UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

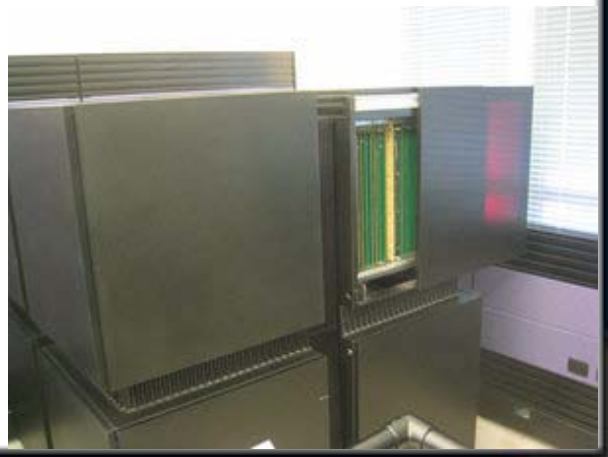
Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Arhitectura Sistemelor de Calcul

14. Descrierea sistemelor reprezentative din clasa SIMD

Exemple de Masini SIMD

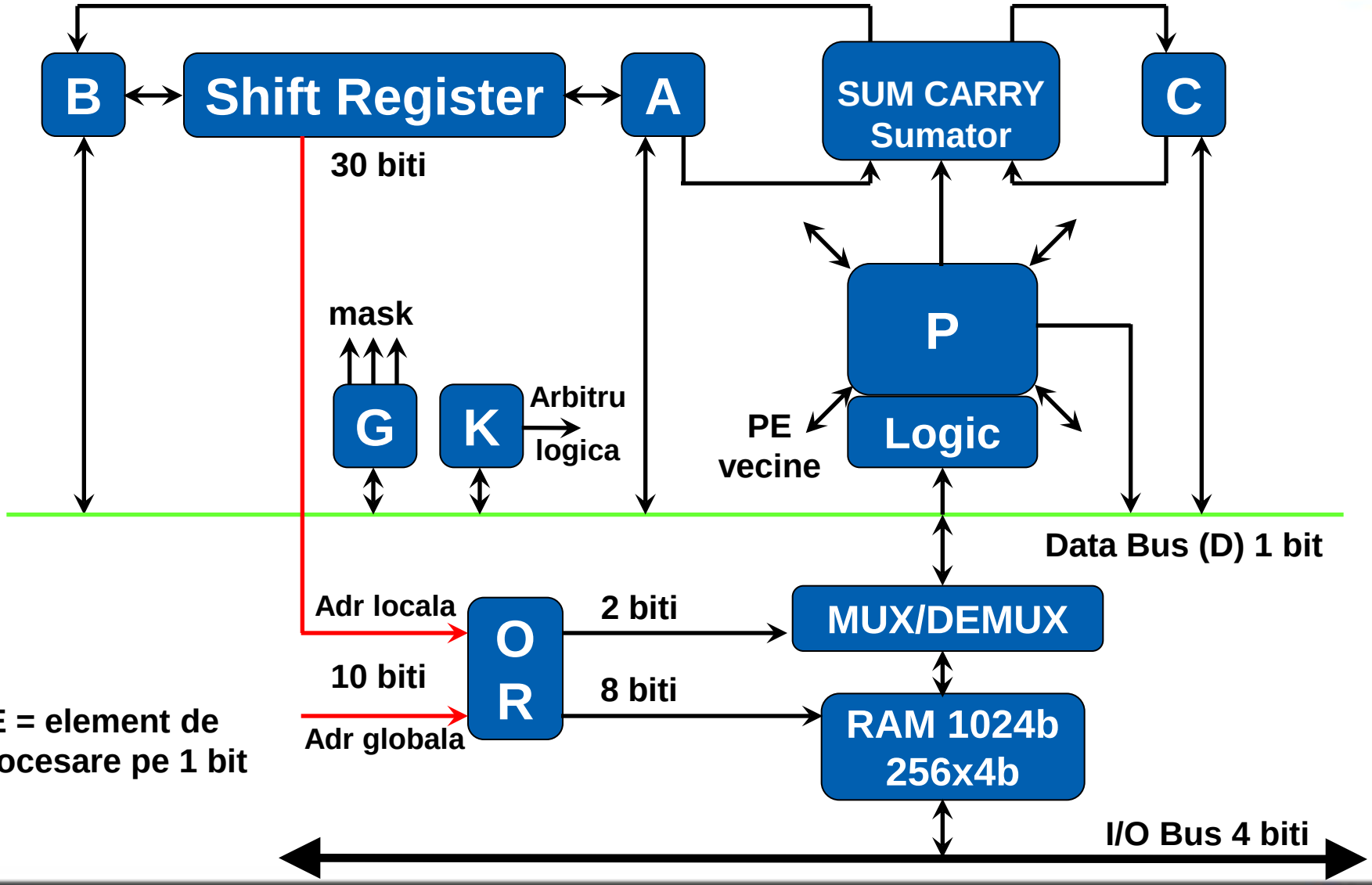
- Before SIMD: Digital Signal Processors (DSP) – masini dedicate cu set propriu de instructiuni si aplicabile unui anume tip de date (Audio/Video)
- The 70's: Vector supercomputers – Cray machines
- The 80's:
  - Massively Parallel Processor (MPP); mai mult de 10.000 de procesoare: NASA/GSFC
  - Connection Machine – MIT: CM-1 & CM-2
- The 90's:
  - Masina Blitzen
  - Intel: MMX, SSE, SSE2 & SSE3
  - AMD: 3DNow!



Cuprins

- Masini SIMD
- Masina Blitzen:
  - Arhitectura
  - Formatul Instructiunilor
  - Exemple
- Masina Blitzen vs. MPP

Arhitectura Masinii Blitzen – PE



Elementele PE

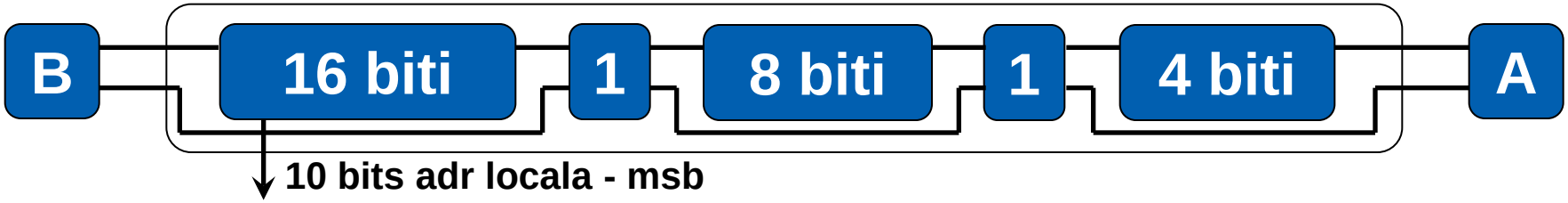
- P – registru pe 1 bit
  - Operanzi
  - Operatori in functiile aritmetice
  - Utilizat la rutare: primeste ops de la PE-uri adiacente
- K – registru de control aritmetico-logic
  - Faciliteaza aparitia ops aritmetice inverse (+/-)
  - Algoritmi de impartiri (non-returning-division)
- G – registru de mascare
  - Se indica daca operatia se executa sau nu
- A & B – registru pe 1 bit
  - Preiau rezultatele de la sumator
  - Contribuie la suma

Elementele PE

- Sumator
  - ADD: A + P + C → B, C
  - HADD: A + C → B, C

A P C	C B
0 0 0	0 0
0 0 1	0 1
.....	.....
1 1 1	1 1

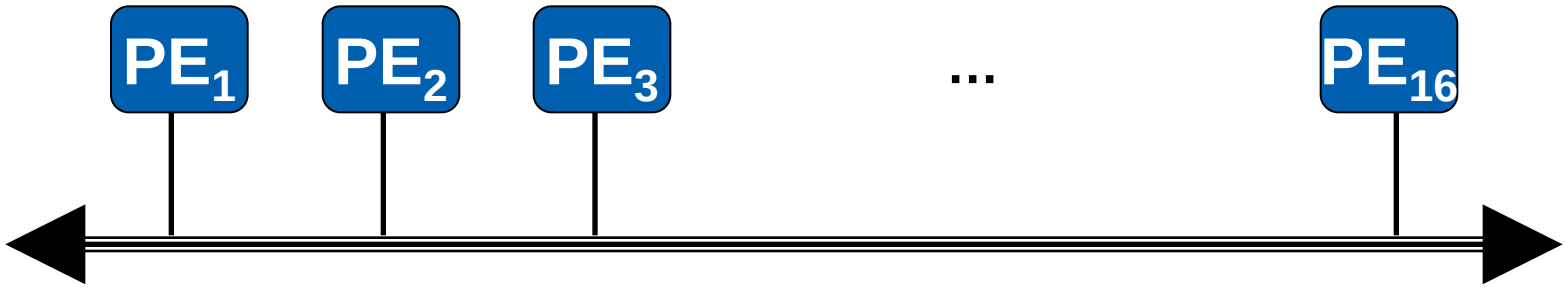
- Shift Register (SR) – Registru de deplasare



  - N stagii = 2, 6, 10, 14, 18, 22, 26, 30
  - 30 biti + A & B = 32 biti
  - Bitul obtinut prin deplasare nu e neaparat salvat
  - La fiecare pas SR se deplaseaza cu o pozitie R/L
  - SR-ul poate fi sters

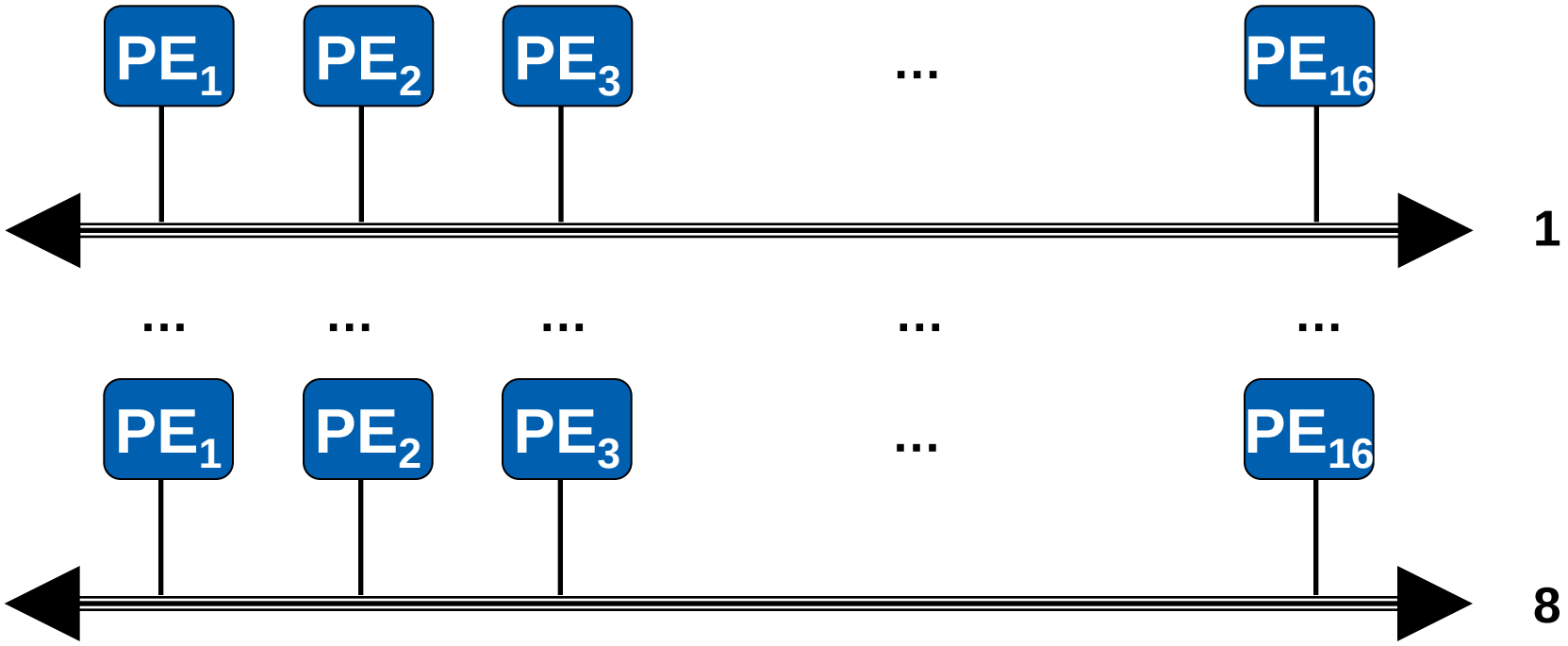
Memoria & I/O Bus

- Accesul la memoria RAM
  - Cu o adresa globala de forma 2^p (p e multiplu de 2)
  - Cu o adresa locala → specific masinii Blitzen (10 biti)
- I/O Bus
  - Formata din 4 biti si utilizata pentru conectarea PE-urilor la resurse externe de stocare a datelor
  - O magistrala I/O este folosita in comun de catre 16 PE-uri

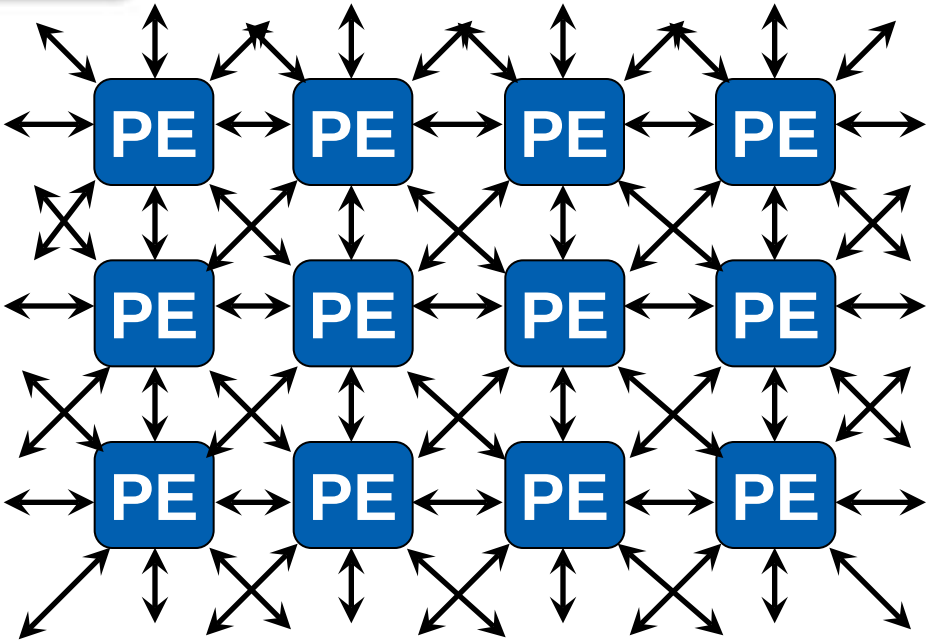


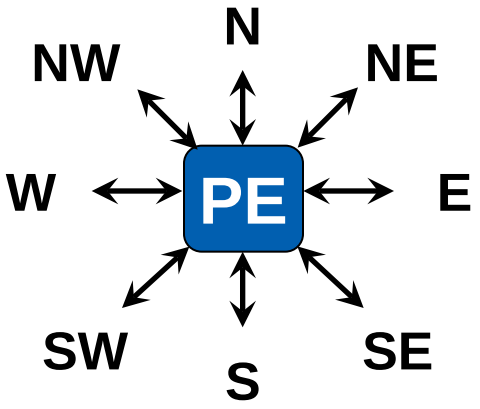
Chip-ul Blitzen

- Un Chip Biltzen este format din
  - 8 magistrale
  - 8 linii a cate 16 procesoare fiecare → 128 PE/chip, fiecare cu cate 1K de RAM
  - Fiecare PE functioneaza la 20MHz cu 8x4=32 biti/ciclu



Un Sistem Blitzen





Structura X-Grid

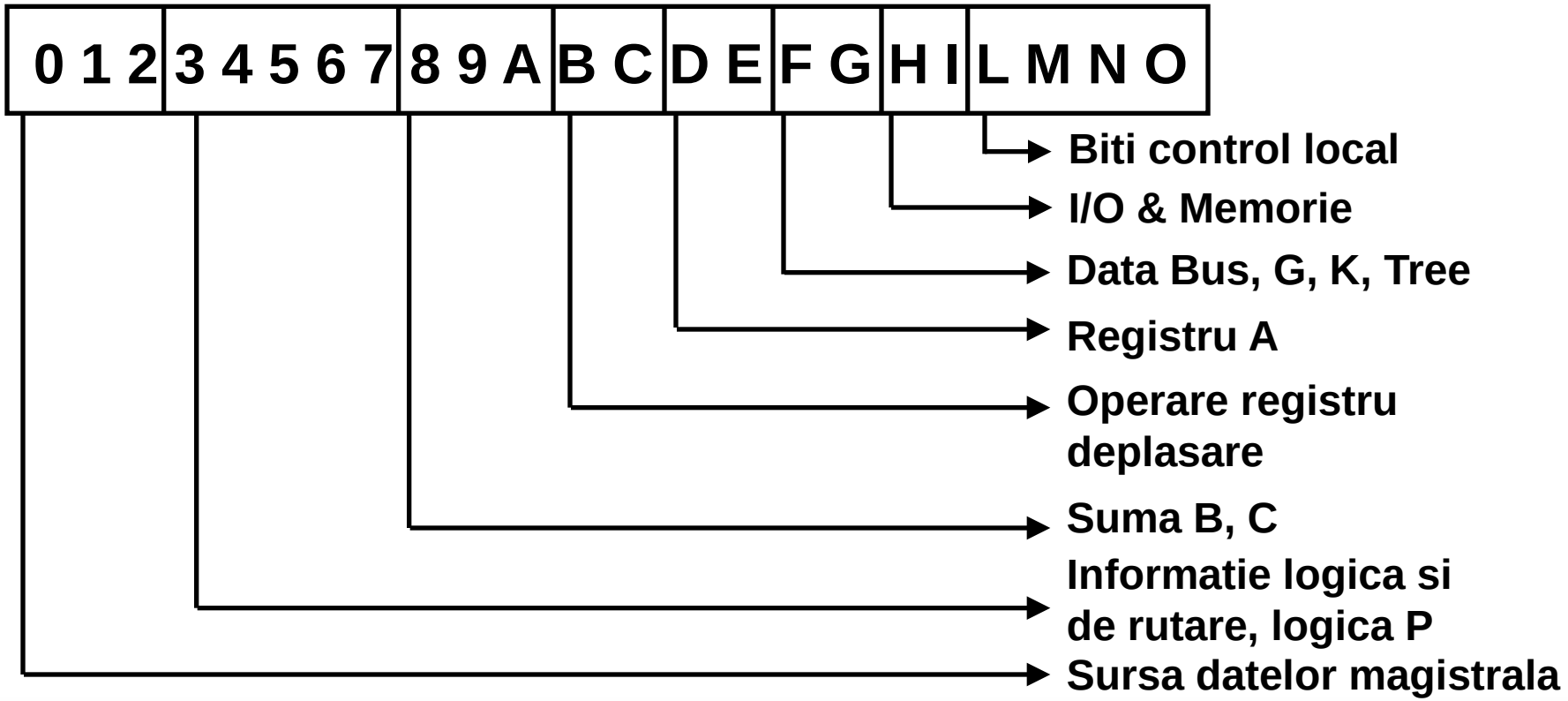
- Sunt 8 directii de rutare
  - N, S, E, W, NE, SE, SW, NW
- Chip-urile Blitzen se grupeaza pe linii de 128 chip-uri → 128 x 128 PE = 16384PE (sistem Blitzen)

Cuprins

- Masini SIMD
- Masina Blitzen:
  - Arhitectura
  - Formatul Instructiunilor
  - Exemple
- Masina Blitzen vs. MPP

Formatul Instructiunilor

- La fiecare ciclu se executa o instructiune intr-un pipeline cu 3 stagii: Decodare, Broadcast, Executie
- Instructiunile sunt pe 23 de biti organizati astfel:



Semnificatia Bitilor Instructiune

- 0, 1, 2: sursa datelor de pe magistrala
  - Registrii A, B, C
  - Registrii G sau K
  - Registrul P
  - Memorie (cand memoria e adresata un read 1b se exec)
- 3, 4, 5, 6, 7: specifica
  - Operatia logica ce trebuie operata in P
  - Operatia de rutare catre unul din cei 8 vecini
  - Configurarea registrului de deplasare (no assigns to P)
- 8, 9, A: operatii executate de sumator
  - Suma & Carry din sumator catre registrii B si C
  - Pot fi utilizate si pentru incarcarea lui B & C

Semnificatia Bitilor Instructiune

- B, C: operatii de deplasare/shiftare
  - Stanga (L)/Dreapta (R)/Stergere (D)
- D, E: operatii cu registrul A
  - Setare directa
  - Setare cu continului registrului de deplasare/shiftare
- F, G: rezultatul e transferat din magistrala de date
  - Catre registrii G, K
  - Catre OR-ul ierarhic (Tree)
- H, I: prelucrarea informatiilor din
  - Memoria locala (R/W operations)
  - I/O Bus (I/O operations)

Semnificatia Bitilor Instructiune

- Bitii de control local – L, M, N, O
- L: operatii mascate & nemascate ale registrului P
  - L = 1: operatiile P mascate; L = 0: operatiile P nemascate
- M: identifica operatii mascate efectuate de memorie, sumator, SR, & registrii A, B sau C
  - M = 1: operatiile mascate; M = 0: operatiile nemascate
- N: poate identifica daca operatiile sunt transmise sau daca a avut loc un broadcast. Este folosit pt:
  - N = 1: op complementate
  - N = 0: operatiile curente
- O: operatii cu
  - O = 0: adresa globala
  - O = 1: cu cei 10 biti din registrul de deplasare (adr locala)

Cuprins

- Masini SIMD
- Masina Blitzen:
  - Arhitectura
  - Formatul Instructiunilor
  - Exemple
- Masina Blitzen vs. MPP

Exemple

- Mai multe instructiuni elementare distincte se pot executa in acelasi timp:
  - ADD + preluare date din memorie + preluare de pe magistrala de date
- Exemple de microinstructiuni:
  - SET_C // C ← 1
  - ADD // A + P + C → B, C
  - MOV_BD // pune B pe magistrala de date D
  - MOV_MD(ADDR) // pune ADDR pe magistrala D
  - MOV_DP // pune in P continutul de pe D
  - ROUTE_E // trimite P catre E si incarca in P din W

Adunarea a doua Numere (8b)


```
#include "blitzen.h"     /*contine setul de instructiuni */
#define XADDR 100
#define YADDR 200
#define WADDR   300
#define NUMBITS 8
main (){
  int i;      /* contor*/
  set_route(GRID);
  load_file("in1.ism",XADDR,XADDR,8);/*citesc X si Y*/
  load_file("in2.ism",YADDR,YADDR,8);
  /* fac suma W = X + Y; valori cum ar fi "X0" fac referinta
   la bitul 0 din X. */
  CLR_C;              /* clear C, A <- x0*/
  MOV_MD(XADDR);
  MOV_DA;
  END;
  MOV_MD(YADDR);     /* P <- Y0*/
  MOV_DP;
  END;
  ADD;                /* adunare */
  END;
```

Adunarea a doua Numere (8b)


```
    for (i=0; i<NUMBITS-1; i++)
    {
      MOV_BD;                        /* W(i) <- B */
      MOV_DM(WADDR+i);
      END;
      MOV_MD(XADDR+1+i);  /* A <- X(i+1) */
      MOV_DA;
      END;
      MOV_MD(YADDR+1+i);  /* P <- Y(i+1) */
      MOV_DP;
      END;
      ADD;                /* adunare*/
      END;
    }
    MOV_BD;          /* W(NUMBITS-1) <- B */
    MOV_DM(WADDR+NUMBITS-1);
    END;
    save_file("sum.osm",WADDR,WADDR,8);  /*salvare rezultat*/
    zyg_end();
}
```

Categorii de Algoritmi

- Algoritmii ce se preteaza la utilizarea masinii Blitzen fac parte din urmatoarele categorii
- 1. Embarrassingly Parallel Algorithms (algoritmi rusinosi paraleli)
  - Fiecare PE actioneaza independent (paralelism maxim);
  - Exemplu: Valoarea de prag (apartenenta la domeniu)
- 2. Near Embarrassingly Parallel Algorithms
  - Un PE are nevoie de informatii pe care le va primi de la alte PE
  - Exemplu: tratarea imaginilor (filtre etc); detectia contururilor

Cuprins

- Masini SIMD
- Masina Blitzen:
  - Arhitectura
  - Formatul Instructiunilor
  - Exemple
- Masina Blitzen vs. MPP

Masina Blitzen vs. MPP

- MPP = Massively Parallel Processor
- Asemanari Blitzen / MPP:
  - Ambele sunt masini SIMD, cu acelasi mod de rutare
  - MPP-urile au > 10k de PE-uri
  - PE-urile lucreaza la nivel de bit
  - Pt B → A avem un registru de shiftare/deplasare
  - Gatuirea majora e datorata transferului de informatii din memoria off-chip catre chip
  - Au acelasi numar de registrii
  - Bitul K de la Blitzen permite o executie a datelor de tip MIMD
  - Blitzen si MPP au utilizari similare

Particularitati Blitzen

- Registrul de Shiftare/Deplasare este bidirectional
- Blitzen-ul are memorie on-chip de 1k (1024 b)
- Este posibila mascarea separata a diverselor tipuri de operatii cu memoria sau cu SR (de deplasare)
- Exista un control local al adresei de memorie (Local Address Modifier)
- Masina Blitzen este mult mai performanta ca MPP