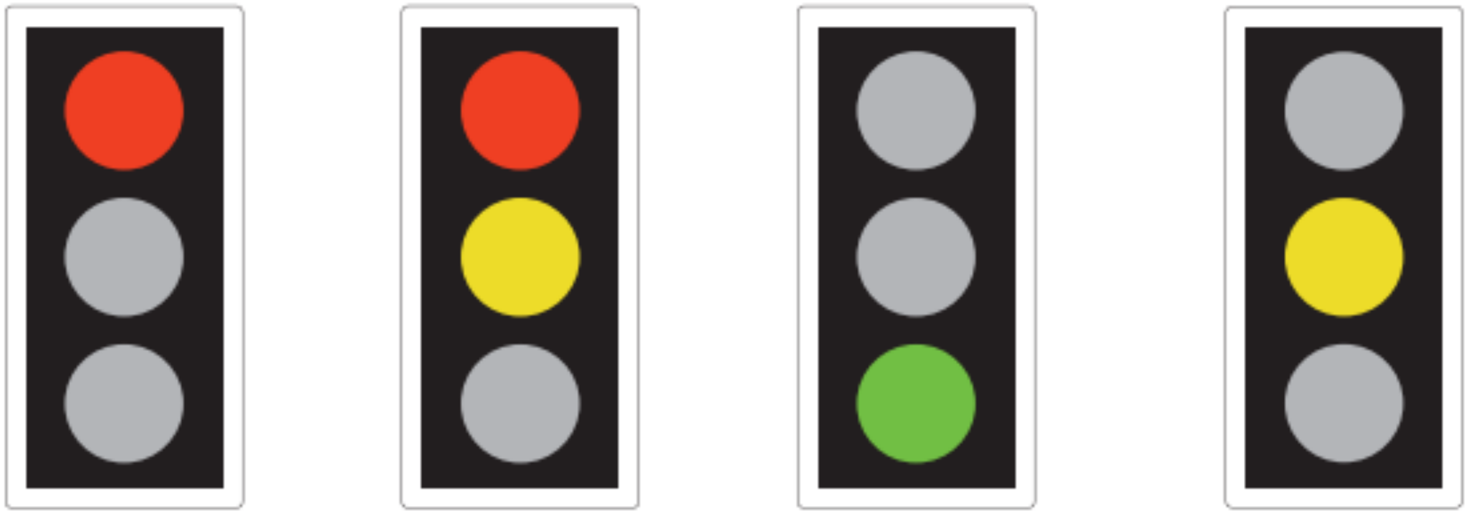
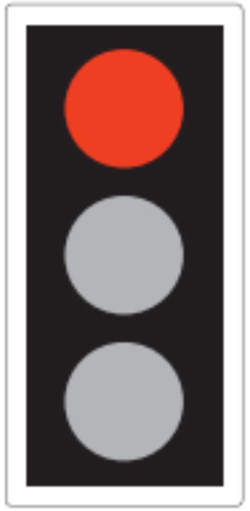


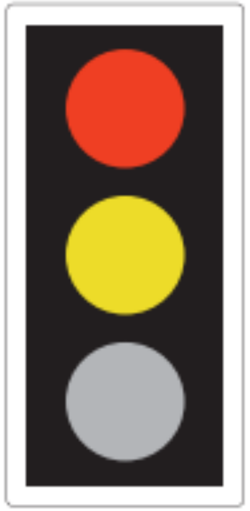
Computation and Logic

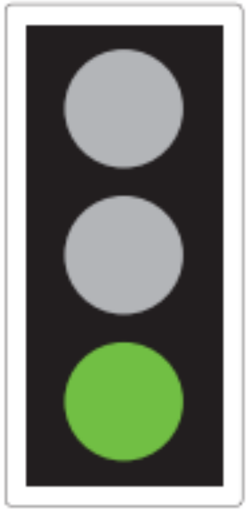
Traffic Lights

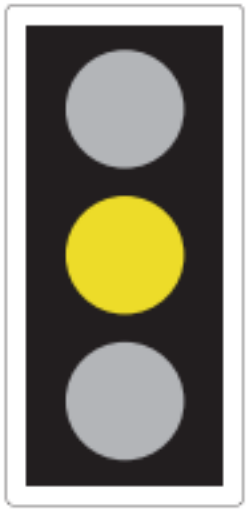
Michael Fourman
@mp4man

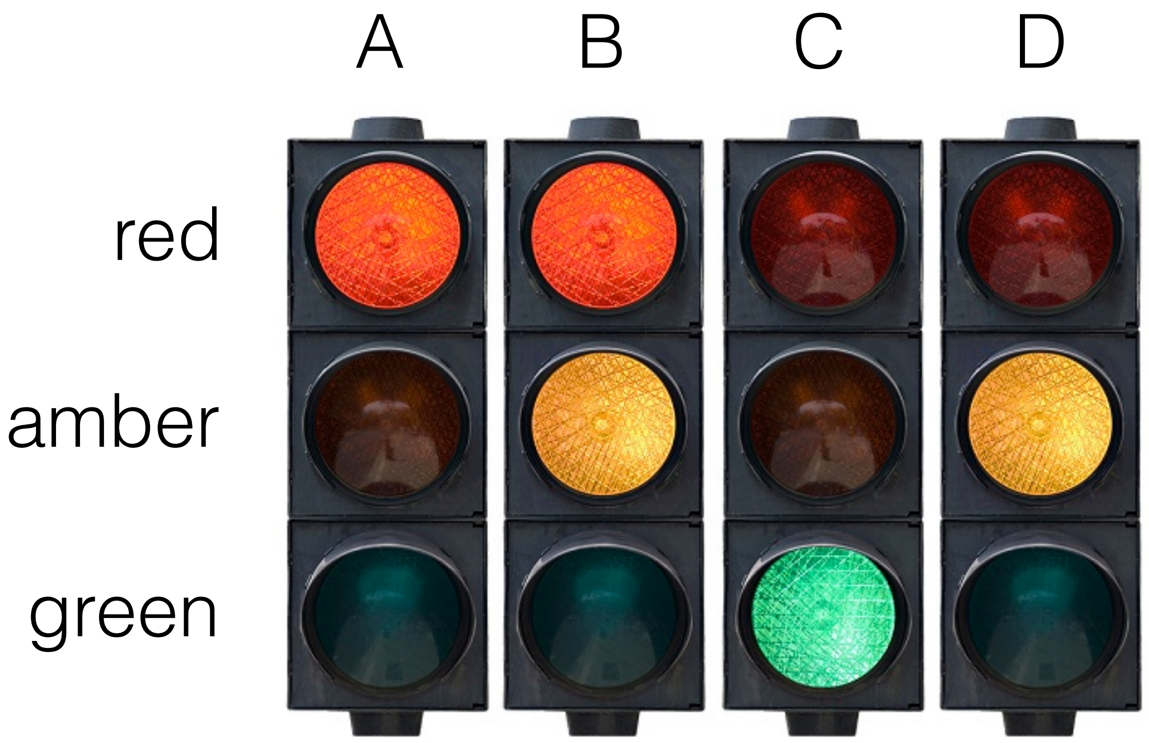
Traffic Light Signals


RED means 'Stop'. Wait behind the stop line on the carriageway


RED AND AMBER also means 'Stop'. Do not pass through or start until GREEN shows


GREEN means you may go on if the way is clear. Take special care if you intend to turn left or right and give way to pedestrians who are crossing


AMBER means 'Stop' at the stop line. You may go on only if the AMBER appears after you have crossed the stop line or are so close to it that to pull up might cause an accident

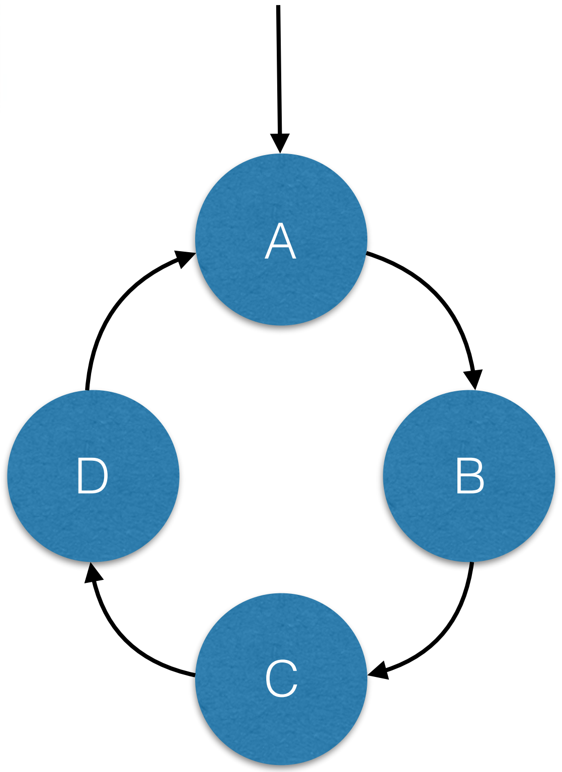


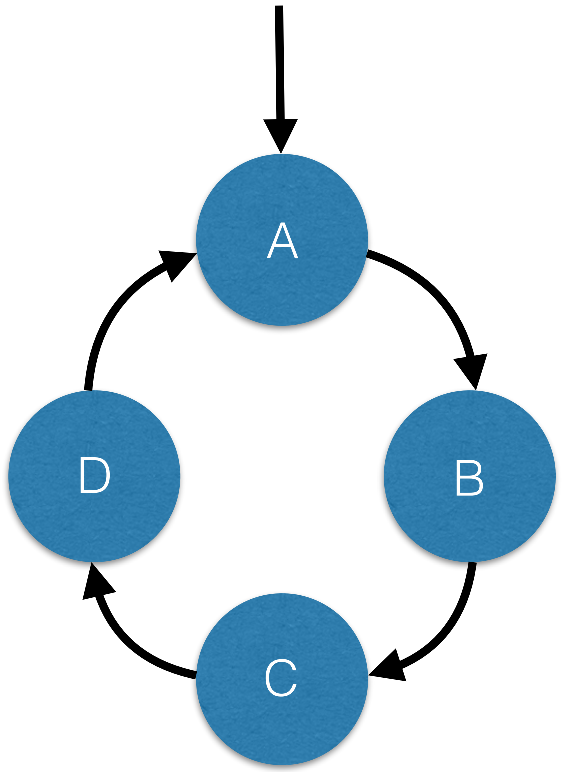
logic & computation

red iff A or B

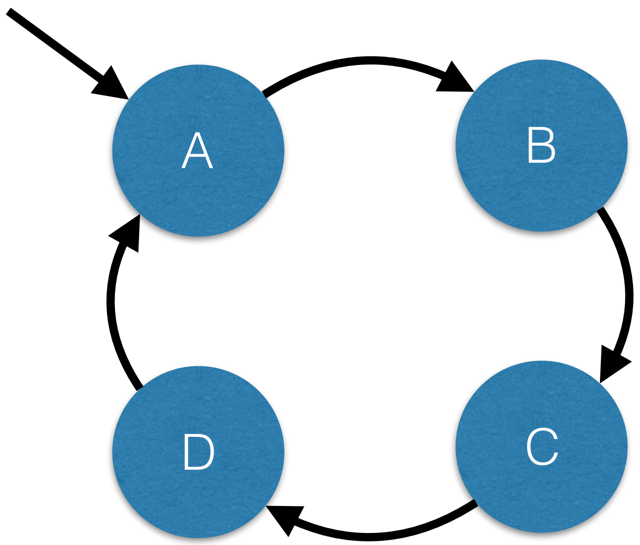
amber iff B or D

green iff C





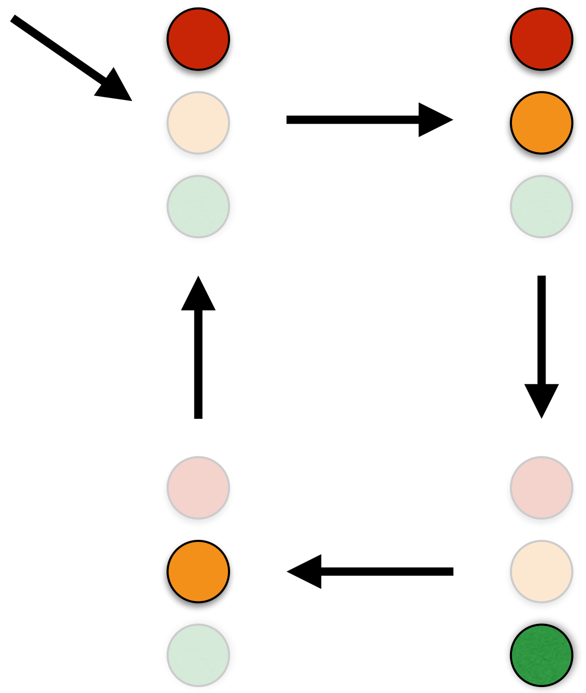
current	A	B	C	D
next	B	C	D	A



current

A	B	C	D
B	C	D	A

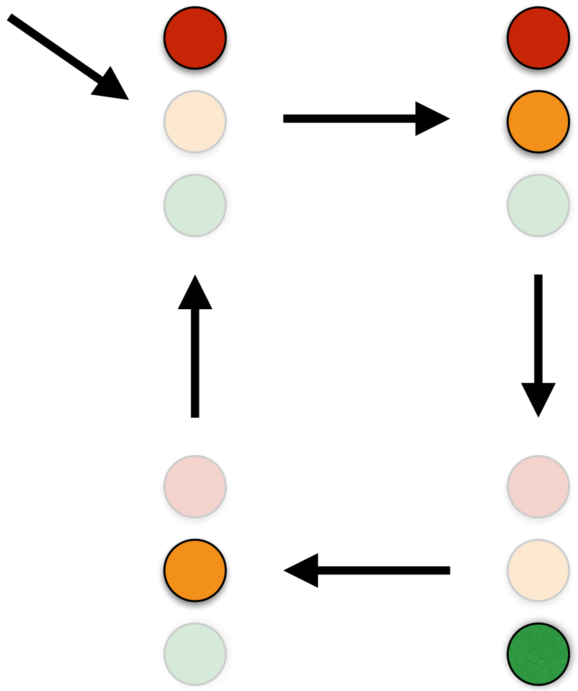
next

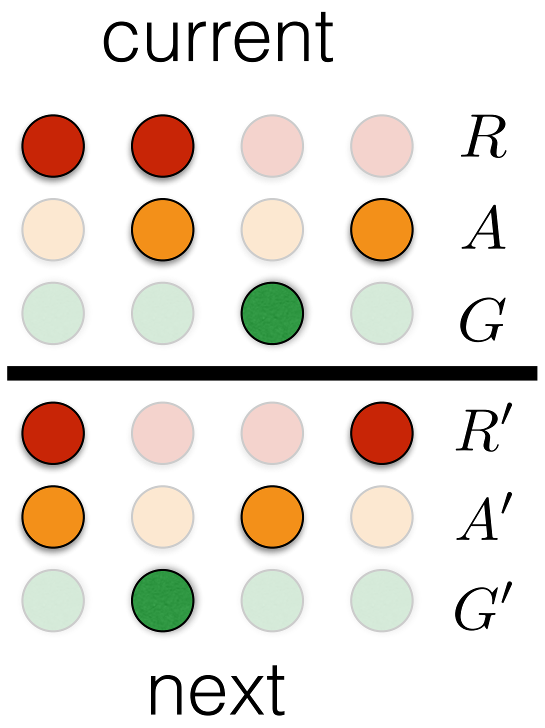


current

●	●	○	○	R
○	●	○	●	A
○	○	●	○	G
●	○	○	●	R'
●	○	●	○	A'
○	●	○	○	G'

next

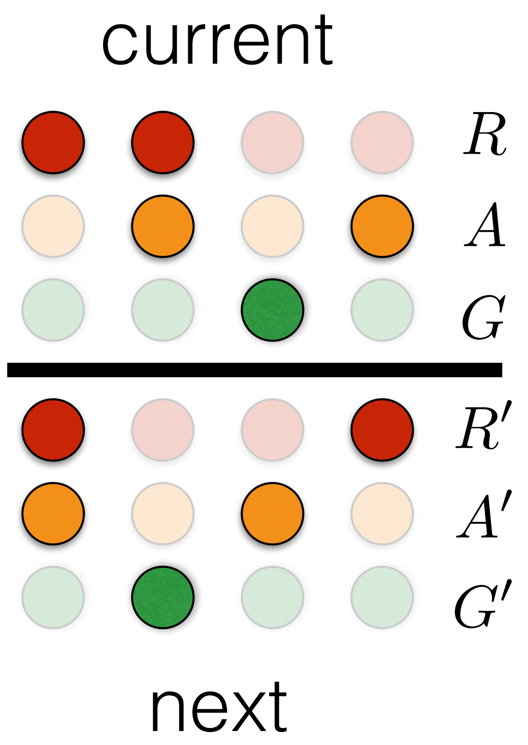



current
R
A
G
R'
A'
G'
next

$$R' = R \text{ xor } A = R \oplus A$$

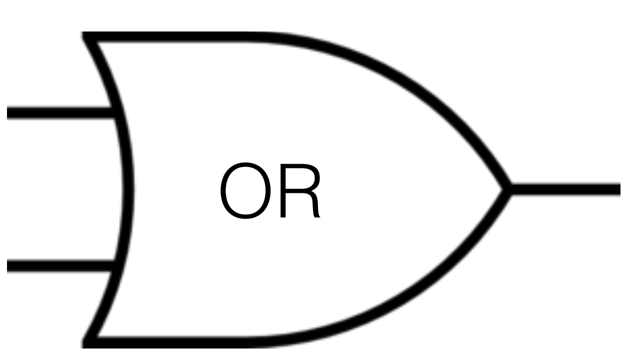
$$A' = \text{not } A = \neg A$$

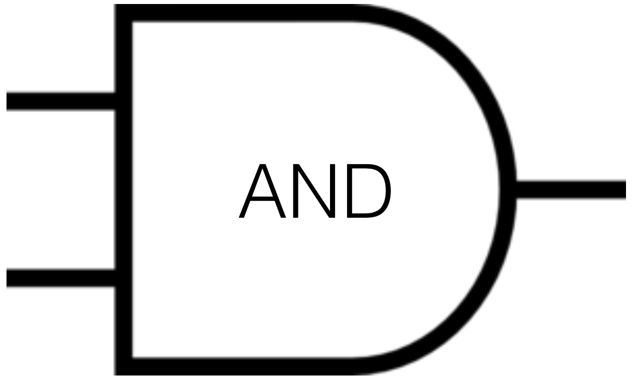
$$G' = R \text{ and } A = R \wedge A$$

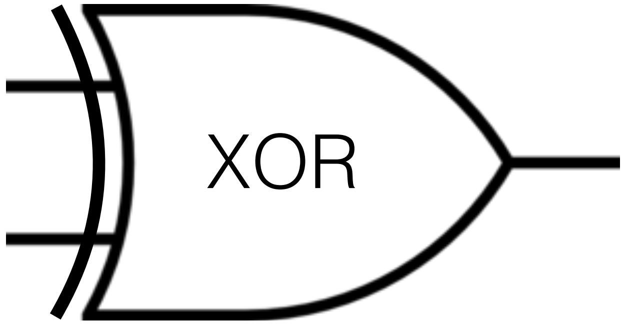


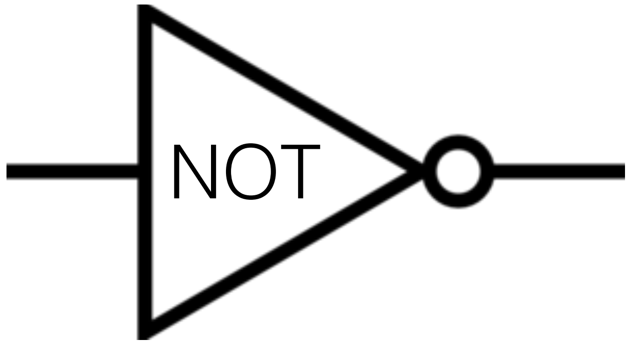
R	A	$R \wedge A$	$R \oplus A$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

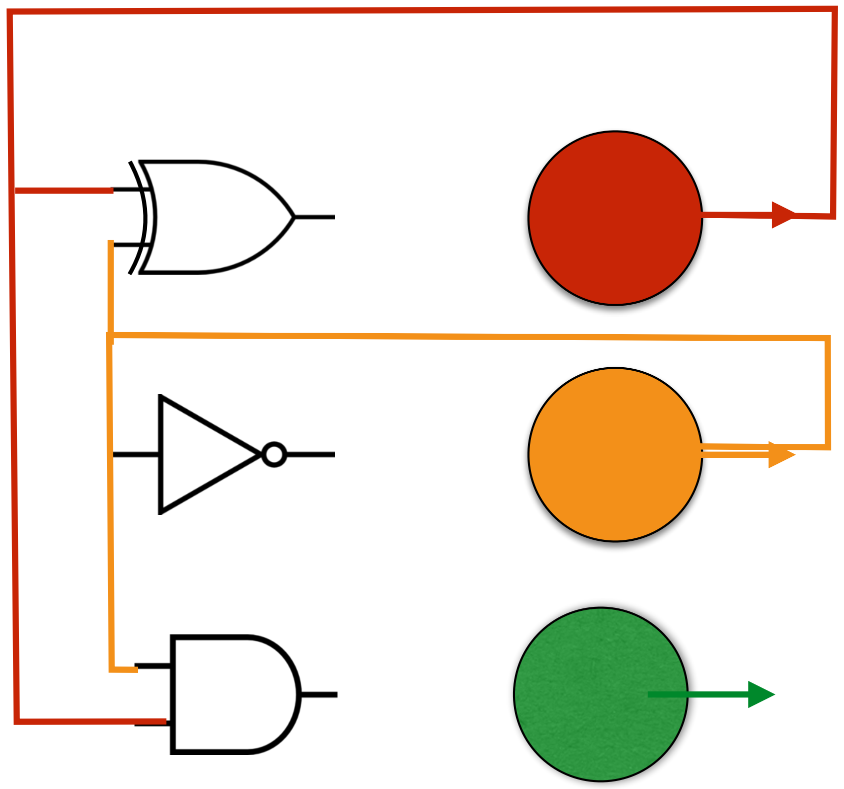
A	$\neg A$
0	1
1	0

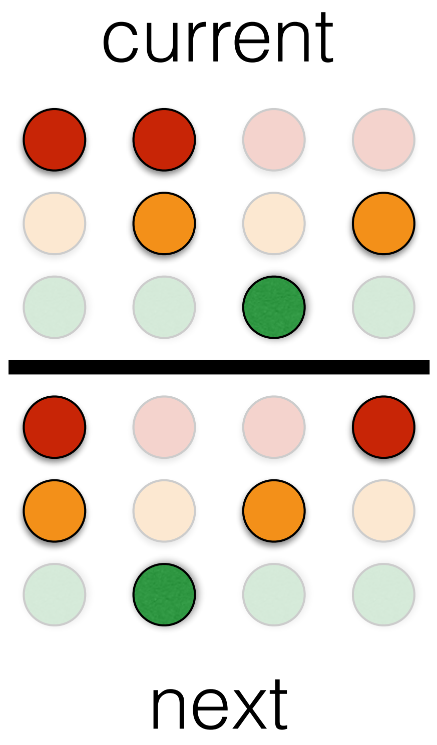

OR


AND

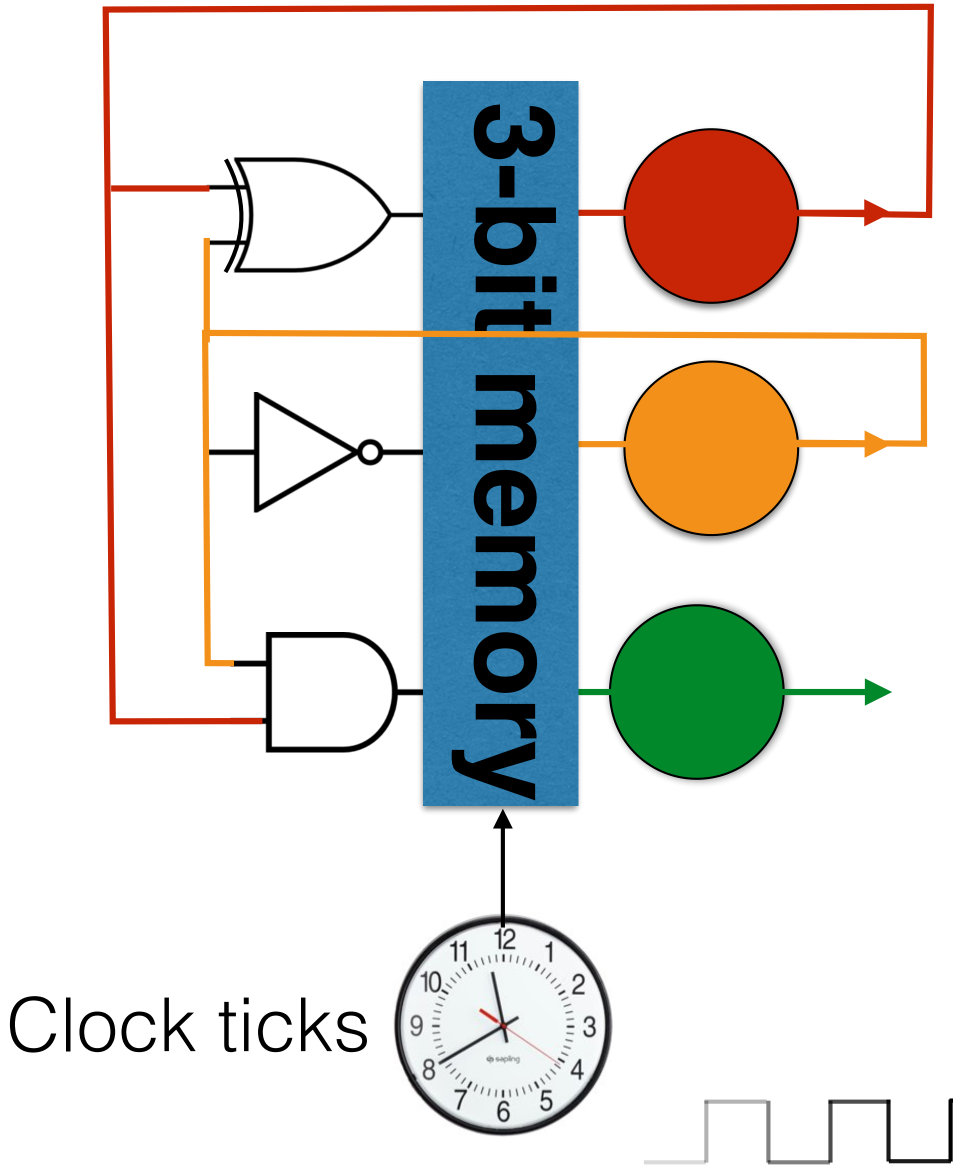

XOR

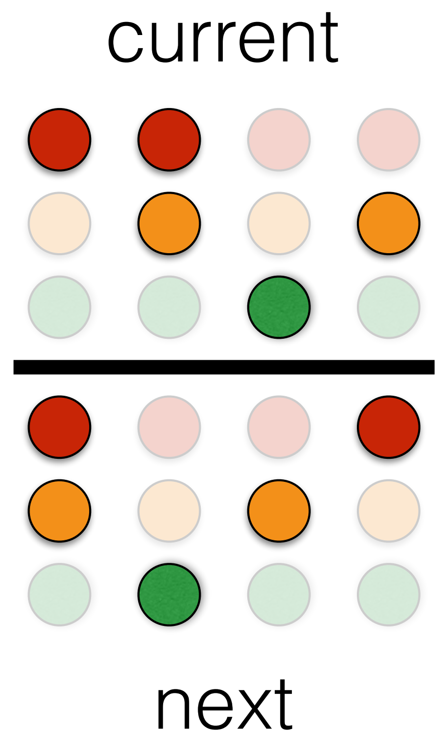

NOT



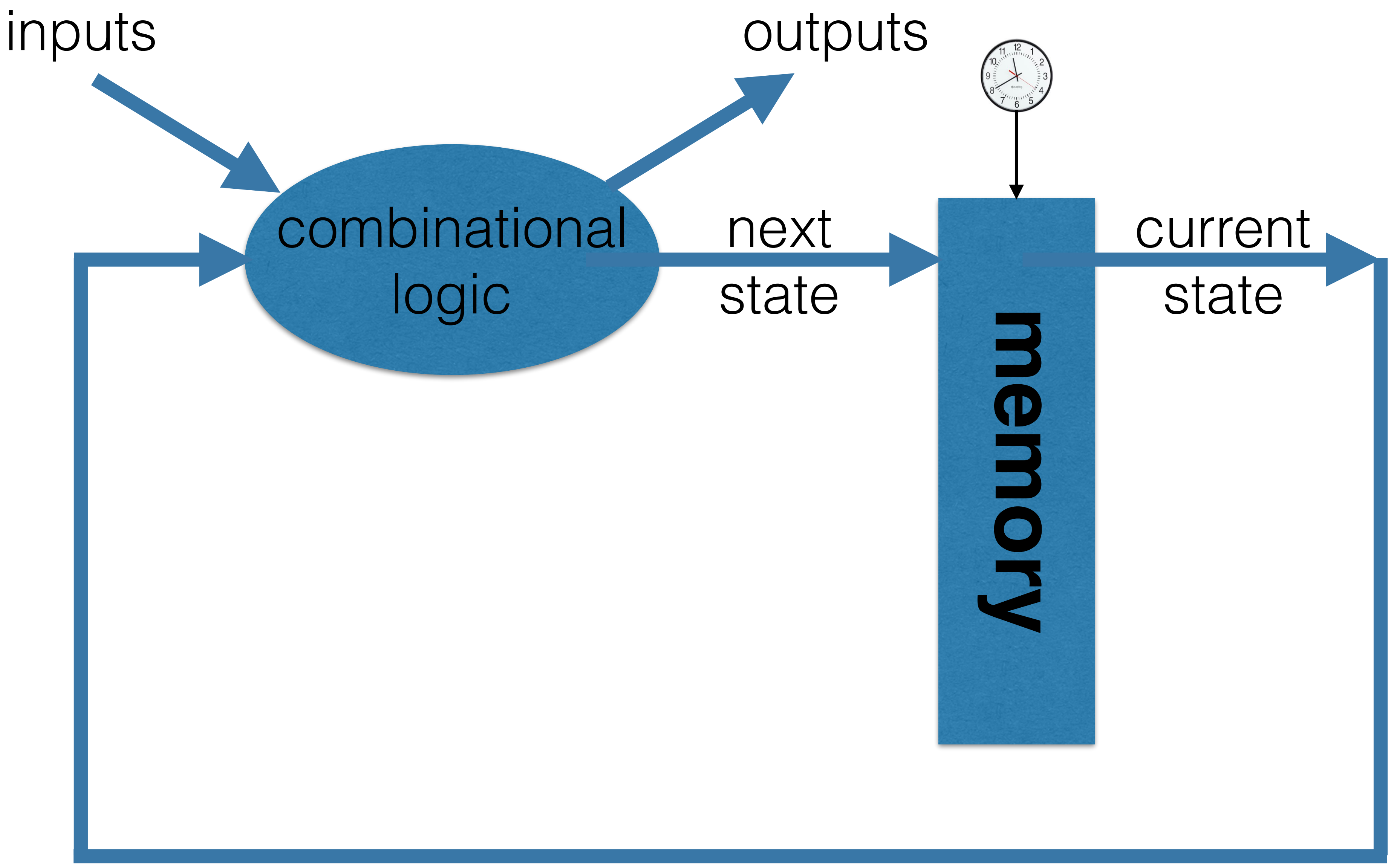


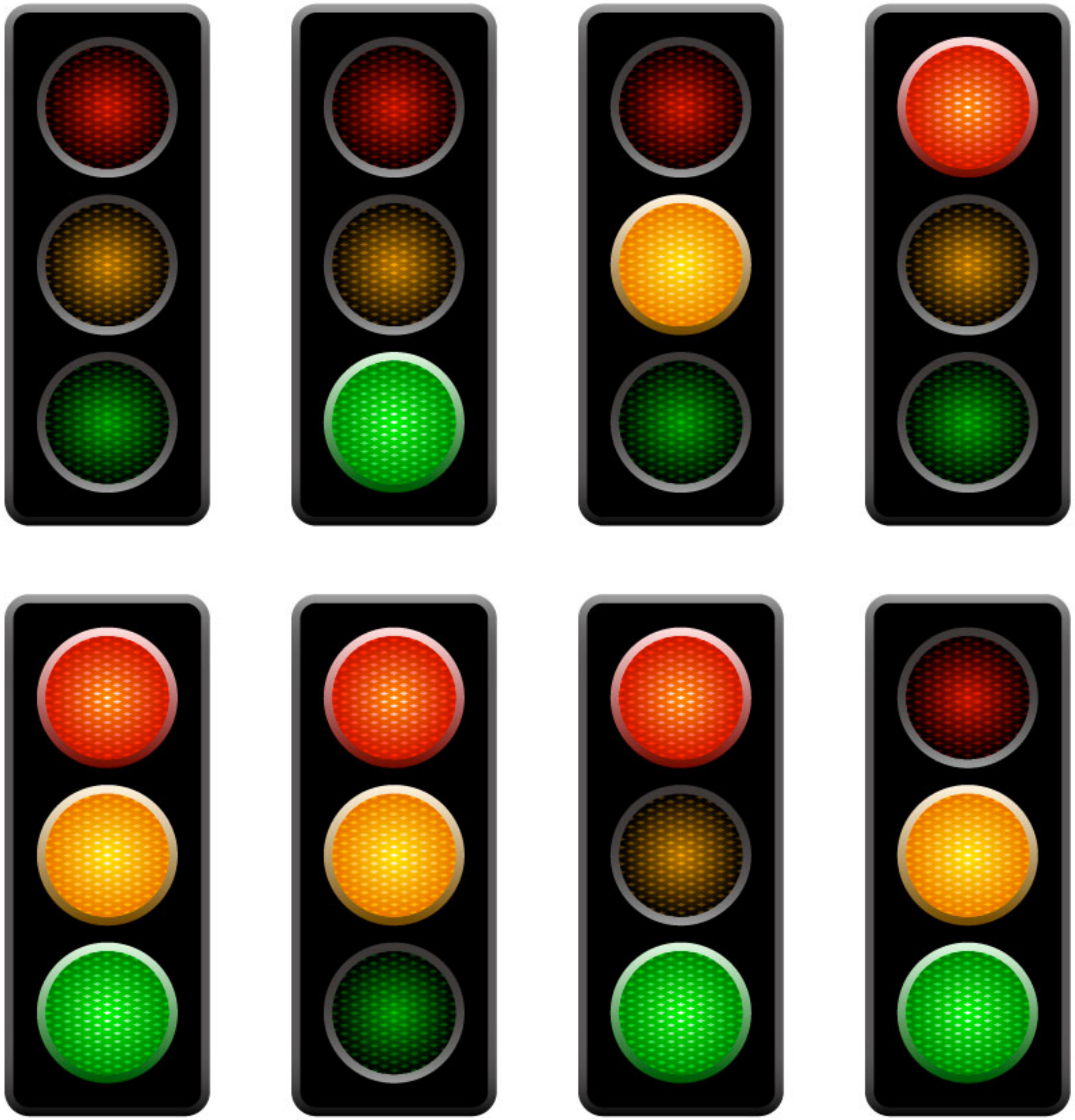
```
R' = R xor A
A' =   not A
G' = R and A
```



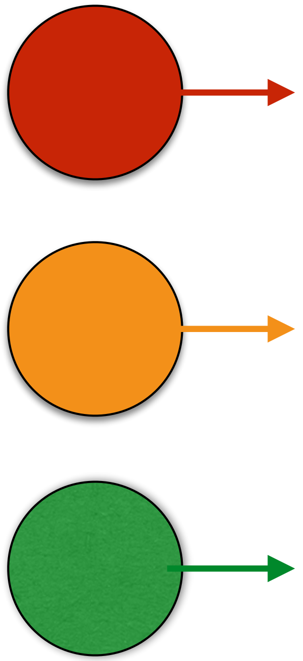


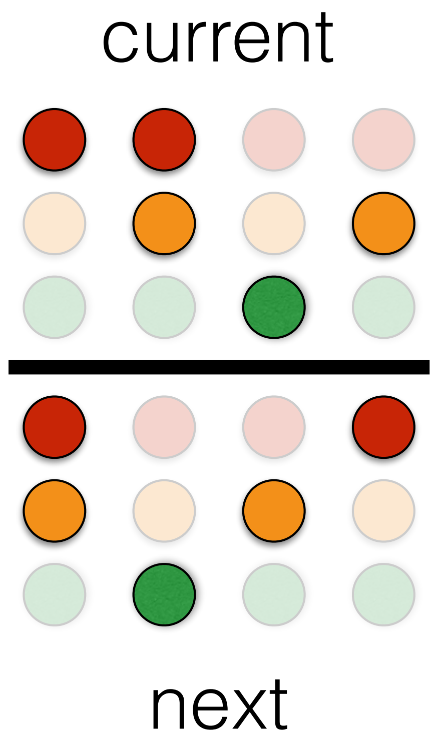
```
R' = R xor A
A' =   not A
G' = R and A
```


inputs
outputs
combinational logic
next state
memory
current state



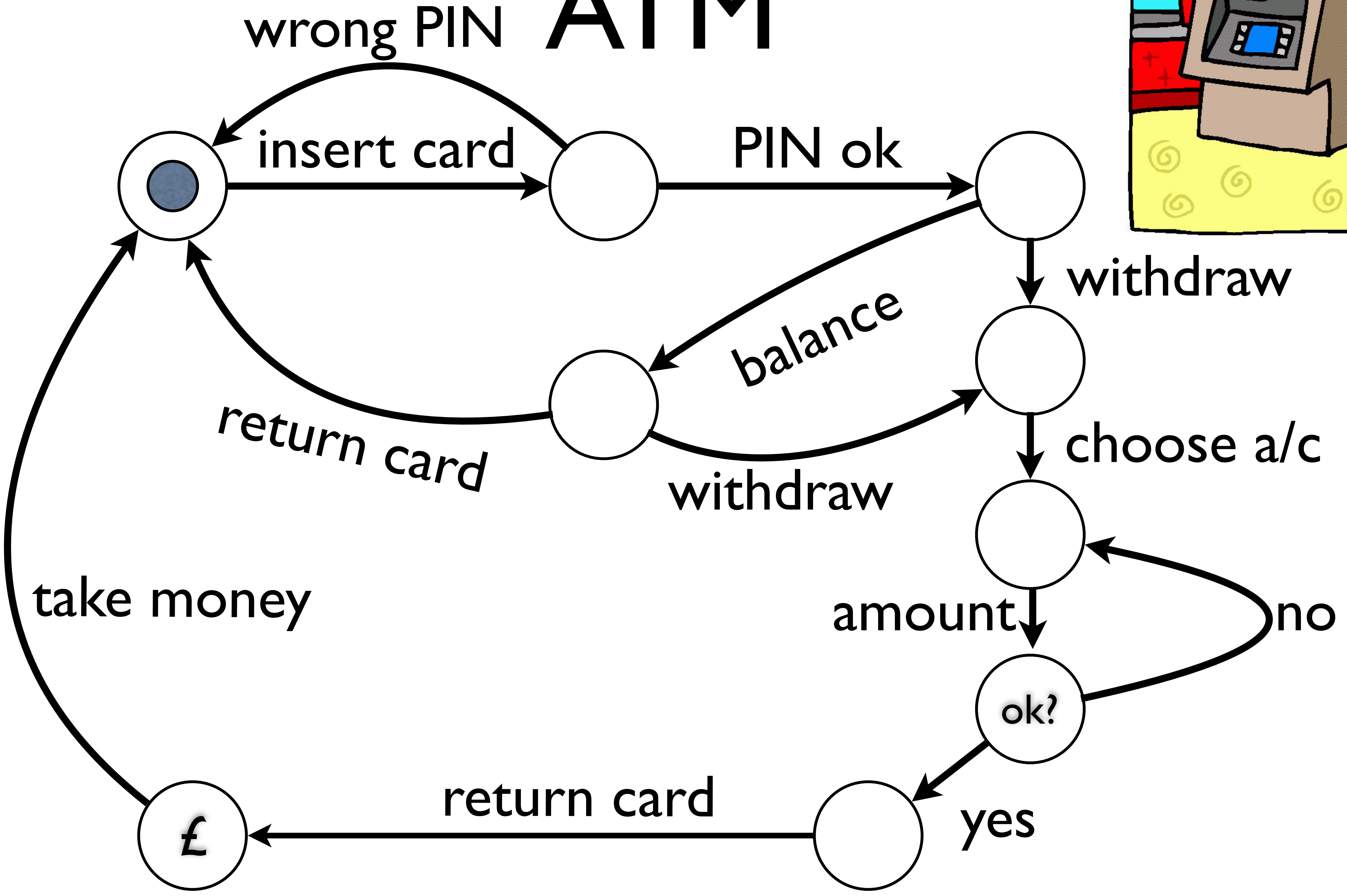
Exercise 1.2



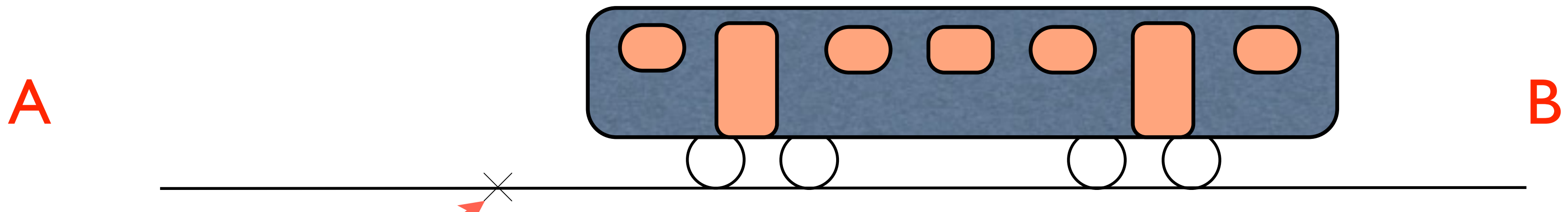


```
R' = R xor A
A' = G or (R and not A)
G' = R and A
```

ATM


wrong PIN
insert card
PIN ok
withdraw
balance
return card
withdraw
choose a/c
amount
no
ok?
yes
return card
£
take money

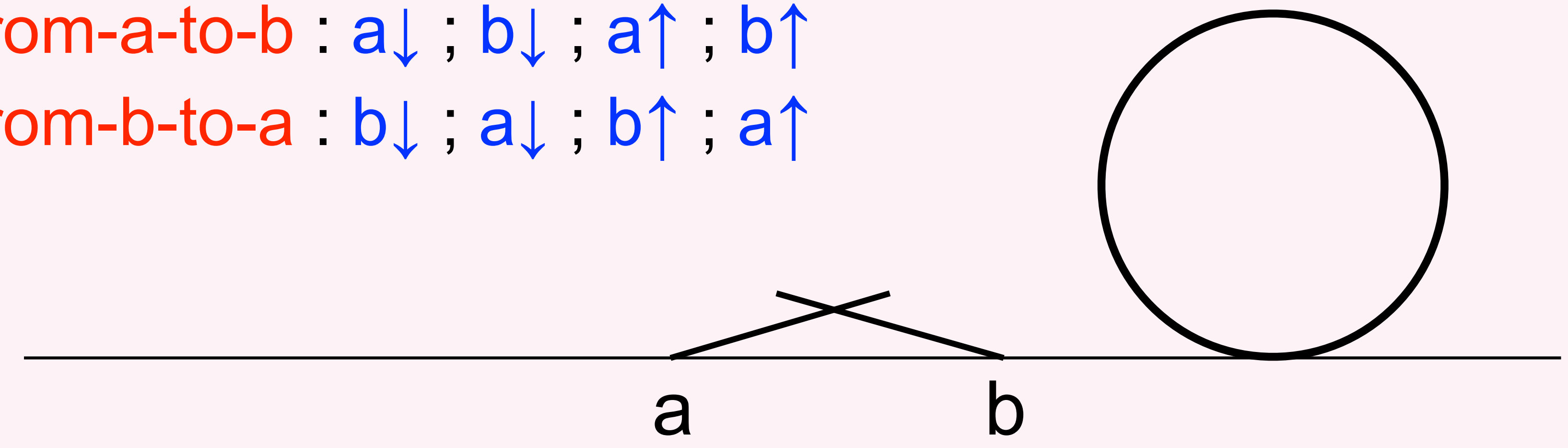
Counting trains



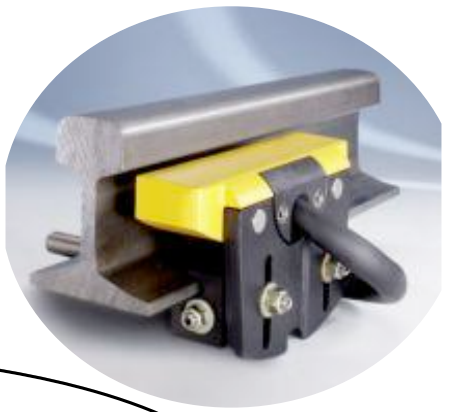
axle sensor (detects passing wheels)

from-a-to-b : a↓ ; b↓ ; a↑ ; b↑

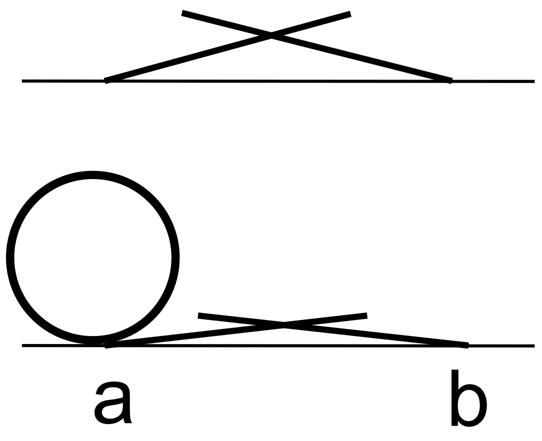
from-b-to-a : b↓ ; a↓ ; b↑ ; a↑



Finite-state machines

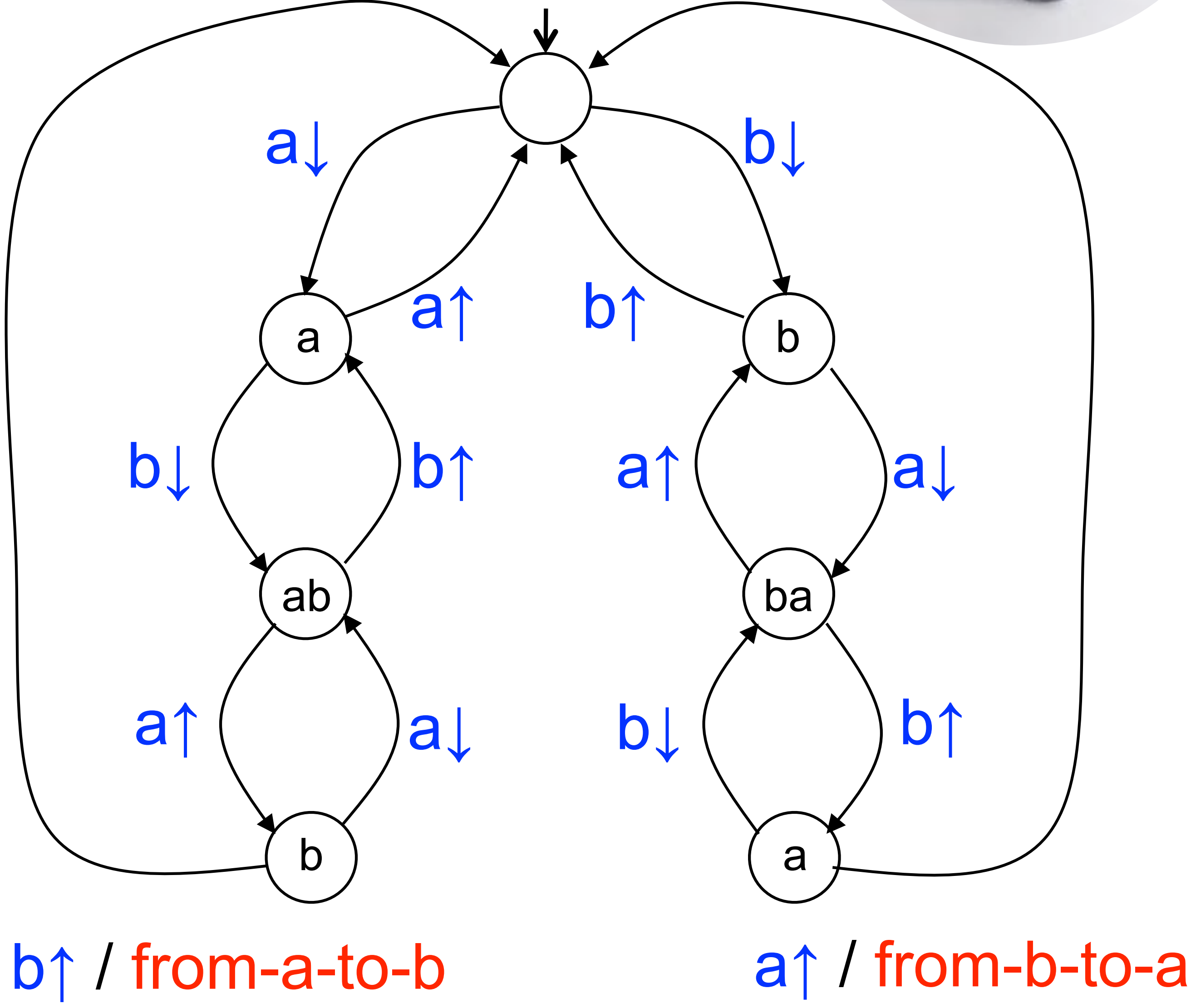


axle sensor

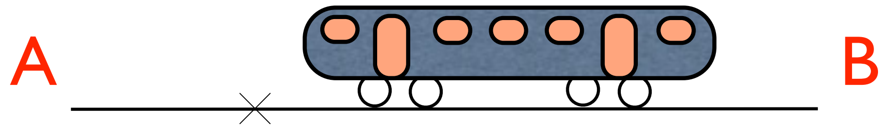


inputs :
a↑, a↓, b↑, b↓

outputs :
from-a-to-b,
from-b-to-a



Hierarchical FSMs



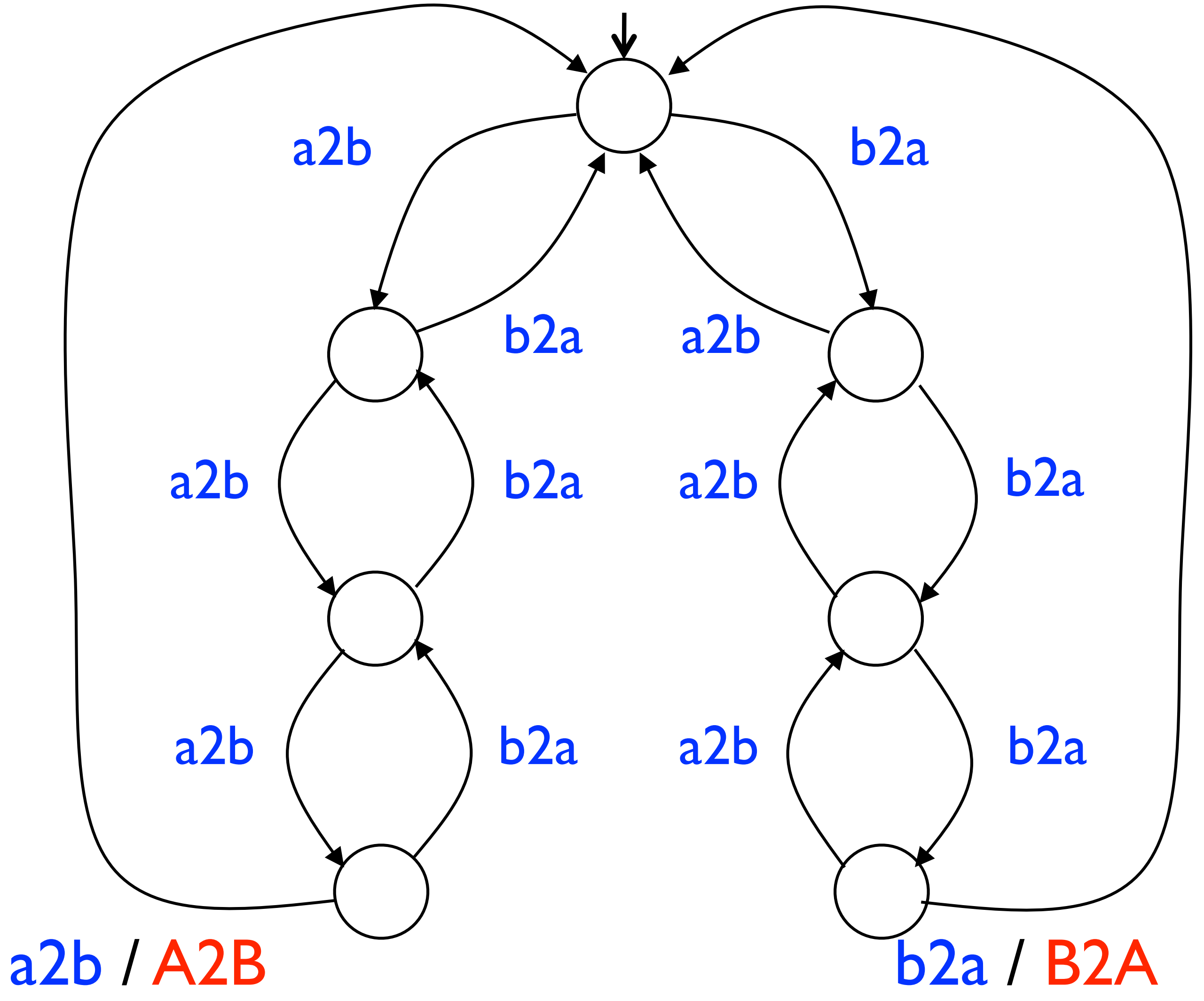
carriage counter

inputs :

a2b, b2a

outputs :

A2B, B2A



a2b = from-a-to-b

Application Fields

Industry

- real-time control, vending machines, cash dispensers, etc.

Electronic circuits

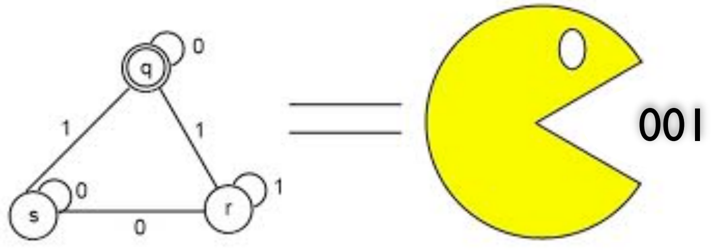
- data path / control path
- memory / cache handling
- protocols, USB, etc.

Communication protocols

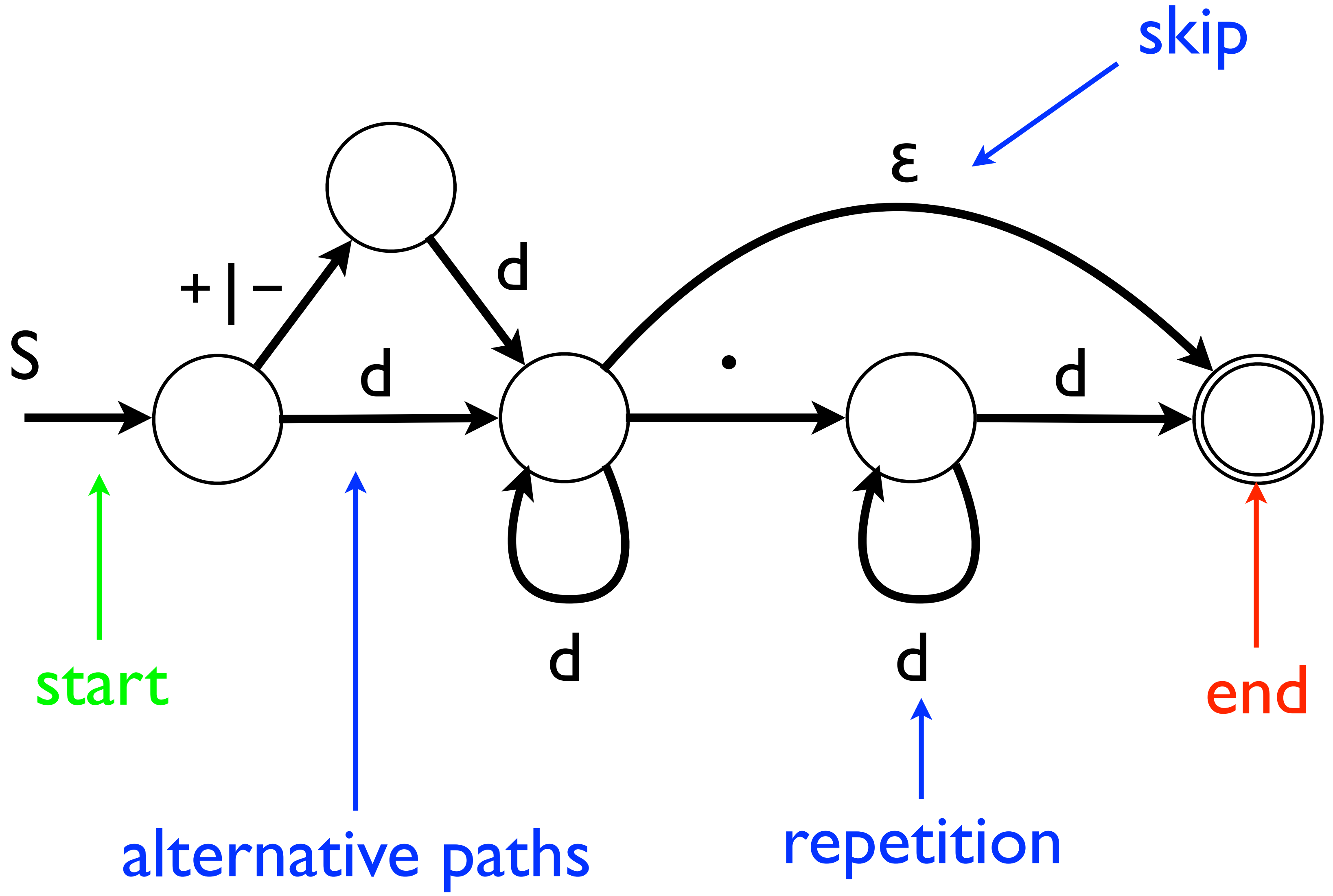
- initiation and maintenance of communication links
- error detection and handling, packet retransmission

Language analysis

- natural languages
- programming languages
- search engines



A Decimal Number


skip
ε
+|−
d
S
d
.
d
d
d
start
alternative paths
repetition
end