Design Automation for Analog-Mixed Signal Circuits with Asynchronous Control

Vladimir Dubikhin(*), Victor Khomenko (*),
Andrey Mokhov(*), Chris Myers (**),
Danil Sokolov(*), Alex Yakovlev (*)

(*)Newcastle University, UK
(**)The University of Utah, USA
Contact: Alex.Yakovlev@ncl.ac.uk
async.org.uk; workcraft.org

Agenda

- Introduction:
  - Motivation, Challenges, Shortcomings of commercial flows
- Part 1. A4A: Asynchronous design for analogue electronics
  - Basics of asynchronous design
  - Design flow for A4A: formal specification, circuit synthesis, verification
  - Examples: multiphase buck, SRAM, ADC

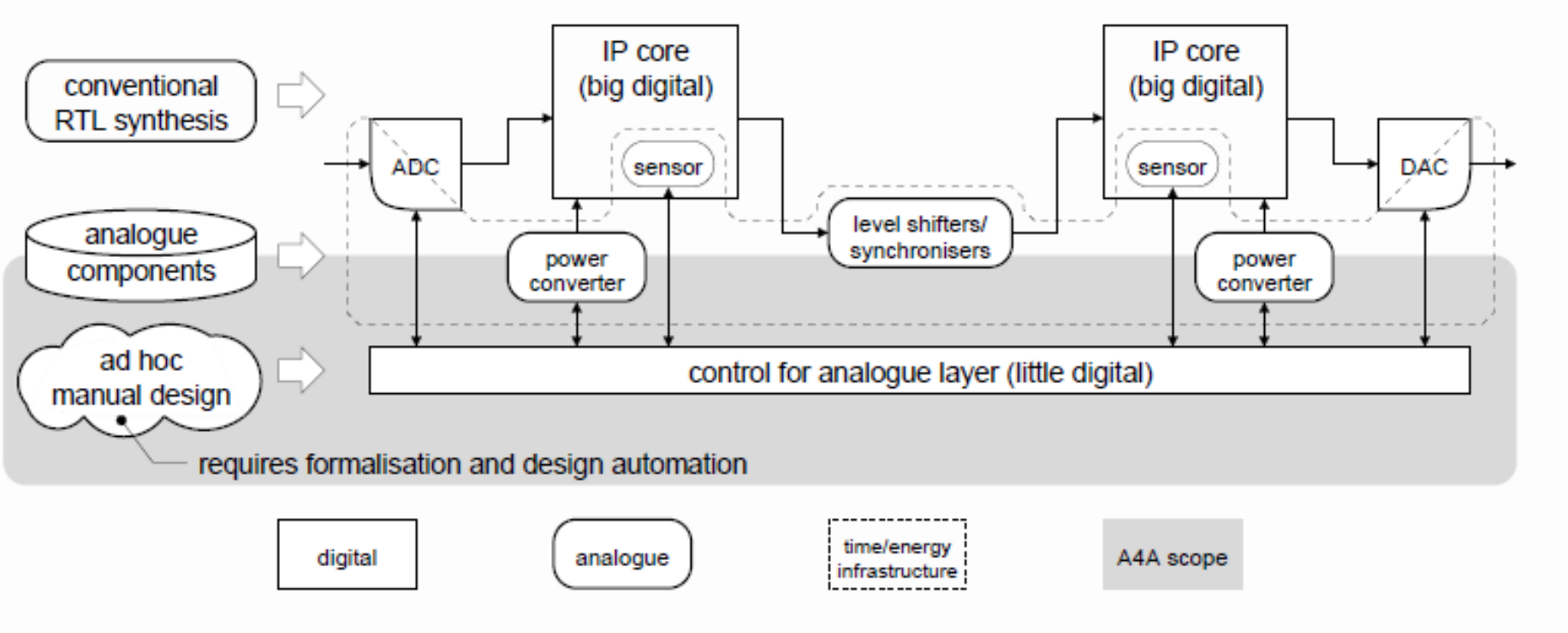
<Break>

- Part 2. AMS design with asynchronous control
  - Analogue verification with LEMA
  - Co-optimization flow: Workcraft and LEMA
  - Examples: C-element, Buck
- Part 3. Workcraft tools demo
- Discussion

Motivation

- **Analog and Mixed Signal (AMS) design becomes more complex:**
  - **More functionality**
  - **Move to deep submicron after all!**
- **According to Andrew Talbot from Intel, recently speaking at the AMS workshop at RAL, *"transistors are very fast switches, netlists are huge, parasitics are phenomenally difficult to estimate, passives don't follow Moore's law, reliability is a brand new landscape"***

Emergence of “little digital”



- **There is a strong drive for having more digital parts in AMS**
- **Analog and digital are often intertwined**
- **Asynchronous design appears good for little digital**

–> A4A project (EPSRC, Dialog supports)

Motivation: power electronics context

- **Efficient implementation of power converters is paramount**
  - **Extending battery life for mobile gadgets**
  - **Reducing energy bill for PCs and data centres (5% and 3% of global electricity production, respectively)**
- **Need for responsive and reliable control circuits – *little digital***
  - **Millions of control decisions per second for years**
  - **A wrong decision may permanently damage the circuit**

Motivation: EDA support is a challenge

- **Poor EDA support**
  - **Mostly supports flow from schematic capture; lacks flow from behavioural capture**
  - **Synthesis from behavioural (RTL) is optimized for data processing logic and supports only synchronous – *big digital***
  - ***Manual* and *ad hoc* solutions are prone to errors and hard to verify (weeks of simulations)**
- **Big challenge is EDA for asynchronous (hence our A4A project)**
- **What do the Industrial gurus say?**

Industry quotes

- "...analog has to budget five or six respins. Silicon has become the validation vehicle for analog, and that's a problem."

  Sandipan Bhanot, CEO of Knowlent

- "If the digital designers did verification the way analog designers do verification, no chip would ever tape out."

  Sandipan Bhanot, CEO of Knowlent

- "...problems are being solved because we have very good analog engineers. But in the future, if we want to improve time-to-market we will have to improve the tools."

  James Lin, VP at National Semiconductor

(Source "Why is analog so difficult?" – DACezine, January 2008)

Analog design in digital context is hard

- **If digital parts don't use clock, they are normally designed by hand and require massive simulations:**
  - **E.g. analogue designers cannot afford simulating power converters from start-up; Instead they force it into known state**
  - **More specifically: 50 us of Spectre simulation time takes approx. 10 hours using 8 CPU cores**
  - **Hence they can only verify cherry-picked corners of digital functionality**

(from Dialog Semiconductor, 2016)

Intel's advice on AMS Design

Intel's advice:

- Partition the Design to separate Analog and Digital
  - Give digital circuits to digital tools
  - Give analog circuits to analog tools
  - Do not pollute the hierarchy with logic gates or analog components!

(Source: Intel's talk about Holistic AMS design in Ultra-DSM at the May 2016 NMI event on AMS)

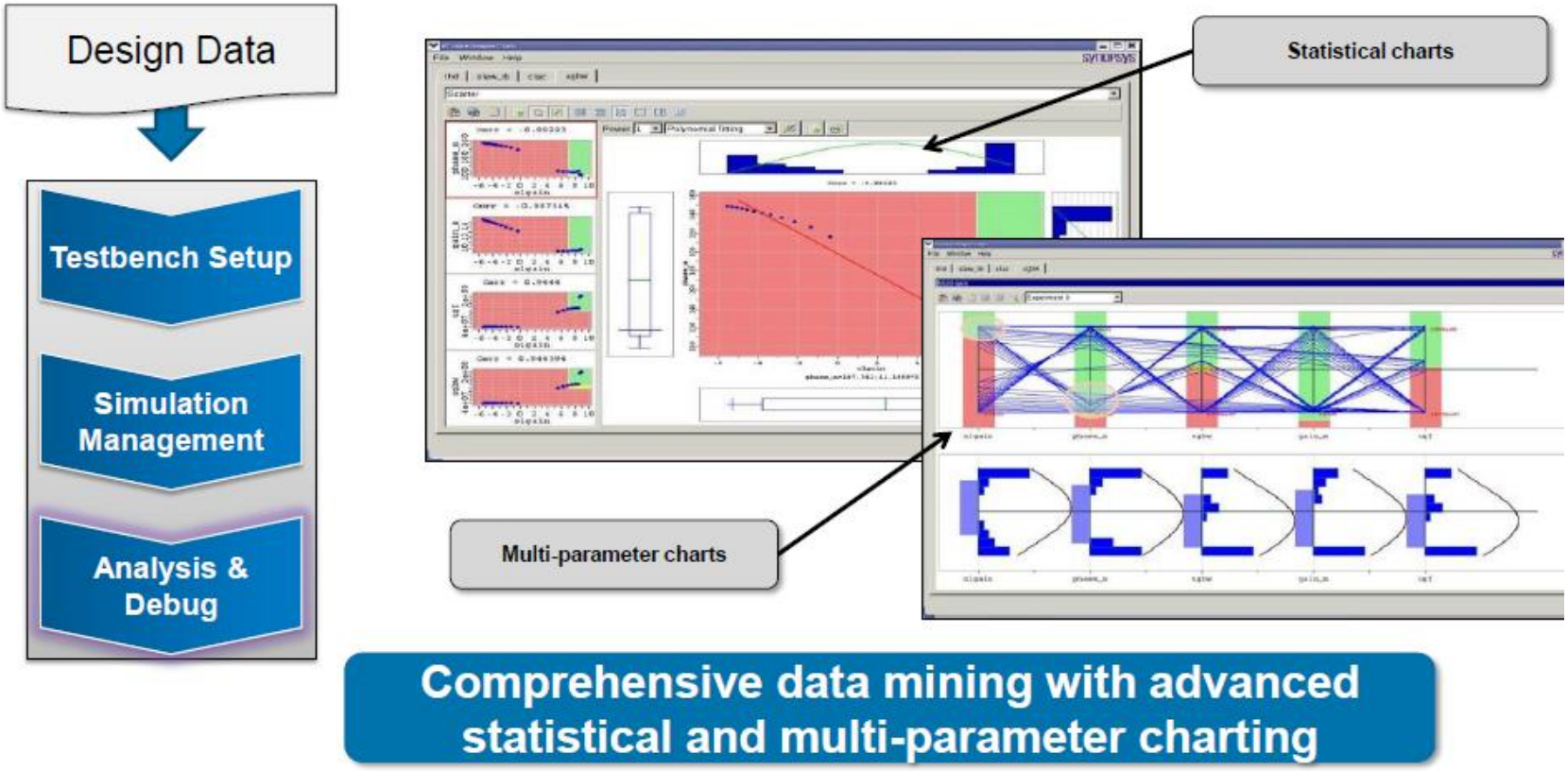
But, how?

We must use Behavioural capture and drive verification from behavioural domain!

View from Synopsys

Analysis and Debug

Data mining

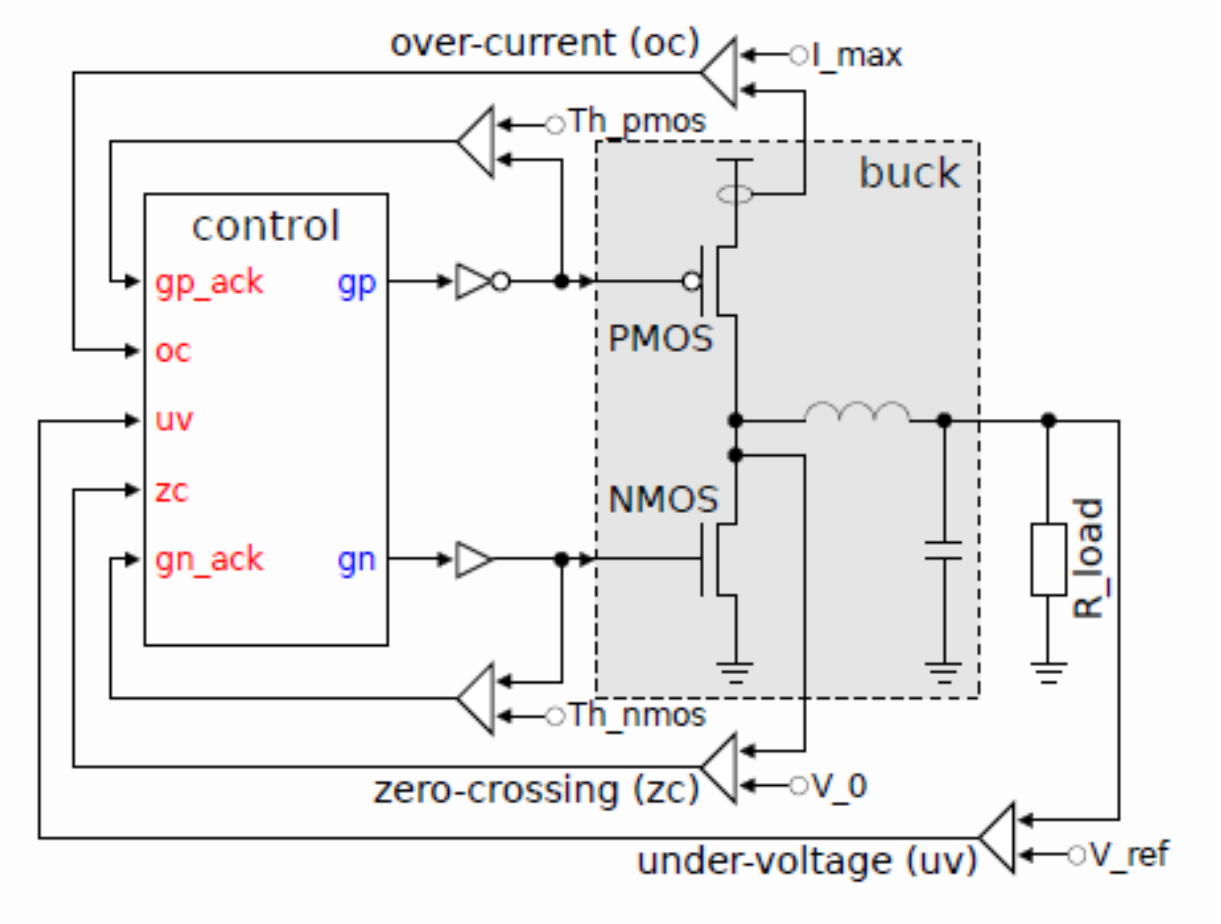


Source: Damian Roberts, AMS Workshop, RAL, April 2016

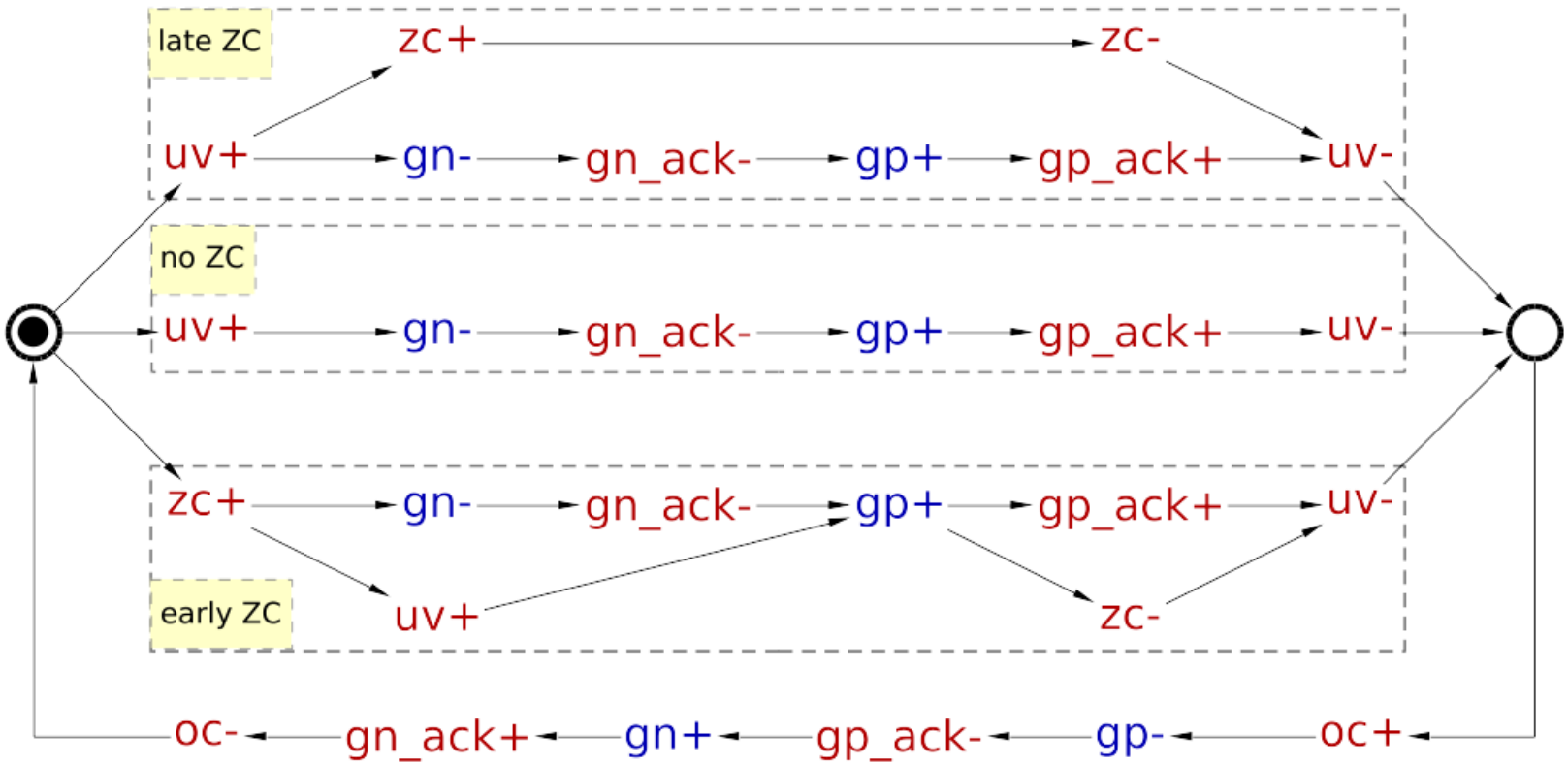
Towards Async Design for Analog

- Asynchronous design offers many advantages for AMS control
- Challenges:
  - It requires behavioural capture and synthesis but commercial EDA tools don't support it
  - Verification of asynchronous designs as part of AMS
  - How to provide non-invasiveness with existing design practices – we need to work with SVA and SPICE simulation traces

Buck example



STG Specification of buck controller



Synchronous design

- Two clocks: phase activation (~5MHz) and sampling (~100MHz)
  - ☺ 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

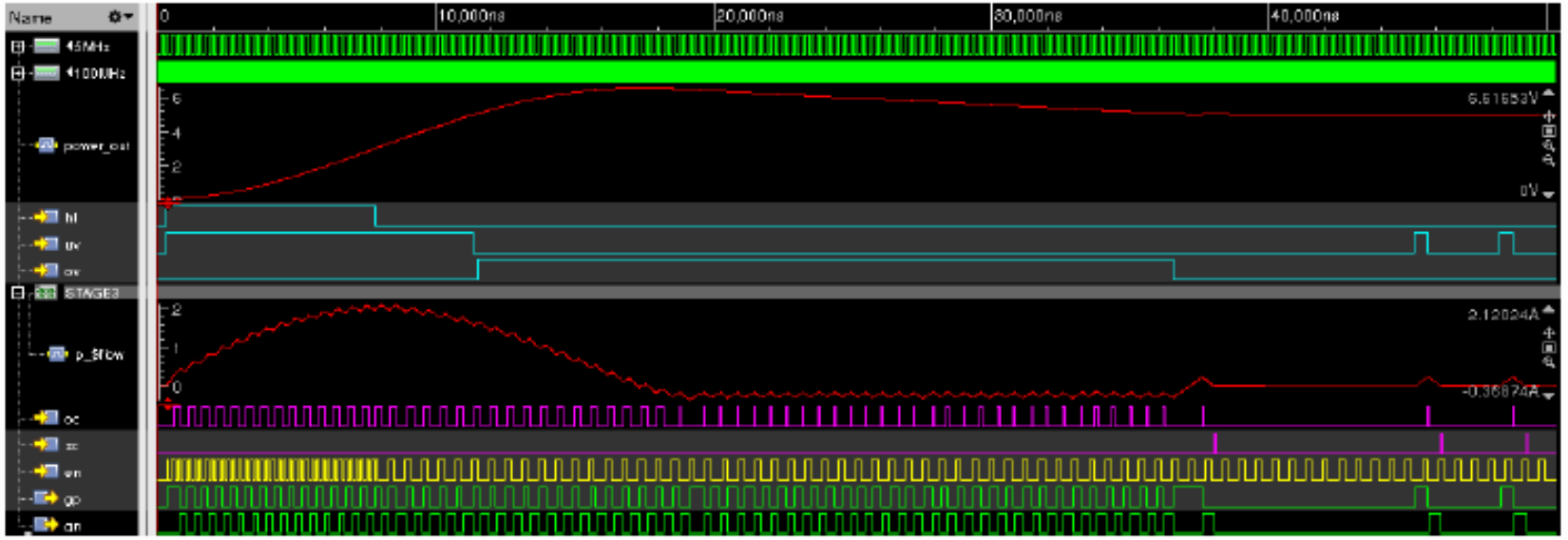
  ☺ Other well known advantages

  ☹ Insufficient methodology and tool support

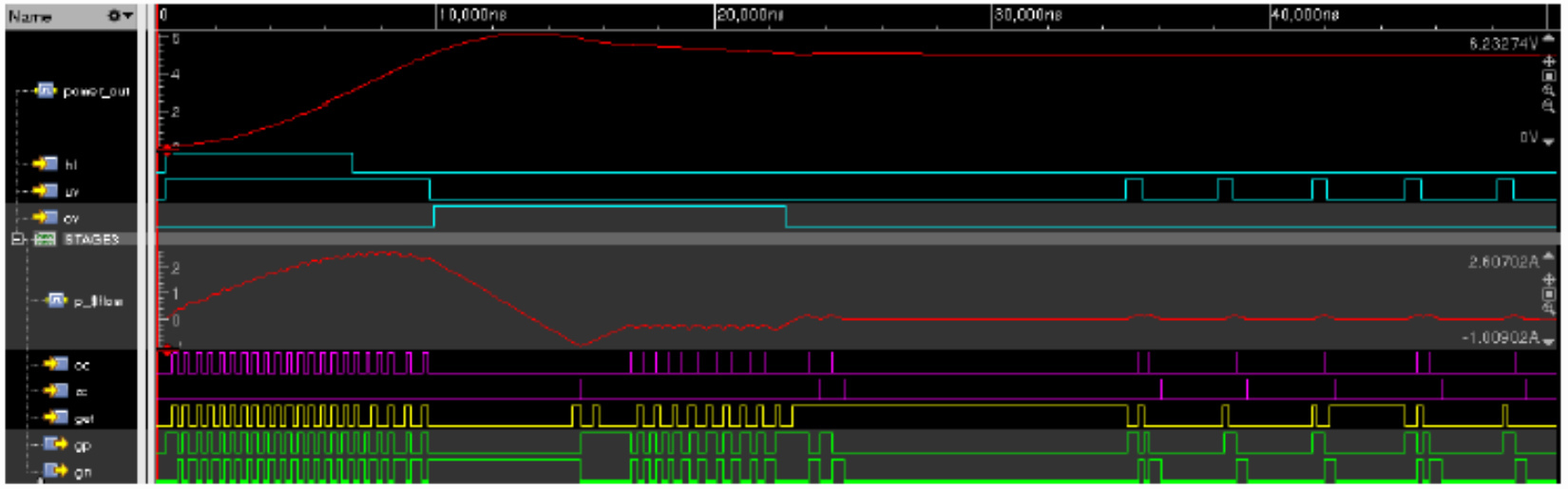
- Our goals
  - Formal specification of power control behaviour
  - Reuse of existing synthesis methods
  - Formal verification of the obtained circuits
  - Demonstrate new advantages for power regulation (power efficiency, smaller coils, ripple and transient response)

Simulation results

synchronous



asynchronous



Simulation results: Comparison

- Verilog-A model of the 3-phase buck
- Control implemented in TSMC 90nm
- AMS simulation in CADENCE NC-VERILOG
- Synchronous design
  - Phase activation clock – 5 MHz
  - Clocked FSM-based control – 100 MHz
  - Sampling and synchronisation
- Asynchronous design
  - Phase activation - token ring with 200 ns timer (= 5 MHz)
  - Event-driven control (input-output mode)
  - Waiting rather than sampling (A2A components)

Specifics of Async Design for Bucks

- Needs to be to a large extent monolithic
- Has inputs that need to be sanitised
- Can have lots of timing assumptions for bounded delay implementation where solving coding and TM problems can be an issue
- I/O response times (constrained or optimised) drive the design and sign-off
- Different types of (de)compositions needed rather than (or not just) handshake ones