Alma Mater Studiorum · Università di Bologna

Dipartimento di Fisica e Astronomia "Augusto Righi"

Corso di Laurea in Fisica

DEVELOPMENT OF NEUTRON TRANSPORT SIMULATION PROCESSES ON FPGA-BASED HARDWARE ACCELERATORS FOR FOURTH-GENERATION NUCLEAR POWER PLANTS

Relatore:

Prof. Alessandro Gabrielli

Correlatore:

Dott. Fabrizio Alfonsi

Presentata da:

Gian Marco Coppari

Anno Accademico 2024/2025

Abstract

The aim of this thesis is to re-examine the computational models currently used in neutron diffusion simulations in uranium-boron materials in order to optimise their computational performance. It is well known that open source simulation framework such as OpenMC require high computational performance because they typically simulate the diffusion of thousands of particles, with variable initial energy, within materials of known composition. Descriptive and simulation software systems are typically designed with high-level abstraction languages such as Python and C++. Despite their descriptive versatility, these languages are not optimised in terms of computational efficiency and, for this reason, require increasingly high computational performance for their execution, such as that obtained through large High Performance Computing centres. My personal contribution to this thesis reproduces neutron diffusion simulations using high-level description algorithms implemented on commercial hardware platforms. This work was carried out using firmware design codes, applying and exploiting the intrinsic parallelism of FPGA devices, thereby accelerating calculation times by approximately two orders of magnitude.

Sommario

L'obiettivo di questa tesi è riesaminare i modelli computazionali attualmente utilizzati nelle simulazioni di diffusione neutronica in materiali contenenti miscele di uranio-boro, al fine di ottimizzarne le prestazioni computazionali. È noto che i framework di simulazione open source come OpenMC richiedono elevate prestazioni di calcolo, poiché tipicamente simulano la diffusione di decine di migliaia di particelle, con energia iniziale variabile, all'interno di materiali di composizione nota. I sistemi software descrittivi e di simulazione sono solitamente progettati con linguaggi ad alto livello di astrazione come Python e C++. Nonostante la loro versatilità descrittiva, questi linguaggi non sono ottimizzati in termini di efficienza computazionale e, per questo motivo, richiedono prestazioni di calcolo sempre più elevate per la loro esecuzione, come quelle ottenute tramite i grandi centri di calcolo ad alte prestazioni HPC. Il mio contributo personale a questa tesi riproduce simulazioni di diffusione neutronica utilizzando algoritmi di descrizione ad alto livello implementati su piattaforme hardware commerciali. Questo lavoro è stato condotto utilizzando codici di progettazione firmware, applicando e sfruttando il parallelismo intrinseco dei dispositivi FPGA, accelerando così i tempi di calcolo di circa due ordini di grandezza.

Contents

List of Figures

List of Tables

List of Codes

Introduction

The renewed industrial interest in advanced Generation IV (Gen–IV) nuclear systems, led by companies such as *new*cleo Company, has redefined the computational requirements for reactor design and safety analysis. The deployment of innovative technologies, such as LFRs and accelerator-driven systems, demands rapid prototyping and iterative assessments. For these industrial applications, reducing high-fidelity simulation times from weeks to hours is an essential requirement. While the Monte Carlo method remains the reference standard for solving the neutron transport equation — due to its exact three-dimensional and energetic resolution — its traditional software execution on general-purpose CPUs suffers from severe scalability limits. The disordered memory access patterns inherent in stochastic particle tracking neutralize cache efficiency, creating a *memory wall* that renders standard Von Neumann architecture execution incompatible with the strict timelines of modern nuclear engineering.

To satisfy this urgent industrial requirement, a paradigm shift in computing architectures is necessary. Field-Programmable Gate Arrays (FPGAs) offer a novel computational approach to this problem: *pipeline* and *spatially parallel computing*. Rather than sequentially executing an instruction stream, FPGAs allow the Monte Carlo algorithm to be mapped directly as a synchronous digital dataflow circuit. This approach unrolls complex computational loops into deep physical pipelines, processing a continuous stream of particles with deterministic latencies and entirely bypassing traditional processor bottlenecks.

Motivated by this explicit industrial need and the innovative potential of spatial computing, this thesis work is fundamentally justified by the requirement to develop faster, scalable hardware accelerators for neutron transport simulations of Gen–IV. The primary objective is to demonstrate the engineering feasibility of migrating the Monte Carlo algorithm, and in particular the OpenMC open-source project, for neutron transport toward a spatially parallel FPGA architecture. Rather than performing a literal porting of existing software codes into hardware description languages, this project establishes a *proof of concept* for a new hardware-native tracking engine. The goal is to decouple the evaluation of stochastic mathematics and interaction kinematics from the architectural constraints of standard processors, developing a datapath capable of handling complex physical phenomena, overcoming traditional CPUs and GPUs bottlenecks.

The present thesis is organized to guide the reader through the process of translating nuclear Physics into digital circuit operating on electric signals. In order to present the theory needed to understand this new programming paradigm, justify why this new technology require high fidelity simulations and highlight why traditional algorithm and computational architecture are inadequate to solve these challenges, the text is divided into five chapters, starting from the underlying Physics mechanisms governing the nuclear reactor, passing through stochastic solvers up to firmware analysis and evaluation, in the following order.

Chapter 1 introduces the theoretical foundations of neutron-matter interaction, keeping the focus on interaction relevant for neutron transport in reactor's core. It describes the physical quantities involved, introduces the Linear Transport Equation (LTE), mentioning the traditional mathematical tools used to solve it. A toy example is then used to highlight why these methods lack the required precision.

Chapter 2 analyses the inherent limitations of (Gen-III) reactors and illustrates how the technologies proposed for Gen–IV systems address these issues. Specifically, it examines the transition from thermal to fast neutron spectra and outlines the new engineering challenges introduced by this paradigm shift. Rather than serving as an exhaustive engineering reference, this chapter aims to provide a contextual overview of the industrial landscape and the specific application domain that motivate the present work.

Chapter 3 is dedicated to the foundation of Monte Carlo method. It highlights the elegance in solving numerical issues related to the traditional deterministic solvers and provides a non-exhaustive description of how it works. Few, focused examples are used to relate physical quantities of Chapter 1 with stochastic computation technique. At the end efficiency problems strictly related to the architecture on which this algorithm runs are presented. This serves as motive for the choice of porting Monte Carlo into dedicated hardware chip.

Chapter 4 pause on describing features and technicalities of the target electronic device chosen for the development of this work: the FPGA. Exploring the design and the available resources of commercial FPGAs the reader is guided towards why it has been chosen for this purpose: many of the choices made during the project development were based on natively supported operations and data representation.

Chapter 5 presents both the structure of the project and the collected results. Focus is posed on the new paradigm of programming, rather than on code itself, highlighting how high-level C++ code is mapped into solid silicon hardware. Next, both preliminary validation tests and final performance evaluation are presented, marking at the same time the limitation intentionally introduced to gain a first

prototype of this new programming paradigm. The hardware prototype is directly compared with an equivalent execution of the reference software OpenMC.

Finally, **Conclusions** summarize the achievements of the project, highlighting the current limitations of the implementation and outlining the roadmap for future developments, with particular attention to the integration of continuous-energy physics and hardware tallying systems.

1 Neutron Transport in Nuclear Reactor

In this chapter the required mathematical foundations to understand neutron transport mechanism in the core of a nuclear reactor are presented. The description starts with a brief introduction to the statistical interpretation of microscopic-cross section and a quick overview of neutron-matter interaction relevant inside reactor's core, then proceeds with the formulation of the macroscopic Boltzmann Linear Transport Equation (LTE) and finally, a simple, but not trivial, problem is presented in order to show the limitations of the deterministic mathematical approach in solving a real-life neutron transport problem.

1.1 Neutron-Matter Interaction in Reactor's Core

The behaviour of a nuclear reactor is essentially dictated by the interaction between *free neutrons* and the atomic nuclei composing the reactor's core. One can quantify these interactions via the *cross section*, a metric directly related to the physical likelihood of a particular event occurring. Let r label one of the possible reactions a neutron can undergo with the surrounding material, and let σ_r be the corresponding microscopic cross section for that process. The *total (microscopic) cross section* is then defined as the sum of all cross sections over the allowed set of processes:

$$\sum_r \sigma_r = \sigma_{\text{tot}} \tag{1.1}$$

Physically, the ratio of a specific cross section to the total cross section, $\sigma_r/\sigma_{\text{tot}}$, represents the relative likelihood, also referred to as *branching ratio*, that a neutron will undergo reaction r given that an interaction has occurred. Because interaction channels are mutually exclusive – a neutron undergoing elastic scattering can not simultaneously disappear through an absorption event – Eq. (1.1) ensures that all possible physical outcomes are accounted for.

It is a well–known result in nuclear and particle physics that many reaction channels exhibit a threshold energy, which is determined by the reaction Q–value. It represents the net energy released (or absorbed, if negative) in the process and, for endothermic reactions, sets the minimum kinetic energy required for the reaction to occur in the laboratory frame [1, 2]. This hints that cross sections are energy-dependent functions rather then simple numbers and Eq. (1.1) should be written as

$$\sum_r \sigma_r(\epsilon) = \sigma_{\text{tot}}(\epsilon) \tag{1.2a}$$

$$\sum_r \wp_r(\epsilon) = 1 \tag{1.2b}$$

It is also worth noting that the index r labelling the allowed interactions will change with energy, as not all the possible interactions are allowed for a given neutron energy ϵ.

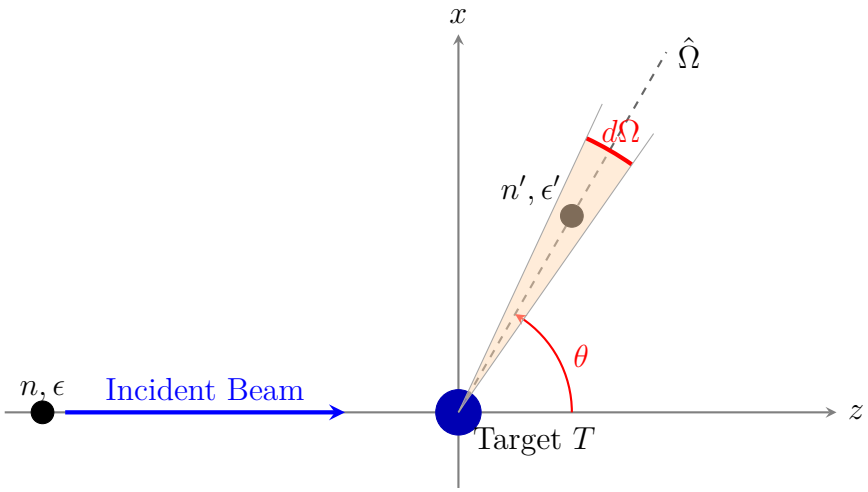

Figure 1.1. Schematic representation of neutron-nucleus inelastic scattering in the laboratory reference frame. An incident neutron n with energy ϵ approaches the target nucleus T along the z-axis. Following the collision, the scattered neutron n' emerges with energy ϵ' at a polar angle θ with respect to the incident direction. The detector subtends a differential solid angle $d\Omega$ around the scattered direction $\hat{\Omega}$.

In an High Energy Physics (HEP) experiment, this probability is computed by *counting* how many particle are produced after the interaction has occurred through channel r. Since a real-life detector only covers a portion $d\Omega$ of the solid angle surrounding the interaction point, as shown in Fig. 1.1, what one actually measures is the *doubly differential cross section* as a function of *incoming beam energy* and *solid angle*, i.e. $\sigma_r(\epsilon, \Omega)$, measuring the probability that the incoming particle of energy ϵ will produce particles in the solid angle $d\Omega$ around direction Ω through process r. Clearly, integration over the full solid angle restores the interaction probability σ_r for interaction channel r, namely

$$\sigma_r(\epsilon) = \int_{4\pi} \sigma_r(\epsilon, \Omega)\, d\Omega \tag{1.3}$$

where 4π indicated the unit sphere surrounding the interaction point. From a physical point of view, this directional dependence of the cross-section allow to describe the *anisotropy* of some neutron-matter interaction, such as fast neutron collision with heavy nuclei: this particular type of interaction shows a forward-peaked distributions in the Laboratory Reference Frame (Lab RF), as it well described in [3].

Dealing with neutron-matter interactions and nuclear reactor physics it is conve-

nient to recursively split cross sections into categories, each composed of a variety of fundamental reactions, as depicted in Fig. 1.2. The first major separation is between *collisions* and *absorption processes*: former ones contains all the *scattering reactions* where the incident neutron also appears as a final product of the process, whereas the latter ones encode all those processes where the neutron disappears after the interaction.

$$\sigma_{\text{tot}}(\epsilon) = \sigma_s(\epsilon) + \sigma_a(\epsilon) \tag{1.4}$$

Fig. 1.2 shows how total microscopic cross-sections fork into elementary processes, which will be analysed individually in the following.

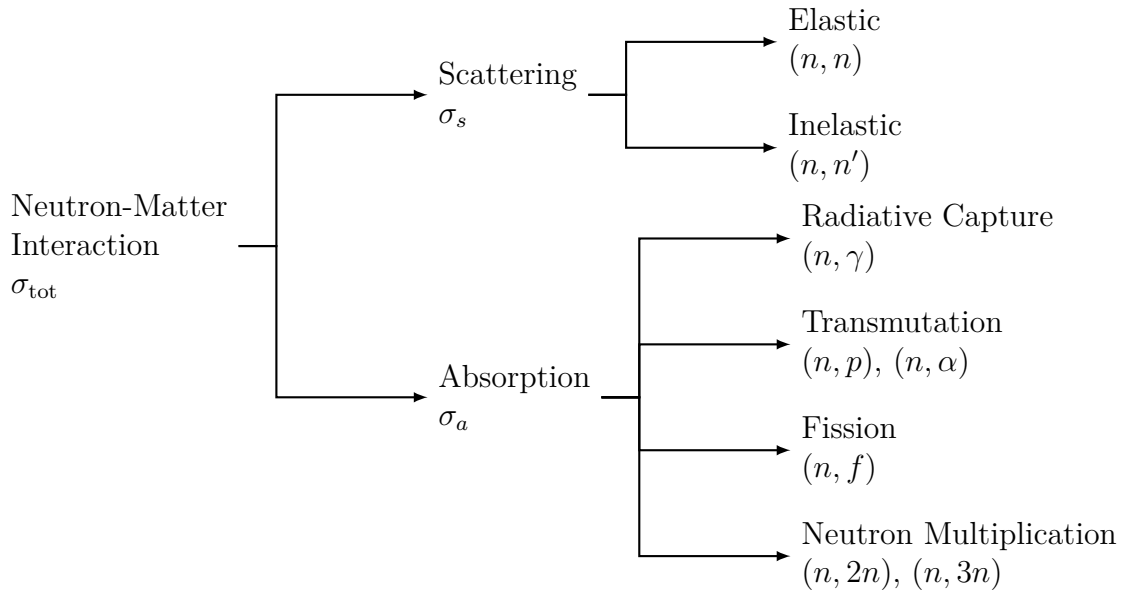

Figure 1.2. Tree diagram of neutron-matter interaction cross sections in nuclear reactors. The total cross section σ_{tot} is hierarchically decomposed into scattering and absorption contributions, each of which further forks into their respective elementary physical processes.

Scattering and Moderation Among the first group one finds *scattering reactions*, which are the fundamental drivers of neutron moderation. These collisions can be broadly categorized into *elastic scattering*, where the total kinetic energy of the system is conserved in the Zero-Momentum Reference Frame (ZMRF) [4] and the target nucleus is left unchanged, and *inelastic scattering*, where kinetic energy is not conserved because the target nucleus absorbs a portion of it, being left in an *excited state*. Using standard nuclear physics notation, these processes are represented respectively by

$${}^{\text{A}}_{\text{Z}}\text{X}\,(n, n)\,{}^{\text{A}}_{\text{Z}}\text{X} \tag{1.5a}$$

$$ {}^{A}_{Z}\mathrm{X}\,(n,n')\,{}^{A}_{Z}\mathrm{X}^{*} \tag{1.5b} $$

where ${}^{A}_{Z}\mathrm{X}^{*}$ indicates an excited state of the target nuclide ${}^{A}_{Z}\mathrm{X}$, and n' represents the scattered neutron emerging with a sharply reduced kinetic energy. Because inelastic scattering requires sufficient energy to excite the nucleus to its first discrete quantum level, it is a threshold reaction [1]. The excited nucleus subsequently undergoes de-excitation via prompt γ-ray emission to return to its ground state:

$$ {}^{A}_{Z}\mathrm{X}\,(n,n')\,{}^{A}_{Z}\mathrm{X}^{*} \longmapsto {}^{A}_{Z}\mathrm{X}^{*} \rightarrow {}^{A}_{Z}\mathrm{X} + \gamma \tag{1.6} $$

These two processes are essential in thermal nuclear reactors as they govern the entire macroscopic slowing-down dynamic, referred to as *moderation*. Fast neutrons typically possess kinetic energies in the range of $\epsilon \sim 0.1$ to 20 MeV. At these high energies, inelastic scattering with heavy fuel isotopes, such as Uranium-238, provides the primary braking mechanism, rapidly dropping the neutron energy below the inelastic threshold, around 100 keV for heavy nuclei) [5].

Once neutrons fall below this threshold and enter the *resonance region*, inelastic scattering becomes kinematically forbidden: here, the dominant slowing-down process becomes elastic scattering. To be efficient, elastic collisions require light moderator nuclei – such as Hydrogen, Deuterium, or Carbon, typically found in water, heavy water, or graphite. Successive elastic collisions with these light targets exhibit a high probability of transferring large fractions of kinetic energy, effectively pushing the neutrons down the energy spectrum.

Collisions in this lower energy range further reduce the neutron energy below 1 eV, at which point the neutron is said to be *thermalized*. In this final stage, the neutron's kinetic energy becomes comparable to the thermal agitation energy of the surrounding medium, and its de Broglie wavelength becomes comparable to interatomic distances. Consequently, elastic collisions now take place between the neutron and the atomic lattice or the molecule as a whole bound system, rather than with isolated nuclei. Because the target atoms are vibrating with thermal energy, a thermalized neutron can also gain energy during a collision if it interacts with a highly excited molecule – a crucial kinematic process known as *up-scattering*. Ultimately, a state of thermodynamic equilibrium is reached, characterized by a Maxwell-Boltzmann energy distribution [5]. Now that the energy of the neutron has been minimized, it takes full advantage of the v^{-1} cross-section behaviour, resulting in a maximized probability of undergoing fission reactions with heavy fissile nuclides, such as ^{235}U, which is strictly necessary for maintaining the fission chain reaction.

Next, absorption processes can be broadly partitioned into four main categories

relevant for reactor physics, namely *radiative capture*, *transmutation*, *fission* chain and *multiplication* processes. In subsequent paragraphs each of these reactions is briefly analysed.

Radiative Capture The first class of processes, formally denoted as (n, γ), is ubiquitous across the neutron energy spectrum and is primarily responsible for both the *breeding* (or fertilization) of new fuel and the *poisoning* of the reactor core. The fundamental reaction involves the absorption of an incident neutron by a target nucleus to form a compound nucleus in a highly excited state. The excitation energy is subsequently radiated away via prompt gamma emission, allowing the nucleus to settle into its ground state:

$$ {}^{\mathrm{A}}_{\mathrm{Z}}\mathrm{X} + n \longrightarrow {}^{\mathrm{A+1}}_{\mathrm{Z}}\mathrm{X}^{*} \longrightarrow {}^{\mathrm{A+1}}_{\mathrm{Z}}\mathrm{X} + \gamma \tag{1.7} $$

Inside nuclear reactors, this capture mechanism plays a dual, critical role. On one hand, the progressive accumulation of high cross-section fission products, such as ^{135}Xe and ^{149}Sm, leads to massive parasitic neutron absorption, a phenomenon known as *poisoning* [6]. On the other hand, radiative capture triggers the fundamental fertilization sequence involving the *fertile* isotope Uranium-238. The capture of a neutron (typically in the epithermal resonance region) produces Uranium-239, which is unstable and undergoes a sequence of two consecutive β^- decays to yield the *fissile* isotope Plutonium-239. The detailed sequence is the following:

$$ {}^{238}_{92}\mathrm{U} + {}^{1}_{0}\mathrm{n} \rightarrow {}^{239}_{92}\mathrm{U} + \gamma \tag{1.8a} $$

$$ {}^{239}_{92}\mathrm{U} \xrightarrow[23.5\,\mathrm{min}]{\beta^-} {}^{239}_{93}\mathrm{Np} + e^- + \overline{\nu}_e \tag{1.8b} $$

$$ {}^{239}_{93}\mathrm{Np} \xrightarrow[2.36\,\mathrm{d}]{\beta^-} {}^{239}_{94}\mathrm{Pu} + e^- + \overline{\nu}_e \tag{1.8c} $$

The relatively short half-lives of the intermediate isotopes (^{239}U and ^{239}Np) compared to the extremely long half-life of the resulting ^{239}Pu, approximately 24110 y, mean that the breeding process acts as a delayed source of new fissile material. As the reactor operates, this newly generated Plutonium is effectively stored inside the reactor core, where thermal neutrons can, in their turn, interact with these nuclei to induce secondary fissions, extending the fuel cycle and deeply affecting the long-term neutron balance.

Transmutation The next class of processes encompasses *transmutation reactions*, occurring when an incident neutron is absorbed by a target nuclide, forming an

excited compound nucleus that subsequently de-excites by emitting a charged particle. The general representation is

$$ {}^{\mathrm{A}}_{\mathrm{Z}}\mathrm{X}\left(n, {}^{\mathrm{A}'}_{\mathrm{Z}'}\mathrm{W}\right) {}^{\mathrm{A}+1-\mathrm{A}'}_{\mathrm{Z}-\mathrm{Z}'}\mathrm{Y} \tag{1.9} $$

where ${}^{A'}_{Z'}W$ denotes the emitted charged particle (with total electric charge $Z'e$), predominantly protons, deuterons, tritons or alpha particles. Because the exiting charged particle must overcome or tunnel through the Coulomb barrier of the nucleus, these reactions typically exhibit high energy thresholds for heavy isotopes. However, for certain light nuclei, ^{10}B and ^{6}Li above all, specific transmutation channels are highly exothermic and occur readily at thermal energies.

A paramount example in reactor physics is the (n, α) reaction, generally described by:

$$ {}^{\mathrm{A}}_{\mathrm{Z}}\mathrm{X}\,(n, \alpha)\, {}^{\mathrm{A}-3}_{\mathrm{Z}-2}\mathrm{Y} \tag{1.10} $$

which remarkably specializes into the neutron-induced disintegration of Boron-10:

$$ {}^{10}_{5}\mathrm{B}\left(n, {}^{4}_{2}\alpha\right) {}^{7}_{3}\mathrm{Li} \tag{1.11} $$

Unlike scattering materials that moderate neutrons, ^{10}B acts as a powerful *neutron poison* or *reactivity control material*. It boasts a massive thermal absorption cross section, approximately 3840 b at 0.0253 eV [7], that strictly follows a v^{-1} behavior. This characteristic makes it one of the major contributors to neutron control: by capturing thermalized neutrons before they can induce fission in Uranium, it allows operators to efficiently suppress the excess multiplication factor of the fission chain reaction.

In commercial nuclear engineering, ^{10}B is deployed in two primary forms to govern the neutron economy. For rapid, localized control and safe shutdown, it is utilized in solid-state neutron-absorbing control rods, typically compounded as Boron Carbide (B_4C). Alternatively, for spatially uniform, long-term reactivity control it is dissolved directly into the liquid moderator of Pressurized Water Reactors (PWRs) in the form of boric acid, H_3BO_3.

Neutron-Induced Fission Among all neutron-nucleus interactions, *neutron-induced fission* is the cornerstone of nuclear reactor physics. When a neutron strikes a heavy fissile nucleus (such as ^{235}U or ^{239}Pu), it can be absorbed to form a highly excited compound nucleus. According to the Liquid Drop Model [1], this excitation energy induces violent internal oscillations, causing the nucleus to deform and ultimately undergo scission into two lighter, highly radioactive intermediate-mass

nuclei, collectively referred to as *fission fragments*, according to the following process

$$\mathrm{X} + n \rightarrow \mathrm{F_H} + \mathrm{F_L} + \nu n + Q \tag{1.12}$$

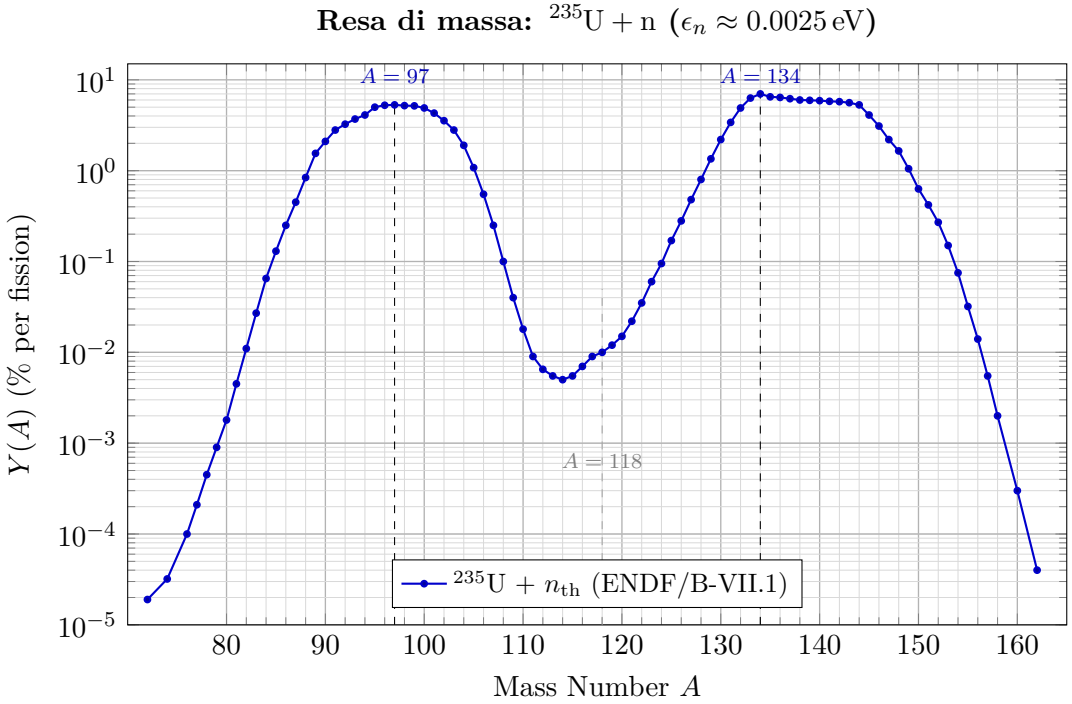

Figure 1.3. Fission product mass yield for ^{235}U for impinging thermal neutrons. The typical double hump shape is evident, showing peaks at $A_L = 97$ and $A_H = 134$, the *light* and *heavy peaks*, respectively. Nuclear data taken from ENDF/B-VII.1 library.

where F_H and F_L denote the heavy and the light fission nuclei, ν is the *neutron yield* of the scission process, and Q is the released energy. The L and H distinction is necessary as the split of the fissile nucleus X is not strictly symmetric but follows a *double-hump* shaped distribution, as depicted in Fig. 1.3. For the specific case of ^{235}U, Eq. (1.12) specializes perfectly into the generation of Strontium-99 and the neutron poison Xenon-135:

$$^{235}\mathrm{U} + n \rightarrow {}^{99}_{38}\mathrm{Sr} + {}^{135}_{54}\mathrm{Xe} + 2n + Q \tag{1.13}$$

The scission event releases an amount of energy approximately 200 MeV per fission, which manifests primarily as the kinetic energy of the massive fragments. In addition, the reaction instantly emits prompt gamma rays and an average of two to three fast neutrons, which are essential for sustaining the chain reaction.

Subsequently, the neutron-rich fission fragments undergo a series of β^- decays.

These decays release delayed neutrons and a massive, continuous flux of electron antineutrinos. Because a commercial nuclear power plant produces such an intense, localized, and well-characterized directional flux of antineutrinos, reactors serve as invaluable artificial sources for fundamental particle physics. Consequently, reactor antineutrinos have been extensively utilized in groundbreaking experiments, such as KamLAND, to study the phenomenon of neutrino oscillation [8].

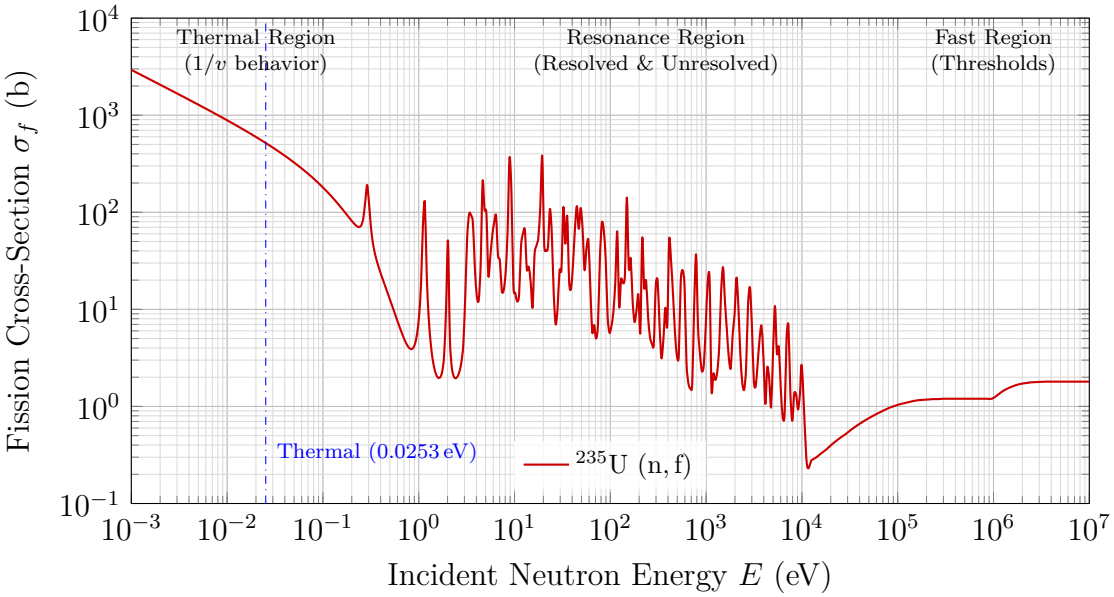

Figure 1.4. Pictorial representation of the ^{235}U microscopic fission cross-section $\sigma_f(E)$.

A paramount physical aspect of this reaction is the strong energy dependence of the microscopic fission cross section, $\sigma_f(\epsilon)$. For fissile isotopes, σ_f exhibits a marked v^{-1} behaviour in the low-energy region, as can be easily seen in Fig. 1.4. This implies that the probability of inducing a fission event is orders of magnitude higher for slow-moving thermal neutrons than for the fast neutrons born directly from the fission event itself.

Multiplication The last category which must be included when evaluating the total absorption cross section is *neutron multiplication*. From a rigorous kinematic perspective, these reactions proceed via an intermediate compound-nucleus mechanism: the incident neutron is completely absorbed and loses its identity, meaning the neutrons in the final state cannot be treated as the same initial particle. Unlike fission, however, $(n, \kappa n)$ reactions are strictly *endothermic* [1]. They require the impinging neutron to possess a kinetic energy high enough to overcome the binding energies of the emitted nucleons. Consequently, they exhibit high threshold energies (i.e., a

highly negative Q–value), generally limiting their impact to the upper fast-energy tail of the neutron spectrum in thermal reactors. At the same time, they act as crucial non-conservative processes that serve as a source of secondary neutrons through the general scheme

$$ {}^{\mathrm{A}}_{\mathrm{Z}}\mathrm{X}\,(n,\kappa n)\,{}^{\mathrm{A}+1-\kappa}_{\mathrm{Z}}\mathrm{X} \tag{1.14} $$

where κ represents the neutron multiplicity, typically 2 or 3. A notable and widely exploited exception to the high-threshold rule is Beryllium-9. Thanks to its loosely bound nuclear structure, ^{9}Be can undergo the multiplication reaction

$$ {}^{9}_{4}\mathrm{Be}\,(n,2n)\,{}^{8}_{4}\mathrm{Be} \rightarrow 2\,{}^{4}_{2}\alpha + 2n \tag{1.15} $$

for incident neutrons with energies greater than 1.66 MeV. This relatively low threshold allows Beryllium to act as an excellent neutron multiplier and reflector, effectively supplying additional neutrons to the reactor core and enhancing the overall neutron economy.

1.2 Neutron Transport Equation

1.2.1 Transition to Macroscopic Reactor Geometry

The interaction between neutrons and matter at the microscopic level can be described in terms of reaction cross sections. In reactor physics, however, one is primarily interested in macroscopic observables such as reaction rates, power distributions, and neutron fluxes inside extended three-dimensional geometries. To connect these two levels of description, it is necessary to transition from the isolated nucleus to the macroscopic bulk material, and subsequently introduce a statistical description of neutron populations.

To describe macroscopic quantities, one must first connect the microscopic cross sections σ_r associated with fundamental neutron-nucleus interactions to the *macroscopic cross sections* Σ_r, which govern the interaction of neutrons with a finite amount of material. Just as the microscopic total cross section gives the probability that a neutron interacts with a single nucleus, the macroscopic total cross section $\Sigma_{\mathrm{tot}}(\boldsymbol{r},\epsilon)$ yields the probability of interaction per unit path length for a neutron traveling inside a material. The relation connecting these two quantities is:

$$ \Sigma_r(\boldsymbol{r},\epsilon) = N(\boldsymbol{r})\,\sigma_r(\epsilon) = \frac{\rho(\boldsymbol{r})\,N_A}{M(\boldsymbol{r})}\sigma_r(\epsilon) \tag{1.16} $$

where N is the *atomic density*, ρ is the mass density, M is the molar mass of the local material, and N_A is the Avogadro constant.

Lastly, one should note that Σ_r has units of an inverse length, i.e. $[\Sigma_r] = [L]^{-1}$. This might appear counterintuitive for a quantity referred to as a cross section, but this situation can be easily understood by means of *surviving probability*. Indeed, the probability of a neutron surviving without collisions falls off exponentially as it moves inside a uniform medium, i.e. $\wp(x) \propto \exp(-\Sigma_{\text{tot}} x)$. As rigorously derived in [9, p. 439], the expectation value of the distance traveled between two consecutive interactions, known as the *mean free path*, is simply:

$$\langle x \rangle = \frac{1}{\Sigma_{\text{tot}}} \tag{1.17}$$

Note also that this reasoning holds true only when considering the *total* macroscopic cross section, since the specific sequence of underlying microscopic processes remains purely stochastic.

It is worth pausing to understand how these quantities behave inside a real reactor, spending a few words on the chemical nature of a reactor core. It is a highly heterogeneous environment, filled with fuel rods, coolant, and control elements of widely varying compositions. In this setup, $M(\boldsymbol{r})$ exhibits sharp *jump discontinuities* when neutrons cross the boundaries separating different materials (e.g., from solid fuel to liquid coolant). Moreover, the mass density $\rho(\boldsymbol{r})$ can also vary continuously within the same material, for instance due to thermal expansion of the coolant.

Finally, the generalization to multi-isotope materials, alloys, or complex molecules (such as water or enriched uranium dioxide) is straightforward. Since macroscopic cross sections represent independent interaction probabilities per unit path length, the macroscopic cross section of a mixture is rigorously obtained by the linear superposition of the individual isotopic contributions:

$$\Sigma_r(\boldsymbol{r}, \epsilon) = \sum_i N_i(\boldsymbol{r}) \, \sigma_{r,i}(\epsilon) \tag{1.18}$$

where $N_i(\boldsymbol{r})$ is the local atomic number density of the i-th specific isotope or element within the compound.

1.2.2 Mathematical Setup of Neutron Transport

To derive the governing equation for the neutron population, one must first rigorously define the physical and mathematical space in which these particles exist. Let $V \subset \mathbb{R}^3$ be a physical three-dimensional volume defining the *reactor domain*, bounded by an

outer surface ∂V. Within this domain, a neutron's state is completely characterized by its spatial position vector $\boldsymbol{r} \in V$, its kinetic energy ϵ, and a unit vector $\hat{\boldsymbol{\Omega}}$ specifying its direction of flight. Together, these variables span the *phase space* of the system.

The starting point in describing free neutron content inside a rector's core is the *angular neutron density* $n\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)$, representing the expected number of neutrons per unit phase-space volume. However, in reactor physics, it is overwhelmingly common to work with the *angular flux* $\psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right) = v\left(\epsilon\right) n\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)$, where $v\left(\epsilon\right)$ is the neutron scalar speed. The angular flux represents the total track length traveled by all neutrons per unit time, per unit volume, per unit solid angle, and per unit energy.

The time evolution of the angular flux at a given phase-space point $\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right)$ is dictated by a strict *particle conservation principle*. Following the phenomenological approach outlined by Prinja and Larsen [9, Ch. 5, Sect. 2.3–2.5], one can express the time rate of change of the neutron density as a simple balance between production and removal mechanisms occurring within an infinitesimal phase-space element:

$$\frac{1}{v\left(\epsilon\right)} \frac{\partial \psi}{\partial t}\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right) = [\text{rate of gain}] - [\text{rate of loss}] \tag{1.19}$$

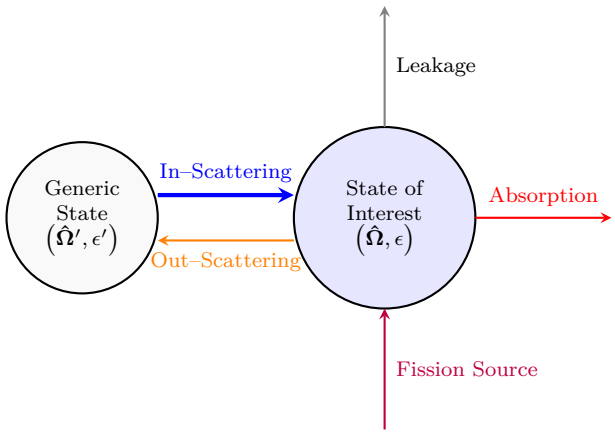

Figure 1.5. Compartimental model of Phase Space. For each point in physical space V one has two compartments, separating neutrons under study from all the others. The diagram illustrates the balance of the neutron transport equation: gains are represented by arrows point inward the blue circle, while losses are associated with arrows pointing outward.

To fully understand the derivation of the neutron scalar flux density equation, it is essential to distinguish between the *physical space description* and the *phase space description* of neutron dynamics. While physical space $V \subset \mathbb{R}^3$ represents the finite geometry of the reactor core, the neutron's phase space is a six-dimensional manifold

at any instant of time. Indeed, the state of a neutron is uniquely defined by three Cartesian coordinates for position $\boldsymbol{r}$, two angular coordinates for direction $\hat{\boldsymbol{\Omega}}$, and one scalar coordinate for energy ϵ. Since the neutron density balance is formulated in phase space, for each spatial point $\boldsymbol{r} \in V$, one can associate distinct *compartment* of the phase space. We define the *compartment under test* as the infinitesimal subspace describing neutrons with energy ϵ moving along direction $\hat{\boldsymbol{\Omega}}$. Conversely, all other possible states are denoted as $\left(\epsilon', \hat{\boldsymbol{\Omega}}'\right)$. Figure 1.5 illustrates the relationship between these compartments. This distinction clarifies why scattering processes are treated as *removal terms* for the compartment $\left(\epsilon, \hat{\boldsymbol{\Omega}}\right)$. Even though the neutron does not physically leave the reactor geometry, a collision changes its energy, direction, or both, effectively transferring it to a different phase space compartment $\left(\epsilon', \hat{\boldsymbol{\Omega}}'\right)$.

To construct the exact integro-differential transport equation, each of these macroscopic gain and loss mechanisms must be mathematically formalized as independent building blocks and that we shall do in the following.

Streaming (Spatial Transport) Neutrons travel in straight lines between collisions, maintaining their energy and direction constant while changing their physical position. The net rate at which neutrons flow out of the spatial volume element d^3r along direction $\hat{\boldsymbol{\Omega}}$ is mathematically described by the convective derivative. Using the divergence theorem, this spatial loss term evaluates to $\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla}\psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)$. It is crucial to note that this is the *only* term in the transport equation that couples different spatial coordinates; all other interaction terms (collision, scattering and fission) are spatially local, redistributing neutrons only within the energy–angle phase space at a fixed position $\boldsymbol{r}$. This spatial coupling is precisely what makes the transport equation computationally demanding to solve in highly heterogeneous geometries.

Collision (Loss) When a neutron interacts with a nucleus, it is effectively removed from its current phase-space coordinate $\left(\epsilon, \hat{\boldsymbol{\Omega}}\right)$. This removal includes both true physical absorption (where the neutron disappears) and scattering (where the neutron simply changes its energy or direction, thus leaving the specific state being considered). The total interaction rate per unit phase-space volume is directly proportional to the total macroscopic cross section, yielding the loss term $\Sigma_t\left(\boldsymbol{r}, \epsilon\right)\, \psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)$.

In-Scattering (Gain) Neutrons can enter the phase-space element $\left(\epsilon, \hat{\boldsymbol{\Omega}}\right)$ by scattering from different initial energies ϵ' and directions $\hat{\boldsymbol{\Omega}}'$. To quantify this gain, one must integrate over all possible initial states. The differential scattering cross section Σ_s encodes the probability of such transitions, leading to the in-scattering

rate:

$$\int_0^\infty \int_{4\pi} \Sigma_s \left(\boldsymbol{r}, \epsilon' \to \epsilon, \hat{\boldsymbol{\Omega}}' \to \hat{\boldsymbol{\Omega}}\right) \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon', t\right) \mathrm{d}\hat{\boldsymbol{\Omega}}' \, \mathrm{d}\epsilon' \tag{1.20}$$

Fission (Gain) In fissile materials, the birth of prompt neutrons from fission acts as a massive internal source. The total fission rate induced by neutrons at incident energy ϵ' is given by $\Sigma_f(\boldsymbol{r}, \epsilon')\,\psi$. Multiplying this by the average neutron yield $\nu(\boldsymbol{r}, \epsilon')$ and distributing the newly born neutrons into the energy spectrum ϵ via the probability density function $\chi(\boldsymbol{r}, \epsilon \mid \epsilon')$ gives the fission source rate. Assuming isotropic emission in the laboratory frame, this gain term takes the form:

$$\int_0^\infty \int_{4\pi} \chi(\boldsymbol{r}, \epsilon \mid \epsilon') \, \nu(\boldsymbol{r}, \epsilon') \, \Sigma_f(\boldsymbol{r}, \epsilon') \, \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon', t\right) \mathrm{d}\hat{\boldsymbol{\Omega}}' \, \mathrm{d}\epsilon' \tag{1.21}$$

External Source (Gain) Any independent source of neutrons that does not rely on the internal fission chain reaction is generically denoted by the gain term $S_{\text{ext}}\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)$. In a physical reactor context, this term accounts for intrinsic sources such as the spontaneous fission of heavy actinides and transuranic isotopes, e.g. ^{240}Pu or ^{252}Cf, as well as (α, n) reactions. These reactions occur naturally within the fuel or are deliberately introduced as specialized radioactive startup sources. While their contribution to the total neutron flux is mathematically negligible during full-power operation, they are fundamentally required for source–range monitoring and for safely initiating the reactor startup sequence [6]. For a more detailed treatment of the specific isotopic contributions to these intrinsic neutron backgrounds, the reader is referred to the comprehensive phenomenological descriptions provided in [9].

The Integro-Differential Transport Equation Imposing the particle conservation principle detailed in Eq. (1.19)—namely, that the time rate of change equals the sum of all source and in-scattering terms (gains) minus the streaming and collision terms (losses)—one definitively assembles the time-dependent Boltzmann integro-differential transport equation for neutrons:

$$\begin{aligned}
&\frac{1}{v(\epsilon)} \frac{\partial \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)}{\partial t} + \hat{\boldsymbol{\Omega}} \cdot \nabla \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right) + \Sigma_t(\boldsymbol{r}, \epsilon) \, \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right) = \\
&\qquad \int_0^\infty \int_{4\pi} \Sigma_s \left(\boldsymbol{r}, \epsilon' \to \epsilon, \hat{\boldsymbol{\Omega}}' \to \hat{\boldsymbol{\Omega}}\right) \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon', t\right) \mathrm{d}\hat{\boldsymbol{\Omega}}' \, \mathrm{d}\epsilon' \\
&+ \int_0^\infty \int_{4\pi} \chi(\boldsymbol{r}, \epsilon \mid \epsilon') \, \nu(\boldsymbol{r}, \epsilon') \, \Sigma_f(\boldsymbol{r}, \epsilon') \, \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon', t\right) \mathrm{d}\hat{\boldsymbol{\Omega}}' \, \mathrm{d}\epsilon' + S_{\text{ext}} \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, t\right)
\end{aligned} \tag{1.22}$$

This equation embodies the complete physical description of the reactor core and provides the exact mathematical foundation for all deterministic and stochastic computational methods in nuclear engineering.

Eq. (1.22) embodies the complete physical description of the reactor core. However, from a mathematical standpoint, it is a first–order integro–differential equation in both time and space. To guarantee the existence of a unique physical solution, it must be formulated as a well–posed Cauchy problem by specifying appropriate initial and boundary conditions.

Since the transport equation features a first-order time derivative, it first requires the specification of the neutron distribution at an initial time $t = 0$ across the entire phase space. If ψ_0 is a known function, the *initial condition* is simply given by:

$$\psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon, 0\right) = \psi_0\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) \quad \forall \boldsymbol{r} \in V \tag{1.23}$$

Furthermore, the convective derivative $\hat{\boldsymbol{\Omega}} \cdot \nabla\psi$ dictates the spatial coupling. To fully determine the solution, one must specify the behavior of the neutrons at the outer bounding surface ∂V of the reactor domain. Let $\boldsymbol{n}$ be the outward-pointing unit normal vector at a generic surface location $\boldsymbol{r}_s \in \partial V$. A neutron is geometrically defined as *entering* the volume if $\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{n} < 0$, and *exiting* if $\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{n} > 0$. The two most prevalent *boundary conditions* in reactor physics are:

- **Vacuum (Free Surface) Boundary:** This models a finite reactor placed in an empty environment. Any neutron leaking out of the boundary is permanently lost, and no neutrons can enter from the outside. Therefore, the incoming angular flux is strictly zero:

$$\psi\left(\boldsymbol{r}_s, \hat{\boldsymbol{\Omega}}, \epsilon, t\right) = 0 \quad \text{for } \hat{\boldsymbol{\Omega}} \cdot \boldsymbol{n} < 0 \tag{1.24}$$

- **Reflective Boundary:** This is crucial for modeling geometric symmetries, such as simulating only a single fuel assembly within an infinite lattice. Any neutron striking the boundary is specularly reflected back into the domain. If $\hat{\boldsymbol{\Omega}}'$ is the incident exiting direction, the reflected incoming direction $\hat{\boldsymbol{\Omega}}$ is given by $\hat{\boldsymbol{\Omega}} = \hat{\boldsymbol{\Omega}}' - 2\left(\hat{\boldsymbol{\Omega}}' \cdot \boldsymbol{n}\right)\boldsymbol{n}$, and the continuity condition reads:

$$\psi\left(\boldsymbol{r}_s, \hat{\boldsymbol{\Omega}}, \epsilon, t\right) = \psi\left(\boldsymbol{r}_s, \hat{\boldsymbol{\Omega}}', \epsilon, t\right) \quad \text{for } \hat{\boldsymbol{\Omega}} \cdot \boldsymbol{n} < 0 \tag{1.25}$$

1.2.3 Steady-State Formulation: Fixed Source and k-Eigenvalue Problems

For the vast majority of reactor core design and shielding analyses, the transient behavior is not the primary focus. By imposing a *steady-state* condition, the macroscopic neutron population is assumed to be in equilibrium, and the time derivative in the transport equation vanishes identically, i.e., $\partial\psi/\partial t = 0$.

With the temporal dependence removed, the time variable t drops out of the arguments, and the physical nature of the system is mathematically dictated by the presence and treatment of the source terms. Based on this, the steady-state Boltzmann equation branches into two fundamental classes of problems [9]: the Fixed Source problem and the k_{eff}-eigenvalue problem.

The Fixed Source Problem In applications such as radiation shielding, detector response analysis, or accelerator-driven subcritical systems, the neutron population is sustained by a known, independent external source $S_{\text{ext}}\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right)$. The physical system is typically subcritical, meaning that the internal fission chain reaction alone cannot sustain the neutron population indefinitely. In this configuration, the transport equation takes the form of a non-homogeneous linear equation:

$$\begin{aligned}\hat{\boldsymbol{\Omega}} \cdot \nabla\psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) + \Sigma_t\left(\boldsymbol{r}, \epsilon\right)\, \psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) = \\ \int_0^\infty \int_{4\pi} \Sigma_s\left(\boldsymbol{r}, \epsilon' \to \epsilon, \hat{\boldsymbol{\Omega}}' \to \hat{\boldsymbol{\Omega}}\right)\, \psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon'\right)\, \mathrm{d}\hat{\boldsymbol{\Omega}}'\, \mathrm{d}\epsilon' \\ + \int_0^\infty \int_{4\pi} \chi\left(\boldsymbol{r}, \epsilon \mid \epsilon'\right)\, \nu\left(\boldsymbol{r}, \epsilon'\right)\, \Sigma_f\left(\boldsymbol{r}, \epsilon'\right)\, \psi\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon'\right)\, \mathrm{d}\hat{\boldsymbol{\Omega}}'\, \mathrm{d}\epsilon' + S_{\text{ext}}\left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right)\end{aligned} \quad (1.26)$$

Because the external source distribution S_{ext} is explicitly provided, Eq. (1.26) admits a unique physical solution for the stationary flux ψ.

The k-Eigenvalue Problem Conversely, in the analysis of a critical nuclear reactor, the system is isolated from any external neutron sources ($S_{\text{ext}} = 0$). The neutron population must be entirely self-sustained by the internal fission chain reaction. Mathematically, setting the external source to zero into Eq.(1.22) yields a homogeneous equation. However, since the specified material cross sections and geometry of a real system rarely result in a perfectly critical balance, a stationary solution would generally not exist.

To force a mathematical balance, a fictitious scaling parameter — the effective multiplication factor k_{eff} — is introduced to artificially scale the neutron production from fission. This transforms the physical balance into a mathematical *eigenvalue*

problem:

$$\hat{\boldsymbol{\Omega}} \cdot \nabla \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) + \Sigma_t \left(\boldsymbol{r}, \epsilon\right) \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) = \int_0^\infty \int_{4\pi} \Sigma_s \left(\boldsymbol{r}, \epsilon' \to \epsilon, \hat{\boldsymbol{\Omega}}' \to \hat{\boldsymbol{\Omega}}\right) \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon'\right) \mathrm{d}\hat{\boldsymbol{\Omega}}' \, \mathrm{d}\epsilon' + \frac{1}{k_{\text{eff}}} \int_0^\infty \int_{4\pi} \chi \left(\boldsymbol{r}, \epsilon \mid \epsilon'\right) \nu \left(\boldsymbol{r}, \epsilon'\right) \Sigma_f \left(\boldsymbol{r}, \epsilon'\right) \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', \epsilon'\right) \mathrm{d}\hat{\boldsymbol{\Omega}}' \, \mathrm{d}\epsilon' . \quad (1.27)$$

The fundamental eigenvalue k_{eff} strictly dictates the criticality state of the system: if $k_{\text{eff}} = 1$, the physical system is exactly critical; if $k_{\text{eff}} > 1$, it is supercritical; and if $k_{\text{eff}} < 1$, it is subcritical.

1.3 The Limits of Analytical Methods and the Monte Carlo Approach

The steady-state Boltzmann transport equation derived in Section 1.2.3 provides an exact mathematical description of the neutron population. However, finding an exact, closed-form analytical solution for Eq. (1.26) in a realistic three-dimensional, multi-material reactor geometry with continuous energy dependence is mathematically impossible. To obtain analytical solutions, one must introduce severe approximations and solution to real-life problems can only be obtained numerically or using *stochastic simulations*.

1.3.1 The Diffusion Approximation and a 1D Model Problem

The most renowned simplification in reactor physics is the *Diffusion Approximation* [5], which systematically eliminates the angular dependence of the transport equation. This formulation relies on the P_1 approximation, assuming that the angular flux is only weakly anisotropic. Mathematically, ψ is expanded in a truncated series of spherical harmonics, retaining only the linear dependence on the direction vector $\hat{\boldsymbol{\Omega}}$:

$$\psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) \approx \frac{1}{4\pi} \phi \left(\boldsymbol{r}, \epsilon\right) + \frac{3}{4\pi} \hat{\boldsymbol{\Omega}} \cdot \boldsymbol{J} \left(\boldsymbol{r}, \epsilon\right) \quad (1.28)$$

In this expression, the fundamental macroscopic quantities emerge as the angular moments of the distribution. The zeroth angular moment is the *scalar flux* ϕ, strictly defined by integrating the angular flux over the entire 4π solid angle:

$$\phi \left(\boldsymbol{r}, \epsilon\right) = \int_{4\pi} \psi \left(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, \epsilon\right) \mathrm{d}\hat{\boldsymbol{\Omega}} \quad (1.29)$$

The first angular moment is the *neutron current density* vector $\boldsymbol{J}$, representing the net directional flow of particles:

$$\boldsymbol{J}(\boldsymbol{r},\epsilon)=\int_{4\pi}\hat{\boldsymbol{\Omega}}\,\psi\left(\boldsymbol{r},\hat{\boldsymbol{\Omega}},\epsilon\right)\,\mathrm{d}\hat{\boldsymbol{\Omega}} \tag{1.30}$$

By inserting this linear angular expansion back into the integro-differential Boltzmann equation and evaluating its zeroth and first angular moments, the complex angular coupling is analytically removed.

Derivation of the Diffusion Equation (P_1 Approximation). Consider the steady-state, monoenergetic transport equation with isotropic scattering and an isotropic external source $S(\boldsymbol{r})$.

Step 1: Zeroth Moment (Continuity Equation). Integrating the transport equation over the entire solid angle yields the neutron balance scalar equation. The streaming term integrates to $\nabla\cdot\boldsymbol{J}$, the collision term to $\Sigma_t\phi$, and the isotropic sources to $\Sigma_s\phi+S$. Rearranging and defining the macroscopic absorption cross section as $\Sigma_a=\Sigma_t-\Sigma_s$, we obtain the continuity equation:

$$\nabla\cdot\boldsymbol{J}(\boldsymbol{r})+\Sigma_a\phi(\boldsymbol{r})=S(\boldsymbol{r}) \tag{1.31}$$

Step 2: First Moment (Fick's Law). Multiplying the transport equation by the direction vector $\hat{\boldsymbol{\Omega}}$ and integrating over 4π extracts the first angular moment. The isotropic source terms vanish because the integral of an odd function over the unit sphere is identically zero. The streaming term is evaluated by substituting the P_1 expansion:

$$\int_{4\pi}\hat{\boldsymbol{\Omega}}\left(\hat{\boldsymbol{\Omega}}\cdot\nabla\left[\frac{1}{4\pi}\phi+\frac{3}{4\pi}\hat{\boldsymbol{\Omega}}\cdot\boldsymbol{J}\right]\right)\mathrm{d}\hat{\boldsymbol{\Omega}}=\frac{1}{3}\nabla\phi(\boldsymbol{r}) \tag{1.32}$$

where we utilized the angular integral identity

$$\int_{4\pi}\hat{\boldsymbol{\Omega}}\left(\hat{\boldsymbol{\Omega}}\cdot\boldsymbol{A}\right)\mathrm{d}\hat{\boldsymbol{\Omega}}=\frac{4\pi}{3}\boldsymbol{A} \tag{1.33}$$

Equating the non-vanishing terms naturally yields Fick's Law of diffusion, defining the diffusion coefficient $D=1/(3\Sigma_t)$:

$$\frac{1}{3}\nabla\phi(\boldsymbol{r})+\Sigma_t\boldsymbol{J}(\boldsymbol{r})=0\implies\boldsymbol{J}(\boldsymbol{r})=-D\nabla\phi(\boldsymbol{r}) \tag{1.34}$$

■

By formally enforcing Fick's Law (1.34), the system of equations is closed. The net neutron current is now strictly proportional to the negative gradient of the scalar flux. By substituting this relationship into the continuity equation (1.31), the complex integro-differential transport equation rigorously reduces to a canonical second-order linear partial differential equation. For a steady-state Fixed Source problem in a homogeneous medium, this governing *Diffusion Equation* reads:

$$-D\nabla^2\phi\left(\boldsymbol{r}\right)+\Sigma_a\phi\left(\boldsymbol{r}\right)=S\left(\boldsymbol{r}\right) \tag{1.35}$$

To demonstrate the analytical approach, consider a one-dimensional finite slab of thickness $2a$ (extending from $x=-a$ to $x=a$) placed in a vacuum. A planar, isotropic fixed source of strength S_0 neutrons per $\mathrm{cm}^2\cdot\mathrm{s}$ is located exactly at the mid-plane $x=0$. Exploiting the spatial symmetry, the domain can be restricted to $x\in[0,a]$. For $x>0$, the external source is zero, and Eq. (1.35) simplifies to:

$$\frac{d^2\phi\left(x\right)}{dx^2}-\frac{1}{L^2}\phi\left(x\right)=0 \qquad x>0 \tag{1.36}$$

where $L=\sqrt{D/\Sigma_a}$ is defined as the *diffusion length*.

This second-order ordinary differential equation requires two boundary conditions. The first is a source condition at the origin: by integrating Fick's law across the singular source plane, the current exiting the source must equal half the total source strength, yielding

$$-D\left.\frac{d\phi}{dx}\right|_{x=0}=\frac{S_0}{2} \tag{1.37}$$

The second is the vacuum boundary condition at the edge of the slab. Because the diffusion approximation inherently fails near physical boundaries (where the neutron angular distribution becomes highly anisotropic due to the lack of incoming particles), a correction derived from exact transport theory is applied. Solving the half-space transport problem (known as Milne's problem) reveals that the scalar flux linearly extrapolates to zero at a small distance outside the physical medium [5]. Therefore, rather than setting the flux to zero exactly at the physical edge $x=a$, the condition is imposed at an *extrapolated boundary* $a_e=a+d$. The extrapolation distance is given by $d\approx 0.71\lambda_{\mathrm{tr}}$, where λ_{tr} is the transport mean free path of the medium. Thus, the boundary condition is rigorously formulated as $\phi\left(a_e\right)=0$.

Applying these explicitly defined boundary conditions, the spatial profile of the scalar flux can be rigorously derived.

Analytical Solution of the 1D Slab Problem. For $x > 0$, the governing differential equation is a homogeneous second-order ordinary differential equation:

$$\frac{d^2\phi(x)}{dx^2} - \frac{1}{L^2}\phi(x) = 0 \tag{1.38}$$

The general solution can be expressed as a linear combination of hyperbolic functions. Anticipating the boundary condition at the extrapolated edge a_e, it is mathematically convenient to shift the argument and write the general solution in the form:

$$\phi(x) = A\sinh\left(\frac{a_e - x}{L}\right) + B\cosh\left(\frac{a_e - x}{L}\right) \tag{1.39}$$

where A and B are constants to be determined.

Step 1: Vacuum Boundary Condition. We apply the zero-flux condition at the extrapolated distance, $\phi(a_e) = 0$. Substituting $x = a_e$ yields:

$$\phi(a_e) = A\sinh(0) + B\cosh(0) \implies B = 0 \tag{1.40}$$

Since $B = 0$, the spatial profile reduces immediately to $\phi(x) = A\sinh\left(\frac{a_e - x}{L}\right)$.

Step 2: Source Condition. Next, we apply the source condition at the center of the slab. We first compute the spatial derivative of the reduced flux:

$$\frac{d\phi}{dx} = -\frac{A}{L}\cosh\left(\frac{a_e - x}{L}\right) \tag{1.41}$$

Evaluating this derivative at $x = 0$ and substituting it into the interface source condition, $-D\left.\frac{d\phi}{dx}\right|_{x=0} = \frac{S_0}{2}$, yields:

$$-D\left[-\frac{A}{L}\cosh\left(\frac{a_e}{L}\right)\right] = \frac{S_0}{2} \tag{1.42}$$

Isolating the remaining unknown constant A gives:

$$A = \frac{S_0 L}{2D\cosh\left(\frac{a_e}{L}\right)} \tag{1.43}$$

■

Substituting the evaluated constant A back into the spatial profile yields the

exact analytical solution for the scalar flux in the right half of the slab [6]:

$$\phi(x) = \frac{S_0 L}{2D} \frac{\sinh\left(\frac{a_e - x}{L}\right)}{\cosh\left(\frac{a_e}{L}\right)} \quad x \geq 0 \tag{1.44}$$

While the analytical solution derived for the 1D slab is mathematically elegant, it simultaneously exposes the severe limitations of deterministic approximations. To achieve such solutions, the true continuous-energy physics must be collapsed, and the geometry must be heavily homogenized. In reality, modern nuclear reactors are characterized by highly complex, three-dimensional heterogeneous structures and employ advanced coolant materials that strictly violate the fundamental assumptions of diffusion theory.

As the nuclear industry undergoes a paradigm shift toward the design of innovative, fast-spectrum architectures these old-fashioned deterministic approaches inevitably fail. This physical and engineering evolution dictates a radical transformation in computational reactor physics: abandoning global analytical solvers in favour of high-fidelity stochastic simulations based on the Monte Carlo method.

2 Industrial Context: Gen-IV Reactors & *new*cleo Company Paradigm

The deterministic transport methods derived in Ch. 1, primarily with the diffusion approximation computations, have historically provided the mathematical foundation for the design of the current fleet of commercial nuclear power plants. However, the future of nuclear engineering is shifting toward profoundly different physical regimes. To justify the absolute necessity of high-fidelity stochastic simulations it is essential to understand the physics governing these advanced architectures.

In this chapter, we shall introduce the Gen-IV reactors, their structure and working principles, highlighting the fundamental differences with old-fashioned Gen-II and Gen-III, PWR and Boiling Water Reactor (BWR). Specifically, we will examine how the transition to a fast neutron spectrum intrinsically resolves the severe operational problems caused by transient fission product poisoning, while simultaneously enabling a closed fuel cycle that drastically reduces the reactor size and long-term radiotoxicity of nuclear waste.

2.1 The Limits of Thermal Spectra: Fission Product Poisoning

For decades, the commercial nuclear industry has been dominated by thermal-spectrum architectures. While highly reliable for baseload power generation, these systems are fundamentally constrained by the neutronic properties of the thermal regime. As introduced in Ch. 1, the moderation of neutrons to thermal equilibrium maximizes the fission probability, but it simultaneously maximizes the parasitic absorption of specific fission products. The most restrictive of these phenomena is neutron poisoning, primarily driven by Xenon-135 and Samarium-149.

2.1.1 Transient Poisoning: Xenon-135 and the Reactor Dead Time

In a thermal spectrum, ^{135}Xe exhibits a huge absorption cross-section of approximately 2.6×10^6 b. During steady-state operation, an equilibrium is reached between its continuous production, via the β^--decay of Iodine-135, and its destruction through neutron absorption (burnout) or subsequent β^--decay. The full production and

depletion chain is the following

$$^{235}_{92}\mathrm{U} + ^{1}_{0}\mathrm{n} \rightarrow ^{135}_{52}\mathrm{Te} + ^{99}_{40}\mathrm{Zr} + 2^{1}_{0}\mathrm{n} \tag{2.1a}$$

$$^{135}_{52}\mathrm{Te} \xrightarrow[19\,\mathrm{s}]{\beta^-} ^{135}_{53}\mathrm{I} + e^- + \overline{\nu}_e \tag{2.1b}$$

$$^{135}_{53}\mathrm{I} \xrightarrow[6.57\,\mathrm{h}]{\beta^-} ^{135}_{54}\mathrm{Xe} + e^- + \overline{\nu}_e \tag{2.1c}$$

Once the Xenon-135 is produced by Iodine-135, two competing processes can happen inside the reactor's core, dictating the destiny of Xenon-135 itself: β^- decay, producing Cesium-135 and neutrun caputre, produing Xenon-136 through reactions

$$^{135}\mathrm{Xe} \rightarrow ^{135}\mathrm{Cs} + e^- + \overline{\nu}_e \tag{2.2a}$$

$$^{135}\mathrm{Xe} + n \rightarrow ^{136}\mathrm{Xe} + \gamma \tag{2.2b}$$

The true limitation of thermal reactors emerges during transient operations. The delicate balance between ^{135}I decay and ^{135}Xe burnout is highly sensitive to power variations. If the reactor undergoes a rapid shutdown or its power is significantly reduced, the thermal neutron flux drops nearly to zero, abruptly halting the Xenon burnout mechanism. However, the accumulated inventory of ^{135}I continues to decay into ^{135}Xe. As can be seen in Eqs. (2.1c) and (2.2a), due to the difference in the half-lives of Iodine-135 and Xenon-135, its production rate temporarily overtakes the natural decay rate. This mismatch causes a massive, temporary surge in the Xenon concentration, a phenomenon known as the *Xenon/Iodine Pit* (or *Xenon Dead Time*). This transient typically peaks around 11.6 h (generally between 10 and 12 h) after shutdown [3, Eq. (6.19), pp. 215-217] [6, Sect. 7.5], strongly suppressing core reactivity and preventing immediate reactor restart. A plot of this situation is depicted in Fig. 2.1.

To understand the severity of this transient, one must consider the physical limits of reactor control. In a steady-state chain reaction, operators balance neutron production and absorption primarily by inserting or extracting control rods. However, the massive accumulation of ^{135}Xe introduces an enormous parasitic neutron absorption, effectively acting as an overpowering, invisible set of control rods.

This phenomenon inserts a profound amount of *negative reactivity* – a measure of the system's deviation from criticality towards a subcritical state [3]. The neutron absorption from the Xenon peak is so immense that it completely exhausts the reactor's available positive reactivity margin. Even if the physical control rods are entirely withdrawn from the core, the neutron population cannot be sustained [5, 10]. Consequently, the reactor is physically paralyzed. Operators are forced to wait from

2 to 3 days for the ^{135}Xe to decay into Cesium-135, according to Eq. (2.2a), before criticality can be restored. This inherent inability to rapidly restart or modulate power severely cripples the load-following capabilities of thermal plants. Such neutronic inflexibility makes legacy reactors poorly suited to complement the highly variable and intermittent generation of modern renewable energy grids, a systemic bottleneck widely recognized in contemporary energy integration studies [11–13].

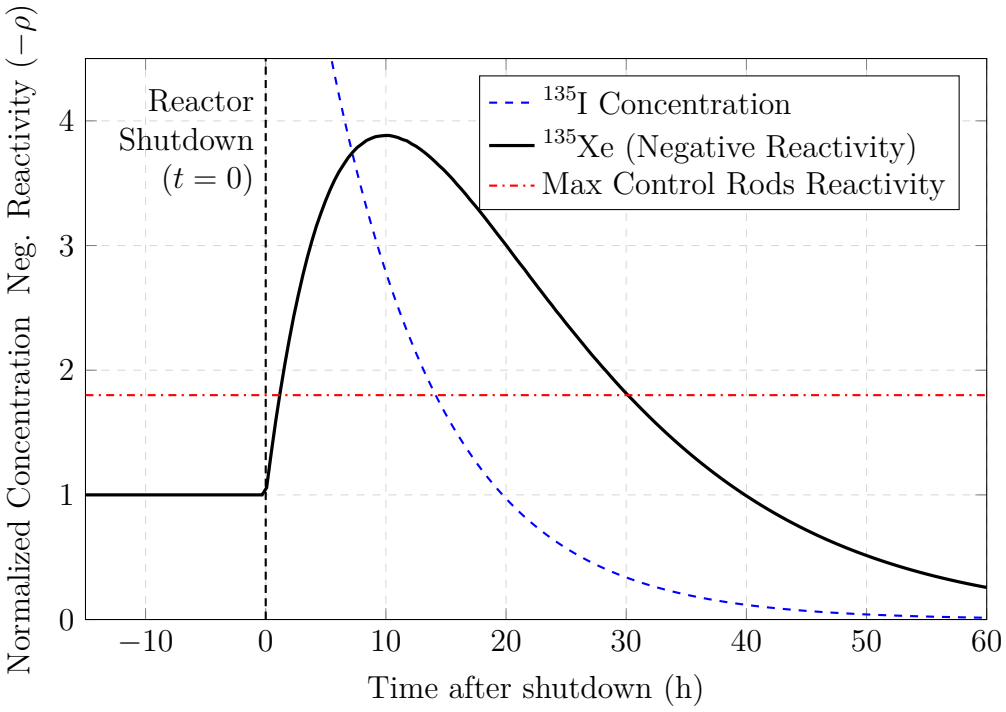

Figure 2.1. Analytical transient of ^{135}I and ^{135}Xe concentrations following a reactor shutdown. For $t < 0$, the system is in equilibrium (steady–state). At $t = 0$, the sudden shutdown halts Xenon burnout, causing a massive surge in negative reactivity ($-\rho$). Due to the decay dynamics, the ^{135}Xe concentration rapidly surpasses the maximum positive reactivity compensable by control rods (red dashed line).

2.1.2 Stable Poisoning: Samarium-149 and the Open Fuel Cycle

Alongside dynamic transients, thermal reactors suffer from the long-term accumulation of stable poisons. As mentioned in Par. 1.1, ^{149}Sm is produced from the decay

of Neodymium-149 through the following chain reactions

$$\text{Fission} \rightarrow {}^{149}_{60}\text{Nd} \tag{2.3a}$$

$${}^{149}_{60}\text{Nd} \xrightarrow[1.73\,\text{h}]{\beta^-} {}^{149}_{61}\text{Pm} + e^- + \overline{\nu}_e \tag{2.3b}$$

$${}^{149}_{61}\text{Pm} \xrightarrow[53.1\,\text{h}]{\beta^-} {}^{149}_{62}\text{Sm} + e^- + \overline{\nu}_e \tag{2.3c}$$

$${}^{149}_{62}\text{Sm} + n \rightarrow {}^{150}_{62}\text{Sm} + \gamma \tag{2.3d}$$

Once formed, ^{149}Sm possesses an enormous thermal cross-section of roughly 4.1×10^4 b for neutron absorption. In Eq. (2.3a), the generic Fission term accounts for the macroscopic production of Neodymium-149 from various fissioning nuclides, such as ^{235}U, ^{238}U and ^{239}Pu.

The behavior of ^{149}Sm differs fundamentally from ^{135}Xe due to its stability. While the *Xenon pit*, described in Sec. 2.1.1, eventually dissipates as the isotope decays into Cesium, the concentration of ^{149}Sm does not decrease after a reactor shutdown. On the contrary, the accumulated reservoir of Promethium-149 continues to decay, leading to a permanent increase in the Samarium concentration. This poison remains trapped in the core and can only be removed through neutron-induced burnout, as shown in Eq. (2.3d), during subsequent reactor operation.

Upon absorbing a thermal neutron, ^{149}Sm transmutes into the stable ^{150}Sm. Because this isotope possesses a significantly lower thermal cross-section, approximately 107 b, it is effectively transparent to the neutron flux. However, while the neutronic poisoning is mitigated, the resulting ^{150}Sm acts as an indestructible physical waste, permanently trapped within the solid fuel matrix.

Because the thermal spectrum cannot efficiently fission or utilize these accumulated *transuranic elements* and stable fission products, the fuel rapidly loses its reactivity. Consequently, conventional reactors are forced to operate on an *open fuel cycle*: once the parasitic absorption overcomes the fissile inventory, the fuel assemblies must be prematurely extracted and discarded as high-level radioactive waste, despite containing substantial amounts of unused fuel. To overcome these intrinsic physical bottlenecks, a radical paradigm shift away from moderated thermal spectra is strictly required.

2.2 The Paradigm Shift to Gen-IV Architectures

To circumvent the fundamental limitations of thermal-spectrum systems discussed in the previous section, the international nuclear engineering community is driving a

paradigm shift toward Gen–IV architectures. The defining characteristic of these advanced designs is the utilization of a *fast neutron spectrum* [14, 15]. Avoiding the use of light moderators such as water or graphite, Gen–IV reactors sustain the fission chain using high-energy neutrons, typically in the range from 0.1 MeV to 1 MeV.

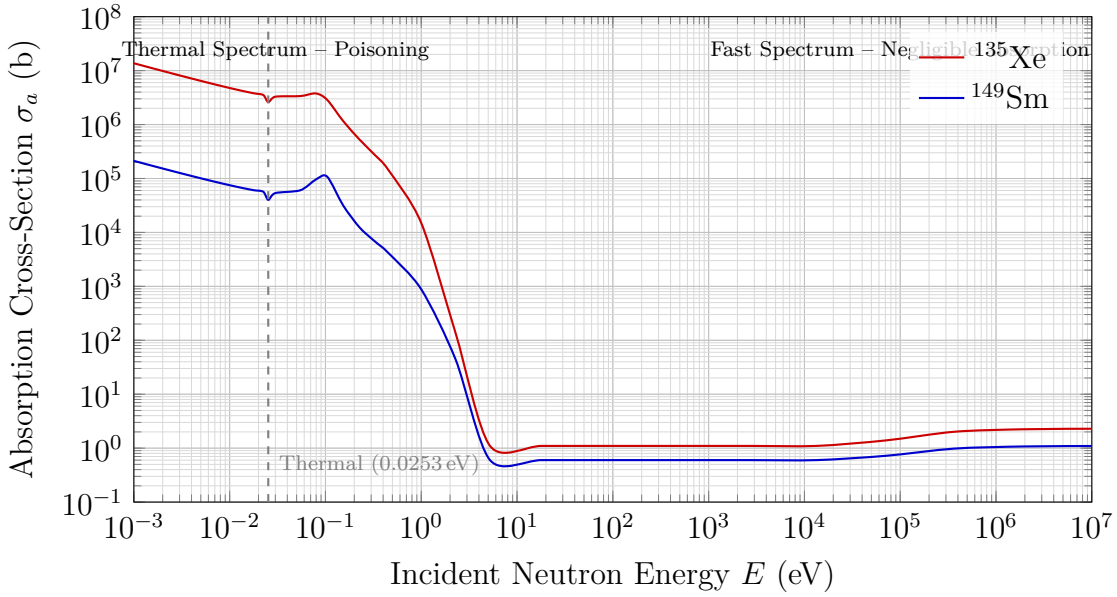

Figure 2.2. Macroscopic absorption cross–sections for ^{135}Xe and ^{149}Sm. The graphic highlights the value of approximately 2.6×10^6 b for Xenon–135 at thermal energy, underscoring the dramatic poisoning effect in thermal systems compared to the negligible absorption in the fast spectrum region.

This shift in the energy regime provides an immediate and elegant solution to fission product poisoning of Sec. 2.1.1 and 2.1.2. In a fast spectrum, the absorption cross-sections of isotopes like ^{135}Xe and ^{149}Sm drop by several orders of magnitude, as emphasized in Fig. 2.2; hence, for a fast neutron, these severe thermal poisons are essentially transparent. Consequently, fast reactors are intrinsically immune to the Xenon dead time problem, granting them *unrestricted power modulation* and immediate restart capabilities, which are crucial features for modern load-following operations.

Moreover, the superior neutron economy of the fast spectrum enables a revolutionary approach to fuel utilization: the *closed fuel cycle*. In conventional thermal reactors, ^{238}U is largely considered a fertile dead weight, as its fission cross-section is negligible for moderated neutrons. While the transmutation of ^{238}U into fissile ^{239}Pu occurs in both spectra, the fast reactor environment fundamentally redefines this process through two mechanisms. First, the high-energy neutron population (typically above 1 MeV) exceeds the fission threshold of ^{238}U, allowing it to act as an

active fuel component that contributes directly to the power density [15]. Second, the increased number of neutrons released per fission enables efficient *breeding*, as already mentioned in Eq. (1.8).

Unlike thermal systems, where Plutonium production is a byproduct that eventually leads to fuel poisoning, a Gen–IV LFR can be designed to produce more fissile material than it consumes.

This breeding capability fundamentally transforms ^{238}U from an inert byproduct into an active fuel source. By reprocessing the spent fuel and extracting the newly bred ^{239}Pu alongside the unburned actinides, Gen–IV reactors can effectively close the nuclear fuel cycle. This approach not only maximizes the energy extraction from the raw uranium but also drastically reduces both the volume and the radiotoxic lifetime of the ultimate high-level waste, shifting the waste management timeframe from hundreds of thousands of years to just a few centuries.

2.3 Physics of Lead-Cooled Fast Reactors (LFR)

Having established the necessity of a fast neutron spectrum, the engineering challenge shifts from neutron spectrum optimization to to the selection of optimal coolants, with low moderating power. Traditional water coolants are inherently unsuitable for fast reactors due to the high presence of hydrogen. Among the Gen–IV candidates, the LFR emerges as one of the most promising architectures, exploiting the unique thermodynamic and neutronic properties of liquid lead.

From a thermodynamic and safety perspective, liquid lead offers unparalleled advantages. With a remarkably high boiling point of approximately 1749 °C, an LFR can operate at atmospheric pressure without the risk of coolant boiling or core voiding [16, 17]. This completely eliminates the need for the massive, high-pressure containment vessels characteristic of conventional PWRs. Furthermore, lead's high thermal inertia and natural convection capabilities provide intrinsic passive safety, ensuring efficient decay heat removal even in total station blackout scenarios [16, 18].

However, it is the neutronic behavior of lead that fundamentally dictates the core physics and, consequently, the computational requirements for its simulation. Lead is a heavy nucleus with an extremely low neutron absorption cross-section and a negligible moderating ratio [19, 20]. When fast neutrons scatter off lead nuclei, they lose very little energy per collision [6], perfectly preserving the fast spectrum required for the breeding of ^{238}U and the transmutation of high-level waste.

This exact neutronic advantage introduces a formidable computational challenge. Because lead acts as a highly scattering but non-moderating medium, neutrons

exhibit extremely long mean free paths. This results in a *highly anisotropic neutron flux* and profound *deep penetration* phenomena within the reactor core and shielding structures.

As anticipated in Ch. 1, the diffusion approximation and traditional deterministic solvers rely on the assumption of a nearly isotropic flux and short mean free paths, making analytical mathematical models intrinsically inadequate for LFR geometries. To accurately evaluate criticality, local power peaking, and shielding efficacy in such a highly transparent fast medium, continuous-energy stochastic transport is unequivocally required. The necessity to track millions of deeply penetrating histories in complex three-dimensional domains justifies the transition from deterministic codes to modern, hardware-accelerated stochastic solvers.

2.4 The Small Modular Reactors Concept and Geometric Leakage

Alongside the transition to fast-spectrum physics and advanced coolants, the nuclear industry is undergoing a profound structural paradigm shift: the move from gigawatt-scale conventional plants to SMRs. While Generation III+ (Gen-III+) architectures like the European Pressurized Reactor (EPR) or Advanced Passive 1000 (AP-1000) range between 1000 and 1600 $\mathrm{MW_e}$, SMRs are defined by the International Atomic Energy Agency (IAEA) as advanced reactors with a power capacity of up to 300 $\mathrm{MW_e}$ per unit [21].

This modular approach represents a departure from the traditional *stick-built* construction method. In conventional large-scale projects, the installation site essentially becomes a decade-long, massive industrial plant, vulnerable to logistical complexities and weather-related delays. Conversely, the SMRs philosophy shifts the bulk of the manufacturing process to a controlled factory environment, where units are fabricated and transported as fully assembled modules [22]. The role of the final installation site is thus reduced to the assembly of standardized, pre-tested components. This transition not only ensures superior quality control but also addresses the immense capital costs and financial risks by shortening construction timelines and enabling a faster return on investment [22, 23].

Beyond economic benefits, the reduced thermal footprint of an SMRs core inherently facilitates the implementation of passive safety systems [23]. For instance, decay heat removal can often be achieved entirely through natural circulation and passive convection, drastically reducing the reliance on active, electrically driven primary pumps [24].

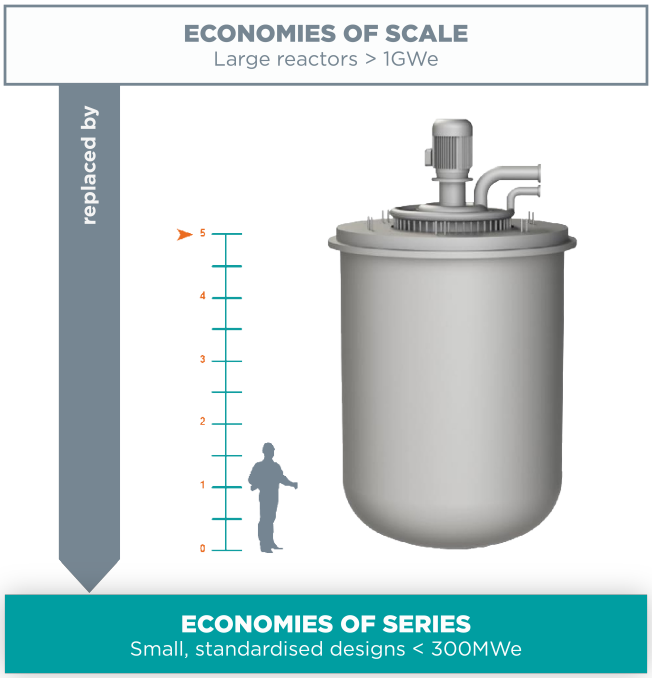

Figure 2.3. SMR size vs human size. Image taken from https://www.newcleo.com/our-technology/reactors/.

Despite their undeniable constructive and safety advantages, the development of fast-spectrum SMRs introduces severe neutronic complexities. This intrinsic competition between high-energy physics and small-scale geometry defines the primary engineering challenge for modern industrial projects, such as those proposed by *new*cleo Company, which aim to fuse these two opposed paradigms into a single Gen–IV architecture. Specifically, the physical compactness of an SMR inevitably features a much higher surface-to-volume ratio compared to traditional reactors. This geometric reality leads to a substantially increased probability of neutron leakage across the core boundaries. To maintain criticality and optimize fuel burnup despite this elevated leakage, the core geometry, reflectors, and shielding structures must be engineered with extreme heterogeneity and precision.

This geometric constraint creates a highly challenging environment for deterministic neutron transport solvers. The spatial heterogeneities in such compact arrangements, combined with the inherently long neutron mean free paths characteristic of fast-spectrum coolants like liquid lead, severely limit the applicability of legacy homogenization techniques [25, 26]. As discussed in Ch. 1, diffusion theory relies on the assumption of a nearly isotropic flux and gentle spatial gradients. In a fast SMR, diffusion theory simply cannot accurately capture the steep flux gradients, the pronounced anisotropic scattering, and the complex leakage spectra at the

core-reflector interfaces.

Consequently, high-fidelity, continuous-energy stochastic simulations become unavoidable for accurate SMR core characterization. The necessity to explicitly track deep-penetrating neutron histories across highly heterogeneous, three-dimensional compact geometries demands immense computational power. This enormous computational load currently represents one of the most critical bottlenecks in the design and licensing phases of modern Gen–IV modular systems, a challenge that the contemporary nuclear industry is actively striving to overcome.

2.5 Industrial Context and the *new*cleo Company Case Study

2.5.1 The Contemporary Advanced Nuclear Landscape

The transition toward Gen–IV technologies and modular architectures represents not only a scientific evolution but also a profound industrial shift. Historically, the nuclear sector has been dominated by massive, state-funded gigawatt-class projects. Today, however, the urgent need for deep decarbonization has catalyzed a *Nuclear Renaissance*, strongly validated by global climate agreements. Notably, the Intergovernmental Panel on Climate Change (IPCC) includes advanced nuclear power in its primary mitigation pathways [27], and the historic declaration at the COP28 climate summit saw over twenty nations commit to tripling global nuclear capacity by 2050 [28].

Within these modern energy frameworks, Gen–IV systems are viewed not in opposition to variable renewable energy sources, but as their necessary counterpart. Reports from the International Energy Agency (IEA) emphasize that SMRs can provide the essential, dispatchable low-carbon power required to stabilize grids dominated by intermittent solar and wind generation [29].

To deliver this resilient, zero-emission infrastructure, a new industrial landscape has emerged, driven by the influx of private capital and agile technology companies. These commercial efforts are heavily focused on deploying SMRs coupled with advanced coolants — such as liquid metals, molten salts, or high-temperature gases. Their common objective is to overcome the historical bottlenecks of legacy Gen-III+ plants: prohibitive capital costs, decade-long construction delays, and the accumulation of long-lived transuranic waste. By leveraging factory fabrication and fast-spectrum physics, the contemporary industry aims to provide scalable, inherently safe, and economically competitive clean energy solutions.

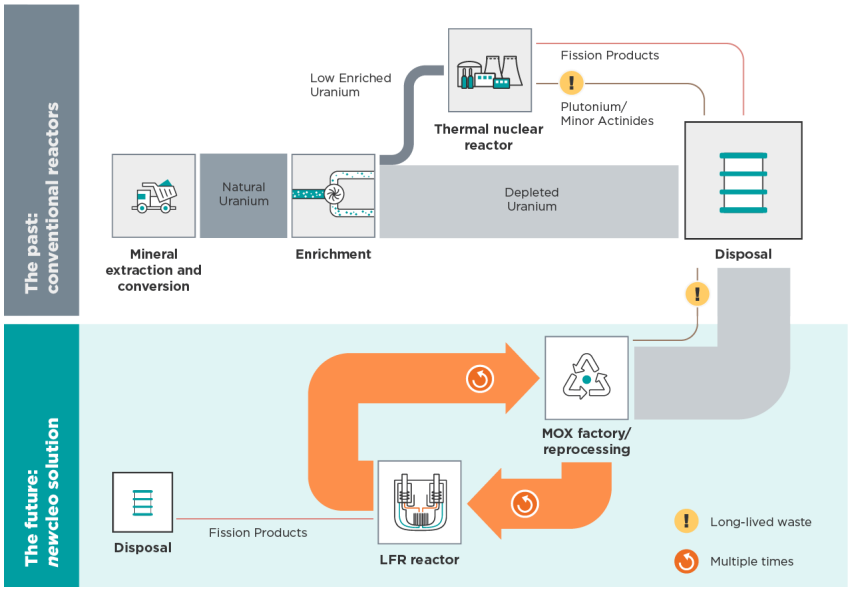

Figure 2.4. Extension and Re-Cycle of fuel material proposed by *new*cleo Company. Exhaust Mixed Oxide (MOX) can potentially be totally recycled, effectively exploiting 100% of fissile fuel loaded in the reactor. Image taken from https://www.newcleo.com/our-technology/fuel/

2.5.2 The *new*cleo Company Case Study: Merging LFRs and SMRs Paradigms

The unparalleled precision offered by continuous-energy stochastic codes is not merely a mathematical luxury; it has become a strict physical necessity for the design and licensing of these modern Gen–IV systems. A prominent industrial embodiment of the advanced paradigms discussed in this chapter is the technology currently being developed by *new*cleo Company, a clean nuclear technology company focused on the deployment of innovative LFRs [30].

Founded in 2021, the company represents a radical departure from traditional, state-led nuclear programs, operating instead on an accelerated, privately-funded schedule. *new*cleo Company aims to commercialize the LFR technology by converging it with an SMR layout, specifically targeting a 200 MW_e module known as the LFR-AS-200. The design operates at near-atmospheric pressure while entirely relying on a fast neutron spectrum, with energies typically peaking above 1 MeV [31, 32].

Beyond the reactor architecture itself, a fundamental pillar of the *new*cleo Company strategy is the implementation of a fully closed nuclear fuel cycle. Unlike legacy thermal systems, the fast neutron spectrum of the LFR enables the efficient utilization and multi-recycling of MOX fuel. *new*cleo Company proposal to extend fuel life is schematically depicted in Fig. 2.4

This fuel is entirely derived from the reprocessed spent nuclear fuel of existing thermal reactors, specifically utilizing depleted uranium and separated plutonium. By actively transmuting long-lived minor actinides, this *waste-to-fuel paradigm* definitively extracts residual energy while drastically reducing the volume, radiotoxicity, and decay time of the final nuclear waste [31]. To sustain this closed cycle and fuel its upcoming reactor fleet, the company is heavily investing in the upstream supply chain, planning the construction of a dedicated MOX manufacturing facility in France by 2030 [30].

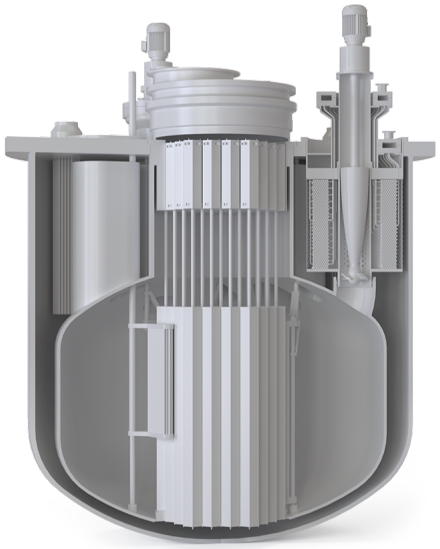

Figure 2.5. Rendering of the LFR-AS-30 precursor from *new*cleo Company. Image taken from https://www.newcleo.com/our-technology/reactors/

To contextualize the industrial urgency driving the need for accelerated computational methods, it is pertinent to outline the corporate roadmap. With the fuel manufacturing infrastructure established, the first physical demonstrator — a 30 MW_e precursor designated as the LFR-AS-30, rendered in Fig. 2.5 — is slated for nuclear criticality in the early 2030s. The operational and neutronic data gathered from this precursor will subsequently serve to license the First-Of-A-Kind commercial deployment of the full-scale LFR-AS-200 module by 2033.

2.5.3 The Computational Bottleneck of the LFR-SMR Convergence

While the advanced features of modern SMRs and LFRs offer unprecedented safety and sustainability, their convergence concurrently introduces profound computational bottlenecks. As established in the preceding sections, the compact core geometry,

coupled with the long neutron mean free paths and the highly anisotropic scattering properties of heavy liquid metals, fundamentally breaks the core assumptions of traditional deterministic solvers.

As a consequence, high-fidelity stochastic simulatins are strictly necessary to capture the exact neutronic behavior, the extreme spatial heterogeneities, and the deep shielding penetration within advanced designs like the *new*cleo Company core. However, solving these complex three-dimensional continuous-energy models requires an immense amount of computational effort.

Accelerating the fundamental algorithms underlying neutron transport – specifically targeting the massive hardware-level parallelism offered by FPGAs – represents a breakthrough strategy to drastically reduce simulation times. Bridging the gap between the complex nuclear physics required by the contemporary industry and high-performance hardware architecture constitutes the core engineering objective of this thesis, and necessitates a deep dive into the stochastic methodologies detailed in the following chapters.

3 Stochastic Approach to Neutron Transport: OpenMC Framework

The derivation of the analytical solutions presented in 1 is mathematically elegant, yet it simultaneously exposes the severe approximations required to solve the LTE in Eq. (1.22). Such approximations are fundamentally inadequate for describing the complex, three-dimensional heterogeneities of a real nuclear reactor, where stringent safety margins and advanced core design, outlined in Ch. 2, demand high-fidelity neutron flux mapping.

To overcome the inherent computational limitations of traditional deterministic solvers — which struggle to scale efficiently across intricate geometries — this chapter introduces the Monte Carlo method for particle transport. The discussion focuses on the statistical foundations of the technique, specifically its sampling mechanics and convergence properties, demonstrating how macroscopic cross-sections are mathematically translated into discrete interaction probabilities. Finally, the open-source software OpenMC is presented, outlining the core features of its particle tracking loop that will subsequently serve as the architectural blueprint for the hardware acceleration developed in this work.

3.1 Foundations of Monte Carlo: Overcoming the Curse of Dimensionality

As established in the previous chapter, analytical solutions to the neutron transport equation (cfr. Eq. (1.22)) are limited to highly idealized scenarios. In practical reactor physics, solving the integro-differential equation for real three-dimensional cores traditionally relies on deterministic numerical methods, such as *Discrete Ordinates* or *Spherical Harmonics*. A comprehensive theoretical treatment of these numerical techniques is beyond the scope of this work and the reader is referred to the foundational literature, including texts by Duderstadt and Hamilton [5, Ch. 13], Lewis and Miller [33, Ch. 4], and Bell and Glasstone [34, Ch. 5], for detailed mathematical derivations.

The fundamental limitation of these deterministic approaches is their reliance on phase-space discretization. The transport equation operates in a seven-dimensional phase space (three spatial, two angular, one energetic, and one temporal variables). If each dimension is discretized into n intervals, the theoretical memory and processing time scale exponentially as $O\left(n^7\right)$. This phenomenon, originally formalized by Bellman and widely known in computational mathematics as the *curse of dimensionality* [35, 36], forces deterministic solvers to adopt severe geometric homogenization and coarse multi-group energy approximations to remain computationally tractable.

To bypass this mathematical bottleneck without sacrificing geometric or energetic

fidelity, the stochastic Monte Carlo method abandons the Eulerian mesh-based perspective in favor of a Lagrangian approach: rather than calculating macroscopic flux distributions across a rigid grid, the algorithm tracks the exact continuous trajectories and interaction sequences of millions of individual discrete neutrons.

3.1.1 Stochastic Estimators and Convergence in High-Dimensional Spaces

In the Monte Carlo framework, macroscopic physical quantities of interest — such as the local neutron flux, specific reaction rates, or the k_{eff} eigenvalue — are not evaluated as deterministic continuous functions. Instead, they are estimated as the expectation value of a random variable. Let x be a random variable representing the physical contribution, or *tally*, recorded during the random walk of a single particle.

If N independent particle histories are simulated, yielding a set of independent and identically distributed observations $x_1, x_2, \ldots, x_N$, the true expectation value $E[x]$ of the desired physical quantity is mathematically approximated by the empirical sample mean, namely $\langle x \rangle$.

The reliability of this estimation is governed by the Central Limit Theorem (CLT). It dictates that, for a sufficiently large number of simulated histories N, the probability distribution of the sample mean $\langle x \rangle$ asymptotically approaches a normal distribution centred around $E[x]$, irrespective of the underlying probability density function of the individual random variable x [37].

To quantify the statistical uncertainty of the simulation, let the true variance of the underlying probability distribution be denoted as $V[x]$. By the fundamental properties of variance for independent variables, the variance of the sample mean is given by $V[\langle x \rangle] = V[x]/N$, making the true standard error of the sample mean exactly $\sqrt{V[x]/N}$. Because determining the exact analytical value of $V[x]$ requires solving the complete phase-space transport equation [38], it is computationally prohibitive for complex 3D geometries. Therefore, in practical Monte Carlo simulations, the theoretical variance $V[x]$ must be approximated by the unbiased sample variance s^2. According to the Law of Large Numbers [37], as the number of histories increases, the sample variance converges to the true variance ($s^2 \to V[x]$ for $N \to \infty$). Consequently, following the standard derivation for Monte Carlo estimators [39], the estimated standard error of the mean, denoted here as Δ, is rigorously computed by the code as:

$$\Delta = \frac{s}{\sqrt{N}} \tag{3.1}$$

This fundamental relationship, $\Delta \propto 1/\sqrt{N}$, demonstrates that the statistical

error diminishes purely as a function of the number of simulated histories, i.e. its computational complexity scales as $O\left(N^{-1/2}\right)$. Because this convergence rate is completely decoupled from the geometric complexity or the dimensionality of the phase space, stochastic methods are uniquely equipped to overcome the curse of dimensionality, providing highly accurate expectation values even in the most intricate 3D reactor models [38].

Driven by this unique mathematical property, this thesis focuses exclusively on the stochastic approach. However, a critical engineering trade-off emerges: while Monte Carlo methods elegantly bypass the exponential memory scaling of deterministic solvers, simulating the millions of independent particle histories required to achieve an acceptable error Δ introduces a massive computational burden of its own. Addressing and accelerating this specific stochastic workload — shifting from traditional sequential execution to highly parallel hardware architectures — is the central objective of the subsequent chapters.

3.2 From Cross Sections to Stochastic Processes: the inverse Sampling Method

As established in Ch. 1, the physical likelihood of a neutron undergoing a specific interaction is strictly governed by its cross sections. For instance, given that a collision has occurred, the relative probability p_r of a specific reaction channel r is given by the ratio of its cross section to the total cross section, namely

$$\wp_r = \frac{\sigma_r}{\sigma_{\text{tot}}} \tag{3.2}$$

While deterministic solvers utilize these physical ratios as continuous macroscopic coefficients to evaluate steady-state reaction rates within the integro-differential transport equation [33, 34], the Monte Carlo algorithm fundamentally reinterprets them: in a stochastic computational framework, the set of physical probabilities $\{p_r\}$ rigorously constitutes a *Discrete Probability Density Function (PDF)* that governs the history of the particle. The primary algorithmic challenge is thus resolving how a deterministic processor can generate individual interaction events that statistically obey this underlying PDF.

Because a deterministic computer cannot inherently generate true randomness, this stochastic behavior is achieved through the generation of Pseudo-Random Numbers (PRNs), uniformly distributed in the interval $[0, 1[$. Let a single PRN be denoted as ξ. To map the uniform distribution of ξ onto the physical PDFs of the

neutron — whether evaluating discrete reaction types or continuous variables — the algorithm relies on the *Inverse Transform Sampling* method [39].

The most critical application of this method in reactor physics is determining the distance d a neutron travels in a straight line before undergoing a collision. In a medium with a total macroscopic cross-section Σ_{tot}, the probability that a neutron interacts within a distance dx is governed by the exponential attenuation law $\mathscr{E}(x)\,dx = \Sigma_{\text{tot}} \exp(-\Sigma_{\text{tot}} x)\,dx$. To materialize this flight path, the code must map the uniformly distributed PRN ξ onto this distribution via the Cumulative Density Function (CDF)

$$\mathscr{E}(x) = \int_0^x \Sigma_{\text{tot}} \exp(-\Sigma_{\text{tot}} u)\,du = 1 - \exp(-\Sigma_{\text{tot}} x) \tag{3.3}$$

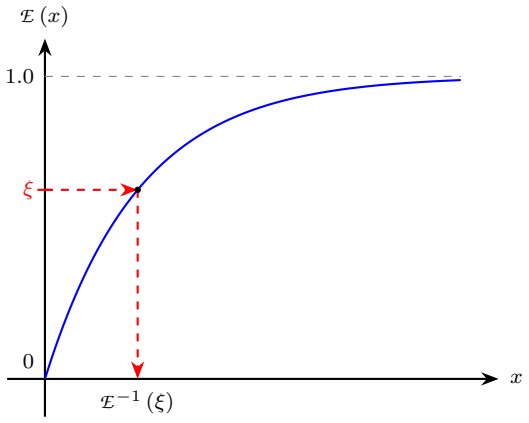

Figure 3.1. Visual representation of the Inverse Transform Sampling method. A PRN $\xi \in [0, 1[$ is mapped through the inverse CDF $\mathscr{E}^{-1}$ to yield the physical variable x.

Solving for the flight distance d yields

$$d_{\text{surv}} = -\frac{\ln(1-\xi)}{\Sigma_{\text{tot}}} \tag{3.4}$$

However, from a coding standpoint it is far convenient to use the relation

$$d_{\text{coll}} = -\frac{\ln \xi}{\Sigma_{\text{tot}}} \tag{3.5}$$

This optimization stems from the fact that interaction and survival are statistically complementary events. While $1 - \exp(-\Sigma_{\text{tot}} x)$ represents the CDF of interaction, the term $\exp(-\Sigma_{\text{tot}} x)$ describes the probability of survival (non-interaction) up to distance x. Since the PRN ξ is sampled from a uniform distribution $\mathscr{U}(x; 0, 1)$, mapping it directly to the survival probability is statistically equivalent to mapping it

to the interaction CDF. This refactoring eliminates a redundant subtraction operation for every flight path calculation, demonstrating how foundational nuclear physics is streamlined for high-performance stochastic execution.

3.3 State of Art in Neutron Transport: OpenMC

To evaluate the feasibility of hardware-based acceleration, this work references OpenMC, a modern, open-source Monte Carlo particle transport code. OpenMC is designed to provide high-fidelity simulations using continuous-energy nuclear data and Constructive Solid Geometry (CSG). As a state-of-the-art tool for High Performance Computing (HPC) clusters, it arguably represents the upper performance bound of what can be achieved on traditional CPUs [40]. However, its execution model serves as an ideal case study to highlight a fundamental structural misalignment: the clash between stochastic logic — which is inherently unpredictable — and general-purpose architectures, which totally rely on predictability and regular data access.

3.3.1 History-Based Workflow and Nested Loop Structure

Following the deterministic initialization phase — during which complex CSG inputs and continuous-energy cross-sections are parsed from HDF5 libraries into the host memory [40] — OpenMC enters its active stochastic simulation phase. The macroscopic execution flow is governed by a rigid, deeply nested loop hierarchy designed to simulate millions of independent histories while ensuring statistical convergence and enforcing strict population control mechanisms across successive neutron generations [41].

In standard Monte Carlo simulations, regardless of whether they are executed in fixed-source or eigenvalue mode, the overarching control structure of the solver is based on a rigid, nested loop hierarchy [42]. The execution flow fundamentally iterates over macroscopic *batches*, and individual *particle histories*. In the specific case of criticality calculations, batches are further explicitly subdivided into successive fission *generations* to ensure the stationary convergence of the source [41].

- **The Batch Loop:** The outermost loop manages the statistical collection of data. Tallies are accumulated and averaged only at the end of each batch to ensure the statistical independence of the results.
- **The Generation Loop:** Nested within each batch, this loop iterates over consecutive generations (or cycles). In each generation, a fixed population of

N neutrons is simulated, and the resulting fission sites are collected to form the spatial source distribution for the subsequent generation.

Inside this macroscopic double-`for` loop lies the *particle loops*. For each generation, OpenMC dispatches the execution to the available CPU threads, deepening the control structure into a third level: a `for` loop iterating over the N particles of the current population. Once a thread takes ownership of a neutron, the logic is handed over to the `transport_history_based` function, which executes the fourth and innermost layer of the architecture: a random walk encoded in the `while(p.alive())` loop.

Conceptually, the entire simulation engine is built upon a 4-deep nested hierarchy. The macroscopic orchestration, managing the outer double `for` loop, can be summarized by the following modern C++ pseudocode

```
for (auto& batch : batches) {
  for (auto& gen : gens) {
    transport_history_based();
  }
}
```

Listing 3.1. The two outermost hierarchical levels of the history-based Monte Carlo method in `OpenMC`. This snippet is strictly illustrative and the detailed source code can be found at https://github.com/openmc-dev/openmc.

Inside this macroscopic structure, for each generation, OpenMC dispatches the execution to the available CPU threads. Once a thread takes ownership of a neutron population, the control logic is handed over to the `transport_history_based` function.

This function acts as the micro-kernel of the OpenMC calculations. It deepens the control structure into the remaining two levels: a third `for` loop iterating over the N assigned particles, and the fourth, innermost layer of the architecture, representing the actual Monte Carlo engine via the random walk `while(p.alive())` loop:

```
// Inside the transport_history_based function...
for (auto& p : particles) {
  while (p.alive()) {
    // interaction roulette and random walk here...
  }
}
```

Listing 3.2. The micro-kernel of the simulation, representing the inner two levels of the hierarchy where the core stochastic transport is performed.

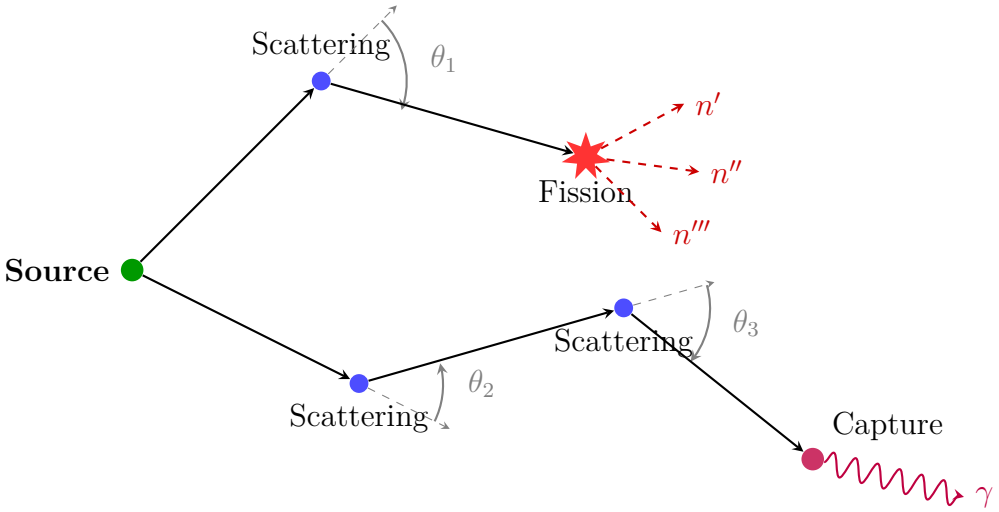

Figure 3.2. 2D schematic representation of the Monte Carlo random walk illustrating stochastic divergence in both outcome and path length. Path 1 terminates after 2 steps with a fission event, whereas Path 2 requires 3 steps before undergoing radiative capture. This variability in the number of loop iterations per particle inherently breaks the efficiency of lock-step execution models (like GPUs).

Computational Complexity and the CPU Bottleneck

From an algorithmic perspective, highlighting this 4-level nesting is crucial to understanding why the history-based approach is intrinsically slow on traditional architectures. The overall computational complexity of the simulation scales as

$$\mathcal{O}\left(N_{\text{batches}} \cdot N_{\text{generations}} \cdot N_{\text{particles}} \cdot N_{\text{steps}}\right) \tag{3.6}$$

While the number of interaction steps per particle is stochastic, the multiplicative nature of this nested structure means that the innermost physical kernels are executed an enormous number of times. Even with multi-threading applied to the `for` particle loop, each thread remains trapped in its own unpredictable `while(p.alive())` loop.

3.3.2 Handling Interactions: The Collision Kernel

When the random walk logic determines that a neutron has collided within a material (rather than crossing a geometric boundary), the control flow of `transport_history_based` is transferred to the collision kernel. This phase is computationally dense and represents the core of the physical simulation, encompassing macroscopic data retrieval, discrete sampling, and kinematic updates.

The sequence of operations executed during a collision typically unfolds as follows

1. **Macroscopic Data Retrieval:** The software must perform memory lookups to retrieve the microscopic cross-sections for all the specific isotopes composing the target material. Since the simulation tracks neutrons in a continuous-energy domain, these values cannot be fetched directly. Instead, the CPU must retrieve the nearest bounding energy grid points from the nuclear data libraries and perform real-time floating-point linear interpolations. Finally, the software calculates the total macroscopic cross-sections on the fly by multiplying the interpolated values by their respective atomic densities, actively applying the fundamental equations introduced in Ch. 1.

2. **The Reaction Roulette:** OpenMC must determine the specific type of interaction (e.g., elastic scattering, inelastic scattering, radiative capture, or fission). This is achieved through a discrete sampling process: a new PRN $\xi \in [0, 1[$ is generated and compared against the cumulative distribution function formed by the partial macroscopic cross-sections. The amplitude of each intervals is the probability $\wp_r$, according to Eq. (3.2) for the corresponding physical interaction.
$$\Xi_{r+1} - \Xi_r = \frac{\Sigma_r}{\Sigma_{\text{tot}}} \equiv \wp_r \tag{3.7}$$
The selected interaction is the one corresponding to the interval, fission in this case.

3. **Post-collision Kinematics:** Depending on the selected reaction, the neutron's phase-space state is updated. For a scattering event, new energy and directional cosines are calculated, often requiring the evaluation of complex, computationally expensive elementary functions to sample specific angular distribution laws [43].

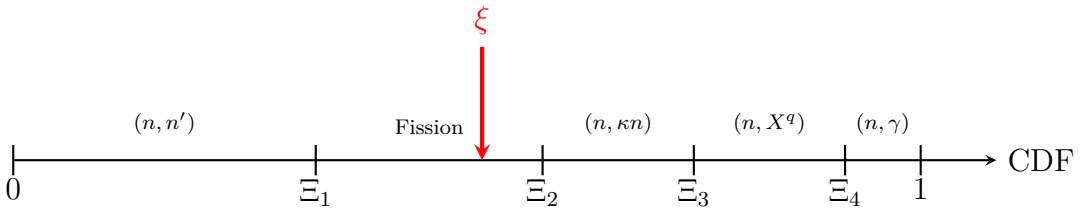

Figure 3.3. Interaction Roulette in one dimension. The interaction type is determined by mapping a PRN ξ onto the cumulative distribution function partitioned by the macroscopic partial cross–sections.

3.3.3 The Branching Bottleneck

From an architectural perspective, the reaction roulette is the most disruptive segment of the history-based workflow. The selection of the interaction channel results in a deeply nested tree of `if-else` statements or `switch-case` structures.

On a standard CPU, this stochastic divergence is devastating for performance. Modern processors rely heavily on *branch prediction mechanisms* to maintain their instruction pipelines full. However, because the outcome of the reaction roulette is driven by a uniformly distributed PRN, the control flow is inherently unpredictable. Consequently, the CPU suffers from frequent branch mispredictions, leading to costly pipeline flushes and stalled clock cycles. Furthermore, the necessity to evaluate complex mathematical functions on demand exacerbates the execution latency.

This heavy reliance on unpredictable branching and complex, dynamically invoked mathematical functions is precisely what makes the collision kernel an ideal candidate for custom logic implementation.

3.3.4 The Architectural Bottleneck: CPU and GPU Limitations

Despite the mathematical elegance of the Monte Carlo method and the maturity of codes like OpenMC, the history-based workflow faces significant performance barriers on standard computing platforms. Crucially, these limitations do not stem from the physical properties of the semiconductor — the underlying silicon in a modern CPU is fundamentally the same as that in an FPGA. Instead, the bottleneck is entirely *architectural*.

It arises from a fundamental mismatch between the highly stochastic, divergent nature of particle transport and the rigid, instruction-driven Fetch-Decode-Execute (FDE) paradigm of general-purpose processors.

CPU Performance Constraints

Traditional Multiple Instruction Multiple Data (MIMD) architectures, such as modern multi-core CPUs, are designed to excel at executing sequential, predictable instruction streams, as shown in Fig. 3.4. In the context of the Monte Carlo nested loop structure, CPUs face severe bottlenecks

- **Instruction Overhead:** General-purpose processors must constantly fetch, decode, and execute instructions from memory for every single algorithmic step. For a deeply nested, stochastic transport kernel, this overhead consumes a massive portion of the execution time compared to the actual arithmetic logic being performed.

Figure 3.4. Schematic overview of CPU working mechanism and cache miss in data retrival.

- **Branch Misprediction:** The Monte Carlo random walk is inherently driven by pseudo-random numbers. At every collision step, the reaction roulette forces the CPU to evaluate unpredictable conditional branches. This destroys the efficiency of the CPU's speculative execution and branch predictor, leading to constant pipeline flushes and wasted clock cycles [43].
- **The Memory Wall:** Because each neutron in a batch possesses a unique energy and spatial position, threads independently access non-contiguous memory addresses to fetch macroscopic cross-sections. This irregular access pattern completely defeats the CPU's spatial locality assumptions and hardware prefetchers, leading to a continuous stream of *cache misses*, shown in Fig. 3.4. Consequently, the processor is forced to stall its execution pipeline — often for hundreds of clock cycles — while waiting for high-latency Dynamic RAM (DRAM) fetches [44].

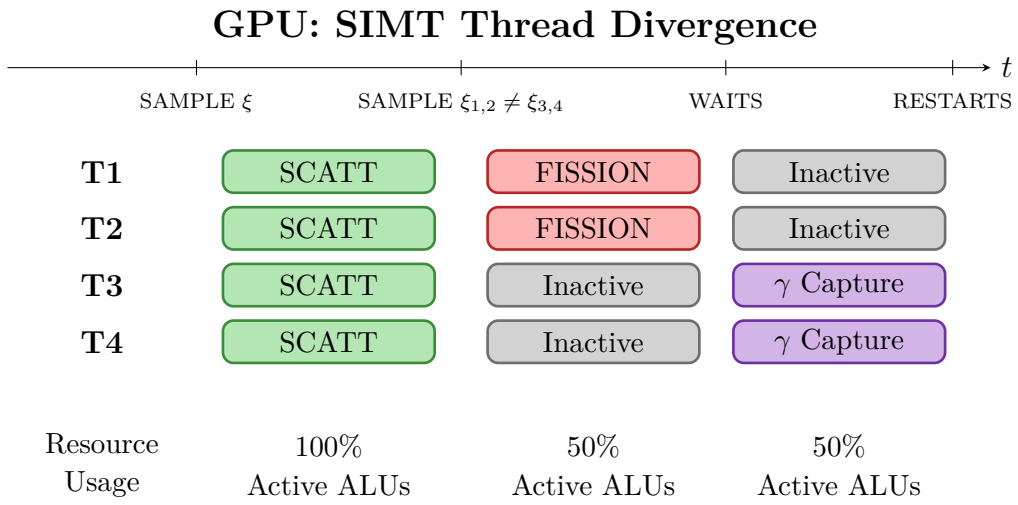

Figure 3.5. Pictorial representazion of thread divergence. If data inside the same wavefront experience different stochastic paths, GPU's ALUs must wait one thread to finish processing its own data, leading to an inefficient resource usage.

GPU Acceleration and the SIMT Paradigm

To combat the sheer volume of particle histories and the high computational complexity, the nuclear engineering community has heavily invested in porting Monte Carlo codes to GPUs. They attempt to solve the performance bottleneck through massive parallelism, leveraging the Single Instruction Multiple Threads (SIMT) paradigm.

In a SIMT architecture, thousands of computational threads are executed concurrently by grouping them into execution batches (often referred to as *warps* or

wavefronts). The fundamental constraint of this paradigm is that all threads within the same group must execute the exact same instruction simultaneously, operating in a strict lock-step fashion.

While SIMT is exceptionally efficient for predictable, heavily structured workloads, it encounters a critical failure point when applied to stochastic neutron transport: *thread divergence*. Because the Monte Carlo random walk is driven by probabilities, particles processed in the same warp will inevitably experience different physical events. For instance, if one neutron in a warp undergoes a simple elastic scattering (requiring trigonometric state updates) while another triggers a complex fission event (requiring the generation and tagging of secondary particles), their execution paths are said to be divergent, meaning they have to perform different tasks.

When this divergence occurs, the GPU hardware cannot execute both branches simultaneously across the warp. Instead, it is forced to serialize the divergent paths, as diepicted in Fig. 3.5. The processor executes the e.g. γ capture instructions while *masking* the thread that needs to undergo fission, and vice versa. This forced serialization leaves a massive portion of the GPU's Arithmetic Logic Units (ALUs) completely idle, drastically degrading the theoretical peak throughput [45].

This phenomenon clearly demonstrates that merely increasing the raw density of compute cores, as GPUs do, fails to resolve the intrinsic hardware-software misalignment of stochastic branching. It proves that the *curse of computational complexity* introduced by Monte Carlo can not be brute-forced by traditional instruction-driven architectures.

4 Hardware Acceleration: Commercial FPGA Architectures

In recent years, the development of digital technologies has driven a massive increment in the complexity of scientific and industrial applications, demanding the processing of enormous streams of data [46]. Fields such as HEP or Monte Carlo particle transport require sophisticated algorithms capable of manipulating independent sets of data with strict constraints on latency, timing closure, and energetic efficiency [47, 48], as pointed out in Chs 1 and 3, respectively.

To meet these stringent real-time requirements, traditional instruction-driven processors often fall short due to their inherent architectural bottlenecks. In light of these considerations, this chapter introduces the FPGA as the optimal electronic device to perform highly parallelized stochastic operations [49].

4.1 Historical Evolution of Programmable Logic

4.1.1 From Programmable Array Logic to Complex Programmable Logic Devices

The concept of *programmable device* began acquiring significance in the 1970s as a response to the rigid nature of factory-programmed masked Read-Only Memorys (ROMs), which required a complete re-manufacturing of the mask to alter their logic [50, 51]. The first step towards flexibility was marked by the advent of Erasable Programmable ROMs (EPROMs) and Electronically Erasable programmable ROMs (EEPROMs), which introduced ultraviolet or electrical erasing mechanisms to reset the memory state [51, 52], respectively.

In parallel with non-volatile memory development, programmable logic underwent a rapid evolution. Early devices such as Programmable Array Logic (PAL) and Programmable Logic Array (PLA) were One-Time Programmable (OTP) chips capable of implementing combinatorial functions in Sum of Products (SoP) form by physically burning internal fuses [50, 51]. By the early 1980s, these were largely superseded by Complex Programmable Logic Devices (CPLDs) [51]. CPLDs offered a reprogrammable set of interconnected macrocells, combining logical `AND/OR` matrices with early Flip–Flops (FFs) integration to support basic sequential logic [50].

However, the true paradigm shift occurred in 1985 when Xilinx Inc. released the XC2064, the world's first commercially viable FPGA [53]. The FPGA architecture completely abandoned the traditional `AND/OR` matrices in favor of Look-Up Tables (LUTs) [54]. Coupled with a massive integration of FFs and flexible routing resources for clock signals, FPGAs offered unprecedented scalability [54].

The theoretical and practical power of this new architecture was strikingly

demonstrated at the end of the 1990s by researchers such as Adrian Thompson, who leveraged the intrinsic physical properties of the FPGA silicon to evolve a voice recognition circuit using genetic algorithms [55]. Since then, increasing transistor density has allowed modern FPGAs to embed millions of LUTs alongside dedicated hardware resources such as Block RAM (BRAM), Digital Signal Processors (DSPs) and many others [56].

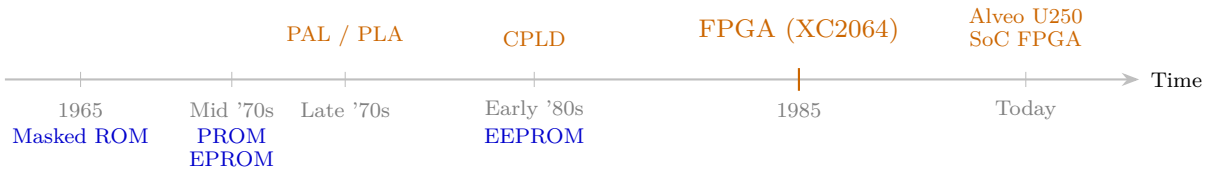

Figure 4.1. Timeline of programmable devices evolution: from fixed logic to high–performance FPGAs.

4.1.2 The Architectural Trade-Off: ASIC Performance vs. General-Purpose Flexibility

To fully appreciate the value of FPGAs in modern HPC, they must be positioned within the broader architectural landscape. Traditional CPUs and GPUs execute code *temporally*. Even when executing parallel threads, they rely on complex memory interfaces, caching mechanisms, and lock-step execution to manage data [57]. This instruction-driven approach introduces variable latencies and massive execution overhead.

On the opposite end of the spectrum lie Application Specific Integrated Circuits (ASICs). While they offer the absolute maximum in terms of performance and energetic efficiency, their logic is hardwired during manufacturing, totally lacking the flexibility required for evolving scientific research.

FPGAs perfectly bridge this gap by offering a *spatial computing* paradigm [58]. By mapping the code directly onto the chip's fabric, FPGAs create a true parallel instantiation of logic circuits. The hardware programming process involves designing a custom, application-specific logic circuit — comprising both combinatorial and sequential elements — that continuously processes a stream of data rather than interpreting sequential instructions [59].

Because the parallelism is realized physically across the available space of the die (utilizing thousands of logic gates, wires, and memory cells simultaneously), the resulting architecture achieves the strict timing predictability of an ASIC [57]. Once the physical routing is established, the latency of the computation becomes fixed

and deterministic, ensuring an exceptionally high and stable rate of data processing without sacrificing the ability to reprogram the device [56, 58].

The selection of an FPGA for accelerating stochastic neutron transport is thus dictated by the current evolutionary stage of nuclear simulation codes. While ASICs represent the absolute ceiling in terms of computational throughput and energetic efficiency, their immutable silicon logic requires the underlying algorithm to be entirely frozen.

Currently, Monte Carlo simulations for advanced nuclear systems, such as SMRs, are far from a frozen state. Physical collision models, variance reduction techniques, and nuclear cross-section libraries undergo continuous research and refinement. Should the stochastic neutron transport algorithms ever reach a definitive, non-plus-ultra mathematical formulation, a dedicated Monte Carlo ASIC would undoubtedly replace current hardware. However, at present, the FPGA represents the optimal technological compromise: it grants the deterministic latency and spatial parallelism typical of an ASIC, while strictly preserving the reconfigurability required to update the hardware architecture in tandem with theoretical advancements in reactor physics.

4.2 Anatomy of a Modern FPGA

Having established the theoretical advantages of the spatial computing paradigm, it is necessary to examine the physical building blocks that make this architecture possible. Unlike a CPU, which is built around rigid ALUs and hierarchical cache memories, a modern FPGA is a highly heterogeneous silicon canvas. It consists of a bi-dimensional array of Configurable Logic Block (CLB) interspersed with specialized hardware primitives and interconnected by a programmable routing matrix.

The evolution of these devices is heavily dictated by semiconductor manufacturing parameters, primarily the *technology node*, measured in nanometers. The transition from early planar transistors, e.g. the 220 nm nodes of the late 1990s, to modern 3D FinFET technologies, such as 16 nm and 7 nm, has drastically reduced power consumption and switching times. This extreme miniaturization allows contemporary FPGAs to embed billions of transistors, translating into millions of available logic gates and operating frequencies in the order of hundreds of megahertz, thus blurring the line between general-purpose FPGAs and dedicated System on Chips (SoCs).

4.2.1 Core Resources: LUTs, Flip-Flops, DSPs, and BRAMs

The fundamental unit of computation inside an FPGA is the CLB. Each CLB contains the primary digital elements required to build any arbitrary sequential or

combinatorial circuit: LUTs, FFs, and dedicated routing logic.

- **Look-Up Tables:** Instead of computing boolean logic through fixed `AND/OR` gates, an FPGA uses LUTs. A LUTs is essentially a small, high-speed Static RAM (SRAM) memory that stores a predefined truth table. When input signals are applied to the LUT, it simply *look up* and outputs the corresponding pre-calculated boolean result. This eliminates the need for instruction decoding: the logic function is intrinsically embedded in the memory state.

- **Flip-Flops:** To perform sequential operations and maintain the state of the system across clock cycles, LUTs are strictly paired with memory registers, specifically DFFs. In the context of the stochastic neutron transport developed in this work, FFs are the backbone of the *spatial pipeline*: they hold the neutron state variables, e.g. id, position, direction, energy and all the others constant, during a clock cycle while the downstream logic processes them.

- **Carry Chains:** Embedded within the same logic blocks as LUTs and FFs are dedicated, high-speed routing paths known as carry chains. While LUTs compute the base logic, carry chains are explicitly designed to propagate the carry bit of arithmetic operations (such as addition or subtraction) between adjacent cells with near-zero delay, bypassing the slower general-purpose interconnect. As will be detailed in Ch 5, this hard-wired infrastructure is the critical enabler for the high-frequency, 64-bit fixed-point additions required to update physical distances and coordinate translations natively in hardware.

While LUTs are extremely versatile for control logic and simple arithmetic, implementing complex mathematical operations — such as the multiplications required for energy cross-section interpolation — purely in LUTs would consume a massive amount of area and severely degrade clock frequency. For this reason, modern architectures embed hardwired computational blocks:

- **Digital Signal Processor:** These are dedicated, high-speed arithmetic blocks distributed across the chip. A standard DSP slice contains a dedicated multiplier and an accumulator. By offloading heavy math to DSPs, the FPGA can execute complex stochastic sampling formulas in a single clock cycle with maximum power efficiency.

Finally, to resolve the critical *Memory Wall* bottleneck described in Ch. 3, FPGAs distribute memory directly alongside the computational logic, avoiding the latency of external Random Access Memory (RAM) access

- **BRAMs and Ultra RAMs (URAMs):** instead of a rigid cache hierarchy, the FPGA provides thousands of independent, dual-port memory blocks physically scattered across the die. These can be configured and wired directly to the specific logic circuit that needs them, guaranteeing a deterministic, single-cycle read latency for critical data, such as the macroscopic cross-sections of the reactor materials.

Figure 4.2. Xilinx Alveo-U250 FPGA board. Image taken from Xilinx website https://www.amd.com/en/products/accelerators/alveo/u250/a-u250-a64g-pq-g.html

4.2.2 The Target Platform: Xilinx Alveo U250

To sustain the massive parallelism required for simulating SMR physics, the hardware platform must provide an exceptional balance of logic resources and on-chip memory bandwidth. For this project, the **Xilinx Alveo U250** Data Center Accelerator Card, shown in Fig. 4.2, was selected as the target device.

Based on the 16 nm FinFET UltraScale+ architecture, the Alveo U250 is an enterprise-grade board designed to tackle the most demanding HPC and machine learning workloads. From a bird's-eye view, the board interfaces with the host server via a high-throughput PCIe Gen3 x16 connection, allowing rapid offloading of the initial nuclear cross-section libraries from the host CPU to the accelerator.

However, the true strength of this platform lies in its immense on-chip resource density, making it an ideal candidate for defeating the memory wall effect in Monte Carlo transport. The device features over 1.7 million LUT and more than 12000 DSP slices, allowing the instantiation of deeply pipelined, parallel reaction evaluators. Crucially, the Alveo U250 provides a massive amount of distributed memory — over

54 Mbit of standard BRAM and an astounding 360 Mbit of URAM. This vast on-chip storage capacity is fundamental for caching large portions of the neutron cross-section data directly adjacent to the computational logic, ensuring zero-wait-state access during the stochastic random walk. The key hardware specifications of the target device are summarized in Table 4.1.

Resource Type	Available Quantity
Architecture	16 nm UltraScale+
System Logic Cells	1341000
Look-Up Tables (LUTs)	1728000
Flip-Flops (FFs)	3456000
DSP Slices	12288
Block RAM (BRAM)	54 Mbit
UltraRAM (URAM)	360 Mbit
Peak Int8 Performance	33.3 TOPs

Table 4.1. Hardware specifications of the target FPGA device, the Xilinx Alveo U250.

4.3 From Instruction Fetch to Dataflow Programming

The transition from a software-based Monte Carlo code (such as OpenMC) to a hardware-accelerated engine requires a radical paradigm shift in how the algorithm is conceptualized. Standard software development is intrinsically tied to the Von Neumann computing model. In this architecture, a Program Counter governs the execution flow: instructions are iteratively fetched from memory, decoded, and executed by a centralized ALU.

While highly flexible, this *instruction-driven* approach is fundamentally inefficient for stochastic neutron transport. As discussed in Chapter 3, Monte Carlo simulations are heavily reliant on conditional branching (e.g., determining the type of nuclear reaction or tracking the neutron across complex geometric boundaries). In a CPU, unpredictable branches cause continuous pipeline flushes and branch prediction penalties, while the need to continuously fetch cross-section data results in severe memory stall cycles.

FPGAs overcome these limitations by replacing the instruction-driven execution with a *dataflow programming paradigm*. In a dataflow architecture, there is no Program Counter and no instruction fetch phase. Instead, the Monte Carlo algorithm is *unrolled in space*. The operations required to process a neutron are *physically*

instantiated as a continuous chain of dedicated hardware modules (implemented via the LUTs, DSPs, and BRAMs, as described in Sect. 4.2).

This spatial realization enables *deep pipelining*: rather than processing a single neutron through all its physical stages sequentially, the hardware pipeline allows multiple independent neutrons to be processed simultaneously at different stages of the datapath. Once the pipeline is filled, the hardware can theoretically complete the evaluation of one physical event per clock cycle, regardless of the mathematical complexity of the underlying equations.

Consequently, programming the Alveo U250 does not involve writing a sequence of commands, but rather designing a fluid network of data paths. The state variables of the neutron become a continuous stream of digital signals flowing through the reconfigurable fabric. This dataflow approach definitively breaks the memory wall and bypasses the Von Neumann bottleneck, laying the architectural foundation for the custom hardware accelerator developed and detailed in the following chapters.

4.3.1 The Computing Device Landscape: Where FPGAs Stand

To fully appreciate the dataflow paradigm, it is instructive to map the FPGA within the broader landscape of modern computing devices. Processors and accelerators are generally classified according to a trade-off between programming flexibility, computational performance, and energetic efficiency.

- **CPUs** Operating under a pure temporal paradigm, CPUs offer the absolute maximum in programming freedom. They can execute any arbitrary code, from operative systems to complex control logic. However, their reliance on instruction fetching and complex cache hierarchies results in the lowest performance-per-watt ratio when deployed for highly parallel mathematical tasks.

- **GPUs:** GPUs extend the temporal paradigm into the SIMT domain. They feature thousands of small ALU that execute the same instruction on different data simultaneously. While they offer massive throughput for structured data, their performance degrades severely when faced with the highly divergent, unpredictable branching typical of Monte Carlo algorithms. Furthermore, their power consumption is typically the highest among computing devices.

- **CPLDs:** Representing the lower end of spatial computing, CPLDs offer instant-on reprogrammability and fixed, highly predictable timing. However, their resources are limited to simple macrocells and EEPROM-based routing, making

them suitable only for glue-logic and basic control tasks, lacking the DSPs and memory required for scientific computation.

- **FPGAs:** sit at the sweet spot for scientific hardware acceleration. They offer the massive spatial parallelism and low-power determinism of custom hardware, combined with a vast array of resources (millions of LUTs and thousands of DSPs on modern devices). Crucially, their SRAM based routing allows full reprogrammability, accommodating the evolving mathematical models of nuclear reactor physics.

- **ASICs:** represent the ultimate extreme of spatial computing. Since the algorithm is permanently etched into silicon during manufacturing, they provide the absolute maximum in operating frequency, throughput, and power efficiency. However, this comes at the cost of zero reprogrammability: an ASIC designed for one specific Monte Carlo codebase cannot be updated if a new cross-section evaluation method is discovered.

4.3.2 Register-Transfer Level (RTL) Programming

Unlike high-level software languages such as C++ and Python that abstract the underlying hardware architecture, programming an FPGA for maximum efficiency requires operating at the Register-Transfer Logic (RTL) level. RTL is a *design abstraction* that models a synchronous digital circuit in terms of the flow of digital signals between hardware registers, and the logical operations performed on those signals.

In RTL design, described using Hardware Description Languages (HDLs) such as VHDL, the fundamental elements are explicitly managed:

- **Registers (FFs):** used to store the state of the system synchronously with the clock signal.

- **Combinational Logic (LUTs and DSPs):** Positioned between registers to perform the actual mathematical evaluations.

By writing RTL code, the designer is not writing instructions for a compiler to execute, but rather *synthesizing* a physical electronic circuit. Every variable or signal assignment translates to a physical wire, and every conditional statement translates to a physical multiplexer. This low-level control is strictly necessary for stochastic neutron transport, as it allows the designer to manually enforce deterministic, single-cycle read accesses to data.

4.3.3 Spatial Pipelining VS Instruction Fetch

The ultimate goal of RTL design is the realization of a deeply pipelined dataflow architecture, which definitively overcomes the limitations of standard processors. As previously established, the Von Neumann architecture forces unpredictable branches to cause continuous pipeline flushes, while the need to continuously fetch instructions results in memory stall cycles.

The *spatial pipeline* unrolls the algorithm spatially on chip. Rather than processing a single neutron through all its physical stages sequentially, the spatial pipeline allows multiple independent neutrons to be processed simultaneously at different stages of the datapath. For instance, while Neutron n_1 is having its scattering angle evaluated in a DSP slice, neutron n_2 is simultaneously fetching its specific cross-section from the URAM in the previous stage, and neutron n_3 is undergoing a distance-to-boundary intersection check.

Once the spatial pipeline is completely filled, the FPGA can theoretically complete the evaluation of one complex physical event per clock cycle, achieving an *initiation interval* of 1. The state variables of the neutron become a continuous stream of digital signals flowing through the reconfigurable fabric, completely bypassing the Von Neumann bottleneck and delivering massive, deterministic throughput.

5 Towards Hardware-Accelerated Monte Carlo: An FPGA Implementation

As established in the preceding chapters, the standard Monte Carlo method provides unparalleled physical fidelity for neutron transport, yet it suffers from severe computational bottlenecks when executed on traditional von Neumann processors. This chapter presents the design and development of a custom hardware accelerator targeting the most computationally intensive core of this workload: the stochastic tracking micro-kernel. While the fundamental reactor physics and the consolidated Monte Carlo algorithmic steps remain strictly unaltered, the execution paradigm undergoes a radical transformation. By migrating the simulation from a temporal, instruction-based software routine into a deeply pipelined, spatial dataflow architecture on an FPGA, this work seeks to demonstrate the feasibility of overcoming inherent CPUs limitations through dedicated hardware acceleration.

In the following, each block, or module, will breifly analyzed, highlighting the difference between software programming and hardware description approaches.

To provide a clear overview of this engineering effort, the chapter is structured following a bottom-up methodology. The first part introduces the fundamental hardware-oriented mathematical primitives required by the simulation. Rather than detailing standard digital design practices, the discussion broadly outlines the custom implementation of pipelined arithmetic units and analytical function evaluators, which are strictly necessary to sustain a continuous spatial dataflow. Subsequently, the focus shifts to the hardware-specific adaptations for the stochastic engine, notably the integration of a high-throughput Pseudo-Random Number Generator (PRNG). Finally, the chapter illustrates how these foundational logic modules are orchestrated into a fully integrated spatial pipeline, specifically designed to efficiently resolve the unpredictable divergence of the neutron random walk.

5.1 Overview of Mathematical Building Blocks

The realization of a high-throughput spatial Monte Carlo engine requires a fundamental departure from software-centric arithmetic. To sustain a continuous stream of data, the mathematical primitives governing the simulation must be entirely redesigned to guarantee deterministic execution without stalling the pipeline. This section details the synthesis of these fundamental hardware blocks.

5.1.1 Fixed-Point Precision and Resource Management

Standard CPU-based Monte Carlo codes, such as OpenMC, rely heavily on IEEE 754 double-precision floating-point arithmetic to handle the vast dynamic range of physical variables. However, as extensively documented in hardware design

literature, instantiating deeply pipelined floating-point units on an FPGA consumes a disproportionate amount of LUTs and introduces severe routing congestion [49, 60]. To meet the stringent resource constraints and high-frequency timing requirements of the FPGA fabric, this architecture adopts a custom 64-bit fixed-point representation.

Specifically, the 64-bit data word is partitioned into a 16-bit integer component, including the sign bit, and a 48-bit fractional component, collectively denoted as $Q16.48$. Table 5.1 summarizes the critical trade-offs between this custom fixed-point format and the standard IEEE 754 double-precision floating-point format.

As highlighted in Table 5.1, while floating-point arithmetic offers a virtually infinite dynamic range, the $Q16.48$ bit-width allocation provides a superior architectural balance for this specific physical application. The 48 fractional bits guarantee the extreme, constant numerical resolution of 10^{-15} required to evaluate continuous microscopic cross-sections and resolve stochastic probabilities near zero. Simultaneously, the 15 integer bits plus sign provide sufficient dynamic range to represent macroscopic geometric flight distances needed in this project, expressed in units of millimetres, within the reactor domain without triggering spatial overflow conditions, and neutron energies scaled in MeV units.

A critical hardware advantage of this choice is sign management. Unlike floating-point formats, which require dedicated, complex, and high-latency logic layers to evaluate the sign bit and perform mantissa alignment before basic addition or subtraction [49, 61], two's complement arithmetic natively manages the sign of physical variables. By treating these variables effectively as large integers, the synthesis tool can map operations directly and efficiently onto the dedicated DSP slices embedded within the FPGA, maximizing computational density [60].

Furthermore, the fixed-point paradigm offers an extreme optimization when handling physical or mathematical constants. In a software environment, a constant occupies memory and requires memory-fetch cycles. At the RTL level, a 64-bit fixed-point constant consumes absolutely zero active logic resources: it is synthesized merely as a 64-bit bus with its wires physically tied to the logic high or logic low voltage levels, i.e. direct connection to the power supply voltage or to ground.

A key observation for the subsequent architectural choices is that basic operations such as *addition* and *bit-shifting*, representing multiplication or division by powers of two depending on exponent' sign, are *native hardware operations*, hence extremely efficient. While a CPU must execute these through its ALU instruction set, an FPGA performs bit-shifts through simple wire routing, costing zero logic, and additions via high-speed dedicated carry chains.

From a dataflow perspective, shifting to fixed-point integer arithmetic allows

Table 5.1. Comparison of arithmetic representation formats for the 64-bit hardware Monte Carlo tracking engine.

Parameter	Custom Fixed-Point (Q16.48)	IEEE 754 Double-Precision
Total Width	64 bits	64 bits
Allocation	1 Sign, 15 Integer, 48 Fractional	1 Sign, 11 Exponent, 52 Mantissa
Dynamic Range	$\approx \pm 3.27 \times 10^4$	$\approx \pm 1.79 \times 10^{308}$
Resolution	$2^{-48} \approx 3.55 \times 10^{-15}$	Variable (depends on exponent)
Sign Management	Two's Complement (Native logic)	Sign-Magnitude (Complex logic)
Hardware Footprint	Low (Direct DSP mapping)	High (Requires massive LUT usage)

the fundamental mathematical modules to be deeply pipelined. The implemented adders and multipliers achieve an Initiation Interval (II) of exactly 1 clock cycle, guaranteeing maximum data throughput. Furthermore, the operational latency is deterministic and strictly limited to the minimal number of pipeline stages required to traverse the internal DSP registers, ensuring that the continuous stream of neutron data is never stalled by basic arithmetic evaluations.

5.1.2 Hardware Division: Sequential Radix-2 Architecture

While multiplication and division by factors of two are trivially resolved via zero-cost wire shifting, spatial transport geometry inherently requires arbitrary division operations. Specifically, calculating the exact distance to a geometric boundary during the ray-tracing phase or distance to next collision necessitate dividing the particle's relative spatial distance by its directional vector component.

Implementing a fully combinational 64-bit divider on an FPGA would induce massive routing delays, severely degrading the maximum clock frequency. To preserve system timing without consuming excessive DSP slices, this architecture implements a sequential Radix-2 restoring divider. Operating iteratively, this module evaluates one bit of the quotient per clock cycle. For the 64-bit Q16.48 format, the division introduces a strict, deterministic latency of 64 clock cycles. While this represents a multi-cycle mathematical block, its predictable latency allows it to be seamlessly integrated into the high-level module dataflow without causing asynchronous pipeline stalls.

5.1.3 Analytical Functions: Unrolled CORDIC Architecture

The stochastic random walk requires the continuous evaluation of complex transcendental functions. Specifically, the tracking algorithm must compute trigonometric pairs and square roots to resolve the neutron's post-collision angular scattering direction, and the natural logarithm to sample the flight distance to the next interaction, according to Eq. (3.5).

In traditional software environments [62], these functions are evaluated using floating-point polynomial expansions, e.g., Taylor or Chebyshev series, implemented via Horner's method [63, 64]. However, mapping these polynomials directly onto a spatial hardware architecture presents severe disadvantages. Achieving the required high precision would necessitate high-degree polynomials, which directly translate to deeply cascaded hardware multipliers. This approach rapidly exhausts the limited dedicated DSP slices on the DSP, complicates routing, and introduces variable

execution times due to necessary range-reduction algorithms.

To overcome the DSP multiplier bottleneck, the COordinate Rotation Digital Computer (CORDIC) algorithm was selected as the foundational architecture for all analytical evaluations. The fundamental hardware advantage of CORDIC is the complete elimination of complex multiplications during its iterative phase. Instead, the algorithm computes transcendental functions relying exclusively on primitive bit-shifts and additions. These operations map exceptionally well to the basic logic cells of the FPGA fabric, ensuring a highly efficient area utilization.

Rather than computing convergence terms on the fly, the CORDIC datapath utilizes small, hardcoded LUTs containing precomputed elementary micro-rotation angles. Depending on the target function, the tables store either circular angles for trigonometric evaluations, or hyperbolic angles, defined respectively as:

$$\alpha_k^{(C)} = \tan^{-1}\left(2^{-k}\right) \tag{5.1a}$$

$$\alpha_k^{(Y)} = \tanh^{-1}\left(2^{-k}\right) \tag{5.1b}$$

Because these constants are resolved at synthesis time, they are mapped as highly optimized distributed logic, entirely bypassing the latencies associated with external memory reads.

It is important to note that the standard CORDIC algorithm inherently suffers from a strictly limited domain of convergence in both its circular and hyperbolic operating modes [65]. However, the mathematical nature of this limitation differs fundamentally between the two modes, dictating distinct hardware solutions.

In both cases convergence is guaranteed in the range $I = [-\theta_{\max}, +\theta_{\max}]$, but the range extension is achieved differently in the two scenarios. Because the scattering kinematics require resolving angles over the full $[-\pi, \pi]$ angular spectrum, it is strictly necessary to manage the sign of the input arguments: by exploiting the inherent periodicity and symmetry of trigonometric functions, the architecture implements a lightweight pre-processing quadrant mapping stage [63]. By conditionally inverting the signs of the input coordinates based on the target quadrant, the required rotation is effectively folded into the native convergence range using simple combinational logic, entirely avoiding pipeline stalls.

In contrast, for the Hyperbolic CORDIC, the concept of periodicity does not exist. Instead, the primary challenge lies in the theoretically unbounded domain of hyperbolic functions. This poses a significant hurdle when sampling the flight distance, as the input pseudo-random variable necessitates the evaluation of logarithms over a wide dynamic range.

Circular CORDIC	
Parameter	**Configuration**
Target Functions	sin, cos
Domain Extension	Quadrant Mapping
Iteration Indices	$k = 0, 1, 2, \ldots, 47$
Reference Angles	$\alpha_k^{(C)} = \tan^{-1}\left(2^{-k}\right)$
Pipeline Latency	50 clock cycles

Table 5.2. Architectural Parameters of the Unrolled Circular CORDIC Algorithm.

While the circular limitation is resolved through sign management, the hyperbolic unbounded domain is addressed through the *Expanded* Hyperbolic CORDIC algorithm, as detailed by Llamocca and Agurto [66]. This expanded formulation introduces *non-positive iterations* into the hyperbolic evaluation sequence. By extending the index set to include non-positive integers, the algorithm effectively broadens the convergence domain, allowing it to directly process the full spectrum of normalized inputs required by the stochastic sampling. From a hardware perspective, these additional negative stages are simply instantiated as extra unrolled blocks at the beginning of the datapath. This perfectly preserves the constant-time, shift-and-add topology without disrupting the global pipeline synchronization.

Expanded Hyperbolic CORDIC	
Parameter	**Configuration**
Target Function	ln, sqrt
Domain Extension	Negative Iterations $\beta = 2$ $M = -5$
Repeated Indices	$k_0 = 4$, $k_{i+1} = 3k_i + 1$
Iteration Indices	$k \leq 0$, plus repeats $(4, 13, 40)$
Reference Angles	$\alpha_k^{(Y)} = \begin{cases} \tanh^{-1}\left(1 - 2^{k-\beta}\right) & k \leq 0 \\ \tanh^{-1}\left(2^{-k}\right) & k > 0 \end{cases}$
Pipeline Latency	57 clock cycles

Table 5.3. Architectural Parameters of the Unrolled Extended Hyperbolic CORDIC Pipeline.

A critical design choice for this Monte Carlo engine is the implementation of a fully unrolled, deeply pipelined topology. Unlike iterative state-machine implementations that reuse the same physical hardware block and require a variable number of cycles to converge, the unrolled architecture instantiates a dedicated physical stage for every specific micro-rotation (including the repeated and negative ones). This structural choice guarantees a strictly deterministic, function-independent latency. Whether

the pipeline is rotating a vector in circular mode to yield a sine-cosine pair, or in expanded hyperbolic mode to extract a logarithm, the execution time remains perfectly constant. This absolute temporal determinism is vital for preventing desynchronization, ensuring a continuous throughput of one valid analytical result per clock cycle once the initial pipeline delay is overcome.

The primary architectural parameters defining the instantiated CORDIC modules are summarized in Tables 5.2 and 5.3.

5.2 Memory Architecture and State Management

In standard CPU-based Monte Carlo codes, the tracking algorithm is notoriously bottlenecked by the *memory wall*, as pointed out in Sect. 3.3.4. As neutrons independently scatter and interact, they trigger highly random, non-contiguous memory accesses to fetch macroscopic cross-sections. This completely disrupts the CPU cache hierarchy, forcing the processor to idle while waiting for data from the main RAM. To sustain the continuous dataflow of the FPGA spatial pipeline, the memory architecture must be entirely redesigned to guarantee deterministic, single-cycle data retrieval.

Serving as a proof of concept for this novel hardware programming paradigm, the architectural philosophy of the current implementation is to embed as many physical constants as possible directly on-chip. By avoiding the explicit use of synchronous memory blocks or communications with external peripherals for static data, the design eliminates the latency overhead associated with external memory interfaces.

This deliberate design choice provides a massive architectural advantage for the current pipeline: it guarantees absolute zero-latency, combinational read access to the nuclear data. By keeping static information completely combinatorial, the design entirely bypasses BRAM access latencies and read-enable synchronizations, freeing these crucial memory primitives to be used exclusively for dynamic state management, such as the asynchronous FIFO buffers of the Fission Bank.

5.2.1 Cross-Section Lookup and Material Mapping

For the prototype validation of this hardware accelerator, the macroscopic cross-section data for scattering, capture, and fission for the test materials are integrated directly into the computational fabric, as pre-processed data. As defined in the dedicated VHDL packages inside configuration modules, these tables are synthesized strictly as hardcoded combinational logic rather than being mapped to memory arrays.

Given the abundant resources of the target FPGA — the Xilinx Alveo U250 — the synthesizer elegantly optimizes these constant arrays at the physical level. Leveraging the properties of the custom $Q16.48$ fixed-point format described in Sect. 5.1.1, these cross-section constants consume no active FFs. Instead, the individual bits of the 64-bit words are physically hard-wired to the high (V_{dd}, logic 1) and low (GND, logic 0) voltage rails of the FPGA.

When a neutron enters the interaction module, a tree of Multiplexers (MUXes) constructed from the FPGA's native Look-Up Tables (LUTs) instantly routes the correct hardwired 64-bit cross-section to the output bus within the same clock cycle. This completely eradicates the cache-miss latency inherent to software simulations. While full-scale continuous-energy reactor simulations will eventually require mapping massive cross-section libraries into dedicated high-density synchronous memory blocks (such as BRAMs or URAMs), the hardcoded monoenergetic approach is optimal for the current scope. It provides the ultra-low latency required to rigorously validate the fundamental spatial tracking pipeline without introducing external memory interface complexities.

5.2.2 Secondary Particle Management and Fission Banking

A neutron traversing the reactor core will eventually undergo a nuclear interaction. For the scope of this hardware prototype, the tracked interactions are limited to *scattering*, *radiative capture*, and *nuclear fission*. While scattering merely updates the primary neutron's kinematics and capture terminates its history, fission introduces a severe structural challenge for a spatial dataflow architecture: the simultaneous birth of multiple secondary neutrons from a single incident particle.

In software-based tracking frameworks like OpenMC, this structural divergence is handled trivially. Secondary particles are instantiated dynamically at runtime via memory allocation on the heap and appended to a secondary stack or array to be simulated sequentially.

However, mapping this generative process onto an FPGA exposes a fundamental dichotomy between software and hardware design. In pure silicon, there is no operating system to manage a dynamic heap, nor does the concept of runtime memory allocation exist. The memory structures required to host the kinematic state of secondary neutrons must physically exist on the chip *before* the fission event ever occurs.

Consequently, in a spatial dataflow architecture, secondary particles are not dynamically created in memory. Instead, their generation translates to routing new kinematic data into pre-instantiated, statically sized hardware structures. As

explicitly implemented in the `eventworker` module, when the core interaction logic triggers a fission event, no dynamic memory allocation occurs. The physical memory slots destined to house the secondary neutrons already exist on the FPGA fabric, having been strictly dimensioned during synthesis.

During this state, the hardware merely evaluates the sampled number of fission children ν and orchestrates the routing of their kinematic data. Rather than requesting new memory blocks from an operating system, the state transitions to a dedicated emitting phase process. During this phase, the secondary offspring are systematically multiplexed into the pre-existing FIFO addresses of the fission bank over consecutive clock cycles. This paradigm shift exposes a critical vulnerability inherent to hardware design: if a dense fission cascade produces secondary neutrons at a rate that exceeds the fission bank's drain capacity, the statically allocated FIFO will assert a `full` flag. This instantly triggers pipeline backpressure and risks a total system deadlock, making the robust compile-time dimensioning of these hardware memory arrays a fundamental cornerstone of the accelerator's stability.

```
1  when EV_COLL_FISSION =>
2  -- indexing childer particles
3  -- omissis --
4
5  -- preparing fission bank memory
6  bank_template    <= v_out;
7  bank_template.id <= std_logic_vector(unsigned(v_out.id) +
    2);
8
9  -- immediate pipeline routing of first child int main fifo
10 -- prevents pipeline bubbles
11 -- omissis --
12
13 -- if nu > 1, activate static Fission Banking for secondary
     children
14 if fiss_nu > 1 then
15 bank_state   <= S_EMITTING; -- moving neutrons from fission
     bank to main fifo
16 bank_counter <= fiss_nu - 1;
17
18 -- The state machine will now spend the next (fiss_nu - 1)
    clock cycles
19 -- multiplexing the remaining children into the pre-
    allocated FIFO.
```

```
20    end if;
```

Listing 5.1. Simplified RTL extraction of the fission memory routing. Immediate emission logic and Text IO logging have been omitted for clarity.

To visually crystallize this hardware–native approach, Listing 5.1 provides a simplified extraction of the core RTL logic governing the fission products. The snippet highlights the complete absence of memory allocation routines, replaced by immediate data manipulation, where memory allocation is replaced by signal updates and state transitions for delayed FIFO banking are connected using multiplexers on chip.

5.3 Stream-Based Neutron Transport Pipeline

The core of the hardware accelerator is a *stream-based multi pipeline transport* architecture. In this dataflow paradigm, the physical lifecycle of a neutron is no longer a sequence of software instructions, but is mapped directly onto a sequence of dedicated RTL hardware modules. The pipeline is intrinsically designed to support multiple particles simultaneously in flight at different stages of evaluation. A quick overview of the modules is given next, based of the scheme reported in Fig. 5.1, showing the closed-loop structure of the datapath together with the relative position of various modules.

5.3.1 Pipeline Injection: FIFO Queue and Scheduler

At the origin of the pipeline lies the `scheduler` module, which natively interfaces with the primary hardware FIFO. This module acts as the global traffic director and pipeline injector. Instead of fetching scattered memory addresses like a CPU, the scheduler is built around a deterministic state machine that pops the state variables of one active neutron from the FIFO and streams it directly into the active datapath, ensuring maximum hardware utilization while respecting the synchronous read latencies of the underlying memory primitives.

Beyond basic pipeline feeding, the scheduler is responsible for critical routing and state management tasks, specifically addressing the challenges of a continuous-loop dataflow:

- **Feedback Priority and Deadlock Prevention:** The scheduler implements a strict input arbitration logic. When the pipeline emits a scattered secondary

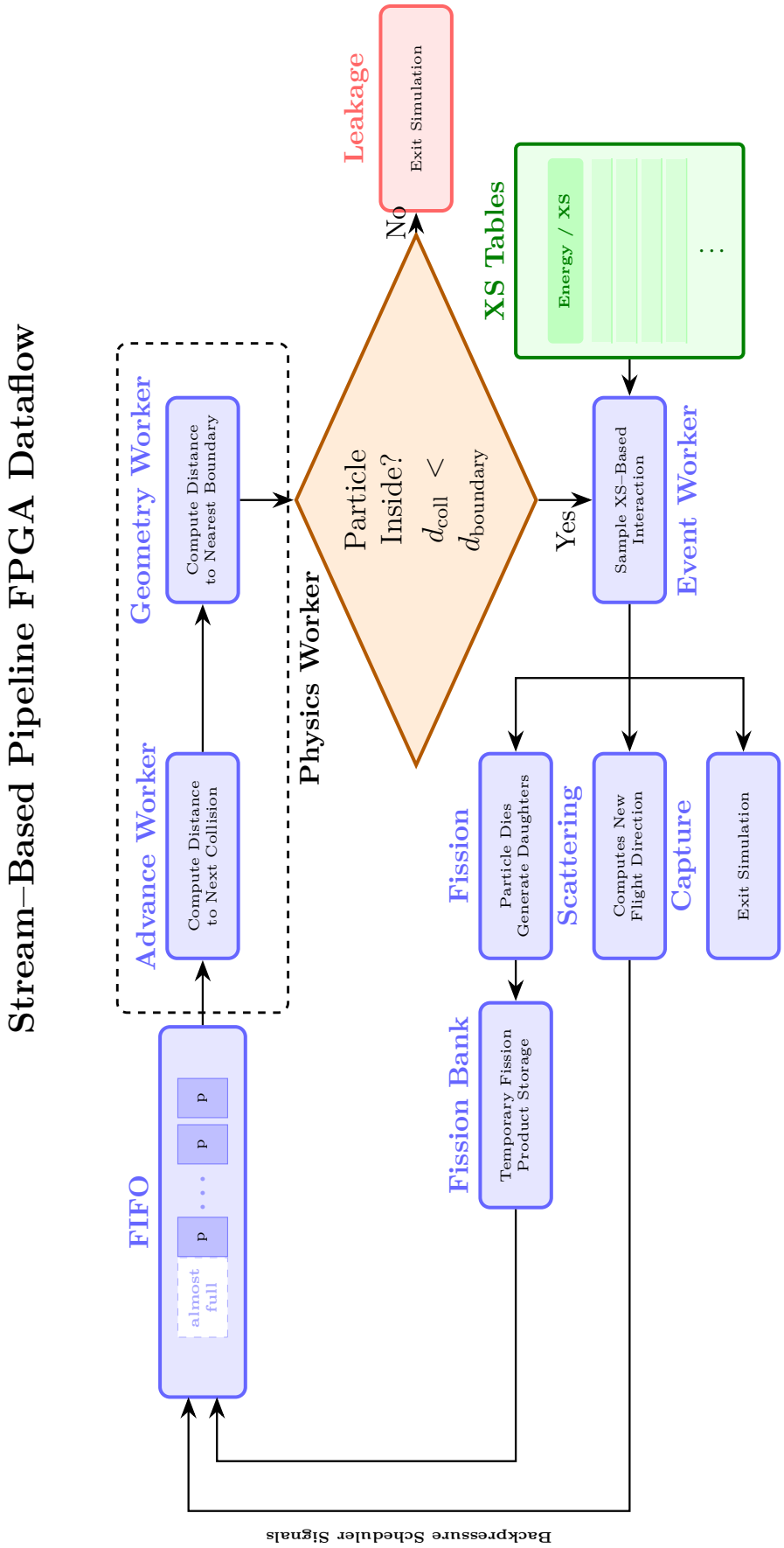

Figure 5.1. Block diagram representing the dataflow of neutrons in simulations. Arrows indicates the flowing direction: a cycle is created throw re–insertion mechanism and flow is controlled with backpressure signals to halt the simulation or control the fulx. Cross–sections enter the simulation as hardwired constant, depicted in green as a table. Each block internally encodes a set of mathematical and/or memory access and flux management operations.

particle back to the origin, this internal feedback stream is granted absolute priority over new particles injected from the external Host. To prevent catastrophic system deadlock, the scheduler employs an *almost full* threshold logic. It deliberately halts external injections before the FIFO completely fills, guaranteeing reserved memory slots for internal particles already in flight that must be recirculated.

- **Global State Tracking:** In a highly parallel spatial architecture, determining when a simulation has truly finished is non-trivial, as particles are constantly born via fission and terminated via capture or leakage. The scheduler maintains a global delta-counter that dynamically increments upon external injection or secondary birth, and decrements upon termination. The global `busy` flag is asserted continuously until this counter returns exactly to zero, signalling to the Host that the entire stochastic cascade has safely concluded.

To maintain perfect traceability throughout the simulation, the pipeline manages a sophisticated 64-bit *tree indexing* system. In traditional software Monte Carlo codes such as OpenMCÏ[40], particle genealogy is tracked dynamically by allocating new secondary particle objects on the heap and linking them via memory pointers. Because dynamic memory allocation is impossible in a pure hardware spatial architecture, the design implements a strictly bitwise, static genealogy tracker. This mechanism is based on the concept of *Implicit k-ary tree indexing*, a structural mapping closely related to Gödel numbering for sequential tree representations [67].

The primary source neutrons injected into the system are assigned an initial deterministic 64-bit ID configured as a multiple of 8. This initialization guarantees that the three Least Significant Bits (LSBs) of the source ID are strictly zero (i.e., the binary representation finishes with the sequence `000`). When a fission event occurs, generating ν secondary neutrons, the hardware inherently traverses the conceptual k-ary tree (with branching factor $k = 8$) without using physical pointers. The parent's ID is logically shifted left by three positions, and the new generation is appended by adding the remainder of the child iterator division by 8. Mathematically, the hardware executes the following combinational routing for the i-th child:

$$\texttt{ID}_\text{c} = \texttt{ID}_\text{p} \cdot 8 + (i \bmod 8) \tag{5.2}$$

This encoding entirely bypasses external software oversight and inherently prevents ID collisions between distinct fission branches. From a physical design perspective, allocating 3 bits per generation natively supports up to $2^3 = 8$ secondary neutrons per

fission event, fully satisfying the physical prompt emission distributions of standard fissile isotopes like ^{235}U and ^{239}Pu.

This deterministic indexing also dictates a hard architectural limit on the maximum trackable cascade depth. Given a 64-bit unsigned integer bus, the total number of available bits can be conceptually partitioned between the primary source particles and the subsequent generation branches. To track over 10000 primary particles — as is typical in standard Monte Carlo simulations — the algorithm requires only the first 20 bits of the bus to identify the initial generation. Because the indexing scheme demands these source IDs to be multiples of 8, these 20 bits effectively provide 17 active bits, safely accommodating up to $2^{17} = 131,072$ unique source particles per batch.

Leaving the remaining bits exclusively for the secondary generations, the maximum number of consecutive prompt fission cascades $G_{\max}$ that can be traced without overlapping any index or overflowing the register is:

$$G_{\max} = \left\lfloor \frac{64 - 20}{3} \right\rfloor = \left\lfloor \frac{44}{3} \right\rfloor = 14 \tag{5.3}$$

Using this scheme, the architecture can confidently track over 130000 primary particles through 14 consecutive, unbroken prompt fission generations. This theoretical capacity significantly exceeds the typical history length of a fast-spectrum medium before leakage or capture terminates the simulation, proving the physical adequacy of the static bitwise tracker.

5.3.2 Kinematic and Spatial Evaluation Stages

Once a particle token is injected into the pipeline, it enters the continuous spatial evaluation phase, handled sequentially by two deeply pipelined hardware blocks.

Advance Worker Within the `advanceworker` module, the theoretical distance to the next nuclear interaction is stochastically determined. The datapath interfaces with the distributed PRNG to sample a uniform random variable $\xi \in [0, 1]$ and evaluates the natural logarithm via an unrolled Expanded Hyperbolic CORDIC unit. The flight distance is subsequently computed according to Eq. (3.5), leveraging the hardware divider module introduced in Section 5.1.2.

In a strictly synchronous dataflow architecture, parallel data paths with different computational depths must be carefully aligned to guarantee data integrity. Specifically, the evaluation of $-\ln(\xi)$ and the retrieval of the macroscopic cross-section Σ_{tot} occur concurrently but possess vastly different latencies. The unrolled Hyperbolic

CORDIC introduces a deterministic latency of $L_{\ln} = 57$ clock cycles. Conversely, Σ_{tot} is retrieved via a binary search algorithm followed by linear interpolation, yielding a total sequence latency of:

$$\begin{aligned} L_{\text{xs}} &= L_{\text{search}} + L_{\text{lspline}} \\ &= \lceil \log_2(m) \rceil + 64 + L_{\text{overhead}} \end{aligned} \tag{5.4}$$

where m is the number of energy grid points, 64 represents the latency of the Radix-2 division module utilized during the interpolation, and L_{overhead} accounts for the internal state machine transitions.

Because this energy-dependent cross-section latency L_{xs} strictly exceeds the 57 cycles of the CORDIC, the computed logarithm would arrive at the final hardware divider prematurely, resulting in a corrupted mathematical evaluation. To ensure exact synchronous arrival at the inputs of the final distance divider, the faster signal path is artificially delayed. The computed logarithm and its associated particle token are routed through a dedicated synchronous shift register, acting as a programmable *delay line*. The required synchronization delay, expressed in clock cycles, is exactly the difference between the two parallel paths:

$$\Delta_{\text{align}} = L_{\text{xs}} - L_{\ln} \tag{5.5}$$

By traversing this Δ_{align}-stage shift register, the logarithm operand perfectly time-aligns with the incoming cross-section, allowing the final continuous division $d = -\ln(\xi)/\Sigma_{\text{tot}}$ to execute safely.

Geometry Tracking Phase Following the stochastic distance sampling, the neutron's physical trajectory must be evaluated against the geometric boundaries of the simulated domain. This spatial tracking phase performs continuous, hardware-accelerated ray tracing within a 3D Cartesian reactor core, modelled as a cubic volume of $1\,\text{m}^3$ centred at the origin.

Rather than evaluating boundary intersections sequentially as is typical in software codes, the proposed architecture exploits extreme spatial parallelism. Exploiting Radix-2 divider module operating perfectly in parallel, this unit simultaneously compute the exact intersection distances to the orthogonal bounding planes based on the neutron's current position and directional vector. To maintain absolute kinematic data consistency, the neutron's state variables are routed through a matching 64-stage shift register, ensuring they arrive at the post-division logic at the exact clock cycle the boundary distances are definitively resolved.

Upon exiting the deeply pipelined dividers, a combinational comparator tree instantly identifies the minimum of the three values, establishing the precise distance to the nearest geometric surface. The fundamental Monte Carlo trajectory condition is then resolved by comparing this geometric boundary distance against the previously sampled nuclear collision distance: if the collision distance is strictly shorter, the neutron undergoes a nuclear interaction within the medium, encoded in the state `OP_COLLISION`, shifting the control of the simulation to the `eventworker` module; conversely, if the boundary is closer, the particle hits the geometric limit and escape. The next event will be `OP_CROSS_SURFACE`, which manage the deadline of particle's history.

The particle's 3D spatial coordinates are subsequently updated using fully parallel Q16.48 fixed-point multipliers, computing the new position as $\boldsymbol{x}' = \boldsymbol{x} + d \cdot \hat{\boldsymbol{\Omega}}$. Since linear combinations are mapped into hardware DSP and the fixed-point multiplication produce a number in $Q32.96$ fixed-point format, a subsequent slicing in necessary to select the relevant digits, as suggested in the comment in Listing 5.2. In the context of this specific mono-cell proof-of-concept, crossing the bounding surface effectively flags the neutron as a Leakage event, terminating its physical history and dropping its token from the active pipeline.

To summarize the continuous dataflow and demonstrate the hardware-native implementation of these operations, Listing 5.2 presents a condensed extraction of the core RTL logic governing this geometric evaluation. This snippet highlights the three fundamental steps of the tracking phase: the combinatorial resolution of the minimum boundary distance, the trajectory decision, and the fully parallel spatial update exploiting the custom fixed-point format.

```
-- 1. Combinatorial tree for minimum boundary distance
d_bound := dist_x_raw;
if dist_y_raw < d_bound then
d_bound := dist_y_raw;
end if;
if dist_z_raw < d_bound then
d_bound := dist_z_raw;
end if;

-- 2. Trajectory Decision: Collision vs Boundary Crossing
if p.dist_coll < d_bound then
d_travel     := p.dist_coll;
p.nextop     := OP_COLLISION;
p.dist_bound := d_bound - p.dist_coll; -- Save remaining
```

```
    bound
15  else
16  d_travel      := d_bound;
17  p.nextop      := OP_CROSS_SURFACE;
18  p.dist_coll   := p.dist_coll - d_bound; -- Save remaining
    coll
19  end if;
20
21  -- 3. Parallel Spatial Update (Q16.48 Fixed-Point
    Multiplication)
22  -- Q16.48 * Q16.48 yields Q32.96. Slice [111 downto 48]
    restores Q16.48.
23  mult_res_x := p.dir.vx * signed(d_travel);
24  p.pos.x    := p.pos.x + mult_res_x(111 downto 48);
25
26  -- (Y and Z coordinates are updated concurrently)
```

Listing 5.2. Core RTL logic of the Geometry Worker. Redundant axis updates (Y, Z) and pipeline synchronization signals are omitted for brevity.

5.3.3 Terminal Routing: Collision Event Resolution

Neutrons that successfully advance within the geometry without leaking enter the terminal physics routing stage. Here, the specific nuclear interaction at the collision site is determined based on the macroscopic cross-sections Σ_{tot} fetched from the ROM configuration. Depending on the stochastically sampled reaction, the continuous data stream is dynamically routed into one of three distinct hardware branches.

Scattering If a scattering event occurs, the datapath executes an *isotropic scattering* kernel. To sample the new trajectory in 3D spherical coordinates, the hardware evaluates the kinematic transformation vector $(\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$ in the following form: two random numbers are extracted ξ_1 and ξ_2, both from a uniform distribution in range $[0, 1]$. Next, the following transformations are applied

$$\xi_1' = 2\xi_1 - 1 \tag{5.6a}$$

$$\xi_2' = 2\pi\xi_2 \tag{5.6b}$$

so as to uniformly map ξ_1' and ξ_2' into $[-1, 1]$ and $[2, 2\pi]$, respectively. The full transformation vector is then computed as

$$\hat{\mathbf{\Omega}} \equiv \left(\sqrt{1-\xi_1'^2}\cos\xi_2', \sqrt{1-\xi_1'^2}\sin\xi_2', \xi_1'\right) \tag{5.7}$$

This complex spatial rotation is physically executed utilizing a *dual CORDIC pipeline* operating in parallel: an unrolled hyperbolic CORDIC computes the necessary square roots, while a circular CORDIC synchronously generates the sine and cosine values.

Fission When a fission event is selected, the incident neutron token is absorbed and the secondary neutron yield ν is sampled. To maximize throughput and prevent the formation of *pipeline bubbles* this module does not instantiates a PRNG, but recieves it as input data from the event worker and selects the number ν using an `if` statement, mapped into a hardware MUX. This choice ensure a 1 clock cycle latency for the setup of the fission interaction. The first daughter particle is instantly assigned its updated ID and routed directly to the primary output bus to continue the simulation without stalling. Only the remaining offspring are pushed into the local Fission Bank FIFO for delayed processing, seamlessly bridging the generation gap and preserving strict dataflow continuity, as outlined in Sect. 5.2.2.

Absorption If the neutron undergoes radiative capture, the particle is physically terminated. Unlike software simulations that actively deallocate memory objects, the hardware simply invalidates the particle's token, immediately dropping it from the active dataflow. Because the strictly synchronous pipeline relies on a continuous stream of data, the sudden termination of a token inherently introduces a *single-cycle bubble*: an empty, non-operational execution slot into the downstream datapath. While this momentarily stalls the arithmetic utilization of subsequent stages for exactly one clock cycle, it effectively reduces the feedback pressure on the global FIFO queues. Ultimately, this hardware invalidation signals the permanent completion of that specific particle's random walk, allowing the scheduler's global counter to decrement.

5.3.4 Backpressure Signals: Closing the Loop

The structural integrity of this stream-based architecture relies heavily on inter-module control signals, specifically backpressure mechanism. Since the pipeline modules operate autonomously, variable-time events — such as the draining of the

fission bank or different datapath associated to different sampled events — can create data congestion.

To prevent token collisions, the `eventworker` module asserts a hardware stall signal to the upstream `geometryworker` and `physicsworker` whenever the local fission bank is emptying its secondary neutrons. This backpressure temporarily halts the dataflow for 2 or 3 clock cycles, depending on the sampled ν from Par. 5.3.3. Furthermore, neutrons that survive a scattering event, or daughter neutrons born from fission, are routed through a feedback loopback directly into the main FIFO. Finally, to prevent two neutron from entering the FIFO at the same time, the `scheduler` gives high priority to already circulating data. This handshake, prioritized protocol ensures that the particle FIFO never overflows and that active tokens are recycled seamlessly for their next physical iteration.

5.3.5 Hardware Performance and Pipeline Latency

The complete integration of all modules culminates in a highly deterministic dataflow engine. From the moment a neutron token is popped from the FIFO to the moment its surviving state is pushed back via the loopback, the intrinsic latency of a single particle path is non-constant, depending on the interaction selected by the interaction roulette inside the `eventworker` module. Indeed, the longest case scenario is the one associated to scattering, leading to the longest latency of ~ 215 clock cycles.

However, due to the fully pipelined nature of the RTL design, this latency is effectively hidden, once the pipeline is filled with particles. As illustrated in Fig. 5.2, CPU-based software relies on nested `for` and `while` loops to track particles sequentially. In that paradigm, the CPU must finish processing neutron n_1 entirely before beginning neutron n_2, accumulating the latency linearly.

In contrast, the spatial FPGA architecture accepts a new particle token into the pipeline at every clock cycle. Once the initial pipeline stages are filled, the architecture achieves its target Initiation Interval (II) of 1. As shown in the overlapping blocks of the FPGA timeline, the engine yields a theoretical processing throughput of **one particle physical iteration per clock cycle**, paving the way for simulating thousands of concurrent particles on modern FPGA fabrics regardless of the deep mathematical latency of the individual interaction kernels, while keeping the processes of different particles independent.

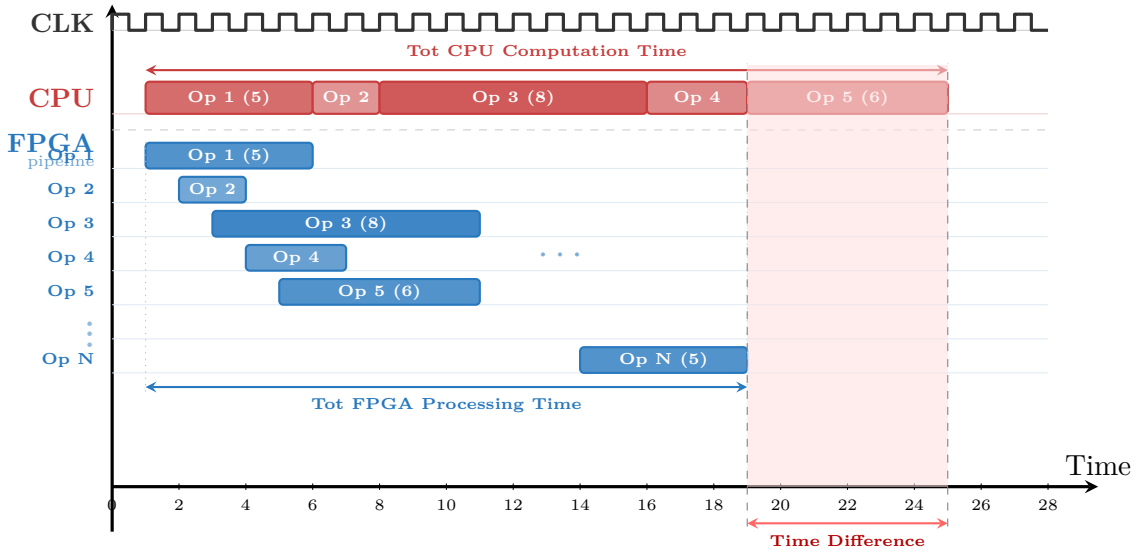

Figure 5.2. Comparison of the execution timelines. Top: the sequential processing timeline characteristic of CPU software loops. Bottom: the overlapping, event–driven pipeline of the FPGA architecture, demonstrating an Initiation Interval (II) of 1 clock cycle.

5.4 Testing and Validation Results

Before deploying the synthesized RTL architecture onto physical FPGA silicon, the structural and mathematical integrity of the spatial pipeline must be rigorously validated. This section details the simulation environment, the testbench methodology, and the preliminary results obtained from evaluating the hardware's continuous dataflow. The hardware validation was conducted using both GHDL, an open-source, high-performance VHDL simulator configured strictly to the VHDL-2008 standard and Vivado IDE. The former has been used to print simulation logs and waveform, while the latter also allowed for the implementation of the code into firmware on the physical chip. Using this tool one can see how many resources are used when the code is printed into silicon and the connection between them, as shown in Tab. 5.4.

LUT	**FF**	**BRAM**	**URAM**	**DSP**	**WNS** (ns)
107287	222274	23	0	146	0.394

Table 5.4. Resources used for firmware printing on FPGA.

Comparing these data with the Alveo U250 available resources in Tab. 4.1 one can see that at least 5 simultaneous instances of the code can be used on the same chip, allowing for a $\times 5$ multiplication factor in data processing.ed the true power of spatially parallel computation provided by FPGAs.

5.4.1 Methodological Approximations and Validation Scope

A critical distinction must be made regarding the handling of physical interactions within this hardware proof of concept. The primary objective of this prototype is to rigorously validate the spatial tracking dataflow, including the Monte Carlo kinematics and the inter-module FIFO and back-pressure logic.

While the collision evaluation stage utilizes realistic macroscopic cross-sections derived from nuclear data libraries to determine interaction probabilities and branching ratios, the post-collision kinematics do not update the neutron's kinetic energy. Consequently, every secondary or scattered neutron retains its initial birth energy. This decoupling strategy is indeed intentional: it ensures that the physical probability ratios governing the control path remain realistic, while isolating the spatial datapath for precise timing and bottleneck analysis.

Implementing full energy-dependent kinematics — specifically, stochastic energy degradation during scattering and prompt spectrum sampling during fission — stands as the highest priority for future development to achieve complete physical realism. Nevertheless, this monoenergetic constraint was a strictly necessary intermediate validation step. Attempting to simultaneously debug a completely novel, deeply pipelined spatial routing architecture alongside a complex continuous-energy datapath would have conflated structural hardware issues with purely physical variance. By temporarily freezing the energy variable, this prototype definitively proves the robustness of the continuous dataflow paradigm, providing a fully verified and stable hardware foundation for the future integration of complete continuous-energy Monte Carlo tracking.

5.4.2 Numerical Precision: Hardware VS CPU Baseline

The fidelity of the simulation starts from the numerical results given by the model. The choice of the numeric format has been dictated by the structure of the FPGA, as discussed in Sect. 5.1.1, preliminary tests addressed the comparison between fixed-point precision arithmetic, for hardware simulation, and IEEE 754 double-precision format, used in software simulations.

Standalone precision tests has been performed as follows: individual arithmetic VHDL modules have been used to compute trigonometric function, natural logarithms and square roots. The resulting number has been translated into a floating-point precision number using a simple C++ script and relative errors have been computed for each function. The fixed-point to floating-point precision conversion functionif

the following

$$x^{(\text{IEEE754})} = -b_{15} \cdot 2^{15} + \sum_{k=-48}^{14} b_k \cdot 2^k \tag{5.8}$$

since the MSB in two's complement representation has negative weigth. Numbers b_k are 0 or 1, generating the binary format string

$$x_{(2)} \equiv b_{15} b_{14} \cdots b_2 b_1 b_0 . b_{-1} b_{-2} \cdots b_{-47} b_{b-48} \tag{5.9}$$

corresponding to the fixed-point precision representation of the number.

Circular CORDIC, Rotating Mode This module is used to compute trigonometric function in outgoing isotropic scattering direction evaluation. In order to test angles both in the convergence region and in the full 2π angle the following tests have been performed. Results are reported in the following tables

Output of Circular CORDIC Module (Rotating Mode)

Input	**Expected** (0x)	**Output** (0x)	**Rel. Error**
0	0001 0000 0000 0000	0001 0000 0000 0009	3.2×10^{-14}
	0000 0000 0000 0000	FFFF FFFF FFFF FFFE	7.1×10^{-15}
$\pi/6$	0000 DDB3 D742 C265	0000 DDB3 D742 C26D	1.3×10^{-15}
	0000 7FFF FFFF FFFF	0000 8000 0000 0005	3.5×10^{-14}
$5\pi/6$	FFFF 224C 28BC 3D9B	FFFF 224C 28BD 3D93	3.1×10^{-14}
	0000 7FFF FFFF FFFF	00008 0000 0000 0005	3.6×10^{-14}

Table 5.5. Numerical precision comparison between the $Q16.48$ hardware Circular CORDIC module and the IEEE 754 double-precision CPU baseline. Top values refer to cos function and bottom values to sin function.

Extended Hyperbolic CORDICH, Vectoring Mode The same approach has been used to test Extended Hyperbolic CORDIC module, testing the precision of natural logarithm and square root functions, crucial in computing distance to next collision and post scattering direction, respectively. Few significant numbers have been chosen in order to test function properties, representable numbers and numeric precision: results are shown in the next table

The right-most column of the Tabs. 5.5 and 5.6 shows relative precisions ranging from 10×10^{-10} to 10×10^{-15}, confirming that for the purpose of this proof of concept the $Q16.48$ format, and specifically, the 48 fractional bits provide sufficient resolution to evaluate continuous scattering angles and stochastic distances without introducing significant quantization bias. A notable exception is $\sqrt{0}$, which shows a significant

Output of Extended Hyperbolic CORDIC Module (Vectoring Mode)

Input	Expected (0x)	Output (0x)	Rel. Error
$\ln 1$	0000 0000 0000 0000	0000 0000 0000 0302	1.8×10^{-13}
$\ln 2$	0000 B172 17F7 D1CF	0000 B172 17F7 DD46	1.5×10^{-11}
$\ln e$	0001 0000 0000 0000	0000 FFFF FFFF EE3A	1.6×10^{-11}
$\sqrt{0}$	0000 0000 0000 0000	0000 0000 10D2 E8F4	$\mathbf{1.002 \times 10^{-6}}$
$\sqrt{1}$	0001 0000 0000 0000	0000 FFFF FFFF FFF1	5.3×10^{-14}
$\sqrt{2}$	0001 6A09 E667 F3BC	0001 6A09 E667 F3B3	2.5×10^{-15}
$\sqrt{4}$	0002 0000 0000 0000	0001 FFFF FFFF FFF3	2.3×10^{-14}

Table 5.6. Precisione Tests of Extended Hyperbolic CORDIC algorithm. Used in firmware to compute natural logarithm and square roots, in computing flight distances and post scatteringdirections.

displacement from the trend, mainly due to the very particular choice of the initial angle.

5.4.3 Dataflow Validation

With the mathematical primitives verified, the validation shifted to the system level. The primary testbench injected a batch of source neutrons into the scheduler's FIFO to monitor the inter-module handshake protocols and backpressure mechanisms.

The simulation successfully demonstrated the robust execution of the primary physical events:

- **Continuous Tracking and Leakage:** As shown in Fig. A.1, the `fleakage` signal detects a particle escaping the geometry. The corresponding `finished_valid` also indicates that the particle's history is coherently terminated. The signal `busy` always high indicate that particles are flowing through the architecture continuously, being processed at each cycle.

- **Fission Multiplicity and Backpressure:** The most critical dataflow event is captured in Figure A.2. Upon evaluating a fission event, the 64-bit Tree Indexing system successfully assigned unique IDs to the daughter neutrons. As the neutron's ID is only displayed when a particle dies, using the constrain $ID > 100_{(10)}$ and `finished_valid` signal high, the Vivado Integrated Logic Analyzer (ILA) was able to detect a fission product escaping the geometry, as reported in Fig. A.3

- **Capture Event and Removal:** Capture event is highlighted in the Vivado ILA by the combination of `fabs` and `finished_valid` signals. Indeed the

`alive` attribute of the corresponding particle is set to 0, removing it subsequent processing. A visualization of the associated waveform is shown in Fig. A.4.

These three events cover all the possibility for a particle's history to terminate in the hardware simulation. However, scattering events require the tracking of direction of flight distant in time (scattering module has an intrinsic latency of 60 cycles) associated to the same `id`.

Finally, as a direct proof that post interaction product preserve their initial energy one can look at signal `energyoutslv` in Fig. A.3: rather than a Watt distributed energy the particle has 1 MeV of energy before escaping the geometry.

5.5 FPGA vs Software Benchmarck

In order to objectively validate hardware architecture performances, a direct comparison with OpenMC framework has been performed. The software was run on a general–purpose CPU, in single–thread configuration.

Test splits into two phases: a small sample validation benchmark and an analytical projection towards realistic workloads. The former validates the functionality of the architecture, also providing a *worst case* scenario in computing effort. The latter aims to estimate the realistic workloads of such Monte Carlo simulations.

5.5.1 Small Benchmark Configuration

Both hardware and software simulations share the same input configuration data: geometric volumes, nuclear data and simulation parameters. This choice guarantees a direct comparison of the two working paradigms, with numerical values shown in Tab. 5.7.

Small sample size is intentional and dictated by the following necessities

- *not overstressing FIFO queues*, keeping the goal of benchmark, validating the architectural dataflow and logic, away from critical conditions.
- *recording and handling deadlocks*: these events must be accounted for during developement as they may induce stall situations in the FIFO or block the access to memory resources.

The reduced sample size translates into a poor interaction event count, as can be seen from Tab. 5.8. Indeed, using the OpenMC tally for counting interactions a very blurred result come up: an average number of 59.1 interaction per particle, with a statistical uncertainty of ±44.0. While useful for our validation purposes, these

Parameter	Value
Mode	fixed source
Geometry	$1\,\mathrm{m}^3$, origin in $(0,0,0)$
Boundary Condition	Vacuum
Material	$\left({}^{238}\mathrm{U}, {}^{235}\mathrm{U}, {}^{10}\mathrm{B}\right)$
Composition	$(99.8\%, 0.1\%, 0.1\%)$
Source	Point–like Monoenergetic $E = 1\,\mathrm{MeV}$ Isotropic
Particles in Source	100
Batches	1

Table 5.7. Parameters of the small benchmark.

numbers are clearly not reliable for a stable Monte Carlo estimate of the expected number of interaction.

Positional Index	Value
Total Collisions	5913
AVG / particle	59.1
Median	52
Dev. Std.	44.0
Min — Max	1 — 233

Table 5.8. OpenMC tally result for average number of interaction per particle.

5.5.2 Realistic Workload Projection

OpenMC software simulation has been run using `OMP_NUM_THREADS` set to 1 so as to force the execution to use only one CPU thread, as in the case of hardware simulation. OpenMC final report shows the following statistics about simulation time. The net

Metric	Value
Initialization	627 ms
Transport Only	6.65 ms
Praticle Process Rate	$11255\,\mathrm{particles} \cdot \mathrm{s}^{-1}$

Table 5.9. OpenMC timing analysis results. Taken from the summary generated at the end of the simulation.

time for neutron transport only is 6.65 ms, together with a rate of 11255 particles processed per second.

5.5.3 Estimate Hardware Performance

Hardware pipeline implementation has an intrinsic latency of 215 clock cycles. This means that the first result will be available after $215 \times 10\,\text{ns}$ and subsequent results will be available after one clock cycle each, i.e. 10 ns one after the other. Having chosen the same input setup, it is reasonable to assume the same average number of 5913 total interactions. This choice yields a computation time of

$$T_{\text{scatt}}^{(\text{FPGA})} = \frac{L_{\text{pipeline}} + (N-1)}{f} = \frac{215 + (5913-1)}{100\,\text{MHz}} = 0.061\,27\,\text{ms} \tag{5.10}$$

providing a computation speedup of

$$G^{(\text{FPGA})} = \frac{T_{\text{scatt}}^{(FPGA)}}{T^{(CPU)}} = \frac{0.061\,27\,\text{ms}}{6.65\,\text{ms}} = 108 \tag{5.11}$$

This is indeed the *best scenario* where the pipeline is always full of data and no stalls or deadlocks occur. Beside this extraordinary, but rather optimistic gain (relative to the sample size), also the *worst case* scenario has been estimated. Indeed, in this multi-pipeline structure, where neutrons can undergo leakage or capture effectively leaving the simulation, the active data is replaced with a *bubble* of inactive data. This bubble propagates into the datapath effectively invalidating intermediate results and propagating a delay into the pipeline equal to the latency. In this worst scenario we can assume 99 particle to undergo either leakage or absorption and the last one to produce a valid output. Further assuming that half of them undergo absorption and the other half undergo leakage, they will introduce a delay in the pipeline respectively equal to

$$\Delta_{\text{leak}} = 0.5 \times 99 \times 147 = 6542 \tag{5.12a}$$

$$\Delta_{\text{capture}} = 0.5 \times 99 \times 224 = 9968 \tag{5.12b}$$

clock cycles delay, for a total of

$$\Delta = \Delta_1 + \Delta_2 = 16510 \tag{5.13}$$

The new computing time $T_{\text{bubble}}^{(\text{FPGA})}$ is then

$$T_{\text{bubble}}^{(\text{FPGA})} = \frac{\Delta + L_{\text{scatt}}}{f} = \frac{16510 + 215}{100\,\text{MHz}} = 167.25\,\mu\text{s} \tag{5.14}$$

Even in this worst case scenario the pipeline structure compensate the lower compu-

tation performances yielding as gain factor of

$$G_{\text{bubble}} = \frac{T_{\text{CPU}}}{T_{\text{bubble}}^{(\text{FPGA})}} = \frac{6.65\,\text{ms}}{0.167\,25\,\text{ms}} = 40 \tag{5.15}$$

achieving one order of magnitude of computation speedup.

5.5.4 Analytical Projection onto Real–Scale Simulations

The previous calculations demonstrate that the hardware architecture provides a notable *speedup* compared to software execution on a local CPU, even under unfavorable testing conditions. However, for such small populations, the absolute performance of the FPGA is intrinsically limited by the continuous filling and partial emptying of the pipeline, referred to as *pipeline priming* and *flushing*. While this transient scenario is sufficient to validate the design and outperform a domestic *general-purpose* processor, the absolute computation rates are not yet comparable to those of large-scale HPC facilities [68, 69]. Realistic nuclear Monte Carlo simulations require millions of histories to reduce statistical variance. By scaling to these larger workloads, the impact of *pipeline bubbles* becomes negligible, allowing the architecture to reach its asymptotic potential.

The primary advantage of the spatial dataflow paradigm emerges upon reaching steady-state conditions. As N increases, the number of bubbles becomes statistically insignificant relative to the continuous data stream fed by fissions and loopbacks, leading the average throughput Γ to converge toward unity, i.e. $\Gamma_0 \approx 1$.

To analytically estimate the asymptotic efficiency of the system and compare it with the standard metric of *Particle Histories per Second* (denoted as H/s), reference is made to the literature on FPGA acceleration [70]. The asymptotic rate R_{h} is given by:

$$R_{\text{h}} = \frac{f \cdot \Gamma_0}{\langle n_{\text{int}} \rangle} \tag{5.16}$$

To obtain an estimate of this computation rate, the OpenMC output for the mean interactions per particle can be used. According to Tab. 5.8, $\langle n_{\text{int}} \rangle \approx 59.1$ for the selected geometry. Substituting these values into Eq. (5.16) yields a particle history rate equal to:

$$R_{\text{h}} = \frac{100\,\text{MHz} \cdot 1}{59.1} \approx 1.69 \times 10^6\,\text{histories} \cdot \text{s}^{-1} \tag{5.17}$$

This projection suggests that, provided all necessary data and logic blocks can be successfully placed and routed on a single chip, a single FPGA could compete with multi-core HPC infrastructures.

Although these are preliminary analytical estimates, this asymptotic projection

highlights the competitiveness of the proposed architecture against modern heterogeneous HPC infrastructures. With a theoretical rate estimated at over 1.6×10^6 histories per second, the hardware design exceeds the typical 1.5×10^4 histories per second of a standard multi-core CPU server node by two orders of magnitude. Furthermore, it aligns with recent benchmarks achieved by exascale-class GPU accelerators, which have recently surpassed the 10^6 histories/sec threshold per single board on complex geometries [68, 69].

Moreover, it is important to note that this theoretical computation rate refers to a single pipeline instance, i.e. a single computational unit mapped into the FPGA. Modern accelerators, such as the Xilinx Alveo U250 FPGA, are equipped with extensive logic resources, DSPs, and internal memory blocks, providing the margin to implement spatial parallelism. By instantiating multiple independent replicas of the firmware on the same chip — each tasked with processing a separate batch of particles — the overall system throughput would scale linearly under ideal conditions

$$R_{\mathrm{h}}^{(k)} = k \cdot R_{\mathrm{h}} \tag{5.18}$$

In such a deployment scenario, a single FPGA device could potentially exceed the absolute performance of a high-end GPU, while concurrently providing higher energy efficiency (*performance-per-watt*) due to its lower operating frequencies. Indeed it was measure a Total On-Chip Power Consumption of only 3.855 W for a single instance firmware.

Finally, to contextualize these estimate, it is important to interpret them in light of some limitations of the actual prototype

1. the *absence of tallies* eliminates the overhead present in OpenMC framework, slightly reducing the latency of the pipeline.

2. the absence of post collision realistic kinematics limitate the fidelity of a direct comparison of the two Monte Carlo suites.

As the present results place the FPGA among the top–level computation architecture for Monte Carlo simulation in neutron transport field, the implementation of the two above–cited features will inevitably refine these estimates.

Conclusions and Future Developments

The primary objective of this thesis was to demonstrate the feasibility of migrating Monte Carlo neutron transport from a traditional, instruction–based software model to a fully unrolled, spatially parallel architecture using FPGAs. Rather than attempting a literal 1:1 code porting of existing C++ codebases into VHDL, this work serves as a foundational proof-of-concept for a new hardware programming paradigm, explicitly bridging the disciplines of digital systems engineering and stochastic particle physics. By decoupling the mathematical equations from the memory-bound constraints of CPU caches, this project successfully established a scalable, hardware-native tracking engine capable of sustaining a deterministic and predictable processing rate.

Crucially, this architecture highlights a new frontier for programmable logic in scientific computing. While FPGAs are already utilized massively within the HEP community for online DAQ and ultra-low latency trigger systems, the successful execution of this spatial Monte Carlo pipeline demonstrates their formidable versatility for computationally intensive *offline* simulations.

To achieve this, the synthesized prototype resolved several critical architectural challenges. The implementation of a 64-bit custom arithmetic and logic unit, outlined in Sect. (), capable of handling complex stochastic kinematics — such as realistic 3D isotropic scattering — can evaluate analytical function entirely in hardware-native numeric format, using fixed-point precision numbers, and achieving sufficient numerical precision these scopes.

As computational efficiency is concerned, estimates obtained in Sect. 5.5.4 consciously projects the FPGAs among the elite group of electronic devices dedicated to Monte Carlo simulation for nuclear engineering. In the next developments of this project, native tally and statistics must be added in order to refine these numbers, or, even better, improve them.

Built upon this validated monoenergetic dataflow, the natural evolution of the architecture involves the complete integration of continuous-energy physics. High priority in subsequent developments must be given to the implementation of energy–dependent interaction products, together with the inclusion of tallies for Monte Carlo estimates.

The choice of keeping nuclear data stored as constant values hardcoded in silicon proved to be highly efficient in fetching data continuously, without stalling the

pipeline architecture of the dataflow. Indeed, synchronization of corresponding data elevates the fetching operation to a simple physical wiring in silicon, requiring zero resources.

This choice has been primarily dictated by the small amount of nuclear and geometric data needed for the current setup. Nevertheless, in future, realistic prototypes exploitation of others resources, such as BRAMs and URAMs, or the cooperation of host software and custom hardware computational unit might also be explored.

Looking toward the full-scale simulation of Gen-IV fast reactors, future development phases must also expand the geometric and diagnostic capabilities of the engine. The current 3D Cartesian ray tracer provides a robust foundation that has to be scaled into a fully featured CSG module, potentially distributing distinct physical reactor regions across multiple networked FPGAs to further parallelize the workload. Finally, the realization of a production-ready hardware accelerator will necessitate the implementation of a distributed hardware tallying system. By embedding dedicated accumulation registers (tallies) alongside the interaction workers, critical reactor parameters such as scalar fluxes, reaction rates, and the effective multiplication factor can be evaluated on the fly without stalling the active particle streams.

In conclusion, the developed prototype successfully validates the proposed hardware architecture, establishing a solid foundation for future development. Addressing the inherent complexities of nuclear particle transport and FPGA design will require continuous, interdisciplinary efforts to expand, optimize, and maintain the codebase. Ultimately, the positive results achieved in this study confirm the viability and the potential of the proposed spatial dataflow architecture. Despite the existing margins for optimization, this prototype represents a successful and concrete step toward providing a scalable, high-performance computing tool for the simulation of advanced Gen-IV nuclear systems.

In addition, my personal contribution to this thesis reproduces neutron diffusion simulations using high-level description algorithms implemented on commercial hardware platforms. This work was carried out using firmware design codes, applying and exploiting the intrinsic parallelism of FPGA devices, thereby accelerating calculation times by approximately two orders of magnitude.

A Vivado IDE Output Graphs

The following appendix summarizes the output recorded using thee Vivado IDE software. Looking at simulation waveform one can visualize digital signal in the same way they move inside the FPGA, due to the deterministic latency and strict timing closure constraints.

On the one hand using the *Virtual Input Output (VIO)* tool one can give individual input to the FPGA visualizing the response as output updates according to firmware description, whereas using the *ILA* tool one can visualize waveforms as in a traditional oscilloscope. Both these tools have been intensively used both to *debug detection* and *result collection* aiming to find events of interest such those reported in the following.

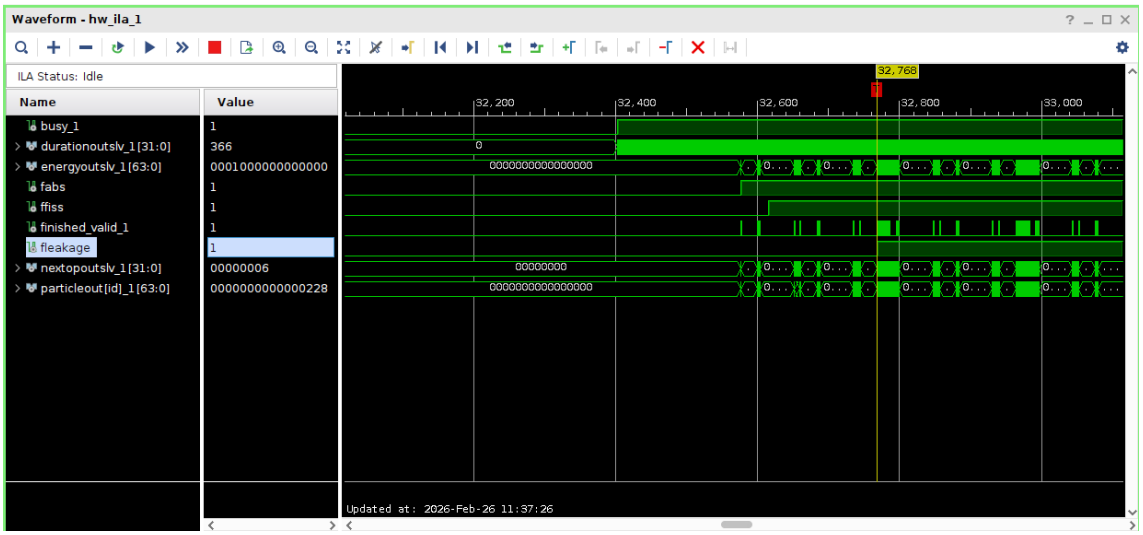

Figure A.1. Detection of a particle escaping the reactor's geometry. Accordingly, both `finished_valid` and `fleakage` flags are activated. The former invalidates the token, while the latter discriminate the event in the `eventworker` module. This kind of event creates a bubble in the pipeline.

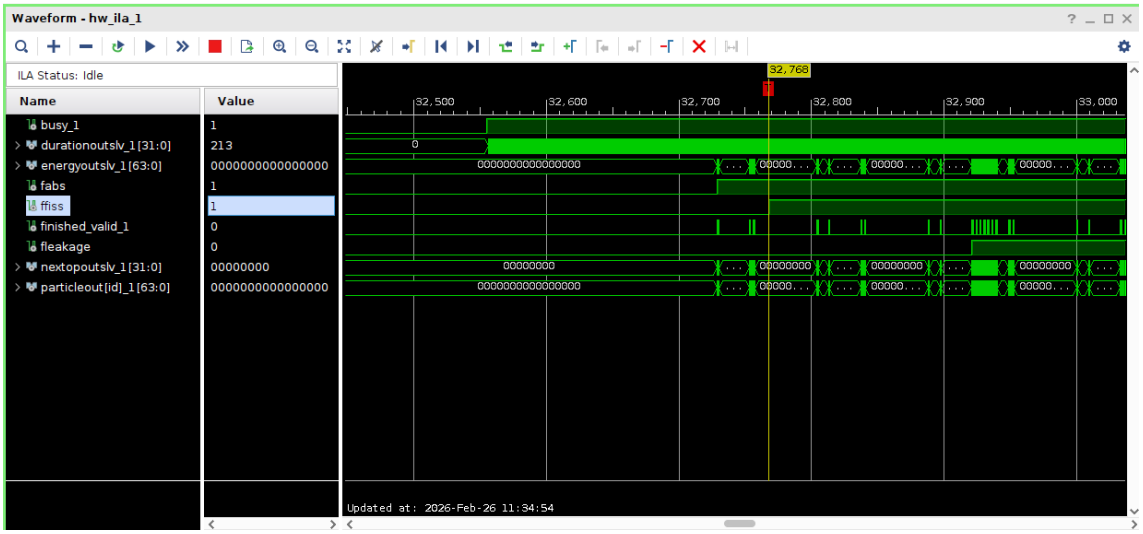

Figure A.2. Fission event from a first generation particle. According to Tree Indexing, the `0x80` particle id is associated to the first generation.

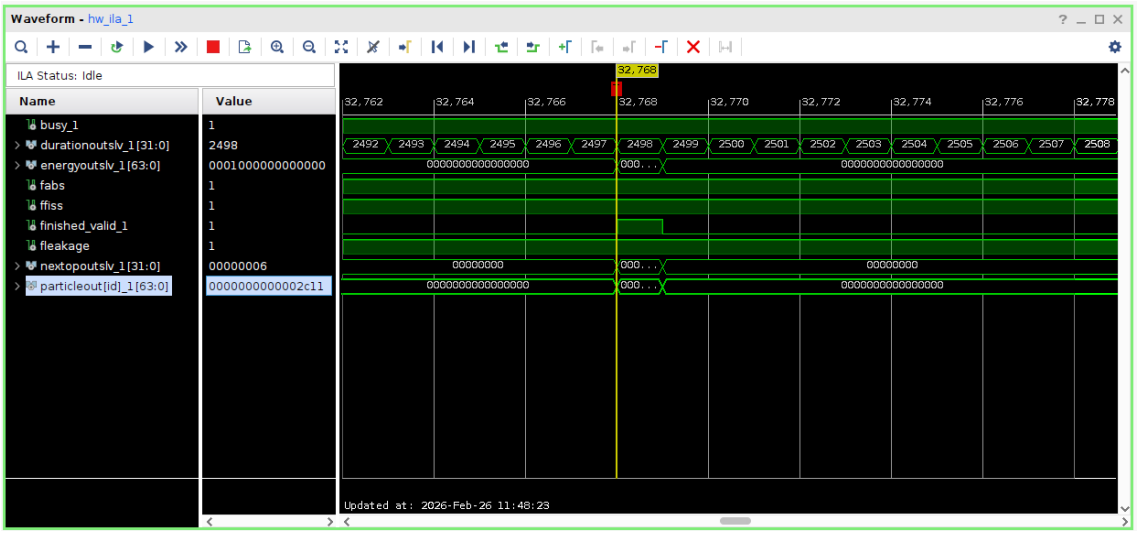

Figure A.3. Leakage event coming from a fission produced particle.

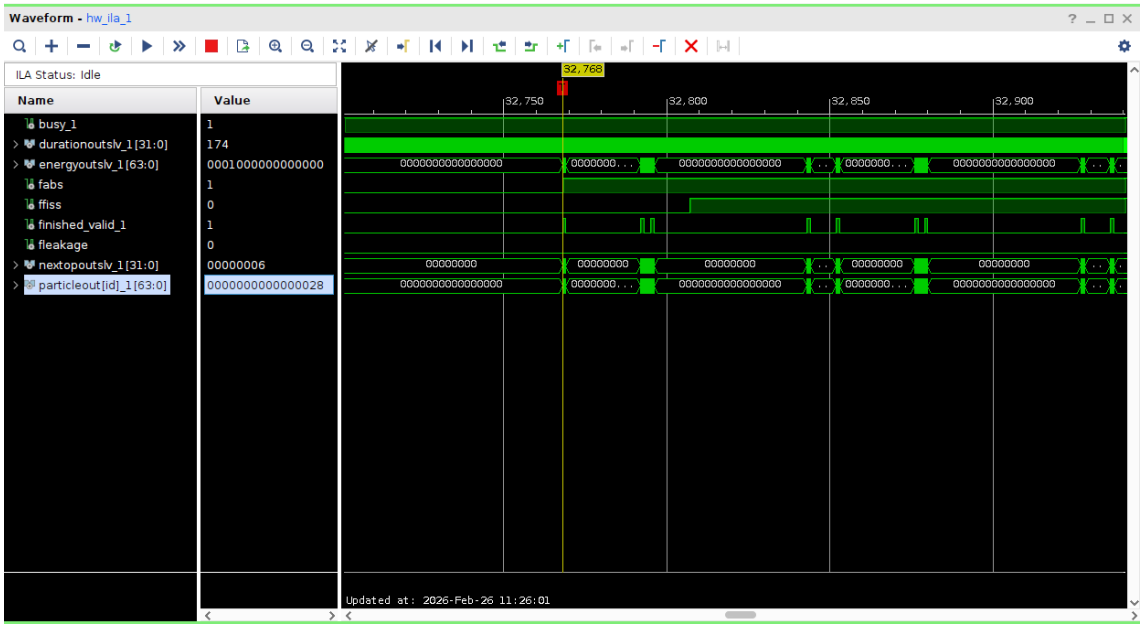

Figure A.4. Absorption flag indicate a neutron being absorbed by the reactor material, essentially representing neutron capture from ^{10}B.

Bibliography

[1] Kenneth S. Krane. *Introductory Nuclear Physics*. New York: John Wiley & Sons, 1988 (cited on pp. 2, 5, 7, 9).

[2] John S. Lilley. *Nuclear Physics: Principles and Applications*. Chichester: John Wiley & Sons, 2001 (cited on p. 2).

[3] Weston M. Stacey. *Nuclear Reactor Physics*. 2nd. Weinheim, Germany: Wiley-VCH, 2007 (cited on pp. 3, 25).

[4] Paul Reuss. *Neutron Physics*. EDP Sciences, 2008 (cited on p. 4).

[5] James J Duderstadt and Louis J Hamilton. *Nuclear Reactor Analysis*. John Wiley & Sons, 1976 (cited on pp. 5, 17, 19, 25, 38).

[6] John R Lamarsh and Anthony J Baratta. *Introduction to Nuclear Engineering*. 3rd. Prentice Hall, 2001 (cited on pp. 6, 14, 21, 25, 29).

[7] D. A. Brown et al. "ENDF/B-VIII.0: The 8th Major Release of the Nuclear Reaction Data Library with CIELO-project Cross Sections, New Standards and Thermal Scattering Data". In: *Nuclear Data Sheets* 148 (2018), pp. 1–142 (cited on p. 7).

[8] K. Eguchi et al. "First Results from KamLAND: Evidence for Reactor Antineutrino Disappearance". In: *Physical Review Letters* 90.2 (Jan. 2003). Pubblicazione fondamentale della collaborazione KamLAND sull'oscillazione degli antineutrini da reattore., p. 021802. DOI: 10.1103/PhysRevLett.90.021802 (cited on p. 9).

[9] Anil K. Prinja and Edward W. Larsen. "General Principles of Neutron Transport". In: *Handbook of Nuclear Engineering*. Ed. by Dan Gabriel Cacuci. Springer, 2010, pp. 427–554 (cited on pp. 11, 12, 14, 16).

[10] Samuel Glasstone and Alexander Sesonske. *Nuclear Reactor Engineering: Reactor Design Basics*. 4th. New York, NY: Chapman and Hall, 1994 (cited on p. 25).

[11] Nestor A. Sepulveda, Jesse D. Jenkins, Fernando J. de Sisternes, and Richard K. Lester. "The role of firm low-carbon energy resources in deep decarbonization of power generation". In: *Joule* 2.11 (2018), pp. 2403–2420 (cited on p. 26).

[12] International Atomic Energy Agency. *Non-baseload Operation in Nuclear Power Plants: Load Following and Frequency Control*. Tech. rep. Vienna, Austria: IAEA Nuclear Energy Series No. NP-T-3.23, 2018 (cited on p. 26).

[13] Mark F. Ruth et al. *Flexible Nuclear Energy for Clean Energy Systems*. Tech. rep. NREL/TP-6A20-77088. Golden, CO: National Renewable Energy Laboratory (NREL), 2020 (cited on p. 26).

[14] Generation IV International Forum. *A Technology Roadmap for Generation IV Nuclear Energy Systems*. Tech. rep. GIF-002-00. The seminal document defining the goals of Gen-IV reactors. U.S. DOE Nuclear Energy Research Advisory Committee and the Generation IV International Forum, 2002 (cited on p. 28).

[15] Alan E. Waltar, Donald R. Todd, and Pavel V. Tsvetkov. *Fast Spectrum Reactors*. Provides the physical basis for the transition from thermal to fast neutron spectra. New York: Springer Science & Business Media, 2011 (cited on pp. 28, 29).

[16] Alessandro Alemberti, Valery Smirnov, Craig F. Smith, and Luciano Cinotti. "The Lead-cooled Fast Reactor: Strategy, activities and prospects". In: *Nuclear Engineering and Design* 272 (2014). Fonte fondamentale per i vantaggi del piombo (ebollizione, pressione atmosferica e sicurezza passiva)., pp. 156–170. DOI: 10.1016/j.nucengdes.2013.12.028 (cited on p. 29).

[17] Bahman Zohuri. *Nuclear Energy for Hydrogen Generation through Intermediate Heat Exchangers (IHX)*. Ottimo per i dati termodinamici e il confronto con i reattori ad acqua (PWR). Cham: Springer, 2015 (cited on p. 29).

[18] IAEA. *Status of Lead-cooled Fast Reactor Designs*. Tech. rep. IAEA-TECDOC-1626. Documento tecnico ufficiale che valida le caratteristiche di sicurezza passiva e inerzia termica. Vienna: International Atomic Energy Agency, 2013 (cited on p. 29).

[19] Luciano Cinotti et al. "The Lead-cooled Fast Reactor Project". In: *Progress in Nuclear Energy* 48.6 (2006). Descrive specificamente la trasparenza neutronica del piombo per i sistemi Gen-IV., pp. 483–490. DOI: 10.1016/j.pnucene.2006.06.002 (cited on p. 29).

[20] Francesco Lodi, Giacomo Grasso, et al. *Development of Lead-cooled Fast Reactor technology and dual-phase lead-bismuth eutectic systems*. Tech. rep. RT/2013/18/ENEA. Ottima fonte italiana (ENEA) sulla fisica del trasporto neutronico in piombo. ENEA, 2013 (cited on p. 29).

[21] International Atomic Energy Agency (IAEA). *Advances in Small Modular Reactor Technology Developments*. Tech. rep. A Supplement to the IAEA Advanced Reactors Information System (ARIS). Comprehensive overview of the global status of SMR designs and technologies. Vienna, Austria: IAEA, 2024. URL: https://aris.iaea.org/ (cited on p. 30).

[22] Giorgio Locatelli, Chris Bingham, and Mauro Mancini. "Small modular reactors: A challenging opportunity for the industrial world". In: *Nuclear Engineering and Design*

280 (2014). Analisi economica e industriale sui paradigmi SMR (anche italiani)., pp. 595–603. DOI: 10.1016/j.nucengdes.2014.09.025 (cited on p. 30).

[23] International Atomic Energy Agency (IAEA). *Advances in Small Modular Reactor Technology Developments*. Tech. rep. A Supplement to the IAEA Advanced Reactors Information System (ARIS), 2020 Edition. Vienna, Austria: IAEA, 2020. URL: https://aris.iaea.org/Publications/SMR_Book_2020.pdf (cited on p. 30).

[24] World Nuclear Association (WNA). *Generation IV Nuclear Reactors*. https://world-nuclear.org/information-library/nuclear-power-reactors/other/generation-iv-nuclear-reactors. Scheda tecnica ufficiale e panoramica internazionale sui sei design approvati dal Generation IV International Forum. 2022 (cited on p. 30).

[25] Alain Hébert. *Applied Reactor Physics*. Testo di riferimento per il calcolo reticolare (lattice physics) e i limiti delle tecniche di omogeneizzazione spaziale. Montréal: Presses inter Polytechnique, 2009 (cited on p. 31).

[26] Weston M. Stacey. *Nuclear Reactor Physics*. 3rd. Riferimento aggiornato per la fisica dei reattori, inclusi SMR e reattori veloci. Weinheim, Germany: John Wiley & Sons, 2018. ISBN: 978-3-527-41366-9 (cited on p. 31).

[27] IPCC. *Climate Change 2022: Mitigation of Climate Change. Contribution of Working Group III to the Sixth Assessment Report*. Tech. rep. Include l'energia nucleare nei principali percorsi (pathways) per limitare il riscaldamento globale a 1.5°C. Intergovernmental Panel on Climate Change, 2022 (cited on p. 32).

[28] United Nations Framework Convention on Climate Change. *Declaration to Triple Nuclear Energy*. COP28 Climate Summit, Dubai. Impegno internazionale per triplicare la capacità nucleare globale come strategia fondamentale per il Net Zero entro il 2050. 2023 (cited on p. 32).

[29] International Energy Agency (IEA). *Nuclear Power and Secure Energy Transitions: From Today's Challenges to Tomorrow's Clean Energy Systems*. Tech. rep. Analisi sul ruolo vitale del nucleare e degli SMR per garantire flessibilità e stabilità in una rete dominata dalle rinnovabili intermittenti. Paris: IEA, 2022 (cited on p. 32).

[30] newcleo. *newcleo: Innovator in the field of nuclear energy*. https://www.newcleo.com/. Sito ufficiale dell'azienda, consultato per la roadmap industriale e la strategia sul ciclo chiuso del combustibile. 2026 (cited on pp. 33, 34).

[31] newcleo. *newcleo's Lead-cooled Fast Reactors for clean, safe and sustainable energy*. Tech. rep. Presentazione tecnica riportante i dati di progetto del reattore LFR-AS-200 (pressione atmosferica, temperature e combustibile MOX). Università di Bologna (UNIBO) – Energy Talks, 2023. URL: https://corsi.unibo.it/magistrale/IngegneriaEnergetica/energy-talks-ciclo-di-seminari-tematici-con-

il-coinvolgimento-di-esperti-del-settore-energetico/presentazione-newcleo.pdf (cited on pp. 33, 34).

[32] Generation IV International Forum (GIF). *Lead Fast Reactors (LFR)*. Tech. rep. Specifiche tecniche generali sui reattori veloci al piombo, validanti l'uso del piombo a pressione atmosferica. 2024. URL: https://www.gen-4.org/generation-iv-criteria-and-technologies/lead-fast-reactors-lfr (cited on p. 33).

[33] E. E. Lewis and W. F. Miller. *Computational Methods of Neutron Transport*. New York: John Wiley & Sons, 1984 (cited on pp. 38, 40).

[34] George I. Bell and Samuel Glasstone. *Nuclear Reactor Theory*. New York: Van Nostrand Reinhold, 1970 (cited on pp. 38, 40).

[35] Richard E. Bellman. *Dynamic Programming*. Foundational text introducing the concept of the curse of dimensionality. Princeton, NJ: Princeton University Press, 1957 (cited on p. 38).

[36] Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo Methods*. 2nd. Provides a rigorous mathematical comparison between deterministic quadrature scaling and stochastic convergence. Weinheim: Wiley-VCH, 2008 (cited on p. 38).

[37] William Feller. *An Introduction to Probability Theory and Its Applications, Volume 1*. 3rd. Provides the foundational mathematical proofs for the Law of Large Numbers and the Central Limit Theorem. New York: John Wiley & Sons, 1968 (cited on p. 39).

[38] Ivan Lux and Laszlo Koblinger. *Monte Carlo Particle Transport Methods: Neutron and Photon Calculations*. Boca Raton, FL: CRC Press, 1991. ISBN: 978-0849360749 (cited on pp. 39, 40).

[39] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. 3rd. Details the asymptotic distribution of sample means and the evaluation of statistical error in Monte Carlo simulations. Hoboken, NJ: John Wiley & Sons, 2016 (cited on pp. 39, 41).

[40] Paul K Romano, Nicholas E Horelik, Bryan R Herman, Adam G Nelson, Benoit Forget, and Kord Smith. "OpenMC: A state-of-the-art Monte Carlo code for research and development". In: *Annals of Nuclear Energy* 82 (2015), pp. 90–97. DOI: 10.1016/j.anucene.2014.07.048 (cited on pp. 42, 74).

[41] OpenMC Development Team. *OpenMC Documentation: Theory and Methodology - Eigenvalue Calculations*. Accessed: March 4, 2026. 2024. URL: https://docs.openmc.org/en/stable/methods/eigenvalue.html (cited on p. 42).

[42] OpenMC Development Team. *OpenMC Documentation: Users Guide - Execution Settings*. Accessed: March 4, 2026. 2024. URL: https://docs.openmc.org/en/stable/usersguide/settings.html (cited on p. 42).

[43] Forrest B. Brown. *Fundamentals of Monte Carlo Particle Transport*. Tech. rep. LA-UR-05-4983. Los Alamos National Laboratory, 2015 (cited on pp. 45, 48).

[44] A. R. Siegel et al. "Analysis of communication costs for data movement in Monte Carlo nuclear reactor analysis". In: *Journal of Parallel and Distributed Computing* 74 (2014) (cited on p. 48).

[45] Steven P Hamilton and Thomas M Evans. "Continuous-energy Monte Carlo neutron transport on GPUs in the Shift code". In: *Annals of Nuclear Energy* 128 (2019). Discusses the severe limitations of history-based tracking on SIMT architectures due to thread divergence., pp. 236–247 (cited on p. 49).

[46] Jeffrey S. Vetter and Sparsh Mittal. "Contemporary computing platforms: Fish or fowl?" In: *Computer* 46.11 (2013), pp. 54–62 (cited on p. 52).

[47] Steven Laney, Chris Leger, et al. "High-throughput Monte Carlo transport on many-core processors". In: *Transactions of the American Nuclear Society* 111 (2014), pp. 385–388 (cited on p. 52).

[48] Nikolaos Alachiotis and Alexandros Stamatakis. "FPGA-based acceleration of scientific computing applications". In: *High-Performance Computing on FPGAs* (2013), pp. 1–20 (cited on p. 52).

[49] Roger Woods, John McAllister, Gaye Lightbody, and Ying Yi. *FPGA-based Implementation of Signal Processing Systems*. Chichester, UK: John Wiley & Sons, 2008 (cited on pp. 52, 64).

[50] Stephen Brown and Zvonko Vranesic. *Fundamentals of Digital Logic with Verilog Design*. New York, NY, USA: McGraw-Hill, 2002 (cited on p. 52).

[51] Clive Maxfield. *The Design Warrior's Guide to FPGAs: Devices, Tools and Flows*. Oxford, UK: Newnes, 2004 (cited on p. 52).

[52] Dov Frohman-Bentchkowsky. "A fully decoded 2048-bit electrically programmable FAMOS read-only memory". In: *IEEE Journal of Solid-State Circuits* 6.5 (1971), pp. 301–306 (cited on p. 52).

[53] William Carter, K. Duong, Ross H. Freeman, H.-C. Hsieh, J. Ja, J. Mahoney, P. Ngo, and S. Sze. "A user programmable reconfigurable logic array". In: *Proceedings of the IEEE 1986 Custom Integrated Circuits Conference*. IEEE. 1986, pp. 233–235 (cited on p. 52).

[54] Stephen D. Brown, Robert J. Francis, Jonathan Rose, and Zvonko G. Vranesic. *Field-Programmable Gate Arrays*. Boston, MA, USA: Springer, 1992 (cited on p. 52).

[55] Adrian Thompson. "An evolved circuit, intrinsic in silicon, entwined with physics". In: *International Conference on Evolvable Systems*. Springer. 1996, pp. 390–405 (cited on p. 53).

[56] Ian Kuon, Russell Tessier, and Jonathan Rose. "FPGA Architecture: Survey and Challenges". In: *Foundations and Trends in Electronic Design Automation* 2.2 (2008), pp. 135–253 (cited on pp. 53, 54).

[57] Scott Hauck and André DeHon. *Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation*. Burlington, MA: Morgan Kaufmann, 2007 (cited on p. 53).

[58] André DeHon. "Spatial computing". In: *Proceedings of the 2002 Workshop on Non-Silicon Computation (NSC)*. Cambridge, MA, 2002 (cited on pp. 53, 54).

[59] Christophe Bobda. *Introduction to Reconfigurable Computing: Architectures, Algorithms, and Applications*. Dordrecht, Netherlands: Springer, 2007 (cited on p. 53).

[60] Uwe Meyer-Baese. *Digital Signal Processing with Field Programmable Gate Arrays*. 4th. Berlin, Heidelberg: Springer, 2014 (cited on p. 64).

[61] Behrooz Parhami. *Computer Arithmetic: Algorithms and Hardware Designs*. 2nd. New York, NY: Oxford University Press, 2010 (cited on p. 64).

[62] *Intel 64 and IA-32 Architectures Software Developer's Manual, Combined Volumes: 1, 2A, 2B, 2C, 2D, 3A, 3B, 3C, 3D and 4*. Order Number: 325462-078US. Intel Corporation. 2023 (cited on p. 66).

[63] Jean-Michel Muller. *Elementary Functions: Algorithms and Implementation*. 3rd. Boston, MA: Birkhäuser, 2016. DOI: 10.1007/978-1-4899-7983-4 (cited on pp. 66, 67).

[64] Donald E. Knuth. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*. 3rd. Reading, MA: Addison-Wesley Professional, 1997 (cited on p. 66).

[65] J. S. Walther. "A unified algorithm for elementary functions". In: *Proceedings of the Spring Joint Computer Conference* (1971), pp. 379–385. DOI: 10.1145/1478786.1478840 (cited on p. 67).

[66] Daniel R. Llamocca and Carola P. Agurto. "A fixed-point implementation of the expanded hyperbolic CORDIC algorithm". In: *Latin American Applied Research* 37.1 (2007), pp. 83–91 (cited on p. 68).

[67] Donald E. Knuth. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. 3rd. Reading, MA: Addison-Wesley Professional, 1997 (cited on p. 74).

[68] John R. Tramm, Andrew R. Siegel, et al. “Performance Portable Monte Carlo Particle Transport on Intel, NVIDIA, and AMD GPUs”. In: *Nuclear Science and Engineering* (2022) (cited on pp. 88, 89).

[69] John R. Tramm et al. *Optimizing OpenMC performance for exascale*. Tech. rep. Available online. Argonne Leadership Computing Facility (ALCF), 2023 (cited on pp. 88, 89).

[70] Nils Voss et al. “Demonstration of FPGA Acceleration of Monte Carlo Simulation”. In: *20th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT 2021)*. Studio specifico che correla i cicli di clock FPGA al numero di eventi per calcolare il rateo di completamento delle storie Monte Carlo. CERN. 2021 (cited on p. 88).