From Cold Start to Critical:
Formal Synthesis of Autonomous Hybrid Controllers

PI: Dane A. Sabo
dane.sabo@pitt.edu

Advisor: Dr. Daniel G. Cole
dgcole@pitt.edu

Track: PhD Mechanical Engineering

Monday 9th March, 2026

This research develops autonomous control systems with mathematical guarantees of safe and correct behavior.

Nuclear reactors today require human operators who follow detailed written procedures and switch between control objectives as plant conditions change. Small modular reactors face a fundamental economic challenge: staffing costs per megawatt far exceed those of conventional plants, threatening economic viability. Autonomous control could manage complex operational sequences without constant supervision—but only if safety assurance equals or exceeds that of human operators.

This research unifies formal methods from computer science with control theory to produce hybrid control systems correct by construction. Human operators already work this way: discrete logic switches between continuous control modes. Formal methods generate provably correct switching logic but cannot verify continuous dynamics. Control theory verifies continuous behavior but cannot prove discrete logic correctness. Both are required for end-to-end correctness. Three stages bridge this gap. First, NASA's Formal Requirements Elicitation Tool (FRET) translates written operating procedures into temporal logic specifications. FRET structures requirements by scope, condition, component, timing, and response. Realizability checking exposes conflicts and ambiguities before implementation begins. Second, reactive synthesis generates deterministic automata provably correct by construction. Third, reachability analysis verifies that continuous controllers satisfy each discrete mode's requirements. Engineers design these controllers using standard control theory.

Continuous modes classify by control objective into three types. Transitory modes drive the plant between conditions. Stabilizing modes maintain operation within regions. Expulsory modes ensure safety under failures. Barrier certificates and assume-guarantee contracts prove mode transitions are safe. This enables local verification without global trajectory analysis. The methodology demonstrates on an Emerson Ovation control system—the industrial platform nuclear power plants already use. This approach manages complex nuclear power operations autonomously while maintaining safety guarantees. It directly addresses the economic constraints threatening small modular reactor viability.

This research, if successful, produces three concrete outcomes:

1. *Synthesize written procedures into verified control logic.* The methodology converts written operating procedures into formal specifications. Reactive synthesis tools then generate discrete control logic from these specifications. Control engineers can generate mode-switching controllers directly from regulatory procedures with minimal formal methods expertise, reducing barriers to high-assurance control systems.
2. *Verify continuous control behavior across mode transitions.* Reachability analysis verifies that continuous control modes satisfy discrete transition requirements. Engineers design continuous controllers using standard practices while maintaining formal correctness guarantees. Mode transitions occur safely and at the correct times—provably.
3. *Demonstrate autonomous reactor startup control with safety guarantees.* This methodology demonstrates on a small modular reactor simulation using industry-standard control hardware. Control engineers implement high-assurance autonomous controls on industrial platforms they already use, enabling autonomy without retraining costs or new equipment development.

Contents

1 Goals and Outcomes

This research develops autonomous hybrid control systems with mathematical guarantees of safe and correct behavior.

Nuclear power plants require the highest levels of control system reliability. Control system failures risk economic losses, service interruptions, or radiological release. Extensively trained human operators control nuclear plants today, following detailed written procedures and strict regulatory requirements. They switch between control modes based on plant conditions and procedural guidance. This reliance on human operators prevents autonomous control and creates a fundamental economic challenge for next-generation reactor designs. Small modular reactors face staffing costs per megawatt far exceeding those of conventional plants, threatening economic viability. Autonomous control could manage complex operational sequences without constant supervision—but only if safety assurance equals or exceeds that of human operators.

This research unifies formal methods with control theory to produce hybrid control systems correct by construction. Human operators already work this way: discrete logic switches between continuous control modes. Formal methods generate provably correct switching logic from written requirements but cannot verify the continuous dynamics governing transitions. Control theory verifies continuous behavior but cannot prove discrete switching correctness. Both are required for end-to-end correctness. Two steps close this gap. First, reactive synthesis generates discrete mode transitions directly from written operating procedures. Second, reachability analysis verifies continuous behavior against discrete requirements. Operating procedures transform into logical specifications that constrain continuous dynamics, producing autonomous controllers provably free from design defects.

The University of Pittsburgh Cyber Energy Center provides access to industry collaboration and Emerson control hardware, ensuring solutions align with practical implementation requirements.

If successful, this approach produces three concrete outcomes:

1. **Translate written procedures into verified control logic.** A methodology converts written operating procedures into formal specifications. Reactive synthesis tools then automatically generate discrete control logic from these specifications. Structured intermediate representations bridge natural language procedures and mathematical logic. Control system engineers can generate verified mode-switching controllers directly from regulatory procedures. They need no formal methods expertise. This lowers the barrier to high-assurance control systems.
2. **Verify continuous control behavior across mode transitions.** Methods for analyzing continuous control modes verify that they satisfy discrete transition requirements. Classical control theory handles linear systems, while reachability analysis handles nonlinear dynamics. Both verify that each continuous mode reaches its intended transitions safely. Engineers design continuous controllers using standard practices while maintaining formal correctness guarantees. Mode transitions occur safely and at the correct times—provably.
3. **Demonstrate autonomous reactor startup control with safety guarantees.** This methodology applies to autonomous nuclear reactor startup procedures, demonstrating on a small modular reactor simulation using industry-standard control hardware. The demonstration proves correctness across multiple coordinated control modes from cold shutdown through criticality to power operation. Autonomous hybrid control becomes realizable in the nuclear industry with current equipment, establishing a path toward reduced operator staffing while maintaining safety.

What makes this research new? No existing methodology achieves end-to-end correctness guarantees for hybrid systems. Section 2 will show that prior work verified discrete logic or continuous dynamics—never both compositionally. This work unifies discrete synthesis with continuous verification through a key innovation: discrete specifications become contracts that continuous controllers must satisfy. Each layer verifies independently while guaranteeing correct composition. Formal methods verify discrete logic. Control theory verifies continuous dynamics. These three outcomes—procedure translation, continuous verification, and hardware demonstration—establish a complete methodology from regulatory documents to deployed systems.

If successful, control engineers will create autonomous controllers from existing procedures with mathematical proofs of correct behavior, making high-assurance autonomous control practical for safety-critical applications. This capability is essential for the economic viability of next-generation nuclear power. Small modular reactors, in particular, offer a promising solution to growing energy demands, but their success depends on reducing per-megawatt operating costs through increased autonomy. This research provides the tools to achieve that autonomy while maintaining the exceptional safety record the nuclear industry requires.

This proposal follows the Heilmeier Catechism. Each section explicitly answers its assigned questions:

- **Section 2 (State of the Art):** What has been done? What are the limits of current practice?
- **Section 3 (Research Approach):** What is new? Why will it succeed?
- **Section 4 (Metrics for Success):** How will success be measured?
- **Section 5 (Risks and Contingencies):** What could prevent success?
- **Section 6 (Broader Impacts):** Who cares? Why now? What difference will it make?
- **Section 8 (Schedule):** How long will it take?

Each section begins by stating its Heilmeier questions and ends by summarizing its answers. This ensures both local clarity and global coherence.

2 State of the Art and Limits of Current Practice

Heilmeier Questions: What has been done? What are the limits of current practice?

No current approach provides end-to-end correctness guarantees for autonomous control. Human-centered operation cannot eliminate reliability limits. Formal methods verify discrete or continuous behavior—never both.

Three subsections structure this analysis: reactor operators and their operating procedures, fundamental limitations of human-based operation, and formal methods approaches that verify discrete logic or continuous dynamics but not both together.

Section 3 addresses the verification gap these limits establish.

2.1 Current Reactor Procedures and Operation

Current practice rests on two critical components: procedures and operators. Procedures define what must be done. Operators execute those procedures. This subsection examines procedures—their hierarchy, development process, and role in defining operational modes. The following subsection examines operators—their reliability limits and contribution to accidents.

Nuclear plant procedures form a strict hierarchy. Normal operating procedures govern routine operations. Abnormal operating procedures handle off-normal conditions. Emergency Operating Procedures (EOPs) manage design-basis accidents. Severe Accident Management Guidelines (SAMGs) address beyond-design-basis events. Extensive Damage Mitigation Guidelines (EDMGs) cover catastrophic damage. All procedures must comply with 10 CFR 50.34(b)(6)(ii); NUREG-0899 provides development guidance [**?**, **?**].

Expert judgment and simulator validation—not formal verification—form the basis for procedure development. Regulations mandate rigorous assessment. Specifically, 10 CFR 55.59 [**?**] requires technical evaluation, simulator validation testing, and biennial review. Yet key safety properties escape formal verification. No mathematical proof confirms that procedures cover all possible plant states. No proof confirms that required actions complete within available timeframes. No proof confirms that transitions between procedure sets maintain safety invariants.

LIMITATION: *Procedures lack formal verification of correctness and completeness.* No proof exists that procedures cover all possible plant states, that required actions complete within available timeframes, or that transitions between procedure sets maintain safety invariants. Paper-based procedures cannot ensure correct application. Even computer-based procedure systems lack the formal guarantees automated reasoning could provide.

Nuclear plants operate with multiple control modes. Automatic control maintains target parameters through continuous reactivity adjustment. Manual control allows operators to directly manipulate the reactor. Various intermediate modes bridge these extremes. In typical pressurized water reactor operation, the reactor control system automatically maintains a floating average temperature, compensating for power demand changes through reactivity feedback loops alone.

Safety systems already employ extensive automation. Reactor Protection Systems trip automatically on safety signals with millisecond response times. Engineered safety features actuate automatically on accident signals—no operator action required. This division between automated and human-controlled functions reveals the fundamental challenge of hybrid control. Highly automated systems already handle reactor protection—automatic trips on safety parameters, emergency core cooling actuation, containment isolation, and basic process control [**?**, **?**]. Human operators retain control of strategic decision-making: power level changes, startup/shutdown sequences, mode transitions, and procedure implementation. This hybrid structure—discrete human decisions combined with continuous automated control—forms the basis for autonomous hybrid control systems.

2.2 Human Factors in Nuclear Accidents

Procedures lack formal verification despite rigorous development. This represents only half the reliability challenge: even perfect procedures cannot guarantee safe operation when humans execute them imperfectly.

Human operators—the second pillar of current practice—introduce reliability limitations independent of procedure quality. Procedures define what to do. Operators determine when and how to act. This discretion introduces persistent failure modes. Training alone cannot eliminate these failures.

Current-generation nuclear power plants employ over 3,600 active NRC-licensed reactor operators in the United States [**?**]. These operators divide into Reactor Operators (ROs), who manipulate reactor controls, and Senior Reactor Operators (SROs), who direct plant operations and serve as shift supervisors [**?**]. Staffing typically requires at least two ROs and one SRO for current-generation units [**?**]. Becoming a reactor operator requires several years of training.

Human error persistently contributes to nuclear safety incidents despite decades of improvements in training and procedures. This persistence cannot be trained away. It motivates the need for formal automated control with mathematical safety guarantees.

Under 10 CFR Part 55, operators hold legal authority to make critical decisions. This includes authority to depart from normal regulations during emergencies. The Three Mile Island (TMI) accident demonstrated how personnel error, design deficiencies, and component failures combine to cause disaster. Operators misread confusing and contradictory indications, then shut off the emergency water system [**?**].

The President's Commission on TMI identified a fundamental ambiguity. Placing responsibility for safe power plant operations on the licensee does not guarantee safety—not without formally verifying that operators can fulfill this responsibility. This tension between operational flexibility and safety assurance remains unresolved. The person responsible for reactor safety often becomes the root cause of failure.

Multiple independent analyses converge on a striking statistic: human error accounts for 70–80% of nuclear power plant events [**?**]. Equipment failures account for only 20%. More significantly, human factors—poor safety management and safety culture—caused all severe accidents at nuclear power plants: Three Mile Island, Chernobyl, and Fukushima Daiichi [**?**]. A detailed analysis of 190 events at Chinese nuclear power plants from 2007–2020 [**?**] found that active errors appeared in 53% of events, while latent errors—organizational and systemic weaknesses that create conditions for failure—appeared in 92%.

LIMITATION: *Human factors impose fundamental reliability limits that training alone cannot overcome.* Four decades of improvements have failed to eliminate human error—these limitations are fundamental to human-driven control, not remediable defects.

2.3 Formal Methods

The previous subsections established two fundamental limitations: procedures lack formal verification, and human operators introduce persistent reliability issues that training cannot eliminate. Both represent fundamental constraints—not remediable defects.

Formal methods could eliminate both limitations by providing mathematical guarantees of correctness. However, even the most advanced formal methods applications in nuclear control leave a critical verification gap.

This subsection examines two approaches illustrating this gap. HARDENS verified discrete logic without continuous dynamics. Differential dynamic logic handles hybrid verification only post-hoc. Each demonstrates the current state of formal methods while revealing the verification gap this research addresses.

2.3.1 HARDENS: The State of Formal Methods in Nuclear Control

The High Assurance Rigorous Digital Engineering for Nuclear Safety (HARDENS) project represents the most advanced application of formal methods to nuclear reactor control systems to date [**?**].

HARDENS addressed a fundamental dilemma: existing U.S. nuclear control rooms rely on analog technologies from the 1950s–60s. These technologies incur significant risk and cost compared to modern control systems. The NRC contracted Galois, a formal methods firm, to demonstrate that Model-Based Systems Engineering and formal methods could design, verify, and implement a complex protection system meeting regulatory criteria at a fraction of typical cost. The project delivered a Reactor Trip System (RTS) implementation with full traceability from NRC Request for Proposals and IEEE standards through formal architecture specifications to verified software.

HARDENS employed formal methods tools and techniques across the verification hierarchy. High-level specifications used Lando, SysMLv2, and FRET (NASA Formal Requirements Elicitation Tool) to capture stakeholder requirements, domain engineering, certification requirements, and safety requirements. Requirements were analyzed for consistency, completeness, and realizability using SAT and SMT solvers. Executable formal models used Cryptol to create a behavioral model of the entire RTS, including all subsystems, components, and limited digital twin models of sensors, actuators, and compute infrastructure. Automatic code synthesis generated verifiable C implementations and SystemVerilog hardware implementations directly from Cryptol models—eliminating the traditional gap between specification and implementation where errors commonly arise.

Despite its accomplishments, HARDENS has a fundamental limitation for hybrid control synthesis: the project addressed only discrete digital control logic. Continuous reactor dynamics remained unmodeled and unverified. The Reactor Trip System specification and verification covered discrete state transitions (trip/no-trip decisions), digital sensor input processing through discrete logic, and discrete actuation outputs (reactor trip commands). Continuous reactor physics remained unaddressed. Real reactor safety depends on interactions between continuous processes—temperature, pressure, neutron flux—evolving in response to discrete control decisions. HARDENS verified the discrete controller in isolation. The closed-loop hybrid system behavior remained unverified.

LIMITATION: *HARDENS addressed discrete control logic without continuous dynamics or hybrid system verification.* Verifying discrete control logic alone provides no guarantee that the closed-loop system exhibits desired continuous behavior such as stability, convergence to setpoints, or maintained safety margins.

HARDENS also faced deployment maturity constraints beyond the technical limitation of omitting continuous dynamics. The project produced a demonstrator system at Technology Readiness Level 2–3 (analytical proof of concept with laboratory breadboard validation) rather than a deployment-ready system validated through extended operational testing. The NRC Final Report explicitly notes [?] that all material is considered in development, not a finalized product, and that "The demonstration of its technical soundness was to be at a level consistent with satisfaction of the current regulatory criteria, although with no explicit demonstration of how regulatory requirements are met." The project did not include deployment in actual nuclear facilities, testing with real reactor systems under operational conditions, side-by-side validation with operational analog RTS systems, systematic failure mode testing (radiation effects, electromagnetic interference, temperature extremes), NRC licensing review, or human factors validation with licensed operators in realistic control room scenarios.

LIMITATION: *HARDENS achieved TRL 2–3 without experimental validation.* While formal verification provides mathematical correctness guarantees for the implemented discrete logic, the gap between formal verification and actual system deployment involves myriad practical considerations: integration with legacy systems, long-term reliability under harsh environments, human-system interaction in realistic operational contexts, and regulatory acceptance of formal methods as primary assurance evidence.

2.3.2 Differential Dynamic Logic: Post-Hoc Hybrid Verification

HARDENS verified discrete control logic without continuous dynamics—leaving half the hybrid system unverified.

Other researchers have attacked the problem from the opposite direction, extending temporal logics to handle hybrid systems directly. This complementary approach produced differential dynamic logic (dL). dL addresses continuous dynamics but encounters different limitations. dL introduces two additional operators into temporal logic: the box operator and the diamond operator. The box operator $[\alpha]\phi$ states that for some region ϕ, the hybrid system α always remains within that region. In this way, it is a safety invariant being enforced for the system. The second operator, the diamond operator $< \alpha > \phi$ says that for the region ϕ, there is at least one trajectory of α that enters that region. This is a declaration of a liveness property.

While dL allows for the specification of these liveness and safety properties, actually proving them for a given hybrid system is difficult. Automated proof assistants such as KeYmaera X exist to help develop proofs of systems using dL, but fail for reasonably complex hybrid systems. State space explosion and non-terminating solutions prevent creating system proofs using dL. Approaches have been made to alleviate these issues for nuclear power contexts using contract and decomposition based methods, but fall far short of a complete design methodology. Instead, these approaches have been used on systems that have been designed a priori, and require expert knowledge to create the system proofs.

LIMITATION: *Logic-based hybrid system verification has not scaled to system design.* While dL and related approaches can verify hybrid systems post-hoc, they require expert knowledge and have been applied only to systems designed a priori. State space explosion prevents their use in the design loop for complex systems like nuclear reactor startup procedures.

2.4 Summary: The Verification Gap

This section addressed two Heilmeier questions: What has been done? What are the limits of current practice?

What has been done? Three approaches currently exist, each with fundamental limitations. Human operators provide operational flexibility but introduce persistent reliability constraints. HARDENS verified discrete logic but omitted continuous dynamics. Differential dynamic logic expresses hybrid properties but requires post-design expert analysis. None addresses both discrete and continuous verification compositionally.

What are the limits of current practice? A clear verification gap emerges: no existing methodology synthesizes provably correct hybrid controllers from operational procedures with verification integrated into design. Current approaches verify discrete logic or continuous dynamics—never both compositionally. Training improvements cannot overcome human reliability limits. Post-hoc verification cannot scale to system design.

Why now? Two forces create urgency. First, economic necessity: small modular reactors cannot compete with per-megawatt staffing costs matching large conventional plants. Second, technical maturity: formal methods tools have matured sufficiently to enable compositional hybrid verification. These forces converge to make this work both necessary and achievable.

Section 2 established what has been done and the limits of current practice. The verification gap is clear. The timing is right. Section 3 addresses the next two Heilmeier questions—what is new and why it will succeed—presenting the technical approach that closes this gap.

3 Research Approach

Heilmeier Questions: What is new? Why will it succeed?

This section presents the complete technical approach for synthesizing provably correct hybrid controllers from operating procedures.

What is new: Three innovations enable compositional verification bridging discrete synthesis with continuous control: contract-based decomposition, mode classification, and procedure-driven structure.

Why it will succeed: The approach leverages existing procedural structure, bounds computational complexity through mode-level verification, and validates against real industrial hardware through Emerson collaboration.

Previous approaches verified discrete switching logic or continuous control behavior—never both simultaneously. Engineers validate continuous controllers through extensive simulation trials and test discrete switching logic through simulated control room testing and human factors research. Neither method provides rigorous guarantees; both consume enormous resources.

This approach bridges that gap by composing formal methods from computer science with control-theoretic verification and formalizing reactor operations as hybrid automata.

Hybrid system verification faces a fundamental challenge: discrete transitions change the governing vector field, creating discontinuities that traditional verification techniques cannot handle directly.

This methodology decomposes the problem by verifying discrete switching logic and continuous mode behavior separately, then composing them to establish guarantees for the complete hybrid system. This two-layer approach mirrors reactor operations: discrete supervisory logic determines which control mode is active, while continuous controllers govern plant behavior within each mode.

Hybrid systems require mathematical formalization. This work draws on automata theory, temporal logic, and control theory to provide that description.

A hybrid system is a dynamical system with both continuous and discrete states. This proposal addresses continuous autonomous hybrid systems specifically—systems with no external input where continuous states remain continuous when discrete states change. This work follows the nomenclature from the Handbook on Hybrid Systems Control [**?**], redefined here for convenience:

$$H = (\mathscr{Q}, \mathscr{X}, \mathbf{f}, Init, \mathscr{G}, \delta, \mathscr{R}, Inv) \tag{1}$$

where:

- $\mathscr{Q}$: the set of discrete states (modes) of the system
- $\mathscr{X} \subseteq \mathbb{R}^n$: the continuous state space
- $\mathbf{f} : \mathscr{Q} \times \mathscr{X} \to \mathbb{R}^n$: vector fields defining the continuous dynamics for each discrete mode q_i
- $Init \subseteq \mathscr{Q} \times \mathscr{X}$: the set of initial states
- $\mathscr{G}$: guard conditions that define when discrete state transitions may occur
- $\delta : \mathscr{Q} \times \mathscr{G} \to \mathscr{Q}$: the discrete state transition function
- $\mathscr{R}$: reset maps that define any instantaneous changes to continuous state upon discrete transitions
- Inv: safety invariants on the continuous dynamics

A HAHACS requires this tuple together with proof artifacts demonstrating that the control system's actual implementation satisfies its intended behavior.

What is new in this research? Existing approaches verify discrete logic or continuous dynamics—never both compositionally. Section 2 established this limitation: reactive synthesis, reachability analysis, and barrier certificates exist independently but have never been integrated into a systematic design methodology. Prior work cannot span from procedures to verified implementation.

Three innovations enable compositional verification:

1. **Contract-based decomposition:** Instead of attempting global hybrid system verification, discrete synthesis defines entry/exit/safety contracts that bound continuous verification, transforming an intractable global problem into tractable local problems.
2. **Mode classification:** Continuous modes classify by control objective—transitory, stabilizing, or expulsory—allowing appropriate verification tools to match each mode type and enabling mode-local analysis with provable composition guarantees.
3. **Procedure-driven structure:** Nuclear procedures already decompose operations into discrete phases with explicit transition criteria, providing existing structure that avoids artificial abstractions and makes the approach tractable for complex systems like nuclear reactor startup.

Why will it succeed? Three factors ensure practical feasibility where prior work has failed.

Existing structure: Nuclear procedures already decompose operations into discrete phases with explicit transition criteria. The approach formalizes this existing structure without imposing artificial abstractions, enabling domain experts to adopt the methodology without formal methods training.

Bounded complexity: Mode-level verification checks each mode against local contracts, avoiding global hybrid system analysis. This decomposition bounds computational complexity, transforming an intractable global problem into tractable local verifications.

Industrial validation: The Emerson collaboration provides domain expertise to validate procedure formalization and industrial hardware to demonstrate implementation feasibility, ensuring solutions address real deployment constraints rather than just theoretical correctness.

These factors combine to demonstrate feasibility on production control systems with realistic reactor models—not merely in principle. Figure **??** illustrates the hybrid structure for a simplified reactor startup sequence.

3.1 System Requirements, Specifications, and Discrete Controllers

The previous subsection established the hybrid automaton formalism: a mathematical framework describing discrete modes, continuous dynamics, guards, and invariants. This formalism provides the mathematical structure for hybrid control. A critical question remains—where do these formal descriptions originate?

Nuclear operations already possess a natural hybrid structure that maps directly to this formalism through three control scopes: strategic, operational, and tactical. This approach constructs formal hybrid systems from existing operational knowledge, leveraging decades of domain expertise already encoded in operating procedures rather than imposing artificial

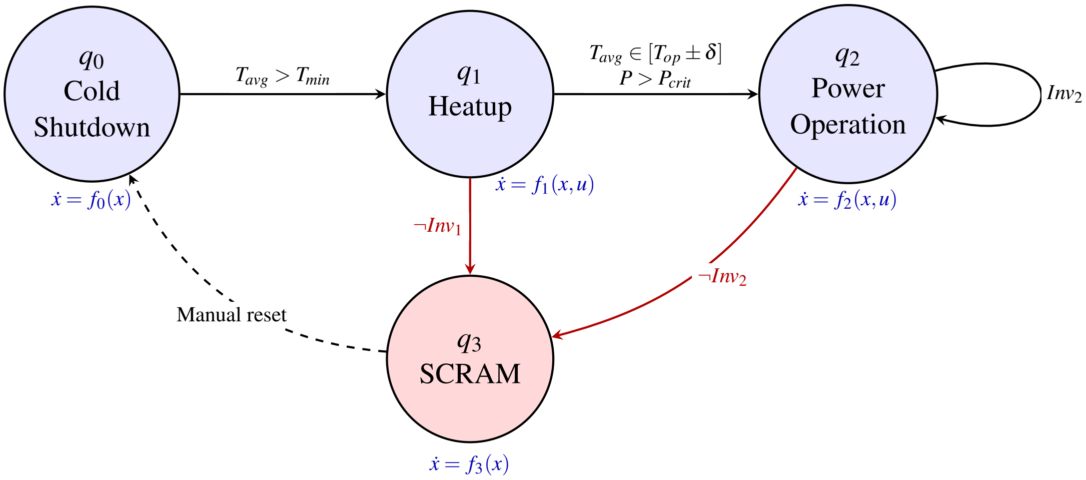

Figure 1: Simplified hybrid automaton for reactor startup. Each discrete state q_i has associated continuous dynamics f_i. Guard conditions (e.g., $T_{avg} > T_{min}$) are predicates over continuous state. Invariant violations ($\neg Inv_i$) trigger transitions to the SCRAM state. The operational level manages discrete transitions; the tactical level executes continuous control within each mode.

abstractions.

Human control of nuclear power divides into three scopes: strategic, operational, and tactical. Strategic control represents high-level, long-term decision making spanning months or years: managing labor needs and supply chains to optimize scheduled maintenance and downtime.

Tactical control manages individual components—pumps, turbines, and chemistry. Nuclear power plants have already automated tactical control through "automatic control" systems: continuous systems that maintain pressurizer level, core temperature, and reactivity through chemical shim. These systems directly impact the physical state of the plant.

The operational scope links these extremes. It represents the primary responsibility of human operators today. Operators implement tactical control sequences to achieve strategic objectives, bridging high-level goals with low-level execution.

Consider refueling as an example. The strategic level sets the refueling schedule. The tactical level maintains core temperature. The operational level issues the shutdown procedure, achieving the strategic goal through several smaller tactical objectives.

This structure reveals why the operational and tactical levels fundamentally form a hybrid controller. The tactical level represents continuous plant evolution according to control input and control law. The operational level represents discrete state evolution determining which tactical control law applies. Together, these two levels form the complete hybrid system. This operational level becomes the target for autonomous control because it bridges strategic intent with tactical execution through discrete mode-switching decisions.

This operational control level explains why nuclear control requires human operators: the hybrid nature of this control system makes proving controller performance against strategic requirements difficult, and no unified infrastructure exists for building and verifying hybrid systems. Humans fill this layer because their general intelligence provides a safe way to

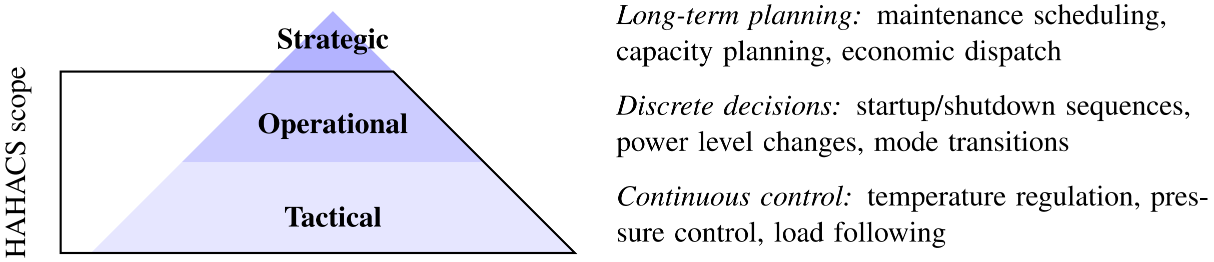
HAHACS scope
Strategic
Operational
Tactical
Long-term planning: maintenance scheduling, capacity planning, economic dispatch
Discrete decisions: startup/shutdown sequences, power level changes, mode transitions
Continuous control: temperature regulation, pressure control, load following

Figure 2: Control scope hierarchy in nuclear power operations. Strategic control (long-term planning) remains with human management. HAHACS addresses the operational level (discrete mode switching) and tactical level (continuous control within modes), which together form a hybrid control system.

manage the system's hybrid nature by following prescriptive operating manuals where strict procedures govern what control to implement at any given time.

These procedures provide the key to HAHACS construction, which leverages two observations about current practice. First, operational scope control is effectively discrete control. Second, operating procedures describe implementation rules before construction begins, meaning a HAHACS's intended behavior can be completely specified before implementation. Requirements define the behavior of any control system: statements about what the system must do, must not do, and under what conditions. For nuclear systems, these requirements derive from multiple sources including regulatory mandates, design basis analyses, and operating procedures. The challenge is formalizing these requirements with sufficient precision that they can serve as the foundation for autonomous control system synthesis and verification. We can build these requirements using temporal logic.

Temporal logic provides powerful semantics for building systems with complex but deterministic behavior, extending classical propositional logic with operators that express properties over time. Temporal logic relates discrete control modes to one another and defines all HAHACS requirements. Boundary conditions between discrete states determine guard conditions $\mathscr{G}$ and specify their behavior. Continuous mode invariants similarly express as temporal logic statements, forming the basis of any proofs about a HAHACS and constituting fundamental truth statements about designed system behavior.

Discrete mode transitions include predicates—Boolean functions over the continuous state space: $p_i : \mathscr{X} \to \{\text{true}, \text{false}\}$. These predicates formalize conditions like "coolant temperature exceeds 315°C" or "pressurizer level is between 30% and 60%." This discrete abstraction is not imposed artificially—operating procedures for nuclear systems already define go/no-go conditions as discrete predicates. Design basis safety analysis determined these thresholds; decades of operational experience validated them. Our methodology assumes this domain knowledge exists and provides a framework to formalize it. The approach proves feasible for nuclear applications because generations of nuclear engineers already completed the hard work of defining safe operating boundaries.

Linear temporal logic (LTL) is particularly well-suited for specifying reactive systems. LTL formulas are built from atomic propositions (our discrete predicates) using Boolean connectives and temporal operators. The key temporal operators are:

- $\mathbf{X}\phi$ (next): ϕ holds in the next state
- $\mathbf{G}\phi$ (globally): ϕ holds in all future states
- $\mathbf{F}\phi$ (finally): ϕ holds in some future state
- $\phi\mathbf{U}\psi$ (until): ϕ holds until ψ becomes true

These operators allow us to express safety properties ("the reactor never enters an unsafe configuration"), liveness properties ("the system eventually reaches operating temperature"), and response properties ("if coolant pressure drops, the system initiates shutdown within bounded time").

I use FRET (Formal Requirements Elicitation Tool)—developed by NASA for high-assurance timed systems—to build these temporal logic statements. FRET provides an intermediate language between temporal logic and natural language. It enables rigid definitions of temporal behavior through syntax accessible to engineers without formal methods expertise. This accessibility proves crucial for industrial feasibility: the current nuclear workforce can adopt these tools without extensive formal methods training.

FRET's key feature is its ability to start with logically imprecise statements and refine them consecutively into well-posed specifications. We leverage this by directly importing operating procedures and design requirements into FRET in natural language, then iteratively refining them into specifications for a HAHACS. This approach provides two distinct benefits: first, it draws a direct link from design documentation to digital system implementation; second, it clearly demonstrates where natural language documents fall short. Human operators may still use these procedures, making any room for interpretation a weakness requiring correction.

FRET has been successfully applied to spacecraft control systems, autonomous vehicle requirements, and medical device specifications. NASA used FRET for the Lunar Gateway project, formalizing flight software requirements that were previously specified only in natural language. The Defense Advanced Research Projects Agency (DARPA) employed FRET in the Assured Autonomy program to verify autonomous systems requirements. These applications demonstrate FRET's maturity for safety-critical domains. Nuclear control procedures present an ideal use case: they are already structured, detailed, and written to minimize ambiguity—precisely the characteristics that enable successful formalization.

3.2 Discrete Controller Synthesis

The previous subsection demonstrated how operating procedures translate into temporal logic specifications using FRET. These specifications define what the system must do—but a critical gap remains: how do we implement those requirements?

Reactive synthesis bridges this gap by automatically constructing controllers guaranteed to satisfy temporal logic specifications. It automates the creation of reactive programs from temporal logic—programs that take input for a given state and produce output. The current discrete state and guard condition status form the input; the next discrete state forms the output.

Reactive synthesis solves a fundamental problem: given an LTL formula φ specifying desired system behavior, automatically construct a finite-state machine (strategy) that produces outputs in response to environment inputs such that all resulting execution traces satisfy φ. If such a strategy exists, the specification is *realizable*. The synthesis algorithm either produces a correct-by-construction controller or reports that no such controller exists. Unrealizable specifications indicate conflicting or impossible requirements in the original procedures—this realizability check catches errors before implementation.

Reactive synthesis offers a decisive advantage: the discrete automaton requires no human engineering of its implementation and is correct by construction. This eliminates human error at the implementation stage entirely, allowing human designers to focus effort where it belongs—on specifying system behavior rather than implementing switching logic.

This shift carries two critical implications. First, complete traceability: the controller changes between modes for reasons that trace back through specifications to requirements. This establishes clear liability and justification for system behavior. Second, deterministic guarantees replace probabilistic human judgment. Human operators cannot eliminate error from discrete control decisions. Humans are intrinsically fallible. Temporal logics define system behavior. Deterministic algorithms synthesize the controller. Strategic decisions follow operating procedures exactly—no exceptions, no deviations, no human factors.

The synthesized automaton translates directly to executable code through standard compilation techniques. Each discrete state maps to a control mode, guard conditions map to conditional statements, and the transition function defines the control flow. This compilation process preserves the formal guarantees by ensuring the implemented code is correct by construction—the automaton from which it derives was synthesized to satisfy the temporal logic specifications.

Reactive synthesis has proven successful in robotics, avionics, and industrial control. Recent applications synthesize robot motion planners from natural language specifications. They generate flight control software for unmanned aerial vehicles. They create verified controllers for automotive systems. These successes demonstrate that reactive synthesis scales beyond toy problems to real-world safety-critical applications.

Reactive synthesis produces discrete mode-switching logic from procedures. The next subsection addresses what executes within each discrete mode: continuous control and its verification.

3.3 Continuous Control Modes

Reactive synthesis produces a provably correct discrete controller that determines when to switch between modes. However, hybrid control requires more than correct mode switching—the continuous dynamics executing within each discrete mode must also verify against requirements.

Control objectives determine the verification approach. Modes classify into three types—transitory, stabilizing, and expulsory—each requiring different verification tools matched to its distinct purpose. This subsection describes each type and its verification method.

This methodology's scope requires clarification: this work verifies continuous controllers but does not synthesize them. The distinction parallels model checking in software verification, which confirms whether an implementation satisfies its specification without prescribing how to write the software. Engineers design continuous controllers using standard control theory techniques—this work assumes that capability exists. The contribution lies in the verification framework confirming that candidate controllers compose correctly with the discrete layer to produce a safe hybrid system.

The operational control scope defines go/no-go decisions that determine what kind of continuous control to implement. The entry or exit conditions of a discrete state are the guard conditions $\mathscr{G}$ that define the boundaries for each continuous controller's allowed state-space region. These continuous controllers all share a common state space, but each individual continuous control mode operates within its own partition—defined by the discrete state q_i and the associated guards.

This partitioning of the continuous state space among several discrete vector fields has traditionally posed a difficult problem for validation and verification. The discontinuity of the vector fields at discrete state interfaces makes reachability analysis computationally expensive, and analytic solutions often become intractable [**?**].

I circumvent these issues by designing the hybrid system from the bottom up with verification in mind. The discrete transitions define each continuous control mode's input and output sets clearly *a priori*.

Each discrete mode q_i provides three key pieces of information for continuous controller design:

(a) **Entry conditions:** $\mathscr{X}_{entry,i} \subseteq \mathscr{X}$, the set of possible initial states when entering this mode
(b) **Exit conditions:** $\mathscr{X}_{exit,i} \subseteq \mathscr{X}$, the target states that trigger transition to the next mode, or is the region in the state space a stabilizing mode remains within.
(c) **Safety invariants:** $\mathscr{X}_{safe,i} \subseteq \mathscr{X}$, the envelope of safe states during operation in this mode. These are derived from invariants *Inv*.

These sets come directly from the discrete controller synthesis and define precise objectives for continuous control. The continuous controller for mode q_i must drive the system from any state in $\mathscr{X}_{entry,i}$ to some state in $\mathscr{X}_{exit,i}$ while remaining within $\mathscr{X}_{safe,i}$.

Continuous controllers classify into three types based on control objectives, each requiring distinct verification tools. Transitory modes drive the plant between operating conditions. Stabilizing modes maintain the plant within operating regions. Expulsory modes ensure safety under degraded conditions.

The following subsections detail each mode type and its verification approach, progressing from nominal operations—transitory and stabilizing modes—to off-nominal scenarios handled by expulsory modes.

3.3.1 Transitory Modes

Transitory modes—the first of three continuous controller types—execute transitions between operating conditions. Their purpose is to move the plant from one discrete operating condition to another. They start from entry conditions, reach exit conditions, and maintain safety invariants throughout. Examples include power ramp-up sequences, cooldown procedures, and load-following maneuvers.

The control objective for a transitory mode has a formal statement. Given entry conditions $\mathscr{X}_{entry}$, exit conditions $\mathscr{X}_{exit}$, safety invariant $\mathscr{X}_{safe}$, and closed-loop dynamics $\dot{x} = f(x, u(x))$, the controller must satisfy:

$$\forall x_0 \in \mathscr{X}_{entry} : \exists T > 0 : x(T) \in \mathscr{X}_{exit} \wedge \forall t \in [0,T] : x(t) \in \mathscr{X}_{safe}$$

This requirement is straightforward: from any valid entry state, the trajectory must eventually reach the exit condition without ever leaving the safe region.

Reachability analysis provides the natural verification tool for transitory modes. It computes the set of all states reachable from a given initial set under the system dynamics. For a transitory mode to be valid, the reachable set from $\mathscr{X}_{entry}$ must satisfy two conditions:

(a) The reachable set eventually intersects $\mathscr{X}_{exit}$—the mode achieves its objective
(b) The reachable set never leaves $\mathscr{X}_{safe}$—safety is maintained throughout the transition

Formally, if $\text{Reach}(\mathscr{X}_{entry}, f, [0,T])$ denotes the states reachable within time horizon T:

$$\text{Reach}(\mathscr{X}_{entry}, f_i, [0,T]) \subseteq \mathscr{X}_{safe} \wedge \text{Reach}(\mathscr{X}_{entry}, f_i, [0,T]) \cap \mathscr{X}_{exit} \neq \emptyset$$

The discrete controller defines clear boundaries in continuous state space. This makes the verification problem for each transitory mode well-posed. The possible initial conditions, target conditions, and safety envelope are all known before verification begins. The verification task then confirms that the candidate continuous controller achieves the objective from all possible starting points.

Several tools exist for computing reachable sets of hybrid systems, including CORA, Flow*, SpaceEx, and JuliaReach. The choice of tool depends on the structure of the continuous dynamics. Linear systems admit efficient polyhedral or ellipsoidal reachability computations. Nonlinear systems require more conservative over-approximations using techniques such as Taylor models or polynomial zonotopes. For this work, we will select tools appropriate to the fidelity of the reactor models available.

3.3.2 Stabilizing Modes

Transitory modes drive the system toward exit conditions. Reachability analysis verifies whether the target can be reached.

Stabilizing modes address a different question: will the system stay within bounds? These modes maintain the system within a desired operating region indefinitely. Examples include steady-state power operation, hot standby, and load-following at constant power level. This different control objective requires a different verification approach.

Reachability analysis answers ”can the system reach a target?” Stabilizing modes instead ask ”does the system stay within bounds?” Barrier certificates provide the appropriate verification tool for this question. Barrier certificates analyze the dynamics of the system to determine whether flux across a given boundary exists. They evaluate whether any trajectory leaves a given boundary. This definition exactly matches what defines the validity of a stabilizing continuous control mode.

Formally, a barrier certificate (or control barrier function) is a scalar function $B : \mathscr{X} \to \mathbb{R}$ that certifies forward invariance of a safe set. The idea parallels Lyapunov functions for stability: rather than computing trajectories explicitly, we seek a certificate function whose properties guarantee the desired behavior. For a safe set $\mathscr{C} = \{x : B(x) \geq 0\}$ and dynamics $\dot{x} = f(x,u)$, the barrier certificate condition requires:

$$\forall x \in \partial\mathscr{C} : \dot{B}(x) = \nabla B(x) \cdot f(x, u(x)) \geq 0$$

This condition states that on the boundary of the safe set (where $B(x) = 0$), the time derivative of B is non-negative. Geometrically, this means the vector field points inward or tangent to the boundary, never outward. If this condition holds, no trajectory starting inside $\mathscr{C}$ can ever leave.

The discrete controller design defines careful boundaries in continuous state space. Therefore, the barrier is known prior to designing the continuous controller. This eliminates the search for an appropriate barrier. It minimizes complication in validating stabilizing continuous control modes. The discrete specifications tell us what region must be invariant. The barrier certificate then confirms that the candidate controller achieves this invariance.

Finding barrier certificates can be formulated as a sum-of-squares (SOS) optimization problem for polynomial systems, or solved using satisfiability modulo theories (SMT) solvers for broader classes of dynamics. The key advantage is that the verification is independent of how the controller was designed. Standard control techniques can be used to build continuous controllers, and barrier certificates provide a separate check that the result satisfies the required invariants. This also allows for the checking of control modes with different models than they are designed for. For example, a lower fidelity model can be used for controller design, but a higher fidelity model can be used for the actual validation of that stabilizing controller.

3.3.3 Expulsory Modes

The first two mode types handle nominal operations. Transitory modes move the plant between conditions using reachability analysis. Stabilizing modes maintain the plant within regions using barrier certificates. Both assume the plant dynamics match the design model.

Expulsory modes address a different scenario: situations where the plant deviates from expected behavior. This deviation may result from component failures, sensor degradation, or unanticipated disturbances. Here, robustness matters more than optimality.

Expulsory controllers prioritize robustness over optimality. The control objective shifts from reaching targets or maintaining regions to driving the plant to a safe shutdown state from potentially anywhere in the state space, under degraded or uncertain dynamics. Examples include emergency core cooling, reactor SCRAM sequences, and controlled depressurization procedures.

Proving controller correctness through reachability and barrier certificates makes detecting physical failures straightforward. The controller cannot be incorrect for the nominal plant model. When an invariant is violated, the plant dynamics must have changed. The HAHACS identifies faults when continuous controllers violate discrete boundary conditions—a direct consequence of verified nominal control modes. Unexpected behavior implies off-nominal conditions.

The mathematical formulation for expulsory mode verification differs from transitory modes in two key ways. First, the entry conditions may be the entire state space (or a large, conservatively bounded region) rather than a well-defined entry set. The failure may occur at any point during operation. Second, the dynamics include parametric uncertainty representing failure modes:

$$\dot{x} = f(x, u, \theta), \quad \theta \in \Theta_{failure}$$

where $\Theta_{failure}$ captures the range of possible degraded plant behaviors identified through failure mode and effects analysis (FMEA) or traditional safety analysis.

We verify expulsory modes using reachability analysis with parametric uncertainty. The verification condition requires that for all parameter values within the uncertainty set, trajectories from the expanded entry region reach the safe shutdown state:

$$\forall \theta \in \Theta_{failure} : \text{Reach}(\mathscr{X}_{current}, f_\theta, [0,T]) \subseteq \mathscr{X}_{shutdown}$$

This is more conservative than nominal reachability, accounting for the fact that we cannot know exactly which failure mode is active.

Traditional safety analysis techniques inform the construction of $\Theta_{failure}$. Probabilistic risk assessment, FMEA, and design basis accident analysis identify credible failure scenarios and their effects on plant dynamics. The expulsory mode must handle the worst-case dynamics within this envelope. This is where conservative controller design is appropriate as safety margins will matter more than performance during emergency shutdown.

3.4 Industrial Implementation

The complete methodology—procedure formalization, discrete synthesis, and continuous verification across three mode types—provides a theoretical framework for hybrid control synthesis. But theory alone does not demonstrate practical feasibility. Advancing from analytical concepts (TRL 2-3) to laboratory demonstration (TRL 5) requires validation on realistic systems using industrial-grade hardware. This research will leverage the University of Pittsburgh Cyber Energy Center's partnership with Emerson to implement and test the HAHACS methodology on production control equipment. Emerson's Ovation distributed control system is widely deployed in power generation facilities, including nuclear plants. The Ovation platform provides a realistic target for demonstrating that formally synthesized controllers can execute on industrial hardware meeting timing and reliability requirements. The discrete automaton produced by reactive synthesis will be compiled to run on Ovation controllers, with verification that the implemented behavior matches the synthesized specification exactly.

For the continuous dynamics, we will use a small modular reactor simulation. The SmAHTR (Small modular Advanced High Temperature Reactor) model provides a relevant testbed for startup and shutdown procedures. The ARCADE (Advanced Reactor Control Architecture Development Environment) interface will establish communication between the Emerson Ovation hardware and the reactor simulation, enabling hardware-in-the-loop testing of the complete hybrid controller.

Working with Emerson on such an implementation is an incredible advantage for the success and impact of this work. We will directly address the gap of verification and validation methods for these systems and industry adoption by forming a two-way exchange of knowledge between the laboratory and commercial environments. This work stands to be successful with Emerson implementation because we will have access to system experts at Emerson to help with the fine details of using the Ovation system. At the same time, we will have the benefit of transferring technology directly to industry with a direct collaboration in this research, while getting an excellent perspective of how our research outcomes can align best with customer needs.

This section addressed two critical Heilmeier questions: What is new? Why will it succeed?

What is new? Three innovations enable compositional verification integrating reactive synthesis, reachability analysis, and barrier certificates:

Contract-based decomposition inverts traditional global analysis. Discrete synthesis defines verification contracts that bound continuous verification.

Mode classification enables mode-local analysis with provable composition guarantees by matching continuous modes to appropriate verification tools.

Procedure-driven structure leverages existing procedural decomposition, avoiding intractable state explosion.

Section 2 established that prior work verified discrete logic or continuous dynamics—never both compositionally. This compositional verification enables what global analysis cannot achieve.

Why will it succeed? Three factors ensure practical feasibility:

Existing structure. Nuclear procedures already decompose operations into discrete phases with explicit transition criteria, allowing formalization without artificial abstractions.

Bounded complexity. Mode-level verification bounds each verification problem locally, avoiding the state explosion that makes global hybrid system analysis intractable.

Industrial validation. Emerson collaboration provides domain expertise to validate procedure formalization and industrial hardware to demonstrate implementation feasibility, ensuring solutions address real deployment constraints.

The complete technical methodology is now established. Sections 2 and 3 addressed the first four Heilmeier questions: what has been done, what limits current practice, what is new, and why it will succeed. Three questions remain. Section 4 addresses how success will be measured. Section 5 identifies what could prevent success. Section 6 explains who cares, why now, and what difference this work will make.

4 Metrics for Success

Heilmeier Question: How will success be measured?

Section 3 established the technical approach: compositional verification bridges discrete synthesis with continuous control and will succeed because it leverages existing procedural structure, bounds computational complexity, and validates against industrial hardware. This section addresses the next Heilmeier question: How will success be measured?

Success is measured by Technology Readiness Level advancement from fundamental concepts (TRL 2–3) to validated prototype demonstration (TRL 5), where system components operate successfully in a relevant laboratory environment. TRL advancement provides the most appropriate success metric because it explicitly measures the gap between academic proof-of-concept and practical deployment. This section explains why TRLs measure success appropriately, then defines specific criteria for each level from TRL 3 through TRL 5.

Technology Readiness Levels provide the ideal success metric for work bridging academic proof-of-concept and practical deployment.

Academic metrics—papers published or theorems proved—fail to capture practical feasibility. Empirical metrics—simulation accuracy or computational speed—fail to demonstrate theoretical rigor. TRLs measure both.

Advancing from TRL 3 to TRL 5 requires maintaining theoretical rigor while progressively demonstrating practical feasibility. The system moves from individual components to integrated hardware testing. Two requirements constrain this progression. First: formal verification must remain valid throughout. Second: the proofs must compose as the system scales.

The nuclear industry requires extremely high assurance before deploying new control technologies. Demonstrating theoretical correctness alone proves insufficient for adoption; conversely, showing empirical performance without formal guarantees fails to meet regulatory requirements. TRLs capture this dual requirement naturally. Each level represents both increased practical maturity and sustained theoretical validity, while TRL assessment forces explicit identification of remaining barriers to deployment. The nuclear industry already uses TRLs for technology assessment, making this metric directly relevant to potential adopters. Reaching TRL 5 provides a clear answer to industry questions about feasibility and maturity that academic publications alone cannot.

Moving from current state to target requires achieving three intermediate levels, each representing a distinct validation milestone:

TRL 3 ***Critical Function and Proof of Concept*** For this research, TRL 3 means demonstrating that each component of the methodology works in isolation. Startup procedures must be translated into temporal logic specifications that pass realizability analysis. A discrete automaton must be synthesized with interpretable structure. At least one continuous controller must be designed with reachability analysis proving transition requirements are satisfied. Independent review must confirm that specifications match intended procedural behavior. This proves the fundamental approach on a simplified startup sequence.

TRL 4 ***Laboratory Testing of Integrated Components*** For this research, TRL 4 means demonstrating a complete integrated hybrid controller in simulation. All startup procedures must be formalized with a synthesized automaton covering all operational modes. Continuous controllers must exist for all discrete modes. Verification must be complete for all mode transitions using reachability analysis, barrier certificates, and assume-guarantee contracts. The integrated controller must execute complete startup sequences in software simulation with zero safety violations across multiple consecutive runs. This proves that formal correctness guarantees can be maintained throughout system integration.

TRL 5 ***Laboratory Testing in Relevant Environment*** For this research, TRL 5 means demonstrating the verified controller on industrial control hardware through hardware-in-the-loop testing. The discrete automaton must be implemented on the Emerson Ovation control system and verified to match synthesized specifications exactly. Continuous controllers must execute at required rates. The ARCADE interface must establish stable real-time communication between the Emerson Ovation hardware and SmAHTR simulation. Complete autonomous startup sequences must execute via hardware-in-the-loop across the full operational envelope. The controller must handle off-nominal scenarios to validate that expulsory modes function correctly. For example, simulated sensor failures must trigger appropriate fault detection and mode transitions, and loss-of-cooling scenarios must activate SCRAM procedures as specified. Graded responses to minor disturbances are outside this work's scope. Formal verification results must remain valid, with discrete behavior matching specifications and continuous trajectories remaining within verified bounds. This proves that the methodology produces verified controllers implementable on industrial hardware.

Progress will be assessed quarterly through collection of specific data comparing actual results against TRL advancement criteria. Specification development status indicates progress toward TRL 3. Synthesis results and verification coverage indicate progress toward TRL 4. Simulation performance metrics and hardware integration milestones indicate progress toward TRL 5. The research plan will be revised only when new data invalidates fundamental assumptions. This research succeeds by achieving TRL 5: demonstrating a complete autonomous hybrid controller with formal correctness guarantees operating on industrial control hardware through hardware-in-the-loop testing in a relevant laboratory environment. This establishes both theoretical validity and practical feasibility, proving the methodology produces verified controllers implementable with current technology.

This section answered the Heilmeier question: How will success be measured?

Answer: Technology Readiness Level advancement from 2–3 to 5. Each level demonstrates both theoretical correctness and practical feasibility through progressively integrated validation.

TRL 3 proves component-level correctness: each methodology element works independently.

TRL 4 demonstrates system-level integration in simulation: components compose correctly.

TRL 5 validates hardware implementation in a relevant environment: the complete system operates on industrial control hardware.

Achieving TRL 5 proves the methodology produces verified controllers implementable with current technology—not merely theoretically sound but practically deployable.

Sections 2 through 4 addressed five Heilmeier questions: what has been done, what limits current practice, what is new, why it will succeed, and how to measure success. Success assumes critical technical challenges can be overcome. Section 5 addresses what could prevent success.

5 Risks and Contingencies

Heilmeier Question: What could prevent success?

Section 4 defined success as reaching TRL 5 through component validation, system integration, and hardware demonstration—a definition that assumes critical technical challenges can be overcome.

Every research plan rests on assumptions that might prove false. Three primary risks could prevent reaching TRL 5: computational tractability of synthesis and verification, complexity of the discrete-continuous interface, and completeness of procedure formalization.

Each risk has identifiable early warning indicators and viable mitigation strategies.

Each risk carries associated early warning indicators and contingency plans that preserve research value even when core assumptions fail. The staged project structure ensures that partial success yields publishable results and clearly identifies remaining barriers to deployment, even when full success proves elusive.

5.1 Computational Tractability of Synthesis

Computational tractability represents the first major risk. The core assumption: formalized startup procedures will yield automata small enough for efficient synthesis and verification. This assumption may fail. Reactive synthesis scales exponentially with specification complexity. Temporal logic specifications from complete startup procedures may produce automata with thousands of states. Synthesis times may exceed days or weeks. This would prevent completion within project timelines. Reachability analysis for continuous modes with high-dimensional state spaces may similarly prove intractable. Either barrier would constitute a fundamental obstacle.

Several indicators would provide early warning of computational tractability problems. Synthesis times exceeding 24 hours for simplified procedure subsets would suggest complete procedures are intractable. Generated automata containing more than 1,000 discrete states would indicate the discrete state space is too large for efficient verification. Specifications flagged as unrealizable by FRET or Strix would reveal fundamental conflicts in the formalized procedures. Reachability analysis failing to converge within reasonable time bounds would show that continuous mode verification cannot be completed with available computational resources.

If computational tractability becomes the limiting factor, we reduce scope to a minimal viable startup sequence covering only cold shutdown to criticality to low-power hold. This scope reduction omits power ascension and other operational phases. The reduced sequence still demonstrates the complete methodology—procedure formalization, discrete synthesis, continuous verification, and hardware implementation—while reducing computational burden. The research contribution remains valid: we prove that formal hybrid control synthesis is achievable for safety-critical nuclear applications and clearly identify which operational complexities exceed current computational capabilities. We document the limitation as a scaling constraint requiring future work, not a methodological failure.

5.2 Discrete-Continuous Interface Formalization

The first risk addressed practical constraints: whether synthesis can complete within reasonable time bounds. The second risk proves more fundamental: whether boolean guard conditions in temporal logic can map cleanly to continuous state boundaries required for mode transitions.

This interface represents the fundamental challenge of hybrid systems: relating discrete switching logic to continuous dynamics. Temporal logic operates on boolean predicates, while continuous control requires reasoning about differential equations and reachable sets. Guard conditions requiring complex nonlinear predicates may resist boolean abstraction. This would make synthesis intractable. Continuous safety regions that cannot be expressed as conjunctions of verifiable constraints would similarly create insurmountable verification challenges.

The risk extends beyond static interface definition to dynamic behavior across transitions. Barrier certificates may fail to exist for proposed transitions. Continuous modes may be unable to guarantee convergence to discrete transition boundaries.

Early indicators of interface formalization problems would appear during both synthesis and verification phases. Guard conditions requiring complex nonlinear predicates that resist boolean abstraction would suggest fundamental misalignment between discrete specifications and continuous realities. Continuous safety regions that cannot be expressed as conjunctions of half-spaces or polynomial inequalities would indicate the interface between discrete guards and continuous invariants is too complex. Failure to construct barrier certificates proving safety across mode transitions would reveal that continuous dynamics cannot be formally related to discrete switching logic. Reachability analysis showing that continuous modes cannot reach intended transition boundaries from all possible initial conditions would demonstrate the synthesized discrete controller is incompatible with achievable continuous behavior.

The primary contingency for interface complexity is restricting continuous modes to operate within polytopic invariants. Polytopes are state regions defined as intersections of linear half-spaces, which map directly to boolean predicates through linear inequality checks. This restriction ensures tractable synthesis while maintaining theoretical rigor, though at the cost of limiting expressiveness compared to arbitrary nonlinear regions. The discrete-continuous interface remains well-defined and verifiable with polytopic restrictions, providing a clear fallback position that preserves the core methodology. Conservative over-approximations offer an alternative approach: a nonlinear safe region can be inner-approximated by a polytope, sacrificing operational flexibility to maintain formal guarantees. The three-mode classification already structures the problem to minimize complex transitions, with critical safety properties concentrated in expulsory modes that can receive additional design attention.

Mitigation strategies focus on designing continuous controllers with discrete transitions as primary objectives from the outset. Rather than designing continuous control laws independently and verifying transitions post-hoc, the approach uses transition requirements as design constraints. Control barrier functions provide a systematic method to synthesize controllers that guarantee forward invariance of safe sets and convergence to transition boundaries. This design-for-verification approach reduces the likelihood that interface complexity becomes insurmountable. Focusing verification effort on expulsory modes—where safety is most critical—allows more complex analysis to be applied selectively rather than uniformly across all modes, concentrating computational resources where they matter most for safety assurance.

5.3 Procedure Formalization Completeness

The previous two risks concerned verification infrastructure—computational scaling and mathematical formalization. The third risk addresses the source material itself: whether existing startup procedures contain sufficient detail and clarity for translation into temporal logic specifications. Nuclear operating procedures, while extensively detailed, were written for human operators who bring contextual understanding and adaptive reasoning to their interpretation. Procedures may contain implicit knowledge, ambiguous directives, or references to operator judgment that resist formalization in current specification languages. Underspecified timing constraints, ambiguous condition definitions, or gaps in operational coverage would cause synthesis to fail or produce incorrect automata. The risk is not merely that formalization is difficult, but that current procedures fundamentally lack the precision required for autonomous control, revealing a gap between human-oriented documentation and machine-executable specifications.

Several indicators would reveal formalization completeness problems early in the project. FRET realizability checks failing due to underspecified behaviors or conflicting requirements would indicate procedures do not form a complete specification. Multiple valid interpretations of procedural steps with no clear resolution would demonstrate procedure language is insufficiently precise for automated synthesis. Procedures referencing "operator judgment," "as appropriate," or similar discretionary language for critical decisions would explicitly identify points where human reasoning cannot be directly formalized. Domain experts unable to provide crisp answers to specification questions about edge cases would suggest the procedures themselves do not fully define system behavior, relying instead on operator training and experience to fill gaps.

The contingency plan treats inadequate specification as itself a research contribution rather than a project failure. Documenting specific ambiguities encountered would create a taxonomy of formalization barriers: timing underspecification, missing preconditions, discretionary actions, and undefined failure modes. Each category would be analyzed to understand why current procedure-writing practices produce these gaps and what specification languages would need to address them. Proposed extensions to FRETish or similar specification languages would demonstrate how to bridge the gap between current procedures and the precision needed for autonomous control. The research output would shift from "here is a complete autonomous controller" to "here is what formal autonomous control requires that current procedures do not provide, and here are language extensions to bridge that gap." This contribution remains valuable to both the nuclear industry and formal methods community, establishing clear requirements for next-generation procedure development and autonomous control specification languages.

Early-stage procedure analysis with domain experts provides the primary mitigation strategy. Collaboration through the University of Pittsburgh Cyber Energy Center enables identification and resolution of ambiguities before synthesis attempts, rather than discovering them during failed synthesis runs. Iterative refinement with reactor operators and control engineers can clarify procedural intent before formalization begins, reducing the risk of discovering insurmountable specification gaps late in the project. Comparison with procedures from multiple reactor designs—pressurized water reactors, boiling water reactors, and advanced designs—may reveal common patterns and standard ambiguities amenable to systematic resolution. This cross-design analysis would strengthen the generalizability of any proposed specification language extensions, ensuring they address industry-wide practices rather than specific quirks.

This section answered the Heilmeier question: What could prevent success?

Answer: Three primary risks threaten TRL 5 achievement: computational tractability of synthesis and verification, complexity of the discrete-continuous interface, and completeness of procedure formalization.

Each risk has identifiable early warning indicators enabling detection before failure becomes inevitable. Each has viable mitigation strategies preserving research value even when core assumptions fail.

The staged project structure ensures partial success yields publishable results identifying remaining deployment barriers. This design maintains contribution regardless of which technical obstacles prove insurmountable. Even ”failure” advances the field by documenting precisely which barriers remain.

Sections 2 through 5 established the complete technical research plan. Section 2 addressed what has been done and what limits current practice. Section 3 established what is new and why it will succeed. Section 4 defined how to measure success. This section identified what could prevent success and how to respond.

One critical question remains: Who cares? Why now? What difference will it make?

Section 6 connects this technical methodology to urgent economic and societal challenges.

6 Broader Impacts

Heilmeier Questions: Who cares? Why now? What difference will it make?

Sections 2 through 5 established the complete technical research plan. Section 2 addressed what has been done. Section 3 explained what is new and why it will succeed. Section 4 defined how to measure success. Section 5 identified what could prevent success.

The technical plan is complete. This section addresses the remaining Heilmeier questions. It connects technical methodology to economic and societal impact: who cares, why now, and what difference this work will make.

Who cares? Three stakeholder groups face the same economic constraint—high operating costs driven by staffing requirements. The nuclear industry faces uncompetitive per-megawatt costs for small modular reactors. Datacenter operators need hundreds of megawatts of continuous clean power for AI infrastructure. Clean energy advocates need nuclear power to be economically competitive with fossil alternatives. All three stakeholders require autonomous control with safety guarantees.

What difference will it make? This research directly addresses a \$21–28 billion annual cost barrier, enabling economically viable small modular reactors for datacenter power and establishing a generalizable framework for safety-critical autonomous systems across critical infrastructure.

Why now? Exponentially growing AI infrastructure demands have transformed this longstanding challenge into an immediate crisis, creating a market demanding solutions that did not exist before.

Nuclear power presents both a compelling application domain and an urgent economic challenge. Recent interest in powering artificial intelligence infrastructure has renewed focus on small modular reactors (SMRs), particularly for hyperscale datacenters requiring hundreds of megawatts of continuous power. SMRs deployed at datacenter sites minimize transmission losses and eliminate emissions. At this scale, nuclear power economics demand careful attention to operating costs.

The U.S. Energy Information Administration's Annual Energy Outlook 2022 projects advanced nuclear power entering service in 2027 will cost \$88.24 per megawatt-hour [**?**]. Datacenter electricity demand is projected to reach 1,050 terawatt-hours annually by 2030 [**?**]. Nuclear power supplying this demand would generate total annual costs exceeding \$92 billion.

Operations and maintenance represents a substantial component. The EIA estimates fixed O&M costs alone account for \$16.15 per megawatt-hour. Additional variable O&M costs are embedded in fuel and operating expenses [**?**]. Combined, O&M-related costs represent approximately 23–30% of total levelized cost. This translates to \$21–28 billion annually for projected datacenter demand.

What difference will it make? This research directly addresses the \$21–28 billion annual O&M cost barrier. High-assurance autonomous control makes small modular reactors economically viable for datacenter power while maintaining nuclear safety standards. Beyond immediate economic impact, the methodology establishes a generalizable framework for safety-critical autonomous systems across critical infrastructure.

Current nuclear operations require full control room staffing for each reactor—whether large conventional units or small modular designs. For large reactors producing 1,000+ MW, staffing costs spread across substantial output. Small modular reactors producing 50-300 MW face the same staffing requirements with far lower output. This makes per-megawatt costs prohibitive. These staffing requirements drive the economic challenge threatening SMR deployment for datacenter applications.

Synthesizing provably correct hybrid controllers from formal specifications automates routine operational sequences that currently require constant human oversight. This enables a fundamental shift from direct operator control to supervisory monitoring. Operators oversee multiple autonomous reactors rather than manually controlling individual units.

The correct-by-construction methodology proves critical for this transition. Traditional automation approaches cannot provide sufficient safety guarantees for nuclear applications, where regulatory requirements and public safety concerns demand the highest levels of assurance. By formally verifying both the discrete mode-switching logic and the continuous control behavior, this research produces controllers with mathematical proofs of correctness. These guarantees enable automation to safely handle routine operations—startup sequences, power level changes, and normal operational transitions—that currently require human operators to follow written procedures. Operators will remain in supervisory roles to handle off-normal conditions and provide authorization for major operational changes, but the routine cognitive burden of procedure execution shifts to provably correct automated systems that are much cheaper to operate.

SMRs represent an ideal deployment target for this technology. Nuclear Regulatory Commission certification requires extensive documentation of control procedures, operational requirements, and safety analyses written in structured natural language. As described in our approach, these regulatory documents can be translated into temporal logic specifications using tools like FRET, then synthesized into discrete switching logic using reactive synthesis tools, and finally verified using reachability analysis and barrier certificates for the continuous control modes. The infrastructure of requirements and specifications already exists as part of the licensing process, creating a direct pathway from existing regulatory documentation to formally verified autonomous controllers.

Beyond reducing operating costs for new reactors, this research will establish a generalizable framework for autonomous control of safety-critical systems. The methodology of translating operational procedures into formal specifications, synthesizing discrete switching logic, and verifying continuous mode behavior applies to any hybrid system with documented operational requirements. Potential applications include chemical process control, aerospace systems, and autonomous transportation, where similar economic and safety considerations favor increased autonomy with provable correctness guarantees. Demonstrating this approach in nuclear power—one of the most regulated and safety-critical domains—will establish both the technical feasibility and regulatory pathway for broader adoption across critical infrastructure.

This section answered three critical Heilmeier questions:

Who cares? Three stakeholder groups face the same constraint. The nuclear industry faces an economic crisis for small modular reactors due to per-megawatt staffing costs. Datacenter operators need hundreds of megawatts of continuous clean power for AI infrastructure. Clean energy advocates need nuclear power to be economically competitive. All three require autonomous control with safety guarantees.

Why now? Two forces converge to create urgency. *First: exponentially growing de-*

mand. AI infrastructure creates immediate need for economical nuclear power at datacenter scale. Projections show datacenter electricity demand reaching 1,050 terawatt-hours annually by 2030. *Second: technical maturity.* Formal methods tools have matured sufficiently to make compositional hybrid verification computationally achievable. What was theoretically possible but practically intractable a decade ago is now feasible. The problem is urgent. The tools exist.

What difference will it make? This research addresses a $21–28 billion annual cost barrier and enables autonomous control with mathematical safety guarantees. Beyond immediate economic impact, the methodology establishes a generalizable framework for safety-critical autonomous systems across critical infrastructure, extending beyond nuclear power to any safety-critical system requiring provable correctness.

Sections 2 through 6 addressed all but one of the Heilmeier questions. Section 2 established what has been done and what limits current practice. Section 3 explained what is new and why it will succeed. Section 4 defined how to measure success. Section 5 identified what could prevent success. This section connected technical methodology to economic and societal impact. One final Heilmeier question remains: How long will it take? Section 8 provides the answer.

7 Schedule, Milestones, and Deliverables

Heilmeier Question: How long will it take?

Section 6 demonstrated that this work addresses a $21–28 billion annual cost barrier and establishes a generalizable framework for safety-critical autonomous systems. This final section addresses the last Heilmeier question: How long will it take?

This research will be conducted over six trimesters (24 months) of full-time effort following the proposal defense in Spring 2026. The University of Pittsburgh Cyber Energy Center and NRC Fellowship provide all computational and experimental resources. The work progresses sequentially through three main research thrusts, culminating in integrated demonstration and validation.

The first semester (Spring 2026) focuses on Thrust 1, translating startup procedures into formal temporal logic specifications using FRET. This establishes the foundation for automated synthesis by converting natural language procedures into machine-readable requirements. The second semester (Summer 2026) addresses Thrust 2, using Strix to synthesize the discrete automaton that defines mode-switching behavior. With the discrete structure established, the third semester (Fall 2026) develops the continuous controllers for each operational mode through Thrust 3, employing reachability analysis and barrier certificates to verify that each mode satisfies its transition requirements. Integration and validation occupy the remaining three semesters.

Figure **??** shows the complete project schedule including research thrusts, major milestones, and planned publications.

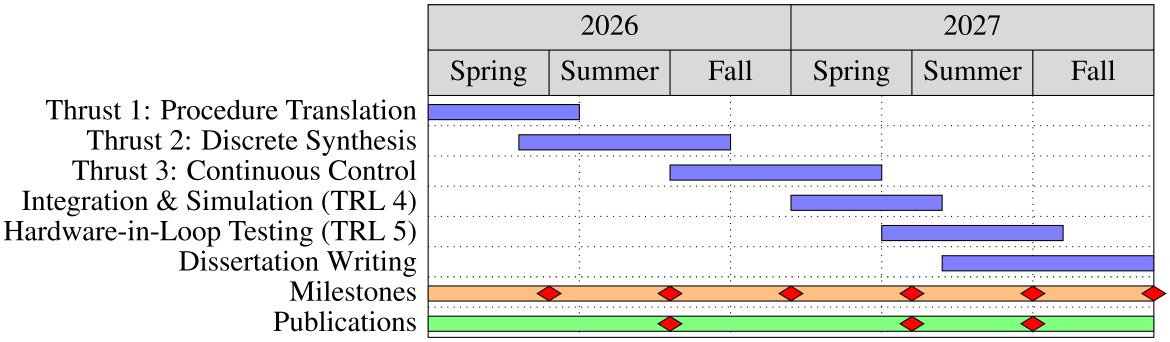

Figure 3: Project schedule showing major research thrusts, milestones (orange row), and publications (green row). Red diamonds indicate completion points. Overlapping bars indicate parallel work where appropriate.

7.1 Milestones and Deliverables

Six major milestones mark critical validation points throughout the research:

M1 (Month 4): Confirms that startup procedures have been successfully translated to temporal logic using FRET with realizability analysis demonstrating consistent and complete specifications.

M2 (Month 8): Validates computational tractability by demonstrating that Strix can synthesize a complete discrete automaton from the formalized specifications. Delivers a conference paper submission to NPIC&HMIT documenting the procedure-to-specification translation methodology.

M3 (Month 12): Achieves TRL 3 by proving that continuous controllers can be designed and verified to satisfy discrete transition requirements. Delivers an internal technical report demonstrating component-level verification.

M4 (Month 16): Achieves TRL 4 through integrated simulation demonstrating that component-level correctness composes to system-level correctness. Delivers a journal paper submission to IEEE Transactions on Automatic Control presenting the complete hybrid synthesis methodology.

M5 (Month 20): Achieves TRL 5 by demonstrating practical implementability on industrial hardware. Delivers a conference paper submission to NPIC&HMIT or CDC documenting hardware implementation and experimental validation.

M6 (Month 24): Completes the dissertation documenting the entire methodology, experimental results, and research contributions.

References

[1] U.S. Nuclear Regulatory Commission, "Guidelines for the preparation of emergency operating procedures," Tech. Rep. NUREG-0899, U.S. Nuclear Regulatory Commission, 1982.

[2] U.S. Nuclear Regulatory Commission, "10 CFR Part 50.34." Code of Federal Regulations.

[3] U.S. Nuclear Regulatory Commission, "10 CFR Part 55.59." Code of Federal Regulations.

[4] "Westinghouse RPS System Description," tech. rep., Westinghouse Electric Corporation.

[5] C. D. Gentillon, D. Marksberry, D. Rasmuson, M. B. Calley, S. A. Eide, and T. Wierman, "Westinghouse reactor protection system unavailability, 1984-1995." Number: INEEL/CON-99-00374 Publisher: Idaho National Engineering and Environmental Laboratory.

[6] U.S. Nuclear Regulatory Commission, "Operator Licensing." `https://www.nrc.gov/reactors/operator-licensing`.

[7] U.S. Nuclear Regulatory Commission, "Part 55—Operators' Licenses." `https://www.nrc.gov/reading-rm/doc-collections/cfr/part055/full-text`.

[8] U.S. Nuclear Regulatory Commission, "§ 50.54 Conditions of Licenses." `https://www.nrc.gov/reading-rm/doc-collections/cfr/part050/part050-0054`.

[9] J. G. Kemeny *et al.*, "Report of the president's commission on the accident at three mile island," tech. rep., President's Commission on the Accident at Three Mile Island, October 1979.

[10] World Nuclear Association, "Safety of nuclear power reactors." `https://www.world-nuclear.org/information-library/safety-and-security/safety-of-plants/safety-of-nuclear-power-reactors.aspx`, 2020.

[11] L. Högberg, "Root causes and impacts of severe accidents at large nuclear power plants," vol. 42, no. 3, pp. 267–284.

[12] M. Zhang, L. Dai, W. Chen, and E. Pang, "Analysis of human errors in nuclear power plant event reports," vol. 57, no. 10, p. 103687.

[13] J. Kiniry, A. Bakst, S. Hansen, M. Podhradsky, and A. Bivin, "High assurance rigorous digital engineering for nuclear safety (hardens) final technical report," Tech. Rep. TLR-RES-RES/DE-2024-005, Galois, Inc. / U.S. Nuclear Regulatory Commission, 2024. NRC Contract 31310021C0014.

[14] U.S. Energy Information Administration, "Levelized costs of new generation resources in the annual energy outlook 2022," report, U.S. Energy Information Administration, March 2022. See Table 1b, page 9.

[15] Environmental and Energy Study Institute, "Data center energy needs are upending power grids and threatening the climate." Web article, 2024. Accessed: 2025-09-29.