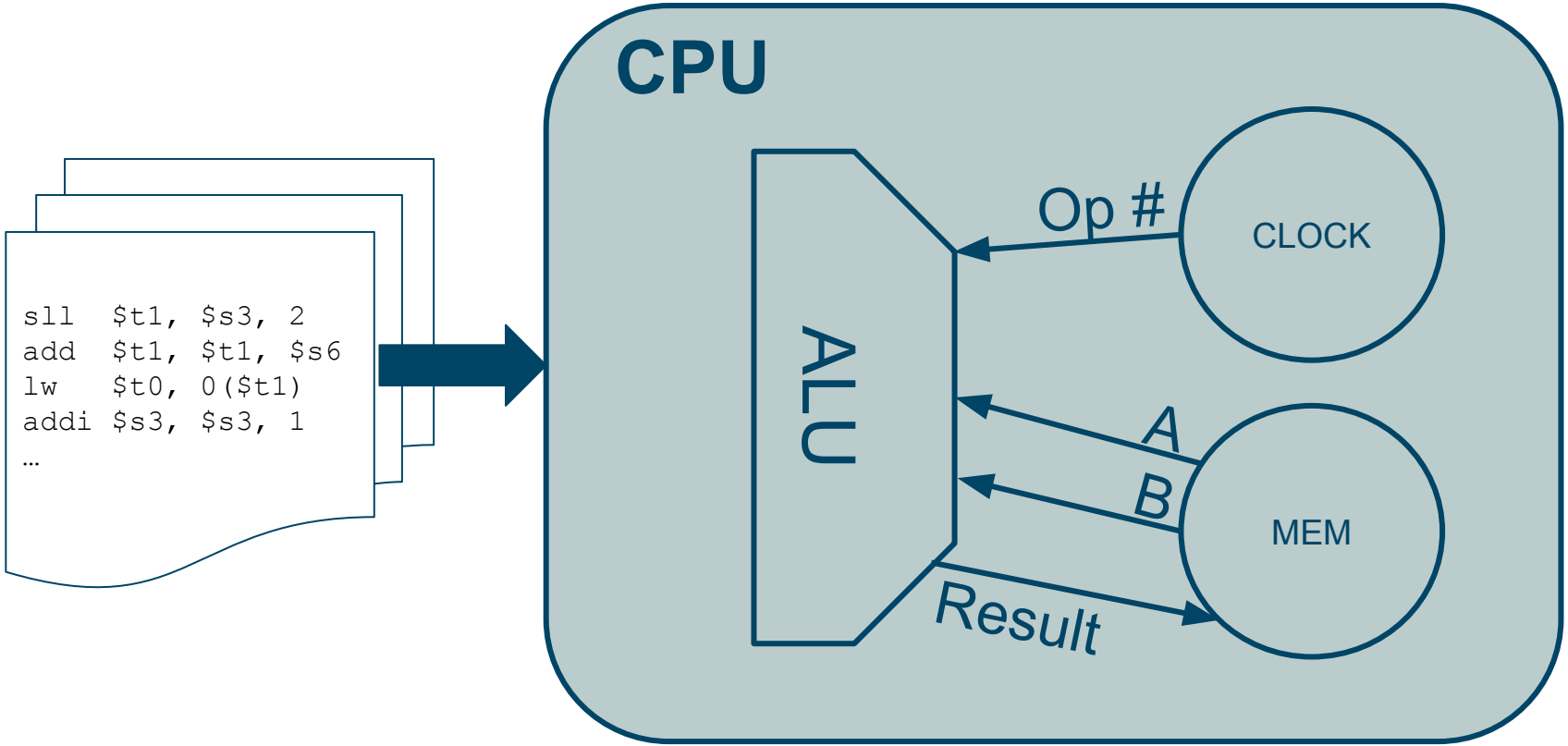
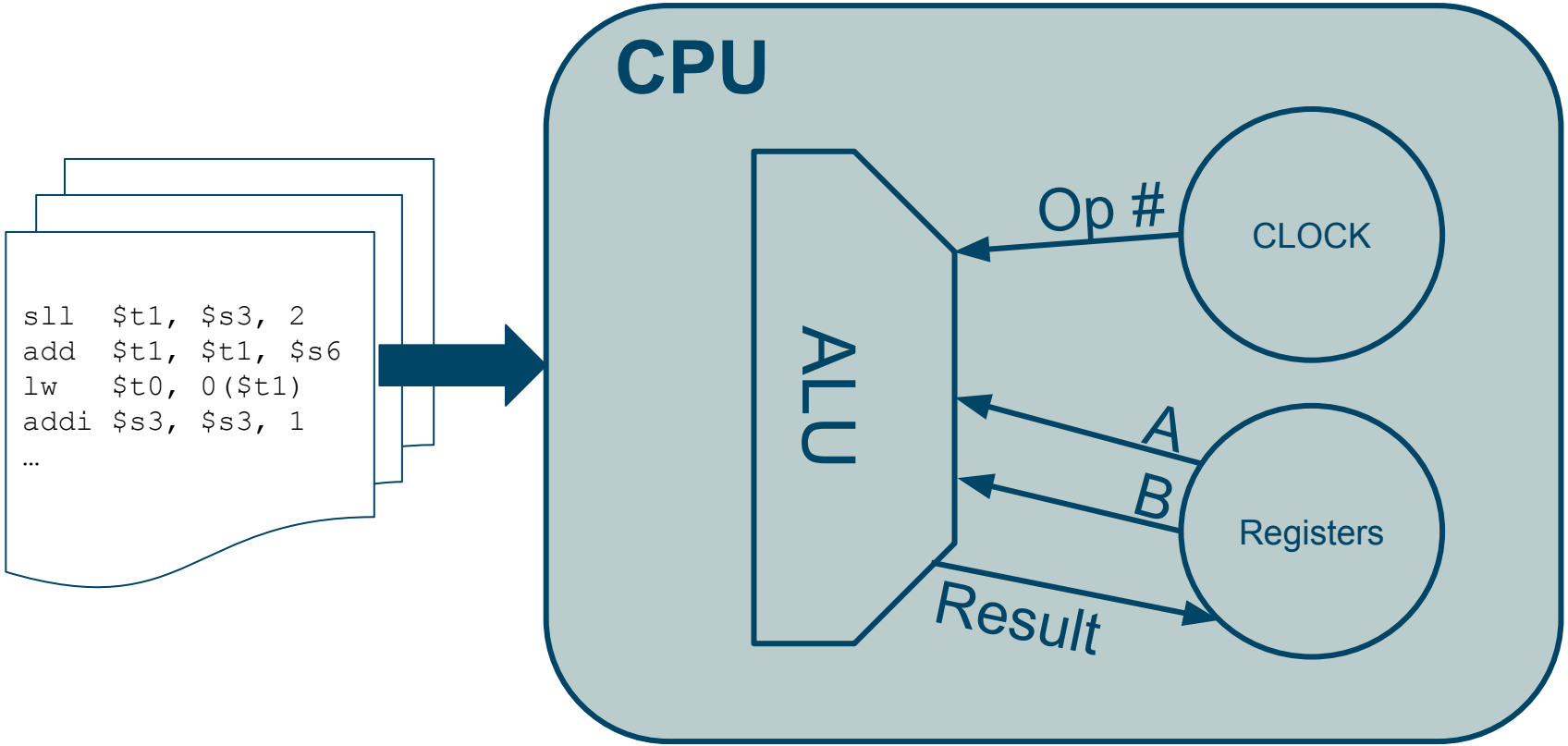
Computer Architecture

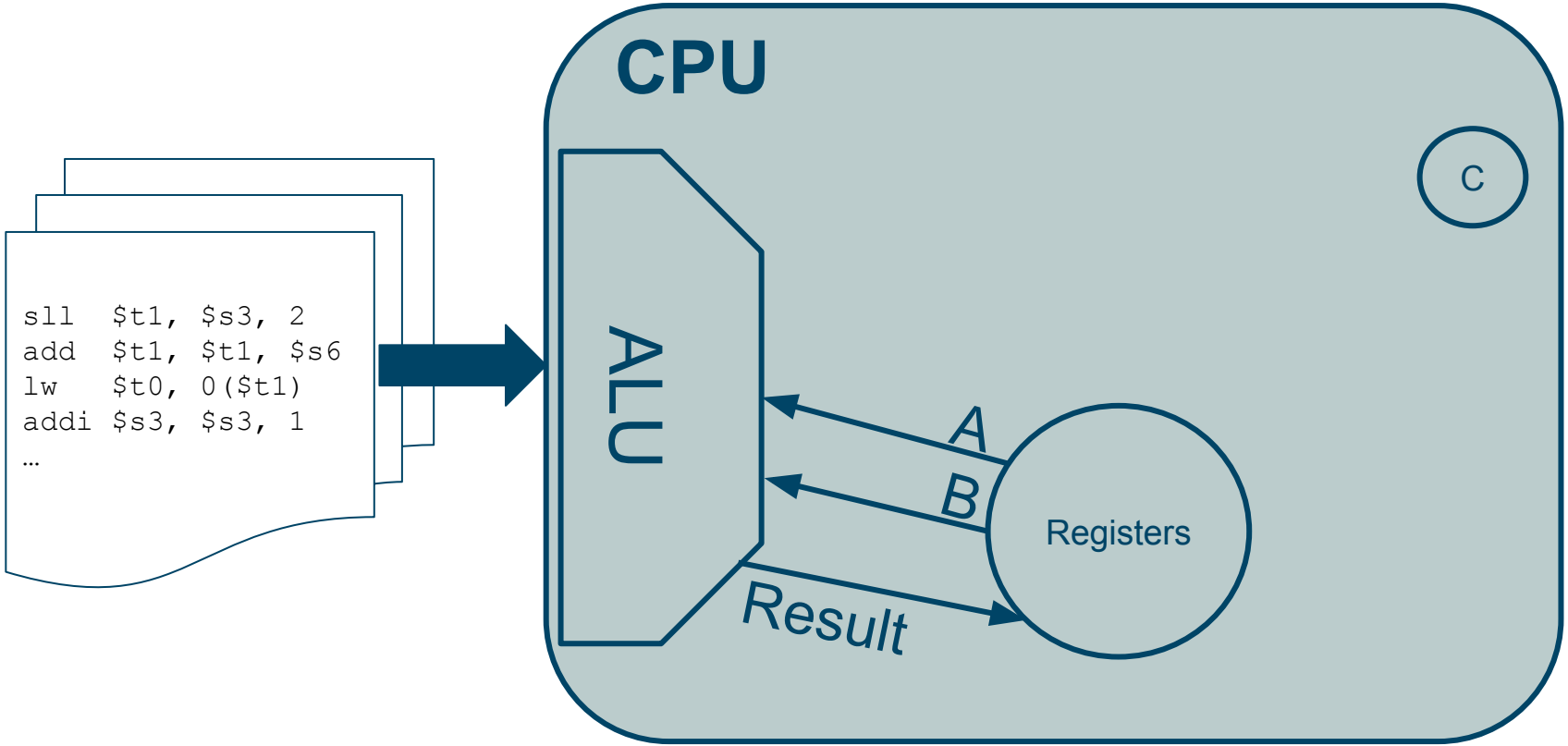
Brent van Bladel

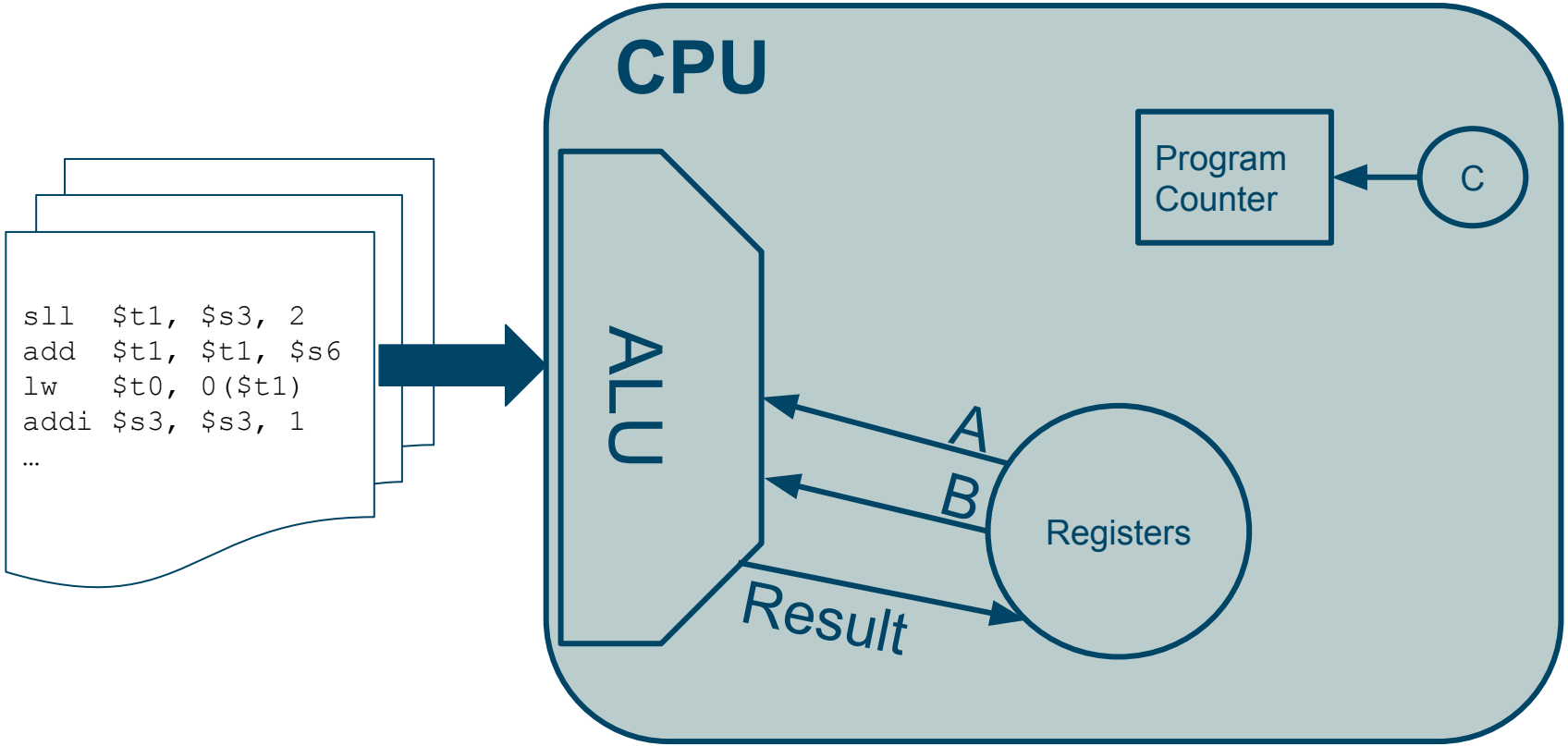
Stephen Pauwels

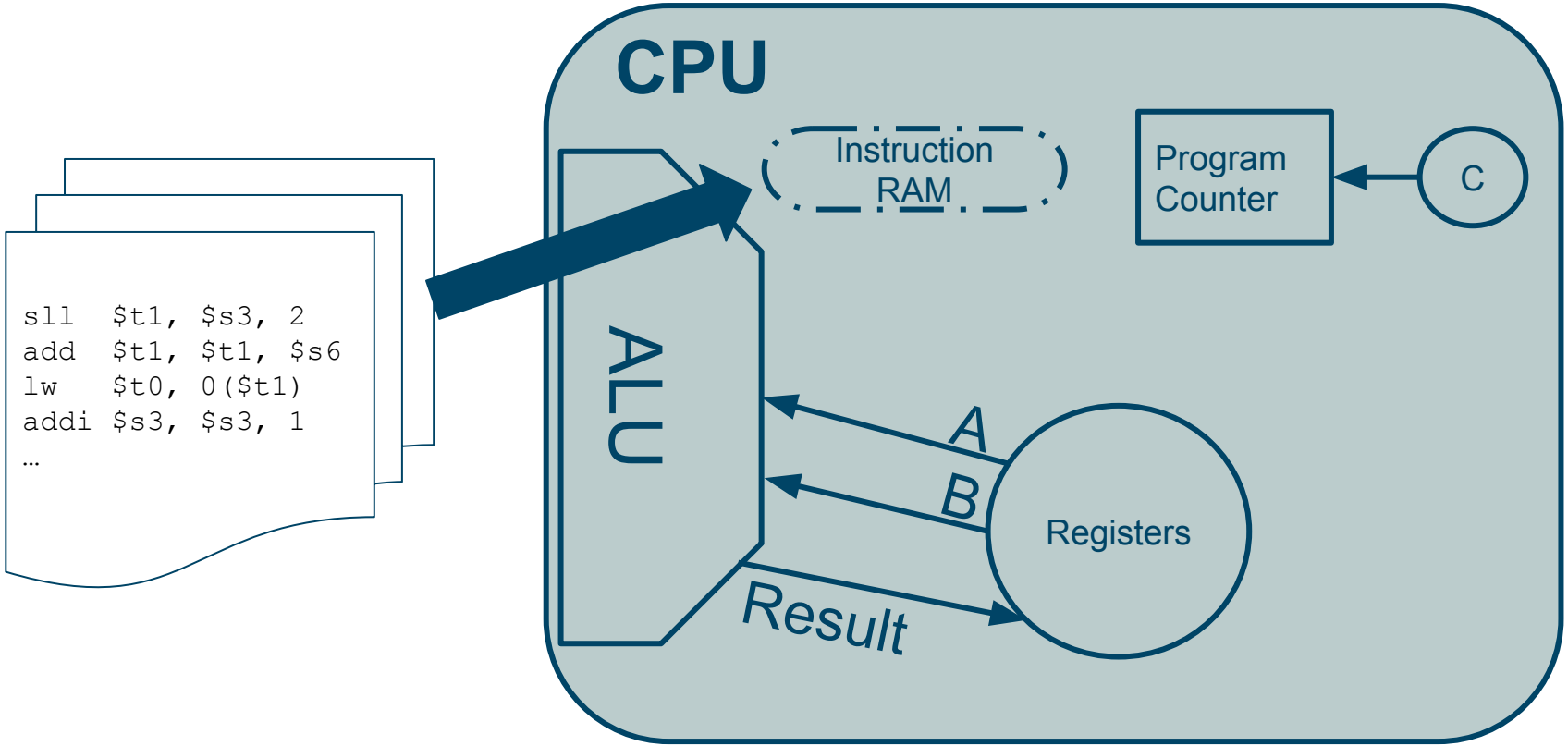
Universiteit Antwerpen

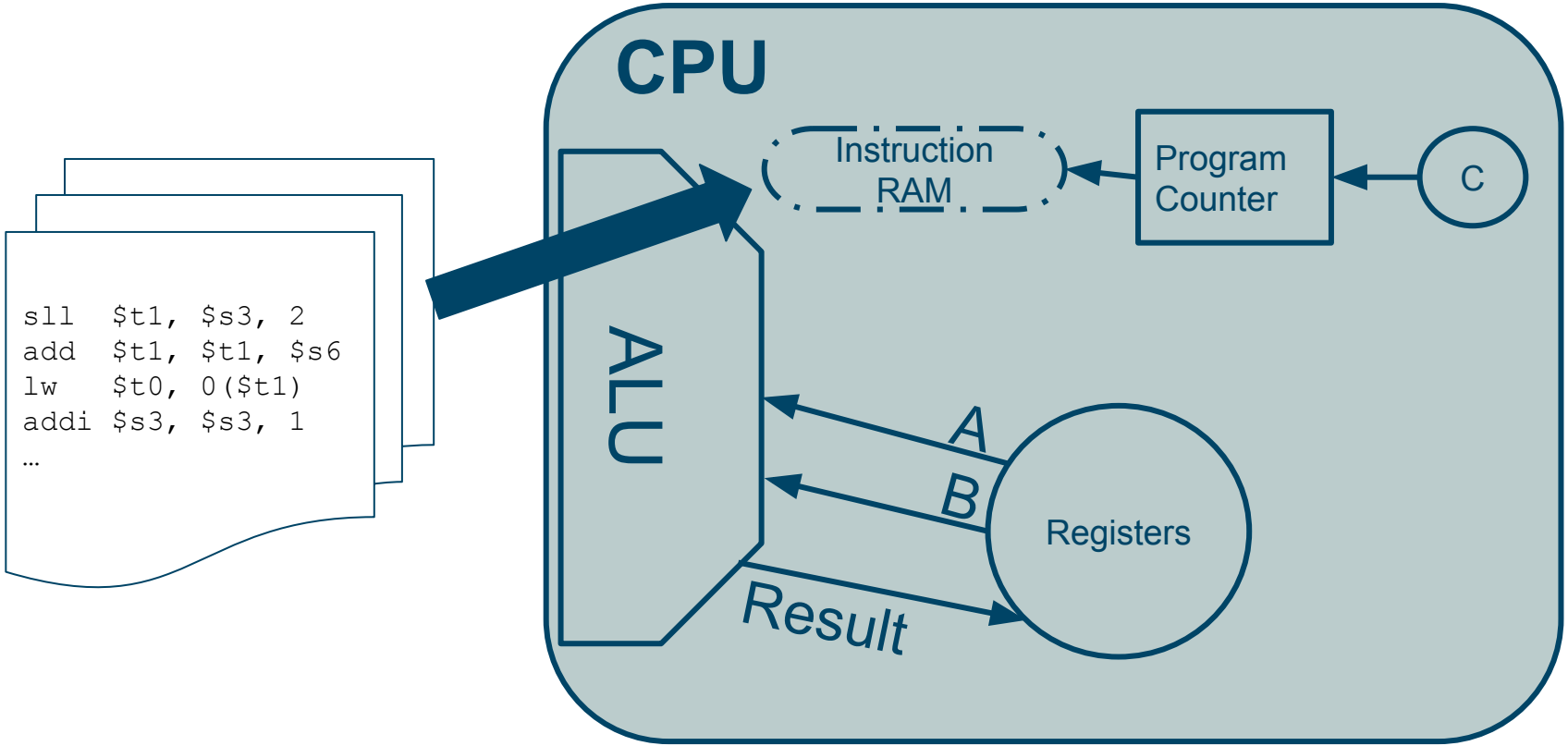

sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…
CPU
ALU
Op #
CLOCK
A
B
MEM
Result

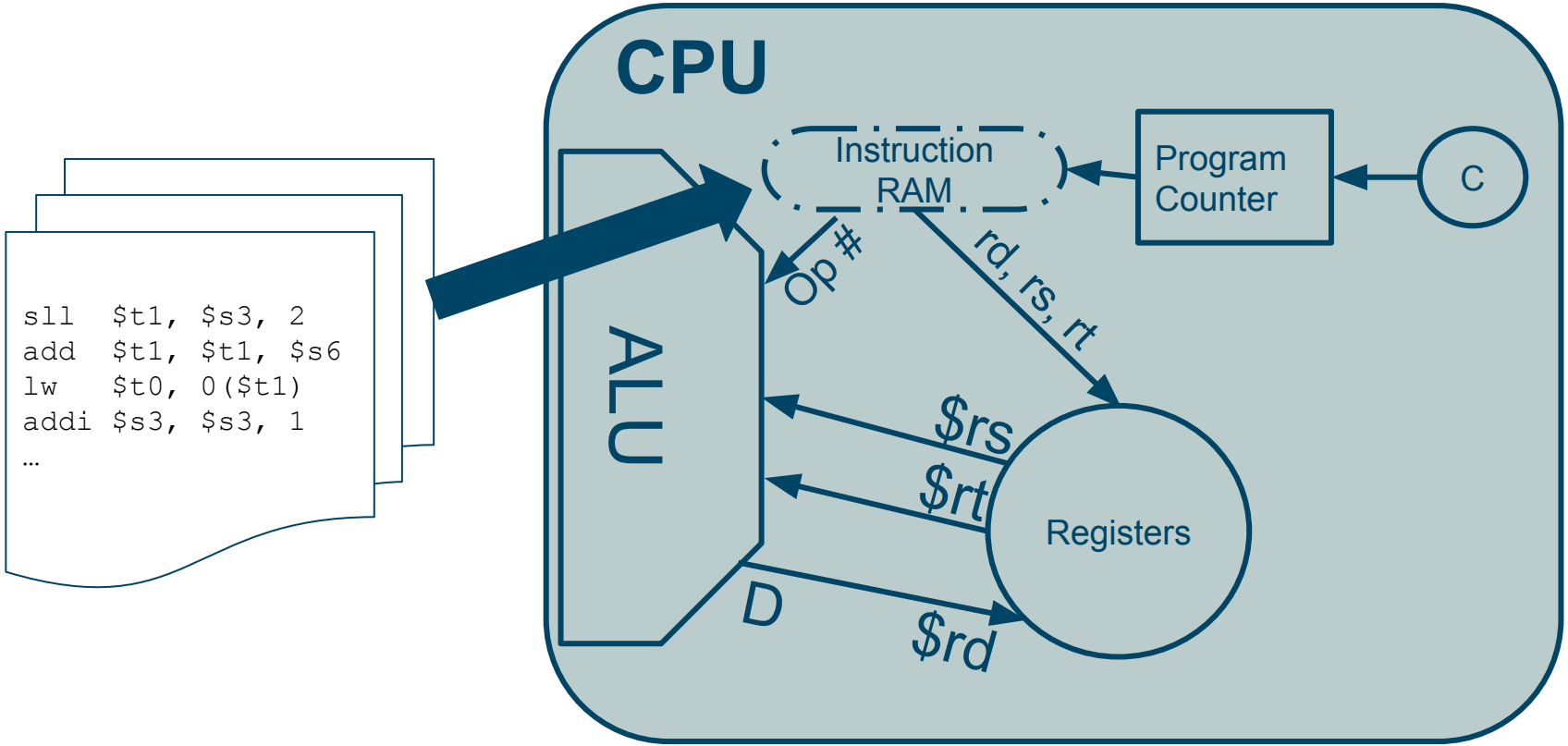

sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…
CPU
ALU
Op #
CLOCK
A
B
Registers
Result


CPU
C
ALU
A
B
Result
Registers
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…


CPU
Program Counter
C
ALU
A
B
Registers
Result
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…


CPU
Instruction RAM
Program Counter
C
ALU
A
B
Registers
Result
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…


CPU
Instruction RAM
Program Counter
C
ALU
A
B
Registers
Result
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…


CPU
Instruction RAM
Program Counter
C
Op #
rd, rs, rt
ALU
$rs
$rt
Registers
D
$rd
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…

Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

Register Instructions

Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

Binary Operations

Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

Unary Operations

Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

Unary Operations

Instruction Set

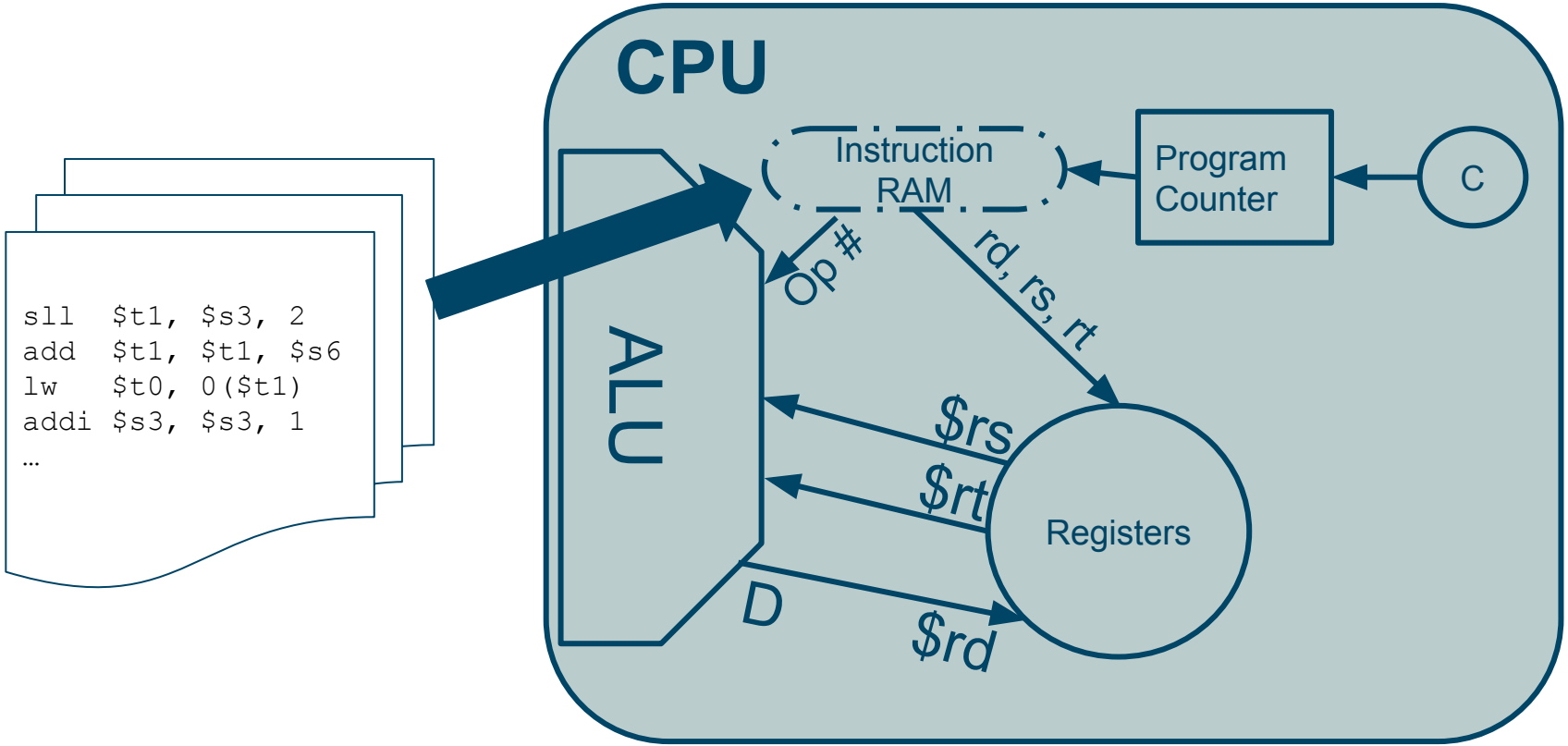
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

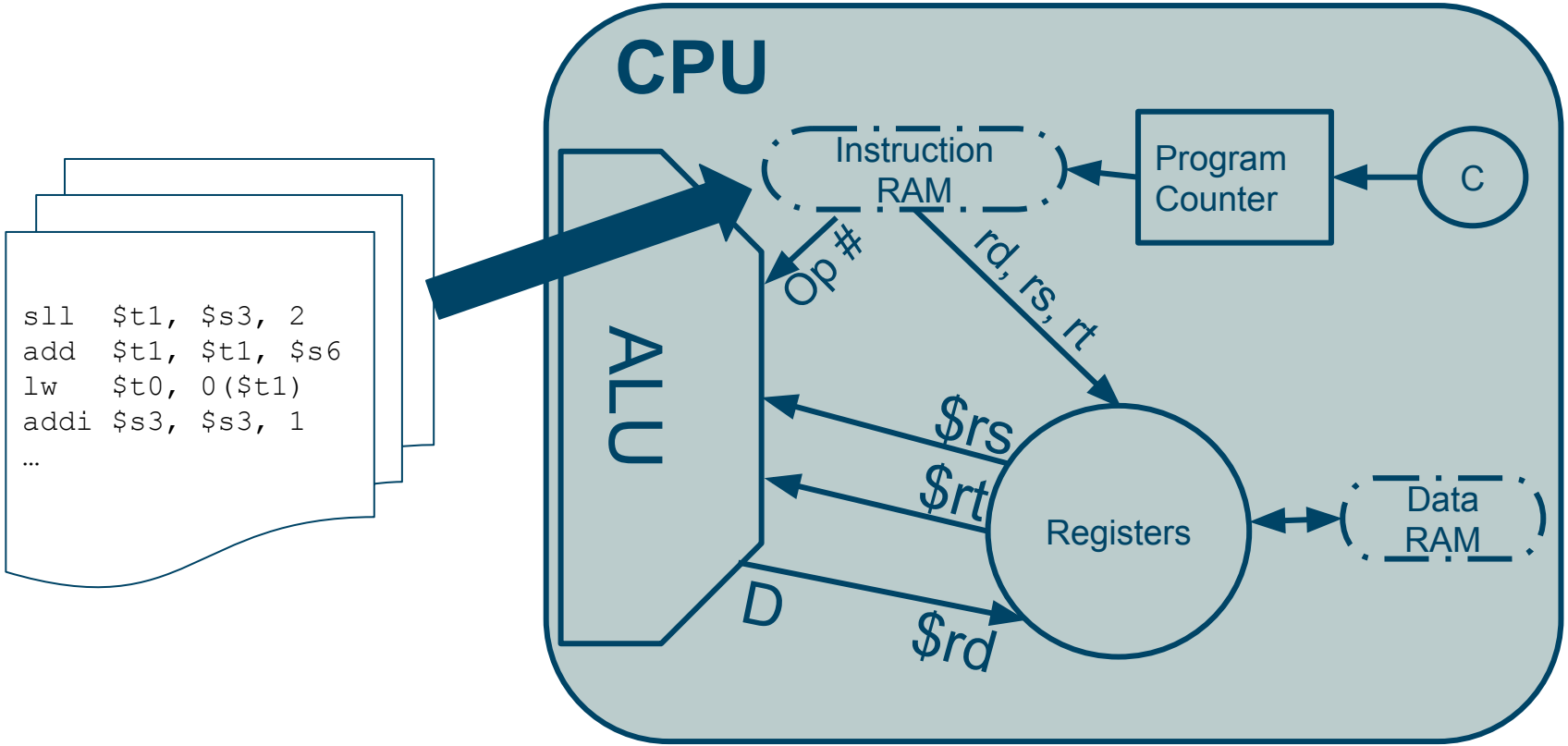
Zero Instruction

Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	name	instruction	description
000			rd			000			000			0000				zero[1]	`zero rd`	$rd := 0
001			rd			rs			000			0001				not[1]	`not rd rs`	$rd := !$rs
001			rd			rs			000			1010				inv[1]	`inv rd rs`	$rd := -$rs
001			rd			rs			000			1011				sll[1]	`sll rd rs`	$rd := $rs << 2
001			rd			rs			000			1100				srl[1]	`srl rd rs`	$rd := $rs >> 2
001			rd			rs			000			1101				sla[1]	`sla rd rs`	$rd := $rs * 2
001			rd			rs			000			1110				sra[1,2]	`sra rd rs`	$rd := $rs / 2
001			rd			rs			000			1111				cp[1]	`cp rd rs`	$rd := $rs
010			rd			rs			rt			0010				and[1]	`and rd rs rt`	$rd := $rs & $rt
010			rd			rs			rt			0011				or[1]	`or rd rs rt`	$rd := $rs | $rt
010			rd			rs			rt			0100				add[1]	`add rd rs rt`	$rd := $rs + $rt
010			rd			rs			rt			0101				sub[1]	`sub rd rs rt`	$rd := $rs - $rt
010			rd			rs			rt			0110				lt[1]	`lt rd rs rt`	$rd := $rs < $rt ? 1 : 0
010			rd			rs			rt			0111				gt[1]	`gt rd rs rt`	$rd := $rs > $rt ? 1 : 0
010			rd			rs			rt			1000				eq[1]	`eq rd rs rt`	$rd := $rs = $rt ? 1 : 0
010			rd			rs			rt			1001				neq[1]	`neq rd rs rt`	$rd := $rs != $rt ? 1 : 0
011			rd			rs			imm (signed)						0	lw	`lw rd rs imm`	$rd := MEM[$rs+imm]
011			rd			rs			imm (signed)						1	sw	`sw rd rs imm`	MEM[$rs+imm] := $rd

Memory Instructions


CPU
Instruction RAM
Program Counter
C
Op #
rd, rs, rt
ALU
$rs
$rt
Registers
D
$rd
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…


CPU
Instruction RAM
Program Counter
C
Op #
rd, rs, rt
ALU
$rs
$rt
Registers
Data RAM
D
$rd
sll $t1, $s3, 2
add $t1, $t1, $s6
lw $t0, 0($t1)
addi $s3, $s3, 1
…