

A systematic approach to the description of fault-tolerant systems

Jakub Lojda


Supervisor: Doc. Ing. Zdeněk Kotásek, CSc.

Brno University of Technology, Faculty of Information Technology
Božetěchova 1/2, 612 66 Brno - Královo Pole
ilojda@fit.vutbr.cz



December 17, 2015

- Fault-tolerance
- Formal definition of digital system
- Algorithms for transformation of digital system to its fault-tolerant version
 - TMR (Triple Modular Redundancy)
 - DwC (Duplex with Comparison)

- Raising the level of integration on chip
 - Increasing pressure on reliability
- A light blue downward-pointing arrow indicating a logical flow or consequence from the previous points.
- Implementing fault-tolerance into digital circuits
-
- **Definition 1.1:** Fault tolerant system is a system, that has the capability to perform its function even in the presence of *faults or errors*.

- **Definition 1.2:** Fault is a phenomenon of losing a capability to perform a function.
- **Definition 1.3:** Error is a concept of dissimilarity of measured value and proper value.

- **TMR (Triple Modular Redundancy)**
 - The main circuit is triplicated
 - New component added – voter
 - voter performs majority function
- **DwC (Duplication with Comparison)**
 - The main circuit is duplicated
 - New component added – comparator
 - Basically exclusive or (XOR) per output

(and many others...)

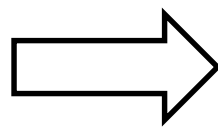
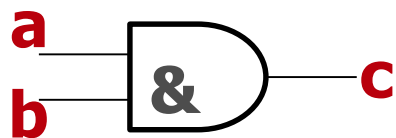
- Digital system is generally composed of *pins*, *signals* and *components* performing a particular function.
- **Definition 2.1:** *Pin* is a place of a circuit allowing communication of a particular component with its neighborhood. Pin is represented by a *symbol* from *alphabet of pins*.
- **Definition 2.2:** *Signal* is a wire connecting pins of components. Signal is a pair (p_1, p_2) , $p_1, p_2 \in P$, P is an *alphabet of pins*.

- **Definition 2.3:** Let

- I be a finite set of input pins,
- O be a finite set of output pins,
- T a fully defined logic table of logic gate with $|I|$ inputs, $|O|$ outputs and 2^s state variables, $s \in \mathbb{N}_0$,

then an ordered triple $G = (I, O, T)$ is a basic logic gate.

- Example of basic logic gate G_{AND}



$$I = \{a, b\}$$

$$O = \{c\}$$

T

a	b	c
0	0	0
0	1	0
1	0	0
1	1	1

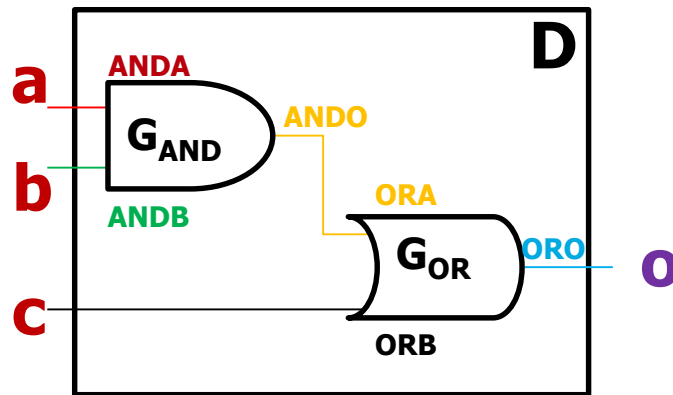
- **Definition 2.4:** Let

- I be a finite set of input pins,
- O be a finite set of output pins,
 - I and O are mutually exclusive, $I \cup O$ form *alphabet of pins*,
- S be a finite set of connecting signals,
- U be a finite set of components performing a particular function, a component can be
 - a digital system,
 - a basic logic gate,

then an ordered quadruple $D = (I, O, S, U)$ is a digital system.

- Recursion allows us to choose a level of abstraction.

- Example of a simple digital system $D = (I, O, S, U)$ performing the function $o = (a \wedge b) \vee c$



$$I = \{a, b, c\}$$

$$O = \{o\}$$

$$S = \{(a, \text{AND}A), (b, \text{AND}B), (\text{AND}O, \text{OR}A), (c, \text{OR}B), (\text{OR}O, o)\}$$

$$U = \{G_{AND}, G_{OR}\}$$

$$G_{AND} = \{I_{AND}, O_{AND}, T_{AND}\}$$

$$I_{AND} = \{\text{AND}A, \text{AND}B\}$$

$$O_{AND} = \{\text{AND}O\}$$

$$T_{AND}$$

	AND A	AND B	AND O
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

G_{OR} defined in a similar way

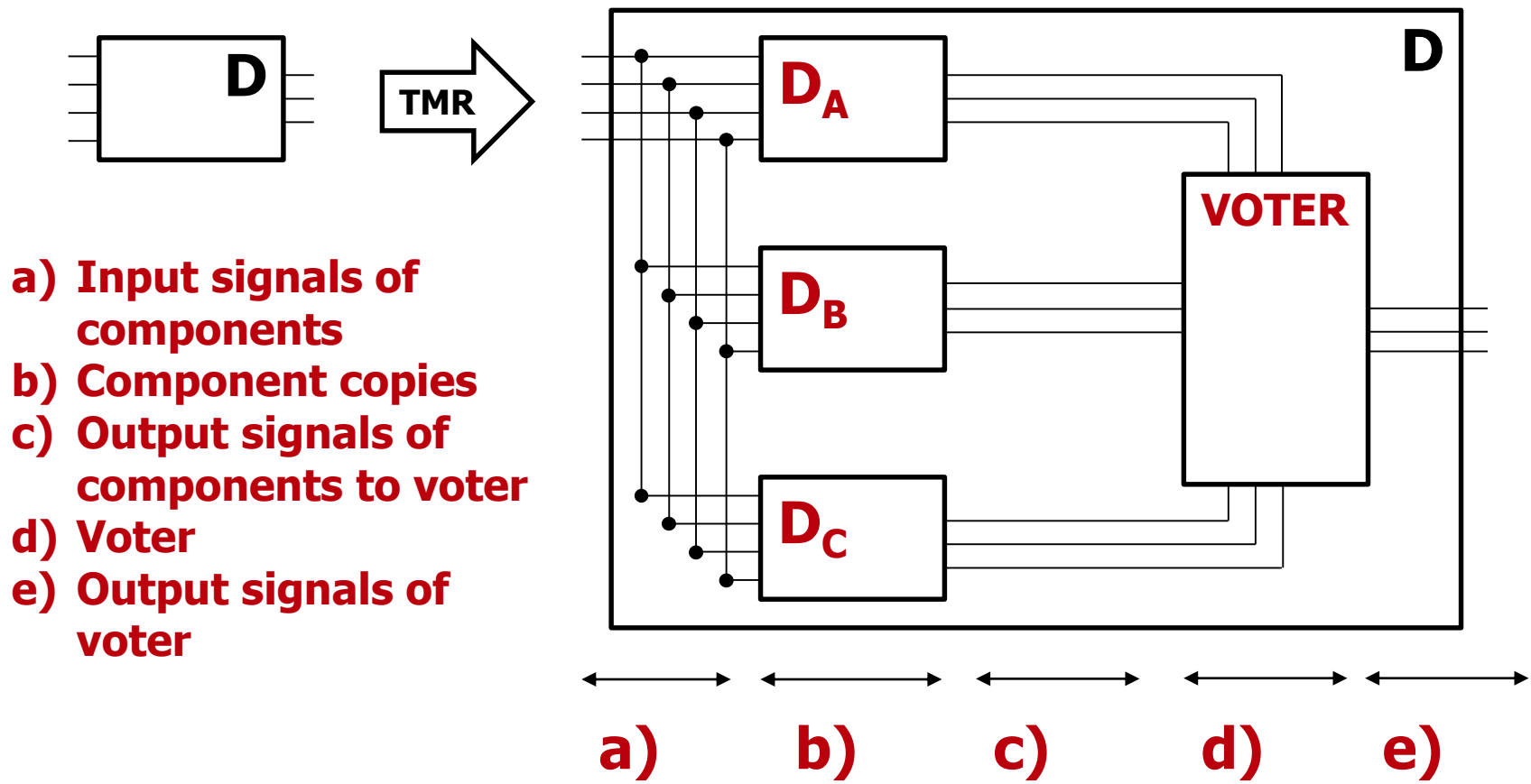
- For better clarity we shall denote the application of algorithm 2.1 on a basic logic gate G and string x as $\text{rename_gate}(G, x)$.

- **Algorithm 2.1:**
 - Input: logic gate $G = (I, O, T)$, arbitrarily chosen string x
 - Output: logic gate $G_x = (I_x, O_x, T_x)$, with all symbols relabeled according to string x
 - The steps of the algorithm:
 - 1) $I_x = \{i_x \mid i \in I\}$
 - 2) $O_x = \{o_x \mid o \in O\}$
 - 3) Developing T_x containing logic table T with all variable names renamed from V to V_x for each variable name.

- For better clarity we shall denote the application of algorithm 2.2 on digital system D and string x as $\text{rename}(D, x)$.
- **Algorithm 2.2:**
 - Input: digital system $D = (I, O, S, U)$, arbitrarily chosen string x
 - Output: digital system $D_x = (I_x, O_x, S_x, U_x)$, with all symbols relabeled according to string x
 - The steps of the algorithm:
 - 1) $I_x = \{i_x \mid i \in I\}$
 - 2) $O_x = \{o_x \mid o \in O\}$
 - 3) $S_x = \{(p_x, q_x) \mid (p, q) \in S\}$
 - 4) $U_x = \{V_x \mid V \in U, \text{ if } V \text{ is a digital system, then } V_x = \text{rename}(V, x), \text{ else } V_x = \text{rename_gate}(V, x)\}$

- Both rename algorithms can be used for
 - making an instance of digital system or basic logic gate from a *template* object,
 - copying an instance of an object.

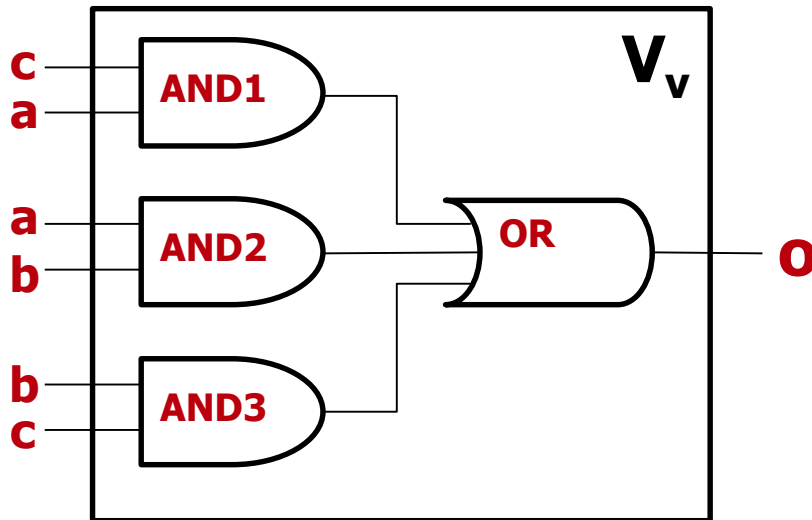
- The basic principles of TMR:



- Let

- $I_V = \{a, b, c\}$
- $O_V = \{o\}$
- $S_V = \{(a, \text{AND1B}), (a, \text{AND2A}), (b, \text{AND2B}), (b, \text{AND3A}), (c, \text{AND3B}), (c, \text{AND1A}), (\text{AND1O}, \text{ORA}), (\text{AND2O}, \text{ORB}), (\text{AND3O}, \text{ORC}), (\text{ORO}, o)\}$
- $U_V = \{D\text{AND1}, D\text{AND2}, D\text{AND3}, D\text{OR}\}$

then an ordered quadruple $V_V = (I_V, O_V, S_V, U_V)$ represents a circuit performing a Boolean *majority function* (median operator).



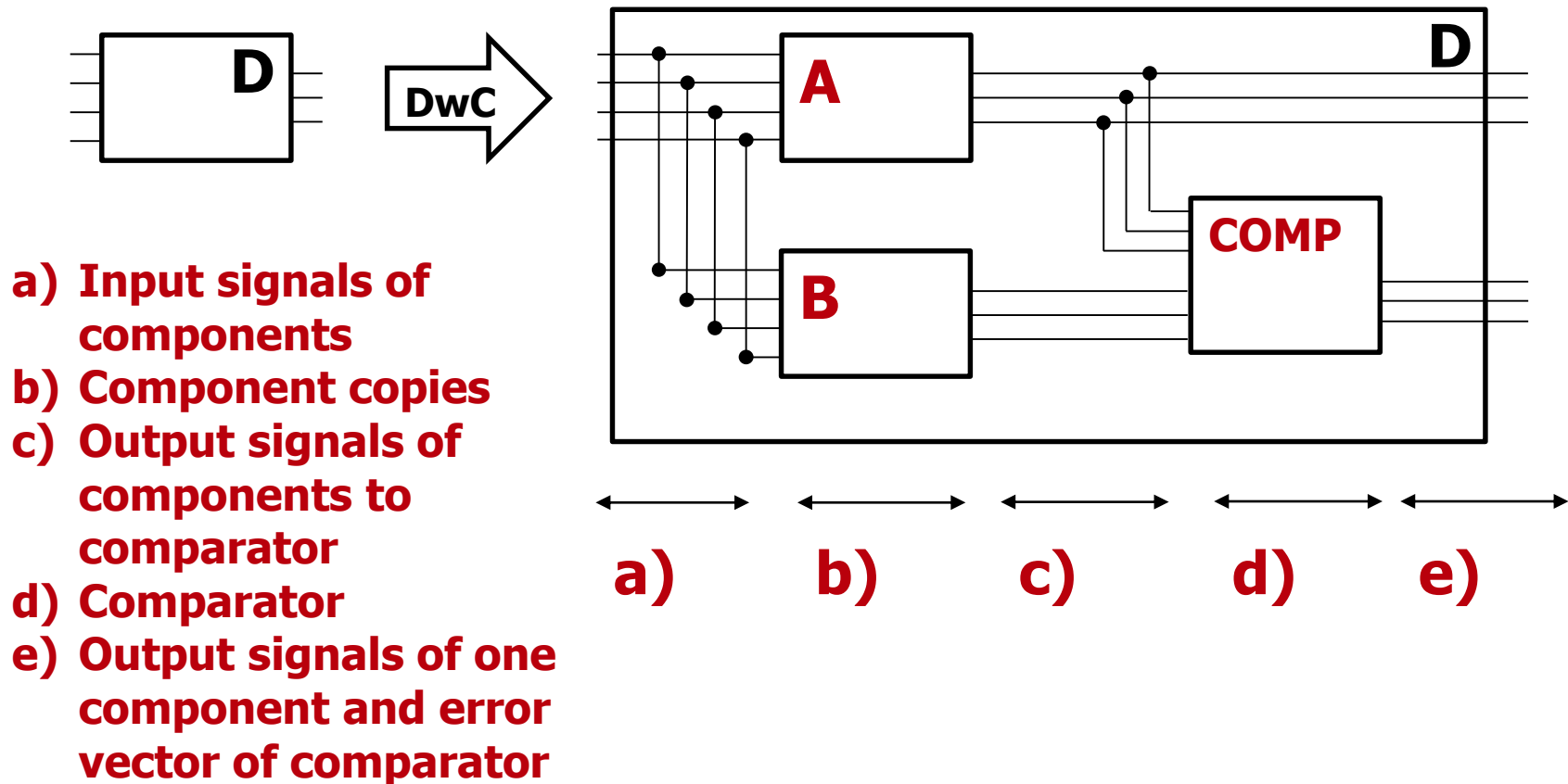
- **Algorithm 2.3:**

- Input: number $n \in \mathbb{N}$ of output signals of the original component,
- Output: component of voter $D_{VOT} = (I_{VOT}, O_{VOT}, S_{VOT}, U_{VOT})$,
- The steps of the algorithm:
 - 1) $i = 1; I_{VOT}, O_{VOT}, S_{VOT}, U_{VOT} = \emptyset$
 - 2) $I_{VOT} = I_{VOT} \cup \{v_{A_i}, v_{B_i}, v_{C_i}\}$
 - 3) $O_{VOT} = O_{VOT} \cup \{w_i\}$
 - 4) $U_{VOT} = U_{VOT} \cup \text{rename}(V_{i'})$
 - 5) $S_{VOT} = S_{VOT} \cup \{(v_{A_i}, a_i), (v_{B_i}, b_i), (v_{C_i}, c_i), (o, w_i)\}$
 - 6) If $(i \neq n)$ then $i = i + 1$; go to 2
- For better clarity we shall denote the application of algorithm 2.3 on number n of output signals as **voter(n)**.

• Algorithm 2.4:

- Input: not fault-tolerant digital system $D = (I, O, S, U)$
- Output: fault-tolerant digital system $D' = (I, O, S', U)$, protected by TMR
- The steps of the algorithm:
 - 1) $D_A = \text{rename}(D, A); D_B = \text{rename}(D, B); D_C = \text{rename}(D, C)$
 - 2) $V_D = \text{voter}(|O|)$
 - 3) $U' = \{D_A, D_B, D_C, V_D\}$
 - 4) $S_{IN} = \bigcup_{i \in I} \{(i, i_A), (i, i_B), (i, i_C)\}$
 - 5) Lets arbitrarily choose bijection $f: O \rightarrow \{1, \dots, |O|\}$
 - 6) $S_{VOT} = \bigcup_{o \in O} \{(o_A, v_{Af(o)}), (o_B, v_{Bf(o)}), (o_C, v_{Cf(o)})\}$
 - 7) $S_{OUT} = \bigcup_{o \in O} \{(w_{f(o)}, o)\}$
 - 8) $S' = S_{IN} \cup S_{VOT} \cup S_{OUT}$

- The basic principles of DwC:



- **Algorithm 2.5:**

- Input: number n of outputs, $n \in \mathbb{N}$
- Output: component of comparator
 $D_{CMP} = (I_{CMP}, O_{CMP}, S_{CMP}, U_{CMP})$.
- The steps of the algorithm:
 - 1) $i = 1; I_{CMP}, O_{CMP}, S_{CMP}, U_{CMP} = \emptyset$
 - 2) $I_{CMP} = I_{CMP} \cup \{v_{A_i}, v_{B_i}\}$
 - 3) $O_{CMP} = O_{CMP} \cup \{e_i\}$
 - 4) $U_{CMP} = U_{CMP} \cup \text{rename_gate}(G_{XOR}, i)$
 - 5) $S_{CMP} = S_{CMP} \cup \{(v_{A_i}, \text{XORA}_i), (v_{B_i}, \text{XORB}_i), (\text{XORO}_i, w_i)\}$
 - 6) If $(i \neq n)$ then $i = i + 1$; go to 2
- Again, we shall denote the application of algorithm 2.5 on number n of output signals as **comparator(n)**.

- **Algorithm 2.6:**

- Input: not fault-tolerant digital system $D = (I, O, S, U)$
- Output: fault-tolerant digital system $D' = (I, O', S', U)$, protected by DwC
- The steps of the algorithm:
 - 1) $U' = \{\text{rename}(D, A), \text{rename}(D, B), \text{comparator}(|O|)\}$
 - 2) $E_{OUT} = \bigcup_{o \in O} \{e_o\}$
 - 3) $O' = E_{OUT} \cup O$
 - 4) $S_{IN} = \bigcup_{i \in I} \{(i, i_A), (i, i_B)\}$
 - 5) Lets arbitrarily choose bijection $f: O \rightarrow \{1, \dots, |O|\}$
 - 6) $S_{CO} = \bigcup_{o \in O} \{(o_A, v_{Af(o)}), (o_B, v_{Bf(o)}), (o_A, o)\}$
 - 7) $S_E = \bigcup_{o \in O} \{(w_{f(o)}, e_o)\}$
 - 8) $S' = S_{IN} \cup S_{CO} \cup S_E$

- Formal model of a digital system
- Two transformation algorithms working with proposed formal model
 - TMR (Triple Modular Redundancy)
 - DwC (Duplication with Comparison)

- Martin Straka: *Methodology of highly reliable systems design*, PhD thesis, Brno, FIT BUT, 2013
- Hlavička, J., Racek, S., Golan, P., Blažek, T.: *Číslicové systémy odolné proti poruchám*, ČVUT, Prague, 1992, ISBN 80-01-00852-5

Thank You For Your Attention !