Async-Analog: Happy Cross-talking?

Alex Yakovlev
School of Engineering
Newcastle University, UK
Contact: Alex.Yakovlev@ncl.ac.uk
async.org.uk; workcraft.org

ASYNC-FAC 2018, Vienna, 16 May 2018
Vienna is now async Capital

• A bit of history:
  – DDECS’10: “Async: Quo vadis?”, Eyjafjallajökull vulcano, getting stuck in Vienna
  – Async design course in 2011, Belvedere
  – Async 2013 – 4 papers from TU Vienna accepted

And finally, the whole of ASYNC is here!
Agenda

• Prologue:
  • Why, where and how A and A meet
• Analog methods for Async circuits
• A4A: Async design for Analog electronics
  • Design flow for A4A
  • Workcraft.org
• AxA: Mixed-signal design with Async control
  • Analog-Async Co-design flow
  • Workcraft and LEMA
• Epilogue:
  • Messages and Open challenges
Key message from our experience

• PRODUCTIVITY is a King

Although

• Robustness, Low Power, Performance ... maybe a Queen

First you show the Queen and then you push the King!
Prologue
Prologue

Async people: How and where do you face with analog?

Analog people: How and where do you face with digital (or even async)?
• Typically in continuous time and level
• Can be represented by EM forces/fields, electric charge, magnetic flux, voltage, current
• In various applications can represent mechanical, chemical, thermal etc. forms of information or energy
• Dynamics can be represented in time and frequency domains
• Typically separates signal flows between “data” and power supply but does not always need to
• Applications: sensing, measurement, signal processing, control, power
• Interfaces between digital and analog: analog elements inside digital components, ADC and DACs, digital control of analog
Analog elements

• Amplifiers:
  – Operational amplifiers
  – Low noise amplifiers
  – Sense amplifiers
• Current mirrors
• Bandgap circuits
• Delay elements
• Oscillators
• Transmission lines
Async (digital) behaviour

• Triggered by events (e.g. level-crossing)
• Modelled by
  – cause-effect relations
  – token flow
  – handshakes
  – data-flow
• Power-driven timing
• Applications: interfacing, control, pipeline
Question to think about:

Do we need to divide the world into analog and digital?

Async helps us to remove or at least lower the A-to-D and D-to-A walls
Question to think about:

Do we need to divide the world into analog and digital?

Async helps us to remove or at least lower the A-to-D and D-to-A walls

Let’s think about other dichotomies, maybe Electronic to Bio?

But better replace “Divide and Conquer” with “Unite and Survive”!
Analog for Asynchronous

Didn’t we do well with this!?
Analog circuit design and analysis has been helping asynchronous design for ages:

– Time comparison (arbiters and synchronizers)
– Metastability analysis
– Noise and Cross-talk analysis
– EMI analysis and optimisation
– Timing optimisation, e.g. Surfing
– ...

*Work by Molnar, Kinniment, Greenstreet, Ginosar and others*
Why and what is timing comparison?

This is probably the oldest story of analog serving for async (and digital as a whole)

Telling if some event happened before another event
Synchronizers and arbiters

- **Synchronizer**
  Decides which clock cycle to use for the input data

- **Asynchronous arbiter**
  Decides the order of inputs
Time Comparison Hardware

- Digital comparison hardware (which compares integers) is easy
  - Fast
  - Bounded time
- Analog comparison hardware (which compares reals like time) is hard
  - Normally fast, but takes longer as the difference becomes smaller
  - Can take forever
- Synchronization and arbitration involve comparison of time
Metastability is...

Not being able to decide...
Typical responses

- We assume all starting points are equally probable
- Most are a long way from the “balance point”
- A few are very close and take a long time to resolve
Event Histogram

- Propagation delay

Events

Log Probability of event depends on $\Delta$ time

The intercept is $\sim T_w$

The slope is $-1/\tau$

Propagation delay

Normal delay
Metastability in a Latch

Stable points

Metastable Point
Linear Model

- Simple linear model leads to two exponentials
- $\tau_a$ is convergent, $\tau_b$ is divergent

\[
\tau_1 = \frac{C_1 \cdot R_1}{A}, \quad \tau_2 = \frac{C_2 \cdot R_2}{A} \quad g_m = \frac{A}{R}
\]

\[0 = \tau_1 \cdot \frac{d^2 V_1}{dt^2} + \left( \frac{\tau_1 + \tau_2}{A} \right) \frac{dV_1}{dt} + \left( \frac{1}{A^2} - 1 \right) V_1\]

\[V_1 = K_a \cdot e^{\tau_a t} + K_b \cdot e^{\tau_b t}\]
How often does it fail?

- The output trajectory is an exponential that depends on the starting condition $K$, $K$ depends on $\Delta t_{in}$
- Suppose the clock frequency is $f_c$, the data rate $f_d$, and $K_a = 0$
- In $M$ seconds we have $M.f_c$ clocks.
- The probability of a data change within $\Delta t_{in}$ of any clock is $\Delta t_{in} . f_d$ so there will be one within $M$ seconds if
- The time taken to resolve this event is $t$ ($T_w$ is the metastability window)

$$V = K . e^{\frac{t}{\tau}}$$

$$\Delta t_{in} = \frac{1}{M . f_c . f_d}$$

$$\frac{V}{K} = \frac{T_w}{\Delta t_{in}} = e^{\frac{t}{\tau}}$$
Synchronizer

- \( t \) is time allowed for the Q to change between CLK a and CLK b
- \( \tau \) is the recovery time constant, usually the gain-bandwidth of the circuit
- \( T_w \) is the “metastability window” (aperture around clock edge in which the capture of data edge causes a delay that is greater than normal propagation delay of the FF)
- \( \tau \) and \( T_w \) depend on the circuit
- We assume that all values of \( \Delta t_{in} \) are equally probable

\[
MTBF = \frac{e^{t/\tau}}{T_w \cdot f_c \cdot f_d}
\]
Two-way arbiter (Mutual exclusion element)

Basic arbitration element: Mutex (due to Seitz, 1979)

This is a key building block for multi-way arbiters, time-to-digital converters, GALS controllers, analog to async interfaces ...
Metastability analysis

• Characterisation of $\Delta t_{in}$, $t$, $\tau$ and MTBF, in relation to the technology parameters and Vdd
• This requires extensive simulation effort
• Techniques such as bisection with numerical integration can be applied:
  – Timing interval around metastability point is minimized obtaining progressively smaller voltage window sizes and their corresponding (larger) settling times (Yang-Greenstreet)
  – Other analog methods for reducing simulation time are used, such as voltage transfer curves and current compensation (Beer-Ginosar)
  – Blendics has tools which use some of these methods
  – Time-dependent gain analysis in multi-stage REAL synchronizers (Reiher-Greenstreet-Jones – this ASYNC!)
What else affects $\tau$

- Vdd matters
- Feature size
- Process variability ($V_t$)
- Duty cycle of clock (for master-slave FF based)
- Multi-stage synchronizers
- Data and synchronizer FFs are totally different

Source: Blendics
Other areas for analog for async

- Logic cell design:
  - GasP
- Power supply:
  - IR degradation,
  - Drooping,
  - Subthreshold
- Interconnect:
  - Transmission line models,
  - Cross-talk and noise
- Delay analysis and design:
  - stray, inertial, pure delays

Applying analog knowledge has always helped in:
- Speeding circuits up,
- Reducing power,
- Improving reliability and robustness
What about the other direction: Async for Analog?
Motivation for Async for Analog

- Analog and Mixed Signal (AMS) design becomes more complex:
  - More functionality
  - Move to deep submicron after all!
- According to Andrew Talbot from Intel (2016)
  “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”
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 (not as fuzzy as genetic circuits!)
Efficient implementation of power converters is paramount:

- Extend battery life for mobile gadgets
- Reduce 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 (not as fuzzy as genetic circuits!)

Motivation: power electronics context

Attract with the Queen:
Robustness, Low latency ...
Emergence of little digital electronics

- Analog and digital electronics are becoming more intertwined
- Analog domain becomes complex and needs digital control
And we have already done well here, too!

Async A-to-D Converters
Async-assisted sensors

...
Async ADC

- Synchronous

- Asynchronous

---

ADC design
Asynchronous controller

- STG specification
- Speed-independent implementation
Reference-free voltage sensor

Energy harvesting source

Storage element

Control

Sampling circuit

Self-timed counter

Req → Ack

Data

V_{dd} → 8
Reference-free voltage sensing

- Voltage sensor requiring only timing reference

Apparatus and method for voltage sensing, Newcastle University, GB Patent Number 2479156, 30 March 2010.
Sensors using asynchronous logic

Cap-to-digital Conversion (sensing)

Y. Xu et al, ICECS’16
Can we go further with Async?

Where we can have a win-win situation!

To something bigger ... such as

Power electronics

And show impact outside the ‘usual’ digital scope’ ....
Example: Buck (DC-DC) converter control
Example: Switched Capacitor (DC-DC)
Converter control
Example: Buck converter

Building asynchronous circuits in Analog-Mixed Signal context requires extending traditional assumptions about speed-independence ...

Phase diagram specification:

Buck conditions:
- under-voltage (UV)
- over-current (OC)
- zero-crossing (ZC)

Operating modes:
- no zero-crossing
- late zero-crossing
- early zero-crossing
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?
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. analog 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)
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. an analog designer cannot simulate 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

Attack with the King: Productivity!

(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*

Comprehensive data mining with advanced statistical and multi-parameter charting

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

- late ZC
- no ZC
- early ZC
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)
**Multiphase Buck: Sync Control**

- Two clocks: phase activation (slow) and sampling (fast)
- Need for multiple synchronizers (grey boxes) - latency & metastability
- Conventional RTL design flow
Multi-phase Buck: Async Control

D. Sokolov, et al.  DATE 2017

- Token ring architecture, no need for phase activation clock
- No need for synchronisers - all signals are asynchronous
- A4A design flow
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)
Simulation results
## Reaction time

<table>
<thead>
<tr>
<th>Buck controller</th>
<th>HL (ns)</th>
<th>UV (ns)</th>
<th>OV (ns)</th>
<th>OC (ns)</th>
<th>ZC (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNC @ 100MHz</td>
<td>25.00</td>
<td>25.00</td>
<td>25.00</td>
<td>25.00</td>
<td>25.00</td>
</tr>
<tr>
<td>SYNC @ 333MHz</td>
<td>7.50</td>
<td>7.50</td>
<td>7.50</td>
<td>7.50</td>
<td>7.50</td>
</tr>
<tr>
<td>SYNC @ 666MHz</td>
<td>3.75</td>
<td>3.75</td>
<td>3.75</td>
<td>3.75</td>
<td>3.75</td>
</tr>
<tr>
<td>SYNC @ 1GHz</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
<td>2.50</td>
</tr>
<tr>
<td>ASYNC</td>
<td>1.87</td>
<td>1.02</td>
<td>1.18</td>
<td>0.75</td>
<td>0.31</td>
</tr>
<tr>
<td>Improvement over 333MHz</td>
<td>4x</td>
<td>7x</td>
<td>6x</td>
<td>10x</td>
<td>24x</td>
</tr>
</tbody>
</table>

Synchronous buck controllers exhibit latency of 2.5 clock cycles.
Peak current
Inductor losses
Design Results

Design flow is automated to large extent

- Library of A2A components
- Automatic logic synthesis
- Formal verification at the STG and circuit levels

Benefits of asynchronous multiphase buck controller

- Reliable, no synchronization failures
- Quick response time (few gate delays)
- Reaction time can be traded off for smaller coils
- Lower voltage ripple and peak current

Analog-2-Async (A2A): Wait, WaitX, Sample…
Logic synthesis and formal verification of asynchronous circuits.
What is Workcraft?

- Framework for interpreted graph models
  - Interoperability between different abstraction levels
  - Consistency for users; convenience for developers

- Elaborate graphical user interface
  - Visual editing, analysis, and simulation
  - Easy access to common operations
  - Possibility to script specialised actions

- Interface to back-end tools for synthesis and verification
  - Reuse of established theory and tools (PETRIFY, MPSAT, PUNF)
  - Command log for debugging and scripting
Why to use Wokcraft?

- **Availability**
  - Open-source front-end and plugins
  - Permissive freeware licenses for back-end tools
  - Frequent releases (4-6 per year)
  - Specialised tutorials and online training materials

- **Extendibility**
  - Plugins for new formalisms
  - Import, export and converter plugins
  - Interface to back-end tools

- **Usability**
  - Elaborated GUI developed with much user feedback

- **Portability**
  - Distributions for Windows, Linux, and OS X
Supported graph models

- Directed Graph
- Finite State Machine
- Petri Net
- Policy Net
- Digital Timing Diagram
- Finite State Transducer
- Signal Transition Graph
- Conditional Partial Order Graph
- Structured Occurrence Net
- Dataflow Structure
- Digital Circuit
- xMAS Circuit

Translation types:
- Lossless translation
- Lossy translation
- Synthesis
- Import: ASTG, Verilog
- Export: ASTG, Verilog, SVG/Dot/PDF/EPS
- Convert: synthesis or translation
- Verify: reachability analysis
Design flow: asynchronous circuits

1. Specification of desired circuit behaviour with an STG model
2. Verification of the STG model
   (a) Standard implementability properties: consistency, deadlock freeness, output persistency
   (b) Design-specific custom properties
3. Resolution of complete state coding (CSC) conflicts
4. Circuit synthesis in one of the supported design styles
5. Manual tweaking and optimisation of the circuit
6. Verification of circuit against the initial specification
   (a) Synthesis tools are complicated and may have bugs
   (b) Manual editing is error-prone
7. Exporting the circuit as a Verilog netlist for conventional EDA backend
What is hidden from the user

Verification that the circuit conforms to its specification

1. Circuit is converted to an equivalent STG – circuit STG
2. Internal signal transitions in the environment STG (contract between the circuit and its environment) are replaced by dummies
3. Circuit STG and environment STG are composed by PCOMP back-end
4. Conformation property is expressed in REACH language
5. Composed STG is unfolded by calling PUNF back-end
6. Unfolding prefix and REACH expression are passed to MPSAT back-end
7. Verification results are parsed by the front-end
8. Violation trace is projected to the circuit for simulation and debugging
Circuit Petri net as assembly language
Circuit Petri nets: data flow pipelines
Workcraft: basic screenshot
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
Future: Analog-Async Codesign!
LEMA (Myers et al, Utah)

Formal verification of LPN models of AMS circuits, generated from simulation traces.
LEMA Tool Flow

- LAMP
  - Property Compiler
- SPICE Simulation Data
  - Model Generator
- Thresholds
- SystemVerilog Model
  - SystemVerilog Translator
- Labeled Petri Net (LPN)
- BDD/SMT Model Checker
- Zone-Based Model Checker
- Octagon-based Model Checker

Output:
- Pass/Fail + Error Trace
Buck converter example
Model generation
Optimized STG

- Concurrency reduction

- Scenario elimination
Epilogue

Challenges
Messages to take away (for async people)

• **Little digital** circuits can be highly concurrent!
• Asynchronous circuits began their life (in the 50s) for ‘little digital’ and today is the right time for them
• Analog and mixed-signal is a good application – it combines:
  – Need for **low latency and high range of feedback types**
  – Analog designers are more inclined towards async than digital designers
• **Productivity in industry** is a good drive!
• **Interesting research problems** are there – tech mapping, holistic analog-mixed signal verification, behavioural mining, dealing with complexity
• In particular, extending the notion of speed-independence into the world of relative timing, circuits with time comparison (arbitration), with analog components
• Where else do we have little digital now? ... Plastic electronics?
Messages to take away (for async people)

• Little digital circuits can be highly concurrent!
• Asynchronous circuits began their life (in the 50s) for ‘little digital’ and today is the right time for them
• Analog and mixed-signal is a good application—
  – Need for low latency and high range of feedback types
  – Analog designers are more inclined towards async than digital designers
• Design tools are (slowly) coming up and industry is a good drive!
• Interesting research problems are there—
  – tech mapping, holistic analog-mixed signal verification, behavioural mining, dealing with complexity
• In particular, extending the notion of speed-independence into the world of relative timing, circuits with time comparison (arbitration), with analog components
• Where else do we have little digital now? … Plastic electronics?

Have a Tryst with an Analog Designer!
Open Challenges

- Complexity of analog behaviour – model generation/mining
- Control stability for AMS with async logic – formal proofs
- Electromagnetic effects – break the “circuit theory wall”

- How to discretize/quantize best?
  - E.g. Make use of the natural spatial (geometric) quantization
- How to decompose and distribute computations for AMS systems analysis?
  - E.g. use relaxation techniques
Inspiration comes from

Now, in Maxwell's theory there is the potential energy of the displacement produced in the dielectric parts by the electric force, and there is the kinetic and magnetic energy of the magnetic force in all parts of the field, including the conducting parts. They are supposed to be set up by the current in the wire. We reverse this; the current in the wire is set up by the energy transmitted through the medium around it. The ... energy of the electric machinery ..is transmitting energy from the battery to the wire. It is definite in amount, and the rate of transmission of energy (total) is also definite in amount.

Oliver Heaviside (1879)

Known for many revolutionary mathematical tools and post-Maxwell innovations

Oliver Heaviside (1850-1925)
Self-taught electrical engineer, mathematician, and physicist – who began as a telegraphy engineer in Newcastle
We employ a simple ring-oscillator to serve as a digital circuit load.

It is due to the fact that ring-oscillator can closely mimic the switching behaviour of many closed loop self-timed circuits.

What is Computational Load?
Circuit model

\[ C \]

\[ C_p \]

\[ 1 \rightarrow 0 \]

\[ 0 \rightarrow 1 \]

\[ C_t = \frac{N+1}{2} C_p \]

\[ C_t = \frac{N-1}{2} C_p \]

\[ C \]

\[ C_p \]
Circuit Model: switching process

$V_0$

$V_1 = K^1 V_0$

$V_2 = K^2 V_0$

$V_3 = K^3 V_0$

$V_n = K^n V_0$

$t_0$

$t_1$

$t_2$

$V$ drop over time

Charge equilibrium at:

$$V_1 = V_0 \frac{C}{C + C_l}$$

$$K = \frac{C}{C + C_l}$$
Solution for Super-threshold

A valid assumption: in super-threshold region we can assume that the propagation delay is inversely proportional to the voltage, so we have:

<table>
<thead>
<tr>
<th>Switching index</th>
<th>( V_N )</th>
<th>( t_s = \frac{A}{V} )</th>
<th>Physical time (( t ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( K^{-0} )</td>
<td>( \frac{A}{K^{-0}} )</td>
<td>( \frac{A}{K^{-0}} )</td>
</tr>
<tr>
<td>1</td>
<td>( K^{1} )</td>
<td>( \frac{A}{K^{1}} )</td>
<td>( \frac{A}{K^{-0}} + \frac{A}{K^{1}} )</td>
</tr>
<tr>
<td>2</td>
<td>( K^{2} )</td>
<td>( \frac{A}{K^{2}} )</td>
<td>( \frac{A}{K^{-0}} + \frac{A}{K^{1}} + \frac{A}{K^{2}} )</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>( K^{n} )</td>
<td>( \frac{A}{K^{-n}} )</td>
<td>( \sum_{i=0}^{n} \frac{A}{K^{-i}} )</td>
</tr>
</tbody>
</table>
Solution for Super-threshold

\[ V_N = \frac{AK}{t(1-K) + AK} \]

Hyperbolic function of time!

For super capacitor \( K \) is 1

increasing \( K \)
(Self-)oscillating circuits are voltage-controlled impedances!

- We are talking about DC, i.e. about average value of $V_f$ in steady state
- In practice due to short-circuit currents $V_f$ is close to $0.5V_{in}$ and does not depend on the number of inverters
Muller C-element:  \( y = x_1 x_2 + (x_1 + x_2)y \)

- After some transient the system will reach steady-state defined by

\[
R_S = \frac{1}{f_s C_p}; \quad R_O = \frac{1}{f_o C_p}; \quad \frac{V_f}{V_{in} - V_f} = \frac{R_O}{R_S} = \frac{f_s}{f_o}
\]
Thanks

• Newcastle Asynchronous Group:
  Victor Khomenko
  Danil Sokolov
  Andrey Mokhov
  Reza Ramezani
  Vladimir Dubikhin

• Our Collaborators:
  Jordi Cortadella (UPC Barcelona)
  Chris Myers (University of Utah)
  David Lloyd (Dialog Semiconductors)
  Alex Kushnerov (BGU, Israel)

• EPSRC and Dialog (funding support)