



Architetture degli Elaboratori e delle Reti I

10

Laboratorio – linea 2 (G-Z)

CPU

- **Docente teoria:** prof. Federico Pedersini
(<https://homes.di.unimi.it/pedersini/AE-INF.html>)
- **Docente laboratorio:** Matteo Re
(<https://homes.di.unimi.it/re/arch1-lab-2015-2016.html>)
- **Sito laboratorio turno 2:**
(<http://basilico.di.unimi.it/doku.php?id=pub:arch1-lab-2015-2016>)



Simulazione CPU – 1

(componenti fondamentali)

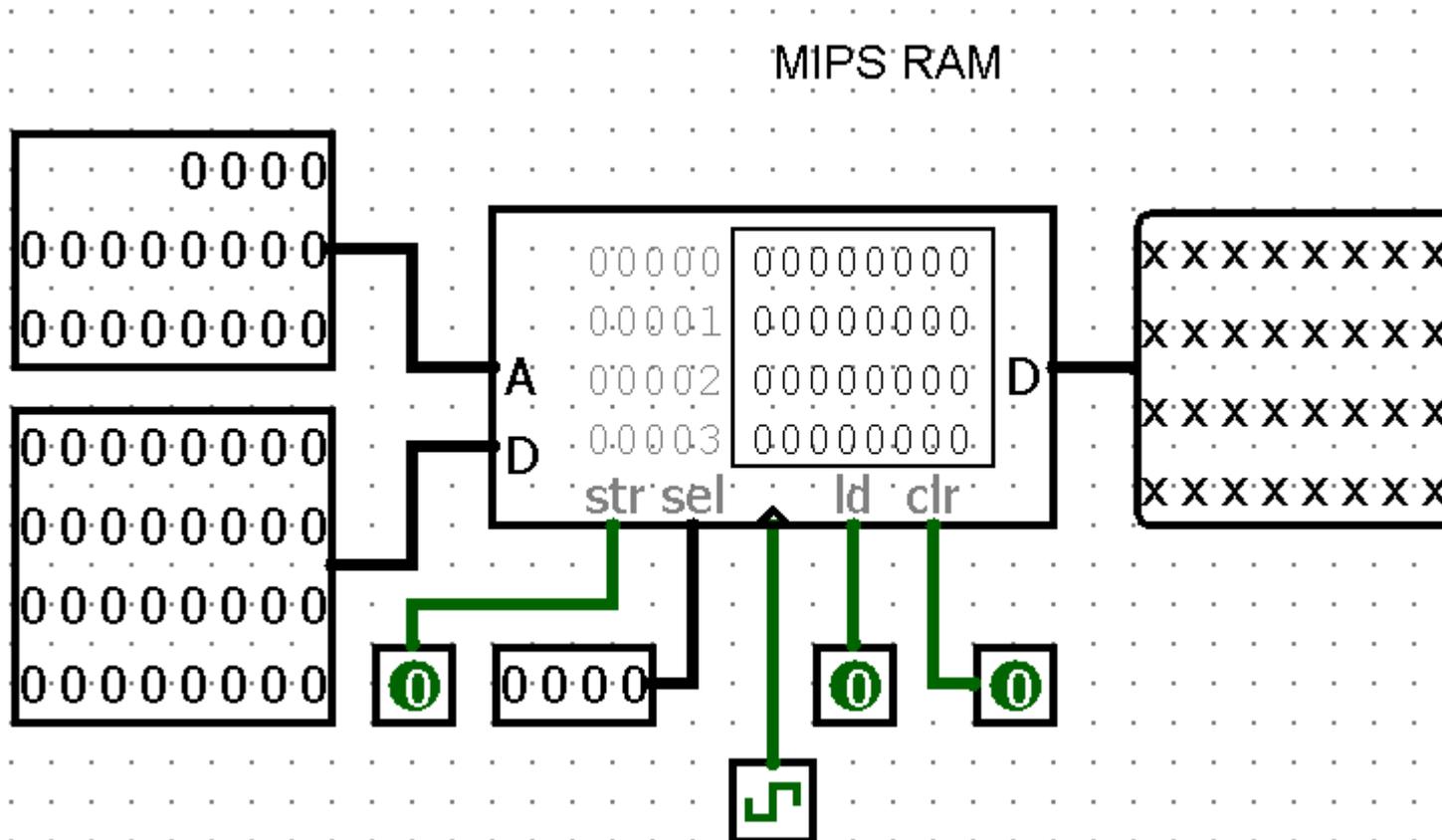
Register File

ALU

Memoria



Memoria





Architetture degli Elaboratori e delle Reti I

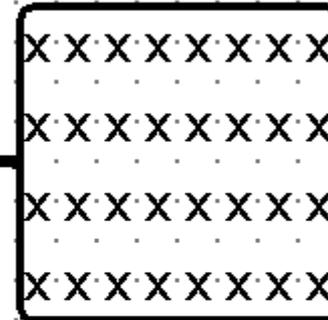
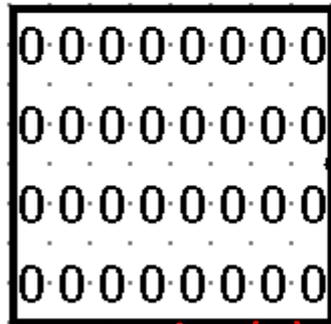
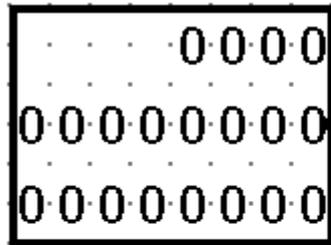
10

Laboratorio – linea 2 (G-Z)

Memoria

MIPS RAM

Address selector (A)



Input value (D)

bytes mask
(access
enable if 1)

Asynchronous reset

Write input at
selected address

Read value at selected
address



Architetture degli Elaboratori e delle Reti I

10

Laboratorio – linea 2 (G-Z)

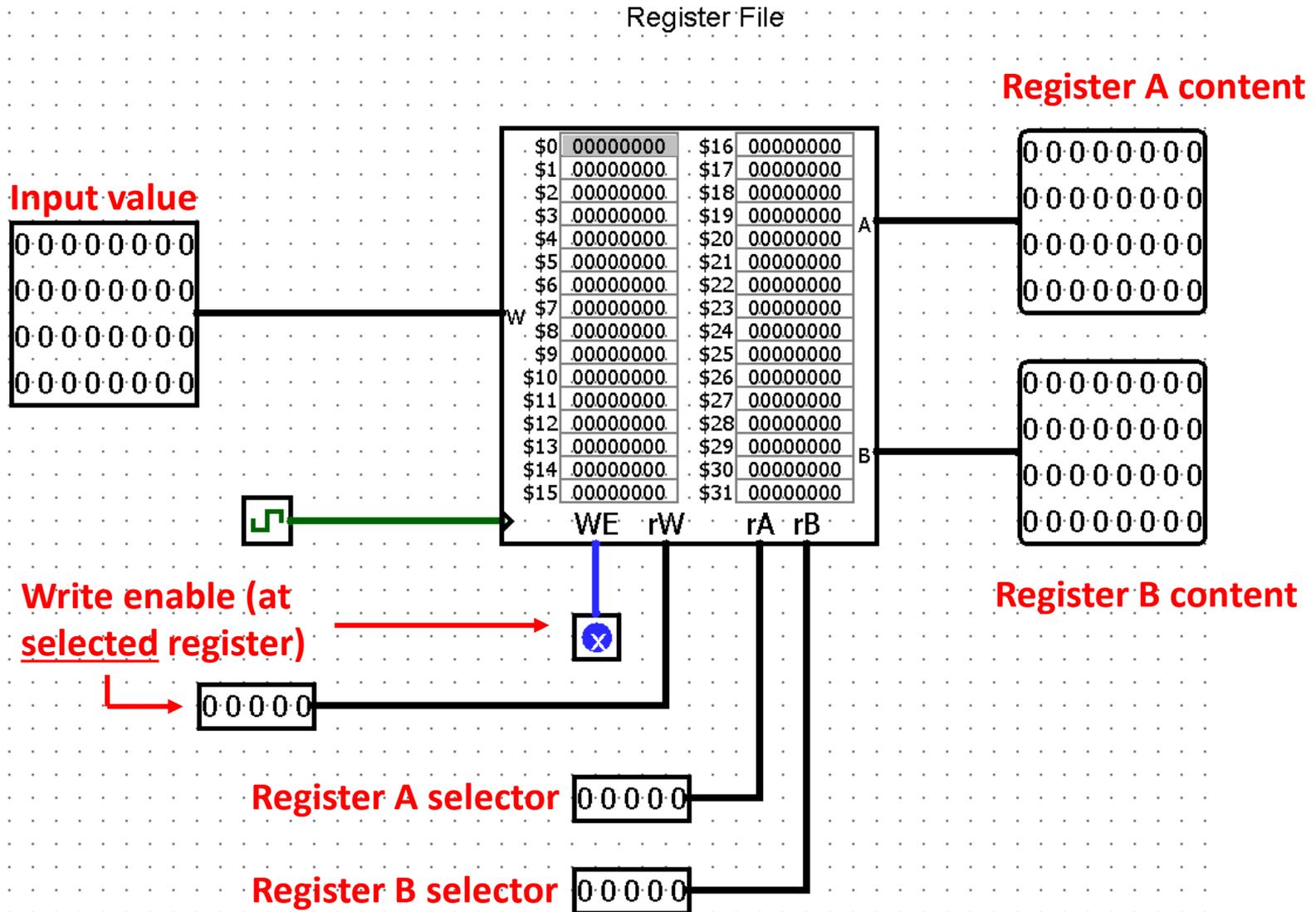
Register File



Architetture degli Elaboratori e delle Reti I

10

Laboratorio – linea 2 (G-Z)



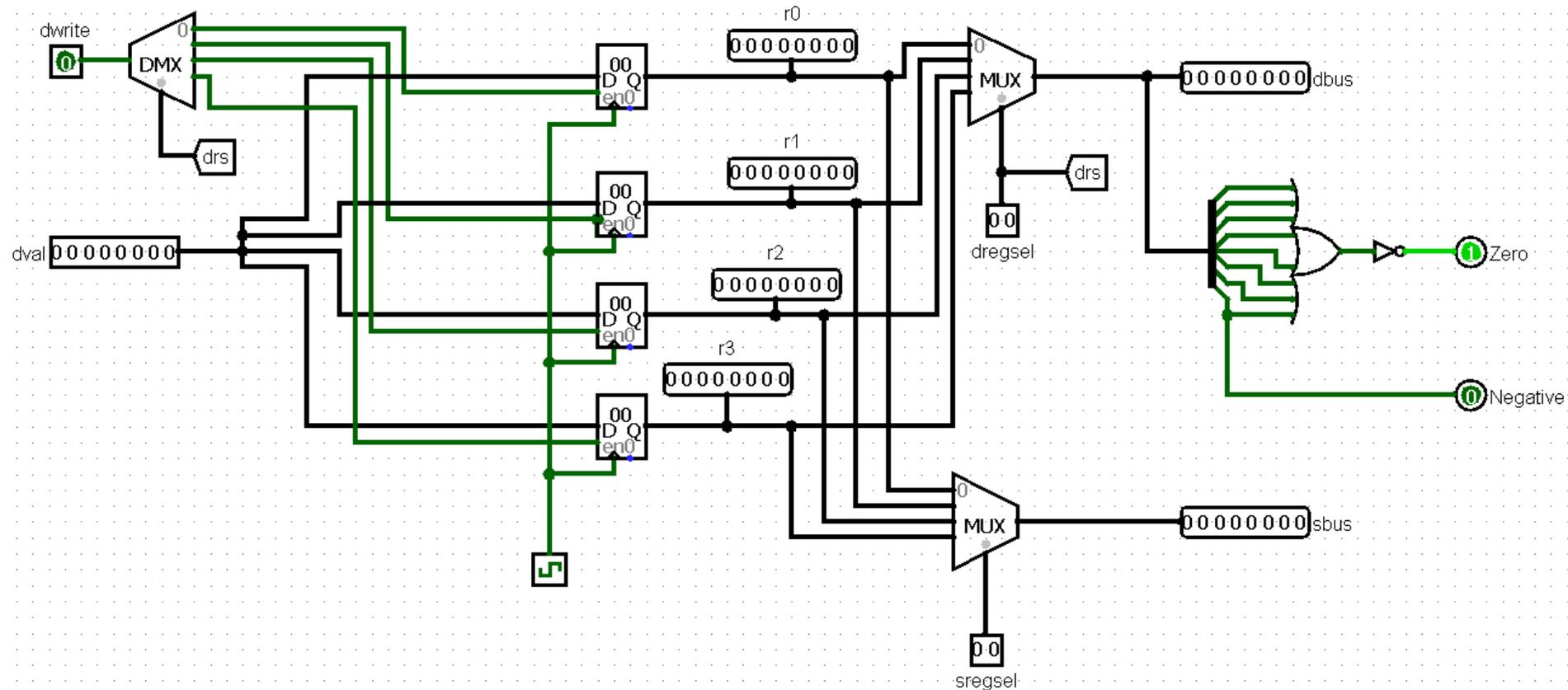


Architetture degli Elaboratori e delle Reti I

10

Laboratorio – linea 2 (G-Z)

Register File



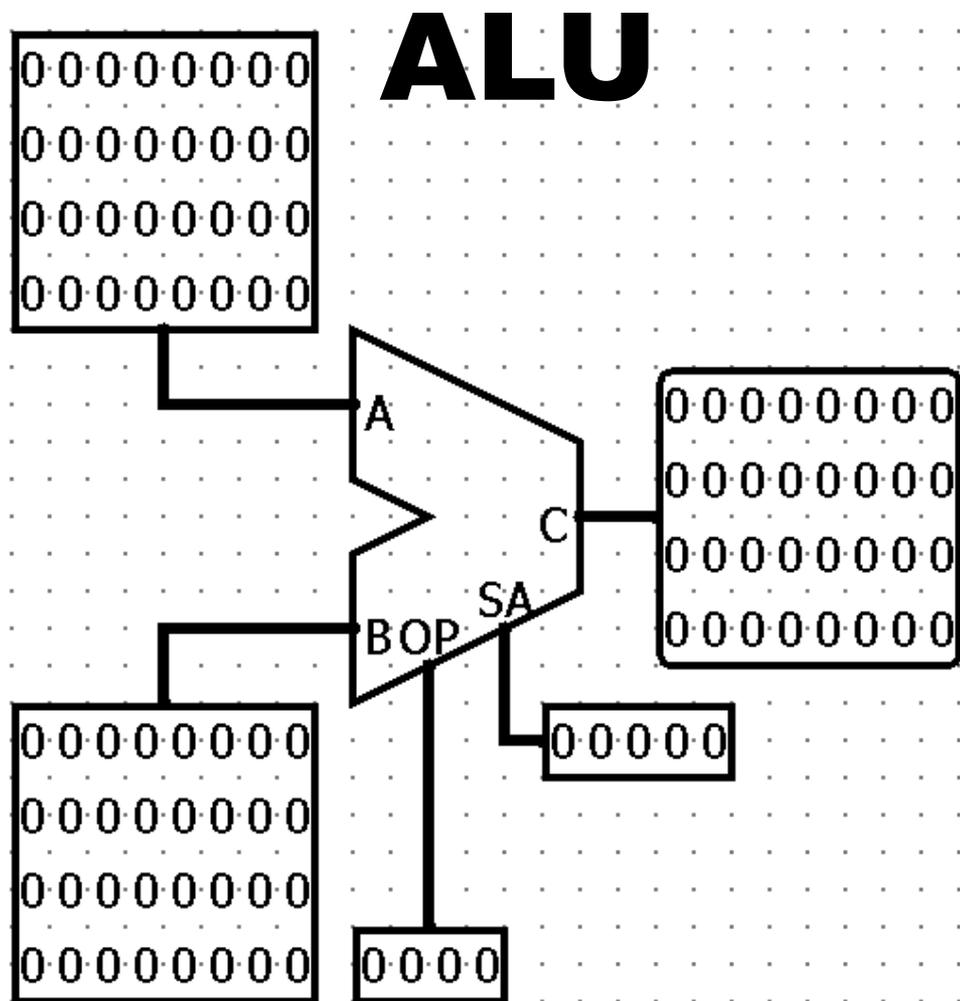
Implementazione mediante componenti libreria standard Logisim



Architetture degli Elaboratori e delle Reti I

10

Laboratorio – linea 2 (G-Z)



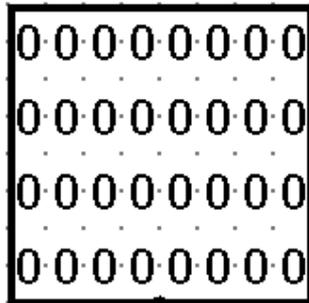


Architetture degli Elaboratori e delle Reti I

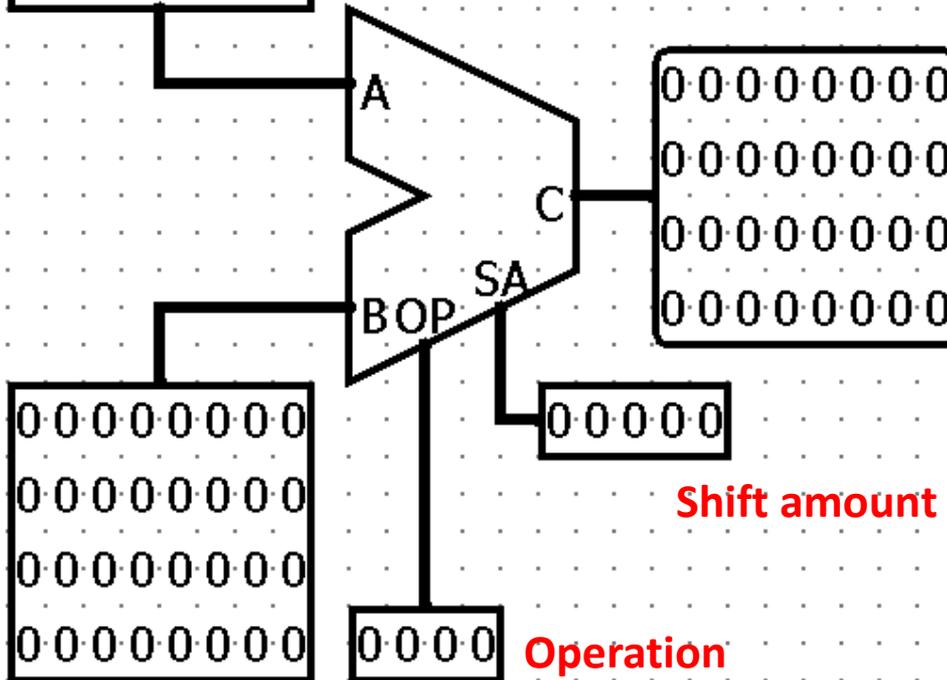
10

Laboratorio – linea 2 (G-Z)

Input value A



ALU



Output value C

Shift amount

Input value B

Operation



Architetture degli Elaboratori e delle Reti I

10

Laboratorio – linea 2 (G-Z)

MIPS ALU. Computes a result as follows:

Op	name	C
----	-----	-----
000x	shift left	$C = B \ll Sa;$
001x	add	$C = A + B$
0100	shift right logical	$C = B \ggg Sa$
0101	shift right arithmetic	$C = B \gg Sa$
011x	subtract	$C = A - B$
1000	and	$C = A \& B$
1010	or	$C = A B$
1100	xor	$C = A \wedge B$
1110	nor	$C = \sim(A B)$
1001	eq	$C = (A == B) ? 1 : 0$
1011	ne	$C = (A != B) ? 1 : 0$
1101	gtz	$C = (A > 0) ? 1 : 0$
1111	lez	$C = (A \leq 0) ? 1 : 0$



Esercizio teoria:

Si traduca il seguente frammento di codice assembly MIPS in linguaggio macchina in formato esadecimale calcolando prima i valori esadecimali Loc1 e Loc2 che permettono di saltare esattamente all'indirizzo indicato nel commento corrispondente.

```
0xA00:      bne    $10,    $11,    Loc1    #salta a: 0x19B8
           j      Loc2      #salta a: 0x01234560
```