A General BSS Model over Arbitrary Structures

Christine Gaßner
Greifswald

Cape Town
February, 2011

A general BSS model over arbitrary structures

1. The classical register machines and the BSS model

2. Comparison of BSS model and RAM's

3. Comparison of Type-2 machines, BSS machines, and classical RAM's

   - Turing machines ⟺ BSS machines

   - Type-2 machines ⟺ RAM's

   - Consequences

4. The Halting problems for several types of machines

5. Remarks to non-deterministic machines

Some known models of computation

- Turing machines over $\{0, 1\}$ (and a blank symbol)
  - Type-1
  - Type-2

  Meaning:
  theory of computability and theory of complexity

- Register machines over $\{a_1, \ldots, a_k\}, \mathbb{N}, \mathbb{R}, \ldots$
  - Finite dimensional machines
  - Infinite dimensional machines
    - Offline
    - Online

  Meaning:
  simple models for existing computers and theory of complexity

gassnerc@uni-greifswald.de

Some known models of computation offline / online

Minsky (1961), Scott (1967), Blum/Shub/Smale (1989),… (offline):

gassnerc@uni-greifswald.de

Some known models of computation offline / online

Minsky (1961), Scott (1967), Blum/Shub/Smale (1989),… (offline):

Cook/Reckhow (1973), Aho/Hopcroft/Ullman (1974),… (offline / online):

Some known models of computation offline / online

Minsky (1961), Scott (1967), Blum/Shub/Smale (1989),… (offline):

$\overbrace{(x_1,\ldots,x_n)}^{x}$ → Input procedure → A program executed until stop criterion → Output procedure → $\overbrace{(y_1,\ldots,y_m)}^{f(x)}$

Cook/Reckhow (1973), Aho/Hopcroft/Ullman (1974),… (offline / online):

$(x_1, x_2, \ldots, x_n[, \ldots])$ suitable inputs or codes of objects → READ → A program executed until stop criterion → PRINT → $(y_1, y_2,\ldots, y_m[,\ldots])$

Weihrauch (1985?),… (online):

$(x_1, x_2, x_3,\ldots)$ code/name of u → READ → A program → PRINT → $(y_1, y_2, y_3,\ldots)$ code/name of $f(u)$

$u \longrightarrow f(u)$

gassnerc@uni-greifswald.de

Our goal

Definition of a general model of computation

- by
  - comparing known standard models
  - generalizing the main concepts

- in order to
  - unify known models
  - gain new insights in the relationship between known models (similarities and differences)
  - transfer results from one theory of computation to another
  - better understand the open problems in the classical theory of complexity (Why is it difficult to solve the classical P-NP problem? …)

gassnerc@uni-greifswald.de

Our register machines

- Registers
  - a countable number of registers for storing numbers
- Instructions
  - start and halt instructions
  - operation and test instructions
- Program
  - a finite sequence of labelled instructions (M. L. Minsky, 1961)
  - represented by flow diagrams (Z. A. Melzak, 1961; J. Lambek, 1961)
  - a program schema with operation and predicate symbols (D. Scott, 1967)
- Machine
  - provides the interpretation for a program (D. Scott, 1967)
- Model of computation
  - the input and the output procedures
  - the machine

Our register machines

- Finite dimensional machines
  - a finite number of registers:
    $Z_1, \ldots, Z_m$
  - Each configuration is determined by the label of the current instruction and the values of the registers.

- Infinite dimensional machines
  - an infinite number of registers:
    $$\underbrace{Z_1, Z_2, \ldots, Z_m}_{m \text{ processor registers (or one accumulator)}}, Z_{m+1}, Z_{m+2}, \ldots$$
  - without indirect addressing
  - with indirect addressing
    (can be realized by index registers $I_1, \ldots, I_k$)

Examples for several structures

$\mathbb{Z}_2 = (\{0, 1\}; 0, 1; +, \cdot ; =)$

⇨ Turing machine

⇨ Type-2 machines

$\mathbb{N}_0 = (\mathbb{N}; 0 ; f_1 , f_2 ; r_1)$, $\quad f_1(n) = n + 1$, $\quad f_2(n) = n \dot{-} 1$, $\quad r_1(n) = true$ iff $n \neq 0$

⇨ counter machine $\quad (0 \dot{-} 1 = 0)$

$\mathbb{N} = (\mathbb{N}; \mathbb{N} ; + , - ; r_1)$

⇨ classical RAM

$\mathbb{R} = (\mathbb{R} ; \mathbb{R} ; + , - , \cdot ; \leq)$

⇨ real RAM

⇨ BSS model

gassnerc@uni-greifswald.de

Instructions over Σ

A structure: $\Sigma = (U; c_1, c_2, ...; f_1, f_2, ...; r_1, r_2, \ldots)$

Computation: l: $Z_k := f_j(Z_{k_1}, \ldots, Z_{k_{m_j}})$;
l: $Z_k := c_j$;

Branching: l: if $r_j(Z_{k_1}, \ldots, Z_{k_{n_j}})$ then goto l_1 else goto l_2;

Copy: l: $Z_{I_k} := Z_{I_j}$;

Index computation: $I_k := 1$; $I_k := I_k + 1$; if $I_k = I_j$ then goto l_1 else goto l_2;

gassnerc@uni-greifswald.de

An example for a finite dimensional machine

By Minsky (1961).

$\Sigma = (\mathbb{N}; 0\,; f_1, f_2\,; r_1),\quad f_1(n) = n+1,\quad f_2(n) = n \dot{-} 1$ $(0 \dot{-} 1 = 0)$, $r_1(n) = \mathit{true}$ iff $n \neq 0$

Registers: Z_1, Z_2

Computation:

l: $Z_1 := Z_1 + 1$; l: $Z_1 := Z_2 + 1$; l: $Z_2 := Z_1 + 1$; l: $Z_2 := Z_2 + 1$;

l: $Z_1 := Z_1 \dot{-} 1$; l: $Z_1 := Z_2 \dot{-} 1$; l: $Z_2 := Z_1 \dot{-} 1$; l: $Z_2 := Z_2 \dot{-} 1$;

Branching:

l: if $Z_1 \neq 0$ then goto l_1 else goto l_2; l: if $Z_2 \neq 0$ then goto l_1 else goto l_2;

Input:

$In(n) = (2^n, 0)$

Output:

$Out(z_1, z_2) = m$, if $(z_1, z_2) = (2^m, 0)$

$Out(z_1, z_2)$ undefined otherwise

An example for an infinite dimensional machine

The Model of L. Blum, M. Shub, S. Smale (1989) is similar to:

Registers: $I_1, I_2, Z_1, Z_2, \ldots$

Computation and Branching: $+, -, \bullet, \ldots, \leq$ Copy: $Z_1 := Z_{I_j}$; $Z_{I_j} := Z_1$;

Input and Output:

$(x_1, \ldots, x_n) \in \cup_{i \geq 1} \mathbb{R}^i$ ⟺ identified ⟺ $(x_1, \ldots, x_n, 0, 0, \ldots) \in \mathbb{R}^\omega,\ x_n \neq 0$

$$In(x_1, \ldots, x_n, 0, 0, \ldots) = (1, 1, x_1, n, x_2, 0, \ldots, x_n, 0, 0, \ldots), \quad x_n \neq 0$$

$$Out(i_1, i_2, z_1, z_2, \ldots) = (z_1, z_3, z_5, \ldots) \in \mathbb{R}^\omega$$

K. Meer (1992): $In(x_1, \ldots, x_n, 0, 0, \ldots) = (1, 1, x_1, 0, x_2, 0, \ldots, x_n, 0, 0, \ldots)$

E. Grädel (2007): $In(x_1, \ldots, x_n, 0, 0, \ldots) = (1, 1, x_1, x_2, \ldots, x_n, 0, 0, \ldots)$

gassnerc@uni-greifswald.de

An further example for an infinite dimensional machine

By C. Gaßner (1996), ...

Registers: $I_1, I_2, \ldots, I_k, Z_1, Z_2, \ldots$

Computation and branching: $+, -, \bullet, \ldots, r_j(Z_{k_1}, \ldots, Z_{k_{n_j}})$ Copy: $Z_{I_k} := Z_{I_j}$;

Input (cp. P. Koiran, 1994) and output space:

$$\boxed{(x_1, \ldots, x_n) \in \cup_{i \geq 1} \mathbb{R}^i}$$

Input:

$$In(x_1, \ldots, x_n) = (n, 1, \ldots, 1, x_1, \ldots, x_n, 0, 0, \ldots)$$

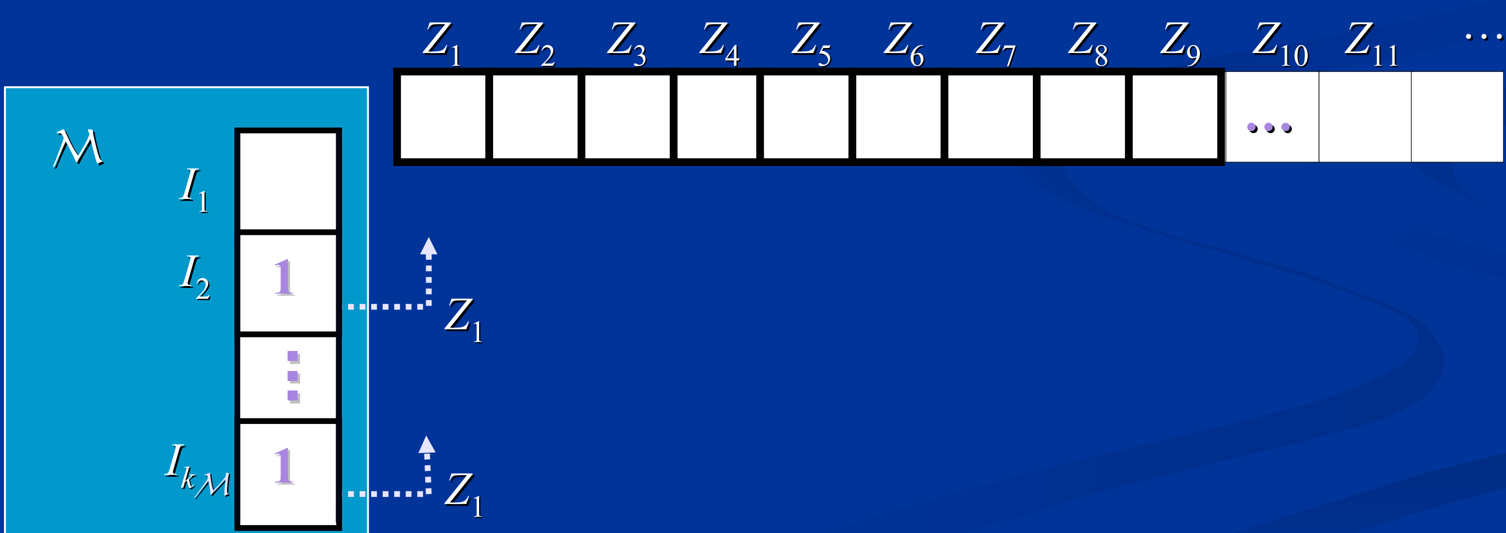
Output:

$$Out(i_1, i_2, \ldots, i_k, z_1, z_2, \ldots) = (z_1, \ldots, z_{i_1}) \in \cup_{i \geq 1} \mathbb{R}^i$$

gassnerc@uni-greifswald.de

The general BSS machine and the input

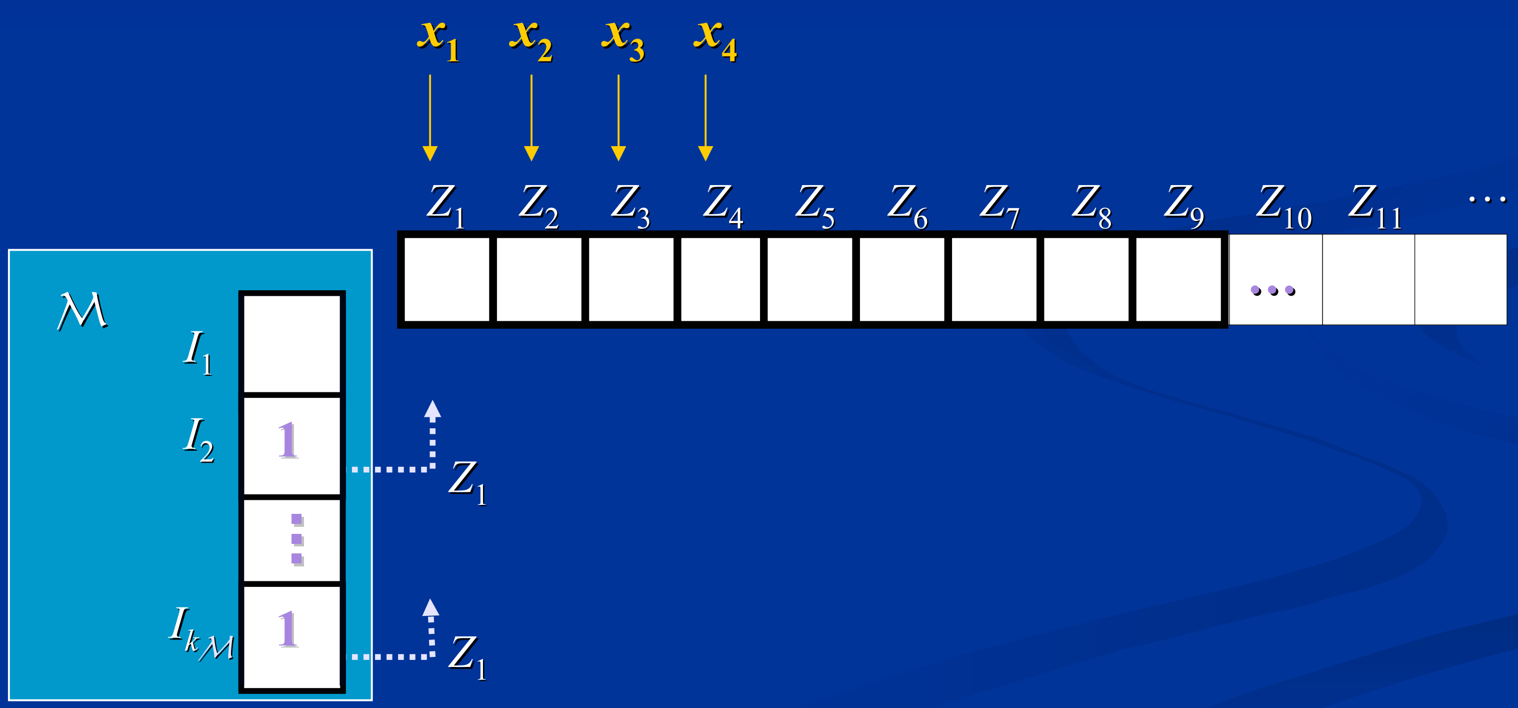
The input: $(Z_1,\ldots,Z_n) := \underbrace{(x_1,\ldots,x_n)}_{\in\, \cup_{i\geq 1}\mathbb{R}^i}$; $I_1 := n$; $I_2 := 1; \ldots ; I_{k_{\mathcal{M}}} := 1$;



gassnerc@uni-greifswald.de

The general BSS machine and the input

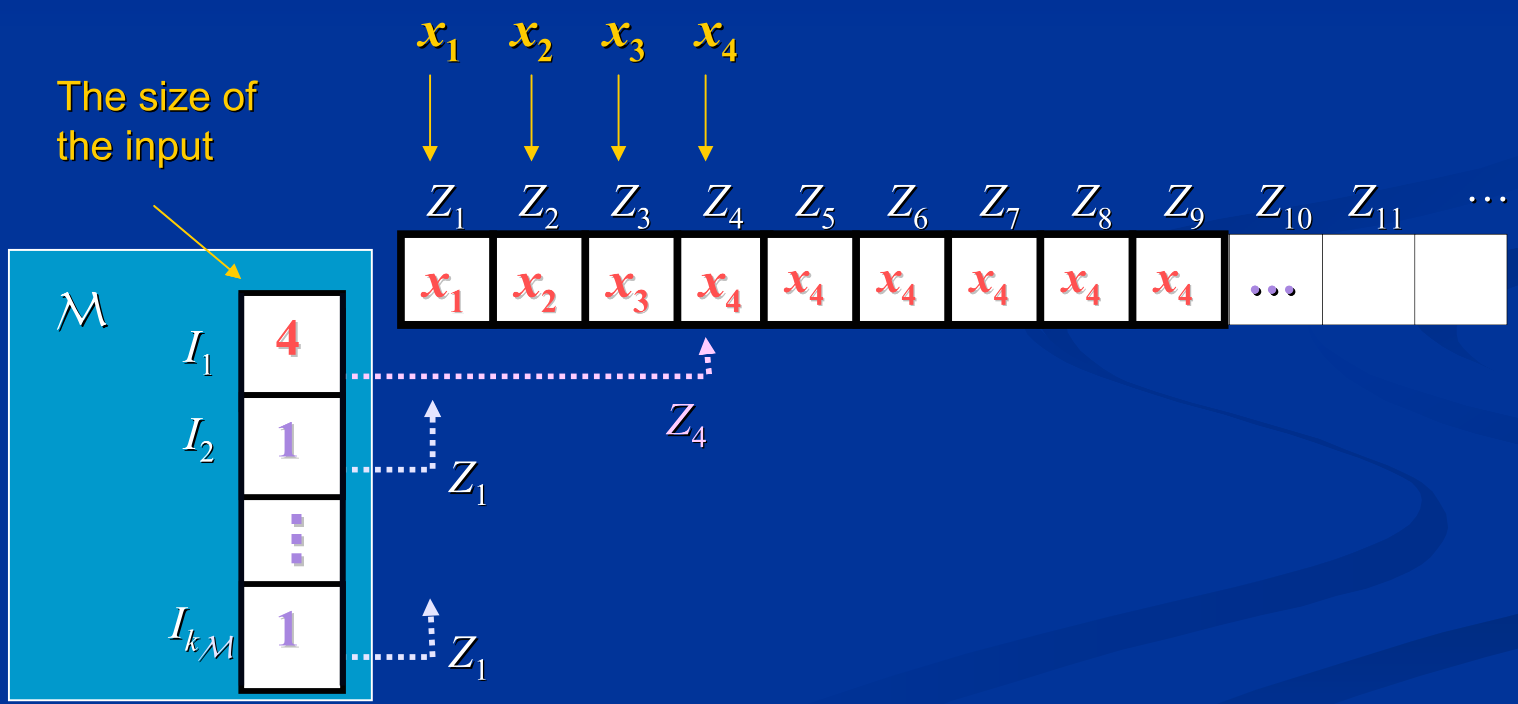
The input: $(Z_1, \ldots, Z_n) := (x_1, \ldots, x_n)$; $I_1 := n$; $I_2 := 1; \ldots ; I_{k_{\mathcal{M}}} := 1$;



gassnerc@uni-greifswald.de

The general BSS machine and the input

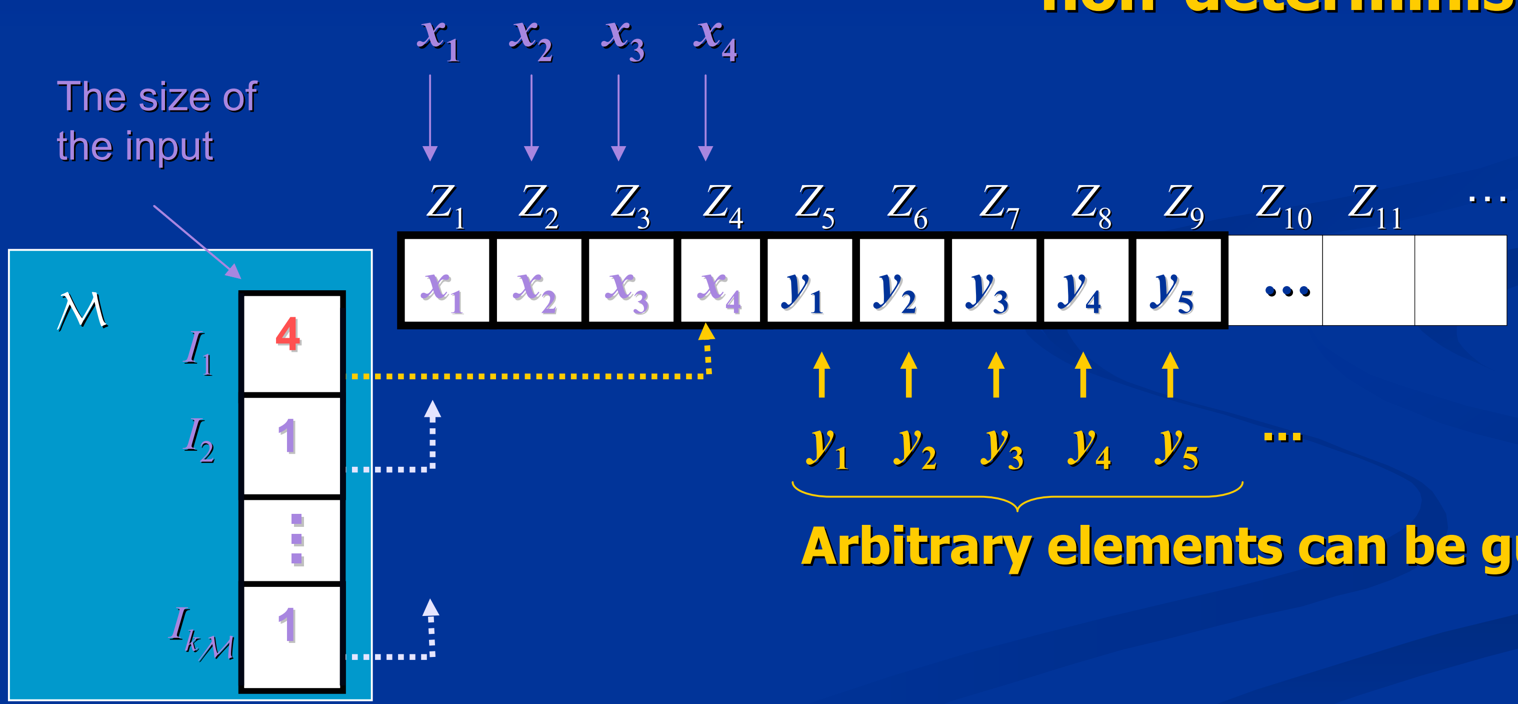
The **input**: $(Z_1, \ldots, Z_n) := (x_1, \ldots, x_n)$; $I_1 := n$; $I_2 := 1; \ldots ; I_{k_\mathcal{M}} := 1$;



The non-deterministic BSS machines (input and guessing)

The guessing: $(Z_{n+1}, \ldots, Z_{n+m}) := (y_1, \ldots, y_m) \in \cup_{i \geq 1} U^i$

non-deterministic !



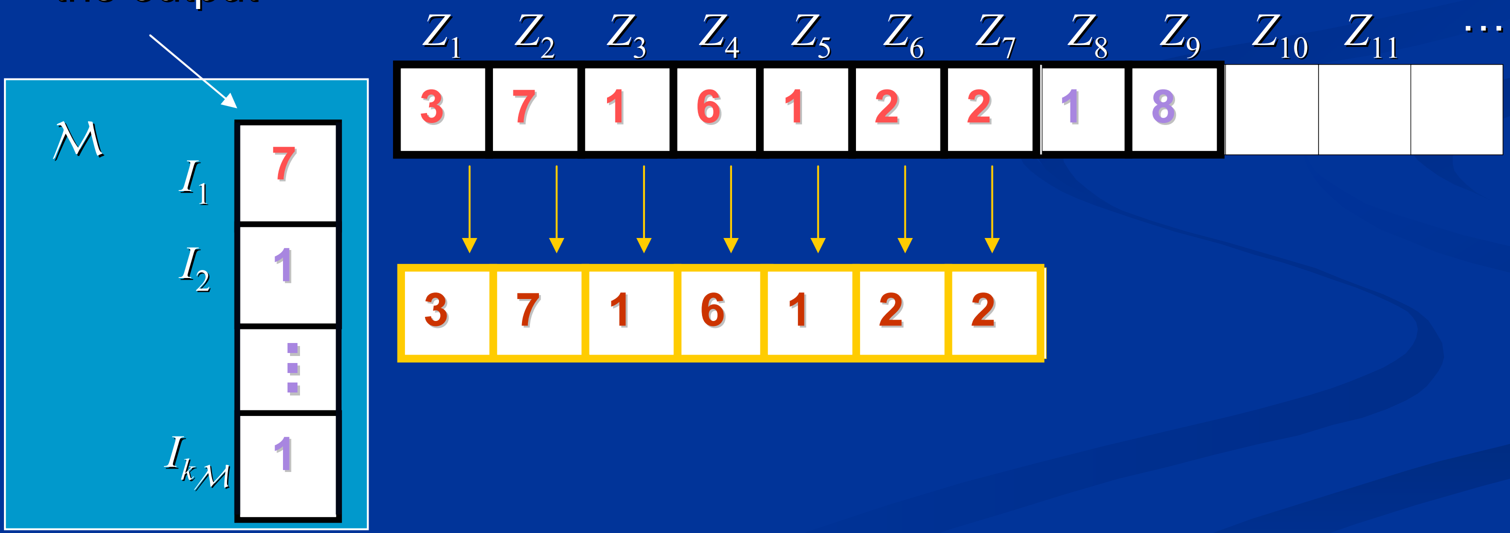
Arbitrary elements can be guessed !

gassnerc@uni-greifswald.de

The general BSS machine and the output

The **output**: $(Z_1, \ldots, Z_{I_1})$

The size of the output

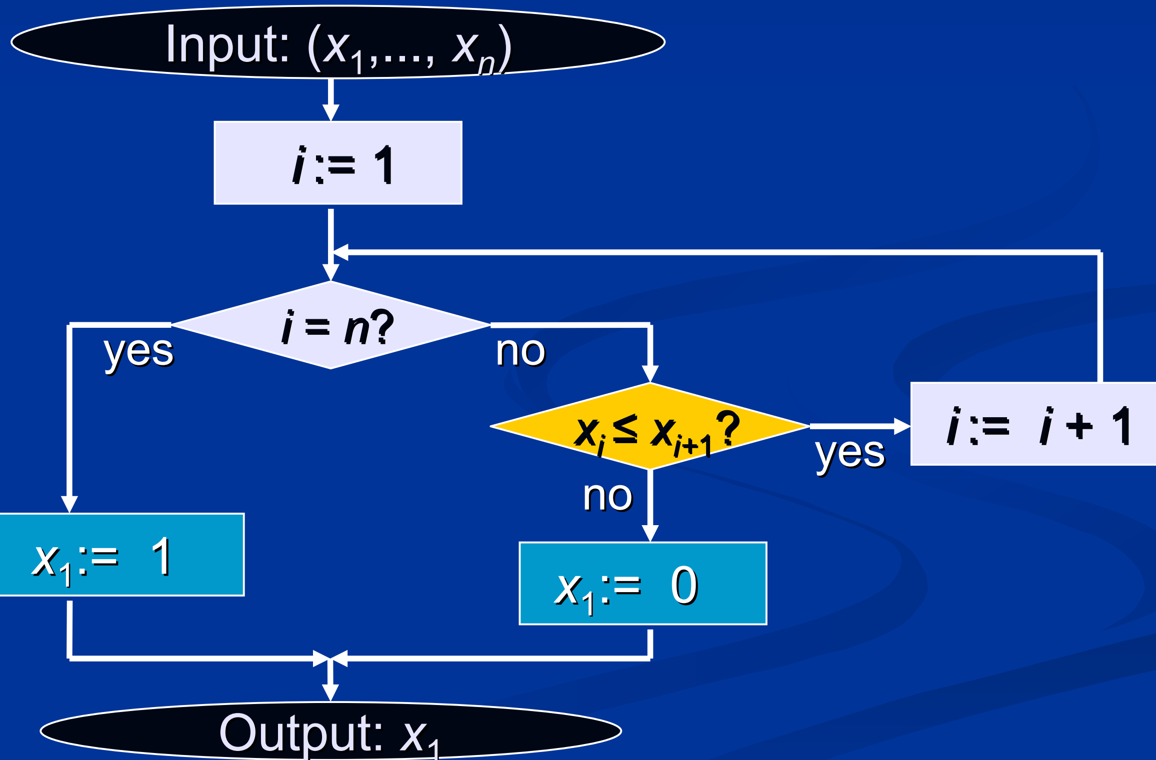


BSS machine for the problem of ordered tuples

Input: a tuple of integers $(x_1,..., x_n) \in \cup_{i \geq 1}\mathbb{N}^i$
Problem: Is the input ordered?
Decision:

$i := 1;$

1: if $i = n$
then $x_1 := 1;$
else
if $x_i \leq x_{i+1}$
then $i := i + 1;$
goto 1;
else $x_1 := 0;$



gassnerc@uni-greifswald.de

BSS machine for the computation of the sum

Input: a tuple of integers $(x_1,..., x_n) \in \cup_{i\geq 1}\mathbb{N}^i$

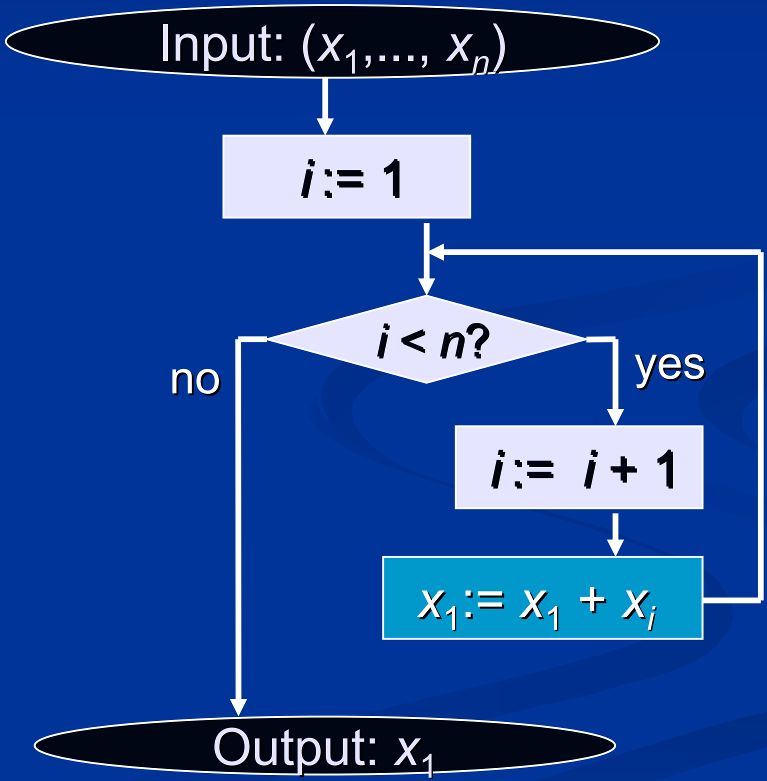
Output: $\sum_{i=1..n} x_i$

Computation:

$i := 1;$

1: if $i < n$
then $i := i + 1;$
$x_1 := x_1 + x_i;$
goto 1;

for $i := 2$ to n do
{
$x_1 := x_1 + x_i;$
}



The simulation of machines by BSS machines

A finite dimensional register machine

↓

A classical RAM ↓ A BSS machine

A Type-2 machine with $I, O \subseteq \{0, 1\}^*$ ↓ A BSS machine

A BSS machine

↓

A system (family) of RAM's for k-dimensional inputs $(k \in \mathbb{N}^+)$

An online RAM with $I, O \subseteq \mathbb{R}^\omega$ ↓ A RAM with one-way write-only output tape

A Type-2 machine with $I, O \subseteq \{0, 1\}^\omega$ ↓ A RAM with one-way write-only output tape

A RAM with one-way write-only output tape

↓ means „can be simulated by"

Our BSS model and classical RAM's with $I \subseteq \cup_{i\geq 1}\mathbb{R}^i$

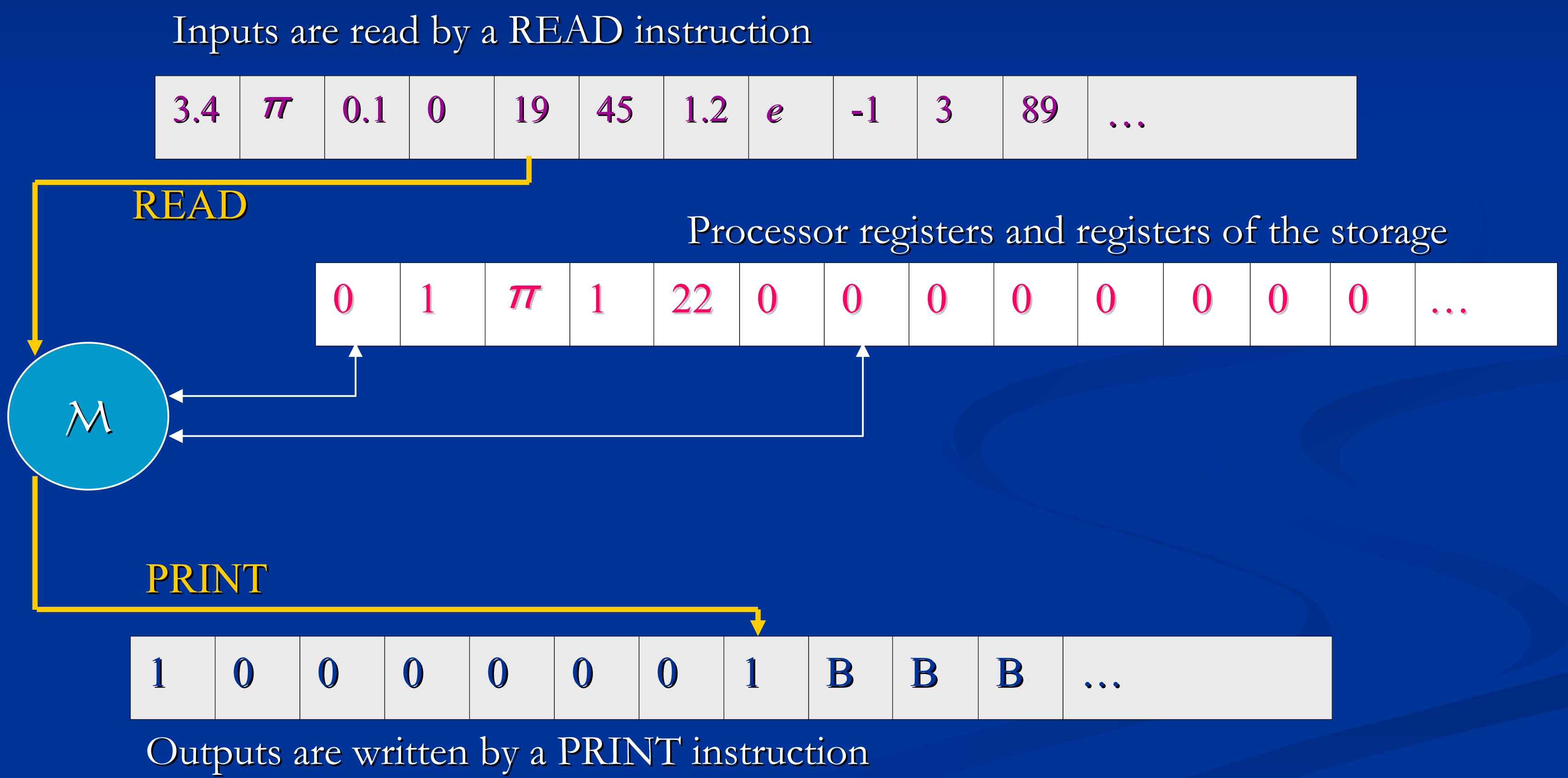
<table>
<tr><th></th><th>BSS machine</th><th colspan="6">Classical RAM's</th></tr>
<tr><td>In-, output mode</td><td colspan="6">offline</td><td>online</td></tr>
<tr><td>In-, output space</td><td>$I, O = \cup_{i\geq 1}\mathbb{R}^i$</td><td colspan="2">$I \subseteq \mathbb{N}^n$
$O \subseteq \mathbb{N}$</td><td colspan="3">$I \subset \cup_{i\geq 1}\mathbb{N}^i$
$I \subset \cup_{i\geq 1}\mathbb{R}^i$</td><td>$I, O \subseteq \mathbb{R}^\omega$</td></tr>
<tr><td>Indirect addressing</td><td>yes</td><td colspan="2">no</td><td colspan="4">yes</td></tr>
<tr><td>Any machine has its own program.</td><td>yes</td><td colspan="6">no / yes</td></tr>
<tr><td>Number of registers</td><td>∞</td><td>$< \infty$</td><td colspan="5">∞</td></tr>
<tr><td>Number of 'tapes'</td><td colspan="4">1</td><td colspan="3">3</td></tr>
<tr><td>Input</td><td colspan="5">input procedure</td><td colspan="2">read instruction</td></tr>
<tr><td>Output</td><td colspan="5">output procedure</td><td colspan="2">print instruction</td></tr>
</table>

Our BSS model and classical RAM's with $I \subseteq \cup_{i\geq 1}\mathbb{R}^i$ (Offline)

The end of the input is not decidable by a classical RAM with READ since there is not a blank symbol.

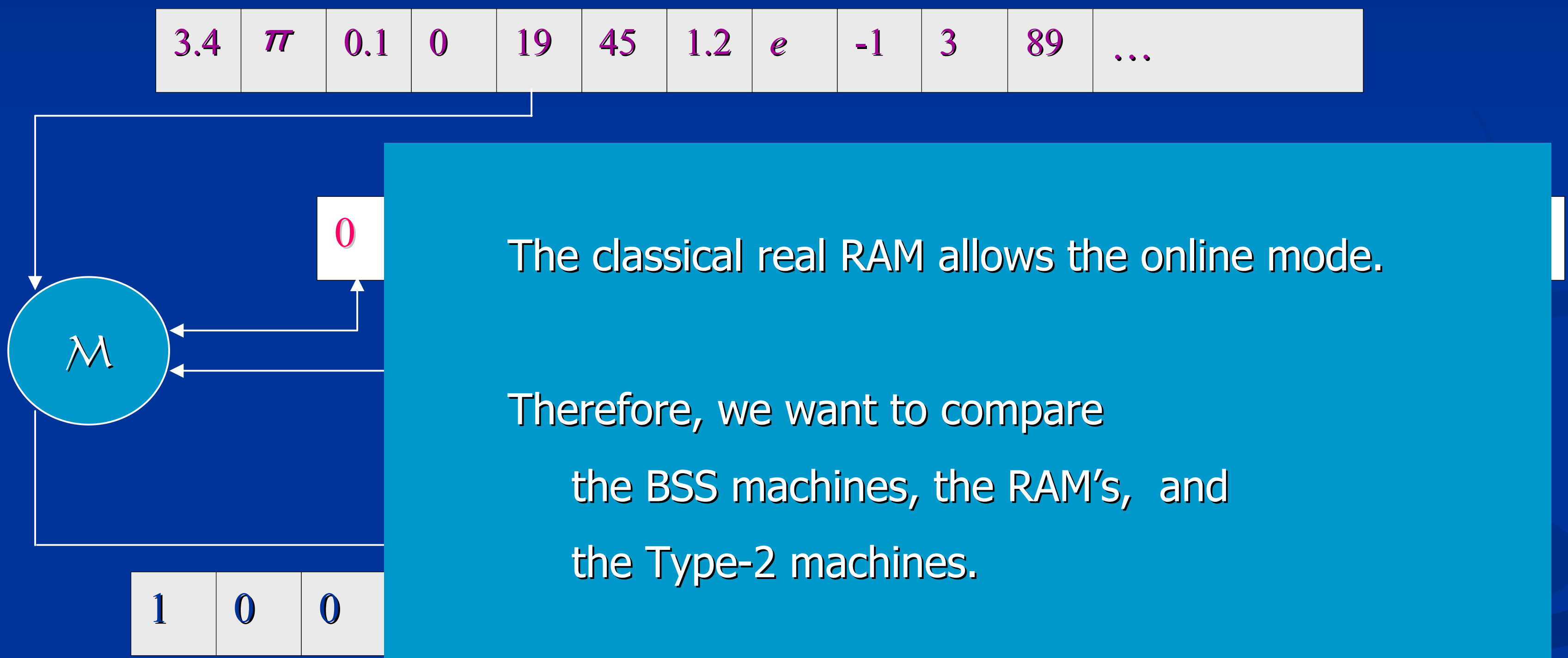
	BSS model	Classical RAM
The size of each input	is stored in an index register.	cannot be computed.
The output of the last value of each input	is possible.	is not possible.
The sum of all the input values	can be computed.	cannot be computed.

The classical real RAM without input and output procedure



The classical real RAM without input and output procedure

Inputs are read by a READ instruction



The classical real RAM allows the online mode.

Therefore, we want to compare

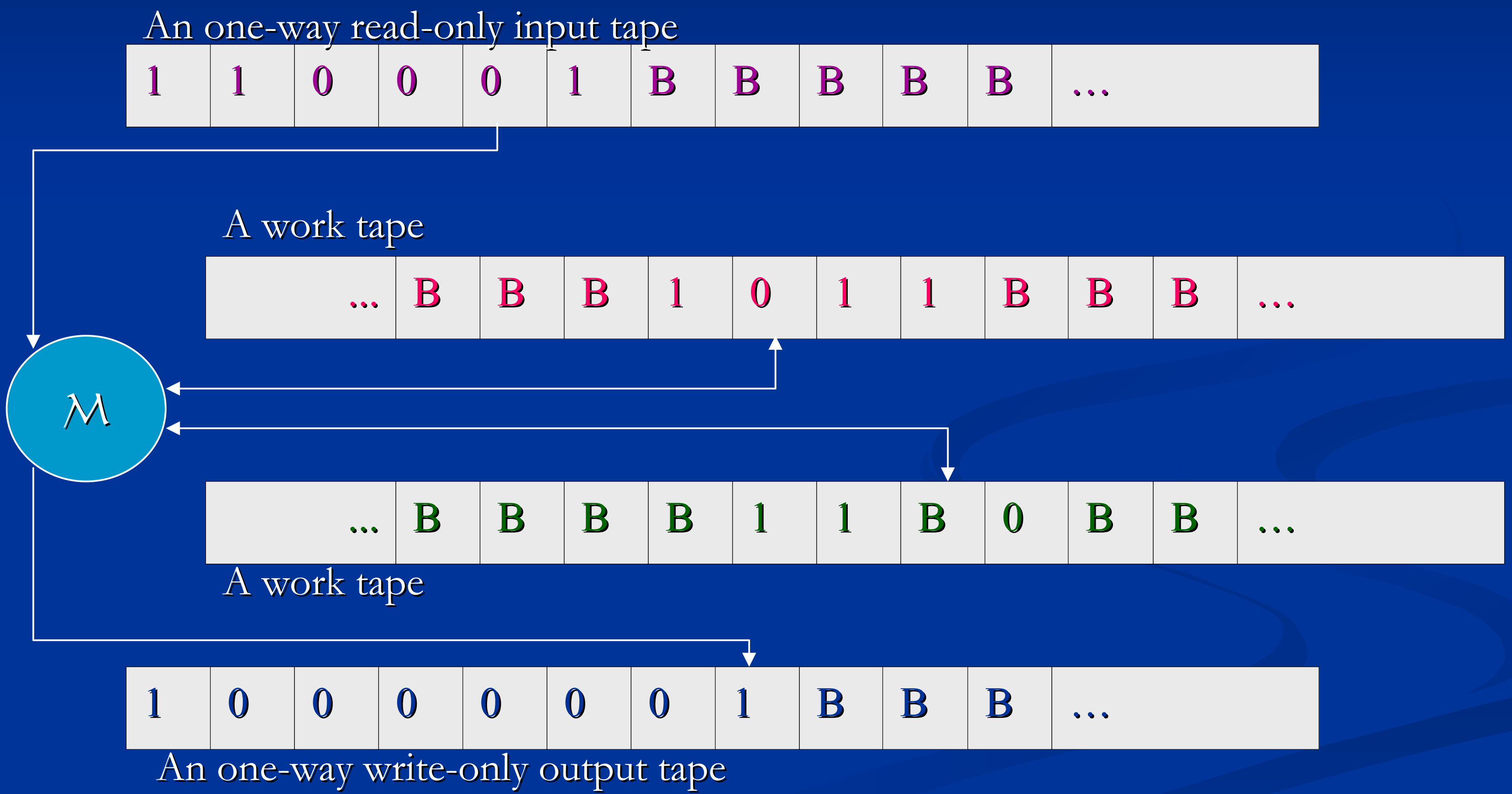
the BSS machines, the RAM's, and

the Type-2 machines.

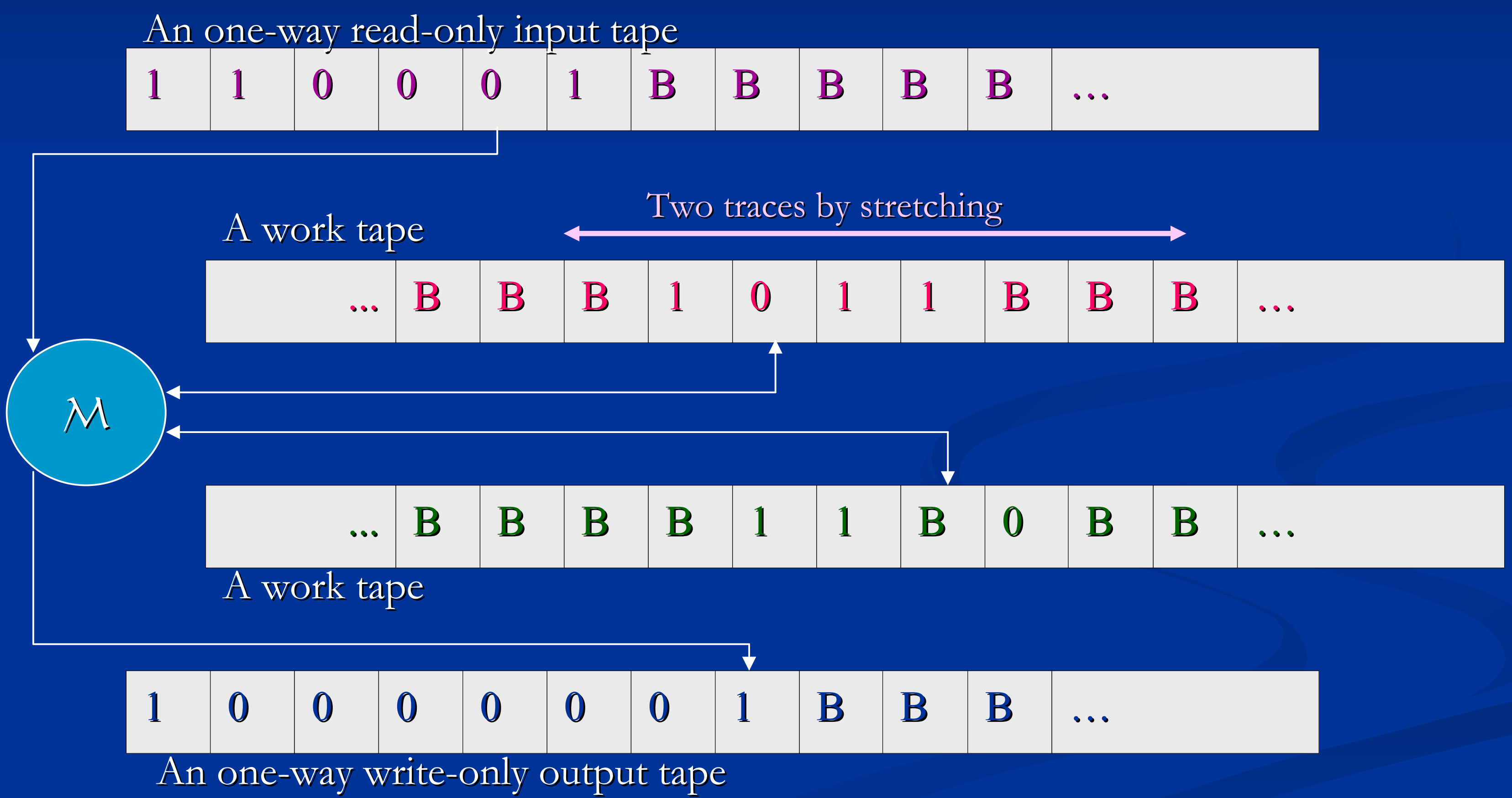
Outputs are written by a PRINT instruction

gassnerc@uni-greifswald.de

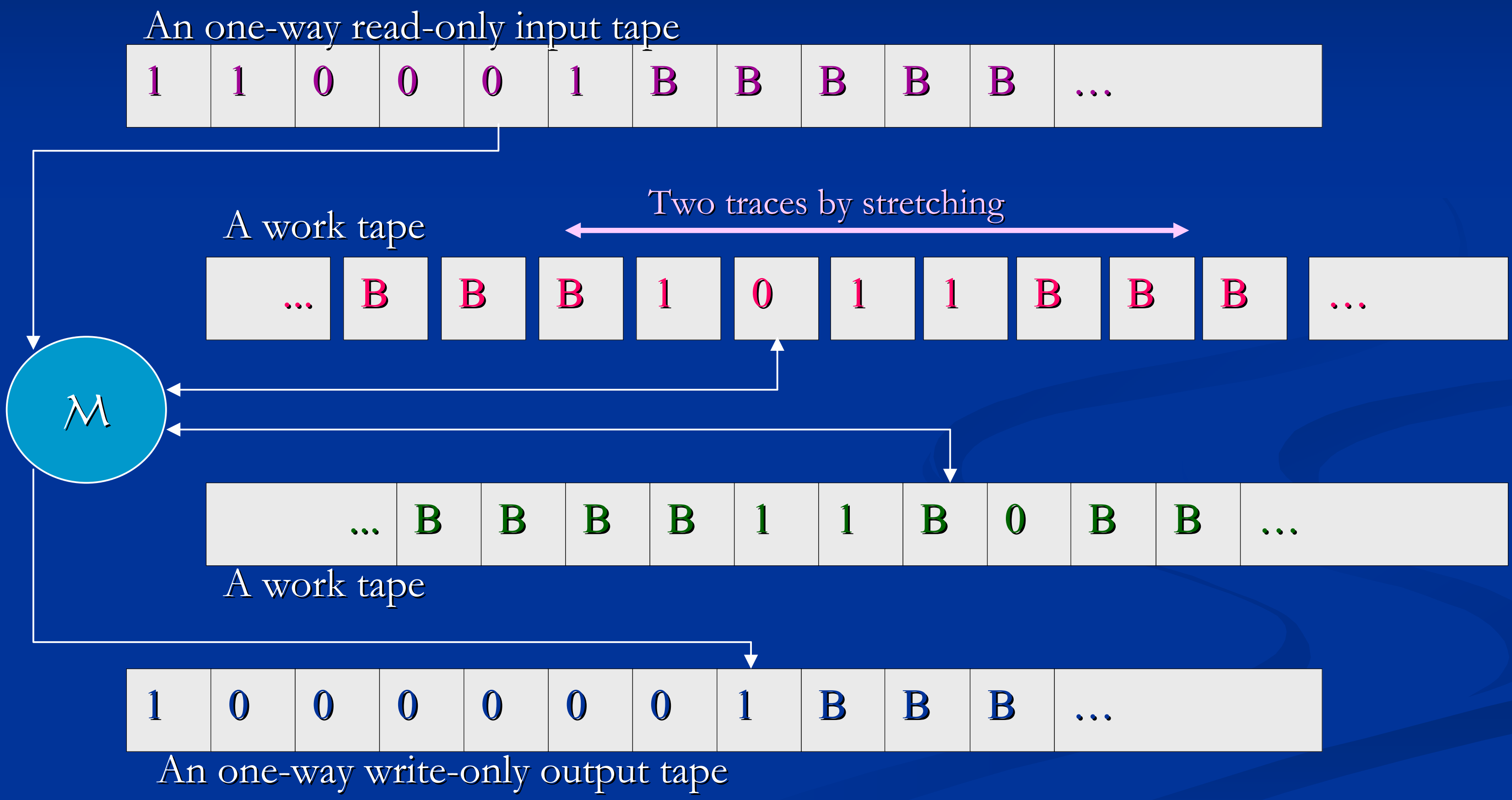
The Type-2 machines with $Y_1, Y_0 = \{0, 1\}^*$ or $Y_1, Y_0 = \{0, 1\}^\omega$



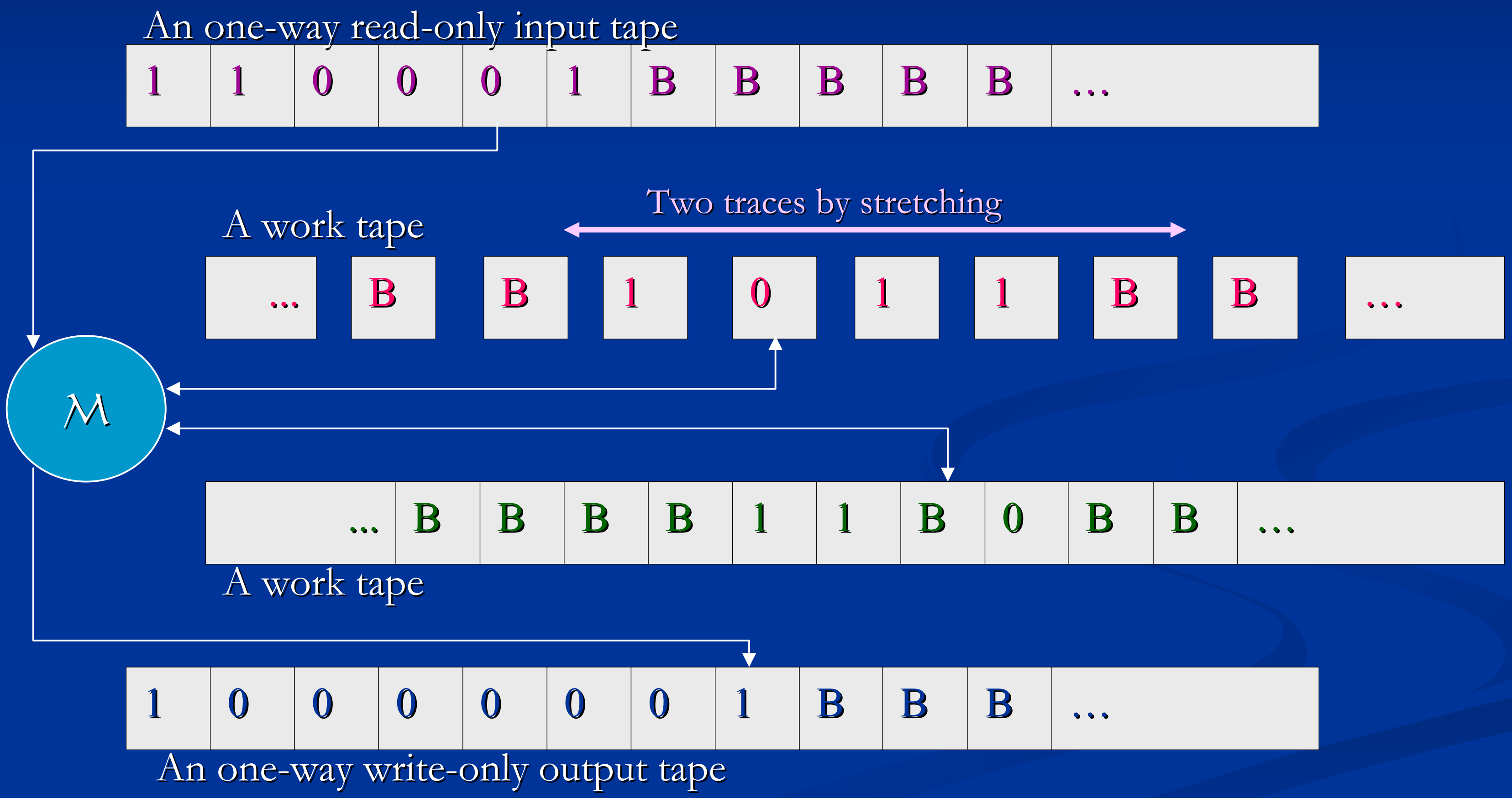
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



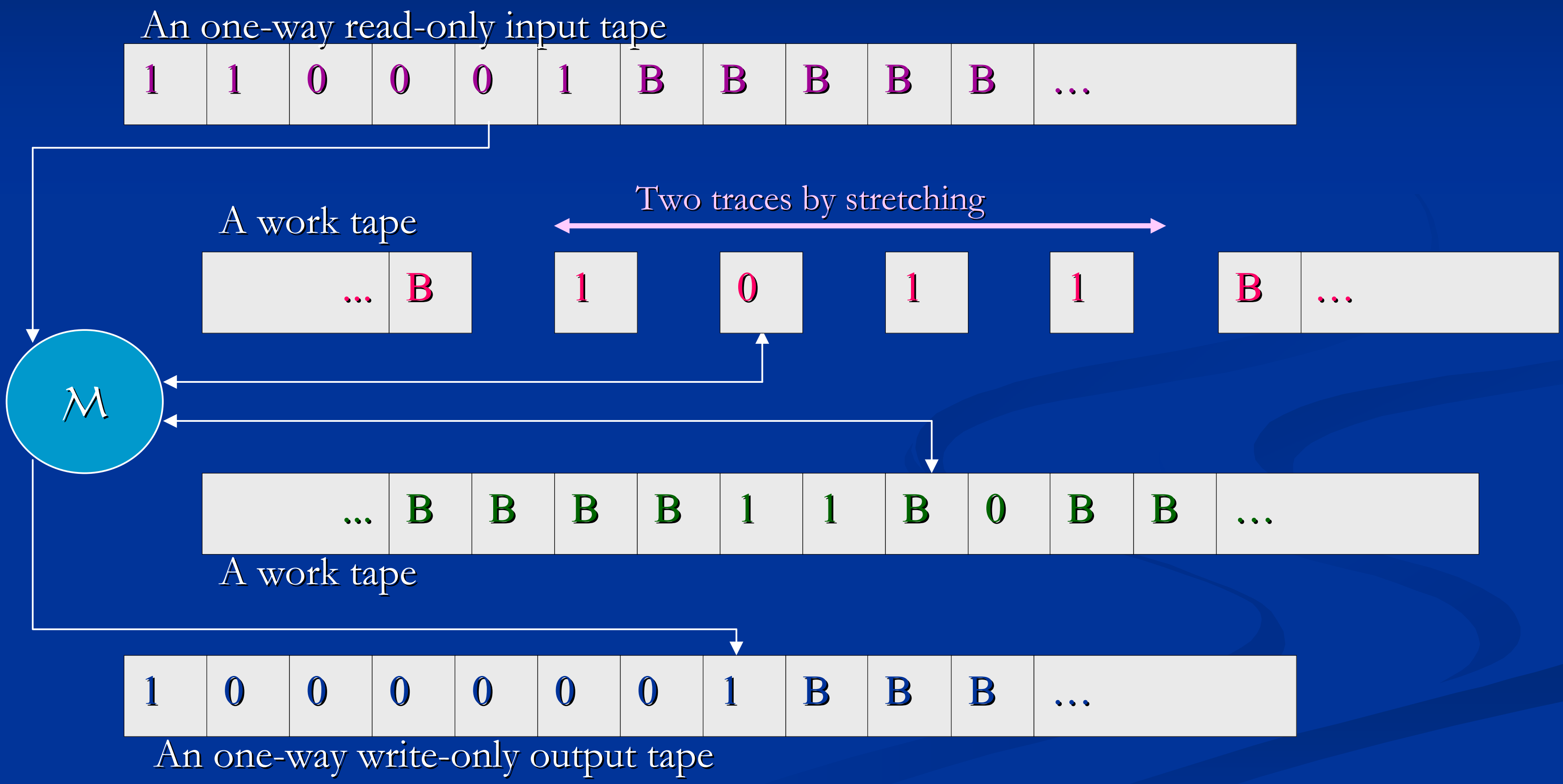
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



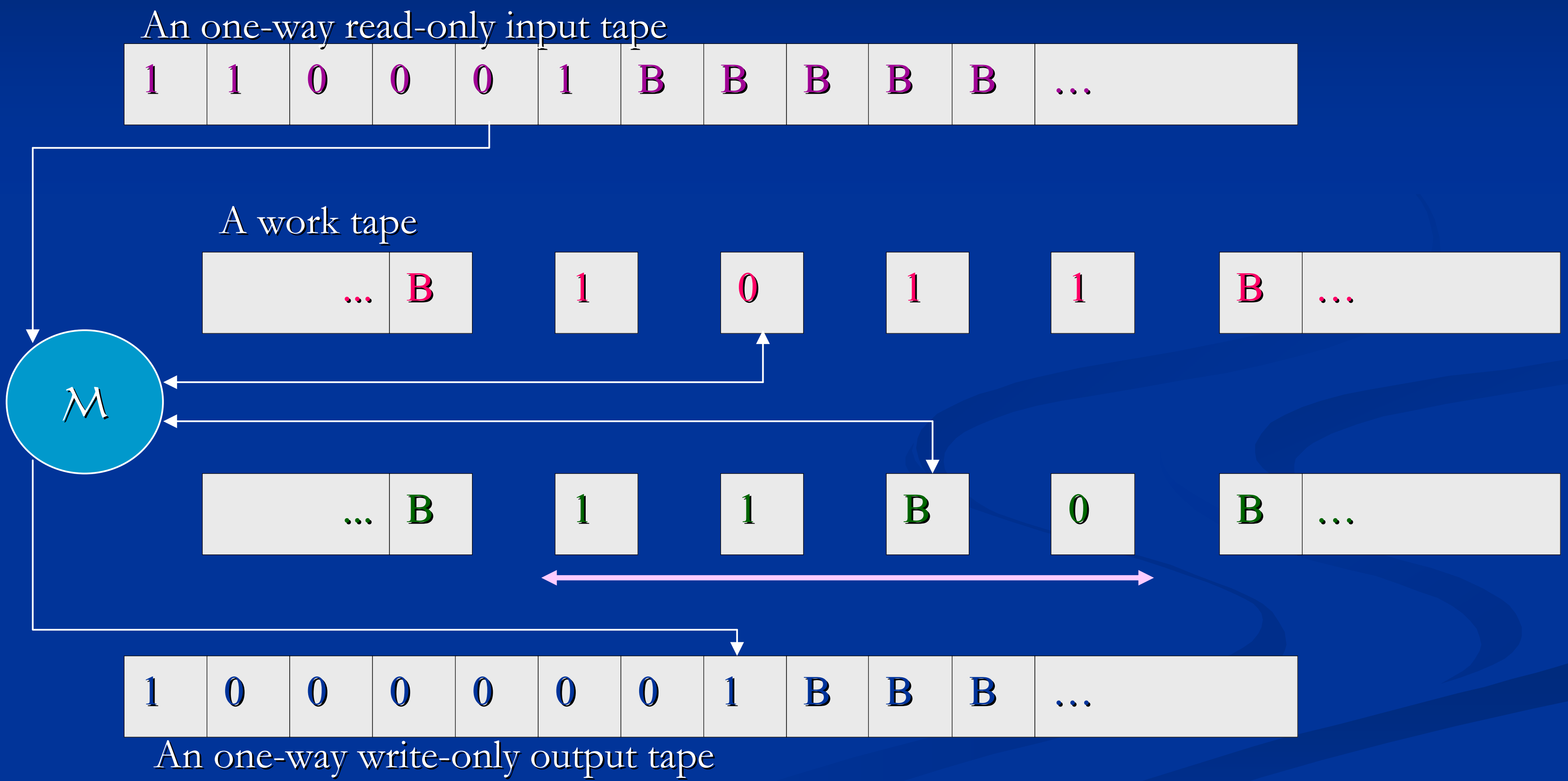
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



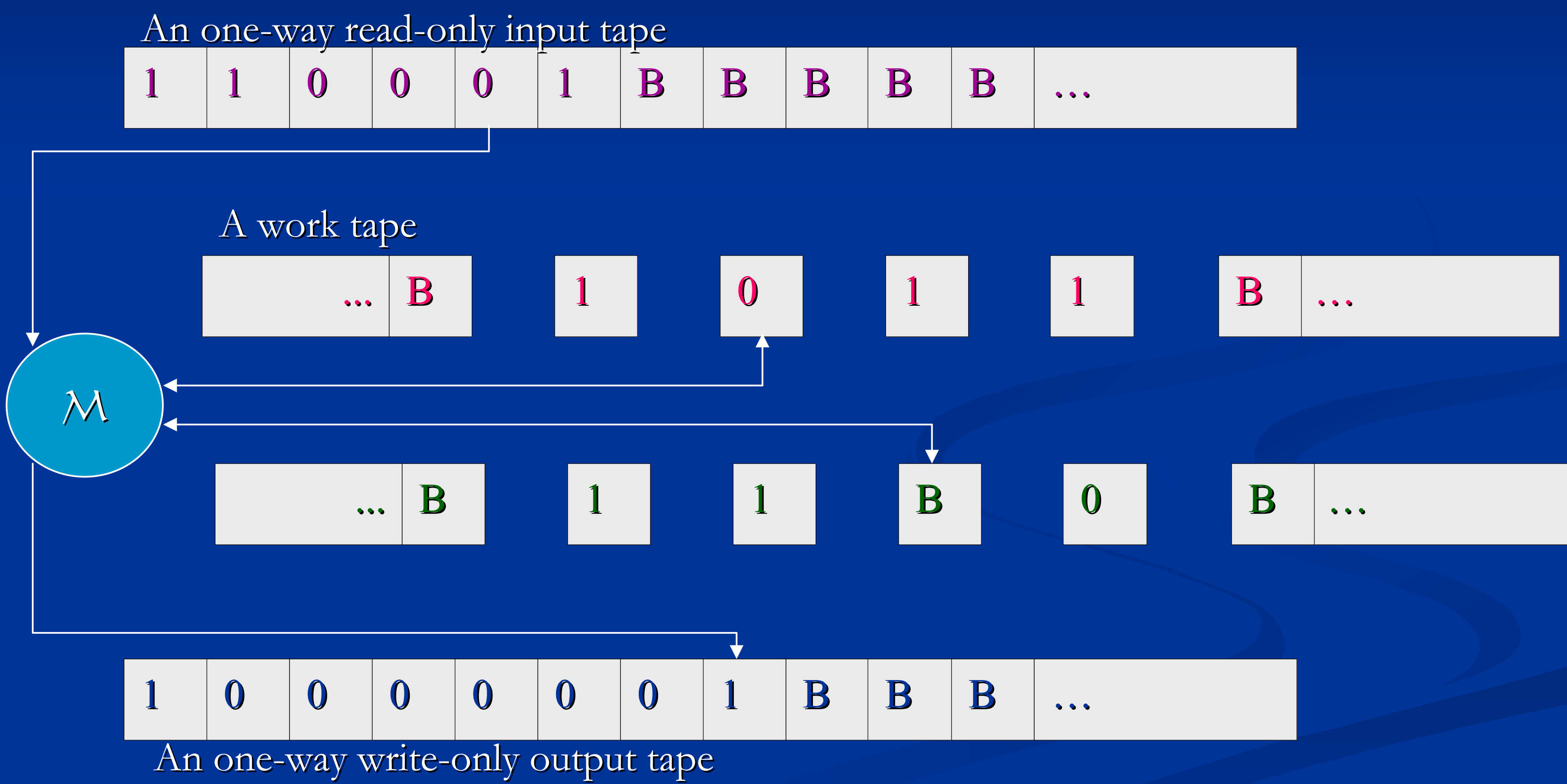
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



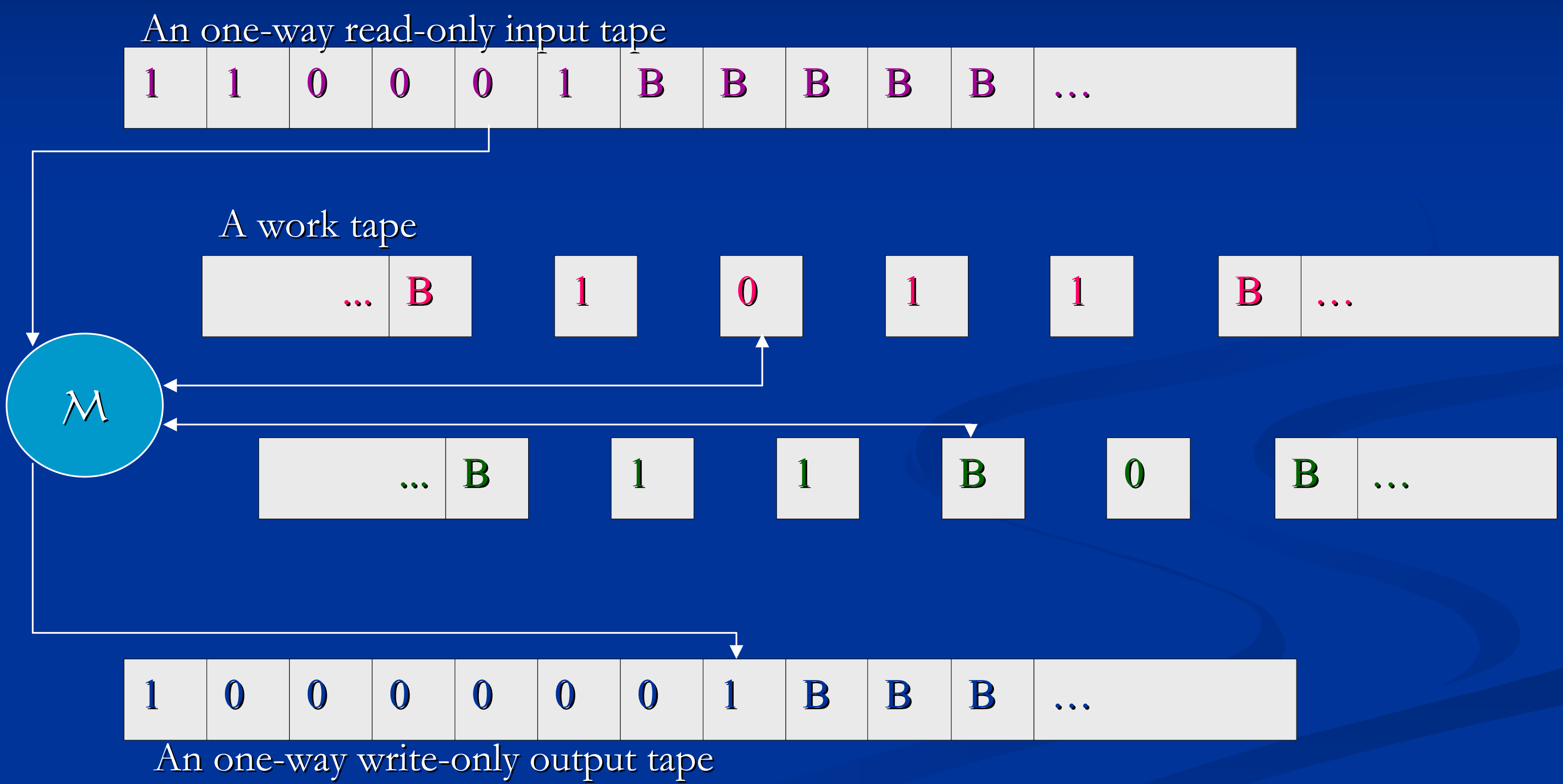
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



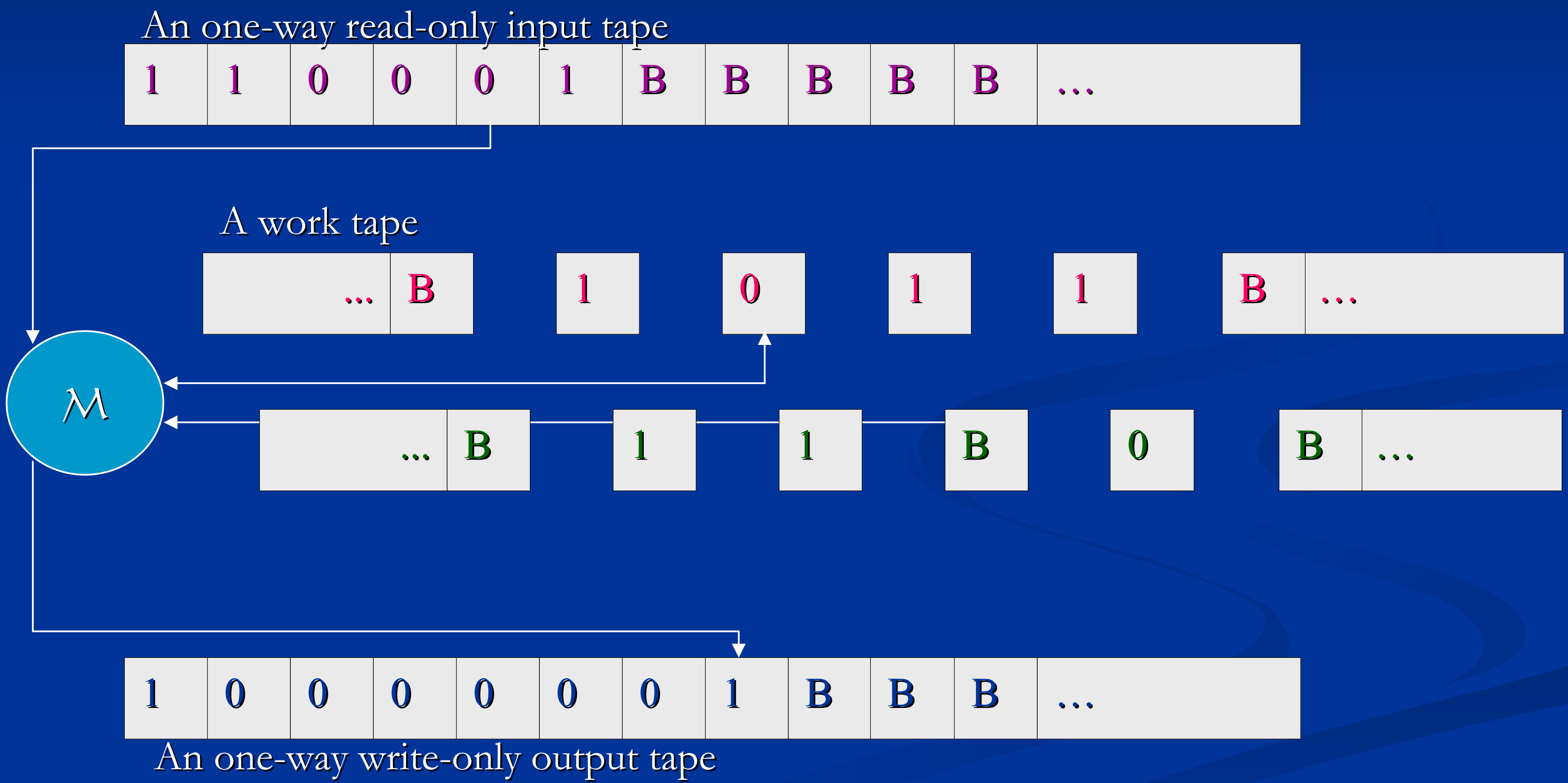
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



From a Turing machine over {0, 1} to a {0, 1}-BSS machine

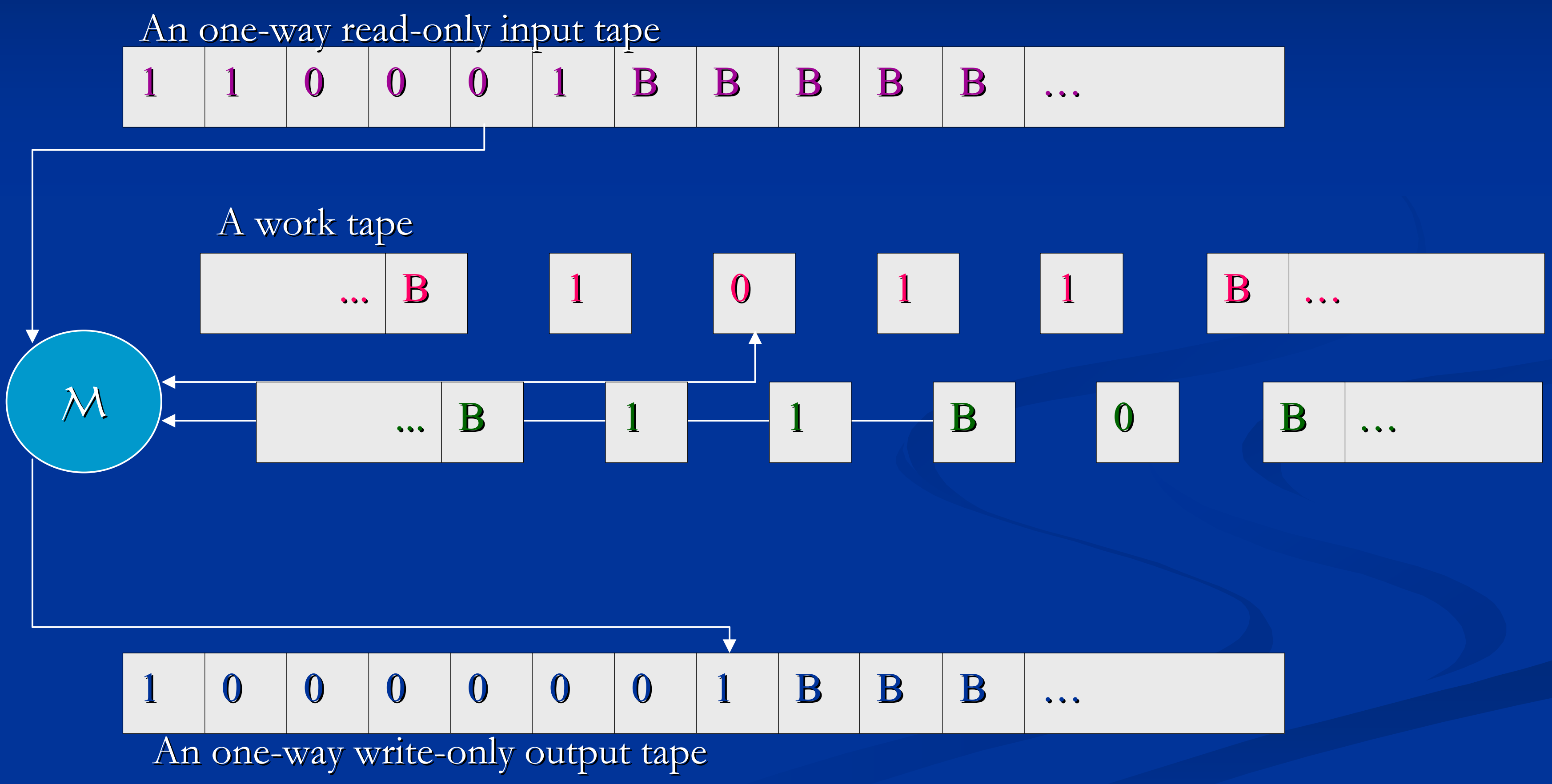


From a Turing machine over {0, 1} to a {0, 1}-BSS machine



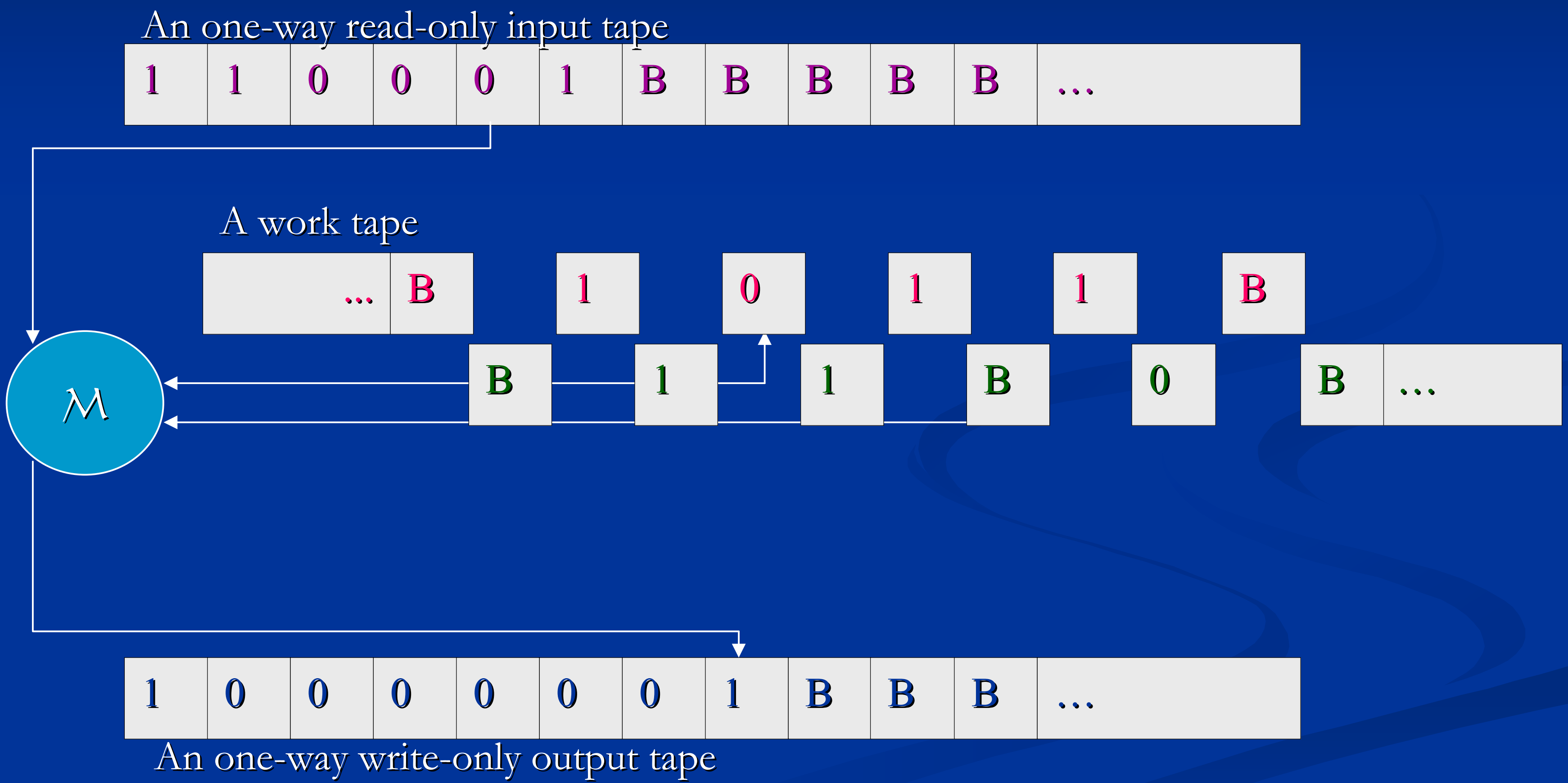
gassnerc@uni-greifswald.de

From a Turing machine over {0, 1} to a {0, 1}-BSS machine

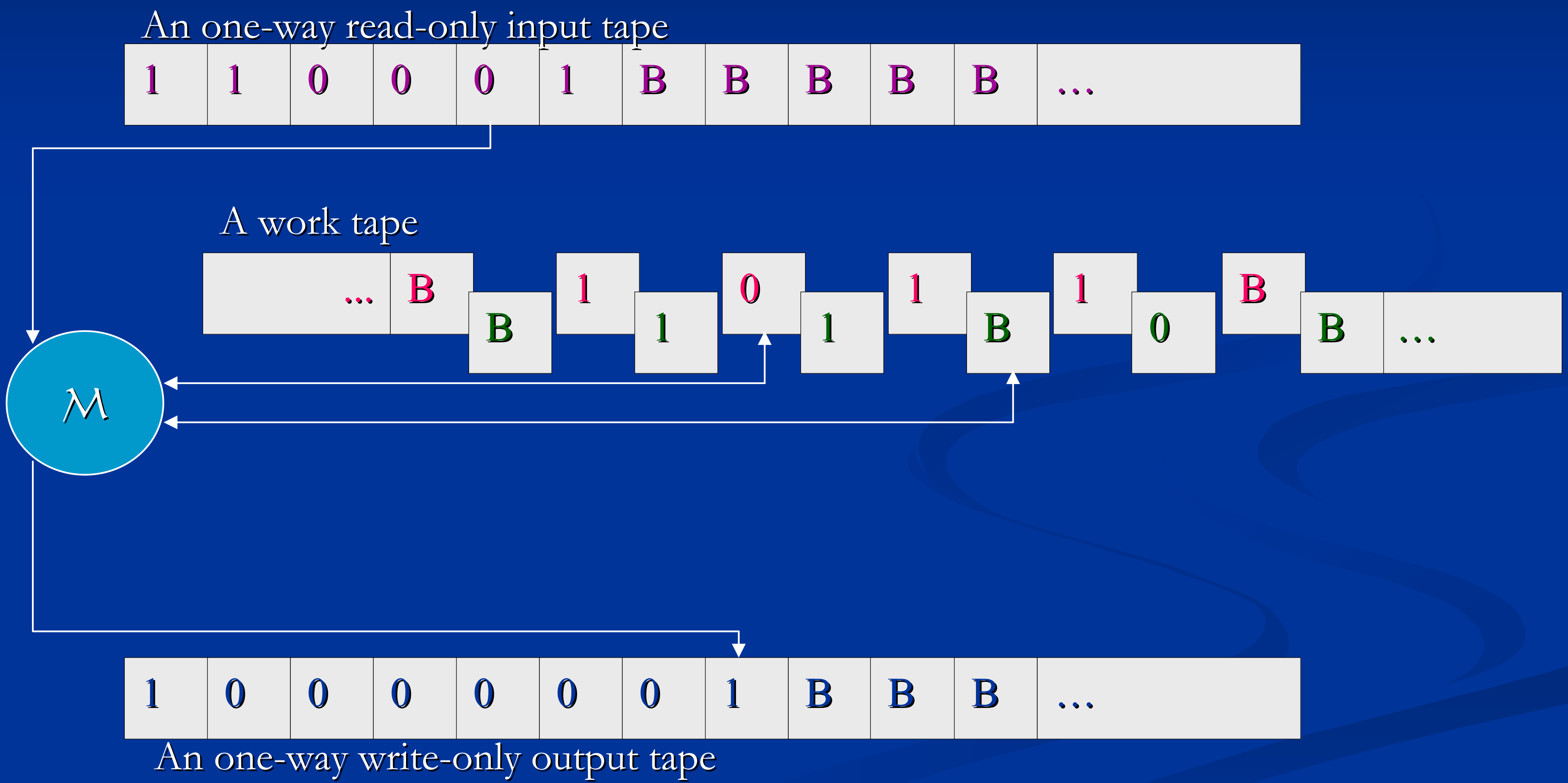


gassnerc@uni-greifswald.de

From a Turing machine over {0, 1} to a {0, 1}-BSS machine

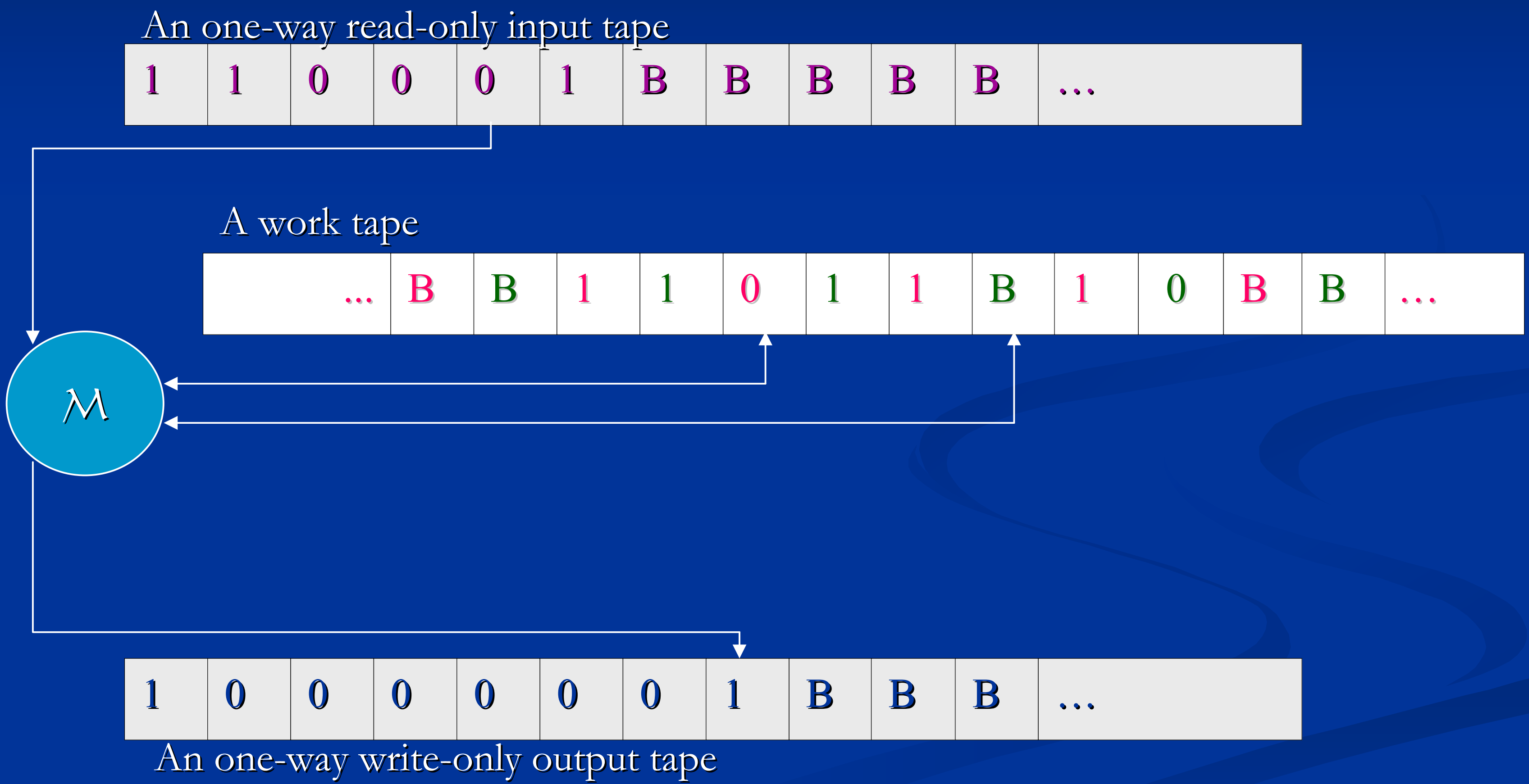


From a Turing machine over {0, 1} to a {0, 1}-BSS machine

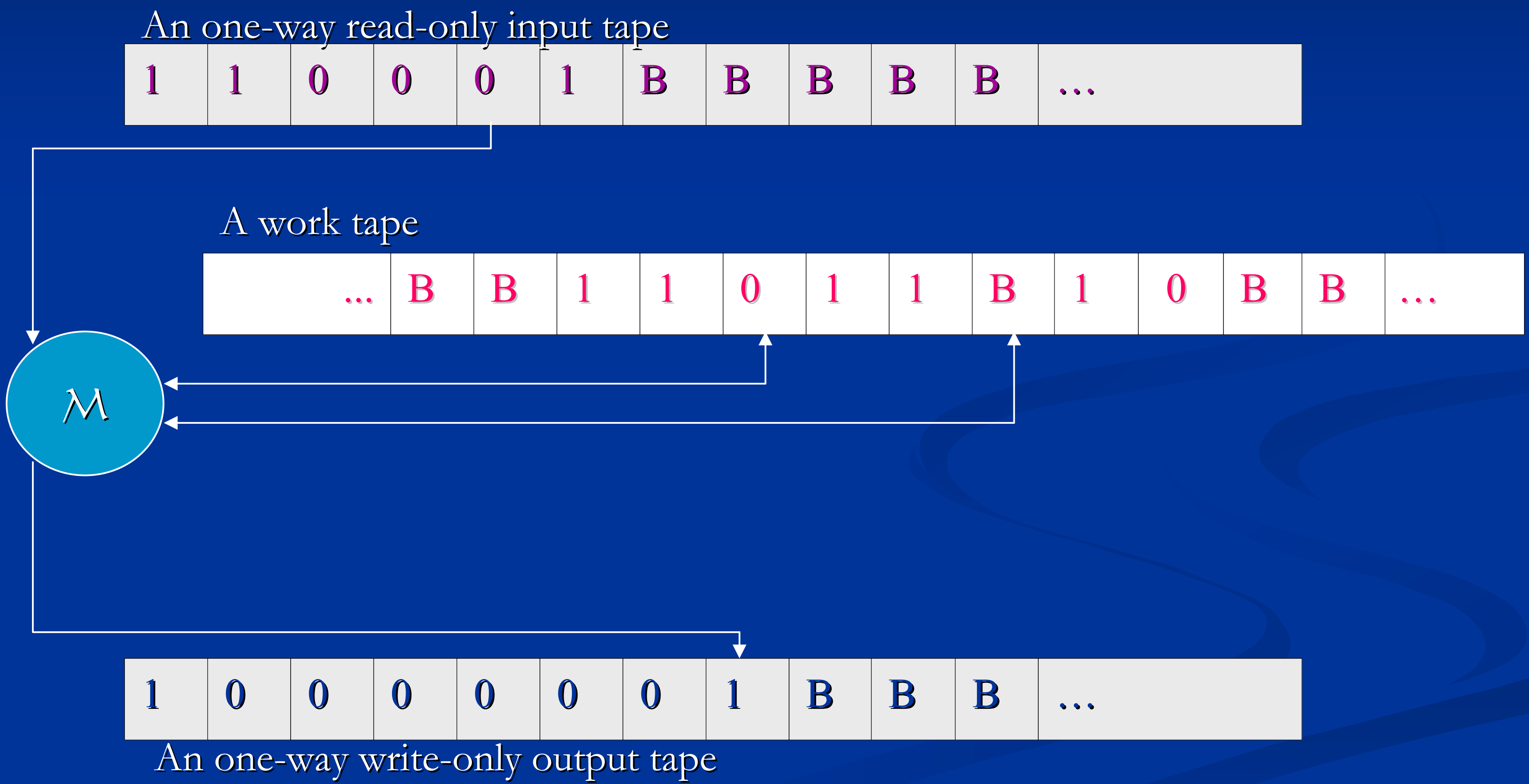


gassnerc@uni-greifswald.de

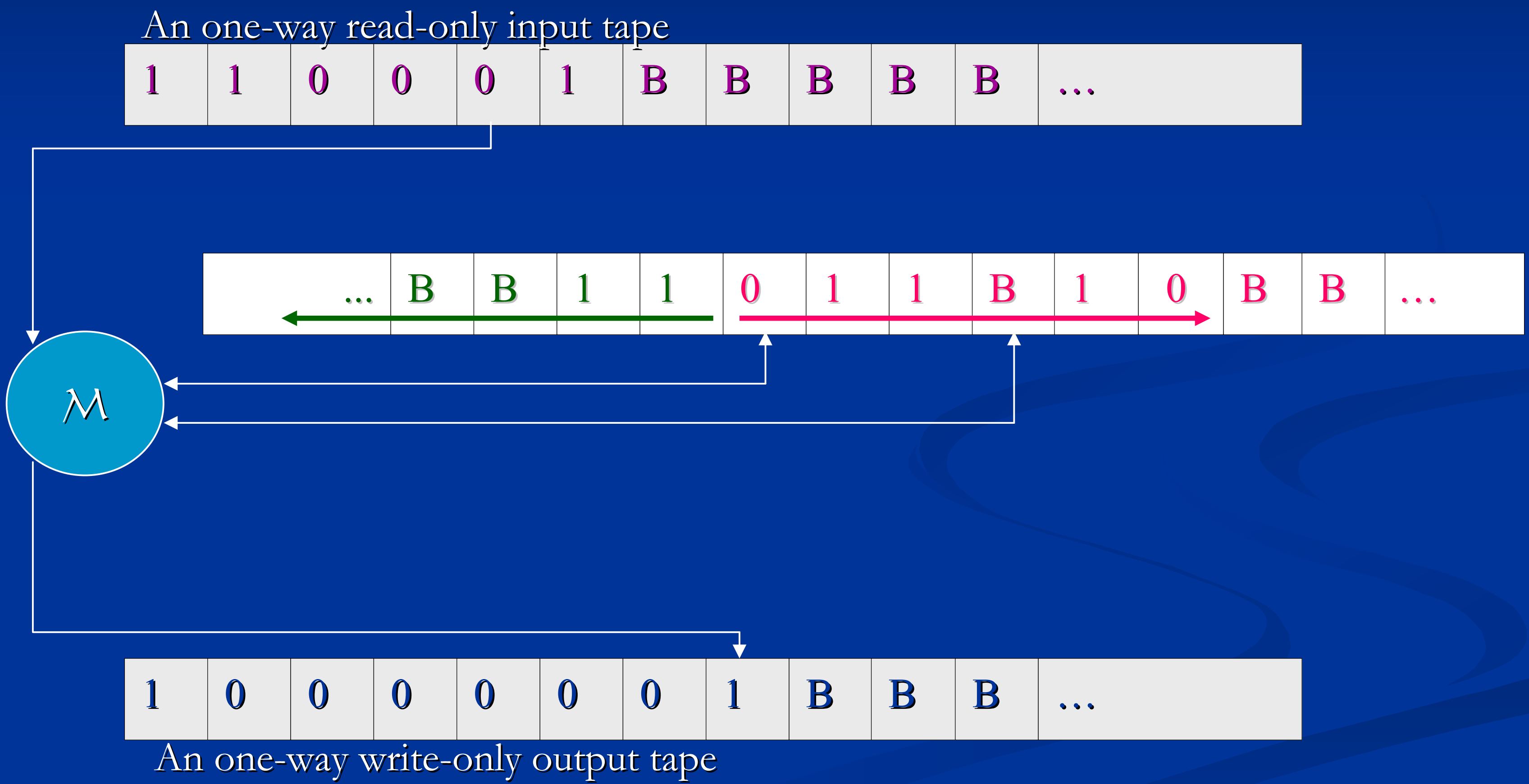
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



From a Turing machine over {0, 1} to a {0, 1}-BSS machine

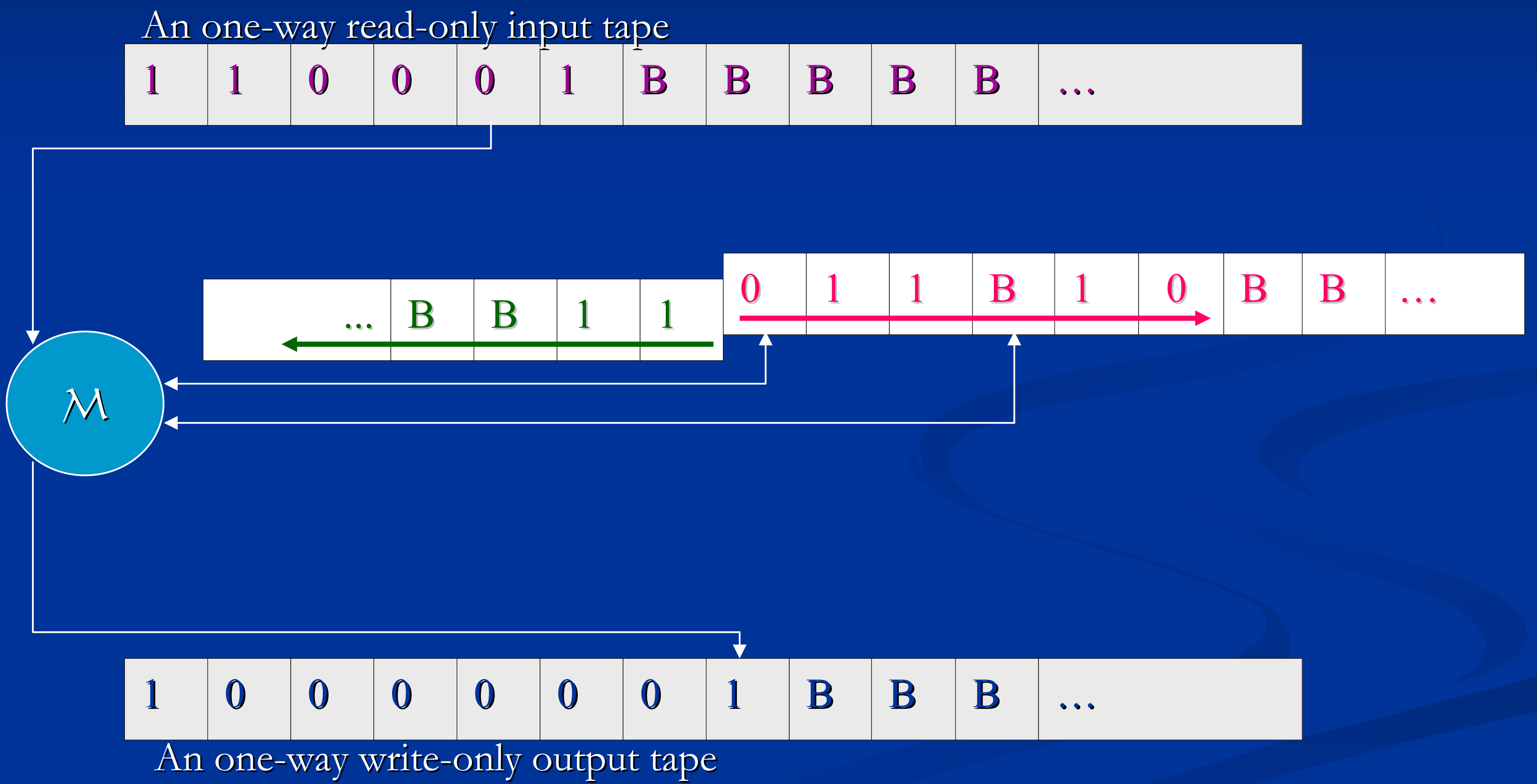


From a Turing machine over {0, 1} to a {0, 1}-BSS machine



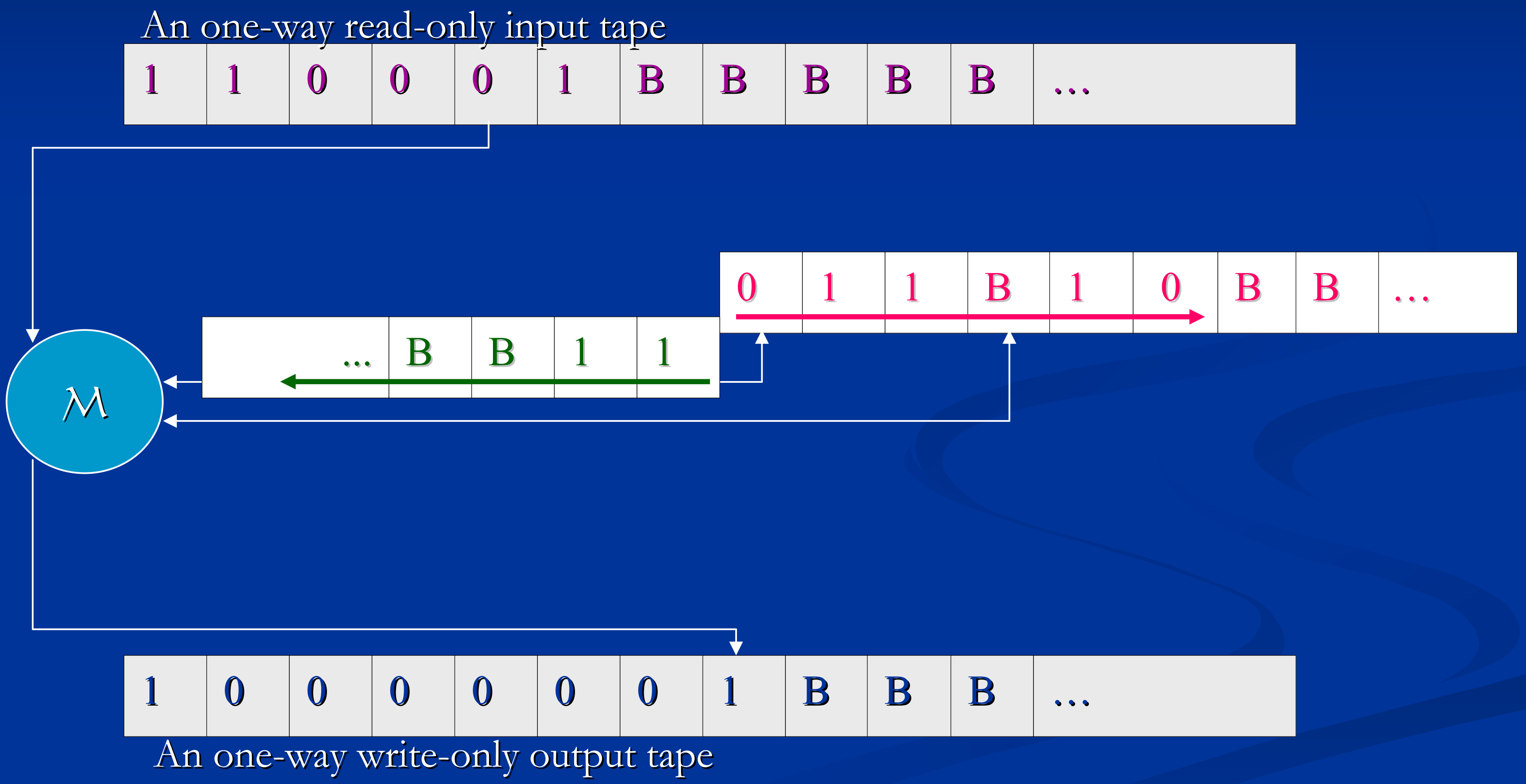
gassnerc@uni-greifswald.de

From a Turing machine over {0, 1} to a {0, 1}-BSS machine

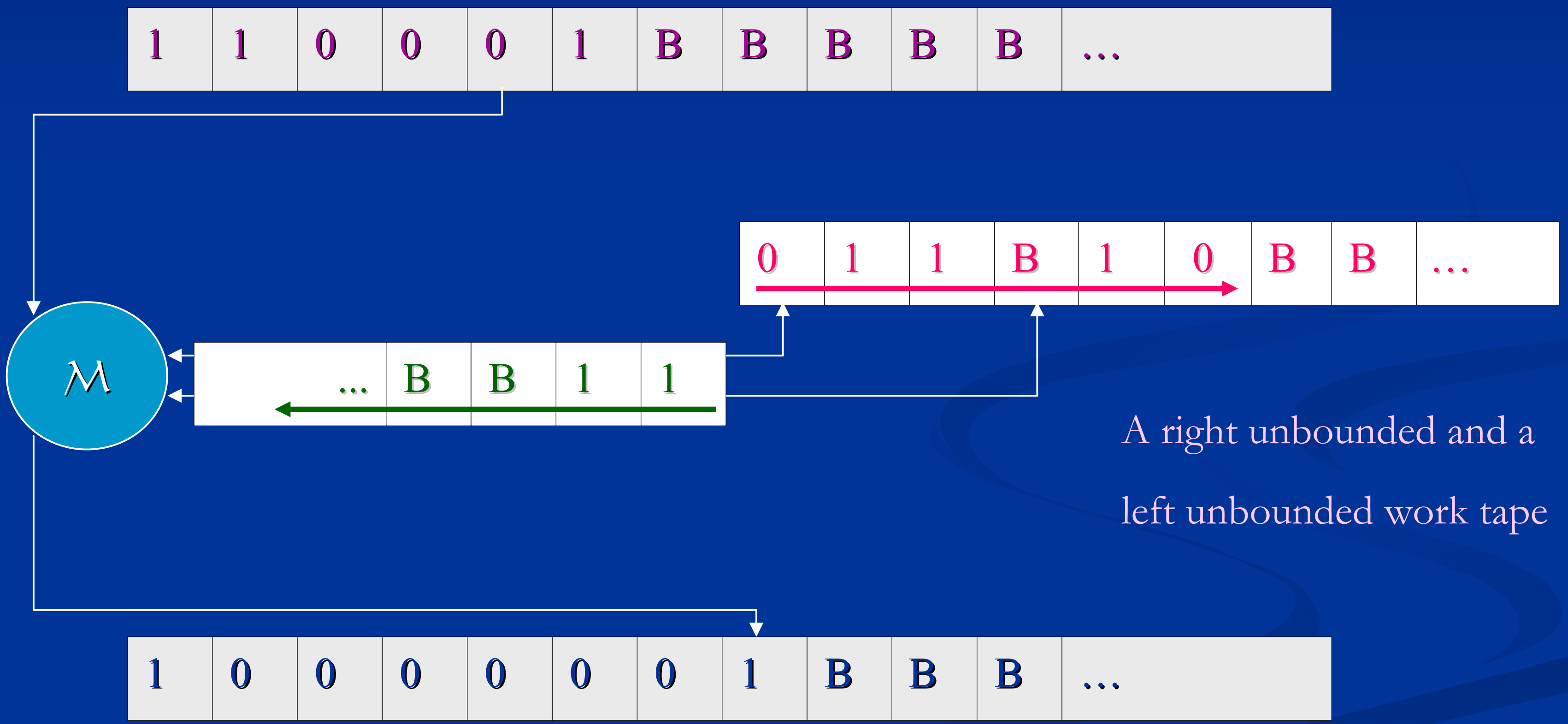


gassnerc@uni-greifswald.de

From a Turing machine over {0, 1} to a {0, 1}-BSS machine

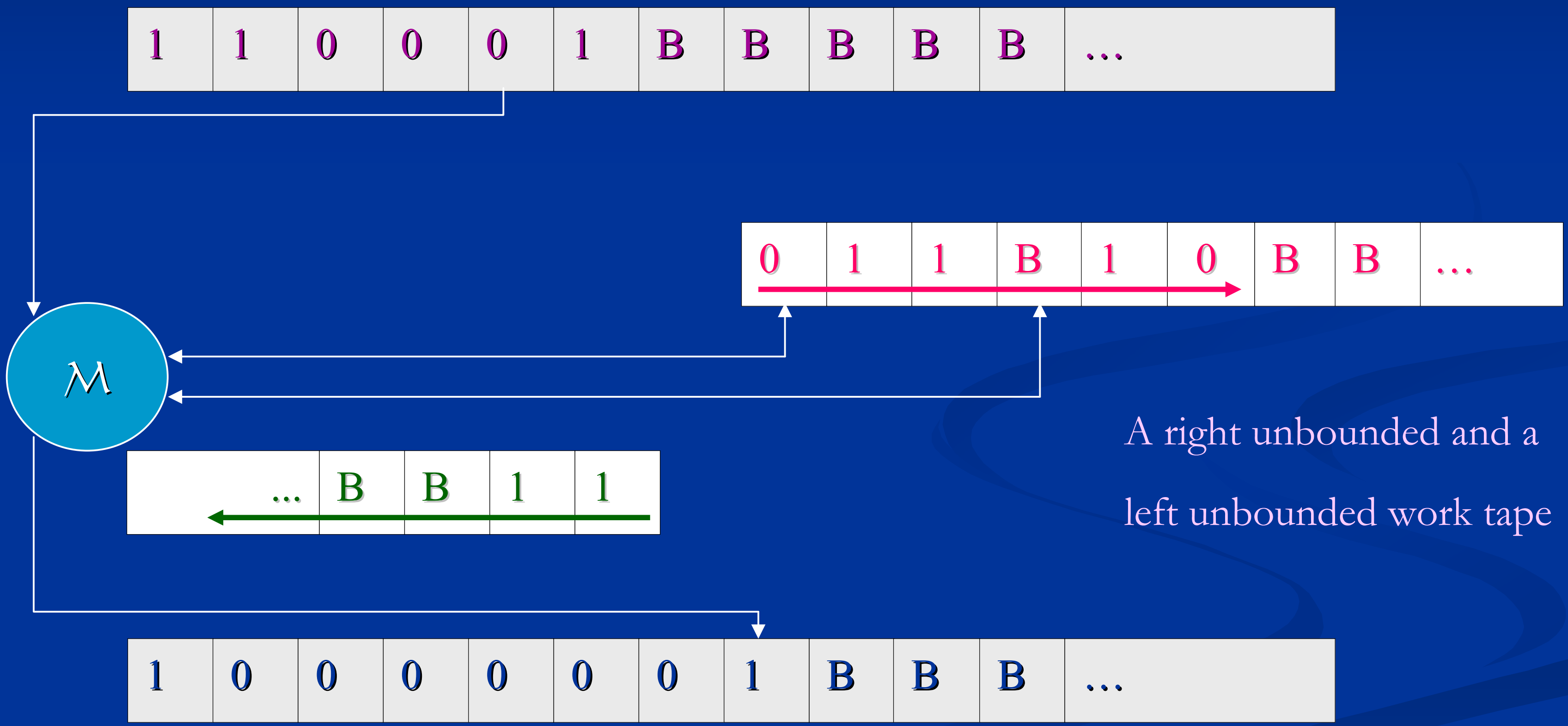


From a Turing machine over {0, 1} to a {0, 1}-BSS machine



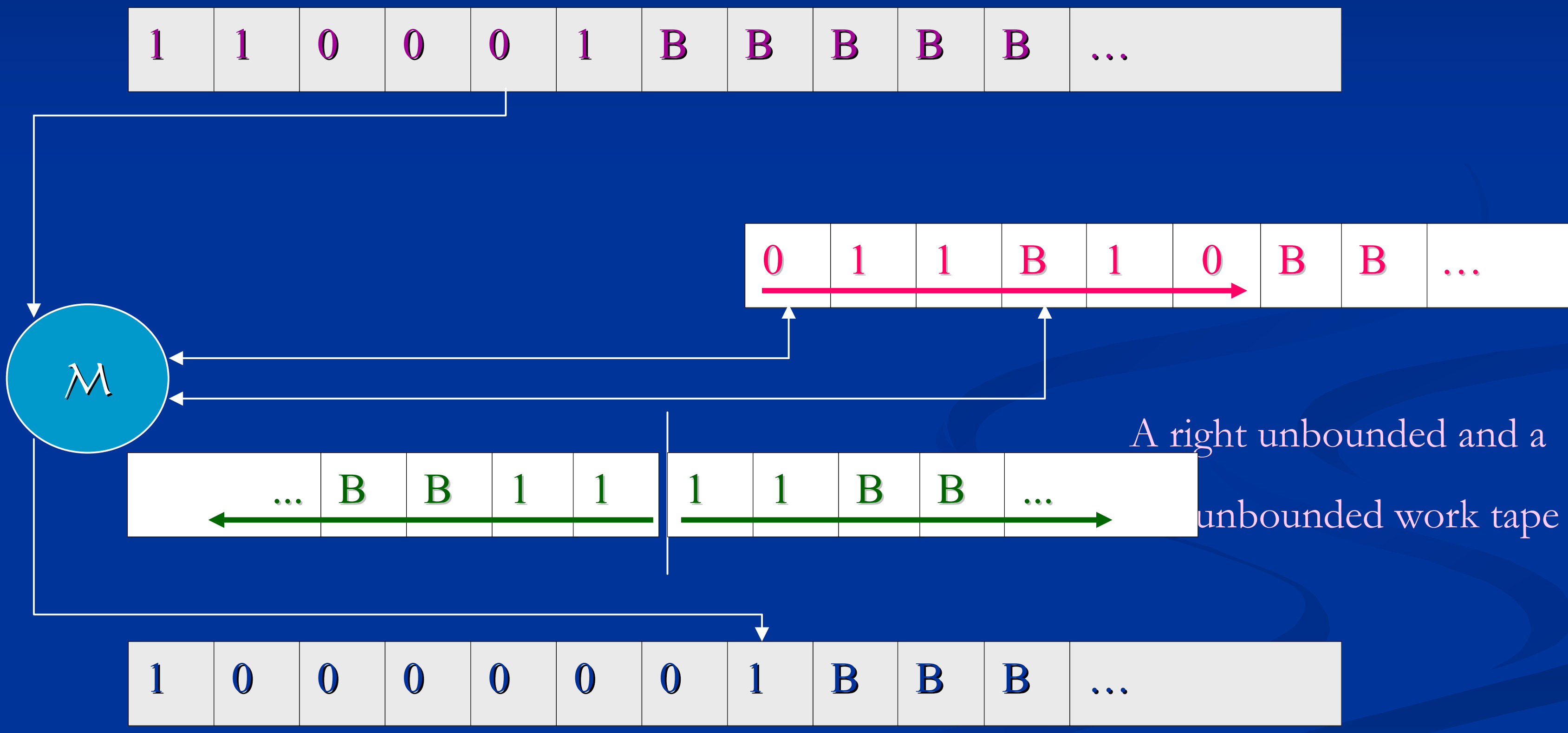
A right unbounded and a left unbounded work tape

From a Turing machine over {0, 1} to a {0, 1}-BSS machine



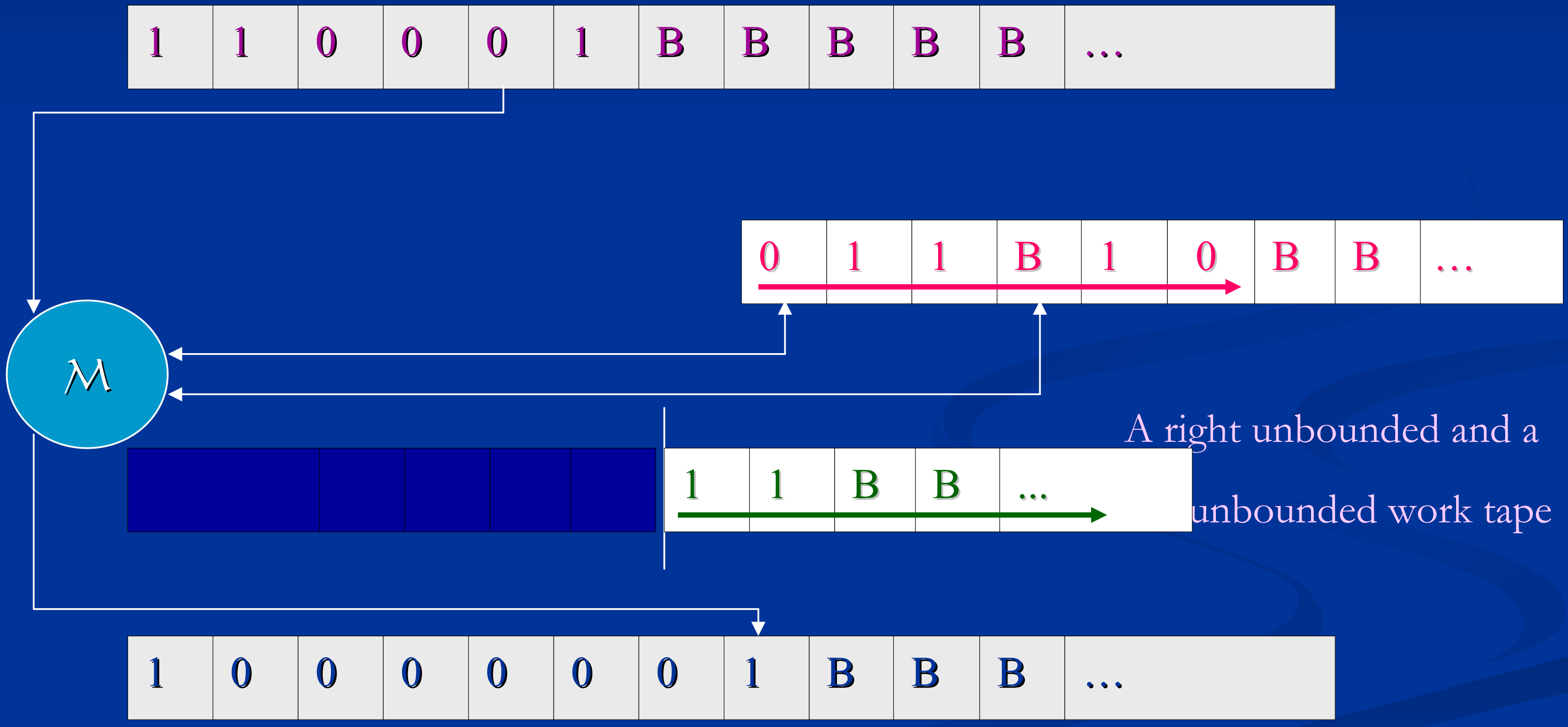
A right unbounded and a left unbounded work tape

From a Turing machine over {0, 1} to a {0, 1}-BSS machine



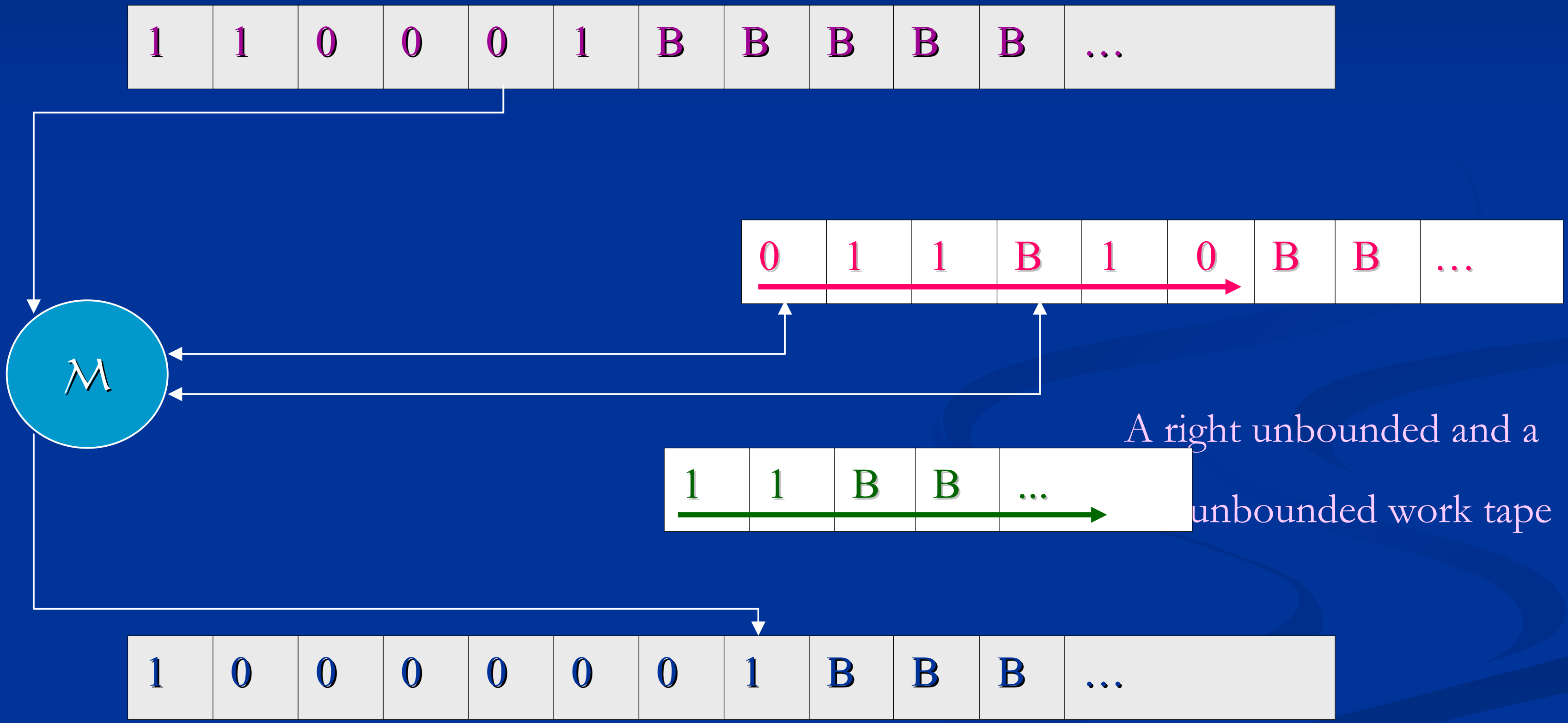
gassnerc@uni-greifswald.de

From a Turing machine over {0, 1} to a {0, 1}-BSS machine



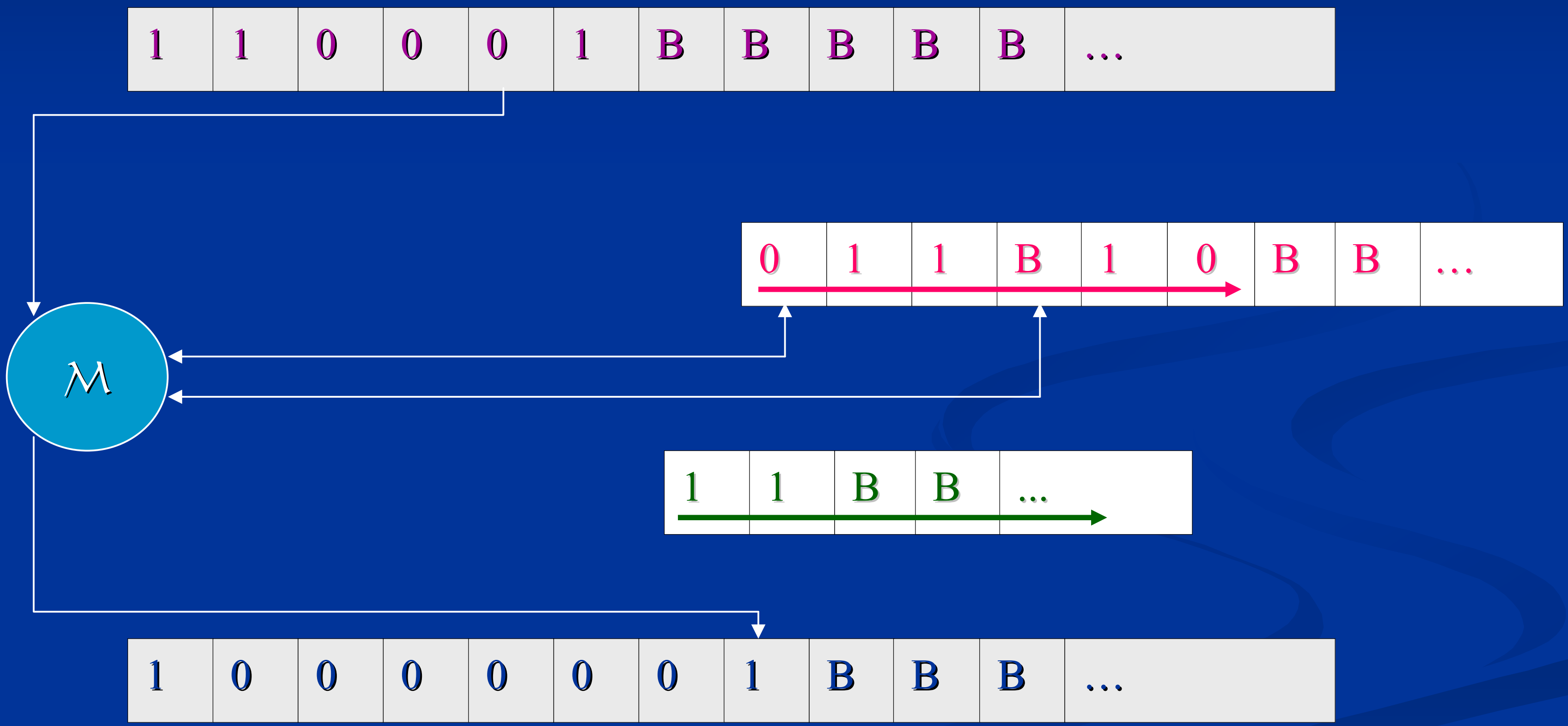
A right unbounded and a unbounded work tape

From a Turing machine over {0, 1} to a {0, 1}-BSS machine

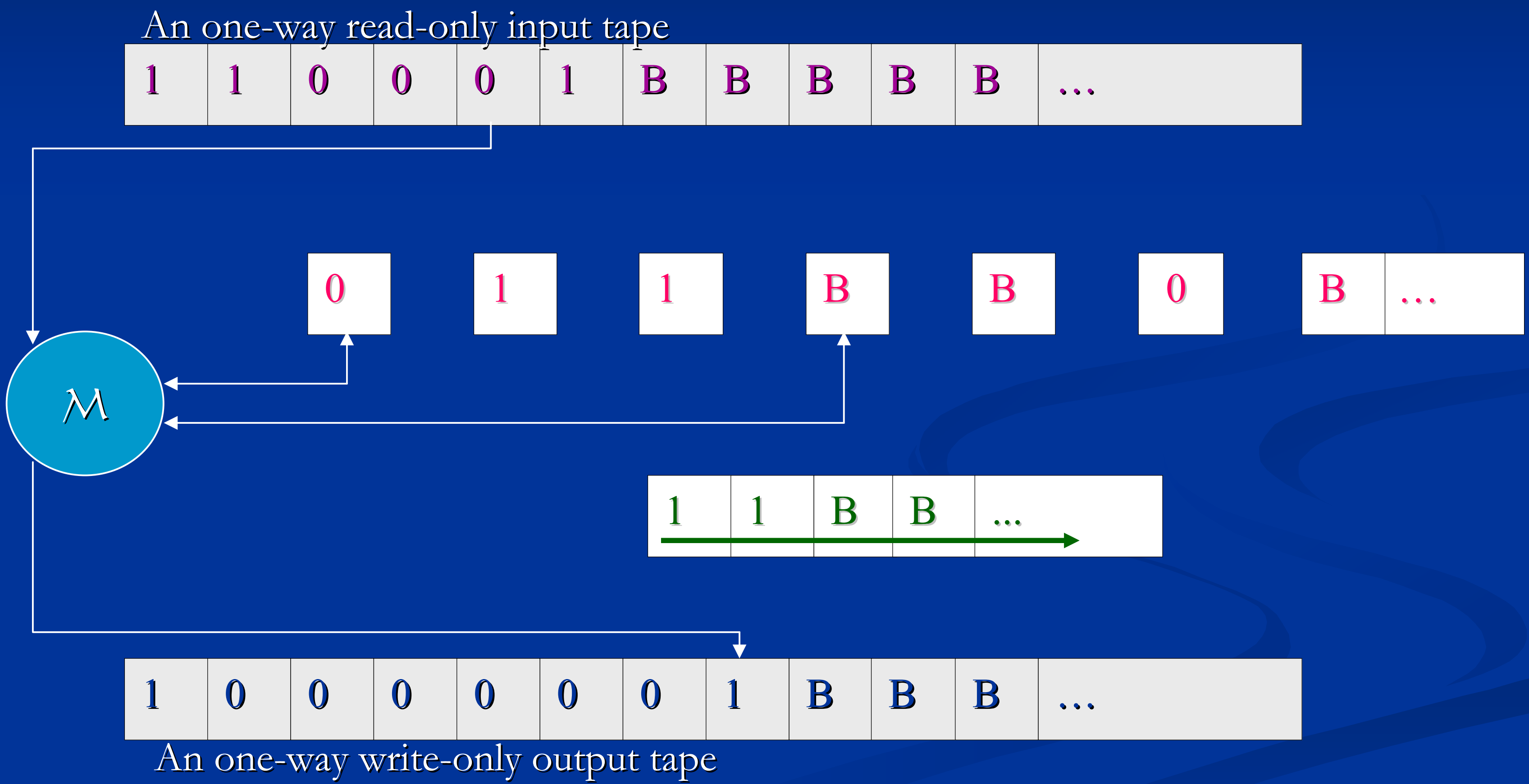


A right unbounded and a unbounded work tape

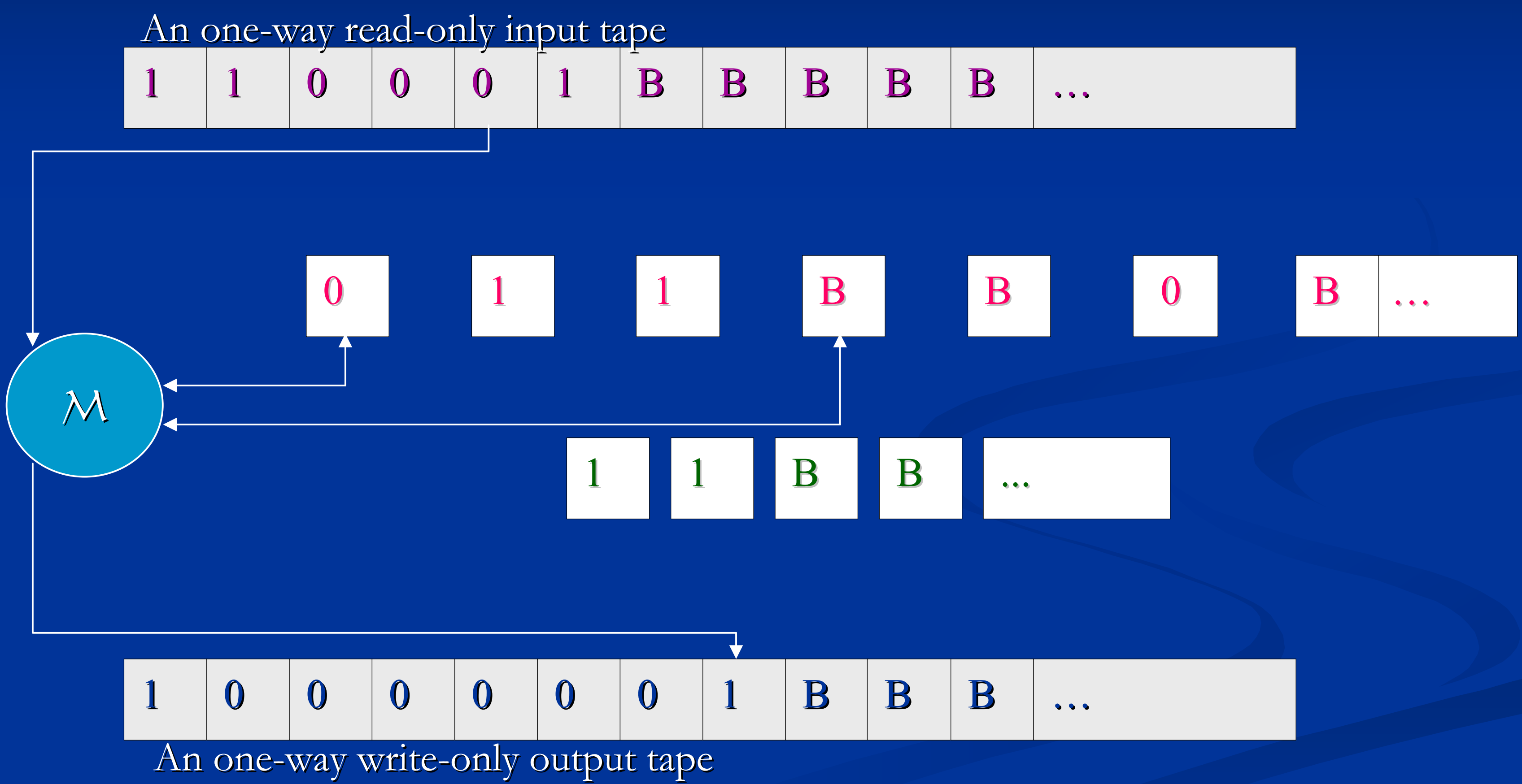
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



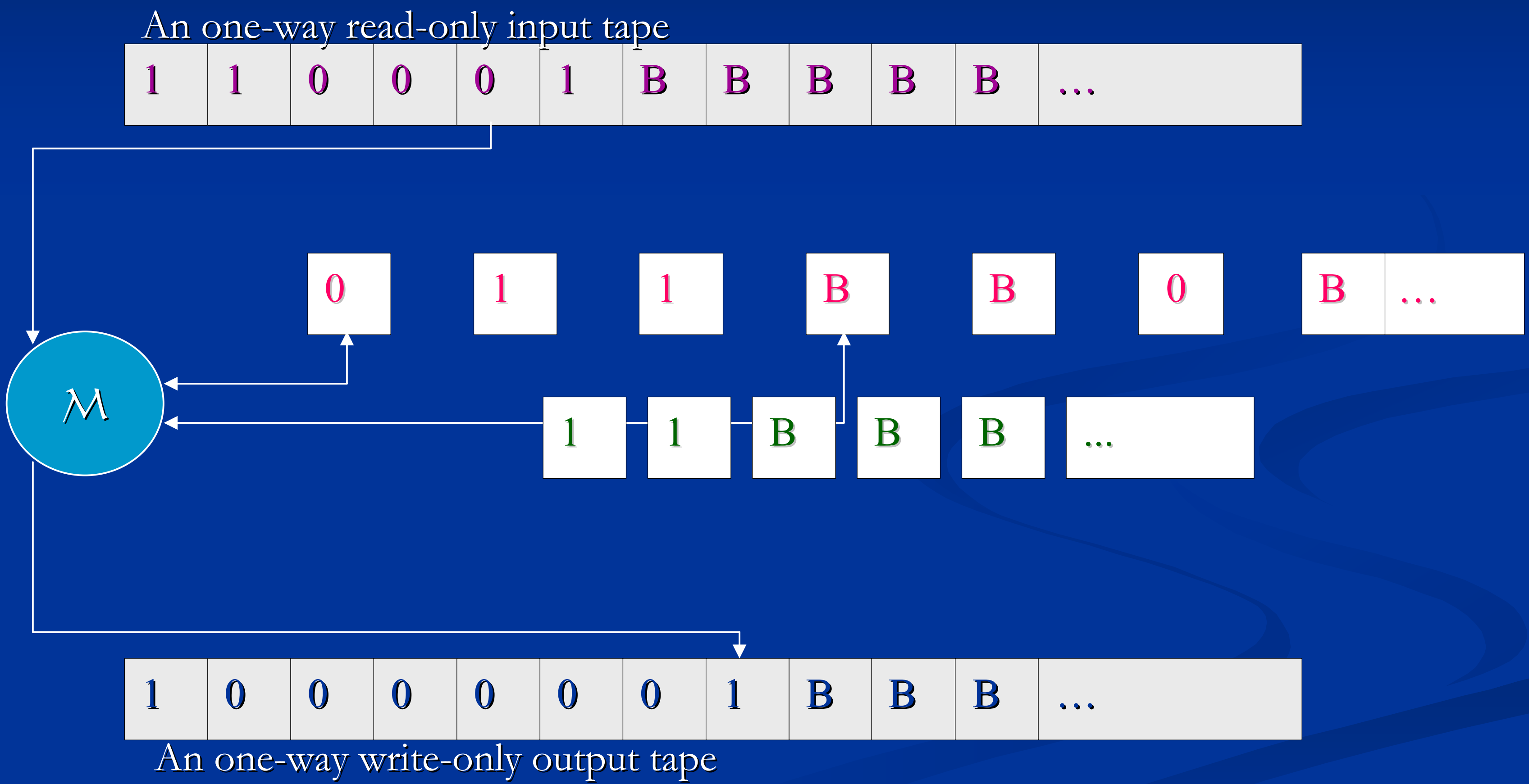
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



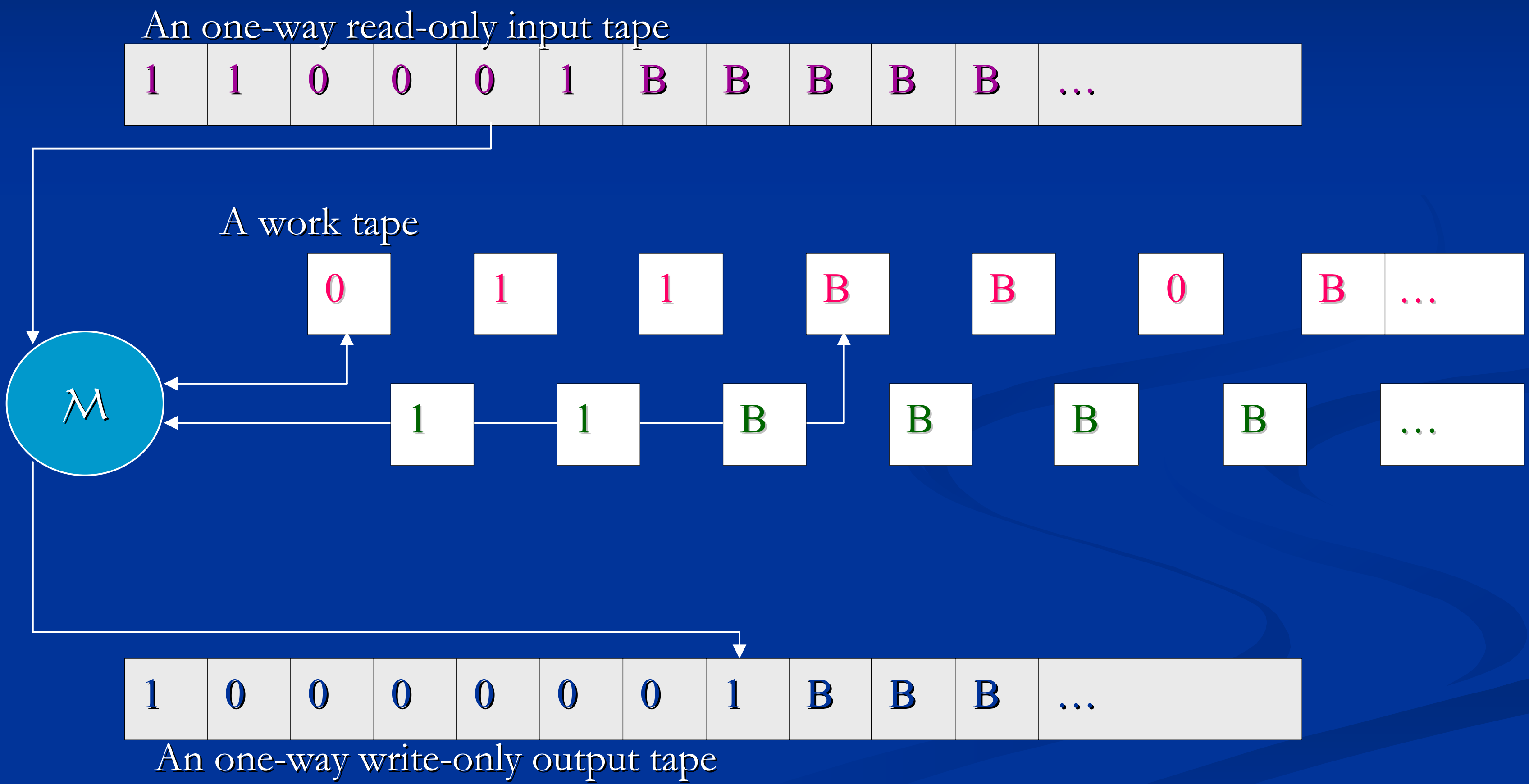
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



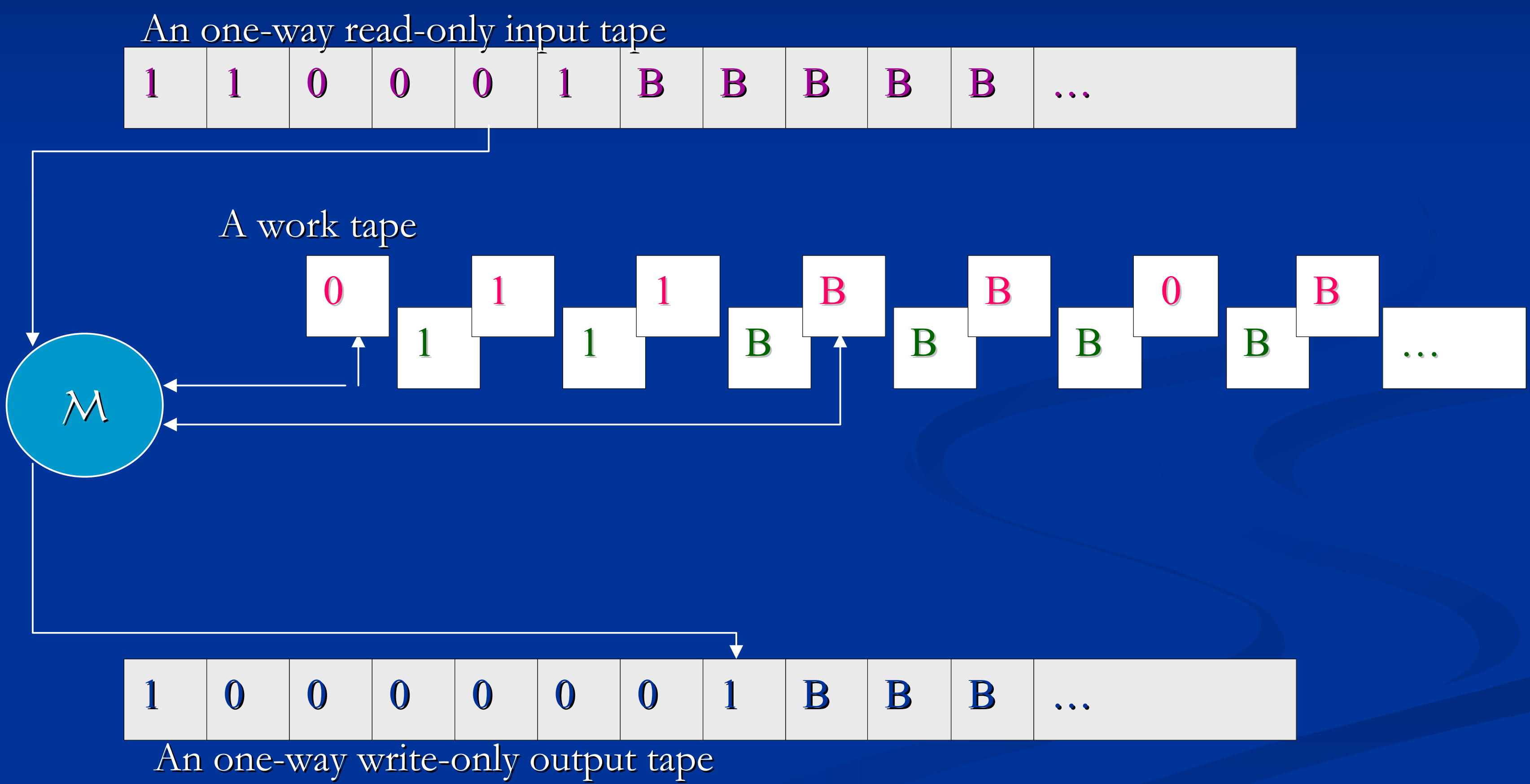
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



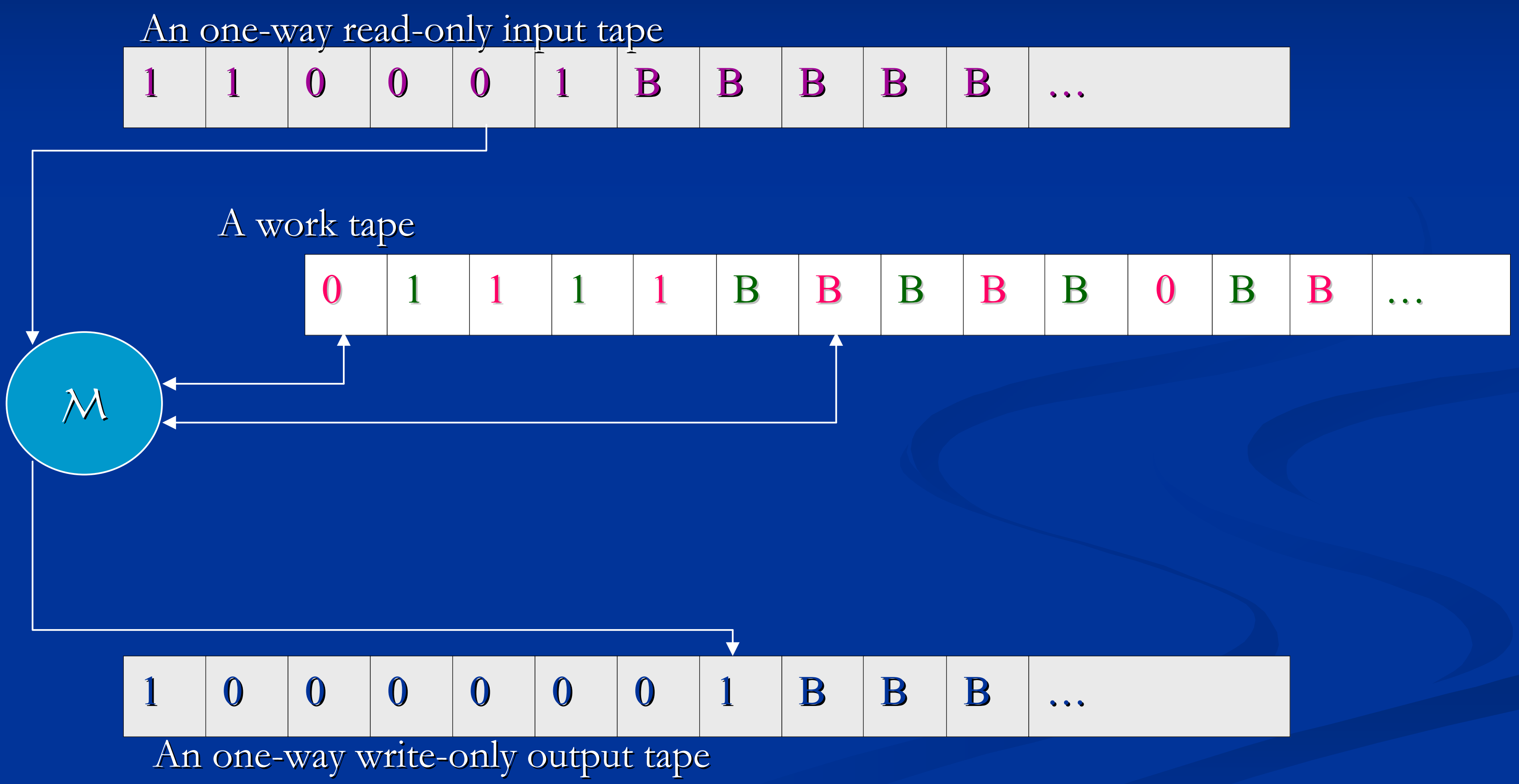
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



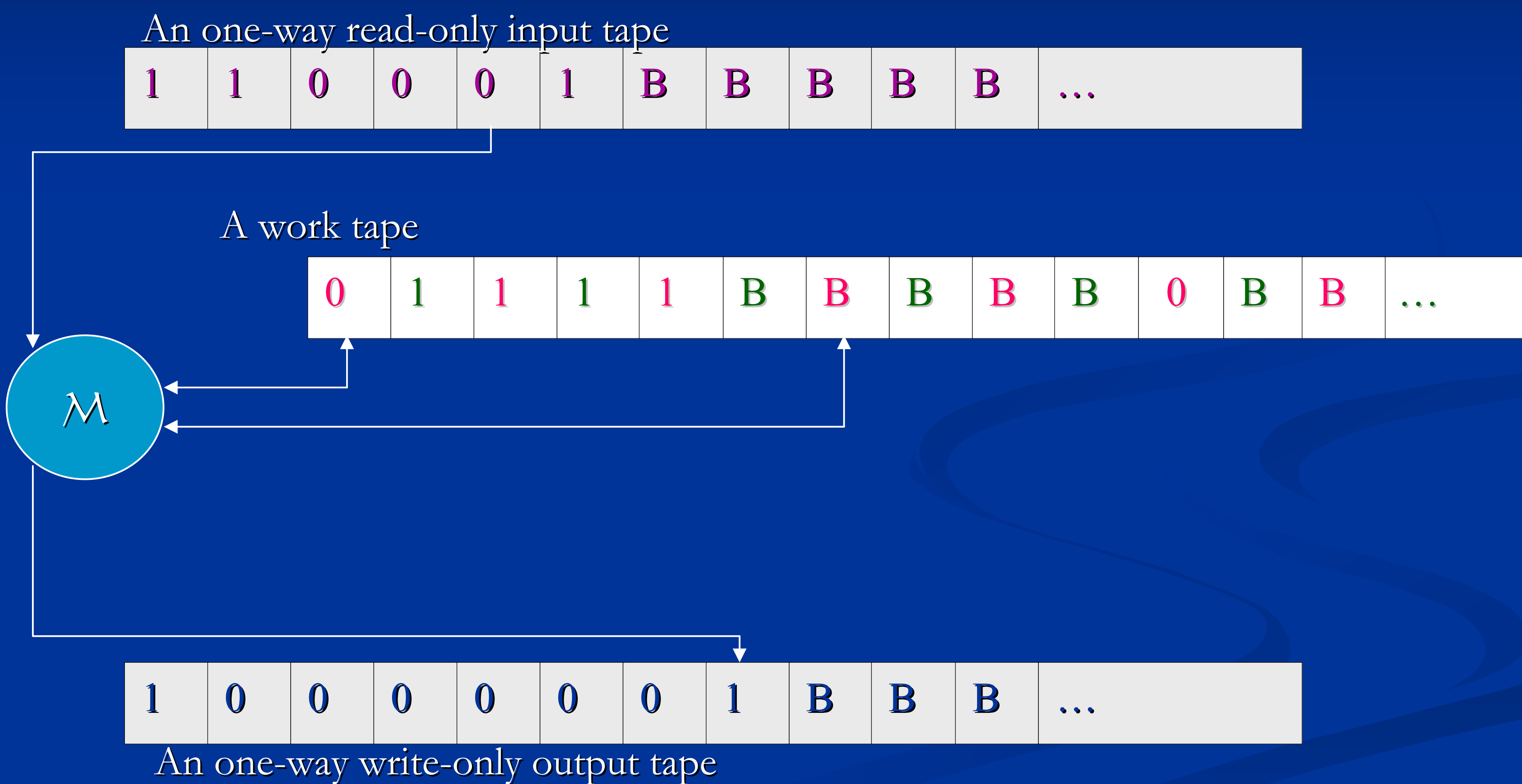
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



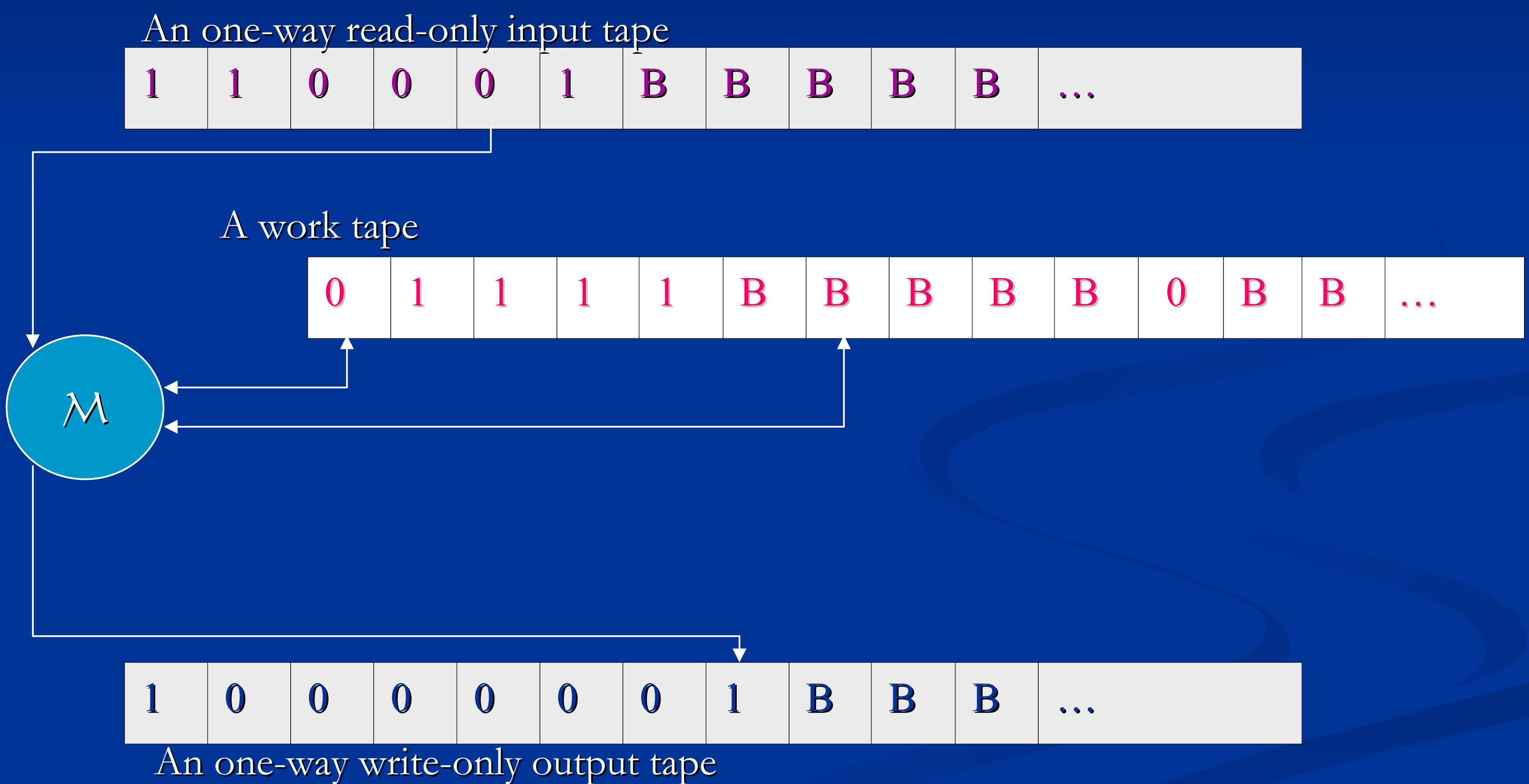
From a Turing machine over {0, 1} to a {0, 1}-BSS machine



From a Turing machine over {0, 1} to a {0, 1}-BSS machine

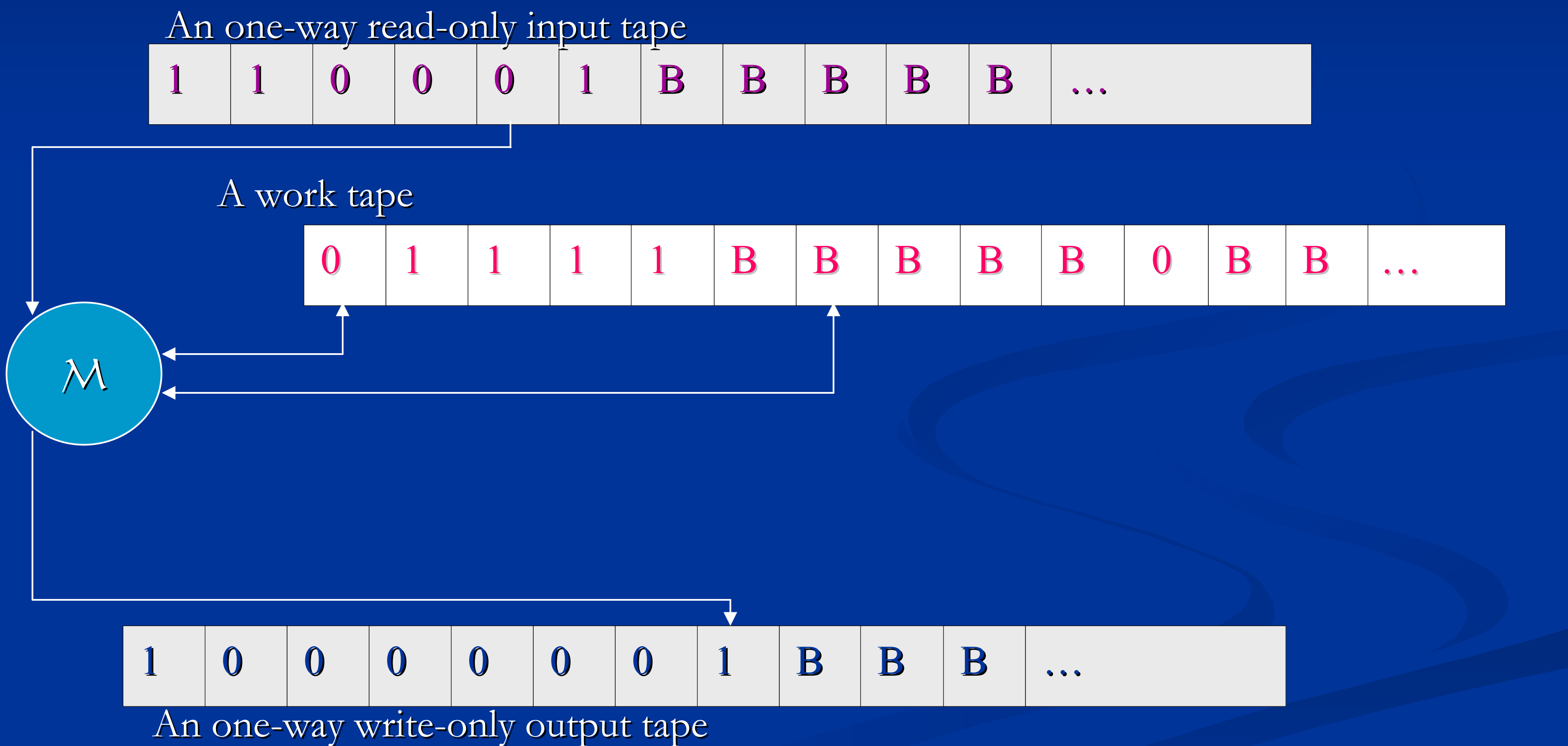


From a Turing machine over {0, 1} to a {0, 1}-BSS machine

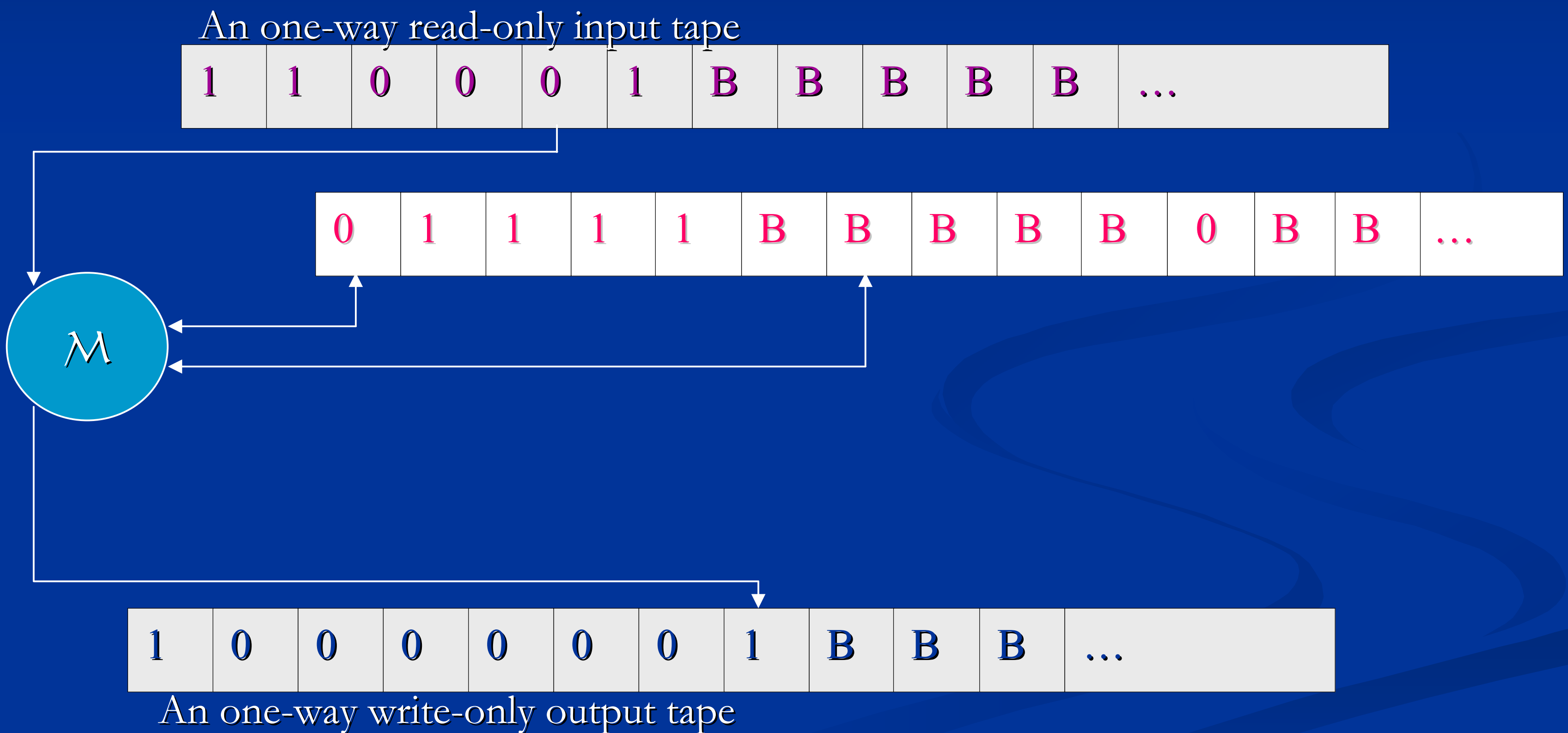


gassnerc@uni-greifswald.de

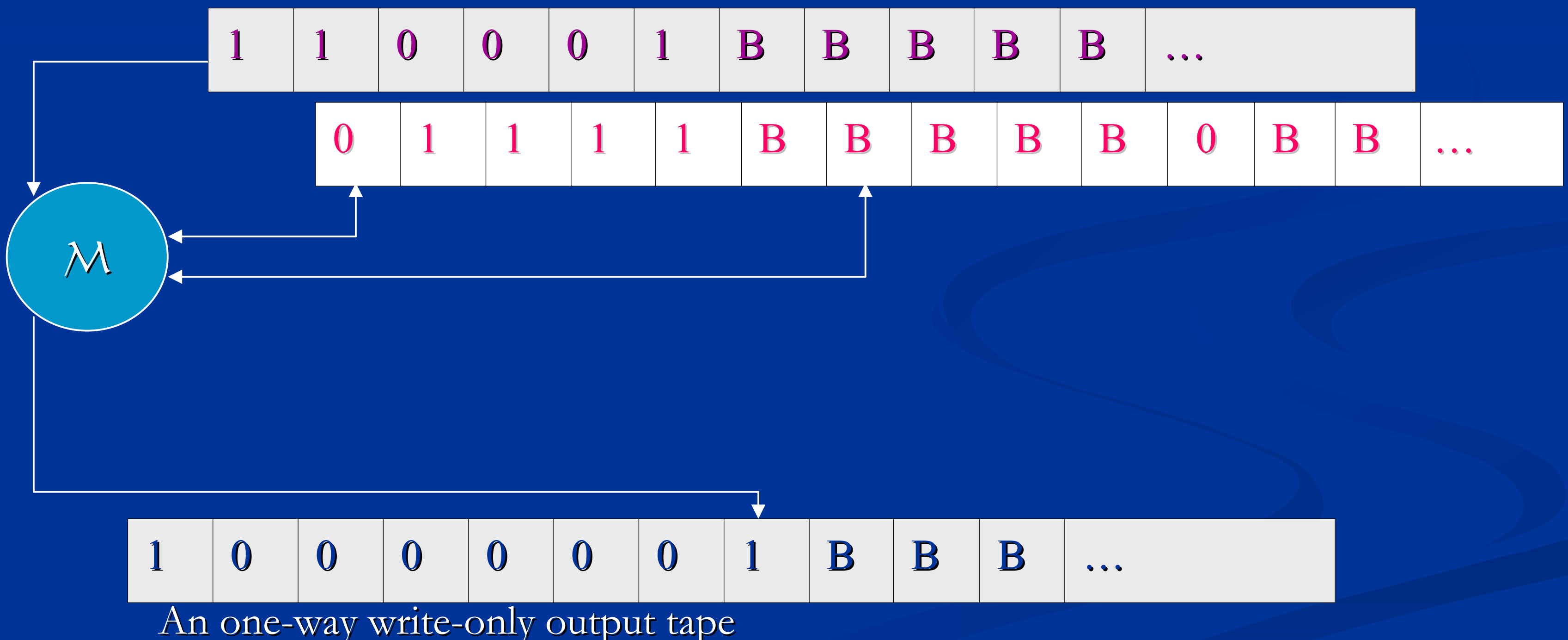
From an input tape of Type-2 machine to a work tape



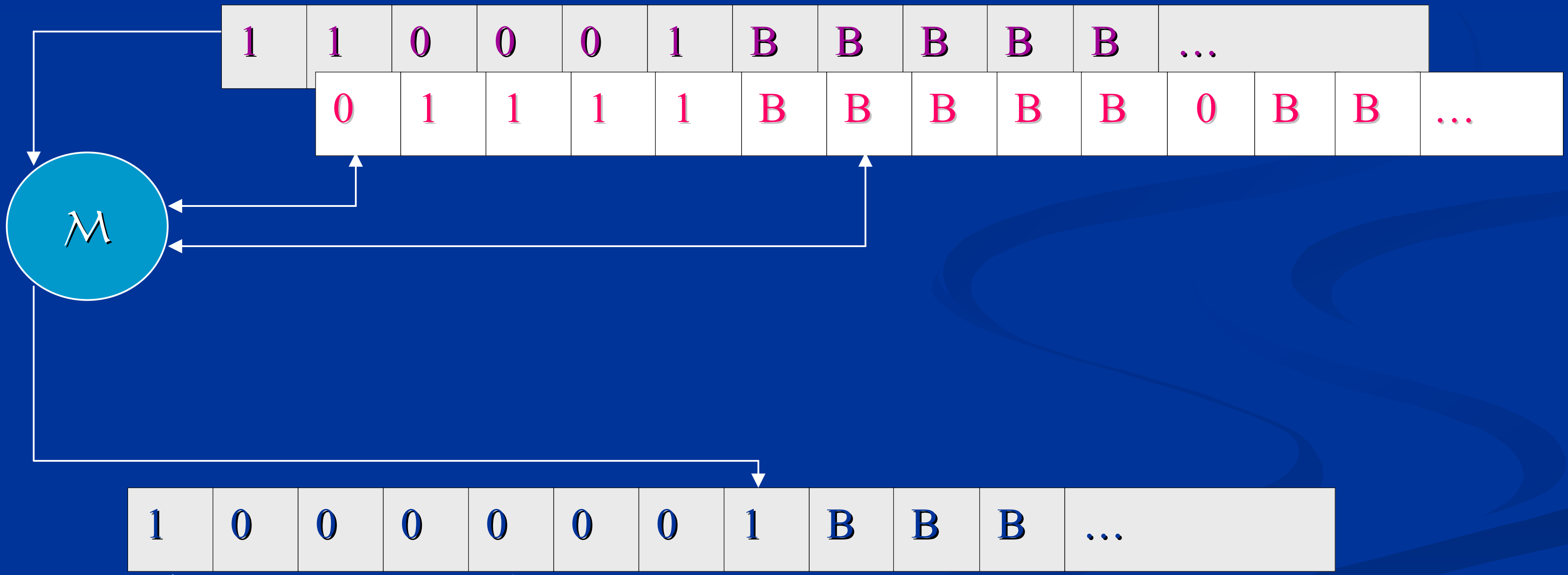
From an input tape of Type-2 machine to a work tape



From an input tape of Type-2 machine to a work tape

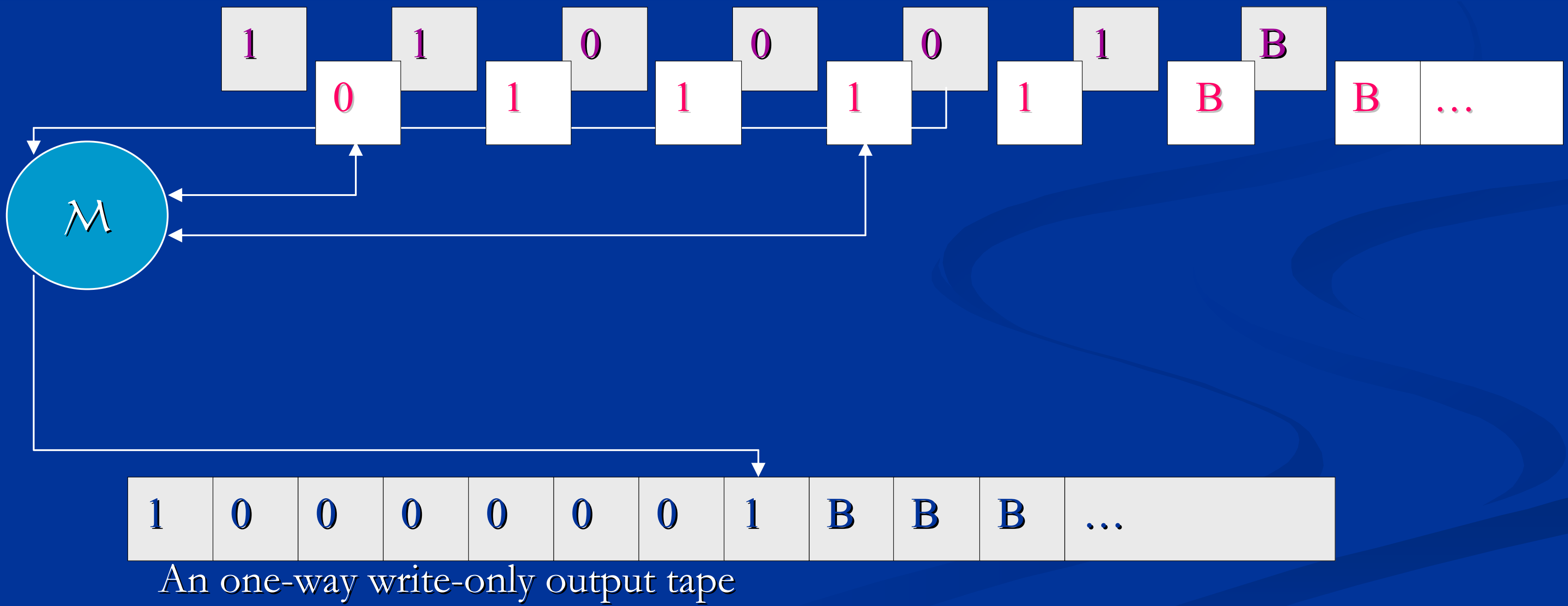


From an input tape of Type-2 machine to a work tape

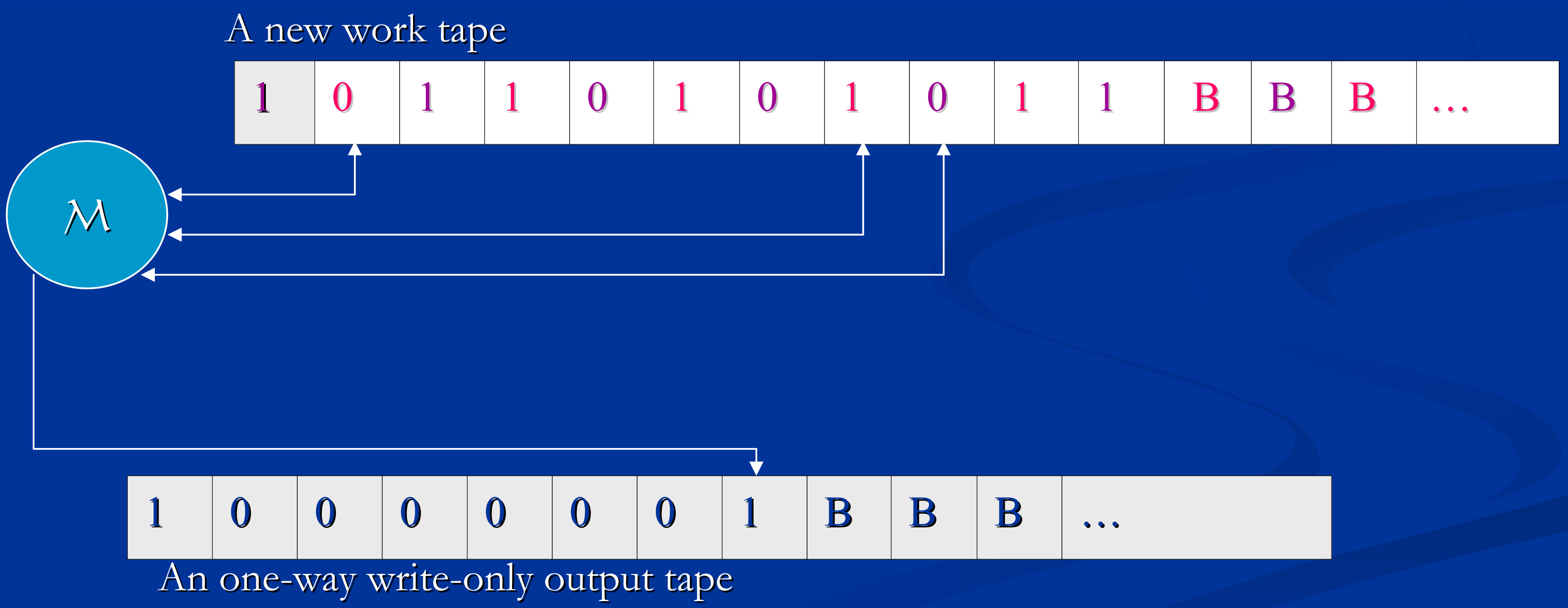


An one-way write-only output tape

From an input tape of Type-2 machine to a work tape

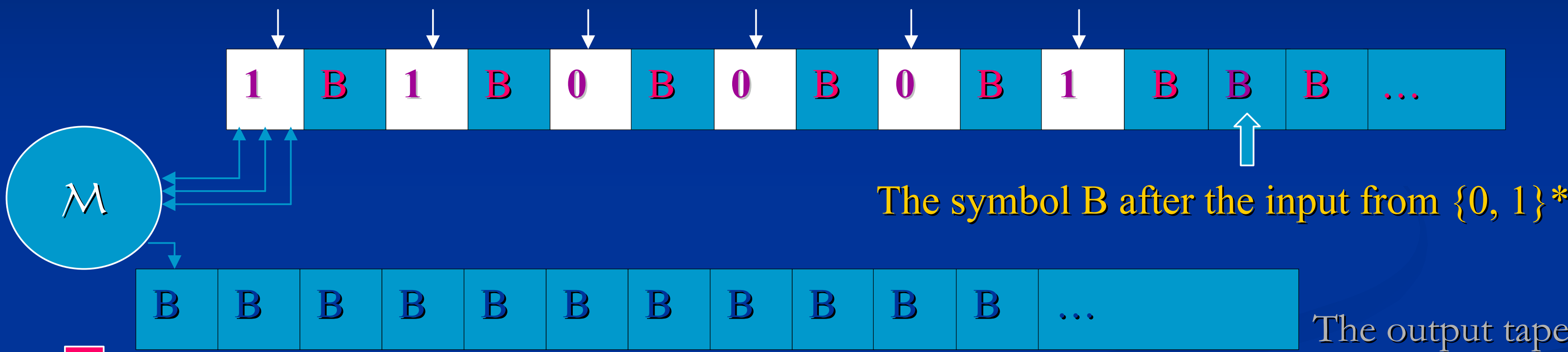


From an input tape of Type-2 machine to a work tape

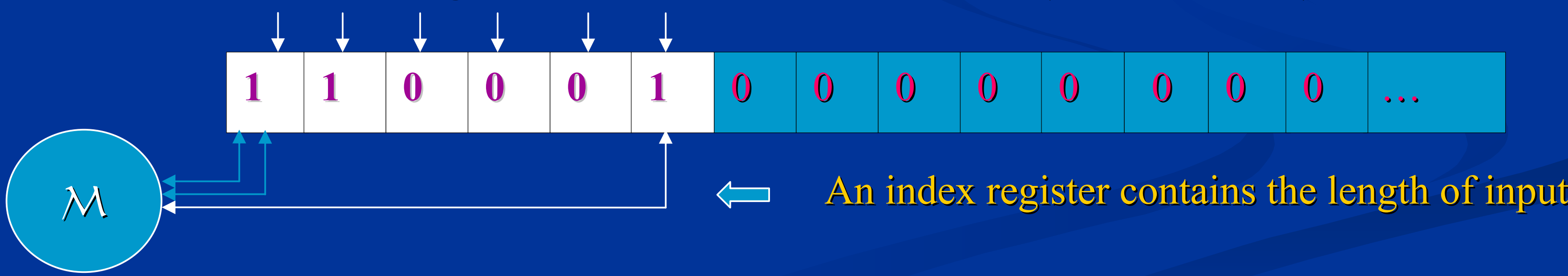


From an input tape of Type-2 machine with Y_1, $Y_0 = \{0, 1\}^*$ to a BSS input

The initial configuration for the 'Type-2' machine and for input 110001:

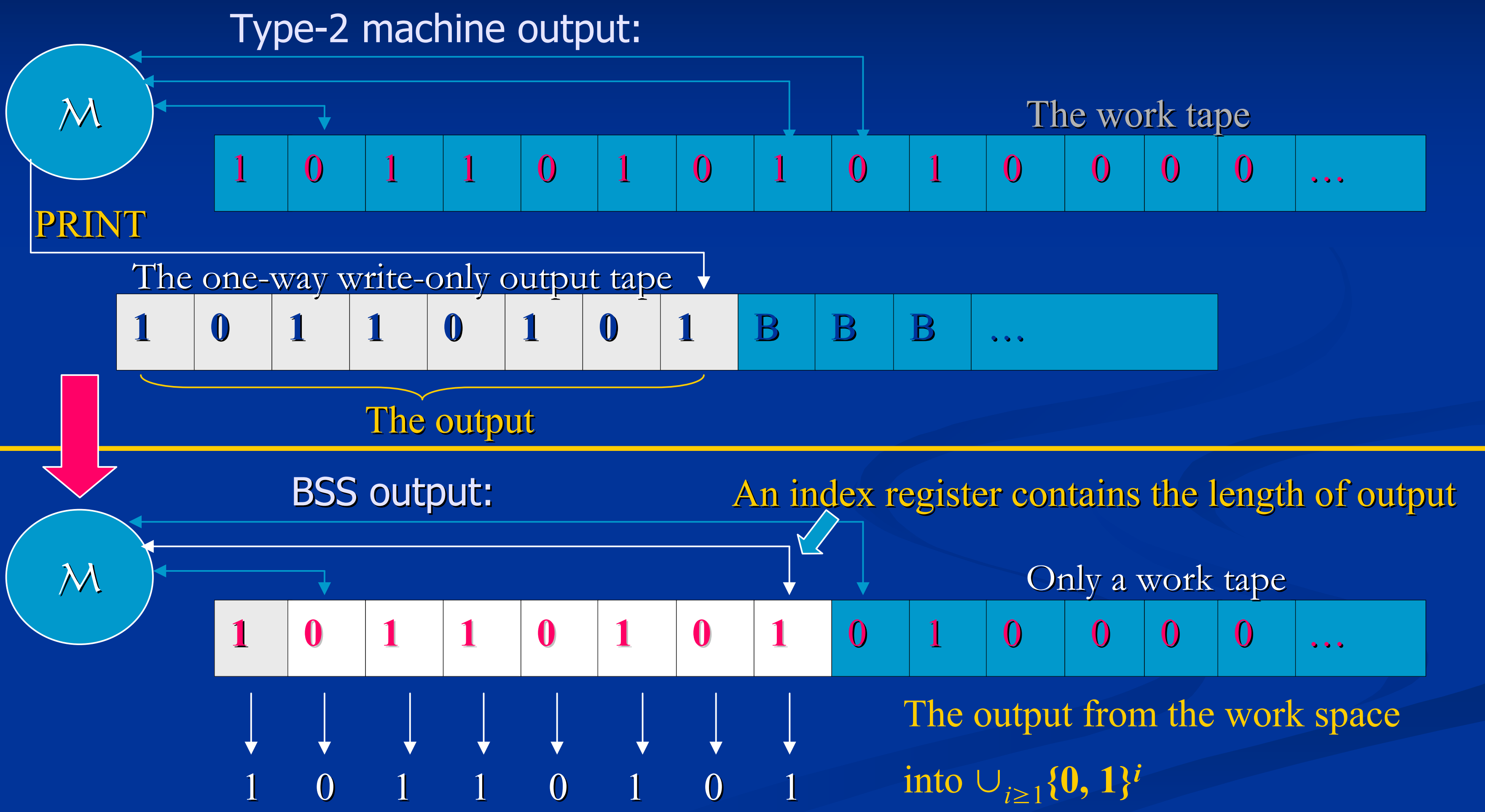


The initial configuration for the BSS input of (1, 1, 0, 0, 0, 1):

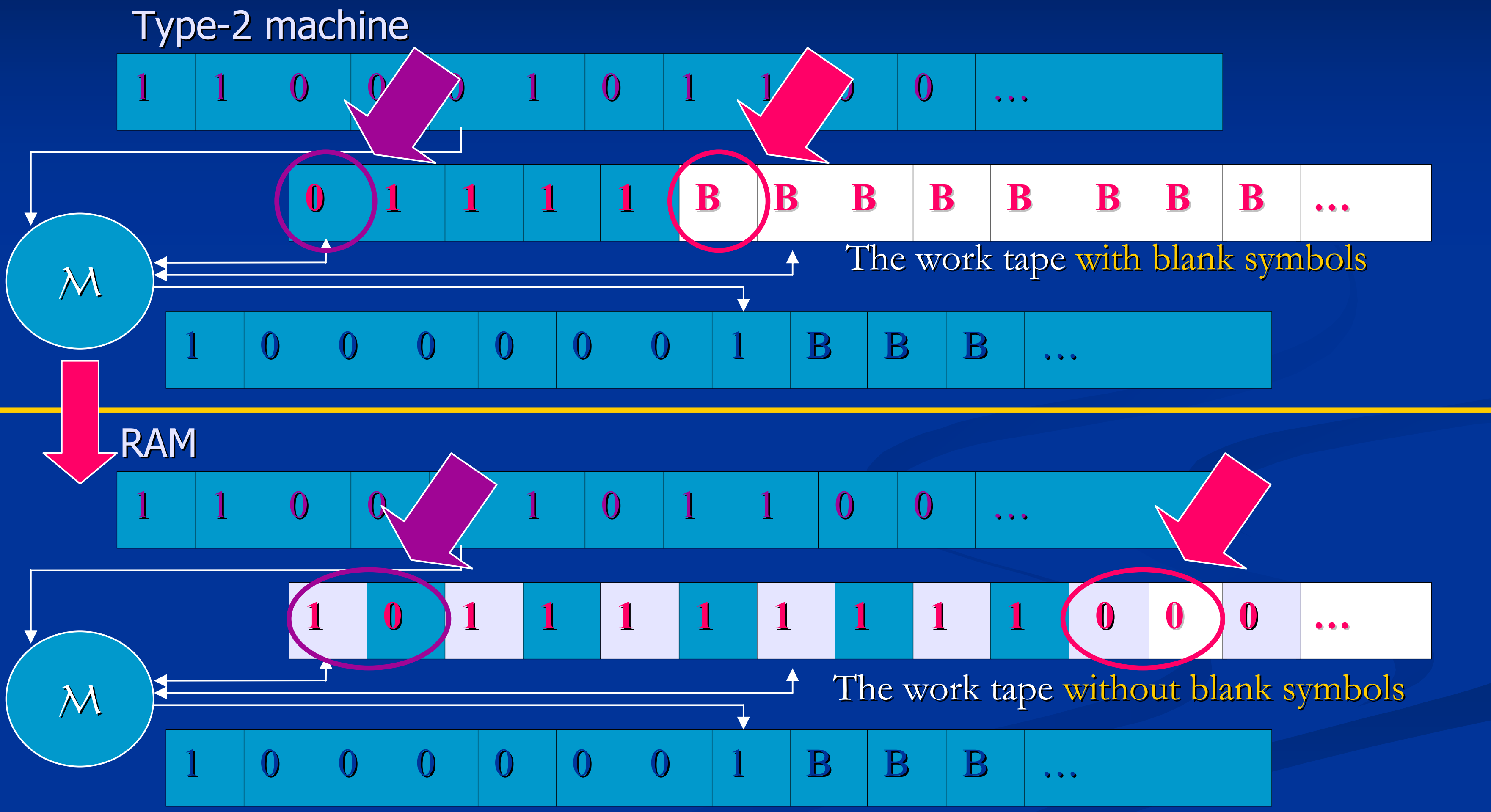


gassnerc@uni-greifswald.de

From an output tape of Type-2 machine with Y_1, $Y_0 = \{0, 1\}^*$ to a BSS output



From a Type-2 machine with $Y_1, Y_0 = \{0, 1\}^\omega$ to a $\{0, 1\}$-RAM



Consequences

Simulation of offline RAM's without input tape

Input space of the RAM	Simulation by	Simulation on space	Tuples are represented by
$\cup_{i\geq 1}\{0,1\}^i$ ⇔	Type-2 machines	$\{0, 1\}^*\{0\}^\omega$	infinite strings
$\cup_{i\geq 1}\mathbb{N}^i$ ⇔	finite dimensional register machines	$\mathbb{N}^k$ for a fixed k	Gödel numbers
⇔	Turing machines	$\{0, 1\}^*$	binary codes
$\cup_{i\geq 1}\mathbb{R}^i$ ⇒	BSS machines	$\cup_{i\geq 1}\mathbb{R}^i$	tuples over the reals

The domain of reals and consequences

- For RAM's with one-way write-only output tape we get (cp. K. Weihrauch, 2001):

⇨ (FP) Every finite portion of the output is already determined by a finite portion of the input.

- H. Friedman, R. Mansfield: 'Algorithmic Procedure' (1992), Theorem 17:

⇨ There is no 1-1 computable mapping of $\mathbb{R}^2$ into $\mathbb{R}$.

⇨ Real RAM's are not really suitable for a theory of complexity over the reals.

Semi-decidability of Halting problems over the reals

$B \subseteq \cup_{i\geq 1} \mathbb{R}^i$. B is decidable if the characteristic function is computable.

$H_{\mathbb{R}} = \{(x_1,\ldots, x_n, Code(M)) \mid (x_1,\ldots, x_n) \in \cup_{i\geq 1} \mathbb{R}^i$ & M is a machine over $\mathbb{R}$ & M halts on $\boldsymbol{x}\}$

$H_{\mathbb{R}}^{\text{spec}} = \{Code(M) \mid M$ is a machine over $\mathbb{R}$ & M halts on $Code(M)\}$

gassnerc@uni-greifswald.de

Semi-decidability of Halting problems over the reals

$B \subseteq \cup_{i\geq 1} \mathbb{R}^i$. B is decidable if the characteristic function is computable.

$$H_{\mathbb{R}} = \{(x_1,\ldots, x_n, Code(M)) \mid (x_1,\ldots, x_n) \in \cup_{i\geq 1} \mathbb{R}^i \text{ \& } M \text{ is a machine over } \mathbb{R} \text{ \& } M \text{ halts on } \boldsymbol{x}\}$$

$$H_{\mathbb{R}}^{\text{spec}} = \{Code(M) \mid M \text{ is a machine over } \mathbb{R} \text{ \& } M \text{ halts on } Code(M)\}$$

Is $H_{\mathbb{R}}$ semi-decidable? Yes, if the codes are suitable.

For BSS machines:	by one machine
For finite dim. register machines:	by a system of machines (cp. D. Scott)
	The number of used registers is unary encoded in the codes of the machines.
For infinite classical real RAM's:	by one machine

gassnerc@uni-greifswald.de

Decidability of Halting problems over the reals

$B \subseteq \cup_{i\geq 1} \mathbb{R}^i$. B is decidable if the characteristic function is computable.

$$H_{\mathbb{R}} = \{(x_1,\ldots, x_n, Code(M)) \mid (x_1,\ldots, x_n) \in \cup_{i\geq 1} \mathbb{R}^i \ \& \ M \text{ is a machine over } \mathbb{R} \ \& \ M \text{ halts on } \boldsymbol{x}\}$$

$$H_{\mathbb{R}}^{\text{spec}} = \{Code(M) \mid M \text{ is a machine over } \mathbb{R} \ \& \ M \text{ halts on } Code(M)\}$$

Can the undecidability of $H_{\mathbb{R}}^{\text{spec}}$ be shown?

For BSS machines:	by diagonalization
For finite dimensional register machines:	? (not by the usual proof)
For infinite classical real RAM's:	by diagonalization

gassnerc@uni-greifswald.de

Reducibility of semi-decidable problems over the reals to the Halting problems

$B \subseteq \cup_{i\geq 1} \mathbb{R}^i$. B is decidable if the characteristic function is computable.

$$H_{\mathbb{R}} = \{(x_1,\ldots, x_n, Code(M)) \mid (x_1,\ldots, x_n) \in \cup_{i\geq 1} \mathbb{R}^i \;\&\; M \text{ is a machine over } \mathbb{R} \;\&\; M \text{ halts on } \boldsymbol{x}\}$$

$$H_{\mathbb{R}}^{\text{spec}} = \{Code(M) \mid M \text{ is a machine over } \mathbb{R} \;\&\; M \text{ halts on } Code(M)\}$$

Can any semi-decidable problem be reduced to the corresponding $H_{\mathbb{R}}$?

For BSS machines:	yes
For finite dimensional register machines:	by a system of machines
For infinite classical real RAM's without READ:	no, e.g. $\{(x_1,\ldots, x_n) \in \mathbb{R}^\infty \mid \exists i\, (x_i \neq 0)\}$

Remarks to the non-deterministic machines

A feasible real RAM
(V. Brattka, P. Hertling)

A non-deterministic
Turing machine

A digital non-deterministic BSS machine (guesses: zeros and ones)

A non-deterministic BSS machine (guesses: real numbers)

A very abstract model of thought

⇩ means „can be simulated by"

A general BSS model over arbitrary structures

Thank you very much!
Christine Gaßner

I also thank my students Paul Grieger, Franz Huwald, and Isabel Desire Schwende.

A finite dimensional register machine

⇓

A classical RAM | A Type-2 machine with $I, O \subseteq \{0, 1\}^*$

⇓

A BSS machine

⇓

A system (family) of RAM's for k-dimensional inputs $(k \in \mathbb{N}^+)$

gassnerc@uni-greifswald.de