



Architecture configurable d'une boucle à verrouillage de phase totalement numérique : application à la mesure de gigue et de la dérive en fréquence en télévision numérique

Christian Mannino

► To cite this version:

Christian Mannino. Architecture configurable d'une boucle à verrouillage de phase totalement numérique : application à la mesure de gigue et de la dérive en fréquence en télévision numérique. Ordinateur et société [cs.CY]. Université Henri Poincaré - Nancy 1, 2006. Français. NNT : 2006NAN10171 . tel-01747392

HAL Id: tel-01747392

<https://hal.univ-lorraine.fr/tel-01747392>

Submitted on 29 Mar 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : ddoc-theses-contact@univ-lorraine.fr

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4

Code de la Propriété Intellectuelle. articles L 335.2- L 335.10

http://www.cfcopies.com/V2/leg/leg_droi.php

<http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm>



U.F.R. Sciences et Techniques, Mathématiques, Informatiques, Automatique
École Doctorale Informatique Automatique Electronique Electrotechnique Mathématiques
Département de formation Doctorale Électronique Électrotechnique

Architecture configurable d'une boucle à verrouillage de phase totalement numérique : application à la mesure de gigue et de la dérive en fréquence en télévision numérique

THÈSE

présentée et soutenue publiquement le 28 juin 2006

pour l'obtention du

Doctorat de l'université Henri Poincaré – Nancy 1

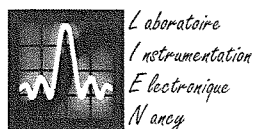
(spécialité Instrumentation et Micro-Électronique)

par

Christian MANNINO

Composition du jury

<i>Rapporteurs :</i>	M. Patrick GARDA	Professeur, Université Pierre et Marie Curie, Paris
	M. Marc GAZALET	Professeur, Université de Valenciennes et du Hainaut Cambrésis
<i>Examineurs :</i>	M. Christophe BOBDA	Professeur, Université de Kaiserslautern, Kaiserslautern
	M. Serge WEBER	Professeur, U.H.P., Nancy I
	M. Hassan RABAH	Maître de Conférence, U.H.P., Nancy I
<i>Invité :</i>	M. Arnaud MAHÉ	Responsable projets, THALES B&M, Cesson-Sevigne



Laboratoire d'Instrumentation Électronique de Nancy
Faculté des Sciences - 54506 Vandoeuvre-lès-Nancy

Remerciements

J'adresse mes sincères remerciements à Monsieur Patrick GARDA, Professeur à l'Université Pierre et Marie Curie ainsi qu'à Monsieur Marc GAZALET, Professeur à l'Université de Valenciennes et du Hainaut Cambrésis, qui m'ont fait l'honneur de juger cette thèse en qualité de rapporteurs.

J'adresse aussi mes remerciements à Monsieur Christophe BOBDA, Professeur à l'Université de Kaiserslautern d'avoir accepté d'examiner ce travail et de participer au jury, ainsi qu'à Monsieur Arnaud MAHÉ, responsable projets au sein du groupe THALES B&M à Cesson-Sevigne, d'être présent parmi les membres du jury.

Je tiens à remercier Monsieur Serge WEBER, Professeur à l'Université de Nancy et mon directeur de thèse, de m'avoir accueilli au sein de son équipe et d'avoir pu travailler sur un sujet très intéressant.

Je souhaite remercier particulièrement Monsieur Hassan RABAH de m'avoir encadré durant cette thèse, ainsi que pour sa qualité scientifique, sa disponibilité et son soutien.

Je tiens à remercier tous les membres du laboratoire et tous les membres du consortium EQUAST que j'ai pu côtoyer durant mes trois années de thèse. Je souhaite aussi remercier Monsieur Camel TANOUGAST, Monsieur Yves BERVILLER et Monsieur Michael JANIAUT de leurs soutiens, de leurs conseils et de leurs sens du détail qui ont donné lieu à tant de discussions constructives.

Christian MANNINO
Avril 2006

*Je dédie cette thèse à mes parents :
Marcelle et Alexandre.*

Résumé

La nature asynchrone du flux transport MPEG-2 en télévision numérique nécessite la transmission d'informations de l'horloge de l'encodeur. Ces informations se présentent sous forme de données numériques appelées "program clock reference" (PCR) et insérées dans les paquets de transport. Le décodeur au niveau du récepteur utilise ces informations en association avec une boucle à verrouillage de phase pour reconstruire l'horloge et se synchroniser ainsi avec l'encodeur. Lors de leur transmission, les PCRs sont soumis à des perturbations telles que par exemple les variations de délai dues au réseau. Ces perturbations induisent notamment de la gigue et différentes dérives qui peuvent avoir un effet significatif sur la qualité de la réception. La mesure de ces paramètres fait également souvent appel à une PLL. Cette fonction est généralement constituée de composants analogiques qui présentent à la fois une reproductibilité délicate et une contrainte pour la technologie d'implantation.

Le travail de thèse présenté porte sur l'étude et la caractérisation d'une fonction PLL numérique associée à un ensemble de fonctions de mesure. Sa particularité est de permettre une implantation sur des technologies numériques y compris FPGA. Une étude portant sur l'état de l'art des techniques de synchronisation, et sur les techniques de génération d'horloge les plus utilisées, met en évidence les limites des systèmes de synchronisation actuels. La solution proposée utilise une boucle à verrouillage de phase totalement numérique (ADPLL). Cela est rendu possible grâce au remplacement de l'oscillateur contrôlé en tension par un synthétiseur de fréquence original basé sur la génération d'un signal de fréquence centrale, subissant un retard ou une avance de phase calibrés. L'architecture de synchronisation proposée est modélisée pour déterminer la gigue intrinsèque au synthétiseur et le comportement de l'ADPLL en terme de stabilité et de bande passante. La solution est utilisée dans un système de mesure des paramètres de gigue, erreur de fréquence centrale, et dérive pour l'évaluation de la qualité de service en télévision numérique terrestre (DVB-T). La solution, bien que travaillant à une fréquence centrale de 27 MHz, est totalement compatible avec les technologies FPGA. Les performances principales obtenues sont une gigue intrinsèque faible (100 ps), l'indépendance de la PLL du temps d'arrivée des PCRs, la portabilité de l'architecture et la possibilité d'intégration de plusieurs PLL dans un FPGA.

Mots clés : ADPLL, FPGA, mesures relatives au PCR, synthétiseur de fréquence, exploration de l'espace de conception, qualité de service, MPEG-2, DVB.

Abstract

The asynchronous nature of MPEG-2 transport stream in digital video broadcasting requires the transmission of the encoder's clock informations. These informations also called program clock reference "PCR" are numerical stamps inserted in the transport packets. In the receiver part, the decoder, utilises theses informations as input data of a phase locked loop (PLL) for clock recovery and synchronisation. During the transmission, the PCRs are submitted to several perturbations such as the delay variation due to the network. These perturbations produce a PCR jitter, frequency offsets and drifts that can affect significantly the quality of reception. The measurement of these parameters can be performed using a PLL system. This system, is usually composed of analog components. The drawbacks of this solution are reproducibility and the hardware implementation requiring fine and individual tuning.

The work of the thesis presented relates to the study and the characterization of a numerical PLL function associated to a set of measurement functions. The particularity is to allow its implementation on digital technologies including FPGA. The designed function allows the clock recovery by synchronisation on time stamps. A study relating on the state of the art of synchronisation techniques, and on the most used clock generation techniques of the most used generation of clock, highlights the limits of the current synchronization systems. The proposed solution uses an All-Digital Phase Locked Loop (ADPLL). That is made possible thanks to the replacement of voltage controlled oscillator by an original frequency synthesizer based on the generation of a signal at a central frequency, undergoing a delay or an advance of phase. The synchronization architecture proposed is modeled to determine the intrinsic jitter of the synthesizer and the behavior of the ADPLL in term of stability and bandwidth. The solution is used in measurement system of jitter, frequency error and drift parameter for the quality of service evaluation in terrestrial digital television (DVB-T). The solution, although working at a central frequency of 27 MHz , is totally compatible with FPGA technologies. The main performances obtained are a low intrinsic jitter (100 ps), PCR arrival time independent, architecture portability and multiple PLL implementation in a single FPGA circuit.

Keywords : ADPLL, FPGA, PCR related measurements, frequency synthesizer, design space exploration, quality of service, MPEG-2, DVB.

Table des matières

Table des figures	1
Glossaire	5
Introduction générale	7
1 Contexte et problématique	7
2 Organisation du manuscrit	12
Chapitre 1 État de l’art	
1.1 Introduction	13
1.2 Définition de la gigue	14
1.3 Les techniques de synchronisation par horodateurs	14
1.3.1 Synchronisation par PLL	16
1.3.1.1 Principe général d’une PLL	16
1.3.1.2 Les différents types de PLL	17
1.3.1.3 PLLs utilisées pour la reconstruction d’horloge par PCR	17
1.3.2 Technique de synchronisation dans les SoCs submicroniques	18
1.3.3 Synchronisation par une méthode de suppression de gigue (dejittering)	20
1.3.4 Synchronisation par des méthodes statistiques	22
1.4 Les techniques de génération d’horloge	24
1.4.1 Introduction	24
1.4.2 Génération d’horloge à l’aide d’oscillateur en anneau	24
1.4.2.1 Principe général	24
1.4.2.2 Contrôle de fréquence par charge variable	26
1.4.2.3 Contrôle de la fréquence par résistance variable	27
1.4.2.4 Contrôle de la fréquence par modification de la polarisation	29
1.4.2.5 Contrôle de la fréquence par sélection de chemin	30
1.4.2.6 Technique de minimisation de la gigue due à l’alimentation	36

1.4.3	Génération d'horloge par synthèse de fréquences	37
1.4.3.1	Architecture de division fractionnaire par N	37
1.4.3.2	Architecture PSDCO	37
1.4.3.3	Architecture de type "Flying Adder"	40
1.5	Conclusion	42

Chapitre 2 Description du système de mesure des paramètres relatifs au PCR utilisant une ADPLL

2.1	Introduction	45
2.2	Principe du synthétiseur de fréquence proposé	46
2.2.1	Principe de fonctionnement et modélisation	46
2.2.2	Gigue intrinsèque du synthétiseur	49
2.3	Le filtre de boucle	51
2.3.1	Schéma intermédiaire de l'ADPLL	51
2.3.2	Conditions de phase et de fréquence sur l'ADPLL : détermination de la structure du correcteur	51
2.4	Etude de la stabilité de l'ADPLL	53
2.4.1	Expression générale de la fonction de transfert	53
2.4.2	Les conditions de stabilité	55
2.5	Etude comportementale de l'ADPLL	56
2.6	Adaptation de la PLL aux temps d'arrivée des PCRs	60
2.6.1	Origine de la variation du temps d'arrivée des PCRs et de la modification du comportement dynamique de l'ADPLL	60
2.6.2	Amélioration de la stabilité de l'ADPLL par normalisation de l'erreur de PCR	61
2.6.3	Obtention d'une bande passante constante de l'ADPLL par une technique de sur-échantillonnage	62
2.7	Extraction des paramètres relatifs au PCR	64
2.7.1	Les filtres de mesure	64
2.7.2	La mesure des paramètres	65
2.7.3	Génération de dérives virtuelles en fréquence en vue du test	65
2.8	Conclusion	66

Chapitre 3 Implantation et caractérisation du système de mesure

3.1	Méthodologie	69
-----	------------------------	----

3.1.1	Modélisation architecturale de haut niveau	69
3.1.2	Dispositif expérimental	71
3.2	Implantation et caractérisation de l'ADPLL	71
3.2.1	Validation des contraintes de temps du synthétiseur de fréquence	71
3.2.2	Caractérisation des ressources matérielles de l'ADPLL	72
3.2.3	Caractérisation de la gigue du synthétiseur	72
3.2.4	Etude comportementale de l'ADPLL	75
3.2.5	Etude comportementale de l'ADPLL avec normalisation de l'erreur de PCR	80
3.2.6	Etude comportementale de l'ADPLL associée à une technique de sur- échantillonnage	81
3.3	Implantation et caractérisation des filtres de mesure	82
3.3.1	Ressources d'implantation des filtres de mesure	82
3.3.2	Temps d'exécution des filtres	84
3.3.3	Résultats de mesure des paramètres relatifs au PCR	84
3.3.4	Validation de la mesure des paramètres relatifs au PCR par le partenaire TDF	88
3.3.4.1	Le paramètre PCR_FO	89
3.3.4.2	Le paramètre PCR_DR	89
3.3.4.3	Le paramètre PCR_AC	90
3.4	Conclusion	90

Chapitre 4 Optimisation des ressources du système de mesure

4.1	Introduction	93
4.2	Optimisation pour l'implantation d'un système de mesure 10 PCR's	93
4.2.1	Implantation parallèle	94
4.2.2	ADPLLs avec réutilisation d'opérateurs matériels	95
4.2.3	Implantation séquentielle	95
4.3	Estimation des ressources matérielles et logicielles	96
4.3.1	Ressources d'une implantation parallèle	96
4.3.2	Ressources ADPLLs avec réutilisation d'opérateurs	97
4.3.3	Ressources ADPLLs utilisant un processeur	97
4.3.4	Temps d'exécution	98
4.3.4.1	Cas du NIOS inséré dans la PLL	98
4.3.4.2	Cas du NIOS traitant les PCR's	99
4.4	Discussion	99
4.5	Conclusion	100

Conclusion

Annexe A Démonstration de la fonction de transfert du synthétiseur de fréquence 105

Annexe B Démonstration de la stabilité de l'ADPLL 107

Annexe C Les filtres de mesure des paramètres relatifs au PCR 111

Annexe D Description de la plate-forme EXCALIBUR 117

Bibliographie 119

Table des figures

1	Trame du flux DVB.	7
2	Modélisation de la génération de PCRs, de leur transmission et des paramètres relatifs au PCR.	9
3	Schéma bloc du système de mesure.	10
4	Schéma bloc de l'ADPLL.	11
1.1	Principe d'une transmission audio-visuelle.	15
1.2	La réception audio-visuelle.	15
1.3	Schéma bloc d'une PLL.	16
1.4	Schéma d'une PLL synchronisée sur des horodateurs.	18
1.5	Technique de re-synchronisation des données.	19
1.6	Proposition d'adaption à la synchronisation sur des PCRs.	19
1.7	Principe général du système de suppression de gigue.	20
1.8	Schéma du système de suppression de gigue.	21
1.9	Modèle du système de suppression de gigue.	21
1.10	Principe de fonctionnement de la LLR.	23
1.11	Architecture de la LLR.	24
1.12	Cellule de délais à inverseur CMOS (a), oscillateur en anneau à trois étages d'inverseurs (b).	25
1.13	Oscillateur en anneau à quatre étages d'inverseurs pour fournir des signaux en quadrature.	25
1.14	Modification du délai par variation de charge	26
1.15	Modification du délai par contrôle numérique de la charge	26
1.16	Modification du délai par résistance variable	27
1.17	Modification du délai par résistance variable (variante)	28
1.18	Evolution du délai en fonction de l'entrée	28
1.19	Modification du délai par contrôle de polarisation	29

1.20 DCO utilisant des inverseurs trois états en parallèle pour l'ajustement de la fréquence.	30
1.21 Variation du délai en fonction de l'entrée.	31
1.22 DCO utilisant une matrice de délai.	32
1.23 DCO portable.	32
1.24 DCO portable disposant d'une résolution fréquentielle élevée.	33
1.25 Schéma détaillé de l'oscillateur en anneau portable.	33
1.26 ADPLL utilisant un DCO.	34
1.27 Gigue relative de l'horloge de sortie de l'ADPLL. Pour $f_{out} = 20\text{ MHz}$ (a), pour $f_{out} = 52\text{ MHz}$ (b).	35
1.28 Oscillateur en anneau à trois étages avec circuit de polarisation (a), simple cellule de délais (b), cellule de délais avec fonction de décalage de phase (c).	36
1.29 Principe de la division fractionnaire par N.	37
1.30 Structure de la division fractionnaire par N.	38
1.31 Architecture simplifiée du synthétiseur de fréquence.	38
1.32 Coeur du synthétiseur de fréquence.	39
1.33 PLL utilisant un VCO générant des horloges polyphases, utilisées pour le synthétiseur de fréquence "Flying Adder".	40
1.34 Architecture du synthétiseur de fréquence "Flying Adder".	41
2.1 Décalage de $\pm\delta T$ d'un signal de fréquence $1/T_{nom}$	46
2.2 Erreur de phase instantanée sur le PCR pendant une durée T_{PCR}	47
2.3 Schéma bloc du synthétiseur de fréquence.	48
2.4 Schéma de la FSM lorsque une horloge de référence $m.F_{nom}$ est utilisée.	49
2.5 Fonction de transfert du synthétiseur de fréquence.	49
2.6 Gigue en fonction de la fréquence pour $m=\{4;8;12\}$	50
2.7 Schéma intermédiaire de l'ADPLL.	51
2.8 Schéma du correcteur PI de l'ADPLL.	54
2.9 Pôles et zéros, module et argument de la fonction de transfert de l'ADPLL pour $T_{PCR} = \{10ms; 40ms; 100ms\}$	57
2.10 Réponses de l'ADPLL à un échelon de fréquence pour $T_{PCR} = \{10ms; 40ms; 100ms\}$	58
2.11 Réponses de l'ADPLL standard à un échelon de fréquence pour $T_{PCR} = 40\text{ ms}$ et $G_1 = \{20; 30; 40; 60\}$	58
2.12 Réponses de l'ADPLL "standard" à un échelon de fréquence pour $T_{PCR} = 40\text{ ms}$ et $G_2 = \{5; 1; 0, 2\}$	59
2.13 Réponses de l'ADPLL à un échelon de fréquence et courbes de gain et de phase pour $T_{PCR} = 40\text{ ms}$, $G_1 = 10$ et $G_2 = 0,001$	59

2.14 Réponses de l'ADPLL à un échelon de fréquence et courbes de gain et de phase pour $T_{PCR} = 40\text{ ms}$, $G_1 = 2$ et $G_2 = 0,0001$	60
2.15 Erreur de PCR en fonction du temps.	61
2.16 Schéma de la fonction réalisant la normalisation du PCR.	62
2.17 Chronogramme illustrant le principe du sur-échantillonnage appliqué au PCR. PCR _s reçus avec son signal de validation (a), interpolation des PCR _s et horloge de sur-échantillonnage associée (b).	63
2.18 Schéma de la structure réalisant l'interpolation du PCR.	64
2.19 Structure permettant de mesurer le paramètre PCR_{AC}	65
2.20 Générateur de PCR fournissant une dérive virtuelle.	67
3.1 Architecture de mesure pour un programme.	70
3.2 Histogramme de gigue à long terme du quartz de $33,333\text{ MHz}$	73
3.3 Histogramme de gigue à long terme du synthétiseur de fréquence fournissant $F_{nom} + 800\text{ Hz}$	73
3.4 Histogramme de gigue au long terme du synthétiseur de fréquence générant F_{nom}	74
3.5 Gigue intrinsèque au synthétiseur en fonction de la fréquence autour de F_{nom}	74
3.6 Comportement de l'ADPLL pour différentes valeurs de T_{PCR} . $T_{PCR} = 10\text{ ms}$ (a), $T_{PCR} = 40\text{ ms}$ (b), $T_{PCR} = 100\text{ ms}$ (c).	76
3.7 Comportement de l'ADPLL pour $G_1 = \{20, 30, 40, 60\}$	77
3.8 Filtre de boucle implanté pour $G_1 = 10$ et $G_2 = 0,001$	77
3.9 Comportement de l'ADPLL pour $G_1 = 10$ et $G_2 = 0,001$	78
3.10 Comportement de l'ADPLL pour une période d'échantillonnage variable. Fréquence de l'ADPLL autour de F_{nom} (a), durée de T_{PCR} (b), erreur de fréquence par rapport à l'entrée (c), erreur de PCR (d).	78
3.11 Comportement de l'ADPLL lorsque la fréquence d'entrée dérive. Fréquence de l'ADPLL autour de F_{nom} (a), erreur de fréquence par rapport à l'entrée (b), erreur de PCR (c).	79
3.12 Erreur de PCR relative moyenne PCR_{error} en fonction de la dérive de la fréquence $\left(\frac{dF_e}{dt}\right)$	80
3.13 Comportement de l'ADPLL avec normalisation de l'erreur de PCR pour plusieurs périodes d'échantillonnage. $T_{PCR} = 20\text{ ms}$ (a), $T_{PCR} = 40\text{ ms}$ (b).	81
3.14 Comportement de l'ADPLL avec des PCR _s interpolés pour plusieurs périodes d'échantillonnage. Réponse indicielle "brute" pour $T_{PCR} = 20\text{ ms}$ (a), comparaison entre théorie et réponse indicielle expérimentale moyennée pour $T_{PCR} = 20\text{ ms}$ (b), réponse indicielle "brute" pour $T_{PCR} = 40\text{ ms}$ (c), comparaison entre théorie et réponse indicielle expérimentale moyennée pour $T_{PCR} = 40\text{ ms}$ (d).	83

3.15 Interfaces du NIOS pour un programme.	83
3.16 Mesure du paramètre PCR_{FO} pour une fréquence d'entrée à $F_{nom} + 800Hz$. . .	85
3.17 Mesure des paramètres PCR pour une fréquence d'entrée à $F_{nom} + 800Hz$ et pour un saut sur la période d'échantillonnage. PCR_{FO} (a), T_{PCR} (b), PCR_{DR} (c), PCR_{OJ} (d), PCR_{AC} avant normalisation (e), PCR_{AC} après normalisation (f).	86
3.18 Mesure des paramètres relatifs au PCR pour une fréquence d'entrée dérivant à $64 Hz/s$. $F_e - F_r$ (a), PCR_{FO} (b), T_{PCR} (c), PCR_{DR} (d), PCR_{OJ} (e) et PCR_{AC} (f).	87
3.19 Mesure des paramètres relatifs au PCR pour une fréquence d'entrée dérivant à $15 mHz/s$. PCR_{FO} (a), PCR_{DR} (b), PCR_{OJ} (c) et PCR_{AC} (d).	88
3.20 Allure du PCR_{FO} et du PCR_{DR}	90
4.1 Optimisation avec réutilisation d'opérateurs.	94
4.2 Optimisation avec un processeur NIOS.	96
4.3 Temps d'exécution du traitement de la boucle de l'ADPLL en fonction des ressources de l'architecture globale.	100
C.1 Structure du filtre passe-bas numérique du premier ordre.	112
C.2 Structure du filtre passe-haut numérique du premier ordre.	114
C.3 Structure du filtre passe-haut numérique du second ordre.	115
D.1 Vue générale de la plate-forme EXCALIBUR utilisant un APEX EP20K200EFC484-2x.	118

Glossaire

ADPLL	All-Digital Phase Locked Loop
APLL	Analog Phase Locked Loop
ASIC	Application Specific Integrated Circuit
ATM	Asynchronous Transfer Mode
CAN	Convertisseur Numérique Analogique
CBR	Constant Bit Rate
CMOS	Complementary Metal Oxide Semiconductor
DCO	Digitally Controlled Oscillator
DDS	Direct Digital Synthesis
DPLL	Digital Phase Locked Loop
DLL	Delay Phase Locked Loop
DSP	Digital Signal Processor
DTS	Decoding Time Stamps
DVB	Digital Video Broadcasting
DVB-T	Digital Video Broadcasting - Terrestrial
ETSI	European Telecommunications Standards Institute
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
IP	Internet Protocol ou Intellectual Property
LC	Logic Cell
LLR	Least-square Linear Regression
LPLL	Linear Phase Locked Loop
MOS	Metal Oxide Semiconductor
MPEG	Moving Picture Expert Group
NCO	Numerically Controlled Oscillator
PCR	Program Clock Reference
PI	Proportionnel Integral

PID	Proportionnel Integral Dérivée
PLL	Phase Locked Loop
PTS	Presentation Time Stamps
QdS	Qualité de Service
RMS	Root Mean Square
RTP	Real Time Protocol
SOPC	System On a Programmable Chip
STC	System Time Clock
TCP	Transmission Control Protocol
TS	Transport Stream
VBR	Variable Bit Rate
VCO	Voltage Controlled Oscillator
VHDL	VHSIC (Very High Speed Integrated Circuit) Hardware Description Language

Introduction générale

1 Contexte et problématique

En DVB-T, le flux de transport (figure 1) est transmis avec un débit de 27 Mbit/s et contient des données audio-vidéo au format MPEG-2. Les données MPEG-2 sont issues d'un encodeur qui multiplexe les données audio-vidéo dans des trames de 188 octets. Ces trames passent ensuite dans un canal de transmission dont le support est l'onde hertziennne. Ainsi, les données sont transmises sur ce réseau en mode asynchrone.

Du côté du récepteur DVB-T, le décodeur MPEG-2 est synchronisé par une horloge en phase avec celle de l'encodeur. Cela est nécessaire car les données audio-vidéo sont échantillonnées par le décodeur. De plus, chaque encodeur n'a pas la même horloge de référence, ce qui signifie que lors du passage d'un programme à l'autre, l'horloge du décodeur doit être réajustée. Ainsi, pour garantir la décodabilité des programmes, le décodeur nécessite une horloge synchronisée sur l'encodeur qui a généré les données du flux MPEG-2.

Pour permettre la synchronisation, le flux contient, en plus des données audio-vidéo, des informations relatives à l'horloge qui a encodé le son et la vidéo. Ces données, qui sont des valeurs numériques, sont des étiquettes temporelles, encore appelées horodateurs (time stamps) ou Program Clock References (PCR) [ETS01]. Ces PCRs sont propres à chaque programme du flux et sont générés par l'encodeur MPEG-2 en utilisant un compteur, incrémenté par son horloge locale, et échantillonné pour transmettre les informations de cette horloge dans le flux. Les PCRs sont des données transmises dans l'en-tête des trames de ce flux et permettent de restituer l'horloge locale à 27 MHz d'un décodeur MPEG-2 se trouvant dans un récepteur vidéo

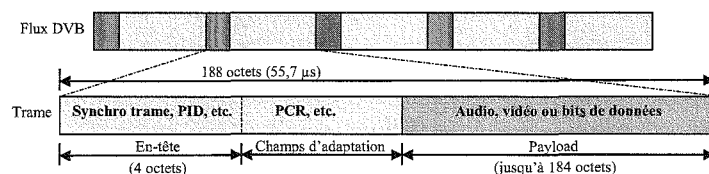


FIG. 1 – Trame du flux DVB.

numérique DVB.

En DVB, comme le réseau de transmission est en mode asynchrone, il est exposé à certains effets qui sont causés par les éléments du réseau. Un des effets prédominants est l'acquisition de gigue dans le réseau affectant la position d'horodateurs contenus dans le flux, ce qui entraîne des erreurs relatives à ces horodateurs. Ces erreurs peuvent ainsi altérer la synchronisation de l'horloge MPEG-2, et donc affecter la qualité de service des programmes diffusés.

Les dérives des PCRs doivent être détectées avant que le flux de transport arrive au récepteur DVB-T. Cette détection peut être effectuée au niveau des multiplexeurs de réseau DVB intelligent, qui correspondent à un exemple d'application de l'analyse de dérives de PCRs. Ce type de multiplexeur reçoit deux flux de transport contenant les mêmes programmes, mais de sources différentes, et sélectionne le flux qui peut fournir les programmes avec la meilleure qualité à l'utilisateur final, ceci en temps réel pour ne pas dégrader la qualité de réception. C'est pourquoi un multiplexeur DVB doit analyser en temps réel les paramètres relatifs au PCR pour tous les programmes d'un flux.

Ce système peut aussi être utilisé comme système de vérification des PCRs du flux avant l'étape d'émission, afin de garantir que les trames ne contiennent pas de valeurs erronées de PCR pour chaque programme, et c'est pourquoi, là aussi, le système doit analyser les paramètres relatifs au PCR en temps réel. Ces mesures sont réalisées en utilisant un système de synchronisation permettant de restituer localement l'horloge système MPEG-2. Ce système de synchronisation, associé à une unité de traitement, fournira les paramètres relatifs au PCR. Plus précisément, ces paramètres mesurent l'offset de fréquence de l'horloge locale de l'encodeur, sa dérive dans le temps, la gigue relative au PCR introduite par les erreurs de PCR créées lors de sa génération et par la gigue de réseau.

Dans le flux DVB, lorsque deux trames successives contiennent des PCRs de programmes différents, ce qui correspond au cas critique, le temps nécessaire à la prise en compte de ces PCRs doit être inférieur à $55,7 \mu s$. De plus, la fréquence d'arrivée des PCRs, qui est de $1/T_{PCR}$, varie de $10 Hz$ à $100 Hz$. Ainsi, les paramètres relatifs au PCR de la norme devront être traités en moins de $10 ms$, pour un programme, pour que les mesures soient effectuées en temps réel.

La figure 2 donne le modèle de l'horloge système de l'encodeur MPEG-2 et des paramètres de mesure relatifs au PCR spécifiques à un programme $PID_{(p)}$. Un encodeur/multiplexeur est modélisé jusqu'au point B et le mécanisme physique de transmission, ou encore appelé le réseau de communication, du flux TS (Transport Stream) est modélisé entre les points B et C. L'équipement de mesure peut habituellement accéder au flux TS au point C. Le modèle est constitué d'une horloge système provenant d'un oscillateur de fréquence $27 MHz$, dont la fréquence instantanée dévie par une fonction $fdev_{(p,t)}$. Cette fonction dépend du temps t et du PCR $PID_{(p)}$. Le paramètre d'offset de fréquence PCR_FO (PCR Frequency Offset) mesure la valeur de $fdev_{(p,t)}$.

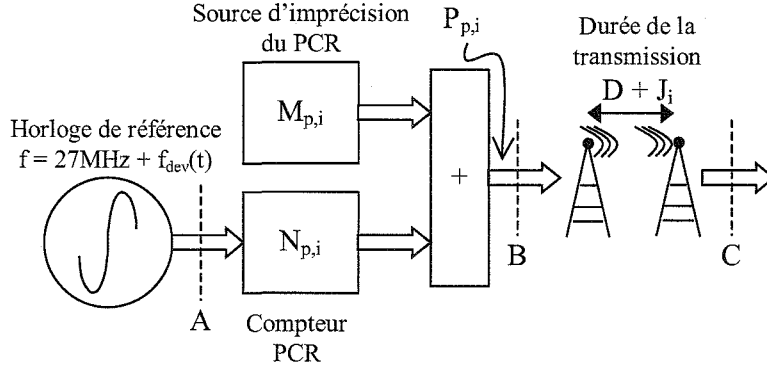


FIG. 2 – Modélisation de la génération de PCRs, de leur transmission et des paramètres relatifs au PCR.

Le paramètre PCR_DR mesure la dérive en fréquence de $f_{dev}(p,t)$. L'horloge système commande un compteur PCR qui génère un PCR idéal $N_{(p,i)}$. p fait référence à un PCR propre à un $PID_{(p)}$ et i fait référence à la position du bit du flux TS. Au terme $N_{(p,i)}$ est ajouté une source d'imprécision $M_{(p,i)}$ du PCR pour former la valeur réelle du PCR du flux TS, $PCRe_{(p,i)}$. La relation entre ces valeurs est donnée par l'équation 1 :

$$PCRe_{(p,i)} = N_{(p,i)} + M_{(p,i)} \quad (1)$$

$M_{(p,i)}$ représente le paramètre PCR_AC donnant la précision sur le PCR. Le réseau de communication se situe au delà du point B et introduit un délai variable entre l'instant de départ T_i et l'instant d'arrivée U_i d'un bit i .

$$U_i - T_i = D + J_i \quad (2)$$

Dans le cas du PCR, U_i est l'instant d'arrivée du dernier bit du dernier octet contenant la base du PCR [II94]. D est une constante représentant le délai moyen entre l'entrée et la sortie du réseau de communication et J_i représente la gigue de réseau dont la valeur moyenne est définie comme nulle. $J_i + M_{(p,i)}$ correspond à la gigue globale, mesurée par le paramètre PCR_OJ (PCR Overall Jitter).

D'après le standard de mesure en DVB [ETS01], les paramètres relatifs au PCR doivent être compris dans les plages de valeur à ne pas dépasser pour être tolérable. Ces plages de valeur sont les suivantes :

$$PCR_FO : \pm 810\text{Hz} (@27\text{MHz}) \quad (3)$$

$$PCR_DR : \pm 75\text{mHz/s} (@27\text{MHz}) \quad (4)$$

$$PCR_OJ : \pm 500\text{ns si gigue de rseau} = 0 \quad (5)$$

$$PCR_AC : \pm 500\text{ns} \quad (6)$$

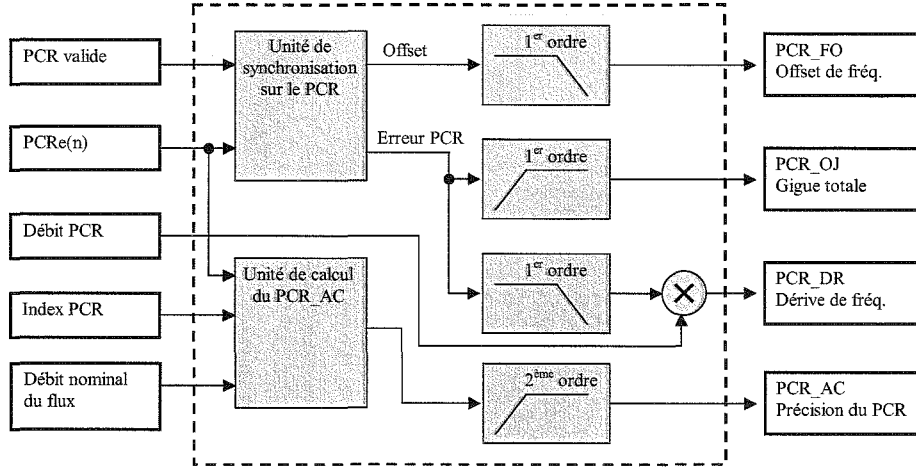


FIG. 3 – Schéma bloc du système de mesure.

Le schéma bloc du système de mesure est donné en figure 3. Celui-ci est composé d'une unité de synchronisation sur le PCR délivrant les informations de gigue de PCR. L'unité de calcul du paramètre mesurant la précision du PCR (PCR_{AC}) permet de calculer l'erreur brute due à l'imprécision sur le PCR et des filtres de mesure sont requis pour extraire les paramètres relatifs au PCR.

L'unité de synchronisation du système de mesure utilise des ADPLLs qui doivent régénérer les horloges systèmes des encodeurs MPEG-2 de chaque programme du multiplexe à l'aide de PCRs [MRT+04a, MRT+04b]. L'étude portera sur une seule ADPLL, car elle est représentative de toutes les autres. Le PCR reçu est noté $PCR_e(n)$. Pour restituer l'horloge d'un encodeur MPEG-2, une PLL au niveau du décodeur génère des données horodatées $PCR_r(n)$. Les $PCR_r(n)$ dépendent du temps d'arrivée des $PCR_e(n)$ et de la fréquence $F_r(n)$ de l'horloge MPEG-2 restituée, comme le montre l'équation 7. Le $PCR_e(n)$ et le $PCR_r(n)$ sont comparés et la synchronisation est réalisée lorsque la différence est nulle.

$$PCR_r(n) = PCR_r(n-1) + T_{PCR} \times F_r(n-1) \quad (7)$$

Le schéma bloc de l'ADPLL est donné en figure 4. Il est composé d'un comparateur de PCR qui calcule l'erreur entre $PCR_e(n)$ et $PCR_r(n)$. Le filtre de boucle, qui contrôle la fréquence de l'horloge fournie par le synthétiseur de fréquence, permet d'annuler l'erreur de PCR suite à un échelon de phase et de fréquence et d'agir sur le comportement dynamique de la PLL à l'aide de coefficients. Le synthétiseur de fréquence délivre un signal d'horloge avec une fréquence moyenne dépendant des variations lentes des erreurs filtrées. Le compteur PCR produit les $PCRs$ locaux estimés.

Cette structure permet d'extraire plusieurs informations pour caractériser la QdS liée au

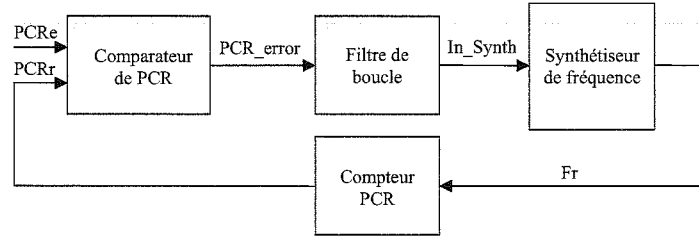


FIG. 4 – Schéma bloc de l'ADPLL.

PCR. L'information d'offset de fréquence de l'horloge système MPEG-2 sera prélevée sur l'entrée du synthétiseur *In_Synth* et les informations sur la dérive et sur la gigue de PCR seront relevées sur la sortie *PCR_error* du comparateur de PCR en utilisant des filtres de mesure adaptés.

Ainsi, l'objectif de cette thèse est de proposer une architecture dédiée à la mesure de paramètres relatifs au PCR de la norme ETSI [ETS01], qui devra analyser l'ensemble des programmes présents dans un flux numérique terrestre. Cette architecture nécessitera des systèmes de synchronisation pour restituer l'horloge locale MPEG-2 de chaque programme. Cette horloge est nécessaire pour avoir une base de comparaison avec l'horloge qui a générée les PCRs du flux. Plusieurs techniques de restitution d'horloge existent comme celles basées sur le traitement statistique du signal ou sur l'utilisation de boucles à verrouillage de phase (PLL : Phase Locked Loop). Elles peuvent être implémentées en utilisant des éléments analogiques, numériques ou mixtes. Si des éléments analogiques sont utilisés, comme un oscillateur contrôlé par tension, il sera nécessaire d'en implémenter autant qu'il y a de programmes dans le flux. Ceci est une solution coûteuse en terme de ressources matérielles car elles occupent beaucoup de surface. Une solution numérique à base de FPGA permet une plus grande flexibilité, car elle permet par exemple, après une première implantation, de pouvoir modifier les coefficients d'un filtre afin d'ajuster par exemple le comportement du dispositif de synchronisation. Cette flexibilité permet aussi d'ajouter des éléments dans le même circuit si le nombre de programmes transmis dans le flux numérique terrestre venait à être modifié dans l'avenir, ce qui ne serait pas possible avec des éléments analogiques. Dans ce sens, la difficulté ne réside pas dans la recherche d'une technique permettant de mesurer la gigue, mais de trouver une architecture numérique, à base de circuit FPGA pour la flexibilité, intégrant autant d'unités de synchronisation qu'il y a de programmes dans le flux en prenant en compte le coût des ressources matérielles. La difficulté se porte donc sur le choix d'une unité de synchronisation totalement numérique qui doit répondre aux exigences.

Nous proposons ainsi dans cette thèse une architecture totalement numérique optimisée en terme de ressources matérielles et satisfaisant les contraintes de surface et de temps. Cette architecture utilise une PLL totalement numérique qui sera dupliquée autant de fois qu'il y a de

programmes dans le flux, en étudiant plusieurs solutions d'optimisation de ressources matérielles. Cette implantation est rendue possible grâce à l'utilisation d'un synthétiseur de fréquence innovant, qui avance ou retarde la phase d'une horloge à des instants précis. Des filtres de mesure associés à ces PLLs, permettent d'extraire les paramètres relatifs au PCR.

2 Organisation du manuscrit

Cette thèse est constituée de quatre chapitres et est organisée de la manière suivante : le chapitre 1 est un état de l'art des techniques de synchronisation dédiées, ou qui peuvent être réadaptées, à la synchronisation sur des PCRs et concernera aussi les techniques de génération d'horloge les plus utilisées dans ces unités de synchronisation qui requièrent une horloge locale. Une première conclusion de synthèse de l'état de l'art présentera les atouts et les manques des techniques existantes par rapport aux attentes du système à mettre en œuvre.

Le chapitre 2 présentera ensuite la solution architecturale proposée et définira les contraintes du système de mesure qui intégrera les systèmes de synchronisation de chaque programme du flux. L'origine et le modèle des différents paramètres à mesurer seront donnés. Puis, une unité de synchronisation sera modélisée, sachant qu'elle est représentative de toutes les autres. Ce modèle permettra d'étudier son comportement en régime dynamique et en terme de gigue. Ainsi, après avoir analysé sa stabilité, sa bande passante et sa réponse indicielle sont étudiées dans différents cas de figures. Des structures de filtre sont ensuite proposées qui vont permettre d'extraire les paramètres de mesure relatifs au PCR.

Le chapitre 3 fera l'objet de la validation du système de mesure. Dans un premier temps, l'architecture est décrite de manière hiérarchique avec SystemC. Un découpage matériel/logiciel est ensuite effectué, où la PLL numérique est implantée matériellement car elle utilise des éléments fonctionnant à de fortes contraintes de temps, et en implémentant les autres fonctions logiciellement, car elles nécessitent l'utilisation d'un format flottant. L'unité de synchronisation sera ensuite étudiée expérimentalement afin de vérifier la conformité avec le modèle théorique. Puis, le système de mesure des paramètres relatifs au PCR est caractérisé en terme de temps de réponse, de résolution et de précision.

Le chapitre 4 concernera l'exploration de l'espace de conception permettant de trouver des solutions d'implantation du système de mesure complet analysant l'ensemble des paramètres relatifs au PCR de chaque programme constituant un flux numérique DVB-T. Une solution consiste à implanter toutes les PLLs en parallèle, mais cela entraîne une consommation importante de ressources matérielles. Pour diminuer les ressources consommées tout en respectant les contraintes de temps, certains opérateurs matériels peuvent être réutilisés ou remplacés par une implémentation logicielle.

Enfin, une conclusion permettra de faire un bilan de l'étude menée au cours de cette thèse.

Chapitre 1

État de l'art

1.1 Introduction

L'introduction générale a présenté une vue générale de l'architecture de mesure proposée. La difficulté n'est pas de trouver une technique de mesure des paramètres relatifs au PCR, mais de trouver une solution numérique assurant la flexibilité de l'implémentation tout en intégrant autant d'unités de synchronisation qu'il y a de programmes dans le flux en prenant en compte les ressources matérielles disponibles. Le but de ce travail de recherche est donc de proposer une unité de synchronisation qui puisse répondre à ces exigences.

Pour assurer la synchronisation en DVB, plusieurs principes existent comme ceux basés sur le calcul de la pente des PCRs ou la comparaison du PCR reçu avec un PCR estimé localement. Les algorithmes de synchronisation peuvent être traités logiciellement s'il n'y a pas de contraintes sévères (fréquence, gigue, ...) sur l'horloge à reproduire, ou matériellement s'il est nécessaire de reproduire une horloge physique ayant des contraintes particulières (fréquence élevée servant d'horloge d'échantillonnage, ...)

Dans le domaine de la synchronisation, la technique la plus connue est l'utilisation d'une PLL (Phase-Locked Loop). La première PLL fut inventée par H. de Bellescize [dB32] dans les années 1930. Depuis, les PLLs ont fortement évoluées avec les avancées dans le domaine de la micro-électronique, les rendant plus performantes sur le plan de l'immunité à la gigue et augmentant leur fréquence de fonctionnement.

Cependant, d'autres techniques de synchronisation existent, qui peuvent être adaptées à la synchronisation sur des horodateurs. Nous présenterons dans ce chapitre les techniques de synchronisation, qui sont directement adaptées à la synchronisation sur des horodateurs.

L'un des points sensibles dans certains systèmes de synchronisation est la génération d'une horloge locale contrôlée, comme par exemple dans les PLLs. Le choix du système de génération d'horloge, qui a une forte incidence sur la gigue intrinsèque, permet de déterminer si l'implanta-

tion s'oriente vers une architecture hybride ou totalement numérique. Une conclusion permettra ensuite de donner leurs performances et leurs limites.

1.2 Définition de la gigue

D'une manière générale, la gigue est définie comme étant "la différence temporelle entre l'occurrence idéale et réelle d'un événement" [New02].

La gigue est constituée d'une composante déterministe et d'une composante Gaussienne (aléatoire). La composante déterministe est non Gaussienne car elle est bornée en amplitude avec une valeur crête à crête. Elle a des causes connues comme la diaphonie, les interférences électromagnétiques ou le basculement de plusieurs sorties en même temps [WAV01]. La composante aléatoire est caractérisée par une distribution Gaussienne. Elle est de nature à croître dans le temps, c'est pour cette raison que la gigue aléatoire est non bornée. Son origine provient du bruit thermique dans les semi-conducteurs, de défauts dans la régularité des dopages dans le silicium, et d'autres contributions comme les radiations cosmiques.

Dans le domaine électrique, la gigue affecte les horloges et les signaux de données. Dans le domaine des télécommunications, la gigue correspond à la variation du délai de transfert de l'information. Elle est due principalement aux délais de transferts variables dans les noeuds du réseau (switches et routeurs) et de l'air pour les transmissions par voie hertzienne. Ces perturbations du flux numérique peuvent dégrader la qualité de service des programmes fournis. Parmi les éléments du flux affectés se trouve le PCR. Ainsi, la mesure de paramètres relatifs au PCR est nécessaire pour l'évaluation de la qualité de service. Ces mesures nécessitent la reconstruction de l'horloge au niveau de la réception et donc une synchronisation du récepteur sur l'émetteur. Les techniques de synchronisation sont nombreuses, leur principe et les différentes possibilités sont décrites dans les paragraphes suivants.

1.3 Les techniques de synchronisation par horodateurs

La figure 1.1 montre le principe général de la numérisation d'informations audio-visuel. Cette figure décrit un système qui pourrait être l'une des chaînes d'émission temps réel d'un studio. Dans un premier temps, une camera fournit le signal vidéo analogique, qui peut être au format RGB. Ce signal est ensuite numérisé. La caméra génère aussi un signal de synchronisation à la fréquence f_{tx} , qui représente le signal de synchronisation ligne du signal vidéo analogique. Un circuit se synchronise sur ce signal, typiquement à l'aide d'une PLL ordinaire, comme illustré par la figure 1.1. Cette PLL produit une horloge, qui est utilisée comme horloge d'échantillonnage du convertisseur analogique numérique. Ce dernier envoie les échantillons à un encodeur MPEG-2, qui compresse le signal vidéo et l'encapsule dans des paquets de flux de transport (TS : Transport

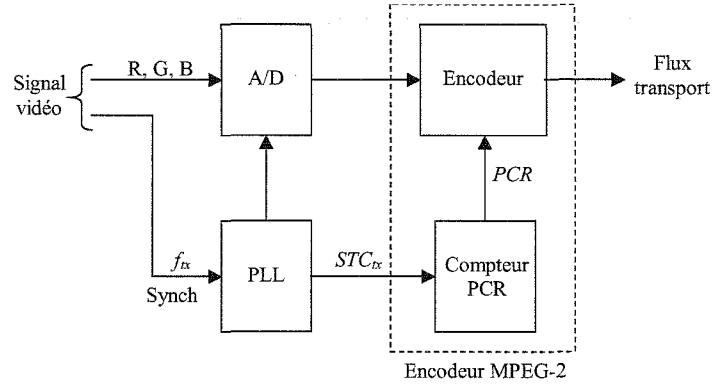


FIG. 1.1 – Principe d'une transmission audio-visuelle.

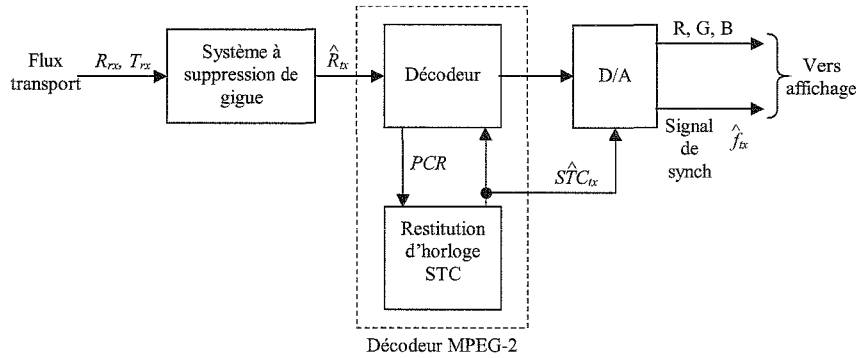


FIG. 1.2 – La réception audio-visuelle.

Stream). L'encodeur utilise un compteur pour fournir des PCRs, compteur qui est incrémenté par l'horloge système (STC : System Time Clock). La fréquence du signal d'horloge est notée $STC_{tx}(t)$ sur la figure et est dérivée du signal de synchronisation du signal vidéo analogique. Les PCRs représentent le temps de référence des DTS (Decoding Time Stamps) et PTS (Presentation Time Stamps), qui représentent respectivement les instants auxquels le décodage de chaque image et la présentation de chaque image doit se faire par le décodeur MPEG-2. Ces PCRs, qui sont des horodateurs, sont insérés dans le flux de transport, et sont utilisés par la suite dans le processus de décodage.

Une vue générale du côté réception est donnée par le schéma de la figure 1.2, qui reçoit les paquets de transport du flux. Les paquets du flux sont affectés par une variation de délai lorsque ceux-ci passent à travers le réseau. Ainsi, le débit instantané des paquets du flux reçu n'est pas identique au débit des paquets émis. Le débit des paquets du flux reçu est noté R_{rx} et les horodateurs de ces paquets sont notés T_{rx} .

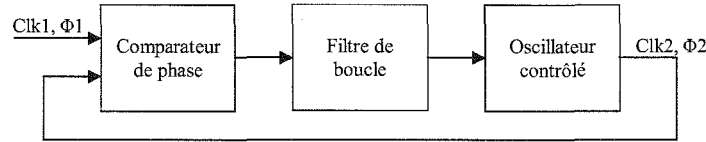


FIG. 1.3 – Schéma bloc d'une PLL.

Concernant le décodeur MPEG-2, celui-ci dispose de sa propre horloge système, qui permet de cadencer le processus de décodage. Cette horloge doit être synchronisée à l'aide du flux TS reçu. La fréquence de cette horloge est une estimation de la fréquence de l'horloge système de l'encodeur et est notée $\hat{STC}_{rx}$ sur la figure 1.2. Cette horloge doit refléter au mieux l'horloge système de l'encodeur, et est généralement restituée par une PLL [Kax00]. Cependant, d'autres systèmes de synchronisation existent et peuvent être adaptés à la synchronisation sur des horodateurs appliqués à la transmission de données audio-visuel.

1.3.1 Synchronisation par PLL

1.3.1.1 Principe général d'une PLL

Les PLLs sont constituées généralement de trois blocs comme le montre le schéma de la figure 1.3. Une comparaison de phase est effectuée entre le signal d'entrée et un signal généré localement à l'aide d'un oscillateur contrôlé. La synchronisation est obtenue lorsque l'écart de phase entre les deux signaux est nul. Le filtre de boucle, encore appelé correcteur, détermine le comportement dynamique de la PLL.

Le comparateur de phase et le filtre de boucle sont des éléments qui présentent peu de difficulté de développement. Le choix du type de filtre de boucle (PI, PID, ...) dépend du comportement dynamique désiré et du taux de réduction de la gigue d'entrée. L'aspect le plus sensible des PLLs est le choix d'une technique pour modifier la fréquence de l'oscillateur. Ce choix va entraîner des compromis sur la nature de l'implantation : hybride ou totalement numérique. Si le choix se porte sur un système totalement numérique, l'architecture peut être implantée facilement en ciblant une nouvelle technologie ; l'effort à fournir reste à un niveau relativement faible en paramétrant éventuellement les outils de synthèse logique. Les PLLs hybrides, devront quant à elles être re-conçues pour s'adapter avec la technologie en re-dessinant les différents composants tels que les VCO au niveau transistor.

Ainsi, les PLLs peuvent être classées suivant qu'elles utilisent des éléments analogiques, numériques ou mixtes, comme présenté dans les lignes qui suivent. Il sera présenté ensuite des architectures d'oscillateurs qui sont utilisés dans les PLLs actuelles. La fréquence de ces oscillateurs peut être contrôlée soit par une grandeur analogique ou soit par une valeur numérique.

Lorsque l'on desire implanter une PLL totalement numérique, on peut avoir recours soit à un oscillateur totalement numérique, soit à une technique de synthèse de fréquence.

1.3.1.2 Les différents types de PLL

Les PLLs peuvent être classées, suivant la nature des éléments utilisés (analogique ou numérique), de la manière suivante [XY02] :

- LPLLs (Linear PLLs) ou APLLs (Analog PLLs) : ces PLLs sont constituées uniquement d'éléments analogiques. Le comparateur de phase utilise un multiplieur analogique (quatre quadrants), et le filtre de boucle utilise des composants analogiques passifs ou actifs. Le signal de sortie de la PLL est généré par un VCO (Voltage Controlled Oscillator) [Kod97]. Ces PLLs ont des bandes passantes très larges et peuvent fonctionner à des fréquences très élevées (jusqu'à plusieurs Giga Hertz), et sont donc utilisées notamment dans les systèmes de transmission par satellite.
- DPLLs (Digital PLLs) : ces PLLs sont des circuits hybrides : tous ces éléments sont numériques mis à part le générateur de base de temps qui est réalisé à l'aide d'un DCO (Digitally Controlled Oscillator), encore appelé NCO (Numerically Controlled Oscillator). Dans le cas des DPLLs, le DCO intègre un VCO qui dispose d'une interface numérique/analogique pour le contrôle de la fréquence. On retrouve les DPLLs notamment dans les systèmes de synthèse de fréquence, sur des plages de fréquence allant jusqu'à quelques centaines de Méga Hertz.
- ADPLLs (All-Digital PLLs) : ces PLLs sont constituées uniquement d'éléments numériques. Elles sont limitées en fréquence de fonctionnement (jusqu'à 50 MHz) et sont utilisées notamment dans des applications qui nécessitent de réduire l'encombrement matériel.

1.3.1.3 PLLs utilisées pour la reconstruction d'horloge par PCR

D'après la classification des PLLs ci-dessus, les PLLs les mieux adaptées à la synchronisation sur des horodateurs sont les DPLLs ou les ADPLLs, du fait que l'information d'entrée est numérique. La génération d'horodateurs dans le cas du PCR se fait en numérisant une horloge avec une précision d'une période, par l'utilisation d'un compteur, comme le montre la figure 1.4 [Nor00]. Cette figure montre le principe de synchronisation d'une horloge par PLL lorsque l'entrée de référence est contrôlée par des PCRs. Des PCRs locaux sont générés, ce qui permet de comparer la phase numérique de l'oscillateur avec les PCRs d'entrée. La synchronisation d'horloge avec une PLL apporte deux fonctionnalités importantes. La première fonctionnalité est que le signal d'horloge de sortie est toujours disponible, même si le signal d'horloge d'entrée est discret ou lorsque le signal de référence est perdu sur une longue durée. La seconde fonctionnalité est que la PLL permet l'absorption de la gigue grâce à la présence du filtre de boucle. Celui-ci est

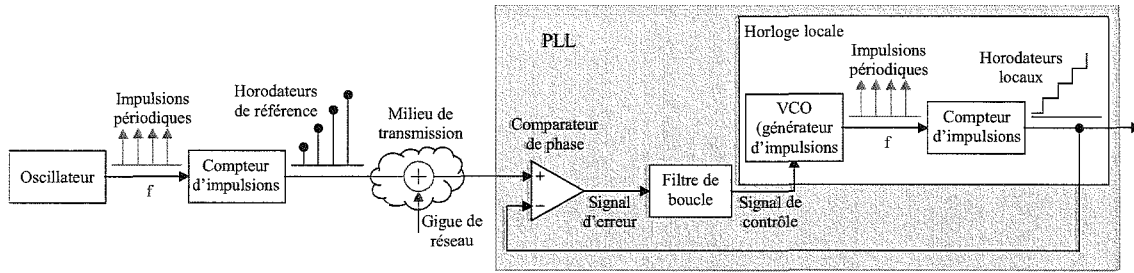


FIG. 1.4 – Schéma d'une PLL synchronisée sur des horodateurs.

en général un filtre proportionnel intégral (PI) [Gar79]. La bande passante des PLLs dans ce type d'application est très faible, généralement moins de $0,1 \text{ Hz}$, ce qui permet d'éliminer les composantes hautes fréquences du spectre de la gigue d'entrée.

1.3.2 Technique de synchronisation dans les SoCs submicroniques

Dans les circuits submicroniques, des technologies inférieures à 130 nm sont utilisées, ce qui permet de faire fonctionner des systèmes à plus de 10 GHz [Sri04]. Ces SoCs peuvent contenir plusieurs IPs. Ainsi, ils intègrent un nombre important de fonctions, ce qui peut rallonger la longueur des interconnexions, qui se comportent alors comme des inductances. Les délais de ces interconnexions sont souvent sous-estimés [Sri04]. Cela rend plus difficile la distribution des horloges dans un système, et nécessite alors l'utilisation de systèmes de resynchronisation.

Une technique de resynchronisation simple est donnée en figure 1.5 [Sri04]. Le générateur de phase produit, à partir de l'horloge d'entrée, des signaux d'horloges dont le déphasage est constant entre les horloges adjacentes. Le détecteur de phase estime la différence de phase dans les transitions entre l'une des phases multiples issues de l'horloge et celles du signal données. Le résultat de cette comparaison est fourni à la fonction de sélection des phases. La phase sélectionnée est ensuite utilisée pour échantillonner les données. En ré-échantillonnant les données avec R_2 en utilisant l'horloge propre au module, le signal d'entrée mésochrone devient synchrone dans le module.

Cette technique de synchronisation peut être adaptée à la synchronisation sur des PCR en numérisant les phases en amont du comparateur. Le schéma de la figure 1.6 donne une proposition d'adaptation à la synchronisation sur des PCRs. La présence d'un registre permet d'échantillonner la sortie du compteur à l'arrivée de chaque nouveau PCR. Par rapport à une PLL, cette solution ne dispose pas de filtre de boucle, ce qui signifie que le comportement dynamique de cette structure n'est pas paramétrable.

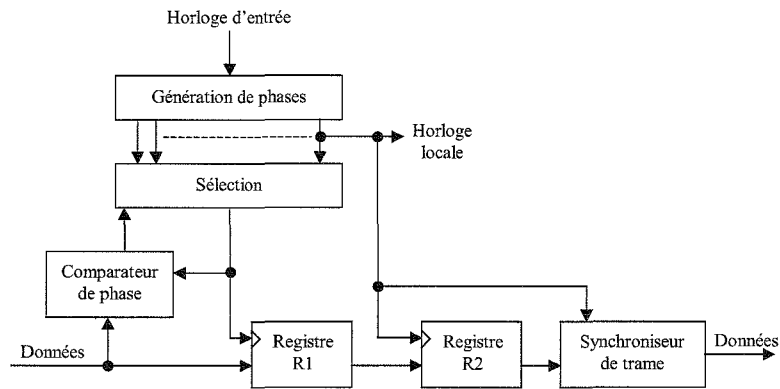


FIG. 1.5 – Technique de re-synchronisation des données.

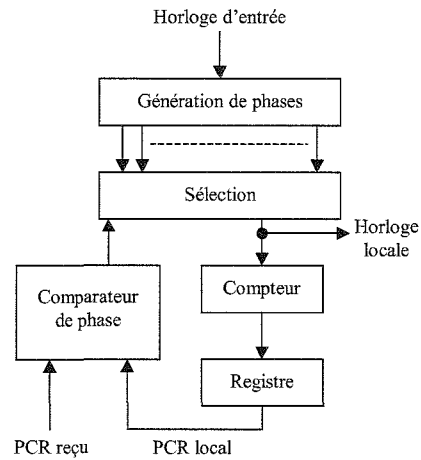


FIG. 1.6 – Proposition d'adaptation à la synchronisation sur des PCRs.

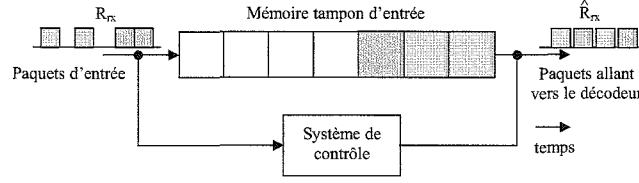


FIG. 1.7 – Principe général du système de suppression de gigue.

1.3.3 Synchronisation par une méthode de suppression de gigue (dejittering)

Le principe d'un système de suppression de gigue est d'estimer le débit des paquets $R_{tx}(n)$ de l'émetteur et d'envoyer le flux de paquets au décodeur MPEG-2 au débit estimé $\hat{R}_{tx}(n)$. Un modèle général de ce système est donné en figure 1.7 [Kax00]. Le flux de paquets arrivant du réseau est mis dans un buffer d'entrée. Un système de contrôle gère l'instant de lecture du paquet par le buffer et l'instant d'envoi au décodeur.

Le système de suppression de gigue peut fonctionner en débit binaire constant ou variable. Lorsqu'il est variable, des horodateurs inclus dans le flux doivent être utilisés afin d'estimer le débit des paquets $R_{tx}(n)$ de l'émetteur. Dans le cas présenté ici, les horodateurs des paquets RTPs (Real Time Protocole) peuvent être utilisés pour estimer le débit des paquets. Les paquets RTPs servent à empaqueter plusieurs paquets de transport de 188 octets ; ces paquets RTPs sont utilisés dans certains modes de transmission en DVB. Les horodateurs des paquets RTPs sont généralement utilisés pour estimer et réduire la gigue qui affecte ces paquets. Cependant, ils ne sont pas destinés être utilisés pour la restitution de l'horloge système MPEG-2.

La figure 1.8 [Kax00] donne le schéma de ce système, qui est implantable soit logiciellement soit matériellement. Dans ce dernier cas, il s'agira généralement d'une DPLL. Ce système, qui prend les horodateurs T_{rx} des paquets RTPs, estime ensuite les horodateurs $\hat{T}_{tx}$ basés sur l'horloge 90 kHz de l'émetteur. $\hat{T}_{tx}$, soustrait à un offset constant, est comparé aux horodateurs RTPs des paquets présents dans le buffer d'entrée. L'offset constant correspond à l'amplitude de gigue maximale que le système peut traiter. Les données des paquets RTPs sont alors envoyées au décodeur MPEG-2 à l'instant indiqué par l'horodateur résultant.

Un modèle du système est donné en figure 1.9 [Kax00]. Ce système fonctionne avec une horloge de fréquence f_s constante, les instants d'échantillonnage étant notés n_s . Il dispose aussi du signal T_{rx} représentant les horodateurs d'entrée et du signal T_l correspondant aux horodateurs issus de l'horloge du récepteur. Ainsi, à chaque nouvelle arrivée d'un horodateur T_{rx} , celui-ci est chargé dans un système d'interpolation.

Sachant que les paquets du flux n'arrivent pas forcément à chaque instant n_s , le système d'interpolation a comme but de générer un horodateur intermédiaire. Cela est réalisé en mesurant l'intervalle de temps entre l'instant d'arrivée de l'horodateur le plus récent et l'instant

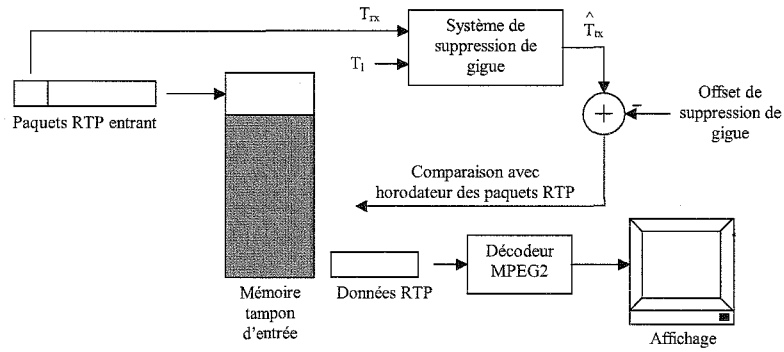


FIG. 1.8 – Schéma du système de suppression de gigue.

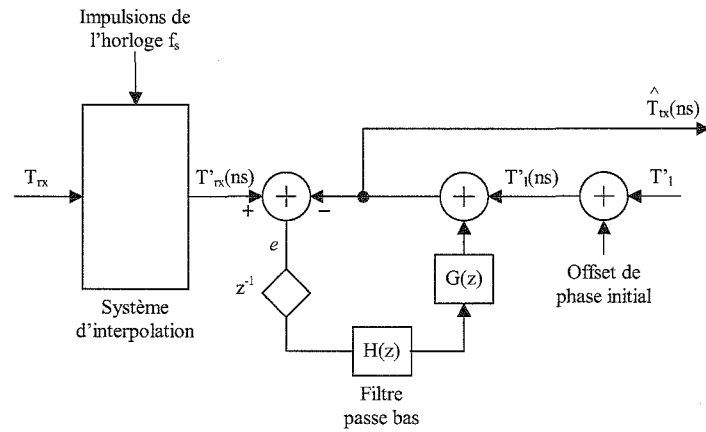


FIG. 1.9 – Modèle du système de suppression de gigue.

d'apparition d'un nouveau front sur l'horloge du système. Cet intervalle de temps numérique est ensuite additionné à la valeur de l'horodateur le plus récent et le nouvel horodateur T'_{rx} est ensuite présenté au système. L'intervalle de temps est mesuré à partir de l'horodateur T_l issu de l'horloge du récepteur :

$$T'_{rx}(n_2) = T_l(n_2) - T_l(n_1) + T_{rx}(n_1) \quad (1.1)$$

où n_1 est l'instant d'arrivée du dernier horodateur et n_2 est l'instant du front d'horloge du système sur lequel il est sensible.

L'horodateur $T'_{rx}(n_s)$ issu du système d'interpolation est comparé avec la valeur courante de l'horodateur estimé $\hat{T}_{tx}(n_s)$. La différence donne un terme d'erreur $e(n_s)$. Ce terme correspond à l'entrée d'un filtre passe bas $H(z)$. Le filtre est utilisé pour moyenniser les fluctuations de l'erreur, et correspond au filtre de boucle d'une DPLL lorsqu'il est implanté matériellement. La sortie de ce filtre commande ensuite un intégrateur discret.

La sortie de l'intégrateur est ensuite ajoutée à la valeur courante de $T'_{rx}(n_s)$. Cette somme donne l'horodateur estimé $\hat{T}_{tx}(n_s)$ de l'horloge de l'émetteur. $T'_l(n_s)$ correspond à l'horodateur T_l soustrait à une constante, qui représente l'offset de phase initial. L'avantage de cette solution est que les traitements peuvent se faire logiciellement ou matériellement. De plus, cette solution peut fonctionner avec un débit de PCR constant ou variable.

1.3.4 Synchronisation par des méthodes statistiques

La synchronisation par des méthodes statistiques, comme la regression linéaire des moindres carrés aussi appelée LLR (Least-square Linear Regression), peut être utilisée au lieu des PLLs [Nor00, NHH99, NH99]. Le principe de la LLR est de réaliser les opérations suivantes :

- collecter un nombre N de PCRs consécutifs et leur temps d'arrivée,
- représenter les échantillons collectés sous forme de points dans un espace bidimensionnel,
- faire passer une ligne droite par le plus de points,
- déterminer la pente de la ligne tracée.

Les points du graphique (figure 1.10) [Nor00] représentent les PCRs x_i en fonction du temps d'arrivée des échantillons y_i . En faisant passer une droite moyenne entre les points obtenus, il est possible de calculer la fréquence de l'horloge qui a générée les PCRs. La fréquence correspond à la pente de cette droite. Comme la gigue de transmission a une valeur moyenne nulle, les points doivent être distribués de manière égale en dessous et au-dessus de la droite. La valeur de la pente calculée représente le facteur de correction à utiliser pour synchroniser l'horloge locale avec l'horloge transmise.

Le PCR estimé par la méthode de la régression linéaire, durant l'intervalle entre l'arrivée de l'échantillon i et l'échantillon $i + 1$, est donné par l'équation 1.2 :

$$PCR^{LLR}(t) = a_i \cdot c_2(t) + b_i \quad (1.2)$$

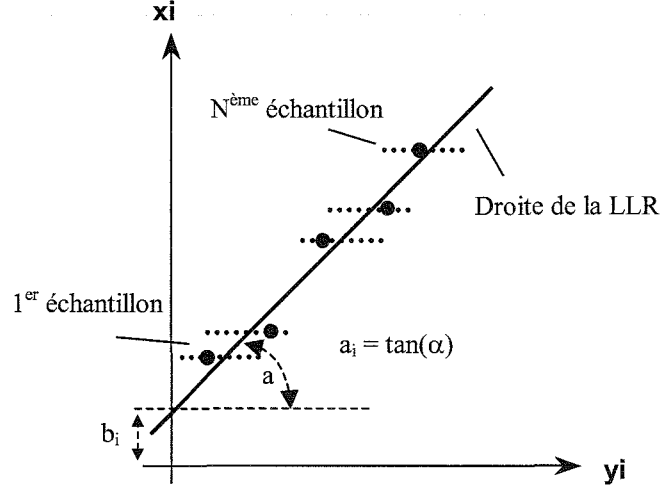


FIG. 1.10 – Principe de fonctionnement de la LLR.

où a_i correspond à la pente calculée pour l'échantillon i et b_i correspond à l'offset.

Le calcul de la pente a_i et de l'offset b_i de la droite est effectué avec les N derniers échantillons d'horloge reçus, incluant le i -ième échantillon. Les échantillons de l'horloge transmise sont notés $\{x_i\} = \{c_1(t_i), \forall i = 1, 2, 3, \dots\}$ et les temps d'arrivée sont notés $\{y_i\} = \{c_2(t_i + J_t^i), \forall i = 1, 2, 3, \dots\}$. Le terme J_t^i représente la gigue de transmission du i -ième échantillon d'horloge. Les équations de a_i et b_i sont les suivantes :

$$a_i = \frac{E_t\{x_j^2\} - E_t\{x_j\}E_t\{x_j\}}{E_t\{x_j y_j\} - E_t\{x_j\}E_t\{y_j\}} = \frac{N \sum x_j^2 - (\sum x_j)^2}{N \sum x_j y_j - \sum x_j \sum y_j} \quad (1.3)$$

$$b_i = \frac{E_t\{x_j\}E_t\{x_j y_j\} - E_t\{y_j\}E_t\{x_j^2\}}{E_t\{x_j y_j\} - E_t\{x_j\}E_t\{y_j\}} = \frac{\sum x_j \sum x_j y_j - \sum y_j \sum x_j^2}{N \sum y_j^2 - (\sum y_j)^2} = E_t\{x_j\} - a_i E_t\{y_j\} \quad (1.4)$$

où j prend les valeurs $\{j = i - N + 1, \dots, i\}$ et $E_t\{\cdot\}$ correspond au temps moyen des N dernières valeurs reçues.

L'algorithme de synchronisation de l'horloge est complété par un filtre passe-bas, connecté à la sortie de la fonction de calcul de a_i (équation 1.3). Le schéma de la figure 1.11 [Nor00] présente l'architecture permettant d'implanter la LLR. Comme le montre cette figure, l'inconvénient de la LLR est que les ressources requises pour le traitement sont importantes, car l'architecture est composée, pour le calcul des moindres carrés, de quatre registres, six additionneurs, six multiplieurs et un diviseur. De plus, il faut ajouter à cela les ressources du filtre de boucle et un oscillateur externe pour la génération de l'horloge locale.

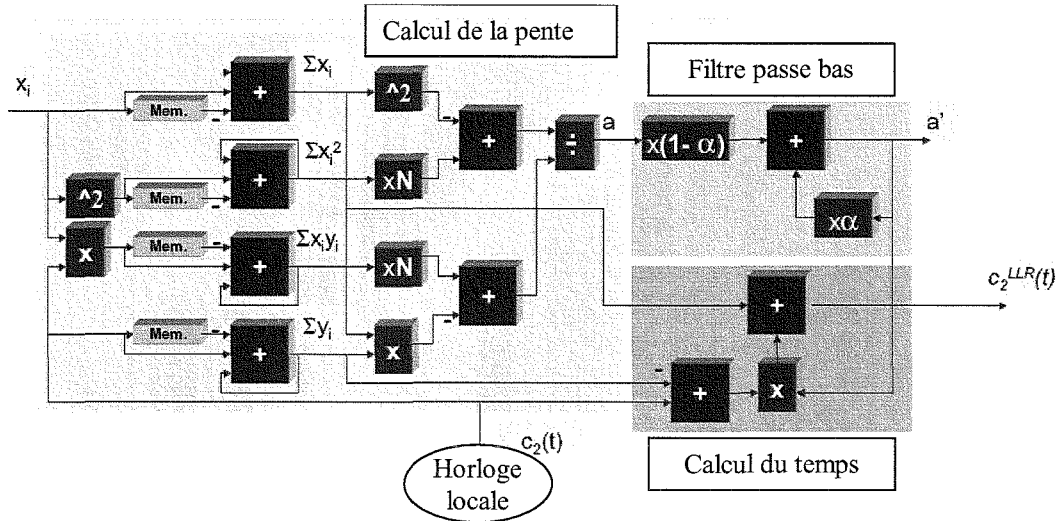


FIG. 1.11 – Architecture de la LLR.

1.4 Les techniques de génération d'horloge

1.4.1 Introduction

Les dispositifs de génération d'horloge contrôlée peuvent être classés suivant deux catégories. Dans un cas, ils utilisent un oscillateur contrôlé, et dans l'autre, ils utilisent un module de synthèse de fréquence. Les structures présentées seront donc soit des VCO ou des DCO, soit des architectures de synthèse. Certaines structures seront décrites de manière isolée, puis présentées au sein d'une de PLL, afin de donner le détail de l'architecture.

1.4.2 Génération d'horloge à l'aide d'oscillateur en anneau

1.4.2.1 Principe général

L'oscillateur en anneau le plus simple est composé d'inverseurs statiques identiques à celui de la figure 1.12(a) [GPB03]. Les inverseurs sont ensuite mis en cascade et le dispositif est rebouclé sur l'entrée du premier inverseur pour former l'oscillateur en anneau (figure 1.12(b)) [GPB03]. La fréquence des oscillations f dépend des temps de propagation des inverseurs comme le montre l'équation 1.5 où N est le nombre d'inverseurs et $\tau_{inverseur}$ est leur temps de propagation.

$$f = \frac{1}{2.N.\tau_{inverseur}} \quad (1.5)$$

L'inconvénient de la structure précédente est qu'elle ne consomme pas un courant constant, ce qui produit du bruit dans l'alimentation qui est dû à la commutation des transistors. Une

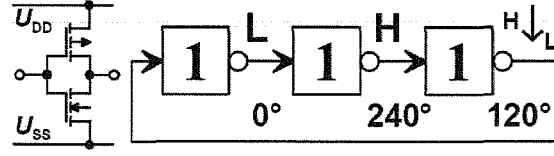


FIG. 1.12 – Cellule de délais à inverseur CMOS (a), oscillateur en anneau à trois étages d'inverseurs (b).

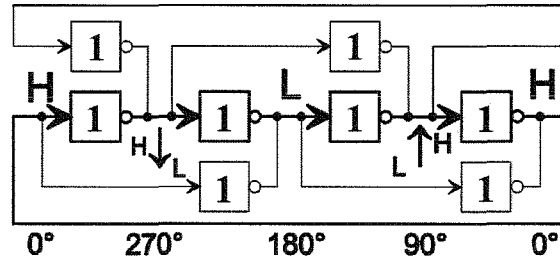


FIG. 1.13 – Oscillateur en anneau à quatre étages d'inverseurs pour fournir des signaux en quadrature.

solution permettant de minimiser ce bruit est la génération d'horloges en quadrature. Cette solution permet de générer un front montant et un front descendant à chaque transition. Le courant d'alimentation de l'oscillateur en quadrature est alors constant, ce qui minimise le bruit de commutation sur les lignes d'alimentation.

Pour générer des sorties en quadrature, il est nécessaire d'utiliser un nombre paire d'inverseurs pour former l'anneau, ce qui entraîne l'impossibilité de générer des oscillations car le point de fonctionnement des inverseurs est statique et stable. Pour résoudre ce problème, une solution est l'emploi d'inverseurs en "feedforward" entre les noeuds où les signaux sont en opposition de phase [GPB03].

Le schéma d'une telle structure est donné en figure 1.13 [GPB03]. Les inverseurs en "feedforward" fonctionnent avec des cellules à gain $-Gm$ [TK97]. Ainsi, ces inverseurs font tendre les sorties des inverseurs de l'anneau, sur lesquels ils sont connectés, à des niveaux opposés, ce qui fait basculer les inverseurs en aval. Contrairement à un oscillateur en anneau avec un nombre impair d'inverseurs, il y a maintenant deux transitions qui se propagent dans les anneaux en même temps. Ainsi, la fréquence de fonctionnement d'un tel dispositif est donnée par l'équation 1.6.

$$F = \frac{1}{N \cdot \tau_{inverseur}} \quad (1.6)$$

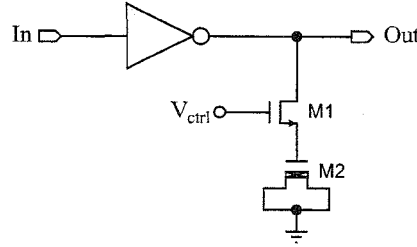


FIG. 1.14 – Modification du délai par variation de charge

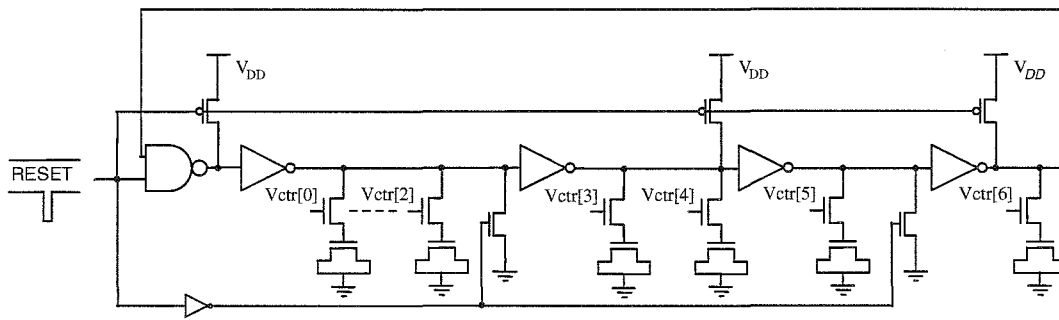


FIG. 1.15 – Modification du délai par contrôle numérique de la charge

1.4.2.2 Contrôle de fréquence par charge variable

La structure permettant de modifier le délai d'un inverseur par modification de la charge des cellules de délais [MMY⁺00] est donnée en figure 1.14 [JH98, MNS03]. Cette figure montre une cellule de délais utilisant le transistor $M2$ comme capacité de charge. Dans ce circuit, le transistor $M1$ contrôle le courant de charge et de décharge de $M2$ via la tension V_{ctrl} . Ainsi, la charge variable en sortie de l'inverseur modifie le délai introduit par la porte.

La modification de la fréquence par charge variable peut aussi se faire à l'aide d'un contrôle numérique. Le schéma de la figure 1.15 [ONM⁺00] donne un exemple où la sortie des inverseurs est chargée par un ensemble de transistors à canal N fonctionnant en tant que capacité. Ces capacités sont connectées aux sorties des inverseurs via des transistors fonctionnant en tout ou rien.

Les mesures effectuées permettent de déterminer une pente de $2,7 \text{ ps/bit}$ [ONM⁺00]. La résolution de ce DCO est donnée par la pente du DCO multipliée par un facteur (qui est utilisé pour une division en fréquence). Le DCO présenté est conçu pour obtenir une résolution de 1 ns pour la valeur du facteur la plus élevée, qui est 256. Les simulations indiquent une plage de fréquences se trouvant entre 556 MHz et 864 MHz . La plage de fréquence mesurée va par contre de 790 MHz à $1,08 \text{ GHz}$.

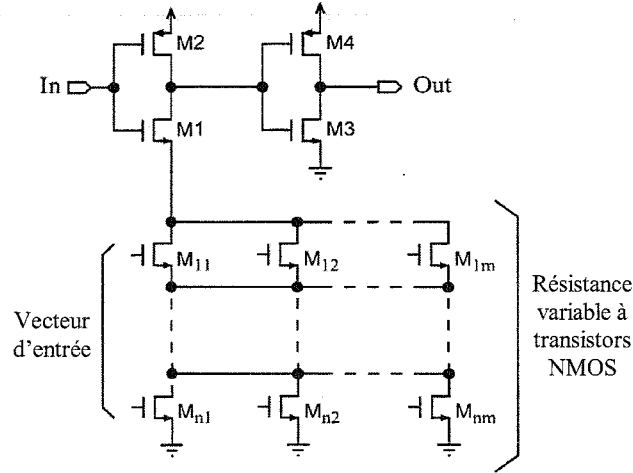


FIG. 1.16 – Modification du délai par résistance variable

L'intérêt de ce DCO est qu'il est possible d'avoir des fréquences élevées en sortie, avec une résolution de 1 ns dans le pire cas. Sachant que l'on souhaite générer dans notre application des fréquences se trouvant dans la plage $27000000\text{ Hz} \pm 800\text{ Hz}$, soit une période allant de $37,0381345\text{ ns}$ à $37,0359397\text{ ns}$, cela nécessite une résolution d'au moins de la femto seconde. Le DCO présenté ici n'a donc pas une résolution suffisante. Aussi, l'utilisation de cette technique de contrôle de fréquence n'est pas très répondue car les délais produits sont difficiles à calibrer [BFL⁺01, BFL⁺02, BFL⁺03, BFL⁺04] et parce qu'il existe des techniques plus simples où le contrôle se fait directement dans la cellule de délais par la modification de sa polarisation.

1.4.2.3 Contrôle de la fréquence par résistance variable

Dans le domaine de la conception microélectronique, une résistance variable est obtenue en faisant fonctionner un transistor CMOS dans sa plage résistive, la valeur étant modifiée en changeant la polarisation du transistor. Un exemple de technique de contrôle de fréquence par résistance variable est présenté en figure 1.16 [MNS03]. Dans cet exemple, la résistance variable est formée d'une matrice de transistors *nMOS*. Dans ce cas, la résistance est contrôlée par une valeur numérique qui modifie le délai introduit par le transistor *M1*.

Une variante de cette structure est donnée en figure 1.17 [MNS03], où le contrôle de la fréquence se fait par une résistance formée d'un vecteur de transistors. La structure de cette figure est un élément de délai où les délais sont déterminés par la taille des transistors.

La figure 1.18 [MNS03] montre l'effet du rapport W/L des transistors *pMOS* sur les délais produits par le circuit. La figure 1.18(a) montre l'effet du rapport W/L de *M5* sur le délai alors que les rapports de *M1* à *M4* sont constants. On constate que *M5* agit principalement sur le

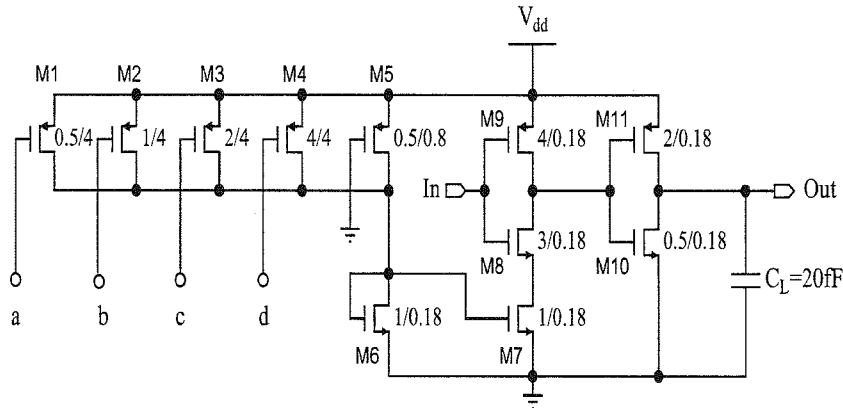


FIG. 1.17 – Modification du délai par résistance variable (variante)

délai maximal. Dans la figure 1.18(b), le rapport W/L de $M5$ est gardé constant alors que ceux de $M1$ à $M4$ sont modifiés. Les transistors $M1$ à $M5$ n'ont pas d'influence sur le délai maximal, mais permettent d'agir sur le délai minimal du circuit.

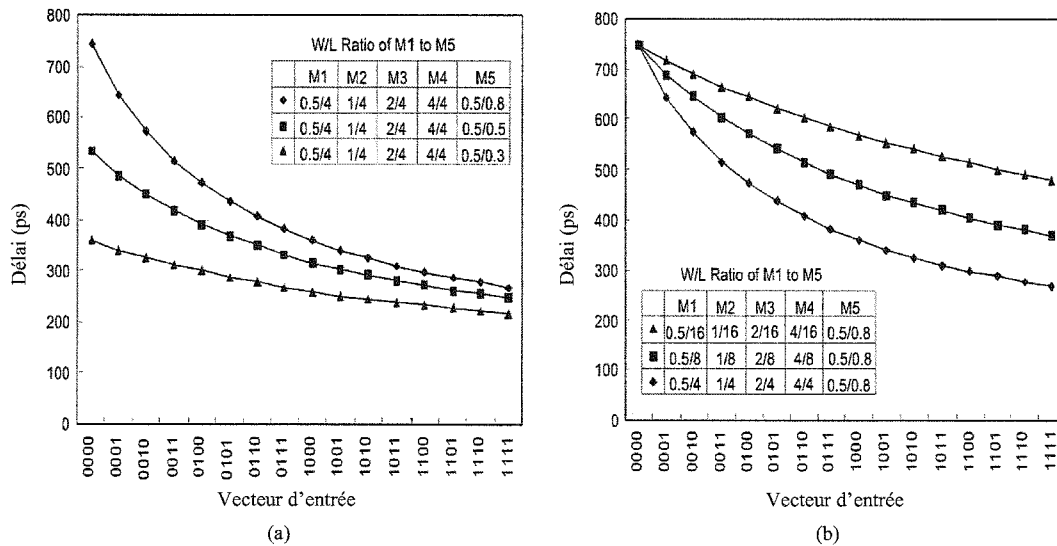


FIG. 1.18 – Evolution du délai en fonction de l'entrée

L'évolution du délai avec la valeur d'entrée n'est pas linéaire. Cependant, les deux courbes de la figure 1.18 qui passent par des points en forme de triangle se rapprochent le plus d'une droite. L'avantage d'avoir une période qui varie linéairement avec la valeur d'entrée est que l'erreur de période relative maximale est constante. Mais la variation linéaire de la période entraîne une variation non linéaire de la fréquence générée. Pour que la variation de fréquence

soit linéaire, la variation de la période en fonction de la valeur d'entrée doit être en $1/x$. Les pentes de ces deux courbes sont d'environ $6,7 \text{ ps/bit}$ pour la courbe de la figure 1.18(a) et d'environ $16,7 \text{ ps/bit}$ pour la figure 1.18(b). Ces pentes correspondent à la résolution du DCO. Pour utiliser cette structure dans le domaine de la mesure de paramètres relatifs au PCR, une fréquence de 27 MHz est requise, ce qui entraîne l'utilisation d'un nombre important de transistors $pMOS$ si la taille des transistors restent dans le même ordre de grandeur que ceux de la figure 1.18. Une autre solution est de redimensionner les transistors, ce qui entraîne des tailles de transistors plus importantes. Dans les deux cas, le générateur d'horloge obtenue occuperait une surface importante, d'autant plus qu'il y a autant de générateur d'horloge à implémenter qu'il y a de programmes dans le flux.

1.4.2.4 Contrôle de la fréquence par modification de la polarisation

Le schéma de la figure 1.19 [MNS03] donne le principe général du contrôle du délai par modification de la polarisation des transistors MOS. La modification de la polarisation est réalisée en agissant directement sur le courant circulant dans les branches des inverseurs [BZVL01, Man96]. Lorsque le contrôle de la fréquence se fait par une valeur numérique, l'oscillateur est un DCO. Pour le réaliser, un CNA (Convertisseur Numérique-Analogique) est utilisé lorsque le contrôle de la fréquence se fait par l'intermédiaire d'une tension, et lorsque le contrôle est fait par le courant, un convertisseur tension-courant est requis.

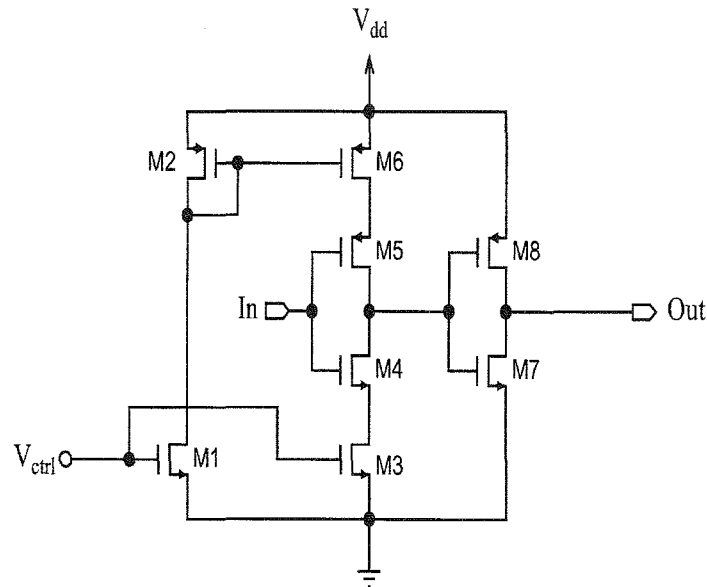


FIG. 1.19 – Modification du délai par contrôle de polarisation

Il existe aussi des DCOs en anneau totalement numériques. Ils sont souvent utilisés dans les

ADPLLs. Ces DCOs sont constitués d'inverseurs en matrice [HWL01, CL03], comme le montre la figure 1.20 [ON04]. Ce DCO est composé d'un ensemble de 21 inverseurs à trois états mis en parallèle. Le premier inverseur de la chaîne est remplacé par une porte NAND pour permettre de bloquer les oscillations. Lorsque les inverseurs à trois états sont validés, un courant additionnel est ajouté à chaque étage d'inverseurs. Les 126 inverseurs à trois états sont contrôlés par un vecteur C de 126 bits, qui est décodé à partir d'un mot de contrôle W de 7 bits. Tous les 126 bits du vecteur sont à '1' lorsque $W = 0$, sont tous à zéro lorsque $W = 126$ ou 127, et pour $W < 126$ ou 127, le nombre de zéros dans C est égal à la valeur de W .

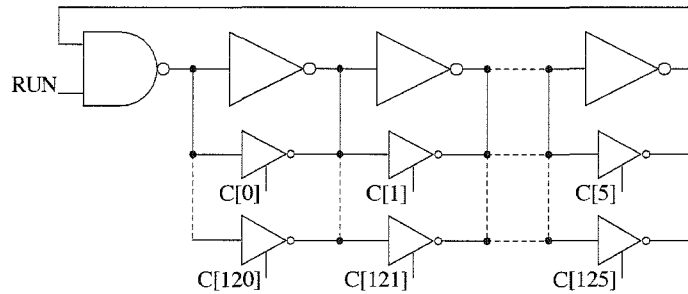


FIG. 1.20 – DCO utilisant des inverseurs trois états en parallèle pour l'ajustement de la fréquence.

Les résultats de mesure de la période en fonction de la valeur numérique d'entrée sont données en figure 1.21 [ON04]. Ces résultats permettent de déterminer la pente du DCO (résolution), qui est variable et qui se situe entre 10 ps et 150 ps. Comme la gigue intrinsèque d'un DCO est liée à sa résolution, plus l'écart minimal entre deux périodes T_1 et T_2 est faible, plus on se rapproche de la période réelle que le synthétiseur doit générer, et plus la gigue intrinsèque au DCO diminue. Ainsi, sa résolution doit être considérablement supérieure aux performances requises en terme de gigue. L'inconvénient de cette structure est qu'elle ne dispose pas d'une résolution suffisante par rapport aux besoins de notre application. Une solution pour affiner les délais est l'insertion, dans la boucle de retour de l'oscillateur en anneau, d'une matrice de délais [SSSS03], à l'aide d'inverseurs interconnectés comme dans le schéma de la figure 1.20 [ON04].

1.4.2.5 Contrôle de la fréquence par sélection de chemin

En modifiant le nombre d'inverseurs utilisés pour réaliser un oscillateur en anneau, il est possible de modifier sa fréquence de fonctionnement. Dans ce cas, cet oscillateur utilisé dans une PLL la rendra aussi imprécise et instable. Une solution pour améliorer l'imprécision et la stabilité est l'insertion, dans la boucle de retour (comme dans la figure 1.22 [SSSS03]), de délais plus faibles à l'aide d'une matrice de délais, comme présentée en figure 1.20 [ON04]. La différence entre le chemin de délai le plus lent et le plus rapide doit être plus petit que le délai d'un élément

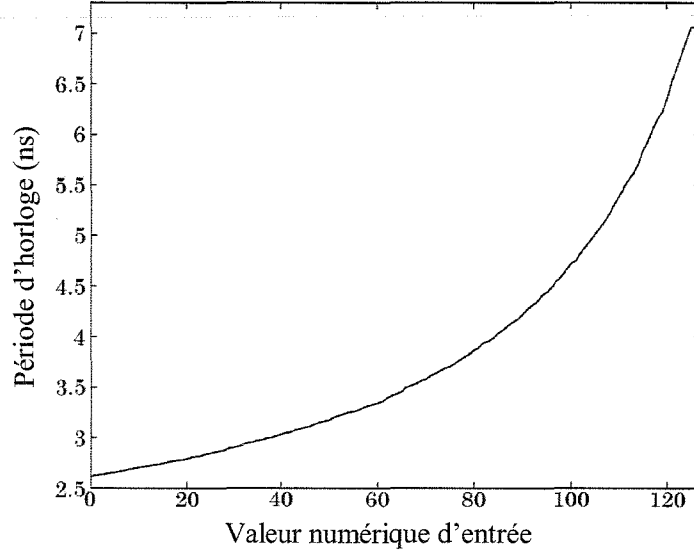


FIG. 1.21 – Variation du délai en fonction de l'entrée.

de de l'anneau. En respectant cette condition, l'oscillateur fournira une fréquence avec une haute résolution. Cependant, comme il existe une relation entre le délai de introduit par un élément et le délai introduit par la matrice de délais, cette matrice doit être redessinée pour chaque technologie. Ainsi, ce DCO ne peut pas être implanté de manière portable. Pour rendre portable cet oscillateur, il est nécessaire d'utiliser la structure présentée en figure 1.23 [SSSS03], où la matrice de délais a été enlevée. La fréquence de sortie de f_{osc} est donnée par :

$$f_{osc} = \frac{1}{2 \cdot (L \cdot \tau_{de} + \tau_{fs})} \quad (1.7)$$

où τ_{de} est le délai introduit par un élément de délai de et τ_{fs} est le délai produit par le sélecteur fs . Les paramètres L et τ_{fs} permettent d'agir sur la fréquence de l'oscillateur.

Dans cette configuration, la résolution fréquentielle de l'oscillateur n'est plus suffisante, car le pas des délais est à nouveau important. L'insertion dans la boucle de contre-réaction de l'oscillateur d'un synthétiseur de fréquence, comme celui qui va être présenté en section 1.4.3.1, permet de disposer d'un oscillateur portable de résolution fréquentielle élevée. Cette structure utilise l'oscillateur en anneau de la figure 1.23 et une unité de contrôle qui commute le nombre d'éléments de délais dans l'oscillateur en anneau suivant deux valeurs entières adjacentes (figure 1.24) [SSSS03]. Ainsi, le nombre moyen d'éléments de délais devient un nombre fractionnaire, qui permet une haute résolution en fréquence. Le diviseur de fréquence fractionnaire de la figure 1.30 (section 1.4.3.1) peut être utilisé comme unité de contrôle. En moyenne, le signal de sortie OVF de ce dernier est à '1' sur N des M cycles du signal d'entrée clk_{osc} et est à '0' sur le restant des $M - N$ de ces cycles. Si l'on pose l'hypothèse qu'un élément de délai de doit fournir

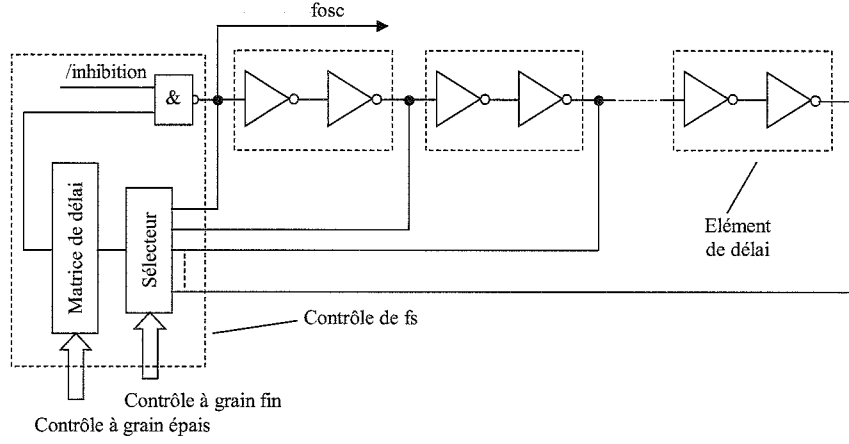


FIG. 1.22 – DCO utilisant une matrice de délai.

une durée représentée par une valeur fractionnaire l pour générer une fréquence donnée f_{osc} , le signal OVF commute la valeur de contrôle L entre deux valeurs entières adjacentes L_1 et L_2 :

$$l = \frac{N.L_2 + (M - N).L_1}{M} = \frac{1}{2.\tau_{de}.f_{osc}} - \frac{\tau_{fs}}{\tau_{de}} \quad (1.8)$$

où

$$L_2 > l > L_1 \quad (1.9)$$

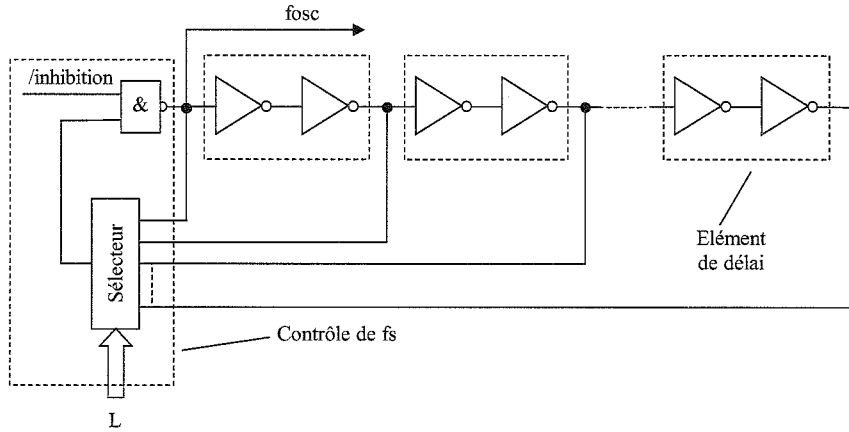


FIG. 1.23 – DCO portable.

Comme le signal de contrôle L de l'oscillateur en anneau de la figure 1.24 commute entre deux valeurs L_1 et L_2 , la période du signal d'horloge clk_{osc} commute aussi entre deux valeurs T_1 et T_2 causant une gigue ΔT sur ce signal. La période T du signal d'horloge clk_{osc} est donnée

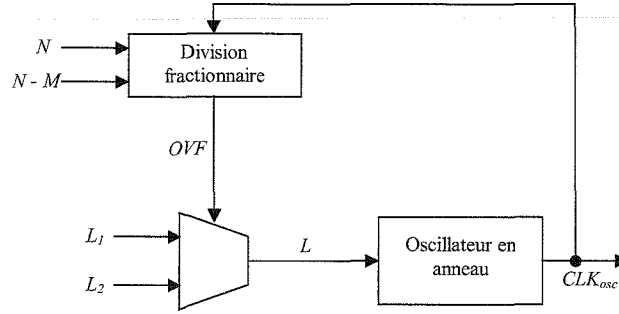


FIG. 1.24 – DCO portable disposant d'une résolution fréquentielle élevée.

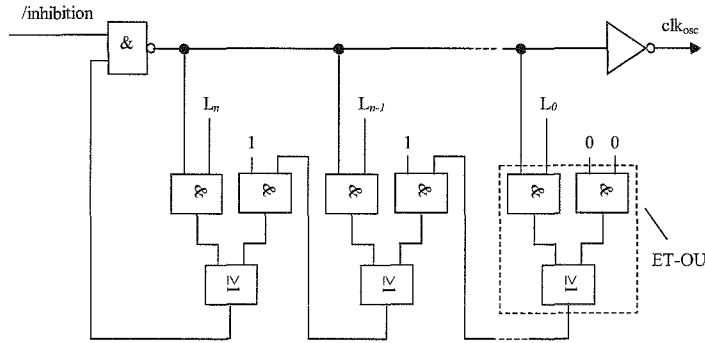


FIG. 1.25 – Schéma détaillé de l'oscillateur en anneau portable.

par :

$$T = 2. (\tau_{fs} + L. \tau_{de}) \quad (1.10)$$

La gigue crête-à-crête ΔT est donnée par :

$$\Delta T = T_2 - T_1 = 2. \tau_{de} \quad (1.11)$$

Le schéma détaillé de l'oscillateur en anneau est présenté en figure 1.25 [SSSS03]. Ce DCO est facilement adaptable pour différentes technologies en choisissant un nombre approprié de portes ET-OU dans l'anneau. Le mot de contrôle L définit le nombre de portes ET-OU actives dans l'oscillateur. Seul un bit de ce mot doit être mis à '1', les autres étant à '0'.

Le DCO présenté est utilisé dans une ADPLL (figure 1.26 [SSSS03]), qui comprend aussi un détecteur de phase, un filtre de boucle, un diviseur de fréquence fractionnaire, un synthétiseur de fréquence et un gestionnaire d'horloge. Le détecteur de phase compare la phase de l'horloge de référence clk_{ref} avec la phase de l'horloge restituée clk_{div} . Le filtre de boucle effectue ses calculs à chaque cycle de l'horloge de référence. L'écart temporel t_e entre le front montant de clk_{ref} et

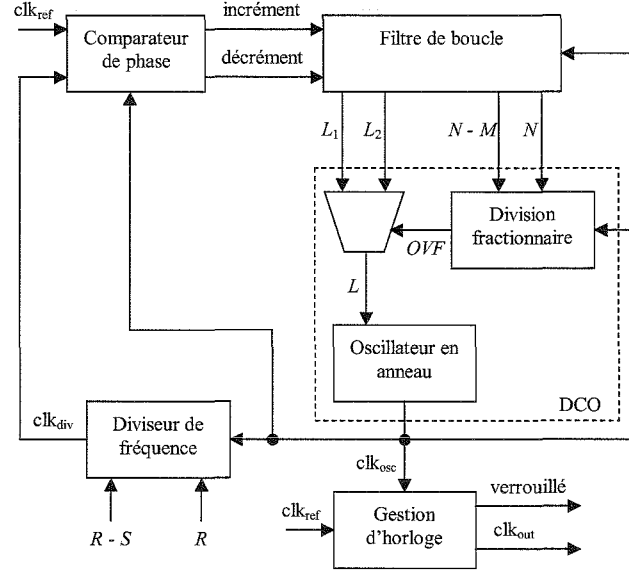


FIG. 1.26 – ADPLL utilisant un DCO.

clk_{div} permet de calculer l'erreur de phase E :

$$E = T_e \cdot f_{osc} \quad (1.12)$$

Le filtre de boucle calcule ensuite la valeur de l en fonction de l'équation suivante :

$$l = l_0 - \frac{l_0}{K} \cdot E \quad (1.13)$$

où l_0 est la valeur nominale de l et où K est déterminée par :

$$K = \frac{S}{R} = \frac{f_{osc}}{f_{ref}} \geq 1 \quad (1.14)$$

et où la valeur de l_0 est donnée par :

$$l_0 = \frac{1}{2 \cdot \tau_{ao,0} \cdot f_{osc}} - \frac{\tau_{nand,0}}{\tau_{ao,0}} \quad (1.15)$$

$\tau_{ao,0}$ et $\tau_{nand,0}$ sont les valeurs nominales des délais des portes NON-ET-OU et de la porte NON-ET de l'oscillateur en anneau de la figure 1.25. A partir de l , le filtre de boucle calcule les valeurs de L_1 , L_2 , N et $N - M$.

Le gestionnaire d'horloge détermine la valeur moyenne de la fréquence de l'oscillateur en anneau et positionne à '1' le signal *locked* lorsque l'ADPLL est verrouillée.

Le système représenté en figure 1.26 est décrit en code VHDL synthétisable et est implanté dans plusieurs cibles : un FPGA Xilinx V400BG432, en technologies CMOS AMS 0,6 μm et 0,35 μm . Le tableau 1.1 [SSSS03] donne la surface occupée et la fréquence maximale du signal

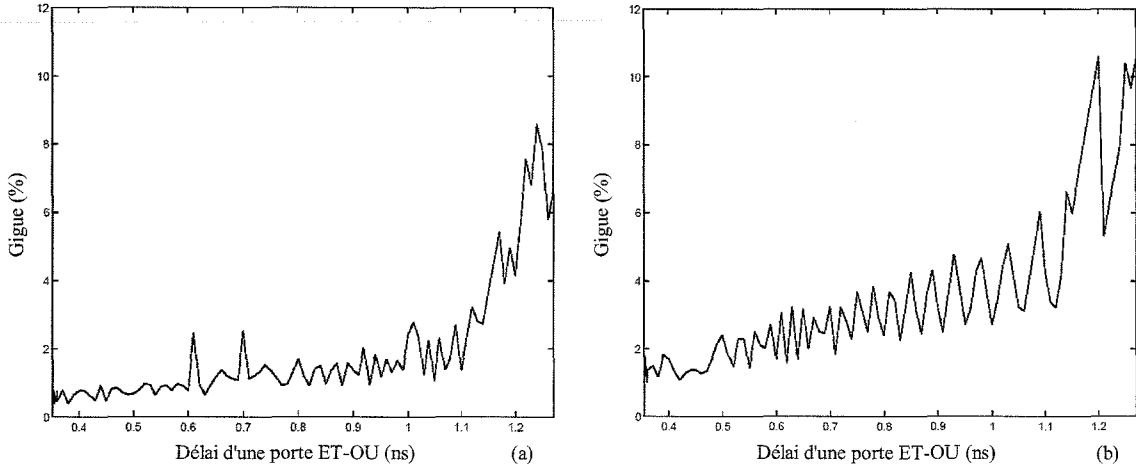


FIG. 1.27 – Gigue relative de l'horloge de sortie de l'ADPLL. Pour $f_{out} = 20 \text{ MHz}$ (a), pour $f_{out} = 52 \text{ MHz}$ (b).

clk_{out} de l'ADPLL pour chaque cible. Les fréquences maximales de fonctionnement présentées dans ce tableau sont déterminées par des mesures dans le cas du FPGA et par simulation pour les technologies AMS.

Cible	Surface/slices	Fréquence d'horloge maximale
FPGA Xilinx V400BG432	672 slices	40 MHz
AMS 0.6 μm CUB	4 mm^2	52 MHz
AMS 0.35 μm CSX_HRDLIB	0.543 mm^2	52 MHz

TAB. 1.1 – Résultats d'implantation de l'ADPLL pour plusieurs cibles.

Dans le cas idéal, la gigue crête à crête du signal de sortie de l'ADPLL est égale au délai de deux éléments de délai. Cependant, la largeur des mots L_1 et L_2 a été limitée lors des tests (pour des raisons d'implantation) [SSSS03], qui ont donné les résultats de la figure 1.27 [SSSS03], ce qui crée des sauts de gigue. La gigue est mesurée pour une fréquence de sortie de l'ADPLL à 20 MHz (figure 1.27(a)) et à 52 MHz (figure 1.27(b)). La gigue reste importante et peut atteindre 8 % pour une fréquence de sortie à 20 MHz et 10 % pour une fréquence de sortie à 52 MHz. Aussi, pour fournir n fréquences discrètes avec une gigue crête à crête plus faible, il est nécessaire d'utiliser un nombre important d'éléments de délais, à savoir $2n$ inverseurs, ce qui entraîne aussi des largeurs importantes des mots de contrôle du délai.

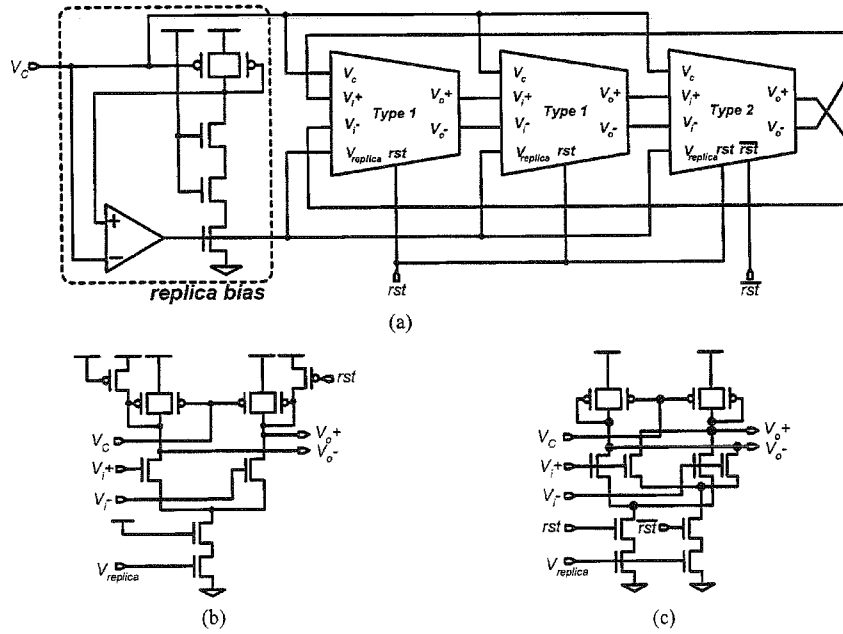


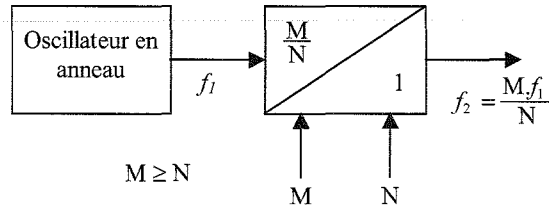
FIG. 1.28 – Oscillateur en anneau à trois étages avec circuit de polarisation (a), simple cellule de délais (b), cellule de délais avec fonction de décalage de phase (c).

1.4.2.6 Technique de minimisation de la gigue due à l'alimentation

Dans le domaine de la restitution d'horloge, cinq sources de gigue peuvent être identifiées [AR01] :

- la gigue générée par une entrée bruyante,
- la gigue d'oscillateur due au bruit électronique des composants,
- le bruit d'alimentation [HP01] et de substrat,
- la perturbation d'un oscillateur par courant de fuite ("leakage") traversant le comparateur de phase lors des transitions des données,
- la gigue d'oscillateur due à l'ondulation sur la grandeur de contrôle.

La gigue d'alimentation peut être minimisée en utilisant des inverseurs totalement différentiels [XY02, WCN⁺99, SLL04, LK00, NM05, Cor04]. La figure 1.28(a) en présente un exemple à trois étages [HSK01]. Cet oscillateur en anneau utilise deux types d'inverseurs différentiels. Les deux premiers étages utilisent de simples cellules de délais (figure 1.28(b)) et le troisième étage est une cellule de délais dotée d'une fonction de décalage de phase (figure 1.28(c)), permettant de bloquer les oscillations lors d'un reset. L'inconvénient de ce type d'oscillateur est d'utiliser un nombre plus important de transistors dans un étage, ce qui augmente le nombre de capacités parasites, qui réduisent la fréquence maximale de fonctionnement par rapport à un étage inverseur non différentiel.

FIG. 1.29 – Principe de la division fractionnaire par N .

1.4.3 Génération d'horloge par synthèse de fréquences

1.4.3.1 Architecture de division fractionnaire par N

Une première variante de synthèse de fréquence peut être basée sur la division entière d'une fréquence élevée et fixe f_1 à l'aide d'un compteur. Cependant, cette solution n'offre pas la possibilité de synthétiser les fréquences avec une grande résolution. Pour accroître cette résolution, une division fractionnaire est nécessaire. Le principe général de cette technique est donné en figure 1.29. La fréquence de sortie f_2 est contrôlée en modifiant N ou M . Le facteur de division change périodiquement en sélectionnant l'une des deux valeurs d'entrée N ou $N - M$, avec $N > 0$ et $N - M < 0$. Ainsi, une division fractionnaire est obtenue en moyennant la fréquence du signal f_2 .

Cette solution est réalisée avec un accumulateur muni d'une entrée multiplexée. Le schéma d'un tel synthétiseur est présenté en figure 1.30 [SSSS03, SSDS02]. Le bit MSB de l'accumulateur, appelé *OVF*, est le signal de sortie du synthétiseur. Ce bit est utilisé pour sélectionner l'une des deux valeurs N ou $N - M$ se trouvant sur l'entrée de l'accumulateur. Si ce bit est à '0', la valeur $N - M$ est sélectionnée, sinon la valeur N est sélectionnée. La gigue crête à crête causée par ce synthétiseur est égale à une période de l'oscillateur haute fréquence. L'effet de cette gigue peut être négligé dans certaines applications, mais pour d'autres, il ne peut pas être toléré, surtout lorsque de faibles facteurs de division sont sélectionnés. De plus, la fréquence f_1 d'entrée de ce synthétiseur est limitée par le temps critique de l'additionneur.

1.4.3.2 Architecture PSDCO

Les figures 1.31 et 1.32 [DCP⁺89] présentent une technique basée sur le décalage de phase. La phase du signal de sortie est modifiée par un registre à décalage [DCP⁺89]. Cette technique permet de surmonter les problèmes de limitation d'horloge car les registres à décalage peuvent fonctionner à des fréquences très élevées. Le système fait re-circuler le motif d'un signal carré dans le registre. Ainsi, la structure réalise une synthèse de fréquence par décalage de phase et l'architecture est appelée PSDCO (PSDCO : Phase Shifting Digitally Controlled Oscilateur).

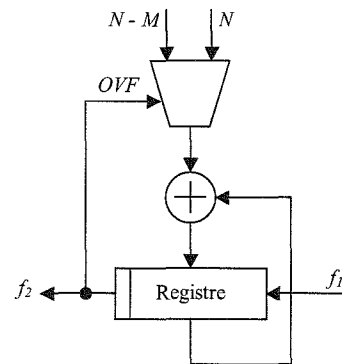


FIG. 1.30 – Structure de la division fractionnaire par N .

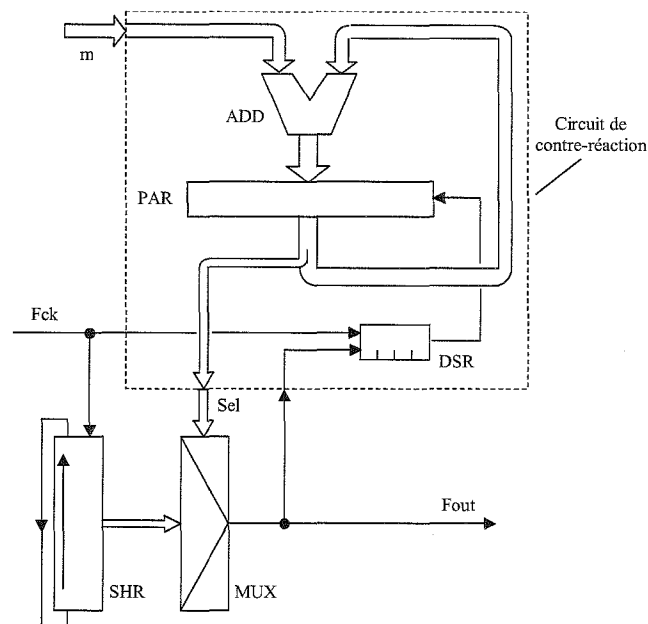


FIG. 1.31 – Architecture simplifiée du synthétiseur de fréquence.

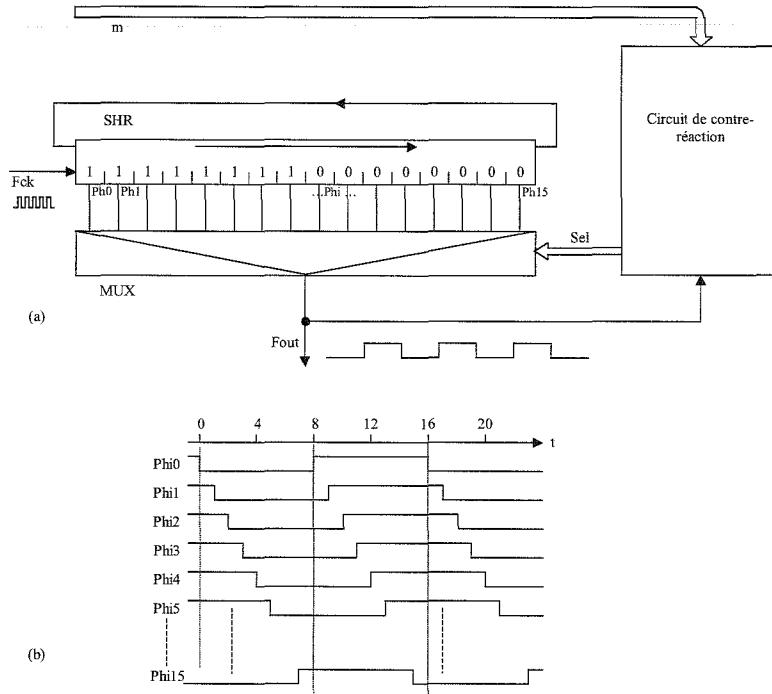


FIG. 1.32 – Coeur du synthétiseur de fréquence.

Plus précisément, un registre à décalage sur N bits est connecté à un multiplexeur. Ce registre à décalage est rebouclé sur lui-même et est préchargé à l'initialisation par un motif binaire constitué d'une succession de '0' puis de '1' pour réaliser le motif d'un signal carré. Ce motif binaire circule à la fréquence de l'horloge de référence F_{ck} . Le multiplexeur sélectionne ensuite l'une des sorties du registre à décalage, en fonction du mot de sélection Sel . La phase du signal de sortie peut être modifiée sur une plage allant au-delà de 2π . La sélection de la phase est mise à jour à chaque cycle de F_{out} et la période de sortie est modifiée en fonction de la valeur de m . La valeur de m , comme le contenu du PAR (Phase Accumulator Register), est une valeur fractionnaire, signée et sur un format à virgule fixe. La partie entière du PAR correspond au terme Sel . Lorsque le synthétiseur fonctionne avec le registre PAR à zéro et lorsque $m = 0$, la fréquence de sortie est égale à :

$$F_{out} = F_0 = \frac{F_{ck}}{N} \quad (1.16)$$

où F_0 est la fréquence centrale du synthétiseur. Lorsque l'on a $m = +1$, le registre PAR est incrémenté à chaque cycle de F_{out} , la valeur de Sel est modifiée et la période du signal généré est raccourcie d'une durée égale à la période de l'horloge F_{ck} . Avec le même raisonnement, si m est négatif, la période est rallongée. La fréquence de sortie est alors donnée par l'équation suivante :

$$F_{out} = \frac{F_{ck}}{N - m} \quad (1.17)$$

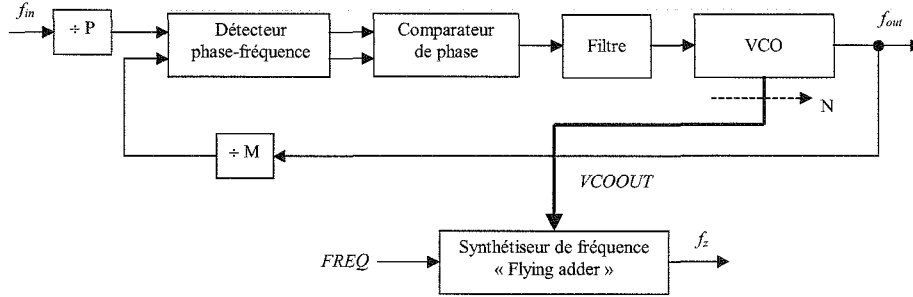


FIG. 1.33 – PLL utilisant un VCO générant des horloges polyphases, utilisées pour le synthétiseur de fréquence "Flying Adder".

Pour des valeurs faibles de m ($m \ll N$), la relation 1.17 peut être linéarisée :

$$F_{out} = \frac{F_{ck}}{N - m} \simeq \frac{F_{ck}}{N} \cdot \left(1 + \frac{m}{N}\right) = F_0 + m \cdot \frac{F_0}{N} \quad (1.18)$$

L'avantage de cette technique est que ce synthétiseur peut fonctionner à des fréquences élevées. Cette structure peut fournir, par exemple, une horloge de fréquence centrale à $28MHz$, qui peut varier avec une précision en phase de $\pm 1/32$ en utilisant 16 cellules SHR dont l'horloge d'échantillonnage est à $448MHz$ [DCP⁺89]. L'avantage de ce synthétiseur est qu'il peut fonctionner avec des technologies à faibles coûts, comme la technologie CMOS $1,2\mu m$ [DCP⁺89]. De plus, le système est totalement numérique et donc approprié aux ADPLLs. De surcroît, la fréquence de sortie est contrôlée par un nombre fractionnaire, ce qui permet de maîtriser la résolution fréquentielle de la PLL.

1.4.3.3 Architecture de type "Flying Adder"

L'architecture "Flying Adder" [MX00, XY02, XY03, XLMP04] utilise une PLL munie d'un VCO générant plusieurs signaux d'horloge à la même fréquence (figure 1.33 [XY03]), décalés uniformément dans le temps, permettant d'obtenir une horloge polyphases [Cor04]. Cette PLL, munie du VCO, fonctionne à fréquence constante. Dans ces conditions, il n'y a pas de contrainte sur la bande passante de la PLL. La figure 1.34 [XLMP04] donne le schéma de l'architecture "Flying Adder". Dans ce circuit, $FREQ[32 : 0]$ est le mot de contrôle de la fréquence, Z est le signal de sortie à la fréquence désirée et $VCOOUT[30 : 0]$ correspond aux signaux polyphases fournis par le VCO. Le principe de cette architecture est d'utiliser plusieurs signaux d'horloge de fréquences identiques mais avec un déphasage constant entre signaux adjacents pour synthétiser la fréquence et la phase. Dans l'architecture "Flying Adder" (figure 1.34), une symétrie apparaît : le signal d'entrée $VCOOUT[30 : 0]$ prend deux chemins. Chaque chemin dispose d'un multiplexeur $32 \rightarrow 1$ qui est utilisé pour sélectionner l'un des signaux du VCO qui va commander

logicielles (filtre de boucle). Le détecteur de phase, le compteur et le DCO sont implantés en un seul bloc physique. Les multiplexeurs $32 \rightarrow 1$ représente les fonctions les plus critiques à implanter. En effet, les chemins allant des 32 entrées vers la sortie doivent avoir le même temps de propagation et le décodage des adresses doit être rapide pour assurer un bon fonctionnement.

Lors des tests, le VCO de la figure 1.33 utilise un diviseur par 8, un quartz ayant une fréquence de fonctionnement de $14,31818\text{ MHz}$ et 31 étages de délai. Cela entraîne une différence de phase de $0,282\text{ ns}$ entre deux phases adjacentes de l'horloge polyphase. La fréquence de sortie suit donc la relation suivante :

$$FREQ[32 : 0] = \frac{T}{0,282 \cdot 10^{-9}} \quad (1.20)$$

avec T , la période du signal à générer. Sachant que la partie fractionnaire est sur 27 bits, la résolution temporelle δT est alors de $0,282 \cdot 10^{-9} / 2^{27}$, soit une résolution temporelle de $2,1 \cdot 10^{-18}\text{ s}$. Ainsi, ce synthétiseur a l'avantage d'avoir une très grande résolution. L'inconvénient de cette ADPLL est d'utiliser une PLL pour la génération du bus d'horloge, qui est constituée d'éléments analogiques. Cette structure demanderait elle aussi beaucoup de ressources matérielles dans le cas de la mesure de paramètres relatifs au PCR pour chaque programme d'un flux.

1.5 Conclusion

Ce chapitre a permis de recenser quatre techniques de synchronisation sur des horodateurs. L'une des techniques s'appuie sur l'utilisation d'une méthode statistique de traitement du signal utilisant la régression linéaire sur les horodateurs d'entrée pour estimer la base de temps de l'encodeur. Cette technique de synchronisation donnerait de meilleures performances en terme de temps de réponse et de réduction de gigue comparativement à une PLL. Cependant, le modèle des DPLLs utilisés correspond à un modèle simplifié des DPLLs, qui sert à comparer leur performance et leur algorithme [Kax00]. De plus, la méthode de la régression linéaire requière beaucoup de ressources et un VCO externe lorsque le système est implanté matériellement. En effet, rien que le calcul de la pente dans la régression linéaire consomme quatre registres, six additionneurs, six multiplieurs et un diviseur alors qu'une PLL utilisant le moins de ressources matérielles et utilisant un flux de PCRs en entrée pour se synchroniser, requière deux registres et un soustracteur pour la comparaison de phase, et nécessite des largeurs de bus moins importantes que dans l'architecture de la LLR. Aussi, dans notre cas, la restitution de l'horloge de tous les programmes d'un flux DVB-T, dont le nombre peut aller jusqu'à dix, doit être effectuée en temps réel. De ce fait, il serait nécessaire d'utiliser autant de VCOs qu'il y a de programmes à traiter, ce qui serait une solution encombrante et très coûteuse en ressources matérielles.

La solution basée sur le dispositif de synchronisation, que l'on peut rencontrer dans les SoCs submicroniques, paraît à première vue intéressante. En effet, il est possible de s'en inspirer : comme les données d'entrée sont dans ce cas des PCRs, il est nécessaire de discrétiser la phase

en sortie du sélecteur et de comparer cette phase avec les PCR_s d'entrée pour réaliser la synchronisation. Le signal d'horloge local sera alors le signal en sortie de ce sélecteur. Or, de manière générale, dans ce type d'application, les signaux polyphases permettent de générer des phases multiples entre deux périodes du signal d'horloge d'entrée. Et, comme la sélection de la phase est remise à jour à chaque nouveau changement sur la sortie du détecteur de phase, qui se fait au bout de 10 *ms* dans le meilleur des cas, la fréquence moyenne de l'horloge produite ne serait modifiée que très légèrement, ce qui ne permettrait pas de couvrir la plage de fréquence requise. De plus, il manque à ce dispositif un ajustement dynamique de la phase et de la fréquence à l'aide d'un filtre, déterminée pour une erreur de position nulle, ce qui permettrait la mesure de gigue sur la sortie du comparateur.

Concernant la technique de synchronisation par suppression de gigue, la structure du dispositif de synchronisation a une forte corrélation avec celle d'une PLL. Aussi, cette structure n'est basée que sur un modèle. En effet, aucune solution concrète n'est proposée pour mettre en oeuvre ce modèle, qu'elle soit logicielle ou matérielle. Pour ce dernier, il aurait été intéressant de voir comment la génération d'horloge aurait été réalisée. Cependant, l'utilisation d'une technique de sur-échantillonnage est intéressante dans notre cas car le temps d'arrivée des PCR_s peut être variable et donc modifier le comportement dynamique du système de synchronisation qui sera utilisé. Ainsi, le sur-échantillonnage permettra d'avoir un comportement dynamique du système de synchronisation indépendant du débit d'entrée.

Dans le domaine de la génération d'horloge, plusieurs techniques existent et peuvent être classées en deux catégories : celles utilisant un oscillateur, qui peut être réalisé de manière analogique, numérique ou hybride, et celles réalisant une synthèse de fréquence. Lorsqu'il s'agit d'oscillateurs contrôlés numériquement (DCO), plusieurs techniques existent comme l'utilisation d'oscillateurs en anneau. Ces derniers sont surtout utilisés dans des réalisations microélectroniques figées. Cependant, cette technique accumule de la gigue et une incertitude dans une transition affecte toutes les transitions suivantes, et son effet persiste indéfiniment [FF00, HLL99, MJA01, MCL⁺00]. En ce qui concerne la génération d'horloge par synthèse de fréquence et de phase, des techniques très diverses existent comme la synthèse directe de fréquence, la synthèse de fréquence fractionnaire, la synthèse basée sur le décalage de phase (architecture PSDCO) et la synthèse de type "Flying Adder". Toutes ces techniques sont basées sur la génération de phases discrètes, ce qui signifie que la variation de phase issue de ces techniques n'est pas continue, ce qui introduit de la gigue qui peut influencer le comportement du système de synchronisation dans lequel le dispositif de synthèse serait éventuellement utilisé. Ainsi, la gigue de l'horloge doit être analysée dans le but d'estimer sa qualité [KII02, LPKK00, MY02, Meh02].

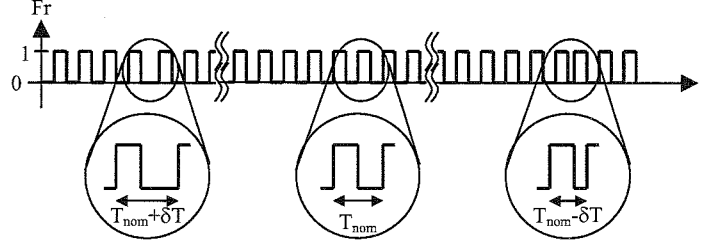
Chapitre 2

Description du système de mesure des paramètres relatifs au PCR utilisant une ADPLL

2.1 Introduction

L'objectif est de restituer l'horloge locale à 27 MHz d'un encodeur MPEG-2 à l'aide d'horodateurs appelés PCRs. Ces derniers sont affectés par de la gigue de réseau, ce qui introduit des erreurs relatives au PCR. Le système de mesure sera décrit et modélisé. Le cœur de ce système est une PLL innovante qui est totalement numérique et qui permet d'implanter autant de PLL qu'il y a de programmes dans le flux DVB-T.

Cette nouvelle PLL est une ADPLL utilisant un synthétiseur de fréquence modifiant la fréquence par une avance ou un retard de phase d'une horloge fonctionnant à une fréquence nominale. Le synthétiseur sera analysé en terme de gigue afin d'analyser ses performances et de vérifier s'il n'apporte pas de perturbations sur les mesures. Le filtre de boucle est ensuite étudié en assurant une erreur de phase nulle lorsque la PLL est synchronisée. L'étude de la stabilité de la PLL permet de déterminer les plages de valeurs possibles des coefficients du filtre. Puis, le comportement de l'ADPLL est analysé en fonction de sa période d'échantillonnage et des coefficients du filtre de boucle. Les filtres de mesure permettant l'extraction des paramètres relatifs au PCR sont ensuite présentés.


 FIG. 2.1 – Décalage de $\pm\delta T$ d'un signal de fréquence $1/T_{nom}$.

2.2 Principe du synthétiseur de fréquence proposé

2.2.1 Principe de fonctionnement et modélisation

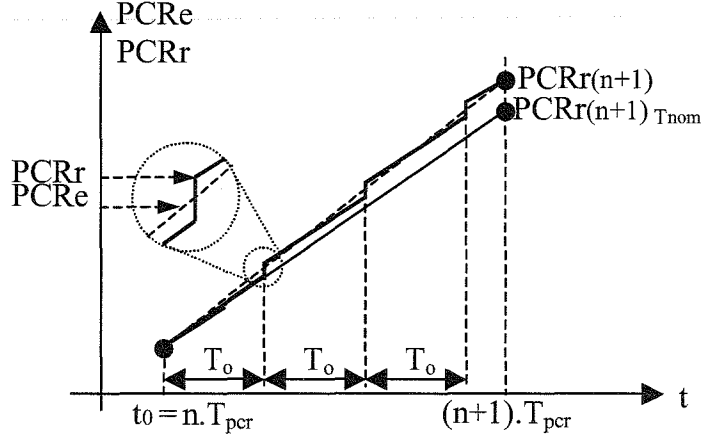
L'originalité du synthétiseur de fréquence proposé repose sur la génération d'un signal de fréquence centrale, subissant un retard ou une avance δT calibré (figure 2.1). La modification de la phase du signal est répartie de manière homogène au cours du temps. La fréquence moyenne du signal de sortie du synthétiseur de fréquence est modifiée en changeant la périodicité d'apparition des avances ou retards de phase. Le remise à jour de la fréquence se fait au bout d'une durée T_{PCR} , qui représente la durée d'arrivée de deux PCR_e successifs. La fréquence centrale est $F_{nom} = \frac{1}{T_{nom}}$.

Le synthétiseur de fréquence incrémente ensuite le compteur PCR_r , qui produit une phase discrète dont la variation est illustrée par la figure 2.2. Les termes $PCR_e(n)$ et $PCR_r(n)$ sont les phases échantillonnées de PCR_e et PCR_r par l'horloge de fréquence $F_{PCR} = \frac{1}{T_{PCR}}$. La phase échantillonnée de PCR_r doit égaler la phase PCR_e pour que l'horloge de fréquence moyenne $\bar{F}_r(n)$ (et de période $\bar{T}_r(n) = \frac{1}{\bar{F}_r(n)}$) soit synchronisée sur l'horloge de fréquence $F_e(n)$. $PCR_r(n+1)_{T_{nom}}$ est la valeur $PCR_r(n+1)$, obtenue lorsqu'il n'y a pas de décalage de phase. Pour décroître ou accroître la fréquence moyenne du synthétiseur de fréquence, la période T_{nom} est remplacée par les périodes $T_{nom} + \delta T$ ou $T_{nom} - \delta T$ respectivement. Pendant une durée T_{PCR} , il y a N_{nom} périodes de T_{nom} et $N_{nom+\delta}$ (or $N_{nom-\delta}$) périodes de $T_{nom} + \delta T$ (ou $T_{nom} - \delta T$). Ainsi, la période moyenne $\bar{T}_r$ peut être calculée à l'aide de l'équation 2.1 :

$$\bar{T}_r = \frac{T_{nom} \cdot N_{nom} + (T_{nom} \pm \delta T) \cdot N_{nom \pm \delta}}{N_{nom} + N_{nom \pm \delta}} \quad (2.1)$$

Pour déterminer l'architecture du synthétiseur de fréquence, il est nécessaire de connaître l'algorithme qui permet de calculer la durée T_o (figure 2.2), après laquelle il y a un décalage de $+\delta T$ ou $-\delta T$, en fonction de la différence entre le PCR_e reçu et le $PCR_r(n)$ estimé localement (notée PCR_error).

PCR_r et PCR_e sont le résultat d'une conversion temps/numérique exprimée par les équations


 FIG. 2.2 – Erreur de phase instantanée sur le PCR pendant une durée T_{PCR} .

tions 2.2 et 2.3 :

$$PCR_r = F_r \cdot (t - t_0) + PCR_0 \quad (2.2)$$

$$PCR_e = F_e \cdot (t - t_0) + PCR_0 \quad (2.3)$$

où t_0 correspond à l'arrivée du premier PCR. A cet instant t_0 , les deux PCRs sont identiques.

Des équations 2.2 et 2.3, l'erreur de PCR peut être calculée à l'aide de l'équation 2.4 :

$$PCR_error = PCR_e - PCR_r = (F_e - F_r) \cdot t \quad (2.4)$$

T_o est le temps après lequel la phase instantanée entre PCR_e et PCR_r doit être resynchronisée, c'est-à-dire lorsque l'erreur de phase (PCR_error) est plus grande que la résolution du synthétiseur de fréquence, qui est égale à $\frac{\delta T}{T_{nom}} (= \frac{1}{m})$. La compensation de phase est donc réalisée après un temps T_o donné par :

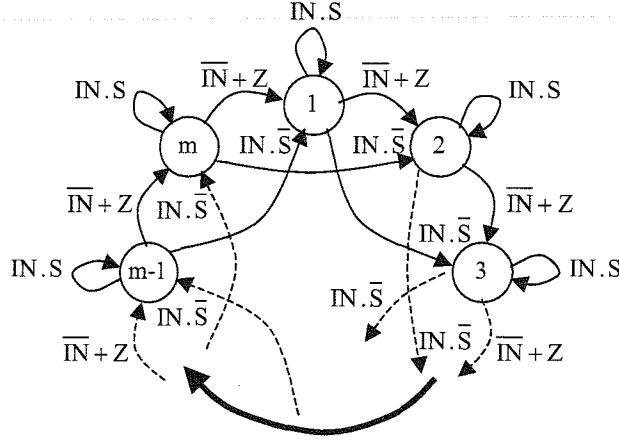
$$T_o = \frac{PCR_e - PCR_r}{F_e - F_r} = \frac{1}{m \cdot F_e - m \cdot F_r} \quad (2.5)$$

Des équations 2.4 et 2.5, il peut être démontré que l'intervalle de phase (délai ou avance) est fonction de l'erreur de phase PCR_error :

$$T_o = \frac{\delta T \cdot t}{T_{nom} \cdot |PCR_error|} \quad (2.6)$$

Une solution pour générer cette durée T_o est d'utiliser un compteur (générateur de rampe) fonctionnant à la fréquence $m \cdot F_{nom}$ et qui est mis à zéro après une durée T_o en comparant sa sortie avec un seuil TH proportionnel à T_o :

$$TH = T_o \times m \cdot F_{nom} = \frac{t \cdot F_{nom}}{|PCR_error|} \quad (2.7)$$


 FIG. 2.4 – Schéma de la FSM lorsque une horloge de référence $m.F_{nom}$ est utilisée.

La fonction de transfert du synthétiseur de fréquence est présentée en figure 2.5. La fréquence

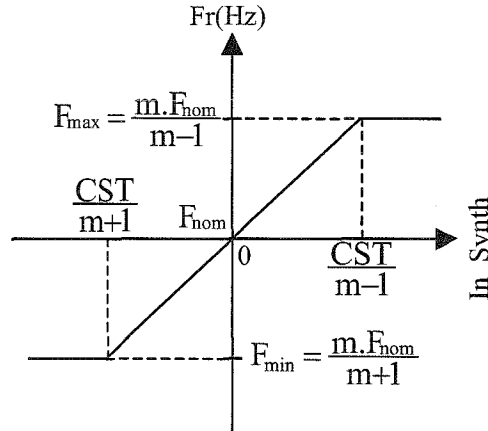


FIG. 2.5 – Fonction de transfert du synthétiseur de fréquence.

du synthétiseur évolue linéairement en fonction du mot de contrôle dans la plage de fréquence :

$$\frac{m.F_{nom}}{m+1} < F_r < \frac{m.F_{nom}}{m-1} \quad (2.11)$$

Il peut être noté que lorsque m augmente, cette plage de fréquence diminue. Soit CST égal à $m.F_{nom}$, la résolution du synthétiseur de fréquence devient $\frac{1}{m}$ (en Hz).

2.2.2 Gigue intrinsèque du synthétiseur

Le signal généré du synthétiseur de fréquence est étudié en terme de gigue. La gigue RMS J_{Synth} du synthétiseur peut être déduite de l'équation 2.1. L'équation 2.12 donne la gigue RMS

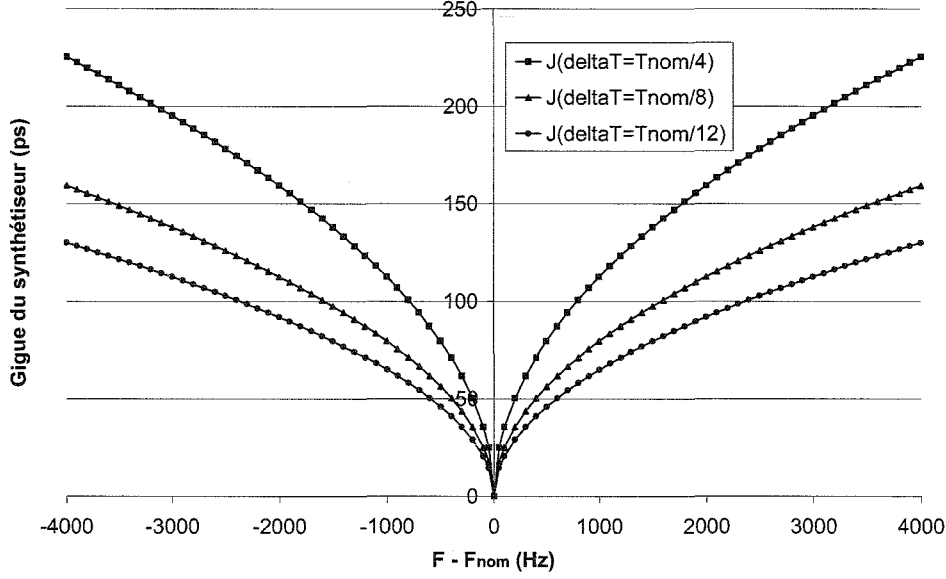


FIG. 2.6 – Gigue en fonction de la fréquence pour $m=\{4;8;12\}$.

J_{Synth} introduite par le synthétiseur en fonction de δT , équation qui ne tient pas compte de la gigue de l'horloge de référence utilisée. $\bar{T}$ est la période moyenne du signal que produit le synthétiseur.

$$J_{Synth} = \sqrt{\frac{(T_{nom} - \bar{T})^2 \cdot N_{nom} + ((T_{nom} \pm \delta T) - \bar{T})^2 \cdot N_{nom \pm \delta}}{F_{nom} \cdot T_{PCR}}} \quad (2.12)$$

La figure 2.6 présente l'évolution de la gigue RMS intrinsèque au synthétiseur de fréquence, autour de la fréquence F_{nom} et pour plusieurs δT ($\delta T = \{\frac{T_{nom}}{m}$ avec $m = 4, 8$ et $12\}$). Ces courbes permettent de choisir la valeur de δT lorsque une limite de gigue RMS est spécifiée dans une plage de fréquence. La valeur minimale de m est 3. Le choix de m est un compromis entre la gigue générée et la fréquence de référence du synthétiseur. Le choix de la fréquence de référence va aussi avoir un effet sur les contraintes de temps et sur les ressources matérielles du synthétiseur. Comme précisé dans les lignes précédentes, le modèle ne prend pas en compte la gigue de l'horloge de référence. Pour calculer la gigue totale RMS J_{Synth_tot} du synthétiseur, l'équation 2.13 doit être utilisée. Concernant la gigue crête-à-crête intrinsèque au synthétiseur, elle est directement donnée par le retard ou l'avance δT . Lorsque la fréquence générée par le synthétiseur est inférieure à F_{nom} , la gigue crête-à-crête est de $+\delta T$, et lorsqu'elle est supérieure à F_{nom} , la gigue crête-à-crête est de $-\delta T$.

$$J_{Synth_tot}^2 = J_{m.Fnom}^2 + J_{Synth}^2 \quad (2.13)$$

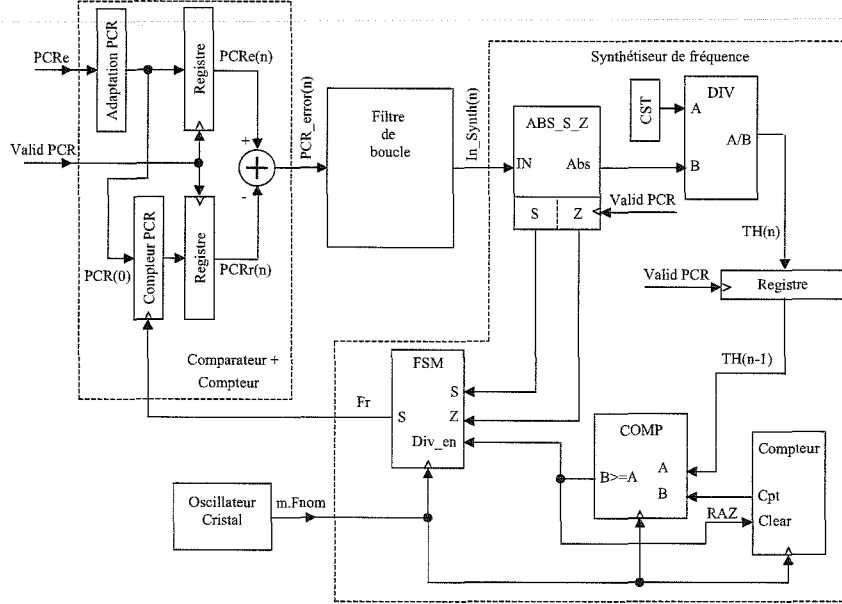


FIG. 2.7 – Schéma intermédiaire de l'ADPLL.

2.3 Le filtre de boucle

2.3.1 Schéma intermédiaire de l'ADPLL

Le schéma de la figure 2.7 est un schéma intermédiaire qui présente tous les éléments de l'ADPLL, hormis la structure du filtre. Cette dernière reste à déterminer en fonction des contraintes de phase et de fréquence imposées.

2.3.2 Conditions de phase et de fréquence sur l'ADPLL : détermination de la structure du correcteur

Les valeurs de sortie du filtre sont modifiées à chaque nouvelle arrivée des échantillons PCR_e . Ainsi, la transformée en z est utilisée pour déterminer la structure du filtre.

$$PCR_error(z) = PCR_e(z) - PCR_r(z) \quad (2.14)$$

L'expression de $PCR_r(z)$ est déduite de l'équation 7 dont la transformée en z est :

$$PCR_r(z) = PCR_r(z).z^{-1} + T_{PCR}F_r(z).z^{-1} \quad (2.15)$$

$PCR_r(z)$ est aussi égal à :

$$PCR_r(z) = \frac{F_r(z)T_{PCR}}{z - 1} \quad (2.16)$$

Il en va de même pour $PCR_e(z)$:

$$PCR_e(z) = \frac{F_e(z)T_{PCR}}{z-1} \quad (2.17)$$

Remplaçons $F_r(z)$ par son expression (équation 2.10) dans l'équation 2.16 :

$$PCR_r(z) = \frac{F_{nom}T_{PCR}}{z-1} \times \left(1 + \frac{In_Synth(z)}{CST.z}\right) \quad (2.18)$$

L'expression entre la sortie $In_Synth(z)$ et l'entrée $PCR_error(z)$ du filtre de boucle est donnée par l'équation 2.19 :

$$In_Synth(z) = PCR_error(z) \times F(z) \quad (2.19)$$

où $F(z)$ est la fonction de transfert du filtre.

Remplaçons cette expression de $In_Synth(z)$ dans l'équation 2.18 :

$$PCR_r(z) = \frac{F_{nom}T_{PCR}}{z-1} \times \left(1 + \frac{PCR_error(z) \times F(z)}{CST.z}\right) \quad (2.20)$$

Remplaçons maintenant $PCR_r(z)$ par son expression dans l'équation 2.14 :

$$PCR_error(z) = PCR_e(z) - \frac{F_{nom}T_{PCR}}{z-1} \times \left(1 + \frac{PCR_error(z) \times F(z)}{CST.z}\right) \quad (2.21)$$

En isolant le terme $PCR_error(z)$, l'équation 2.22 est obtenue :

$$PCR_error(z) = \frac{(z-1)z \times PCR_e(z)}{(z-1)z + F_{nom}T_{PCR} \frac{F(z)}{CST}} - \frac{z \times F_{nom}T_{PCR}}{(z-1)z + F_{nom}T_{PCR} \frac{F(z)}{CST}} \quad (2.22)$$

Cette expression est constituée d'une partie statique comprenant $z \times F_{nom}T_{PCR}$ au numérateur et d'une partie dynamique dépendant de la valeur d'entrée $PCR_e(z)$. Cette partie dynamique sera prise en compte pour la détermination de la fonction de transfert du filtre de boucle.

La structure de ce filtre est déterminée pour une erreur de position nulle. En effet, une erreur de phase nulle sera spécifiée pour un échelon de phase et une erreur de fréquence nulle sera spécifiée pour un échelon de fréquence.

Un échelon de phase dans le domaine temporel $\Gamma(n)$ a comme transformée en z :

$$PCR_e(n) = \Delta PCR.\Gamma(n) \Rightarrow PCR_e(z) = \Delta PCR \times \frac{z}{z-1} \quad (2.23)$$

où ΔPCR est l'amplitude de l'échelon d'entrée.

L'erreur de phase pour un échelon de phase en entrée est donnée par l'équation 2.24 :

$$PCR_error(z) = \Delta PCR \times \frac{z^2}{(z-1)z + F_{nom}T_{PCR} \frac{F(z)}{CST}} \quad (2.24)$$

D'après le théorème de la valeur finale :

$$\lim_{n \rightarrow +\infty} PCR_error(n) = \lim_{z \rightarrow 1} (z-1) \times PCR_error(z) \quad (2.25)$$

$$\lim_{n \rightarrow +\infty} PCR_error(n) = \lim_{z \rightarrow 1} \Delta PCR \times \frac{(z-1).z^2}{(z-1)z + F_{nom}T_{PCR}\frac{F(z)}{CST}} = 0 \quad (2.26)$$

Pour que l'erreur de phase du PCR soit nulle, il faut que $F(1) \neq 0$, comme par exemple $F(z) = G_1$. Ainsi, la valeur finale de l'erreur de phase sera égale à zéro pour un échelon d'entrée si la fonction de transfert du filtre $F(z)$ a un terme proportionnel.

Un échelon de fréquence dans le domaine temporel correspond à une variation de phase égale à :

$$PCR_e(n) = \Delta PCR.n.\Gamma(n) \quad (2.27)$$

et a comme transformée en z :

$$PCR_e(z) = \Delta PCR \times \frac{z}{(z-1)^2} \quad (2.28)$$

L'erreur de phase pour un échelon de fréquence en entrée est donnée par l'équation 2.24 :

$$PCR_error(z) = \Delta PCR \times \frac{z^2}{(z-1)((z-1)z + F_{nom}T_{PCR}\frac{F(z)}{CST})} \quad (2.29)$$

D'après le théorème de la valeur finale :

$$\lim_{n \rightarrow +\infty} PCR_error(n) = \lim_{z \rightarrow 1} (z-1) \times PCR_error(z) \quad (2.30)$$

$$\lim_{n \rightarrow +\infty} PCR_error(n) = \lim_{z \rightarrow 1} \Delta PCR \times \frac{z^2}{(z-1)z + F_{nom}T_{PCR}\frac{F(z)}{CST}} = 0 \quad (2.31)$$

Pour que l'erreur de fréquence soit nulle, il faudrait que $\frac{1}{F(1)} = 0$, comme par exemple $F(z) = G_2 \times \frac{z}{z-1}$. Ainsi, la valeur finale de l'erreur de fréquence sera égale à zéro pour un échelon en entrée si la fonction de transfert du filtre $F(z)$ a un terme intégral.

La fonction de transfert $F(z)$ du filtre de boucle doit alors avoir un terme proportionnel G_1 et un terme intégral G_2 , et son expression est donnée par l'équation 2.32 :

$$F(z) = \frac{In_Synth(z)}{PCR_error(z)} = G_1 + G_2 \times \frac{z}{z-1} \quad (2.32)$$

que l'on peut encore écrire :

$$F(z) = G_1 + G_2 \times \frac{1}{1-z^{-1}} \quad (2.33)$$

De l'équation 2.33, on en déduit le schéma structurel (figure 2.8) du correcteur PI (Proportionnel-Intégral).

2.4 Etude de la stabilité de l'ADPLL

2.4.1 Expression générale de la fonction de transfert

Pour assurer le bon fonctionnement de l'ADPLL, une étude de stabilité doit être effectuée. Cette étude permettra de déterminer les plages de valeurs possibles des paramètres de l'ADPLL (G_1 , G_2 et CST).

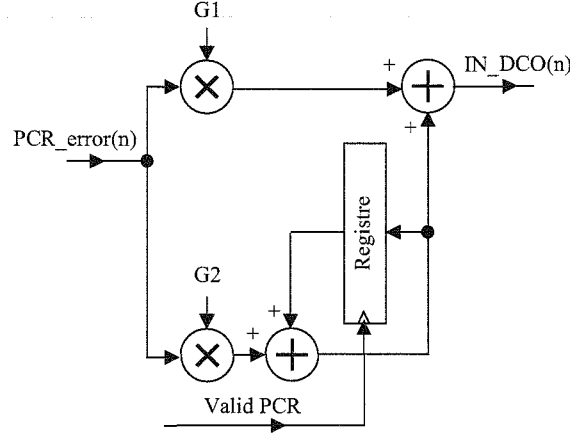


FIG. 2.8 – Schéma du correcteur PI de l'ADPLL.

La fonction de transfert $H(z)$ de l'ADPLL dans le domaine des transformées en z est déduite des équations 2.14, 2.16, 2.17, 2.22 et 2.32. La forme de la fonction de transfert $H(z)$ est donnée par l'expression 2.34 qui ne prendra en compte que la partie dynamique de l'équation 2.22. Cette fonction de transfert montre que l'ADPLL est un système numérique du troisième ordre.

$$H(z) = H_0 \cdot \frac{a_1 \cdot z + a_0}{z^3 + b_2 \cdot z^2 + b_1 \cdot z + b_0} \quad (2.34)$$

avec :

$$H_0 = \frac{F_{nom} \cdot T_{PCR}}{CST} \quad (2.35)$$

$$a_0 = -G_1 \quad (2.36)$$

$$a_1 = G_1 + G_2 \quad (2.37)$$

$$b_0 = -\frac{F_{nom} \cdot T_{PCR} \cdot G_1}{CST} \quad (2.38)$$

$$b_1 = 1 + \frac{F_{nom} \cdot T_{PCR}}{CST} \cdot (G_1 + G_2) \quad (2.39)$$

$$b_2 = -2 \quad (2.40)$$

La stabilité d'un système numérique est obtenue si tous les pôles de sa fonction de transfert dans le domaine des transformées en z sont à l'intérieur du cercle unitaire ($|z| = 1$). Or, ces pôles dépendent du temps d'arrivée des PCRs (T_{PCR}), ce qui peut entraîner le déplacement des pôles en dehors du cercle unitaire et donc créer une instabilité.

2.4.2 Les conditions de stabilité

Comme le système est du troisième ordre, sa fonction de transfert $H(z)$ est composée de trois pôles. $H(z)$ peut donc s'écrire ainsi :

$$H(z) = H_0 \cdot \frac{z - z_1}{(z - p_1)(z - p_2)(z - p_3)} \quad (2.41)$$

Le calcul de ces pôles passe par le calcul de deux termes intermédiaires pour des raisons de simplification d'écriture des équations. Ces termes appelés u et v sont donnés par les équations 2.42 et 2.43 :

$$u = \sqrt[3]{-(2b_2^2 - 9b_1) \cdot \frac{b_2}{54} - \frac{b_0}{2} + \sqrt{[(2b_2^2 - 9b_1) \cdot \frac{b_2}{54} + \frac{b_0}{2}]^2 + \frac{1}{27} \cdot (b_1 - \frac{b_2^2}{3})^3}} \quad (2.42)$$

$$v = \sqrt[3]{-(2b_2^2 - 9b_1) \cdot \frac{b_2}{54} - \frac{b_0}{2} - \sqrt{[(2b_2^2 - 9b_1) \cdot \frac{b_2}{54} + \frac{b_0}{2}]^2 + \frac{1}{27} \cdot (b_1 - \frac{b_2^2}{3})^3}} \quad (2.43)$$

Ces termes u et v permettent ensuite de calculer les pôles de l'ADPLL, qui sont donnés par les expressions suivantes :

$$p_1 = u + v - \frac{b_2}{3} \quad (2.44)$$

$$p_2 = \left\{ -\frac{u}{2} - \frac{v}{2} - \frac{b_2}{3} \right\} + \frac{1}{2}i\sqrt{3} \cdot \{u - v\} \quad (2.45)$$

$$p_3 = \left\{ -\frac{u}{2} - \frac{v}{2} - \frac{b_2}{3} \right\} - \frac{1}{2}i\sqrt{3} \cdot \{u - v\} \quad (2.46)$$

La détermination des conditions de stabilité (pôles à l'intérieur du cercle unitaire) passe par des équations difficiles à résoudre, comme le montre les équations présentées ci-dessus. Cette difficulté peut être surmontée en utilisant le critère de Jury pour un polynôme du troisième ordre. L'annexe B présente le détail de l'étude de stabilité. L'inégalité suivante en est déduite de cette annexe :

$$0 < G_2 < \frac{G_1}{1 - \frac{G_2}{G_1}} < \frac{CST}{F_{nom} \cdot T_{PCR}} \quad (2.47)$$

Maintenant que les critères de stabilité sont définis, il est possible de positionner les pôles et les zéros dans le plan z pour un système stable. Le module et l'argument de la fonction de transfert de l'ADPLL sont déterminés à partir de l'équation 2.34. Cette équation peut encore s'écrire :

$$H(z) = H_0 \cdot \frac{a_1 \cdot z^{-2} + a_0 \cdot z^{-3}}{1 + b_2 \cdot z^{-1} + b_1 \cdot z^{-2} + b_0 \cdot z^{-3}} \quad (2.48)$$

Pour obtenir la fonction de transfert de l'ADPLL en fonction de la fréquence, le changement de variable suivant est appliqué :

$$z^{-1} \rightarrow e^{-i \cdot \theta} = \cos \theta - i \cdot \sin \theta \quad (2.49)$$

où θ vaut :

$$\theta = 2\pi f.T_{PCR} \quad (2.50)$$

Ainsi, la fonction de transfert de l'ADPLL en fonction de la fréquence est donnée par :

$$\underline{H(f)} = \frac{a_1 \cdot \cos 2\theta + a_0 \cdot \cos 3\theta - i \cdot (a_1 \cdot \sin 2\theta + a_0 \cdot \sin 3\theta)}{1 + b_2 \cdot \cos \theta + b_1 \cdot \cos 2\theta + b_0 \cdot \cos 3\theta - i \cdot (b_2 \cdot \sin \theta + b_1 \cdot \sin 2\theta + b_0 \cdot \sin 3\theta)} \quad (2.51)$$

A partir de cette expression, le module et la phase sont calculés :

$$|\underline{H(f)}| = \frac{\sqrt{(a_1 \cdot \cos 2\theta + a_0 \cdot \cos 3\theta)^2 + (a_1 \cdot \sin 2\theta + a_0 \cdot \sin 3\theta)^2}}{\sqrt{(1 + b_2 \cdot \cos \theta + b_1 \cdot \cos 2\theta + b_0 \cdot \cos 3\theta)^2 + (b_2 \cdot \sin \theta + b_1 \cdot \sin 2\theta + b_0 \cdot \sin 3\theta)^2}} \quad (2.52)$$

$$\text{Arg}(\underline{H(f)}) = -\tan^{-1} \frac{a_1 \cdot \sin 2\theta + a_0 \cdot \sin 3\theta}{a_1 \cdot \cos 2\theta + a_0 \cdot \cos 3\theta} + \tan^{-1} \frac{b_2 \cdot \sin \theta + b_1 \cdot \sin 2\theta + b_0 \cdot \sin 3\theta}{1 + b_2 \cdot \cos \theta + b_1 \cdot \cos 2\theta + b_0 \cdot \cos 3\theta} \quad (2.53)$$

2.5 Etude comportementale de l'ADPLL

La figure 2.9 donne la représentation des pôles et du zéro de $H(z)$ et du module et de l'argument de la fonction $\underline{H(f)}$ pour T_{PCR} égal à 10 ms, 40 ms et 100 ms. Les valeurs 10 ms et 100 ms correspondent aux durées minimale et maximale de T_{PCR} , alors que la valeur 40 ms représente la durée nominale de T_{PCR} . Les autres paramètres du système, qui satisfont les conditions de stabilité, ont les valeurs suivantes : $G_1 = 20$, $G_2 = 1$, $F_{nom} = 27\text{MHz}$ et $CST = 108.10^6$. La représentation dans le plan complexe montre que le zéro reste constant lorsque la valeur de T_{PCR} est modifiée alors que les pôles se déplacent dans le plan complexe, ce qui entraîne la modification dynamique des conditions de stabilité en fonction de la période d'échantillonnage T_{PCR} . On peut aussi constater ce phénomène sur les courbes de module et d'argument de $\underline{H(f)}$ (figure 2.9). Les représentations dans le domaine fréquentiel se font sur une plage de fréquence allant de 0 à $\frac{F_{PCR}}{2}$. En fonction de T_{PCR} , le gain maximal de $|\underline{H(f)}|$ et la fréquence de résonance sont modifiés, ce qui change dynamiquement la bande passante de l'ADPLL.

L'étude dans le domaine temporel est effectuée en appliquant un échelon de fréquence de +800 Hz. La figure 2.10 donne la courbe de réponse en fréquence de l'ADPLL pour T_{PCR} égal à 10 ms, 40 ms et 100 ms. On peut noter que lorsque T_{PCR} augmente, le temps de réponse normalisé à $\pm 5\%$ diminue. Le temps de réponse normalisé correspond au nombre N de PCR reçu pour être dans la plage de $\pm 5\%$ de la valeur finale. Chacune des réponses en fréquence de la figure 2.10 présente un dépassement plus ou moins important, ce qui était prévisible car chacune des courbes de gain décrites ci-dessus possède une pseudo résonance.

Dans la réalité et dans la majorité des cas, la valeur de T_{PCR} se situe autour de sa valeur nominale, à savoir $T_{PCR} = 40\text{ ms}$. Pour que le dépassement soit le moins important, il faut étudier différentes réponses de l'ADPLL en fonction des coefficients G_1 et G_2 . G_1 correspond au coefficient proportionnel du filtre de boucle et G_2 représente le coefficient intégral. En gardant

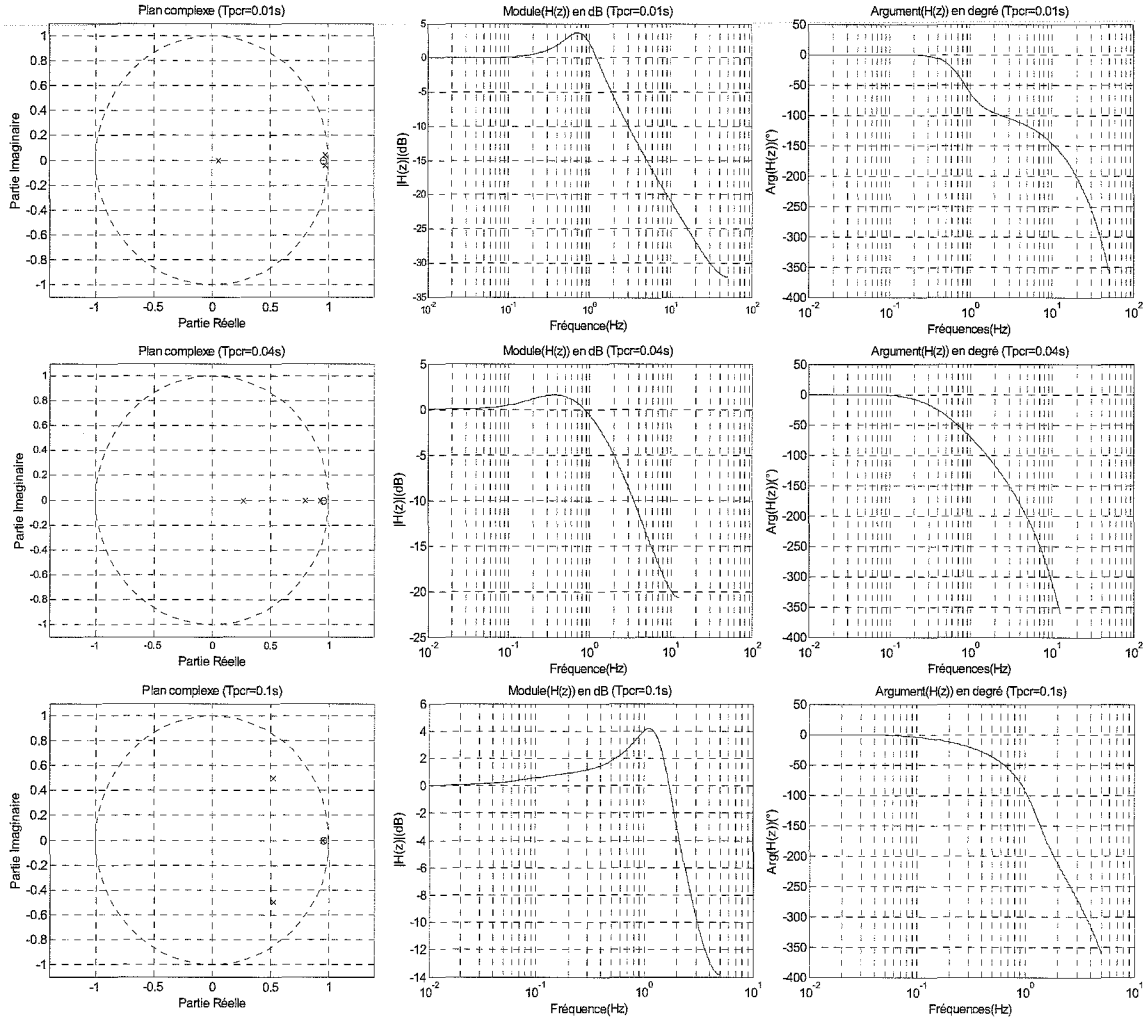


FIG. 2.9 – Pôles et zéros, module et argument de la fonction de transfert de l'ADPLL pour $T_{PCR} = \{10ms; 40ms; 100ms\}$.

toujours $G_2 = 1$, les courbes de réponse de la figure 2.11 sont obtenues. Pour une plage de G_1 allant de 20 à 60, le dépassement le moins important est obtenu pour $G_1 = 30$ précisément. Avec cette valeur de G_1 , le temps de réponse normalisé correspond au temps mis pour recevoir 29 échantillons. Le temps de réponse le plus faible est donné pour $G_1 = 40$ et est égal à la durée correspondant à la réception de 9 échantillons. Au-delà de la valeur de $G_1 = 40$, la courbe de réponse devient pseudo-périodique. Ainsi, G_1 permet d'agir sur le temps de convergence.

Le rôle du coefficient G_2 est analysé en prenant pour G_2 les valeurs 5, 1 et 0,2. G_1 sera fixé à la valeur où il y avait le moins de dépassement dans le cas précédent, à savoir pour $G_1 = 30$. La figure 2.12 illustre la réponse indicielle de l'ADPLL pour ces différentes valeurs de G_2 . On

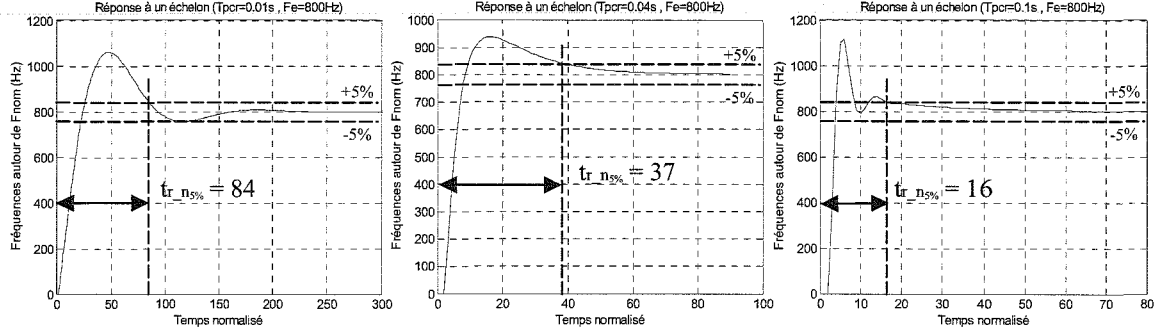


FIG. 2.10 – Réponses de l'ADPLL à un échelon de fréquence pour $T_{PCR} = \{10ms; 40ms; 100ms\}$.

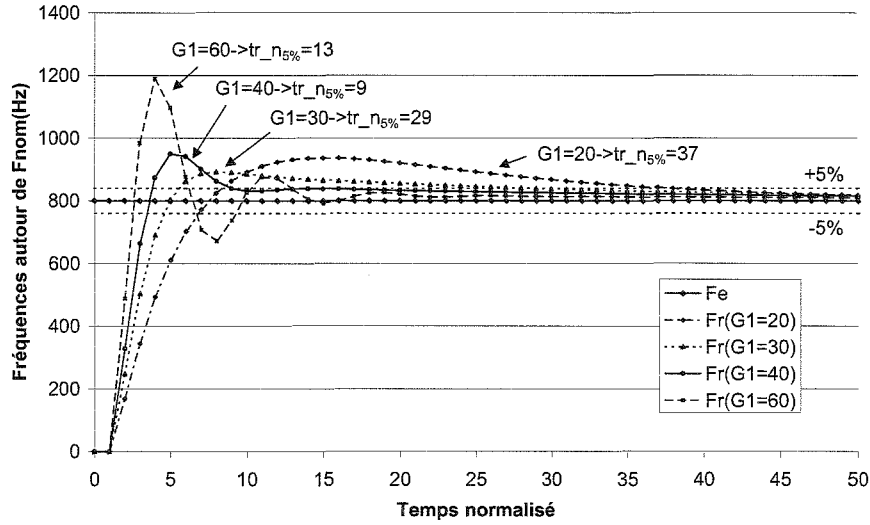


FIG. 2.11 – Réponses de l'ADPLL standard à un échelon de fréquence pour $T_{PCR} = 40 \text{ ms}$ et $G_1 = \{20; 30; 40; 60\}$.

peut constater que l'amplitude du dépassement est modifiée en fonction du coefficient intégral G_2 . Ainsi, plus ce dernier diminue, plus le dépassement est faible.

Pour que la courbe de réponse à un échelon ne présente plus de dépassement, l'étude du comportement de l'ADPLL montre que les conditions suivantes doivent être réunies : $0 < G_2 \ll 1$ et $\frac{G_1}{G_2} \gg 1000$. La figure 2.13 présente la réponse à un échelon de l'ADPLL et les représentations de gain et de phase pour $T_{PCR} = 40 \text{ ms}$, $G_1 = 10$ et $G_2 = 0,001$. La réponse à un échelon sans dépassement donne un temps de réponse de 28 échantillons. En ce qui concerne le gain, l'atténuation maximale se trouvant à $\frac{F_{PCR}}{2}$ est de 27 dB. Pour augmenter l'atténuation dans la bande rejetée, il faut diminuer G_1 . Pour conserver la condition $\frac{G_1}{G_2} \gg 1000$, G_2 doit donc aussi être diminuée.

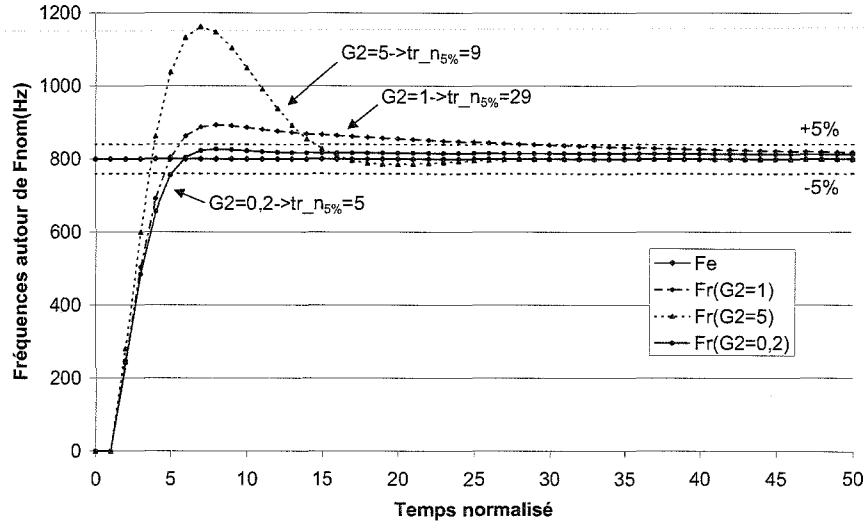


FIG. 2.12 – Réponses de l'ADPLL "standard" à un échelon de fréquence pour $T_{PCR} = 40 \text{ ms}$ et $G_2 = \{5; 1; 0, 2\}$.

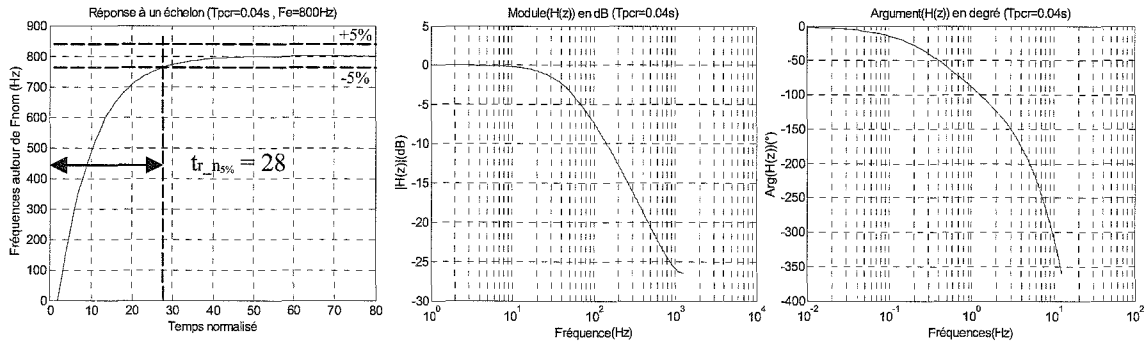


FIG. 2.13 – Réponses de l'ADPLL à un échelon de fréquence et courbes de gain et de phase pour $T_{PCR} = 40 \text{ ms}$, $G_1 = 10$ et $G_2 = 0,001$.

En choisissant $G_1 = 2$ et $G_2 = 0,0001$, les courbes de la figure 2.14 sont obtenues. L'atténuation maximale est maintenant de 40 dB à $\frac{F_{PCR}}{2}$, soit une atténuation en dB de 48 % de plus que dans le cas précédent. Cette atténuation plus importante est obtenue grâce à une bande passante plus étroite. Cela a aussi une répercussion sur le temps de réponse qui est de 143 échantillons de PCR, ce qui va entraîner une convergence plus lente de l'ADPLL. Dans le cas de l'application de la mesure des paramètres relatifs au PCR, cette lenteur est plutôt favorable car cela permettra d'obtenir une meilleur réjection de la gigue et une meilleur précision sur les paramètres à analyser.

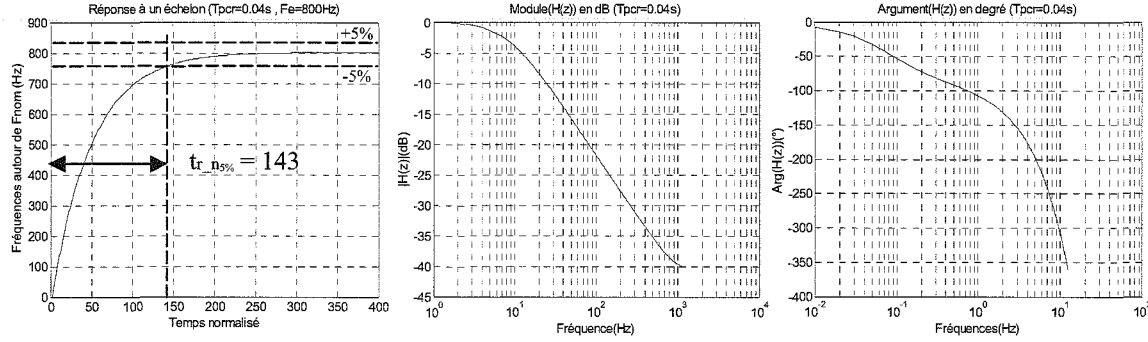


FIG. 2.14 – Réponses de l'ADPLL à un échelon de fréquence et courbes de gain et de phase pour $T_{PCR} = 40 \text{ ms}$, $G_1 = 2$ et $G_2 = 0,0001$.

2.6 Adaptation de la PLL aux temps d'arrivée des PCR

2.6.1 Origine de la variation du temps d'arrivée des PCR et de la modification du comportement dynamique de l'ADPLL

La variation du temps d'arrivée des PCR est due à différentes raisons. Ces variations peuvent être dues à l'encapsulation des PCR dans les trames ou le multiplexage des trames des différents programmes dans le flux. Cela entraîne des variations de délai sur les PCR. La nature asynchrone du réseau qui affecte le temps d'arrivée des trames par de la gigue de réseau est une source de variation aléatoire du temps d'arrivée des PCR. Par contre, lorsque l'on souhaite se synchroniser sur un autre programme, cela ne représente pas une source de variation du temps d'arrivée des PCR, car une remise à zéro de l'ADPLL est nécessaire dans ce cas, qui va précharger le PCR du programme considéré dans les registres contenant les valeurs de PCR_e et PCR_r , qui est une opération qui ne tient pas compte du temps d'arrivée.

L'étude de la stabilité de l'ADPLL montre que les pôles de sa fonction de transfert se déplace en fonction de la période d'échantillonnage T_{PCR} . Cela engendre une modification du comportement dynamique de la PLL, et donc une modification de la stabilité de l'ADPLL au cours du temps, ce qui est visible sur les courbes de réponse à un échelon. De plus, lorsque la période T_{PCR} sort d'une certaine plage de valeurs, la PLL peut devenir instable.

La dépendance des pôles aux temps d'arrivée des PCR est due au fait que l'erreur de PCR (PCR_{error}) dépend de T_{PCR} comme le montre l'équation 2.54 et la figure 2.15.

$$PCR_{error}(z) = PCR_e(z) - PCR_r(z) = \frac{T_{PCR}}{z - 1} \cdot (F_e(z) - F_r(z)) \quad (2.54)$$

Cette figure présente l'évolution de l'erreur de PCR lorsque l'erreur de fréquence entre F_e et F_r est constante. Dans ce cas, lorsque le temps T_{PCR} double, l'erreur de PCR est deux fois plus grande. Ainsi, une solution assurant la stabilité de la PLL consiste à normaliser l'erreur de

FIG. 2.15 – Erreur de PCR en fonction du temps.

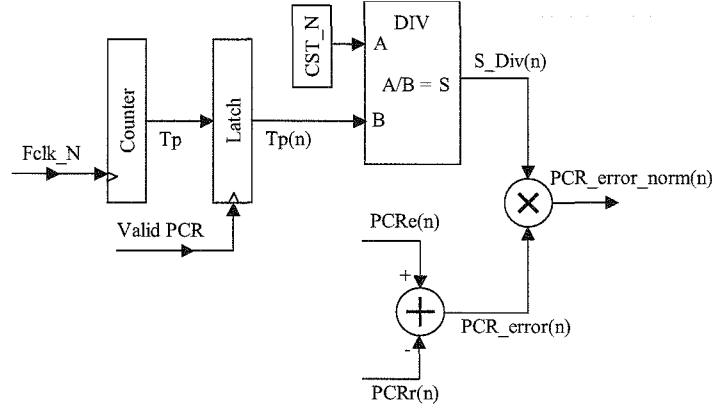


FIG. 2.16 – Schéma de la fonction réalisant la normalisation du PCR.

Dans ces conditions, les coefficients de la fonction de transfert de l'ADPLL (équation 2.34) sont les suivants :

$$H_0 = \frac{F_{nom}}{F_{clk_N}} \cdot \frac{CST_N}{CST} \quad (2.57)$$

$$a_0 = -G_1 \quad (2.58)$$

$$a_1 = G_1 + G_2 \quad (2.59)$$

$$b_0 = -\frac{F_{nom}}{F_{clk_N}} \cdot \frac{CST_N}{CST} \cdot G_1 \quad (2.60)$$

$$b_1 = 1 + \frac{F_{nom}}{F_{clk_N}} \cdot \frac{CST_N}{CST} \cdot (G_1 + G_2) \quad (2.61)$$

$$b_2 = -2 \quad (2.62)$$

D'après ces coefficients, la réponse du système ne dépend plus de T_{PCR} , et cela malgré qu'il soit cadencé à un rythme T_{PCR} variable.

L'étude de stabilité de l'ADPLL aboutit sur l'inégalité ci-dessous :

$$0 < G_2 < \frac{G_1}{1 - \frac{G_2}{G_1}} < \frac{F_{clk_N} \cdot CST}{F_{nom} \cdot CST_N} \quad (2.63)$$

2.6.3 Obtention d'une bande passante constante de l'ADPLL par une technique de sur-échantillonnage

Une solution pour rendre la bande passante de l'ADPLL indépendante du temps d'arrivée des PCRs est l'utilisation d'une technique de sur-échantillonnage. De cette manière, l'horloge d'échantillonnage de l'ADPLL sera constante, mais cela entraîne la mise en place d'une technique d'interpolation des PCR_e afin de fournir à chaque front de l'horloge de sur-échantillonnage des PCRs adaptés. La figure 2.17(a) donne un exemple de PCR_e reçus du flux et la figure 2.17(b) présente le résultat de l'interpolation des PCRs (PCR_{e_SE}) associés à l'horloge de sur-échantillonnage

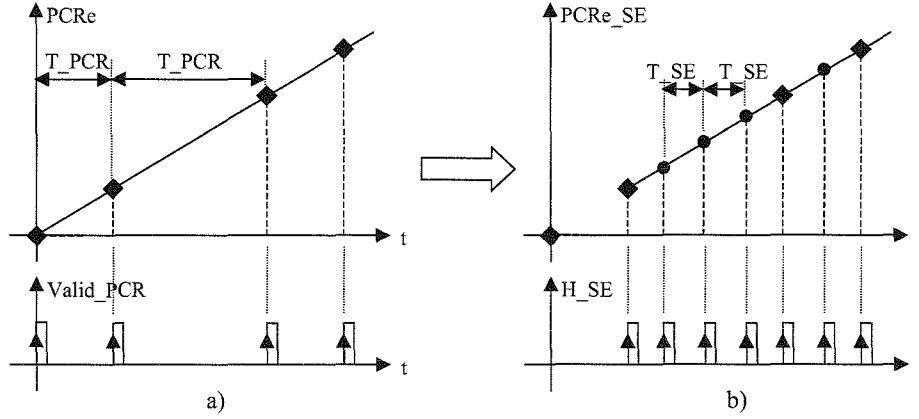
H_{SE} .

FIG. 2.17 – Chronogramme illustrant le principe du sur-échantillonnage appliqué au PCR. PCRs reçus avec son signal de validation (a), interpolation des PCRs et horloge de sur-échantillonnage associée (b).

Pour parvenir à ce résultat, la structure de la figure 2.18 est utilisée. Cette structure reçoit le signal PCR_e et le signal de validation $Valid_PCR$ associé, ainsi qu'une horloge de comptage H_{Idx} , de période T_{Idx} et synchrone avec le signal $Valid_PCR$, permettant de mesurer la durée séparant l'arrivée de deux PCR_e successifs, noté Idx_Time . Ensuite, un produit en croix, prenant en compte l'écart entre les deux PCR_e les plus récents et la durée les séparant, est effectué pour obtenir l'écart entre deux PCR_e_SE successifs. Cet écart est ensuite accumulé au rythme de l'horloge de sur-échantillonnage, qui est obtenue en divisant la fréquence de l'horloge H_{Idx} par un facteur K , qui est pris en compte dans le calcul de l'écart de deux PCR_e_SE successifs.

Concernant le comportement de l'ADPLL, le modèle de la section 2.5 est pris en compte, car l'effet de l'interpolation des PCR_e ne doit pas avoir d'influence sur le comportement de l'ADPLL. Deux changements sont cependant à faire : le premier consiste à enlever la fonction d'adaptation du PCR car elle est déjà présente dans la structure de l'interpolateur. Le second changement se situe au niveau du filtre de boucle, où les coefficients doivent être adaptés à une fréquence d'échantillonnage plus élevée. Comme la fréquence d'échantillonnage est constante, la bande passante sera toujours la même quelque soit la durée T_{PCR} . Ainsi, si les coefficients du filtre de boucle sont bien choisis, l'ADPLL sera toujours stable.

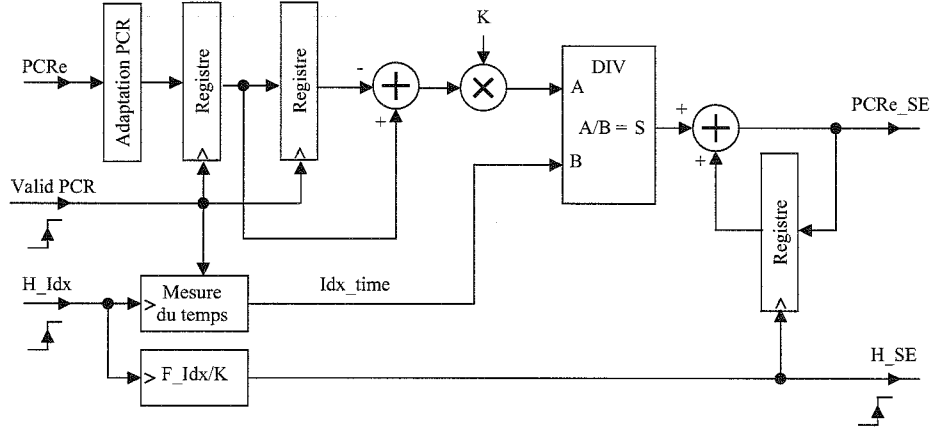


FIG. 2.18 – Schéma de la structure réalisant l'interpolation du PCR.

Profils	Fréquence de coupure
<i>MGF1</i>	10 mHz
<i>MGF2</i>	100 mHz
<i>MGF3</i>	1 Hz
<i>MGF4</i>	Défini en manufacture

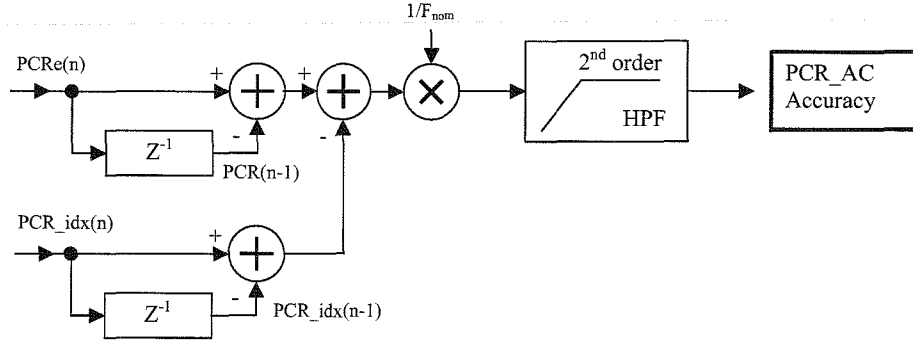
TAB. 2.1 – Profils des filtres de mesure.

2.7 Extraction des paramètres relatifs au PCR

2.7.1 Les filtres de mesure

Des filtres de mesure sont utilisés pour extraire les différents paramètres relatifs au PCR. Ils peuvent avoir quatre configurations suivant la fréquence de coupure choisie. Chaque fréquence de coupure est référencée dans un profil [ETS01]. Le tableau 2.1 présente les différents profils qui vont de *MGF1* à *MGF4*. Ainsi, pour chaque filtre, l'annexe C fournira les valeurs des coefficients pour une périodicité nominale d'arrivée des PCRs, c'est-à-dire pour $T_{PCR} = 40ms$, et pour le système utilisant l'ADPLL en sur-échantillonnage (où $T_{SE} = 8ms$).

Les profils allant de *MGF1* à *MGF3* sont prédéfinis alors que le profil *MGF4* est à définir par le manufacturier. Le choix du profil repose sur un dilemme entre temps de réponse et précision des mesures. Dans certain cas, le profil *MGF3* ou *MGF2* est suffisant pour disposer d'une bonne précision de mesure comme dans le cas du *PCR_{FO}*, qui est la mesure de l'offset de fréquence, fréquence qui converge rapidement dans l'ADPLL. Pour d'autres paramètres à extraire, il sera préférable d'utiliser le profil *MGF1*, comme dans le cas du *PCR_{DR}* car la dérive des fréquences à mesurer est très lente.


 FIG. 2.19 – Structure permettant de mesurer le paramètre PCR_{AC} .

2.7.2 La mesure des paramètres

La mesure des paramètres PCR_{DR} et PCR_{FO} est réalisée en utilisant des filtres passe-bas. Ces derniers sont déterminés dans l'annexe C, en section C.2 pour un filtre du premier ordre. Le paramètre PCR_{FO} est obtenu en filtrant le mot de contrôle du synthétiseur de fréquence. Quant au PCR_{DR} , la sortie du comparateur de PCR est utilisée. Cette sortie correspond à l'erreur de phase échantillonnée relative au PCR, qui est une erreur normalisée indépendante de la durée de la période. Afin de retrouver l'erreur de phase temporelle relative au PCR, la sortie PCR_{error} est multipliée par $1/F_{nom}$. Le résultat de cette opération est ensuite appliqué au filtre passe-bas, dont la sortie est multipliée par un terme dépendant du temps d'arrivée du PCR, permettant de trouver la valeur du PCR_{DR} .

Le paramètre PCR_{OJ} est obtenu en analysant les variations sur l'erreur de phase temporelle. Cette analyse des variations est effectuée à l'aide d'un filtre passe-haut du premier ordre, dont la structure est donnée en annexe C.

Le calcul du PCR_{AC} est réalisé de manière autonome, c'est-à-dire sans utiliser une ADPLL, comme le précise le schéma de la figure 2.19, car la PLL est sensible à la fois à la gigue due à l'imprécision du PCR d'entrée et à la gigue du réseau. Le PCR_{AC} est déduit du PCR_e et du nombre d'octets séparant deux PCR_e à l'aide d'un index, ce qui permet de s'affranchir de la gigue du réseau. La différence des PCR_e avec la différence des indexes, ramenée sur une période de T_{nom} , est ensuite filtrée par un passe-haut du second ordre comme le préconise la norme [ETS01] (la structure du filtre est présentée en annexe C).

2.7.3 Génération de dérives virtuelles en fréquence en vue du test

Afin de vérifier le bon fonctionnement du système de mesure, celui-ci sera modélisé avec un langage de haut niveau, en l'occurrence SystemC. Le comportement du système devra être validé pour des réponses indicielles, mais aussi lorsqu'il est soumis à une dérive. En effet, l'un

n	$F_e(n)$	$PCR_e(n)$
0	$F_e(0)$	$PCR_e(0)$
1	$F_e(0) + \Delta F_e$	$PCR_e(0) + T_{PCR} \cdot (F_e(0) + \Delta F_e)$
2	$F_e(0) + 2 \cdot \Delta F_e$	$PCR_e(0) + PCR_e(1) + T_{PCR} \cdot (F_e(0) + 2 \cdot \Delta F_e)$
3	$F_e(0) + 3 \cdot \Delta F_e$	$PCR_e(0) + PCR_e(1) + PCR_e(2) + T_{PCR} \cdot (F_e(0) + 3 \cdot \Delta F_e)$

 TAB. 2.2 – Evolution de la fréquence F_e et du PCR en présence d'une dérive linéaire.

des paramètres relatifs au PCR concerne la mesure de la dérive en fréquence de l'horloge qui a produit les PCRs. Or, avec un outil de modélisation de haut niveau comme SystemC, il est très difficile de générer directement une horloge avec une dérive. Pour remédier à cette difficulté, une solution est de produire des PCRs avec une horloge de fréquence F_{e0} constante et d'ajouter au PCR résultant une dérive linéaire virtuelle à l'aide d'un algorithme. Pour déterminer cet algorithme, il est nécessaire de modéliser la conséquence d'une dérive en fréquence sur les PCRs. En considérant que la dérive modélisée est linéaire, à chaque nouvelle arrivée de PCR, la fréquence a évolué d'une valeur constante ΔF_e virtuelle, qui aura une répercussion sur le PCR, comme le montre le tableau 2.2.

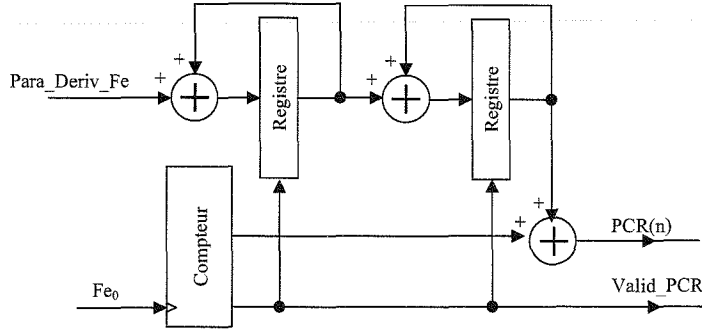
Le tableau 2.2 permet ainsi de calculer la valeur de PCR_e en fonction de n . Son expression est donnée par l'équation 2.64, qui prend en compte la valeur du PCR initial $PCR_e(0)$. Pour faire apparaître le terme de dérive $Deriv_F_e$ dans cette expression, les relations 2.65 sont utilisées. Ainsi, le schéma structurel d'un générateur de PCR produisant une dérive virtuelle est montré en figure 2.20. Le paramètre $Para_Deriv_F_e$ fixe cette dérive virtuelle : si l'on souhaite une dérive de $0,5 \text{ Hz/s}$ avec une période $T_{PCR} = 40 \text{ ms}$, $Para_Deriv_F_e$ va être égal à $0,5 \times (40 \cdot 10^{-3})^2$, soit $0,8 \cdot 10^{-3}$.

$$PCR_e(n) = PCR_e(0) + n \cdot T_{PCR} \cdot F_e(0) + \Delta F_e \cdot T_{PCR} \sum_{i=1}^{i=n} i \quad (2.64)$$

$$\Delta F_e \cdot T_{PCR} = Deriv_F_e \cdot T_{PCR}^2 = Para_Deriv_F_e \quad (2.65)$$

2.8 Conclusion

Dans ce chapitre nous avons présenté la modélisation de l'ensemble du système de mesure des paramètres relatifs au PCR. Ce système comprend deux éléments principaux : une unité de synchronisation qui régénère l'horloge système MPEG-2 et une unité de traitement permettant d'extraire les paramètres relatifs au PCR, dont trois de ces paramètres sont extraits à partir de paramètres fournis par l'unité de synchronisation.

FIG. 2.20 – Générateur de PCR fournissant une dérive virtuelle.

Après une étude de l'état de l'art qui a permis de recenser de manière non exhaustive les techniques de synchronisation et de synthèses de fréquence, des insuffisances ont pu être mises en évidence. En effet, certaines structures prennent trop de ressources FPGAs, ne sont pas adaptées à la mesure des paramètres relatifs au PCR et nécessitent un VCO externe. D'autres structures ne sont réalisables qu'avec un ASIC. Ainsi, la solution proposée, qui peut être implantée soit dans un FPGA, soit sous forme d'un ASIC, repose sur une ADPLL permettant de se synchroniser sur des PCRs et pouvant être intégrée entièrement dans un circuit FPGA, ceci grâce à son synthétiseur de fréquence.

L'originalité de ce synthétiseur repose sur la génération d'un signal de fréquence centrale, subissant un retard ou une avance de phase calibrés. La modélisation de ce synthétiseur permet d'estimer ses performances en terme de résolution en fréquence et de gigue RMS et crête-à-crête intrinsèques en fonction de la fréquence de référence appliquée. L'étude montre que la résolution en fréquence est, dans le pire cas, de $0,25\text{ Hz}$ lorsque le décalage temporel est de $\pm \frac{T_{nom}}{4}$. Dans ce cas de figure, la gigue crête-à-crête intrinsèque au synthétiseur est de $\frac{T_{nom}}{4}$ et la gigue RMS est inférieure à 100 ps dans la plage de fréquence dans laquelle la norme suppose qu'il n'y a pas d'erreurs sur l'offset de fréquence. Ainsi, sachant que ce synthétiseur va incrémenter le compteur PCR_r de l'ADPLL qui fournit des entiers et donc que la résolution du PCR est d'une période et que le PCR est une valeur cumulée sur une durée T_{PCR} , l'impact de la gigue crête-à-crête et de la gigue RMS sur le PCR est négligeable.

Après une étude de stabilité en utilisant le critère de Jury, le comportement de l'ADPLL est analysé pour des valeurs limites de T_{PCR} (10 ms et 100 ms) ainsi que pour sa valeur nominale (40 ms). Ensuite, sa réponse indicielle est étudiée en fonction des coefficients du filtre pour la valeur nominale de T_{PCR} . Lorsque ceux-ci sont des entiers, la réponse de l'ADPLL présente un dépassement. Ce dernier peut être supprimé en respectant les conditions suivantes : $0 < G_2 \ll 1$ et $\frac{G_1}{G_2} \gg 1000$. En appliquant ces conditions, la bande passante de l'ADPLL devient plus faible, ce qui permet de mieux atténuer la gigue d'entrée. Cette première version de l'ADPLL présente

cependant un inconvénient qui est la dépendance de sa stabilité à la période d'échantillonnage T_{PCR} . Ainsi, une version de l'ADPLL avec normalisation de l'erreur de PCR est ensuite proposée, permettant la suppression de cette dépendance. Cependant, avec cette version, la bande passante n'est pas constante, ce qui engendre un comportement différent dans le spectre des fréquences en fonction de la période d'échantillonnage. Ce comportement a une conséquence directe sur les paramètres à mesurer (dérives, gigue, ...). Dans le but d'obtenir une bande passante constante, la première version de l'ADPLL est utilisée avec une technique de sur-échantillonnage, permettant d'avoir une bande passante constante de l'ADPLL. Dans ce cas, les PCRs reçus doivent être interpolés.

De manière générale, l'ADPLL fournit plusieurs signaux nécessaires à la mesure des paramètres relatifs au PCR. Ces signaux doivent être filtrés pour extraire les paramètres. Les filtres utilisés sont obtenus par synthèse numérique de filtres analogiques, leur fréquence de coupure étant spécifiée par la norme [ETS01]. Les structures des filtres sont ensuite décrites (en annexe), et les fréquences de coupures sont à choisir suivant la qualité du filtrage désirée.

Comme le système de mesure a été jusqu'à présent modélisé de manière mathématique, il doit être décrit en un langage synthétisable en vu de l'implanter puis de le caractériser expérimentalement afin de valider le modèle théorique. Ainsi, après une caractérisation de la gigue du synthétiseur de fréquence, l'ADPLL sera validée pour différent temps d'arrivée des PCRs et pour plusieurs réponses dynamiques. Les filtres de mesures seront ensuite caractérisés sur plan du temps de convergence, du temps d'exécution pour un processeur donné et sur le plan de la précision.

Chapitre 3

Implantation et caractérisation du système de mesure

3.1 Méthodologie

3.1.1 Modélisation architecturale de haut niveau

Afin de vérifier la fonctionnalité du système de mesure, l'ADPLL et les filtres de mesures sont modélisés avec SystemC [Sys02, Swa01]. Cet outil permet d'utiliser différents niveaux d'abstraction et de modéliser les parties matérielle et logicielle dans le même projet. En effet, il n'est pas nécessaire de décrire en détail les éléments du système comme le filtre de boucle de l'ADPLL ou le dispositif de monitoring. Le code en SystemC suivant montre l'instantiation de blocs de haut niveau constituant le système et son environnement.

```
// Code SystemC du système complet
...
// déclaration des éléments
PCR_GENERATOR      *u_PCR_GENERATOR;
ADPLL_TOP          *u_ADPLL_TOP;
MONITOR            *u_MONITOR;
...
```

Le code SystemC décrit est structuré de manière hiérarchique [Lém95]. Les différents blocs utilisés sont présentés dans le code SystemC suivant :

```
// Code SystemC du module ADPLL
// Déclaration des ports
sc_in<bool>         clk_4x;           //Entrée horloge 4Fnom
sc_in<bool>         clk_PCR;         //Entrée Valid PCR
sc_in<bool>         RAZ;             //Entrée RAZ
sc_in<sc_int<42> > IN_PCR_E;         //Entrée PCR_e 42 bits
sc_out<sc_int<42> > PCR_ERROR;       //Sortie erreur PCR error
sc_out<float >      OUT_PI_FILTER;   //Sortie filtre de boucle
```

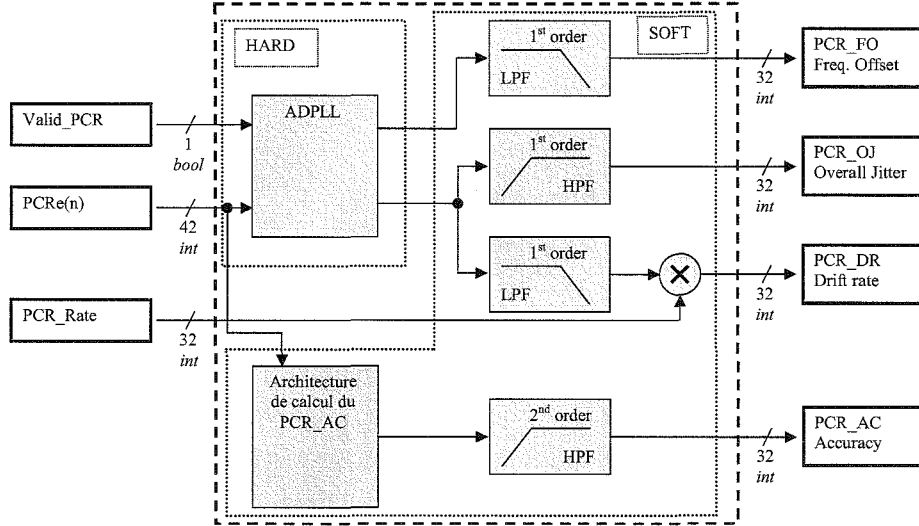


FIG. 3.1 – Architecture de mesure pour un programme.

```
// component declaration
COUNT_TOP      *u_CPT_COUNT;
LATCH_COMPARATOR_TOP *u_LATCH_COMPARATOR_TOP;
PI_FILTER_TOP    *u_PI_FILTER_TOP;
DCO_TOP          *u_DCO_TOP;
...
```

Ce code SystemC est ensuite traduit en langage VHDL [oEEE00, JHM01, AFF⁺01] synthétisable pour l'implantation matérielle à l'aide de l'outil Nepsys [Nep02], alors que les éléments à implanter de manière logicielle seront à adapter pour le langage C embarqué requis pour le processeur.

Ainsi, le système de mesure est implanté en deux parties (figure 3.1). La première partie comprend l'ADPLL qui est implanté matériellement car le synthétiseur de fréquence fonctionne avec un compteur qui est incrémenté par une horloge de fréquence élevée. La deuxième partie inclut le calcul de la précision du PCR (PCR_{AC}) et les filtres de mesure permettant d'extraire les paramètres relatifs au PCR. Ces filtres nécessitent des traitements sur des données au format flottant. Ainsi, cette partie est implantée logicielle. Ces filtres peuvent être implantés soit par un processeur externe soit à l'aide d'un processeur enfoui dans le FPGA permettant de traiter tous les paramètres relatifs au PCR de tous les programmes dans l'optique d'intégrer tous les éléments du système de mesure dans un circuit unique.

3.1.2 Dispositif expérimental

Dans le but de l'estimation des ressources et de la validation du système de mesure, nous utilisons une plate-forme EXCALIBUR incluant un FPGA APEX EP20K200EFC484-2x d'ALTERA (voir annexe D). Cette plate-forme est composée d'un oscillateur à quartz fonctionnant à 33,333 MHz. Pour générer l'horloge de référence du synthétiseur de fréquence, l'une des deux PLLs disponibles dans le FPGA est utilisée. Dans le cas où cette horloge est configurée sur $4.F_{nom}$ (avec $F_{nom} = 27\text{ MHz}$), l'une des PLLs du FPGA pourra fournir au mieux la fréquence $4.F_{nom} = 108,333333\text{ MHz}$ en appliquant un facteur de multiplication de 13/4, soit $F_{nom} = 27,083333\text{ MHz}$.

Une mémoire flash de 1 Mo est disponible afin de stocker le fichier de configuration de l'APEX et une mémoire de 256 Ko est utilisée lorsque le NIOS est implanté pour l'extraction des paramètres relatifs au PCR. Cette mémoire contiendra le programme du NIOS et les variables à sauvegarder.

Une fois que le système est synthétisé puis implanté, il doit être validé. Ainsi, l'ADPLL est caractérisée en terme de gigue, de réponse en fréquence lorsqu'elle est stimulée par un échelon ou une rampe de fréquence. Du côté des paramètres relatifs au PCR, les temps d'exécution des filtres sont analysés, ainsi que leur temps de réponse et la précision obtenue.

3.2 Implantation et caractérisation de l'ADPLL

3.2.1 Validation des contraintes de temps du synthétiseur de fréquence

Dans le but de tester le fonctionnement de l'ADPLL sous des contraintes de temps sévères, l'architecture est validée en prenant en compte le routage interne au FPGA. En effet, le signal propagé dans le circuit ayant la fréquence la plus élevée correspond au signal $4.F_{nom}$ dont la fréquence est de 108.333333 MHz et de 9.26 ns de période. Les éléments de l'architecture qui sont cadencés par cette horloge font partie du synthétiseur de fréquence. Il s'agit de la machine d'état, du comparateur et du compteur du synthétiseur. Pour valider le fonctionnement de ces trois éléments, la fréquence de fonctionnement de l'horloge est spécifiée à 120 MHz dans l'outil de développement QUARTUS II. L'architecture du synthétiseur de fréquence initiale est basée sur le traitement de mots de 32 bits. Dans ces conditions, l'analyseur temporel donne une fréquence maximale de fonctionnement de 107 MHz. Une solution pour atteindre la fréquence de fonctionnement spécifiée est d'optimiser le format des données, ce qui réduit le nombre de bascules que cette horloge doit commander. Le format des données traitées tient au minimum sur 27 bits. L'analyse temporelle avec ce format de données permet d'atteindre une fréquence de fonctionnement de 129 MHz.

3.2.2 Caractérisation des ressources matérielles de l'ADPLL

Les résultats de synthèse VHDL de l'ADPLL sont donnés dans le tableau 3.1. Ces résultats de synthèse VHDL sont obtenus après routage du circuit FPGA et sont issus du rapport de synthèse détaillé. Ce tableau montre que l'ADPLL nécessite 1680 cellules logiques correspondant à 20,2 % des ressources disponibles du circuit APEX 200. Le bloc consommant le plus de ressources matérielles est le diviseur du synthétiseur de fréquence, qui requière 46,8 % des ressources de l'ADPLL.

Fonctions	Cellules Logiques (LC)	% Ressources utilisées
Compteur PCR	193 (11,4 %)	2,3 %
Comparateur PCR	410 (24,4 %)	4,9 %
Filtre de boucle	76 (4,5 %)	0,9 %
Synth_Freq diviseur	786 (46,8 %)	9,4 %
autres	215 (12,8 %)	2,6 %
Total (20K200 ; 8320 LC)	1680	20,2 %

TAB. 3.1 – Résultats de synthèse de l'ADPLL.

3.2.3 Caractérisation de la gigue du synthétiseur

L'analyse de la gigue de l'ADPLL est effectuée avec un oscilloscope WAVEPRO 950 de LeCroy. Cet appareil de mesure dispose d'une fréquence d'échantillonnage de 16 GS/s, d'une bande passante de 1 GHz et est équipé d'un module d'analyse temporelle et de gigue.

Pour mesurer la gigue introduite par la technique de décalage du synthétiseur de fréquence, il est nécessaire de caractériser la gigue de l'horloge de référence à 33.333 MHz. Cette horloge est disponible sur la plate-forme EXCALIBUR et permettra de générer la fréquence $4.F_{nom}$. Une analyse de gigue à long terme de l'horloge fournie par le quartz de la plate-forme donne l'histogramme de gigue présenté en figure 3.2. La gigue RMS de cette horloge est de 24.65 ps.

Lorsque le synthétiseur de fréquence est implanté uniquement et configuré pour produire la fréquence $F_{nom} + 800Hz$ qui correspond au cas où la fréquence $4.F_{nom}$ est divisée par trois ou quatre, la gigue à long terme de la figure 3.3 est obtenue. La gigue RMS mesurée est de 107 ps. L'histogramme présente la gigue autour de la gigue crête à crête. La raie se trouvant à droite correspond à la gigue de la période T_4 , qui est décalée de 4.6 ns du centre du graphique. La gigue crête à crête mesurée est de 9.280 ns. Cette valeur correspond au saut de phase d'un quart de période, et qui est aussi la résolution du synthétiseur de fréquence mesurée, dont la valeur

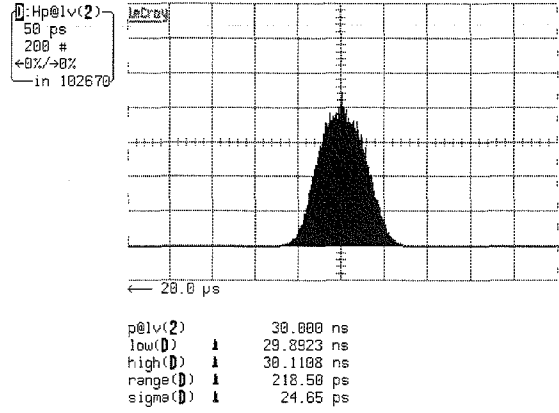
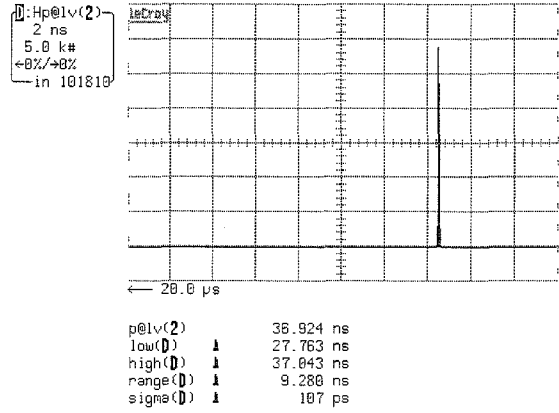


FIG. 3.2 – Histogramme de gigue à long terme du quartz de 33,333 MHz.


FIG. 3.3 – Histogramme de gigue à long terme du synthétiseur de fréquence fournissant $F_{nom} + 800Hz$.

théorique est de $1/(4.F_{nom}) = 9.260ns$. Le signal produit par le synthétiseur va incrémenter le compteur PCR local de l'ADPLL. Sachant que, dans notre application, la fréquence est remise à jour au bout d'une durée T_{PCR} , cette gigue sera compensée sur cet intervalle.

Pour obtenir l'histogramme de gigue de la période T_{nom} , le synthétiseur de fréquence doit être configuré pour générer la fréquence F_{nom} , qui correspond à la division de la fréquence $4.F_{nom}$ par quatre uniquement. Ainsi, $In_Synth(n)$ doit être égal à zero. L'histogramme de gigue au long terme qui en résulte est donné en figure 3.4 : la gigue RMS mesurée est de 24.13 ps. Dans ce cas, il peut être supposé que cette gigue est due à la gigue de l'horloge de référence, et que la gigue due au synthétiseur de fréquence est nulle.

Comme le montre l'équation 2.12, la gigue intrinsèque au synthétiseur de fréquence dépend de sa valeur d'entrée et donc de la fréquence moyenne fournie. La figure 3.5 présente la gigue

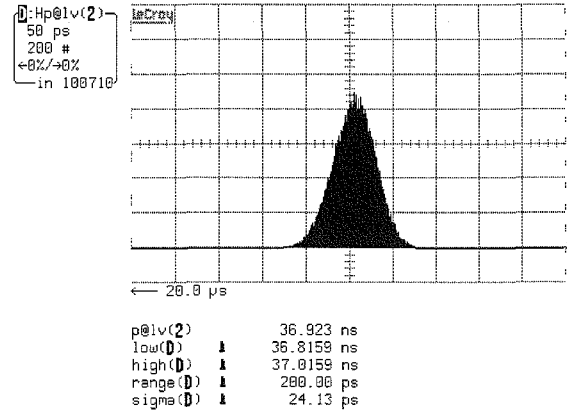


FIG. 3.4 – Histogramme de gigue au long terme du synthétiseur de fréquence générant F_{nom} .

intrinsèque au synthétiseur de fréquence en fonction de la fréquence autour de F_{nom} dans la plage de $\pm 800 Hz$. Dans le standard DVB, la fréquence de l'horloge de l'encodeur MPEG-2 est acceptable quand elle se trouve dans cette plage. La gigue de fréquence maximale intrinsèque au synthétiseur est de $100 ps$ dans cette plage. Comme la résolution sur le PCR est d'une période, cette gigue RMS aura un impact négligeable sur les PCRs estimés.

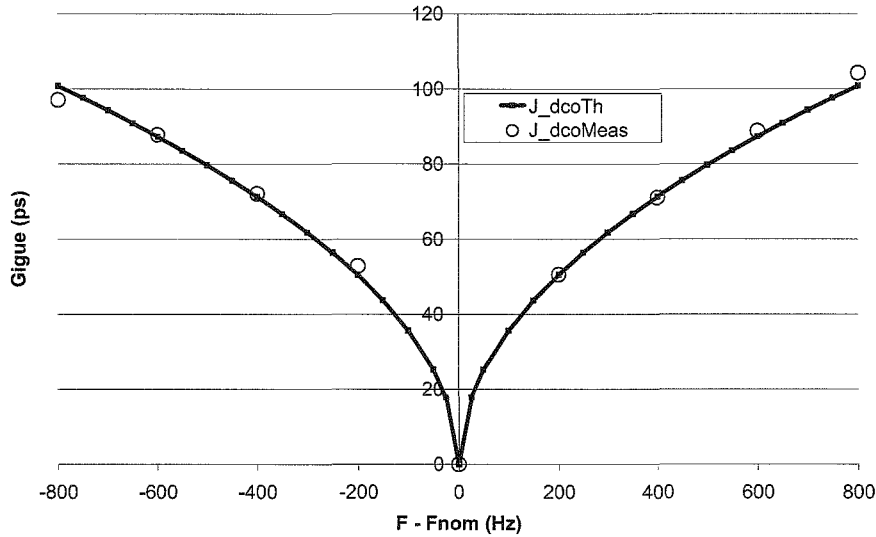


FIG. 3.5 – Gigue intrinsèque au synthétiseur en fonction de la fréquence autour de F_{nom} .

3.2.4 Etude comportementale de l'ADPLL

Le comportement de l'ADPLL est tout d'abord analysé pour des périodes d'échantillonnage variables afin de vérifier les résultats théoriques. Les valeurs minimale et maximale de T_{PCR} sont 10 *ms* et 100 *ms*, sa valeur nominale étant à 40 *ms*. La réponse indicielle de l'ADPLL vis-à-vis de ces périodes d'échantillonnage est donnée en figure 3.6 lorsque l'entrée est soumise à un échelon allant de F_{nom} à $F_{nom} + 800Hz$.

Les points apparaissant sont les points expérimentaux et les courbes représentent les réponses théoriques de l'ADPLL. Les valeurs expérimentales des fréquences sont des valeurs déduites directement des PCR et de leur temps d'arrivée. On constate que les résultats expérimentaux concordent avec le modèle théorique. Ainsi, en moyennant les points expérimentaux, on retrouverait la courbe théorique. De ce fait, les temps de réponse normalisés théorique et réel sont identiques. Aussi, comme la théorie le prévoyait, la réponse est pseudo-périodique pour $T_{PCR} = 10ms$, elle présente un dépassement sans pseudo-période pour $T_{PCR} = 40ms$ puis redevient à nouveau pseudo-périodique pour $T_{PCR} = 100ms$. Comme mentionné ci-dessus, la mesure de la fréquence de l'ADPLL doit être moyennée dans le temps, car l'ADPLL fournit des fréquences par palier et la fréquence de sortie est déduite par les PCR et leur temps d'arrivée. Mais pour des raisons de clarté, la courbe moyennée n'est pas représentée.

L'écart entre deux paliers successifs varie en fonction de cette période d'échantillonnage, comme le montre les trois courbes de la figure 3.6. L'équation qui détermine l'écart ΔF_r entre deux paliers successifs est :

$$\Delta F_r = \frac{1}{T_{PCR}} \quad (3.1)$$

Le comportement de l'ADPLL est ensuite analysé en fonction des coefficients du filtre à la valeur nominale de T_{PCR} . La figure 3.7 montre les résultats obtenus avec plusieurs valeurs de $G1$ ($G1 = 20, 30, 40$ et 60), et avec $G2 = 1$. Les points représentés correspondent aux points expérimentaux, alors que la courbe tracée illustre la réponse théorique de l'ADPLL. Là aussi, l'étude comportementale théorique coïncide avec le comportement de l'ADPLL implantée. Les dépassements qui sont présents dans ces courbes de réponse peuvent être atténués ou supprimés comme mentionné dans l'étude théorique. Pour cela, la valeur du coefficient intégral $G2$ doit être plus faible, c'est-à-dire inférieure à un, mais aussi supérieure à zéro pour respecter les conditions de stabilité.

Afin de valider l'étude théorique, les coefficients des filtres auront les valeurs $G1 = 10$ et $G2 = 0,001$ pour l'implantation. Dans la pratique, une solution pour implanter $G2$ serait d'utiliser un format de donnée à virgule fixe pour réaliser le filtre de boucle. La partie fractionnaire aurait alors une largeur de 10 *bits* : $0,001_{10} \simeq 0,0000000001_2 = (2^{-10})_{10}$. Mais cela entraînerai alors une augmentation des ressources matérielles du filtre de mesure. Une autre solution ne nécessitant

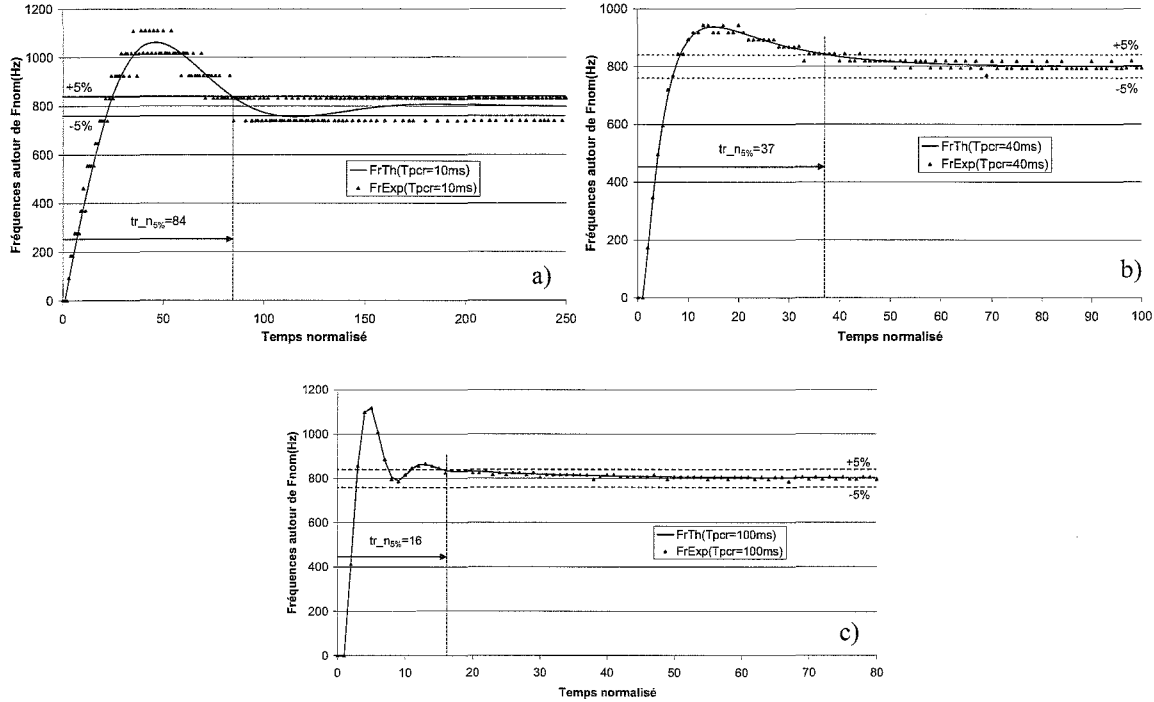
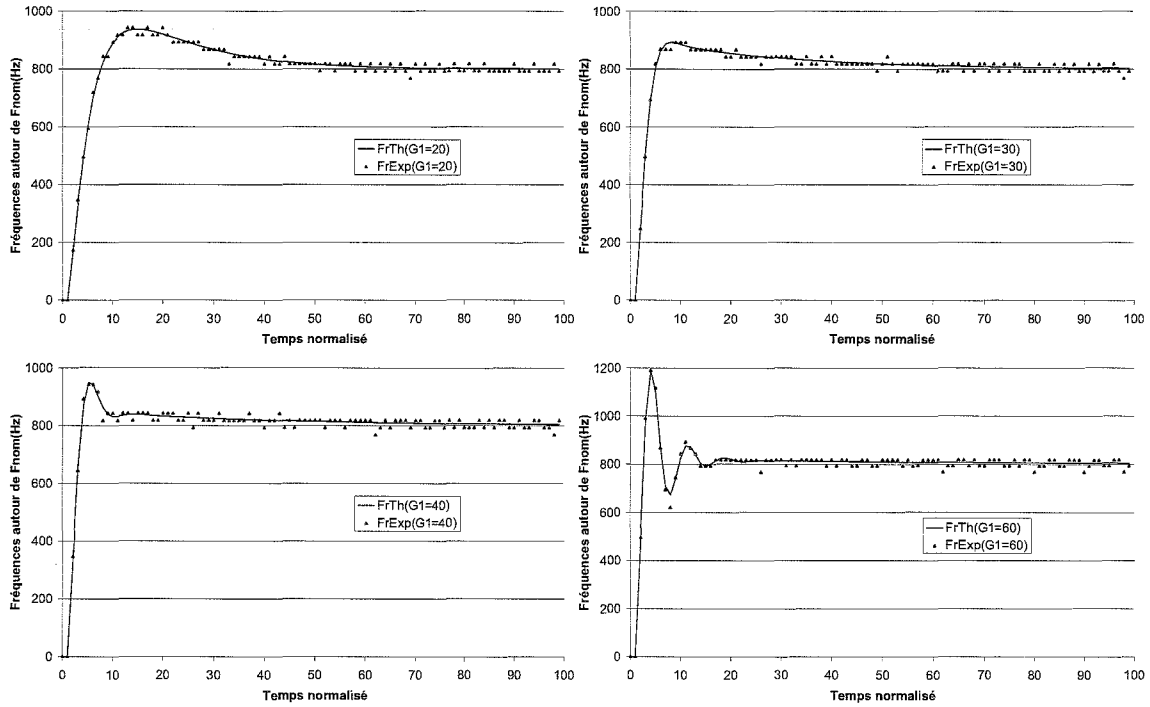
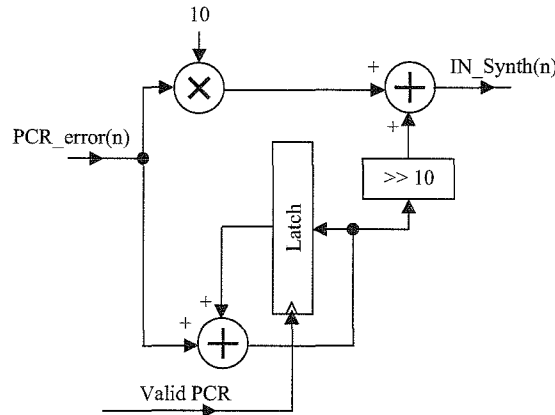


FIG. 3.6 – Comportement de l'ADPLL pour différentes valeurs de T_{PCR} . $T_{PCR} = 10ms$ (a), $T_{PCR} = 40ms$ (b), $T_{PCR} = 100ms$ (c).

pas de ressources matérielles supplémentaires est l'insertion d'une division câblée (un décalage par 10) à la sortie de la fonction intégration du filtre de boucle, comme le montre la figure 3.8. L'implantation de l'ADPLL avec ce filtre permet d'obtenir les résultats de la figure 3.9. Ces résultats expérimentaux (ensemble des points représentés) sont bien en corrélation avec l'analyse théorique (courbe passant entre les points). Ainsi, le comportement de l'ADPLL est aussi validé lorsque les coefficients du filtre implanté ne sont plus des entiers, mais des nombres fractionnaires.

Maintenant que le comportement dynamique de l'ADPLL est validé par rapport à l'étude théorique, il est nécessaire de vérifier qu'une variation de la période d'échantillonnage au cours du temps n'a pas d'influence sur le comportement de l'ADPLL lorsqu'elle est synchronisée. Ainsi, la période T_{PCR} est modifiée au cours du temps à un instant donné. La fréquence d'entrée est configurée sur une fréquence arbitraire de $F_{nom} + 800Hz$. La figure 3.10, courbe (a), présente le comportement en fréquence de l'ADPLL, autour de F_{nom} , dont la fréquence est déduite directement des PCRs générés localement. Le changement de la période d'échantillonnage a comme effet une modification de la résolution de la fréquence des PCRs, comme prévu par l'équation 3.1. La période d'arrivée des PCRs passe de 20 ms environ à 40 ms, comme le montre la figure 3.10(b).


FIG. 3.7 – Comportement de l'ADPLL pour $G1 = \{20, 30, 40, 60\}$.

FIG. 3.8 – Filtre de boucle implanté pour $G1 = 10$ et $G2 = 0,001$.

Le changement de la période d'échantillonnage n'a, par contre, pas d'influence sur la synchronisation en fréquence et en phase de l'ADPLL (figure 3.10(c) et (d)). Celle-ci reste synchronisée quelque soit le saut sur T_{pcr} , l'écart de fréquence entre F_e et F_r (et donc aussi l'erreur de PCR) n'ayant pas été affectés.

L'étude du comportement de l'ADPLL est aussi nécessaire lorsqu'une dérive se trouve sur

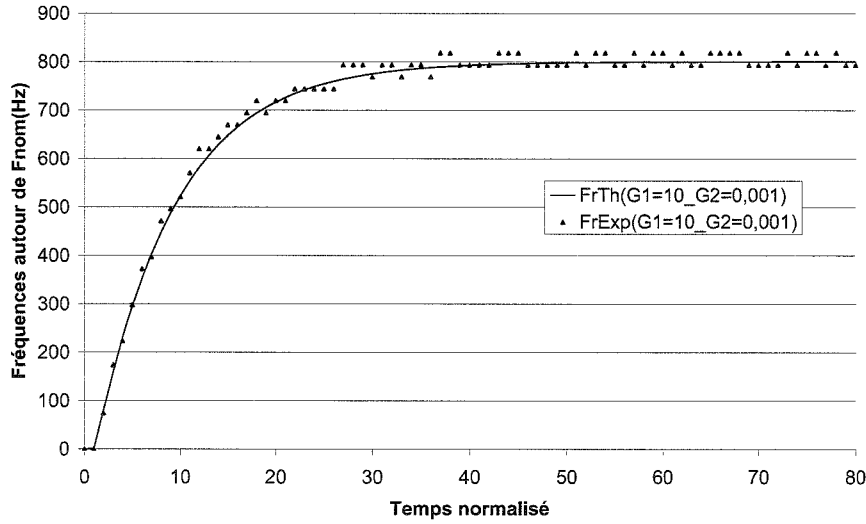


FIG. 3.9 – Comportement de l'ADPLL pour $G1 = 10$ et $G2 = 0,001$.

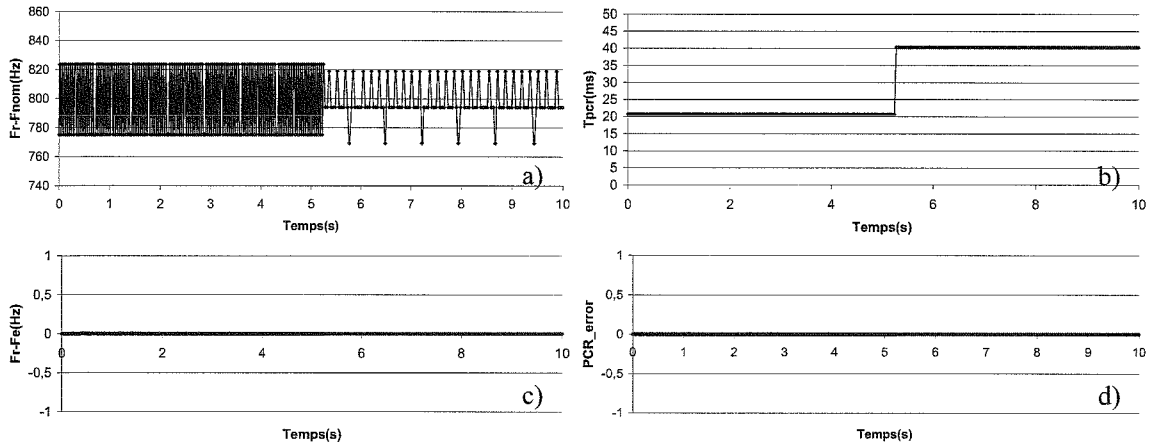


FIG. 3.10 – Comportement de l'ADPLL pour une période d'échantillonnage variable. Fréquence de l'ADPLL autour de F_{nom} (a), durée de T_{PCR} (b), erreur de fréquence par rapport à l'entrée (c), erreur de PCR (d).

l'entrée. En effet, l'un des paramètres relatifs au PCR à mesurer concerne la mesure de la dérive en fréquence de l'horloge système MPEG-2. Une rampe de fréquence de pente arbitraire de 64 Hz/s est produite sur la fréquence d'entrée. L'ADPLL doit avoir un temps de réponse rapide pour suivre la fréquence d'entrée, ce qui n'est pas possible avec $G1 = 10$ et $G2 = 0,001$ (paramètres utilisés précédemment). Les paramètres de l'implantation sont $G1 = 30$, $G2 = 1$, $T_{PCR} = 40 \text{ ms}$ et $CST = 108333333$. Les valeurs de $G1 = 30$ et $G2 = 1$ permettent d'assurer un temps de réponse rapide (voir la réponse en fréquence pour ces paramètres en figure 3.7)

permettant de suivre la fréquence d'entrée. La courbe a) de la figure 3.11 montre la fréquence de sortie de l'ADPLL, autour de F_{nom} . L'ADPLL est constamment re-synchronisée sur F_e en présence d'une dérive et cela sans erreur de traînage (figure 3.11, courbe b)). En effet, en cas d'erreur de traînage non nulle, un décalage par rapport à zéro apparaîtrait sur la courbe b) donnant l'erreur entre F_r et F_e . Or, dans notre cas, la courbe b) est centrée sur zéro. Par contre, il existe une erreur sur la phase échantillonnée PCR_{error} . Sa valeur relative par rapport à la différence entre deux PCR_e successifs, exprimée en ppm , est donné en figure 3.11, courbe c). Cette erreur est, dans le cas d'une dérive de 64 Hz/s , de 9 ppm en moyenne. Cela s'explique par la relation qui lie PCR_{error} à l'erreur de fréquence $F_r - F_e$ et à T_{PCR} (équation 2.4) : l'erreur de phase correspond à la dérivée de la différence entre F_e et F_r , multipliée par le temps d'arrivée des PCRs.

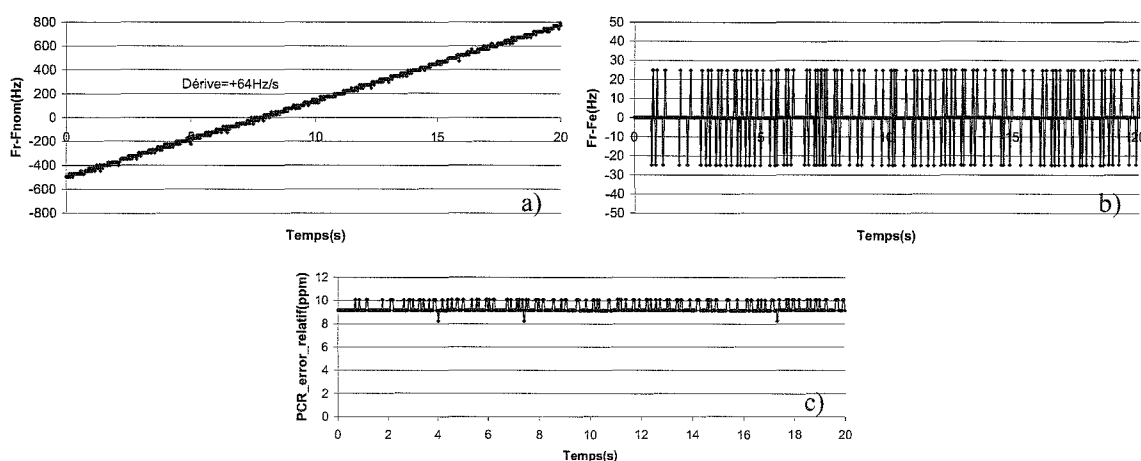


FIG. 3.11 – Comportement de l'ADPLL lorsque la fréquence d'entrée dérive. Fréquence de l'ADPLL autour de F_{nom} (a), erreur de fréquence par rapport à l'entrée (b), erreur de PCR (c).

Comme précisé dans le paragraphe précédent, l'ADPLL prend place dans un système qui mesure la dérive de l'horloge qui a généré les PCRs. D'après la norme [ETS01], cette dérive de fréquence est mesurée à partir de l'erreur de PCR et doit être inférieure à 75 mHz/s . Cette erreur de PCR moyennée doit évoluer linéairement en fonction de la pente de la fréquence d'entrée. La figure 3.12 montre que l'erreur de PCR varie linéairement en fonction de la dérive lorsque cette dérive est supérieure à $0,5 \text{ mHz/s}$. L'erreur de PCR et la dérive sont liées par une équation de la forme $y = ax + b$ où $a = 0,1477 \text{ ppm.s.Hz}^{-1}$ et $b = 0$. En dessous d'une dérive de $0,5 \text{ mHz/s}$, l'erreur de PCR moyennée n'évolue plus linéairement en fonction de la dérive dû à la présence de bruit sur le signal PCR_{error} . Dans le but de réaliser des mesures de dérives inférieures à $0,5 \text{ mHz/s}$, ce signal doit être filtré en choisissant une fréquence de coupure adaptée, comme précisé dans la section 3.3.

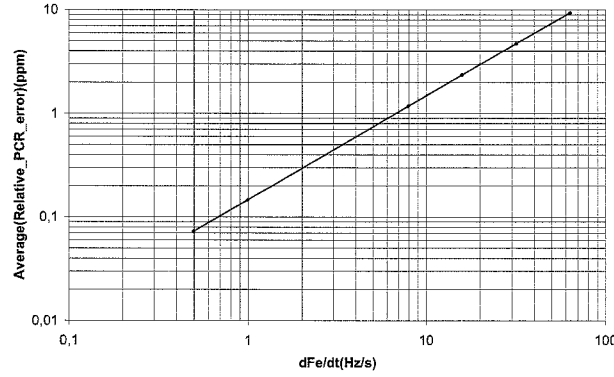


FIG. 3.12 – Erreur de PCR relative moyenne PCR_error en fonction de la dérive de la fréquence ($\frac{dF_c}{dt}$).

3.2.5 Etude comportementale de l'ADPLL avec normalisation de l'erreur de PCR

L'inconvénient de l'ADPLL vue précédemment est sa modification du comportement dynamique en fonction de la période d'échantillonnage T_{PCR} . Bien que la période nominale soit de 40 ms , il est nécessaire de prévoir un système fonctionnant pour des périodes allant de 10 ms à 100 ms . Ainsi, il est présenté dans le chapitre 2, section 2.6.2, une solution permettant de rendre la stabilité de l'ADPLL indépendante de la période d'échantillonnage. Cette solution repose sur une normalisation de l'erreur de PCR, afin que le comportement dynamique de l'ADPLL soit insensible face aux variations de la période d'échantillonnage.

La structure implantée est modélisée par le schéma de la figure 2.16 du chapitre 2. Sur le plan matériel, cette solution consomme 466 cellules logiques de plus que la première version de l'ADPLL (soit +28% par rapport à la solution initiale), ainsi que 17 bits de mémoire FPGA en plus.

Afin de valider le principe de l'ADPLL avec une normalisation de l'erreur de PCR, celle-ci sera soumise à un échelon de fréquence dans le but de comparer sa réponse indiciaire pour plusieurs périodes d'échantillonnage. Lorsque l'ADPLL est commandée par un échelon allant de 0 à $+800\text{ Hz}$, les réponses de la figure 3.13 sont obtenues pour $T_{PCR} = 20\text{ ms}$ (courbe a)) et pour $T_{PCR} = 40\text{ ms}$ (courbe b)). Il est possible de constater que, mis à part la précision de la mesure de la fréquence (déduite des PCRs) qui dépend de T_{PCR} , il y a une très forte corrélation de la réponse de l'ADPLL pour des périodes d'échantillonnage différentes. Ainsi, cette très forte corrélation entre les réponses permet de valider le modèle théorique de l'ADPLL avec normalisation de l'erreur de PCR et de montrer que l'ADPLL peut fonctionner à des périodes d'échantillonnages différentes sans en affecter la stabilité. Cependant, le temps de réponse de l'ADPLL varie en

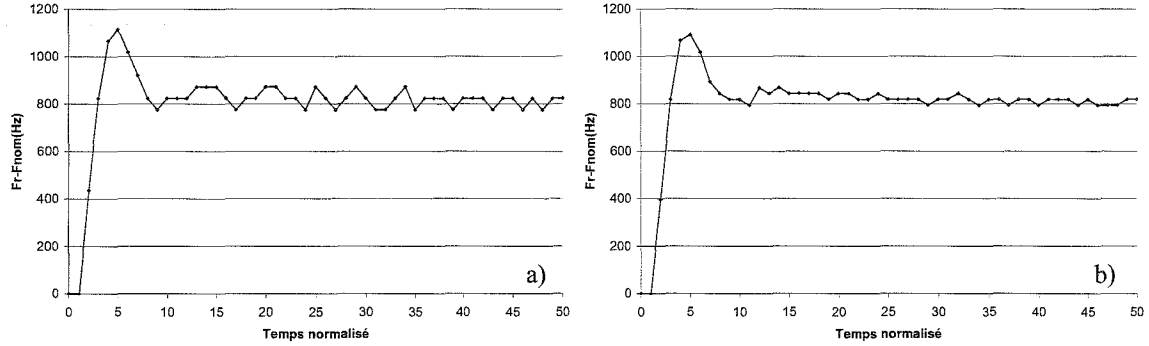


FIG. 3.13 – Comportement de l'ADPLL avec normalisation de l'erreur de PCR pour plusieurs périodes d'échantillonnage. $T_{PCR} = 20ms$ (a), $T_{PCR} = 40ms$ (b).

fonction de la période d'échantillonnage, ce qui a une influence sur l'ADPLL.

3.2.6 Etude comportementale de l'ADPLL associée à une technique de sur-échantillonnage

Comme présenté dans la section 2.6.3, une technique permettant de rendre la bande passante constante, quelque soit la durée T_{PCR} , est l'utilisation du sur-échantillonnage. Pour ce faire, les PCRs reçus sont re-calculés afin d'en fournir à période constante, cela à l'aide d'une interpolation des PCR_e . La structure implantée correspond à celle présentée en figure 2.18 (section 2.6.3). L'horloge H_{Idx} , qui permet de mesurer la durée séparant deux PCRs, et synchrone avec le signal $Valid_PCR$, a une fréquence de $F_{Idx} = 2 kHz$, soit une période $T_{Idx} = 0,5 ms$. Si l'on souhaite avoir une fréquence de sur-échantillonnage de $8 ms$, le facteur de division K doit être de 16. Les autres paramètres de l'implantation sont $G1 = 45$, $G2 = 1$ et $CST = 108333333$.

L'implantation du dispositif de sur-échantillonnage, qui comprend un procédé d'interpolation des PCRs, consomme 824 cellules logiques (soit +49% par rapport à la solution initiale) et 84 bits de mémoire FPGA. L'opérateur de division fournit le quotient, qui représente la valeur entière de l'écart entre deux PCRs résultant du sur-échantillonnage et un reste, qui est pris en compte dans l'accumulateur, pour ainsi éviter une perte de précision dû à la troncature sur des valeurs entières.

Dans le but de valider le fonctionnement de l'ADPLL en sur-échantillonnage, un échelon en fréquence est appliqué sur la fréquence générant les PCR_e , pour $T_{PCR} = 20ms$ et $T_{PCR} = 40ms$. La figure 3.14 montre les résultats du comportement en fréquence de l'ADPLL. La figure 3.14(a) et (c) sont les offsets de fréquence déduits directement des PCR_e_SE pour $T_{PCR} = 20ms$ et $T_{PCR} = 40ms$ respectivement. A partir de ces deux courbes, où la fréquence est directement déduite des PCRs, il est difficile de voir l'allure de la fréquence au cours du temps. On voit

cependant des pics apparaître après le régime transitoire (pour $t > 1,4s$), qui correspondent à des pics présents à l'entrée aux mêmes instants. Ces derniers correspondent eux mêmes aux moments où une période supplémentaire de l'horloge F_e est prise en compte sur le PCR due à une rotation de phase, c'est-à-dire une phase cumulée qui n'est pas prise en compte sur le PCR tant que cette phase cumulée n'est pas représentable sur une période complète.

Pour déterminer l'allure de la réponse en fréquence, un moyennage glissant sur quatres points est appliqué aux courbes (a) et (c) de la figure 3.14. La courbe *OffsetFrMoy4* est obtenue dans chaque cas (voir figure 3.14(b) et (d)). Ces courbes sont comparées aux courbes théoriques, appelées *OffsetFrTh*. Il est possible de constater qu'il y a une forte corrélation entre la réponse en fréquence théorique et la réponse en fréquence expérimentale moyennée. La seule différence que l'on peut observer est la présence d'un décalage sur la valeur finale de la fréquence. Ce décalage provient du fait que le PCR_e est une valeur entière, qui ne tient pas compte de la phase de l'horloge qui a générée les PCRs. Ainsi, l'interpolateur utilise une valeur d'entrée qui est tronquée. Lorsque l'interpolateur calcule l'écart entre deux PCRs successifs sur la base de l'horloge de sur-échantillonnage, il se base sur cette valeur tronquée, qui est une source d'imprécision. Ainsi, le résultat du calcul de cet écart est aussi affecté. Cet écart est ensuite cumulée au valeur précédente du PCR d'entrée de l'ADPLL, d'où un cumul d'imprécisions. Cela entraîne ainsi un décalage de la valeur finale de la fréquence. Une solution pour résoudre ce problème serait de recevoir des PCRs avec une partie fractionnaire représentative de la phase de l'horloge de l'encodeur.

3.3 Implantation et caractérisation des filtres de mesure

3.3.1 Ressources d'implantation des filtres de mesure

Le choix de l'unité de traitement pour l'implantation des filtres de mesure se porte sur un processeur NIOS, ce qui permet d'intégrer l'ensemble du système dans un unique FPGA. Sa description succincte est effectuée en annexe D. Ce processeur NIOS récupère, par l'intermédiaire d'un multiplexeur 3 : 1, le PCR d'entrée et le PCR local, ainsi que le mot de contrôle du synthétiseur. Ces derniers sont sélectionnés à l'aide du signal *Para_Ctrl* issu du NIOS. Le signal *Time_Idx* donne une information sur le temps séparant l'arrivée de deux PCR_e successifs servant à calculer les paramètres PCR_{DR} et PCR_{AC} . L'implantation de ce NIOS utilise 2562 cellules logiques en ressources matérielles (soit 30,8% d'un circuit APEX 20K200) et 26496 bits de mémoire interne au FPGA, dont 8192 bits sont nécessaires au boot loader. Une mémoire externe de type SRAM est utilisée en tant que mémoire programme et mémoire de donnée pour stocker les variables. Le code à exécuter, fournissant les quatre paramètres relatifs au PCR, occupe environ 80000 bits.

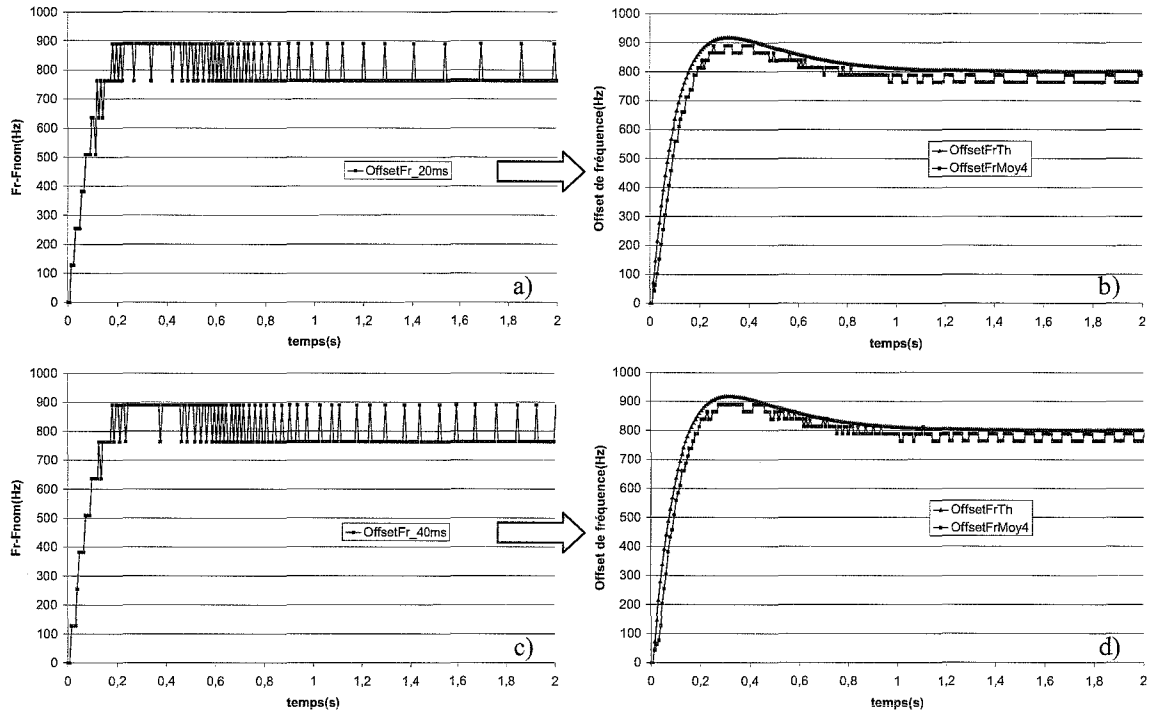


FIG. 3.14 – Comportement de l'ADPLL avec des PCRs interpolés pour plusieurs périodes d'échantillonnage. Réponse indicielle "brute" pour $T_{PCR} = 20ms$ (a), comparaison entre théorie et réponse indicielle expérimentale moyennée pour $T_{PCR} = 20ms$ (b), réponse indicielle "brute" pour $T_{PCR} = 40ms$ (c), comparaison entre théorie et réponse indicielle expérimentale moyennée pour $T_{PCR} = 40ms$ (d).

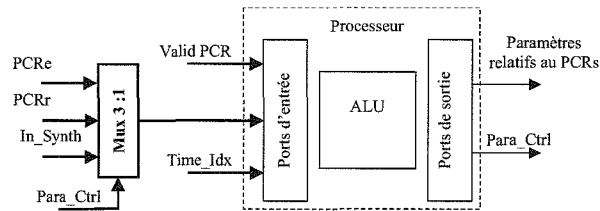


FIG. 3.15 – Interfaces du NIOS pour un programme.

Paramètres	PCR_FO	PCR_DR	PCR_OJ	PCR_AC
Durée max.	365 μs	400 μs	580 μs	1353 μs

TAB. 3.2 – Temps d'exécution de l'extraction des paramètres relatifs au PCR.

3.3.2 Temps d'exécution des filtres

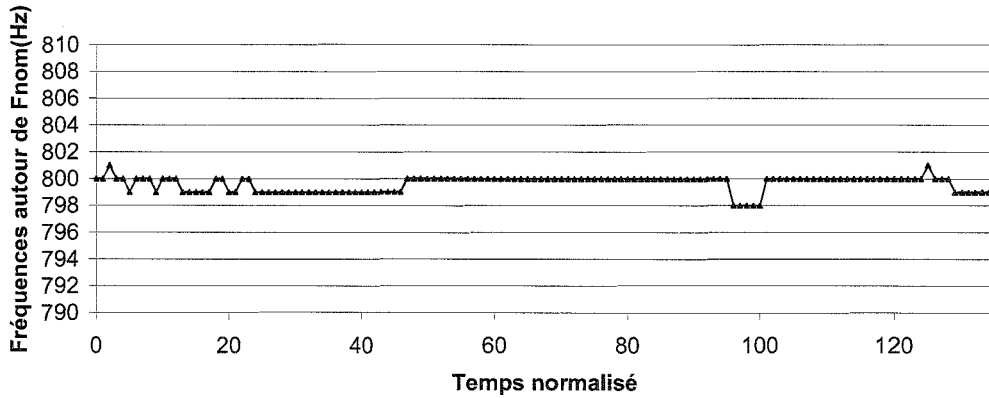
Les temps d'exécution maximaux des filtres de mesure fournissant les paramètres PCRs sont présentés dans le tableau 4.4. Ces valeurs sont données pour un NIOS fonctionnant à 33,333 MHz . Les trois premiers paramètres sont extraits à l'aide de l'ADPLL et de filtres passe-bas et passe-haut du premier ordre (voir annexe C). La durée pour exécuter chaque filtre va de 365 μs à 580 μs pour un programme. Le PCR_{AC} est, quant à lui, déduit des PCRs d'entrées, du temps séparant deux PCR_e successifs d'un même programme et à l'aide d'un filtre passe-haut du second ordre. Comme l'algorithme de calcul de ce paramètre est plus complexe, la durée d'extraction est supérieure à celle des autres paramètres et peut atteindre 1353 μs .

3.3.3 Résultats de mesure des paramètres relatifs au PCR

Plusieurs profils sont disponibles pour effectuer les mesures, comme mentionné dans la section 2.7.1. Leur choix repose sur un dilemme entre la précision des valeurs issues des filtres et leur temps de réponse (ou temps de convergence). Ainsi, pour une bonne précision sur les paramètres, c'est-à-dire sans ondulation sur la mesure, le temps de réponse doit être très long. Mais, par contre, il sera alors difficile de quantifier une erreur apparaissant sur une durée très courte.

L'extraction des paramètres est fournie avec une résolution de 1 Hz pour le PCR_{FO} , de 1 mHz/s pour le PCR_{DR} et de 1 ns pour le PCR_{OJ} et PCR_{AC} . La précision, quant à elle, est de 1 Hz pour le PCR_{FO} , de 1 mHz/s pour le PCR_{DR} et de $\frac{1}{F_{nom}} = 37 ns$ pour le PCR_{OJ} et le PCR_{AC} .

En imposant une fréquence de $F_{nom} + 800Hz$ sur la fréquence d'entrée, la mesure du PCR_{FO} donne les résultats de la figure 3.16 en utilisant un filtre du premier ordre, avec le profil $MGF3$. On constate la présence d'ondulations mentionnées ci-dessus. Lorsque le profil $MGF2$ est utilisé, les résultats de la figure 3.17 sont obtenus. Il est possible dans ce cas de constater que les ondulations disparaissent sur le PCR_{FO} (figure 3.17(a)), ce qui améliore la précision. Ces résultats sont obtenus lorsqu'un saut est produit sur la période d'échantillonnage, afin de valider la mesure des paramètres relatifs au PCR pour un débit variable. Ce saut n'a pas d'effet sur la mesure des paramètres PCR_{FO} , PCR_{DR} et PCR_{OJ} (figure 3.17, courbes a), c) et d)), déduit à partir de signaux recueillis sur l'ADPLL. Par contre, la courbe e) du PCR_{AC} de la figure 3.17 montre une discontinuité lors de l'apparition du saut sur la période d'échantillonnage. Cette mesure est réalisée à l'aide du dispositif de mesure de la figure 2.19 (chapitre 2). Ainsi, pour

FIG. 3.16 – Mesure du paramètre PCR_{FO} pour une fréquence d'entrée à $F_{nom} + 800Hz$.

Profils	$tr_{5\%}$ des filtres ($T_{PCR} = 40ms$)
$MGF1$ ($f_c = 10mHz$)	47,72 s
$MGF2$ ($f_c = 100mHz$)	4,80 s
$MGF3$ ($f_c = 1Hz$)	0,52 s

TAB. 3.3 – Temps de réponse à 5 % des filtres de mesure.

déterminer le PCR_{AC} , la différence de deux PCR_e successifs et celle de deux valeurs d'index de temps successives est effectuée. Ensuite, le résultat de ces deux différences est soustrait. Or, à ce niveau, en partant de l'hypothèse que la fréquence d'entrée F_e est constante, et lorsque la période T_{PCR} change, la soustraction fournit des valeurs différentes, car elle est proportionnelle à T_{PCR} . Ainsi, lors d'un changement de cette durée T_{PCR} , un saut apparaît sur la soustraction, qui se traduit par un artefact sur la courbe e) de la figure 3.17. Une solution pour remédier à ce problème est la normalisation de l'erreur, en utilisant le même principe appliqué sur l'ADPLL avec une normalisation de l'erreur de PCR (chapitre 2, figure 2.16). La figure 3.17(f) donne les résultats de mesure du PCR_{AC} lorsque cette solution est utilisée. En rappelant que le PCR_{AC} mesure la précision sur le PCR_e , il est possible de constater qu'un saut sur T_{PCR} ne produit plus d'artefact et que, là aussi, la précision sur le PCR dépend de son temps d'arrivée T_{PCR} . Comme on ne produit pas de gigue de réseau, la valeur moyenne du PCR_{OJ} et du PCR_{AC} doit être égale, ce qui est le cas ici.

Comme il est précisé plus haut, lorsque le profil $MGF2$ est employé, la précision des mesures est améliorée. En contre parti, les temps de convergence sont plus importants. Le tableau 3.3 présente les temps de réponse à 5% des différents filtres.

Pour valider le paramètre PCR_{DR} , mesurant la dérive sur la fréquence qui a générée les

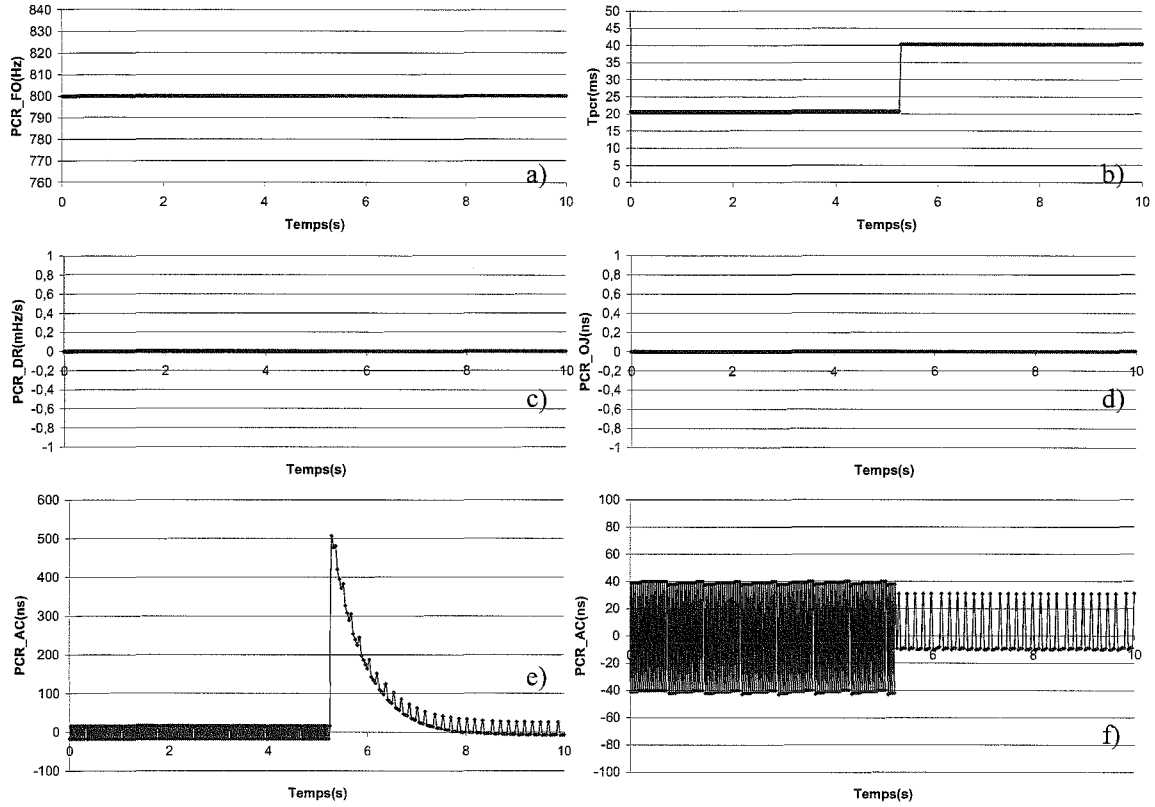


FIG. 3.17 – Mesure des paramètres PCR pour une fréquence d'entrée à $F_{nom} + 800Hz$ et pour un saut sur la période d'échantillonnage. PCR_{FO} (a), T_{PCR} (b), PCR_{DR} (c), PCR_{OJ} (d), PCR_{AC} avant normalisation (e), PCR_{AC} après normalisation (f).

PCR_e , une dérive connue doit être imposée sur l'entrée. En imposant une dérive arbitraire de $64 Hz/s$, et plus précisément de $63,578 Hz/s$, les résultats de la figure 3.18 sont obtenus. La mesure des paramètres est effectuée avec le profil $MGF2$ ($f_c = 0,1Hz$). La courbe a) donne le relevé du paramètre PCR_{FO} et montre la dérive appliquée sur l'entrée. Cette dérive est mesurée par le paramètre PCR_{DR} avec une ondulation résiduelle (figure 3.18(b)) et avec une valeur moyenne de $63,52 Hz/s$. Cela montre que le profil $MGF2$ est encore insuffisant pour la mesure du PCR_{DR} , si l'on souhaitait faire des mesures avec des ondulations très faibles. En comparant cette mesure avec la dérive réelle d'entrée, une erreur de $0,058 Hz/s$ est présente, soit une erreur relative de moins de $0,1 \%$. Cela permet d'affirmer que la mesure de dérives est effectuée avec un taux d'erreur très faible. Ces performances peuvent encore être améliorées, comme il sera possible de le voir par la suite en utilisant un profil de mesure plus adapté. Les courbes c) et d) de la figure 3.18 présentent la mesure de la gigue. Comme cette gigue est déduite de la phase échantillonnée d'une horloge à F_{nom} , l'erreur de phase peut être d'une période de

valeur $\frac{1}{F_{nom}}$. C'est ce que l'on peut constater sur ces courbes de gigue, qui présentent des pics de valeur pouvant aller jusqu'à 37 ns. La valeur moyenne de la gigue du paramètre PCR_{OJ} est plus faible que celle du paramètre PCR_{AC} (0,8 ns contre 11,8 ns respectivement) car le paramètre PCR_{AC} est directement déduit des PCR d'entrée alors que le paramètre PCR_{OJ} est déduit de l'erreur de phase de l'ADPLL, qui réduit la gigue d'entrée.

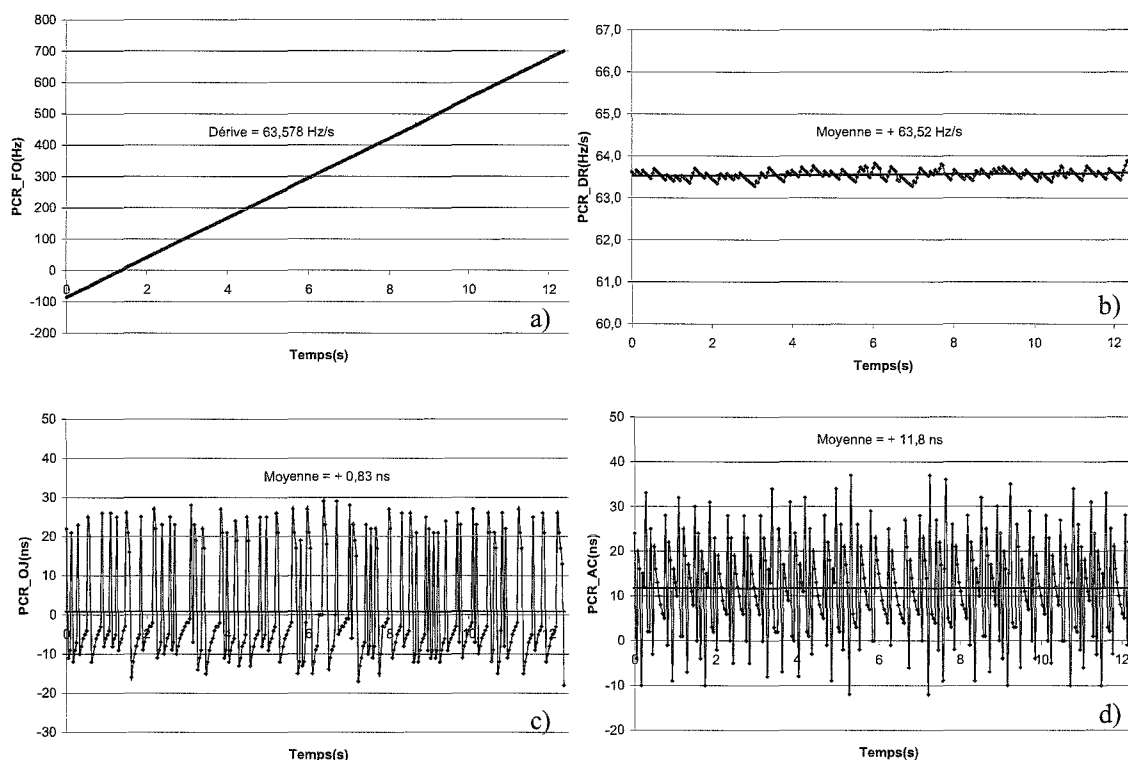


FIG. 3.18 – Mesure des paramètres relatifs au PCR pour une fréquence d'entrée dérivant à 64 Hz/s. $F_e - F_r$ (a), PCR_{FO} (b), T_{PCR} (c), PCR_{DR} (d), PCR_{OJ} (e) et PCR_{AC} (f).

La norme spécifie qu'il n'y a pas d'erreurs sur le paramètre PCR_{DR} s'il ne dépasse pas le seuil de 75 mHz/s. Ainsi, le système doit pouvoir mesurer des dérives inférieures à cette valeur. En imposant une dérive à l'entrée de 15 mHz/s, les résultats de la figure 3.19 permettent de voir le comportement de chaque paramètre, et ceci, avec une gigue de réseau nulle. Les filtres utilisés pour le PCR_{FO} et le PCR_{DR} ont une fréquence de coupure de 1 mHz (profil MGF4). La figure 3.19(a) donne l'évolution du PCR_{FO} . Ceci permet de voir que la fréquence d'entrée augmente graduellement au cours du temps avec la dérive imposée de 15 mHz/s. La mesure de la dérive à l'aide du paramètre PCR_{DR} (figure 3.19(b)) est maintenant effectuée avec une très grande précision, cela au mHz/s près, en utilisant un filtre adapté. Concernant la mesure du PCR_{OJ} , comme la gigue de réseau est nulle, la valeur moyenne du signal devra être nulle,

ce qui est vérifié par les résultats de la figure 3.19(c). Ces mesures sont réalisées à partir de l'erreur de PCR de l'ADPLL. Ainsi, les ondulations présentes de $\pm 36ns$ sont dues à l'erreur possible sur le PCR_error de l'ADPLL, à savoir de ± 1 , qui elle même est due à une constante re-synchronisation de l'ADPLL, au bout d'une durée T_{PCR} . La figure 3.19(d), quant-à elle, présente le PCR_AC . Comme ce dernier est une valeur entière, et que le signal $Valid_PCR$ apparaît entre deux fronts du signal d'horloge F_e , la précision est à une période près dans le pire des cas.

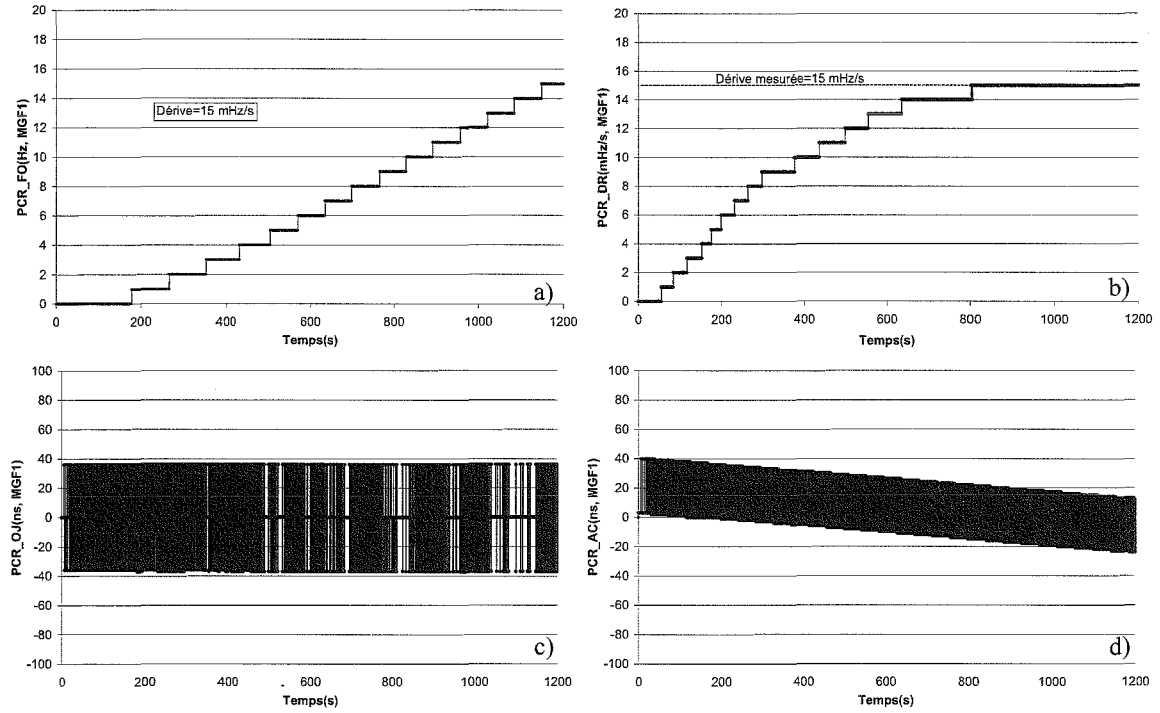


FIG. 3.19 – Mesure des paramètres relatifs au PCR pour une fréquence d'entrée dérivant à 15 mHz/s. PCR_FO (a), PCR_DR (b), PCR_OJ (c) et PCR_AC (d).

Il reste maintenant à valider les paramètres PCR_OJ et PCR_AC . L'un de nos partenaire, en l'occurrence TDF, dispose d'un matériel de mesure adéquat pour le test des paramètres relatifs au PCR. Ainsi, il caractérisera l'ensemble des paramètres relatifs au PCR.

3.3.4 Validation de la mesure des paramètres relatifs au PCR par le partenaire TDF

Le système de mesure des paramètres relatifs au PCR est mis à disposition à deux partenaires du projet : THALES B&M et Sodielec. THALES B&M, qui est intéressé depuis le début du projet par les paramètres relatifs au PCR, a fourni un démonstrateur au partenaire TDF. Ce

Seuil TR 101 290	Flux de test	Min (Hz)	Max (Hz)	Moy (Hz)	Résultats
810 Hz	PCR_FO_500Hz	447	447	447	OK
	PCR_FO_700Hz	647	648	647	OK
	PCR_FO_900Hz	847	847	847	OK
	PCR_FO_1000Hz	947	948	947	OK

TAB. 3.4 – Résultats de test du *PCR_FO*.

Seuil TR 101 290	Flux de test	Min	Max	Moy	Er (%)	Résultats
75 mHz	PCR_DR_150mhz/s	-160 mHz/s	161 mHz/s	-	7,3	OK

TAB. 3.5 – Résultats de test du *PCR_DR*.

dernier s'est proposé de tester le système de mesure en envoyant un flux MPEG-2 par paramètres à analyser directement sur le démonstrateur (mesures en point-à-point). Le flux MPEG-2 est ensuite monitoré par le démonstrateur de THALES, appelé AMETHYST sous sa version commerciale. Les tests sont effectués avec le profil *MGF1*, car il est le plus adapté pour avoir des résultats de mesures précis, résultats qui sont présentés ci-dessous.

3.3.4.1 Le paramètre *PCR_FO*

Le tableau 3.4 donne les résultats de test du paramètre *PCR_FO*. Le test est réalisé en générant plusieurs offsets de fréquence du côté émetteur. On constate qu'il existe un écart systématique de 53 Hz entre la fréquence de l'encodeur et la mesure sur le démonstrateur. Cet écart peut provenir de l'horloge système MPEG-2 de l'horloge de référence du synthétiseur dans l'ADPLL, ou des deux. Or, d'après la documentation constructeur du quartz utilisé pour l'horloge de référence, sa précision est très grande, ce qui permet de dire que le décalage de fréquence systématique de 53 Hz proviendrait du côté émetteur. Ainsi, le paramètre *PCR_FO* est validé.

3.3.4.2 Le paramètre *PCR_DR*

Pour le test du paramètre *PCR_DR*, le flux émis fait varier de manière sinusoïdale la fréquence générant les *PCR_e* avec une dérive maximale de 150 mHz. La figure 3.20 donne l'allure de la fréquence à l'aide du paramètre *PCR_FO*. La mesure de la dérive par le paramètre *PCR_DR* donne une valeur maximale de 161 mHz, comme le montre le tableau 3.5, ce qui reste dans l'ordre de grandeur de la dérive imposée. De ce fait, le paramètre *PCR_DR* est validé.

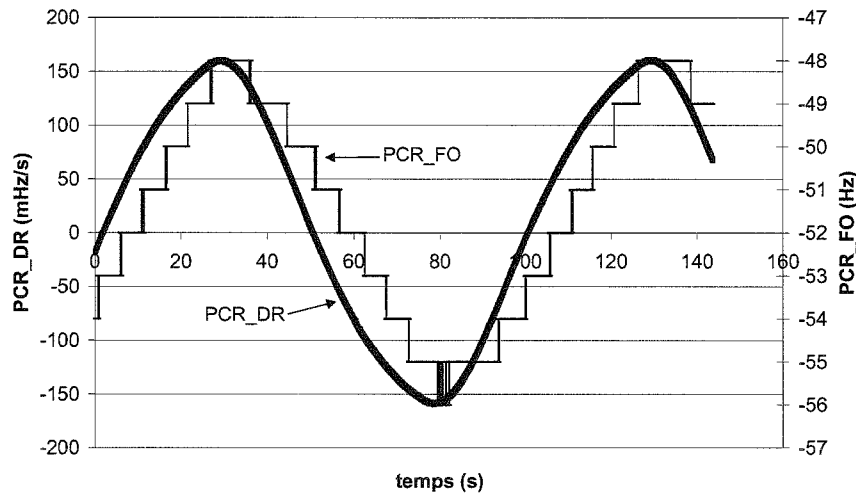


FIG. 3.20 – Allure du PCR_{FO} et du PCR_{DR} .

Seuil TR 101 290	Flux de test	Min (ns)	Max (ns)	Moy (ns)	Er (%)	Résultats
500 ns	PCR_AC_400ns	-651	639	420	5,0	OK
	PCR_AC_500ns	-799	824	490	2,0	OK
	PCR_AC_600ns	-1058	1007	600	0,0	OK
	PCR_AC_700ns	-1206	1055	675	3,6	OK

TAB. 3.6 – Résultats de test du PCR_{AC} .

3.3.4.3 Le paramètre PCR_{AC}

Le paramètre PCR_{AC} est testé en appliquant un bruit au PCR émis qui va engendrer une gigue calibrée sur l'ADPLL. Or, comme celle-ci est aussi sensible à la gigue de réseau, une architecture indépendante de la PLL est utilisée, permettant de s'affranchir de cette gigue de réseau. En émettant des flux avec des PCRs dont la précision est altérée et qui entraîne une gigue connue, on relève les résultats de mesure présentés dans le tableau 3.6. La différence entre la gigue calibrée fournie par le générateur de flux et la gigue moyenne mesurée est très faible, avec une erreur relative maximale de 5 %. Pour un système de mesure de la qualité de service, TDF a jugé cette erreur acceptable, et a donc validé le paramètre de mesure PCR_{AC} .

3.4 Conclusion

La validation du système de mesure des paramètres relatifs au PCR, intégrant une ADPLL, est effectuée dans ce chapitre. Cette validation concerne le test de l'ADPLL et de la mesure des

paramètres relatifs au PCR d'un programme, sachant que l'architecture finale comportera une chaîne de mesure identique par programme à analyser, ce qui se traduit par un comportement et des performances identiques pour chaque chaîne.

En terme de ressources, les aspects matériels, logiciels et temporels pour un programme sont étudiés. Ainsi, l'architecture de l'ADPLL consomme 20% de cellules logiques d'un circuit *APEX EP20K200*, ce qui montre que plusieurs ADPLLs peuvent être implantées dans un unique circuit FPGA. Concernant le traitement des paramètres relatifs au PCR, plusieurs solutions d'implantation existent comme l'utilisation d'un processeur externe (micro-contrôleur, DSP, ...) ou d'un processeur embarqué. Lorsqu'un processeur enfouis est utilisé pour l'extraction des paramètres de mesure, celui-ci nécessite 30,8% de cellules logiques FPGA d'un *APEX 20K200*, quelque soit le nombre de programmes à analyser, tant que le temps d'exécution du code respecte les contraintes de temps. Seuls les ressources mémoires requises augmenteront en fonction du nombre de programmes à traiter. Le fichier de configuration du FPGA correspondant au système de mesure traitant un programme requière 243769 octets des 1 *Mo* de mémoire flash disponible sur la plate-forme EXCALIBUR.

Ce chapitre a décrit la validation des éléments du système, dont le synthétiseur de fréquence. La mesure de la gigue intrinsèque a permis de valider la gigue estimée en théorie, déduite de son modèle mathématique. Ainsi, dans la plage où la norme considère qu'il n'y a pas d'erreur sur l'offset de fréquence, la gigue RMS intrinsèque est au maximum de 100 *ps*. Comme la résolution sur le PCR est d'une période, la gigue crête à crête introduite par la technique de décalage de phase n'affectera pas les PCRs. De plus, sa gigue crête à crête est de 9.280 *ns*, ce qui peut paraître comme une valeur importante. Cette valeur est cependant acceptable dans notre application car cette valeur de gigue crête à crête est compensée dans l'intervall T_{PCR} , et donc, n'aura pas de conséquences sur la valeur estimée des PCRs.

Ce synthétiseur de fréquence est utilisé dans plusieurs versions de l'ADPLL. La première version voit sa stabilité modifiée en fonction du temps d'arrivée des PCRs, mais en choisissant les valeurs adaptées aux coefficients du filtre de boucle, la stabilité peut être assurée dans la plage de variation de la période d'échantillonnage. La dépendance de la stabilité de l'ADPLL à la durée de la période T_{PCR} est liée à l'erreur de PCR qui est fonction de cette période d'échantillonnage. Ainsi, la seconde version proposée améliore la stabilité de l'ADPLL. Dans cette version, la stabilité de l'ADPLL est insensible aux variations du temps d'arrivée des PCRs. Cela est possible grâce à une normalisation de l'erreur de PCR. Cependant, les versions étudiées jusque là ont des bandes passantes qui varient avec la période T_{PCR} . Pour obtenir une bande passante constante quelque soit la durée T_{PCR} , une technique de sur-échantillonnage est utilisée qui, associée à la première version de l'ADPLL, permet d'appliquer à celle-ci une fréquence d'échantillonnage constante. L'intérêt d'avoir une bande passante constante est que le spectre des fréquences se trouvant à l'entrée de l'ADPLL est traité de manière identique quelque soit la

période d'échantillonnage.

Sur le plan de la mesure des paramètres relatifs au PCR, quatre mesures sont effectuées : l'offset de fréquence de l'horloge locale MPEG-2, sa dérive, l'imprécision du PCR qui se traduit par de la gigue, et la gigue cumulée provenant à la fois du réseau de transmission et de l'imprécision du PCR. Ces paramètres sont traités par un processeur NIOS qui utilise des valeurs numériques fournis par l'ADPLL. Comme la norme le précise, différentes fréquences de coupures peuvent être utilisées suivant que l'on privilégie le temps de réponse ou la précision. Si le système de mesure utilise l'une des deux ADPLLs sans sur-échantillonnage, et si l'on souhaite avoir une fréquence de coupure constante, il faut recalculer les coefficients des filtres en fonction du temps d'arrivée des PCRs, mais cela engendre un temps de traitement plus importants. Par contre, si le système de sur-échantillonnage est implanté, la bande passante sera toujours constante, ce qui ne nécessite pas de traitement supplémentaire. Cependant, avec la méthode de sur-échantillonnage, les contraintes de temps seront plus sévères.

Le système de mesure utilisant la première version de l'ADPLL, sans sur-échantillonnage, a été testé par un partenaire du projet, qui est spécialisé dans la caractérisation des paramètres intervenant dans l'évaluation de la qualité de service en télévision numérique. Ce partenaire a validé la mesure des paramètres relatifs au PCR.

Maintenant que le système de mesure des paramètres relatifs au PCR pour un programme est validé, il est nécessaire de trouver des solutions d'implantation de ce système de mesure pour l'ensemble des programmes constituant le flux de transport DVB-T. Pour cela, plusieurs solutions d'implantation sont possibles. Cependant, le système de mesure pour plusieurs programmes doit pouvoir être implanté dans un seul circuit, tout en respectant les contraintes de temps afin d'effectuer un traitement temps réel de tous les programmes du flux MPEG-2 DVB-T. De ce fait, certaines solutions d'implantation seront plus avantageuses que d'autres comme il sera possible de le constater dans le chapitre suivant.

Chapitre 4

Optimisation des ressources du système de mesure

4.1 Introduction

Le chapitre précédent a permis de valider le système de mesure pour un programme. L'étude des ressources a montré que le système de mesure de paramètres relatifs au PCR pour un programme peut largement prendre place dans le circuit utilisé pour l'expérimentation. Cependant, si l'on veut analyser les paramètres relatifs au PCR de chaque programme d'un flux DVB-T pouvant contenir jusqu'à 10 programmes, une exploration de l'espace de conception doit être menée pour optimiser l'implantation globale de l'architecture en terme de ressources tout en respectant les contraintes de temps [MRT⁺05, MRW⁺06].

En prenant en compte dans cette étude la première version de l'ADPLL sans l'utilisation du sur-échantillonnage, cela permettra d'avoir un premier aperçu des ressources nécessaires à l'implantation du système complet. Concernant les contraintes de temps, certains éléments de cette ADPLL fonctionnent sur de fortes contraintes de temps. Ainsi, concernant l'optimisation des ressources de l'ADPLL pour plusieurs programmes, certains éléments de l'ADPLL ne pourront pas être réutilisés.

4.2 Optimisation pour l'implantation d'un système de mesure 10 PCRs

L'utilisation d'un processeur enfoui NIOS implique l'utilisation de ressources FPGAs systématiques appelées R_{nios} . L'envoi des données brutes (non filtrées) entre une ADPLL et le NIOS est réalisé à l'aide d'un multiplexeur 4 : 1 sur 32 bits et consomme $R_{m(4:1)}$ cellules logiques (pour un programme).

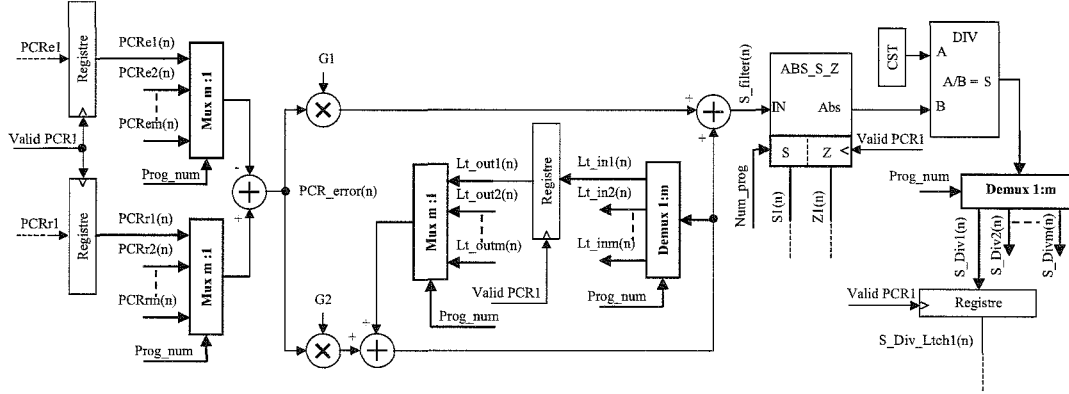


FIG. 4.1 – Optimisation avec réutilisation d'opérateurs.

Dans l'évaluation des ressources, trois cas sont à distinguer : le premier cas correspond uniquement à l'estimation des ressources FPGAs concernant l'implantation de m ADPLLs. Ce cas permettra de connaître les ressources FPGAs pour chaque solution d'implantation proposée, sachant que les optimisations sont relatives aux ADPLLs. Le second cas correspond à l'utilisation d'un processeur externe pour les traitements relatifs aux filtres de mesure. Le dernier cas intégrera un processeur FPGA enfoui pour l'exécution de ces filtres. La distinction de ces deux derniers cas a comme objectif de justifier ou non l'intégration d'un processeur enfoui suivant les ressources FPGAs disponibles pour un circuit donné.

4.2.1 Implantation parallèle

L'une des solutions d'implantation consiste à dupliquer les ADPLLs en fonction du nombre de programmes existant dans le flux. Soit R_{ad1} , les ressources FPGAs d'une ADPLL et R_{adm} , les ressources FPGAs de m ADPLLs, la relation de l'équation 4.1 montre que cette solution entraîne une consommation linéaire des ressources FPGA.

$$R_{adm} = R_{ad1} \cdot m \quad (4.1)$$

Si le processeur n'est pas implanté dans le circuit FPGA, il faut tenir compte des ressources utilisées pour la mise en place de multiplexeurs afin d'aiguiller toutes les données PCR brutes (issues des ADPLLs) vers un port unique du circuit FPGA. Le processeur pourra ainsi récupérer directement les données sur un port qui est dédié à la récupération des paramètres relatifs au PCR. Une solution est de mettre en place des multiplexeurs 4 : 1 réservés à chacune des m ADPLLs et d'utiliser un multiplexeur m : 1 consommant $R_{m(m:1)}$ cellules et permettant de sélectionner l'une des sorties des m multiplexeurs 4 : 1 pour ne former plus qu'un bus. La mise

en place de cette structure consomme les ressources données par l'équation 4.2.

$$R_{mes_ext_m} = R_{ad_m} + R_{m(4:1)} \cdot m + R_{m(m:1)} \quad (4.2)$$

Dans le cas où le traitement des filtres est réalisé par un processeur enfoui, les ressources consommées sont données par l'équation 4.3.

$$R_{mes_nios_m} = R_{mes_ext_m} + R_{nios} \quad (4.3)$$

Cette solution utilisant des ADPLLs fonctionnant en parallèle va entraîner une consommation des ressources la plus élevée. Par contre, comme il n'y a pas d'optimisation de ressources FPGAs au niveau des ADPLLs et donc de réutilisation d'opérateurs, les ADPLLs vont fonctionner de manière autonome et les PCR de chaque programme seront traités indépendamment.

4.2.2 ADPLLs avec réutilisation d'opérateurs matériels

Une optimisation possible est la réutilisation d'opérateurs matériels en insérant des multiplexeurs. En effet, l'ADPLL utilise des éléments qui ne fonctionnent pas sur de fortes contraintes de temps. Il s'agit du filtre de boucle, du comparateur de PCR et des deux premiers éléments se trouvant à l'entrée du synthétiseur de fréquence, à savoir la fonction *ABS_S_Z* et la fonction division. Ces éléments sont rythmés par un signal *Valid_PCR* dont la période minimale est de 10 *ms*. Des multiplexeurs seront placés après les registres contenant les PCR et après le registre du filtre de boucle, comme le montre la figure 4.1. De plus, deux démultiplexeurs sont nécessaires pour modifier les valeurs respectives des registres affectés à chaque programme, un démultiplexeur consommant $R_{d(1:m)}$ cellules logiques. L'un des registres concernés est celui se trouvant dans le filtre de boucle et l'autre se situe après le diviseur du synthétiseur. Les éléments des ADPLLs fonctionnant avec des contraintes de temps importantes seront dupliqués en fonction du nombre d'horloges programmes à restituer. Toutes les parties à dupliquer ne sont pas présentées sur la figure 4.1.

La mise en place de cette solution d'optimisation pour m ADPLLs utilise les ressources données par l'équation 4.4.

$$R_{ad_m} = R_{ad_ru} + R_{ad_d} \cdot m + 3R_{m(m:1)} + 2R_{d(1:m)} \quad (4.4)$$

R_{ad_ru} correspond aux ressources réutilisées et R_{ad_d} correspond aux ressources à dupliquer.

4.2.3 Implantation séquentielle

Une autre solution qui peut être utilisée consiste à remplacer les opérateurs à faible contrainte de temps qui sont réutilisés dans la méthode de le paragraphe 4.2.2 par des opérateurs séquentiels. Ces opérateurs pourront être traités par un processeur enfoui NIOS, comme le montre la

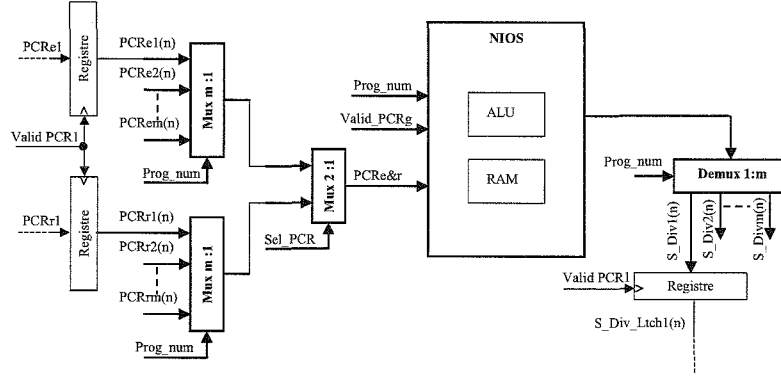


FIG. 4.2 – Optimisation avec un processeur NIOS.

figure 4.2. La mémoire programme du NIOS sera embarquée dans le FPGA. Des multiplexeurs seront utilisés pour amener les valeurs de PCRe et PCRr de chaque programme au niveau du NIOS et des démultiplexeurs sont mis en place pour envoyer les signaux de commande aux synthétiseurs de fréquence de chaque programme.

Les ressources requises pour cette méthode de traitement sont données par l'équation 4.5. R_{ad_d} correspond à la même structure à dupliquer que dans le cas d'utilisation de multiplexeur uniquement (paragraphe 4.2.2). Dans cette configuration, le multiplexeur et le démultiplexeur du paragraphe 4.2.2 sont remplacés par des fonctions logicielles exécutées par le NIOS. De plus, la division n'est pas implantée directement, car elle est réalisée de manière séquentielle par l'ALU du NIOS, ce qui permet encore un gain de ressources combinatoires. De surcroît, les multiplexeurs servant à aiguiller les données de chaque programme vers le processeur de traitement des paramètres relatifs au PCR (cas des paragraphes 4.2.1 et 4.2.2) serviront en même temps à alimenter le NIOS réalisant les opérations de l'ADPLL de chaque programme. En contre partie, ces gains peuvent être perdus car l'utilisation d'un NIOS requière des ressources FPGAs supplémentaires utilisées pour la mémoire programme.

$$R_{ad_m} = R_{nios_ad} + R_{ad_d} \cdot m + 2R_{m(m:1)} + R_{d(1:m)} \quad (4.5)$$

4.3 Estimation des ressources matérielles et logicielles

4.3.1 Ressources d'une implantation parallèle

Cette solution consomme le plus de ressources de toutes les solutions présentées. Dix ADPLLs totalement autonomes nécessitent 16800 cellules logiques, comme le montre le tableau 4.1. Le recours à un processeur externe pour le traitement des filtres de mesure nécessite l'ajout de ressources supplémentaires pour l'aiguillage des données. Cette solution occupe 17596 cellules

logiques et utilise de la mémoire programme qui n'est pas possible d'estimer car l'occupation mémoire dépend du processeur mis en place. Lorsque l'on opte pour l'utilisation d'un circuit FPGA seul, avec un NIOS embarqué qui consomme 2562 cellules logiques et 26496 bits de mémoire FPGA, le système de mesure requière 20158 cellules logiques et 8904 octets de mémoire programme en plus de la mémoire FPGA utilisée. Pour implanter cette solution, il est nécessaire d'utiliser au moins un circuit APEX EP20K600 avec 24320 cellules logiques et 311296 bits de mémoire. Le taux d'occupation de l'APEX est alors de 83 % de cellules logiques et 9 % de mémoire.

	Ressources Logiques (LC)	Mémoire FPGA (bits)	Mémoire Externe (bits)
10.ADPLLs	16800	0	0
10 ADPLLs + EXT_CPU	17596	0	-
10.ADPLLs + NIOS	20158	26496	80000

TAB. 4.1 – ADPLLs implantés parallèlement.

4.3.2 Ressources ADPLLs avec réutilisation d'opérateurs

La solution de réutilisation d'opérateurs à l'aide de multiplexeurs et de demultiplexeurs permet de diminuer considérablement les ressources utilisées, comme le montre le tableau 4.2. Dans le cas d'ADPLLs implantés sans processeur de traitement des PCR, l'implantation requière 8850 cellules logiques au lieu de 16800 cellules logiques, ce qui représente un gain de 47,3 % par rapport à la solution d'implantation des ADPLLs en parallèle. Lorsqu'un processeur externe est utilisé pour l'extraction des paramètres relatifs au PCR, le système de mesure requière globalement 9646 cellules logiques, soit un gain de 45,2 % et lorsque ce processeur est un NIOS, le système nécessite 12208 cellules logiques. Dans ce dernier cas, il sera possible d'implanter le système dans un circuit de taille inférieure à l'APEX EP20K600, à savoir un circuit APEX EP20K400 disposant de 16640 cellules logiques et de 212992 bits de mémoire. Le taux d'occupation de cet APEX est alors de 73 % de cellules logiques et 13 % de mémoire.

4.3.3 Ressources ADPLLs utilisant un processeur

L'utilisation d'un processeur NIOS pour le remplacement des opérations à faibles contraintes de temps des ADPLLs permet un gain similaire à la solution précédente lorsque les ADPLLs sont implantées seules d'après le tableau 4.3. Quand le système de mesure est implanté, que ce soit avec processeur externe ou un NIOS pour effectuer le traitement de paramètres relatifs au PCR, le gain est de deux à trois points par rapport à la solution à réutilisation d'opérateurs

	Ressources Logiques (LC)	Mémoire FPGA (bits)	Mémoire Externe (bits)	Gains en LC (%)
10.ADPLLs	8850	0	0	47,3
10 ADPLLs + EXT_CPU	9646	0	-	45,2
10.ADPLLs + NIOS	12208	26496	80000	39,4

TAB. 4.2 – ADPLLs implantées avec réutilisation d'opérateurs.

par multiplexeur-démultiplexeur. Ce gain en ressources combinatoires se fait au détriment d'une utilisation plus importante des ressources mémoires FPGA (42880 bits en plus). La mémoire programme nécessaire au NIOS remplaçant les opérateurs à faibles contraintes de temps requière 4068 octets, soit une mémoire de 4 Koctets, qui sont compris dans les 42880 bits consommés par ce NIOS. Dans le cas où les ADPLLs sont implantées avec les deux NIOS, le recours à un circuit APEX EP20K400 au moins est nécessaire avec un taux d'occupation de 70 % de cellules logiques et 33 % de mémoire.

	Ressources Logiques (LC)	Mémoire FPGA (bits)	Mémoire Externe (bits)	Gains en LC (%)
10.ADPLLs	8879	42880	0	47,1
10 ADPLLs + EXT_CPU	9163	42880	-	47,9
10.ADPLLs + NIOS	11725	69376	80000	41,8

TAB. 4.3 – ADPLLs implantées avec un NIOS.

4.3.4 Temps d'exécution

Dans le cas de l'optimisation des ressources par la ré-utilisation d'opérateurs, les contraintes de temps sont respectées. Par contre, il est nécessaire de faire une étude dans le cas du NIOS inséré dans la PLL et dans le cas où le NIOS traite les PCRs.

4.3.4.1 Cas du NIOS inséré dans la PLL

Le processeur NIOS est inséré dans la boucle et est cadencé par une horloge de 33,333 MHz. La durée nominale d'un cycle de calculs dure 24 μ s lorsque le PCR d'entrée s'incrémente toujours de la même valeur (sans variation de fréquence en entrée). La durée maximale de 47 μ s apparaît lorsque le PCR passe de sa valeur maximale à sa valeur minimale. Cette durée maximale ne dépasse pas le temps critique où deux PCRs de deux programmes arriveraient sur deux trames successives de 188 octets à 27 Mbits de débit.

4.3.4.2 Cas du NIOS traitant les PCRs

Les temps d'exécution maximaux des filtres de mesure fournissant les paramètres relatifs au PCR sont présentés dans le tableau 4.4. Ces valeurs sont données pour un NIOS fonctionnant à 33,333 *MHz*. Les trois premiers paramètres sont extraits à l'aide de l'ADPLL et de filtres passe-bas et passe-haut du premier ordre. La durée pour calculer chaque valeur va de 365 μs à 580 μs pour un programme. Le *PCR_AC* est, quand à lui, déduit des PCRs d'entrées, du temps séparant deux PCRe successifs d'un même programme et à l'aide d'un filtre passe-haut du second ordre. Comme l'algorithme de calcul de ce paramètre est plus complexe, la durée d'extraction est supérieure aux autres paramètres et peut atteindre 1353 μs . Pour respecter les contraintes de temps, à savoir le traitement des filtres de mesure en moins de 10 *ms* pour tous les programmes d'un flux DVB-T, il faut utiliser un NIOS fonctionnant au moins à 50 *MHz* pour le traitement des paramètres *PCR_FO*, *PCR_DR* et *PCR_OJ* et un autre NIOS dédié au traitement du *PCR_AC* et fonctionnant aussi au moins à 50 *MHz*. Le test des contraintes de temps est effectué avec un APEX 2x, qui n'est pas le circuit le plus rapide de sa gamme. Les APEX 1x sont des circuits plus rapides et avec ce type de circuit, le NIOS peut être cadencé à 80 *MHz*.

Paramètres relatifs au PCR	PCR_FO	PCR_DR	PCR_OJ	PCR_AC	Total
Durée max.	365 μs	400 μs	580 μs	1353 μs	2698 μs

TAB. 4.4 – Temps d'exécution des filtres de paramètres relatifs au PCR.

4.4 Discussion

L'étude résultante de l'optimisation des ressources et des contraintes de temps permet de choisir le meilleur compromis entre le temps d'exécution et la surface de l'architecture à implanter. Ces résultats sont résumés par la figure 4.3. Cette figure présente le temps d'exécution critique du traitement de la boucle de l'ADPLL en fonction des ressources du système complet, dans le cas où le processeur exécutant les paramètres relatifs au PCR est embarqué dans le FPGA. Le temps d'exécution limite (deadline) de 55.7 μs correspond au temps maximal disponible qui correspond au cas critique où les PCRs de programme différents arrivent dans deux paquets de transport successifs. Lorsque toutes les ADPLLs sont implantées en parallèle, le temps d'exécution est très inférieur à la deadline, mais l'architecture consomme le plus de ressources matérielles. Dans le cas de l'implantation matérielle/logicielle de l'ADPLL utilisant un processeur embarqué pour le traitement de la boucle, cadencé à 33.333 *MHz*, la contrainte de

temps est tout juste respectée. De meilleurs résultats sont obtenus avec le même NIOS rythmé à 54 *MHz*, mais la meilleure solution est basée sur la réutilisation des opérateurs matériels, qui a de bonnes performances en terme de contraintes de temps et la consommation de ressources la plus basse.

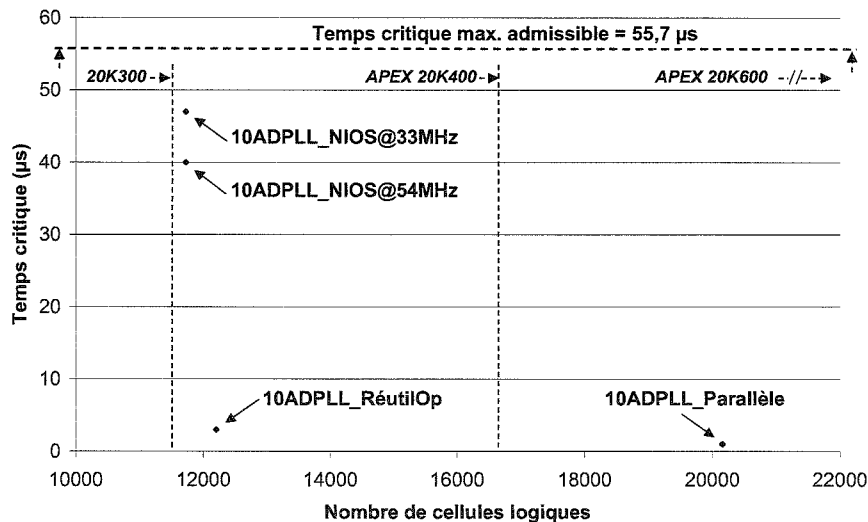


FIG. 4.3 – Temps d'exécution du traitement de la boucle de l'ADPLL en fonction des ressources de l'architecture globale.

4.5 Conclusion

Dans ce chapitre, plusieurs solutions d'implantation FPGA du système de mesure des paramètres relatifs au PCR pour l'ensemble des programmes d'un flux DVB-T sont présentées. Ces solutions permettent d'implanter autant d'ADPLLs qu'il y a de programmes dans un flux DVB-T. Le but est d'effectuer des mesures en temps réel des paramètres relatifs au PCR pour chaque programme du flux en vue de l'évaluation de la qualité de service. Cette étude permet de quantifier les ressources FPGAs nécessaires à l'implantation du système de mesure en fonction du processeur utilisé pour l'extraction des paramètres et de la méthode d'implantation des ADPLLs. Avec les optimisations présentées, lorsque l'on souhaite implanter le système de mesure sans circuit externe, un circuit APEX EP20K400 pourra être utilisé, dans la catégorie de vitesse 1x. Cela permettra d'utiliser un seul processeur embarqué, fonctionnant à 80 *MHz*, pour traiter l'ensemble des paramètres relatifs au PCR de chaque programme d'un flux. De plus, il sera plus avantageux de choisir l'optimisation reposant sur la réutilisation d'opérateurs. En effet, contrairement à l'utilisation d'un NIOS dans la boucle de l'ADPLL, cette solution ne fonctionne pas sur de fortes contraintes de temps et n'utilise pas de mémoire FPGA supplémentaire.

Conclusion

Dans le domaine de l'évaluation de la qualité de service en télévision numérique, plusieurs paramètres sont à analyser, qui sont soit des indicateurs, soit des mesures de grandeurs. La description de l'ensemble de ces paramètres est fournie dans des guides [ETS05, ETS01], qui sont normalisés par ETSI. Parmi ces paramètres, on retrouve la mesure de grandeurs relatives à des horodateurs, appelés PCRs, qui sont requis pour la synchronisation de l'horloge système d'un décodeur MPEG-2. Ces horodateurs peuvent être affectés par la gigue de réseau, ce qui peut influencer le processus de synchronisation. Ainsi, la mise en place de paramètres de mesures relatifs au PCR permet d'évaluer la qualité de synchronisation de l'horloge du décodeur. Or, l'implantation d'un système de mesure complet pour la mesure des paramètres relatifs au PCR de chaque programme nécessite l'implantation d'une unité de synchronisation pour chaque programme, ce qui représente une tâche difficile si l'on a comme objectif d'employer un unique circuit.

Une étude de l'état de l'art des techniques adaptées, ou pouvant être facilement adaptées, à la synchronisation sur des horodateurs a permis de recenser plusieurs méthodes de synchronisation. Les deux techniques les plus connues utilisent le traitement statistique du signal, reposant sur le calcul d'une régression linéaire, ou la synchronisation à l'aide d'une boucle à verrouillage de phase. Cette dernière a l'avantage de comparer directement la phase échantillonnée, tout en nécessitant très peu de ressources. De plus, comme les paramètres sont déduits de l'erreur de phase et du signal de commande d'un VCO (sauf pour le *PCR_{AC}* qui n'utilise pas d'unité de synchronisation), elle est directement adaptée à la mesure de ces paramètres. Cependant, une difficulté se présente en utilisant une PLL. En effet, si l'on souhaite implanter autant de PLL qu'il y a de programmes à analyser, cela signifie qu'il faudrait aussi autant de VCO externe. Ainsi, si l'on désire implanter le système dans un unique circuit, il sera judicieux de recourir à des architectures totalement numériques. Ainsi, un état de l'art sur les techniques de génération d'horloge contrôlée en fréquence et/ou en phase, reposant sur des architectures analogiques et/ou numériques, a permis de comparer différentes solutions pour générer une base de temps contrôlée. De manière générale, deux possibilités existent pour agir sur la fréquence d'un signal : soit on modifie la fréquence en intervenant directement dans l'architecture de l'oscillateur, soit

on synthétise le signal issu d'un oscillateur. Cette étude montre que dans le cas de générateurs numériques, la phase, qui subit des variations discontinues, peut entraîner l'instabilité de la PLL. De plus, ces variations discontinues auront un impact sur la gigue de la PLL. De ce fait, le générateur de base de temps représente souvent l'un des points sensibles d'une PLL.

Ainsi, cette thèse présente l'étude d'une architecture totalement numérique de mesure des paramètres relatifs au PCR, dont le coeur est constitué de plusieurs ADPLLs. Chacune de ces ADPLLs utilise un synthétiseur de fréquence original. Ce dernier est caractérisé en terme de gigue à l'aide d'un modèle mathématique. L'étude montre que dans l'intervalle de fréquence que la norme tolère, cette gigue n'a pas d'influence sur les PCRs générés localement et donc n'affectera pas la mesure des paramètres relatifs au PCR. Aussi, l'ADPLL présentée peut fonctionner avec un débit de PCR constant ou variable, tout en assurant sa stabilité. De plus, sa bande passante est rendue constante par l'utilisation d'une technique de sur-échantillonnage, associée à un dispositif d'interpolation des PCRs pour fournir des PCRs à un rythme régulier. A partir de l'erreur de PCR et de la commande du synthétiseur, trois des quatre paramètres relatifs au PCR peuvent être déterminés. Le quatrième paramètre, qui correspond à la mesure de la précision du PCR, est extrait à l'aide d'un algorithme qui effectue la mesure indépendamment de la gigue de réseau. L'extraction de ces paramètres se fait par l'intermédiaire de filtres de mesure dont les fréquences de coupure sont répertoriées dans des profils présentés dans la norme ETSI. Chaque profil répond à un besoin, qui repose sur un dilemme entre le temps de convergence et la précision des mesures.

Le système est ensuite implanté dans un circuit FPGA. Les ressources consommées par une ADPLL, qui nécessite 20,2 % d'un circuit APEX 20K200, montrent que plusieurs ADPLLs peuvent prendre place dans un unique circuit. La gigue théorique, intrinsèque au synthétiseur de fréquence, et le comportement de l'ADPLL déduit du modèle, sont en corrélation avec les résultats expérimentaux, ce qui permet de valider le modèle théorique. Plusieurs architectures d'ADPLL sont proposées pour répondre à des contraintes de stabilité et de bande-passante. Concernant la gigue intrinsèque au synthétiseur, la mesure de la gigue RMS est au maximum de 100 ps dans la plage de fréquence dans laquelle la norme considère qu'il n'y a pas d'anomalie sur l'offset de fréquence. Cette gigue est ainsi négligeable, sachant que la résolution sur le PCR est d'une période de l'horloge qui les a générés, et n'aura donc pas d'impact sur la mesure des paramètres.

Comme la valeur du PCR généré localement n'est pas influencée par les décalages de phase du synthétiseur, la mesure des paramètres relatifs au PCR n'est pas altérée. Ces paramètres sont extraits à l'aide de filtres implantés dans un processeur enfouis NIOS. Ces filtres ont des fréquences de coupure définie par la norme ETSI, reposant sur un dilemme entre temps de convergence et précision des mesures. L'ensemble de ces paramètres a été validé par un partenaire du projet.

Une exploration de l'espace de conception permet d'étudier plusieurs solutions de l'implan-

tation du système de mesure complet. Cette étude prend en compte les ressources matérielles, logicielles et temporelles du système. La solution retenue est basée sur l'implantation d'ADPLLs avec un filtre de boucle implanté matériellement et partagé. Cela est réalisé par un aiguillage des données des ADPLLs par des multiplexeurs-démultiplexeurs. Ainsi, la mise en place de cette optimisation permet d'utiliser un seul circuit FPGA et, d'après l'étude des ressources matérielles et temporelles, un circuit APEX EP20K400 peut recevoir l'ensemble du système de mesure des paramètres relatifs au PCR.

Conclusion

Annexe A

Démonstration de la fonction de transfert du synthétiseur de fréquence

La fréquence moyenne F_r après une durée T_{PCR} est :

$$F_r = \frac{N_r}{T_{PCR}} \quad (\text{A.1})$$

où N_r est égal à :

$$N_r = N_{nom} + N_{nom \pm \delta} \quad (\text{A.2})$$

$N_{nom \pm \delta}$ est égal à :

$$N_{nom \pm \delta} = \frac{m \cdot F_{nom} \cdot T_{PCR}}{|TH(n-1)|} \quad (\text{A.3})$$

N_{nom} est égal à :

$$N_{nom} = \frac{T_{PCR} - N_{nom \pm \delta} \cdot (T_{nom} \pm \delta T)}{T_{nom}} \quad (\text{A.4})$$

N_{nom} est aussi égal à :

$$N_{nom} = T_{PCR} \cdot F_{nom} \left(1 - \frac{m \pm 1}{|TH(n-1)|}\right) \quad (\text{A.5})$$

D'après les équations A.3 et A.5, N_r est égal à :

$$N_r = T_{PCR} \cdot F_{nom} \left(1 \pm \frac{1}{|TH(n-1)|}\right) \quad (\text{A.6})$$

En remplaçant $|TH(n-1)|$ par son expression 2.9, l'équation A.6 devient :

$$N_r = T_{PCR} \cdot F_{nom} \left(1 \pm \frac{|In_Synth(n-1)|}{CST}\right) \quad (\text{A.7})$$

d'où l'expression de $F_r(n)$:

$$F_r(n) = F_{nom} \left(1 + \frac{In_Synth(n-1)}{CST}\right) \quad (\text{A.8})$$

Annexe B

Démonstration de la stabilité de l'ADPLL

B.1 ADPLL

La fonction de transfert de l'ADPLL est rappelée par l'équation B.1. Cette fonction correspond à un système numérique du troisième ordre.

$$H(z) = H_0 \times \frac{a_1.z + a_0}{z^3 + b_2.z^2 + b_1.z + b_0} \quad (\text{B.1})$$

avec :

$$H_0 = \frac{F_{nom}.T_{PCR}}{CST} \quad (\text{B.2})$$

$$a_0 = -G_1 \quad (\text{B.3})$$

$$a_1 = G_1 + G_2 \quad (\text{B.4})$$

$$b_0 = -\frac{F_{nom}.T_{PCR}.G_1}{CST} \quad (\text{B.5})$$

$$b_1 = 1 + \frac{F_{nom}.T_{PCR}.(G_1 + G_2)}{CST} \quad (\text{B.6})$$

$$b_2 = -2 \quad (\text{B.7})$$

La stabilité de l'ADPLL est analysée à l'aide du critère de Jury pour un système du troisième ordre. Soit $D(z)$, le dénominateur de $H(z)$:

$$D(z) = z^3 + b_2.z^2 + b_1.z + b_0 \quad (\text{B.8})$$

Les conditions du critère de Jury sont :

– Condition 1 :

$$|b_0| < 1 \quad (\text{B.9})$$

– Condition 2 :

$$D(1) > 0 \quad (\text{B.10})$$

– Condition 3 :

$$D(-1) < 0 \quad (\text{B.11})$$

– Condition 4 :

$$b_0^2 - 1 < b_0 \cdot b_2 - b_1 \quad (\text{B.12})$$

L'étude de la stabilité de l'ADPLL donne les résultats suivants :

– Condition 1 : En remplaçant b_0 par son expression, la condition 1 devient :

$$\left| -\frac{F_{nom} \cdot T_{PCR} \cdot G_1}{CST} \right| < 1 \quad (\text{B.13})$$

Ainsi, la contrainte déduite de la condition 1 est :

$$G_1 < \frac{CST}{F_{nom} \cdot T_{PCR}} \quad (\text{B.14})$$

– Condition 2 : Après une première simplification, la condition 2 devient :

$$D(1) = \frac{F_{nom} \cdot T_{PCR} \cdot G_2}{CST} > 0 \quad (\text{B.15})$$

Ainsi, la condition 2 donne l'inégalité suivante :

$$G_2 > 0 \quad (\text{B.16})$$

– Condition 3 : la condition 3 devient, après simplification :

$$D(-1) = -4 - \frac{F_{nom} \cdot T_{PCR} \cdot (2G_1 + G_2)}{CST} < 0 \quad (\text{B.17})$$

Ainsi, la condition 3 est :

$$2G_1 + G_2 > -\frac{4 \cdot CST}{F_{nom} \cdot T_{PCR}} \quad (\text{B.18})$$

– Condition 4 : après une première simplification, nous avons :

$$\left(\frac{F_{nom} \cdot T_{PCR} \cdot G_1}{CST} \right)^2 < \frac{F_{nom} \cdot T_{PCR}}{CST} \cdot (G_1 - G_2) \quad (\text{B.19})$$

La condition 4 devient :

$$G_1^2 < \frac{CST}{F_{nom} \cdot T_{PCR}} \cdot (G_1 - G_2) \quad (\text{B.20})$$

De cette étude de stabilité, il est possible de déduire l'inégalité suivante :

$$0 < G_2 < \frac{G_1}{1 - \frac{G_2}{G_1}} < \frac{CST}{F_{nom} \cdot T_{PCR}} \quad (\text{B.21})$$

B.2 ADPLL avec normalisation de l'erreur de PCR

Dans le cas de l'ADPLL "améliorée", les coefficients de la fonction de transfert (équation B.1) sont les suivants :

$$H_0 = \frac{F_{nom}}{F_{clk_N}} \cdot \frac{CST_N}{CST} \quad (B.22)$$

$$a_0 = -G_1 \quad (B.23)$$

$$a_1 = G_1 + G_2 \quad (B.24)$$

$$b_0 = -\frac{F_{nom}}{F_{clk_N}} \cdot \frac{CST_N}{CST} \cdot G_1 \quad (B.25)$$

$$b_1 = 1 + \frac{F_{nom}}{F_{clk_N}} \cdot \frac{CST_N}{CST} \cdot (G_1 + G_2) \quad (B.26)$$

$$b_2 = -2 \quad (B.27)$$

Les résultats de l'étude de stabilité sont donnés ci-dessous :

– Condition 1 :

$$G_1 < \frac{F_{clk_N} \cdot CST}{F_{nom} \cdot CST_N} \quad (B.28)$$

– Condition 2 :

$$G_2 > 0 \quad (B.29)$$

– Condition 3 :

$$2G_1 + G_2 > -\frac{4F_{clk_N} \cdot CST}{F_{nom} \cdot CST_N} \quad (B.30)$$

– Condition 4 :

$$G_1^2 < \frac{F_{clk_N} \cdot CST}{F_{nom} \cdot CST_N} \cdot (G_1 - G_2) \quad (B.31)$$

Dans ces conditions, les contraintes sur la stabilité de l'ADPLL "améliorée" sont données par l'inégalité suivante :

$$0 < G_2 < \frac{G_1}{1 - \frac{G_2}{G_1}} < \frac{F_{clk_N} \cdot CST}{F_{nom} \cdot CST_N} \quad (B.32)$$

Annexe C

Les filtres de mesure des paramètres relatifs au PCR

C.1 Introduction

Les filtres de mesure permettent d'extraire les paramètres relatifs au PCR de la norme DVB-T [ETS01]. Ces filtres sont obtenus par synthèse numérique de filtres analogiques par une transformation bilinéaire. Ainsi, le changement de variable suivant est appliqué :

$$s \rightarrow \frac{1 - z^{-1}}{T_{PCR}} \quad (\text{C.1})$$

Ces filtres peuvent avoir quatre configurations suivant la fréquence de coupure choisie. Chaque fréquence de coupure est référencée dans un profil [ETS01]. Le tableau 2.1 présente les différents profils qui vont de *MGF1* à *MGF4*. Les trois premiers profils ont des fréquences de coupure prédéterminées alors que le dernier reste à définir. Ainsi, un tableau résumera les valeurs à affecter aux différents coefficients de chaque filtre étudié. La période d'échantillonnage correspondra à la valeur nominale du temps d'arrivée des PCRs, à savoir $T_{PCR} = 40ms$. Les structures des filtres de mesure seront étudiées dans la suite de cette annexe.

<i>Profils</i>	Fréquence de coupure
<i>MGF1</i>	10 mHz
<i>MGF2</i>	100 mHz
<i>MGF3</i>	1 Hz
<i>MGF4</i>	Défini en manufacture

TAB. C.1 – Profils des filtres de mesure.

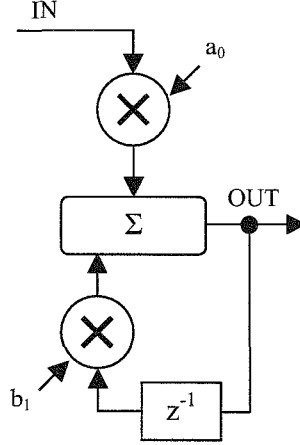


FIG. C.1 – Structure du filtre passe-bas numérique du premier ordre.

C.2 Filtre passe-bas numérique du 1er ordre

La fonction de transfert d'un filtre passe-bas analogique du premier ordre est donnée par :

$$H(s) = \frac{1}{1 + s} \quad (\text{C.2})$$

En appliquant la transformation bilinéaire, la fonction de transfert du filtre passe-bas numérique devient :

$$H(z) = \frac{1}{1 + \frac{1-z^{-1}}{2\pi F_0 \cdot T_{PCR}}} \quad (\text{C.3})$$

Après simplification, la fonction de transfert devient :

$$H(z) = \frac{a_0}{b_0 + b_1 \cdot z^{-1}} \quad (\text{C.4})$$

avec :

$$a_0 = \frac{2\pi F_0 \cdot T_{PCR}}{1 + 2\pi F_0 \cdot T_{PCR}} \quad (\text{C.5})$$

$$b_0 = 1 \quad (\text{C.6})$$

$$b_1 = -\frac{1}{1 + 2\pi F_0 \cdot T_{PCR}} \quad (\text{C.7})$$

Le schéma structurel de ce filtre est donné par la figure C.1 et les valeurs des coefficients du filtre, pour les différents profils de mesure, sont présentées dans le tableau C.2.

C.3 Filtre passe-haut numérique du 1er ordre

Un filtre passe-haut du premier ordre a une fonction de transfert de la forme :

$$H(s) = \frac{1}{1 + \frac{1}{s}} \quad (\text{C.8})$$

Périodes d'échantillonnage	Profils	Fréquences de coupure	Coefficients des filtres	
			a_0	b_1
$T_{SE} = 8 \text{ ms}$	<i>MGF1</i>	10 mHz	$5,024023.10^{-4}$	$-9,994976.10^{-1}$
	<i>MGF2</i>	100 mHz	$5,001408.10^{-3}$	$-9,949986.10^{-1}$
	<i>MGF3</i>	1 Hz	$4,785979.10^{-2}$	$-9,521402.10^{-1}$
$T_{nom} = 40 \text{ ms}$	<i>MGF1</i>	10 mHz	$2,506973.10^{-3}$	$-9,974930.10^{-1}$
	<i>MGF2</i>	100 mHz	$2,451657.10^{-2}$	$-9,754834.10^{-1}$
	<i>MGF3</i>	1 Hz	$2,008486.10^{-1}$	$-7,991514.10^{-1}$

TAB. C.2 – Coefficients du filtre passe-bas du premier ordre pour les différents profils.

La transformation bilinéaire donne la fonction de transfert du filtre passe-haut numérique suivante :

$$H(z) = \frac{1}{1 + \frac{2\pi F_0 T_{PCR}}{1-z^{-1}}} \quad (\text{C.9})$$

Après simplification, la fonction de transfert devient :

$$H(z) = \frac{a_0 + a_1.z^{-1}}{b_0 + b_1.z^{-1}} \quad (\text{C.10})$$

avec :

$$a_0 = \frac{1}{1 + 2\pi F_0 T_{PCR}} \quad (\text{C.11})$$

$$a_1 = -\frac{1}{1 + 2\pi F_0 T_{PCR}} \quad (\text{C.12})$$

$$b_0 = 1 \quad (\text{C.13})$$

$$b_1 = -\frac{1}{1 + 2\pi F_0 T_{PCR}} \quad (\text{C.14})$$

La figure C.2 présente le schéma structurel de ce filtre et le tableau C.3 donne les valeurs des coefficients du filtre.

C.4 Filtre passe-haut numérique du 2° ordre

La fonction de transfert d'un filtre passe-haut analogique du premier ordre est donnée par :

$$H(s) = \frac{s^2}{1 + 2ms + s^2} \quad (\text{C.15})$$

En appliquant la transformation bilinéaire, la fonction de transfert du filtre passe-haut numérique devient :

$$H(z) = \frac{\left(\frac{1-z^{-1}}{2\pi F_0 T_{PCR}}\right)^2}{1 + 2m \cdot \frac{1-z^{-1}}{2\pi F_0 T_{PCR}} + \left(\frac{1-z^{-1}}{2\pi F_0 T_{PCR}}\right)^2} \quad (\text{C.16})$$

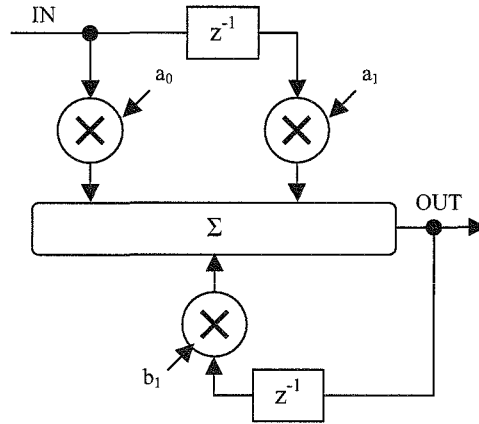


FIG. C.2 – Structure du filtre passe-haut numérique du premier ordre.

Périodes d'échant.	Profils	Fréquences de coupure	Coefficients des filtres		
			a_0	a_1	b_1
$T_{SE} = 8 \text{ ms}$	<i>MGF1</i>	10 mHz	$9,994976.10^{-1}$	$-9,994976.10^{-1}$	$-9,994976.10^{-1}$
	<i>MGF2</i>	100 mHz	$9,949986.10^{-1}$	$-9,949986.10^{-1}$	$-9,949986.10^{-1}$
	<i>MGF3</i>	1 Hz	$9,521402.10^{-1}$	$-9,521402.10^{-1}$	$-9,521402.10^{-1}$
$T_{nom} = 40 \text{ ms}$	<i>MGF1</i>	10 mHz	$9,974930.10^{-1}$	$-9,974930.10^{-1}$	$-9,974930.10^{-1}$
	<i>MGF2</i>	100 mHz	$9,754834.10^{-1}$	$-9,754834.10^{-1}$	$-9,754834.10^{-1}$
	<i>MGF3</i>	1 Hz	$7,991514.10^{-1}$	$-7,991514.10^{-1}$	$-7,991514.10^{-1}$

TAB. C.3 – Coefficients du filtre passe-bas du premier ordre pour les différents profils.

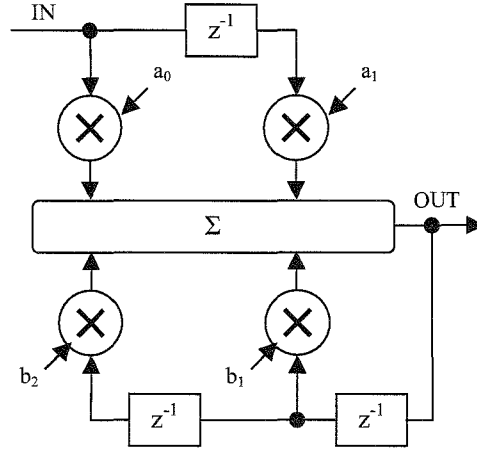


FIG. C.3 – Structure du filtre passe-haut numérique du second ordre.

La fonction de transfert est la suivante après simplification :

$$H(z) = \frac{a_0 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2}}{b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2}} \quad (\text{C.17})$$

avec :

$$a_0 = \frac{1}{1 + 4\pi \cdot m \cdot F_0 \cdot T_{PCR} + (2\pi F_0 \cdot T_{PCR})^2} \quad (\text{C.18})$$

$$a_1 = -\frac{1}{1 + 4\pi \cdot m \cdot F_0 \cdot T_{PCR} + (2\pi F_0 \cdot T_{PCR})^2} \quad (\text{C.19})$$

$$a_2 = \frac{1}{1 + 4\pi \cdot m \cdot F_0 \cdot T_{PCR} + (2\pi F_0 \cdot T_{PCR})^2} \quad (\text{C.20})$$

$$b_0 = 1 \quad (\text{C.21})$$

$$b_1 = -\frac{2 \cdot (1 + 2\pi \cdot m \cdot F_0 \cdot T_{PCR})}{1 + 4\pi \cdot m \cdot F_0 \cdot T_{PCR} + (2\pi F_0 \cdot T_{PCR})^2} \quad (\text{C.22})$$

$$b_2 = \frac{1}{1 + 4\pi \cdot m \cdot F_0 \cdot T_{PCR} + (2\pi F_0 \cdot T_{PCR})^2} \quad (\text{C.23})$$

La figure C.3 montre le schéma structurel de ce filtre, dont les coefficients sont présentés dans le tableau C.4.

Périodes d'échant.	Profils	Coefficients des filtres				
		a_0	a_1	a_2	b_1	b_2
$T_{SE} = 8 \text{ ms}$	<i>MGF1</i>	$9,989954.10^{-1}$	$-1,997991$	$= a_0$	$-1,998995$	$= a_0$
	<i>MGF2</i>	$9,900222.10^{-1}$	$-1,980044$	$= a_0$	$-1,989997$	$= a_0$
	<i>MGF3</i>	$9,065710.10^{-1}$	$-1,813142$	$= a_0$	$-1,904280$	$= a_0$
$T_{nom} = 40 \text{ ms}$	<i>MGF1</i>	$9,949923.10^{-1}$	$-1,989985$	$= a_0$	$-1,994986$	$= a_0$
	<i>MGF2</i>	$9,515679.10^{-1}$	$-1,903136$	$= a_0$	$-1,950967$	$= a_0$
	<i>MGF3</i>	$6,386429.10^{-1}$	$-1,277286$	$= a_0$	$-1,598303$	$= a_0$

TAB. C.4 – Coefficients du filtre passe-bas du premier ordre pour les différents profils.

Annexe D

Description de la plate-forme EXCALIBUR

Afin de valider l'ADPLL et les filtres de mesure, une plate-forme EXCALIBUR est utilisée. Cette plate-forme EXCALIBUR est fournie par Altera, et est particulièrement adaptée au développement de processeurs NIOS, qui peuvent être synthétisés par l'outil de développement Quartus II et embarqués dans le FPGA. Le FPGA disponible sur cette plate-forme est un APEX EP20K200EFC484-2x [ALT04].

Le NIOS permet au concepteur de développer rapidement une application à base de processeurs à l'aide d'outils adéquats. De ce fait, il peut prendre place dans la plupart des systèmes sur puce programmable (SOPC : System On a Programmable Chip). En effet, le NIOS s'adapte à la majorité des applications car il est paramétrable. Il dispose de plusieurs périphériques comme l'UART, le timer, des entrées/sorties parallèles, de la mémoire SRAM et FLASH. De part son architecture fondée sur un processeur RISC 16 ou 32 bits, le NIOS est doté d'une grande puissance de calcul car la plupart des instructions sont exécutées en un cycle d'horloge. Dans le cas du EP20K200EFC484-2x, le NIOS peut être cadencé jusqu'à 50 *MHz*, ce qui signifie qu'il peut effectuer certaine opération en 20 *ns*. De plus, en utilisant le circuit FPGA le plus rapide de la gamme APEX, c'est-à-dire en version APEX 1x, il peut fonctionner jusqu'à 80 *MHz*. Mais le NIOS peut aussi être implanté dans une version plus récente de l'APEX, qui est l'APEX II, ou un FPGA de la gamme supérieure, comme un CYCLONE ou un STRATIX.

La plate-forme EXCALIBUR dispose des éléments suivants :

- un APEX EP20K200EFC484-2x de 8320 cellules logiques, de 106496 bits de mémoire RAM interne et d'un boîtier FBGA de 484 broches,
- une horloge de 33,333 *MHz* fournie par un quartz,
- une mémoire FLASH de 1 *Mo* qui peut soit contenir le fichier de configuration qui sera chargé par le contrôleur de configuration dans le circuit APEX lors de la mise sous tension,

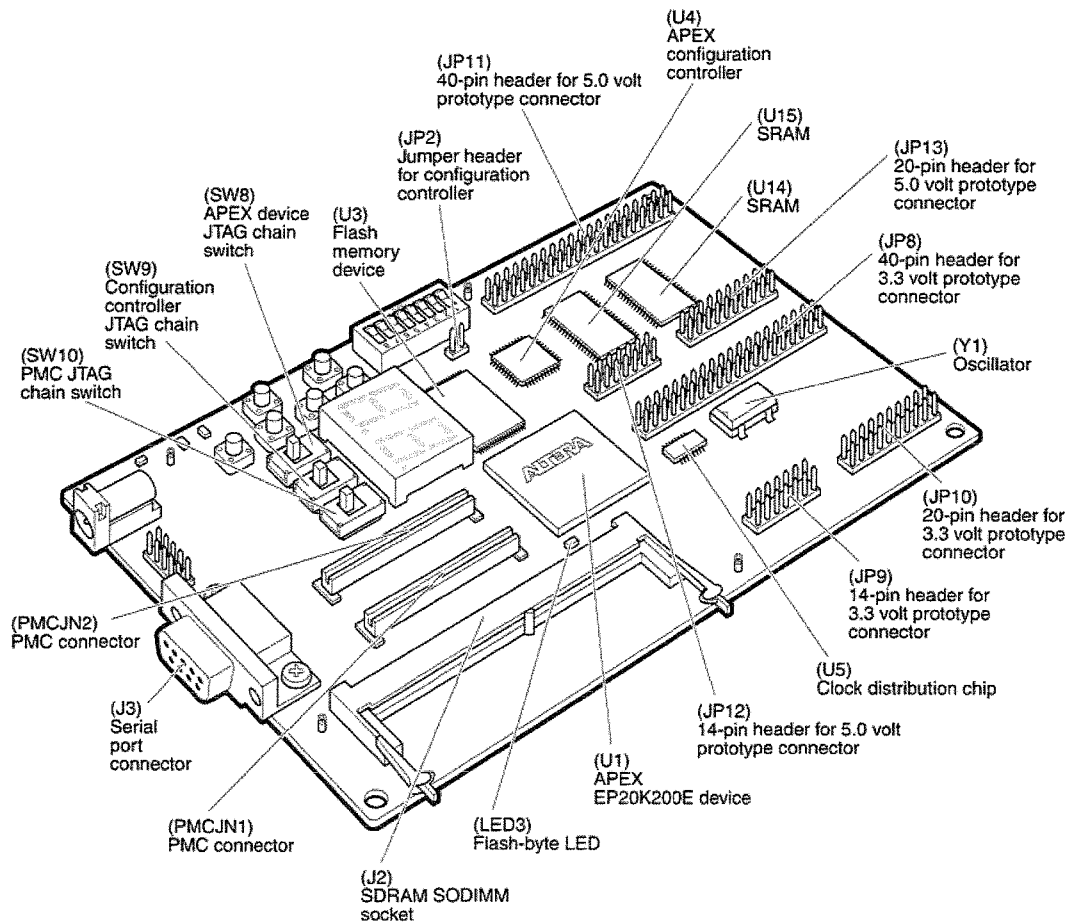


FIG. D.1 – Vue générale de la plate-forme EXCALIBUR utilisant un APEX EP20K200EFC484-2x.

soit être utilisé par le NIOS comme mémoire à usage général et pour le stockage de données non-volatiles,

- une mémoire SRAM de 256 Ko, servant de mémoire programme et de mémoire de données,
- un connecteur SODIMM de 144 broches pour recevoir une mémoire de 64 bits,
- deux connecteurs PCI 64 broches pour l'enfichage de cartes parallèles,
- un connecteur JTAG pour le téléchargement de la configuration de l'APEX,
- un port série RS-232,
- trois connecteurs 3,3 volts et trois connecteurs 5 volts constituant 40 broches d'entrées/sortie,
- un afficheur double 7 segments,
- deux LEDs,
- un block de 8 micro commutateurs,
- un circuit d'alimentation avec reset.

Bibliographie

- [AFF⁺01] N. AGLIADA, A. FIN, F. FUMMI, M. MARTIGNANO et G. PRAVADELLI : On the reuse of vhdl modules into systemc designs. *In Proc. IEEE Forum on Design Languages (FDL'01)*, Lyon, France, septembre 2001.
- [ALT04] ALTERA : Apex 20k programmable logic device family data sheet, 2004.
- [AR01] S. ANAND et B. RAZAVI : A cmos clock recovery circuit for 2.5-gb/s nrz data. *IEEE J. Solid-State Circuits*, 36(3):432–439, mars 2001.
- [BFL⁺01] F. BARONTI, L. FANUCCI, D. LUNARDINI, R. RONCELLA et R. SALETTI : On-line calibration for non-linearity reduction of delay-locked delay-lines. *In Proc. IEEE Int. Conference on Electronics, Circuits and Systems (ICECS'01)*, volume 2, pages 1001–1005, Malta, septembre 2001.
- [BFL⁺02] F. BARONTI, L. FANUCCI, D. LUNARDINI, R. RONCELLA et R. SALETTI : A non-linearity self-calibration technique for delay-locked loop delay-lines. *In Proc. 19th IEEE Instrumentation and Measurement Technology Conference (IMTC'02)*, volume 2, pages 1007–1010, Anchorage, USA, 2002.
- [BFL⁺03] F. BARONTI, L. FANUCCI, D. LUNARDINI, R. RONCELLA et R. SALETTI : A technique for nonlinearity self-calibration of dlls. *IEEE Trans. Instrum. Meas.*, 52(4):1255–1260, août 2003.
- [BFL⁺04] F. BARONTI, L. FANUCCI, D. LUNARDINI, R. RONCELLA et R. SALETTI : A self-calibrating delay-locked loop delay line with shunt-capacitor circuit scheme. *IEEE J. Solid-State Circuits*, 39(2):384–387, février 2004.
- [BZVL01] R. BETANCOURT-ZAMORA, S. VERNA et T. LEE : 1-ghz and 2.8-ghz cmos injection-locked ring oscillator prescalers. *In Proc. IEEE Symposium on VLSI Circuits*, pages 47–50, Kyoto, Japan, 2001.
- [CL03] C.-C. CHUNG et C.-Y. LEE : An all-digital phase-locked loop for high-speed clock generation. *IEEE J. Solid-State Circuits*, 38(2):347–351, février 2003.
- [Cor04] D. CORDEAU : *Etude Comportementale et Conception d'Oscillateurs Intégrés Polypases Accordables en Fréquence en Technologies Si et SiGe pour les Radiocom-*

- munications*. Thèse de doctorat, Ecole Supérieur d'Ingénieurs de Poitiers, France, 2004.
- [dB32] H. de BELLESCIZE : La reception synchrone. *Onde Electrique*, 11:230–240, 1932.
 - [DCP⁺89] G. DONZELLINI, D. CAVIGLIA, G. PARODI, D. PONTA et P. REPETTO : A digital controlled oscillator based on controlled phase shifting. *IEEE Trans. Circuits Syst.*, 36(8):1101–1105, août 1989.
 - [ETS01] Etsi tr101290 - digital video broadcasting (dvb) ; measurement guidelines for dvb systems, 2001.
 - [ETS05] Etsi tr101154 - digital video broadcasting (dvb) ; implementation guidelines for the use of mpeg-2 systems, video in satellite, cable and terrestrial broadcasting applications, 2005.
 - [FF00] D. J. FOLEY et M. P. FLYNN : A 3.3v, 1.6 ghz, low-jitter, self-correcting dll based clock synthesizer in 0.5 μ m cmos. In *Proc. IEEE Int. Symposium on Circuit and Systems (ISCAS'00)*, volume 2, pages 249–252, Geneva, Switzerland, mai 2000.
 - [Gar79] F. M. GARDNER : *Phaselock Techniques*. John Wiley & Sons, Inc., New York, 1979.
 - [GPB03] M. GROZING, B. PHILIPP et M. BERROTH : Cmos ring oscillator with quadrature outputs and 100 mhz to 3.5 ghz tuning range. In *Proc. European Solid-State Circuits Conference (ESSCIRC'03)*, pages 679–682, Estoril, Portugal, septembre 2003.
 - [HLL99] A. HAJIMIRI, S. LIMOTYRAKIS et T. H. LEE : Jitter and phase noise in ring oscillators. *IEEE J. Solid-State Circuits*, 34(6):790–804, 1999.
 - [HP01] P. HEYDARI et M. PEDRAM : A 3.3-v high-speed cmos pll with a two-stage self-feedback ring oscillator. In *Proc. International Conference on Computer Design (ICCD'01)*, pages 209–213, Austin, Texas, septembre 2001.
 - [HSK01] I. HWANG, S. SONG et S. KIM : A digitally controlled phase-locked loop with a digital phase-frequency detector for fast acquisition. *IEEE J. Solid-State Circuits*, 36(10):1574–1581, octobre 2001.
 - [HWL01] T. HSU, C. WANG et C. LEE : Design and analysis of a portable high-speed clock generator. *IEEE Trans. Circuits Syst. II*, 48(4):367–375, avril 2001.
 - [II94] ISO-IEC : Iso/iec 13818-1 - information technology - generic coding of moving pictures and associated audio : Systems, 1994.
 - [JH98] M. JOHNSON et E. HUDSON : A variable delay line pll for cpu-coprocessor synchronization. *IEEE J. Solid-State Circuits*, 23(10):1218–1223, octobre 1998.
 - [JHM01] I. JANISZEWSKI, B. HOPPE et H. MEUTH : Vhdl-based design methodology for reusable high performance direct digital frequency synthesizers. In *Proc. 38th Design Automation Conference (DAC'01)*, pages 573–578, Las Vegas, USA, 2001.

-
- [Kax00] B. KAXE : Synchronisation of mpeg-2 based digital tv services over ip networks. Mémoire de D.E.A., Telia Research AB, Communication Services, Farsta, 2000.
 - [KII02] K. KISHINE, K. ISHII et H. ICHINO : Loop-parameter optimisation of a pll for a low-jitter 2.5-gb/s one-chip optical receiver ic with 1 :8 demux. *IEEE J. Solid-State Circuits*, 37(1):38–50, janvier 2002.
 - [Kod97] M. KODRNJA : *Etude des Oscillateurs Controlés en Tension pour les Circuits à Frequence Intermediaire, Analyse et Simulation du Bruit des Oscillateurs*. Thèse de doctorat, Institut National Polytechnique de Grenoble (INPG), France, 1997.
 - [LK00] J. LEE et B. KIM : A low-noise fast-lock phase-locked loop with adaptive bandwidth control. *IEEE J. Solid-State Circuits*, 35(8):1137–1145, août 2000.
 - [Lém95] F. LÉMERY : *Modélisation Comportementale des Circuits Analogiques et Mixtes*. Thèse de doctorat, Institut National Polytechnique de Grenoble (INPG), France, 1995.
 - [LPKK00] K. LIM, C.-H. PARK, D.-S. KIM et B. KIM : A low-noise phase-locked loop design by loop bandwidth optimisation. *IEEE J. Solid-State Circuits*, 35(6):807–815, 2000.
 - [Man96] J.G. MANEATIS : Low-jitter and process-independent dll and pll based on self-biased techniques. *IEEE J. Solid-State Circuits*, 31(11):1723–1732, novembre 1996.
 - [MCL⁺00] Y. MOON, J. CHOI, K. LEE, D.-K. JEONG et M.-K. KIM : An all-analog multiphase delay-locked loop using a replica delay line for wide-range operation and low-jitter performance. *IEEE J. Solid-State Circuits*, 35(3):377–384, mars 2000.
 - [Meh02] A. MEHROTRA : Noise analysis of phase-locked loops. *IEEE Trans. Circuits Syst. I*, 49(9):1309–1316, septembre 2002.
 - [MJA01] Y. MOON, D.-K. JEONG et G. AHN : A 0.6-2.5-gbaud cmos tracked 3 x oversampling transceiver with dead-zone phase detection for robust clock/data recovery. *IEEE J. Solid-State Circuits*, 36(12):1974–1983, décembre 2001.
 - [MMY⁺00] K. MINAMI, M. MIZUNO, H. YAMAGUCHI, T. NAKANO, Y. MATSUSHIMA et Y. SUMI : A 1 ghz portable digital delay-locked loop with infinite phase capture ranges. In *Proc. IEEE Int. Solid-State Circuits Conference*, pages 350–351, 469, février 2000.
 - [MNS03] M. MAYMANDI-NEJAD et M. SACHDEV : A digitally programmable delay element : Design and analysis. *IEEE Trans. VLSI Syst.*, 11(5):871–878, octobre 2003.
 - [MRT⁺04a] C. MANNINO, H. RABAH, C. TANOUGAST, Y. BERVILLER, M. JANIAUT et S. WEBER : Fpga implementation of a novel all digital pll architecture for pcr related measurements in dvb-t. In *Proc. 14th International Conference on Field Program-*

- mable Logic and Applications (FPL'04)*, pages 1027–1031, Antwerpen, Belgium, août/septembre 2004.
- [MRT⁺04b] C. MANNINO, H. RABAH, C. TANOUGAST, Y. BERVILLER, M. JANIAUT et S. WEBER : Fpga implementation of a novel architecture for pcr related measurements in dvb-t. *In Proc. The 2004 International Conference on Embedded Systems and Applications (ESA'04)*, pages 606–610, Las Vegas, Nevada, USA, 2004.
- [MRT⁺05] C. MANNINO, H. RABAH, C. TANOUGAST, Y. BERVILLER, M. JANIAUT et S. WEBER : Optimised fpga implementation of a multi program pcr measurement system in dvb-t. *In Proc. 2005 International Workshop on Applied Reconfigurable Computing (ARC'05)*, pages 13–21, Algarve, Portugal, février 2005.
- [MRW⁺06] C. MANNINO, H. RABAH, S. WEBER, C. TANOUGAST, Y. BERVILLER et M. JANIAUT : Optimised fpga implementation of a multi program pcr related measurements system in dvb-t : architecture exploration. *International Journal of Electronics*, 93(6):373–383, 2006.
- [MX00] H. MAIR et L. XIU : An architecture of high-performance frequency and phase synthesis. *IEEE J. Solid-State Circuits*, 35(6):835–846, 2000.
- [MY02] M. MANSURI et C.-K. YANG : Jitter optimisation based on phase-locked loop design parameters. *IEEE J. Solid-State Circuits*, 37(11):1375–1382, novembre 2002.
- [Nep02] NEPSYS : *Nepsys Version 1.0, User's Guide*. Prosilog, <http://www.prosilog.com>, 2002.
- [New02] Jitter and jitter testing for modulator drivers. Application Note 12, 2002.
- [NH99] R. NORO et J. P. HUBAUX : Clock synchronization of mpeg-2 services over packet networks. *Telecommunication Systems Journal*, 10(1-2):3–16, mars 1999.
- [NHH99] R. NORO, M. HAMDI et J. P. HUBAUX : Circuit emulation over ip networks. *In Proc. 6th International Workshop on Protocols for High-Speed Networks*, pages 187–201, Salem- MA, USA, août 1999.
- [NM05] N. NOURI et S. MIRABBASI : A 900mhz-2ghz low-swing low-power 0.18 μ m cmos pll. *In Proc. 2005 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'05)*, pages 1500–1503, Saskatoon, Saskatchewan, Canada, mai 2005.
- [Nor00] R. NORO : *Synchronization over packet-switching network : theory and applications*. Thèse de doctorat, Swiss Federal Institute of Technology (EPFL), Lausanne, mai 2000.
- [oEEE00] The Institute of ELECTRICAL et Inc. ELECTRONICS ENGINEERS : Ieee standard vhdl language reference manual, 2000.

-
- [ON04] T. OLSSON et P. NILSSON : A digitally controlled pll for soc applications. *IEEE J. Solid-State Circuits*, 39(5):751–760, mai 2004.
- [ONM⁺00] T. OLSSON, P. NILSSON, T. MEINCKE, A. HEMANI et M. TORKELOSON : A digitally controlled low-power clock multiplier for globally asynchronous locally synchronous designs. In *Proc. IEEE International Symposium on Circuits and Systems (ISCAS'00)*, volume 3, pages 13–16, Geneva, Switzerland, mai 2000.
- [SK01] L. SUN et A. KWASNIEWSKI : A 1.25-ghz 0.35- μ m monolithic cmos pll based on a multiphase ring oscillator. *IEEE J. Solid-State Circuits*, 36(6):910–916, 2001.
- [SLL04] Z. SHU, K. LEE et B. LEUNG : A 2.4-ghz ring-oscillator-based cmos frequency synthesizer with a fractional divider dual-pll architecture. *IEEE J. Solid-State Circuits*, 39(3):452–462, mars 2004.
- [Sri04] R. SRIDHAR : System-on-chip (soc) : Clock and synchronization issues. In *Proc. of the 17th International Conference on VLSI Design (VLSID'04)*, 2004.
- [SSDS02] R. STEFO, J. SCHREITER, J. DOHNDORF et R. SCHNUFFNY : A portable all-digital phase-locked loop for frequency synthesis. In *Proc. 9th International Conference on Design of Integrated Circuits and Systems (MIXDES'02)*, pages 217–222, 2002.
- [SSSS03] R. STEFO, J. SCHREITER, J.-U. SCHLUSSLER et R. SCHNUFFNY : High resolution ad-pll frequency synthesizer for fpga-and asic-based applications. In *Proc. 2003 IEEE International Conference on Field-Programmable Technology (FPT'03)*, pages 28–34, Tokyo, Japan, décembre 2003.
- [Swa01] S. SWAN : An introduction to system level modeling in systemc 2.0. White Paper, mai 2001.
- [Sys02] SYSTEMC : *Version 2, User's Guide*. Prosilog, <http://www.systemc.org>, 2002.
- [TK97] M. THAMSIRIANUNT et T. KWASNIEWSKI : Cmos vco's for pll frequency synthesis in ghz digital mobile radio communications. *IEEE J. Solid-State Circuits*, 32(10):1511–1524, octobre 1997.
- [WAV01] Understanding jitter - getting started, 2001.
- [WCN⁺99] L. WU, H. CHEN, S. NAGAVARAPU, R. GEIGER et W. BLACK : A monolithic 1.25gbits/sec cmos clock/data recovery circuit for fiber channel transceiver. In *Proc. IEEE International Symposium on Circuit and Systems (ISCAS'99)*, volume 2, pages 565–568, Orlando, Florida, USA, mai 1999.
- [XLMP04] L. XIU, W. LI, J. MENIERS et R. PADAKANTI : A novel all-digital pll with software adaptive filter. *IEEE J. Solid-State Circuits*, 39(3):476–483, mars 2004.
- [XY02] L. XIU et Z. YOU : A flying-adder architecture of frequency and phase synthesis with scalability. *IEEE Trans. VLSI Syst.*, 10(5):637–649, octobre 2002.

- [XY03] L. XIU et Z. YOU : A new frequency synthesis method based on flying-adder architecture. *IEEE Trans. Circuits Syst. II*, 50(3):130–134, mars 2003.

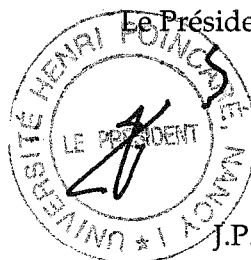
Monsieur MANNINO Christian

DOCTORAT DE L'UNIVERSITE HENRI POINCARÉ, NANCY 1
en INSTRUMENTATION & MICRO-ELECTRONIQUE

VU, APPROUVÉ ET PERMIS D'IMPRIMER n° 1215

Nancy, le 10/07/06

Le Président de l'Université



J.P. FINANCE



Résumé : La nature asynchrone du flux transport MPEG-2 en télévision numérique nécessite la transmission d'informations de l'horloge de l'encodeur. Ces informations se présentent sous forme de données numériques appelées « program clock reference » (PCR) et insérées dans les paquets de transport. Le décodeur au niveau du récepteur utilise ces informations en association avec une boucle à verrouillage de phase pour reconstruire l'horloge et se synchroniser ainsi avec l'encodeur. Lors de leur transmission, les PCRs sont soumis à des perturbations telles que par exemple les variations de délai dues au réseau. Ces perturbations induisent notamment de la gigue et différentes dérives qui peuvent avoir un effet significatif sur la qualité de la réception. La mesure de ces paramètres fait également souvent appel à une PLL. Cette fonction est généralement constituée de composants analogiques qui présentent à la fois une reproductibilité délicate et une contrainte pour la technologie d'implantation.

Le travail de thèse présenté porte sur l'étude et la caractérisation d'une fonction PLL numérique associée à un ensemble de fonctions de mesure. Sa particularité est de permettre une implantation sur des technologies numériques y compris FPGA. Une étude portant sur l'état de l'art des techniques de synchronisation, et sur les techniques de génération d'horloge les plus utilisées, met en évidence les limites des systèmes de synchronisation actuels. La solution proposée utilise une boucle à verrouillage de phase totalement numérique (ADPLL). Cela est rendu possible grâce au remplacement de l'oscillateur contrôlé en tension par un synthétiseur de fréquence original basé sur la génération d'un signal de fréquence centrale, subissant un retard ou une avance de phase calibrés. L'architecture de synchronisation proposée est modélisée pour déterminer la gigue intrinsèque au synthétiseur et le comportement de l'ADPLL en terme de stabilité et de bande passante. La solution est utilisée dans un système de mesure des paramètres de gigue, erreur de fréquence centrale, et dérive pour l'évaluation de la qualité de service en télévision numérique terrestre (DVB-T). La solution, bien que travaillant à une fréquence centrale de 27 MHz, est totalement compatible avec les technologies FPGA. Les performances principales obtenues sont une gigue intrinsèque faible (100 ps), l'indépendance de la PLL du temps d'arrivée des PCRs, la portabilité de l'architecture et la possibilité d'intégration de plusieurs PLL dans un FPGA.

Mots clés : ADPLL, FPGA, mesures relatives au PCR, synthétiseur de fréquence, exploration de l'espace de conception, qualité de service, MPEG-2, DVB.