Playing with Time and Space in Circuits and Programs

Gérard Berry

Collège de France

Chaire Algorithmes, machines et langages

PPS, 24/04/2013

COLLÈGE DE FRANCE
1530

Agenda

1. 2-adic numbers and space / time exchange in synchronous circuits
2. Never determinize non-deterministic automata !
3. Use hierarchical automata for another exponential gain in space and timing optimization

Agenda

1. 2-adic numbers and space / time exchange in synchronous circuits
2. Never determinize non-deterministic automata !
3. Use hierarchical automata for another exponential gain in space and timing optimization

Source of the 2-adic Part

J. Vuillemin. *On circuits and numbers*,
IEEE Trans. on Computers, 43:8:868-79, 1994

2-adic Numbers (Hensel, ~1900)

- R is a completion de Q. Is it the only one ?
  No : p-adic numbers for p prime
  infinite numbers written low-order bits first

- Beautiful, but physical ? *cf. Matière à Pensée, p. 32*
  Alain Connes / JP Changeux

- Jean Vuillemin : 2-adiques integers are the right model of arithmetic digital circuits
  Let us create their physics!

2-adic numbers unify computable arithmetic with Boolean logic

${}_2Z$: the Ring of 2-adic Numbers

$x = {}_2x_0x_1x_2 \ldots$ low-order bits first

operations $+$ and $\times$ from left to right

$0 = {}_200000\ldots = {}_2(0)$

$1 = {}_210000\ldots = {}_21(0)$ $\quad -1 = {}_211111\ldots = {}_2(1)$

$2 = {}_201000\ldots = {}_201(0)$ $\quad -2 = {}_201111\ldots = {}_20(1)$

$x = {}_2101010\ldots = {}_2(10)$

$= {}_2100000\ldots + {}_2001010\ldots$

$= 1 + 4x$

$x = -1/3$

$y = {}_2010101\ldots = {}_2(01)$

$y = 2x$

or $x + y = -1$

$y = -2/3$

$_2Z$: *the Ring of 2-adic Numbers*

±p/q exists for all integer p, q iff q est odd (cf. Euclide)

1/2 does not exist
because x_0+x_0 cannot have value 1

No order compatible with the operations

~~$-1 \leq 0 \leq 1$~~

${}_2Z$ *as a Boolean Algebra*

- 2-adic x seen as the set $\{ i \mid x_i = 1 \}$

  example: $-1/3 = {}_2101010... = \{ i \mid i \text{ even} \}$

- pointwise Boolean operations

  $x \wedge y \qquad x \vee y \qquad \neg x$

  $(x \wedge y)_n = x_n \wedge y_n$ etc.

- Fundamental arithmetico-logical equality :

$$x + \neg x = -1$$

$$\begin{array}{l} {}_2100011... \\ {}_2011100... \\ \hline {}_2111111... \end{array}$$

Cantor Metric Space

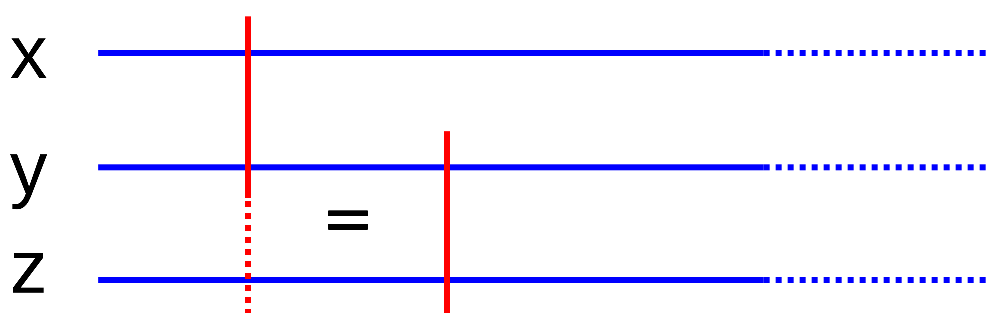
$d(x,x) = 0$

$d(x,y) = 2^{-n}$ $\quad$ n minimal s.t. $x_n \neq y_n$

Example : $d\,({}_2 011\underline{1}1\ldots,\ {}_2 011\underline{0}1\ldots = 1/8)$

- <u>Lemma :</u> ${}_2Z$ is ultrametric :

$$d(x,z) \leq \max(d(x,y), d(y,z))$$



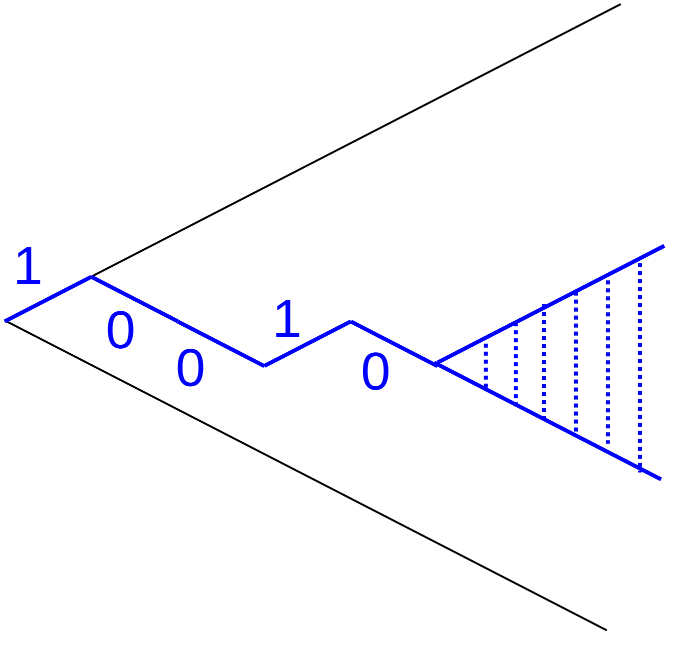
$$d(x,z) = \min(d(x,y), d(y,z))$$

Cantor Metric Space

- Open set basis : finite prefixes

$$x_0x_1...x_n \rightarrow \{_2x_0x_1...x_ny_0y_1...y_n... \mid y \in {_2Z}\}$$

ex.: open set for $_2 10010$



- Compact – very different from reals !

Continuous functions

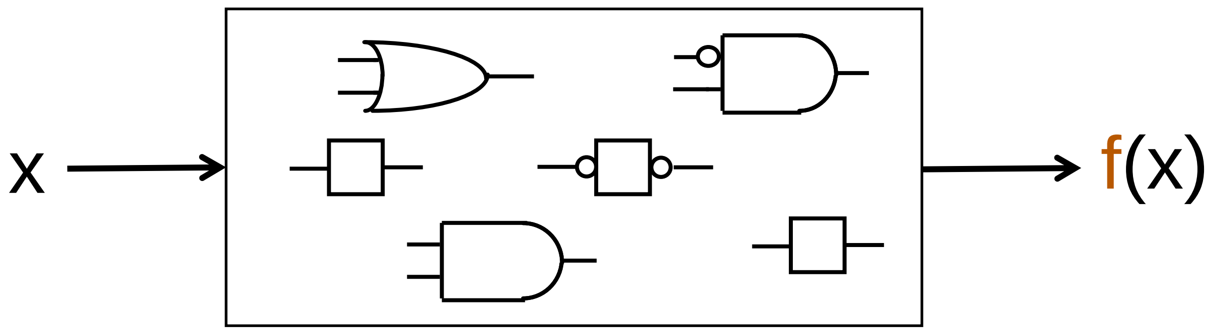
<u>Lemma :</u> $f : {}_2Z \rightarrow {}_2Z$ continuous iff

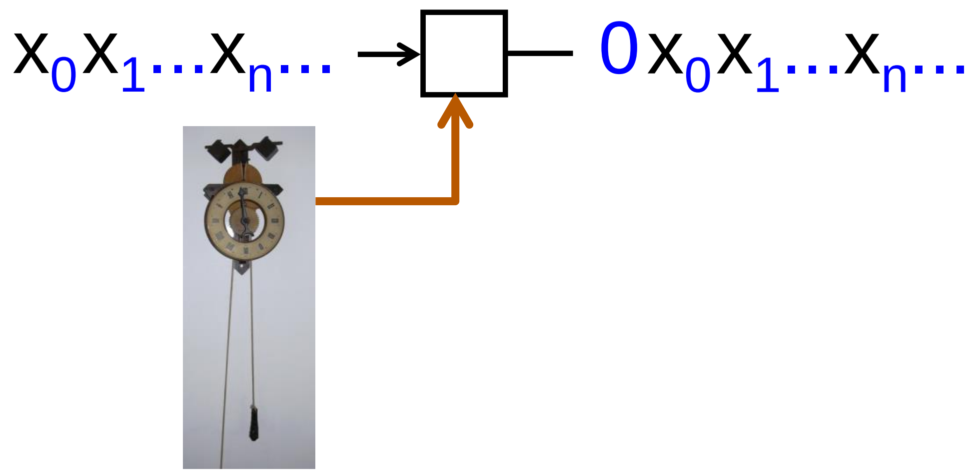
$f(x)_n$ depends on a finite number of x_m

$${}_2x_0x_1\ldots\ldots x_m\ldots \longrightarrow {}_2y_0y_1\ldots y_n\ldots$$

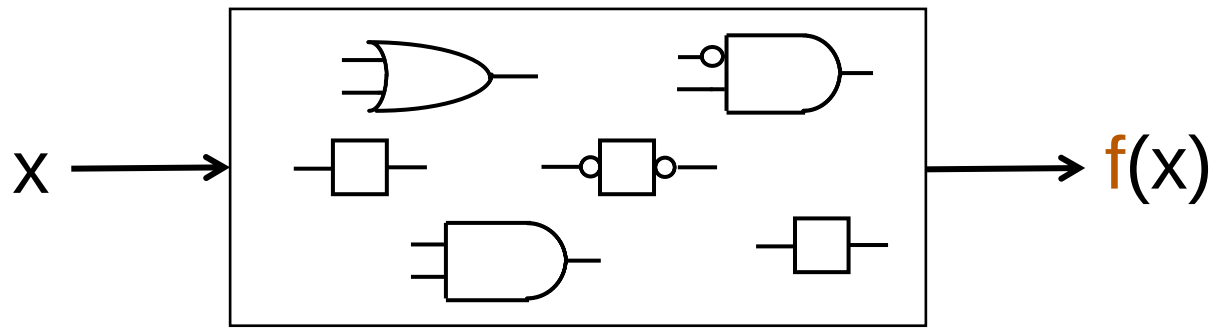
Continuity =
preservation of information finiteness

Synchronous Functions





Synchronous and Contracting Functions



- <u>Definition :</u> f : $_2Z \to {}_2Z$ synchronous iff computable by a synchronous circuit (with finite or infinite memory)
- <u>Theorem :</u> f : $_2Z \to {}_2Z$ is synchronous iff $f(x)_n$ only depends on $x_0x_1 \ldots x_n$, i.e., iff f is contracting

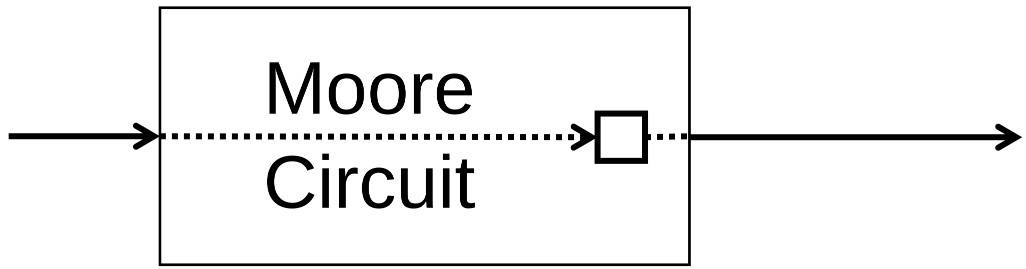
$$\forall x,y.\ d(f(x),f(y)) \leq d(x,y)$$

<u>Preuve</u> : « only if » trivial,

for « if » see SDD construction later on

Moore Circuits and Strict Contraction

- A Moore synchronous circuit is such that any wire between an input and an output traverses a register

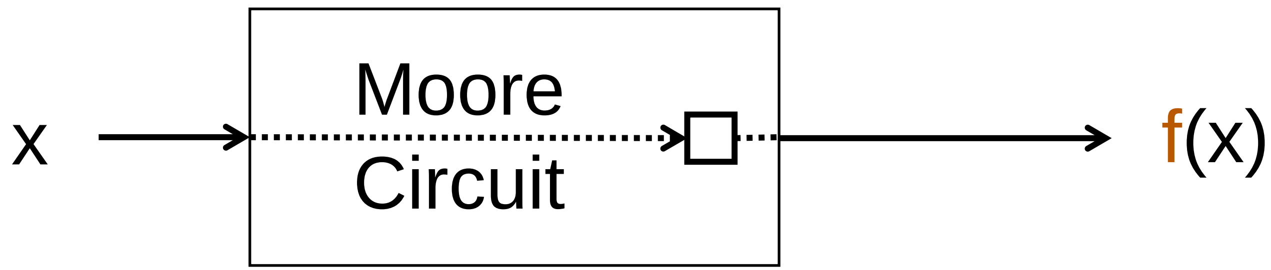


- A function f : $_2Z \rightarrow {}_2Z$ is strictly contracting iff f(x)$_n$ only depends on $x_0 x_1 \ldots x_{n-1}$

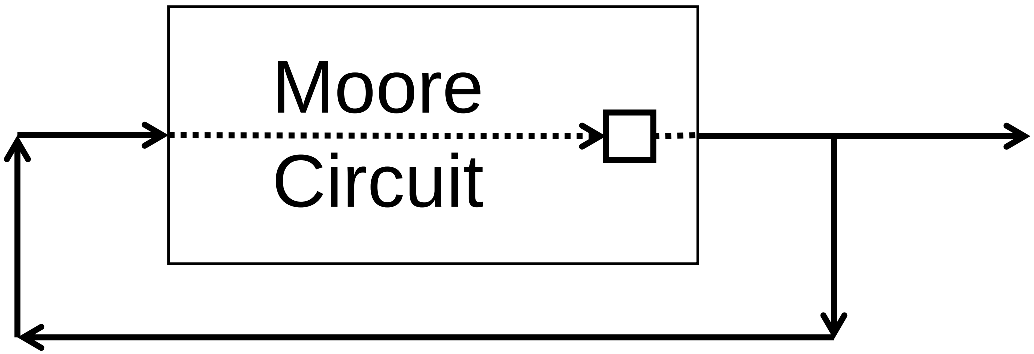
$$\forall\, x,y.\ d(f(x),f(y)) < d(x,y)$$

Theorem : a function is computable by a Moore circuit if and only if it is contracting

Feedbacks in Moore Circuits are Legal



Feedbacks in Moore Circuits are Legal



$f(x) = x$

$$\forall\, x,y.\; d(f(x),f(y)) < d(x,y)$$

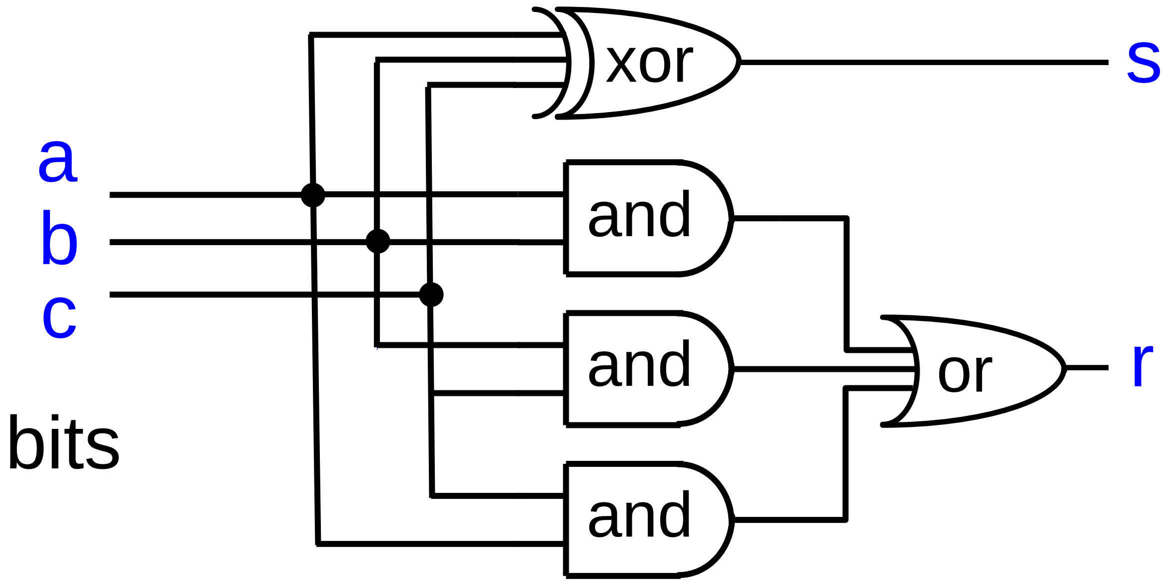
$$\Updownarrow$$

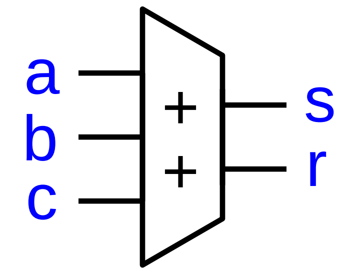
$$\forall\, x,y.\; d(f(x),f(y)) < 0{,}6\; d(x,y)$$

⟵ Lipschitz

<u>Banach theorem:</u> any Lipschitzian function over a compact set has a unique fixpoint

Full Adder

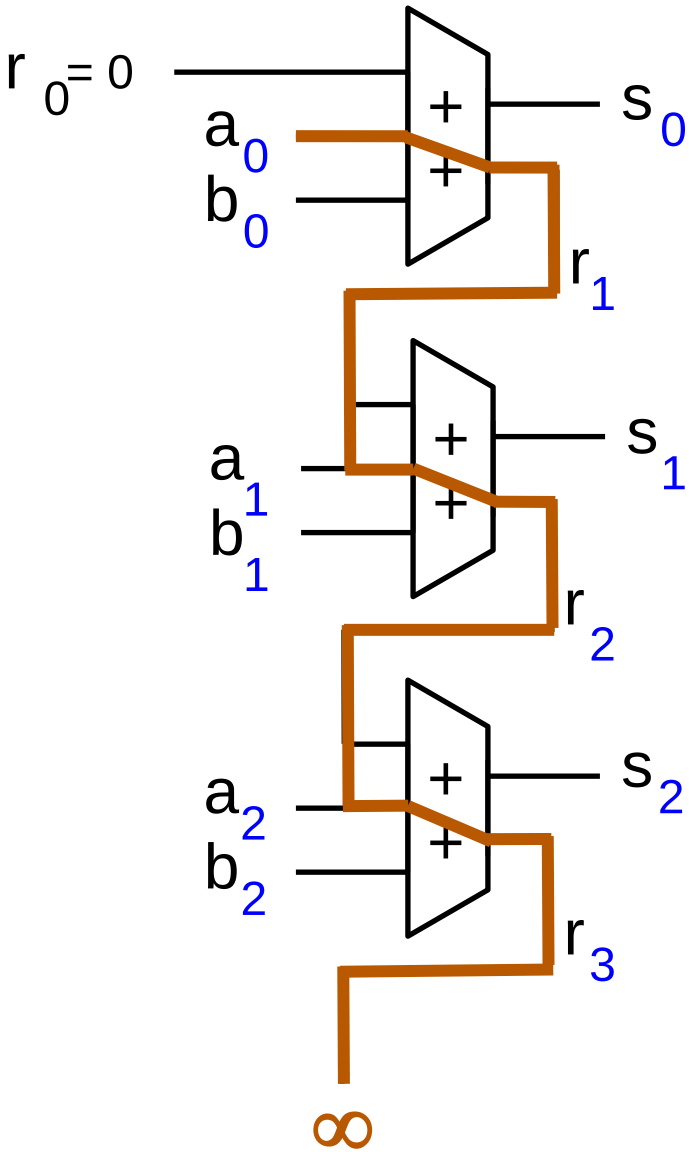




s = a xor b xor c

r = (a and b) or (b and c) or (c and a)

Addition in Space



$s = a + b$

but within infinite time!

continuity:

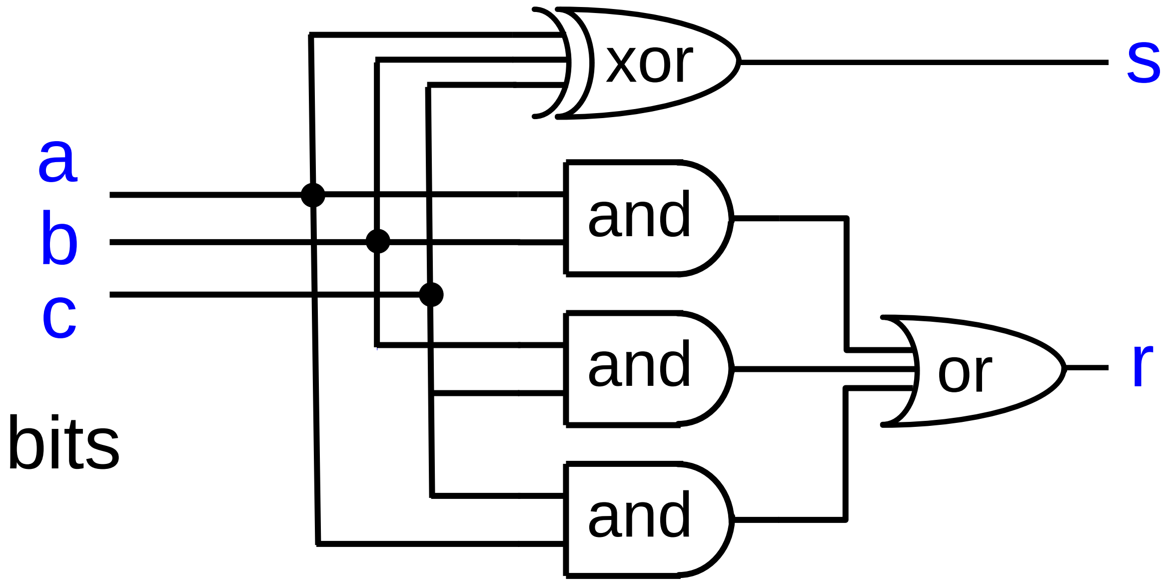
cut at n bits

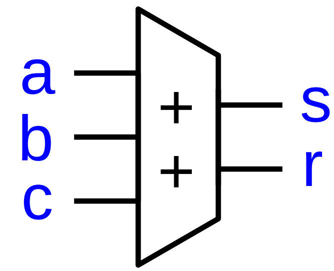
for n output bits

$x{\bullet}2^n = x \bmod 2^n$

$s{\bullet}2^{n+1} = a{\bullet}2^n + b{\bullet}2^n$

Full Adder



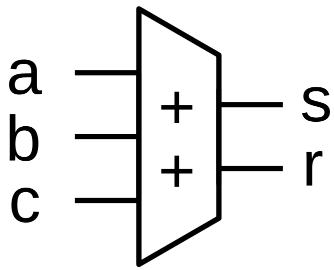


a + b + c = s + 2r

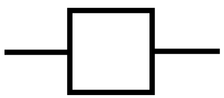
s = a xor b xor c

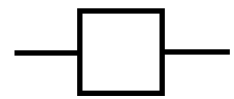
r = (a and b) or (b and c) or (c and a)

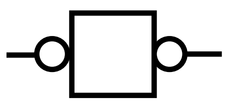
Basic 2-adic Operators

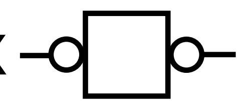


$a + b + c = s + 2r$

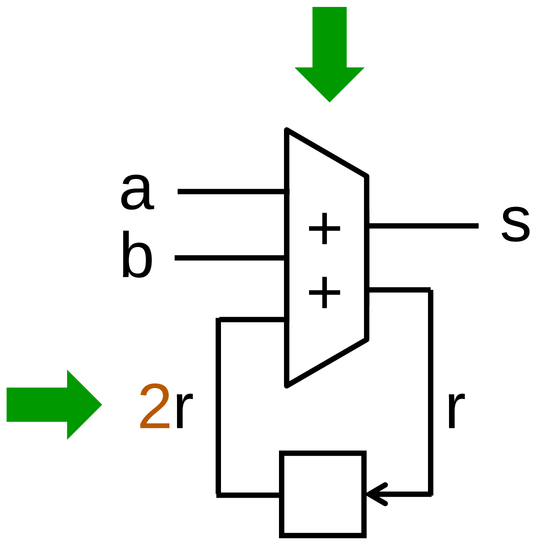
${}_2x_0x_1\ldots x_n\ldots$  ${}_2 0x_0x_1\ldots x_n\ldots$

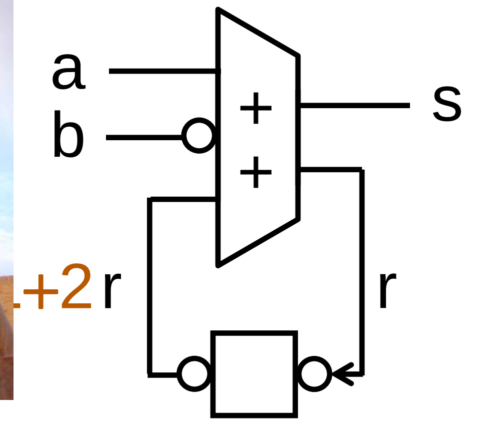
x  $2x$

${}_2x_0x_1\ldots x_n\ldots$  ${}_2 1x_0x_1\ldots x_n\ldots$

x  $1+2x$

Addition and Subtraction Over Time





$$a+b+\cancel{2}r = s+\cancel{2}r$$

$$\boxed{s = a+b}$$

same equation
as over space!

$$a+\neg b+1+\cancel{2}r = s+\cancel{2}r$$

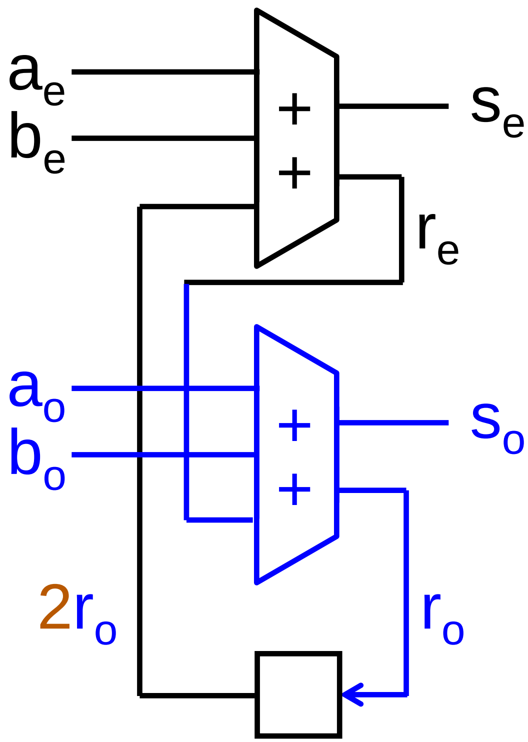
$$b+\neg b = -1$$

$$\neg b+1 = -b$$

$$a-b = s$$

$$\boxed{s = a-b}$$

Mixed Space / Time Addition



$$x \odot y = {}_2x_0y_0x_1y_1\ldots$$

$$a = a_e \odot a_o$$

$$b = b_e \odot b_o$$

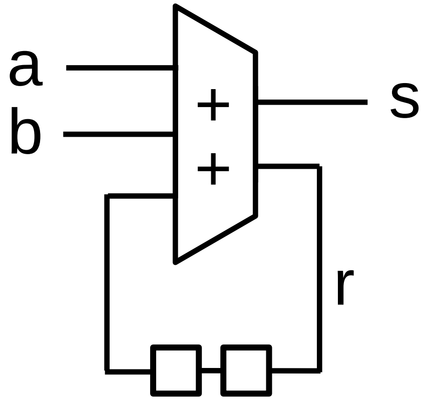
$$s = s_e \odot s_o$$

$$s = a + b$$

still the same equation !

Same source code
for any space / time tradeoff

Stereo Addition

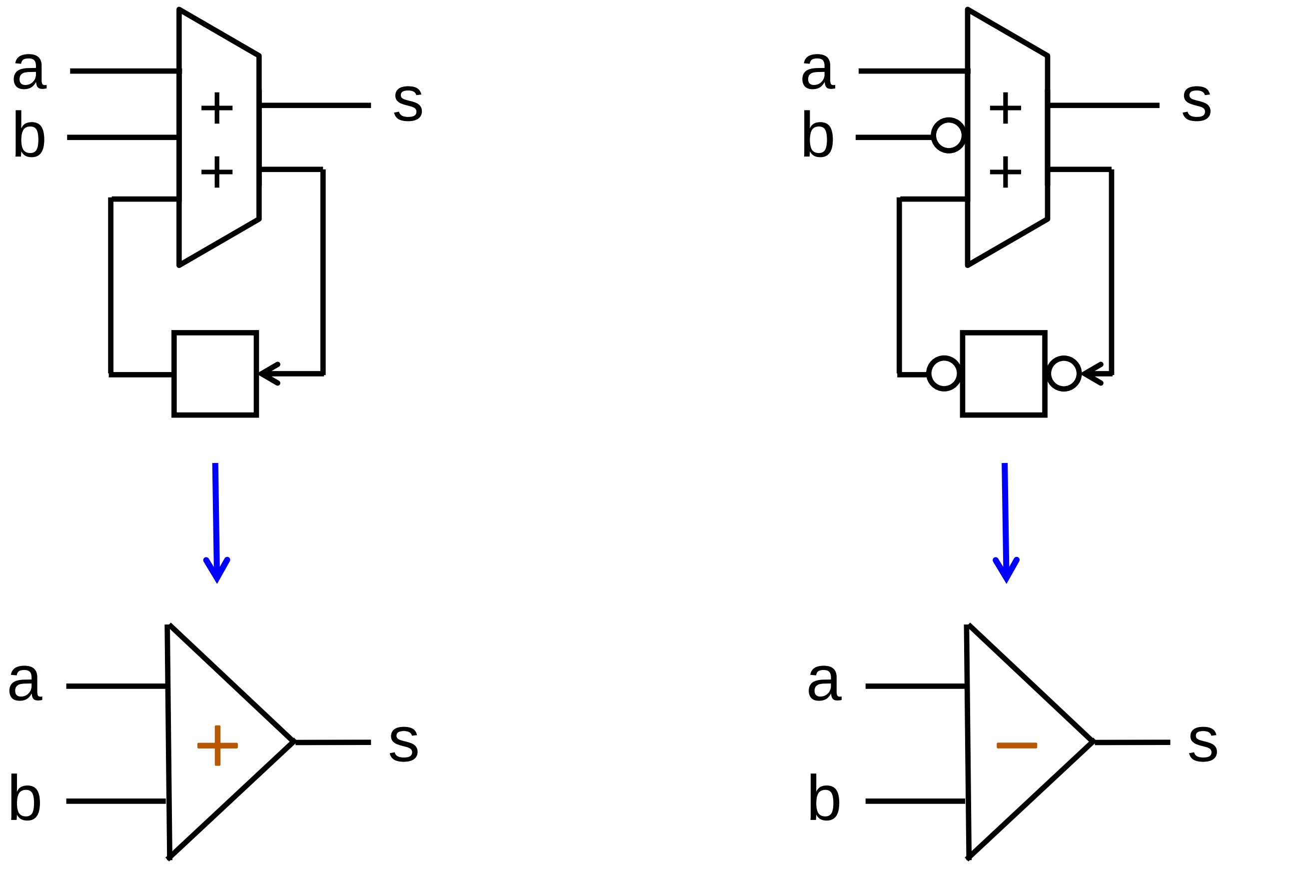


stereo
adder

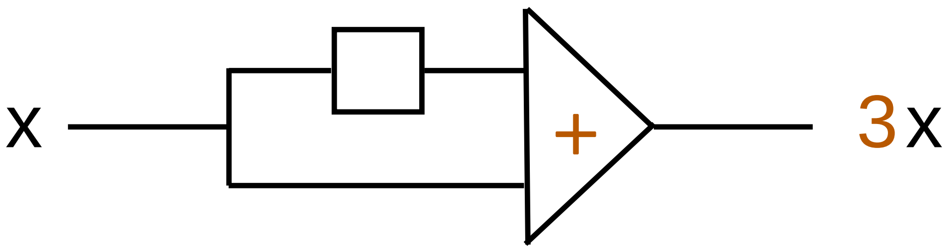
$$s_e \odot s_o = (a_e + b_e) \odot (a_o + b_o)$$

Alternates 2 additions over time (even / odd bits)
stereo = left / right channels

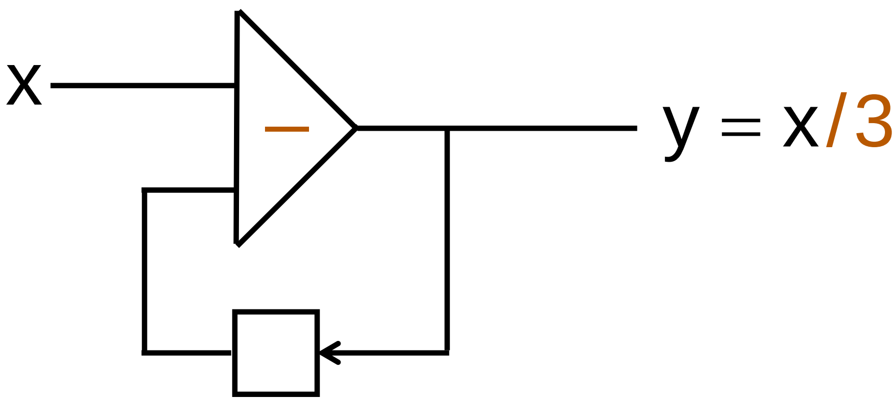
Addition and Subtraction Over Time



Multiplication and division by a constant



proof : x+2x = 3x



proof : y= x−2y

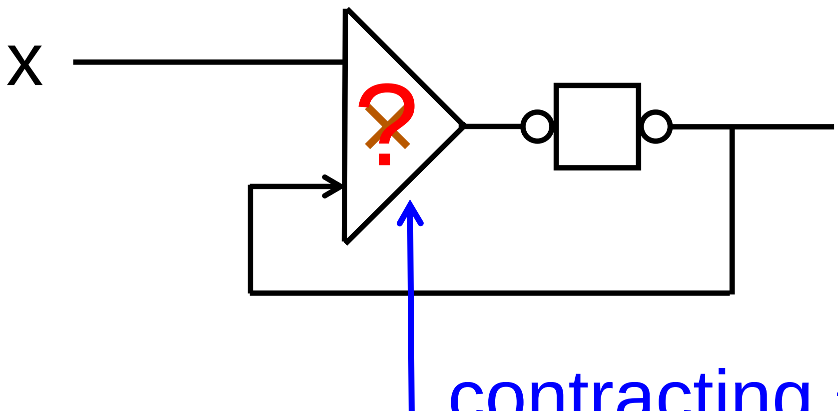
division only
by odd integers!

Quasi-inverse

$$y = 1/(1-2x)$$

$$y - 2xy = 1$$

$$y = 1 + 2xy$$

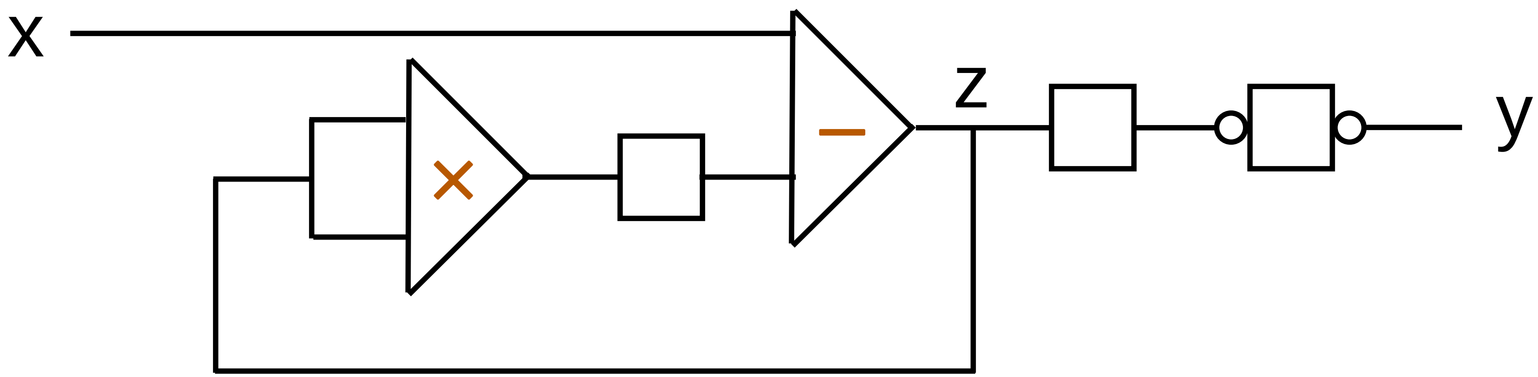


$y = 1 + 2xy = 1/(1-2x)$

contracting ⇒ synchronous
but infinite memory
(cf. SDD construction)

Quasi Square Root

$$y = \sqrt{1+8x}$$



$$y = 1+4z$$

$$y^2 = 1+8z+16z^2$$

$$z = x-2z^2$$

$$y^2 = 1+8x-\cancel{16}z^2+\cancel{16}z^2$$

... but tells us nothing about bit transformations!

Spatio-Temporal Decomposition of f Synchronous

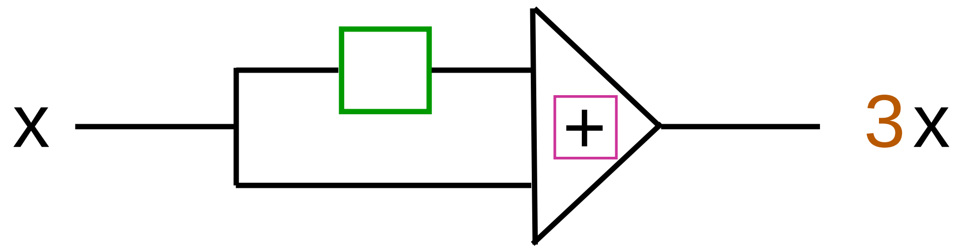
$f \cdot 0$ = first bit output by f for inputs 0...
$f \cdot 1$ = ... 1...

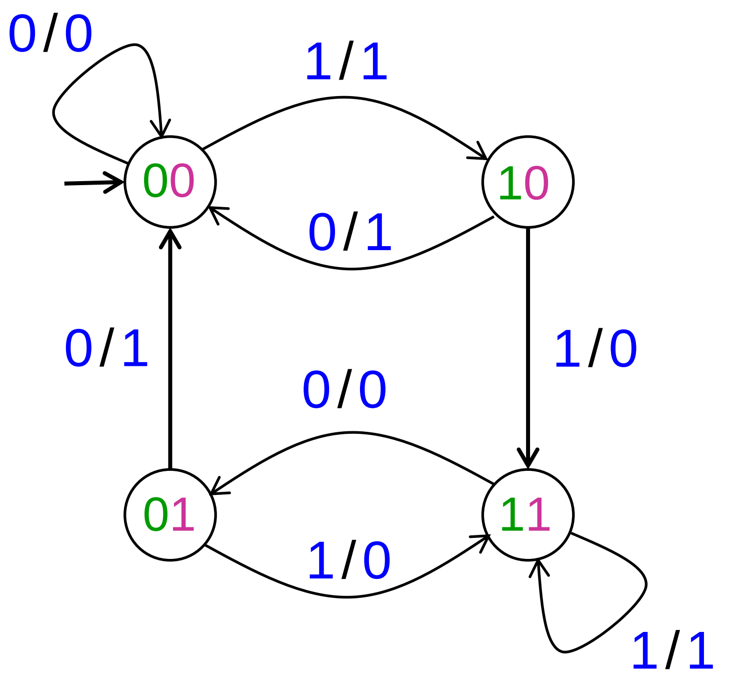
$f \cdot w$ = last bit output by f for the finite word w

f^0 = 0-predictor : $f^0 \cdot w = f \cdot (w0)$ for any word w
f^1 = 1-predictor : $f^1 \cdot w = f \cdot (w1)$

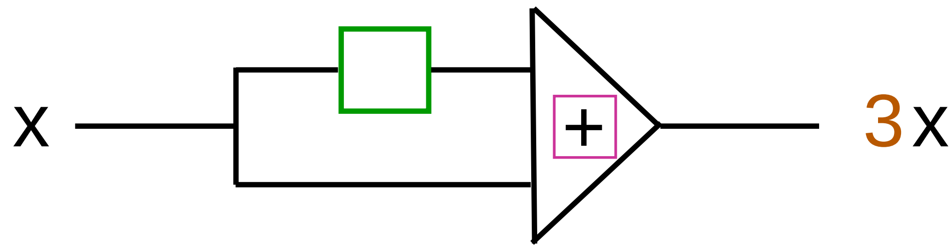
f^u = u-predictor : $f^u \cdot w = f \cdot (wu)$ for any words w, u

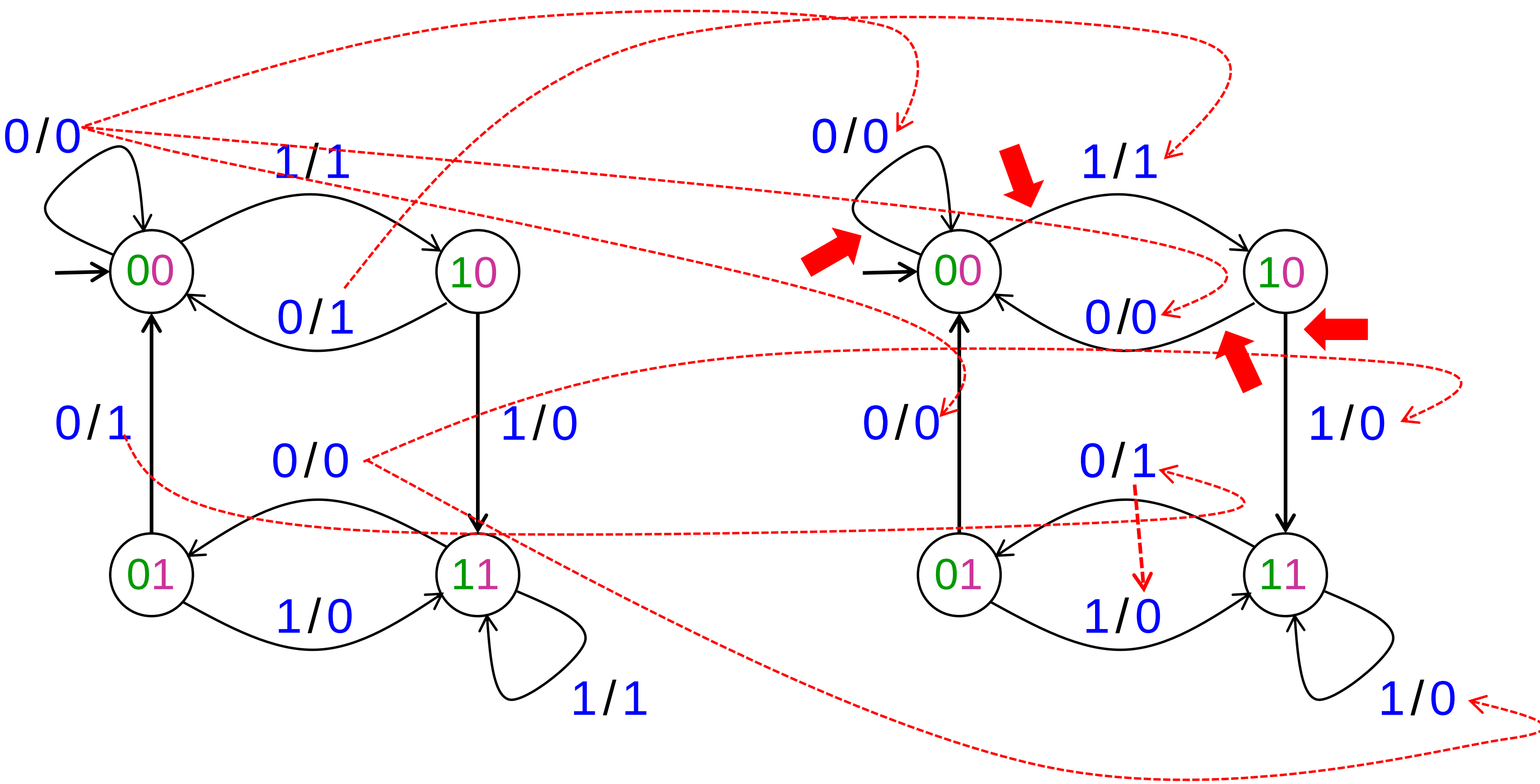
Automaton of x → 3x



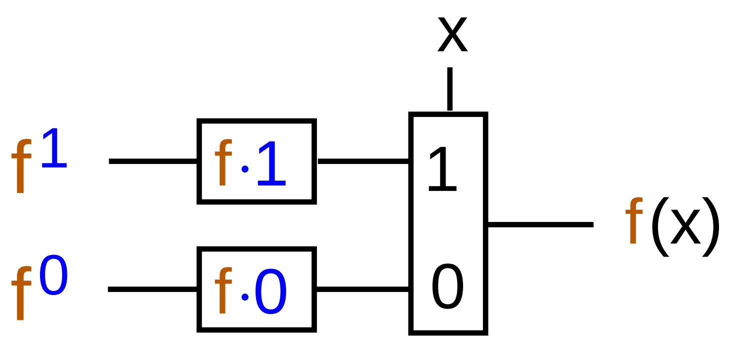


Predictor 0 of x → 3x



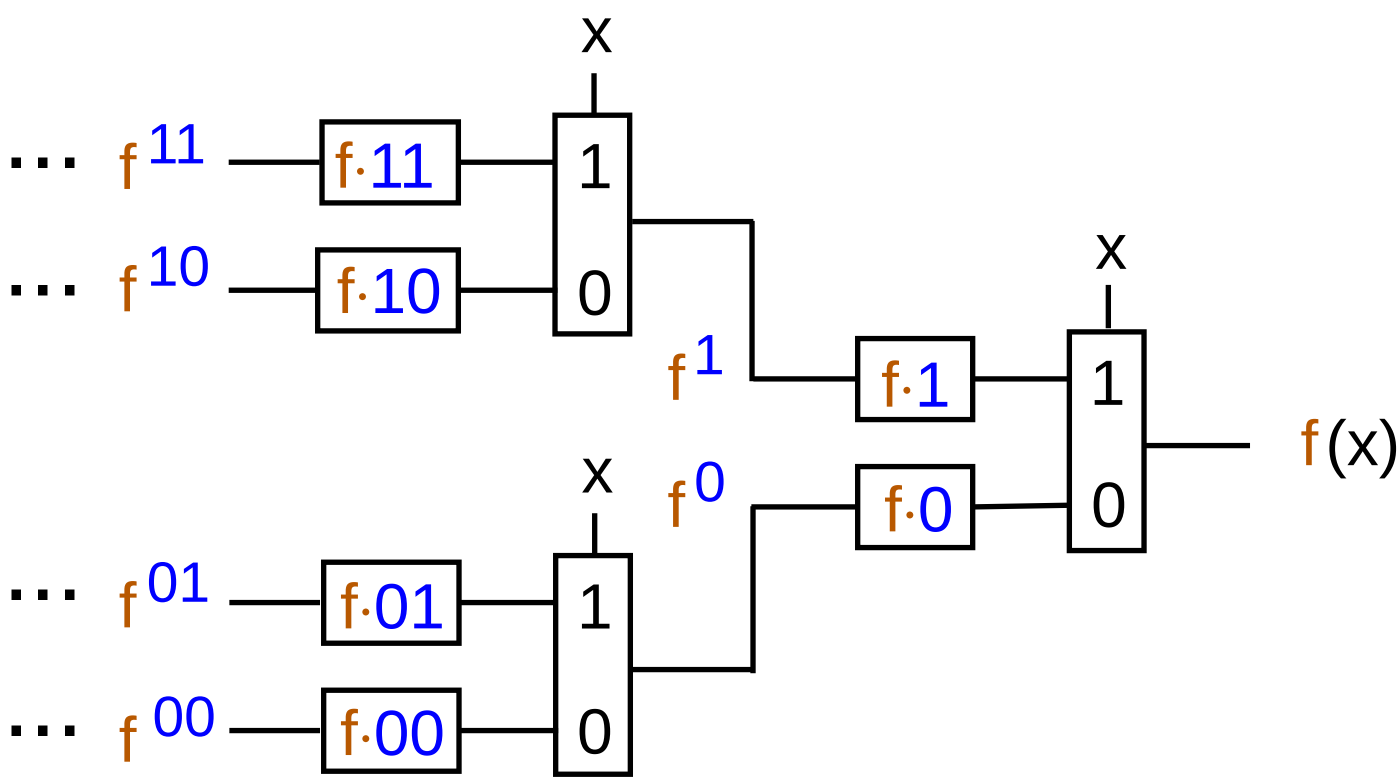


SDD Decomposition Step



$$f(x) = mux(x,\ f{\cdot}1{+}2f^{1}(x),\ f{\cdot}0{+}2f^{0}(x))$$

SDD Space/Time Normal Form of *f*

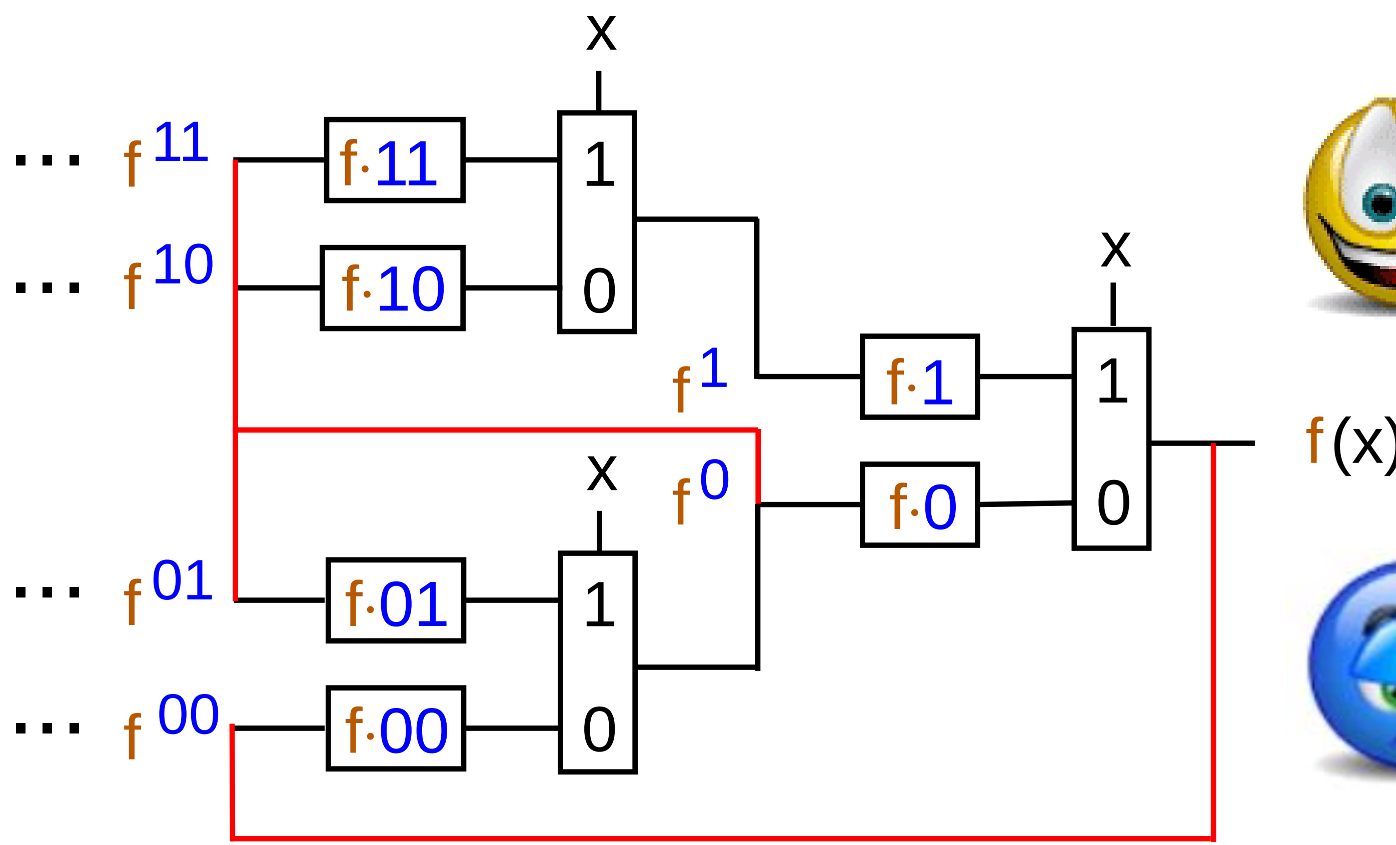


Truth-table in space and time

ultra-fast : critical path = one mux

Half of the bits disappear at each cycle

Shared SDD of f with Finite Memory



f finite memory ⇒ finitely many distinct predictors f^u

f with n registers ⇒ SDD(f) may have 2^{2^n} registers

From continuous functions to circuits

f continuous but not synchronous:

- over space : trivial if infinite space
- over time : expand the time

2-adic number : < value, validity >

0	0	1	0	1	1	0	1	1	1	0	0	...
0	1	1	0	0	1	1	0	0	1	1	0	...
	0	1			1	0			1			

<u>Theorem :</u> every continuous function can be realized by a synchronous circuit with validity

Trace of a Synchronous Function

$Tr(f) = {}_2 f{\cdot}0\ f{\cdot}1\ f{\cdot}00\ f{\cdot}01\ f{\cdot}10\ f{\cdot}11\ \ f{\cdot}000\ f{\cdot}001 \ldots$

$= f{\cdot}0 + 2\, f{\cdot}1 + 4\, (Tr(f^0) \odot Tr(f^1))$

Application of a trace Tr(f) to an argument x is continuous ⇒ λ–calculus ?

Power series over Z/2Z : $S(f) = \Sigma_n\, Tr(f)_n\, z^n$

<u>Theorem :</u> $f : {}_2Z \rightarrow {}_2Z$ synchronous has finite memory iff S(f) is algebraic over Z/2Z

From Synchronous Traces to Transcendental Numbers

<u>Theorem</u> (Van der Porten) : if f has finite memory, them the real number

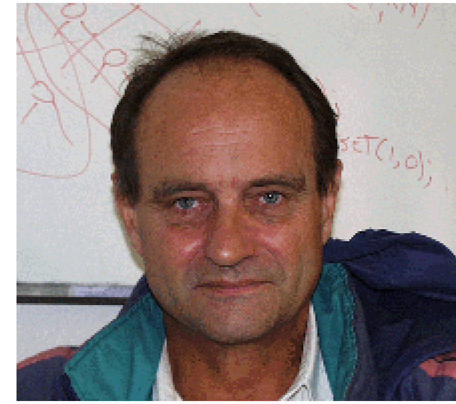
0, f·0 f·1 f·00 f·01 f·10 f·11 f·000 f·001 ...

is either rational or transcendantal

Almost any finite automaton generates a transcendental number!

Automatic Sequences: Theory, Applications, Generalizations
Jean-Paul Allouche et Jeffrey Shallit
Cambridge University Press (21 juillet 2003)

Conclusion



Thanks to Jean Vuillemin

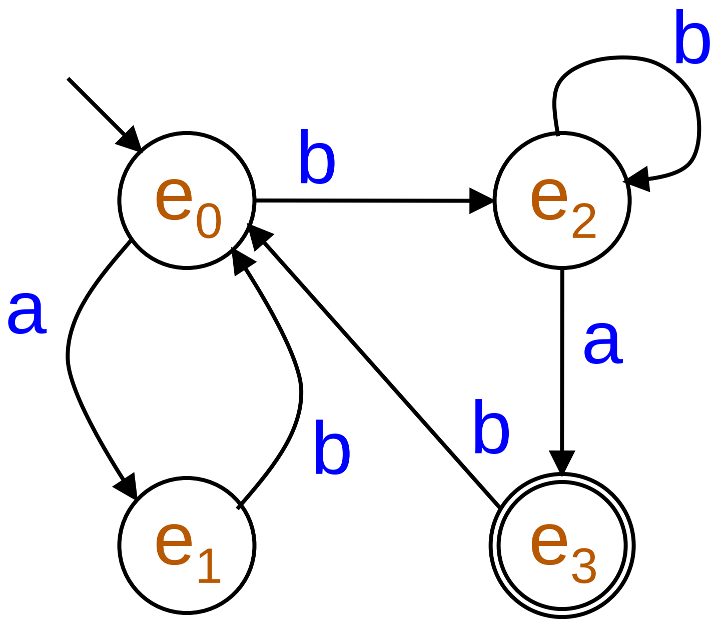
- 2-adic numbers are the good model of arithmetic synchronous circuits (only?)
- the 2-adic metric, continuity, and synchronism are fundamental notions to explore further
- The structure of the predictor space is largely unknown
- The relation between continuous functions and validity-circuits remains to be studied (λ–calculus?)

Agenda

1. 2-adic numbers and space / time exchange in synchronous circuits
2. Never determinize non-deterministic automata !
3. Use hierarchical automata for another exponential gain in space and timing optimization

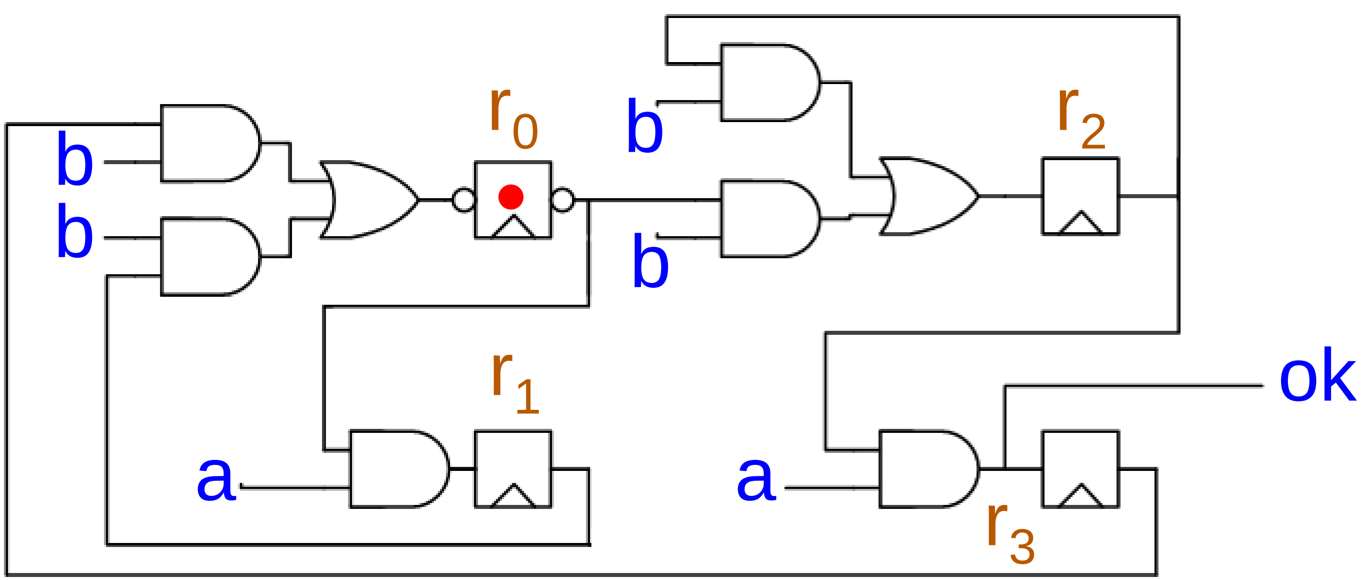
From Deterministic Automata to Circuits

(ab+b)*ba



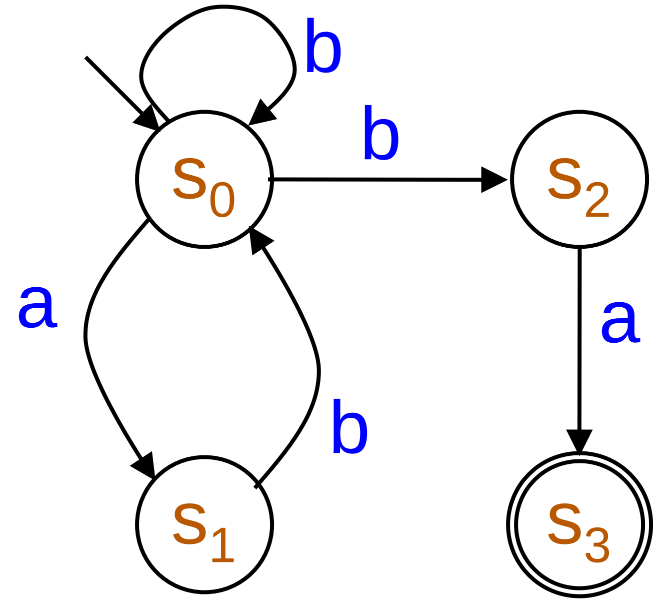
1-hot encoding
(only one r_i to 1)

size explosion!

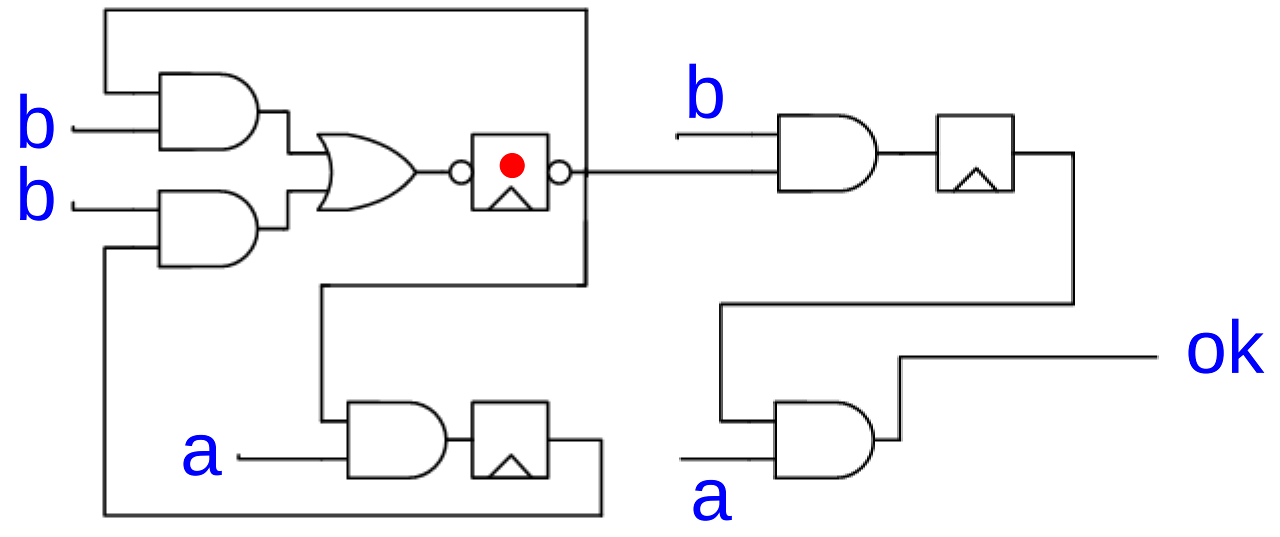


The Non-Deterministic Case

(ab+b)*ba

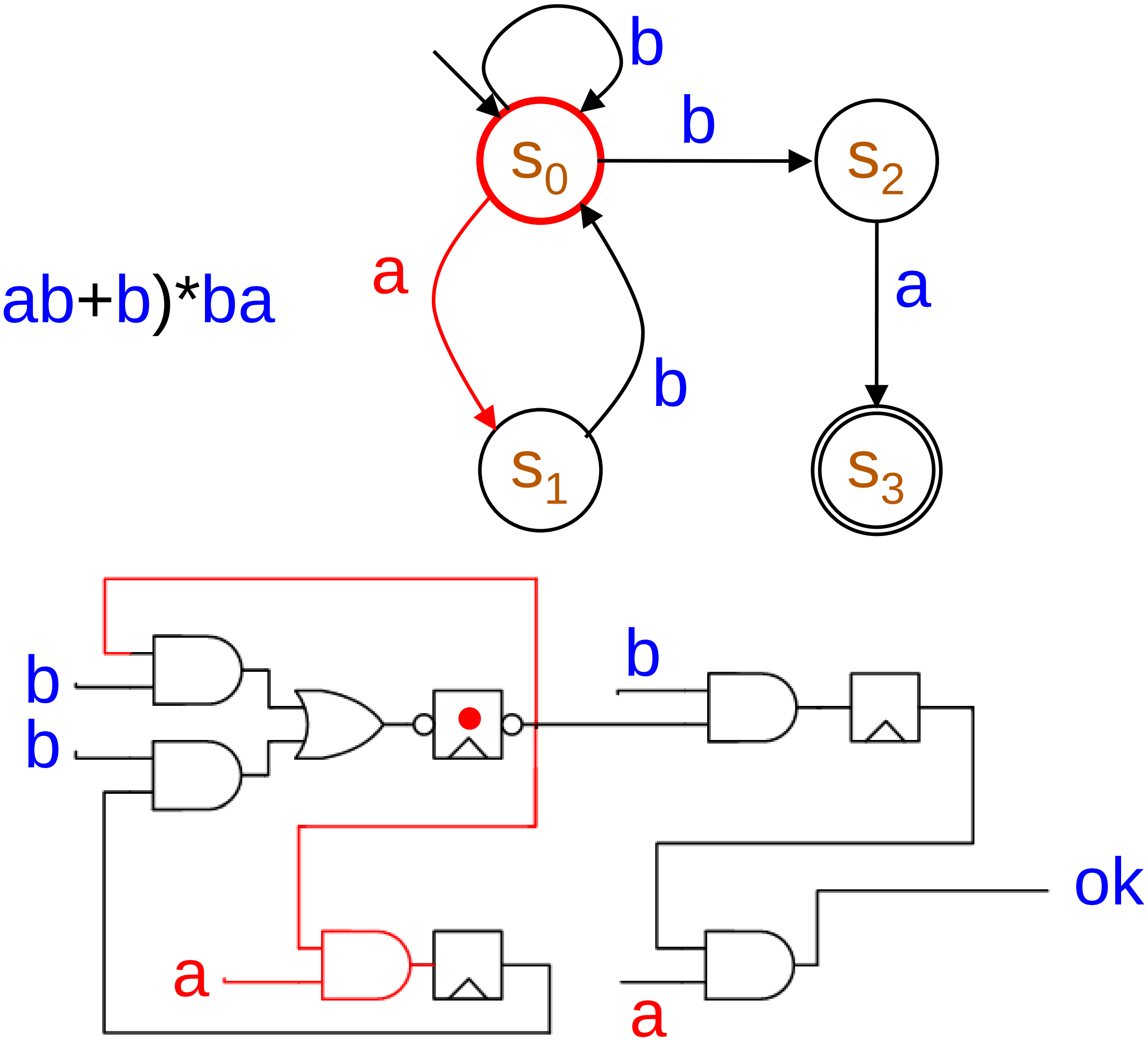


no size explosion
⇒ much better!



Electrical Subset Construction

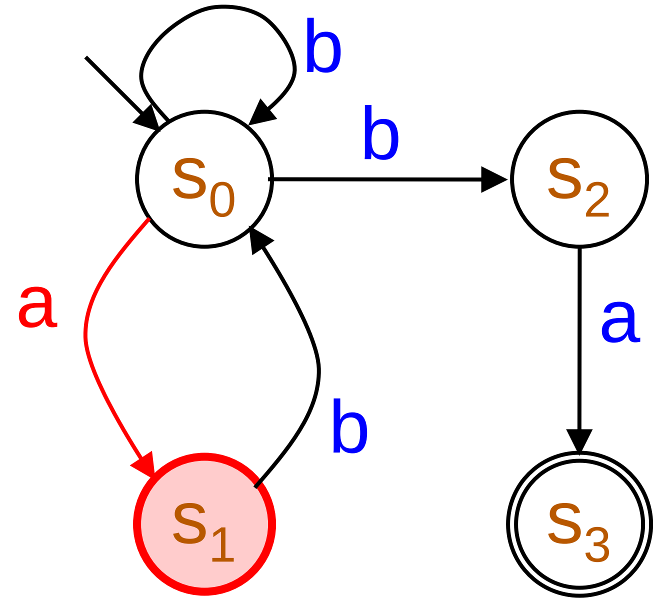
(ab+b)*ba



a

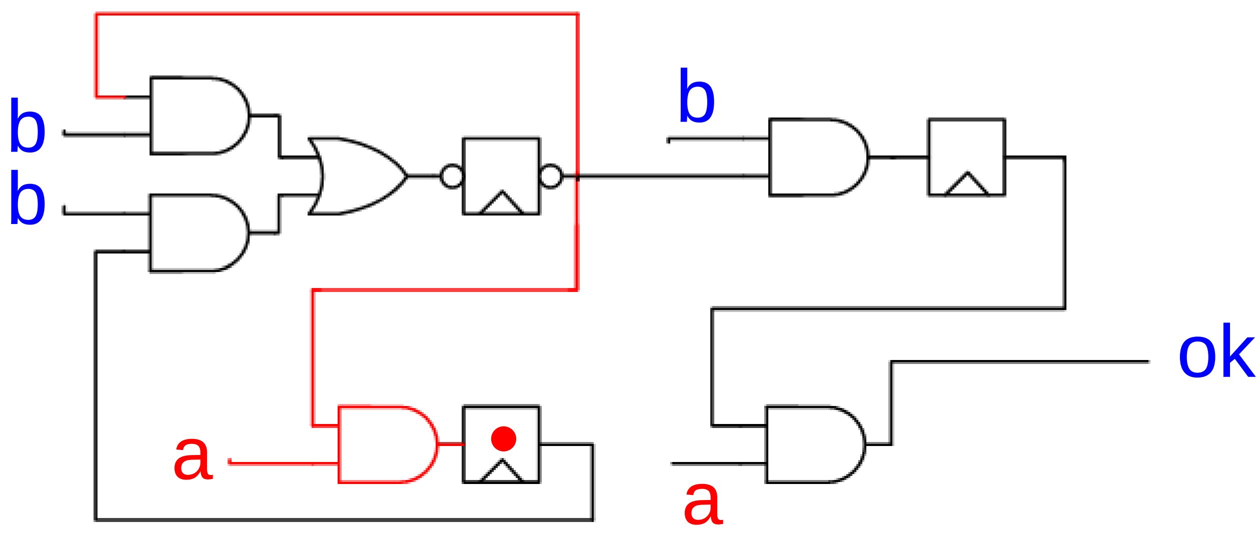
Electrical Subset Construction

(ab+b)*ba



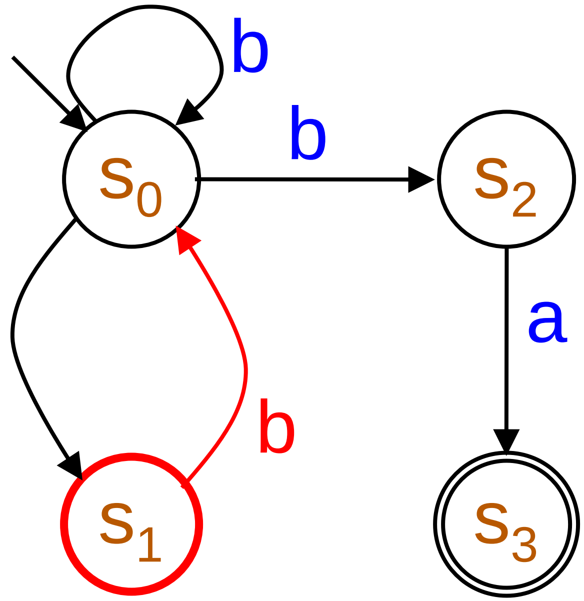
tick!

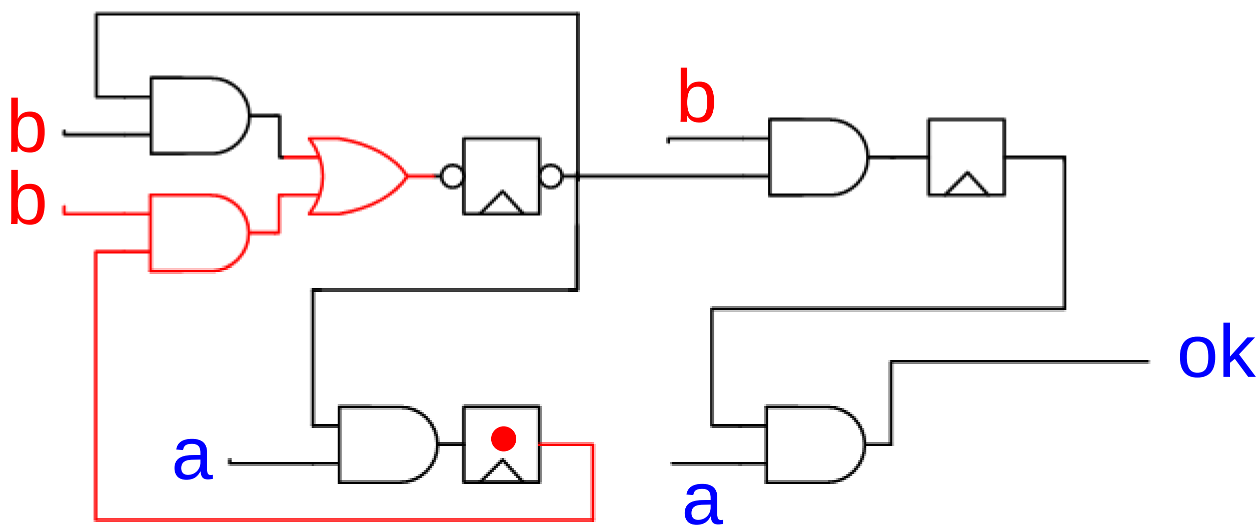
a



Electrical Subset Construction

(ab+b)*ba

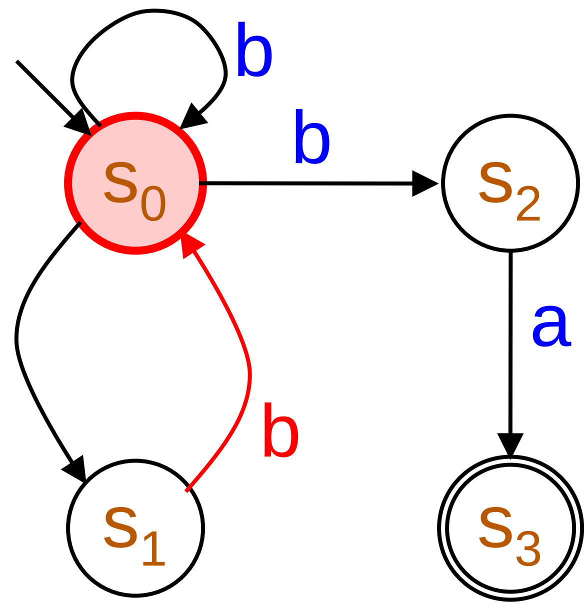




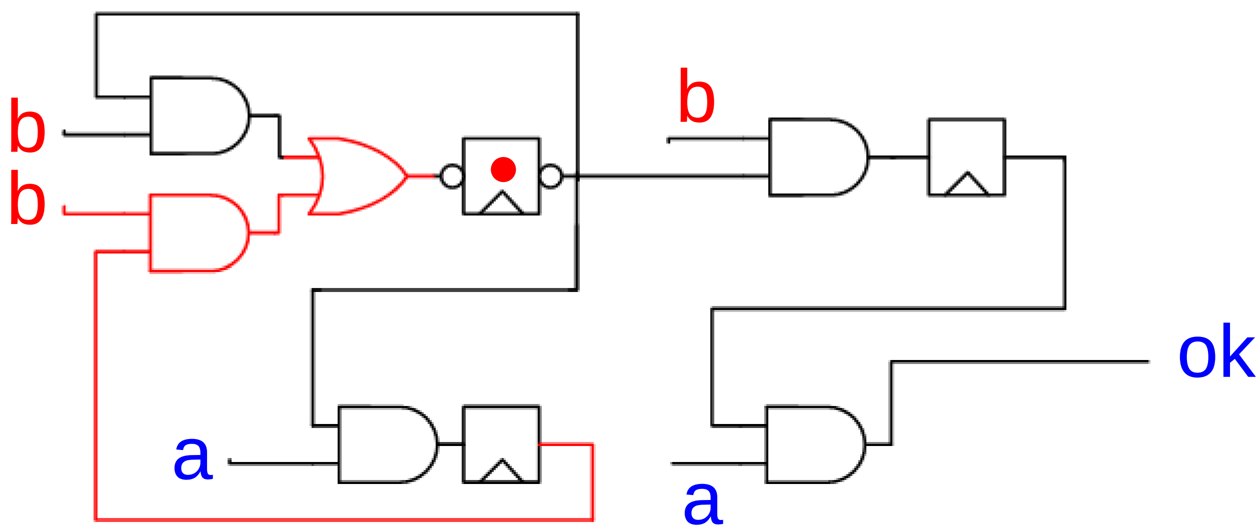
ab

Electrical Subset Construction

(ab+b)*ba



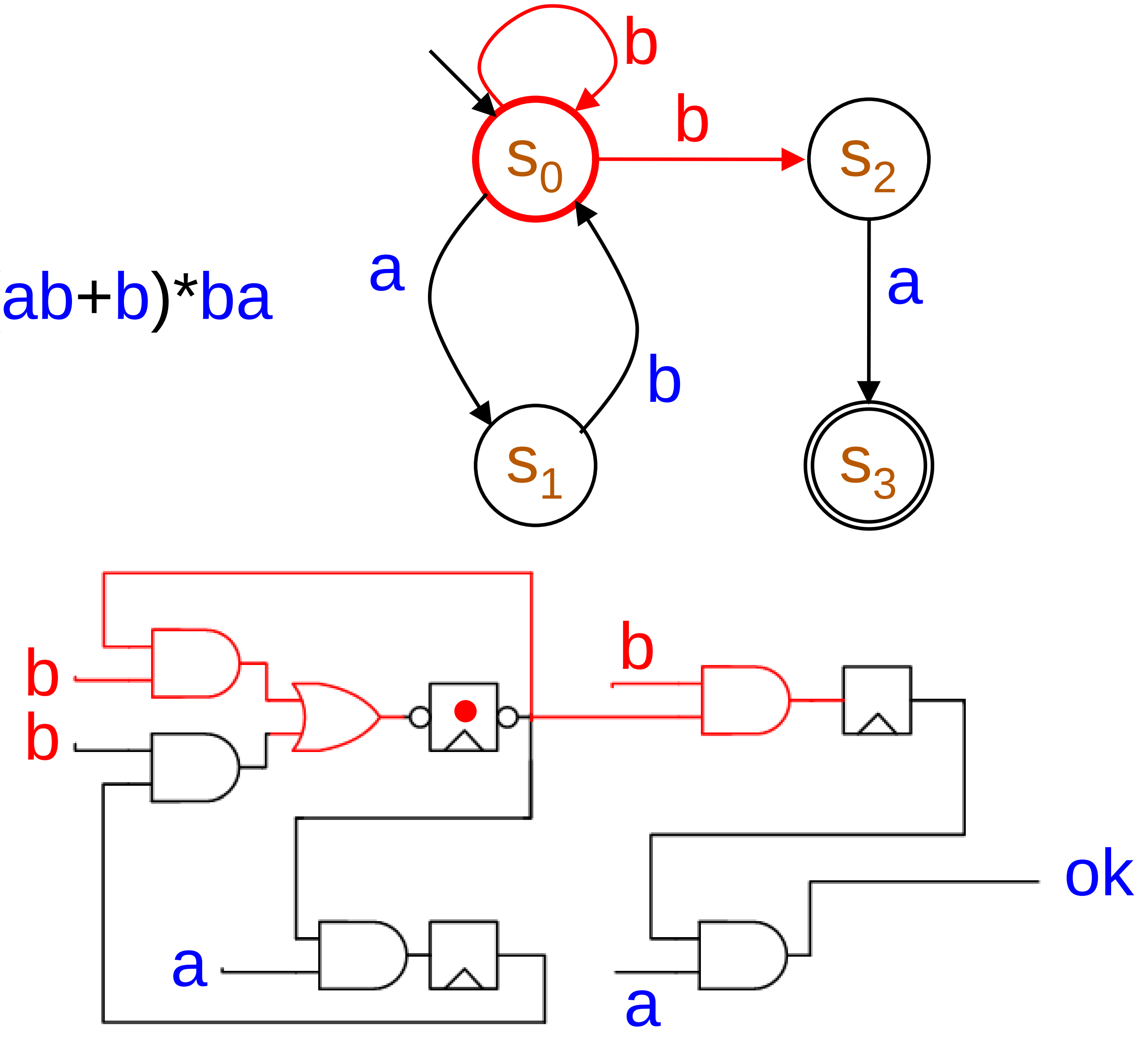
tick!



ab

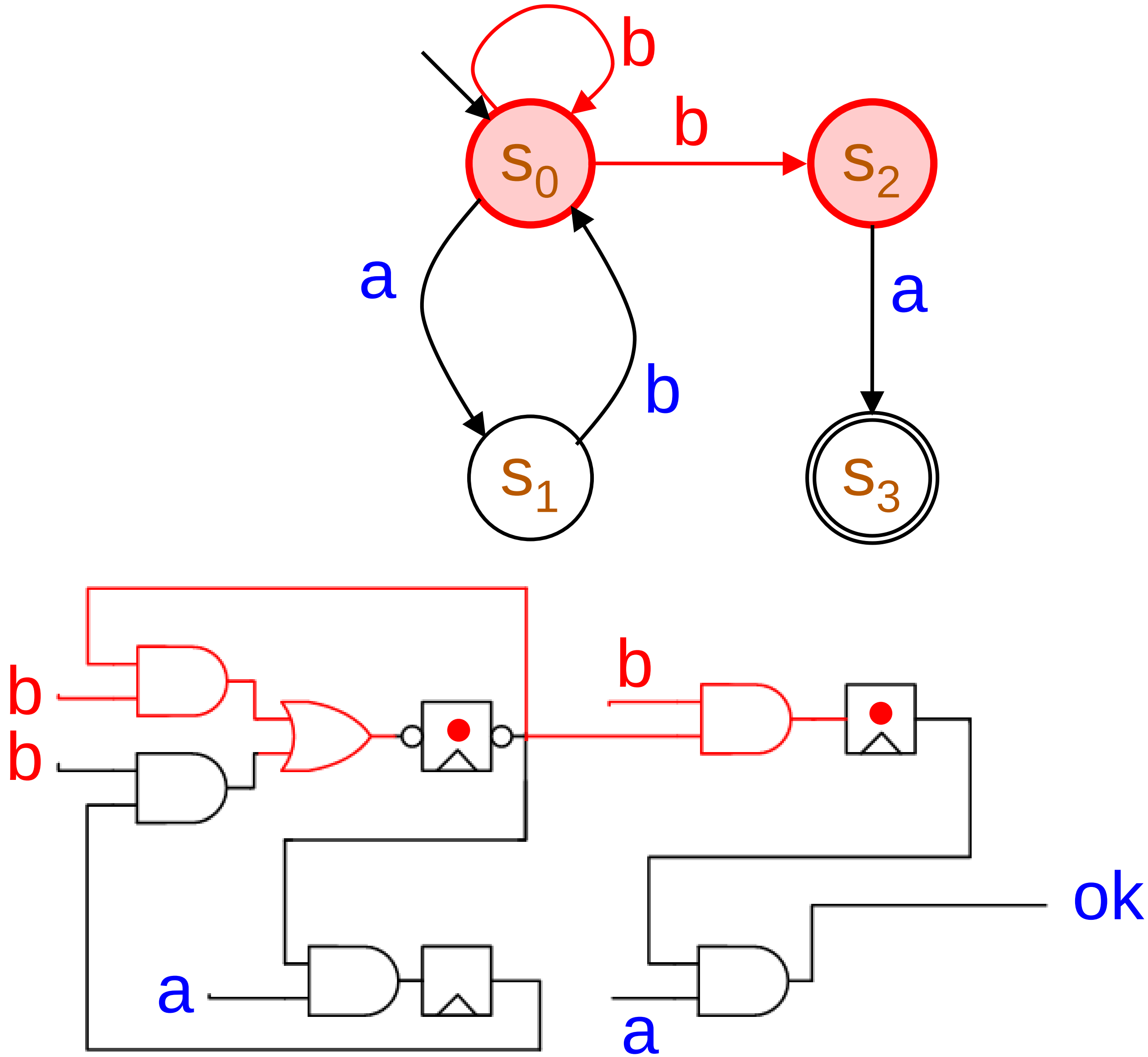
Electrical Subset Construction

(ab+b)*ba



abb

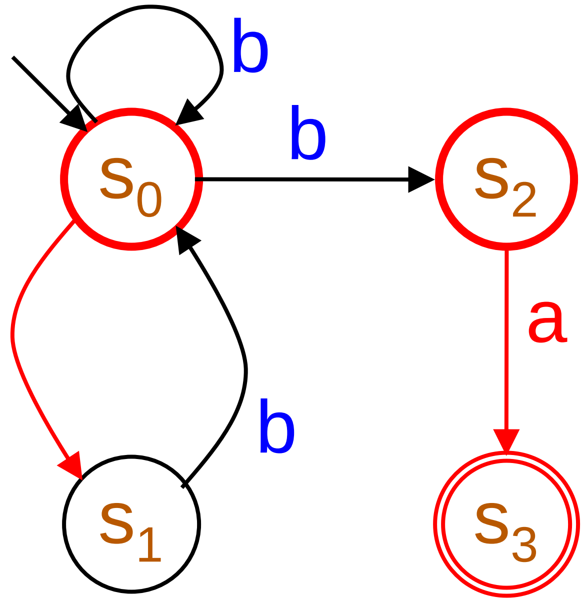
Electrical Subset Construction



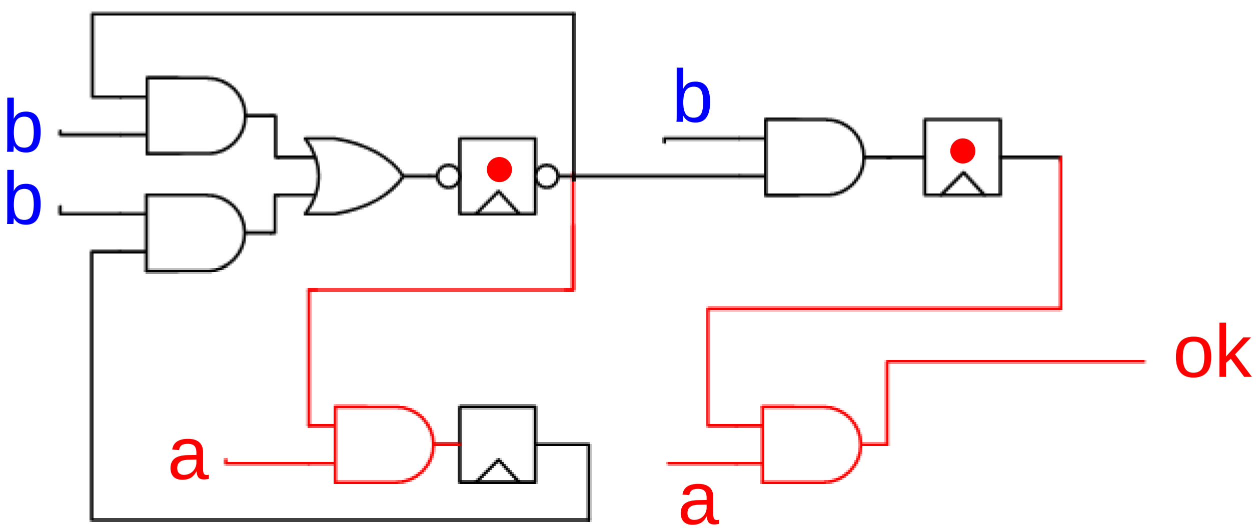
tick!

abb

Electrical Subset Construction



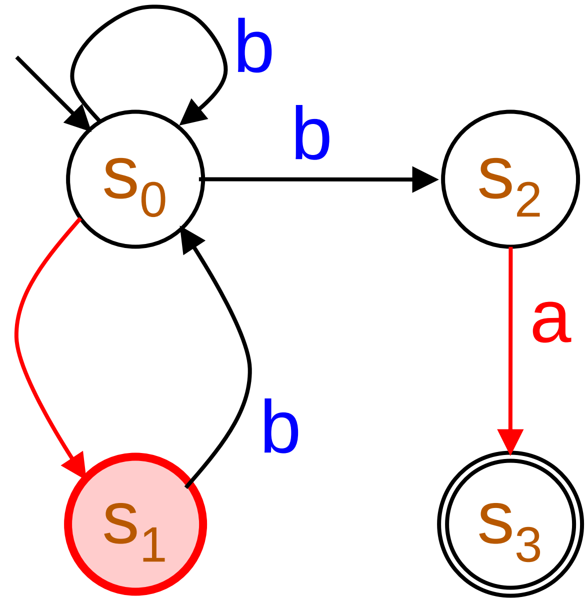
(ab+b)*ba



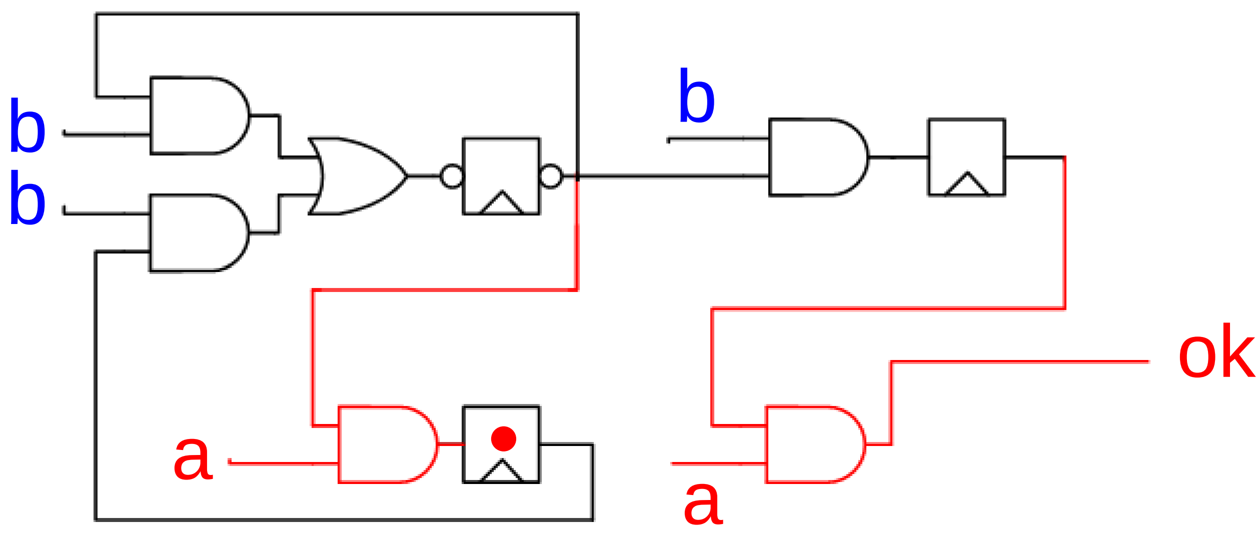
abba

Electrical Subset Construction

(ab+b)*ba

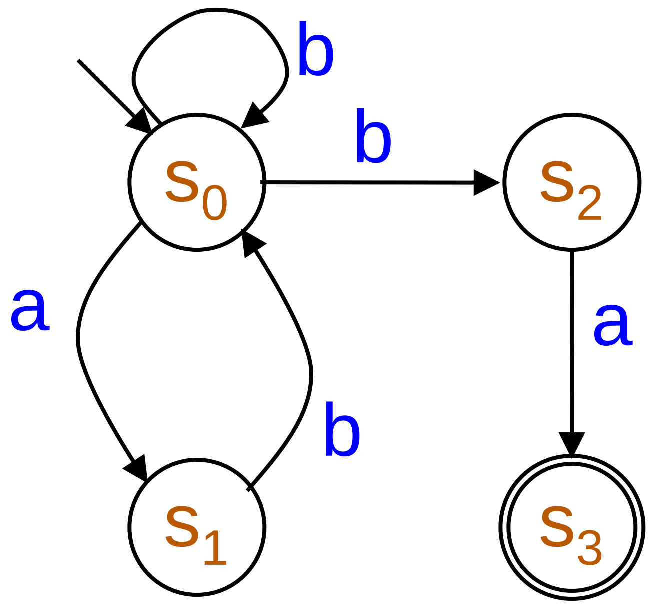


tick!



abba

Esterel v7 implementation




```
module Autom :
input a, b ;
output ok;
local {r0, r1, r2, r3} : reg;
refine r0 : init true;
sustain {
   next r0 <= (r0 or r1) and b,
   next r1 <= r0 and a,
   next r2 <= r0 and b,
   next r3 <= r2 and a,
   ok <= next r3 }
end module
```

Compiled into C, C++, VHDL, Verilog, etc.

Agenda

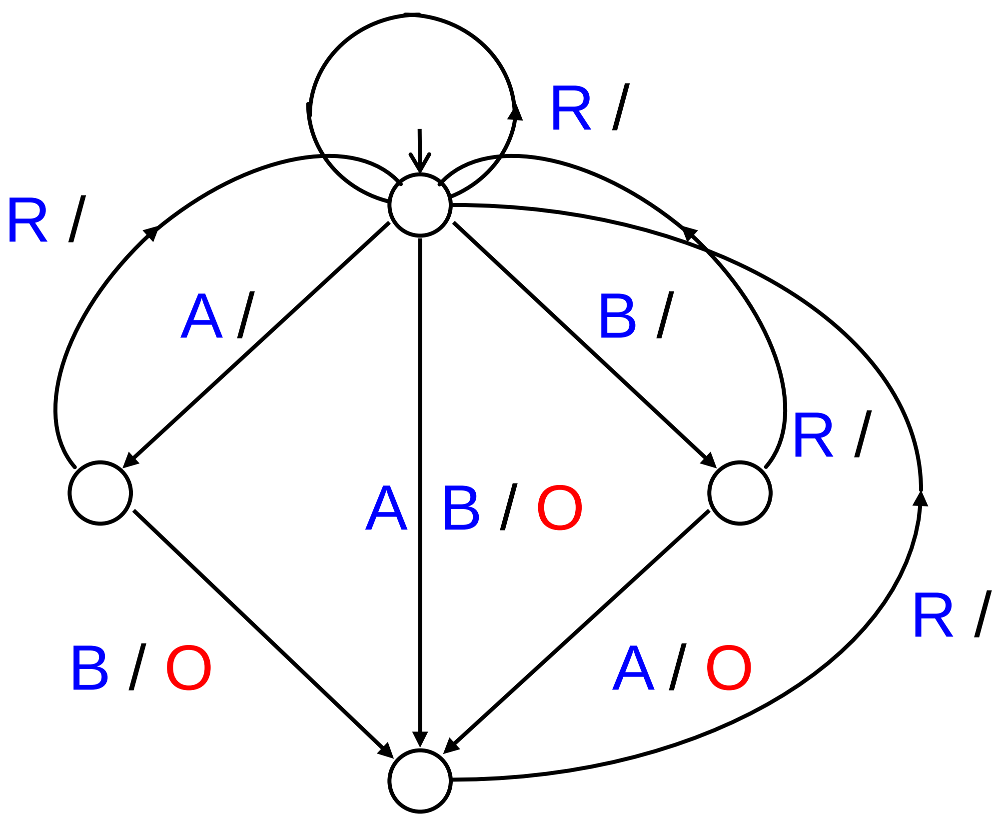
1. 2-adic numbers and space / time exchange in synchronous circuits

2. Never determinize non-deterministic automata !

3. Use hierarchical automata for another exponential gain in space and timing optimization

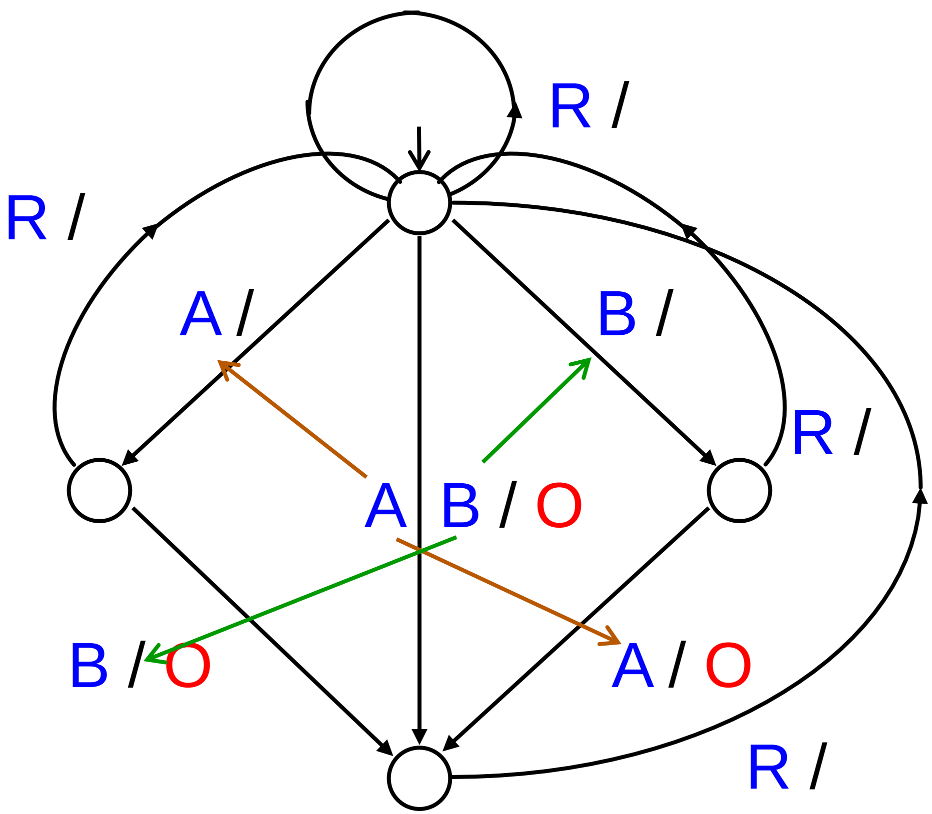
The ABRO Example

Emit O as soon as A and B have arrived
Reset behavior each time R is received



Memory write
R : Request
A : Address
B : Data
O : Write

Esterel : Linear Program

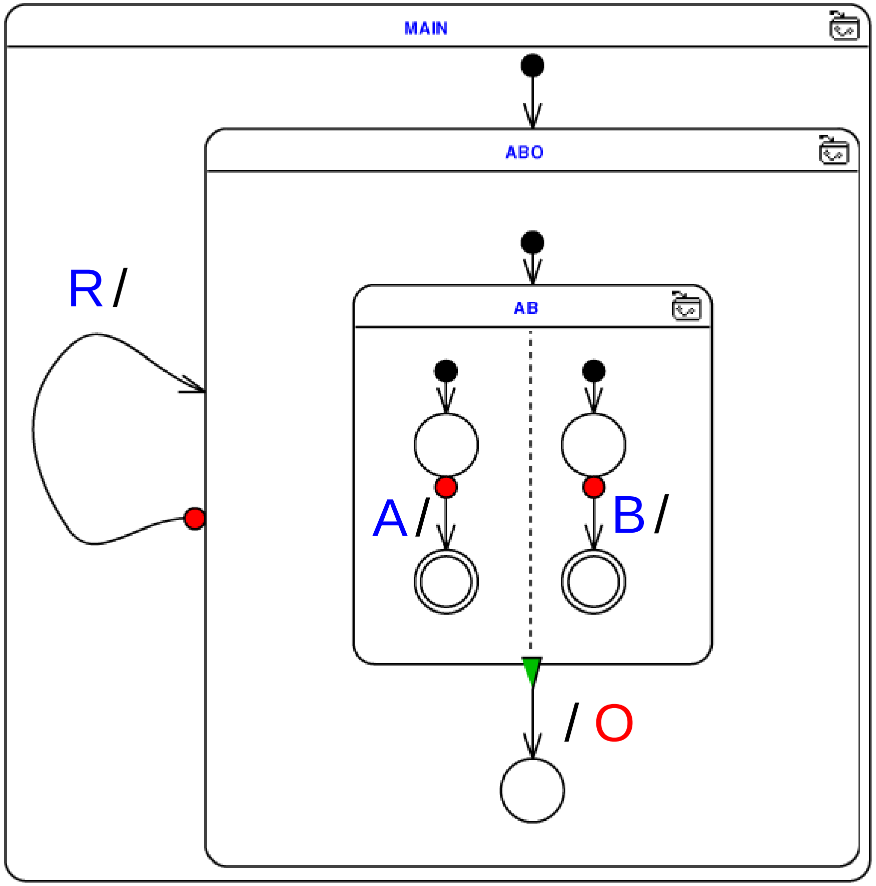



```
loop
   abort
      { await A || await B };
      emit O ;
      halt
   when R
end loop
```

copies = residuals !

Esterel = sharing residuals

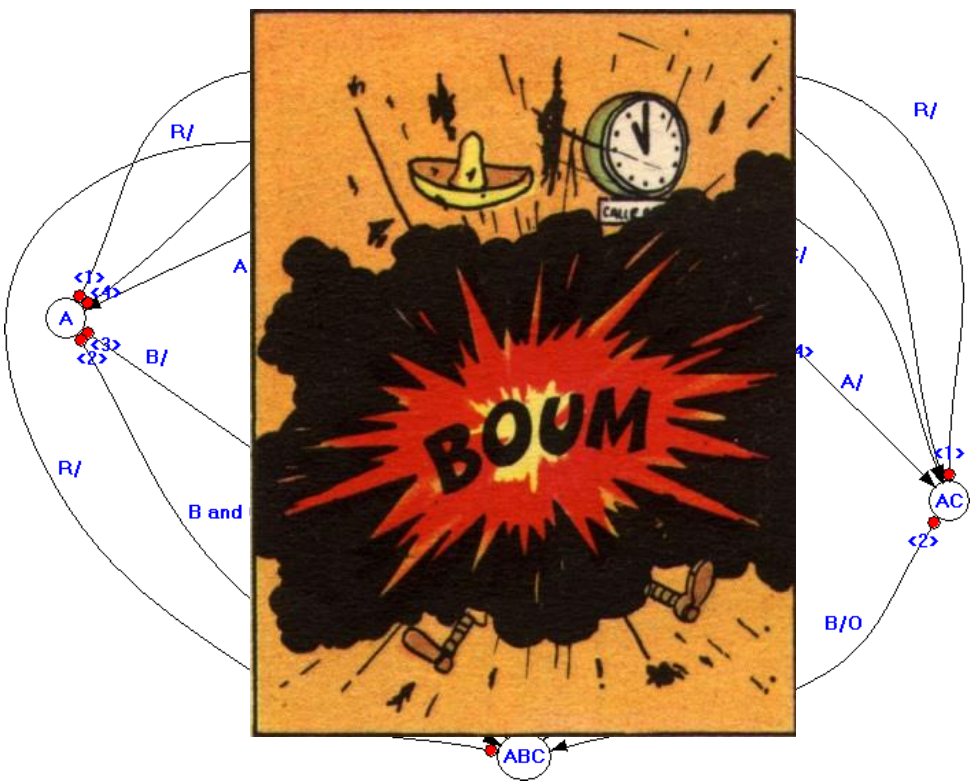
SyncCharts (Charles André)

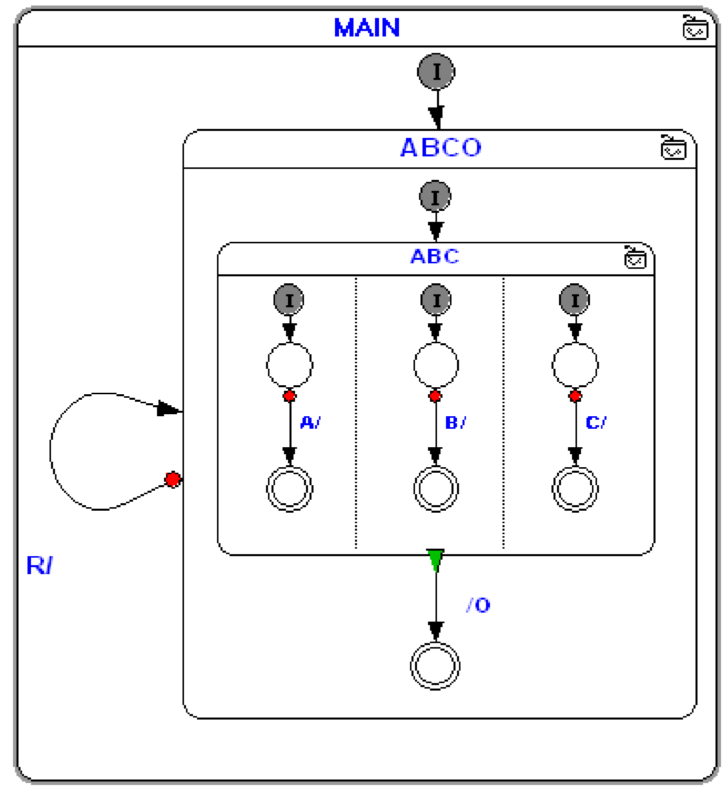


Hierarchical synchronous
concurrent automata
(Synchronous Statecharts)

```
loop
  abort
     { await A || await B };
     emit O ;
     halt
  when R
end loop
```

The ABCRO example



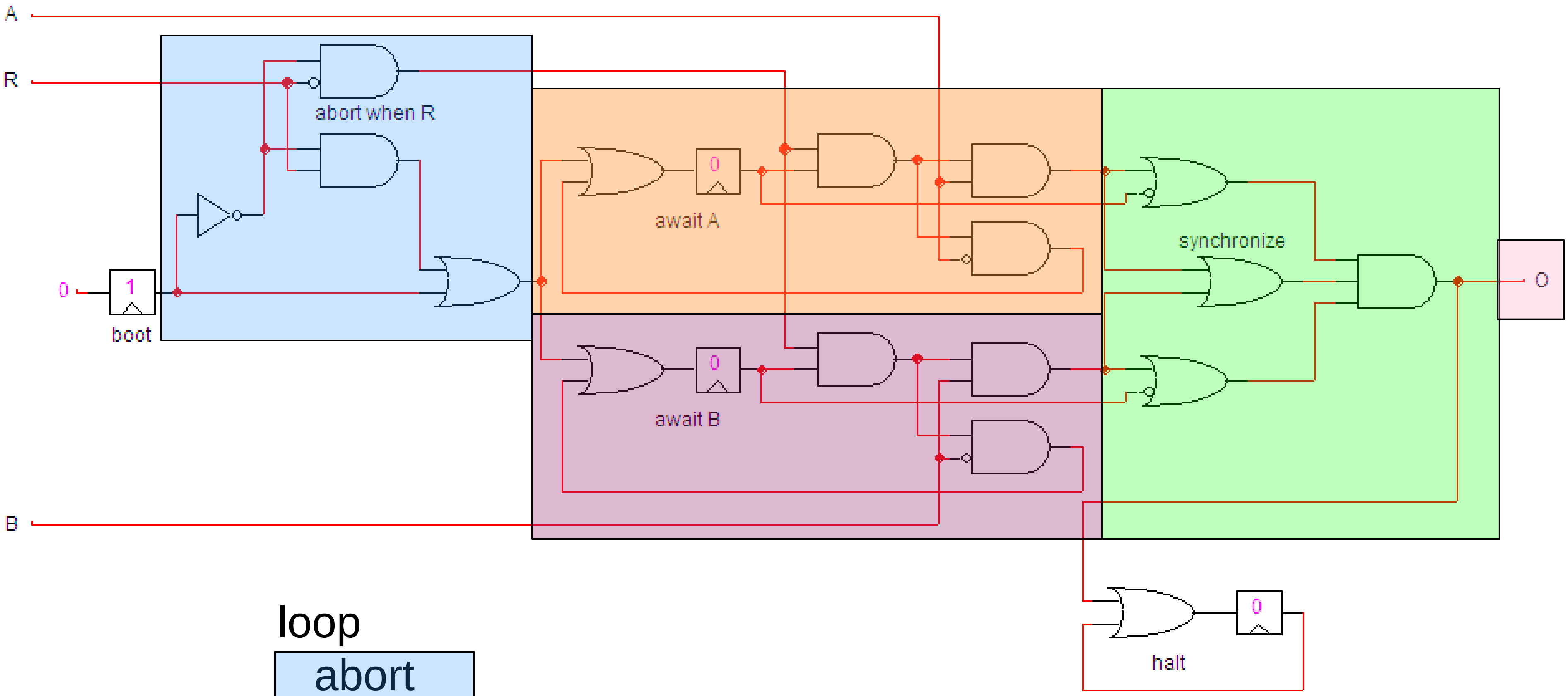


flat automaton

Hierarchical automaton
linear

source: l'oreille cassée, Hergé

From Esterel to Circuits



loop
  abort
    { await A || await B };
    emit O ;
    halt
  when R
end loop

Circuit =
constructive proof network

Group-Hot Coding and Optimization

```
loop
   { await A || await B } ;
   emit O
each  R
```

parallel threads => independent groups
sequence => group-hot
1-hot: 4 bits
log: 2 bits
group-hot: 3 bits – better scaling