GasP Control for Domino Circuits

Jo Ebergen
Jonathan Gainsley
Jon Lexau
Ivan Sutherland

Sun Microsystems Laboratories

Presentation Outline

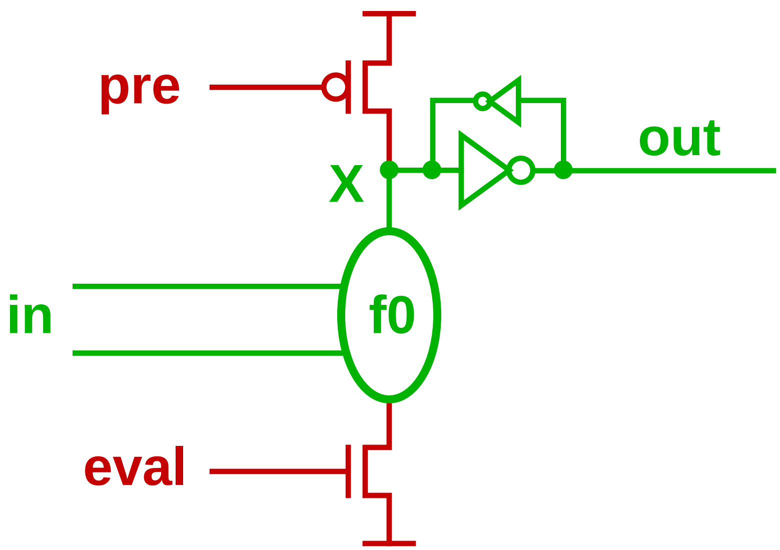
- **Domino Datapath**
- **Control Design Constraints**
- **GasP2 - complex**
- **GasP1 - simpler**
- **Test Chips**
- **Conclusions**

Motivation

- elastic nature of FIFO
- energy savings when input is “bursty”
- very hard to generate fine-grained “clocks” synchronously
- possibly better matching between control and datapath
- can we do it?

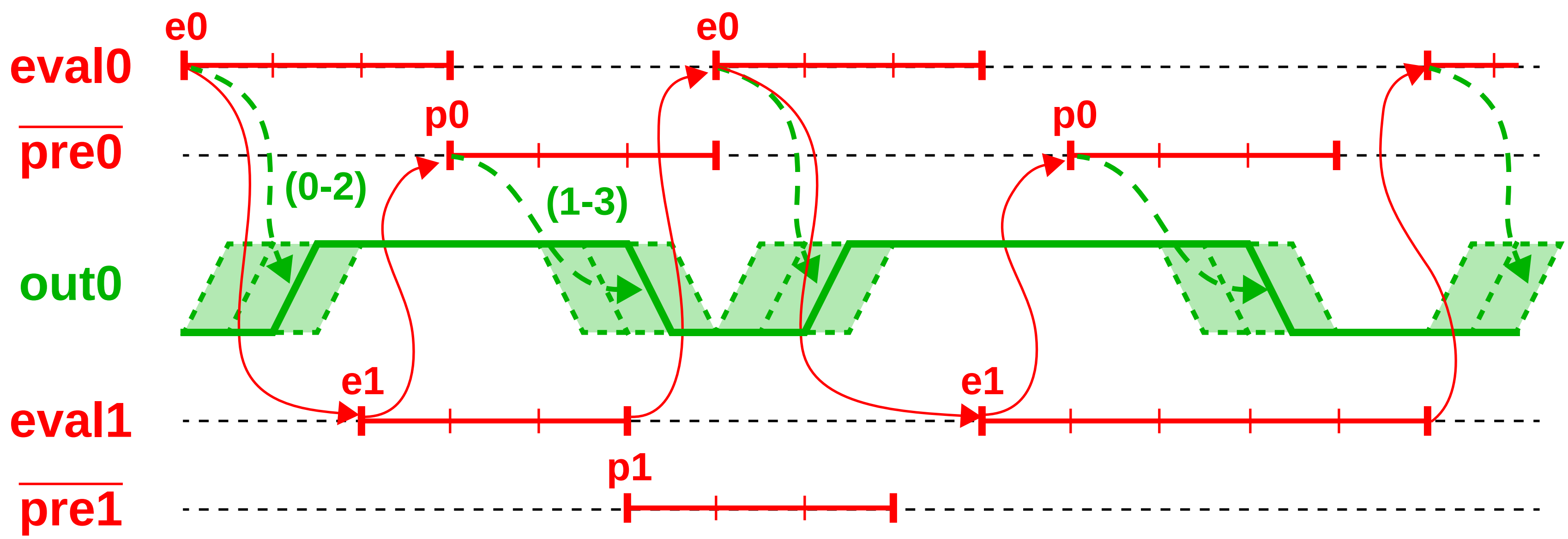
Domino Datapath

- Separate evaluate / precharge lines
- One domino stage per control stage

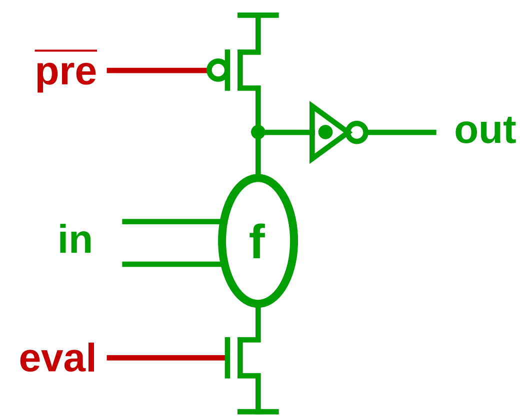


eval	pre	state
LO	LO	precharge
HI	HI	evaluate
LO	HI	hold
HI	LO	invalid

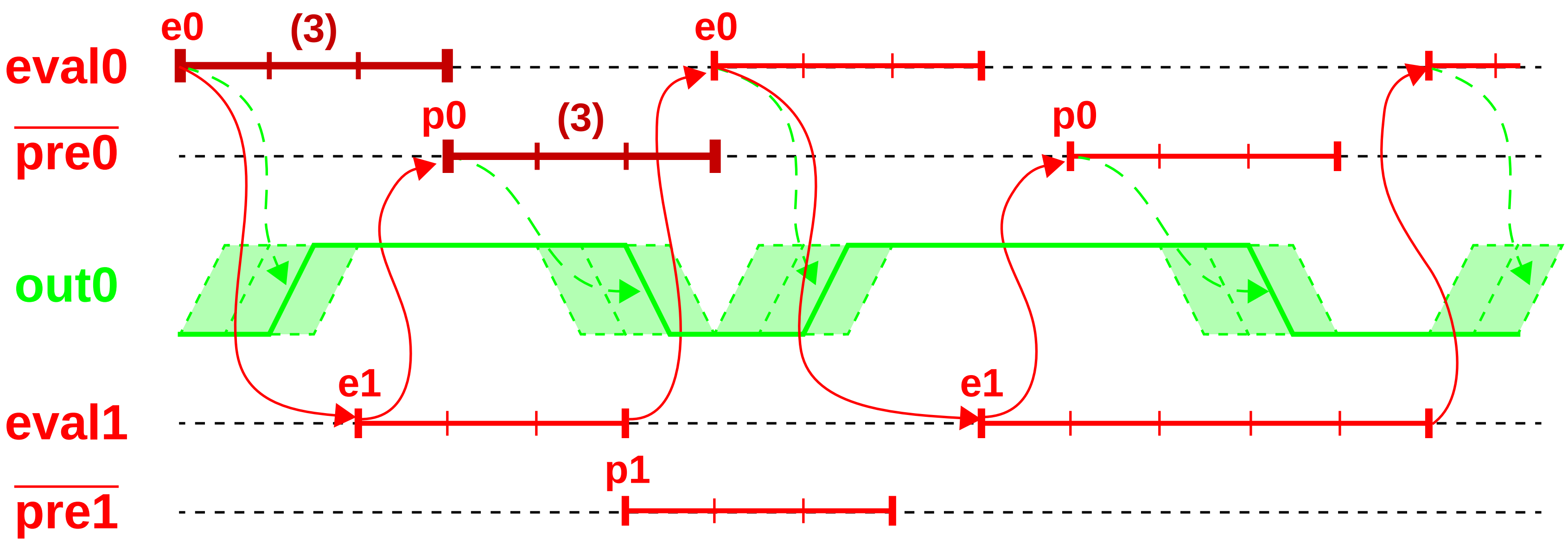
Control Design Constraints



- **Datapath timing**
  - **$0 < \delta_{eval} < \Delta_{eval} \le 2D$**
  - **$D < \delta_{pre} < \Delta_{pre} \le 3D$**

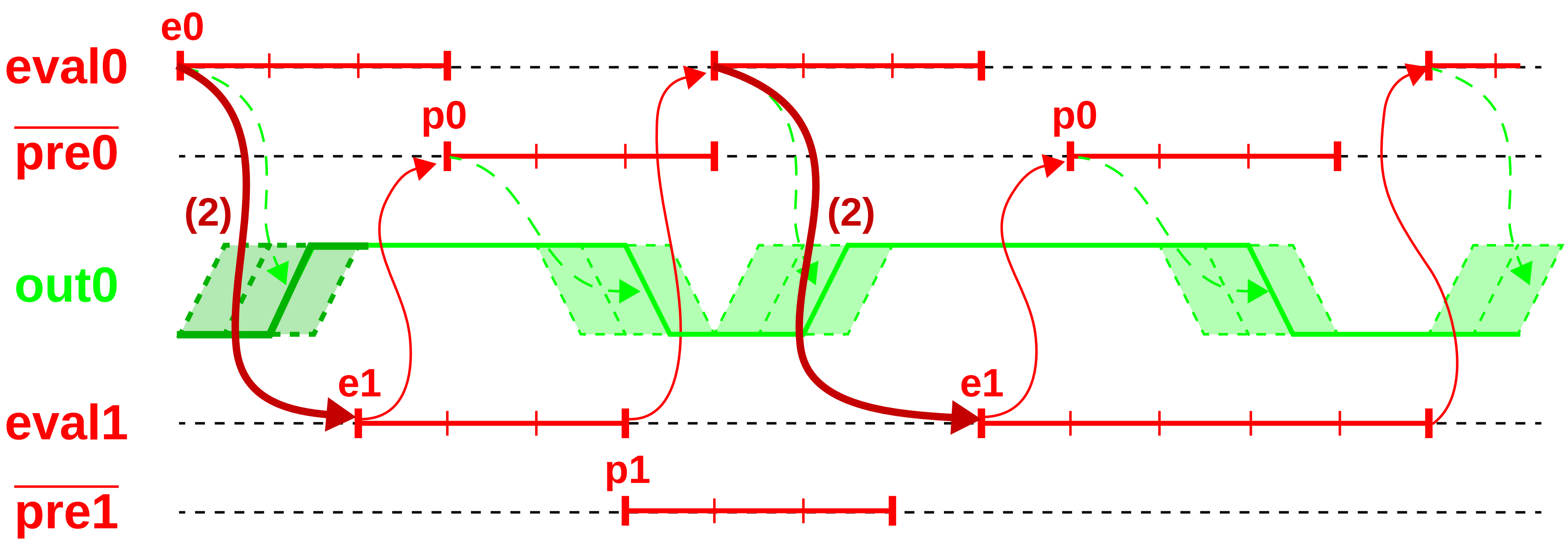


Control Design Constraints



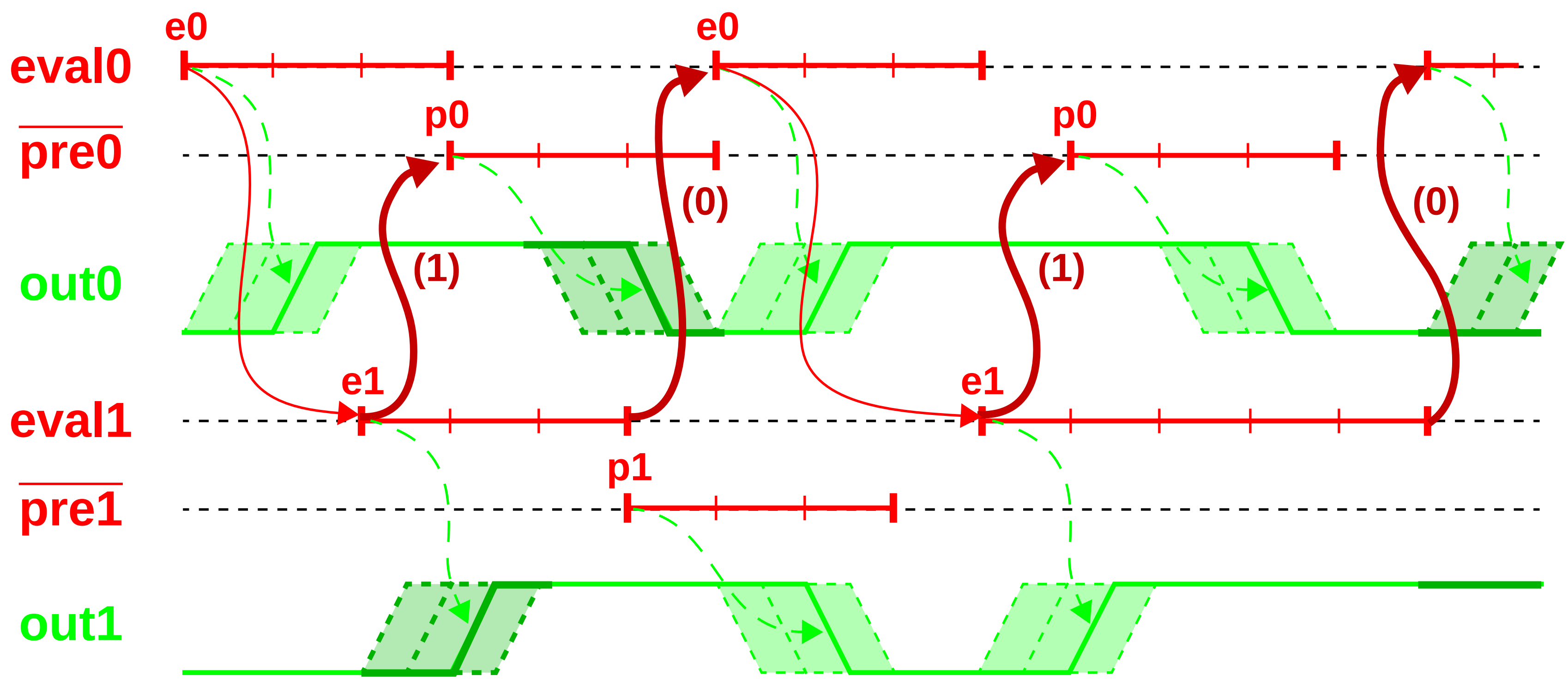
- **Evaluate / precharge timing**
  - **e0 $\xrightarrow{3}$ e0*** **p0 $\xrightarrow{3}$ p0***
  - **No overlap of evaluate and precharge**

Control Design Constraints



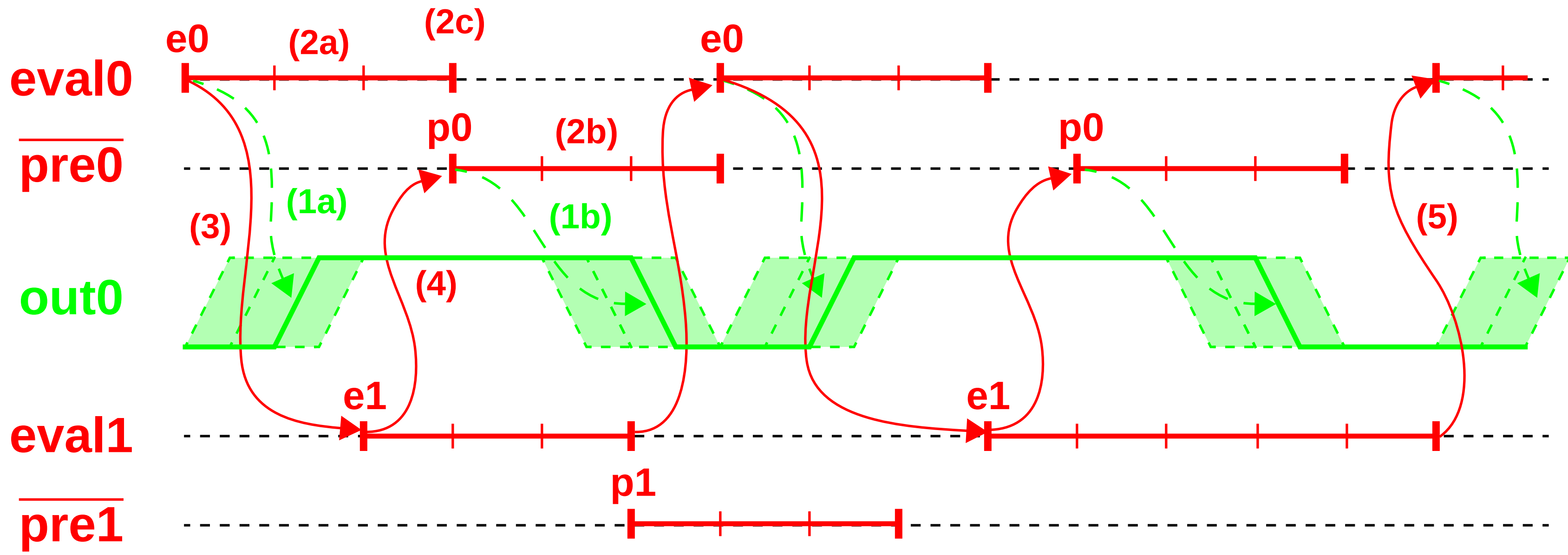
- **Setup time**
  - **evaluation must not start until inputs are valid**
  - **e0 $\xrightarrow{2}$ e1**

Control Design Constraints



- **Hold time**
  - **evaluation must finish before prior stage may precharge or begin the next evaluation**
  - **e1 $\xrightarrow{1}$ p0** **e1* $\xrightarrow{0}$ e0**

Constraint Summary



(1a) $0 < \delta_{eval} < \Delta_{eval} \le 2D$

(1b) $D < \delta_{pre} < \Delta_{pre} \le 3D$

(2a) $e0 \xrightarrow{3} e0^*$

(2b) $p0 \xrightarrow{3} p0^*$

(2c) no eval/pchg overlap

(3) $e0 \xrightarrow{2} e1$

(4) $e1 \xrightarrow{1} p0$

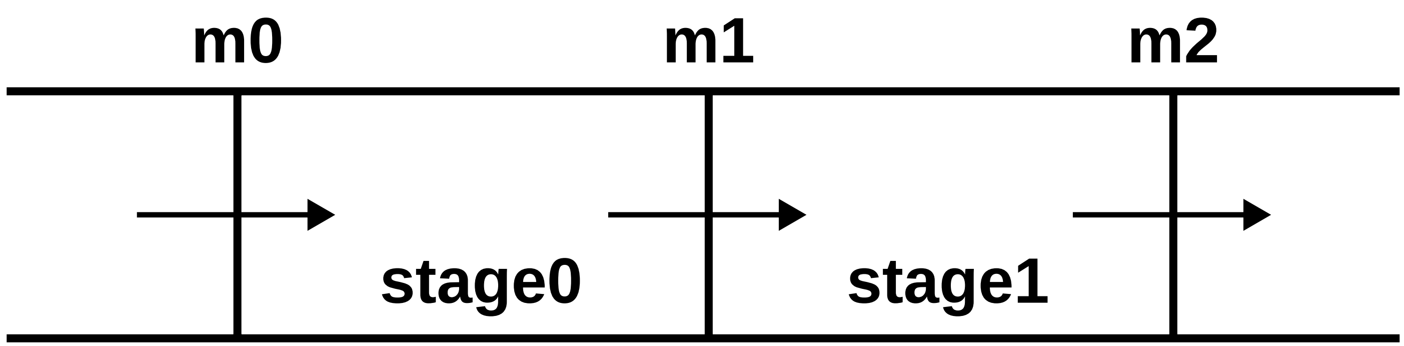
(5) $e1^* \xrightarrow{0} e0$

GasP2 (Complex)

- **Our first control circuit attempt**
- **Not designed using formal constraints**
- **Does not take advantage of “hold” state**

GasP2

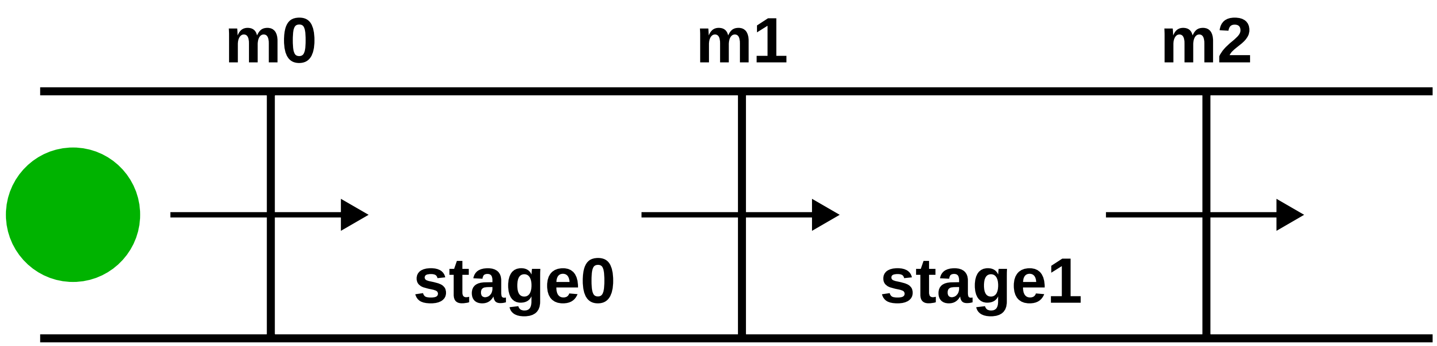
- **Assign FULL=eval, EMPTY=precharge**
- **Only one token per pair of stages**
- **Identify border crossings m0, m1, m2**



m0 → m1 → m2

GasP2

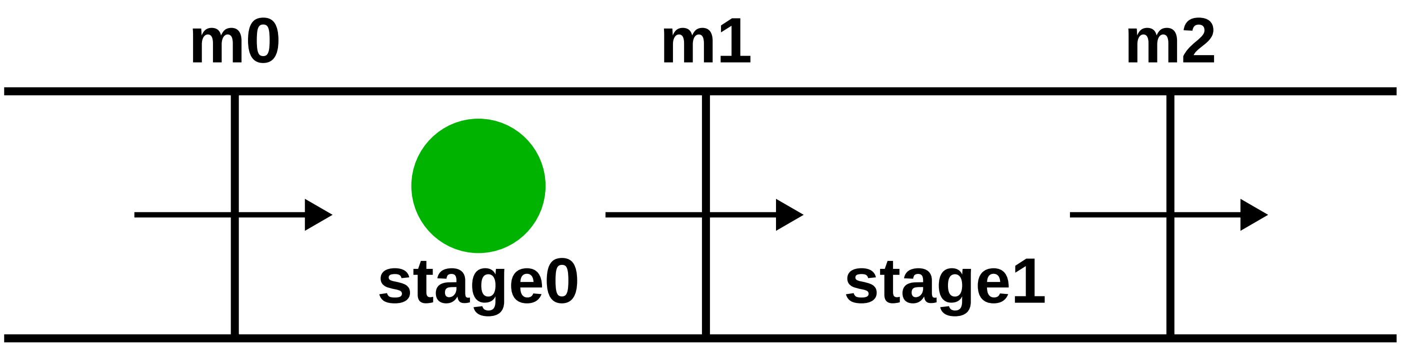
- **Assign FULL=eval, EMPTY=precharge**
- **Only one token per pair of stages**
- **Identify border crossings m0, m1, m2**

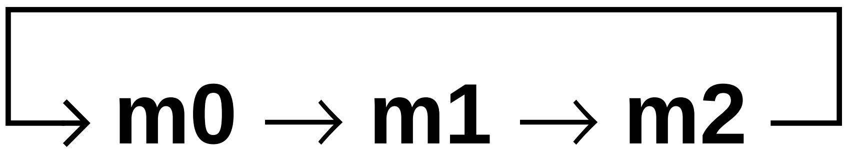


$\rightarrow$ m0 $\rightarrow$ m1 $\rightarrow$ m2

GasP2

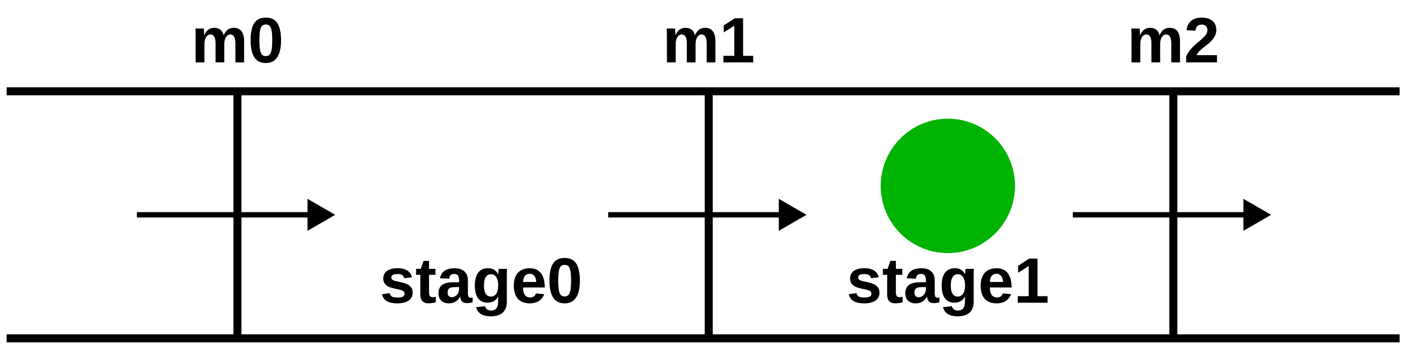
- **Assign FULL=eval, EMPTY=precharge**
- **Only one token per pair of stages**
- **Identify border crossings m0, m1, m2**





GasP2

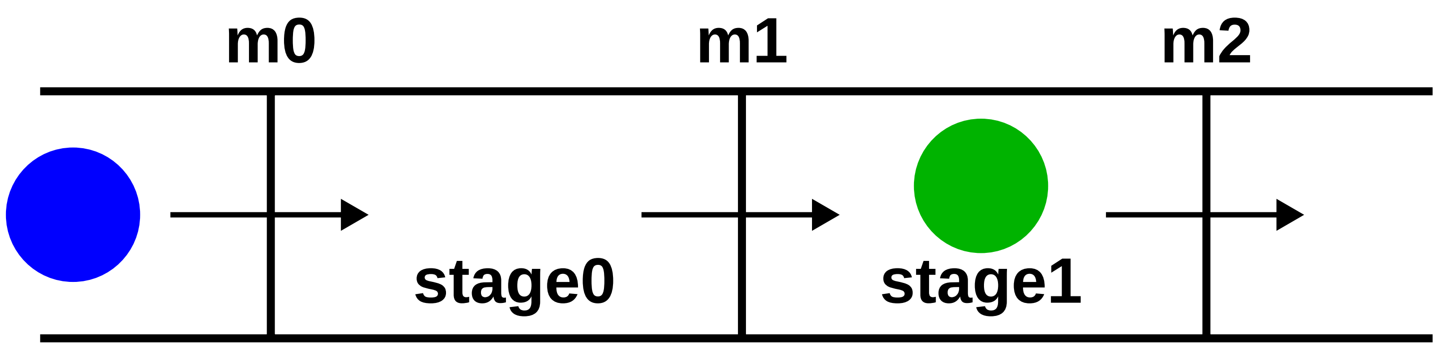
- Assign FULL=eval, EMPTY=precharge
- Only one token per pair of stages
- Identify border crossings m0, m1, m2



$\rightarrow$ **m0 $\rightarrow$ m1 $\rightarrow$ m2**

GasP2

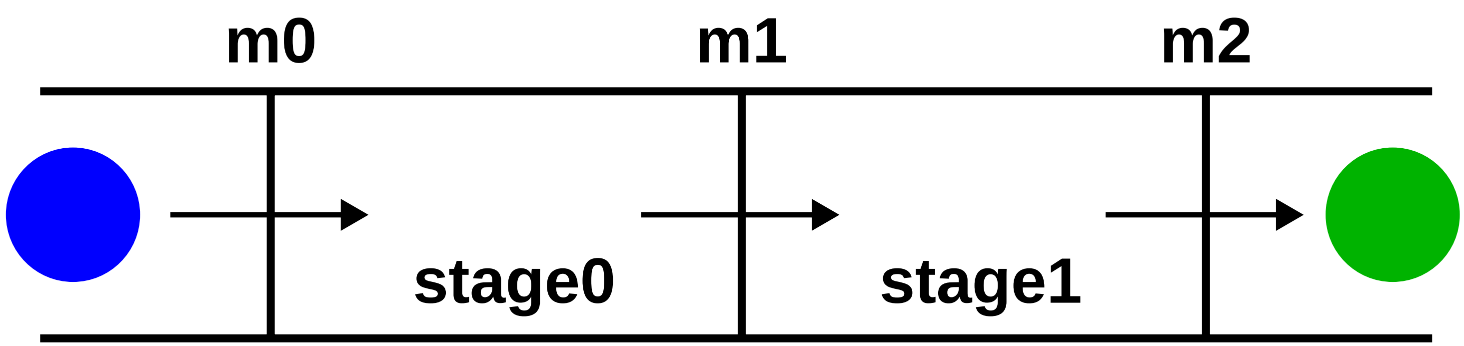
- **Assign FULL=eval, EMPTY=precharge**
- **Only one token per pair of stages**
- **Identify border crossings m0, m1, m2**



→ m0 → m1 → m2

GasP2

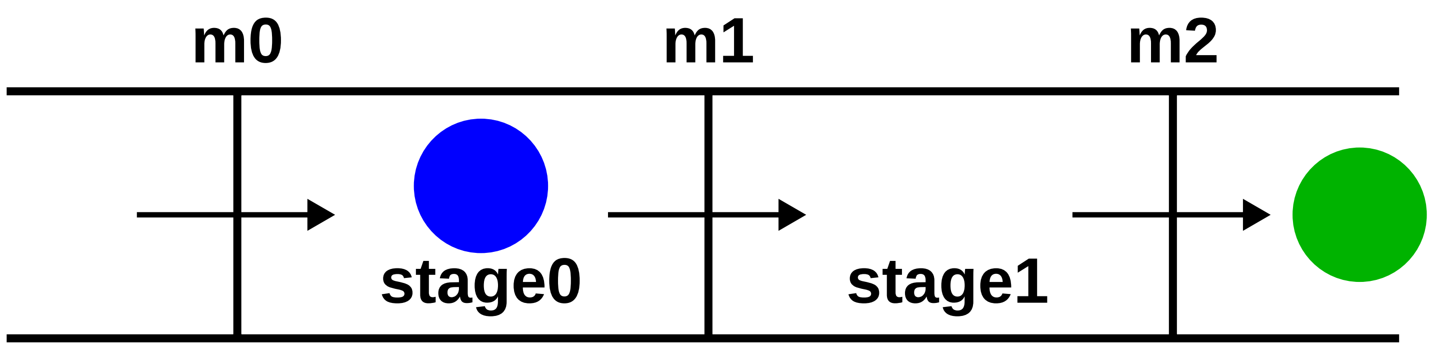
- **Assign FULL=eval, EMPTY=precharge**
- **Only one token per pair of stages**
- **Identify border crossings m0, m1, m2**



$\to$ m0 $\to$ m1 $\to$ m2

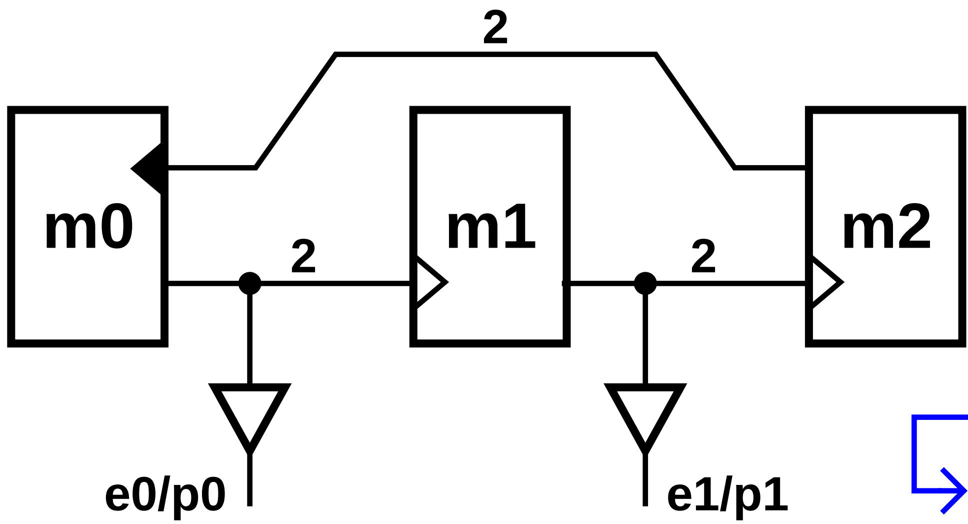
GasP2

- **Assign FULL=eval, EMPTY=precharge**
- **Only one token per pair of stages**
- **Identify border crossings m0, m1, m2**

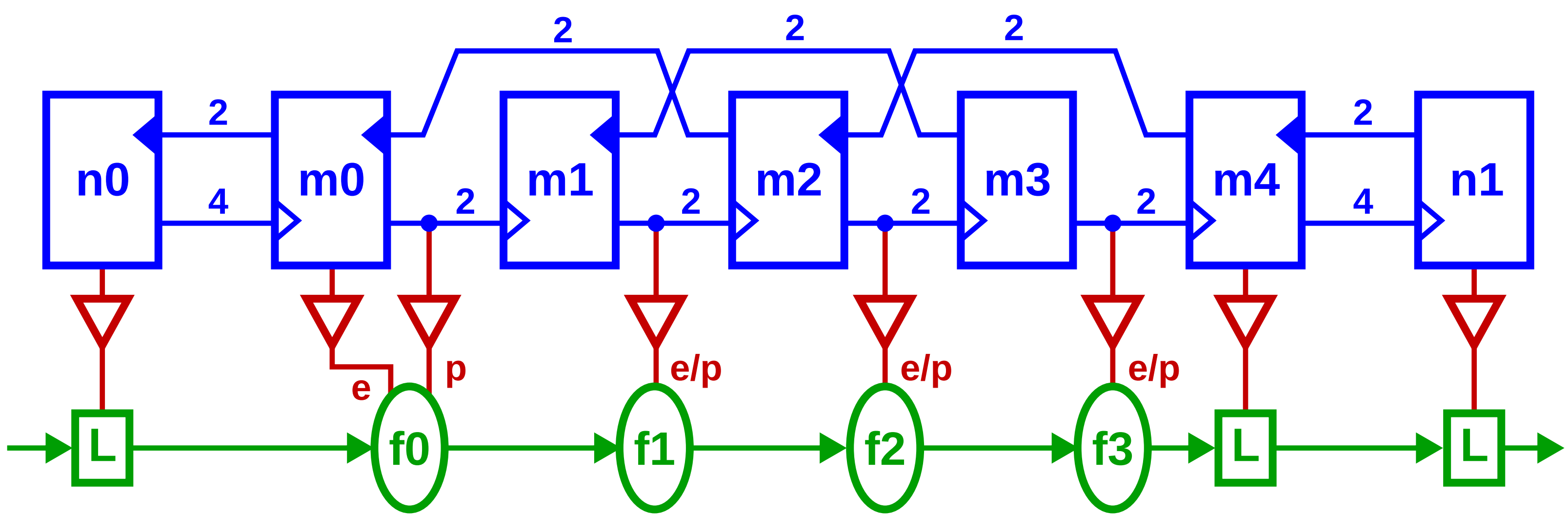


m0 → m1 → m2

GasP2


2
m0
2
m1
2
m2
e0/p0
e1/p1


m0 → m1 → m2

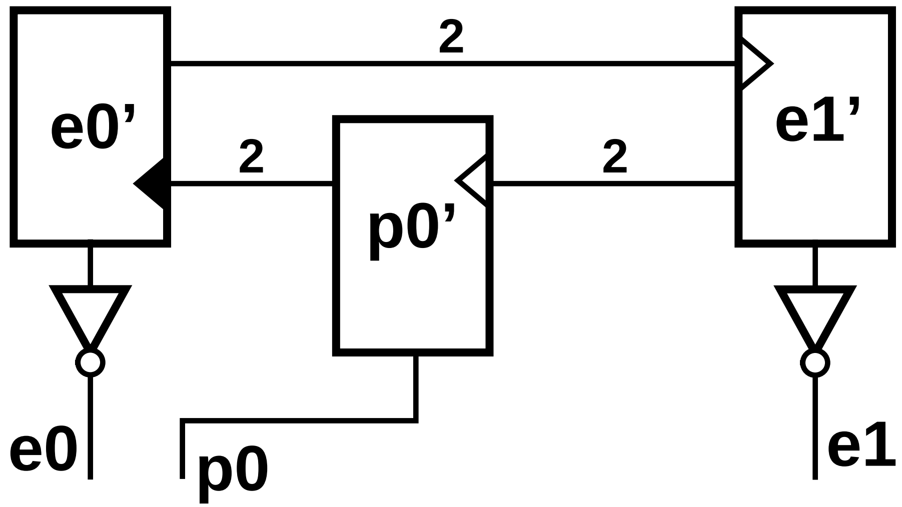

2
2
2
2
n0
m0
m1
m2
m3
m4
n1
4
2
2
2
2
4
2
e
p
e/p
e/p
e/p
L
f0
f1
f2
f3
L
L

GasP2 Observations

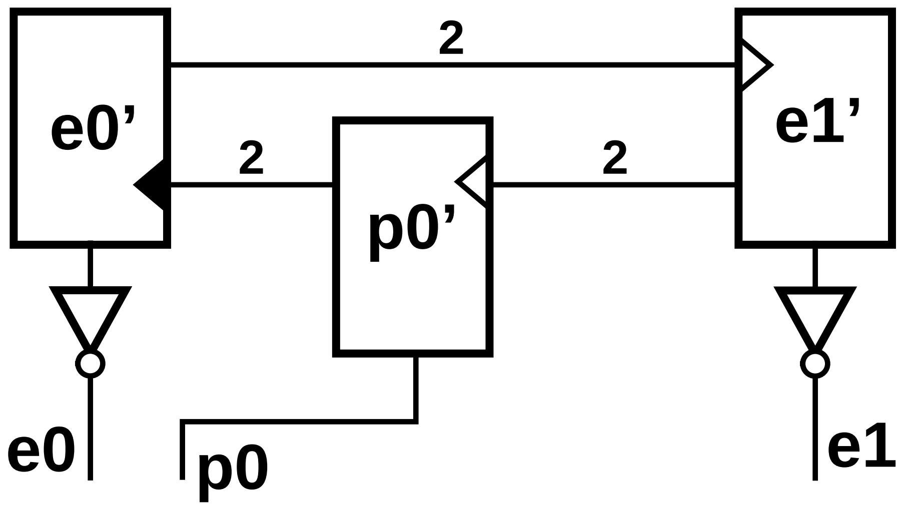
- **Feedback state wires which must cross over a stage**
- **Four unique "interface" stages required (m0, m1, m3, m4)**
- **Combined evaluate/precharge signals force one token per two stages**
- **Branch/merge would be difficult**

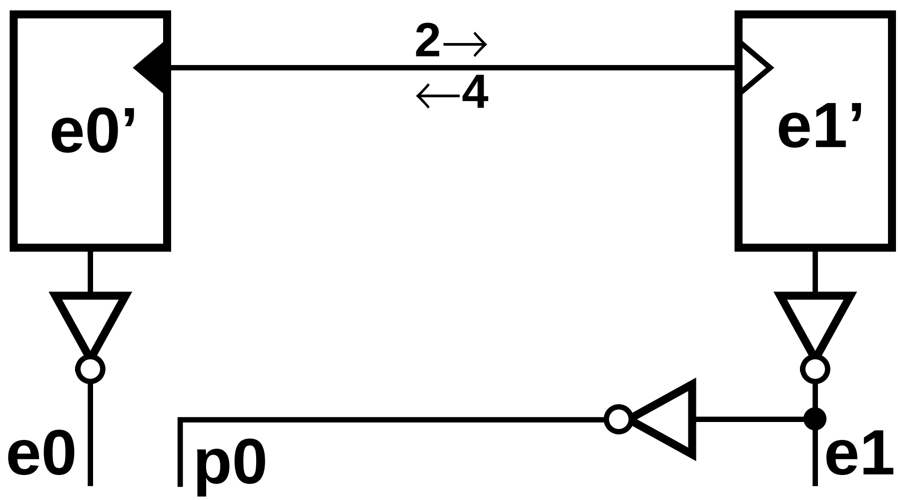
GasP1 (Simpler)

- $S0 = (e0 \xrightarrow{2} e1 \xrightarrow{1} p0 \xrightarrow{3} S0)$
- $U0 = (e0' \xrightarrow{2} e1' \xrightarrow{2} p0' \xrightarrow{2} U0)$

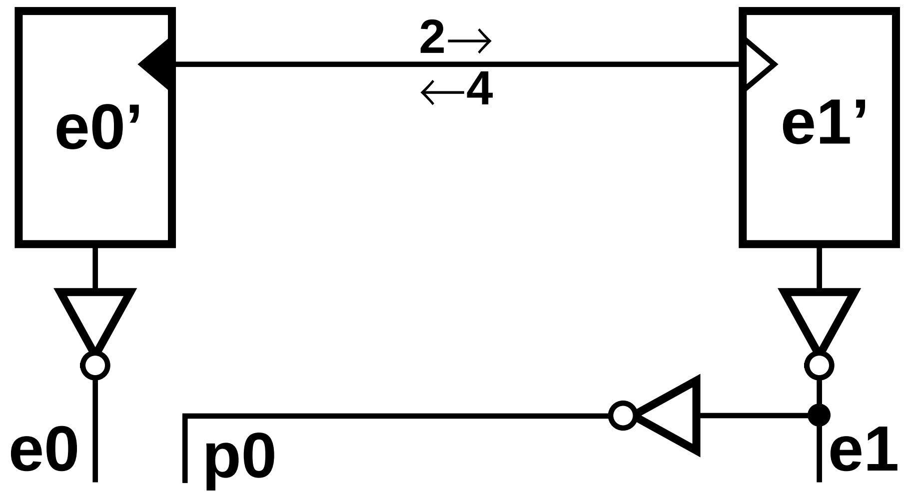


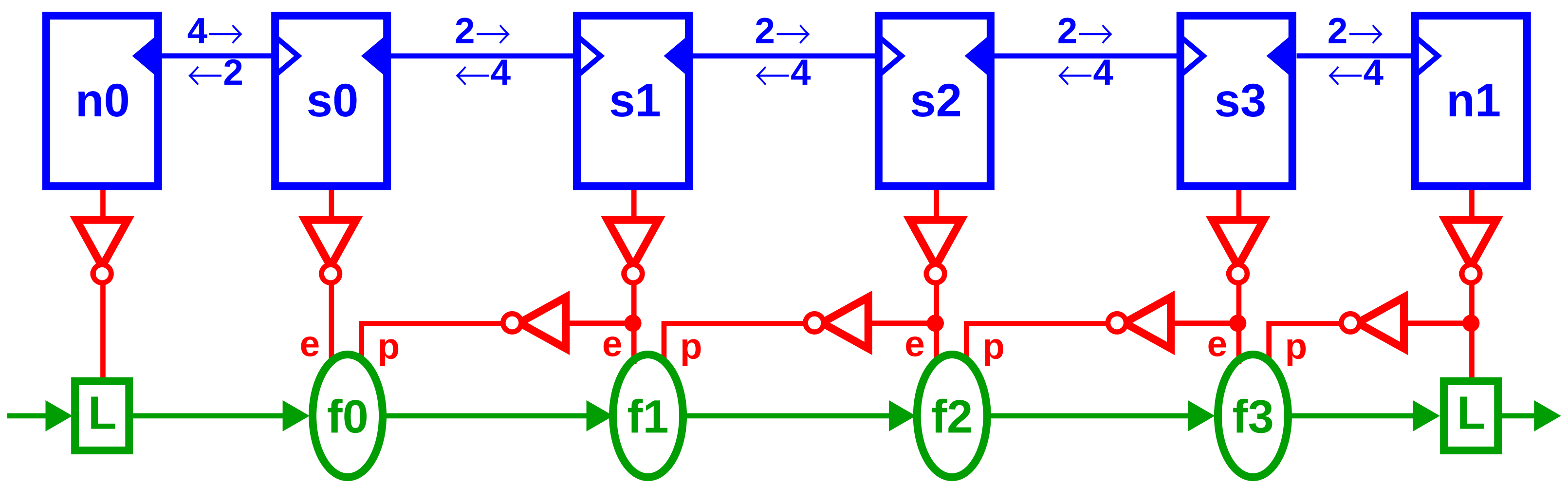
GasP1





GasP1

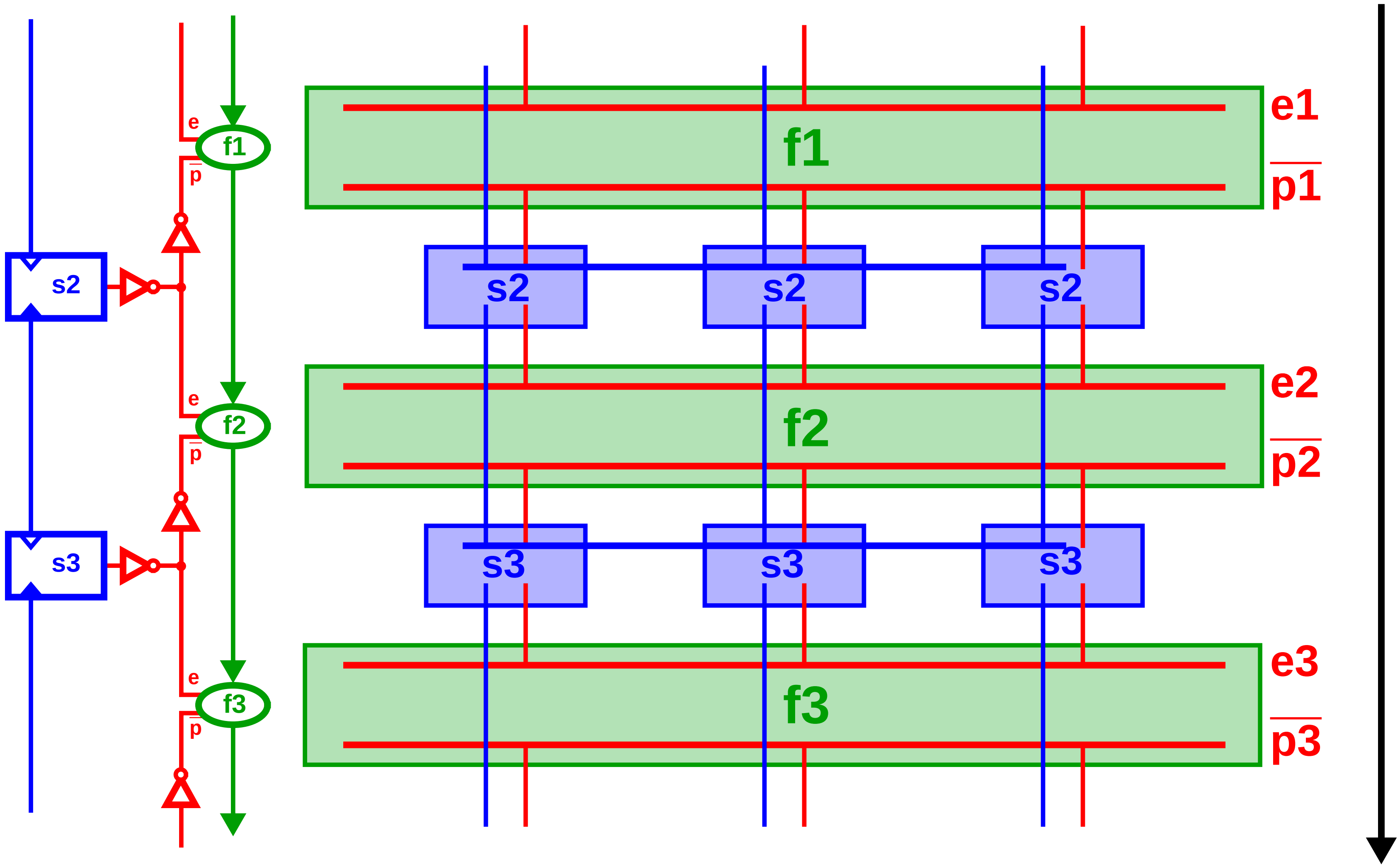

2→
←4
e0'
e1'
e0
p0
e1


4→
←2
2→
←4
2→
←4
2→
←4
2→
←4
n0
s0
s1
s2
s3
n1
e
p
e
p
e
p
e
p
L
f0
f1
f2
f3
L

GasP1 Advantages

- **Only one state wire per stage**
- **Two unique “interface” stages (s0,n1)**
- **Evaluate/precharge are separate which allows a token in every stage**
- **Domino stages (2/4) are just FIFO stages (4/2) “backwards”**
- **Branch/merge/interface is simple**

GasP1 Layout



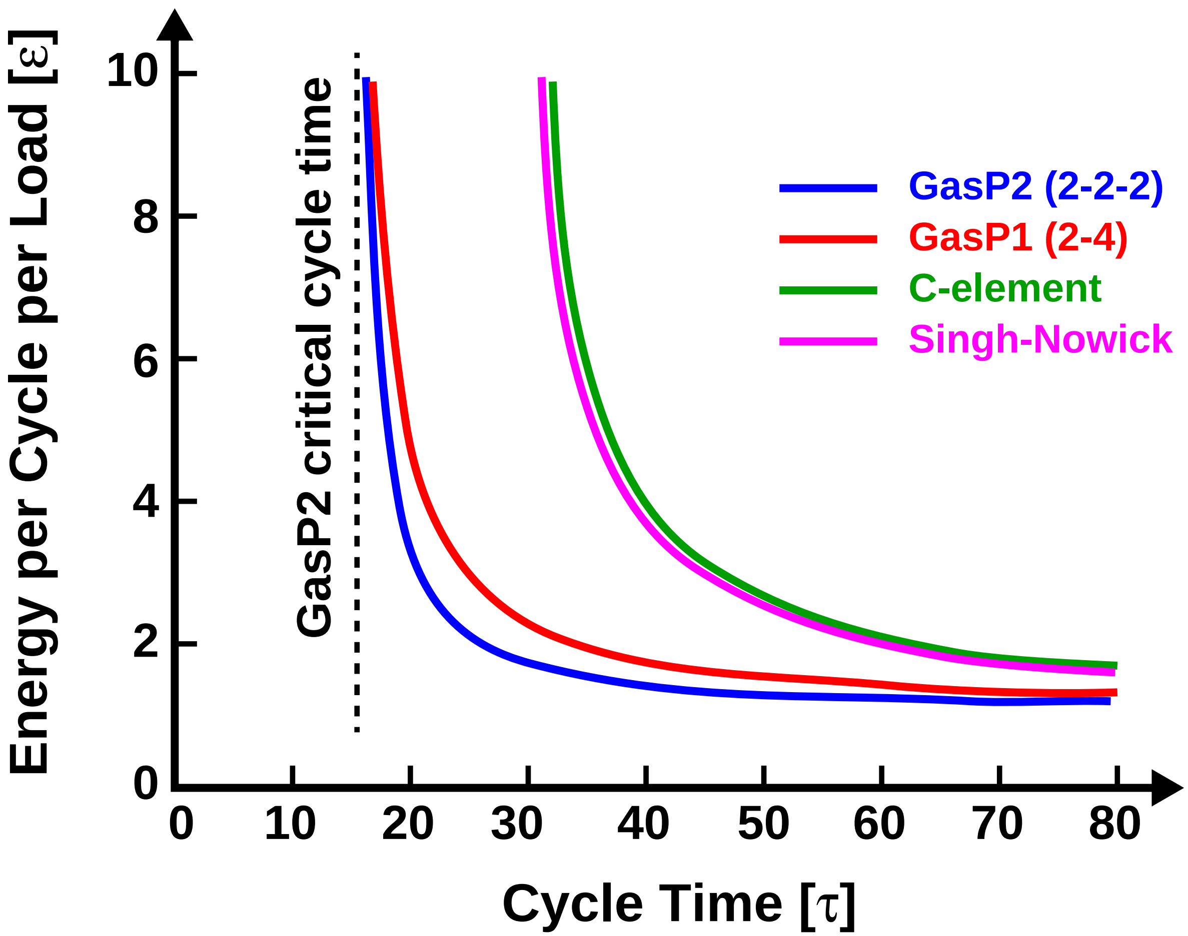
Design Comparison

- **Critical cycle time**
  - **lower bound on cycle time**
  - **estimated using Logical Effort**

design style	critical cycle time	gate delays
C-element chain	27.4 τ	6
Singh-Nowick (LP_{HC})	27.0 τ	8
GasP2 (2-2-2)	15.1 τ	6
GasP1 (2-4)	14.5 τ	6

Design Comparison

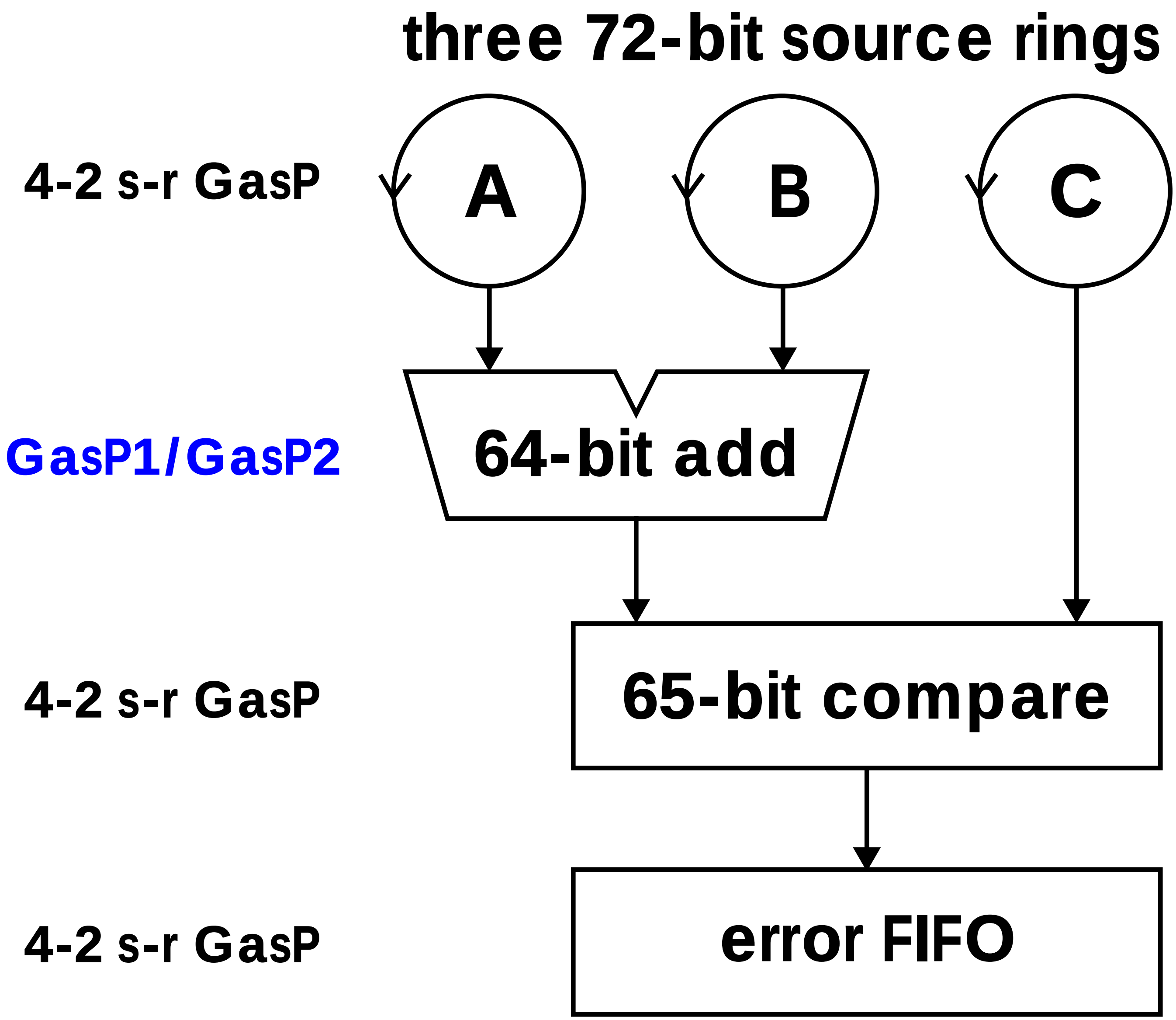
- Energy vs cycle time



Test Chips

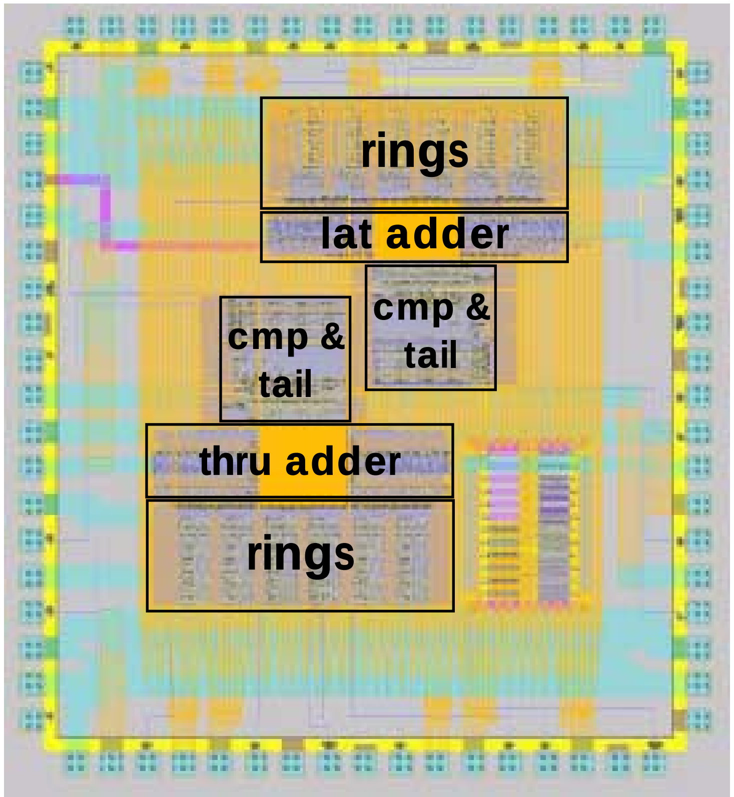
- **Built two different test chips with various domino adder datapaths**
- **TSMC 180nm CMOS process**
- **All control sized using Logical Effort**
- **Layout sizing was semi-automated, using libraries of self-resetting components**

Test Structure



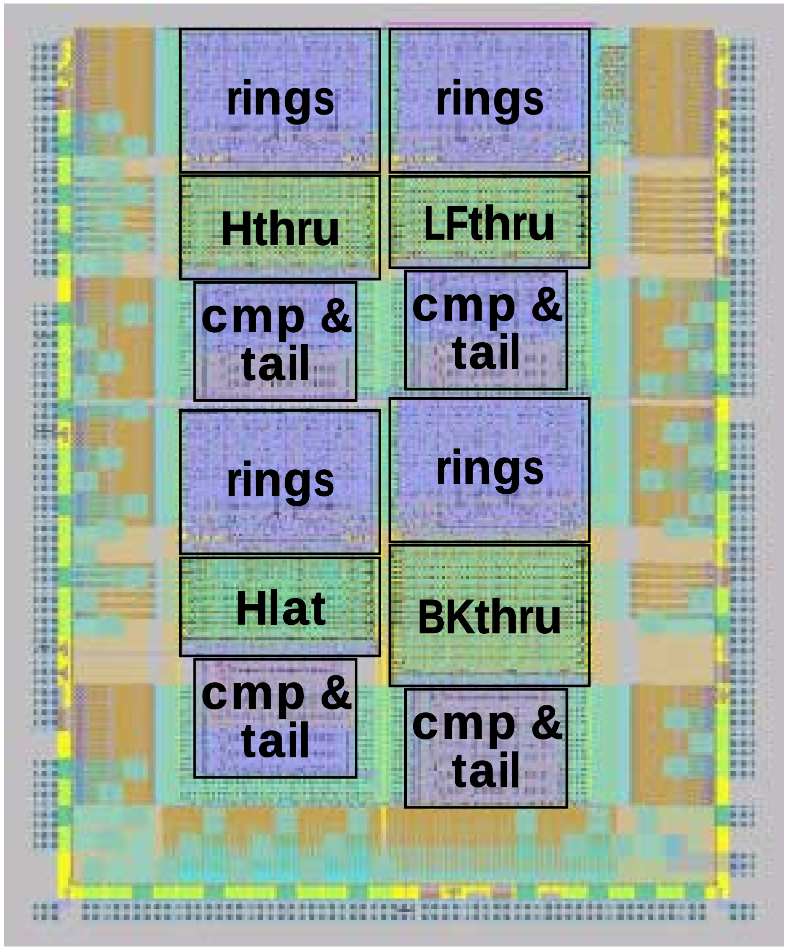
Chip Plots

PGKadder



2.8mm x 3.17mm

four adders



3mm x 3.76mm

The Chips

chip	PGKadder	four adders
# of adders	1 thru / 1 lat	3 thru / 1 lat
control style	GasP2 self-reset	GasP1 self-reset
fab date	Jan 2002	Dec 2002
ring perf	3.7 GHz	3.2 GHz
adder perf	3.7 GHz	2.2 GHz
adder latency	980 ps (8 stage)	1032 ps (9 stage)
datapath area	1238 x 88 μm	813 x 246 μm

Learning Opportunities

- PGKadder chip
  - data bundling violations
  - cross-coupling between data wires
  - control sizing was too aggressive
  - slow precharge in datapath
  - domino control not well matched to datapath
- Four adders chip
  - two power supplies accidentally shorted
  - precharge / evaluate overlap
  - adder control is slower than expected

Lessons Learned

- **We like GasP1 better than GasP2**
  - **fewer, shorter state wires**
  - **lower complexity**
- **Layout aspect ratio is important**
- **Extracted simulation is required for a high performance design**
- **Would like an adjustable control delay**