ISSN 2186-7437

NII Shonan Meeting Report

No. 133

Asynchronous Circuit Design and its Applications: Past, Present and Future

Tomohiro Yoneda
Peter A. Beerel
Alex Yakovlev
Masashi Imai

May 15–19, 2019

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Asynchronous Circuit Design and its Applications: Past, Present and Future

Organizers:
Tomohiro Yoneda (National Institute of Informatics, Japan)
Peter A. Beerel (University of Southern California, USA)
Alex Yakovlev (Newcastle University, United Kingdom)
Masashi Imai (Hirosaki University, Japan)

May 15–19, 2019

1 Introduction

1.1 What is this workshop about?

Asynchronous design is the study of circuits without a global clock. It is in contrast to synchronous design that relies on a global clock distributed to all sequential elements whose periodic nature provides a simplified discretized time model of system behavior. This model assumes that the clock is distributed across the chip and arrives at all the sequential elements nearly simultaneously, providing a unified time when all state is updated.

The problem of synchronous design is that the simplified model belies the underlying physics of modern and future technologies for which 1) increasing wire delay and increasing variations make simultaneous arrival times increasingly difficult to achieve 2) the economics warrant extensive module re-use which have different and sometimes inconsistent clocking constraints and 3) the interface to the analog world do not adhere to such a simplified model.

Asynchronous design has potential to solve this problem of synchronous design, because it does not rely on a global clock. This workshop is about such asynchronous design.

1.2 Why is this workshop held?

Asynchronous researchers have produced a range of techniques, that span support for point-to-point communication between different synchronous islands which are asynchronous to each, network-on-chips that facilitate communication among a collection of synchronous islands, and a variety of techniques to create asynchronous islands, either targeting high-performance, low-power, low-energy, or other application-specific requirements (e.g., the creation of low electromagnetic interference, resilience to soft-errors, robustness to extreme process variations, support of wide operating ranges, and support for next-generation non-CMOS technologies).

For each of these domains, active research has been performed across both industry and academia, spanning the Americas, Europe, and Asia. Numerous

startup companies have sought to disrupt various markets using asynchronous design, including general-purpose processors, networking chips, chips with high security and assurances, and vision processing and machine learning for the internet-of-things markets. Moreover, the problems associated with crossing clock domains and efficiently interfacing with the analog world is an evolving challenge as community technology evolves and new objective functions become important.

Because the community of asynchronous researchers active in this area spans the globe it is important to have regular activities for them to share experiences and shape the future. The IEEE Symposium on Asynchronous Circuits and Systems (ASYNC) is the leading conference in this area and is planned to occur in Japan in May, 2019. However, the symposium does not provide the opportunity for researchers to present position papers, discuss and debate the various benefits of different research directions, as well as plan collaborations where they make sense.

We believe that this type of interaction can be a catalyst and guide for the future of this area and that a Shonan meeting planned just after ASYNC can achieve these goals.

1.3 What has this workshop tried to solve?

The theme of this workshop is to discuss the past, present, and future of asynchronous circuits.

The past we think is an important element of this meeting, as many start-up lessons in this community have failed and the reasons why are not often discussed. Yet, we as a community can learn from these lessons, as the next generation of asynchronous entrepreneurs actively seek to change the world. Moreover, the past is fundamental to re-search as many older ideas become more practical as technology evolves. While past research is embodied in papers for eternity, it is not a replacement of face-to-face discussions among the old and new generation of asynchronous academics.

At present, there are at least four startups actively working on commercializing asynchronous designs and significant activity in multiple large VLSI companies. Understanding the challenges they are facing and guiding them with the collective knowledge of the community will enhance the chances of their success; a success which will help the entire community. Nevertheless, one common concern that all researchers in this community may have is that the asynchronous design style is not yet popular in practical VLSI design. Actually, it is fair to say that we all striving towards the same goal making asynchronous design a mainstream design style. Thus, as a main and specific theme of this meeting, we have discussed how best to change the world using asynchronous design technologies in the near future.

The future indeed is providing a plethora of opportunities as we are facing the slowing down of Moore's law and the gaining of interest in alternative computing technologies and frameworks, including new non-volatile technologies, three-dimensional VLSI integration, superconductive technologies as well as an explosion of interest in neuromorphic and machine learning styles of computation. All of them have potential to be strongly enhanced under the asynchronous design style, but achieving high impact may require greater forms of collaboration than we have had in the past. We thus believe that it is very valuable

for researchers who specialize in these different technologies to have in-person discussions on the common goal to change the world using asynchronous design and the best strategies to achieve this goal.

In order to achieve these goals, this workshop has provided a wide-variety of researchers with an opportunity to present their past experiences, present efforts, and future vision for asynchronous designs.

2 Meeting Schedule

Check-in Day: May 15 (Wed)

- 19:30-21:00 Welcome Reception

Day1: May 16 (Thu)

- 09:00-10:00 **Opening, Introduction by everybody** (1 min. per person)
- 10:00-11:00 **6 Talks** (10min. per person)
  Ivan, Marly, Yong, Peter, Hiroshi, Shogo
- 11:00-11:15 (Break)
- 11:15-12:15 **6 Talks** (10min. per person)
  Jordi, Masashi, Ney, Scott, Andreas, Montek
- 12:15-14:00 (Lunch)
- 14:00-14:20 Photo Session
- 14:20-15:20 **6 Talks** (10min. per person)
  Mika, Jens, Snorre, Jia, Huimei, Rajit
- 15:30-15:45 (Break)
- 15:45-16:45 **6 Talks** (10min. per person)
  Andrew, Alex, Warren, Milos, Matthias, Hong
- 17:00-17:15 (Break)
- 17:15-18:25 **7 Talks** (10min. per person)
  Ken, Tomohiro, Daniel, William, Mark, Takahiro, Naoya
- 18:45-19:45 (Dinner)
- 19:45- **Discussions for tomorrow's panels with drinks**
  Peter (Chair)

Day2: May 17 (Fri)

- 09:00-10:10 **Panel 1: Open source**
  Chair: Alex
  Panelists: Andrew, Rajit
- 10:10-10:15 (Break)
- 10:15-11:10 **Panel 2: Community involvement**
  Chair: Montek
  Panelists: All
- 11:10-11:20 (Break)
- 11:20-12:00 **Panel 3: Next day planning**
  Chair: Peter
- 12:00-13:30 (Lunch)
- 13:30-20:45 **Excursion**

Day3: May 18 (Sat)

- 09:00-11:00 **Panel 4: Education**
  Chair: Mika
  Panelists: Peter, Rajit, Montek, Jens
- 11:00-11:15 (Break)
- 11:15-12:10 **Panel 5: IP & library developer**
  Chair: Scott
  Panelists: William, Jia, Ney, Jordi
- 12:10-13:30 (Lunch)
- 13:30-14:15 **Panel 5: IP & library developer** (continued)
  Chair: Scott
  Panelists: William, Jia, Ney, Jordi
- 14:15-15:15 **Panel 6: User application**
  Chair: Ken
  Panelists: Milos, Hong
- 15:15-15:45 (Break)
- 15:45-16:30 **Panel 7: Next day planning**
  Chair: Peter
- 18:00- (Dinner and Discussion in Lounge)

Day4: May 19 (Sun)

- 09:00-11:00 **Panel 8: Tangible Next Steps**
  Chair: Peter
- 11:00-11:10 (Break)
- 11:10-11:30 **Wrap-up and closing**
- 11:30-13:00 (Lunch and Departure)

3 Overview of Talks

There is Enough for All

Ivan Sutherland, Marly Roncken, Yong Hei
Asynchronous Research Center, Portland State University, USA

We separate self-timed systems carefully into two parts:
1) parts that act to compute and control the flow of execution, and
2) parts that store and communicate.
We use the name "Joints" for the former and "Links" for the latter.

Data enter the input end of a communication Link and arrive at its output end. Storing data and state in Links allows Joints to be pure combinational circuits with the possible exception of arbiters.

Joints take data from FULL input Links, do their computation, and deliver results to EMPTY output Links. Joints choose which FULL Links to use as inputs and which EMPTY Links to use for outputs. This model remains valid for pure flow control with zero-width data.

Links and Joints communicate through shared variables rather than message passing. A Joint can access and "re-use" information carried over a FULL input Link multiple times, until it drains the Link. A FULL Link, when drained,

changes its state to (a) not-FULL and (b) EMPTY depending on which Link end you see. Similarly, a Joint can "re-use" an EMPTY output Link multiple times until it fills it.

Each action can be described as a guarded command. Concurrently with other Links and Joints, each Joint executes its own actions sequentially, as does each Link. Each Joint action and any resulting Link state changes visible to the Joint form one atomic action. This atomicity simplifies design, verification, and test.

A circuit we call "MrGO" in each Joint action provides the same flexibility that software debuggers get from breakpoints. MrGO can safely stop any Joint action just as a breakpoint can stop software. After stopping the system we examine its state to find flaws and bottlenecks and to characterize performance.

The Link and Joint model can serve as a universal asynchronous Register Transfer Level (RTL) model bridging high-level and low-level design.

Notes:

- defined CONFOUND concept and spoke about two business opportunities; "UNITED transport and storage", "DATA ACTION Inc."
- JOINT and LINK action with GO control – will be new RTL
- Guard and Command
- GO provide run-time breakpoints
- Algorithm to LINK-JOINT RTL translation
- RTL then is translated to circuits with relative timing, timing and functional verification

Useful Variations of Bundled-Data Design

Peter A. Beerel
University of Southern California, USA

Bundled-data design (BD) is one of the most well-studied sub-domains of asynchronous design and yet new variants pose unexplored advantages and interesting challenges. Bundled data design naturally admits to fine-grain voltage scaling but its benefits from a point of side-channel security have only begun to be looked at. Latch-based bundled design is tolerant to hold times but quantifying this benefit seems to be an open question. Minimizing the number of latches required and optimizing the clock gating for bundled-data design has only begun to be looked at. One variant of bundled-data design, timing-resilient BD design, or Blade, has been shown to lead to efficient near-threshold computing, but efficient place-and-route of Blade designs with its multiple clock trees is still largely unexplored. Another variant, radiationed hardened BD design extends the template to be resilience to soft-errors caused by radiation strikes, but detailed controller and circuit optimization remains incomplete. Moreover, while mapping RTL to bundled-data design is somewhat well studied, mapping CSP to bundled-data design offers new optimization possibilities that are also unexplored.

Notes:

- Open source tool Edge

- BD Timing Resilient Design
- BD SEU resilient design
- SERAD controller design – BM tools

A Design Support Tool Set for Bundled-data Implementation

Hiroshi Saito
University of Aizu, Japan

Although asynchronous circuits have a great advantage such as low power consumption, the use of asynchronous circuits in real designs is very limited. One of the reason is the difficulty of asynchronous circuit designs. To solve this problem, many researches focus on asynchronous circuit designs based on the design flow of synchronous circuits with commercial EDA tools. In our work, we propose a design support toolset for bundled-data implementation in which automates constraint generation, timing verification, and delay adjustment. The proposed design support toolset supports both application specific integrated circuit designs and field programmable gate array designs with commercial EDA tools. In this talk, we are going to present the overview of the proposed toolset, the result of a preliminary experiment, and the future direction of the proposed toolset.

Notes:

- Japanese industry needs low power but engineers no knowledge or skills
- C-to-RTL and FPGA
- Use conventional tools as much as possible

Conversion from Synchronous RTL Models to Asynchronous Ones

Shogo Semba
University of Aizu, Japan

To design asynchronous circuits easily, conversion methods from synchronous circuits to asynchronous circuits have been proposed. These conversion methods converts synchronous gate-level netlists to asynchronous ones (i.e., GL conversion). On the other hand, the target synchronous netlist may include a control circuit based on a finite state machine. It is particularly possible when synchronous circuits are obtained by high-level synthesis tools. In such a case, the current conversion methods may not work well because GL conversion targets simple pipeline structure. Moreover, GL conversion may be not suitable for the design of asynchronous circuits on commercial Field Programmable Gate Arrays (FPGAs) because the standard design entry of commercial FPGAs is Register Transfer Level (RTL) model. From this background, in our work, we research a conversion method from synchronous RTL models to asynchronous ones. In this talk, we introduce our method, developed tool, and current status.

Notes:

- We must start with RTL models, not with separate FSM for data-flow
- Aim: Sync RTL → Async BD via Model-XML files

High-Level Synthesis and Dataflow Circuits

Jordi Cortadella
La Universidad Peruana de Ciencias Aplicadas, Spain

FPGA-based hardware accelerators have emerged as an alternative between CPU-based software and ASICs for boosting the performance of computing-intensive functions. In this context, design automation is essential for a productive use of FPGAs and, for this reason, high-level synthesis has been gaining momentum in the last decade.

Dataflow computing was proposed in the 70's at MIT as a paradigm to specify concurrent programs with dataflow languages based on data-driven execution models that are inherently asynchronous. Unfortunately, asynchronous design has been often considered as a risky and error-prone option for engineers educated to design synchronous circuits and the prohibitive manufacturing cost of ASICs has become a showstopper for disruptive technologies.

Recently, elastic circuits have shown to provide promising results in FPGA-based systems. This opens an avenue of possibilities for exploiting dataflow computing and asynchrony in a environment in which both synchronous and asynchronous implementations can be experimented with low risk. The advances in high-level synthesis can foster the introduction of asynchronous techniques in hardware acceleration.

Notes:

- Different ways to implement algorithms
- HLS goes mostly into the FPGA
- Timing models – PLL and VCO
- Opportunity – HW acceleration

QDI and SDI Design in Subthreshold Region

Masashi Imai
Hirosaki University, Japan

As VLSI fabrication technology shrinks and the amount of circuits in a chip increases, power saving circuits are required. In addition, highly reliable circuits against delay variations under ultra-low voltage (subthreshold region) environment are also required. Here, we have research questions; which is better, a QDI (Quasi-Delay-Insensitive) model or SDI (Scalable-Delay-Insensitive) model implementation, and a bundled or encoded implementation? In this talk, I will explain the definition of the SDI model and its implementation rule again. Second, the pros and cons of QDI-model-based circuits and SDI-based circuits are explained. Then, the behavior of clock signals in the subthreshold region will be shown and the frequency of the clock signal in any part in a chip is the same as that of the source clock signal. Finally, I will show some preliminary evaluation results using 28nm process technology. We designed N-bits 3 stage pipeline circuits with ripple carry adders based on synchronous and asynchronous styles. They are evaluated using 28nm process technology and the Synopsys HSPICE analog simulator. As the result, it is concluded that the asynchronous dual-rail circuits have great potential to operate in the environment where the supply voltage is significantly low and the temperature is high.

Notes:

- SDI – scalable delay insensitive model (coming from 1997 – Takashi Nanya)
- New delay model – accounts for delay-variation
- Diagrams in Vdd vs Temperature operation for sync and async

QDI Circuits: Efficient Templates and Design Techniques

Ney Laert Vilar Calazans
Pontifical Catholic University of Rio Grande do Sul, Brazil

As I entered the field of asynchronous circuits in 2006, I imagined it would be easy to collect example circuits, tools and libraries. It was not. Apart from some tools like Petrify, 3D and Minimalist and some example circuits packed with them, there was nothing there. To design and fabricate asynchronous circuits we are thus used to build everything from scratch, and this could be made way better, helping others to learn use and join us in doing asynchronous design.

For some time, we have learned and taught how asynchronous circuits work, building up libraries based on a design flow called ASCEnD (Moreira'10, SOCC'11), analyzing the structure of fundamental blocks like C-elements, NCL gates and MTNCL gates (ICCD'13, SBCCI'17) and proposing new ways to build cells (ASYNC'14-1, ICECS'11) and solve specific design problems. From a set of design blocks, we now explore logic templates and their utility to effectively build asynchronous circuits. Examples are DIMS/WCHB, NCL, NCL+ (SBCCI'12), SDDS-NCL (ASYNC'14-2), VELO and SDDS-Velo (GUAZZELLI'17).

The above bottom-up strategy can be coupled with design capture techniques, available and/or under proposition (ICECS'17), to provide a full set of aids for asynchronous design, especially for QDI design. QDI design for near and/or sub-threshold operation is a major goal of my current research. The SDDS-VELO template (GUAZZELLI'18) seems to be a good starting point. Derived from MTNCL/SCL design, it has low area overhead (compared to NCL) and presents opportunities for robust operation under very low supply voltages. IoT edge nodes are a target application field.

Notes:

- In 2006 there was nothing in terms of tools for async design and fab
- Developing cells – started with C-elements
- Cell Libraries – cooperation with Silvaco and Nangate
- Used in Universities already
- Async design templates
- SDDS-VELO template patented

Verification of QDI Circuits

Scott C. Smith
North Dakota State University, USA

Validation is a critical part of any commercial design cycle; and formal verification establishes correctness using mathematical proofs, which is extremely

useful for ensuring design correctness and finding corner-case errors that often escape traditional testing. One of the more popular formal verification approaches that has been found to be extremely scalable and useful in digital design is equivalence checking, which can check, with a high degree of automation and efficiency, that the golden model (i.e., the design that has been extensively validated) and its derivate are functionally equivalent. There are a number of commercial equivalence checkers for synchronous circuits, including IBM Sixth Sense, Jasper Gold Sequential Equivalence Checker, Calypto SLEC, Mishchenko EBCCS13, and Cadence Encounter Conformal Equivalence Checker; however, there are currently no commercial equivalence checkers for QDI circuits. This talk will discuss an equivalence checking methodology for QDI circuits, including NCL, MTNCL, and PCHB, which guarantees both safety (i.e., functional correctness) and liveness (i.e., deadlock free). The methodology automatically transforms the QDI circuit into an equivalent synchronous circuit, which is then compared against the specification synchronous circuit, using WEB refinement as the notion of correctness. The converted synchronous circuit, specification synchronous circuit, and the WEB refinement property are automatically encoded in the Satisfiability Modulo Theory Library (SMT-Lib) language, and the resulting equivalence property checked using an SMT solver. Additional checks (e.g., no illegal states, handshaking correctness, input-completeness, observability) must also be performed to ensure that the QDI circuit is live. Thus, the overall verification method has three high-level steps: (1) Conversion from QDI to synchronous; (2) Verification of converted synchronous against specification synchronous; and (3) Additional checks on original QDI circuit to ensure liveness. The methodology can also be used to check the equivalence of two QDI circuits by applying the conversion technique to both QDI circuits to obtain two corresponding synchronous circuits, functionally verifying these two synchronous circuits against each other, and performing the additional liveness checks on both QDI circuits.

Notes:

- Verification is a must for all companies
- Previous work: Bounded delay circuits, Deadlock detection, modeling QDI as transition system: NCL using web refinement with stuttering and PCHB using model checking
- NCL verification Methodology; Functional – replace each NCL gate to normal boolean, SMT-Lib; Invariant check check rail0 is invert (rail1); Liveness: handshaking and input-completeness and observability
- Applied to MT-NCL and PCHB

Fault Modeling and Fault Masking in Asynchronous Circuits

Andreas Steininger
Vienna University of Technology, Austria

We start from the observation that asynchronous circuits, in particular QDI circuits, have a very robust timing behavior, which is an attractive property for implementing resilient circuits. In the value domain, their resilience can be assessed through their immanent ability to mask faults. In case of transient

faults, we are interested to compare the diverse masking effects with the synchronous world. Here, particularly the temporal masking seems to make a key difference. We want to study and experimentally evaluate that. The vision is to come up with a very generic understanding of masking effects in asynchronous circuits that allows for quantitative comparisons of implementation options and for targeted hardening of circuits. In case of permanent faults we are interested in better understanding and potentially leveraging the known "fail-stop" behavior of asynchronous (QDI) circuits for fast repair and recovery. Here a key challenge is to identify those cases where the fail-stop assumption does not hold. For both aims we need to select useful target circuits and design styles, and we need models that allow investigating these circuits on a level detailed enough to capture all relevant effects, but abstract enough to make the fundamental relations visible. I will be happy to receive inputs and thoughts about these questions.

Notes:

- Time domain agreed for async but what about value domain? (cf SET tolerance)
- Leverage for permanent faults
- A lot in the state of the art, but how to get systematic? Joint work with Milos (ENROL)
- Masking! Electrical, logical, temporal, matching bubbles and tokens/ Completion detection
- Approach: set of rep functional blocks – what is representative? what is appropriate analytical model?

Camera Sensors

Montek Singh
UNC Chapel Hill, USA

Notes:

- N Photons per time T – why time T is fixed? N varies from 10 to 50K
- Give each pixel its own time – fix the N and vary time.
- Astronomer's challenge – 1-5 photons – noise level?

Subjects:

- Teaching and Learning students – async in one semester!
- Research grants – we don't get enough funds to develop async tools!
- Startups
- Penetration into existing companies

Teaching Asynchronous

Mika Nyström
University of Southern California, USA

Notes:

- Work at intel for 20 years – showed some 'dark side'

- Async is looked esoteric at Intel
- Teaching: specification → decomposition → process graph design

Asynchronous in Mesochronous (Timing Analysis). Teaching and Tools: Where are We?

Jens Sparsø
Technical University of Denmark, Denmark

The first issue is CAD tools.

I have followed the field for 30 years and have seen tools come and go. When I teach asynchronous design to graduate students I can't really point them to tools except perhaps Petrify/WorkCraft for synthesizing speed independent controllers, should they need such control circuits in their design. Could our workshop produce an updated list of what CAD tools are available? What tools could/should I use in the next edition of my class on asynchronous circuit design.

The second issue is timing-analysis.

At DTU, we have developed an asynchronous time-division-multiplexed network-on-chip (called Argo) that we use in our real-time multi-core platform (called T-CREST). The timing organization of this multi-core processor allow: (a) individually clocked processors, (b) mesochronously clocked network interfaces, and (c) connected by an asynchronous network of routers and links. In every clock cycle, every network interface reads one token from the asynchronous network and writes one token into the asynchronous network. Tokens can carry valid data or voids, and in this way the circuit mimics a global-clock. The operation is similar to Mark Greenstreet's STARI, generalized to a structure with multiple input and output ports. The advantage is that the design offers time elasticity without any synchronizers. We believe the scheme can be used in many other contexts.

The correct and safe operation requires that the interfaces to the asynchronous network are able to complete a handshake cycle in less time than the duration of a clock period. This timing analysis problem was addressed by Hulgaard et al. in the 1990s and more recently in 2018 by Hua and Manohar. Questions: Are the timing models of the circuit realistic? Is it enough to focus on the steady-state operation. Are the developed algorithms implemented in tools? Etc.

Notes:

- Material is based on the book published 17 years ago
- Do structural design using Data Flow components
- Timing organization in multi-core; Core-NI-NoC-NI-Core; Network itself is scheduled at strictly time-shared tokenised slots. Mesochronous. Network should run faster than local clocks. Automatic verification is needed.

Asynchronous Ultra Low Voltage / Low Power CMOS - What and Why, but not much about How.

Snorre Aunet
Norwegian Univ. of Science and Technology, Norway

Notes:

- RISC implementation – 90% reduction in Energy/operation down to 300mV
- 32 bit adder logic was possible for 84mV for static CMOS (80nm)
- Sacrifice – doubling the area
- Delays vary by orders of magnitude due to temp and process variations when operated in subTh

Applications of QDI Circuits

Jia Di
University of Arkansas, USA

With multi-rail encoding and 4-phase handshaking, QDI circuits have some unique advantages and drawbacks, which make them suitable for a variety of niche market applications. This presentation introduces our previous and ongoing QDI circuit development work on several of such applications.

Notes:

- Don't compete with sync – identify applications/scenarios with async have unmatched advantages
- Robust circuit operation: flexible timing, wide range temperature, aggressive Vdd scaling, building with emerging devices; circuit stacking
- High Energy Efficiency
- No clock: distributed switching activities, 3D IC, power balancing for security
- DR handshake – Rad hard

Opportunity and Challenge of Latch-based Designs

Huimei Cheng
University of Southern California, USA

Latch-based designs have many benefits over their flip-flop based counterparts but have limited use partially because most RTL specifications are flop-centric and automatic conversion of FF to latch-based designs is challenging. Most conventional flows convert the FF-based designs into pulsed-latch designs or two-phase latch-based designs controlled by either master-slave clocks or bundled-data asynchronous controllers.

Pulsed-latch schemes are an intermediate approach that lies between latch and flip-flop based designs, however, are subject to hold problems and pulse width variations. Whereas two-phase designs are inherently more robust than pulsed-latch designs, multi-phase latch-based designs can sometimes be an attractive alternative. In this talk, I will show some details of a novel conversion algorithm targeting 3-phase bundled-data latch-based designs, evaluate its benefits, and articulate related remaining research challenges.

Notes:

- Desynchronization flow: convert to latch-based, and add async controller
- 3-phase latch based design; replace doubling with Master-Slave, to something like Master-Master-Slave
- Challenge – clock tree power in a multiphase clocking
- Idea: Combine retiming with clock gating

Teaching: from Principles to Tapeout in a Semester

Rajit Manohar
Yale University, USA

Where are the asynchronous designers? Where are the interesting asynchronous chips? Anyone that has tried to adopt asynchronous circuits for their application knows how hard it is to find qualified designers. We need a different way to educate students in asynchronous design that goes beyond a specialized graduate course. As an experiment, I tried to teach the classic "Mead and Conway" class, except students were asked to design asynchronous circuits rather than synchronous ones. I will report on the results of this experiment.

Notes:

- It's not just async! Many sync families exist. But abandoned!
- Will Zwaenepoel : we make things complicated! Fabricated complexity!
- Companies just do work that works; they ignore research
- Feynman's quote: "I couldn't explain it to the freshman level. This means we don't really understand it"
- Experiment: a Mead and Conway async class. Used new MOSIS schedule.

CAST+CSP+Proteus

Andrew Lines
Intel Strategic CAD Labs, USA

Notes:

- CAST + CSP + Proteus = Synthesis, Palce, and Route for QDI, BD, or SHS Circuits
- 2-input C-element 12 transistors (van berkel's) is used.
- 3-inputs C-element can be done in 12T – see Wikipedia on C-element

Asynchronous control for Analog-Mixed Signal

Alex Yakovlev
Newcastle University, UK

For years asynchronous design was considered in the context of constructing digital systems of ever increasing complexity. At the same time, in parallel with these developments there has been significant evolution of analog electronics. In the recent years due to the trends of the Internet of Things and pervasive IT we now witness new types of electronics, such as autonomous sensors, where the

boundary between digital and analog parts is blurred. These are exemplified in the needs of power-compute codesign which aims to look holistically at the issues of power efficiency and sustainability. Moreover, what used to be primarily analog devices, such as power converters the emergence of digital processing is evident – multi-phase bucks. In this talk we will explore how asynchronous logic helps interfacing analog and digital subworlds, by improving the quality of the overall mixed-signal system – power-compute efficiency, response time to events, size of components to name but a few. Entirely new computational paradigms based on time-base data representation become possible to drive new applications in the sphere of machine intelligence and 5G communications. The new wave of asynchronous for analog is paved through with the help of tools such as those under the Workcraft.org toolkit.

Notes:

- Target: to develop an EDA tool for the design of little digital or mixed-signal systems with asynchronous control and power modulated computing to achieve energy autonomy
- Tool support: Workcraft.org
- Newcastle's power-proportional, fully self-timed CPU: Intel 8051 in 130nm CMOS (2013)
- Self-timed SRAM chip: UMC CMOS 90nm (2010)

Specification and Verification of Link-Joint-Style System Designs

Warren A. Hunt, Jr.
University of Texas, USA

Notes:

- Boxing designs without crossing wires
- Layering the design is a better way
- You can even sometimes cut the design somewhere in the middle
- Different views (and reps) of designs can be useful for verification
- What's the biggest IP in the world now – x86 code!

Fault Tolerant Asynchronous Design

Milos Krstic
IHP-Leibniz-Institut fur Innovative Microelektronik, Germany

Asynchronous circuit design is known for its robust operation. The respective circuits could operate under wide operating conditions (voltages, temperatures), which makes them suitable for the reliability critical applications. The additional feature which is frequently required for such applications is fault tolerance against transient or even permanent faults. There have been several studies exploring the usage of asynchronous logic in fault tolerance domain. In thic context the major targets were either low-power applications, where timing faults play important role, or space applications, where the soft errors appear in registers and logic due to the radiation effects. Since today circuits can be utilized in the complex environments and scaled technology are increasingly error

prone, the new synergistic methods in design are required when we aim for error resilience. In this talk some ideas and initial results will be discussed addressing buth upsets/transients and timing faults in asynchronous logic.

Notes:

- Where are significant benefits: application which is async per se; in-memory computing for spiking neurons; trigger-signals in mixed-signal designs (designers don't want clock in high-speed circuits)
- Indirect benefits: EMI, security
- Are async design FT? what about SET, SEU?
- Control path sensitivity

Subjects:

- SETs can be detected by protocol monitors
- Async design can be stopped locally at short time
- Datapath sensitivity
- Memory protection against SEUs: DICE, TMR, HIT FFs; ECC codes for SRAM, NVM
- ENROL project – collaboration with TU Vienna
- Expectations from Shonan: better connection with async community; tool exchange; common benchmarks

Gradient Clock Generation

Matthias Függer
CNRS & LSV, ENS Paris-Saclay & Inria, France

Notes:

- They have similar things – Go signal, they may have handshakes or VCOs
- Sync and async – not much difference
- Phase difference is between adjacent block
- Topic 1: Gradient-clock synchronization algorithms
- Topic 2: discrete or continuous decision – controller that transforms continuous domains Discrete can be easier but you still have to use synchronizers. Alluding to MC-circuits
- Topics 3: Circuits that work in distributed environments – microbiological circuits

Asynchronous Energy-Efficient CNN Accelerator Design with Reconfigurable Architecture

Hong Chen
Tsinghua University, China

Notes:

- Low power design for APP detection system; MCU subTh
- Click based MCU design
- Async CNN accelerator

- ASIC design – 3x energy efficiency (GOPs/W) reported
- IBM and Intel had async Spiking NNs in 2015 and 2016

Breaking Through the Inertia: the Force to Move Asynchronous Design into the Mainstream

Ken Stevens
University of Utah, USA

Notes:

- Define the Goals: can we unite on a vision? Technology, visibility, products; Model: open source/patent rights; 3-5 ambitions yet realistic
- Drill down to Forces: why change? What to change and how to change?
- Leadership: create statement and governing body
- Ant-dream-team philosophy; pick teams by personality – right players rather that best players
- Bold and Achievable initiative: Digital h/w is as easy to write as Digital s/w – TIMING is the key aspect
- 1956 – FORTRAN; 1958 – Clocked design
- S/w progressed since then, but FSMs are still clocked
- Relative Timing

Coarse Grained vs. Fine Grained Architectures for Asynchronous Reconfigurable Devices

Tomohiro Yoneda
National Institute of Informatics, Japan

Currently, FPGAs are widely used in many applications, and it's possible to implement very large circuits on them. One drawback in implementing a large circuit on an FPGA is that it becomes more and more difficult to satisfy the timing constraints related to clock signals. Furthermore, some applications require circuit blocks with different clock frequencies. For these reasons, the GALS (Globally Asynchronous Locally Synchronous) approach is useful also for FPGAs. In this work, we consider a reconfigurable asynchronous component based on a coarse grained architecture for implementing a low-overhead communication mechanism for GALS systems. Its element block is based on the MOUSETRAP pipeline stages and the configurable logic blocks are specialized for implementing communication functions. In this talk, I will show some details of the proposed architecture, and evaluate its performance comparing it with a conventional fine grained FPGA architecture.

Notes:

- Problems in current FPGAs: clock to distribute in the entire chip; multi-clock solutions maybe possible
- Examples in state of the art: GALS FPGA; Achronics – fine grain; Coarse grain – CGRA
- Proposed- more flexible that GALS, use coarse grained async
- More specialized for comms than for GALS

- Separate data path from control path in interconnect
- Almost completed now. Evaluation is coarse grained vs fine grained; async FIFO, crossbar, async router; 130nm technology
- Throughput comparison – everything about 3x against fine-grained
- Future: Domain specific FPGAs – eg. Deep learning accelerators

Correct-by-Construction Hardware Synthesis

Daniel Zimmerman
Galois, Inc., USA

Notes:

- About Galois: est. 1999 to apply functional programming to information assurance; about 100 people working on more than 40 projects for both US government agencies (NSA, DARPA, DHS, AFRL) and commercial companies (Amazon, MS); "high assurance everything", but until recently, mostly software.
- Cryptol is Galois's core cryptographic specification and verification tool, and oldest project; in 2006, spun off a company to synthesize cryptographic algorithms from Cryptol specifications to FPGA implementations, but this effort failed due to the 2007-08 financial crisis.
- In 2015, implemented an asynchronous ASIC with 3 ciphers (AES, Simon, Speck) by synthesizing SystemVerilog-CSP from Cryptol, as part of GULPHAAC: Galois Ultra Low Power High Assurance Asynchronous Cryptography.
- This year, started a new DARPA project – 21st Century Cryptography – to extend Galois's synthesis capabilities and implement a side-channel resistant, low power proof-of-concept cryptographic ASIC. $5.4M over 16 months, with USC and Portland State. The resulting synthesis capabilities need not be restricted to cryptographic algorithms.
- Galois loves to collaborate, and has many academic and industry partners on our programs.

Practical Mixed Asynchronous/Synchronous Design

William Koven
Galois, Inc., USA

Notes:

- Interests: build interesting chips; sell interesting chips, design async circuits
- The more interesting chips you sell, the more you can build!
- Previously built the REM R0: Mixed async/sync; sync IP: MIPI-CSI, USB3, LPDDR4, AXI – we just didn't want to rebuild them ourselves (not interesting) – easier to buy and plug together
- Async blocks included AI core and CPUs. CPUs were de-synchronized. But AI Core was designed as an async block from the ground up.
- GLASS-CV, new chip at Galois: async components include – Vision accelerator, RISC-V cores, JPEG encode/decode, Security Core (all built in 14nm)

- If anyone wants to build an open-source async AXI that would be extremely welcome
- It doesn't matter how a chip compares to sync. It matters how it compares to devices on the market (that may happen to be synchronous)

Meditation on Metastability

Mark Greenstreet
The University of British Columbia, Canada

Notes:

- What do I do: m/s and synchronizers, CDC, Verification
  What makes a good synchronizer? Can we test it in the lab? What are the fundamental limits on bandwidth/latency? What are the fundamental bottlenecks? Architects are risk averse – they need compelling guarantees before trying something new!
- Inflection:
  Moore's law has been innovation killer! IBM 360, PDP-11, x68, ARM, RISC-V all the same; UNIX turns 50 next month; all PLs are the same – FORTRAN, C, Java, Python $\cdots$ End of Moore's law $\rightarrow$ parallel architectures are mainstream: GPUs, FPGAs, Machine learning; Data flow and FP are used in practice!
- Is it time for async?
  Chips are multi-timed ensembles of comm systems, Async gives fine grained concurrency, activity management, data-transport; What are compelling cases to exploit async
- What I hope to learn from you?
  Compelling async use cases; ways to exploit fine-grained concurrency: system-level description; formal verification; Post-Silicon, Post-von Neumann. Can anything dethrone Si?

Challenge of Nonvolatile Logic LSI for IoT Applications

Takahiro Hanyu
Tohoku University, Japan

The impact of artificial intelligence (AI) is being widely understood in several application fields and its technology/application challenge will be increased more and more. In contrast, when you use AI software under the cloud environment, its performance would be limited in several applications (ex. when many people access the cloud computer simultaneously, its data traffic might be hanged up. Therefore, it would be difficult to always receive the response from the cloud machine within a real time). In this sense, it is getting important to develop special-purpose AI hardware. I would like to explain several problems of conventional CMOS-only-based AI hardware in terms of power dissipation, and its solution to overcome the above problems. As a concrete example, l introduce the possibility and the usefulness of non-standard logic-circuit design methodology using functional devices (such as magnetic tunnel junction devices); called "non-volatile logic-in-memory" architecture.

Notes:

- History: Many ISSCC papers from 1985 till now: 4-valued image processor, multi-valued current mode multiplier LSI, Ferroelectric based NV logic LSI, MTJ-based MCU LSI
- MTJ-based NV MCU LSI (ISSCC 2019)
- Motivation: demand for LP, HP MCUs
- EH – 100uW – requirement for IoT sensor nodes powered by EH
- Operation frequency: > 200MHz
- Research interest in future: Logic in memory structure – non-standard logic – based on device/material
- Spin/charge/orbital/lattice

CMOS Invertible Logic using Stochastic Computing

Naoya Onizawa
Tohoku University, Japan

Invertible logic can operate in one of two modes:1) a forward mode, in which inputs are presented and a single, correct output is produced, and 2) a reverse mode, in which the output is fixed and the inputs take on values consistent with the output. It is possible to create invertible logic using various Boltzmann machine configurations. Such systems have been shown to solve certain challenging problems quickly, such as factorization and combinatorial optimization. In this talk, we show that invertible logic can be implemented using simple spiking neural networks based on stochastic computing. We present a design methodology for invertible stochastic gates, which can be implemented using a small amount of CMOS hardware. We demonstrate that our design can not only correctly implement basic gates with invertible capability, but can also be extended to construct invertible stochastic adder and multiplier circuits. Experimental results are presented which demonstrate correct operation of synthesizable invertible circuitry performing both multiplication and factorization, along with fabricated ASIC measurement results for an invertible multiplier circuit.

Notes:

- Lack of tools for async design – and hence gave up some time ago
- CMOS invertible logic: probabilistic magnetoresistive device model (sonventional work) Input is analog voltage; output is digital stochastic sequence
- Comparisons with traditional, reversible, invertible – and now CMOS invertible
- Approximation using stochastic computing with CMOS digital circuits

4 Overview of Panels

Panel 1: Open source

Chair: Alex / Panelists: Andrew, Rajit

Position talk and discussion:

- Andrew:
  - Our 25 years of async and CAST (Caltech Asynchronous Synthesis Tools)
    - 1993-2000 @ Caltech: DSP,CPU / QDI PCHB / CAST Language / Minimal CAD / Manual Layout
    - 2000-2011 @ Fulcrum Microsystem: Network switch products / QDI PCHB / CAST + Some Verilog / Vastly improved CAD / Proteus Synthesis, Place & Route
    - 2011-2015 @ Intel: Networking / GALS + QDI / Some CAST / Mostly Verilog / Similar CAD
    - 2015-2019 @ Intel Labs: Neuromorphic / Bundled-Data / CAST Language / Verilog for CPU / New CAD for BD
  - CAST is a HDL and tools for async circuit and is also used to define the whole system at the top level
  - Synchronous blocks in Verilog and they are pointed
  - For simulation purposes parts in CAST is also translated to Verilog
  - Synthesizable CSP and Proteus
  - Very compact 25x shorted than Verilog – productivity advantage
  - CAST/CSP example: Mandelbrot Set
- Rajit:
  - Called Hierarchical Production Rules
  - Re-implemented everything
  - CHP (renamed CSP at Tony Hoare's request; e.g.: Probes are not in CSP)
    - defproc – to define a process type
    - defchan – for channel type
    - deftype – for data type

    all definitions are available on the web
  - Tools:
    - CHP is a starting point
    - Decide the microarchitecture
    - *CHP – this captures all the concurrency – explicit and at this level of abstraction you can start doing h/w
    - One route Handshaking expansion (HSE) and PRS
    - Sizing, Sized PRS, SPICE
  - no technology mapper!
  - 3 categories;
    - Front-end
    - Middle-end
    - Back-end

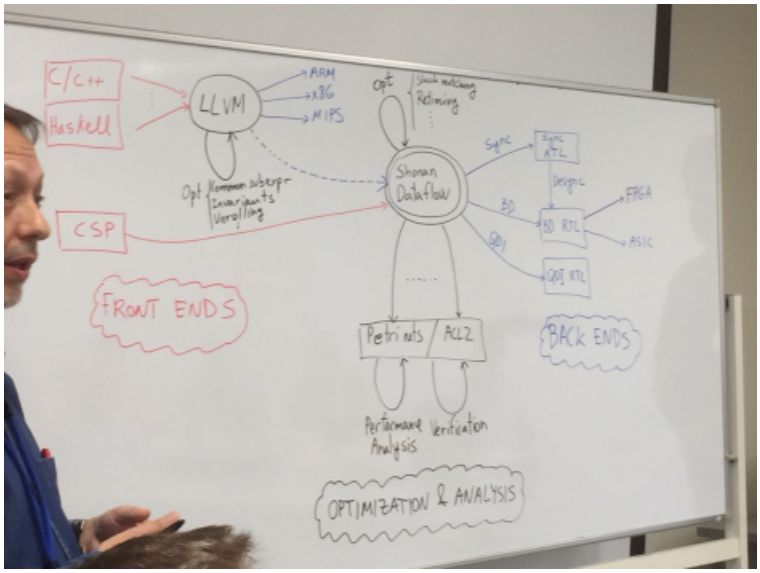
Panel 2: Community involvement

Chair: Montek / Panelists: All

Goal of the session: how multiple stakeholders can be part of the community in reflection to what Andrew and Rajit

Position talk and discussion:

- Marly: where we might fit in. CSP spec is OK, there is still a gap between CSP and single cell process; I want more compilation there. We do guarded commands at channels, at behavioral level.
- Ken: what level of community would you like to involve? Can it be put on github.
- Peter: I am agnostic to control, I care about logic and latches. Where do I enter this flow? I'll have a body for CSP for combination logic. What to do about latches? If I have sync RTL (in Verilog), how do you recommend this added to the flow?
- Rajit: we have ACT is the language and we have v2act and act2v
- Jordi: Dataflow Intermediate Representation as a common language to build front/back-ends from different high-level languages to various asynchronous/synchronous implementations (with FPGAs or ASICs).



- Mark: We need a regression suite. So we can fix the shortcomings.
- Ney: We need a roadmap – of what parts and routes through the flow-map we can have and when
- Rajit: Showed Toolflow and interfaces
- Alex: What kind of user do you envisage for the open source flow: an SME, students ??

- William answered from Galois – they intend to use them, contribute and keep them in the open source domain

Panel 3: Next day planning

Chair: Peter

Position talk and discussion:

- People come with a wish list (10 mins max):
  - This is what I like in this tool flow.
  - What would like I to have in this flow?
  - What am I willing to do?
- Proposed panels and discussions:
  - Panel Education: People who want to teach Async – Morning (1.5hrs) – chair: Mika
  - Panel Applications: People who want to build things (power and/or performance) – Morning (1.5hrs) – chair: Hong
  - Panel Developers: People who want to write some CAD and IP, in particular design style (e.g. BD) (1.5hrs) – chair: Scott
  - Free discussion

Panel 4: Education

Chair: Mika / Panelists: Peter, Rajit, Montek, Jens

Position talk and discussion:

- Mika's intro:
  - Teaching Engineers is a mission to give tools and weapons to people for survival.
  - Quotes from the Bible
  - John 8:32: And ye shall know the truth, and the truth shall make you free
  - (also Motto of Caltech)
  - Program testing can be used to show the presence of bugs but never to show their absence – EWD249 (Dijkstra)
  - It is the professor's task to bring the relevant insights and ability into the public domain by explicit formulation EWD1175 (Dijkstra)
- Peter:
  - 17 weeks*3hrs/week = 51 hrs
  - 1st year Masters, compulsory
  - I'd love to put this on web and expect some additions or editions from others
  - There are Verilog, System Verilog, SVM – for those who will never see async in their lives.

- Rajit:
  - 1999, Caltech:
  - CS 181 VLSI Design Laboratory
  - Now: Yale EENG 426/ CPSC 876: Silicon Compilation
- Montek:
  - Teaching Computer Organization, Computational Photography (Image sensors etc.)
  - Teaching about 50 students, who may not see digital design in the future at all.
- Jens:
  - Teaching Async design for EEE students. Somebody else is teaching VLSI design.
  - 12 week course
  - Use the Book (by Sparsøand Furber)
  - Models used DF, STGs, Workcraft
  - Knowledge: theory, performance analysis, m/stability, NOCs, GALS, desynchronization
  - Projects: based on some papers – different topics to be decided.

Panel 5: IP & library developer

Chair: Scott / Panelists: William, Jia, Ney, Jordi

Position talk and discussion:

- William: What is it to build an industrial chip?
  - Re-use 80%-90% of existing IP blocks and the rest is to be build.
  - Wish to plug things together in largely delay-insensitive manner
  - Examples: used Mark's FIFO; would be nice to have async AXI
  - How to increase productivity of the design team – where with 10% of time we could get 90% performance improvement
- Jia:
  - Technology independence. Technology dependent – would be IP sensitive so can't be open sourced
  - Applied to NSF three times: Building a website with async design solutions for whoever is interested in async; examples:
    - NCL synthesis tool: UNCLE: `https://www.sites.google.com/site/asynctools/`
    - NCL VHDL library and older technology CMOS libraries: `https://www.ndsu.edu/pubweb/~scotsmit/CCLI_async.html`
  - We can have "faked delay" Liberty libraries
  - Suggestions from the audience:
    - Maybe we can have our library in PTM?
    - Through MOSIS you can have access to repositories of libraries

- Ney:
  - On wiki in Asynchronous circuits, there is no mention of benchmarks, libraries and tools!
  - Open cell libraries for Async:
    - 1993: Wuu/Wrudula, USC/UArizona – 500nm
    - 1998: Renaudin, Grenoble
    - 2007: Chong et al
    - 2004/2005: Beerel/Ferretti
    - 2003: Maurine, Grenoble
    - 2010: ASCEnD Libs from Ney's group
    - Examples: from 2-input C-elements to 4-input NCL gates
    - ASCEnD-ST65/ST28: 1080 cells (600 layout/ready)/several hundreds and no layout
    - ASCEnD-FreePDK45: 30 cells
    - Ongoing: ASCEnD-TSMC180: 48 cells
  - Questions:
    - Do we need libraries? Yes, to attract more async users, fast design, further the cell design process
    - Do we have libraries to make available?
      - NDA problems, company restrictions
      - Services – Mosis, French CMP, Europractice, IHP
    - What do libraries support?
      - NCL, PCHB, Blade
      - NearTh and SubTh for all the above ⋯
    - Do we need to have circuits made with these libraries?
      - Benchamarks for Micropipelines (BD), NCL,MTNCL, PCHB; Blade
  - Roadmap for Open Cell Libs?
    - Define std for Open lib structure
    - Provide examples with ND clearance
    - Produce a model for lib
  - Examples shown. Provided: desc, lef, lib, Verilog
  - Virtual Functions – e.g. mappings from NCL gates to NANDs,NORs etc. to 'fool commercial tools'
- Jordi:
  - Satisfiability-based approach for layout synthesis
    - Tool reads a netlist of transistors with design rules and produces a layout (routing) on a 3-D grid.
    - Video of Sat-solver in action
    - Available on the web: `http://www.cs.upc.edu/~jpetit/CellRouting/`
    - SAT-solver gives the legality check
    - Placement is done separately; some placements are routable and some aren't
  - Data Flow
    - Started in the 1970s at MIT (Jack Dennis and David Misunas)

    * In Async community it was in 2001 book by Sparsøand Furber
    * 2004: Rajit had a paper
    * 2018: EPFL people in FPGA
    * 2012: xMAS at Intel
    * EPFL method:
        · Throughput constraints and slack matching → ILP

Panel 6: User application

Chair: Ken / Panelists: Milos, Hong

Position talk and discussion:

The background is that we can all build new methods and tools but at the end of the day we need people to use them.

- Milos: Applications of interest
  - Not sure Async would be useful in apps where Sync has been established.
  - So, those apps where things haven't been done
  - Mixed-signal design
  - Robust design
    * Low noise: EMI, substrate noise
    * Reliability: FT (SEU,SET), Extreme Temps, Voltages
    * Side channel attack resistant (IHP did with GALS)
  - Neuromorphic (CMOS+, such as added memristors)
  - Flow issues: Spec, RTL-netlist, Layout; plus DFT
- Hong
  - Low power is the main focus!
  - Analog part consumes most power
  - SubTh microprocessor design
  - Wireless Monitor System of the Total Knee replacement: SoC design
  - CNN accelerator
- Discussion: Creating a library of regression tests
- Question: What about language to adopt?
  - The community came together and rallied around CHP (a variant of Hoare's CSP) as the common programming language for the community
  - Rajit will lead development and manage a repository
- Question: What about Energy Harvesting?
  - Use of PZT power source from weight for monitoring an implant in the knee:
    * Harvested Power: 400uW
    * Output voltage – 2-2.5V, ripple voltage – 400mV
- Question: Can we do anything for G5?

Panel 7: Next day planning

Chair: Peter

Position talk and discussion:

- Questions: advantage often cited, not-often cited @ASYNC1994
- Advantage often cited @ ASYNC1994:
  - 1. Achieve average Case Performance
  - 2. Power consumed only when needed
  - 3. Ease of modular composition
  - 4. No clock alignment at the interfaces
  - 5. Metastability has time to end
  - 6. Avoid clock distribution costs
  - 7. Easier to exploit concurrency
  - 8. Intellectual challenge
  - 9. Intrinsic elegance
  - 10. Global synchrony does not exist anyway
- additional advantage often cited @ SHONAN133:
  - 11. Low EMI/noise
  - 12. Process Bring-up support
  - 13. Robust to PVT variations
  - 14. Easier to interface to analog
  - 15. Outlives CMOS
  - 16. Intrinsically bio-inspired
  - 17. Continuous time information processing
- NOT often cited @ ASYNC1994:
  - 1. It really pisses my boss off
  - 2. I like reinventing wheels
  - 3. I like to be different
  - 4. Gee – I really don't know
  - 5. People and circuits need to play by same rules
  - 6. I don't understand synchronous circuits
  - 7. World problems stem from glithces
  - 8. Synchronous design gives me gas
  - 9. Clock radiation causes hair loss
  - 10. It's none of your business
- addtional NOT often cited @ SHONAN133:
  - 11. Clocks make me late
  - 12. Asynchronous design makes me GasP
  - 13. Forever the future technology
  - 14. Timing is where the rubber hits the road
  - 15. It confuses proposal reviewers
  - 16. What else to call computing without clocks
  - 17. I'm still waiting for an ack

Panel 8: Tangible Next Steps

Chair: Peter

Position talk and discussion:

- Asynchronous mailing list
  - Managed by Slack
    - Teaching
    - Open source / community information
    - Applications / users
- Web pages / Asynchronous on Wikipedia
  - Asynchronous web page managed by Manchester's group has been gone
  - Steering Committee of ASYNC symposium supports the asynchronous web pages hosted at Wiki
- Meet again
  - One day follow-up meeting after ASYNC2020
  - Invitation only, separate registration and separate web page
  - Local arrangement chair will manage them
  - Apply Dagstuhl meeting at ASYNC2022
- Teaching materials, videos, tools
  - Links from web page (Huge materials cannot be gatherd)
  - Community Github
- Dream: Asynchronous Funding

5 List of Participants

- Prof. Tomohiro Yoneda, *National Institute of Informatics, Japan*
- Prof. Peter A. Beerel, *University of Southern California, USA*
- Prof. Alex Yakovlev, *Newcastle University, UK*
- Prof. Masashi Imai, *Hirosaki University, Japan*
- Prof. Snorre Aunet, *Norwegian Univ. of Science and Technology, Norway*
- Prof. Ney Laert Vilar Calazans, *Pontifical Catholic University of Rio Grande do Sul, Brazil*
- Prof. Hong Chen, *Tsinghua University, China*
- Ms. Huimei Cheng, *University of Southern California, USA*
- Prof. Jordi Cortadella, *La Universidad Peruana de Ciencias Aplicadas, Spain*

- Prof. Jia Di, *University of Arkansas, USA*
- Dr. Matthias Függer, *CNRS & LSV, ENS Paris-Saclay & Inria, France*
- Prof. Mark Greenstreet, *The University of British Columbia, Canada*
- Prof. Takahiro Hanyu, *Tohoku University, Japan*
- Prof. Yong Hei, *Portland State University, USA*
- Prof. Warren A. Hunt, Jr, *University of Texas, USA*
- Mr. William Koven, *Galois, Inc., USA*
- Prof. Milos Krstic, *IHP-Leibniz-Institut fur Innovative Microelektronik, Germany*
- Mr. Andrew Lines, *Intel Strategic CAD Labs, USA*
- Prof. Rajit Manohar, *Yale University, USA*
- Dr. Mika Nyström, *University of Southern California, USA*
- Prof. Naoya Onizawa, *Tohoku University, Japan*
- Prof. Marly Roncken, *Portland State University, USA*
- Prof. Hiroshi Saito, *University of Aizu, Japan*
- Mr. Shogo Semba, *University of Aizu, Japan*
- Prof. Montek Singh, *UNC Chapel Hill, USA*
- Prof. Scott C. Smith, *North Dakota State University, USA*
- Prof. Jens Sparsø, *Technical University of Denmark, Demark*
- Prof. Andreas Steininger, *Vienna University of Technology, Brazil*
- Prof. Ken Stevens, *University of Utah, USA*
- Prof. Ivan Sutherland, *Portland State University, USA*
- Prof. Nobuyuki Yoshikawa, *Yokohama National Univ, Japan*
- Dr. Daniel Zimmerman, *Galois, Inc., USA*