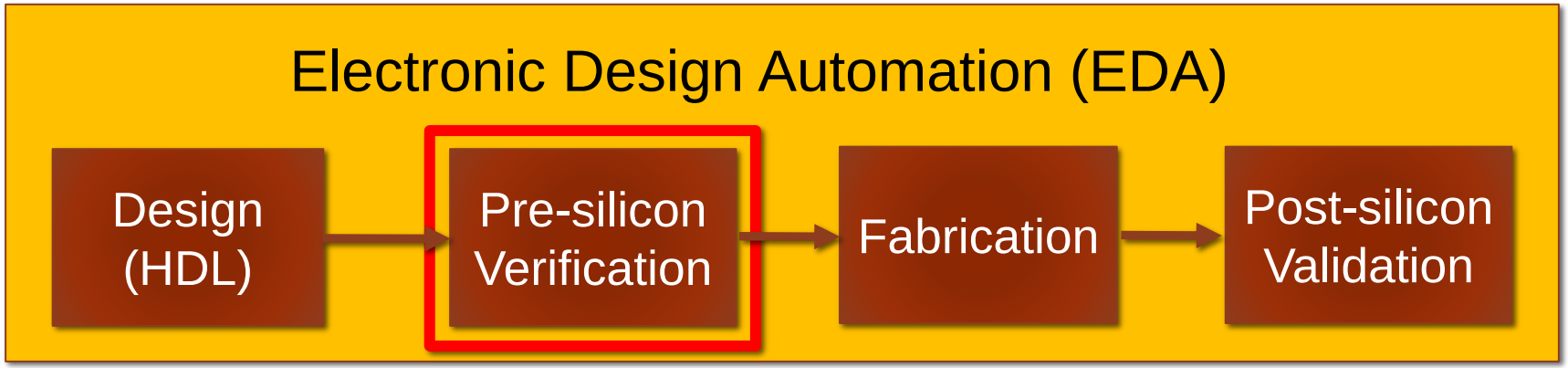
A Theoretical Framework for Symbolic Quick Error Detection

FLORIAN LONSING
SUBHASISH MITRA
CLARK BARRETT

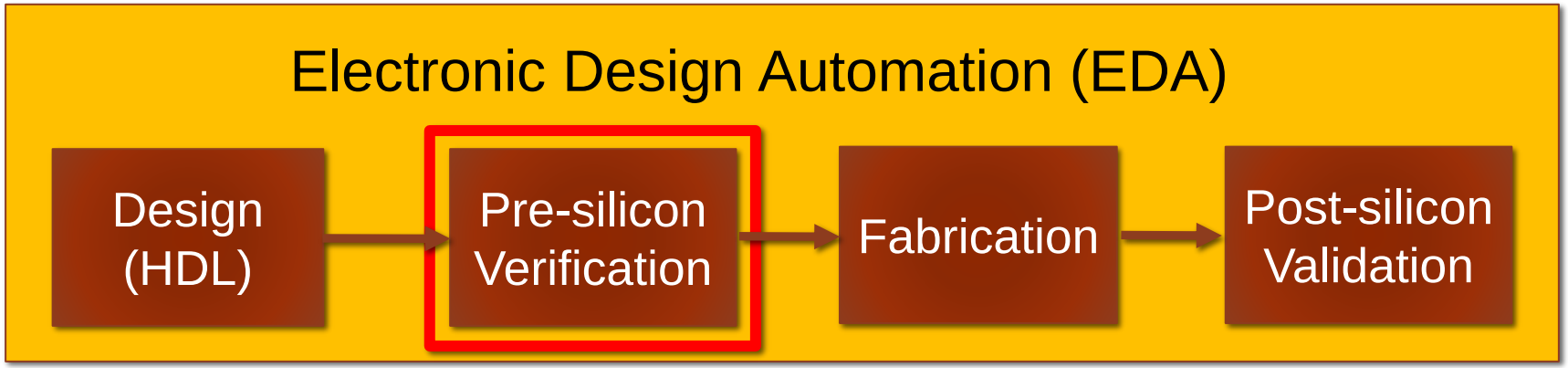
Paper published at Formal Methods in Computer-Aided Design (FMCAD) 2020
Preprint: https://arxiv.org/abs/2006.05449

Context: Pre-Silicon Verification



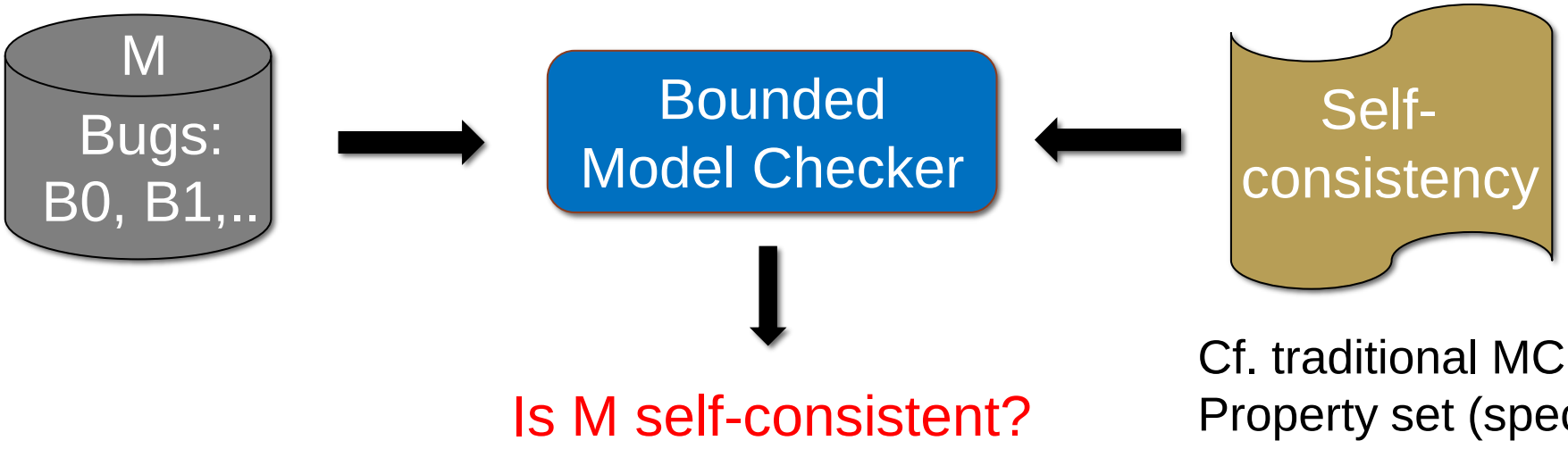
- Our focus: processor designs.
- Formally verify model of a design (e.g. Verilog).
- Model checking vs. non-formal simulation or testing.

Context: Pre-Silicon Verification



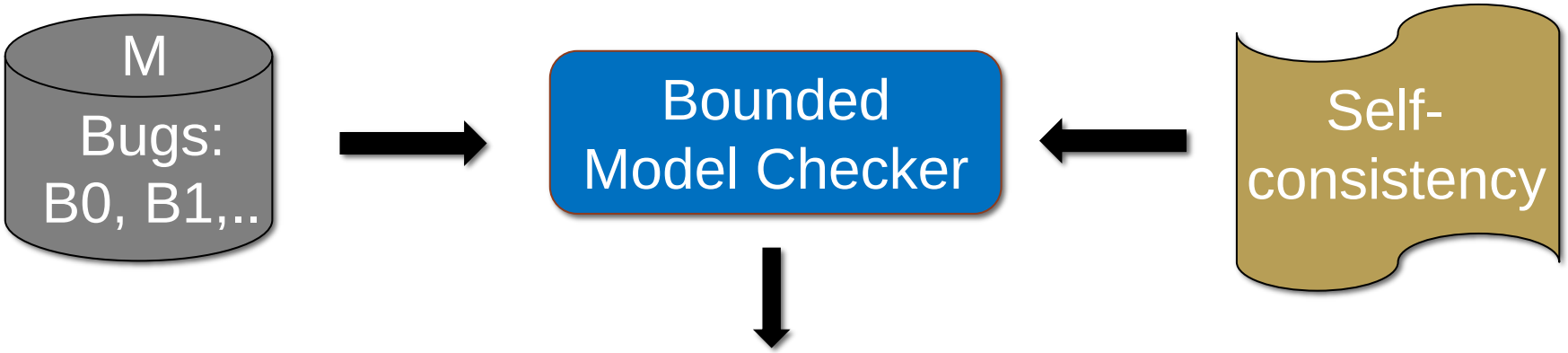
- Our focus: processor designs.
- Formally verify model of a design (e.g. Verilog).
- Model checking vs. non-formal simulation or testing.

Symbolic Quick Error Detection (SQED)



- No formal spec or implementation-specific properties.
- Self-consistency: universal property.
- Industrial-strength technique.

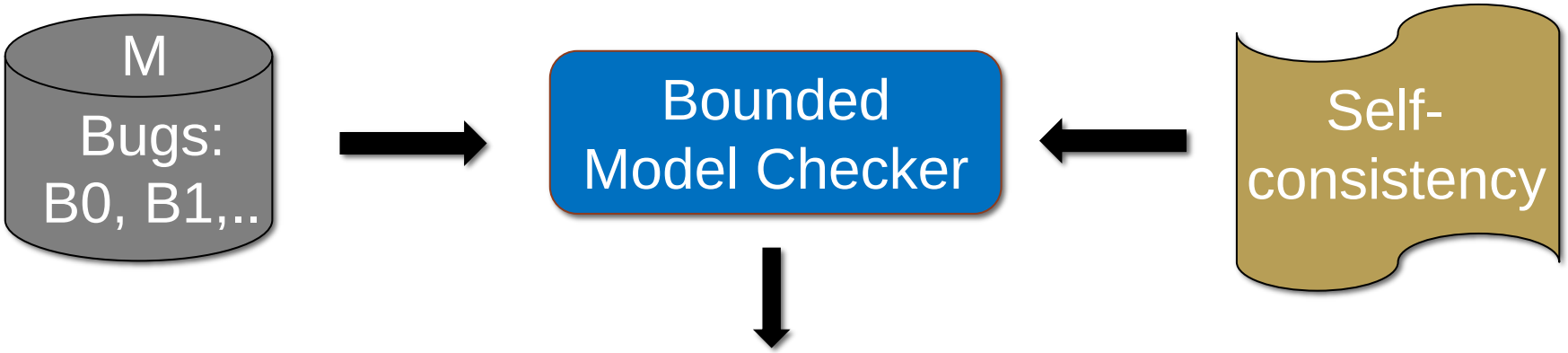
Our Contributions: SQED Soundness Proof



- If M not self-consistent then B ∈ M.

No spurious cex

Our Contributions: SQED Completeness Proof

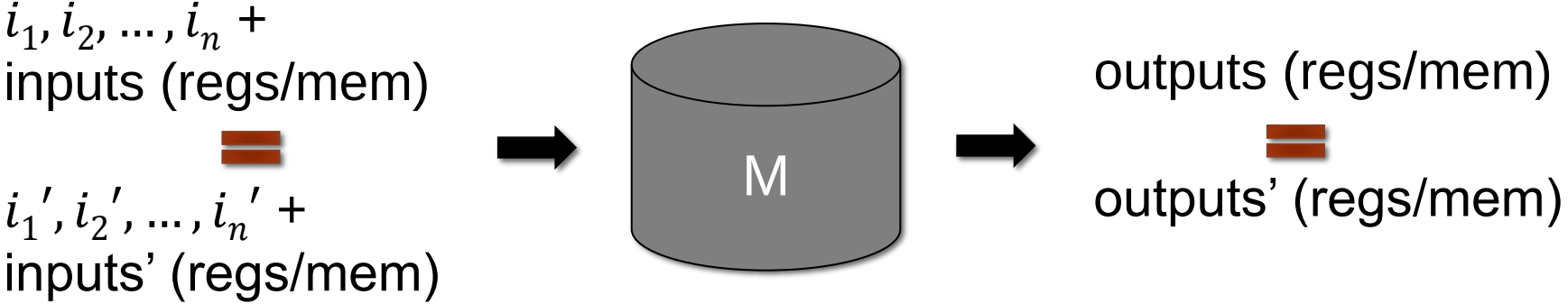


- If B ∈ M then M not self-consistent.

Conditional/full Completeness

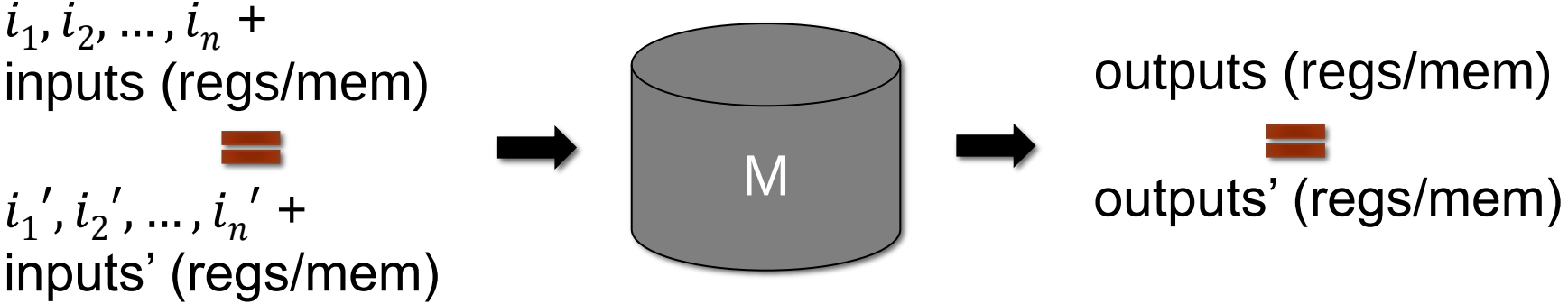
Self-Consistency

- Processor Design M:



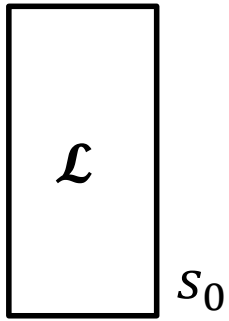
Self-Consistency

- Processor Design M:



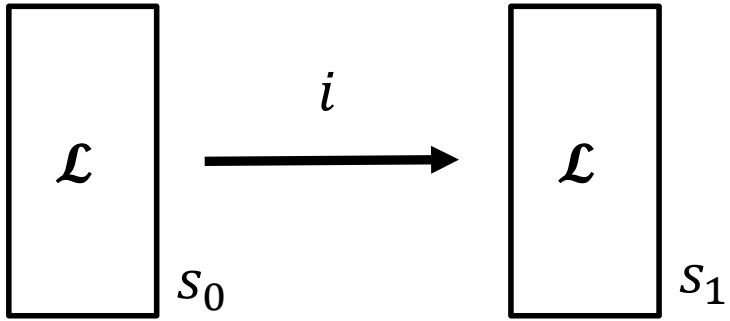
HW designs have complex internal state (pipeline,…).

Formal Model of Processors and SQED



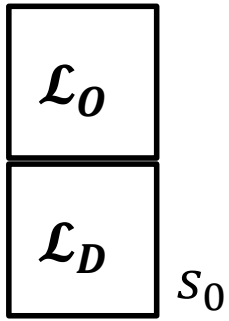
- State s_0: mapping from locations $\mathcal{L}$ to values.
- $\mathcal{L}$: register and memory locations.

Formal Model of Processors and SQED



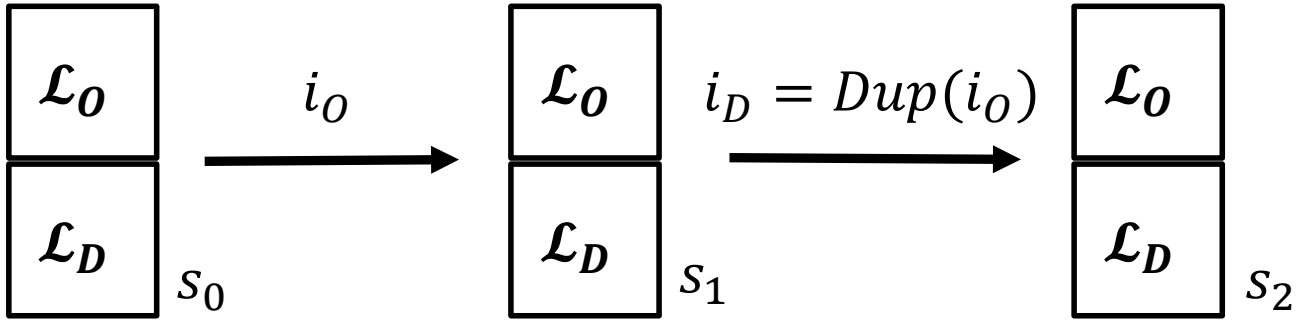
- Executing instruction i: read from input locations in s_0.
- Update output location in s_1 by opcode.

Formal Model of Processors and SQED



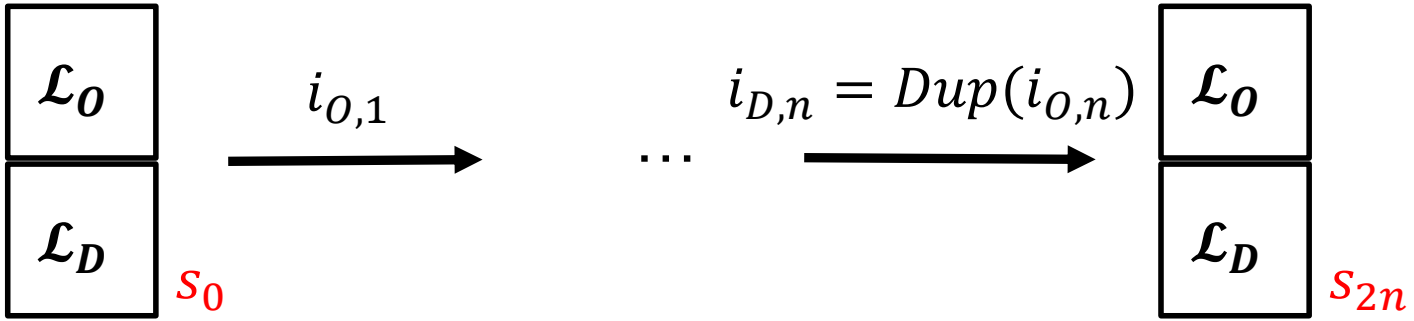
- Original locations $\mathcal{L}_O$.
- Duplicate locations $\mathcal{L}_D$.
- Arbitrary, fixed bijective mapping $\mathrm{L_D} : \mathcal{L}_O \rightarrow \mathcal{L}_D$.

Formal Model of Processors and SQED



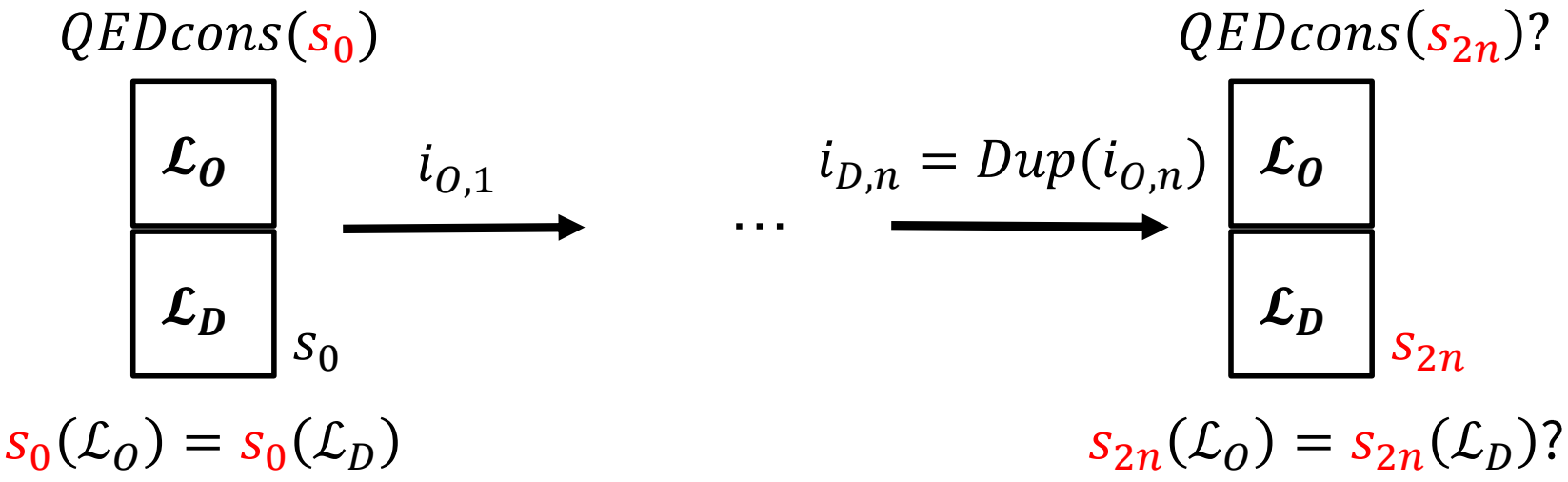
- Original instruction i_O: read/write $\boldsymbol{\mathcal{L}_O}$ only.
- Duplicate i_D: same opcode, read/write $\boldsymbol{\mathcal{L}_D}$ only.

Formal Model of Processors and SQED



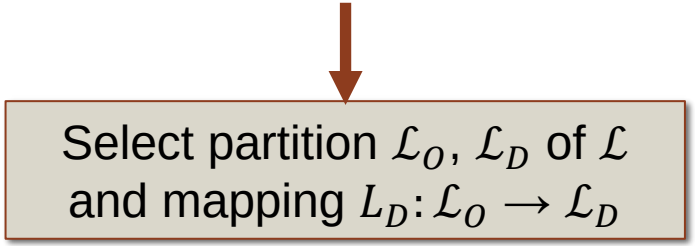
- $\boldsymbol{i}_O = i_{O,1}, \ldots, i_{O,n}$, $\boldsymbol{i}_D = i_{D,1}, \ldots, i_{D,n}$, $\boldsymbol{i}_D = Dup(\boldsymbol{i}_O)$.
- QED test: concatenation $\boldsymbol{i} = \boldsymbol{i}_O :: \boldsymbol{i}_D$ of $2n$ instructions.

Formal Model of Processors and SQED

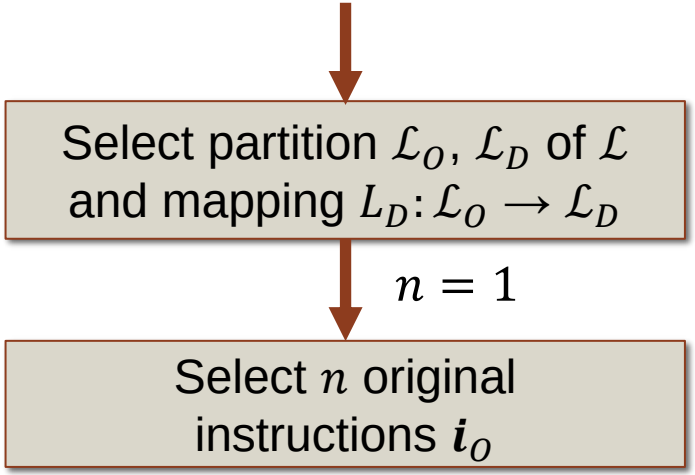


- $\boldsymbol{i}_O = i_{O,1}, \ldots, i_{O,n}$, $\boldsymbol{i}_D = i_{D,1}, \ldots, i_{D,n}$, $\boldsymbol{i}_D = Dup(\boldsymbol{i}_O)$.
- QED test: concatenation $\boldsymbol{i} = \boldsymbol{i}_O :: \boldsymbol{i}_D$ of $2n$ instructions.
- QED-consistent state: matching values at $\mathcal{L}_O$ and $\mathcal{L}_D$.

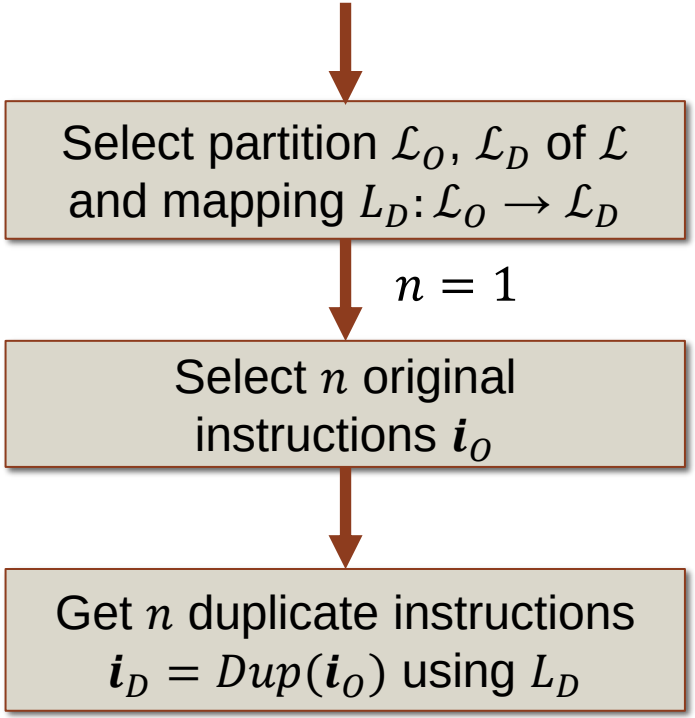
Using BMC in SQED



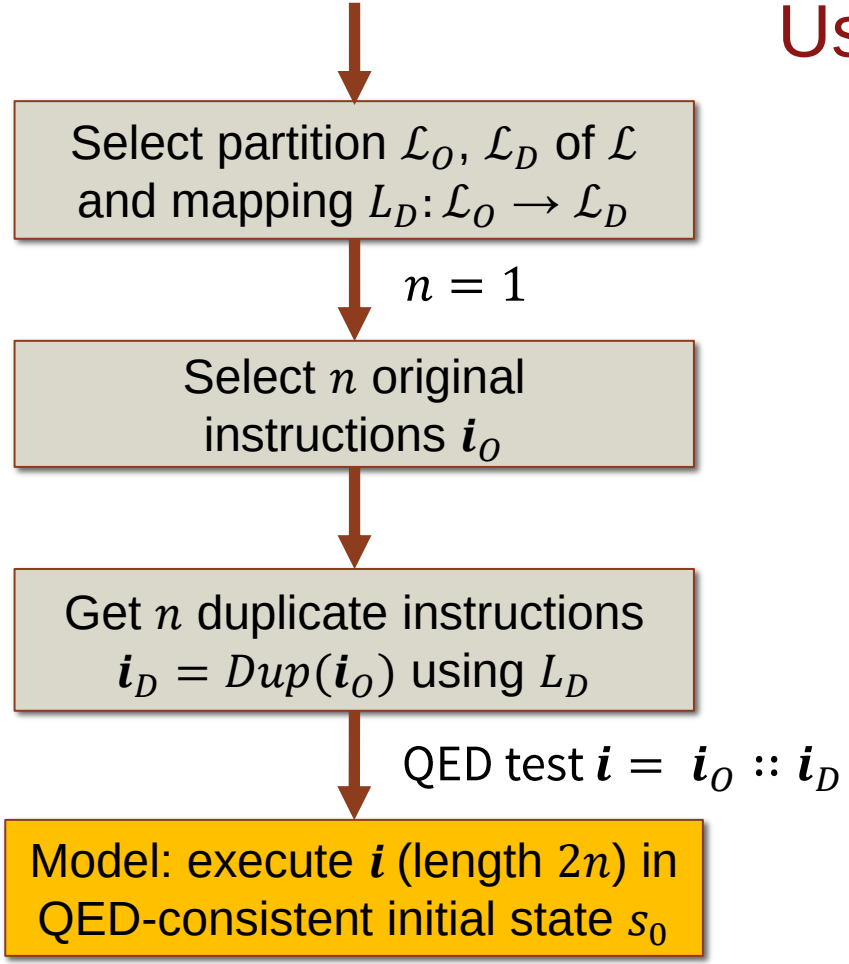
Using BMC in SQED



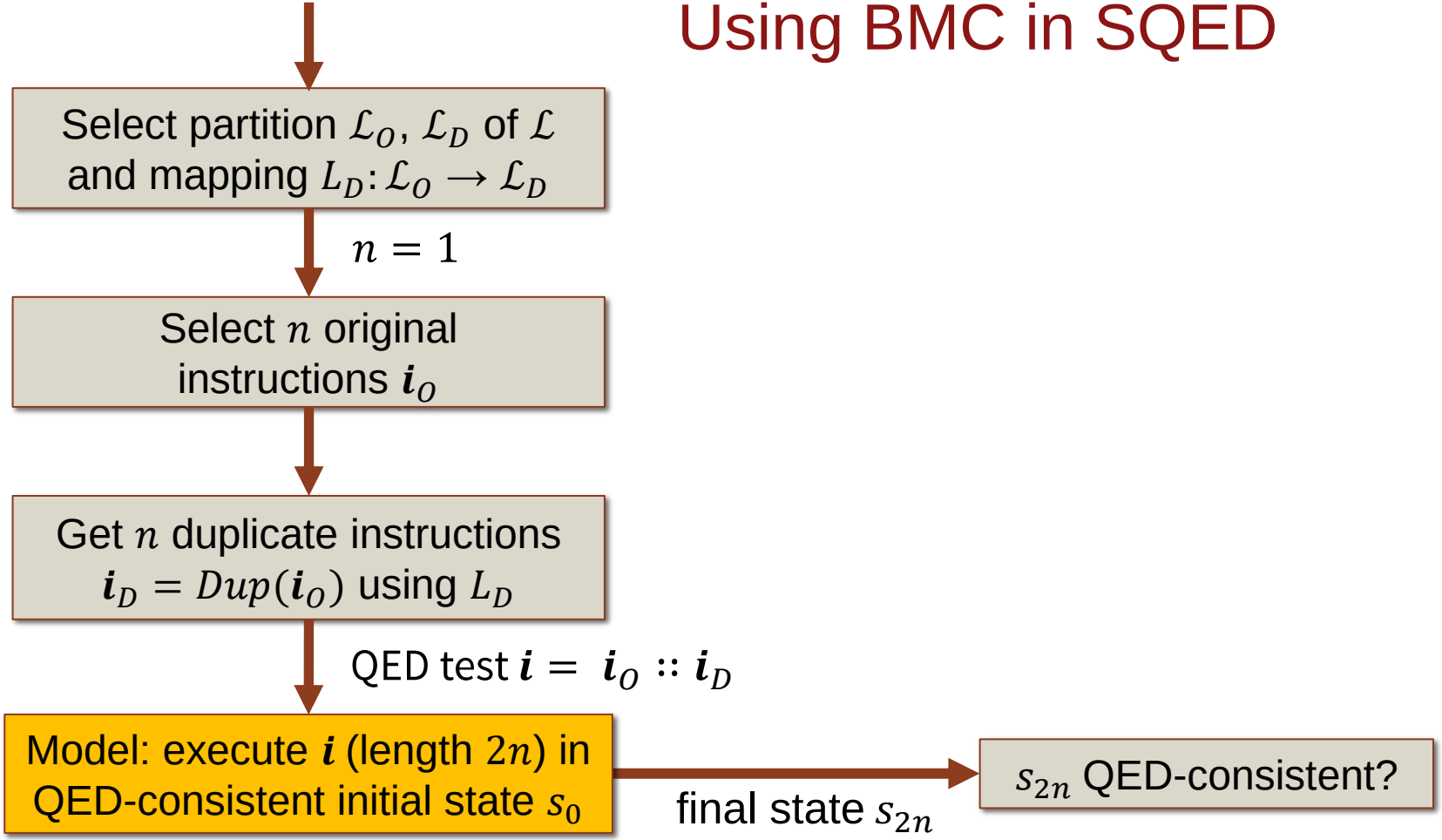
Using BMC in SQED



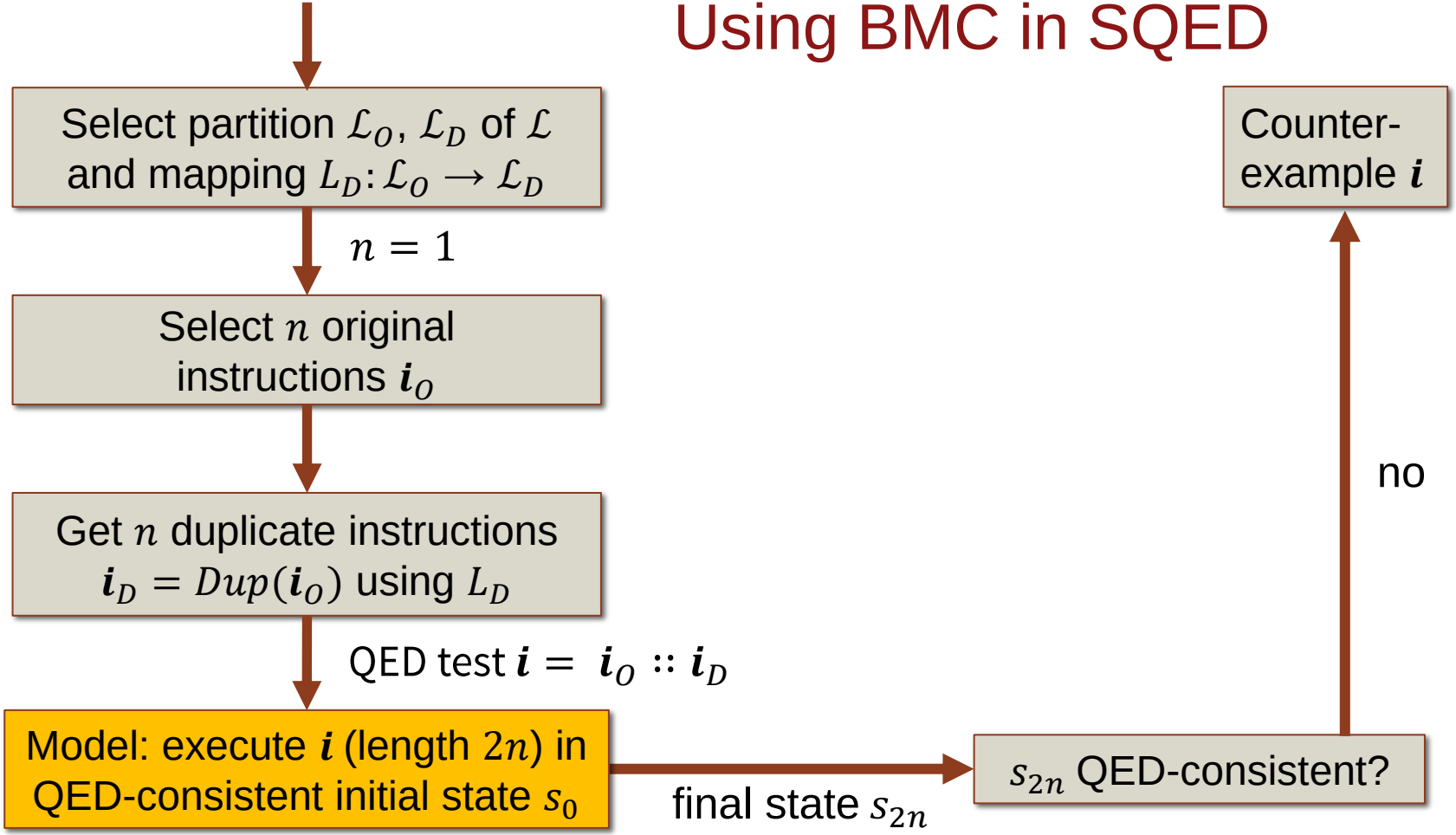
Using BMC in SQED



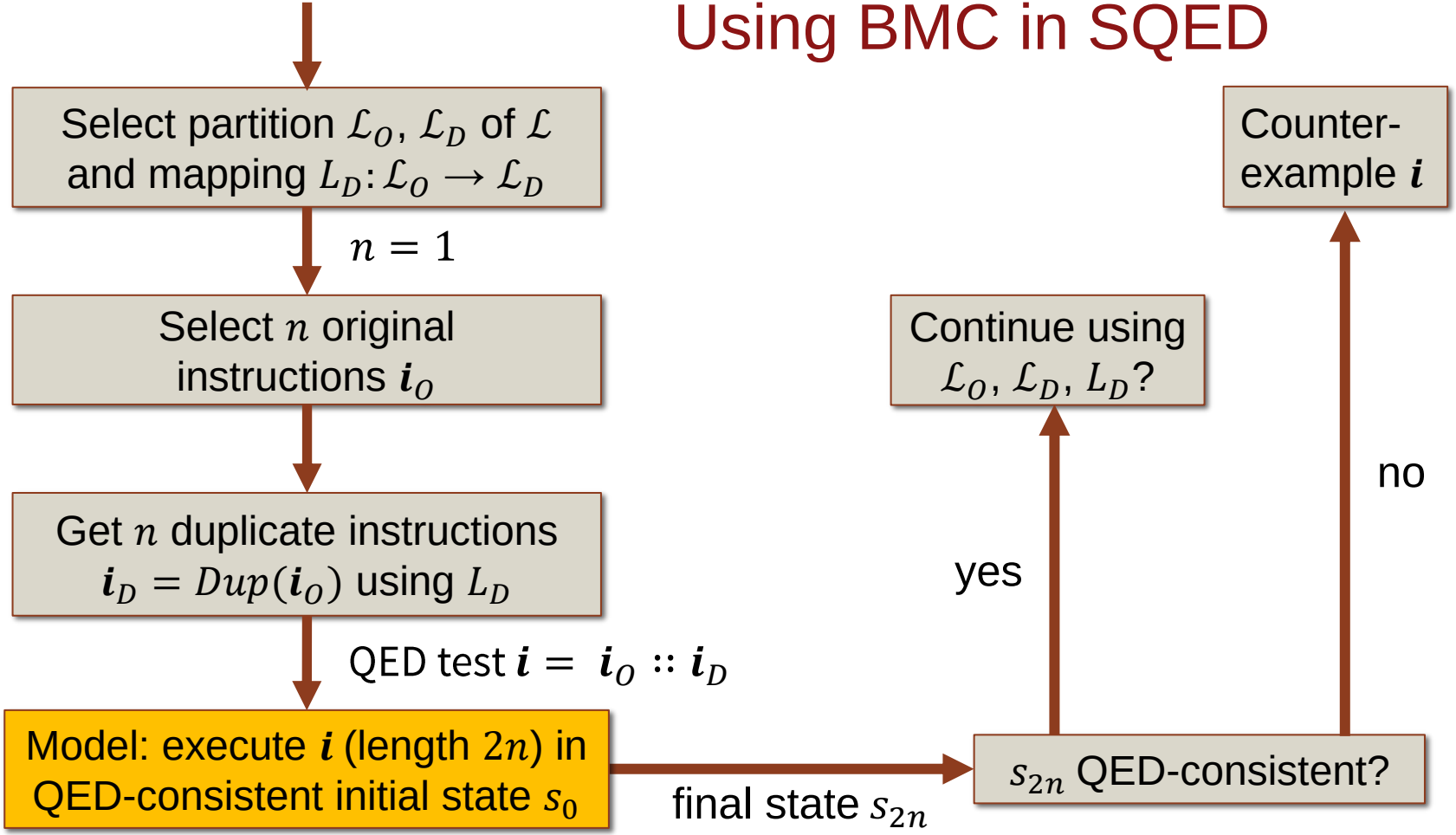
Using BMC in SQED



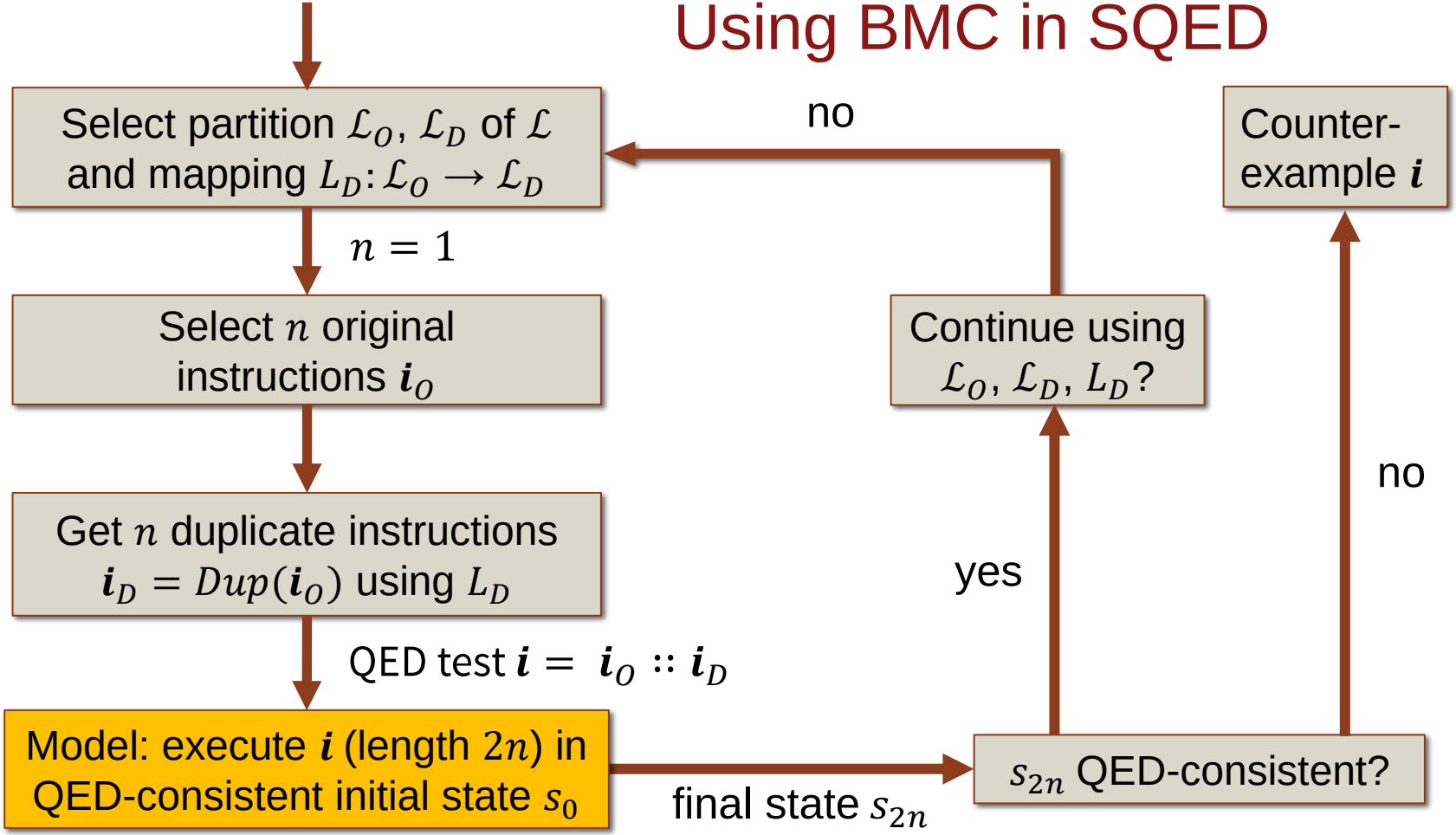
Using BMC in SQED



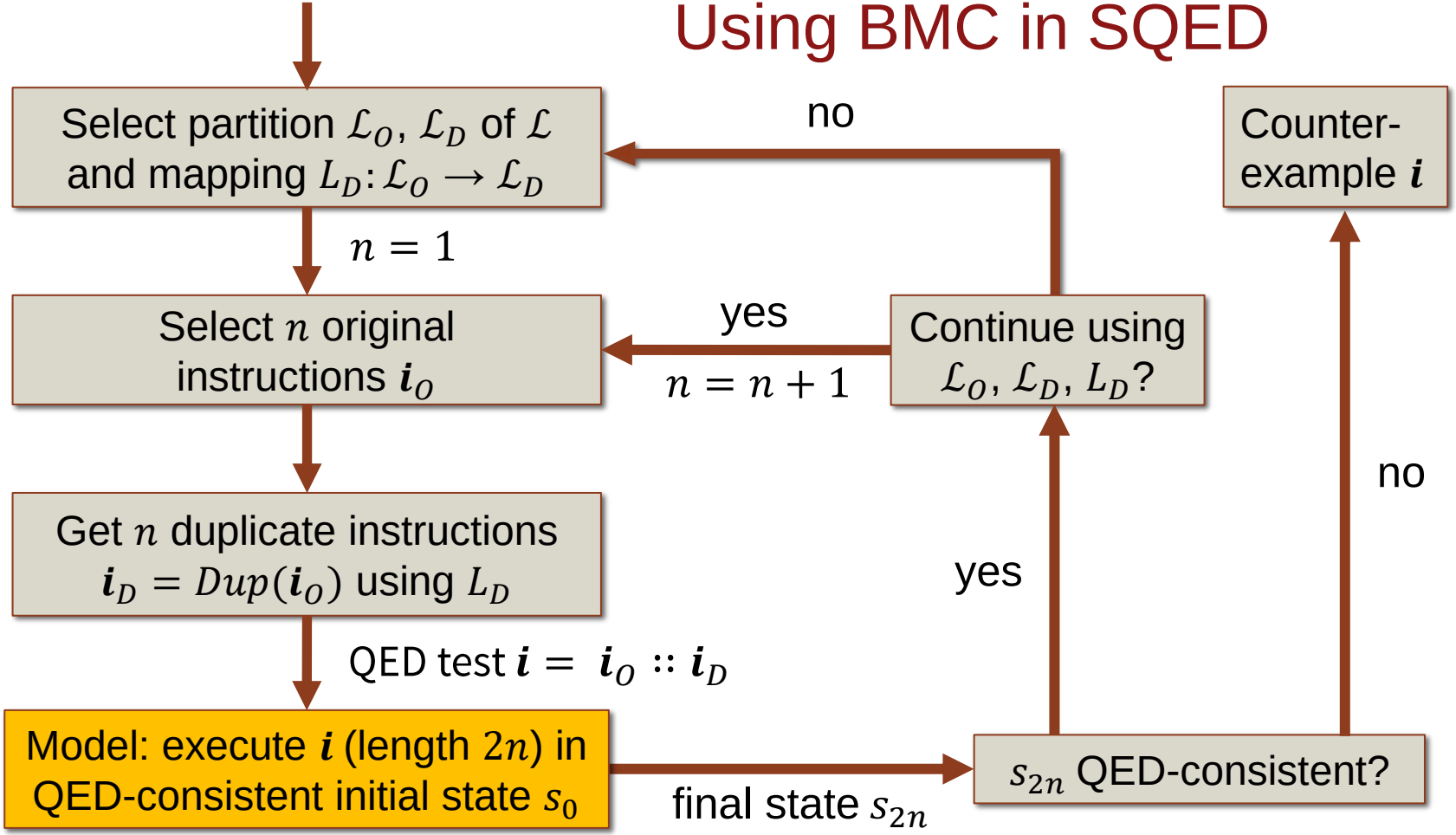
Using BMC in SQED



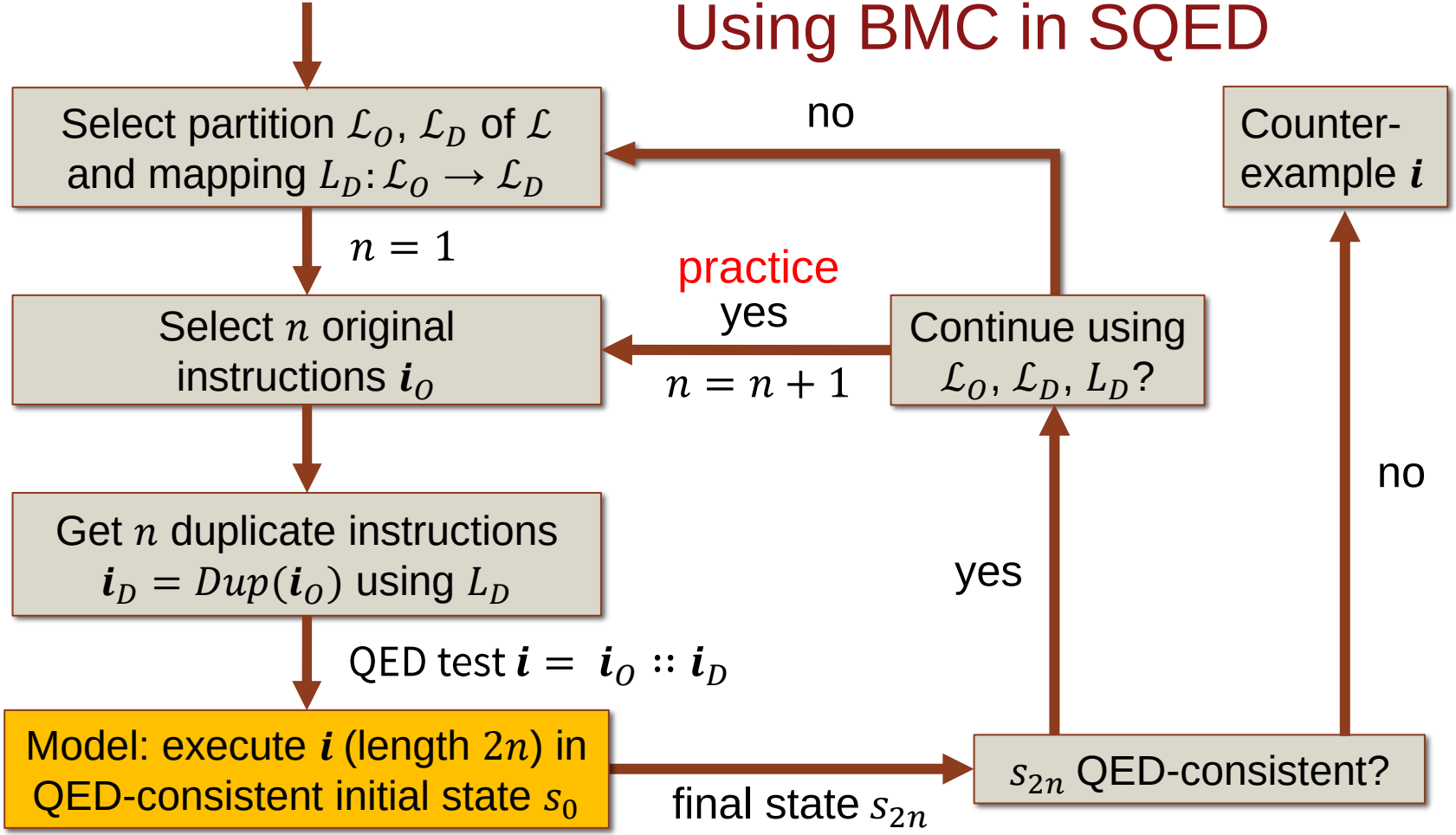
Using BMC in SQED



Using BMC in SQED



Using BMC in SQED



Processor Correctness

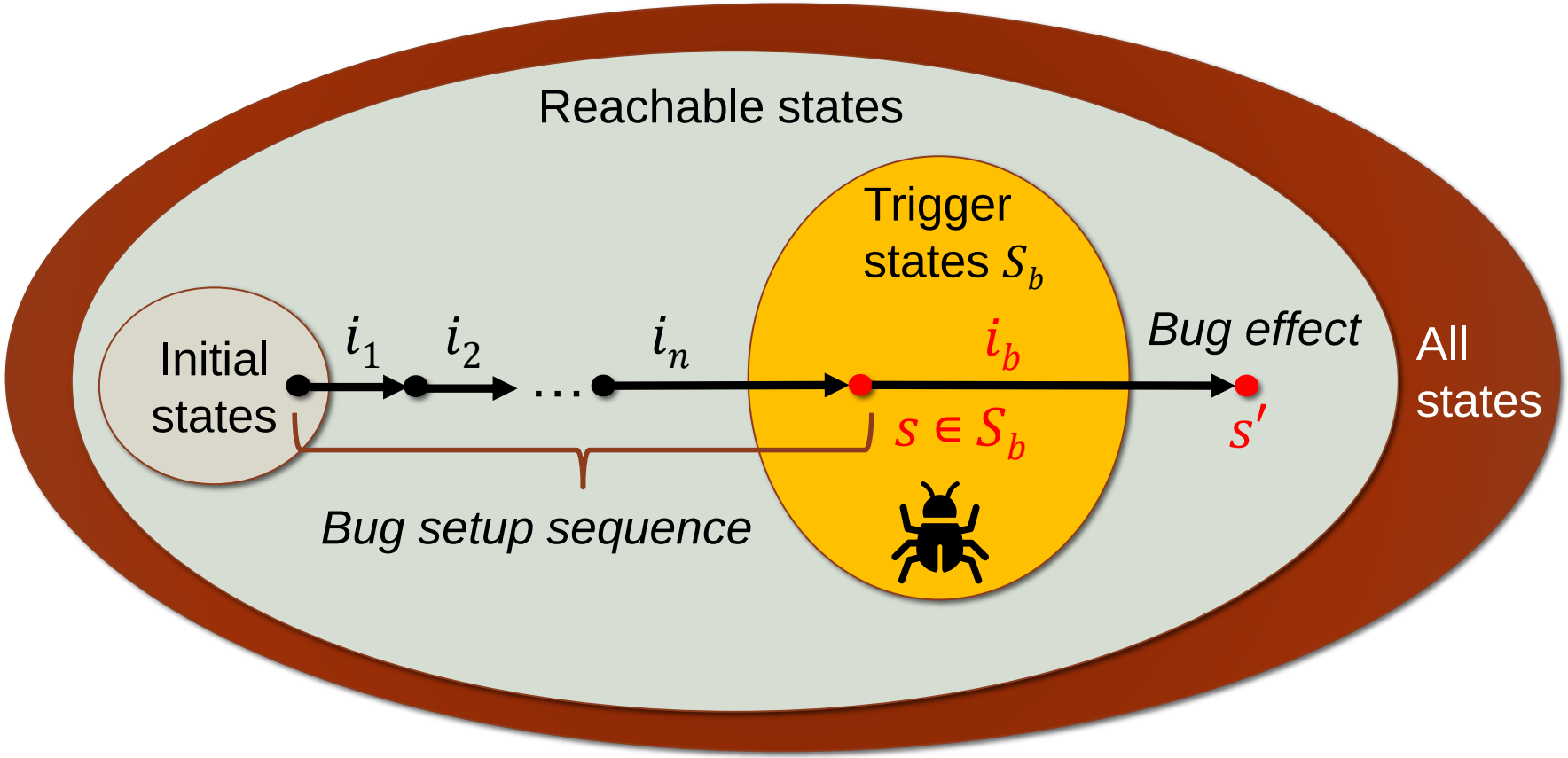
Processor P is correct:

- $\forall s \in S, i \in I.\ reach(s) \rightarrow Spec(s, i, T(s, i))$.
- All instructions execute correctly in all reachable states.

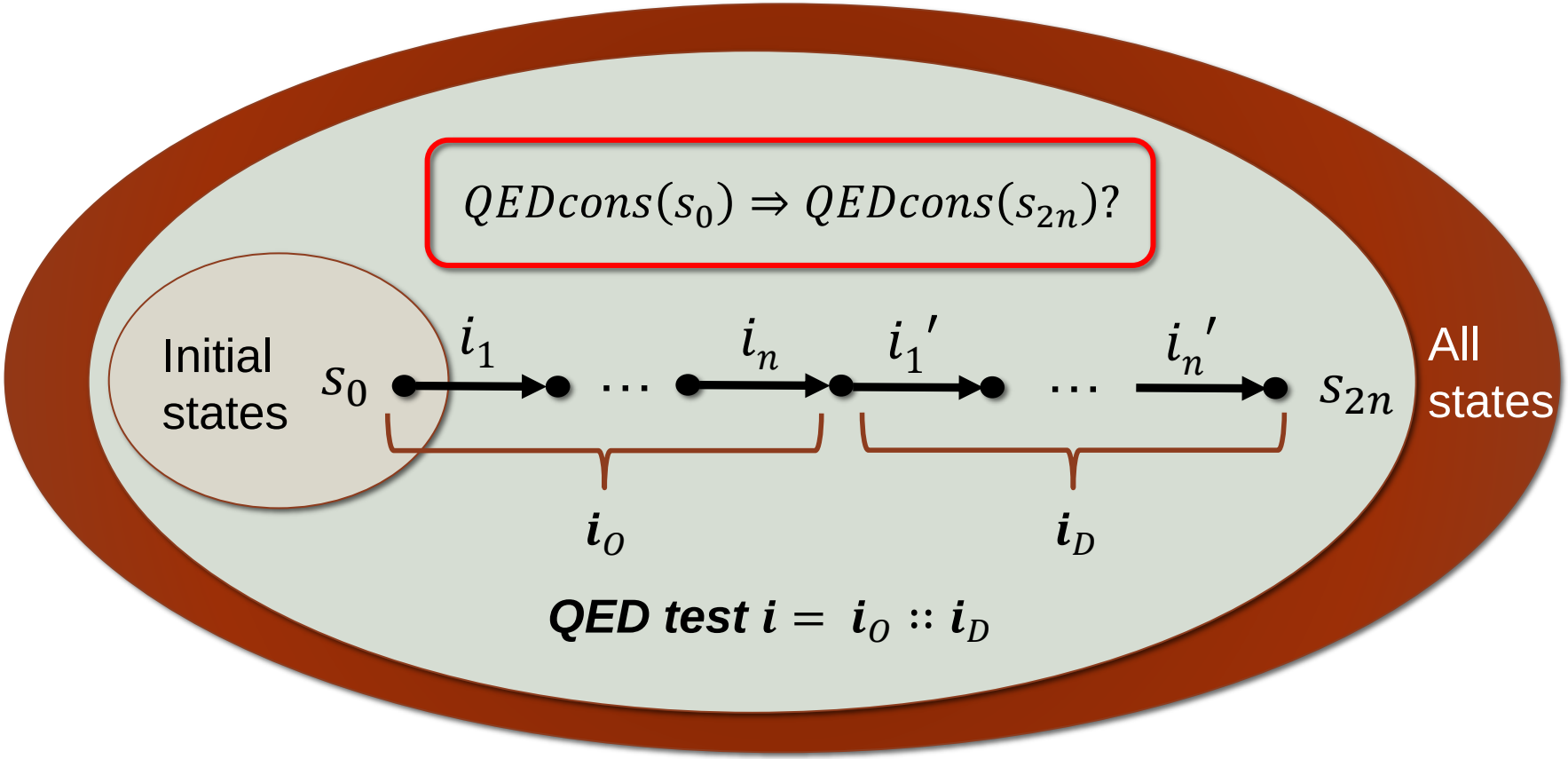
Abstract specification $Spec$ (theory only):

1. Correct output value product.
2. All non-output locations unchanged.

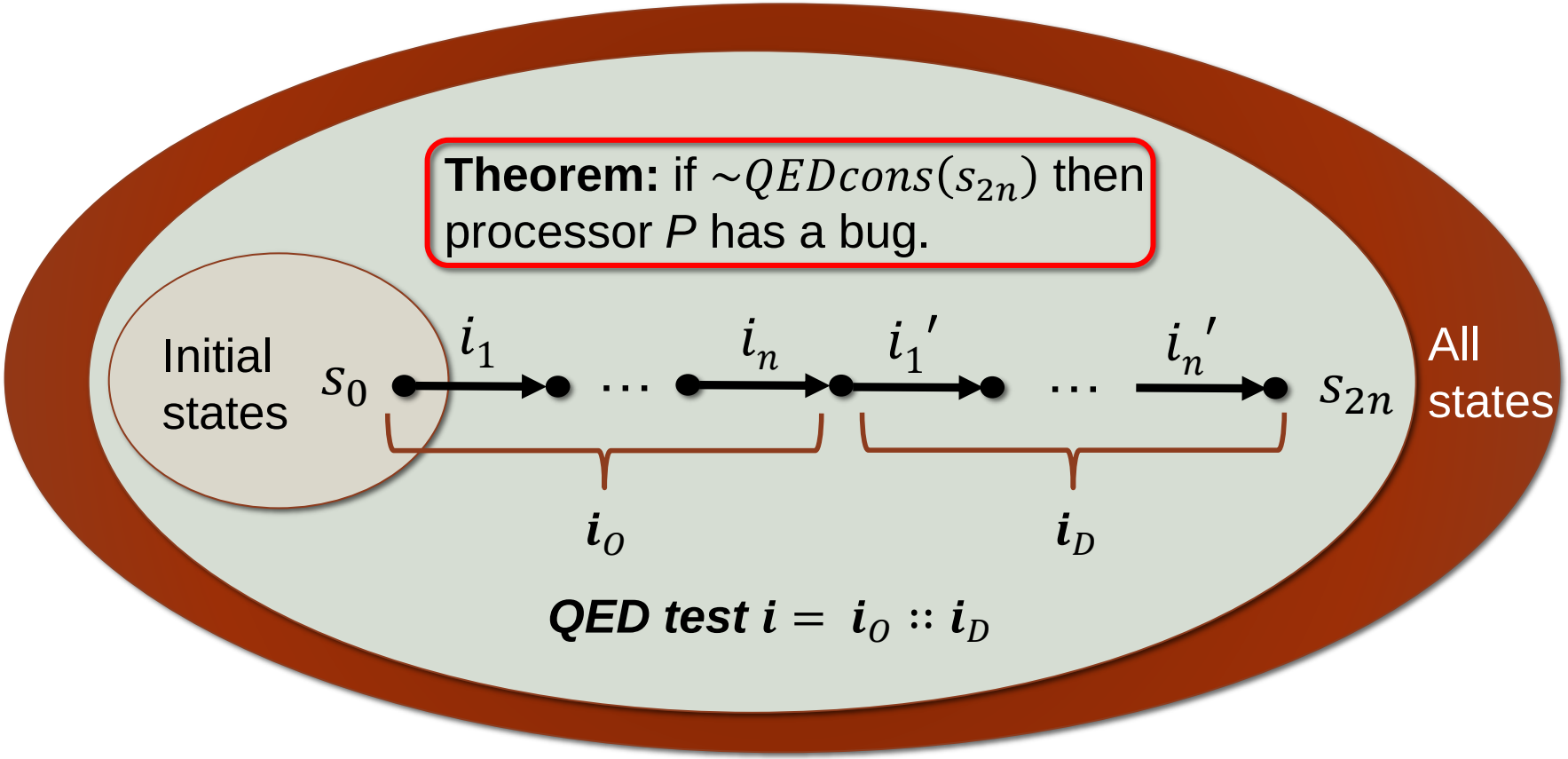
Bugs


Reachable states
Trigger states S_b
Initial states
i_1
i_2
…
i_n
i_b
$s \in S_b$
Bug effect
s'
All states
Bug setup sequence

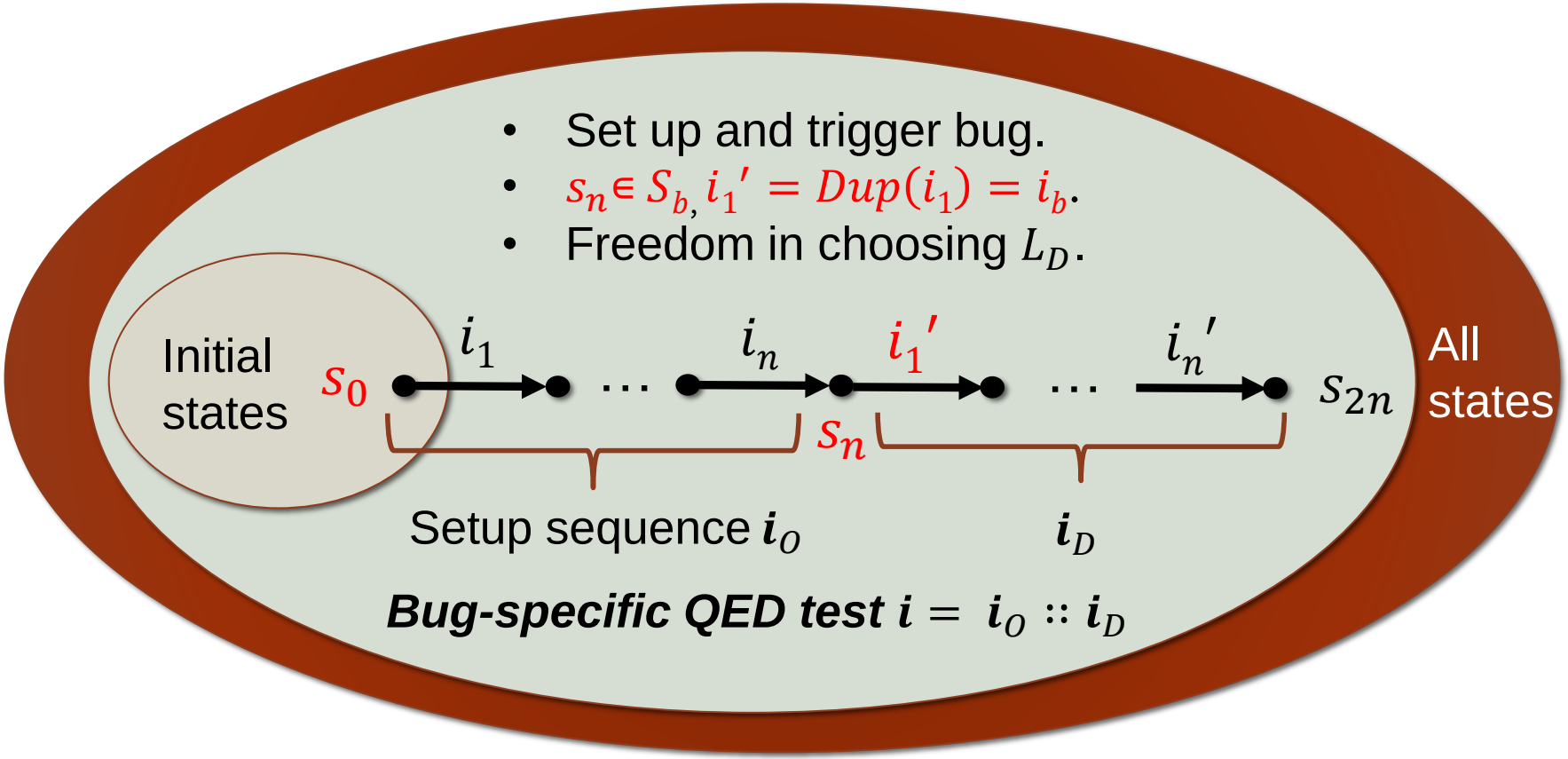
Soundness of SQED



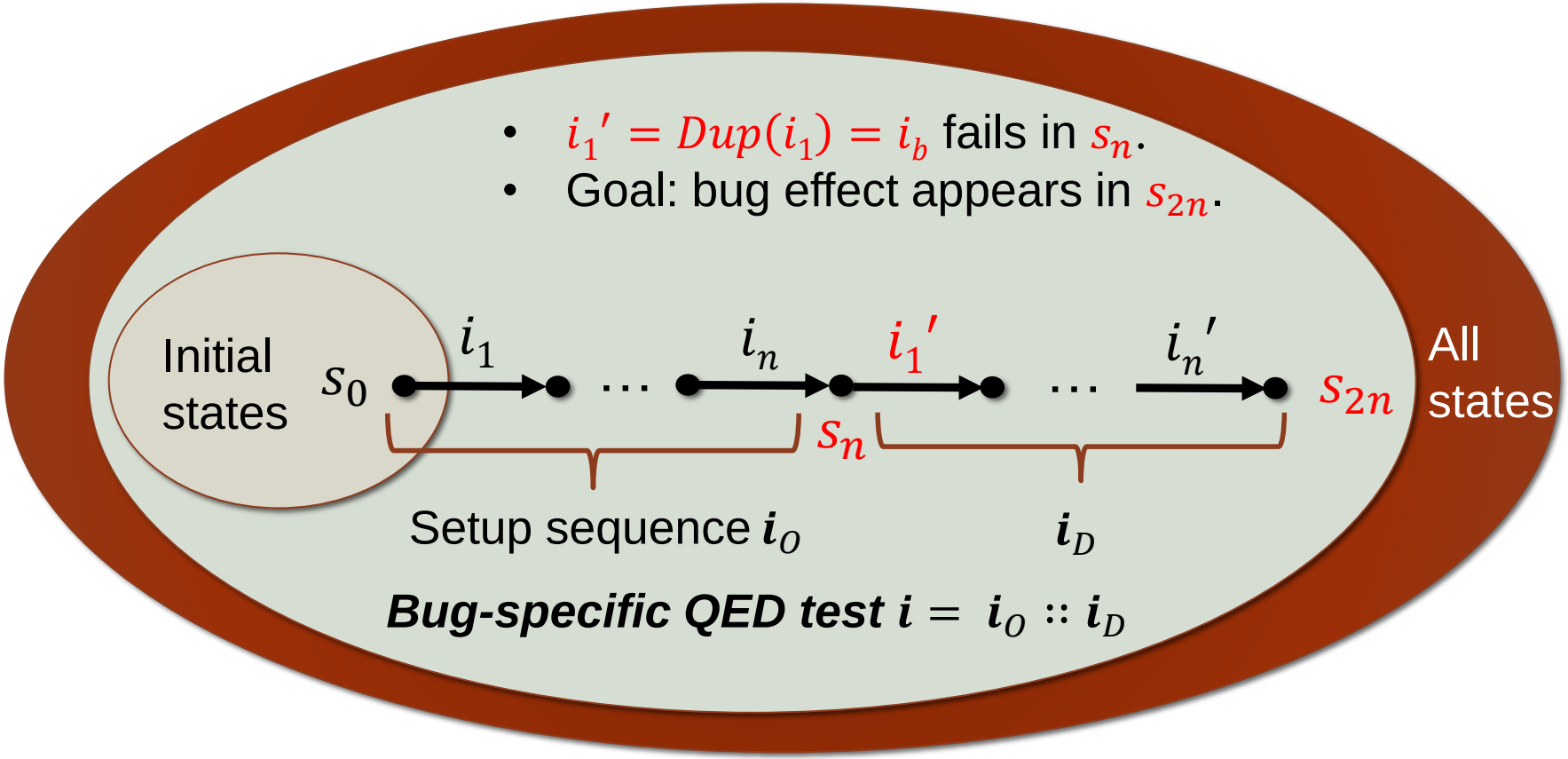
Soundness of SQED



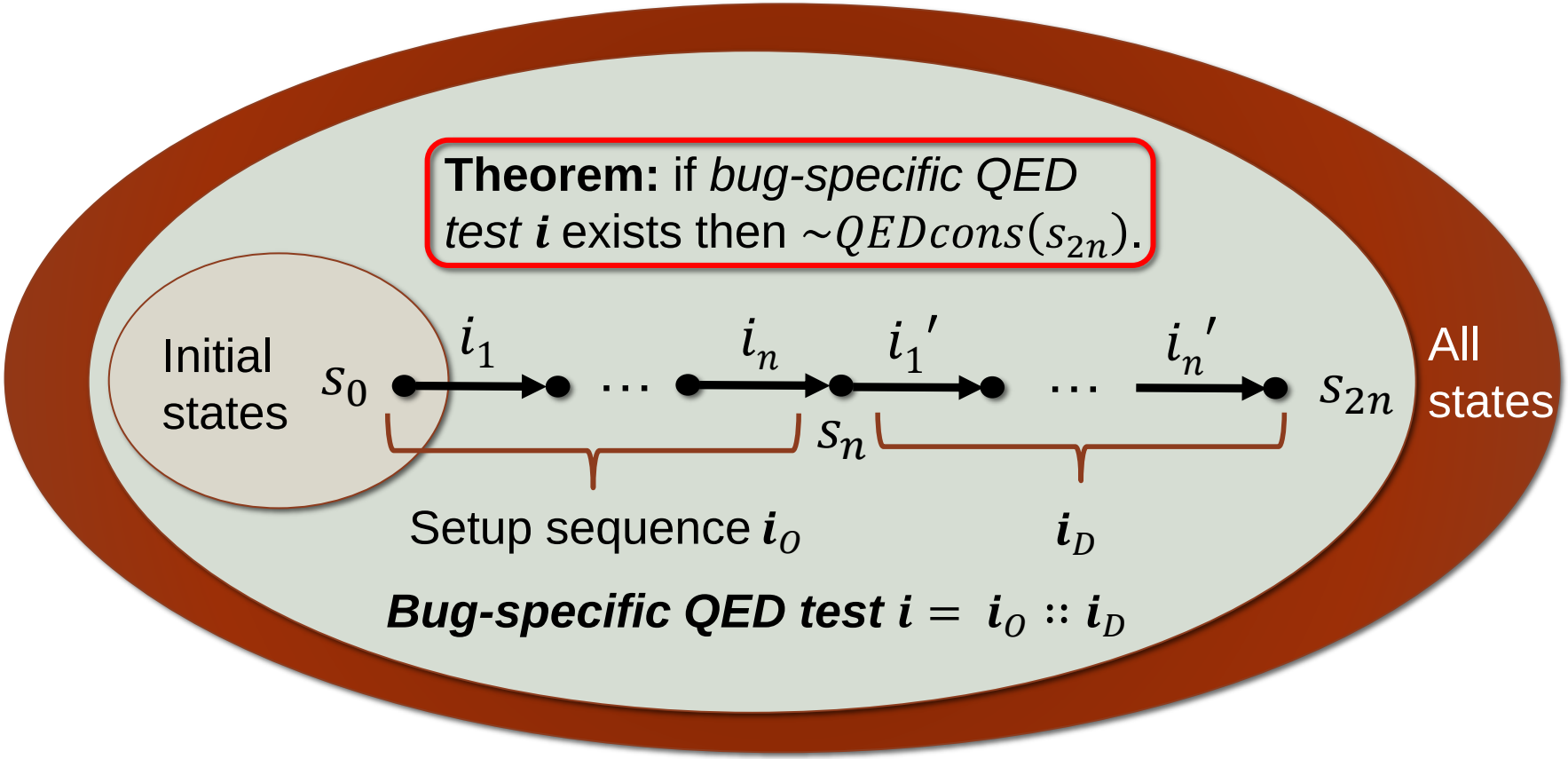
Conditional Completeness of SQED



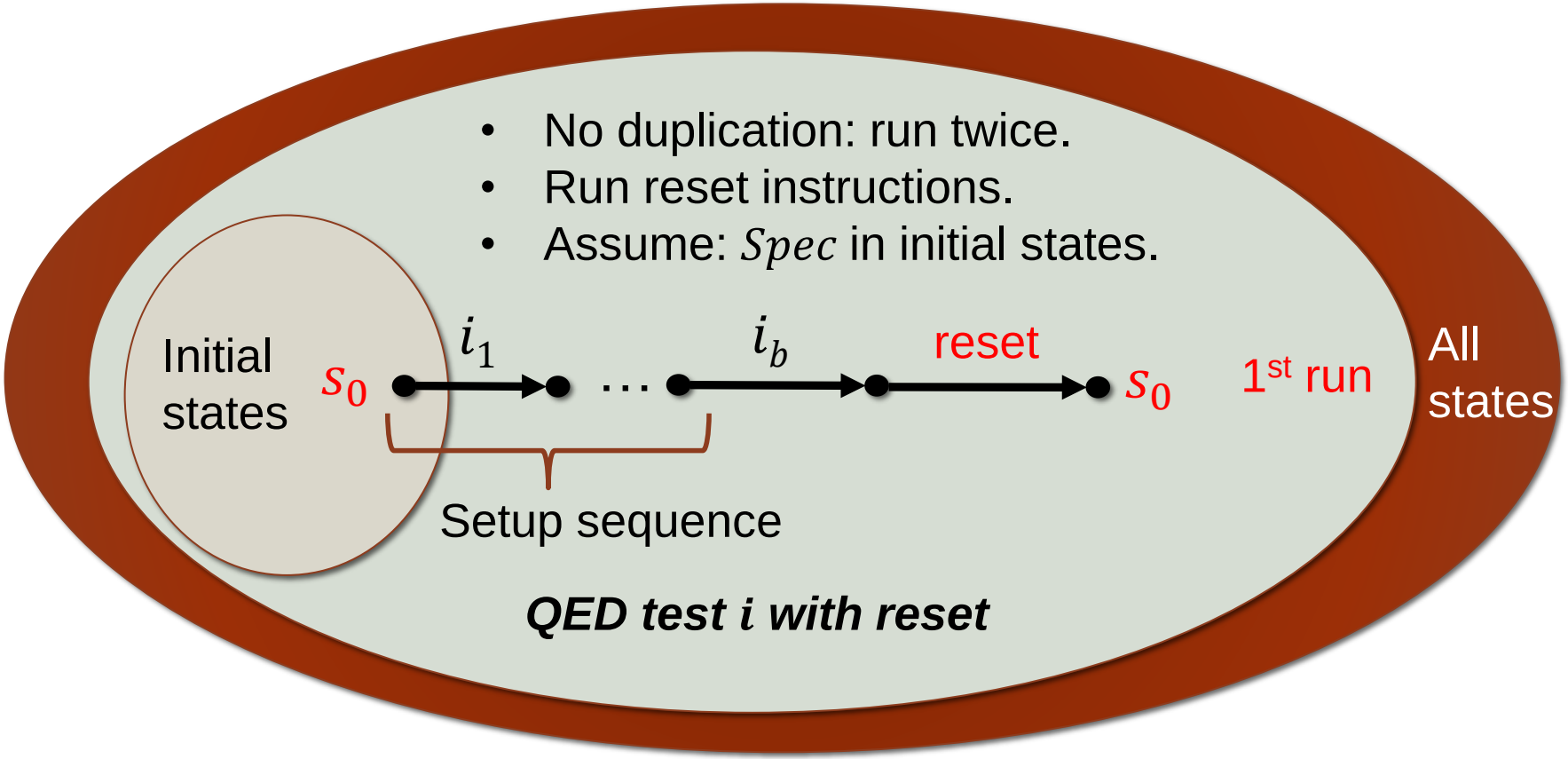
Conditional Completeness of SQED



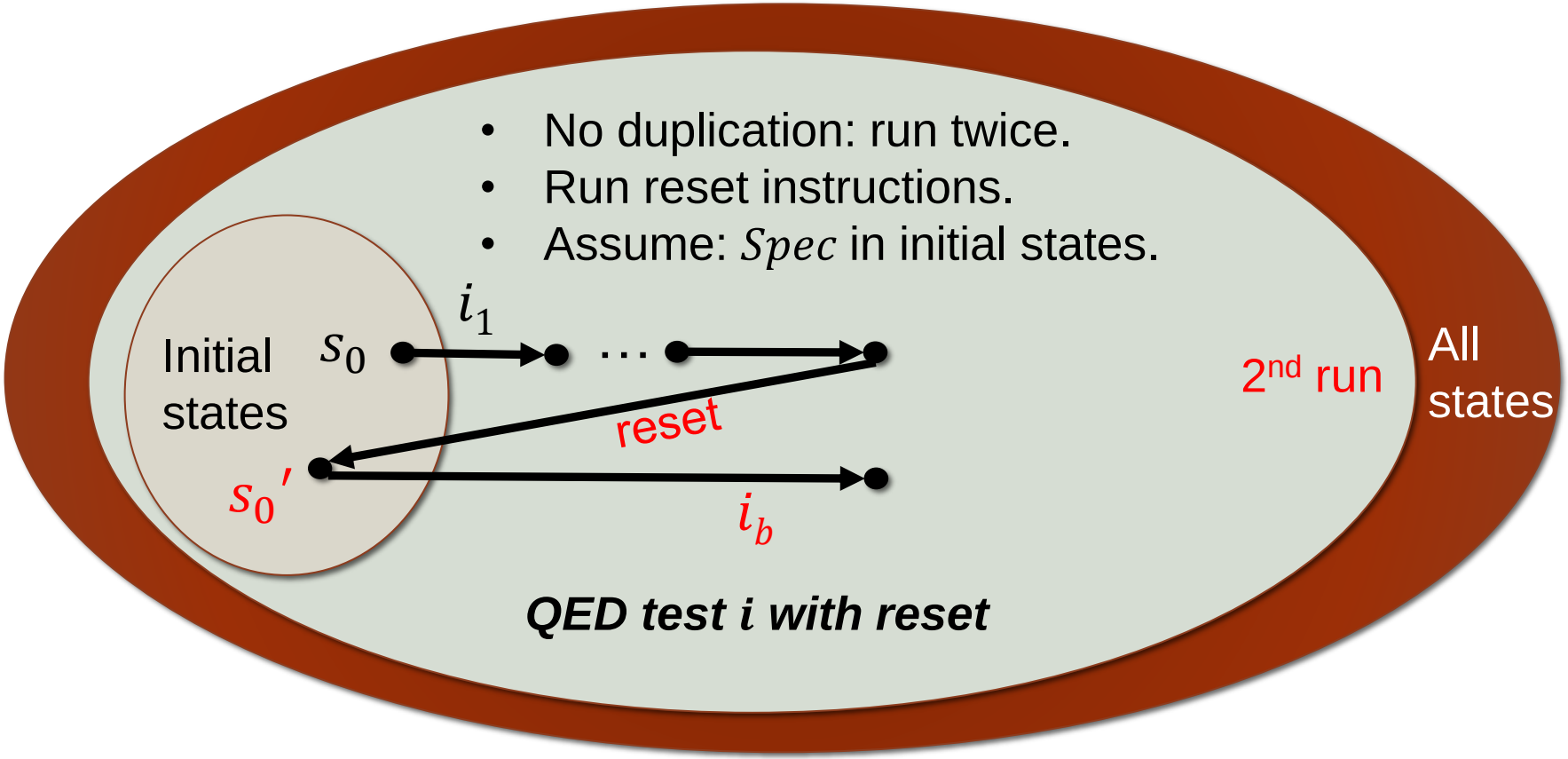
Conditional Completeness of SQED



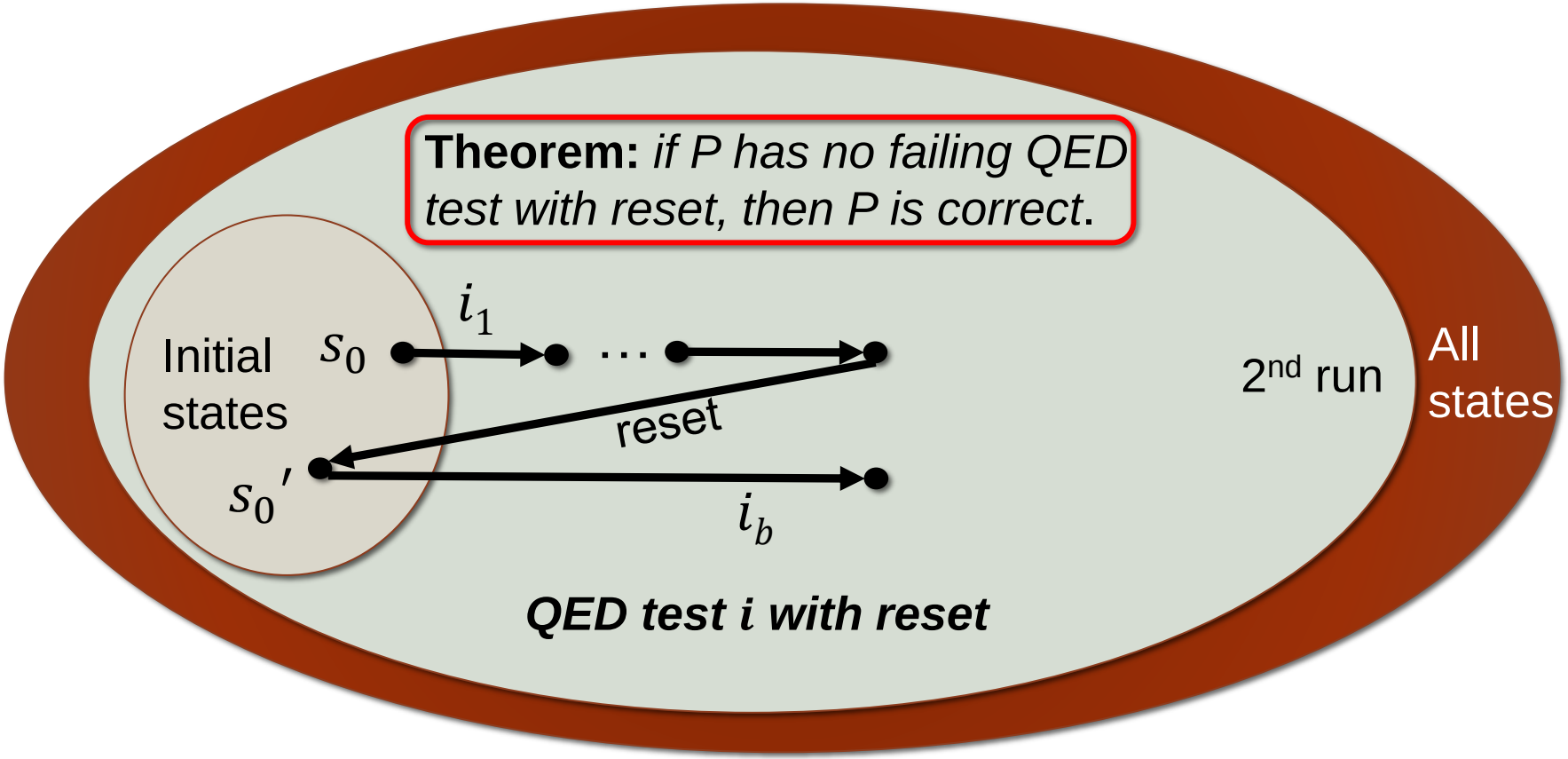
Full Completeness of SQED



Full Completeness of SQED



Full Completeness of SQED



Future Work

Leveraging QED test extensions:

- Soft/hard reset not yet applied in practice.
- Design-for-verification approach.

Formal model refinements:

- Instruction pipelines, multiprocessor systems.
- Deadlock detection.
- Symbolic starting states.

Thank you for your attention!