A formal study of two physical countermeasures against side channel attacks

PROOFS'2012

Sébastien Briais Sylvain Guilley Jean-Luc Danger

2012, September 13th

THE TRUSTED COMPUTING COMPANY

Physical Attacks

Cryptographic devices need to be protected.

Physical Attacks

Cryptographic devices need to be protected.

Side-Channel Attacks

- Passive attacks.
- Power consumption, electromagnetic radiation, computation time... may leak sensitive data.

Physical Attacks

Cryptographic devices need to be protected.

Side-Channel Attacks

- Passive attacks.
- Power consumption, electromagnetic radiation, computation time... may leak sensitive data.

Extra logic is required in order to mask the sensitive data or to balance the leakage.

Dual-rail Precharge Logic

- Aims at making the device activity independent on the data being processed.

Dual-rail Precharge Logic

- Aims at making the device activity independent on the data being processed.
- A signal is represented by a pair of wires: $T = 10$, $F = 01$.

Dual-rail Precharge Logic

- Aims at making the device activity independent on the data being processed.
- A signal is represented by a pair of wires: $T = 10$, $F = 01$.
- A cycle of computation alternates two phases:
  - precharge phase: propagation of $\mathrm{NULL} = \{(0, 0)\}$ through the combinational part of the circuit.
  - evaluation phase: the data is processed by the combinational part of the circuit.

Dual-rail Precharge Logic

- Aims at making the device activity independent on the data being processed.
- A signal is represented by a pair of wires: $T = 10$, $F = 01$.
- A cycle of computation alternates two phases:
  - precharge phase: propagation of $\mathrm{NULL} = \{(0, 0)\}$ through the combinational part of the circuit.
  - evaluation phase: the data is processed by the combinational part of the circuit.

Many proposals: WDDL, STTL, DRSL, BCDL, ...

Dual-rail Precharge Logic

- Aims at making the device activity independent on the data being processed.
- A signal is represented by a pair of wires: $T = 10$, $F = 01$.
- A cycle of computation alternates two phases:
  - precharge phase: propagation of $\mathrm{NULL} = \{(0, 0)\}$ through the combinational part of the circuit.
  - evaluation phase: the data is processed by the combinational part of the circuit.

Many proposals: WDDL, STTL, DRSL, BCDL, ...
Possible vulnerabilities:

- Glitches
- Early evaluation

Dual-rail Precharge Logic

- Aims at making the device activity independent on the data being processed.
- A signal is represented by a pair of wires: $T = 10$, $F = 01$.
- A cycle of computation alternates two phases:
  - precharge phase: propagation of $\mathrm{NULL} = \{(0, 0)\}$ through the combinational part of the circuit.
  - evaluation phase: the data is processed by the combinational part of the circuit.

Many proposals: WDDL, STTL, DRSL, BCDL, ...
Possible vulnerabilities:

- Glitches
- Early evaluation

THE TRUSTED COMPUTING COMPANY

THE TRUSTED COMPUTING COMPANY

Syntax

A combinational circuit is a directed acyclic graph of logical gates.

Syntax

A combinational circuit is a directed acyclic graph of logical gates.

Syntax

A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

Syntax

A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0}$$

empty circuit

Syntax

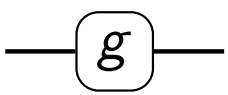
A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0} \mid g$$



logical gate $g \in \mathcal{G}$

Syntax

A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0} \mid g \mid \mathbf{I}$$

a single wire

Syntax

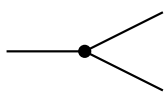
A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0} \mid g \mid \mathbf{I} \mid \mathbf{Y}$$



a fork

Syntax

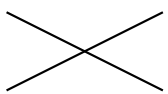
A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0} \mid g \mid \mathbf{I} \mid \mathbf{Y} \mid \mathbf{X}$$



a swap

Syntax

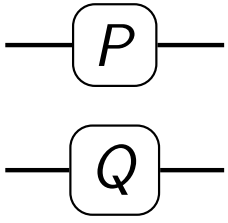
A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0} \mid g \mid \mathbf{I} \mid \mathbf{Y} \mid \mathbf{X} \mid P \| Q$$



parallel composition

Syntax

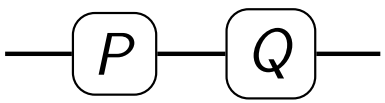
A combinational circuit is a directed acyclic graph of logical gates.

Combinational circuits

Let $\mathcal{G}$ be a set of logical gates.

We define by induction the set of combinational circuits over $\mathcal{G}$:

$$P, Q ::= \mathbf{0} \mid g \mid \mathbf{I} \mid \mathbf{Y} \mid \mathbf{X} \mid P \| Q \mid P \,;\, Q$$



sequential composition

Well-formedness

Circuit with n inputs and m outputs

$$\frac{\mathcal{T}(g) = (n, m)}{g : n \otimes m}\ g \in \mathcal{G}$$

$$\frac{}{\mathbf{0} : 0 \otimes 0} \qquad \frac{}{\mathbf{I} : 1 \otimes 1} \qquad \frac{}{\mathbf{Y} : 1 \otimes 2} \qquad \frac{}{\mathbf{X} : 2 \otimes 2}$$

$$\frac{P_1 : n_1 \otimes m_1 \qquad P_2 : n_2 \otimes m_2}{P_1 \parallel P_2 : n_1 + n_2 \otimes m_1 + m_2} \qquad \frac{P_1 : n \otimes m \qquad P_2 : m \otimes p}{P_1 \,;\, P_2 : n \otimes p}$$

Well-formedness

Circuit with n inputs and m outputs

$$\frac{\mathcal{T}(g) = (n, m)}{g : n \otimes m}\ g \in \mathcal{G}$$

$$\frac{}{\mathbf{0} : 0 \otimes 0} \qquad \frac{}{\mathbf{I} : 1 \otimes 1} \qquad \frac{}{\mathbf{Y} : 1 \otimes 2} \qquad \frac{}{\mathbf{X} : 2 \otimes 2}$$

$$\frac{P_1 : n_1 \otimes m_1 \qquad P_2 : n_2 \otimes m_2}{P_1 \parallel P_2 : n_1 + n_2 \otimes m_1 + m_2} \qquad \frac{P_1 : n \otimes m \qquad P_2 : m \otimes p}{P_1 \,;\, P_2 : n \otimes p}$$

Well-formedness

Circuit with n inputs and m outputs

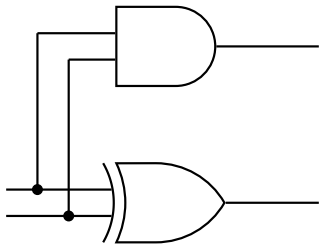
$$\frac{\mathcal{T}(g) = (n, m)}{g : n \otimes m}\ g \in \mathcal{G}$$

$$\frac{}{\mathbf{0} : 0 \otimes 0} \qquad \frac{}{\mathbf{I} : 1 \otimes 1} \qquad \frac{}{\mathbf{Y} : 1 \otimes 2} \qquad \frac{}{\mathbf{X} : 2 \otimes 2}$$

$$\frac{P_1 : n_1 \otimes m_1 \qquad P_2 : n_2 \otimes m_2}{P_1 \parallel P_2 : n_1 + n_2 \otimes m_1 + m_2} \qquad \frac{P_1 : n \otimes m \qquad P_2 : m \otimes p}{P_1 ; P_2 : n \otimes p}$$

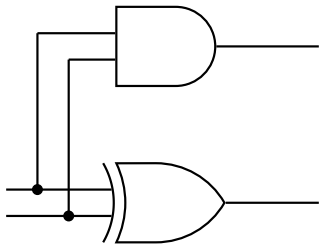
Example: a half-adder

$$\mathcal{G} = \{\mathrm{AND}, \mathrm{XOR}\}$$



Example: a half-adder

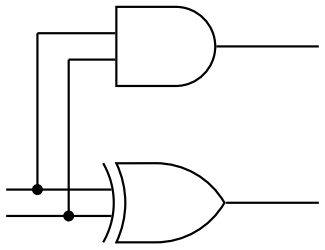
$\mathcal{G} = \{\mathrm{AND}, \mathrm{XOR}\}$



$$\mathsf{Half} := (\mathbf{Y} \,\|\, \mathbf{Y})\,;\, (\mathbf{I} \,\|\, \mathbf{X} \,\|\, \mathbf{I})\,;\, (\mathrm{AND} \,\|\, \mathrm{XOR})$$

Example: a half-adder

$\mathcal{G} = \{\mathrm{AND}, \mathrm{XOR}\}$



$$\mathsf{Half} := (\mathbf{Y} \,\|\, \mathbf{Y})\,;\, (\mathbf{I} \,\|\, \mathbf{X} \,\|\, \mathbf{I})\,;\, (\mathrm{AND} \,\|\, \mathrm{XOR})$$

Half is a combinational circuit with 2 inputs and 2 outputs, i.e. $\mathsf{Half} : 2 \otimes 2$.

Preliminary definitions

Alphabet, words, concatenation.

An alphabet Σ is a finite set of letters.
A word u over Σ is a finite sequence of letters $u = u_1 \cdots u_n$ where $u_i \in \Sigma$.
The set of words over Σ is noted Σ^*.
The integer n is the length of u and noted $|u|$.
The empty word is noted ϵ and is the unique word of length 0.
The set of words of length n is noted Σ^n.
The concatenation of $u = u_1 \cdots u_n$ and $v = v_1 \cdots v_m$ is defined $u \bullet v := u_1 \cdots u_n v_1 \cdots v_m$.

Formal semantics

C computes y on input x

The semantics of circuits is given by a relation of $\Sigma^* \times \Sigma^*$.

$$\frac{x \in \Sigma^* \qquad \mathcal{E}(g)(x) = y \in \Sigma^*}{g \Vdash x \rightsquigarrow y}\ g \in \mathcal{G} \qquad \frac{}{\mathbf{0} \Vdash \epsilon \rightsquigarrow \epsilon}$$

$$\frac{}{\mathbf{I} \Vdash a \rightsquigarrow a}\ a \in \Sigma \qquad \frac{}{\mathbf{Y} \Vdash a \rightsquigarrow aa}\ a \in \Sigma \qquad \frac{}{\mathbf{X} \Vdash ab \rightsquigarrow ba}\ a, b \in \Sigma$$

$$\frac{P_1 \Vdash x_1 \rightsquigarrow y_1 \qquad P_2 \Vdash x_2 \rightsquigarrow y_2}{P_1 \parallel P_2 \Vdash x_1 \bullet x_2 \rightsquigarrow y_1 \bullet y_2} \qquad \frac{P_1 \Vdash x \rightsquigarrow y \qquad P_2 \Vdash y \rightsquigarrow z}{P_1 \,;\, P_2 \Vdash x \rightsquigarrow z}$$

Formal semantics

C computes y on input x

The semantics of circuits is given by a relation of $\Sigma^* \times \Sigma^*$.

$$\frac{x \in \Sigma^* \qquad \mathcal{E}(g)(x) = y \in \Sigma^*}{g \Vdash x \rightsquigarrow y}\ g \in \mathcal{G} \qquad \frac{}{\mathbf{0} \Vdash \epsilon \rightsquigarrow \epsilon}$$

$$\frac{}{\mathbf{I} \Vdash a \rightsquigarrow a}\ a \in \Sigma \qquad \frac{}{\mathbf{Y} \Vdash a \rightsquigarrow aa}\ a \in \Sigma \qquad \frac{}{\mathbf{X} \Vdash ab \rightsquigarrow ba}\ a, b \in \Sigma$$

$$\frac{P_1 \Vdash x_1 \rightsquigarrow y_1 \qquad P_2 \Vdash x_2 \rightsquigarrow y_2}{P_1 \parallel P_2 \Vdash x_1 \bullet x_2 \rightsquigarrow y_1 \bullet y_2} \qquad \frac{P_1 \Vdash x \rightsquigarrow y \qquad P_2 \Vdash y \rightsquigarrow z}{P_1 \,;\, P_2 \Vdash x \rightsquigarrow z}$$

$\mathcal{E}(g)$ is a partial function $\Sigma^* \rightharpoonup \Sigma^*$, defined consistently w.r.t. the typing function.

Formal semantics

C computes y on input x

The semantics of circuits is given by a relation of $\Sigma^* \times \Sigma^*$.

$$\frac{x \in \Sigma^* \qquad \mathcal{E}(g)(x) = y \in \Sigma^*}{g \Vdash x \rightsquigarrow y}\ g \in \mathcal{G} \qquad \frac{}{\mathbf{0} \Vdash \epsilon \rightsquigarrow \epsilon}$$

$$\frac{}{\mathbf{I} \Vdash a \rightsquigarrow a}\ a \in \Sigma \qquad \frac{}{\mathbf{Y} \Vdash a \rightsquigarrow aa}\ a \in \Sigma \qquad \frac{}{\mathbf{X} \Vdash ab \rightsquigarrow ba}\ a, b \in \Sigma$$

$$\frac{P_1 \Vdash x_1 \rightsquigarrow y_1 \qquad P_2 \Vdash x_2 \rightsquigarrow y_2}{P_1 \parallel P_2 \Vdash x_1 \bullet x_2 \rightsquigarrow y_1 \bullet y_2} \qquad \frac{P_1 \Vdash x \rightsquigarrow y \qquad P_2 \Vdash y \rightsquigarrow z}{P_1 \,;\, P_2 \Vdash x \rightsquigarrow z}$$

$$P \simeq Q \iff \forall x, y : P \Vdash x \rightsquigarrow y \iff Q \Vdash x \rightsquigarrow y$$

Structural congruence

$\equiv$ identifies circuits that only differ in some minor wiring details.

It is the smallest congruence that satisfies the following equations:

- $(P_1 \| P_2) \| P_3 \equiv P_1 \| (P_2 \| P_3)$
- $P \| \mathbf{0} \equiv \mathbf{0} \| P \equiv P$

Structural congruence

$\equiv$ identifies circuits that only differ in some minor wiring details.

It is the smallest congruence that satisfies the following equations:

- $(P_1 \parallel P_2) \parallel P_3 \equiv P_1 \parallel (P_2 \parallel P_3)$
- $P \parallel \mathbf{0} \equiv \mathbf{0} \parallel P \equiv P$
- $(P_1 \,;\, P_2) \,;\, P_3 \equiv P_1 \,;\, (P_2 \,;\, P_3)$
- If $P : n \otimes m$ then $\mathbf{I}^n \,;\, P \equiv P \,;\, \mathbf{I}^m \equiv P$

Structural congruence

$\equiv$ identifies circuits that only differ in some minor wiring details.
It is the smallest congruence that satisfies the following equations:

- $(P_1 \| P_2) \| P_3 \equiv P_1 \| (P_2 \| P_3)$
- $P \| \mathbf{0} \equiv \mathbf{0} \| P \equiv P$
- $(P_1 \,;\, P_2) \,;\, P_3 \equiv P_1 \,;\, (P_2 \,;\, P_3)$
- If $P : n \otimes m$ then $\mathbf{I}^n \,;\, P \equiv P \,;\, \mathbf{I}^m \equiv P$
- If $P_1 : n \otimes m$ and $P_2 : m \otimes p$ then
  $(P_1 \,;\, P_2) \| (P_3 \,;\, P_4) \equiv (P_1 \| P_3) \,;\, (P_2 \| P_4)$

Structural congruence

$\equiv$ identifies circuits that only differ in some minor wiring details.
It is the smallest congruence that satisfies the following equations:

- $(P_1 \| P_2) \| P_3 \equiv P_1 \| (P_2 \| P_3)$
- $P \| \mathbf{0} \equiv \mathbf{0} \| P \equiv P$
- $(P_1 \,;\, P_2) \,;\, P_3 \equiv P_1 \,;\, (P_2 \,;\, P_3)$
- If $P : n \otimes m$ then $\mathbf{I}^n \,;\, P \equiv P \,;\, \mathbf{I}^m \equiv P$
- If $P_1 : n \otimes m$ and $P_2 : m \otimes p$ then
  $(P_1 \,;\, P_2) \| (P_3 \,;\, P_4) \equiv (P_1 \| P_3) \,;\, (P_2 \| P_4)$
- $\mathbf{Y} \,;\, (\mathbf{I} \| \mathbf{Y}) \equiv \mathbf{Y} \,;\, (\mathbf{Y} \| \mathbf{I})$
- $\mathbf{Y} \,;\, \mathbf{X} \equiv \mathbf{Y}$
- $\mathbf{X} \,;\, \mathbf{X} \equiv \mathbf{I} \| \mathbf{I}$
- $\mathbf{X} \,;\, (\mathbf{Y} \| \mathbf{Y}) \equiv (\mathbf{Y} \| \mathbf{Y}) \,;\, (\mathbf{I} \| \mathbf{X} \| \mathbf{I}) \,;\, (\mathbf{X} \| \mathbf{X}) \,;\, (\mathbf{I} \| \mathbf{X} \| \mathbf{I})$

Some results

- If $P : n \otimes m$ and $P : n' \otimes m'$ then $n = n'$ and $m = m'$.
- If $P \equiv Q$ then $P : n \otimes m \iff Q : n \otimes m$.
- If $P \Vdash x \rightsquigarrow y$ then $P : |x| \otimes |y|$.
- If $P : n \otimes m$ and $P \Vdash x \rightsquigarrow y$ then $|x| = n$ and $|y| = m$.
- If $P : n \otimes m$ then for any x such that $|x| = n$ there exists y such that $P \Vdash x \rightsquigarrow y$.
- If $P \Vdash x \rightsquigarrow y$ and $P \Vdash x \rightsquigarrow z$ then $y = z$.
- $\simeq$ is a congruence.
- If P and Q are ill-formed then $P \simeq Q$.
- $\equiv \subseteq \simeq$.

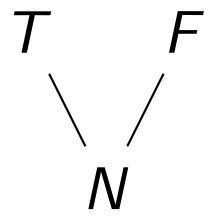
THE TRUSTED COMPUTING COMPANY

Definitions

- In the following, let $\Sigma = \{0, 1\}$.
- We pose $T = 10$, $F = 01$, $N = 00$ et $E = 11$.
  $\mathrm{NULL} = \{N\}$, $\mathrm{VALID} = \{T, F\}$, $\mathrm{FAULT} = \{E\}$.

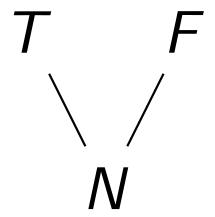
Definitions

- In the following, let $\Sigma = \{0, 1\}$.
- We pose $T = 10$, $F = 01$, $N = 00$ et $E = 11$.
  $\mathrm{NULL} = \{N\}$, $\mathrm{VALID} = \{T, F\}$, $\mathrm{FAULT} = \{E\}$.
- Let $\preceq$ be the partial order defined by:



Definitions

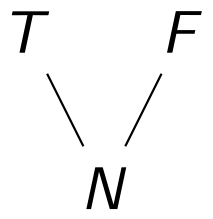
- In the following, let $\Sigma = \{0, 1\}$.
- We pose $T = 10$, $F = 01$, $N = 00$ et $E = 11$.
  $\mathrm{NULL} = \{N\}$, $\mathrm{VALID} = \{T, F\}$, $\mathrm{FAULT} = \{E\}$.
- Let $\preceq$ be the partial order defined by:



- Let $\sim$ be the equivalence relation on Σ^2 whose equivalence classes are NULL, VALID and FAULT.
- We extend these definitions to words of even length.

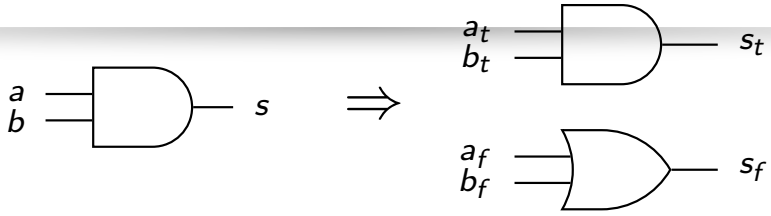
Definitions

- In the following, let $\Sigma = \{0, 1\}$.
- We pose $T = 10$, $F = 01$, $N = 00$ et $E = 11$.
  $\mathrm{NULL} = \{N\}$, $\mathrm{VALID} = \{T, F\}$, $\mathrm{FAULT} = \{E\}$.
- Let $\preceq$ be the partial order defined by:

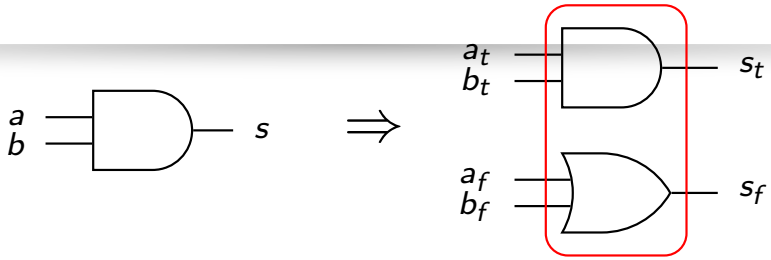


- Let $\sim$ be the equivalence relation on Σ^2 whose equivalence classes are NULL, VALID and FAULT.
- We extend these definitions to words of even length.
- For $u \in \Sigma^*$, we let $[u] \in \mathrm{VALID}^*$ be the corresponding word in dual-rail logic.
  example: $[0110] = FTTF = 01101001$

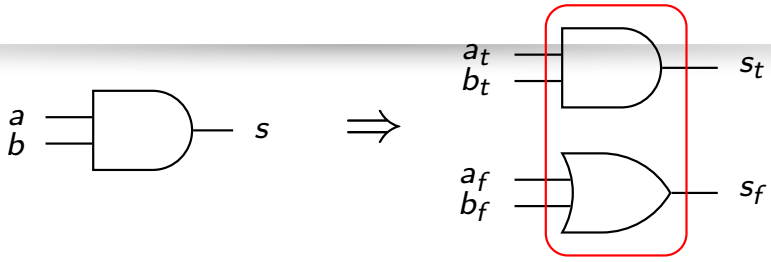
Transformation process

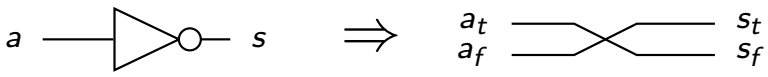


Transformation process

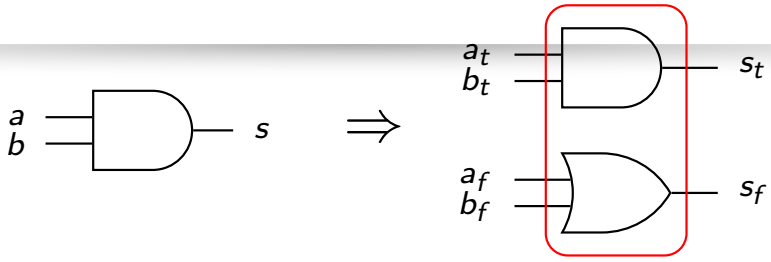


Transformation process

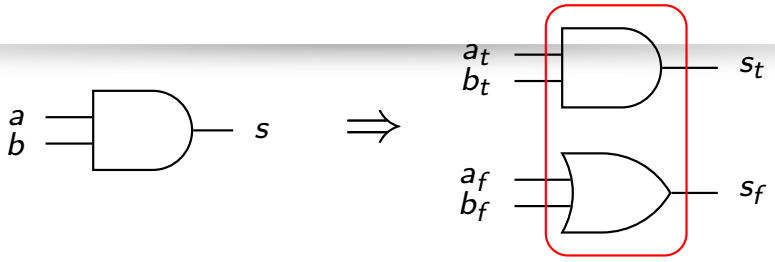


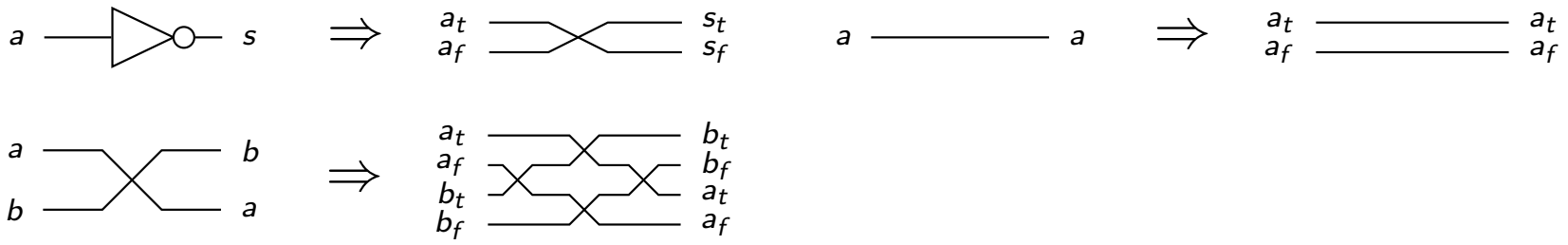


Transformation process

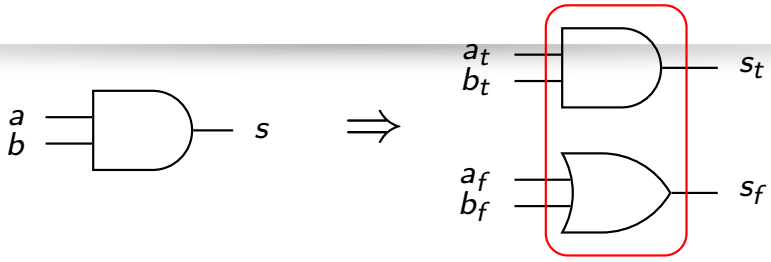


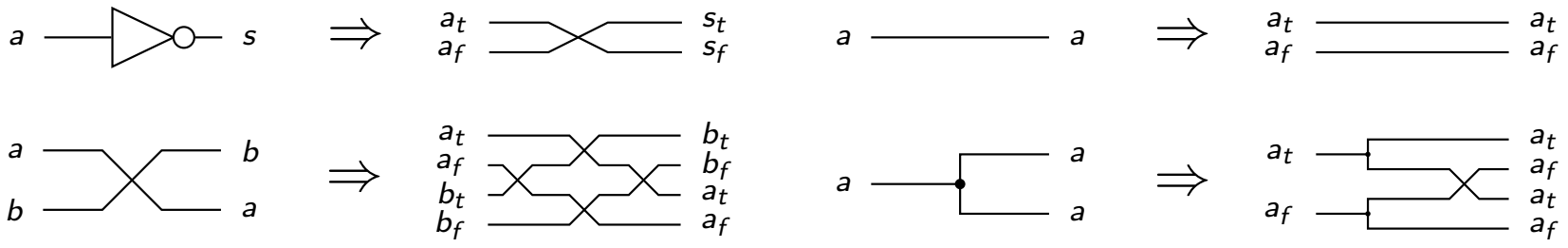
Transformation process



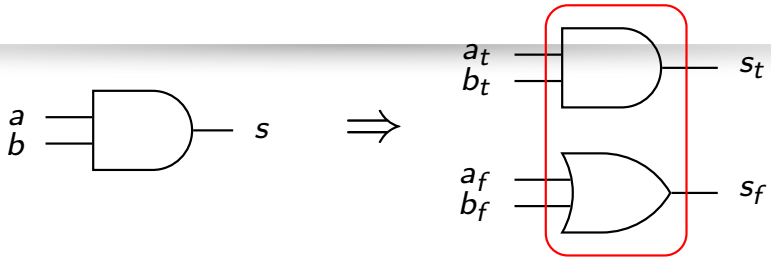


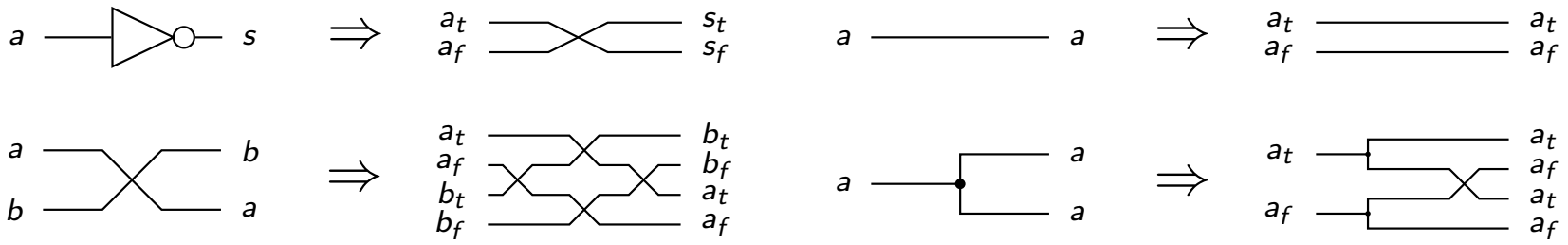
Transformation process

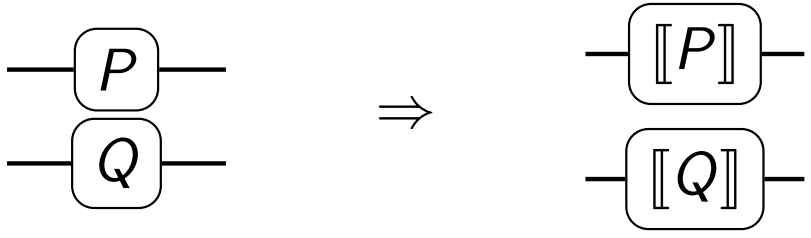




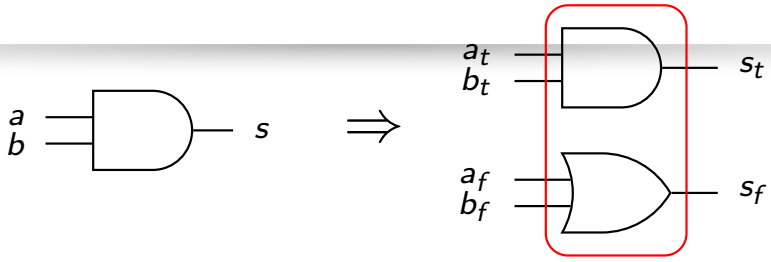
Transformation process

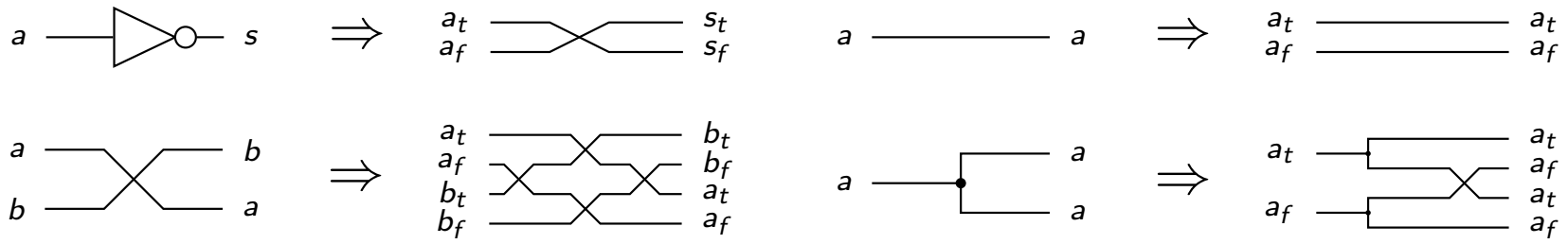


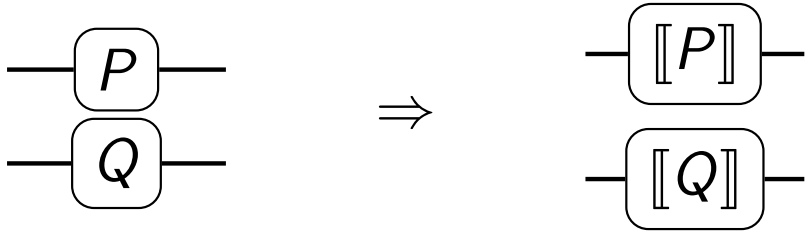


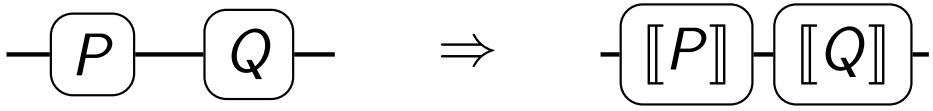


Transformation process









Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash [x] \rightsquigarrow [y]$.

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash [x] \rightsquigarrow [y]$.

No glitches are possible.

$$[\![P]\!] \Vdash x \rightsquigarrow y$$

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

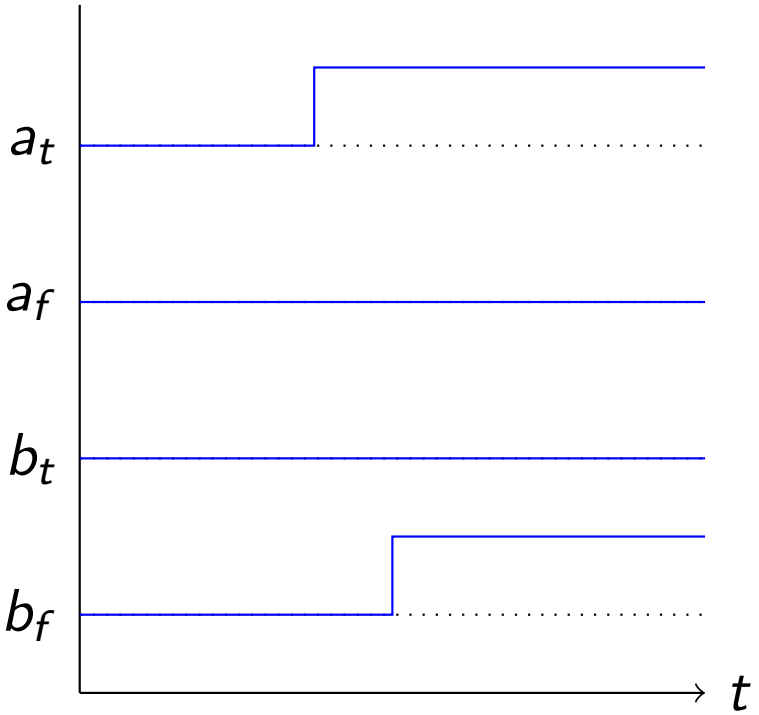
If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash [x] \rightsquigarrow [y]$.

No glitches are possible.

$$[\![P]\!] \Vdash x \rightsquigarrow y$$
$$\preceq$$
$$x'$$

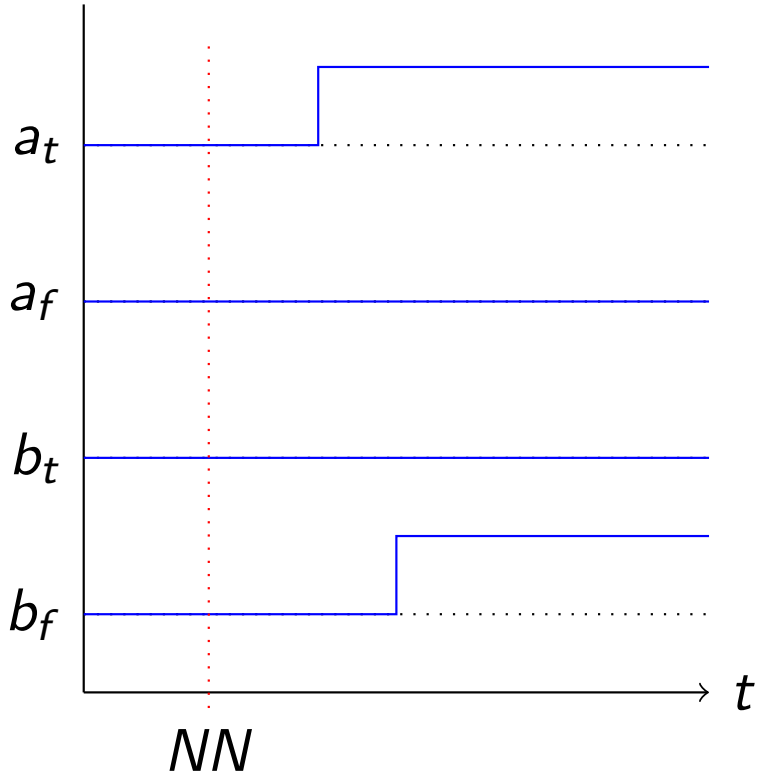
Interpretation of $\preceq$

$\preceq$ models the transition of signals from precharge state (NULL) to evaluation state (VALID).



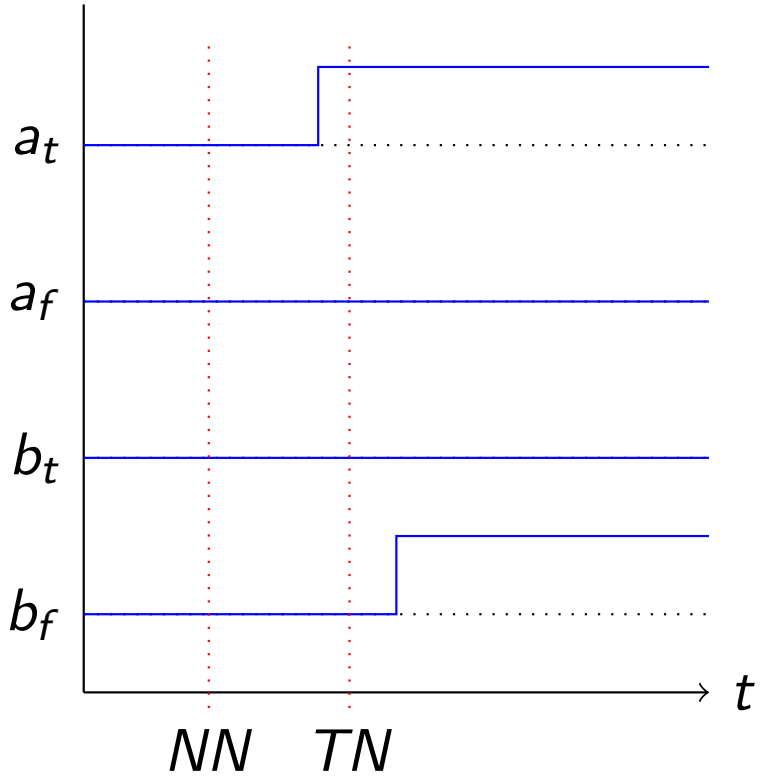
Interpretation of $\preceq$

$\preceq$ models the transition of signals from precharge state (NULL) to evaluation state (VALID).



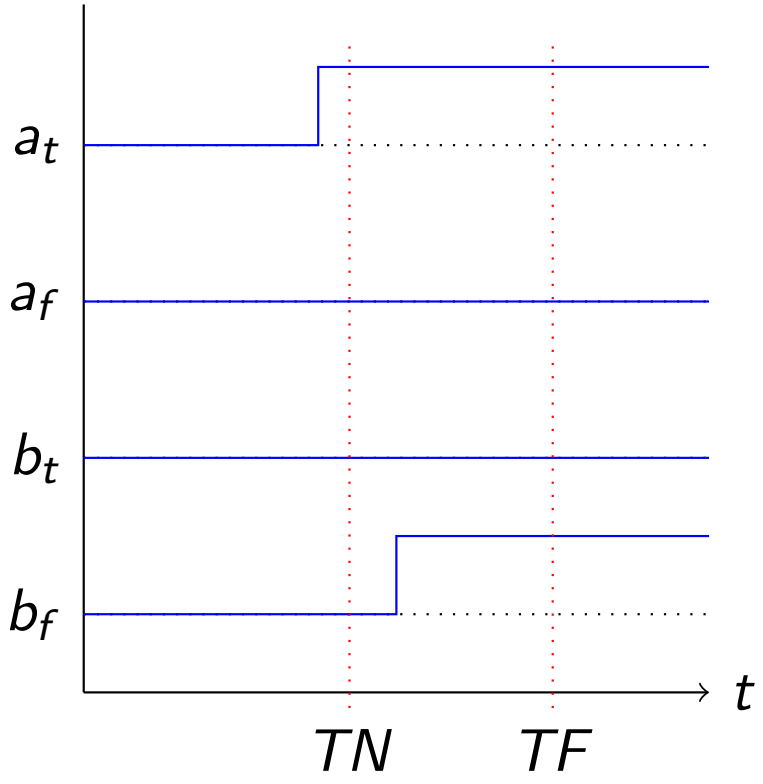
Interpretation of $\preceq$

$\preceq$ models the transition of signals from precharge state (NULL) to evaluation state (VALID).



Interpretation of $\preceq$

$\preceq$ models the transition of signals from precharge state (NULL) to evaluation state (VALID).



Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash [x] \rightsquigarrow [y]$.

No glitches are possible.

$$\begin{array}{c} [\![P]\!] \Vdash x \rightsquigarrow y \\ \preceq \\ x' \end{array}$$

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash [x] \rightsquigarrow [y]$.

No glitches are possible.

$$\begin{array}{c} [\![P]\!] \Vdash x \rightsquigarrow y \\ \preceq \\ [\![P]\!] \Vdash x' \rightsquigarrow y' \end{array}$$

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

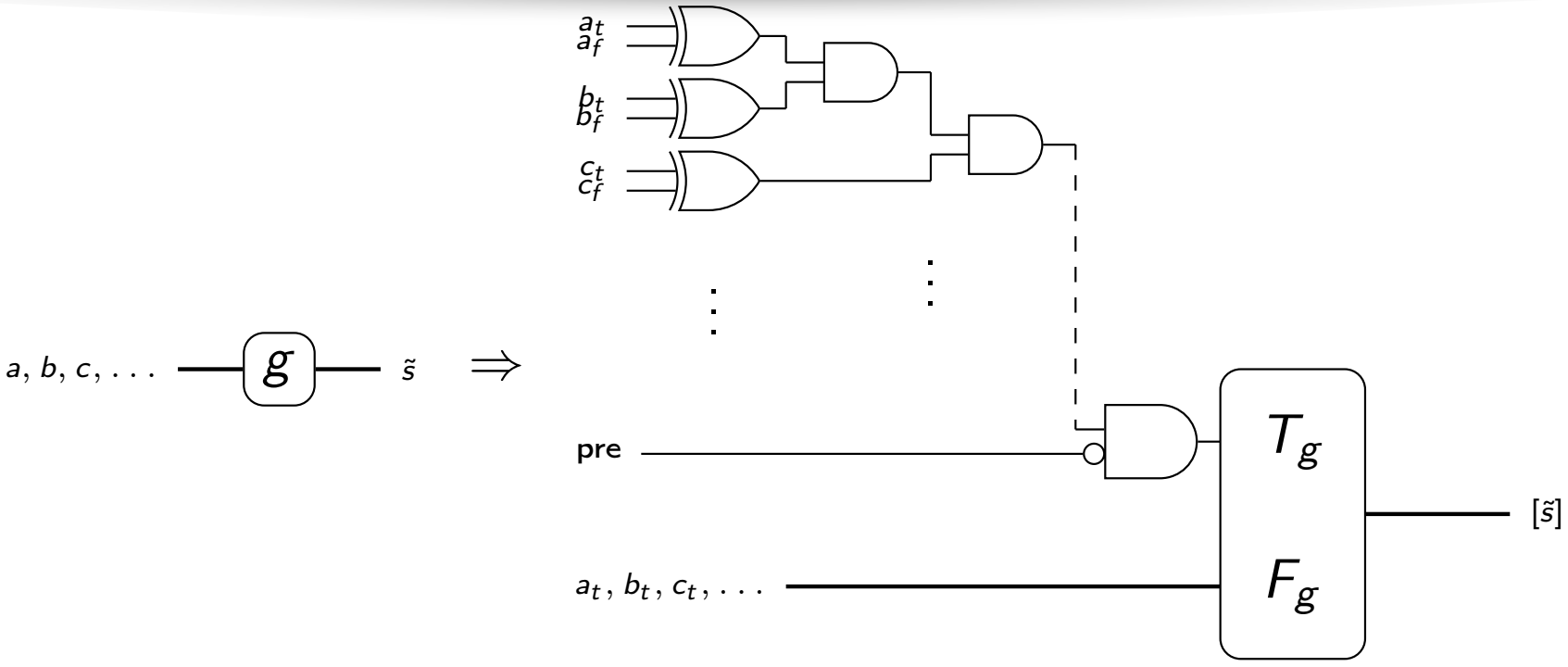
If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash [x] \rightsquigarrow [y]$.

No glitches are possible.

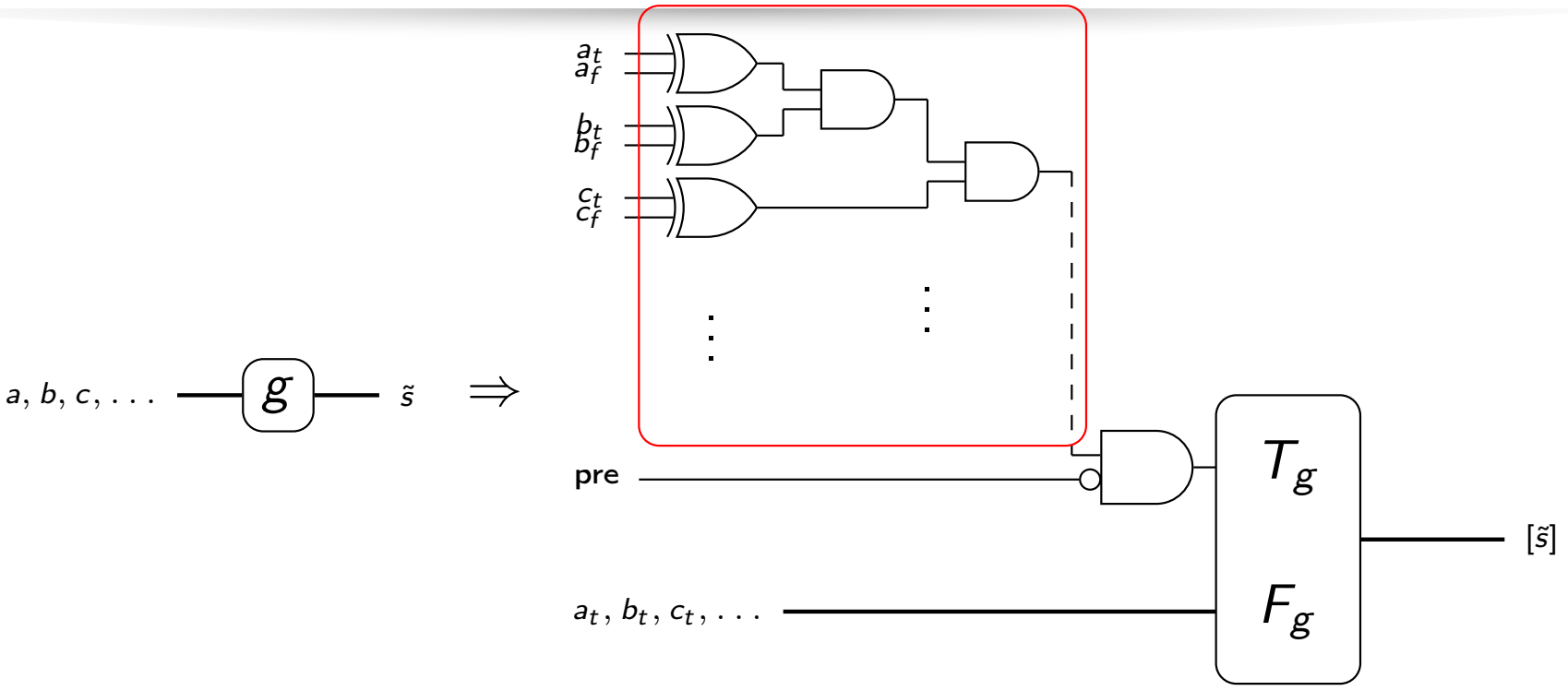
$$\begin{array}{ccc} [\![P]\!] \Vdash x & \rightsquigarrow & y \\ & \preceq \quad & \preceq \\ [\![P]\!] \Vdash x' & \rightsquigarrow & y' \end{array}$$

THE TRUSTED COMPUTING COMPANY

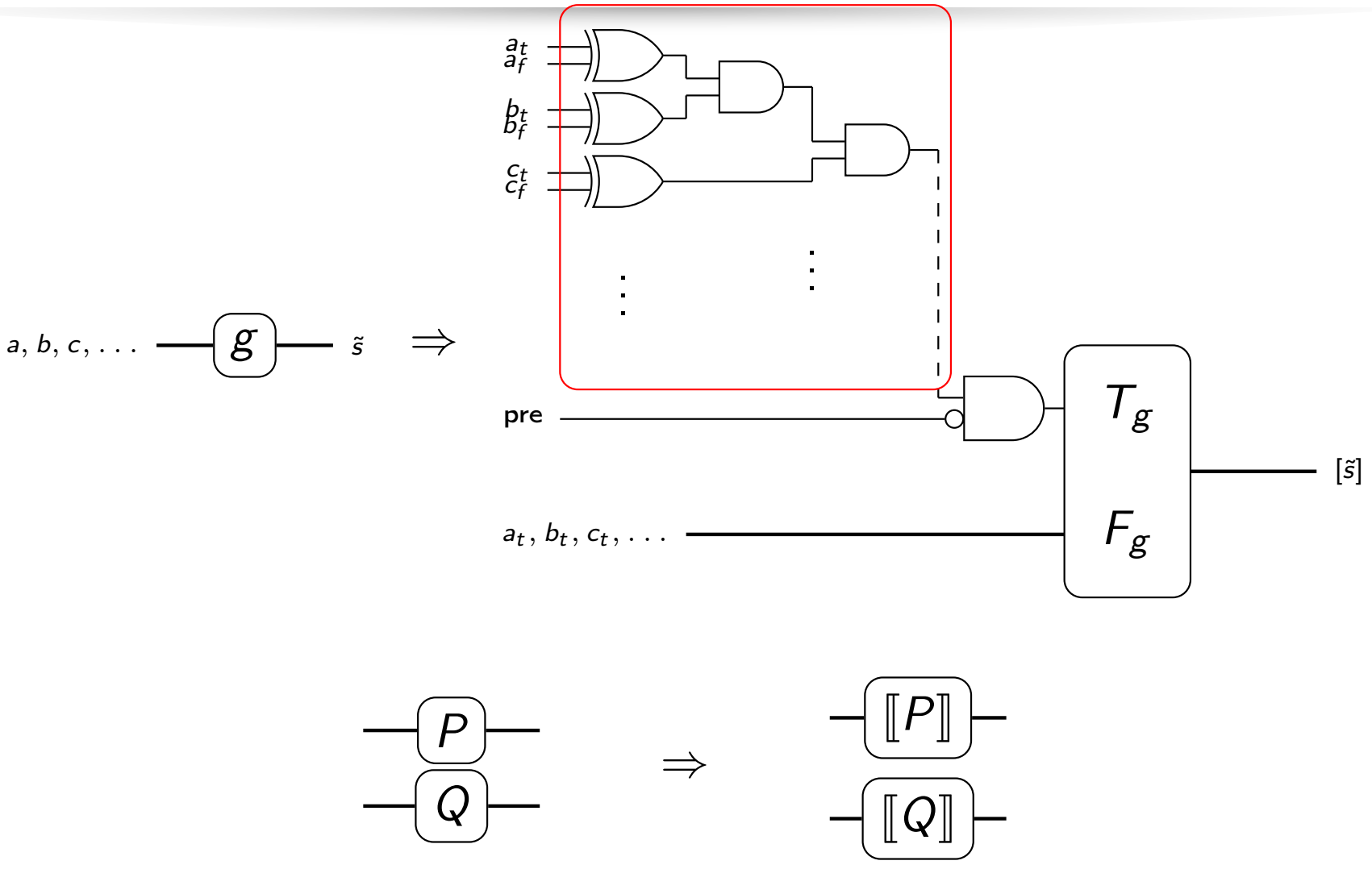
Transformation process



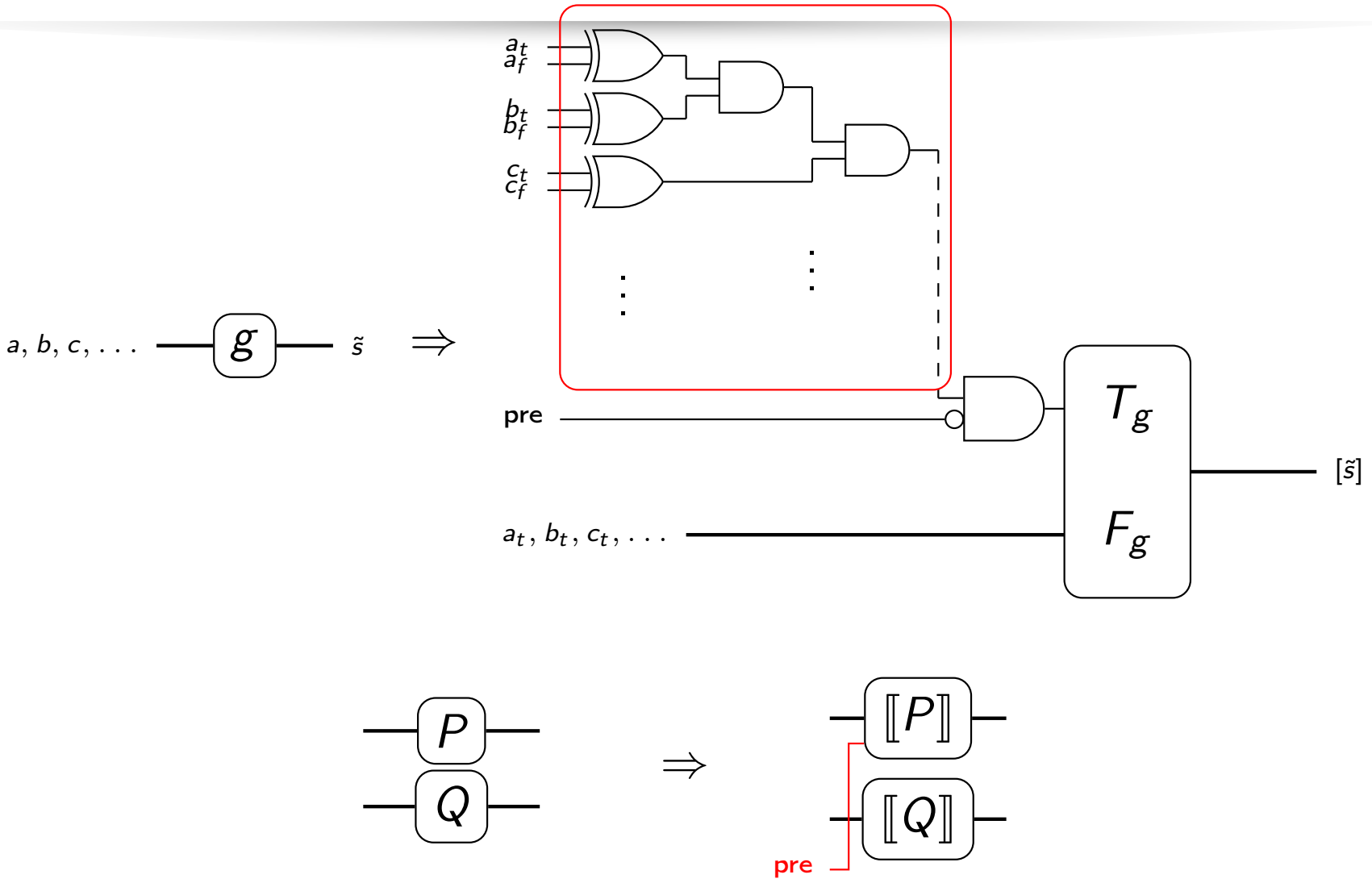
Transformation process



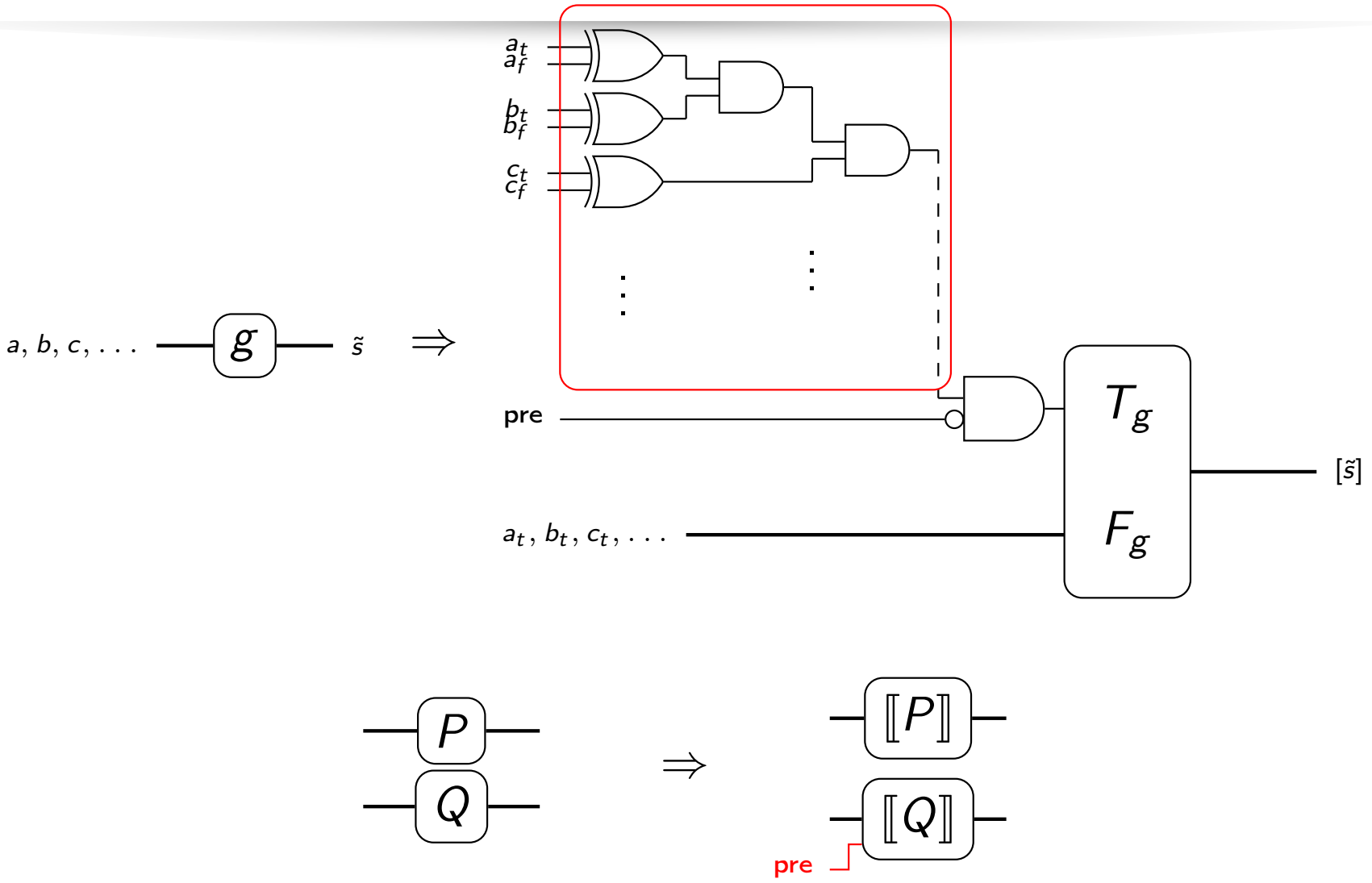
Transformation process



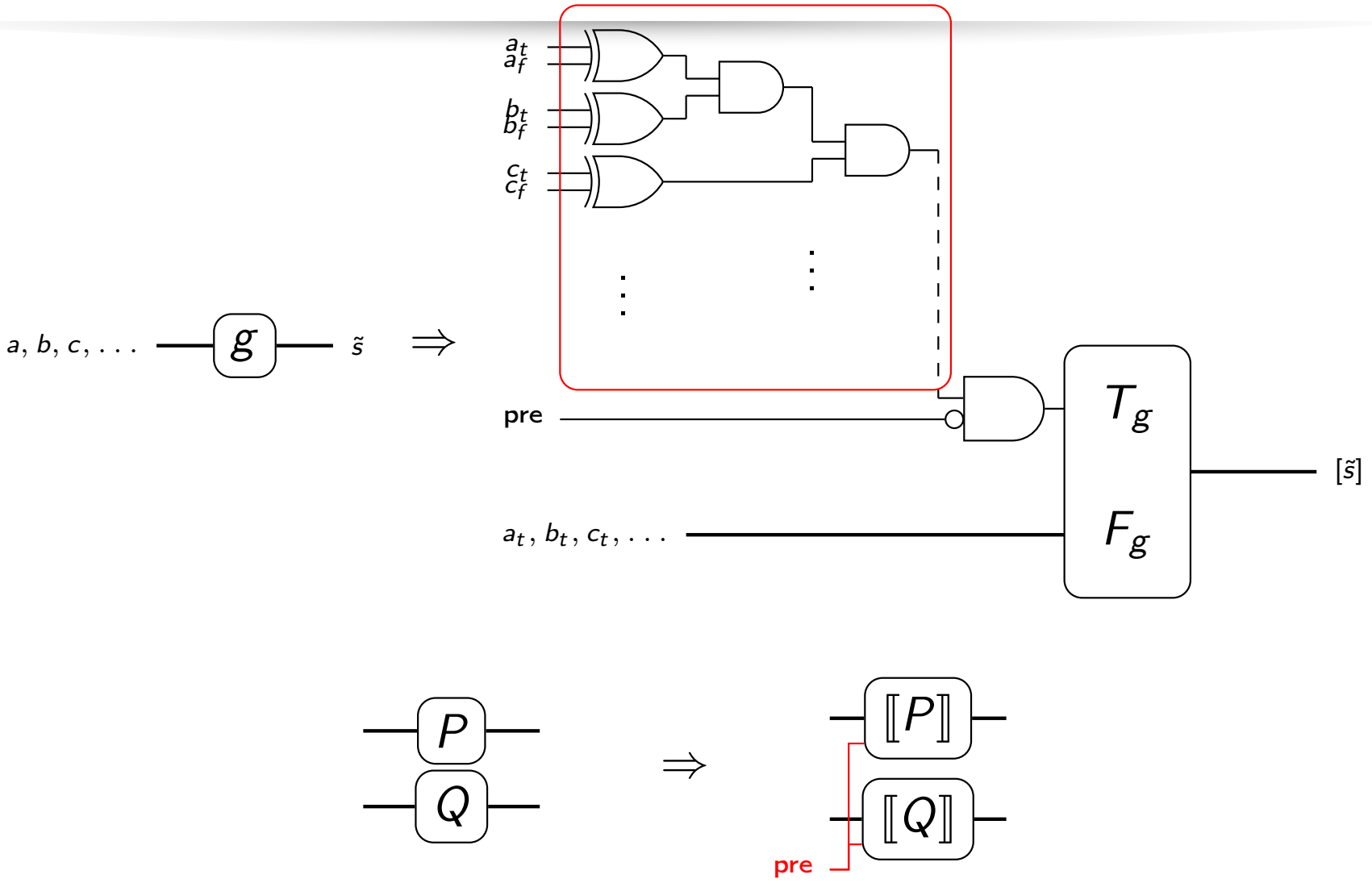
Transformation process



Transformation process



Transformation process



Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![g]\!] \Vdash 1x \rightsquigarrow y$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash \hat{1}x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![g]\!] \Vdash 1x \rightsquigarrow y$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash \hat{1}x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash \hat{0}[x] \rightsquigarrow [y]$.

Properties

$[\![P]\!]$ fulfils the DPL invariants.

- If $[\![g]\!] \Vdash 1x \rightsquigarrow y$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash \hat{1}x \rightsquigarrow y$ and $x \in \mathrm{NULL}^*$ then $y \in \mathrm{NULL}^*$.
- If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $x \in \mathrm{VALID}^*$ then $y \in \mathrm{VALID}^*$.

The transformation is sound.

If $P \Vdash x \rightsquigarrow y$ then $[\![P]\!] \Vdash \hat{0}[x] \rightsquigarrow [y]$.

No glitches are possible.

If $[\![P]\!] \Vdash \hat{p}x \rightsquigarrow \hat{p}y$, $x \preceq x'$ and $[\![P]\!] \Vdash \hat{p}x' \rightsquigarrow \hat{p}y'$ then $y \preceq y'$.

Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

The transformation is complete.

If $[\![P]\!] \Vdash \hat{0}x' \rightsquigarrow y'$ and $y' \in \mathrm{VALID}^*$ then
there exists x, y such that $x' = [x]$, $y' = [y]$ and $P \Vdash x \rightsquigarrow y$.
(provided that P does not contain gates with 0 outputs)

Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

The transformation is complete.

If $[\![P]\!] \Vdash \hat{0}x' \rightsquigarrow y'$ and $y' \in \mathrm{VALID}^*$ then
there exists x, y such that $x' = [x]$, $y' = [y]$ and $P \Vdash x \rightsquigarrow y$.
(provided that P does not contain gates with 0 outputs)

The secured circuit behaves the same on equivalent inputs.

$$[\![P]\!] \Vdash \hat{p}x \rightsquigarrow y$$

Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

The transformation is complete.

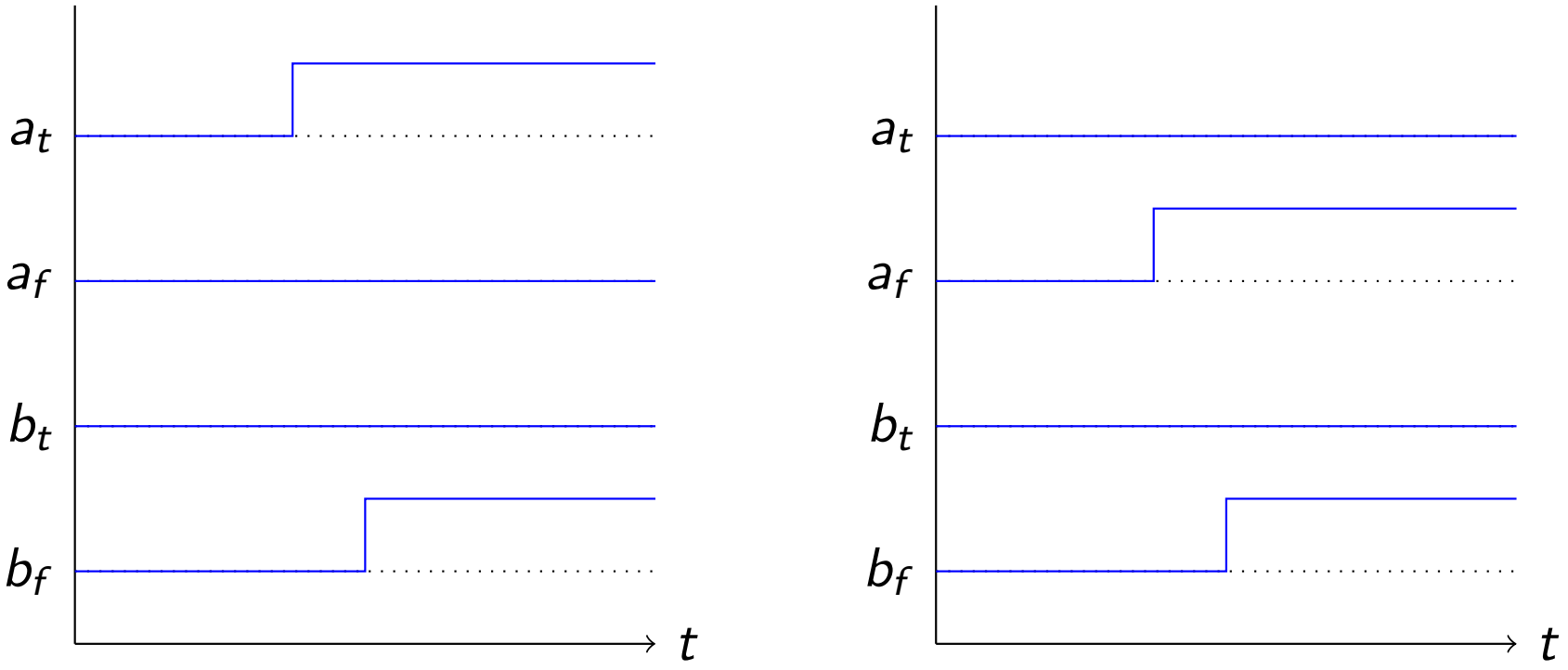
If $[\![P]\!] \Vdash \hat{0}x' \rightsquigarrow y'$ and $y' \in \mathrm{VALID}^*$ then
there exists x, y such that $x' = [x]$, $y' = [y]$ and $P \Vdash x \rightsquigarrow y$.
(provided that P does not contain gates with 0 outputs)

The secured circuit behaves the same on equivalent inputs.

$$[\![P]\!] \Vdash \hat{p}x \rightsquigarrow y$$
$$\sim$$
$$\hat{p}x'$$

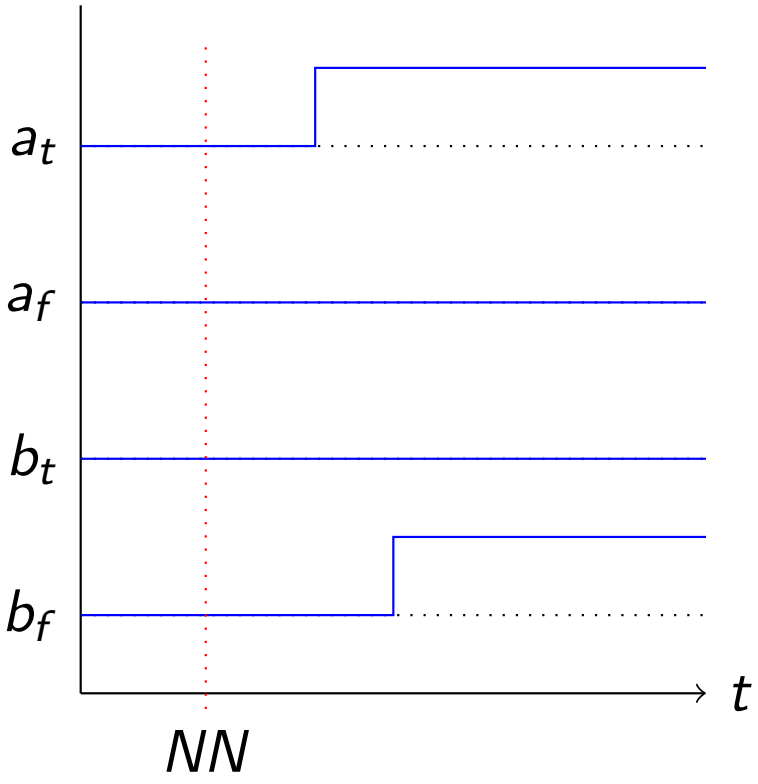
Interpretation of $\sim$

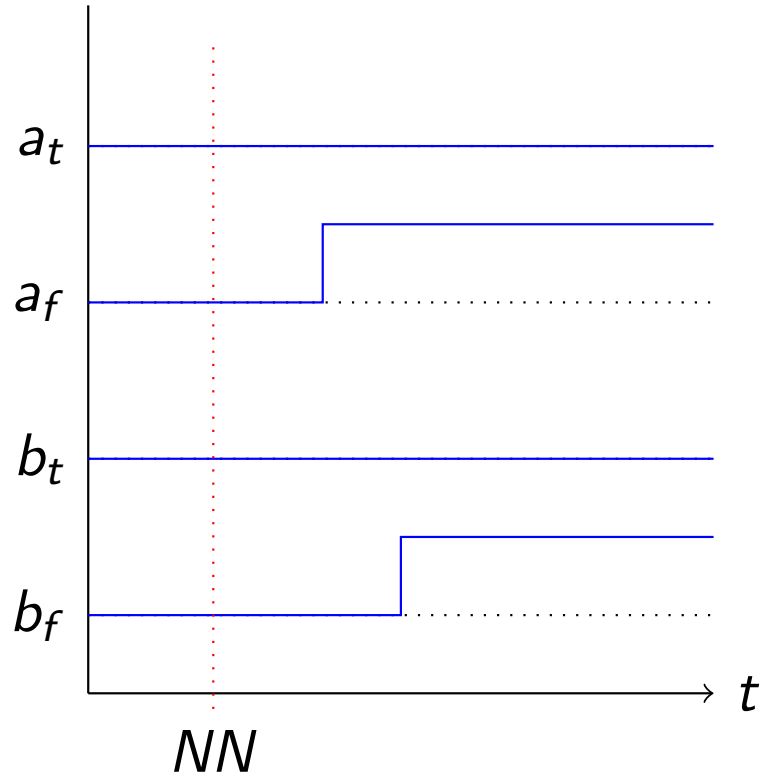
$\sim$ equates words which have the same amount of information, i.e. in which corresponding dual-rail signals have the same nature.



Interpretation of $\sim$

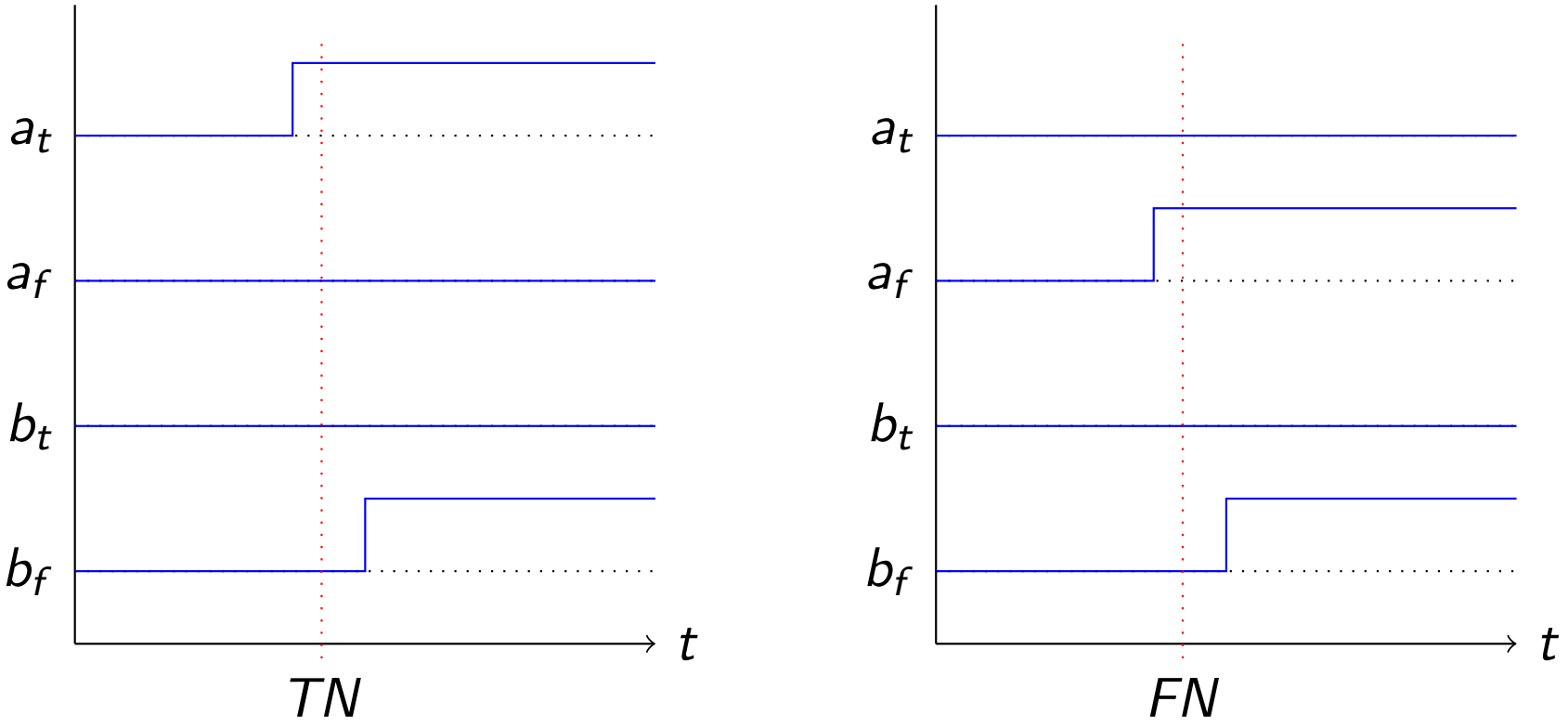
$\sim$ equates words which have the same amount of information, i.e. in which corresponding dual-rail signals have the same nature.





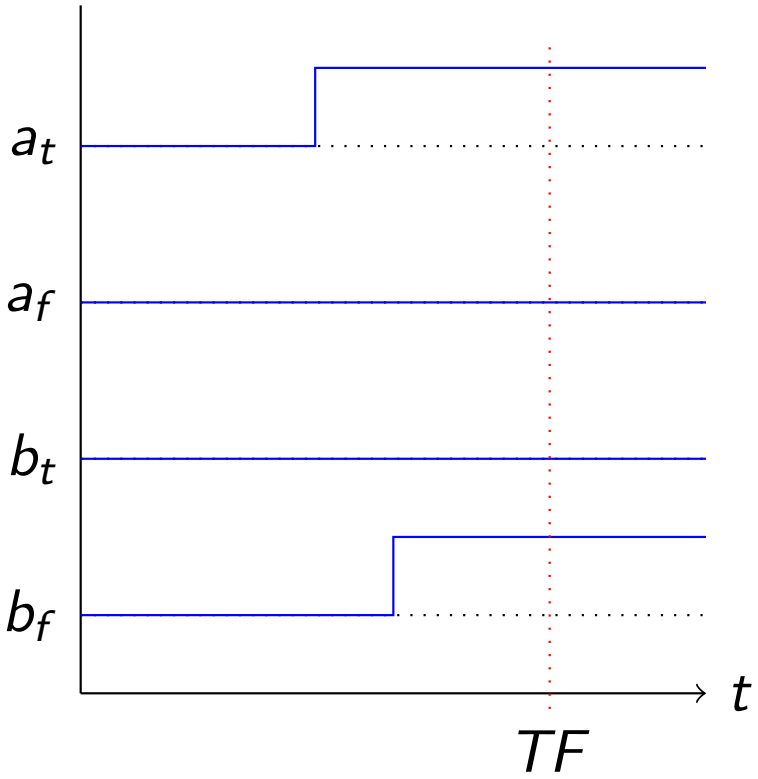
Interpretation of $\sim$

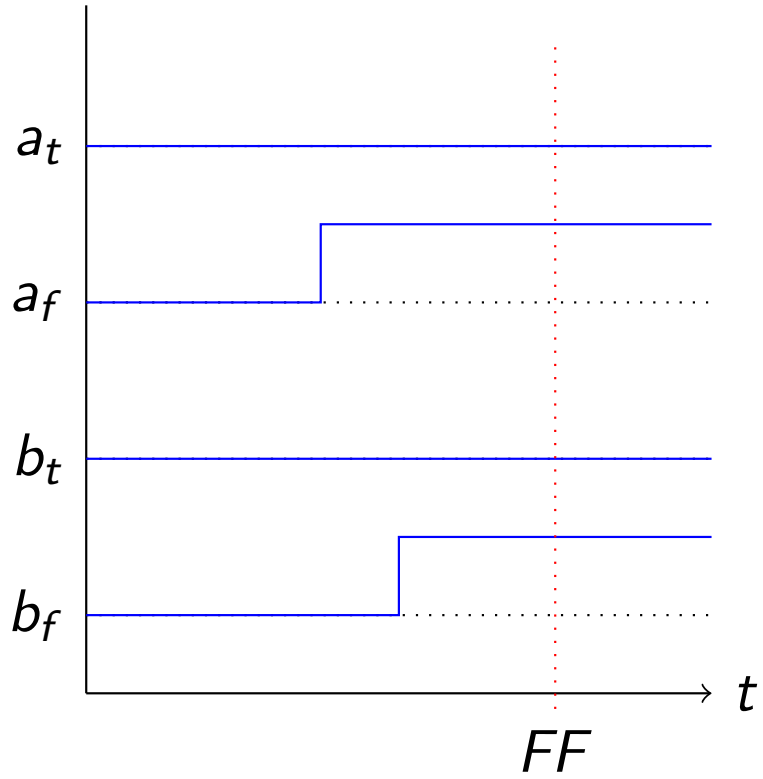
$\sim$ equates words which have the same amount of information, i.e. in which corresponding dual-rail signals have the same nature.



Interpretation of $\sim$

$\sim$ equates words which have the same amount of information, i.e. in which corresponding dual-rail signals have the same nature.





Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

The transformation is complete.

If $[\![P]\!] \Vdash \hat{0}x' \rightsquigarrow y'$ and $y' \in \mathrm{VALID}^*$ then
there exists x, y such that $x' = [x]$, $y' = [y]$ and $P \Vdash x \rightsquigarrow y$.
(provided that P does not contain gates with 0 outputs)

The secured circuit behaves the same on equivalent inputs.

$$[\![P]\!] \Vdash \hat{p}x \rightsquigarrow y$$
$$\sim$$
$$\hat{p}x'$$

Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

The transformation is complete.

If $[\![P]\!] \Vdash \hat{0}x' \rightsquigarrow y'$ and $y' \in \mathrm{VALID}^*$ then
there exists x, y such that $x' = [x]$, $y' = [y]$ and $P \Vdash x \rightsquigarrow y$.
(provided that P does not contain gates with 0 outputs)

The secured circuit behaves the same on equivalent inputs.

$$[\![P]\!] \Vdash \hat{p}x \rightsquigarrow y$$
$$\sim$$
$$[\![P]\!] \Vdash \hat{p}x' \rightsquigarrow y'$$

Properties

There is no early-evaluation.

If $[\![P]\!] \Vdash \hat{0}x \rightsquigarrow y$ and $y \in \mathrm{VALID}^*$ then $x \in \mathrm{VALID}^*$.
(provided that P does not contain gates with 0 outputs)

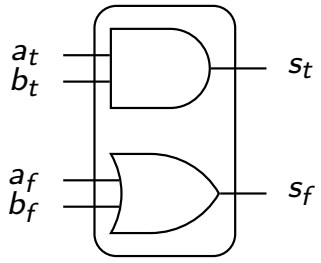
The transformation is complete.

If $[\![P]\!] \Vdash \hat{0}x' \rightsquigarrow y'$ and $y' \in \mathrm{VALID}^*$ then
there exists x, y such that $x' = [x]$, $y' = [y]$ and $P \Vdash x \rightsquigarrow y$.
(provided that P does not contain gates with 0 outputs)

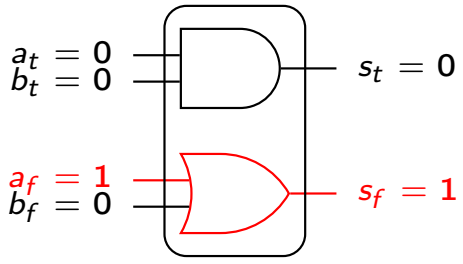
The secured circuit behaves the same on equivalent inputs.

$$\begin{array}{ccc} [\![P]\!] \Vdash \hat{p}x & \rightsquigarrow & y \\ \sim & & \sim \\ [\![P]\!] \Vdash \hat{p}x' & \rightsquigarrow & y' \end{array}$$

Back to WDDL



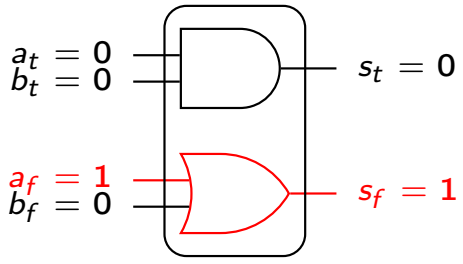
Back to WDDL



The $\mathrm{AND}_{\mathrm{WDDL}}$ gate suffers from early-evaluation.

$x = 0100 \notin \mathrm{VALID}^*$ and $y = 01 \in \mathrm{VALID}^*$

Back to WDDL



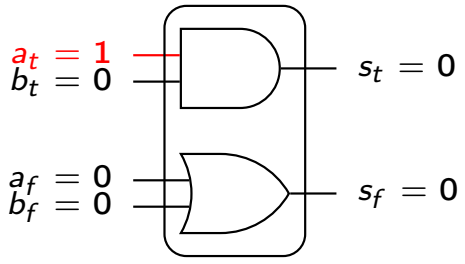
The $\mathrm{AND}_{\mathrm{WDDL}}$ gate suffers from early-evaluation.

$x = 0100 \notin \mathrm{VALID}^*$ and $y = 01 \in \mathrm{VALID}^*$

The $\mathrm{AND}_{\mathrm{WDDL}}$ gate behaves differently on equivalent inputs.

$x = 0100$ and $y = 01$

Back to WDDL



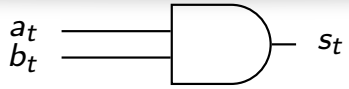
The $\mathrm{AND}_{\mathrm{WDDL}}$ gate suffers from early-evaluation.

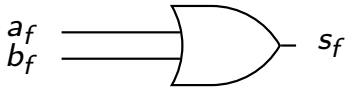
$x = 0100 \notin \mathrm{VALID}^*$ and $y = 01 \in \mathrm{VALID}^*$

The $\mathrm{AND}_{\mathrm{WDDL}}$ gate behaves differently on equivalent inputs.

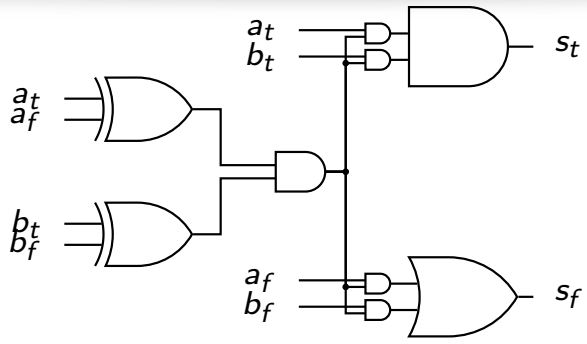
$x = 0100 \sim x' = 1000$ and $y = 01 \nsim y' = 00$

Discussion



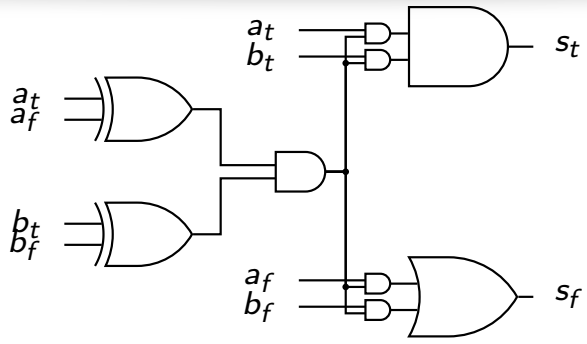


Discussion



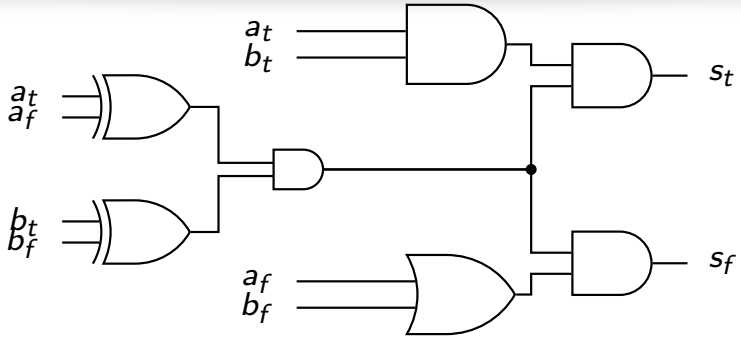
- BCDL fixes WDDL by adding a synchronisation barrier.

Discussion



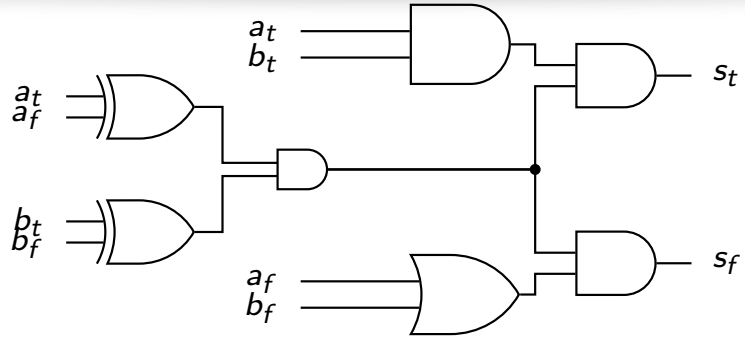
- BCDL fixes WDDL by adding a synchronisation barrier.
- How to address the race between the synchronisation signal and the data signals? (DRSL vulnerability)

Discussion



- BCDL fixes WDDL by adding a synchronisation barrier.
- How to address the race between the synchronisation signal and the data signals? (DRSL vulnerability)
- How to discriminate this circuit?

Discussion



- BCDL fixes WDDL by adding a synchronisation barrier.
- How to address the race between the synchronisation signal and the data signals? (DRSL vulnerability)
- How to discriminate this circuit?
  Measure the activity of circuits and show that the activity of a circuit is constant on equivalent inputs, i.e.

$$x \sim x' \Rightarrow \mu_C(x) = \mu_C(x')$$

Summary

- We defined a calculus to describe combinational circuits.

Summary

- We defined a calculus to describe combinational circuits.
- We defined formally WDDL and BCDL securisation process.

Summary

- We defined a calculus to describe combinational circuits.
- We defined formally WDDL and BCDL securisation process.
- We proved the correctness of these two transformations.

Summary

- We defined a calculus to describe combinational circuits.
- We defined formally WDDL and BCDL securisation process.
- We proved the correctness of these two transformations.
- Regarding security properties, we identified some necessary conditions to fulfil.

Perspectives

- Apply the model to other dual-rail styles.

Perspectives

- Apply the model to other dual-rail styles.
- Refine the model.

The End

Thank You

Rotations

On words

- $\overrightarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overrightarrow{ua} = au$
- $\overleftarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overleftarrow{au} = ua$

Rotations

On words

- $\overrightarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overrightarrow{ua} = au$
- $\overleftarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overleftarrow{au} = ua$

We define by induction on $n \in \mathbb{N}$ the circuit ror_n:

$$\begin{aligned}\mathrm{ror}_0 &:= \mathbf{0}\\ \mathrm{ror}_1 &:= \mathbf{I}\\ \mathrm{ror}_{n+2} &:= (\mathbf{I}^n \parallel \mathbf{X})\,;\,(\mathrm{ror}_{n+1} \parallel \mathbf{I})\end{aligned}$$

We have

$$\mathrm{ror}_n \Vdash x \rightsquigarrow y \iff |x| = n \wedge y = \overrightarrow{x}$$

Rotations

On words

- $\overrightarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overrightarrow{ua} = au$
- $\overleftarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overleftarrow{au} = ua$

We define by induction on $n \in \mathbb{N}$ the circuit rol_n:

$$\begin{aligned}\mathrm{rol}_0 &:= \mathbf{0}\\ \mathrm{rol}_1 &:= \mathbf{I}\\ \mathrm{rol}_{n+2} &:= (\mathrm{rol}_{n+1} \parallel \mathbf{I})\,;\,(\mathbf{I}^n \parallel \mathbf{X})\end{aligned}$$

We have

$$\mathrm{rol}_n \Vdash x \rightsquigarrow y \iff |x| = n \wedge y = \overleftarrow{x}$$

Rotations

On words

- $\overrightarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overrightarrow{ua} = au$
- $\overleftarrow{\epsilon} = \epsilon$ and, for $a \in \Sigma, u \in \Sigma^*$, $\overleftarrow{au} = ua$

We also have that:

- $\mathrm{ror}_n \,;\, \mathrm{rol}_n \equiv \mathsf{I}^n$
- $\mathrm{rol}_n \,;\, \mathrm{ror}_n \equiv \mathsf{I}^n$

Interleaving

On words

$\epsilon \,|||\, \epsilon := \epsilon$ and for $a, b \in \Sigma,\ u, v \in \Sigma^*,\ (au) \,|||\, (bv) := ab(u \,|||\, v)$

Interleaving

On words

$\epsilon \mid\mid\mid \epsilon := \epsilon$ and for $a, b \in \Sigma$, $u, v \in \Sigma^*$, $(au) \mid\mid\mid (bv) := ab(u \mid\mid\mid v)$

We define by induction on $n \in \mathbb{N}$ the circuit int_n:

$$\mathrm{int}_0 := \mathbf{0}$$
$$\mathrm{int}_{n+1} := (\mathbf{I} \| \mathrm{ror}_{n+1} \| \mathbf{I}^n) \,;\, (\mathbf{I} \| \mathbf{I} \| \mathrm{int}_n)$$

We have

$$\mathrm{int}_n \Vdash x \rightsquigarrow y \iff x = u \bullet v \wedge |u| = |v| = n \wedge y = u \mid\mid\mid v$$

Interleaving

On words

$\epsilon \mid\!\mid\!\mid \epsilon := \epsilon$ and for $a, b \in \Sigma$, $u, v \in \Sigma^*$, $(au) \mid\!\mid\!\mid (bv) := ab(u \mid\!\mid\!\mid v)$

We define by induction on $n \in \mathbb{N}$ the circuit unint_n:

$$\mathrm{unint}_0 := \mathbf{0}$$
$$\mathrm{unint}_{n+1} := (\mathbf{I} \| \mathbf{I} \| \mathrm{unint}_n) \,;\, (\mathbf{I} \| \mathrm{rol}_{n+1} \| \mathbf{I}^n)$$

We have

$$\mathrm{unint}_n \Vdash x \rightsquigarrow y \iff y = u \bullet v \wedge |u| = |v| = n \wedge x = u \mid\!\mid\!\mid v$$

Interleaving

On words

$\epsilon \,|||\, \epsilon := \epsilon$ and for $a, b \in \Sigma$, $u, v \in \Sigma^*$, $(au) \,|||\, (bv) := ab(u \,|||\, v)$

We also have that:

- $\mathrm{int}_n \,;\, \mathrm{unint}_n \equiv \mathsf{I}^{2n}$
- $\mathrm{unint}_n \,;\, \mathrm{int}_n \equiv \mathsf{I}^{2n}$