



# Arhitectura Sistemelor de Calcul



**Universitatea Politehnica Bucuresti**  
**Facultatea de Automatica si Calculatoare**

[cs.ncit.pub.ro](http://cs.ncit.pub.ro)  
[curs.cs.pub.ro](http://curs.cs.pub.ro)



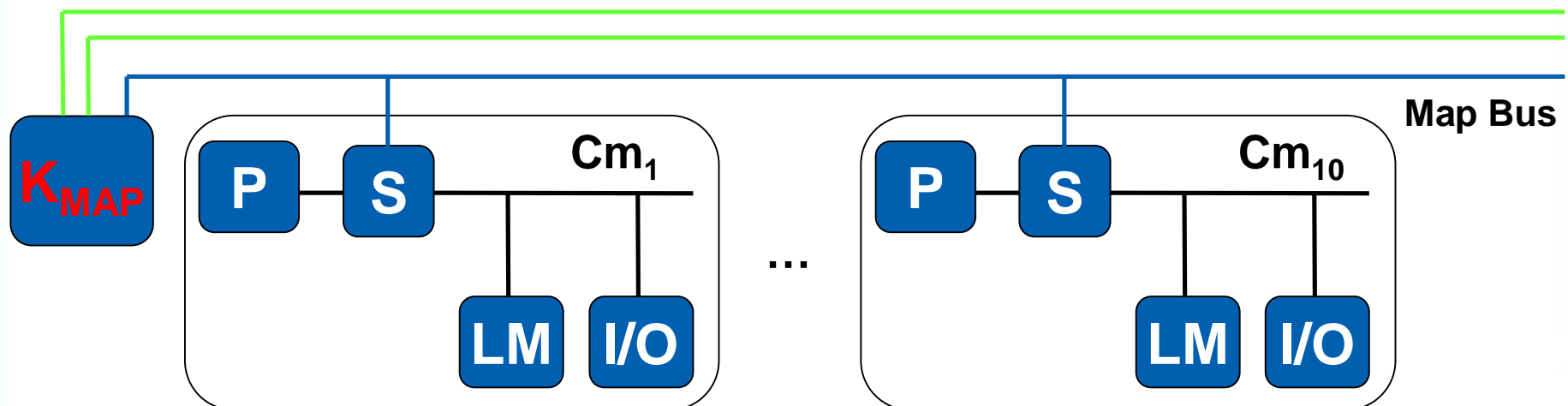
- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Arhitectura Cm\*

3

- Structura Cm\* conecteaza mai multe module individuale → structura multipla de calculatoare == cluster
- $K_{MAP}$  = gestioneaza magistrala Map Bus si cererile pentru Cm-uri
- Un cluster e format din mai multe Cm-uri,  $K_{MAP}$  & Map Bus
- Performantele sunt determinate de interconectarea clusterelor & viteza de comunicatie a procesoarelor din  $K_{MAP}$   
Intercluster Bus





# Accesul la Resursele Locale

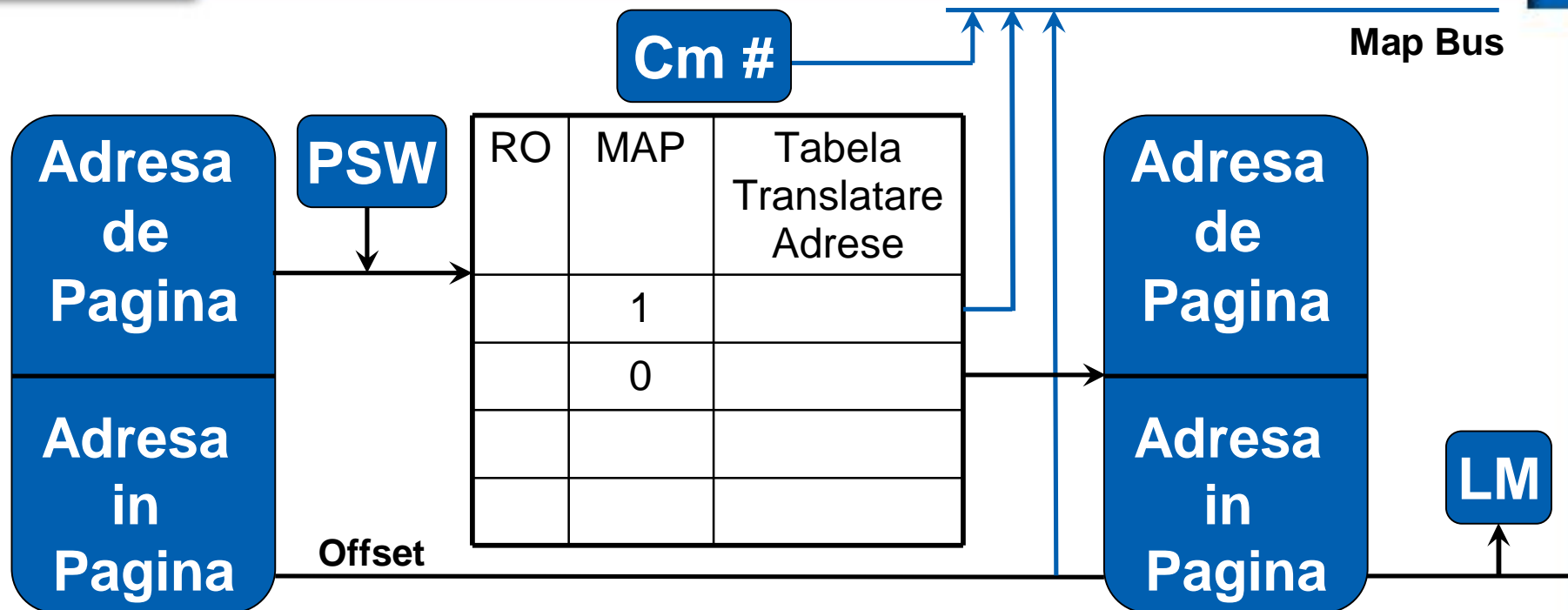
4

- Functiile switch-ului S dintr-o structura Cm:
  - Primeste si interpreteaza cererile de acces de la P-uri locale si externe la memoria locala si la sistemul I/O
  - S permite unui P local sa acceseze resurse externe Cm
- Pentru a realiza interpretarea cererilor locale si externe S-ul asigura:
  - O translatie a adreselor la nivel local
  - Formarea adresei externe prin concatenarea adresei in cadrul paginii si numarul de identificare al procesorului
- Memoria este paginata cu dimensiunea paginii de 4K, cu 16 pagini in total → 256K
- Spatiul total de adresare este de 28 de biti (256M)



# Mecanismul de Adresare pentru Referinte la Memoria Locala

5



- PSW = Processor Status Word – ofera adresa in tabela de translatate (adresa fizica/virtuala)
- RO = Read Only bit (0 = RO; 1 = RW)
- MAP = Maparea adreselor (0 = adresa locala; 1 = adresa globala)
- Cm # = in ce Cm se afla adresa externa;  $K_{MAP}$  interpreteaza si decide daca adresa este in cluster-ul curent sau inafara lui



- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Funcțiile $K_{MAP}$

7

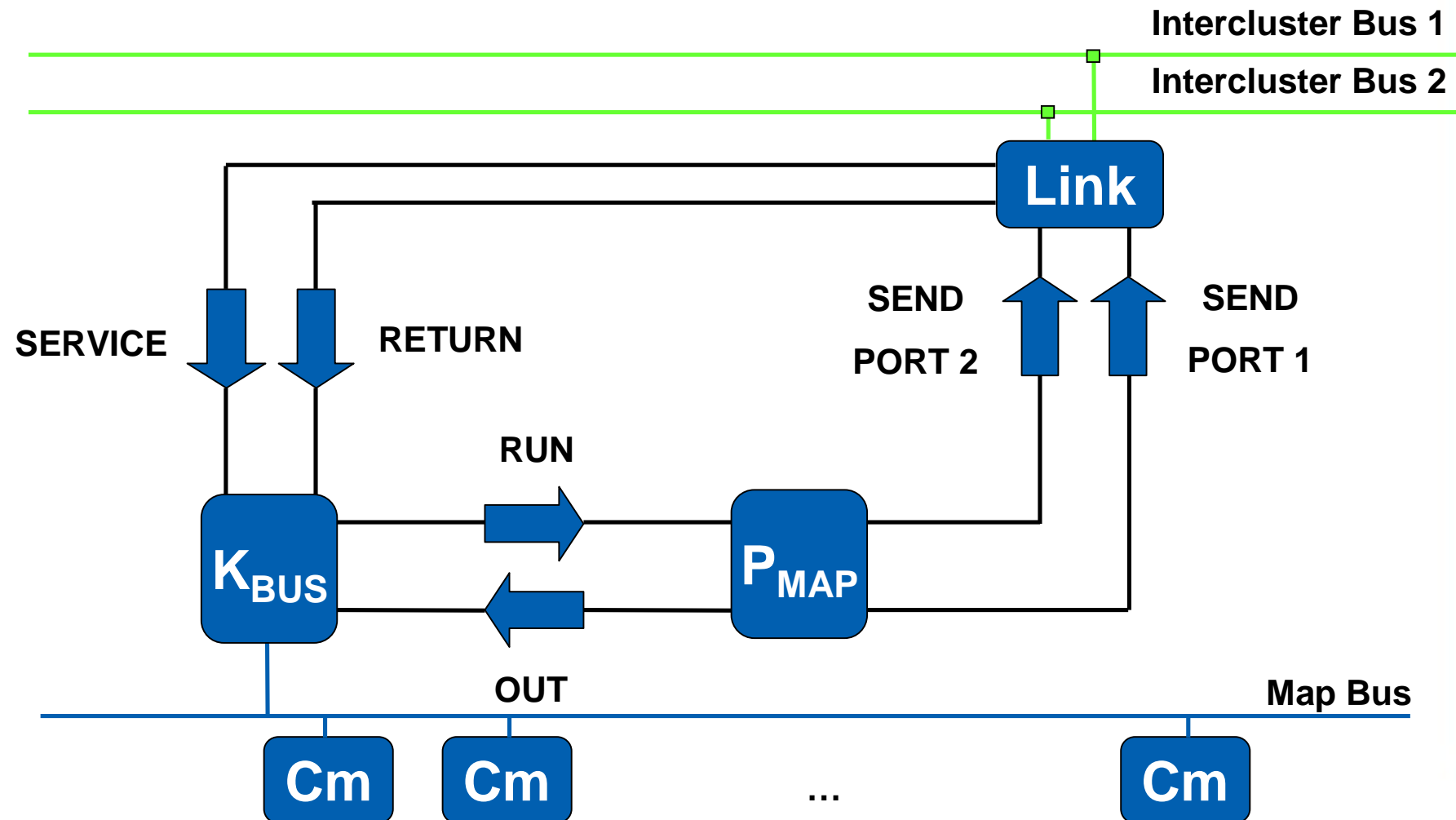
- $K_{MAP}$  asigura:
  - Controlul in cadrul unui cluster – prin excluderea mutuala a accesului la MapBus (arbitrare intre Cm-urile din acelasi cluster)
  - Comunicarea intre clustere prin Intercluster Bus
- Funcțiile de baza ale  $K_{MAP}$ :
  - Mapare de adrese
  - Comunicare
  - Sincronizare
- O serie din primitivele OS sunt implementate in  $K_{MAP}$
- Nucleul OS din Cm se simplifica → transparenta



# Structura $K_{MAP}$

8

- $K_{MAP}$  este compus din 3 procesoare:  $P_{MAP}$ , Link,  $K_{BUS}$







## Elementele $K_{MAP}$

9

- $K_{BUS}$ : unitate de comanda a Map Bus si va controla toate tranzactiile de pe aceasta magistrala
- $P_{MAP}$ : procesor de mapare care face toate translatarile de adresa (maxim 8 procese simultan)
- Link: este cel care gestioneaza si supravegheaza transmiterea mesajelor intercluster (send/recv)
- Exista doua tipuri de comutatie:
  - De circuite – in cadrul cluster-ului
  - De mesaje – intre cluster
- $K_{MAP}$  prelucreaza tranzactii (contexte) concurente
- Fiecare context are asociate registre speciale si registre de legatura la subrutine pt translatarea adreselor



- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Comunicatia $K_{BUS}$ – $P_{MAP}$

11

- $K_{BUS}$  mentine starea a 8 contexte si le aloca cereri de serviciu (fiecare va fi rezident in  $P_{MAP}$ )
- Numarul contextului si alte informatii sunt plasate in coada RUN
- Din RUN sunt preluate de  $P_{MAP}$  care activeaza contextul si eventual translateaza adresa si o plaseaza in OUT
- In acest timp  $K_{BUS}$  poate citi alte cereri sau efectua alte functii cerute de  $P_{MAP}$
- In  $P_{MAP}$  se executa o schimbare de context prin intermediul celor doua cozi (RUN/OUT)



## Comunicatia $K_{BUS}$ – $P_{MAP}$

12

- $K_{BUS}$  preia acum cererea din OUT si transfera adresa si datele la S-ul local al destinatiei; apoi se ocupa de alte cereri
- Cand accesul la memorie este terminat,  $K_{BUS}$  citeste confirmarea de terminare sau citeste datele si plaseaza iara contextul in RUN, reactivandu-l
- $K_{BUS}$  trimite apoi confirmarea la procesul ce a initiat cererea de serviciu
- Contextul poate fi utilizat acum pentru o noua cerere de serviciu



- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Comunicatia Link – $P_{MAP}$

14

- $P_{MAP}$  trimite in SEND o cerere de transmitere mesaj inter-cluster
- Se suspenda contextul initiator al mesajului intercluster & il trece intr-o coada de asteptare
- La primirea raspunsului pentru acest context se pune in RETURN
- Link reactiveaza contextul in RUN
- La primirea raspunsului inter-cluster in Link acesta il plaseaza in SERVICE
- Se emite spre  $K_{BUS}$  o cerere de alocare a unui nou context in RUN



- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Comunicatia $K_{BUS}$ – Map Bus

16

- $K_{BUS}$  trebuie sa execute o mare varietate de task-uri printre care si tratarea inteligenta a erorilor
- $K_{BUS}$  este un procesor microprogramat cu o memorie de control ROM de 256 de cuvinte a cate 40 de biti fiecare
- Map Bus este o magistrala de comunicatii de 38 de biti dintre care 20 bidirectionali de adrese/date intre  $K_{BUS}$  si S-urile locale
- $K_{BUS}$  controleaza toate accesele pe Map Bus precum si cozile si registrele pentru comunicarea cu  $P_{MAP}$
- Maparea Map Bus este sincrona cu  $K_{BUS}$





- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Complexitatea Arhitecturii $P_{MAP}$

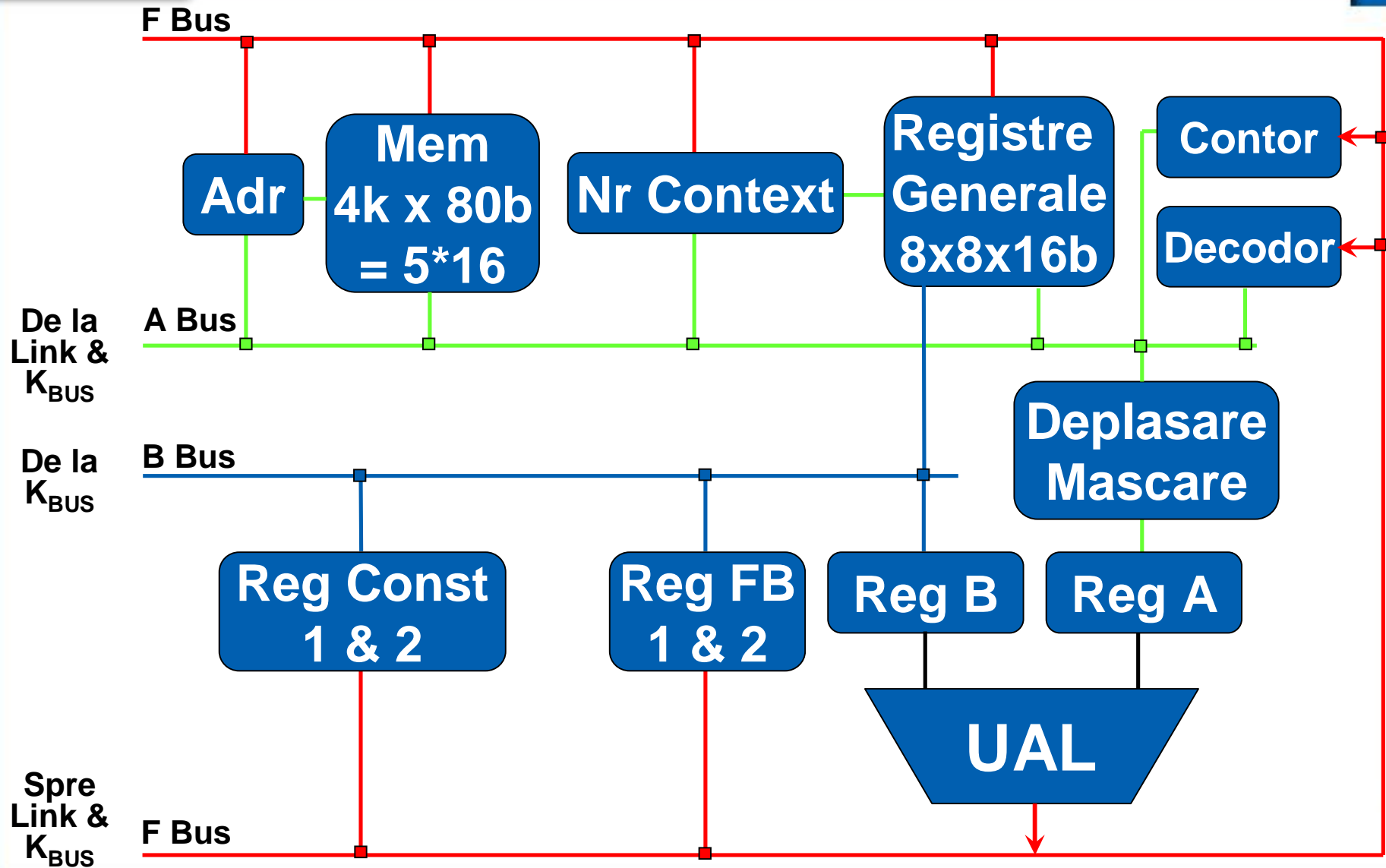
18

- $P_{MAP}$  este un procesor dedicat pe 16 biti, microprogramat, cu codificare orizontala
- Memoria de control este formata din  $4k * 80$  de biti
- Memoria de control RAM foarte rapida are cuvantul de 80 biti ( $5*16$ ) – Writable Control Store
- Functiile fundamentale ale procesorului  $P_{MAP}$  sunt:
  - Translatarea adreselor nelocale din cluster
  - Implementarea primitivelor OS critice din punctul de vedere al timpului de executie
  - Culegerea informatiilor statistice pentru imbunatatirea solutiilor de proiectare



# Arhitectura Procesorului P<sub>MAP</sub>

19





## Componentele Procesorului P<sub>MAP</sub>

20

- In registrele generale (8x128b sau 8x8x16b) este salvat contextul executiei in curs
- Este organizat in jurul a 3 magistrale:
  - A, B pentru operanzi
  - F pentru rezultate
- Pentru a putea mapa adresele P<sub>MAP</sub> trebuie sa aibe acces rapid la informatia pt translatarea adr virtuale in adr fizice
- UAL are un operator la nivel de bit pentru Reg A ce asigura:
  - Deplasarea
  - Rotirea
  - Mascarea (cu 30 de masti predefinite), permitand preluarea descriptorului de context
- Prelucrarea este in banda de asamblare – operatii UAL si operanzii ce vin simultan, separat, prin registrii tampon
- Accesul la o memorie in acelasi cluster este mult mai rapid decat in alt cluster

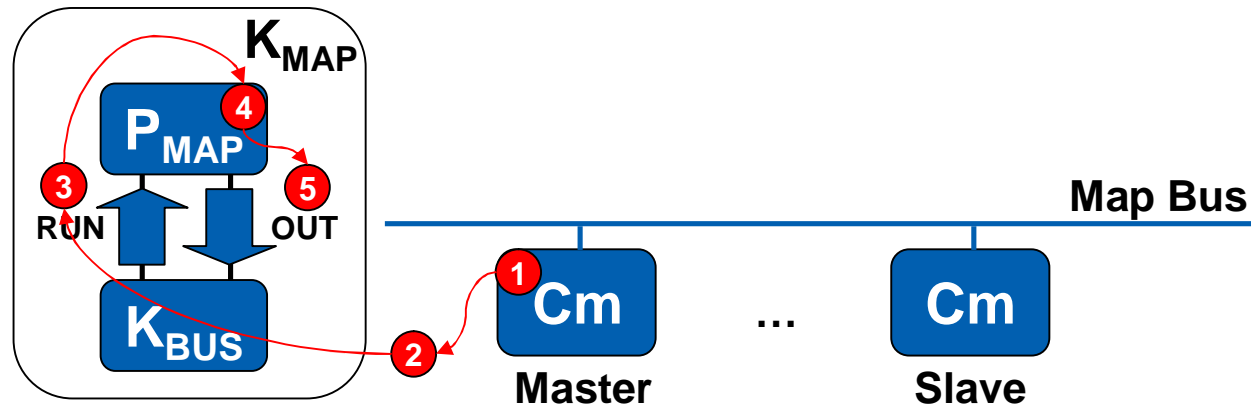


- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Etapele de Acces in Cadrul unui Cluster

22

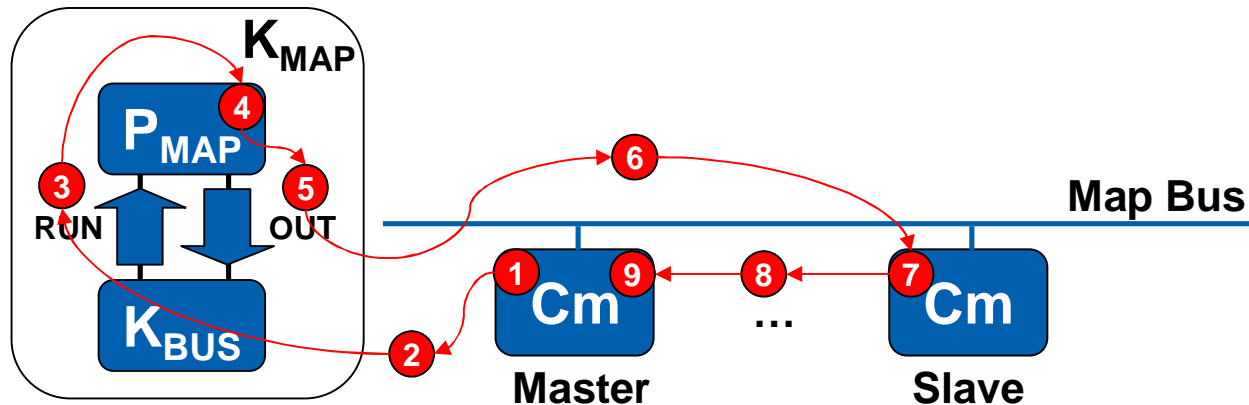


1. In Cm Master se initiaza acces la o memorie nelocala
2. Cm Master emite adresa virtuala catre  $K_{BUS}$
3.  $K_{BUS}$  activeaza un context (crearea structurii de date specifica tranzitiei) ce se pune in coada RUN catre  $P_{MAP}$
4.  $P_{MAP}$  trateaza contextul si face translatarea adresei
5.  $P_{MAP}$  pune in coada OUT o cerere pentru ciclu de memorie pentru Cm Slave din cluster-ul curent



# Etapele de Acces in Cadrul unui Cluster

23



6.  $K_{BUS}$  trimite adresa fizica catre  $C_m$  Slave prin Map Bus
7. S-ul local din  $C_m$  Slave face acces la memoria locala prin furt de ciclu
8.  $K_{BUS}$  "permite" ca rezultatul operatiei de acces la memorie sa fie furnizat catre  $C_m$  Master
9.  $C_m$  Master preia datele, termina operatia in curs si continua executia



- Pachetele trimise au urmatorul format:
  - U/K – User/Kernel
  - R/W – Read/Write
  - Cm # – numarul Cm-ului ce efectueaza transferul
  - Page – utilizata pentru translatarea locala
  - Offset – In cadrul paginii, adresa relativa
- Memoriile Cm-urilor trebuie sa aiba posibilitatea de dublu acces



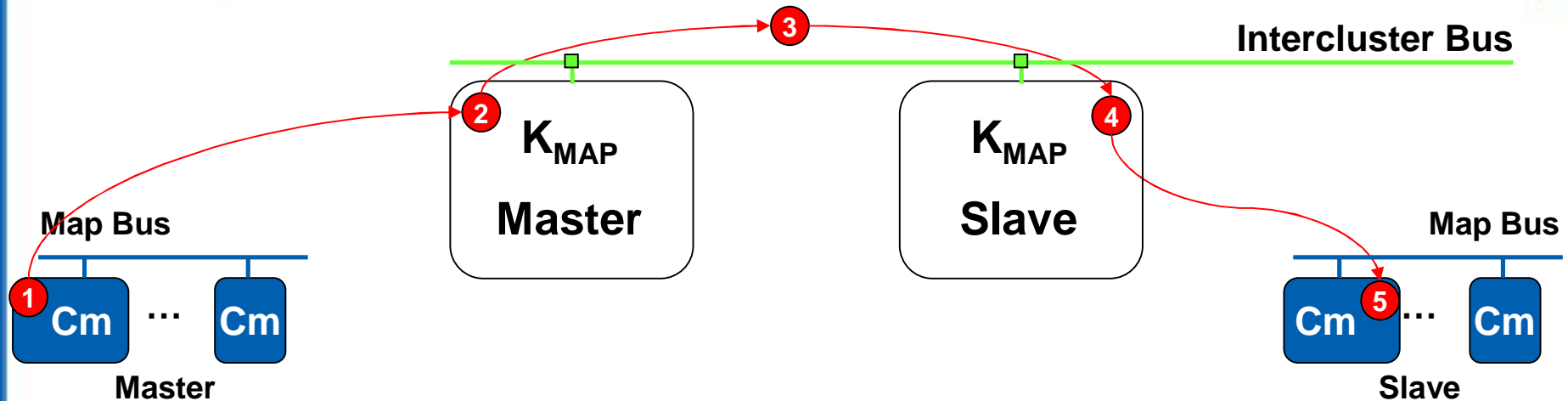


- Accesul la resursele locale – Switch-ul S
- Structura si functiile  $K_{MAP}$ 
  - Comunicatia  $K_{BUS} - P_{MAP}$
  - Comunicatia Link –  $P_{MAP}$
  - Comunicatia  $K_{BUS} - \text{Map Bus}$
- Arhitectura procesorului de mapare  $P_{MAP}$
- Comunicarea in cadrul unui cluster
- Comunicarea intercluster



# Comunicarea Intercluster

26

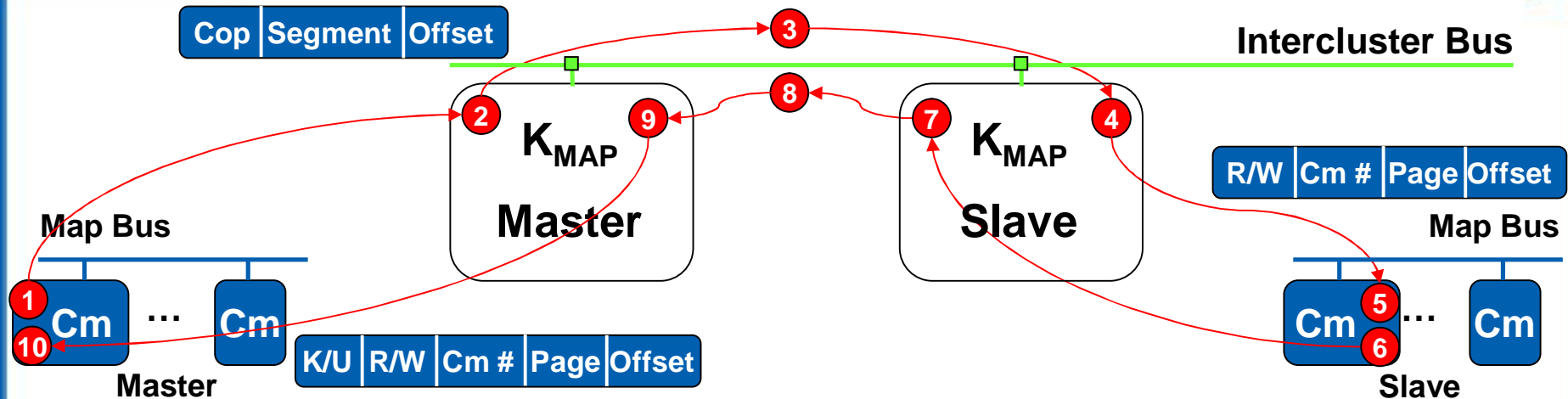


1. Cm Master trimite o cerere de transfer la  $K_{MAP}$  Master
2.  $K_{MAP}$  Master pregateste mesaj/pachet intercluster si codifica cererea
3. Mesajul intercluster e transmis pe Intercluster Bus pe baza algoritmilor de rutare
4.  $K_{MAP}$  Slave decodifica cererea primita si o trimite catre Clusterul sau local
5. Cererea de ciclu de memorie este trimisa la Cm Slave



# Comunicarea Intercluster

27



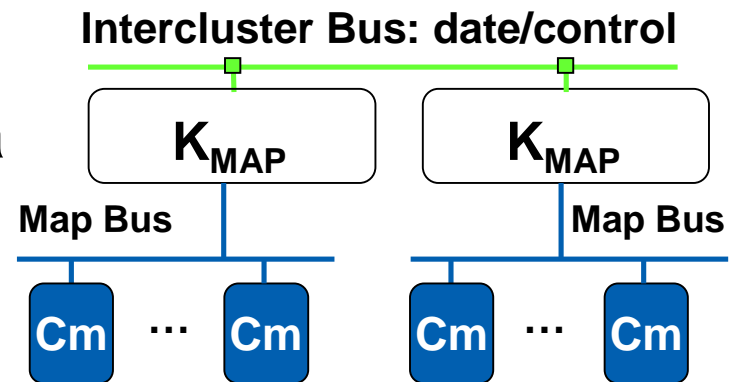
6. Cm Slave transmite rezultatul catre  $K_{MAP}$  Slave
7.  $K_{MAP}$  Slave pregateste mesajul intercluster (reactivarea contextului respectiv)
8.  $K_{MAP}$  Slave transmite rezultatul catre  $K_{MAP}$  Master
9.  $K_{MAP}$  Master receptioneaza si interpreteaza mesajul primit
10. Rezultatul este trimis catre Cm Master



# Protocolul Magistralei Intercluster

28

- Magistrala intercluster
  - Contine linii de date si control
  - **Nu** contine linii de adresa pt ca se realizeaza comutare de pachete si nu de circuite



- Comunicarea este:
  - Asicrona
  - De tip intrebare-raspuns (Q & A)
  - Cu interblocare completa – pentru fiecare actiune se primeste raspuns si abia apoi se anuleaza comanda
- Mesajele intercluster sunt formate din 1-8 cuvinte
- Cele mai utilizate: mesajele **directe** si de **raspuns**



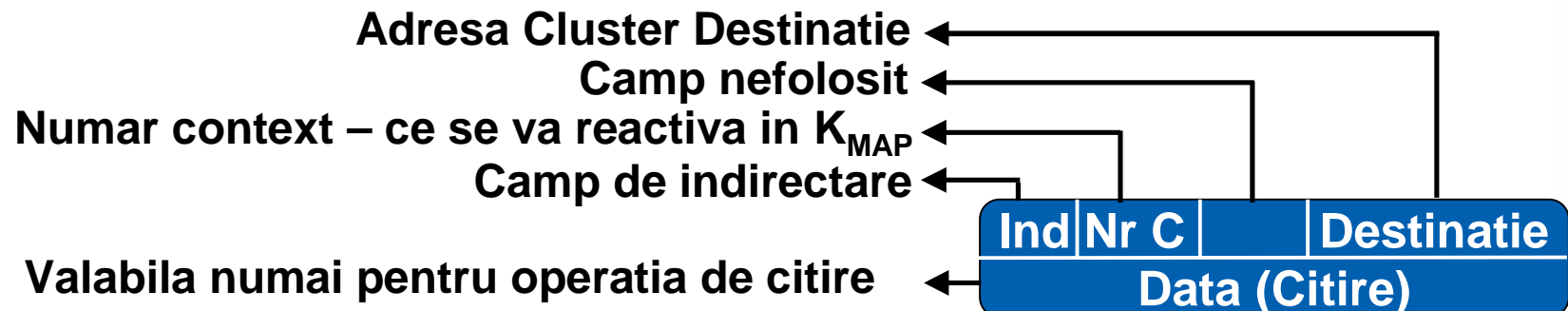
# Formatul Mesajelor Intercluster

29

- Mesajul Direct – format din 4 cuvinte



- Mesajul de Raspuns – format din 2 cuvinte





# What Next?

30

- Q & A?
- Next time:
  - Benchmarking – nevoia de a compara sisteme de calcul
  - Benchmark-uri pentru masini seriale
  - HPCC: HPC Challenge Benchmark
    - Motivatie
    - Prezentarea componentelor software
    - Grafice & Date