Culture Sciences de l'Ingénieur

La Revue 3E.I

Édité le 24/07/2025

école normale supérieure paris-saclay

L'émergence de l'architecture RISC-V

Théo BALLET[1] – Anthony JUTON[2]

[1] *Doctorant au Centre de Nanosciences et NanoTechnologies (C2N), moniteur à l'ENS Paris-Saclay - DER Sciences de l'Ingénierie Électrique et Numérique*

[2] *Professeur agrégé à l'ENS Paris-Saclay - DER Sciences de l'Ingénierie Électrique et Numérique*

Cette ressource fait partie du N°116 de La Revue 3EI du 3ième trimestre 2025.

Cette ressource présente l'architecture RISC-V, ses applications industrielles et l'intérêt qu'elle présente pour l'enseignement de l'informatique et l'introduction à la conception de circuits intégrés informatiques (co-design). Elle est suivie d'une seconde ressource présentant des exemples pratiques d'implémentation d'un cœur RISC-V sur un FPGA : « Exemples d'implémentation d'un processeur RISC-V sur un FPGA » [1].

1 - Qu'est-ce que RISC-V

Pour qu'un programme puisse fonctionner, sur un certain processeur, il utilise le vocabulaire de l'interface programme/machine de ce processeur. On parle alors de jeux d'instructions (ISA pour Instruction Set Architecture dans la langue de Shakespeare).

Dans les années 70-80 les processeurs étaient souvent programmés directement avec le langage de leur ISA, aussi appelé assembleur (asm). Aussi, pour avoir plus de performance, par exemple pour calculer des fonctions trigonométriques, on ajoutait un élément dans le processeur qui faisait ces calculs et on ajoutait des instructions pour réaliser ces opérations.

Cette philosophie a conduit à une accumulation de composants dans les cœurs des processeurs et à une accumulation d'instructions dans les ISA. On parle donc de processeurs CISC (Complex Instruction Set Computer) parce que leur vocabulaire de base est complexe.

Cette approche avait beaucoup d'avantages pour l'industrie des années 70 qui était une industrie de niche, avec peu de modèles de processeurs (principalement le Zilog z80 et le intel 8008 ainsi que quelques Motorola et Microchip) et peu de programmes à faire tourner. Cependant, dans les années 80, l'arrivée de l'ordinateur personnel en masse va bouleverser ce paradigme.

Il devient alors clair que l'industrie va être amenée à produire un nombre important de processeurs, de même le nombre de programmes grand public va connaître une forte augmentation, l'utilisation de l'assembleur implique qu'il faudra alors reprogrammer chacun de ces programmes pour chaque processeur sorti.

Pire encore, le besoin de performance a amené à l'augmentation du nombre d'instructions que chaque processeur supporte. Ainsi il devient de plus en plus dur pour un programmeur de connaître toutes les instructions d'un processeur.

La solution choisie pour faire face à ces problèmes fut le développement des compilateurs qui permettent de programmer un processeur avec un langage de haut niveau et de traduire ce langage vers l'ISA du processeur. Il n'y a plus besoin de coder chaque programme pour chaque nouveau modèle, et plus besoin de connaître par cœur un ISA.

Seulement les compilateurs originels ne sont pas très efficaces pour gérer les instructions CISC, en effet ces instructions prennent souvent plusieurs cycles pour s'effectuer (on parle de CPI, Clock Per Instruction, en général autour de 10 et plus pour une instruction CISC des années 80), il est donc très compliqué d'optimiser des programmes et de gérer les dépendances entre les instructions.

Des chercheurs de Berkley et Stanford (David Patterson et John Hennessey) ont alors poussé pour le développement de processeurs avec des jeux d'instructions plus petits, on parle de Reduced Instruction Set Computer (RISC). Ces processeurs sont plus performants parce que leurs instructions sont plus efficacement exploitées par les compilateurs. De plus ils sont plus petits et prennent moins de temps à designer, ce qui en pleine époque de la loi de Moore est important.

Au cours des années, l'écart de performance entre les processeurs CISC et RISC s'accroît si bien que les fabricants ont le choix entre se convertir au RISC ou disparaître, AMD et Intel trouveront une solution, traduire leurs instructions CISC en micro-instructions RISC pour rester compatible avec leurs anciens programmes, DEC n'y arrivera que trop tard et sera absorbé par Compaq en 1998 lui-même absorbé en 2002 par HP.

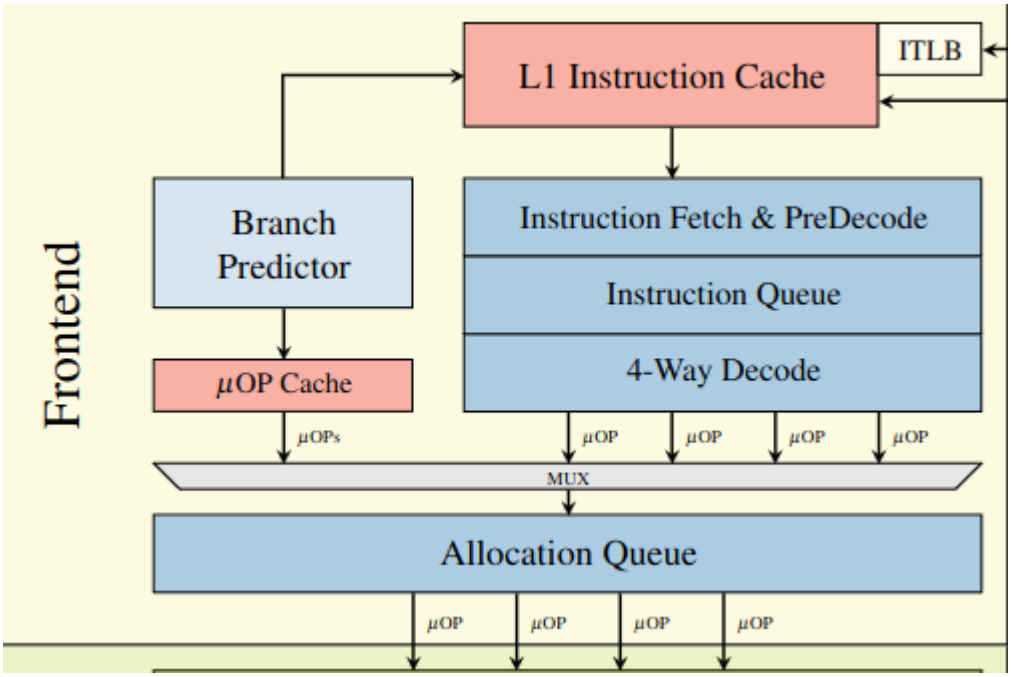

Figure 1 : Schéma bloc de la lecture et de la conversion d'instructions sur processeur x86 Skylake

On voit ici une représentation du premier étage d'un processeur Intel Skylake (2015), les instructions CISC sont lues dans un cache dédié et stockées dans une file d'attente (queue), elles sont ensuite décodées en sous-instructions RISC qui sont envoyées pour exécution aux étages de calcul plus bas.

Puis les années 2000 voient l'apparition d'un marché pour l'architecture 100% RISC : l'embarqué. En effet il y a peu de logiciels conçus pour l'embarqué, les fabricants vont pouvoir choisir leurs ISA sans avoir à se soucier de problèmes de compatibilité avec de vieux logiciels x86. Ils vont alors se tourner vers une entreprise qui vend des licences pour des design de processeurs RISC et leur compilateur : ARM. Cette structure de vente permet de séparer le design de la fabrication de processeurs, aussi elle permet à un fabricant d'obtenir des designs et d'ajouter juste ce dont il a besoin. En somme ARM permet une grande liberté de design et couvre une grande variété de besoins

qui vont de simples contrôleurs d'écran LCD, jusqu'aux cœurs A17 du dernier IPhone codesigné par Apple et ARM.

Pendant ce temps l'idée d'un ISA RISC open-source fait son chemin, et plusieurs projets sont lancés : MIPS, SPARC, SOAR, mais ils ne réussissent pas à accrocher l'industrie ni les chercheurs.

Et puis en 2010 David Patterson lance à Berkeley le projet RISC-V. Face à ARM le projet se veut simple : un ISA gratuit, simple à utiliser et à modifier pour que les chercheurs puissent tester de nouveaux designs. Mais au fur et à mesure que le projet avance, des industriels et des chercheurs de différents milieux se lancent dans des designs RISC-V. On pourra citer parmi les industriels Google puis IBM. Du côté de la recherche Lucas Benini, professeur à ETH Zurich, lance le groupe PULP, qui produira jusqu'à aujourd'hui beaucoup de design open-source allant de simples processeurs comme Snitch, jusqu'à Occamy, un processeur 432 cœurs pour l'inférence IA.

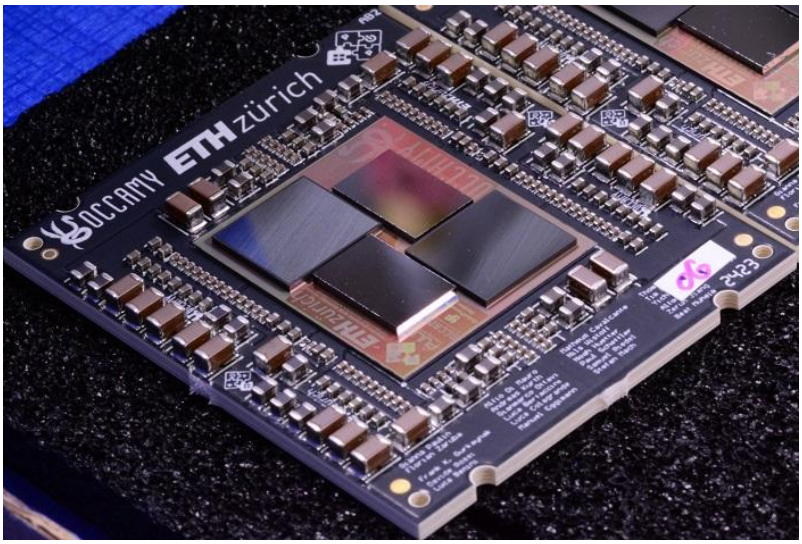

Figure 2 : Le processeur Occamy, un système sur puce 432+2 cœurs RISC-V, avec une mémoire sur puce HBM2e, une véritable prouesse technologique du groupe PULP, designée en moins d'un an par une dizaine de personnes

En 2017 David Patterson et John Hennessy reçoivent le prix Turing pour leur contribution au domaine du design de processeurs, aussi bien autour des projets RISC, et en particulier RISC-V, mais aussi pour l'influence de la publication des tomes de *Computer Architecture : A Quantitative Approach*, un livre très recommandable qui constitue la base de l'architecture des processeurs modernes.

Aujourd'hui (2025) l'architecture RISC domine des pans entiers de l'industrie.

Le succès d'ARM dans l'embarqué est tel que des tentatives apparaissent dans le domaine des serveurs et des PC, ces puces sont souvent des Systèmes Sur Puce (SOC) qui intègrent plusieurs composants, dont des accélérateurs cryptographiques, IA, GPU. et qui reposent sur une spécialisation que les processeurs x86 d'Intel ou AMD peuvent plus difficilement atteindre. Un bon exemple de ce type de percée est la série de processeur M produite par Apple.

De son côté L'architecture RISC-V a réussi à accrocher aussi bien le milieu académique que les industriels, et on voit émerger des puces dans tous les domaines, l'embarqué, le PC, le serveur. Aussi les processeurs RISC-V se sont très bien imposés dans le domaine des sous-composants, ainsi tous les GPU Nvidia modernes ont des contrôleurs RISC-V.

Ce succès est aussi fortement porté par la Chine. En effet celle-ci se trouve sujette à un embargo américain qui limite fortement son accès aux technologies de pointe, que ce soit pour la production

(ASML, TSMC) ou le design (ARM, Cadence). Les technologies OpenSource comme RISC-V sont donc une aubaine pour la Chine dans son effort dans le domaine des semi-conducteurs.

En plus de NVIDIA précédemment cité, on peut noter que le fabricant américain Microchip vend d'ores et déjà des processeurs 64 bits à cœurs RISC-V, l'anglais Raspberry Pi, un microcontrôleur PI Pico 2 avec un cœur ARM et un cœur RISC-V 32 bits, les chinois ESWIN, Kendryte et Starfive, des SoC à cœur RISC-V 64 bits et les chinois Buffalo Lab et WCH, des microcontrôleurs faible coût sur cœur RISC-V 32 bits. Ubuntu et Debian proposent des distributions Linux pour RISC-V.

Concernant le futur, il y a eu beaucoup d'annonces d'industriels sur des investissements vers RISC-V, par exemple avec l'annonce d'une future compatibilité Android sur RISC-V. Le futur nous dira si ces tentatives seront des réussites, mais il est sûr qu'à moyen terme RISC-V aura beaucoup d'opportunités de se développer.

2 - La philosophie RISC-V

Il y a donc eu plusieurs grandes enseignes de design RISC, certains privées comme ARM, d'autres publiques comme MIPS ou SPARC. RISC-V n'est donc pas le premier essai du genre, mais pourtant, on observe un engouement des industriels et du monde académique pour cette architecture. L'objectif de cette partie sera donc d'expliquer les raisons de cet intérêt.

Et à écouter David Patterson il y a plusieurs points importants qui ont rendu RISC-V aussi attirant et facile a utiliser :

2.1 - L'OpenSource

RISC-V est un projet à la base académique, il a donc été conçu pour être partagé. Le standard est accessible à tous, toutes les instructions sont publiques, les outils de compilation/simulation le sont aussi, ce qui n'est pas le cas du x86 et des ISA ARM. N'importe qui avec un ordinateur peut concevoir un design RISC-V et/ou simuler l'exécution d'un programme avec des outils gratuits, ouverts, vérifiés et fiables.

De même beaucoup de designs RISC-V sont libres de droits, vous pouvez très bien récupérer un processeur comme Ibex (produit par PULP), en faire un ASIC et le vendre dans le commerce.

2.2 - La simplicité

La base la plus petite pour un processeur RISC-V ne comporte que 40 instructions, la norme supporte des adresses sur 32 bits, elle requiert au minimum 16 registres généraux de 32 bits chacun.

La simplicité est aussi un avantage pour les industriels, parce qu'elle leur permet de changer d'ISA rapidement.

Mais ce point est aussi un avantage considérable pour l'enseignement, ainsi il est tout à fait envisageable de commencer une formation au design digital par un processeur RISC-V très simple comme Snitch. Comparé à un cours sur un processeur MIPS, le cours utilisant le RISC-V pourra se reposer sur des outils de compilation, de simulation et sur une documentation bien plus fiable et utilisée.

2.3 - La modularité

Les instructions supportées par un processeur RISC-V sont regroupées en "extensions". La grande majorité des processeurs RISC-V disponibles en ligne et utilisés dans l'enseignement sont dits RV32IMC. Ce code se décompose en trois parties :

- La **partie "RV"** qui traduit le fait que notre processeur répond à la norme RISC-V.
- Le **"32"** désigne la taille des registres et de l'espace d'adressage en bits. La norme définit aussi le RV64 et le RV128 pour un espace d'adressage 64 et 128 bits. (Il est à noter que ce sujet est assez contraint pour des raisons historiques et a fait l'objet de beaucoup de débats.)
- La **partie "IMC"** décrit les ensembles d'instructions supportées par le processeur, le "I" est la base de notre jeu d'instructions, les "M"' et "C" sont des extensions qui se rajoutent à cette base des instructions.
    - **"I"** : c'est la base de notre processeur, on y trouve la base de ce qui fait d'un processeur RISC-V une machine dite *turing complète*, c'est-à-dire une machine capable de réaliser n'importe quelle fonction calculable par une machine de Turing. Il existe une base E' qui est une version réduite de la base 'I'.
    - **"M"** : l'extension de multiplication et division des entiers, cette extension rajoute les instructions de multiplication et de division des entiers.
    - **"C"** : l'extension des instructions compressées. Là où un processeur RV32I lit des instructions sur 32 bits, un processeur RV32IC sera capable de lire des instructions de 16 bits. En effet il existe des instructions qui n'ont pas besoin de 32 bits (par exemple celles avec moins d'opérandes). L'avantage est donc que sur un programme simple, on peut quasiment diviser par deux la taille de la mémoire prise par nos instructions.

Cette extension est surtout utile dans un contexte FPGA ou embarqué où la mémoire disponible peut être faible. Si vous donnez un cours et que la mémoire des FPGA pédagogique pose problème, l'utilisation de cette extension peut vous être utile.

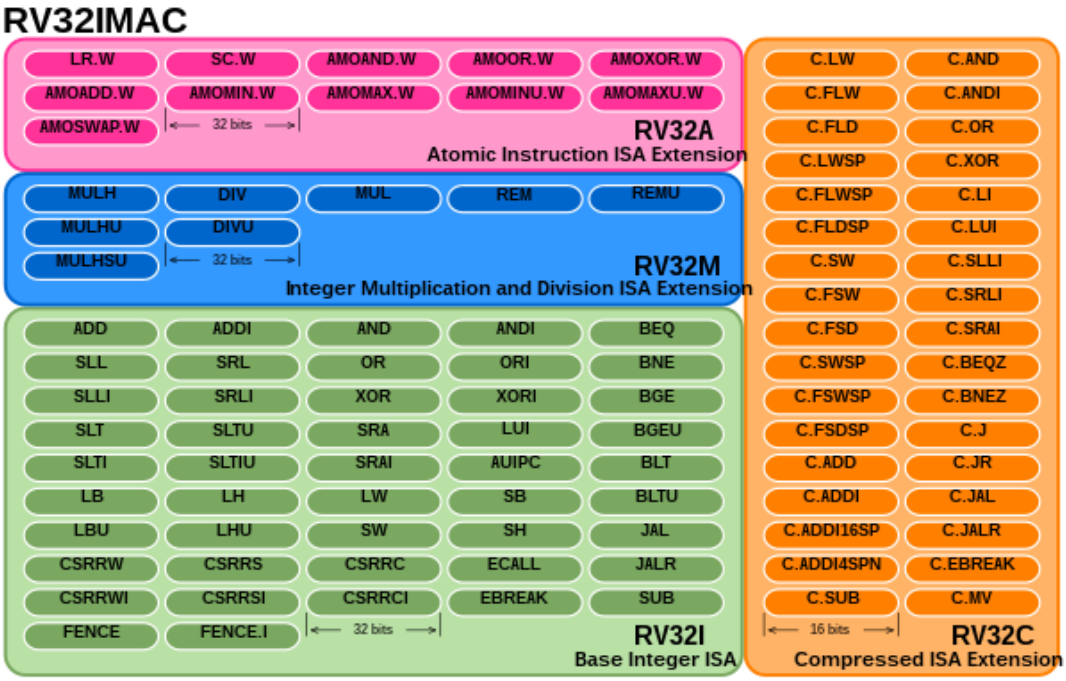

Figure 3 : Instructions supportées par un processeur RV32IMC

Il en existe aussi d'autres : F pour les flottant 32 bits, D pour les flottants 64 bits, Zicsr pour les "control and status registers", X pour des instructions non spécifiées, etc...

Pédagogiquement cette modularité peut permettre de réaliser un projet de design tout au cours d'un semestre en faisant évoluer un design en y ajoutant des modules et les extensions qui correspondent tout en continuant à bénéficier du support de la suite d'outils RISC-V.

2.4 - La stabilité

Les extensions RISC-V sont dites "gelées", en effet ces descriptions ne changent pas dans le temps. Ainsi un programmeur peut aujourd'hui compiler un projet pour un processeur RV32IMC et ce programme fonctionnera toujours sur des processeurs ayant ces extensions des années plus tard.

De même, un design RISC-V de 2014 est normalement encore fonctionnel aujourd'hui.

2.5 - La facilité à spécialiser

La norme RISC-V permet assez facilement d'ajouter des instructions spéciales aux outils de compilation et de simulation, ce qui permet de gagner de la performance très spécifiquement sur les instructions qui nous intéressent sans avoir à payer le coût d'instructions autres qui ne nous intéressent pas. On peut donc adopter une démarche CISC pour certaines applications tout en bénéficiant des avantages du RISC.

Il existe une norme pour ces instructions non spécifiées : l'extension "X". Elle est très bien documentée, et permet d'ajouter assez simplement vos nouvelles instructions au compilateur.

Un exemple de projet pédagogique connu est l'ajout d'une instruction racine carrée à un processeur RISC-V et l'ajout de cette instruction au compilateur.

2.6 - La Communauté

L'ISA RISC-V appartient aujourd'hui à la société à but non-lucratif appelée RISC-V. C'est un consortium d'académiques et d'entreprises qui prennent les décisions sur l'évolution de la norme.

Cette institution, et les engagements de ses membres sont une garantie que le projet RISC-V sera pérenne, et continuera d'être maintenu au fil des années.

2.7 - La sécurité

Au sein de ce consortium il y a plusieurs groupes de travail, le plus actif est celui de la sécurité et de la sûreté. En effet l'OpenSource a longtemps été vu comme un risque de sécurité, le secret des designs et spécifications des puces (ou autres pièces d'ingénierie) était vu comme une protection. Encore aujourd'hui des entreprises comme Nvidia, Intel, AMD et ARM utilisent le secret de leurs designs comme une protection.

Or il faut bien comprendre que cette protection n'a pas vocation à durer dans le temps. À partir du moment où un agent comprend le fonctionnement du système, il peut trouver des failles de sécurité.

Une illustration de ce genre de failles dans le monde des ISA est la découverte en 2020 d'instructions Intel secrètes qui permettent à un agent disposant d'un accès physique à la machine de modifier les programmes exécutés sur cette machine.

Cette faille n'est pas bien grave pour des serveurs, mais dans le monde de l'embarqué elle aurait été sévère.

Le fait que la norme soit OpenSource amène à ce que personne ne soit tenté d'utiliser le secret d'une instruction comme une sécurité, il a donc fallu que le consortium RISC-V fasse un travail de preuves formelles pour démontrer la sécurité de la norme. Celle-ci contient plusieurs recommandations aux designers pour assurer la sécurité de leurs designs, de même la suite d'outils RISC-V contient des outils de vérification de la sûreté et de la sécurité de designs RISC-V.

2.8 - Les défauts de l'ISA RISC-V

Les problèmes de RISC-V existent, bien que les sections précédentes puissent faire penser le contraire, par exemple il n'est pas optimal en termes de densité d'instructions.

Défaut plus important encore : la norme de calcul vectoriel RISC-V tarde à être ratifiée. Les instructions vectorielles ont été figées à date de septembre 2021, là où l'arrivée du calcul SIMD sur x86 date de 1999 ! Il faut bien reconnaître que la fondation RISC-V peut difficilement publier des normes avec la même vitesse que ses concurrents.

Le calcul SIMD réfère à des instructions qui s'appliquent sur plusieurs données en même temps, c'est une conception très efficace dans des applications très intensives en calcul parallélisable. Le calcul SIMD est devenu une nécessité pour tout un tas d'application dont l'IA, et sa réalisation la plus efficace et performante est le calcul vectoriel.

L'attente de cette norme fut donc très longue, et ceux qui voulaient travailler sur l'IA furent obligés d'utiliser des moyens détournés et moins efficaces que le calcul vectoriel : en faisant des clusters de CPU, ou en ajoutant des coprocesseurs SIMD, qui utilisent des ISA dédiées. Ces solutions sont soit moins performantes, soit doivent être programmées en assembleur. Et programmer en assembleur des fonctions pour faire de l'IA est une tâche peu enviable.

3 - Exploitation pédagogique de l'architecture RISC-V

L'ouverture de RISC-V en fait une architecture intéressante pour l'enseignement de l'architecture des processeurs et pour initier les étudiants à la conception de circuits intégrés informatiques.

MounRiver Studio (https://mounriver.com/) est un environnement de développement dédié à la programmation et au débogage des petits microcontrôleurs WCH à cœur RISC-V 32 bits. La revue Elektor a fait un essai dans son numéro de mai 2025.

Hervé Discours propose sur sa chaîne YouTube un exemple d'exploitation, basée sur son expérience en BUT GEII, de la carte Maixduino utilisant le puissant SoC Kendryte K210 à double-cœur RISC-V 64 bits : https://www.youtube.com/watch?v=C-kiOvwal0s

Pour entrer plus en profondeur dans le cœur RISC-V et lui ajouter des périphériques, il est intéressant de travailler sur son implémentation dans un FPGA, à partir d'une IP VHDL ou Verilog.

Il est possible d'utiliser la simulation. ModelSim/QuestaSim est très connu, mais il existe aussi des alternatives open source gratuites, comme Verilator (pour le Verilog et le SystemVerilog), avec des exemples d'implémentation de cœur RISC-V.

La deuxième méthode, objet de la ressource « Exemples d'implémentation d'un processeur RISC-V sur un FPGA », est l'implémentation sur FPGA, les cartes pédagogiques (Altera DE10, Digilent Basys3

par exemple) étant suffisantes pour faire tourner un cœur RISC-V. Le dépôt Git de Jacques-Olivier Klein est un autre exemple d'exploitation de l'implémentation d'un cœur RISC-V sur FPGA pour l'enseignement en BUT GEII : https://github.com/JOKleinGe1/Module_Initiation_Riscv.

La troisième méthode est de loin la plus complexe et la plus chère : la fabrication d'une puce sur silicium (ASIC). Cette méthode nécessite des suites d'outils de design très onéreux même pour le milieu éducatif (Cadence, Synopsys). Il est important de noter que là aussi des outils Opensource émergent, comme OpenLane : une suite d'outils développés par Google et basés sur des spécifications de fabrication Opensource venant de l'entreprise Skywater. Aujourd'hui ce design kit 130nm est complètement fini et utilisé dans l'industrie.

Cette ressource donne un premier aperçu de l'architecture RISC-V, ouverte et très bien documentée. Les lecteurs intéressés pourront approfondir avec les nombreux exemples d'implémentations sur FPGA dont un fourni dans ce numéro 116 de la revue 3EI [1]. Un retour sur l'usage d'un microcontrôleur basé sur RISC-V pour l'enseignement serait le bienvenu pour un prochain numéro de la revue.

Référence

[1] : Exemples d'implémentation d'un processeur RISC-V sur un FPGA, T. Ballet, juillet 2025, https://sti.eduscol.education.fr/si-ens-paris-saclay/ressources_pedagogiques/exemples-dimplementation-dun-processeur-riscv-sur-un-fpga