Formal Methods
for
Interactive Systems

Part 10 — Reverse Engineering with Matrix Algebra

Antonio Cerone
United Nations University
International Institute for Software Technology
Macau SAR China
email: antonio@iist.unu.edu
web: www.iist.unu.edu
Reverse Engineering

Define a model of an existing device
Reverse Engineering

Define a model of an existing device

Danger: model accuracy relies on accuracy of the reverse engineering
Reverse Engineering

Define a model of an existing device

Danger: model accuracy relies on accuracy of the reverse engineering

⇒ manufacturers do not provide formal specifications
Reverse Engineering

Define a model of an existing device

Danger: model accuracy relies on accuracy of the reverse engineering

⇒ manufacturers do not provide formal specifications

⇒ formal specifications must be reconstructed
Reverse Engineering Process

Reconstruct formal specification from

- checking interaction results against the real device
Reverse Engineering Process

Reconstruct formal specification from

- checking interaction results against the real device
- observing user-machine interaction in the real operating environment
Reverse Engineering Process

Reconstruct formal specification from:

- checking interaction results against the real device
- observing user-machine interaction in the real operating environment
- interviewing real users and manufacturer
Reverse Engineering Process

Reconstruct formal specification from

- checking interaction results against the real device
- observing user-machine interaction in the real operating environment
- interviewing real users and manufacturer
- checking documentation (e.g., user manual)
RE Advantages

- model of a real system $\implies$ no need to specify it in a paper)
RE Advantages

- model of a real system $\Rightarrow$ no need to specify it in a paper)
- results can be verified against the device by the reader
**RE Advantages**

- **model of a real system** $\implies$ no need to specify it in a paper)
- **results can be verified against the device** by the reader
- **case studies are more credible** $\iff$ not tailored for the used modelling / verification approach
Matrix Algebra

to describe interaction human-machine
Matrix Algebra

to describe interaction human-machine

Matrices

- are standard mathematical objects
- with a long history
Matrix Algebra

to describe interaction human-machine

Matrices

• are standard mathematical objects with a long history
  no need to introduce yet another notation

• are easy to calculate
Matrix Algebra

to describe interaction human-machine

Matrices

• are standard mathematical objects with a long history
  ➞ no need to introduce yet another notation

• are easy to calculate

• semantics reflects human way of thinking
Matrix Algebra

to describe interaction human-machine

Matrices

• are standard mathematical objects with a long history
  \(\Rightarrow\) no need to introduce yet another notation

• are easy to calculate

• semantics reflects human way of thinking

• have
Matrix Algebra

to describe interaction human-machine

Matrices

- are standard mathematical objects with a long history
  → no need to introduce yet another notation
- are easy to calculate
- semantics reflects human way of thinking
- have
  - structure → easy modelling
Matrix Algebra

to describe interaction human-machine

Matrices

• are standard mathematical objects with a long history
  no need to introduce yet another notation

• are easy to calculate

• semantics reflects human way of thinking

• have
  • structure easy modelling
  • properties verification tool
Matrix Algebra

to describe interaction human-machine

Matrices

• are standard mathematical objects with a long history
  ⇒ no need to introduce yet another notation

• are easy to calculate

• semantics reflects human way of thinking

• have
  • structure ⇒ easy modelling
  • properties ⇒ verification tool

[Thimbleby 04]
Example: Pushbutton Switch

A light bulb controlled by a pushbutton switch
Example: Pushbutton Switch

A light bulb controlled by a pushbutton switch

Pressing the switch alternatively turns the light on or off.
Example: Pushbutton Switch

A light bulb controlled by a pushbutton switch

Pressing the switch alternatively turns the light on or off.

• 2 states: ON and OFF
• 1 operation: push
Example: Pushbutton Switch

A light bulb controlled by a pushbutton switch

Pressing the switch alternatively turns the light on or off.

- 2 states: ON and OFF
- 1 operation: push

[Thimbleby 04]
Model of Pushbutton Switch

- OFF
- ON

push

Push

Remarks on Matrix Algebra
Model of Pushbutton Switch

OFF = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad ON = \begin{bmatrix} 1 & 0 \end{bmatrix}
Model of Pushbutton Switch

\[ \text{OFF} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{ON} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \]

\[ \text{push} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \]
**Pushbutton Switch Transition**

\[
\text{OFF} = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad \text{ON} = \begin{bmatrix} 1 & 0 \end{bmatrix}
\]

\[
\text{push} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]
Pushbutton Switch Transition

OFF

\[
\text{OFF} = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad \text{ON} = \begin{bmatrix} 1 & 0 \end{bmatrix}
\]

\[
\text{push} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]
Pushbutton Switch Transition

OFF = \begin{bmatrix} 0, 1 \end{bmatrix}  

ON = \begin{bmatrix} 1, 0 \end{bmatrix}  

\text{push} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
Pushbutton Switch Transition

\[
\begin{align*}
\text{OFF} \quad \text{push} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \\
\text{OFF} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \quad \text{ON} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \\
\text{push} &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\end{align*}
\]
**Pushbutton Switch Transition**

\[
\text{OFF} \quad \text{push} = \begin{bmatrix} 0, 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1, \end{bmatrix}
\]

\[
\text{OFF} = \begin{bmatrix} 0, 1 \end{bmatrix} \quad \text{ON} = \begin{bmatrix} 1, 0 \end{bmatrix}
\]

\[
\text{push} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]
**Pushbutton Switch Transition**

\[
\text{OFF } \begin{bmatrix} \text{push} \\ \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}
\]

\[
\text{OFF} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{ON} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}
\]

\[
\begin{bmatrix} \text{push} \\ \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]
Pushbutton Switch Transition

\[
\text{OFF} \begin{bmatrix} \text{push} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} = \text{ON}
\]

\[
\text{OFF} = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad \text{ON} = \begin{bmatrix} 1 & 0 \end{bmatrix}
\]

\[
\begin{bmatrix} \text{push} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]
Switch Properties

OFF  push  =  ON
Switch Properties

OFF push = ON

ON push = OFF
Switch Properties

OFF push = ON

ON push = OFF

push push
Switch Properties

OFF \[\text{push}\] = ON

ON \[\text{push}\] = OFF

\[\begin{bmatrix} \text{push} & \text{push} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\]
Switch Properties

OFF \[\text{push}\] = ON

ON \[\text{push}\] = OFF

\[\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\]
Switch Properties

OFF push = ON

ON push = OFF

push push = \[
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
= \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix} = I
Switch Safety

$$\text{OFF} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} = \text{OFF}$$
Switch Safety

\[
\text{OFF} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} = \text{OFF}
\]

\[
\text{ON} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 1, 0 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} = \text{OFF}
\]
Switch Safety

OFF \[
x = \begin{bmatrix} 0, 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} = \text{OFF}
\]

ON \[
x = \begin{bmatrix} 1, 0 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} = \text{OFF}
\]

\[x_{2,1} = 0 \quad x_{2,2} = 1\]
Switch Safety

OFF

\[
\begin{pmatrix}
0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
x_{1,1} & x_{1,2} \\
x_{2,1} & x_{2,2}
\end{pmatrix}
\begin{pmatrix}
0 \\
1
\end{pmatrix}
= \begin{pmatrix}
0 \\
1
\end{pmatrix}
= \text{OFF}
\]

ON

\[
\begin{pmatrix}
1 & 0 \\
\end{pmatrix}
\begin{pmatrix}
x_{1,1} & x_{1,2} \\
x_{2,1} & x_{2,2}
\end{pmatrix}
\begin{pmatrix}
0 \\
1
\end{pmatrix}
= \begin{pmatrix}
0 \\
1
\end{pmatrix}
= \text{OFF}
\]

\[
x_{1,1} = 0 
\quad x_{1,2} = 1 \\
x_{2,1} = 0 
\quad x_{2,2} = 1
\]
Switch Safety

\[
\begin{align*}
\text{OFF } \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF} \\
\text{ON } \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF}
\end{align*}
\]

\[
\begin{align*}
x_{1,1} &= 0 & x_{1,2} &= 1 \\
x_{2,1} &= 0 & x_{2,2} &= 1
\end{align*}
\]
Switch Safety

$$\begin{align*}
\text{OFF} \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF} \\
\text{ON} \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF}
\end{align*}$$

$$x_{1,1} = 0 \quad x_{1,2} = 1 \\
x_{2,1} = 0 \quad x_{2,2} = 1 \\
\Rightarrow \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$\exists \ n \text{ such that } \begin{bmatrix} \text{push} \end{bmatrix}^n = \begin{bmatrix} x \end{bmatrix} ?$$
Switch Safety

\[
\text{OFF} \quad \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF}
\]

\[
\text{ON} \quad \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF}
\]

\[
x_{1,1} = 0 \quad x_{1,2} = 1
\]
\[
x_{2,1} = 0 \quad x_{2,2} = 1
\]

\[
\Rightarrow \quad \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}
\]

\[
\exists \ n \text{ such that} \quad \begin{bmatrix} \text{push} \end{bmatrix}^n = \begin{bmatrix} x \end{bmatrix} \text{?}
\]

\[
\text{No because} \quad \begin{bmatrix} \text{push} \end{bmatrix}^2 = I
\]
Safer System

\[
\begin{align*}
\text{OFF} \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 0, 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 1, 0 \end{bmatrix} = \text{ON} \\
\text{ON} \begin{bmatrix} y \end{bmatrix} &= \begin{bmatrix} 1, 0 \end{bmatrix} \begin{bmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{bmatrix} = \begin{bmatrix} 0, 1 \end{bmatrix} = \text{OFF}
\end{align*}
\]
Safer System

OFF \[ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \text{ON} \]

ON \[ \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \text{OFF} \]

\[ x_{2,1} = 1 \quad x_{2,2} = 0 \quad \Rightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} - \\ - \\ 1 \\ 0 \end{bmatrix} \]
Safer System

\[
\begin{align*}
\text{OFF} \quad \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} = \text{ON} \\
\text{ON} \quad \begin{bmatrix} y \end{bmatrix} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF}
\end{align*}
\]

\[x_{2,1} = 1 \quad x_{2,2} = 0 \quad \Rightarrow \quad \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} - & - \\ 1 & 0 \end{bmatrix}\]

\[y_{1,1} = 0 \quad y_{1,2} = 1 \quad \Rightarrow \quad \begin{bmatrix} y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ - & - \end{bmatrix}\]
Safer System

\[
\begin{align*}
\text{OFF} \quad \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} = \text{ON} \\
\text{ON} \quad \begin{bmatrix} y \end{bmatrix} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} = \text{OFF}
\end{align*}
\]

\[
\begin{align*}
x_{2,1} &= 1 \quad x_{2,2} = 0 \\
\Rightarrow \quad \begin{bmatrix} x \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} = \text{on}
\end{align*}
\]

\[
\begin{align*}
y_{1,1} &= 0 \quad y_{1,2} = 1 \\
\Rightarrow \quad \begin{bmatrix} y \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} = \text{off}
\end{align*}
\]
Two-position Switch

A light bulb controlled by a two-position switch
Two-position Switch

A light bulb controlled by a two-position switch

One switch position (on) turns the light on; the other (off) turns the light off.
Two-position Switch

A light bulb controlled by a two-position switch

One switch position (on) turns the light on; the other (off) turns the light off.

Properties

• The system is closed
Two-position Switch

A light bulb controlled by a two-position switch

One switch position (on) turns the light on; the other (off) turns the light off.

Properties

• The system is closed

• any sequence of actions is equivalent to the last actions the user does
Remarks

- matrix algebra can be used to find interesting properties
Remarks

- matrix algebra can be used to find interesting properties through
- direct manipulation of matrices (operations) rather than vectors (states)
Remarks

- matrix algebra can be used to find interesting properties through direct manipulation of matrices (operations) rather than vectors (states)
  ⇐ number of operations much smaller than number of states
Remarks

• **matrix algebra** can be used to find interesting properties through

• **direct manipulation** of matrices (**operations**) rather than vectors (**states**)  
  \[ \iff \text{number of operations much smaller than number of states} \]

• **the methodology is**
  • **scalable**
Remarks

- **matrix algebra** can be used to find interesting properties through
- **direct manipulation of matrices (operations)** rather than **vectors (states)**
  \[ \text{number of operations much smaller than number of states} \]
- **the methodology is**
  - scalable
  - mechanisable [Gow and Thimbleby 04]
Calculator

• Model: CASIO HL-820LC
Calculator

- Model: CASIO HL-820LC
- State
Calculator

- **Model:** CASIO HL-820LC

- **State**
  - $d = \text{display contents}$
Calculator

- **Model:** CASIO HL-820LC

- **State**
  - $d = \text{display contents}$
  - $m = \text{memory contents}$
Calculator

- Model: CASIO HL-820LC
- State
  - \( d = \) display contents
  - \( m = \) memory contents
  - state representation: \( \text{vector } s = [d, m] \)
Calculator

- **Model**: CASIO HL-820LC

- **State**
  - \( d = \) display contents
  - \( m = \) memory contents
  - state representation: \( \text{vector } s = [d, m] \)

- **Operations**: binary \( 2 \times 2 \) matrices
Calculator Functions

- **M+** add display to memory
Calculator Functions

- **M+** add display to memory
- **M−** subtract display from memory
Calculator Functions

- **M+** add display to memory
- **M−** subtract display from memory
- **AC** clear display
Calculator Functions

- **M+**: add display to memory
- **M−**: subtract display from memory
- **AC**: clear display
- **MRC**: recall memory
**Calculator Functions**

- \[ \text{M+} \] add display to memory
- \[ \text{M–} \] subtract display from memory
- \[ \text{AC} \] clear display
- \[ \text{MRC} \] recall memory
- \[ \text{MRC MRC} \] recall and clear memory
What is the precise semantics of MRC?

- pressed once
**MRC**

What is the precise semantics of MRC?

- pressed once
  - if $m = 0$ then no effect
MRC

What is the precise semantics of MRC?

- pressed once
  - if $m = 0$ then no effect
  - if $m \neq 0$ then $d := m$
**MRC**

What is the precise semantics of MRC?

- **pressed once**
  - if $m = 0$ then **no effect**
  - if $m \neq 0$ then $d := m$

- **pressed twice**
**MRC**

What is the precise semantics of MRC?

- **pressed once**
  - if $m = 0$ then no effect
  - if $m \neq 0$ then $d := m$

- **pressed twice**
  - $m := 0$
Remarks on MRC

• \( \text{MRC} \times \text{MRC} \) is not the same as the product \( \text{MRC} \cdot \text{MRC} \)
Remarks on MRC

- \( \text{MRC} \times \text{MRC} \) is not the same as the product \( \text{MRC} \cdot \text{MRC} \).

- Semantics of pressed once depends on previous content of memory.
Remarks on MRC

- $\text{MRC} \times \text{MRC}$ is not the same as the product $\text{MRC} \cdot \text{MRC}$

- Semantics of pressed once depends on previous content of memory

- Is $\text{MRC} (\text{MRC} \text{MRC}) = (\text{MRC} \text{MRC}) \text{MRC}$?
Remarks on MRC

- **MRC** is not the same as the product **MRC** · **MRC**

- Semantics of pressed once depends on previous content of memory

- Is \( \text{MRC} \left( \text{MRC} \text{MRC} \right) = \left( \text{MRC} \text{MRC} \right) \text{MRC} \)?

- Calculator:
  \[
  \text{MRC} \left( \text{MRC} \text{MRC} \right) = \text{MRC} \text{MRC}
  \]
Bad Design?

• Do we really need to give such a special function to MRC MRC?
Bad Design?

- Do we really need to give such a special function to \(\text{MRC MRC}\)?

- No!

\[
\text{MRC MRC} = \text{MRC M}\]

A. Cerone, UNU-IIST – p.18/26
Bad Design?

• Do we really need to give such a special function to $\text{MRC}_1 \text{MRC}_2$?

• No!
  
  $\text{MRC}_1 \text{MRC}_2 = \text{MRC}_2 - \text{MRC}_1$

• $\text{MRC}_1 \text{MRC}_2$ is a bad design choice!
Bad Design?

• Do we really need to give such a special function to MRC MRC?

• No!

MRC MRC = MRC M–

• MRC MRC is a bad design choice!

[Thimbleby 00]
Safety-critical

- Is a calculator safety-critical?
Safety-critical

- Is a calculator safety-critical?
- May a calculator-like interface be safety-critical?
Safety-critical

- Is a calculator safety-critical?
- May a calculator-like interface be safety-critical?
- Yes!

Syringe pump have calculator-like interfaces
Examinations
Seminar 6 — Usable Calculator

Topic: Calculator Interfaces

Towards a truly usable calculator

- Harold Thimbleby
  *Computer Algebra for User Interface Design*, 2004

- Harold Thimbleby
  *A Novel Pen-based Calculator and its Evaluation*, 2004

- Will Thimbleby and Harold Thimbleby
  *A Novel Gesture-based Calculator and its Design Principles*, 2005

- Websites on
  - Computer Algebra: [http://www.cs.swan.ac.uk/~csharold/CA/](http://www.cs.swan.ac.uk/~csharold/CA/)
  - Calculator: [http://www.cs.swan.ac.uk/calculators/](http://www.cs.swan.ac.uk/calculators/)
References
Reverse Engineering

- [Thimbleby 04]: Reverse Engineering and Matrix Algebra
- [Gow and Thimbleby 2004]: Matrix Algebra and Tools
- [Thimbleby 00]: Reverse Engineering
[Thimbleby 04]

Harold Thimbleby.

*User interface design with matrix algebra.*


About:

- **Reverse Engineering**
- **Matrix Algebra**
[Gow and Thimbleby 04]


About:

- **Tools**
Harold Thimbleby.

Calculators are needlessly bad.


About:

- Reverse Engineering of Calculators