The Early History of Asynchronous Circuits and Systems

Charles L. Seitz, Ph.D.
President & CEO of Myricom, Inc.
chuck@myri.com

Async 2009
UNC Chapel Hill
My many debts to Ivan E. Sutherland

• Ivan Sutherland invited me to give this talk, and suggested the topic
• I have had several wonderful mentors in my career, none more significant in their influence on me than Ivan
  – Ivan has been a senior colleague and friend since we met in the summer 1966 at MIT Lincoln Laboratory. We have worked together off and on ever since
    • Harvard, Evans & Sutherland Computer Corporation, Caltech, ...
• Ivan's answer to many requests for guidance through the years has been:
  "Something old, something new, something borrowed, something blue."
  – and I’ll try to follow some of that advice today.
The 1950s: The Dawn of Switching Theory

- Mathematical techniques started catching up with what practical “logic designers” had been doing.
- Time-discrete network models such as McCulloch-Pitts Neurons (1943-).
  - In 1956 Kleene showed the correspondence between these finite automata and regular sets (i.e., regular expressions)
    - thus making a connection between switching and automata theory and the adjacent field of formal languages.
- Finite-state machines
  - David A. Huffman: “The Synthesis of Sequential Switching Circuits” (1953, MIT) -- Asynchronous
  - George H. Mealy: “A Method for Synthesizing Sequential Circuits” (1955, Bell Labs) -- Synchronous
  - Edward F. Moore: “Gedanken-experiments on Sequential Machines (1956) -- Synchronous
The View from 1960 (~50 years ago)

- Finite-state machines were the model
  - Shown equivalent in capabilities to regular expressions
  - Canonical. No theoretical issues about equivalence
  - Extensive body of theory

- Design of “Asynchronous Sequential Circuits”
  - Regarded as a solved problem (Huffman’s synthesis techniques)
  - Generalized assignments, etc.

- What more could there be to do?
Speed-Independent Circuits

- In addition to Huffman circuits, there were also the speed-independent circuits of David Muller (1955 - 1963)
- Stephen H. Unger, in his 1969 book, Asynchronous Sequential Switching Circuits, writes in one section:

"The concept of speed-independent circuits, in which completion signals obviate the need for worst case designs based on estimates of maximum internal delays, was first introduced by Muller and his associates [refs]. These sources deal principally with autonomous circuits, do not present general synthesis techniques, and are not easy to read."
An aside about David A. Huffman

- I knew him quite well both at MIT and later at UC Santa Cruz
  - A key figure in a series of education meetings I organized for IFIPS
- Endless interests
  - Computational origami
  - kept rattlesnakes as pets

David A. Huffman in the mid-1990s
Nevertheless

• There are serious limitations of Huffman synthesis of asynchronous sequential machines
  – Single input changes only
    • no concurrent input changes
  – No explicit representation of “domain constraints” (the behavior of the environment in which the circuit operates)
    • essential for designing circuits that generate completion signals
  – Complex rules about races and hazards
  – Useful only for “design in the small”
    • finite-state models are often impractical for the design of digital systems of more than "textbook example" complexity. The number of items required to specify a finite-state machine grows as the product of the number of states (exponential with the number of state bits) and the number of input combinations (exponential with the number of input signals).
    • This limitation applies also to synchronous state machines.
A simple example of Huffman Synthesis

Primitive flow table: \( ns, c \)

\[
\begin{array}{|c|c|c|c|}
\hline
ab & 00 & 01 & 11 & 10 \\
\hline
0 & 2 & - & 1 \\
0 & - & 3 & 1 \\
0 & 2 & 3 & - \\
- & 4 & 1 & 5 \\
0 & 4 & 1 & - \\
0 & - & 3 & 5 \\
\hline
\end{array}
\]

Merged flow table: \( ns, c \)

\[
\begin{array}{|c|c|c|c|}
\hline
ab & 00 & 01 & 11 & 10 \\
\hline
0 & 2 & 3 & 1 \\
0 & 4 & 3 & 5 \\
\hline
\end{array}
\]
How I would represent a C element

after I met Anatol W. Holt in 1967

or, depending on taste, you could equivalently represent the C element with a set of production rules

The marked graph above happens to be conflict-free, no cases of:
How I would represent a C element

after I met Anatol W. Holt in 1967

or, depending on taste, you could equivalently represent the C element with a set of production rules

The marked graph above happens to be *conflict-free*, no cases of:
How I would represent a C element

after I met Anatol W. Holt in 1967

The marked graph above happens to be *conflict-free*, no cases of:

or, depending on taste, you could equivalently represent the C element with a set of production rules
How I would represent a C element

after I met Anatol W. Holt in 1967

or, depending on taste, you could equivalently represent the C element with a set of production rules

The marked graph above happens to be conflict-free, no cases of:
How I would represent a C element

after I met Anatol W. Holt in 1967

or, depending on taste, you could equivalently represent the C element with a set of production rules

The marked graph above happens to be conflict-free, no cases of:

Myricom  www.myri.com  Chuck Seitz, Async’09, Chapel Hill, NC - 20 May 2009
How I would represent a C element 

after I met Anatol W. Holt in 1967

or, depending on taste, you could equivalently represent the C element with a set of production rules

The marked graph above happens to be conflict-free, no cases of:

\[ \text{Conflict} \]
How I would represent a C element

after I met Anatol W. Holt in 1967

The marked graph above happens to be conflict-free, no cases of:

or, depending on taste, you could equivalently represent the C element with a set of production rules.
Just for fun
The view from 1970 (~40 years ago)

• Blossoming of interest in concurrency
  – Anatol Holt’s “events and conditions” theory
  – Asynchrony was assumed a part of concurrent systems
  – Computational schemata
    • Partial orderings of computations (such as “WSJ” networks)
  – Request/acknowledge signaling; completion signals; …

• The Macromodule project at Washington University
  – Elegant! (Wesley Clark and Charlie Molnar)
  – “MSI” level blocks, separate control and data cables
  – Operators stacked over registers, 12 bits, 24 bits, 36 bits, …
  – Similar to graphical programming (WSJ networks)

• At MIT, Jack B. Dennis taught in his classes that future computers would be asynchronous and exploit concurrency.
Jack Dennis’s Woods Hole meeting (1970)

Record of the
PROJECT MAC
CONFERENCE ON
CONCURRENT SYSTEMS
AND PARALLEL
COMPUTATION

June 2-5, 1970
Woods Hole, Massachusetts

ASSOCIATION FOR COMPUTING MACHINERY 1970

Available as a pdf download from portal.acm.org
Interest in the study of concurrency and parallelism in systems and in computation is rapidly increasing, and the concepts evolving from this interest seem destined to have profound influence on the future course of computing practice. The Project MAC Conference on Concurrent Systems and Parallel Computation was arranged to encourage intercommunication among workers in four distinct but intimately related lines of conceptual development in this field.

The oldest line concerns the concept of speed independent sequential circuits, first formulated definitively by David Muller. Associated with this concept is the notion of building digital systems in the form of modules that intercommunicate by asynchronous signalling. Although the design of the Illiac II computer was influenced by the concept of speed independent logic, only recently have Muller's ideas been adopted by others as a subject of serious research. The papers in Part II of these proceedings are representative of current thinking in this area.

The second line of research has been called "systemics" by Anatol Holt who is singularly responsible for its development in the United States. Systemics is the result of many years' study of alternative methods of representing the behavior of complex systems. The present form of the material developed from the acquaintance of Holt with the ideas of Carl Adam Petri expressed in his dissertation to the University at Bonn, Germany.

Part I of this Record is the first formal publication of the work of Holt and his colleagues, and we hope it will be as stimulating to the neophyte as it has been to those of us fortunate enough to have had informal access to these ideas.
Woods Hole contents, part 1

TABLE OF CONTENTS

| PARTICIPANTS | viii |
| FOREWORD | v |

PART I. SYSTEMICS

| Events and Conditions | Anatol W. Holt |
| Frederick Commoner |

I. Introduction | 3 |
II. System Description | 7 |
III. Properties of Marked Graphs | 25 |
IV. State Machines and Information | 33 |

PART II. ASYNCHRONOUS MODULAR SYSTEMS

| Modular, Asynchronous Control Structures | Jack B. Dennis |
| for a High Performance Processor |
| Decompositions of Control Networks | Stanley M. Altman |
| Peter J. Denning |
| Asynchronous Machines Exhibiting Concurrency | Charles L. Seitz |
| Closure Properties of Interconnections of Determinate Systems | Suhas S. Patil |

53 55 81 93 107
Woods Hole contents, part 2

PART III. COMPUTATION SCHEMATA

Comparative Schematology
Michael S. Paterson
Carl E. Hewitt

Flow Graph Schemata
Donald R. Slutz

On the Determinacy of Schemata
Peter J. Denning

PART IV. COMPUTATIONAL PROCESSES

Coordinated Sharing of Resources in Asynchronous Systems
Prakash G. Nebalkar

Locking
Jim Gray

Computational Processes
Dennis Tsichritzis

BIBLIOGRAPHY
# Woods Hole participants

**List of Participants**

<table>
<thead>
<tr>
<th>Name</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prof. S. Altman</td>
<td>Princeton University</td>
</tr>
<tr>
<td>Mr. F. Bradshaw</td>
<td>Case Western Reserve University</td>
</tr>
<tr>
<td>Prof. T. Brodt</td>
<td>Stanford University</td>
</tr>
<tr>
<td>Mr. W. Brinsfield</td>
<td>Bell Telephone Laboratories</td>
</tr>
<tr>
<td>Prof. D. Cohen</td>
<td>Harvard University</td>
</tr>
<tr>
<td>Mr. P. Commoner</td>
<td>Applied Data Research</td>
</tr>
<tr>
<td>Prof. P. Denning</td>
<td>Princeton University</td>
</tr>
<tr>
<td>Prof. J. Dennis</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>Dr. J. Gray</td>
<td>University of California, Berkeley</td>
</tr>
<tr>
<td>Mr. P. Hebalkar</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>Mr. C. Hewitt</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>Mr. R. Kalin</td>
<td>Lincoln Laboratory, M.I.T.</td>
</tr>
<tr>
<td>Dr. W. Kautz</td>
<td>Stanford Research Institute</td>
</tr>
<tr>
<td>Mr. R. Keller</td>
<td>University of California, Berkeley</td>
</tr>
<tr>
<td>Prof. D. Luckham</td>
<td>Stanford University</td>
</tr>
<tr>
<td>Prof. D. Martin</td>
<td>University of California, Los Angeles</td>
</tr>
<tr>
<td>Prof. E. McCluskey</td>
<td>Stanford University</td>
</tr>
<tr>
<td>Mr. R. Miller</td>
<td>University of California, Berkeley</td>
</tr>
<tr>
<td>Prof. D. Muller</td>
<td>University of Illinois</td>
</tr>
<tr>
<td>Prof. M. Paterson</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>Dr. S. Patil</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>Mr. C. Rose</td>
<td>Case Western Reserve University</td>
</tr>
<tr>
<td>Prof. C. Seitz</td>
<td>University of Utah</td>
</tr>
<tr>
<td>Mr. L. Seligman</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>Dr. D. Slutz</td>
<td>International Business Machines</td>
</tr>
<tr>
<td>Dr. R. Strong</td>
<td>International Business Machines</td>
</tr>
<tr>
<td>Prof. D. Tsichritzia</td>
<td>University of Toronto</td>
</tr>
</tbody>
</table>
Design in the Large

• Practical designers were by 1970 doing large-scale asynchronous designs in spite of the lack of “theory” and mathematical models.

• Computational schemata offered
  – Quasi-formal designs and documentation
  – Some ties to analysis
  – Some ties to (distributed and concurrent) programming

• Modular design
  – Macromodules at Washington University, St. Louis
  – Jack Dennis’s modular design project at MIT
  – Modular (processors+memories+IO) systems from companies
Synchronization Failure & Mutual-Exclusion Elements

Also a view from 1970, +/- a few years

“Getting the word out”
Figure 1(a): Simple-minded Arbiter (mutual exclusion) circuit

The circuit is intended to arbitrate the use of some part of a system between A and B, permitting only one requesting user at a time. The AND gates have an intrinsic delay.
Figure 1(b): Behavior of the simple-minded arbiter, illustrating the "After you, Alphonse!" behavior at a "simultaneous" request. (Traced directly from an oscilloscope, 20 nsec./cm. horizontal.)
Similar picture from Cheney & Molnar 1973

Fig. 3. \( Q \) and \( \overline{Q} \) TTL cross-tied NAND gates (SN7410) with the \( R \) and \( S \) inputs switched high simultaneously (5 ns/div, 1V/div).
As noted, this paper followed the April–1972 Workshop on Synchronizer Failures (Bromwoods, outside St. Louis)
The switching behavior of ECL, used in Macromodules, is similar to that of CMOS.
Who discovered synchronization failure?

- 1965 published paper by Ivor Catt
  - Incorrect in a few details, and widely disputed
  - Of course, this paper was largely ignored
- However, many other people knew about synchronization failure and metastability
  - e.g., in 1972-73, I heard David Wheeler give an elegant exposition at a symposium at the University of Newcastle
Of course, synchronization failure is now solved
Mutual Exclusion Elements

I started making and testing ME elements from SSI and discrete components at MIT in 1968, and then at the University of Utah. ME elements on chips follow the same principles. The following figure is from chapter 7 in Mead & Conway, Introduction to VLSI Systems, 1980.
**Pausible (Start/Stop) Clocks**

The Evans & Sutherland Line Drawing System 1 (LDS-1), somewhat a derivative of the Harvard 3D display system. First shipped 1968 (?)

![Diagram of the graphics pipeline]

Each computing unit in this “graphics pipeline” had its own start/stop clock, and the communication between the computing units used self-timed, asynchronous, request/acknowledge signaling (bundled data).

Not done this way to be “clever,” but to solve clocking problems we encountered with the Harvard 3D display. “Synchronize the clocks to the signals, not the signals to the clocks.”
Chip version of the start/stop clock

Exactly like the start/stop clocks in the E&S LDS-1
Self-timed Systems

• Why I coined the name (mid-1970s):
  – “Each system part keeps time to itself”
  – To me, “asynchronous” meant Huffman asynchronous sequential machines
  – Calling something “asynchronous” is giving it a name that says what it isn’t
  – Meant as a framework for design in the large, encompassing
    • GALS
    • circuits with completion signals
      – delays to model circuits
      – data-dependent completion signals
  – This broad discipline of design got a good reception during the years when I worked for Burroughs (1972-1977)
The view from 1980 (~30 years ago)

(I joined the Caltech CS faculty in the spring 1977)

• The hope, partly realized, that VLSI would release the stranglehold of synchronous design.
  – Not stuck with building systems from synchronous LSI parts
• The first (1979) Caltech Conference on VLSI included a whole session on self-timed design
  – Not necessarily by popular request. I organized this conference.
• Chapter 7 in Mead & Conway (1980) got quite a few people interested in self-timed and asynchronous design
  – and aware of synchronization failure, …
• The “VLSI Architecture” movement
  – “Microelectronics and Computer Science” in Scientific American, 1977 (Sutherland and Mead)
Recent History

• In my period at Caltech, 1977-1994, I did research on concurrent architectures (including programming) and VLSI Design
  – Multicomputers
    • Message-passing concurrent computers
      – People today refer to this style of multicomputer or cluster as using “asynchronous” message-passing.
  – Practical self-timed VLSI:
    • My students and I designed routing and communication chips (the low-hanging fruit)
    • We left the hard designs, such as asynchronous processors, to others
• During the past 15 years, 1994-2009, I’ve been working at Myricom, an innovative computer-networking company
  – Yes, a lot of our chips use asynchronous techniques
An example of a Myri-10G switch
Thank you

More questions?