
Asynchronous Circuits and Dataflow Architectures: A Survey

www.surveyx.cn

Abstract

Asynchronous circuits represent a pivotal shift in digital design, offering distinct advantages over traditional synchronous systems by operating without a global clock signal. This survey delves into the multifaceted aspects of asynchronous circuits and dataflow architectures, underscoring their transformative potential in modern computing. Key benefits include enhanced power efficiency, adaptability, and robustness, making them ideal for energy-constrained applications and systems requiring high concurrency. The survey articulates the principles and methodologies of asynchronous design, focusing on handshake protocols, bundle data, and the integration of Click elements to facilitate efficient communication. It highlights the advantages of dataflow architectures in optimizing resource utilization and supporting asynchronous communication, aligning with the principles of asynchronous circuits. Performance analyses demonstrate substantial improvements in latency and energy consumption, with case studies illustrating applications in digital signal processing, secure systems, and neuromorphic computing. Challenges such as verification complexity and design integration are acknowledged, alongside emerging trends that promise to drive further innovation. Future research directions include optimizing design methodologies and expanding theoretical frameworks to enhance the understanding and application of asynchronous systems. By addressing these challenges and leveraging the unique properties of asynchronous circuits, there is significant potential to revolutionize digital circuit design and advance future developments in this field.

1 Introduction

1.1 Significance of Asynchronous Circuits

Asynchronous circuits are integral to contemporary digital design, providing significant advantages over synchronous systems by eliminating the need for a global clock signal. This characteristic reduces power consumption and mitigates issues like clock skew and jitter, which are prevalent in synchronous designs [1]. Such benefits are particularly crucial in very large-scale integration (VLSI) applications, where they simplify clock network complexity and enhance tolerance to variations in process, voltage, and temperature.

In digital signal processing (DSP) systems, asynchronous methodologies are increasingly recognized for their energy efficiency, especially in energy-constrained applications [2]. These circuits are robust against variations in processing speeds and environmental conditions, making them suitable for diverse applications, including fault-tolerant systems and neuromorphic computing, where asynchronous events facilitate computation and communication [3].

Asynchronous circuits also enhance security by addressing vulnerabilities such as side-channel attacks (SCAs), which are more likely in synchronous designs. Quasi Delay Insensitive (QDI) asynchronous circuits specifically target vulnerabilities like Differential Power Analysis (DPA) attacks, underscoring their importance in modern digital design [4].

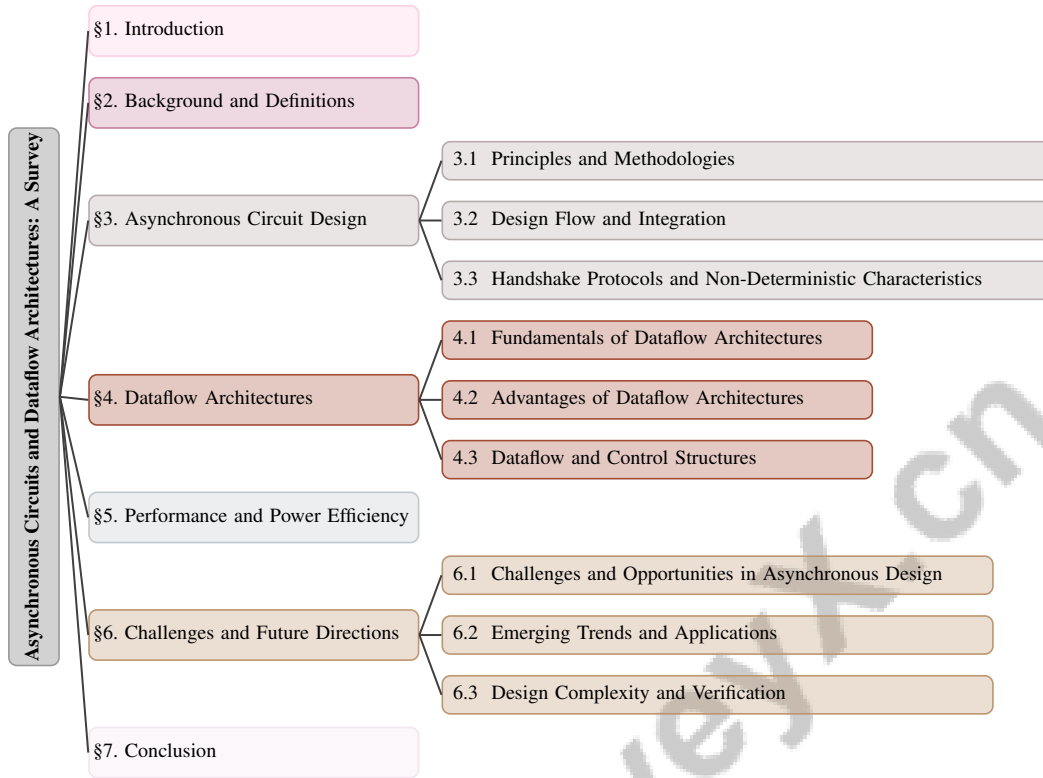


Figure 1: chapter structure

Advanced theoretical frameworks, including studies of regular asynchronous systems and non-anticipation properties, deepen our understanding of asynchronous circuit behavior, enhancing their applicability in digital electrical engineering [3]. These mathematical models support the design of circuits that are efficient, reliable, and secure.

The relevance of asynchronous circuits is underscored by their ability to enhance power efficiency, improve reliability, and address the complexities of VLSI systems. Their operation without a global clock alleviates issues such as clock skew and reduces power consumption, which is critical for miniaturization and the growing demand for mobile and IoT devices. Furthermore, advancements in asynchronous design methodologies, including robust synthesis techniques and delay-insensitive codes, have emerged to tackle challenges like hazards and glitches, establishing them as a promising alternative to traditional synchronous designs [5, 6, 7, 8]. The scope of their application continues to expand as researchers explore new methodologies and frameworks.

1.2 Motivation for Asynchronous Design

The motivation to adopt asynchronous design methodologies arises from the need to overcome the limitations of traditional synchronous circuits, particularly in environments with non-persistent requests and dynamic conditions that can induce hazards and unreliable operations [9]. Asynchronous circuits, operating without a global clock, mitigate issues such as clock skew and jitter, which are common in synchronous systems. This flexibility is vital in multi-core architectures, where effective asynchronous communication and routing are essential for scalability and performance [10].

Energy efficiency is another significant driver for asynchronous design, especially in applications like personalized healthcare and IoT devices, where minimizing power consumption is critical. Asynchronous circuits inherently reduce electromagnetic noise, which is advantageous in sensitive electronic environments [11]. The complexity of synchronous circuit design, reliant on precise timing and synchronization, further emphasizes the appeal of asynchronous methodologies that simplify the design process and minimize potential errors [12].

Additionally, the necessity for innovative methodologies that enhance performance and reduce pipeline depth addresses the limitations of existing CMOS-based design algorithms [13]. The inadequacy of current hardware description languages to effectively support modern hardware design needs, particularly for asynchronous circuits, highlights the need for novel approaches and complex verification processes [14].

The development of formal mathematical frameworks for understanding asynchronous systems is also a significant motivator, as existing models often fail to capture the complexity of asynchronous behavior. This has led to research into innovative modeling approaches, drawing from switching theory and other foundational concepts. The challenge of constructing comprehensive theories of asynchronous systems drives the exploration of formalized models and tools for investigation [15].

Moreover, asynchronous circuits offer enhanced resilience against physical attacks due to their independence from a clock signal [4]. The need to generalize parameter instantiations in timed automata to ensure correct behavior across a range of parameter valuations also propels the adoption of asynchronous methodologies [16]. The limitations of current methodologies in eliminating all sources of leakage in QDI circuits necessitate a new design flow to minimize vulnerabilities [17]. Collectively, these factors underscore the compelling reasons for adopting asynchronous design methodologies in modern digital systems.

1.3 Structure of the Survey

This survey is structured to comprehensively explore asynchronous circuits and dataflow architectures, emphasizing their significance and potential impact on modern digital design. The introduction establishes the foundational importance of asynchronous circuits, highlighting benefits such as power efficiency and enhanced reliability across various applications. The discussion on adopting asynchronous design methodologies addresses the limitations of synchronous circuits, particularly issues related to clock distribution and power consumption. Asynchronous circuits, operating without a global clock, promise enhanced energy efficiency and facilitate simplified design processes through relaxed timing constraints and modular designs. This shift is particularly relevant given the increasing demand for lower power requirements and the ability to manage process variations. The development of robust design tools, such as Balsa, is facilitating broader implementation of asynchronous systems, allowing designers to leverage their advantages in performance and reliability [8, 7, 11, 18].

The next section, "Background and Definitions," provides a detailed overview of core concepts essential for understanding asynchronous circuits, including definitions of key terms such as bundle data, Click, handshake protocols, and dataflow architectures. This section lays the groundwork for subsequent discussions by elucidating the fundamental principles underlying asynchronous systems.

In "Asynchronous Circuit Design," the survey delves into foundational principles and methodologies underpinning the design of asynchronous circuits, emphasizing delay theory, logical function modeling, and hazard implications on circuit performance. It discusses the advantages of asynchronous systems over synchronized circuits, including reduced power consumption and improved performance while addressing challenges such as potential errors and complexities in logic synthesis [19, 5, 6, 7]. The role of handshake protocols and bundle data in facilitating efficient communication and data transfer, along with the design flow process and integration of Click elements, is also examined.

The section on "Dataflow Architectures" investigates the structure and operation of dataflow systems employing asynchronous circuits, discussing data-driven operations and the advantages these architectures offer in computing systems, supported by examples of applications where they are particularly beneficial.

"Performance and Power Efficiency" analyzes the enhancements and power savings achievable through asynchronous circuits compared to traditional synchronous designs, incorporating case studies and examples from existing literature to substantiate the analysis.

The penultimate section, "Challenges and Future Directions," identifies design and implementation challenges associated with asynchronous circuits and dataflow architectures. It explores potential solutions and areas for future research, highlighting emerging trends and technologies that may influence the development of asynchronous systems.

The "Conclusion" section summarizes the survey's key insights, emphasizing the critical role of asynchronous circuits and dataflow architectures in enhancing digital design. It underscores the

importance of robust synthesis techniques, such as those utilizing delay-insensitive codes and the 4-phase return-to-zero handshake protocol, essential for improving the reliability and efficiency of asynchronous systems. Additionally, it addresses existing challenges in the field, including limitations of current methodologies and the need for more general logic synthesis algorithms, thereby highlighting ongoing advancements and research opportunities within this domain [19, 5]. The conclusion suggests avenues for future research and development, aiming to inspire continued exploration and innovation in this field. The following sections are organized as shown in Figure 1.

2 Background and Definitions

2.1 Core Concepts in Asynchronous Circuits

Asynchronous circuits, characterized by their event-driven processing without a global clock, offer significant energy efficiency and power reduction benefits, particularly in low-power applications like Quasi Delay Insensitive (QDI) asynchronous adders, which excel in latency, area, and power dissipation across various handshake protocols [4, 1]. This clockless operation, however, introduces non-determinism, complicating system behavior prediction and posing challenges for modeling and analysis due to the absence of a formal mathematical framework for asynchronous automata [20, 21]. Ensuring deadlock freedom, an inherent property in asynchronous systems, adds further complexity and is difficult to verify [22].

The operation of asynchronous circuits relies on precise delay modeling—whether bounded, unbounded, fixed, or inertial—to ensure functionality and stability, involving the determination of steady states for signals over time [20]. Security in asynchronous design is enhanced by the absence of a global clock, reducing vulnerabilities such as Differential Power Analysis (DPA) attacks [4]. Classifying sequential circuits based on causal function characteristics offers insights into their behavior, informing security strategies and design approaches [2].

Understanding these core principles is crucial for advancing Network Function Virtualization (NFV) services, highlighting the potential for innovative applications across various domains [23].

2.2 Communication Protocols and Data Handling

In asynchronous systems, efficient data transfer without a global clock is achieved through communication protocols and data handling mechanisms. Handshake protocols enable synchronization between circuit components using request and acknowledgment signals, providing flexible communication pathways that address timing issues like clock skew and jitter [18]. The Quasi Delay Insensitive (QDI) template, notable for its robustness, employs a four-phase handshake protocol comprising request, acknowledgment, data transfer, and reset phases, ensuring reliable data transfer and system recovery from transient faults [18].

Data handling in asynchronous systems utilizes bundle data, grouping multiple data signals under a single control signal. This approach simplifies communication interface design and aligns with the asynchronous philosophy by decoupling data transfer from timing constraints, offering greater processing and routing flexibility [18].

Advanced simulation tools like SPICE and VHDL-AMS are essential for evaluating asynchronous communication protocols' performance and power characteristics. These tools allow for detailed analysis and optimization of QDI templates, providing insights into trade-offs among speed, area, and energy efficiency [18]. The simulation data guides the design of asynchronous circuits to meet specific performance and power requirements, fostering the development of more efficient and scalable communication systems.

In examining the complexities of asynchronous circuit design, it is essential to understand the underlying principles and methodologies that govern this field. As illustrated in Figure 2, the hierarchical structure of asynchronous circuit design is depicted, emphasizing critical components such as the design flow and integration processes. This figure also highlights the significant role of handshake protocols in effectively managing the non-deterministic characteristics inherent in asynchronous systems. By visualizing these elements, we can better appreciate the intricacies involved in designing robust asynchronous circuits.

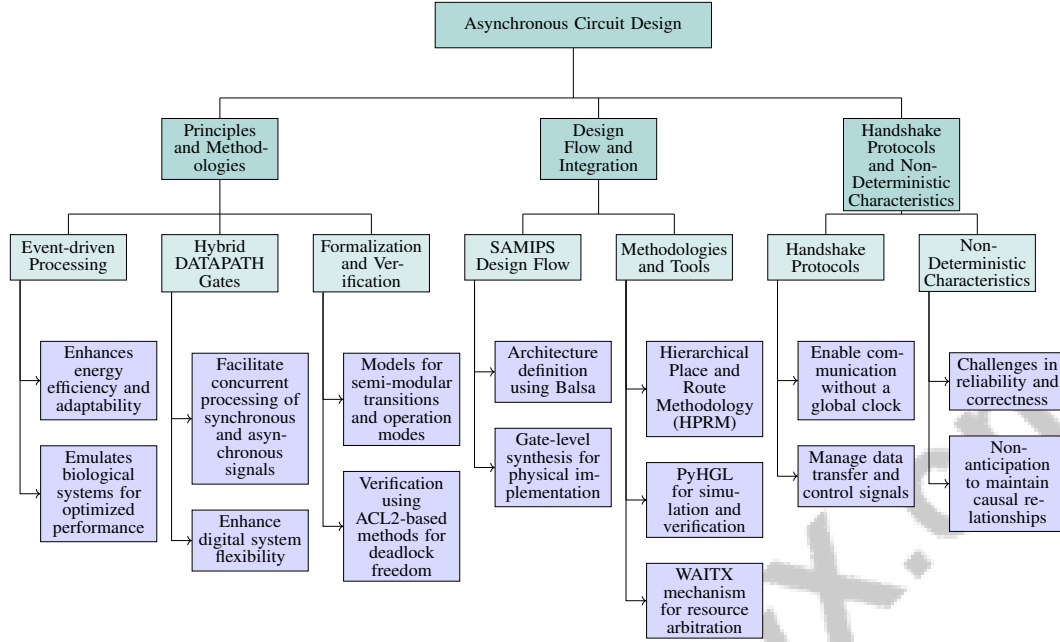


Figure 2: This figure illustrates the hierarchical structure of asynchronous circuit design, highlighting key principles and methodologies, design flow and integration processes, and the role of handshake protocols in managing non-deterministic characteristics.

3 Asynchronous Circuit Design

3.1 Principles and Methodologies

Asynchronous circuit design eschews global clock reliance, promoting event-driven processing that enhances energy efficiency and adaptability, particularly in dynamic applications like the Event-driven Saliency-based Selective Attention Model (ESSAM) for visual data processing [24]. By emulating biological systems, these circuits dynamically manage attention and resources, optimizing performance. A significant advancement is the use of hybrid DATAPATH gates, which facilitate concurrent processing of synchronous and asynchronous signals, enhancing digital system flexibility [25]. The RS-Buffer in Dual-Rail Logic stabilizes states by ensuring single-signal changes during transitions, thus preventing errors and improving reliability [7].

Formalizing asynchronous circuit behavior is crucial for understanding system dynamics. Vlad et al. introduced models that formalize governing equations, incorporating semi-modular transitions and fundamental operation modes to clarify semantics [21]. These models are vital for accurately modeling input-output relationships and ensuring deterministic behavior without a global clock. Verification methods, such as ACL2-based verification for Click primitives, ensure robust designs by formalizing behavior and capturing interactions to verify deadlock freedom [22]. Stability frameworks, based on mathematical definitions, provide insights into system stability, essential for robust operations [20].

Asynchronous routing circuits integrated with heterogeneous memory structures offer scalable solutions for neuromorphic processors, enabling efficient data handling [26]. This scalability is crucial for developing systems that manage large data volumes efficiently. The core principle of asynchronous design is to simplify circuits by eliminating complex clock networks, thereby enhancing performance and reducing design complexity [13].

As illustrated in Figure 4, the hierarchical structure of asynchronous circuit design encompasses design principles, modeling and verification, and applications and scalability, highlighting key methodologies and innovations. This figure presents a viable alternative to traditional synchronous systems by eliminating the need for a global clock, potentially enhancing performance and power efficiency. The visual representations highlight key principles and methodologies in asynchronous circuit

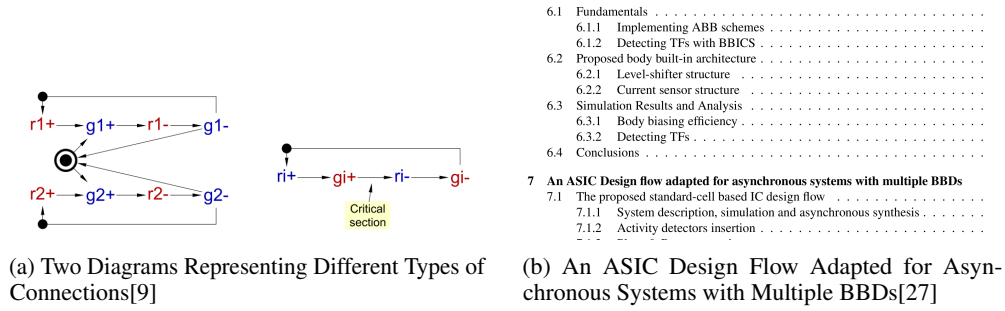


Figure 3: Examples of Principles and Methodologies

design. The first diagram showcases different types of connections, emphasizing the complexity and interconnectivity of asynchronous systems. The second diagram outlines an ASIC design flow tailored for asynchronous systems, detailing various aspects of the design process, such as implementing ABB schemes, and emphasizing the structured approach necessary for adapting ASIC design flows to accommodate the unique characteristics of asynchronous systems.

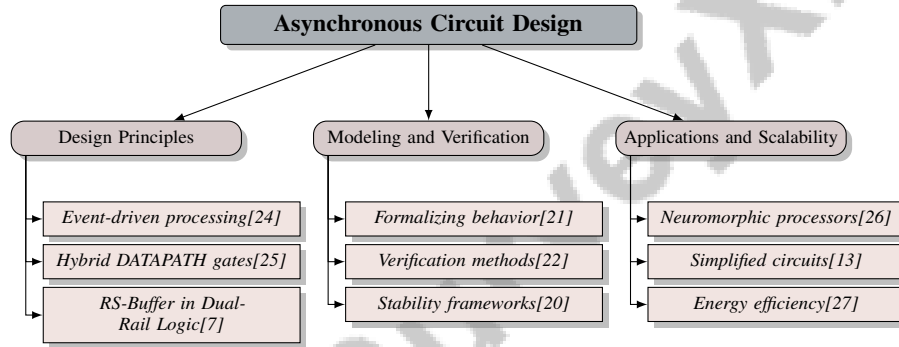


Figure 4: This figure displays the hierarchical structure of asynchronous circuit design, encompassing design principles, modeling and verification, and applications and scalability, highlighting key methodologies and innovations.

3.2 Design Flow and Integration

Method Name	Design Methodologies	Integration Strategies	Performance Optimization
Yak[28]	Yak Language	Token Flow Graphs	Graph Optimizations
NCL-DATAPATH[25]	Balsa	Datapath Gates	Memory Optimized Routing
HPRM[17]	Hierarchical Place And Route	Datapath Gates	Memory Optimized Routing
HGL[14]	Python-based Framework	Event-driven Simulator	Dynamic Typing
WAITX[9]	Workcraft Framework	Datapath Gates	Memory Optimized Routing
MOR[26]	Memory Optimized Routing	Asynchronous Routing Circuits	Hierarchical Place Methodology

Table 1: Comparison of various asynchronous circuit design methodologies, integration strategies, and performance optimization techniques. The table summarizes the key features of each method, highlighting their unique approaches to design, integration, and optimization within asynchronous systems.

Asynchronous circuit design flow encompasses several stages for seamless integration into larger systems. Table 1 presents a comparative analysis of different methodologies employed in asynchronous circuit design, detailing their respective design methodologies, integration strategies, and performance optimization techniques. The SAMIPS design flow begins with architecture definition using Balsa, which specifies asynchronous control and data paths, culminating in gate-level synthesis for physical implementation [8]. Token flow graphs, as implemented in Yak, are critical for managing data and control signal flow, ensuring efficient communication [28]. Integration is further enhanced by DATAPATH gates, converting critical synchronous paths to asynchronous operation without altering overall design [25].

Methodologies like the Hierarchical Place and Route Methodology (HPRM) organize circuit components to ensure balanced paths, mitigating vulnerabilities to Differential Power Analysis (DPA) attacks [17]. Tools like PyHGL streamline the design process by integrating simulation and verification within a single framework [14]. In resource arbitration, the WAITX mechanism isolates arbitration from unpredictable signals, crucial for maintaining system integrity [9]. The Memory Optimized Routing (MOR) scheme optimizes memory usage while maximizing network programmability through a mixed tag-based method [26].

Design flow may involve identifying reference parameter valuations and partitioning the parametric space with behavioral cartography algorithms, essential for exploring the design space and ensuring performance and reliability [16]. These methodologies and tools highlight the sophistication and adaptability of asynchronous circuit design flow, enabling effective integration across applications.

3.3 Handshake Protocols and Non-Deterministic Characteristics

Handshake protocols are fundamental to asynchronous circuits, enabling communication without a global clock. The modified 4-phase handshake protocol effectively controls data flow in systems like FIR filter designs without clocked registers [11]. These protocols manage data transfer and control signals, minimizing latency and enhancing reliability. Non-deterministic characteristics pose challenges in reliability and correctness, stemming from potential multiple outputs from a single input under varying conditions [29]. Analysis often involves examining fixed points and bifurcations, providing insights into dynamic behavior [30].

Addressing non-deterministic behavior is crucial, with non-anticipation ensuring outputs do not anticipate future inputs, maintaining causal relationships [15]. System stability, explored through various theorems, emphasizes the importance of subsystem stability for overall system stability [20]. Asynchronous gates with self-resetting features mitigate non-deterministic behavior by addressing flux trapping in RSFQ circuits [13]. However, defining serial connections that preserve system regularity remains challenging [31].

The non-uniqueness of generator and computation functions complicates analysis and understanding, necessitating sophisticated approaches for design and verification [29]. These considerations underscore the critical role of handshake protocols in managing non-deterministic behavior and ensuring the robustness and reliability of asynchronous circuits.

4 Dataflow Architectures

Dataflow architectures redefine computational models, particularly in network function virtualization (NFV) services and asynchronous circuits, by improving communication and routing efficiency. Frameworks like RDCL 3D demonstrate this integration, while tools such as Yak automate asynchronous circuit design via timing constraint generation. Advances in core interface optimization for multi-core neuromorphic processors illustrate dataflow strategies' role in enhancing performance and resource efficiency, underscoring their significance in modern computational design [5, 28, 23, 10]. These architectures prioritize data availability over traditional control-flow paradigms, enhancing processing efficiency and enabling greater parallelism and adaptability across applications.

4.1 Fundamentals of Dataflow Architectures

Dataflow architectures shift from traditional control-flow computing by emphasizing data flow as the primary driver of computation. Execution is dictated by input data availability rather than a sequential program counter, facilitating natural parallelism and efficient resource utilization [2]. This approach aligns with asynchronous circuit design, where operations are triggered by data availability, minimizing global synchronization needs and enhancing flexibility [3].

The architecture consists of processing nodes linked by communication channels transporting data tokens. Each node, or actor, executes computations when all required inputs are available, producing output data tokens that propagate through the network [15]. This model supports dynamic execution patterns and adapts to varying workloads, making it suitable for applications with irregular data access or high concurrency requirements [1].

Token-based execution is crucial, where tokens at input ports signify readiness for operation. This mechanism supports fine-grained parallelism, allowing multiple nodes to operate concurrently without explicit synchronization [20]. Data-driven scheduling and dynamic resource allocation further optimize resource use and enhance system efficiency [26].

Dataflow architectures also support asynchronous communication, enabling data transfer between nodes without a global clock. This feature aligns with asynchronous circuit principles, fostering robust and scalable designs resilient to processing speed variations and environmental conditions [30]. Decoupling computation from timing constraints allows efficient handling of data dependencies while reducing overhead associated with traditional synchronization mechanisms [29].

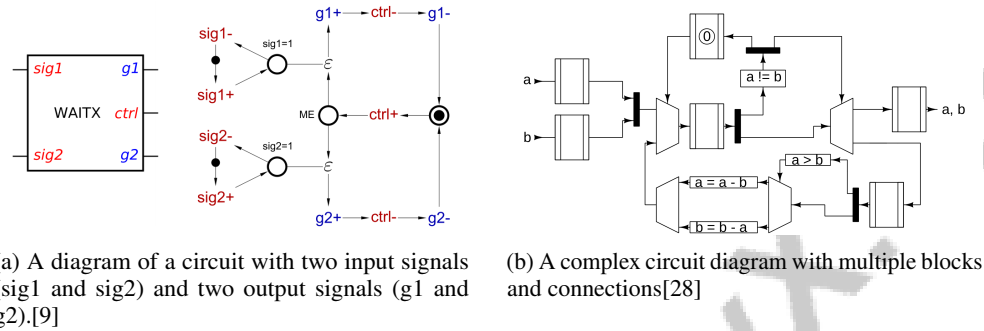


Figure 5: Examples of Fundamentals of Dataflow Architectures

As shown in Figure 5, understanding dataflow architectures is essential for grasping data processing and management in computational systems. The first diagram depicts a "WAITX" circuit with two input signals (sig1 and sig2) and two output signals (g1 and g2), illustrating basic dataflow operations. The second diagram presents a complex circuit with multiple interconnected blocks, demonstrating data and control flow within the system. These examples highlight the foundational concepts of dataflow architectures, paving the way for further exploration and innovation in the field [9, 28].

4.2 Advantages of Dataflow Architectures

Dataflow architectures offer several advantages that enhance computing system efficiency and performance. They exploit inherent parallelism, where computations driven by data availability allow simultaneous execution, significantly boosting throughput and resource utilization. This is crucial in asynchronous circuits, which must manage concurrent requests for optimal performance and reliability [19, 5, 9].

Decoupling computation from control flow enables dynamic adaptation to varying workloads and data access patterns, beneficial in applications with irregular dependencies or high concurrency needs, such as multimedia processing and scientific simulations [1]. By focusing on data-driven execution, these architectures simplify system designs by reducing complex control logic and synchronization mechanisms [20].

Their support for asynchronous communication enhances robustness and scalability, allowing operations to proceed independently of a global clock. This capability improves reliability and performance by accommodating variations in processing speeds and environmental conditions [30].

The token-based execution model optimizes resource allocation by enabling fine-grained parallelism, allowing multiple nodes to operate concurrently and maximizing hardware utilization [26]. Data-driven scheduling techniques dynamically allocate resources based on current demands, enhancing system efficiency and responsiveness [29].

4.3 Dataflow and Control Structures

Dataflow operations in asynchronous systems execute tasks based on data availability rather than a global clock, allowing a more natural expression of parallelism and adaptability [3]. The interaction between dataflow operations and control structures is essential for optimizing performance and ensuring reliable execution.

Control structures in dataflow architectures manage the flow of data tokens between processing nodes, synchronizing operations without clock-based timing. Implemented through handshake protocols, these structures enable nodes to communicate readiness and task completion, coordinating data-driven execution [29]. The absence of a global clock necessitates robust control mechanisms to handle processing speed variations and manage dependencies [20].

Integration of control structures often employs token-based execution models, where tokens indicate data availability for processing, supporting fine-grained parallelism by allowing multiple operations to proceed concurrently [15]. The dynamic token flow facilitates efficient resource utilization and minimizes idle times, enhancing overall system throughput and performance [26].

Advanced control techniques, such as data-driven scheduling and resource allocation, optimize dataflow operations by dynamically adjusting resource allocation based on current workload demands, ensuring effective utilization of processing nodes and minimizing bottlenecks [30]. By prioritizing operations with available data, these mechanisms improve the responsiveness and efficiency of asynchronous systems [1].

5 Performance and Power Efficiency

5.1 Performance Optimization Techniques

Optimizing performance in asynchronous circuits involves architectural innovations and theoretical advancements, with a focus on efficiency and robustness. Key contributions include the proof of -limit sets and invariance properties, enhancing the understanding of dynamic behavior and circuit performance [3]. Delay optimization in combinational logic offers provably optimal solutions, significantly improving timing over traditional methods, crucial for applications requiring precise timing [32].

Scalable and automated verification methods streamline complex asynchronous circuit verification, ensuring reliability without manual proofs [22]. This automation reduces design validation time and effort. User-centric design principles, such as those applied in RDCL 3D, benefit from user feedback and successful service deployment [23]. The classification of sequential circuits under a unified framework aids in understanding and analyzing asynchronous systems, enhancing performance [2].

Security considerations are increasingly integrated with performance optimization. Designing shields for detecting physical attacks exemplifies the dual focus on performance and security, ensuring robust circuit operation [4]. Asynchronous circuits, particularly in secure microcontroller units for IoT devices, offer clock-independent operation, reducing power consumption and improving electromagnetic compatibility. They also facilitate robust logic synthesis and enable scalable programmable metasurfaces, highlighting their versatility in security and advanced electromagnetic applications [6, 4, 19, 5, 33]. This comprehensive optimization framework underscores the benefits of asynchronous design methodologies in advancing digital technology.

5.2 Latency and Energy Consumption Improvements

Asynchronous circuit design significantly reduces latency and energy consumption by eliminating the global clock, which decreases power dissipation and enhances adaptability [1]. The Latency-Optimized Early Output Ripple Carry Adder (LEO-RCA) exemplifies this, reducing latency by up to 35.3% compared to traditional designs [34].

The approach in [11] achieves notable latency and energy efficiency reductions, suitable for ASIC and FPGA implementations. Quasi Delay Insensitive (QDI) adders following the Return-to-One (RTO) protocol typically exhibit lower latency and power dissipation than Return-to-Zero (RTZ) protocol-based adders [1].

In multi-core architectures, integrating asynchronous Content Addressable Memory (CAM) circuits with HAT arbitration architecture demonstrates substantial latency and energy consumption reductions, reinforcing asynchronous design benefits. The Asynchronous Time-Multiplexed (ATM) processor architecture notably lowers latency and energy consumption compared to synchronous implementations, addressing modern VLSI design challenges such as clock distribution complexity and power consumption. This architecture employs advanced data encoding and asynchronous design principles to enhance performance and efficiency [8, 34].

Quasi Delay Insensitive (QDI) methodologies significantly reduce Differential Power Analysis (DPA) sensitivity, achieving a balanced design that minimizes leakage and enhances power efficiency. Recent studies demonstrate significant latency and energy consumption reductions in asynchronous designs, establishing this approach as transformative in digital circuit design. Systems like the SAMIPS microprocessor and innovative asynchronous FIR filters alleviate power constraints and clock distribution challenges inherent in synchronous designs, enhancing performance and promoting energy efficiency in applications like wireless sensor networks and portable medical devices [5, 8, 11].

5.3 Applications and Case Studies

Asynchronous circuits find significant applications across various domains, enhancing system performance and efficiency. In Network Function Virtualization (NFV) services, asynchronous circuits improve scalability and flexibility, as demonstrated by the RDCL 3D model [23]. In digital signal processing, they offer substantial benefits in energy efficiency and reduced power consumption, particularly in energy-constrained systems. Implementing Quasi Delay Insensitive (QDI) asynchronous adders improves latency, area, and power dissipation, making them suitable for low power and high efficiency scenarios [1].

Asynchronous circuits also play a critical role in secure system design, mitigating vulnerabilities like Differential Power Analysis (DPA) attacks. Designing dedicated asynchronous circuits enhances security, especially in high-security environments [4]. In neuromorphic computing, asynchronous circuits efficiently mimic biological processes, offering scalable solutions for processing complex data patterns [26].

Case studies in multi-core architectures highlight asynchronous design advantages. Integrating asynchronous CAM circuits and HAT arbitration architecture in multi-core systems achieves significant latency reductions and energy savings, emphasizing asynchronous methodologies' performance benefits. The ATM processor architecture exemplifies these advantages, demonstrating notable latency and energy consumption reductions compared to synchronous implementations. This efficiency is crucial in modern electronics, where lower power consumption and higher performance are critical, especially in mobile, IoT, and edge devices. Asynchronous logic addresses common synchronous VLSI design challenges, such as clock skew and power consumption, while facilitating a modular design approach that simplifies technological upgrades. The ATM architecture showcases these benefits, illustrating asynchronous systems' potential to enhance overall computing efficiency [5, 8, 35].

Collaboration among asynchronous circuit design communities could optimize buffer sizing strategies, as suggested by recent surveys [35]. This collaborative approach may enhance asynchronous circuit performance, ensuring they remain at the forefront of digital circuit innovation.

These applications and case studies collectively underscore asynchronous circuits' transformative potential in advancing digital technology. Harnessing their distinctive operational characteristics, asynchronous circuits drive innovation across diverse domains, including secure microcontroller units for the Internet of Things (IoT), scalable neuromorphic computing systems utilizing event-driven neural network architectures for efficient real-time processing, and programmable metasurfaces that dynamically manipulate electromagnetic waves, thereby enhancing performance across various applications [4, 26, 5, 33, 10].

6 Challenges and Future Directions

6.1 Challenges and Opportunities in Asynchronous Design

Asynchronous circuit design faces significant challenges due to its non-deterministic nature and lack of a global clock, complicating system behavior modeling and prediction [36]. Integrating these circuits into large-scale systems, such as neuromorphic processors, increases the complexity of asynchronous event routing and memory demands [26]. The computational burden of exploring the entire parametric state space further complicates the design process [16]. Verification remains critical, with current methods necessitating monolithic approaches to ensure properties like deadlock freedom, often inadequately addressed by isolated checks [22]. Theoretical claims lack empirical validation, exacerbating these challenges [20]. This fragmentation parallels challenges in Network Function Virtualization (NFV) service design, highlighting the need for cohesive frameworks [23].

Despite these challenges, asynchronous design offers substantial innovation opportunities, overcoming synchronous circuits' limitations, such as clock skew and jitter. Its energy efficiency and adaptability make it ideal for energy-constrained environments and high-concurrency applications [1]. A unified framework for classifying sequential circuits enhances the understanding and analysis of asynchronous systems, contributing to performance improvements [2]. Advancements in Electronic Design and Automation (EDA) tools are crucial for managing asynchronous systems' complexity and integrating them into contemporary digital workflows. However, aligning asynchronous design methodologies with industry-standard IC design tools and modeling languages remains challenging [11]. The historical neglect of asynchronous systems in mathematical literature indicates a pressing need to expand the theoretical framework to encompass more complex dynamics [3]. Addressing existing limitations and leveraging asynchronous design's unique advantages could revolutionize digital circuit design and propel future developments in this domain.

6.2 Emerging Trends and Applications

Asynchronous circuits are increasingly prominent in digital design, with emerging trends and applications highlighting their transformative potential across various fields. A notable trend is the shift from mathematical frameworks to practical engineering applications, broadening asynchronous systems' exploration in real-world contexts [3]. This transition is evident in model-agnostic frameworks in NFV, paralleling asynchronous circuits' evolution towards more adaptable and scalable designs [23].

Digital signal processing benefits significantly from asynchronous design advancements, exemplified by the Latency-Optimized Early Output Ripple Carry Adder (LEO-RCA), achieving substantial latency and energy consumption reductions [34]. Such improvements are crucial for DSP applications, where performance and efficiency are paramount. The exploration of event-driven architectures for real-time visual processing, particularly in FPGA implementations, underscores the growing interest in asynchronous systems for managing complex sensory data [24].

In multi-core architectures, innovations in current sensing circuits and adaptive body biasing techniques promise reduced latency and power consumption, enhancing overall system performance [26]. Asynchronous circuits' integration is pivotal for scalable systems, where efficient data routing and memory management are critical. Research into refining FD-SOI technology presents opportunities for further enhancing asynchronous designs' energy efficiency [27].

Developing comprehensive design tools that facilitate asynchronous logic integration into mainstream practices is another emerging trend. Enhancing EDA tools for asynchronous design could significantly impact architectures like SAMIPS, fostering further optimizations and broader adoption [8]. Future research should focus on optimizing area consumption and developing dedicated storage fabrics and lossless transport protocols to bridge the gap between asynchronous circuits and networking [35].

Theoretical advancements remain crucial for asynchronous circuits' future. Expanding the framework to encompass more complex systems and exploring diverse dependencies between components could yield more robust and versatile designs [37]. Refining theoretical models to include various dynamical behaviors could enhance asynchronous systems' understanding and application in diverse contexts [30].

In industrial applications, validating designs like WAITX and developing robust circuits with guaranteed response times are vital for advancing asynchronous circuits' practical implementations [9]. Future work should concentrate on developing empirical methods to validate theoretical constructs and investigating pseudo-systems' implications in practical circuit design [36]. Extending IMITATOR II to hybrid systems and optimizing algorithms indicates emerging trends in asynchronous circuit design [16].

These efforts highlight the ongoing transition from theoretical research to real-world applications, paving the way for broader adoption of asynchronous design methodologies across various industries. Future research could explore optimizations for the SCL template, expand benchmarks to include additional QDI templates, and refine the hierarchical design process while enhancing DPA resistance in QDI circuits with minimal area overhead [18, 17]. The compatibility of proposed gates with energy-efficient RSFQ bias schemes and their potential integration into more complex RSFQ circuit designs are also emerging trends [13].

6.3 Design Complexity and Verification

The design complexity of asynchronous circuits presents multifaceted challenges, primarily due to the need to balance area, power efficiency, and security enhancements. Addressing this complexity involves optimizing designs for improved area and power efficiency while exploring countermeasures to bolster security. For instance, the RS-Buffer design introduces additional complexity in circuit implementation, necessitating careful consideration to ensure optimal performance [7].

Verification of asynchronous systems is complicated by the necessity for effective online testing mechanisms, crucial for ensuring reliability and correctness in environments where traditional testing methods may fall short [38]. The current theoretical framework used in designing asynchronous circuits may not fully encapsulate all practical complexities encountered in real-world applications, highlighting the need for more comprehensive models [39].

Designing asynchronous systems is further complicated by managing leakage sources while maintaining predictability and performance. This challenge is particularly pronounced in Quasi Delay Insensitive (QDI) circuits, where a delicate balance between security and efficiency must be achieved [17]. Additionally, the control logic circuitry required for asynchronous operation incurs extra area costs, potentially limiting these systems' applicability in resource-constrained settings [11].

Current methodologies may not fully capture all practical scenarios in complex circuit designs, necessitating further refinement of the models employed [40]. While the routing methodology is effective in many contexts, it may encounter challenges in networks with highly variable connectivity patterns, potentially leading to increased memory demands [26].

Reliance on theoretical proofs to address complexity and verification challenges underscores the importance of simplifying mathematical models and exploring practical applications of these findings. Future research should investigate the implications of these findings in real-world applications and consider expanding benchmarks to include additional asynchronous circuit designs [1]. Furthermore, future work could focus on developing a more robust mathematical theory for asynchronous automata, exploring new modeling techniques, and applying these concepts to practical circuit design [21]. By tackling these challenges, future work can pave the way for more robust and efficient asynchronous circuit designs that meet modern digital applications' demands.

7 Conclusion

This survey has explored the multifaceted nature of asynchronous circuits and dataflow architectures, highlighting their significant influence on modern digital design. By functioning independently of a global clock, asynchronous circuits offer distinct advantages in terms of power efficiency, flexibility, and resilience, particularly in applications where low power consumption and high concurrency are critical. Their integration within dataflow architectures further amplifies these benefits by enabling data-driven operations that exploit inherent parallelism and enhance system adaptability.

Key principles and methodologies essential to asynchronous circuit design have been detailed, with a focus on the role of handshake protocols and bundled data in facilitating effective communication and data transfer. The design flow process and the incorporation of Click elements have been examined, underscoring their importance in embedding asynchronous circuits within larger systems. Additionally, the investigation into dataflow architectures has demonstrated their capacity to optimize resource utilization and support asynchronous communication, aligning well with the core principles of asynchronous circuit design.

Performance evaluations reveal notable improvements in efficiency and power savings through asynchronous methodologies, supported by case studies showcasing their application in areas such as digital signal processing, secure system design, and neuromorphic computing. Challenges related to verification and design complexity have been identified, as well as emerging trends that hold promise for further innovation in this field.

Future research should focus on optimizing RS-Buffer designs for diverse applications and developing advanced algorithms for one-step encoding to address structural hazards. The survey's findings also affirm that asynchronous flows uphold properties of consistency, composition, and causality, pointing to future research opportunities that could enhance both theoretical understanding and practical implementation of asynchronous systems. By addressing these challenges and leveraging

the distinctive characteristics of asynchronous circuits, there is considerable potential to transform digital circuit design and drive progress in this domain.

www.SurveyX.cn

References

- [1] Padmanabhan Balasubramanian. Comparative evaluation of quasi-delay-insensitive asynchronous adders corresponding to return-to-zero and return-to-one handshaking. *Facta Universitatis, Series: Electronics and Energetics*, 31(1):25–39, 2017.
- [2] Shunji Nishimura. Classification of sequential circuits as causal functions, 2023.
- [3] Serban E. Vlad. Universal regular autonomous asynchronous systems: omega-limit sets, invariance and basins of attraction, 2010.
- [4] Radu Mateescu, Wendelin Serwe, Aymane Bouzafour, and Marc Renaudin. Modeling an asynchronous circuit dedicated to the protection against physical attacks, 2020.
- [5] P Balasubramanian. Critique of "asynchronous logic implementation based on factorized dims", 2018.
- [6] Serban E. Vlad. Topics in asynchronous systems, 2004.
- [7] Florian Deeg, Jie Zhu, and Sebastian M Sattler. Asynchronous design. In *AmE 2020-Automotive meets Electronics; 11th GMM-Symposium*, pages 1–5. VDE, 2020.
- [8] Qianyi Zhang and Georgios Theodoropoulos. Samips: A synthesised asynchronous processor. *arXiv preprint arXiv:2409.20388*, 2024.
- [9] Victor Khomenko, Danil Sokolov, Andrey Mokhov, and Alex Yakovlev. Waitx: An arbiter for non-persistent signals. In *2017 23rd IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, pages 33–40. IEEE, 2017.
- [10] Zhe Su, Hyunjung Hwang, Tristan Torchet, and Giacomo Indiveri. Core interface optimization for multi-core neuromorphic processors, 2023.
- [11] Basel Halak and Hsien-Chih Chiu. Modified micropipeline architecture for synthesizable asynchronous fir filter design, 2016.
- [12] Luca Cardelli, Marta Kwiatkowska, and Max Whitby. Chemical reaction network designs for asynchronous logic circuits. *Natural computing*, 17:109–130, 2018.
- [13] Gleb Krylov and Eby G Friedman. Asynchronous dynamic single-flux quantum majority gates. *IEEE Transactions on Applied Superconductivity*, 30(5):1–7, 2020.
- [14] Jintao Sun, Zeke Wang, Tao Lu, and Wenzhi Chen. Pyhgl: A python-based hardware generation language framework, 2023.
- [15] Serban E. Vlad. The non-anticipation of the asynchronous systems, 2008.
- [16] Étienne André. Imitator ii: A tool for solving the good parameters problem in timed automata, 2010.
- [17] G. F. Bousesse, M. Renaudin, S. Dumont, and F. Germain. Dpa on quasi delay insensitive asynchronous circuits: formalization and improvement, 2007.
- [18] Ricardo A Guazzelli, Matheus T Moreira, and Ney LV Calazans. A comparison of asynchronous qdi templates using static logic. In *2017 IEEE 8th Latin American Symposium on Circuits & Systems (LASCAS)*, pages 1–4. IEEE, 2017.
- [19] Serban E. Vlad. The consistency, the composition and the causality of the asynchronous flows, 2015.
- [20] Serban E. Vlad. Some first thoughts on the stability of the asynchronous systems, 2004.
- [21] Serban E. Vlad. Selected topics in asynchronous automata, 2001.
- [22] Freek Verbeek and Julien Schmaltz. Verification of building blocks for asynchronous circuits, 2013.

-
- [23] Stefano Salsano, Francesco Lombardo, Claudio Pisa, Pierluigi Greto, and Nicola Blefari Melazzi. Rdcl 3d, a model agnostic web framework for the design and composition of nfv services, 2017.
 - [24] Hajar Asgari, Nicoletta Risi, and Giacomo Indiveri. Fpga implementation of an event-driven saliency-based selective attention model, 2022.
 - [25] Dallas Aaron Phillips. Critical datapath cells for ncl asynchronous circuit area reduction. Master’s thesis, University of Cincinnati, 2022.
 - [26] Saber Moradi, Ning Qiao, Fabio Stefanini, and Giacomo Indiveri. A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (dynaps). *IEEE transactions on biomedical circuits and systems*, 12(1):106–122, 2017.
 - [27] Thiago Ferreira de Paiva Leite. *FD-SOI technology opportunities for more energy efficient asynchronous circuits*. PhD thesis, Université Grenoble Alpes, 2019.
 - [28] Carsten Nielsen, Zhe Su, and Giacomo Indiveri. Yak: An asynchronous bundled data pipeline description language, 2023.
 - [29] Serban E. Vlad. Some properties of the regular asynchronous systems, 2008.
 - [30] Serban E. Vlad. Universal regular autonomous asynchronous systems: Fixed points, equivalencies and dynamic bifurcations, 2013.
 - [31] Serban E. Vlad. On the serial connection of the regular asynchronous systems, 2012.
 - [32] Ulrich Brenner and Anna Hermann. Delay optimization of combinational logic by and-or path restructuring, 2020.
 - [33] Loukas Petrou, Petros Karousios, and Julius Georgiou. Asynchronous circuits as an enabler of scalable and programmable metasurfaces, 2019.
 - [34] P Balasubramanian and K Prasad. Latency optimized asynchronous early output ripple carry adder based on delay-insensitive dual-rail data encoding, 2017.
 - [35] Rajit Manohar and Robert Soulé. Buffer sizing problems in networks and asynchronous circuits: similarities and differences.
 - [36] Serban E. Vlad. Asynchronous pseudo-systems, 2005.
 - [37] Serban E. Vlad. The decomposition of the regular asynchronous systems as parallel connection of regular asynchronous systems, 2012.
 - [38] Jürgen Maier and Andreas Steininger. Online test vector insertion: A concurrent built-in self-testing (cbist) approach for asynchronous logic, 2020.
 - [39] Serban E. Vlad. Defining the delays of the asynchronous circuits, 2004.
 - [40] Serban E. Vlad. On the inertia of the asynchronous circuits, 2004.

Disclaimer:

SurveyX is an AI-powered system designed to automate the generation of surveys. While it aims to produce high-quality, coherent, and comprehensive surveys with accurate citations, the final output is derived from the AI's synthesis of pre-processed materials, which may contain limitations or inaccuracies. As such, the generated content should not be used for academic publication or formal submissions and must be independently reviewed and verified. The developers of SurveyX do not assume responsibility for any errors or consequences arising from the use of the generated surveys.

www.SurveyX.cn