



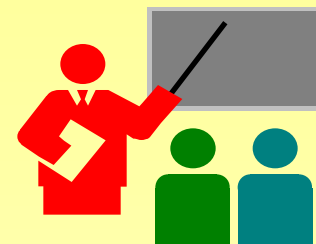
Circuitos Assíncronos

ou “O Problema de Sincronização em Sistemas Digitais”

Ney Laert Vilar Calazans
(ney.calazans@pucrs.br)

29 de Abril de 2013

- 0 - Introdução e Motivação
- 1 - Representações As(síncronas)
- 2 - Síncronos versus Assíncronos
- 3 - Histórico, Ferramentas, etc.
- 4 - Interfaces Assíncronas
- 5 - Um Exemplo de Implementação
- 6 - Conclusões



0 – Introdução e Motivação

O International Technology Roadmap for Semiconductors (ITRS) diz:

“By the end of the decade, SOCs using 50-nm transistors operating below one volt, will grow to 4 billion transistors running at 10GHz”,

ou seja

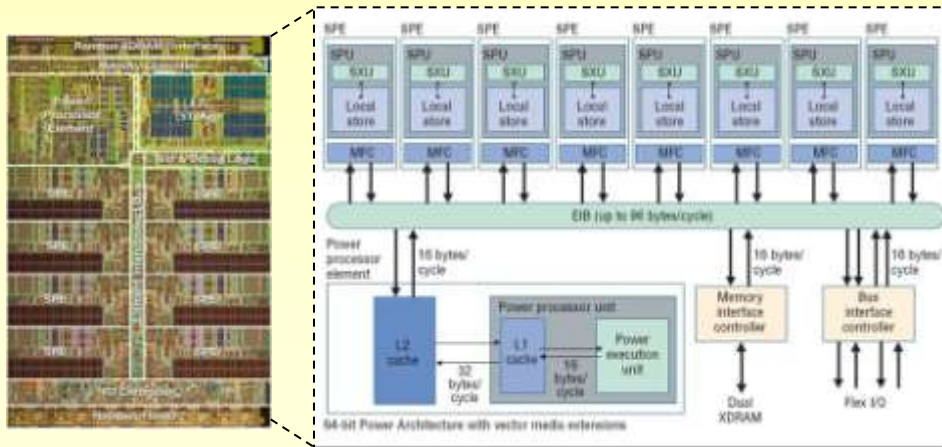
“Ao final desta década existirão chips com transistores de 50-nm, operando com alimentação abaixo de 1 Volt, e contendo 4 bilhões de transistores operando a 10GHz”

0 – Motivação - Circuitos e Estilos de Projeto

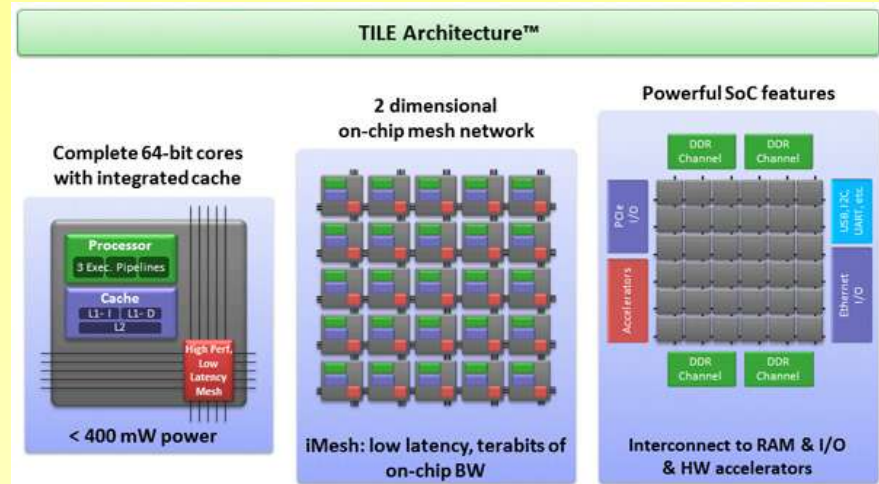
Produtos Comerciais

- Exemplos:
 - » CELL – IBM-Toshiba-Sony (8+1 Núcleos)
 - » Tile-Gx72 – Tileria (72 Núcleos)
 - » Intel – (experimental 80 Núcleos PF)
 - » Dual e Quad Core
- MPSoCs Homogêneos
- Tile- Baseados em NoCs, projeto GALS Mesócrono

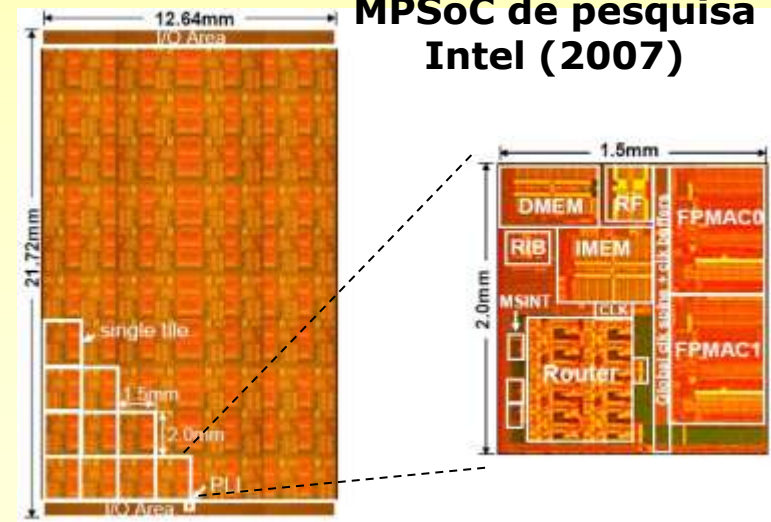
Cell (IBM-Toshiba-Sony) p/ PlayStation3 (2006)

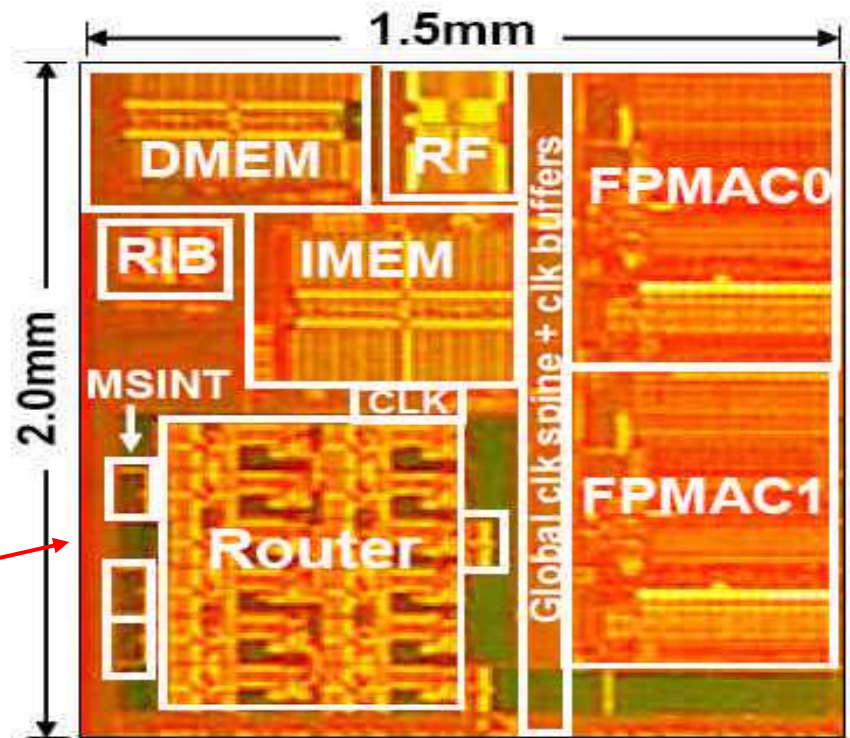
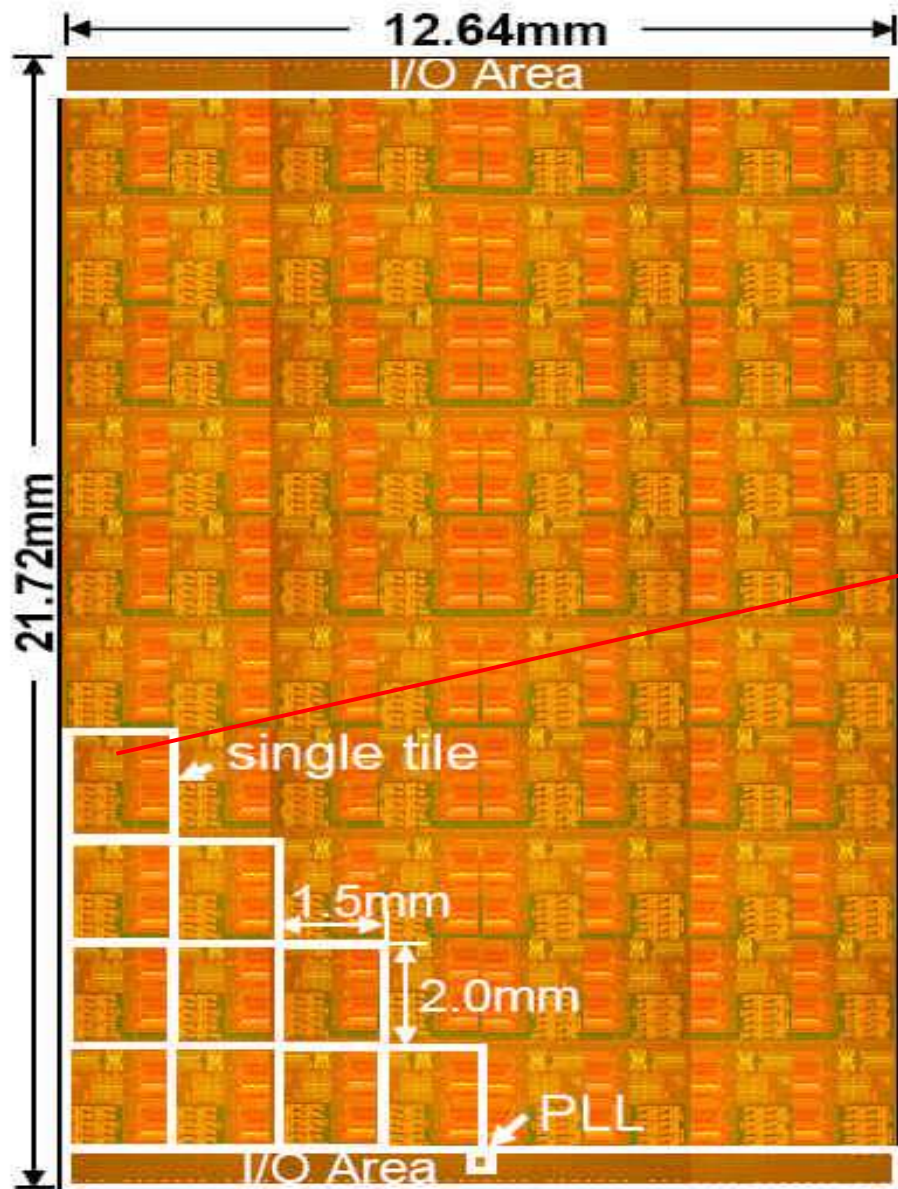


TILE Architecture™



MPSoC de pesquisa Intel (2007)



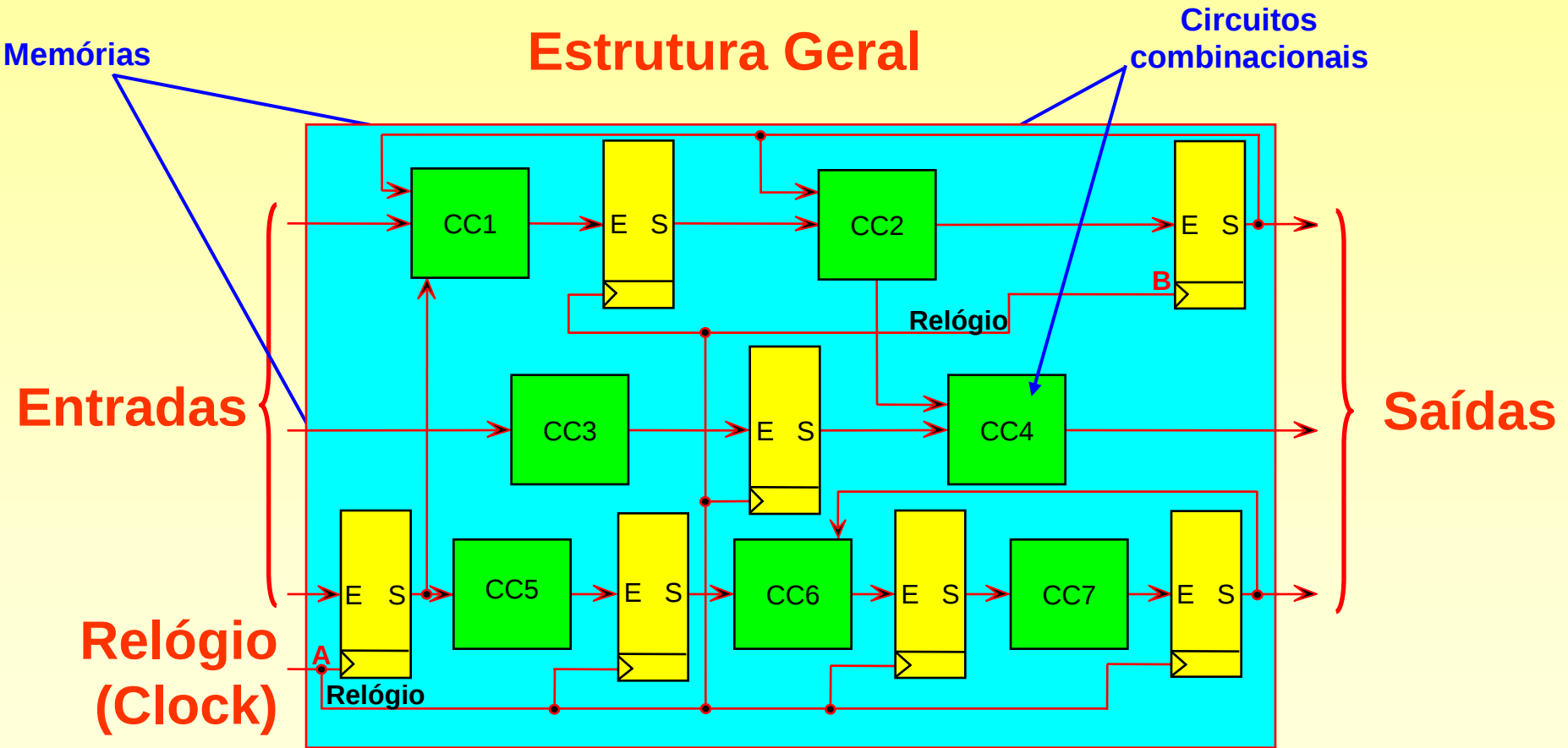


Tecnologia	CMOS, 65nm
Interconexões	1 poly, 8 metal (Cobre)
Transistores	100 Milhões
Área do Chip	275 mm ²
Área do <i>Tijolo</i>	3 mm ²
Encapsulamento	LGA de 1248 pinos, 14 camadas, 343 pinos de E/S
Projeto	GALS – mesócrono

Vangal et al. ISSCC 2007

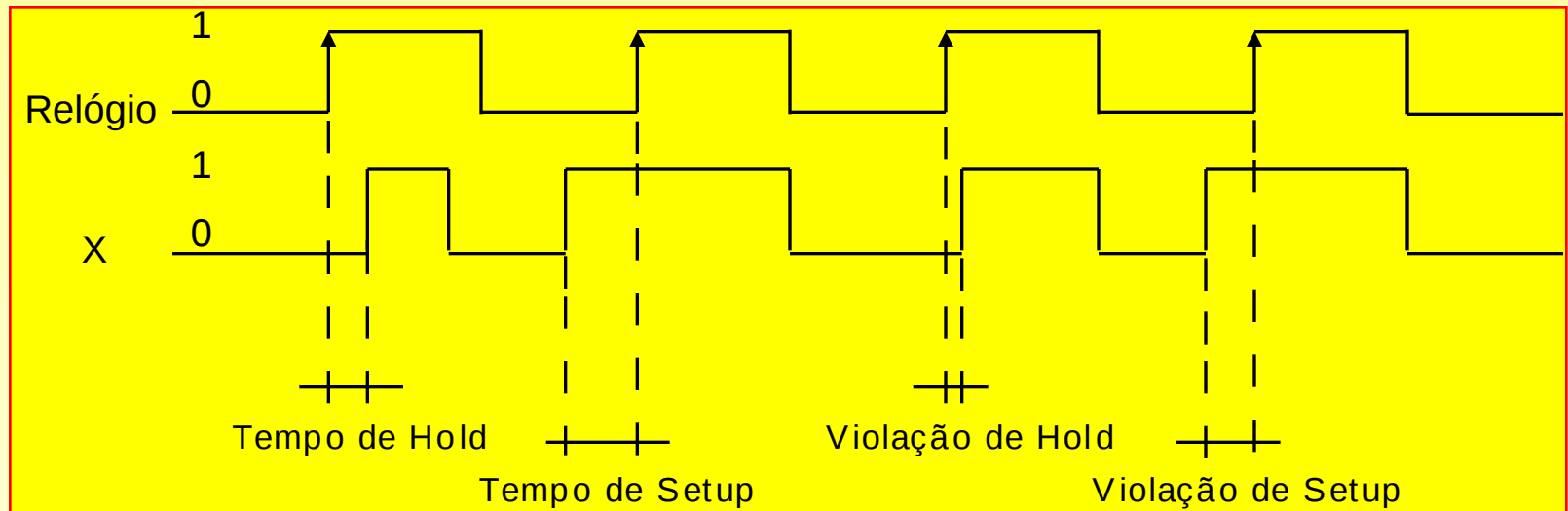
0 - Motivação – Sistemas Não-síncronos

→ Afinal, o que é um circuito síncrono?



0 - Motivação – Sistemas Não-síncronos

Pressupostos Operacionais



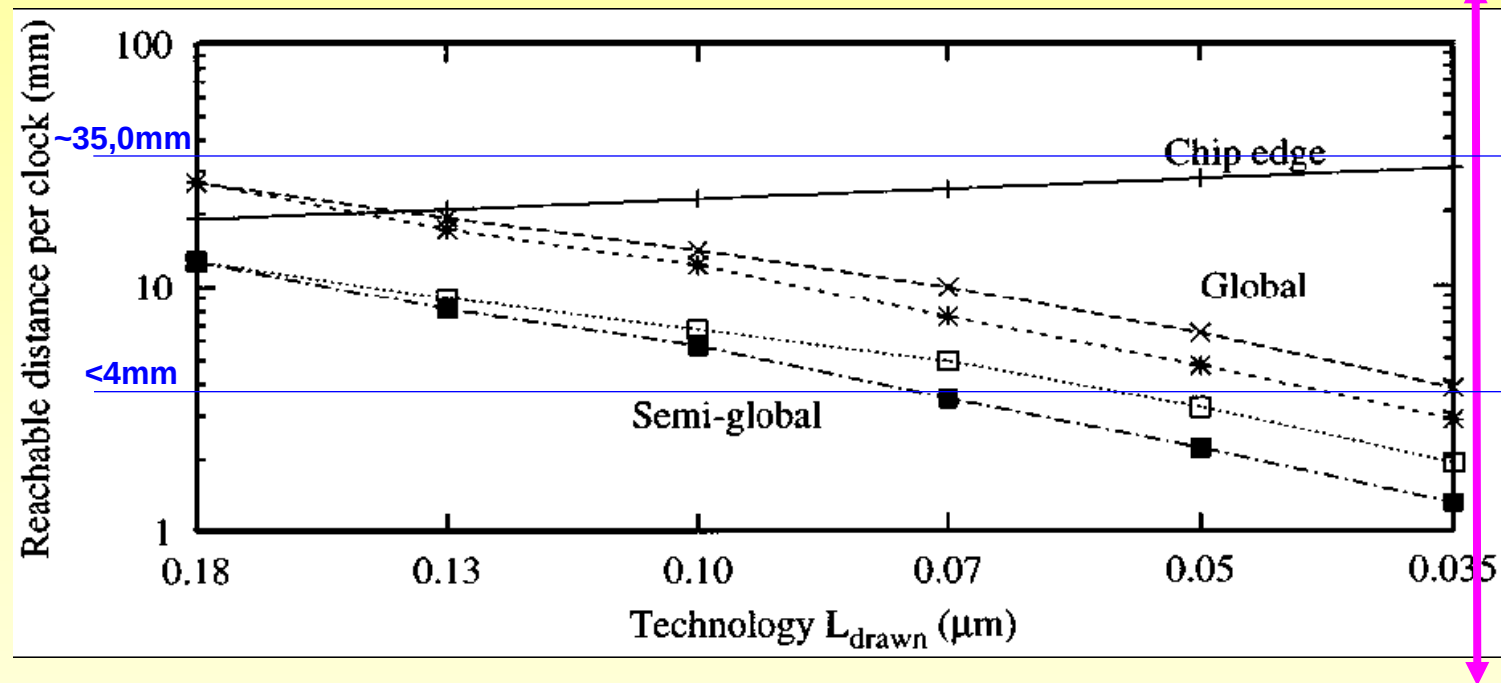
Tempo de Setup – tempo mínimo para que entradas estabilizem antes da transição do sinal para relógio

Tempo de Hold – tempo mínimo que entradas devem ficar estáveis após uma transição do sinal para relógio

0 - Motivação – Sistemas Não-síncronos

Nas tecnologias modernas, temos (Gráfico de Ho et al. 2001, artigo “The Future of Wires” na revista Proceedings of the IEEE):

Estado da Arte Comercial hoje: 28nm e 22nm

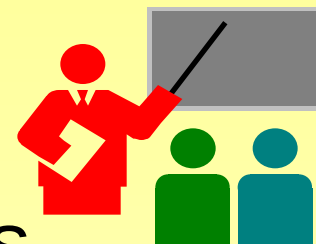


Ou seja, gasta-se hoje mais de **5 ciclos** para o sinal de relógio atravessar um chip do estado da arte! Logo, como usar projeto síncrono? Evitar este problema é **muito caro!!** Além disso, pode-se gastar **60-70%** da energia consumida pelo chip só com o relógio!

0 - Motivação – Sistemas Não-síncronos

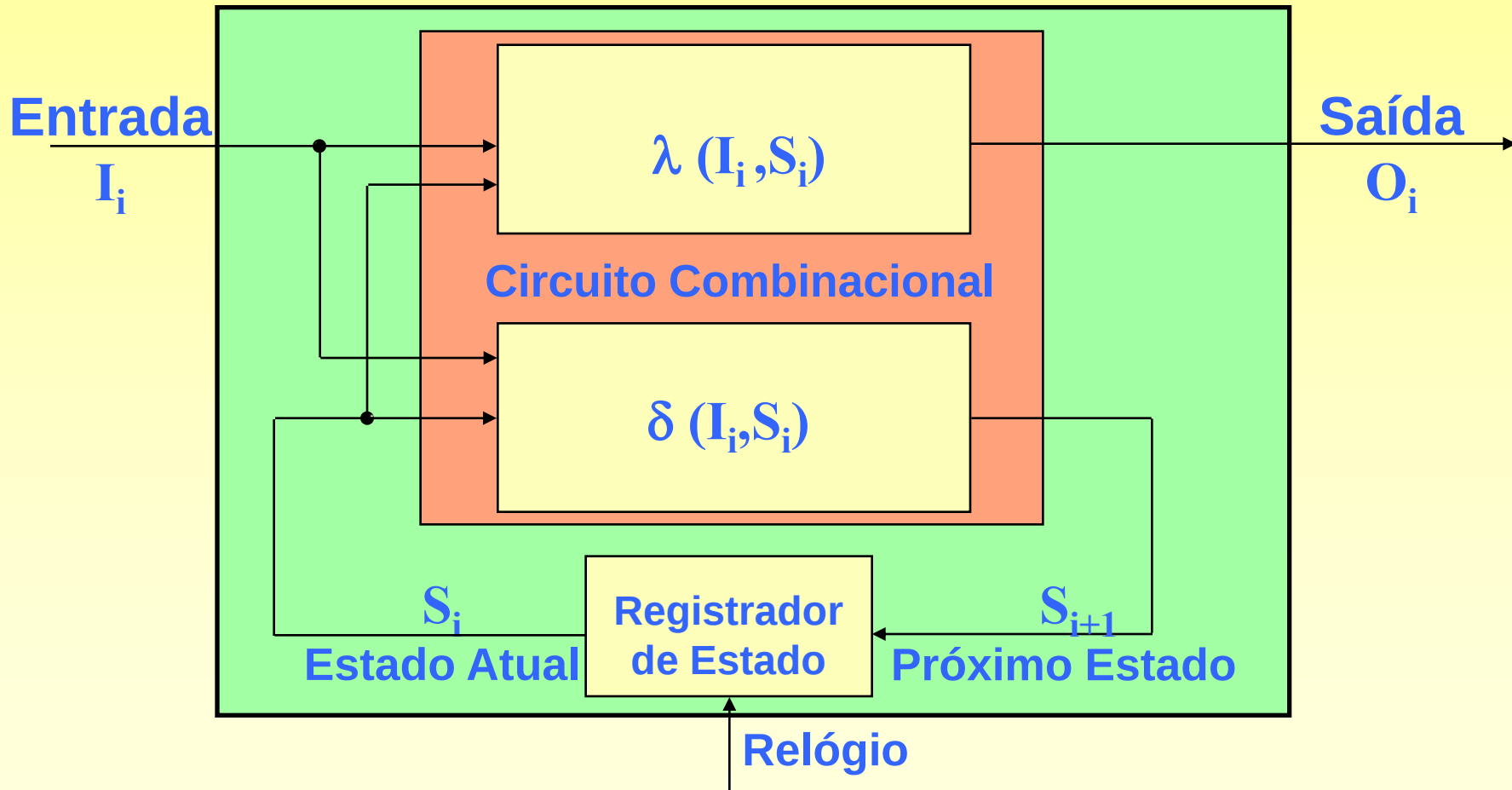
- Vale a pena investigar sistemas não-síncronos
- É útil pensar considerar interfaces entre diferentes módulos de um chip → elas podem limitar o desempenho (ou não)
- Projeto síncrono está chegando ao limite, quais alternativas existem?
- Que suporte projeto não síncrono possui ou exige? Este suporte existe? → Síncronos têm dezenas de anos de desenvolvimento de EDA (Electronic Design Automation) por trás

- ✓ 0 - Introdução e Motivação
- 1 - As(síncronos) - Representações
- 2 - Síncronos versus Assíncronos
- 3 - Histórico, Ferramentas, etc.
- 4 - Interfaces Assíncronas
- 5 - Um Exemplo de Implementação
- 6 - Conclusões



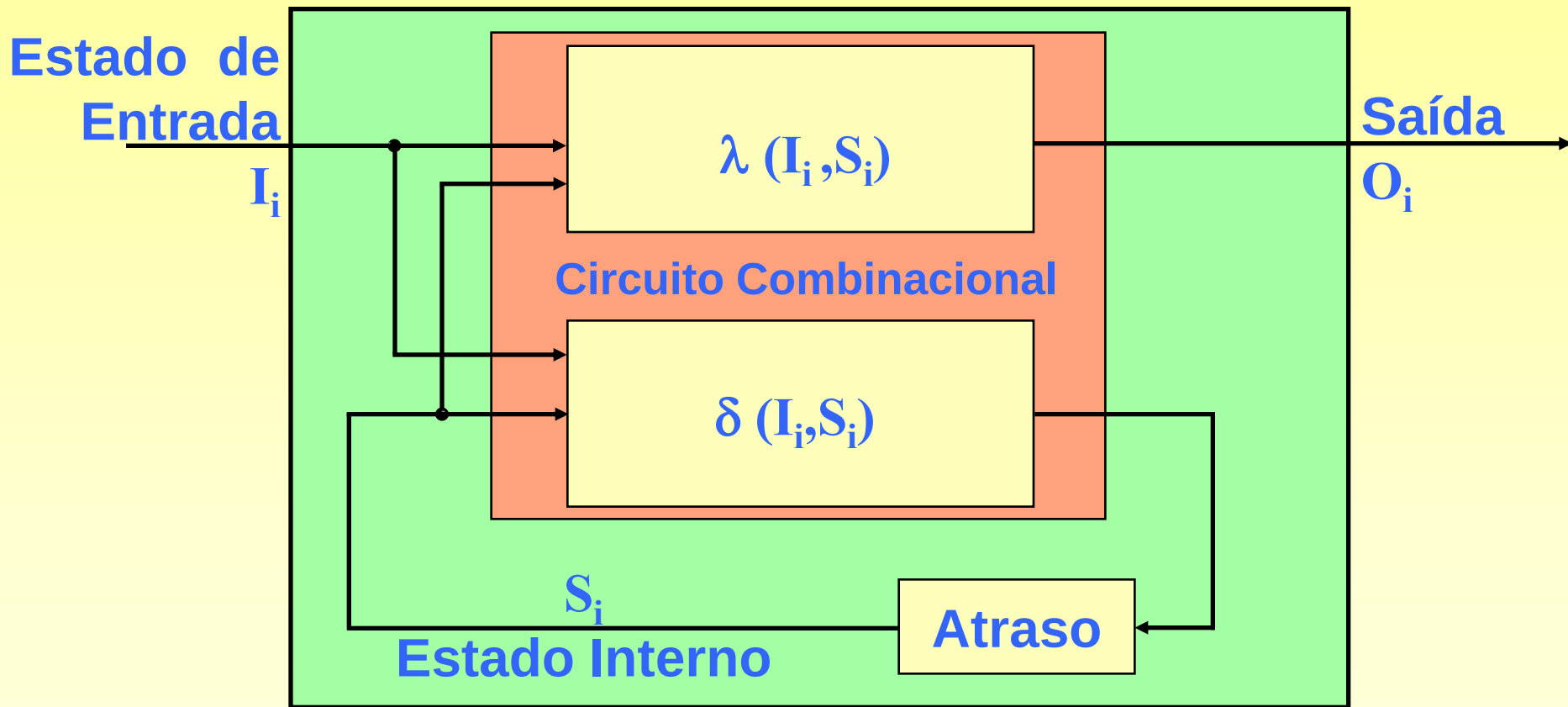
1 - As(síncronos) - Representações

Modelo Estrutural para FSMs Síncronas



1 - As(síncronos) - Representações

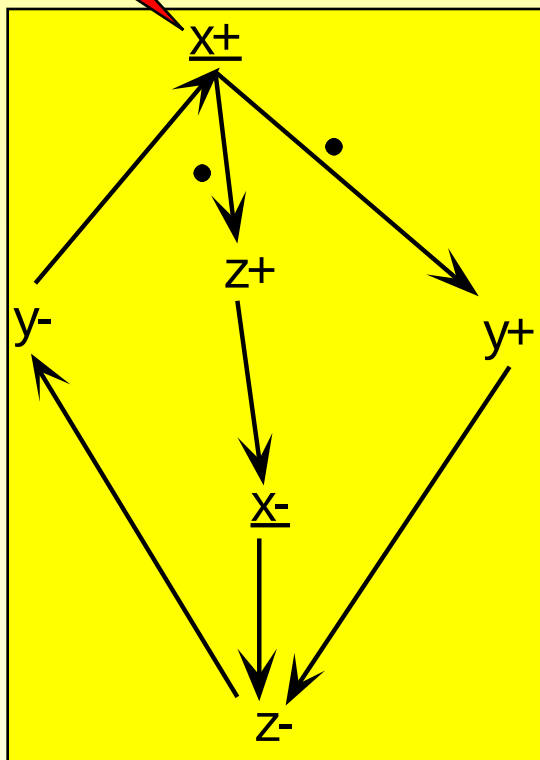
Modelo Estrutural para FSMs Assíncronas



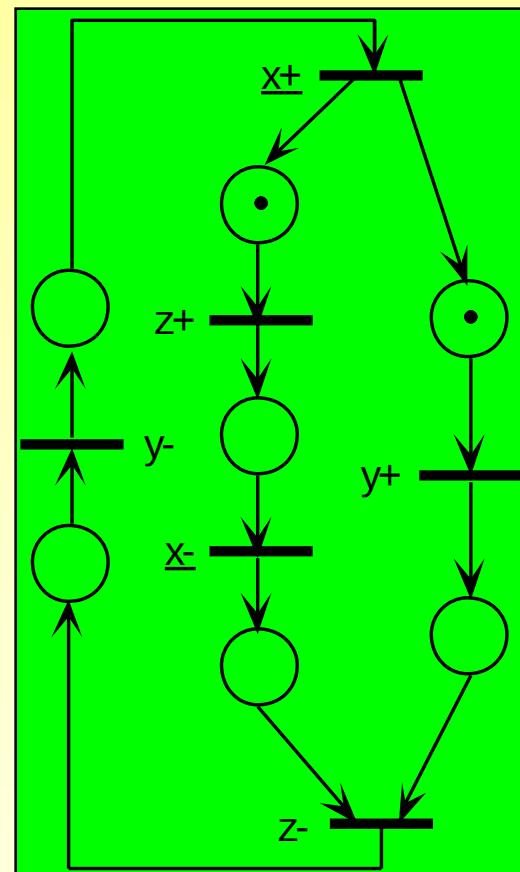
1 - As(síncronos) – Exemplo de Representação

Entrada Primária!

ASTG



Rede de Petri Equivalente



- ✓ 0 – Introdução e Motivação
- ✓ 1 - As(síncronos) - Representações
- 2 - Síncronos versus Assíncronos
- 3 - Histórico, Ferramentas, etc.
- 4 - Interfaces Assíncronas
- 5 - Um Exemplo de Implementação
- 6 - Conclusões

2 - Síncronos versus Não-síncronos

- Diferença mais fundamental - modelo temporal
 - ◆ Síncronos - sinais discretos/ tempo discreto
 - ◆ Assíncronos - sinais discretos/ tempo contínuo
- Estilo síncrono - uma família coerente de métodos
- Diversos estilos assíncronos
 - ◆ dependem das restrições adicionais impostas

2 - Síncronos versus Não-síncronos

Abordagem a adotar

- Eliminação de tempo discreto e consequências sobre métodos de projeto
- Principais considerações de projeto derivadas da eliminação

2 - Síncronos versus Não-síncronos

Vantagens do Projeto Síncrono

- 1 - Abstração de atrasos durante o projeto
- 2 - Facilidade de decomposição
- 3 - Abstração de eventos espúrios (devido à propagação diferenciada)
- 4 - Total eliminação de corridas
- 5 - Manipulações Booleanas e algébricas diretas

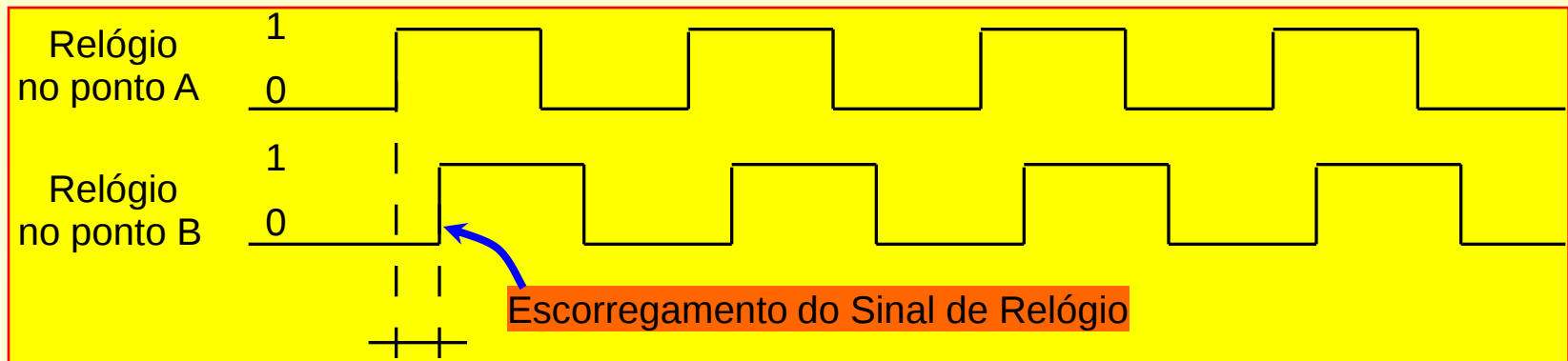
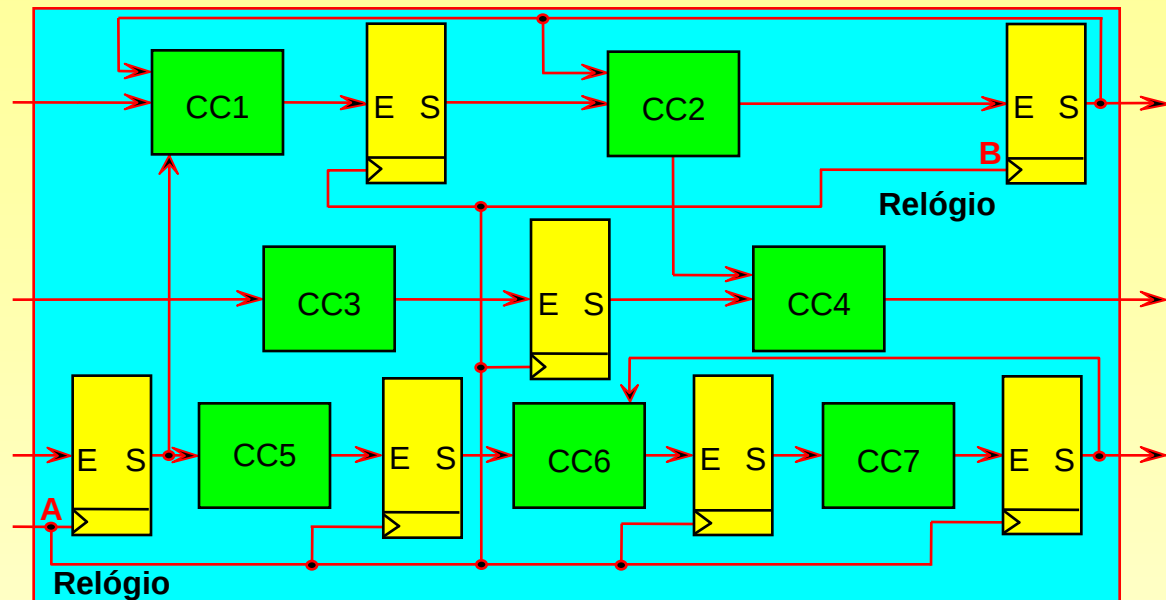
2 - Síncronos versus Não-síncronos

Inconvenientes do Projeto Síncrono

- 1 - Escorregamento de relógio
- 2 - Potencial para desperdiçar energia
- 3 - Desempenho de pior caso
- 4 - Baixa propensão à migração tecnológica
- 5 - Inadaptabilidade à variações de propriedades físicas como tensão de alimentação
- 6 - Tratamento do fenômeno de meta-estabilidade

2 - Síncronos versus Não-síncronos

Conceito de
escorregamento
do sinal de
relógio



2 - Síncronos versus Não-síncronos

- Conjunto (incompleto) de modelos para tratamento de problemas de temporização
- Classes de modelos:
 - ◆ representação de informação
 - ◆ ambiente
 - ◆ fenômenos temporais
- Estilo assíncrono de projeto determinado pela escolha de subconjunto de modelos

2 - Síncronos versus Não-síncronos

→ Representação de informação

◆ Modelos p/ codificação física de informação

- ◆ Tradicional – 1 bit=1 fio

- ◆ Insensível a atrasos (DI) – códigos DI

◆ Assinalamento da informação

- ◆ assinalamento por nível ou por transição

◆ Protocolos de comunicação

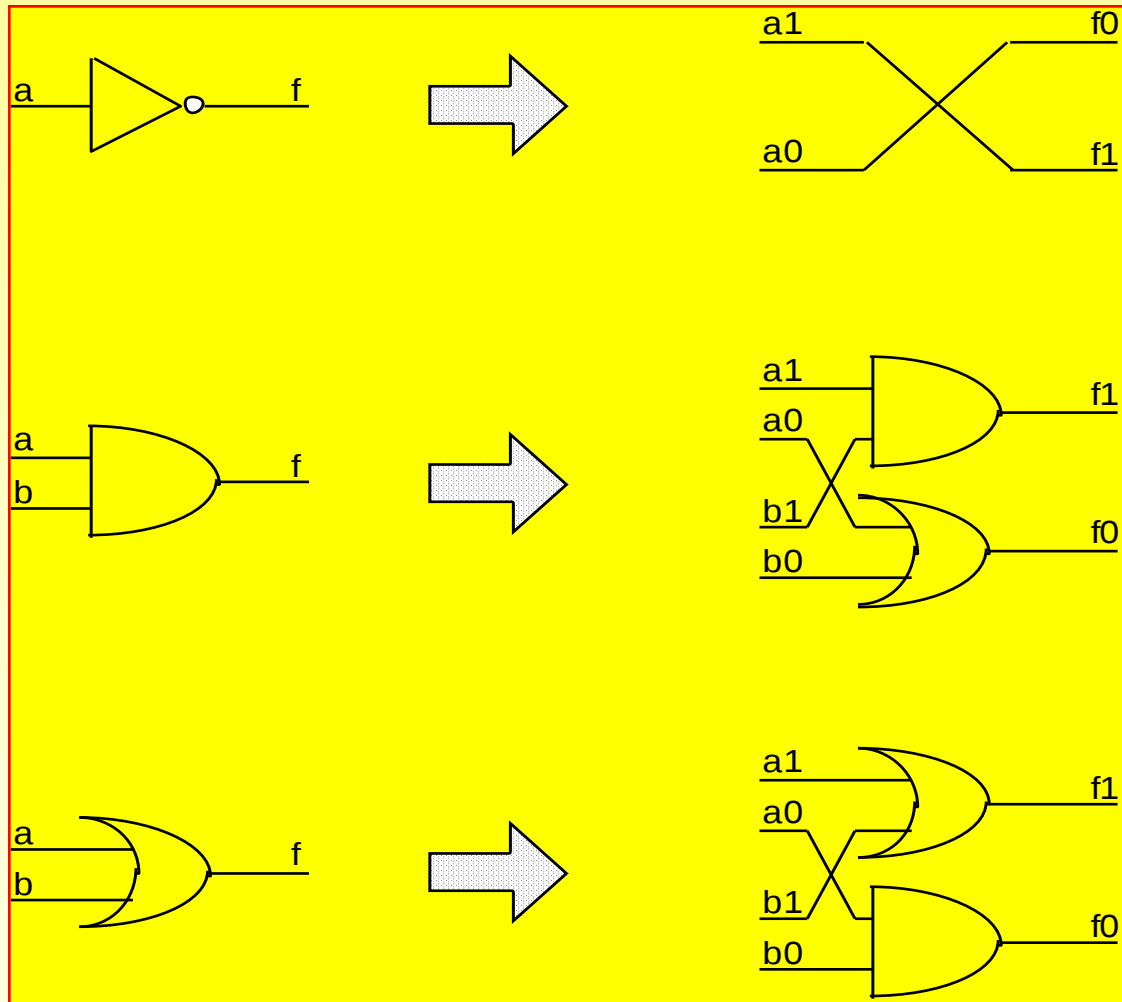
- ◆ duas fases ou quatro fases

2 - Síncronos versus Não-síncronos

**Conceito de
codificação em
trilha dupla – 1
exemplo de código
DI (há diversos)**

a0a1	Interpretação
00	Sem informação
01	'0'
10	'1'
11	Inválido

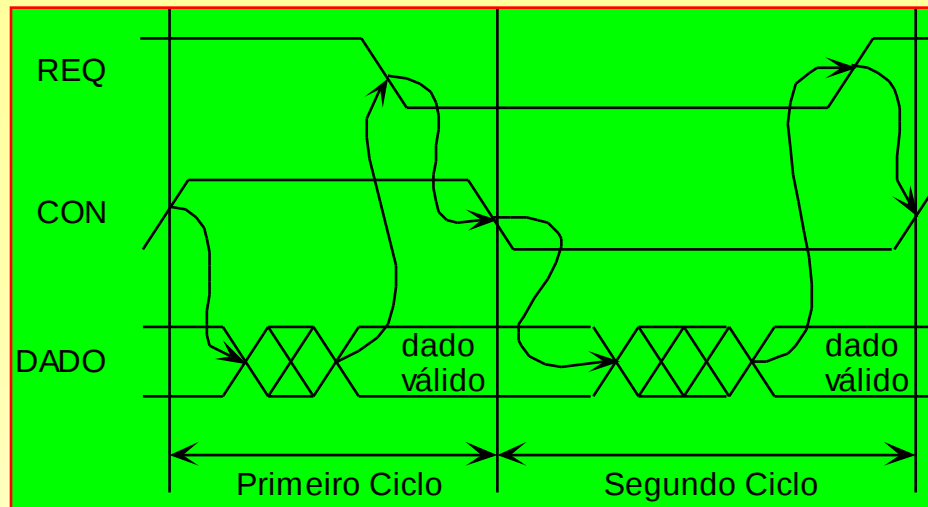
Trilha Simples



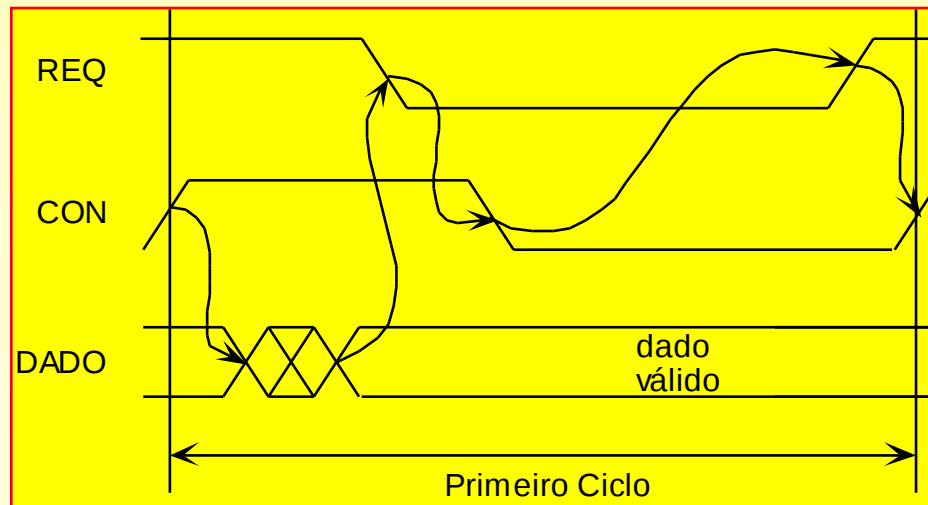
Trilha Dupla

2 - Síncronos versus Não-síncronos

Protocolos de Comunicação



Duas Fases

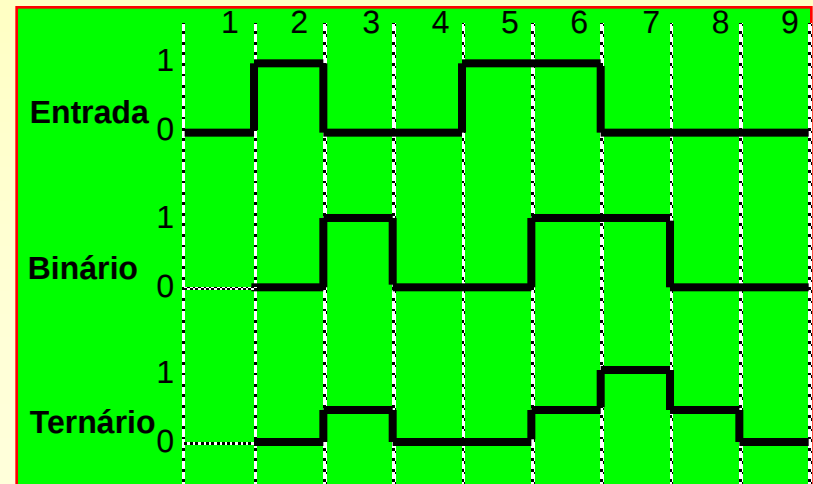
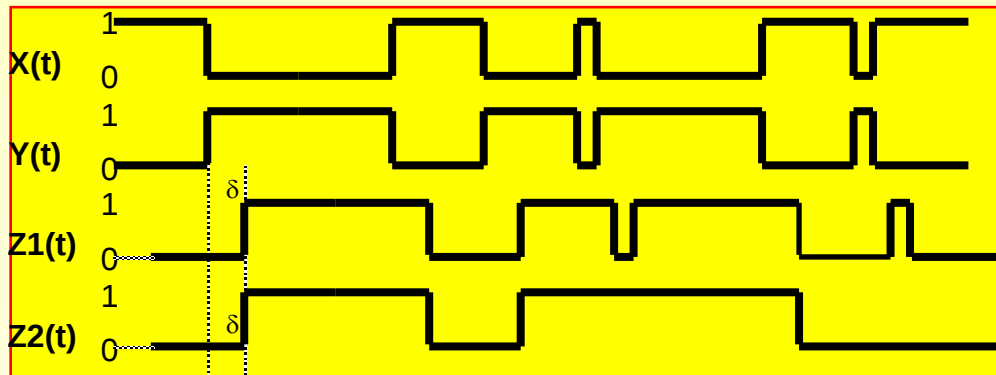


Quatro Fases

2 - Síncronos versus Não-síncronos

- Modelagem do comportamento do ambiente é muito importante!
 - ◆ mudanças simultâneas, não-simultâneas, inválidas
- Modelagem de Atrasos é muito importante!
 - ◆ atraso ideal ou atraso inercial

Atrasos de Componentes

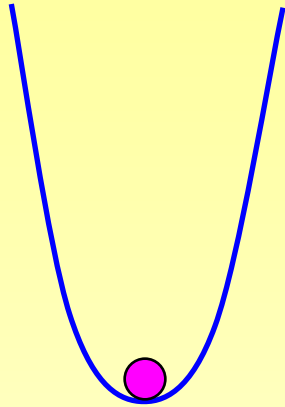


2 - Síncronos versus Não-síncronos

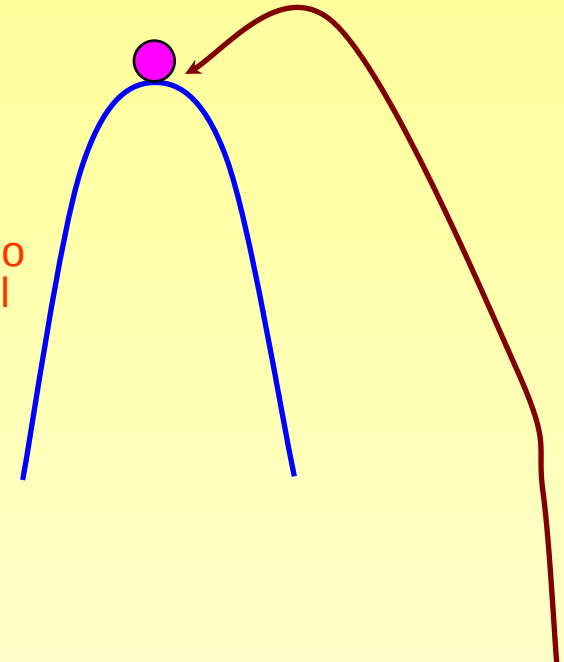
Metaestabilidade

Analogia Mecânica

Equilíbrio
Estável

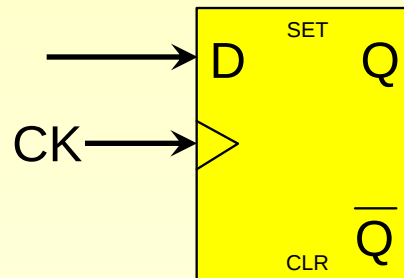


Equilíbrio
Instável



→ Eletricamente, o mesmo pode ocorrer:

Supor que uma
transição ocorre em D
ao mesmo tempo que
uma transição do
sinal de relógio!



O que pode acontecer
na saída Q?

2 - Síncronos versus Não-síncronos

Metaestabilidade

→ Eletricamente:

- ◆ Se a entrada D transicionar ao mesmo tempo que o sinal que comanda seu armazenamento em um elemento de memória, coisas *horríveis* podem acontecer:
 - ◆ O valor finalmente armazenado pode não ser o desejado (**Ruim**)
 - ◆ O valor finalmente armazenado (certo ou errado) pode sofrer uma demora arbitrária para aparecer (**PIOR**)
 - ◆ O valor pode ser armazenado ora de forma correta, ora incorreta (**MUITO RUIM**)
 - ◆ A saída do circuito de armazenamento pode ficar em valor lógico inválido (nem 0, nem 1) por um tempo arbitrário! (**CATASTRÓFICO**)

→ Metaestabilidade deve ser evitada a todo custo!

2 - Síncronos versus Não-síncronos

Uma Taxonomia de Assíncronos (Sparsø)

- *speed-independent* – opera corretamente assumindo atrasos positivos, limitados mas desconhecidos em portas e atraso ideal, nulo em fios
- *delay-insensitive* - opera corretamente assumindo atrasos positivos, limitados mas desconhecidos em portas e fios
- *quasi-delay-insensitive (QDI)* – similar a delay-insensitive mas assume que algumas derivações (*forks*) são isocrônicas
- *self-timed* - operação correta baseia-se em pressupostos de temporização e/ou de engenharia + elaborados

- ✓ 0 - Motivação
- ✓ 1 - As(síncronos) - Representações
- ✓ 2 - Síncronos versus Não-síncronos
- 3 - Histórico, Ferramentas, etc.
- 4 - Interfaces Assíncronas
- 5 - Um Exemplo de Implementação
- 6 - Conclusões

3 – Histórico, Ferramentas, etc.

→ Não-síncronos não são novidade:

- ◆ Huffman (1954) e Miller (1963) → várias contribuições ao projeto assíncrono.
- ◆ Unger → uso de AFTs em 1969.
- ◆ Anos 1970-1990 → abandono de assíncronos, pois
 - ◆ Projeto síncrono → prático para circuitos complexos, muito mais difíceis de projetar, implementar e testar sem o clock
 - ◆ CAD evoluiu muito, automatizando muitas tarefas (de projeto síncrono)
- ◆ Assim → assíncronos usados apenas quando inevitável → e.g. interfaces externas entre chips usando clocks diferentes (processador-periféricos)

→ interesse em assíncronos ressurgindo → síncronos
→ beco sem saída de complexidade de projeto

3 – Histórico, Ferramentas, etc.

→ Onde há recursos para assíncronos?

◆ **Livros antigos e alguns novos**

1. Unger e Huffman – antigos, com a teoria fundamental
2. Livro do Sparso – 2001, quase atual e muito bom
3. Livro do Myers – 2003, também muito bom e mais novo
4. Livro do Ney – não tão novo (1998), com intro (2 Caps)
5. Parte teórica de 2 e 4 disponíveis na Internet

◆ **Sites na Internet com muito material**

◆ <http://intranet.cs.man.ac.uk/apt/async> - Site na Universidade de Manchester com muito material para iniciantes e especialistas

- ☞ Artigos, ponteiros para livros, etc
- ☞ Ferramentas e ponteiros para ferramentas de CAD
- ☞ Listas de grupos de pesquisa e sites de conferências

◆ **Principal conferência específica annual - ASYNC**

3 – Histórico, Ferramentas, etc.

→ Alguns dos principais grupos de pesquisa em Assíncronos:

◆ **Universidade de Manchester (Inglaterra)**

1. Semente de processador hoje comercial, o AMULET
2. Ambiente de projeto semi-profissional – BALSA

◆ **California Institute of Technology (EUA)**

1. Primeiro processador assíncrono do mundo (1988)
2. Teoria básica por trás do BALSA (Alain Martin)

◆ **Politécnica de Grenoble (França)**

1. Ambiente de projeto virando comercial – TAST
2. Pesquisa em NoCs, FPGAs e ASICs (Marc Renaudin)
3. Atualmente, pesquisa migrou para empresa de CAD

◆ **Ver mais no repositório de Manchester**

1. Politécnica da Catalunha, Berkeley, TU-Denmark, U-Tokyo, etc.

- ✓ 0 - Motivação
- ✓ 1 - As(síncronos) - Representações
- ✓ 2 - Síncronos versus Não-síncronos
- ✓ 3 - Histórico, Ferramentas, etc.
- ➔ 4 - Interfaces Assíncronas
- ➔ 5 - Um Exemplo de Implementação
- ➔ 6 - Conclusões

4 - Interfaces Assíncronas

- Ao que tudo indica, migração de projeto síncrono para assíncrono → Gradual
- Cenário da Gradualidade
 - ◆ Síncrono → GALS e Dessincronizados
 - ◆ GALS e Circuitos Elásticos → Mistos: Elásticos Síncronos e módulos totalmente assíncronos selecionados
 - ◆ Mistos → Maior parte do sistema totalmente assíncrono
 - ◆ etc.

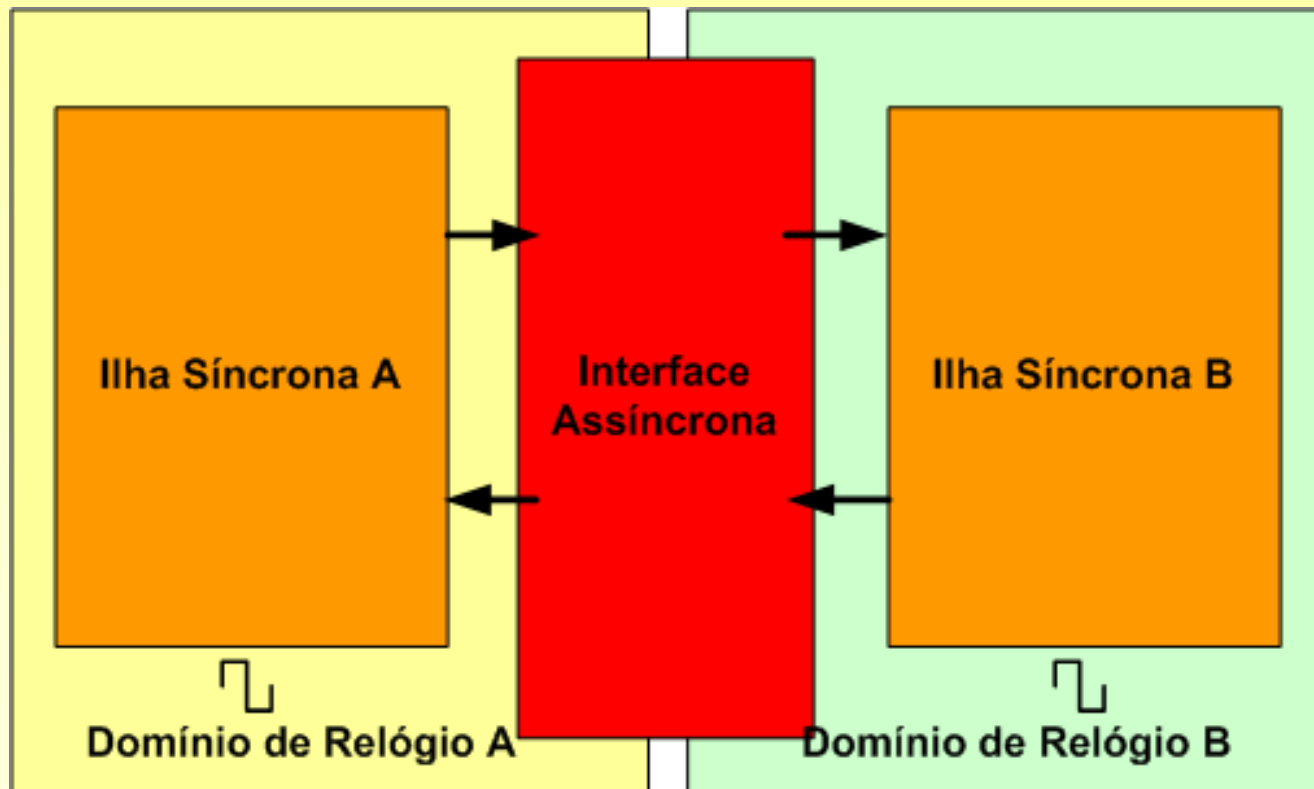
4 - Interfaces Assíncronas

- O que é **GALS** ?
- Sigla para Sistemas **G**lobalmente **A**ssíncronos e **L**ocalmente **S**íncronos
- Todos os elementos processadores são internamente síncronos
- A comunicação é assíncrona
- Interfaces de comunicação síncrono→assíncrono e assíncrono→síncrono
- Assim, estudar interfaces torna-se **fundamental**

4 - Interfaces Assíncronas

→ Sistemas GALS

- ◆ Computação → Ilhas síncronas
- ◆ Comunicação → Interfaces Assíncronas



4 - Interfaces Assíncronas

→ Tipos de interfaces de comunicação

- ◆ Sincronizadores
- ◆ Relógio Pausável
- ◆ FIFO Bissíncrona

→ Exemplo

- ◆ Comparação de área e desempenho
- ◆ Verificação da robustez
 - ◆ Simulação com temporização
 - ◆ Prototipação em FPGAs

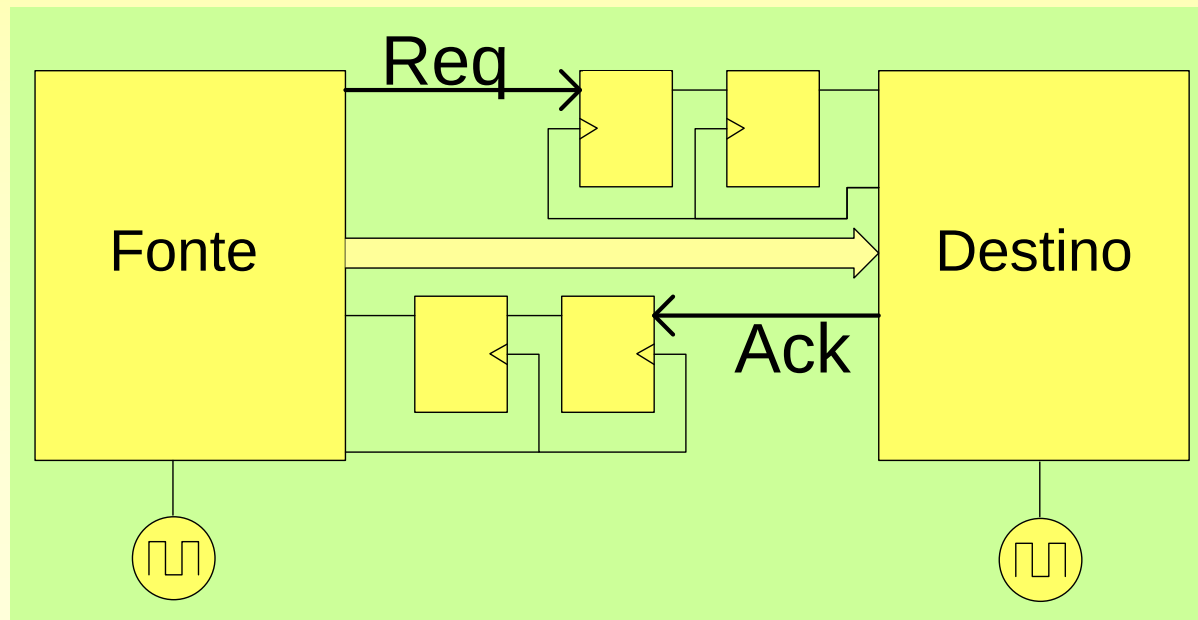
4 - Interfaces Assíncronas

- Circuito de teste - Produtor-consumidor com relógios independentes
 - ◆ Variar frequência de operação um dos módulos, enquanto a do outro mantida fixa
 - ◆ Resultados obtidos de simulação com temporização
 - ◆ Circuitos prototipados em FPGA Spartan 3 XC3S200

4 - Interfaces Assíncronas

→ Sincronizadores

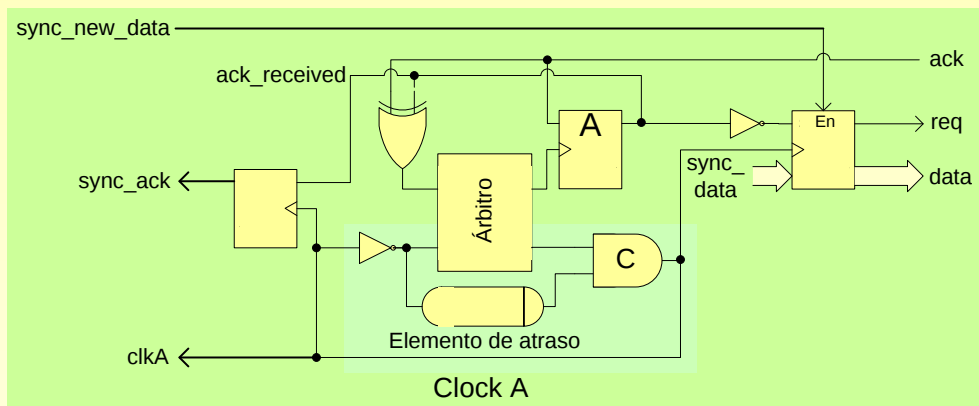
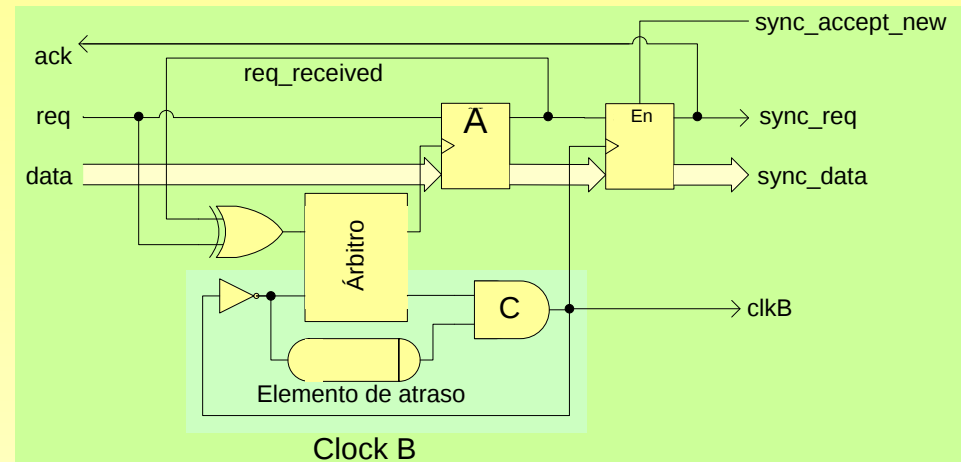
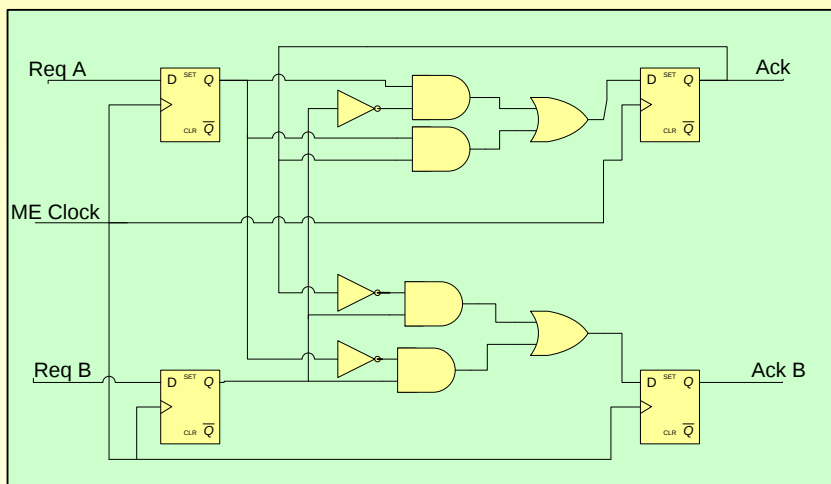
- ◆ Não eliminam metaestabilidade; Área Reduzida
- ◆ Baixo desempenho; Descrição em VHDL
- ◆ Handshake 2 fases



4 - Interfaces Assíncronas

→ Relógio Pausável

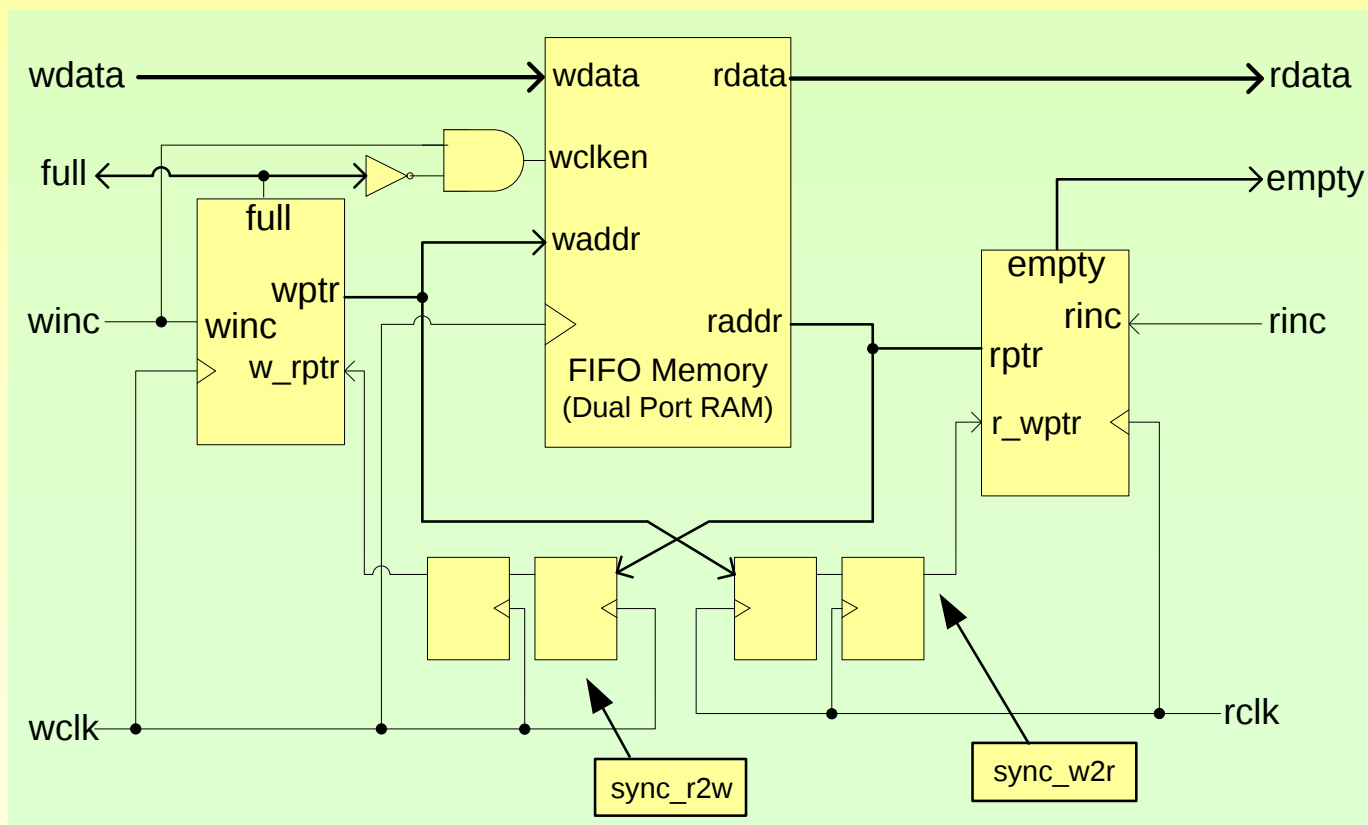
- ◆ Bundled Data e Handshake 2 fases
- ◆ Elimina totalmente o risco de metaestabilidade
- ◆ Gerador de relógio construído através de hard macros
- ◆ Problema: Latência do Árbitro



4 - Interfaces Assíncronas

→ Filas Bi-Síncronas

- ◆ Baseada em créditos; Transmite 1 dado a cada ciclo de relógio
- ◆ 2 domínios de relógio; Sincronizadores; Código Gray



4 - Interfaces Assíncronas

Comparação de Área das Implementações de Interfaces Assíncronas

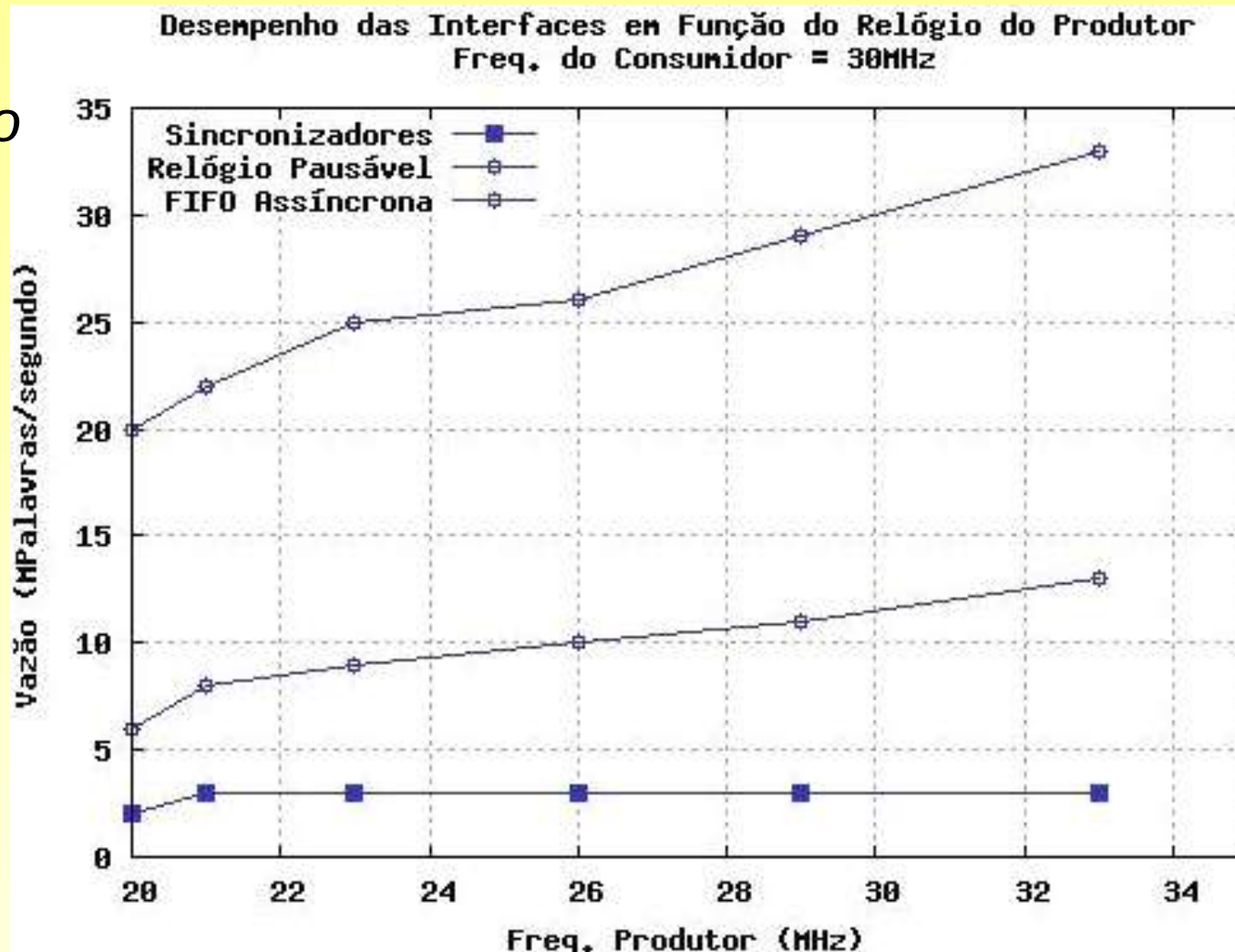
Abordagem	Flip Flops	LUTs	LUT RAMs	Hard Macros	Portas Equivalentes
Relógio Pausável	135	118	0	6	2096(*)
Sincronizadores	104	86	0	0	1656
Fila Bi-Síncrona	90	152	32(**)	0	3988

(*) Não inclui as *hard macros*

(**) LUTRAMs usadas como RAM de porta dupla

4 - Interfaces Assíncronas

- Interfaces fila
→ desempenho cresce com a frequência do produtor
- Relógio pausável → desempenho cresce menos
- Sincronizadores → desempenho constante



- ✓ 0 - Motivação
- ✓ 1 - As(síncronos) - Representações
- ✓ 2 - Síncronos versus Não-síncronos
- ✓ 3 - Histórico, Ferramentas, etc.
- ✓ 4 - Interfaces Assíncronas
- ➔ 5 - Um Exemplo de Implementação
- ➔ 6 - Conclusões

- ◆ Proposto por Nowick e Singh

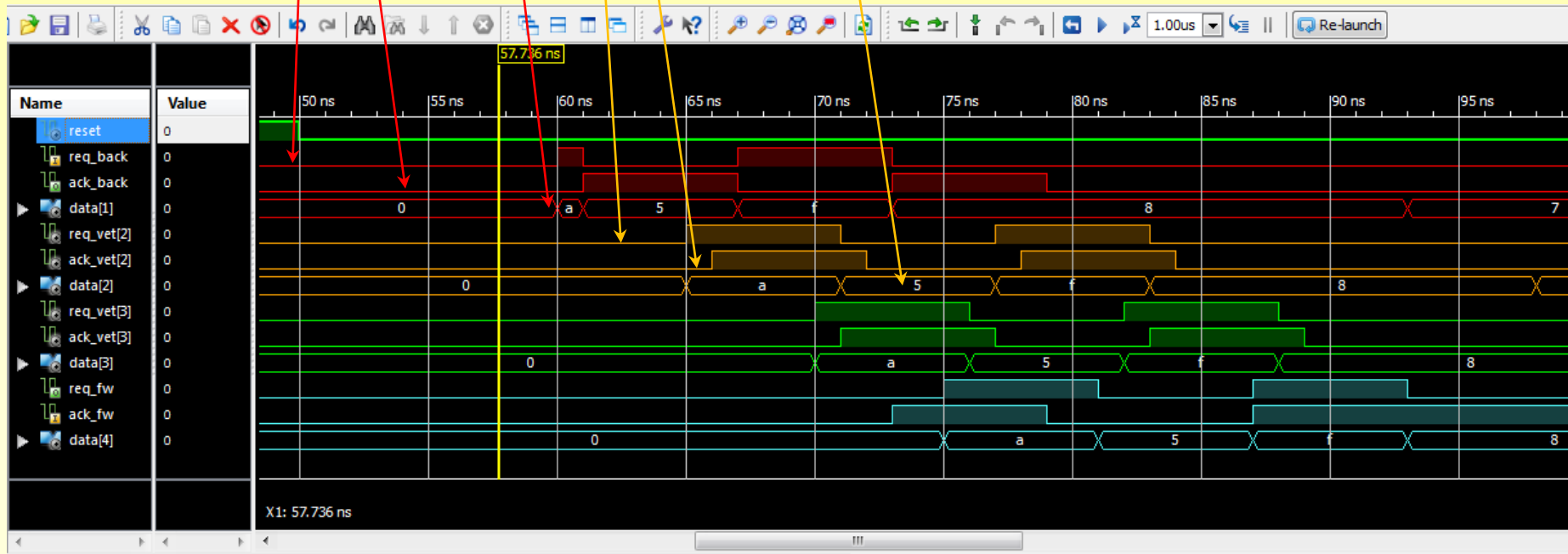
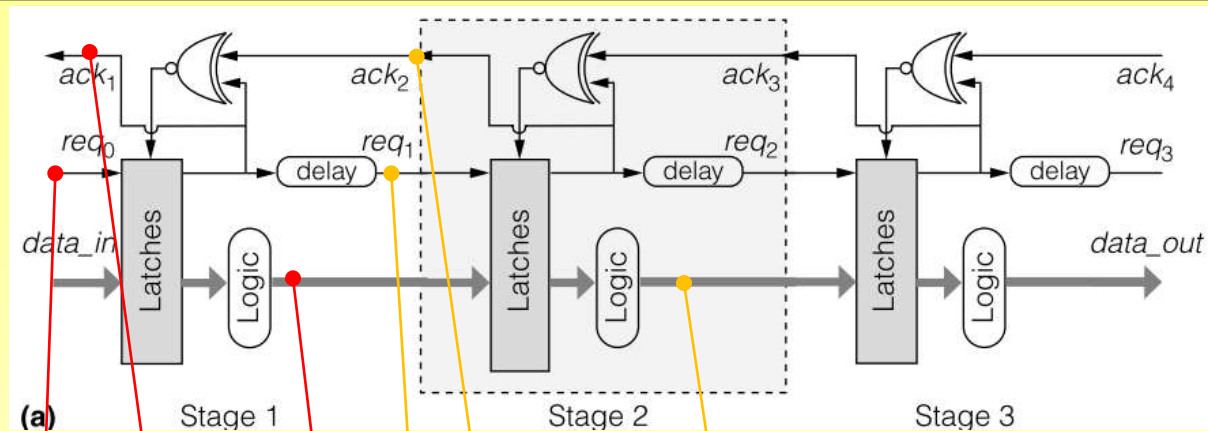


5 - Um Exemplo de Implementação

→ Pressupostos do Mousetrap

- ◆ protocolo de comunicação 2 fases
- ◆ dados empacotados (Bundled-data) – notar os elementos de atraso!
- ◆ lógica de eventos baseada em transições, não níveis lógicos
- ◆ Lógica de controle muito simples (uma porta Xnor)

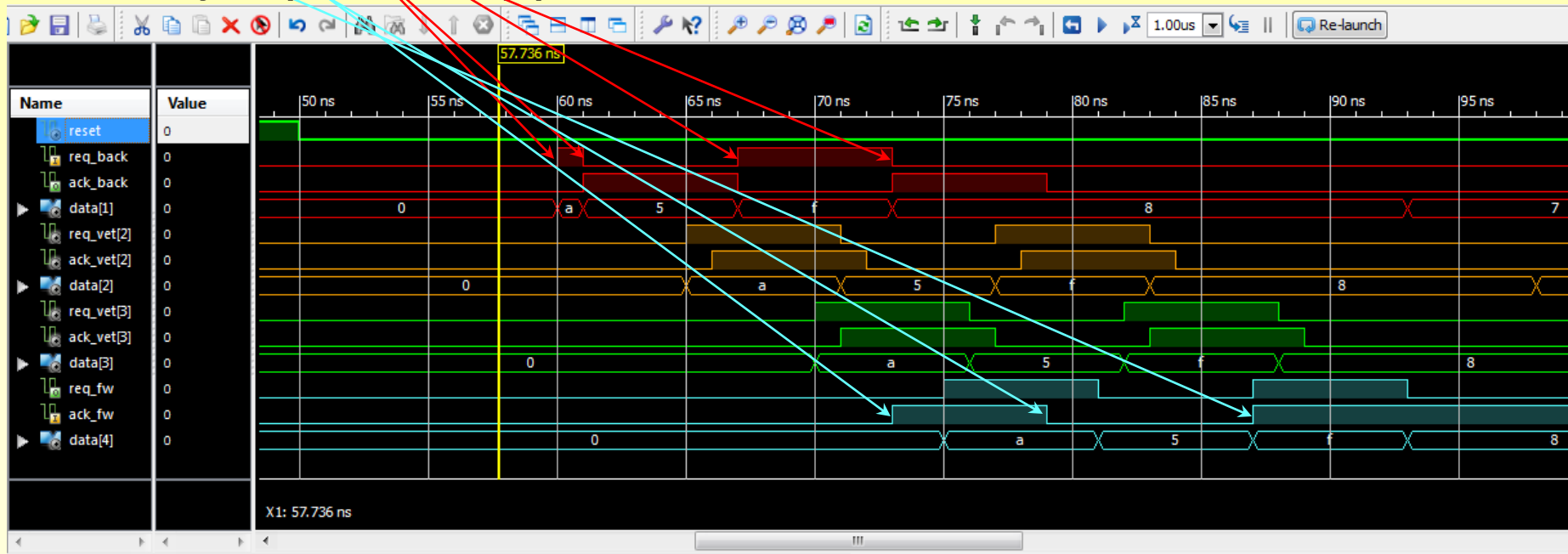
5 - Micropipelines - Estrutura de controle



5 - Micropipelines - Estrutura de controle

Passos desta simulação

- injeta primeiro dado, pipe contém 1 elemento
- injeta segundo dado, pipe contém 2 elementos
- injeta terceiro dado, pipe contém 3 elementos
- remove primeiro dado e injeta quarto dado, pipe contém 3 elementos
- remove segundo dado, pipe contém 2 elementos
- remove terceiro dado, pipe contém 1 elemento
- injeta quinto dado, pipe contém 2 elementos



- ✓ 0 - Motivação
- ✓ 1 - As(síncronos) - Representações
- ✓ 2 - Síncronos versus Não-síncronos
- ✓ 3 - Histórico, Ferramentas, etc.
- ✓ 4 - Interfaces Assíncronas
- ✓ 5 - Um Exemplo de Implementação
- ➔ 6 - Conclusões

6 - Conclusões

- Não-síncronos **SÃO INEVITÁVEIS!** Logo, são o **FUTURO!**
- Todo chip usa alguma parte assíncrona hoje. Maioria limita-se a sincronizadores de **BAIXO DESEMPENHO!**
- Comunicação (NoCs) já são **GALS** e mais tarde vão ser **ASSÍNCRONAS ! Atendem!**
- **EDA** (Electronic Design Automation) é um **PROBLEMÃO** e portanto uma **CLARA OPORTUNIDADE**, pois o tamanho dos chips só cresce.
- Adoção de não-síncronos será **GRADUAL!**
- Comecem a se preocupar **AGORA!** Aprendam a **PROJETAR** e **IMPLEMENTAR** assíncronos!
- **NINGUÉM** está pesquisando assíncronos no Brasil, com **1** exceção, que eu conheço! 😊