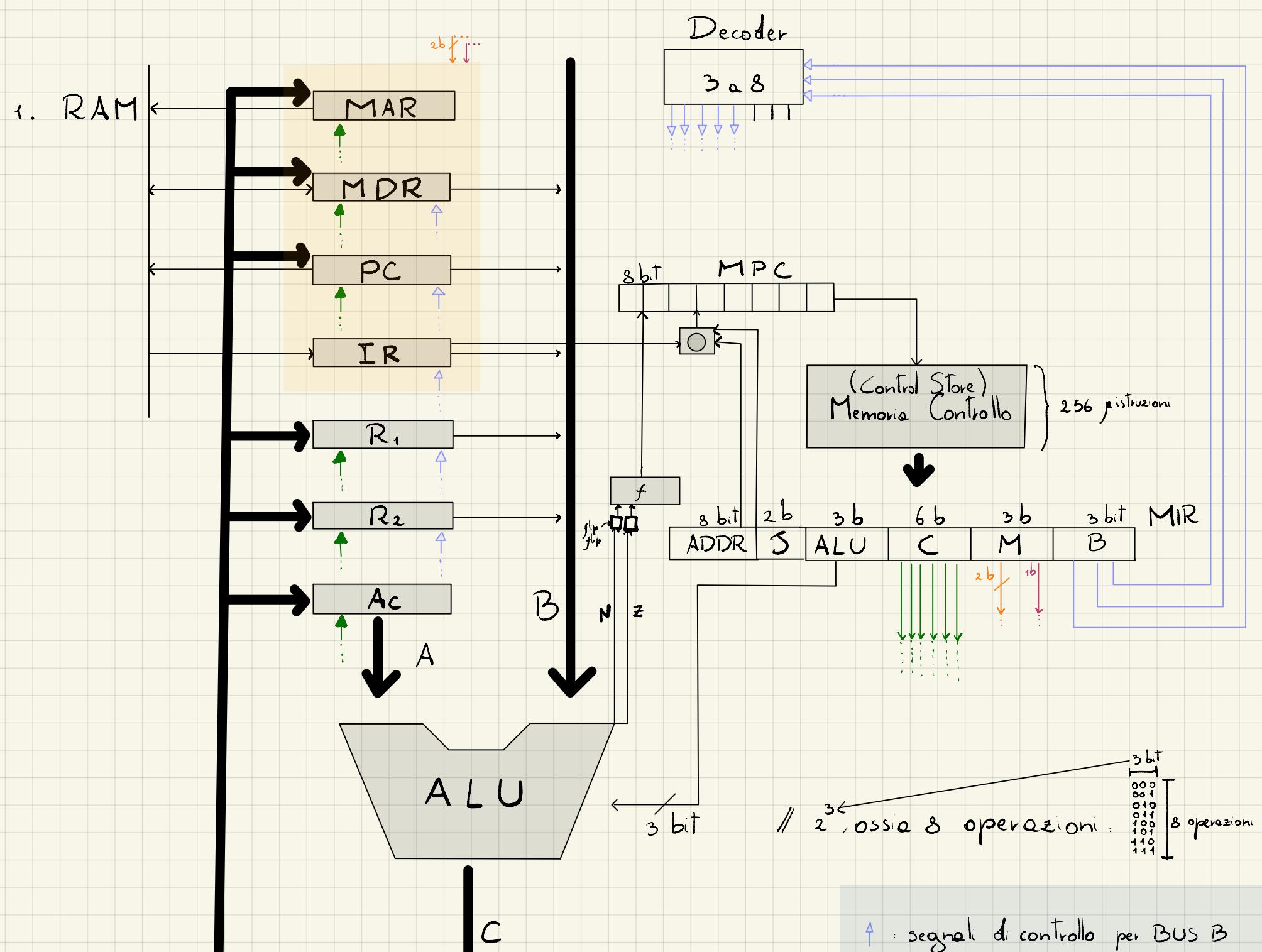


Si vuole realizzare una semplice CPU con architettura CISC a 8 bit, dotata di 2 registri general purpose, 2 registri per il fetch delle istruzioni (il program counter e il registro istruzione corrente) e 2 registri per il trasferimento dei dati da/per la memoria (1 per gli indirizzi, l'altro per i dati).
 La CPU deve essere in grado di svolgere 8 operazioni aritmetiche a numeri interi. Previsti salti condizionati.
 Tutte le altre specifiche possono essere liberamente scelte.

1. Disegnare l'architettura generale (in particolare il data path) di tale CPU (comprendere dei segnali di controllo) e illustrare concisamente il suo funzionamento.
2. Definire il formato di una microistruzione per Tale architettura, cercando di minimizzare la sua lunghezza.
3. Indicare possibili modifiche dell'architettura proposta, in grado di migliorare le prestazioni.

Elaborazione dati:

- CPU CISC a 8 bit
- 2 registri di uso generale: R_1, R_2 //general purpose
- 4 registri per RAM: . 2 per le istruzioni : . PC (Program Counter) //indirizzi per memoria di controllo
- . IR (o MBR) (Instruction Register) //istruzione corrente
- 2 per i dati :: MAR (Memory Address Register) //indirizzi
- MDR (Memory Data Register) //dati
- Previsione salti condizionati
- ALU: 8 oper. aritmetiche a numeri e Z
- Libertà per le restanti specifiche: AC = Accumulatore ; A, B, C = BUS ; MPC ; Memoria Controllo ; MIR Decoder ; ADDR: per mandare avanti il ciclo



2. In MIR, B è stato minimizzato a 3 bit anziché 5 bit, grazie al successivo utilizzo di un decoder 3 a 8.

3. potrei, per esempio, introdurre una pipeline.

Si vuole realizzare una semplice CPU con architettura RISC a 8 bit, dotata di 1 reg. gen, 1 reg. Accumulatore, 2 registri per il fetch delle istruzioni (il program counter e il registro istruzione corrente) e 2 registri per il trasferimento dei dati da/per la memoria (1 per gli indirizzi, l'altro per i dati).

La CPU deve essere in grado di svolgere 16 operazioni aritmetiche a numeri interi.

Tutte le altre specifiche possono essere liberamente scelte.

1. Disegnare l'architettura generale (in particolare il data path) di Tale CPU (comprendendo dei segnali di controllo) e illustrare concisamente il suo funzionamento.

2. Definire il formato di una istruzione per Tale architettura, cercando di minimizzare la sua lunghezza.

3. Indicare possibili modifiche dell'architettura proposta per trasformarla in un'architettura CISC.

Elaborazione dati:

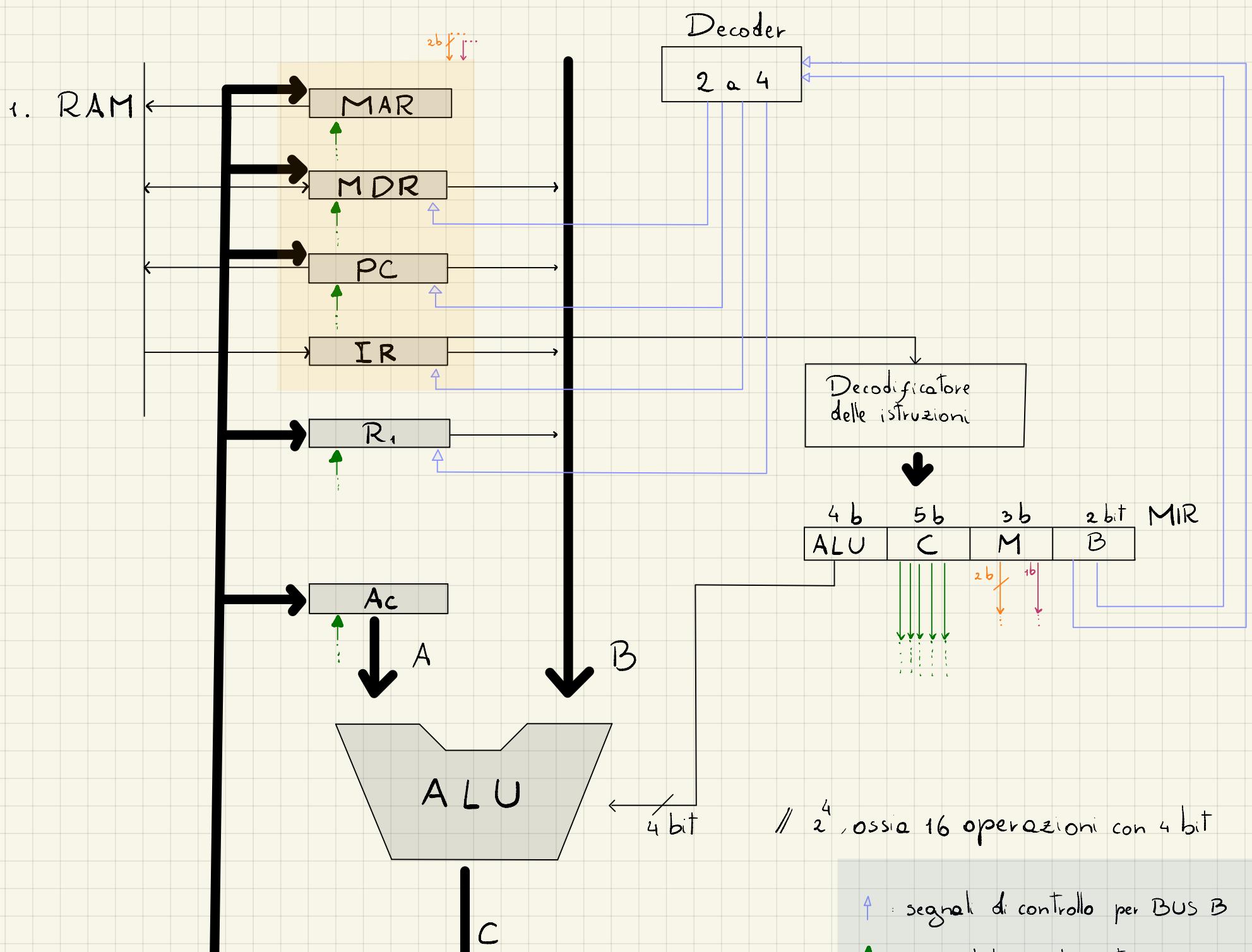
• CPU RISC a 8 bit

• 2 registri: R_i: di uso generale; Ac: Accumulatore

• 4 registri per RAM: • 2 per le istruzioni : • PC (Program counter) //indirizzi per memoria di controllo
 • IR (o MBR) (Instruction Register) //istruzione corrente
 • 2 per i dati : • MAR (Memory Address Register) //indirizzi
 • MDR (Memory Data Register) //dati

• ALU: 16 oper. aritmetiche a numeri e Z

• Libertà per le restanti specifiche: Decoder; A, B, C = BUS; MPC; Memoria Controllo; MIR
 ADDR: per mandare avanti il ciclo



2. CPU a 8 bit quindi istruzione macchina di 8 bit

3. RISC → CISC: introdurre una MPC, Memoria di controllo, Addr.

↑ : segnali di controllo per BUS B

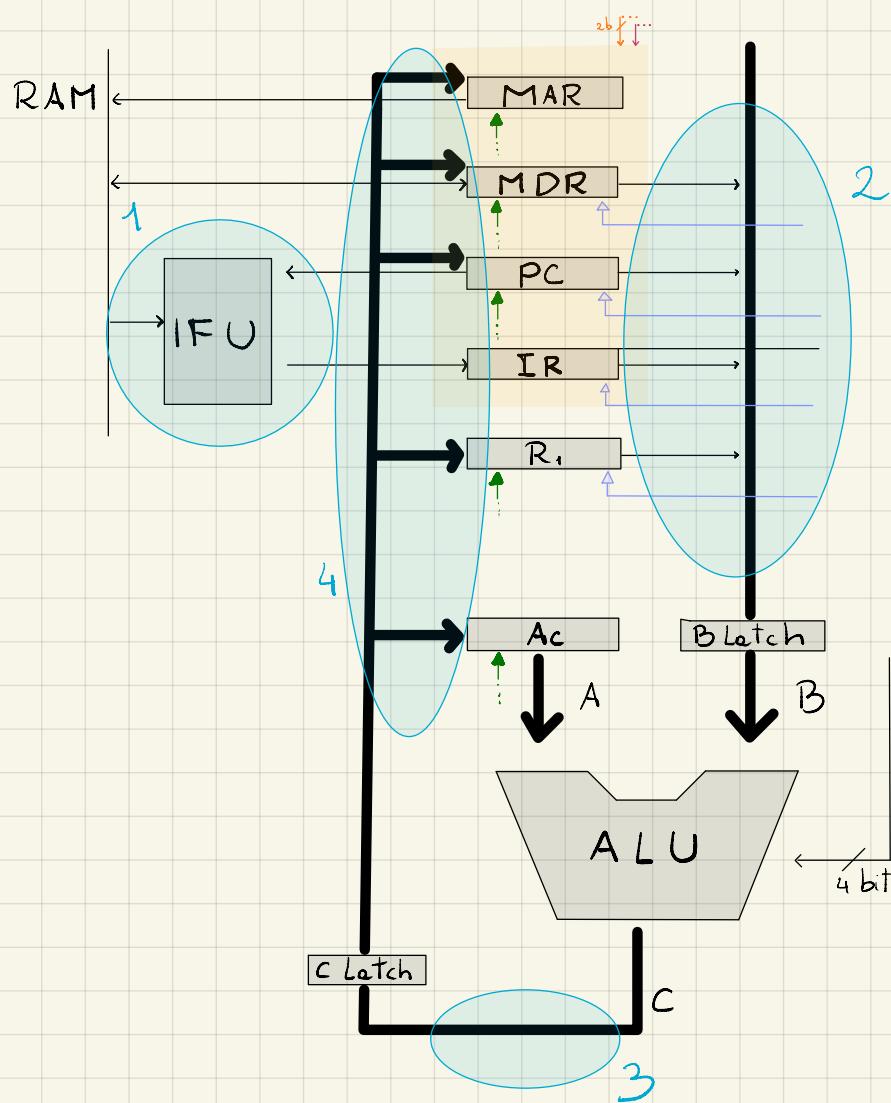
↑ : scrive il bus C nel registro

— : read / write // 2 bit, poiché $2^2 = 4$ casi

— : fetch

→ : n bit = n linee rappresentate in uno nbit

4. Disegnare solo il datapath dotando l'architettura di 4 stadi di pipeline, il primo costituito da una IFU.

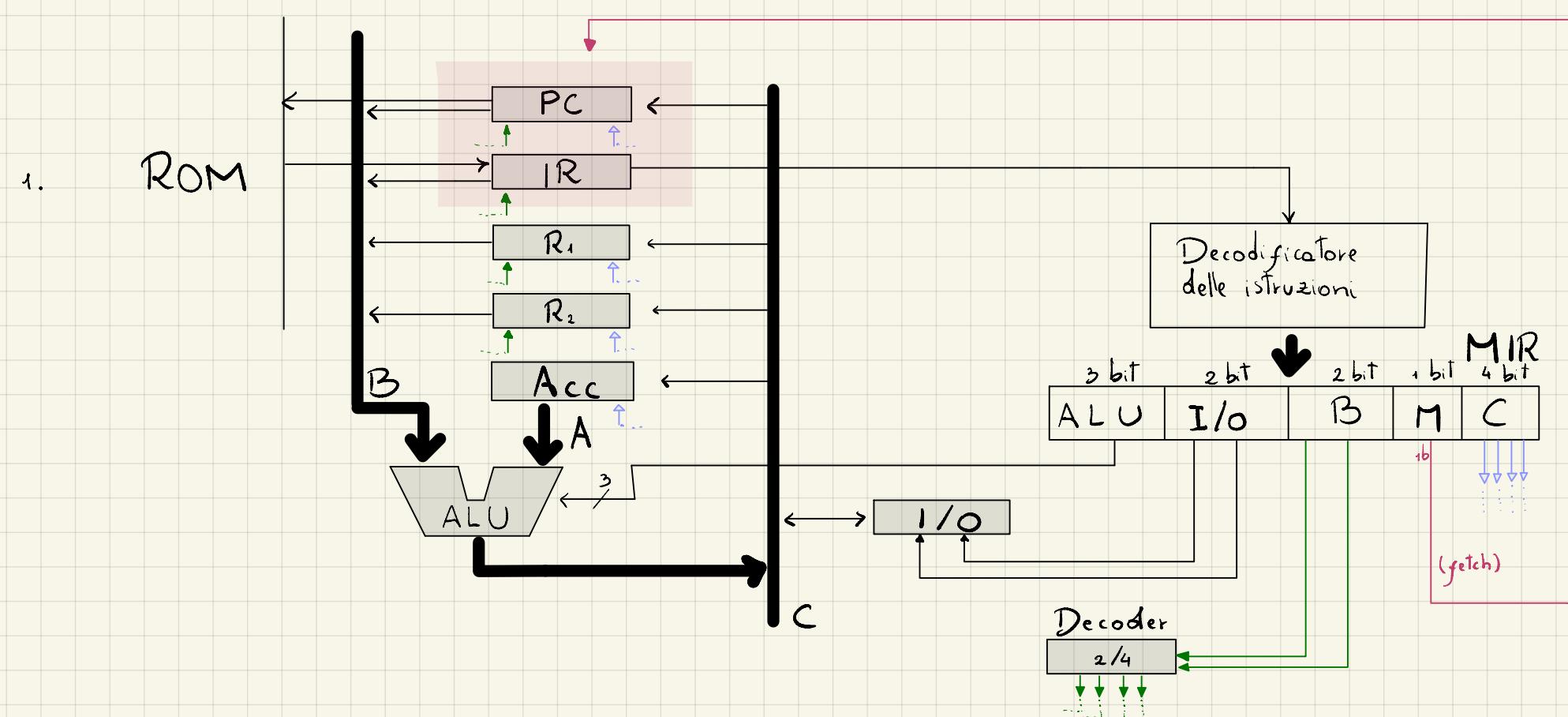


Si vuole realizzare una CPU per applicazioni embedded che non possiede RAM e nella quale tutte le istruzioni macchina da eseguire sono memorizzate in una ROM. Tale CPU è dotata di due registri general purpose, un registro accumulatore, una porta di I/O e due registri per il caricamento delle istruzioni dalla ROM. La CPU deve essere in grado di eseguire 8 operazioni aritmetiche a numeri e \mathbb{Z} . L'esecuzione delle istruzioni macchina è strettamente sequenziale. Tutte le altre specifiche possono essere liberamente scelte.

1. Disegnare l'architettura generale (in particolare il data path) di tale CPU (comprendendo dei segnali di controllo) secondo i principi RISC e illustrare concisamente il suo funzionamento.

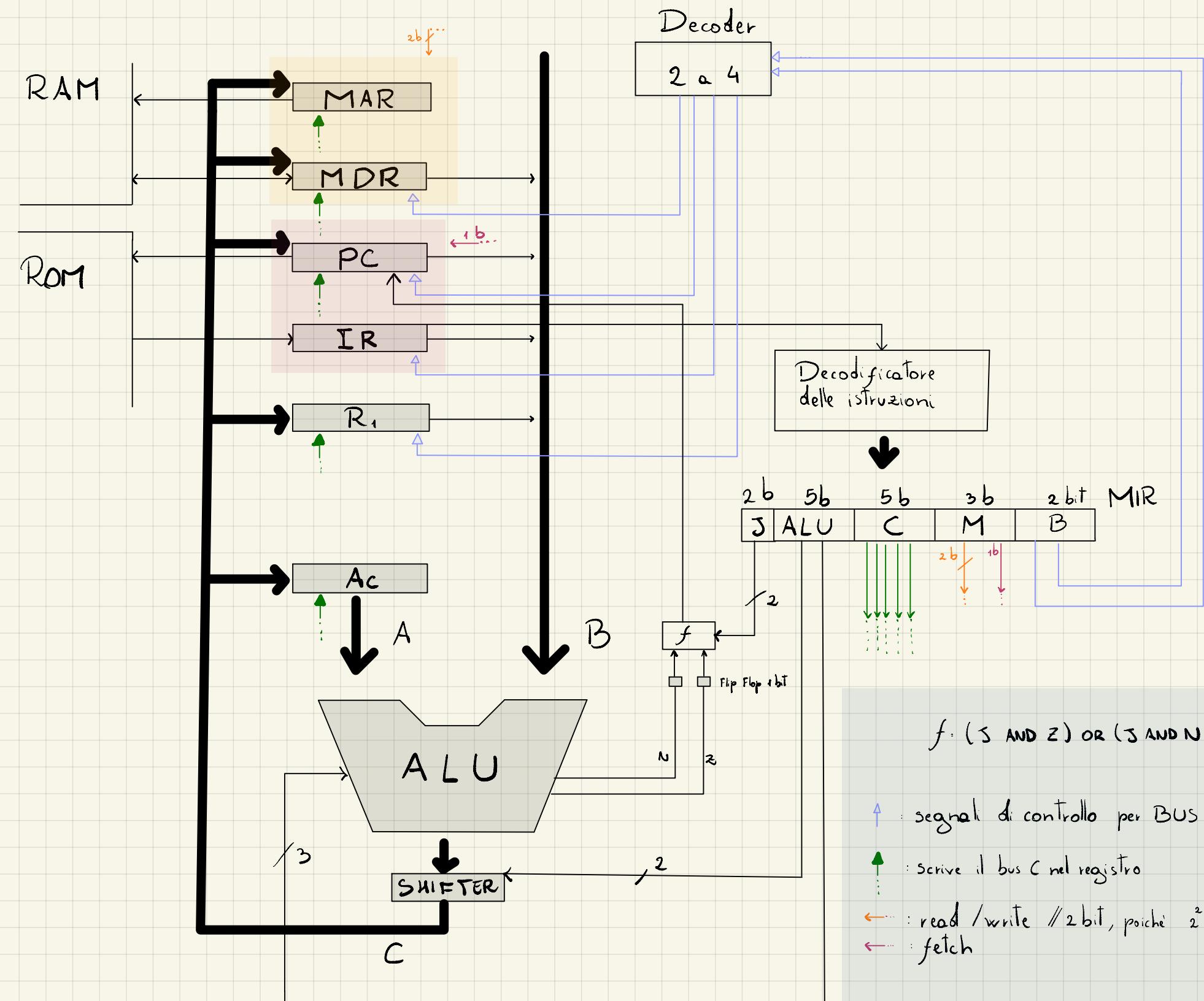
Elaborazione Dati:

- System Embedded RISC
- No RAM, Solo ROM
- 2 registri general purpose R
- 1 registro accumulatore
- 1 porta I/O
- 2 registri per istruzioni ROM : 1 PC e 1 IR
- 8 operazioni per CPU
- esecuzione sequenziale, no pipeline
- restanti specifiche libere: C: MAIN BUS



- CPU RISC A 8 bit
- 1 Registro gen.
- 1 R. Accumulatore
- MAR / MDR (RAM)
- PC / IR (ROM)
- 8 op. per ALU con scalatura a 2
- Possibilità salti condizionali al risultato pari o e negativo delle ALU

1. Disegno:



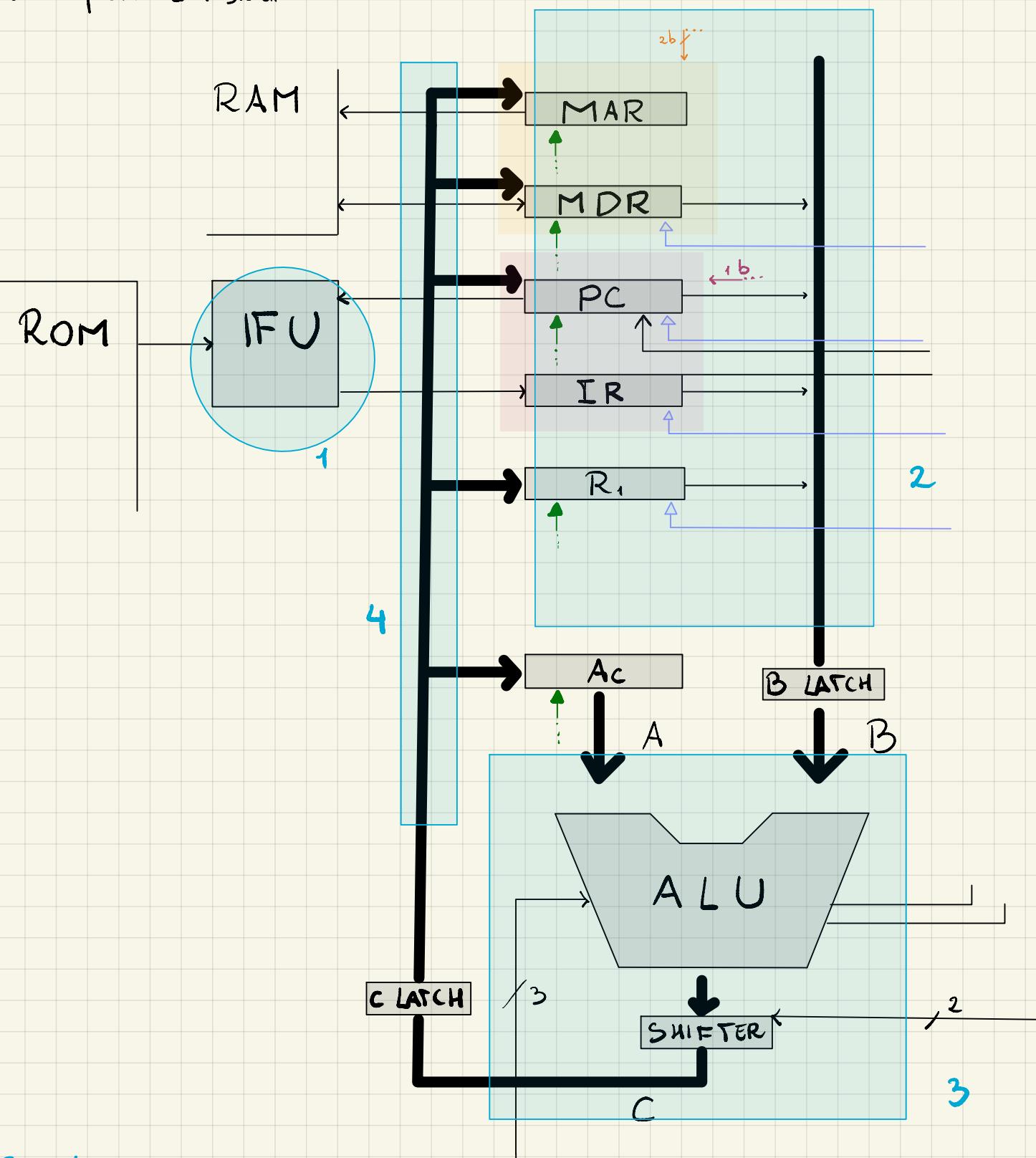
Nel Datapath sono presenti 6 registri: R₁: general purpose; Ac: Accumulatore; MAR / MDR: permettono di scrivere e leggere con la RAM; PC / IR: permettono la lettura del programma eseguibile (flusso di Byte);

Il M.I.R. contiene: 2 bit per il JAM (J);
 5 bit per le operazioni: 3 per ALU, poiché $8 \cdot 2^3$ oper. bastano 3 bit;
 2 per SHIFTER, poiché permette una scalatura a 2;
 5 bit per il Bus C: IR non è connesso al Bus C;
 3 bit per M: 2 bit (read / write) vanno nella sezione colorata gialla (MAR / MDR);
 1 bit (fetch) va nella sezione colorata rossa (PC / IR)

2 bit per B: poiché l'inserimento di un Decoder 2/4 ha permesso la minimizzazione nel MIR.
 Destinazione MDR, PC, IR, R₁.

2. Pipeline a 4 stadi:

CANULLI



STADI:

1. Prelevo istruzioni
2. Accesso operandi
3. Esecuzione operazioni ALU
4. Scrittura nei registri

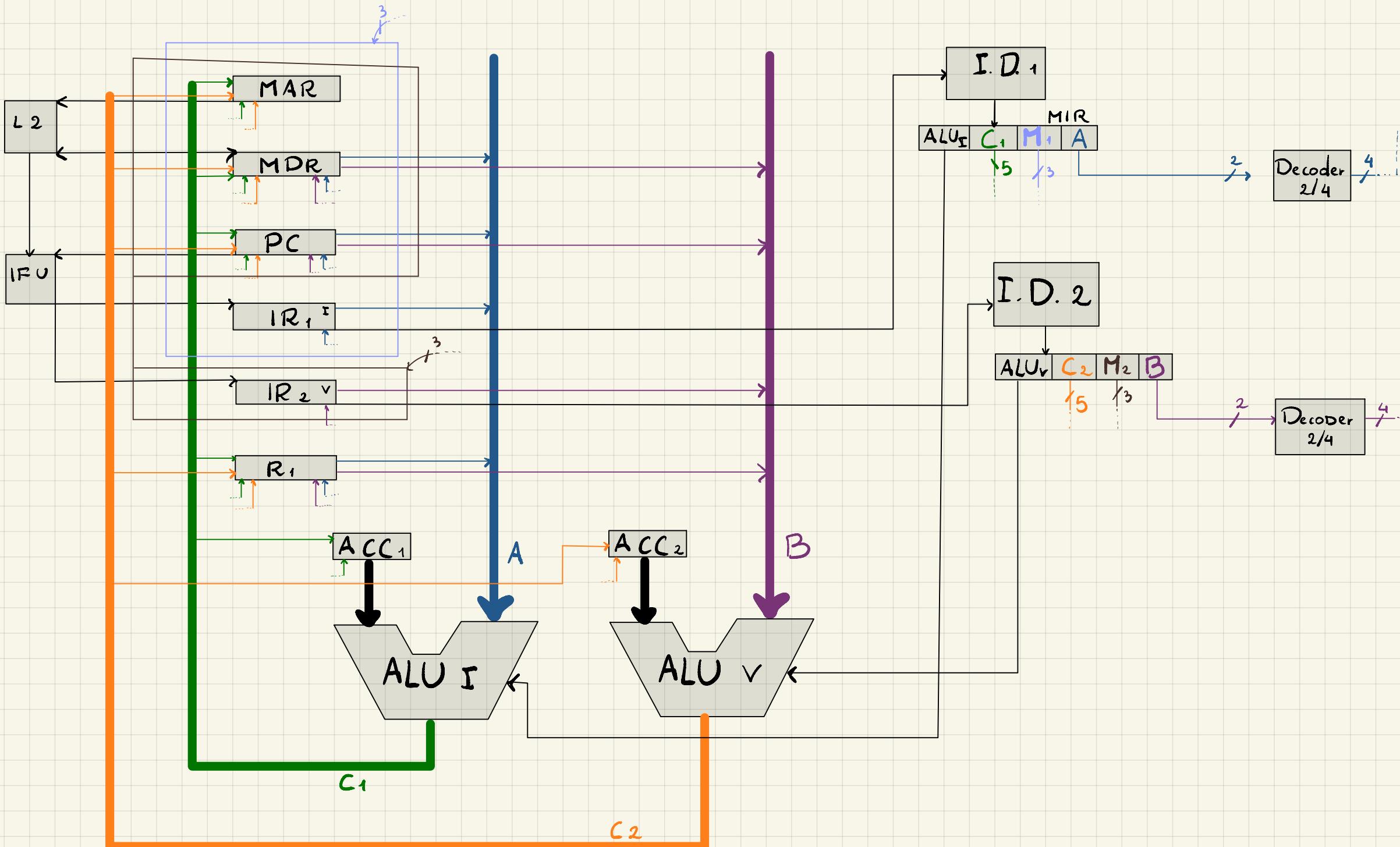
	CICLO 1	CICLO 2	CICLO 3	CICLO 4	CICLO 5	CICLO 6
ISTRUZIONE 1	1	2	3	4	1	2
ISTRUZIONE 2		1	2	3	4	1
ISTRUZIONE 3			1	2	3	4
ISTRUZIONE 4				1	2	3

Tempo →

CPU - RISC

- Dual core con cache L2 condivisa
- Architettura super scalare : 2 ALU : 1 Interi, 1 Virgola Mobile
- Pre-fetching

Microarchitettura Core:



CPU - EMBEDDED (approccio RISC)

