



Smart Transformer for Enhanced Efficiency and Sustainability in Maritime Transportation

Deliverable 4.2– Report on Model System Draft

Saber Sakhrieh, Christos Kamizoulis, Mehdi Zadeh, and Roger Skjetne

Norwegian University of Science and Technology (NTNU), Department of Marine Technology

Project: 101192740 — STEESMAT — HORIZON-CL5-2024-D5-01

Start date: 2025-01-01

Duration: 42 months

Project coordinator: Maritime CleanTech

Coordinator contact: Erlend Bertheussen, erlend@maritimecleantech.no

Document information

Project Acronym	STEESMAT
Project No.	101192740
Title	Report on Model System Draft
Deliverable no.	D4.2
Draft delivery data	30-12-2025
Dissemination level	PU - Public
Deliverable type	R — Document, report
Authors	Saber Sakhrieh, Christos Kamizoulis, Mehdi Zadeh, and Roger Skjetne

Document history:

Version	Date of issue	Content and changes	Edited by
Revision 0	30.12.2025	First version	Saber Sakhrieh (main writer), Christos Kamizoulis (circuit diagrams), Roger Skjetne (review).

Executive summary

Deliverable abstract	
This report serves as the first deliverable for Task 4.3 in Work Package 4 (WP4) of the STEESMAT project. It specifies the numerical models and the software architecture of STEESMAT's transparent Python-based system-level simulator of the hybrid ship power system. The report defines the unit-model equations, interfaces, configuration schema, and integration patterns that will underpin the implementation targeted for D4.4 (Repository with simulation model at system level). The model spans the three-stage Solid State Transformer (SST), DC/DC converters, transformer, DC Hub, Battery Energy Storage System (BESS), and Fuel-Cell (FC). Moreover, it includes detailed information about representative loads for the Energy House, the demonstration vessel model, and a scaled-up system. All component limits, operating modes (grid-forming/following), and start–stop sequences are encoded as design constraints and state-machine specifications. The simulator is organized as modular, plug-and-play units with scenario files and an adaptor layer to couple external operational-load models. The report also states modeling assumptions, intended use, and out-of-scope aspects at this stage, defines verification checks for power balance, and lays out the implementation and validation plan toward the development of the Python-based system-level simulation model.	

Contents

Executive summary	3
Contents	4
Abbreviations	6
1 Introduction	7
1.1 Objective	7
1.2 Purpose and intended use	8
1.2.1 Workflow for simulator-based testing	9
1.3 Scope	9
2 Background and System Context	10
2.1 STEESMAT architecture overview	10
2.1.1 Simulator considerations	12
2.2 Reference single-line diagrams and power flow philosophy	13
2.2.1 Power flow modeling philosophy	15
3 Modeling Assumptions	15
3.1 Electrical abstraction	16
3.2 Time base	16
3.3 Environment	17
3.4 Data availability	17
4 Design Specifications	18
4.1 Nomenclature and per-unit conventions	18
4.1.1 Control and supervisory symbols	19
4.2 Solid-State Transformer (SST)	20
4.2.1 MVAC/MVDC AFE converter	20
4.2.2 MVDC/LVDC DAB converter	24
4.2.3 LVDC/LVAC inverter	26
4.3 LVDC grid	27
4.3.1 DC hub and non-isolated LVDC converter (BESS interface)	27
4.3.2 Galvanically isolated LVDC converter (Fuel-Cell interface)	30
4.4 Fuel-cell system	32
4.5 Battery energy storage system	35

4.6	Transformer.....	38
4.7	Loads	40
4.7.1	Propulsion electric load	40
4.7.2	Operational load emulator	42
4.7.3	Hotel/Aux load model	43
5	System Composition and Operational Modes.....	44
5.1	Bus topology.....	44
5.2	Operational modes	44
5.2.1	Simulator state machine, guards, and fault handling.....	46
6	Simulator Architecture and Interfaces.....	47
6.1	Package layout.....	48
6.2	Configurability.....	49
6.3	Module I/O contracts	49
6.4	Extensibility hooks	51
7	Integration with Energy and Emission Management System	52
8	Validation and Verification Strategy.....	53
9	Conclusion	55
	References	57
	Appendix	58

Abbreviations

AFE	Active Front-End	MPC	Model Predictive Control
AI	Artificial Intelligence	MVAC	Medium-Voltage Alternating Current
API	Application Programming Interface	MVDC	Medium-Voltage Direct Current
BoL	Beginning of Life	NTNU	Norwegian University of Science and Technology
BoP	Balance of Plant	NPC	Neutral-Point Clamped
CPP	Controllable Pitch Propeller	OCV	Open-Circuit Voltage
DAB	Dual-Active Bridge	ODCA	Open Direct Current Alliance
DAN	Danfoss Drives	PCC	Point of Common Coupling
DP	Dynamic Programming	PF	Power Factor
EATON	Eaton	PI	Proportional–Integral
EBL	Electronic Bus-Link	PID	Proportional–Integral–Derivative
EC	Equivalent Circuit	PLL	Phase-Locked Loop
EEMS	Energy & Emissions Management System	PMS	Power Management System
EFC	Equivalent Full Cycles	p.u.	Per-Unit
EM	Energy Management	RB	Rule-Based
EMI	Electromagnetic Interference	RL	Reinforcement Learning
EoL	End of Life	RMS	Root-Mean-Square
FPP	Fixed Pitch Propeller	SE	Sustainable Energy AS
FC	Fuel-Cell	SFC	Specific Fuel Consumption
GFM	Grid-Forming Mode	SLD	Single-Line Diagram
GFL	Grid-Following Mode	SiC	Silicon Carbide
GHG	Greenhouse Gas	SoC	State of Charge
ISOP	Input-Series Output-Parallel	SoH	State of Health
JSON	JavaScript Object Notation	SST	Solid-State Transformer
KPI	Key Performance Indicator	SWITCH	The Switch
LFT	Low-Frequency Transformer	THDi	Total Harmonic Distortion of current
LHV	Lower Heating Value	TRL	Technology Readiness Level
LVAC	Low-Voltage Alternating Current	VSD	Variable Speed Drive
LVDC	Low-Voltage Direct Current	WAER	Wärtsilä Norway AS
LUT	Lappeenranta–Lahti University of Technology	WP	Work Package
ML	Machine Learning	YAML	YAML Ain't Markup Language
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor		

1 Introduction

1.1 Objective

This report sets out the design framework for a transparent Python-based medium fidelity system-level simulator of hybrid ship power systems studied in STEESMAT. The report formalizes the framework for the numerical unit models, their interfaces and boundaries, the overall system composition, and the software organization that will guide implementation and validation toward the development of the system-level simulator. The simulator targets studies of smart power and energy management algorithms, and the resulting power flow and conversion losses, energy and fuel consumption, and emissions under representative operating scenarios. Its design prioritizes configurability by declaring scenarios as data and having the ability to swap or resize modules without code changes. Also, it enforces transparency through documented governing equations, units, and clear sign conventions for every interface. Moreover, it ensures extensibility by exposing clean hooks for supervisory control and for coupling to external models. The numerical models and simulator serve as a simulation-based learning platform for understanding ship power-system integration and operation. Moreover, they function as a testbed for prototyping and evaluating the performance of optimization algorithms within the STEESMAT layered control architecture. Also, the models can be used as a generator of synthetic operational datasets to conceptually support AI workflows in forecasting and optimization that will later inform smart management algorithms. Finally, they operate as a configurable environment for verification and scale-up analysis to assess different vessels and parameter sets prior to hardware deployment. The Python system-level simulator specified in this report is referred to as *mcpowersimpy*.

The STEESMAT system is a maritime MVDC-LVDC architecture built around three integrated pillars: a Solid State Transformer (SST) based power-electronic system that converts MVAC to LVDC, a secondary LVDC grid integrating BESS and FC, and a smart management and control layer that optimizes fuel and energy use, reduces emissions, and interfaces with existing power-management systems through developed communication interfaces. The SST implements a three-stage conversion sequence: an MVAC/MVDC Active Front-End (AFE) that inverts 3.3 kV AC to 5.0 kV DC, an MVDC/LVDC isolated Dual-Active Bridge (DAB) that regulates the LVDC bus at 800 V, and an LVDC/LVAC inverter to supply local AC loads. The LVDC grid includes a DC Hub for Energy Storage System (ESS) integration and both isolated and non-isolated LVDC converters, selected to ensure operational safety requirements for the FC and BESS. Moreover, STEESMAT integrates a smart management and control system that aims for energy and emission consumption. Building on that, the simulator represents the core electrical structure relevant for system-level studies: power-electronic conversion stages, LVDC grid with interfaces for BESS and FC subsystems, and representative ship-load models (propulsion and hotel). Rather than reproducing switching-level behavior, the simulator focuses on control-relevant dynamics, operational limits, sequencing, and loss accounting, enabling repeatable evaluation of supervisory strategies and their impact on system KPIs.

To support testing and comparison across reference cases, *mcpowersimpy* incorporates configurations through versioned YAML/JSON scenario templates that define buses, unit modules, interconnections, and parameter sets. This enables rapid assembly of testbeds that mirror the Energy House and demonstration-vessel configurations, as well as scale-up scenarios for vessel archetypes with different

ratings and load partitions. An adapter is included to supply time-stamped demanded load powers from external operational-load models. The simulator reports standardized outputs and KPIs, including bus power balance, converter losses, energy and fuel use, emissions factors, and constraint violations. The overall goal is to support learning, algorithm prototyping, verification, and scale-up analysis before and alongside hardware testing.

1.2 Purpose and intended use

The numerical modules and system-level simulator are developed to serve as a multi-purpose environment that supports the following objectives:

1. Simulation learning platform:
 - It should provide a transparent Python-based environment in which users can assemble reference configurations and trace signal flows across relevant power system architectures.
 - It should support guided exploration of operating modes, start–stop sequences, interlocks, and stability margins.
 - It is intended to be built with ready-to-use templates for the relevant testbeds and use cases, enabling comparative studies under consistent assumptions.
2. Algorithm prototyping:
 - The goal is to offer a sandbox for developing and testing the performance of the optimization methods within the proposed control hierarchy (mission, guidance, and control layers).
 - It should enable closed-loop tests under sufficiently realistic constraints, including converter voltage/current limits, battery SoC bounds, FC power and temperature limits with ramp-rate restrictions.
 - It should provide clean interfaces for supervisory control, supporting both baseline rule-based strategies as well as more advanced optimization-based methods.
3. Synthetic data and AI training:
 - If fidelity is deemed high enough, it may generate labeled time series under controlled mission profiles with configurable disturbances and measurement noise.
 - A future goal is for it to support forecasting and optimization workflows, where the resulting datasets are structured for subsequent integration into smart algorithms.
 - A clear intention is to design the load emulation module to enable interface with relevant hydrodynamic models such as *mcsimpy*¹ (Gezer et al., 2025) which originally was developed as the Python package *MCSimPython* by (Hygen, 2023) and (Mo et al., 2023). This interface may be used either via offline load profile import or online co-simulation to reflect vessel-specific operational load patterns.

¹ The *mcsimpy* open-source can be accessed at <https://doi.org/10.5281/zenodo.17274093>

4. Verification and scale-up analysis:

- It should provide a configurable environment for comparing grid architectures, component sizing (batteries, FCs, and converters ratings), and power-sharing strategies. The simulator-based test workflow and acceptance criteria are summarized in Section 1.2.1 and formalized in Chapter 8.
- It should allow parameter sweeps and sensitivity studies, incl. quantification of key performance indicators, including power flow, conversion losses, total energy and fuel use, emission factors, and constraint violations.

To make the intended use actionable and verifiable, the following section summarizes how the simulator will be used as a structured testing environment, from unit checks to system integration and closed-loop PMS/EEMS evaluation. Detailed architecture, interfaces, and verification thresholds are specified later in Chapters 6–8.

1.2.1 Workflow for simulator-based testing

The simulator is designed for analysis and to serve as a testing environment for supervisory control (PMS/EEMS interaction) and for verifying configuration and component sizing before hardware deployment. The same scenario definitions (SLDs, ratings, limits, operating timelines) used in simulation are reused for testbed work to ensure traceability and comparability across experimental testing and real vessel trials.

Testing levels targeted by the future simulator (not all of which may be fulfilled in STEESMAT):

1. Unit-model checks (component level): each unit is validated for sign conventions, limit enforcement, and non-negative loss accounting under representative operating points.
2. Integration checks (system level): for a given scenario SLD, the simulator verifies bus-level power conservation, DC-link regulation behavior, and correct execution of start/stop sequences and interlocks.
3. Controller/supervisory tests (closed-loop): the simulator supports baseline versus optimized runs using the same loads and constraints, enabling repeatable comparisons of KPIs (energy, losses, fuel/H₂, emissions, constraint violations).
4. Coupled operational-load testing: propulsion and hotel demand can be driven by external models to test robustness under mission-relevant varying load profiles.

The simulator could also support a staged deployment philosophy where supervisory setpoints can be evaluated in shadow/advisory mode, then in limited authority under PMS-enforced envelopes, reflecting the intended validation path from lab to vessel trials.

1.3 Scope

The scope of the first version of the simulator is limited to what is sufficient to support the STEESMAT project studies and needs. Additional capabilities described as potential “use areas” will be considered as features for future extensions. This framing is consistent with the role of this report as a planning and specification report that highlights the later implementation work.

In scope for the first version:

The first simulator version is intended to be a control-oriented, medium-fidelity, averaged representation of the STEESMAT electrical architecture, suitable for repeatable studies of supervisory operation. Concretely, it should include:

- System-level architecture and composition aligned with the STEESMAT reference testbeds and scale-up configurations, expressed via scenario files and modular unit composition.
- Averaged component and converter models capturing the dominant behaviors needed for PMS/EEMS interaction: limits, ramps, efficiencies/loss accounting, and power conservation across buses.
- Operational modes and supervisory sequencing, represented at the functional/state-machine level to support closed-loop supervisory testing.
- Functional testing of typically EEMS functions and algorithms.
- Verification outputs suitable for STEESMAT studies: bus power-balance checks, constraint violations, and KPI reporting for energy use, conversion losses, fuel/H₂ use, and emissions factors under scenario-defined missions and load profiles.

Out of scope for the first version:

The simulator is not intended to replace detailed electrical design and power-quality tools. In particular, the following are out of scope at this stage:

- Detailed electrical system analysis, including short-circuit and fault transient analysis, protection selectivity studies, grounding/earthing studies, insulation coordination, and detailed network impedance modeling beyond simple R/L equivalents.
- High-frequency power electronics phenomena, including switching transients, harmonic distortion, and fast electromagnetic oscillations that require EMT-type simulation fidelity.
- Detailed thermal networks and physics-based aging models, beyond simplified derating hooks or bookkeeping variables where needed for constraints.
- Simulation and testing of faults and failures through modelling and implementation of failure modes at component and subsystem level.
- Full-fidelity vessel hydrodynamics and propulsion physics as a primary modeling objective. Where needed, operational load effects are represented through simplified or externally provided load profiles, rather than detailed onboard mechanical modeling.

2 Background and System Context

This chapter summarizes the STEESMAT electrical architecture at a functional level, links *mcpowersimpy* simulator to the reference single-line diagrams, and defines the load-driven power-flow philosophy used throughout the simulation loop.

2.1 STEESMAT architecture overview

The SST is the enabling element of STEESMAT's MVDC architecture, providing regulated and galvanically isolated conversion while simplifying the integration of distributed resources such as batteries and fuel-cells. The configuration follows a three-stage chain:

1. **SST first stage:** An MVAC/MVDC AFE converter that is designed to transform 3.3 kV AC to 5 kV DC.
2. **SST second stage:** An MVDC/LVDC DAB converter that regulates the LVDC bus at 0.8 kV and forms the core of the SST. It will be implemented as modular power-electronic building blocks in an Input-Series and Output-Parallel (ISOP) arrangement using SiC devices and a medium-frequency (MF) transformer to achieve high efficiency, bidirectional power flow, and reliable high-to-low voltage conversion.
3. **SST third stage:** an LVDC/LVAC inverter to supply LVAC loads connected to the system. Here, the project novelty lies in control strategies rather than new hardware development.

The overall SST targets a high efficiency with lower power losses and substantial weight and volume reductions compared to conventional ferromagnetic transformers. To connect the SST to AC generation, a three-phase transformer with a 3.3 kV primary and 400–440 V secondary, operating at 50–60 Hz, is planned to be used in STEESMAT.

The LVDC Hub (0.8 kV) serves as the central distribution node, interconnecting the SST with BESS and FC via reliable DC/DC converters. The LVDC Hub will include a non-isolated LVDC converter to connect the SST to the BESS interface. Moreover, an innovative galvanically isolated LVDC converter will be designed to support connecting the SST with the FC interface. The FC integrated in the Energy House is a PowerCell PS185 proton-exchange membrane fuel-cell (PEMFC) rated at 185 kW (Sustainable Energy, 2021). The Energy House also contains a marine BESS from Corvus Energy. The general specifications of the STEESMAT system components are outlined in Table 1.

Table 1: Specifications of STEESMAT system Components.

Component	Role in the system	Partner
SST-1: MVAC→MVDC converter	Converts 3.3 kV AC to 5 kV DC	DAN, LUT
SST-2: MVDC→LVDC converter	Regulates LVDC bus at 0.8 kV	EATON, UTBM
SST-3: LVDC→LVAC inverter	Supplies LVAC hotel/aux and AC machinery	SWITCH
LVDC Hub and Non-isolated LVDC converter (BESS Interface)	Enables bidirectional power exchange between the 0.8 kV DC bus and the BESS	SWITCH
Isolated LVDC converter (FC Interface)	Ensures voltage compatibility and galvanic isolation between the 0.8 kV DC bus and the FC	DAN
Fuel-Cell system	Low-emission generation; mid/high steady-load supply	SE
Battery Energy Storage System	DC-link support, transients, peak shaving, reserve	WAER
Step-up transformer	Step up AC generation to 3.3 kV AC for SST	SE

The STEESMAT project has the ambition to develop a smart energy and emission management system (EEMS). This system will be integrated with state-of-the-art power-management systems (PMS) to enable prediction and optimization of energy consumption. The system supports both real-time and predictive control of shipboard energy flows, enhancing efficiency while preserving fault tolerance. The architecture follows a mission–guidance–control hierarchy to set operating modes, perform predictive coordination of energy sources, and execute real-time setpoints, respectively. *mcpowersimpy* simulator will be explicitly structured to demonstrate this hierarchy.

2.1.1 Simulator considerations

The *mcpowersimpy* simulator is implemented in Python as a modular system with physics-based and data-driven unit models (power converters, BESS, FC, bus connection, and loads) with a configurable system-level composition. Its primary objectives are simulation-based studies of power flow and conversion losses, energy and fuel consumption, and emissions across relevant use cases, with standardized KPIs to enable fair comparison of system responses. The modeling scope comprises component-level behavior with constraints and efficiencies, bus-level composition and operating modes, and mission-level load demand. Default parameter sets that seed the model are defined based on STEESMAT components ratings and testbed scenarios. Operating modes are defined by state-transition logic with explicit start/stop order, readiness checks, and role allocation among the AFE, DAB, BESS, and FC. These sequences become verification targets via checks on power conservation across buses, and bounds on SoC, voltages, currents, and losses. Moreover, *mcpowersimpy* will be explicitly structured to demonstrate the smart mission–guidance–control hierarchy. Each layer will be specified by state-transition logic with measurable inputs/outputs, allowing controlled closed-loop experiments and standardized data logging. As a result, the environment will be able to generate high-quality datasets to support the implementation of optimization and learning-based energy management (EM) strategies, including labeled trajectories under varied missions, disturbances, and constraints.

To ensure consistent terminology and control-relevant relationships, *mcpowersimpy* adopts a unified notation and a common set of device constraints. The simulator natively supports per-unit (p.u.) system with bus-specific base quantities, enabling normalized voltages, currents, and powers for better numerical conditioning and cross-scenario comparison. Bases are defined per scenario, and conversions to and from SI units are handled automatically in I/O and KPI reporting. These conventions support the KPI framework defined above, keeping results comparable across scenarios and vessel archetypes. Where mechanical and hydrodynamic effects shape electrical demand, the simulator couples to external ship load models, where realistic sealoads and propulsion dynamics are reflected in operational profiles used for control and optimization studies. Moreover, the simulator follows established power-system simulation design: modular unit models with uniform I/O contracts, bus-level power aggregation, and supervisory logic that supports end-to-end operation. This philosophy underpins the abstractions and interfaces used here and facilitates integration with PMS/EEMS, while preserving transparency and reproducibility for later validation activities.

2.2 Reference single-line diagrams and power flow philosophy

To support reproducible studies across the testbeds, demonstration vessels, and scale-up analysis use cases, *mcpowersimp*y simulator is anchored to a set of reference single-line diagrams (SLDs). These SLDs define the electrical topology at system level.

Figure 1 represents a specified testbed installation with an 800 V LVDC bus connected to an MVDC backbone at 5 kV and interfaced to AC supplies via transformers. The SLD also illustrates the placement of key assets at the system level, including the 185 kW FC and 1 MWh BESS feeding the LVDC bus. A proposed scale-up configuration is to update the SLD shown in Figure 1 by including an MVAC generator set feeding the MVAC/MVDC conversion stage directly instead of using a step-up transformer. Moreover, the scale-up topology includes connecting main propellers to the primary 5 kV MVDC grid. This SLD is used as the main reference for scalability studies of power routing, loss accounting, and supervisory coordination across voltage domains.

Nominal component ratings and control roles are specified in the simulator for testbeds and scale-up scenarios to ensure forward applicability. These definitions provide the bus voltages, roles, and device responsibilities that the simulator reproduces as default scenarios. To avoid code changes in the simulator when moving between different use cases, scenario templates are combined with parameter overlays (e.g., updated ratings, number of modules, alternative load timelines), while the module I/O contracts and bus power-balance logic remain unchanged. This ensures that differences in results are traceable to differences in topology and configuration rather than differences in implementation.

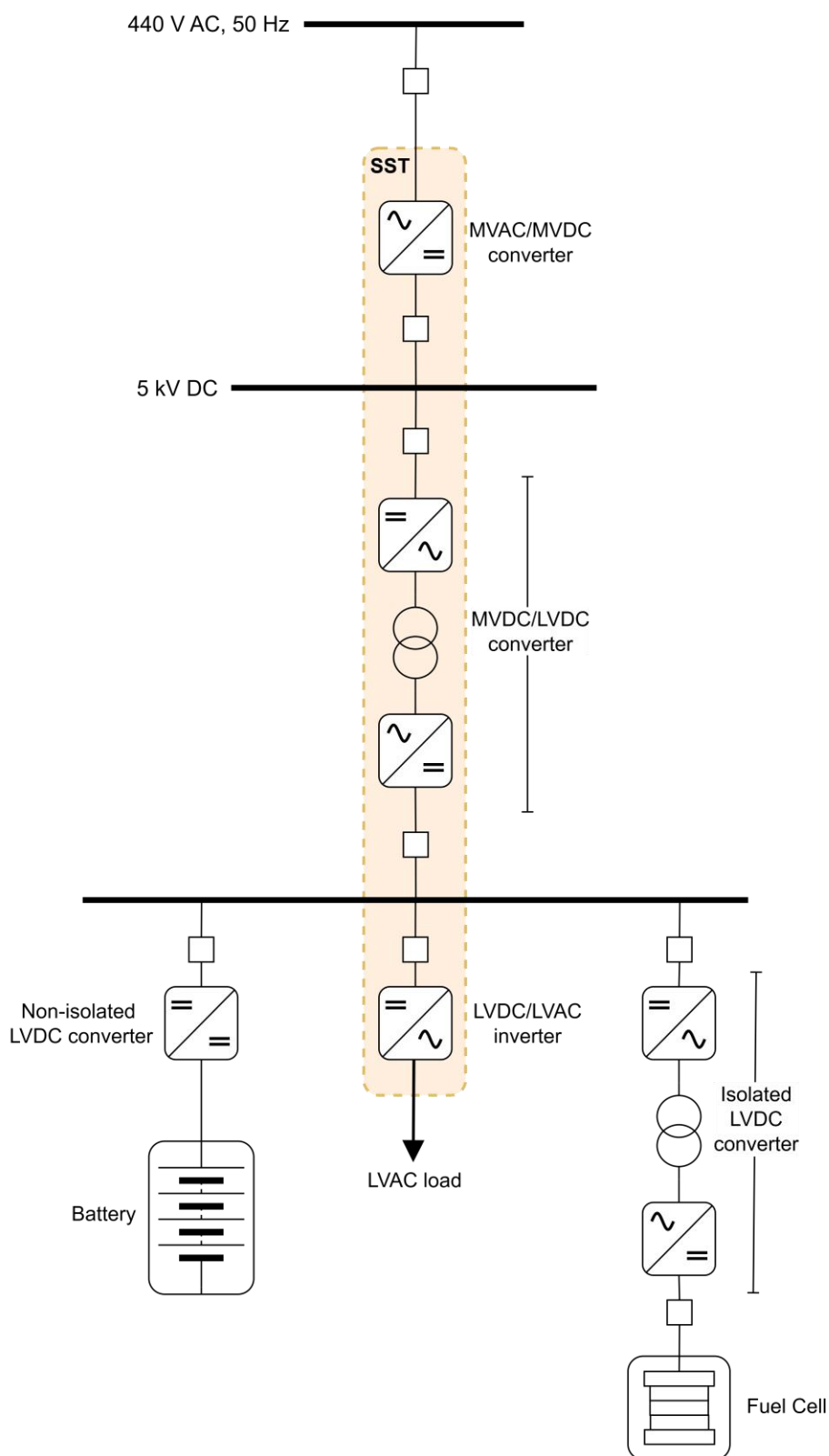


Figure 1: STEESMAT specified testbed SLD representation.

2.2.1 Power flow modeling philosophy

mcpowersimpy follows a load-driven power-flow philosophy: at each time step, the system first determines the electrical demand of the emulated ship loads, and then computes the upstream dispatch required to supply this demand subject to device limits, efficiencies, and operating modes. This is consistent with the simulator's use as a testing environment, where realistic mission profiles (propulsion and hotel) are provided and the supervisory control (PMS/EEMS) responds by allocating setpoints to sources and converters.

Concretely, each step proceeds conceptually as:

1. Load synthesis (consumption): Obtain propulsion power $P_{\text{prop}}(t)$ and hotel power $P_{\text{hotel}}(t)$, and map them to the relevant electrical buses.
2. Bus aggregation: Compute per-bus net demand/injection using the sign convention and power-balance relations, including losses.
3. Dispatch (supply): PMS/EEMS allocate source and interface setpoints (e.g., FC and BESS power, MVDC/LVDC transfer power, DC-link targets) so that the demand is met (satisfying power balance) while respecting ramp limits, SoC/voltage/current constraints, and operating mode ownership (GFM/GFL). If the total available generation/transfer capability under active constraints and operating mode ownership cannot satisfy the dictated demand, the simulator resolves the step by introducing a controlled demand shortfall variable (unserved load) and/or load shedding variables with explicit priorities. This ensures the simulation remains numerically well-posed and produces diagnostic outputs that are useful for PMS/EEMS evaluation.
4. Energy accounting: Report Energy flows from prime sources → conversion stages → loads with cumulative losses, enabling consistent KPI computation and traceable comparison across scenarios.
5. Fuel consumption: Compute fuel/ H_2 use from the dispatched power and efficiency models (e.g., FC efficiency map), then integrate over time to obtain cumulative consumption.
6. Emissions: Convert fuel use (or source power) to emissions using source-specific factors and report instantaneous and cumulative emissions as KPIs.

3 Modeling Assumptions

This chapter states the cross-cutting assumptions used to specify and validate the numerical models and their composition. The intent is to keep unit-level specifications in Chapter 4 focused and avoid repeating shared choices. The *mcpowersimpy* simulator is a system-level, control-oriented (averaged) model intended for studies of power flow, conversion losses, energy/fuel use, emissions, and supervisory management/control under realistic operational profiles. Power-electronic stages are represented as continuous-time averaged blocks at the control time scale (switching ripple, harmonics, and EMI are out of scope). The philosophy is to keep models transparent and reproducible while retaining the dominant behaviors required for PMS/EEMS interaction: limits, ramps, start/stop sequencing, and power conservation across buses. *mcpowersimpy* supports p.u. representations with bus-specific bases to improve numerical conditioning and comparability across scenarios.

3.1 Electrical abstraction

1. Averaged converter models:
 - Power-electronic stages (AFE, DAB, LV inverter, LVDC DC/DCs) are modeled as continuous-time averaged blocks at the control time scale. Switching ripple and Electromagnetic interference (EMI) are not explicitly simulated.
 - Stage efficiencies are represented by efficiency ($\eta(P,V,I)$) maps or parametric curves. Switching losses are lumped and applied as heat sinks in the energy balance (electrical only—no detailed thermal network).
 - Galvanic isolation is explicit where present (DAB, FC DC/DC). Non-isolated BESS interface is modeled with ideal coupling plus limits and losses.
2. Buses and flows:
 - Three electrical domains are assumed: MVAC (3.3 kV), MVDC (5 kV), and LVDC (0.8 kV). With an optional LVAC branch from the inverter.
 - Sign convention: positive power flows from sources to buses and from buses to loads. By default, bus aggregation is the algebraic sum of unit terminal powers minus stage losses.
 - Per-unit support: each bus declares (V_{base}, S_{base}). Model I/O can be expressed in p.u. or SI, with automatic conversion at interfaces and in KPI reporting.
3. Controls and modes:
 - Grid-forming/grid-following behavior is captured at the functional level (voltage- or current-controlled sources with synchronization and ramp limits).
4. Component constraints:
 - BESS: SoC dynamics, SoC/C-rate/voltage/temperature bounds, and ramp-rate limits.
 - FC: Power/current/voltage/temperature limits, hydrogen supply/purge constraints, warm-up/cool-down, startup/shutdown, and ramp-rate limits; BoP consumption may be lumped into net efficiency if detailed signals are unavailable.
 - Converters: Current/voltage limits and controller saturation.
 - Loads: Propulsion and hotel/aux loads are treated as demanded powers with optional dynamics.
5. Out of scope (for the simulation model at system level, delivered in D4.4):
 - Harmonic distortion, parasitic capacitances, detailed switching transients, electric oscillation phenomena, and cable/network impedance modeling beyond simple R/L equivalents.
 - Detailed thermal networks, aging/degradation physics.
 - Protection selectivity studies and arc/short-circuit transient simulation.

However, such detailed electric phenomena may be included in models coming out of more specialized studies under WP4.

3.2 Time base

1. Sampling and solver:

- Default simulation step $\Delta t = 1 - 10 \text{ ms}$ for electrical and control dynamics. Slower processes (e.g., SoC, temperature) are updated at integer multiples of Δt .
- Discretization, will consider use of forward Euler, Tustin's method, or Runge-Kutta methods based on required integration accuracy vs. computational speed.
- All time stamps are absolute (UTC in logs) with scenario-defined mission timelines for mode changes, load steps, and disturbances.

2. Latency and ramping:

- Measurement and actuation latencies are modeled as configurable pure delays or first-order lags.
- Converter and unit ramp-rate limits are enforced in setpoint tracking; violations are logged as KPI events.

3.3 Environment

1. Operating conditions:

- Ambient temperature and cooling state are scenario parameters affecting allowable limits but not considered modeled via detailed thermal dynamics. Simplified methods to be considered if relevant.
- Sea state, wind, and current are represented through operational-load profiles coming from a load emulation module that also may be interfaced to external models.
- External operational load models (e.g., vessel mechanical/hydrodynamic simulators) can be coupled to provide demanded power partitions.

2. Disturbances and noise:

- Sensor noise will typically be modeled as zero-mean additive stochastic processes with configurable variance and seed.
- Relevant faults and random events are not a prime objective to simulate in the simulator version developed for STEESMAT (but may be targeted for future versions). However, particularly relevant failure modes may be modelled and simulated in a dedicated manner.

3.4 Data availability

The simulator data are intended to adhere to the following standards:

- Default parameters (ratings, bus levels, limits, efficiency maps) are seeded from project-internal specifications and test assets (testbeds and demonstration/scaleup use cases). When unavailable, parameters are flagged for later updates.
- Configurations are declarative (YAML/JSON) and versioned. They define systems, unit selections, parameter sets, and mission timelines.
- The simulator logs standardized KPIs, at minimum including power balance, conversion losses per stage, energy and fuel (H_2) use, emissions factors, constraint violations, and cycling metrics.
- Results can be exported in p.u. or SI, with base values recorded for each bus to ensure consistent post-processing.

4 Design Specifications

This chapter defines the numerical unit models, their interfaces, control modes, limits, and efficiency representations. All models are averaged at the control time scale (no switching ripple) and support per-unit representation. Each subsection follows a consistent pattern: *role & scope* → *interfaces (I/O)* → *control modes* → *equations/abstractions* → *parameters & limits* → *efficiency/loss model* → *numerical scheme* → *validation/status*.

4.1 Nomenclature and per-unit conventions

Symbol style:

- Upper case (V) typically means the variable in real SI units.
- Lower case (v) is used for per-unit (p.u.) representation of the variable.
- A hat ($\hat{V}, \hat{v}$) is used for RMS values.
- A tilde ($\tilde{V}, \tilde{v}$) indicates a complex phasor.

Bus bases and p.u. mapping:

$S_{base,b}$	Three-phase apparent power base on bus b [VA].
$V_{base,b}$	Base voltage on bus b (L-L RMS for AC; DC link voltage for DC) [V].
$I_{base,b} = \frac{S_{base,b}}{\sqrt{3}V_{base,b}}$	AC base current on bus b [A].
$I_{base,b} = \frac{S_{base,b}}{V_{base,b}}$	DC base current on bus b [A].
$Z_{base,b} = \frac{V_{base,b}^2}{S_{base,b}}$	Base impedance on bus b [Ω].
$\omega_{base,b} = 2\pi f_{base,b}$	Base electrical angular frequency at AC bus [rad/s].
$C_{base,b} = \frac{S_{base,b}}{\omega_{base,b} V_{base,b}^2}$	Base capacitance for DC buses
$v = \frac{v}{v_{base}}$	Per-unit voltage [p. u.].
$i = \frac{I}{I_{base}}$	Per-unit current [p. u.].
$c = \frac{C}{C_{base,b}}$	Per-unit capacitance [p. u.].
$p = \frac{P}{S_{base}}$	Per-unit active power [p. u.].
$q = \frac{Q}{S_{base}}$	Per-unit reactive power [p. u.].
$r_i = \frac{R_i}{Z_{base}}$	Per-unit resistance [p. u.].
$x_i = \frac{\omega_{base} L_i}{Z_{base}}$	Per-unit reactance [p. u.].

Note that for each electrical bus $b \in \{\text{MVAC}, \text{MVDC}, \text{LVDC}, \text{LVAC}\}$.

4.1.1 Control and supervisory symbols

$(\cdot)^*$	Reference/setpoint.
Δt	Simulation/control time step.
$\tilde{V}_{abc}, \tilde{v}_{abc}$	Three-phase AC phasor at a bus [V_{rms}] and [p.u.].
$\hat{V}_{ll}, \hat{v}_{ll}$	RMS magnitude; line-to-line RMS [V_{rms}] and [p.u.].
V_d^g, V_q^g	dq components of grid/PCC voltage [V].
I_d, I_q	dq components of PCC current [A].
v_d, v_q	Per-unit dq components of bus voltage
i_d, i_q	Per-unit dq components of bus current
P, Q	Three-phase active/reactive power at PCC [W], [var].
V_{dc}, v_{dc}	DC-link voltage [V] and [p.u.].
I_{dc}, i_{dc}	DC current [A] and [p.u.].
C_{dc}, c_{dc}	DC-link capacitance [F] and [p.u.].
R_{Σ}, r_{Σ}	PCC series resistance [Ω] and [p.u.].
L_{Σ}	PCC series inductance [H].
x_{Σ}	Per-unit PCC series reactance.
R_v, r_v	Virtual resistance used in GFM [Ω] and [p.u.].
L_v	Virtual inductance used in GFM [H].
x_v	Per-unit virtual reactance.
ω_s	dq synchronous angular frequency [rad/s].
ω_0	nominal angular frequency [rad/s].
ω^*	Commanded frequency.
θ	dq electrical angle [rad].
f	Electrical frequency [Hz].
$P_{\text{loss}}, p_{\text{loss}}$	Power losses [W] and [p.u.].
η	Efficiency.
i_{pre}	Precharge/bleed current.
p_{down}	Total downstream DC power draw.
α	Precharge threshold for RUN (fraction of v_{dc}^*).
$r_{\text{bleed}, dc}$	DC bleeder resistance.
$k_{p,v}, k_{i,v}$	Voltage-loop PI gains.

$k_{p,i}, k_{i,i}$	Current-loop PI gains.
k_{ω}	Frequency-droop gain [$\text{rad} \cdot \text{s}^{-1}$ per p.u. power].
i_{max}	Current magnitude limit.
ρ_P, ρ_Q	Active/reactive ramp limits [p.u./s].
$\text{sat}(\cdot)$	Saturation / window limiting.
$\text{rl}(\cdot, \rho)$	Rate limiter with ramp ρ .
$\text{aw}(\cdot)$	Anti-windup term/clamp.

4.2 Solid-State Transformer (SST)

4.2.1 MVAC/MVDC AFE converter

Role & scope:

The AFE rectifies the MVAC (nominal $\hat{V}_{mvac} = 3.3 \text{ kV}$) to a regulated MVDC link (nominal $V_{mvdc} = 5.2 \text{ kV}$) and can operate in grid-following (GFL) or grid-forming (GFM) behavior at a functional level. Detailed harmonic performance, PWM, and EMI are out of scope. Figure 2 shows the equivalent circuit that can be used to represent the AFE converter.

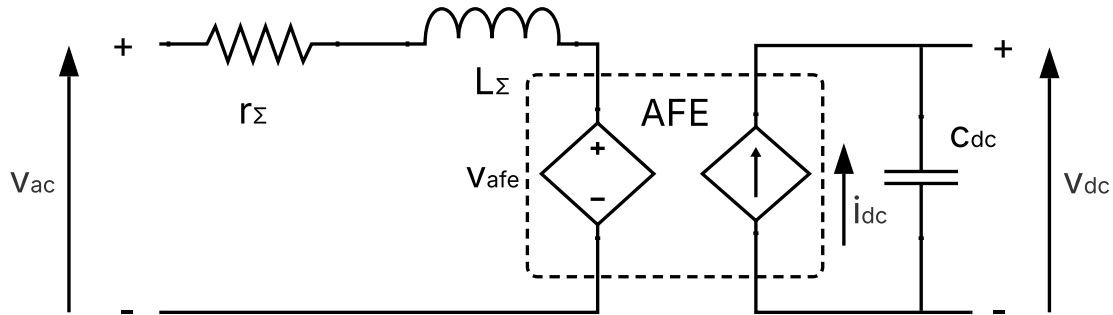


Figure 2: Averaged electrical equivalent circuit of the AFE converter.

Interfaces (I/O):

- Inputs: MVAC line-to-line voltage $\hat{v}_{mvac}$, frequency f , phase reference/sync signal, active/reactive setpoints (p^*, q^*) or DC-link voltage setpoint v_{mvdc}^* , enable/READY flags.
- Outputs: MVDC link voltage v_{mvdc} , p, q at MVAC, converter currents, loss power p_{loss}^{afe} , status/faults.

Modeling approach:

An averaged dq model at the electrical/control time scale is going to be used (no switching ripple). The grid can be represented in its synchronous frame. In grid-following, the PLL aligns the d-axis with the grid voltage ($v_q^g = 0$). In grid-forming, an internal oscillator and voltage loop can replace the PLL. The

DC link is a single energy store shared with downstream converters. Losses are applied through an $\eta(p)$ or p_{loss} relation consistent with the converter datasheet.

- Grid/transformer interface:

When fed via a dedicated transformer, the lumped line inductance used in the control model is

$$L_{\Sigma} = L_s + L_{xf} = \frac{V_{ac}^2}{2\pi f S_{sc}} + \frac{V_{ac}^2 X_s}{2\pi f S_{xf}}$$

where L_s is the source inductance, L_{xf} represents any input inductance at the AFE, $\hat{V}_{ac}$ is the MVAC voltage, S_{sc} is grid short-circuit power, S_{xf} is transformer base power and X_s is the relative leakage.

If needed, estimate L_s from the PCC short-circuit power $S_{sc,pcc}$ (on the MV secondary):

$$L_s \approx \frac{\hat{V}_{ac}^2}{2\pi f S_{sc,pcc}} \rightarrow x_s = \frac{\omega_{base} L_s}{Z_{base}}$$

Plant & power relations:

The grid side will be represented as an averaged dq model in the synchronous frame aligned with the grid voltage (PLL when grid-following and internal oscillator when grid-forming). At the Grid-side (per phase, lumped):

$$\begin{aligned} \frac{di_d}{dt} &= \frac{\omega_{base}}{x_{\Sigma}} (v_d^g - r_{\Sigma} i_d - v_d^{afe}) + \omega_s i_q \\ \frac{di_q}{dt} &= \frac{\omega_{base}}{x_{\Sigma}} (v_q^g - r_{\Sigma} i_q - v_q^{afe}) - \omega_s i_d \end{aligned}$$

where v_d^g, v_q^g are the PCC dq voltages, i_d, i_q are PCC dq currents, v_d^{afe}, v_q^{afe} are the AFE dq voltage commands, ω_s is the synchronous electrical frequency of the dq frame, and r_{Σ}, x_{Σ} are the PCC series resistance and reactance (excluding transformer leakage). Note that $v_q^g = 0$ under a standard PLL.

Three-phase power can be represented as

$$\begin{aligned} p_{ac} &= \frac{3}{2} (v_d^g i_d + v_q^g i_q) \\ q_{ac} &= \frac{3}{2} (v_q^g i_d - v_d^g i_q) \end{aligned}$$

where p_{ac} and q_{ac} are the three-phase active and reactive power at the PCC in per-unit.

The DC current delivered to MVDC link can be represented by

$$i_{dc}^{afe} = \frac{(\eta_{afe} * p_{ac})}{v_{dc}}$$

where i_{dc}^{afe} is the AFE DC current delivered to the MVDC link, v_{dc} is the MVDC link voltage, and $\eta_{afe} \in (0,1]$ is the operating-point efficiency of the AFE.

A DC-link capacitor may be represented (optional) by

$$c_{dc} \frac{dv_{dc}}{dt} = i_{dc}^{afe} - \frac{p_{down}}{v_{dc}} - i_{pre}$$

where c_{dc} is the dimensionless DC-link capacitance, p_{down} is the total per-unit draw to downstream DC units, and i_{pre} is the pre-charge/bleed current used only in PRECHARGE/DISCHARGE states.

Here $i_{dc}^{downstream}$ aggregates the DAB/LV side draw; i_{pre} models pre-charge/bleed events (zero in normal run).

Control structure:

- Grid-following mode (GFL):

Outer DC-voltage loop (PI) sets an active-current reference; reactive current is set by PF/Q objectives:

$$p^* = k_{p,v}(v_{dc}^* - v_{dc}) + k_{i,v} \int (v_{dc}^* - v_{dc}) dt + p_{ff}$$

$$i_d^* = \frac{2}{3} \frac{p^*}{v_d^g}$$

$$i_q^* = i_{q,PF^*} \quad (e.g., 0 \text{ for } PF = 1)$$

where p^* is the active-power reference, v_{dc}^* is the DC-voltage setpoint, p_{ff} is a power feed-forward term, and i_{q,PF^*} sets PF. Inner current PI with decoupling tracks (i_d^*, i_q^*) and generates $v_{d,q}^{AFE}$:

$$v_d^{afe} = k_{p,i}(i_d^* - i_d) + k_{i,i} \int (i_d^* - i_d) dt - \omega_s \left(\frac{x_\Sigma}{\omega_{base}} \right) i_q + r_\Sigma i_d + v_d^g$$

$$v_q^{afe} = k_{p,i}(i_q^* - i_q) + k_{i,i} \int (i_q^* - i_q) dt + \omega_s \left(\frac{x_\Sigma}{\omega_{base}} \right) i_d + r_\Sigma i_q + v_q^g$$

where $k_{p,i}, k_{i,i}$ are the inner-loop PI gains, and the cross-terms implement dq decoupling via $x_\Sigma/\omega_{base} = L_\Sigma/Z_{base}$ in p.u.

- Grid-forming mode (GFM):

The PLL can be replaced with a voltage source loop (voltage synthesis + virtual impedance + P- ω droop):

$$v_d^* = \hat{v}_{ac}^* + k_{p,v}(\hat{v}_{ac}^* - v_d^g) + k_{i,v} \int (\hat{v}_{ac}^* - v_d^g) dt - r_v i_d + \omega_s \left(\frac{x_v}{\omega_{base}} \right) i_q$$

$$v_q^* = -r_v i_q - \omega_s \left(\frac{x_v}{\omega_{base}} \right) i_d$$

$$\frac{d\theta}{dt} = \omega^*, \quad \omega^* = \omega_0 + k_\omega(p^* - p)$$

where $\hat{v}_{ac}^*$ is the per-unit RMS voltage reference, r_v, x_v are the virtual resistance/reactance, θ is the dq electrical angle, ω^* is the commanded frequency, ω_0 is the nominal electrical frequency, and k_ω is the frequency-droop gain.

Loss/efficiency model:

Power loss can be fitted using a parametric loss curve

$$p_{loss}^{afe}(p) = a_0 + a_1 |p_{mvac}| + a_2 p_{mvac}^2$$

$$\eta_{afe}(p) = 1 - \frac{p_{loss}^{afe}(p)}{|p_{mvac}|}$$

where $p_{loss,afe}$ is the AFE loss model, a_0, a_1 and a_2 are fitted constants, and η_{afe} is the resulting efficiency. The coefficients a_0, a_1 and a_2 will be chosen to match the supplier efficiency curves.

Then, the AC/DC side powers satisfy:

$$p_{mvdc} = p_{mvac} - p_{loss}^{afe}, \quad p_{loss}^{afe} \geq 0$$

where p_{mvac} be the AC-side active power absorbed by the AFE and p_{mvdc} the DC-side power delivered to the MVDC link.

Discrete-time form:

For a fixed step Δt (e.g., 1–10 ms):

$$i_d[k+1] = i_d[k] + \Delta t \left[\left(\frac{\omega_{base}}{x_{total}} \right) (v_d^g - r_{total} i_d - v_d^{afe}) + \omega_s i_q \right]$$

$$i_q[k+1] = i_q[k] + \Delta t \left[\left(\frac{\omega_{base}}{x_{total}} \right) (v_q^g - r_{total} i_q - v_q^{afe}) - \omega_s i_d \right]$$

$$v_{dc}[k+1] = v_{dc}[k] + \Delta t \left(\frac{1}{c_{dc}} \right) \left(i_{dc}^{afe} - \frac{p_{down}}{v_{dc}} - i_{pre} \right)$$

PI integrators use clamped updates for anti-windup:

$$\xi[k+1] = \text{clamp}(\xi[k] + \Delta t e[k], \xi_{min}, \xi_{max})$$

Parameters:

- MVAC base voltage and apparent power: $V_{base,mvac}, S_{base,mvac}$.
- MVDC base voltage and apparent power: $V_{base,mvdc}, S_{base,mvdc}$.
- Series impedance: r_Σ, x_Σ from PCC short-circuit level and any line reactor (excluding transformer leakage).
- Controller gains: $k_{p,v}, k_{i,v}, k_{p,i}, k_{i,i}$ tuned per scenario.
- Limits: $i_{max}, \rho_P, \rho_Q, v_{mvdc}^{min}, v_{mvdc}^{max}$.
- Loss fit: a_0, a_1, a_2 (refined as efficiency data become available).

4.2.2 MVDC/LVDC DAB converter

Role & scope:

The MVDC/LVDC stage is a modular DAB operated in Input-Series Output-Parallel (ISOP) configuration to step down the 5 kV DC to 0.8 kV DC with galvanic isolation and bidirectional power flow. It is represented as an averaged two-port power converter controlled by a requested transfer power (or an equivalent control variable such as phase shift). Figure 3 shows the equivalent circuit that can be used to represent the DAB converter.

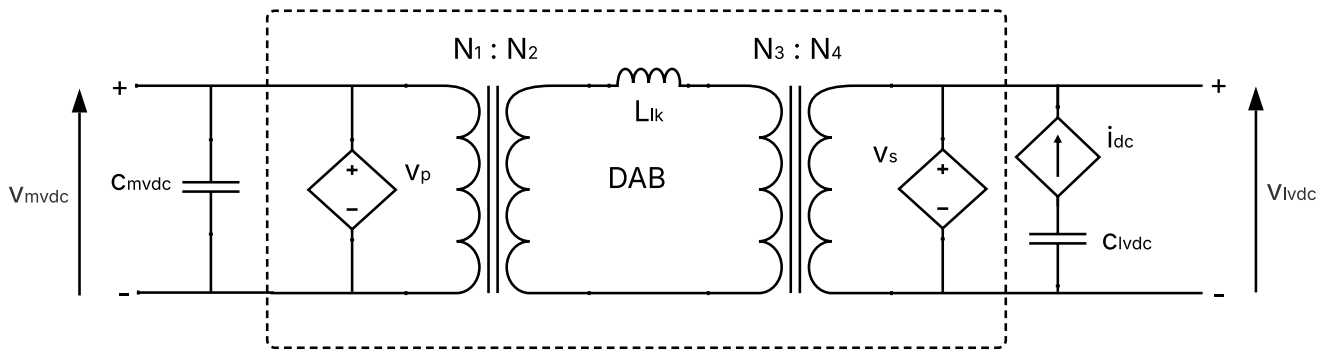


Figure 3: Averaged electrical equivalent circuit of the DAB converter.

Interfaces (I/O):

- Inputs: MV port voltage v_{mv} , LV port voltage v_{lv} , set-points v_{lv}^* and p_{dab}^* , enable/READY flags.
- Outputs: MV port current i_{mv}^{dab} , LV port current i_{lv}^{dab} , terminal power p_{lvdc} , converter power loss $p_{loss,dab}$, efficiency η_{dab} , status/faults.
- States: phase shift δ and LV bus v_{lv} capacitor state (if modeled explicitly).

Plant & power relations:

Let v_{mv} and v_{lv} be the port DC voltages at the MV and LV sides of the stack. The module turns ratio is $n = N_1/N_2$ (MV:LV). Define the normalized leakage $x_\ell = \omega_{base} L_\ell / Z_{base}$ for the equivalent series leakage referred to one side (sum of bridge + MF transformer leakage).

With bridge square-wave modulation and phase shift $\delta \in [-\pi, \pi]$, the averaged active power from MV to LV can be modeled as

$$p_{dab} = k_l v_{mv} v_{lv} \phi(\delta), \quad \phi(\delta) = \delta \left(1 - \frac{|\delta|}{\pi} \right)$$

where p_{dab} is the converter internal transferred power (positive MV→LV), $k_l = \kappa / (x_l)$ is a gain inversely proportional to the normalized leakage x_ℓ , and κ depends on the modulation waveform (e.g., $\kappa = \frac{1}{\pi}$ for a common square-wave approximation). The precise κ will be identified from converter tests and may differ slightly from the textbook value. Furthermore, $\phi(\delta)$ represents the standard square-wave phase-shift power shape. The sign of δ sets flow direction, positive MV→LV or negative LV→MV.

Let i_{mv}^{dab} be the MV-link current drawn from the MVDC bus by the DAB (positive when MV→LV), and i_{lv}^{dab} be the current injected into the LV bus:

$$i_{mv}^{dab} = \frac{p_{dab} + p_{loss,dab}}{v_{mv}}, \quad i_{lv}^{dab} = \frac{\eta_{dab} p_{dab}}{v_{lv}}$$

where p_{loss} is the DAB loss, η_{dab} represents its efficiency, i_{mv}^{dab} loads the MVDC link, and i_{lv}^{dab} supplies the LVDC link (signs invert automatically when $\delta < 0$).

To balance the LVDC bus energy, let c_{lv} represent the LV bus capacitor:

$$c_{lv} \frac{dv_{lv}}{dt} = i_{lv}^{dab} - \frac{p_{down,lv}}{v_{lv}}$$

where $p_{down,lv}$ is the total LV-side DC draw (e.g., BESS DC/DC, FC DC/DC) in p.u.

Control modes:

(A) LV-bus voltage regulation (default):

Outer PI on v_{lv} generates a power (or phase-shift) command:

$$p_{dab}^* = k_{p,v}^{lv}(v_{lv}^* - v_{lv}) + k_{i,v}^{lv} \int (v_{lv}^* - v_{lv}) dt + p_{ff}^{lv}, \quad \delta^* = \phi^{-1} \left(\frac{p_{dab}^*}{k_{\ell} v_{mv} v_{lv}} \right)$$

where p_{dab}^* is the requested transfer power from the LVDC outer loop, v_{lv}^* is the LVDC setpoint, $k_{p,v}^{lv}, k_{i,v}^{lv}$ are PI gains, p_{ff}^{lv} is an optional feed-forward term (e.g., predicted LV load), and ϕ^{-1} clips to $[-\pi, \pi]$ with $|\delta| \leq \delta_{max}$.

(B) Power control (dispatch mode):

The supervisory layer may command net power p^* (e.g., to achieve FC/BESS sharing), using the same δ^* map as above.

(C) Standby/idle:

$\delta \rightarrow 0$ and gate-signals minimized; the LV bus is supported by other grid-forming sources

Loss & efficiency model:

A smooth loss map consistent with the converter efficiency curve can be fitted. Let p_{dab} be the ideal transferred power from the phase-shift law (before losses). The loss is modeled as a quadratic in p_{dab} and the efficiency follows from power balance:

$$p_{loss,dab} = b_0 + b_1 |p_{dab}| + b_2 p_{dab}^2$$

$$\eta_{dab}(p) = \frac{p_{dab}}{p_{dab} + p_{loss,dab}}$$

where $p_{loss,dab}$ is the DAB loss model, b_0, b_1 and b_2 are fitted constants, and η_{afe} is the resulting efficiency. The coefficients b_0, b_1 and b_2 will be chosen to match the supplier efficiency curves.

Operating limits & interlocks:

- Phase-shift limit: $|\delta| \leq \delta_{max} < \pi$.
- Module current limit: $i_{lv,k} \leq i_{max,k}$, stack limit $\rightarrow \sum i_{lv,k} \leq i_{max,stack}$.
- Voltage windows: $v_{mv,min} \leq v_{mv} \leq v_{mv,max}$, $v_{lv,min} \leq v_{lv} \leq v_{lv,max}$.
- Ramp limits: $|d\delta/dt| \leq \rho_\delta$, $|dp/dt| \leq \rho_p^{dab}$.

Discrete-time implementation:

With a fixed step Δt (e.g., 1–10 ms) and controller saturations:

- Command mapping & saturation:

$$p_{dab}^* \leftarrow PI(v_{lv}^* - v_{lv}) + p_{ff}^{lv}$$

$$\delta^* = clip\left(\phi^{-1}\left(\frac{p_{dab}^*}{k_\ell v_{mv} v_{lv}}\right), -\delta_{max}, +\delta_{max}\right)$$

$$\delta[k+1] = \delta[k] + \left(\frac{\Delta t}{\tau_\delta}\right)(\delta^* - \delta[k]) \quad (\text{firstorder lag to enforce } |d\delta/dt| \leq \rho_\delta)$$

- LV bus update (if C_{lv} modeled):

$$v_{lv}[k+1] = v_{lv}[k] + \left(\frac{\Delta t}{C_{lv}}\right)\left(i_{lv}^{dab} - \frac{p_{down,lv}}{v_{lv}}\right)$$

Note that module-level sharing states (e.g., MV voltage offsets and LV current equalizers) are simple first-order lags and omitted here for brevity (they can be included if the modules are simulated explicitly).

Parameters:

- Bases: $V_{base,mvdc}$, $V_{base,lvdc}$, $P_{dab,base}$.
- Leakage x_ℓ : identified from MF-transformer/bridge design or tests.
- Efficiency fit: b_0 , b_1 , and b_2 fitted to the converter efficiency curve.
- Controllers: $k_{p,v}^{lv}$, $k_{i,v}^{lv}$ (outer LV-voltage PI), ρ_δ , ρ_p^{dab} (ramps).
- Limits/windows: δ_{max} , $i_{max,k}$, $v_{mv,min}$, $v_{mv,max}$, $v_{lv,min}$, $v_{lv,max}$.

4.2.3 LVDC/LVAC inverter

Role and scope:

The inverter interfaces LVDC to an LVAC branch. It is modeled as an averaged stage with functional GFM/GFL behavior and explicit limits.

Interfaces (I/O):

- Inputs: V_{hdc} , LVAC measurements, LVAC real and reactive power setpoints p_{lvac}^* , q_{lvac}^* , LVAC voltage and frequency setpoints $\hat{V}^*$, ω^* , ENABLE/READY, ramps.
- Outputs: LVAC terminal real and reactive power p_{lvac} , q_{lvac} , LVDC injection p_{hdc} , inverter losses $p_{loss,inv}$, status flags.

Functional modes:

GFL: track p_{lvac}^* , q_{lvac}^* with first-order/PI dynamics and saturation.

GFM: provide voltage/frequency setpoints with droop:

$$\omega = \omega_0 - k_\omega p_{ac}$$

$$\hat{V} = \hat{V}_0 - k_V q_{ac}$$

where ω_0 and V_0 are nominal (no-load) frequency and voltage setpoints used as droop intercepts, k_ω and k_V are frequency and voltage droop gains.

Power balance and losses:

$$p_{hdc} = -p_{hvac} - p_{loss,inv}$$

$$p_{loss,inv} = f_{inv}(p_{hvac}, V_{hdc}; \theta_{inv}), \quad p_{loss,inv} \geq 0.$$

where f_{inv} is a generic loss function (parametric curve or lookup map) used to compute $p_{loss,inv}$ from an operating point.

4.3 LVDC grid

The LVDC grid operates at a nominal 0.8 kV and forms the secondary DC bus that couples the SST's LV stage to the BESS via a non-isolated LVDC converter within the DC-Hub and to the FC via a galvanically isolated LVDC converter. Both converters are modeled in the simulator as averaged two-port units with uniform I/O contracts and calibrated loss/efficiency maps.

4.3.1 DC hub and non-isolated LVDC converter (BESS interface)

Role & scope:

This unit interfaces a battery system to the LVDC bus (bidirectional) and may participate in LVDC voltage support depending on the active operating mode. Detailed switching dynamics are excluded. In the simulator, the whole DC-Hub section is represented as an averaged two-port with a net-efficiency model that includes converter, auxiliaries, and passive losses measured at the cabinet section.

Interfaces (I/O):

- Inputs: LVDC bus voltage v_{lvdc} , BESS terminal voltage v_b . mode: $m \in \{\text{current control (CC)}, \text{voltage control (CV)}, \text{power control (CP)}\}$; LV-bus voltage reference v_{lv}^* , BESS current reference i_b^* , BESS power reference p_b^* , battery terminal voltage window $v_b^{\min}$ and $v_b^{\max}$, current slew-rate limit ρ_i , battery state-of-charge bounds $\text{SoC}_{\min}$ and $\text{SoC}_{\max}$; and boolean flags such as PRECHARGE, READY, TRIP.
- Outputs: BESS current i_b (> 0 discharge to LVDC bus), current injected to the LV bus i_{lv}^{hub} , terminal input/output powers of the DC-Hub section p_{in}^{hub} and p_{out}^{hub} , DC hub losses p_{loss}^{hub} , DC hub net efficiency η_{hub} . Status flags: ACTIVE, LIMITING, FAULT_CODE.

Plant & power relations:

Let i_b be the BESS port current, positive when discharging into the LV bus. Let v_b be the BESS terminal voltage and v_{lv} the LVDC bus voltage. The current magnitude used on the efficiency curve of the DC Hub can be represented by

$$I_b = |i_b| I_{base,lv}, \quad I_{base,lv} = \frac{S_{base,lv}}{V_{base,lv}}$$

Two cases may be defined for power equations: Discharge case, when the DC-Hub section input is the BESS, and Charge case, when the DC-Hub section input is at the LV bus.

A. Discharge (battery $\rightarrow$ bus), $i_b \geq 0$:

$$\begin{aligned} p_{in} &= v_b i_b \\ p_{out} &= \eta_{hub} p_{in} \\ i_{lv}^{hub} &= \frac{p_{out}}{v_{lv}} \\ p_{loss}^{hub} &= p_{in} - p_{out} \end{aligned}$$

where p_{in} is the terminal input power of the DC-Hub section at the BESS port during discharge, p_{out} is the terminal output power at the LV bus port, η_{hub} is the DC Hub net efficiency (includes non-isolated LVDC converter, auxiliaries, passives), i_{lv}^{hub} is the section current injected into the LV bus, and p_{loss}^{hub} is the internal dissipation within the DC Hub section.

B. Charge (bus $\rightarrow$ battery), $i_b < 0$:

$$\begin{aligned} p_{out} &= v_b |i_b| \\ p_{in} &= \frac{p_{out}}{\eta_{hub}} \\ i_{lv}^{hub} &= -\frac{p_{in}}{v_{lv}} \\ p_{loss}^{hub} &= p_{in} - p_{out} \end{aligned}$$

where p_{out} is the terminal output power at the BESS port during charge, p_{in} is the terminal input power drawn from the LV bus. The negative sign in i_{lv}^{hub} ensures the bus sees the current leaving it when charging the battery.

LV bus capacitor (if modeled explicitly) can be represented by

$$c_{lv} \frac{dv_{lv}}{dt} = i_{lv}^{hub} + i_{lv}^{dab} + i_{lv}^{other} - \frac{p_{load,lv}}{v_{lv}}$$

where c_{lv} is the LV link capacitance, i_{lv}^{dab} is the SST MVDC/LVDC DAB injection, i_{lv}^{other} lumps any other LV-bus sources/sinks (e.g., FC converter), and $p_{load,lv}$ aggregates LV-side loads (hotel/aux, etc.).

- Current loop (PI with anti-windup):

$$\begin{aligned} i_b^{cmd} &= \text{sat}(i_b^*, -i_{max}, i_{max}) \\ u_i &= k_{p,i} (i_b^{cmd} - i_b) + k_{i,i} \xi_i \\ \dot{\xi}_i &= \text{aw}(i_b^{cmd} - i_b, u_i, u_{min}, u_{max}) \end{aligned}$$

$$i_b = \alpha (u_i - i_b)$$

where i_b^{cmd} is the limited current reference, u_i is the PI controller output, ξ_i is the integrator state under a standard anti-windup clamp, and α is a first-order plant gain that lumps DC/DC dynamics (can be replaced with a more detailed duty-cycle/inductor model later if desired).

Control modes:

- Current control (CC):

Track the commanded battery current i_b^* (charge < 0 or discharge > 0) with SoC/voltage/slew limits.

- DC-link voltage control (CV)

A. LV-bus CV:

Regulate the LVDC bus voltage v_{lv} to v_{lv}^* . Generates a current reference for the inner CC loop:

$$i_b^* = k_{p,v}^{lv}(v_{lv}^* - v_{lv}) + k_{i,v}^{lv} \int (v_{lv}^* - v_{lv}) dt$$

B. battery-side CV:

Regulate the battery terminal voltage v_b to v_b^* under BMS supervision:

$$i_b^* = k_{p,v}^b(v_b^* - v_b) + k_{i,v}^b \int (v_b^* - v_b) dt$$

- Power control (CP):

Track a battery power target p_b^* with inner CC:

$$i_b^* = \frac{p_b^*}{v_b}$$

Loss & efficiency model:

The net efficiency η_{hub} can be modeled as a parametric curve or efficiency map using the DC Hub efficiency curve. This efficiency includes losses in the non-isolated LVDC converter, auxiliary units, and passive components within the cabinet. p_{loss}^{hub} may be modeled using a quadratic:

$$p_{loss}^{hub} = c_0 + c_1 |i_b| + c_2 i_b^2$$

where the constant term represents nearly fixed auxiliary/standby losses and the linear and quadratic terms represent current-dependent conduction and switching losses.

The net DC Hub efficiency follows directly from output power and losses:

$$\eta_{hub} = \frac{p_{out}^{hub}}{p_{out}^{hub} + p_{loss}^{hub}}$$

Operating limits & interlocks:

- Voltage windows: $v_b^{min} \leq v_b \leq v_b^{max}$.
- Battery constraints: $SoC_{min} \leq SoC \leq SoC_{max}$ and placeholders for T and SoH .

- Current limits: $|i_b| \leq i_{max}$ and slew/ramp $|i_b| \leq \rho_i$.

Parameters:

- Bases: $V_{base,lv}$, $I_{base,lv}$, P_{base}^{hub} .
- Control gains: $k_{p,i}$, $k_{i,i}$.
- Limits: i_{max} , ρ_i , v_b^{min} , v_b^{max} , SoC_{min} , SoC_{max} .
- Bus capacitance (if modeled): c_{lv} .

4.3.2 Galvanically isolated LVDC converter (Fuel-Cell interface)

Role & scope:

This converter provides galvanic isolation and bidirectional DC/DC conversion between the fuel-cell stack and the 0.8 kV LVDC bus. It conditions stack power, enforces stack-specific limits, and coordinates with the Balance-of-Plant (BoP) and the fuel-cell management system (FCMS).

Interfaces (I/O):

- Inputs: FC stack voltage v_{fc} , FC stack current i_{fc} (positive when the converter draws from the stack), LVDC bus voltage v_{lv} , mode select $\in \{CC, CV, CP\}$; bus-side current reference i_{lvdc}^* , bus voltage reference v_{lvdc}^* .
- Outputs: Converter output current to the LVDC bus $i_{lvdc,fc}$ (positive when injecting power to the LVDC bus), power on the LVDC bus side $p_{out}^{fc,conv}$, power on FC stack side $p_{in}^{fc,conv}$.

Control modes:

- Current control (CC): Track bus-side current with SoC/voltage/ramp limits applied via FCMS/BoP and stack windows.

$$i_{lvdc}^* = \frac{I^*}{I_{base,lvdc}}, \quad u_i = k_{p,i}(i_{lv}^* - i_{lv}) + k_{i,i} \xi_i, \quad \dot{\xi}_i = aw(i_{lv}^* - i_{lv}, u_i)$$

- DC-link voltage control (CV): Regulate the LV bus voltage:

$$i_{lv}^* = k_{p,v}(v_{lv}^* - v_{lv}) + k_{i,v} \int (v_{lv}^* - v_{lv}) dt,$$

- Power control (CP): Track FC power setpoint:

$$i_{lv}^* = \frac{p^*}{v_{lv}}, \quad p^* = \frac{P^*}{S_{base}}$$

Plant & power relations:

An averaged two-port power balance can be used. On the FC side $p_{in}^{fc,conv} = v_{fc} i_{fc}$ and on the bus side $p_{out}^{fc,conv} = v_{lvdc} i_{lvdc}$. Net converter efficiency η_{dcfc} maps input to output power:

$$p_{out}^{fc,conv} = \eta_{dcfc} p_{in}^{fc,conv}$$

$$p_{loss}^{fc,conv} = p_{in}^{fc,conv} - p_{out}^{fc,conv}$$

where $p_{in}^{fc,conv}$ and $p_{out}^{fc,conv}$ are terminal powers at the converter input (FC stack) and output (LV bus), respectively; $p_{loss}^{fc,conv}$ is the total converter loss (including magnetics, semiconductors, and cabinet auxiliaries tied to the converter). The converter efficiency η_{dcfc} depends on one or more of $\{i_{lv}, v_{fc}, v_{fc}, i_{fc}\}$ depending on the supplier efficiency curve used.

- BoP-aware stack current:

The net FC output current is defined as:

$$i_{fc} = i_{out,gross} + i_{BoP}$$

where i_{fc} represents FC net output current, $i_{out,gross}$ denotes the gross output current from the FC membranes, and i_{BoP} represents the current supplied to the BoP (e.g., compressors and other auxiliary components). Since i_{BoP} is internally consumed, the net output current i_{fc} is always lower than $i_{out,gross}$. This relation is enforced whenever combined FC+LVDC efficiency data are used.

Operating limits & interlocks:

- Stack window: $v_{fc}^{min} \leq v_{fc} \leq v_{fc}^{max}$, $0 \leq i_{fc} \leq i_{fc}^{max}$.
- LV bus window: $v_{lv}^{min} \leq v_{lv} \leq v_{lv}^{max}$; current limit $i_{lv} \leq i_{lv}^{max}$.

Loss & efficiency model:

Two interchangeable models are supported, selected by scenario:

A. Converter-only map:

When a converter-only efficiency curve is available, a smooth quadratic is fitted in the chosen loading variable (default: bus-side output current i_{lv} or output power p_{out}):

$$p_{loss}^{fc,conv} = d_0 + d_1 |p_{out}^{fc,conv}| + d_2 (p_{out}^{fc,conv})^2$$

$$\eta_{dcfc} = 1 - \frac{p_{loss}^{fc,conv}}{|p_{in}^{fc,conv}|}$$

where d_0, d_1, d_2 are calibrated from supplier curves.

B. Combined FC + LVDC map:

Based on the combined efficiency map of the FC and the isolated LVDC converter, the loss and efficiency equations can be modeled as:

$$p_{out}^{fc,conv} = \eta_{fc+dcfc} v_{fc} i_{fc}$$

$$p_{loss}^{fc+conv} = v_{fc} i_{fc} - p_{out}^{fc,conv}$$

With,

$$i_{fc} = i_{fc,gross} + i_{BoP}$$

Parameter:

- Bases: $V_{base,fc}, V_{base,lv}, S_{base,lv}$.
- Limits: $V_{fc}^{min}, V_{fc}^{max}, I_{fc}^{max}, V_{lv}^{min}, V_{lv}^{max}, I_{lv}^{max}, \rho_P, \rho_I$.
- Efficiency: selector {A,B}; coefficients d_0, d_1, d_2 (A) or BoL/EoL lookup surfaces (B).
- BoP parameters: P_{BoP} or I_{BoP} schedule if using the combined map directly.

4.4 Fuel-cell system

Role & scope:

The fuel-cell subsystem converts hydrogen to DC electric power. In the simulator, it is represented by a control-oriented stack model plus optional Balance-of-Plant (BoP) consumption, with explicit start/stop sequencing and operating limits. Typical net efficiency peaks near mid-load and decreases at high load; lifetime and SoH gradually reduce efficiency and maximum net power. Figure 4 shows the equivalent circuit that can be used to represent the Fuel-cell power system.

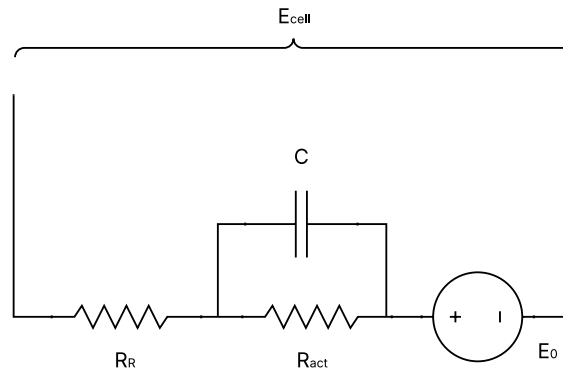


Figure 4: Theoretical fuel-cell equivalent circuit. Courtesy: (Barbir, 2013).

Interfaces (I/O):

- Inputs: setpoint FC power p_{fc}^* , setpoint FC stack current i_{fc}^* , Command selector $m_{fc} \in \{PR, CR\}$ (Power-request or Current-request mode), ramp-rate limits ρ_I, ρ_P , Enable/READY and emergency stop (ES), Degradation scaler $\kappa_{eol} \in [0,1]$ (BoL = 1)
- Outputs: FC stack voltage v_{fc} , FC stack current i_{fc} , Net electrical power before the isolated LVDC converter $p_{fc,net}$, gross power p_{gross} , BoP power p_{BoP} , hydrogen flow $\dot{m}_{H_2}$ [kg/s], State $\sigma_{fc} \in \{OFF, STBY, START, RUN, SHDN, TRIP\}$.

Control modes:

- Power-request mode (PR): track p_{fc}^* (follows Ext_PowReq) with a current command internally:

$$i_{fc}^* = \text{sat} \left(\frac{p_{fc}^*}{v_{fc}}, 0, i_{fc}^{max} \right)$$

$$i_{fc} = rl(i_{fc}^* - i_{fc}, \rho_I)$$

where "sat" applies window limits and "rl" enforces ramp-rate.

- Current-request mode (CR): track i_{fc}^* directly (follows Ext_CurReq) with slew limit ρ_I .

Plant & power relations:

The FC system delivers a specified net power output, defined as the electrical power available ($p_{fc,net}$) after subtracting the consumption of all auxiliary components. The electrical power produced directly by the FC stack is referred to as gross power (p_{gross}), which represents the total generated power before deducting the demand of the Balance of Plant (BoP). The BoP (p_{BoP}) comprises all auxiliary components and subsystems required to support FC operation. The system designation reflects its maximum net power rating. In typical applications, operating the system at 60–80% of its maximum net output offers a suitable compromise between efficiency and FC lifetime.

- Gross-net split and BoP:

$$p_{gross} = v_{fc} i_{fc}$$

$$p_{net} = p_{gross} - p_{BoP}(i_{fc})$$

where p_{gross} represents the FC gross electrical power, p_{net} is the FC net power, and $p_{BoP}(i_{fc})$ captures the auxiliary consumption (compressors, pumps, controllers). p_{BoP} is identified from the FC electrical performance data (affine or quadratic fit vs current).

- Stack polarization (BoL):

A quadratic fit over the operating band reproduces the curve:

$$V_{fc}(I_{fc}) \approx \alpha_0 + \alpha_1 I_{fc} + \alpha_2 I_{fc}^2 \quad [V]$$

where α_0, α_1 , and α_2 are fitted coefficients. Per-unit: $v_{fc} = V_{fc}/V_{base,fc}$.

For completeness, a commonly used theoretical equivalent circuit representation of the fuel cell is shown below, although the FC is modeled functionally in this work.

Efficiency model:

The performance characteristics of the FC system at both BoL and EoL usually describe the relationship between efficiency and Specific Fuel Consumption (SFC) as functions of the net power output. These characteristics are generally obtained under reference conditions, which define the standard operating parameters of the FC system (e.g., temperatures, pressures, etc.).

- Net efficiency and hydrogen flow:

$$\eta_{net}(P_{net}) = \frac{P_{net}}{\dot{m}_{H_2} LHV}$$

where η_{net} is the FC net efficiency, $\dot{m}_{H_2}$ is the hydrogen flow, and LHV is the hydrogen lower heating value, which equals around 120 MJ/kg or 33.33 kWh/kg.

It is numerically more stable to fit SFC [g/kWh] and then derive efficiency. In the simulator, piecewise polynomials across the power range may be used, which results in very small errors compared to the efficiency curve. Then, the net electrical efficiency can be obtained using SFC by

$$\eta_{net}(P_{net}) = \frac{1000}{LHV} \frac{1}{SFC(P_{net})}$$

with $LHV(H_2) = 33.33 \text{ kWh/kg}$,

$$\eta_{net}[\%] = \frac{3000}{SFC(P_{net})}$$

Figure 5 shows the typical FC net electrical efficiency and hydrogen consumption as functions of the FC electrical power output (Najjaran & Skjetne, 2025).

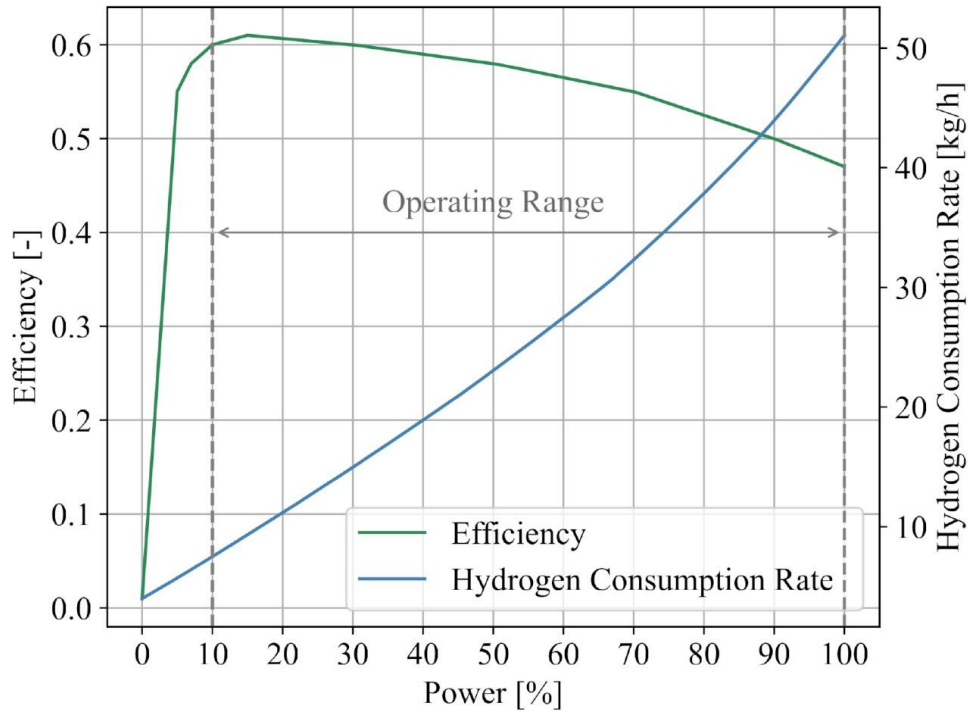


Figure 5: Fuel-cell net efficiency and hydrogen consumption versus net electrical power. Courtesy: (Najjaran & Skjetne, 2025).

- Lifetime/degradation:

In general, stack replacement is recommended when performance has degraded by approximately 5-10% from its initial level. The End of Life (EoL) is commonly defined as a 10% voltage reduction at a current of 300 A. For example, at an average load of 65%, the expected stack lifetime is around 20,000 hours, during which efficiency decreases from 50% at Beginning of Life (BoL) to 46% at EoL. Increasing the average load to 75% shortens the lifetime to roughly 16,500 hours, with efficiency dropping from 44% to 43%.

In the simulator, we can apply a degradation scaler $\kappa_{eol}(t)$ ($0 \rightarrow \text{EoL}$, $1 \rightarrow \text{BoL}$) based on hours-at-load:

$$\eta_{net}(P_{net}, t) = \kappa_{eol}(t) \eta_{net}^{BoL}(P_{net}) + (1 - \kappa_{eol}(t)) \eta_{net}^{EoL}(P_{net})$$

Operating limits & interlocks:

- Voltage range: $V_{fc}^{min} \leq V_{fc} \leq V_{fc}^{max}$.
- Current range: $I_{fc}^{min} \leq I_{fc} \leq I_{fc}^{max}$.
- Current ripple and ramp limit ρ_I [A/s].

Parameters (defaults):

- Bases: $V_{base,fc}$, $V_{base,lv}$, $P_{base,fc}$.
- BoP model: coefficients for $P_{fc,net}$ and P_{gross} .
- Efficiency curves: coefficients for SFC_{BoL} and SFC_{EoL} .
- Limits: V_{fc}^{min} , V_{fc}^{max} , I_{fc}^{min} , I_{fc}^{max} , ρ_I , ρ_P ; start/stop timers.
- Degradation: $\kappa_{eol}(t)$ schedule (hours-at-load weighting).

4.5 Battery energy storage system

Role & scope:

The BESS provides short-term energy buffering on the 0.8 kV LVDC bus for DC-link regulation, peak-shaving, spinning reserve, and black-bus support. The simulator will handle the battery internal electro-thermal dynamics (SoC, OCV, limits, degradation hooks) in the battery module while the converter and DC-Hub net losses are handled by the DC-Hub model (i.e., the BESS equations below represent the pack up to its DC terminals). Device-specific ratings are provided via a configuration file. Figure 6 shows the equivalent circuit that can be used to represent the BESS.

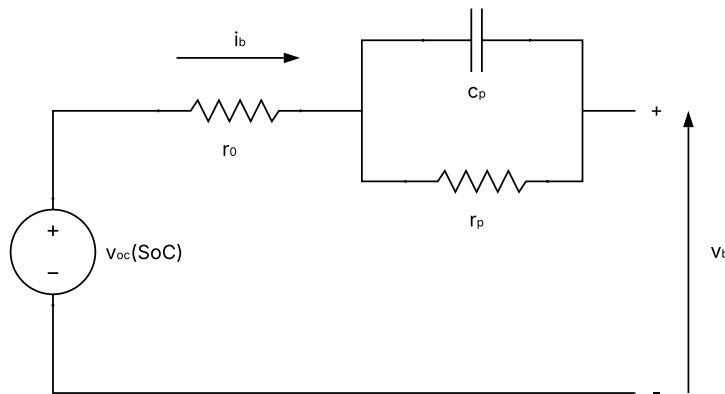


Figure 1: Electrical equivalent circuit of the BESS.

Interfaces (I/O):

- Inputs: BESS current reference i_b^* or BESS power reference p_b^* (issued by the DC-Hub controller/PMS).
- Outputs: BESS terminal voltage v_b and current i_b (positive when discharging to the DC-Hub), BESS terminal power $p_b = v_b i_b$, state of charge (SoC) s and state of health (SoH) h , available charge/discharge limits $r^-(s)$, $r^+(s)$, internal power losses $p_{loss,bat}$.

Plant & power relations:

- State of Health (SoH):

The SoH is represented by (h) and quantifies the capacity degradation as

$$h = \frac{C_{max}}{C_{nom}} = \frac{Q_{max}}{Q_{nom}}.$$

The battery end-of-life (EoL) is typically defined as the point where SoH drops below a specified threshold, commonly $h = 0.8$.

- State of Charge (SoC):

The SoC is represented by (s) defined as

$$s = \frac{C}{C_{max}} = \frac{C}{hC_{nom}} = \frac{Q}{Q_{max}} = \frac{Q}{hQ_{nom}}.$$

The SoC is usually constrained within prescribed bounds s_{min} and s_{max} by the BMS; however, the simulator will allow variation within the full physical range.

- Energy, power, and C-rate relations:

The C-rate (r) is a measure of how fast the capacity of the battery is discharged or charged, defined by

$$r = \frac{I_b}{C_{nom}} = \frac{P_b}{Q_{nom}}.$$

Assuming SoH is constant, the SoC can be used to represent the remaining capacity of the BESS

$$\dot{s} = \frac{\dot{C}}{hC_{nom}} = -\frac{r}{3600h} \quad \Rightarrow \quad s[k+1] = s[k] - \frac{\Delta t}{3600h[k]} r[k].$$

This can be used to calculate how much time it takes to charge the BESS. Assuming a constant C-rate, the duration it takes between two SoC levels is

$$T = \frac{(s_2 - s_1)h}{-r} [hrs].$$

Instantaneous power bounds can be limited using C-rate by

$$r_{min} Q_{nom} \leq P_b \leq r_{max} Q_{nom}$$

Figure 7 visualizes the admissible C-rate envelope used to impose SoC-dependent charge and discharge limits in the BESS module, with piecewise-linear (PWL) bounds (Najjaran et al., 2026).

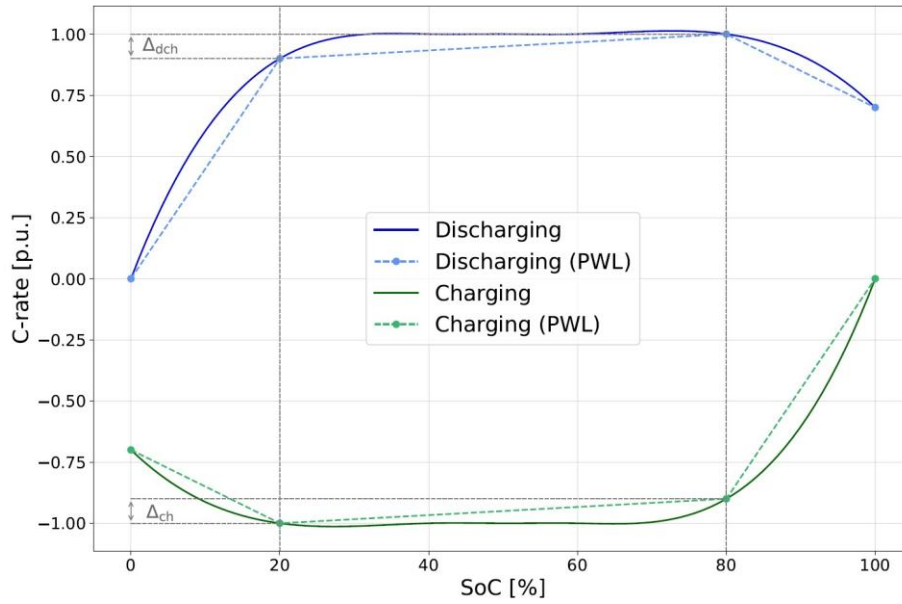


Figure 7: Charging and discharging curves of a lithium-ion BESS. Courtesy:(Najjaran et al., 2026).

- State of Charge (SoC) dynamics:

$$\dot{s} = \begin{cases} -\frac{I_b}{Q_{nom}}, & I_b \geq 0 \text{ (discharge)} \\ -\frac{\eta_c I_b}{Q_{nom}}, & I_b < 0 \text{ (charge)} \end{cases}$$

where $\eta_c \in (0,1]$ is coulombic charge efficiency (typically 0.97-0.99).

- Open-circuit voltage (OCV) vs SoC:

A first-order fit consistent with module limits may be used:

$$v_{oc}(s) = v_{min} + (v_{max} - v_{min}) s$$

with $V_{min} = 691 \text{ V}$, $V_{max} = 907 \text{ V}$, and $V_{base,lv} = 800 \text{ V}$ à $v_{min} = 0.8678 \text{ p.u.}$ and $v_{max} = 1.1338 \text{ p.u.}$ So,

$$v_{oc}(s) = 0.8678 + 0.2660 s$$

- Thevenin (1-RC) terminal model:

$$\begin{aligned} \dot{v}_p &= -\frac{1}{\tau_p} v_p + \frac{r_p}{\tau_p} i_b \\ v_b &= v_{oc}(s) - r_0 i_b - v_p \end{aligned}$$

where r_0 is ohmic series resistance; r_p, τ_p are polarization parameters. Initial values are set conservatively and will be calibrated to measured string droop and recovery.

- Internal losses & heat:

$$\begin{aligned}
 p_{r_0} &= i_b^2 r_0, & p_{r_p} &= \frac{v_p^2}{r_p} \\
 p_{loss,int} &= p_{r_0} + p_{r_p} \\
 C_{th} \dot{T} &= p_{loss,int} - \frac{T - T_{amb}}{R_{th}}
 \end{aligned}$$

Where p_{r_0} is ohmic loss, p_{r_p} is polarization loss, $p_{loss,int}$ is battery internal loss, $C_{th}[\text{J/}^\circ\text{C}]$ and $R_{th}[\text{C/W}]$ represent container thermal inertia and cooling to the ambient, T is the thermal state, and T_{amb} is ambient thermal state.

The equivalent circuit model of the BESS, capturing the electrical behavior relevant at system level, is depicted as follows.

Discrete-time implementation:

With a (fixed step Δt),

$$\begin{aligned}
 s_{k+1} &= \text{sat}_{[s_{min}, s_{max}]}(s_k + \Delta t \dot{s}_k), \\
 v_{p,k+1} &= v_{p,k} + \Delta t \left(-\frac{v_{p,k}}{\tau_p} + \frac{R_p}{\tau_p} I_{b,k} \right), \\
 T_{k+1} &= T_k + \Delta t \left(\frac{p_{loss,int,k}}{C_{th}} - \frac{T_k - T_{amb,k}}{C_{th} R_{th}} \right),
 \end{aligned}$$

with anti-windup on SoC and clamping of I_b to current and ramp limits prior to state updates.

Parameters:

- Electrical: $V_{base,lvdc}$, Q_{arr}
- Resistive & polarization: R_0, R_p, τ_p . (Tune to reproduce measured VI droop).
- Thermal: C_{th} and R_{th} from container cooling sheet, T_{amb} from scenario.
- Limits: $V_{b,min}, V_{b,max}, I_{b,max}, s_{min}, s_{max}$.

4.6 Transformer

Role & scope:

A transformer may interface an upstream AC source to the MVAC bus. In the simulator, it is modeled as a lumped series impedance with optional core loss, suitable for voltage-drop and loss accounting (not for detailed fault studies). Figure 8 shows the equivalent circuit that can be used to represent a transformer.

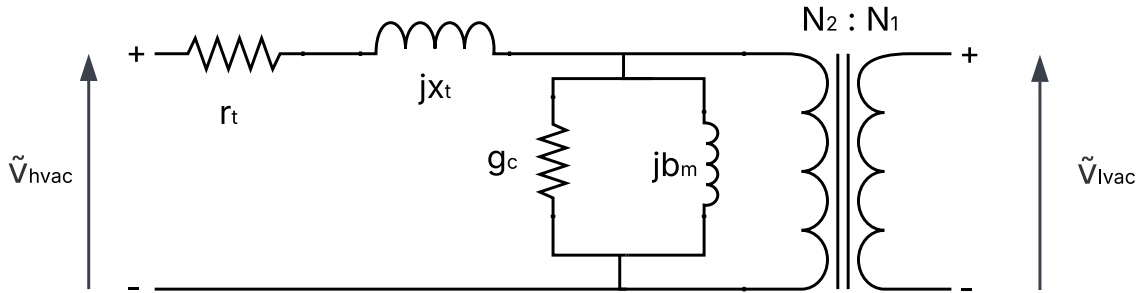


Figure 8: The electrical equivalent circuit of the transformer.

Interfaces (I/O):

- Inputs: L-L generated AC voltage $\tilde{v}_{lvac}$, grid frequency f , tap command n , enable/ready flags.
- Outputs: L-L HVAC $\tilde{v}_{hvac}$, three-phase powers ($p_{in}^{trans}, p_{out}^{trans}$), transformer power loss p_{loss}^{trans} , status/alarms.

Plant & power relations:

- Ideal ratio and vector-group displacement:

$$\tilde{v}_{hvac} = a(n) \tilde{v}_{lvac} e^{j\phi}$$

$$\tilde{i}_{lvac} = a(n) \tilde{v}_{hvac} e^{j\phi}$$

where $a(n) = \frac{V_{base,hvac}}{V_{base,lvac}} (1 + \alpha n)$ is the turns ratio including the tap step.

- Series leakage:

$$\tilde{v}_{drop} = (r_t + jx_t) \tilde{i}_{hvac} = z_t \tilde{i}_{hvac}$$

where r_t is transformer series resistance, x_t is transformer series reactance, and z_t is transformer impedance.

- Shunt (core-loss) branch:

$$g_c = \frac{p_{core}^{trans}}{S_{base}}$$

where g_c is a per-unit conductance that reproduces core loss at rated voltage and p_{core}^{trans} is the core (iron) loss. Note that the large susceptance b_m is to be set from the no-load current when available.

- Power relations:

$$p_{in}^{trans} = \text{Real} \left\{ \frac{3}{2} \tilde{v}_{lvac} \tilde{i}_{lvac}^* \right\}$$

$$p_{out}^{trans} = \text{Real} \left\{ \frac{3}{2} \tilde{v}_{hvac} \tilde{i}_{hvac}^* \right\}$$

where p_{in}^{trans} is the active power drawn on the primary side and p_{out}^{trans} is the active power delivered by the transformer on the secondary side.

- Loss & efficiency model:

The total power loss in the transformer can be represented as

$$p_{loss}^{trans} = p_{core}^{trans} + p_{cu}^{trans}$$

where p_{loss}^{trans} is total power loss, p_{core}^{trans} represents the iron loss, and p_{cu}^{trans} is the copper (winding) loss. Core loss is taken as constant with voltage-squared dependence:

$$p_{core}(v) = g_c v_{lvac}^2$$

Copper loss follows load squared with a temperature factor:

$$p_{cu}(T) = k_{cu}(T) |i_{hvac}|^2 r_t$$

with the temperature factor $k_{cu}(T)$:

$$k_{cu}(T) = \frac{R(T)}{R(75^\circ C)}$$

$$R(T) = R_{20} [1 + \alpha_{Cu} (T - 20^\circ C)]$$

where $R(T)$ follows the copper linear model with a temperature coefficient $\alpha_{Cu} = 0.00393 / ^\circ C$, and R_{20} is the winding resistance at 20 °C (used only to define the temperature law; it cancels in the ratio).

$$\eta_{trans}(T) = \frac{p_{out}^{trans}}{p_{in}^{trans}} = \frac{p_{out}^{trans}}{p_{out}^{trans} + p_{cu}(T) + p_{core}(v)}$$

Parameters:

- Bases: $V_{base,hvac}$, $V_{base,lvdc}$, S_{base} , f_{base} .
- Series leakage (HV-referred): r_t , x_t .
- Core-loss conductance: g_c .
- Magnetizing susceptance: b_m .

4.7 Loads

The simulator treats loads in two categories: propulsion load (mechanical work on the water column mapped to electrical power) and hotel/auxiliary load. Propulsion loads are modeled using two alternative approaches. The internal approach injects a time series of demanded electrical propulsion power at the relevant bus location. A more advanced approach couples the simulator to an external operational model (e.g., *mcsimpy*) to generate propulsion demand from simulated ship operations and environmental conditions. Hotel/auxiliary loads are modeled separately using phase-based schedules and stochastic components. In STEESMAT, propulsion loads are ultimately sourced from the HVDC 5.0 kV or the LVDC 0.8 kV buses regulated by the SST and hotel loads are consumed on the LVAC side. All electrical powers reported to the buses use per-unit $p = P/S_{base}$.

4.7.1 Propulsion electric load

This internal option in the simulator represents propulsion as an electrical power demand applied at HVDC or LVDC distribution buses through a motor drive (VSD/inverter). The simulator does not model

propeller hydrodynamics internally. Instead, the demanded propulsion power is provided directly as a time series $p_{prop,dem}(t)$ derived from mission profiles, measurements, or synthetic scenarios.

Interfaces (I/O):

For each thruster motor:

- **Inputs:** Demanded propulsion power $p_{prop,dem}(t)$ (electrical demand), mode flags (e.g., enabled/disabled).
- **Outputs:** Electrical propulsion consumption $p_{prop,cons}(t)$, unserved propulsion $p_{prop,unserved}(t)$ (if enabled), drive losses $p_{loss,drv}(t)$.

Plant & power relations:

$$p_{prop,cons}(t) = \text{sat}(p_{prop,dem}(t), p_{prop,min}(t), p_{prop,max}(t))$$

where $p_{prop,max}$ and $p_{prop,min}$ may be time-varying due to bus limits, converter operating modes, or supervisory allocations.

If the dictated demand cannot be satisfied due to active constraints (e.g., insufficient available source/transfer capability), the simulator introduces a controlled demand shortfall variable:

$$p_{prop,unserved}(t) = \max(0, p_{prop,dem}(t) - p_{prop,cons}(t))$$

This ensures the simulation remains well-posed and produces diagnostic outputs that can be useful for evaluating supervisory control strategies and load-shedding policies.

Efficiency and loss model

To support consistent energy accounting, the propulsion interface includes a control-oriented drive efficiency model. The drive efficiency may be represented either as a constant, a piecewise curve, or a lookup map. The corresponding drive losses and (optional) mechanical shaft power estimate are:

$$p_{loss,drv}(t) = (1 - \eta_{drv}(t)) p_{prop,cons}(t)$$

where η_{drv} is the drive efficiency.

Note that multiple propulsors are represented by defining one load block per unit and summing their electrical consumption on the relevant buses.

Figure 9 shows typical efficiency characteristics for the LVAC propulsion drive (DC/AC inverter and induction motor) that can be used to parameterize the composite chain efficiency applied to propulsion power accounting (Najjaran & Skjetne, 2025).

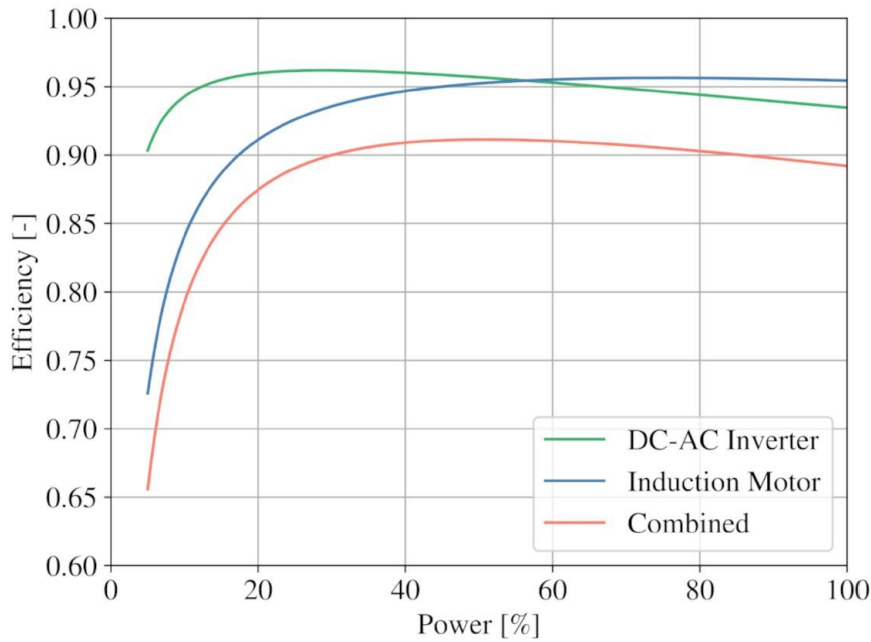


Figure 9: Efficiency maps for a DC–AC inverter and an induction motor representative of the LVAC propulsion.
Courtesy: (Najjaran & Skjetne, 2025).

4.7.2 Operational load emulator

For more realistic propulsion loads from simulated ship operations, the simulator uses *mcsimpy* as an operational load emulator to generate vessel-specific propulsion patterns from a physics-based ship model (3/6-DOF), with state η (pose) and ν (velocities) evolved by either Forward-Euler or RK4 integrators. At each step, *mcsimpy* integrates the marine-craft equations with environmental inputs (current speed U_c , direction β_c) and generalized loads τ (thrusters, wind, waves), and exposes getters for η, ν needed by STEESMAT supervisory logic. Figure 10 shows the STEESMAT simulation loop.

Coupling modes:

- Offline (profile import): precomputed time series ($P_{prop}^{dem}(t)$) is replayed into the simulator.
- Online (co-simulation): at each Δt , the simulator sends available power caps $\bar{P}_{prop}$. *mcsimpy* returns demanded P_{prop}^{dem} . The simulator computes delivered powers subject to SST/DC-Hub/BESS/FC constraints and echoes them back for closed-loop adaptation.

Signal contract (minimal set):

- From *mcsimpy* $\rightarrow$ *mcpowersimpy*: demanded propulsion power P_{prop}^{dem} .
- From *mcpowersimpy* $\rightarrow$ *mcsimpy*: available power $\bar{P}_{prop}$ and delivered power P_{prop}^{del} .

Optional kinematic signals (pose/velocities) may be exchanged when required for specific studies, but they are not required for the core electrical simulation.

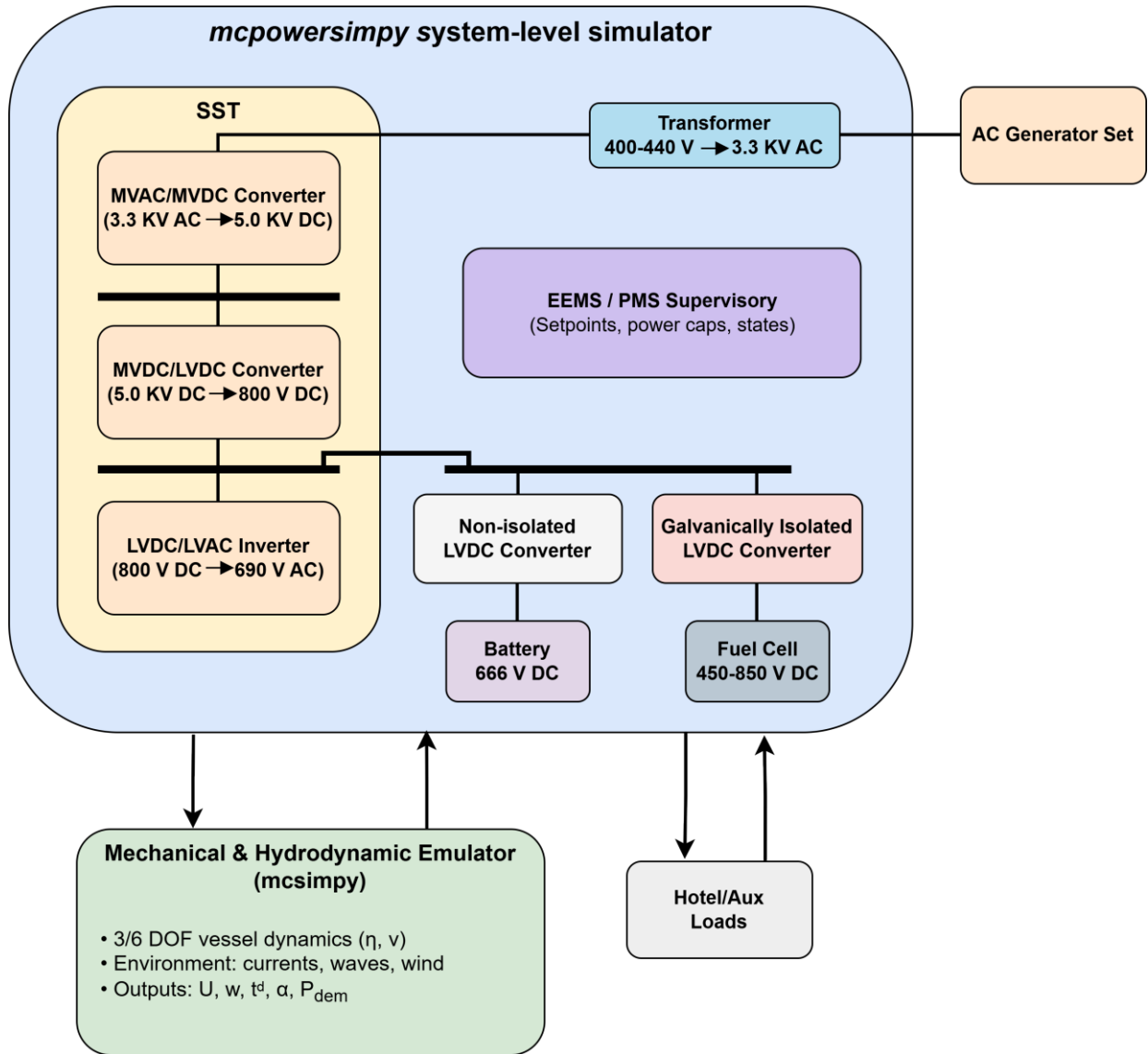


Figure 10: *mcpowersimpy* simulation loop.

4.7.3 Hotel/Aux load model

Hotel/aux demand is modeled as a base level with added operational schedules and stochastic components:

$$P_{hotel}(t) = P_0(phase) + \sum_k A_k \xi_k(t) + \sum_m P_m \chi_m(t)$$

where P_0 is phase-dependent baseload, $\xi_k(t)$ are colored-noise processes (e.g., HVAC cycles) with amplitudes A_k , $\chi_m(t) \in 0,1$ are scheduled block loads (cranes, winches), and P_m represents their magnitudes. Per-unit representation is $p_{hotel} = P_{hotel}/S_{base}$.

5 System Composition and Operational Modes

This chapter summarizes how the system powertrain is composed, how it is operated (state-machine view), and how those operating modes map to *mcpowersimpy* simulator's signals and constraints.

5.1 Bus topology

The power chain is MVAC $\rightarrow$ MVDC $\rightarrow$ LVDC, with LVAC derived locally from LVDC when required for hotel/aux loads:

- **MVAC:** Fed from AC generation via a 400–440 V $\leftrightarrow$ 3.3 kV step-up transformer.
- **MVAC/MVDC AFE:** Rectifier regulates the MVDC link to 5.0 kVDC.
- **MVDC/LVDC DAB:** Stepping down the MVDC to LVDC at 0.8 kV.
- **LVDC Hub:** Contains two converter legs attach to distributed resources:
 - Non-isolated LVDC $\leftrightarrow$ BESS
 - Isolated LVDC $\leftrightarrow$ FC
- **LVDC/LVAC inverter:** supplies hotel/aux loads from the LVDC node as needed.
- **Loads:** propulsion demand (from the hydrodynamics/mission model) and hotel/aux (from inverter or DC, where applicable).
- **EEMS + PMS:** supervisory power sharing, start/stop sequencing, and mode selection.

Each bus/node exposes uniform I/O contracts to *mcpowersimpy*: (v, i, p) per-unit, device READY/INTERLOCK flags, and loss meters. Testbeds share the same logical topology. Differences are captured by parameter sets (ratings, number of modules, limits, etc.)

5.2 Operational modes

This section encodes the operational modes and the step-by-step start/stop order defined for STEESMAT and expresses them as a deterministic state machine that *mcpowersimpy* executes. Two top-level operating modes are supported and sequenced:

A. Grid-forming mode (GFM)

The MVAC/MVDC AFE converter synthesizes MVAC voltage/frequency. Typical responsibility split:

- MVAC/MVDC converter forms the grid.
- MVDC/LVDC converter regulates the voltage at 5 kV or power control as commanded by PMS/EEMS.
- Non-isolated LVDC converter regulates the DC link voltage at 0.8 kV and provides transient support.
- Isolated LVDC converter runs in Power/current control, references taken from PMS/EEMS.
- PMS provide the frequency setpoint to control the power flow.

B. Grid-following mode (GFL)

The MVAC/MVDC AFE converter synchronizes to MVAC via PLL. Typical responsibility split:

- MVAC/MVDC converter maintains the MVDC at 5 kV as required.

- MVDC/LVDC converter runs in power control mode.
- Non-isolated LVDC converter regulates the DC link voltage at 0.8 kV and provides transient support.
- Isolated LVDC converter runs in Power/current control, references taken from PMS/EEMS.
- PMS/EEMS allocate power setpoints subject to device limits and mission objectives.

In the simulator, modes only change setpoint ownership. Inner-loop realizations (dq current loops, PLL, voltage loops, DAB phase-shift control) are the same blocks described in Chapter 5. The mode switch selects which loop runs as the outer objective (voltage vs power) for each converter.

Start/Stop sequences:

Tables 2 and 3 illustrate the start and stop sequences for GFM and GFL, respectively. Each step waits for the corresponding READY/RUN feedback and tolerance windows (e.g., $|v_{lv} - 800 \text{ V}| \leq \Delta V$, $|v_{mv} - 5 \text{ kV}| \leq \Delta V$) before advancing. Timeouts raise TIMEOUT and fall back to a safe state. EBLs must be healthy/closed to tie hub sections. SoC and temperature windows are enforced for the BESS, and warm-up/availability is enforced for the FC.

Table 2: GFM Start/Stop sequence.

Step	Start (GFM)	Stop (GFM)
1	• Verify: BESS OK, FC OK, Non-isolated LVDC OK, Isolated LVDC OK, DAB OK, AFE OK, EBLs OK	• Receive STOP command from EEMS/PMS.
2	• Close BESS contactors (BMS permits). • Command non-isolated LVDC to LVDC-voltage control, ramp to $v_{lvdc} = 800 \text{ V}$.	• STOP AFE (exit GFM).
3	• Enable isolated LVDC with zero power (or current) reference.	• STOP DAB.
4	• Start DAB and hold $v_{mvdc} = 5 \text{ kV}$ (or ramp to target).	• STOP isolated LVDC (FC leg).
5	• Start AFE in grid-forming (Voltage/frequency). • System enters RUN.	• STOP non-isolated LVDC • Open BESS contactors • EBL to safe state • DC-link discharge.

Table 3: GFL Start/Stop sequence.

Step	Start (GFL)	Stop (GFL)
1	• Verify: BESS OK, FC OK, Non-isolated LVDC OK, Isolated LVDC OK, DAB OK, AFE OK, EBLs OK	• Receive STOP command from EEMS/PMS.
2	• Close BESS contactors (BMS permits). • Command non-isolated LVDC to LVDC-voltage control, ramp to $v_{lvdc} = 800\text{ V}$.	• STOP DAB • STOP non-isolated LVDC.
3	• Enable isolated LVDC with zero power (or current) reference.	• STOP non-isolated LVDC • Open BESS contactors • EBL to safe state • DC-link discharge.
4	• Start AFE with PLL and hold $v_{mvdc} = 5\text{ kV}$ (or ramp to target).	• STOP AFE.
5	• Start DAB with $P^* = 0$, then track PMS/EEMS dispatch. • System enters RUN.	-

5.2.1 Simulator state machine, guards, and fault handling

mcpowersimpy simulator turns the verbal operational modes and start/stop procedures into a finite-state machine (FSM) executed at each control step. The FSM defines the states, the transitions between them, and the guards that must be satisfied before advancing. By encoding GFM/GFL as differences in setpoint ownership (who regulates V, I, or P at each step) rather than different plant models, the same control blocks from Chapter 5 are reused while the FSM guarantees safe ordering, rate-limited ramps, and reproducible behavior. All guards operate on per-unit signals (with SI mirrors for logging), and include DC-Hub/EBL permissives, BESS SoC and temperature windows, FC BoP availability and ramp envelopes, MVDC/LVDC voltage bands, and converter current/thermal limits.

States:

{PRECHECK, BESS_ON, FC_ARMED, DAB_ON, AFE_ON, RUN, STOPPING, FAULT}, with Mode $\in$ {GFM, GFL}.

Transitions:

- PRECHECK $\rightarrow$ BESS_ON when all READY and EBL OK.
- BESS_ON $\rightarrow$ FC_ARMED when LVDC is within $\pm \Delta V$ of 800 V.
- FC_ARMED $\rightarrow$ DAB_ON when isolated LVDC RUN with $P^* = 0$.
- DAB_ON $\rightarrow$ AFE_ON
 - GFM: when MVDC within $\pm \Delta V$ of 5 kV;
 - GFL: directly (AFE must lock first).
- AFE_ON $\rightarrow$ RUN when AFE RUN and buses are in tolerance.
- RUN $\rightarrow$ STOPPING on STOP command or fault
 - STOPPING $\rightarrow$ PRECHECK when devices are orderly stopped
 - Any $\rightarrow$ FAULT on trip.

Interlocks & limits:

- Voltage windows: LVDC 800 V $\pm \Delta V$; MVDC 5 kV $\pm \Delta V$.
- Current limits: converter ratings, EBL rating, cable/sector limits.
- Ramp limits: references to obey the configured $|dP/dt|$, $|dI/dt|$.
- BESS: SoC window $[s_{\min}, s_{\max}]$, temperature limits, contactor logic.
- FC: warm-up/availability, stack voltage/current window, BoP readiness.
- EBL/DC-Hub: section status, fault isolation, permissives.
- Pre-charge: link pre-charge before enable; controlled discharge after trips.

Power flow:

At the LVDC Hub node (800 V), power balance in per-unit:

$$p_{\text{lvdc}}^{\Sigma}(k) = p_{\text{sst} \rightarrow \text{lvdc}}(k) + p_{\text{bess}}(k) + p_{\text{fc}}(k) - p_{\text{loads}}(k) - p_{\text{hub_loss}}(k) = 0$$

where p_{lvdc}^{Σ} is the net LVDC node power mismatch, $p_{\text{sst} \rightarrow \text{lvdc}}$ is MVDC/LVDC delivered power, p_{bess} is the non-isolated LVDC/BESS exchange (discharge > 0 , charge < 0); p_{fc} is FC via isolated LVDC; p_{loads} aggregates propulsion and hotel loads, $p_{\text{hub_loss}}$ are DC-Hub net losses (converter, aux, and passive). Supervisory references come from EEMS/PMS per the agreed mode: in GFM, AFE sets frequency while power is steered downstream; in GFL, AFE tracks the existing grid and PMS ramps the FC/BESS setpoints and MVDC/LVDC power.

Droop on LVDC node for parallelism/robustness:

$$i^*(v) = \frac{v^* - v}{r_{\text{droop}}}$$

where i^* is the per-device current reference contribution, v^* is the LVDC nominal (1.0 p.u.), v is the measured LVDC, r_{droop} the slope shaping sharing and stability.

6 Simulator Architecture and Interfaces

This chapter specifies how *mcpowersimpy* is organized to represent the STEESMAT system transparently and reproducibly. It defines the package structure, configuration strategy, module I/O contracts, execution model, and the extensibility hooks for advanced control (EEMS, MPC/RL) and external load emulation (e.g., *mcsimpy*). All modules support p.u. operation (lower-case symbols) with mirrored SI signals (upper-case), ensuring consistent views across components and stakeholders. *mcpowersimpy* is designed to achieve three main goals:

- 1) Configurability: a “scenario-as-data” approach using YAML/JSON formatting to declare buses, units, connections, timelines, limits, and logging.
- 2) Transparency: it contains documented equations, explicit sign conventions, and traceable provenance.
- 3) Extensibility: it is built using plug-and-play unit models, controller hooks, and co-simulation adaptors.

6.1 Package layout

The *mcpowersimpv* system-level simulator is delivered as a layered Python package “*mcpowersimpv*” organized for transparency, configurability, and extensibility. Figure 11 provides a high-level view of the codebase, showing entry points, core runtime, plant models, control modules, adapters (including the mcsimpv bridge), and the KPI/logging pipeline, alongside the configuration assets. The layout includes the scenario set for Energy House, the relative case vessel, and the Scale-up scenario as configuration artifacts. The detailed directory tree is provided in Appendix A. The simulator software is planned to be made publicly available as an open-source codebase. However, the STEESMAT-specific model configurations (e.g., parameter sets) will remain restricted. To support this separation, *mcpowersimpv* architecture is going to be designed so that all case-specific information is stored outside the core code in dedicated parameterization files.

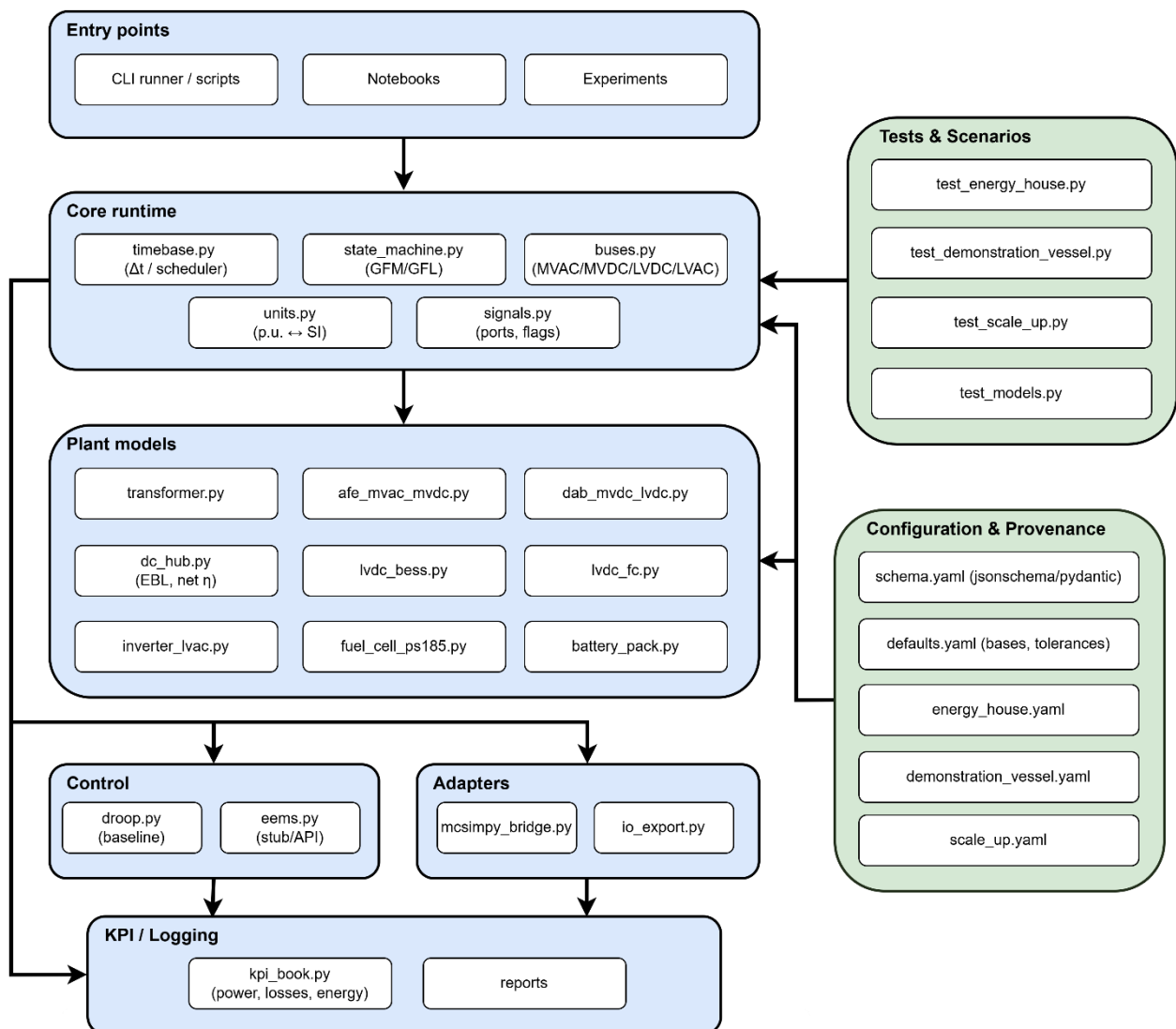


Figure 11: *mcpowersimpv* package diagram.

6.2 Configurability

The simulator follows a “scenario-as-data” approach: every study configuration is captured in a human-readable YAML/JSON file that the runtime validates against a formal schema. A scenario file is the single source of truth for bus bases, unit selections, parameters, interconnections, operating timelines, load profiles, controller modes, limits, and KPI settings. This makes experiments reproducible, reviewable, and portable. The scenario travels with its exact parameters, efficiency maps, and random seeds.

Concretely, a scenario declares:

- Metadata & provenance: name, description, owner, git commit/tag of the model, and a random number generator (RNG) seed for deterministic runs.
- Per-unit bases: V_{base} , S_{base} , and f_{base} for AC for consistent p.u. $\leftrightarrow$ SI conversion across all buses.
- System composition: a list of units (transformer, AFE, DAB, DC-Hub, BESS and FC interfaces, inverter, PS185 FC, battery), each with parameters, limits, and links that define how ports connect.
- Operating timelines & modes: encoded state-machine steps (PRECHARGE/READY/RUN), GFM/GFL role assignments, ramp-rate and interlock thresholds, and start/stop ordering.
- Profiles & missions: references to external load/mission profiles (e.g., mcsimpy-derived propulsion), including noise/disturbance settings to emulate realistic duty cycles.
- Efficiency/loss maps: pointers to vendors $\eta(p)$ tables for converters and hub net efficiency, and performance maps for the FC/BESS used by the plant models.
- Constraints & KPIs: voltage/current/SoC bounds, EBL logic, alarms, and which KPIs to compute (power balance, losses per stage, energy/H₂ use, emissions, cycling metrics).

Scenarios support overlays so “Energy House”, “Relative case vessel,” and “Scale-up” share a common base but diverge in ratings, maps, or timelines. The schema enforces required fields and units, cross-checks map dimensions, and refuses inconsistent configurations (e.g., missing efficiency curves or invalid bus bases). Together, this design ensures that configurability is rigorous, transparent, and scalable.

6.3 Module I/O contracts

This section defines the uniform contracts that every plant model, converter, storage unit, and adapter follows. The goal is to make modules plug-compatible across different scenarios, preserve power/sign consistency, and keep the execution order deterministic so results are reproducible. All modules expose the same three port classes:

- Electrical ports (per bus):
 - DC: DC voltage v , DC current i , and power $p = v i$ (SI: V , I , and P).
 - AC: L–L RMS voltage $\hat{v}$ [p.u.], phase RMS current $\hat{i}$ [p.u.], phasors $\tilde{v}, \tilde{i}$, electrical frequency ω (SI: $\hat{V}$, $\hat{I}$, P , Q).
- Control ports (setpoints & modes):
 - References (examples): p^*, q^*, i^*, v^* , and mode $\in \{GFM, GFL, V/I/P\}$.

- Limits (declared, can be changed at runtime): $v_{min/max}, i_{min/max}$, SoC/SoH windows.
- Status & flags:
 - Boolean: READY, ENABLE, RUN, FAULT, INTERLOCK, EBL_OPEN/CLOSED, GFM_ROLE, GFL_ROLE.
 - Numeric health: SoC, SoH, estimated losses, KPIs.

A consistent notation rule is enforced: lower-case symbols are p.u. and upper-case are SI. Every bus declares $(V_{base}, S_{base}, f_{base})$ for AC in the scenario. Conversions are handled centrally (core/units.py) and applied uniformly by the buses. Power signs follow a common terminal convention: positive power is defined as flowing from the port into the unit. For example, a DC/DC converter's HV side, $p_{in} > 0$ means the converter absorbs power from the upstream bus; on the LV side, $p_{out} < 0$ means the converter delivers power to the bus. Net device power is $p_{device} = p_{in} + p_{out} + p_{aux}$, which equals losses in steady state. Bus conservation is checked at each step: each bus sums all attached terminal powers. The simulator checks $\sum p_k = 0$ up to a small tolerance.

Each unit model shall:

1. Implement the common unit contract (declared ports, setpoints, limits, and flags).
2. Support per-unit signals with mirrored SI outputs at interfaces.
3. Enforce voltage/current/power/SoC limits internally, exposing limit-hit flags and clipped outputs.
4. Provide a loss/efficiency representation and guarantee non-negative losses.
5. Ship with minimal unit tests: power-sign consistency, loss non-negativity, limit enforcement, and NaN/Inf prevention.

All modules implement a common lifecycle:

- PRECHARGE → READY → RUN → STOPPED. FAULT is latched and requires RESET.
- Interlocks (e.g., EBL closed, DC-link above threshold αv^* , SoC windows) are checked at state transitions.
- Start–stop order and GFM/GFL role assignment are driven by the scenario state-machine (core/state_machine.py) and reflected by GFM_ROLE/GFL_ROLE flags.

Each time step $t_k \rightarrow t_{k+1}$ uses a deterministic evaluate → aggregate → commit pattern:

1. Evaluate: each unit receives its bus voltages, control references, and flags sampled at (t_k) and computes proposed terminal injections and next internal states.
2. Aggregate: buses check power balance, apply EBL/open-circuit conditions, and enforce small residual redistribution if needed. Also, limit violations are flagged.
3. Commit: units commit their states and published outputs, flags are latched, and KPIs are accumulated.

Anti-windup, rate limiters $(\dot{p}, \dot{i}, \dot{v})$, and saturation are implemented inside units, power clipping is owned by the unit and is visible in its outputs and flags.

6.4 Extensibility hooks

mcpowersimpy is designed to host advanced supervisory control, such as STEESMAT EEMS. Moreover, it is built to couple external load models as *mcsimpy*, without modifying the plant code. Hooks are typed, versioned, and validated at load time.

EEMS supervisory control API: The supervisor integrates the mission–guidance–reactive hierarchy. All observations, actions, and constraints seen by supervisory controllers can be logged as structured rows (timestamp, scenario metadata, git commit, RNG seed). A controller plugin implements the following interface:

- `observe(t)`: returns a view of bus voltages/currents, unit states, role flags (GFM/GFL), and constraint margins.
- `set_objectives(obj)`: sets mission-level goals such as minimizing fuel consumption and emissions with reserve/SoC targets.
- `forecast(h)`: supplies forecasts for horizon (h) such as demand profiles, wind/current disturbances, and price/emissions factors.
- `plan(t, h)`: computes a plan over h by giving setpoints such as $p_{fc}^*(k)$, $p_{bess}^*(k)$, $v_{dc}^*(k)$.
- Timing contract: reactive layer runs every simulation step, while guidance can run at a slower rate (e.g., 0.5–2 s), and mission updates are event-driven.

The pipeline records per-run energy use, H₂ mass, emissions, cycling metrics, and degradation proxies, enabling reproducible computation of KPIs that target 10-20% energy-efficiency and 5-10% components lifetime improvements across all scenarios when using both the STEESMAT simulation software and EEMS.

Co-simulation adapter: External mechanical/hydrodynamic models (such as *mcsimpy*) can drive realistic propulsion and hotel loads. Two coupling modes are supported *mcpowersimpy*:

- Offline import: The adapter reads time-stamped traces of demanded shaft torque or electrical power partitions (propulsion or hotel). The simulator converts these to electrical demands per bus and feeds them into the DC or AC loads. This is the recommended mode for batch studies and repeatability.
- Online co-simulation: A light inter-process communication library, such as ZeroMQ, or an in-process bridge advances the external model on the simulator clock. Clock master is the simulator; at each (t_k) , it:
 1. Sends electrical setpoints or bus states as needed.
 2. Requests the next mechanical/electrical load vector.
 3. Receives loads and applies them to plant models.

Rate mismatches are handled via sample-and-hold or interpolation, while optional buffering can emulate latency.

The adapter supports either:

- Mechanical to electrical: $\tau_{\text{shaft}}, \omega_{\text{shaft}} \mapsto P_{\text{prop}}(t)$, then partitioned to the relevant bus.
- Electrical direct: $P_{\text{prop}}(t)$ and $P_{\text{hotel}}(t)$ are given as a time series.

Extending the plant side: To add a new unit (e.g., an alternative LVDC converter or a shore charger), implement the base Unit contract:

- Describe ports (DC/AC), setpoints, limits, and flags.
- Provide an averaged model and efficiency loss map.
- Register the unit type in “schema.yaml” and supply default parameters in “defaults.yaml”.
- Add unit tests such as loss non-negativity and limit enforcement.

These hooks keep *mcpowersimpy* open for EEMS integration, algorithm prototyping, and scale-up, while ensuring that all additions remain schema-validated, power-consistent, and traceable.

7 Integration with Energy and Emission Management System

This chapter describes how *mcpowersimpy* integrates with the EEMS to develop, validate, and demonstrate a smart AI-based supervisory control system that minimizes fuel/energy use, emissions, and operating cost while preserving safety and equipment health. The EEMS is complementary to the vessel’s Power Management System (PMS), PMS retains authority for switchboard protection, breaker control, and class-critical interlocks, whereas EEMS issues high-level references and mode decisions (e.g., fuel-cell and battery power setpoints, DC-link targets, and GFM/GFL role scheduling) within the operating envelopes enforced by PMS, protection, and unit controllers. NTNU leads the simulator and EEMS development in WP4. The impact of EEMS is quantified with a standardized KPI. Each scenario is run in baseline (rule-based) versus optimized (EEMS), under identical loads and limits. The combined software targets 10-20 % energy-efficiency gains and 5-10 % lifetime improvements in components, mainly battery and FC.

PMS/EEMS functional interface:

PMS retains authority for protection, breaker/EBL control, and class-critical interlocks. EEMS provides supervisory references within envelopes enforced by PMS and unit controllers. *mcpowersimpy* shall accept relevant commanded reference signals from PMS/EEMS and include the necessary control and constraint-handling functionality to apply these commands in the simulated power system. *mcpowersimpy* exposes a minimal interface set to support this interaction:

- Inputs from PMS: start/stop commands, mode selection (GFM/GFL), breaker/EBL status, interlocks, and (in GFM) frequency reference; (in GFL) power-reference routing.
- Inputs from EEMS (via PMS): setpoints such as p_{fc}^* , p_{bess}^* , v_{dc}^* , MVDC/LVDC power targets, and reserve targets.
- Outputs to PMS/EEMS: bus voltages/currents/powers, unit status flags (READY/RUN/TRIP), constraint margins (SoC/voltage/current), per-stage loss accounting, and KPI summaries.

Optimization core:

In closed-loop runs, *mcpowersimpy* provides the EEMS with observations (bus voltages/currents, unit states such as SoC/SoH, flags, and recent KPIs) and accepts supervisory setpoints (e.g., p_{fc}^* , p_{bess}^* , v_{dc}^* , MVDC/LVDC transfer targets, and reserve requests). The interaction follows the mission–guidance–reactive hierarchy already used throughout this report: the mission layer defines operating modes and objectives, the guidance layer performs predictive optimization to coordinate energy

sources based on load demand and forecasts, energy availability, and environmental conditions, and the reactive layer executes setpoints at the simulator step while honoring ramp-rate, voltage/current, and SoC constraints. The EEMS runs at a slower, supervisory rate (e.g., 0.5–2 s), synchronized to the simulator's fixed timebase; all saturation and anti-windup remain inside the plant units, so any clipping is transparent in unit outputs and flags.

EEMS integration *mcpowersimpy* follows the contracts in Section 6.4. *mcpowersimpy* offers a typed *observe(t)* snapshot that includes bus voltages/currents, unit states, and flags such as GFM/GFL roles. The EEMS returns a set of references for the plant, such as p_{fc}^* to the isolated LVDC converter, p_{bess}^* to the non-isolated LVDC converter, and v_{dc}^* to the MVDC/LVDC regulation loop. The same integration used in simulation applies to testbeds experiments. For scale-up studies, the EEMS operates unchanged; only scenario YAMLs, maps, and ratings differ, enabling technology to market extrapolation without recoding. The EEMS is a plugin to *mcpowersimpy* simulator with a stable configuration that lives in scenario YAML. Every run records observations, actions, and setpoints so results can be audited and replicated.

8 Validation and Verification Strategy

This chapter defines what *mcpowersimpy* simulator produces at the end of a run and how we verify that results are physically consistent, numerically robust, and traceable to inputs and assumptions. Outputs are standardized across testbeds and scale-up scenarios. Verification combines device-level checks, integration checks, and system KPIs.

Each run emits a compact and reviewable record consists of:

- Timeseries (CSV):
 - Buses: voltage, current, power, and frequency where applicable on MVAC/MVDC/LVDC/LVAC.
 - Units: terminal powers and internal references (AFE, DAB, DC-Hub, BESS/FC interfaces, inverter, transformer), losses per stage, and status/flags.
 - Storage states: BESS SoC/SoH, FC net power, hydrogen flow, efficiency, and degradation scaler κ_{eol} .
 - Loads: propulsion and hotel partitions, including the received mechanical/electrical profiles from the external load emulator.
- Run summary (JSON): scenario fingerprint, solver settings (step Δt), and the KPI book.
- Plots: SoC/SoH trajectories, DC-link regulation, per-stage loss traces, bus power-balance residuals, start/stop state chart.
- Power-flow report: per-stage power flow with cumulative losses.
- Event log: time-stamped state-machine events, limit hits, and FAULT/RESET transitions.

Validation compares *mcpowersimpy* outputs to partners data and measurements, progressing from components to full system: Component validation, subsystem validation, system validation.

Default targets unless partner-specific limits are provided:

- Efficiencies at supplied points: $\pm 0.3\%$ absolute.

- DC-link regulation in steady state: $\pm 1\%$ of setpoint; transient peak within declared band.
- Bus power residual: $\leq 1 \times 10^{-3}$ p.u. per step.
- SoC tracking error over profile: $\leq 1\%$ absolute.
- Hydrogen use over profile: $\leq 3\%$ relative error vs measurements.

The simulator and EEMS are evaluated by paired runs: a baseline (rule-based approach) and an optimized run. Both runs are under identical scenarios (loads, disturbances, and limits). Program-level targets are explicit: 10-20 % energy-efficiency gains and 5-10 % lifetime improvements for battery and FC. The simulator computes the following KPIs:

1. Total energy efficiency gain:

$$Gain_{energy} = 1 - \frac{E_{opt}}{E_{base}}, \quad E = \int_0^T P_{prime}(t) dt$$

where $E_{optimized}$ is the optimized energy, E_{base} is the baseline energy, and P_{prime} is the energy drawn from prime sources.

2. Hydrogen-use reduction (FC scenarios):

$$Gain_{H_2} = 1 - \frac{m_{H_2,opt}}{m_{H_2,base}}, \quad m_{H_2} = \int_0^T \dot{m}_{H_2}(t) dt$$

$\dot{m}_{H_2}$ is derived from FC net power and the scenario's net efficiency map.

3. Battery lifetime improvement:

$$Gain_{BESSlife} = 1 - \frac{Aging_{opt}}{Aging_{base}}, \quad Aging = \alpha_{cyc} EFC + \alpha_{rate} |\bar{C}_{rate}|$$

with,

- Equivalent Full Cycles EFC = $\frac{1}{E_{nom}} \sum_{k=0}^{K-1} |P_{batt}[k]| \Delta t$
- Average C-rate magnitude $|\bar{C}| = \frac{1}{T} \sum_{k=0}^{K-1} \frac{|I_{batt}[k]|}{I_{nom}} \Delta t$

where E_{nom} is nominal stored energy, P_{batt} is battery terminal power, I_{batt} is battery current, I_{nom} nominal current, and $\alpha_{cyc}, \alpha_{rate}$ are scenario weights.

4. Fuel-cell lifetime improvement:

$$Gain_{FClife} = 1 - \frac{Degrad_{opt}}{Degrad_{base}}, \quad Degrad = \beta_V V_{decay} + \beta_J \Phi_J$$

with,

- Voltage-decay per throughput $V_{decay} = \frac{\Delta V_{ref}}{Q_{through}}$ and $Q_{through} = \sum_{k=0}^{K-1} \frac{|I_{fc}[k]|}{1000} \Delta t$
- High current-density fraction $\Phi_J = \frac{1}{T} \sum_{k=0}^{K-1} J[k] \geq J_{thr} \Delta t$ and $J = \frac{|I_{fc}|}{A_{act}}$

where I_{fc} is FC stack current, A_{act} is the active area, J_{thr} is the current-density threshold, ΔV_{ref} is the reference voltage loss over the run normalized by total charge throughput $Q_{through}$, and β_V, β_J are scenario weights.

5. Emissions reduction: Computed from prime-source energy and scenario emissions factors (CO₂ per kWh).
6. Auxiliary KPIs: per-stage loss shares for each component, reserve compliance, bus power-balance residuals, maximum DC-link deviation, and ramp compliance.

Finally, for a scenario to be marked “verified”, the following must hold:

1. Sequences: all mandated start/stop transitions executed in order, no missed interlocks, and no illegal role changes.
2. Conservation: per-bus power residual $|\sum p_k| \leq \epsilon_p$ at all steps (default $\epsilon_p = 10^{-3}$ p.u.).
3. Limits: no sustained violation of voltage/current/SoC windows, and transient overshoots are less than declared margins.
4. Loss maps: stage efficiencies within supplier-fit tolerance.
5. Numerics: time-step sensitivity within bounds, no NaN/Inf.
6. KPI targets (when EEMS enabled): energy-efficiency improvement within the 10–20 % band for target missions, and lifetime proxies improved by 5–10 % for BESS/FC, or an evidence-based explanation if targets are not met.

9 Conclusion

This report defines a transparent system-level model of the STEESMAT hybrid-electric power system and its Python-based simulator implementation plan. Building on the system definition and characteristics, the report specifies the simulator's intended use and modeling assumptions. Numerical unit models are provided for the principal subsystems. These include the SST, the DC-Hub with both the isolated and non-isolated LVDC converters, the PS185 FC system, the BESS, the transformer, and representative propulsion and hotel loads. Operational modes and start–stop state machines are formalized. The software organization, I/O contracts, and execution model are documented so that modules remain plug-compatible and results remain traceable. Integration with the EEMS is described, and standardized outputs and verification strategies are established.

The approach is organized around three principles. Configurability is achieved through “scenario-as-data” YAML files that declare bus bases, unit selections, parameters, efficiency maps, limits, and state-machine sequences. Transparency is ensured by documented equations, explicit sign conventions, and bus-level power-balance checks. Extensibility is supported through uniform module I/O contracts and a deterministic evaluate, aggregate, and commit stepper. Clean hooks are provided for supervisory control and optimization in EEMS and for external load emulation via mcsimpy, which enables realistic propulsion profiles. Loss and efficiency models are calibrated where partner data are available, while degradation tracking is enabled through a fuel-cell end-of-life scaler and battery aging proxies. The resulting KPI framework supports paired baseline versus optimized runs and quantifies the program targets. The expected energy-efficiency gain is 10–20%, and the expected lifetime improvement for battery and FC is 5–10%.

Some limitations and planned refinements are noted and will be refined as data is released. Thermal models in the BESS and FC interfaces are simplified and will be extended with validated temperature dynamics when available. The mcsimpy co-simulation path is supported and will be profiled for latency and robustness before Energy House and relative case vessel trials. Numerical sensitivity and acceptance criteria are defined so that future additions preserve determinism and reproducibility.

The same integration described for the simulation will be used in Energy House and on the vessel demonstrator. EEMS will operate first in shadow mode for advisory comparison and then in limited authority under agreed envelopes and PMS oversight. By D4.4, the repository will include implemented modules, unit tests, acceptance benches, scenario packs for Energy House, relative case vessel, and scale-up exemplars, as well as calibration notebooks and continuous-integration recipes that regenerate KPI reports. With this report, a coherent and auditable model system is now in place. It supports the development of smart management and control functions in WP4 and provides a credible route from simulation to demonstration, and onward to standardization and commercialization of SST-based MVDC ship power systems.

References

- Barbir, F. (2013). PEM Fuel Cells: Theory and Practice, 2nd Edition.
- Gezer, E. C., Moreau, M. K. I., Høgden, A. S., Nguyen, D. T., Skjetne, R., & Sørensen, A. (2025). *Digital-physical testbed for ship autonomy studies in the Marine Cybernetics Laboratory basin* Int. Conf. Ships and Offshore Structures (ICSOS),
- Hygen, J.-E. (2023). *Deterministic Response Prediction of Wave-induced Vessel Motions* <https://hdl.handle.net/11250/3095040>
- Mo, H., Kongshaug, M. F., Skjetne, R., Marley, M. H., & Gezer, E. C. (2023). *Real-time compensation of residual loads in dynamic positioning control systems* <https://hdl.handle.net/11250/3093878>
- Najjaran, S., & Skjetne, R. (2025). Nested optimal energy scheduling and power allocation for zero-emission high-speed passenger vessels. *Ocean Engineering, Elsevier*, 322, 120223. <https://doi.org/10.1016/j.oceaneng.2024.120223>
- Najjaran, S., Sundvor, I., Thorne, R. J., & Skjetne, R. (2026). Feasibility Assessment of Zero-Emission High-Speed Passenger Vessels Using Optimal Energy Scheduling and Power Allocation. *Under review in Ocean Engineering, Elsevier*.
- Sustainable Energy. (2021). *Energy House Datasheet*. https://sustainableenergy.no/wp-content/uploads/2021/01/220119_EnergyHouse_-_Datasheet.pdf

Appendix

Appendix A — *mcpowersimpy* simulator package layout (full directory tree)

```

mcpowersimpy/
├── core/
│   ├── timebase.py      # Execution engine (time, buses, state machine)
│   ├── buses.py         # Fixed-step scheduler ( $\Delta t$ ), clock, seeds
│   ├── state_machine.py # MVAC/MVDC/LVDC/LVAC bases, power sums
│   ├── units.py         # GFM/GFL start-stop, interlocks, guards, timers
│   └── signals.py       # Per-unit helpers (upper=SI, lower=p.u.), conversions
│
├── plant/
│   ├── transformer.py   # Typed ports, flags (READY/FAULT), events
│   ├── afe_mvdc_mvdc.py # Averaged numerical unit models
│   ├── dab_mvdc_lvdc.py # 3.3 kV/0.4–0.44 kV, copper/core losses, tap
│   ├── dc_hub.py        # MVAC/MVDC AFE, GFL/GFM,  $\eta(p)$ 
│   ├── lvdc_bess.py     # MVDC/LVDC DAB, power/flux,  $\eta(p)$ 
│   ├── lvdc_fc.py       # LVDC hub (net  $\eta$  incl. auxiliaries, EBL)
│   ├── inverter_lvdc.py # Non-isolated BESS LVDC converter (I/V/P modes)
│   ├── fuel_cell_ps185.py # Isolated FC LVDC converter (V/I/P modes)
│   └── battery_pack.py  # LVDC/LVAC inverter (GFM/GFL stubs)
│
├── control/
│   ├── droop.py         # PS185 net/BoP maps, SFC, degradation  $\kappa_{eol}(t)$ 
│   ├── mpc.py           # SoC/SoH dynamics, limits
│   └── eems.py          # Supervisory & advanced control hooks
│
├── adapters/
│   ├── mcsimpy_bridge.py # Baseline droop, DC-link support, sharing
│   └── io_export.py      # API/stub for MPC (predict/optimize)
│
├── kpi/
│   ├── kpi_book.py      # Orchestration between mission/guidance/reactive (API/stub)
│   └── reports.py
│
├── config/
│   ├── schema.yaml      # External coupling & I/O
│   ├── defaults.yaml    # Offline profile import / online co-simulation
│   ├── eff_maps/        # CSV / JSONL logging, checkpoints
│   │   ├── afe_vacon3000.csv, dab_isop.csv, dc_hub_net.csv, fc_perf.csv
│   │   └── profiles/
│   │       ├── harbor_ops.parquet
│   │       ├── transit.parquet
│   │       └── dp_stationkeeping.parquet
│   └── scenarios/
│       ├── energy_house.yaml # Metrics, reports, provenance
│       └── # Power balance, losses, energy,  $H_2$ , emissions
│           # Auto figures/tables, scenario summaries
│
├── # "Scenario-as-data"
│   ├── schema.yaml      # Pydantic/JSON-Schema for scenarios
│   ├── defaults.yaml    # Voltage/power bases, tolerances, solver opts
│   ├── eff_maps/        # Vendor curves & digitized  $\eta(p)$  maps
│   │   ├── afe_vacon3000.csv, dab_isop.csv, dc_hub_net.csv, fc_perf.csv
│   │   └── profiles/
│   │       ├── harbor_ops.parquet
│   │       ├── transit.parquet
│   │       └── dp_stationkeeping.parquet
│   └── scenarios/
│       ├── energy_house.yaml # *** Target systems ***
│       └── # Land-based integration target

```

```
|   ├── demonstration_vessel.yaml    # At-sea demonstrator
|   └── scale_up.yaml                # Higher power system
|
|   └─ cli/                          # Entry points & batch utilities
|       ├── run_scenario.py          # CLI to load scenario, run, emit KPIs
|       └── sweep.py                 # Parameter/architecture sweeps, uncertainty
|
|   └─ tests/                         # Unit & scenario tests
|       ├── test_models.py           # Per-unit invariants,  $\eta(p)$  fits, limits
|       └── scenarios/
|           ├── test_energy_house.py # Short sim; power balance, no violations
|           ├── demonstration_vessel.py # Ditto for vessel config
|           └── test_scale_up.py      # Ditto for archetypes
```