Self-timed and Asynchronous Design

Functions of clock in synchronous design
1) Acts as completion signal
2) Ensures the correct ordering of events

Truly asynchronous design
1) Completion is ensured by careful timing analysis
2) Ordering of events is implicit in logic

Self-timed design
1) Completion ensured by completion signal
2) Ordering imposed by handshaking protocol
### Synchronous Datapath

In

CLK

[Schematic diagram of synchronous datapath with components R1, D, Q, Logic Block #1, and t_{pd,reg}, t_{pd1}, t_{pd2}, t_{pd3}]

### Self-Timed Pipelined Datapath

[Schematic diagram of self-timed pipelined datapath with components HS, Req, Ack, Start, Done, F1, F2, F3, and t_{pf1}, t_{pf2}, t_{pf3}]
Completion Signal Generation

Using Delay Element (e.g. in memories)

<table>
<thead>
<tr>
<th>B</th>
<th>B0</th>
<th>B1</th>
</tr>
</thead>
<tbody>
<tr>
<td>in transition (or reset)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>illegal</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Using Redundant Signal Encoding
Completion Signal in DCVSL

Self-Timed Adder

(a) Differential carry generation

(b) Completion signal
Hand-Shaking Protocol

(a) Sender-receiver configuration

(b) Timing diagram

Two Phase Handshake

Event Logic – The Muller-C Element

(a) Schematic

(b) Truth table

(c) Dynamic
2-Phase Handshake Protocol

Advantage: FAST - minimal # of signaling events (important for global interconnect)
Disadvantage: edge-sensitive, has state

Example: Self-timed FIFO

All 1s or 0s -> pipeline empty
Alternating 1s and 0s -> pipeline full
2-Phase Protocol

Example

From [Horowitz]
Example

4-Phase Handshake Protocol

Also known as RTZ

Slower, but unambiguous
4-Phase Handshake Protocol

Implementation using Muller-C elements

Self-Resetting Logic

Post-charge logic
Asynchronous-Synchronous Interface

Synchronizers and Arbiters

- **Arbiter**: Circuit to decide which of 2 events occurred first
- **Synchronizer**: Arbiter with clock $\phi$ as one of the inputs
- **Problem**: Circuit HAS to make a decision in limited time - which decision is not important
- **Caveat**: It is impossible to ensure correct operation
- **But**, we can decrease the error probability at the expense of delay
A Simple Synchronizer

- Data sampled on rising edge of the clock
- Latch will eventually resolve the signal value, but ... this might take infinite time!

Synchronizer: Output Trajectories

Single-pole model for a flip-flop

\[ v(t) = V_{MS}^+ (v(0) - V_{MS}) e^{t/\tau} \]
Mean Time to Failure

\[ N_{\text{sync}}(0) = \frac{P_{\text{init}}}{T_{\phi}} = \left( \frac{V_{\text{IH}} - V_{\text{IL}}}{V_{\text{swing}}} \right) \frac{t_{r}}{T_{\text{signal}}} \]

\[ N_{\text{sync}}(T) = \frac{P_{\text{init}} e^{-T/t_{r}}}{T_{\phi}} - \left( \frac{V_{\text{IH}} - V_{\text{IL}}}{V_{\text{swing}}} \right) e^{-T/t_{r}} \frac{t_{r}}{T_{\text{signal}} T_{\phi}} \]

Example

\[ T_{f} = 10 \text{ nsec} = T \]
\[ T_{\text{signal}} = 50 \text{ nsec} \]
\[ t_{r} = 1 \text{ nsec} \]
\[ t = 310 \text{ psec} \]
\[ V_{\text{IH}} - V_{\text{IL}} = 1 \text{ V} \quad (V_{\text{DD}} = 5 \text{ V}) \]

\[ N(T) = 3.9 \times 10^{-9} \text{ errors/sec} \]
\[ \text{MTF (T)} = 2.6 \times 10^{8} \text{ sec} = 8.3 \text{ years} \]
\[ \text{MTF (0)} = 2.5 \mu \text{sec} \]
Influence of Noise

Initial Distribution

Uniform distribution around VM

Low amplitude noise does not influence synchronization behavior

Typical Synchronizers

2 phase clocking circuit

Using delay line
Cascaded Synchronizers Reduce MTF

\[ \text{In} \rightarrow \text{Sync} \rightarrow \text{Sync} \rightarrow \text{Sync} \rightarrow \text{Out} \]

\( \phi \)

Arbiters

(a) Schematic symbol

(b) Implementation

(c) Timing diagram
PLL-Based Synchronization

\[ f_{\text{system}} = N \times f_{\text{crystal}} \]

Crystal Oscillator

Data
reference clock

PLL

Divider

Digital System

Chip 1

PLL

Clock Distribution

Goal: Minimization of uncertainty

- Clock skew (spatial uncertainty)
  - Systematic
- Clock jitter (temporal uncertainty)
  - Random cycle-to-cycle changes
Reading

- Chapter 13, (Chandrakasan et al), Clock Distribution by Bailey
- Chapter 12, (Chandrakasan et al), PLLs and DLLs by Maneatis
- Chapter 10, Rabaey et al.

Clock Distribution

- Tree
  Common, e.g. IBM S/390

- Clock grid
  - DEC Alpha

- Length-matched Serpentines
  - Intel P6
Clock Distribution

H-Tree Network
Observe: Only Relative Skew is Important

Example:
PowerPC 603
Gerosa, JSSC 12/94

Clock Network with Distributed Buffering

Local Area
secondary clock drivers

Reduces absolute delay, and makes Power-Down easier
Sensitive to variations in Buffer Delay
**Predriver**

- Binary tree
- H-tree
- X-tree
- Arbitrary matched tree

**Example IBM S/390**

- Clock skew

Webb, JSSC 11/97
Clock Tree Delays

Impact of clock network sizing
Impact of clock network sizing

Final Stage: Tree vs. Grid

RC-matched Tree

Grid

**IR Emission Images**

Central buffer

Sector buffers

Clock repeaters

Local clocks

Sanda, ISSCC’99

---

**DEC Alpha Evolution**

Clock driver placements

21064

21164

21164

Gronowski, JSSC 5/98
Clock Skews

Example: DEC Alpha 21164
Clock Skew in Alpha Processor

Hybrid Grid

DEC Alpha 21264
Bailey JSSC 11/98
Alpha 21264

Global clock

Major clock grids

<table>
<thead>
<tr>
<th>Major Clock</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCLK</td>
<td>bus interface unit</td>
</tr>
<tr>
<td>ICLK</td>
<td>integer issue and execution</td>
</tr>
<tr>
<td>FCLK</td>
<td>floating point issue and</td>
</tr>
<tr>
<td></td>
<td>execution units</td>
</tr>
<tr>
<td>JCLK</td>
<td>instruction fetch and branch</td>
</tr>
<tr>
<td></td>
<td>prediction unit</td>
</tr>
<tr>
<td>NCLK</td>
<td>load/store unit</td>
</tr>
<tr>
<td>PCLK</td>
<td>padding</td>
</tr>
</tbody>
</table>
Data-Dependent Gate Loading

Multi-GHz Clock Networks

Phillip Restle, IBM Research
IEEE SSCTC Workshop on Design for Multi-GigaHertz Processors,
San Francisco, Feb. 7, 2000

http://www.research.ibm.com/people/r/restle/MSGHz.html
http://www.research.ibm.com/people/r/restle/Animations/DAC01top.html
Clock Generation

Delay-Locked Loop (Delay Line Based)

Phase-Locked Loop (VCO-Based)

Phase-Locked Loop Based Clock Generator

Acts also as Clock Multiplier
Loop Components

- **Phase Comparator**
  - Produces UP/DN pulses corresponding to phase difference

- **Charge Pump**
  - Sources/sinks current for duration of UP/DN pulses

- **Loop Filter**
  - Integrates current to produce control voltage

- **Voltage-Controlled Delay Line**
  - Changes delay proportionally to voltage

- **Voltage-Controlled Oscillator**
  - Generates frequency proportional to control voltage

PLL Jitter
DLL Locking

Two clock spines, two DLLs, and a PD that controls them

Clock Deskewing

Two clock spines, two DLLs, and a PD that controls them
Clock Ring

Clocks routed in parallel, opposite directions
LCG aligns to the middle

Synchronous Distributed Oscillators

Mizuno, ISSCC’98
Distributed PLLs

Chip Boundary

Tile Boundary

Phase Detector

Loop Filter & VCO

Gutnik, ISSCC'2000

Intel Itanium™

Global Clock Distribution

- Balanced H-tree routed in M5 and M6
- Lateral shielding
- Distributes both main and reference clock
- Optimized to account for inductive effects

Rusu, ISSCC'2000
Regional Clock Distribution

- Distributed array of deskew buffers to reduce process related skew
  - 8 deskew clusters each holding up to 4 buffers
- Regional clock grids driven by modular Regional Clock Drivers
  - M4-M5 grid tailored for the clock load density of the underlying block
  - Full support for scan and clock gating

[Diagram showing distribution and labels]

DSK = Cluster of 4 deskew buffers
CDC = Central Deskew Controller