Power Consumption Modeling in Embedded Systems Hardware

Majdi Richa

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

THESE DE DOCTORAT DE .

L'INSTITUT NATIONAL DES SCIENCES APPLIQUEES RENNES

ECOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique , Signal, Systèmes, Electronique
Spécialité : *Electronique*

Par
Majdi Y. RICHA

Power Consumption Modeling in Embedded Systems Hardware

Thèse présentée et soutenue à INSA Rennes, le 22 septembre 2023
Unité de recherche : UMR CNRS 6164
Thèse N° : 23ISAR 23 / D23 - 23

Rapporteurs avant soutenance :

Cécile Belleudy	MCF HDR, Université Côte d'azur, LEAT
Eric Senn	MCF HDR, Université de Bretagne Sud, Lab-STICC

Composition du Jury :

Président :	Bertrand Granado	Professeur, Sorbonne Paris, LIP6-SYEL
Examinateurs :	Smail Niar	Professeur, Univ. Polytechnique Hauts-de-France, LAMIH
	Bertrand Granado	Professeur, Sorbonne Paris, LIP6-SYEL
Dir. de thèse :	Jean-Christophe Prévotet	Professeur, INSA, Rennes, IETR
Co-dir. de thèse :	Abed Ellatif Samhat	Professeur, Faculté de Génie, Univ. Libanaise
Enc. de thèse :	Mickaël Dardaillon	MCF, INSA, Rennes, IETR
Co-enc. de thèse :	Mohamad Mroué	Professeur Associé, Faculté de Génie, Univ. Libanaise

Declaration of Authorship

I, Majdi RICHA, declare that this thesis titled, "Power Consumption Modeling in Embedded Systems Hardware" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: October 9, 2023

"I prefer to be a dreamer among the humblest, with visions to be realized, than lord among those without dreams and desires."

Khalil Gibran

Abstract

Majdi RICHA

Power Consumption Modeling in Embedded Systems Hardware

Power optimization has become a major concern for most digital hardware designers, particularly in early design phases and especially in limited power budget systems (battery-operated hand-held devices, electro-optical pluggable modules, IoT and green energy systems, etc.). Subsequently, early power consumption estimation at design time is crucial for power optimization. This research covers multiple topics serving the digital circuits power optimization notably FPGAs. Initially, a short overview on power consumption factors, energy optimization techniques and low-level power estimation approaches is briefly covered. An overview of High-Level power estimation techniques currently available along with a comprehensive comparison between different methodologies and their applications on estimated models is thoroughly presented. Then, we elaborate on the proposed learning-based power modeling and estimation methodology of FPGA IPs targeting both offline and online domains. This covers also the automated data generation and acquisition system along with a detailed and automatic training data sets construction approach. For the offline mode, we estimate the power consumption based on the state machine control signals and the input activity of the data path. For the online counterpart, we estimate in situ and in real-time the power consumption based on its most significant modes of operation and its input activity. The proposed application for the online alternative involves a fault detection algorithm that relies on real-time power consumption monitoring and power profiling. A moving fault score window is used to represent the possibility of fault occurrences. Methodology validation and experimental results show an absolute percentage error of $< 0.5\%$, $< 1\%$ and $< 2\%$ for the data path, the state machine and online power monitoring respectively.

Acknowledgements

I would like to express my deepest gratitude to my directors Pr. Jean-Christophe Prévotet and Pr. Abed Ellatif Samhat for all the guidance and expertise they provided me throughout my doctoral studies.

In addition, I would like to express my profound appreciation to my supervisors Dr. Mickaël Dardaillon and Dr. Mohamad Mroué for their vast support and continuous follow up.

Finally, I would like to acknowledge with affection the understanding and love of my family. This achievement would not have been possible without their endless support.

Contents

List of Figures

List of Tables

List of Abbreviations

ACDGAS	**A**utomated and **C**entralized **D**ata **G**eneration and **A**cquisition **S**ystem
ADC	**A**nalog to **D**igital **C**onverter
AI	**A**rtificial **I**ntelligence
ANN	**A**rtificial **N**eural **N**etwork
API	**A**pplication **P**rogramming **I**nterface
ASIC	**A**pplication **S**pecific **I**ntegrated **C**ircuit
BRAM	**B**lock **R**andom **A**ccess **M**emory
CISA	**C**ustomized **I**nstruction **S**et **A**rchitecture
CM	**C**ontrol signal or operation **M**odes
CMOS	**C**omplementary **M**etal **O**xide **S**emiconductor
CS	**C**ontrol **S**ignal
Cy	**C**ycles
DAC	**D**igital to **A**nalog **C**onverter
DAQ	**D**ata **A**cquisition
DGA	**D**ata **G**eneration and **A**cquisition
DP	**D**ata **P**ath
DPM	**D**ynamic **P**ower **M**anagement
DSP	**D**igital **S**ignal **P**rocessing
DVFS	**D**ynamic **V**oltage and **F**requency **S**caling
EDA	**E**lectronic **D**esign **A**utomation
FE	**F**ast **E**thernet
FF	**F**lip-**F**lop
FIFO	**F**irst **I**n **F**irst **O**ut
FPGA	**F**ield **P**rogrammable **G**ate **A**rray
FRC	**F**ree **R**unning **C**ounter
FSM	**F**inite **S**tate **M**achine
HDL	**H**ardware **D**escriptive **L**anguage
HL	**H**igh **L**evel
HLS	**H**igh **L**evel **S**ynthesis
HSDI	**H**igh **S**peed **D**igital **I**nput
HSDIO	**H**igh **S**peed **D**igital **I**nput **O**utput
HSDO	**H**igh **S**peed **D**igital **O**utput

HW	**H**ardware
I2C	**I**nter **I**ntegrated **C**ircuit
IP	**I**ntellectual **P**roperty
LA	**L**ogic **A**nalyzer
LUT	**L**ook-**U**p **T**able
MAPE	**M**ean **A**bsolute **P**ercentage **E**rror
ML	**M**achine **L**earning
MLP	**M**ulti-**L**ayer **P**erceptron
MPE	**M**ean **P**ercentage **E**rror
MPIF	**M**icroprocessor **I**nterface
MSE	**M**ean **S**quared **E**rror
MSM	**M**ost **S**ignificant **M**ode
PG	**P**attern **G**enerator
PLB	**P**rogrammable **L**ogic **B**lock
PLL	**P**hase **L**ocked **L**oop
PM	**P**ower **M**odel
PMBus	**P**ower **M**anagement **B**us
RAM	**R**andom **A**ccess **M**emory
RTL	**R**egister **T**ransfer **L**ayer
SMBus	**S**erial **M**anagement **B**us
SO	**S**ampling **O**scilloscope
SoC	**S**ystem on **C**hip
SOTA	**S**tate **O**f **T**he **A**rt
SPI	**S**erial **P**eripheral **I**nterface
SRAM	**S**tatic **R**andom **A**ccess **M**emory
SW	**S**oftware
TB	**T**estbench
TLM	**T**ransaction-**L**evel **M**odeling
UML	**U**nified **M**odeling **L**anguage
USB	**U**niversal **S**erial **B**us
VCD	**V**alue **C**hange **D**ump
VCO	**V**oltage **C**ontrolled **O**scillator
XPA	**X**ilinx **P**ower **A**nalyzer

To my Father…
Although he is no longer
physically present in my life, I
still feel his impact every day.

Résumé de la thèse

Résumé — L'optimisation de la puissance est devenue une préoccupation majeure pour la plupart des concepteurs de systèmes numériques, en particulier dans les premières phases de conception et surtout dans les systèmes à énergie limitée (appareils portables fonctionnant sur batterie, modules enfichables électro-optiques, systèmes IoT et énergies vertes, etc.). Par conséquent, l'estimation précoce de la consommation d'énergie au moment de la conception est devenue cruciale pour l'optimisation de la puissance. Cette recherche couvre plusieurs sujets liés à l'optimisation de la puissance des circuits numériques, notamment les circuits reconfigurables FPGA. Dans un premier temps, un bref aperçu des facteurs de consommation d'énergie, des techniques d'optimisation énergétique et des approches d'estimation de puissance de bas niveau est présenté. Une vue d'ensemble des techniques d'estimation de puissance de haut niveau actuellement disponibles, ainsi qu'une comparaison détaillée entre différentes méthodologies et leurs applications, sont ensuite présentées en détails. Ensuite, nous développons la méthodologie proposée de modélisation et d'estimation de puissance basée sur l'apprentissage des FPGA IP. Ces travaux ciblent à la fois les domaines hors-ligne et en-ligne. Ces derniers incluent également le système automatisé de génération et d'acquisition de données ainsi qu'une approche détaillée et automatique de construction des ensembles de données d'entraînement. Pour le mode hors-ligne, nous estimons la consommation d'énergie en fonction des signaux de contrôle de la machine à états et de l'activité d'entrée du chemin de données. Pour le mode en-ligne, nous estimons en temps réel la consommation d'énergie en fonction de ses modes de fonctionnement les plus significatifs et de son activité d'entrée. L'application proposée pour l'alternative en ligne implique un algorithme de détection de panne qui repose sur la surveillance en temps-réel de la consommation d'énergie et le profilage de la puissance. Une fenêtre de score est utilisée pour représenter la possibilité d'occurrence de pannes. La validation de la méthodologie et les résultats expérimentaux montrent une erreur absolue en pourcentage inférieure à $0,5\%$, 1% et 2% respectivement pour le chemin de données, la machine à états et la surveillance de puissance en ligne.

I.1 Introduction

Les systèmes embarqués sont de plus en plus présents dans une large gamme d'applications, des produits électroniques grand public aux systèmes de contrôle industriel. À mesure que ces systèmes deviennent plus complexes et riches en fonctionnalités, leur consommation d'énergie devient une préoccupation de plus en plus cruciale. Prédire et gérer avec précision la consommation d'énergie dans les systèmes embarqués est essentiel pour garantir le bon fonctionnement dans les limites de leur budget énergétique, ainsi que pour optimiser l'utilisation de l'énergie.

Cette thèse explore le problème de la modélisation de la consommation d'énergie dans le matériel des systèmes embarqués et les propriétés intellectuelles FPGA (IP). L'objectif est de développer des méthodes précises et efficaces pour prédire la consommation d'énergie et les appliquer à des systèmes réels. La recherche se concentre sur le niveau matériel, plus précisément sur le domaine de haut niveau et la consommation d'énergie de différents circuits FPGA.

L'originalité de cette recherche réside dans l'importance croissante de la gestion de l'énergie dans les systèmes embarqués. Les méthodes et techniques développées dans cette thèse seront utiles aux concepteurs de systèmes numériques et aux développeurs de systèmes embarqués, garantissant un fonctionnement fiable et efficace.

La travail de thèse comprend un aperçu des sources de consommation d'énergie, des dépendances et des techniques d'optimisation. Un état de l'art sur les techniques récentes de modélisation et d'estimation de la consommation d'énergie de haut niveau est mené afin de situer les travaux par rapport aux études existantes. La nécessité d'une approche anticipée et fiable de la modélisation de la consommation d'énergie est identifiée, ce qui conduit à la proposition d'une plateforme de caractérisation pour collecter des données réelles issues du matériel. Un système automatisé et centralisé de génération et d'acquisition de données est développé pour caractériser les IPs FPGA et collecter des données pour l'entraînement de l'apprentissage automatique.

La thèse se concentre sur la modélisation et l'estimation de la consommation d'énergie dans les domaines hors-ligne et en-ligne. Dans le domaine hors-ligne, la consommation d'énergie est estimée avec précision et rapidement pendant la phase de conception d'une IP FPGA. Ce travail s'étend au domaine en-ligne, où les modèles de puissance générés sont mis en œuvre à l'intérieur du FPGA et la puissance IP simultanée est surveillée et rapportée à un système de gestion de puissance avec une latence minimale.

L'organisation de la thèse est la suivante : Le chapitre 2 fournit des informations de base sur les sources de consommation d'énergie, les dépendances et les techniques d'optimisation. Le chapitre 3 présente un aperçu complet des techniques de modélisation et d'estimation de la consommation d'énergie de haut niveau. Le chapitre 4 développe la méthodologie de modélisation et d'estimation de la consommation d'énergie de haut niveau pour les IPs FPGA dans les applications hors-ligne et en-ligne. Le chapitre 5 traite de la plateforme de caractérisation automatisée pour la génération et l'acquisition de données. Le chapitre 6 aborde la génération de modèles de puissance et la construction automatisée d'un jeu de données d'entraînement. Le chapitre 7 présente les résultats expérimentaux pour les applications hors ligne et en ligne, avec une extension à la détection de défauts IP basée sur l'estimation de puissance en ligne et le profilage de puissance. Enfin, le chapitre 8 conclut la thèse.

L'annexe A répertorie les articles et revues publiés, tandis que l'annexe B inclut des photos du matériel réel, y compris la plateforme de caractérisation, le dispositif FPGA testé (DUT) et le logiciel d'interface utilisateur.

I.2 Contexte

Les systèmes embarqués sont des systèmes informatiques spécialisés conçus pour des tâches spécifiques au sein de systèmes ou d'appareils plus vastes. Ils sont composés de composants logiciels et matériels, la partie matérielle incluant souvent une logique programmable telle que les circuits intégrés programmables (FPGA). Les FPGA sont des circuits intégrés qui peuvent être programmés pour exécuter des fonctions logiques numériques spécifiques.

La modélisation de la consommation d'énergie des systèmes embarqués consiste à estimer la consommation d'énergie d'un système avant sa construction et son déploiement. Cette estimation permet de prédire la consommation d'énergie dans différents scénarios et d'identifier les problèmes potentiels liés à l'énergie. Les techniques de modélisation de la consommation d'énergie comprennent la modélisation analytique, la simulation et la mesure. La modélisation analytique utilise des équations mathématiques pour estimer la consommation d'énergie en fonction de l'architecture du système et des caractéristiques des composants. La simulation consiste à modéliser le comportement du système et sa consommation d'énergie à l'aide d'outils logiciels. La mesure implique de mesurer physiquement la consommation d'énergie du système.

La consommation d'énergie dans les circuits CMOS numériques a deux

sources principales : l'énergie dynamique et l'énergie statique. La consommation d'énergie dynamique provient des transitions de signal et comprend la consommation d'énergie du circuit d'horloge, de la logique et de l'interconnexion. La consommation d'énergie statique est due au courants de fuite et est indépendante des calculs effectués.

La consommation d'énergie dans les systèmes embarqués dépend de divers facteurs tels que la partition logique, le routage, le placement et l'utilisation des ressources, la dissipation thermique, la technologie utilisée et les algorithmes logiciels. Les techniques d'optimisation de la consommation d'énergie visent à réduire la consommation d'énergie tout en maintenant ou en améliorant la fonctionnalité et les performances. Ces techniques peuvent être appliquées au niveau du système, du dispositif, du circuit et de l'architecture. Des exemples de techniques d'optimisation de la consommation d'énergie comprennent l'utilisation de blocs intégrés à granularité grossière, le pipelining, l'optimisation de la longueur de mot, le clock-gating, l'utilisation de marges thermiques et la variation dynamique de la tension.

Les techniques d'estimation et de modélisation de la consommation d'énergie aident à prédire la consommation d'énergie dans les systèmes embarqués. Les techniques de bas niveau prennent en compte les détails d'implémentation et de circuit. Des exemples de techniques d'estimation de puissance de bas niveau comprennent les méthodes probabilistes, basées sur la simulation et statistiques. Les méthodes probabilistes utilisent les caractéristiques des données, tandis que les méthodes basées sur la simulation impliquent l'application de signaux de stimuli et la réalisation de simulations. Les méthodes statistiques utilisent des séquences de bits aléatoires et des simulations avec des estimateurs de puissance.

En résumé, la modélisation et l'estimation de la consommation d'énergie sont importantes pour optimiser la consommation d'énergie des systèmes embarqués et identifier les problèmes potentiels liés à l'énergie. Diverses techniques sont utilisées pour estimer et modéliser la consommation d'énergie, et l'optimisation de la consommation d'énergie peut être réalisée grâce à différentes techniques de conception au niveau du système, du dispositif, du circuit et de l'architecture.

I.3 Techniques d'estimation de haut niveau dans le matériel des systèmes embarqués - État de l'art

Dans cette section, nous introduisons le sujet des techniques d'estimation et de modélisation de la consommation d'énergie de haut niveau pour les conceptions FPGA et ASIC. À mesure que la complexité de la conception augmente et que les outils de conception de haut niveau deviennent plus répandus, il y a un intérêt croissant à adopter des méthodologies de conception de haut niveau sur le marché des FPGA. De même, les ASIC jouent un rôle crucial dans les systèmes embarqués, en particulier ceux dotés de processeurs intégrés et de logique interne étroitement couplée.

Traditionnellement, les langages de description matérielle (HDL) ont été utilisés pour les conceptions FPGA/ASIC, mais ils sont souvent insuffisants lorsqu'il s'agit de traiter la complexité des systèmes matériels numériques avancés. Pour remédier à cela, des outils commerciaux de synthèse de haut niveau (HLS) sont apparus, permettant aux concepteurs de mettre en œuvre des conceptions FPGA/ASIC, ou des parties de celles-ci, à l'aide de langages de haut niveau. Cependant, ces avancées ont apporté de nouveaux défis, en particulier dans le domaine de l'estimation de la consommation d'énergie.

La modélisation précise de la consommation d'énergie est essentielle dans deux scénarios. Tout d'abord, elle est nécessaire pendant la phase de conception pour l'exploration de l'espace de conception axée sur la consommation d'énergie. Les concepteurs doivent itérer entre l'estimation de la consommation d'énergie et les étapes de raffinement de la conception pour parvenir à des conceptions économes en énergie. Disposer de modèles de consommation d'énergie précis et efficaces peut accélérer l'exploration de la conception et faciliter les simulations de haut niveau. Deuxièmement, la modélisation de la consommation d'énergie est également pertinente en temps d'exécution pour la surveillance et la gestion de l'alimentation. Des techniques efficaces de gestion de l'alimentation, telles que l'adaptation dynamique de la tension et de la fréquence (DVFS) et la planification des tâches dans les systèmes CPU-FPGA, reposent sur des modèles de consommation d'énergie mis en œuvre dans les dispositifs eux-mêmes.

Les concepteurs de systèmes embarqués qui considèrent des méthodologies de conception de haut niveau bénéficient de niveaux d'abstraction plus élevés, de cycles de conception plus courts et de la vérification anticpée de la fonctionnalité. Les méthodologies d'estimation et de modélisation de la consommation d'énergie de haut niveau se sont étendues au-delà des FPGA et ASIC, englobant les systèmes à base de microcontrôleurs et de microprocesseurs avec

des architectures diverses (par exemple, les cœurs logiciels, les cœurs matériels, les jeux d'instructions personnalisés, les systèmes sur une seule puce).

Nous fournissons un aperçu complet des techniques de modélisation et d'estimation de la consommation d'énergie de haut niveau. Nous présentons une classification et une comparaison complètes des différentes méthodologies en fonction de leurs modèles cibles. Les sections suivantes approfondissent les techniques spécifiques, la classification et les discussions, suivies d'une conclusion.

De plus, nous présentons un aperçu des différentes techniques d'estimation de la consommation d'énergie et de leurs modèles estimés correspondants. Les quatre principales techniques d'estimation identifiées sont les méthodes basées sur la simulation, les approches basées sur l'apprentissage, les méthodes basées sur les statistiques et les alternatives basées sur les mesures. Les modèles cibles pour l'estimation de la consommation d'énergie peuvent être catégorisés en trois groupes : les composants/circuits, les IP en boîte blanche/boîte noire et les microprocesseurs/CISA/(MP)SoC.

- Les méthodes basées sur la simulation consistent à simuler le système avec différentes entrées et à capturer les sorties souhaitées pour estimer la consommation d'énergie. Elles nécessitent des informations telles que les valeurs de courant et de tension, la capacité, la fréquence de fonctionnement et l'activité de commutation. Les exemples de modèles cibles pour les méthodes basées sur la simulation comprennent les composants/circuits, les IP en boîte blanche/boîte noire et les microprocesseurs/CISA/(MP)SoC.

- Les approches basées sur l'apprentissage utilisent des algorithmes d'apprentissage automatique pour prédire la consommation d'énergie en se basant sur des données d'entraînement. Elles apprennent les motifs et les corrélations entre les paramètres d'entrée et la consommation d'énergie pour effectuer des estimations précises.

- Les méthodes basées sur les statistiques utilisent des modèles statistiques pour estimer la consommation d'énergie. Elles analysent le comportement du système et formulent des hypothèses basées sur des distributions statistiques et des probabilités.

- Les alternatives basées sur les mesures consistent à mesurer directement la consommation d'énergie du système à l'aide de matériels ou d'outils spécialisés. Elles fournissent des données précises et en temps réel sur la consommation d'énergie, mais peuvent nécessiter des ressources supplémentaires.

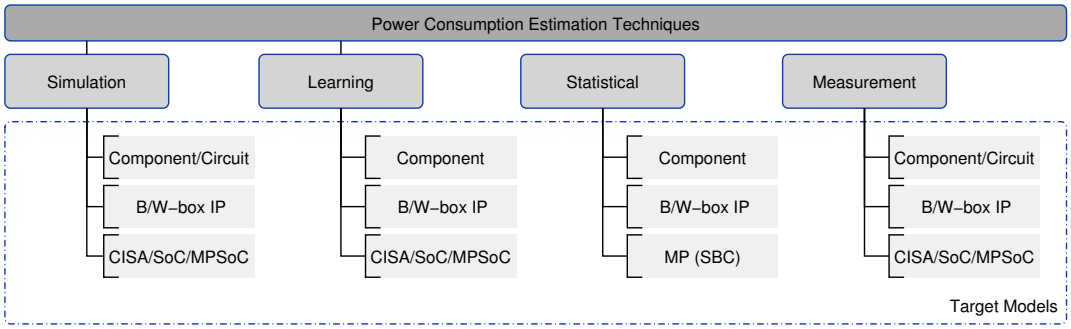

FIGURE 1: Techniques d'estimation et modèles cibles

Les modèles cibles pour l'estimation de la consommation d'énergie peuvent être catégorisés en trois groupes :

- Composants/Circuits : Cette catégorie comprend des composants individuels tels que des opérateurs et des éléments logiques de base. Elle englobe également la modélisation de la capacité et des fils d'interconnexion dans les circuits.
- IP en boîte blanche/boîte noire : Il s'agit d'unités de cellules logiques réutilisables qui peuvent faire partie à la fois des FPGA et des ASIC. Les IP en boîte blanche ont des entrées et des sorties prédéfinies, tandis que les IP en boîte noire ont des entrées et des sorties inconnues.
- Microprocesseurs/CISA/(MP)SoC : Cette catégorie comprend les microprocesseurs, les architectures de jeu d'instructions personnalisées (CISA) et les systèmes multiprocesseurs sur puce (MPSoC). L'estimation de la consommation d'énergie pour ces cibles complexes dépend de leur architecture et du logiciel d'application.

La Figure 1 donne un aperçu des techniques d'estimation de la consommation d'énergie et de leurs modèles cibles correspondants. Nous discutons de la classification des techniques d'estimation de la consommation d'énergie et des modèles cibles en fonction de diverses mesures. Les mesures incluent la dépendance, la caractérisation, l'effort d'estimation, l'erreur d'estimation, l'effort de modélisation et le niveau de modélisation. Les techniques et les modèles sont classés et triés en fonction du nombre d'occurrences dans la littérature.

Les discussions fournissent une comparaison complète des différentes techniques d'estimation appliquées à des modèles cibles courants. On observe que les méthodes basées sur la simulation nécessitent généralement un effort

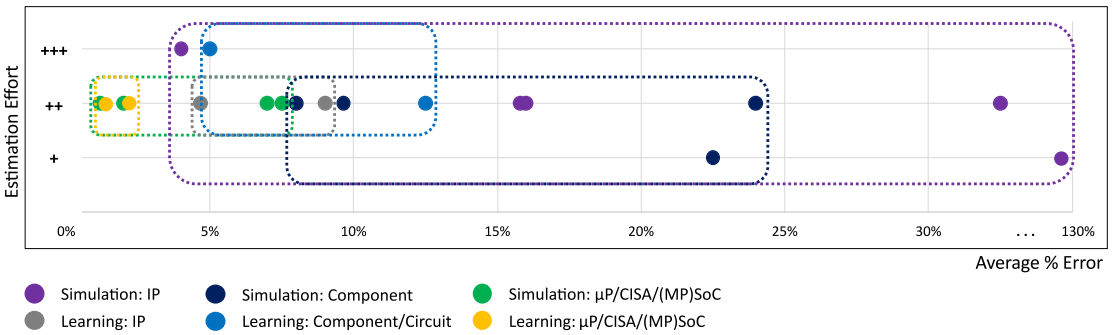

FIGURE 2: Effort d'estimation par rapport à l'erreur moyenne en pourcentage - Graphique des clusters

modéré à considérable, mais peuvent fournir des résultats d'estimation satisfaisants, notamment pour les applications logicielles ciblant les microprocesseurs. Les méthodes basées sur l'apprentissage montrent de meilleures performances en termes d'estimation de la consommation d'énergie avec un effort modéré. Pour les modèles cibles de composants/circuits, les méthodes basées sur l'apprentissage présentent des erreurs d'estimation plus petites et plus étroites par rapport aux méthodes basées sur la simulation.

Une représentation basée sur des clusters, comme illustré dans la Figure 2, met en évidence l'efficacité des différentes techniques d'estimation appliquées à différents modèles cibles. Les techniques basées sur la simulation conviennent aux modèles cibles de microprocesseurs et de systèmes sur puce, tandis que les techniques basées sur l'apprentissage offrent une plus grande efficacité pour les modèles d'IP en boîte blanche/boîte noire et de microprocesseur/système sur puce. Les techniques basées sur les statistiques et les techniques basées sur les mesures sont également discutées, mettant en évidence leurs avantages et leurs inconvénients.

En conclusion, nous avons étudié une enquête sur les techniques d'estimation de la consommation d'énergie et les modèles cibles. Cela souligne l'importance de choisir des techniques d'estimation appropriées en fonction des performances, de la complexité de la conception et des modèles cibles. La sélection d'une technique d'estimation appropriée peut accélérer le processus de conception, améliorer la précision de la modélisation et permettre une exploration plus efficace de l'espace de conception axée sur la consommation d'énergie. La recherche future dans ce domaine peut se concentrer sur l'amélioration des techniques d'estimation existantes, l'exploration de nouvelles approches et la prise en compte de la variabilité et de la sensibilité des résultats d'estimation.

I.4 Modélisation et méthodologie d'estimation de la consommation d'énergie des FPGA

Dans cette section, nous abordons la modélisation et la méthodologie d'estimation de la consommation d'énergie des FPGA. La modélisation et l'estimation de la consommation d'énergie des FPGA consistent à prédire la consommation d'énergie d'une conception FPGA avant sa mise en œuvre matérielle. Ce processus aide à optimiser la conception afin de minimiser la consommation d'énergie. Nous explorons des approches d'estimation de puissance à la fois hors ligne et en ligne.

Dans le domaine hors ligne, l'estimation de puissance est réalisée pendant la phase de conception. Nous proposons une méthodologie d'estimation de puissance basée sur l'apprentissage automatique et la mesure. Le circuit FPGA IP (propriété intellectuelle) est décomposé en une machine à états finis (FSM) et un chemin de données (DP). Les signaux de contrôle et l'activité des entrées sont extraits d'un banc d'essai lors de la simulation. Ces valeurs sont utilisées pour entraîner un modèle d'apprentissage automatique qui prédit la consommation d'énergie du circuit IP. Le modèle est stocké dans une base de données et peut être utilisé pour l'estimation de puissance de nouveaux scénarios.

Dans le domaine en ligne, l'estimation de puissance est réalisée en temps réel pendant le fonctionnement du FPGA. Nous proposons une méthodologie de surveillance de puissance en ligne basée sur l'apprentissage automatique et les réseaux de neurones artificiels supervisés (ANN). La consommation d'énergie du FPGA IP est estimée in situ et en temps réel. Cette estimation est utilisée pour des techniques d'optimisation de puissance en temps réel telles que l'échelonnage dynamique de la tension et de la fréquence (DVFS). En surveillant la consommation d'énergie, des points de fonctionnement optimaux en termes de consommation d'énergie peuvent être déterminés dynamiquement. La Figure 3 présente le flux global de modélisation et d'estimation de puissance pour les domaines hors ligne et en ligne.

Pour générer des données d'entraînement pour les modèles d'apprentissage automatique, des signaux de stimulation sont générés à l'aide d'algorithmes dédiés. Ces signaux de stimulation, ainsi que les signaux de contrôle ou les modes de fonctionnement, sont appliqués au FPGA IP. La consommation d'énergie est mesurée à l'aide d'un système d'acquisition de données matérielles. Les données collectées sont utilisées pour construire des ensembles de données d'entraînement pour les modèles d'apprentissage automatique.

Dans l'ensemble, notre approche vise à fournir une estimation précise et efficace de la puissance pour les FPGA IP. Nous nous appuyons sur des données

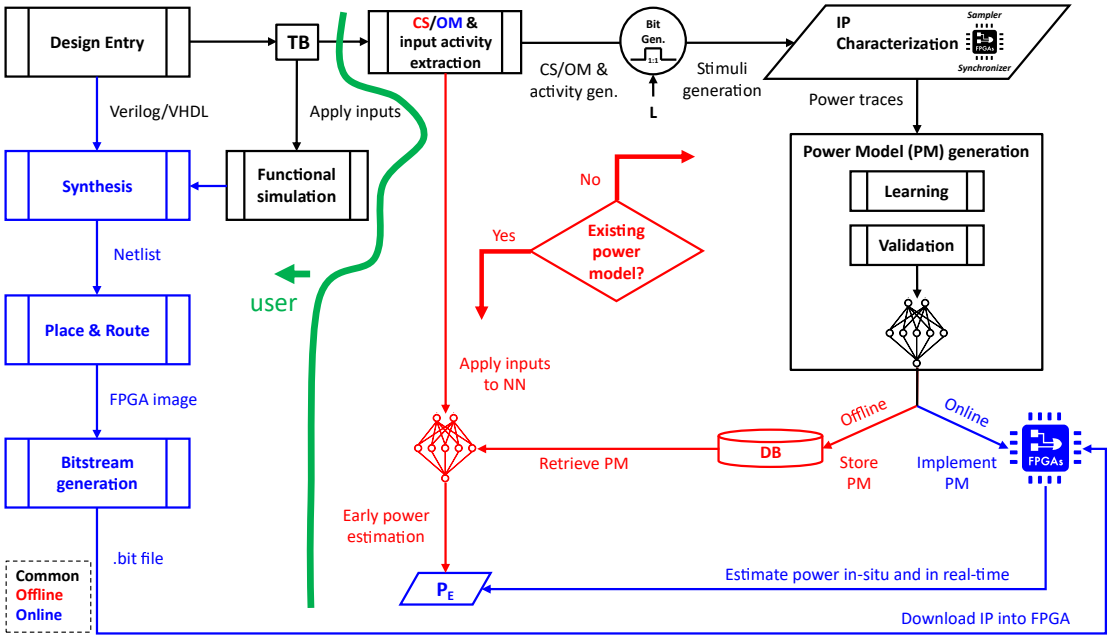

FIGURE 3: Flux global de modélisation et d'estimation de puissance couvrant les domaines hors ligne et en ligne

réelles obtenues à partir de mesures matérielles et d'algorithmes logiciels pour construire des ensembles de données d'entraînement pour l'apprentissage automatique. Cela conduit à des modèles de puissance plus réalistes et donc à une estimation de puissance plus précise. L'estimation de puissance peut être utilisée à la fois dans l'exploration de l'espace de conception hors ligne et dans la surveillance et la gestion de la puissance en ligne.

I.5 Caractérisation d'IP FPGA

Le processus de caractérisation d'IP FPGA consiste à mesurer et analyser les caractéristiques de performance d'un bloc IP destiné à être implémenté dans un FPGA spécifique. La consommation d'énergie de l'IP, en particulier pendant des modes de fonctionnement spécifiques, est influencée par l'activité de commutation des entrées, telle que le taux de commutation (SR) et le pourcentage du temps à l'état haut (PLH) des signaux d'entrée. Le processus de caractérisation nécessite une plateforme de mesure fiable, et un système d'acquisition et de génération de données automatisé et centralisé basé sur un FPGA (ACDGAS) est proposé à cette fin.

L'ACDGAS combine les fonctionnalités d'un oscilloscope d'échantillonnage avec des entrées analogiques, d'un analyseur logique et d'un générateur de motifs binaires. Il offre un mécanisme de génération et d'acquisition de données

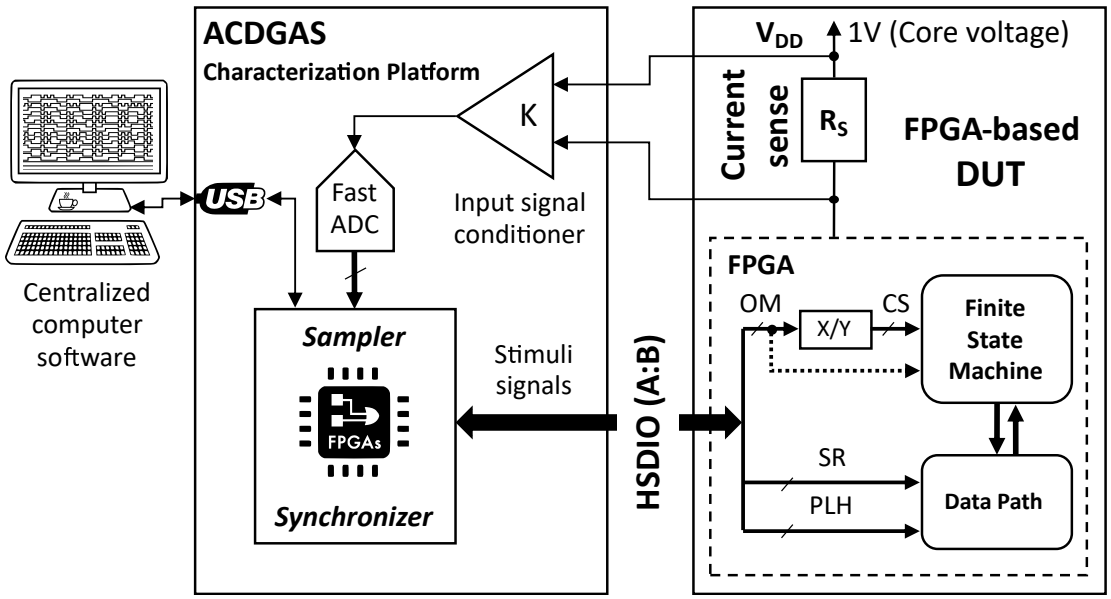

FIGURE 4: Configuration matérielle du système de mesure et de caractérisation avec l'IP en test

synchronisé et aligné, ce qui est essentiel pour des mesures précises de consommation d'énergie. Le système est composé d'une plateforme matérielle à carte unique interfacée à un ordinateur externe via USB ou Fast Ethernet. Le matériel comprend un FPGA, un microprocesseur, des E/S numériques haute vitesse et un convertisseur analogique-numérique (CAN) différentiel parallèle haute vitesse pour les mesures de consommation d'énergie.

L'architecture logicielle du système offre un processus entièrement automatisé pour la caractérisation d'IP. Elle comprend des modules pour une interface utilisateur graphique, la construction de stimuli, le couplage des signaux de contrôle et des modes de fonctionnement, la génération de séquences binaires, l'interface avec la plateforme de mesure, la construction de données d'entraînement, la modélisation de la puissance et l'évaluation. Le logiciel facilite l'interaction entre ces modules et assure une synchronisation et un alignement précis des données générées et acquises.

Le système ACDGAS (Figure 4) surmonte les limitations des alternatives disponibles en fournissant une plateforme hybride et synchrone de génération et d'acquisition de données de manière compacte et rentable. Il élimine le besoin de plusieurs instruments et logiciels, simplifie l'intégration d'un logiciel de contrôle centralisé et garantit une collecte de données précise et synchronisée.

Le matériel du système est composé de composants à la fois numériques et analogiques. La partie numérique comprend le FPGA, l'organisation de la mémoire et le mécanisme de contrôle et d'état. Le FPGA synchronise la génération de sorties numériques haute vitesse (HSDO) avec l'acquisition d'entrées

numériques haute vitesse (HSDI) et d'entrées analogiques. Le microprocesseur externe sert de passerelle entre le FPGA et l'ordinateur hôte, permettant l'accès aux registres du système pour la configuration et les informations d'état.

La partie analogique du matériel comprend deux canaux analogiques indépendants pour l'acquisition de données. Ces canaux sont équipés de convertisseurs analogique-numérique (CAN) à large bande passante pour échantillonner précisément les signaux d'entrée analogiques. Les CAN ont une fréquence d'échantillonnage de 20MSa/s et offrent d'excellentes performances dynamiques.

Le logiciel comprend le micrologiciel fonctionnant sur le matériel et le logiciel d'interface PC pour le contrôle et le traitement des données. Le micrologiciel peut être implémenté à l'aide du microprocesseur externe ou du processeur intégré du FPGA, en fonction de la configuration du système. Le logiciel PC offre une interface utilisateur graphique pour un contrôle facile de l'instrument et prend en charge le mode batch pour les procédures automatisées.

La méthodologie de génération et d'acquisition se concentre sur l'alignement des signaux numériques sortants (HSDO) avec les signaux numériques entrants (HSDI) et les signaux analogiques entrants. Un alignement précis est obtenu en assurant un échantillonnage synchronisé de tous les signaux. Le logiciel contrôle la génération des signaux HSDO et capture simultanément les échantillons HSDI et analogiques. Les données alignées sont ensuite disponibles pour une analyse ultérieure et une modélisation de la consommation d'énergie.

Le système ACDGAS offre une solution polyvalente et efficace pour la caractérisation d'IP FPGA. Il fournit une plateforme de génération et d'acquisition de données entièrement automatisée et synchronisée, permettant des mesures précises de consommation d'énergie et une analyse approfondie. Les composants matériels et logiciels du système fonctionnent ensemble pour garantir une collecte de données précise et alignée, en en faisant un outil précieux pour la caractérisation d'IP et la modélisation de la puissance.

I.6 Génération de modèle de consommation d'énergie

Dans la section suivante, nous nous concentrons sur la génération de modèles de consommation d'énergie pour les circuits numériques à l'aide de techniques d'apprentissage automatique. Le processus comprend plusieurs étapes, notamment la mesure de puissance, l'estimation des paramètres du modèle et la validation du modèle. Une mesure précise de la puissance est cruciale pour la construction de modèles de puissance fiables.

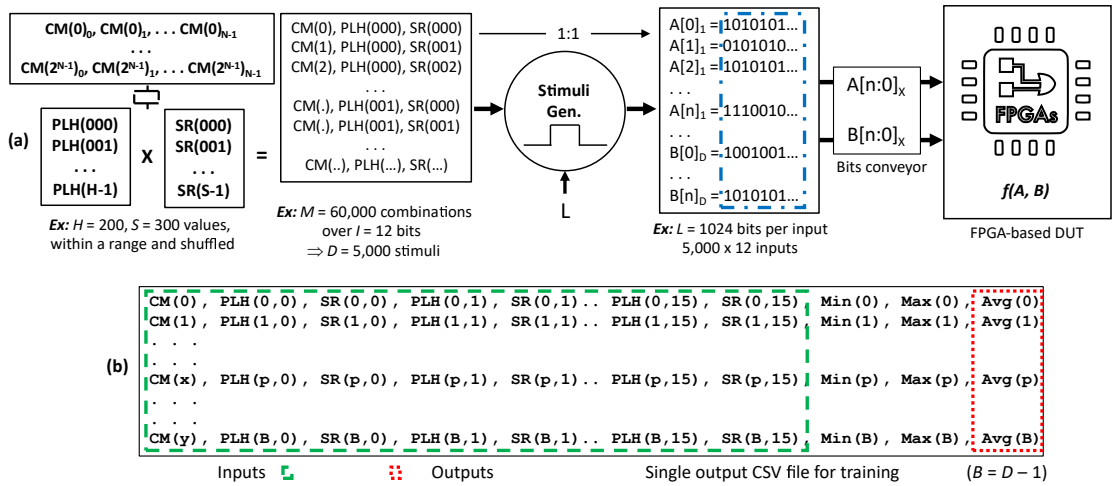

FIGURE 5: a) Génération et appariement des valeurs PLH, SR et stimuli couplés avec CM - b) Format des données d'entraînement

Le modèle proposé d'estimation de la consommation d'énergie dans ce travail est basé sur des réseaux neuronaux supervisés. L'apprentissage automatique et l'intelligence artificielle jouent un rôle clé dans la formation des modèles à l'aide d'algorithmes et de données appropriées. La construction des ensembles de données d'entraînement est essentielle pour la précision du modèle dérivé. Dans ce chapitre, une méthodologie automatisée de construction d'ensembles de données d'entraînement est présentée, où les données sont collectées à partir de sources matérielles et logicielles. La conception et le flux de données, y compris l'interaction entre le logiciel et le matériel, sont décrits.

La construction automatisée d'ensembles de données d'entraînement implique la création d'ensembles de données de haute qualité à l'aide d'outils logiciels et d'algorithmes. Cela garantit que les données utilisées pour former les modèles d'apprentissage automatique sont précises et pertinentes, ce qui permet d'améliorer la précision et l'efficacité. L'évolutivité du processus de construction automatisée est également mise en évidence.

Comme le montre la Figure 5, les ensembles de données d'entraînement de ce travail sont construits à partir d'algorithmes de génération de stimuli et d'un système d'acquisition de données matériel. Les algorithmes de génération de stimuli produisent des paires de valeurs PLH (pourcentage de "1") et SR (pourcentage de transitions), couplées à des signaux de contrôle ou des modes de fonctionnement (CM). Ces combinaisons de stimuli sont utilisées pour générer des matrices de signaux de stimuli. Le système d'acquisition de données matérielles collecte les valeurs de consommation d'énergie correspondant aux signaux de stimuli appliqués à une IP donnée. Les données collectées sont ensuite utilisées pour construire les ensembles de données d'entraînement pour le modèle de puissance.

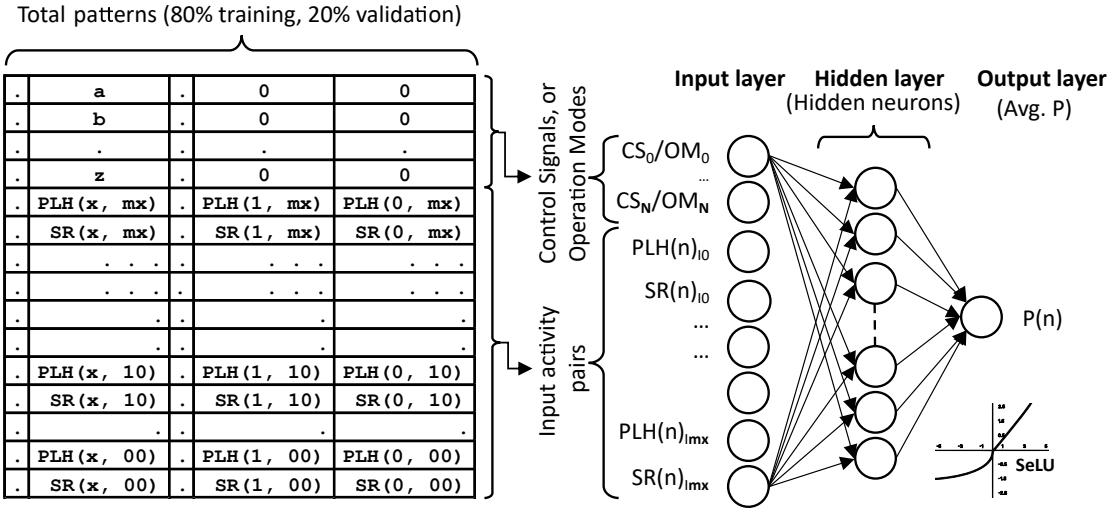

FIGURE 6: Architecture du réseau neuronal montrant les données d'entraînement, les couches, les neurones et la fonction d'activation (SeLU)

Le modèle de puissance proposé (Figure 6) est basé sur une architecture de réseau neuronal implémentée à l'aide de la bibliothèque Python TensorFlow/Keras. L'architecture comprend une couche d'entrée, une couche cachée et une couche de sortie. Le nombre de neurones cachés est optimisé pour trouver un équilibre entre la vitesse de prédiction et la précision. Le réseau est entraîné à l'aide des ensembles de données d'entraînement, une partie des données étant réservée à la validation et à l'évaluation. La métrique de perte d'entraînement est l'erreur quadratique moyenne (MSE), et les métriques d'évaluation sont l'erreur absolue moyenne (MAE) et l'erreur absolue moyenne en pourcentage (MAPE).

Le flux de travail automatisé complet est illustré, y compris les étapes de génération de stimuli, d'exécution du système d'acquisition, de collecte de données, de normalisation des données, d'entraînement du réseau neuronal et de construction du modèle de puissance. Un outil logiciel est développé pour orchestrer ces étapes et fournir une interface entre le système de mesure et le processus de génération du modèle de puissance.

L'implémentation matérielle du modèle de puissance est également discutée, mettant en évidence l'importance de cette implémentation dans un FPGA pour la gestion en temps réel de la puissance. Les FPGA permettent un traitement et une prise de décision en temps réel, ce qui les rend adaptés aux applications nécessitant une gestion de puissance en direct.

Nous concluons en présentant l'architecture logicielle en couches proposée. L'architecture comprend des modules pour l'interface graphique, la construction de stimuli, le couplage des signaux de contrôle, la génération de séquences

de bits, l'interface de la plate-forme de mesure, la construction des données d'entraînement, la modélisation de puissance et l'évaluation.

I.7 Validation, Résultats et Applications

Dans cette section, nous présentons le processus de validation et les résultats obtenus à partir du projet de recherche. Elle est divisée en deux sections principales : les résultats expérimentaux et l'évaluation du modèle.

Dans la section des résultats expérimentaux, nous abordons les cas de tests hors-ligne et en-ligne. Les cas de tests hors ligne se concentrent sur l'estimation de puissance des IPs FPGA avant leur implémentation physique. Le scénario de chemin de données uniquement est considéré, où différents composants d'IP FPGA tels que des multiplieurs, des unités de multiplication et d'accumulation (MAC) et des aligneurs de données d'unité de transport optique (OTU-DA) sont sélectionnés comme cas d'études. La consommation d'énergie de ces IPs est mesurée à l'aide d'un FPGA Xilinx Artix-7 disponible dans le commerce, et les résultats sont collectés pour l'analyse.

Dans les cas de tests en ligne, l'estimation de puissance en temps réel d'une IP FPGA boîte noire est effectuée. L'IP boîte noire est un circuit numérique dont les détails internes sont masqués, et l'estimation de puissance est basée sur l'activité de commutation des entrées du chemin de données de l'IP. Deux modèles d'estimation de puissance sont générés à l'aide de réseaux neuronaux avec un nombre différent de neurones cachés, et leurs performances sont évaluées. En tant qu'extension de la surveillance de puissance en ligne, un algorithme de détection de défaut est présenté, basé sur l'estimation de puissance en temps réel et le profilage de puissance.

Dans la section d'évaluation du modèle, les résultats de la modélisation de puissance sont évalués. Des courbes d'apprentissage sont présentées pour montrer la relation entre les performances du modèle et la quantité de données d'entraînement. Les courbes démontrent la diminution de la perte de prédiction et de l'erreur moyenne absolue en pourcentage à mesure que le nombre d'époques d'entraînement augmente, ce qui indique l'efficacité de l'architecture proposée.

Les ressources et les résultats de performance sont également analysés, notamment l'utilisation des ressources de l'IP FPGA (LUT, FF, DSP), la consommation de puissance (statique, dynamique, puissance totale maximale), le nombre de neurones cachés dans le modèle de puissance, et les métriques d'évaluation telles que l'erreur absolue moyenne (MAE) et l'erreur absolue moyenne en pourcentage (MAPE). Les résultats montrent les caractéristiques

de consommation d'énergie des différents composants de l'IP et l'exactitude du modèle d'estimation de puissance.

Dans l'ensemble, ce chapitre fournit un aperçu complet du processus de validation et des résultats obtenus à partir du projet de recherche, démontrant l'efficacité et la fiabilité de la méthodologie d'estimation de puissance proposée et de ses applications.

I.8 Conclusion

Ce travail présente une approche novatrice pour estimer la consommation d'énergie dans les blocs de propriété intellectuelle (IP) à base de matrices programmables sur site (FPGA) en utilisant des techniques d'apprentissage automatique. Un modèle basé sur les réseaux neuronaux est développé, ce qui permet de prédire avec précision la consommation d'énergie. La méthodologie proposée est appliquée à des composants IP spécifiques, et les résultats expérimentaux démontrent une grande précision avec une erreur absolue en pourcentage inférieure à $0,5\%$, 1%, et 2% respectivement pour le chemin de données, la machine à états et la surveillance en ligne de la consommation d'énergie. Cette recherche améliore significativement l'efficacité et l'efficacité de la gestion de l'énergie dans les IP FPGA. De plus, une méthodologie de détection de défauts est présentée, basée sur l'extension de l'estimation en ligne de la consommation d'énergie en utilisant la technique de profilage de puissance.

Sur la base du travail présenté, il existe plusieurs pistes potentielles pour des travaux futurs et une expansion :

- Application à des conceptions complexes au niveau du système : Les recherches actuelles se concentrent sur l'estimation de la consommation d'énergie au niveau de l'IP. Les travaux futurs peuvent explorer l'extension de la méthodologie à des conceptions complexes au niveau du système incorporant plusieurs IP et interconnexions. Cela nécessiterait le développement de techniques pour modéliser la consommation d'énergie au niveau du système et tenir compte des interactions entre différents composants afin d'obtenir des estimations de puissance plus précises.

- Exploration des modèles et objectifs alternatifs de consommation d'énergie: La recherche actuelle porte sur la modélisation et l'estimation de la consommation d'énergie des IP FPGA, cependant, une bonne alternative (ou expansion) serait les ASICs. Les circuits intégrés spécifiques sont

largement utilisés et consomment généralement moins d'énergie que les circuits logiques basés sur la reconfiguration.

- Exploration de techniques alternatives d'apprentissage automatique : Alors que les recherches actuelles utilisent des réseaux neuronaux pour l'estimation de la consommation d'énergie, les travaux futurs peuvent explorer d'autres techniques d'apprentissage automatique telles que l'apprentissage profond ou les méthodes d'ensemble. Des études comparatives peuvent être menées pour évaluer l'efficacité et la pertinence de ces approches alternatives dans le contexte de l'estimation de la consommation d'énergie pour les IP FPGA.

- Extension de l'algorithme de détection de défauts : Les travaux actuels abordent la possibilité de détecter des défauts généraux dans les IP FPGA causés par plusieurs facteurs. Cependant, se concentrer sur l'aspect de la sécurité, considéré comme un sujet d'actualité récemment, serait d'un grand intérêt.

En conclusion, cette recherche fournit une base solide pour de futurs progrès dans l'estimation et la gestion de la consommation d'énergie dans les IP FPGA. A l'aide de ce travail, les chercheurs peuvent affiner et étendre la méthodologie, ce qui conduit à des conceptions plus efficaces et optimisées des systèmes numériques.

Chapter 1

Introduction

Embedded systems are becoming increasingly prevalent in a wide range of applications, from consumer electronics to industrial control systems. As these systems become more complex and feature-rich, their power consumption becomes an increasingly critical concern. Accurately predicting and managing power consumption in embedded systems is essential for ensuring that the systems will operate reliably within the constraints of their power budgets, and for optimizing the use of energy.

This thesis investigates the problem of power consumption modeling in embedded systems hardware and notably FPGA Intellectual Properties (FPGA IPs). The main objective of this research is to develop accurate and efficient methods for predicting power consumption in embedded systems, and to apply these methods to real-world systems. The research focuses on the hardware level, specifically on the high level domain and subsequently on the power consumption of different FPGA IP circuits.

The significance of this research lies in the increasing importance of power management in embedded systems, and the need for accurate and efficient methods for predicting and managing power consumption. The methods and techniques developed in this thesis will be useful for digital hardware designers and developers of embedded systems, and will help to ensure that these systems are able to operate reliably and efficiently.

1.1 Thesis Scope

In this research, we first of all, conduct an overview covering the power consumption sources, dependencies and main optimization techniques; followed by a survey on the recent high-level power modeling and estimation of embedded systems hardware. The main purpose was to highlight the state of the art and to position the scope of our work versus the related work. Subsequently, we discovered the need of a reliable approach to model and estimate power consumption as early as possible during design time. The methodology that

we propose, in comparison to most related work, relies on real data derived from hardware via a measurement system that we denote as the characterization platform. Here also, we discovered the need of such a platform capable of generating stimuli signals, and, simultaneously, collecting aligned digital and analog samples in a synchronous manner. Consequently, we propose an automated and centralized data generation and acquisition system that will be used to characterize FPGA IPs and to collect data to be used for machine learning training purposes later on. In this thesis, we focus on the power consumption modeling and estimation for both offline and online domains. Initially, for the offline field, we estimate, accurately and fast, the power consumption of an FPGA IP in parallel with its simulation procedure during design time. This work expands to cover the online field as well. For that purpose we apply and implement generated power models inside the FPGA; the power of the concurrent IP is monitored and reported to a power manager system with minimum to no latency compared to recent related work. As a direct extension to online power estimation, we propose a fault detection algorithm capable of detecting anomalies in specific FPGA IP inputs. This is done by providing a moving fault score window.

As a quick summary, the contributions of the presented work can be briefed in chronological order as follows:

1. An overview, classification and comparison between various high-level power consumption modeling and estimation techniques and their target applications.

2. The conception, design and implementation of a hardware characterization platform for FPGA IPs, capable of synchronously generating stimuli signals and acquiring aligned digital and analog data.

3. An offline methodology for high-level power estimation of FPGA IP based on supervised machine learning relying on state machine control signals and data path input activity characteristics.

4. An automatic and efficient training data construction process based on measurements for high-level learning-based FPGA IP power modeling.

5. A high-level online and in-situ power monitoring for FPGA IP based on machine learning by providing most significant operating modes and input activity specifications.

6. A learning-based extension of the online power monitoring capable of detecting faults in FPGA IPs via power profiling technique.

1.2 Thesis Organization

This work is organized as follows: in chapter 2, as a background, we present an overview of power consumption sources and dependencies, we also list the main optimization techniques. Next, we briefly define the power consumption modeling and estimation methods at both low- and high-level. Chapter 3 provides an extensive overview on high-level power estimation and modeling techniques, being our main focus. Embedded systems including FPGAs, System On-a-Chip (SoC), Multi-Processor System On-a-Chip (MP-SoC), Application Specific Integrated Circuit (ASIC) and microprocessors architectures are investigated as various power consumption estimation and modeling platforms. We classify the estimation techniques according to the adopted methodology such as simulation-, learning-, measurement- and statistical-based. We also show the target models in each category. We then provide a quantitative and a qualitative analysis based on a number of metrics including dependency, estimation effort, estimation error, modeling level, etc. In chapter 4 we elaborate on the high-level power modeling and estimation methodology of FPGA IPs for both the offline and online applications. The learning-based methodology is thoroughly described. In chapter 5 we present the automated and centralized data generation and acquisition system considered as the characterization platform and serving our methodology. We also present experimental results proving the efficiency and fidelity of the system when targeting FPGA measurements. Chapter 6 details the power model generation along with the automated training data sets construction for the learning-based estimation. This chapter also briefly covers the power estimator hardware implementation for online applications. In chapter 7 we present the experimental results covering test cases for both offline and online applications including fault detection application. We also assess the models by providing learning curves, estimation and performance results following specific metrics. Finally we conclude in chapter 8.

In appendix A we list all of the papers and journals that have been published or currently under review. In appendix B, we show circuit schematics and photos of the real hardware covering the characterization platform, the FPGA Device Under Test (DUT), and also the user interface software.

Chapter 2

Background

2.1 Introduction

Embedded systems are specialized computer systems designed to perform specific tasks within larger systems or devices. It typically consists of interacting software and hardware sections. The software part includes a microcontroller or a microprocessor, memory, and various input and output peripherals. The hardware section often includes digital circuits and programmable logic. Usually both software and hardware are integrated in a single chip. In this work, we focus on the hardware part of the embedded systems, mainly the Filed Programmable Gate Array (FPGA). An FPGA is a type of integrated circuit that can be programmed to perform specific digital logic functions. It consists of an array of configurable logic blocks, interconnected by programmable interconnects as shown in Figure 2.1. These logic blocks can be configured to implement any digital logic function, including arithmetic operations, memory storage, and complex control logic.

Power consumption modeling for embedded systems refers to the process of estimating the power consumption of an embedded system, which is a computer system with a dedicated function within a larger electrical system, before it is built and deployed. The goal of power consumption modeling is to predict how much power the system will consume under different scenarios, such as different workloads or operating conditions, and to identify potential power issues that may arise. This information can be used to optimize the system's power consumption, either by reducing its overall power consumption or by making sure that the system can operate within a specific power budget.

Power consumption modeling can be performed using a variety of techniques, such as analytical modeling, simulation, or measurement [69]. Analytical modeling involves using mathematical equations to estimate the power consumption of the system based on its architecture and the characteristics of its components. Simulation involves using software tools to model the system's

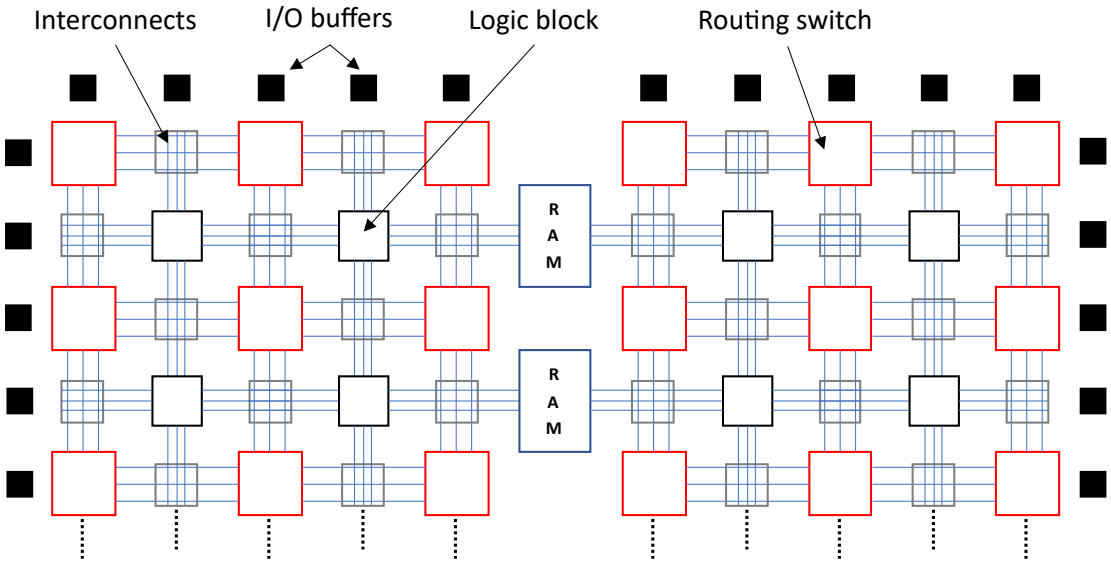

FIGURE 2.1: FPGA internal blocks and interconnects.

behavior and power consumption under different test cases. Measurement involves physically measuring the power consumption of the system using test equipment.

Power consumption modeling is important for embedded systems since it can help to ensure the system's reliable operation within the constraints of its power budget, and to identify potential power issues early in the design process. It also helps to optimize the power consumption and prolong the power source's life of the embedded systems, thus saving energy and cost.

This chapter is organized as follows: Section 2.2 is a context study covering a brief overview of power consumption sources and dependencies. It also summarizes multiple power optimization techniques. Section 2.3 defines power estimation and modeling techniques at both low- and high-levels with a short focus on low-level alternative. Section 2.4 is a wrap up for the background presentation.

2.2 Context study

In order to explore power consumption modeling and estimation world, it is inevitable to investigate power sources and their dependencies. Also, the power optimization techniques with various aspects (system-, device-, circuit- and architecture-level) have to be examined. Power optimization in digital circuits is the process of reducing the power consumption of a circuit while maintaining or improving its functionality and performance. This is a critical design goal in many digital systems, as reducing power consumption can lead to longer battery life, lower operating costs, and reduced heat dissipation.

As mentioned earlier, our main focus is directed towards embedded systems and FPGAs in particular. However, the following subsections provide a broader coverage in order to accumulate a better understanding of the subject in context.

2.2.1 Overview of Power Consumption Sources

Power consumption of digital Complementary Metal Oxide Semiconductor (CMOS) circuits is dependent on two main sources. The first source is the dynamic power, which arises when signals transition their values, and the second source is static power, which causes circuits to dissipate power even when no switching activity is happening [64].

Dynamic Power

The dynamic power, Figure 2.2 a), is divided into three components: 1) power of clock circuitry (with dedicated routing resources), 2) logic power consumed in the functional units and memories and 3) power of interconnects between units (where the load capacitance depends on wires types and lengths) [28]. The dynamic power can be obtained by the following formula (in Watt):

$$P = \alpha.C_l.V_{dd}^2.f \tag{2.1}$$

where α (referred to as switching activity) is the average number of $0 \rightarrow 1$ or $1 \rightarrow 0$ transitions in one clock cycle, C_l is the load capacitance, V_{dd} is the power supply voltage and f is the clock frequency.

Static Power

The static power, Figure 2.2 b), is derived from leakage current. For a given hardware component, it depends on the state of the power supply (on or off, and voltage) while being independent of the computation being performed. The leakage current tends to increase when the size of transistors is reduced. Since heating silicon increases its conductivity, the static power also increases with temperature [54]. Static power can be obtained using the following formula (in Watt):

$$P_{static} = V_{dd}.I_{leakage} \tag{2.2}$$

where V_{dd} is the power supply voltage and $I_{leakage}$ is the leakage current.

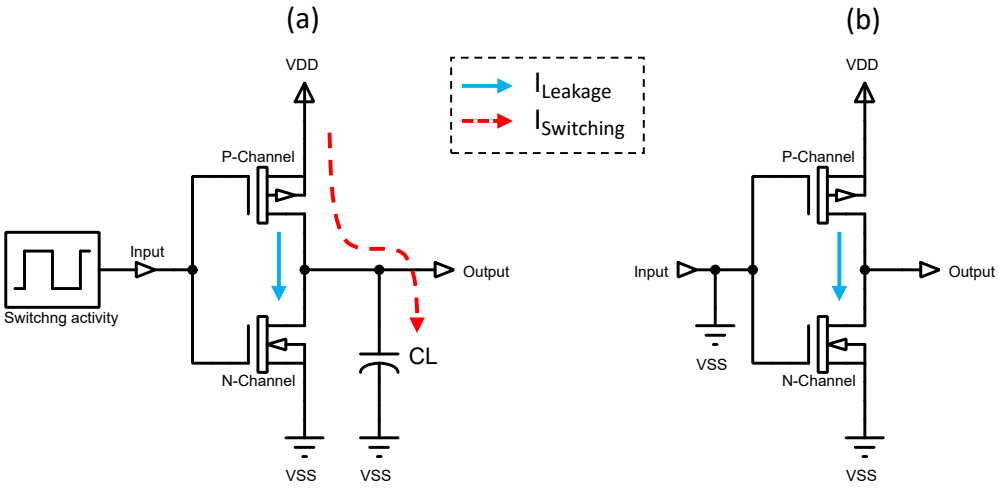

FIGURE 2.2: Dynamic v/s static power.

Power Consumption Dependencies

The power consumption of any digital system heavily depends on the following major factors [37]:

- Logic partition: this can be done by 1) functionality and block size, 2) clock domain and timing criticality and 3) registers (inputs and outputs).
- Route and placement, done using tools provided by the FPGA vendor or another software manufacturer.
- Mapping and resource utilization: block RAM (BRAM), flip-flops (FF), digital signal processing (DSP) and their percentage utilization and mapping.
- Thermal dissipation [33]; temperature leads to increased leakage current [25].
- Technology used, frequency of operation and transmitted power [22].
- Software algorithms and implementation techniques, heavily affecting microprocessors (both soft and hard cores) performance and thus power consumption [7].

2.2.2 Overview of Power Optimization Techniques

Power optimization in embedded systems and power consumption modeling and estimation are two interrelated concepts that are crucial for designing power-efficient embedded systems. The relationship between power optimization and power consumption modeling is that power consumption modeling

is a critical tool for identifying areas where power optimization can be applied. By understanding the power consumption behavior of the system, designers can make informed decisions about the most effective power optimization techniques to implement. In turn, power optimization can inform the power consumption modeling process, by providing new insights into the power consumption behavior of the system after optimization has been applied.

Power consumption optimization in digital circuits involves a variety of techniques to reduce the power consumption of a digital circuit while maintaining or improving its performance. As mentioned earlier, power consumption represents a major concern for most digital hardware designers; therefore the need for power optimization and related techniques is a must.

A brief description of low-power techniques as listed in [36] and [14] will be presented in this section. In particular, we will cover both system-level and device-level design techniques.

System-level design techniques

Here we list and describe various low-power design techniques that have been applied to current FPGA and ASIC technologies:

- Coarse-grained embedded blocks are usually preferable over fine-grained since they are more power-efficient. FPGAs contain a large number of Programmable Logic Blocks (PLBs) that can be interconnected to form complex digital circuits. They can be divided into two main categories: coarse-grained and fine-grained. The difference between these two categories lies in their logic capacity and the complexity of the circuits that they can implement. Coarse-grained PLBs are suitable for large and complex circuits, while fine-grained PLBs are suitable for small and simple circuits.

- Pipelining, a simple and effective way of reducing glitches and hence minimizing power consumption.

- Word-length optimization, mainly applied to obtain the best trade-off in speed, area, power consumption, flexibility and accuracy.

- Clock-gating, usually used to reduce dynamic power consumption by disabling the clock for the inactive regions in order to prevent logic transitions [66].

- Leveraging of thermal margin [30], tightly coupled with:

- Dynamic voltage scaling, used to adapt supply voltage to the target (FPGA) as the temperature changes [38].

Device-Level design techniques

Many of the latest FPGA devices incorporate various low-power device-level technologies. This also applies to processors. We list some:

- Triple gate oxide technology, providing a choice of three different gate thicknesses and subsequently providing a trade-off between performance and static power.
- Increase in LUT sizes within the logic block.
- Incorporation of low power techniques in FPGA CAD tools aiming at early power prediction.
- Utilization of power-managed modes (RUN, IDLE, SLEEP, etc...) and clock frequency switching (fast $\rightarrow$ slow and vice versa) specially in microprocessors/microcontrollers targets.

Circuit- and architecture-level design techniques

The architecture- and the circuit-level implementations of the FPGA are key in reducing power, since they directly affect the efficiency of mapping applications to FPGA resources, and the amount of circuitry to implement these resources. Here we list some:

- Energy-efficient routing and low-swing signaling.
- Power-gating, applied to the switches in the routing resources to reduce static power.
- Optimizing number of interconnects.
- Delay insertion for glitch elimination by adding configurable delay elements to the input of each logic block.

Significant improvements have been made to enhance power and energy efficiency of FPGAs, ASIC and processors. At the system level, power reduction can be obtained by optimizing management and scheduling of system resources.

2.3 Power Estimation and Modeling Techniques

Power consumption modeling and subsequent estimation for embedded systems refers to the process of creating power models and then estimating the power consumption of a digital circuit usually before being deployed or even built. Many existing commercial tools tackle this issue with different approaches and alternatives on different design levels. Subsequently, two subsets of modeling and estimation techniques have been widely investigated and applied on low- and high-level design alternatives.

2.3.1 Low-Level Techniques

Low-level power estimation techniques refer to methods used to predict the power consumption of an embedded system by considering the detailed implementation of the system, such as the specific components and their properties, and the circuit-level details.

In order to acquire a better understanding of High-Level power estimation methodologies, going through the low-level techniques is a must. Therefore, we briefly list and discuss low-level estimation and modeling techniques paving the way for the High-Level version. For this purpose we only focus on hardware.

FPGAs are being used in various applications due to their reconfigurability (a major advantage over ASIC) and scalability. In this section, we tackle the issues related to power modeling and estimation of both FPGAs and ASIC. The main focus will be on CMOS and SRAM technologies, both under the low-level power estimation and modeling domains.

Power Estimation Techniques

Power estimation techniques are considered to be very efficient alternatives to traditional power measurements given that the characterization phase is not required. Three methods are briefly described: 1) probabilistic-based, 2) simulation-based and 3) statistical-based methods [56].

The probabilistic power estimation methods use data characteristics rather than real data. They are generally derived from the static probability and the transition probability of a given signal.

The simulation-based power estimation, widely used in CAD tools, is achieved by applying stimuli signals at the input of a specific logic design and initiating a simulation in order to capture the outputs. Current and voltage values, capacitance, operating frequency and switching activity are the main

required set of information in order to obtain power consumption estimation. The abstraction level has a direct effect on the previously listed type of information. Precision and genericity are considered to be the main advantages, while heavy memory usage and execution time remain a major disadvantage especially when speed (rapid prototyping) and resource optimization are required.

The statistical power estimation methods have common features with the previously described simulation-based techniques. A random bit sequence (stimuli) is applied to the primary inputs of a specific logic circuit and subsequent simulation is performed using an off-the-shelf power estimator until a satisfactory precision is hit.

As a quick comparison, the simulation-based techniques provide two main advantages: 1) high accuracy and 2) generality. As for the disadvantages, long simulation time and significant memory resources remain a major limitation. The probabilistic-based methods deliver faster estimation since they do not require waveforms generation (usually time-consuming), but they provide relatively low-accuracy results due to the use of simple delay models for components and average signal probabilities. The statistical-based power estimation approaches can be classified somewhere in between since they are considered as a trade-off between the accuracy of the simulation-based and the estimation speed of the probabilistic-based techniques.

Power Modeling Techniques

In the following section we present the power modeling methodologies divided into four categories: 1) analytic, 2) table-based, 3) polynomial and 4) Neural Networks (learning) [56]. As for the comparison metrics, four criteria have been defined: modeling level (circuit or component), modeling effort (low, moderate and high), modeling granularity (fine or coarse) and characterization techniques.

Relating power consumption to the switching activity and the capacitance of a specific design is commonly known as the analytic modeling technique. In other words, these kind of techniques are based on the theoretical equation of the dynamic power dissipation of a CMOS transistor. In complexity-based techniques, the capacitance value is roughly estimated from the design architecture. In activity-based models, the power modeling issue is addressed based on the entropy concept. It is used to evaluate the average activity of a circuit. A relatively good accuracy is achieved using the analytical-based modeling when applied to simple components, however the effect of glitches is not taken into consideration.

The tabulation process of power values is considered to be a table-based modeling technique following a characterization phase. Look-up tables do not require mathematical models in comparison to the previously described analytical approach. While feasibility and moderate modeling efforts being the main advantages of such techniques, considerable computational efforts (with growing tables) and heavy storage memory usage represent the limitations.

While long simulations and extended list of parameters usually limit the Design Space Exploration (DSE), polynomial (regression-based) power modeling may be a suitable choice for power prediction. Polynomial models can be used to assess the linearity between power values and one or more independent design parameter(s). The majority of these regression-based methods propose models for hardware-specific components such as operators and interconnect elements, however complex logic circuits are not considered; not forgetting the limitation in the number of input variables during modeling.

As a definition, the Artificial Neural Networks (ANNs) are information processing structures providing the (often unknown) connection between the inputs and the outputs [23]. They are based on connected neurons that propagate specific information among them. Compared to the previously mentioned polynomial-based methodologies, neural networks can perform both linear and non-linear regressions while providing higher efficiency. Multi-Layer Perceptron (MLP) is a type of feedforward artificial neural network that consists of multiple layers of interconnected perceptron units, also known as nodes or neurons. MLP networks are widely used due to their high performance and high accuracy (<3% error), however Convolutional Neural Networks (CNNs) seem to offer more promising results.

As a quick comparison, analytic models deliver reasonable fitting accuracy with relatively low computational and modeling efforts since no power characterization is performed; however, generalizing an analytical model to a global system remains a challenge. Neural networks method generally outperforms other modeling techniques as far as accuracy, scalability/expandability and memory usage are concerned; however, modeling efforts are considerable in this case. Finally, the polynomial-based techniques provide a good trade-off where accuracy, modeling efforts and memory resources are balanced; thus they are considered as a good alternative.

2.3.2 High-Level Techniques

High-level power estimation techniques refer to methods used to predict the power consumption of an embedded system without requiring detailed information about the implementation of the system. These techniques typically use

models or abstractions of the system to estimate power consumption. Some examples of high-level power estimation techniques include:

- Statistical modeling: This method uses statistical techniques to estimate power consumption based on measurements or simulations of the system.
- Machine learning-based techniques: These methods use machine learning algorithms to learn the relationship between system inputs and power consumption. By definition, machine learning techniques are a set of algorithms and statistical models that allow computer systems to automatically learn patterns and insights from data, without being explicitly programmed for each individual task. These techniques enable computers to improve their performance on a task by learning from experience or data, rather than relying solely on rules programmed by humans.
- Hardware-in-the-loop simulation: This method uses hardware-in-the-loop simulation to estimate the power consumption of the system by simulating the system's behavior and interactions with its environment.
- Power-aware simulation: This method uses simulation tools that include power models of the system's components to estimate the power consumption of the system.

These techniques can be used alone or in combination in order to provide accurate and efficient power consumption estimates, which can be used for power management, optimization, and design of embedded systems. This specific topic will be thoroughly detailed in Chapter 3.

2.4 Conclusion

In summary, power consumption optimization is an essential aspect of digital circuit design, particularly in embedded systems, for which power budgets are often limited. To achieve efficient power consumption, various techniques can be employed, such as voltage scaling, clock gating, and power gating. Power consumption dependencies, including hardware components, software algorithms, and user behavior, must be taken into account during power optimization in embedded systems. Understanding the dependencies is critical in developing power-efficient embedded systems that meet the power budget requirements.

Furthermore, power optimization and power modeling and estimation are two interrelated concepts that are essential for designing power-efficient embedded systems. Power consumption modeling is a critical tool for identifying areas for power optimization, while power optimization techniques can inform the power consumption modeling process and validate its effectiveness.

In this background chapter we have presented a context study covering an overview of power consumption sources and optimization techniques. Power estimation and modeling methodologies have been briefly described covering both low- and high-level domains. Techniques for the low-level alternative have been summarily listed. The high-level alternatives, considered as a hot topic today and subsequently being our main concern, will be thoroughly detailed in chapter 3.

Chapter 3

High-Level Estimation Techniques in Embedded Systems Hardware - SOTA

3.1 Introduction

With design complexity on the rise and High-Level design tools becoming more widely available, the Field Programmable Gate Array (FPGA) market is poised for large-scale adoption of High-Level design methodologies. In fact, most analysts and industry leaders agree that a considerable number of FPGA designers will eventually switch to High-Level design techniques [13]. This also applies to Application Specific Integrated Circuit (ASIC) [35][3], considered as an important part of embedded systems especially with integrated hard-core processors, tightly coupled with internal logic [8]. As the FPGA/ASIC capabilities have grown, so have the designs, current tools and languages (mainly HDL) do not match the complexity required for advanced digital hardware systems. In the recent years several commercial High-Level Synthesis (HLS) tools have reached maturity, providing a new method to implement FPGA/ASIC designs, or at least some parts of it but at the same time creating challenges in power consumption estimation [52].

Today, modeling FPGA power consumption is a necessity in two scenarios. First, it is required to perform power-oriented design space exploration at design time. The elaboration of power-efficient designs usually requires multiple iterations of power estimation and design refinement steps, which leads to a long design time and low productivity. In this scenario, having accurate and efficient power models can help to rapidly explore design choices and perform high-level simulations. Second, power modeling may also be used at run-time. In most systems today, it is required to have efficient power monitoring and management. This can be highly useful in taking decisions regarding Dynamic

Voltage and Frequency Scaling (DVFS) or tasks' scheduling in CPU-FPGAs systems. In such cases, a power model has to be implemented in the device itself and run either under software control on a processor or in a dedicated hardware component. The main requirement is to have a "simple" yet efficient model that is capable of extracting basic features of input signals to obtain an accurate value of the circuit consumed power while taking into consideration both speed and accuracy.

Embedded systems designers making the switch to High-Level design methodologies, enjoy some obvious advantages, mainly the easy-handling of increased complexity at a higher level of abstraction, shortening design cycles and improving quality by early functionality verification. Widening the scope, High-Level power estimation and modeling methodologies crossed beyond FPGA and ASIC limits, opening new horizons for microcontrollers/microprocessors-based systems with different but various techniques and architectures (soft-cores, hard-cores, customized instruction set, systems on a single chip, etc.). That said, power estimation and modeling at high level became a necessity. This chapter provides an extensive overview on high-level power estimation and modeling techniques. It also presents a comprehensive classification and comparison between different methodologies when applied to designated target models.

This chapter is organized as follows: in section 3.2 we list and define various power estimation techniques. In section 3.3 we sort and classify estimation techniques and their respective target models. The classification discussion is presented in section 3.4. Finally we conclude in section 3.5.

3.2 Estimation Techniques

In this section we list and define various power consumption estimation techniques and their corresponding target estimated models. Based on the covered literature, four major power consumption estimation techniques were extracted: 1) the simulation-based methods, 2) the learning-based approaches, 3) the statistical-based methods and 4) the measurement-based alternatives. As for the target models, for each of the previously mentioned estimation techniques, three categories of modeling targets were detected and defined as follows:

1. Components/Circuits: components cover both, operators (arithmetic and logical) and basic logic elements (multiplexers, shift registers, etc.). Circuits modeling is mainly related to capacitance and interconnect wires.

2. White/Black-box IPs: commonly known as reusable units of logic cells. IPs, considered as part of the High-Level design, are valid for both FPGAs and ASIC. A white-box IP is a well known entity where its inputs and outputs are pre-defined. A black-box IP represents an often unknown entity where its inputs can be randomly driven and its outputs are likely to be ignored.

3. Microprocessors/CISA/(MP)SoC: microprocessors (µPs) are complex modeling targets, their power consumption heavily relies on their architecture and application software. The Customized Instruction Set Architecture (CISA) is a highly optimized approach for simplifying digital hardware and thus reducing power consumption. The Multi-Processor (MP)SoC is mainly an array of processors in a single System on Chip (SoC).

Figure 3.1 shows an overview of the power estimation techniques and their respective estimated models.

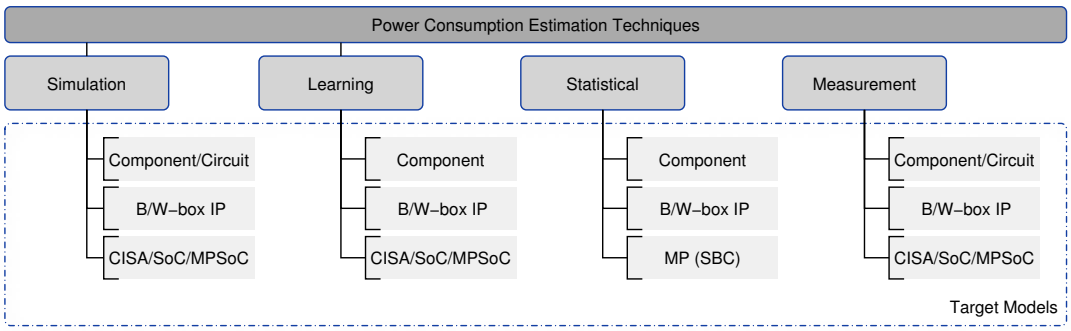

FIGURE 3.1: Estimation techniques and target models.

3.2.1 Functional simulation-based methods

The simulation-based power consumption estimation techniques are commonly used in Computer-Aided Design (CAD) tools. This is mainly done by applying various sets of inputs for a given system to be simulated and subsequently collecting and capturing sets of desired outputs leading to power consumption values. Current and voltage values, capacitance, operating frequency and switching activity are the main required set of information in order to obtain power consumption estimation. The accuracy of such techniques is proportional to both processing time and computer memory resources' usage [56]. Here we list some of this technique's target models based on related work:

Components/Circuits

The simulation-based modeling technique described in [65] is initiated by an iterative optimization step during model generation phase. This approach consists in defining modeling variables that are suitable for High-Level Power Estimation (HLPE). Five variables are mainly defined: 1) FPGA device, 2) basic component (generated using Coregen), 3) Hamming distance, 4) signal distance and 5) signal bitwidth. The power characterization is done using XPower while static energy is being deducted from the total power consumption and pads power is being estimated using the ORINOCO pad estimator tool (mainly used for pad capacitance extraction). The main advantage of such a technique is that every component has to be characterized only once. As for the disadvantages, this method does not include neither registers nor memory power estimation and it is not fully automated. The RMS error varies between 6.68% and 12.61%.

TLM POWER3, a methodology described in [24], consists in creating an add-on SystemC library that accumulates dynamic energy. It is based on the extended generic payload where new physical units for voltage, distance and area have been added. Output reports are generated at the end of this process listing: 1) power consumption, utilization and physical layout and 2) automatic add-up of power and energy used globally. This approach is described to be an easy-to-use power estimation add-on to SystemC TLM.

A pre-silicon power estimation methodology using circuit-level simulation for coarse-grained FPGA architecture has been proposed in [16]. Resource characterization has been used in order to find the effective capacitance. This is done by global wire modeling taking into account the longest interconnect wires. The efficiency in this case is dependent on a number of known simplifications of the FPGA architecture. The power estimation is therefore achieved by finding the utilization of each resources' switching activity. This includes extraction of resource usage (LUT, FF, Carry, etc...) and subsequently averaging the switching activity of each resource. As a result, an average estimation error of 18% has been recorded while the maximum went up to 27%. This proposed method was found to be efficient and accurate, based on clock power, switching activity and dominant interconnect capacitance power. However, being specific to Spartan-3 family, presents a major disadvantage in addition to RAM and shift registers being excluded from estimation.

The hierarchical library-based power estimation concept presented in [44] applies to novel circuit of components, emerging memory devices, architecture of time-multiplexing technique and many others at different levels. It supports coarse-grain or fine-grain estimation defined by users for achieving

complexity-accuracy trade-off. The power estimation procedure is a top-down approach; FPGA power is divided into logic, routing and clock power. After extracting power estimation parameters, the calculation of the corresponding power is done while ignoring the detailed implementation of the component. The power estimation of a single component (when reaching the user-defined granularity) is achieved by extracting static power consumption from library while the dynamic power is extracted from the switching activity and the operating frequency. As results, the accuracy of this power estimation methodology depends on the power information given by the hierarchical power library. The information in context includes: static power, dynamic power, ReRAM power estimation and power gating.

In [43], *fpgaEVA-LP2* is a mixed-level power model associated with cluster-based logic blocks and island style routing structures. This methodology takes into consideration 3 power sources: 1) switching power, 2) short-circuit power and 3) static power. Two types of signal transitions are covered: the functional transition (used to perform the required logic) and the spurious transition also known as glitches. The dynamic power model consists of a switch-level model for interconnects and a macromodel for LUTs. This particular paper focuses on transaction density and glitch analysis. While assuming a signal probability of 0.5 and a transaction probability of 0.85, a large number of input vectors is generated. The total power consumption can be broken down into static power and dynamic power (switching and short-circuit). The dynamic power model includes a switch-level model for interconnects and a macromodel for LUTs. The static power, also known as leakage power includes: reverse-bias, sub-threshold, gate tunneling, gate induced drain, etc. This power analysis framework can be used to investigate the impact of circuits, architectures and CAD algorithms upon FPGA power dissipation. As a result, the absolute average error is about 8%.

In [6], Modeling and Analysis of Real-time and Embedded systems (MARTE), a standard Unified Modeling Language (UML) profile promoted by the Object Management Group, is used to create an extension for modeling dynamic power management. For this purpose, a subset of MARTE has been selected, mainly the Hardware Resource modeling (HRM). The Dynamic Power Management (DPM) profile has been developed taking into consideration the requirements of modeling the DPM schemes of modern embedded systems with complex strategies over multiple hardware components. The DPM profile provides model reuse by separating component-level aspects from platform-level aspects on one side and platform-level aspects from application-level on another side.

White/Black-box IPs

The proposed simulation-based *Transpilation* method in [31], translates the gate-level component from Verilog to C, trains an energy model while providing an executable script to run system simulation and subsequently calculates energy consumption. This method is fast compared to co-simulation and test-benches approaches but it is partially valid since it does not take into consideration glitches effects and static energy. The resulting estimation error is relatively high (between 55.2% and 152.8% at 10000 samples).

The power consumption estimation of an IP in [11] at Transaction-Level modeling (TLM) is done by capitalization of its existing transaction level model. For this proposed method, switching activity and cells extractions are used. Switching information is extracted using test cases after generation of Value Change Dump (VCD) files. Cells are extracted after converting TLM into a synthesizable format by design compiler. The total power consumption is then calculated as the summation of leakage power and dynamic power. These simulation-based TLM results reveal a total power estimation error of 12.03% and a speed-up of 20x compared to RTL's.

A simulation-based High-Level IP characterization methodology presented in [70] is based on separating the activity of the IP from its implementation. The aim was to create a power model that can be used at different frequencies, layouts and implementation technologies. Therefore, the goal was to abstract the power consumption model of an IP with relatively simple equations. Two IPs were analyzed: 1) a microblaze processor and 2) BRAM memory. While being precise and simple, the average percentage error of the created models was found to be under 15%.

The methodology described in [29] aims at providing power estimates of communication systems ahead of implementation. The proposed approach is solely related to hardware with no software considerations assuming that the wireless communication system is entirely constructed using interconnected hardware IP cores set. This process can be done in two phases: 1) IP characterization using Xilinx Power Analyzer (XPA) and modeling using SystemC and 2) building the system by IP interconnections (previously developed in phase one). The aforementioned technique has been applied to an Orthogonal Frequency Division Multiple Access (OFDMA) IP part of the Long-Term Evolution (LTE), the 4^{th} generation of radio technology for mobile communications. As for the results, the maximal absolute error was found to be less than 4% compared to estimated values by XPA.

In comparision with [11], [54] took one step forward and included thermal analysis. Several tools were presented and compared in order to simulate the

power and thermal behavior of a chip along with its functionality. The aim was to augment a functional SystemC/TLM model with non-functional information and subsequently perform power and thermal analysis. The focus was on co-simulating an existing thermal model with a functional SystemC/TLM simulation. Experiments were conducted on both a soft-core processor (MicroBlaze) and an IP (VGA controller); no fundamental issues were detected with the co-simulation techniques on both the SystemC/TLM simulation and the external power/thermal solvers.

The methodology described in [60] is used to estimate energy dissipation in hardware at any level of abstraction in SystemC with *Powersim*. This technique is based on a *Measure space* that consists of the definition of the universe set formed by an elementary event and its duration (as a pair). As for the application of the model described at system-level and composed of different modules, the total energy dissipation can be found by summing the energy dissipated by each module. The energy models can be divided into three sets: 1) logic operators (and, or, xor, not, etc.), 2) arithmetic operators (add, subtract, etc.) and 3) the computational cost. Two test applications were analyzed: a JPEG encoder implemented in hardware and FIR filter implemented in a microcontroller. As for the results, the mean relative error in estimation was about 15.8%.

Microprocessors/CISA/(MP)SoC

The HLS-designed, Customized Instruction Set Architecture (CISA) proposed in [5] can be categorized as a simulation-based approach. A well defined selected number of MIPS instructions is being used. For synthesis, power estimation and area analysis, Vivado HLS has been adopted. The estimation is based on the number of consumed cycles by CISA. This technique has been tested on a Dot Product (DP) model and a 128-bit Advanced Encryption Standard (AES-128). The main advantages of such a methodology can be highlighted as follows: a) the power consumption is only 50% to 70% of the total power, b) the dynamic power is 20% to 60% of the fully implemented soft core, c) the smaller the number of instructions, the less resources used and the smaller the occupied area, and d) the pipelining ability. One major disadvantage to mention is that CISA is slower in execution than a full featured ISA.

The technique described in [63] is based on abstraction execution profiles called event signatures. The system-level power modeling framework is based on the *Sesame* MPSoC simulation tool. This signature-based power estimation

provides an abstraction of processor activity and communication in comparison to traditional power models. It includes computational and communication event signatures as well as signature-based system-level power estimation. This simulation-based methodology was validated by *Daedalous* tool with an M-JPEG encoder application. The results were compared to measurements done using a PMBus (based on I^2C) current sensor. The estimation error was about 7% with a standard deviation of 5%. One major disadvantage of this technique is that it relies heavily on many existing tools which prevents full integration and automation.

However, in [73] a top-down power and performance estimation methodology for heterogeneous multiprocessor SoC is proposed and applied at the Electronic System Level (ESL). This is achieved by combining model-based design and spreadsheet power models. The benefits of such a technique is that 1) the estimations can be done early in the design, 2) different design options can be assessed with minimal efforts, 3) dynamic power management is taken into consideration and 4) the possibility of a fast power and performance analysis. For the modeling process, two steps are followed: 1) the characterization of the MPSoC platform and 2) the application modeling. As a use case, an actor-oriented model of an MP3 decoder has been adopted. The estimation performance accuracy was not expected to be high since this approach is intended to provide relative figures for different design options comparison.

The simulation-based methodology presented in [74] is mainly applied for extending system performance modeling frameworks. An *XScale*-based case study is used for verification and a number of components is characterized such as: processors, caches, buses, SRAM, DRAM, peripherals, etc. This technique provides a realistic validation of a system-level execution-driven power modeling (with a fair level of power details for various models). It is considered to be scalable, efficient and validated for incorporating fast and accurate power modeling capabilities into SystemC. The worst case percentage error between measured and estimated values was below 10% with an average error of 5%.

In [75] the High-Level power estimation model for SoC is based on extended and optimized event signatures. The aforementioned estimation model has three primary parts: 1) the summation of every section's power consumption of the SoC design, 2) power consumption produced by event signatures consisting of bus, memory and communication power sub-models and 3) power consumption disparity due to the subtrahend of the number of LUTs of the complete design and subsequent summation of each part. This proposed model

takes into consideration the effect of experiential power estimation data, performance events in addition to the LUT changes. The validation results revealed an error rate of 1.185%.

The multi-core processor (dynamic) power consumption estimation tool presented in [27] extracts information at the instruction level as well as at the architectural level. The MIPS32-based processor has been created using Open Virtual Platform (OVP) and virtual machine interface libraries of Imperas' Instruction Set Simulator (ISS). Two, four and eight processor cores were simulated using ISS where different benchmarks were used as application. As a result, it was noted that the energy consumption is inverse-proportional to the number of cores used on the platform. The reason behind is that the same application takes more time on a platform having less processor cores.

On another hand, a formal description of a power consumption estimation approach of embedded systems is presented in [72]. An embedded system consisting of a hardware and a software is denoted as a System Model (SM). The hardware part of the system is reflected in the Operational Model (OM) while the software part is described in the Application Model (AM); UML extended with MARTE profile elements, was used for this purpose. To analyze the power consumption of the system, SM, both OM and AM were combined and converted into a stochastic Petri net. As an example and direct application to the proposed method, an industrial control system was used and its average power consumption was estimated without being compared to other references.

3.2.2 Learning-based methods

By definition, learning-based techniques are subsets of Artificial Intelligence (AI) methods, providing automatic power consumption estimation, acquired and improved from experience without being explicitly stored or programmed. Usually, in such techniques, data sets, commonly known as training sets, correlating inputs to outputs have to be collected and applied prior to initiating learning methods. Based on the covered literature, tree-based algorithms and Artificial Neural Networks (ANNs), both under the supervised learning methods, are being widely adopted. A supervised learning method uses special patterns to identify specific characteristics. A pattern is always consistent and recurrent [23]. Here we list some of this technique's target models based on related work:

Components/Circuits

In [57] a power model based on neural networks was presented in order to predict the power consumption of digital operators in FPGAs. Any digital hardware system can be represented by a set of components, each with a specific function and selected during the High-Level design process. The suggested method represents each operator (component) by two sub-models denoted by M1 and M2. M1 predicts the power consumption for given Signal Rates (SR) and Percentage Logic High (PLH), while M2 estimates the signal activity of the outputs in addition to the PLH according to the inputs. A characterization phase is required by applying a set of stimuli to a given operator and collecting power information using XPA tool. Both M1 and M2 are implemented using Multi-Layer Perceptrons (MLP) feed forward neural networks. As for the results, various operators were chosen (adders, multipliers, etc...); the percentage error was about 0.01% at the component level and less than 8% at the system level (mainly composed of cascaded components).

NeuPow, a complement of the methodology described in [57] and [59], is an Artificial Neural Network (ANN) for power and behavioral modeling of arithmetic components in 45 nm ASIC described in [58]. The proposed method is achieved by propagating predictors between the connected neural models in order to estimate dynamic power consumption of individual components. Two different ANN models are used for each arithmetic component: 1) for estimating power 2) for determining the activity propagation. Initially, data set vectors are generated and fed into the inputs of both ANNs. Dynamic power and output data depend only on the input data for a fixed technology and frequency. Bit width, packet length and number of packets represent the stimuli parameters. This technique is divided into two sections: power and behavioral characterization, and, power and behavioral modeling. The final step is to construct the library based on the two previously mentioned ANNs: ANNP to predict power consumption and ANNB to predict the output feature vector. This learning-based estimation methodology applies for both component and system level. As for the results, an RMSE of +/- 1.5% was recorded for the component level and 8.5% for the system level with an estimation speed up of about 2490x compared to traditional tools.

White/Black-box IPs

The *HL-Pow*, a learning-based methodology described in [46], is a power modeling framework based on machine learning. It describes an automated feature construction flow to efficiently identify and extract features that exert a major

influence on power consumption. The HL-Pow design flow has two phases: 1) power model training with a collection of applications and 2) power inference for new applications. In the training phase, a number of representative applications described in C or C++ are used to generate training samples for power modeling. Two main types of features are taken into consideration: architecture and activity. For the power model generation, *HL-Pow* constructs a total number of 256 features consisting of 11 architecture features and 245 activity features. The regression models were built using many supervised learning methods, such models are: 1) linear regression, 2) Support Vector Machine (SVM) and 3) tree-based model. As for the results, the power modeling error was 4.67% away from measured power.

The decision tree-based learning method described in [41] relies on extracting data-dependent invocation-by-invocation power model from gate-level cycle-by-cycle power traces. The process starts by applying input vectors simulation to black box IP and its TLM model. Based on output traces, invocation-by-invocation power model with fine-grain data-dependent is used while internal signal activity is indirectly observed from switching activity of I/O signals. SciKit-learn machine learning library was used for model selection and training. The trained cycle-level models are combined to predict invocation-level execution power of the IP. As a result, the method provides a 9x faster prediction than cycle-level with less than 2% average error.

However and opposed to [41], the learning-based power modeling approach described in [39] is based on annotating functional hardware descriptions and capabilities allowing the capture of data-dependent resources, block or I/O activity without significant loss in simulation speed. Activity models are first generated then machine learning techniques are leveraged in order to synthesize power models at different granularities. Compared to commercial gate-level or RTL estimation tools, this approach achieved an estimation accuracy of 10%, 9% and 3% for cycles-, block- and invocation-level respectively . This technique is fully automated by integrating with commercial HLS tools.

Microprocessors/CISA/(MP)SoC

Existing CPU power models rely on either generic analytical power models or simple regression-based techniques that suffer from large inaccuracies. Today, machine learning techniques are proposed to build accurate power models. Authors in [32] present a hierarchical power modeling approach that deals with power models for CPUs at micro-architecture level. A decision tree is build for a RISC-V core that can predict cycle by cycle power with less than 2.2% error.

More recently, a micro-architecture power modeling framework has been proposed in [79]. It provides a power modeling flow and methodology and trains ML calibration models using given configurations of the core and provide power estimates for other configurations. Compared with state-of-the-art power models, this approach can reduce the mean absolute percentage error (MAPE) under different cross-validation strategies by 3% to 6%.

3.2.3 Statistical-based methods

The statistical power estimation methods have common features with the previously described simulation-based techniques. A random bit sequence (stimuli) is applied to the primary inputs of a specific logic circuit and subsequent simulation is performed using an off-the-shelf power estimator until a satisfactory precision is hit. Here we list some of this technique's target models based on related work:

Components/Circuits

For the statistical-based methods applied to component/circuits, it is very rare to find on-going works on the topic. A previous work, [47] deals with power consumption estimation by analyzing statistical properties of interface signals. It is mainly based on the evaluation of the transition activity which is 40% smaller than the real values, which leads to an accurate power consumption estimation. More recently, authors in [62] presented an evaluation of power estimation flows used in standard cells ASIC for video applications. The results show that flows based on statistical approaches can present a relative error about 40% for evaluated designs.

White/Black-box IPs

Again, due to lack of recent work, the amount of references is found to be at minimum. In [19], the authors propose a High-Level power estimation model based on input signals statistics defined as: the input signal probability, the average input signal density and the spatial correlation. In their experiments, authors indicate an average error of 29.63% regarding power estimation. More recently, in [10], the methodology has been applied on a SDRAM IP core and the maximum estimation error was found to be 12.5%.

Microprocessors/CISA/(MP)SoC

In contrast with [27], [61] is a power estimation prediction technique based on a statistical model. Its portability is analyzed on new generations of Raspberry Pi Single Board Computer (SBC). In such systems, the power consumption does not depend on the hardware only but also on the use of the software and its internal characteristics. Two different benchmarking tools have been used having different computational behaviors, parallel applications source code compilations, collection of performance counters, linear regression model training and model validation. The final statistical model is based on linear regression and subsequent generation of 8 power coefficients leading to one unified model formula. As for the results, the maximum prediction error was about 18.46%, an error dispersion of 4.14% and an average error of 4.76%

3.2.4 Measurement-based methods

A natural approach for evaluating power consumption consists in conducting real measurements on hardware after design implementation. This is usually done either a) via built-in current sensors (on-board or on-chip), or b) via a well defined external instrumentation setup [67].

Built-in measurements: This kind of measurements is portable and may be achieved using 1) on-board current sensors and voltage regulators (mostly monitored over the standard Power Management (PM) Bus or using on-board Analog-to-Digital Converters - ADCs) or 2) on-chip current sensors (mainly implemented in ASIC). These techniques' integrity is however dependent on the fidelity of the sensors in context and is highly affected by both the electrical noise and interference.

External measurements: Another alternative would be to use external instruments to measure power consumption. This is mainly achieved by connecting analog acquisition instruments (basically built using high accuracy ADCs) to the Device Under Test (DUT). This is only possible when the DUT permits such a procedure by exposing 1) the core supply load (via a Shunt power resistor) and/or 2) a set of test points representing the analog power consumption on specific sub-circuits. These acquisition systems are usually hooked to PC software for storage and analysis.

Here we list some of this technique's target models based on related work:

Components/Circuits

The power characterization methodology described in [20] relies upon a set of measurements where both static and dynamic power are extracted. The proposed method is based on downloading different hardware designs with a variable number of resources (connections, CLBs, LUTs, registers, IOBs, etc.) on FPGAs. The total power consumption is determined based on power consumption subsets related to each component. The variation of static and dynamic power consumption with respect to multiple FPGA targets is monitored by implementing a unified design on many identical FPGA boards. This methodology is mainly used to determine the unit power consumption of the direct horizontal and vertical lines in addition to LUTs.

White/Black-box IPs

A cycle-accurate energy measurement and High-Level characterization methodology based on the switched capacitor technique was adopted in [42]. For the measurement, a pipelined Analog to Digital Converter (ADC), a vector generator and an Ethernet-based system management CPU has been utilized. The energy, for both low- and High-Level approaches, has been characterized. Since our main focus is on High-Level techniques, we only present the HL part of the energy characterization. For this purpose, a state-machine-based technique has been selected while separating static and dynamic power consumption. This method applies to SRAM-based FPGAs. Two IPs were analyzed, LCD and SDRAM controllers, however no comparison with alternative methods has been explicitly revealed, thus no % error has been published.

Microprocessors/CISA/(MP)SoC

The method of power consumption estimation proposed in [2] is based on an experimental bench developed for Virtex-6 FPGA. This method consists of duplicated Processing Units (PUs) mainly MicroBlaze softcore processors (1, 2, 4, 8 and 16 cores) where three different C programs were executed for each of the five designs. Two versions v1 and v2 of the five implemented designs were used to evaluate the proposed method. Results of measured values were compared to XPA simulated values and it was noticed that the 16 cores system has higher estimation accuracy (% error less that 4) than the 1 core system (% error higher than 30).

3.3 Approaches classification

Various power consumption estimation techniques and methodologies were investigated along with associated target models. In order to sort and classify the aforementioned techniques and models, a set of metrics had to be defined. Given the variety and extensiveness of the covered literature, and due to the lack of common ground, the metrics had to be divided into quantitative and qualitative sets. We list and define the extracted metrics as follows:

- Dependency: a given estimation technique may or may not be dependent on specific hardware, software, tools and/or technologies.
- Characterization: a given estimation technique may or may not require a characterization phase ahead of estimation and modeling. The characterization reveals the distinctive nature of a given target model and highlights its special features/characteristics such as capacitance, switching activity, resources, etc.
- Estimation effort: a qualitative metric represented by either little effort (+), moderate effort (++) and considerable effort (+++).
- Estimation error: this quantitative metric is not always explicit yet it could be totally absent. The % error could be either minimum (min), maximum (max) or average (mean/avg).
- Modeling effort: similarly to the estimation effort, this qualitative metric is also represented by either (+), (++) or (+++).
- Modeling level: this metric reveals the level at which the model was created. Possible options are: RTL, system, TLM, component, HLS, IP or a combination of any two (ex: Comp/Sys, IP/Sys, etc.).

Table 3.1 encapsulates and classifies both the power consumption estimation techniques and their investigated target models based on the previously described set of metrics. References were selected based on their exposed data and on the extracted information following the diagram in Figure 3.1. The table's entries were sorted based on the number of occurrences in the literature for both the estimation techniques and their respective target models in descending order.

3.4 Discussions

Based on the covered literature, the following is a comprehensive comparison between various estimation techniques applied to common target models:

TABLE 3.1: Classification

Methodology	Target model	Reference	Dependency	Characterization	Estimation effort	Estimation error	modeling effort	modeling level
Simulation	IP	[31]	n/a	No	+	mean 100%, max 150%	n/a	System
		[11]	n/a	No	++	mean 12%. max 20%	n/a	TLM
		[70]	n/a	Yes	++	mean 15%, max 50%	n/a	TLM
		[29]	n/a	Yes	++	max 4%	++	System
		[54]	LIBTLMPWT	No	+++	n/a	+	TLM
		[60]	Powersim	No	++	mean 15.8%	+	IP/Sys
	µP/CISA/(MP)SoC	[5]	Vivado DS	No	++	n/a	n/a	HLS
		[63]	n/a	No	++	avg 7%	++	System
		[74]	XScale P	Yes	++	mean 5%, max 10%	+++	TLM
		[75]	n/a	No	+++	1.185%	+	System
		[27]	Virtual platforms	No	++	n/a	n/a	Instruction
		[72]	MARTE	No	++	max 2%	++	Comp/Sys
		[73]	ESL	Yes	++	n/a	++	System
	Component/Circuit	[65]	User Intervention	Yes	++	min 6.68%, max12.61%	++	Component
		[24]	TLM Power3	No	+++	n/a	n/a	TLM
		[16]	Spartan-III	Yes	+	mean 18%, max 27%	n/a	Comp/Sys
		[44]	n/a	No	++	min 24%	n/a	Comp/Sys
		[43]	n/a	Yes	++	avg 8%	++	Component
		[6]	MARTE	No	++	n/a	++	Comp/Sys
Learning	IP	[46]	n/a	Yes	++	4.67 %	++	HLS
		[41]	Virtual platforms	No	++	mean 3%, max 15%	++	TLM
		[39]	n/a	Yes	++	avg 7.33%	++	System
	µP/CISA/(MP)SoC	[32]	RISC-V	Yes	++	max 2.2%	++	Gate
		[79]	RISC-V	Yes	++	max 3%	++	System
	Component	[58]	n/a	Yes	+++	mean 1.5%, avg 8.5%	+++	Comp/Sys
		[57]	XPA	Yes	++	min 8%, max 17%	++	Comp/Sys
Statistical	IP	[19]	n/a	Yes	+	avg 29.63%	+	System
		[10]	DataFit tool	Yes	++	max 12.5%	++	System
	P (SBC)	[61]	RPI SBC	No	++	avg 4.76%	+	System
	Component	[47]	REPLICA	Yes	++	n/a	++	System
		[62]	RC/iRun/Cadence	No	+	40%	n/a	RTL
Measurement	IP	[42]	SRAM FPGA	Yes	+	n/a	+	System
	µP/CISA/(MP)SoC	[2]	n/a	No	++	min 4%, max 30%	n/a	System
	Component	[20]	n/a	Yes	+	n/a	n/a	Component

- For the white/black-box IPs target models, the estimation error was found to be almost inverse-proportional to the estimation effort targeting the simulation-based methods; the error range was higher and wider than the learning-based counterparts. Considerable estimation efforts in simulation-based methods definitely lead to satisfactory estimation % error, highly comparable to the learning-based alternatives, yet providing better results in some specific cases. As a global comparison, the learning-based techniques provide better performance in estimating power consumption at moderate efforts.

- For the component/circuit target models, a smaller and narrower range of estimation error was clearly detected with the learning-based methods. These latter, achieved with considerable efforts, provide higher efficiency compared to the simulation-based alternatives where little to moderate efforts were applied.

- µP/CISA/(MP)SoC target models, estimated using the simulation-based methods, showed a % error >30%. Learning-based alternatives presented better results with error around 3%. However, good estimation results can be obtained by just applying moderate efforts.

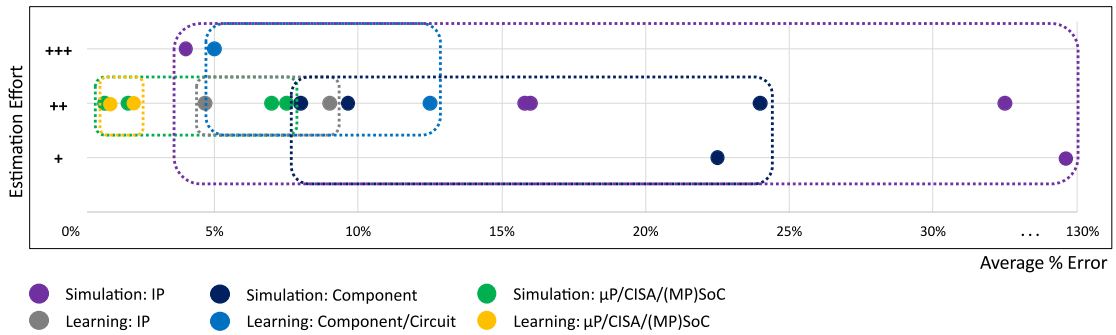

FIGURE 3.2: Estimation Effort v/s Average % Error - Clusters Chart.

Figure 3.2, a cluster-based representation, combines a number of power consumption estimation techniques applied to various target models. The purpose of having such a global chart is to get a wider understanding of [estimation technique, target model] pairs' efficiency and subsequent selection. The vertical axis represents the estimation effort while the horizontal one represents the average estimation % error. The chart shows only simulation- and learning-based methods as the first method is widely used and the second method is the most recent.

This graphical representation reveals the distribution of the cluster-based power estimation methodologies applied to specific target models v/s their respective estimation efforts. The following analysis was noted:

Simulation-based techniques: The simulation-based power consumption estimation techniques are best suitable for µP/CISA/(MP)SoC target models with estimation error <5% in specific cases, especially when software applications targeting (soft- and hard-core) µPs are involved. This is usually done with moderate efforts. The component/circuit target models come next in the classification under the same estimation methods with also moderate efforts, leaving behind the white/black-box IP models, unless considerable efforts were exerted. The major advantages of such a technique are: a) its generic nature and b) its relatively high precision. High memory requirements and slow execution time remain major drawbacks especially when fast prototyping is required and when resources are limited.

Learning-based techniques: Also known as pattern recognition, Machine Learning-based power consumption estimation techniques, when applied to white/black-box IP and µP/CISA/(MP)SoC models even with moderate efforts, provide higher efficiency to when applied to component/circuit models. However, considerable estimation efforts, definitely push the efficiency higher

when dealing with component/circuit target models especially when artificial neural networks are adopted. The major advantages of such techniques can be summarized as follows: 1) easy identification of trends and patterns, 2) no human intervention is needed, 3) ability to handle multi-dimensional and multi-variety data, and most importantly 4) continuous improvements coupled with wider range of target models. Time/resources requirements, errors susceptibility and the need of prior data acquisition/collection can be considered as main hurdles.

Statistical-based techniques: even though these methods have not been used in recent work, they made it to the classification. Little to moderate estimation efforts lead to estimation errors around 3.5% for the processor-based targets and 12.5% for IP targets. These techniques are usually simple to apply and easy to interpret their outputs. However, the estimation ability may decrease drastically due to data sparsity and modeling overfitting.

Measurement-based techniques: Even though tightly coupled with instruments' fidelity, probing level and sensing accuracy, the measurement-based methods are considered as reliable alternatives in estimating power consumption. Due to lack in estimation accuracy results exposure, these techniques did not make it to the global clusters chart. As for the estimation efforts, little to moderate levels were applied. It's important to mention that measurement results are often used in characterization for later data processing and subsequent power estimation using various methodologies but mainly machine learning techniques. Measurement-based techniques are non-invasive and relatively easy to handle but the dependency on the instrumentation (that can be sometimes bulky) and its accuracy might affect the fidelity of the results and thus the estimation % error. Moreover, these techniques often target a particular IP or circuit and lacks of genericity.

3.5 Conclusion

In this overview chapter many power consumption estimation-related topics were covered. High-Level power estimation techniques and target models were presented as the main concern and purpose of this chapter. The covered literature was extensive with a considerable variety of evaluation methods and contribution levels applied to many sets of target models.

Knowing that, the papers' classification had to be done in a tabular format (Table 3.1) where some of the metrics were quantitative and some others qualitative. A global cluster-based chart (Figure 3.2) representing the power estimation techniques and their respective target models v/s the estimation efforts has been generated. The purpose of that representation was to have a general and wide understanding of which estimation technique applied to which target model results in a better estimation accuracy while keeping an eye on required estimation efforts. The ultimate purpose of this work was to provide guidance for designers (dealing with power estimation and modeling methods) in terms of assessment, comparison and selection of the appropriate power estimation techniques applied to target models based on performance, accuracy and efforts.

As a future sight and given the extensiveness of current applications and continuous enhancements coupled with hardware-backed computational power, machine learning techniques might be nominated and foreseen as the most effective and global approaches for power consumption modeling and subsequent power estimation. That being said, in this work we tackle the FPGA power consumption modeling and estimation based on supervised machine learning. In chapter 4, we present a high-level learning based FPGA IP power consumption modeling and estimation methodology for both the offline and the online domains.

Chapter 4

FPGA Power Modeling and Estimation Methodology

4.1 Introduction

FPGA power modeling and estimation refer to the process of predicting the power consumption of an FPGA design before it is implemented in hardware. In modern systems, power estimation can also be implemented in real-time. FPGA power estimation is an essential aspect of the FPGA design process, as it helps to optimize the design to minimize power consumption. FPGA power modeling involves creating a mathematical model of the power consumption of an FPGA design based on the design's circuit topology, input data patterns, and other parameters. The model can be derived using various techniques, including analytical methods, simulation, and measurement.

Today, modeling FPGA power consumption may be used in two scenarios. First, it is useful to perform power-oriented design space exploration at design time. The elaboration of power-efficient designs usually requires multiple iterations of power estimation and design refinement steps, which leads to a long design time and low productivity. In this scenario, having accurate and efficient power models can help to rapidly explore design choices and perform high-level simulations. Second, power modeling may also be used at run-time. In most systems today, it is required to have efficient power monitoring and management. This can be highly useful in taking decisions regarding DVFS or tasks' scheduling in CPU-FPGAs systems. In such cases, a power model has to be implemented in the device itself and run either under software control on a processor or in a dedicated hardware component. Our main target is to have a "simple" yet efficient model that is capable of extracting basic features of input signals to obtain an accurate value of the circuit consumed power while taking into consideration both speed and accuracy.

This chapter is organized as follows: in section 4.2 we present our proposed power consumption modeling and estimation of FPGA IPs. We elaborate on

IP decomposition and on the methodology of estimating power in both offline and online domains. Section 4.3 reveals the global methodology flow where we detail various options and possible paths. Finally we conclude in section 4.4.

4.2 Power Estimation of FPGA IPs

After investigating recent related works, concerning power modeling, we found that in [55] or [15] the approach consists in monitoring influential signals within specific modules. Power consumption is then measured and a linear power model is built and updated online. The main drawbacks of these previous works are the lack of accuracy of the proposed models which rely on simple linear mathematical models. To counteract this issue, authors in [81] propose to collect a pertinent set of signal activities through simulation and construct machine-learning models based on decision trees, ensemble models, and neural networks. This makes it possible to take into consideration the non-linearity of the power behavior. Nevertheless, this work only targets ASICs models at this point and cannot be used for FPGA devices.

Authors in [40] extract activity features during C-level program execution, and propose machine learning models to learn power characteristics of an FPGA implementation. The power model directly makes use of transaction-level I/O activity and control information for fast estimation. Although very accurate for a high-level approach, power data used for training only derive from gate-level simulations, which is far from a real scenario of execution.

Authors in [59], proposed *NeuPow*, a power estimation methodology based on neural networks that model the power and behavior of arithmetic components in both ASIC and FPGA circuits. The proposed method is achieved by propagating predictors between the connected neural models in order to estimate dynamic power consumption of individual components. Although the approach seems very promising, neural networks have only been trained on data coming from low-level simulations, which lacks of accuracy and genericity.

In this work, in comparison to most approaches, we rely on real data obtained from hardware measurements and software algorithms in order to efficiently build training sets for machine learning. We estimate power consumption of FPGA IP circuits based on the coupling between the control signals (offline), or operation modes (online), and the data path activity. Having the most realistic data leads to a more realistic power model and subsequently to a more accurate power estimation. That been said, as a first step, we investigate FPGA IPs as targets for the power modeling and subsequent estimation.

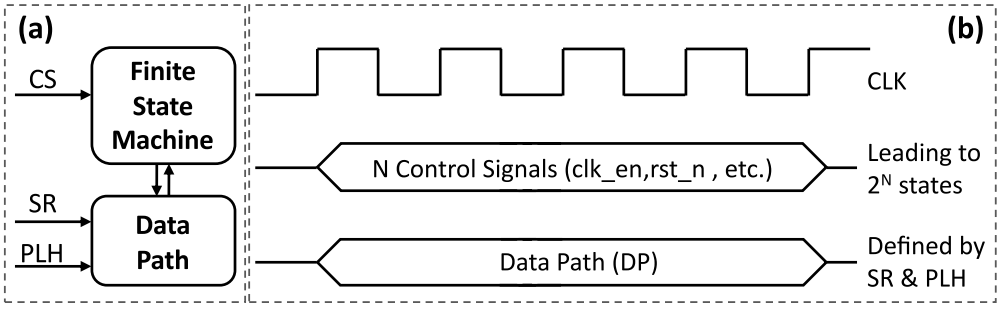

FIGURE 4.1: FPGA IP decomposition (FSM and DP) (a) and its input signals (b).

4.2.1 IP decomposition

IP blocks are pre-designed and pre-verified digital circuit designs that can be integrated into larger designs to save time and effort. FPGA IP characterization involves assessing the electrical behavior of the IP block when implemented on an FPGA, such as its power consumption, timing characteristics, and other key metrics. This process helps ensuring that the IP block operates correctly and meets its performance specifications in the target FPGA technology.

Any given FPGA IP circuit, as shown in Figure 4.1 (a), can be represented as a Finite State Machine (FSM) and a Data Path (DP). The combination of the Control Signals (CS) and the data path input activity results in multiple functional and power states. In this model, the control signals are directly connected to the FSM and the input signals (stimuli), feeding the DP, are characterized by their activity parameters i.e Switching Rate (SR) and Percentage Level High (PLH). The SR represents the ratio of transitions in a given fixed-size bit sequence whereas PLH is the percentage of logic 1 bits in the same sequence. In this work, we present an IP power consumption estimation model based on machine learning in which training data sets are automatically collected from real measurements. Figure 4.1 (b) reveals the inputs of any given IP. At high-level, we characterize specific IPs, record the power consumption estimation absolute percentage error and compare measured v/s estimated power consumption.

4.2.2 Methodology

The proposed approach consists in estimating the power consumption of a given FPGA IP after being decomposed into an FSM and a DP. For the offline domain, we estimate per-state power consumption of IP circuits based on the

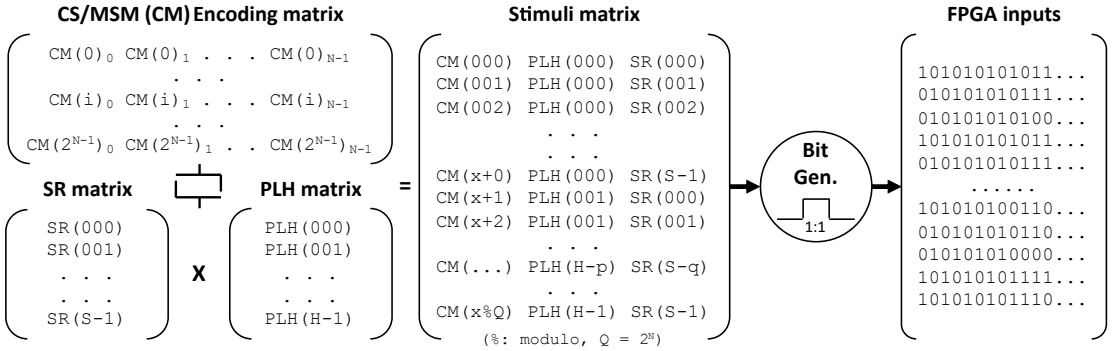

FIGURE 4.2: Stimuli generation matrices along with the actual FPGA input bits.

coupling between the state machine control signals and the data path activity. Both the control signals and the input activity are extracted from a testbench during the simulation phase, and subsequently applied to a previously stored power model. For the online domain, we estimate power consumption of IP's modes of operation, in realtime, based on the coupling between the most power-influential modes and the data path activity. Operating modes are specific functional behaviors of a given IP (for ex: idle mode, half-duplex mode, full-duplex mode, loopback mode, burst mode, etc.), out of which we select the most power-hungry subset. In this case the generated power estimator is implemented inside the same FPGA.

In summary, the proposed methodology consists of four sequential steps:

1. Stimuli signals generation using dedicated, in-house implemented algorithms.

2. IP characterization using a well-defined high fidelity platform fed by the previously generated stimuli.

3. Build of the training data sets using the collected power information and subsequently build of the power model.

4. Storage of the compiled neural network in a database for the offline domain, or, implementation of the neural network inside the same FPGA target for the online domain.

Stimuli generation

Regarding the parametric stimuli generation as shown in Figure 4.2, two sets of PLH and SR of H and S values are respectively used to generate $M = H \times S$ distinct combinations of [PLH, SR] pairs. The aforementioned combinations are coupled with N encoding bits representing either Control Signals (CS) for

offline domain or Operating Modes (OM) for the online alternative. These CS/OM bits, denoted as CM, lead to 2^N power consumption scenarios. Each permutation of the CM bits is appended to I pairs of [PLH, SR], subsequently generating one stimuli matrix. Each pair of [PLH, SR] generates 1 fixed-length sequence of bits that will be eventually feeding a given IP's data path input. The bit sequence generator in context ensures the [PLH, SR] pairs' compatibility and fulfills the stimuli signal's requirements while achieving a high degree of entropy in the generated bits (adequate SR distribution and minimal bit-clustering). The detailed stimuli and bit sequence generation is presented in chapter 6.

IP characterization

FPGA IP characterization reveals the distinctive nature of a designated digital circuit, specifically, its power consumption information. For that purpose, we have proposed an, in-house, FPGA-based Automated and Centralized Data Generation and hybrid Acquisition System (ACDGAS) combining the features of three instruments: a sampling oscilloscope with analog inputs, a logic analyzer and a bit-pattern generator [67].

The platform's main role is to apply stimuli signals at the IP inputs and synchronously collect aligned power consumption samples under software control. The proposed layered software architecture delivers a fully automated process for IP characterization. It provides a solid and synchronized interaction between the various system modules such as: graphical user interface, stimuli construction, CS/OM (CM) coupling, bit sequence generation, measurement platform interface, training data construction, power modeling and evaluation.

High Speed Digital I/Os feed the DUT-FPGA (configured for 1 IP only) as stimuli input signals and the power consumption measurement is sampled via a high speed parallel differential Analog to Digital Converter (ADC) through a precision shunt resistor R_S on the FPGA core voltage. The FPGA IP's characterization is thoroughly detailed in chapter 5.

Training data sets generation

For many machine learning techniques, especially the ones related to supervised methods, the construction of the training data highly affects the quality and accuracy of the derived model [48]. In this work, the collected data sets are derived from two different sources: the stimuli generation algorithms and a hardware data acquisition system providing real power consumption values

collected after applying the generated stimuli on a given FPGA IP circuit. The power model generation including the training data sets generation will be thoroughly detailed in chapter 6.

4.2.3 Power Estimation Applications

In general, power estimation is obtained by evaluating the signals' activity of a given circuit in a scenario of execution that runs within a certain time frame. In this work we investigate power consumption modeling and estimation in both offline and online domains.

Offline Estimation

Offline FPGA IP's power consumption estimation using machine learning and measurement refers to the process of predicting the power consumption of an FPGA based on both machine learning algorithms and measured data from the FPGA itself. In this process, machine learning models are trained using data from previously measured power consumption values for the FPGA loaded with specific IPs. These models then use the trained algorithms to predict the power consumption of the FPGA for new scenarios including states and input activity that have not been previously measured. This process is called "offline" because it does not involve the actual operation of the FPGA. Instead, it uses models and generated testbench data in order to estimate the power consumption of the FPGA IP based on its design and the task it will be performing.

The proposed machine learning-based power model aims at estimating the power consumption P_E of an FPGA IP based on the control signals and on the features of its inputs i.e their PLH and SR at early design stages. Figure 4.3 illustrates the usage of the power model at design time. Test-bench (TB) data, initially generated for the functional simulation, are also used to extract CS, SR and PLH of all inputs. The next step is to apply the extracted values to a power model built using the proposed work and retrieved from a previously constructed database (DB). This latter contains a collection of power models each for a previously characterized IP.

Online Estimation

Online power monitoring of a digital circuit is the process of periodically collecting its energy usage for subsequent optimization using well-known mechanisms such as DVFS. The proposed online power monitoring methodology aims at estimating a given FPGA IP power consumption in-situ and in real-time. It is based on machine learning and precisely on supervised Artificial

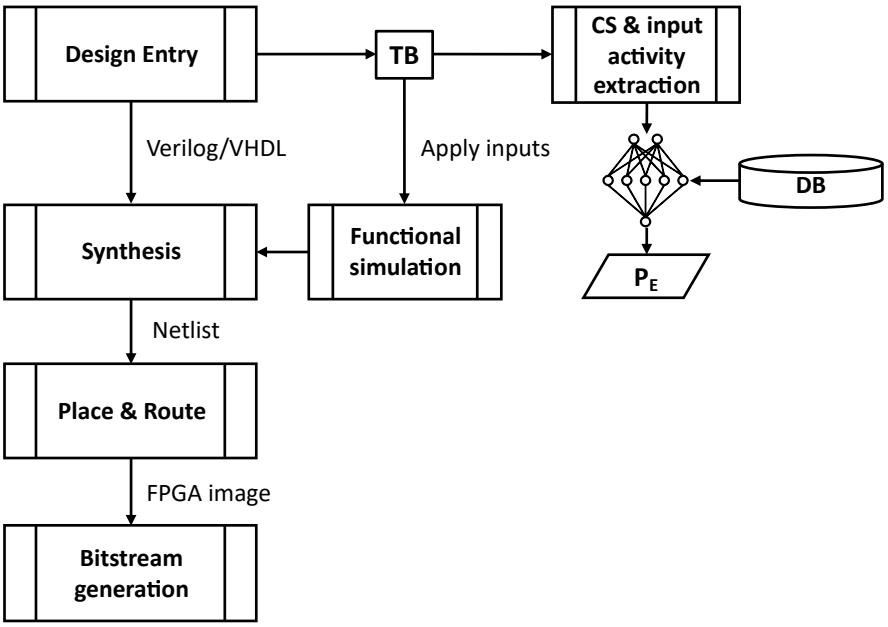

FIGURE 4.3: Early power estimation flow (at design time).

Neural Networks (ANN) [23]. Online power monitoring is becoming indispensable for controlling the power consumption of large digital circuit systems at runtime [68].

Runtime adaptive systems determine dynamically and autonomously the optimal operating points for power consumption. Such systems take application-specific requirements into account and specifically adapt the workload. Traditionally, current sensors (either built-in or external) have been used to periodically, under software control, collect power information to be transferred to variable voltage regulators and/or configurable Phase-Locked Loops (PLLs) or simply Voltage-Controlled Oscillators (VCOs). These latter, under the DVFS mechanism, regularly tune the core supply voltage and/or the input clock frequency of the device under optimization. Appropriate design of DVFS-based applications can result in up to 65% improvement in energy consumption [4]. The main issue to tackle remains in the communication overhead that can have undesirable effects on a running application on FPGAs; notably the latency between the sensors and the DVFS, and also between the DVFS from one side and the variable voltage regulators and/or the tunable frequency devices from the other.

Some works aim at supporting emerging power management techniques, for example fine grained DVFS. In [45], a dedicated hardware circuit is proposed to obtain internal signal activities during runtime while predicting power consumption on-the-fly. The authors claim that dynamic power may be estimated within an error margin of 1.90% compared with commercial gate-level power estimation tool. The major limitation though is that the model is

trained on simulated data from the Vivado power analyzer, which does not take into consideration the real conditions of execution.

At RTL, most recent works that study high-level power modeling of FPGAs usually rely on linear regression methods. For example, in [55] or [15] the approach consists in monitoring influential signals within specific modules. Power consumption is then measured and a linear power model is built and updated online. The main drawbacks of these previous works is the lack of accuracy of the proposed models which rely on simple linear mathematical models.

In this work, compared to most related methods and in order to build robust training data sets for the learning-based estimation, we rely on real power values obtained from a physical acquisition system, under software control. Subsequently, we estimate, in real-time, FPGA IP power consumption, based on the coupling of the IP's Most Significant Modes (MSM) of operation and the activity of its data path inputs.

Figure 4.4 provides a high-level overview of the proposed power monitoring and management mechanism. The power monitor in context, considered as our main concern, is divided into two blocks. The first block extracts: 1) the input activity of the IP's data path in terms of the switching rate and the percentage of logic "1" occurrences and 2) the most energy-significant modes of operation (MSM) that can be derived from the control signals of the IP's state machine or even, in some cases, can be explicitly accessible. The second block is the power estimator itself having the extracted input activity parameters along with the most significant modes of operation as inputs. This latter consists of the implementation of an artificial neural network inside the FPGA. Both the implementation details and the neural network architecture are discussed in later sections in chapter 6.

Figure 4.4 also reveals the existing latencies at various locations of the proposed model. The latency between the power estimator and the DVFS mechanism depends on the number of cycles of the power estimator itself from the moment it captures the inputs until it generates an output. This work aims at keeping this specific latency at the very minimum. The latency between the DVFS and the tunable devices (voltage and/or frequency) is highly dependent on the Management Interface (MIF). Standard communication latency is observed when dealing with PMBus/I^2C protocols, and minimal latency is produced when dealing with fast Digital to Analog Converters (DACs), as data are directly driven from discrete fast outputs. Note that this specific type of latency is out of the scope of this work. As for the DVFS system itself, it may be decomposed into a set of comparators acting on pre-defined power consumption

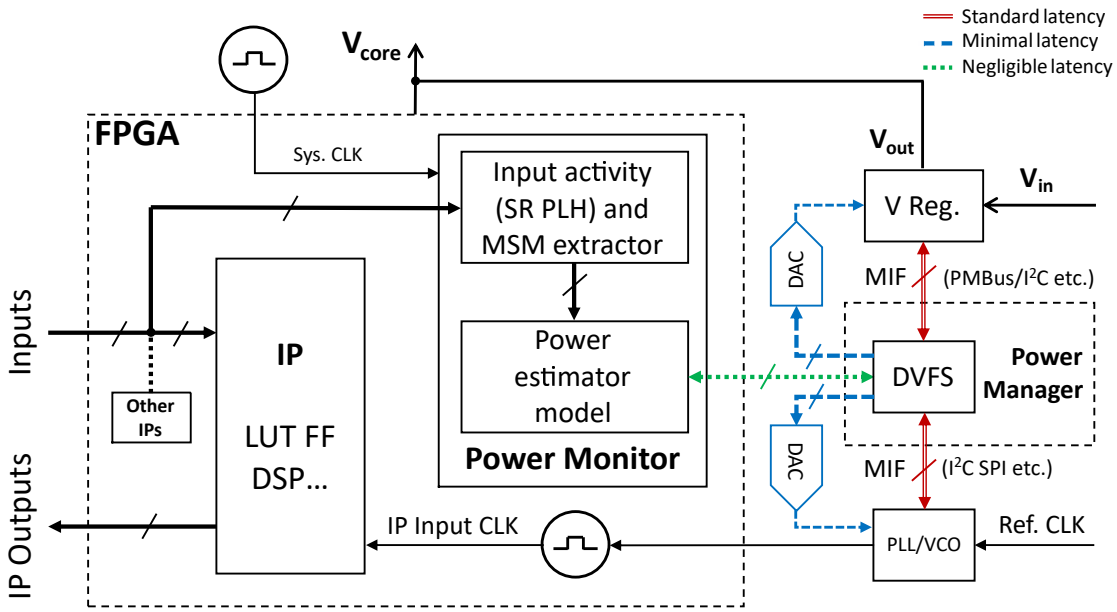

FIGURE 4.4: Power monitoring and management mechanism along with latency options.

threshold values and dynamically tuning the corresponding parameters within pre-defined operating limits. That said, the DVFS' operation does not add significant latencies to the power monitoring and management loop (power estimator $\rightarrow$ DVFS $\rightarrow$ tunable devices and again.. $\hookleftarrow$). The proposed methodology consists of three sequential steps:

1. IP characterization using a well-defined high fidelity platform.
2. Build of the training data sets using the collected power information and subsequently build of the power model.
3. Implementation of the neural network inside the same FPGA target.

4.3 Methodology flow

Ultimately, combining both power estimation domains could create a more comprehensive approach in order to present the proposed methodology. Figure 4.5 reveals the global power modeling and estimation flow for both offline and online domains. For both alternatives, some of the procedures are common and others differ; however, the aim is to always get the estimated power P_E. From a user perspective, a typical design flow starts by the Hardware Descriptive Language (HDL) entry, testbench generation mainly for the functional simulation followed by the synthesis, place and route, and bitstream generation for subsequent FPGA image loading. The proposed methodology is initiated (as early as

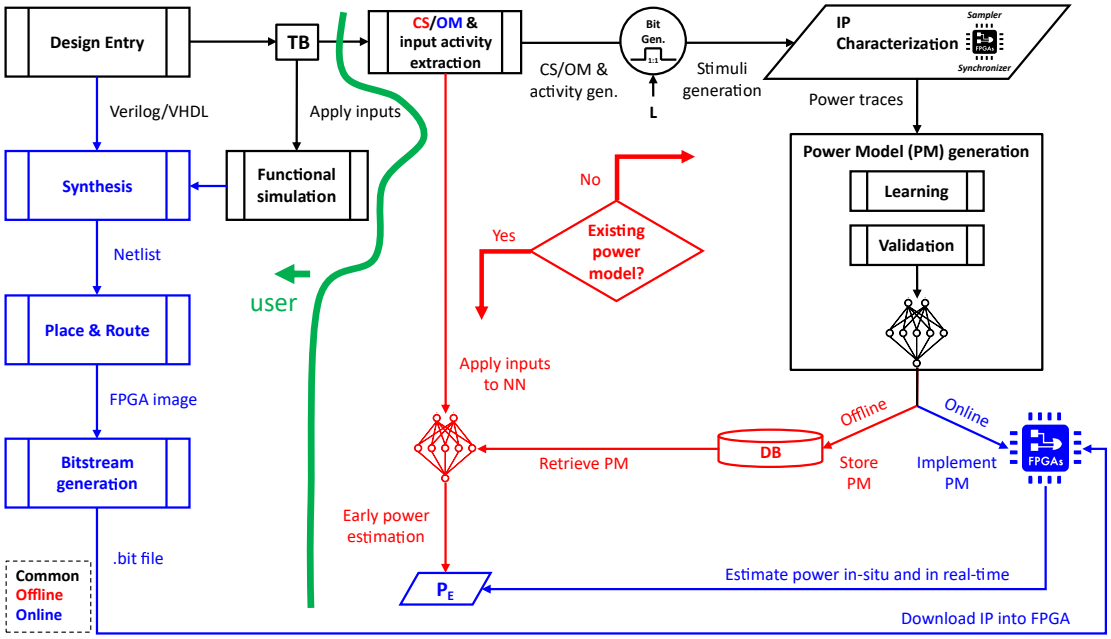

FIGURE 4.5: Global power modeling and estimation flow covering both offline and online domains.

possible) at the testbench creation step. At this level, the IP's data path inputs' activity is extracted (PLH and SR pairs), and the FSM control signals (offline) or the most significant operation modes (online) are enumerated. At this points, two options are possible: is the IP in context has been previously characterized and its power model generated? If the answer is yes then the power model is retrieved from a previously compiled database, the neural network inputs are applied and subsequently the power estimation is delivered.

Otherwise, the power model generation procedure is initiated starting by the stimuli generation based on the input activity coupled with either the FSM control signals or the most significant modes of operation. The IP characterization is then triggered leading to the generation of the training data sets for the supervised machine learning procedure. Both the learning and the validation processes result in the compilation of the Power Model (PM). At this point, two options are available: 1) for the offline domain, the generated power model is stored in a database to be later on retrieved when estimating the power consumption of that specific IP (under the same characterization conditions) early at design time. 2) for the online domain, the generated power model is implemented inside the same FPGA for subsequent in-situ and realtime power reporting.

In summary, whether offline or online, the power model generation procedure remains the same, leading to the power consumption estimation for either storage or deployment.

4.4 Conclusion

FPGA power modeling and estimation are crucial for optimizing the power consumption of an FPGA design. By predicting the power consumption of the circuit early in the design process, designers can, offline, identify and eliminate power-hungry components or circuits, optimize the design's clock frequency, and minimize power consumption by selecting appropriate input data patterns. This optimization can lead to significant power savings and improved performance for FPGA-based systems. On another hand, online power monitoring and subsequent management is becoming a hot topic since it provides on-the-fly energy optimization by adjusting the IP's voltage and/or frequency.

In this chapter we have presented an FPGA IP power consumption estimation methodology covering both offline and online domains. For the offline option, we estimate power by just providing the control signals of the IP's FSM and the activity characteristics of the data path inputs. The subsequently generated power models are store in databases. For the online counterpart, power estimation is done in real-time by providing the IP's operation modes and its inputs characteristics. The power estimator and the target IP coexist in the same FPGA. Finally we have combined both offline and online in a single methodology flow diagram. In the following chapter we detail the FPGA IP characterization procedure, revealing the proposed measurement platform and the corresponding results.

Chapter 5

FPGA IP Characterization

5.1 Introduction

FPGA Intellectual Property (IP) characterization refers to the process of measuring and analyzing the performance characteristics of an IP block that is intended to be implemented on a specific FPGA. FPGA IP characterization reveals the distinctive nature of a given target digital circuit and highlights its special features such as resources and most importantly its power consumption information (a special interest in our study). Here we should point out that, as shown in Figure 5.1 (right-hand side), we decompose an FPGA IP into a Finite State Machine (FSM) fed by a certain number of Control Signals (CS), and a Data Path (DP) fed by the actual inputs toggling at specific rates.

The IP's power consumption, during well-defined states or operating modes, is highly affected by its input switching activity mainly the Switching Rate (SR) and the Percentage Level High (PLH). The SR represents the ratio of transitions in a given bit sequence, whereas the PLH is the percentage of logic "1" bits in the same sequence. Operating modes are specific functional behaviors of a given IP (for ex: idle mode, half-duplex mode, full-duplex mode, loopback mode, burst mode, etc.), out of which we select the most power-hungry subset denoted as Most Significant Modes (MSM). These operating modes are in a close correlation with the control signals feeding the IP's FSM, or in some other cases, they are explicitly exposed. In Figure 5.1 (right-hand side) the relationship between the control signals and the operation modes is represented by a decoder (X/Y). In order to collect precise power information, a reliable measurement platform is required. For that purpose, we propose an FPGA-based Automated and Centralized Data Generation and hybrid Acquisition System (ACDGAS) combining the features of three instruments: a sampling oscilloscope with analog inputs, a logic analyzer and a bit-pattern generator [67].

The question that may arise at this point is "why designing and implementing an in-house instrument rather than getting an off-the-shelf alternative?" The

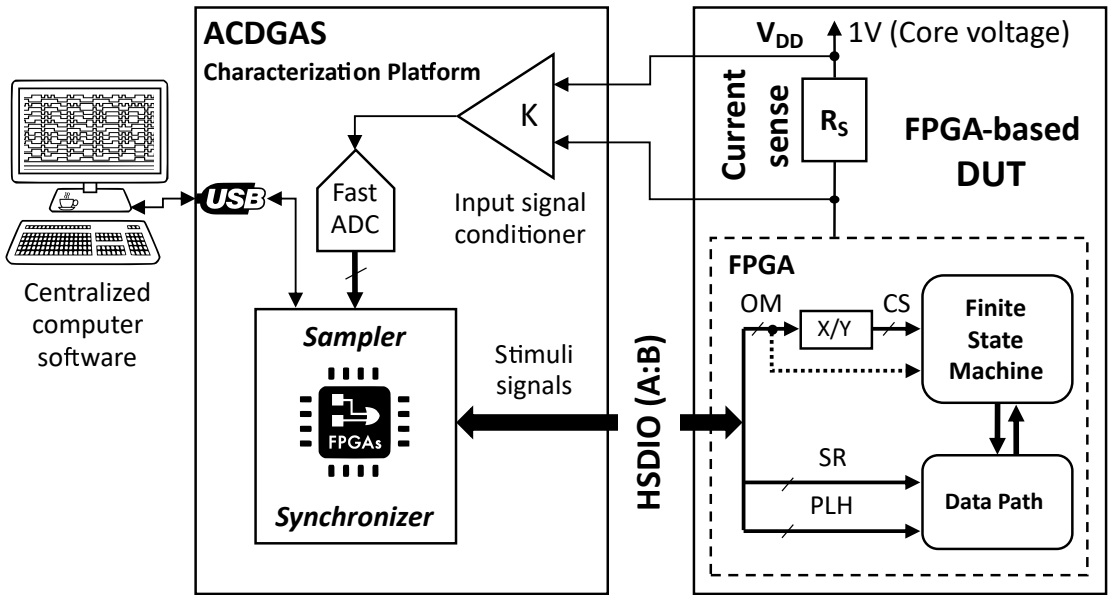

FIGURE 5.1: Measurement and characterization system hardware setup along with the IP under test.

answer is simple "such an instrument is not readily available in the market given its customizable features and behavior". The FPGA IP characterization system that meets our needs has to include a bit pattern generator and a hybrid data acquisition mechanism to collect both digital and analog samples. Most importantly, both the generated and acquired data have to be synchronized, and most critically they need to be aligned. Having multiple instruments hooked together in order to satisfy our needs is, first of all, cumbersome and expensive. Moreover and above all, assuring synchronization and alignment is not guaranteed, and also, the integration of a centralized control software is almost impossible.

This chapter focuses on the design, the methodology of generation and acquisition, the hardware/software interface, the end-usage and the possible applications of the device. The platform's main role is to apply stimuli signals at the IP inputs and synchronously collect aligned power consumption samples under software control. The proposed layered software architecture delivers a fully automated process for IP characterization. It provides a solid and synchronized interaction between the various system modules such as: graphical user interface, stimuli construction, CS/MSM coupling, bit sequence generation, measurement platform interface, training data construction, power modeling and evaluation.

This chapter is organized as follows: in section 5.2 we present the automated and centralized data generation and acquisition platform including both hardware and software aspects. The hardware sections encapsulate the digital

and the analog parts. The generation and acquisition methodology is detailed along with experimental results and comparison with recent related work. In section 5.3 we demonstrate the application of the aforementioned platform on series of FPGA measurements. Finally we conclude in section 5.4.

5.2 Automated Data Generation and Acquisition System

Automated data generation and acquisition refer to the process of using combined software and hardware tools to automatically collect, create, and process data without human intervention. This process involves using various techniques to gather, analyze, and generate data that can be used for different purposes, such as research, analysis, and decision-making.

Data Generation and Acquisition (DGA) has become a major requirement for data processing and subsequent information analysis, especially in data science, big data, and pattern recognition (which is a cornerstone of machine learning techniques). A synchronized hybrid data generation and acquisition system has major advantages over traditional counterparts given the mixed-mode nature of its input signals (digital and analog) being aligned with its output signals. The proposed DGA system, in addition to being hybrid and synchronous, is a single-board programmable, compact and fully automated open-source hardware/software instrumentation device. As shown in Figure 5.1, High Speed Digital I/Os (HSDIO) are feeding the DUT-FPGA (configured for 1 IP only) as N-bit input signals and the power consumption measurement is sampled via a high speed parallel differential Analog to Digital Converter (ADC) through a precision shunt resistor R_S on the FPGA core voltage (1V).

Here we list few recent related work with a brief description revealing features and limitations. In [34], a low-cost USB data acquisition hardware is described based on the ATmega32 microcontroller. The system acquires heater temperature from a 1-wire sensor and various voltage variations through relatively low-speed ADC channels. The system in [21] was developed for the purpose of evaluating DACs and ADCs. An automated system has been developed for multi-frequency dynamic tests. Multiple instruments and tools were used for that purpose (NI-PXI, Tektronix, LabVIEW). A smart recording power analyzer prototype using LabVIEW and a low-cost DAQ are presented in [12]. The system is capable of simultaneously monitoring and recording numerical data and low frequency waveforms in both normal and faulty conditions. It proposes the usage of a low-cost DAQ device and the commercial LabVIEW for a smart renewable monitoring system. In [18], a virtual instrumentation

software has been applied to integrated circuit testing procedure based on LabVIEW. The system provides the possibility of software-controlled automated test system development for IC parameters. Most of the related work relies on multiple instruments and software tools that may imply complexity, bulkiness and unwanted additional cost. An application of high accuracy DAQ in power plants is presented in [77]. The system is based on an 8051 microcontroller connected to an external ADC via Serial Peripheral Interface (SPI). The PC interface is implemented over RS232, a relatively low speed connection.

The core of the DGA system we are presenting is based on an FPGA (Intel Cyclone V) interfaced to the outside world via either full speed USB through an on-board microprocessor and/or via Fast Ethernet (FE). The interconnection between the FPGA and the microprocessor is done using an in-house designed high speed parallel Microprocessor Interface (MPIF) described throughout the chapter. The choice of an FPGA at the heart of the system was made given its high speed, its parallel processing nature and most importantly its reconfigurability. The proposed system is a 20 MSpS synchronous hybrid DGA capable of simultaneously generating and sampling High Speed Digital Inputs and Outputs (HSDIO) accurately aligned with analog input samples.

5.2.1 System Description

In this section various system modules is detailed covering the single board hardware, firmware and software. In hardware subsection, both the digital and the analog parts is described. The digital part covers the FPGA high level design, the memory organization and the control and status mechanism. The analog part covers the dual sampling ports' hardware. In the Software subsection, we briefly describe the firmware implementation along with PC interfacing. Access to the project's repository is granted through a *copyleft CC-BY-SA 4.0* license following an email request.

Hardware

The high-level hardware block diagram is shown in Figure 5.2. The single board instrument interfaces to an external computer via two options: USB and Fast Ethernet (FE). In the following subsections, we describe both the digital and the analog parts of the hardware.

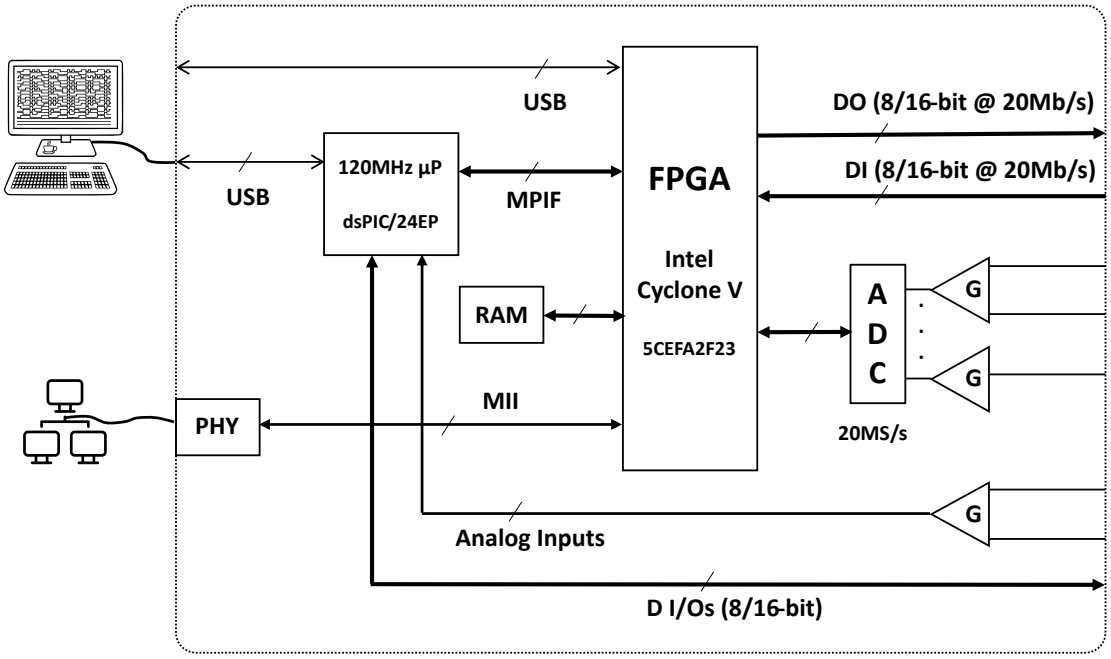

FIGURE 5.2: System block diagram and interface.

Digital: The main role of the FPGA is to synchronize the generation of the High-Speed Digital Outputs (HSDO) with the acquisition of both the High-Speed Digital Inputs (HSDI) and the analog inputs at a (programmable) maximum sampling rate of 20 MSpS. The outcome is a precisely aligned hybrid data. The FPGA's system clock is set to 120 MHz. The external processor is a Microchip dsPIC/24EP 120 MHz controller acting as a gateway between the FPGA and the host PC. Whether with a softcore or an external processor, the HSDIO control mechanism remains the same, i.e., via a well-defined set of registers shown as a logic block in figure 5.3 (Regs). The access to these registers can be done either via MPIF (in case of an external processor) or via Avalon interface (in case of a softcore processor).

Figure 5.3 shows the internal FPGA logic. Two options are valid: adding a System-on-a-Programmable-Chip (SoPC) or enabling the external on-board processor. In both cases the HSDIO control mechanism is driven through the same register map. Two options are also available for the memory management and segmentation, internal and external RAM. Three FIFOs (caches) are allocated: one "write" buffer and two "read" buffers for the HSDO and the HSDI samples concatenated with the analog samples respectively. The HSDIO/Sampling controller is the main IP responsible for: 1) reading and transmitting the previously downloaded HSDO samples, and simultaneously 2) capturing both the HSDI samples along with the analog samples.

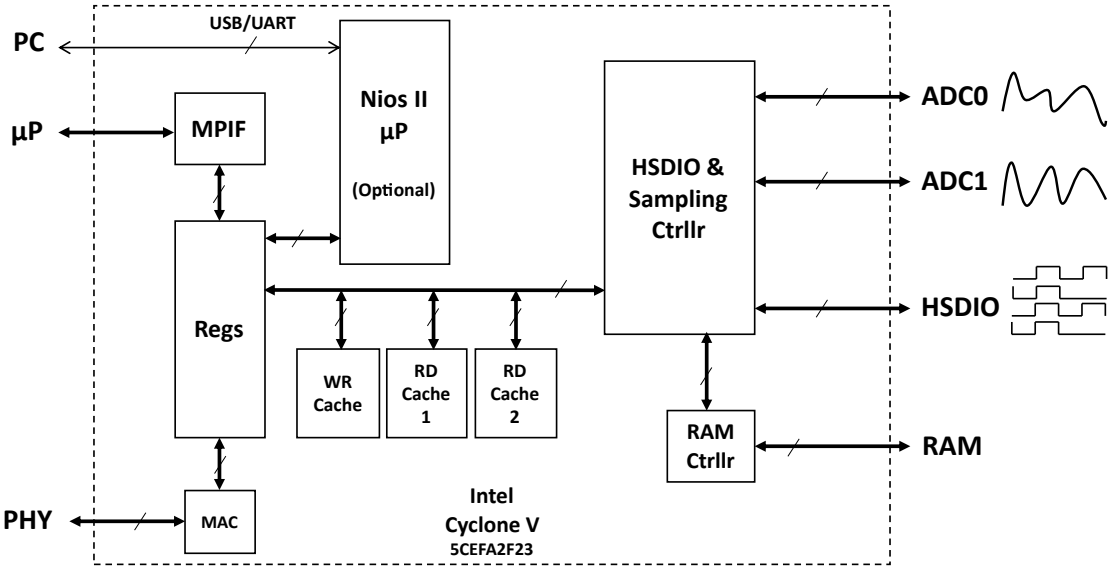

FIGURE 5.3: FPGA internal logic and external connections.

The interface to the outside world can be fulfilled in two different (but coexistent) paths: either via full speed (12 MbpS) USB in the case where the external processor is used or via fast Ethernet through the on-chip MAC IP, driven by the embedded processor in the case of an SoC.

The interconnection between the FPGA and the external processor is implemented via a dedicated 16-bit parallel interface. The MPIF allows device configuration (including samples download/upload) and status information collection through a set of registers. Two main operations "Read" and "Write" are used to access the previously mentioned registers.

Two options are available for RAM selection: internal and external. For simplicity FPGA internal RAM has been used given its parallel access nature and that the available memory size is enough (following our requirements) in order to perform the required samples "write" and "read" operations. All available RAM is equally divided (by three) between one output FIFO and two input FIFOs. The output buffer entries consist of the 16-bit HSDO samples and the two identical input buffers entries consist of the HSDI (four bits) concatenated with the ADC sample (10 bits). The available RAM is divided into two sections: a) section to store output samples of size X (X follows the memory type and contains N blocks of samples) and b) section to store input samples of size 2X (2 analog channels, containing 2N blocks of samples). All sections have the same number of memory blocks N and the block size is dependent on the memory type (internal or external). Output and input samples are precisely aligned over time. Writing samples to RAM is done in equally-sized blocks (N), each block is equal to the cache memory size. Reading samples is also done in equally-sized blocks (N).

Analog: The system is equipped with two independent 730 MHz bandwidth, 20 MSpS analog channels. The first channel (ADC[0]) is a differential 100 mV peak-to-peak input dedicated for current sensing. The second channel (ADC[1]) is a general purpose 3.3V peak-to-peak input. The internal ADC input range is between 1.5V and 3.5V (2V peak-to-peak). Therefore, input signals conditioning is required.

Two identical parallel analog-to-digital converters were used, ADC10321 by Texas Instruments for the analog acquisition. The ADC10321 is a low-power, low-cost, high performance CMOS pipelined analog-to-digital converter that digitizes signals to 10 bits resolution at sampling rates up to 25 MSpS (absolute maximum) while consuming a typical 98 mW from a single 5V supply. The ADC10321 maintains excellent dynamic performance for input signals up to half the clock frequency. The use of an internal sample-and-hold amplifier enables sustained dynamic performance for signals of input frequency beyond the clock rate, lowers the converter's input capacitance and reduces the number of external components.

Software

The system's software is broken down into two layers: the firmware that runs in the hardware itself and the PC graphical user interface software that orchestrates all the system's functionalities.

Firmware: Two options are available for the firmware implementation: a) using the external processor via the MPIF from one side and USB from the other or 2) using the FPGA embedded processor via Ethernet interface. In both cases, the configuration and control of the presented system is done through the register map.

PC software: The software interfaces to the presented instrument via either USB or FE, thus, two flavors are possible. The 1^{st} flavor has been adopted initially for its simplicity and its fast implementation. This option uses the on-board processor for the FS USB interface. The Application Programming Interface (API), written in VS.NET, provides portability and code integration. A Graphical User Interface (GUI) has been built based on the provided instrument's API. The PC software also supports batch mode in order to automate lengthy procedures.

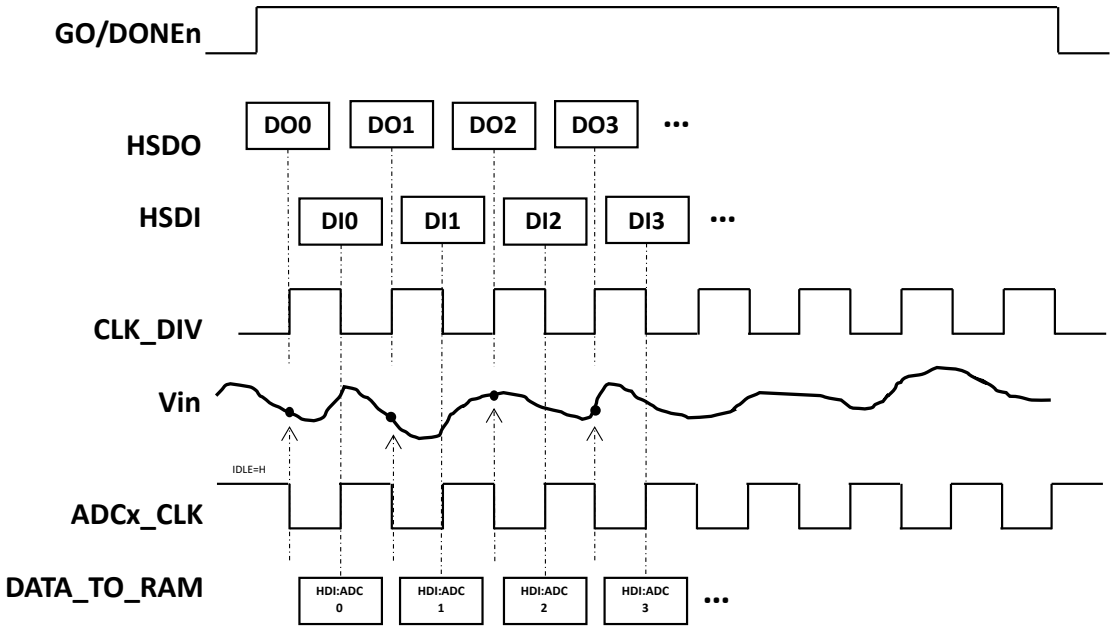

FIGURE 5.4: Data synchronization and alignment.

5.2.2 Generation and Acquisition Methodology

Aligning the digital outgoing signals (HSDO) with the digital incoming signals (HSDI) and the analog incoming signals is the most important feature of the presented system. Figure 5.4 illustrates both data synchronization and alignment. After downloading samples into hardware, the process is initiated by setting GO/DONEn bit (in a specific register) to 1. The HSDO bits are sampled out at the rising edge of the programmable synchronization clock CLK_DIV. The HSDI bits are sampled in at the falling edge of the same clock to ensure data setup time and thus input data validity. The ADC clock ADCx_CLK is the complement of CLK_DIV since the analog signal is sampled in at the falling edge and stored in RAM at the rising edge of the same clock to ensure analog input data validity. When all bits are sampled out and in, the GO/DONEn bit is automatically cleared declaring the end of the burst cycle. The sampling clock is derived from the 120 MHz system clock and configured through a prescalar register. For the ADC timing, data for any given sample is available by the ADC's pipeline delay right after that sample is taken. New data is available at every clock cycle, but the data lags the conversion by the the same pipeline delay. Data adjustment is done automatically by API.

The above mechanism is orchestrated by the FPGA including clocking, synchronization, alignment and status reporting; all under API control via a specific set of (control/status) registers.

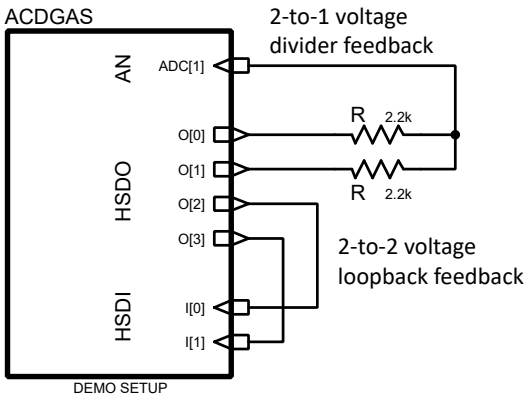

FIGURE 5.5: Test scenario - hardware setup.

5.2.3 Experimental Results and comparison

For the experimental results, a test scenario has been implemented in order to, at the same time, inspect and validate the synchronization, the data alignment and the accuracy of the proposed system. For that purpose and as shown in figure 5.5, two HSDO outputs (O[0] and O[1]) were connected together via 2200 Ω resistors R and the summation output was fed into the 2nd analog (AN) input ADC[1]. Additional two HSDO outputs, O[2] and O[3], were looped-back to two HSDI inputs, I[0] and I[1] respectively. For the analog input voltage, it is calculated using the following voltage divider formula:

$$V_{ADC[1]} = (V_{O[0]} + V_{O[1]})(\frac{R}{R+R}) = \frac{V_{O[0]} + V_{O[1]}}{2} \tag{5.1}$$

Given that $V_{O[0]}$ and $V_{O[1]}$ are both digital outputs thus their possible values are either low (L = V_{SS} = 0V) or high (H = V_{DD} = 3.3V), the resulting $V_{ADC[1]}$ voltage cannot be but one of the following values: V_{SS}, $V_{DD}/2$ or V_{DD} taking into consideration all possible combinations in equation 5.1. For the digital HSDI inputs, I[0] and I[1] were shorted to O[2] and O[3] respectively and thus two identical pairs are expected to be captured. Figure 5.6 represents a selected window of HSDO (O[3:0] blue traces), HSDI (I[1:0] red traces) and analog (ADC[1] yellow trace) samples generated and captured at 20 MSpS. The HSDO is a pseudo-random bit sequence generated in software and downloaded into the instrument's hardware over USB. The uploaded data has been exported via API to a comma-separated-values file. The total number of the generated and acquired samples is 73728 (72K) samples per burst. Four HSDO outputs, two HSDI inputs and one analog input are shown. ADC[1] is the analog summation result of O[0] and O[1]; I[0] and I[1] are identical to O[2] and O[3] respectively. All traces have been ported to rail-to-rail scale

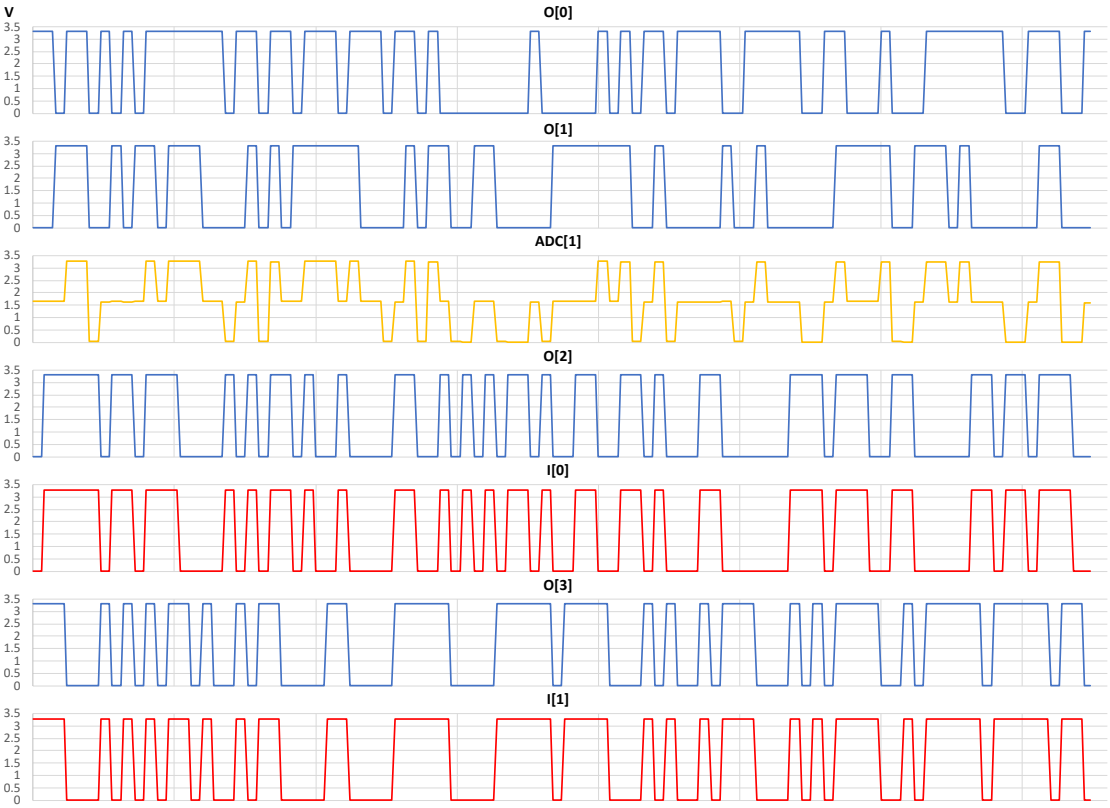

FIGURE 5.6: Generation and acquisition results.

(V_{SS} to V_{DD}). The presented results show output and input data synchronization, alignment and cycle-accurate analog measurement and thus the proposed system's requirements and specifications (data synchronization, alignment and cycle-accurate analog measurement at maximum rate) have been met.

In addition to being compact, centralized, automated, programmable and portable, the presented system combines features of three different instruments: 1) sampling oscilloscope functionalities (analog input), 2) logic analyzer capabilities (HSDI) and 3) pattern generator abilities (HSDO) as illustrated in Figure 5.7 (a). If the synchronization of three different instruments can be somehow possible, the automatic data alignment is definitely a hassle; the proposed system provides seamless data synchronization and alignment at a fraction of the instrumentation cost integrating three different instruments into a single one as shown in Figure 5.7 (b).

Moreover, in Table 5.1, we present a comparative study covering most of the recent related work. Many criteria have been selected of that purpose, for instance: PC interface, instrumentation functionalities including Pattern Generator (PG), Logic Analyzer (LA) and Sampling Oscilloscope (SO), dependency, portability, reconfigurability, cost, etc. This comparison has highlighted the main similarities and differences, and showed the presented system's efficiency and points of strength.

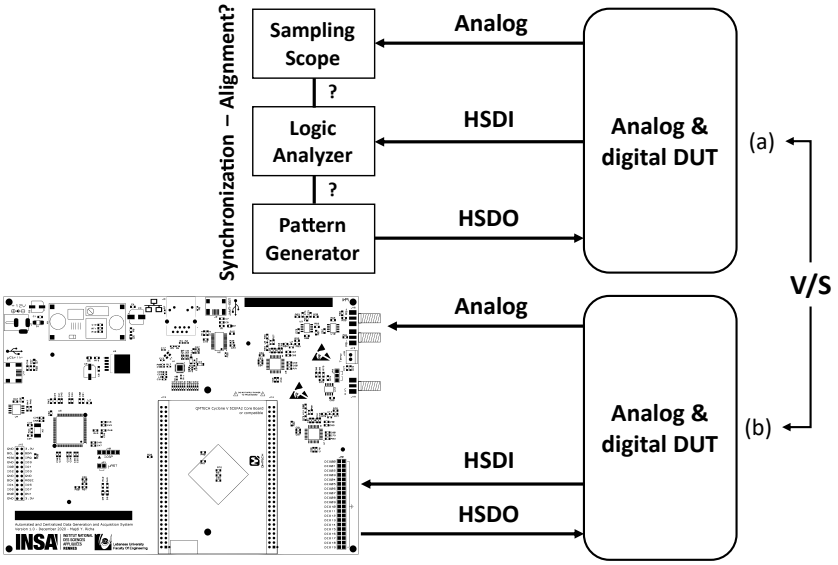

FIGURE 5.7: Three different instruments solution (a) v/s a single board 3-in-1 solution: PG, LA and SO (b).

TABLE 5.1: Comparison

		[34]	[21]	[12]	[18]	[77]	This work
PC Interface		USB	PXI	USB	PCI	RS232	USB/FE
Instrumentation	PG	no	yes	no	yes	no	yes
	LA	no	no	no	no	no	yes
	SO	yes	yes	yes	yes	yes	yes
Dependency		none	LabVIEW	LabVIEW	LabVIEW	none	none
Portable/Compact		yes	no	no	no	yes	yes
Single-board		no	no	no	no	no	yes
General purpose		no	no	no	no	no	yes
Reconfigurable		yes	no	no	yes	no	yes
Open source		no	no	no	no	no	yes
Cost		low	moderate	high	moderate	low	low

5.3 FPGA measurements

The accuracy and fidelity of the acquisition platform in context refer to how well it measures the power consumption of the IP core and how closely the measured values match the actual power consumption. In order to validate the accuracy and fidelity of the power IP characterization system, the measurement platform applies a series of test patterns or stimuli to a variety of IP cores and measures their power consumption. For that purpose we conduct a series of power consumption acquisitions, we plot the resulting power curves and we discuss the findings.

Here we describe the measurement setup. High Speed Digital I/Os (HS-DIO) are feeding the FPGA as 16-bit input signals and the power consumption

measurement is sampled via a high speed differential Analog to Digital Converter (ADC) through a precision shunt resistor R_S. The measured core power is calculated using equation 5.2 given that the core current is provided by the instrument's input signal conditioner using equation 5.3.

$$P_{meas} = V_{DD}.I_{core} - R_S.I_{core}^2 \quad (W), \qquad [V_{DD} = 1V] \tag{5.2}$$

$$I_{core} = \frac{\frac{ADC}{2^{10}}(V_{Ref+} - V_{Ref-})}{40R_S} \quad (A), \qquad [R_S = 0.5\Omega] \tag{5.3}$$

ADC is the 10-bit converted value, and V_{Ref+} and V_{Ref-} are the positive and negative references respectively. Figure 5.8 reveals the capabilities of the instrumentation system where 9 sets of components are measured. Each set consists of U parallel 8×8 multipliers (U ranging from 25 to 400). The 16-bit stimuli is driving the inputs of the multipliers. The graph reveals four phases of power levels for each set. Phase 1 only shows static power, where the clock is disabled and the stimuli signals are at logic 0. In this case, power consumption is minimal. Phase 2 exhibits both static and clock power, when the clock is enabled and stimuli signals remain at logic 0. In this phase, a slight increase in power consumption is detected. Phase 3 shows both static and dynamic power, when the clock is enabled and the stimuli signals toggle to logic 1, thus leading to a single transition. A single power peak is recorded upon the stimuli transition from logic 0 to logic 1. Finally, phase 4 shows static and dynamic power, when the clock is enable and stimuli signals are toggling at a maximum switching rate. The power consumption is at a maximum level for all 9 configurations.

Note that the 9 power traces are identical in shape with different power level offsets. The tested FPGA has a V_{DD} core voltage of $1V$ and a shunt (current sense) resistor R_S of 0.5Ω. The same circuit will be used throughout this work. It's also worth noting that the slow slope edges especially in phase 4 are strictly due to power supply filtering (capacitors) at the DUT FPGA core voltage.

The measurement process is automatically performed under a centralized software control. To ensure data integrity, both the output stimuli and input analog samples are synchronously generated, captured and aligned at the instrument level by hardware (FPGA). To ensure temperature stabilization, a fan is placed in close proximity of the FPGA. This hardware setup will be also used to characterize various IP components and subsequently evaluate our proposed work (chapters 6 and 7 respectively).

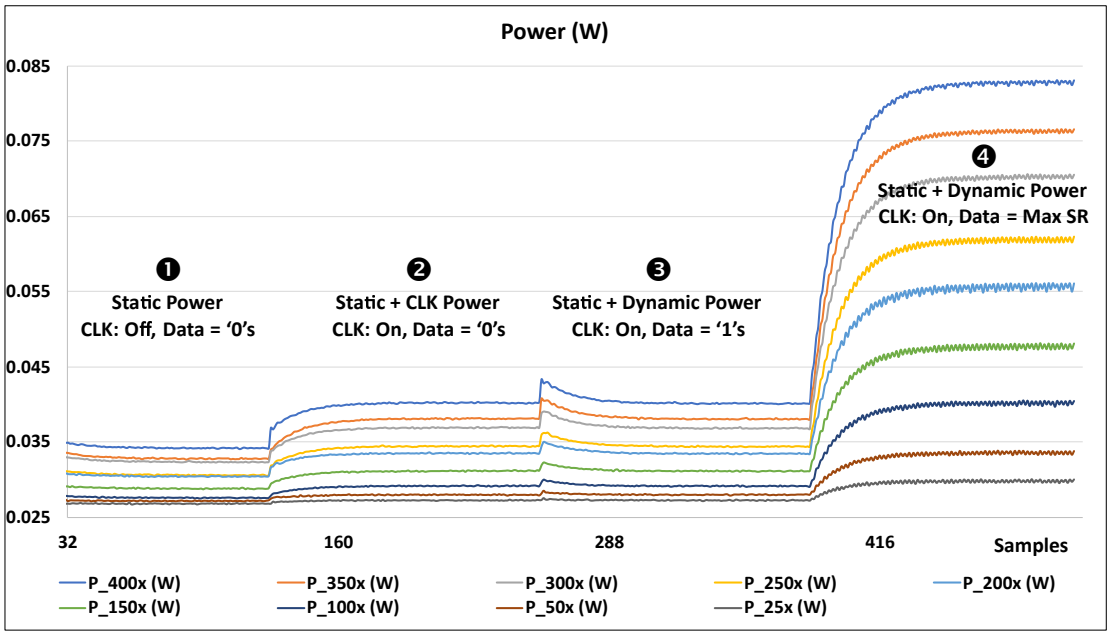

FIGURE 5.8: Power traces of 9 sets of parallel multipliers during 4 different stimuli phases.

5.4 Conclusion

In this chapter we have presented an automated and centralized data generation and hybrid acquisition system. First of all, we have described its various modules: high-level hardware covering the internal FPGA logic (including interfacing options) and the digital and analog parts; and the software aspect covering both the firmware alternatives and the PC interfacing options. The generation and acquisition methodology has been detailed revealing data flow procedures and relevant information extraction. Experimental results have been exposed using a special hardware setup and a specific testing scenario. The collected data showed equally important data synchronization, data alignment and cycle-accurate analog measurements using pseudo-random bit sequences at the HSDO. Also, we have compiled a comparative table showing main differences between the presented system and related work. A series of FPGA measurements have been also conducted proving the capabilities of the presented generation and acquisition system.

The presented characterization platform is used throughout this work for collecting power measurements from FPGA-based target IPs. The power information will be subsequently used for later data processing and training data sets creation for machine learning techniques. The main purpose is the ability to estimate power consumption in early design phases or at runtime, and to subsequently create power consumption models. Chapter 6 elaborates more on that subject.

Chapter 6

Power Model Generation

6.1 Introduction

Power or energy system modeling is the process of building abstract models to perform analysis and subsequently estimate power consumption of digital circuits according to specific criteria. The power model generation process typically involves several steps, including power measurement, model parameter estimation, and model validation. Accurately measuring the power consumption of an FPGA design is essential for generating reliable power models.

In this work, the proposed power consumption estimation model is based on machine learning and specifically on supervised neural networks [23]. As defined earlier, Machine Learning (ML) is the process of developing Artificial Intelligence (AI) in computers, where the generated models are trained using appropriate learning algorithms and training data. For many machine learning techniques, especially the ones related to supervised methods, the construction of the training data highly affects the quality and accuracy of the derived model. In this chapter we present an automated training set construction methodology where data is synchronously collected from both hardware and software. The complete design and data flow including the interaction between software and hardware, are thoroughly described. As a direct application, this work targets the construction of an FPGA-based circuit power modeling for subsequent early power estimation, or, online power monitoring. The constructed artificial neural network model is an MLP trained using real measurement data sets extracted using a dedicated in-house designed and implemented generation and acquisition platform. The designated application falls under the power optimization area, becoming nowadays a major concern for most digital hardware designers, particularly in early design phases and especially in limited power budget systems. The power optimization approach in context is also extended in order to support online power management.

This chapter is organized as follows: in section 6.2 we present the automated training data sets construction that includes a brief description of the

measurement system and an elaboration on the training sets. We also describe the stimuli generation algorithms. Section 6.3 reveals the proposed learning-based power model including the automated work flow, and the software architecture and interface. In section 6.4, as a proof of concept, we implement the power estimator's neural network in FPGA and we discuss the hardware usage and the realtime performance. Finally we conclude in section 6.5.

6.2 Automated Training Data Sets Construction

Automated training data set construction is a crucial aspect of machine learning and artificial intelligence. It involves creating high-quality training data sets that can be used to train and improve machine learning models. Automated training data set construction involves using software tools and algorithms to extract relevant information from various sources and compile it into a single data set. These tools can also be used to clean and pre-process the data, ensuring that it is of the highest quality.

For the work related to training data construction and generated models, in [49], authors describe a method to quantify the effect of the training data on the derived machine learning model. Their work includes the quantification of the variation exhibited by several algorithms using permutations of a given training data set. As a result, they demonstrate that this kind of variation can be significant and that training data ordering is an important consideration. However, in [53], authors analyze how the accuracy of prediction will vary with the different combinations of training and test data. The mean absolute error is calculated by comparing the actual values from testing data and predicted values from the model.

There are several benefits of using automated training data set construction, including:

- Improved accuracy: Automated training data set construction ensures that the data used to train machine learning models is accurate and relevant, which can result in better performance and more accurate predictions.

- Increased efficiency: By automating the data set construction process, organizations can significantly reduce the time and cost involved in creating high-quality training data sets.

- Scalability: Automated training data set construction can be easily scaled to handle large amounts of data, which is particularly important when dealing with big data applications.

For the presented work and in order to build an efficient machine-learning-based prediction system, training sets have to be carefully constructed. In our case, the collected data is derived from two different sources: the stimuli generation algorithms and a hardware data acquisition system providing the power consumption values collected after applying the generated stimuli on a given IP.

6.2.1 Data acquisition system

As described in chapter 5 and in order to collect precise power information, a reliable measurement platform is required. For that purpose, we have proposed in a previous work an FPGA-based Automated and Centralized Data Generation and hybrid Acquisition System (ACDGAS) [67].

6.2.2 Elaboration of the training sets

Training data sets are essentially the foundation upon which machine learning algorithms are built. These data sets are used to teach machine learning models to recognize patterns, make predictions, and perform various other tasks. Therefore, the quality of the training data sets is directly proportional to the performance of the machine learning model.

In order to build an efficient machine-learning-based prediction system, training sets have to be carefully constructed. In our case, the collected data is derived from two different sources: the stimuli generation algorithms and a previously presented hardware data acquisition system providing the power consumption values collected after applying the generated stimuli on a given IP.

Regarding the stimuli generation shown in Figure 6.1 a), two sets of PLH and SR of H and S values are respectively used to generate M ($= H \times S$) distinct combinations of [PLH, SR] pairs. The aforementioned combinations are coupled with N control signals or operating modes (denoted as CS/OM or simply CM) thus leading to 2^N states or modes. Each permutation of the CM signals is appended to I pairs of [PLH, SR], subsequently generating one stimuli matrix as shown in Figure 6.2. The corresponding training data size D is equal to $M \div I$. All generated sets belong to specific ranges and are randomly shuffled. Each pair will eventually generate a bit sequence using a well-defined stimuli generation algorithm as shown in Figure 6.2, where the inputs are PLH, SR and the bit width is L. The generated pairs of [PLH, SR] will be sequentially applied to I-bit dash-marked inputs (–.–.–) (Figure 6.1 a) feeding the FPGA and resulting in D matrices of stimuli signals. For the hardware-based section, D

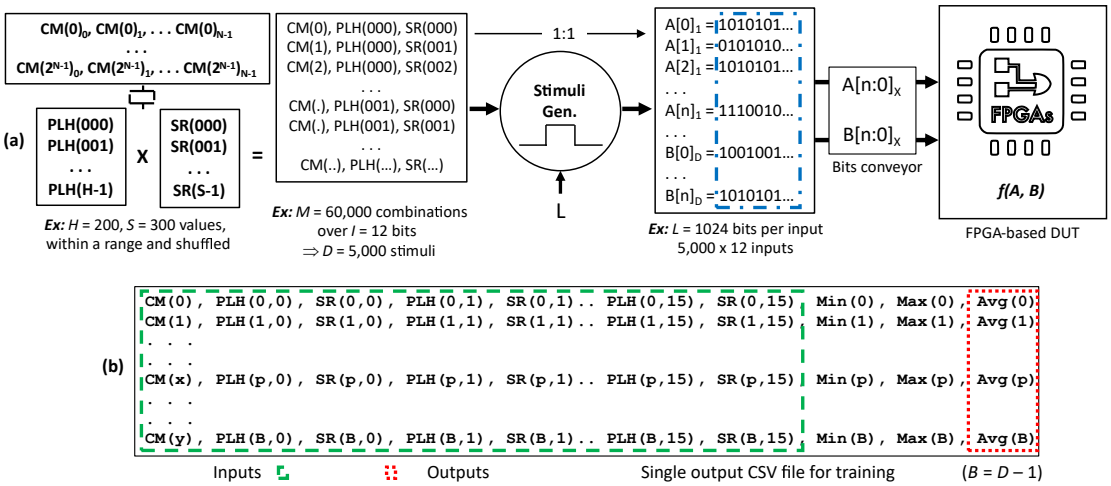

FIGURE 6.1: a) Generation and pairing of PLH, SR and stimuli coupled with CM - b) Training data format.

power consumption measurement sets corresponding to the aforementioned stimuli signals are collected and appended to the initial software-based, [CM, PLH, SR] combinations and subsequently forming the complete sorted training data for the power model. Figure 6.1 b) reveals the structure of the D-entry training data set with dash-marked inputs (– – –) and dot-marked outputs (...).

Stimuli bits generation

Stimuli generation refers to the process of creating binary codes or digital signals that can be used to stimulate or trigger a response in a specific target device. The subsequently generated bit sequence can be deterministic or random. In deterministic bit sequence generation, the sequence is generated according to a specific rule or algorithm, which produces the same output each time the process is run. In contrast, random bit sequence generation produces a sequence of bits that appear to be randomly generated and do not follow a specific rule or algorithm. Here, the bit sequence construction process used for the stimuli matrix generation, where both PLH (percentage of 1s) and SR (percentage of transitions) are compatible and relative to the bit width L, may be implemented using different alternatives. In this work we propose a two-layer deterministic bits generation mechanism where the sequence is generated according to a specific rule or algorithm, and produces the same output each time the process is initiated with the same parameters. The bottom layer, denoted as Iterative Bit Sequence Generation (IBSG), is a two-step procedure:

1. It generates a sequence of alternating bits that fulfill the total number of required transitions minus one, and

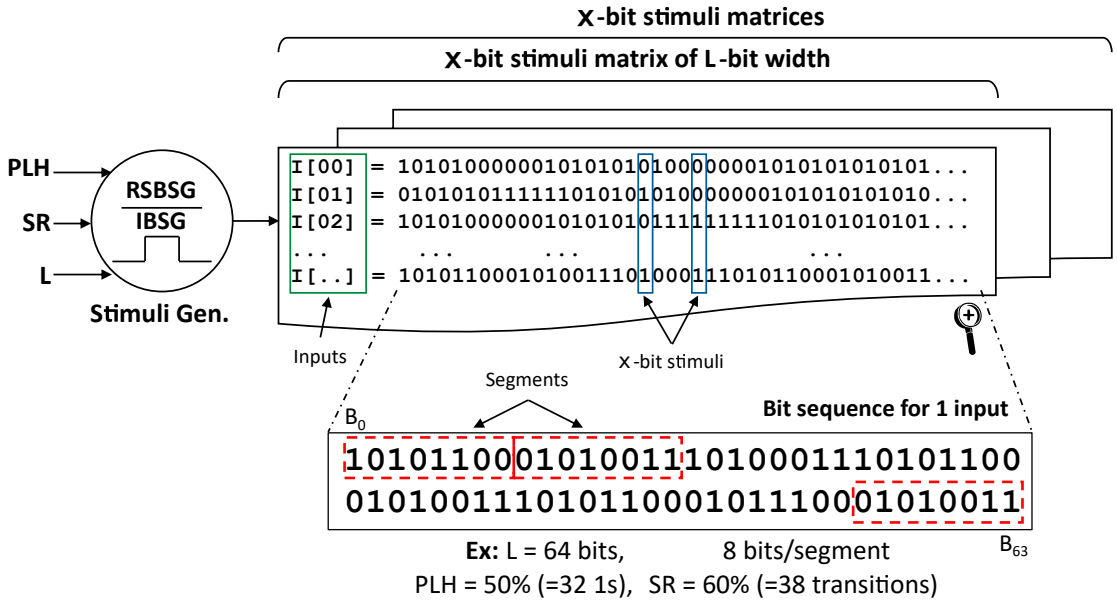

FIGURE 6.2: Parametric bit sequence generation using RSBSG/IBSG algorithm.

2. It pads the generated bits with two contiguous sequences of (the remaining) zeros and ones, thus adding one final transition and fulfilling the sequence's requirements.

The upper layer, denoted as Recursively Segmented Bit Sequence Generation (RSBSG), leverages the aforementioned iterative linear algorithm by employing it as the core bit generator in a recursive segmentation approach. Instead of generating one single sequence with the required parameters, it recursively generates and concatenates multiple bit-segments by evenly distributing the PLH ratios to each segment while keeping the segments' SR parameter constant. Special consideration had to be applied to the bits' values of two neighboring segments when being attached together, as to avoid the undesirable side-effect of added transitions at the segment boundaries. In summary, the proposed algorithm ensures the [PLH, SR] pairs' compatibility and fulfills the stimuli signal's requirements while achieving a high degree of entropy in the generated bits (adequate SR distribution and minimal bit-clustering). Figure 6.2 reveals the RSBSG/IBSG algorithm with a sample sequence derived from specific parameters.

6.3 Proposed model

As mentioned earlier, the proposed data set consists of thousands of patterns of I pairs ($I \times 2$) inputs representing matches of (SR, PLH) coupled with N CM signals (a, b, etc.), in addition to the output average power. The proposed

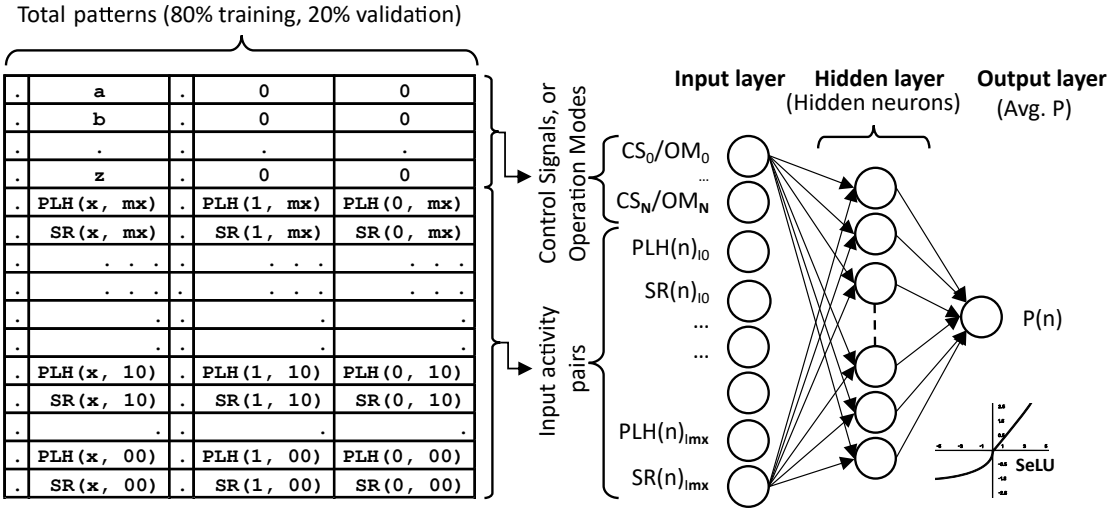

FIGURE 6.3: Neural network architecture showing training data, layers, neurons and activation function (SeLU).

neural network architecture is compiled under the TensorFlow/Keras Python library. As shown in Figure 6.3 (right-hand side), it consists of the input layer fed by N binary CM inputs and I pairs of SR and PLH, one hidden layer and one output layer representing the estimated (average) output power. The number of hidden neurons has been optimized following each test case providing a trade off between prediction speed and accuracy. The Scaled Exponential Linear Unit (SeLU) is chosen as the activation function due to its self normalizing nature and its high learning robustness [50]. The training set size represents 80% of the total data resulting in thousands of entries, out of which 80% are used for training and 20% for validation. The remaining samples are reserved for evaluating the proposed neural network model, and represent 20% of all available patterns. The network optimizer is selected to be *Adam* that is a replacement optimization algorithm for stochastic gradient descent for training deep learning models [80]. The adopted training loss metric is the Mean Squared Error (MSE) while the evaluation metrics are the Mean Absolute Error (MAE) and the Mean Absolute Percentage Error (MAPE). The number of training epochs is optimized to be 25 rounds of 1 batch per iteration. An epoch refers to a complete pass through the entire training data set during the learning process.

The complete automated work and data flow is illustrated in Figure 6.4. For that purpose, a specially developed software tool interfaced to both the measurement system's Application Programming Interface (API) and to Python interpreter, is orchestrating the stimuli generation, the acquisition system and the power model generation (Figure 6.5). This proposed process encapsulates the following sequential steps:

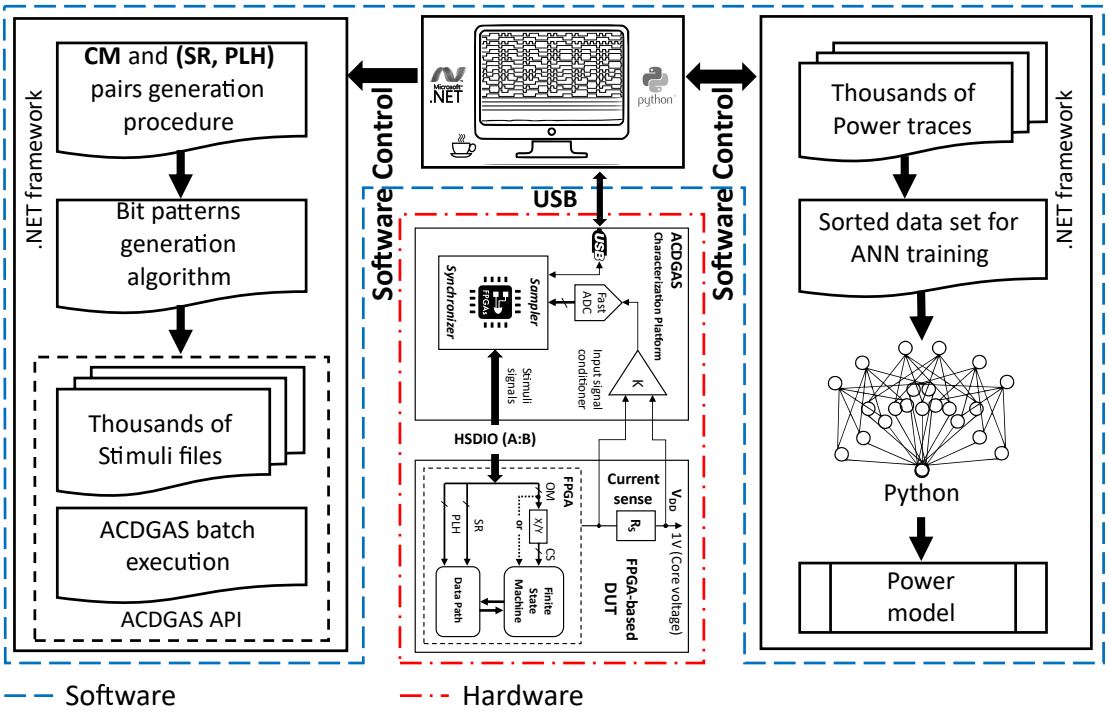

FIGURE 6.4: Automated work and data flow showing data generation and acquisition on the left-hand side and ANN training and power modeling on the right-hand side.

1. Generation of CM (CS/OM), [PLH, SR] tables and combinations, and resulting stimuli matrices.

2. Generation of stimuli files, creation of subsequent ACDGAS batch-job and execution.

3. Collection of power consumption samples (in hardware) and construction of training data sets.

4. Data normalization, neural network compilation, training and evaluation.

The proposed layered software architecture shown in Figure 6.5 delivers a fully automated process for IP characterization. It provides a solid and synchronized interaction between the various system modules such as: graphical user interface, stimuli construction, control signals coupling, bit sequence generation, measurement platform interface, training data construction, power modeling and evaluation.

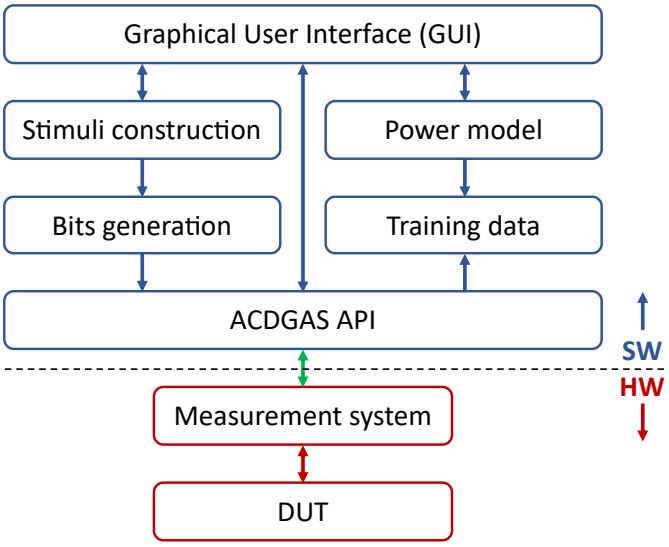

FIGURE 6.5: Software layers going from graphical user interface to the hardware interface.

6.4 Hardware implementation

As discussed in chapter 4 the proposed power consumption modeling and estimation covers both offline and online domains. For the online power management solution, implementing an artificial neural network (the power model) in FPGA is becoming increasingly important especially if the IP circuit is residing inside of the same FPGA. Moreover, FPGAs can be used to perform real-time processing, making them suitable for applications requiring rapid and live decision-making.

Traditionally, power consumption monitors (also known as current sensors), whether built-in or external, measure the total consumed energy of the FPGA's core and cannot differentiate between various co-existing IPs' consumption. In general, these sensors operate over the Power Management Bus (PMBus). The PMBus is a variant of the System Management Bus (SMBus) which is targeted at digital management of power supplies. It is a relatively low speed, two-wire serial communication protocol, based on Inter-Integrated Circuit (I^2C) [76]. This also applies to the voltage/frequency controllers (variable/configurable regulators/PLLs/VCOs), also operating over the same PMBus. That said, and as an undesirable effect, this scenario injects delays in the power monitoring and subsequent management system's response. For instance, a PMBus clocked at 400 KHz (fast mode) produces, at best, a latency of 300μs per one iteration, assuming no (embedded) software overhead is present [26].

Implementing an artificial neural network (ANN) inside the FPGA can be done either by software via an embedded processor or by hardware using a

TABLE 6.1: ANN hardware implementation: usage and latency.

Bits per iteration	Hidden neurons	Initial interval (cy)	Latency (cy)	DSP	FF	LUT
512	4	512	926	12	6,104	11,936
	8	512	978	12	6,389	11,918
	16*	850*	1,362	12	6,396	11,926
1024	4	1,024	1,438	12	6,105	11,938
	8	1,024	1,559	12	6,218	11,891
	16	1,024	2,023	12	6,226	11,899

dedicated digital circuit implementation [78]. In order to accelerate the proposed ANN and subsequently eliminate its prediction latency effect, the hardware implementation is a must. This latter can be pipelined while processing in parallel and in real-time. However, a trade off between FPGA logic usage and prediction accuracy/speed is inevitable. Table 6.1 shows the neural network usage and latency (in cycles) when implemented using 4, 8 and 16 hidden neurons respectively. To provide additional implementation flexibility, at the input layer, data are scanned in batches of either 512 or 1024 bits per iteration in order to extract PLH and SR and subsequently estimate power consumption. Processing at 100 MHz, the recorded latencies show that the neural network is performing in real-time except for the 16 (*) hidden neurons version when operating in 512 bits per iteration. In this case the initial interval cycles surpass the number of bits per iteration. The recorded power estimator latencies ($< 10\mu s$ for 512 bits and $< 20\mu s$ for 1024 bits), compared to most standard communication-based overhead, showed an improvement of above 90%. The target FPGA used is the AMD/Xilinx Artix-7 (xc7a35tftg256-2) under the High-Level Synthesis (HLS) tool provided by Vitis HLS 2022.2 using an off-the-shelf neural network IP.

6.5 Conclusion

Machine learning-based power consumption estimation is a hot topic for digital hardware designers. For instance, it can be used to explore various design choices very rapidly in the design process. It can also be used in a given circuit to efficiently predict and then accurately manage power at run-time. Therefore, online power monitoring and subsequent management is becoming a hot topic since it provides on-the-fly energy optimization.

In this chapter, we leverage the machine learning techniques to establish a novel neural-network-based power consumption model and estimation approach for IPs in FPGAs. The proposed model is capable of providing fast and accurate average power estimation by just specifying the control signals and the inputs characteristics (PLH and SR) of a given IP when dealing with

the offline domain. For the online management, we estimate the power consumption in-situ and in real-time by just providing the most energy-significant modes of operation coupled with the input activity of individual FPGA IPs. The complete work flow is presented, covering the automated software control, the measurement system and the description of the machine learning training sets. We also proposed an efficient neural network model to be evaluated in the upcoming chapter.

Overall, power model generation for FPGA IPs is a critical aspect of FPGA design that enables the development of energy-efficient systems. By accurately estimating the power consumption of FPGA designs, power models help designers optimize power consumption and improve the performance of FPGA-based systems. In the following chapter, we demonstrate that the resulting power estimation shows highly accurate results with a minimized prediction mean absolute percentage error.

Chapter 7

Validation, Results and Applications

7.1 Introduction

The ultimate goal of any research framework is to produce valid and reliable results that can be used to advance knowledge and inform decision-making. This chapter is a critical component of any research project, as it focuses on the process of validating the research findings and presenting the results in a clear and concise manner. In this chapter, we provide a detailed description of the methods used to validate the data, as well as an in-depth analysis of the obtained results. The validation process ensures that the data collected is accurate, reliable, and consistent. The results presented in this chapter are essential for ensuring that this research work meets the highest standards of quality and rigor, and for demonstrating its value and impact to the broader scientific community.

This chapter is organized as follows: in section 7.2 we present the experimental results covering both the offline and online areas of power estimation. For the offline domain, the results include FPGA IPs representing data path only, and FPGA IPs representing finite state machines in addition to the data path. For the online alternative, the results cover the realtime power estimation of FPGA IPs characterized during the most significant (and power influential) modes of operation. Furthermore, we investigate fault detection methods based on the online power estimation using profiling. In section 7.3, we assess the generated power models by first presenting the machine learning curves including training and validation results, and by revealing the IPs resources and power estimation performance in addition to fault detection proof of concept.

7.2 Experimental results

In order to evaluate the proposed power consumption estimation model, it is essential to conduct experimental tests. We have applied our methodology on several FPGA circuits and collected power information in order to build the power model and record the obtained results. An off-the-shelf Xilinx Artix-7 (xc7a35tftg256-2) FPGA running at 100 MHz was used in this context. Here we should mention that, in the following test cases, the dynamic power is dominant. The presented results cover both offline and online power consumption scenarios. This latter was also extended to cover fault detection technique.

7.2.1 Offline test cases

Offline power estimation of FPGA IP refers to the process of predicting its power consumption before it is physically implemented on the device. The purpose of offline power estimation is to optimize the power efficiency of the FPGA design by identifying potential power-hungry areas and adjusting the design accordingly. This can help to reduce the power consumption of the FPGA design and improve its overall performance and reliability. The offline power estimation covers the following targets: 1) data path only and 2) state machine and data path.

Data path only

In order to demonstrate the feasibility of the approach, we selected different FPGA IP components as case studies. First, basic multipliers, then Multiply and ACcumulate (MAC) units and finally Optical Transport Unit Data Aligners (OTU-DA) have been proposed since they feature different characteristics in terms of logical resources and internal connections. Each IP has a distinct number of resources, therefore it has different power levels. A variable number of identical IPs' instances has been grouped together in an attempt to place the consumed power of various IP groups in the same range. This had to be done to adapt to the instrument's current limit setup and calibration.

The output wires of each component were not routed to FPGA pins and thus ignored. All IPs were tested under the same neural network architecture excluding the number of hidden neurons. Each component has been characterized by generating its input signals according to three stimuli criteria (PLH, SR and L = $4Kbits$). Then, the average output power per stimuli matrix has been collected. Note that $10,000$ matrices of 16-bit stimuli have been considered in total, where each matrix encapsulates $4K$ stimuli. Figure 7.1 represents

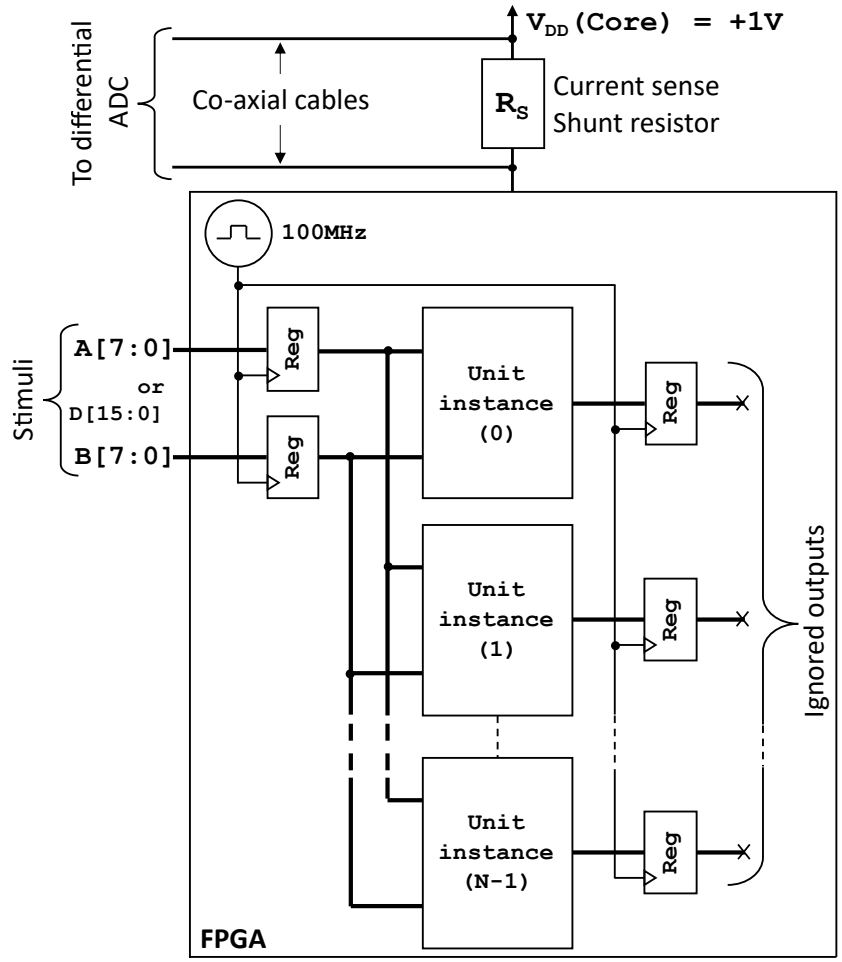

FIGURE 7.1: Generic FPGA design and corresponding interconnects and I/Os.

a generic group of IPs in which the inputs are connected to stimuli and the outputs are ignored.

Multipliers example: 75 instances of 8-bit multipliers ($f(A[7:0], B[7:0]) = A \times B$) were instantiated with all having the same inputs. Note that the size of inputs is limited due to the number of I/O pins that are accessible on the measurement platform. In this first example, DSP blocks that are available in the FPGA have not been used. The stimuli generated by the proposed methodology are fed in parallel into the multipliers inputs. The acquired analog samples represent the IPs consumed power while their digital inputs were toggling.

MACs example: 50 instances of 8-bit MAC units ($f(A[7:0], B[7:0]) = f(A, B) + (A \times B)$) were instantiated with all having the same inputs. Two MAC flavors were tested: one without DSP blocks and one with DSP.

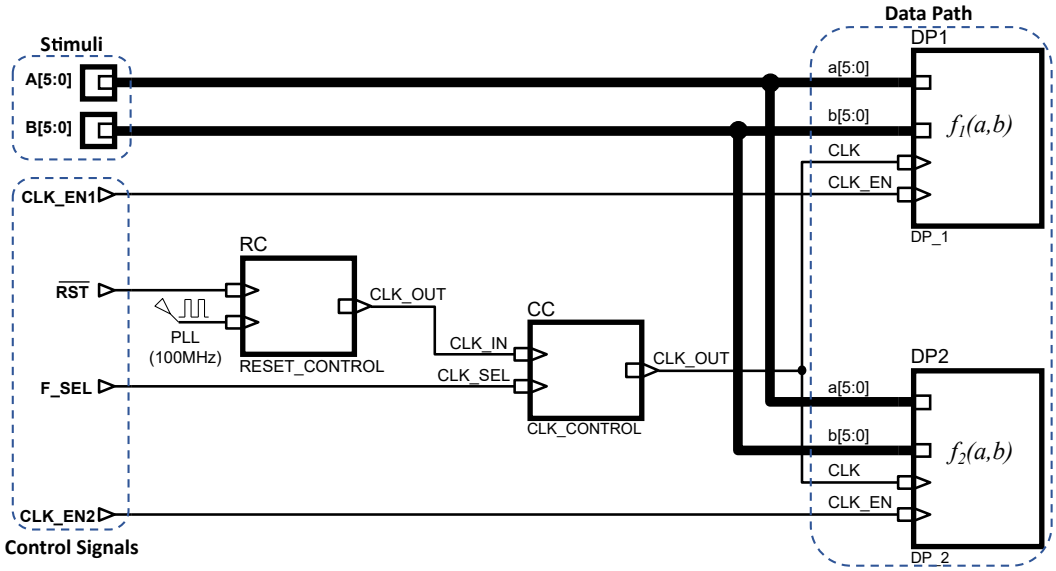

FIGURE 7.2: FPGA IP circuit with 4 control signals and 2 DPs.

OTU data aligners example: By definition, an OTU-DA compares incoming data from a deserializer unit to a specific Frame Alignment Signal (FAS) and synchronizes it accordingly. When the FAS is detected, the OTU-DA signals a lock detection [1]. This specific target, in addition to being an off-the-shelf available IP, has been selected as it provides a real use case with a combination of various logic components such as accumulators, shifters, comparators, registers, etc. with the exception of multipliers. In this example, 100 instances of 16-bit ($D[15:0]$) OTU-DA units were instantiated with all having the same inputs.

State machine and data path

Figure 7.2 represents a generic FPGA IP circuit with 4 control signals (leading to $2^4 = 16$ states) and 12 data path inputs. In order to obtain diversified combinations subsequently leading to additional states, the data path is divided into two sections DP1 and DP2. Each section is defined by a specific function, $f_1(a,b)$ and $f_2(a,b)$ for DP1 and DP2 respectively. Each function includes arithmetic and/or logical operations. The control signals are adequately chosen as follows: reset ($\overline{RST}$), clock enable for DP1 and DP2 (CLK_EN1 and CLK_EN2 respectively) and frequency selection between CLK and $CLK/2$ (F_SEL). Figure 7.3 represents another FPGA IP circuit with 4 control signals (leading to $2^N = 16$ states) and 12 data path inputs. Two IP circuits are considered as test cases: IP1 with $f_1(a,b)$ and $f_2(a,b)$, and IP2 with $f_1'(a,b)$ and $f_2'(a,b)$ for DP1 and DP2 respectively. The aforementioned functions are detailed in Table 7.2 and Table 7.3.

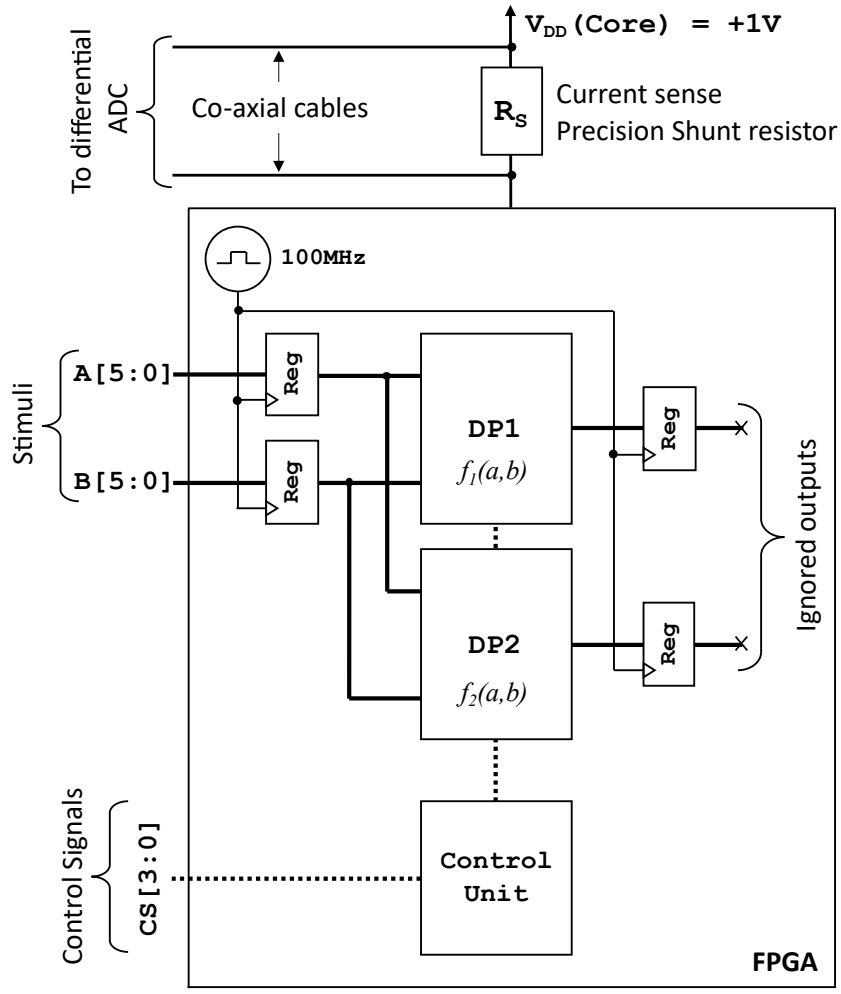

FIGURE 7.3: Generic FPGA design and corresponding interconnects and I/Os.

7.2.2 Online test cases and application

Online, or real-time, power estimation of FPGA IP refers to the process of continuously monitoring its power consumption during its operation. The purpose of online power estimation is to provide designers with real-time feedback on the power usage of the FPGA design, allowing them to optimize the power efficiency of the IP block while it is in operation. This can help to ensure that the design stays within its power budget and remains reliable, while also enabling designers to make adjustments and optimize the design further based on real-world operating conditions. As a test case, we hereby present:

1. A black-box IP online power monitoring and estimation, and

2. A realtime FPGA IP input fault detection application using online power monitoring extension coupled with power profiling.

Black-box IP

Since the proposed methodology is not aware of the FPGA IP under monitoring (functionality, components, interconnections, etc.) and in order to generalize our real-time power estimator, we have selected a black-box FPGA IP to put under test. A black-box IP is a digital circuit whose functionality and internal connections are masked. As stated earlier, the online power estimator relies only on the extracted switching activity of the IP's data path inputs while being coupled with a certain number of operation modes. These latter can be either extracted from the control signals of the same IP's state machine or simply provided by the designer. The black-box IP in context has 12 inputs and 16 modes of operation encoded over 4 bits. It has been tested under two different power estimation models, each generated using a specific neural network implementation: one with 4 hidden neurons and one with 8. The 16-neuron alternative (and above) has been discarded due its relatively high latency and complexity.

Fault detection methodology

Realtime fault detection in digital circuits using power estimation involves monitoring and analyzing the power consumption of a circuit to identify potential faults or anomalies. By continuously monitoring the power consumed by different components of a digital circuit, it is possible to detect abnormal power patterns that may indicate the presence of faults, or in some cases, intentional meddling. One of the main mechanisms closely related to realtime power-based fault detection is the power profiling.

To begin with, we list and investigate some of the recent related work. In [71], authors use FPGA internal sensors to detect high voltage drop events at the core supply, and subsequently to monitor dynamic voltage changes. These events indicate fault/attack possibility. The main drawback resides in the accuracy of such internally-implemented sensors. Authors in [9] investigate the possibility of FPGA designs power profiling using direct board-level measurements via built-in current sensors. Multiple design parameters were selected and varied in order to detect their influence on power consumption. However, the relationship between those various parameters and their combined effect on power remains unidentified. In [51], authors perform both hardware and software power profiling of FPGA systems. An incremental profiling approach is used starting by simple components and up to multi-core architectures. The main limitation remains in the relatively low measurement sampling rate ($10Hz$) that could result in power information loss. Authors in [17]

propose a monitor for detecting timing violation-based fault attacks. The monitor in context, detects frequency or voltage manipulations using internal ring oscillators. The main drawback is the oscillation frequency: high-speed provides better resolution but consumes more power while low-speed requires less power but may not be able to detect faults properly and thus leading to false alarms.

In the proposed fault detection methodology, shown in Figure 7.4, we rely on both the (previously described) real-time power consumption estimation and a power profiling structure. This latter is derived from both input activity limits and power thresholds. We detect abnormal power consumption scenarios with the ability to individually pin-point the defective (or meddled) DP input using fault scores. The input fault scores in context, represented in a moving window, serve as an indication of possible fault inference. In this case, the previously described power monitor, in addition to relaying power information to tuning/control mechanisms such as DVFS, is now extended to accommodate a power profile. This extension allows the detection of abnormal input activities closely related to out-of-range power consumption patterns.

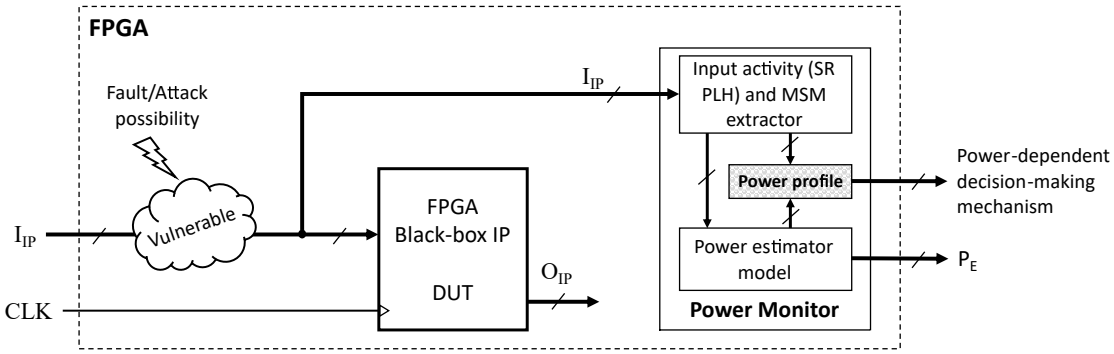

FIGURE 7.4: Fault detection mechanism of FPGA IP using online power monitoring and power profiling.

Power profile: In an online power consumption monitoring application, a power profile refers to a detailed representation of the power consumption behavior of a digital circuit (FPGA IP) over time. By analyzing the power profile, it is possible to identify abnormal patterns or corruption that could indicate the presence of a circuit fault or an attachment, such as a malicious hardware component or an attack. The proposed power profile, shown in Figure 7.5, where $M(x)$ is a given operating mode x and n a given data path input, consists of the following elements:

- Switching rate limits matrix, denoted as $SR(x)(n)_{Max}$ and $SR(x)(n)_{Min}$ and shown in Figure 7.5 (a).

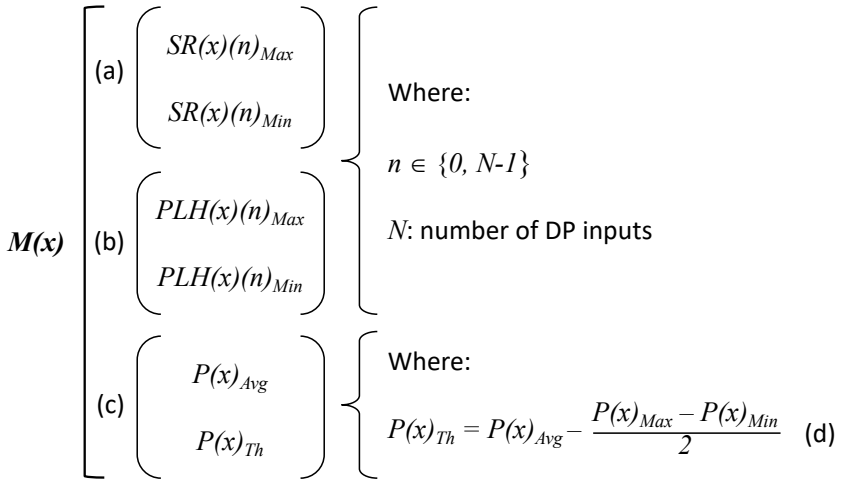

FIGURE 7.5: FPGA IP $M(x)$ power profile content: (a) SR limit (b) PLH limit and (c) Power matrices.

- Percentage of level high limits matrix, denoted as $PLH(x)(n)_{Max}$ and $PLH(x)(n)_{Min}$ and shown in Figure 7.5 (b).
- Average power and threshold matrix, respectively denoted as $P(x)_{Avg}$ and $P(x)_{Th}$ and shown in Figure 7.5 (c).
- $P(x)_{Th}$ is a threshold to compare the power difference against and is shown in Figure 7.5 equation (d).

In matrices (a) and (b), n goes from 0 to $N-1$ (N being the total number of inputs). Here we should note that all the values representing the power profile are extracted and/or calculated from the characterization phase (involving high accuracy measurements) and the training data sets previously described throughout this work.

Detection algorithm The proposed detection mechanism, shown in Algorithm 1 simultaneously scans the activity of all data path inputs in a designated operation mode $M(x)$ after detecting an abnormal power behavior. The process starts by acquiring the average estimated power $P(x)_E$ and comparing it against an expected power value $P(x)_{Avg}$ retrieved from the power profile. This comparison has a tolerance threshold of $P(x)_{Th}$ also retrieved from the same power profile. If discrepancies are detected between the estimated and stored power consumption values, the algorithm, simultaneously checks the activity of all data path inputs to make sure they all fall within the characterized limits. If any of the inputs activity is off limits, its corresponding fault score is incremented or otherwise decremented. This creates a, per-input, moving fault score window over time as shown in Figure 7.6. $F(t)_S(n)$ denotes a

snapshot of the fault score of input n at time t. It's also possible to accumulate the inputs fault score at a given moment m in time, thus providing an assessment for a given operation mode $M(x)$ at that time.

Algorithm 1: High-level, per-mode fault detection scanner.

Data: x is a designated OM, n is a selected input (out of N inputs), P_E is the estimated power.
Result: Scan all inputs of mode x and update corresponding fault scores $F(t_x)_S(n)$

```
P_E ← GetEstimatedPower(x);                          ▷ Read estimated power
if (Absolute(P_E − P(x)_Avg) > P(x)_Th) then
    for (n ← 0 to N − 1) do
        if (SR(x) is within (SR_Min , SR_Max) limits) and
        (PLH(x) is within (PLH_Min , PLH_Max) limits) then
            Decrement(F(t_x)_S(n));                  ▷ Decrement fault score
        else
            Increment(F(t_x)_S(n));                  ▷ Increment fault score
        end
    end
else
    for (n ← 0 to N − 1) do
        Decrement(F(t_x)_S(n));                      ▷ Decrement fault score
    end
end
```

7.3 Model assessment

Power modeling is an essential aspect of Electronic Design Automation (EDA). However, accurately assessing the results of power modeling is equally important to ensure that the predictions are reliable and can be used for design optimization. This is where power modeling results assessment comes in. It involves comparing the predicted power consumption values with the actual power consumption values of the circuit or system once it is physically implemented, and analyzing the differences between them. Additionally, power modeling results assessment can provide valuable insights into the performance and efficiency of the design, helping designers to optimize the power consumption and ensure the reliability of the final product. The following assessment involves both machine learning curves, and, resources and performance analysis.

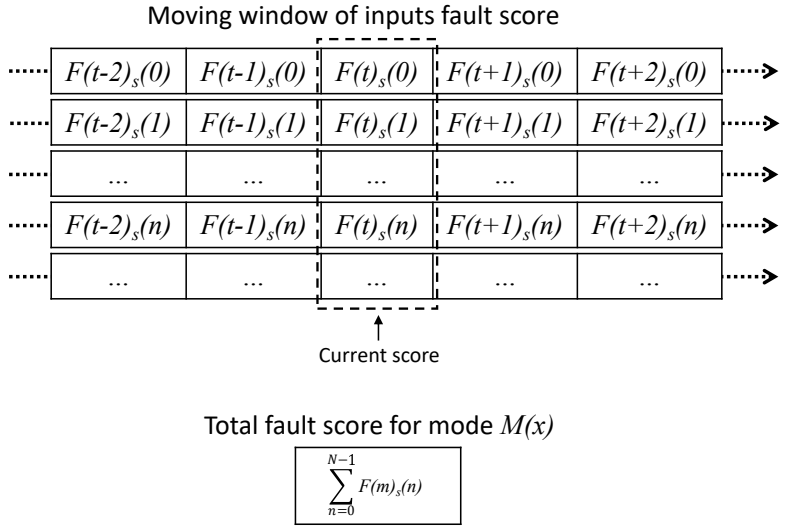

FIGURE 7.6: Per-input, fault score moving window and corresponding accumulation.

7.3.1 Learning Curves

In machine learning, learning curves are graphical representations of the relationship between a model's performance and the amount of data used to train it. Typically, the X-axis represents the amount of training data, while the Y-axis represents the performance metric, such as accuracy or loss. Learning curves can be used to evaluate the performance of a model and to help diagnose and address potential problems. Learning curves can be used to make decisions about model training, such as whether more data are necessary or if the model is suffering from overfitting or underfitting. Additionally, learning curves can be used to compare the performance of different models or to evaluate the effectiveness of different feature sets.

Data path only

Figures 7.7 (a), (b), (c) and (d) represent the training loss and the MAPE of both data training and model validation for the different IPs (multiplier, MAC, MAC DSP and OTU-DA) respectively. The loss metric is selected to be the MSE and the MAPE represents the power consumption prediction/estimation percentage error.

State machine and data path

Figure 7.8 represents the training loss and the MAPE of both data training and model validation for the IP in context. The loss metric is selected to be the MSE whereas the MAPE represents the power consumption prediction/estimation

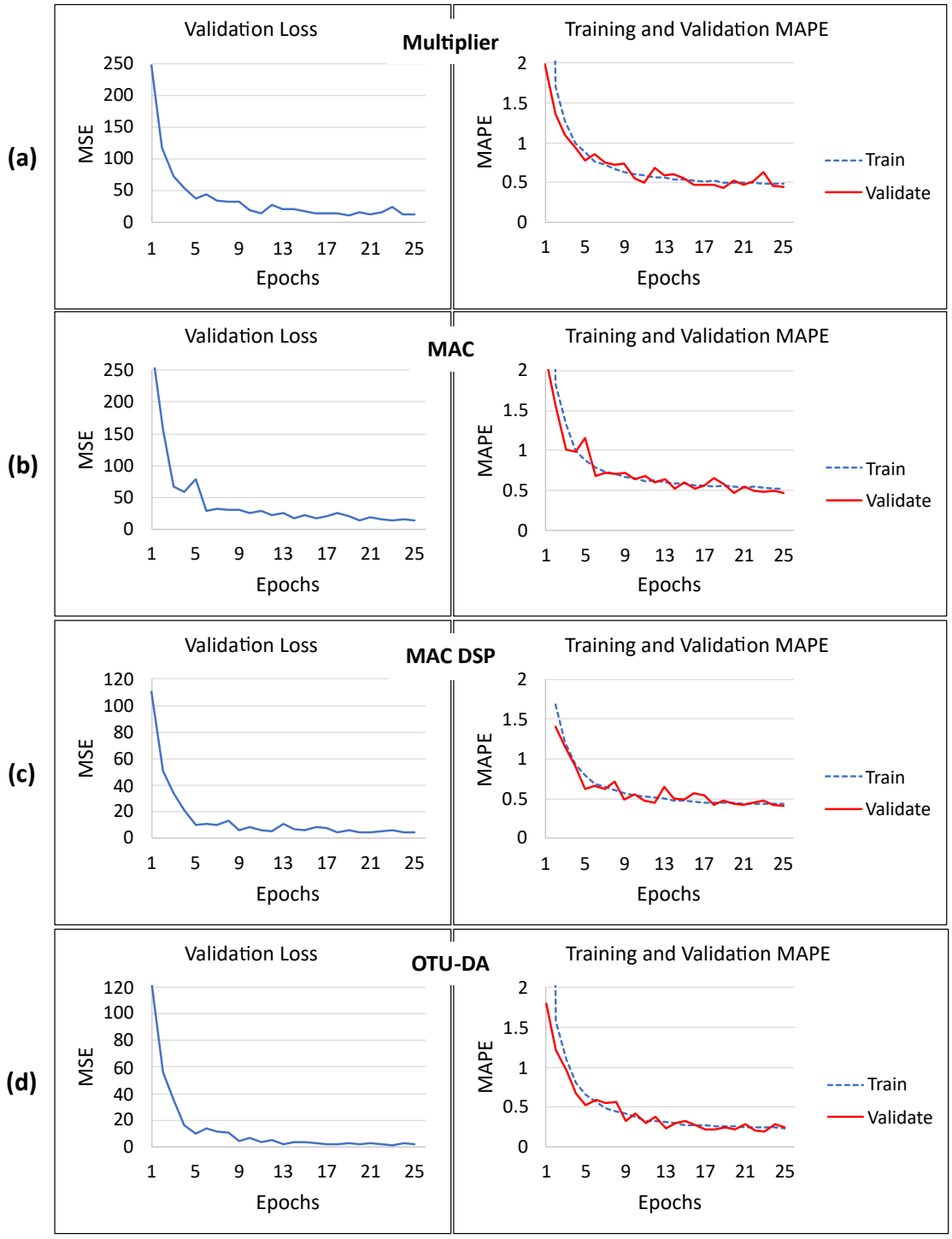

FIGURE 7.7: Learning curves for each component showing training loss and MAPE v/s number of epochs.

percentage error. Both the loss and MAPE are displayed on the vertical axis of each graph. The number of training iterations denoted as Epochs is represented on the horizontal axis. As the number of epochs increases, both the prediction loss and the MAPE decrease drastically, proving the efficiency of the proposed architecture. Beyond 25 epochs, the numbers hit a steady state.

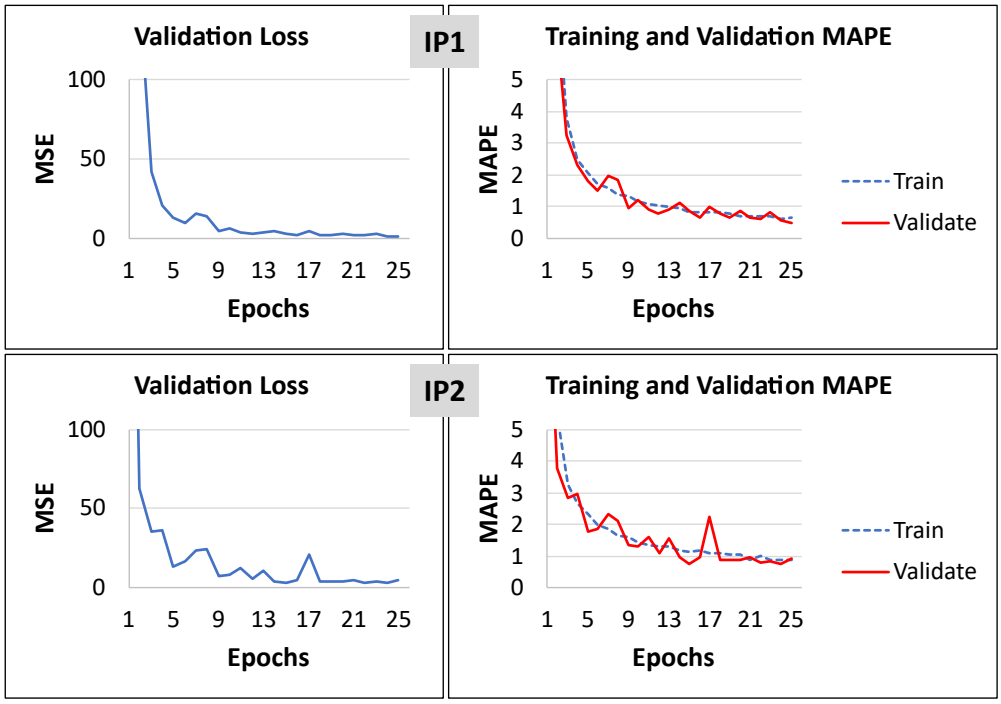

FIGURE 7.8: Learning curves showing training loss and MAPE v/s number of epochs.

Online estimation

Figure 7.9 represents the training loss and the MAPE of both data training and model validation for the different power estimators. The 4 hidden neurons version is shown in (a) whereas the 8 hidden neurons model is shown in (b). Both the MSE and MAPE are displayed on the vertical axis of each graph pair respectively. The number of training iterations denoted as Epochs is represented on the horizontal axis. As the number of epochs increases, both the prediction loss and the mean absolute percentage error decrease drastically, proving the efficiency of the proposed architecture. Beyond 25 epochs, the training loss (MSE) and the validation (MAPE) values hit a steady state.

7.3.2 Resources and performance results

Data path only

Table 7.1 combines the IP resources utilization (LUTs, FFs and DSPs), the consumed power (static, dynamic range and maximum total power in mW), the number of hidden neurons and the power model evaluation metrics (MAE and MAPE). The power values shown in the table correspond to a single IP. They are obtained by dividing the power of multiple instances of the same IP by the number of its corresponding instances. The minimum dynamic power is

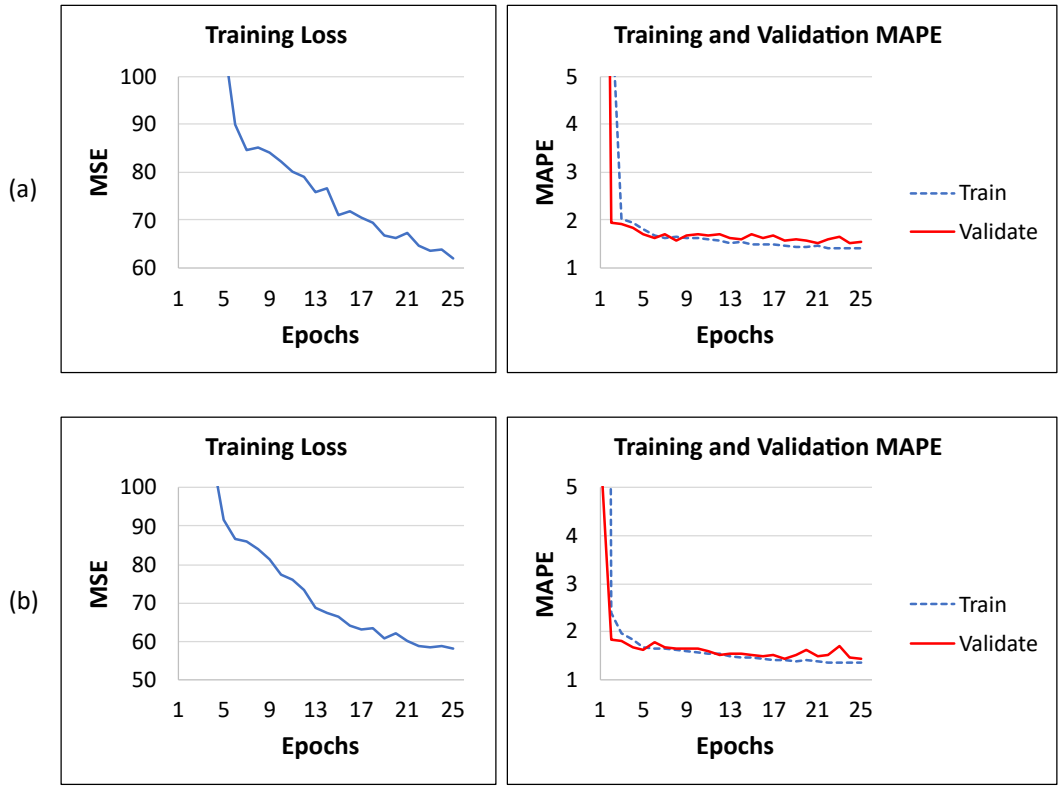

FIGURE 7.9: Black-box IP learning curves, showing training loss and MAPE v/s number of epochs.

recorded when the combination of [PLH, SR] results in low stimuli activity. The maximum dynamic power is detected when high stimuli activity occurs. The MAE values designate the absolute error in raw power levels ranging between 1 and 3, out of 1024 total levels (10-bit resolution ADC), while the MAPE is the estimation absolute percentage error ranging between 0.25% and 0.50%. Both MAE and MAPE metrics evaluate the average power consumption of a complete group, encapsulating multiple instances of the same IP.

As an observation, the proposed power modeling methodology is equally efficient on LUT- and DSP-based IP. The experimental results show fast and very accurate estimation values when applied to a variety of target IP components. This proves the robustness as well as the coherence of the presented neural network's architecture, and subsequently, the high adaptivity of the power consumption estimation model.

State machine and data path

Table 7.2 combines the data path functions, the IP resources utilization (LUTs, FFs and DSPs), the estimated average power consumption range and the power model evaluation metrics (MAE and MAPE). The MAE values designate the absolute error in raw power levels ranging between 0.85 and 1.55, out of 1024

TABLE 7.1: Performance results and power information.

		Multiplier	MAC	MAC (DSP)	OTU-DA
IP Utilization	LUT	69	87	6	18
	FF	16	32	16	36
	DSP	n/a	n/a	1	n/a
Avg. static (mW)		0.37	0.5	0.26	0.25
Min. dynamic (mW)		0.40	0.69	0.25	0.06
Max. dynamic (mW)		0.64	0.96	0.34	0.09
Max. total (mW)		1.01	1.46	0.60	0.34
Hidden neurons		8	16	8	16
Prediction MAE		2.76	3.01	1.57	1.17
Prediction MAPE		**0.45%**	**0.47%**	**0.40%**	**0.25%**

total levels (10-bits resolution), whereas the MAPE is the estimation absolute percentage error being under 1%. Figure 7.10 shows multiple random states for IP1 along with their respective measured and estimated power consumption levels in mW. Each permutation of the control signals, coupled with the input activity of the data path, yields to a functional state with a specific power level. For a given state, the power may vary following the data path input activity. For instance, a specific value of the control signals leads to a functional state with variable power levels depending on the data path input activity (PLH and SR). Subsequently, we may notice different average power values for the same functional state. The estimated power is represented by the maximum and minimum average levels following the $\pm$ percentage error.

TABLE 7.2: Resources and performance results.

		IP1	IP2
Data path 1: f_1, f_1'		$a^2 + b^2$ (Pythagorean formula)	$(2[a:b] \times (2[a:b] - 1))/2$ (Hexagonal numbers formula)
Data path 2: f_2, f_2'		$(a^2 + b^2) \oplus (a^2 - b^2)$ (Random formula)	$([a:b]([a:b]^2 + 1))/2$ (Magic square formula)
Resources	LUT	1361	128
	FF	25	25
	DSP	n/a	30
Min. avg. power		14 mW	14.13 mW
Max. avg. power		27.8 mW	29.65 mW
MAE		0.85	1.53
MAPE		**0.50%**	**0.84%**

Online estimation

Table 7.4 combines the IP usage (LUTs, FFs and DSPs), the estimated average power consumption range (minimum and maximum) and the power model evaluation metrics (MAE and MAPE) for 4 and 8 hidden neurons respectively.

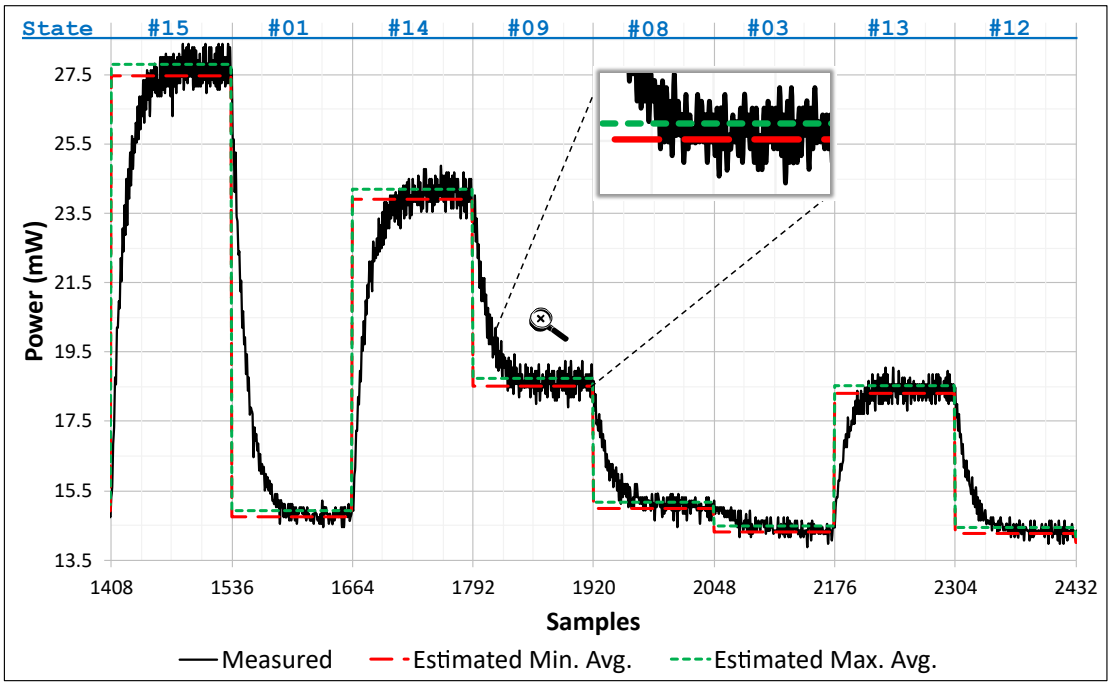

FIGURE 7.10: IP1 per-state measured v/s estimated power consumption.

The MAE values designate the absolute error in raw power levels ranging between 6 and 6.67, out of 1024 total levels (10-bit resolution). The MAPE is the estimation absolute percentage error being less than 1.5% for both cases. With the 8 hidden neurons model an estimation improvement of 16% is recorder over the 4-neuron alternative model. As an observation, the proposed power estimation methodology is highly efficient on LUT- and DSP-based IP circuits. The experimental results show fast and very accurate estimation values when applied to black-box IPs. This proves the robustness as well as the coherence of the presented neural network's architecture, and subsequently, the high adaptivity of the presented online power monitoring technique.

Figure 7.11 shows multiple unsorted operation modes for the black-box IP along with their respective measured and estimated power consumption levels in mW. Each permutation of the OM bits, coupled with the input activity of the data path, yields to a functioning mode with a specific power level. For instance, a specific value of the OM bits leads to an operation mode (spread over 128 samples) with variable power levels depending on the data path input activity (PLH and SR). Subsequently, we may notice different average power values for the same operation mode. The estimated power is represented by the maximum and minimum average levels following the $\pm$ prediction percentage error. In Figure 7.11 (a), two neighboring operating modes are detected with

TABLE 7.3: Resources and performance results.

		IP1	IP2
Data path 1: f_1, f_1'		$\frac{[a:b]^2+[a:b]+2}{2}$ (Central polygonal series)	$(a \times b + a^2) \oplus (a \times b - b^2)$ (Random)
Data path 2: f_2, f_2'		$\frac{3[a:b]^2-[a:b]}{2}$ (Pentagonal series)	$\frac{[a:b]^2-[a:b]}{2}$ (Triangular series)
Resources	LUT	132	909
	FF	25	25
	DSP	30	10
Min. avg. power		13.6 mW	14.5 mW
Max. avg. power		27.34 mW	32 mW
MAE		1.23	1
MAPE		**0.76%**	**0.58%**

TABLE 7.4: Resources and performance results for black-box IP.

		Black-box IP
IP utilization	LUT	14296
	FF	3761
	DSP	50
Min. avg. power (mW)		25.00
Max. avg. power (mW)		94.50
Estimation MAE	4 neurons	6.67
	8 neurons	6.00
Estimation MAPE	4 neurons	1.55%
	8 neurons	1.30%

very close power levels, yet the estimator was able to differentiate and to properly estimate the corresponding power consumption.

Fault detection: proof of concept

In order to evaluate the proposed fault detection methodology, it is essential to conduct experimental tests. Figure 7.12 (a) represents a black-box IP with input fault injection option through a meddler circuit. This latter is shown in Figure 7.12 (b) where a Free Running Counter (FRC) is optionally XORed with a number of inputs. Fault injection refers to a deliberate and controlled testing technique that can be used to evaluate fault detection methodologies. It involves intentionally introducing various types of faults, errors, or anomalies into a system's inputs. The random meddling of inputs in the context of fault injection, typically involves introducing unexpected or erroneous bits into a system without following a specific pattern. This randomness helps simulate real-world scenarios where inputs might deviate from expected norms due to

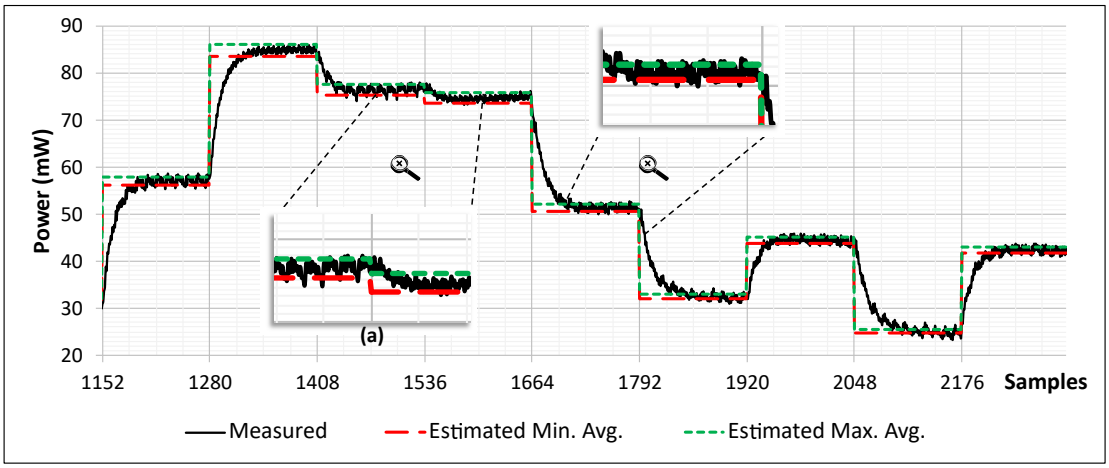

FIGURE 7.11: Black-box IP, per-mode measured v/s estimated power consumption.

various reasons, such as data corruption, communication errors, hardware malfunctions, etc. When enabled, the resulting output is a random sequence of bits. This randomness has a direct effect on the number of bit transitions and on the percentage of logic '1' bits in the affected fixed-length inputs; thus yielding to a totally different switching activity. The fault detection scanner senses variations in the switching activity characteristics (SR and PLH) as soon as the estimated power consumption falls outside of the characterized range enclosed within the power profile. This fault injection mechanism was selected due to its simplistic implementation yet its efficient and precise outcome on designated IP inputs.

Figure 7.13 reveals a normal IP behavior where all estimated power values fall within the expected thresholds extracted from the associated power profile. This is reflected on the inputs moving fault scores that are aligned with the graph underneath. For that purpose we have selected inputs 0, 2, 4 and 6. As an observation, all fault scores are almost zeros except upon operation mode change. In that case, a slight and negligible increase is detected during transition phases; the corresponding fault scores are represented in **bold**.

However, in Figure 7.14, intentional meddling is exercised on inputs 2 and 4 during operation mode $M(x)$. In this case the estimated power consumption is completely off the threshold limits triggering the detection algorithm that scans the inputs activity. Since inputs 2 and 4 were intentionally meddled, their activity is now outside the characterization limits and thus implying a noticeable increase in their respective fault score. As an observation, the saturation of the scores, denoted in **bold**, at inputs 2 and 4 indicate a totally faulty operation mode $M(x)$. Upon the switching to mode $M(y)$, the meddling at the inputs is disabled thus leading to a normal switching activity. A noticeable transition

phase is detected in the fault score values until a normal operation is restored again. This also demonstrates the system's ability to recover and resume orderly execution.

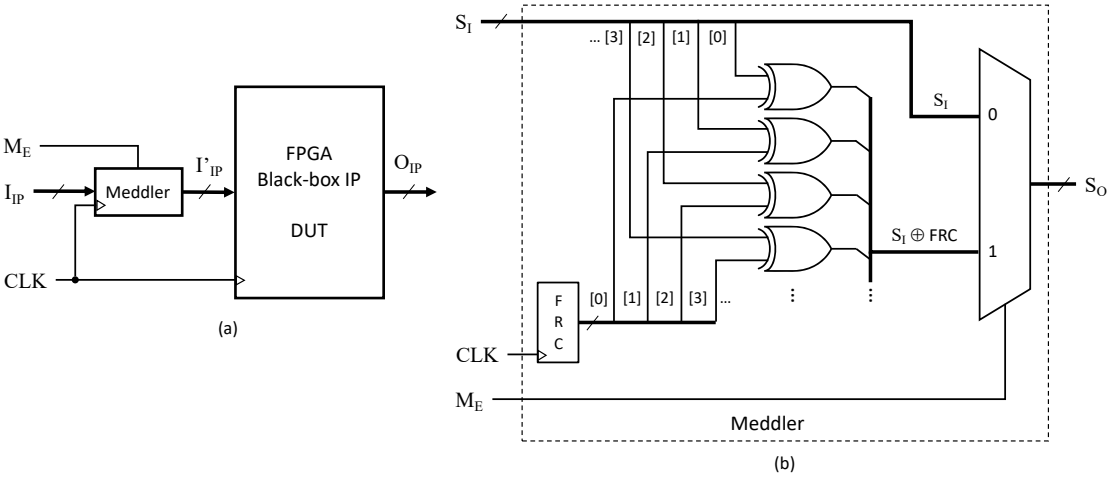

FIGURE 7.12: Fault injection mechanism into FPGA IP (a), using a meddler circuit (b).

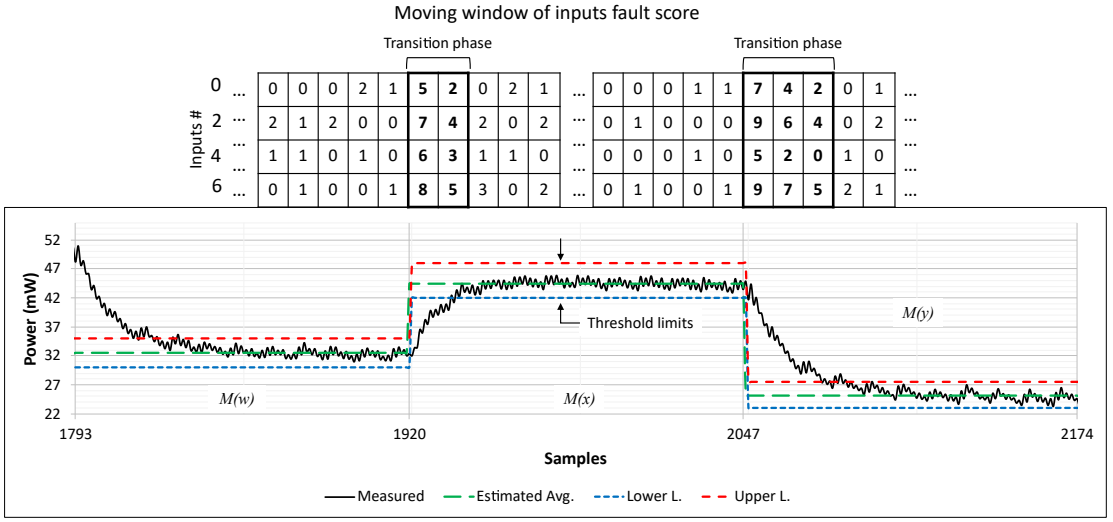

FIGURE 7.13: Normal moving window fault score.

To wrap up, the presented results outline a comprehensive experimental setup to evaluate a fault detection methodology. The use of a meddler circuit, controlled fault injection, power profiles and fault scores collectively contribute to understanding the methodology's effectiveness in detecting abnormal behavior induced by faults. The proposed methodology is tailored for detecting deviations in power consumption as well as input activities, which are indicative of faults. It is tuned through threshold settings, enclosed within the power profile, to achieve a desired balance between sensitivity and false positives.

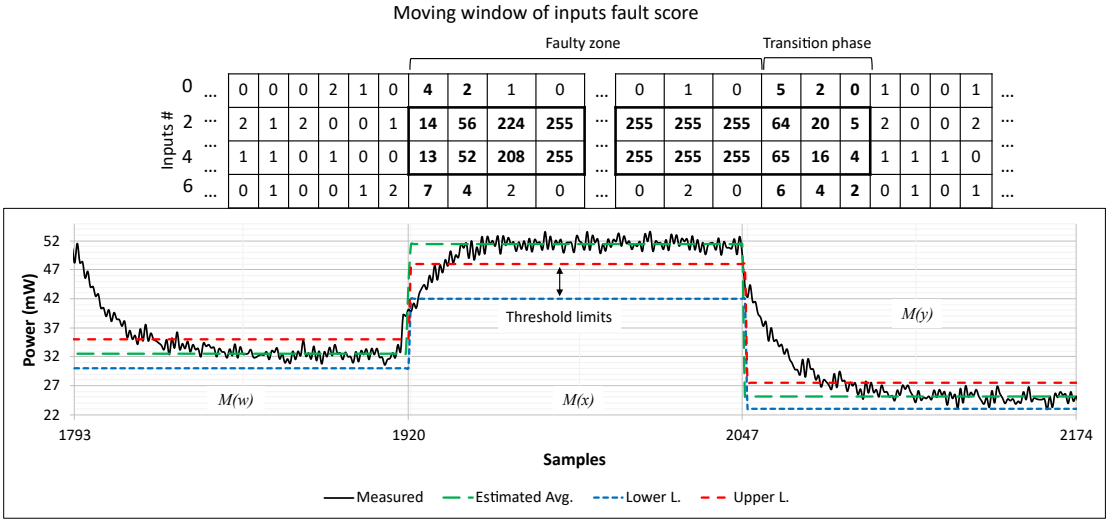

FIGURE 7.14: Abnormal moving window fault score.

7.4 Conclusion

This chapter offers a thorough examination of the validation process and the outcomes achieved throughout this research, effectively showcasing the efficiency and adaptivity of the proposed power estimation methodology. The experimental results, along with the assessment of the power modeling, affirm the accuracy and reliability of the estimated power consumption for various FPGA IP configurations, both in offline and online scenarios. The learning curves presented demonstrate the significant improvement in model performance as the training epochs increase, validating the effectiveness of the employed neural network architecture.

Furthermore, the analysis of resources and performance provides valuable insights into the utilization of FPGA IP resources, power consumption characteristics, and evaluation metrics. This detailed information enables digital circuit designers to make informed decisions and optimize energy consumption in their designs.

In summary, the proposed power estimation methodology based on machine learning and neural networks proves to be an efficient and reliable approach for estimating FPGA IP power consumption. The obtained results validate the effectiveness of this approach and pave the way for practical applications in the design and management of energy consumption in digital circuits. The aforementioned results show an absolute percentage error of $< 0.5\%$, $< 1\%$ and $< 2\%$ for the data path, the state machine and online power monitoring respectively. Moreover a fault detection methodology is presented based on the extension of the online power estimation alternative using power profiling technique. As a proof of concept, a controlled meddler circuit was introduced

in order to inject faulty bits in FPGA IP inputs thus leading to a change in its inputs activities. Subsequently, these latter, become out of the power profile characterized limits, triggering an increase in their corresponding fault scores. The proposed fault detection methodology can be investigated and developed as a possible hardware security reinforcement.

Chapter 8

Conclusion

The estimation of power consumption is a highly relevant and widely discussed subject among digital hardware designers. It plays a crucial role in rapidly exploring different design options during the conception process. Furthermore, it enables efficient prediction and precise management of power in real-time for a specific circuit. In our research, we capitalized on machine learning techniques to develop an innovative approach for estimating power consumption in Field-Programmable Gate Array (FPGA) Intellectual Properties (IPs). Our approach involves the creation of a novel neural network-based model that accurately predicts power consumption. By leveraging this method, we aim to enhance the efficiency and effectiveness of power management in FPGA IPs. We have presented the proposed power consumption model by first describing the measurement system used to collect power information from specific FPGA-based targets. Next, we have described the proposed power modeling methodology including the neural network model and data training. Finally we applied the proposed methodology on specific IP components and evaluated the experimental results for both offline and online domains. As a suggested application for the online alternative, we have proposed a fault detection algorithm based on the realtime power consumption monitoring and power profiling. The fault occurrence possibility was represented by a moving fault score window. Methodology validation and experimental results show an absolute percentage error of $< 0.5\%$, $< 1\%$ and $< 2\%$ for the data path, the state machine and online power monitoring respectively.

The presented work makes several contributions to the scientific community. Firstly, it provides an overview, classification, and comparison of various high-level techniques for modeling and estimating power consumption, including their specific applications. Secondly, a hardware characterization platform is developed for FPGA IPs, allowing for synchronized generation of stimuli signals and collection of aligned digital and analog data. Thirdly, an offline methodology is introduced, utilizing supervised machine learning to

estimate power consumption at a high level, with inputs limited to state machine control signals and data path input activity characteristics. Additionally, an automatic and efficient approach is devised to construct training data for high-level FPGA power modeling using measurements. Furthermore, a high-level online and in-situ power monitoring technique is presented, based on machine learning, which leverages significant operating modes and input activity specifications. Lastly, the online power monitoring approach is extended with a learning-based technique capable of detecting faults in FPGA IPs through power profiling.

8.1 Potential Future Work

Based on the presented work, there are several potential avenues for future work and expansion:

- Application to complex system-level designs: The current research focuses on IP-level power consumption estimation. Future work can explore extending the methodology to complex system-level designs incorporating multiple IPs and interconnects. This would involve developing techniques to model power consumption at the system level and considering interactions between different components to achieve more accurate power estimations.

- Exploration of alternative power modeling and estimation targets: The current research investigates power consumption modeling and estimation of FPGA IPs, however, a good alternative (or expansion) would be ASICs. Application-specific integrated circuits are widely used and usually consume less power than reconfigurable-based logic circuits.

- Exploration of alternative machine learning techniques: While the current research utilizes neural networks for power consumption estimation, future work can explore other machine learning techniques such as deep learning or ensemble methods. Comparative studies can be conducted to evaluate the effectiveness and suitability of these alternative approaches in the context of power consumption estimation for FPGA IPs.

- Extension of the fault detection algorithm: The current work tackles the possibility of detecting general FPGA IP faults caused by several factors. However the focus on the hardware security aspect, considered lately as a hot topic, would be of a great interest.

Overall, the proposed research provides a strong foundation for future advancements in power consumption estimation and management in FPGA IPs. By addressing these potential areas of expansion, researchers can further refine and extend the methodology, ultimately leading to more coverage and broader space exploration.

Appendix A

Publications

A.1 Journals

A.1.1 Published

1. "High-Level Power Estimation Techniques in Embedded Systems Hardware: an Overview"
   M. Richa, JC. Prévotet, M. Dardaillon, M. Mroué and AE. Samhat.
   Published in: Springer Nature, The Journal of Supercomputing - September 2022 - DOI: 10.1007/s11227-022-04798-5

A.1.2 Under Review

1. "Learning-Based Online Power Monitoring Usage for FPGA IP Fault Detection"
   M. Richa, M. Mroué, JC. Prévotet, M. Dardaillon, and AE. Samhat.
   Submitted to: Springer Nature, The Journal of Systems Architecture.

A.2 International Conferences

A.2.1 Published

1. "An Automated and Centralized Data Generation and Acquisition System"
   M. Richa, JC. Prévotet, M. Dardaillon, M. Mroué and AE. Samhat.
   Published in: 2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS) - January 2021 - Dubai, United Arab Emirates.
   DOI: 10.1109/ICECS53924.2021.9665490

2. "High-Level Early Power Estimation of FPGA IP Based on Machine Learning"
   M. Richa, JC. Prévotet, M. Dardaillon, M. Mroué and AE. Samhat.

Published in: 2022 29th IEEE International Conference on Electronics, Circuits and Systems (ICECS) - December 2022 - Glasgow, United Kingdom. DOI: 10.1109/ICECS202256217.2022.9970952

3. "Automated Training Data Construction using Measurements for High-Level Learning-Based FPGA Power Modeling"
   M. Richa, JC. Prévotet, M. Dardaillon, M. Mroué and AE. Samhat.
   Published in: 2022 International Conference on Smart Systems and Power Management (IC2SPM) - December 2022 - Beirut, Lebanon.

   DOI: 10.1109/IC2SPM56638.2022.9988835

4. "High-Level Online Power Monitoring of FPGA IP Based on Machine Learning"
   M. Richa, JC. Prévotet, M. Dardaillon, M. Mroué and AE. Samhat.
   Published in: Springer Nature Switzerland - Workshop on Design and Architectures for Signal and Image Processing, in conjunction with the 18th HiPEAC Conference 2023 - Toulouse, France. DOI: 10.1007/978-3-031-29970-4_9

Appendix B

Hardware Schematics and Photos

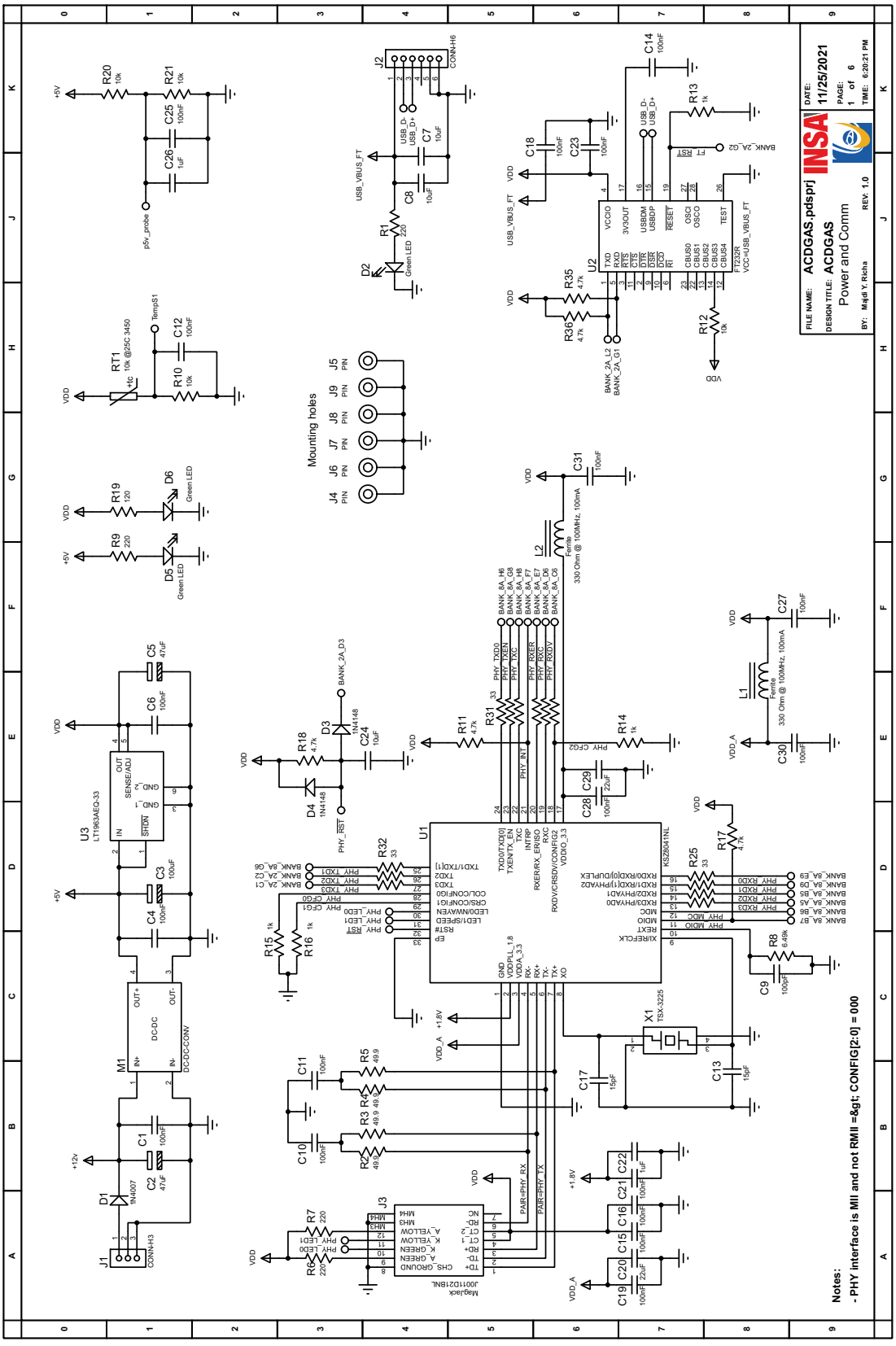

FIGURE B.1: ACDGAS schematics page 1: power and communication.

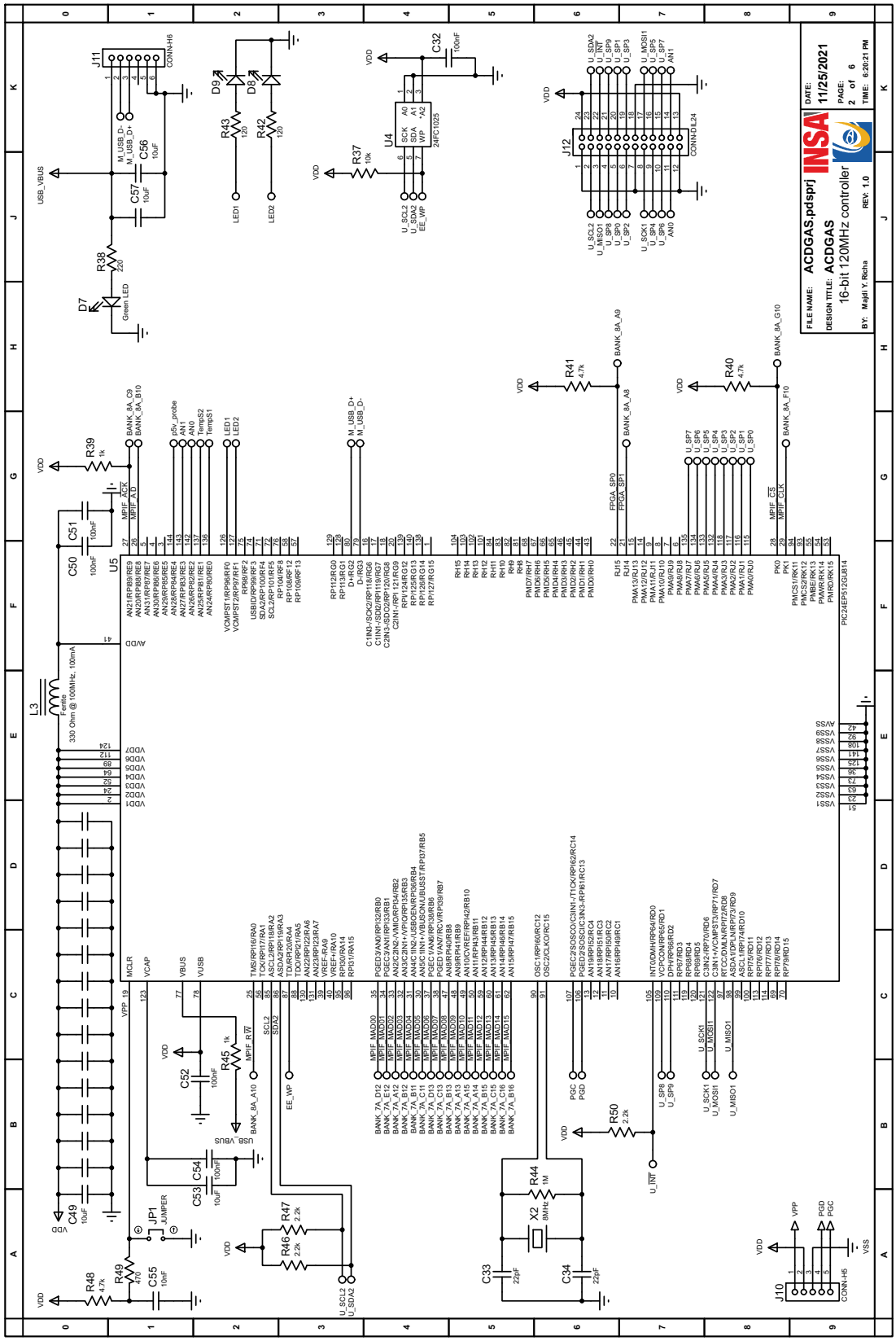

FIGURE B.2: ACDGAS schematics page 2: main controller.

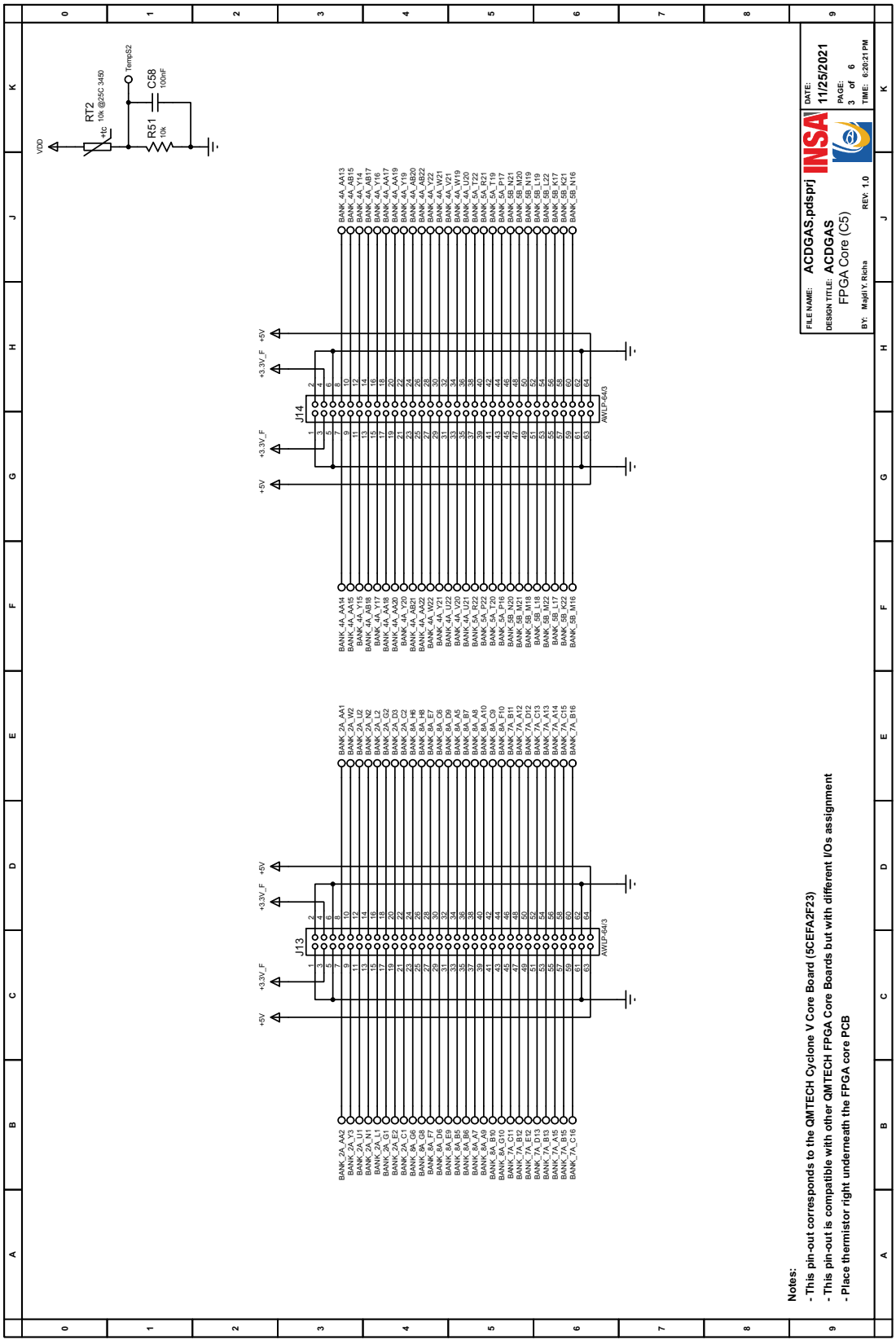

FIGURE B.3: ACDGAS schematics page 3: FPGA core socket interface.

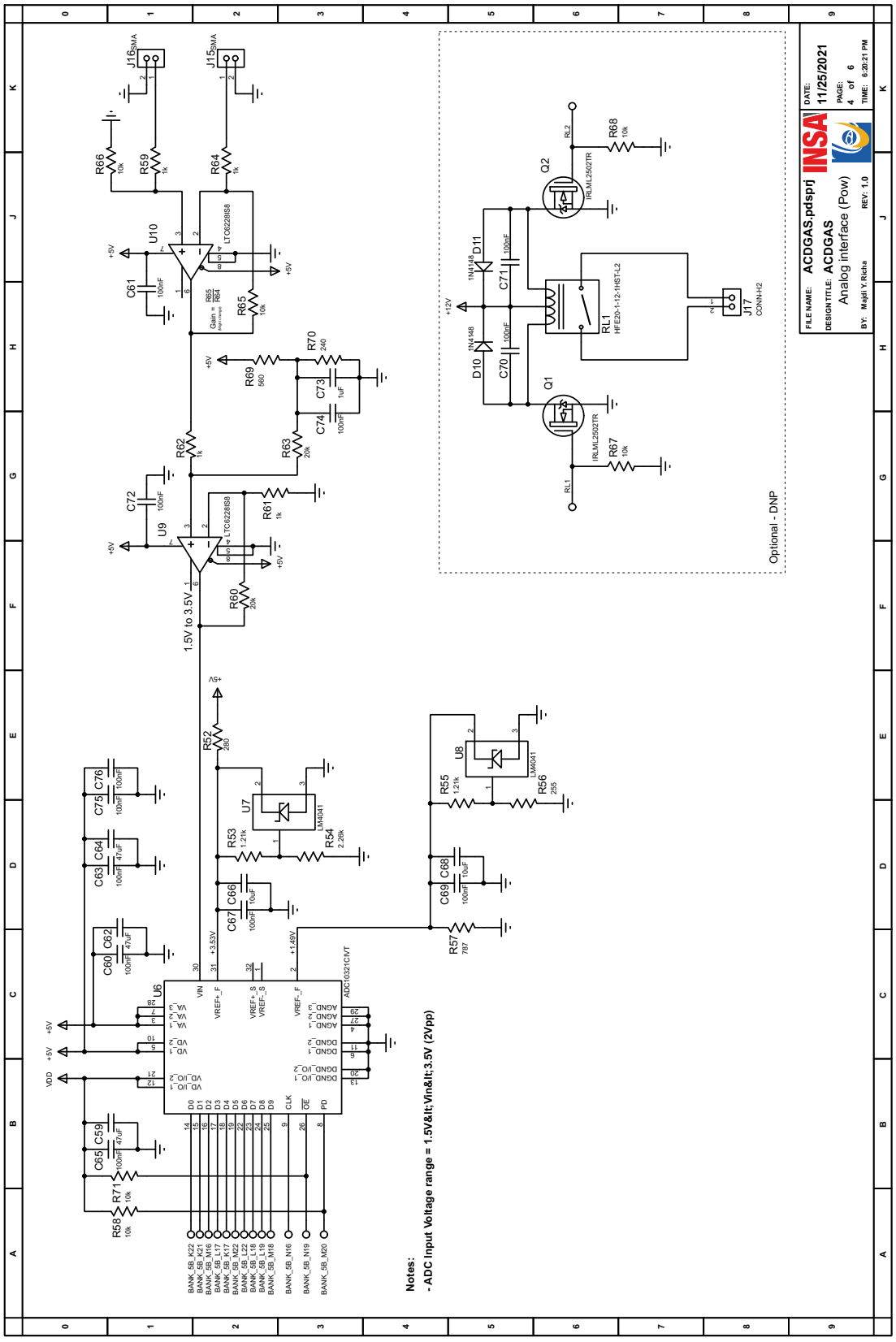

FIGURE B.4: ACDGAS schematics page 4: differential analog channel.

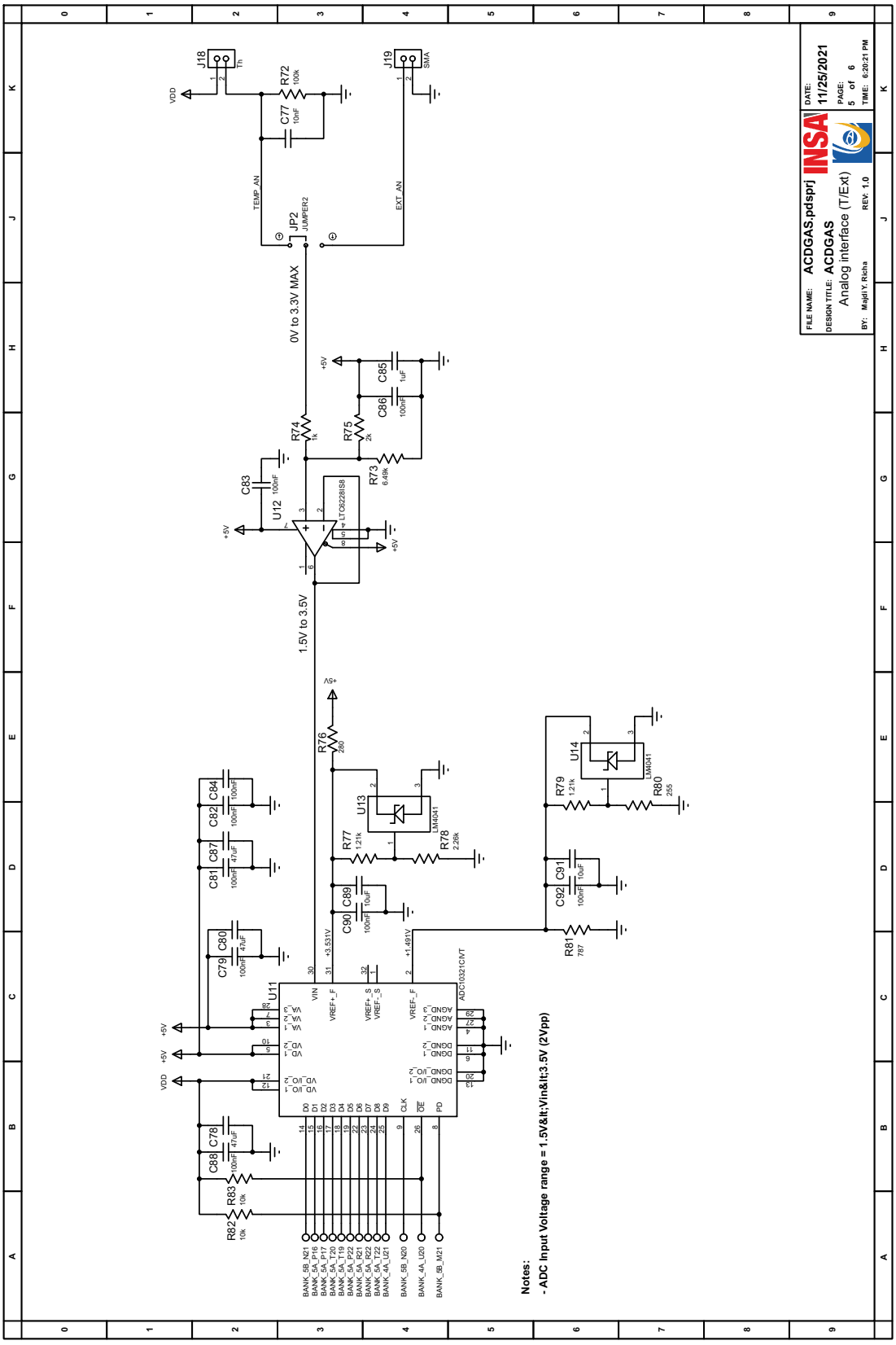

FIGURE B.5: ACDGAS schematics page 5: single-ended auxiliary analog channel.

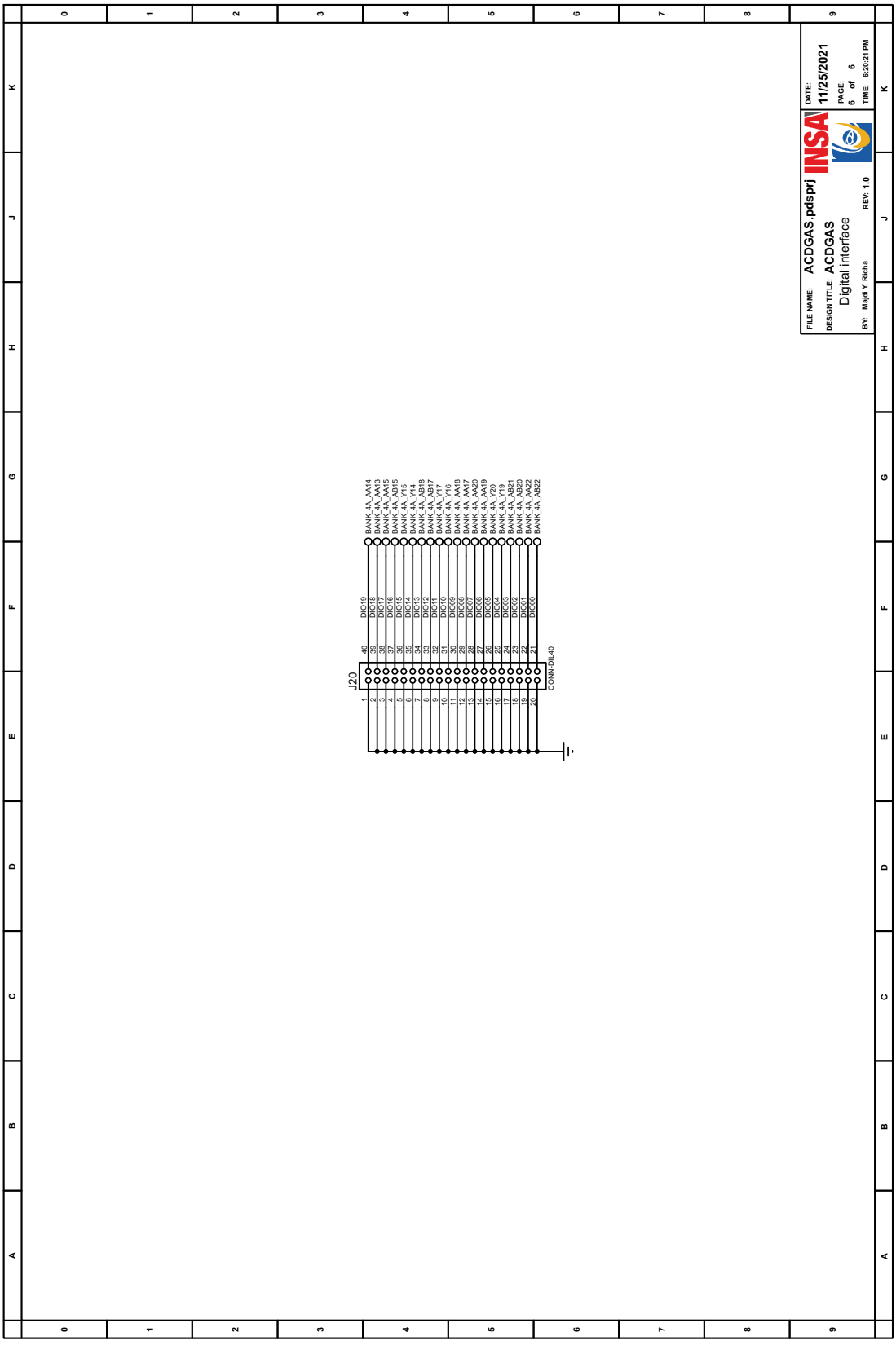

FIGURE B.6: ACDGAS schematics page 6: HSDIO interface.

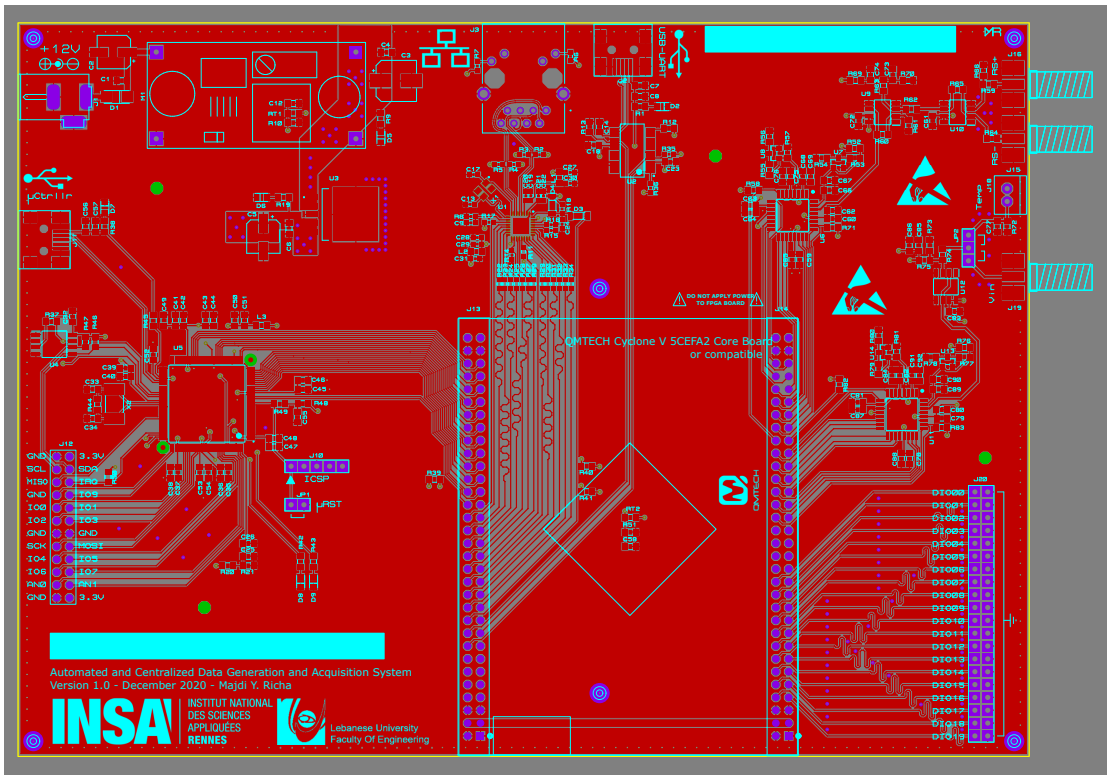

FIGURE B.7: Characterization platform (ACDGAS) 2D 4-layer PCB.

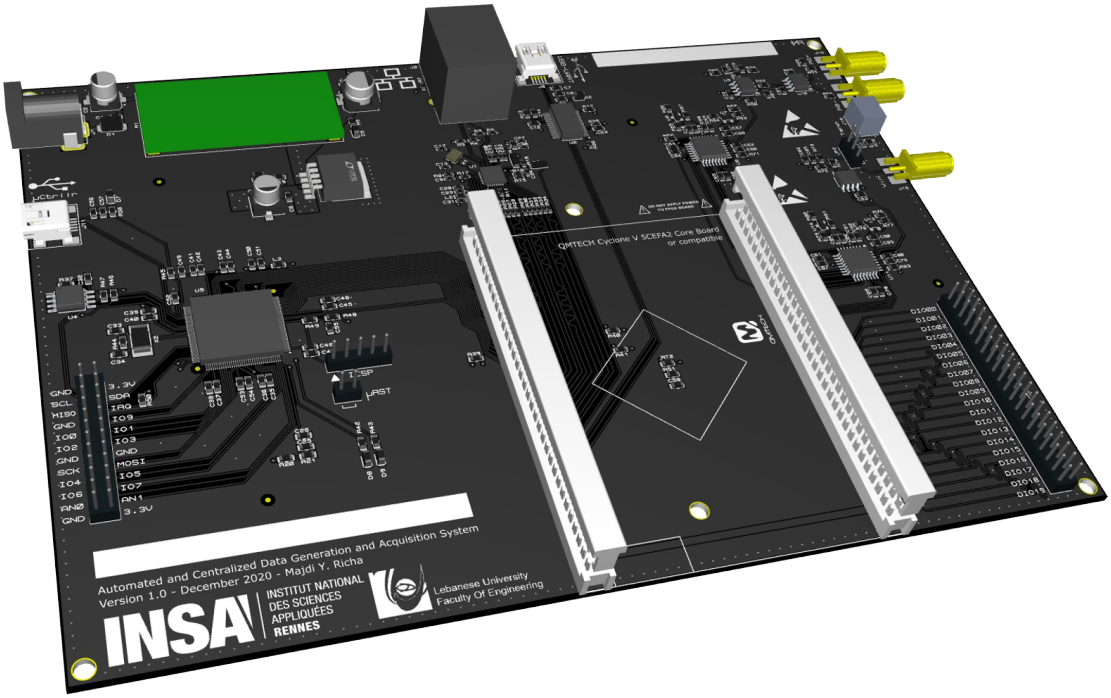

FIGURE B.8: Characterization platform (ACDGAS) 3D-generated PCBA.

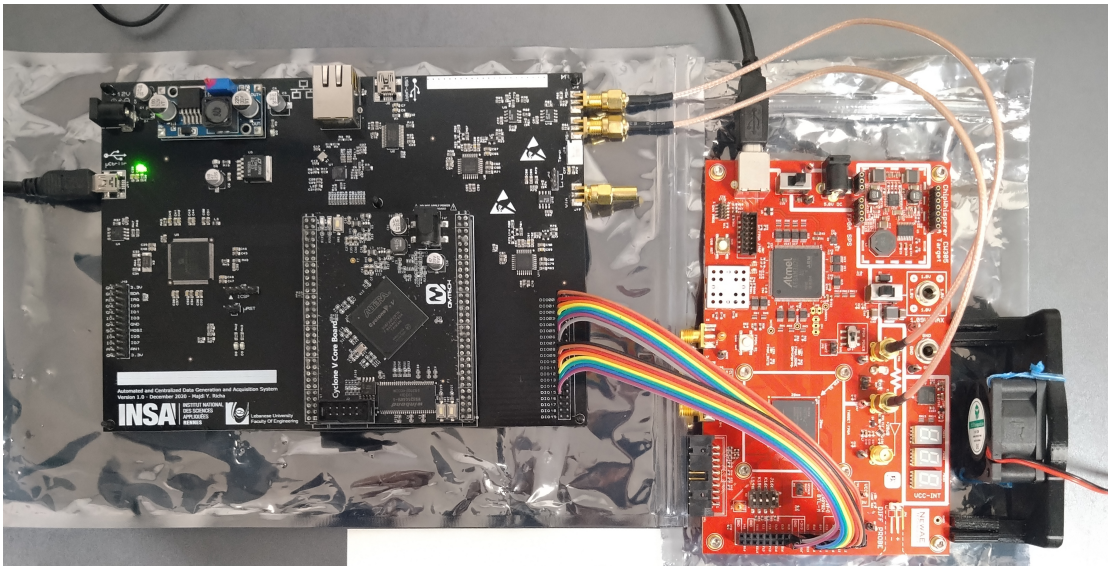

FIGURE B.9: Measurement platform (left) and FPGA DUT (right).

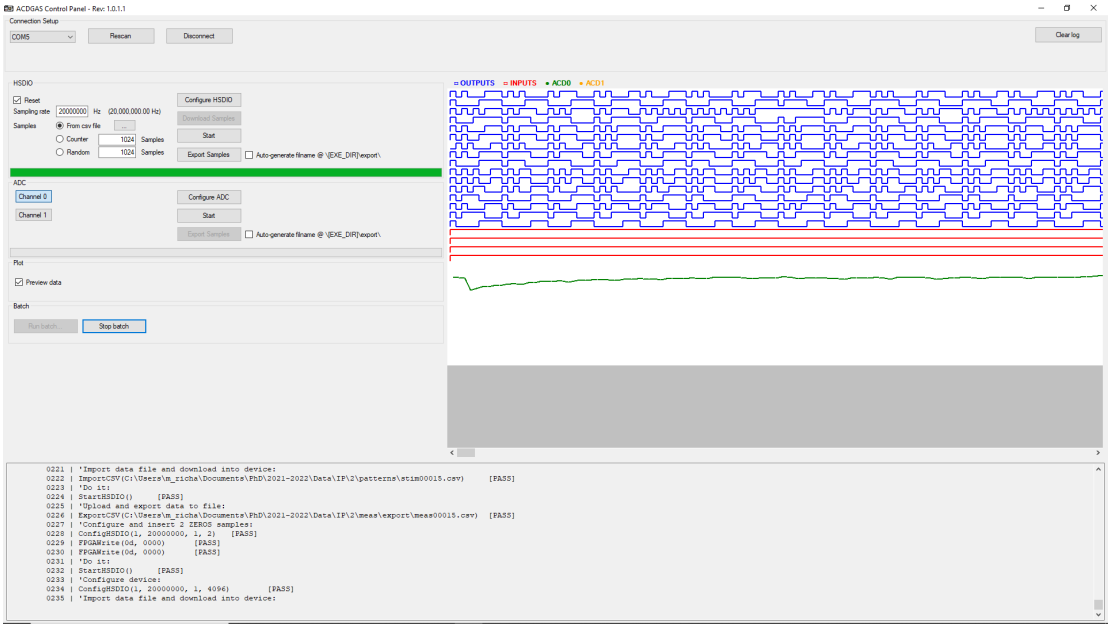

FIGURE B.10: Characterization platform (ACDGAS) Software GUI.

Bibliography

[1] Ghani Abbas and Khurram Kazi. "Enhanced Optical Transport Network standards". In: *7th International Symposium on High-capacity Optical Networks and Enabling Technologies*. 2010, pp. 22–25. DOI: 10.1109/HONET.2010.5715757.

[2] Shereen Moataz Afifi, François Verdier, and Cécile Belleudy. "Power estimation method based on real measurements for processor-based designs on FPGA". In: *Proceedings - 2014 International Conference on Computational Science and Computational Intelligence, CSCI 2014* 2 (2014), pp. 260–263. DOI: 10.1109/CSCI.2014.133.

[3] Sumit Ahuja et al. "Power estimation methodology for a high-level synthesis framework". In: *Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009* (2009), pp. 541–546. DOI: 10.1109/ISQED.2009.4810352.

[4] Gokhan Akgun, Muhammad Ali, and Diana Gohringer. "Power-Aware Computing Systems on FPGAs: A Survey". In: *31st International Conference on Field-Programmable Logic and Applications (FPL)* (2021), pp. 45–51. DOI: 10.1109/FPL53798.2021.00016.

[5] Tejaswini Ananthanarayana, Sonia Lopez, and Marcin Lukowiak. "Power analysis of HLS-designed customized instruction set architectures". In: *Proceedings - 2017 IEEE 31st International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2017* (2017), pp. 207–212. DOI: 10.1109/IPDPSW.2017.59.

[6] Tero Arpinen et al. "MARTE profile extension for modeling dynamic power management of embedded systems". In: *Journal of Systems Architecture* 58.5 (2012), pp. 209–219. ISSN: 13837621. DOI: 10.1016/j.sysarc.2011.01.003. URL: http://dx.doi.org/10.1016/j.sysarc.2011.01.003.

[7] Mostafa Bazzaz, Mohammad Salehi, and Alireza Ejlali. "An accurate instruction-level energy estimation model and tool for embedded systems". In: *IEEE Transactions on Instrumentation and Measurement* (2013). ISSN: 00189456. DOI: 10.1109/TIM.2013.2248288.

[8] Pedro P. Carballo et al. “Scalable video coding deblocking filter FPGA and ASIC implementation using high-level synthesis methodology”. In: *Proceedings - 16th Euromicro Conference on Digital System Design, DSD 2013* (2013), pp. 415–422. DOI: 10.1109/DSD.2013.52.

[9] Cosmin Cernazanu-Glavan et al. “Energy profiling of FPGA designs”. In: *2014 IEEE International Symposium on Robotic and Sensors Environments (ROSE) Proceedings*. 2014, pp. 118–123. DOI: 10.1109/ROSE.2014.6953034.

[10] Anu Chalil. “Implementation of power estimation methodology for intellectual property at SoC level”. In: *Proceedings of the 2nd International Conference on Communication and Electronics Systems, ICCES 2017* 2018-January (2018), pp. 1010–1013. DOI: 10.1109/CESYS.2017.8321234.

[11] Nidhi Chandoke and Ashish Kumar Sharma. “A novel approach to estimate power consumption using SystemC transaction level modelling”. In: *12th IEEE International Conference Electronics, Energy, Environment, Communication, Computer, Control: (E3-C3), INDICON 2015* (2016), pp. 1–6. DOI: 10.1109/INDICON.2015.7443519.

[12] Chompoo Inwai Chow and Jade Mungkornassawakul. “A smart recording power analyzer prototype Using LabVIEW and low-cost data acquisition (DAQ) in being a smart renewable monitoring system”. In: *IEEE Green Technologies Conference* (2013), pp. 49–56. ISSN: 21665478. DOI: 10.1109/GreenTech.2013.16.

[13] Jason Cong et al. “High-level synthesis for FPGAs: From prototyping to deployment”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 30.4 (2011), pp. 473–491. ISSN: 02780070. DOI: 10.1109/TCAD.2011.2110592.

[14] Pawel Piotr Czapski and Andrzej Sluzek. “A survey on system-level techniques for power reduction in field programmable gate array (FPGA)-based devices”. In: *Proceedings - 2nd Int. Conf. Sensor Technol. Appl., SENSORCOMM 2008, Includes: MESH 2008 Conf. Mesh Networks; ENOPT 2008 Energy Optim. Wireless Sensors Networks, UNWAT 2008 Under Water Sensors Systems* (2008), pp. 319–328. DOI: 10.1109/SENSORCOMM.2008.39.

[15] James J. Davis et al. “KAPow: High-accuracy, low-overhead online per-module power estimation for FPGA designs”. In: *ACM Transactions on Reconfigurable Technology and Systems* (2018). ISSN: 19367414. DOI: 10.1145/3129789.

[16] Vijay Degalahal and Tim Tuan. "Methodology for high level estimation of FPGA power consumption". In: *Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC* 1 (2005), pp. 657–660. DOI: 10.1145/1120725.1120986.

[17] Chinmay Deshpande et al. "A Configurable and Lightweight Timing Monitor for Fault Attack Detection". In: *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 2016, pp. 461–466. DOI: 10.1109/ISVLSI.2016.123.

[18] Božidar R. Dimitrijević and Milan M. Simić. "Virtual instrumentation software applied to integrated circuit testing procedure". In: *2010 27th International Conference on Microelectronics, MIEL 2010 - Proceedings* Miel (2010), pp. 353–356. DOI: 10.1109/MIEL.2010.5490465.

[19] Yaseer A. Durrani, Ana Abril, and Teresa Riesgo. "Efficient power macro-modeling technique for IP-based digital system". In: *Proceedings - IEEE International Symposium on Circuits and Systems* (2007), pp. 1145–1148. ISSN: 02714310. DOI: 10.1109/ISCAS.2007.378252.

[20] Dalia El-dib et al. "Automated FPGA Power Characterization Methodology". In: *International Journal of Electrical and Computer Sciences IJECS-IJENS* 16.2 (2016).

[21] Pavel Fexa and Josef Vedral. "Developing automated data acquisition system for ADC and DAC testing". In: *Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS'2011* 1.September (2011), pp. 39–42. DOI: 10.1109/IDAACS.2011.6072707.

[22] Tomas Fryza and Miroslav Waldecker. "Precise Measurement of Power Consumption and Execution Time for Embedded Platforms". In: *International Conference on Systems, Signals, and Image Processing* 2018-June (2018), pp. 1–4. ISSN: 21578702. DOI: 10.1109/IWSSIP.2018.8439486.

[23] Crescenzio Gallo. "Artificial neural networks". In: *Artificial Neural Networks* January 2015 (2011), pp. 1–426. ISSN: 19310145. DOI: 10.4324/9781315154282-3.

[24] David Greaves and Mehboob Yasin. "TLM POWER3: Power estimation methodology for SystemC TLM 2.0". In: *Lecture Notes in Electrical Engineering* 265 LNEE (2014), pp. 53–68. ISSN: 18761119. DOI: 10.1007/978-3-319-01418-0-4.

[25] Domenik Helms et al. "Leakage Models for High-Level Power Estimation". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37.8 (2018), pp. 1627–1639. ISSN: 02780070. DOI: 10.1109/TCAD.2017.2760519.

[26] Kurt Hesse. "Using PMBus for Improved System-Level Power Management". In: *Texas Instruments Seminars* (2008), p. 19. URL: https://www.ti.com/download/trng/docs/seminar/Topic_6_Hesse.pdf.

[27] Muhammad Irfan, Shahid Masud, and Muhammad Adeel Pasha. "Development of a High Level Power Estimation Framework for Multicore Processors". In: *Proceedings of 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference, IMCEC 2018*. 2018. ISBN: 9781538618035. DOI: 10.1109/IMCEC.2018.8469473.

[28] Ruzica Jevtic and Carlos Carreras. "Power measurement methodology for FPGA devices". In: *IEEE Transactions on Instrumentation and Measurement* 60.1 (2011), pp. 237–247. ISSN: 00189456. DOI: 10.1109/TIM.2010.2047664.

[29] Lorandel Jordane, Jean-Christophe Prévotet, and Maryline Hélard. "Fast Power and Energy Efficiency Analysis of FPGA-based Wireless Baseband Processing". In: *HIP3ES, Prague, Czech Republic* (2016). arXiv: 1601.00834. URL: http://arxiv.org/abs/1601.00834.

[30] Behnam Khaleghi et al. "FPGA Energy Efficiency by Leveraging Thermal Margin". In: *IEEE International Conference on Computer Design (ICCD)* (2019). arXiv: 1911.07187. URL: http://arxiv.org/abs/1911.07187.

[31] Johannes Knodtel et al. "A Novel Methodology for Evaluating the Energy Consumption of IP Blocks in System-Level Designs". In: *2018 28th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)* (2018), pp. 46–53. DOI: 10.1109/PATMOS.2018.8464149. URL: https://ieeexplore.ieee.org/document/8464149/.

[32] Ajay Krishna Ananda Kumar and Andreas Gerstlauer. "Learning-based CPU power modeling". In: *2019 ACM/IEEE 1st Workshop on Machine Learning for CAD, MLCAD 2019* (2019). DOI: 10.1109/MLCAD48534.2019.9142100.

[33] Pratyush Kumar and Lothar Thiele. "System-level power and timing variability characterization to compute thermal guarantees". In: *IEEE/ACM/IFIP International Conference on Hardware/software codesign and system synthesis (CODES+ISSS)* (2011), p. 179. DOI: 10.1145/2039370.2039400. URL: http://dl.acm.org/citation.cfm?doid=2039370.2039400.

[34] Nanda Kumar Lakkoju, Sateesh Gudla, and Bhanu Sridhar Mantravadi. "AVR-USB data acquisition". In: *3rd International Conference on Electronics Computer Technology* (2011), pp. 35–39. DOI: 10.1109/ICECTECH.2011.5941555. URL: https://dx.doi.org/10.1109/ICECTECH.2011.5941555.

[35] Avinash Lakshminarayana, Sumit Ahuja, and Sandeep Shukla. "High Level Power Estimation Models for FPGAs". In: *2011 IEEE Computer Society Annual Symposium on VLSI* (2011), pp. 7–12. ISSN: 2159-3469. DOI: 10.1109/ISVLSI.2011.79.

[36] Julien Lamoureux and Wayne Luk. "An overview of low-power techniques for field-programmable gate arrays". In: *Proceedings of the 2008 NASA/ESA Conference on Adaptive Hardware and Systems, AHS 2008* (2008), pp. 338–345. DOI: 10.1109/AHS.2008.71.

[37] Najeem Lawal, Fahad Lateef, and Muhammad Usman. "Power consumption measurement & configuration time of FPGA". In: *2015 Power Generation Systems and Renewable Energy Technologies, PGSRET 2015* (2015), pp. 1–5. DOI: 10.1109/PGSRET.2015.7312250.

[38] Hugo Lebreton and Pascal Vivet. "Power modeling in SystemC at Transaction Level, application to a DVFS architecture". In: *Proceedings - IEEE Computer Society Annual Symposium on VLSI: Trends in VLSI Technology and Design, ISVLSI 2008*. 2008, pp. 463–466. ISBN: 9780769531700. DOI: 10.1109/ISVLSI.2008.71.

[39] Dongwook Lee and Andreas Gerstlauer. "Learning-Based, Fine-Grain Power Modeling of System-Level Hardware IPs". In: *ACM Transactions on Design Automation of Electronic Systems* 23.3 (2018). DOI: 10.1145/3177865. URL: https://dl.acm.org/doi/10.1145/3177865.

[40] Dongwook Lee et al. "Learning-based power modeling of system-level black-box IPs". In: *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2015. URL: https://ieeexplore-ieee-org.rproxy.insa-rennes.fr/document/7372659/.

[41] Dongwook Lee et al. "Learning-Based Power Modelling of System-Level Black-Box IPs". In: *IEEE/ACM International Conference on Computer-Aided Design* (2015), pp. 847–853.

[42] Hyung Gyu Lee, Sungyuep Nam, and Naehyuck Chang. "Cycle-accurate energy measurement and high-level energy characterization of FPGAs". In: *Proceedings - International Symposium on Quality Electronic Design, ISQED* 2003-Janua (2003), pp. 267–272. ISSN: 19483295. DOI: 10.1109/ISQED.2003.1194744.

[43] Fei Li and Lei He. "Power Modeling and Characteristics of Field Programmable Gate Arrays". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 24 (2005), pp. 1–13.

[44] Hao Liang et al. "Hierarchical Library Based Power Estimator for Versatile FPGAs". In: *Proceedings - IEEE 9th International Symposium on Embedded Multicore/Manycore SoCs, MCSoC 2015* (2015), pp. 25–32. DOI: 10.1109/MCSoC.2015.44.

[45] Zhe Lin, Sharad Sinha, and Wei Zhang. "An Ensemble Learning Approach for In-Situ Monitoring of FPGA Dynamic Power". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38.9 (2019), pp. 1661–1674. ISSN: 19374151. DOI: 10.1109/TCAD.2018.2859248. arXiv: 2009.01432.

[46] Zhe Lin et al. "HL-Pow: A Learning-Based Power Modeling Framework for High-Level Synthesis". In: *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2020, pp. 574–580. ISBN: 978-1-7281-4123-7. DOI: 10.1109/ASP-DAC47756.2020.9045442. URL: https://ieeexplore-ieee-org.rproxy.insa-rennes.fr/stamp/stamp.jsp?tp={\&}arnumber=9045442.

[47] R. Ludewig et al. "Power estimation based on transition activity analysis with an architecture precise rapid prototyping system". In: *Proceedings of the International Workshop on Rapid System Prototyping* 2002-January (2002), pp. 138–143. ISSN: 10746005. DOI: 10.1109/IWRSP.2002.1029749.

[48] Jeremy Mange. "Effect of Training Data Order for Machine Learning". In: *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*. 2019, pp. 406–407. DOI: 10.1109/CSCI49370.2019.00078.

[49] Jeremy Mange. "Effect of Training Data Order for Machine Learning". In: *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*. 2019, pp. 406–407. DOI: 10.1109/CSCI49370.2019.00078.

[50] David C. Marcu and Cristian Grava. "The impact of activation functions on training and performance of a deep neural network". In: *2021 16th International Conference on Engineering of Modern Electric Systems (EMES)*. 2021, pp. 1–4. DOI: 10.1109/EMES52337.2021.9484108.

[51] Marius Marcu et al. "An investigation on FPGA based energy profiling of multi-core embedded architectures". In: *2016 IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. 2016, pp. 720–723. DOI: 10.1109/ICECS.2016.7841303.

[52] Jan Marjanovic. "Low vs high level programming for FPGA". In: *Proceedings of the 7th International Beam Instrumentation Conference, IBIC 2018* (2018), pp. 527–533. DOI: 10.18429/JACoW-IBIC2018-thoa01. URL: http://jacow.org/IPAC2018/papers/thob03.pdf.

[53] Ramesh Medar, Vijay S. Rajpurohit, and B. Rashmi. "Impact of Training and Testing Data Splits on Accuracy of Time Series Forecasting in Machine Learning". In: *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*. 2017, pp. 1–6. DOI: 10.1109/ICCUBEA.2017.8463779.

[54] Matthieu Moy et al. "Modeling Power Consumption and Temperature in TLM Models To cite this version : HAL Id : hal-01339441 Modeling Power Consumption and Temperature in TLM Models". In: *Leibniz Transactions on Embedded Systems* 3.3 (2016), pp. 0–29.

[55] Mohamad Najem et al. "Method for dynamic power monitoring on FPGAs". In: *Conference Digest - 24th International Conference on Field Programmable Logic and Applications, FPL 2014* (2014). DOI: 10.1109/FPL.2014.6927457.

[56] Yehya Nasser and Jordane Lorandel. "RTL to Transistor Level Power Modelling and Estimation Techniques for FPGA and ASIC : A Survey". In: *IEEE TCAD Circuits and Systems* (), pp. 1–15.

[57] Yehya Nasser, Jean Christophe Prévotet, and Maryline Hélard. "Power modeling on FPGA: A neural model for RT-level power estimation". In: *2018 ACM International Conference on Computing Frontiers, CF 2018 - Proceedings* 1 (2018), pp. 309–313. DOI: 10.1145/3203217.3204462.

[58] Yehya Nasser et al. "Neu Pow: Artificial Neural Networks for Power and Behavioral Modeling of Arithmetic Components in 45nm ASICs Technology". In: *ACM International Conference on Computing Frontiers 2019, CF 2019 - Proceedings* (2019), pp. 183–189. DOI: 10.1145/3310273.3322820.

[59] Yehya Nasser et al. "NeuPow: A CAD Methodology for High-level Power Estimation Based on Machine Learning". In: *ACM Transactions on Design Automation of Electronic Systems* 25.5 (2020). DOI: 10.1145/3388141. URL: https://doi.org/10.1145/3388141.

[60] Simone Orcioni et al. "Energy estimation in SystemC with Powersim". In: *Integration, the VLSI Journal* 55 (2016), pp. 118–128. ISSN: 01679260. DOI: 10.1016/j.vlsi.2016.04.006. URL: http://dx.doi.org/10.1016/j.vlsi.2016.04.006.

[61] Juan Manuel Paniego et al. "Unified Power Modeling Design for Various Raspberry Pi Generations Analyzing Different Statistical Methods". In: *Communications in Computer and Information Science*. 2020. ISBN: 9783030483241. DOI: 10.1007/978-3-030-48325-8-4.

[62] Murilo R. Perleberg et al. "ASIC power-estimation accuracy evaluation: A case study using video-coding architectures". In: *9th IEEE Latin American Symposium on Circuits and Systems, LASCAS 2018 - Proceedings* (2018), pp. 1–4. DOI: 10.1109/LASCAS.2018.8399919.

[63] Roberta Piscitelli and Andy Pimentel. "A high-level power model for MPSoC on FPGA". In: *IEEE Computer Architecture Letters* 11.1 (2012), pp. 13–16. ISSN: 15566056. DOI: 10.1109/L-CA.2011.24.

[64] Sherief Reda and Abdullah N. Nowroz. "Power modeling and characterization of computing devices: A survey". In: *Foundations and Trends in Electronic Design Automation* (2012). ISSN: 15513939. DOI: 10.1561/1000000022.

[65] Axel Reimer, Arne Schulz, and Wolfgang Nebel. "Modelling macromodules for high-level dynamic power estimation of FPGA-based digital designs". In: *Proceedings of the International Symposium on Low Power Electronics and Design* 2006 (2006), pp. 151–154. ISSN: 15334678. DOI: 10.1145/1165573.1165609.

[66] Mohsen Riahi Alam, Mostafa Ersali Salehi Nasab, and Sied Mehdi Fakhraie. "Power Efficient High-Level Synthesis by Centralized and Fine-Grained Clock Gating". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 34.12 (2015), pp. 1954–1963. ISSN: 02780070. DOI: 10.1109/TCAD.2015.2445734.

[67] Majdi Richa et al. "An Automated and Centralized Data Generation and Acquisition System". In: *28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*. 2021, pp. 1–4. ISBN: 9781728182810. DOI: 10.1109/ICECS53924.2021.9665490. URL: https://ieeexplore.ieee.org/document/9665490/.

[68] Majdi Richa et al. "High-Level Online Power Monitoring of FPGA IP Based on Machine Learning". In: *Design and Architecture for Signal and Image Processing*. Springer Nature Switzerland, 2023, pp. 107–119. ISBN: 978-3-031-29970-4. DOI: 10.1007/978-3-031-29970-4_9.

[69] Majdi Richa et al. "High-level power estimation techniques in embedded systems hardware: an overview". In: *The Journal of Supercomputing* 79.4 (2023), pp. 3771–3790. ISSN: 1573-0484. DOI: 10.1007/s11227-022-04798-5.

[70] Michel Rogers-Vallée et al. "IP characterization methodology for fast and accurate power consumption estimation at transactional level model". In: *Proceedings - IEEE International Conference on Computer Design: VLSI in Computers and Processors* (2010), pp. 534–541. ISSN: 10636404. DOI: 10.1109/ICCD.2010.5647622.

[71] Falk Schellenberg et al. "An inside job: Remote power analysis attacks on FPGAs". In: *2018 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 2018, pp. 1111–1116. DOI: 10.23919/DATE.2018.8342177.

[72] Dmitriy Shorin and Armin Zimmermann. "Formal description of an approach for power consumption estimation of embedded systems". In: *2014 24th International Workshop on Power and Timing Modeling, Optimization and Simulation, PATMOS 2014*. 2014. ISBN: 9781479954124. DOI: 10.1109/PATMOS.2014.6951890.

[73] Martin Streubühr et al. "ESL power and performance estimation for heterogeneous MPSOCS using SystemC". In: *Forum on Specification and Design Languages*. 2011. ISBN: 9782953050448.

[74] Ankush Varma et al. "Accurate and fast system-level power modeling: An XScale-based case study". In: *ACM Trans. Embed. Comput. Syst.* 7.3 (2008), pp. 1–20. ISSN: 15399087. DOI: 10.1145/1347375.1347378.

[75] Lijia Wang et al. "High-level power estimation model for SOC with FPGA prototyping". In: *Proceedings - 4th International Conference on Computational Intelligence and Communication Networks, CICN 2012* (2012), pp. 491–495. DOI: 10.1109/CICN.2012.124.

[76] Robert V. White. "PMBus: A Decade of Growth: An open-standards success". In: *IEEE Power Electronics Magazine* 1.3 (2014), pp. 33–39. DOI: 10.1109/MPEL.2014.2330492.

[77] Xiaohong Yang et al. "Application of high accuracy data acquisition in power plant". In: *Proceedings - 3rd International Symposium on Information Science and Engineering, ISISE 2010* (2010), pp. 11–14. DOI: 10.1109/ISISE.2010.59.

[78] Qian Yi. "FPGA Implementation of Neural Network Accelerator". In: *2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)*. 2018, pp. 1903–1906. DOI: 10.1109/IMCEC.2018.8469659.

[79] Jianwang Zhai et al. “McPAT-Calib: A RISC-V BOOM Microarchitecture Power Modeling Framework”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2022), pp. 1–1. ISSN: 0278-0070. DOI: 10.1109/tcad.2022.3169464. URL: https://ieeexplore.ieee.org/document/9761982/.

[80] Zijun Zhang. “Improved Adam Optimizer for Deep Neural Networks”. In: *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. 2018, pp. 1–2. DOI: 10.1109/IWQoS.2018.8624183.

[81] Yuan Zhou et al. “PRIMAL: Power Inference using Machine Learning”. In: *2019 56th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2019. URL: https://ieeexplore-ieee-org.rproxy.insa-rennes.fr/document/8806775/.

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Titre : Modélisation de la consommation d'énergie dans le matériel des systèmes embarqués.

Mots clés : Modélisation et estimation de la puissance, apprentissage machine, réseau neuronal artificiel, IP FPGA.

Résumé : L'optimisation de la puissance est devenue une préoccupation majeure pour la plupart des concepteurs de systèmes numériques, en particulier dans les premières phases de conception et surtout dans les systèmes à énergie limitée (appareils portables fonctionnant sur batterie, modules enfichables électro-optiques, systèmes IoT et énergies vertes, etc.). Par conséquent, l'estimation précoce de la consommation d'énergie au moment de la conception est devenue cruciale pour l'optimisation de la puissance. Cette recherche couvre plusieurs sujets liés à l'optimisation de la puissance des circuits numériques, notamment les circuits reconfigurables FPGA. Dans un premier temps, un bref aperçu des facteurs de consommation d'énergie, des techniques d'optimisation énergétique et des approches d'estimation de puissance de bas niveau est présenté. Une vue d'ensemble des techniques d'estimation de puissance de haut niveau actuellement disponibles, ainsi qu'une comparaison détaillée entre différentes méthodologies et leurs applications, sont ensuite présentées en détails. Ensuite, nous développons la méthodologie proposée de modélisation et d'estimation de puissance basée sur l'apprentissage des FPGA IP. Ces travaux ciblent à la fois les domaines hors-ligne et en-ligne. Ces derniers incluent également le système automatisé de génération et d'acquisition de données ainsi qu'une approche détaillée et automatique de construction des ensembles de données d'entraînement. Pour le mode hors-ligne, nous estimons la consommation d'énergie en fonction des signaux de contrôle de la machine à états et de l'activité d'entrée du chemin de données. Pour le mode en-ligne, nous estimons en temps réel la consommation d'énergie en fonction de ses modes de fonctionnement les plus significatifs et de son activité d'entrée. L'application proposée pour l'alternative en ligne implique un algorithme de détection de panne qui repose sur la surveillance en temps-réel de la consommation d'énergie et le profilage de la puissance. Une fenêtre de score est utilisée pour représenter la possibilité d'occurrence de pannes. La validation de la méthodologie et les résultats expérimentaux montrent une erreur absolue en pourcentage inférieure à 0,5%, 1% et 2% respectivement pour le chemin de données, la machine à états et la surveillance de puissance en ligne.

Title: Power Consumption Modeling in Embedded Systems Hardware

Keywords: Power modeling and estimation, machine learning, artificial neural network, FPGA IP

Abstract: Power optimization has become a major concern for most digital hardware designers, particularly in early design phases and especially in limited power budget systems (battery-operated hand-held devices, electro-optical pluggable modules, IoT and green energy systems, etc.). Subsequently, early power consumption estimation at design time is crucial for power optimization. This research covers multiple topics serving the digital circuits power optimization notably FPGAs. Initially, a short overview on power consumption factors, energy optimization techniques and low-level power estimation approaches is briefly covered. An overview of High-Level power estimation techniques currently available along with a comprehensive comparison between different methodologies and their applications on estimated models is thoroughly presented. Then, we elaborate on the proposed learning-based power modeling and estimation methodology of FPGA IPs targeting both offline and online domains. This covers also the automated data generation and acquisition system along with a detailed and automatic training data sets construction approach. For the offline mode, we estimate the power consumption based on the state machine control signals and the input activity of the data path. For the online counterpart, we estimate in situ and in real-time the power consumption based on its most significant modes of operation and its input activity. The proposed application for the online alternative involves a fault detection algorithm that relies on real-time power consumption monitoring and power profiling. A moving fault score window is used to represent the possibility of fault occurrences. Methodology validation and experimental results show an absolute percentage error of <0.5%, <1% and <2% for the data path, the state machine and online power monitoring respectively.