

Design and Verification of Speed-Independent Multiphase Buck Controller

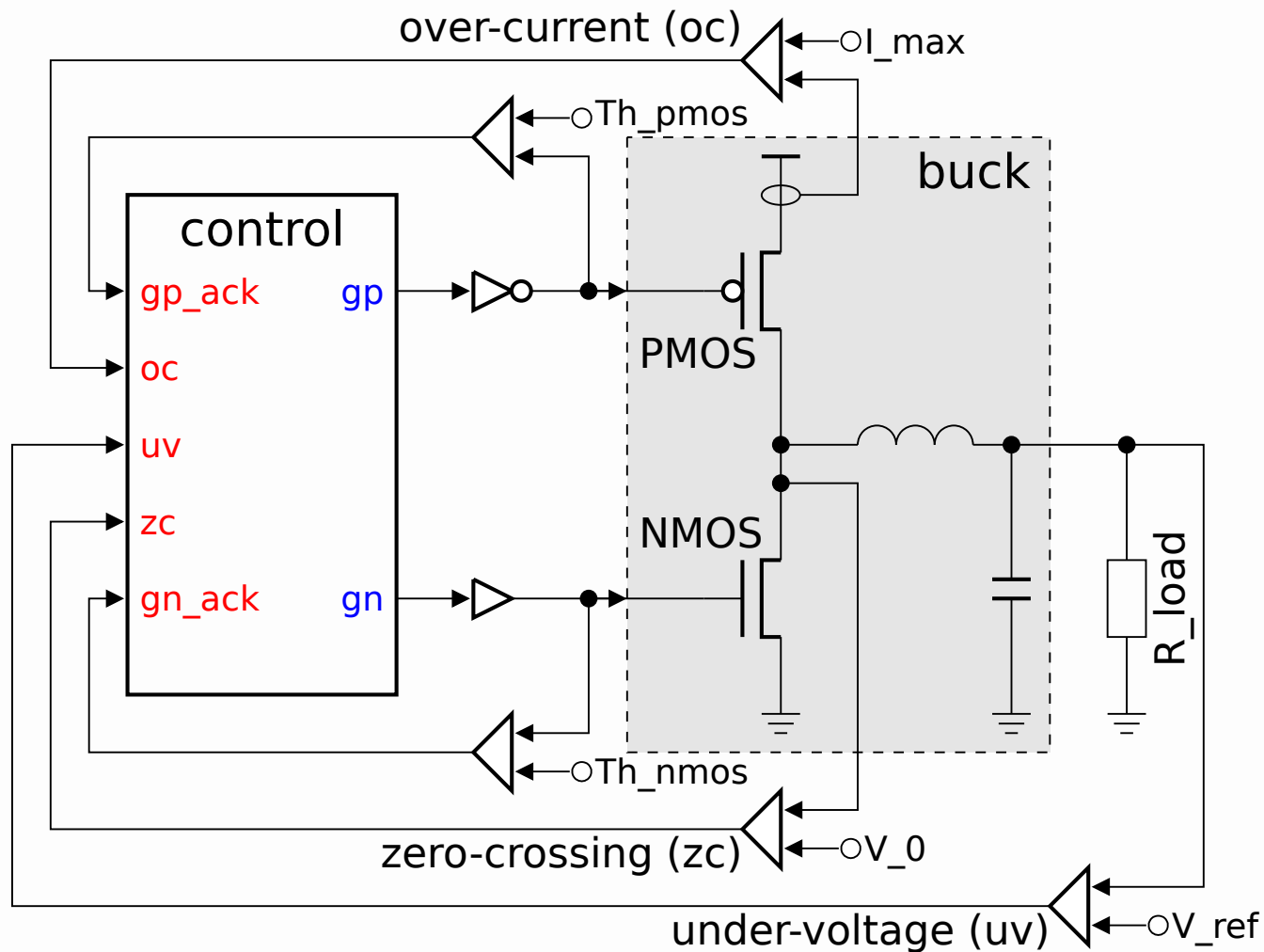
Danil Sokolov¹, Victor Khomenko¹, Andrey Mokhov¹,
Alex Yakovlev¹, David Lloyd²

¹*Newcastle University, UK;* ²*Dialog Semiconductor, UK*

Motivation

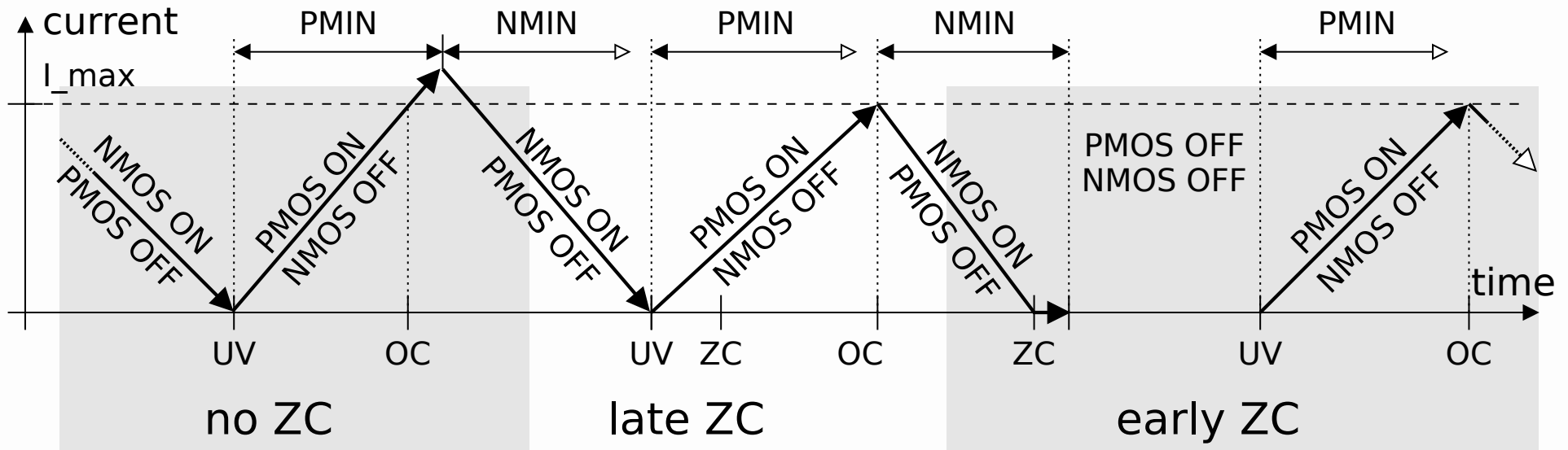
- Efficient implementation of power converters is paramount
 - Extending the battery life of mobile gadgets
 - Reducing the energy bill for PCs and data centres (5+3% of global electricity production)
- Need for responsive and reliable control circuitry - *little digital*
 - Millions of control decisions per second for years
 - An incorrect decision may permanently damage the circuit
- Poor EDA support
 - Synthesis is optimised for data processing - *big digital*
 - *Ad hoc* solutions are prone to errors and cannot be verified

Basic buck converter: Schematic



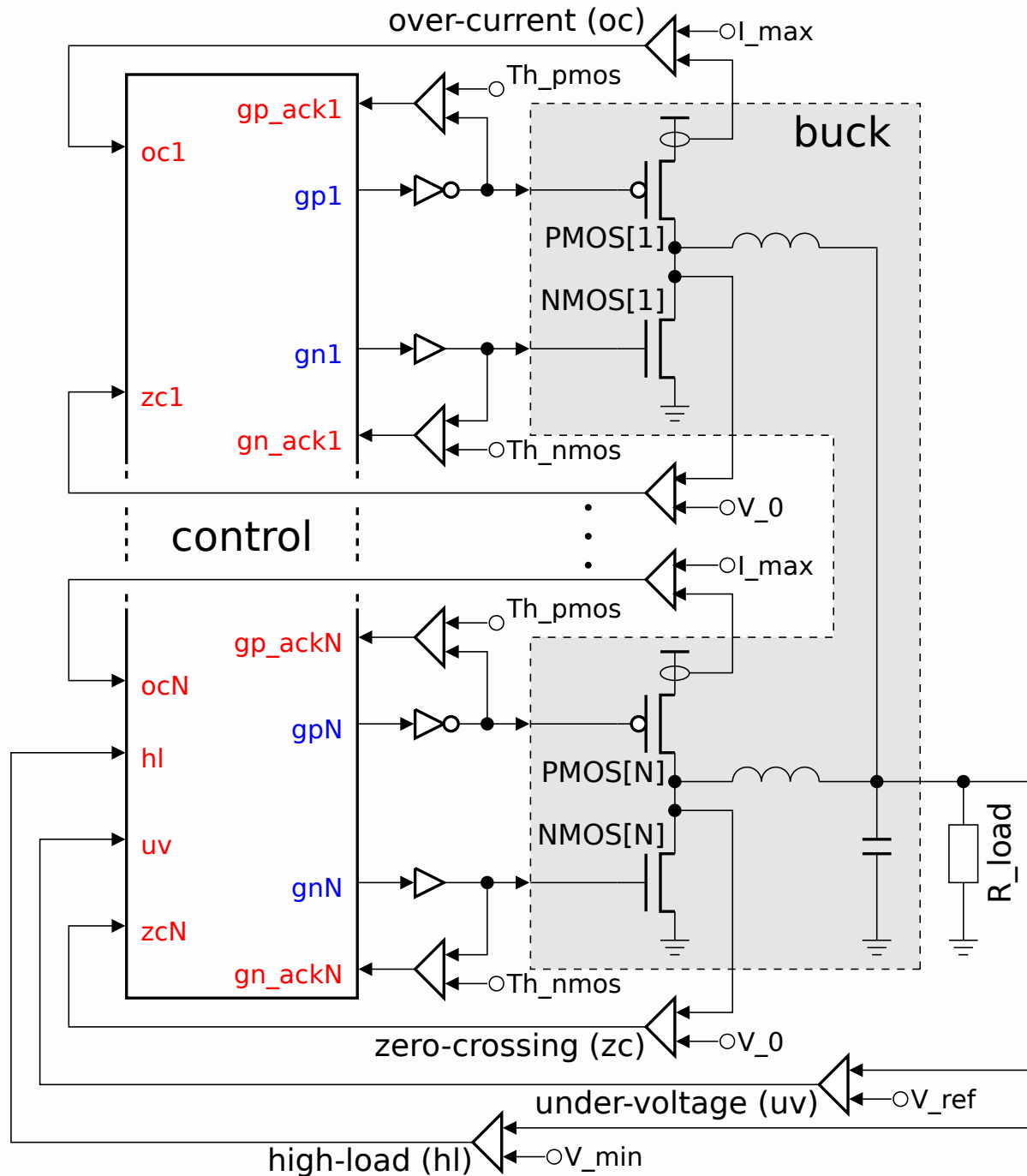
- In the textbook buck a diode is used instead of NMOS transistor

Basic buck converter: Informal specification



- **no ZC** – under-voltage without zero-crossing
- **late ZC** – under-voltage before zero-crossing
- **early ZC** – under-voltage after zero-crossing

Multiphase buck converter: Schematic



Multiphase buck converter: Informal specification

- Normal mode
 - Phases are activated sequentially
 - Phases may overlap
- High-load mode
 - All phases are activated simultaneously
- Benefits
 - Faster reaction to the power demand
 - Heat dissipation from a larger area
 - Decreased ripple of the output voltage
 - Smaller transistors and coils

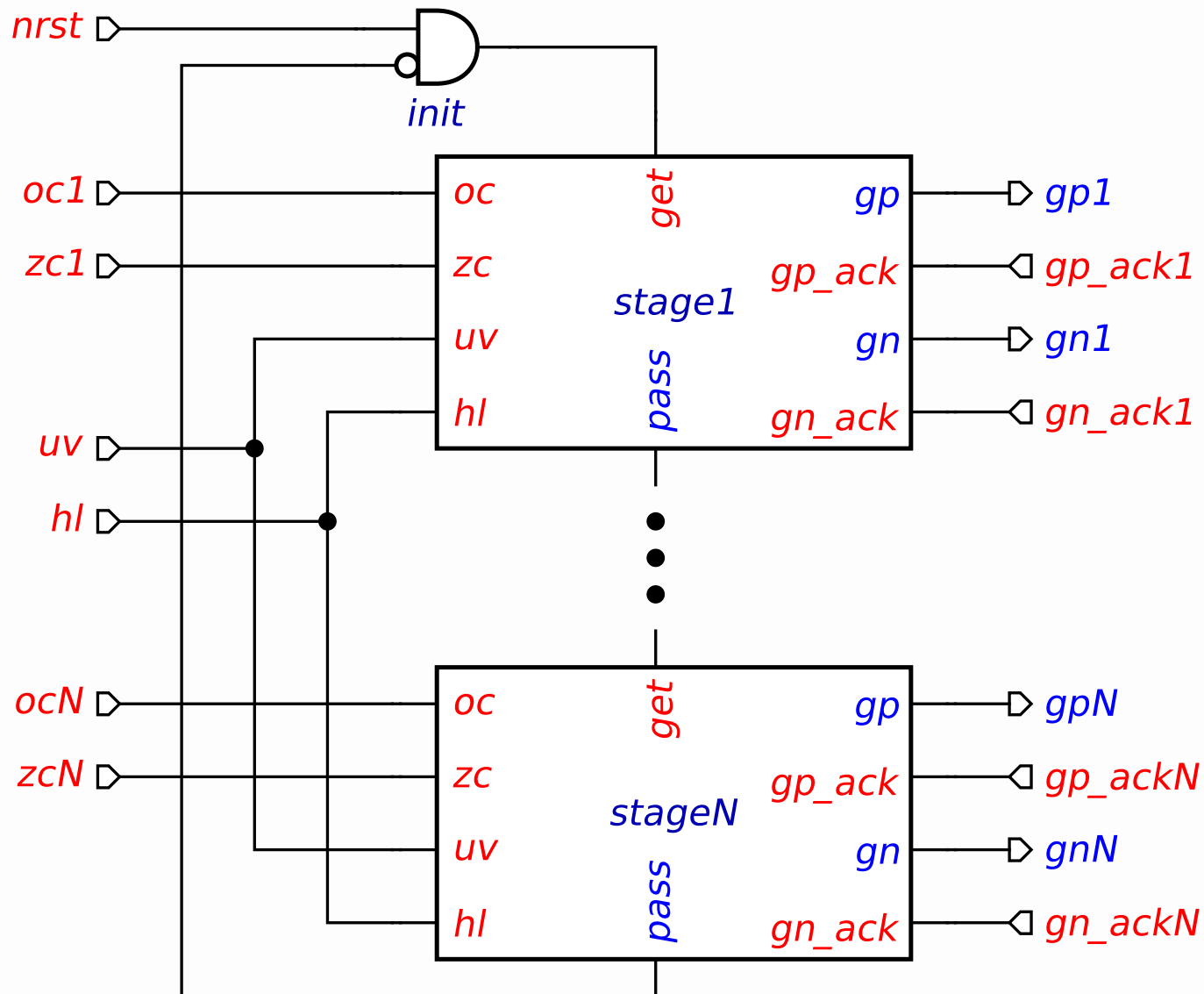
Synchronous design

- Two clocks: phase activation ($\sim 5\text{MHz}$) and sampling ($\sim 100\text{MHz}$)
 - ☺ Easy to design (RTL synthesis flow)
 - ☹ Response time is of the order of clock period
 - ☹ Power consumed even when idle
 - ☹ Non-negligible probability of a synchronisation failure
- Manual ad hoc design to alleviate the disadvantages
 - ☹ Verification by exhaustive simulation

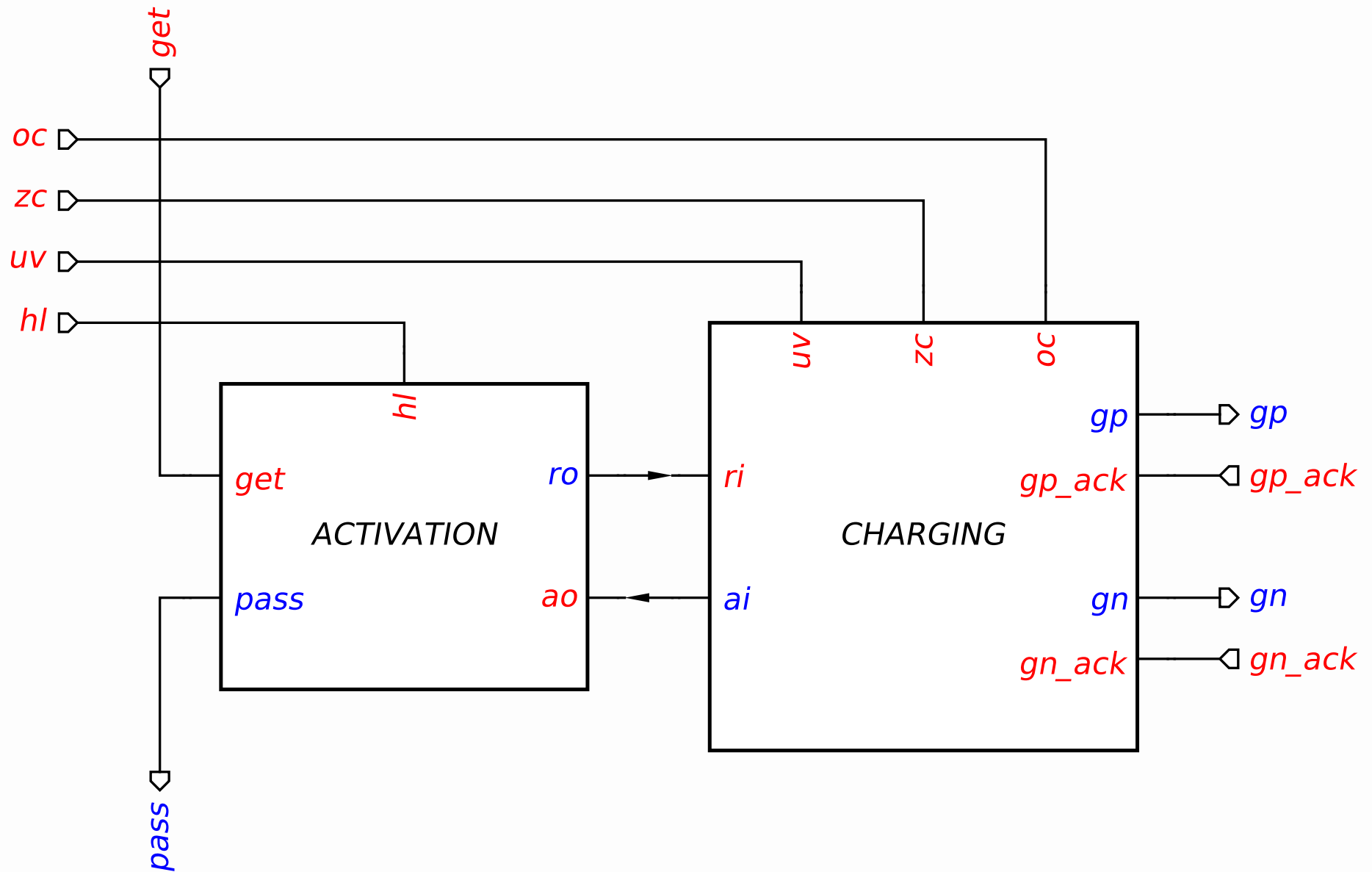
Asynchronous design

- Event-driven control decisions
 - 😊 Prompt response (a delay of few gates)
 - 😊 No dynamic power consumption when the buck is inactive
 - 😊 Other well known advantages
 - 😞 Insufficient methodology and tool support
- Our goals
 - Formal specification of power control behaviour
 - Reuse of existing synthesis methods
 - Formal verification of the obtained circuits
 - Demonstrate new advantages for power regulation (power efficiency, smaller coils, ripple and transient response)

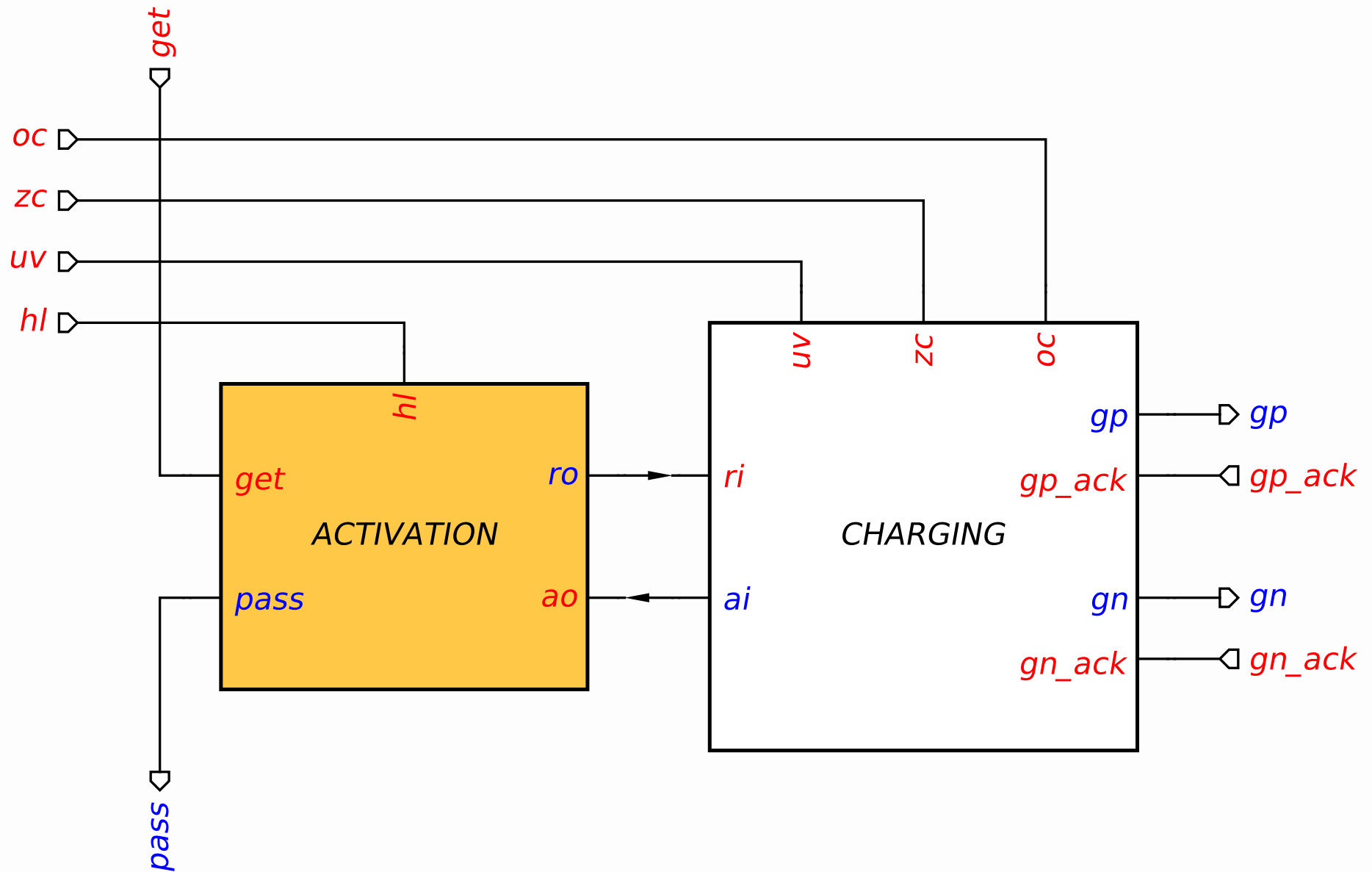
High-level architecture: Token ring



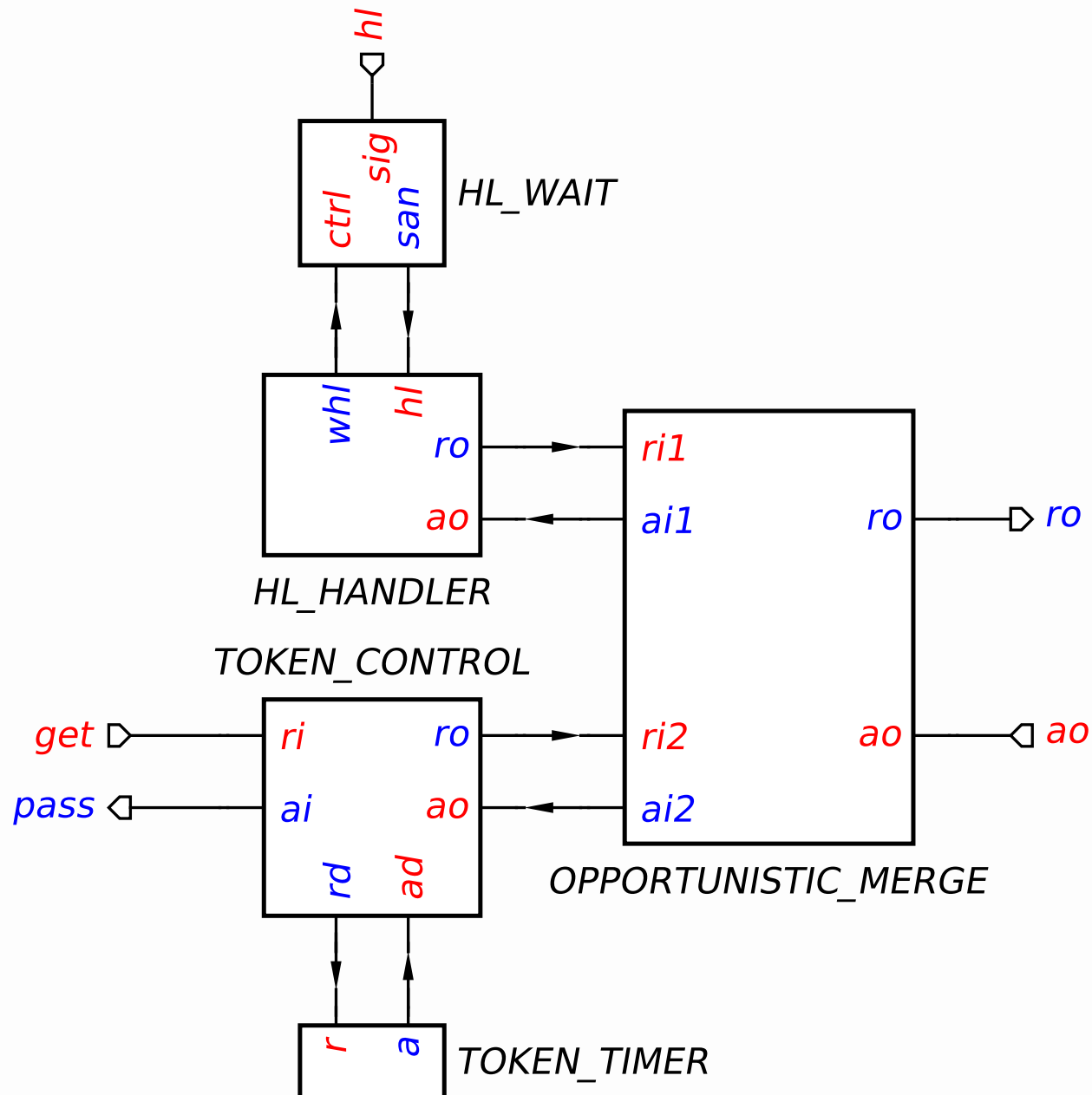
High-level architecture: Stage



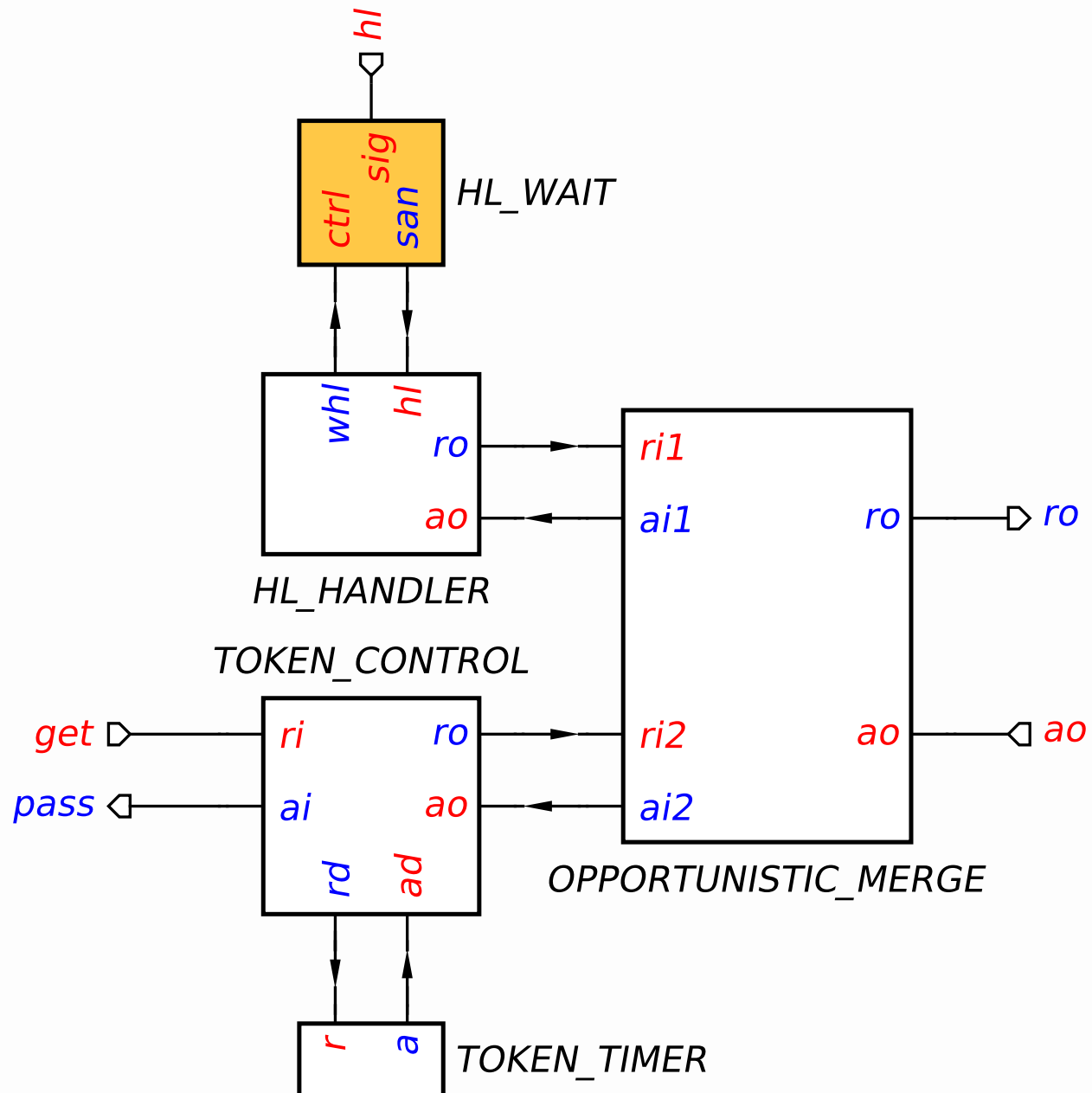
High-level architecture: Activation



High-level architecture: Activation

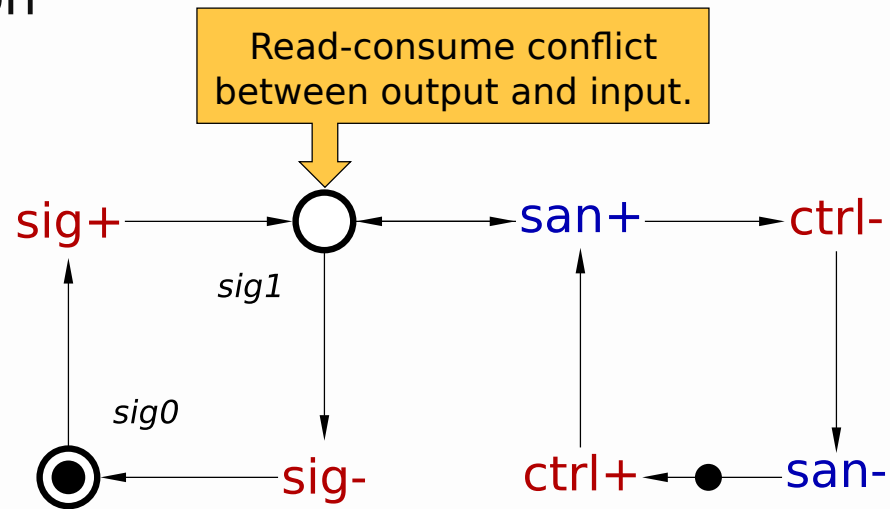


WAIT element

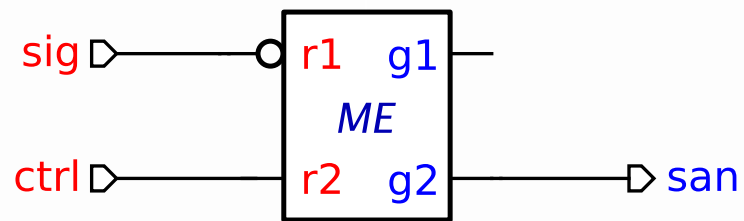


WAIT element

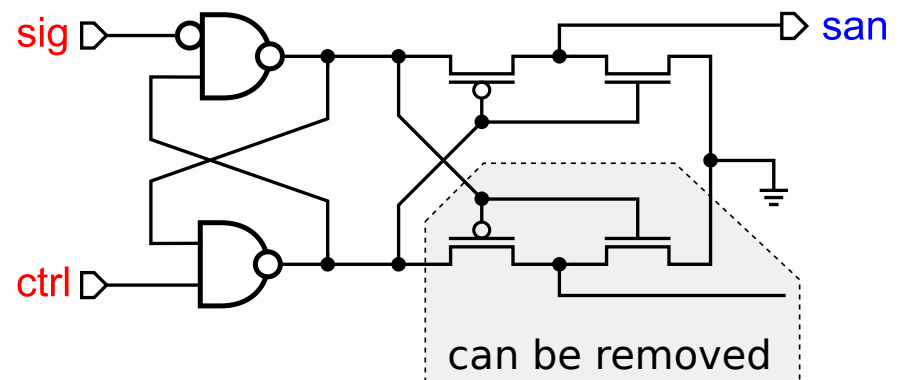
- STG specification



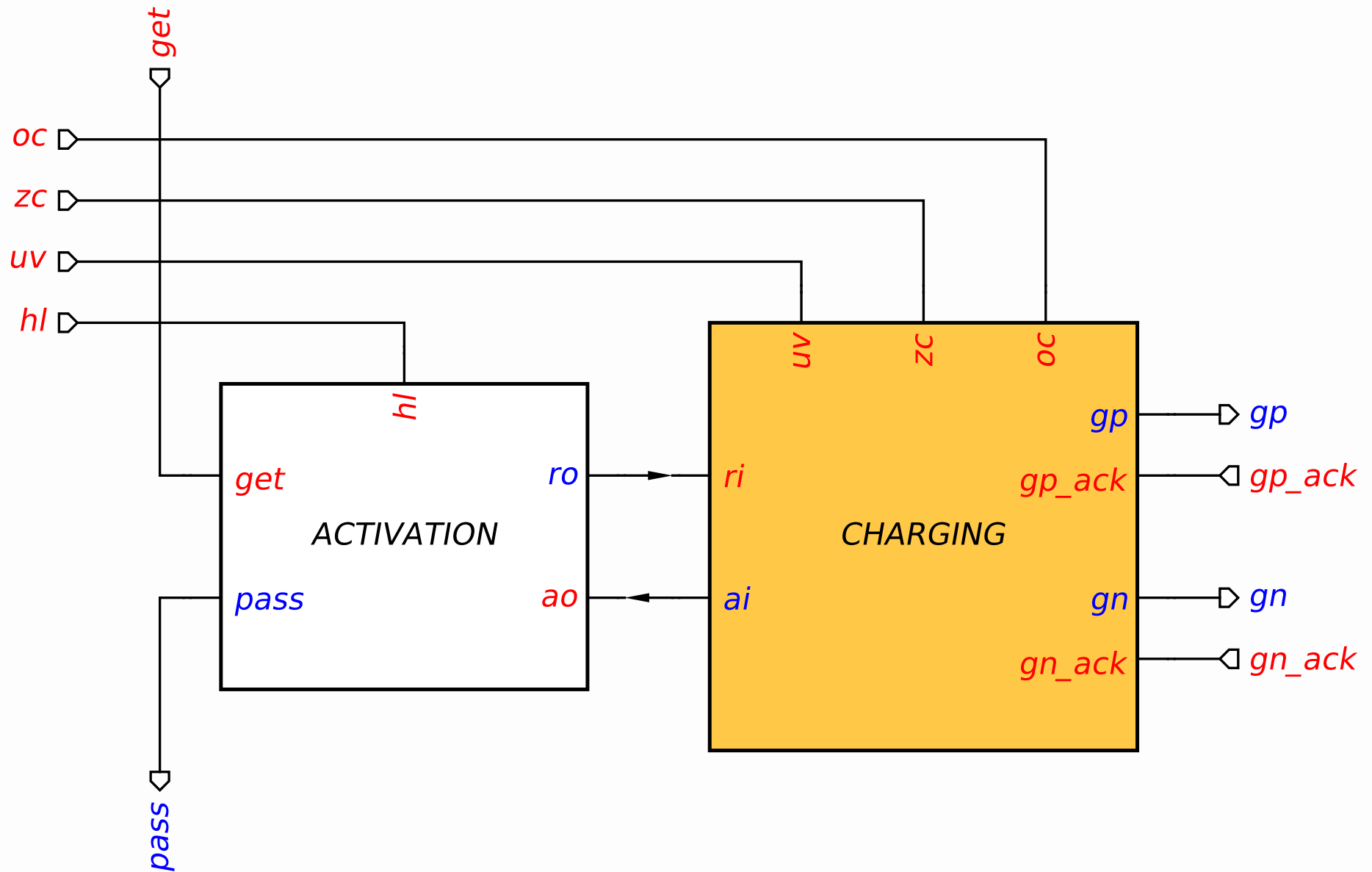
- ME-based solution



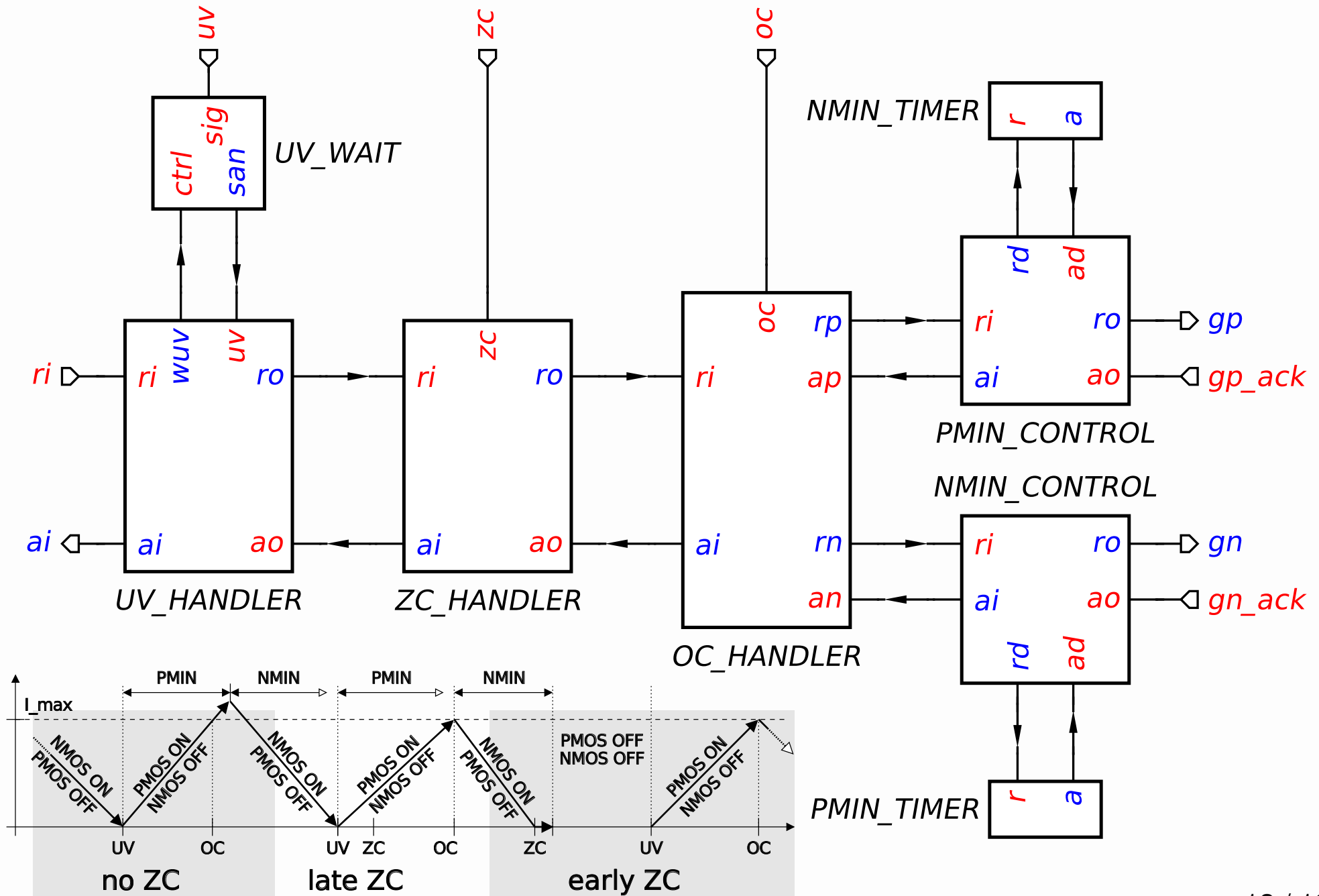
- Gate-level implementation



High-level architecture: Charging



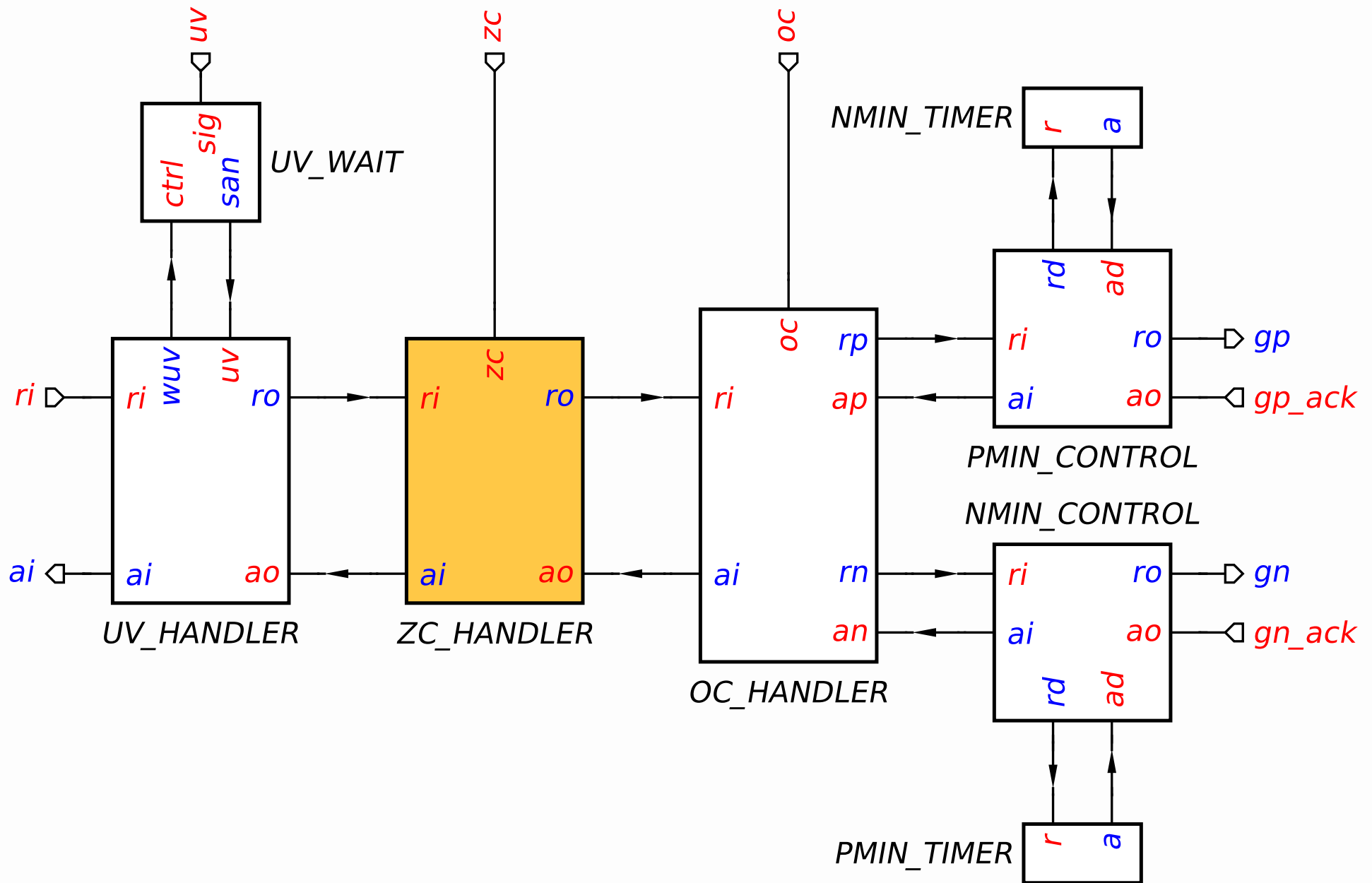
High-level architecture: Charging



Synthesis flow

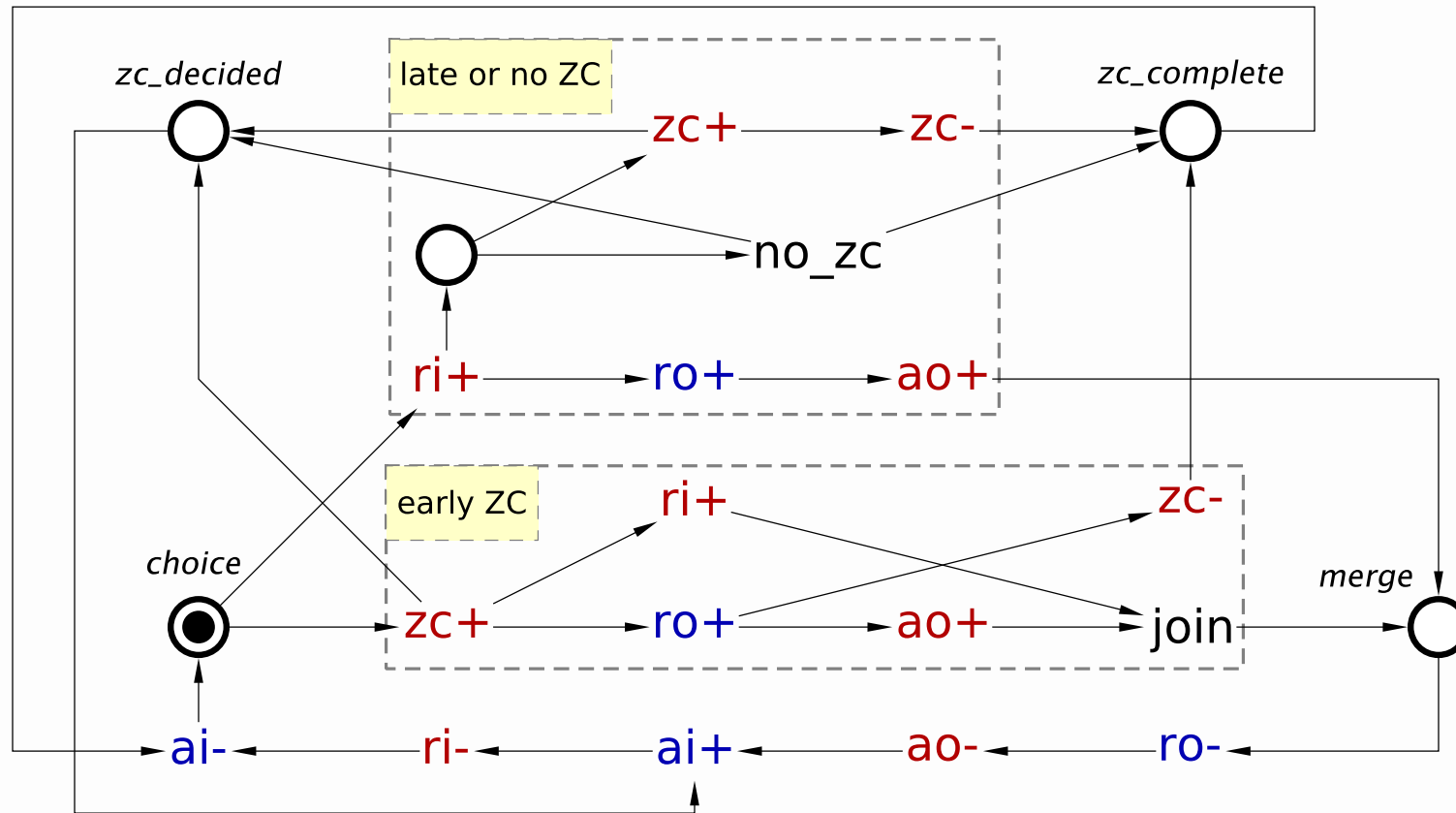
- Manual decomposition of the system into modules
 - To create formal specification from informal requirements (feedback loop with engineers)
 - To simplify specification and synthesis
 - Some modules are reusable
 - Some modules (WAIT and OPPORTUNISTIC MERGE) are potential standard components
- Each component is specified using STGs
- Automatic synthesis into speed-independent circuits

ZC_HANDLER module



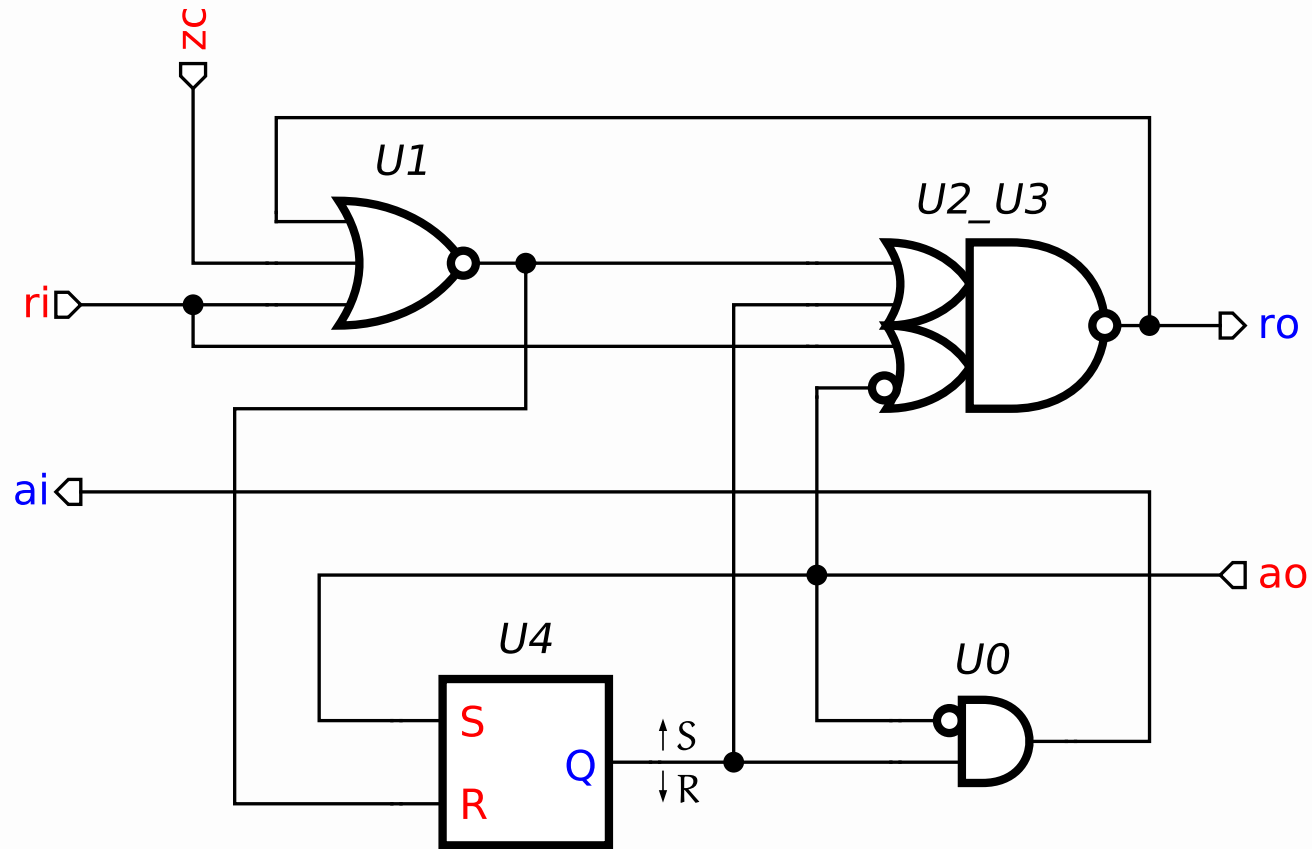
ZC_HANDLER module

- STG specification



ZC_HANDLER module

- Speed-independent implementation



Formal verification

- STG verification
 - All standard speed-independence properties
 - PMOS and NMOS are never ON simultaneously (no short circuit)
 - Some timers are used in a mutually exclusive way and can be shared
- Circuit verification
 - Conforms to the environment
 - Deadlock-free and hazard-free under the given environment

Tool support: WORKCRAFT

- Framework for *interpreted graph models* (STGs, circuits, FSMs, dataflow structures, etc.)
 - Interoperability between models
 - Elaborated GUI
- Includes many backend tools
 - PETRIFY – STG and circuit synthesis, BDD-based
 - PUNF – STG unfold
 - MPSAT – unfolding-based verification and synthesis
 - PCOMP – parallel composition of STGs

Tool support: WORKCRAFT

The screenshot displays the Workcraft software interface with several panels:

- *circuit-ZCH-map [circuit]:** A logic circuit diagram with inputs *ri*, *ai*, and *zc*, and outputs *ro* and *ao*. It includes a flip-flop and several logic gates.
- stg-ZCH [STG]:** A state transition graph with nodes and transitions labeled with signals like *ri+*, *ro+*, *ao+*, *ai+*, *ro-*, *ao-*, *ai-*, *ri-*, *zc+*, and *zc-*. Two regions are highlighted: "late or no ZC" and "early ZC".
- *circuit-ZCH-map 1 [STG]:** A detailed state transition graph showing the internal state of the circuit with nodes labeled *U1+*, *U1-*, *U4+*, and *U4-*.
- pcompressresult6863112684546701504 [STG]:** A complex state transition graph representing a compressed model.
- Property editor [model]:** A panel for editing model properties, currently showing "Environment... /stg-ZC...".
- Tool controls:** A panel with navigation and simulation icons.
- Editor tools:** A panel with editing tools like selection, zoom, and pan.
- Output:** A panel showing the circuit's configuration and simulation results.
- Message:** A dialog box with the text: "Under the given environment (stg-ZCH.work) the circuit is: * conformant, * deadlock-free, * hazard-free".
- workspace:** A panel showing the project structure, including files like *circuit-ZCH-map 1.work*, *circuit-ZCH-map.work **, *stg-ZCH.work*, and *pcompressresult68631126845*.

```
INORDER = ao ri zc ai ro csc0;  
OUTORDER = [ai] [ro] [csc0];  
[ai] = csc0 ao'; # gate and2_1:combinational  
[1] = ri' zc' ro'; # gate nor3:combinational  
#PRAGMA: zero delay  
[2] = ao'; # gate inv:combinational  
[ro] = [1]' csc0' + [2]' ri'; # gate oai22:combinational  
[csc0] = csc0 [1]' + ao; # gate sr_nor:async  
  
# Set/reset pins: reset(ro)  
Exporting model "Untitled" to file "/tmp/workcraft-circuit-ZCH-map-8245652389417126911/dev.g".
```

Conclusions

- Fully asynchronous design of multiphase buck controller
 - Quick response time: few gate delays, all mutexes are outside the critical path
 - Reliable: no synchronisation failures
- Design flow is automated to large extent
 - Automatic logic synthesis
 - Formal verification at the STG and circuit levels
- New standard components (WAIT and OPPORTUNISTIC MERGE)
- Future work
 - Measurements!
 - Development of WORKCRAFT to support hierarchical design
 - Co-simulation and co-verification of digital/analogue circuits
 - Better integration with the Synopsys and Cadence flows