CISC Design

Hardware Flowchart

Virendra Singh

Computer Architecture and Dependable Systems Lab

Department of Electrical Engineering

Indian Institute of Technology Bombay

http://www.ee.iitb.ac.in/~viren/

E-mail: viren@ee.iitb.ac.in

EE-309: Microprocessors

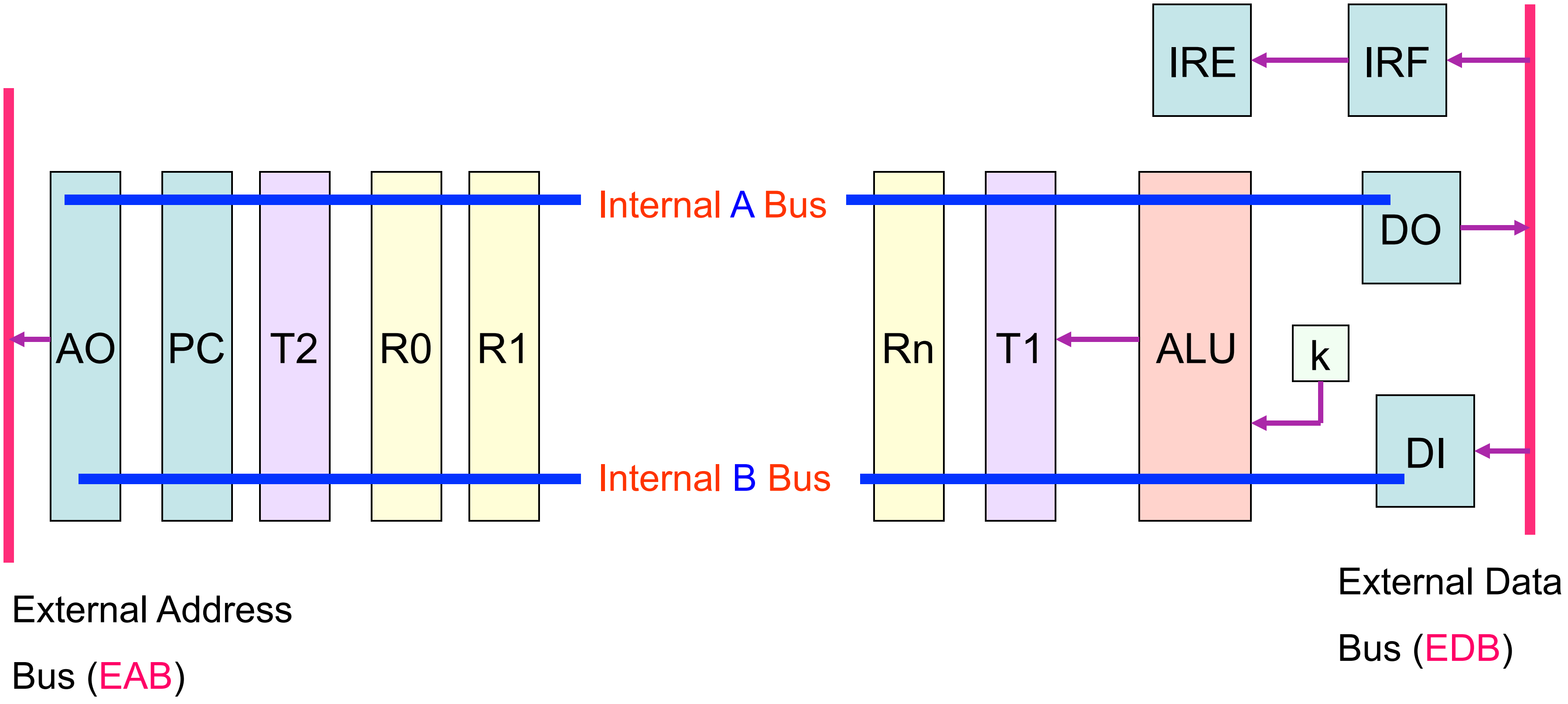
Lecture 24 (21 Sep 2015)

CADSL

MIN Instruction Set

- ✓ ADD
- ✓ AND
- ✓ BZ – Branch if zero bit is set. (Register Indirect only)
- ✓ LOAD – Second operand is source and Rx is destination
- ✓ POP – Postincrement with register indirect only
- ✓ PUSH – Predecrement with register indirect only
- ✓ STORE
- ✓ SUB
- ✓ TEST

MIN Datapath



Level1 Flowchart

Level1 Flowchart

- At the beginning of instruction execution, IRE is assumed to contain the current instruction
- Instruction execution begins with the address mode sequence
- The execution sequences for Register-to-Register instructions cannot be shared
- The execution sequences for standard dual operand instructions are identical

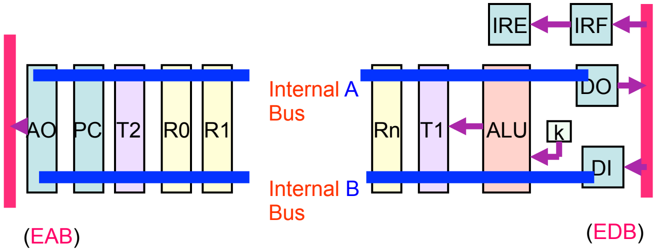
Level 1 Flowchart-
Address Mode Sequences

Base Plus Displacement

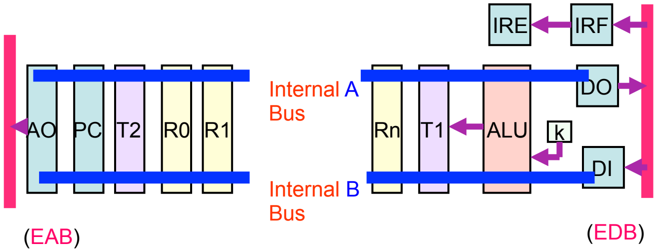
edb → di pc → a → alu, ao +1 → alu 🟥	
t1 → a → pc	
di → b → alu ry → a → alu	
edb → di t1 → b → ao, t2 🟥	

Register Indirect

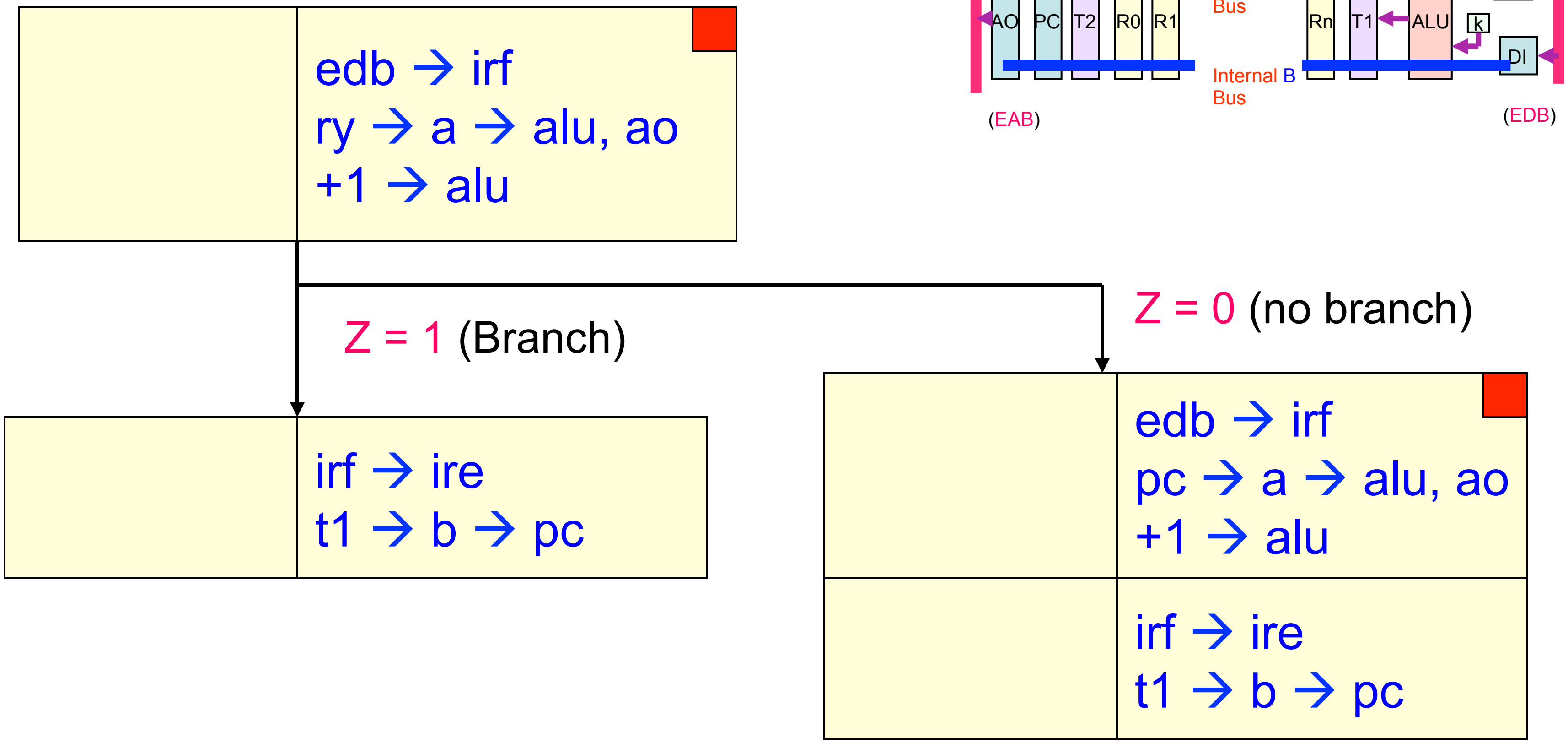
edb → di ry → b → ao, t2 🟥	



Level 1 Flowchart- Address Mode Sequences



Branch Instruction



edb → irf
ry → a → alu, ao
+1 → alu

Z = 1 (Branch)

irf → ire
t1 → b → pc

Z = 0 (no branch)

edb → irf
pc → a → alu, ao
+1 → alu

irf → ire
t1 → b → pc

Level 1 Flowchart - Execution Sequences

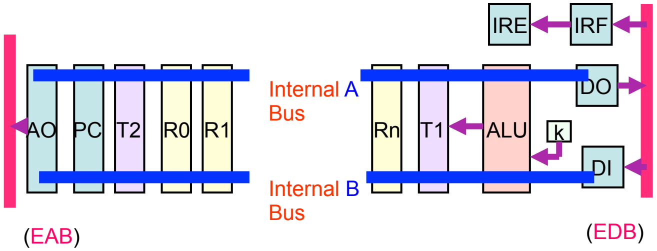
Execution sequences with memory operand reference

LOAD

di → b → rx, t2	edb → irf pc → a → alu, ao +1 → alu 🟥
t2 → a → alu 0 → alu	irf → ire t1 → b → pc

STORE

rx → a → alu, do t2 → b → ao 0 → alu 🟥	edb → irf pc → a → alu, ao +1 → alu 🟥
	irf → ire t1 → b → pc



Level 1 Flowchart - Execution Sequences

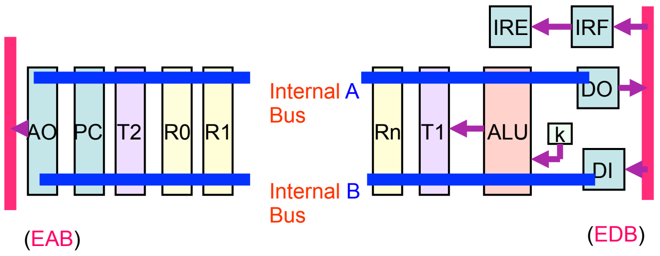
Execution sequences with memory operand reference

ADD

di → b → alu rx → a → alu	edb → irf 🟥 pc → a → alu, ao +1 → alu
t1 → a → do 🟥 t2 → b → ao	irf → ire t1 → b → pc

SUB

di → b → alu rx → a → alu	edb → irf 🟥 pc → a → alu, ao +1 → alu
t1 → a → do 🟥 t2 → b → ao	irf → ire t1 → b → pc



Level 1 Flowchart - Execution Sequences

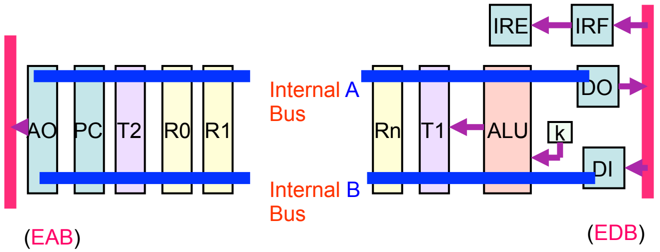
Execution sequences with memory operand reference

AND

di → b → alu rx → a → alu	edb → irf pc → a → alu, ao +1 → alu 🟥
t1 → a → do t2 → b → ao 🟥	irf → ire t1 → b → pc

TEST

di → b → t2	edb → irf pc → a → alu, ao +1 → alu 🟥
t2 → a → alu 0 → alu	irf → ire t1 → b → pc



Level 1 Flowchart - Execution Sequences

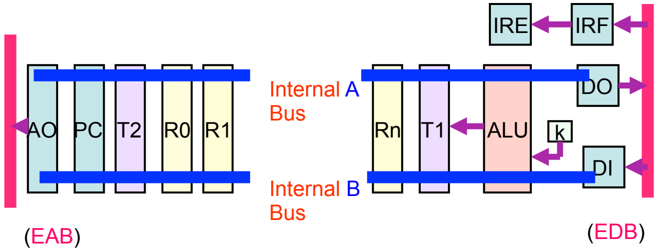
Execution sequences for Register-to-Register and special instructions

LOAD

ry → a → alu,rx 0 → alu	🟥 edb → irf pc → a → alu, ao +1 → alu
	irf → ire t1 → a → pc

STORE

rx → a → alu, ry 0 → alu	🟥 edb → irf pc → a → alu, ao +1 → alu
	irf → ire t1 → a → pc



Level 1 Flowchart - Execution Sequences

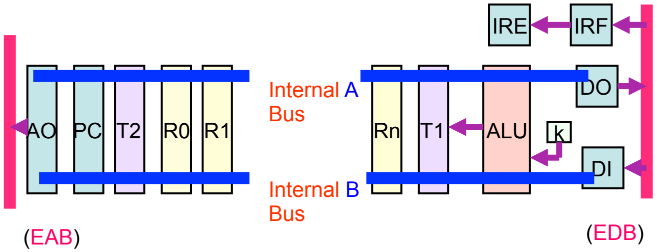
Execution sequences for Register-to-Register and special instructions

ADD

rx → a → alu ry → b → alu	edb → irf pc → a → alu, ao +1 → alu
t1 → a → ry	irf → ire t1 → a → pc

SUB

rx → a → alu ry → b → alu	edb → irf pc → a → alu, ao +1 → alu
t1 → a → ry	irf → ire t1 → a → pc



Level 1 Flowchart - Execution Sequences

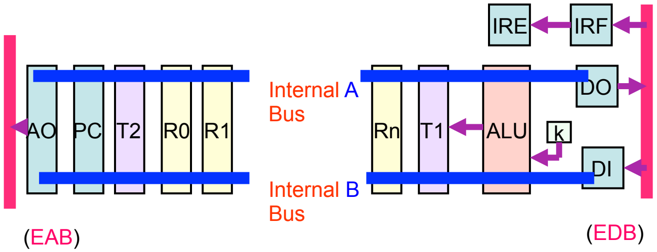
Execution sequences for Register-to-Register and special instructions

POP

🟥 edb → di ry → a → alu, ao +1 → alu	🟥 edb → irf pc → a → alu, ao +1 → alu
di → b → rx t1 → a → ry	irf → ire t1 → a → pc

PUSH

ry → a → alu -1 → alu	🟥 edb → irf pc → a → alu, ao +1 → alu
🟥 rx → a → do t1 → b → ao, ry	irf → ire t1 → a → pc



Merged Level 1 Flowchart: Address Mode Sequences

Base Plus Displacement

edb → di
pc → a → alu, ao
+1 → alu
abdm1

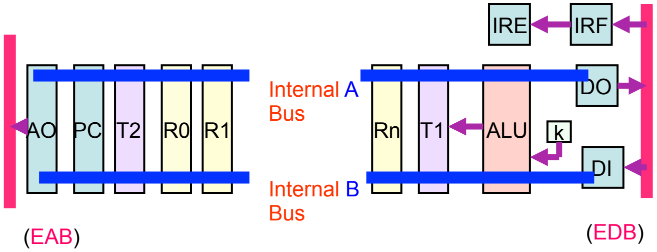
t1 → a → pc
abdm2

di → b → alu
ry → a → alu
abdm3

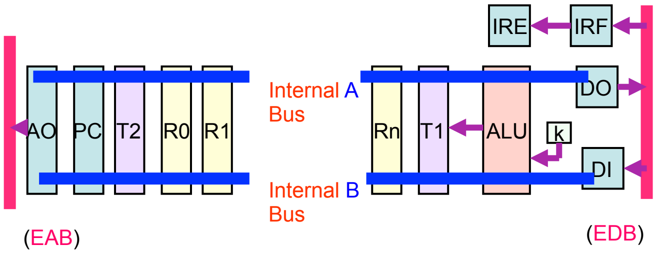
edb → di
t1 → b → ao, t2
abdm4

Register Indirect

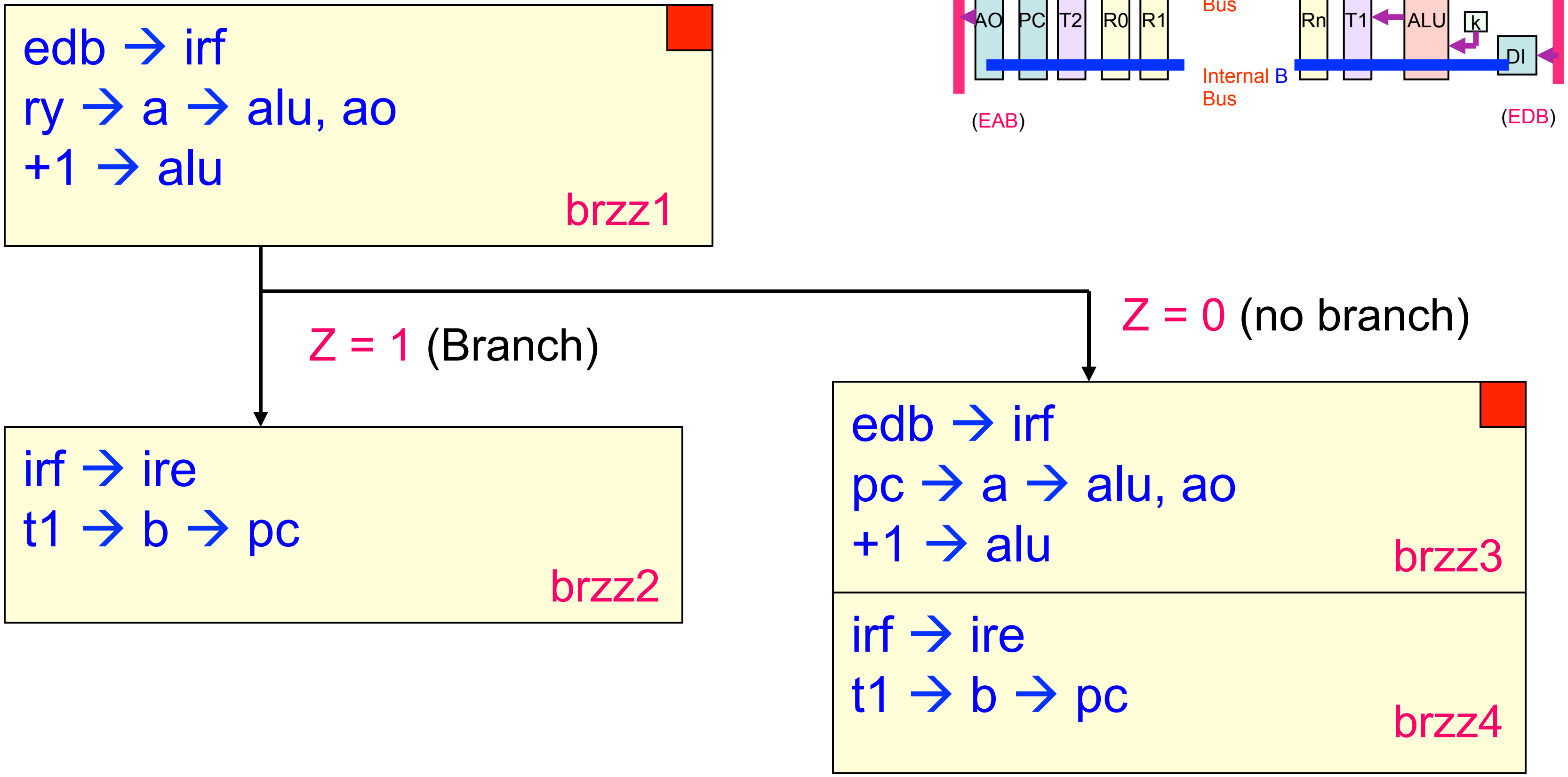
edb → di
ry → b → ao, t2
adrm1



Merged Level 1 Flowchart: Address Mode Sequences



Branch Instruction



edb → irf
ry → a → alu, ao
+1 → alu
brzz1

Z = 1 (Branch)

irf → ire
t1 → b → pc
brzz2

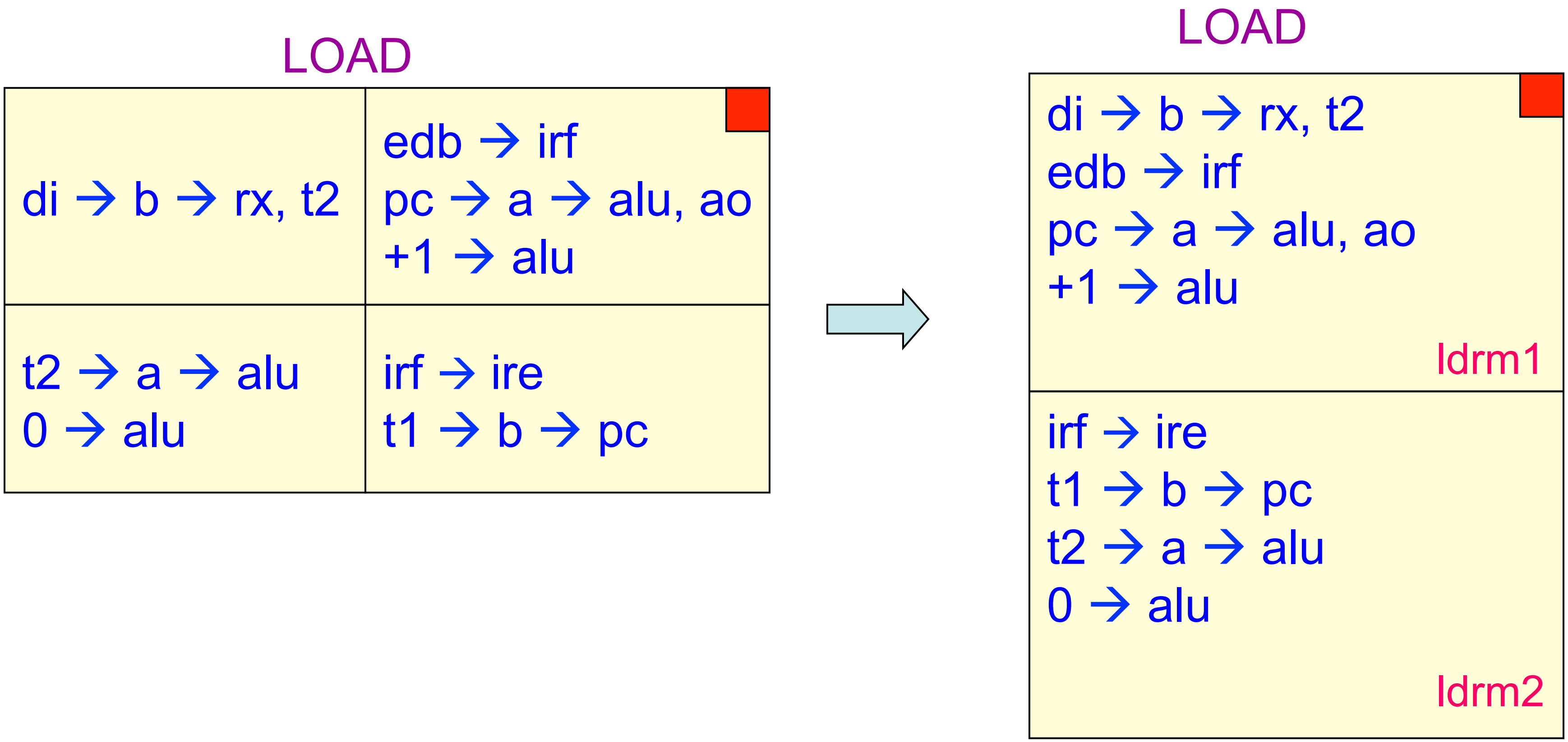
Z = 0 (no branch)

edb → irf
pc → a → alu, ao
+1 → alu
brzz3

irf → ire
t1 → b → pc
brzz4

Merged Level 1 Flowchart: Execution Sequences

Execution sequences with memory operand reference



LOAD

di → b → rx, t2	edb → irf pc → a → alu, ao +1 → alu
t2 → a → alu 0 → alu	irf → ire t1 → b → pc

LOAD

di → b → rx, t2
edb → irf
pc → a → alu, ao
+1 → alu

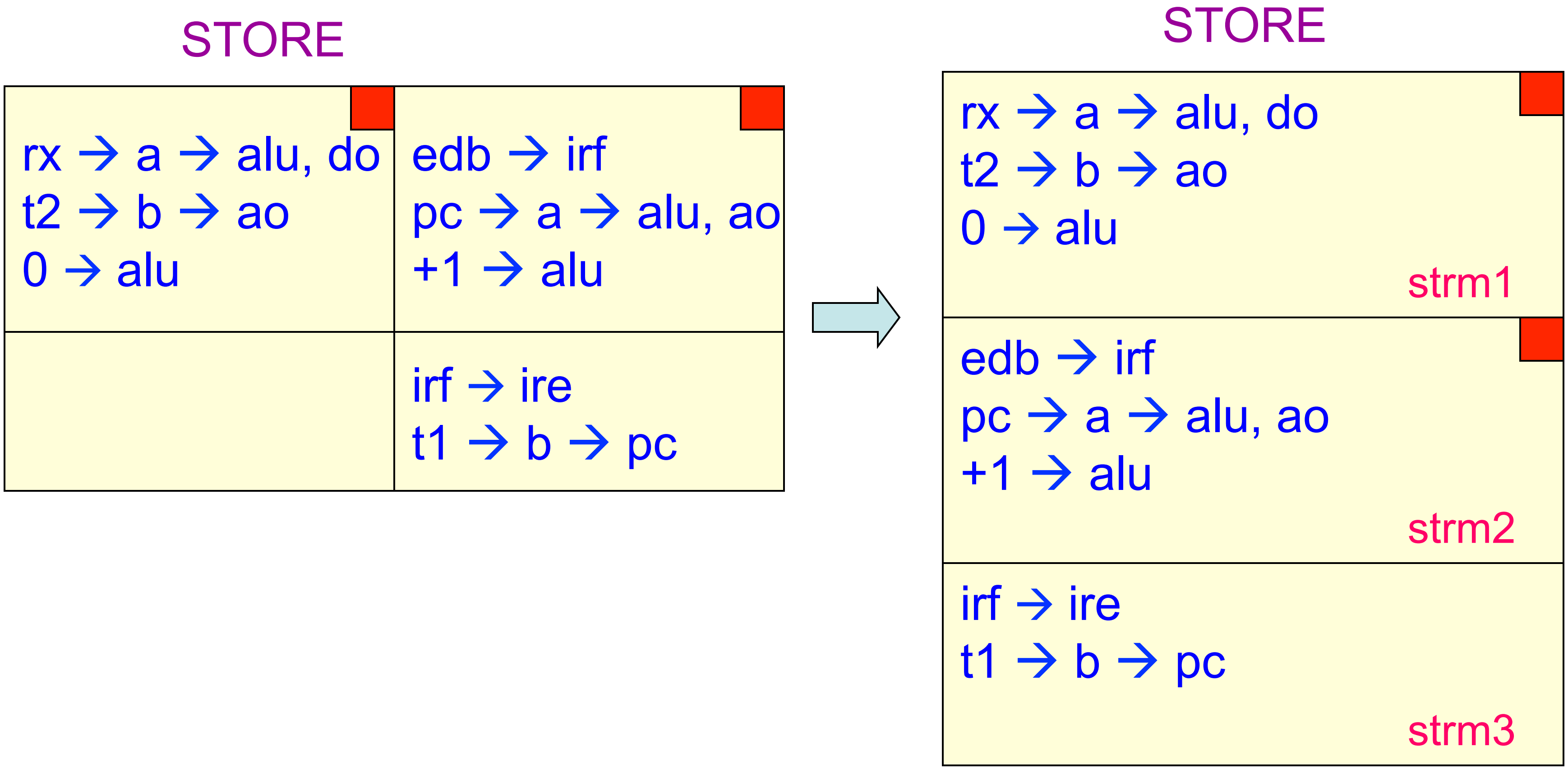
ldrm1

irf → ire
t1 → b → pc
t2 → a → alu
0 → alu

ldrm2

Merged Level 1 Flowchart: Execution Sequences

Execution sequences with memory operand reference



STORE

rx → a → alu, do
t2 → b → ao
0 → alu

edb → irf
pc → a → alu, ao
+1 → alu

irf → ire
t1 → b → pc

STORE

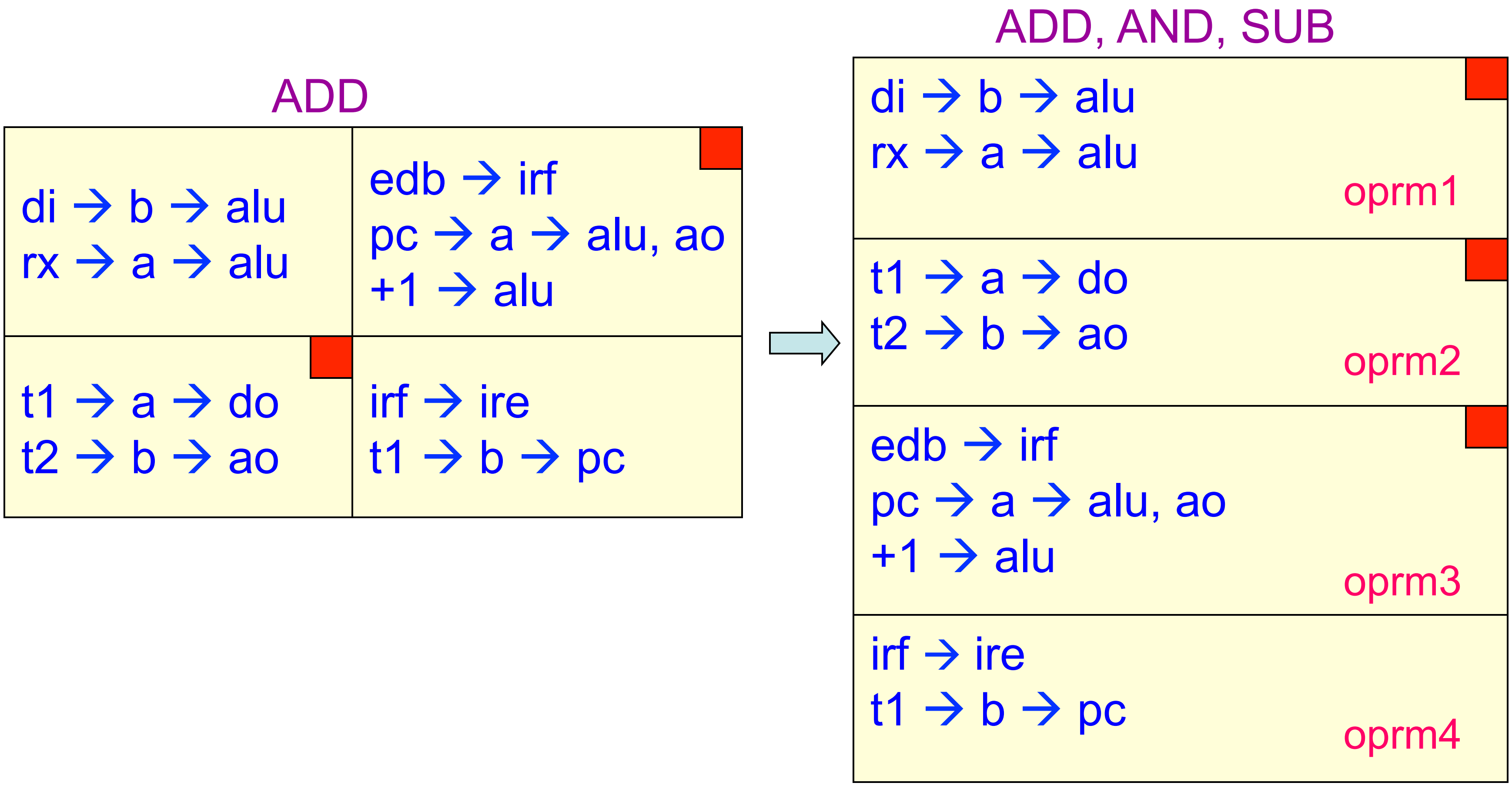
rx → a → alu, do
t2 → b → ao
0 → alu
strm1

edb → irf
pc → a → alu, ao
+1 → alu
strm2

irf → ire
t1 → b → pc
strm3

Merged Level 1 Flowchart: Execution Sequences

Execution sequences with memory operand reference



ADD

di → b → alu rx → a → alu	edb → irf pc → a → alu, ao +1 → alu
t1 → a → do t2 → b → ao	irf → ire t1 → b → pc

ADD, AND, SUB

di → b → alu
rx → a → alu
oprm1

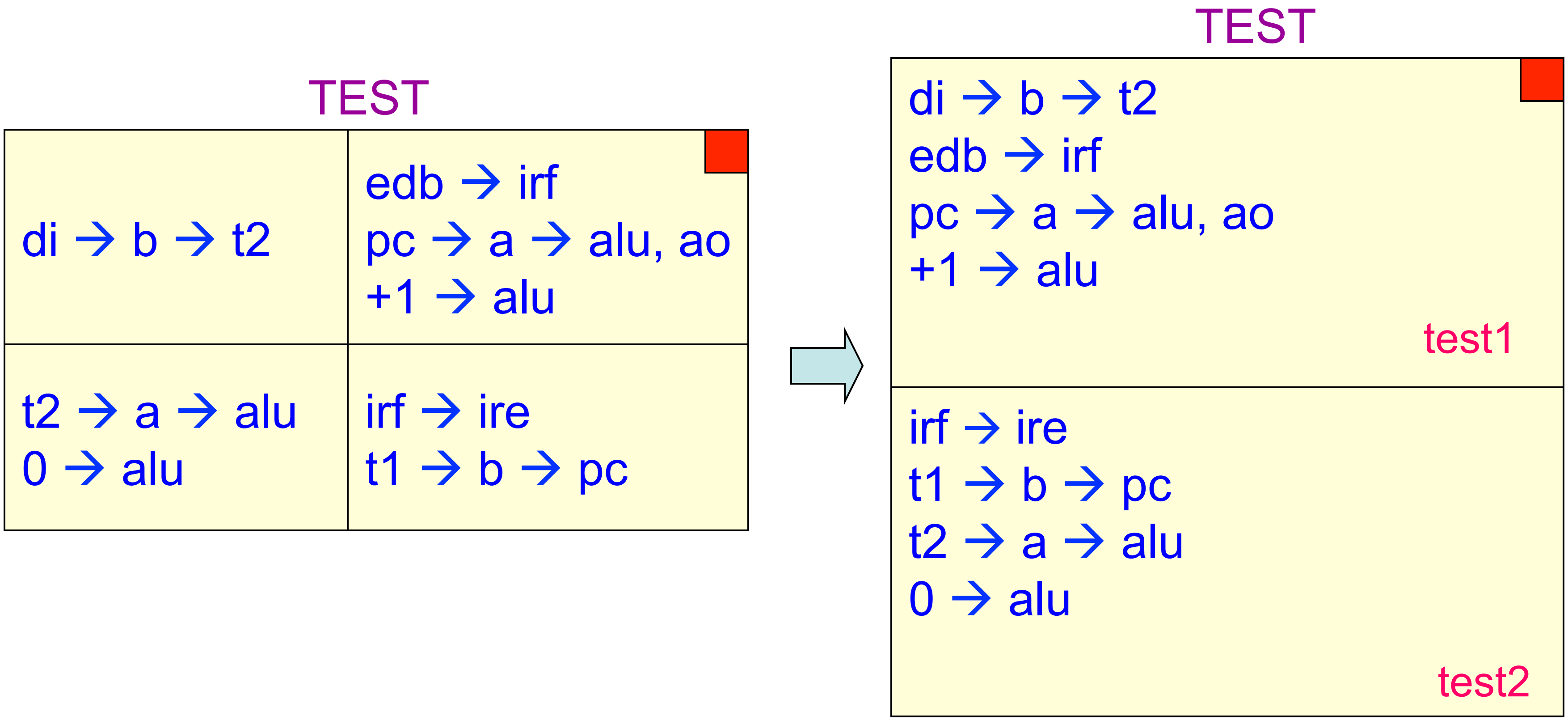
t1 → a → do
t2 → b → ao
oprm2

edb → irf
pc → a → alu, ao
+1 → alu
oprm3

irf → ire
t1 → b → pc
oprm4

Merged Level 1 Flowchart: Execution Sequences

Execution sequences with memory operand reference



TEST

di → b → t2	edb → irf pc → a → alu, ao +1 → alu
t2 → a → alu 0 → alu	irf → ire t1 → b → pc

TEST

di → b → t2
edb → irf
pc → a → alu, ao
+1 → alu

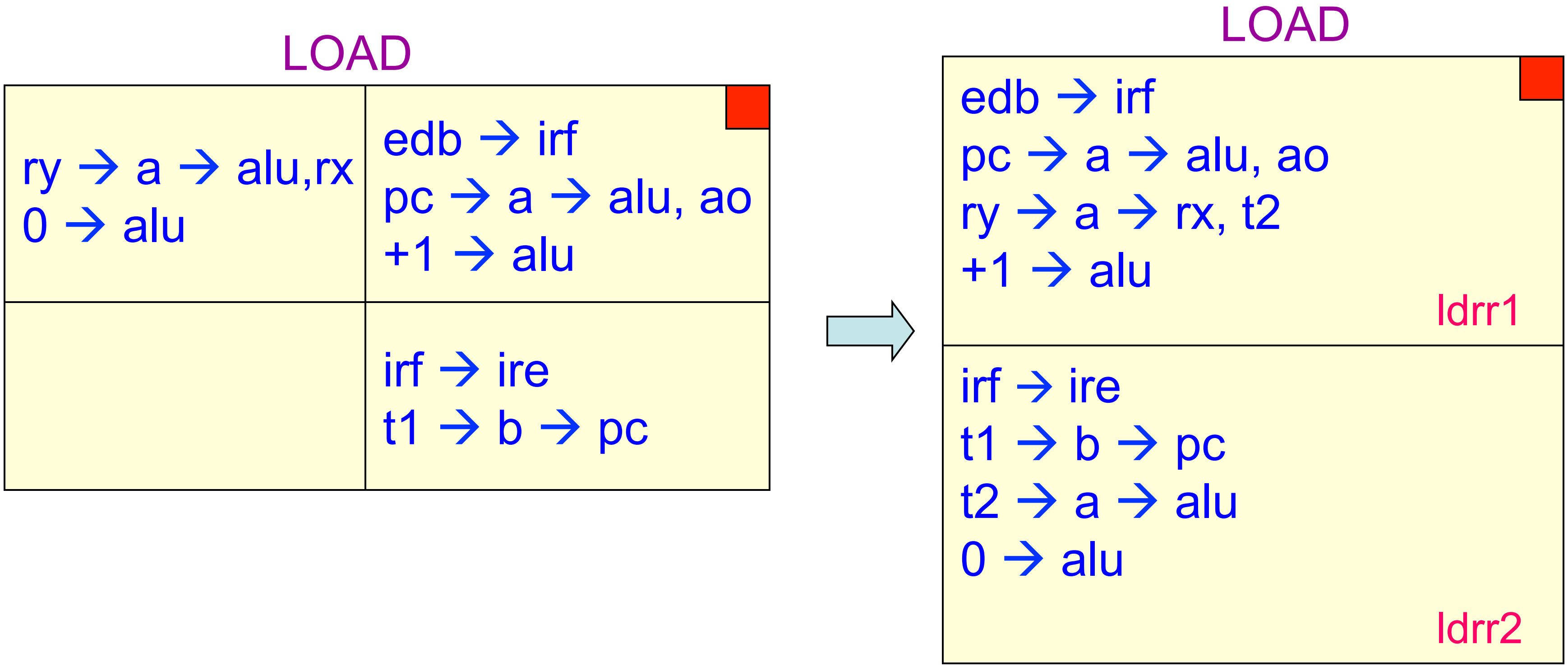
test1

irf → ire
t1 → b → pc
t2 → a → alu
0 → alu

test2

Merged Level 1 Flowchart: Execution Sequences

Execution sequences for Register-to-Register and special instructions



LOAD

ry → a → alu,rx 0 → alu	edb → irf pc → a → alu, ao +1 → alu
	irf → ire t1 → b → pc

LOAD

edb → irf
pc → a → alu, ao
ry → a → rx, t2
+1 → alu

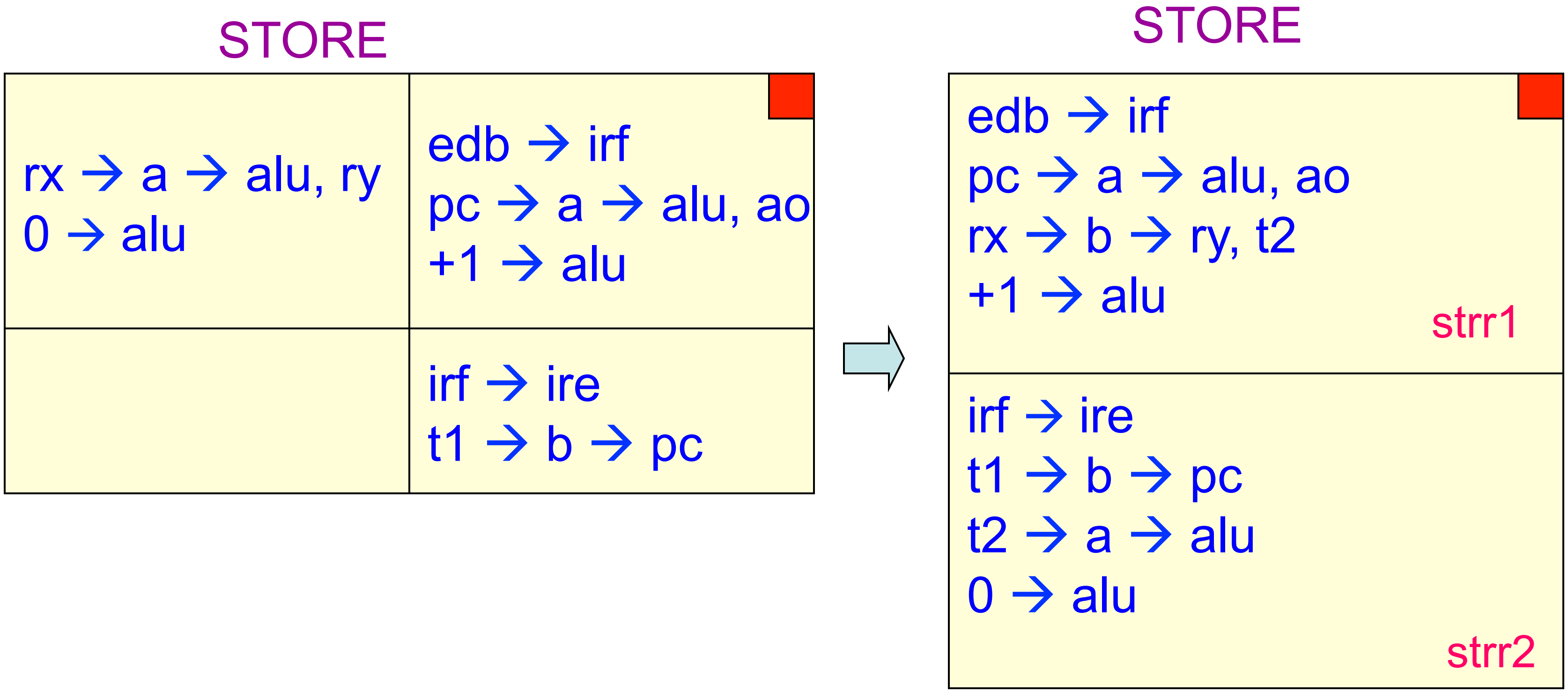
ldrr1

irf → ire
t1 → b → pc
t2 → a → alu
0 → alu

ldrr2

Merged Level 1 Flowchart: Execution Sequences

Execution sequences for Register-to-Register and special instructions



STORE

rx → a → alu, ry
0 → alu

edb → irf
pc → a → alu, ao
+1 → alu

irf → ire
t1 → b → pc

STORE

edb → irf
pc → a → alu, ao
rx → b → ry, t2
+1 → alu

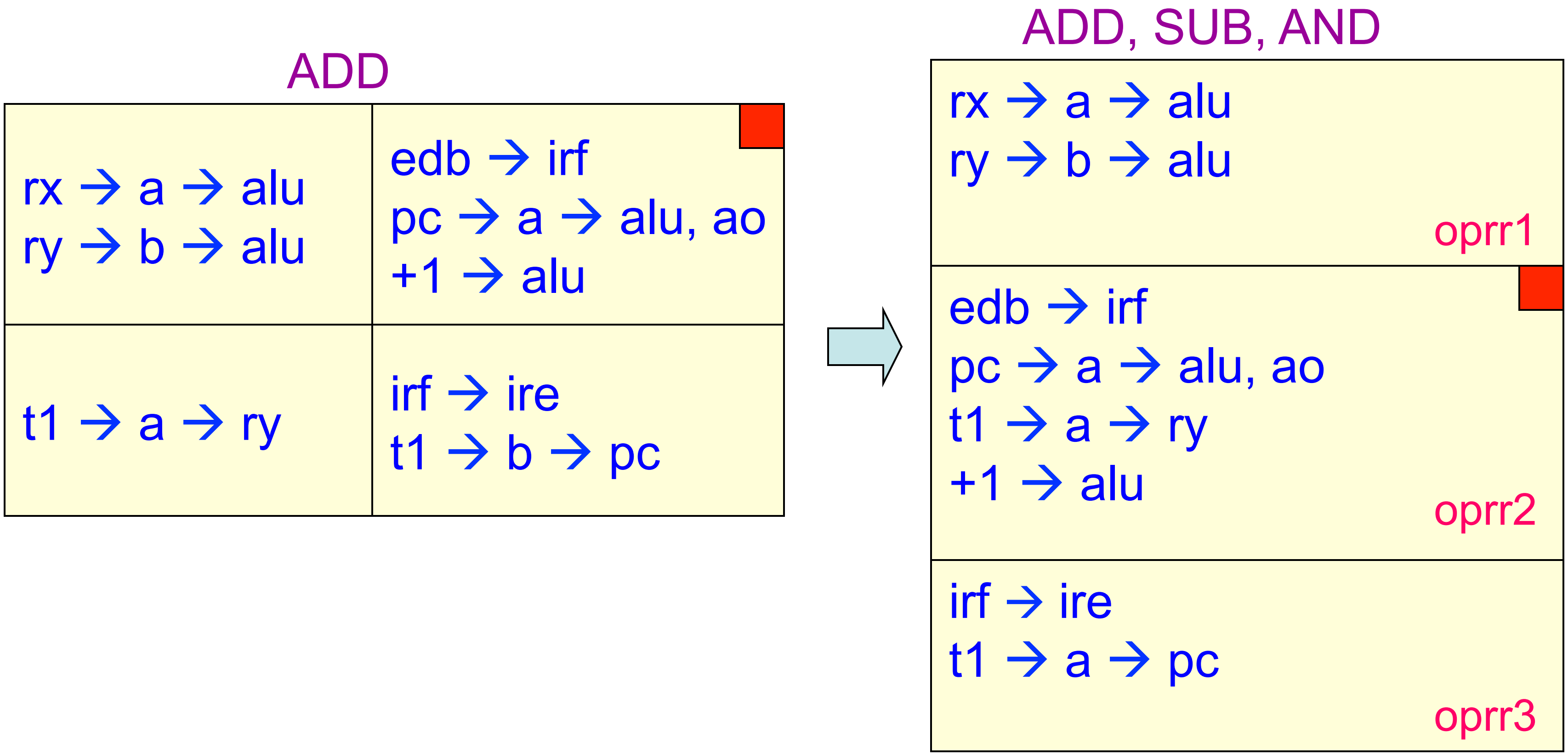
strr1

irf → ire
t1 → b → pc
t2 → a → alu
0 → alu

strr2

Merged Level 1 Flowchart: Execution Sequences

Execution sequences for Register-to-Register and special instructions



ADD

rx → a → alu ry → b → alu	edb → irf pc → a → alu, ao +1 → alu
t1 → a → ry	irf → ire t1 → b → pc

ADD, SUB, AND

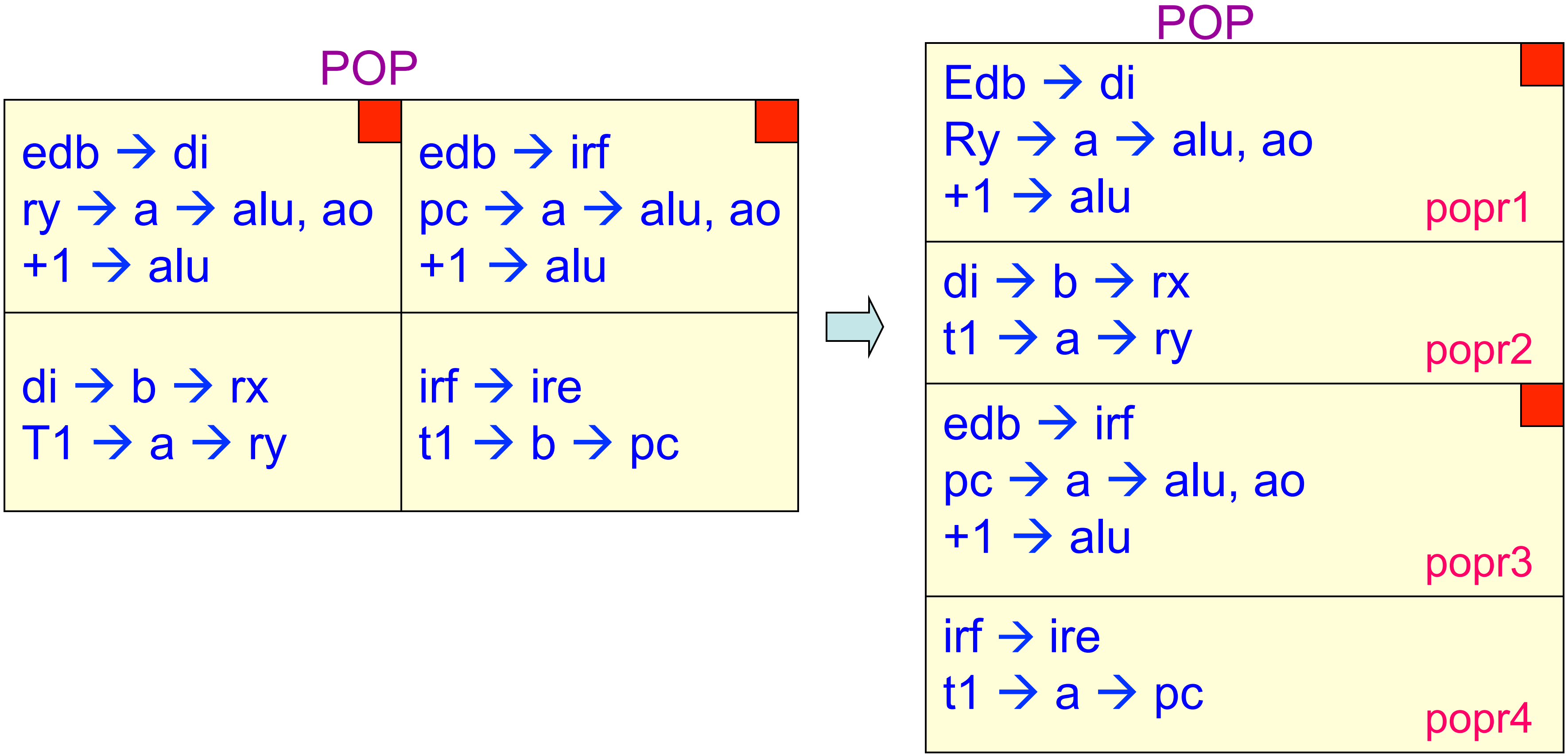
rx → a → alu
ry → b → alu
oprr1

edb → irf
pc → a → alu, ao
t1 → a → ry
+1 → alu
oprr2

irf → ire
t1 → a → pc
oprr3

Level 1 Flowchart - Execution Sequences

Execution sequences for Register-to-Register and special instructions



POP

edb → di
ry → a → alu, ao
+1 → alu

edb → irf
pc → a → alu, ao
+1 → alu

di → b → rx
T1 → a → ry

irf → ire
t1 → b → pc

POP

Edb → di
Ry → a → alu, ao
+1 → alu
popr1

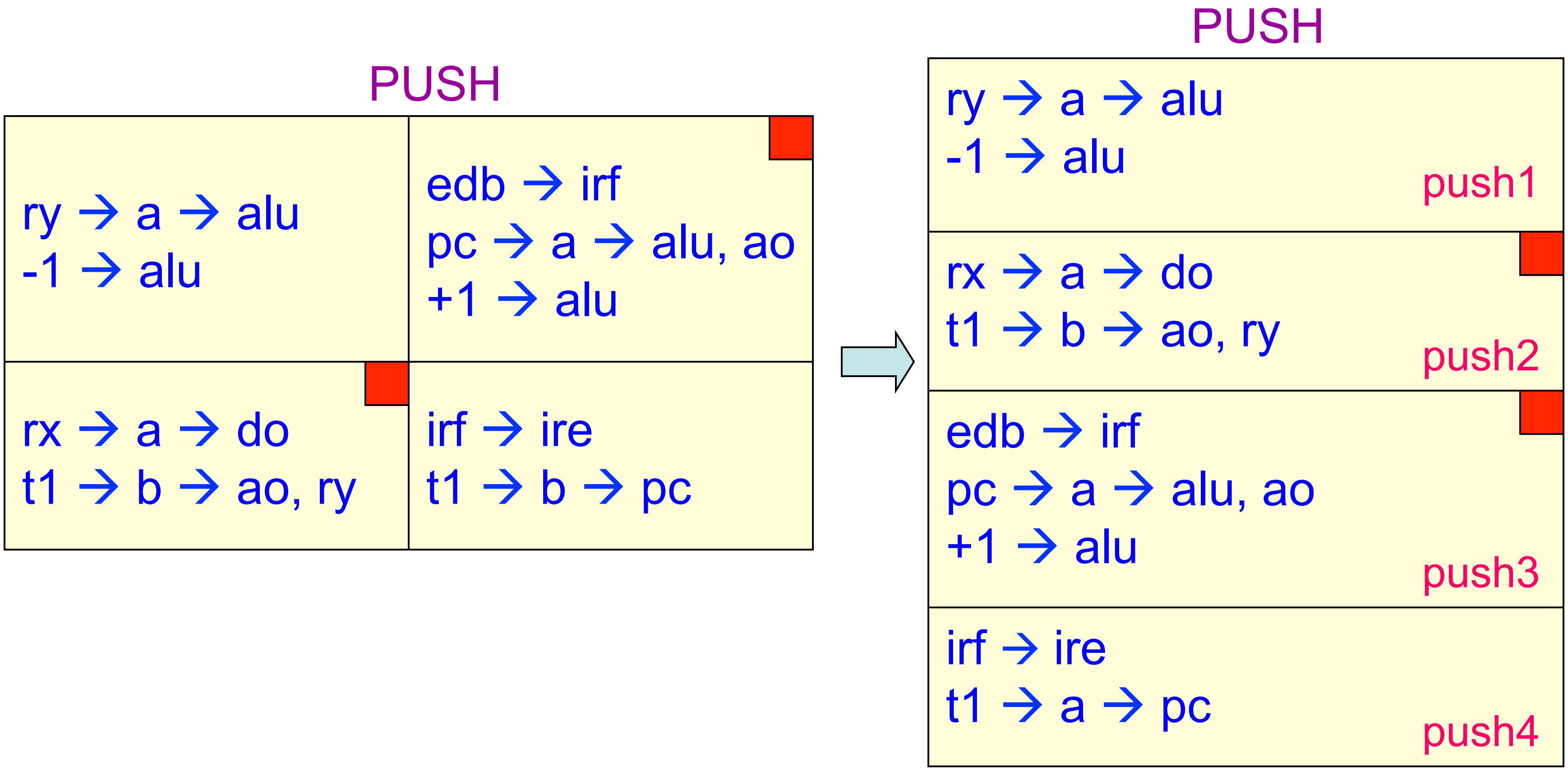
di → b → rx
t1 → a → ry
popr2

edb → irf
pc → a → alu, ao
+1 → alu
popr3

irf → ire
t1 → a → pc
popr4

Merged Level 1 Flowchart: Execution Sequences

Execution sequences for Register-to-Register and special instructions



PUSH

ry → a → alu
-1 → alu

edb → irf
pc → a → alu, ao
+1 → alu

rx → a → do
t1 → b → ao, ry

irf → ire
t1 → b → pc

PUSH

ry → a → alu
-1 → alu
push1

rx → a → do
t1 → b → ao, ry
push2

edb → irf
pc → a → alu, ao
+1 → alu
push3

irf → ire
t1 → a → pc
push4

Thank You