Exploring learning techniques for edge AI taking advantage of non-volatile memories

Michele Martemucci

université de BORDEAUX

THÈSE PRÉSENTÉE POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE

SCIENCES PHYSIQUES ET DE L'INGÉNIEUR

ÉLECTRONIQUE

Par **Michele MARTEMUCCI**

Exploration de techniques d'apprentissage pour l'edge AI exploitant des mémoires non-volatiles

Sous la direction de **Sylvain SAÏGHI**

Soutenue le 9 décembre 2024

Membres du jury :

M. Ian O'CONNOR	Rapporteur
Professeur des universités, École Centrale de Lyon – University of Lyon	
Mme. Melika PAYVAND	Rapporteure
Professor, University of Zurich – ETH Zurich (SUISSE)	
M. Sylvain SAÏGHI	Directeur de thèse
Professeur, Université de Bordeaux	
M. Jean-Michel PORTAL	Président
Professeur, Aix-Marseille Université	
M. Simon JEANNOT	Examinateur
Ingénieur de recherche, STMicroelectronics (Grenoble)	
M. Adrien F. VINCENT	Examinateur
Maître de conférences, Bordeaux INP	

Membres invités :

M. François RUMMENS	Co-encadrant
Ingénieur de recherche, CEA, LIST, Université Grenoble Alpes	
Mme. Elisa VIANELLO	Co-encadrante
Ingénieure de recherche, CEA, LETI, Université Grenoble Alpes	

Exploration de techniques d'apprentissage pour l'edge AI exploitant des mémoires non-volatiles

Résumé: Les systèmes embarqués d'intelligence artificielle (IA) capables d'apprentissage nécessitent à la fois des capacités d'inférence et d'apprentissage, tout en étant écoénergétiques. Cependant, aucune technologie de mémoire existante n'intègre pleinement toutes les caractéristiques souhaitées pour ces systèmes. Les mémoires résistives – memristors – sont idéales pour l'inférence, mais souffrent d'une endurance limitée et d'une consommation énergétique élevée lors de la programmation. En revanche, les condensateurs ferroélectriques (FeCAPs) sont adaptés pour l'apprentissage, mais leur processus de lecture, qui détruit la donnée stockée, les rend inadaptés à l'inférence. Un même dispositif à base d'hafnia peut être optimisé pour fonctionner soit comme un FeCAP, soit comme un memristor, selon ses conditions d'exploitation. Un tel dispositif à double usage a été réalisé au cours de cette thèse en intégrant un film de 10 nm d'oxyde d'hafnium dopé au silicium, recouvert d'une couche de titane, entre deux couches de métallisation d'une technologie CMOS du nœud 130 nm.
Un circuit intégré utilisant cette technologie de mémoire hybride a été validé expérimentalement. Il inclut à la fois des FeCAPs et des memristors dans le même back-end-of-line du procédé CMOS 130 nm, créant un réseau hybride de FeCAPs et de memristors interconnectés. Cette matrice hybride permet un transfert direct de données depuis plusieurs cellules FeCAP vers un seul memristor, sans circuit intermédiaire. Cette technologie est compatible avec un entraînement par descente de gradient stochastique de réseaux de neurones artificiels embarqués. Les FeCAPs stockent des poids de plus grande précision pour l'apprentissage, tandis que les memristors stockent des poids analogiques pour l'inférence et l'apprentissage. Les poids cachés (FeCAPs) sont mis à jour pour chaque échantillon, tandis que les poids analogiques (memristors) sont mis à jour tous les k entrées via le processus de transfert. Deux versions d'implémentation de ce système ont été proposées. La première, utilisant des FeCAPs de même taille pour stocker des poids de plus haute précision et un memristor pour le poids analogique, a des capacités d'apprentissage limitées. La seconde implémentation utilise des FeCAPs de différentes tailles pour coder un poids entier de 10 bits, tandis que deux memristors encodent des poids analogiques positifs et négatifs dans leur conductance différentielle. Cette méthode atteint une précision élevée tout en réduisant considérablement la consommation énergétique par poids lors de la programmation. Le nombre d'opérations reste bien en dessous des limites d'endurance des memristors et des FeCAPs. L'évaluation de la robustesse aux erreurs, évaluée en fonction des mesures de transfert analogique, montre des performances comparables aux modèles logiciels à précision flottante sur plusieurs benchmarks.
Pour aller plus loin, un second circuit, plus modulable, a été conçu et fabriqué en technologie CMOS 22 nm pour explorer les avantages de la combinaison des dispositifs de mémoire ferroélectrique et résistive pour l'inférence et l'apprentissage dans les réseaux de neurones binaires (BNNs). Ce design utilise des matrices de mémoire ferroélectrique pour stocker des poids cachés de haute précision, au format entier ou flottant, avec des largeurs de bits de 8 bits ou multiples. Les mémoires résistives, stockant les poids binaires en configuration différentielle à travers deux dispositifs, sont utilisés pour effectuer du calcul proche-mémoire pour l'implémentation de la multiplication matrice-vecteur dans les BNNs. Les simulations électriques du circuit atteignent une consommation énergétique de multiplication binaire d'environ 100 fJ, avec un potentiel de réduction à quelques dizaines de fJ. Plusieurs stratégies algorithmiques sont envisagées en utilisant ce design de circuit, ouvrant la voie à des dispositifs embarqués capables d'apprentissage pour des applications d'IA.

Mots-clés : Mémoire non-volatile, Réseaux de neurones artificiels, IC design

Exploring learning techniques for edge AI taking advantage of non-volatile memories

Abstract: Learning-capable edge artificial intelligence (AI) systems require both inference and learning capabilities combined with energy efficiency. However, no existing memory technology fully integrates all the desirable features for these systems. Resistive memory – Memristor – arrays are ideal for AI inference but suffer from limited endurance and high programming energy. In contrast, ferroelectric capacitors (FeCAPs) are effective for learning, but their data-destructive read process makes them incompatible for inference. The same hafnium-based device can be optimized to function either as a FeCAP or as a memristor, depending on its operating conditions. Such a dual-use device was developed during this thesis by integrating a 10 nm film of silicon-doped hafnium oxide, with a titanium oxygen-scavenging layer, between two metal layers of a 130 nm CMOS process.

An application-specific integrated circuit using this hybrid memory technology was experimentally validated. It includes both FeCAPs and memristors in the back-end-of-line of foundry 130 nm CMOS, creating a hybrid array of interconnected FeCAPs and memristors. This hybrid array enables direct data transfer from multiple FeCAP cells to a single memristor device without intermediate circuits. This technology is compatible with on-chip training of artificial neural networks with stochastic gradient descent. FeCAPs store higher-precision weights for training, while memristors store analog weights for both inference and training. Hidden weights (FeCAPs) are updated for each sample, while analog weights (memristors) are updated every k inputs via the transfer process. Two system implementations were proposed. The first, using equally sized FeCAPs to store higher-precision weights and a single memristor for the analog weight, has limited learning capabilities. The second implementation uses FeCAPs with different areas to encode a 10-bit integer higher-precision weight, while two memristors encode positive and negative analog weights through their differential conductance. This method achieves high accuracy while significantly reducing energy consumption per weight during programming. The number of operations remains well below the endurance limits of memristors and FeCAPs. The evaluation of error robustness, assessed based on analog transfer measurements, shows performance comparable to floating-point precision software models across several benchmarks.

To go further, a second, more flexible circuit was designed and manufactured in a 22 nm CMOS technology to explore the benefits of combining ferroelectric and resistive memory devices for inference and training in binarized neural networks (BNNs). This design utilizes ferroelectric memory arrays to store higher-precision hidden weights in either integer or floating-point formats, with bit-widths of 8 bits or multiples thereof. Resistive memories, which store binary weights in a differential configuration across two devices, are used to perform near-memory computation for implementing matrix-vector multiplication in BNNs. Electrical simulations of the circuit achieve an energy consumption of approximately 100 fJ for binary multiplication, with the potential for reduction to a few tens of fJ. Several algorithmic strategies are envisioned using this circuit design, paving the way for devices capable of learning for edge AI applications.

Keywords: Non-volatile memory, Artificial neural networks, IC design

---

Unité de recherche
Laboratoire IMS, UMR 5218,
CNRS, Université de Bordeaux, Bordeaux INP, 33400 Talence, France.

Acknowledgements

The research presented in this thesis would not have been possible without the invaluable contributions, guidance, and support of numerous individuals and institutions. I am deeply grateful to everyone who played a role in helping me bring this work to fruition.

My sincere thanks go to my PhD supervisors, François Rummens and Elisa Vianello, for selecting me and supporting me throughout these three years. François, I learned a lot from your experience as an analog designer. Thank you for being a constant source of support whenever I encountered a problem and for your guidance. It was a pleasure sharing the office with you. Elisa, this project would not have existed without your intuition and encouragement. Thank you for always pushing me to achieve the best possible results while giving me the space and time to work at my own pace. I could not have asked for better supervisors, and I will miss working with both of you.
My deepest gratitude also goes to my PhD director, Sylvain Saïghi. Sylvain, thank you for always taking the time to discuss my research results and for managing the logistics of my PhD defense. I am grateful we had the opportunity to work together despite the distance and to meet in both Grenoble and Bordeaux.
Thanks are also due to Adrien Vincent for his valuable participation in our discussions and for offering insightful contributions throughout the course of this project. Your perspectives and feedback have been instrumental in refining various aspects of this work.
I would also like to express my gratitude to Damien Querlioz. I truly appreciated your insights and expertise throughout our collaboration. Your contributions to the publications have been invaluable, significantly enhancing the quality of the research.
This thesis was developed at CEA LETI and CEA LIST. I would like to express my sincere appreciation to both institutions for their excellent facilities, which were vital to the research presented in this thesis.
From the CEA LIST side, I would like to thank Alexandre Valentian for welcoming me to the LSTA lab and supporting this project. My gratitude goes to everyone in the lab, with special thanks to the "AI team": François Rummens, Thomas Mesquida, Yannick Malot, Paul Donsez Carod, Ivan Miro Panades, Lilian Billod, as well as former members Manon Dampfhoffer and Stefan Burel. I deeply appreciated your help to the development of this work and all the fruitful discussions. I would also like to thank the PhD students from LSTA for participating in the student meetings and Adrian Evans for organizing them. Sharing ideas from each other's work has greatly deepened my understanding of the field of microelectronics as a whole.
From the CEA LETI side, I would like to thank Tifenn Hirtzilin for participating in the development of the hybrid memory circuit concept. Thanks are also due to the PhD students (some already alumni) of the "neuromorphic team" at CEA LETI: Simone D'Agostino, Filippo Moro, Djohan Bonnet, and Tarcisius Januel, for their valuable discussions and

help. My gratitude also goes to the "memory team" for developing and manufacturing the memory technology presented in this thesis, as well as for providing the initial memory macro designs for further development. In particular, I would like to thank Laurent Grenouillet, Simon Martin, Julie Laguerre, Olivier Billoint, and Yann Beillard for their direct support with this project. I would also like to acknowledge the people from the characterization labs at CEA LETI for their assistance in the electrical characterization of the hybrid memory technology and circuit, as well as for their help in developing the test printed circuit board for the FeЯNet chip.
Finally, special thanks to my friends who stood by me throughout this journey. I am grateful for your encouragement, kindness, and patience. I am deeply grateful to my family, whose support has been a source of strength throughout these years. To my parents, thank you for your love and sacrifices. To my sister, thank you for your unwavering support in every aspect of my life.

Résumé

Introduction

La mise en œuvre des réseaux de neurones artificiels (ANN) est divisée en deux phases : l'apprentissage et l'inférence. La phase d'apprentissage consiste à modifier un ensemble de paramètres du réseau, les poids synaptiques, selon un algorithme d'apprentissage, afin de les faire converger vers des valeurs permettant au réseau d'accomplir la tâche pour laquelle il est entraîné avec une précision suffisamment élevée. La phase d'inférence consiste à appliquer le modèle appris précédemment à de nouvelles données d'entrée pour accomplir la tâche. Ainsi, les poids synaptiques sont modifiés plusieurs fois au cours de la phase d'apprentissage pour converger vers un ensemble optimal de valeurs pour la tâche souhaitée, tandis qu'ils restent fixes lors de l'opération d'inférence. Le développement relativement récent et les résultats remarquables des ANN sont dus à la construction de gigantesques bases de données et à des innovations algorithmiques nécessitant d'importantes ressources matérielles, ce qui entraîne des consommations d'énergie tout aussi importantes. Comme l'intelligence artificielle (IA) est de plus en plus intégrée dans divers objets connectés, allant des implants médicaux aux voitures autonomes, il est clair que les solutions algorithmiques et matérielles disponibles dans les centres de données ne pourront pas couvrir tous les besoins d'intégration de l'IA.
Le traitement des données de l'IA, en particulier l'apprentissage, dans les architectures de calcul de type von Neumann nécessite des échanges d'informations importants entre la mémoire et l'unité de calcul. Le débit limité, c'est-à-dire la quantité de données transférées par unité de temps, entre les accès à la mémoire et le calcul, est connu sous le nom de "von Neumann bottleneck". Cette problématique devient de plus en plus pertinente avec l'avancée des nœuds technologiques CMOS.
S'inspirant de la co-localisation du calcul et du stockage dans le cerveau biologique, les architectures de calcul en mémoire (IMC) ont émergé comme une solution économe en énergie pour le traitement des modèles ANN. Ces architectures exploitent un niveau élevé de parallélisme dans le calcul pour diminuer la complexité temporelle de certaines opérations et les déplacements de données. Ici, le calcul est effectué directement dans les matrices de mémoire, les dispositifs mémoires se comportant comme des synapses artificielles. Combinées à des dispositifs de mémoire non volatile (NVM), les solutions IMC peuvent considérablement améliorer l'efficacité énergétique des systèmes d'IA, offrant des opportunités notamment dans les systèmes embarqués.
Parmi les solutions NVM, les memristors sont particulièrement prometteurs pour permettre la prochaine génération de matériel IMC, car ils offrent potentiellement une meilleure efficacité énergétique, des temps d'accès plus rapides et une plus grande densité par rapport aux dispositifs de stockage traditionnels. Néanmoins, ils souffrent de plusieurs non-

idéalités, limitant actuellement leur fiabilité globale. Bien que les technologies memristives à l'état de l'art aient démontré une précision d'inférence proche de celle des logiciels dans de nombreux domaines de l'IA, leur exploitation dans des systèmes capables d'apprentissage reste un défi. En effet, les exigences de mémoire pour l'entraînement sont plus exigeantes, car les processus d'apprentissage dans les ANN nécessitent un raffinement itératif précis des forces synaptiques dans le réseau.
Les dispositifs de mémoire à accès aléatoire résistive (RRAM), également appelés memristors filamentaires, et les mémoires à accès aléatoire ferroélectrique (FeRAM) apparaissent comme des candidats appropriés pour développer un système d'apprentissage sur puce. En particulier, la RRAM est une technologie de mémoire résistive, c'est-à-dire stockant l'information dans la conductance du dispositif, qui peut être contrôlée par des impulsions électriques et reste inchangée si l'alimentation est coupée. De plus, les dispositifs RRAM peuvent être utilisés comme cellules multi-niveaux, car plusieurs états de conductance peuvent être réglés dans le même dispositif. D'autre part, la technologie FeRAM repose sur la possibilité de changer la polarisation de dipôles électriques semi-permanents à l'intérieur d'un matériau ferroélectrique par des impulsions électriques dans deux directions opposées. Ainsi, les dispositifs FeRAM se présentent comme des dispositifs intrinsèquement binaires. La polarisation reste inchangée si l'alimentation est coupée.
L'endurance virtuellement infinie en lecture des dispositifs RRAM et leur faible endurance en écriture les rendent adaptés aux applications d'inférence uniquement, tandis que la grande endurance en écriture des dispositifs FeRAM permettrait effectivement de déplacer l'apprentissage sur puce. Finalement, la migration de l'inférence et de l'apprentissage des centres de données vers les dispositifs embarqués leur permettra de s'adapter à l'évolution des données d'entrée, de spécialiser chaque appareil à son utilisateur, de conserver les données privées et d'offrir un service plus rapide.
Ainsi, les objectifs de cette thèse sont d'étudier la compatibilité de différentes pistes algorithmiques pour l'apprentissage des réseaux neuronaux avec les caractéristiques des technologies de mémoire ferroélectrique et résistive développées au CEA LETI et les contraintes matérielles de l'électronique embarquée, afin de produire sur silicium un circuit de démonstration combinant dispositifs NVM et technologies CMOS.

Résultats

La fabrication de deux technologies NVM différentes sur le même substrat, bien que potentiellement bénéfique pour l'implémentation matérielle des réseaux de neurones artificiels, se fait au détriment d'un coût de fabrication élevé et d'une intégration complexe. En revanche, le même empilemnet de matériaux peut être optimisée pour fonctionner comme FeRAM ou RRAM dans des conditions de fonctionnement différentes, dans le cas des dispositifs à base de HfO_2.
Cela a été réalisé en intégrant un film d'oxyde de hafnium dopé au silicium de 10 nm avec une couche d'absorption d'oxygène en titane dans un procédé CMOS de 130 nm. Cette empilement mémoire combine une couche d'oxyde de hafnium cristallisée dans la phase orthorhombique – nécessaire pour la commutation ferroélectrique – avec une couche d'absorption d'oxygène – nécessaire pour une commutation résistive fiable.
La mémoire hybride a été testée dans deux configurations : dans des matrices FeRAM et dans des matrices RRAM fabriquées dans la back-end-of-line (BEOL) de la technologie

CMOS de 130 nm. En tant que FeRAMs, ces dispositifs ont montré qu'ils fonctionnent comme des mémoires binaires avec une bonne endurance sur plus de 10 millions de cycles et une faible énergie de programmation, inférieure à 200 fJ/bit. Après avoir subi un processus de formation pour créer des filaments conducteurs, les mêmes dispositifs intégrés dans le BEOL de matrices RRAM peuvent être utilisés comme des dispositifs de mémoire resistive à plusieurs niveaux avec une endurance plus faible, environ 100,000 cycles. Ces résultats soulignent le potentiel de cette approche hybride pour exploiter les avantages des deux types de mémoire pour les charges de travail en IA.

Un circuit intégré exploitant la technologie de mémoire hybride développée a été validé expérimentalement. Ce circuit comprend à la fois des condensateurs ferroélectriques (FeCAPs) et des memristors dans le même BEOL de la fonderie CMOS de 130nm, afin de créer une matrice hybride de FeCAPs et de memristors interconnectés. L'unité fondamentale de la matrice hybride est un circuit synaptique hybride, constitué d'une collection de cellules une-transistor-un-FeCAP, où la bit-line de chaque cellule est directement connectée à la grille des transistors de sélection dans une cellule une-transistor-un-memristor. Ce sous-circuit permet le transfert direct de données numérique-à-analogique depuis plusieurs cellules FeCAP vers un seul dispositif mémristif, sans nécessiter de circuits intermédiaires. Cette technologie est compatible avec l'entraînement sur puce des ANN. Les dispositifs FeCAP stockent des poids cachés de haute précision qui subissent de nombreuses opérations de programmation pendant l'entraînement, tandis que les dispositifs memristor stockent des poids analogiques lus à la fois pendant l'inférence et l'entraînement.
Deux implémentations système ont été proposées. La première utilise des FeCAPs de même taille pour stocker les poids de haute précision et un seul memristor pour stocker le poids analogique. Cette approche limite la précision des poids cachés à n+1 états discrets, où n est le nombre de FeCAPs dans le circuit synaptique. Un réseau de neurones binarisé a été entraîné sur la tâche de détection d'arythmie ECG. En tenant compte des contraintes du circuit synaptique, le réseau atteint une précision de 88 %, avec huit FeCAPs stockant uniquement chaque poids de haute précision et un memristor utilisé pour stocker le poids binaire.
La deuxième implémentation utilise des condensateurs ferroélectriques de différentes surfaces pour encoder les poids de haute précision entiers sur 10 bits dans un format signe-et-amplitude et deux memristors pour encoder les poids analogiques positifs et négatifs dans la conductance différentielle des memristors. Des ANN fully-connected ont été entraînés à l'aide d'un algorithme de descente de gradient stochastique. En particulier, pour chaque échantillon d'entraînement, les activations des neurones sont calculées par une multiplication matrice-vecteur en mode feed-forward entre les poids analogiques (memristors) et les activations de la couche précédente. Les erreurs à la couche de sortie sont rétropropagées pour évaluer les gradients et mettre à jour les poids cachés. Aucun momentum n'est utilisé pour minimiser les coûts matériels. Les poids cachés (entiers de 10 bits dans les FeCAPs) sont mis à jour pour chaque échantillon, tandis que les poids analogiques (memristors) sont mis à jour tous les k entrées via la procédure de transfert développée. Sur le jeu de données MNIST, avec k=100, la méthode atteint une précision de 96.7 % avec une consommation totale d'énergie de programmation d'environ 38 nJ par poids, représentant une réduction de 38 fois sans perte de précision par rapport à k=1. Le nombre d'opérations de programmation reste 17 fois en dessous de la limite d'endurance des memristors et 75 fois en dessous de la limite des FeCAPs. De plus, la robustesse aux erreurs de mémoire a également été évaluée à partir de mesures du transfert analogique des FeCAPs vers

les memristors. Les résultats obtenus sur une variété d'applications (détection binaire des arythmies ECG, classification d'images sur les jeux de données MNSIT et Fashion MNIST) sont compétitifs par rapport à ceux obtenus par des modèles logiciels en précision flottante. Enfin, l'approche d'entraînement proposée a été évaluée sur une tâche d'apprentissage par transfert. Pour créer un scénario d'apprentissage par transfert, un réseau neuronal adapté aux dispositifs embarqués, MobileNet-V2, a été pré-entraîné sur le jeu de données CIFAR-100. Les couches convolutionnelles ont été utilisées comme extracteur de caractéristiques fixe, et une couche entièrement connectée a été ajoutée, qui a été entraînée sur le jeu de données CIFAR-10 en utilisant la stratégie d'apprentissage adaptée aux contraintes du circuit mémoire hybride. L'apprentissage par transfert adaptée au circuit mémoire hybride ne réduit la précision que d'environ deux points de pourcentage, à 88 %, confirmant que cette approche fonctionne bien même avec des jeux de données sophistiqués.

Un deuxième circuit développé vise à étudier les avantages de la combinaison de dispositifs mémoire ferroélectriques (Fe) et résistifs (Re), intégrés dans le BEOL de la technologie FDSOI de fonderie de 22 nm, pour l'implémentation matérielle de l'inférence et de l'entraînement dans les réseaux de neurones binarisés (BNN), désigné sous le nom de FeЯNet.
FeЯNet utilise des matrices FeRAM pour stocker des poids cachés de haute précision soit au format entier, soit au format à virgule flottante, avec des largeurs de bits de 8 bits ou multiples. Les poids binarisés sont stockés dans une configuration différentielle sur deux dispositifs mémoire résistifs. Les circuits logiques de transfert permettent le transfert parallèle de jusqu'à seize bits de signe des FeRAMs vers les RRAMs, accélérant la communication entre les deux matrices.
Les matrices RRAM sont équipées de circuits de lecture qui effectuent des opérations XNOR entre les poids binarisés stockés dans les dispositifs RRAM et les entrées présentées aux circuits de lecture. Cette opération, combinée avec des circuits de comptage de population, permet l'évaluation de multiplications matrice-vecteur dans les réseaux de neurones binarisés, accélérant et améliorant l'efficacité énergétique pendant les passes avant et arrière de l'entraînement BNN. Les simulations électriques Monte Carlo du circuit de détection conçu suggèrent la possibilité d'atteindre une énergie de multiplication binaire d'environ 100 fJ, avec une réduction supplémentaire à quelques dizaines de fJ possible avec un design de ligne optimisé.
La capacité synaptique, ou le nombre de poids qui peuvent être stockés dans le circuit FeЯNet, dépend de la précision souhaitée des poids cachés. La capacité maximale est atteinte avec des poids cachés de 8 bits, ce qui donne 16,000 poids synaptiques ; à mesure que la largeur de bits des poids cachés double, le nombre de synapses est divisé par deux.
Des mesures préliminaires de la puce FeЯNet valident partiellement la fonctionnalité de la carte de test développé et du circuit FeЯNet. Des tests supplémentaires sont nécessaires pour valider pleinement la fonctionnalité des différents blocs. Plusieurs pistes algorithmiques sont envisagées pour une implémentation utilisant FeЯNet, ce qui permettrait le développement de systèmes embarqués capables d'apprentissage.

List of Abbreviations

AI	Artificial intelligence
AGI	Artificial general intelligence
ANN	Artificial neural network
CMOS	Complementary metal-oxide semiconductor
SRAM	Static random access memory
IMC	In-memory computing
NVM	Non-volatile memory
CEA	Commissariat à l'énergie atomique et aux énergies alternatives
CNN	Convolutional neural network
ReLU	Rectified linear unit
SGD	Stochastic gradient descent
MCU	Micro-controller unit
QNN	Quantized neural network
PTQ	Post-training quantization
QAT	Quantization aware training
STE	Straight-through estimator
BNN	Binarized neural network
QT	Quantized training
ASIC	Application-specific integrated circuit
NPU	Neural processing unit
FA	Feedback alignment
DFA	Direct feedback alignment
LL	Local learning
SG	Synthetic gradients
EP	Equilibrium propagation
STDP	Spike-timing-dependent plasticity
SNN	Spiking neural network
CPU	Central processing unit
GPU	Graphics processing unit
TPU	Tensor processing unit
MVM	Matrix-vector multiplication
NMC	Near-memory computing
ADC	Analog-to-digital converter
DAC	Digital-to-analog converter
BEOL	Back-end of line
DRAM	Dynamic random access memory
PCM	Phase change memory
MRAM	Magnetic random access memory

MTJ	Magnetic tunnel junction
STT	Spin transfer torque
SOT	Spin orbit torque
FeRAM	Ferroelectric random access memory
PZT	Lead zirconate titanate
FeFET	Ferroelectric field effect transistor
MFMFET	Metal-ferroelectric-metal field effect transistor
FTJ	Ferroelectric tunnel junction
RRAM	Resisitive random access memory
LCS	Low conductance state
HCS	High conductance state
MOSFET	Metal-oxide semiconductor field-effect transistor
FeCAP	Ferroelectric capacitor
PUND	Positive up negative down
PVD	Physical vapour deposition
ALD	Atomic layer deposition
WL	Word line
SL	Source line
BL	Bit line
BER	Bit-error rate
MLC	Multi-level cell
TL	Transfer line
ECG	Electrocardiogram
MSB	Most significant bit
LSB	Least significant bit
HM	Hybrid memory
FP	Floating point
FDSOI	Fully-depleted silicon on insulator
PCB	Printed circuit board
GO2	Thick gate oxide
GO1	Thin gate oxide
LDMOS	Laterally-diffused metal-oxide semiconductor
DSA	Differential sense amplifier
FPGA	Field-programmable gate array
I/O	Input/Output

Contents

List of Figures

List of Tables

Preface

Intelligence is defined as: "The faculty of understanding; intellect. Also as a count noun: a mental manifestation of this faculty, a capacity to understand" [1]. Traditionally, intelligence was regarded as the exclusive domain of living beings, particularly humans. Machines were purely mechanical instruments, entirely lacking awareness or autonomous reasoning capacity. In recent times, the distinction between human cognition and machine intelligence is becoming increasingly blurred, creating widespread debate across philosophy, theology, law, economics, sociology, and virtually every facet of contemporary life.
Alan Turing, in 1950, was the first to suggest the possibility that *machines* could *think*. In the opening line of his article, *Computing Machinery and Intelligence*, he asks: "I propose to consider the question, «Can machines think?»" To answer this, he introduced the renowned *Imitation Game* (Figure 1) [1]. His pioneering research in the field of computation and machine intelligence laid the groundwork for other scientists to carry on. The term *artificial intelligence* (AI) was later coined by John McCarthy in 1956, in a proposal that he wrote for the Dartmouth conference, together with Minsky, Rochester and Shannon, scientists which are regarded among the founding fathers of AI.
The field of AI is vast and ever-changing. The techniques enabling machine intelligence have significantly changed since the early developments of the field. In general, the main goals of AI research include – but are not limited to – reasoning, knowledge representation, planning, learning, natural language processing, perception, and support for robotics. Ultimately, the objective is to develop a system able to reach artificial general intelligence (AGI), i.e. the ability to complete any task performable by a human on an at least equal level, as dictated for example by the Turing test [2].
Although still far from reaching AGI, it is safe to say that AI is driving the fourth industrial revolution by transforming industries through automation, data analysis, and advanced decision-making [3]. Moreover, technological advancements have made AI tools and devices ubiquitous in our daily lives, raising relevant concerns about data collection and the potential misuse of personal information. This rapid technological progress required, for the first time in history, regulation on the development of AI systems. The Council of Europe led the drafting of the Framework Convention on Artificial Intelligence and Human Rights, Democracy, and the Rule of Law, a treaty defining a risk-based approach to regulate AI and defining a set of general principles and obligations related to activities within the entire lifecycle of AI systems. Its general principles include, among others, respect for human dignity, transparency and oversight, accountability and responsibility, non-discrimination, and privacy and personal data protection.

[1] Oxford English Dictionary, s.v. "intelligence (n.), sense 1," June 2024

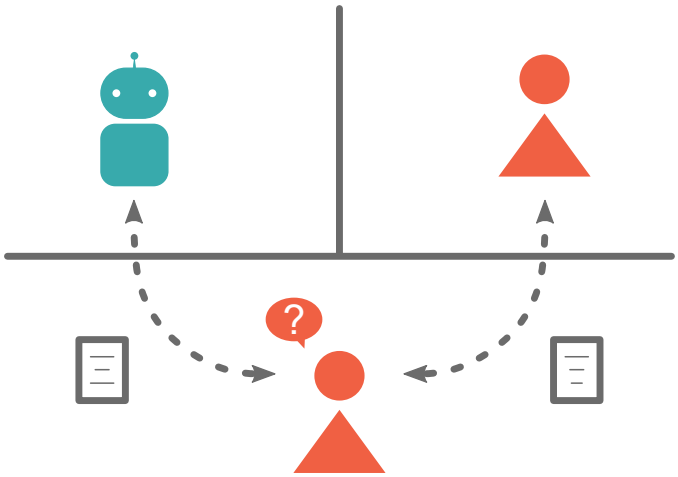

Figure 1: **How do we define artificial intelligence?** The imitation game proposed by Alan Turing in 1950 consists in a human judge interacting with both a human and a machine through written communication, without knowing which is which. The judge's task is to identify which participant is the machine. If the judge frequently mistakes the machine for the human, the machine is considered to have passed the test, demonstrating artificial intelligence.

Data on the cloud, data on the edge. Data is the fuel of AI, particularly in the development and training of artificial neural networks (ANNs), systems designed to mimic the biological brain's learning processes. These networks rely heavily on large, diverse datasets to learn patterns, make predictions, and improve over time. The more comprehensive the data, the more accurate and powerful these AI models can become. However, as AI and neural networks' reliance on data grows, so do concerns about privacy and security. Indeed, processing this vast amount of data is a power-hungry task that typically requires data-center hardware solutions, thus necessitating to shuttle data from the user to the Cloud. On the other hand, the processing capabilities of devices on the edge keep increasing, suggesting the possibility to incorporate more and more AI features on edge devices. Edge AI has the potential to offer several advantages compare to cloud-based solutions: faster response times, reduced latency, improved energy-efficiency, security and privacy. Currently, edge AI systems focus on inference and lack on-chip training capabilities. However, for personalized applications in user-specific environments, such as autonomous driving, smart manufacturing, medical equipment, and home automation, adaptive on-chip local training and fine-tuning of neural network parameters are essential. It is commonly agreed that current embedded hardware solutions cannot support the power-hungry learning process of a neural network, thus necessitating fundamentally new solutions to improve the energy-efficiency of these systems. With a power consumption of approximately only 20 W, the human brain is capable of accomplishing remarkable things, pointing the way of not only algorithmic solutions but also hardware innovations [4].

Closing the gap between memory and computing. There's probably never been a more exciting time to explore the world of microelectronics. With the slowing of Moore's law and, more importantly, the already broken down Dennard scaling trend [5], it seems obvious that the little gain obtained at each newly introduced complementary metal-

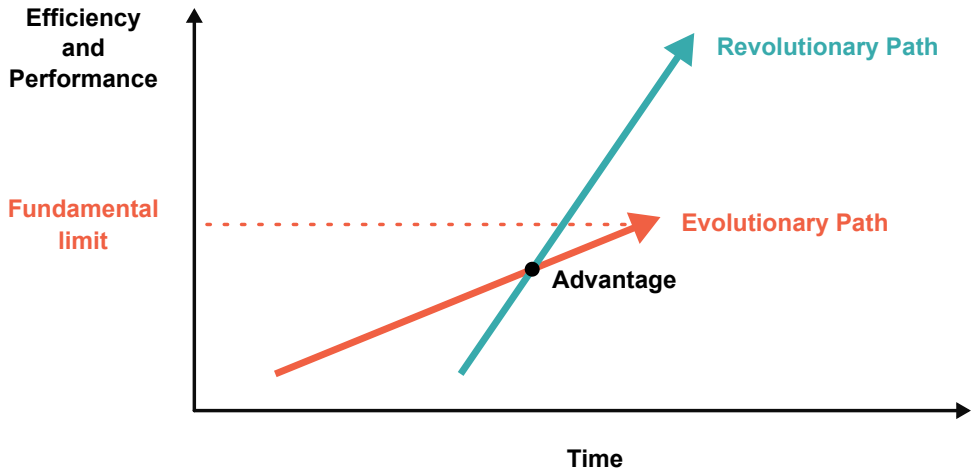

Figure 2: **Evolutionary and revolutionary paths in microelectronics.** CMOS scaling will eventually reach fundamental limits. The exploration of new devices, microarchitectures, and system paradigms aims to extend efficiency and performance, beyond the ones achievable by following the current evolutionary path.

oxide semiconductor (CMOS) technology node cannot keep the pace with the increasing demand of computational power required by today's workloads in classical computing architectures.

The "lucky" coincidence of AI booming and slowing down of improvements in computing performance calls for out of the box solutions to get back on track. Indeed, the *evolutionary path* offered by CMOS process scaling will eventually reach a fundamental physical limit. *Revolutionary* solutions, beyond CMOS, could ultimately provide an advantage in terms of efficiency and performance (Figure 2). Nevertheless, this requires novel materials, devices and processes, and system architectures.

AI data processing in von Neumann computing architectures requires significant information shuttling between memory and processing unit. In a commercial 45 nm processor supplied at 0.9 V, the energy consumption to perform an 8 bits integer multiplication is 0.2 pJ. In comparison, reading 64 bits from an 8 kilo-byte static random access memory (SRAM) cache requires 10 pJ, whereas retrieving them from the main memory dissipates 2 nJ, respectively resulting in a factor 50x and 10,000x compared to the cheap multiplication operation [6]. Equivalent considerations can be extrapolated in terms of time, pointing out the fact that communication dominates modern computation [7]. The difference in throughput, i.e. the amount of data transferred or processed per unit of time, between memory accesses and computation has become known as von Neumann bottleneck. This issue in becoming more and more relevant with advancing CMOS technology nodes.

Taking inspiration from the co-location of computing and storage in the biological brain, in-memory computing (IMC) architectures have emerged as an energy-efficient solution for processing ANN models. These architectures leverage a high-level of parallelism in computation to decrease time-complexity of specific operations and data-movement. Here, computation is performed within the memory arrays, with memory devices behaving as artificial synapses. When combined with non-volatile memory (NVM) devices, IMC solutions can skyrocket energy-efficiency of AI systems, providing opportunities particularly in edge settings.

Among NVM solutions, memristors[2] are particularly promising to enable next-generation IMC hardware, because they potentially offer improved energy efficiency, faster access times and larger density compared to traditional storage devices. Moreover, many memristive technologies naturally include non-volatility. Nevertheless they suffer from several device non-idealities, limiting their overall reliability at present. Although state of the art memristive technologies have proven near software-equivalent inference accuracy in many AI domains, their exploitation in learning-capable systems remains challenging. Indeed, memory requirements for training are more demanding, as learning processes in ANNs requires accurate iterative refinement of the synaptic strengths in the network. A memory technology simultaneously satisfying the demands for inference and learning does not yet exist [4]. This thesis seeks to address this challenge, by proposing a novel memory technology and co-designing hardware and algorithmic solutions for training on-chip.

AI research at CEA LETI and CEA LIST. This thesis was carried out at Commissariat à l'Énergie Atomique et aux Énergies Alternatives (CEA) in Grenoble, France. Artificial intelligence plays a major role in the research axis at CEA. The CEA LETI Institute focuses on the development the technological bricks to enable the development of future embedded AI systems, such as advanced transistor technologies, three-dimensional interconnections and emerging memory devices. The CEA LIST Institute creates innovative architectural solutions for frugal and trusted AI, while also offering expertise in algorithm development and embedded systems design.
This thesis, supported by CEA LETI and CEA LIST, seeks to study the compatibility of various learning algorithms for ANNs with emerging memory technologies developed at CEA LETI. The objective is the development of a demonstration integrated circuit on silicon that combines both emerging memories and conventional electronics, taking into account hardware constraints in on-board electronics.

Thesis contributions. The main contributions of this thesis are:

- The manufacturing and electrical characterization of a unified ferroelectric/resistive memory stack, which is based on a silicon-doped hafnia (HfO_2) layer sandwiched between two metal electrodes. The characterization of 16,384-device ferroelectric and resistive memory arrays evaluates the effectiveness and reliability of this memory technology for AI workloads.
- The design, manufacturing and test of an embedded circuit, interconnecting ferroelectric and resistive memory devices based on the unified memory stack, in the 130 nm CMOS technology node. System-level simulations of co-designed algorithmic solutions validate the potential of this technology for on-chip learning of ANNs.
- The design of an embedded circuit in the 22 nm CMOS technology node, combining ferroelectric and resistive memory arrays for near-memory inference and training in binarized neural networks. Preliminary manufacturing and characterization results highlight the potential of this technology for edge learning.

[2]The memristor is the fourth passive circuit element, introduced by Leon Chua in 1971 [8].

Thesis outline. This thesis is organized as follows:

- Chapter 1 is introductory, it has as objective introducing the vocabulary for both algorithmic and hardware solutions used throughout this thesis. Firstly, learning with back-propagation of errors are described, discussing the common practice solutions developed up until now to improve the overall training quality of an ANN and the challenges and opportunities of embedding these solutions in resource-constrained hardware. In the second part of this chapter, an overview of the optimal hardware solutions for ANNs are presented. In particular, in-memory and near-memory computing solutions based on emerging non-volatile memory devices are discussed. The chapter presents some of the most mature solutions in this field, for both inference and training, pointing out the need of an hybrid memory technology in the latter case.

- Chapter 2 presents the technological building block of this thesis, i.e. a hybrid memory technology based on ferroelectric and resistive devices. The key idea of the chapter is that the two kind of devices can be obtained with the same memory stack, a silicon-doped HfO_2 layer sandwiched between a bottom electrode and an active top electrode. Firstly, the chapter describes HfO_2-based ferroelectric and resistive devices in terms of operation, array implementation and process integration. Then, the developed unified memory stack is presented with extensive electrical characterization in both operating modes. Finally, insights for device optimization are discussed.

- Chapter 3 presents a proof-of-concept circuit utilizing the unified ferroelectric / resistive memory stack for implementing back-propagation-based learning algorithms. The design explores the potential of this memory architecture to enhance performance and energy efficiency for the training of binarized and quantized neural networks. Two distinct system implementations are presented and analyzed, highlighting their design trade-offs, operational benefits, and potential applications. The approach is validated on silicon and benchmarked via simulations on several AI tasks.

- Chapter 4 introduces a second circuit design that combines ferroelectric (Fe) and resistive (Re) memory devices for hardware implementation of inference and training in binarized neural networks, FeЯNet. The circuit design is presented alongside electrical simulations validating some of its components. The developed electrical characterization setup and some preliminary measurements are presented. Finally, potential applications are also suggested as future research directions.

- Chapter 5 concludes with a summary of the main findings of this thesis and discusses potential future directions.

List of publications

- **M. Martemucci**, F. Rummens, Y. Malot, T. Hirtzlin, O. Guille, S. Martin, C. Carabasse, A. F. Vincent, S. Saïghi, L. Grenouillet, D. Querlioz, E. Vianello, "Unified ferroelectric/memristive memory for neural network inference and training", 2024. (*Under Review*)
- **M. Martemucci**, F. Rummens, T. Hirtzlin, S. Martin, O. Guille, T. Januel, C. Carabasse, O. Billoint, J. Laguerre, J. Coignus, A. F. Vincent, D. Querlioz, L. Grenouillet, S. Saïghi, E. Vianello, "Hybrid FeRAM / RRAM synaptic circuit enabling on-chip inference and learning at the edge", in *2023 International Electron Devices Meeting (IEDM)*, pp. 1-4, 2023, doi: 10.1109/IEDM45741.2023.10413857.

List of Patents

- J. Borrel, **M. Martemucci**, Y. Beillard, L. Grenouillet, T. Hirtzlin, F. Rummens, E. Vianello, "Procédé de fabrication d'une matrice comportant au moins deux dispositifs de stockage et matrice associée". (*Pending*)
- **M. Martemucci**, F. Rummens, E. Vianello and T. Hirtzlin, "Hybrid FeRAM / OxRAM data storage circuit ", Patent US 2024/0135979 A1, 2024.

List of talks and posters

- The hybrid synapse for on-chip learning (Talk),
  PEPR Électronique Scientific Days, Grenoble, France, 2024.
- Exploring learning techniques for edge AI taking advantage of NVMs (Poster),
  GDR BioComp, Banyuls-sur-Mer, France, 2023.
- Learning at the edge taking advantage of NVMs (Poster),
  MEMRYSIS, Turin, Italy, 2023.

Chapter 1

Algorithmic and hardware solutions in artificial neural networks

The AI boom, or AI spring, characterized by the rapid progress of AI methods in the last decade, has been fueled by the development of machine learning techniques oriented towards the exploitation of artificial neural network models. As the name suggests, an artificial neural network is a computational model loosely inspired by the structure and connectivity of neurons and synapses in a biological brain. Multiple layers of neurons are connected by artificial synapse in order to create a deep network of connections. This layered structure is exploited to implement deep-learning methods, which are based on multiple levels of representation, obtained by composing non-linear modules that each transform the representation at one level (starting with the raw input data) into a representation at a higher, slightly more abstract level [9].
The implementation of ANNs is divided into two phases: learning and inference. The learning phase involves modifying the values of a set of parameters of the network, the synaptic weights, according to a learning algorithm. The goal is to make these weights converge toward values that enable the network to accomplish the task for which it was trained, achieving a sufficiently high level of accuracy. The inference phase involves applying the previously learned model to new input data.
Different architectures of ANNs have been developed to solve a variety of problems: convolutional neural networks (CNNs) have been used for image recognition [10, 11, 12] (detection, classification, segmentation) tasks, whereas recurrent neural networks for audio/speech recognition, and text translation [13, 14]. More recently, transformer network architectures have revolutionized the field of deep learning, in particular for sequence modeling and generation [15]. Indeed, transformer architectures are the building block of large language models like Open AI's GPT series, Google's Gemini, Meta's LLaMA, and many others, which are by now ubiquitous. The trend for most ANN models is to dramatically scale up in size. GPT models grow in size at each new release, reaching 1.8 trillion parameters with GPT-4 (approximately 10x bigger than its predecessor GPT-3). Nevertheless, as artificial intelligence is now being embedded more and more into various connected objects, ranging from medical implants to autonomous cars, it is clear that the algorithmic and hardware solutions available in data centres will not be able to cover all the AI integration needs. Different solutions have to be designed to allow also resource-constrained devices to operate AI tasks, from inference to learning [16].

This chapter explores the development of intelligent resource-constrained devices. Initially, it discusses algorithmic solutions within the context of ANNs, outlining common practices for training neural network models. The challenges and opportunities associated with moving the training process to edge devices are also examined. The second part of the chapter focuses on hardware solutions aiming at enhancing the energy efficiency of AI systems, highlighting the direction of this thesis with respect to technology and system architectures.

1.1 Artificial neural network training and inference

Artificial neural networks are computational graphs, whose structure is roughly inspired by the organization of neuron and synapses in the biological brain. The nodes of such graph represent the neurons and the connections between these nodes are the synaptic weights, or simply weights.
Biological and artificial neurons share similarities beyond just their connectivity (Figure 1.1). At the center of a biological neuron lies the cell body, or soma. The dendrites, branch-like cellular extensions forming a complex structure often referred to as a dendritic tree, stem from the soma. This is the primary region where the neuron receives inputs coming from other neurons. The axon, a thinner, cable-like projection, can extend to lengths many times greater than the diameter of the soma. Its main function is to transmit signals away from the soma. Typically, a neuron has only one axon, which frequently branches extensively, allowing it to connect with multiple target cells. At the farthest end of the axon from the soma lies the axon terminal, where synapses are located. These synaptic boutons are specialized sites where neurotransmitters are released to facilitate communication with other neurons. The nature of communication between biological neurons is electro-chemical. Indeed, the propagation of neurotransmitter travelling across the neuron, via ion pumps, alters the membrane potential of the neuron, defined as the difference in electric potential between the interior and the exterior of the neuron. At some point in time, an action potential can occur, when the membrane potential of a specific axon location rapidly rises and falls: this depolarisation then causes adjacent locations to similarly depolarise. The action potential event occurs when the membrane potential overcomes a threshold voltage. Therefore, action potentials play a central role in intra-cellular and cell-to-cell communication by providing for the propagation of signals along the neuron's axon toward synaptic boutons situated at the ends of an axon; these signals can then connect with other neurons at synapses. Action potentials in neurons are also known as "spikes", and the temporal sequence of action potentials generated by a neuron is called "spike train". A neuron that emits an action potential is often said to "fire" [17]. It is important to point out that this description of the biological neuron is over-simplified and it does not cover all existing types of neuron structures and observed phenomena. Nonetheless, it provides a general idea of the working principle of a biological neuron, inspiring their artificial counterpart.
Neuron elements in artificial neural networks perform similar computation compared to biological neurons. Indeed, an artificial neuron implements a weighted sum of the inputs. Then, an activation function is applied to this weighted sum to evaluate the output. The simplest form of artificial neural network is the perceptron network, introduced by Rosenblatt in 1958 [18]. The perceptron network is a single-layer – inputs directly connected

to the output layer – or multi-layer – inputs are connected to one or more hidden layers, then connected to the output one – linear network. In the perceptron network, the activation function is a step Heaviside function, meaning that the output is set to either 0 or 1 according to a threshold value. The similarities between the artificial neuron and the biological one can be understood here:

- Biological neurons operate based on an all-or-nothing principle [19]. When the combined input signals (after summing and weighting) exceed a certain threshold, the neuron "fires", sending an electrical signal down the axon to other neurons. If the threshold is not reached, the neuron does not fire. The step function in a perceptron mimics this behaviour by producing a binary output: if the weighted sum of inputs reaches or exceeds a certain threshold, the perceptron outputs 1 (analogous to the neuron firing); otherwise, it outputs 0 (analogous to the neuron not firing).
- The concept of a threshold in the step function is directly inspired by the threshold potential in biological neurons. In a biological neuron, the threshold is the critical level that the membrane potential must reach for the neuron to activate. Similarly, in a perceptron, the step function checks whether the input signal exceeds a certain threshold to determine the output.

While the perceptron and its step function are inspired by biological neurons, they represent a highly abstract and simplified model of a biological neural network. The step function abstracts the complex, continuous processes of a biological neuron into a simple, discrete decision-making rule.

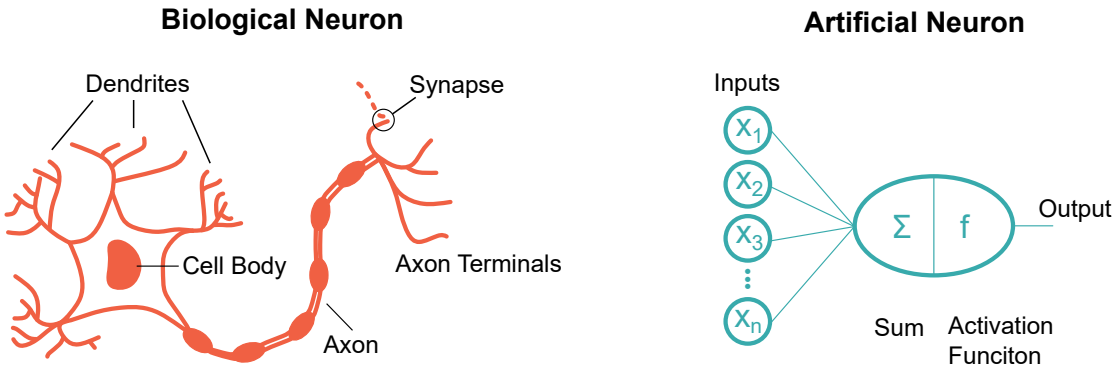

Figure 1.1: **Biological and artificial neurons.** The structure of a biological neuron (left) includes the cell body, the dendritic tree, the axon and the axon terminal where synapses are located. The inputs, integrated at the dendritic locations can propagate through the axon to reach the synapses. An artificial neuron (right) performs the multiplication and accumulation of weights and inputs. An activation function is applied to the weighted sum to evaluate the output.

As first attempt to emulate a biological neural network, the perceptron network had limited success, as it could only solve linearly separable problems. A significant milestone in deep-learning came in 1986 with the publication by Rumelhart, Hinton, and Williams, which introduced the renowned back-propagation algorithm as we know it today [20].

This paper marked the first introduction of a learning rule for actual multi-layered networks. Indeed, in perceptrons, the "feature analyzers" situated between the input and output layers are not genuine hidden units since their input connections are manually set, meaning their states are entirely determined by the input vector, and they do not learn representations. Learning becomes more complex but also more meaningful when hidden units are introduced, i.e. units whose states are neither provided nor required by the task. The back-propagation algorithm allows to determine when these hidden units should activate to achieve the desired input-output behaviour, essentially deciding what these units should represent.

1.1.1 The back-propagation algorithm

The back-propagation algorithm is a supervised learning method designed to automatically adjust the parameters, or weights, of an artificial neural network model. It works by minimizing an objective function, also called a cost function, which measures the error between the model's predicted outputs and the expected outputs. This cost function is computed as the average error across all training examples, and it is an N-dimensional function where N represents the number of parameters in the model. Thus, the cost function minimization is obtained by tweaking the model's parameters. This process is akin to solving a gradient descent optimization problem, a common approach in unconstrained mathematical optimization. The key idea is to take repeated steps in the opposite direction of the gradient of the cost function at the current point because this direction corresponds to the steepest descent. When the cost function is convex, all local minima are also global minima, meaning that gradient descent can potentially lead to the best possible solution.
The key finding of the back-propagation algorithm is a simple rule to evaluate this gradients in a multi-layer neural network architecture. Indeed, if the activation functions used at each layer of the network architecture are differentiable functions, the gradients of the cost function with respect to the weights, i.e. the weights updates, can be simply evaluated by exploiting the derivative chain rule. Figures 1.2a and 1.2b elucidate the key equations for the forward propagation of the inputs from the input to the output layer and the backward propagation of the errors from the output to the input layer, respectively.
During forward propagation, the activations at each layer are evaluated as the weighted sum of the previous layer's activations, passed through the non-linear activation function. The hidden layer activations can be various, such as the hyperbolic tangent function, sigmoid function, rectified linear unit (ReLU), leaky ReLU. The hidden layers transform the input through non-linear distortions, making the categories linearly separable in the final layer. Finally, the activation function at the output layer defines the output scores as well as how the errors should be evaluated, for the backward pass.
During the backward pass, for each hidden layer, the error derivative with respect to the output of each unit are evaluated, as a weighted sum of the error derivatives with respect to the total inputs to the units in the layer above. Then, the error derivative with respect to the output are converted into the error derivative with respect to the input by multiplying it by the gradient of the layer activation function. At the output layer, the error derivative with respect to the output of a unit is computed by evaluating the derivative of the cost function. For example, in binary detection problem a sigmoid function is used at the output layer, whereas for multi-class classification a softmax function is used to eval-

uate the output layer predictions. Thus, for a cross-entropy cost functions, which is the common-practice cost function for classification problems, the errors at the output layer can simply be calculated as the difference between the expected and predicted outputs. Once the error derivatives with respect to the input of each layer are known, the error derivative for the weight w_{ij} on the connection from unit i in the layer below is just $y_i \cdot \partial E/\partial z_j$. The weight update rule is therefore:

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} - \eta^{(t)} \left[y_i \frac{\partial E}{\partial z_j} \right]^{(t)} \tag{1.1}$$

where $\eta^{(t)}$ is the learning rate at iteration t. The learning rate is a scaling factor of the weight update defining the speed of the convergence. Tuning of the learning rate is essential in order to converge at the right rate, avoiding over- or under-shooting the optimal solution. Learning rate scheduling techniques exists in order to scale the learning rate as the training proceeds.
It is important to notice that convergence to poor local minima rarely poses a significant problem in large networks. Regardless of starting conditions, the system typically converges on solutions of similar quality. Theoretical and empirical findings suggest that local minima are not a major concern overall in deep neural networks [9].

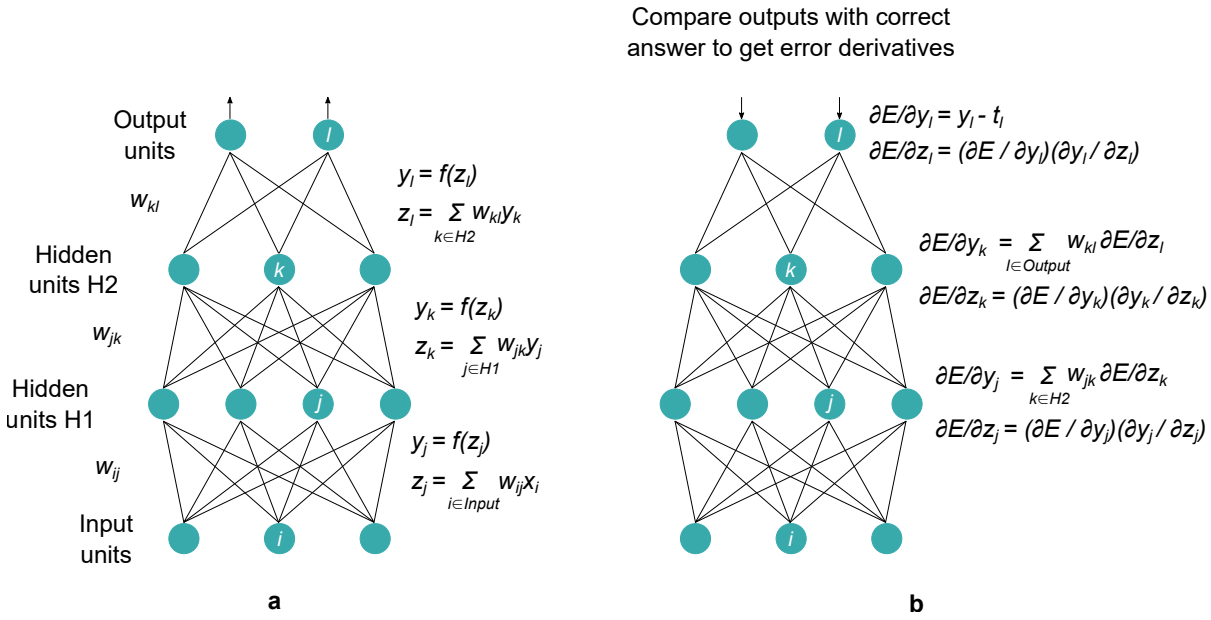

Figure 1.2: **Multi-layer neural networks and back-propagation. a** Equations used during the forward pass in a neural network with two hidden layers and one output layer. Bias terms are omitted for clarity. **b** Equations used for computing the backward pass for the same network. Reproduced from [9].

Setting up an optimization problem

By definition, supervised learning techniques like back-propagation require a ground truth to evaluate errors and adjust the parameters of a model to minimize the objective function. In the context of deep-learning, this ground truth is provided by datasets containing a

list of inputs and their corresponding labels. Datasets for deep learning applications are usually divided into three subsets:

- Training set: This dataset provides the inputs and labels that are used during training to evaluate the errors at the output layer and guide the updates to the model's parameters.
- Validation set: This subset is used during training to monitor the model's performance on unseen data and fine-tune hyperparameters, such as the learning rate and network size.
- Test set: The test set is used after the model has been fully trained to assess its generalization ability. It provides a final, unbiased estimate of the model's performance on new, unseen data.

Data from training, validation and test sets should belong to the same distribution for the training to be well defined. Sometimes, the test set is not used and the validation set acts as test set too. This practice, although not entirely rigorous, is quite common for smaller, simpler datasets. Indeed, the validation set indirectly participates in the training phase, by utilizing the performance information evaluated on this dataset to change the model hyperparameters accordingly. Common datasets, of increasing complexity, for image classification tasks are the MNIST [21], Fashion-MNIST [22], CIFAR-10 and CIFAR-100[23], and ImageNet [24] datasets. Some of these datasets are used later in this thesis, as they represent a standard benchmark for ANN systems for computer vision. Nevertheless, many other datasets exist for diverse tasks, which could in principle be more adapted in edge learning settings.
Assessing a neural network's accuracy on the training and validation sets helps identify issues related to the bias-variance trade off. Bias error arises from incorrect assumptions in the learning algorithm. High bias can lead to underfitting, where the algorithm fails to capture the underlying relationships between features and target outputs. Variance error, on the other hand, stems from the model's sensitivity to minor fluctuations in the training data. High variance often results in overfitting, where the algorithm captures random noise rather than the true data patterns. Typically, underfitting can be resolved by training the network longer, or utilizing bigger networks. Overfitting on the other hand might require more data to better generalize the training (e.g. techniques of data augmentation [25]), smaller networks, or other regularization techniques, such as weight decay [26] or Dropout [27].
After defining the network architecture, the model parameters should be properly initialized to start off training. An optimal initialization is essential to speed up training and avoid vanishing or exploding gradients. Indeed, the forward and backward propagation in deeper networks can cause the gradients to either have very small values, thus freezing the learning process, or large values, thus letting the learning process diverge, depending on the magnitude of the network weights.

Stochastic gradient descent and batch learning

There are several ways to implement the back-propagation algorithm to train a neural network for a given dataset. The standard gradient descent algorithm takes an optimization

step after all training examples are processed, by evaluating the average over the updates evaluated for each training sample. In the context of neural networks, this technique is called batch gradient descent. The new optimization step is taken by iterating again over the whole dataset, from the updated starting point.
The speed of convergence to the optimal solution can be adjusted, by splitting the training dataset in smaller subsets of data, called mini-batches, and taking one optimization step for each mini-batch. This technique is called mini-batch gradient descent. After all mini-batches are processed, the training dataset is shuffled before being split again in subsets for the next iterations. One entire iteration over the full dataset is called training epoch. If a dataset is composed of N samples and each mini-batch has size m, N/m optimization steps are taken for each training epoch.
If the size of the mini-batch is equal to one, an optimization step is taken for each training sample. This technique is called stochastic gradient descent (SGD). The name stochastic gradient descent is somehow misleading, as no stochastic processes occur during the training. Rather, the optimization procedure becomes more and more noisy as the size of the mini-batch decreases.
Figure 1.3 qualitatively illustrates the optimization process in the case of a supposed objective function of two parameters (w_1 and w_2) in the case of batch, mini-batch and stochastic gradient descent. The figure highlights the idea that the learning curve in batch gradient descent is the smoothest, whereas a noisier behaviour is observed for mini-batch and stochastic gradient descent. Moreover, batch gradient descent is typically the slowest form of learning. Indeed, even if less steps are required to get to the optimal solution, each optimization step requires processing the whole dataset each time. Conversely, in mini-batch gradient descent more optimization steps might be necessary, but less iterations over the whole dataset. The size of the mini-batches should be optimized according to the data distribution, the size of the dataset and the training settings. In edge contexts, SGD-like learning is typically preferred, as it enables real-time adaptation in changing conditions. Moreover, performing online updates consumes fewer resources compared to batch learning, which requires larger memory to store and process data in bulk. SGD-like algorithms are explored later in this thesis for the implementation of ANN learning in edge environments.

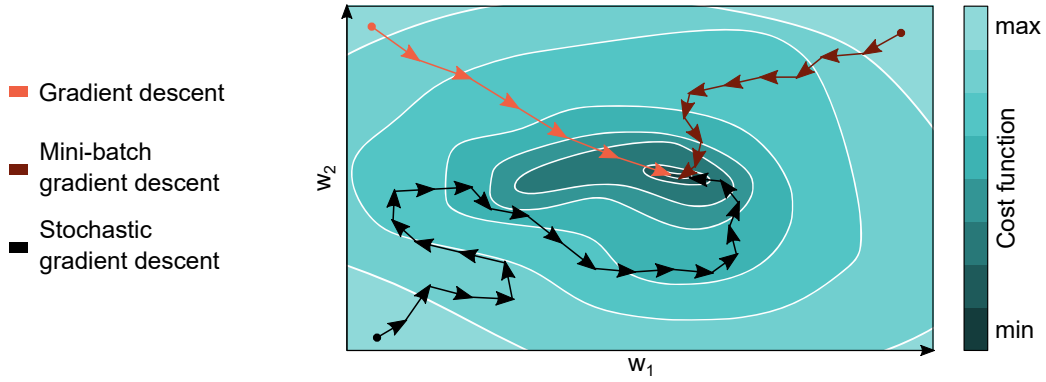

Figure 1.3: **Stochastic gradient descent and batch learning.** Optimization process in the case of a supposed objective function of two parameters (w_1 and w_2) in the case of batch, mini-batch and stochastic gradient descent.

Training deep neural networks is challenging because the input distribution to each layer can shift during training as the parameters of preceding layers are updated. This phe-

nomenon, known as internal covariate shift, complicates the training process by necessitating lower learning rates and meticulous parameter initialization. To tackle this issue, normalization techniques are typically used to keep the activation values at each layers with similar distributions. When working with batches, a common practice is to use batch normalization. This approach allows to use of higher learning rates and reduces the need for careful initialization. Additionally, it serves as a regularizer, sometimes removing the necessity for dropout [28]. It is important to notice that using batch normalization during training has a significant impact on the memory budget of the neural network as the activation for each neuron and each sample should be stored before an update could be evaluated. Moreover, batch normalization typically exploits trainable shift and scaling parameters that complexify the training procedure and require additional storage.
Other normalization techniques exist to deal with the problem of the internal covariate shift, such as layer [29], instance [30] or group normalization [31] techniques. Each technique presents some advantages over the other in different applications, training settings or network architectures.

Techniques to improve convergence

When training neural networks, optimization techniques play a crucial role in fine-tuning the learning process and ensuring that the model converges efficiently and accurately. Three widely used optimization methods are Momentum, RMSprop, and Adam. Each of these methods builds upon the standard gradient descent approach by incorporating additional features to enhance performance.
Momentum [20] is an optimization technique that helps accelerate the convergence of the gradient descent algorithm by smoothing out oscillations in the gradient updates. Instead of just moving directly along the negative gradient, Momentum introduces a velocity term that accumulates the gradients of previous steps. This accumulated velocity is then used to update the weights, allowing the model to maintain direction even when faced with small or noisy gradients. By doing this, Momentum helps the optimization process to navigate through the cost function's fluctuations more efficiently, reducing the risk of getting stuck in local minima or oscillating around the optimal solution.
RMSprop [32], or Root Mean Square Propagation, takes a different approach by adapting the learning rate for each parameter individually. It does this by maintaining a moving average of the squared gradients for each parameter, which is then used to scale the learning rate. The key idea behind RMSprop is to slow down the learning in dimensions where the gradients are large while speeding it up in dimensions where the gradients are small. This adaptive learning rate prevents the model from overshooting minima in steep directions while allowing it to move faster in flatter regions, ultimately leading to a more balanced and stable convergence.
Adam [33], short for Adaptive Moment Estimation, combines the benefits of both Momentum and RMSprop. It keeps track of an exponentially decaying average of past gradients (like Momentum) and an exponentially decaying average of past squared gradients (like RMSprop). These two components are then used to compute adaptive learning rates for each parameter. Additionally, Adam includes bias correction terms to counteract the initial conditions where the moving averages might be biased towards zero. This makes Adam highly effective in handling noisy gradients and sparse data, providing fast convergence and often performing well out of the box without the need for extensive hyperparameter

tuning. As a result, Adam is one of the most popular and widely used optimization algorithms in deep learning today.
These optimization strategies come at the expense of an increased memory cost. Indeed, Momentum and RMSprop optimizers require storing double the amount of parameters to keep tracks of both weights and moving averages on gradients and square gradients respectively. As both components are used for the Adam optimizer, the memory requirements are triplicated in this case. Although effective for off-chip training, these techniques might not be supported on systems with a limited amount of memory.

1.1.2 Embedding artificial neural networks on-chip

With the rapid expansion and widespread adoption of Internet of Things (IoT) devices, and the continuous rise in connected everyday gadgets, the volume of data being gathered from the world is skyrocketing. This situation raises several issues:

- Increasing privacy concerns and the security risks linked to access to such vast amounts of data.
- Costs or limitations associated with transmitting all this data for inference and/or training purposes.

A question arises: Given the immense data generated by these edge devices, is it really necessary for the data to leave the device? Why not allow the training process to occur directly on the edge device? This concept significantly expands on the common notion of edge AI. Edge AI has the potential to revolutionize healthcare and enhance the security of individuals, buildings, and industrial setups. This potential has driven considerable research into developing edge devices capable of performing AI tasks with minimal energy consumption. Currently, these devices primarily focus on inference and lack on-chip training capabilities. However, for personalized applications in user-specific environments, such as autonomous driving, smart manufacturing, medical equipment, and home automation, adaptive on-chip local training and fine-tuning of neural network parameters are essential. Embedding AI capabilities at the edge poses significant challenges in terms of power consumption and memory capacity. What is typically called edge refers to systems with peak power consumption below 10 W approximately [34]. For instance, a typical digital embedded platform is a micro-controller unit (MCU). MCUs are essential to the automation of various products and devices, including automobile engine control systems, implantable medical devices, household appliances, and other embedded systems. By integrating the microprocessor, memory, and input/output components into a single unit, micro-controllers significantly reduce the size and cost of designs, making digital control feasible for a wider range of devices and processes. Mixed-signal micro-controllers, which incorporate analog components, are especially common for managing non-digital electronic systems, such as sensors and actuators. In the realm of IoT, micro-controllers are a cost-effective and widely used solution for collecting data, sensing, and actuating in the physical world as edge devices. Some micro-controllers are designed with low power consumption in mind, operating on low-bit formats and frequencies in the order of kHz, which can result in power usage in the range of milliwatts or even microwatts.

As an example, AI face identification has been proven to run on a STM32H7 micro-controller, enabling security-grade applications without compromising performance or biometric data privacy [35]. The STM32H7 micro-controller series is based on the 32-bit Arm Cortex-M7 core and offers embedded flash memory ranging from 64 Kbytes to 2 Mbytes, manufactured using a 40 nm process. In the best case, an MCU with 2 Mbytes memory used exclusively to store a neural network model with 32-bits weights would allow to store a model with 500,000 parameters, or a million parameters if half-precision representation is used. A convolutional neural network architecture specifically conceived for embedded application for image classification purposes like MobileNet totals 4.2 million parameters [36], clearly pointing out the need for more compact solutions for the storage of weights.

Overcoming memory constraints: Quantized neural networks

Deploying a neural network model at the edge requires reducing the numerical precision of the parameters of the model to lower-bit formats. Quantized neural networks (QNNs) are a type of neural network where the precision of the network's parameters is reduced from the standard floating-point representation (typically 32-bit) to a lower-bit representation, such as 8-bit, 4-bit, or even 1-bit [37]. This reduction in precision is known as quantization. Of course, lower precision leads to more significant reductions in model size and computational load, but it may also affect the model's accuracy. There are several strategies to obtain a quantized neural network model to deploy, which can be classified in three categories:

- Post-training quantization (PTQ): This approach converts a pre-trained full-precision network to a fixed point one. It takes a trained network and quantizes it with little or no data, requiring minimal hyperparameter tuning and no end-to-end training. Several quantization pipelines and techniques have been engineered to reduce the quantization error, defined as the accuracy difference between the full-precision and quantized model. In most cases, PTQ is sufficient for achieving 8-bit quantization with close to floating-point accuracy. On the other hand, moving to lower bits resolution requires more advanced techniques [38].
- Quantization aware training (QAT): This technique requires fine-tuning and access to labeled training data but enables lower bit quantization with competitive results. QAT models the quantization noise source during training. This allows the model to find more optimal solutions than post-training quantization. However, the higher accuracy comes with the usual costs of neural network training, i.e., longer training times, need for labeled data and hyper-parameter search. In QAT, during the forward pass of back-propagation, the weights (and activations) are quantized to evaluate the next layer activations. During the backward-pass, the quantization blocks of the computational graphs are bypassed by using a technique called straight through estimator (STE), which approximates the gradient of the quantization operation to one. This procedure effectively allows to evaluate the weights gradients for the current fine-tuning iteration. The fine-tuning is stopped when the accuracy obtained with the quantized model is close enough to the one obtained with the floating-point model [38].

A technique similar in principle to QAT consists in training a neural network from scratch, simulating quantized weights (and activations) during forward propagation and updating a full-precision copy of the weights. In this case, instead of starting with a pre-trained model, the weights are initialized and a complete training is performed. Binarized neural networks (BNNs) have emerged in this context as a technique to obtain an extremely quantized model where only weights [39] or both activations and weights can only take two values, i.e. ± 1 [40, 41]. For image classification applications, BNNs have shown nearly state of-the-art accuracy performance on the MNIST, CIFAR-10 and SVHN datasets [40], as well as on the more complex ImageNet dataset [41].

- Quantized training (QT): Quantized training (QT) involves updating the model exclusively within the quantized parameter space. The ultimate goal is to develop a framework where all elements involved in network training – such as weights, activations, gradients, and errors – are quantized. However, training deep neural networks with reduced precision poses significant challenges, particularly in preserving the accuracy of gradients during back-propagation.
  Various studies explored different levels of quantization. For instance, DoReFa-Net applies low-bitwidth floating-point quantization to gradients during the backward pass [42], while TernGrad reduces gradient updates to ternary values to minimize gradient synchronization overhead in distributed training [43]. Despite these approaches, DoReFa-Net and TernGrad still store and update weights using 32-bits floating point precision throughout training. Additionally, these methods do not address the quantization of batch normalization or its derivative. On the other hand, the WAGE framework facilitates a fully low-bitwidth integer dataflow in ANNs for all parameters during both training and inference. WAGE also eliminates batch normalization by implementing a novel initialization technique and a layer-specific constant scaling factor. WAGE achieves satisfying accuracy across multiple datasets with a 2-8-8-8 bit width configuration for weights, activations, gradients, and errors, respectively [44].

Figure 1.4 qualitatively summarizes the three quantization strategies in the case of a supposed objective function of two parameters (w_1 and w_2).
Although the result of all the aforementioned techniques is a quantized neural network model, PTQ or QAT require a copy of the full-precision neural network model to start with or during simulation. Therefore, these two methods are effective for training a model off-chip in order to obtain a portable model to be embedded on a resource-constrained edge device for inference-only applications. On the other hand, embedded training requires updating the quantized model without having any information of the full-precision one. Thus, a quantized training strategy should be chosen in this case.
Beside the memory required for weights, embedding training on chip requires additional memory to store intermediate quantities to evaluate gradients and errors. If batch normalization is used, the amount of supplementary memory required to evaluate the updates for one iteration is approximately proportional to the number of neurons in the network, the size of the mini-batch and the number of bits required to store the normalized activations. Moving to online training strategies, i.e. online stochastic gradient descent, could be beneficial for embedding training on chip. Moreover it can be argued that an online learning process is more adapted to an edge setting, where data arrives sequentially rather than in batches.

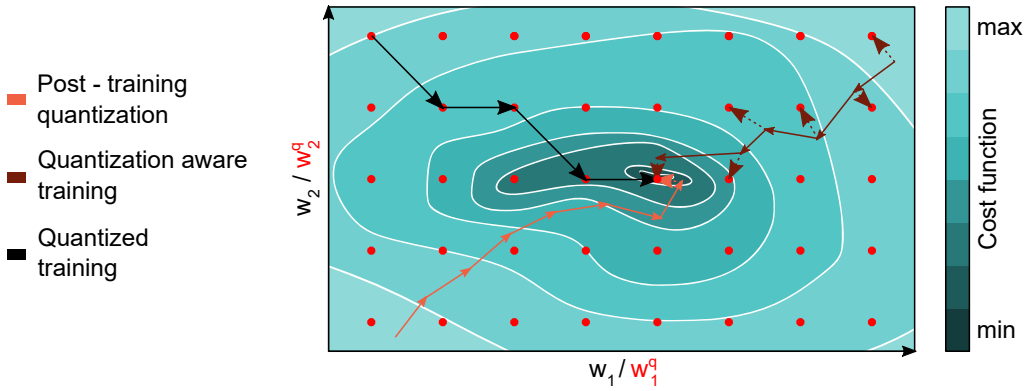

Figure 1.4: **Quantization techniques for artificial neural networks.** In the space of the model parameters (w_1 and w_2), a grid of quantized values is defined (red dots). In PQT, the training is performed in the real valued parameter space (solid orange lines). After the training procedure is terminated, a quantization step occurs (dotted orange line). In QAT, the quantized weights are simulated during the forward step (dotted brown lines), while a full-precision latent copy of the weights (solid brown lines) is optimized. In QT (solid black lines), the parameters optimization during training is constrained to quantized values only.

Overcoming energy constraints: Application-specific integrated circuits

Quantization of neural networks offers advantages not only from a memory perspective but also for reducing the energy consumption during inference and training.
Considering deployment on a conventional hardware, compared with 32-bits floating point representation, 8-bit integer multiplications and additions respectively require 18 and 30 times less energy (estimated on a 45 nm technology at 0.9 V) to be performed [44]. Moreover, having a quantized model reduces the memory accesses costs and memory size requirements during training, which can greatly benefit mobile devices with on-site learning capability.
Application-specific integrated circuits (ASICs) can be developed in order to implement neural networks operations as defined by a training algorithm in the most efficient way, thus reducing the energy consumption to the bare minimum. In the second part of this chapter, the focus of the discussion shifts to the hardware implementation of artificial neural networks, highlighting the advantages of specific hardware architectures for deep-learning applications, in terms of energy and area efficiency.

1.1.3 Challenges and perspectives on edge artificial intelligence

It is clear by know that the term edge AI, i.e. the possibility to embed artificial intelligence capabilities in low power systems, covers a broad range of applications and techniques. Although still requiring further optimization, edge inference is already present in some commercial products. As an example, the two best-selling smartphone companies nowadays, Apple and Samsung, already included neural processing units (NPUs) in their top-level products to locally process data for some AI tasks. The Apple A16 Bionic system on a chip – used in iPhone 14 Pro and 14 Pro Max, and 15 and 15 Plus – includes a 16 cores NPU for improved computational photography capabilities and for handling

screen-related features. Similarly, Samsung includes a NPU – Qualcomm Hexagon – for accelerating AI inference tasks in the Snapdragon 8 Gen 3 processors, used in the latest Galaxy S24 series. The greatest challenge now is finding a strategy to move training off the cloud as well, for improved data security and privacy, efficient processing and reduced latency. This is a challenge from both an algorithmic and hardware perspective, or more precisely for the hardware-software co-development.

The computational efficiency of edge systems is constantly increasing, feature that is driving the possibility to include more and more AI features in edge systems. However, a single edge device still has limited data collection and relatively modest computational power. In many scenarios, there are numerous devices performing similar training tasks. If these devices could collaborate and collectively train a shared model — one that benefits from the vast amounts of data they collectively gather – it would be a game changer. If this could be accomplished while keeping the data on the devices themselves, it would eliminate the need to store the data in a central data center. This approach, known as federated learning is an active research area to enable collaborative training at the edge [45]. Federated learning is not the only decentralized approach used to develop edge AI systems. Swarm learning is another technique, inspired by the collective behaviours observed in social organisms like birds and insects, to create decentralized and self-organizing AI systems. In swarm learning-based networks, edge devices collaborate to address complex problems and make decisions by sharing insights in a fully decentralized way. Unlike federated learning, where data is centralized, swarm learning allows edge devices to share information without relying on a central cloud. This enables edge AI systems to independently adapt, evolve, and enhance their performance and efficiency in real time [46].

Another challenge in the context of edge learning is continual learning, also known as lifelong, sequential or incremental learning, with slightly different interpretations. Generally speaking, with continual learning, it is meant the ability of a system to continually learn new information without losing knowledge of previously learned data. This is achieved by dealing with the stability-plasticity dilemma, i.e. the ability of a system to be stable enough not to forget previous knowledge, still allowing sufficient plasticity to learn new tasks. If this is not achieved, a phenomenon known as catastrophic forgetting can occur, i.e. the fact that a neural network trained for a given task can lose accuracy on the former task while a new one is learned. The development of continual learning strategies would be beneficial for both cloud and edge learning, nevertheless it is at the edge that this would show the greatest advantage, where re-training of the whole network with new information is more problematic, as previous data is not accessible at all time and energy constraints pose significant challenges for a complete re-training. Nevertheless, finding lightweight strategies for continually learning at the edge can be challenging. Several methods have been proposed for implementing continual learning. These can be grouped into three categories, i.e. regularization methods, parameter isolation techniques or replay-based approaches [47]. Regularization methods draw inspiration from synaptic consolidation in the brain, where frequent activation of certain neural pathways strengthens the connections between those neurons [48]. From an algorithm perspective, this approach involves adding a regularization term to the classification loss function to retain previous knowledge, encouraging the mapping function for new tasks to remain close to that of prior tasks [49, 50, 51]. Parameter isolation methods, analogous to neurogenesis in the brain, involve the development of new neurons to assimilate and solidify new information. In these methods, specific parameters that correspond to previously learned tasks are preserved and frozen while a new set of parameters is adapted to handle the new task [52]. Finally,

replay-based methods encourage the model to retain previous knowledge by revisiting earlier examples or their approximations. This approach is inspired by the complementary learning systems theory, which explains memory consolidation in the brain through the interaction between the hippocampus and the neocortex. From an algorithm perspective, replaying old examples during the learning of new tasks helps integrate new knowledge with existing knowledge, similar to the regularization in other methods, preventing the model from deviating significantly from its correct behaviour. Replay methods utilize the inherent plasticity of artificial neural networks by reinforcing old knowledge during the learning of new tasks, rather than suppressing this ability [47, 53].

A third challenge for learning-capable edge systems is data management, in particular working with unstructured and/or unlabeled data. Standard techniques for supervised training of neural networks deal with highly structured data with pre-defined labels. Moreover, training is typically performed by re-iterating several times on a whole dataset to converge to an optimal or near-optimal solution. Nevertheless, edge AI systems face storage constraints that prevent them from fully leveraging large numbers of data points. Datasets might not be entirely stored in an edge system to perform the iterative training process. Generally, multiple epochs can help the model generalize better by allowing it to fine-tune its parameters. If the data points are independently and identically distributed (iid), each data point is drawn from the same probability distribution and is independent of others. This means that any given subset of the data should represent the overall distribution well. It could be argued that, a large enough dataset with independently and identically distributed data would ensure that the model is exposed to a sufficient amount of diverse examples covering the entire distribution during just one pass. In this case, the model could, in theory, learn the underlying patterns in the data from this single pass, i.e. without the need to store the whole dataset. Moreover, data collected near the sensor could be highly noisy, unlabeled and sometimes corrupted. In this scenario, data selection and labeling could be provided by either humans or automated tools. For example, learning algorithms could actively query the user/teacher for labels. This type of iterative supervised learning is called active learning [54]. Finally strategies like self-supervised learning, or a mixture of supervised and unsupervised learning could ultimately provide a viable solution to effectively learn in typical edge settings. Indeed, according to Bengio, Hinton and LeCun, unsupervised learning will gain significant importance in the future, since this method reflects how humans and animals learn, primarily through observation rather than being explicitly being taught the labels of objects [9].

Indeed, different learning strategies might be more adapted to different settings. The constraints imposed by embedded electronics might limit the training capabilities of a system. Training a whole neural network from scratch can be energy expensive, thus unsuited for edge applications. In this context, techniques like fine-tuning or transfer learning can be used to exploit larger pre-trained networks to adapt to a specific task, which can be trained locally with new input data [55].

In other settings, in which a dynamic environments interact with an intelligent agent, strategies like reinforcement learning can be particularly powerful. This could be often the case in edge environments, such as an industrial robot learning how to optimize a specific process, a system able to develop personalized treatment for a patient, or a delivery drone adjusting its flight trajectory.

1.1.4 Alternative learning strategies

Error back-propagation is one – yet the most successful – of the many learning strategies proposed in the context of artificial neural networks. Many training algorithms, supervised or unsupervised, have been proposed as possible substitutes to back-propagation.
Two main criticisms usually arise when comparing back-propagation with respect to other algorithms. First of all, back-propagation is well-suited for digital hardware like graphic processing units, due to their parallel processing and precision. Co-designing algorithms with hardware could reduce these the energy cost required to train larger models. Indeed, alternatives like in-memory computing with emerging memory devices offer energy efficiency and parallelism but face challenges with device non-idealities that make the implementation of back-propagation difficult [56]. On a more fundamental, algorithmic perspective, back-propagation is not believed to be a bio-inspired solution for training, as no evidence of an error signal back-propagating in the brain to adjust the synaptic connections has been conclusively observed. A key factor contributing to the biological implausibility of back-propagation is the weight transport problem. Back-propagation relies on identical forward and feedback paths with matching synaptic weights for effective training. While biological neural networks might also feature separate forward and feedback pathways, transferring weights between these paths is not feasible because it would require extremely rapid information transmission along the axon from each synapse output [57]. Taking inspiration from the brain has always been a traction force in the field of deep learning, as the brain effectively processes data with high energy frugality. It is believed that synaptic adjustments in the brain occur at a more local level, depending on the neighbouring neuron activities [58]. For this reason more bio-inspired solution are under investigation for enhanced energy efficiency. In the following, some of the proposed candidates to replace standard back-propagation are discussed, highlighting the key ideas behind the various approaches.
Feedback alignment (FA) is similar in principle to back-propagation. It leverages fixed random weight matrices to transmit error gradient information back to hidden layers, bypassing the need for symmetric weights [59]. It has been demonstrated that aligning the sign between forward and feedback weights is sufficient for transmitting effective error signals. While FA addresses the issue of weight transport, it still requires that the forward and backward passes to be completed sequentially before parameter updates can occur, i.e. it is forward pass and backward pass locked.
Direct feedback alignment (DFA) improves upon FA by directly propagating errors from the output to each hidden layer via random matrices, eliminating the dependency on the sequential backward pass [60]. DFA has shown comparable performance to back-propagation on CIFAR-10 for small fully-connected networks with dropout, but it underperforms on convolutional neural networks. Nevertheless, algorithms based on FA also depend on systematic feedback connections to layers and neurons, but such extensive reciprocal connectivity has not been observed in the brain.
Local learning (LL) is an alternative that reduces feedback connectivity. These kind of algorithms train parts of the network – typically layers – independently, calculating a separate loss for each layer using an auxiliary classifier [61]. LL methods have achieved results close to back-propagation on CIFAR-10 and are advancing on more complex tasks. These algorithms still employ back-propagation for training individual layers and classifiers and face the weight transport issue, as they remain dependent on both forward and backward passes at the layer level. Combining FA with LL can potentially unlock layers

from the backward pass, meaning that weight updates can be implemented as soon as the forward pass is completed, and enable greedy learning without relying on a global learning signal.
Another method, called synthetic gradients (SG), allows independent layer training by using auxiliary networks to predict the gradient of the backward pass, known as synthetic gradients [62]. Similar to LL, SG methods use back-propagation to train the auxiliary networks. Until these networks are sufficiently trained, SG faces the weight transport problem and is locked in both forward and backward passes, i.e. both forward and backward passes need to be completed before performing weight update.
Signal propagation (Sigprop) is a new approach that is forward unlocked, meaning that updates can be evaluated as soon as the forward pass is completed for each layer. Sigprop generates targets from learning signals and reuses the forward path to propagate these targets to hidden layers for parameter updates. Sigprop has several advantages: it utilizes the same forward path for inputs and learning signals, eliminating the need for additional feedback connectivity, weight transport, or a backward pass. Parameters are updated as soon as the forward pass with the learning signal reaches them, without blocking subsequent inputs or storing activations, making Sigprop ideal for parallel training of layers or modules. Since it relies on a single type of computation, Sigprop effectively addresses all the constraints posed by other approaches using a global learning signal [57].
Equilibrium Propagation (EP) offers another alternative. It is an energy-based model employing local contrastive hebbian learning, utilizing the same computational processes during inference and learning phases. EP operates through a continuous recurrent neural network that minimizes the difference between two fixed points: one where only the input is received and one where the target is used for error correction. It has been shown that the signal propagated during the second phase corresponds to the error derivatives and encodes the gradient of the objective function, when the weight updates align with a standard form of spike-timing-dependent plasticity (STDP). This method makes it more plausible that the brain might implement a mechanism similar to back-propagation [63].
Communication in the brain occurs via spiking signals traveling through neurons, with STDP believed to play a crucial role in synaptic changes. STDP links the difference in time between post-synaptic and pre-synaptic spikes to the changes in synaptic weights, as observed in biological neurons. However, its role as part of a learning algorithm requires further exploration. For instance, while STDP has been demonstrated to approximate back-propagation update rule [64], by itself it is not the most suitable method for training deep networks to achieve high accuracy in deep learning applications. Nevertheless, Spiking neural networks (SNNs) represent a step closer to mimicking the brain's dynamics, by distributing information over time through binary spikes. Although SNNs have shown promise in energy efficiency compared to traditional ANNs, they are computationally demanding to train due to added time complexity and the non-differentiable nature of spiking signals. The most common methods to train deep SNNs directly use back-propagation or indirectly convert a pre-trained ANN into an SNN format. Conversion methods struggle with high latency, while direct training increases offline training costs and reduces accuracy [65].
All of the aforementioned approaches are under active investigation to find a more bio-inspired, hardware-friendly, and energy-efficient learning strategy capable of providing performance equivalent to back-propagation in terms of accuracy and the ability to solve increasingly complex tasks across various contexts. Additionally, it should be noted that different solutions may be more suitable for different types of hardware, making the "lot-

tery ticket" in terms of optimal training strategy elusive. The memory technology and circuit implementations developed in the remainder of this thesis appear better suited for back-propagation-like learning. For these reasons, the focus of this thesis remains on error back-propagation-like algorithms. Nevertheless, some of the aforementioned learning strategies could be adapted to the proposed hardware architectures for further exploration.

1.2 Deep learning hardware accelerators and non-volatile memory technologies

In the rapidly evolving landscape of artificial intelligence and deep learning, the successful implementation of neural networks depends not only on the efficacy and performance of the algorithms but also on the underlying hardware support.
Although many of the key algorithmic elements essential for deep neural networks were established in the 80s, e.g. the back-propagation of errors coupled with ANNs in 1986, it was not until thirty years later that these networks gained widespread recognition as a viable research avenue. The delay between these theoretical breakthroughs and their practical success was largely due to hardware limitations at the time [66].

1.2.1 A quest for parallelization

Undeniably, the success of AI in recent years is due to the utilization of graphics processing units (GPUs) for both training and inference, rather than the more classical central processing units (CPUs). Architecturally, when we look at the CPU, we find it typically comprises a limited number of cores alongside a sufficiently large cache memory. Cache memory improves locality of data but has limited capacity (Figure 1.5). This design allows the CPU to efficiently handle a modest number of software threads simultaneously. In sharp contrast, GPU stands out with its multitude of cores, typically hundreds, enabling it to manage thousands of threads concurrently. Modern deep learning neural networks training present a formidable challenge, as it involve adjusting parameters ranging from millions to well over one billion. Additionally, these networks demand substantial volumes of training data to achieve the desired high accuracy levels. This implies that hundreds of thousands to even millions of input samples must traverse both forward and backward passes during training. Neural networks are inherently parallel in structure, as they consist of numerous identical neurons, and this inherent parallelism aligns perfectly with the architecture of GPUs. Thus, GPUs offer a remarkable acceleration in computational speed compared to CPU-only training, making them the preferred choice for training neural networks. Also, the parallel nature of inference operations also aligns seamlessly with GPU execution.
Nevertheless, GPUs are still processors that must support a vast number of applications and software. Thus, in a GPU each calculation necessitates access to registers or shared memory for reading and storing intermediate results, which brings us back to the enduring challenge known as the von Neumann bottleneck. For this reason, Google introduced the tensor processing unit (TPU), a purpose-built matrix processor tailored for the unique demands of neural network workloads [67]. Indeed, TPUs excel at handling the extensive multiplications and additions inherent in neural networks. The pivotal breakthrough here

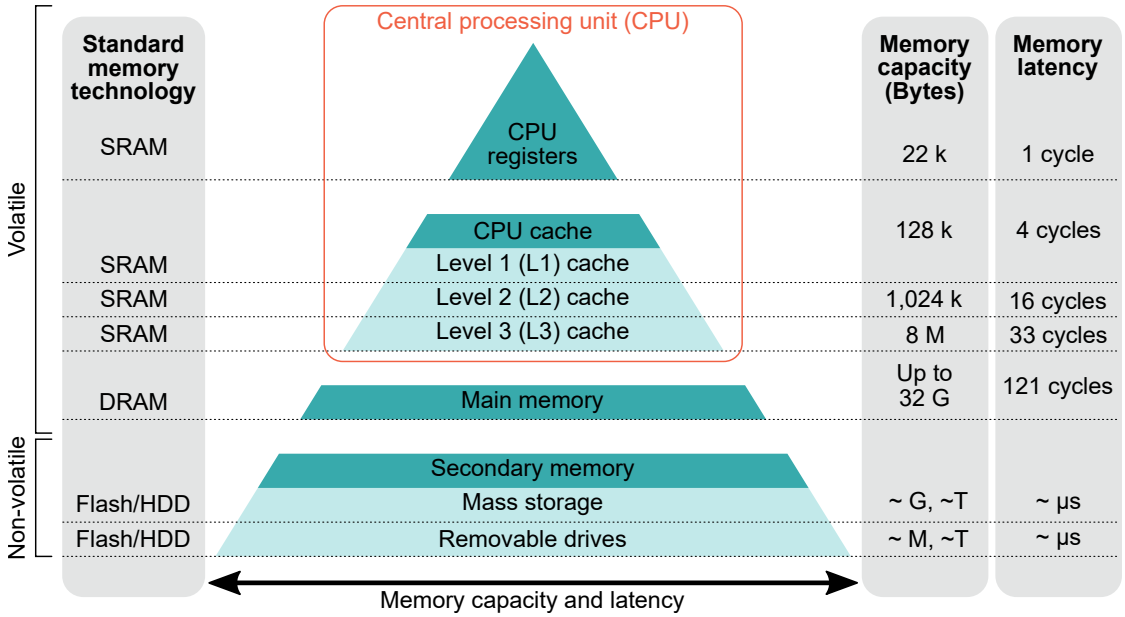

Figure 1.5: **Memory hierarchy in a modern computer micro-architecture.** Memory instances in a modern computer are organized in a register, cache, main memory configuration to optimize data-handling. Memory capacity and latency data are evaluated for a quad-core desktop CPU at 3 GHz, data retrieved from [68].

is the substantial reduction of the von Neumann bottleneck. In a TPU, thousands of multipliers and adders are directly interconnected to create a physical matrix of operators. This architectural approach is known as a "systolic array". To elucidate the operation of a systolic array in executing neural network computations, the TPU initiates by loading the neural network parameters from memory into the matrix comprising these multipliers and adders. Subsequently, input data is fetched from memory. Each multiplication operation is executed sequentially, with the results propagated to the next set of multipliers while concurrently undergoing summation. As a result, the output represents the cumulative sum of all multiplication outcomes between the data and parameters. Remarkably, throughout this extensive computational process and data propagation, there is no need for memory access whatsoever. This distinctive approach is the very reason why TPUs can achieve exceptional computational throughput in neural network calculations while consuming significantly less power and occupying a smaller physical footprint. Although TPUs significantly reduce the von Neumann bottleneck, a clear separation between memory and processing elements still stands.

1.2.2 A computational memory as solution to the von Neumann bottleneck

In the pursuit of energy efficiency for the hardware implementation of deep neural networks, in-memory computing has emerged in the last decade as an alternative approach to the compute–storage paradigm of the von Neumann architecture. In-memory computing is a quite general term, including a variety of strategies. In general, the main difference between a von Neumann architecture and an in-memory computing approach is that, in the latter case, data stored in a single device is never requested to perform computation outside of the memory, rather the result of a computation is transmitted between the

processor and the computational memory.
The typical computation performed by IMC cores is the matrix-vector multiplication (MVM), ubiquitous in ANNs implementation. On the other hand, the scenario in which the computation is performed in close proximity of the memory, but not within the boundaries of the computational memory, is typically referred to as near-memory computing (NMC). Digital and analog in-memory computing strategies exist, depending on whether the multiplication between weights and activations and accumulations are performed in the digital domain, exploiting digital multiplier and adder circuits, or in the analog domain, if multiplication and accumulation are performed by exploiting Ohm's and Kirchhoff's laws respectively. These approaches not only reduces latency and energy consumption associated with data transfer but also has the potential to dramatically improve the time complexity of specific computational tasks. The main advantage comes from the high level of parallelism provided by arrays of memory devices simultaneously performing computation. Therefore, by integrating processing capabilities directly with memory units, in-memory computing significantly boosts computational efficiency. Nevertheless, this efficiency gain comes with the trade-off of sacrificing the flexibility found in traditional computing architectures.
Non-volatile memory has emerged as a crucial component in this regard, providing storage solutions that keep data even when power is removed and also in-memory computing capability. Moreover, NVM technologies offer unique advantages over traditional volatile memory, such as faster access times, lower power consumption, and enhanced data retention. In particular, emerging NVM technologies revolutionized the field since 2008, when HP Labs first linked the resistive switching behaviour of a solid-state device to the conceptual memristor [69]. Memristors were first introduced by Chua in 1971, as the fourth circuit element, linking the electric charge and the magnetic flux with a nonlinear relationship [8]. Memristors are characterized by a pinched hysteresis loop in the current-voltage plot. The change in the slope of the pinched hysteresis curves indicates a transition between distinct resistance states, a key phenomenon in resistive memory devices. In 2010, a memristive analog device was used to demonstrate synaptic functions, paving the way for the investigation of neuro-inspired computing based on memristive devices [70]. Since then, increasing research efforts have been devoted to the devices, architectures, chips and algorithms for NVM based neuro-inspired computing.
Figure 1.6 offers a schematic representation of the distinction between a von Neumann architecture and the in-memory computing paradigm. The difference between a standard CPU architecture, a GPU and a systolic array of processing elements, building block of a TPU, is shown. These solutions are compared to the implementation of an in-memory matrix vector multiplication on a crossbar array of memristive devices. Here, the memristors act as artificial synaptic elements, mapping in their conductance the value of a synaptic weight. The cross-bar array implementation maps the all-to-all connectivity found in the fully-connected layers of an artificial neural networks. The vector-matrix multiplication, basic operation to evaluate the activations at successive layers of a neural network, can be implemented by using the Ohm's law and the Kirchhoff's current summation law. Indeed, if the inputs are mapped as voltage pulses applied to the row lines, the current collected at each column line corresponds the sum of the products between the applied inputs and the synaptic weights. It is important to notice that the applied voltage pulses must not perturb the physical state of the memristor, i.e. it should not change its conductance value when it is applied. If the voltage pulses are applied to the column lines and the currents are collected to the row lines, the multiplication between an input vector and

the transposed of the weight matrix is performed. This operation is performed during the backward pass of the back-propagation algorithm, which can be implemented by means of the same in-memory computing core.

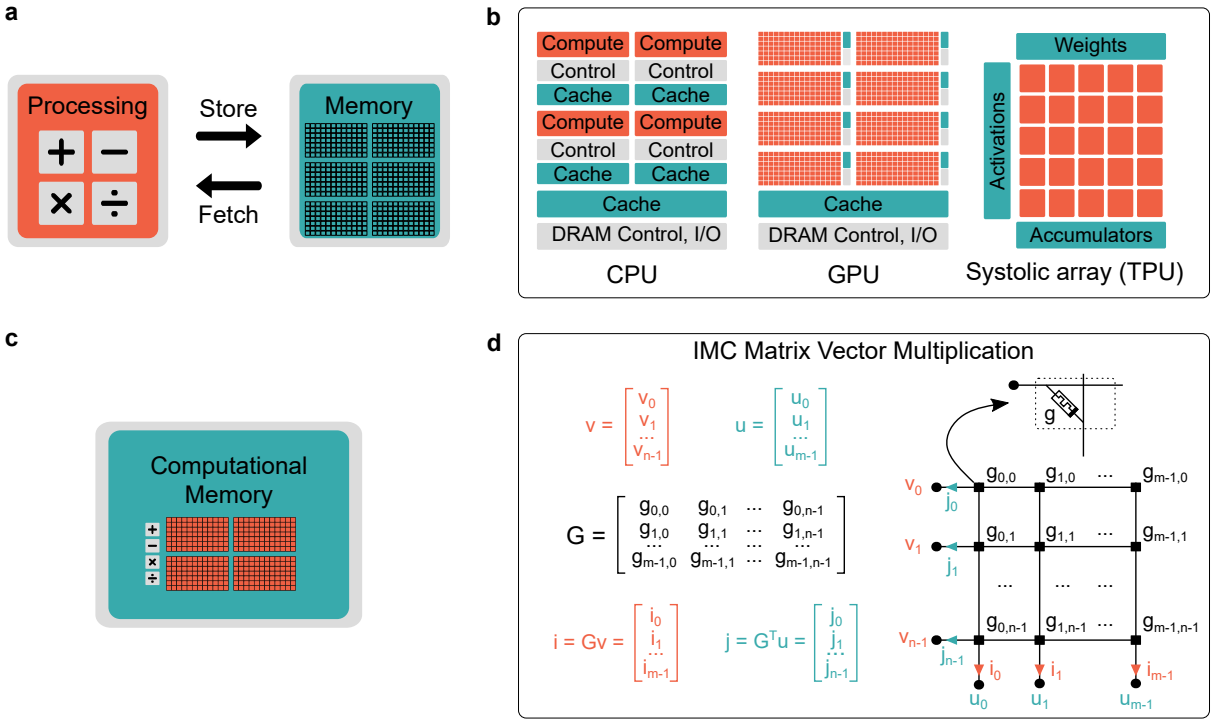

Figure 1.6: **Von Neumann and in-memory computing architectures.a** In a von Neumann computer architecture memory and processing units are physically separated. **b** Central, graphic and tensor processing units are example of von Neumann architectures with increasing degree of parallelism in computation. **c** An in-memory computing architecture performs the computation within the boundaries of the computational memory. **d** A crossbar array of memristive devices can perform the matrix vector multiplication within the memory array. The vector elements are mapped to applied read voltages and the matrix element are mapped to the memristors conductance values.

It is important to notice that an analog in-memory computing core might include many other blocks, apart from the array of memristors itself. First of all, the selector transistors connected in series with each memristors are not shown in Figure 1.6 for clarity, but they should be included to avoid sneak path currents and add more flexibility. Each IMC core should also include analog-to-digital converters (ADCs), to provide a digital results corresponding to the analog summation, on both vertical and horizontal lines if both forward and backward passes should be performed in-place. Also digital-to-analog converters (DACs) are necessary to apply the inputs to the array if these are stored as digital values [71]. The IMC core should also include the circuitry to program the synaptic weights, which can be quite complex in case of multi-level storage, because of the need of program-verify schemes [72, 73, 74, 75]. Designing a complete IMC core is therefore a real challenge, that requires accurate circuit optimization, dependent on the specific device technology [76].

1.2.3 Neuromorphic hardware

In- or near-memory computing cores are often classified under the broader category of hardware accelerators known as neuromorphic processors. The term neuromorphic processor generally refers to hardware accelerators that emulate, to varying degrees, processes observed in the biological brain. Indeed, the possibility to perform the MVM operation within the memory boundaries is one key similarity, among others, between biological and neuromorphic systems.
However, neuromorphic systems can emulate biological processes of the nervous system even more closely. A significant area of research within neuromorphic computing focuses on the hardware implementation of SNNs. SNNs naturally incorporate the temporal dynamics observed in the biological brain and enable the development of biologically plausible learning rules. Several noteworthy research chips demonstrating advancements in this area have been introduced in recent years, including:

- Loihi chip: Fabricated using Intel's 14nm process, Loihi comprises 128 neuromorphic cores, each capable of simulating up to 1,024 neurons. It features a programmable microcode learning engine that supports STDP-based on-chip learning rules, enabling real-time adaptation and self-modification without external supervision. The chip utilizes an asynchronous network-on-chip (NoC) for efficient spike-based communication [77].

- ODIN: It is a digital spiking neuromorphic processor designed for online learning, prototyped in a 28nm CMOS process. The chip comprises 256 neurons and 64K synapses, utilizing stochastic spike-driven synaptic plasticity (SDSP) learning rules to facilitate real-time adaptation [78].

- ReckOn: It is a spiking recurrent neural network processor developed to enable on-chip learning and particularly suited for tasks requiring temporal processing such as navigation and gesture recognition. Implemented in a 28nm CMOS process, ReckOn occupies a compact area of $0.45\,\mathrm{mm}^2$, making it suitable for embedded applications where space and power efficiency are critical. The processor employs a modified E-prop algorithm for online learning, allowing it to adapt to new information in real-time without the need for external computation resources [79].

- ROLLS processor: It is a mixed-signal neuromorphic chip, fabricated using a 180 nm CMOS process and designed to emulate various neural systems through its reconfigurable architecture. It contains 256 silicon neurons and 128K plastic synapses, with an additional 256K programmable synapses, allowing for complex neural network configurations. The chip implements STDP learning rules [80].

Unlike the aforementioned approaches, the developments presented in this thesis exhibit a limited degree of emulation of biologically plausible behaviors due to specific algorithmic choices. Specifically, the implementation of back-propagation-like algorithms in classical ANNs introduces a key divergence from biologically plausible processes. While back-propagation has proven effective for training ANNs, it relies on mechanisms – such as the global error calculation and weight updates based on gradients – that do not naturally align with the distributed, asynchronous, and energy-efficient dynamics observed in

biological neural systems. This limitation highlights a fundamental trade-off in current neuromorphic research: achieving high computational performance and training efficiency often comes at the cost of strict biological realism. Future efforts may focus on developing alternative learning paradigms that more closely reflect the brain's adaptive processes. Integrating such bio-inspired algorithms into neuromorphic hardware could bridge the gap between performance and biological fidelity, advancing the field toward systems capable of emulating the complexity of biological neural networks while remaining effective across a wide range of applications.

1.2.4 Overview of commercial AI accelerators and future perspectives

Figure 1.7 presents the announced peak performance, measured as number of operations per second, of commercially available AI accelerators against the peak dissipated power. The exhaustive list of accelerators can be found in [34]. The different accelerators are categorized as chips, cards or systems according to the form factor and they are split different groups:

- Very Low Power for speech processing, very small sensors, etc.;
- Embedded for cameras, small drones and robots, etc.;
- Autonomous for driver assist services, autonomous driving, and autonomous robots;
- Data Center Chips and Cards;
- Data Center Systems.

In the figure, filled markers define the performance of accelerators with training capabilities, whereas empty markers denote inference-only accelerators. Different markers are utilized to group accelerators with different computation precision. We can extrapolate several pieces of information of the general trends for AI accelerators from this plot. In particular, it can be observed that almost all accelerators with training capabilities require larger power consumption and they belong therefore to the data center category. Currently, there is no commercial product at the edge (Very low power and embedded accelerators) able to train a neural network system. The inference-only accelerators at the edge typically exploit low-bits integer computing precision (from one to eight bits) or analog computation, as in the case of the Mythic76 and Mythic108 chips. On the other hand training requires larger precision representation, such as 16 bits or 32 bits floating point representations. We can also observe that most of accelerators lie below the threshold of 100 fJ/Operation, i.e. an energy efficiency of 10 TOps/W. Going beyond the threshold of 10 TOps/W is proving to be challenging, even for in-memory computing approaches, such as the chips by Axelera and Mythic.

In this graph, this thesis is positioned in the leftmost part of the plot. No commercial product exists for comparison to the approaches that are proposed in the rest of the thesis, not because of the absence of a market for such systems, but rather for the early stage of its technology readiness level (TRL). Indeed, even though not yet commercialized NVM-based accelerator hold promise to dramatically increase the energy efficiency of such AI

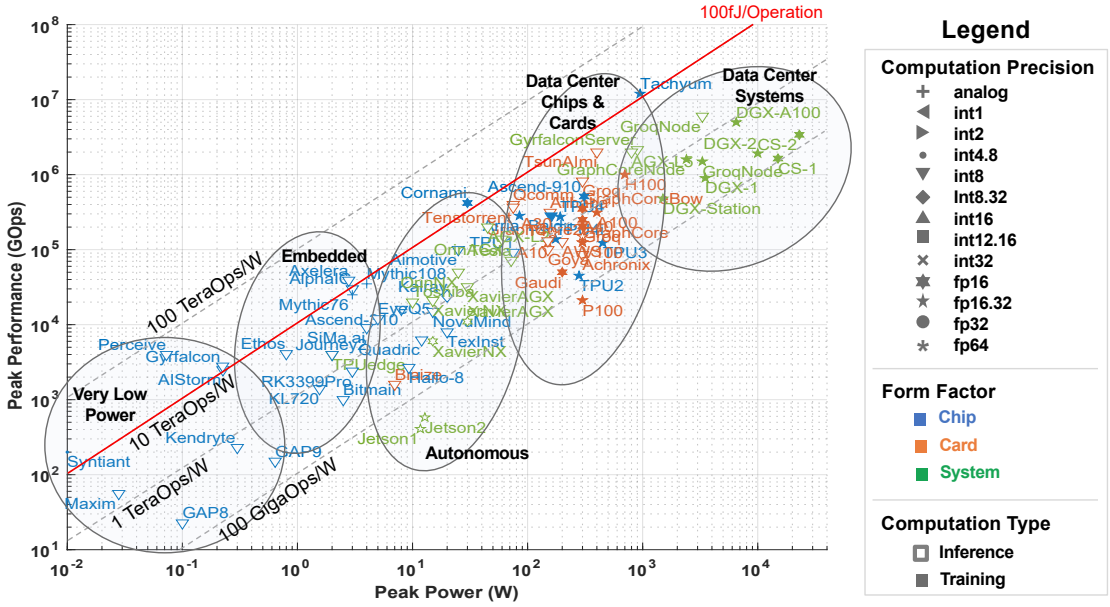

Figure 1.7: **Trends of commercially available AI hardware accelerators.** Peak power vs peak performance scatter plot of AI accelerators available on the market. Reproduced from [34].

accelerators, taking advantage of increasing memory density, non-volatility and in-memory computing capabilities.

1.2.5 Comparison of various NVM solutions

Although not yet commercialized, memristor-based accelerators are highly researched for a number of reasons. First of all, the intrinsic non-volatility of these devices offers a clear advantage compared to standard CMOS solutions, which either require larger power consumption to keep the system in the on state or reloading of the neural network parameters each time the system is shut-off, which results in larger latency. Moreover, conversely to SRAM, memristors have the potential to be scaled down in size below 2 nm [81] and to be integrated into high-density three-dimensional arrays [82], thus potentially increasing the amount of on-chip memory at equivalent surface. Moreover, because of their non-linear I-V characteristic, memristors offer several other advantages, such as the possibility to be used for multi-bit storage or to emulate some bio-plausible behaviours. Many flavours of memristive technologies exist, depending on the physical properties exploited for the resistive switching. Memristive technologies are anyway a subset of a broader class of NVM technologies. Sometimes the two terms are used interchangeably, even though they should not be confused: the first being defined by its pinched-hysteresis current-voltage loop, whereas the second includes any kind of memory technology with the capability to retain information when power is shut off. Furthermore, many NVM devices offer manufacturing processes compatible with the CMOS back-end of line (BEOL), thus providing the possibility to offering a more globally packed system solution.
In Figure 1.8, the latest taxonomy update of memory technologies from the International Roadmap for Devices and Systems is reported [83]. Volatile memories include the classical SRAM and dynamic random access memory (DRAM), whereas the field of non-volatile

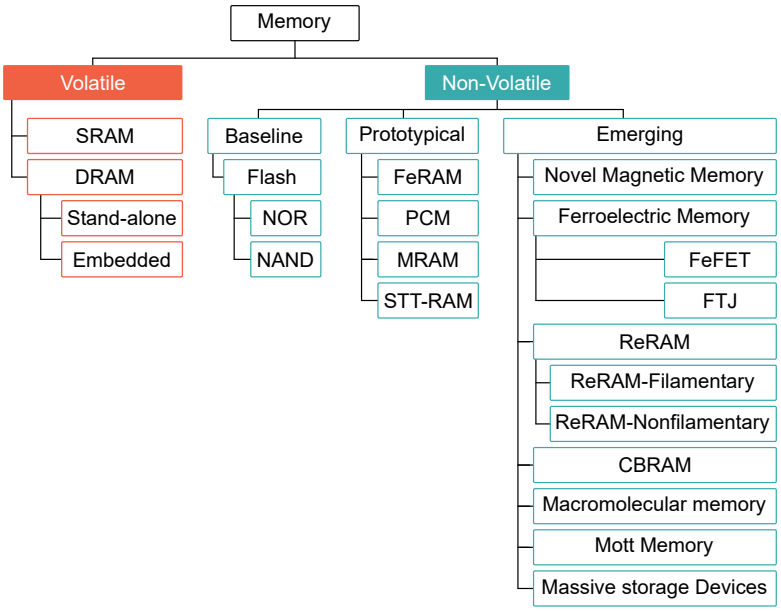

Figure 1.8: **Taxonomy of memory technologies.** Classification of conventional volatile and non-volatile memories and emerging memory technologies.

memories has seen a great expansion in recent years. The baseline flash memories, in their NAND and NOR configurations, are now accompanied by emerging memory technologies, some of which are already commercialized for storage applications or in the prototypical stage. In the following, a brief description of the most promising emerging NVM technologies is presented, highlighting the strengths and weaknesses of each one of them.

Phase-change memories. Phase change memories (PCMs) are among the most mature resistive memory technologies. They are based on chalcogenide glasses, typically compounds of Ge, Sb and Te (GST), which have the characteristic of changing their crystalline phase thanks to heat produced during the application of electrical pulses. The typical structure of a PCM is the so-called mushroom cell, which includes a layer of GST sandwiched between a top metal electrode and a bottom electrode containing a metallic heater element. When a large amplitude voltage pulse is applied to the top electrode, the current flowing in the crystalline GST heats up the material by Joule effect, creating an amorphous dome on the heater via a melt-quenching process. This results in a larger resistance of the memory device, which is effectively reset. On the other hand, lower amplitude pulses applied to the top electrode can crystallize the amorphous GST, thus increasing the conductance of the memory element, bringing it back to a set state. The crystallization process is typically a cumulative process. Indeed, the application of several set pulses crystallizes larger portions of the amorphous dome. Therefore intermediate conductance levels can be created, for multi bit-storage. On the other hand, the reading operation of a PCM device simply consists in evaluating the cell resistance via small-amplitude voltage pulses. As all others emerging NVMs, PCMs suffer from several device non idealities, first of all conductance drift. With conductance drift it is meant the temporal evolution of the programmed conductance, arising from a spontaneous structural relaxation of the amorphous phase. Depending on the application, this phenomenon might be more or less problematic [84].

Because the operating principle of PCMs requires significant atomic rearrangement in the active volume of these devices, these are characterized by a limited write endurance (in the order of 1-10 million cycles) and large write energy consumption (approximately 10 pJ per bit). On the other hand, read endurance is potentially unlimited and the read energy remains low [85, 86].
Phase-change memory is among the few emerging NVM technologies to have been commercialized to date. Indeed, the 3D XPoint technology, developed by Intel and Micron Technology, was based on multiple stacked layers of one selector-one resistor devices, where each resistor was constituted by a phase-change device. Intel Optane, the brand name of 3D Xpoint memory chips, was announced in July 2015 and was available on the open market from April 2017 to July 2022. Intel Optane offered itself as one of the first solutions for the so called storage class memory, a new class of memory aiming at bridging the performance gap between DRAM and storage solutions. Despite the excellent performance, it is expected that thermal disturbance and smaller write program margin will hinder the scalability of PCMs. Therefore chalcogenide-based selector-only memories are investigated for next generation storage class memory [87]. PCMs are also explored as an embedded memory solution for automotive micro-controllers applications by STMicroelectronics [88] or for processing in memory solutions by IBM and others [89, 90].

Magnetic memories. Magnetic random access memory (MRAM) exploits the tunnel magnetoresistance effect occurring in a magnetic tunnel junction (MTJ), which is a component consisting of two ferromagnets (one with pinned magnetic moment and the other with a free one) separated by a thin insulator. If the insulator layer is thin enough, electrons can tunnel from one side to the other of the tunneling barrier, producing a current. The resistance of the MTJ device is low when the magnetic moment of the free layer is parallel to the fixed layer, whereas it becomes large when the free layer moment is oriented anti-parallel to the fixed layer one. Therefore two resistance states can occur depending on the relative orientation of the magnetic moments of the two ferromagnets. The initial MRAM devices utilized toggle memory switching, where a magnetic field altered the electron spin. The second generation of MRAM, known as STT-MRAM (Spin-Transfer Torque MRAM). In STT-MRAM devices, the spin of electrons is flipped using a spin-polarized current in a MTJ, rather then a magnetic field. The standard STT-MRAM structure uses an in-plane MTJ (iMTJ), whereas a more optimized configuration exploits a perpendicular MTJ (pMTJ) structure, where the magnetic moments are perpendicular to the silicon substrate. Perpendicular STT-MRAM is more scalable and cost-effective compared to iMTJ STT-MRAM, making it a promising technology for replacing DRAM and other memory types. Finally, SOT-MRAM (Spin-Orbit Torque MRAM) has the potential to surpass STT-MRAM in speed, density, and efficiency. SOT-MRAM switches the free magnetic layer by injecting an in-plane current into an adjacent SOT layer, unlike STT-MRAM where the current is injected perpendicularly into the MTJ, and both read and write operations occur through the same path [91].
For comparison to other NVM technologies, STT-MRAM is considered, as this is the most investigated solution at present. For write operations, STT-MRAM offers high endurance (from 10^{11} to more than 10^{14} cycles) and large energy consumption (in the order of the pJ per bit) [85, 86]. The large write energy has the potential to scale down due to the unique current-induced switching mechanism of MRAM [92], nevertheless this still

remains challenging to realize at chip level. Read endurance is virtually unlimited, but the small high to low resistance ratio complicates the design of the sensing circuitry. MRAM technology is currently commercialized by many companies, both for embedded and stand-alone memory applications [92, 93].

Ferroelectric memories. Ferroelectric RAM (FeRAM) is a memory technology similar in principle to DRAM, as it uses a Metal-Insulator-Metal capacitor to store data in combination with a selector transistor. However, unlike DRAM, FeRAM incorporates a ferroelectric material instead of a dielectric layer as the insulator, which gives the device non-volatility by encoding a binary data through the alignment of ferroelectric dipoles in two opposite directions. Ferroelectric materials are characterized by a polarization-voltage hysteresis loop, meaning that a residual charge can be measured across the ferroelectric layer when the power is removed. This residual polarization is due to the creation of semi-permanent electrical dipoles inside the material, aligning to the direction of an external electric field. The creation of such dipoles comes from the crystalline structure of the material, typically a non-centrosymmetrical one. Therefore, the write operation involves applying a voltage pulse across the device to switch the polarization of the ferroelectric domains according to the voltage polarity of the pulse. On the other hand, reading is performed by applying a write pulse to one of the two electrodes and sensing the emitted charge on the other. The amount of sensed charge indicates whether a polarization switch has occurred, effectively allowing the information stored in the device to be read. This makes the reading process data-destructive, requiring a write-back operation each time information is retrieved from the device.
Historically, the ferroelectric material used for FeRAMs was lead zirconate titanate (PZT) [94]. Indeed, PZT-based FeRAM was used as early as mid-1990 by Samsung and Hyundai Electronics (now SKHynix) for the production of the first stand-alone ferroelectric memories. Still today, PZT-based are the most mature solution for stand-alone FeRAMs, manufactured by companies as Infineon, Texas Instruments and others. Nevertheless, scalability issues of PZT materials hampered the utilization of FeRAM for embedded applications at more advanced technology nodes and for high density memory applications [95]. On the other hand, in 2011, ferroelectricity was discovered in doped hafnium oxide thin films [96]. Since then, HfO_2 based FeRAM as been highly researched for embedded applications, because of its high speed, low power usage, and compatibility with CMOS technology. Different kinds of ferroelectric memories are now being explored, but the simple one-transistor-one-capacitor (1T-1C) ferroelectric random access memory stands out due to its moderate write voltage, long retention time, and high endurance (projected to up to 10^{15} cycles [97]). Moreover, the low current required to switch the ferroelectric capacitor allows to decrease the programming energy consumption even below 100 fJ per bit [98]. The same energy is required for reading operations. Moreover, because of the data-destructive read procedure, read endurance is limited and linked to the write one. In FeRAMs, the read signal weakens as the ferroelectric capacitor area decreases, limiting its cell scalability for planar capacitors. A solution to this problem sees the exploitation of cylindrical three-dimensional capacitors. Micron has recently disclosed major advancement in this direction, by presenting a dual layer 2 Gb memory macro of BEOL integrated three-dimensional ferroelectric capacitors. The performance of the presented memory technology achieves DRAM-like latency and endurance and achieves a record density higher than Micron's leading DRAM technology [97]. A second major advance-

ment for 1T-1C cells is related to the possibility to read data stored in the capacitor in a non-destructive way [99].

In contrast, the ferroelectric field-effect transistor (FeFET) enhances scalability by amplifying ferroelectric dipoles via the transistor's field effect. Despite this, FeFETs suffer from high write voltage and poor endurance respectively due to the voltage division between the ferroelectric and channel layer and charge trapping at the ferroelectric/channel interface [100]. The one-transistor metal–ferroelectric–metal field-effect transistor (1T-MFMFET) addresses the high write voltage issue by optimizing the area ratio between the capacitor and the transistor, though its retention is affected by leakage current into the floating gate node [101]. Recently, a two-transistor MFMFET (2T-MFMFET) has been introduced to enhance the scalability of embedded ferroelectric memory in advanced logic nodes. This configuration decouples read and write paths, improving data retention compared to the 1T-MFMFET solution [102]. At present, small scale macro have been presented for these technologies, meaning that further study is still necessary to validate the technology potential.

Finally, ferroelectric tunnel junctions (FTJs) have emerged as a memristive technology based on ferroelectric thin films, sandwiched between two metal electrodes. FTJs are two terminal devices where the ferroelectric material is employed as a tunneling barrier. The effective barrier height and width can be modulated by changing the direction of polarization, resulting in a high or low conductance state. Moreover, the tunneling barrier can be gradually switched by partial domain switching of polycrystalline ferroelectric layer, for multi-bit storage applications. Therefore, FTJs are characterized by a simple fabrication process, low-power operation, fast switching and nondestructive sensing. Nevertheless, the practical implementation of arrays remains elusive because of sneak-path currents during programming as well as low read currents [103, 104].

Resistive memories. Resistive RAM, or RRAM, typically refers to a non-volatile memory based on an insulating metal oxide resistive switching layer sandwiched between two metal electrodes. The concept of a resistive memory cell is apparently quite simple; nevertheless, the switching behaviour depends not only on the oxide materials but also on the choice of metal electrodes and their interfacial properties. In RRAMs, the nominally insulating layer can switch between a low conductance state (LCS) and a high conductance state (HCS). The switching event from LCS to HCS is called set operation, whereas the switching event from HCS to LCS is called reset. Before set/reset cycling, a voltage larger than the set voltage is needed to trigger the resistive switching behaviours for the subsequent cycles. This is called the forming process. Recently, forming-free metal-oxide memory stacks relying on bulk switching have also been investigated for their enhanced compatibility with more advanced technology nodes, due to the absence of the need for a larger voltage electroforming step [105]. The switching modes of metal–oxide RRAM can be broadly classified into two classes: unipolar and bipolar. Unipolar switching means the switching direction depends on the amplitude of the applied voltage but not on the polarity of the applied voltage. Thus, set/reset can occur at the same polarity. Bipolar switching means the switching direction depends on the polarity of the applied voltage. For either switching modes, a selector device enforces a set compliance current in order to avoid a permanent dielectric breakdown in the set process. The switching mode depends on the choice of the resistive switching layer and electrodes [106].

RRAM devices typically have limited write endurance at array level (up to one million cycles), even though the intrinsic switching endurance has been proven to be much larger. Write energy consumption are in the order of the pJ per bit. The read endurance of the device is virtually unlimited and low-power sensing strategies can be efficiently designed [85, 86].
RRAM is a promising low-cost solution for next-generation embedded and stand-alone memory applications. Nevertheless the intrinsic variability of the switching process has blocked the commercial interest of this memory technology. The pronounced variability and related resistance distribution spread for high and low resistive states reduces the read window margin and limits the maximum memory capacity that can be achieved. Currently, there is no commercial product based on metal-oxide resistive memory technology. The company WeebitNano has announced to be working on a stand-alone memory based on HfO_2 resistive memories, but no commercial product is yet available. Several proof of concept have been presented for embedded RRAMs.

1.2.6 Memory requirements for ANN accelerators

The previous description of the different memory technologies aimed at depicting the current advancement state of the memory technology panorama, highlighting the concept that at present, a universal memory technology with optimal properties does not exist. Each memory technology has advantages and properties that can be partially tuned by device optimization. Nevertheless some intrinsic characteristic cannot be modified, thus limiting the application domain for each kind of memory device.
Hardware accelerators for deep learning can be grouped in two main classes depending on whether such systems have learning capabilities or not, thus limited to inference-only applications.
As previously mentioned, at inference time, the parameters of a neural network are fixed. Thus, whatever the strategy to perform the successive multiplications between the vector of activations and the matrix of weights is, inference does not require any modification of the network parameters. Therefore, a device with the possibility to perform extensive read operations reliably and at low energy cost would be optimal in this case.
On the other hand training requires successive modifications of the parameters of the network with new input data. In most cases, modifying the network parameters requires knowledge of the parameter value itself. This means that training not only requires reliable and efficient write operations, but as-reliable and efficient read operations too.
Implementing both inference and training with a memristive technology requires going beyond the basic considerations on energy consumption and consider other performance metrics, i.e. yield, retention, number of analog states, on/off ratio, symmetry, linearity and endurance (Figure 1.9). For a memristive device, yield can be defined as the average percentage of manufactured devices that function with analog switching behaviour. Achieving high yield memristive devices is quite complex, because this not only includes stuck-at faults but also incorrect intermediate state programming or abrupt conductance changes. High yield is crucial in inference-only applications as misprogrammed weights might have significant negative impact on the achievable accuracy of the network. On the other hand, high yield can be less critical for training as artificial neural networks can keep into account faulty devices throughout training, by modifying other fully functional synaptic connections. Good retention is also essential for inference-only application,

where for retention it is meant the stability with time of the programmed conductance. Conversely, for training itself the requirement on retention might be less stringent as training requires subsequent modification of the programmed state. In order for these modification to be applied, a sufficiently large number of analog states must be guaranteed, whereas inference can run on lower bit equivalent models. Linearity and symmetry are also important aspects related to the possibility to easily potentiate or depress the programmed synaptic weights during training. Specifically with linearity it is meant a direct proportionality between the conductance value and the number of applied pulses. Symmetry, instead, refers to the different trajectory that the conductance increase might follow with respect to the conductance depression as function of the number of applied programming pulses. Finally, endurance is a key property to keep into account when designing memristive-based training and inference ANN accelerators, for both read and write operations [4]. Three scenarios can be foreseen for such accelerators:

- **Inference-only:** An inference-only engine based on a memristive technology needs to load the parameters of the neural network ideally only once. After this initial programming, the weights are available on chip indefinitely (in the limit of the data retention bound). Therefore, a large write endurance is not at all mandatory. Instead, read operations are be performed for the whole lifetime of the system, thus requiring sufficiently large read endurance.

- **Punctual (one-time) training and inference:** A system might be trained only once in its lifetime for a given task and then perform only inference on new input data. In this case the write endurance limit of the memristive technology should be sufficiently high to support the number of updates necessary to train the network for the desired task. This upper bound write endurance depends on the size of the training dataset and the weight update strategy. In the context of back-propagation-like learning, if a maximum of k programming operations are applied to update the weight stored in each memristor for each update event, and the network is trained for e epochs with n update events for each epoch, the write endurance limit should be larger than $k{\cdot}e{\cdot}n$ to support the whole training procedure. As before, read operations are performed for the whole lifetime of the system, thus requiring sufficiently large read endurance.

- **Continual training and inference:** Lifelong training is not only a challenge at the algorithmic level, as shown earlier in this chapter, but also from an implementation perspective with memristive-based architectures. This scenario is indeed the most demanding in terms of endurance requirements for both read and write operations. Training might be performed several times during the lifetime of the circuit, thus multiplying the endurance limit evaluated in the case of a one-time training case by an ulterior factor. A near-sensor learning engine can be envisioned in which training events are not punctual in time, but are spread across the whole time operating window of the system, further increasing the write endurance requirements. Again, read operations must be ensured at all time, calling for a memristive device with sufficiently large read endurance.

Among the previously analysed memristive technologies, i.e. RRAM, MRAM, PCM, FeFET and FTJ, none has all the above-mentioned characteristics, therefore allowing the

deployment of an hardware accelerator with learning capabilities. In the next sections some examples of ANN accelerators are analysed, either for inference-only applications or inference and learning, showcasing the need of a hybrid memory technology in the latter case.

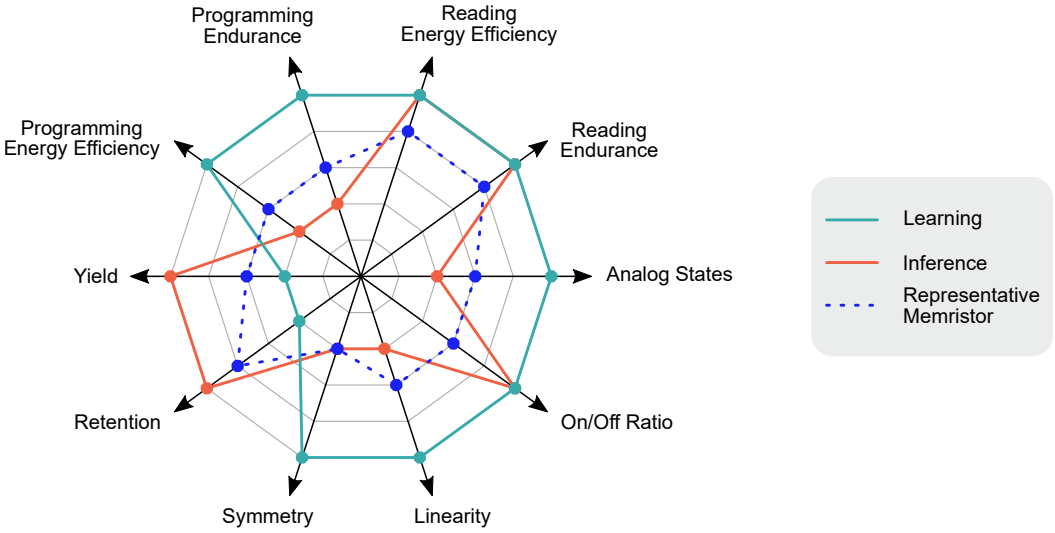

Figure 1.9: **Memory device requirements for ANNs learning and inference.** Ranking of qualitative device requirements ANNs implementation. Higher values along each axis signify greater demands for the corresponding metric. Reproduced and adapted from [4].

1.2.7 NVM-based inference accelerators

There is a large amount of research works utilizing NVMs for inference acceleration, exploiting various memory technologies. Providing a comprehensive listing of all works in the field is somehow impractical and outside the main focus of this work. Nevertheless, the implementation of training on-chip with NVMs, which is the goal of this thesis, necessitates implementing the inference step too. Therefore, in the following, some major contributions in the field of NVM-based inference accelerators are discussed, presenting the key ideas of each work, some of which are also relevant for this thesis.

PCM-based in-memory computing architectures. IBM Research has been working for several years now on the development of IMC architectures exploiting phase-change memories. These efforts resulted in two main realizations, a mixed-signal solution [107] and a fully-analog one [108]. The former, i.e. the Hermes project chip, is an analog-IMC processor based on 64-cores. The chip is developed in the 14 nm CMOS technology node with BEOL-integrated PCMs. The unit cell includes four PCM devices to encode a synaptic weight in analog precision. The chip allows to perform highly parallel matrix-vector multiplications by utilizing one current-controlled oscillator ADC for each unit-cell row, thus decreasing latency and increasing throughput [109]. The near-software accuracy proven for several tasks is obtained thanks to a hardware-aware training strategy, taking into account device non-idealities and a realistic crossbar model [110]. The chip includes

high-precision digital units for all other computation required by other network architectures, like convolutional and long-short term memory layers. A second solution proposed an analog-AI chip, combining 35 million PCMs across 34 cores. The communication between tiles is highly parallel and it is performed in the analog domain, without the need of ADCs. Finally, the same research group showed the need for synaptic weights to be fully stationary, meaning that every weight must be pre-programmed once before inference workload execution begins. Indeed, emerging NVM devices are typically characterized by finite endurance and slow, power-hungry programming, thus preventing the possibility of frequent re-programming. Even though a fully-weight-stationary system occupies more area, compared to a partial-weight-stationary system in which subsets of weights have to be reloaded, the former offers much more throughput, effectively increasing the area efficiency [111].

MRAM-based inference engines. MRAM is another candidate for memristor-based IMC inference accelerators. Nevertheless, compared to other NVMs, MRAM have lower resistance values and small high-to-low resistance ratios, making the implementation of standard crossbar-based multiply-accumulate operation energy-consuming, thus reducing any advantage brought by an in-memory computing implementation. One possible approach is to use a resistance summation, rather than a current summation in order to perform the accumulate operation in the analog domain. This was implemented with a modified bit-cell design in a small scale crossbar array of 64x64 devices. This result demonstrates for the first time the implementation of an MRAM-based IMC engine for ANN inference [112]. Most of the other approaches using MRAM devices develop bit-wise logical operations within the memory for the implementation of digital IMC systems [113, 114, 115, 116].

FeFET arrays for in-memory computing. FeFET devices were also recently used for the first time to demonstrate a crossbar multiply accumulate operation between 2-bits inputs and 2-bits weights. Contrary to analog computing implementations of the multiply-accumulate macro, the presented approach does not perform direct analog multiplication of the input and weight, which is highly prone to variations and requires a high degree of linearity in the stored states. Instead, a current-limited cell is used such that each cell that is activated has the same current contribution, which limits the impact of variations and improves operation accuracy. Each cell in the crossbar structure uses a 1FeFET-1R configuration, which includes a single FeFET and a single resistor. In this macro, the FeFET cell acts as a memory for the entire weight value. The input is encoded in applied voltage duration and magnitude, the multi-bit weight is stored in the FeFET, and the output is accumulated as a capacitor voltage that depends on the activation time and number of FeFETs activated [117]. Other works with FeFET devices considered only binary logic operations to compute a single-bit multiplication operations [118, 119, 120].

Filamentary resistive memory for in- and near-memory computing. Filamentary RRAM are highly researched for the implementation of neural network inference, because they offer a simple, affordable solution for non-volatile weight storage, despite device non-idealities. For example, high-yield, uniform RRAM crossbar arrays have been

used for the implementation of CNNs. This solution integrates eight 2,048-device arrays, in the 130 nm CMOS technology node, to improve parallel-computing efficiency. Moreover, a fine-tuning technique is proposed in order to adapt the system to the device imperfections. This fine-tuning training is performed in place and it adjusts some of the synaptic weights in the network in place to recover the accuracy loss after weight mapping [121]. A solution similar in principle to the previous one, yet more mature, was proposed by Stanford university. It is a 48-core filamentary RRAM-IMC processor, with 3 million devices integrated in the 130 nm CMOS BEOL. The neuron circuitry is designed in such a way to allow a bidirectional transposable neurosynaptic array operation, for the implementation of forward and backward passes with minimal area and energy overheads. The need for the forward and backward passes is due to the designed hardware-algorithm co-optimization technique used to mitigate the impact of hardware non-idealities on inference accuracy. Indeed, a chip-in-the-loop progressive fine-tuning technique is used, meaning that weights are progressively mapped onto the chip one layer at a time. Then, the hardware-measured outputs from layer one are used as inputs to fine-tune the remaining successive layers, thus requiring both forward and backward passes to be implemented on the-chip itself [122]. The limited endurance, power-consuming programming or RRAMs can in principle support these fine-tuning strategies. Nevertheless, this procedure would require, in the long-term vision, that any manufactured chip should undergo this fine-tuning step before deployment, which significantly increases the complexity of the approach.
It is clear that IMC strategies, although appealing, deal with significant challenges due to device non-idealities and fundamental limitations due to voltage drops on the array lines, issue exacerbated for filamentary RRAMs because of their larger conductance values compared to other technologies. For this reason, RRAMs have also been explored for the implementation of near-memory computing inference accelerators. RRAM arrays, coupled with specific sense amplifiers can implement the multiplication between binary inputs and binary weights, which corresponds to a logic XNOR operation [123]. This approach inspired part of one of the two circuit realizations presented in this thesis and it is further discussed in chapter 4. This same NMC approach has also been used to prove image classification in a near-sensor setting. Indeed, the chip was used in conjunction with a miniature wide-bandgap solar cell optimized for edge applications as energy harvester, making a step forward in self-powered AI [124].

1.2.8 NVM-based inference and training accelerators

The previous section highlighted the fact that the implementation of inference with IMC systems can be challenging because of device non-idealities. This problem is less relevant in the case of systems with learning capabilities. By learning on-chip, the network architecture should automatically learn how to deal with hardware imperfections and compensate for some of them, up to a point [125]. In the last years, more and more research works started addressing the implementation of training on-chip exploiting NVM devices, because of their promise for improved energy and area efficiency [4].

ANN training with PCM devices and volatile capacitors cells. One major step towards the implementation of training on-chip with memristive devices was introduced

by Ambrogio et al. in [126]. Here, PCM devices were used to store synaptic weights in analog precision to implement the typical MVM operation at inference time. Device non-idealities, reduced number of analog states, as well as limited endurance and large programming power make the implementation of learning with PCM devices only ineffective. Therefore, a volatile capacitor is used to store lower significance weights and to perform near-linear updates. The volatile capacitors are updated for several training samples, before transferring the lower significance weight updates to the higher significance weights stored in PCM devices. Weight-data transfer is performed with polarity inversion for each update, to cancel out inherent device-to-device variations in the volatile capacitors. The transfer process is performed in a closed-loop, iterative fashion, resulting in accurate weight tuning at the expense of increasing energy-cost. One main drawback of this approach is the need for large capacitors in order to obtain the target of at least 1,000 resolvable states for weight updates [127]. In the proposed implementation of the capacitive unit cell in 90 nm node uses, the gate capacitance of a metal-oxide semiconductor field-effect transistor (MOSFET) is used, leading to a gate area of $2.2\,\mu m^2$. Mixed hardware-software simulations showed test accuracy close to the software baselines for the MNIST, CIFAR-10 and CIFAR-100 datasets. A similar approach is discussed in [128] with projections for a system level implementation, confirming the same conclusions.

Edge training with filamentary memristors. A different solution exploiting filamentary resistive memories was proposed by Zhang et al. in [129]. From a device technology point of view, the memristor device used a material stack of $TiN/HfO_x/TaO_y/TiN$, compatible with the standard CMOS process. The memristors were integrated in the CMOS BEOL with an excellent yield (almost 100% over 160,000 devices). The fabricated memristors exhibited uniform and repeatable bidirectional analog switching with identical pulse trains. The 160,000 total on-chip memristor cells could be uniformly programmed to 32 conductance states with average success rates of 99.90%. The unit cell includes resistive memories in a differential configuration, i.e. a 2T-2R structure with a shared source line to implement current summation in the analog domain, effectively reducing the voltage drop across the line. The manufactured chip includes controllers for configuration, a two-transistor-two-resistor (2T-2R) array, line drivers for computing and programming, low-cost analog-to-digital converters, modules for error and updates evaluation and input and output buffers. The employed training algorithm features sign-based weight update calculation, meaning that the differential conductance can be increased or decreased by discrete amount for each update. This approach requires reliable memristive devices with sufficiently linear updates and a large-enough dynamic range, for sufficient precision. The error calculation, which results in ternary values, at different layers is implemented with reconfigurable thresholds. The tuning of these threshold is essential to improve accuracy and it impacts the number of updates implemented in the resistive memory arrays. The updates are implemented in a row-wise parallel fashion. Even though the reconfigurable thresholds strategy decreases the amount of updates performed on the memristors, it might not be sufficient for extensive training throughout the device lifetime. Indeed, even for a small dataset like MNIST, approximately 500,000 programming pulses are applied to each device, which is already demanding for typical resistive memory devices. The proposed approach is applied to other training settings, such as a fine-tuning process or a class-incremental learning task which require less iterations, thus more adapted to the device technology.

Combining digital and analog IMC for training on-chip. An alternative solution to the hybrid memory challenge of learning at the edge would be to combine the high energy efficiency and storage density of memristors with the high accuracy of digital SRAM, as SRAM is built from standard transistors. This was proposed by Wen et al. in [130]. The key idea of this approach is to exploit digital (SRAM-based) and analog (memristor-based) IMC for different parts of the network, trading off high accuracy of digital IMC with energy-efficiency of analog IMC, according to the sensitivity of the network layer to errors.
This fusion-IMC approach has therefore three operating modes, i.e. memristor-IMC, mixed-device IMC, and SRAM-IMC. Memristor-IMC and mixed-device IMC modes are suitable for layers that are less sensitive to readout accuracy degradation but require a massive number of dot product operations, thereby necessitating high energy efficiency and high computing throughput. The memristor-IMC mode with mult-level memristors provides the highest energy efficiency. However, it suffers the most computing accuracy degradation because of process variation and a data-retention time. The mixed-device IMC mode is suitable for layers that depend on high computing accuracy as well as dot product operations with high energy efficiency and computing throughput. The pure SRAM-IMC mode is suitable for layers that require dot product operations of ultrahigh accuracy, such as fully connected layers. Adaptive local training capabilities are also included in this chip. This training is supported only in the digital-IMC mode, in order to remove the need to adjust weights stored in the memristors. After the training is performed the weights stored in the SRAM are redistributed to the memristors, if required by the operating mode of the network layer. This work demonstrates that memristor technology has the potential to be used beyond in-lab development stages and now has manufacturability for AI edge processors. However, in advanced CMOS nodes, SRAM has a considerable area footprint, which is problematic for learning systems that require large amounts of memory. Additionally, as a volatile memory, SRAM presents limitations for lifelong training, which can require non-volatile storage solutions for weights.

Other approaches. Alternative solutions based on novel, less mature technologies have been proposed. For example, the 3-D monolithic integration of memristors for in-memory computing, ternary content-addressable memory arrays, and buffers have been explored to facilitate one-shot learning [131]. Another approach features a duplex device structure with a FeFET with a monolayer MoS_2 channel, suggested to decouple training and inference operations, demonstrated in small 8×3 device arrays [132].
Nevertheless, these approaches face significant challenges, including material compatibility issues, thermal budgets during fabrication and cost issues during the co-integration of distinct memory technologies on the same die, as well as the integration of novel, more complex device structures requiring new masks and process steps.

This thesis's approach to on-chip ANN inference and training. The previous analysis of the current research landscape regarding memristor-based inference accelerators highlighted that the most mature solutions for such kind of systems are based on either PCMs or RRAMs. However, the limited endurance, high programming energy, and intrinsic non-idealities of these devices restrict their suitability for tasks requiring frequent updates, such as on-chip learning. As a result, the implementation of ANN training often

demands the integration of memristive elements with complementary technologies that support frequent updates, such as DRAM-like capacitors or SRAM-based devices.
This thesis builds upon these hybrid approaches by proposing the use of ferroelectric capacitors in conjunction with resistive memory devices to enable efficient implementation of both training and inference. Ferroelectric capacitors offer several compelling advantages for learning applications, including superior endurance, low programming energy, excellent scalability at advanced technology nodes and non-volatility. However, their destructive read process makes them unsuitable for inference. For this reason their integration with resistive memory devices could allow to develop systems with optimal inference and training capabilities. As shown in the next chapter, these two memory technologies can be easily co-integrated within the same BEOL of a foundry CMOS process, thus simplifying their co-manufacturing. Proof-of-concept circuit implementations and co-developed learning strategies are later presented, validating the potential of this approach.

1.3 Summary

This chapter introduces key concepts related to both algorithmic and hardware solutions in the context of artificial neural networks, that are central to the work presented in this thesis. The first section focuses on algorithmic approaches, starting with an in-depth explanation of learning through back-propagation of errors. This section explores the foundational principles behind back-propagation, as well as the various common techniques and best practices that have been developed over time to enhance the overall training efficiency and quality of artificial neural network models. Additionally, the challenges of implementing these algorithmic solutions in resource-constrained hardware environments are examined, emphasizing the trade-offs between computational complexity and hardware limitations. In the second part of this chapter, the focus shifts to the hardware perspective, where an overview of state-of-the-art hardware solutions for implementing ANNs is provided. Special emphasis is given to in-memory and near-memory computing architectures, which have gained significant attention for their potential to overcome the von Neumann bottleneck and improve energy efficiency in both neural network inference and training. These solutions are typically based on emerging non-volatile memory technologies, which offer promising advantages in terms of speed, scalability, and energy consumption. After comparing the properties of different non-volatile memory technologies, the current progress in integrating these memory technologies into neural network hardware accelerators is discussed, highlighting some of the most mature solutions available today. Exemplary inference and training hardware accelerators are presented, with particular focus on the latter, where the need for a hybrid memory technology – capable of handling both low programming energy, large endurance and read stability – is increasingly recognized as critical to advancing the hardware implementation of neural networks.

Chapter 2

Hybrid ferroelectric/resistive memory technology

The successful implementation of learning-capable deep learning hardware accelerators requires a memory technology with exceptional performance. However, no current memory technology combines all the desirable features for these systems. Resistive memory – also known as memristor – arrays are ideal for inference [121, 122] but suffer from limited endurance and high programming energy. Conversely, ferroelectric capacitors are ideal for learning [133], due to their higher endurance and lower programming energy, but their destructive read process makes them unsuitable for inference.
In this chapter, a unified memory stack which functions as both a memristor and a ferroelectric capacitor is presented. This chapter provides a detailed analysis of these two memory technologies, with a particular focus on ferroelectric and filamentary resistive memories based on hafnium oxide. First, the operation of hafnia-based ferroelectric capacitors and resistive memories is examined, highlighting the differences and similarities between the two technologies in terms of switching phenomena, process technology, device and array implementations. Next, the unified memory stack is introduced, accompanied by extensive electrical characterization in both operating modes. Finally, perspectives on device optimization and future outlooks are discussed.

2.1 Hafnium oxide

Hafnium oxide – HfO_2 – appeared in the CMOS technology scene since early 2000's, when it was researched as a possible candidate to substitute silicon dioxide (SiO_2) for metal gate dielectrics.
Up to the 45 nm CMOS process technology node, scaling SiO_2-based dielectrics was essential for enhancing transistor performance in line with Moore's Law. However, as devices approached sub-45 nm nodes, the required effective oxide thickness for SiO_2 had to be less than 1 nm, which is about 3 monolayers and near the physical limit. This resulted in significant gate leakage currents due to quantum tunneling effects. To continue scaling down transistor dimensions, high-κ dielectrics were proposed as a solution to achieve

comparable transistor performance while maintaining a thicker physical layer. Among the various candidates to replace SiO_2, hafnium-based oxides emerged as the most suitable dielectric materials due to their superior overall performance [134]. Since 2007, Intel – followed by other major semiconductor foundries – has been employing HfO_2 as high-κ dielectric for its more advanced technology nodes, making the integration of hafnium oxide with CMOS technology a standard, well-established practice [135].
Since its introduction as high-κ dielectric in metal gate MOSFETs, HfO_2 has also emerged as one of the most promising candidates for the development of next generation non-volatile memory devices. In particular, resistive [136] and ferroelectric [96] switching phenomena have been observed in similar HfO_2 films. Despite relying on similar material substrates, the underlying switching mechanisms of the two memory technologies are substantially different, as analyzed in the following.

2.2 HfO_2-based ferroelectric memory technology

Ferroelectric capacitors, often referred to as FeCAPs, consist of a ferroelectric thin film sandwiched between two symmetric metal electrodes. The ferroelectric film in FeCAPs can switch between two polarization states, i.e., up or down polarization, through the application of an external electric field, thereby encoding digital information. The polarization of the ferroelectric dipoles remains unchanged after the power is removed, providing the device with non-volatility.
The polarization - electric field hysteresis loop (Figure 2.1a) characterizes the operation of a FeCAP device. It is defined by three quantities: the remanent polarization P_R, i.e. the residual polarization when the applied field is removed; the saturation polarization P_S, i.e. the maximum amount of polarization that can be induced in the material at high electric field; and the coercitive field E_C, i.e. the field value for which the polarization is switched between negative and positive values. The most standard characterization procedure to evaluate the hysteresis loop is the double-wave method, also known as PUND (positive up negative down) technique [137]. It consists in applying triangular voltage pulses to one of the electrodes of the ferroelectric capacitors, twice for each pulse polarity, and measuring the capacitor current response. Doubling the number of pulses for each polarity allows to to discriminate undesirable components (dielectric and linear response) from the ferroelectric one. Indeed, the components that do not participate in the hysteresis loop are automatically subtracted from the ferroelectric one without assumptions. The polarization - electric field hysteresis loop is therefore obtained by integrating the current response of the capacitor.
Several materials that exhibit a polarization-electric field hysteresis loop have been investigated for memory applications. Among them, HfO_2-based ferroelectric materials are the most promising due to their superior compatibility with advanced CMOS technology nodes.

2.2.1 Ferroelectricity in HfO_2 thin films

In 2011, it was discovered that thin films of hafnium oxide doped with SiO_2 can exhibit ferroelectric behaviour. Among others, two major factors allowing the stabilization of the

ferroelectric phase in HfO_2 are the presence of an encapsulation layer (a top electrode) and the SiO_2 concentration. It has been observed that hafnium oxide has three distinct crystal phases under ambient pressure: a monoclinic phase at room temperature, a tetragonal phase above 2050 K, and a cubic phase above 2803 K. In nano-scale crystallites, surface energy effects lower the temperature range of the stable tetragonal phase. For this reason, crystallization in thin films often starts with tetragonal phase nucleation, followed by a transformation to the monoclinic phase as the crystals grow. The three above mentioned phases are centrosymmetric and they do not allow the creation of semi-permanent dipoles in hafnium oxide.
It has been observed that larger SiO_2 concentration, between 5 and 10 %, stabilizes the tetragonal phase, whereas mechanical encapsulation during crystallization can hamper the creation of the monoclinic phase. Therefore, a new phase can be obtained by lowering the SiO_2 concentration below the one stabilizing the tetragonal phase and encapsulating the material during crystallization. As shown in Figure 2.1b, this transformation leads to a polar (non-centrosymmetric) orthorombic phase. Specifically, films that are 10 nm thick and contain less than 4 % SiO_2 can crystallize into an orthorombic phase. This orthorhombic phase displays a notable piezoelectric response, with polarization measurements indicating a remanent polarization above $10\,\mu C/cm^2$ and a coercitive field of 1 MV/cm, thus suggesting ferroelectric characteristics [96].
The hysteresis loop of a FeCAP evolves with cycling, in particular the remanent polarization shows two distinct trends (Figure 2.1c):

- Wake-up phase: the remanent polarization memory window, defined as the difference between positive and negative remanent polarizations ($2P_R$ in Figure 2.1c), increases with cycling. This increase of the remanent polarization corresponds with an opening of the pristine hysteresis loop with field cycling;
- Fatigue phase: after a certain number of cycles, the remanent polarization hysteresis decreases resulting in a closure of the memory window due to aging mechanisms.

A stable phase can occur between the wake-up and fatigue phases, during which the remanent polarization memory window remains relatively stable with field cycling.
The wake-up and fatigue phenomena arise from a complex interaction of ferroelectric and dielectric properties. In the wake-up phase, already existing defects in the ferroelectric film are redistributed, decreasing the defect induced built-in field and creating a uniform field distribution. Additionally, field-induced phase transitions can transform the initially pristine monoclinic-phase to a woken-up orthorhombic-phase with cycling. Modeling the diffusion mechanisms of ions and vacancies, along with phase transitions, has demonstrated that these factors contribute to the disappearance of the built-in field, the formation of a more uniform field within the device, and an increase in the volume fraction involved in the switching process. Moreover, dielectric degradation has been identified as the primary cause of limited endurance in doped HfO_2-based ferroelectric capacitors. An increase in trap density with device cycling indicates that trap generation, along with domain pinning, is the main mechanism behind the degradation of ferroelectric behaviour. This trapping occurs both in the interfacial regions near the electrodes and within the bulk. Electron trapping at these defects alters the field distribution within the stack, reducing the field in the bulk of the ferroelectric layer. Further generation of vacancies leads to the creation of leakage paths, eventually resulting in stack breakdown before the

memory window fully closes [138].
Polarization recovery techniques have been recently proposed in order to extend the endurance of ferroelectric devices. Positively charged oxygen vacancies formed at the interface between the ferroelectric film and electrodes can cause domain pinning and the fatigue can be recovered by applying a high stress field. As a result, the recovery effect is attributed to domain depinning and the switching of new domains due to the redistribution of oxygen vacancies [139].

2.2.2 Process technology

HfO_2-based ferroelectric capacitors can be integrated in the BEOL of CMOS technology, for increased integration density. This is possible thanks to the compatibility of the fabrication process with the thermal budget of the CMOS back-end of line. The process integration relies on standard deposition techniques in microelectronics, such as physical vapour deposition (PVD) and atomic layer deposition (ALD), for the metal electrodes and HfO_2 layers. Specifically, the thickness of the HfO_2 layer has to be precisely controlled to ensure the stabilization of the ferroelectric phase. A trade-off between crystallization temperature in the orthorhombic phase and film thickness exists, preventing significant scaling of the HfO_2 thickness below 10 nm, which would be beneficial to reduce the switching voltage and, as a consequence, the power consumption of the device. The doping control is also essential to achieve stable ferroelectric behaviour. Ref [140] reported the integration of 10 nm Si-doped (1% mean concentration) HfO_2 devices, sandwiched between two titanium nitride (TiN) metal electrodes, between the fourth and fifth metal layers of a 130 nm foundry process, while keeping the process temperature below 450 °C.
Beside Si-doped HfO_2, other dopants allow the stabilization of the ferroelectric phase, such as yttrium (Y) [141], gadolinium (Gd) [142], aluminium (Al) [143], strontium (Sr) [144], lanthanum (La) [145], and others. Moreover, a mixture of zirconium and hafnium oxide (HZO) has also shown excellent ferroelectric properties [146]. Ref. [147] reported for the first time BEOL integrated TiN/HZO/TiN capacitors, in the 130 nm CMOS node, proving the possibility to scale down the capacitor dimensions to 300 nm diameter. Excellent performance were reported, such as remanent polarization memory window larger than 40 $\mu C/cm^2$, endurance larger than 10^{11} cycles, switching speeds below 100 ns, operating voltages below 4 V, and 10-years data retention at 125°C. Since then, major advancements have been reported on HZO ferroelectric stacks [148, 97, 149], making HZO the most optimal ferroelectric material from both performance and reliability perspectives at present [150].

2.2.3 1T-1C FeRAM arrays

HfO_2-based FeCAPs can be serially connected with a selector transistor (one transistor one capacitor cell, 1T-1C) and organized in an array configuration for memory applications (Figure 2.2 a). Such arrays are often referred to as FeRAM arrays. A 1T-1C cell can be accessed via three ports, a word line (WL) connected to the gate of the selector transistor, a source line (SL) connected to the top electrode of the FeCAP and a bit line (BL) connected to the source of the selector transistor. The bottom electrode of the FeCAP device is directly connected to the drain of the selector transistor.

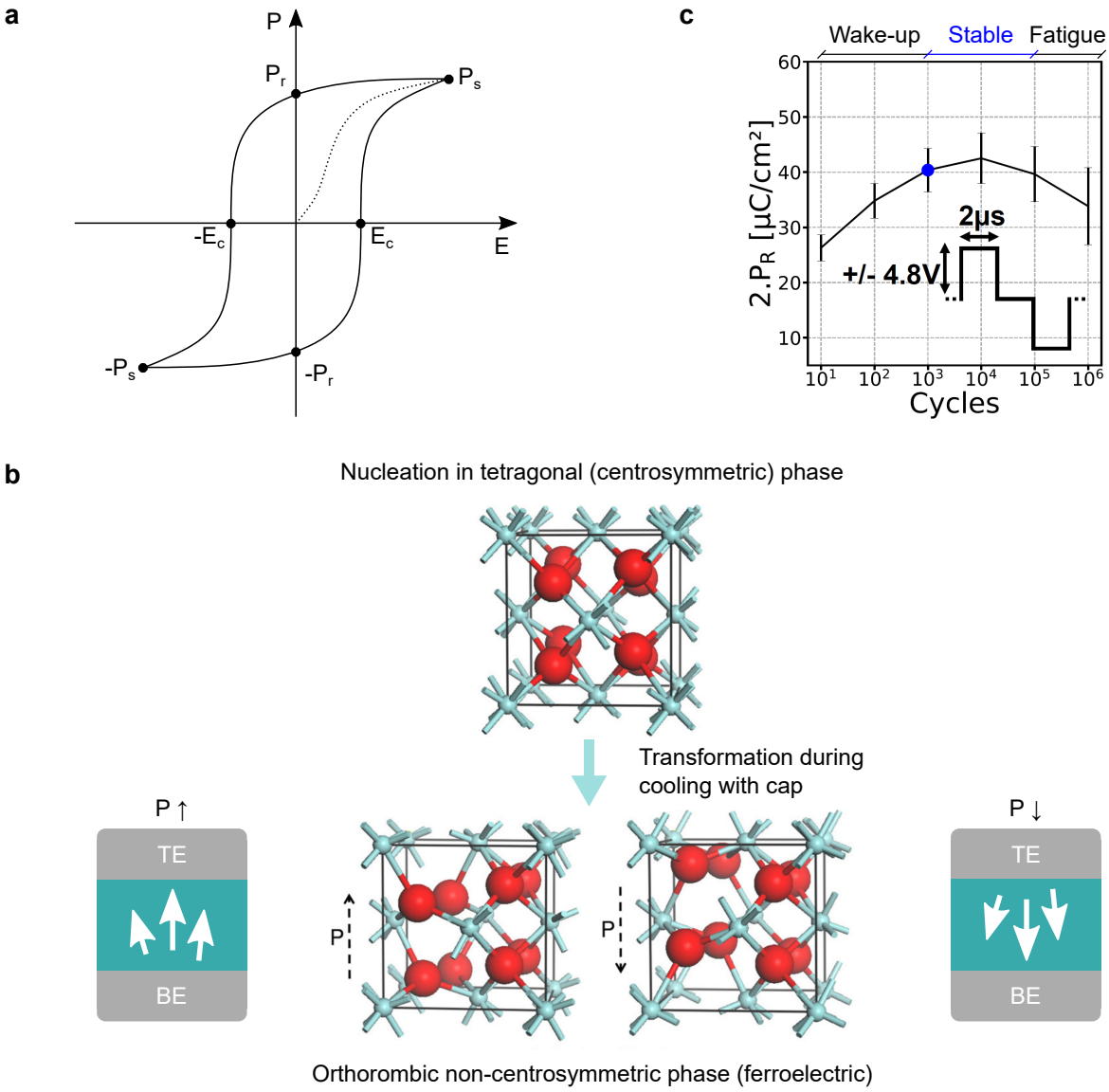

Figure 2.1: **Ferroelectric capacitors concepts.** **a** Polarization - Electric Field hysteresis loop of a ferroelectric capacitor. **b** HfO_2 nano-crystallites (hafnium atoms in light blue, oxygen atoms in red) can be stabilized in a non-centrosymmetric crystalline phase by metal capping and doping control (Reprinted and adapted from [96]). **c** Polarization memory window as a function of cycling. FeCAPs were cycled with square electrical pulses to mimic the electrical signal waveform used for FeRAM arrays characterization. The polarization memory window is extracted using PUND methodology (4.8V-10kHz). Measurements are performed and averaged over five different dies on the same wafer. FeCAPs are made of 10 nm thick Si-doped HfO_2 sandwiched between TiN metal electrodes (Reprinted from [151]).

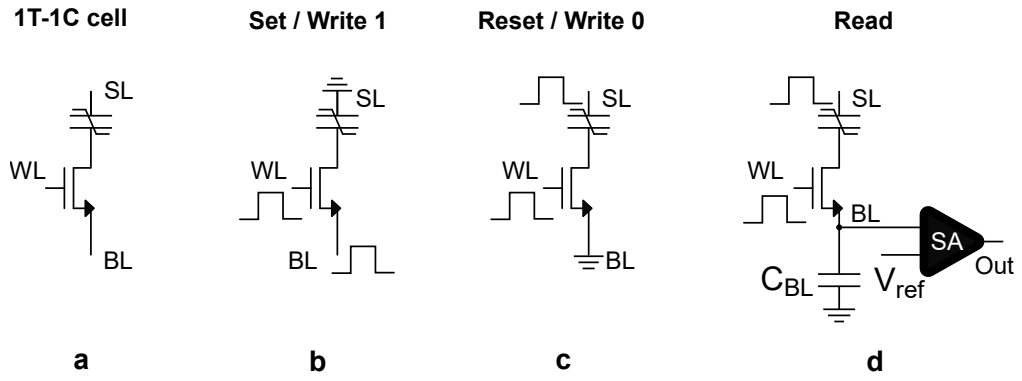

Figure 2.2: **Programming and read operations of FeRAM arrays. a** FeRAM bitcell consisting of a selector transistor connected to the bottom electrode of a FeCAP device (1T-1C structure). **b** Set operation of a 1T-1C cell. **c** Reset operation of a 1T-1C cell. **d** Read operation of a 1T-1C cell.

Performing a set operation, i.e. writing a 1 in the FeCAP, consists in pulsing the BL of the 1T-1C cell while the WL is selected and the SL is precharged or connected to ground (Figure 2.2b). On the other hand, the device can be reset (Writing a 0 in the FeCAP) by pulsing the SL while the WL is selected and the BL is precharged or connected to ground (Figure 2.2c). The set and reset pulse amplitude and duration can be optimized for improving device reliability properties, such as endurance and retention. On the other hand, the WL is typically connected to the supply voltage when selected, as no accurate control of the current flowing in the device is required for programming.
The read operation consists in applying a reset pulse to the device and comparing the BL voltage response to a reference voltage via a sense amplifier (SA), as shown in Figure 2.2d. The output of the sense amplifier (Out) is a 0 or a 1, respectively if the BL voltage is smaller or larger than the reference one. The voltage obtained on the BL, V_{BL}, when pulsing the SL depends on several factors and it can be expressed as:

$$V_{BL} = \frac{C}{C + C_{BL}} V_{SL} \ , \tag{2.1}$$

where C is the capacitance value of the ferroelectric capacitor, C_{BL} the BL parasitic capacitance and V_{SL} the amplitude of the SL pulse. The capacitance C can be written as the sum of a dielectric component (C_D) and a ferroelectric one (C_F):

$$C = C_D + C_F = \epsilon_0 \epsilon_r \frac{S}{d} + \frac{2 P_R S}{V_{SL}} \delta \ , \tag{2.2}$$

where S and d are the area and thickness of the capacitor, ϵ_0 is the vacuum permittivity, ϵ_r is the relative permittivity of the ferroelectric material, P_R is the remanent polarization of the capacitor, and δ is equal to one or zero depending on the polarization state stored into the ferroelectric capacitor. In the approximation $C_F \ll C_D + C_{BL}$, equation (2.1) can be rewritten as:

$$V_{BL} = \frac{C_D}{C_D + C_{BL}} V_{SL} + \frac{2 P_R S}{C_D + C_{BL}} \delta \ . \tag{2.3}$$

Therefore, the BL response while performing a read operation is the sum of a dielectric component and a ferroelectric one, if a polarization switching event occurs during the read operation.

By considering typical values for ferroelectric capacitors, e.g. area S=0.36 µm^2, thickness d=10 nm, relative permittivity ϵ_r=29.7, vacuum permittivity ϵ_r=8.854·10^{-15} F/m, remanent polarization P_r=15 µC/cm^2, applied read voltage V_{SL}=3.5 V and BL parasitic capacitance C_{BL}=200 fF, the error due to the approximation $C_F \ll C_D + C_{BL}$ in equation 2.3 can be evaluated. Indeed, C_D=9.5 fF and C_F=31 fF, which results in a 15% error on the BL voltage evaluated with equation (2.3).
This reading scheme is therefore data-destructive, if polarization-switching occurs during read, and reprogramming to the previous state needs to be performed to restore the initial information. This can be achieved by dedicated circuitry in the sensing element. Therefore, the reading procedure of ferroelectric capacitors links and limits the read endurance to the write one. In the worst case scenario, i.e. reading a 1, a single read operation requires two polarization switching events in order to read and restore data. Reading a zero also requires two operations, but no polarization switching occurs in this case, either during read and write-back.
The typical sensing element for reading the data stored in a ferroelectric device is a clocked latch sense amplifier (Figure 2.3a). When the sensing element is disabled (en signal is low), the nodes $V_{in,a}$ and $V_{in,b}$ are loaded to V_{BL} and V_{ref} respectively. When the sensing element is enabled (en signal is high), the latch switches depending on the relative voltage of the two pre-charged nodes (Figure 2.3b). It is important to notice that this comparator has no dedicated output. Each input $V_{in,a}$ and $V_{in,b}$ can be an output once the comparison is done. In Figure 2.3a, $V_{in,b}$ is used as output node to which a digital buffer is connected, because it provides a logical '1' when V_{BL} is higher than V_{ref}.
Therefore, this sensing element directly reads the BL voltage which depends on the BL parasitic capacitance. This could eventually become a problem for large bank size memories, as the memory window decreases with increasing bit line capacitance, as shown in equation (2.3).
A new charge-based sensing scheme for read operation in FeRAM arrays has been recently proposed in order to increase the median memory window and making it quasi-independent of array size, overcoming the problem of the memory window closure with bit line capacitance increase, observed in FeRAM arrays with conventional voltage-based sense. This new scheme relies on a capacitive trans-impedance amplifier (CTIA)-based sensing, i.e. a two-stage operational amplifier with a feedback capacitance and a reset switch in parallel [152]. From an area point of view, a cross-coupled inverters comparator occupies a 40 µm^2 area while the CTIA sense occupies 630 µm^2 in the 130 nm CMOS technology node, with the possibility to reduce area of a factor 3 for the CTIA-based sense with an optimized design, as claimed in [152]. Depending on the application domain and trade-offs between area and latency, one solution might be more optimal than the other.
Neglecting resistive losses, FeRAM arrays have zero static power consumptions. For this reason, operations in FeRAM arrays can be parallelized, e.g. programming/erasing or reading (depending on the chosen sensing strategy) an entire line at the same time, without incurring in excessive peak power consumption. Moreover, dynamic power consumption can be minimized by optimizing the array configuration, i.e. the relative orientation of BLs, SLs and WLs. For instance, the 1T-1C bit cell described in Figure 2.2 in a square array with N×N bit cells, with one sensing element for each BL, can be analyzed. The power consumption during any operation performed on the array can be evaluated as the sum of the power consumption of the different supplies taking part in said operation:

$$P_{total} = P_{BL} + P_{SL} + P_{WL}, \tag{2.4}$$

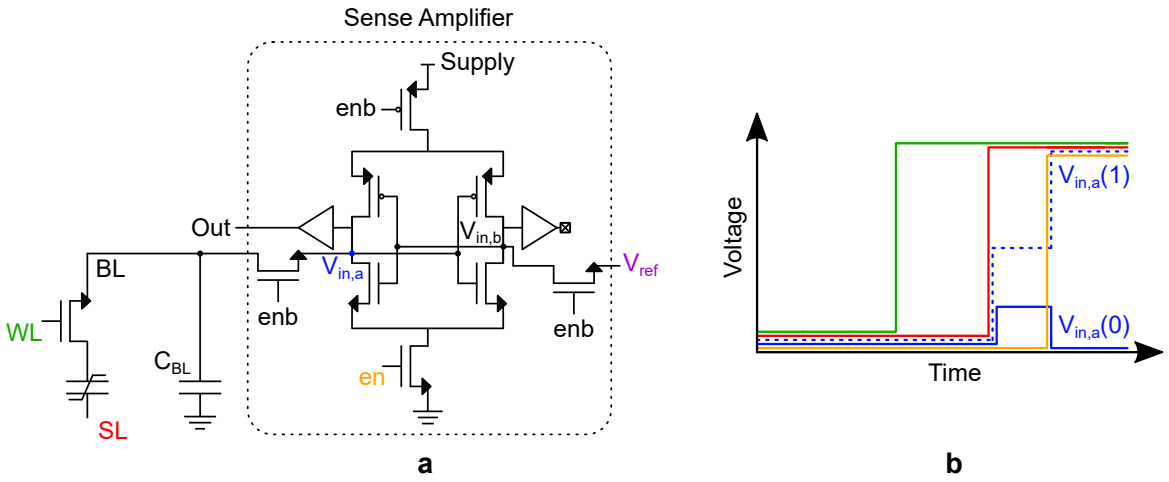

Figure 2.3: **Detailed description of a FeRAM array sense amplifier. a** Clocked latch sense amplifier (SA) connected to one BL of a FeRAM array. The signals en (and its negate version enb) enable (disable) the sensing element. The circuitry to write-back data after read is not shown for clarity. **b** Chronograph of a read operation for reading a 1 (dotted blue line, $V_{in,a}(1)$) or a 0 (solid blue line, $V_{in,a}(0)$).

$$P_{BL} = C_{BL}V_{BL}^2 f_{BL} \tag{2.5}$$

$$P_{SL} = C_{SL}V_{SL}^2 f_{SL} \tag{2.6}$$

$$P_{WL} = C_{WL}V_{WL}^2 f_{WL} \tag{2.7}$$

where C_{BL}, C_{SL}, C_{WL} are the capacitive loads of the supplies V_{BL}, V_{SL}, V_{WL} respectively and f_{BL}, f_{SL}, f_{WL} the average switching frequencies of the signals propagating through the three sets of lines. Three different array configurations are possible in terms of relative line orientation. Each configuration has a different impact on the load of the different supplies and therefore the total power consumption. Considering the worst case scenario in terms of peak power consumption, e.g. reading an entire line at the same time, the three different array configurations are analyzed.

- Parallel WLs and SLs and perpendicular BLs: Performing a read operation requires charging a single SL to the programming voltage level and a single WL to the supply voltage.
- Parallel BLs and SLs and perpendicular WLs: Performing a read operation requires charging N SLs to the programming voltage level and a single WL to the supply voltage.
- Parallel WLs and BLs and perpendicular SLs: Performing a read operation requires charging N SLs to the programming voltage level and N WLs to the supply voltage.

Clearly, for each of the three configurations, N BLs switch during a full line read operation. The first option is more energy efficient than the second and third ones for reading an entire line of capacitors in parallel.
A ferroelectric memory array with 128 parallel WLs and SLs and 128 perpendicular BLs, with a clocked latch sense amplifier for each BL is used in section 2.4.3 to evaluate the performance of the proposed unified memory stack as a ferroelectric memory.

2.3 HfO_2-based resistive memory technology

Resistive memory devices – also known in other contexts as filamentary memristors or valence change memories – consist of an insulating thin film, made of a metal-oxide material, sandwiched between two metal electrodes. The insulating material can switch between two conductive states, a high conductance state and a low conductance state, with the application of electrical pulses. Typically, devices in their initial pristine state require a one-off "forming", or "electro-forming", step in order to initiate resistive switching behaviours for subsequent cycles. The transition low-to-high conductance is called set operation, whereas the high-to-low conductance transition is called reset operation. The switching modes of metal-oxide resistive memory devices can be generally divided into two categories: unipolar and bipolar. In unipolar switching, the switching direction is influenced by the amplitude of the applied voltage, regardless of its polarity, allowing set/reset operations to occur at the same polarity (Figure 2.4a). On the other hand, in bipolar switching, the switching direction is determined by the polarity of the applied voltage, meaning that set and reset operations occur at opposite polarities (Figure 2.4b). To prevent permanent dielectric breakdown during the set process in either switching modes, a set compliance current is typically enforced.

2.3.1 Filamentary switching in HfO_2 resistive memory devices

Memory switching in HfO_2-based devices relies on the creation or dissolution of a conductive filament of oxygen vacancies. In the forming process, a soft dielectric breakdown occurs, causing oxygen ions to move towards the top electrode interface under a high electric field. Here, if the electrode is made of noble metals, the ions are released as neutral non-lattice oxygen; otherwise, they react with the anode material to form an interfacial oxide layer. Consequently, the electrode/oxide interface acts as an oxygen reservoir, also known as oxygen scavenging layer. In a memory cell in the high conductance state, current travels through the conductive filaments within the bulk oxide. During the reset process, oxygen ions migrate back into the bulk, recombining with oxygen vacancies, transitioning the memory cell to the low conductance state. In unipolar switching devices, Joule heating from the current thermally activates the diffusion of oxygen ions. These ions diffuse from the interface or surrounding conductive filaments due to the concentration gradient, hence unipolar switching typically requires a higher reset current to increase the local temperature around conductive filaments. For bipolar switching devices, the interfacial oxide layer can act as a significant diffusion barrier, making pure thermal diffusion inadequate. Thus, oxygen ion migration must be assisted by the reverse electric field. The described physical model, summarized in Figure 2.4b, integrates both the thermal model and ionic migration model, providing a phenomenological explanation of the experimental observations for hafnia-based devices, as well as other binary metal-oxide devices.
The unipolar or bipolar operation of HfO_2-based devices strongly depends on the electrodes interfaces. For instance, unipolar switching was observed in $Pt/HfO_2/Pt$ structures [153], whereas using oxidizable interfaces, such as Ti/TiN, favours bipolar operation. Bipolar operation is usually favoured for HfO_2-based devices because of the simpler integration of Ti and TiN electrodes and the lower reset current requirements.

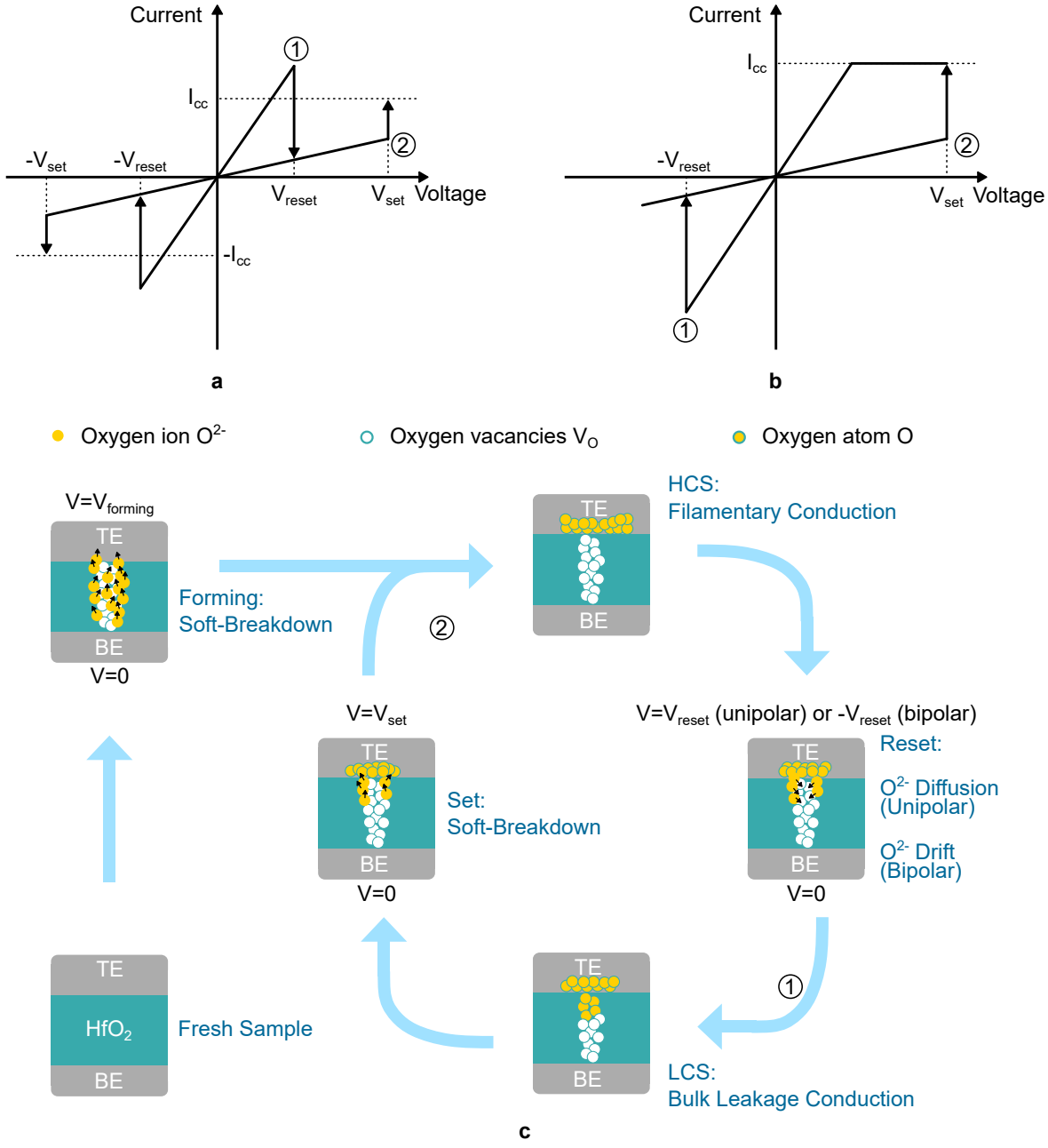

Figure 2.4: **Resistive switching in hafnia-based memristive devices. a,b** Typical current-voltage quasi-static response of a unipolar (a) and bipolar (b) resistive memory device. Circled digits indicate reset (1) and set (2) operations **c** Phenomenological explanation of conduction in metal-oxide resistive memory devices. Reproduced and adapted from [106].

2.3.2 Process technology

Resistive memories can also be integrated in the BEOL of CMOS technology, thanks to the compatibility of the manufacturing process with the thermal budget of the CMOS back-end. Standard deposition techniques are used to manufacture the metal/insulator/metal stack, making the integration of resistive devices cost-effective, and compatible with existing semiconductor fabrication processes.
Refs. [154, 155, 156] reported similar exemplary manufacturing processes of BEOL-integrated memory stacks, for Si-implanted 5 nm or 10 nm HfO_2 films combined with a Ti/TiN top electrode. The presence of a Ti layer in the top electrode is essential for reliable bipolar resistive switching of HfO_2 films. As reported in [157], for a 8 nm thick HfO_2 film, devices with thicker Ti layers have larger initial leakage currents, resulting in smaller forming voltages. It was also observed that with a Ti layer below 3 nm, no resistive switching could be measured after forming. Indeed, the Ti layer serves as an oxygen reservoir, not only extracting oxygen during electrical forming, but also facilitating resistive switching thereafter.
Moreover, larger doping concentrations (mean concentration in the range 0.5-5%) are beneficial to reduce forming forming voltage. In particular, local Si-implantation has shown to significantly decrease forming, set and reset voltages, improving data retention, while not being detrimental for endurance [155].
Thickness scaling is also beneficial for reducing the forming voltage. Indeed, 5 nm HfO_2 films, combined with 5 nm Ti scavenging layer are nowadays more common in hafnia-based RRAMs [156, 158]. Although ultra-thin layers (3 nm) of doped HfO_2 have been observed to be forming-free [159], reducing the oxide thickness can lower the oxide layer's resistivity, which in turn may increase the conductance of the LCS. As a result, there is a trade-off between the forming voltage and the memory window.
Contrarily to ferroelectric devices, the crystalline phase of resistive memory devices does not play a crucial role in the switching mechanism. Nevertheless, it can impact operating voltages and reliability characteristics [160].

2.3.3 1T-1R RRAM arrays

Resistive memory devices can be serially connected with a selector transistor (one transistor one resistor cell, 1T-1R) and organized in an array configuration for memory applications (Figure 2.5a). Such arrays are often referred to as RRAMs. Typically an n-MOSFET is used as selector device.
A 1T-1R cell can be accessed via three ports, a WL connected to the gate of the selector transistor, a SL connected to the top electrode of the resistive device and a BL connected to the source of the selector transistor. The bottom electrode of the RRAM device is directly connected to the drain of the selector transistor.
In bipolar devices, performing a set operation, i.e. writing a 1 in the resistive device, consists in pulsing the SL of the 1T-1R cell while the WL is pulsed and the BL is connected to ground (Figure 2.5b). On the other hand, the device can be reset (Write a 0) by pulsing the BL while the WL is pulsed and the SL is connected to ground (Figure 2.5c). Optimizing the programming conditions of RRAM arrays is essential for improved reliability. Enforcing the correct compliance current and SL pulse amplitude during forming and set operations in essential for optimal management of the conductive filament. Indeed, the

multi-level capability of resistive memory devices relies on the possibility to create several high conductance states by tuning the compliance current enforced during a set operation. This is achieved by controlling the selector transistor gate-to-source voltage, with an n-MOSFET with a grounded source. On the other hand, during reset operations, the compliance current is not controlled, but finding the optimal BL voltage is of the utmost importance.
The read operation of a resistive memory cell consists in reading its resistance value by typically applying a voltage pulse of sufficiently low amplitude, such that the conductive filament configuration is not altered (Figure 2.5d).

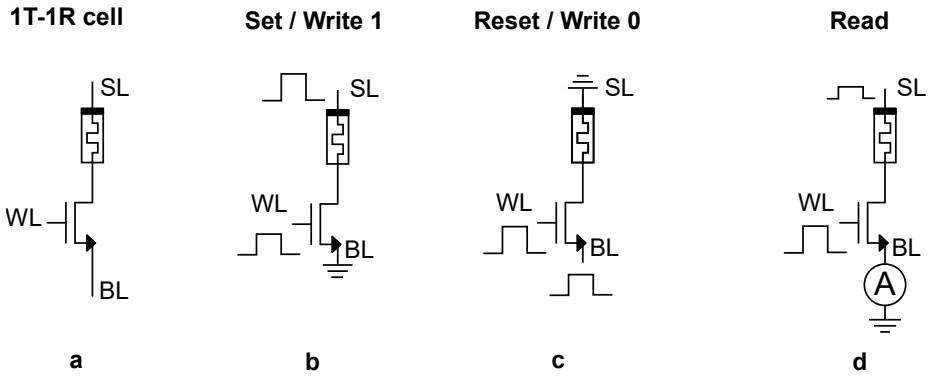

Figure 2.5: **Programming and read operations of RRAM arrays. a** RRAM bitcell consisting of a selector transistor connected to the bottom electrode of a memristive device (1T-1R structure). **b** Set operation of a 1T-1R cell. **c** Reset operation of a 1T-1R cell. **d** Read operation of a 1T-1R cell.

This general idea of evaluating the RRAM resistance without altering the device physical state is the reason that allows RRAM devices to have virtually unlimited read endurance. Clearly, embedded RRAM arrays require a sensing element translating the analog information of the RRAM resistance into a digital information. The design of the sensing element depends on the intended operation of the RRAM device – i.e. binary or multilevel operation – and array – digital memory bank, near-memory computing or analog in-memory computing engine. The design of these sensing elements is a considerably large research topic that is out of the primary scope of this thesis and it is therefore not be further explored here. In chapter 4, the design of a sensing element for the implementation of a near-memory computing inference engine is presented.

2.4 Unified ferroelectric and resistive memory technology

The previous analysis of HfO_2 based ferroelectric and resistive memories shows that even though the underlying switching mechanism of the two memory technologies relies on different – and sometimes contrasting – physical properties of the material, the active memory substrate is very much similar for both technologies. The fabrication processes of ferroelectric and resistive memories also share many similarities, hinting at the possibility to find common ground between the two in order to produce a single memory stack enabling both switching phenomena to co-exist in the same memory stack.
Combining ferroelectric and resistive properties within the same memory stack can drastically simplify the integration of these two memory types, which, when combined, offer

an optimal solution for implementing deep learning hardware with learning capabilities, as further discussed in the next chapters. In this section, the existing literature on hybrid ferroelectric and resistive memory technologies is reviewed. Following this, the unified ferroelectric/resistive memory technology proposed in this thesis is introduced, along with its fabrication process and single-device characterization in both operating modes. Finally, the electrical characterization of ferroelectric and resistive memory arrays based on this unified stack is presented and discussed.

2.4.1 Existing research on hybrid ferroelectric/resistive memory technology

Interplay between ferroelectric and resistive switching has already been observed in hafnia-based devices. Ref. [161] explored the underlying principles of combining ferroelectric and resistive switching within a single capacitor cell. The analysed structure is a polycrystalline strontium-doped hafnium oxide sample, with TiN as the bottom electrode and Pt as the top electrode. To define the capacitors, platinum top electrodes with a diameter of 200 µm were evaporated, i.e. capacitor area of of 31 416 µm^2. It was observed that crystalline structures – essential requirement for the operation of the capacitor as a ferroelectric memory – require a higher forming voltage but exhibit a lower reset voltage compared to amorphous structures, which are a more standard choice for resistive memory devices. The switching performance revealed stable bipolar operation with off-to-on ratios of 10 and 50 for the amorphous and crystalline samples, respectively. The ferroelectric behaviour of the sample was stable with a remanent polarization of up to 12 µC/cm^2. Successful resistive switching was also observed, with filament formation in the capacitor stack demonstrating a double reset and set operation with a current off-to-on ratio spanning four orders of magnitude. Additionally, the device's high resistance state, achieved through a complete reset of the oxide layer, allowed for further ferroelectric switching. This second ferroelectric switching operation exhibited a remanent polarization of 5 µC/cm^2. The study further investigated how ferroelectric field cycling impacts resistive switching performance, given that both memory mechanisms depend on the distribution and concentration of oxygen vacancies in the oxide layer. The study showed that different ferroelectric states – pristine, woken-up, and fatigued – affect subsequent resistive switching behaviour. Specifically, increased field cycling, which induces additional oxygen defects in the hafnium oxide layer, lowers the forming voltage and makes filament formation easier. However, the set and reset voltages were not dependent on the distribution of oxygen vacancies.
Ref. [162] focused on bipolar resistive switching in ferroelectric epitaxial HZO. The investigation on was performed on $Hf_{0.5}Zr_{0.5}O_2$ epitaxially grown on $La_{0.8}Sr_{0.2}MnO_3$-buffered $SrTiO_3$. In this crystalline system, filamentary switching was observed without the need for electroforming and current compliance, alongside symmetric ferroelectric switching with distinctive hysteresis loops. Alternating between these switching mechanisms is reversible and does not affect the subsequent ferroelectric performance. The study explicitly demonstrates the transition from resistive switching back to ferroelectric switching, enabled by the common reset serving as a deep reset to the virgin state. The parallel and non-interfering nature of these phenomena suggests that the mechanisms operate independently on different time and voltage scales within the same device. This study is also limited to single device structures with relatively large size. Indeed, the smallest investi-

gated capacitor structure has an area of 49 µm^2.
Ref. [163] explored for the first time the ferroelectric and resistive operations of memory stacks based on TiN/Si:HfO_2/Ti/TiN integrated in the BEOL of 130 nm CMOS technology node, thus paving the way for the integration of such memory devices in denser array implementations. It was found that the ferroelectric properties of these structures are modulated by the presence of a Ti oxygen scavenging layer. Stacks based on TiN/Si:HfO_2/Ti/TiN were integrated in 130 nm CMOS BEOL between the fourth and fifth metal layers, to create 0.28 µm^2 memory elements. Bulk chemical analysis revealed that the amount of oxygen vacancies, already present in Si implanted hafnia films without Ti, significantly increased with Ti thickness. Moreover, the structural analysis of the film proved that the increased oxygen vacancies concentration induced by the presence of Ti increased the orthorombic (o) phase proportion at the expense of the tetragonal (t) one for 10 nm Si:HfO_2 films and also induced partial crystallization in the o/t phase for 5 nm Si:HfO_2 films. These structural observations were validated by electrical characterization of FeCAPs structures. The presence of Ti increased the remanent polarization memory window to 40 µC/cm^2, double with respect to the same stack without Ti, and reduced wake-up. Moreover, weak ferroelectricity was induced in thinner 5 nm films, with a remanent polarization memory window smaller than 2 µC/cm^2. These results obtained at device level were finally confirmed at array level. Indeed, the increased remanent polarization with Ti-based stack resulted in up to three times memory window with respect to Ti-free reference arrays. As a result of higher o-phase proportion induced by engineering of oxygen vacancies, Ti-based FeRAM arrays could be operated at lower voltages while maintaining a sufficiently large memory window.
This analysis revealed that the presence of Ti in the top electrode can have some positive effects on the ferroelectric properties of these devices. Some preliminary results on the resistive operation of such devices were also reported. Indeed, the increase of oxygen vacancies through Ti interface engineering was found to facilitate conduction mechanisms, leading to the creation of a oxygen vacancies-rich filament in the ferroelectric layer, resulting in a resistive memory operation. Nevertheless, resistive operation was proven only for 5 nm films, which did not show reliable ferroelectric operation.

2.4.2 A novel ferroelectric/resistive memory stack

In this thesis, a single memory stack based on silicon-doped hafnium oxide is introduced, enabling the fabrication of resistive memories and ferroelectric capacitors in the same BEOL process. This integration, requires no additional masks, simplifying manufacturing by combining both technologies into a unified memory stack. Initially fabricated as FeCAP, these devices can be transformed into resistive memory elements through a unique electrical forming operation, allowing FeCAP and resistive memories to coexist on the same chip (Figure 2.6).
The memory stack combines a 10 nm ferroelectric silicon-doped HfO_2 film with a titanium scavenging layer. Figure 2.6 illustrates the crystallization of the HfO_2 film, along with the top and bottom electrodes. The inclusion of the Ti layer serves a dual purpose: it enhances the ferroelectric properties of the structure, leading to a higher remanent polarization in the ferroelectric capacitor [164], and it increases the concentration of oxygen vacancies at the interface. This intentional increase in oxygen vacancies promotes the formation of conductive filaments within the ferroelectric layer, facilitating the operation

as a resistive memory, as shown in [163].
For the first time a unified memory stack based on Si-doped hafnia is characterized at array level, showing state of the art performance for both ferroelectric and resistive operations. For this purpose, FeRAM arrays and RRAM arrays were fabricated and tested on the same wafer using the developed metal/ferroelectric/metal memory stack.

Fabrication process. Memory devices based on the unified ferroelectric/resistive stack were fabricated in the BEOL of a 130 nm CMOS foundry process with four metal layers. The unified memory, based on ferroelectric Si-doped hafnium oxide was fabricated on tungsten vias on top of metal layer four. The unified memory stack consists of a 150 nm thick TiN bottom electrode, a 10 nm thick Si-doped HfO_2 ferroelectric layer, a 4 nm thick Ti oxygen scavenging layer, and a 100 nm thick TiN top electrode. TiN electrodes were deposited by PVD at 350°C. The HfO_2 layer was deposited by ALD at 300°C with $HfCl_4$ and H_2O as precursors. The oxide film was then doped with silicon by ion implantation to achieve a 1% Si mean concentration. The titanium scavenging layer was deposited by PVD at the top electrode interface, with no air break between the Ti and TiN top electrode. The resulting stack was etched to define square capacitors with 600 nm sides. Capacitors were then encapsulated by SiN and SiO_2 deposited at 400°C, followed by a planarization step before via opening and metal five deposition at 450°C.

Single device characterization. Figure 2.6 shows five polarization-electric field hysteresis loops measured using the PUND technique at ±3 V-10 kHz on 550 nm diameter capacitors after a wake-up phase of 10^4 triangular pulses at ±3 V-10 kHz. The polarization - electric field plot is obtained by averaging the measured displacement current of 1,000 capacitors.

Figure 2.6 also shows five quasi-static current-voltage curves for the set and reset operations of a 1T-1R device, with minimum length and 660 nm wide access transistor and 600 nm side square capacitors. For the set operation, the WL of the access transistor was set to 1 V, the BL was grounded, and the top electrode was swept between 0 V and 2.6 V in 50 mV steps. For the reset operation, the WL of the access transistor was set to 4.5 V, the top electrode was grounded, and the BL was swept between 0 V and 1.45 V in 50 mV steps. Set operations were always preceded by reset ones. Prior to cycling, the 1T-1R device underwent forming, during which the WL of the access transistor was set to 1.05 V, the bottom electrode was grounded, and the top electrode was swept between 0 V and 5.2 V in 50 mV steps.

2.4.3 FeRAM operation with the unified memory stack

The 10 nm ferroelectric Si:HfO_2 film with Ti scavenging layer was integrated in the 130 nm CMOS BEOL to produce 0.36 µm^2 square capacitors in 16,384 1T-1C FeRAM arrays with sense amplifiers operating from 4.8 V down to 2.5 V, as in [151]. FeRAM arrays, used to evaluate the memory performance of the proposed unified memory stack as a binary ferroelectric device, include shift registers to address different lines, control circuits for the shift registers, line drivers to connect the addressed lines to an external voltage, and

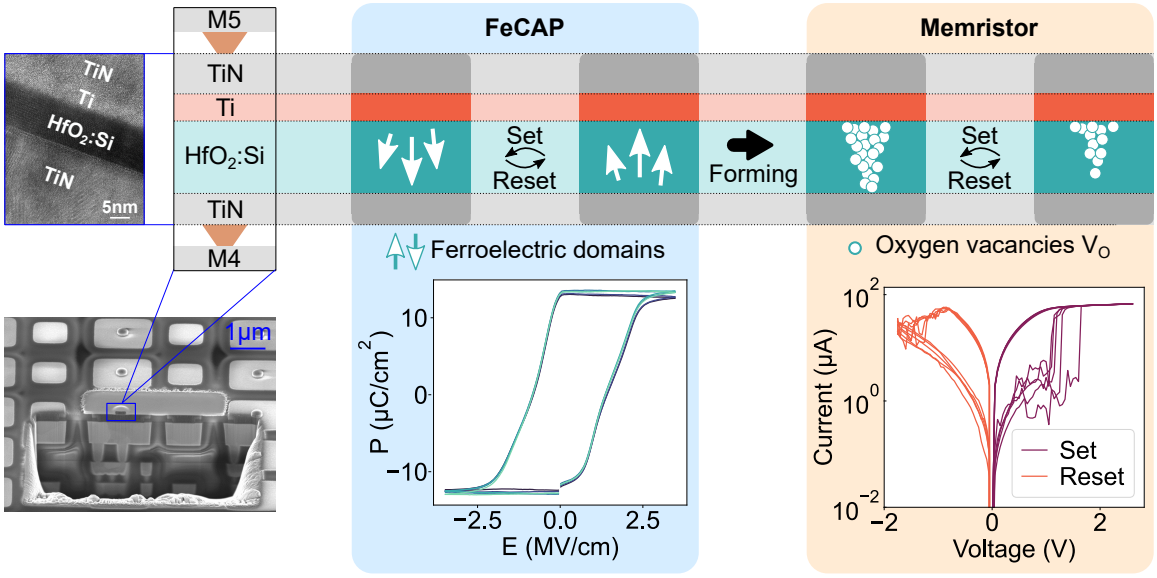

Figure 2.6: **Unified memory stack enabling the fabrication of memristors and ferroelectric capacitors.** The same BEOL integrated metal-ferroelectric-metal stack (High-resolution transmission electron microscopy cross section illustrating 10 nm Si:HfO_2 film crystallization and tilted scanning electron microscope, SEM, view with focused ion beam cross section) can be used as either ferroelectric capacitor (showing Polarization-Electric Field hysteresis loops) or memristor (exhibiting butterfly-shaped Current-Voltage curve) after undergoing a forming operation.

sense amplifiers for data readout. A clocked latch sense amplifier is used to compare the bit line voltage to an external reference voltage during the read operation (Figure 2.3). The digital peripheral circuitry is supplied with 4.8 V. A 660 nm wide minimum length selector transistor is serially connected to the designed capacitor.

Measuring analog distributions of FeRAM arrays. After undergoing wake-up cycling, i.e. alternating 1000 reset-set pulses, the ferroelectric capacitors were characterized. The distributions of 0 and 1 states (Figure 2.7a) were obtained by alternating programming and read pulses for increasing values of reference voltage V_{ref}, i.e. the voltage to which the BL potential is compared during a reading operation. The median memory window, defined as the separation between the median V_{ref} associated with the zero and one states, is 360 mV. The worst-case memory window, the difference between the V_{ref} values associated with the least separated zero and one states, is 120 mV. The reference voltage was swept between 0 V and 1 V with a 20 mV step. Programming and reading pulses of 3 V amplitude and 2 µs duration were employed for this characterization. This analysis highlights the reliability of the proposed stack as a ferroelectric memory. No bit error was measured with the described conditions.

Array-level switching efficiency. After performing wake-up cycling with 10^3 pulses, the switching efficiency on the 16 kbit array across various pulse amplitudes and durations was measured. The fixed reading conditions were set to V_{ref}=0.56 V and 3 V-2 µs pulses to identify the optimal programming conditions for the FeCAPs. The value V_{ref}=0.56 V was

chosen as the midpoint in the median memory window in Figure 2.7a. Figures 2.7b and 2.7c illustrate the trade-off between pulse width and amplitude for programming a zero (source line pulse) or a one (bit line pulse) in cells initially programmed to the opposite state. The two highlighted programming conditions – A and B – both achieved 100% switching efficiency. All these measurements relied on on-chip embedded delay blocks, designed to generate pulse timings down to 520 ps.
It is worth noticing that this analysis provides information on the intrinsic switching efficiency of the ferroelectric capacitors, influenced by the array design. Indeed, the same analysis evaluated on a different design, might provide slightly different results, in particular for the smallest pulse widths or lowest pulse amplitudes, i.e. conditions for which the parasitic contributions introduced by design choices might impact the results the most.

Programming energy estimation. The programming energy to switch one bit (Figure 2.7d) was evaluated for several programming conditions with 100% switching efficiency at array level. The programming energy of a ferroelectric memory cell can be evaluated as

$$\Delta E = 2P_R \cdot S \cdot V_{prog} = MW_{0\sigma} \cdot (C_D + C_{BL}) \cdot V_{prog}, \tag{2.8}$$

where P_R is the remanent polarization, S is the capacitor area, $MW_{0\sigma}$ is the median memory window, C_D is the dielectric capacitance, C_{BL} is the BL parasitic capacitance, and V_{prog} is the programming voltage. The relationship between remanent polarization memory window ($2P_R$) and array-level memory window ($MW_{0\sigma}$) can be derived from equation 2.3. Indeed, for a ferroelectric capacitor programmed either in the one or zero states,

$$V_{BL,1} = \frac{C_D}{C_D + C_{BL}} V_{SL} + \frac{2P_R S}{C_D + C_{BL}},$$
$$V_{BL,0} = \frac{C_D}{C_D + C_{BL}} V_{SL}.$$

The memory window of a capacitor is defined as:

$$MW = V_{BL,1} - V_{BL,0} = \frac{2P_R S}{C_D + C_{BL}}, \quad \text{i.e.} \quad 2P_R S = MW(C_D + C_{BL}) \tag{2.9}$$

For an array of capacitors, equation 2.9 can be rewritten as:

$$2P_R S = MW_{0\sigma}(C_D + C_{BL}).$$

The area enclosed by a polarization - electric field hysteresis loop (Figure 2.1a) represents the energy per unit volume dissipated during one complete cycle of polarization switching. This is the energy required to switch the polarization state twice. Therefore the energy to switch the polarization of a ferroelectric capacitor from the 0 to 1 state (or vice-versa) is equal to the product between the volume of the ferroelectric capacitor (Sd) and half the area of the hysteresis loop (approximately $2P_R V_{prog}/d$). Thus,

$$\Delta E \approx \frac{2P_R V_{prog}}{d} Sd = 2P_R S V_{prog} = MW_{0\sigma}(C_D + C_{BL}) V_{prog}.$$

Following equation 2.8, the switching energy is obtained from the experimentally measured median memory window for the different programming conditions and the estimated capacitance values. For the median memory window extraction, the distributions of the

zero and one states, with analogous writing and reading pulses for each pulse amplitude-duration combination, were measured. The dielectric capacitance for the memory element was estimated as $C_D = \epsilon_0 \epsilon_r S/d = 9.5\,fF$, with $\epsilon_r = 29.7$, and the BL parasitic capacitance as $C_{BL} = 188\,fF$ [165]. A switching energy below 200 fJ/bit is reached for programming condition B (2.75 V - 62.5 ns), approximately 30% smaller than the switching energy of condition A (3 V - 2 µs). The programming energy of ferroelectric capacitors can be further reduced by either decreasing the programming voltage to 2.5 V or the pulse duration to approximately 20 ns, without impacting the programming reliability.

Endurance analysis. The median memory window evolution after several cycling phases was measured (Figure 2.7e), confirming the expected wake-up, stable and fatigue phases of Si-doped HfO_2-based ferroelectric capacitors. A positive $MW_{0\sigma}$ is ensured for both programming conditions after 10^7 cycles, with programming condition A exhibiting enhanced memory window. Nevertheless, the distribution of 0 and 1 states after 10^7 cycles show bit failures occurring for programming condition A. whereas almost no bit failure is observed for programming condition B, for V_{ref}=0.4 V (Figure 2.7f). Indeed, it can be observed that 0 and 1 state distributions for programming condition A overlap for larger values of V_{ref}, indicating that approximately 25% of devices in the array are either in soft or hard breakdown.

2.4.4 RRAM operation with the unified memory stack

A RRAM array with 16,384 1T-1R devices was fabricated, alongside FeRAM array. 0.36 µm^2 square resistors were integrated in the BEOL of RRAM arrays, exploiting the same memory stack used for FeRAM devices. RRAM arrays, used to evaluate the performance of the proposed unified memory stack as an analog memory, include line decoders and drivers to connect the addressed lines to an external voltage. The digital peripheral circuitry is supplied with 4.8 V. A 3.5 µm wide minimum length selector transistor is serially connected to the designed resistor.

Forming operation. Prior to array operation, a forming step was applied for each device in the array. A device was considered to be formed if its resistance was below a target resistance. The most common forming strategy consists in applying an adaptive pulse staircase to the top electrode (SL) of the devices in the array, while the BL of the device is grounded and the WL is set to a voltage level limiting the compliance current flowing in the device [166]. The adaptive pulse staircase consists in alternating programming and read pulses on all devices in the array that are above the target resistance. The limitation on the compliance current flowing in the device is necessary in order to avoid a hard breakdown of the device and control the filament formation. Optimizing the current density during forming is essential in order to optimize other properties of the resistive memory element, such as window margin, endurance and retention. Indeed, the forming current controls the radius of the conductive filament. A smaller forming current leads to a smaller conductive filament radius, which results in an increase in the amplitude of the random telegraph noise signal in the reset state. This translates to a broadening of the reset distributions, which eventually reduces the resistance window [167]. A higher

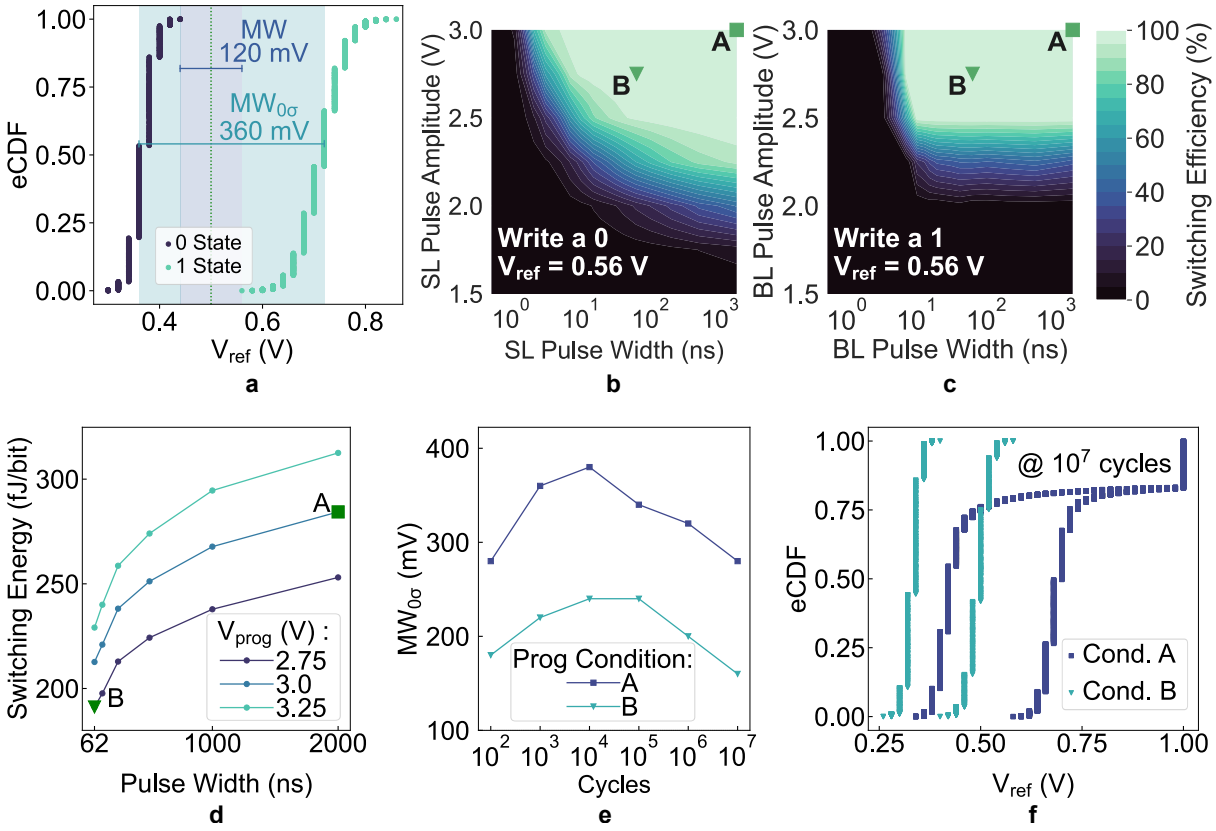

Figure 2.7: **Electrical characterization of the unified memory stack as ferroelectric memory.** **a** Raw distributions of 0 and 1 states in a 16 kbit 1T-1C FeRAM array, measured after wake-up cycling (3 V-2 µs read and programming pulses). **b, c** Array-level switching efficiency measured for programming 0 (a) and 1 (b) states using different pulse widths and amplitudes. A fixed reference voltage (V_{ref} = 0.56 V) was used for reading (3 V-2 µs read pulses). Two programming conditions (A and B) are highlighted in green. **d** Polarization-switching energy as a function of pulse duration and amplitude **e** Measured Memory Window at 0σ as a function of cycling. The same conditions were used for reading. **f** Raw distributions of 0 and 1 states in 16 kbit 1T-1C FeRAM arrays, measured after 10^7 endurance cycles, for programming conditions A and B (3 V-2 µs read pulses).

forming current, would therefore be beneficial in order to improve the window margin, but it would have a negative impact on both retention [168] and endurance [169].
This forming procedure was applied to the manufactured RRAM array, sweeping the programming voltage (V_{SL}) between 3 V and 6 V with 200 mV step. The WL was pulsed with 1.05 V-100 µs pulses. This sweep was repeated for 5 times. A read operation (400 mV-70 µs pulses) was performed on the whole array after a programming pulse was applied to each device to check if the target resistance, which was set to 50 kΩ, was reached. Figure 2.8a shows the forming statistics of a 16,384 1T-1R array. During the first sweep, approximately 58% of the devices were formed with pulse amplitudes between 4 V and 6 V. Repeating the adaptive staircase forming slightly decreased the amount of devices left to form, saturating around 38% of devices not yet formed.
A second strategy used for forming RRAM arrays consists in applying an adaptive pulse train [166]. It consists in alternating read and programming pulses on devices which have not yet reached the target resistance. This time, the amplitude of the programming pulses is fixed to the maximum amplitude used in the case of the adaptive staircase strategy. This second strategy was applied to a pristine RRAM array, applying programming pulses of amplitude V_{SL}=6 V. Pulses of 1.05 V amplitude were applied to the WL. The programming pulse duration was 300 µs. A read operation (400 mV-70 µs pulses) was performed on the whole array after a programming pulse was applied to each device to check if the target resistance was attained. Figure 2.8b shows the forming statistics of a 16,384 1T-1R array. After 20 programming pulses approximately 87% of the devices in the RRAM array reached the target resistance of 50 kΩ. By increasing the number of pulses to 1,000, only 5% of the devices (less than 900 devices) were not yet formed.
A combination of the two strategies can also be used in order to form the devices. First, an adaptive staircase forming could be applied in order to stress as little as possible the devices that require a smaller forming voltage. Then, an adaptive pulse train strategy could be used to form the devices requiring additional operations.
Devices that are not yet formed, after the limit on number of programming pulses is reached, can often reach the target resistance after few set-reset cycles.
It should be noted that the maximum programming voltage of 6 V used during the forming operation goes beyond the nominal operating voltage for the 130 nm CMOS technology node which is fixed to 4.8 V. Nevertheless, only 50% of the devices in a 16,384 1T-1R array are formed at V_{SL}=4.8 V after an adaptive staircase cycle (Figure 2.8a). Therefore, optimization strategies at device level should be designed in order to reduce the forming voltage, thus improving the compatibility of the proposed technology with more advanced CMOS technology nodes. Some improvement possibilities are discussed later in this chapter.

Programming conditions optimization. The programming conditions were optimized to improve the performance of resistive devices in the array. Following the forming step, cycling tests were conducted under various programming conditions to minimize the bit error rate (BER) and enhance the memory window. Figures 2.9a and 2.9b illustrate the BER and the median memory window, after 100 reset-set cycles, as functions of the word line voltage applied during the reset and set operations, respectively. The BER is defined as the value of the experimental cumulative distribution function (eCDF) at the conductance value for which the set and reset distributions cross each other. On the other hand, the median memory window is defined as the difference between the high

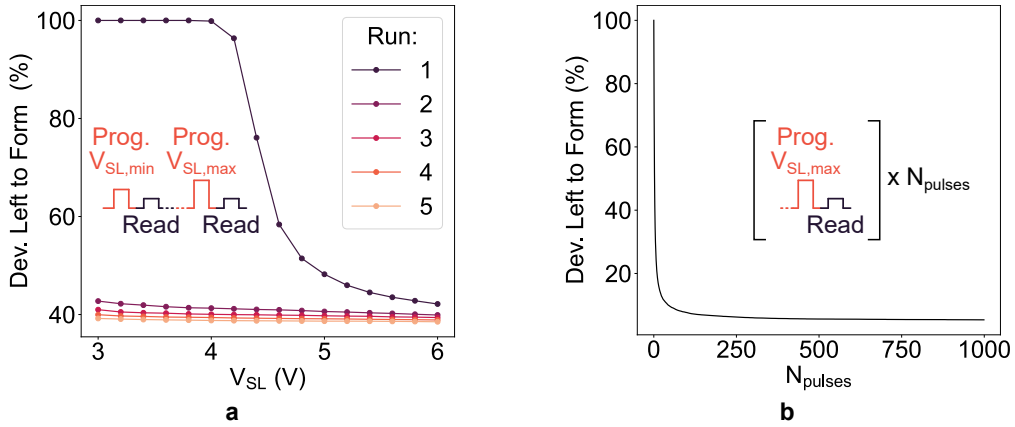

Figure 2.8: **Forming operation of the unified memory stack. a** Adaptive staircase forming statistics repeated five times on a 16,384 1T-1R array. **b** Adaptive pulse train statistics repeated five times on a 16,384 1T-1R array.

and low conductance states distributions at the median value of the eCDFs for the set and reset states respectively. Even though the two quantities are correlated with each other – typically a low BER corresponds to a larger memory window – they can provide complementary information. The median memory window provides information on how far apart the high and low conductance states are globally, whereas the BER specifies how many outliers are present in the two distributions.
The optimal reset condition was achieved at $V_{BL} = 3.0\,V$ and $V_{WL} = 3.1\,V$, while the optimal set condition for binary operation was obtained at $V_{SL} = 2.3\,V$ and $V_{WL} = 1.0\,V$. For analog programming, V_{WL} could be modulated within the range of $0.85\,V$ to $1.0\,V$. $1\,\mu s$ long pulses were employed for both set and reset operations. During sweeping of the WL reset voltage, optimal set operations were employed. Vice-versa during sweeping of the WL set voltage, optimal reset operation were employed.
For the sake of clarity, Figures 2.9a and 2.9b show a simplified analysis of the procedure employed for the optimization of the programming conditions. Indeed, a design of experiment (DOE) routine was used to evaluate the performance of the memory technology under different programming conditions. The DOE routine loads and executes a condition table file and applies each condition on a unique subset of devices randomly chosen. The programming conditions table specifies for each condition:

- rise-time, width and fall-time for the reset operation,
- BL and WL voltage for the reset operation,
- rise-time, width and fall-time for the set operation,
- SL and WL voltage for the set operation.

The rise-time, width and fall-time of the pulses were fixed to 20 ns, $1\,\mu s$ and 20 ns respectively. On the other hand, several BL and WL voltages for reset and SL and WL voltages were cross-checked in order to find the optimal conditions. A subset of approximately 100 devices was used to test each programming condition, in order to obtain statistical information.

Low and high conductance states distributions and endurance analysis. The distributions of the low conductance state and high conductance state after several cycling phases are shown in Figure 2.9c. The optimal programming conditions described in the previous paragraph were used for this characterization. The measured 16,384 1T-1R array underwent 10,000 reset-set cycles and a read operation of the whole array was performed at each cycling decade.
The median conductance values of the LCS range from 7.3 µS to 2.2 µS, i.e. it decreases with cycling, whereas the median conductance of the HCS is quite stable (51 µS to 54 µS). A possible explanation for the shift of the LCS state towards smaller conductance values with cycling is the synergy between filamentary conduction and modulation of the Schottky barrier height by the polarization state and oxygen vacancy concentration [163]. Therefore, up to 10^5 cycles, the median memory window of the resistive memory arrays increases, as well as the BER, which reaches a value of 1% after 10^5 reset-set cycles. The smallest BER is measured after 10^4 cycles and it is equal to $7{\cdot}10^{-3}$.
Several considerations can be be extrapolated from this analysis:

- The median conductance value of the low conductance state is smaller than typical values measured on similar stacks [170].
- The median conductance value of the high conductance state is also smaller than typical values measured on similar stacks, which can go above 100 µS, as show in Refs. [170, 171]. The reason for this is that smaller programming currents are employed for set operations for optimal BER and median memory window. Indeed, the compliance current during set operations is kept below the forming current. As mentioned above larger forming currents, and therefore larger set currents, could be used for improved window margin. Nevertheless, for the proposed memory stack, larger compliance currents during forming and set operation were found to be detrimental for the global memory performance.
- It can be argued that improvements of the BER for the proposed memory stack are therefore linked to the optimization of the forming strategy and device engineering.

Conductance stability over time. RRAM devices experience conductance instability issues primarily due to the localized random diffusion of oxygen vacancies into and out of the conductive filament. This phenomenon, also known as relaxation, occurs at smaller time scales, typically in the first seconds after programming. It differs from long-term retention characteristics, i.e. the reliability of the programmed states according to the typical industry standard of 10 years at 85°C [172].
The conductance of 8,192 devices programmed to LCS and HCS was monitored over time, up to almost one day ($7.2{\cdot}10^4$ s) at room temperature (Figure 2.9d). The 10-year retention at room temperature was estimated by employing a linear regression model, using the last five measured conductance values plotted against the logarithm of the cumulative time. The extrapolated average LCS after 10 years is 3.8 µS, whereas the average HCS is 18.9 µS.
Better retention characteristics have been typically observed for similar resistive memory stacks. For the hybrid ferroelectric/resistive stack, the measured high conductance state gradually decreases over time. This could be due to the relatively low value of the programmed high conductance state. Indeed, the lower the conductance is, the more

the conductive filaments is unstable, resulting in larger conductance instability over time [158].

Conductance tuning and multi-level cell operation. Resistive memory devices can be used not only as binary devices, but also as multi-level cells. Indeed, the cell conductance can be tuned by controlling the programming current flowing in the device, thanks to the voltage applied to the gate of the selector transistor, by leaving the top electrode voltage unchanged. Figure 2.9e shows the average conductance read after a single programming pulse as function of the imposed programming current. The WL voltage was swept between 0.85 V up to 1.01 V with 20 mV step. The SL pulse amplitude was 2.3 V. Each set operation was preceded by a reset one. A subset of 1,489 devices was tested for each programming condition to obtain statistical information. The inset in the Figure 2.9e shows the conductance standard deviation as a function of the average conductance value. Information on the dispersion of the conductance distributions for various conductance values allows to define the best allocation strategy for multi-level RRAM cells.
The most common procedure to program a resistive memory device to a specific conductance state consists of a an iterative program-verify scheme. A device, initially in a reset state, is set with a compliance current defined by the average conductance value of the bin to which the conductance of the device should belong. After programming, the conductance of the device is read and its value is compared to the limits of the bin for the defined conductance level. If the programmed conductance is within the bin limits, the device is considered to be programmed correctly. Otherwise, the device is reset and the same operations are repeated until a successful programming occurs or the limit on the number of iterations is reached.
In order to implement this programming algorithm, the conductance bins and gaps between the different levels have to be defined. Linear allocation – using equal widths for all conductance bins and equal widths for all gaps for the various levels – or sigma-based-allocation – using the standard deviation of conductance values (referred to as sigma) to allocate the conductance ranges – strategies have been proposed [171, 173]. The inset in Figure 2.9e shows no significant variations of the conductance standard deviation in the programmed conductance range. For this reason, a linear based allocation scheme with equally spaced bins was used for the multi-level cell allocation of four and eight levels, respectively shown in Figures 2.9f and 2.9g. A program-verify scheme was used, with a limit of 200 iterations. The programming conditions (conductance bins and gaps and relative measured programming currents) are summarised in Table 2.1.
A program-verify scheme taking into account RRAM relaxation after programming has been recently proposed [158]. The iterative procedure includes a wait time between the conductance programming and reading operation – typically of a few seconds – in order to take into account conductance instability issues. Thanks to this scheme, improved conductance stability has been proven at the cost of larger global programming time. This scheme has also been applied to resistive memory arrays with hybrid ferroelectric/resistive memory stacks but no significant improvement was observed.

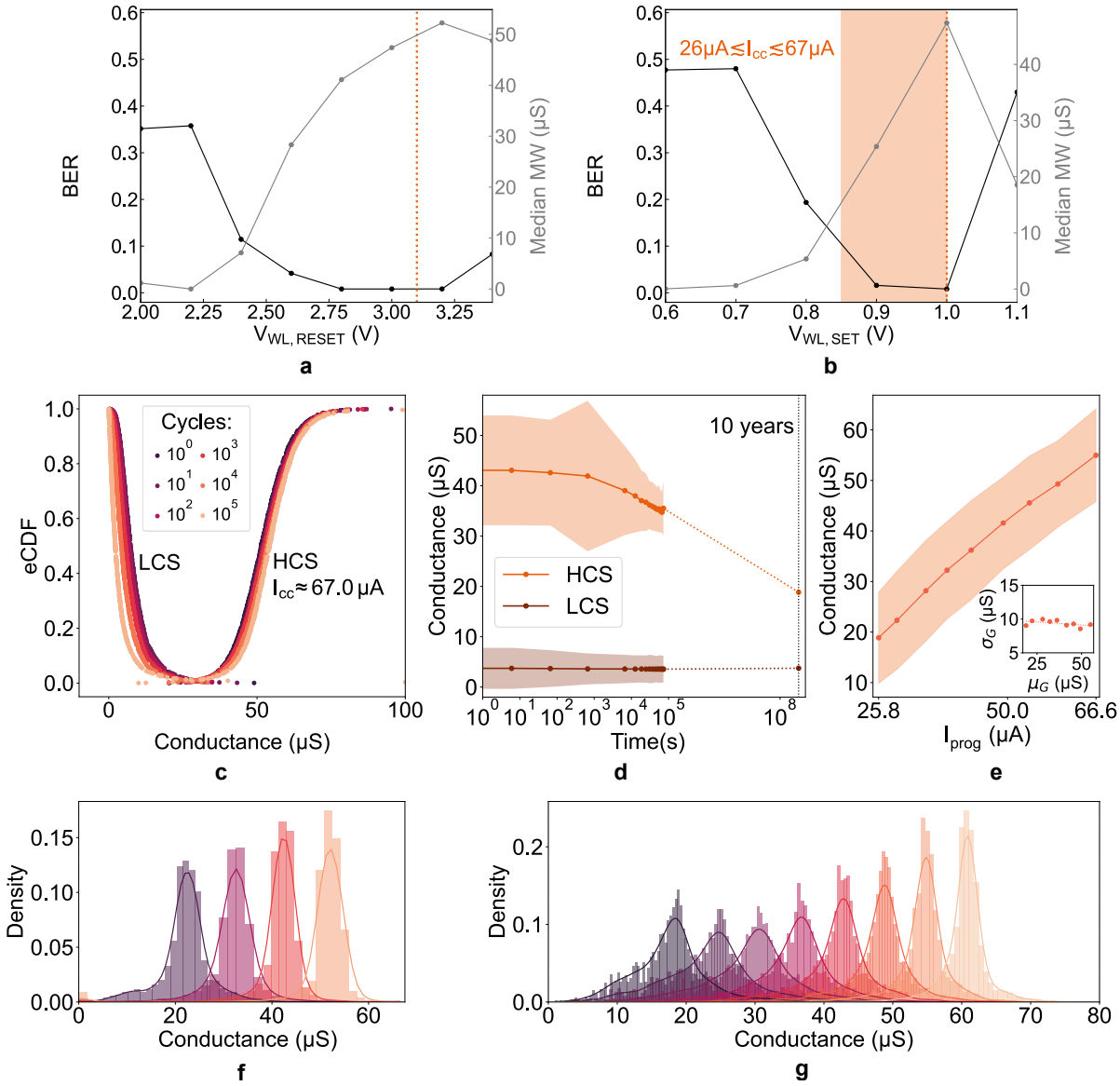

Figure 2.9: **Electrical characterization of the unified memory stack as resistive memory. a, b** BER and median memory window (MW) of a 1T-1R ferroelectric RRAM array for different programming conditions. Orange lines represent the conditions for binary programming, while the orange area represents the conditions for multi-level cell programming. **c** Measured HCS and LCS distributions of 16,384-device RRAM array after several set-reset cycling phases. **d** Measured mean LCS and HCS evolution of RRAM array over time and extrapolated 10 years retention. Shaded area corresponds to standard deviation at 1σ. **e** Average RRAM conductance and standard deviation at 1σ for increasing programming currents. **f, g** Experimental conductance distributions for programming 4 and 8 levels per cell.

Level		I_{prog} (µA)		G_{min} (µS)		G_{max} (µS)	
1	1	30.3	24.4	20	18	25	20
	2		30.8		24		26
2	3	39.5	37.3	30	30	35	32
	4		43.8		36		38
3	5	48.7	50.3	40	42	45	44
	6		56.8		48		50
4	7	57.9	63.2	50	54	55	56
	8		69.7		60		62

Table 2.1: **Programming conditions for multi-level programming**. Imposed target conductance ranges for allocating 4 and 8 levels per cell and average programming currents used for programming each conductance level.

Interplay between ferroelectric and resistive switching. Ferroelectric and resistive switching mechanisms operate at different voltage scales in the developed unified memory stack. Indeed, ferroelectric switching occurs for voltage pulses below the forming voltage. The amplitude of the applied pulses has to be sufficiently high to ensure polarization switching, but low enough to decrease the creation and migration of oxygen vacancies inside the bulk and at the electrodes interfaces. On the other hand, resistive switching cannot occur if forming operation is not performed, i.e. it requires a conductive filament of oxygen vacancies.
Once forming is performed, the resistance of the metal-ferroelectric-metal structure decreases. This resistance decrease masks the displacement current coming from ferroelectric switching, which cannot be measured anymore. Refs. [161, 162] proved the possibility to retrieve ferroelectric switching after resistive operation by either employing a deep reset – i.e. a dissolution of the conductive filament – or forming-free stacks which could return to a virgin state when resetting the device, respectively.
The impossibility to measure ferroelectric switching once forming is performed does not mean that polarization reversal cannot occur in the bulk regions of the device which are not interested by the conductive filament migration and that are still crystallized in the orthorombic phase. Indeed, during set and reset operation of the conductive filament, the polarization state of the ferroelectric domains might be perturbed if the electric field imposed by the resistive programming pulses is above the ferroelectric coercitive field. It is possible that the polarization reversal during resistive set-reset operation might have an impact on the creation and dissolution of the conductive filament. Indeed, during optimization of the programming conditions, some peculiar switching phenomena were observed for resistive operation of the unified memory stack.
Figure 2.10a shows ten different cycles of quasi-static current-voltage (applied to the top electrode) curves measured on a 1T-1R device, with minimum length and 660 nm wide access transistor and 600 nm side square capacitors. For the set operation, the WL of the access transistor was set to 1 V, the BL was grounded, and the top electrode was swept between 0 V and 2.6 V in 50 mV steps. For the reset operation, the WL of the access transistor was set to 3.2 V, the top electrode was grounded, and the BL was swept between 0 V and 3.25 V in 50 mV steps. Set operations were always preceded by reset ones. Prior to cycling, the 1T-1R device underwent forming, during which the WL of the access transistor was set to 1.05 V, the bottom electrode was grounded, and the top electrode was swept between 0 V and 5.2 V in 50 mV steps. Cycles 1 to 3 show the typical

reset-set operation of a bipolar resistive memory. In cycle 4, the reset operation effectively increases the device resistance, but so does the set operation. During cycles 5 to 7, the reset operation decreases the device resistance. Indeed the abrupt increase in the current flowing into the device, by approximately one order of magnitude, can be clearly observed. During cycle 8, both set and reset operations decrease the resistance of the device abruptly. Finally, cycles 9 and 10 have a similar trend to cycle 4.
Similar behaviours were observed at array level with pulsed programming. For instance, Figure 2.10b shows the set and reset distributions for increasing value of WL voltage during set after 100 cycles. All other conditions were fixed to the optimal ones defined in Figure 2.9a,b. For a WL voltage equal to 1 V the typical high and low conductance states distributions of a bipolar RRAM can be observed. Starting from WL voltages larger than 1 V, an increasing portion of devices programmed with positive pulses applied on the top electrode shifts towards a low conductance state; conversely, when a reset pulse is applied – a positive pulse is applied to the BLs of the array – the devices' conductance increases. In this setting, set and reset distributions are reversed. The interesting property about this operating mode is that a large median memory window can be observed, i.e. a factor greater than 100 between the high and low conductance states. Such a large memory window was not obtained for the typical bipolar resistive operation.
Further investigation is necessary to understand the physical origin of these phenomena.

2.5 Perspectives on memory technology optimization

The electrical characterization of the memory arrays based on the unified ferroelectric/resistive devices presented just above showcases the great potential of this memory technology for the dual-mode operation of a single memory stack.
Nevertheless, further optimization is necessary for increasing the reliability of these devices either as ferroelectric capacitors or resistive memory devices.
To do so, two possibilities are conceivable: either finding a good trade-off between ferroelectric and resistive memory response in a unified ferroelectric memory stack, or optimize the two memory technologies separately and combine them together in the same back-end of line with minimal added manufacturing cost. In the following, the two solutions are discussed.

2.5.1 Optimization of the unified memory stack

The thickness and oxygen vacancies profile of the Si-doped HfO_2 layer are crucial for both ferroelectric and resistive memory technologies and must be optimized through adjustments in deposition parameters and interface design between the active material and the memory electrodes [174].
An optimized concentration of oxygen vacancies can enhance ferroelectricity by stabilizing the orthorhombic phase but must be carefully managed to avoid promoting non-ferroelectric phases and increasing leakage currents. On the other hand, the presence of oxygen vacancies in a resistive switching device greatly influences the formation and dissolution of conductive filaments, which directly impact the switching between high conductance state and low conductance state. Finding the sweet-spot in terms of layer

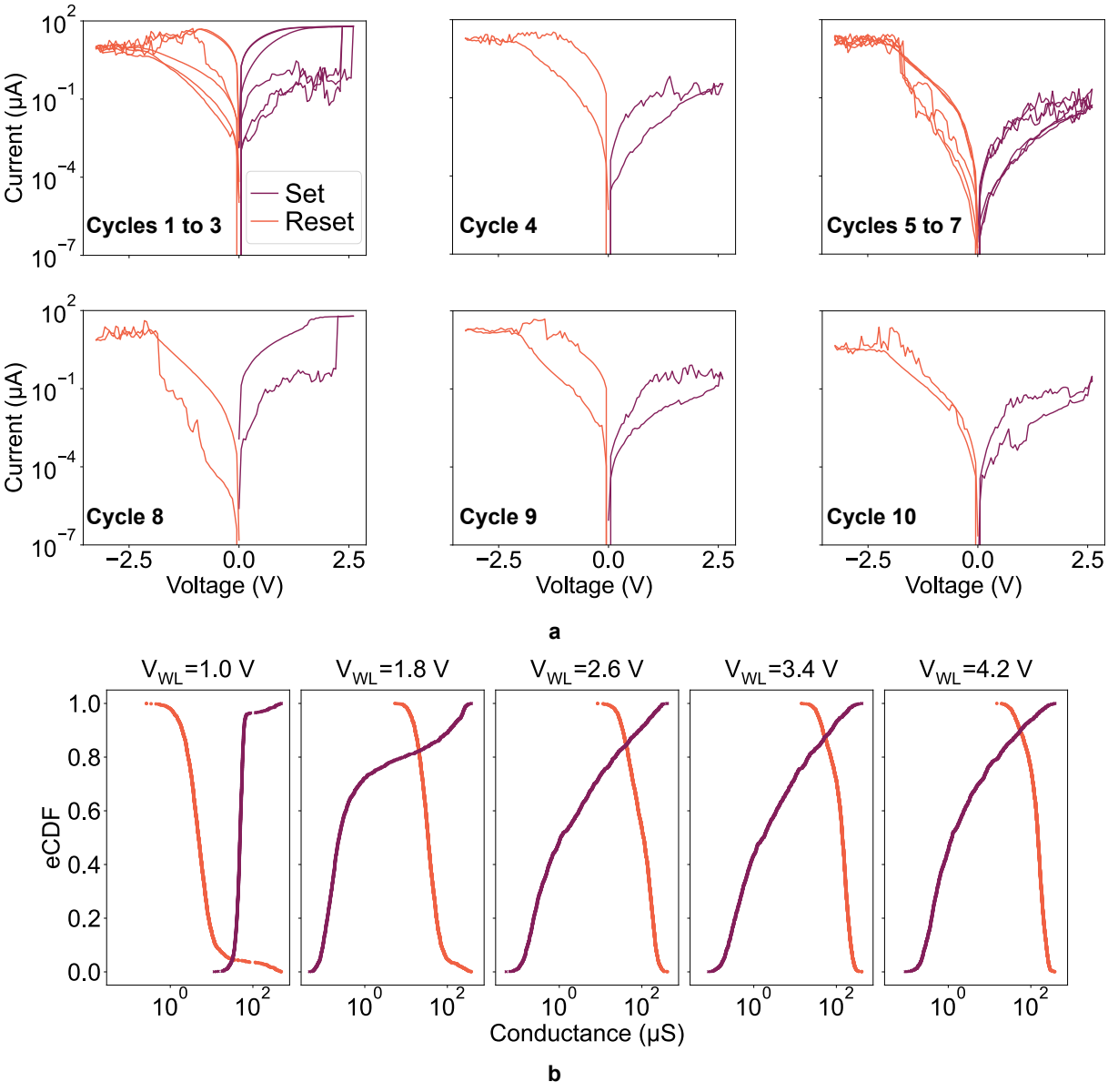

Figure 2.10: **Interplay between ferroelectric and resistive switching.** **a** Quasi-static current measurement as function of the voltage applied to the top electrode, with grounded BL in a 1T-1R structure, for 10 different cycles. During set operation, the WL is set to 1 V, during reset to 3.2 V. **b** Experimental cumulative distribution functions of set (purple) and reset (orange) conductance states for different WL voltages applied during set operation. During set, the top electrode is set to 2.3 V and the array BL is grounded. During reset, the WL voltage is set to 3.1 V and the BL voltage to 3 V, whereas the RRAMs'top electrode is grounded.

deposition control, doping level and interface design in order to enhance at the same time ferroelectric and resistive properties is not evident, as competing and contrasting phenomena can occur.
The geometry of the device also plays an important role for the optimization of resistive and ferroelectric switching characteristics. Indeed, the thickness of the Si-doped HfO_2 cannot be scaled below 10 nm to ensure the crystallization of the film in the ferroelectric orthorhombic phase while keeping the annealing temperature compatible with the BEOL fabrication process [150]. Conversely, the optimal thickness for resistive memory devices is typically thinner to reduce the forming voltage [175]. Lateral scaling is essential to increase the memory density of the devices in order to integrate them into an industrial process. This lateral scaling is expected to reduce the number of interface defects and provide higher endurance for the FeCAPs [176]. On the other hand, lateral scaling affects the forming voltage, which decreases with increasing area as a larger area offers more locations to trigger forming [175].
Finally, switching to more advanced materials, such as hafnium zirconium oxide could allow higher endurance for the FeCAPs without affecting the quality of the resistive memory response.
This qualitative analysis, combined with quantitative estimation of the impact of the different factors in the design of a unified ferroelectric/resistive device geometry and manufacturing process could ultimately allow the definition of an optimized device for a specific application.

2.5.2 Ferroelectric/resistive memory co-integration with minimal mask additional cost

An alternative strategy to the unified memory stack is the possibility to have in the same BEOL two different, yet similar, memory stacks in order to create more reliable ferroelectric and resistive memory devices.
As a general remark, designing a manufacturing process that co-integrates two different non-volatile memory technologies in the same BEOL is not a straightforward task. Different thermal budgets required to deposit various materials might not be compatible with each other. Moreover, in terms of manufacturing cost, doubling the memory technology to integrate requires additional masks and processing steps. Let us suppose that the first memory technology requires N masks and the second memory technology requires M mask to be manufactured. The total number of masks to co-integrate the two memory technologies is N+M. If the two memory technologies have to be integrated between the same metal lines, a supplementary mask is necessary to cover the wafer area where the first memory technology is developed while the second one is fabricated. Thus, the only way to decrease the total number of masks in such co-integration is to choose two memory technologies that share part of the manufacturing process. Luckily, it is the case for HfO_2-based ferroelectric and resistive memory devices.
Ref. [177] delineates a sophisticated process for the simultaneous fabrication of ferroelectric and resistive memory, aiming to streamline production and enhance device integration. The main processing steps are shown in Figure 2.11 This method starts with the deposition of a first electrode layer uniformly across both memory zones designated for FeRAM and RRAM. Following this, a layer of HfO_2-based active material is deposited identically over both zones. The idea behind the invention is indeed that the doping of the hafnium

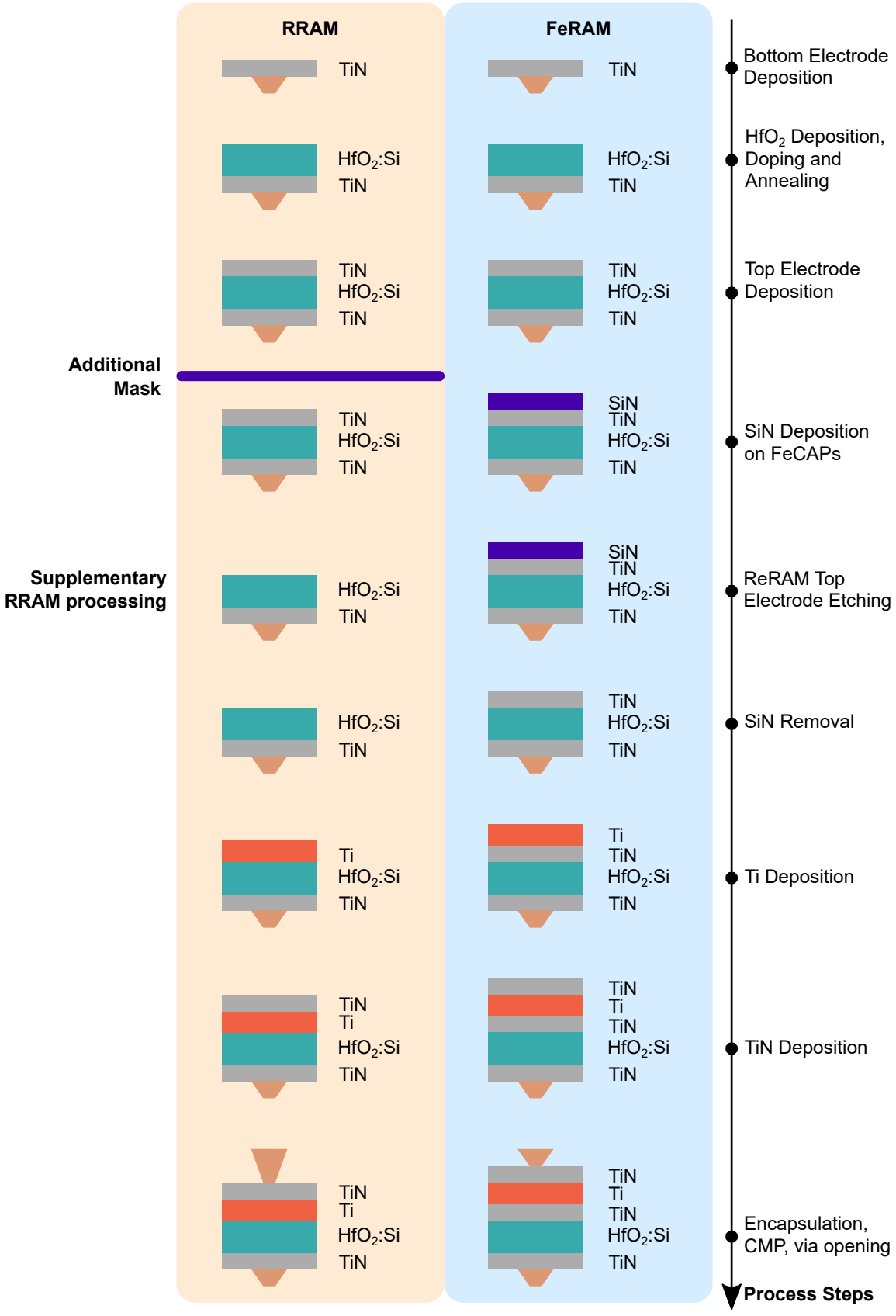

Figure 2.11: **Double integration with minimal mask additional cost.** Main process steps for co-manufacturing of ferroelectric and resistive memories based on Si-doped hafnium oxide with one supplementary mask only [177].

dioxide active layer, which is necessary for FeRAM, is also sufficient for the resistive operation. Next, a first conductive layer is applied across both memory zones. To differentiate the two types of memory during the manufacturing process, a mask is applied over the FeRAM zone, protecting it while the first conductive layer is selectively removed from the RRAM zone. This step ensures that the active material layer for the RRAM zone is exposed and ready for further processing. Once the mask is removed, a second conductive layer is deposited. This layer is crucial as it makes direct contact with the active material in the RRAM zone and is engineered to create oxygen vacancies within the HfO_2 layer, which is essential for the resistive switching properties. The process concludes with the deposition of a third conductive layer, applied identically across both memory types. This third layer ensures robust electrical connectivity and completes the device structure. Notably, this method eliminates the need for separate doping procedures and additional masking steps typically required in conventional processes, thereby reducing complexity and cost. The integration of both FeRAM and RRAM in this manner ensures that each memory type retains its unique electrical properties, making the combined device highly efficient and functional.

2.6 Summary

This chapter discussed the development of a hybrid memory device that combines the features of filamentary resistive memories and ferroelectric capacitors. This was achieved by integrating a 10 nm silicon-doped hafnium oxide film with a titanium scavenging layer into a 130 nm foundry CMOS process. This stack combines an active layer of hafnium oxide crystallized in the orthorombic phase – necessary for ferroelectric switching – with a scavenging layer – necessary for reliable resistive switching.
The hybrid memory was tested in two configurations: in FeRAM arrays and in RRAM arrays. As FeRAMs, such devices were shown to work as binary memories with good endurance over 10 million cycles and low programming energy, below 200 fJ/bit. After undergoing a forming process to create conductive filaments, the same devices integrated in the BEOL of RRAM arrays, can be used as analog multi-level memory devices with lower endurance, about 100,000 cycles. These results highlights the potential of this hybrid approach to leverage the strengths of both memory types for artificial intelligence workloads.
The next chapter shows in detail how to leverage this technology in order to create a synaptic circuit for the implementation of training and inference in deep neural networks.

Acknowledgments

For the results presented in this chapter, I would like to acknowledge Laurent Grenouillet for leading the manufacturing of the hybrid memory technology in the CEA LETI cleanroom facilities, as well as the team members who directly contributed to the fabrication process. I also extend my gratitude to Simon Martin and the whole team from the characterization lab of the DCOS department at CEA LETI for their training and support during the electrical characterization of the hybrid memory technology.

Chapter 3

Unified ferroelectric/memristive memory circuit for neural network inference and training

Resistive memories, RRAMs, are particularly promising for inference accelerators. Neural network models learned off-chip, or pre-trained, can be mapped on-chip to perform inference on previously unseen data. Nevertheless, these devices lack the possibility to undergo updates effectively because of device non-idealities and, fundamentally, due to larger programming power consumption and limited endurance. Therefore, they need to be combined with devices that can be extensively and effectively updated to enable systems with training capabilities.
In principle, such additional devices can be classical CMOS-based capacitors or SRAM cells, which offer virtually unlimited endurance. Volatile CMOS-based capacitors (DRAM-like) have the potential to scale down in size at more advanced technology nodes, but they require larger power consumption to refresh data. If used as analog devices with sufficiently large number of analog states for training, they require significantly large surface [128]. On the other hand, SRAM has a relatively large area footprint, in advanced CMOS nodes, which is problematic for learning systems that require large amounts of memory [178]. Additionally, as volatile memories, these two solutions present limitations for life-long training, which can require non-volatile storage solutions for the learning weights. Ferroelectric memories could be used as a low-power non-volatile alternative, to reliably perform updates on-chip throughout the whole training procedure. Conversely to SRAM, FeRAM devices can be packed in dense three-dimensional arrays, to store information in a digital format. Nevertheless, ferroelectric memories alone cannot enable the efficient IMC implementation of the matrix-vector-multiplication because of the data-destructive nature of the read operation.
Thus, the combination of resistive and ferroelectric memories could eventually allow to merge the IMC inference capabilities of resistive memories with the possibility to perform extensive updates on ferroelectric devices (Figure 3.1). Moreover, as observed in the previous chapter, the integration of the two technologies potentially comes without any additional manufacturing cost, compared to the integration of a single memory technology. These devices can be conceived as a single memory device, which is specialized to either ferroelectric or resistive operation by an electrical operation, ferroelectric domains

wake-up or electro-forming respectively.
Application-specific integrated circuits are key enablers for optimizing energy efficiency in hardware implementations. In this chapter, a proof-of-concept ASIC is presented that leverages the unified ferroelectric/resistive memory stack for the implementation of back-propagation-based learning algorithms.
First, the motivations justifying the development of the proposed circuit are outlined. Then, two different system implementations are discussed. For each implementation, electrical characterization results and system-level simulations are presented, validating the potential of the proposed approach.

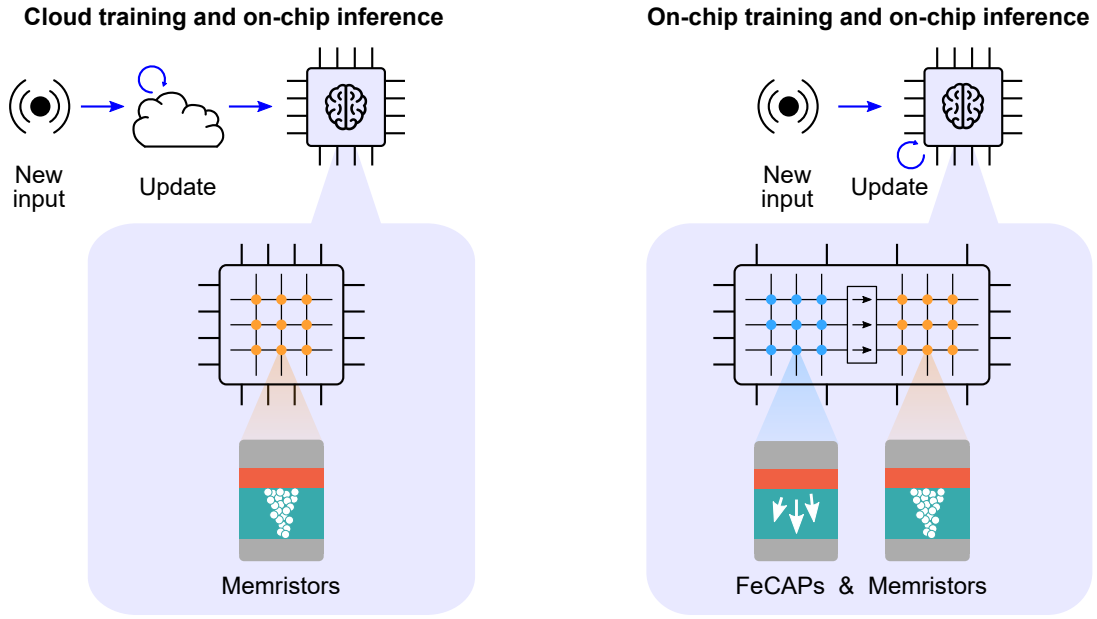

Figure 3.1: **A single memory stack, which functions both as memristor and ferroelectric capacitor, for neural network inference and training.** High-level description of the training procedures implemented off-chip (left) and on-chip (right). For on-chip inference, only pre-programmed memristors are required to store the analog weights. For on-chip training, each weight is associated with a lower precision value - used for inference - that is stored in memristors, and a higher-precision hidden value - for use only during training - that is stored in FeCAPs.

3.1 Assumptions and definition of the training strategy

Before diving into the circuit implementation, it is essential to understand the motivations supporting the design of this circuit.
In artificial neural networks, the precision requirements for weights and activations vary depending on whether the network is in the training or inference phase. During training, small updates are applied to the model parameters to converge toward an optimal solution for a given dataset. The exact magnitude of these "small" updates remains a topic of debate. For instance, some studies suggest that tracking only the direction of these updates may suffice for training various neural network architectures [129, 179, 180]. However, sufficient precision is still necessary for the weights to ensure proper accumulation of gradients throughout the training process.

On the other hand, lower precision is required for model parameters during inference to achieve near-equivalent accuracy compared to a high-precision baseline. In fact, training techniques for quantized neural networks have demonstrated minimal accuracy loss, even with very low-bit precision down to a single bit for weights and activations. During the training of quantized neural networks, lower-precision weights are simulated in the forward pass, while higher-precision weights are used for optimization. This approach ensures that the final quantized models deployed for inference maintain minimal accuracy loss.
The memory technology proposed in chapter 2 can be exploited to satisfy the precision requirements for embedding ANNs training on-chip. On one side, ferroelectric memories can store data in a digital format – encoding 1-bit information in each FeCAP – resulting in a numerical precision imposed by the selected data format (integer unsigned, two's complement, floating point, etc..). On the other hand, resistive memory devices can be used in the analog domain to map a weight value into a conductance level. Nevertheless device non-idealities, such as time relaxation over time and read noise, can limit the equivalent precision of these devices, which are used as multi-level cells (MLCs) with few levels per cell.

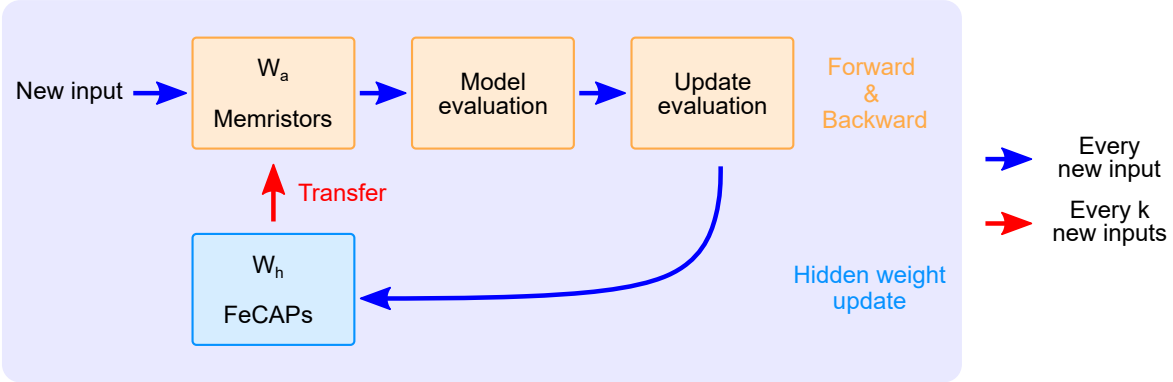

Figure 3.2: **High-level description of the training strategy leveraging the unified memory technology.** The multiply-and-accumulate operations, critical for forward and backward data propagation to train the artificial neural network, as well as for inference, are performed in situ using the memristor devices. The higher-precision FeCAP memory is updated upon receiving each new input, whereas the memristor analog weights are only periodically updated.

Therefore, taking inspiration from quantized neural networks, the key ideas of the training strategy (Figure 3.2) used throughout this chapter are the following:

- Each weight is associated with a lower precision value (W_a) – used for inference – that is stored in one or more memristive devices, and a higher-precision hidden value (W_h) – for use only during training – that is stored in FeCAPs.
- Each training sample is presented to the network individually. Indeed, moving from a mini-batch to a stochastic gradient descent implementation reduces the memory footprint for the accessory information required to evaluate weight updates. Moreover, learning in an online fashion, sample by sample, can be more adapted to edge applications, where data can be envisioned to arrive sequentially rather than in batches.

- The forward and backward propagation operations are performed by means of the memristive memory array. The activations and error gradients evaluated at different layers are used to evaluate the weight gradients. Activations and error gradients evaluation requires performing read operations of the current model, which can be extensively performed by means of memristive arrays.
- The hidden weights, stored in FeCAPs, are updated for each presented training sample, whereas the analog weights stored in memristive devices are updated each k new inputs. This procedure ensures the convergence of the training algorithm, which also requires updating the weights W_a to reliably perform the forward and backward steps, while also reducing the overall number of programming operations on memristive devices by a factor of k.

The circuit implementation described in the following deals with an efficient implementation of the transfer operation of the hidden weights values, stored in FeCAPs, to analog weights values, stored in memristors. This circuit takes advantage of the read operation of FeCAPs to program the memristors, thus removing the need of digital to analog converters for the analog weights programming.
The idea of exploiting ferroelectric and resistive memory technologies for the implementation of learning algorithms was also introduced in [181]. This patent describes a fully digital implementation of a generic training algorithm. Here, the weights, stored as N-bits digital quantities are split into two parts: a high-significance sub-word and a low-significance sub-word. The high significance sub-word is stored in resistive memories, updated less frequently during training and used extensively during inference, and the low significance one in ferroelectric devices, only used during training. Different configurations are discussed according to the overlap between low and high significance sub-words. The approach proposed in this chapter, although similar in the basic principles to the one discussed in [181], goes beyond a digital implementation, which could offer more flexibility at the cost of reduced energy-efficiency.

3.2 The unified ferroelectric/memristive memory circuit

A hybrid array comprising FeCAPs and memristors was designed and fabricated. The front-end CMOS devices and first four metal lines were fabricated in a 130 nm commercial technology process and BEOL memories and last metal layer were deposited in the CEA LETI cleanroom facilities.
The array includes 128 vertical transfer lines (TL), each with 128 FeCAPs and 16 memristive devices (Figure 3.3a). This circuit also included on-chip CMOS components such as registers to address each memristor and FeCAP device, drivers to program the devices, sense amplifiers for the FeCAP devices, and a timing block for automatic pulse generation for the FeCAPs array. The digital peripheral circuitry of the hybrid memory array is supplied with 4.8 V. A 1 µm wide minimum length selector transistor is serially connected to the memristors, and a 500 nm wide minimum length access transistor is used for the FeCAPs. Both FeCAPs and memristors are square structures with 600 nm side. Details of the whole array design can be found in Appendix B. An optical micrograph of the circuit is shown in Figure 3.3b.

The basic building block of the hybrid array is the synapse circuit [182] depicted in Figure 3.3c. This circuit comprises an ensemble of one-transistor-one-FeCAP cells, where the bit line of each cell is directly connected to the gate of the selection transistors of a one-transistor-one-memristor cell. This circuit is used as part of a vertical transfer line: the right numbers of FeCAPs and memristors in a vertical transfer line can be activated, using the word lines of the devices access transistors. This sub-circuit allows direct digital-to-analog data transfer from n FeCAP cells to one memristive device without requiring intermediate circuits.
The transfer is implemented by directly reading in parallel the n FeCAPs of the hybrid synapse. This operation loads of the transfer line parasitic capacitance (C_{TL}) to a voltage level that depends on the data stored in the plurality of ferroelectric capacitors. This voltage can be expressed as:

$$V_{TL} = \frac{\sum_{i=0}^{n-1} C_{D,i} + 2P_R \sum_{i=0}^{n-1} S_i \delta_i / V_{SL,Fe}}{\sum_{i=0}^{n-1} C_{D,i} + 2P_R \sum_{i=0}^{n-1} S_i \delta_i / V_{SL,Fe} + C_{TL}} V_{SL,Fe} \tag{3.1}$$

where $C_{D,i}$ is dielectric component of the i-th capacitance, S_i the area of the i-th capacitor, δ_i is equal to one or zero depending on the polarization state stored into the i-th ferroelectric capacitor, P_R the remanent polarization and $V_{SL,Fe}$ the applied read voltage of FeCAPs. Ferroelectric capacitors with equal thickness of the ferroelectric layer are considered. All ferroelectric capacitors share the same film thickness, so to have the same switching voltage, which is an essential requirement for the reliable parallel read operation.
Thus, this FeCAPs-data-dependent voltage is then used to set the compliance current flowing into the memristive device, while performing a set operation, effectively adjusting its conductance. Indeed, the compliance current of the memristor selector transistor is function of its gate to source voltage, which is the difference between the transfer line voltage and the bias imposed to the bit line of the 1T-1R cell (BL_{mem}). The tuning of this bias voltage offers a supplementary degree of freedom to optimize the transfer operation. This transfer operation can be used in a program-verify scheme to precisely tune the conductance of the memristor, as depicted in Figure 3.3d. The procedure starts by resetting the target memristor. Then, before implementing the transfer operation, the content of the source FeCAP devices is copied in a temporary/cache memory. Saving data in a cache memory is necessary because the destructive read operation of ferroelectric capacitors, combined with the summation operation performed during transfer makes the FeRAM-data unrecoverable. Indeed, polarization reversal occurs simultaneously for all devices programmed in the 1 state during the parallel read operation, removing the possibility to discriminate the information stored in each capacitor individually, for an eventual subsequent write-back. After source FeCAPs data are stored in a cache memory, the transfer operation is performed, which tunes the memristor conductance. The source FeCAPs are therefore rewritten with the data stored in the cache memory. Then, the memristor conductance is compared to the data stored in the cache memory to check if data is correctly transferred from the FeCAPs to the memristor cell. If not, a new iteration is performed, otherwise the procedure ends. The iterative transfer can stop if a maximum number of iterations is reached. Depending on the application, a single iteration of the program-verify scheme might be sufficient, removing the need to compare data in the cache memory to the programmed conductance, hence simplifying the system implementation.
Although a single transfer iteration might be sufficient to program the memristive device

effectively, a cache memory is still required to reload the source FeCAPs data after the transfer. However, the size of the temporary memory needed to store the source FeCAPs data remains relatively small. When transferring information from FeCAPs to memristors, a set operation is executed. Due to peak power limitations in embedded devices, the maximum current that can be drawn restricts the number of memristors that can be programmed in parallel to only a few tens at most. Assuming that M memristors can be programmed simultaneously and data is transferred from n FeCAPs, the size of the required cache memory is n × M bits. As it is shown later, for reliable transfer, n is typically kept below 10, resulting in a maximum cache memory size of only a few hundreds of bits.
Two circuit implementations and their training strategies are presented in the following. The first involves transferring data from n equally sized FeCAPs to a single memristor, taking advantage of the fabricated array as is, where all ferroelectric capacitors have the same size. The second implementation, more advanced, uses capacitors of varying sizes to transfer data into a differential memristor cell. This second circuit is reliably emulated by means of the fabricated array.

3.3 Initial implementation of the hybrid memory circuit

All ferroelectric capacitors in the fabricated hybrid array have the same size. Therefore, equation 3.1, expressing the transfer line voltage level when reading multiple ferroelectric capacitors in parallel can be rewritten as:

$$V_{TL} = \frac{nC_D + N_{SW}2P_RS/V_{SL,Fe}}{nC_D + N_{SW}2P_RS/V_{SL,Fe} + C_{TL}}V_{SL,Fe} \tag{3.2}$$

where C_D is the dielectric component of each ferroelectric capacitor and N_{SW} is the number of ferroelectric capacitors programmed in the 1 state, i.e. the number of capacitors switching during the parallel read operation performed during data transfer. Therefore, the transfer line voltage and, by consequence the memristor conductance, is function of the sum of the number of devices programmed in the 1 state, rather than the positional information of each data stored in the capacitors. Therefore, n capacitors of the same size can provide n+1 distinct transfer line voltage levels. The array of hybrid synapses was electrically characterized in this configuration and a learning strategy was co-developed to train a binarized neural network for detecting heartbeat anomalies [183].

3.3.1 Electrical characterization of the hybrid memory circuit

An hybrid array of synaptic circuits was characterized to validate the reliability of the transfer operation from n FeCAPs to a memristive device. Figure 3.4 a shows the dependence of the TL voltage on the number of activated WL_{Fe} (n), when the respective SL_{Fe} lines are pulsed, for all devices pre-set to the 0 or 1 states. For n=8, the voltage difference of approximately 200 mV between the configurations with all devices at 0 and 1 states aligns with the required programming range for multilevel operation in memristors (cf. Figure 2.9b in chapter 2). This graph was obtained by means of the sense amplifier circuits of FeRAM arrays. As in the case of Figure 2.7a in chapter 2, where the reference

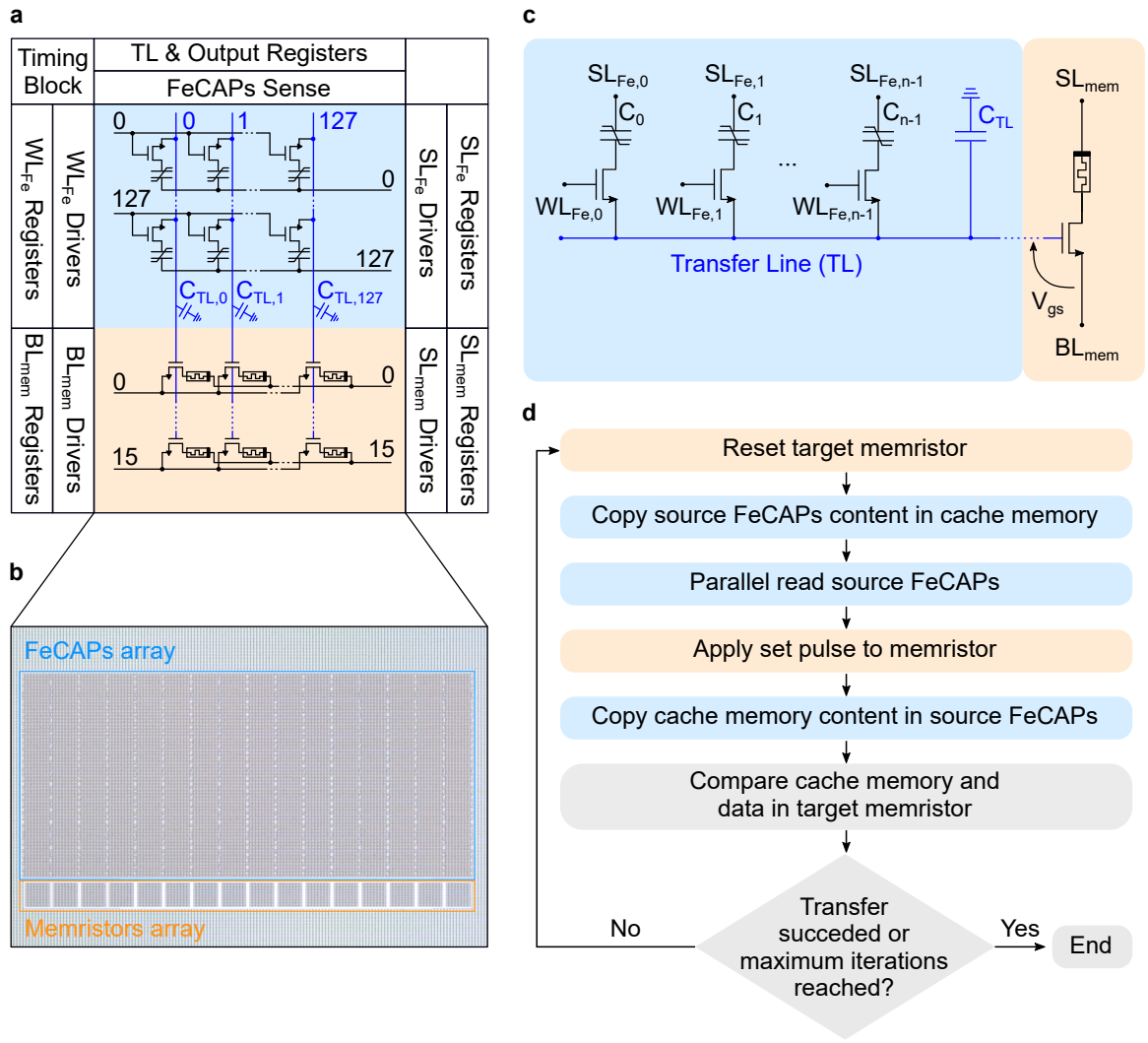

Figure 3.3: **Hybrid FeCAP/memristor memory circuit and array. a**. Array-level organization of the hybrid memories and peripheral circuit, and optical micro-graph of the fabricated array. **b** Schematic representation of the hybrid FeCAP/memristor memory circuit. **c** Flow-chart describing the steps to implement the transfer operation.

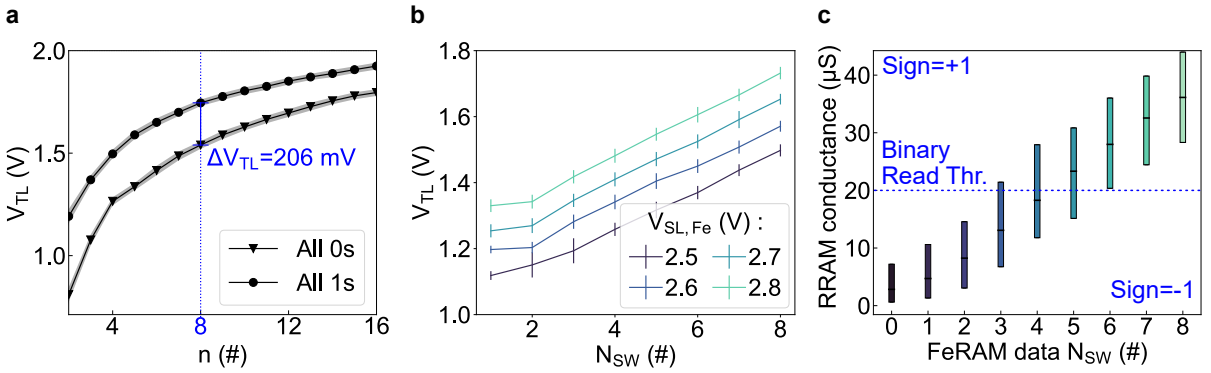

Figure 3.4: **Electrical characterization of the analog transfer operation. a** Average TL voltage measured as a function of the number of activated WLs in 1T-1C cells (n), with all FeCAPs pre-programmed to either 0 or 1. Shaded area corresponds to standard deviation at 1σ measured over 126 synaptic circuits. **b** Average TL voltage for n=8 as a function of the number of 1 states programmed in the FeCAPs (N_{SW}). The measurements have been repeated for different SL pulse amplitudes. Vertical lines correspond to the standard deviation at 1σ measured over 126 synaptic circuits. **c** Measured data transfer from 8 FeCAPs to one memristor cell. Median and inter-quartile ranges are shown as boxplots for 107 synaptic circuits.

voltage of the sense amplifier was ramped to evaluate the bit line voltage when reading a zero or a one in a FeCAP, the same procedure can be applied for evaluating the transfer line voltage when reading multiple capacitors in parallel. This characterization was performed with 3 V amplitude, 2 µs long programming and read pulses.
At fixed n, the number of 1s (N_{SW}) and 0s (n-N_{SW}) written in the ferroelectric capacitors tunes the transfer line voltage, as shown in Figure 3.4b. The amplitude of the SL_{Fe} pulse can be adjusted to change the transfer line voltage level. For this characterization, 2 µs long programming and read pulses were used.
Figure 3.4c illustrates the measured data transfer from the FeCAPs data to the memristor. The programmed conductance shows a direct proportionality to the number of 1s stored in FeCAPs. As the number of stored 1s decreases, the transfer line voltage correspondingly drops. Lower transfer line voltages lead to reduced gate-to-source voltages associated with the memristor selector transistor, which in turn result in lower compliance currents and, consequently, reduced programmed conductance (cf. Figure 2.9e). Data stored in memristors can be binarized through a low-power binary reading, if this is to be used as a binary device. This characterization was performed with 3.25 V-10 µs pulses applied to the SL of the 1T-1FeCAP cells, while the memristor top electrode was pulsed with 2.3 V-10 µs and the memristor BL was grounded. The test setup used to perform these measurements is presented in Appendix A.

3.3.2 Binarized neural network training for ECG anomaly detection

A training strategy was co-developed with the designed structure, in order to optimize the overall performance and taking into account the strengths and limitations of the hybrid memory circuit. A back-propagation-like algorithm was implemented to train a 2-layers fully connected BNN on the ECG-arrhythmia detection task, with 512 neurons in the

hidden layer and one sigmoid output neuron was used (Figure 3.5a).
The MIT-BIH heart arrhythmia database [184] was used for the arrhythmic heartbeat detection task. It includes half-hour dual-channel electrocardiogram recordings from 48 subjects. The Python Numpy FFT function was used to generate a frequency spectrum of each recording and the 64 features with the highest values for the test chi-squared statistic [185] were used as input features to train/test the model. Each heartbeat was labelled as either a healthy or arrhythmic heartbeat. A subset of 18,000 data points were taken randomly from all subjects and used to train the ANN, whereas a subset of 2,000 previously unseen data points were used to test the model.
Typically, the implementation of training of BNN requires not only binarized weights ($W_b = \pm 1$), but also a full precision (W_h) version of the parameters of the model at train-time (Figure 3.5a). Nevertheless, the designed synapse is not suitable to store the full precision synaptic weights in a floating-point format, rather as integer values of very limited precision. The hybrid synaptic circuit described in the previous section can be used to store signed integers in FeCAPs. In the case of n FeCAPs connected in parallel, with even n, the state in which all capacitors are programmed in the 0 state can map the weight value -n/2, vice versa the state in which all capacitors are programmed in the 1 state can map the weight value +n/2. The transfer operation with the associated binary reading of the memristor performs the binarization of the hidden weights.
Figure 3.5b schematically describes the training procedure. During the feed-forward and backward phases of training, the binarized version of the parameters of the model, stored into the memristor array and read to evaluate the activations (forward cache) and gradients (errors and backward cache) at different layers of the network. The evaluated error gradients are clipped and ternarized to either 0 or ± 1 with probability proportional to their relative magnitude. These are then used to evaluate the weight updates implemented on the hidden weights stored in the FeCAPs array. Hidden weights are initialised to either 0 or ± 1. The weight update, equal to the product of the respective binarized activation and ternarized gradient, is then applied or not with a given probability. In can therefore assume three different values: 0 or ± 1. The hidden weights are updated at each sample in the FeCAPs array and the sign is transferred every k=100 samples to the binarized weights in the RRAM array. As the purpose of the designed algorithm and hardware support is to implement the training on-chip, each training sample is processed just once, i.e. the dataset is iterated for a single training epoch. Finally, once training is completed, only the binarized weights are used to evaluate the model and perform inference on previously unseen samples.
An interesting aspect connected with the probabilistic nature of the weight update is that it could be applied with or without reading the value of the hidden weight. Two update strategies are proposed:

- Read and update: Each time an update is prescribed, i.e. the weight update is equal to ± 1, the hidden weight value is read from the FeRAM array, the weight update is digitally processed, and the updated hidden value is then stored back in the FeRAM array.
- Blind update: Each time an update is prescribed, i.e. the weight update is equal to +1 or -1, one randomly picked FeCAP in the synaptic circuit is set or reset respectively, without knowing if said FeCAP was already set or reset. This could result in a missed update if the set (or reset) pulse is applied to a FeCAP already programmed in a set (or reset) state.

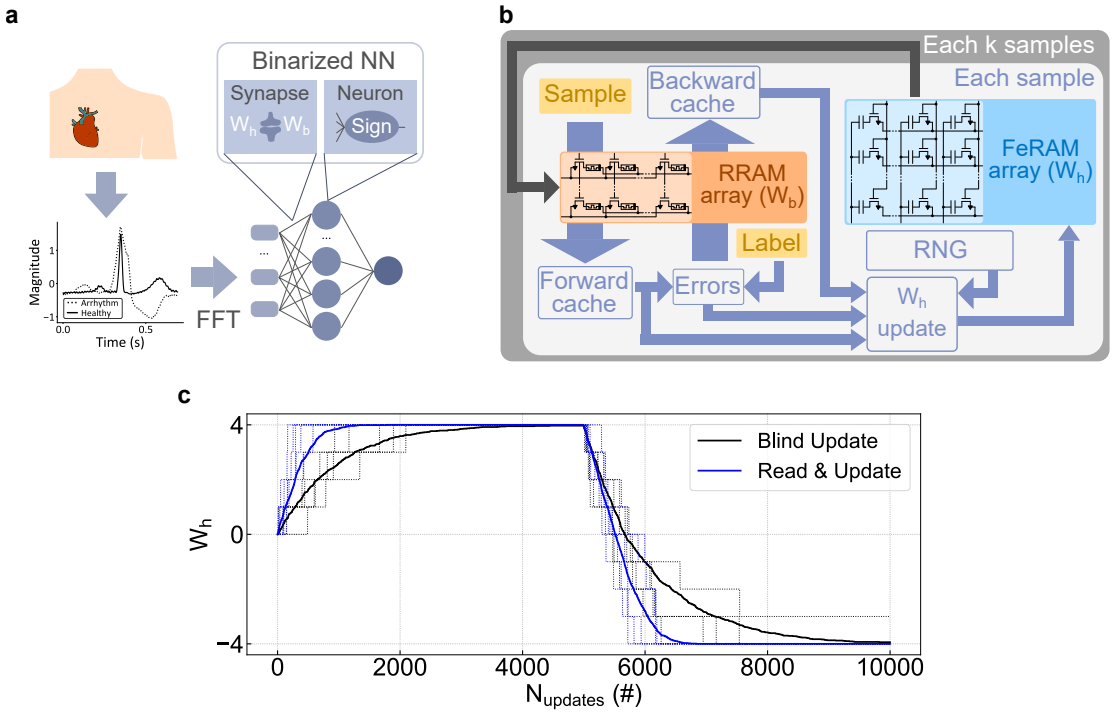

Figure 3.5: **Learning to detect ECG anomalies with the hybrid Fe-CAP/memristor memory circuit. a** ECG signals converted into 64 features through Fast Fourier Transform (FFT), are used as inputs for a two-layer fully-connected binarized neural network. **b** Detailed flow diagram of the training algorithm implementation using the hybrid memory circuit: the FeRAM array is utilized to store the hidden values (W_h) used during training, while the RRAM array is used for the binary weights (W_b). **c** Simulated synapse plasticity achieved on n=8 using a probabilistic programming scheme with p=10^{-3}. The blue lines represent the weight read before being updated, while the black lines represent the weight updated blindly. Solid lines are evaluated as average over 1000 tests. Five test examples are shown with dotted lines.

Figure 3.5c shows the simulated plastic response of a synapse, made of 8 probabilistic binary components, to 5,000 potentiation updates, followed by 5,000 depression updates, in the cases in which the weight is read before applying the update or this is blindly applied. Convergence to the saturation value is reached faster in the case in which the hidden weight is read before being updated, for the same update probability. Updating the hidden weights blindly during training leverages the properties of each device optimally: FeCAPs devices are written at each training iteration, but read only during weight transfer operations. Conversely, memristor devices are read at each learning and inference step, but programmed only during weight transfer operations.
Table 3.1 presents the table of simulated accuracy evaluated on the test set for the two weight update strategies, with update probability of $5{\cdot}10^{-4}$. The simulations were performed for different number of FeCAPs in the hybrid synapse, i.e. n equal to 4, 8 or 16. Accuracy values between 85% and 90% were observed, with increasing average accuracy and decreasing standard deviation for larger values of n. Performing updates blindly resulted in slightly inferior accuracy. Moving from 8 to 16 FeCAPs in the hybrid memory circuit improves the accuracy of approximately 1.1%, for a "blind update" strategy, and 0.3% only, for a "read and update strategy". Therefore, n=8 could be selected for this specific application.

Training Method	**Test accuracy(%)**		
	n=4	**n=8**	**n=16**
Read and Update	87.48 ± 1.89	89.03 ± 1.08	89.30 ± 0.89
Blind Update	85.23 ± 3.82	88.04 ± 1.42	89.15 ± 1.22

Table 3.1: Simulated test accuracies for the ECG anomaly detection task. Mean and standard deviation were evaluated over 10 runs.

3.3.3 Discussion

The proposed training approach is appealing for the implementation of the weight update strategy. In particular, the blind update strategy could accelerate the weight update step, as it does not require reading the specific value stored in each FeCAP before applying the update. With minimal circuit overhead, all potentiation updates could be applied in parallel across the entire array, and similarly for all depression updates.
Nevertheless, this training strategy was tested on more complex tasks, e.g. the MNIST digits classification, with unsatisfying results. The reason for this is primarly due to the extremely limited precision of the hidden weights. Although it could be possible to increase the number of FeCAPs in the hybrid synaptic circuit in order to increase the precision of the hidden weights, this would result in an unaffordable memory cost, because of the impossibility to distinguish the significance of the information stored in each FeCAP. Moreover, the reliability of the transfer operation could be negatively impacted by increasing too much the number of FeCAPs in the hybrid synaptic circuit, as the transfer line voltage range decreases with increasing number of FeCAPs read in parallel, as shown in Figure 3.4a.
Finally, this implementation does not leverage the multi-level capability of memristive devices and their implementation in energy-efficient IMC inference engines. Indeed, the weights used during inference would be binarized with a sensing circuit (not implemented

in this design), reading each memristor individually.
The optimized implementation of the hybrid memory circuit proposed in the next section addresses all the aforementioned issues, resulting in satisfying performance across a variety of benchmarks.

3.4 Optimized implementation of the hybrid memory circuit

This section details an optimized implementation of the hybrid memory circuit, leveraging more compact data formats for the hidden weight storage. Moreover, the digital to analog conversion implemented during data transfer in this second implementation leverages the multi-level capability of memristors to store positive and negative weights in memristors, to be used in the design of IMC inference engines. As shown in the following, the enabling idea of this circuit is the exploitation of FeCAPs of various size in the synaptic circuit. Therefore, the circuit implementation presented here is an extrapolation based on the designed synapse array described in the previous section, which includes equally sized capacitors. First, the concept of the optimized memory circuit and the emulated electrical characterization of the transfer operation are presented. Then, the co-developed training strategy is presented. For clarity, the main results are discussed in sections 3.4.1 and 3.4.2, with details of the various procedures reported in section 3.4.3. Finally, a discussion and outlook is presented to summarize key findings and explore potential future directions.

3.4.1 Concept and electrical characterization

The basic building block of the optimized hybrid FeCAP/memristor array is the synapse circuit depicted in Figure 3.6a. This circuit comprises an ensemble of one-transistor-one-FeCAP cells, where the bit line of each cell is directly connected to the gates of the selection transistors of two one-transistor-one-memristor cells.
In this implementation, the hidden weights used during training are stored in FeCAPs in a sign-and-magnitude 10-bit integer format, using one bit for the sign and nine for the magnitude. The first three magnitude bits are referred to as the most significant bits (MSBs) and the last six as the least significant bits (LSBs). The sign-and-magnitude representation, rather than the more conventional two's complement representation, facilitates data transfer operations. The analog weight used for forward and backward propagation steps, as well as inference, is stored in two memristor cells in a differential configuration, allowing both positive and negative values to be stored.
An example elucidating data transfer from high-precision hidden weights to analog memristors is shown in Figure 3.6b. It is carried out through the following steps:

1. **Reset the Memristor Cells**: The first step is to reset the two memristor cells to the low conductance state.

2. **Read the Sign Bit**: The sign bit stored in the first FeCAP cell is read to determine which memristor cell should be programmed. In this example, the positive memristor cell is selected.

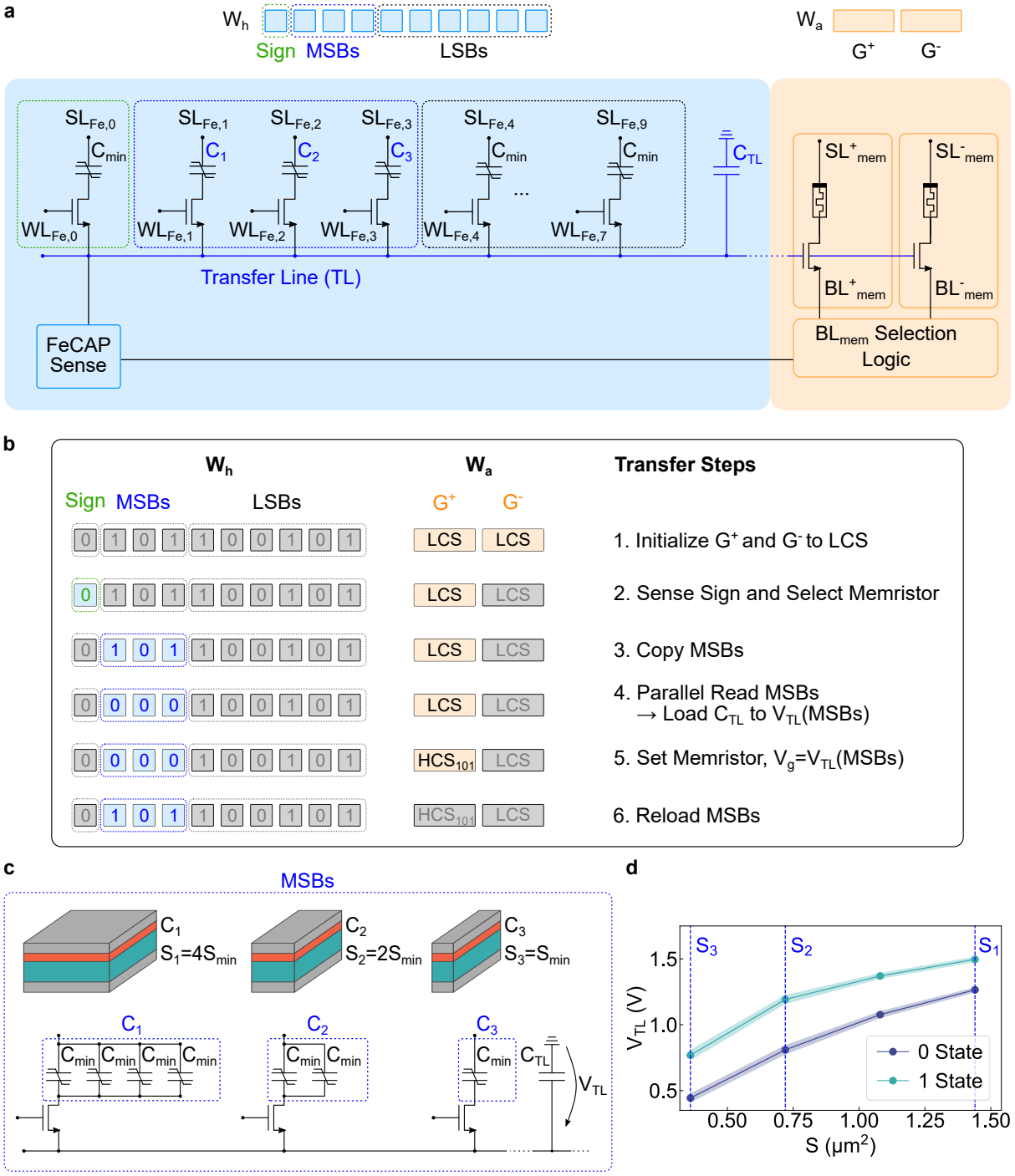

Figure 3.6: **Improved hybrid FeCAP/memristor memory circuit.** **a** Schematic representation of the improved hybrid FeCAP/memristor memory circuit. **b** Example of a data transfer operation from the FeCAPs to the analog memristors, elucidating all the steps the FeCAP and memristors devices undergo. **c** To account for the significance of the MSBs during the data transfer process, the MSBs are implemented using capacitors with distinct areas: four times the minimum area, twice the minimum area, and minimum area, respectively for the first, second, and third MSBs. **d** Average voltage measurements after loading the transfer line while reading a zero or a one as a function of the capacitor area, with standard deviation at 1 σ, measured over 126 transfer lines.

3. **Store the MSBs Temporarily**: The three MSBs from the FeCAP cells are temporarily stored elsewhere. Only three additional memory bits are required for this operation.

4. **Read the MSB FeCAP Cells in Parallel**: The MSB FeCAP cells are read in parallel, which loads the transfer line parasitic capacitance to a voltage level determined by the data stored in the three FeCAPs.

5. **Program the Selected Memristor Cell**: The selected memristor cell is programmed to the high conductance state, with a compliance current determined by the transfer line voltage. A voltage pulse is applied to the to-be-programmed memristor source line (SL_{Re}) while the memristor bit line (BL_{Re}) is grounded. This process eliminates the need for symmetrical, linear updates of memristor values, allowing for reset and set operations with compliance current determined by the FeCAPs' stored values. During this operation, the data stored in the FeCAPs are lost, resulting in all devices being set to zero.

6. **Reload the MSBs**: The MSBs, which were previously stored elsewhere, should be reloaded after the transfer operation. This reloading is necessary to resume training after the transfer operation in the same configuration as before updating the memristor's differential pair.

To account for the relative significance of MSBs, capacitors of different areas are used: four times, twice, and the minimum area for the first, second, and third MSBs, respectively (Figure 3.6c). This choice results in near-linear spacing between V_{TL} values for different bit combinations (details in section 3.4.3). A hybrid memory with distinct capacitor sizes for the three MSBs can be implemented by connecting a 1T-4C, 1T-2C, and 1T-1C to the same transfer line, all with minimum-sized capacitors (Figure 3.6c). Capacitors for the sign bit and six LSBs are minimum-sized as they do not participate in the analog transfer.
The direct transfer from digital words (FeCAPs) to analog conductance values (memristors) was validated using the fabricated hybrid array. Figure 3.6d shows the transfer line voltage measured when reading a zero or a one for one to four capacitors connected in parallel. The possibility to emulate the transfer of the improved memory circuit using the fabricated array is detailed in section 3.4.3. The measured transfer line voltage follows the equivalent areas, defined as the area of a single capacitor multiplied by the number of capacitors read in parallel and the state stored in the capacitor, i.e., zero or one. This curve confirms that capacitors with varied sizes can be exploited to reflect for the significance of the MSBs.
Finally, Figure 3.7 presents the measured data for the digital-to-analog transfer of the sign (green) and the three MSBs (blue) from the FeCAP to the two memristor cells. The difference in conductance between the two memristors encoding the positive weight (G^+) and the negative weight (G^-) was measured for all possible sign and MSBs combinations. The integer values from -7 to 7, plus zero (represented by two combinations), stored in the FeCAPs were successfully transferred as 15 distinct differential conductance levels. The electrical characterization details of the transfer procedure are provided in section 3.4.3.

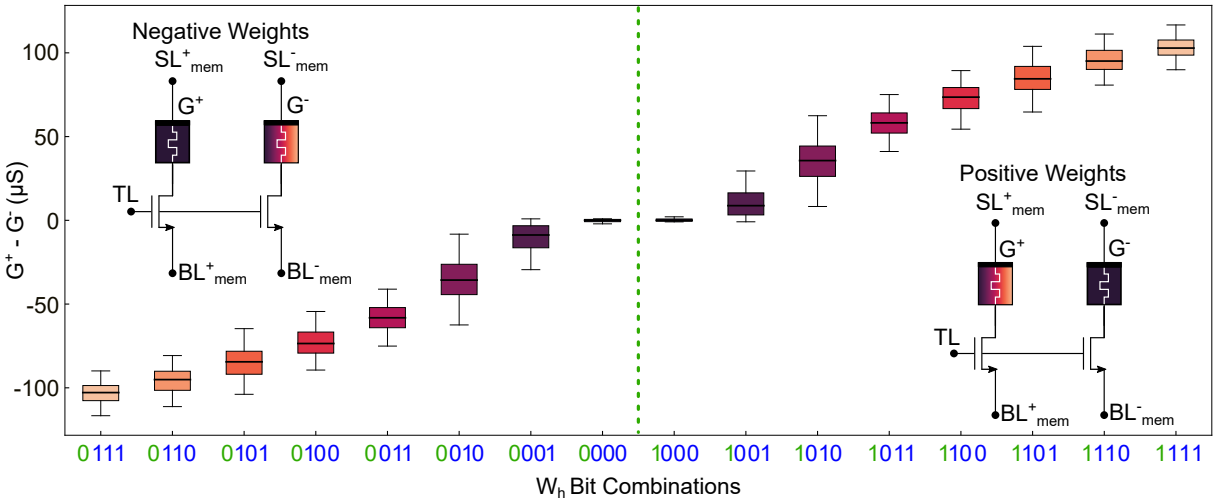

Figure 3.7: **Electrical characterization of the transfer operation in the improved hybrid FeCAP/memristor memory circuit.** Memristor conductance measured after transfer as a function of the sign (green) and MSBs (blue) stored in the FeCAP. Interquartile ranges (boxes) and standard deviation at 1σ (error bars) are evaluated for 33 hybrid memory circuits. The transfer is repeated 6 times for each synapse.

3.4.2 Neural network training with the optimized hybrid memory circuit

The performance and power consumption of the proposed FeCAP/memristor memory circuit for on-chip training and transfer learning were evaluated using hardware-aware simulation validated by extensive statistical measurements (Figure 3.7). The core idea behind this approach is that while the analog precision of memristors suffices for inference, it falls short for learning, which requires small, progressive weight adjustments. Inspired by QNNs, a hybrid approach was adopted: Forward and backward passes use low-precision weights stored in analog in memristors, while updates are done using higher-precision FeCAPs. Memristors are periodically reprogrammed based on the MSBs stored in FeCAPs, ensuring efficient and accurate learning.

This approach was implemented in a three-layer fully connected neural network for the MNIST digit classification task. Figure 3.8a depicts the on-chip training procedure using SGD. For each training sample, neuron activations are calculated by feed-forward matrix-vector multiplication between the analog weights (memristors) and the previous layer's activations. Errors at the output layer are back-propagated to evaluate loss gradients and update hidden weights. The hidden weight updates matrix, $\Delta\ W_h$, is multiplied by a binary mask M dictating which weights to update in FeCAPs. Mask elements follow a Bernoulli distribution with a given probability, p. This stochastic update, loosely inspired by biophoton emission and propagation in the brain [186], enhances convergence (details in section 3.4.3)[187, 188]. No momentum is used to minimize hardware costs. Hidden weights (10-bit integers in FeCAPs) are updated for each sample, while analog weights (memristors) are updated every k inputs via the digital-to-analog transfer procedure. Further details about the training algorithm, network architecture and hyperparameters are in section 3.4.3. Also, the pseudo-code elucidating the neural network training procedure is presented in Appendix C.

Accuracy, total programming energy, and number of programming operations (details in section 3.4.3) were evaluated for FeCAP and memristor devices at the end of training on

the MNIST task against the parameter k, i.e. the amount of samples between two analog weights updates. Figure 3.8c shows that test accuracy remains stable for k values up to 100 but drops for larger values. The impact of different update probabilities was evaluated at fixed k=100, showing that the probabilistic weight update strategy effectively improves convergence of the training algorithm (Figure 3.8b). Decreasing k increases the number of both memristor and FeCAP programming operations (Figure 3.8d), which increases energy consumption (Figure 3.8e). With k values lower than ten, the number of required memristor programming operations also exceeds the endurance of memristors. For k values greater than 50, memristor programming energy drops below the one of FeCAP, stabilizing between 20 nJ and 30 nJ: The programming energy is dominated by hidden weight updates rather than the transfer operations. At k=100, the method achieves 96.7% accuracy with approximately 38 nJ total programming energy consumption, representing a 38-fold reduction with no loss of accuracy with respect to k=1. The number of programming operations remains 17 times below the memristor endurance limit and 75 times below the FeCAP limit. Therefore, k=100 was used for further analysis.
The robustness to memory errors was also assessed. Figure 3.8f compares accuracy for MNIST, Fashion-MNIST, and ECG detection tasks using classical ANN (FP) and the hybrid memory (HM) approach, with and without induced transfer errors due to device variability. Device variability was estimated from the characterization of electrical transfer in Figure 3.8. Details on the architectures and hyperparameters for Fashion-MNIST and ECG tasks, as well as the procedure to take into account device variability in simulations, are in section 3.4.3. Training with full-precision weights yields the highest accuracy. For all three datasets, the hybrid memory approach with hidden weight quantization reduces accuracy by, approximately, one percentage point, and transfer-induced errors by, approximately, one additional percentage point.
All these simulations were performed with "online" training, i.e., updating the hidden weights after each training sample presentation. Figure 3.8f also shows that using mini-batch training with batch normalization (Figure 3.8f) would improve accuracy and resilience to transfer errors, with only a 0.8 percentage point loss on MNIST between full precision and hybrid memory with errors. However, this approach limits continuous on-chip learning due to memory costs for storing activation values and batch normalization parameters.
Finally, the proposed training approach was benchmarked on a transfer-learning task. To create a transfer-learning scenario, an edge-friendly neural network, MobileNet-V2, was pre-trained on the CIFAR-100 dataset. The convolutional layers were used as as a fixed feature extractor, and a fully connected layer was added, which was trained on the CIFAR-10 dataset using the online learning strategy adapted to the hybrid memory circuit constraints (Figure 3.9). Table 3.2 presents the accuracy levels obtained for implementation of the classifier using offline training with floating-point and four-bit integer weights, and those obtained using the proposed learning strategy. In itself, quantifying the classifier weights using four bits does not impact accuracy, and the online transfer learning only reduces accuracy by approximately two percentage points, to 88.0%, confirming that this approach performs well even with sophisticated datasets. Details of the transfer learning training methodology are presented in section 3.4.3.

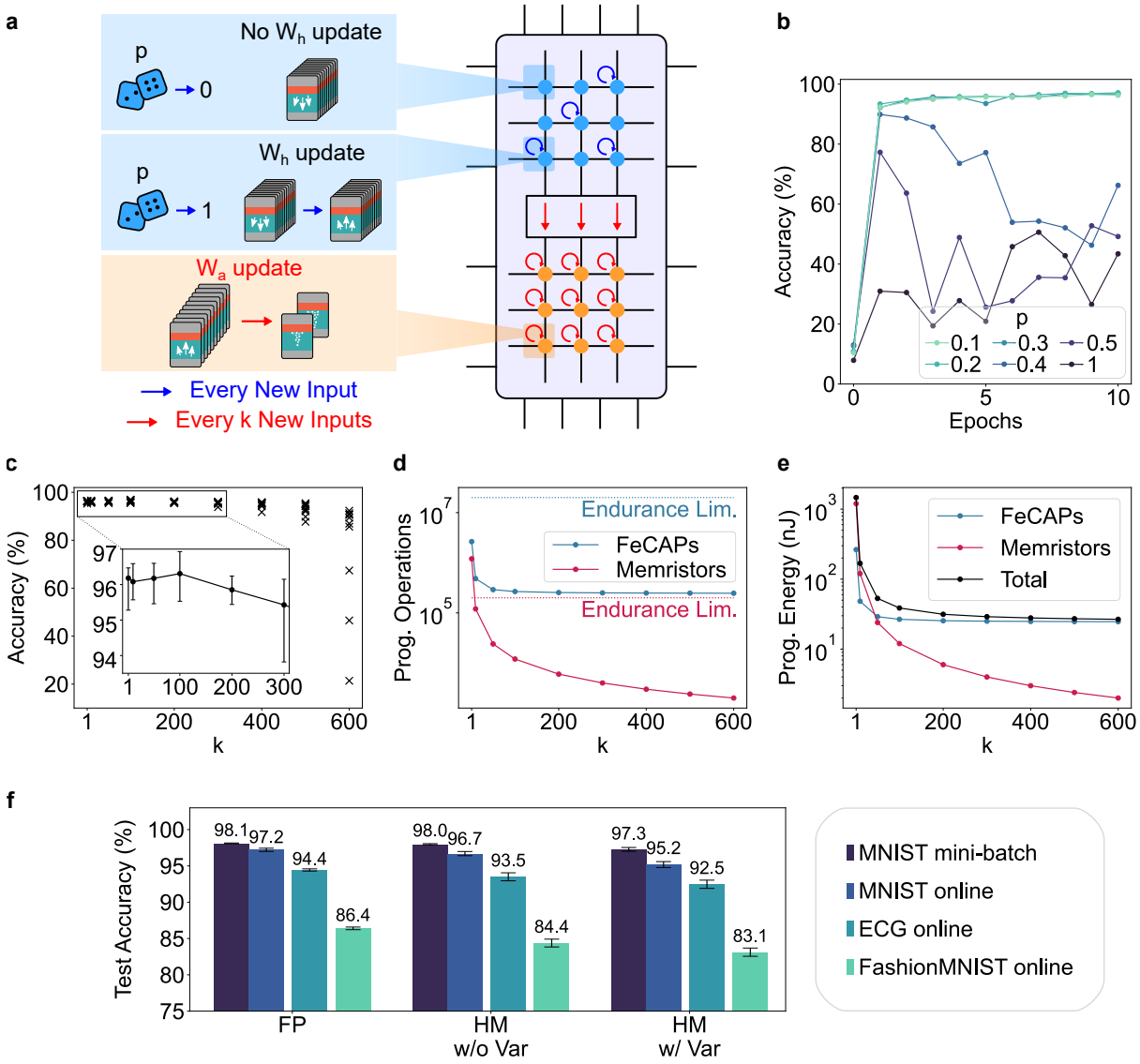

Figure 3.8: **Training at the edge with the improved hybrid FeCAP/memristor arrays.** **a** Representation of the on-chip probabilistic weight update. The FeCAP array used to store the hidden weights is stochastically updated with input of each new image, whereas the memristor array used to store the analog weights is updated after input of every k images. **b** MNIST test classification accuracy as function of the training epochs, for increasing values of update probability. **c** Accuracy after 10 training epochs as a function of the k parameter. The average, minimum and maximum test accuracy (points, error bars in inset) were evaluated for 10 individual training runs. **d** Number of memristor and FeCAP programming operations at the end of one training round as a function of k. **e** Total programming energy for memristors, and FeCAPs at the end of one training round as a function of k. A programming energy of 1 pJ and 100 fJ was assumed for a single memristor and FeCAP programming operation, respectively. **f** MNIST classification, Fashion-MNIST classification and ECG detection accuracies obtained with the classical ANN (FP) and the proposed training strategy based on the hybrid memory (HM) in both the ideal memory case (w/o Var) and with artificially induced errors in the transfer operation due to device variability (w/ Var). For the MNIST dataset, a case where the network was trained with 10-image mini-batches, with batch normalization, is also included for comparison.

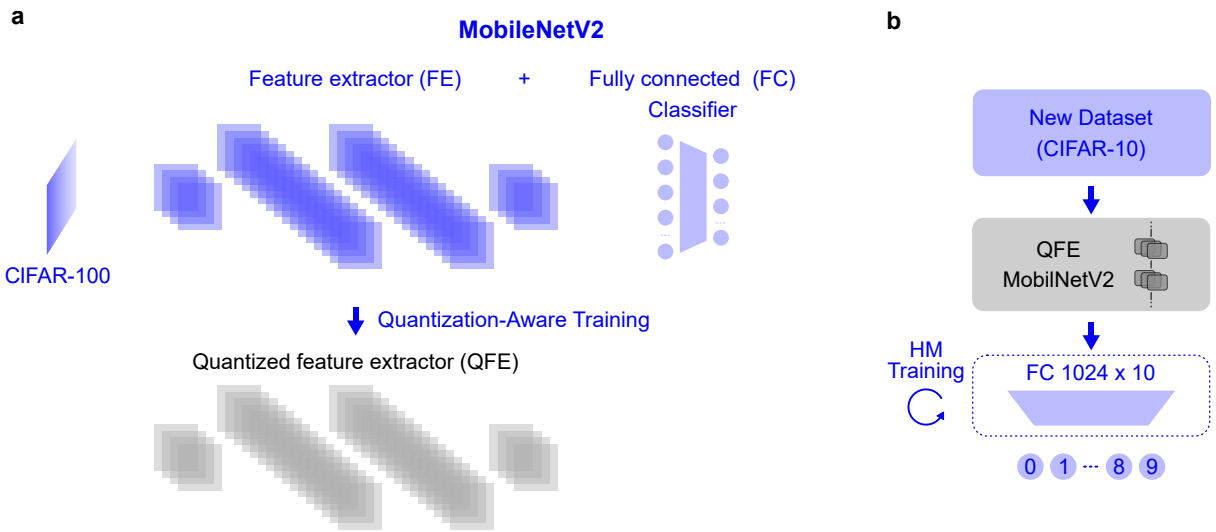

Figure 3.9: **Transfer learning with the improved hybrid FeCAP/memristor memory circuit.** **a** Schematic showing the transfer-learning strategy with initial MobileNet-V2 architecture pre-trained on the CIFAR-100 dataset using a quantization-aware training strategy to obtain a fixed 4-bit integer feature extractor. **b** The MobileNet-V2 4-bit feature extractor was used in conjunction with a newly initialized classifier for transfer learning of the pre-trained model to the CIFAR-10 classification task. The proposed hybrid memory-based training strategy was used to train the classifier.

Feature Extractor		**Classifier**		**Test Accuracy (%)**
W_a **Format**	**Training Method**	W_a **Format**	**Training Method**	
4b / INT	Offline	32b / FP	Offline	89.7 ± 0.1
4b / INT	Offline	4b / INT	Offline	90.0 ± 0.2
4b / INT	**Offline**	**4b / INT**	**HM - online**	**88.0±0.1**

Table 3.2: Comparison of the accuracy performance on the CIFAR-10 classification task for different formats of inference weights and training strategies. Offline refers to standard training via mini-batch with the Adam optimizer, while HM-online refers to the proposed training strategy adapted to the hybrid memory circuit.

3.4.3 Supplementary information

Optimizing FeCAP area ratios for precise transfer line voltage level control. In this note, the optimization of FeCAP area ratios to ensure precise control of transfer line voltage levels is explored. Different FeCAPs are used with varying areas to account for the significance of each MSB. Specifically, the ratio between the areas of two adjacent FeCAPs is set to two. This configuration allows the combined state of the n MSBs stored in these capacitors to generate 2^n distinct voltage levels on the transfer line, V_{TL}. Each combination of MSBs corresponds to a unique voltage level with near-linear spacing between the V_{TL} values, which is essential for programming the memristors. This result can be demonstrated analytically.
For n capacitors connected to the same transfer line, the V_{TL} can be expressed as

$$V_{TL} = \frac{\sum_{i=0}^{n-1} C_i}{\sum_{i=0}^{n-1} C_i + C_{TL}} V_{SL}, \tag{3.3}$$

where C_i is the capacitance of the i-th ferroelectric capacitor, C_{TL} the transfer line parasitic capacitance and V_{SL} the applied read voltage on the source line.
The capacitance of the FeCAPs is the sum of a dielectric and ferroelectric component

$$C_i = C_{D,i} + C_{F,i} = \epsilon_0 \epsilon_r \frac{S_i}{d} + \delta_i \frac{2 P_R S_i}{V_{SL}}, \tag{3.4}$$

where S_i and d are the area and thickness of the i-th FeCAP, ϵ_0 is the vacuum permittivity, ϵ_r the relative permittivity of the ferroelectric material, P_R is the remanent polarization of the capacitor, and δ_i is equal to one or zero depending on the state stored into the ferroelectric capacitor.
Under the approximation:

$$\sum_{i=0}^{n-1} C_{F,i} \ll \sum_{i=0}^{n-1} C_{D,i} + C_{TL}, \tag{3.5}$$

equation (3.3) can be rewritten as:

$$V_{TL} \approx \frac{\sum_{i=0}^{n-1} C_{D,i} + \sum_{i=0}^{n-1} C_{F,i}}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} V_{SL}. \tag{3.6}$$

By combining Equation (3.4) and (3.6)

$$V_{TL} = \frac{\sum_{i=0}^{n-1} C_{D,i} + \sum_{i=0}^{n-1} C_{F,i}}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} V_{SL} = V_{TL,D} + \frac{2 P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} \sum_{i=0}^{n-1} \delta_i S_i. \tag{3.7}$$

The transfer line voltage is the sum of a fixed dielectric component $V_{TL,D}$ and a component dependent on the data stored into the ferroelectric capacitors.
As an example, the case n=2 is analyzed, in which the V_{TL} can assume 4 different values

depending on the stored data:

$$V_{TL,00} = V_{TL,D} \tag{3.8}$$

$$V_{TL,01} = V_{TL,D} + \frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_0 \tag{3.9}$$

$$V_{TL,10} = V_{TL,D} + \frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_1 \tag{3.10}$$

$$V_{TL,11} = V_{TL,D} + \frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} (S_0 + S_1). \tag{3.11}$$

To ensure equal spacing between V_{TL} values corresponding to adjacent bit combinations in the FeCAPs, the condition $V_{TL,01}$-$V_{TL,00}$=$V_{TL,10}$-$V_{TL,01}$ is imposed. Therefore

$$\frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_0 = \frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_1 - \frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_0, \tag{3.12}$$

which simplifies to

$$S_1 = 2S_0. \tag{3.13}$$

By setting S_0=S_{min}, the four V_{TL} levels are

$$V_{TL,00} = V_{TL,D} \tag{3.14}$$

$$V_{TL,01} = V_{TL,D} + \frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_{min} \tag{3.15}$$

$$V_{TL,10} = V_{TL,D} + 2\frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_{min} \tag{3.16}$$

$$V_{TL,11} = V_{TL,D} + 3\frac{2P_r}{\sum_{i=0}^{n-1} C_{D,i} + C_{TL}} S_{min}. \tag{3.17}$$

In conclusion, if the two FeCAPs have areas S_1=$2S_0$ four equally spaced transfer line voltage levels are obtained. This approach easily extends to larger values of n.
The error in the calculated transfer line voltage levels (Eqs. 3.14-3.17) under the Eq. 3.5 approximation depends on the data stored in the capacitors as well as their area. The transfer line voltage levels without the Eq. 3.5 approximation for the case n=2 are:

$$V_{TL,00} = V_{TL,D} \tag{3.18}$$

$$V_{TL,01} = \left[V_{TL,D} + \frac{2P_r}{C_{TL}} S_{min}\right] / \alpha_{01}\,, \quad \alpha_{01} = 1 + \frac{C_{F,0}}{C_{D,0} + C_{D,1} + C_{TL}} \tag{3.19}$$

$$V_{TL,10} = \left[V_{TL,D} + 2\frac{2P_r}{C_{TL}} S_{min}\right] / \alpha_{10}\,, \quad \alpha_{10} = 1 + \frac{C_{F,1}}{C_{D,0} + C_{D,1} + C_{TL}} \tag{3.20}$$

$$V_{TL,11} = \left[V_{TL,D} + 3\frac{2P_r}{C_{TL}} S_{min}\right] / \alpha_{11}\,, \quad \alpha_{11} = 1 + \frac{C_{F,0} + C_{F,1}}{C_{D,0} + C_{D,1} + C_{TL}}. \tag{3.21}$$

Considering S_0=0.36 µm^2 and S_1=$2S_0$=0.72 µm^2, thickness d=10 nm, relative permittivity ϵ_r=29.7, remanent polarization P_r=15 µC/cm^2, applied read voltage V_{SL}=3.5 V and transfer line parasitic capacitance C_{TL}=200 fF, the correction coefficients can be evaluated. For n=2, $\sum_{i=0}^{n-1} C_{D,i} \approx$ 28 fF and $\sum_{i=0}^{n-1} C_{F,i}$ is equal to 0 fF, 31 fF, 62 fF, 93 fF for data 00, 01, 10 and 11 stored in the capacitors, respectively. The corresponding correction coefficients are α_{01}=1.14, α_{10}=1.27, α_{11}=1.41.

Impact of selector transistors on 1T-1C cells with different FeCAPs area. In this note, the impact of selector transistors on 1T-1C cells is investigated for FeCAPs with varying areas. The three MSBs are stored in capacitors of sizes $4S_{min}$, $2S_{min}$, and S_{min} to ensure near-linear spacing between the transfer line voltage values and the different bit combinations, as demonstrated in the previous paragraph. Since each FeCAP bit cell in the fabricated array (Figure 3.6a) is a 1T-1C cell, the MSBs were stored using 1T-1C, 2T-2C, and 4T-4C cells configurations.
SPICE simulations were performed to check if the supplementary selector transistors in the manufactured array affect the transfer line voltages achieved during the transfer operation. Specifically, the circuits in Figure 3.10a and Figure 3.10c were simulated and compared, for several durations of parallel read pulses applied to the 1T-2C/2T-2C and 1T-4C/4T-4C cells. Prior to the parallel read operation, FeCAP devices were programmed to either one or zero by applying 1 µs long pulses. The pulse amplitude for read and write operations was set to 4.8 V. Thick gate oxide transistors with a length and width of 500 nm were used as selector transistors. A fixed transfer line capacitance C_{TL}=200 fF was used. For simulations, a Verilog-A model for the FeCAPs was employed, modeling devices with a remanent polarization of 19 µC/cm^2, a saturation polarization of 20 µC/cm^2, a coercive field of 1.5 MV/cm, a leakage resistance of 1 MΩ, a capacitor thickness of 10 nm, a capacitor surface area of 0.25 µm^2, and a relative permittivity of the ferroelectric material of 30.
The simulation results shown in Figure 3.10b for the comparison of the 1T-2C and 2T-2C cells, as well as the results for the 1T-4C and 4T-4C cells comparison in Figure 3.10d, indicate a negligible difference in the transfer line voltage levels for sufficiently long read pulse durations. For read pulses of 1 µs, the worst-case difference between the transfer line voltage levels achieved with single or multiple access transistors is approximately 20 mV, confirming that the 1T-2C cells and 1T-4C cells can be reliably emulated by the fabricated array with minimal-area 1T-1C cells.

Weight transfer electrical characterization. For the weight transfer characterization in Figure 3.7, the manufactured hybrid memory array was employed to transfer data from seven minimum-sized 1T-1C cells, corresponding to the three MSBs of the hidden weight, to one 1T-1R cell. The first MSB is represented by 4T-4C cells, the second by 2T-2C cells, and the third by 1T-1C cell. This is equivalent to the transfer from three 1T-1C cells with sizes $4S_{min}$, $2S_{min}$, and S_{min} (see previous paragraph).
Before each transfer operation, the memristor device is reset to a low conductance state, and the data to transfer is written into the seven FeCAPs. The transfer line is then precharged to 0 V. During the transfer operation, the source lines (SLs) of the seven FeCAP devices are pulsed with a 3.5 V-10 µs pulse, the WLs are set to 4.8 V, and the transfer line is floating. At the same time, a 2.3 V-10 µs pulse is applied to the memristor source line. The memristor BL is set to −0.9 V. This negative voltage setting for the memristor was necessary to obtain a gate-source voltage (V_{gs}) suitable for the RRAM's selector transistor during the transfer operation. Nevertheless, accurate co-design of the BL parasitic capacitance, material stack, and device geometry could eliminate the need to set any bias voltage on the memristor's bottom electrode, simplifying the transfer procedure. Finally, the memristor's conductance is externally read with 400 mV voltage pulses.

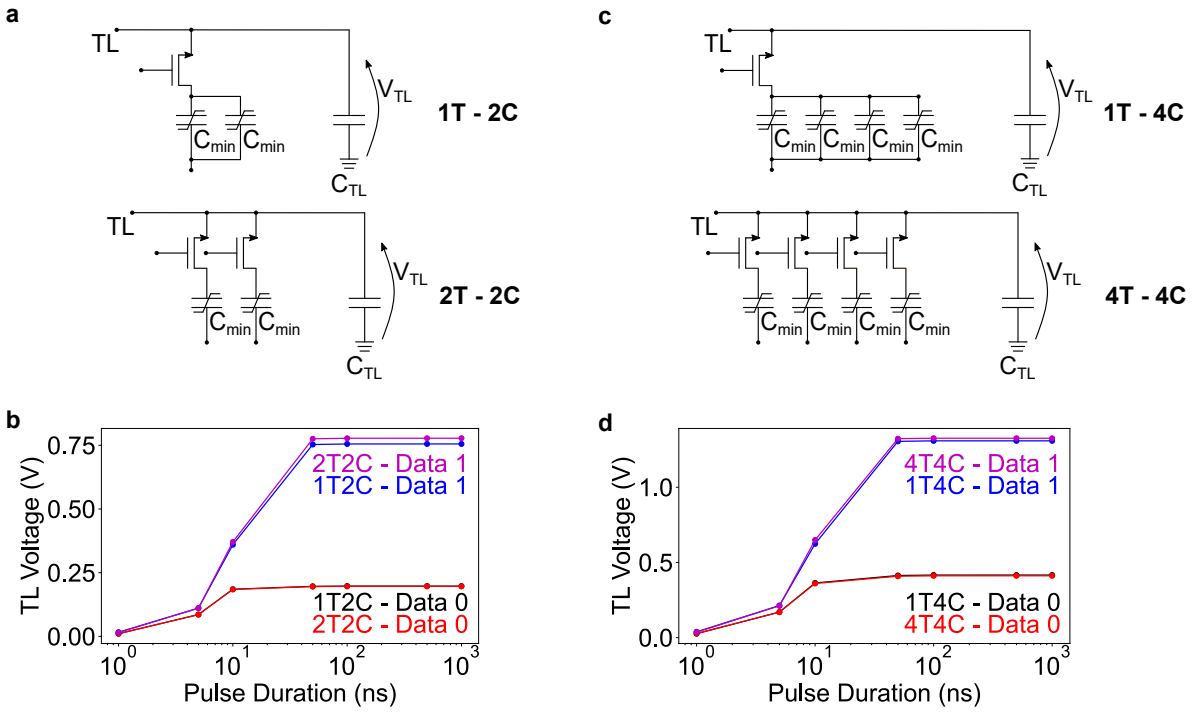

Figure 3.10: **Comparison of 1T-2C vs 2T-2C cells and 1T-4C vs 4T-4C cells. a** Schematic of 1T-2C and 2T-2C cells connected to transfer lines with parasitic capacitance C_{TL}. **b** Electrical simulations of the transfer line voltage as function of the parallel read pulse duration for capacitors programmed to either zero (Data 0) or one (Data 1) for the 1T-2C/2T-2C cells. **c** Schematic of 1T-4C and 4T-4C cells connected to transfer lines with parasitic capacitance C_{TL}. **b** Electrical simulations of the transfer line voltage as a function of the parallel read pulse duration for capacitors programmed to either zero (Data 0) or one (Data 1) for the 1T-4C/4T-4C cells.

Justification for the probabilistic weight update. Learning with integer quantized hidden weights is challenging for convergence. The main reason is the fact that the quantized space of parameters might not allow to capture weight updates that are small in magnitude. For this reason larger learning rates need to be used to trigger the updates. Nevertheless, excessively large learning rates might cause weight updates to be overly large, resulting in the exploding gradients problem.
The stochastic weight update introduced in the context of the hybrid memory training provides an effective way to reduce, in average, the learning rate and ensure convergence. Indeed, each hidden weight between neuron i and neuron j in a network layer is updated with the following rule:

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} - \eta^{(t)} \Delta w_{ij}^{(t)}(p) = w_{ij}^{(t)} - \eta^{(t)} \Delta w_{ij}^{(t)} X(p). \tag{3.22}$$

where $\eta^{(t)}$ is the learning rate at iteration t and X(p) is a random variable distributed according to a Bernoulli distribution with probability p. The weight update is therefore a random variable with expected value:

$$\mathbb{E}[\Delta w_{ij}^{(t)}(p)] = \mathbb{E}[\Delta w_{ij}^{(t)} X(p)] = \Delta w_{ij}^{(t)} \mathbb{E}[X(p)] = p \Delta w_{ij}^{(t)}. \tag{3.23}$$

The variance of the weight update can be analogously evaluated:

$$Var[\Delta w_{ij}^{(t)}(p)] = Var[\Delta w_{ij}^{(t)} X(p)] = \Delta w_{ij}^{(t)} Var[X(p)] = p(1-p) \Delta w_{ij}^{(t)}. \tag{3.24}$$

Therefore, the stochastic update implements an average scaling of the weight update and improves the convergence of training algorithm. Moreover, the stochastic nature of the update introduces noise in the weight updates, which can be seen as a form of regularization, thus improving overfitting issues.

Neural networks simulations. A three-layer fully connected neural network was trained to address the MNIST dataset [21]. 60,000 images were used for each training epoch, and the MNIST test set of 10,000 images was used for validation. The network has one input layer with 784 neurons, two hidden layers with 200 and 100 neurons each, and one output layer with 10 neurons. The inputs are gray-scale images with values from 0 to 255, normalized between -1 and 1. For the online training simulations (details in Appendix C, Algorithms 1-4), each hidden layer undergoes layer normalization [29] (where scale and shift trainable parameters γ and β are set to one and zero, respectively), followed by ReLU activation. The activation function of the output layer is a softmax function, preceded by a layer normalization. The ANN with hybrid memories was trained for 10 epochs, with a learning rate of 0.1. The calculated hidden weight updates matrix is multiplied by a binary mask M that dictates whether a hidden weight should be updated or not. Each element of the mask is independently drawn for each example during training, following a Bernoulli distribution with a given probability of 20%. For full-precision floating-point hidden weight simulations, the same update strategy and hyperparameter values were used. For the mini-batch training simulations (details in Appendix C, Algorithms 2-5) the normalization layer was replaced by a batch normalization layer (scale and shift trainable parameters γ and β set to one and zero respectively). The same hyperparameters are used, as in the online training case. The mini-batch size is 10 and the number of training epochs is increased to 100.

A four-layer fully connected neural network architecture was used for the Fashion-MNIST dataset[22] with 784-200-100-100-10 neurons. As in the previous case, the inputs are gray-scale images with values from 0 to 255, normalized between -1 and 1. Again, 60,000 images were used for each training epoch, and the test set of 10,000 images was used for validation. The network was trained for 10 epochs at a learning rate of 0.1 and a hidden weight update probability of 10%. The same hyperparameters were used for the full-precision floating-point hidden weight simulations.

For the ECG binary detection task, a fully connected neural network with one hidden layer of 200 neurons was used. No normalization layers were added. A ReLU activation function was used for the hidden layer, and a sigmoid activation function was used for the output neuron. Data from the MIT-BIH heart arrhythmia database[184] were used for this study. The dataset consists of half-hour dual-channel electrocardiogram recordings from 48 subjects. Individual heartbeats were extracted from each recording, forming 700 ms time series centered on the R-wave peak. Using the FFT function from NumPy, a frequency spectrum was generated for each time series. The ten lowest frequency components were selected from each channel, resulting in 20 features characterizing each heartbeat as a static data point. Heartbeats were categorized as either normal and healthy ("N" label) or exhibiting signs of arrhythmia ("L", "R", "e", "j", "A", "a", "J", "S", "V", "E", "F", "/", "f", and "Q" labels). A subset of 20,000 data points (10,000 healthy and 10,000 unhealthy heartbeats) was randomly sampled from 47 subjects for model training. Subsequently, models were tested using data points from a previously unseen subject (patient 208). The network was trained for 10 epochs, with a learning rate of 0.3 and an update probability of 20%. For the floating-point hidden weight simulations, the same hyperparameters were used.

The parameters in the hybrid memory circuit consist of FeCAP and memristor devices, which are not as precise as their 32-bit floating-point counterparts. Therefore, to emulate the training when using the hybrid memory operation, the representation of the hidden and analog weights is set to 10-bit and 4-bit integers, respectively, to match the proposed hybrid memory circuit. The hidden weights are stored in ten binary FeCAP cells, and the analog weights are stored in two memristor cells with eight conductance values each. During inference, the 10-bit values are truncated to their 4-bit approximations (Algorithm 3, Appendix C,), thereby mimicking the operation applied during the proposed digital-to-analog transfer procedure in the unified memory. Furthermore, memristors and FeCAPs suffer from other types of non-idealities such as device-to-device and cycle-to-cycle variability. The method by which device non-idealities are taken into account and their impact on the transfer operation is described in the following paragraph and in Appendix C, Algorithm 4.

Hardware-aware neural networks simulations. The statistical electrical characterization of the transfer operation on the hybrid memory circuit was used to account for the device non-idealities in the hardware-aware neural network simulations. The 10-bits hidden weights (W_h) were approximated to their 4-bit equivalent analog version (W_a) each time a transfer operation was performed. For each analog weight, a random number was sampled from a normal distribution with average value μ_W equal to the corresponding W_a value and standard deviation σ_W. The profile of the analog weight is based on statistical characterization of the hybrid memory circuit (Figure 3.7). The detailed simulated

procedure is described below.
The conductance of the memristor device in the differential pair undergoing the analog transfer was modeled as a random variable with normal probability distribution:

$$G = \sigma_G X + \mu_G \tag{3.25}$$

where σ_G is the conductance standard deviation, μ_G the average conductance of a transferred level and X a standard normal random variable. Moreover, a linear relationship was considered between the mapped conductance and the weight value:

$$G = \alpha W_a + \beta = \frac{\mu_{G,max} - \mu_{G,min}}{W_{a,max} - W_{a,min}} W_a + \mu_{G,min} \tag{3.26}$$

where $\mu_{G,max}$ and $\mu_{G,min}$ are the average conductance values of the corresponding average maximum ($W_{a,max}$) and minimum ($W_{a,min}$) transferred weights in absolute value respectively. By equating the first member of equations (3.26) and (3.25):

$$\sigma_G X + \mu_G = \alpha W_a + \beta \ . \tag{3.27}$$

For three MSBs, $W_{a,max} = 7$ and $W_{a,min} = 0$. Therefore,

$$W_a = \sigma_W X + \mu_W = \frac{\sigma_G}{\alpha} X + \frac{\mu_G - \beta}{\alpha} = \frac{7\sigma_G}{\mu_{G,max} - \mu_{G,min}} X + \frac{7(\mu_G - \mu_{G,min})}{\mu_{G,max} - \mu_{G,min}} \ . \tag{3.28}$$

Therefore the analog weight values are sampled from a normal distribution with standard deviation $7\sigma_G/(\mu_{G,max} - \mu_{G,min})$ and average value $7(\mu_G - \mu_{G,min})/(\mu_{G,max} - \mu_{G,min})$. The average conductance for each transferred value were obtained experimentally from the characterization in Figure 3.7. In particular the average conductance values for the 8 levels are $\mu_{G,min}$=$\mu_{G,0}$=1.4 µS, $\mu_{G,1}$=11.9 µS, $\mu_{G,2}$=36.5 µS, $\mu_{G,3}$=58.3 µS, $\mu_{G,4}$=73.9 µS, $\mu_{G,5}$=85.5 µS, $\mu_{G,6}$=95.9 µS, $\mu_{G,max}$=$\mu_{G,7}$=103.9 µS. For each level, a constant standard deviation equal to the average of the standard deviations of the 8 levels was considered, i.e. σ_G=8.7 µS. Consequently, the σ_W parameter is

$$\sigma_W = \frac{7\sigma_G}{\mu_{G,max} - \mu_{G,min}} \approx 0.6$$

Since two separate memristors were used to encode either positive or negative weights, no sign error could occur during the transfer operation. This approach was also implemented in the hardware-aware simulations. The same hyperparameters mentioned above for each dataset were used for the hardware-aware simulations.

Number of memristor and FeCAP programming operations and programming energy evaluation. To estimate the number of programming operations and corresponding programming energy in FeCAPs and memristors during training, as reported in Figures 3.8d and 3.8e in the Results section, the following procedure was used. The number of updates for each hidden weight were counted, taking into account the probabilistic weight update scheme. The number of operations on FeCAPs was defined as twice the maximum number of hidden weight updates because each time the weight is read, it has to be rewritten. Indeed, the encoding of the hidden weights in a sign and magnitude configuration does not allow to perform blind updates, as was possible in the

initial implementation of the hybrid memory circuit described in section 3.3.2. Moreover, for each transfer operation, each FeCAP incurs four additional programming operations: reading and rewriting data to save it in another FeCAP cell for further reloading, and then actually transferring (reading) and reloading (writing) the weight.
For the memristors, the number of operations is defined by the parameter k, i.e., the number of inputs after which the weights are transferred from FeCAPs to memristors. For each transfer operation, each memristor requires two programming operations: one for resetting the device and one to write the analog weight.
The programming energy consumption of the two arrays was evaluated by multiplying the number of programming operations by the estimated switching energy for FeCAPs and memristors. The energy required for FeCAP and memristor programming operations was set to 100 fJ and 1 pJ[85, 86] respectively.

CIFAR-10 transfer-learning simulations. The MobileNet-V2 neural network architecture [189] was trained on the CIFAR-100 dataset[23] to obtain a weight quantized model (4b/INT) using a learned step size quantization method[190]. The images contained in the CIFAR-100 dataset were resized to 128×128 pixels before being fed to the network. The hyperparameters used during this step were: batch size 128, 500 epochs, Adam optimizer, learning rate 10^{-4}, and weight decay 10^{-5}. The full-precision model achieved a top-1 test accuracy of 79.5% on the CIFAR-100 dataset, whereas the quantized model achieved a top-1 test accuracy of 77.4%.
The MobileNet-V2 architecture 4-bit feature extractor was used in connection with a newly initialized classifier for transfer learning from the pre-trained model for the CIFAR-10 classification task. During this step, the feature extractor was fixed, and only the classifier parameters were updated. For the offline transfer-learning simulations (Offline in Table 3.2), a standard optimizer (Adam), batch size 128, learning rate 10^{-4}, and learned step size quantization of the inference weights (4b/INT) were used. For the online transfer learning with the hybrid memory training strategy developed (HM-online in Table 3.2), the hidden weights were updated for each training sample with an update probability of 1. The inference weights were updated after 256 new inputs, the learning rate was set to 10^{-2} with a learning rate decay of 0.1. The classifier was trained for 50 epochs. For the transfer learning experiments, the images of the CIFAR-10 dataset were resized to 128×128 pixels before being fed to the network.
The number of updates for each hidden weight at the end of training was $2.5 \cdot 10^6$, and the number of transfer operations from hidden to analog weights was $9.8 \cdot 10^3$. Therefore, the maximum number of programming operations was $5 \cdot 10^6$ for FeCAPs and $2 \cdot 10^4$ for memristors. Thus, the online transfer-learning strategy is compatible with the endurance constraints of both ferroelectric and resistive devices.

3.4.4 Discussion

The proposed approach leverages the high programming endurance/low programming energy of FeCAPs and the near-infinite read endurance and analog IMC compatibility of memristors. This dual memory approach addresses the limitations associated with using a single memory technology. The data-disruptive nature of reading from FeCAP devices makes them inefficient for inference once the training has been completed. In

addition, it causes the read endurance to be limited by the write endurance and therefore restricts the useful lifetime of trained devices. In contrast, a full memristor implementation would benefit from the read stability of conductance states and energy efficient in-memory computing, allowing lifelong inference. However, the limited write endurance and considerable programming energy of memristors are incompatible with meaningful applications for training. Even with a small dataset like MNIST, the number of programming operations – 1.2×10^6 write operations (k=1 case in Figures 3.8d, e) – exceeds the endurance limit of the memristor devices in the proposed stack. In the following of this discussion, outlooks in terms of device optimization, scalability enhancement and the implementation of a full system based on the proposed technology are examined.

Device optimization. Optimizing the unified memory stack to function as both ferroelectric memory and memristor was essential for the success of this approach. The thickness and oxygen-vacancy profile of the Si-doped HfO_2 layer are crucial for both technologies and must be optimized by adjusting deposition parameters and through interface design. To ensure crystallization of the film in the ferroelectric orthorhombic phase while maintaining an annealing temperature compatible with the BEOL fabrication process, the Si-doped HfO_2 cannot be scaled to less than 10 nm [150]. However, to reduce the forming voltage, memristor devices typically have a thinner optimal thickness. For this reason, a thickness of exactly 10 nm was adopted. An optimized oxygen-vacancy concentration can enhance ferroelectricity by stabilizing the orthorhombic phase and at the same time reduce the forming voltage, but vacancies must be carefully managed to avoid promoting non-ferroelectric phases. To increase memory density, devices will undergo lateral scaling when integrated into an industrial process. This lateral scaling is expected to reduce the number of interface defects and provide higher endurance for the FeCAPs [176]. Additionally, switching to more advanced materials, such as hafnium zirconium oxide (HZO), could further enhance the endurance of FeCAPs without compromising the quality of the memristor response.

Enhancing scalability with 3-D memory structures. A significant cost of the optimized hybrid memory circuit is the need for FeCAPs with distinct areas or multiple capacitors per bit to store the MSBs of the hidden weights. This solution allows for the transfer and conversion of digital data, stored in a sign and magnitude configuration, into multi-level analog conductance values without requiring dedicated digital-to-analog converters. The number of MSBs requiring larger capacitors is determined by the maximum number of analog conductance levels that can be reliably programmed in memristor devices. The presented technology can store a maximum of eight distinct analog levels in memristors, corresponding to three bits, or four bits per weight when including the sign. This represents a good compromise: four-bit is a classic weight resolution in highly quantized neural networks, providing high accuracy even on advanced AI tasks [191]. To enhance scalability, high aspect-ratio 3-D FeCAPs could be a viable solution [97, 149, 192]. A unified FeCAP bitcell can be proposed, integrating up to four 3-D ferroelectric cylindrical capacitor structures on the same selector transistor to achieve different equivalent areas without increasing the bitcell's current area. Figure 3.11a illustrates the main process steps for fabricating 3-D capacitors. The critical steps include the cylinder etching and the conformal deposition of both the electrodes and the ferroelectric layer.

3-D ferroelectric capacitors have been demonstrated at the 130 nm technology node, with an aspect ratio of three, indicating the ratio between the cylinder height and its diameter [149]. In the following analysis, an aspect ratio of three is imposed. This aspect ratio is quite conservative: Larger aspect ratios, ranging between 15 and 20, have been achieved at more advanced technology nodes, suggesting the possibility of further increasing the integration density [97]. For an aspect ratio of three, to achieve an area equivalent to the one in the measured array in Figure 3.6d, a cylindrical capacitor with a radius of 190 nm and a height of 570 nm should be fabricated, as shown in Figure 3.11b. Neglecting the flange contribution to the area of the cylindrical capacitor, capacitors with area equal to four and two times that of the minimum-sized one can be obtained by connecting multiple minimum-area capacitors in parallel in the BEOL. As shown in Figure 3.11b, four capacitors fit in the same space occupied by a planar minimum area capacitor structure, with a spacing between the 3-D structures of 220 nm. Figure 3.11c shows the side view of the capacitors to be fabricated over the same access transistor, sharing the same top and bottom electrode for parallel connection of the minimum surface 3-D structures.

Towards a full system implementation. The manufactured test scribe of hybrid synapses allowed to validate and emulate the transfer procedure from different sized capacitors to a memristor device. Nevertheless, a complete system able to learn on-chip would require the integration of further computing blocks. For the implementation of an in-memory computing inference engine, the arrangement of the memristors array lines should be appropriately designed to allow the matrix-vector multiplication implementation via Ohm's and Kirchhoff's laws. The array dimensions should be properly sized taking into account the IR drops on the accumulation lines for the given memristor conductance dynamic. The memristor array periphery should be re-configurable in order to perform the matrix-vector multiplication also during the backward step. A complete system would require the implementation of DACs to provide the inputs to the array in an analog format and ADCs for the current readout of the forward and backward accumulation lines. Finally, a digital processing unit should be designed in order to evaluate the weight gradients to add to the weights stored in the FeCAPs. For the proposed approach, one or multiple pseudo random number generators should be designed for the probabilistic weight update. Thanks to the designed hybrid synaptic circuit, there would be no need of a DAC for the weight programming of the RRAM array, as this would be implemented thanks to the proposed transfer procedure.
The impact of device variability during the transfer operation has to be taken into account when designing a complete reliable system. To further investigate this aspect, the test accuracy on the MNIST dataset was evaluated for different number of MSBs, stored in FeCAPs in the optimized hybrid synaptic circuit, to be transferred to the analog weights, stored in memristors (Figure 3.12). This analysis compared the transfer of three, two and one MSBs, plus the sign bit, to 15 (MLC 15), 7 (MLC 7) and 3 (MLC 3) differential conductance values of memristors, respectively. The total number of bits for the hidden weights was kept equal to ten. It should be noted that data transfer was actually measured only in the case of transferring three MSBs. The simulations utilizing two MSBs and one MSB considered the conductance states corresponding to the first two MSBs and first MSB, respectively, in Figure 3.7. The analysis was performed for an online and 10-images mini-batch learning strategy, using the same training configurations and hyperparameters described in section 3.4.3. The same conclusions can be extracted for both the online and

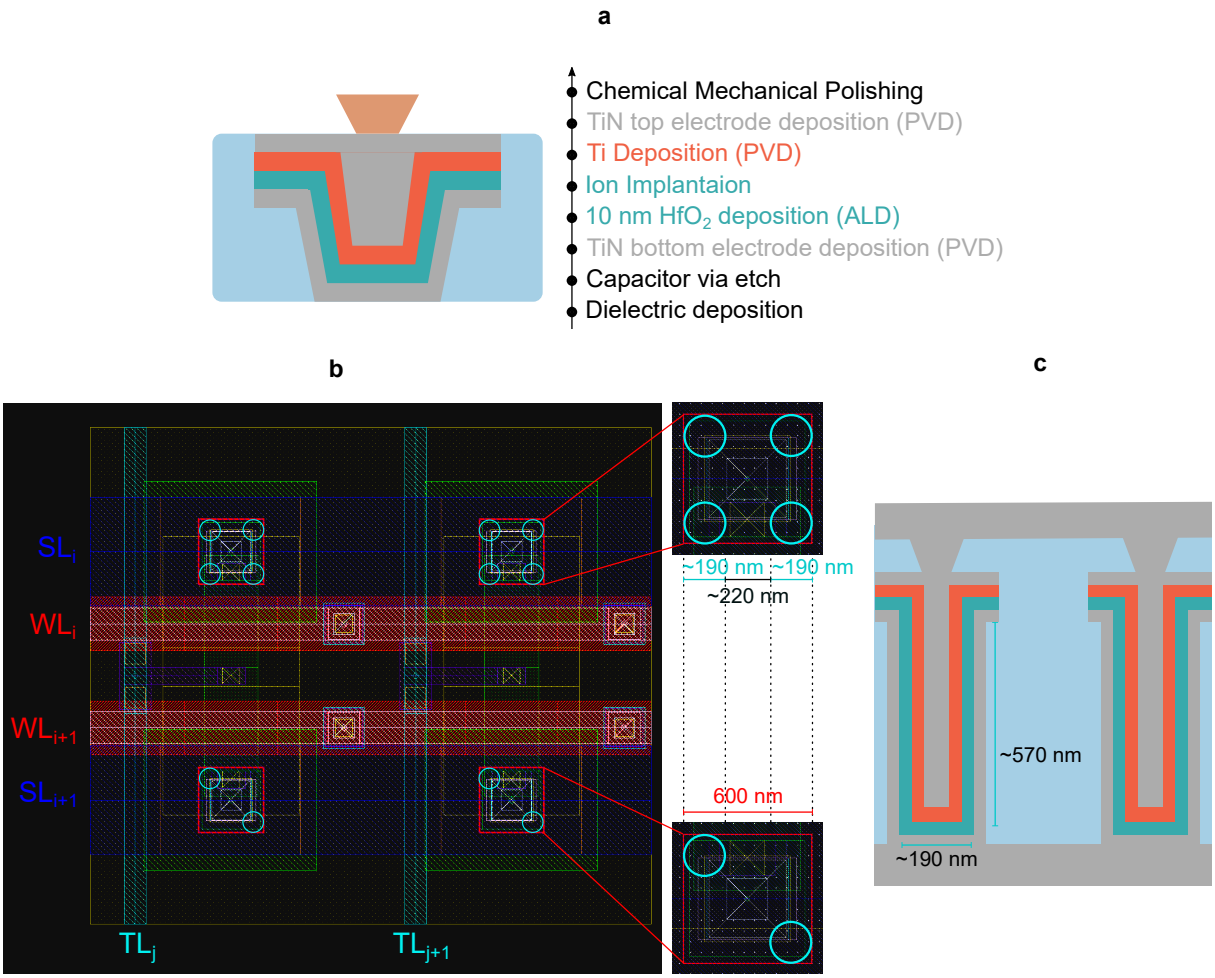

Figure 3.11: **Unified memory bitcell with 3-D capacitors.** **a** Process flow highlighting the key steps in the fabrication of 3-D ferroelectric cylindrical capacitors. **b** Bitcell layout with equal 0.36 µm^2 planar capacitors (red squares) and potential parallel-connected 3-D capacitors (teal circles) with cylinder diameter of 190 nm and aspect ratio of three, for an open cylinder area of 0.37 µm^2. Either four or two 3-D capacitors can be connected in parallel to achieve larger effective surfaces to store the MSBs of the proposed hybrid memory circuit. **c** Schematic side view of the cylindrical capacitors fabricated over the same selector transistor. The top and bottom electrodes are shared to connect either four or two capacitors in parallel.

mini-batch approaches. Without considering device variability, increasing the number of MSBs from one to three in the hybrid memory circuit increases the accuracy, even though no significant difference can be measured between the accuracy results for two and three MSBs. If device variability is considered, training with one, two or three MSBs results in approximately equivalent accuracy. This analysis highlights the fact that until the reliability of memristive devices is not improved, their use for storing multiple levels per cell might not provide a significant overall advantage compared to binary devices. This is true for the analyzed use case, but it is reasonable to assume that similar considerations apply to more complex tasks. Beyond weight quantization, the impact of activations and gradients quantization should be eventually considered for further development.

Finally, this technology also holds potential for other forms of on-chip learning, par-

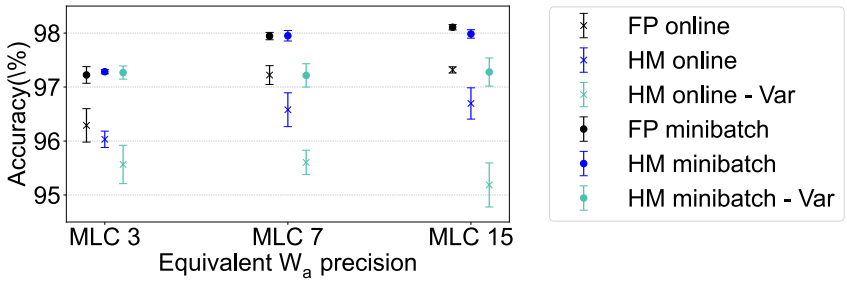

Figure 3.12: **Comparative study of the hybrid memory training for different equivalent precision of the analog weights.** The average test accuracy and standard deviation were evaluated for 10 individual training runs.

ticularly continuous, adaptive learning, which was not explored here. Hidden weights in binarized and quantized neural networks have been shown to function as metaplastic variables, and modifying their training techniques can mitigate forgetting [51, 193]. The proposed hybrid memory circuit is the first to harness the unique properties of two memory technologies for non-volatile storage of both hidden and analog weights. It could naturally enable continuous learning and offer robust solutions for adaptive and lifelong learning applications.

3.5 Summary

This chapter experimentally demonstrated an application-specific integrated circuit leveraging the hybrid memory technology introduced in chapter 2, hence including both FeCAPs and memristors within the same BEOL of foundry 130 nm CMOS, in order to create an hybrid array of interconnected FeCAPs and memristors.

The fundamental unit of the hybrid array is an hybrid synapse circuit, consisting of a collection of one-transistor-one-FeCAP cells, where the bit line of each cell is directly connected to the gate of the selection transistors in a one-transistor-one-memristor cell. This sub-circuit enables direct digital-to-analog data transfer from multiple FeCAP cells to a single memristive device, without the need for intermediate circuitry.

This technology is compatible with on-chip training of ANNs. FeCAP devices store higher-precision hidden weights that undergo numerous programming operations during training, while memristor devices store analog weights read during both inference and training.

Two system implementations were proposed. The first uses equally sized FeCAPs to store

the higher-precision weights and a single memristor to store the analog weight. This approach limits the precision of the hidden weights to n+1 discrete states, where n is the number of FeCAPs in the synaptic circuit. A binarized neural network was trained on the ECG-arrhythmia detection task. Taking into account the constraints of the synaptic circuit, the network achieves 88% accuracy, with eight FeCAPs only storing each higher-precision weight and one memristor used to store the binary weight. The second implementation uses ferroelectric capacitors with different areas to encode the 10-bit integer higher-precision weights in an sign-and-magnitude format and two memristors to encode positive and negative analog weights in the memristors' differential conductance. Fully-connected ANNs were trained using a stochastic gradient descent algorithm. Based on measurements of analog transfer from FeCAPs to memristors, the results obtained across a variety of edge benchmarks are competitive with those achieved by floating-point precision software models, without the endurance limitations associated with hardware constraints.

Acknowledgments

For the developments presented in this chapter, I would like to thank those who provided and developed the original design of ferroelectric memory arrays, which served as the foundation for the hybrid memory array design. I would also like to acknowledge Olivier Guille for his contribution to the characterization of the hybrid memory circuit. Finally, I extend my gratitude to Yannick Malot for developing the transfer learning simulations tailored to the optimized implementation of the hybrid memory circuit.

Chapter 4

FeЯNet: Ferroelectric and resistive memory circuits for near-memory inference and training

The hybrid memory circuit presented in the previous chapter showcases the potential of combining ferroelectric and resistive memories for implementing ANN learning on-chip. The key idea of the hybrid memory circuit, in its optimized implementation, relies on transferring information from weights stored in a signed integer format in ferroelectric devices to differential conductance levels stored in memristive, or resistive, devices without the need of any digital to analog converter. In this approach, ferroelectric memories are exclusively used to accumulate weight gradients during training and resistive memories are used for forward and backward passes, and periodically updated during training. Potentially, the matrix-vector multiplication performed on memristive devices is implemented in an in-memory computing fashion for improved energy-efficiency. This approach was benchmarked on several tasks with promising results.
Nevertheless, several considerations have to be taken into account for further developments:

- Despite the theoretical advantages of in-memory computing systems, the practical realization of these systems faces several technical hurdles. A significant challenge is the high variability of memristors. Variability affects the precise resistance states of the memristors, which are critical for analog computation, leading to inaccuracies in the MVM operations, impacting the final system accuracy [194]. In addition to device variability, the imperfections of analog CMOS circuits used in the peripheral circuitry can reduce the overall system performance. Voltage drop effects also play a significant role in degrading the performance of memristor-based in-memory computing. In large crossbar arrays, the voltage applied to the memristors is not uniformly distributed across the array due to resistive paths in the interconnects. This voltage drop results in reduced computation accuracy, particularly for deep neural networks where precise MVM operations are essential for maintaining model accuracy. To mitigate these issues, smaller crossbar arrays are used and complex peripheral circuits are designed to enhance the overall reliability and accuracy of the computation [195]. These peripheral circuits include ADCs, DACs, and sense

amplifiers, which facilitate the interface between the analog memristor array and digital control logic, but at the same time limit the energy-efficiency potential of a full-analog system.

- The hybrid memory circuit architecture limits the representation of the hidden weights stored in FeCAPs to integer values only. Fundamentally, the hybrid memory circuit imposes that the equivalent-representation of the weights stored in the memristors' conductance levels has to be the same of the representation of the weights stored in FeCAPs in a digital format. Although training with integer hidden weights might be appealing for improved energy-efficiency, its scalability to more complex tasks is not at all certain. It could be envisioned to store the hidden weights in FeCAPs in a floating-point – sign, exponent and mantissa – format and transferring the sign only, for the implementation of binarized neural networks. Nevertheless, transferring information from one FeCAP only to a single memristor, with the developed analog procedure, would be less advantageous than reading the content of the FeCAP digitally and subsequently programming the memristors, because of the need for storing and reloading of data to be transferred. On the other hand, using the proposed analog procedure for transferring information from one FeCAP to two memristors in a differential configuration, as in the case of the optimized hybrid memory circuit, would be equivalent to perform a digital read of the FeCAP content and subsequently programming the memristors accordingly.

Based on these considerations, this chapter explores a second circuit design aimed at investigating the advantages of combining ferroelectric (Fe) and resistive (Re) memory devices for the hardware implementation of inference and training in binarized neural networks (Net), referred to as FeЯNet (Figure 4.1). In the following sections, the concept of binarized neural networks is presented in more detail, highlighting the key operations performed during training and inference to explain some design choices and considerations. Next, the designed circuit is introduced, along with electrical simulations validating some of its components. Then, the development of the electrical characterization setup is discussed, followed by preliminary measurements confirming the communication between the designed circuit and the test support. Finally, some potential use cases are explored as possible directions for future research.

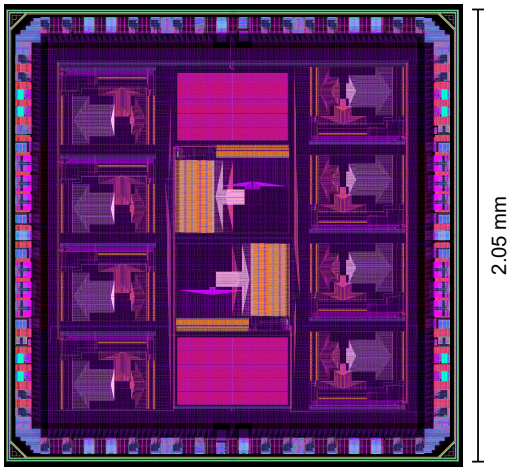

Figure 4.1: **Layout view of the FeЯNet chip.**

4.1 Binarized neural networks

In traditional neural networks, weights and activations use continuous values, requiring significant memory and computational power. By constraining these values to +1 or -1, during forward propagation, binarized neural networks drastically reduce the computational complexity, allowing for faster and more efficient implementations, particularly on hardware with limited resources. To address the challenge of training with binary weights and activations, specialized techniques for both forward and backward propagation are introduced [39, 40, 41].
Although the weights in BNNs are constrained to binary values during inference, training requires a full-precision version of the model's parameters, often referred to as hidden weights. The binarization of hidden weights and activations is achieved using the sign function. Hence, real-valued variables (hidden weights and activations) are converted into their binary counterparts by applying:

$$x_b = \begin{cases} 1 & x_h \geq 0 \\ -1 & x_h < 0 \end{cases} \tag{4.1}$$

where x_b is the binarized variable and x_h the real-valued one.
Thus, in the forward pass for training BNNs, both weights and activations are constrained to binary values, +1 or -1. This binarization reduces the computational complexity by replacing traditional multiplications with efficient XNOR and bit-count operations.
One of the challenges with training binarized networks arises in the backward pass, since the sign function is not differentiable. To enable gradient flow, a straight-through estimator is introduced. The STE approximates the gradient of the non-differentiable sign function by effectively passing the gradient unchanged, treating the gradient as if it came from real-valued activations or weights. The STE technique proposed for BNNs makes the assumption

$$g_h = g_b 1_{|h<1|}, \tag{4.2}$$

where g_h and g_b represent the weight (activation) gradient of the loss function with respect to the hidden weight (real-valued activation) and binarized weight (binarized activation) respectively, and $1_{|h<1|}$ cancels the gradient value if the hidden weight is outside the ± 1 range, for improved performance. Therefore, STE technique effectively allows to evaluate the hidden weight updates from the gradients evaluated on the binary weights.
After the backward pass, the hidden weights are updated based on the computed gradients. The weight update follows the standard rule used in stochastic gradient descent, such as:

$$W_h^{(t+1)} = W_h^{(t)} - \eta^{(t)} g_{W_b}^{(t)} \tag{4.3}$$

where $\eta^{(t)}$ is the learning rate at iteration t. After updating the hidden weights, the binarized weights are evaluated using the sign function for the next forward pass.
To stabilize training, batch normalization is typically applied after each binarized convolution or fully connected layer. This technique helps normalize the activations, which would otherwise be highly irregular due to binarization. By normalizing the intermediate outputs, batch normalization reduces the risk of gradient explosion or vanishing, improving the convergence and generalization of the network. Nevertheless, as previously mentioned, batch normalization has a significant memory and computing overhead. Alternative solutions should be evaluated for more efficient data processing in edge constrained devices.

The FeЯNet circuit takes advantage of RRAM arrays to establish a near-memory computing system, aimed at enhancing the speed and energy-efficiency of both the forward and backward passes during training. In particular the XNOR operation evaluating the multiplication between 1-bit activations and 1-bit weights is performed within the RRAM array sensing circuitry. This operation takes advantage of the virtually unlimited read endurance of RRAM devices. At the same time, the FeRAM array stores the higher-precision hidden weights optimized during training. By exploiting training techniques similar to the ones developed in the previous chapter, the hidden weights in the FeRAM arrays could be updated for each presented training sample, while updating the binarized weights in the RRAM array only periodically. This procedure can effectively reduce the number of programming operation performed on RRAMs and take advantage of the larger programming endurance and lower programming energy of ferroelectric devices.

4.2 FeЯNet design

Two versions of the FeЯNet system were designed, sharing the same core architecture, but different Input/Output (I/O) interfaces, for processing with two different manufacturing flows. The FeЯNet core is either connected to an I/O pad ring, if the complete manufacturing flow is processed, or to two 25-pads test scribes, if only the partial manufacturing flow is processed. Either manufacturing flows consist of a hybrid process, based on a 22 nm fully-depleted silicon on insulator (FDSOI) CMOS foundry technology. FDSOI is a planar CMOS technology that relies on the deposition of a very thin silicon film, which forms the transistor channel, on top of an ultra-thin layer of insulator, called the buried oxide, positioned above the base silicon. FDSOI is designed to offer significantly improved transistor electrostatic properties compared to traditional bulk technology. The buried oxide layer reduces parasitic capacitance between the source and drain. It also effectively confines the electron flow between the source and drain, substantially minimizing leakage currents that can degrade performance [196].
In the full manufacturing flow, the CMOS front-end, plus the first four metal layer are developed by the CMOS technology supplier. The NVM elements are subsequently developed in the CEA LETI cleanroom facilities, between the fourth and the fifth metal layer. Finally, the last seven metal layers are processed by the foundry, in order to obtain the complete stack, counting a total of eleven metal layers. This full process allows the manufacturing of a complete I/O subsystem, responsible for communicating data between the chip and the external world. The design and specifications of the I/O pad ring are detailed in Appendix D. The development of a complete I/O subsystem allows to wire-bond the designed chip to a package. Thus, the packaged chip can be tested by means of a printed circuit board (PCB), interfacing the chip and a work station (Figure 4.2a).
The partial manufacturing flow shares the first two steps depicted in the previous case, i.e. the CMOS front-end development by the foundry, plus the NVM elements developed in-house. Differently than the previous case, the partial manufacturing flow is terminated in the CEA LETI cleanroom facilities, with the deposition of the fifth, last, metal layer. The 25-pads test scribes are therefore developed up to the fifth metal layer. The pads in each test scribe do not include the circuitry that can be found in the I/O subsystem pads, rather they directly connect the core to the external world via metal lines. The pin listing of the two test scribes used to address the FeЯNet core is presented in Appendix D. The

purpose of this manufacturing flow is to have faster access to the testing of the core, via an automatic or semi-automatic probe station. The test scribes on the silicon wafer are addressed via probe card (Figure 4.2b).
Each FeЯNet core, in either designs, is made of two sub-cores (Figure 4.2c). The two sub-cores can be operated exclusively, according to an input control signal. Architecturally, the two sub-cores are identical, but different selector transistors were used for 1T-1C cells and 1T-1R cells in the ferroelectric and resistive memory arrays of the two sub-cores. In the "left" sub-core, thick gate oxide (referred to as GO2) selector transistors were used, whereas on the "right" sub-core thin gate oxide (referred to as GO1) transistors were used as selector devices. Transistors with thinner gate oxide layer generally have better control over the channel, leading to faster switching speeds and lower gate drive voltages. However, thinner gate oxides can make the transistor more susceptible to issues like gate leakage current and reduced reliability due to the increased electric field strength, resulting less robust under high-voltage conditions. Indeed, the nominal supply voltage of GO1 transistors in the 22 nm FDSOI technology is 0.8 V. A thicker gate oxide layer improves the transistor's ability to handle higher voltages and enhances reliability, as it provides better insulation and reduces leakage currents. It can be more robust against stress and long-term wear. On the other hand, a thicker oxide layer might lead to slower switching speeds and higher gate drive voltages compared to transistors with thinner oxides. The nominal supply voltage of GO1 transistors in the 22 nm FDSOI technology is 1.8 V. GO1 transistors also have smaller minimal channel length compared to GO2 transistors, allowing for more compact array density implementations. Therefore, the two sub-cores were developed in order to evaluate the impact of the selector transistor on the system performance and reliability.

Bit-cells and arrays The basic building blocks of the FeЯNet core are the ferroelectric and resistive memory arrays, which store the weights optimized during training and their corresponding binarized values, respectively.
The arrays, based on 1T-1C and 1T-1R structures, are organized with parallel bit lines and source lines, and perpendicular word lines, as shown in Figure 4.3. Each FeRAM and RRAM array contains 16,384 bit-cells, arranged in a grid of 128 word lines and 128 bit/source lines.
Table 4.1 summarizes the physical dimensions of the selector transistors used in the two designed sub-cores. In both sub-cores, the same non-volatile memory elements were used, i.e. square memory devices with 170 nm side. The designed resistive and ferroelectric memory arrays with GO1 selector transistors have 141% and 18% larger density, respectively, compared to their GO2 counterparts.
While both 1T-1C and 1T-1R bit-cell configurations are used in FeRAM and RRAM arrays, resistive arrays employ two 1T-1R bit-cells (2T-2R unit cell) in a differential configuration to store one bit of information. In a resistive unit-cell, data is encoded as follows:

- For a given word line (WL_{Re}), if the resistive device in $BL_{Re,i}$ ($SL_{Re,i}$) is in a low-conductance state and $BL_{Re,i+1}$ (SL_{i+1}) is in a high-conductance state, the cell encodes a logic 1.
- Vice versa, if the resistive device in $BL_{Re,i}$ ($SL_{Re,i}$) is in HCS and the device in $BL_{Re,i+1}$ ($SL_{Re,i+1}$) is in LCS, the cell encodes a logic 0.

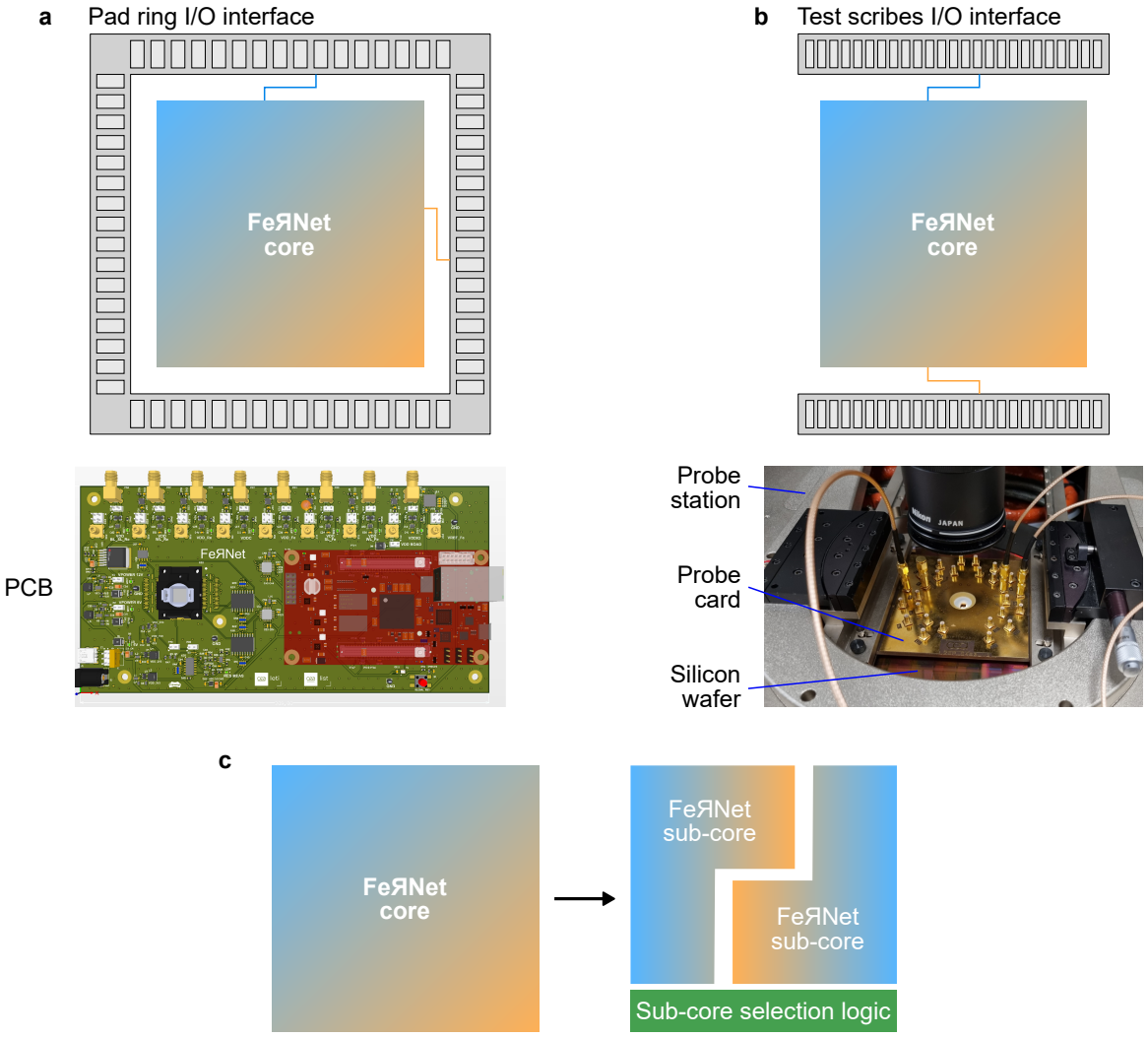

Figure 4.2: **Designs of the FeЯNet circuit.** The same FeЯNet architecture is either connected to an input/output pad ring, for testing with a PCB support (**a**), or to a double test scribe input/output interface for testing with a automatic or semi-automatic probe station (**b**). **c** Each FeЯNet core is made of two architecturally analogous sub-cores.

Here, the index i refers to an even-numbered bit line or source line. In contrast, ferroelectric memory arrays store a single bit using a 1T-1C cell.

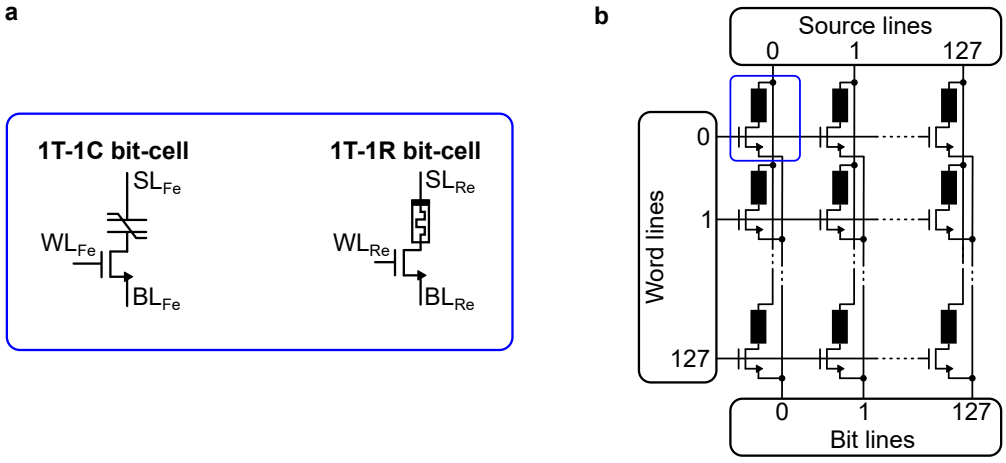

Figure 4.3: **Ferroelectric and resistive bit-cell arrays in the FeЯNet architecture.** **a** Ferroelectric and resistive arrays are based on 1T-1C and 1T-1R structures. **b** The memory arrays are organized with parallel bit lines and source lines and perpendicular word lines. Each memory arrays has 16,384 bit-cells.

Bit-cell type	Selector transistor		NVM area (μm^2)	Bit-cell area (μm^2)
	Length (nm)	Width (nm)		
1T-1C (GO1)	28	252	0.029	0.11
1T-1C (GO2)	150	320	0.029	0.14
1T-1R (GO1)	28	880	0.029	0.11
1T-1R (GO2)	150	880	0.029	0.28

Table 4.1: **Bit-cell size specifications for the FeЯNet sub-cores.** List of the selector transistors (1T) dimensions, memory element (1C or 1R) area and bit-cell (1T-1C or 1T-1R) area for both sub-cores (GO1 or GO2 bit-cell types).

Non-volatile memory integration The memory stack presented in chapter 2 shows promise for the realization of a unified memory that can be used either as ferroelectric and resistive memory. Nevertheless, at present, its scalability at more advanced technology nodes is prevented by the large forming voltage applied to the memristor top electrode (approximately 6 V). The primary feature that allows to reduce the forming voltage is reducing the oxide thickness. Nevertheless, thickness scaling is incompatible with ferroelectric operation, which is hampered by excessively thin oxides. Therefore, further research at material and device level is necessary to induce ferroelectricity in thinner films at BEOL-compatible temperature, for integrating this stack at more advanced technology nodes, such as the 22 nm FDSOI technology.
For this reason, the flow described in section 2.5.2 is envisioned for the manufacturing of FeЯNet. This memory co-integration flow allows to have slightly different processing for FeRAMs and RRAMs developed in the same BEOL with the addition of a single mask. This supplementary cost might be essential for the successful co-integration of high-performance ferroelectric and resistive memories.

Architecture of a FeЯNet sub-core The top level schematic view of each sub-core is shown in Figure 4.4. Each sub-core includes four FeRAM arrays and one RRAM array. One out of four FeRAM arrays can be operated at the same time and, simultaneously, one RRAM array. This is possible thanks to independent pads used to address the two array types (see Appendix D).
The FeЯNet circuit leverages the RRAM array to create a near-memory computing system, accelerating both the forward and backward passes during training. Meanwhile, the FeRAM array is used to store the hidden weights in either integer or floating-point formats, with widths of 8 bits or multiples thereof. 8-bits floating point formats – based on a sign, exponent, mantissa representation similar to the standard half- and full-precision formats – have been proposed to train neural networks in order to improve energy-efficiency and reduce memory requirements. Refs.[197, 198] train deep neural networks with 8-bit minifloat numbers achieving the same accuracy levels as the full-precision across a wide spectrum of benchmarks and datasets. At present, no research paper analysed the possibility to train quantized neural networks, thus BNNs, with 8-bit mini-float representation for the hidden weights. The FeЯNet core was also designed for this purpose.
Transfer logic circuits are included to transfer the signs of the hidden weights from the FeRAM arrays to the RRAM array. The logic is designed to transfer the signs stored in a full word line from one out of four FeRAM arrays to the RRAM array, completing a total of sixteen transfers per programming step, in the case of 8-bit hidden weights. During each programming step, thirty-two 1T-1R devices are reset, followed by the parallel set of sixteen devices. Additional details are provided in section 4.2.3.
The synaptic capacity, or the number of weights that can be stored in each FeЯNet sub-core, depends on the desired precision of the hidden weights. The maximum capacity is achieved with 8-bit hidden weights, allowing each sub-core to store up to 8,000 synapses (16,000 parameters in a full FeЯNet core). As the bit-width of the hidden weights doubles, the number of synapses is halved.

4.2.1 FeRAM arrays

Each FeRAM array includes the bit-cells array and the peripheral circuitry. The periphery of FeRAM arrays is completely designed with GO2 transistors and standard cells, with nominal supply voltage of 1,8 V. Scaling the operating voltage of FeCAPs devices is essential for integrating FeRAMs at more advanced nodes. Reliable ferroelectric array operation at 2 V was for example obtained through thickness-scaling of a HZO material from 10 to 8 nm [199]. Further scaling the operating voltage requires decreasing the thickness of the ferroelectric film. Nevertheless, a clear trade-off between the BEOL crystallization temperature and the ferroelectric film thickness exits.
In the following, a top level description of the FeRAM array is presented, with a detailed description of the different sub-blocks (Figure 4.5).

Top level description Each FeRAM array features independent line drivers, input scan chains, sense amplifiers, and an output scan chain. Scan chains, scan chain control circuitry, BL drivers and sense amplifiers are supplied by the supply voltage VDD_Fe (pad 7 and 42, Table D1). Word line and source line drivers are supplied by supply voltages VDD_WL_Fe (pad 5 and 44, Table D1) and VDD_SL_Fe (pad 6 and 43, Table D1),

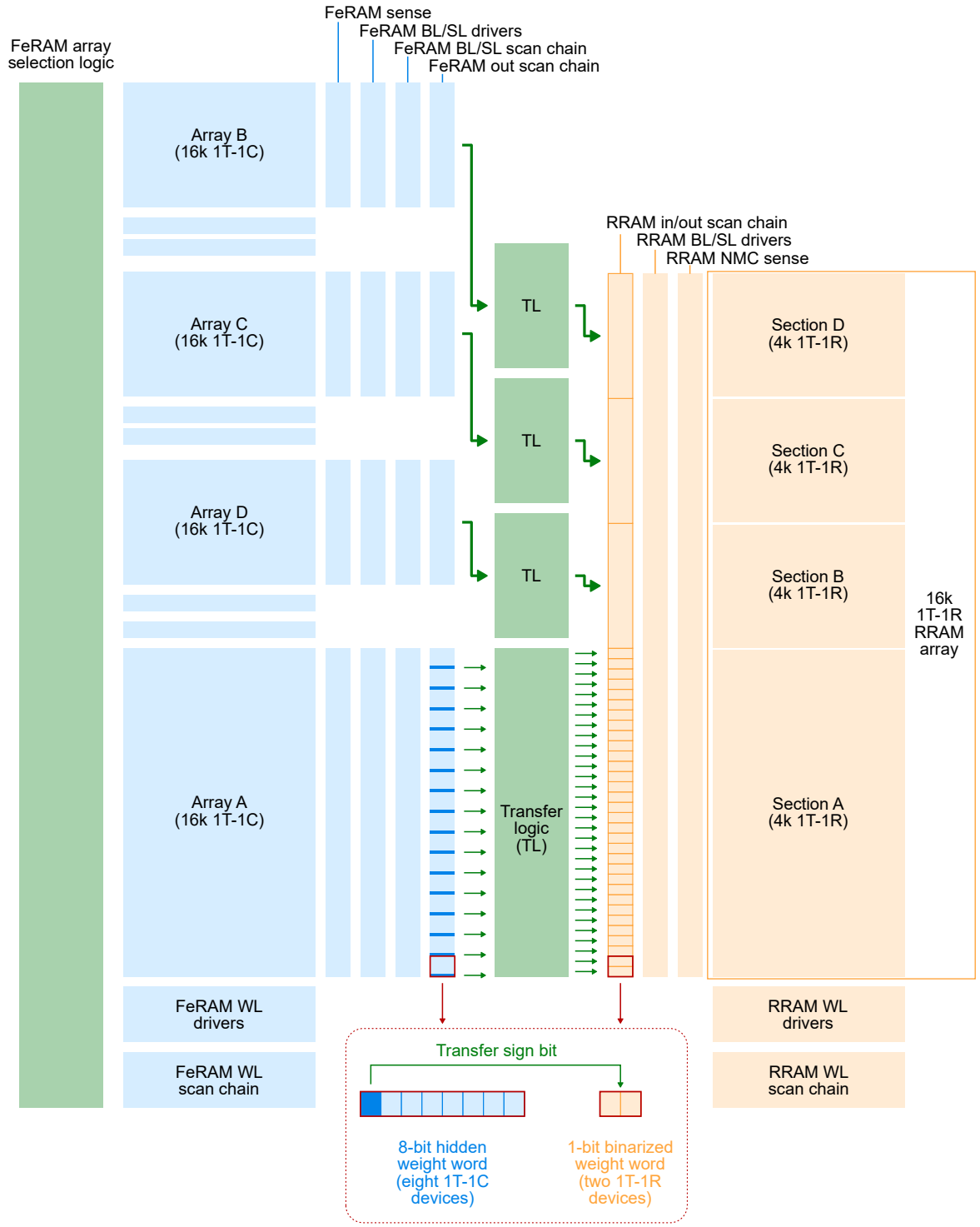

Figure 4.4: **Schematic architecture of a FeЯNet sub-core.** Each sub-core includes four FeRAM arrays (light blue boxes), one RRAM array (orange boxes), array selection logic and transfer logic circuits (green boxes). The transfer logic takes as input a bit from the FeRAM output scan chain, representing the sign of a hidden weight, and outputs two complementary bits to the input/output scan chain of the RRAM array, defining which device to set in the 2T-2R unit cell.

respectively. VDD_Fe, VDD_WL_Fe and VDD_SL_Fe should be optimized according to device technology and should preferably stay as close as possible to 1,8 V.
Inputs to the FeRAM array are introduced via a selection logic block, necessary to bring the input signals to the default value, when a specific FeRAM array is not selected for operation. For active-high signals, the selection logic is simply an AND gate between the input signal and the enabling signal for such array. For active-low input signals, a NAND gate is used as selection logic. In this way, all input signals provided by the input pads are active-high. The array enable signal is decoded starting from the sub-core selection signal (Left_RightB, pad 15, Table D1) and the Matrix_sel[1:0] signals (pad 55 and 56, Table D1).
FeRAM arrays have a serial output provided by the output of the last flip-flop in the output scan chain. The serial output of each FeRAM array can be connected to the SC_OUT_Fe pad (pad 58, Table D1), according to the array enable signal. Also, each FeRAM array has sixteen parallel outputs, provided by the output of sixteen registers equally spaced in the output scan chain. Further details on the usage of these signals are provided in section 4.2.3.

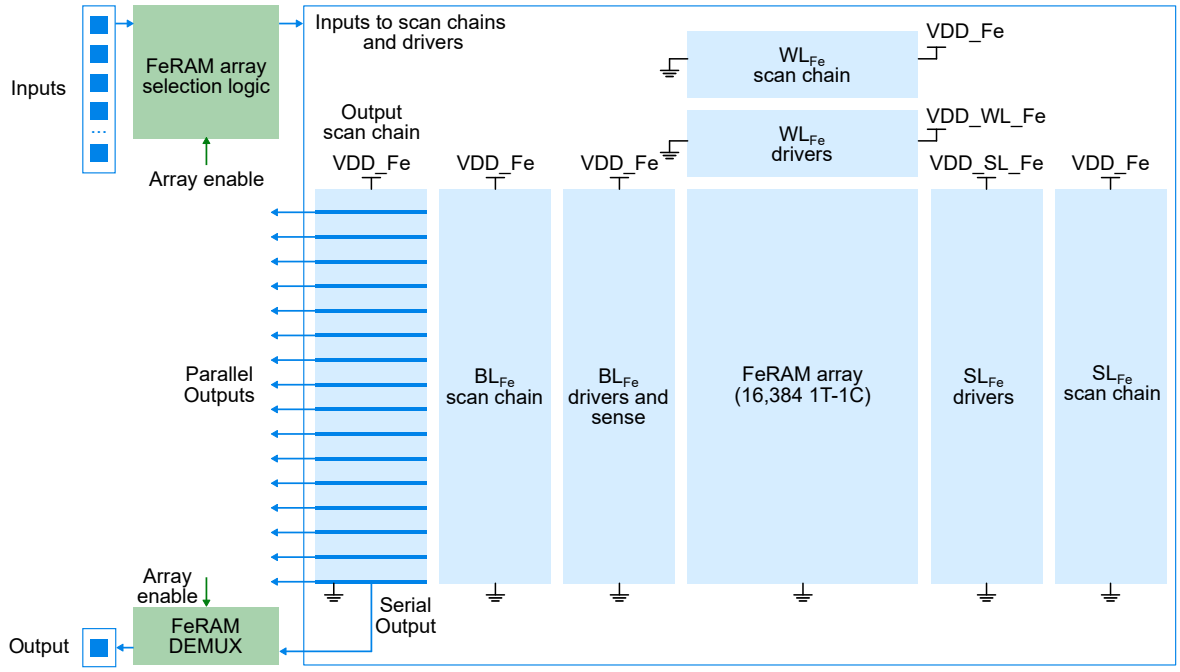

Figure 4.5: **Schematic architecture of a FeRAM array in the FeЯNet design.** The inputs dedicated to FeRAM arrays, provided by the I/O subsystem, are input to each FeRAM array selection block. Only the active FeRAM array receives the inputs. Each FeRAM array has a serial output, provided by the last flip-flop in the output scan chain and demultiplexed according to the array enable signal, and sixteen parallel outputs. The supply voltages for the different blocks are indicated in the figure.

Scan chain system. The inputs to the array drivers, defining the active and inactive lines of the circuit are provided to the circuit via scan chains. A single scan chain at the time can be used by the enabled FeRAM array. In order to select the active scan chain, a 2-bit input signal is provided, SC_Sel_Fe[1:0] (pads 50 and 49, Table D1). The SC_Sel_Fe[1:0] signal is decoded with a 2-to-4 decoder in order to select the desired scan

chain. A schematic of the scan chain system for each FeRAM array is shown in Figure 4.6. The four scan chains have three common signals, that are the clock (Ck_SC_Fe, pad 48 in Table D1), the input (SC_In_Fe, pad 53 in Table D1) and the reset (Rst_Fe, pad 47 in Table D1). The size of each scan chain, i.e. the number of flip flops composing it, is indicated in Figure 4.6. As the same inputs are provided to SL_{Fe} and BL_{Fe}, their respective scan chain share the same enabling signal.

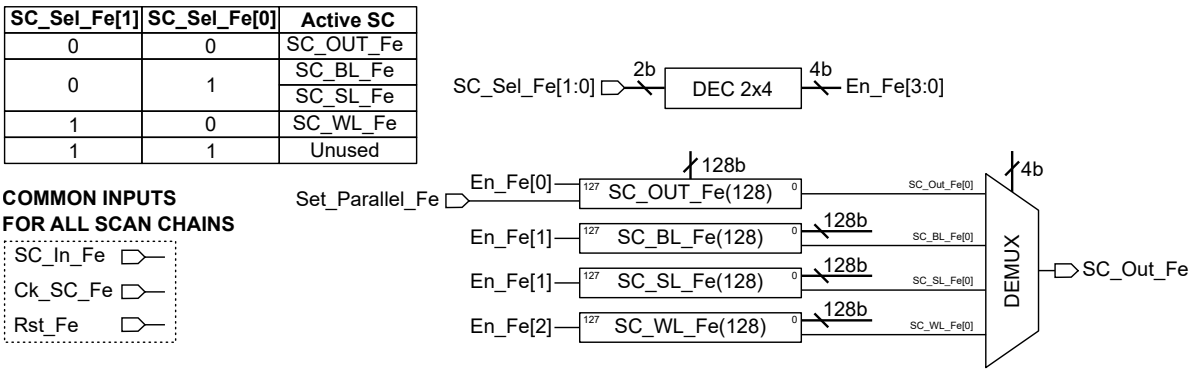

Figure 4.6: **Scan chain system of FeRAM arrays in the FeЯNet design.**

Word line and source line drivers. The WL_{Fe} (SL_{Fe}) drivers are simply implemented as the AND operation of the value written in the respective scan chain element and an external trigger signal, WL_Pulse_Fe (SL_Pulse_Fe), pad 64 in Table D1 (pad 59 in Table D1). The output of the AND cells are buffered in order to provide sufficiently large driving capability. The buffers and AND gates are supplied by VDD_WL_Fe (VDD_SL_Fe). Thus, a WL_{Fe} (SL_{Fe}) can be connected to VDD_WL_Fe (VDD_SL_Fe) only if a 1 is inserted in the scan chain for said WL_{Fe} (SL_{Fe}) and the trigger signal is active. Otherwise, the WL_{Fe} (SL_{Fe}) is grounded.

Sensing elements and bit line drivers. The BL_{Fe} block comprises both the bit line drivers as well as the sense amplifiers for the FeRAM array (circuit in Figure 4.7). The supply voltage of the BL_{Fe} block is VDD_Fe. The BL_{Fe} block also requires a reference voltage for the sensing operation (Vref_Fe, provided by pad 45, Table D1), which is implemented by a clocked latch sense amplifier. The input signals to the BL_{Fe} block of the enabled FeRAM array are:

- The precharge trigger signal, Pre_Pulse_Fe, pad 1 in Table D1. This trigger signal allows to ground the BL_{Fe}, if the sense amplifier is not activated. It is necessary to set the BL_{Fe} to ground before performing a read operation, in order to obtain a reliable reading.
- The sense amplifier enable signal, SA_Pulse_Fe (pad 61 in Table D1), to connect or disconnect each latch sense to BL_{Fe}. Specifically, the two inputs of the latch are pre-charged to the BL_{Fe} voltage and the reference voltage, until the sensing operation is enabled. When the sense operation is enabled, the latch toggles.

- The write back trigger signal, WB_Pulse_Fe (pad 60 in Table D1). This trigger signal allows automatic write-back of the value stored in the FeRAM device after destructive reading. This is possible as the output obtained during the reading operation stays unchanged unless the BL_{Fe} is pre-charged to ground.
- The write enable signal, WE_Pulse_Fe (pad 63 in Table D1), and the data written in the BL_{Fe} scan chain element. The NAND operation of these two signals allows to connect the BL_{Fe} to the supply voltage VDD_Fe, thus programming the ferroelectric device.

The BL_{Fe} block outputs the result of the sense operation. The outputs of the sensing elements are connected to the inputs of the respective registers in the output scan chain. The Set_Parallel_Fe (pad 54, Table D1) signal has to be activated for copying the outputs of the sensing circuits in the output scan chain.

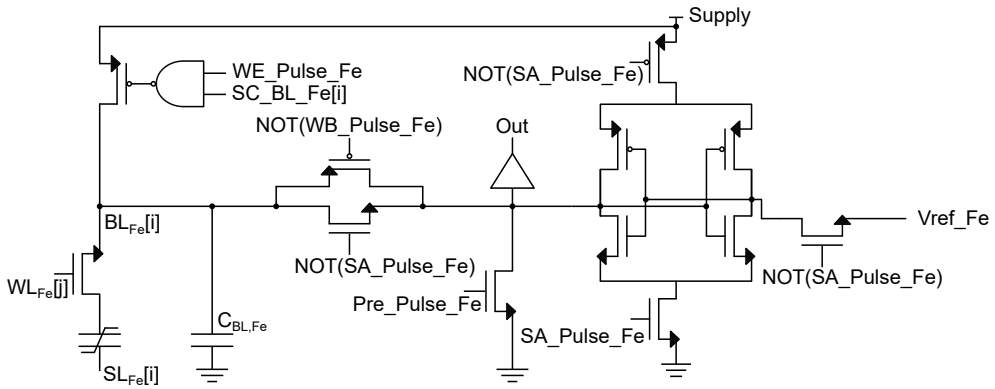

Figure 4.7: **Schematic of the bit line block of FeRAM arrays in the FeЯNet design.** Circuits of a bit line block connected to a 1T-1C cell in the FeRAM array. The BL_{Fe} parasitic capacitance is shown as $C_{BL,Fe}$.

Layout view. The layout view of each designed FeRAM array is shown in Figure 4.8. The different blocks are highlighted in green boxes.

4.2.2 RRAM arrays

Each RRAM array includes the bit-cells array and the peripheral circuitry. The periphery of RRAM arrays is designed with GO2 transistors and standard cells, and laterally-diffused metal-oxide semiconductor (LDMOS) devices [200]. In the 22 nm FDSOI technology, LDMOS can support larger drain-to-source voltages, up to 3.3 V thanks to a drain extension region that makes them non-symmetric devices. They can be used in combination with GO2 logic to supply the various lines with voltages larger than 1.8 V. Indeed, a voltage larger than 1.8 V is typically necessary for the forming operation of RRAM devices. For this reason LDMOS were used together with standard CMOS devices. In the following, a top level description of the RRAM array is presented, with a detailed description of the different sub-blocks (Figure 4.9).

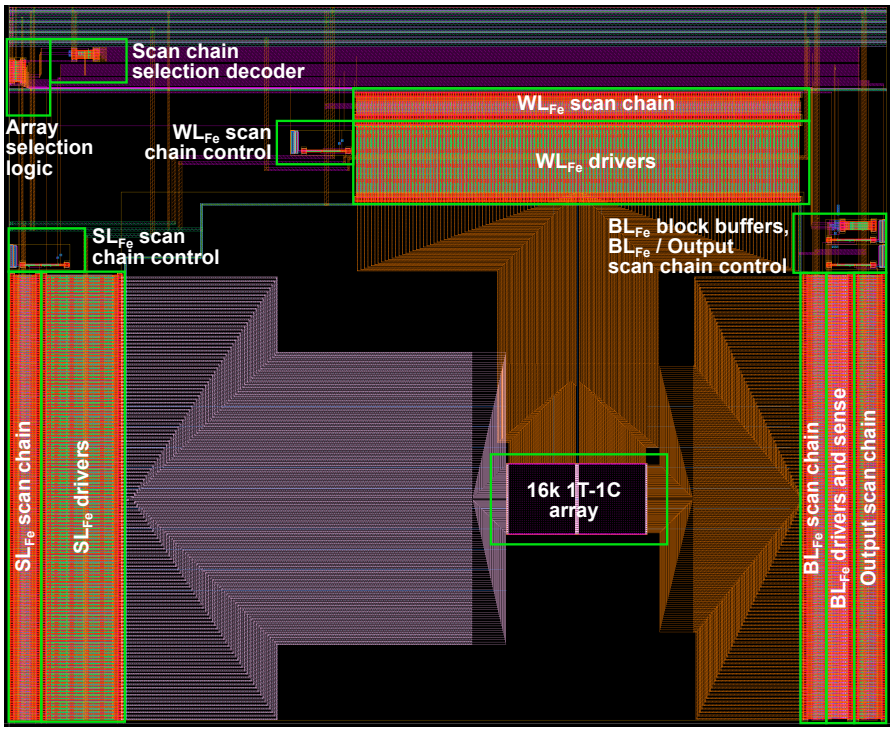

Figure 4.8: **Layout view of FeRAM arrays in the FeЯNet design.** The different blocks in the layout are highlighted in green boxes.

Top level description The RRAM array includes its own line drivers and logic, scan chains, and sense amplifiers. The sense amplifiers in RRAM arrays are used to to sense the binary information stored in the 2T-2R RRAM unit cells and to perform the XNOR operation in-place, between the data stored in the memory and a binary logic input. An input/output scan chain in the RRAM array is used at the same time to present the inputs to the sensing elements and to collect the outputs, as explained later in this section. All scan chains and scan chain control circuity, as well as lines logic blocks are supplied by the supply voltage VDD_Re (pads 10 and 39, Table D1), typically set to 1.8 V. Word line drivers are supplied by supply voltages VDD_WL_Re (pads 11 and 38, Table D1), whereas both bit line and source line drivers are supplied by supply voltage VDD_BL_SL_Re (pads 12 and 37, Table D1). VDD_WL_Re and VDD_BL_SL_Re are optimized according to the device technology. The sensing element is supplied by VDDC (pads 9, 25, 41, 57 in Table D1), typically set at 0.8 V.
Inputs to the RRAM array are introduced via a selection logic block, as in the case of the FeRAM array, in order to bring the input signals to the default value, when a specific array is not selected for operation. The array enable signal is the same as the sub-core selection signal (Left_RightB, pad 15 in Table D1). Moreover, inputs provided to the input/output scan chain can also come from the transfer logic block, if the transfer operation from FeRAM arrays to RRAM array is enabled via the Transfer signal (pad 2, Table D1).
RRAM arrays have a serial output provided by the output of the last flip-flop in the input/output scan chain. The serial output of each RRAM array can be connected to the SC_OUT_Re pad (pad 24, Table D1), according to the array enable signal.

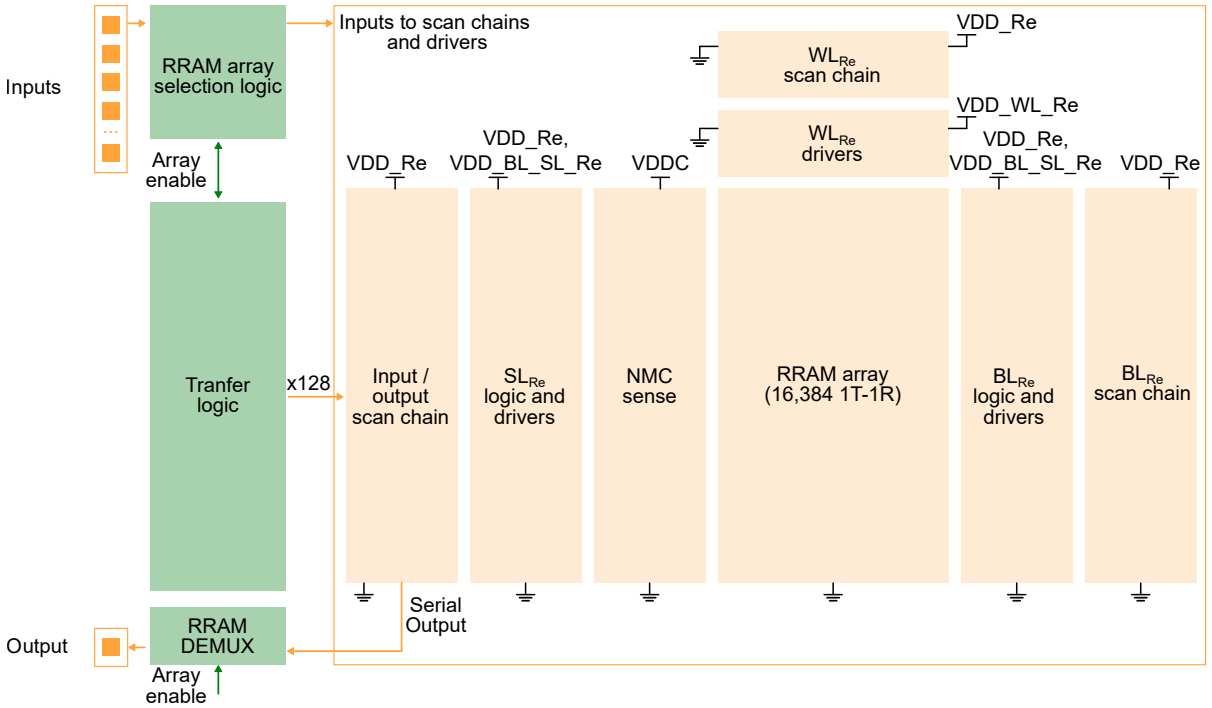

Figure 4.9: **Schematic architecture of a RRAM array in the FeЯNet design.** The inputs dedicated to the RRAM arrays, provided by the I/O subsystem, are input to each RRAM array selection block. Parallel inputs can be provided to the input/output scan chain by the transfer logic circuits. Only the active RRAM array receives the inputs. Each RRAM array has a serial output, provided by the last flip-flop in the output scan chain and demultiplexed according to the array enable signal. The supply voltages for the different blocks are indicated in the figure.

Scan chain system. The RRAM array includes three different Scan Chains, the in/out scan chain, the WL_{Re} scan chain, the BL_{Re} scan chain. Each one is composed of 128 registers. Therefore, three enable signals are necessary to select all different scan chains, which are obtained with a two-to-four decoder (Figure 4.10a). The three scan chains have two common input signals, that are the clock (Ck_SC_Re, pad 27 in Table D1) and the input (SC_In_Re, pad 23 in Table D1). An independent reset signal is used for BL_{Re} (Rst_BL_Re, pad 18 in Table D1), WL_{Re} (Rst_WL_Re, pad 16 in Table D1) and SL_{Re} (Rst_SL_Re, pad 17 in Table D1) scan chains. A common output is shared among the all scan chains (SC_Out_Re, pad 24 in Table D1). The scan chain outputs are de-multiplexed according to the enabling signals of each scan chain.
The operation of the in/out scan chain is described in Figure 4.10b. As the name suggests, the same scan chain is used to push data inside the scan chain or to pull data from it. The in/out scan chain can be used as input scan chain in two different configurations:

- Serial mode: When using the RRAM array alone, it is necessary to select the SL_{Re} that should be read or programmed. This is possible with the in/out scan chain, if the Set_Parallel_Re (pad 26 in Table D1) command signal is low.
  Data that are input in a serial mode can also be used to provide inputs to the sensing elements. Indeed, as shown in the following, the differential sensing element performs an XNOR operation in place between the provided inputs and the values stored inside the memory. Whether data in the in/out scan chain are used as input

to the sensing element is determined by the Sense_Re signal (pad 58 in Table D1), as shown in Figure 4.10b. If the Sense_Re signal (pad 32 in Table D1) is high, the output of the registers in even positions along the chain are input to the sensing elements. It should be noted that the in/out scan chain has 128 registers, in order to address all devices. Nevertheless, each sensing element is connected to two 1T-1R devices. Therefore, if the scan chain is used to load inputs for the sense, the inputs should be loaded by skipping one clock cycle between two provided inputs.
If the Sense_Re signal is low, the in/out scan chain can be used to provide information about which BL to activate.

- Parallel mode: Inputs to the scan chain registers can also be provided in parallel. This functionality was implemented in order to allow a parallel transfer of information from FeRAM to RRAM arrays. This transfer operation is detailed in section 4.2.3. For now, it is sufficient to know that if the Set_Parallel_Re signal is high and the Sense_Re command is low, all 128 registers of the chain receive parallel input data from the transfer logic block.

The in/out scan chain can be used as Output scan chain. In this configuration data from the sensing elements are transferred as inputs to the scan chain, to be collected at the output pad SC_OUT_Re. The transfer of data from the sense to the in/out scan chain occurs only if both the Set_Parallel_Re and Sense_Re signals are high. If this condition is met, the output of the sensing is loaded in the registers in the odd positions of the scan chain, as shown in Figure 4.10b.

Word line logic and drivers. The WL_{Re} block includes a logic block, designed with GO2 transistors and supplied by VDD_Re, and the drivers block, designed with LDMOS devices and supplied by VDD_WL_Re. The WL_{Re} logic block controls the operation performed by the WL_{Re} drivers.
Each WL_{Re} logic block takes as input the logic value in the respective WL_{Re} scan chain element (SC_WL_Re[i]) and outputs two signals:

- Ctrl_VDD_WL[i] = SC_WL_Re[i]
- Ctrl_GND_WL[i] = $\overline{\text{SC\_WL\_Re[i]}}$

Each WL_{Re} driver takes as input the respective Ctrl_VDD_WL[i] and Ctrl_GND_WL[i] signals to either connect the WL_{Re} to VDD_WL_Re or to ground. Figure 4.11a shows the WL_{Re} driver circuit. In order to connect the WL to ground, a n-LDMOS is used, controlled by the active-high signal Ctrl_GND_WL[i]. On the other hand, if the CTRL_VDD_WL[i] signal is high, the WL is connected to VDD_WL_Re.
This circuit implementation creates a connection between VDD_WL_Re and ground, through the resistor, when Ctrl_VDD_WL[i] is high, resulting in large power consumption. The reason preventing to use a simple inverter circuit, controlled by the signal Ctrl_GND_WL[i], which would result in zero static power consumption is the following. The high level of the logic block is VDD_Re, which is set to 1.8 V. This value might not be sufficiently high to open the p-LDMOS in an inverter circuit, when the Ctrl_GND_WL[i]

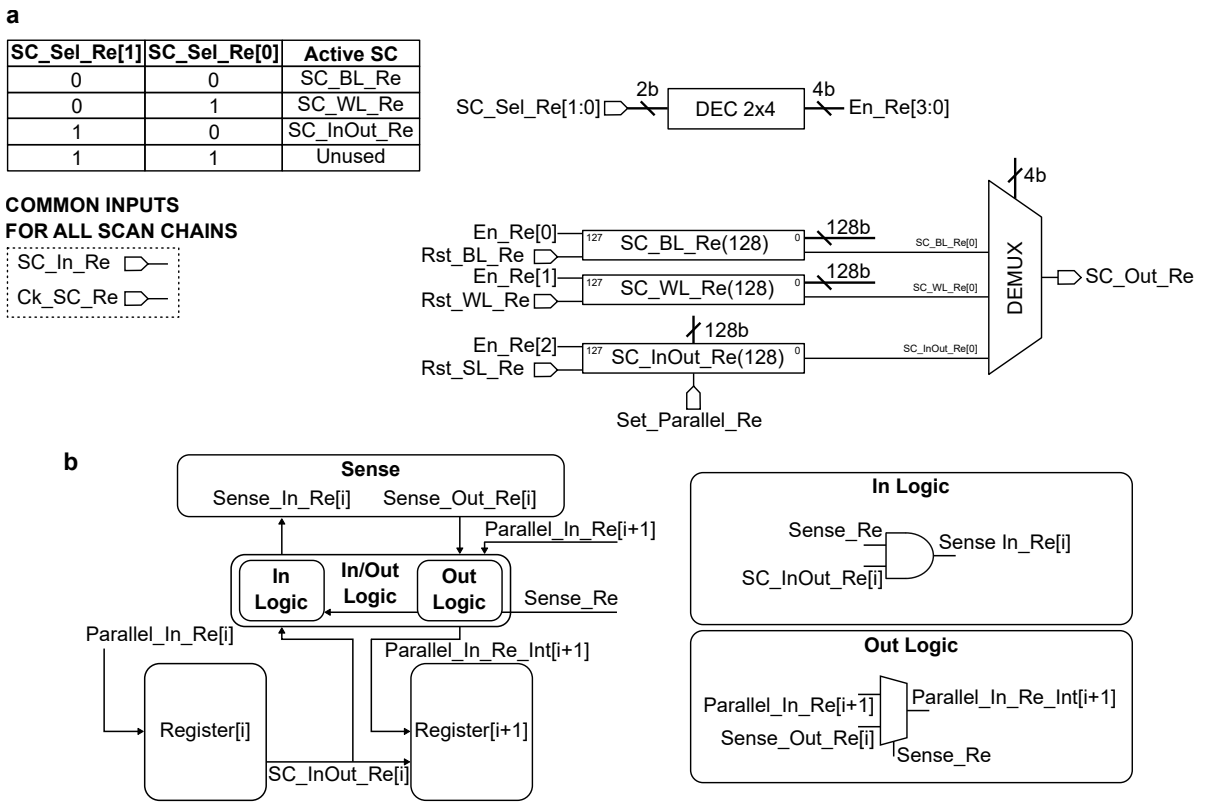

Figure 4.10: **Scan chain system of FeRAM arrays in the FeЯNet design. a** Global description of the scan chain system and decoding table. **b** Detailed schematic description of two registers in the in/out scan chain of the RRAM array. The same blocks are repeated for each couple of registers.

input is high, depending on the magnitude of VDD_WL_Re, preventing to effectively ground the word line. Therefore, a properly sized resistor and an n-LDMOS (controlled by the Ctrl_VDD_WL[i]) are used to control the gate of the p-LDMOS.

Bit line and source line logic and drivers. The BL_{Re} and SL_{Re} blocks includes logic blocks, designed with GO2 transistors and supplied by VDD_Re, and the drivers block, designed with LDMOS devices and supplied by VDD_BL_SL_Re. The BL_{Re} logic block controls the operation performed by the BL_{Re} drivers, and the same applies to SL_{Re}. Each BL_{Re} logic block takes as input the SC_BL_Re[i] signal, i.e. the data loaded in the scan chain element corresponding to the driver under consideration, and two more control signals, i.e. Reset_Re (pad 29 in Table D1) and Probe_Re(pad 28 in Table D1). The block has three output signals:

- $\text{Ctrl\_VDD\_BL[i]} = \text{SC\_BL\_Re[i]} \cdot \text{Reset\_Re}$
- $\text{Ctrl\_GND\_BL[i]} = \overline{\text{SC\_BL\_Re[i]}} + \text{SC\_BL\_Re[i]} \cdot \overline{\text{Reset\_Re}} \cdot \overline{\text{Probe\_Re}}$
- $\text{Ctrl\_Probe\_BL[i]} = \text{SC\_BL\_Re[i]} \cdot \text{Probe\_Re}$

Each SL_{Re} logic block takes as input the SC_SL_Re[i] signal, i.e. the data loaded in the scan chain element corresponding to the driver under consideration, and three more

control signals, i.e. Set_Re (pad 31 in Table D1), Probe_Re and Sense_Re. The block has three output signals:

- $\text{Ctrl\_VDD\_SL[i]} = \text{SC\_SL\_Re[i]} \cdot \text{Set\_Re} \cdot \overline{\text{Sense\_Re}}$
- $\text{Ctrl\_GND\_SL[i]} = (\overline{\text{SC\_SL\_Re[i]}} + \text{SC\_SL\_Re[i]} \cdot \overline{\text{Set\_Re}} \cdot \overline{\text{Probe\_Re}}) \cdot \overline{\text{Sense\_Re}}$
- $\text{Ctrl\_Probe\_SL[i]} = \text{SC\_SL\_Re[i]} \cdot \text{Probe\_Re} \cdot \overline{\text{Sense\_Re}}$

Each BL_{Re}, or SL_{Re}, driver takes as input the respective signals defined just above to either connect the BL_{Re}, or SL_{Re}, to VDD_BL_SL_Re, ground or to the external pads to probe the resistance (BL_Probe_Re and SL_Probe_Re, pads 13 and 36 in Table D1 respectively). Figure 4.11b shows the BL_{Re} and SL_{Re} driver circuit. In order to connect the BL_{Re} (SL_{Re}) to ground, a n-LDMOS is used, controlled by the active-high signal Ctrl_GND_BL[i] (Ctrl_GND_SL[i]). If the Ctrl_VDD_BL[i] (Ctrl_VDD_SL[i]) signal is high, the BL_{Re} (SL_{Re}) is connected to VDD_BL_SL, via the coupled n-LDMOS-resistor and p-LDMOS structure explained before for the WL_{Re} drivers. A third path is possible in the BL_{Re} and SL_{Re} drivers, conversely to the WL_{Re} driver. This path connects the BL_{Re} (SL_{Re}) to the BL_Probe_Re (SL_Probe_Re) pads via a n-LDMOS activated by the active-high signal Ctrl_Probe_BL[i] (Ctrl_Probe_SL[i]).
Finally, it should be noted that the SL_{Re} logic includes also a configuration in which none of the three driver command signals is active. This means that none of the three paths previously defined are active and the SL_{Re} is left in a high-impedance mode. This configuration occurs when the Sense_Re signal is high, thus when it is necessary to perform a read operation via the sense amplifiers.

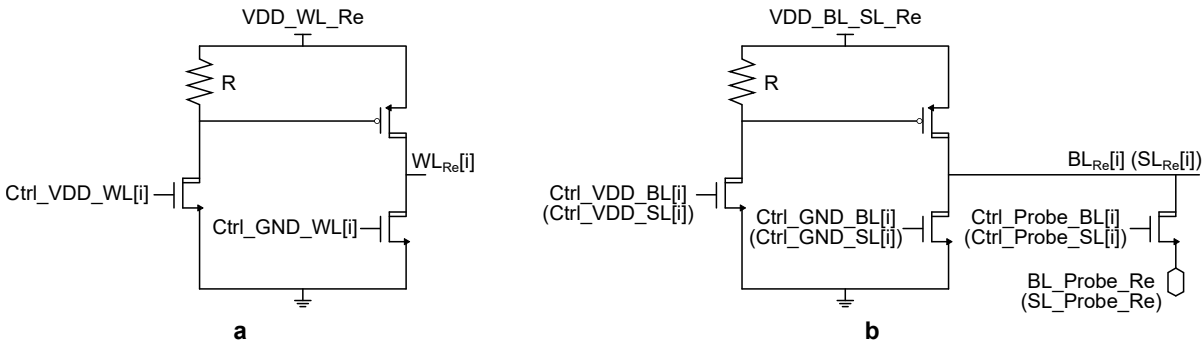

Figure 4.11: **Driver circuits of RRAM arrays in the FeЯNet design.** Word line drivers (**a**) and bit line and source line drivers (**b**). The resistance value is R=8 kΩ.

Near-memory computing sensing element. A differential sense amplifier (DSA) circuit was designed to sense the binary information stored in the 2T-2R RRAM cells [201]. Each sensing element is connected to two consecutive SL_{Re}. Thus, 64 DSAs are present for each RRAM array. The designed DSA also allows to perform the XNOR operation in-place, between the data stored in the array (a logic 0 or 1) and a binary logic input signal presented to the DSA. The XNOR operation between two 1-bit logic values maps the arithmetic multiplication between two quantities which can assume only two values, -1 (mapped to the logic value 0) and 1 (mapped to the logic value 1). The XNOR operation performed by the DSA is summarised in Table 4.2.

x_i	X_i	w_{ij}	W_{ij}	$x_i \cdot w_{ij}$	XNOR(X_i,W_{ij})
-1	0	1	1	-1	0
-1	0	-1	0	1	1
1	1	1	1	1	1
1	1	-1	0	-1	0

Table 4.2: **Equivalence between the exclusive-NOR (XNOR) logic operation (0/1) and multiplication between binarized values (±1).** In the table, X_i and W_{ij} represent binary values (0/1), while x_i and w_{ij} represent their binarized equivalents (±1). The XNOR result corresponds to the arithmetic multiplication.

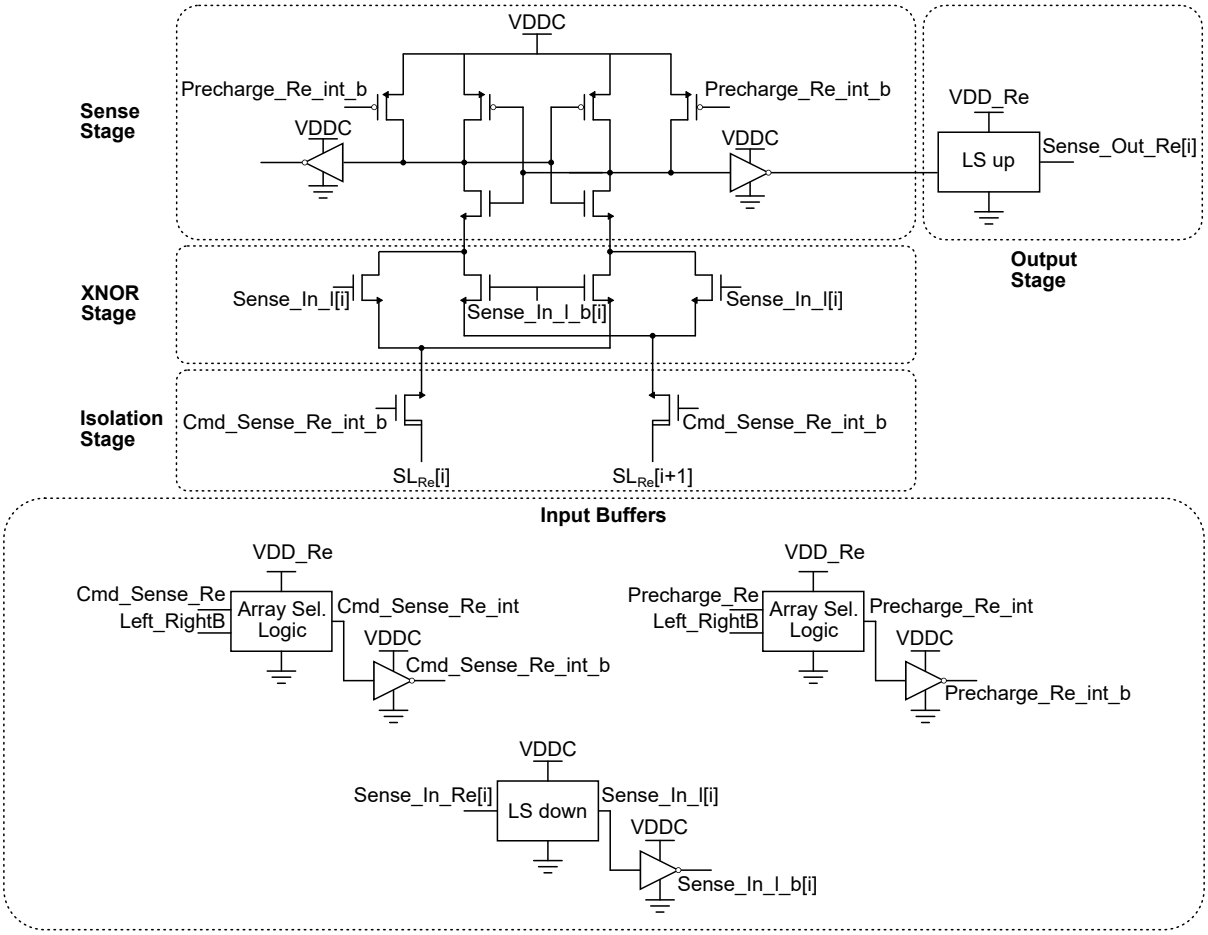

Figure 4.12: **Schematic of the differential sense amplifier with in-place XNOR operation.** This sensing circuit enables the implementation of the XNOR operation between a binary input and data stored in a differential 2T-2R unit cell. The signals Precharge_Re (pad 33 in Table D1) and Cmd_Sense_Re (pad 34 in Table D1) define when to precharge the output of the latch circuit to the supply and when to activate the sense.

The designed DSA circuit, shown in Figure 4.12, is made of several blocks:

- The isolation stage connects the two SL_{Re}, thus the two resistive devices, to the sensing element via n-LDMOS controlled by the input signal Cmd_Sense_Re_int_b.

- The XNOR stage includes four GO1 high-threshold voltage n-MOSFET transistors an it implements the XNOR operation between the data stored in the 2T-2R bitcell and the input presented to the DSA, by switching or not the two SL_{Re} with respect to the sensing latch. The inputs are presented to the gates of the MOSFETs in this stage, via the signals Sense_In_l[i] and Sense_In_l_b[i]. The signal Sense_In_l[i] is the output of a level-shifter circuit (supplied by VDDC), taking as input the Sense_In[i] signal provided by the in/out scan chain (supplied by VDD_Re).
- The sense stage includes the latch circuit, the pre-charge transistors and the output buffers. The pre-charge transistors, GO1 high-threshold voltage p-MOSFET, connect the supply voltage and the output nodes of the latch, with gates controlled by the signal Precharge_Re_int_b. The latch is implemented with high-threshold voltage n-MOSFET p-MOSFET. The output nodes of the latch are then buffered with inverter circuits.
- The output stage takes as input one of the buffered outputs of the DSA. This signal is input to a level-shifter circuit that is supplied with VDD_Re. The output of the level shifter circuit is the input provided to the in/out scan chain if the Set_Parallel_Re and Sense_Re signals are high.

The operation of the DSA occurs in three phases. First, the sense amplifier is disconnected from the two SL_{Re}, i.e. the command Cmd_Sense_Re_int_b is low, and the output nodes of the sense latch are connected to VDDC, i.e. the Precharge_Re_int_b signal is low. During this time the input can be provided to the sense amplifier. Then, the sense amplifier is connected to the two SL_{Re}, i.e. the Cmd_Sense_Re_int_b becomes high. While Cmd_Sense_Re_int_b is high and Precharge_Re_int_b is low, the parasitic SL_{Re} capacitors are partially charged, at different speeds, depending on the resistance of the resistive memory devices. Finally, the Precharge_Re_int_b signal is switched and the latch performs the comparison between the SL_{Re} voltage values, toggling in the VDDC/Ground or Ground/VDDC configuration. Figures 4.13a and 4.13b show the weight encoding in the 2T-2R structure and the transient simulations of one DSA circuit, performing the four possible operations between inputs and weights. During the second phase, a direct path is created between VDDC and ground, resulting in static power consumption. By decreasing the time during which the SL_{Re} parasitic capacitors are charged, the energy consumption can be reduced. SPICE Monte Carlo simulations in the nominal design corner were performed to evaluate the energy consumption of the designed sensing element, considering a fixed LCS of 50 µS and a HCS of 100 µS (HCS/LCS ratio of 2) and SL_{Re} parasitic capacitance of 640 fF at supply voltage of 0.8 V. For a phase two duration of 15 ns, the average sensing element energy consumption is 153 fJ with standard deviation of 8.8 fJ, and a yield estimate of 100% evaluated over 200 runs. Decreasing the phase two delay to 10 ns results in an average energy consumption of 111 fJ and standard deviation of 6.4 fJ, with 98.5% yield. Decreasing the SL_{Re} parasitic capacitance and increasing the HCS/LCS ratio allows to decrease the phase two delay, resulting in reduced average energy consumption down to few tens of fJ. Indeed, the assumption of a HCS/LCS ratio of 2 is quite conservative and typically larger on/off ratios are observed for filamentary resistive memories. For instance, considering an HCS/LCS ratio of 10 (125 µS for the HCS and 12.5 µS for the LCS) and a source line parasitic capacitance of 384 fF – evaluated from the process design kit (PDK) specifications for a SL in the GO1 RRAM array – the phase two duration can be reduced to 4 ns for a reliable sense operation.

This results in an average energy consumption of 50.7 fJ per binary multiplication, with a standard deviation of 2.3 fJ and perfect yield evaluated over 200 runs.
A similar sensing circuit was proposed in [123] and [202] for the implementation of the MVM in binarized and ternary neural networks, respectively. In this implementation, the isolation stage is removed and the differential sense amplifier is directly connected to the resistive devices. In this strategy, the sensing is performed in two phases. First, the lines parasitic capacitors are pre-charged to the supply voltage of the sense. Then, they are discharged at different rates according to the resistance of the resistive memory devices, toggling the latch outputs [203]. This implementation potentially has zero static power consumption, but it requires to charge the parasitic capacitors to the supply voltage, each time a sense operation is performed. Depending on the size of the array, the parasitic capacitors can be relatively large, resulting in increased dynamic power consumption. The proposed DSA partially removes this limitation by charging the parasitic capacitors sufficiently enough to detect the difference between the two resistance values, using a charging time sufficiently long.

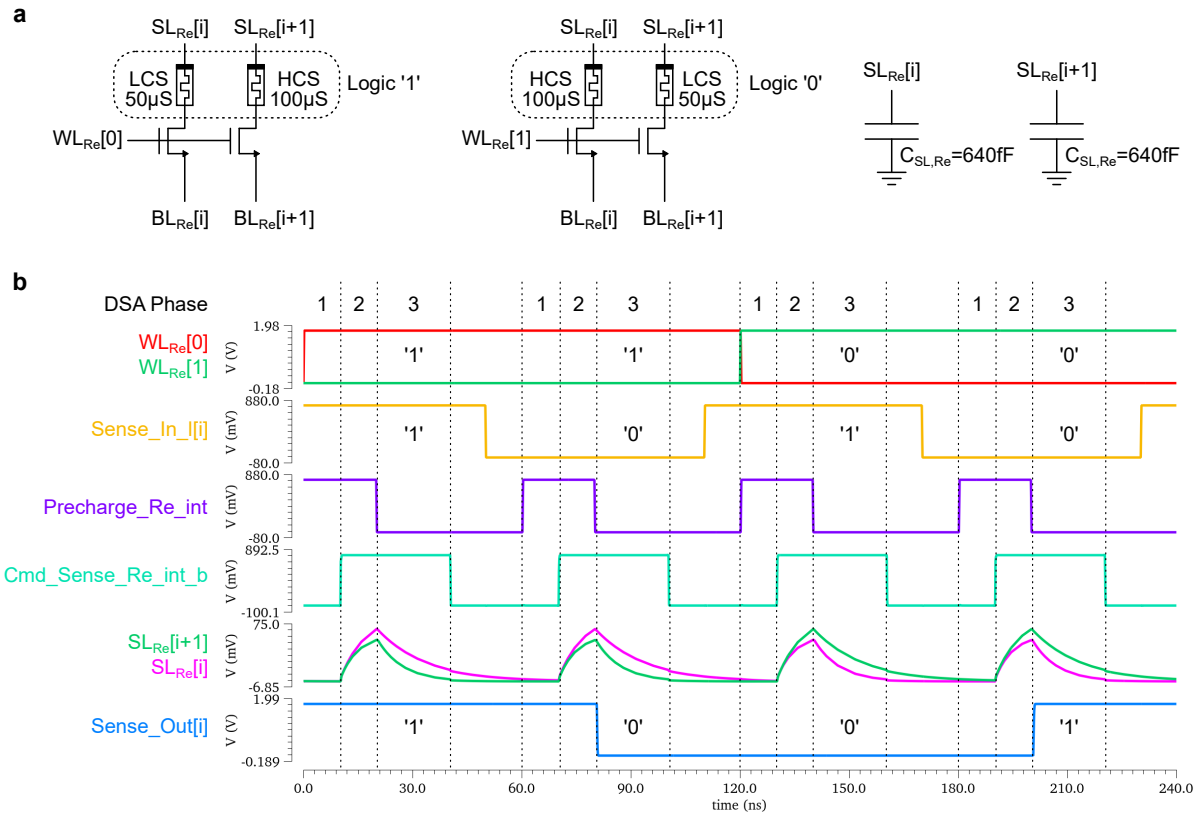

Figure 4.13: **Electrical simulations of the differential sense amplifier with in-place XNOR operation. a** Two WL_{Re} encoding the logic states 1 and 0 in differential conductance configurations are simulated. The SL_{Re} parasitic capacitance is set to 640 fF. **b** Transient simulations of the DSA circuit. The DSA operation occurs in three phases. The XNOR operation is simulated for the four different input (yellow trace) and weight (green and red traces) configuration.

Layout view. The layout view of each designed RRAM array is shown in Figure 4.14. The different blocks are highlighted in green boxes.

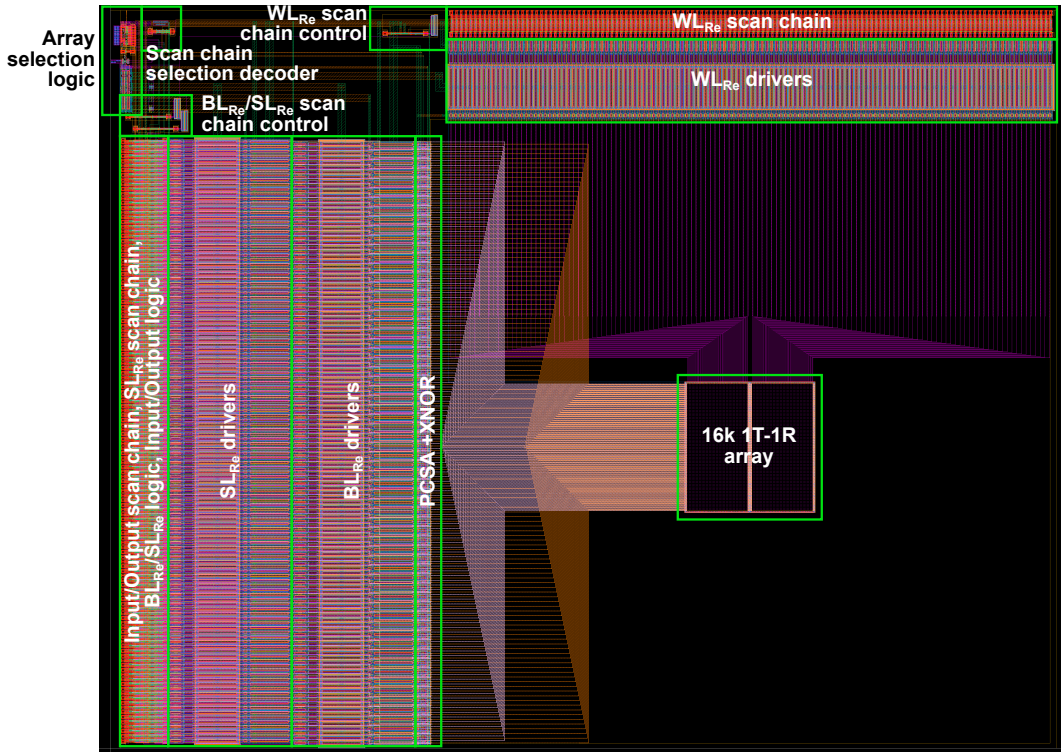

Figure 4.14: **Layout view of RRAM arrays in the FeЯNet design.** The different blocks in the layout are highlighted in green boxes.

4.2.3 Transfer logic

Transfer logic circuits were designed in order to optimize the communication between FeRAM and RRAM arrays. In particular, the transfer logic enables to transfer the signs of the hidden weights from the FeRAM arrays to the RRAM array in each sub-core. The signs in FeRAM arrays are stored in a 1T-1C cell, whereas binary weights in RRAM arrays are stored in 2T-2R cells encoding information in a differential configuration. The purpose of the transfer logic block is to set a logic one in the correct registers of the RRAM in/out scan chain, for programming 1T-1R devices accordingly.
A full word line from one out of four FeRAM arrays is transferred to the RRAM array, resulting in the parallel transfer of 16 signs (for 8-bit hidden weights) each time. The full transfer logic block takes as input data in pre-defined positions in the output scan chains of the four FeRAM arrays (i.e. the output of 16 out of 128 registers for each array) and outputs logic values to be stored in the 128 registers of the RRAM in/out scan chain. Only the registers of the RRAM in/out scan chain corresponding to the RRAM array section related to the active FeRAM array are modified according to the elements of the FeRAM output scan chain. The remaining registers in the RRAM in/out scan chain are set to zero by the transfer logic. The full transfer logic block is made of four sub-blocks, corresponding to the four FeRAM arrays. Each sub-block takes as input the outputs from the FeRAM output scan chain registers, the respective FeRAM array enable signal and a trigger signal enabling the transfer. Each sub-block is made of 16 equal instances, allowing the transfer from one FeRAM output scan chain register to two RRAM in/out scan chain registers, as shown in Figure 4.15. Each transfer logic block is supplied by the supply voltage VDD_Re.

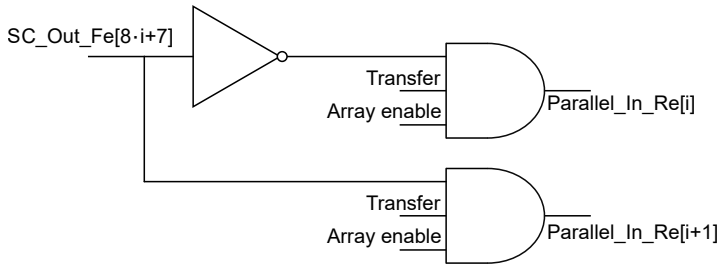

Figure 4.15: **Schematic of a transfer logic instance in the FeЯNet design.** Each transfer logic instance takes as input one element of the output scan chain of a FeRAM array, the array enable signal for said FeRAM array and the Transfer signal (pad 2, Table D1). Each transfer logic instance outputs two complementary logic signals.

An example is provided for a single instance of the transfer logic circuit, illustrating the operations occurring during transfer. If a 0 is written into a 1T-1C cell, storing the sign of a hidden weight, a read operation results in a logic 0 being transferred in one of the FeRAM output scan chain elements. A logic 0 in the differential RRAM unit cell is encoded in the HCS / LCS configuration for resistive devices in SL_{Re}[i] / SL_{Re}[i+1], with i index of an even SL_{Re}. Then:

1. Before transferring the sign between FeRAM and RRAM scan chains, both resistive devices in the 2T-2R unit cell are programmed to LCS.
2. The transfer logic, taking as input the logic 0 in the FeRAM output scan chain, outputs a 1, provided as input to RRAM in/out scan chain element corresponding to SL_{Re}[i], and a zero, provided as input to RRAM in/out scan chain register corresponding to SL_{Re}[i+1].
3. Finally a set operation is initiated for the RRAM array, effectively setting the resistive device in SL_{Re}[i], since a 1 is inserted in the respective scan chain element, and leaving the resistive device in SL_{Re}[i+1] in the LCS.

Vice-versa, if a 1 is written to a 1T-1C cell that stores the sign of a hidden weight, the transfer results in setting the RRAM unit cell in the LCS / HCS configuration, encoding a logic 1.
The proposed transfer circuits allow up to sixteen sign bits to be transferred from FeRAMs to RRAM arrays, enabling the parallel programming of up to sixteen RRAM devices simultaneously. However, programming multiple resistive memory devices in parallel may strain the current delivery capability of an embedded system, potentially reducing the programming current for each device. Ref. [123] evaluated the impact of bit errors on fully-connected and convolutional binarized neural networks in the MNIST and CIFAR-10 classification tasks, respectively, demonstrating negligible accuracy loss for bit error rates up to 10^{-3}. The resilience of BNNs to such errors, combined with the enhanced reliability provided by differential encoding in the 2T-2R cell, allows for programming resistive devices under weaker conditions, while also improving the endurance of RRAM devices. This could be achieved in the FeЯNet design by properly decreasing the compliance current during the parallel set operation, using a smaller word line voltage.

4.3 Characterization support for FeЯNet

A printed circuit board was designed and manufactured by an external company to test the FeЯNet design. Support in defining the hardware and software specifications and requirements was provided.
Besides the PCB, the provided test system includes a socket to install the packaged chips and a field-programmable gate array (FPGA) to manage the PCB elements and the digital inputs/outputs to the chip. The socket and FPGA are mounted on the PCB. The elements designed on the PCB are:

- Low-dropout regulator circuits and buffers for the power supplies;
- Digital-to-analog converters to tune the power supplies;
- Analog-to-digital converters to measure the current and voltages provided by the power supplies;
- A resistance measurement circuit to evaluate the resistance of RRAM devices via the pads BL_Probe_Re and SL_Probe_Re;

The FPGA manages the communication between the elements on the PCB in order to power on the supplies, set specific voltage levels for the supplies, collect data from ADCs, enable the resistance evaluation circuit and collect the results. Moreover, the FPGA enables the communication with the chip via the thirty-three digital input/output signals. Digital inputs of the chip are classified as either pseudo-static or dynamic inputs. Pseudo-static inputs are inputs that need to be set to a given value, which is fixed until a new value is set. Pseudo-static signals need to be set to either 0 or 1 manually without any specific timing requirement. On the other hand, dynamic inputs need to be controlled with logic sequences requiring precise timing between each other. Digital outputs are dynamic signals. All commands are sent to the FPGA via a python interface.

4.4 Current progress and future outlook

The FeЯNet circuit was manufactured with a purely RRAM BEOL process and subsequently packaged for validation with the produced test PCB. Preliminary measurements were conducted to validate the functionality of the test setup and the communication with the chip. As an initial test, the ability to push data into a specific scan chain of the circuit and retrieve the input data at the output pad was verified. In particular, the following test protocol was implemented to confirm proper communication with the word line scan chain of one RRAM array:

1. Power on the pad ring and core supplies (VDDIO = 1,8 V, VDDC = 0,8 V).
2. Power on all other supplies (VDD_Re, VDD_WL_Re, VDD_BL_SL_Re, VDD_WL_Fe, VDD_SL_Fe, VDD_ Fe = 1,8 V).

3. Set pseudo-static digital inputs (unspecified signals are set to 0), in the following order:
   (a) Rst_BL_Re ← 1 (Activate reset of the BL_{Re} scan chain.)
   (b) Rst_SL_Re ← 1 (Activate reset of the SL_{Re} scan chain.)
   (c) Rst_WL_Re ← 1 (Activate reset of the WL_{Re} scan chain.)
   (d) Rst_BL_Re ← 0 (Deactivate reset of the BL_{Re} scan chain.)
   (e) Rst_SL_Re ← 0 (Deactivate reset of the SL_{Re} scan chain.)
   (f) Rst_WL_Re ← 0 (Deactivate reset of the WL_{Re} scan chain.)
   (g) SC_Sel_Re[0] ← 1 (Select the WL_{Re} scan chain.)
4. Send a dynamic sequence of digital inputs (unspecified signals are set to 0) to write a random pattern in the selected scan chain and collect digital output. In particular:
   (a) Send a random pattern to SC_In_Re (common input pad of the scan chains of the RRAM array) for the at least first 128 clock cycles of the scan chain.
   (b) Input the scan chain clock signal (i.e. 010101010...) to Ck_SC_Re to propagate the input sent to SC_In_Re within the scan chain.
   (c) At the output pad SC_Out_Re, the signal remains low for the firs 128 clock cycles of the scan chain, since it was initially reset. Then, the random pattern input at SC_In_Re is retrieved.

The results of this test are shown in Figure 4.16. For these measurements, a logic analyzer was connected to the test PCB to visualize the signal waveforms. It can be observed that the SC_Out_Re signal (blue) replicates the pattern of the SC_In_Re signal (red) after 128 clock cycles of the scan chain (yellow). This result partially validates the functionality of both the test PCB and the manufactured chip. Further tests will be necessary to validate the functionality of the RRAM arrays in the FeЯNet design.

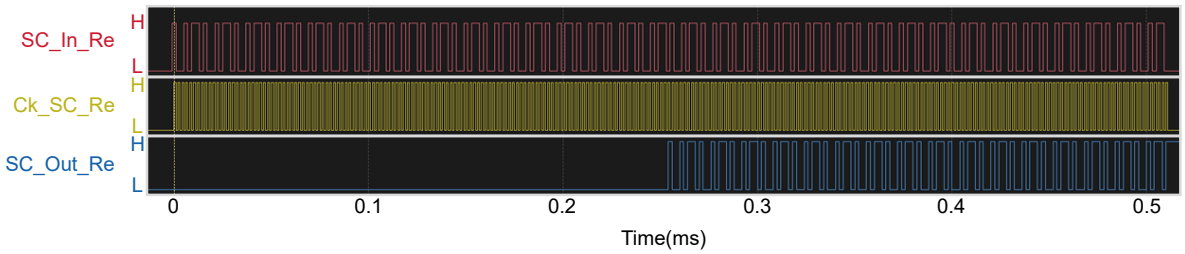

Figure 4.16: **Experimental validation of a scan chain in the FeЯNet chip.** "L" and "H" represent the low and high voltage levels for the three signals, respectively.

Additionally, a partial manufacturing flow up to the fifth metal layer was implemented using a purely FeRAM BEOL process. Testing of the double-scribe version of the FeЯNet design will help validate the functionality of the FeRAM arrays.
The manufacturing flow that enables the co-integration of ferroelectric and resistive memories, described in section 2.5.2, is currently under development and is expected to be completed by 2025. Once the manufacturing flow is finalized, the complete design will be tested either using the double-scribe version or the test PCB.

The test of the designed chip is still an ongoing work that will be carried on in the future. Beyond validating the current design of the FeЯNet chip, further development is necessary to evaluate the efficiency of the proposed system. For example, the current design includes the near-memory sensing elements to perform the bit-wise multiplication, but it still lacks the circuits to perform the accumulation. In BNNs, accumulation can be efficiently implemented using digital popcount circuits, which may be realized with adder trees or counter circuits, depending on the trade-offs between area, power, and time efficiency. Moreover, further work at software level will point out the need for ulterior circuits for the implementation of a complete neural network system. Finally, it should be noted that the designed periphery still require further optimization. In particular, in a neural network system a different addressing strategy will likely be necessary, substituting the scan chains with decoding circuits, as well as optimized driver circuits.

4.4.1 System use cases

From an application standpoint, the FeЯNet circuit was designed to enable on-chip training of binarized neural networks , where both weights and activations are constrained to 1-bit precision. Beyond vanilla BNN training, various strategies have emerged that leverage BNNs to tackle a wide range of AI challenges.
One such approach highlights the potential of BNNs for implementing lifelong learning. Specifically, it has been demonstrated that by introducing a minor modification to the standard stochastic gradient descent weight update rule, the hidden weights within BNNs can act as metaplastic variables. Under this framework, the absolute value of a hidden weight serves as an indicator of the consolidation strength of its corresponding binary weight. A larger absolute value implies stronger consolidation, making the synapse less likely to flip to the opposite sign. This behaviour is implemented by scaling the weight update using a metaplastic function, which is parameterized by a scalar value and depends on the hidden weight's magnitude. The chosen function ensures that the likelihood of a binary weight switch decreases exponentially as the hidden weight increases, while the switching behaviour remains unaffected when the hidden weight is close to zero. A key advantage of this method is its flexibility, as it eliminates the need for explicitly partitioning tasks, thus enabling more continuous learning of a single task. This technique has been benchmarked across several datasets and continual learning scenarios, including multitask and stream learning, and has proven effective in mitigating catastrophic forgetting [51]. However, the current training strategy relies on full-precision hidden weights and employs mini-batch learning with batch normalization. Optimizing this approach in conjunction with the FeЯNet circuit could pave the way for a continual learning solution that is better aligned with the hardware constraints of embedded devices.
This approach draws inspiration from the synapse cascade model, initially proposed in [204]. In this model, each synapse is characterized by a synaptic strength, equivalent to its synaptic weight. Furthermore, each synaptic strength is represented by a cascade of n states, reflecting the metaplasticity of the synaptic weight. In the model, a synapse that undergoes repeated potentiation does not simply increase its weight. Instead, it becomes more resistant to subsequent depression. The cascade levels are discrete, and the probability of a synapse shifting to the opposite strength decreases exponentially with increasing depth in the plasticity levels. This exponential scaling introduces a broad spectrum of transition rates: fast synapses at the top of the cascade transition easily, while slower

synapses deeper in the cascade are less likely to switch. Additionally, the study demonstrates that the memory lifetime of a collection of synapses—defined by the number of candidate plastic events before the signal-to-noise ratio of the memory trace drops below one—depends on both the number of synapses and the optimal number of cascade states. The optimal memory lifetime is directly influenced by these two factors. The process of transitioning between states with a given probability recalls the weight update strategy proposed in chapter 3. Drawing inspiration from [204] and adapting the weight update rule proposed in the previous chapter – originally designed to improve convergence in quantized neural networks with integer hidden weights – to include update probabilities based on the hidden state of the synaptic strength, it is likely possible to mitigate the effects of catastrophic forgetting, even in binarized neural networks with discrete integer hidden weights. A similar approach was recently explored in [205, 206].
The success of BNNs relies on the possibility of training them with gradient descent methods, despite it being a discrete optimization problem. This was originally possible thanks to the introduction of the STE technique. Nevertheless, the reason why this is the case is not entirely clear. It has been recently proven that the Bayesian learning rule, when applied to estimate a Bernoulli distribution over the binary weights, results in an algorithm which justifies some of the algorithmic choices made by the previous approaches, the STE in particular, starting from principled assumptions. This algorithm not only obtains state-of-the-art performance but, as a Bayesian method, it enables uncertainty estimation, which can be useful for decision making and for continual learning [207, 208]. The FeЯNet circuit could provide a platform to investigate and evaluate these algorithms taking into account the constraints of embedded electronics.

4.5 Summary

This chapter presented the design of a circuit aimed at investigating the advantages of combining ferroelectric (Fe) and resistive (Re) memory devices, embedded in the BEOL of foundry 22 nm FDSOI technology, for the hardware implementation of inference and training in BNNs, referred to as FeЯNet.
The FeЯNet design uses FeRAM arrays to store higher-precision hidden weights in either integer or floating-point formats, with bit-widths of 8-bits or multiples thereof. Binarized weights are stored in a differential configuration across two resistive memory devices. Transfer logic circuits enable the parallel transfer of up to sixteen sign bits from FeRAMs to RRAMs, speeding up communication between the two arrays.
The RRAM arrays are equipped with sensing elements that perform XNOR operations between the binarized weights stored in RRAM devices and the inputs presented to the sensing elements. This operation, combined with population count circuits, allows the evaluation of matrix-vector multiplications in binarized neural networks, accelerating and improving energy efficiency during the forward and backward passes of BNN training. Monte Carlo electrical simulations of the designed sensing circuit suggest the potential to achieve binary multiplication energy of approximately 100 fJ, with further scaling down to a few tens of fJ possible with optimized line design.
The synaptic capacity, or number of weights that can be stored in the FeЯNet design, depends on the desired precision of the hidden weights. Maximum capacity is achieved with 8-bit hidden weights, resulting in 16,000 synaptic weights; as the bit-width of the

hidden weights doubles, the number of synapses is halved.
Preliminary measurements of the FeЯNet chip partially validate the functionality of the developed test PCB and the FeЯNet design. Further testing is required to fully validate the functionality of the various blocks. Several algorithmic tracks are envisioned for implementation using the FeЯNet design, which would enable the development of learning-capable devices for edge applications.

Acknowledgments

In relation to the work detailed in this chapter, I would like to acknowledge Olivier Billoint and his team for developing the initial design of independent ferroelectric and resistive memory arrays, which served as the foundational building block for the FeЯNet design. I would also like to thank Sylvain Dumas for his contribution to the development of the PCB test support used for the characterization of the FeЯNet chip.

Chapter 5

Conclusion

Summary

This thesis deals with the hardware implementation of artificial neural networks with non-volatile memory devices. Among NVM solutions, memristors are particularly promising to enable next-generation IMC hardware, because they potentially offer improved energy efficiency, faster access times and larger density compared to traditional storage devices. Nevertheless they suffer from several device non-idealities, limiting their overall reliability at present. Although state of the art memristive technologies have proven near software-equivalent inference accuracy in many AI domains, their exploitation in learning-capable systems remains challenging. Indeed, memory requirements for training are more demanding, as learning processes in ANNs requires accurate iterative refinement of the synaptic strengths in the network.
RRAM, also known as filamentary memristor, and FeRAM devices appear as suitable candidates to enable on-chip learning systems. The virtually unlimited read endurance of RRAMs and their poor write endurance makes them suitable for inference-only applications, whereas the reported large write endurance of FeRAMs would effectively allow to move training on-chip as well. The manufacturing of two NVM technologies on the same substrate, although beneficial for the hardware implementation of ANNs, comes at the expense of a large manufacturing cost and complex integration. On the other hand, the same material stack can be optimized to work as FeRAM or RRAM under different operating conditions, in the case of HfO_2-based devices.

In this thesis, this was achieved by integrating a 10 nm silicon-doped hafnium oxide film with a titanium scavenging layer between two metal lines of a 130 nm foundry CMOS process. This stack combines an active layer of hafnium oxide crystallized in the orthorombic phase – necessary for ferroelectric switching – with a scavenging layer – necessary for reliable resistive switching. The hybrid memory was tested in two configurations: in FeRAM arrays and in RRAM arrays manufactured in the back-end of line of 130 nm CMOS foundry technology. As FeRAMs, such devices were shown to work as binary memories with good endurance over 10 million cycles and low programming energy, below 200 fJ/bit. After undergoing a forming process to create conductive filaments, the same devices integrated in the BEOL of RRAM arrays, can be used as analog multi-level memory devices with lower endurance, about 100,000 cycles. These results highlights

the potential of this hybrid approach to leverage the strengths of both memory types for artificial intelligence workloads.

An application-specific integrated circuit leveraging the developed hybrid memory technology was experimentally validated. This circuits includes both ferroelectric capacitors and memristors within the same BEOL of foundry 130 nm CMOS, in order to create an hybrid array of interconnected FeCAPs and memristors. The fundamental unit of the hybrid array is an hybrid synapse circuit, consisting of a collection of one-transistor-one-FeCAP cells, where the bit line of each cell is directly connected to the gate of the selection transistors in a one-transistor-one-memristor cell. This sub-circuit enables direct digital-to-analog data transfer from multiple FeCAP cells to a single memristive device, without the need for intermediate circuitry [182]. This technology is compatible with on-chip training of ANNs. FeCAP devices store higher-precision hidden weights that undergo numerous programming operations during training, while memristor devices store analog weights read during both inference and training. Two system implementations were proposed. The first uses equally sized FeCAPs to store the higher-precision weights and a single memristor to store the analog weight. This approach limits the precision of the hidden weights to n+1 discrete states, where n is the number of FeCAPs in the synaptic circuit. A binarized neural network was trained on the ECG-arrhythmia detection task. Taking into account the constraints of the synaptic circuit, the network achieves 88% accuracy, with eight FeCAPs only storing each higher-precision weight and one memristor used to store the binary weight [183]. The second implementation uses ferroelectric capacitors with different areas to encode the 10-bit integer higher-precision weights in an sign-and-magnitude format and two memristors to encode positive and negative analog weights in the memristors' differential conductance. Fully-connected ANNs were trained using a stochastic gradient descent algorithm. In particular, for each training sample, neuron activations are calculated by feed-forward matrix-vector multiplication between the analog weights (memristors) and the previous layer's activations. Errors at the output layer are back-propagated to evaluate loss gradients and update hidden weights. Hidden weights (10-bit integers in FeCAPs) are updated for each sample, while analog weights (memristors) are updated every k inputs via the digital-to-analog transfer procedure. On the MNIST dataset, at k=100, the method achieves 96.7% accuracy with approximately 38nJ total programming energy consumption per weight, representing a 38-fold reduction with no loss of accuracy with respect to k=1. The number of programming operations remains 17 times below the memristor endurance limit and 75 times below the FeCAP limit. Moreover, the robustness to memory errors was also assessed based on measurements of analog transfer from FeCAPs to memristors. The results obtained across a variety of edge benchmarks (ECG arrhythmia binary detection, image classification on MNSIT and Fashion MNIST datasets) are competitive with those achieved by floating-point precision software models, without the endurance limitations associated with hardware constraints. Finally, the proposed training approach was benchmarked on a transfer-learning task. To create a transfer-learning scenario, an edge-friendly neural network, MobileNet-V2, was pre-trained on the CIFAR-100 dataset. The convolutional layers were used as as a fixed feature extractor, and a fully connected layer was added, which was trained on the CIFAR-10 dataset using the online learning strategy adapted to the hybrid memory circuit constraints. The online transfer learning only reduces accuracy by approximately two percentage points, to 88.0%, compared to the full-precision baseline, confirming that this approach performs well even with sophisticated datasets.

A second circuit was developed with the aim of investigating the advantages of combining ferroelectric (Fe) and resistive (Re) memory devices, embedded in the BEOL of foundry 22 nm FDSOI technology, for the hardware implementation of inference and training in binarized neural networks, referred to as FeЯNet. The FeЯNet design uses FeRAM arrays to store higher-precision hidden weights in either integer or floating-point formats, with bit-widths of 8-bits or multiples thereof. Binarized weights are stored in a differential configuration across two resistive memory devices. Transfer logic circuits enable the parallel transfer of up to sixteen sign bits from FeRAMs to RRAMs, speeding up communication between the two arrays. The RRAM arrays are equipped with sensing elements that perform XNOR operations between the binarized weights stored in RRAM devices and the inputs presented to the sensing elements. This operation, combined with population count circuits, allows the evaluation of matrix-vector multiplications in binarized neural networks, accelerating and improving energy efficiency during the forward and backward passes of BNN training. Monte Carlo electrical simulations of the designed sensing circuit suggest the potential to achieve binary multiplication energy of approximately 100 fJ, with further scaling down to a few tens of fJ possible with optimized line design. Preliminary measurements of the FeЯNet chip partially validate the functionality of the developed test PCB and the FeЯNet design. Further testing is required to fully validate the functionality of the various blocks. Several algorithmic tracks are envisioned for implementation using the FeЯNet design, which would enable the development of learning-capable devices for edge applications.

Future outlook

Combining ferroelectric and resistive switching within the same memory stack is advantageous in terms of process technology, as it does not require any additional cost with respect to integrating a single memory device in the BEOL of CMOS. In this case, differentiation from a ferroelectric to a resistive memory operation can occur by means of an irreversible electro-forming process. Additionally, to facilitate scaling down to more advanced CMOS technology nodes, a slightly modified BEOL integration is proposed for ferroelectric and resistive devices, allowing for lower forming voltages. This modification further differentiates the two technologies at the manufacturing level [177]. Both these approaches rely on the assumption that two physically separated sets of devices should be embedded in the system: one functioning as a ferroelectric device and the other as a resistive device. At system level this means that ferroelectric and resistive memory arrays are physically apart, not sharing any programming or reading circuitry. However, this is not the only available option.
A unified array of memory elements could be conceived, with a unified memory stack integrated in the BEOL, where the programming and reading circuitry is designed in order to function for both ferroelectric and resistive memory operation. Indeed, in terms of write operations, FeRAM and RRAM technologies require generally similar circuits. To increase or decrease the conductance value in RRAMs, voltage pulses must be applied to the SL or BL. Similarly, to reset or set a FeRAM device. Therefore, RRAM drivers could also be repurposed to polarize FeRAM devices. Nevertheless, as explained in chapter 2, the current levels involved during programming are notably different for RRAM and FeRAM technologies. Hence, the significantly higher write currents required for RRAM would require larger transistors, for programming and selection, compared to a purely

FeRAM implementation, resulting in a relatively less dense array. The sense circuitry could also be shared in this scenario. For instance, the clocked latch sense amplifier in Figure 2.3 developed for FeRAM arrays could be used to evaluate the the time constant when charging the BL with the application of a voltage pulse on the source line, allowing to distinguish a RRAM device in a set or reset state [209]. The approach of co-locating FeRAM and RRAM operation within the same array could potentially allow globally denser solution in the case of a learning-capable systems. For instance, in the case of back-propagation-like algorithms, it would not require having separate devices to perform the forward/backward steps and the weight update one, as proposed instead in this thesis. Nevertheless, this approach appears more suited to a fully-digital implementation, where FeRAM and RRAM devices are used to store weights in a digital format [181]. A system, initially trained with the array in FeRAM mode, could be switched to an RRAM mode to perform inference-only. Going back to a FeRAM operation could allow to perform other learning cycles throughout the lifetime of the system. This approach would require a unified memory stack with optimal performance in both FeRAM and RRAM modes to fully leverage the potential of the systems. Nevertheless, as discussed in chapter 2, optimizing ferroelectric and resistive properties simultaneously in HfO_2-based devices presents a challenge, as it requires careful engineering of the oxygen vacancy profile in the memory stack. On one hand, reducing oxygen vacancies generation improves ferroelectric performance by boosting device endurance, while on the other hand, enhancing vacancies generation is crucial for lowering the forming voltage and achieving reliable resistive switching. Although combined optimization might be difficult to achieve, a satisfactory compromise between ferroelectric and resistive performance is certainly possible. Moreover, further study at device level is necessary to explore the possibility to go back to a ferroelectric operation after an electro-forming step. Figure 2.10 showed the possibility to program devices exploiting the unified memory stack in a conductance state comparable to the pristine state. This operation could allow to retrieve the ferroelectric operation after an electro-forming step. Still, further characterization is necessary to explore this behaviour. This thesis marked a first step towards the development of this hybrid memory technology for the implementation of embedded AI systems. The circuit implementation described in chapter 3 allowed to tackle some key issues of embedding back-propagation on-chip exploiting emerging non-volatile memory devices. Dealing with imperfect devices and finding strategies to overcome their limitations is a necessity is this context. Sometimes, taking advantage of these limitations is possible and beneficial for the implementation of more exotic learning strategies [210]. The circuit implementation described in chapter 4 has the potential to further explore some of these algorithms while taking advantage of a technology that is inherently well-suited to learning systems and showing promise for further optimization.

More fundamentally, further study at the algorithmic level is necessary to unlock the true potential of edge AI. This thesis relies solely on supervised learning; however, edge devices are typically interfaced with an unlabelled world. To develop more adaptive, learning-capable systems, it is crucial to explore unsupervised or semi-supervised learning approaches, enabling edge AI to learn autonomously from its environment. Additionally, edge AI systems need the ability to learn new types of data without forgetting previously acquired knowledge, a critical feature for lifelong learning in dynamic environments. Moreover, given the limited computational power of individual edge devices, distributed intelligence across multiple devices could offer a promising solution. By sharing resources and processing tasks collaboratively, these systems could handle more complex tasks that

would otherwise be beyond the capabilities of a single device. Ultimately, distributed intelligence would help realize a more efficient, connected world where edge AI systems work together to process data in real time, driving smarter decision-making and enhanced user experiences.

Appendix A

Electrical characterization setup

This appendix specifies the experimental setup used to perform the electrical characterization on single device structures, ferroelectric and resistive memory arrays, as well as the hybrid memory circuit based on the unified memory stack, presented in chapter 2 and chapter 3.

PUND measurements on ferroelectric capacitors and quasi-static characterization of memristors (Figure 2.6) were performed using signals generated by a Keysight B1530 waveform generator/fast measurement unit module and analyzed by a Keysight B1500A semiconductor parameter analyzer, which is controlled by a Python-coded interface.

The same setup was used to characterize FeCAP arrays (Figure 2.7, Figure 3.4a,b and Figure 3.6d). The Keysight B1530 module was connected to a custom-made PCB with a co-integrated Arduino microcontroller for signal and instruction management.

To program and read the memristor arrays (Figure 2.8, Figure 2.9, Figure 2.10), voltage pulses were produced externally using a RIFLE NplusT engineering test system. This system is equipped with a 100 MHz arbitrary waveform generator and a C++ programmable computer. The computer controlled the pulses generated by the arbitrary waveform generator, which were applied to the memristor arrays.

The hybrid memory array characterization(Figure 3.4c and Figure 3.7) was performed via external power supplies (Rohde & Schwarz HMP4040, Keysight E3631A), a pulse pattern generator (Anritsu MP1763C), an arbitrary waveform generator (Tektronix AFG1062), and a digital multimeter (Keysight 34470A). The instruments are managed by a Python interface, and communication with the computer occurs via GPIB connections.

All signals were connected to a 200 mm wafers through a 25-pin probe card, interfacing with 25 metal pads.

Appendix B

Unified ferroelectric/memristive memory circuit design

This appendix describes the design of the manufactured hybrid memory circuit presented in chapter 3. The building blocks of the design are shown in Figure 3.3. This circuit was designed in a 130 nm foundry CMOS technology node up to metal layer four. The non-volatile memory stack and the fifth, last, metal layer were fabricated in CEA LETI. The circuit can be accessed by 25 pads on the last metal layer. Table B5 describes the pad listing to communicate with the circuit. The design developed in this Thesis is based on a previously validated design including an array of FeCAPs, in a 1T-1C configuration, and its peripheral circuitry. This design was modified in order to add the memristors array, in a 1T-1R configuration, so to obtain the array of hybrid memory circuits.

B.1 Design specifications

In the following, the various blocks composing the whole test scribe design will be detailed in the following order: Scan chain system, FeCAPs array source line (SL_{Fe}) and word line (WL_{Fe}) drivers, memristors array bit line (BL_{mem}) and source line (SL_{mem}) drivers, FeCAPs sense and transfer line drivers, and timing block.

B.1.1 Scan chain system

The inputs to the array drivers, defining the active and inactive lines of the circuit, as well as the inputs to the timing block, describing its operating mode and the pulse properties, are provided to the circuit via scan chains. As shown in the pad listing, data introduced in the scan chains are input with the SC_IN pad (pad 20) and output via the SC_OUT pad (pad 21). Therefore, a single scan chain at the time can be used. In order to select the active scan chain, a 3-bits input signal is provided, SC_SEL. The three bits (SC_SEL[2], SC_SEL[1], SC_SEL[0]) are independently provided with three different pads (pads 22, 23 and 24). The SC_SEL signal is decoded with a 3-to-8 decoder in order to select the desired scan chain. A schematic of the scan chain system is shown in Figure B1.

The active scan chains according to the SC_SEL signal configurations are described in Table B1. The five scan chains have two common signals, that are the clock (CK_SC, pad 19) and the input (SC_IN, pad 20). The size of each scan chain, i.e. the number of flip flops composing it, is indicated in Figure B1. As the same inputs are provided to SL_{Fe} and WL_{Fe}, their respective scan chain share the same enabling signal. The same is done for SL_{mem} and BL_{mem} scan chains. All scan chains are supplied by VDD, pad 14. The ground is provided via pads 13/25, GND.

Active Scan Chain	**SC_SEL[2]**	**SC_SEL[1]**	**SC_SEL[0]**
SC_OUT	1	1	1
SC_CONTROL	1	1	0
SC_SL_Fe	1	0	1
SC_WL_Fe			
SC_TL	1	0	0
SC_SL_mem	0	1	1
SC_BL_mem			
Unused	0	1	0
Unused	0	0	1
Unused	0	0	0

Table B1: **Scan chain selection in the hybrid memory circuit.**

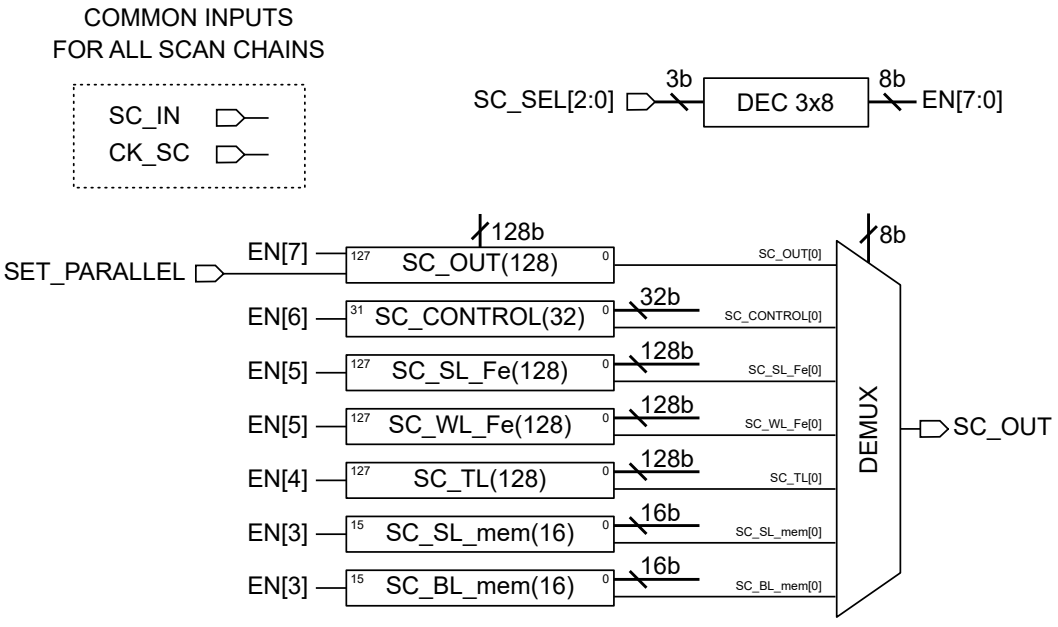

Figure B1: **Scan chain system in the hybrid memory circuit.**

B.1.2 FeRAM drivers

The WL_{Fe} and SL_{Fe} drivers are simply implemented as inverter gates, with command signals provided by the NAND operation of the value written in the respective scan

chain element and an external trigger signal (WL_Fe, pad 6, and SL_Fe, pad 5, respectively). The inverter and NAND gates are supplied by the VDD_WL_Fe (pad 15) and VDD_SL_Fe (pad 17) external supply voltages . Therefore, the WL_{Fe}/SL_{Fe} can be connected to VDD_WL_Fe/VDD_SL_Fe only if a 1 is inserted in the scan chain for a given WL_{Fe}/SL_{Fe} and the trigger signal is high. Otherwise, the WL_{Fe}/SL_{Fe} is grounded. The ground is provided via pads 13/25, GND. The WL_{Fe}/SL_{Fe} drivers operation is summarized in Table B2.

SC_WL_Fe[i] (SC_SL_Fe[i])	**WL_Fe (SL_Fe)**	**WL_{Fe} (SL_{Fe})**
1	1	VDD_WL_Fe (VDD_SL_Fe)
1	0	GND
0	1	GND
0	0	GND

Table B2: **WL_{Fe} and SL_{Fe} drivers operation in the hybrid memory circuit.** SC_WL_Fe[i] and SC_SL_Fe[i] are the elements written in the scan chains SC_WL_Fe and SC_SL_Fe in poition i. WL_Fe and SL_Fe are the trigger signals provided by pads 6 and 5 respectively.

B.1.3 RRAM drivers

The BL_{mem}/SL_{mem} drivers are implemented as transmission gates controlled by a command signal cmd and its negate version cmd_b. The command signal cmd for the BL_{mem}/SL_{mem} drivers is obtained as AND (NAND + INV) of the value inserted in the scan chain and an external trigger signal (WL_Fe/BL_mem, pad 6, for the BL_{mem} and SL_Fe/SL_mem, pad 5, for the SL_{mem}). The transmission gates are correctly sized in order to be able to provide a sufficiently high current during the memristor write operations. Therefore, if a 1 is inserted in a scan chain element of a certain BL_{mem}/SL_{mem} and the trigger signal is activated, the BL_{mem}/SL_{mem} are respectively connected to the external voltages VDD_BL_mem/VDD_SL_mem (pads 12 and 18). Let us notice that the same pads for the trigger signals used for the WL_{Fe} and SL_{Fe} drivers are used for BL_{mem} and SL_{mem} respectively. This pad sharing is not an issue and it does not require to empty the scan chains of one of the arrays since independent supply voltages are used for the different drivers. In particular if the FeCAPs array has to be programmed independently, the VDD_BL_mem/VDD_SL_mem voltages can be left at 0 V, thus not changing the programmed resistance values in the memristors array regardless of the values in the scan chains. On the other hand, if the memristors array has to be programmed independently, the supply voltages of the FeCAPs array drivers can be set to 0 V, thus not changing the programmed values in the FeCAPs array even if a trigger signal is sent to the WL_{Fe} and SL_{Fe}, regardless of the values in the scan chains. The hybrid operation mode, allowing the analog transfer of the values programmed into the FeCAPs to the memristor requires both FeCAPs and memristors drivers to be active, for the parallel reading of FeCAPs and writing of the memristor, respectively. In this case, the external trigger signal controls both the timing of the FeCAPs reading and the writing of the memristor at the same time. The BL_{mem}/SL_{mem} drivers operation is summarized in Table B3.

SC_BL_mem[i] (SC_SL_mem[i])	**BL_mem (SL_mem)**	**BL_{mem} (SL_{mem})**
1	1	VDD_BL_mem (VDD_SL_mem)
1	0	GND
0	1	GND
0	0	GND

Table B3: **BL_{mem} and SL_{mem} drivers operation in the hybrid memory circuit.** SC_BL_mem[i] and SC_SL_mem[i] are the elements written in the scan chains SC_BL_mem and SC_SL_mem in position i. BL_mem and SL_mem are the trigger signals provided by pads 6 and 5 respectively.

B.1.4 Sensing elements and transfer line drivers

The block related to the transfer lines (TLs) comprises both the TL drivers and the sense amplifiers for the FeCAPs array. A driver element and a sense amplifier are connected to each TL of the array. The TL block is directly connected to the TLs, TL scan chain and output scan chain. The supply voltage of the TL block is provided by pad 16, i.e. VDD_TL. The ground is provided via pads 13/25, GND. The TL block also requires a reference voltage for the sense operation, provided by pad 11, VREF.
The input signals to the TL block are:

- WB, i.e. the write-back trigger signal, provided by pad 3. This signal allows automatic write-back of the value stored in a FeCAP device after reading. This is possible as the output obtained during the reading operation stays unchanged unless the TLs are precharged to 0 V.
- PRE, i.e. the precharge trigger signal, provided by pad 8. This trigger signal allows to ground the TLs, if the sense amplifiers are not activated. Typically, it is necessary to set the TLs to 0 V before performing a read operation, to obtain a reliable read.
- SA, i.e. the sense amplifier trigger signal, allowing to connect or disconnect the sensing element to the TLs, provided by pad 4. The same sensing elements as in Figure 2.3 were used in this design. The signals en and enb in Figure 2.3 are the buffered and negate version of the input SA. The 2 inputs, $V_{in,a}$ and $V_{in,b}$ in Figure 2.3, are precharged to the TL voltage and the reference voltage respectively, if the sense amplifier is disabled.
- set_write: obtained as NAND operation between the write enable signal (WE, provided by pad 9) and the data written in the TL scan chain element (SC_TL[i]). The set_write signal is active-low and allows to connect the TL to the supply voltage VDD_TL.
- set_write_0: obtained as AND operation between mem_MODE, WE and the negate of SC_TL[i]. The signal mem_MODE is provided by pad 7. The signal set_write_0 is active-high and it allows to ground the TL if a zero is written in SC_TL[i] and the mem_MODE signal is high.

The operation of the TL block is summarized Table B4. Other signal combinations that differ from the ones specified in Table B4 could lead to short circuits. In particular, it is important to do not connect the TL to both VDD_TL and GND at the same time, by enabling together the set_write and PRE signals, as well as to do not allow the precharging of the TL while the sensing is enabled.

WE/TL	WB	PRE	SA	mem_MODE	SC_TL[i]	TL
1	0	0	0	0	1	VDD_TL
1	0	0	0	1	0	GND
0	0	1	0	0	*	GND
0	*	0	1	0	*	HZ
0	1	0	*	0	*	HZ

Table B4: **Typical operation of the TL block in the hybrid memory circuit.**

B.1.5 Timing block

The inputs to the timing block are provided by the control scan chain and two bias voltages respectively, i.e. PW_TUNE_ANA and PW_TUNE_ANA_D (pads 1 and 2 respectively), used to control the pulse and delay generators. The supply voltage VDD (pad 14) and ground connection GND (pads 13 and 25) are also provided to the timing block. The outputs are the trigger signals for the word and source lines, write back and sense amplifiers. The whole FeCAPs array can be driven in 3 different ways, with or without pulse generators. The enabling of the pulse generators are handled by the control scan chain. To sum-up, the 3 modes are:

- The "auto" mode, triggered by a single pad (WL_Fe), where pulse generators are successively activated in a specific order with internally controlled widths and delays.
- A "semi-auto" mode, where the pulse generators are enabled, but not the auto mode. In this configuration, each pulse generator can be separately triggered. Hence, the pulse widths are internally controlled, but the delays between each pulses and the order of the pulses is defined externally by the way the pads 3 to 6 are activated. If a pulse generator is enabled (1 in the correct position of the scan chain), the pulse will be triggered by the corresponding signal, e.g. the SL_{Fe} pulse generators pulse will be triggered by the pad 5 “SL_Fe”.
- An “external” mode, where both pulse generators and auto mode are not enabled. In this configuration, the signals sent to the enabling pads 3 to 6 are the ones seen by the array, i.e. the widths and delays are monitored directly by the software.

The detailed description defining the functionality of each configuration bit in the control scan chain is not reported here. It is available in internal documentation.

B.2 Layout view

The layout view of the designed circuit, with the description of the different blocks composing it is shown in Figure B2.

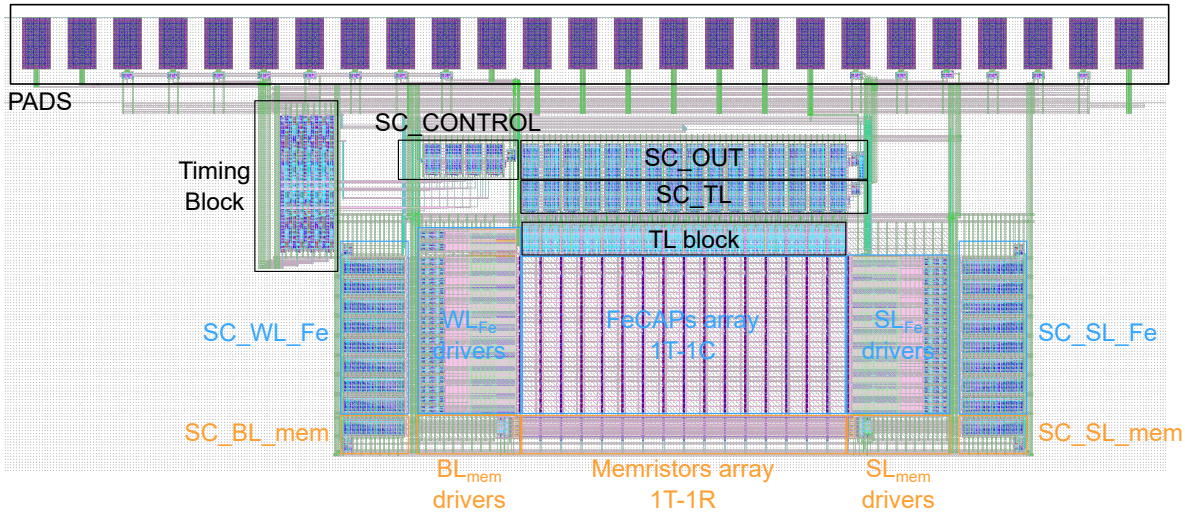

Figure B2: **Layout view of the designed hybrid memory circuit.** The leftmost pad in the layout corresponds to pad 1 in Table B5.

pad number	**pad name**	**Type**	**Description**
1	PW_TUNE_ANA	BIAS	Constant voltage bias for pulse/delay generator.
2	PW_TUNE_ANA_D	BIAS	Constant voltage bias for pulse/delay generator.
3	WB	DIGITAL IN	Write back for FeCAPs.
4	SA	DIGITAL IN	Sense enable for FeCAPs.
5	SL_Fe / SL_mem	DIGITAL IN	SL_{Fe} and SL_{mem} trigger.
6	WL_Fe / BL_mem	DIGITAL IN	WL_{Fe} and BL_{mem} trigger.
7	mem_MODE	DIGITAL IN	Enable grounding of unselected TLs.
8	PRE	DIGITAL IN	Precharge TLs.
9	WE/TL	DIGITAL IN	Write enable / TL trigger.
10	SET_PARALLEL	DIGITAL IN	Set parallel input for output scan chain.
11	VREF	BIAS	Reference voltage for FeCAPs sense amplifier.
12	VDD_BL_mem	SUPPLY	BL_{mem} drivers supply voltage.
13	GND	SUPPLY	Connection to ground.
14	VDD	SUPPLY	FeCAPs periphery supply voltage.
15	VDD_WL_Fe	SUPPLY	WL_{Fe} drivers supply voltage.
16	VDD_TL	SUPPLY	FeCAPs sense amplifier supply voltage / TL drivers supply voltage.
17	VDD_SL_Fe	SUPPLY	SL_{Fe} drivers supply voltage.
18	VDD_SL_mem	SUPPLY	SL_{mem} drivers supply voltage.
19	CK_SC	DIGITAL IN	Scan chain clock.
20	SC_IN	DIGITAL IN	Scan chain input.
21	SC_OUT	DIGITAL OUT	Scan chain output.
22	SC_SEL<2>	DIGITAL IN	Scan chain selection bit.
23	SC_SEL<1>	DIGITAL IN	Scan chain selection bit.
24	SC_SEL<0>	DIGITAL IN	Scan chain selection bit.
25	GND	SUPPLY	Connection to ground.

Table B5: **Pin listing of the hybrid memory circuit.**

Appendix C

Training algorithms: Pseudo-code

This appendix details the different training algorithms used in Section 3.4.

Algorithm 1.1: Train an ANN (online hybrid-memory implementation) with L layers for multi-class classification. The function *LayerNorm()* specifies how to layer-normalize the activations. *LayerNormBackward()* specifies how to backpropagate through the normalization. *ReLU()* and *Softmax()* specify how to apply respectively ReLU and Softmax functions to the activations. *ReLUBackward()* specifies how to backpropagate through ReLU. *Update()* specifies how to update the weights when their gradients are known, function described in Algorithm 2. *Quantize()* and *Transfer()* functions (Algorithms 3 and 4 respectively) specify how to quantize the hidden weights or evaluate analog weights respectively. k is the counter limit after which the analog/quantized weights are updated, η is the learning rate, p is the hidden weights update probability, $VariabilityFlag$ defines whether variability during transfer is considered.
Require: One input and target (a^0, $a*$), previous hidden weights ($W^{(h)}$), previous analog weights ($W^{(a)}$), counter state ($count$)
Ensure: Updated weights ($W^{(h),t+1}$ and $W^{(a),t+1}$), updated counter ($count^{t+1}$)

{**1. Computing the parameters gradients**}

{**1.1 Forward Propagation:**}
for $i = 1$ **to** L **do**
    $s_i \leftarrow W_i^{(a)} a_{i-1}$
    $a_i, \mu_i, \sigma_i \leftarrow LayerNorm(s_i)$
    if $i < L$ **then**
        $a_i \leftarrow ReLU(a_i)$
    else
        $a_i \leftarrow SoftMax(a_i)$
    end if
end for

{**1.2 Backward Propagation:**} **for** $i = L$ **to** 1 **do**
    if $i = L$ **then**
        $g_{a_i} \leftarrow a_i - a*$
    else

$g_{a_i} \leftarrow ReLUBackward(g_{a_i}, a_i)$
end if
$g_{s_i} \leftarrow LayerNormBackward(g_{a_i}, s_i, \mu_i, \sigma_i)$
$g_{a_{i-1}} \leftarrow W_i^{(a)\top} g_{s_i}$
$g_{W_i} \leftarrow g_{s_i} a_{i-1}^{\top}$
end for

{**2. Accumulating the parameters gradients**}

for $i = 1$ **to** L **do**
$W_i^{(h),t+1} \leftarrow Update(W_i^{(h)}, \eta, p, g_{W_i})$
if $count = k$ **then**
if $VariabilityFlag = True$ **then**
$W_i^{(a),t+1} \leftarrow Transfer(W_i^{(h),t+1})$
else
$W_i^{(a),t+1} \leftarrow Quantize(W_i^{(h),t+1})$
end if
end if
$count^{t+1} = count + 1$
end for

Algorithm 1.2: Train an ANN (online hybrid-memory implementation) with *L* layers for binary detection. The function *ReLU()* and *Sigmoid()* specify how to apply respectively ReLU and Sigmoid functions to the activations. *ReLUBackward()* specifies how to backpropagate through ReLU. *Update()* specifies how to update the weights when their gradients are known, function described in Algorithm 2. *Quantize()* and *Transfer()* functions (Algorithms 3 and 4 respectively) specify how to quantize the hidden weights or evaluate analog weights respectively. k is the counter limit after which the analog/quantized weights are updated, η is the learning rate, p is the hidden weights update probability, $VariabilityFlag$ defines whether variability during transfer is considered.
Require: One input and target (a^0, $a*$), previous hidden weights ($W^{(h)}$), previous analog weights ($W^{(a)}$), counter state ($count$)
Ensure: Updated weights ($W^{(h),t+1}$ and $W^{(a),t+1}$), updated counter ($count^{t+1}$)

{**1. Computing the parameters gradients**}

{**1.1 Forward Propagation:**}
for $i = 1$ **to** L **do**
$a_i \leftarrow W_i^{(a)} a_{i-1}$
if $i < L$ **then**
$a_i \leftarrow ReLU(a_i)$
else
$a_i \leftarrow SoftMax(a_i)$
end if
end for

{**1.2 Backward Propagation:**} **for** $i = L$ **to** 1 **do**
  if $i = L$ **then**
    $g_{a_i} \leftarrow a_i - a*$
  else
    $g_{a_i} \leftarrow ReLUBackward(g_{a_i}, a_i)$
  end if
  $g_{a_{i-1}} \leftarrow W_i^{(a)\top} g_{a_i}$
  $g_{W_i} \leftarrow g_{s_i} a_{i-1}^{\top}$
end for

{**2. Accumulating the parameters gradients**}

for $i = 1$ **to** L **do**
  $W_i^{(h),t+1} \leftarrow Update(W_i^{(h)}, \eta, p, g_{W_i})$
  if $count = k$ **then**
    if $VariabilityFlag = True$ **then**
      $W_i^{(a),t+1} \leftarrow Transfer(W_i^{(h),t+1})$
    else
      $W_i^{(a),t+1} \leftarrow Quantize(W_i^{(h),t+1})$
    end if
  end if
  $count^{t+1} = count + 1$
end for

Algorithm 2: Update integer hidden weights ($W^{(h)}$) of one ANN layer (online hybrid-memory implementation). $Binomial()$ specifies how to generate a binary mask for a given layer. $Clip()$ specifies how to clip the weights. $Round()$ specifies how to round the integer weights to the closest integer value. η is the learning rate, p is the hidden weights update probability, N_h is the number of bits of the hidden weights, j is the layer index. $\circ$ denotes the element-wise multiplication.
Require: Previous hidden weights ($W^{(h)}$) and weights gradients (g_W)
Ensure: Updated hidden weights ($W^{(h),t+1}$)

$M \leftarrow Binomial(p, j)$
$W^{(h),t+1} \leftarrow Clip(Round((W^{(h)} - \eta g_W \circ M)2^{N_h-1}), 1 - 2^{N_h-1}, 2^{N_h-1} - 1)/2^{N_h-1}$

Algorithm 3: Quantize integer hidden weights ($W^{(h)}$) of one ANN layer to their low bit-version ($W^{(a)}$). $Sign()$ specifies the sign of the weight. $Trunc()$ returns the greatest integer less than or equal to its input. $Abs()$ evaluates the absolute value. N_h is the number of bits of the hidden weights and N_a is the number of bits of the analog weights.
Require: Previous hidden weights ($W^{(h)}$)
Ensure: Updated analog weights ($W^{(a),t+1}$)

$W^{(a),t+1} \leftarrow Sign(W^{(h)})Trunc(Abs(W^{(h)})2^{N_a-1})/2^{N_a-1}$

Algorithm 4: Transfer integer hidden weights ($W^{(h)}$) of one an ANN to their analog version ($W^{(h)}$) considering transfer operation variability. $Quantize()$ performs the operations described in *Algorithm* 3. $Round()$ specifies how to round the integer weights to the closest integer value. $Normal()$ samples random numbers from a probability distribution for a given layer. $Sign()$ specifies the sign of the weight. $Clip()$ specifies how to clip the weights. N_h is the number of bits of the hidden weights and N_a is the equivalent number of bits of the analog weights. j is the layer index, μ and σ are the average and standard deviation of the different analog weights levels.
Require: Previous hidden weights ($W^{(h)}$)
Ensure: Updated analog weights ($W^{(a),t+1}$)

$x \leftarrow Quantize(W^{(h)})2^{N_a-1}$
$y \leftarrow \sigma Normal(j) + \mu$
$Mask \leftarrow Sign(x)Sign(y) > 0$
$W^{(a),t+1} \leftarrow y \cdot Mask/2^{N_a-1}$

Algorithm 5: Train an ANN (minibatch hybrid-memory implementation) with L layers. The function *BatchNorm()* specifies how to batch-normalize the activations. *BatchNormBackward()* specifies how to backpropagate through the normalization. *ReLU()* and *Softmax()* specify how to apply respectively ReLU and Softmax functions to the activations. *ReLUBackward()* specifies how to backpropagate through ReLU. *Update()* specifies how to update the weights when their gradients are known, function described in Algorithm 2. *Quantize()* and *Transfer()* functions (Algorithms 3 and 4 respectively) specify how to quantize the hidden weights or evaluate analog weights respectively. k is the counter limit after which the analog/quantized weights are updated, η is the learning rate, p is the hidden weights update probability, $VaribilityFlag$ defines whether variability during transfer is considered, ϵ is the moving average coefficient.
Require: One input and target (a^0, $a*$), previous hidden weights ($W^{(h)}$), previous analog weights ($W^{(a)}$), counter state (*count*), previous moving average values (μ_{test} and σ_{test}).
Ensure: Updated weights ($W^{(h),t+1}$ and $W^{(a),t+1}$), updated counter ($count^{t+1}$), updated moving averages for test (μ_{test}^{t+1} and σ_{test}^{t+1})

{1. Computing the parameters gradients}

{1.1 Forward Propagation:}
for $i = 1$ **to** L **do**
    $s_i \leftarrow W_i^{(a)} a_{i-1}$
    $a_i, \mu_i, \sigma_i \leftarrow BatchNorm(s_i)$
    if $i < L$ **then**
        $a_i \leftarrow ReLU(a_i)$
    else
        $a_i \leftarrow SoftMax(a_i)$
    end if
end for

{1.2 Backward Propagation:} **for** $i = L$ **to** 1 **do**
    if $i = L$ **then**

$g_{a_i} \leftarrow a_i - a*$
else
$g_{a_i} \leftarrow ReLUBackward(g_{a_i}, a_i)$
end if
$g_{s_i} \leftarrow BatchNormBackward(g_{a_i}, s_i, \mu_i, \sigma_i)$
$g_{a_{i-1}} \leftarrow W_i^{(a)\top} g_{s_i}$
$g_{W_i} \leftarrow g_{s_i} a_{i-1}^{\top}$
end for

{2. Accumulating the parameters gradients}

for $i = 1$ **to** L **do**
$W_i^{(h),t+1} \leftarrow Update(W_i^{(h)}, \eta, p, g_{W_i})$
$\mu_{test}^{t+1} \leftarrow \epsilon\mu_{test} + (1-\epsilon)\mu_{test}$
$\sigma_{test}^{t+1} \leftarrow \epsilon\sigma_{test} + (1-\epsilon)\sigma_{test}$
if $count = k$ **then**
if $VariabilityFlag = True$ **then**
$W_i^{(a),t+1} \leftarrow Transfer(W_i^{(h),t+1})$
else
$W_i^{(a),t+1} \leftarrow Quantize(W_i^{(h),t+1})$
end if
end if
$count^{t+1} = count + 1$
end for

Appendix D

FeЯNet: Input/output interface design

This appendix provides a detailed description of the Input/Output (I/O) interface of the FeЯNet design. Two versions have been designed of the whole system, sharing the same core design, but different I/O interfaces, according to the manufacturing flow and, therefore, the subsequent characterization strategy. The FeЯNet core is either connected to an I/O pad ring, if the complete manufacturing flow is processed, or to two 25-pads test scribes, if only the partial manufacturing flow is processed.

D.1 I/O pad ring design

An I/O pad ring subsystem was developed for the design version intended to be packaged and tested with a printed circuit board (PCB) support. The purpose of this I/O subsystem is to create an interface between the external world, i.e. the electrical signals provided to the pins of the package, and the chip, which is wire-bonded to the package. The pad ring has twenty-three cells on each side, plus four corner cells. Sixteen cells for each side, out of twenty-three, are to be wire-bonded to the package. The remaining ones are either connected to the ground grid of the package or do not require to be connected. The pad ring description, detailing all the cells composing it, is shown in Figure D1. The pin listing, detailing the function of each signal is presented in Table D1 All the cells in the pad ring are taken from the internally developed PDK. Indeed, because of the hybrid process technology – foundry CMOS front-end and BEOL integrated memory devices at CEA LETI – the cells provided by the foundry PDK had to be modified in order to take into account the processing of NVMs between the fourth and fifth metal lines. The cells used in the I/O subsystem are the following:

- **Bias generator:** This cell is used to provide command signals to the digital input and output cells, according to the supply voltage used for the ring. This cell was set in auto-detection mode, meaning that it can automatically detect if the supply voltage of the pad ring is either 3.3 V or 1.8 V. One bias generator was used to drive at most four digital cells. This cell is not wire-bonded to the package.
- **Control break:** This cell is used to break the command signals coming from one

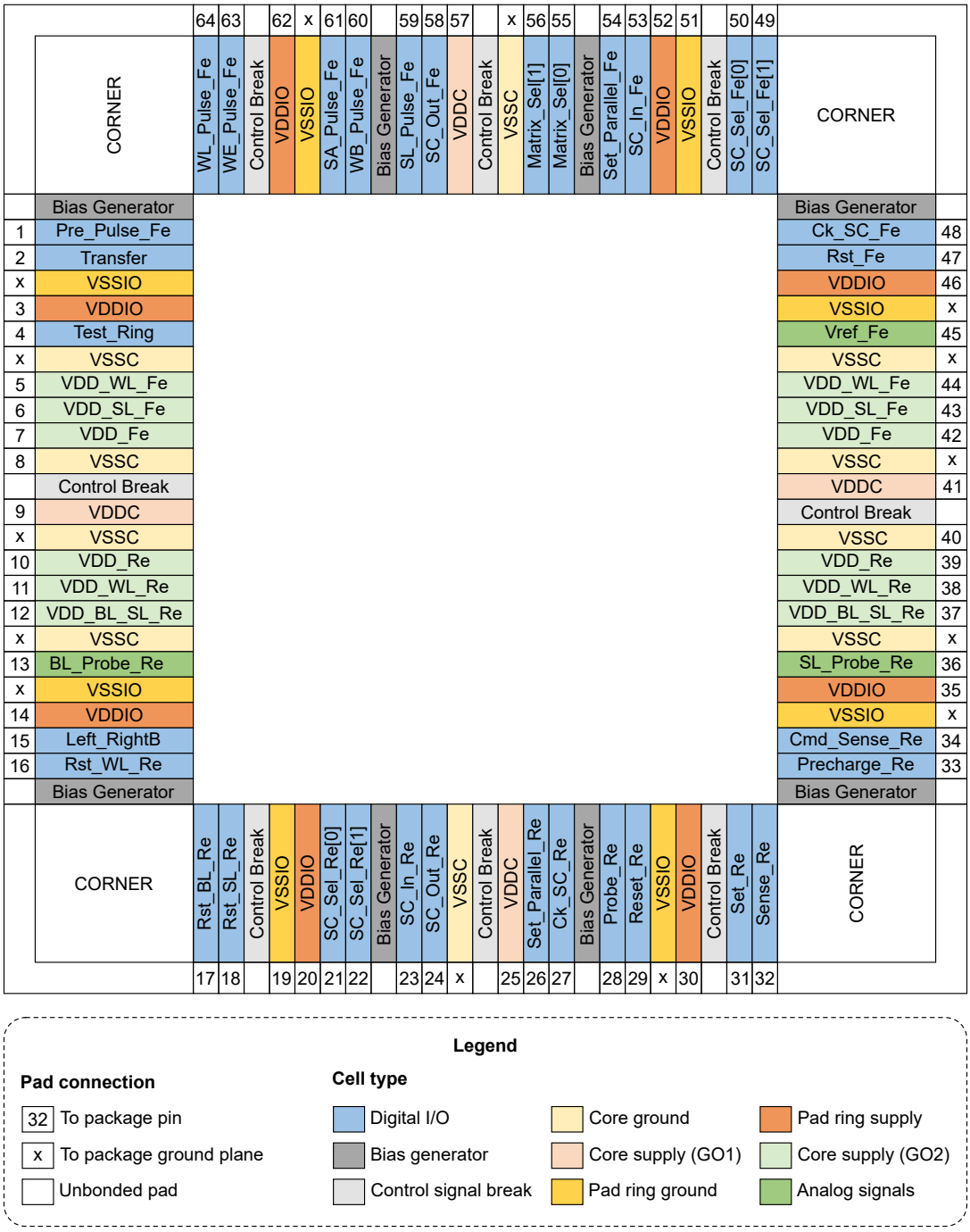

Figure D1: **Schematic view of the I/O pad ring design of the FeЯNet circuit.**

bias generator. A control break is necessary between two bias generators. This cell is not wire-bonded to the package.

- **Digital input cell:** This cell provides digital inputs from the package pins to the core.
- **Bidirectional cell:** This cell is configured for digital outputs provided to the package pins from the core.
- **VSSIO:** This cell is used to provide the ground connection of the ring. Some cells are connected to the package power grid, whereas, some others are wire-bonded to the package pins.
- **VDDIO**: This cell is used for the supply voltage connection of the ring (1.8 V or 3.3 V). This cell is wire-bonded to the package.
- **VSSC:** This cell is used to provide the ground connection of the core. Some cells are connected to the package power grid, whereas, some others are wire-bonded to the package pins.
- **VDDC:** This cell is used for the supply voltage connection of the core (0.8 V). This cell is wire-bonded to the package.
- **Analog cell:** This cell is used for the input/output analog signals. This cell is wire-bonded to the package.
- **External power supply:** This cell is used to provide a supply voltage different from the one of the ring (1.8 V or 3.3 V) or the one of the core(0.8 V). This cell is wire-bonded to the package.

The layout view of the I/O subsystem and a power grid is shown in Figure D2. The ring size is 1969 µm×1981 µm.

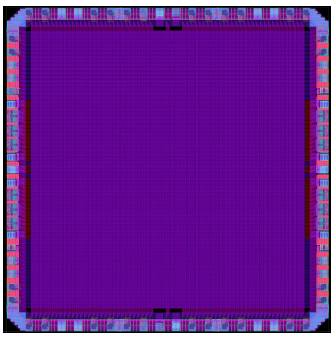

Figure D2: **Layout view of the I/O pad ring of the FeЯNet circuit.** Eight metal grids (one for each supply voltage in the chip) were interleaved over the whole area defined by the pad ring to optimize the power distribution across the whole chip.

pad number	pad name	Type	Description
1	Pre_Pulse_Fe	DIGITAL IN	Precharge FeRAM BL to ground.
2	Transfer	DIGITAL IN	Transfer data from output FeRAM scan chain to I/O RRAM scan chain.
3	VDDIO	SUPPLY	Pad ring supply voltage.
4	Test_Ring	DIGITAL OUT	Output test pad. Directly connected to pad 14.
5	VDD_WL_Fe	SUPPLY	FeRAM WL supply voltage.
6	VDD_SL_Fe	SUPPLY	FeRAM SL supply voltage.
7	VDD_Fe	SUPPLY	FeRAM BL and logic supply voltage.
8	VSSC	GROUND	Core Ground.
9	VDDC	SUPPLY	Core Supply.
10	VDD_Re	SUPPLY	RRAM logic supply voltage.
11	VDD_WL_Re	SUPPLY	RRAM WL supply voltage.
12	VDD_BL_SL_Re	SUPPLY	RRAM BL and SL supply voltage.
13	BL_Probe_Re	ANALOG	RRAM SL current test pad.
14	VDDIO	SUPPLY	Pad ring supply voltage.
15	Left_RightB	DIGITAL IN	Select left (GO2) or right (GO1) sub-core.
16	Rst_WL_Re	DIGITAL IN	Reset RRAM WL scan chain to all 0s.
17	Rst_SL_Re	DIGITAL IN	Reset RRAM BL scan chain to all 0s.
18	Rst_BL_Re	DIGITAL IN	Reset RRAM SL scan chain to all 0s.
19	VSSIO	GROUND	Pad ring ground.
20	VDDIO	SUPPLY	Pad ring supply voltage.
21	SC_Sel_Re[0]	DIGITAL IN	RRAM scan chain selection bit.
22	SC_Sel_Re[1]	DIGITAL IN	RRAM scan chain selection bit.
23	SC_In_Re	DIGITAL IN	RRAM scan chain input.
24	SC_Out_Re	DIGITAL OUT	RRAM Scan chain output.
25	VDDC	SUPPLY	Core supply voltage.
26	Set_Parallel_Re	DIGITAL IN	Set parallel input to I/O RRAM scan chain
27	Ck_SC_Re	DIGITAL IN	RRAM scan chain clock.

pad number	pad name	Type	Description
28	Probe_Re	DIGITAL IN	RRAM drivers in probe mode. Use pads 13 and 36 for read.
29	Reset_Re	DIGITAL IN	Reset pulse for RRAM.
30	VDDIO	SUPPLY	Pad ring supply voltage.
31	Set_Re	DIGITAL IN	Set pulse for RRAM.
32	Sense_Re	DIGITAL IN	RRAM drivers in sense mode. Use RRAM sense for read.
33	Precharge_Re	DIGITAL IN	Precharge RRAM sense latch to ground.
34	Cmd_Sense_Re	DIGITAL IN	Connect RRAM sense latch to BLs.
35	VDDIO	SUPPLY	Pad ring supply voltage.
36	SL_Probe_Re	ANALOG	RRAM BL read voltage for probe mode.
37	VDD_BL_SL_Re	SUPPLY	RRAM BL and SL supply voltage.
38	VDD_WL_Re	SUPPLY	RRAM WL supply voltage.
39	VDD_Re	SUPPLY	RRAM logic supply voltage.
40	VSSC	GROUND	Core ground.
41	VDDC	SUPPLY	Core supply voltage.
42	VDD_Fe	SUPPLY	FeRAM BL and logic supply voltage.
43	VDD_SL_Fe	SUPPLY	FeRAM SL supply voltage.
44	VDD_WL_Fe	SUPPLY	FeRAM WL supply voltage.
45	Vref_Fe	ANALOG	FeRAM read reference voltage.
46	VDDIO	SUPPLY	Pad ring supply voltage.
47	Rst_Fe	DIGITAL IN	Reset all FeRAM scan chains to all 0s.
48	Ck_SC_Fe	DIGITAL IN	FeRAM scan chain clock.
49	SC_Sel_Fe[1]	DIGITAL IN	FeRAM scan chain selection bit.
50	SC_Sel_Fe[0]	DIGITAL IN	FeRAM scan chain selection bit.
51	VSSIO	GROUND	Pad ring ground.
52	VDDIO	SUPPLY	Pad ring supply voltage.
53	SC_In_Fe	DIGITAL IN	FeRAM scan chain input.
54	Set_Parallel_Fe	DIGITAL IN	Set parallel input for FeRAM output scan chain.
55	Matrix_Sel[0]	DIGITAL IN	FeRAM array selection bit.
56	Matrix_Sel[1]	DIGITAL IN	FeRAM array selection bit.

pad number	pad name	Type	Description
57	VDDC	SUPPLY	Core supply voltage.
58	SC_Out_Fe	DIGITAL OUT	FeRAM scan chain output.
59	SL_Pulse_Fe	DIGITAL IN	SL trigger FeRAM.
60	WB_Pulse_Fe	DIGITAL IN	Write-back trigger FeRAM.
61	SA_Pulse_Fe	DIGITAL IN	Sensse trigger FeRAM.
62	VDDIO	SUPPLY	Pad ring supply voltage.
63	WE_Pulse_Fe	DIGITAL IN	BL trigger FeRAM.
64	WL_Pulse_Fe	DIGITAL IN	WL trigger FeRAM.

Table D1: **Pin listing of the I/O pad ring of the FeЯNet circuit.**

D.2 Double test scribes I/O

The following two tables detail the pin listing for the double test-scribe implementation of the FeЯNet circuit.

pad number	pad name	Type	Description
1	GND	GROUND	Ground reference.
2	VDD_Fe	SUPPLY	FeRAM BL and logic supply voltage.
3	GND	GROUND	Ground reference.
4	VDD_Fe	SUPPLY	FeRAM BL and logic supply voltage.
5	VDD_SL_Fe	SUPPLY	FeRAM SL supply voltage.
6	VDD_WL_Fe	SUPPLY	FeRAM WL supply voltage.
7	Transfer	DIGITAL IN	Transfer data from output FeRAM scan chain to I/O RRAM scan chain.
8	Pre_Pulse_Fe	DIGITAL IN	Precharge FeRAM BL to ground.
9	WL_Pulse_Fe	DIGITAL IN	WL trigger FeRAM.
10	WE_Pulse_Fe	DIGITAL IN	BL trigger FeRAM.
11	SA_Pulse_Fe	DIGITAL IN	Sensse trigger FeRAM.
12	Matrix_Sel[0]	DIGITAL IN	FeRAM array selection bit.
13	SC_Out_Fe	DIGITAL OUT	FeRAM scan chain output.
14	Matrix_Sel[1]	DIGITAL IN	FeRAM array selection bit.
15	WB_Pulse_Fe	DIGITAL IN	Write-back trigger FeRAM.
16	SL_Pulse_Fe	DIGITAL IN	SL trigger FeRAM.
17	Set_Parallel_Fe	DIGITAL IN	Set parallel input for FeRAM output scan chain.

pad number	pad name	Type	Description
18	SC_In_Fe	DIGITAL IN	FeRAM scan chain input.
19	Vref_Fe	ANALOG	FeRAM read reference voltage.
20	SC_Sel_Fe[1]	DIGITAL IN	FeRAM scan chain selection bit.
21	SC_Sel_Fe[0]	DIGITAL IN	FeRAM scan chain selection bit.
22	Rst_Fe	DIGITAL IN	Reset all FeRAM scan chains to all 0s.
23	Ck_SC_Fe	DIGITAL IN	FeRAM scan chain clock.
24	VDD_Buff_Fe	SUPPLY	FeRAM input level shifters and output buffers supply voltage.
25	VDD_Fe	SUPPLY	FeRAM BL and logic supply voltage.

Table D2: **Pin listing of the first I/O test scribe of the FeЯNet circuit.**

pad number	pad name	Type	Description
1	BL_Probe_Re	ANALOG	RRAM SL current test pad.
2	VDD_0p8_Re	SUPPLY	RRAM sense supply voltage.
3	VDD_WL_Re	SUPPLY	RRAM WL supply voltage.
4	VDD_BL_SL_Re	SUPPLY	RRAM BL and SL supply voltage.
5	VDD_Re	SUPPLY	RRAM logic supply voltage.
6	GND	GROUND	Ground reference.
7	Left_RightB	DIGITAL IN	Select left (GO2) or right (GO1) sub-core.
8	Rst_WL_Re	DIGITAL IN	Reset RRAM WL scan chain to all 0s.
9	Rst_BL_Re	DIGITAL IN	Reset RRAM BL scan chain to all 0s.
10	Rst_SL_Re	DIGITAL IN	Reset RRAM SL scan chain to all 0s.
11	SC_Sel_Re[0]	DIGITAL IN	RRAM scan chain selection bit.
12	SC_Sel_Re[1]	DIGITAL IN	RRAM scan chain selection bit.
13	SC_Out_Re	DIGITAL OUT	RRAM Scan chain output.
14	SC_In_Re	DIGITAL IN	RRAM scan chain input.

pad number	pad name	Type	Description
15	Set_Parallel_Re	DIGITAL IN	Set parallel input to I/O RRAM scan chain
16	Ck_SC_Re	DIGITAL IN	RRAM scan chain clock.
17	Probe_Re	DIGITAL IN	RRAM drivers in probe mode. Use pads 13 and 36 for read.
18	Reset_Re	DIGITAL IN	Reset pulse for RRAM.
19	Set_Re	DIGITAL IN	Set pulse for RRAM.
20	Sense_Re	DIGITAL IN	RRAM drivers in sense mode. Use RRAM sense for read.
21	Cmd_Sense_Re	DIGITAL IN	Connect RRAM sense latch to BLs.
22	Precharge_Re	DIGITAL IN	Precharge RRAM sense latch to ground.
23	GND	GROUND	Ground reference.
24	VDD_Buff_Re	SUPPLY	RRAM input level shifters and output buffers supply voltage.
25	SL_Probe_Re	ANALOG	RRAM BL read voltage for probe mode.

Table D3: **Pin listing of the second I/O test scribe of the FeЯNet circuit.**

Bibliography

[1] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460, 1950.

[2] M. R. Morris, J. Sohl-Dickstein, N. Fiedel, T. Warkentin, A. Dafoe, A. Faust, C. Farabet, and S. Legg, "Position: Levels of AGI for operationalizing progress on the path to AGI," in *Proceedings of the 41st International Conference on Machine Learning*, vol. 235, pp. 36308–36321, 2024.

[3] H. Bristol, H. de Boer, D. de Kroon, R. Shahani, and F. Torti, "Adopting AI at speed and scale: The 4IR push to stay competitive," *McKinsey & Company*, 2024. https://www.mckinsey.com/capabilities/operations/our-insights/adopting-ai-at-speed-and-scale-the-4ir-push-to-stay-competitive. Accessed: 2024-09-04.

[4] W. Zhang, B. Gao, J. Tang, P. Yao, S. Yu, M.-F. Chang, H.-J. Yoo, H. Qian, and H. Wu, "Neuro-inspired computing chips," *Nature Electronics*, vol. 3, pp. 371–382, July 2020.

[5] R. H. Dennard, J. Cai, and A. Kumar, "A perspective on today's scaling challenges and possible future directions," *Solid-State Electronics*, vol. 51, pp. 518–525, 2007.

[6] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14, 2014.

[7] W. Dally, "On the model of computation: Point," *Commun. ACM*, vol. 65, p. 30–32, 2022.

[8] L. Chua, "Memristor - the missing circuit element," *IEEE Transactions on Circuit Theory*, vol. 18, pp. 507–519, 1971.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, 1997.

[14] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[16] W. Su, L. Li, F. Liu, M. He, and X. Liang, "AI on the edge: A comprehensive review," *Artificial Intelligence Review*, vol. 55, pp. 6125–6183, 2022.

[17] J. Zhang, "Basic neural units of the brain: Neurons, synapses and action potential," 2019. arXiv: 1906.01703.

[18] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological Review*, vol. 65, pp. 386–408, 1958.

[19] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[21] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, pp. 141–142, 2012.

[22] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017. arXiv: 1708.07747.

[23] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., University of Toronto, 2009.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 248–255, 2009.

[25] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, p. 60, 2019.

[26] S. Hanson and L. Pratt, "Comparing biases for minimal network construction with back-propagation," in *Advances in Neural Information Processing Systems*, vol. 1, 1988.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, p. 1929–1958, 2014.

[28] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, p. 448–456, 2015.

[29] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016. arXiv: 1607.06450.

[30] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," 2017. arXiv: 1607.08022.

[31] Y. Wu and K. He, "Group normalization," 2018. arXiv: 1803.08494.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[33] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017. arXiv: 1704.04861.

[34] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "AI and ML accelerator survey and trends," in *2022 IEEE High Performance Extreme Computing Conference (HPEC)*, IEEE, Sept. 2022.

[35] STMicroelectronics, "Face identification with ID3 technologies." . https://www.st.com/content/st_com/en/st-edge-ai-suite/case-studies/face-identification-with-id3-technologies.html. Accessed: 2024-08-17.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. arXiv: 1412.6980.

[37] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 18, p. 6869–6898, 2017.

[38] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A white paper on neural network quantization," 2021. arXiv: 2106.08295.

[39] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: training deep neural networks with binary weights during propagations," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, p. 3123–3131, 2015.

[40] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, p. 4114–4122, 2016.

[41] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: Imagenet classification using binary convolutional neural networks," in *Computer Vision – ECCV 2016*, pp. 525–542, 2016.

[42] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," 2018. arXiv: 1606.06160.

[43] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, "Terngrad: Ternary gradients to reduce communication in distributed deep learning," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[44] S. Wu, G. Li, F. Chen, and L. Shi, "Training and inference with integers in deep neural networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[45] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, pp. 50–60, 2020.

[46] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, and V. Garg et al., "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, pp. 265–270, 2021.

[47] M. Solinas, M. Reyboz, S. Rousset, J. Galliere, M. Mainsant, Y. Bourrier, A. Molnos, and M. Mermillod, "On the beneficial effects of reinjections for continual learning," *SN Computer Science*, vol. 4, p. 37, 2022.

[48] W. C. Abraham and M. F. Bear, "Metaplasticity: the plasticity of synaptic plasticity," *Trends in Neurosciences*, vol. 19, pp. 126–130, 1996.

[49] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, pp. 3521–3526, 2017.

[50] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, p. 3987–3995, 2017.

[51] A. Laborieux, M. Ernoult, T. Hirtzlin, and D. Querlioz, "Synaptic metaplasticity in binarized neural networks," *Nature Communications*, vol. 12, p. 2549, 2021.

[52] P. Zhang, Y. Yan, C. Li, S. Wang, X. Xie, G. Song, and S. Kim, "Continual learning on dynamic graphs via parameter isolation," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, p. 601–611, 2023.

[53] G. M. van de Ven, H. T. Siegelmann, and A. S. Tolias, "Brain-inspired replay for continual learning with artificial neural networks," *Nature Communications*, vol. 11, p. 4069, 2020.

[54] S. Das, W.-K. Wong, T. Dietterich, A. Fern, and A. Emmott, "Incorporating expert feedback into active anomaly discovery," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 853–858, 2016.

[55] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345–1359, 2010.

[56] S.-i. Yi, J. D. Kendall, R. S. Williams, and S. Kumar, "Activity-difference training of deep neural networks using memristor crossbars," *Nature Electronics*, vol. 6, pp. 45–51, 2023.

[57] A. Kohan, E. A. Rietman, and H. T. Siegelmann, "Signal propagation: The framework for learning and inference in a forward pass," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, pp. 8585–8596, 2024.

[58] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton, "Backpropagation and the brain," *Nature Reviews Neuroscience*, vol. 21, pp. 335–346, 2020.

[59] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman, "Random synaptic feedback weights support error backpropagation for deep learning," *Nature Communications*, vol. 7, p. 13276, 2016.

[60] A. Nøkland, "Direct feedback alignment provides learning in deep neural networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, p. 1045–1053, 2016.

[61] A. Nøkland and L. H. Eidnes, "Training neural networks with local error signals," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 4839–4850, 2019.

[62] M. Jaderberg, W. M. Czarnecki, S. Osindero, O. Vinyals, A. Graves, D. Silver, and K. Kavukcuoglu, "Decoupled neural interfaces using synthetic gradients," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, p. 1627–1635, 2017.

[63] B. Scellier and Y. Bengio, "Equilibrium propagation: Bridging the gap between energy-based models and backpropagation," *Frontiers in Computational Neuroscience*, vol. 11, 2017.

[64] J. K. Eshraghian, M. Ward, E. O. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu, "Training spiking neural networks using lessons from deep learning," *Proceedings of the IEEE*, vol. 111, pp. 1016–1054, 2023.

[65] M. Dampfhoffer, T. Mesquida, A. Valentian, and L. Anghel, "Backpropagation-based learning techniques for deep spiking neural networks: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–16, 2023.

[66] S. Hooker, "The hardware lottery," *Commun. ACM*, vol. 64, p. 58–65, 2021.

[67] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, and R. Bajwa et al., "In-datacenter performance analysis of a tensor processing unit," *SIGARCH Comput. Archit. News*, vol. 45, p. 1–12, 2017.

[68] E. Bolotin, D. Nellans, O. Villa, M. O'Connor, A. Ramirez, and S. W. Keckler, "Designing efficient heterogeneous memory architectures," *IEEE Micro*, vol. 35, pp. 60–68, 2015.

[69] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, pp. 80–83, 2008.

[70] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano Lett.*, vol. 10, pp. 1297–1301, 2010.

[71] R. Vignali, R. Zurla, M. Pasotti, P. L. Rolandi, A. Singh, M. L. Gallo, A. Sebastian, T. Jang, A. Antolini, E. F. Scarselli, and A. Cabrini, "Designing circuits for AiMC based on non-volatile memories: A tutorial brief on trade-off and strategies for adcs and dacs co-design," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, pp. 1650–1655, 2024.

[72] Y. Luo, X. Han, Z. Ye, H. Barnaby, J.-S. Seo, and S. Yu, "Array-level programming of 3-bit per cell resistive memory and its application for deep neural network inference," *IEEE Transactions on Electron Devices*, vol. 67, pp. 4621–4625, 2020.

[73] V. Milo, A. Glukhov, E. Pérez, C. Zambelli, N. Lepri, M. K. Mahadevaiah, E. P.-B. Quesada, P. Olivo, C. Wenger, and D. Ielmini, "Accurate program/verify schemes of resistive switching memory (RRAM) for in-memory neural network circuits," *IEEE Transactions on Electron Devices*, vol. 68, pp. 3832–3837, 2021.

[74] M. Martemucci, B. Kersting, R. Khaddam-Aljameh, I. Boybat, S. R. Nandakumar, U. Egger, M. Brightsky, R. L. Bruce, M. Le Gallo, and A. Sebastian, "Accurate weight mapping in a multi-memristive synaptic unit," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[75] A. Vasilopoulos, J. Büchel, B. Kersting, C. Lammie, K. Brew, S. Choi, T. Philip, N. Saulnier, V. Narayanan, M. Le Gallo, and A. Sebastian, "Exploiting the state dependency of conductance variations in memristive devices for accurate in-memory computing," *IEEE Transactions on Electron Devices*, vol. 70, pp. 6279–6285, 2023.

[76] H. Tsai, P. Narayanan, S. Jain, S. Ambrogio, K. Hosokawa, M. Ishii, C. Mackin, C.-T. Chen, A. Okazaki, A. Nomura, I. Boybat, R. Muralidhar, M. M. Frank, T. Yasuda, A. Friz, Y. Kohda, A. Chen, A. Fasoli, M. J. Rasch, S. Woźniak, J. Luquin, V. Narayanan, and G. W. Burr, "Architectures and circuits for analog-memory-based hardware accelerators for deep neural networks (invited)," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2023.

[77] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C.-K. Lin, A. Lines, R. Liu, D. Mathaikutty, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y.-H. Weng, A. Wild, Y. Yang, and H. Wang, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, pp. 82–99, 2018.

[78] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm^2 12.7-pj/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, pp. 145–158, 2019.

[79] C. Frenkel and G. Indiveri, "ReckOn: A 28nm sub-mm2 task-agnostic spiking recurrent neural network processor enabling on-chip learning over second-long timescales," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, pp. 1–3, 2022.

[80] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses," *Frontiers in Neuroscience*, vol. 9, 2015.

[81] S. Pi, C. Li, H. Jiang, W. Xia, H. Xin, J. J. Yang, and Q. Xia, "Memristor crossbar arrays with 6-nm half-pitch and 2-nm critical dimension," *Nature Nanotechnology*, vol. 14, pp. 35–39, 2019.

[82] P. Lin, C. Li, Z. Wang, Y. Li, H. Jiang, W. Song, M. Rao, Y. Zhuo, N. K. Upadhyay, M. Barnell, Q. Wu, J. J. Yang, and Q. Xia, "Three-dimensional memristor circuits as complex neural networks," *Nature Electronics*, vol. 3, pp. 225–232, 2020.

[83] "International Roadmap for Devices and Systems, Beyond CMOS," 2018. https://irds.ieee.org/editions/2018.

[84] M. L. Gallo and A. Sebastian, "An overview of phase-change memory device physics," *Journal of Physics D: Applied Physics*, vol. 53, p. 213002, 2020.

[85] M. Lanza, A. Sebastian, W. D. Lu, M. L. Gallo, M.-F. Chang, D. Akinwande, F. M. Puglisi, H. N. Alshareef, M. Liu, and J. B. Roldan, "Memristive technologies for data storage, computation, encryption, and radio-frequency communication," *Science*, vol. 376, p. 9979, 2022.

[86] G. Molas and E. Nowak, "Advances in emerging memory technologies: From data storage to artificial intelligence," *Applied Sciences*, vol. 11, 2021.

[87] J. Yi, M. Kim, J. Seo, N. Park, S. Lee, J. Kim, G. Do, H. Jang, H. Koo, S. Cho, S. Chae, T. Kim, M.-H. Na, and S. Cha, "The chalcogenide-based memory technology continues: Beyond 20nm 4-deck 256Gb cross-point memory," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, pp. 1–2, 2023.

[88] F. Arnaud, P. Ferreira, F. Piazza, A. Gandolfo, P. Zuliani, P. Mattavelli, E. Gomiero, G. Samanni, J. Jasse, C. Jahan, J. P. Reynard, R. Berthelon, O. Weber, A. Villaret, B. Dumont, J. C. Grenier, R. Ranica, C. Gallon, C. Boccaccio, A. Souhaite, L. Desvoivres, D. Ristoiu, L. Favennec, V. Caubet, S. Delmedico, N. Cherault, R. Beneyton, S. Chouteau, P. O. Sassoulas, L. Clement, P. Boivin, D. Turgis, F. Disegni, J. L. Ogier, X. Federspiel, O. Kermarrec, M. Molgg, A. Viscuso, R. Annunziata, A. Maurelli, P. Cappelletti, and E. Ciantar, "High density embedded PCM cell in 28nm FDSOI technology for automotive micro-controller applications," in *2020 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2020.

[89] A. Sebastian, M. L. Gallo, and E. Eleftheriou, "Computational phase-change memory: Beyond von neumann computing," *Journal of Physics D: Applied Physics*, vol. 52, p. 443002, aug 2019.

[90] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electronics*, vol. 1, pp. 333–343, 2018.

[91] S. Bhatti, R. Sbiaa, A. Hirohata, H. Ohno, S. Fukami, and S. Piramanayagam, "Spintronics based random access memory: A review," *Materials Today*, vol. 20, pp. 530–548, 2017.

[92] T. Y. Lee, J. M. Lee, M. K. Kim, J. S. Oh, J. W. Lee, H. M. Jeong, P. H. Jang, M. K. Joo, K. Suh, S. H. Han, D.-E. Jeong, T. Kai, J. H. Jeong, J.-H. Park, J. H. Lee, Y. H. Park, E. B. Chang, Y. K. Park, H. J. Shin, Y. S. Ji, S. H. Hwang, K. T. Nam, B. S. Kwon, M. K. Cho, B. Y. Seo, Y. J. Song, G. H. Koh, K. Lee, J.-H. Lee, and G. T. Jeong, "World-most energy-efficient MRAM technology for non-volatile RAM applications," in *2022 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2022.

[93] J. J. Sun, M. DeHerrera, B. Hughes, S. Ikegawa, H. K. Lee, F. B. Mancoff, K. Nagel, G. Shimon, S. M. Alam, D. Houssameddine, and S. Aggarwal, "Commercialization of 1Gb standalone spin-transfer torque MRAM," in *2021 IEEE International Memory Workshop (IMW)*, pp. 1–4, 2021.

[94] T. Mikolajick, C. Dehm, W. Hartner, I. Kasko, M. Kastner, N. Nagel, M. Moert, and C. Mazure, "FeRAM technology for high density applications," *Microelectronics Reliability*, vol. 41, pp. 947–950, 2001.

[95] F. P. G. Fengler, M. Pešić, S. Starschich, T. Schneller, U. Böttger, T. Schenk, M. H. Park, T. Mikolajick, and U. Schroeder, "Comparison of hafnia and PZT based ferroelectrics for future non-volatile FRAM applications," in *2016 46th European Solid-State Device Research Conference (ESSDERC)*, pp. 369–372, 2016.

[96] T. S. Böscke, J. Müller, D. Bräuhaus, U. Schröder, and U. Böttger, "Ferroelectricity in hafnium oxide thin films," *Applied Physics Letters*, vol. 99, p. 102903, 2011.

[97] N. Ramaswamy, A. Calderoni, J. Zahurak, G. Servalli, A. Chavan, S. Chhajed, M. Balakrishnan, M. Fischer, M. Hollander, D. P. Ettisserry, A. Liao, K. Karda, M. Jerry, M. Mariani, A. Visconti, B. R. Cook, B. D. Cook, D. Mills, A. Torsi, C. Mouli, E. Byers, M. Helm, S. Pawlowski, S. Shiratake, and N. Chandrasekaran, "NVDRAM: A 32Gb dual layer 3D stacked non-volatile ferroelectric memory with near-DRAM performance for demanding AI workloads," in *2023 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2023.

[98] T. Francois, L. Grenouillet, J. Coignus, N. Vaxelaire, C. Carabasse, F. Aussenac, S. Chevalliez, S. Slesazeck, C. Richter, P. Chiquet, M. Bocquet, U. Schroeder, T. Mikolajick, F. Gaillard, and E. Nowak, "Impact of area scaling on the ferroelectric properties of back-end of line compatible $Hf_{0.5}Zr_{0.5}O_2$ and Si:HfO_2-based MFM capacitors," *Applied Physics Letters*, vol. 118, p. 062904, 2021.

[99] S. Mukherjee, J. Bizindavyi, Y.-C. Luo, S. Clima, J. Read, M. I. Popovici, Y. Xiang, N. Bazzazian, A. Belmonte, R. Delhougne, G. S. Kar, F. Catthoor, V. V. Afanas'Ev, S. Yu, and J. Van Houdt, "Pulse-based capacitive memory window with high non-destructive read endurance in fully beol compatible ferroelectric capacitors," in *2023 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2023.

[100] S. Beyer, S. Dünkel, M. Trentzsch, J. Müller, A. Hellmich, D. Utess, J. Paul, D. Kleimaier, J. Pellerin, S. Müller, J. Ocker, A. Benoist, H. Zhou, M. Mennenga, M. Schuster, F. Tassan, M. Noack, A. Pourkeramati, F. Müller, M. Lederer, T. Ali, R. Hoffmann, T. Kämpfe, K. Seidel, H. Mulaosmanovic, E. T. Breyer, T. Mikolajick, and S. Slesazeck, "FeFET: A versatile CMOS compatible device with game-changing potential," in *2020 IEEE International Memory Workshop (IMW)*, pp. 1–4, 2020.

[101] M.-H. Yan, M.-H. Wu, H.-H. Huang, Y.-H. Chen, Y.-H. Chu, T.-L. Wu, P.-C. Yeh, C.-Y. Wang, Y.-D. Lin, J.-W. Su, P.-J. Tzeng, S.-S. Sheu, W.-C. Lo, C.-I. Wu, and T.-H. Hou, "BEOL-compatible multiple metal-ferroelectric-metal (m-MFM) FETs designed for low voltage (2.5 V), high density, and excellent reliability," in *2020 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2020.

[102] M.-H. Wu, C.-Y. Cho, H.-H. Huang, T.-S. Huang, I.-T. Wang, W.-Y. Jang, S.-Z. Chang, and T.-H. Hou, "Two-transistor metal–ferroelectric–metal field-effect transistor (2T-MFMFET) for scalable embedded nonvolatile memory—part II: Experiment," *IEEE Transactions on Electron Devices*, vol. 70, pp. 6268–6272, 2023.

[103] A. Chanthbouala, V. Garcia, R. O. Cherifi, K. Bouzehouane, S. Fusil, X. Moya, S. Xavier, H. Yamada, C. Deranlot, N. D. Mathur, M. Bibes, A. Barthélémy, and J. Grollier, "A ferroelectric memristor," *Nature Materials*, vol. 11, pp. 860–864, 2012.

[104] E. Covi, Q. T. Duong, S. Lancaster, V. Havel, J. Coignus, J. Barbot, O. Richter, P. Klein, E. Chicca, L. Grenouillet, A. Dimoulas, T. Mikolajick, and S. Slesazeck, "Ferroelectric tunneling junctions for edge computing," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[105] J. Park, A. Kumar, Y. Zhou, S. Oh, J.-H. Kim, Y. Shi, S. Jain, G. Hota, E. Qiu, A. L. Nagle, I. K. Schuller, C. D. Schuman, G. Cauwenberghs, and D. Kuzum, "Multilevel, forming and filament free, bulk switching trilayer RRAM for neuromorphic computing at the edge," *Nature Communications*, vol. 15, p. 3492, 2024.

[106] H.-S. P. Wong, H.-Y. Lee, S. Yu, Y.-S. Chen, Y. Wu, P.-S. Chen, B. Lee, F. T. Chen, and M.-J. Tsai, "Metal–oxide RRAM," *Proceedings of the IEEE*, vol. 100, pp. 1951–1970, 2012.

[107] M. Le Gallo, R. Khaddam-Aljameh, M. Stanisavljevic, A. Vasilopoulos, B. Kersting, M. Dazzi, G. Karunaratne, M. Brändli, A. Singh, S. M. Müller, J. Büchel, X. Timoneda, V. Joshi, M. J. Rasch, U. Egger, A. Garofalo, A. Petropoulos, T. Antonakopoulos, K. Brew, S. Choi, I. Ok, T. Philip, V. Chan, C. Silvestre, I. Ahsan, N. Saulnier, V. Narayanan, P. A. Francese, E. Eleftheriou, and A. Sebastian, "A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference," *Nature Electronics*, vol. 6, pp. 680–693, 2023.

[108] S. Ambrogio, P. Narayanan, A. Okazaki, A. Fasoli, C. Mackin, K. Hosokawa, A. Nomura, T. Yasuda, A. Chen, A. Friz, M. Ishii, J. Luquin, Y. Kohda, N. Saulnier, K. Brew, S. Choi, I. Ok, T. Philip, V. Chan, C. Silvestre, I. Ahsan, V. Narayanan, H. Tsai, and G. W. Burr, "An analog-AI chip for energy-efficient speech recognition and transcription," *Nature*, vol. 620, pp. 768–775, 2023.

[109] R. Khaddam-Aljameh, M. Stanisavljevic, J. Fornt Mas, G. Karunaratne, M. Braendli, F. Liu, A. Singh, S. M. Müller, U. Egger, A. Petropoulos, T. Antonakopoulos, K. Brew, S. Choi, I. Ok, F. L. Lie, N. Saulnier, V. Chan, I. Ahsan, V. Narayanan, S. R. Nandakumar, M. Le Gallo, P. A. Francese, A. Sebastian, and E. Eleftheriou, "Hermes core – a 14nm cmos and pcm-based in-memory compute core using an array of 300ps/lsb linearized cco-based adcs and local digital processing," in *2021 Symposium on VLSI Circuits*, pp. 1–2, 2021.

[110] M. J. Rasch, C. Mackin, M. Le Gallo, A. Chen, A. Fasoli, F. Odermatt, N. Li, S. R. Nandakumar, P. Narayanan, H. Tsai, G. W. Burr, A. Sebastian, and V. Narayanan, "Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators," *Nature Communications*, vol. 14, p. 5282, Aug 2023.

[111] G. W. Burr, H. Tsai, W. Simon, I. Boybat, S. Ambrogio, C.-E. Ho, Z.-W. Liou, M. Rasch, J. Büchel, P. Narayanan, T. Gordon, S. Jain, T. M. Levin, K. Hosokawa, M. Le Gallo, H. Smith, M. Ishii, Y. Kohda, A. Chen, C. Mackin, A. Fasoli, K. El-Maghraoui, R. Muralidhar, A. Okazaki, C. T. Chen, M. M. Frank, C. Lammie, A. Vasilopoulos, A. M. Friz, J. Luquin, S. Teehan, I. Ahsan, A. Sebastian, and V. Narayanan, "Design of analog-AI hardware accelerators for transformer-based language models (invited)," in *2023 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2023.

[112] S. Jung, H. Lee, S. Myung, H. Kim, S. K. Yoon, S.-W. Kwon, Y. Ju, M. Kim, W. Yi, S. Han, B. Kwon, B. Seo, K. Lee, G.-H. Koh, K. Lee, Y. Song, C. Choi, D. Ham, and S. J. Kim, "A crossbar array of magnetoresistive memory devices for in-memory computing," *Nature*, vol. 601, pp. 211–216, 2022.

[113] S. Angizi, Z. He, A. Awad, and D. Fan, "MRIMA: An MRAM-based in-memory accelerator," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, pp. 1123–1136, 2020.

[114] Y. Pan, P. Ouyang, Y. Zhao, W. Kang, S. Yin, Y. Zhang, W. Zhao, and S. Wei, "A multilevel cell STT-MRAM-based computing in-memory accelerator for binary convolutional neural network," *IEEE Transactions on Magnetics*, vol. 54, pp. 1–5, 2018.

[115] S. Jain, A. Ranjan, K. Roy, and A. Raghunathan, "Computing in memory with spin-transfer torque magnetic RAM," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, pp. 470–483, 2018.

[116] Z. Chowdhury, J. D. Harms, S. K. Khatamifard, M. Zabihi, Y. Lv, A. P. Lyle, S. S. Sapatnekar, U. R. Karpuzcu, and J.-P. Wang, "Efficient in-memory processing using spintronics," *IEEE Computer Architecture Letters*, vol. 17, pp. 42–46, 2018.

[117] T. Soliman, S. Chatterjee, N. Laleni, F. Müller, T. Kirchner, N. Wehn, T. Kämpfe, Y. S. Chauhan, and H. Amrouch, "First demonstration of in-memory computing crossbar using multi-level cell FeFET," *Nature Communications*, vol. 14, p. 6348, 2023.

[118] T. Soliman, N. Laleni, T. Kirchner, F. Müller, A. Shrivastava, T. Kämpfe, A. Guntoro, and N. Wehn, "FELIX: A ferroelectric FET based low power mixed-signal in-memory architecture for DNN acceleration," *ACM Trans. Embed. Comput. Syst.*, vol. 21, 2022.

[119] S. De, F. Müller, N. Laleni, M. Lederer, Y. Raffel, S. Mojumder, A. Vardar, S. Abdulazhanov, T. Ali, S. Dünkel, S. Beyer, K. Seidel, and T. Kämpfe, "Demonstration of multiply-accumulate operation with 28 nm FeFET crossbar array," *IEEE Electron Device Letters*, vol. 43, pp. 2081–2084, 2022.

[120] T. Soliman, F. Müller, T. Kirchner, T. Hoffmann, H. Ganem, E. Karimov, T. Ali, M. Lederer, C. Sudarshan, T. Kämpfe, A. Guntoro, and N. Wehn, "Ultra-low power flexible precision FeFET based analog in-memory computing," in *2020 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2020.

[121] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, and H. Qian, "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, pp. 641–646, 2020.

[122] W. Wan, R. Kubendran, C. Schaefer, S. B. Eryilmaz, W. Zhang, D. Wu, S. Deiss, P. Raina, H. Qian, B. Gao, S. Joshi, H. Wu, H.-S. P. Wong, and G. Cauwenberghs, "A compute-in-memory chip based on resistive random-access memory," *Nature*, vol. 608, pp. 504–512, 2022.

[123] M. Bocquet, T. Hirztlin, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "In-memory and error-immune differential RRAM implementation of binarized deep neural networks," in *2018 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2018.

[124] F. Jebali, A. Majumdar, C. Turck, K.-E. Harabi, M.-C. Faye, E. Muhr, J.-P. Walder, O. Bilousov, A. Michaud, E. Vianello, T. Hirtzlin, F. Andrieu, M. Bocquet, S. Collin, D. Querlioz, and J.-M. Portal, "Powering AI at the edge: A robust, memristor-based binarized neural network with near-memory computing and miniaturized solar cell," *Nature Communications*, vol. 15, p. 741, 2024.

[125] G. Burr, R. Shelby, C. di Nolfo, J. Jang, R. Shenoy, P. Narayanan, K. Virwani, E. Giacometti, B. Kurdi, and H. Hwang, "Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses), using phase-change memory as the synaptic weight element," in *2014 IEEE International Electron Devices Meeting*, pp. 1–4, 2014.

[126] S. Ambrogio, P. Narayanan, H. Tsai, R. M. Shelby, I. Boybat, C. di Nolfo, S. Sidler, M. Giordano, M. Bodini, N. C. P. Farinha, B. Killeen, C. Cheng, Y. Jaoudi, and G. W. Burr, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, pp. 60–67, 2018.

[127] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations," *Frontiers in Neuroscience*, vol. 10, 2016.

[128] S. Yu, W. Shim, X. Peng, and Y. Luo, "RRAM for compute-in-memory: From inference to training," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, pp. 2753–2765, 2021.

[129] W. Zhang, P. Yao, B. Gao, Q. Liu, D. Wu, Q. Zhang, Y. Li, Q. Qin, J. Li, Z. Zhu, Y. Cai, D. Wu, J. Tang, H. Qian, Y. Wang, and H. Wu, "Edge learning using a fully integrated neuro-inspired memristor chip," *Science*, vol. 381, pp. 1205–1211, 2023.

[130] T.-H. Wen, J.-M. Hung, W.-H. Huang, C.-J. Jhang, Y.-C. Lo, H.-H. Hsu, Z.-E. Ke, Y.-C. Chen, Y.-H. Chin, C.-I. Su, W.-S. Khwa, C.-C. Lo, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, M.-S. Ho, C.-C. Chou, Y.-D. Chih, T.-Y. J. Chang, and M.-F. Chang, "Fusion of memristor and digital compute-in-memory processing for energy-efficient edge computing," *Science*, vol. 384, pp. 325–332, 2024.

[131] Y. Li, J. Tang, B. Gao, J. Yao, A. Fan, B. Yan, Y. Yang, Y. Xi, Y. Li, J. Li, W. Sun, Y. Du, Z. Liu, Q. Zhang, S. Qiu, Q. Li, H. Qian, and H. Wu, "Monolithic three-dimensional integration of RRAM-based hybrid memory architecture for one-shot learning," *Nature Communications*, vol. 14, p. 7140, 2023.

[132] H. Ning, Z. Yu, Q. Zhang, H. Wen, B. Gao, Y. Mao, Y. Li, Y. Zhou, Y. Zhou, J. Chen, L. Liu, W. Wang, T. Li, Y. Li, W. Meng, W. Li, Y. Li, H. Qiu, Y. Shi, Y. Chai, H. Wu, and X. Wang, "An in-memory computing architecture based on a duplex two-dimensional material structure for in situ machine learning," *Nature Nanotechnology*, vol. 18, pp. 493–500, 2023.

[133] Y. Luo, Y.-C. Luc, and S. Yu, "A feram based volatile/non-volatile dual-mode buffer memory for deep neural network training," in *2021 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pp. 1871–1876, 2021.

[134] A. P. Huang, Z. C. Yang, and P. K. Chu, "Hafnium-based high-k gate dielectrics," in *Advances in Solid State Circuit Technologies*, ch. 16, InTech, 2010.

[135] K. Mistry, C. Allen, C. Auth, B. Beattie, D. Bergstrom, M. Bost, M. Brazier, M. Buehler, A. Cappellani, R. Chau, C.-H. Choi, G. Ding, K. Fischer, T. Ghani, R. Grover, W. Han, D. Hanken, M. Hattendorf, J. He, J. Hicks, R. Huessner, D. Ingerly, P. Jain, R. James, L. Jong, S. Joshi, C. Kenyon, K. Kuhn, K. Lee, H. Liu, J. Maiz, B. McIntyre, P. Moon, J. Neirynck, S. Pae, C. Parker, D. Parsons, C. Prasad, L. Pipes, M. Prince, P. Ranade, T. Reynolds, J. Sandford, L. Shifren, J. Sebastian, J. Seiple, D. Simon, S. Sivakumar, P. Smith, C. Thomas, T. Troeger, P. Vandervoorn, S. Williams, and K. Zawadzki, "A 45nm logic technology with high-k+metal gate transistors, strained silicon, 9 Cu interconnect layers, 193nm dry patterning, and 100% Pb-free packaging," in *2007 IEEE International Electron Devices Meeting*, pp. 247–250, 2007.

[136] L. Goux, P. Czarnecki, Y. Y. Chen, L. Pantisano, X. P. Wang, R. Degraeve, B. Govoreanu, M. Jurczak, D. J. Wouters, and L. Altimime, "Evidences of oxygen-mediated resistive-switching mechanism in TiN/HfO_2/Pt cells," *Applied Physics Letters*, vol. 97, p. 243509, 2010.

[137] M. Fukunaga and Y. Noda, "New technique for measuring ferroelectric and antiferroelectric hysteresis loops," *Journal of the Physical Society of Japan*, vol. 77, p. 064706, 2008.

[138] M. Pešić, F. P. G. Fengler, L. Larcher, A. Padovani, T. Schenk, E. D. Grimley, X. Sang, J. M. LeBeau, S. Slesazeck, U. Schroeder, and T. Mikolajick, "Physical mechanisms behind the field-cycling behavior of HfO_2-based ferroelectric capacitors," *Advanced Functional Materials*, vol. 26, pp. 4601–4612, 2016.

[139] J. Okuno, T. Yonai, T. Kunihiro, Y. Shuto, R. Alcala, M. Lederer, K. Seidel, T. Mikolajick, U. Schroeder, M. Tsukamoto, and T. Umebayashi, "Investigation of recovery phenomena in $Hf_{0.5}Zr_{0.5}O_2$-based 1T1C FeRAM," *IEEE Journal of the Electron Devices Society*, vol. 11, pp. 43–46, 2023.

[140] L. Grenouillet, T. Francois, J. Coignus, S. Kerdilès, N. Vaxelaire, C. Carabasse, F. Mehmood, S. Chevalliez, C. Pellissier, F. Triozon, F. Mazen, G. Rodriguez, T. Magis, V. Havel, S. Slesazeck, F. Gaillard, U. Schroeder, T. Mikolajick, and E. Nowak, "Nanosecond laser anneal (NLA) for Si-implanted HfO_2 ferroelectric memories integrated in back-end of line (BEOL)," in *2020 IEEE Symposium on VLSI Technology*, pp. 1–2, 2020.

[141] J. Müller, U. Schröder, T. S. Böscke, I. Müller, U. Böttger, L. Wilde, J. Sundqvist, M. Lemberger, P. Kücher, T. Mikolajick, and L. Frey, "Ferroelectricity in yttrium-doped hafnium oxide," *Journal of Applied Physics*, vol. 110, p. 114113, 2011.

[142] S. Mueller, C. Adelmann, A. Singh, S. V. Elshocht, U. Schroeder, and T. Mikolajick, "Ferroelectricity in Gd-doped HfO_2 thin films," *ECS Journal of Solid State Science and Technology*, vol. 1, 2012.

[143] S. Mueller, J. Mueller, A. Singh, S. Riedel, J. Sundqvist, U. Schroeder, and T. Mikolajick, "Incipient ferroelectricity in Al-doped HfO_2 thin films," *Advanced Functional Materials*, vol. 22, pp. 2412–2417, 2012.

[144] T. Schenk, S. Mueller, U. Schroeder, R. Materlik, A. Kersch, M. Popovici, C. Adelmann, S. Van Elshocht, and T. Mikolajick, "Strontium doped hafnium oxide thin films: Wide process window for ferroelectric memories," in *2013 Proceedings of the European Solid-State Device Research Conference (ESSDERC)*, pp. 260–263, 2013.

[145] J. Müller, T. S. Böscke, S. Müller, E. Yurchuk, P. Polakowski, J. Paul, D. Martin, T. Schenk, K. Khullar, A. Kersch, W. Weinreich, S. Riedel, K. Seidel, A. Kumar, T. M. Arruda, S. V. Kalinin, T. Schlösser, R. Boschke, R. van Bentum, U. Schröder, and T. Mikolajick, "Ferroelectric hafnium oxide: A CMOS-compatible and highly scalable approach to future ferroelectric memories," in *2013 IEEE International Electron Devices Meeting*, pp. 1–4, 2013.

[146] J. Müller, T. S. Böscke, U. Schröder, S. Mueller, D. Bräuhaus, U. Böttger, L. Frey, and T. Mikolajick, "Ferroelectricity in simple binary ZrO_2 and HfO_2," *Nano Letters*, vol. 12, pp. 4318–4323, 2012.

[147] T. Francois, L. Grenouillet, J. Coignus, P. Blaise, C. Carabasse, N. Vaxelaire, T. Magis, F. Aussenac, V. Loup, C. Pellissier, S. Slesazeck, V. Havel, C. Richter, A. Makosiej, B. Giraud, E. T. Breyer, M. Materano, P. Chiquet, M. Bocquet, E. Nowak, U. Schroeder, and F. Gaillard, "Demonstration of BEOL-compatible ferroelectric $Hf_{0.5}Zr_{0.5}O_2$ scaled FeRAM co-integrated with 130nm CMOS for embedded NVM applications," in *2019 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2019.

[148] J. Okuno, T. Kunihiro, K. Konishi, Y. Shuto, F. Sugaya, M. Materano, T. Ali, M. Lederer, K. Kuehnel, K. Seidel, T. Mikolajick, U. Schroeder, M. Tsukamoto, and T. Umebayashi, "Reliability study of 1T1C FeRAM arrays with $Hf_{0.5}Zr_{0.5}O_2$ thickness scaling," *IEEE Journal of the Electron Devices Society*, vol. 10, pp. 778–783, 2022.

[149] J. Okuno, T. Kunihiro, T. Yonai, R. Ono, Y. Shuto, R. Alcala, M. Lederer, K. Seidel, T. Mikolajick, U. Schroeder, M. Tsukamoto, and T. Umebayashi, "A highly reliable 1.8 V 1 Mb $Hf_{0.5}Zr_{0.5}O_2$-based 1T1C FeRAM array with 3-D capacitors," in *2023 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2023.

[150] L. Grenouillet, J. Barbot, J. Laguerre, S. Martin, C. Carabasse, M. Louro, M. Bedjaoui, S. Minoret, S. Kerdilès, C. Boixaderas, T. Magis, C. Jahan, F. Andrieu, and J. Coignus, "Reliability assessment of hafnia-based ferroelectric devices and arrays for memory and AI applications (invited)," in *2023 IEEE International Reliability Physics Symposium (IRPS)*, pp. 1–8, 2023.

[151] T. Francois, J. Coignus, A. Makosiej, B. Giraud, C. Carabasse, J. Barbot, S. Martin, N. Castellani, T. Magis, H. Grampeix, S. Van Duijn, C. Mounet, P. Chiquet, U. Schroeder, S. Slesazeck, T. Mikolajick, E. Nowak, M. Bocquet, N. Barrett, F. Andrieu, and L. Grenouillet, "16kbit HfO_2:Si-based 1T-1C FeRAM arrays demonstrating high performance operation and solder reflow compatibility," in *2021 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2021.

[152] O. Billoint, S. Martin, J. Laguerre, L. Hosier, J. Coignus, C. Carabasse, F. Andrieu, and L. Grenouillet, "Charge-based sense demonstration in 1T-1C HZO FeRAM arrays to overcome C_{BL}-induced bank size limitations," in *2024 IEEE International Memory Workshop (IMW)*, pp. 1–4, 2024.

[153] Y.-M. Kim and J.-S. Lee, "Reproducible resistance switching characteristics of hafnium oxide-based nonvolatile memory devices," *Journal of Applied Physics*, vol. 104, p. 114115, 2008.

[154] A. Grossi, E. Nowak, C. Zambelli, C. Pellissier, S. Bernasconi, G. Cibrario, K. El Hajjam, R. Crochemore, J. Nodin, P. Olivo, and L. Perniola, "Fundamental variability limits of filament-based RRAM," in *2016 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2016.

[155] M. Barlas, A. Grossi, L. Grenouillet, E. Vianello, E. Nolot, N. Vaxelaire, P. Blaise, B. Traoré, J. Coignus, F. Perrin, R. Crochemore, F. Mazen, L. Lachal, S. Pauliac, C. Pellissier, S. Bernasconi, S. Chevalliez, J. F. Nodin, L. Perniola, and E. Nowak, "Improvement of HfO_2 based RRAM array performances by local Si implantation," in *2017 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2017.

[156] L. Grenouillet, N. Castellani, A. Persico, V. Meli, S. Martin, O. Billoint, R. Segaud, S. Bernasconi, C. Pellissier, C. Jahan, C. Charpin-Nicolle, P. Dezest, C. Carabasse, P. Besombes, S. Ricavy, N.-P. Tran, A. Magalhaes-Lucas, A. Roman, C. Boixaderas, T. Magis, M. Bedjaoui, M. Tessaire, A. Seignard, F. Mazen, S. Landis, E. Vianello, G. Molas, F. Gaillard, J. Arcamone, and E. Nowak, "16kbit 1T1R OxRAM arrays embedded in 28nm FDSOI technology demonstrating low BER, high endurance, and compatibility with core logic transistors," in *2021 IEEE International Memory Workshop (IMW)*, pp. 1–4, 2021.

[157] Z. Fang, X. P. Wang, J. Sohn, B. B. Weng, Z. P. Zhang, Z. X. Chen, Y. Z. Tang, G.-Q. Lo, J. Provine, S. S. Wong, H.-S. P. Wong, and D.-L. Kwong, "The role of Ti capping layer in HfO_x-based RRAM devices," *IEEE Electron Device Letters*, vol. 35, pp. 912–914, 2014.

[158] E. Esmanhotto, T. Hirtzlin, D. Bonnet, N. Castellani, J.-M. Portal, D. Querlioz, and E. Vianello, "Experimental demonstration of multilevel resistive random access memory programming for up to two months stable neural networks inference accuracy," *Advanced Intelligent Systems*, vol. 4, p. 2200145, 2022.

[159] H. Y. Lee, P. S. Chen, T. Y. Wu, Y. S. Chen, C. C. Wang, P. J. Tzeng, C. H. Lin, F. Chen, C. H. Lien, and M.-J. Tsai, "Low power and high speed bipolar switching with a thin reactive ti buffer layer in robust HfO_2 based RRAM," in *2008 IEEE International Electron Devices Meeting*, pp. 1–4, 2008.

[160] A. Grossi, E. Perez, C. Zambelli, P. Olivo, and C. Wenger, "Performance and reliability comparison of 1T-1R RRAM arrays with amorphous and polycrystalline HfO_2," in *2016 Joint International EUROSOI Workshop and International Conference on Ultimate Integration on Silicon (EUROSOI-ULIS)*, pp. 80–83, 2016.

[161] B. Max, M. Pešić, S. Slesazeck, and T. Mikolajick, "Interplay between ferroelectric and resistive switching in doped crystalline HfO_2," *Journal of Applied Physics*, vol. 123, p. 134102, 2018.

[162] J. Knabe, F. Berg, K. Thorben Goβ, U. Boettger, and R. Dittmann, "Dual-mode operation of epitaxial $Hf_{0.5}Zr_{0.5}O_2$: Ferroelectric and filamentary-type resistive switching," *physica status solidi (a)*, p. 2300409, 2023.

[163] L. Grenouillet, S. Martin, J. Laguerre, J. Barbot, N. Vaxelaire, P.-M. Deleuze, O. Renault, R. Kies, C. Carabasse, P. Dezest, F. Aussenac, R. Souil, O. Billoint, E. Vianello, W. Hamouda, N. Barrett, F. Andrieu, and J. Coignus, "Synergetic FeRAM/OxRAM memory array operation induced by oxygen vacancy interface engineering in Si:HfO_2 ferroelectric films," in *2023 International Conference on Solid State Devices and Materials (SSDM)*, 2023.

[164] N. Barrett, W. Hamouda, C. Lubin, J. Laguerre, C. Carabasse, N. Vaxelaire, J. Coignus, S. Martin, and L. Grenouillet, "Oxygen vacancy engineering in Si-doped, HfO_2 ferroelectric capacitors using Ti oxygen scavenging layers," *Applied Physics Letters*, vol. 125, p. 043502, 2024.

[165] J. Laguerre, M. Bocquet, O. Billoint, S. Martin, J. Coignus, C. Carabasse, T. Magis, T. Dewolf, F. Andrieu, and L. Grenouillet, "Memory window in Si:HfO_2 FeRAM arrays: Performance improvement and extrapolation at advanced nodes," in *2023 IEEE International Memory Workshop (IMW)*, IEEE, 2023.

[166] T. Bauvent, G. Pillonnet, and G. Molas, "Optimizing RRAM performance: A comparative analysis of forming strategies," in *2024 IEEE International Memory Workshop (IMW)*, pp. 1–4, 2024.

[167] C. Nguyen, C. Cagli, G. Molas, B. Sklenard, C. Nail, K. El Hajjam, J.-F. Nodin, C. Charpin, S. Bernasconi, and G. Reimbold, "Study of forming impact on 4Kbit

RRAM array performances and reliability," in *2017 IEEE International Memory Workshop (IMW)*, pp. 1–4, 2017.

[168] S. Yu, X. Guan, and H.-S. P. Wong, "Understanding metal oxide RRAM current overshoot and reliability using kinetic monte carlo simulation," in *2012 International Electron Devices Meeting*, pp. 1–4, 2012.

[169] C. Nail, G. Molas, P. Blaise, G. Piccolboni, B. Sklenard, C. Cagli, M. Bernard, A. Roule, M. Azzaz, E. Vianello, C. Carabasse, R. Berthier, D. Cooper, C. Pelissier, T. Magis, G. Ghibaudo, C. Vallée, D. Bedeau, O. Mosendz, B. De Salvo, and L. Perniola, "Understanding RRAM endurance, retention and window margin trade-off using experimental results and simulations," in *2016 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2016.

[170] E. Esmanhotto, T. Hirtzlin, N. Castellani, S. Martin, B. Giraud, F. Andrieu, J. F. Nodin, D. Querlioz, J.-M. Portal, and E. Vianello, "Experimental demonstration of single-level and multi-level-cell RRAM-based in-memory computing with up to 16 parallel operations," in *2022 IEEE International Reliability Physics Symposium (IRPS)*, pp. 1–4, 2022.

[171] E. Esmanhotto, L. Brunet, N. Castellani, D. Bonnet, T. Dalgaty, L. Grenouillet, D. R. B. Ly, C. Cagli, C. Vizioz, N. Allouti, F. Laulagnet, O. Gully, N. Bernard-Henriques, M. Bocquet, G. Molas, P. Vivet, D. Querlioz, J. Portal, S. Mitra, F. Andrieu, C. Fenouillet-Beranger, E. Nowak, and E. Vianello, "High-density 3D monolithically integrated multiple 1T1R multi-level-cell for neural networks," in *2020 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2020.

[172] Y. Xi, B. Gao, J. Tang, X. Mu, F. Xu, P. Yao, X. Li, W. Zhang, M. Zhao, H. Qian, and H. Wu, "Impact and quantization of short-term relaxation effect in analog RRAM," in *2020 4th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)*, pp. 1–4, 2020.

[173] B. Q. Le, A. Grossi, E. Vianello, T. Wu, G. Lama, E. Beigne, H.-S. P. Wong, and S. Mitra, "Resistive RAM with multiple bits per cell: Array-level demonstration of 3 bits per cell," *IEEE Transactions on Electron Devices*, vol. 66, pp. 641–646, 2019.

[174] J. Lee, K. Yang, J. Y. Kwon, J. E. Kim, D. I. Han, D. H. Lee, J. H. Yoon, and M. H. Park, "Role of oxygen vacancies in ferroelectric or resistive switching hafnium oxide," *Nano Convergence*, vol. 10, p. 55, 2023.

[175] A. Chen, "Area and thickness scaling of forming voltage of resistive switching memories," *IEEE Electron Device Letters*, vol. 35, pp. 57–59, 2014.

[176] R. Alcala, M. Materano, P. D. Lomenzo, L. Grenouillet, T. Francois, J. Coignus, N. Vaxelaire, C. Carabasse, S. Chevalliez, F. Andrieu, T. Mikolajick, and U. Schroeder, "BEOL integrated ferroelectric HfO_2-based capacitors for FeRAM: Extrapolation of reliability performance to use conditions," *IEEE Journal of the Electron Devices Society*, vol. 10, pp. 907–912, 2022.

[177] L. Grenouillet, J. Coignus, and E. Vianello, "Method for co-fabricating a ferroelectric memory and an OxRAM resistive memory and device co-integrating a ferroelectric memory and an OxRAM resistive memory," Patent US 2023/0133523 A1, 2023.

[178] S.-Y. Wu, C. Chang, M. Chiang, C. Lin, J. Liaw, J. Cheng, J. Yeh, H. Chen, S. Chang, K. Lai, M. Liang, K. Pan, J. Chen, V. Chang, T. Luo, X. Wang, Y. Mor, C. Lin, S. Wang, M. Hsieh, C. Chen, B. Wu, C. Lin, C. Liang, C. Tsao, C. Li, C. Chen, C. Hsieh, H. Liu, P. Chen, C. Chen, R. Chen, Y. Yeo, C. Chui, W. Chang, T. Lee, K. Huang, H. Lin, K. Chen, M. Tsai, K. Chen, X. Chen, Y. Cheng, C. Wang, W. Shue, Y. Ku, S. M. Jang, M. Cao, L. Lu, and T. Chang, "A 3nm CMOS FinFlex™ platform technology with enhanced power efficiency and performance for mobile soc and high performance computing applications," in *2022 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2022.

[179] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "SignSGD: Compressed optimisation for non-convex problems," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 560–569, 2018.

[180] E. Zamanidoost, F. M. Bayat, D. Strukov, and I. Kataeva, "Manhattan rule training for memristive crossbar circuit pattern classifiers," in *2015 IEEE 9th International Symposium on Intelligent Signal Processing (WISP) Proceedings*, pp. 1–6, 2015.

[181] T. Mesquida, F. Rummens, L. Grenouillet, A. Valentian, and E. Vianello, "Computer for executing algorithms carried out from memories using mixed technologies," Patent US 2023/0176816 A1, 2023.

[182] M. Martemucci, F. Rummens, E. Vianello, and T. Hirtzlin, "Hybrid FeRAM/OxRAM data storage circuit," Patent US 2024/0135979 A1, 2024.

[183] M. Martemucci, F. Rummens, T. Hirtzlin, S. Martin, O. Guille, T. Januel, C. Carabasse, O. Billoint, J. Laguerre, J. Coignus, A. F. Vincent, D. Querlioz, L. Grenouillet, S. Saïghi, and E. Vianello, "Hybrid FeRAM/RRAM synaptic circuit enabling on-chip inference and learning at the edge," in *2023 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2023.

[184] G. Moody and R. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, pp. 45–50, 2001.

[185] H. Liu and R. Setiono, "Chi2: feature selection and discretization of numeric attributes," in *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, pp. 388–391, 1995.

[186] P. Zarkeshian, T. Kergan, R. Ghobadi, W. Nicola, and C. Simon, "Photons guided by axons may enable backpropagation-based learning in the brain," *Scientific Reports*, vol. 12, p. 20720, 2022.

[187] J. Koščak, R. Jakša, and P. Sinčák, "Stochastic weight update in the backpropagation algorithm on feed-forward neural networks," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–4, 2010.

[188] A. Salvetti and B. M. Wilamowski, "Introducing stochastic processes within the backpropagation algorithm for improved convergence," *Proceedings of the Artificial Neural Networks in Engineering (ANNIE '94) (Intelligent Engineering Systems Through Artificial Neural Networks)*, vol. 4, pp. 205–209, 1994.

[189] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobilenetV2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[190] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," 2020. arXiv: 1902.08153.

[191] B. Zhuang, C. Shen, M. Tan, L. Liu, and I. Reid, "Towards effective low-bitwidth convolutional neural networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7920–7928, 2018.

[192] N. Haratipour, S.-C. Chang, S. Shivaraman, C. Neumann, Y.-C. Liao, B. G. Alpizar, I.-C. Tung, H. Li, V. Kumar, B. Doyle, S. Atanasov, J. Peck, N. Kabir, G. Allen, T. Hoff, A. Oni, S. Dutta, T. Tronic, A. Roy, F. Hamzaoglu, R. Bristol, M. Metz, I. Young, J. Kavalieros, and U. Avci, "Hafnia-based FeRAM: A path toward ultra-high density for next-generation high-speed embedded memory," in *2022 International Electron Devices Meeting (IEDM)*, pp. 1–4, 2022.

[193] S. D'Agostino, F. Moro, T. Hirtzlin, J. Arcamone, N. Castellani, D. Querlioz, M. Payvand, and E. Vianello, "Synaptic metaplasticity with multi-level memristive devices," in *2023 IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, IEEE, 2023.

[194] T.-J. Yang and V. Sze, "Design considerations for efficient deep neural networks on processing-in-memory accelerators," in *2019 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, 2019.

[195] D. Ielmini and G. Pedretti, "Device and circuit architectures for in-memory computing," *Advanced Intelligent Systems*, vol. 2, p. 2000040, 2020.

[196] F. Andrieu, O. Weber, S. Baudot, C. Fenouillet-Béranger, O. Rozeau, J. Mazurier, P. Perreau, J. Eymery, and O. Faynot, "Fully depleted silicon-on-insulator with back bias and strain for low power and high performance applications," in *2010 IEEE International Conference on Integrated Circuit Design and Technology*, pp. 59–62, 2010.

[197] N. Wang, J. Choi, D. Brand, C.-Y. Chen, and K. Gopalakrishnan, "Training deep neural networks with 8-bit floating point numbers," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, p. 7686–7695, 2018.

[198] X. Sun, J. Choi, C.-Y. Chen, N. Wang, S. Venkataramani, V. V. Srinivasan, X. Cui, W. Zhang, and K. Gopalakrishnan, "Hybrid 8-bit floating point (HFP8) training and inference for deep neural networks," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[199] J. Okuno, T. Kunihiro, K. Konishi, H. Maemura, Y. Shuto, F. Sugaya, M. Materano, T. Ali, M. Lederer, K. Kuehnel, K. Seidel, U. Schroeder, T. Mikolajick, M. Tsukamoto, and T. Umebayashi, "High-endurance and low-voltage operation of 1T1C FeRAM arrays for nonvolatile memory application," in *2021 IEEE International Memory Workshop (IMW)*, pp. 1–3, 2021.

[200] S. J. C. H. Theeuwen and J. H. Qureshi, "Ldmos technology for rf power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 60, pp. 1755–1763, 2012.

[201] F. Rummens, "Low-consumption rram memory differential reading," Patent US 2022/0238154 A1, 2022.

[202] A. Laborieux, M. Bocquet, T. Hirtzlin, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "Implementation of ternary weights with resistive RAM using a single sense operation per synapse," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, pp. 138–147, 2021.

[203] W. Zhao, C. Chappert, V. Javerliac, and J.-P. Noziere, "High speed, high stability and low power sensing amplifier for MTJ/CMOS hybrid logic circuits," *IEEE Transactions on Magnetics*, vol. 45, pp. 3784–3787, 2009.

[204] S. Fusi, P. J. Drew, and L. Abbott, "Cascade models of synaptically stored memories," *Neuron*, vol. 45, pp. 599–611, 2005.

[205] F. T. Zohora, V. Karia, A. R. Daram, A. M. Zyarah, and D. Kudithipudi, "Metaplasticnet: Architecture with probabilistic metaplastic synapses for continual learning," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[206] F. T. Zohora, V. Karia, N. Soures, and D. Kudithipudi, "Probabilistic metaplasticity for continual learning with memristors," 2024. arXiv: 2403.08718.

[207] X. Meng, R. Bachmann, and M. E. Khan, "Training binary neural networks using the bayesian learning rule," in *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[208] M. E. Khan and H. Rue, "The bayesian learning rule," *Journal of Machine Learning Research*, vol. 24, pp. 1–46, 2023.

[209] F. Rummens, T. Mesquida, L. Grenouillet, A. Valentian, and E. Vianello, "Non-volatile memories with mixed OxRAM/FeRAM technologies," Patent US 2023/0186061 A1, 2023.

[210] D. Bonnet, T. Hirtzlin, T. Januel, T. Dalgaty, D. Querlioz, and E. Vianello, "Bayesian metaplasticity from synaptic uncertainty," 2023. arXiv: 2312.10153.