Fully abstract categorical semantics for digital circuits

George Kaye, David Sprunger and Dan Ghica
University of Birmingham

20 July 2022

ACT 2022

Joint work with...

David Sprunger

Dan Ghica

Introduction

Digital circuits are everywhere!

Introduction

Digital circuits are everywhere!

How do we reason with them?

Introduction

Generally by simulation

Introduction

Generally by simulation

Reasoning in software is more reduction-based:

$$((\lambda x.\lambda y.\, x + y)\, 2)\, 5$$

Introduction

Generally by simulation

Reasoning in software is more reduction-based:

$$((\lambda x.\lambda y.\, x + y)\, 2)\, 5 \;=_\beta\; (\lambda y.2 + y)\, 5$$

Introduction

Generally by simulation

Reasoning in software is more reduction-based:

$$((\lambda x.\lambda y.\, x + y)\, 2)\, 5 \;=_\beta\; (\lambda y.2 + y)\, 5 \;=_\beta\; 2 + 5$$

Introduction

Generally by simulation

Reasoning in software is more reduction-based:

$$((\lambda x.\lambda y.\, x + y)\, 2)\, 5 \;=_\beta\; (\lambda y.2 + y)\, 5 \;=_\beta\; 2 + 5 \;=_\eta\; 7$$

Introduction

Generally by simulation

Reasoning in software is more reduction-based:

$$((\lambda x.\lambda y.\, x + y)\, 2)\, 5 \;=_\beta\; (\lambda y.2 + y)\, 5 \;=_\beta\; 2 + 5 \;=_\eta\; 7$$

We want an equational theory for digital circuits

Syntax

Combinational circuit components

Combinational circuit components

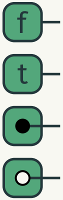
Values

false

true

Combinational circuit components

Values

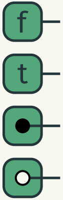


- false
- true
- disconnected
- short circuit

(Belnap's four valued logic)

Combinational circuit components

Values



false

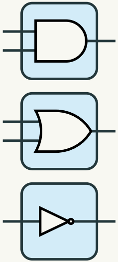
true

disconnected

short circuit

(Belnap's four valued logic)

Gates



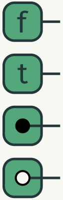
AND gate

OR gate

NOT gate

Combinational circuit components

Values



false

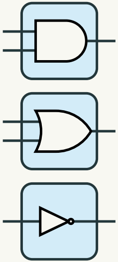
true

disconnected

short circuit

(Belnap's four valued logic)

Gates



AND gate

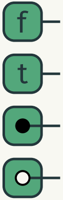
OR gate

NOT gate

Structure

Combinational circuit components

Values



false

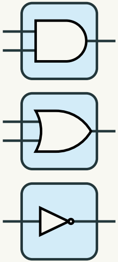
true

disconnected

short circuit

(Belnap's four valued logic)

Gates

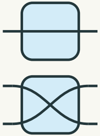


AND gate

OR gate

NOT gate

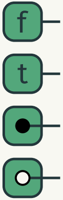
Structure



identity

symmetry

Values



false

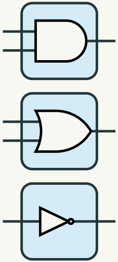
true

disconnected

short circuit

(Belnap's four valued logic)

Gates

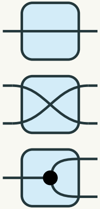


AND gate

OR gate

NOT gate

Structure



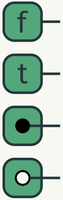
identity

symmetry

fork

Combinational circuit components

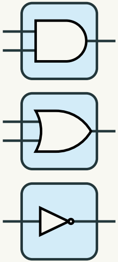
Values



false
true
disconnected
short circuit

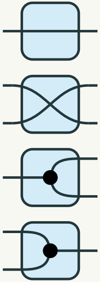
(Belnap's four valued logic)

Gates



AND gate
OR gate
NOT gate

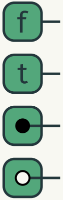
Structure



identity
symmetry
fork
join

Combinational circuit components

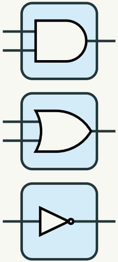
Values



false
true
disconnected
short circuit

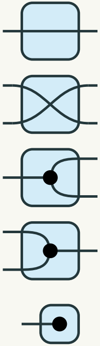
(Belnap's four valued logic)

Gates



AND gate
OR gate
NOT gate

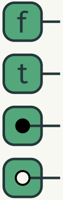
Structure



identity
symmetry
fork
join
stub

Combinational circuit components

Values



false

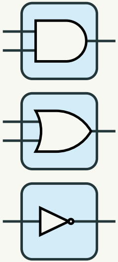
true

disconnected

short circuit

(Belnap's four valued logic)

Gates

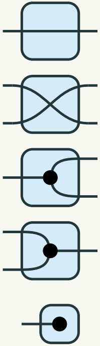


AND gate

OR gate

NOT gate

Structure



identity

symmetry

fork

join

stub

Light circuits 

only contain gates and structure.

Sequential circuit components

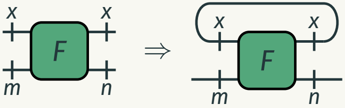
Sequential circuit components

Delay

Sequential circuit components

Delay

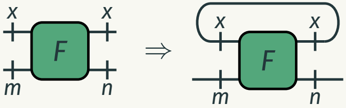
Feedback



Sequential circuit components

Delay

Feedback

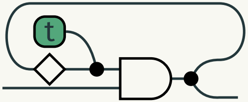


Dark circuits 

may contain delay or feedback.

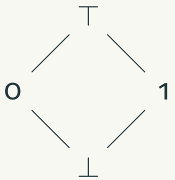
Circuit morphisms

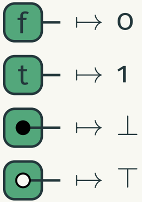
Morphisms in a freely generated symmetric traced monoidal category



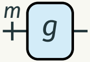
Semantics

Values are interpreted in a lattice **V**:





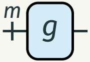
Interpretation



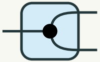
monotone functions

$\overline{g} \colon \mathbf{V}^m \to \mathbf{V}$

Interpretation

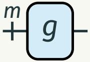


monotone functions $\overline{g} : \mathbf{V}^m \to \mathbf{V}$

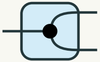


copy $x \mapsto (x, x)$

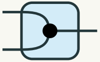
Interpretation



monotone functions $\overline{g} \colon \mathbf{V}^m \to \mathbf{V}$

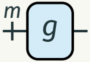


copy $x \mapsto (x, x)$

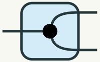


join in the lattice $(x, y) \mapsto x \sqcup y$

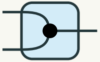
Interpretation



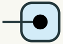
monotone functions $\overline{g} \colon \mathbf{V}^m \to \mathbf{V}$



copy $x \mapsto (x, x)$



join in the lattice $(x, y) \mapsto x \sqcup y$



discard $x \mapsto \bullet$

Stream functions

The semantics of circuits is that of stream functions.

Stream functions

The semantics of circuits is that of stream functions.

A stream $\mathbf{V}^\omega$ is an infinite sequence of values.

Stream functions

The semantics of circuits is that of stream functions.

A stream $\mathbf{V}^\omega$ is an infinite sequence of values.

A stream function $f\colon (\mathbf{V}^m)^\omega \to (\mathbf{V}^n)^\omega$ consumes and produces streams.

Causal stream functions

Not all stream functions correspond to sequential circuits...

Causal stream functions

Not all stream functions correspond to sequential circuits...

Causal
Depends on past inputs

Causal stream functions

Not all stream functions correspond to sequential circuits...

Causal
Depends on past inputs

Monotone
with respect to the lattice

Causal stream functions

Not all stream functions correspond to sequential circuits...

Causal
Depends on past inputs

Monotone
with respect to the lattice

'Finite'
Specifies finite behaviours

Causal stream functions

Not all stream functions correspond to sequential circuits...

Causal
Depends on past inputs

Monotone
with respect to the lattice

‘Finite’
Specifies finite behaviours

Theorem

Every monotone causal stream function with ‘finite behaviours’ corresponds to a class of sequential circuits.

Equational reasoning

Equality of circuits

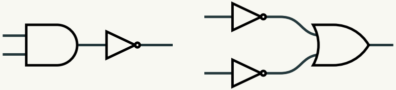
When are two circuits equal?

Equality of circuits

When are two circuits equal? When they have the same behviour

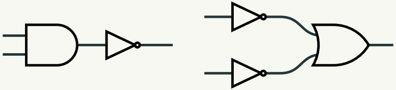
Equality of circuits

When are two circuits equal? When they have the same behviour



Equality of circuits

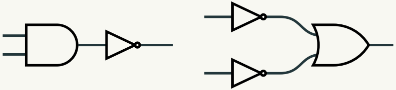
When are two circuits equal? When they have the same behviour



When they have the same stream function

Equality of circuits

When are two circuits equal? When they have the same behviour



When they have the same stream function

Reasoning with streams is a pain.

Productivity

We want to reason equationally: what equations do we need?

Productivity

We want to reason equationally: what equations do we need?

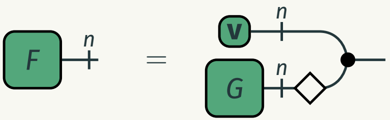
First goal: productivity.

Productivity

We want to reason equationally: what equations do we need?

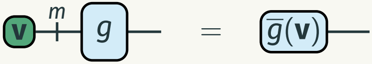
First goal: productivity.

A closed circuit is productive if it is equal to an instant value and a delayed subcircuit under the equational theory.

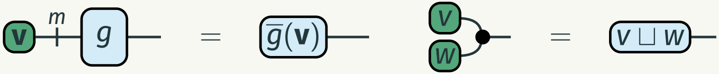


Combinational equations

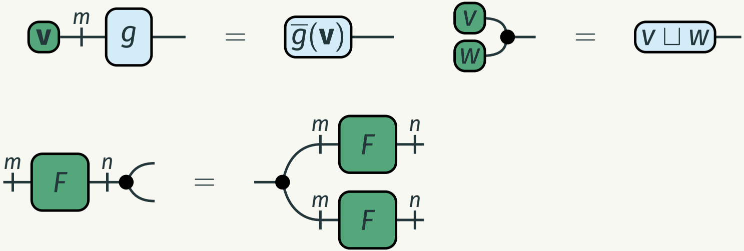
Combinational equations



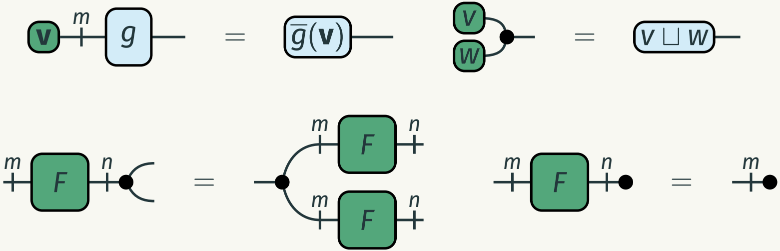
Combinational equations



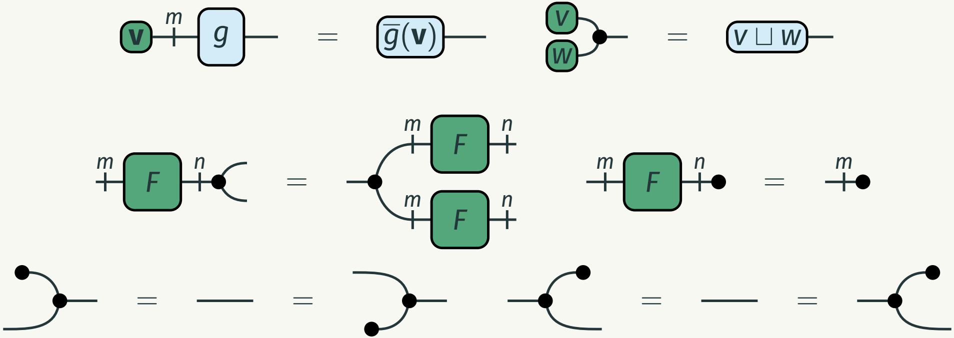
Combinational equations



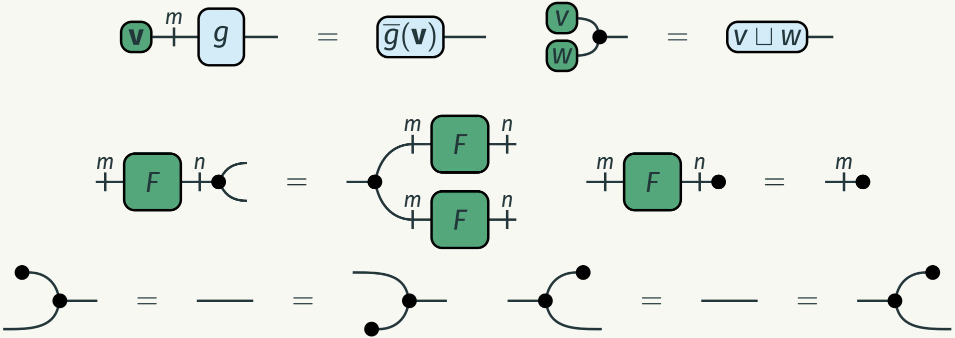
Combinational equations

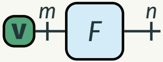


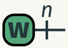
Combinational equations



Combinational equations

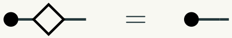


These reduce any closed combinational circuit 

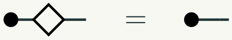
to some 

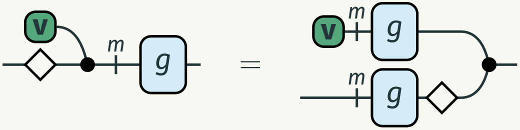
.

Sequential equations



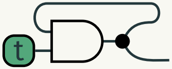
Sequential equations





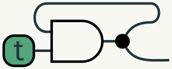
Non delay-guarded feedback

How do we deal with something like this?

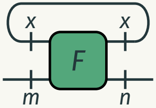


Non delay-guarded feedback

How do we deal with something like this?



We need a way to eliminate non delay-guarded feedback.



Non delay-guarded feedback

Non delay-guarded feedback

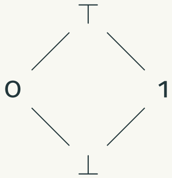
Our gates are monotonic, so they must have a least fixed point...

Non delay-guarded feedback

Our gates are monotonic, so they must have a least fixed point...

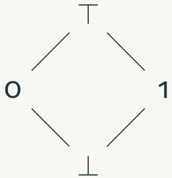
Because the value set **V** is finite, we can always find this fixpoint!

Non delay-guarded feedback

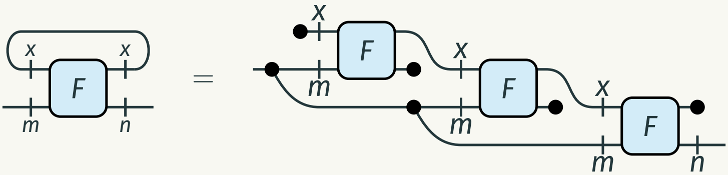


In **V**, the length of the longest chain is 2...

Non delay-guarded feedback

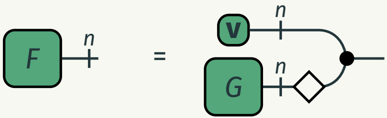


In **V**, the length of the longest chain is 2...



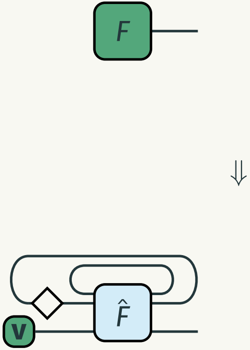
Productivity

We want



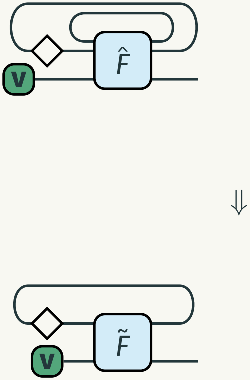
Productivity

Axioms of STMCs



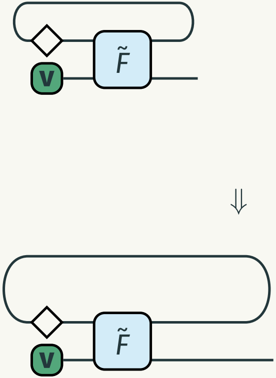
Productivity

Eliminating ‘instant feedback’

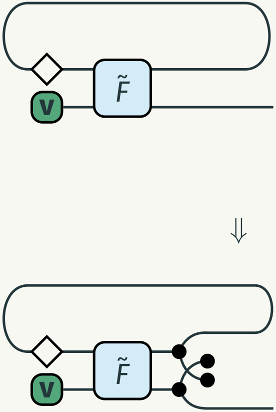


Productivity

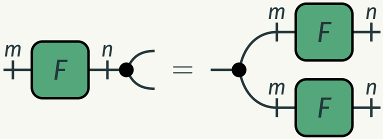
Axioms of STMCs

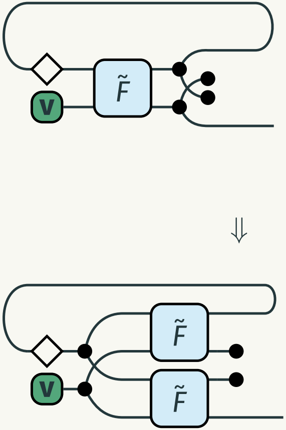


Productivity

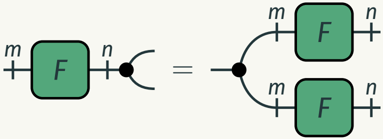


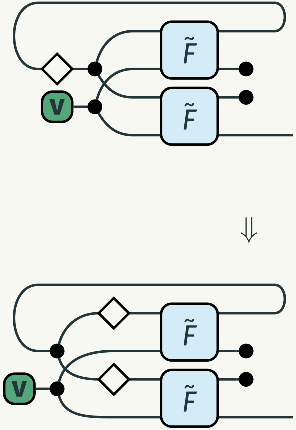
Productivity





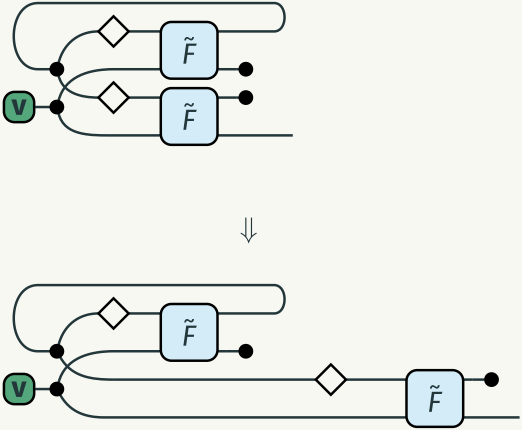
Productivity





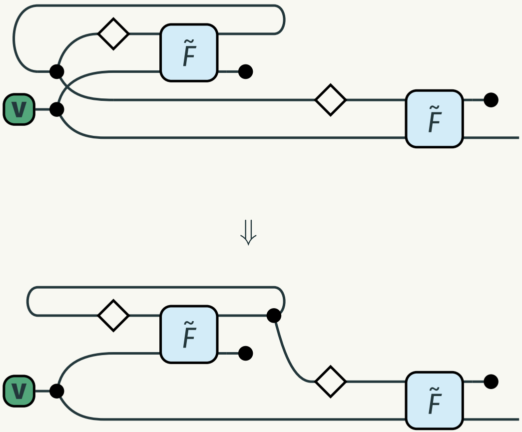
Productivity

Axioms of STMCs

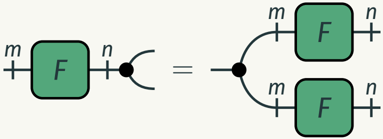


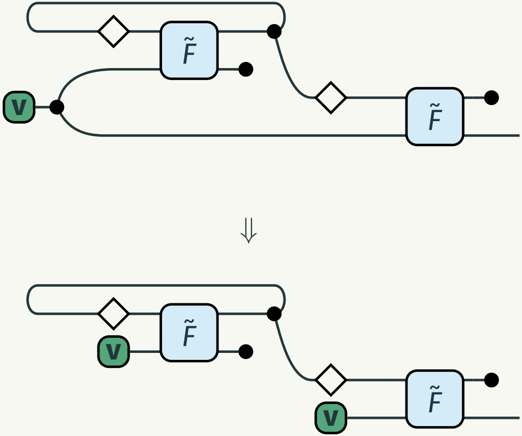
Productivity

Axioms of STMCs

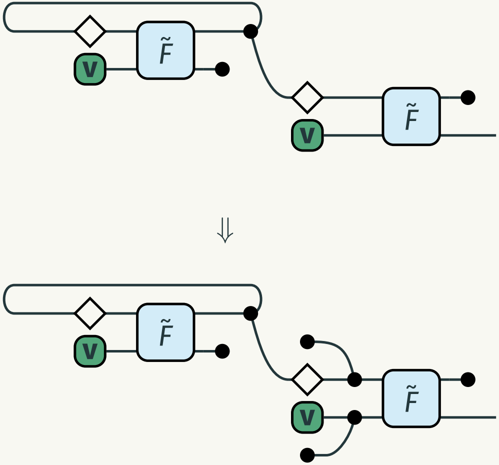


Productivity

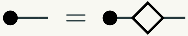


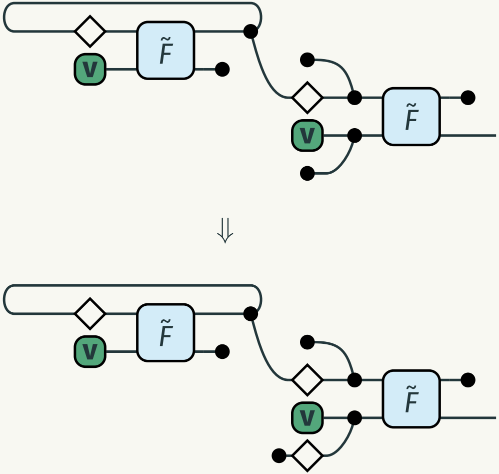


Productivity

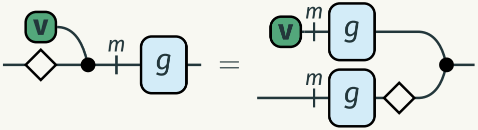


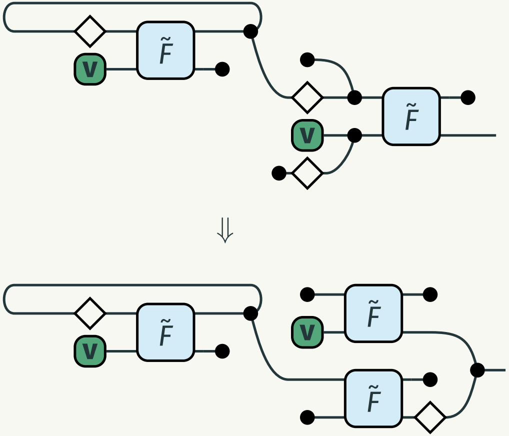
Productivity





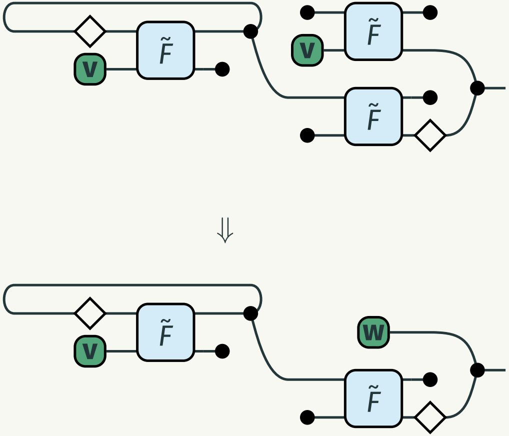
Productivity





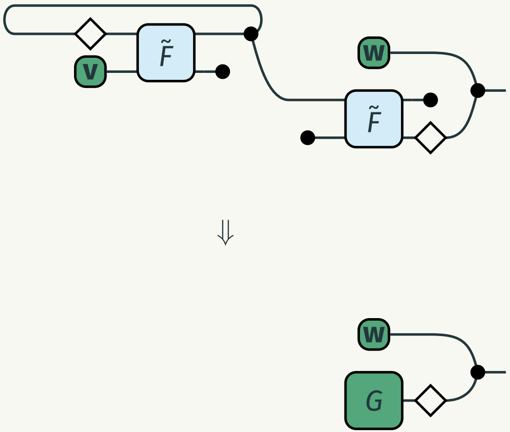
Productivity

Combinational circuit equations



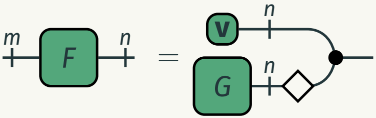
Productivity

Tidying up



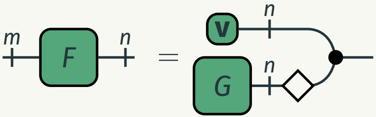
Productivity

Any circuit has an instantaneous value and a delayed subcircuit.



Productivity

Any circuit has an instantaneous value and a delayed subcircuit.



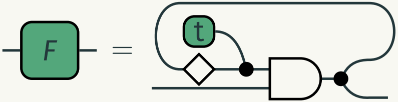
These values are the elements of the corresponding stream!

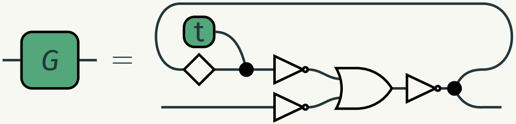
Open circuits

We still cannot translate between open circuits with the same behaviour.

Open circuits

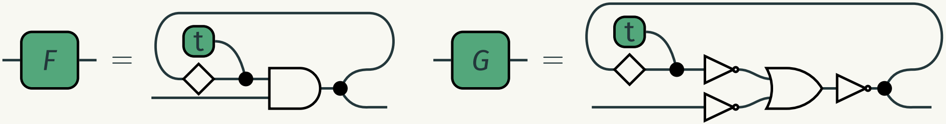
We still cannot translate between open circuits with the same behaviour.





Open circuits

We still cannot translate between open circuits with the same behaviour.



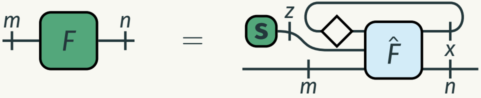
When do two circuits have the same stream?

Open circuits

We can think of circuits as state machines:

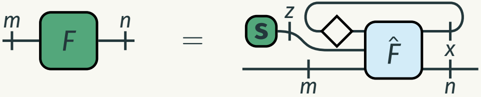
Open circuits

We can think of circuits as state machines:



Open circuits

We can think of circuits as state machines:



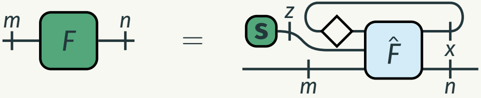
The circuit 

produces the state transition and output of 

.

Open circuits

We can think of circuits as state machines:



The circuit 

produces the state transition and output of 

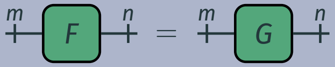
.

Idea: for all accessible states, if the outputs are equal then the original circuits are equal under the equational theory.

(cf. Mealy machine bisimulation)

Full abstraction

Theorem



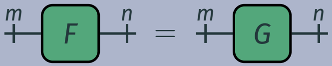
if and only if their streams are equal.

Proof.

□

Full abstraction

Theorem



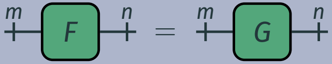
if and only if their streams are equal.

Proof.

□

Full abstraction

Theorem



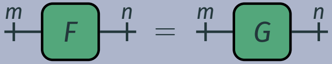
if and only if their streams are equal.

Proof.

□

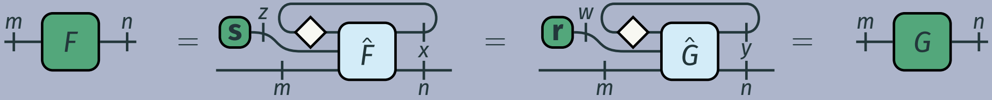
Full abstraction

Theorem



if and only if their streams are equal.

Proof.



□

Conclusion

We have presented a categorical framework for sequential circuits

Conclusion

We have presented a categorical framework for sequential circuits

Circuits have semantics as stream functions

Conclusion

We have presented a categorical framework for sequential circuits

Circuits have semantics as stream functions

It is easier to reason equationally

Conclusion

We have presented a categorical framework for sequential circuits

Circuits have semantics as stream functions

It is easier to reason equationally

We have full abstraction: a correspondence between syntactic and semantic

Conclusion

We have presented a categorical framework for sequential circuits

Circuits have semantics as stream functions

It is easier to reason equationally

We have full abstraction: a correspondence between syntactic and semantic

Next step: refine the rewriting system