant que laboratoire de recherche et trois utilisateurs finaux : Airbus (figure 1.6), Intespace et récemment le CEAT.

Figure 1. 6 Application du projet SACER chez Airbus

Il a pour ambition de développer un système de capteurs autonomes communicant sans fils répondant aux besoins des utilisateurs finaux et également exploitable pour d'autres types d'applications tout en prenant en compte l'ensemble des enjeux technologiques :

- Autonomie :
  - Gestion de l'énergie: mise en veille, réveil
  - Autonomie, Auto alimentation
- Communication :
  - Directivité des antennes
  - Protocoles, environnement contraint, très haut débit
- Réseau Capteur
  - Micro, Nano systèmes (packaging)
  - Capteur passif, faible consommation
- Méthodologie, conception et outils de simulation
  - Modélisation de l'architecture, topologie,
  - Gestion des configurations

Spécifications du projet SACER

Le projet SACER est une réponse aux besoins des sociétés AIRBUS et INTESPACE qui consistent en la collecte de données décrivant le comportement d'un avion en phase d'essais en vol ou d'un satellite avant commercialisation. Le réseau de capteurs sans fil est destiné à remplacer les équipements de mesure filaires actuels.

L'approche méthodologique générale du projet SACER :

La méthodologie générale (IEEE-EIA-632) de conception du réseau de capteurs mise en œuvre implique de coordonner les grandes étapes suivantes :

- Elaboration du document de spécifications
- Conception préliminaire
- Conception détaillée
- Implémentation
- Tests unitaires
- Intégration

Bien que ces étapes soient présentées de manière séquentielle, le processus de conception est un processus itératif qui implique des allers-retours entre chacune d'elles jusqu'à la convergence vers une solution technologique acceptable.

A chaque étape du processus il est largement fait appel à la simulation.

i. Elaboration du document de spécifications

C'est le document obtenu à partir du cahier des charges de l'objet à concevoir et à réaliser. Il s'appuie sur le cahier des charges d'origine et le complète de manière à ce qu'aucun flou ne subsiste quand à la finalité du produit à réaliser. Le document de spécifications est rédigé en déterminant les acteurs, le nombre et la nature des entrées-sorties, toutes les fonctions supportées par le produit, le contexte environnemental (gamme de température de fonctionnement, de stockage, présence d'humidité, poussière, vibrations, parasites électromagnétiques (CEM), les besoins en énergie, la signature électromagnétique, la durée de vie…).

ii. La conception préliminaire

Elle est une première formalisation des spécifications en utilisant par exemple des langages comme SYSML ou UML. Elle a pour but de définir, l'architecture logique qui sera retenue en décomposant le système en sous-systèmes. Cette architecture logique doit être vérifiée : se pose alors la question de choisir les techniques mises en œuvre pour converger vers une solution logico-temporelle matérialisée acceptable.

Cette phase est souvent décrite comme une étape « meet in the middle » car elle illustre les échanges répétés entre vision logique et vision physique, jusqu'à aboutir a une solution définitive.

iii. La conception détaillée et l'optimisation système

En principe, à ce stade de la conception, les solutions technologiques ne sont pas encore complètement finalisées : c'est une confrontation qui doit se mettre en place entre des solutions technologiques disponibles et le simulateur pour aboutir a un ensemble cohérent vérifiant toutes les exigences initiales fixées par les spécifications. Il faut donc un accès aux bibliothèques technologiques et faire jouer au maximum le retour d'expérience qui favorise grandement cette étape. A chaque grande fonction doit être affecté un plan de test qui sera fourni aux réalisateurs en même temps que les scénarii-type de test à créer afin de vérifier que la fonction répond aux spécifications.

Notre démarche consistera, dans ce contexte méthodologique, à créer un simulateur qui va aider l'équipe projet à fixer, pour chaque fonction identifiée, les solutions technologiques à admettre.

iv. L'intégration et le prototype

L'intégration correspond, dans un premier temps, à l'assemblage matériel et logiciel des constituants retenus par l'étape précédente : on est dans la deuxième branche du cycle en V, avec mission d'appliquer les tests et les vérifications partielles jusqu'à la validation du produit complet, c'est à dire lorsque tous les modules sont connectés entre eux. Finalement, le produit sera testé et validé dans son environnement réel.

1.3.3 Le rôle de la simulation logico-temporelle :

L'objectif de la simulation est, dans un premier temps, de concevoir et valider un prototype virtuel du système avant fabrication et dans un deuxième temps de prévoir ses performances lors d'une modification de configuration ou un redéploiement dans un environnement différent. Bien que nos travaux s'inscrivent dans ce dernier cadre nous avons opté pour un environnement de simulation et un langage de description qui offrent la possibilité de créer des modèles « fins » capables de simuler les phénomènes physiques mis en jeu. Ainsi il sera possible à l'utilisateur et en fonction de ses besoins de mélanger des modèles « haut niveau » et des modèles « fins » dans son architecture. Bien évidemment cette possibilité induit des temps de simulation rallongés.

La conception du simulateur implique le développement d'une bibliothèque de modèles à partir de laquelle il sera possible de construire une architecture de réseau et d'en évaluer les performances. Nous avons donc défini un certain nombre d'éléments de base constituant un réseau de capteurs communicant sans fil. Nous avons identifié les cinq éléments de base suivant :

- Le capteur
- Le nœud (qui peut recevoir jusqu'à huit capteurs)
- Le canal de transmission (qui représente le lien RF entre nœuds, routeurs et concentrateur)
- Le routeur vers lequel sont acheminées les données collectées par un ou plusieurs nœuds
- Le concentrateur qui reçoit les données issues d'un ou plusieurs routeurs.

Enfin nous avons défini quels étaient les paramètres les plus pertinents pour caractériser les performances d'une architecture de réseau à savoir :

- Le débit des données
- Le taux d'erreur de bits (BER : Bit Error Rate)
- Le taux d'erreur de bits par trame (PER : Packet Error Rate)

- Le nombre d'erreur dans une trame
- La latence (délai entre l'instant ou la donnée est prélevée et l'instant ou elle est mise à disposition de l'utilisateur final)
- La consommation énergétique
- La synchronisation des données prélevées

Pour effectuer des simulations représentatives et fiables du réseau dans des configurations de déploiement variées, nous avons développé un ensemble de modèles comportementaux et génériques représentatifs de ses éléments constitutifs ainsi que des sondes de mesure. Ainsi il est possible, par association des différents éléments, de bâtir différentes architectures, les simuler et évaluer leurs performances respectives.

1.3.4 La place de la simulation logico-temporelle dans le processus de conception:

La grande variabilité des choix technologiques dans la conception d'un réseau de capteurs implique la mise en place d'un véritable processus de conception.

Ce processus définit une démarche en étapes successives de conception-vérifications comme l'imposent les normes IEEE/EIA-632 et le cycle en V dans l'ingénierie système:

1- Sur la base des idées qui ont motivé la démarche de conception, la première étape est d'écrire le cahier des charges (spécifications), entre tous les partenaires, en explicitant les ambitions de toutes natures, les exigences et les contraintes.

2- Il convient ensuite, d'extraire de ce cahier des charges une représentation fonctionnelle conforme et vérifiable dans son architecture et dans sa logique fonctionnelle.

3- Le passage de la représentation logique à une représentation matérialisée suppose d'abord de revoir le redécoupage fonctionnel en tenant compte de la position « organique » de la fonction dans le système réel. La norme EIA-632 nous invite à faire une nouvelle représentation en « BUILDING BLOCKS » en s'appuyant sur une vision organique du système, de spécifier chaque building block comme un

système autonome, de diffuser les spécifications de chaque building block aux fournisseurs internes et externes susceptibles de pouvoir y répondre…

4- Notre contribution dans cette démarche générale est de proposer d'organiser ce passage de la représentation logique à une représentation matérialisée par une **succession de proposition d'un choix technologique-évaluation de ce choix, par simulation :**

- Partant des exigences de haut niveau, on va puiser dans la bibliothèque des technologies ou dans des opérations de re-use pour sélectionner et proposer une option technologique de la fonction considérée
- Cette option devra être évaluée, dans son contexte d'utilisation, par simulation : elle pourra être retenue, corrigée ou rejetée, ce qui suppose alors de revenir sur un nouveau cycle proposition-évaluation….

5- Dans cette approche, il est nécessaire de disposer d'un simulateur où les modèles sont parfaitement représentatifs de cette réalité technologique et, pour être exhaustif, de disposer d'une méthode de sélection pour arbitrer des solutions alternatives très proches : Des travaux, sur la sélection multicritères ont déjà été développés et répondent à ce second aspect du problème.

Notre centre d'intérêt est donc la définition d'un simulateur utile à cette transition entre la vue logique du système et la vue matérielle : nous regardons plus précisément ici les simulateurs de réseaux.

La figure 1.7 représente la démarche de conception que l'on adopte avec pour finalité la création d'une solution physique que l'on décrit comme le prototype virtuel de notre système. Ce type de solution, intégrant notamment les contraintes technologiques, permet l'optimisation du système (modèle prédictif). La solution logique quant à elle, va être une description du système sous la forme d'arrangement de fonctions et de leurs interactions, laquelle définit le séquencement de leur exécution, les conditions sur leur flux de données et contrôle et les performances requises pour répondre au référentiel des exigences (modèle descriptif). Avant d'entrer dans les choix qui ont été les nôtres, nous allons étudier l'offre existante pour la simulation des réseaux de capteurs.

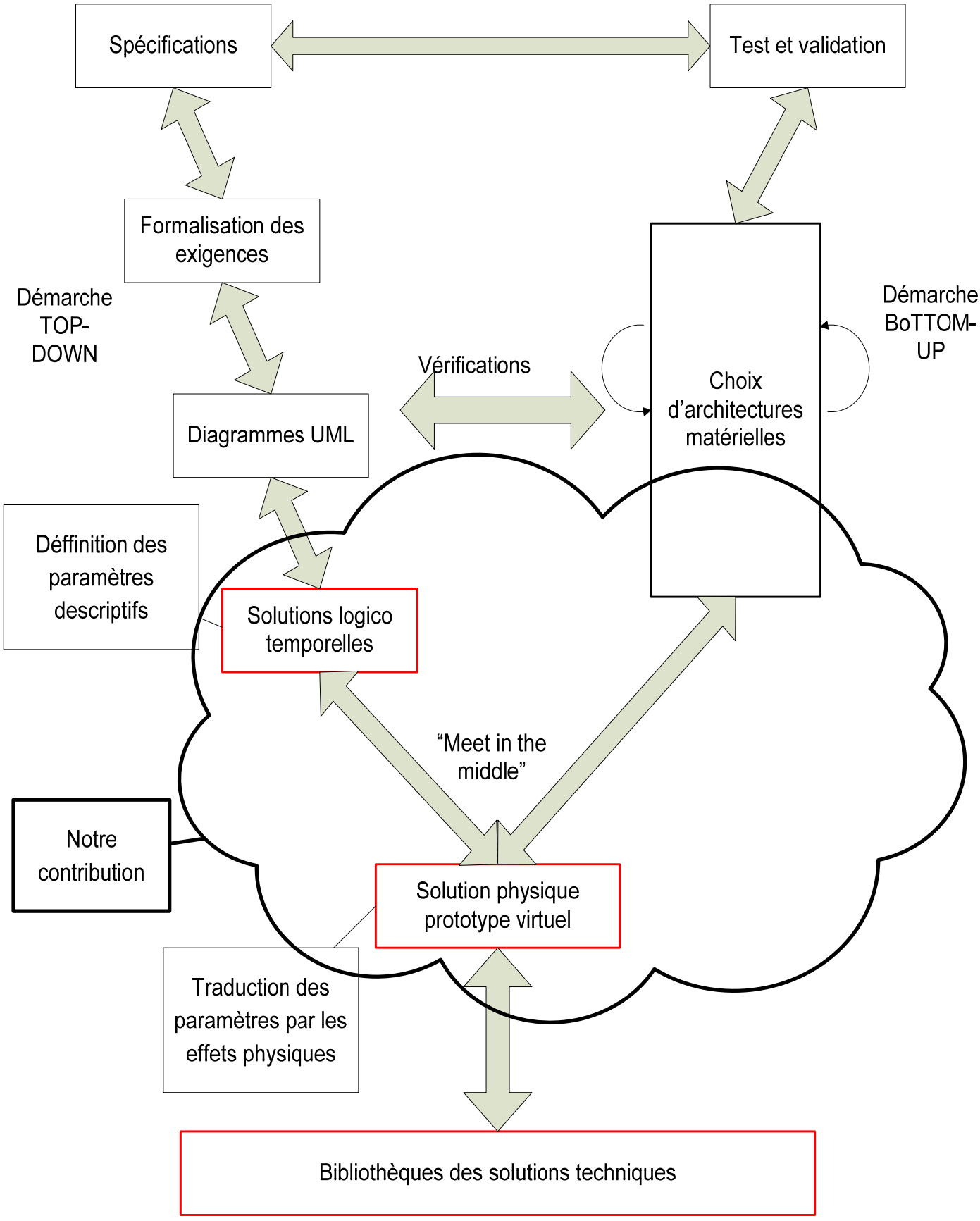


Figure 1. 7 Place de la simulation dans le processus de conception

1.4 Proposition de simulateurs

1.4.1 Etat de l'art des simulateurs de réseaux de capteurs sans fils

Les simulateurs de réseaux peuvent être classés en grandes catégories selon la nature des résultats qui peuvent en être tirés. En d'autres termes, ces simulateurs, le plus souvent, sont destinés à répondre à des développements de RCSFs spécifiés par leurs contraintes : contraintes de communication (protocole et pile de communication, auto configuration, latence de communication, erreurs, …) ou contraintes sur les nœuds (encombrement, correspondance des capteurs, latence de traitement, …).

Le premier groupe de simulateurs exploite la notion de réseau : il se base sur les simulateurs de réseaux informatiques (pour les ordinateurs) tels que : NS-2 [43], GloMoSim [44, 45], QualNet [46, 47], OPNET [48], OMNET [49], Scalable Simulation Framework (SSF) [50], et J-Sim [51]. Ces simulateurs se focalisent sur le comportement du réseau sans fil et sur la pile du protocole d'opération. Une illustration de ces simulateurs est représentée dans le tableau 1.1

Tableau 1. 1 Liste des simulateurs orientés réseau et leur simulateur de base [69]

Simulateurs de RCSF orientés réseau	**Simulateurs de réseaux informatiques de base**
"SensorSim" [52, 53]	étend "NS-2"
"Naval Research Laboratory's (NRL) sensor network simulator" [54]	étend "NS-2"
"sQualnet [55] "	"Qualnet"
"Simulator for Wireless Ad-hoc Networks (SWAN)" [56]	"SSF"
"SENSIM" [57]	"OmNet++"
"simulation template" pour "EYES" [58]	"OmNet++"
"J-Sim sensor simulator" [59]	J-Sim
"VisualSense" [60]	"Ptolemy II" [61]
"Prowler" [62]	"MATLAB"
"H-MAS" [63]	environnements de simulation personnalisés.
"SENSE" [64]	environnements de simulation personnalisés.

Ces simulateurs se basent sur des simulateurs préexistants qui disposent de moyens de transmission réalistes, de piles de protocoles plus ou moins développées, de modèles de canaux de transmissions variables dans leur complexité et dans les phénomènes qu'ils intègrent. Certains simulateurs disposent d'une chaîne de mesure séparée contenant les mesurandes afin de modéliser les phénomènes captés. En ce qui concerne les capacités de la plate-forme de nœud, seule la consommation est prise en compte, soit par des modèles de puissance détaillés, lesquels prennent en compte le taux de décharge de la batterie, soit par des modèles linéaires de batteries.

Les simulateurs orientés réseau modélisent le moyen de transmission en détail et sont plus adaptés à la simulation des RCSFs grande échelle. Ces simulateurs sont du domaine des spécialistes des réseaux : ils étudient les protocoles, permettent l'ajout ou le retrait d'éléments, la recherche d'un trajet de communication optimal, ... Ces simulateurs sont plutôt destinés aux personnes qui travaillent sur les éléments précités et sur le développement de plates-formes software pour les RCSFs.

Le deuxième groupe de simulateurs focalise principalement sur le fonctionnement d'un seul nœud tout en mettant en œuvre un modèle de communication léger. Ces simulateurs sont spécifiques à des nœuds bien définis ainsi qu'aux systèmes d'exploitation qui y sont intégrés et vérifient la compatibilité d'un nœud à une application donnée en répondant à ses exigences. Une illustration de ces simulateurs est représentée dans le tableau 1.2

Tableau 1. 2 Liste des simulateurs orientés nœuds et leur système d'exploitation associé [69]

Simulateurs de RCSF orientés nœuds	**Système d'exploitation associé**
" TinyOS Simulator " (TOSSIM) [40]	"TinyOS motes"
" ATmel EMUlator " (ATEMU) [65]	"TinyOS motes"
" TinyOS Scalable Simulation Framework " (TOSSF), une extension de " SWAN " [66]	"TinyOS motes"
" SENS " [67]	"TinyOS motes"
"EmSim " [68]	"Em*Linux"
" Sensor Network Asynchronous Processor " (SNAP) [69]	plusieurs processeurs sur un réseau sur puce (" Network-on-Chip " NoC).

Dans ces simulateurs, le canal de transmission est simplifié : il est réalisé soit par un simple BER soit par des modèles idéaux ou spéciaux. De même les piles de

protocoles sont simplifiées voire décrites par l'utilisateur, les phénomènes physiques sont modélisés par des modèles propres au simulateur ou par la lecture d'un capteur via une source externe. Aucun de ces simulateurs ne modélise la consommation de l'énergie. Les applications de tous les simulateurs orientés nœuds de capteurs et protocoles peuvent être directement mises en correspondance avec les plates-formes matérielles. Cependant, la plate-forme est limitée à une structure particulière, et ne peut pas être utilisée pour les autres plateformes.

Suite aux deux groupes de simulateurs déjà cités, une tentative d'en créer un qui en combine les avantages a été initiée : ce simulateur, qui s'appelle WISNESS, est toujours en cours de développement et non commercialisé à ce jour.

"WIreless SEnsor NEtwork Simulator" (WISENES) est un outil de simulation qui permet la conception, la simulation, la mise en œuvre et l'évaluation des réseaux de capteurs avec des informations pré-annotées mesurées. WISENES est ciblé pour la conception et le déploiement des réseaux RCSF réels. La principale différence qu'il présente comparativement aux autres est qu'il n'est pas nécessaire de procéder à un autre projet de modélisation de haut niveau du RCSF et un autre projet de développement pour la mise en œuvre effective. Au lieu de cela, WISENES supporte toutes les phases de la procédure de conception. Toutefois, WISENES peut également être utilisé comme simulateur simple comme les autres simulateurs de RCSF. [70]

D'autres simulateurs du marché qui prennent en compte les deux aspects sont généralement spécifiques à des constructeurs définis, lesquels fournissent des outils de développement pour tester et utiliser les cartes et les composants de réseaux propriétaires comme le « MiWi™ Development Environment » de « Microchip » qui permet de choisir un protocole et une architecture du réseau et les éléments constituant [39]. « Texas Instrument » de son coté a conçu un outil de développement qui sert à réaliser des RCSFs en utilisant le protocole « SimpliciTI » sur les plateformes RF en testant ses émetteurs-récepteurs et microcontrôleurs à faible consommation. [38]

1.4.2 Objectifs et limites de nos travaux

La conception des systèmes embarqués devenant de plus en plus un processus complexe et multidisciplinaire il devient impératif, dans la démarche de conception, avant de procéder à la fabrication ou au déploiement et après la définition des

spécifications et la formalisation des exigences, d'intégrer une phase de définition d'une solution logico-temporelle : Celle-ci consiste en la création d'un prototype virtuel simulable grâce auquel on pourra vérifier la fonctionnalité système et le respect des exigences du cahier des charges.

Cette phase de vérification est ensuite suivie par une procédure de recherche de solution technologique. Cette procédure, qui est de nature itérative, peut être qualifiée de méthodologie « Meet in the Middle » et a pour but, en confrontant les résultats des simulations et les solutions technologiques choisies, de converger vers une solution technologique finale acceptable. Elle se démarque de la méthodologie de conception des systèmes purement logiciels (approche Top Down) en ce sens que les choix technologiques, faits tardivement dans le processus de conception, peuvent s'avérer incompatibles avec les contraintes du cahier des charges (temps d'exécution, consommation, encombrement, …). Cette méthode de recherche n'est pas normalisée à ce jour et chacun la conduit à sa manière, en se basant sur sa propre expérience. Toutefois vue la complexité des systèmes liés au milieu socio-économique, tels que ceux de télécommunications, contrôle de procédés ou autres, ce processus spécifique de simulation convergeant vers une solution technologique acceptable devient incontournable.

En général, pour vérifier des solutions techniques complexes, on a recours à plusieurs simulations et tests unitaires séparés dans des environnements divers, ces simulations correspondant à chaque composition et à chaque élément du système. Afin d'améliorer la fiabilité on essaie d'avoir une simulation regroupant l'ensemble des simulations partielles. Cette simulation globale, idéalement faite dans un environnement unique, devrait couvrir les différents domaines de la physique et tous les éléments composant le système. C'est une procédure consommatrice de temps et c'est un des facteurs les plus exigeants dans la réduction du « time to market ». Ainsi afin d'accélérer cette phase de conception et réduire les délais de commercialisation, un environnement de simulation homogène, rapide et performant s'avère utile pour le concepteur. [71]

Une simulation rapide et performante : rapide implique que les modèles utilisés seront des modèles fonctionnels comportementaux. Ils représentent les fonctions assurées par chaque élément pris à haut niveau sans nécessairement entrer dans les détails des équations physiques de chacun des éléments ; performante, (conduisant à

des résultats de la simulation précis) implique que les modèles comportementaux des composants élémentaires du système doivent être la représentation fidèle des éléments réels dans leurs influences et réponses à tout phénomène inter-agissant : perturbation, affaiblissement, retard,…

Pour atteindre cet objectif, nous proposons de construire un modèle fonctionnel de base associé à une fiche de paramètres. Cette approche permet ainsi d'intégrer un certain nombre de ces paramètres, accessibles et modifiables par le concepteur, qui permettent de rendre compte des phénomènes physiques observés.

Dans ce qui suit, nous présenterons les travaux réalisés dans le cadre du développement d'un émulateur de réseau de capteurs autonomes communicant sans fil. Pour ce faire deux simulateurs seront développés :

- le premier sera un simulateur purement logiciel basé sur la création de modèles comportementaux, décrits en langage VHDL-AMS, et représentant chaque élément du réseau. Ces modèles seront ensuite interconnectés entre eux dans l'environnement "SystemVision" de "Mentor Graphics" suivant une architecture à tester pour la simuler. [72]
- le deuxième sera un simulateur matériel basé sur la création de modèles logiques comportementaux, décrits en langage VHDL synthétisable, de chaque élément du système. Une fois l'architecture du réseau définie (choix et interconnexion des éléments entre eux), celle-ci est compilée puis synthétisée dans l'environnement approprié. Le fichier de programmation obtenu est téléchargé sur un composant FPGA cible dans lequel s'effectuera la simulation. [73]

La fonction essentielle de ces émulateurs sera d'abord de vérifier et décider sur la pertinence d'une configuration ou architecture de réseau de capteurs pour répondre à un besoin précis décrit dans un cahier des charges. Pour ce faire et afin de disposer de marges de manœuvre suffisantes, les modèles constituant les différentes architectures doivent être génériques et prendre en compte un certain nombre de paramètres,

exprimés en termes de fonctions de coût, ayant une incidence sur les performances finales (consommation, temps d'exécution, …). Ainsi pour avoir des modèles génériques couvrant tous les cas de figures sans alourdir la simulation nous nous sommes orientés vers le développement de modèles comportementaux intégrant des paramètres essentiels pour l'évaluation des performances d'une architecture. Ces modèles peuvent être des modèles de composants, matériels ou virtuels, du réseau ou des modèles de « sondes » de mesures (consommation, latence, taux d'erreur de bits, …). Lors des simulations il est alors possible de suivre l'évolution des divers paramètres importants et de statuer sur l'adéquation de l'architecture choisie.

L'autre objectif de ce travail sera, grâce à notre façon de structurer les modèles de base, de mettre en place une approche type « meet in the middle » pour guider le choix des technologies qui matérialiseront le modèle logico-temporel. Cette approche est une méthode itérative de confrontation pas à pas entre les choix technologiques et la simulation-évaluation.

Identification des éléments constitutifs d'un réseau de surveillance

i. Le capteur

Les capteurs sans fils sont des éléments indépendants les uns des autres, et doivent disposer d'une alimentation autonome. Leur durée d'utilisation est limitée par la durée de vie de leur batterie. Cette contrainte forte a une influence majeure sur l'ensemble des techniques mises en place pour le déploiement de tels réseaux. Un effet majeur de cette limitation énergétique est la limitation maximale des transmissions par voie hertzienne, très coûteuses en énergie. Il est donc primordial d'effectuer autant que possible le traitement de l'information localement au niveau du nœud ou bien de chercher des sources intégrées pour le rechargement de ces batteries, (cellules photovoltaïques, micro éoliennes, accumulateur piézo-électrique dans le cas de présence de vibrations…)

ii. Le nœud

Le nœud est un élément qui regroupe plusieurs capteurs. Il a pour tâche essentielle d'agréger les données issues de capteurs et de générer une trame constituée :

- d'un en-tête
- de données utiles organisées selon les besoins du protocole de communication
- d'une datation
- d'ajouter un code correcteur d'erreur
- d'envoyer la trame ainsi constituée via une communication sans fil et un type de modulation vers un routeur.

iii. Le canal de transmission

Le canal de transmission est la voie par laquelle les données sont transmises d'un émetteur vers un récepteur. Cette partie de la ligne de transmission est la plus difficile à modéliser car certaines données sont perdues et / ou modifiées en fonction de la qualité du média ... Le comportement de cet élément est fonction de nombreux facteurs: le type de modulation, le débit des données, la température ambiante, l'environnement, le nombre d'émetteurs récepteurs utilisant ce canal, le bruit, la distance émetteur-récepteur, la puissance d'émission,...

iv. Le routeur

Dans la plupart des réseaux de capteurs la communication est ascendante vers le concentrateur (la destination finale) ou descendante (pour une configuration) et pas entre les capteurs. Le routeur joue un rôle de collecteur de données des différents nœuds ou d'autres routeurs pour les retransmettre vers le concentrateur. Il peut être un simple répéteur avec ou sans fonctions de correction d'erreurs.

v. Le concentrateur

Le concentrateur est le centre de surveillance, il collecte l'ensemble des données issues des différents capteurs qui composent le réseau. Il applique les règles de correction incluses dans les codes correcteurs, a une fonction d'analyse des données, de représentation et de stockage de masse. En général il dispose d'une IHM plus ou moins sophistiquée et a également pour rôle la configuration du réseau.

vi. Sondes de mesure

En plus des éléments composant le réseau, des sondes de mesure doivent être ajoutées pour scruter les paramètres fondamentaux caractérisant les performances des architectures sous test. On peut citer parmi ces paramètres : le taux d'erreurs de bits, le nombre de paquets perdus, les retards (latence), la consommation énergétique, la désynchronisation, …

1.4.3 La vision software de la simulation

Un réseau de capteurs sans fil est un système complexe qui réunit plusieurs domaines : l'électronique analogique et numérique, l'instrumentation, les technologies sans fil, les protocoles de communication, le traitement du signal, le thermique ... Pour simuler un tel système dans un environnement unique, nous avons utilisé le langage VHDL-AMS comme langage de description. Celui-ci prend en charge la description des systèmes à signaux mixtes (continus et discontinus) et est capable de décrire les systèmes multi-disciplines, notamment électriques, mécaniques, hydrauliques, thermiques, etc., à différents niveaux de complexité. Il est donc possible d'échanger et de combiner des modèles de différents domaines sans devoir les traduire manuellement.

Le simulateur consiste à

- Interconnecter différents modèles représentant les éléments du réseau selon une architecture choisie en regard de l'application à réaliser.
- Paramétrer ces modèles en fonction des contraintes du cahier des charges et des marges de manœuvre disponibles.
- Définir, via des sondes de mesure, les grandeurs du réseau à observer et rendant compte de l'efficacité de l'architecture choisie.
- Simuler l'architecture et accepter ou modifier les paramètres de réglage voire tester une autre architecture.

1.4.4 La vision hardware de la simulation

Le simulateur software décrit précédemment offre une exploration architecturale convenable en regard des critères à étudier. Cependant du fait qu'il reste une simulation dans un environnement type PC, les temps de simulation sont intimement liés aux performances de la machine (vitesse du processeur, lectures-écritures en mémoire, …) et se trouvent très dégradés lorsque les réseaux simulés atteignent plusieurs centaines de capteurs. Pour accélérer les temps de simulation et faciliter l'exploration architecturale on a recours à une simulation exécutée sur une machine matérielle dédiée. Dans ce cas les modèles utilisés restent des modèles comportementaux mais décrits en langage VHDL synthétisable plutôt qu'en VHDL-AMS.

Une architecture du réseau sera formée par l'association des différents modèles de base. Celle-ci, une fois compilée, sera synthétisée au niveau portes logiques puis téléchargée dans un circuit numérique cible type FPGA. Les temps de simulation se trouvent de fait fortement réduits (plusieurs centaines de fois) puisque les modèles comportementaux ont été remplacés par des circuits numériques réels. Par ailleurs **les simulations peuvent être faites en temps réel**, et certains paramètres liés aux modèles de l'architecture testée peuvent être modifiés à la volée en fonction des résultats constatés.

Sur la base de ce principe trois possibilités sont offertes :

- Créer une architecture « sur mesure » qui répond au besoin exprimé, la compiler puis la télécharger dans le circuit cible pour simulation.
- Créer une architecture maximaliste qui répond au déploiement d'un réseau comprenant un très grand nombre de capteurs, la compiler puis la télécharger également dans le circuit.
- Créer un ensemble d'architectures intermédiaires, chacune étant relativement bien adaptée à un type de problème.

Enfin les travaux effectués peuvent être représentés par la figure 1.8

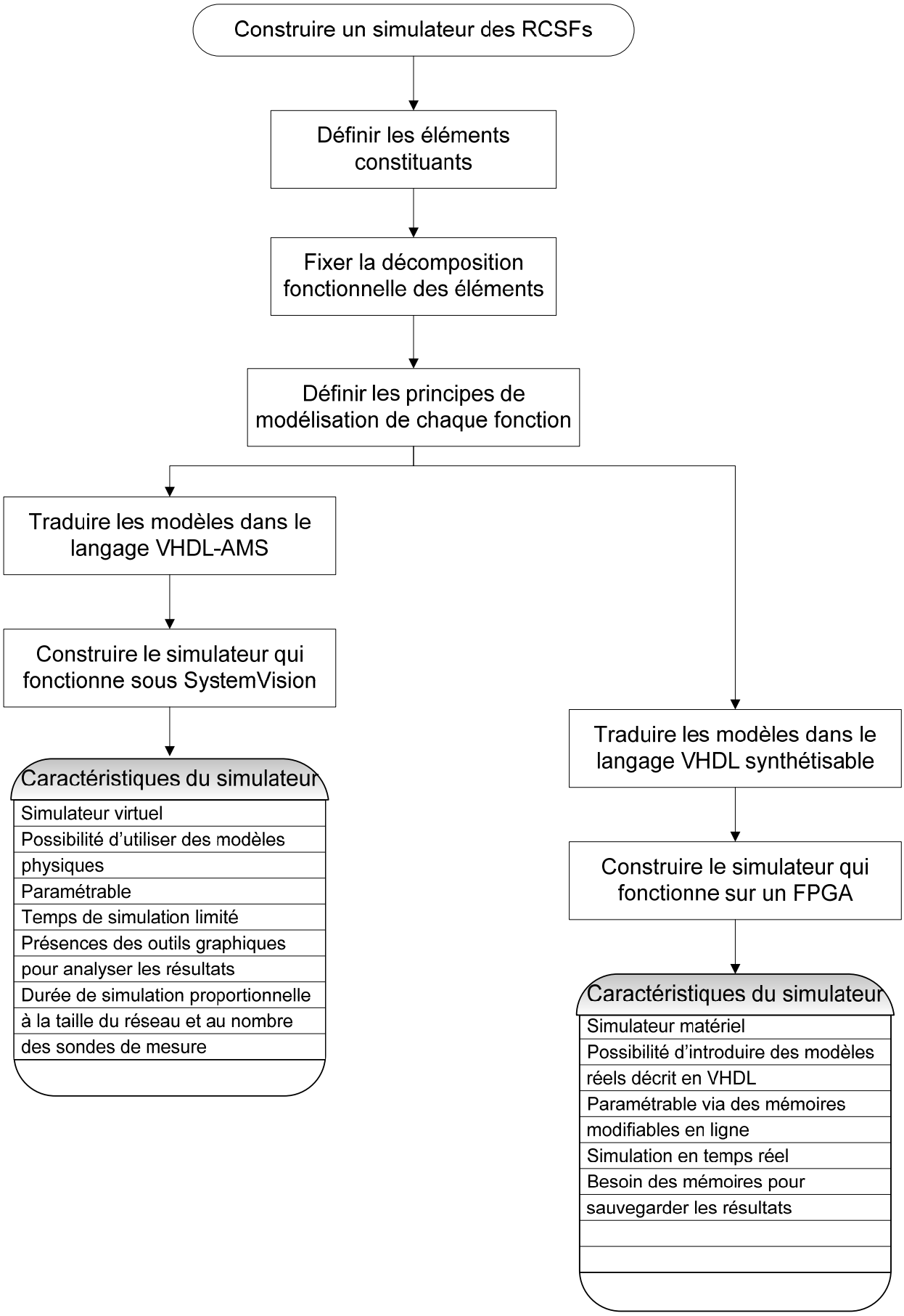


Figure 1. 8 Démarche suivie pour développer les deux simulateurs

1.5 Conclusion

Ce chapitre identifie et présente les difficultés principales intervenant dans la conception des systèmes de surveillance :

Il s'agit de systèmes de microsystèmes répartis, autonomes énergétiquement, communicants sans fil. Nous avons détaillé ensuite les domaines d'applications des systèmes de surveillance et des réseaux de capteurs correspondant par rapport aux technologies microsystèmes, en présentant les différentes caractéristiques des RCSFs et les techniques utilisées dans leur développement.

Nous avons présenté l'application qui nous sert de véhicule de test : le projet coopératif SACER de surveillance temps réel des structures aéronautiques.

Sur le plan du développement, il s'agit d'un système complexe dont la conception doit s'appuyer sur les méthodes et les outils de l'ingénierie des systèmes : nous avons brièvement rappelé les normes EIA-632 et situé notre contribution aux niveaux des choix architecturaux des réseaux, des méthodologies de modélisations-simulations de réseaux, des choix technologiques à mettre en œuvre pour couvrir les exigences du cahier des charges.

Un état de l'art des simulateurs de réseaux a été ensuite présenté et analysé par rapport aux besoins du projet SACER. Ces simulateurs ciblent généralement quelques points précis et souvent de manière exclusive: étude de protocoles par exemple, évaluation de l'énergie consommée pour un mode de fonctionnement, adéquation des ressources d'un nœud en regard d'une application donnée ou possibilité de fonctionnement dans un environnement donné. Ils ne permettent pas l'étude simultanée des différents points cités et ne permettent pas non plus de concevoir des nœuds propres à un projet ou d'évaluer des points importants tels que la synchronisation et la latence cumulée, …

Nous avons enfin présenté les contributions que nous proposons:

- La conception d'un simulateur de RCSF adapté à la demande des concepteurs
- La mise en évidence de son intérêt sur deux grands verrous :
    1. L'architecture réseau

2. Le choix des technologies en conformité avec les spécifications du réseau.

En conséquence nous avons présenté les spécifications d'une nouvelle approche de simulateurs caractérisés par la création de modèles comportementaux et paramétrables par des fiches techniques descriptives. Cette approche comprend deux volets : software et hardware qui seront détaillés dans les chapitres suivants.

Le chapitre 2 sera consacré à la présentation de la démarche suivie dans le développement de nos simulateurs et aux principales notions abordées dans ce développement.

Références

[1] EIA. Processes for Engineering a System : EIA-632. EIA Electronics Industries Alliance, Janvier 1999. 3, 9, 12, 28, 102.
[2] REVVA. VV&A methodological guidelines Reference Manual. Common Validation, Verification and Accreditation Framework for Simulation, 2004. disponible sur http://www.revva.eu/. 14, 28, 29.
[3] AFIS. Association Française d'Ingénierie Système. http ://www.afis.fr/. 10.
[4] Vincent ALBERT, Evaluation de la Validité de la Simulation dans le Cadre du Développement des Systèmes Embarqués, doctorat, Université Toulouse III - Paul Sabatier, 2009.
[5] Pasha, M.A., Derrien, S., Sentieys, O., A complete design-flow for the generation of ultra low-power WSN node architectures based on micro-tasking, Design Automation Conference (DAC), 2010 47th ACM/IEEE, pp 693 – 698.
[6] Hoermann Leander B., Glatz Philip M., Steger Christian, Weiss Reinhold, Energy Efficient Supply of WSN Nodes using Component-Aware Dynamic Voltage Scaling Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), 11th European, pp 1 – 8.
[7] Zhuancheng Zhang, Meng, M.Q.-H., Fuqing Wu, Xijun Chen, Design of WSN node based on CC2431 applicable to lunar surface environment, Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference, pp 1087 – 1092.
[8] Marino, P. ; Fontan, F.P. ; Dominguez, M.A. ; Otero, S., Test-bed measurements for ad-hoc WSN applications, Fourth International Conference on Communications and Networking in China, 2009. ChinaCOM 2009, pp 1 – 5.
[9] Mr. Patrice KADIONIK, Héni JMEL, Maxime CAUDRON, Aurélien BRISSET, Pierre Marc GUITARD, PROJET AVANCE SE : RESEAU DE CAPTEURS SANS FILS.
[10] Anguita, D. ; Brizzolara, D. ; Ghio, A. ; Parodi, G. , Smart Plankton: a Nature Inspired Underwater Wireless Sensor Network, Fourth International Conference on Natural Computation, 2008. ICNC '08, pp 701 – 705.
[11] Ze Li ; Haiying Shen ; Alsaify, B., Integrating RFID with Wireless Sensor Networks for Inhabitant, Environment and Health Monitoring, 14th IEEE International Conference on Parallel and Distributed Systems, 2008. ICPADS '08, pp 639 – 646.
[12] Su-Lim Tan ; Nguyen Ha Duy ; Garcia-Guzman, J. ; Garcia-Orduna, F., A wireless activity monitoring system for monkey behavioural study, IEEE 15th International Symposium on Consumer Electronics (ISCE), 2011, pp 40 – 45.
[13] Jia Xiangyu ; Wang Chao, The Security Routing Research for WSN in the Application of Intelligent Transport System, the 2006 IEEE International Conference on Mechatronics and Automation, pp 2318 – 2323.
[14] Maimour, M. ; Pham, C. ; Hoang, D., A Congestion Control Framework for Handling Video Surveillance Traffics on WSN, CSE '09. International Conference on Computational Science and Engineering, 2009, pp 943 – 948.

[15] Viani, F. ; Oliveri, G. ; Donelli, M. ; Lizzi, L. ; Rocca, P. ; Massa, A., WSN-based solutions for security and surveillance, 2010 European Microwave Conference (EuMC), pp 1762 – 1765.
[16] Hussain, M.A. ; khan, P. ; Kwak kyung Sup, WSN research activities for military application, 11th International Conference on Advanced Communication Technology, 2009. ICACT 2009, pp 271 – 274.
[17] Dessart, N. ; Fouchal, H. ; Hunel, P. ; Rabat, C., Simulation of large scale WSN for medical care, 2010 IEEE Symposium on Computers and Communications (ISCC), pp 1115 – 1120.
[18] Lawrence, E. ; Navarro, K.F. ; Doan Hoang ; Yen Yang Lim, Data Collection, Correlation and Dissemination of Medical Sensor Information in a WSN, Fifth International Conference on Networking and Services, 2009. ICNS '09, pp 402 – 408.
[19] Salvador, P. ; Nogueira, A. ; Valadas, R., Markovian Models for Medical Signals on Wireless Sensor Networks, IEEE International Conference on Communications Workshops, 2009. ICC Workshops 2009, pp 1 – 5.
[20] Messier, G.G. ; Finvers, I.G., Traffic models for medical wireless sensor networks , Communications Letters, IEEE, pp 13 – 15.
[21] Furtado, H. ; Trobec, R., Applications of wireless sensors in medicine, MIPRO, 2011 Proceedings of the 34th International Convention, pp 257 – 261.
[22] Baghyalakshmi, D. ; Ebenezer, J. ; Satyamurty, S.A.V., WSN based temperature monitoring for High Performance Computing cluster, International Conference on Recent Trends in Information Technology (ICRTIT), 2011, pp 1105 – 1110.
[23] Xinrong Zhang ; Bo Chang, Research of temperature and humidity monitoring system based on WSN and fuzzy control, International Conference on Electronics and Optoelectronics (ICEOE), 2011, pp V4-300 - V4-303.
[24] Garzon, C.A.L. ; Riveros, O.J.R., Temperature, humidity and luminescence monitoring system using Wireless Sensor Networks (WSN) in flowers growing, ANDESCON, 2010 IEEE, pp 1 – 4.
[25] Xiaoya Hu ; Bingwen Wang ; Xiaojun Hu, A Novel Energy-Balanced Time Synchronization Protocol in Wireless Sensor Networks for Bridge Structure Health Monitoring, 2nd International Workshop on Database Technology and Applications (DBTA), pp 1 – 5.
[26] Jianjun Niu ; Zhidong Deng ; Fengguang Zhou ; Zongsheng Cao ; Zhanqing Liu ; Fangzhou Zhu, A Structural Health Monitoring System Using Wireless Sensor Network, 5th International Conference on Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09, pp 1 – 4.
[27] Xuefeng Liu ; Jiannong Cao ; Youlin Xu ; Hejun Wu ; Yang Liu, A multi-scale strategy in wireless sensor networks for structural health monitoring, 5th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009, pp 361 – 366.
[28] Wijetunge, S. ; Gunawardana, U. ; Liyanapathirana, R., Wireless Sensor Networks for Structural Health Monitoring: Considerations for communication protocol design, IEEE 17th International Conference on Telecommunications (ICT), 2010, pp 694 – 699.
[29] Pin Nie; Zhihua Jin, Requirements, challenges and opportunities of wireless sensor networks in structural health monitoring, 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT), 2010, pp 1052 – 1057.
[30] Sohraby K, Minoli D, Znati T. Wireless Sensor Networks: Technology, Protocols, and Applications. John Wiley & Sons, Inc. 2007.
[31] Gislason D. Zigbee Wireless Networking. 2008. Newnes.
[32] Bhagwat, P., Bluetooth: technology for short-range wireless apps, Internet Computing, IEEE, pp 96 – 103.
[33] Shorey, R. ; Miller, B.A., The Bluetooth technology: merits and limitations, IEEE International Conference on Personal Wireless Communications, pp 80 – 84.
[34] Shizhuang Lin ; Jingyu Liu ; Yanjun Fang, ZigBee Based Wireless Sensor Networks and Its Applications in Industrial, IEEE International Conference on Automation and Logistics, 2007, pp 1979 – 1983.
[35] You Ke ; Liu Ruiqiang, Initialization of ZigBee Communication Module and Information Processing, Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 2008. PACIIA '08, pp 617 – 620,
[36] You Ke ; Liu Ruiqiang ; Zhang Cuixia, Work Mode of ZigBee WSN, International Conference on Information Management, Innovation Management and Industrial Engineering, 2008. ICIII '08, pp 536 – 538.
[37] Zensys. Sensor Networks - Z-Wave. 2007. sur: "http://www.zen-sys.com".

[38] http://www.ti.com/tool/simpliciti. visitée en 2012
[39] http://www.microchip.com/stellent. visitée en 2012
[40] P. Levis, N. Lee, M. Welsh, D. Culler, TOSSIM: accurate and scalable simulation of entire TinyOS applications, Proc. 1st ACM Conf. on Embedded Networked Sensor Systems (2003) 126–137.
[41] Julien Champ, Vincent Boudet, ADNL: Accurate Distributed Node Localization Algorithm in Wireless Sensor Networks, *European Wireless 2010*, Italie (2010).
[42] A. Hakiri, P. Berthou, J. Henaut, D. Dragomirescu, T. Gayraud, Performance Evaluation of Wireless Sensor Network for Spacial and Aeronautic Systems, International Conference on Telecommunications (IEEE ICT 2010), pp 879 - 886.
[43] Junguo Zhang ; Wenbin Li ; Dongxu Cui ; Xueliang Zhao ; Zhongxing Yin, The NS2-Based Simulation and Research on Wireless Sensor Network Route Protocol, 5th International Conference on Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09, pp 1 – 4.
[44] Zeng, X. ; Bagrodia, R. ; Gerla, M., GloMoSim: a library for parallel simulation of large-scale wireless networks, Twelfth Workshop on Parallel and Distributed Simulation, 1998. PADS 98. Proceedings, pp 154 – 161.
[45] Berthe, A. ; Lecointre, A. ; Dragomirescu, D. ; Plana, R., Simulation Platform for Wireless Sensor Networks Based on Impulse Radio Ultra Wide Band, Eighth International Conference on Networks, 2009. ICN '09, pp 61 – 66.
[46] Yahaya, F.H. ; Yussoff, Y.M. ; Rahman, R.A. ; Abidin, N.H., Performance analysis of Wireless Sensor Network, 5th International Colloquium on Signal Processing & Its Applications, 2009. CSPA 2009, pp 400 – 405.
[47] Mohanty, S. ; Patra, S.K., Quality of service analysis in IEEE 802.15.4 mesh networks using MANET routing, International Conference on Computing Communication and Networking Technologies (ICCCNT), 2010, pp 1 – 7.
[48] Hammoodi, I.S. ; Stewart, B.G. ; Kocian, A. ; McMeekin, S.G., A Comprehensive Performance Study of OPNET Modeler for ZigBee Wireless Sensor Networks, Third International Conference on Next Generation Mobile Applications, Services and Technologies, 2009. NGMAST '09, pp 357 – 362.
[49] Xiaodong Xian; Weiren Shi; He Huang, Comparison of OMNET++ and other simulator for WSN simulation, 3rd IEEE Conference on Industrial Electronics and Applications, 2008. ICIEA 2008, pp 1439 – 1443.
[50] Henry, R.R. ; Kahan, S.H. ; Liu, J. ; Nicol, D.M., An implementation of the SSF scalable simulation framework on the Cray MTA, Seventeenth Workshop on Parallel and Distributed Simulation, 2003. (PADS 2003). Proceedings, pp 77 – 85.
[51] Sobeih, A. ; Hou, J.C. ; Lu-Chuan Kung ; Ning Li ; Honghai Zhang ; Wei-Peng Chen ; Hung-Ying Tyan ; Hyuk Lim, J-Sim: a simulation and emulation environment for wireless sensor networks, IEEE Wireless Communications 2006, pp 104 – 119.
[52] Sung Park, Andreas Savvides, Mani B. Srivastava, SensorSim: a simulation framework for sensor networks, Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems, pp 104 – 111.
[53] S. Park, A. Savvides, M.B. Srivastava, Simulating networks of wireless sensors, Proc. Winter Simulation Conference 2001 (2001) 1330–1338.
[54] I. Downard, Simulating sensor networks in ns-2, http://pf.itd.nrl.navy.mil/nrlsensorsim/.
[55] sQualnet: A Scalable Simulation Framework for Sensor Networks website, http://nesl.ee.ucla.edu/projects/squalnet/.
[56] J. Liu, L.F. Perrone, D.M. Nicol, M. Liljenstam, Simulation modeling of large-scale ad-hoc sensor networks, in: Proc. 2001 Simulation Interoperability Workshop, (2001).
[57] C. Mallanda, A. Suri, V. Kunchakarra, S.S. Iyengar, R. Kannan, A. Durresi, S. Sastry, Simulating wireless sensor networks with omnet++, http://csc.lsu.edu/sensor_web/simulator.html.
[58] S. Dulman, P. Havinga, A simulation template for wireless sensor networks, IEEE Int. Symp. on Autonomous Decentralized Systems, 2003, fast abstract
[59] A. Sobeih, W.-P. Chen, J.C. Hou, L.-C. Kung, N. Li, H. Lim, H.-Y. Tyan, H. Zhang, J-Sim: A Simulation and Emulation Environment for Wireless Sensor Networks, http://www.j-sim.org/.
[60] P. Baldwin, S. Kohli, E.A. Lee, X. Liu, Y. Zhao Modeling of sensor nets in ptolemy II, in: Proc. 3rd Int. Symp. On Information Processing in Sensor Networks, 2004, pp. 359– 368.
[61] RAJENDRA M. PATRIKAR, SUDHIR G. AKOJWAR, Neural Network Based Classification Techniques For Wireless Sensor Network with Cooperative Routing, 12th WSEAS International Conference on COMMUNICATIONS, Heraklion, Greece, July 23-25, 2008.

[62] G. Simon, P. Vo¨ lgyesi, M. Maro´ ti, A ´ . Le´deczi, Simulationbased optimization of communication protocols for largescale wireless sensor networks, Proc. IEEE 2003 Aerospace Conference 3 (2003) 1339–1346.

[63] B.C. Mochocki, G.R. Madey, H-MAS: a heterogeneous, mobile, ad-hoc sensor network simulation environment, in: Proc. 7th Annual Swarm Users/Researchers Conference, 2003.

[64] G. Chen, J. Branch, M.J. Pflug, L. Zhu, B. Szymanski, SENSE: A sensor network simulator, http://www.cs.rpi.edu/~cheng3/sense/.

[65] M. Karir, J. Polley, D. Blazakis, J. McGee, D. Rusk, J.S. Baras, ATEMU: a fine-grained sensor network simulator, in Proc. 1st IEEE Int. Conf. on Sensor and Ad Hoc Communication Networks, 2004.

[66] L.F. Perrone, D.M. Nicol, A scalable simulator for TinyOS applications, Proc. Winter Simulation Conference 2002 (2002) 679–687.

[67] S. Sundresh, K. Wooyoung, A. Gul, SENS: a sensor, environment and network simulator, in: Proc. 37th Annual Simulation Symposium, 2004, pp. 221–228.

[68] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, D. Estrin, Em*: a software environment for developing and deploying wireless sensor networks, CENS Technical Report 0034, 2003. http://research.cens.ucla.edu/.

[69] C. Kelly IV, V. Ekanayake, R. Manohar, SNAP: a sensor network asynchronous processor, Proc. 9th Int. Symp. On Asynchronous Circuits and Systems (2003) 24–35.

[70] Mauri Kuorilehto, Marko Hännikäinen, Timo D. Hämäläinen, Rapid design and evaluation framework for wireless sensor networks. Ad Hoc Networks 6 (2008) 909–935.

[71] J.L.BOIZARD, N.NASREDDINE, C.BAUD, D.GUIHAL, C.ROSSI, C.ESCRIBA, J.Y.FOURNIOLS. Méthodologie de conception des SOC-SIP hétérogènes : application à la conception d'un réseau de capteurs. Ecole d'Hiver Francophone sur les Technologies de Conception des systèmes embarqués Hétérogènes (FETCH 2008), Montebello (Canada), 7-9 Janvier 2008.

[72] N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS, J.HENAUT, D.DRAGOMIRESCU, A.COUSTOU. VHDL-AMS behavioral models for the simulation of wireless sensors networks. SSD'2009 : 6th International Multi Conference on Systems, Signals and Devices (SSD'09) International Conference on Sensors, Circuits and Instrumentation Systems (SCI), Djerba (Tunisie), 23-26 Mars 2009, IEEE, 2009, pp.214-216.

[73] N.NASREDDINE , J.L.BOIZARD , C.ESCRIBA , J.Y.FOURNIOLS. Synthesizable VHDL models for the simulation of a wireless sensors network. ECMS'2009 : 9th International Workshop on Electronics, Control, Modelling, Measurement and Signals (ECMS 2009), Mondragon (Espagne), 8-10 Juillet 2009.

Chapitre 2 : La modélisation des éléments matériels du réseau pour la création du simulateur

Chapitre 2. La modélisation des éléments matériels du réseau pour la création du simulateur

Les applications des Réseaux de capteurs sans fils (RCSF) sont diverses, allant des réseaux locaux et embarqués au déploiement de grands systèmes militaires. Les défis typiques des RCSF sont liés au fait qu'ils sont :

- A grande échelle (des centaines de capteurs dans le cas d'applications SHM),
- En constante évolution topologique,
- Soumis à la possibilité de communiquer des erreurs,

Alors que dans les nœuds, les capacités de traitement et de stockage, ainsi que les ressources énergétiques sont limitées. Le plus souvent, les réseaux de capteurs sont priés d'être robustes face aux contraintes de l'environnement et capables de recouvrir de manière autonome les situations d'erreurs. De plus, selon les applications visées et les interactions avec l'environnement, les exigences sur la synchronisation des mesures et la sécurité des horloges peuvent être élevées [1] [2].

Cependant, l'espace de conception (degrés de liberté pour le concepteur) est très large et rend l'automatisation de conception le plus important défi pour le bon fonctionnement des RCSFs. Un concepteur, tout simplement, ne peut pas gérer tous les paramètres, les fonctions et leurs dépendances complexes sans le support d'un outil [3].

Pour simuler un réseau de capteurs sans fils lors de la phase d'exploration architecturale, des modèles comportementaux de chaque élément doivent être créés : ces modèles sont des descriptions très haut niveau des fonctions supportées. Ainsi ces modèles peuvent remplacer, dans la simulation, les éléments réels du réseau à concevoir. Ils ont été créés comme des modèles généraux: ils peuvent être paramétrés pour remplacer l'élément dans une configuration d'architecture donnée en précisant par exemple le type de modulation, la technologie utilisée, l'architecture matérielle et ainsi de suite ...

Comme nous l'avons dit dans le précédent chapitre le réseau de capteurs sans fil étant un système multi-domaines, la modélisation en langage VHDL-AMS [4] est la

plus appropriée. Ainsi le système complet, ou une partie, peut être simulé, après définition de l'architecture correspondante, dans un seul environnement tout en profitant des trois types de connexion de VHDL-AMS : TERMINAL, QUANTITY et SIGNAL et de la caractéristique multi-domaine du langage, lequel permet de mélanger des modèles comportementaux, logiques, électriques et mécaniques. [5][6]

Il est alors possible, pour chaque construction architecturale choisie, d'effectuer une simulation (ici dans l'environnement SystemVision de Mentor Graphics), d'évaluer la performance associée et dresser un tableau comparatif de ces architectures.

Selon la taille du réseau et les facteurs intéressants comme résultats de simulation, la simulation du réseau peut être gourmande en ressources informatiques et en temps. Une autre solution pour le simulateur peut alors être envisagée, celle-ci est basée sur :

- la mise en œuvre de circuits numériques programmables type FPGA comme environnement de simulation (la description des modèles se fait alors en langage VHDL synthétisable).
- La mise en œuvre de stratégies multiples à partir de cette base (modèles de réseau très large échelle configurables in situ ou modèles réduits téléchargés à la demande par exemple) [7].

Cette implémentation du simulateur sur une plateforme matérielle, permet de réduire la durée de simulation de façon très appréciable, ce qui permet, à durée de simulation équivalente, de tester beaucoup plus efficacement une architecture de réseau, de mélanger dans l'architecture testée des composants réels et des modèles comportementaux (couches de communication, codes correcteurs, horloges, concentrateur…).

Dans ce qui suit nous décrivons les modèles comportementaux de tous les éléments qui composent un réseau de capteurs sans fil, en expliquant le principe de modélisation utilisé dans le simulateur développé en VHDL-AMS, puis en faisant apparaître certaines modifications introduites pour une adaptation au langage VHDL synthétisable.

2.1 Démarches spécifiques utilisées dans le simulateur sur FPGA

Pour la création des modèles VHDL des éléments du réseau, nous allons utiliser la même approche que celle utilisée dans le simulateur VHDL-AMS en l'adaptant au nouveau langage. La description fonctionnelle restera la même et seuls les algorithmes des fonctions internes seront légèrement différents. Dans cette version de simulateur et bien que parfaitement modélisables, toutes les fonctions modélisées en VHDL-AMS ne nécessitent pas forcement d'être modélisées car certains résultats ne nécessitent pas un long temps de simulation et leur degré de modélisation ne demande pas une longue durée de calcul pour les simuler : il suffit de multiplier un résultat par un facteur pour avoir celui-ci sur une plus longue durée de simulation (la consommation par exemple) ou bien c'est un résultat répétitif (la désynchronisation des nœuds par exemple). Ici nous nous sommes particulièrement intéressés aux valeurs des BER et PER, lesquelles sont difficilement observables sur des importants temps de simulation dans le simulateur VHDL-AMS.

La modélisation en matériel des éléments du système permet aussi d'intégrer les vrais éléments dans la procédure de simulation, soit par connexion de cet élément à la place de son modèle, soit par insertion de l'élément lui-même dans le simulateur, si celui-ci est décrit en VHDL. En effet certains modules sont développés en VHDL pour être synthétisés et implémentés : c'est le cas par exemple du module qui gère la communication entre l'émetteur et le récepteur par application du protocole retenu (Ethernet ou autre). Ce module, historiquement développé en software dans la plupart des applications, peut être développé en matériel pour répondre aux contraintes du temps réel.

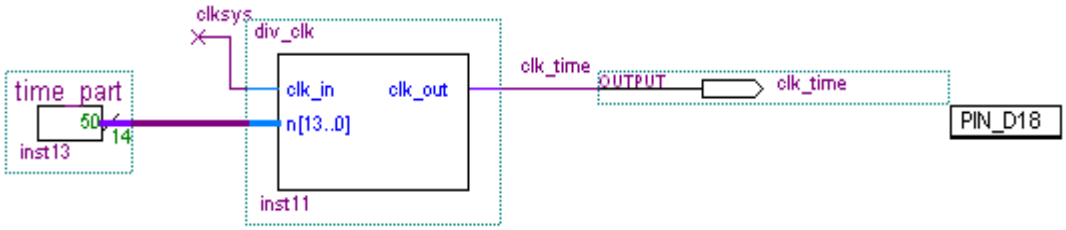
La compilation et la synthèse du fichier téléchargeable sur le FPGA étant une procédure consommatrice de temps, nous nous sommes orientés vers la création d'un jeu d'architectures large échelle, relativement génériques, dont l'essentiel des paramètres peut être modifié de manière dynamique pendant le fonctionnement du circuit via l'utilitaire « In System Memory Content Editor ». Cet aspect offre au concepteur une grande interactivité, laquelle facilite la convergence vers un

paramétrage optimal de l'architecture. Le concepteur dispose donc de 2 niveaux de conception :

- Le premier se situe au niveau du logiciel de synthèse et de génération du fichier de configuration, c.à.d. les modifications de l'architecture se font avant sa compilation.
- Le deuxième se situe au niveau du FPGA lui-même en venant modifier des contenus de mémoires ou de registres représentant certains paramètres de l'architecture.

Le passage d'une modélisation VHDL-AMS à une modélisation VHDL synthétisable entraine un certain nombre de difficultés de réalisation. C'est le cas pour l'introduction dans les modèles de phénomènes de retard, lesquels représentent physiquement des retards purs ou des retards liés à des temps de traitement des données. Dans le simulateur basé VHDL-AMS, il est fait par appel à l'instruction « wait » ou à l'attribut « 'delayed » pour retarder la propagation des signaux pendant des durées spécifiées dans des tableaux (lesquelles étant caractéristiques du nœud, du protocole de communication, du code correcteur d'erreur, du temps de traitement dans le concentrateur, …). Cette méthode n'étant pas exploitable en VHDL synthétisable, nous avons contourné le problème en injectant les données dans des pipe-lines basés sur la mise en œuvre de registres à décalage étant contrôlés par un signal de validation issu d'une machine à état. Le retard introduit par ce dispositif est un multiple, paramétrable, de l'unité de temps associée à la machine à état.

Le simulateur ayant besoin d'injecter des retards se situant dans une fourchette relativement large, nous avons généré plusieurs horloges obtenues par divisions successives et paramétrables de l'horloge de base de la carte utilisée Figure 2.1.




```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
ENTITY div_clk IS
    PORT (clk_in : IN  std_logic;
            n : in integer range 0 to 10000;
            clk_out : out std_logic
            );
END div_clk;
ARCHITECTURE arch_div_clk OF div_clk IS
signal s_clk_out : std_logic;
signal count : integer range 1 to 10000 :=1;
BEGIN
    gen_clk: process (clk_in)
    begin
        if clk_in'event and clk_in ='1'  then
            if count = n then
            count <= 1;
            s_clk_out <= not s_clk_out ;
            else  count <= count + 2;
            end if;
        end if;
    end process gen_clk;
    clk_out <= s_clk_out;
END arch_div_clk;
```

Figure 2. 1 Division de l'horloge de base

Parallèlement à ces deux aspects, le simulateur VHDL-AMS manipule des quantités continues, lesquelles doivent être remplacées convenablement par des valeurs discrètes, et des valeurs de type réel qui doivent être remplacées par des entiers (on peut par exemple utiliser des facteurs multiplicatifs pour exprimer des valeurs de BER sous forme de nombre entier)…

Un autre point lié au simulateur VHDL-AMS, est que les signaux et les grandeurs calculés sont enregistrés durant la simulation, dans des fichiers de simulation ou de sortie constituant ainsi un historique. Ces fichiers peuvent être ensuite analysés ou tracés sous forme de courbes dans l'utilitaire graphique associé à l'environnement de simulation. Dans le simulateur matériel, ces signaux sont des signaux numériques (électriques) échangés dans les conducteurs (ou bus de données).

Ainsi pour extraire de la simulation les résultats pertinents, il est nécessaire d'enregistrer ceux-ci dans des mémoires internes associées à des sondes de mesure, puis de venir les lire en temps réel via un utilitaire tel que le « In System Memory Content Editor ».

2.2 Proposition d'un modèle générique pour chacun des éléments

2.2.1 Le capteur

Le bloc capteur représente un phénomène physique transformé par un capteur en un signal électrique. Ce signal est ensuite amplifié et conditionné (filtres, réglage d'offset, calibration…) avant d'être converti en numérique par un convertisseur analogique-numérique et disposer ainsi d'une sortie série de la trame du capteur, figure 2.2. Ces éléments sont commandés par une unité de contrôle, des horloges et des alimentations. Le phénomène physique ainsi que le capteur et sa chaîne de conditionnement peuvent être définis par une relation mathématique, généralement linéaire. Le circuit de conditionnement est une succession d'amplification, de filtrage, de réglage d'offset… qui dépendent du type de capteur, de la technologie utilisée et de l'environnement d'utilisation : il peut être modélisé, comme le convertisseur, par des modèles électriques.

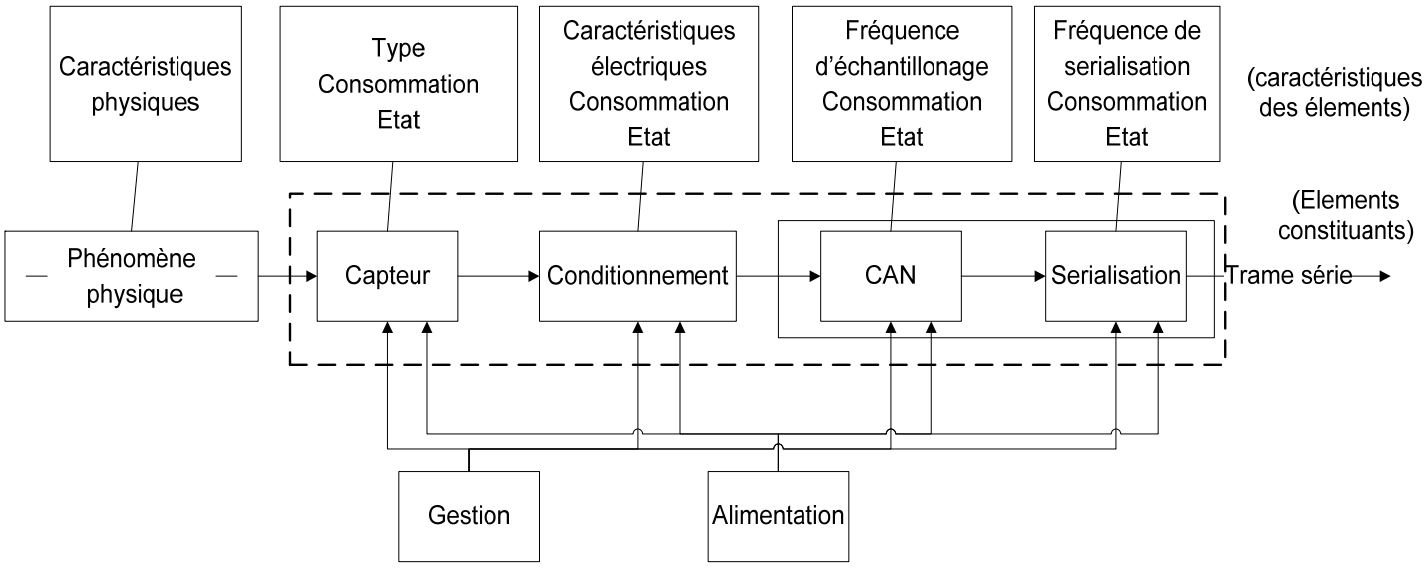


Figure 2. 2 Décomposition fonctionnelle du capteur

Ce niveau de modélisation impose un modèle propre pour chaque type de capteurs et exclut la possibilité de le rendre générique. Ce degré de modélisation est intéressant pour les concepteurs de la chaîne de traitement, mais notre problématique de simulation ne se situe pas à ce niveau : elle se situe à un niveau plus général : celui de la faisabilité du réseau, donc un choix d'architecture, par rapport à des exigences client précises. Dans ce contexte les capteurs et le conditionnement associé, déjà définis et fabriqués, ne constituent pas des degrés de liberté: il suffit de leur associer au niveau des nœuds, la partie correspondant à la communication sans fils. Dans notre simulateur, les capteurs sont considérés comme des produits sur étagère et insérés dans le réseau bien que cela n'empêche pas de leur associer un modèle bas niveau, à condition que les bornes de connexions extérieures soient les mêmes.

Enfin pour créer un modèle générique de capteur à un niveau hiérarchique supérieur, nous avons procédé comme suit : le phénomène physique mesuré, conditionné et converti est modélisé par une mémoire dont le contenu est balayé périodiquement en fonction des fréquences d'échantillonnage et de sérialisation des données qui doivent être en cohérence. Le contenu de la mémoire, bien qu'il puisse être quelconque, est représentatif des signaux physiques à simuler: dynamique, rapport signal/bruit, bande passante ... Ce contenu peut être facilement obtenu à partir d'un outil, tel Matlab, en appliquant une transformée de Fourier inverse au gabarit du spectre du signal réel. Ce contenu, dont le spectre est connu, car issu du cahier des charges du réseau, nous donne la possibilité de filtrer le signal à la réception, facilitant ainsi la suppression d'artefacts qui n'auraient pu être corrigés par le code correcteur d'erreurs. Les effets induits par les éléments formant le bloc capteur sont introduits par les paramètres descriptifs qui lui sont associés : état, latence liée au conditionnement, latence de conversion et de sérialisation, profil de consommation, fréquence d'échantillonnage et de sérialisation. Ces éléments peuvent être définis dans les spécifications, dans le datasheet du capteur ou mesurés sur maquette réelle, figure 2.3.

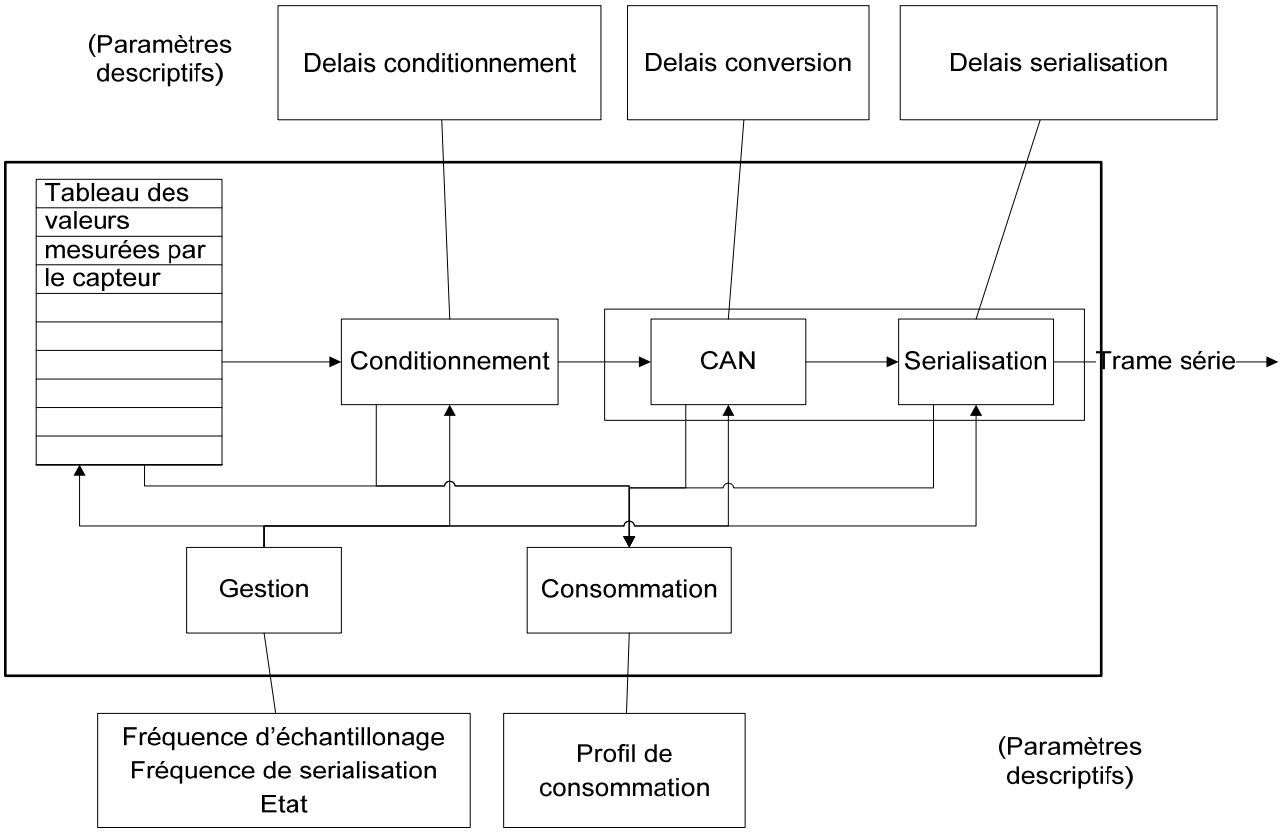


Figure 2. 3 Modèle général du capteur

Autrement dit, dans l'approche que nous proposons, le système est fondamentalement représenté par une architecturation de ses éléments constitutifs. Cela peut résulter d'une analyse descendante jusqu'à un niveau de conception que l'on jugera utile pour le problème traité. Ce modèle de base, ici le capteur, est simulé fonctionnellement par l'injection de données (table des valeurs mesurées par le capteur) et par un gestionnaire fonctionnel qui fixe les fréquences d'échantillonnage, de sérialisation…

A ce modèle de base nous proposons d'associer des fonctions locales, représentatives des facteurs d'influences, sur chaque élément du système et des fonctions locales de mesure : chacune de ces fonctions locales devant faire l'objet d'une modélisation qui va permettre l'observation du système par le concepteur pour l'évaluer.

Nous appliquons ces principes pour le capteur au niveau de l'évaluation de la consommation et de la latence. L'introduction d'autres fonctions reste possible si l'utilité s'en fait sentir. Cette démarche de modélisation sera appliquée sur tous les éléments du réseau.

Une partie du modèle général, en VHDL-AMS, du premier bloc composant le capteur (la mémoire balayée en fonction de la fréquence d'échantillonnage et celle de conversion), est créée comme indiqué dans la figure 2.4. A ce bloc on associe les blocs correspondants aux autres fonctions pour former le capteur:

```
library WORK;
use WORK.pack_capt_3.all;
entity capt_test_comment is
  generic (
    table_lenght: integer := 16;--taille par défaut
    -- contenu par défaut
    capt_table: table_capt2 := ("0000000000000010","0000000000000100",
    t_ech: real := 47.0E-6;      --valeur par défaut
    bit_rate: real := 100.0E6;   --valeur par défaut
    bit_time: real := 100.0E-9   --valeur par défaut
  );
  port (
    signal out_capt: out std_logic
  );
end entity capt_test_comment;
architecture default of capt_test_comment is
signal stream: data_capt;
begin
        stream_out : process is
            variable bit_period: time;
            variable vect_period: time;
        begin
            bit_period := 1.0/bit_rate * sec;
            vect_period:= (t_ech-16.0/bit_rate)* sec;
            loop
                for j in 0 to table_lenght-1 loop
                    stream <= capt_table(j);
                        for i in 0 to 15 loop
                            out_capt <= stream(i);
                            wait for bit_period;
                        end loop;
                    out_capt <= '0';
                wait for vect_period;
                end loop;
            end loop;
          end process stream_out;
```

Figure 2. 4 Partie du modèle VHDL-AMS du capteur

Ce modèle général peut être paramétré pour correspondre au capteur réel mis en œuvre. Ainsi, en fonction des besoins et du contexte, on peut choisir de l'activer ou non, fixer le débit binaire, la période d'échantillonnage, le type de capteur, son profil de consommation et de délai de réponse, ceci en choisissant parmi des paramètres prédéfinis, ou bien définir de nouveaux profils si les capteurs à utiliser ne figurent pas dans la liste de choix. La figure 2.5 illustre le symbole d'un tel modèle avec un extrait des paramètres modifiables et un exemple de l'évolution de sa sortie au cours du temps.

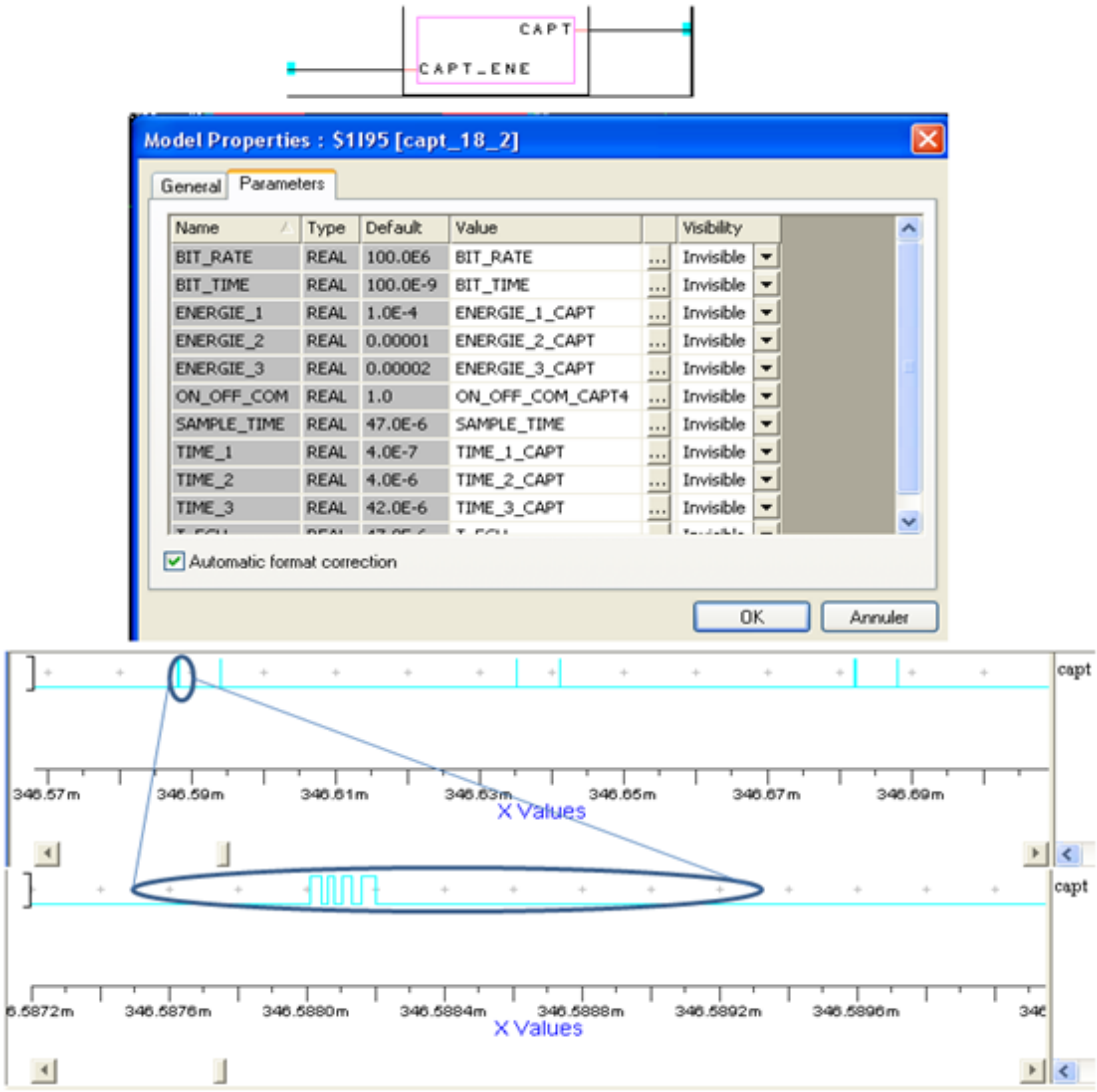


Figure 2. 5 Le symbole graphique du capteur et la fenêtre des paramètres

Fonction de la consommation de l'énergie du capteur:

Le capteur peut être dans l'un des trois états suivants : en veille, mesure, conditionnement. Pour chacun de ces états, et selon le type de capteur, un profil de consommation est associé. Sur cette base le module de consommation d'énergie indique la consommation instantanée en totalisant les consommations de chaque élément du capteur.

Modélisation du capteur en VHDL

En VHDL, le modèle du capteur décrit précédemment, peut être traduit par la machine à état qui suit, Figure 2.6, laquelle est commandée par 2 horloges : la première, sample_clk, correspondant à la fréquence d'échantillonnage et la seconde,

conv_clk, à celle de la sérialisation (qui se traduit par la fréquence du débit série des bits de sortie). La Figure 2.7 en donne le code VHDL.

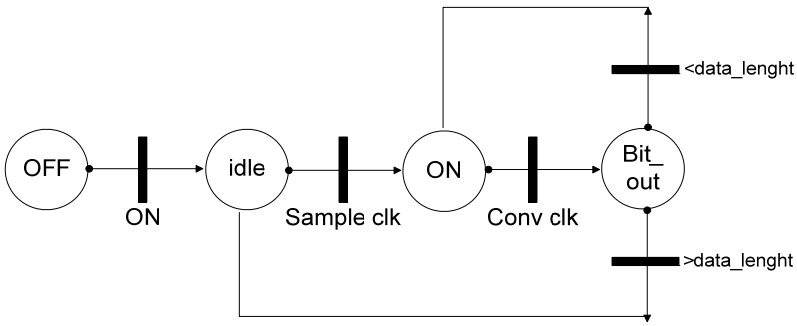


Figure 2. 6 Machine à état du capteur

```
ARCHITECTURE capt_data_architecture OF capt_data IS
type states is (off, idle, actif);
signal stream: data_capt;
BEGIN
stream_out : PROCESS  (clk_conv)
    variable status  : states  := off;
    variable j : integer :=0;
    variable i : integer :=0;
begin
    IF ((clk_conv'EVENT) AND (clk_conv = '1') )THEN
case status is
      when off   =>              -- initial off test to go on
        if on_off = '1' then
        status          := idle;
        out_capt <= '1';
        end if;
      when idle =>
        out_capt <= '0';
        IF  (clk_echant = '1') THEN
            stream <= capt_table(j);
            status          := actif;
        end if;
      when actif  =>
            out_capt <= stream(i);
            if i < DATA_LENGHT-1 then
                i:= i+1;
            else
                i:=0;
                status := idle;
                if j < TABLE_LENGHT-1 then
                    j:= j+1;
                else
                    j:=0;
                end if;
            end if;
    end case;
end if;
  end process stream_out;
END capt_data_architecture;
```

Figure 2. 7 Code VHDL du modèle capteur

Ce modèle général n'est pas paramétrable directement mais via ses horloges externes (fréquences d'échantillonnage et de sérialisation qui doivent être cohérentes),

lesquelles sont configurées pour correspondre au fonctionnement réel du capteur en cours d'utilisation.

2.2.2 Le nœud

Un nœud du réseau de capteurs peut être décomposé en quatre sous-systèmes principaux. Il s'agit des sous-systèmes de calcul, de communication, de captage et de gestion de l'énergie. Le sous-système de calcul a pour tâches d'insérer un en-tête (datation + adresse identifiant), d'organiser et coder les données selon les besoins du protocole de communication et d'ajouter un code correcteur d'erreur. Le sous-système de communication envoie ces données en utilisant une liaison sans fil au routeur selon un type de modulation défini. Le sous-système de captage peut regrouper un ou plusieurs capteurs. Le sous-système de gestion de l'énergie fournit l'énergie et gère la consommation, la recharge de la batterie, l'activation ou la désactivation des capteurs…

Pour effectuer ces fonctions, le bloc doit utiliser des mémoires, des multiplexeurs, des bascules et des circuits pour la sérialisation, la modulation et l'émission. Si l'on introduit, dans l'étape de simulation, ce niveau de représentation, il devient nécessaire d'utiliser de nombreuses équations mathématiques et/ou des modèles électriques à granularité fine des éléments constitutifs... L'exploitation des phénomènes physiques pour décrire le modèle, complique l'obtention d'un modèle **générique** utilisable dans tous les types d'architectures et impose des contraintes supplémentaires par l'augmentation drastique de la durée de simulation.

A titre d'exemple la figure 2.8 représente une partie de la simulation d'un modèle physique bas niveau de nœud dont l'entrée correspond à un signal physique et la sortie à la commande de l'unité de communication. Cette dernière est basée sur une modulation OOK avec les paramètres suivants :

- Débit des data continu de 10 Mbits/s
- Fréquence porteuse de 50MHz
- Nœud de 8 capteurs

La simulation sur un PC (dual core, 2 GHz) pour une durée de temps de 10ms prend 6 heures et génère un fichier de 800MBytes de données.

Ce simple résultat signifie que nous sommes incapables d'effectuer une exploration architecturale pour un grand système dans des temps acceptables avec ce type de modèle. Donc l'idée de modéliser les différentes fonctions au niveau physique, pour voir l'influence des perturbations (atténuation, bruits…) n'est pas appropriée.

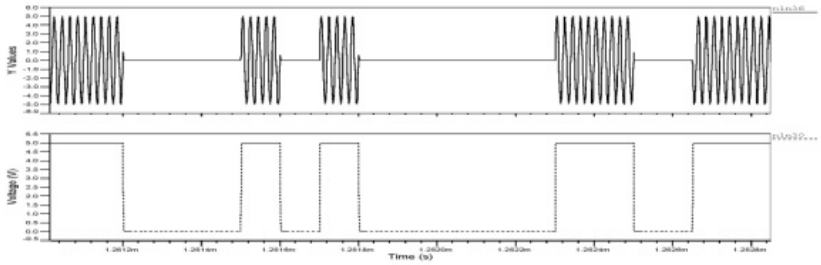


Figure 2. 8 Simulation d'un modèle avec modulation OOK

Il est donc essentiel de disposer d'un modèle comportemental de plus haut niveau pour ces fonctions. En effet, il n'est pas nécessaire, à cette étape de la conception, de moduler les données de sortie pour les transmettre et de les démoduler dans le récepteur pour les récupérer dans le bloc suivant. Au lieu d'échanger les données modulées, il suffit de les échanger directement en introduisant les éventuels effets induits. Le modèle développé permet d'atteindre cet objectif sans pour autant ignorer ce qui se passe au cours de ce processus : puissance consommée, datation interne, retard cumulé, taille des données ...

Ainsi le modèle comportemental est lui-même composé des sous-modèles:

- Ajout de l'entête et du code correcteur d'erreurs, figure 2.9.
- Sérialisation et retard, figure 2.9.
- Consommation d'énergie.
- Synchronisation/désynchronisation des nœuds.

Adresse Concentrateur 8 bits	Type de trame 4 bits	Date 24 bits	Statut du nœud 4 bits	Donnée capteur 1 16 bits		Donnée capteur 8 16bits	Statut capteur 8 bits	FEC TPC 176 bits

Figure 2. 9 Constitution d'une trame au niveau du nœud (SACER)

Afin de le rendre générique, le modèle de nœud peut gérer jusqu'à 8 capteurs, lesquels peuvent être activés ou désactivés à la demande.

Ce modèle de nœud peut également être paramétré pour se conformer aux caractéristiques de l'architecture réelle. Ainsi, en se basant sur le cahier des charges du projet, il est possible de choisir :

- Le nombre de capteurs associés à ce nœud
- Le type de capteur ainsi que le type de communication
- La datation et le code de correction (la forme de la trame)
- Les profils de consommation et de délai de chaque élément.
- ….

Ces paramètres peuvent être choisis dans une bibliothèque prédéfinie ou fixés par l'utilisateur. La figure 2.10 illustre le tableau de saisie d'une partie des paramètres du modèle du nœud. Ce modèle est conforme à l'approche décrite dans le paragraphe précédent. Des fonctions supplémentaires peuvent être ajoutées comme par exemple la synchronisation et la consommation...

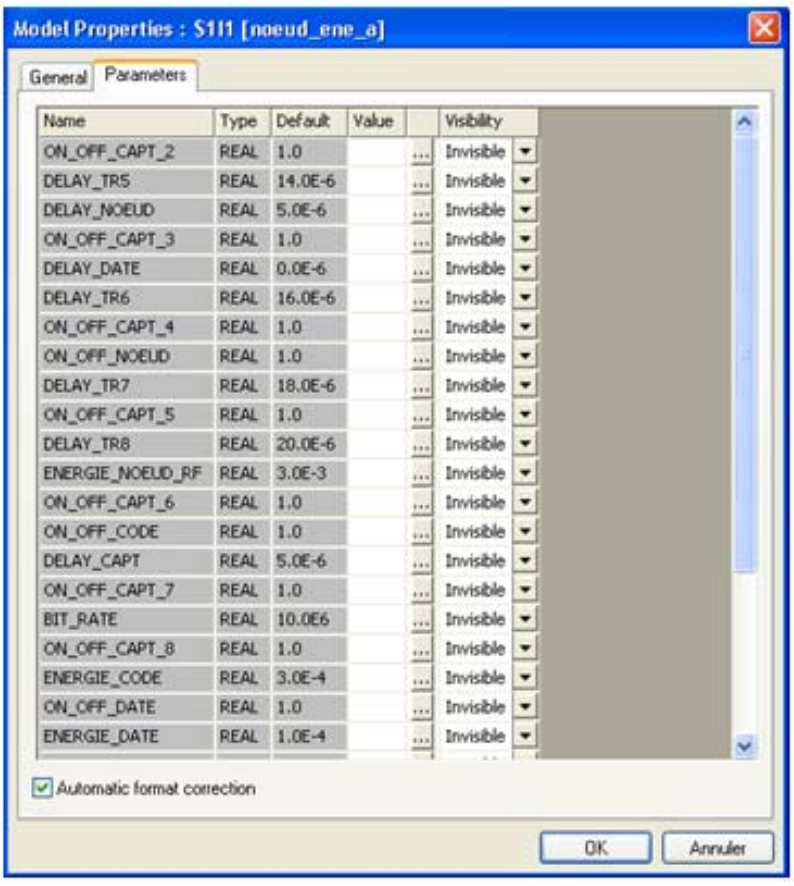


Figure 2. 10 Partie des paramètres du modèle du nœud

Etude de la synchronisation des nœuds

Dans les réseaux de capteurs, l'indépendance entre nœuds et leur autonomie s'avèrent des facteurs importants, cependant dans certaines applications la synchronisation des valeurs mesurées peut être également un facteur critique. Pour cela une procédure de synchronisation des horloges internes de chaque nœud doit être envisagée, soit en se référant par exemple à une horloge externe qui utilise l'horloge GPS (mais cette solution est en général inintéressante à cause de son importante consommation d'énergie), soit en utilisant un protocole de synchronisation envoyé par le concentrateur par voie descendante.

Le protocole de synchronisation consiste à échanger un certain nombre de messages entre le concentrateur et les nœuds, figure 2.11. Ce nombre de messages dépend du protocole utilisé, du nombre de trames à échanger durant un cycle de synchronisation et de la taille du réseau [8], ce qui permet d'avoir une précision de synchronisation, et impose par conséquent une certaine consommation d'énergie et une durée d'application.

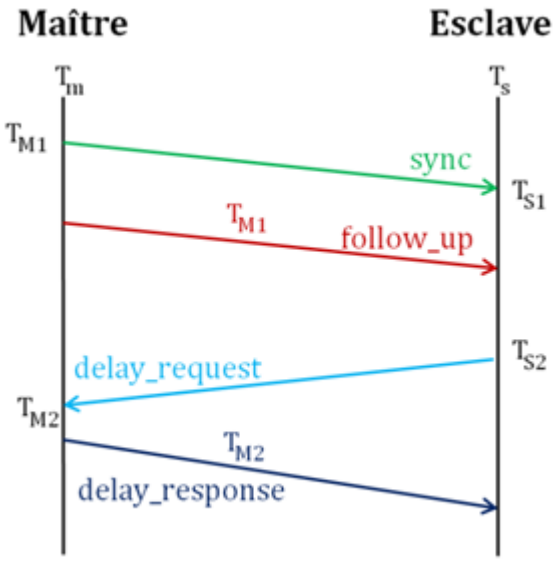


Figure 2. 11 Messages échangés par un protocole de synchronisation [8]

Au niveau de la simulation comportementale, le protocole de synchronisation est alors caractérisé par sa précision, sa consommation au niveau du nœud, sa fréquence et sa durée d'application, le tableau 2.1 présente des exemples de protocoles de synchronisation. L'intérêt de la simulation, est donc d'évaluer l'efficacité d'un mode

de synchronisation choisi (protocole et fréquence) pour voir s'il y a nécessité de revoir l'organisation du réseau.

Tableau 2. 1 Exemples de protocoles de synchronisation [8]

Protocoles	Caractéristiques		
	Nombre de messages	*Précision de la synchronisation*	*Consommation d'énergie*
IEEE 1588 (PTP- Precision Time Protocol)	4×N×L	200 ns	Haute
RBS (Reference Broadcast Synchronization)	$N \times L^2$	29.1 µs	Haute
TPSN (Timing-sync Protocol for Sensor Networks)	2×N×L	16.9 µs	Moyenne
PBS (Pairwise Broadcast Synchronization)	2×N	29.1 µs	Faible

Où L est le nombre de paquets échangés dans un cycle de synchronisation et N le nombre de nœuds du réseau.

i. Fonction de désynchronisation des horloges :

Pour tester les protocoles de synchronisation nous avons modélisé le phénomène de désynchronisation. Pour cela nous avons proposé deux sources de décalage entre les fronts des différentes horloges figure 2.12:

1. Imprécision des oscillateurs : variation de la période de l'horloge
2. Décalages causés par l'asynchronisme des diviseurs de fréquence (jitter)

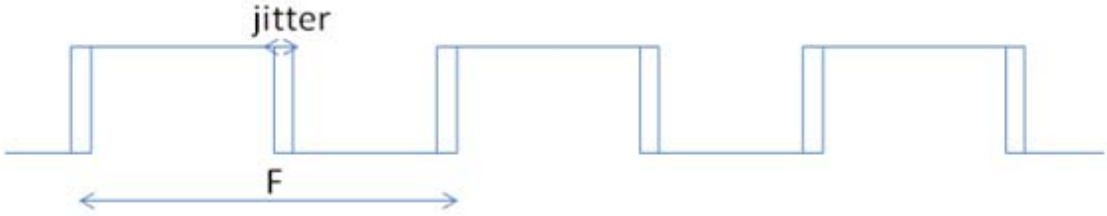


Figure 2. 12 Sources de désynchronisation des nœuds

Nous avons donc créé des modèles d'horloge, figure 2.13, qui peuvent subir ces perturbations de façon aléatoire, lesquelles sont limitées par une valeur maximale de jitter et une valeur maximale de variation de fréquence, ces valeurs étant caractéristiques des horloges utilisées. Ces horloges sont périodiquement resynchronisées par un signal de synchronisation résultant de l'application du

protocole de synchronisation. Ce signal a pour effet de supprimer le cumul des perturbations afin de ne pas dépasser la valeur limite autorisée par le cahier des charges.

```
    generic (period     : time := 10 ns;       -- Clock period
             seed1_init : positive := 53;      -- Initial value of seed1
             seed2_init : positive := 632;     -- Initial value of seed2
             jitter_max : real := 0.5e-9);     -- Max. (peak) jitter fro
    port    (   reset  : in  std_logic;
                clk_out : out std_logic:='0');
end entity clk_synch;
architecture behavioral of clk_synch is
constant jitter_max_time : time := integer(jitter_max * 1.0e15) * 1 fs;
begin
CreateClock: process
variable out_variable : std_logic;
variable seed1 : positive := seed1_init;
variable seed2 : positive := seed2_init;
variable scale_jit_0_1 : real := 0.5;    -- Random number, jitter scale
variable jitter_real : real := 0.0;      -- Jitter offset of next edge (
variable jitter_time : time := 0.0 ps;   -- Jitter offset of next edge (
variable period_var : time := period;
begin
    out_variable := '1';
    clk_out <= out_variable;
    clk_loop : loop
        if reset = '1' then
        clk_out <= '0';
        period_var:= period;
        out_variable:='1';
        wait for (jitter_max_time);
        else
         wait for (period_var/2.0 -  jitter_max_time);    --wait for (pe
            uniform(seed1, seed2, scale_jit_0_1);
            jitter_real := 2.0*jitter_max*scale_jit_0_1;
            jitter_time := integer(jitter_real * 1.0e15) * 1 fs;
            out_variable := not out_variable;
            period_var:= period_var+jitter_time;
            clk_out <= out_variable after jitter_time;
           wait for (jitter_max_time);
        end if;
        end loop clk_loop;
```

Figure 2. 13 Insertion de jitter dans les horloges

ii. Fonction de la consommation de l'énergie du nœud :

Le nœud peut être dans l'un des quatre états suivants : en veille, mesure, traitement, communication. Pour chacun de ces états chaque élément du nœud a une consommation spécifique. Ces valeurs sont également caractéristiques du nœud utilisé et le module de consommation en cumulant les valeurs et selon le contexte indique la consommation instantanée du nœud.

Après la configuration et le paramétrage du réseau à simuler, les sondes de mesure sont placées aux endroits critiques (au niveau des capteurs ou des nœuds). Cela permet de visualiser le profil de la puissance instantanée consommée et en cumulant ces consommations instantanées il est possible de déterminer l'énergie consommée par chaque nœud, estimer la durée de vie des batteries et affiner le choix des systèmes de recharge ou de stockage.

Modélisation du nœud en VHDL

Le nœud, comme il a été décrit précédemment, est un élément qui regroupe de nombreux capteurs. Il a pour tâches d'insérer un en-tête (datation), d'organiser les données selon les besoins du protocole de communication, d'ajouter un code correcteur d'erreur et de transmettre ces données au routeur via une communication sans fil et moyennant un type de modulation prédéfini. Pour conserver un caractère générique, le modèle, qui intègre registres, multiplexeurs, sérialiseurs et blocs de retard, est conçu en VHDL sur la base d'un réseau de Petri, figure 2.14, dont les états peuvent contenir plusieurs jetons, lesquels représentent le nombre d'unités de temps nécessaires avant que l'état suivant puisse être atteint. L'unité de temps associée à ce bloc est la période de l'horloge externe.

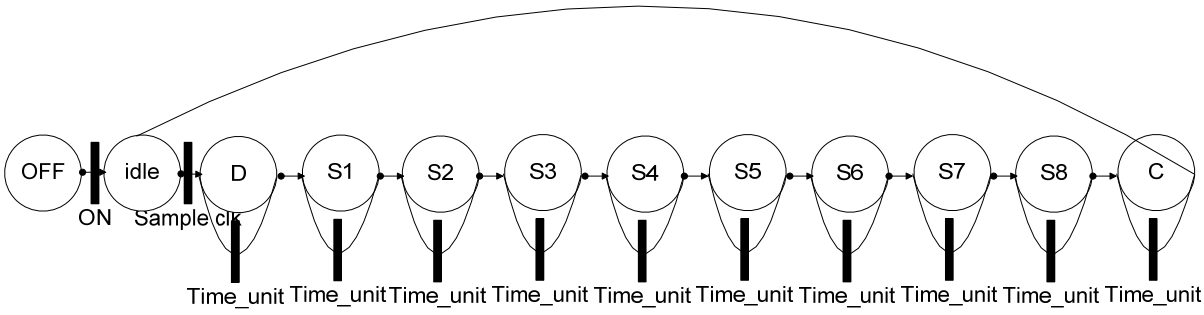


Figure 2. 14 Réseau de Petri du gestionnaire de temps

Les fréquences d'échantillonnage et de sérialisation, la datation, le multiplexage et le calcul de code correcteur sont contrôlés par un processus basé sur l'unité de temps et le protocole de communication. Dans le modèle comportemental, la date est remplacée par un compteur incrémenté à chaque prise d'échantillon et le code correcteur est remplacé par un compteur d'erreurs initialisé à zéro dans le nœud. La Figure 2.15 illustre le schéma bloc du nœud.

En général, dans les réseaux, un en-tête est inséré pour indiquer le début du paquet transmis. Pour éviter, dans le modèle comportemental, ce degré de

modélisation et la détection afférente, une simple impulsion synchrone avec le début de la trame est générée dans le nœud. Cette impulsion est acheminée tout le long de la chaîne de transmission.

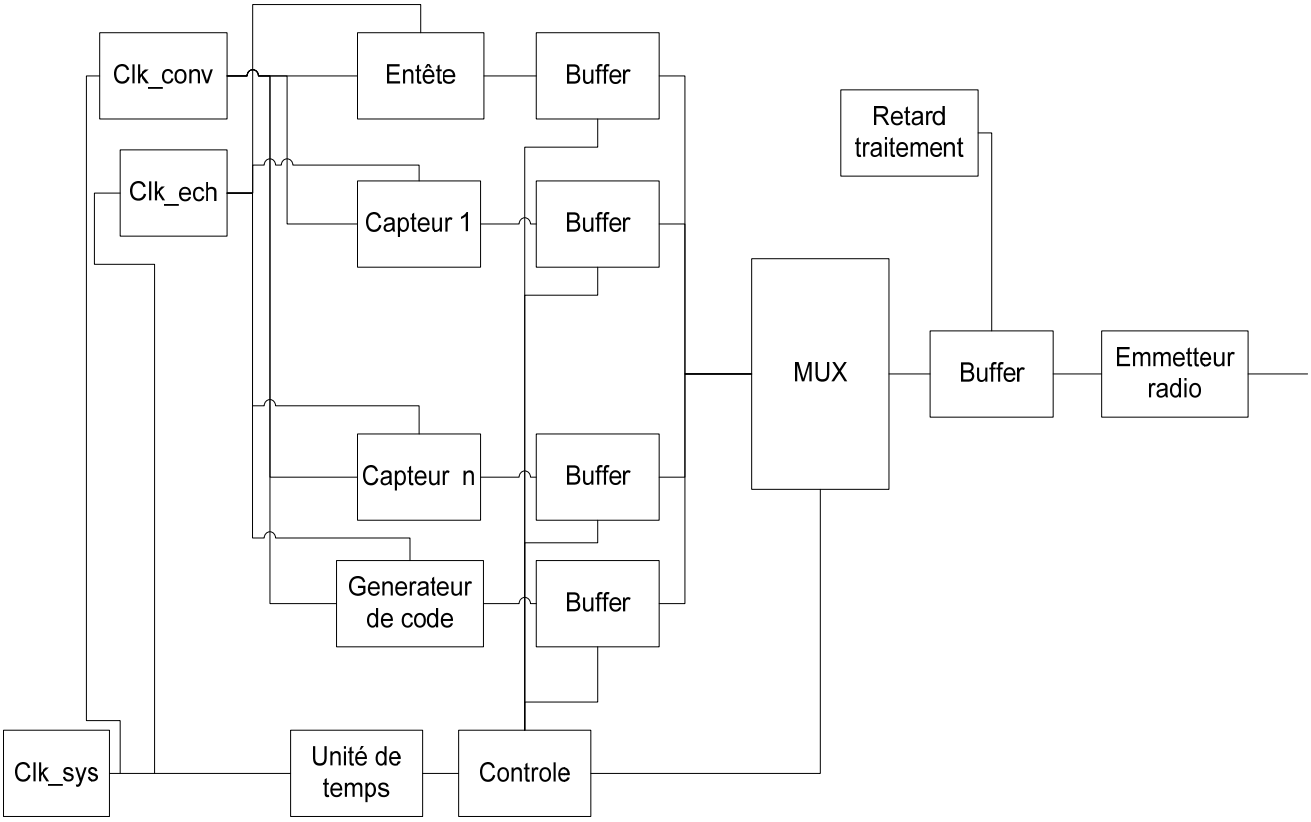


Figure 2. 15 Schéma bloc du modèle du nœud

2.2.3 Le canal de transmission:

Le canal de transmission est la voie par laquelle les données sont transmises d'un émetteur vers un récepteur. Cette partie de la ligne de transmission est la plus difficile à modéliser. C'est la partie du réseau dans laquelle les signaux échangés peuvent être modifiés en fonction de la qualité du média et ses caractéristiques, ce qui provoque des erreurs dans les données et la possibilité d'en perdre une partie... Le comportement de cet élément est fonction de nombreux facteurs: le type de modulation, le débit des données, la température ambiante, la topologie de l'environnement, le nombre d'émetteurs récepteurs utilisant le même canal, le bruit, les distances émetteurs-récepteurs, ...

Les obstacles, les phénomènes d'interférences, de diaphonie, les bruits thermiques et électromagnétiques, les absorptions, les échos, l'affaiblissement et d'autres phénomènes peuvent modifier la puissance du signal radio transmis entre

l'émetteur et le récepteur, ce qui peut se traduire par une source d'erreurs qui modifie les données échangées. Or dans les modèles comportementaux, les connexions entre les éléments ne se font ni par les signaux réels ni par les grandeurs physiques qui les représentent. Il est donc difficile de venir superposer à ces signaux l'ensemble des bruits et perturbations caractéristiques d'un canal de propagation. De plus, développer un modèle mathématique général d'un canal de transmission prenant en compte tous types de perturbations s'avère vite très compliqué voire impossible et induirait de toute façon des durées de simulation prohibitifs. Il est donc nécessaire de définir un modèle comportemental suffisamment robuste pour prendre en compte ces différents aspects.

Pour cela nous avons évalué l'influence des différentes perturbations sur l'intégrité des données échangées et l'avons quantifiée en termes de taux d'erreur de bits à introduire dans la communication. En conséquence un canal de transmission bruité peut être représenté par les blocs fonctionnels suivants figure 2.16:

1. Générateur d'erreurs
2. Localisateur d'erreurs
3. Modificateur de données

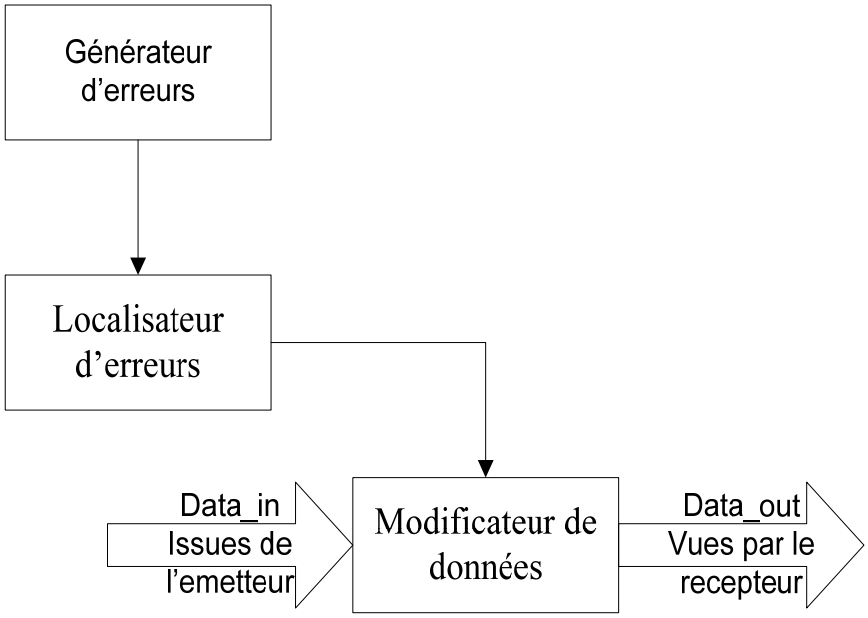


Figure 2. 16 Schéma fonctionnel du canal de transmission

Le générateur injecte de manière aléatoire plus ou moins d'erreurs selon la densité d'erreur de bits calculée à partir des caractéristiques et des paramètres du canal.

Le modèle générique du canal est créé par association des différents blocs fonctionnels précédemment cités. Le paramétrage de ces blocs est effectué en fonction de la topologie et des caractéristiques du réseau. Pour faciliter l'analyse des résultats et le diagnostic sur la pertinence de l'architecture simulée, les erreurs injectées sont encore acheminées séparément des données échangées.

Pour définir une architecture matérielle d'un réseau de capteurs, les paramètres tels que l'organisation topologique, le type de modulation, le type de code correcteur d'erreurs, … sont des facteurs à définir en fonction des spécifications imposées. Pour simuler l'influence de ces éléments, une fonction d'injection d'erreurs basée sur un générateur est introduite dans le modèle du canal de transmission conformément aux propriétés du canal.

Calcul de la densité d'erreur de bits dans le canal de transmission

Le générateur injecte les erreurs dans la trame transmise selon une densité d'erreur définie. La densité de départ est calculée à partir de l'atténuation causée par la distance séparant le récepteur de l'émetteur et des effets du bruit thermique multiplié par le facteur du bruit efficace. Cette densité d'erreur est appelée le taux d'erreurs de bits ou BER « Bit Error Rate ». Cette valeur sera plus ou moins augmentée selon le réglage des autres paramètres. Une fois la valeur globale du BER obtenue, les erreurs seront générées selon une répartition statistique et injectées dans la série de paquets transmis.

i. Calcul du BER de départ

Le bruit

Le bruit est un signal aléatoire s'ajoutant au signal utile pendant la transmission. Suivant sa puissance il peut être une source d'erreurs non négligeable. Il existe plusieurs types de bruit :

- Le bruit blanc : Sa densité spectrale de puissance est constante en fonction de la fréquence. Il peut être d'origine thermique ou d'origine quantique (bruit de

grenaille). Le bruit thermique est généré par l'agitation thermique des électrons dans les conducteurs et les éléments passifs du système de transmission. Le bruit de grenaille quant à lui, est généré par les éléments discrets constituant le système de transmission (transistors, diodes, …).

- Le bruit impulsionnel : il est d'origine électromagnétique et vient des sources électriques proches. Ce type de bruit dépend fortement de l'environnement du support. Sa puissance est faible mais peut atteindre de fortes valeurs pendant de courts instants d'où son nom.
- Le bruit additif : il s'ajoute au signal indépendamment de celui-ci. Il peut donc être facilement caractérisé en absence du signal utile.

La qualité d'une transmission sera définie par le rapport signal sur bruit SNR= Eb/No (2.1). Eb étant la puissance du signal utile et No la puissance du bruit. Ce rapport peut être également donné en dB, dans ce cas $SNR_{dB}=10\log_{10}(Eb/No)$ (2.2).

L'atténuation

L'atténuation est l'affaiblissement que subit le signal émis lors de sa réception. Il est représenté par le rapport de la puissance du signal reçu sur la puissance du signal à l'émission. Cette atténuation est indépendante de la puissance à l'émission. Elle dépend de la distance séparant l'émetteur du récepteur et est caractéristique du canal lui-même et de la fréquence de la transmission radio.

Calcule du BER de départ

Selon le domaine d'application du RCSF, la variation du bruit thermique, liée aux variations de température, peut être minime ou très importante (applications aéronautiques, spatiales, sous-marines…).

Dans le premier cas où les variations de la température et du bruit thermique sont négligeables, le calcul du BER de départ peut se faire en utilisant la courbe d'atténuation du signal mesurée sur le canal utilisé, ce qui peut se traduire par l'équation suivante, dépendante du canal lui-même et indépendante de la température (le BER étant majoré à 1) :

$$\text{BER} = kd - \left(b + \frac{P}{\upsilon}\right) \quad (2.3)[9]$$

Dans laquelle d est la distance entre l'émetteur et le récepteur (m), P est la puissance d'émission (dBm), k (m^{-1}), b (sans unité) et υ (dBm^{-1}) sont des constantes tirées de la courbe ou des caractéristiques du canal utilisé.

Ces paramètres sont définis par l'utilisateur du simulateur. Par défaut ces valeurs sont : k= 0.2385/m, b=2.7, υ=18.0dBm^{-1} (ces valeurs de paramètres ont été utilisées dans une application dans [9]).

Dans le deuxième cas, où les variations de la température et du bruit thermique sont importantes, il faut avoir un moyen de calcul du BER de départ qui tienne compte de la valeur de la température.

Les courbes d'atténuation issues de mesures expérimentales préliminaires faites sur le canal de transmission pour différentes fréquences porteuses seront introduites dans le modèle sous forme de tableaux figure 2.17.

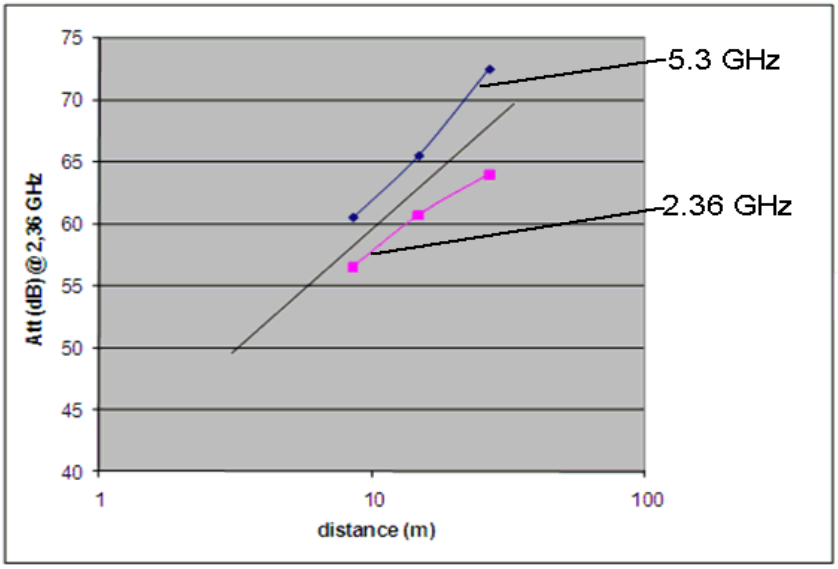


Figure 2. 17 Exemple de courbes d'atténuation

Lors du paramétrage du modèle, il est possible de choisir la courbe d'atténuation dans une bibliothèque prédéfinie ou bien de la définir en remplissant un tableau à partir d'une campagne de mesures adéquate. Une fois l'architecture paramétrée en fixant la distance, la puissance d'entrée (PSE) et la fréquence porteuse utilisée, on peut calculer la puissance reçue au bout du canal à la réception (PS).

$$\text{PSE(dBm)} - \text{att(dB)} = \text{PS (dBm)} \quad (2.4)$$

Pour déterminer le rapport signal/bruit, le bloc considéré utilise en entrées la température T (K), la bande passante Bp (HZ) et la valeur du bruit thermique déterminée par l'équation (2.5) dans laquelle K est la constante de Boltzmann. Ce bruit se situe généralement dans un intervalle allant de quelques microWatts à quelques mW. Cependant, dans les RCSF, les sources d'énergie étant très limitées, la puissance de transmission est souvent inférieure à 1mW ce qui fait que ce bruit ne doit pas être négligé.

$$PB(\text{dBm}) = 10\log[(Bp \times K \times T)/1mW] \quad (2.5)$$

En plus des éléments précédents ce bloc prend en compte le facteur efficace de bruit comme paramètre d'entrée et donne en sortie la valeur du SNR (2.6) figure 2.18.

$$\text{PS(dBm)} - \text{PB(dBm)} - \text{NFeff(dB)} = \text{SNR(dB)} \quad (2.6)$$

```
01 library IEEE;
02 use IEEE.std_logic_1164.all;
03 use IEEE.MATH_REAL.all;
04
05 library FUNDAMENTALS_VDA;
06     use FUNDAMENTALS_VDA.TLU_VDA.all;
07
08 entity calc_snr is
09   generic (
10             Bp : real :=500.0E6;   --valeur par défaut
11             NEeff: real := 8.0;   --valeur par défaut
12             K: real :=1.380E-20 );--1.380e-23 *10^3 pour mw pour dBm
13   port (quantity T : in real:=300.0;
14         quantity Ps : in real:= -50.0;
15         quantity SNR : out real:=10.0);
16 end entity calc_snr;
17
18 architecture ideal of calc_snr is
19   quantity   Pb : real:=50.0;
20   quantity   Pbdbm : real:=50.0;
21 begin
22     Pb== Bp*K*T;
23     Pbdbm== 10.0 * LOG10(Pb+0.01e-40) ;--LOG10
24     SNR==Ps-Pbdbm-NEeff;
25
26 end architecture ideal;
```

Figure 2. 18 Code VHDL-AMS du calculateur du SNR

La valeur du BER est reliée à la valeur du SNR (ou Eb/No) par des lois propres à chaque technique de modulation. Ces lois sont décrites dans le bloc de calcul du BER,

soit à partir d'équations mathématiques, soit à partir de tableaux de valeurs. Ainsi en fonction du type de modulation, on choisit la loi correspondante si celle-ci est prédéfinie sinon on peut l'introduire à partir d'un nouveau tableau de valeurs. La valeur du SNR fournie par le bloc précédent permet de fixer la valeur du BER de départ. Cette valeur servira d'entrée pour le bloc suivant et sera modifiée en fonction des perturbations que l'on souhaite faire apparaître dans le canal de transmission. Le tableau 2.2 montre des exemples de lois, la figure 2.19 montre les courbes correspondantes et la figure 2.20 le code VHDL-AMS associé :

Tableau 2. 2 Exemples des lois BER (Eb/No)

La loi entre BER et SNR		Type de modulation
$BER = erfc\left(\sqrt{s(x)} \cdot \sin\left(\frac{\pi}{8}\right)\right)$	(2.7)	8-PSK
$BER = erfc\left(\sqrt{s(x)} \cdot \sin\left(\frac{\pi}{4}\right)\right)$	(2.8)	4-PSK
$BER = \frac{1}{2} \cdot erfc\left(\sqrt{s(x)}\right)$	(2.9)	BPSK

Où $s(x) = 10^{\frac{Eb}{No} \cdot \frac{1}{10}}$ (2.10), s(x) étant fonction de Eb/No (SNR).

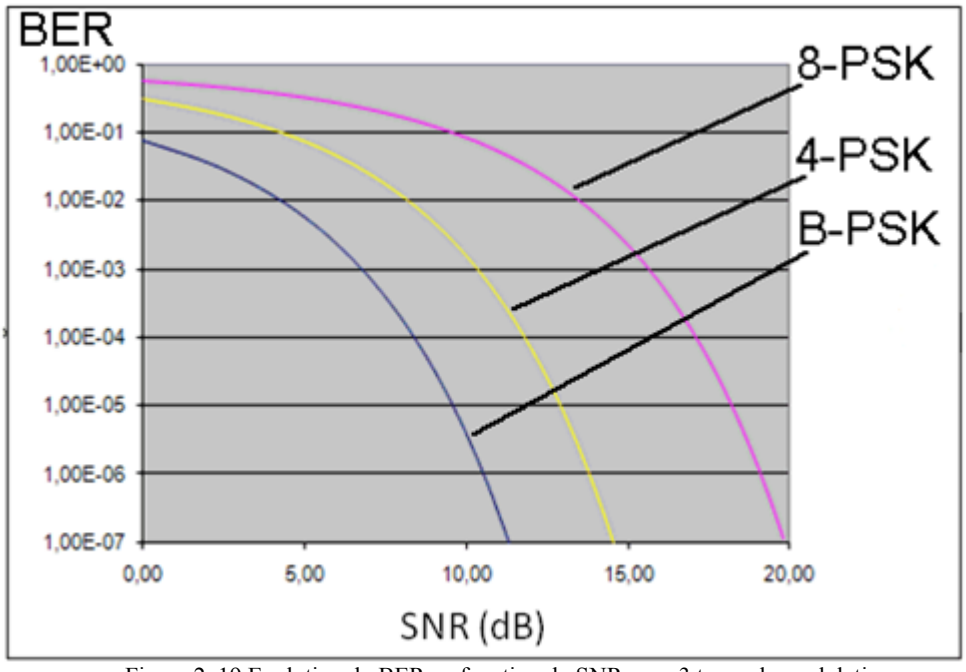


Figure 2. 19 Evolution du BER en fonction du SNR pour 3 types de modulation

```
04
05 library FUNDAMENTALS_VDA;
06     use FUNDAMENTALS_VDA.TLU_VDA.all;
07
08 entity ber_table_2 is
09   generic ( snr_vect  : REAL_VECTOR := (0.00,5.44,7.78,10.00,12.5
10             ber_2psk : REAL_VECTOR := (7.86E-02,4.08E-03,2.66E-04
11             ber_4psk : REAL_VECTOR := (3.17E-01,6.14E-02,1.43E-02
12             ber_8psk : REAL_VECTOR := (5.88E-01,3.11E-01,1.85E-01
13             type_mod : real :=2.0
14             );
15   port (quantity snr : in real:=10.0;
16         quantity ber : out real:=0.001);
17 end entity ber_table_2;
18
19 architecture ideal of ber_table_2 is
20   quantity   ber_int : real:=0.001;
21 begin
22
23     if type_mod= 2.0 use
24         ber_int == LOOKUP_1D(snr, snr_vect, ber_2psk);
25     elsif type_mod= 4.0 use
26         ber_int == LOOKUP_1D(snr, snr_vect, ber_4psk);
27     else
28         ber_int == LOOKUP_1D(snr, snr_vect, ber_8psk);
29     end use;
30     if ber_int>0.0 use
31     ber==ber_int;
32     else
33     ber==0.0000001;
34     end use;
35 end architecture ideal;
```

Figure 2. 20 Modélisation du BER à partir du SNR

La figure 2.21 résume le calcul du BER de départ en tenant compte du bruit thermique :

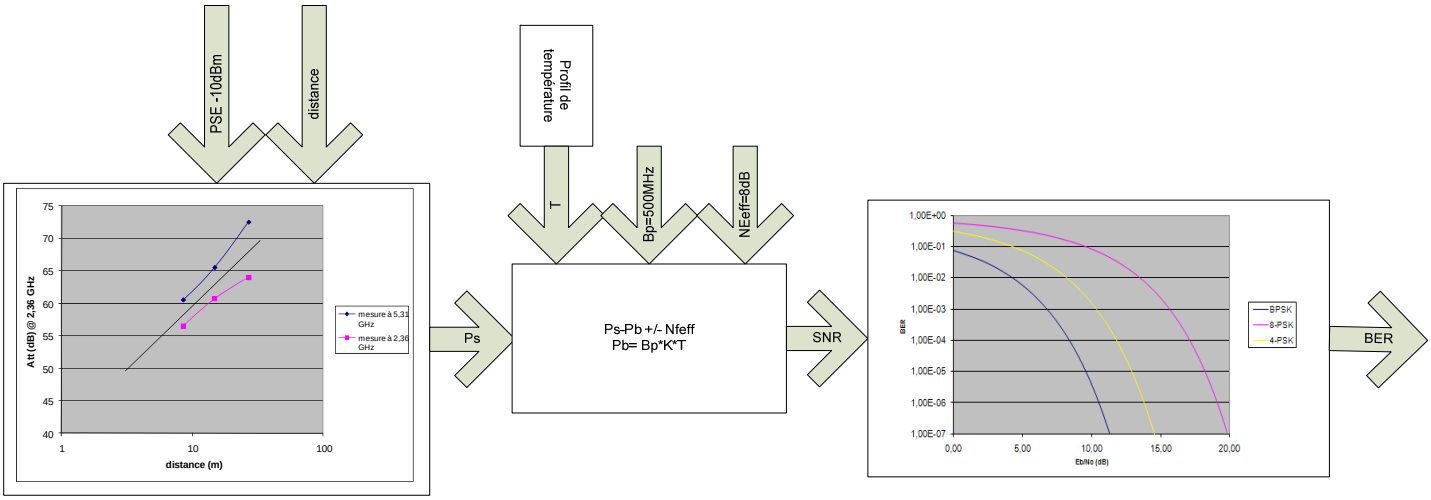


Figure 2. 21 Calcul du BER de départ avec bruit thermique

ii. Calcul du BER final

La valeur du BER obtenue précédemment est multipliée par les facteurs de dispersion, de diaphonie et d'interférence, lesquels sont issus soit de tableaux enregistrés dans le modèle du calculateur à partir des paramètres du canal, soit fixés directement par l'utilisateur. Ces différents facteurs sont caractéristiques du canal et de sa topologie. Le BER résultant est alors introduit dans le bloc de génération d'erreurs.

Générateur d'erreurs

En fonction de la valeur du BER, le bloc générateur d'erreurs injecte plus ou moins d'erreurs dans le flux de données à des endroits définis par le bloc localisateur d'erreurs (sélection des bits de la trame qui doivent être modifiés).

Pour générer des erreurs compatibles avec la valeur du BER, l'injection est répétée de manière cyclique avec une période moyenne $\tau = 1/BER$ (2.11). Cependant si nous injectons une erreur pour chaque période (τ), la distribution obtenue n'est pas représentative de la réalité et notamment ne traduit pas une éventuelle concentration de bits erronés dans un paquet de données. Pour résoudre ce problème (respect du BER et présence d'erreurs dans toutes les répartitions possibles), nous avons créé un bloc qui génère simultanément k erreurs avec une période ($k \times \tau$). Ces erreurs sont générées par k processus pseudo aléatoires différents (figure 2.22 et code VHDL-AMS associé : figure 2.23), c'est à dire qu'ils génèrent aléatoirement k entiers, chacun représentant la position des bits erronés dans une série de trames transmises. La raison pour laquelle nous avons recours à cette méthode peut être illustrée dans la figure 2.24. Dans le cas 1 où nous utilisons par exemple un générateur de 1 erreur par période (τ), le nombre d'erreurs par trame, compte tenu du nombre de bits dans celle-ci, ne dépasse pas 2. Par contre dans le deuxième cas où 5 erreurs sont générées tous les ($5\times\tau$), le BER moyen mesuré sur les ($5\times\tau$) correspond bien au BER réel mais le nombre d'erreurs par trame est plus réaliste (il peut atteindre d'autres valeurs : 3,4) donc il traduit la probabilité d'avoir une concentration de bits erronés dans une fraction de la trame. Cet aspect est important pour le choix du code correcteur d'erreur car celui-ci est spécifié par sa capacité à corriger un nombre de bits par trame et pour évaluer le pourcentage des données perdues : erronées et non corrigées.

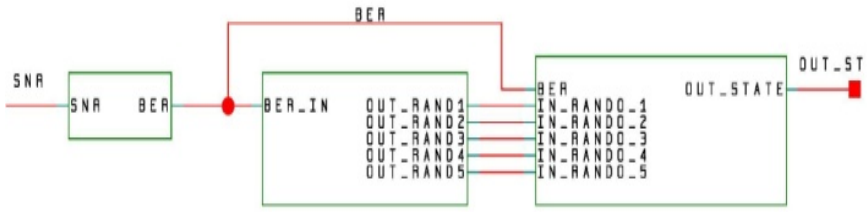


Figure 2. 22 Générateur d'erreurs

```
25 variable seed8:integer:= 120000 ;
26 variable seed9:integer:= 1000 ;
27 variable seed10:integer:= 14000 ;
28 variable unf1,unf2,unf3,unf4,unf5 : real:=1.0;
29 variable temp1, temp2,temp3,temp4,temp5 : real:=1.0;
30 variable ber : real :=0.1;
31 variable tau : time:=20.0ns;
32 variable bit_period : time:=100.0ns;
33 begin
34 If ber_in > 0.0 Then
35 ber:= ber_in ;
36 bit_period := 1.0/bit_rate * sec;
37 tau := 5.0/ber * bit_period;
38 uniform(seed1,seed2, unf1 );
39 uniform(seed3,seed4, unf2 );
40 uniform(seed5,seed6, unf3 );
41 uniform(seed7,seed8, unf4 );
42 uniform(seed9,seed10, unf5 );
43 temp1 := 5.0/ber * unf1;
44 temp2 := 5.0/ber * unf2;
45 temp3 := 5.0/ber * unf3;
46 temp4 := 5.0/ber * unf4;
47 temp5 := 5.0/ber * unf5;
48 End If;
49 If unf1 > 0.0 Then out_rand1<= integer(temp1) ;
50 ELSE null;
51 End If;
52 If unf2 > 0.0 Then out_rand2<= integer(temp2) ;
53 ELSE null;
54 End If;
55 If unf3 > 0.0 Then out_rand3<= integer(temp3) ;
56 ELSE null;
57 End If;
```

Figure 2. 23 Partie du code du générateur d'erreurs

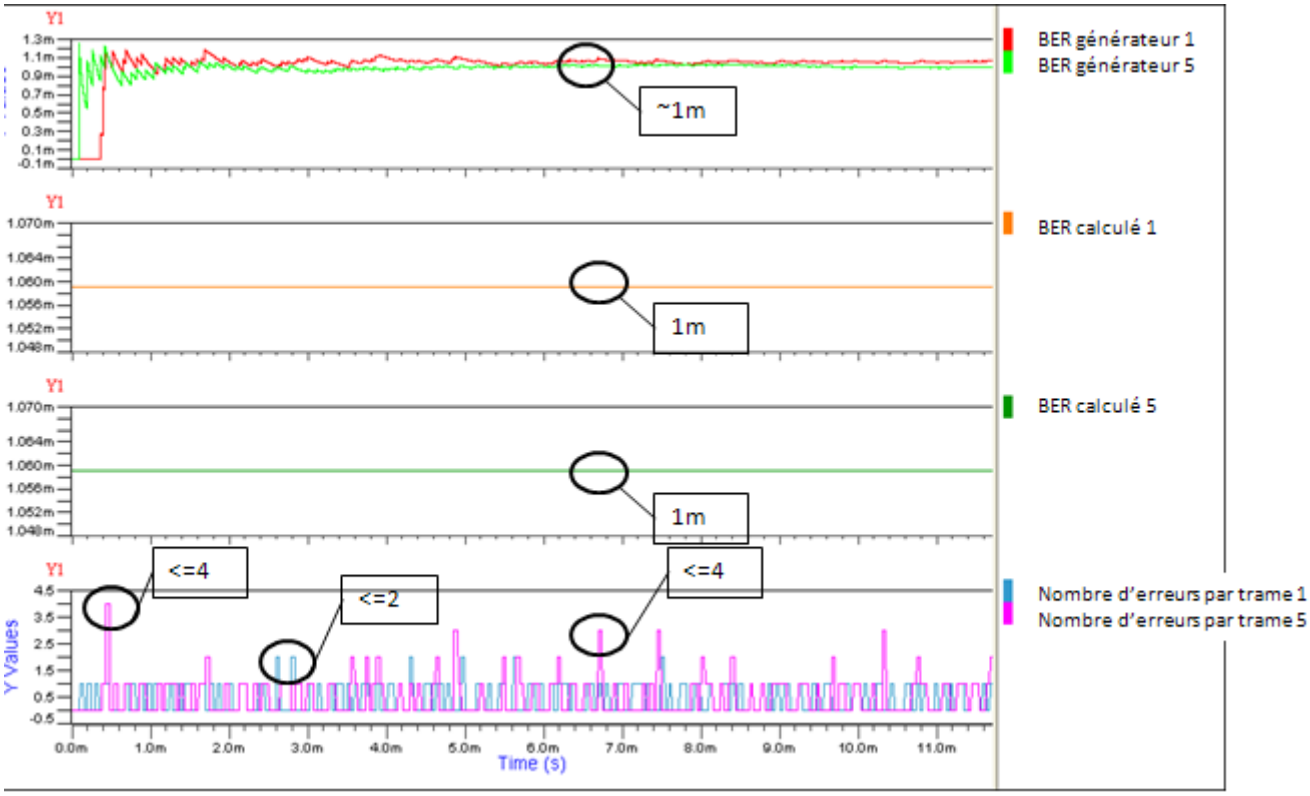


Figure 2. 24 Différence entre les deux générateurs d'erreurs

Pour tester ce générateur, et vérifier que la distribution d'erreurs suit bien une loi uniforme qui est la conséquence d'un bruit dont la puissance suit une loi normale, (répartition pseudo aléatoire), nous avons simulé un réseau dans lequel le BER est différent d'un nœud à l'autre. Nous avons regardé, figure 2.25, la valeur de BER calculée et la répartition des erreurs correspondante pour trois nœuds (plus le BER est grand et plus elles sont denses), puis nous avons tracé la distribution statistique de ces erreurs. Nous remarquons que les trois distributions correspondent à des distributions uniformes, bien que cela est plus visible dans (a) que dans (b) et (c). Ceci est dû au fait que dans (a), le BER étant plus grand, nous avons plus d'échantillons pour une même durée de simulation, ce qui donne une meilleure représentation de la loi statistique. Nous pouvons également noter qu'une plus grande durée de simulation donne une distribution plus uniforme.

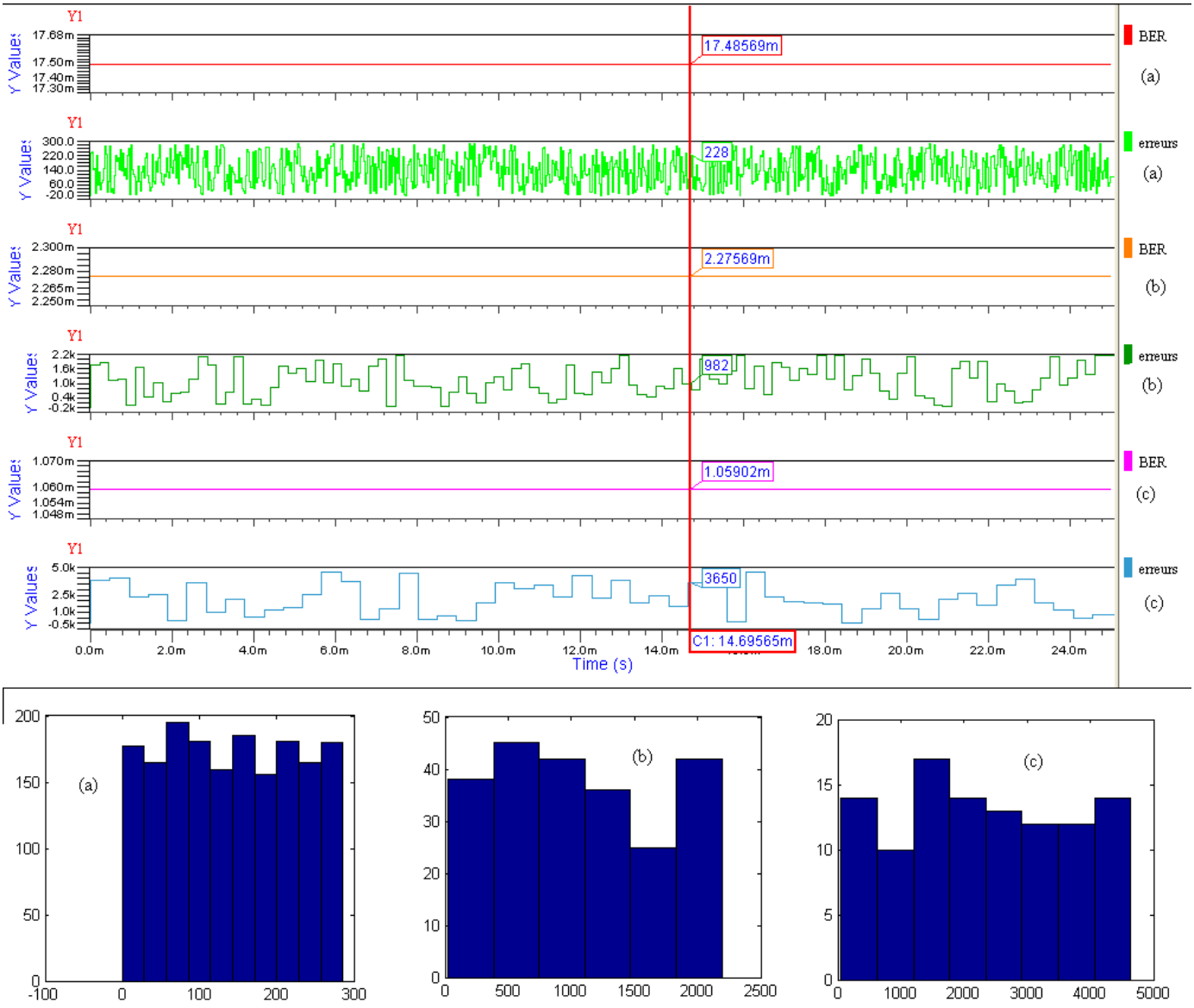


Figure 2. 25 Simulation du générateur d'erreurs et répartition statistique des erreurs

Modélisation du canal en VHDL

La modélisation du canal en langage VHDL synthétisable suit la même démarche que précédemment. Les lois décrivant le comportement du générateur d'erreurs sont discrétisées puis modélisées dans le bloc de calcul du BER par des tableaux de valeurs. Ainsi en précisant en paramètres d'entrée les caractéristiques du canal et de la communication, nous déduisons à partir des lois correspondantes le taux d'erreurs à injecter. Celui-ci étant plus petit que 1 et pouvant avoir des valeurs de 10^{-3}, 10^{-5} voire moins, il est nécessaire, le VHDL synthétisable n'étant pas adapté à l'utilisation des nombres réels, de remplacer ce taux par un nombre de bits erronés par million. Donc la valeur minimale que peut prendre le BER dans notre simulateur est le 10^{-6}. En dessous de cette valeur (BER < 10^{-6}) la plupart des études de BER en fonction de

SNR montre que de telles valeurs ne nécessitent plus d'être prises en compte dans une transmission (une telle communication étant considérée comme étant la meilleure que l'on puisse avoir) .Par exemple la figure 2.26 illustre la variation du BER en fonction du SNR pour différents types de modulation : 10^{-5} est considérée comme limite.

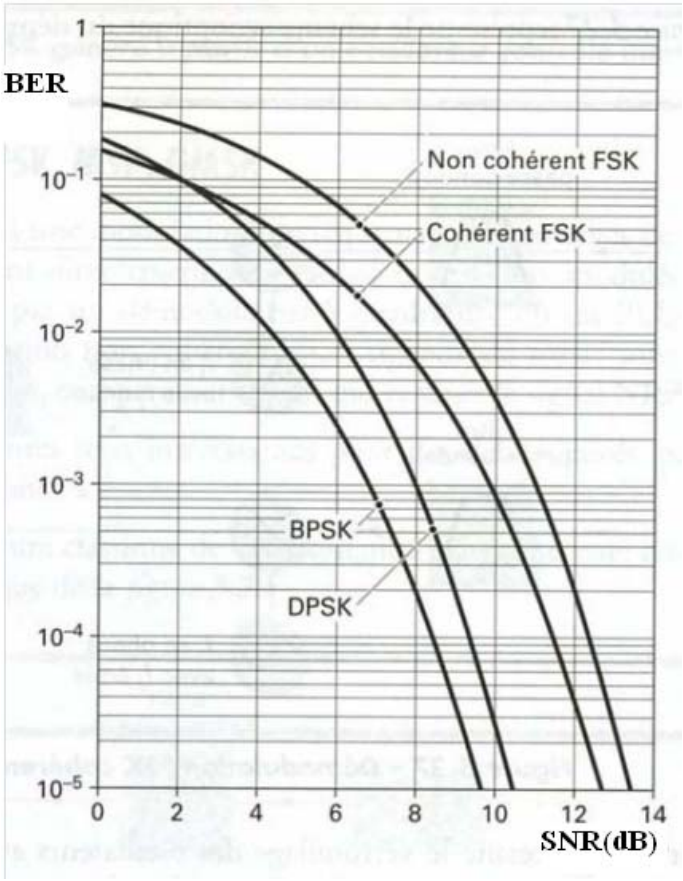


Figure 2. 26 Exemple de lois de BER en fonction du SNR [10]

Les lois décrivant l'influence d'autres phénomènes (par exemple : facteurs de dispersion, diaphonie et interférence, …) sont aussi quantifiées, discrétisées et également stockées dans des tableaux intégrés au modèle du calculateur. Afin d'avoir une répartition équiprobable des bits erronés (chaque bit de la trame doit avoir la même probabilité d'être modifié que son voisin), un générateur d'erreurs détermine de manière aléatoire le rang des bits qui seront modifiés.

A l'origine le générateur pseudo aléatoire de notre modèle était basé sur la mise en œuvre de bascules D rebouclées avec un « ou exclusif » sur le principe de la Figure 2.27. Mais du fait du besoin de plusieurs générateurs pseudo aléatoires différents et indépendants de 20 bits chacun, nous avons eu recours à une solution moins consommatrice de bascules basée sur des mémoires de 2048 mots, lesquels représentent des valeurs aléatoires comprises entre 1 et 1000000. Ces dernières sont

préalablement obtenues hors ligne par un outil approprié (fonction « rand » de Excel par exemple).

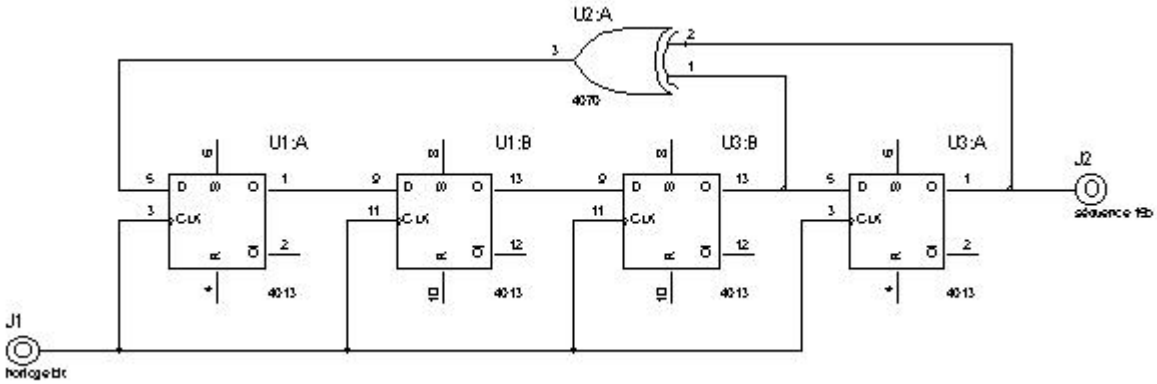


Figure 2. 27 Générateur pseudo aléatoire à base de bascules D

De même que pour le simulateur VHDL-AMS, nous avons testé ces générateurs en les activant un millier de fois et avons étudié leurs distributions statistiques. La Figure 2.28 montre les résultats représentatifs de 3 générateurs. Les Figures 2.28(a) représentent les valeurs générées aléatoirement au cours du temps, ce sont des valeurs entières entre 1 et 1000000 : on voit bien pour les 3 générateurs, des valeurs indépendantes dont la répartition correspond à des distributions pseudo aléatoires. Les figures 2.28(b) et 2.28(c) représentent les distributions statistiques des valeurs générées, lesquelles permettent de vérifier leur uniformité dans leur probabilité d'apparition. Ces valeurs sont classées selon 25 et 10 intervalles pour les figures (b) et (c) respectivement. On peut vérifier que la probabilité d'apparition d'erreurs pour chaque intervalle est assez uniforme, elle le sera d'autant plus que:

- On dispose de davantage d'échantillons
- La taille de la mémoire interne associée aux générateurs est plus importante

Pour avoir une distribution plus uniforme sur des intervalles plus petits il faut prendre un échantillon plus grand et augmenter la taille de la mémoire interne des générateurs.

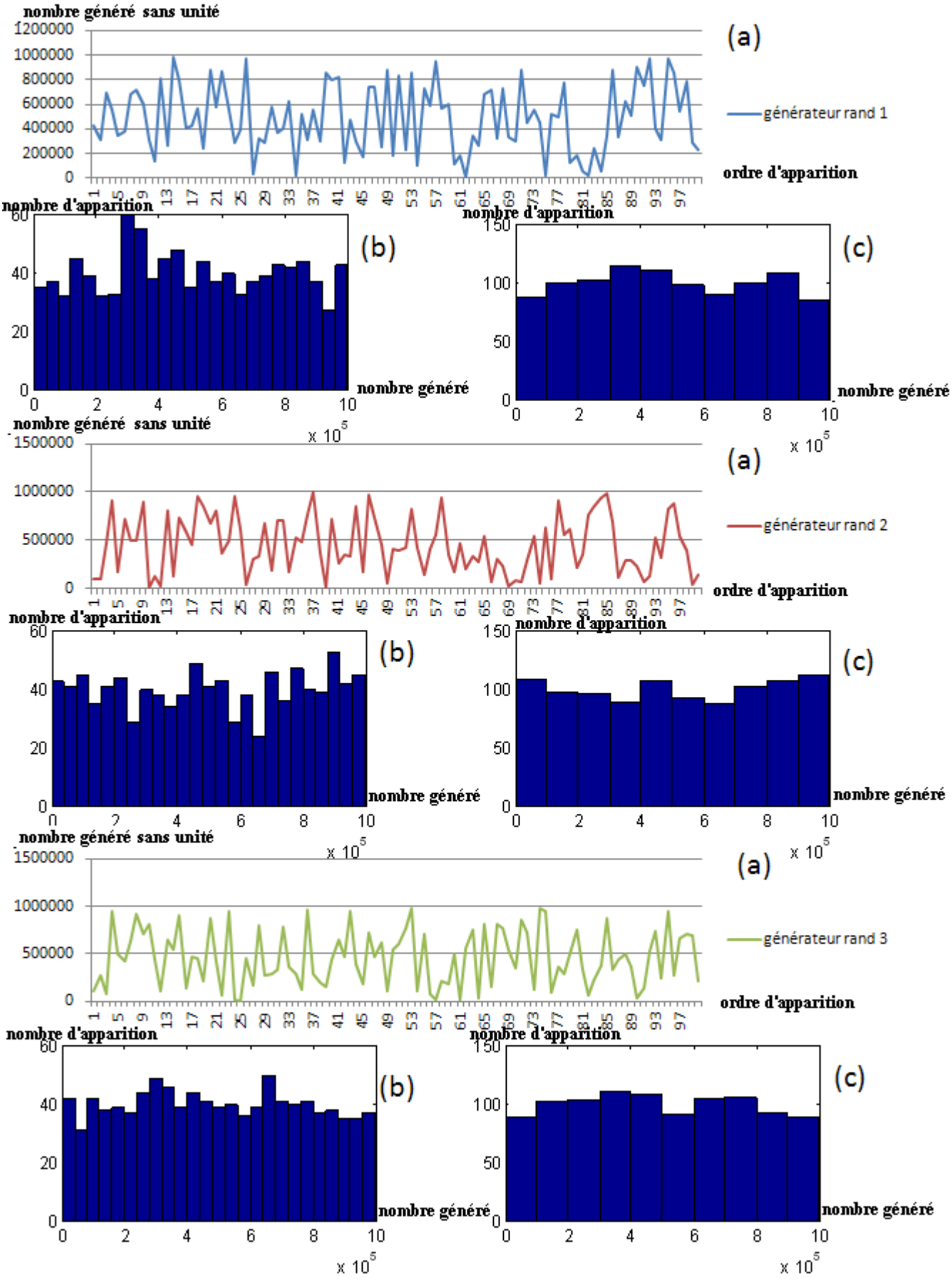


Figure 2. 28 Etude statistique des générateurs pseudos aléatoires

La Figure 2.29 montre le modèle du générateur d'erreurs et ses éléments constituants développé sous Quartus.

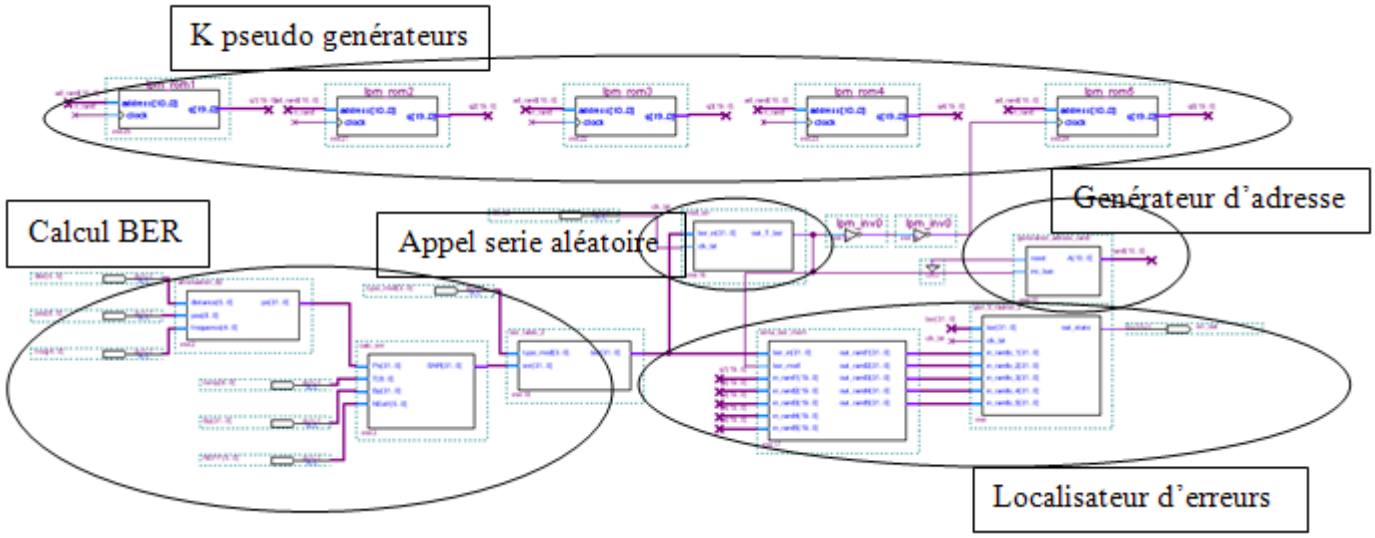


Figure 2. 29 Synoptique du générateur d'erreurs

La génération des erreurs peut être décrite par l'organigramme ci-dessous, Figure 2.30. Parallèlement à la génération de ces k valeurs aléatoires (V_1, …V_i, ...V_k) comprises entre 1 et 1000000 (pour travailler avec des entiers), on calcule le taux d'erreurs à injecter et on détermine conformément au BER, le nombre nb_bits de bits parmi lesquels on doit modifier k bits, puis on les localise dans la trame pour les modifier par inversion.

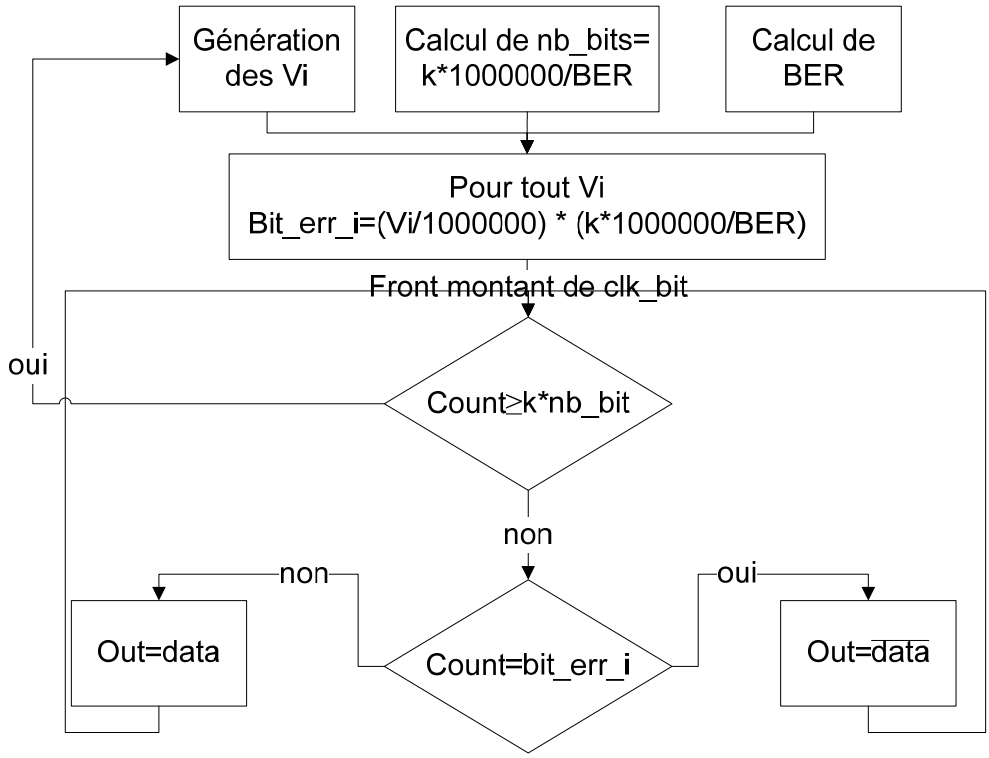


Figure 2. 30 Organigramme du générateur d'erreurs

Le routeur:

La principale tâche du routeur est d'assurer le lien radio entre les différents éléments du réseau par amplification et réémission des signaux reçus. Dans les RCSF à haut débit, la communication est en général ascendante ou descendante ce qui fait qu'il n'y a pas de communication latérale entre les nœuds. Dans ce cas le routeur joue le rôle d'un simple relais (répéteur) dans le chemin de communication entre nœuds et concentrateur, entre nœuds et routeurs ou entre routeurs. Le nombre de routeurs est en général beaucoup plus petit que celui des nœuds ce qui explique que plusieurs nœuds leurs soient associés et que leur localisation soit bien définie. Ils rassemblent les données récupérées pour les renvoyer au destinataire via un trajet de communication adéquat. S'agissant du modèle associé, celui-ci reste comportemental et ne fait donc pas apparaître le protocole de communication. Le routeur est modélisé par un simple répéteur comportant des délais fixes ou paramétrables selon qu'on lui associe ou non des fonctions de correction d'erreurs ou d'autres fonctions supplémentaires.

2.2.4 Le concentrateur

Le concentrateur est le centre de surveillance. Dans le réseau sans fil, il joue le rôle de collecteur et de traitement des données. Il applique des règles de correction incluses dans les codes correcteurs d'erreurs et modifie les données en conséquence. Ce qui présente un intérêt pour la modélisation comportementale et la simulation de cet élément, c'est le temps que prennent ces procédures informatisées avant de disposer des résultats finaux car les applications des RCSF sont en général critiques en termes de délai de disponibilité des grandeurs mesurées par les capteurs. Bien que le concentrateur dispose par ailleurs de fonctions de mesure et d'évaluation des données récupérées, celles-ci sont différentes des évaluations liées à la faisabilité, la validation et la vérification des architectures des RCSF. Dans la modélisation comportementale le concentrateur est donc modélisé en fonction du code correcteur implanté. Chaque trame reçue est analysée pour savoir si elle contient des erreurs ou non, si celles-ci peuvent être corrigées ou non, ce qui permet de fournir en sortie un taux de données perdues, un retard moyen et un retard maximal. En conséquence le type de code correcteur devient un élément du paramétrage du modèle. Cet élément se présente sous la forme d'un tableau de valeurs représentant les délais associés, pour chaque type, aux procédures de vérification et de correction. Ainsi le simulateur n'applique pas les vraies règles de vérification mais prend en considération le temps nécessaire pour les faire. Il corrige également les erreurs injectées si le code correcteur était a priori capable de le faire (c. à d. si le nombre d'erreurs dans la trame est plus petit que les capacités du code correcteur). Enfin le simulateur applique un retard final qui correspond à la procédure d'analyse et d'affichage des grandeurs mesurées par le réseau.

2.3 Les outils d'évaluation

Pour simuler et valider une architecture, certaines informations spécifiques des systèmes embarqués doivent être extraites de la simulation. Pour cela, en plus des modèles des éléments composant le réseau, nous avons créé des "sondes" dont le rôle est la surveillance, durant toute la simulation, des paramètres les plus pertinents. Parmi ces paramètres on peut citer :

- La valeur moyenne du BER pour une partie ou pour toute la chaîne de communication,
- Le PER (taux d'erreur de paquets),
- Le nombre d'erreurs par trame,
- Le comptage des trames perdues ou irrécupérables,
- La latence (temps au bout duquel les données sont disponibles pour l'utilisateur),
- La consommation d'énergie,
- La désynchronisation des horloges,

Ces sondes nous permettent d'évaluer une architecture de RCSF, de vérifier et valider son fonctionnement par rapport au cahier des charges et dans le cas contraire, de localiser les sources de dysfonctionnement.

2.4 Utilisation du simulateur

La mise en œuvre du simulateur, vue dans le prochain chapitre, suit les étapes suivantes (figure 2.31):

- Sélectionner, à partir de la bibliothèque, et placer dans la feuille de travail les éléments qui composeront l'architecture du réseau.
- Interconnecter et paramétrer, en complétant leurs fiches caractéristiques, ces éléments en fonction de l'environnement effectif du réseau.
- Placer les sondes de mesure sur les points critiques en lien avec les contraintes du cahier des charges.
- Fixer le temps de simulation et lancer son exécution.
- Analyser les grandeurs de sortie.
- Accepter ou modifier les éléments utilisés

- Accepter ou modifier l'organisation de l'architecture matérielle du réseau.

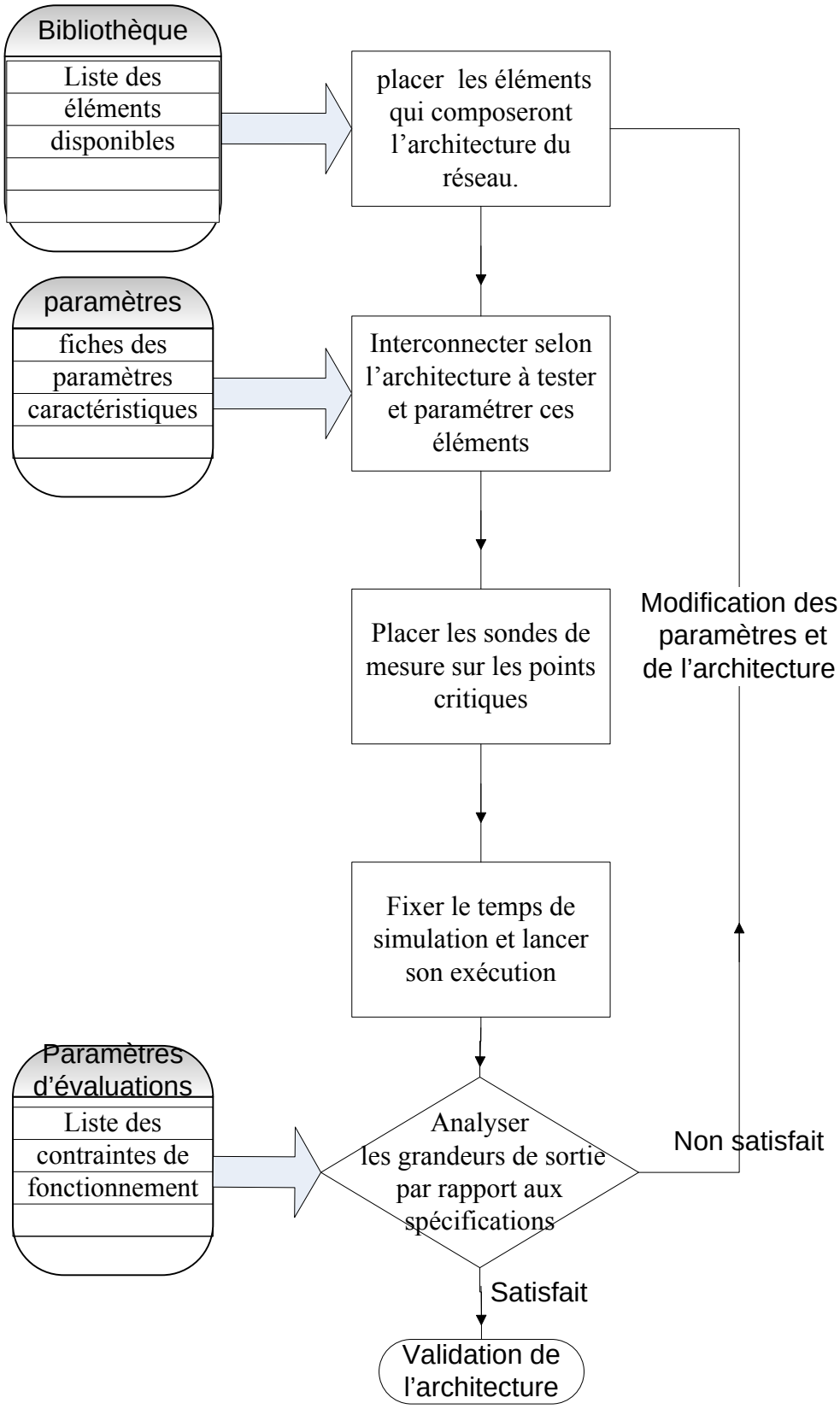


Figure 2. 31 Utilisation du simulateur sous SytèmeVision

2.5 Conclusion

Dans ce chapitre nous avons présenté les principes de base mis en œuvre pour développer tous les modèles comportementaux des éléments constitutifs des RCSFs en détaillant plus particulièrement celui du canal de transmission qui permet de modéliser les effets des perturbations, des atténuations et du bruit thermique, lesquels se traduisent par des erreurs injectées dans les trames de données.

Ces modèles sont conçus pour les deux simulateurs : le premier développé en VHDL-AMS pour une utilisation dans l'environnement SystemVision comme support de simulation, le deuxième développé en VHDL synthétisable pour une implémentation sur un FPGA. Bien que les mêmes principes soient utilisés pour développer les modèles des deux simulateurs, certains aspects ont dû être pris en considération compte tenu des différences dans la puissance de description de chacun des langages. Dans ce chapitre nous avons présenté les principes mis en œuvre pour la modélisation comportementale des éléments du réseau et cité quelques modifications qu'il était possible de faire pour exécuter ces modèles dans un environnement matériel.

Dans le développement des modèles, nous avons proposé une méthode pour prendre en compte, à partir de fiches de paramètres et dans les boucles d'évaluation, les caractéristiques technologiques des éléments du réseau.

S'agissant du simulateur implémenté sur FPGA, nous avons introduit une méthode originale permettant de faciliter l'exploration architecturale sans avoir à recompiler le fichier VHDL et reconfigurer le circuit. Ceci a été obtenu:

- En surdimensionnant les modèles pour contenir tous les paramètres de réglage possibles,
- En traduisant les caractéristiques modifiables en des paramètres externes aux modèles,
- En profitant de la possibilité de lire et modifier le contenu des mémoires internes d'un FPGA durant son fonctionnement.

Dans le troisième chapitre nous allons illustrer la mise en œuvre de ces simulateurs en suivant la démarche décrite dans la fin de ce chapitre. Nous prendrons pour cela un exemple de RCSF et dresserons un tableau comparatif des performances respectives de ces deux simulateurs.

Références

[1] Akyildiz, I.F., Su, W., Sankarasubramaniam, Y. and Cayirci, E., A survey on sensor networks. IEEE Communications Magazine. v40 i8. 102-114.

[2] J.A. Stankovic, T.F. Abdelzaher, L. Chengyang, S. Lui, and J.C. Hou, Real-time communication and coordination in embedded sensor networks. Proceedings of the IEEE. v91. 1002-1022.

[3] Eduardo F. Nakamuraa, b, Carlos M.S. Figueiredoa, Fabiola G. Nakamurac, Antonio A.F. Loureirob, Diffuse: A topology building engine for wireless sensor networks Signal Processing Volume 87, Issue 12, December 2007, Pages 2991-3009.

[4] U. Heinnkel, M. Padeffke, W. Haas, T. Buerner, H. Braisz, T. Gentner, A. Grassmann, The VHDL Refernece, Jhon Wielly & Sons, LTD, England, 2000.

[5] N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS. Defects localization tool for wireless sensors network conception. 10th IEEE ECMS-2011 (Electronic Control, Measurement and Signals), June 1-3, 2011, Liberec, Czech Republic,

[6] N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS. Transmission channel behavioral model for a wireless sensors network simulator. IEEE International Conference on Electronics Circuits and Systems (ICECS 2010), Athènes (Grèce), 12-15 Décembre 2010, pp.924-927

[7] N.NASREDDINE, J.L.BOIZARD, C.ESCRIBA, J.Y.FOURNIOLS. Wireless sensors networks emulator implemented on a FPGA. IEEE International Conference on Field-Programmable Technology (FPT'10), Beijing (Chine), 8-10 Décembre 2010, pp.279-282

[8] E.ALBU. Architecture de communication pour les réseaux d'instrumentation sans fil. Thèse Institut National Polytechnique, Toulouse, 11 Juillet 2011

[9] Mauri Kuorilehto, Marko Hännikäinen, Timo D. Hämäläinen, Rapid design and evaluation framework for wireless sensor networks, Ad Hoc Networks Volume6, Issue 6 (August 2008) pp 909-935.

[10] Céline GUILLEMINOT. Etude et intégration numérique d'un système multicapteurs AMRC de télécommunication base sur un prototype virtuel utilisant le langage de haut niveau VHDL-AMS. Thèse de l'université de Toulouse le Mirail, 1 décembre 2005.

Chapitre 3 : Simulation d'un réseau de capteurs

Chapitre 3. Simulation d'un réseau de capteurs

L'objectif de ce chapitre est de combiner les modèles comportementaux développés afin d'explorer différentes architectures d'un réseau de capteurs dédiés à l'instrumentation.

La question posée est quelle architecture et quel type de modulation garantissent une puissance, un BER et PER fixés ?

La démarche que nous construisons ici va partir d'une conception préliminaire puis une exploration et optimisation de différentes solutions.

Deux environnements de simulation ont été choisis :

1. Un environnement de conception, en l'occurrence SystemVision comme simulateur VHDL-AMS
2. Un simulateur « time dependant »décrit en VHDL synthétisable combiné à une carte FPGA de type (NIOS-DEVKIT-1S40 d'ALTERA qui intègre un FPGA Stratix 1S40).

3.1 Simulation sur SystemVision :

La méthode mise en œuvre pour simuler un réseau de capteur se base sur une première définition de l'architecture matérielle (organisation du réseau : nombre de nœuds, de routeurs, topologie du réseau, répartition géographique…), à partir de laquelle on va effectuer plusieurs simulations afin d'évaluer différents types de solutions et réglages de paramètres. On éprouvera cette démarche sur une application concrète de réseau de capteurs pour une application d'instrumentation en aéronautique: il s'agit d'un dispositif de mesure de la pression sur les ailes d'avion (une miniaturisation du réseau SACER). L'architecture et la topologie du réseau sont décrites sur la figure 3.1.

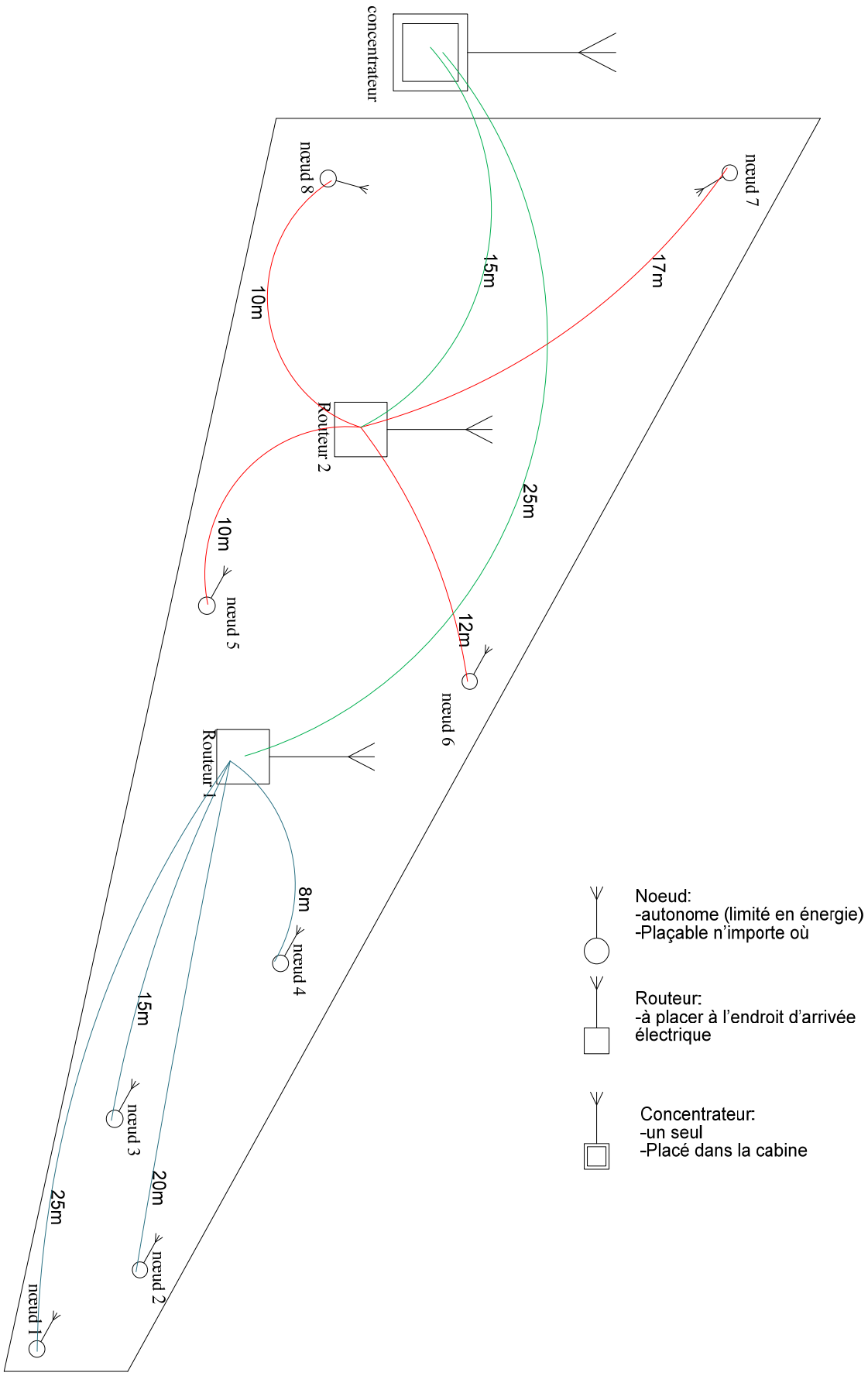


Figure 3. 1 Architecture du réseau simulé

La définition de l'architecture repose sur le choix des éléments à utiliser (type et nombre de capteurs, nœuds, routeurs, concentrateur, niveaux de hiérarchie, forme de trame, répartition géographique…).

L'exemple « cas test » considéré se décompose de 8 nœuds contenant chacun 8 capteurs. Ces nœuds sont répartis sur l'aile de l'avion et communiquent avec un concentrateur situé dans l'avion au travers d'un ou plusieurs routeurs. La conception repose sur les règles de spécifications suivantes :

- Un nœud peut être placé sur tout point de l'aile de l'avion, ce qui conditionne la contrainte de faible consommation électrique tout en garantissant la réussite de la communication.
- La période d'échantillonnage est de 47µs.
- Les routeurs garantissent la communication avec le concentrateur mais sont beaucoup moins nombreux que les nœuds et sont moins limités en énergie car ils peuvent être fixés à proximité de points d'alimentation.
- La trame des données envoyée par les nœuds est constituée de 352 bits qui se décomposent selon la figure 3.2.
- Les trames envoyées disposent d'un code correcteur d'erreurs qui permet au concentrateur de les reconstruire si le nombre d'erreurs qu'elle contient est inférieur à 1.
- La datation des données se fait au niveau des nœuds.

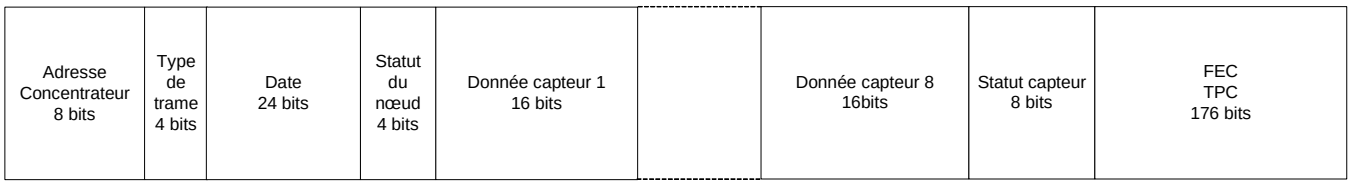


Figure 3. 2 Format de la trame envoyée par les nœuds

Les caractéristiques techniques des éléments du réseau de départ sont décrites dans le tableau 3.1. L'objectif est d'explorer ces conditions et spécificités de fonctionnement pour obtenir une solution d'architecture.

L'arbre des critères à explorer en solution d'architecture est présenté ci-après en déclinant les contraintes d'utilisation :

- Architecture :
  - Distance séparant l'émetteur du récepteur de 1m à 50m maximum.
- Télécommunication :
  - Le type de modulation utilisé
  - Dans cette simulation on a le choix parmi 2PSK, 4PSK et 8PSK
  - Protocole de synchronisation (fréquence d'application)
- Puissance :
  - Puissance émission du nœud 0.1mW, 0.2mW (contraintes sur l'énergie)
  - Puissance émission des routeurs 0.1mW-1mW
- Environnement :
  - Température de l'environnement dans lequel le réseau fonctionne (pour nos test 220K et 300K)

Tableau 3. 1 Première description technique

	Communique avec	Distance	Modulation	Puissance	température	Synchronisation
Nœud 1	Routeur 1	25m	8 PSK	0.1mW	300 K	1Hz
Nœud 2		20m	8 PSK	0.1mW		
Nœud 3		15m	8 PSK	0.1mW		
Nœud 4		8m	8 PSK	0.1mW		
Nœud 5	Routeur 2	10m	8 PSK	0.1mW		
Nœud 6		12m	8 PSK	0.1mW		
Nœud 7		17m	8 PSK	0.1mW		
Nœud 8		8m	8 PSK	0.1mW		
Routeur 1	Concentrateur	25m	8 PSK	0.1mW		
Routeur 2		15m	8 PSK	0.1mW		

3.1.1 Simulation de l'architecture proposée

On explore ici des solutions avec un critère de choix représenté par le nombre d'erreurs par trame envoyé par un nœud (err_noeud_i) à la réception. La transmission

est considérée réussie si ce nombre est inférieur à 1 (le code correcteur choisi permet, au niveau du concentrateur, la récupération de l'information à condition d'avoir au maximum une seule erreur par trame). Ces erreurs sont causées par l'atténuation du signal émis, la dispersion, l'influence du bruit thermique et les interférences entre différentes transmissions… Ces erreurs sont injectées dans la trame au niveau de chaque canal de transmission par un injecteur d'erreurs. La densité de ces erreurs est calculée dans le générateur d'erreurs à partir des facteurs influents. Ce calcul, détaillé dans le paragraphe 2.4 du deuxième chapitre, se résume aux étapes suivantes :

- Calcul de l'atténuation
- Calcul de la puissance des bruits
- Calcul de la valeur du rapport SNR
- Calcul de la valeur de BER de départ en s'appuyant sur les équations de chaque type de modulation
- Calcul du BER final en multipliant sa valeur par les coefficients des autres facteurs : dispersion, diaphonie, interférence…

L'injecteur d'erreurs utilise cette valeur pour générer les erreurs dans les trames échangées en utilisant des générateurs à base de fonctions aléatoires.

Après la construction de l'architecture, on place les sondes de mesure du nombre d'erreurs par trame à la réception et on affiche les résultats. Les résultats d'une première simulation figure 3.3 sont présentés ci-après :

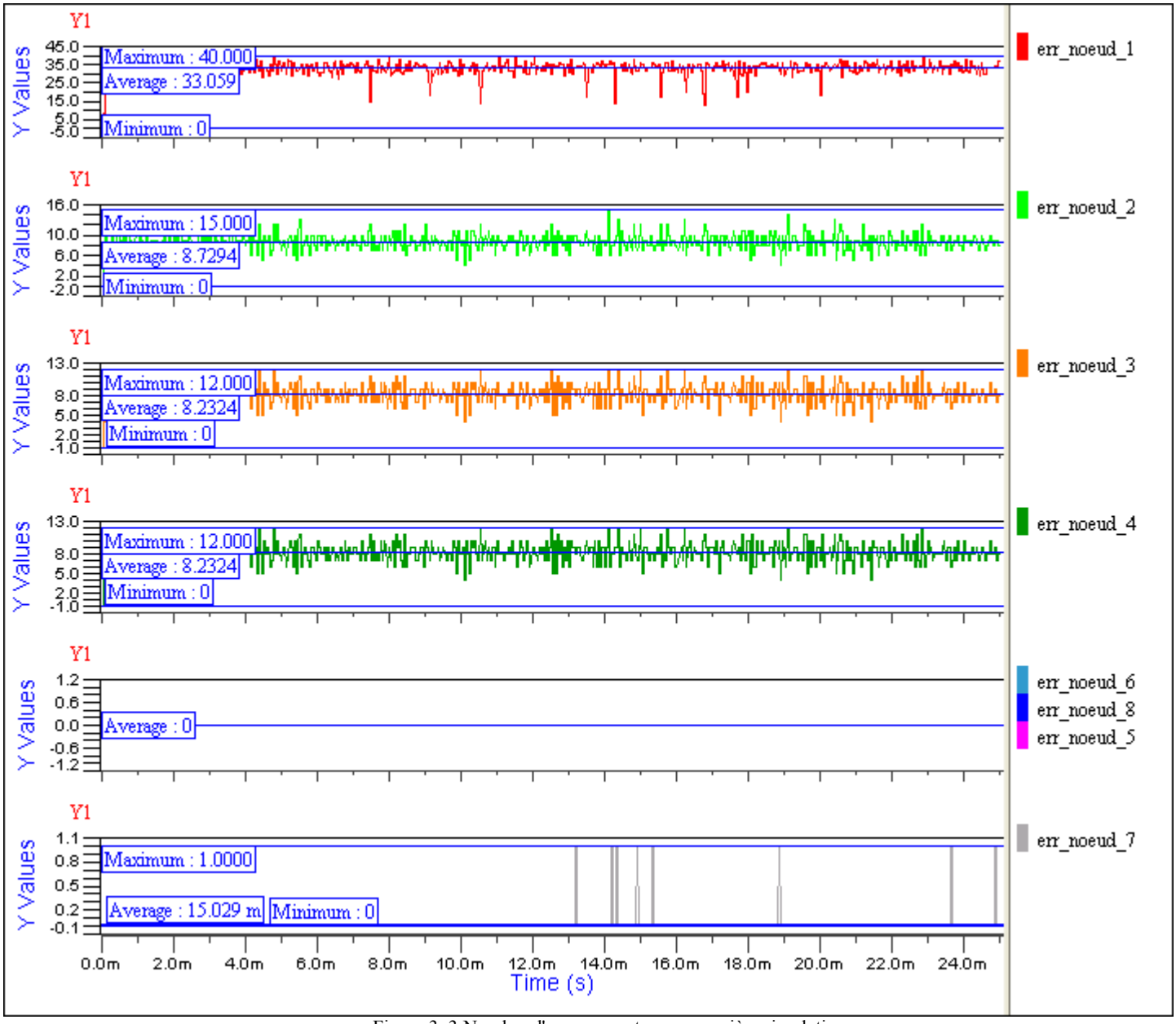


Figure 3. 3 Nombre d'erreurs par trame, première simulation

L'analyse des courbes précédentes démontre que pour les 4 premiers nœuds on a des nombres d'erreurs important (40, 15, 12, 12). On rappelle que la valeur maximale autorisée est égale à 1, ce qui veut dire que les données envoyées par ces nœuds sont perdues !!

Pour remédier à ce problème, on va localiser le point faible de ce réseau, source de ces erreurs, en observant les BER sur chaque étage de la transmission. Pour cela on place des sondes de mesure de BER sur toutes les liaisons sans fil. Ces sondes comptent le nombre d'erreurs et calculent le taux de leur présence par rapport au nombre de bits échangés.

En l'absence d'erreurs cette valeur est à zéro mais elle peut l'être également du fait que pour des faibles valeurs théoriques de BER, les erreurs peuvent tarder à apparaitre, d'où l'importance d'avoir des temps de simulation suffisamment longs !!

Les valeurs mesurées par les sondes pour cette simulation apparaissent dans les figures 3.4 et 3.5

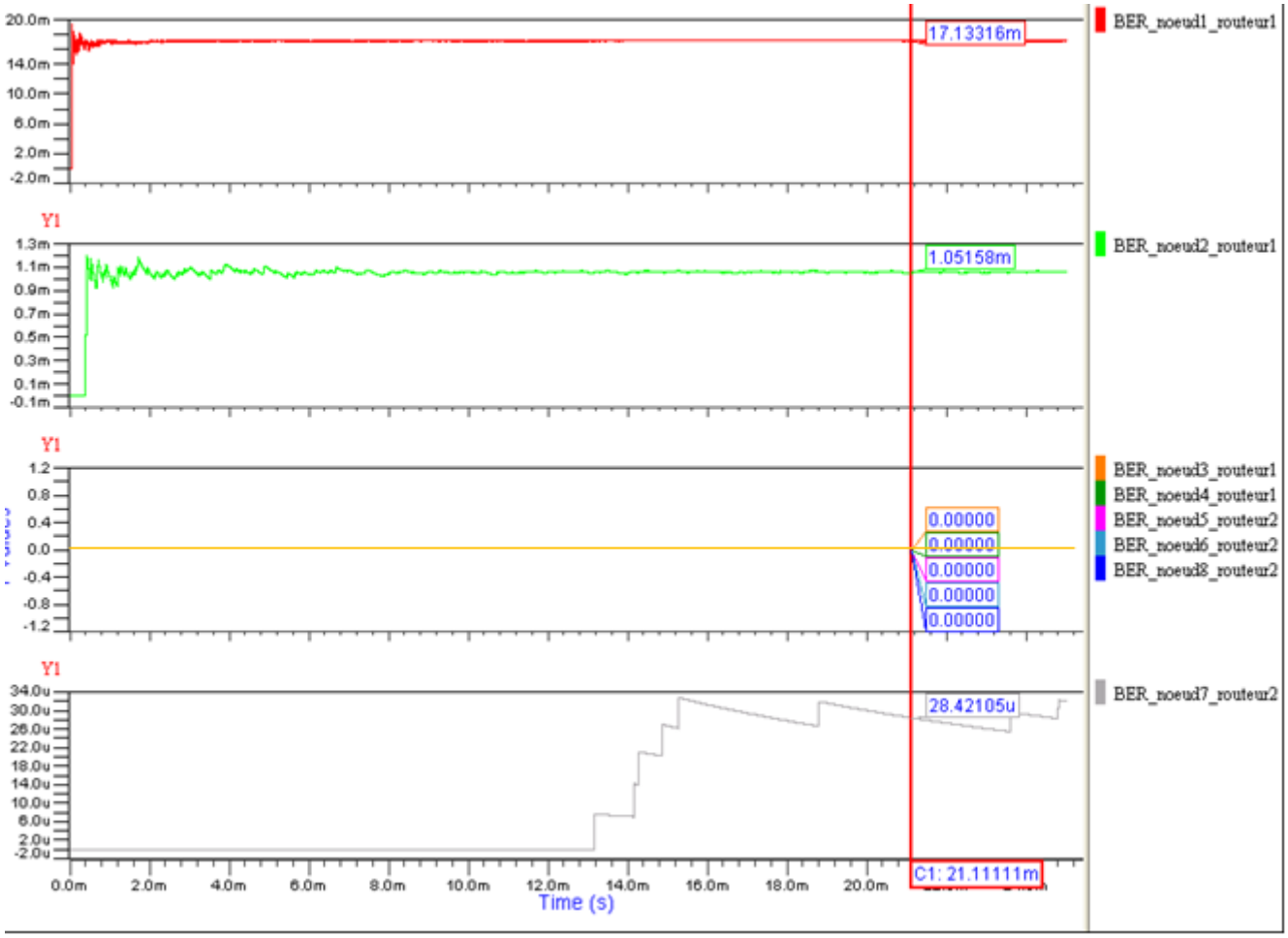


Figure 3. 4 BER noeud_routeur, première simulation

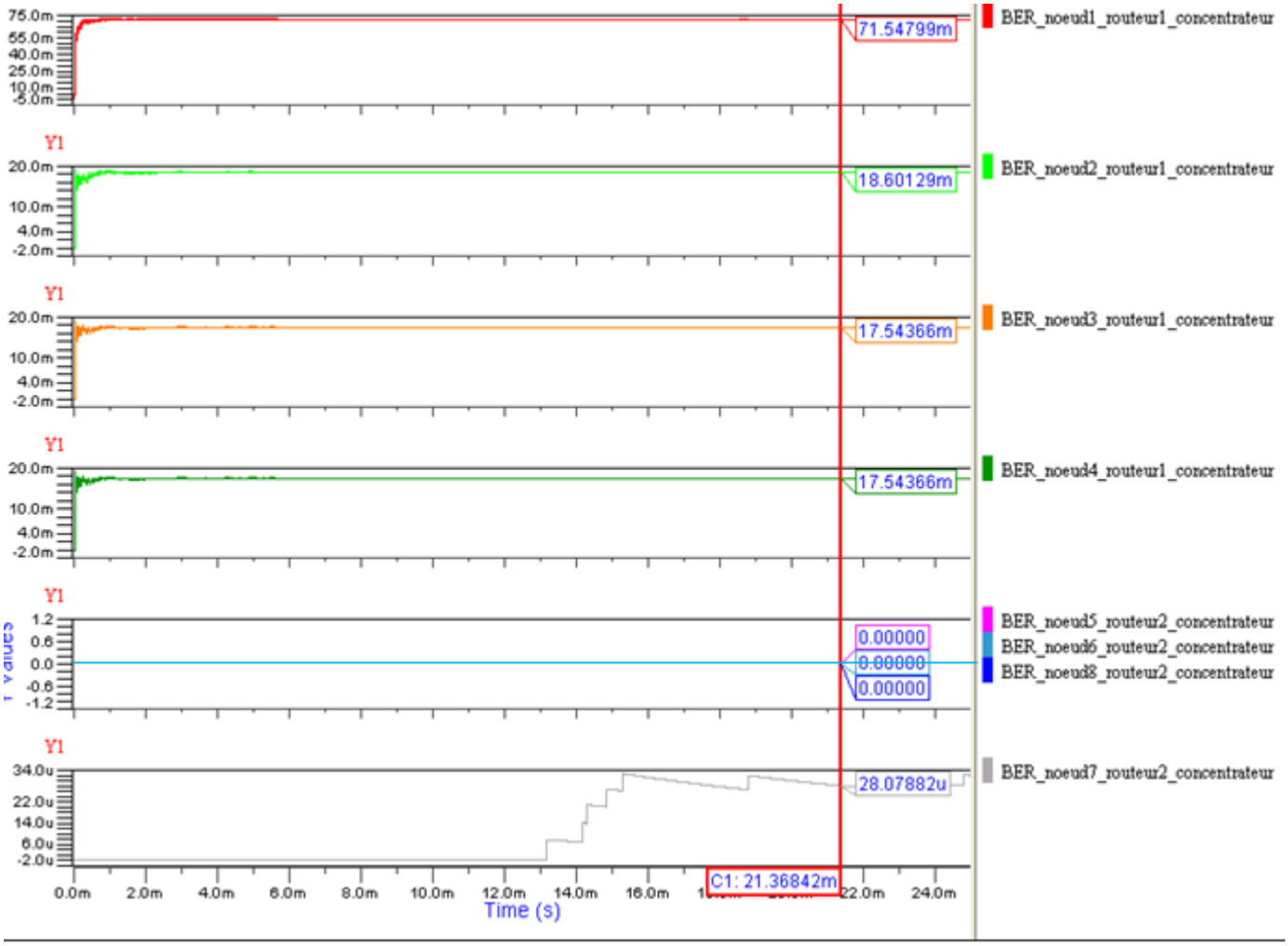


Figure 3. 5 BER noeud_routeur_concentrateur, première simulation

Les résultats obtenus par la première simulation sont classés dans le tableau suivant :

Tableau 3. 2 Résultats de la première simulation (valeurs de BER)

Connexion	Valeur BER
Nœud_1-Routeur_1	17.1 m
Nœud_2-Routeur_1	1.05 m
Nœud_3-Routeur_1	0
Nœud_4-Routeur_1	0
Nœud_5-Routeur_2	0
Nœud_6-Routeur_2	0
Nœud_7-Routeur_2	28.04 µ
Nœud_8-Routeur_2	0

Connexion	Valeur BER
Nœud_1-Routeur_1-Concentrateur	71.5 m
Nœud_2-Routeur_1-Concentrateur	18.6 m
Nœud_3-Routeur_1-Concentrateur	17.5 m
Nœud_4-Routeur_1-Concentrateur	17.5 m
Nœud_5-Routeur_2-Concentrateur	0
Nœud_6-Routeur_2-Concentrateur	0
Nœud_7-Routeur_2-Concentrateur	28.04 µ
Nœud_8-Routeur_2-Concentrateur	0

La simulation conduite mais en exergue que le BER est très élevé dans les connexions suivantes : noeud1-routeur1, routeur1-concentrateur. Ces deux connexions ne sont donc pas bien établies et il est nécessaire de modifier l'architecture du réseau pour les améliorer. Ce résultat est cependant en conformité avec l'éloignement qui affecte la puissance du signal utile reçu.

Un des premiers tests qui peut être fait, est de vérifier si l'influence du bruit thermique est prépondérante : on va donc simuler le réseau dans un contexte où la température est plus basse et par conséquent le bruit thermique plus faible.

Remarque : Il est possible d'avoir des valeurs de BER plus précises si l'on augmente le temps de simulation. Cependant le simple cas de figure précédent a nécessité 1h de simulation et a généré 5 GOctets de données. L'effet du temps de simulation pour une durée de fonctionnement donnée et la quantité de données générées sera illustré ultérieurement.

3.1.2 Etude de l'influence de la température

Le réseau d'instrumentation peut être utilisé durant la phase de vol de l'avion où la température peut atteindre 220K. Il est intéressant de vérifier d'une part son comportement pour évaluer sa sensibilité ou sa robustesse à cette température et d'autre part vérifier la capacité du simulateur à prendre en compte son influence. Pour cette simulation les caractéristiques du réseau resteront donc inchangées à l'exception de la température qui sera portée à 220K.

L'étude des résultats de simulation commençant par la vérification de validité des données à la réception, on affiche le nombre d'erreurs par trame envoyée figure 3.6.

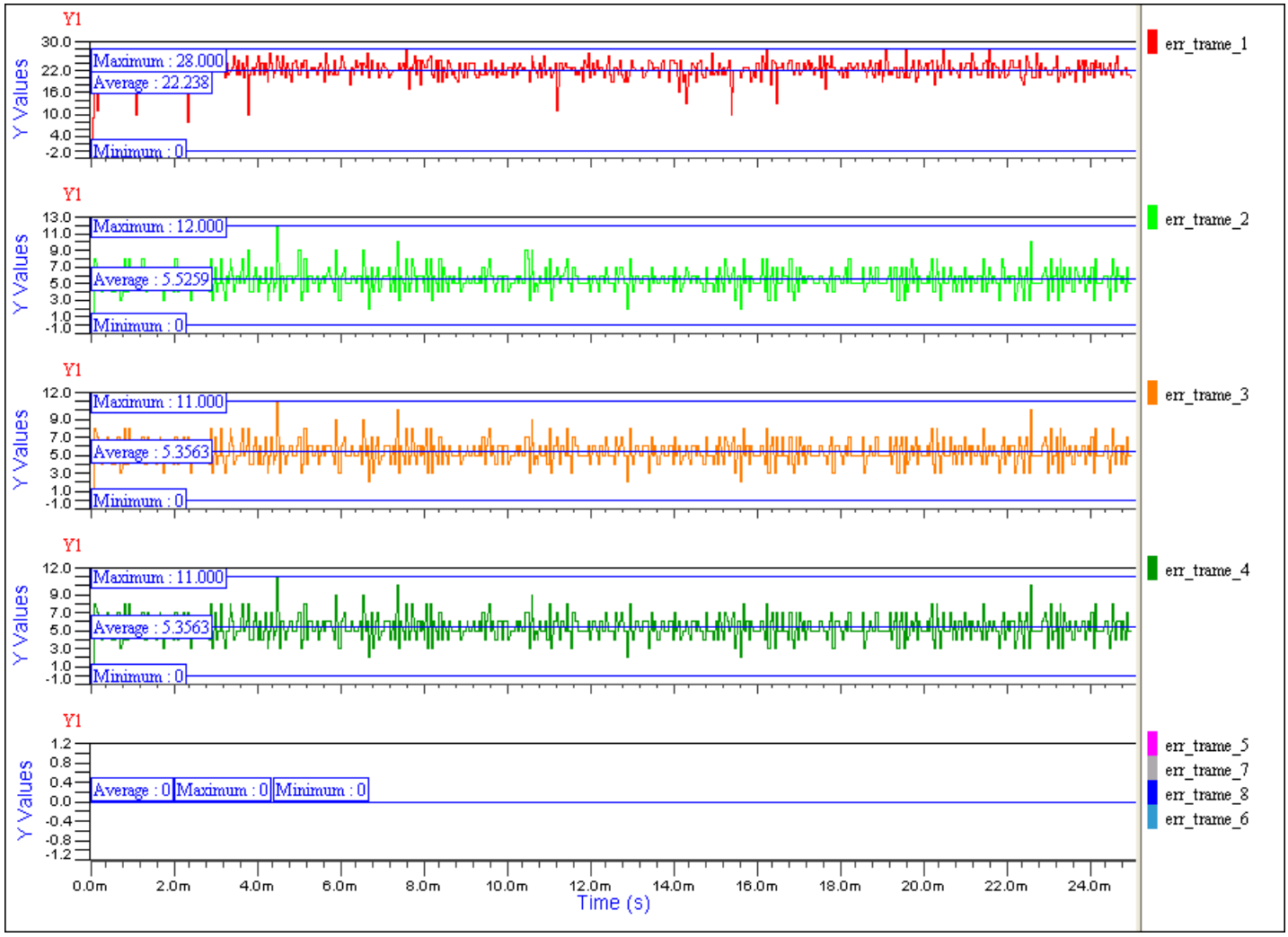


Figure 3. 6 Nombre d'erreurs par trame, deuxième simulation à 220K

Comme pour la simulation précédente à 300K, on remarque que pour les 4 premiers nœuds on a des nombres d'erreurs très grands (très supérieurs à 1 qui est la limite pour laquelle le code correcteur d'erreurs est efficace). En conséquence les données envoyées par ces nœuds seront également perdues, même s'il y a une amélioration notable, tableau 3.3.

Tableau 3. 3 Comparaison des nombres d'erreurs par trame à 300K et 220K

La trame	Valeur moyenne du nombre d'erreurs par trame à 300K	Valeur moyenne du nombre d'erreurs par trame à 220K
Err_trame_1	33	22.2
Err_trame_2	8.7	5.5
Err_trame_3	8.2	5.3
Err_trame_4	8.2	5.3

Focalisons-nous sur les valeurs de BER sur les différentes transmissions, figure 3.7 et 3.8, pour vérifier l'influence de la température.

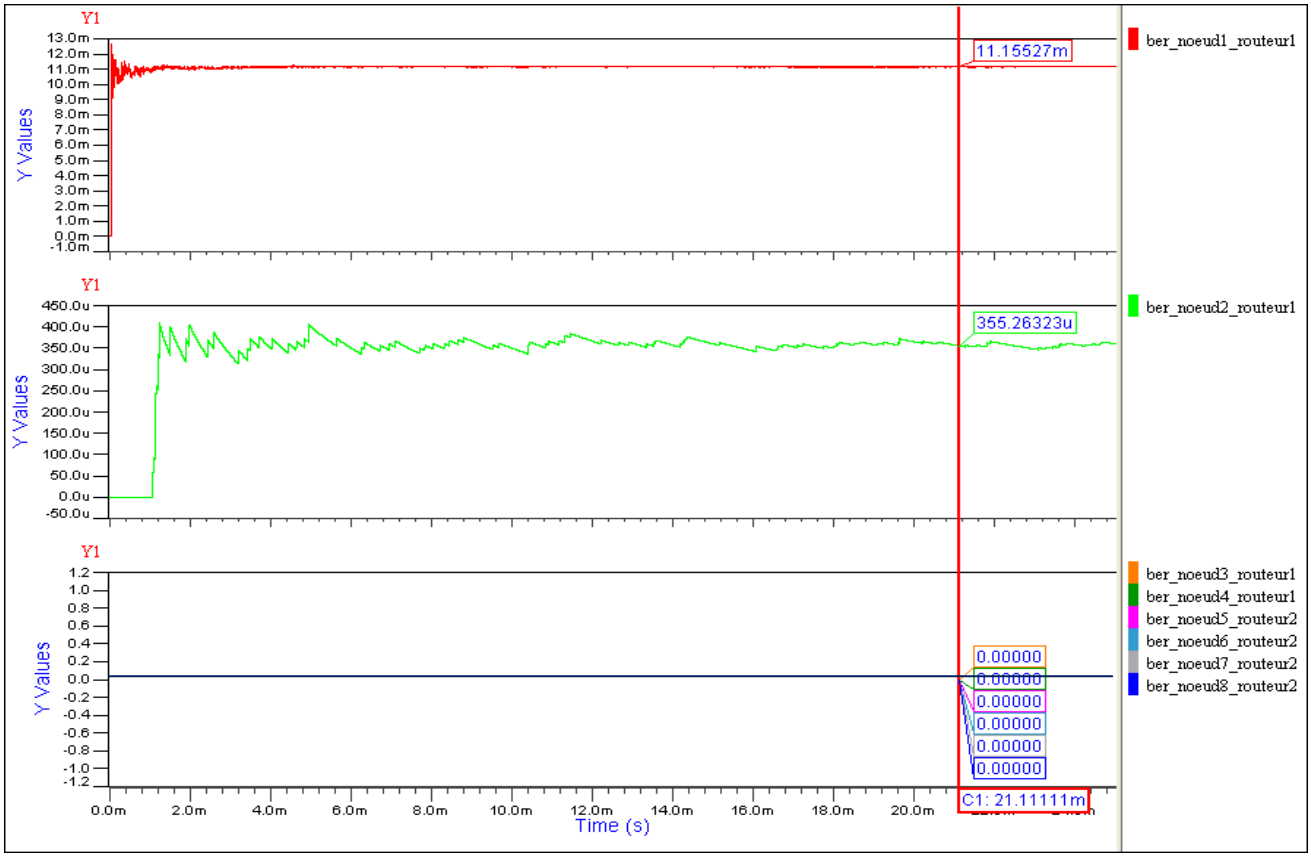


Figure 3. 7 BER noeud_routeur, deuxième simulation

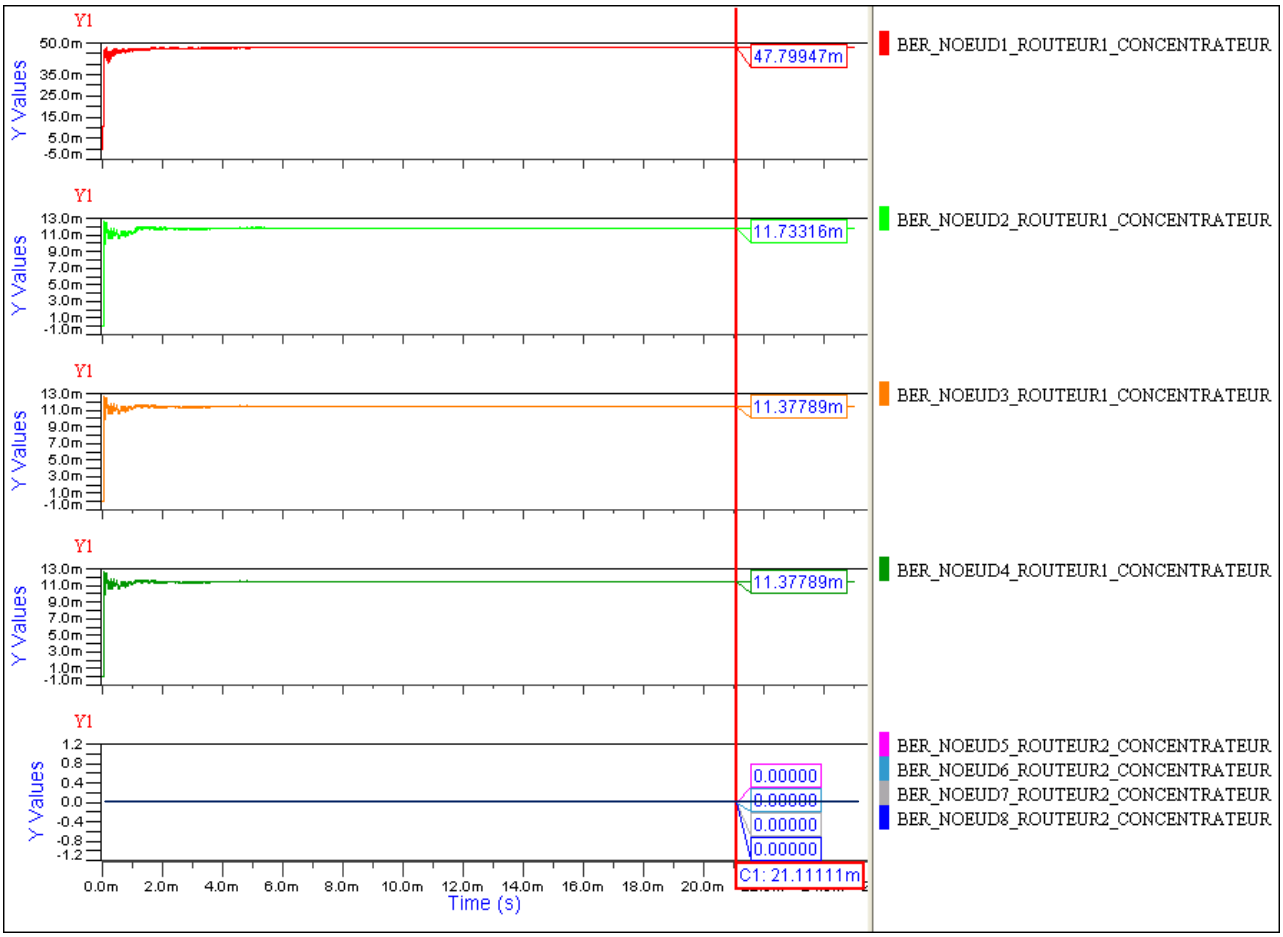


Figure 3. 8 BER noeud_routeur_concentrateur, deuxième simulation

On remarque que, par comparaison avec les anciens résultats, les valeurs de BER sont améliorées, tableau 3.4, mais leur influence sur le nombre d'erreurs par trame n'est pas suffisante.

Tableau 3. 4 Comparaison des valeurs de BER à 300K et 220K

Connexion	BER 300K	BER 220K
Nœud_1-Routeur_1	17.1 m	11.1 m
Nœud_2-Routeur_1	1.05 m	355 µ
Nœud_3-Routeur_1	0	0
Nœud_4-Routeur_1	0	0
Nœud_5-Routeur_2	0	0
Nœud_6-Routeur_2	0	0
Nœud_7-Routeur_2	28.04 µ	0
Nœud_8-Routeur_2	0	0

Connexion	BER 300K	BER 200K
Nœud_1-Routeur_1-Concentrateur	71.5 m	47.7 m
Nœud_2-Routeur_1-Concentrateur	18.6 m	11.7 m
Nœud_3-Routeur_1-Concentrateur	17.5 m	11.3 m
Nœud_4-Routeur_1-Concentrateur	17.5 m	11.3 m
Nœud_5-Routeur_2-Concentrateur	0	0
Nœud_6-Routeur_2-Concentrateur	0	0
Nœud_7-Routeur_2-Concentrateur	28.04 µ	0
Nœud_8-Routeur_2-Concentrateur	0	0

Cette simulation a permis de constater que même avec une température de fonctionnement plus basse, le réseau n'était toujours pas opérationnel.

Dans le chapitre qui suit il est proposé de revenir à une température de fonctionnement du réseau de 300K et de jouer sur la puissance d'émission.

3.1.3 Influence de la puissance d'émission :

Essayons, comme première proposition de solution, de changer la valeur de la puissance d'émission. Reprenons la première analyse à 300K et appliquons un principe simple : une puissance d'émission plus grande améliore le rapport signal sur bruit, donc améliore le BER et induit par conséquent moins d'erreurs par trame. En contrepartie cette solution entraîne une consommation plus grande et donc un raccourcissement de la durée opérationnelle du réseau. Doublons par exemple la puissance d'émission des nœuds de 0.1mW à 0.2mW (limitée par le cahier de charge) et triplons la puissance d'émission des routeurs en passant de 0.1mW à 0.3mW. Ainsi, les caractéristiques du réseau seront comme indiquées dans le tableau 3.5

Tableau 3. 5 Troisième description technique

	Communique avec	distance	Modulation	puissance	température	Synchronisation
Nœud 1	Routeur 1	25m	8 PSK	0.2mW	300K	1Hz
Nœud 2		20m	8 PSK	0.2mW		
Nœud 3		15m	8 PSK	0.2mW		
Nœud 4		8m	8 PSK	0.2mW		
Nœud 5	Routeur 2	10m	8 PSK	0.2mW		
Nœud 6		12m	8 PSK	0.2mW		
Nœud 7		17m	8 PSK	0.2mW		

Nœud 8		8m	8 PSK	0.2mW		
Routeur 1	Concentrateur	25m	8 PSK	0.3mW		
Routeur 2		15m	8 PSK	0.3mW		

Comme les deux simulations antérieures, la première évaluation porte sur le nombre d'erreurs par trame à la réception, figure 3.9.

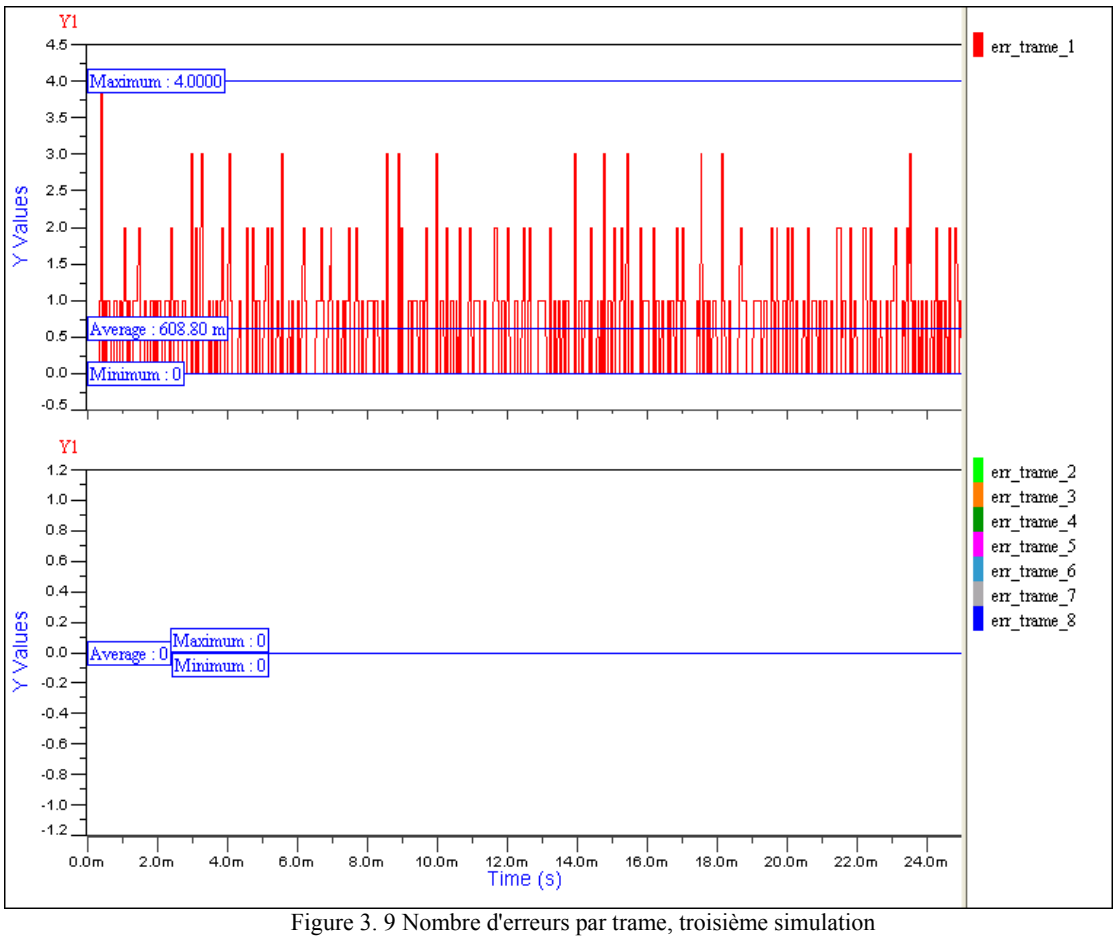


Figure 3. 9 Nombre d'erreurs par trame, troisième simulation

Le fait de doubler la puissance d'émission des nœuds et de tripler celles des routeurs permet de garantir la communication et la collecte des données de tous les nœuds sauf pour le nœud1 lequel est le plus éloigné du routeur (le nombre d'erreurs par trame peut atteindre 4) ce qui démontre que la communication routeur1_ concentrateur est bien établie. Il faut cependant chercher une autre solution pour la communication nœud1_routeur1. On peut vérifier cela en observant la variation des valeurs de BER de ces communications figure 3.10 et 3.11.

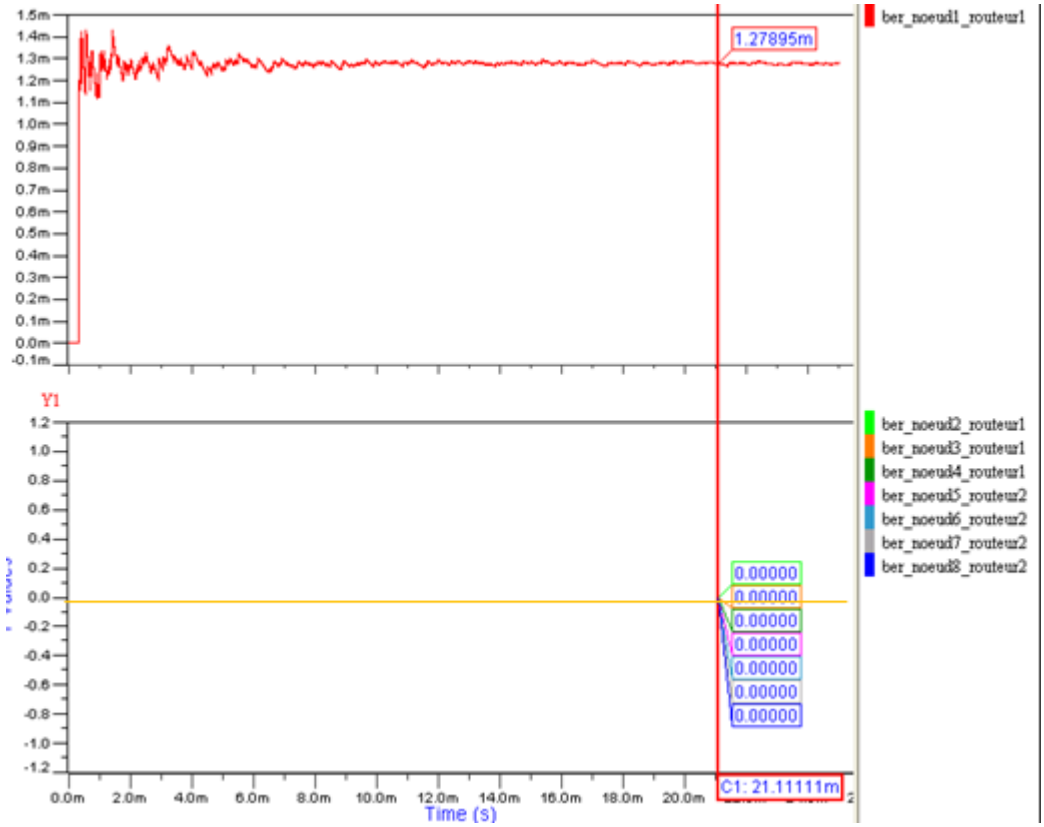


Figure 3. 10 BER noeud_routeur, troisième simulation

Figure 3. 11 BER noeud_routeur_concentrateur, troisième simulation

Comme la valeur de BER de la connexion nœud_1-routeur_1 est la même que celle du nœud_1-routeur_1-concentrateur, on peut déduire que le BER le plus influent

est celui de la première connexion. N'aboutissant pas encore à un réseau de fonctionnement garanti à 100%, cherchons une autre solution : pour cela essayons de changer le type de communication.

3.1.4 Influence du type de communication :

La solution acceptable du point de vue de l'exploitation du réseau n'est toujours pas obtenue puisque bien que les valeurs de BER aient été améliorées, le nombre d'erreurs par trame reste supérieur à 1. L'idée d'avoir augmenté de 100% les énergies consommées par les nœuds est trop simpliste puisqu'elle pénalise la durée de vie de ceux-ci sans même avoir résolu le problème du nœud_1 ! Nous nous proposons dans ce qui suit, tout en gardant 0.1mW comme puissance de transmission, de changer le type de modulation de la communication nœud-routeur en une modulation 4-PSK. Cette dernière a pour effet de diminuer la valeur du BER avec comme contrepartie la diminution du débit. Les caractéristiques du réseau deviennent comme indiqué dans le tableau 3.4.

Tableau 3. 6 Quatrième description technique

	Communique avec	distance	Modulation	puissance	température	Synchronisation
Nœud 1	Routeur 1	25m	4 PSK	0.1mW	300K	1Hz
Nœud 2		20m	4 PSK	0.1mW		
Nœud 3		15m	4 PSK	0.1mW		
Nœud 4		8m	4 PSK	0.1mW		
Nœud 5	Routeur 2	10m	4 PSK	0.1mW		
Nœud 6		12m	4 PSK	0.1mW		
Nœud 7		17m	4 PSK	0.1mW		
Nœud 8		8m	4 PSK	0.1mW		
Routeur 1	Concentrateur	25m	8 PSK	0.3mW		
Routeur 2		15m	8 PSK	0.3mW		

On constate, dans ce dernier cas, que le nombre d'erreurs par trame (figure 3.12) pour cette durée de simulation, est inférieur ou égal à 1. La majorité des données transmises étant récupérées à la réception on peut admettre que c'est une solution acceptable pour réaliser ce réseau.

Remarque : Les communications des différents nœuds avec le concentrateur sont établies et le BER étant petit, le nombre d'erreurs par trame reste inférieur à 1. Dans cette simulation, limitée à 25ms, le cahier des charges est respecté mais le moyennage du BER sur cette durée est insuffisant pour converger vers des valeurs stables. Cet aspect est visible sur la figure 3.12 au niveau de la valeur du BER du nœud_1, lequel

a subi des variations importantes avec l'apparition des premières erreurs dans la trame et que la simulation se termine. Pour converger vers une valeur stable il nous faut un plus grand nombre de bits échangés et par conséquent un temps de simulation plus grand. On verra à ce sujet dans le paragraphe 3.1.6 l'influence du temps de simulation sur la valeur du BER mesurée.

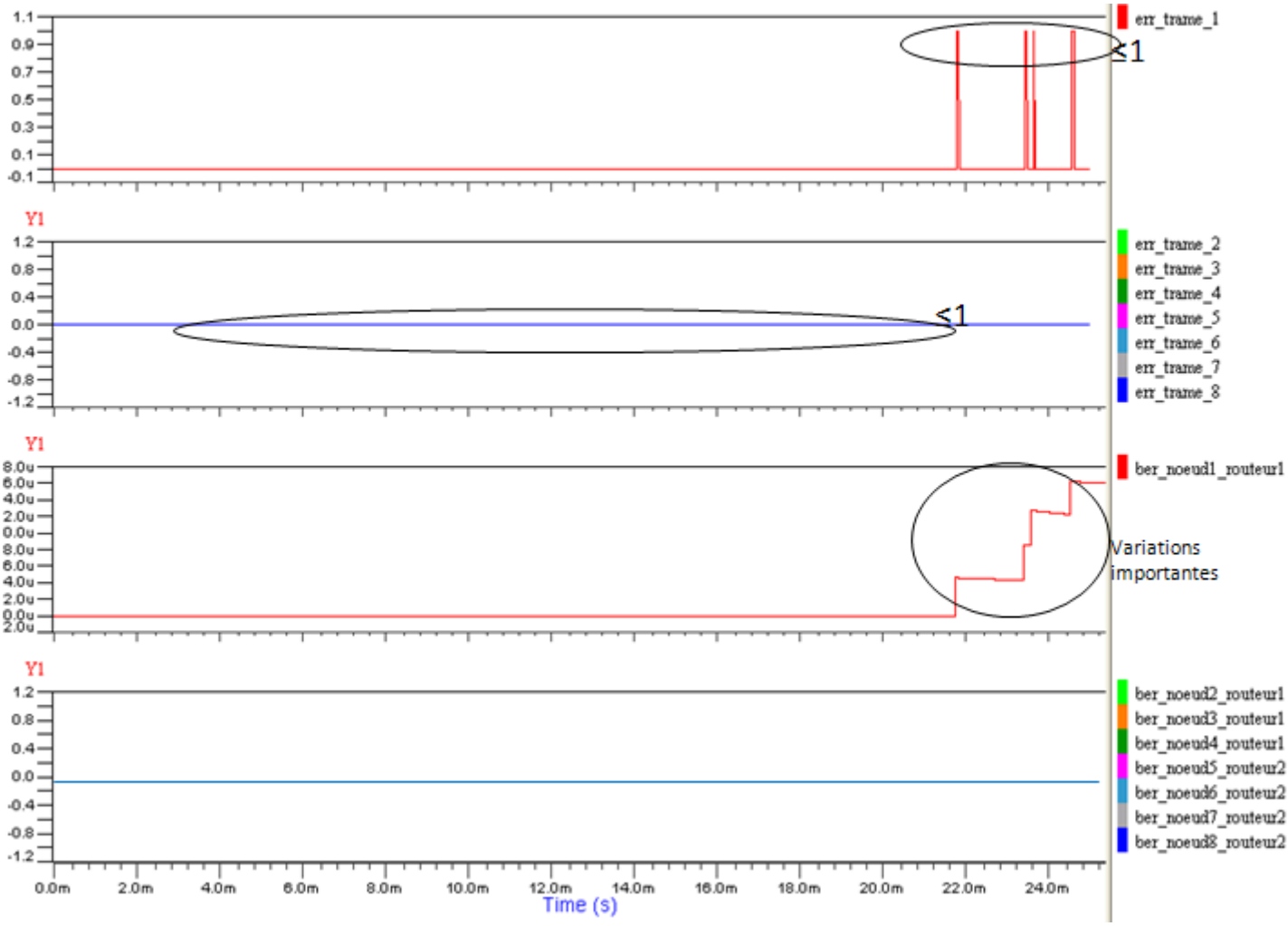


Figure 3. 12 Résultats quatrième simulation

3.1.5 Simulations multi-variables

Un des avantages des modèles construits, est la capacité à créer une sonde pour observer l'évolution d'un paramètre. Pour étayer cet avantage, analysons la consommation d'énergie selon l'évolution temporelle du trafic dans le réseau de capteurs, figure 3.13. Dans cette figure on affiche la puissance et l'énergie consommées par un nœud, dans la première et troisième simulation, (émission à 0,1mW et 0,2 mW respectivement) puis la différence entre les deux.

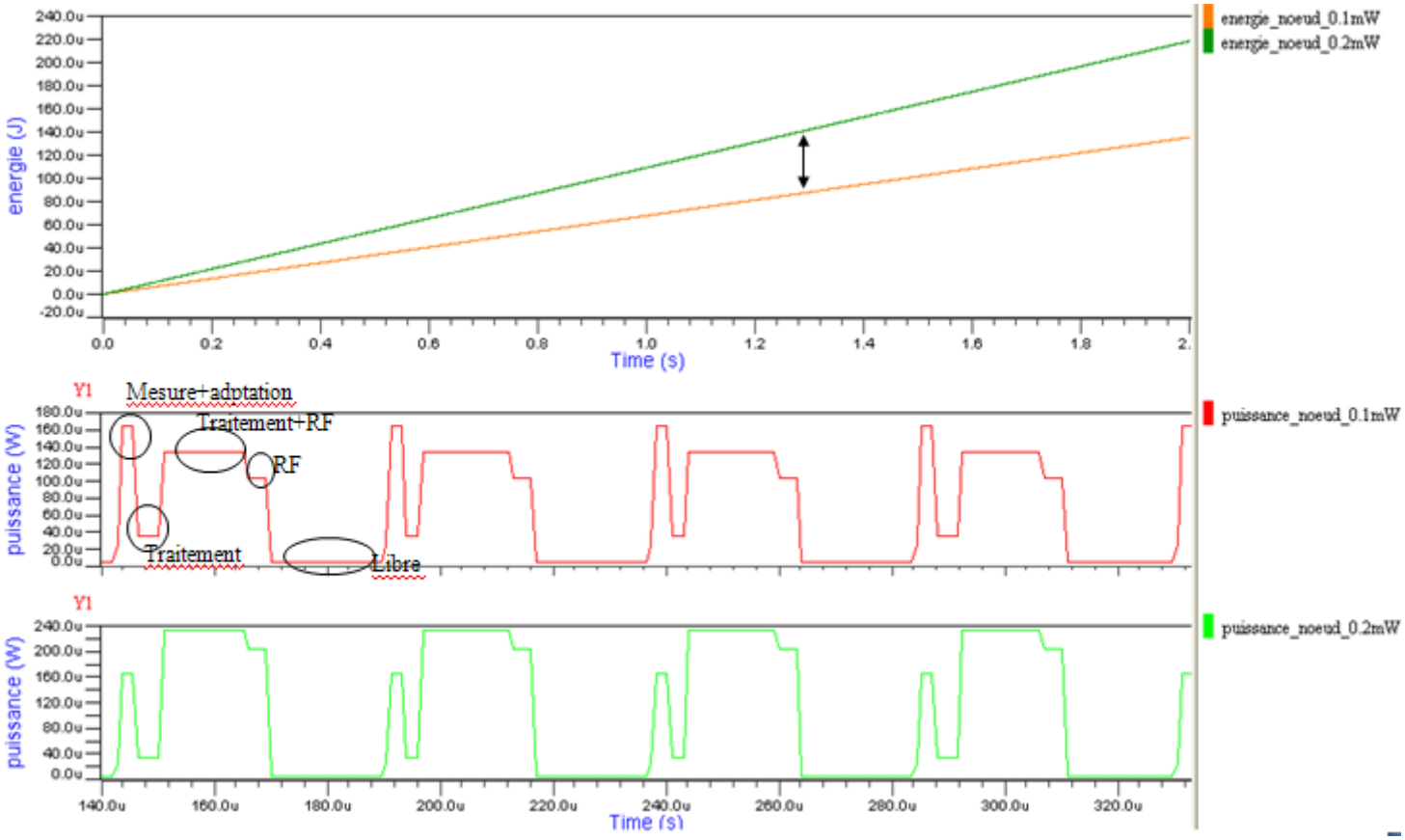


Figure 3. 13 Puissance et énergie consommées par un nœud pour 2 émissions de 0,1 et 0,2 mW

L'énergie consommée dans un nœud est un cumul de celles consommées par les différents capteurs : celle de l'adaptation, celle du traitement et celle de la transmission RF. Ce n'est pas une valeur constante ni une succession de ces valeurs parce que deux ou plusieurs opérations peuvent s'effectuer en parallèle. Dans cette figure on trouve la puissance instantanée pour le nœud pour une émission à 0.1mW et à 0.2mW. Même si on double la puissance d'émission, celles de la mesure et du traitement restent inchangées, donc la consommation globale n'est pas s doublée. En outre, les courbes d'énergie montrent bien l'écart de consommation entre les deux cas pour une période de 2 secondes. Pour une durée plus longue, si on garde le même mode de fonctionnement, la puissance instantanée est répétée, par contre l'énergie consommée est proportionnelle à celle consommée pour cette période de simulation.

Dans le paragraphe 2.3.1 du deuxième chapitre, on avait abordé le modèle de désynchronisation des horloges des nœuds et le modèle comportemental du protocole de synchronisation caractérisé par sa fréquence, sa précision et sa durée d'application. La synchronisation des horloges des différents nœuds peut être intéressante dans certaines applications. Dans la simulation qui suit on va illustrer comment on peut profiter du simulateur pour mesurer la désynchronisation des horloges des nœuds et

étudier l'effet de l'application d'un protocole de synchronisation sur les données envoyées et sur les horloges.

Pour cela dans le réseau abordé dans ce chapitre dont la période d'échantillonnage est de 47µs, on va appliquer un protocole de synchronisation caractérisé par:

- une fréquence d'application d'1Hz
- une durée d'application de 100µs
- une précision de 200ns

Comme résultats de simulation on cherche à voir la désynchronisation des horloges par rapport à une horloge de référence, ici celle du nœud_1, et on va regarder l'influence de ce protocole sur le réseau, figure 3.14.

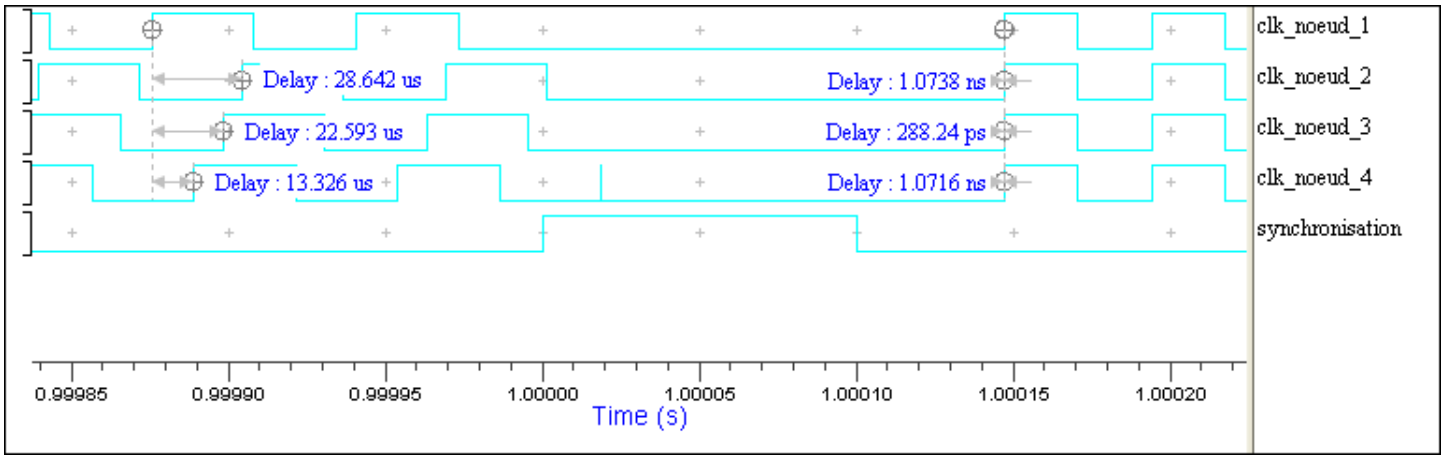


Figure 3. 14 Affichage de la désynchronisation

Dans ces résultats on remarque qu'avant l'application de ce protocole la désynchronisation maximale de ces 4 nœuds était de 28,6µs. La durée d'application du protocole bloque le réseau pour deux échantillons successifs et après l'application on observe des désynchronisations inférieures à 200ns. Ces grandeurs doivent être comparées avec le cahier de charge pour décider de garder ou non ce protocole de synchronisation avec ces caractéristiques.

3.1.6 Dimensionnement du temps de simulation

Notre étude de faisabilité du réseau s'est focalisée sur les critères BER et nombre d'erreurs par trame à la réception. Pour ce faire, on a mis des sondes de mesure de BER sur toutes les communications (16 sondes). En ce qui concerne la simulation du

réseau lui-même, nous avons utilisé dans la modélisation des modèles comportementaux de haut niveau où la simulation du réseau seul, sans sondes, ne pose pas de contraintes sérieuses sur la taille du réseau et sur le temps de simulation.

L'avantage de la simulation réside dans la capacité à analyser au choix et précisément, au travers de sondes de mesure le comportement du réseau. Le « prix à payer » de l'analyse fine va être, en ajoutant au réseau les sondes de mesure, l'augmentation du temps de simulation ainsi que la taille des données intermédiaires à enregistrer. Le tableau 3.5 donne une idée sur ces contraintes lors de la simulation du réseau :

Tableau 3. 7 Dimensionnement des simulations

Nombre de sondes de BER	Temps de simulation	Durée de simulation	Taille des données sauvegardées
16 sondes	25ms	1 heure	5 GOctet
16 sondes	50ms	3 heures	7 GOctet
16 sondes	500ms	Echec	Echec
8 sondes	500ms	8 heures	22 GOctet
1 sonde	500ms	30min	1 GOctet
1 sonde	1sec	50min	2 GOctet
1 sonde	2sec	2 heures	4 GOctet
1 sonde	3sec	4heures	8 GOctet
0 sonde	2sec	15 min	160 MOctet

A l'évidence qui consiste à dire qu'il faut augmenter le temps de simulation pour affiner les résultats il faut opposer l'explosion de la durée de la simulation et de la taille des fichiers de simulation ainsi que les problèmes de convergence numérique.

500ms avec 8 sondes imposent 8h de simulation et 22Go de datas générées et à traiter ! Le concepteur doit donc rechercher un compromis entre nombre de sondes et simulations multiples pour répartir la charge CPU exigée par les modèles exécutés par le simulateur.

Le tableau 3.6 et les figures 3.15 - 3.18 confirment la variation des résultats déjà affichés en fonction du temps de simulation et la corrélation avec les résultats théoriques.

Tableau 3. 8 Variations des résultats de BER en fonction du temps de simulation

		Temps de simulation				Théorique
Communication	Distance	25ms	1sec	2sec	3sec	MatLab
noeud1_routeur1_ 25m_4PSK_0,1mW	25m	7,00E-06	1,40E-05	1,80E-05	1,80E-05	1,80E-05
noeud2_routeur1_ 20m_4PSK_0,1mW	20m	0,00E+00	1,10E-05	1,40E-05	1,40E-05	1,40E-05
noeud3_routeur1_ 15m_4PSK_0,1mW	15m	0,00E+00	0,00E+00	2,00E-08	7,00E-08	1,00E-06
noeud4_routeur1_ 8m_4PSK_0,1mW	8m	0,00E+00	0,00E+00	0,00E+00	0,00E+00	1,20E-08
noeud5_routeur2_ 10m_4PSK_0,1mW	10m	0,00E+00	0,00E+00	0,00E+00	1,00E-09	2,00E-08
noeud6_routeur2_ 12m_4PSK_0,1mW	12m	0,00E+00	0,00E+00	0,00E+00	1,10E-09	1,50E-07
noeud7_routeur2_ 17m_4PSK_0,1mW	17m	0,00E+00	7,00E-06	1,00E-05	1,20E-05	1,20E-05
noeud8_routeur2_ 10m_4PSK_0,1mW	10m	0,00E+00	0,00E+00	0,00E+00	1,00E-09	2,00E-08
routeur1_concentrateur_25m_8PSK_0,3mW	25m	0,00E+00	7,00E-07	7,00E-06	7,00E-06	7,00E-06
routeur2_concentrateur_15m_8PSK_0,3mW	15m	0,00E+00	0,00E+00	1,00E-08	5,00E-08	5,00E-07

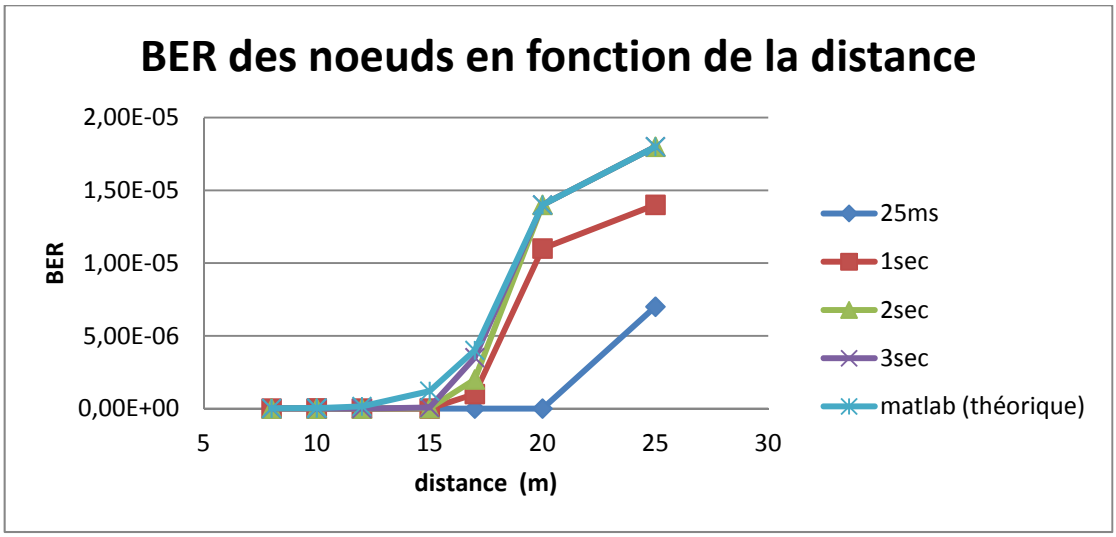


Figure 3. 15 Variations des résultats du BER en fonction du temps de simulation (linéaire)

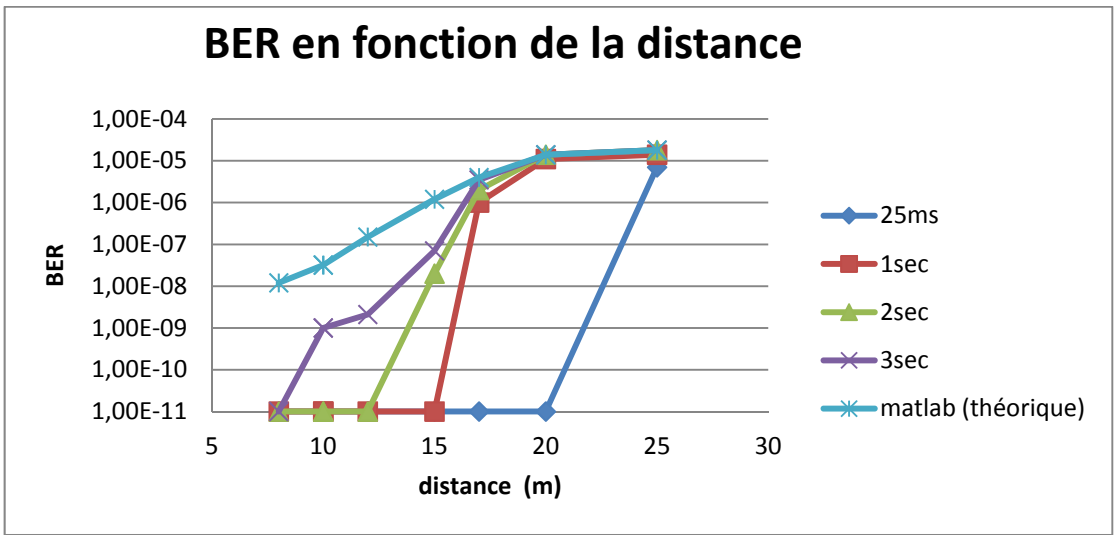


Figure 3. 16 Variations des résultats du BER en fonction du temps de simulation (logarithmique)

Dans la figure 3.15 on affiche la variation de la valeur du BER des différents nœuds en prenant la distance séparant les nœuds des routeurs comme variable de l'axe des abscisses. Les résultats de simulation rapprochent de plus en plus vers les valeurs

théoriques au fur et à mesure qu'on augmente le temps de simulation. Ainsi les valeurs affichées comme des zéros dans les simulations à 25ms cités dans le paragraphe 3.2, ne le seront plus si on change le temps de simulation. Pour le rendre plus visible graphiquement on peut se reporter à l'affichage en échelle logarithmique de la figure 3.16. Pour les très faibles valeurs de BER les 3sec de simulation ne paraissent pas suffisantes, il en faut plus.

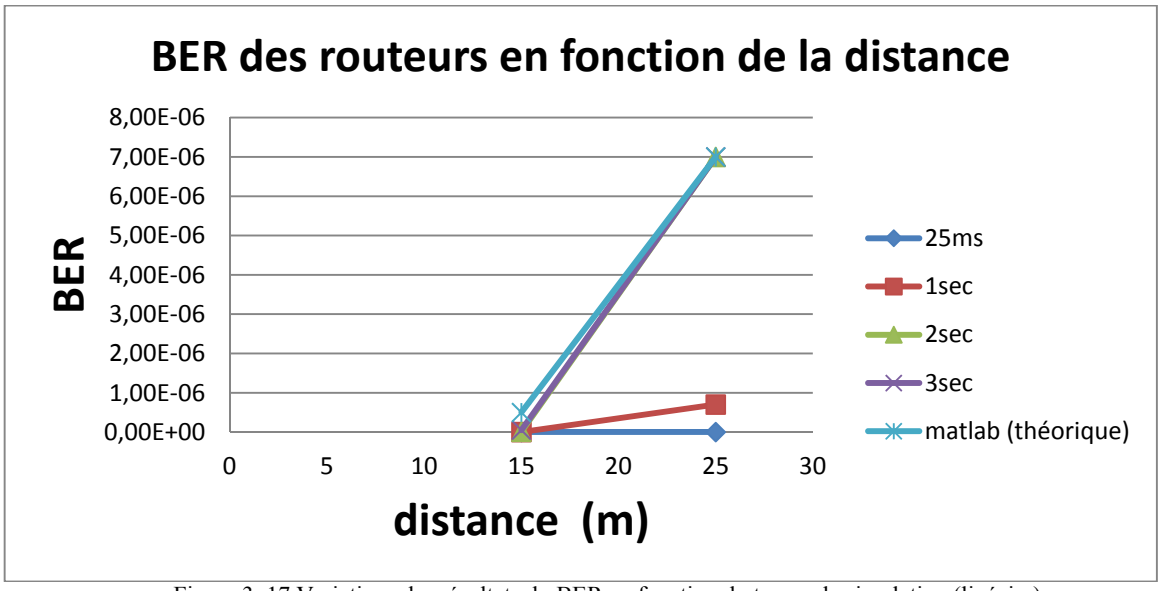


Figure 3. 17 Variations des résultats du BER en fonction du temps de simulation (linéaire)

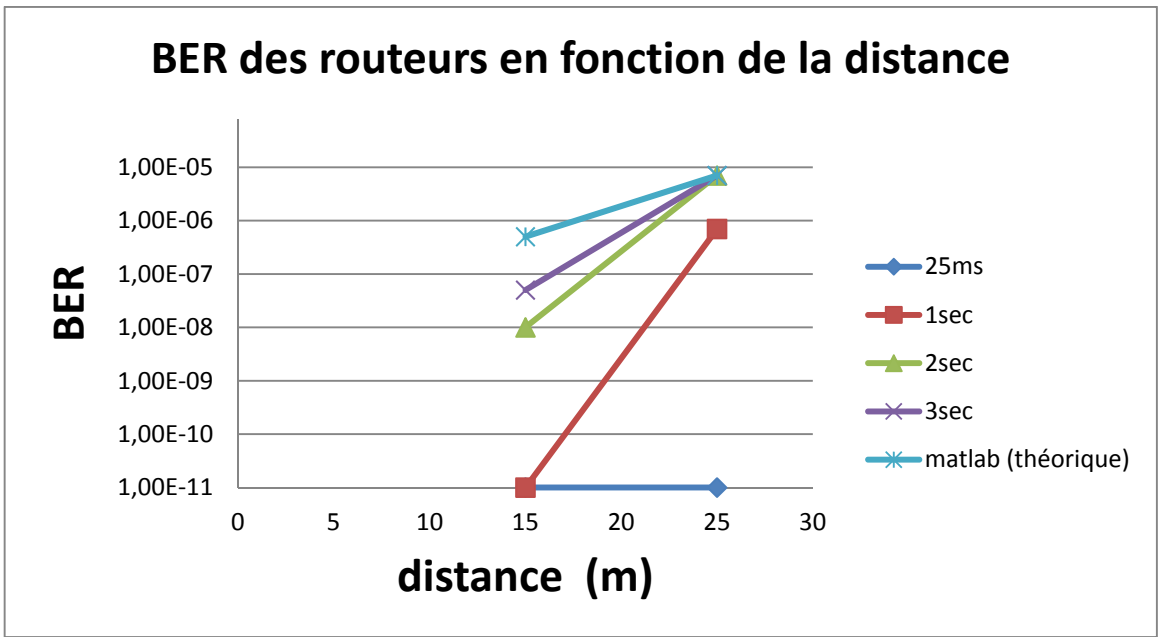


Figure 3. 18 Variations des résultats du BER en fonction du temps de simulation (logarithmique)

Les figures 3.17 et 3.18, semblablement aux deux figures 3.15 et 3.16, affichent la variation de la valeur du BER des différents routeurs en prenant la distance séparant les routeurs du concentrateur comme variable de l'axe des abscisses.

3.2 Simulation sur FPGA :

Afin d'accélérer le processus de conception et minimiser le délai de mise sur le marché, il est intéressant pour le concepteur, de disposer d'un outil de simulation rapide et performant. Dans le paragraphe précédent, cet objectif a été atteint partiellement en utilisant, pour chaque constituant du réseau, des modèles comportementaux de haut niveau, et en effectuant plusieurs simulations successives.

Cependant, les phases itératives d'optimisation conduites sont limitées par la capacité de la simulation à donner des résultats « fiables », ce que conditionne le temps de simulation qui hélas est limité par le CPU de la machine supportant le simulateur. Il parait donc incontournable d'envisager une autre approche.

A présent, focalisons-nous sur l'utilisation d'un environnement de simulation matériel pour un réseau de capteurs sans fil, basé sur la mise en œuvre de circuits logiques. L'idée est de décrire les modèles comportementaux dans un langage de synthèse type VHDL, les associer entre eux pour constituer une architecture, la compiler (transformation en portes) puis la télécharger sur un composant cible type FPGA (NIOS-DEVKIT-1S40 de ALTERA). On conserve donc, la description comportementale précédente des éléments, mais on adapte la traduction des modèles dans le langage approprié.

Une fois téléchargée, cette architecture peut être simulée en temps réel sur des durées qui peuvent être relativement longues si on le souhaite !!!

Plusieurs études et comparaisons [1,2] montrent que l'exécution matérielle directe peut être plus rapide de plusieurs centaines de fois que la simulation logicielle. En outre, la reconfiguration des FPGA permet de reprogrammer directement toute modification dans la phase de conception. Cependant cette possibilité reste à relativiser car toute modification nécessite une recompilation de l'ensemble de l'architecture du réseau et un nouveau téléchargement dans le circuit. Ce processus peut s'avérer très long et peut minimiser le gain de temps espéré par la simulation matérielle.

Pour contrebalancer cet aspect nous avons opté pour 2 méthodes :

- Méthode 1 : concevoir une architecture maximaliste configurable, compilée et téléchargée une seule fois dans le composant et modifiable de manière dynamique par le concepteur.
- Méthode 2 : concevoir un jeu d'architectures plus simples puis choisir celle qui se rapproche le plus de celle désirée.

La deuxième méthode consiste à concevoir plusieurs architectures du réseau, les compiler puis les télécharger sur la cible FPGA pour les simuler et choisir celle qui convient le mieux au cahier de charges. Cette méthode semble a priori acceptable mais si on regarde la faisabilité, surtout en disposant des modèles complets des éléments, on se trouve face à une barrière importante qui est la limite des capacités des FPGA. On va voir dans la figure 3.21 qu'un FPGA peut être presque saturé par une partie du réseau.

Abordons à présent la méthode 1 : celle-ci consiste à concevoir une architecture maximaliste qui permet de configurer le réseau de plusieurs façons sans avoir recours à une recompilation et une reprogrammation du FPGA. Cela veut dire que les modèles sont paramétrables en termes de configuration topologique et autres, et accessibles et modifiables directement sur la cible. Bien que cette solution puisse entraîner une saturation des ressources du FPGA, il est possible d'y remédier en remplaçant le réseau initial par un sous-réseau incluant un ou deux nœuds, un ou deux canaux de transmission, un routeur et un concentrateur. Ce sous-réseau, compilé et téléchargé une seule fois, peut servir ensuite de base pour la simulation du réseau complet. Ainsi plusieurs simulations pourront être lancées successivement après paramétrage des sous-réseaux concernés, figure 3.19 et 3.20.

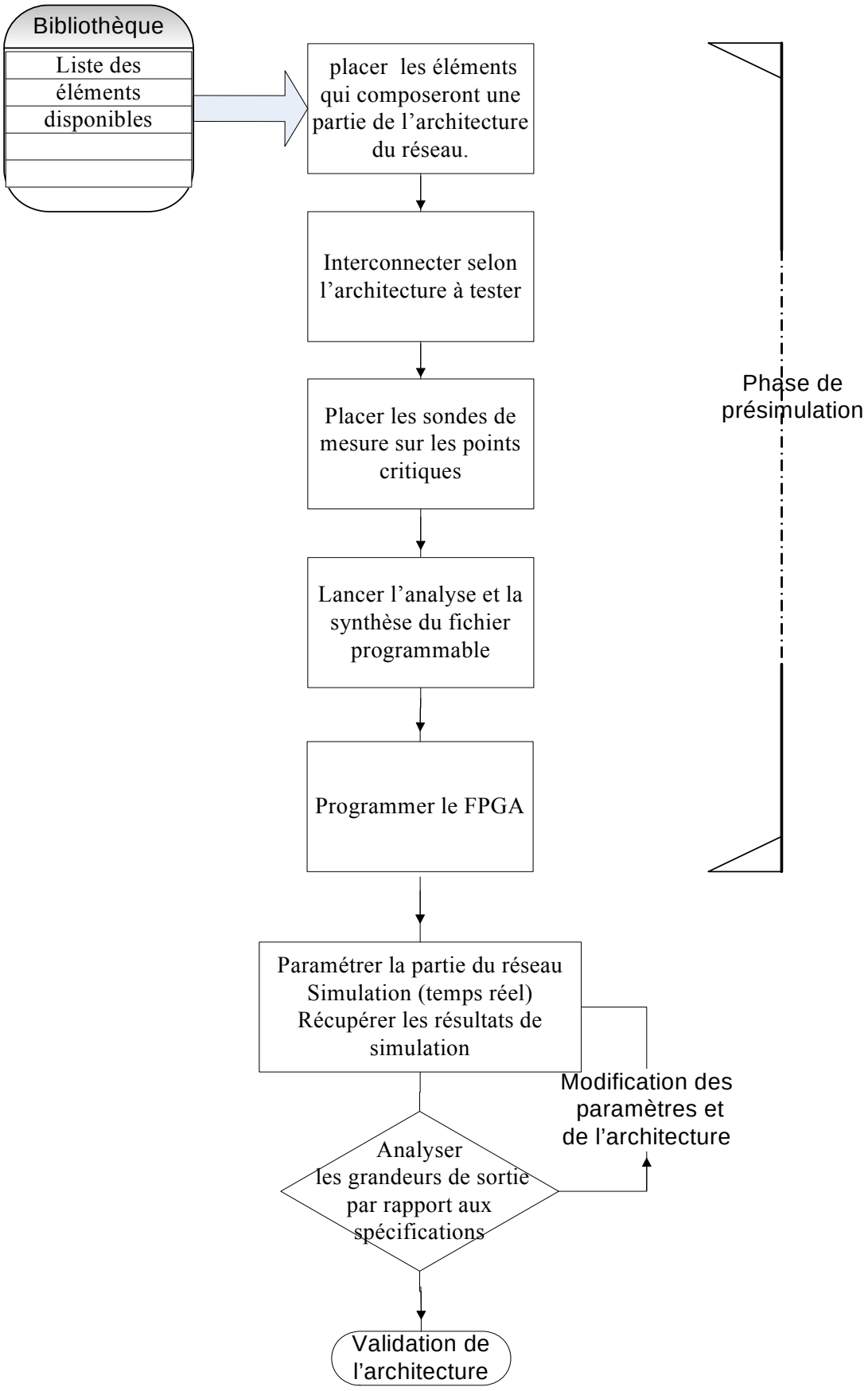


Figure 3. 19 Démarche à suivre pour la simulation sur FPGA

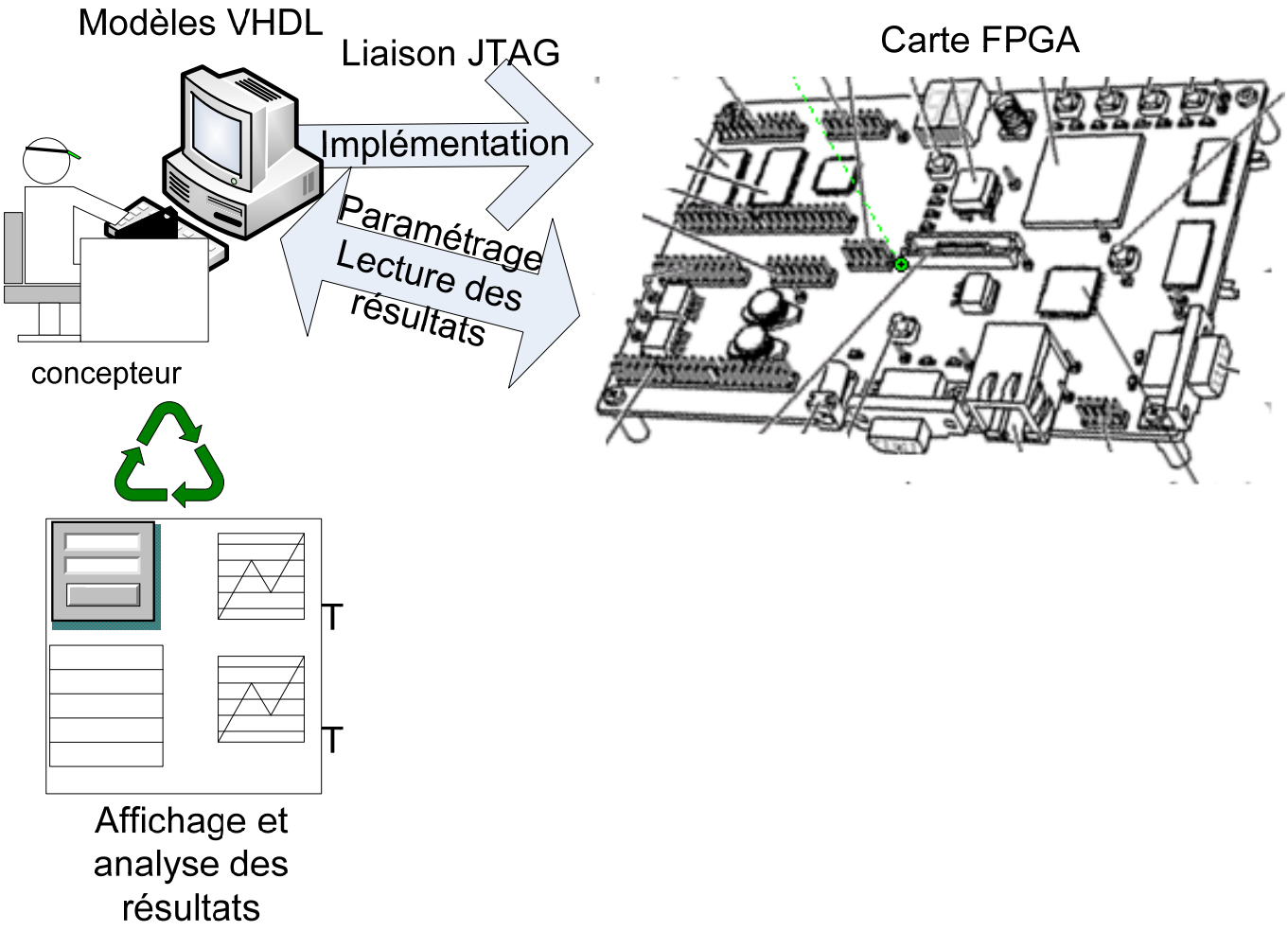


Figure 3. 20 Utilisation du simulateur sur la carte FPGA

Pour expliquer la démarche on va l'illustrer à partir du même exemple décrit et simulé dans le paragraphe 3.2. La procédure commence par la construction de l'architecture complète du réseau. Une telle architecture dépassant largement la capacité de notre FPGA, on prend seulement une partie du réseau (mais qui en intègre tous les éléments de base) et on prend en compte l'influence des canaux de transmission des autres sous-réseaux par un paramétrage adéquat des modèles.

Les paramètres modifiables seront :

- la dispersion
- le nombre de canaux de transmission voisins
- le type de modulation et la distance
- la puissance d'émission et la température
- la fréquence utilisée en GHz

- la bande passante
- le facteur de bruit effectif.

Sur cette partie du réseau on place des sondes de mesure de BER (mémoire de 128 Octets), de nombre d'erreurs par trame (mémoire de 65 kOctets) et la valeur maximale du nombre d'erreurs par trame (mémoire de 32 Octets). La figure 3.21 montre le rapport de la synthèse de notre maquette du réseau : on est à 67% du nombre des éléments logiques disponibles et à 43% de la mémoire disponible.

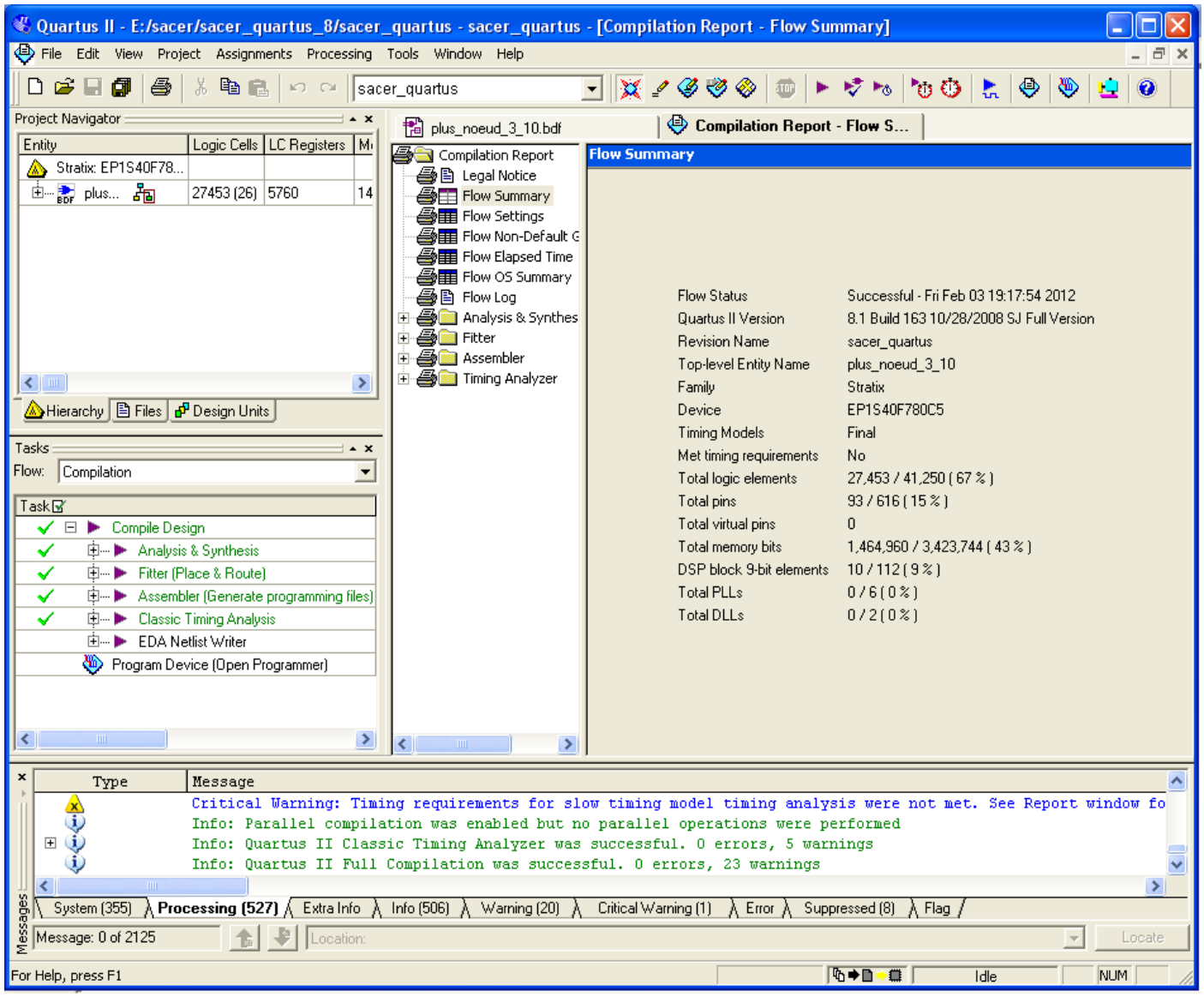


Figure 3. 21 Rapport de la synthèse du réseau

Cette compilation prend 50 min pour l'analyse, la synthèse, le routage, la génération du fichier programmable et l'analyse temporelle. Une fois terminée, on la télécharge sur le FPGA et on lance l'utilitaire « In-System Memory Content Editor »

pour afficher les résultats de la simulation et pour accéder aux paramètres modifiables.

On peut répéter ainsi la même démarche, décrite dans le paragraphe 3.2, en essayant les mêmes paramètres pour converger vers une solution acceptable. Ces différentes simulations itératives seront faites par des changements de paramètres, lesquels sont instantanés, et le temps de simulation « se confond » avec la durée de simulation : on est donc en pseudo temps réel !

La valeur du BER calculée et celle du maximum d'erreurs par trame correspondent à un temps de simulation qui coïncide avec celui de fonctionnement du FPGA : **on peut donc simuler des secondes, des minutes, des heures ou des jours**… Les répartitions des erreurs dans les trames sont récupérées par des échantillons sur des durées limitées par la taille de la mémoire dédiée (dans le cas où on échantillonne toutes les 47µs, 65KOctets correspondent à 3sec de fonctionnement. Au-delà soit les données sont écrasées, soit elles sont récupérées via le bus JTAG).

Dans ce qui suit on se propose d'illustrer la simulation du réseau avec la configuration finale acceptée dans le paragraphe 3.2.4 et résumée ci-dessous, tableau 3.9

Tableau 3. 9 Configuration réseau simulé sur FPGA

	Communique avec	distance	Modulation	puissance	température	Synchronisation
Nœud 1	Routeur 1	25m	4 PSK	0.1mW	300K	1Hz
Nœud 2		20m	4 PSK	0.1mW		
Nœud 3		15m	4 PSK	0.1mW		
Nœud 4		8m	4 PSK	0.1mW		
Nœud 5	Routeur 2	10m	4 PSK	0.1mW		
Nœud 6		12m	4 PSK	0.1mW		
Nœud 7		17m	4 PSK	0.1mW		
Nœud 8		8m	4 PSK	0.1mW		
Routeur 1	Concentrateur	25m	8 PSK	0.3mW		
Routeur 2		15m	8 PSK	0.3mW		

La figure 3.22 est une copie d'écran de la fenêtre de « In system Memory Content Editor » et illustre la définition des paramètres du premier nœud

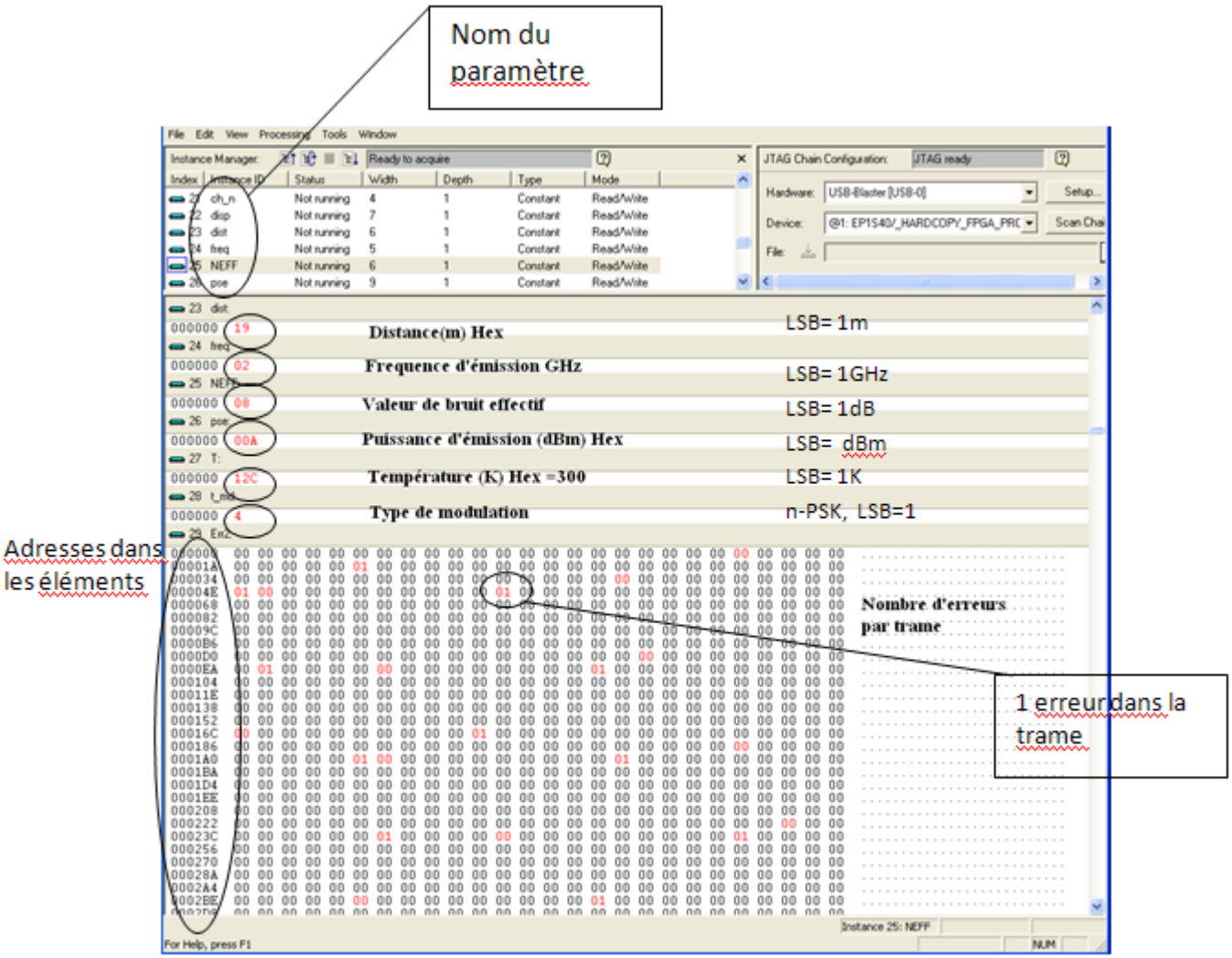


Figure 3. 22 Paramètres du premier nœud

Après l'écriture de ces paramètres dans les mémoires correspondantes du système, on peut récupérer les résultats de simulation. Dans la figure 3.22 apparaît une partie de la mémoire contenant le nombre d'erreurs par trame : ces valeurs peuvent être récupérées dans un tableur (Excel, origine…) pour les analyser et les tracer en fonction du temps ; d'autres résultats apparaissent dans la figure 3.23 : ce sont les valeurs de BER calculées et la valeur maximale d'erreurs par trame.

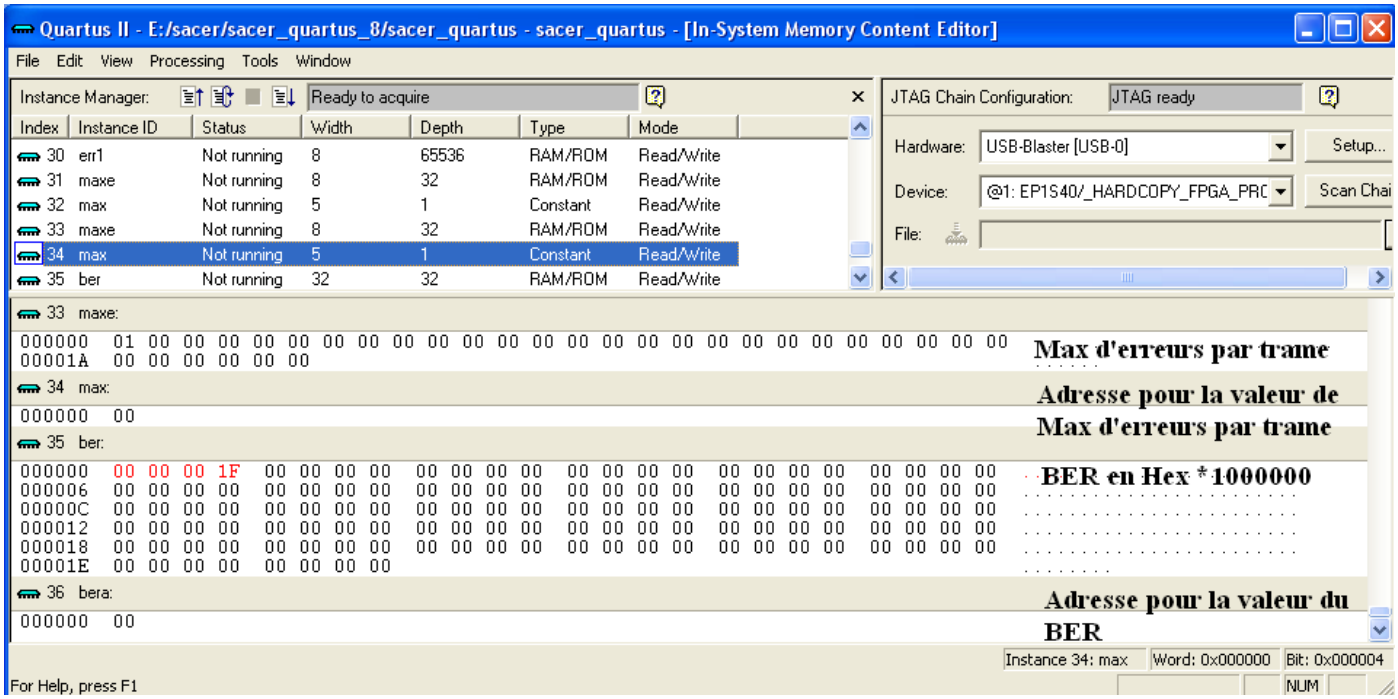


Figure 3. 23 Résultats de simulation du premier nœud

Dans la figure 3.21 on voit que le max d'erreurs obtenues est 1 et la valeur du BER est de $1\text{Fhex}\times10^{-6}=31\times10^{-6}$. Ces valeurs sont en corrélation avec celles trouvées dans le paragraphe 3.2.4

De la même façon on peut continuer pour paramétrer tous les éléments du réseau comme décrit dans le tableau 3.9 et illustré dans la figure 3.24

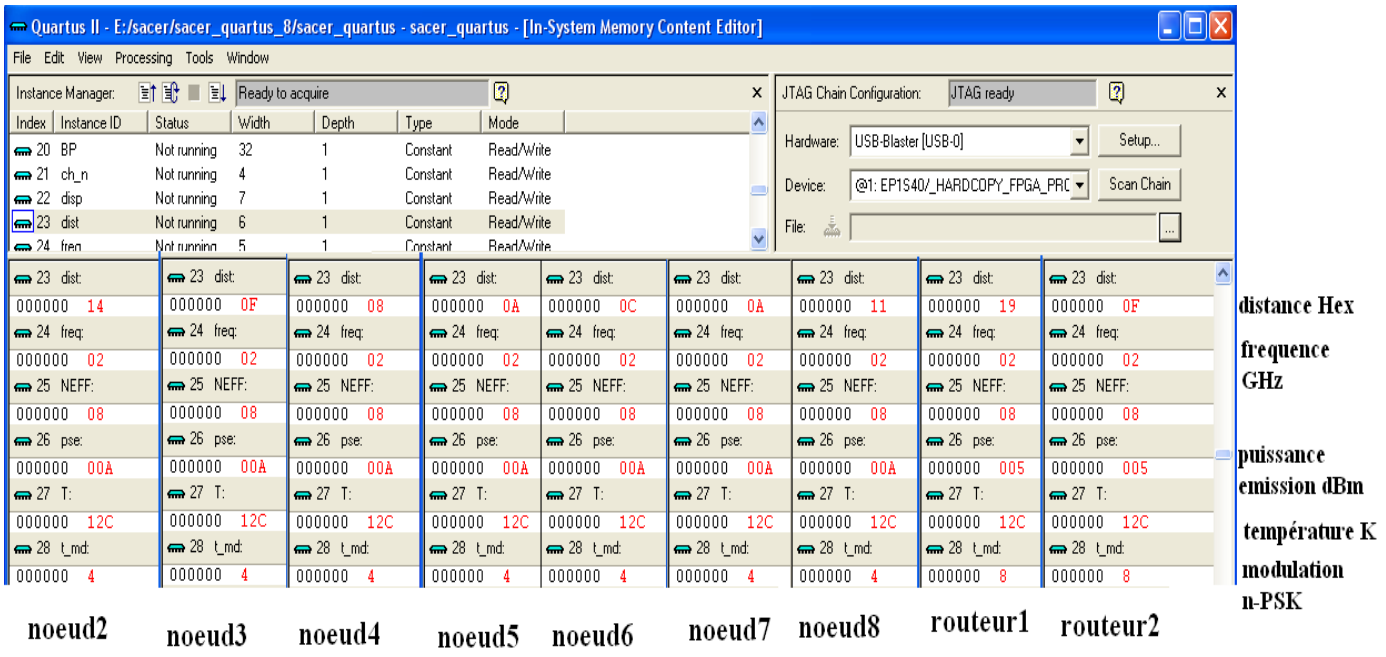


Figure 3. 24 Paramétrage du réseau

Les résultats de la simulation sont listés dans la figure 3.25

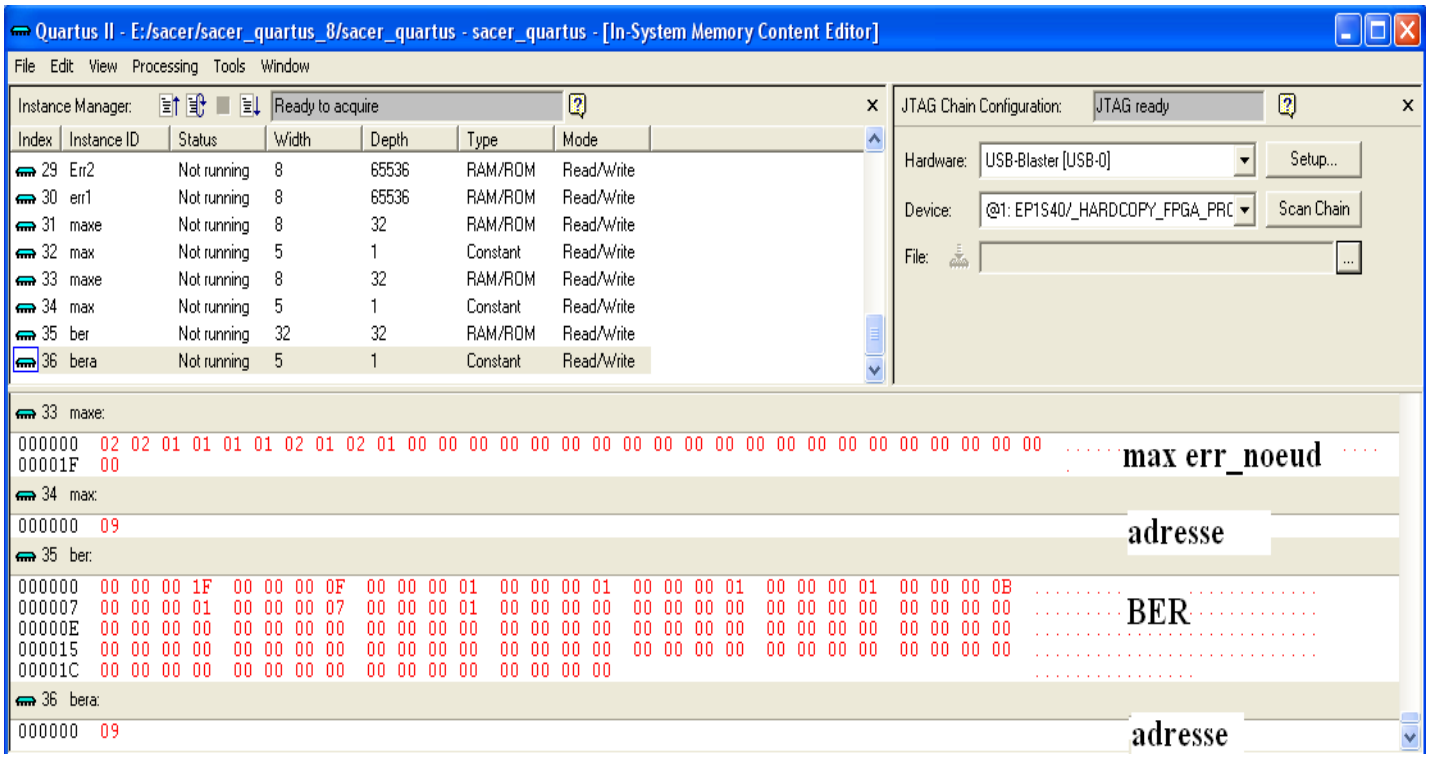


Figure 3. 25 Résultats de la simulation du réseau

Ces résultats correspondent bien à ceux qui ont été obtenus avec l'ancien simulateur pour des temps de simulation plus importants (1 minute pour chaque simulation). Ces simulations tournent en temps réel.

Pour simuler le réseau complet, on a fait une succession de 8 simulations (l'architecture construite et téléchargée dans le FPGA ne représente qu'une partie du réseau).

Chaque simulation se déroule comme suit :

- Le paramétrage en une minute
- La simulation qui dure 1 minute pour 1 minute de simulation.
- deux minutes pour la lecture et l'analyse rapide des résultats.

Donc la simulation et la lecture des résultats du réseau complet pour un temps de simulation de 1 minute peuvent se faire en 30 minutes après le téléchargement sur le FPGA. Il reste à rappeler qu'on travaille avec le VHDL synthétisable et donc seuls les entiers sont autorisés. Aussi les valeurs réelles (comme le BER) sont arrondies et multipliées par 10^6 pour être représentées en entier, ce qui implique une valeur

minimale de BER = 1×10^{-6}. On a supposé cette valeur comme étant une valeur minimale possible dans les communications sans fils.

3.3 Comparaison des deux simulateurs :

Le tableau 3.10 illustre une comparaison entre les deux simulateurs pour les différentes valeurs de BER récupérées par les simulations.

Tableau 3. 10 Comparaison des valeurs de BER entre résultats théoriques, SystemVision et FPGA

		3sec	1 min	Théorique
Communication	Distance	SystemVision	FPGA	MatLab
noeud1_routeur1_ 25m_4PSK_0,1mW	25m	1,80E-05	31E-6	1,80E-05
noeud2_routeur1_ 20m_4PSK_0,1mW	20m	1,40E-05	16E-6	1,40E-05
noeud3_routeur1_ 15m_4PSK_0,1mW	15m	7,00E-08	1E-6	1,00E-06
noeud4_routeur1_ 8m_4PSK_0,1mW	8m	0,00E+00	1E-6	1,20E-08
noeud5_routeur2_ 10m_4PSK_0,1mW	10m	1,00E-09	1E-6	2,00E-08
noeud6_routeur2_ 12m_4PSK_0,1mW	12m	1,10E-09	1E-6	1,50E-07
noeud7_routeur2_ 17m_4PSK_0,1mW	17m	1,20E-05	11E-6	1,20E-05
noeud8_routeur2_ 10m_4PSK_0,1mW	10m	1,00E-09	1E-6	2,00E-08
routeur1_concentrateur_25m_8PSK_0,3mW	25m	7,00E-06	7E-6	7,00E-06
routeur2_concentrateur_15m_8PSK_0,3mW	15m	5,00E-08	1E-6	5,00E-07

Pour illustrer le gain dans la durée de simulation, la figure 3.26 montre la différence de la durée de simulation entre le simulateur écrit en VHDL-AMS sur SystemVision et celui implémenté sur un FPGA pour différents temps de simulation. La figure 3.27 montre la différence des durées de simulation entre les deux simulateurs pour plusieurs simulations.

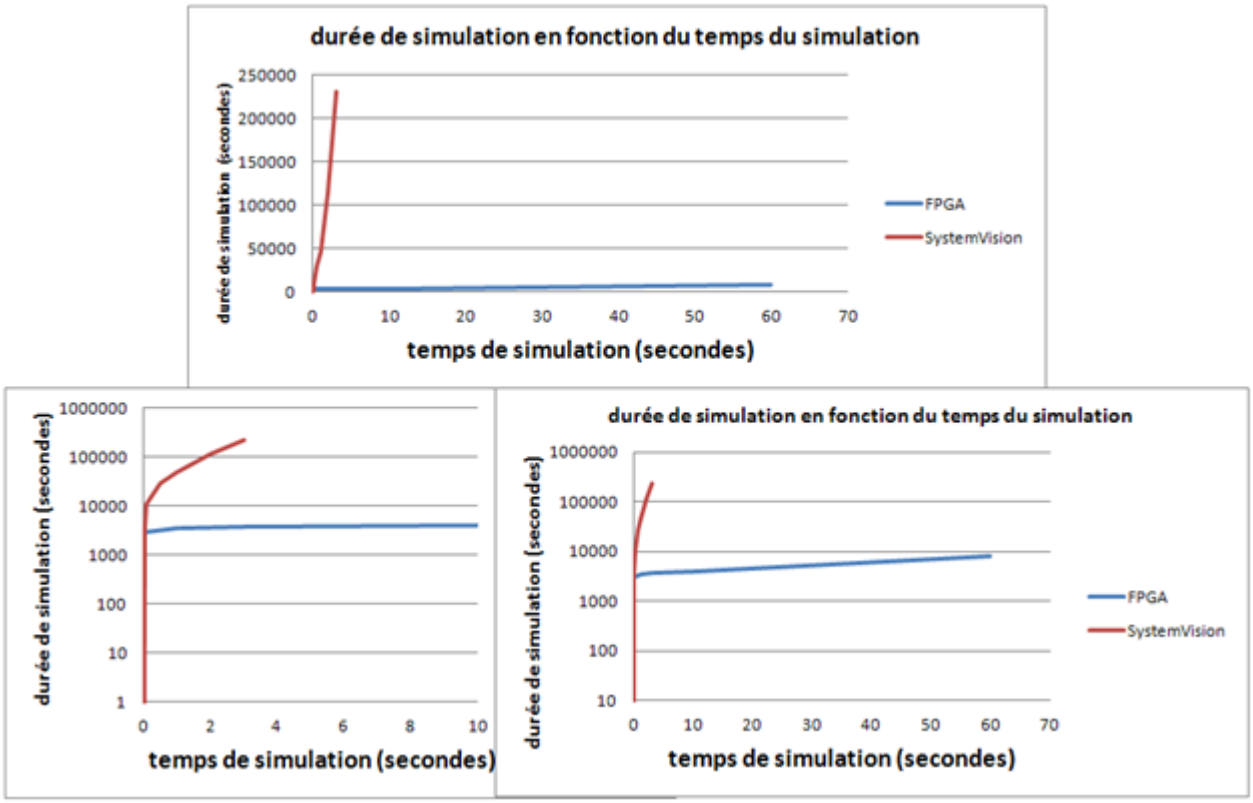


Figure 3. 26 Evolution de la durée de simulation en fonction du temps de simulation

Dans la figure 3.26 la limitation du simulateur codé en VHDL-AMS et simulé sous SystemVision, est bien visible : la durée de simulation est croissante avec le temps de simulation avec une pente importante. Par contre, la durée de simulation pour le simulateur codé en VHDL et utilisé sur un FPGA est modestement croissante. Des simulations avec des temps de simulation importants sont alors envisageables.

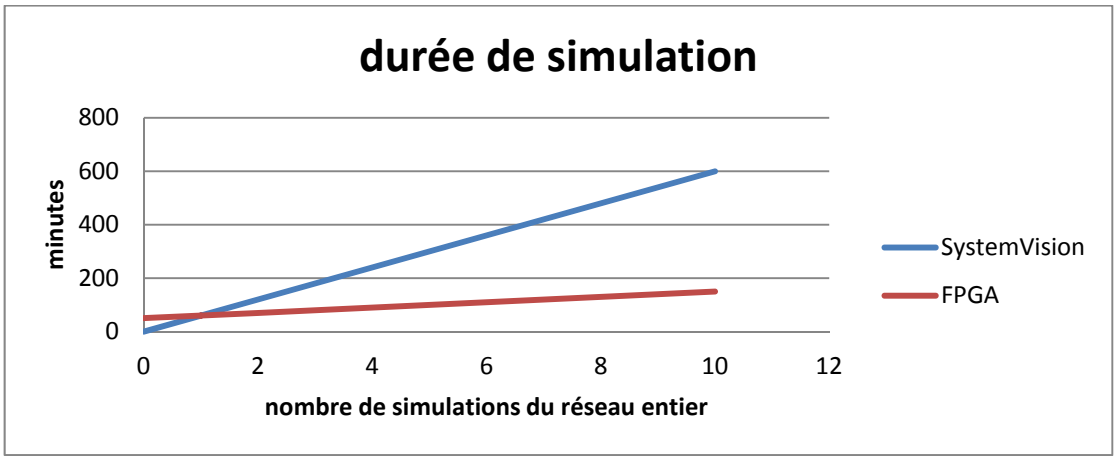


Figure 3. 27 Evolution de la durée de simulation en fonction du nombre de simulations

La figure 3.27 illustre l'évolution de la durée de simulation en fonction du nombre de simulations : pour le simulateur codé en VHDL-AMS, la durée globale est proportionnelle à la durée d'une seule (k simulations demandent k×durée d'une

simulation). Par contre pour le simulateur opérant sur le FPGA, la durée est égale au nombre de simulations multiplié par le temps de simulation et en ajoutant la durée de compilation.

3.4 Simulation du projet SACER

La simulation du réseau SACER a permis de converger vers une topologie en grappe à 2 niveaux de hiérarchie : les nœuds regroupent 8 capteurs chacun et sont eux-mêmes regroupés dans 3 gloves répartis sur les ailes de l'avion figure 3.28 (vert : nœuds, bleu : routeurs).

Figure 3. 28 Architecture du réseau SACER [3]

Vu le contexte d'utilisation du projet SACER (domaine aéronautique ou spatial), où la sécurité est l'impératif premier, l'utilisation d'une bande de fréquence permettant un niveau d'émission compatible avec les modulations bandes étroites classiques de type FSK ou PSK est impossible. Pour contourner cette difficulté nous avons utilisé des modulations de type Ultra Large Bande (UWB), lesquelles nous ont permis de passer à des puissances d'émission extrêmement basses (inférieures à -41dBm/MHz). Dans cette bande, parmi les trois méthodes éligibles : technologie par impulsion (IR-UWB), porteuses orthogonales (OFDM) et multi porteuses à porteuses orthogonales (MB-OFDM), c'est cette dernière qui a été retenue car elle nous a permis d'établir les connexions avec les débits souhaités de 100Mbits/s, pour des distances comprises entre 7 m et 17 m. [3][4].

La première, bien que paraîssant avantageuse en terme de consommation, nécessite de disposer de convertisseurs numériques analogiques puissants et a une faible efficacité spectrale. Ces résultats, validés en simulation, initient l'étape de fabrication des composants aux normes aéronautiques.

3.5 Conclusion

Dans ce chapitre nous avons testé les deux simulateurs de réseau de capteurs sans fil : celui décrit en VHDL-AMS et celui implémenté sur circuit FPGA. L'objectif principal de ces simulateurs est de disposer d'un outil d'évaluation très rapide qui nous permette de faire des choix quant à la pertinence d'une architecture de réseau pour un cahier des charges donné. Le dernier outil permet d'obtenir des gains de temps en simulation dépassant un facteur 100 comparé au premier. [5][6].

Le tableau 3.11 résume les différences entre les 2 simulateurs :

Tableau 3. 11 Caractéristiques des simulateurs

Simulateur VHDL-AMS	**Simulateur implémenté sur FPGA**
Durées de simulation	
Tableau 3.7	Temps de compilation (45min) + temps réel de période de simulation
Le même temps pour toute modification de l'architecture ou des solutions techniques	Instantané avec les modifications des paramètres + temps réel de période de simulation
Limité par les capacités des PC	Aucune limite sur le temps de simulation
Modélisation	
Modèles mixtes,	Modèles logiques
quantités réelles	Valeurs entières
équations mathématiques et différentielles	Relations mathématiques simples
Instructions gérant le temps	Pas d'instructions de gestion du temps, présence d'horloges
Lecture des résultats	
Affichage graphique et outil de calcul	Sauvegardés dans les mémoires internes des sondes
Possibilité de visualiser tous les signaux intermédiaires	Pas de sauvegarde autre que dans les sondes de mesure
Consommation énorme de mémoire	Seulement mémoire interne des sondes
Tous les résultats sont visibles à la fin de la simulation	Seul, le dernier remplissage de mémoire est visible (on réécrit dans les mémoires une fois pleines)

Références

[1] M. Bauer, W. Ecker, R. Henftling, A. Zinn, "A Method for Accelerating Test Environments", Proc. 25th Euromicro Conf., vol. 1, IEEE Press, 1999. pp. 477-480.

[2] Del Re A.; Nannarelli A.; Re M., "Fast prototyping techniques applied to the hardware simulation of telecommunication systems" Signals, Systems and Computers, 2001. Vol 2, 4-7 Nov. 2001 pp. 1314 – 1317.

[3] J.HENAUT, D.DRAGOMIRESCU, R.PLANA. Développement d'un réseau de capteurs pour des applications aérospatiales: topologie réseau et interface radio. Journées Nationales du Réseau Doctoral en Microélectronique (JNRDM 2010), Montpellier (France), 7-9 Juin 2010.

[4] J.HENAUT, D.DRAGOMIRESCU, F.PERGET, R.PLANA. Validation of the MB-OFDM Modulation for High Data Rate WSN for Satellite Ground Testing. 5th International Conference on Systems (ICONS 2010), Menuires (France), 11-16 Avril 2010.

[5] N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS, C.ESCRIBA,. Channel transmission behavioral modeling dedicated to WSNs. Journal of Energy and Power Engineering, à apparaitre Oct., 2012.

[6] N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS, C.ESCRIBA,. Embedded FPGA simulator for Wireless Sensor Network design. Journal of Energy and Power Engineering, à apparaitre Oct., 2012.

Généralisation et conclusion générale

Généralisation et conclusion générale

Généralisation de l'utilisation des simulateurs

L'objectif de nos travaux était de développer un outil de simulation de réseaux de capteurs sans fils. Nous avons développé ce simulateur en nous basant sur une application industrielle qui est le projet SACER. Celui-ci consiste à répartir un grand nombre de capteurs sur les ailes d'un avion pour effectuer des mesures en continu à une fréquence élevée lors des essais en vol. Les données collectées sont envoyées à très haut débit directement au concentrateur, sans traitement local des valeurs mesurées. Le réseau complet est situé sur l'avion et ne fait pas l'objet de reconfiguration topologique. Les modèles ont été développés pour répondre au contexte et prendre en compte les divers facteurs influents.

Lors de ces développements nous avons essayé de construire des modèles généraux afin de couvrir toute forme de réseau : ce sont des modèles génériques paramétrables ayant la possibilité d'ajout de fonctions pour répondre à d'éventuels besoins ultérieurs.

Bien que la simulation décrite dans ce chapitre représente une miniaturisation du réseau SACER possédant les mêmes caractéristiques, on peut utiliser ce dernier pour d'autres types de réseaux en adaptant les paramètres des modèles convenablement et en ajoutant d'autres fonctions si nécessaire. Pour illustrer ce propos nous allons décrire les démarches à faire pour effectuer une simulation d'un nouveau réseau dont les caractéristiques sont passablement différentes.

Supposons que l'on veuille simuler un réseau de capteurs sans fils destiné à surveiller une voiture prototype de « formule 1 » durant une course, figure 4.1, ce réseau peut être décrit comme suit:

- nombre relativement petit de nœuds
- nœuds différents : mesure de pression, de température, des oscillations, accéléromètres, émissions…
- chaque nœud a sa fréquence de mesure

- Les nœuds sont intelligents, ils analysent la valeur mesurée pour décider d'envoyer ou non l'information au concentrateur
- Les nœuds sont en déplacement par rapport aux routeurs d'où un effet possible de la vitesse sur le BER (effet Doppler)

D'autres réseaux sont disponibles dans le même endroit

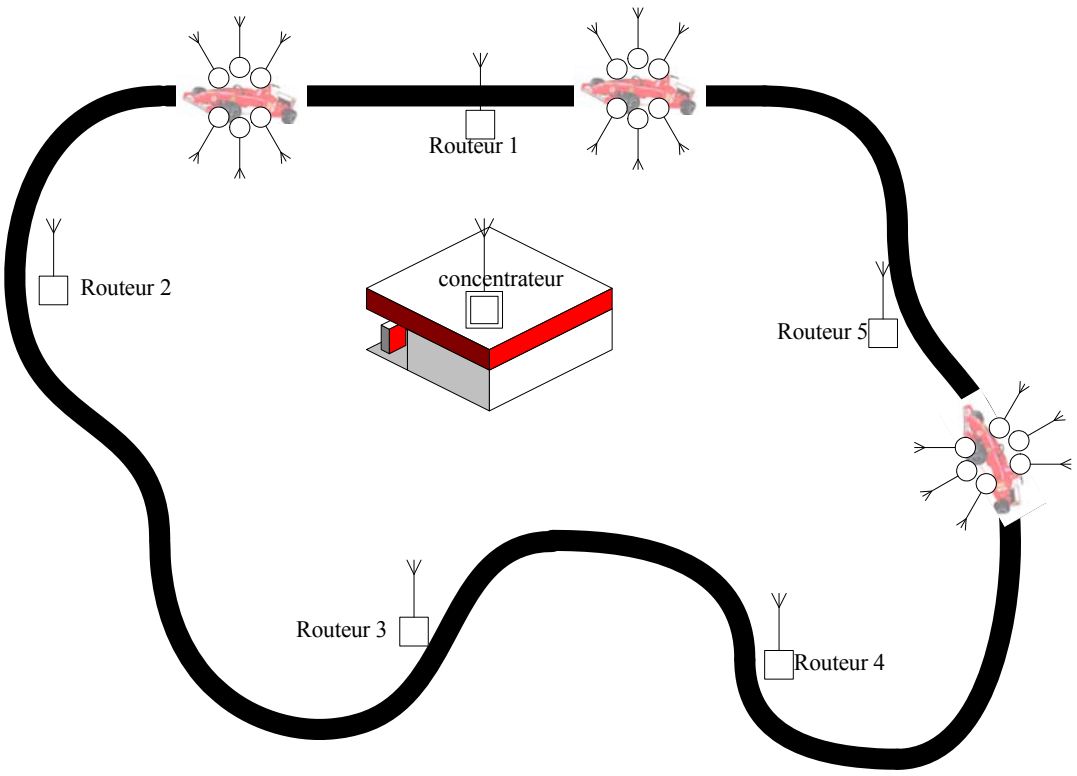


Figure 4. 1 Réseau de capteurs sans fils pour surveiller les voitures de F1

Pour simuler ce réseau :

- Le choix du nombre de nœuds ne pose pas de problèmes dans le simulateur.
- Chaque nœud est composé d'un seul capteur : nous avons la possibilité dans le simulateur de désactiver les 7 autres.
- Le choix de la fréquence d'échantillonnage et celle de la conversion numérique se fait par des paramètres propres à chaque nœud.
- Même si les capteurs sont actifs, le nœud peut être activé ou non. C'est en ajoutant une fonction de comparaison par rapport à un seuil au niveau du

capteur dont la sortie vient modifier le paramètre d'activation du nœud, qu'on peut ajouter l'aspect intelligent du nœud. Ainsi la transmission ne sera active que si la valeur mesurée nécessite d'être envoyée au concentrateur. Bien que dans ce cas cette fonction de décision soit une simple comparaison, il est possible de la choisir plus complexe.

- Les déplacements nœuds/routeurs n'existaient pas dans l'exemple précédent : ici l'influence de ceux-ci peut se traduire par une perturbation sur la liaison de communication. La première influence est que la distance séparant le nœud du routeur est une distance variable et donc ce paramètre n'est plus une valeur constante : c'est une fonction triangulaire dont la pente est l'image de la vitesse, figure 4.2. La deuxième influence est la perturbation causée par le déplacement sur les ondes émises : l'effet Doppler. Cet effet peut être introduit sous deux formes : la première est d'introduire l'effet de la vitesse sur le rapport SNR, la deuxième est d'introduire l'effet directement comme un facteur de multiplication de la valeur du BER. En effet notre modèle de canal de transmission calcule un BER de départ puis, en fonction des facteurs perturbant, vient modifier cette valeur. En conséquence il faut traduire les lois qui décrivent l'influence de la vitesse sur la valeur du BER par un tableau de facteur multiplicateur, la figure 4.3 illustre une étude [1] de l'influence de la vitesse sur le BER pour différentes valeurs du SNR. C'est à partir de telles études qu'on peut construire ce tableau. Le modèle du nœud est surdimensionné pour être enrichi par plusieurs tableaux à deux dimensions comme celui-ci.

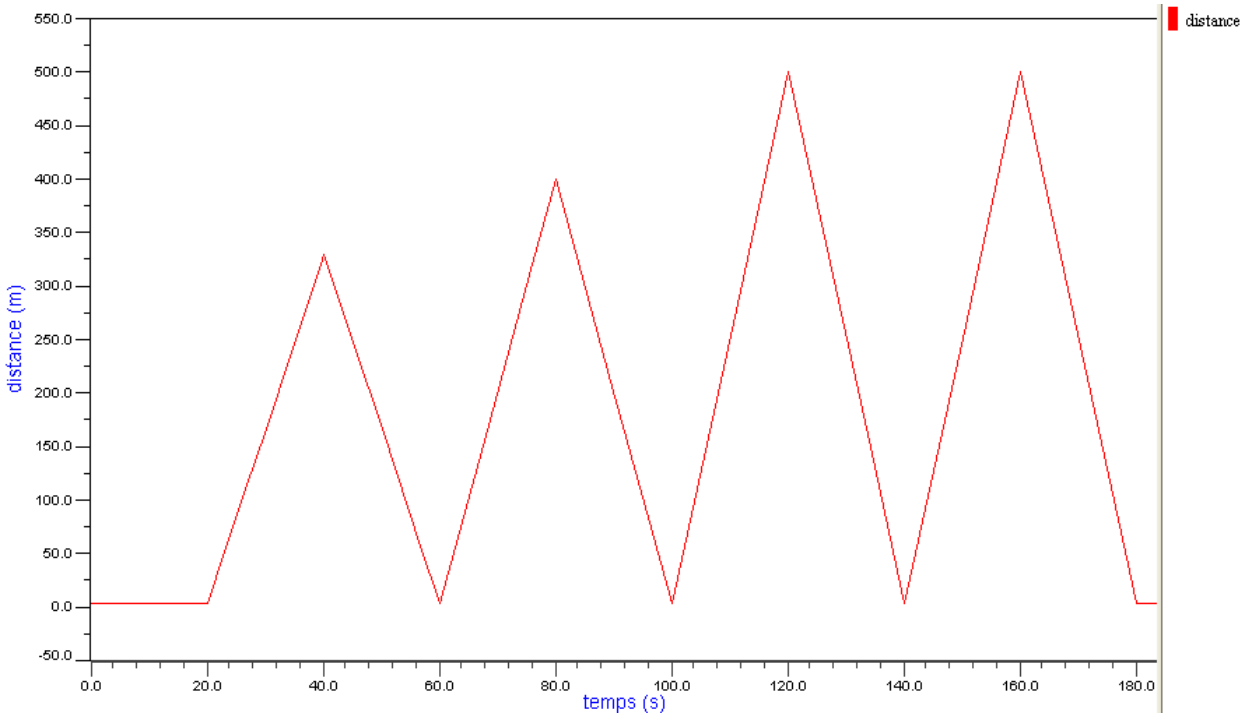


Figure 4. 2 Variation de la distance séparant la voiture des nœuds

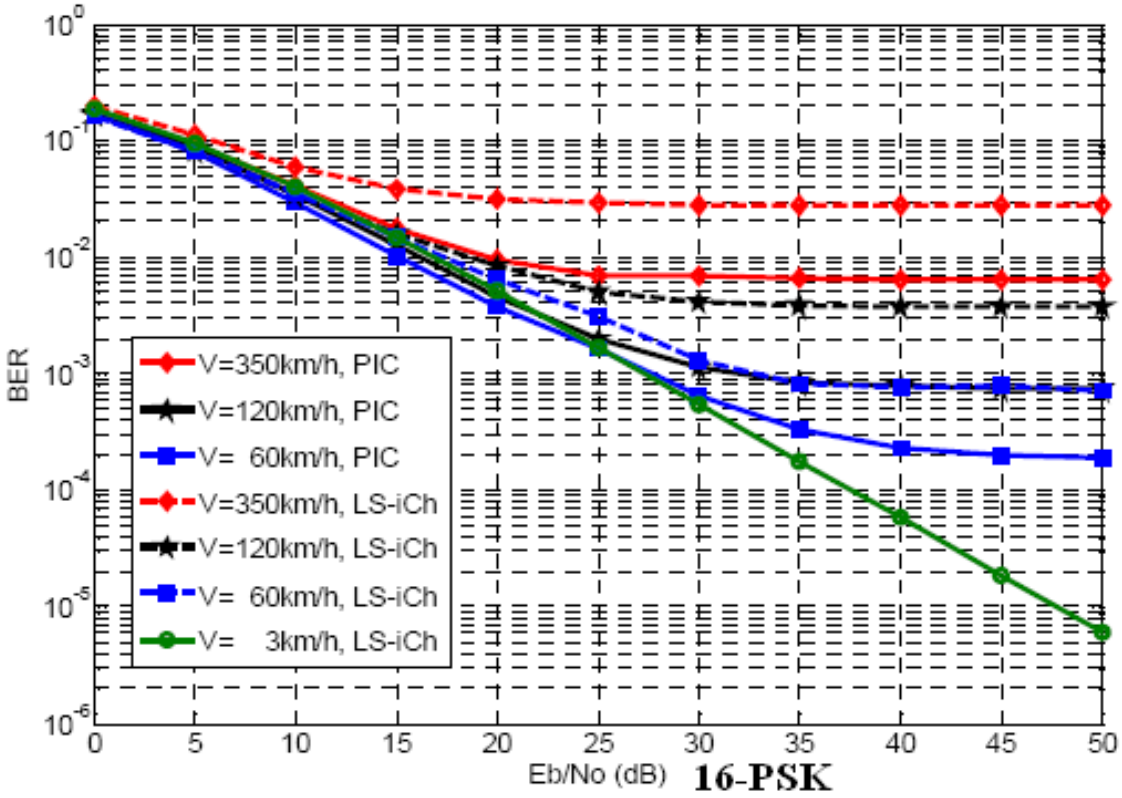


Figure 4. 3 Etude de l'effet de la vitesse sur le BER [1]

- La présence d'autres réseaux est aussi représentée par le tableau qui décrit l'effet de la diaphonie sur le BER.

Conclusion générale

Les systèmes de surveillance et de diagnostic basés sur les réseaux de capteurs sont une classe importante de systèmes complexes nécessitant le développement de méthodes et d'outils de conception. En se fondant sur l'exemple du projet SACER et en nous appuyant sur les recommandations de l'ingénierie système, nous nous sommes efforcés de dégager des solutions nouvelles pour les verrous identifiés dans les étapes de choix technologiques et d'optimisation du réseau.

Quand on parle de réseau, deux approches complémentaires sont nécessaires :

- Une approche centrée sur le protocole de communication : couches du protocole, composition de la trame, reconfiguration, recherche du trajet de communication optimum… Ces aspects sont communs à tout réseau, embarqué ou non, et constituent le domaine des spécialistes réseaux.
- L'autre approche focalise sur la faisabilité technique d'un réseau conformément aux performances attendues. Cette approche doit prendre en compte l'environnement dans lequel le réseau va fonctionner : bruits divers, type de modulation, qualité du media, consommation énergétique, contraintes temps réel…

Nos travaux se sont clairement déroulés dans le cadre de cette deuxième approche. Après avoir, dans un premier temps, présenté le cadre de notre travail, puis montré l'insuffisance des outils existants, nous avons étudié et proposé deux outils de simulation comportementale pour les RCSFs permettant de favoriser l'exploration architecturale et la reconfiguration topologique par la réduction des durées de simulation et nous avons clôturer l'étude par une démonstration applicative de ces simulateurs et la généralisation à d'autres domaines d'utilisation. Il est à noter toutefois que certains aspects liés à la première approche peuvent être pris en compte par un paramétrage adéquat des modèles :

Le premier simulateur est décrit en langage VHDL-AMS

Le second simulateur en VHDL synthétisable

Ils sont tous basés sur des approches comportementales. Les deux langages, bien que VHDL-AMS soit dans le prolongement de VHDL, présentent des différences dans l'analyse du comportement, dans la façon de coder le même algorithme ou le même comportement et dans la complexité à réaliser une tâche.

Dans le premier chapitre, nous introduisons la classe des systèmes de surveillance basés sur la mise en œuvre de réseaux de capteurs autonomes, communicants sans fil. C'est une classe de systèmes très importante et de grande actualité dans le cadre de la dynamique d'évolution de l' « intelligence ambiante ». La conception de tels systèmes nécessite de développer des méthodes et des outils adaptés que nous proposons de puiser dans les recommandations de l'ingénierie systèmes. Cette nécessité est d'autant plus évidente que les performances visées pour ces réseaux de capteurs sont extrêmes : nous basons notre réflexion sur les exigences du projet SACER de déploiement de 800 capteurs, fonctionnant en temps réel, pour la surveillance d'une aile d'avion lors d'essais en vol.

La présentation de nos objectifs nous conduit à privilégier la modélisation-simulation du réseau de transmission selon deux priorités :

- L'évaluation des performances dans les conditions d'usages,
- La recherche de solutions technologiques alternatives et de leur optimisation en relation avec le cahier des charges.

La deuxième partie du premier chapitre est consacrée à l'analyse des outils existant pour la modélisation-simulation des réseaux : en fait la plupart de ces simulateurs sont orientés vers la simulation des protocoles de communications, mal adaptés à l'optimisation d'une architecture de mesure et à la recherche des choix technologiques : nous fixons donc les spécifications d'un autre type de simulateur mieux adapté au travail d'ingénieurs tel qu'il nous importe dans le projet SACER, par exemple.

Les spécifications sont de pouvoir associer modèles fonctionnels et fiches de paramétrage pour chacun des éléments du réseau : capteur, nœud, canal de transmission,… : on doit aboutir ainsi, par assemblage de ces modèles à un modèle de réseau complet simulable. L'originalité de notre approche est de proposer à l'ingénieur-concepteur de « paramétrer » son modèle selon les alternatives que lui

offrent les technologies. En s'appuyant sur le simulateur, il peut comparer des options très diverses et selon des critères déduits de l'application du cahier des charges, hiérarchiser puis sélectionner la meilleure solution possible ! On voit que ces spécifications générales pour le simulateur correspondent bien aux deux exigences d'évaluer les performances et de pouvoir choisir une option technologique par essais successifs, aller retour entre bibliothèque des technologies et simulateur (approche : meet in the middle).

Nous proposons alors dans le deuxième chapitre les deux simulateurs : « logiciel » basé sur un langage de modélisation-simulation : VHDL-AMS, et « matériel » basé sur le langage de description matériel VHDL. Le VHDL-AMS est un langage utilisé pour décrire des systèmes mixtes, logiques et analogiques, tandis que le VHDL est réservé exclusivement aux systèmes logiques et le VHDL synthétisable, du fait de sa capacité à être implémenté sur circuit logique, est d'un usage encore plus restrictif. Ses restrictions viennent essentiellement de sa difficulté à gérer le temps et les différents formats de variables et c'est la virtuosité du concepteur électronicien qui permet de les contourner. S'agissant de la modélisation VHDL-AMS, celle-ci est beaucoup plus puissante car elle permet de manipuler des équations différentielles, des modèles analogiques, des instructions temporelles,… Le VHDL-AMS, pour l'instant, ne peut être que simulé sous forme logicielle, et donc l'environnement matériel reste un PC ou une station de travail. Les performances en termes de temps de simulation, stockage de données, restent hautement liées aux performances de ces machines, ce qui limite les possibilités d'analyse d'un réseau.

En contrepartie le VHDL synthétisable peut être implémenté sur une cible matérielle, FPGA par exemple, et s'exécuter en temps réel. Cet aspect offre une simulation plus rapide et réduit les problèmes de saturation de mémoire et de temps calcul.

Bien que les FPGA disposent également de ressources limitées en termes de cellules logiques et de mémoire, il n'y a pas nécessité de sauvegarder les données intermédiaires. Il suffit d'avoir une surveillance des paramètres importants, lesquels peuvent éventuellement être redirigés vers un calculateur hôte pour sauvegarde ou affichage. En ce qui concerne la récupération et l'exploitation des résultats de simulation, le VHDL-AMS devient plus avantageux que le VHDL synthétisable car il est possible de tirer profit des outils d'affichages et d'analyse graphiques associés à

tout logiciel de simulation et offre une possibilité d'investigation inégalable. En ce qui concerne les FPGA, les résultats peuvent être sauvegardés dans des mémoires internes et accessibles via des utilitaires tel celui intégré à l'environnement de développement Quartus. Celui-ci permet de venir lire et écrire de manière dynamique dans ces mémoires internes ainsi que dans différents registres configurés pour la circonstance. Il est alors possible d'exploiter ces données dans un autre environnement plus adapté. Cette possibilité trouve toutefois ses limites dans le fait que les mémoires internes peuvent être écrites à une cadence très supérieure à celle de la liaison JTAG par laquelle ces données sont récupérées, ce qui peut entraîner des pertes d'information si on ne met pas en place de mécanismes de synchronisation. Un autre intérêt de pouvoir modifier des contenus de registres ou de mémoires à la volée réside dans le fait qu'il est possible de faire évoluer dynamiquement les paramètres de l'architecture et vérifier quasi instantanément leur incidence sur les performances du réseau. Ceci évite de reconcevoir une nouvelle architecture, de la recompiler et la re-synthétiser : le gain de temps s'en trouve ainsi très appréciable.

Les simulateurs que nous avons développés permettent à un concepteur d'évaluer les performances d'un réseau de capteurs en lui indiquant les BER, PER et retard à chaque niveau de la chaîne de communication. Cela permet de choisir la meilleure répartition des nœuds et routeurs ainsi que le code correcteur d'erreurs le plus approprié pour satisfaire un cahier de charges. Il est également possible d'avoir, en temps réel, une bonne estimation de la consommation énergétique du réseau afin d'intégrer ce paramètre supplémentaire dans la définition de l'architecture ainsi qu'une simulation du mécanisme de synchronisation de l'échantillonnage des données. Ces critères ont été le sujet de présentation dans l'application exemple démontrée dans le troisième chapitre pour valider les simulateurs que nous avons développés et pour présenter leurs capacités dans la simulation. Cette présentation est suivie par une comparaison entre les caractéristiques et la puissance des deux simulateurs.

Nos propositions sont évidemment générales : le chemin de recherche et de mise au point que nous avons suivi pourra être utilisé pour d'autres applications, notamment dans la conception des systèmes de surveillance multi sensoriels pour des domaines très variés : Transport, Energie, Santé, Environnements,… C'est pour

démontrer cette faisabilité que nous avons commencé ce chapitre par une généralisation de l'utilisation des simulateurs sur d'autres applications.

Perspectives

Il reste enfin à noter que les modèles créés dans ces simulateurs sont des modèles génériques pouvant représenter une majorité de cas possibles de réseaux de capteurs. Chaque modèle dispose d'un tableau de paramètres configurés par défaut et aisément modifiables par le concepteur soit à partir de jeux de valeurs proposées soit à partir de nouvelles valeurs qui pourraient être issues d'une expérimentation par exemple.

Notre travail s'est centré sur la simulation des réseaux de capteurs : comme nous l'avons souligné lors du lancement de ce projet, il convient d'inscrire cette contribution nouvelle dans la démarche générale de l'ingénierie systèmes consacrée aux systèmes électroniques : une extension utile de ce travail sera de construire une bibliothèque de modèles VHDL-AMS et VHDL, connectée sur une bibliothèque des technologies pour rendre plus automatique la démarche de conception entre le niveau des spécifications et la formalisation des exigences et le prototypage virtuel tel que nous le préconisons ici. Aussi une autre extension peut être utile : c'est la construction d'une interface visuelle de paramétrage et d'évaluation des résultats, pour améliorer l'interface homme-machine dans l'utilisation de ces simulateurs.

Références

[1] BAOUNI ABDELKHALEK, mémoire UNIVERSITE MENTOURI CONSTANTINE FACULTE DES SCIENCES DE L'INGENIEUR, 2010.

Liste des publications

Journaux internationaux

N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS, C.ESCRIBA,. Channel transmission behavioral modeling dedicated to WSNs. Journal of Energy and Power Engineering. A apparaitre Oct., 2012[3*].

N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS, C.ESCRIBA,. Embedded FPGA simulator for Wireless Sensor Network design. Journal of Energy and Power Engineering. A apparaitre Oct., 2012[4*].

Congrès internationaux

N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS. Defects localization tool for wireless sensors network conception. 10th IEEE ECMS-2011 (Electronic Control, Measurement and Signals), June 1-3, 2011, Liberec, Czech Republic,

N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS. Transmission channel behavioral model for a wireless sensors network simulator. IEEE International Conference on Electronics Circuits and Systems (ICECS 2010), Athènes (Grèce), 12-15 Décembre 2010, pp.924-927

N.NASREDDINE, J.L.BOIZARD, C.ESCRIBA, J.Y.FOURNIOLS. Wireless sensors networks emulator implemented on a FPGA. IEEE International Conference on Field-Programmable Technology (FPT'10), Beijing (Chine), 8-10 Décembre 2010, pp.279-282

N.NASREDDINE , J.L.BOIZARD , C.ESCRIBA , J.Y.FOURNIOLS. Synthesizable VHDL models for the simulation of a wireless sensors network. ECMS'2009 : 9th International Workshop on Electronics, Control, Modelling, Measurement and Signals (ECMS 2009), Mondragon (Espagne), 8-10 Juillet 2009, 6p.

---

[3] Sujet: Acceptance Letter of paper JEPE11042802
Date : Wed, 14 Sep 2011 16:59:00 +0800
De : Journal of Energy and Power Engineering <energy@davidpublishing.com>
Organisation : Davidpublishing
Dear Dr.
I am pleased to inform you that the manuscript JEPE11042802* " Channel transmission behavioral modeling dedicated to WSNs "//*has been accepted by the */Journal of Energy and Power Engineering/*//for publication.

[4] Sujet: Acceptance Letter of paper JEPE11042803
Date : Wed, 14 Sep 2011 16:59:02 +0800
De : Journal of Energy and Power Engineering <energy@davidpublishing.com>
Organisation : Davidpublishing
Dear Dr.
I am pleased to inform you that the manuscript JEPE11042803 *" Embedded FPGA simulator for Wireless Sensor Network design "//*has been accepted by the */Journal of Energy and Power Engineering/*//for publication.
* Thanks for your mail. The papers are arranged to publish about Oct., 2012. We will contact you when there is any information about them. Please be a little patient.
Any question, please feel free to contact us.
Best regards,
Journal of Energy and Power Engineering ISSN 1934-8975, USA
David Publishing Company

N.NASREDDINE, J.L.BOIZARD, J.Y.FOURNIOLS, J.HENAUT, D.DRAGOMIRESCU, A.COUSTOU. VHDL-AMS behavioral models for the simulation of wireless sensors networks. SSD'2009 : 6th International Multi Conference on Systems, Signals and Devices (SSD'09) International Conference on Sensors, Circuits and Instrumentation Systems (SCI), Djerba (Tunisie), 23-26 Mars 2009, IEEE, 2009, pp.214-216.

Congrès nationaux

J.L.BOIZARD, N.NASREDDINE, J.Y.FOURNIOLS. Modélisation d'un canal de transmission à bruit thermique pour un émulateur d'un réseau de capteurs communicant sans fil. JNRDM'10 : Journées Nationales du Réseau Doctoral en Microélectronique, 7-9 juin 2010, Montpellier, France, 4 pages.

Ecole d'Hiver Francophone

J.L.BOIZARD, N.NASREDDINE, C.BAUD, D.GUIHAL, C.ROSSI, C.ESCRIBA, J.Y.FOURNIOLS. Méthodologie de conception des SOC-SIP hétérogènes : application à la conception d'un réseau de capteurs. Ecole d'Hiver Francophone sur les Technologies de Conception des systèmes embarqués Hétérogènes (FETCH 2008), Montebello (Canada), 7-9 Janvier 2008, 2p.

Glossaire

BER	Bit Error Rate
CEAT	Centre d'Etudes Aéronautiques de Toulouse
CNRS	Centre National de la Recherche Scientifique
EIA-632	Electronic Industry Alliance 632
FPGA	Field Programmable Gate Array
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronic Engineer
IHM	Interface Homme Machine
IR-UWB	Impulse Radio Ultra-wideband
LAAS	Laboratoire d'Analyse et d'Architecture des Systèmes
M&S	Modélisation et Simulation
MANET	Mobile Ad hoc NETworks
MB-OFDM	Multiband Orthogonal Frequency Division Multiplexing
MCU	MicroController Unit
OFDM	Orthogonal Frequency Division Multiplexing
PAN	Personal Area Network
PBS	Pairwise Broadcast Synchronization
PC	Personnal Computer
PER	Packet Error Rate
PME	Petite et Moyenne Entreprise
PTP	Precision Time Protocol
RBS	Reference Broadcast Synchronization
RCSF	Réseau de Capteurs Sans Fil
ROM	Read Only Memory
SACER	Système Autonome Communicant En Réseau
SHM	Structural Health Monitoring
SNR	Signal to Noise Ratio
SysML	Systems Modelling Language
TPSN	Timing-sync Protocol for Sensor Networks
UML	Unified Modeling Language
UWB	Ultra Wide Band
V&V	Vérification et Validation
VHDL-AMS	VHSIC Hardware Description Language – Analog Mixed Signal
WPAN	Wireless Personal Area Networks
WSN	Wireless Sensor Network

Liste des figures

Liste des tableaux