



(12) CERERE DE BREVET DE INVENȚIE

(21) Nr. cerere: a 2013 00812

(22) Data de depozit: 06.11.2013

(41) Data publicării cererii:
30.09.2014 BOPI nr. 9/2014

(71) Solicitant:
• LINEAR ALGEBRA TECHNOLOGIES
LIMITED, 19 MOUNTJOY SQUARE EAST,
DUBLIN, IR

(72) Inventatori:
• MOLONEY DAVID,
42 IONA ROAD, GLASNEVIN, DUBLIN 9, IR;
• RICHMOND RICHARD,
14 SUNBURY AVENUE, BELFAST,
ANTRIM, GB;
• DONOHOE DAVID,
206 TEMPEST DRIVE, STITTSVILLE,
ONTARIO, CA;

• BARRY BRENDAN,
8 LANDSCAPE GARDENS,
CHURCHTOWN, DUBLIN 14, IR;
• BRICK CORMAC,
57 CARNLOUGH ROAD, CABRA,
DUBLIN 7, IE;
• VESA OVIDIU ANDREI,
MARTIRII DE LA FANTANA ALBA STREET,
BLOCK B27, ENTRY B, APARTMENT 9,
TIMIȘOARA, TM, RO

(74) Mandatar:
CABINET DE PROPRIETATE
INDUSTRIALĂ TUDOR ICLĂNZAN,
PIAȚA VICTORIEI NR.5, SC.D, AP.2,
TIMIȘOARA

(54) APARAT, SISTEM ȘI METODĂ PENTRU A REALIZA O
BANDĂ CONFIGURABILĂ ȘI EXTENSIBILĂ DE PROCESARE
DE IMAGINI

(57) Rezumat:

Invenția se referă la un dispozitiv electronic și la o metodă de realizare a unei benzi configurabile și extensibile de procesare de imagini. Dispozitivul electronic, conform invenției, este alcătuit dintr-un dispozitiv (400) de procesare paralelă, cuprinzând: o multitudine de elemente de procesare, fiecare configurat să execute instrucțiuni, un subsistem de memorie, care cuprinde o multitudine de zone de memorie, și un sistem de interconectare, configurat să cupleze multitudine de elemente de procesare și subsistemul de memorie, sistemul de interconectare incluzând o interconectare locală și o interconectare globală, și dintr-un procesor (2602) care comunică apoi cu dispozitivul (400) de procesare paralelă, și care este configurat să ruleze un modul stocat în memorie, care este configurat să recepționeze un grafic de flux de date asociat unui proces de prelucrare a datelor, graficul cuprinzând o multitudine de noduri și o multitudine de arce care conectează două sau mai multe dintre noduri, fiecare nod identificând o operație, și fiecare arc identificând o relație între nodurile conectate, și să alocă un prim nod din multitudine de noduri la un prim element de procesare al dispozitivului de procesare paralelă, iar un

al doilea nod din multitudine de noduri, la un al doilea element de procesare al dispozitivului de procesare paralelă, paralelizând astfel operațiile asociate cu primul nod și cu al doilea nod.

Revendicări: 31
Figuri: 26

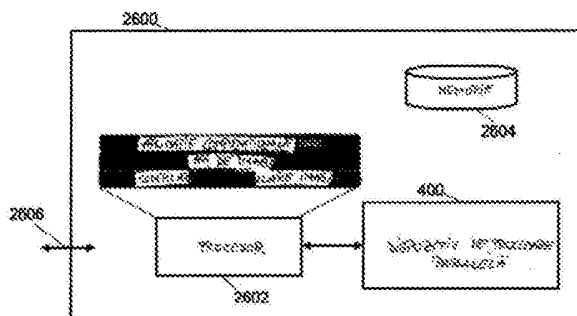


Fig. 26



APARAT, SISTEME ȘI METODE PENTRU A REALIZA
O BANDĂ (SECVENȚĂ ÎN CARE SE POT SUPRAPUNE FAZELE/OPERATIILE)
CONFIGURABILĂ ȘI EXTENSIBILĂ DE PROCESARE DE IMAGINI

Trimitere la cererea conexă

- 5 [0001] Această cerere revendică beneficiul datei prioritare anterioare a cererii de brevet din Regatul Unit nr. GB1314263.3, intitulată „Bandă configurabilă și extensibilă de procesare de imagini,” care a fost depusă în 8 august 2013 de către Linear Algebra Technologies Limited și care este inclusă integral în mod explicit în acest document prin trimitere.

Domeniul de aplicație

- 10 [0002] Această cerere este legată, în general, de dispozitivele de procesare adecvate pentru procesarea de imagini și video.

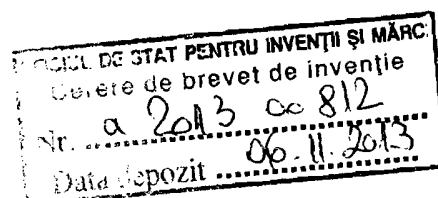
Context

- [0003] Procesarea computațională de imagini și video este foarte solicitantă din punctul de vedere al lățimii de bandă de memorie, deoarece rezoluțiile imaginilor și frecvențele cadrelor sunt ridicate, fiind obișnuite valori agregate de ordinul a multe sute de megapixeli pe secundă. Mai mult, având în vedere că acest domeniu este relativ la început, apar mereu algoritmi noi. De aceea este dificilă implementarea acestora în întregime în hardware, componentele hardware putând fi în imposibilitatea de a se adapta la modificările algoritmilor. În același timp, este nerealistă o abordare de tip software, bazată pe
- 15
- 20 implementarea doar la nivel de procesor. În consecință, este de dorit, în general, o arhitectură/infrastructură flexibilă, care să poată include procesoare și acceleratoare hardware.

- [0004] În același timp, cererea pentru o asemenea procesare de video și de imagini vine, în mare parte, dinspre dispozitivele electronice portabile, cum ar fi tabletele și dispozitivele
- 25
- mobile, în cazul cărora consumul de putere este un considerent-cheie. Prin urmare, există o nevoie generală de o infrastructură flexibilă care să reunească procesoarele multinucleu și acceleratoarele hardware cu un subsistem de memorie cu lățime mare de bandă, care să le permită să asigure o rată de transfer de date susținută în condițiile unui consum de putere redus, potrivit cerințelor pentru dispozitivele electronice portabile.

30 **Rezumat**

- [0005] Conform obiectului prezentat al cererii, se prezintă un aparat, sisteme și metode de realizare a unei benzi configurabile și extensibile de procesare de imagini.



[0006] Obiectul prezentat include un dispozitiv de procesare paralelă. Dispozitivul de procesare include o multitudine elemente de procesare, fiecare configurat să execute instrucțiuni, și un subsistem de memorie care cuprinde o multitudine de zone de memorie, inclusiv o primă zonă de memorie asociată unuia dintre multiplele elemente de procesare.

- 5 Prima zonă de memorie cuprinde o multitudine de blocuri de memorie cu acces aleatoriu (RAM), fiecare având porturi individuale de citire și scriere. Dispozitivul de procesare paralelă poate include un sistem de interconectare configurat pentru a cupla multitudinea de elemente de procesare și subsistemul de memorie. Sistemul de interconectare poate include o interconectare locală configurată să cupleze prima zonă de memorie și unul dintre multiplele
- 10 elemente de procesare, precum și o interconectare globală, configurată să cupleze prima zonă de memorie și restul multitudinii de elemente de procesare.

- [0007] În oricare dintre implementările prezentate aici, unul dintre multiplele blocuri de memorie RAM este asociat unui bloc de arbitrare, acest bloc de arbitrare fiind configurat să primească solicitări de accesare a memoriei din partea unuia dintre multiplele elemente de
- 15 procesare și să permită accesul unuia dintre multiplele elemente de procesare la unul dintre multiplele blocuri de memorie RAM.

[0008] În oricare implementare prezentată aici, blocul de arbitrare este configurat să acorde acces la unul dintre multiplele blocuri de memorie RAM conform unui algoritm round robin (coadă circulară de priorități).

- 20 [0009] În oricare dintre implementările descrise aici, blocul de arbitrare cuprinde un detector de conflicte configurat să monitorizeze solicitările de accesare a memoriei vizând unul dintre multiplele blocuri de memorie RAM și să determine dacă există două sau mai multe dintre multiplele elemente de procesare care încearcă să acceseze simultan același bloc dintre multiplele blocuri de memorie RAM.

- 25 [0010] În oricare dintre implementările prezentate aici, detectorul de conflicte este cuplat la o multitudine de decodificatoare de adrese, fiecare dintre aceste multiple decodificatoare de adrese fiind cuplat la unul dintre multiplele elemente de procesare și fiind configurat să determine dacă unul dintre multiplele elemente de procesare încearcă să acceseze unul dintre multiplele blocuri de memorie RAM asociate blocului de arbitrare.

- 30 [0011] În oricare dintre implementările prezentate aici, multitudinea de elemente de procesare cuprinde cel puțin un procesor vectorial și cel puțin un accelerator hardware.

[0012] În oricare dintre implementările prezentate aici, dispozitivul de procesare paralelă include o multitudine de controlere ale zonelor de memorie, fiecare configurat să asigure accesul la una dintre multiplele zone de memorie.

5 [0013] În oricare dintre implementările prezentate aici, sistemul de interconectare cuprinde o primă magistrală configurată să asigure comunicarea dintre cel puțin un procesor vectorial și subsistemul de memorie.

[0014] În oricare dintre implementările prezentate aici, sistemul de interconectare cuprinde un al doilea sistem de magistrală, configurat să asigure comunicarea dintre cel puțin un accelerator hardware și subsistemul de memorie.

10 [0015] În oricare dintre implementările prezentate aici, al doilea sistem de magistrală cuprinde un filtru de solicitări de adrese ale zonelor de memorie configurat să medieze comunicarea dintre cel puțin un accelerator hardware și subsistemul de memorie, recepționând o solicitare de accesare a memoriei din partea cel puțin unui accelerator hardware și permițând cel puțin unui accelerator hardware să acceseze subsistemul de
15 memorie.

[0016] În oricare dintre implementările prezentate aici, unul dintre multiplele dispozitive de procesare cuprinde o memorie-tampon pentru a mări debitul sistemului de memorie, numărul de elemente din cadrul memoriei-tampon fiind mai mare decât numărul de cicluri necesare pentru preluarea datelor din subsistemul de memorie.

20 [0017] Obiectul prezentei cereri include o metodă de operare a unui sistem de procesare paralelă. Metoda include asigurarea unei multitudini de elemente de procesare, inclusiv un prim element de procesare și un al doilea element de procesare, fiecare dintre multiplele elemente de procesare fiind configurat să execute instrucțiuni. De asemenea, metoda include asigurarea unui subsistem de memorie care cuprinde o multitudine de zone de memorie,
25 inclusiv o primă zonă de memorie asociată primului element de procesare, această primă zonă de memorie cuprinzând o multitudine de blocuri de memorie cu acces aleatoriu (RAM), fiecare bloc având porturi individuale de citire și de scriere. În plus, metoda include recepționarea – de către un bloc de arbitrare asociat unuia dintre multiplele blocuri de memorie RAM printr-o interconectare locală a unui sistem de interconectare – unei prime
30 solicitări de accesare a memoriei din partea primului element de procesare. În plus, metoda include trimiterea – de către blocul de arbitrare prin interconectarea globală – unui prim mesaj de autorizare către primul element de procesare, pentru a autoriza accesul primului element de procesare la unul dintre multiplele blocuri de memorie RAM.

[0018] În oricare dintre implementările prezentate aici, metoda mai include recepționarea – de către blocul de arbitrare printr-o interconectare globală a sistemului de interconectare – unei a doua solicitări de accesare a memoriei din partea unui al doilea element de procesare, precum și trimiterea – de către blocul de arbitrare prin interconectarea globală – unui al
5 doilea mesaj de autorizare către al doilea element de procesare pentru a autoriza accesul celui de al doilea element de procesare la unul dintre multiplele blocuri de memorie RAM.

[0019] În oricare dintre implementările prezentate aici, metoda mai include trimiterea – de către blocul de arbitrare – unei multitudini de mesaje de autorizare către multitudinea de elemente de procesare, pentru a autoriza accesul la unul dintre multiplele blocuri de memorie
10 RAM conform unui algoritm round robin.

[0020] În oricare dintre implementările prezentate aici, metoda mai include monitorizarea – de către un detector de conflicte din componența blocului de arbitrare – solicitărilor de accesare a memoriei adresate unuia dintre multiplele blocuri de memorie RAM, precum și
identificarea situației în care două sau mai multe dintre multiplele elemente de procesare
15 încearcă să acceseze simultan același bloc dintre multiplele blocuri de memorie RAM.

[0021] În oricare dintre implementările prezentate aici, multitudinea de elemente de procesare cuprinde cel puțin un procesor vectorial și cel puțin un accelerator hardware.

[0022] În oricare dintre implementările prezentate aici, metoda mai include asigurarea unei multitudini de controlere ale zonelor de memorie, fiecare controler fiind configurat să
20 asigure accesul la una dintre multiplele zone de memorie.

[0023] În oricare dintre implementările prezentate aici, metoda mai include asigurarea comunicării dintre cel puțin un procesor vectorial și subsistemul de memorie printr-un prim sistem de magistrală al sistemului de interconectare.

[0024] În oricare dintre implementările prezentate aici, metoda mai include asigurarea comunicării dintre cel puțin un accelerator hardware și subsistemul de memorie printr-un al
25 doilea sistem de magistrală al sistemului de interconectare.

[0025] În oricare dintre implementările prezentate aici, al doilea sistem de magistrală cuprinde un filtru de solicitări de adrese ale zonelor de memorie configurat să medieze comunicarea dintre cel puțin un accelerator hardware și subsistemul de memorie,
30 recepționând o solicitare de accesare a memoriei din partea cel puțin unui accelerator hardware și permițând cel puțin unui accelerator hardware să acceseze subsistemul de memorie.

[0026] Obiectul prezentat al cererii include un dispozitiv electronic. Dispozitivul electronic include un dispozitiv de procesare paralelă. Dispozitivul de procesare include o multitudine de elemente de procesare, fiecare configurat să execute instrucțiuni, și un subsistem de memorie care cuprinde o multitudine de zone de memorie, inclusiv o primă zonă de memorie asociată unuia dintre multiplele elemente de procesare. Prima zonă de memorie cuprinde o multitudine de blocuri de memorie cu acces aleatoriu (RAM), fiecare având porturi individuale de citire și scriere. Dispozitivul de procesare paralelă poate include un sistem de interconectare configurat pentru a cupla multitudinea de elemente de procesare și subsistemul de memorie. Sistemul de interconectare poate include o interconectare locală configurată să cupleze prima zonă de memorie și unul dintre multiplele elemente de procesare, precum și o interconectare globală, configurată să cupleze prima zonă de memorie și restul multitudinii de elemente de procesare. De asemenea, dispozitivul electronic include un procesor care comunică cu dispozitivul de procesare paralelă, configurat să ruleze un modul stocat în memorie. Modulul este configurat să recepționeze un grafic de flux de date asociat unui proces de procesare de date, graficul de flux de date cuprinzând multiple noduri și multiple arce care conectează două sau mai multe dintre multiplele noduri, fiecare nod identificând o operație și fiecare arc identificând o relație dintre nodurile conectate; și să aloce un prim nod dintre multiplele noduri la un prim element de procesare al dispozitivului de procesare paralelă, iar un al doilea nod dintre multiplele noduri la un al doilea element de procesare al dispozitivului de procesare paralelă, paralelizând astfel operațiile asociate cu primul nod și cu al doilea nod.

[0027] În oricare dintre implementările prezentate aici, graficul de flux de date este prezentat într-un format de limbaj extensibil de marcare (XML).

[0028] În oricare dintre implementările prezentate aici, modulul este configurat să aloce primul nod din multitudinea de noduri la primul element de procesare pe baza unei performanțe anterioare a unui subsistem de memorie din dispozitivul de procesare paralelă.

[0029] În oricare dintre implementările prezentate aici, subsistemul de memorie al dispozitivului de procesare paralelă cuprinde un contor configurat să contorizeze un număr de conflicte de memorie într-o perioadă de timp predeterminată, iar performanța anterioară a subsistemului de memorie cuprinde numărul de conflicte de memorie măsurat de contor.

[0030] În oricare dintre implementările prezentate aici, modulul este configurat să aloce primul nod din multitudinea de noduri la primul element de procesare în timp ce dispozitivul de procesare paralelă operează cel puțin o porțiune a graficului de flux de date.

[0031] În oricare dintre implementările prezentate aici, modulul este configurat să recepționeze o multitudine de grafice de flux de date și să aloce toate operațiile asociate multitudinii de grafice de flux la un singur element de procesare din dispozitivul de procesare paralelă.

5 [0032] În oricare dintre implementările prezentate aici, modulul este configurat să intercaleze accesările memoriei de către elementele de procesare, pentru a reduce conflictele de memorie.

[0033] În oricare dintre implementările prezentate aici, dispozitivul electronic cuprinde un dispozitiv mobil.

10 [0034] În oricare dintre implementările prezentate aici, graficul de flux de date este specificat folosind o interfață de programare a aplicațiilor (API) asociată dispozitivului de procesare paralelă.

[0035] În oricare dintre implementările prezentate aici, modulul este configurat să furnizeze date de imagine de intrare către multitudinea de elemente de procesare divizând

15 datele de intrare de tip imagine în fâșii și furnizând o fâșie de date de intrare de tip imagine la unul dintre multiplele elemente de procesare.

[0036] În oricare dintre implementările prezentate aici, numărul de fâșii de date de intrare de tip imagine este identic cu numărul de elemente dintre multiplele elemente de procesare.

20 **Descrierea schemelor**

[0037] Prezenta aplicație va fi descrisă în continuare făcând trimitere la scheme.

[0038] FIG. 1 descrie o platformă de procesare computațională de imagini de la Chimera.

[0039] FIG. 2 descrie o arhitectură multinucleu a unui procesor Cell.

[0040] FIG. 3 descrie o arhitectură cu un microprocesor eficient de mică putere (ELM).

25 [0041] FIG. 4 ilustrează un subsistem de memorie perfecționat conform anumitor implementări.

[0042] FIG. 5 ilustrează o secțiune a dispozitivului de procesare paralelă conform anumitor implementări.

[0043] FIG. 6 ilustrează un sistem centralizat de detectare a coliziunilor într-o logică de

30 control pe bloc conform anumitor implementări.

- [0044] FIG. 7 ilustrează un sistem distribuit de detectare a coliziunilor într-o logică de control pe bloc conform anumitor implementări.
- [0045] FIG. 8 prezintă un bloc de arbitrare pentru raportarea unui semnal de coliziune către un solicitant conform anumitor implementări.
- 5 [0046] FIG. 9 ilustrează un bloc de arbitrare orientat pe ciclu conform anumitor implementări.
- [0047] FIG. 10 ilustrează un mecanism de reducere a latenței accesului la memorie datorate arbitrării accesului la memorie conform anumitor implementări.
- [0048] FIG. 11 ilustrează o aplicație de software de planificare conform anumitor
10 implementări.
- [0049] FIG. 12 ilustrează o structură ierarhică a unui sistem care conține un dispozitiv de procesare paralelă conform anumitor implementări.
- [0050] FIG. 13 ilustrează modul în care poate fi folosită descrierea graficului aciclic dirijat (directed acyclic graph – DAG) sau a graficului de flux de date pentru a controla
15 operațiile unui dispozitiv de procesare paralelă conform anumitor implementări.
- [0051] FIG. 14A-14B ilustrează planificarea și emiterea de activități de către compilator și de către planificator conform anumitor implementări.
- [0052] FIG. 15 ilustrează funcționarea unui compilator DAG în timp real conform anumitor implementări.
- 20 [0053] FIG. 16 compară o planificare generată de un planificator OpenCL cu o planificare generată de planificatorul DAG online propus conform anumitor implementări.
- [0054] FIG. 17 ilustrează un mecanism de barieră pentru sincronizarea unei operații a procesoarelor și/sau a acceleratoarelor filtrelor conform anumitor implementări.
- [0055] FIG. 18 ilustrează dispozitivul de procesare paralelă cu diferite tipuri de elemente
25 de procesare conform anumitor implementări.
- [0056] FIG. 19 ilustrează subsistemul de memorie multinucleu propus conform anumitor implementări.
- [0057] FIG. 20 ilustrează o singură zonă a infrastructurii matricei de conectare (connection matrix – CMX) conform anumitor implementări.
- 30 [0058] FIG. 21 ilustrează o arhitectură crossbar (arhitectură de comutare) a controlerelor de memorie pentru acceleratoare (accelerator memory controller – AMC) conform anumitor implementări.

- [0059] FIG. 22 ilustrează un controler AMC cu porturi crossbar conform anumitor implementări.
- [0060] FIG. 23 ilustrează o operație de citire folosind un AMC conform anumitor implementări.
- 5 [0061] FIG. 24 ilustrează o operație de scriere folosind un AMC conform anumitor implementări.
- [0062] FIG. 25 ilustrează dispozitivul de procesare paralelă conform anumitor implementări.
- [0063] FIG. 26 ilustrează un dispozitiv electronic care include un dispozitiv de procesare
- 10 paralelă conform anumitor implementări.

Descriere detaliată

- [0064] Una dintre modalitățile posibile de a interconecta asemenea resurse de procesare diferite (de exemplu, procesoare și acceleratoare hardware) este utilizând o magistrală ca aceea prezentată în motorul computațional pentru fotografii de la Chimera, conceput de
- 15 NVidia. FIG. 1 ilustrează motorul computațional pentru fotografii de la Chimera. Motorul computațional pentru fotografii 100 de la Chimera include multiple nuclee de unitate de procesare grafică (GPU) 102 conectate la un subsistem de procesor ARM multinucleu 104 și la acceleratoare hardware (HW) de procesare a semnalului de imagine (Image Signal Processing – ISP) 106 printr-o infrastructură de magistrală neierarhizată 108 (de exemplu, un
- 20 sistem de magistrală cu o singură ierarhie care conectează toate elementele de procesare). Motorul computațional pentru fotografii de la Chimera este prezentat în general ca fiind un cadru software care abstractizează detaliile nucleelor GPU de la bază 102, ale CPU-urilor 104 și ale blocurilor ISP 106 de la programator. În plus, motorul computațional pentru fotografii 100 de la Chimera descrie fluxul de date prin motorul fotografic computațional ca realizându-
- 25 se prin două magistrale informaționale 108-0, 108-1, prima magistrală 108-0 transportând date de tip imagine sau cadru, iar a doua magistrală 108-1 transportând informații de stare asociate fiecărui cadru.
- [0065] Infrastructura de magistrală neierarhizată, ca în cazul Chimera, poate fi ieftin și convenabil de implementat. Cu toate acestea, infrastructura de magistrală neierarhizată poate
- 30 avea mai multe dezavantaje considerabile dacă se utilizează ca mijloc de a interconecta elemente de procesare eterogene (de exemplu, elemente de procesare de diferite tipuri), cum ar fi nuclee de GPU 102, CPU-uri 104 și blocuri ISP 106. În primul rând, utilizarea unei magistrale pentru a interconecta resurse computaționale presupune posibilitatea de a distribui

memoria în întregul sistem local la fiecare unitate centrală de procesare (CPU) 104, la o unitate de procesare grafică (GPU) 102 și/sau la un bloc de procesor de semnal tip imagine (ISP) 106. Prin urmare, memoria nu se poate alocă flexibil în cadrul benzii de procesare potrivit cerințelor benzii computaționale pentru fotografii pe care dorește programatorul să o

5 implementeze. Această lipsă de flexibilitate poate fie să îngreuneze implementarea anumitor aspecte ale procesării de imagini și video, fie să limiteze o implementare din punctul de vedere al frecvenței cadrelor, al calității imaginii sau din alte puncte de vedere.

[0066] În al doilea rând, utilizarea unei infrastructuri de magistrală neierarhizată poate duce la situația în care diferite resurse computaționale (CPU-urile 104, GPU-urile 102 și

10 blocurile ISP 106) trebuie să concureze pentru lățimea de bandă a magistralei. Această concurență necesită arbitrare, ceea ce reduce lățimea de bandă disponibilă pe magistrală. Prin urmare, scade progresiv lățimea de bandă teoretică disponibilă pentru activitatea propriu-zisă. Ca urmare a reducerii lățimii de bandă, o bandă de procesare poate să nu mai facă față cerințelor de performanță ale aplicației în ceea ce privește frecvența cadrelor, calitatea

15 imaginii și/sau puterea.

[0067] În al treilea rând, lipsa memoriei suficiente în apropierea unei anumite resurse computaționale poate necesita transferul datelor înainte și înapoi între memoria asociată unui anumit GPU 102, CPU 104 sau bloc ISP hardware 106 și o altă resursă computațională. Această indisponibilitate a memoriei poate duce la un consum suplimentar de lățime de bandă

20 a magistralei și la un overhead de arbitrare. Mai mult, indisponibilitatea memoriei face să crească și consumul de putere. De aceea poate fi dificilă sau chiar imposibilă susținerea unui anumit algoritm la o anumită frecvență-țintă a cadrelor.

[0068] În al patrulea rând, utilizarea unei infrastructuri de magistrală neierarhizată poate crea dificultăți la construirea unei benzi de procesare din elemente de procesare eterogene,

25 care pot avea fiecare caracteristici de latență diferite. De exemplu, nucleele de GPU 102 sunt concepute să tolereze latența prin rularea mai multor fire de proces suprapuse pentru a suporta accesări restante multiple ale memoriei (în general memorie DRAM externă) astfel încât să acopere latența, în timp ce CPU-urile 104 obișnuite și blocurile ISP hardware 106 nu sunt concepute să tolereze latența.

30 [0069] Un alt mod de a interconecta diferite resurse de procesare îl găsim la o arhitectură de procesoare Cell concepută de IBM, ilustrată în FIG. 2. Arhitectura de procesoare Cell 200 include o stocare locală (local storage – LS) 202 disponibilă fiecărui procesor 204, cunoscută și ca unitate de execuție sinergetică (synergistic execution unit –

- SXU). Procesorul Cell 200 se bazează pe o infrastructură partajată în timp și pe transferuri ale accesului direct la memorie (direct memory access – DMA) 206 pentru a planifica programatic transferurile de date dintre LS 202 a unui procesor și LS 202 a altui procesor. Dificultatea arhitecturii Cell 200 constă în complexitatea cu care se confruntă
- 5 programatorul în încercarea de a planifica explicit transferurile de date din fundal cu multe sute de cicluri înainte (datorită nivelurilor ridicate de latență din arhitectura Cell 200), astfel încât să asigure disponibilitatea datelor partajate pentru fiecare procesor 204 la momentul necesar. Dacă programatorul nu programează explicit transferurile de date din fundal, procesoarele 204 s-ar putea bloca, fapt ce ar afecta performanțele.
- 10 [0070] Un alt mod de a interconecta diferite resurse de procesare este utilizând un subsistem de memorie multinucleu partajată, pentru partajarea eficientă a datelor între procesoarele unui sistem de procesare multinucleu. Acest subsistem de memorie multinucleu partajată este folosit în sistemul cu microprocesor eficient de mică putere (efficient low-power microprocessor – ELM). FIG. 3 ilustrează sistemul ELM. Sistemul ELM 300 include
- 15 un ansamblu 302, unitatea fizică primară pentru resursele de calcul dintr-un sistem ELM. Ansamblul 302 include un cluster de patru procesoare 304 slab cuplate. Clusterul de patru procesoare 304 partajează resurse locale, cum ar fi o memorie a ansamblului 306 și o interfață cu rețeaua de interconectare. Memoria ansamblului 306 captează instrucțiuni și date în seturi de lucru aproape de procesoare 304, iar bancurile de memorie sunt configurate astfel încât
- 20 memoria să poată fi accesată simultan de procesoarele locale 304 și de controlerul interfeței de rețea. Fiecărui procesor 304 din cadrul unui ansamblu 302 i se alocă un banc de memorie preferat din memoria ansamblului 306. Accesurile unui procesor 304 la bancul său preferat beneficiază de prioritate față de accesurile altor procesoare și ale interfeței de rețea (pe care le vor bloca). Instrucțiunile și datele care aparțin numai unuia dintre procesoare 304 pot fi
- 25 stocate în bancul de memorie preferat al acestuia, pentru a asigura timpi de acces determiniști. Arbitrii care controlează accesul la porturile de citire și de scriere au tendința de a stabili o afinitate între procesoare 304 și bancurile de memorie 306. În acest fel, software-ul poate să estimeze mai sigur lățimea de bandă disponibilă și latența în cazul accesării unor date care pot fi partajate de mai multe procesoare.
- 30 [0071] Cu toate acestea, arhitectura ELM 300 poate avea un consum de putere ridicat din cauza blocurilor de memorie cu acces aleatoriu (RAM) mari din punct de vedere fizic. Mai mult, arhitectura ELM 300 poate suferi de debit redus în cazul în care se partajează multe date între procesoare 304. În plus, nu există prevederi privind partajarea datelor între

procesoare 304 și acceleratoarele hardware, ceea ce poate fi avantajos în anumite cazuri din punctul de vedere al puterii și al performanțelor.

[0072] Cele prezentate aici privesc un aparat, sisteme și metode menite să permită mai multor procesoare și acceleratoare hardware și acceseze date partajate concomitent cu alte
5 procesoare și acceleratoare hardware. Acest document prezintă un aparat, sisteme și metode de a accesa simultan date partajate, fără a fi blocat de un procesor local care are o afinitate ridicată (de exemplu, o prioritate mai ridicată) pentru accesarea stocării locale.

[0073] Aparatul, sistemele și metodele prezentate oferă avantaje substanțiale față de arhitecturile de memorie multinucleu existente. Arhitecturile de memorie multinucleu
10 existente folosesc un singur bloc monolitic de RAM pentru fiecare procesor, ceea ce poate limita lățimea de bandă la care pot fi accesate datele. Arhitectura prezentată poate oferi un mecanism de accesare a memoriei la o lățime de bandă considerabil superioară în comparație cu arhitecturile de memorie multinucleu existente care folosesc un singur bloc monolitic de memorie RAM. Arhitectura prezentată obține această lățime de bandă superioară instanțind
15 mai multe blocuri RAM fizice per procesor, în loc de a instanția un singur bloc RAM mare per procesor. Fiecare bloc RAM poate include un bloc dedicat de arbitraj a accesului și o infrastructură înconjurătoare. Prin urmare, fiecare bloc RAM din subsistemul de memorie poate fi accesat independent de altele de către multiple elemente de procesare din sistem, cum ar fi procesoare vectoriale, procesoare de calcul cu se redus de instrucțiuni (RISC),
20 acceleratoare hardware sau motoare DMA.

[0074] Este oarecum contraintuitiv că utilizarea mai multor instanțe RAM mici este avantajoasă în comparație cu utilizarea unei singure instanțe RAM mari, deoarece un banc de memorie bazat pe o singură instanță RAM mare prezintă o eficiență spațială sporită față de
unul bazat pe multiple instanțe RAM mai mici. Totuși, disiparea de putere pentru instanțele
25 RAM mai mici este, în general, semnificativ redusă în comparație cu a unei singure instanțe RAM mari. Mai mult, dacă o singură instanță RAM fizică mare ar atinge aceeași lățime de bandă ca blocurile RAM cu instanțe multiple, instanța RAM fizică mare ar avea un consum de putere substanțial mai ridicat decât consumul cumulat al instanțelor RAM fizice multiple. Prin urmare, cel puțin din perspectiva disipării puterii, subsistemul de memorie poate avea
30 mai mult de câștigat dacă se utilizează instanțe RAM fizice multiple decât dacă se utilizează o singură instanță RAM mare.

[0075] Subsistemul de memorie cu instanțe RAM fizice multiple poate avea un avantaj sporit prin faptul că, în general, costul per accesare RAM – de exemplu, timpul de accesare a

151

- memoriei sau consumul de putere – este mult mai scăzut în cazul blocurilor RAM mai mici față de blocurile RAM mai mari. Acest fapt se datorează liniilor de biți mai scurte folosite pentru citirea/scrierea datelor din blocurile RAM. În plus, timpul de acces pentru operații de citire și scriere în cazul blocurilor RAM mai mici este și el redus (datorită constantelor de timp reduse ale circuitelor rezistor-condensator (RC) asociate liniilor de biți mai scurte). Prin urmare, elementele de procesare cuplate la subsistemul de memorie cu blocuri RAM multiple pot funcționa la o frecvență mai ridicată, ceea ce reduce puterea statică datorată pierderilor de curent de repaus. Acest fapt poate fi util în special atunci când procesoarele și memoria sunt izolate în domenii de putere. De exemplu, atunci când un anumit procesor sau accelerator de filtru și-a încheiat activitatea, domeniul de putere asociat respectivului procesor sau accelerator de filtru poate fi izolat în mod avantajos. Prin urmare, subsistemul de memorie din arhitectura prezentată are caracteristici superioare din punctul de vedere al lățimii de bandă disponibile și al disipării puterii.
- [0076] În plus, un subsistem de memorie cu instanțe RAM multiple, fiecare cu accesuri arbitrate, poate oferi numeroase moduri de partajare a datelor între procesoare și acceleratoarele hardware, fără a dedica un bloc RAM unui anumit procesor prin blocarea blocului RAM. În principiu, dacă un RAM mai mare este subdivizat în N subblocuri, lățimea de bandă disponibilă pentru date crește aproximativ cu factorul N. Această idee pornește de la ipoteza că datele pot fi partiționate în mod oportun pentru a reduce partajarea concomitentă (de exemplu, un conflict de accesare) de către mai multe elemente de procesare. De exemplu, atunci când un procesor consumator sau un accelerator consumator citește date dintr-o memorie-tampon de date alimentată de un procesor producător sau un accelerator producător, are loc o partajare concomitentă a unei memorii-tampon de date, ceea ce duce la un conflict de acces.
- [0077] În anumite implementări, arhitectura prezentată poate oferi mecanisme de reducere a partajării concomitente a datelor. În particular, arhitectura prezentată se poate preta la reducerea partajării concomitente printr-un mecanism de alocare statică a memoriei și/sau un mecanism de alocare dinamică a memoriei. De exemplu, în mecanismul de alocare statică a memoriei, datele sunt mapate pe diferite porțiuni de memorie înainte de lansarea programului – de exemplu, în faza de compilare a programului –, pentru a reduce partajarea concomitentă a datelor. Pe de altă parte, în modelul de alocare dinamică a memoriei, datele sunt mapate pe diferite porțiuni de memorie în timpul execuției programului. Mecanismul de alocare statică a memoriei asigură un mecanism previzibil de alocare a memoriei pentru date

și nu se caracterizează prin overhead substanțial în ceea ce privește puterea sau performanțele.

- [0078] Tot ca un exemplu, arhitectura prezentată poate fi folosită împreună cu un planificator care rulează pe un controler (de exemplu, un procesor RISC de supervizare) sau
- 5 unul sau mai multe procesoare care mediază accesul la structurile de date partiționate pe mai multe blocuri RAM. Planificatorul poate fi configurat să intercaleze momentele de start ale diferitor elemente de procesare care lucrează asupra unor fragmente (de exemplu, linii sau blocuri) de date (de exemplu, un cadru de imagine), astfel încât să reducă accesul simultan la datele partajate.
- 10 [0079] În anumite implementări, la planificator se poate adăuga un bloc de arbitrare hardware. De exemplu, blocul de arbitrare hardware poate fi configurat să medieze accesurile procesoarelor (cum ar fi procesoarele vectoriale) la memoria partajată printr-o interconectare deterministă partajată, concepută pentru a reduce blocarea procesoarelor. În anumite cazuri, blocul de arbitrare hardware poate fi configurat să efectueze o planificare orientată pe ciclu.
- 15 Planificarea orientată pe ciclu poate include planificarea unei utilizări a unei resurse la granularitate de ciclu de procesor, nu la granularitate de nivel de activitate, care poate necesita cicluri de procesor multiple. Planificarea alocării resurselor la granularitate de ciclu de procesor poate asigura o performanță sporită.
- [0080] În alte implementări, la planificator se poate adăuga o multiplicitate de
- 20 acceleratoare hardware, fiecare dintre acestea putând include o memorie-tampon de intrare și una de ieșire, pentru stocarea datelor. Memoria-tampon de intrare și cea de ieșire pot fi configurate să absoarbă (sau să ascundă) variația întârzierilor care apar în accesarea resurselor externe, cum ar fi memoria externă. Memoria-tampon de intrare și cea de ieșire pot include o memorie-tampon de tip primul intrat, primul ieșit (FIFO), iar memoria-tampon
- 25 FIFO poate include un număr suficient de sloturi pentru a stoca o cantitate suficientă de date și/sau instrucțiuni astfel încât să absoarbă variația întârzierilor care apar în accesarea resurselor externe.
- [0081] În anumite implementări, aparatul, sistemele și metodele prezentate prevăd un dispozitiv de procesare paralelă. Dispozitivul de procesare paralelă poate include o
- 30 multitudine de procesoare, cum ar fi un procesor paralel, fiecare dintre acestea putând executa instrucțiuni. Dispozitivul de procesare paralelă poate să includă, de asemenea, o multitudine de zone de memorie, fiecare zonă de memorie fiind asociată unuia dintre dispozitivele de procesare paralelă și oferind procesorului respectiv acces preferențial în

raport cu alte dispozitive de procesare din dispozitivul de procesare paralelă. Fiecare zonă de memorie poate include o multitudine de blocuri RAM, fiecare bloc RAM putând include un port de citire și un port de scriere. În unele cazuri, fiecare zonă de memorie poate fi prevăzută cu un controler al zonei de memorie, pentru a oferi acces la o zonă de memorie conexă.

5 Procesoarele și blocurile RAM pot fi cuplate unul cu altul printr-o magistrală. În unele cazuri, magistrala poate cupla oricare dintre procesoare cu oricare dintre zonele de memorie. După caz, fiecare bloc RAM poate include o logică de control pe bloc de memorie. Logica de control pe bloc de memorie este denumită uneori logică de control pe bloc sau bloc de arbitrare.

10 [0082] În unele implementări, dispozitivul de procesare paralelă mai poate include cel puțin un accelerator hardware configurat să îndeplinească o funcție de procesare predefinită, cum ar fi procesarea de imagini. În unele cazuri, funcția de procesare predefinită poate include o operație de filtrare.

15 [0083] În unele implementări, cel puțin un accelerator hardware poate fi cuplat cu zonele de memorie printr-o magistrală separată. Magistrala separată poate include un controler de memorie pentru acceleratoare (AMC), configurat să primească solicitări de la cel puțin un accelerator hardware și să permită acceleratorului hardware accesul la o zonă de memorie prin controlerul de zonă de memorie aferent. Se apreciază astfel că traseul de accesare a memoriei aplicat de acceleratoarele hardware poate fi diferit de cel aplicat de procesoarele vectoriale. În unele implementări, cel puțin un accelerator hardware poate include o memorie-tampon internă (de exemplu, o memorie FIFO) pentru a compensa întârzierile survenite la accesarea zonelor de memorie.

25 [0084] În unele implementări, dispozitivul de procesare paralelă poate include un procesor gazdă. Procesorul gazdă poate fi configurat să comunice cu AMC printr-o magistrală gazdă. Dispozitivul de procesare paralelă poate fi prevăzut și cu o interfață de programare pentru aplicații (application-programming interface – API). API asigură o interfață de nivel înalt pentru procesoarele vectoriale și/sau acceleratoarele hardware.

30 [0085] În unele implementări, dispozitivul de procesare paralelă poate funcționa împreună cu un compilator care oferă instrucțiuni pentru dispozitivul de procesare paralelă. În unele cazuri, compilatorul este configurat să ruleze pe un procesor gazdă, care este distinct de elementele de procesare, cum ar fi un procesor vectorial sau un accelerator grafic. În unele cazuri, compilatorul este configurat să recepționeze un grafic de flux de date prin API pentru imagini/video 1206 (FIG. 12), specificând un proces de procesare de imagini. Compilatorul

mai poate fi configurat să mapeze unul sau mai multe aspecte ale graficului de flux de date pe unul sau mai multe elemente de procesare, cum ar fi un procesor vectorial sau un accelerator hardware. În unele implementări, un grafic de flux de date poate include noduri și arce, fiecare nod identificând o operație și fiecare arc identificând o relație dintre noduri (de exemplu, operații), cum ar fi ordinea în care se execută operațiile. Compilatorul poate fi configurat să aloce un nod (de exemplu, o operație) la unul dintre elementele de procesare pentru a paraleliza calcularea graficului de flux de date. În unele implementări, graficul de flux de date poate fi oferit într-un format de limbaj extensibil de marcare (XML). În unele implementări, compilatorul poate fi configurat să aloce mai multe grafice de flux de date la un singur element de procesare.

[0086] În unele implementări, dispozitivul de procesare paralelă poate fi configurat să își măsoare performanțele și să comunice informația către compilator. Prin urmare, compilatorul poate folosi informațiile privind performanțele anterioare primite de la dispozitivul de procesare paralelă, pentru a determina alocarea activităților curente la elementele de procesare din dispozitivul de procesare paralelă. În unele implementări, informațiile privind performanțele pot indica un număr de conflicte de acces survenite la unul sau mai multe elemente de procesare din dispozitivul de procesare.

[0087] În unele cazuri, dispozitivul de procesare paralelă poate fi folosit în aplicații video, ceea ce poate fi costisitor din punct de vedere computațional. Pentru a răspunde cererii computaționale a aplicațiilor video, dispozitivul de procesare paralelă își poate configura subsistemul de memorie astfel încât să reducă conflictele de acces dintre unitățile de procesare în timpul accesării memoriei. În acest scop, după cum s-a discutat mai sus, dispozitivul de procesare paralelă poate subdiviza bancurile de memorie monolitice în mai multe instanțe RAM fizice, în loc de a utiliza bancurile de memorie monolitice ca un singur bloc de memorie fizic. Prin această subdivizare, fiecare instanță RAM fizică poate fi arbitrată pentru operațiile de scriere și de citire, crescând astfel lățimea de bandă disponibilă de atâtea ori, câte instanțe RAM fizice există în bancul de memorie.

[0088] În unele implementări, arbitrarea hardware orientată pe ciclu poate să prevadă, de asemenea, mai multe clase de trafic și măști de planificare programabile. Clasele de trafic multiple și măștile de planificare programabile pot fi controlate folosind planificatorul. Blocul de arbitrare hardware orientat pe ciclu poate include un bloc de arbitrare pentru porturi, care poate fi configurat să aloce o resursă partajată unică la mai mulți solicitanți potrivit unui algoritm round robin. În algoritmul round robin, solicitanții (de exemplu,

elementele de procesare) primesc acces la o resursă (de exemplu, memoria) în ordinea primirii solicitărilor din partea solicitanților. În unele cazuri, blocul de arbitrare pentru porturi poate intensifica algoritmul round robin pentru a ține cont de multiplele clase de trafic.

Resursa partajată unică poate include un bloc RAM, regiștri partajați sau alte resurse pe care

- 5 le pot accesa procesoarele vectoriale, acceleratoarele de filtre și procesoarele RISC pentru a partaja date. În plus, blocul de arbitrare poate permite întreruperea alocării resurselor după algoritmul round robin cu un vector prioritar sau vector cu prioritate maximă. Vectorul prioritar sau vectorul cu prioritate maximă poate fi furnizat de un planificator pentru prioritizarea anumitor clase de trafic (de exemplu, clase de trafic video), în funcție de
- 10 necesitățile aplicației particulare de interes.

[0089] În unele implementări, un element de procesare poate include unul sau mai multe procesoare, cum ar fi un procesor vectorial sau o unitate de procesare vectorială cu arhitectură hibridă pentru streaming, un accelerator hardware și un operator de filtre hardware.

- 15 [0090] FIG. 4 ilustrează un dispozitiv de procesare paralelă cu un subsistem de memorie, care permite mai multor procesoare (de exemplu, unități de procesare vectoriale cu arhitectură hibridă pentru streaming – SHAVE) să partajeze un subsistem de memorie cu porturi multiple conform unor implementări. În particular, FIG. 4 prezintă un dispozitiv de procesare paralelă 400, care este adecvat pentru procesarea de date de tip imagine și video.

- 20 Dispozitivul de procesare 400 cuprinde o multitudine de elemente de procesare 402, cum ar fi un procesor. În configurația exemplificată în FIG. 4, dispozitivul de procesare 400 include 8 procesoare (SHAVE 0 402-0 – SHAVE 7 402-7). Fiecare procesor 402 poate include două unități de citire-scriere 404, 406 (LSU0, LSU1), prin care datele pot fi citite din memorie și scrise în memorie 412. Fiecare procesor 402 poate include, de asemenea, o unitate de

- 25 instrucțiuni 408 în care pot fi încărcate instrucțiuni. O implementare particulară în care procesorul include un SHAVE, acest SHAVE poate include unul sau mai multe procesoare de calcul cu set redus de instrucțiuni (RISC), procesoare de semnal digital (DSP), un procesor de tip cuvânt de instrucțiune foarte lung (very long instruction word – VLIW) și/sau o unitate de procesare grafică (GPU). Memoria 412 cuprinde o multitudine de zone de memorie 412-0 ...

- 30 412-7 denumite aici zone de matrice de conectare (CMX). Fiecare zonă de memorie 412 este asociată unui procesor corespunzător 402-7.

[0091] Dispozitivul de procesare paralelă 400 include, de asemenea, un sistem de interconectare 410 care cuplează procesoarele 402 și zonele de memorie 412. Sistemul de

interconectare 410 este denumit aici interconectare inter-SHAVE (ISI). ISI poate include o magistrală prin care procesoarele 402 să poată citi sau scrie date în orice componentă a oricărei zone de memorie 412.

[0092] FIG. 5 ilustrează o secțiune a dispozitivului de procesare paralelă conform anumitor implementări. Secțiunea 500 include un procesor unic 402-N, o zonă de memorie 412-N asociată procesorului unic 402-N, ISI 410 care cuplează procesorul unic 402-N și alte zone de memorie (nereprezentate în figură) și o logică de control pe bloc 506 pentru arbitrarea comunicării dintre un bloc din zona de memorie 412-N și procesoare 402. Conform ilustrației din secțiunea 500, procesorul 402-N poate fi configurat să acceseze direct zona de memorie 412-N asociată procesorului 402-N; procesorul 402-N poate accesa și alte zone de memorie (nereprezentate în figură) prin ISI.

[0093] În anumite implementări, fiecare zonă de memorie 412-N poate include o multitudine de blocuri RAM sau blocuri RAM fizice 502-0 ... 502-N. De exemplu, o zonă de memorie 412-N cu capacitatea de 128 kB poate include patru blocuri de memorie de 32 kB cu un singur port (de exemplu, elemente RAM fizice) organizate sub forma a 4 000 de cuvinte de 32 de biți. În unele implementări, un bloc 502 poate fi denumit și bloc RAM logic. În unele implementări, un bloc 502 poate include o memorie RAM complementară metal-oxid-semiconductor (CMOS) cu un singur port. Avantajul unei memorii RAM cu port unic este că ea este disponibilă în general în majoritatea proceselor care țin de semiconductoare. În alte implementări, un bloc 502 poate include o memorie RAM CMOS cu mai multe porturi.

[0094] În unele implementări, fiecare bloc 502 poate fi asociat unei logici de control pe bloc 506. Logica de control pe bloc 506 este configurată să primească solicitări de la procesoare 402 și să permită accesul la porturile individuale de citire și de scriere ale blocului asociat 502. De exemplu, când un element de procesare 402-N dorește să acceseze date într-un bloc RAM 502-0, înainte ca elementul de procesare 402-N să trimită solicitarea pentru date de memorie direct către blocul RAM 502-0, elementul de procesare 402-N poate trimite o solicitare de accesare a memoriei către logica de control pe bloc 506-0 asociată blocului de memorie RAM 502-0. Solicitarea de accesare a memoriei poate să includă o adresă de memorie a datelor solicitate de elementul de procesare 402-N. Ulterior, logica de control pe bloc 506-0 poate analiza solicitarea de accesare a memoriei și poate determina dacă elementul de procesare 402-N poate accesa memoria solicitată. Dacă elementul de procesare 402-N poate accesa memoria solicitată, logica de control pe bloc 506-0 poate trimite un

146

mesaj de permitere a accesului către elementul de procesare 402-N, iar ulterior, elementul de procesare 402-N poate trimite o solicitare de date de memorie către blocul RAM 502-0.

[0095] Deoarece există posibilitatea accesării simultane de către elemente de procesare multiple, în unele implementări, logica de control pe bloc 506 poate include un detector de conflicte, care este configurat să detecteze o situație în care două sau mai multe elemente de procesare, cum ar fi un procesor sau un accelerator, încearcă să acceseze oricare dintre blocurile dintr-o zonă de memorie. Detectorul de conflicte poate monitoriza accesul la fiecare bloc 502 pentru a detecta o încercare de accesare simultană. Detectorul de conflicte poate fi configurat să raporteze planificatorului că s-a produs un conflict de acces care trebuie rezolvat.

[0096] FIG. 6 ilustrează un sistem centralizat de detectare a coliziunilor într-o logică de control pe bloc conform anumitor implementări. Sistemul de detectare a conflictelor poate include un bloc de arbitraj centralizat 608, care include o multitudine de detectoare de conflicte 604 și o multitudine de codificatoare de adresă „one-hot” (cu un bit activ) 602. În unele implementări, codicatorul de adresă one-hot 602 este configurat să primească o solicitare de accesare a memoriei din partea unuia dintre elementele de procesare 402 și să determine dacă solicitarea de accesare a memoriei este pentru datele stocate în blocul RAM 502 asociat codicatorului de adresă one-hot 602. Fiecare detector de conflicte 604 poate fi cuplat la unul sau mai multe codificatoare de adresă one-hot 602, care sunt cuplate și la unul dintre elementele de procesare 402 care pot accesa blocul 502 asociat detectorului de conflicte 602. În unele implementări, un detector de conflicte 604 poate fi cuplat la toate codificatoarele de adresă one-hot 602 asociate unui bloc RAM particular 502.

[0097] Dacă solicitarea de accesare a memoriei este pentru date stocate în blocul RAM 502 asociat codicatorului de adresă one-hot 602, atunci codicatorul de adresă one-hot 602 poate comunica o valoare de bit „1” către detectorul de conflicte 604 al respectivului bloc RAM dacă solicitarea de accesare a memoriei nu este pentru date stocate în blocul RAM 502 asociat codicatorului de adresă one-hot 602, iar apoi codicatorul de adresă one-hot 602 poate comunica o valoare de bit „0” către detectorul de conflicte 604 al blocului RAM respectiv.

[0098] În unele implementări, codicatorul de adresă one-hot 602 este configurat să determine dacă solicitarea de acces este pentru date stocate în blocul RAM 502 asociat codicatorului de adresă one-hot 602 analizând adresa-țintă a solicitării de acces la memorie. De exemplu, atunci când blocul RAM 502 asociat codicatorului de adresă one-hot 602 este

desemnat cu un interval de adrese de memorie de la 0x0000 la 0x00ff, atunci codicatorul de adresă one-hot 602 poate determina dacă adresa-țintă a solicitării de accesare a memoriei se înscrie în intervalul de la 0x0000 la 0x00ff. Dacă da, solicitarea de accesare a memoriei este pentru date stocate în blocul RAM 502 asociat codicatorului de adresă one-hot 602; dacă

5 nu, solicitarea de accesare a memoriei nu este pentru date stocate în blocul RAM 502 asociat codicatorului de adresă one-hot 602. În unele cazuri, codicatorul de adresă one-hot 602 poate utiliza un bloc de comparare a intervalului pentru a determina dacă adresa-țintă a solicitării de acces la memorie se înscrie în intervalul de adrese asociat unui bloc RAM 502.

[0099] Odată ce detectorul de conflicte 604 primește valori de bit de la toate

10 codificatoarele de adresă one-hot 602, detectorul de conflicte 604 poate număra câți „1” există în valorile de bit primite (de exemplu, poate aduna valorile de bit) pentru a determina dacă există mai mult de un element de procesare 402 care solicită în momentul respectiv accesarea aceluiași bloc RAM 502. Dacă există mai mult de un element de procesare care solicită acces la același bloc RAM 502, detectorul de conflicte 604 poate raporta un conflict.

15 [0100] FIG. 7 ilustrează un sistem distribuit de detectare a coliziunilor într-o logică de control pe bloc conform anumitor implementări. Sistemul distribuit de detectare a conflictelor poate include un arbitru distribuit 702, care include o multitudine de detectoare de conflicte 704. Funcționarea sistemului distribuit de detectare a conflictelor este substanțial similară funcționării sistemului centralizat de detectare a conflictelor. În acest caz, detectoarele de

20 conflicte 704 sunt dispuse distribuit. În particular, arbitrul distribuit 702 poate include detectoare de conflicte 704 care sunt dispuse în serie, fiecare detector de conflicte 704 fiind cuplat la un singur subset de codificatoare de adresă one-hot 602 asociate unui anumit bloc RAM 502. Această dispunere este diferită de sistemul centralizat de detectare a conflictelor, în care un detector de conflicte 704 este cuplat la toate codificatoarele de adresă one-hot 602

25 asociate unui anumit bloc RAM 502.

[0101] De exemplu, atunci când un anumit bloc RAM 502 poate fi accesat de 64 de elemente de procesare 402, un prim detector de conflicte 704-0 poate primi o solicitare de acces la memorie de la 32 de elemente de procesare, iar al doilea detector de conflicte 704-1 poate primi o solicitare de acces la memorie de la celelalte 32 de elemente de procesare.

30 Primul detector de conflicte 704-0 poate fi configurat să analizeze una sau mai multe solicitări de acces la memorie primite de la cele 32 de elemente de procesare cuplate la el și să determine un prim număr de elemente, dintre cele 32 de elemente de procesare cuplate la el, care solicită accesul la un anumit bloc RAM 502-0. În paralel, al doilea detector de

conflicte 704-1 poate fi configurat să analizeze una sau mai multe solicitări de acces la memorie primite de la cele 32 de elemente de procesare cuplate la el și să determine un prim număr de elemente, dintre cele 32 de elemente de procesare cuplate la el, care solicită accesul la respectivul bloc RAM 502-0. Apoi, al doilea detector de conflicte 704 poate aduna primul
5 număr și al doilea număr pentru a determina câte dintre cele 64 de elemente de procesare solicită accesul la respectivul bloc RAM 502-0.

[0102] Odată ce un sistem de detectare a conflictelor detectează un conflict, sistemul de detectare a conflictelor poate trimite un semnal de stop către un solicitant 402. FIG. 8 prezintă un bloc de arbitrare pentru raportarea unui semnal de coliziune către un solicitant
10 conform anumitor implementări. Particularizând și mai mult, ieșirile blocurilor de comparare a intervalelor din sistemele de detectare a conflictelor sunt combinate folosind o poartă OR pentru a genera un semnal de stop destinat solicitantului. Jumătatea de semnal arată că există mai mult de un element de procesare care încearcă să acceseze același subbloc RAM fizic din cadrul zonei de memorie asociate solicitantului. La primirea semnalului de stop, solicitantul
15 poate întrerupe operația de accesare a memoriei până la dispariția conflictului. În unele implementări, conflictul poate fi șters de hardware independent de codul programului.

[0103] În unele implementări, blocul de arbitrare poate funcționa la granularitate de ciclu. În asemenea implementări, blocul de arbitrare alocă resurse la granularitate de ciclu de procesor, nu la granularitate de nivel de activitate, aceasta din urmă putând include cicluri de
20 procesor multiple. O asemenea planificare orientată pe ciclu poate spori performanțele sistemului. Blocul de arbitrare poate fi implementat în hardware, astfel încât blocul de arbitrare să poată efectua planificarea orientată pe ciclu în timp real. De exemplu, în orice instanță particulară, blocul de arbitrare implementat în hardware poate fi configurat să aloce resurse pentru următorul ciclu de procesor.

[0104] FIG. 9 ilustrează un bloc de arbitrare orientat pe ciclu conform anumitor implementări. Blocul de arbitrare orientat pe ciclu poate include un bloc de arbitrare pentru
25 porturi 900. Blocul de arbitrare pentru porturi 900 poate include un prim bloc de selectare a porturilor 930 și un al doilea bloc de selectare a porturilor 932. Primul bloc de selectare a porturilor 930 este configurat să determine care dintre solicitările de accesare a memoriei
30 (identificate ca poziție a unui bit din vectorul de solicitare al clientului) este alocată portului [0] al zonei de memorie pentru a accesa o zonă de memorie cuplată la portul [0] al zonei, iar al doilea bloc de selectare 932 este configurat să determine care dintre vectorii de solicitare ai

clientului este alocat portului [1] al zonei pentru accesarea unei zone de memorie cuplate la portul [1] al zonei.

[0105] Primul bloc de selectare a porturilor 930 include un prim detector de „1” inițial (leading one detector – LOD) 902-0 și un al doilea LOD 902-1. Primul LOD 902-0 este configurat să recepționeze un vector de solicitare din partea clientului, care poate include o multitudine de biți. Fiecare bit din vectorul de solicitare din partea clientului indică dacă a fost sau nu primită o solicitare de acces sub formă de mesaj de la un solicitant asociat poziției bitului respectiv. În unele cazuri, vectorul de solicitare din partea clientului funcționează în modul „activ pe nivel logic 1”. Odată ce primul LOD 902-0 primește vectorul de solicitare de la client, primul LOD 902-0 este configurat să detecteze o poziție de bit, numărând de la stânga la dreapta, la care solicitarea devine pentru prima dată diferită de zero, identificând astfel prima solicitare de accesare a memoriei, numărând de la stânga la dreapta, către primul bloc de selectare a porturilor 930. În paralel, vectorul de solicitare de la client poate fi mascat de un operator logic ȘI 912 pentru a genera un vector mascat de solicitare de la client generat de un registru de mască 906 și un deplasator de mască spre stânga 904. Registrul de mască 906 poate fi setat de un procesor care comunică cu registrul de mască 906, iar deplasatorul de mască spre stânga 904 poate fi configurat să deplaseze spre stânga masca reprezentată de registrul de mască 906. Al doilea LOD 902-1 poate recepționa vectorul mascat al solicitării clientului de la operatorul logic ȘI 912, detectând apoi primul 1 din vectorul mascat al solicitării clientului.

[0106] Ieșirea de la primul LOD 902-0 și cea de la al doilea LOD 902-1 sunt apoi transmise către blocul de selectare 908 a câștigătorului portului [0]. Blocul de selectare 908 a câștigătorului portului [0] mai primește încă două intrări suplimentare: un vector prioritar și un vector cu prioritate maximă. Blocul de selectare 908 a câștigătorului portului [0] este configurat să determine care dintre solicitările de acces la memorie primite ar trebui alocate portului [0] al zonei de memorie, pe baza priorităților intrărilor. În unele implementări, prioritățile intrărilor pot fi ordonate astfel: începând de la vectorul cu prioritate maximă, continuând cu vectorul prioritar care împarte vectorul LOD mascat în solicitări prioritare și neprioritare, urmat de vectorul LOD nemascat, care are cea mai mică prioritate. În alte implementări se pot specifica alte priorități.

[0107] În timp ce primul bloc de selectare a porturilor 930 poate fi configurat să determine dacă vectorul de solicitare de la client poate fi alocat portului [0] al zonei de memorie, al doilea bloc de selectare a porturilor 932 poate fi configurat să determine dacă

vectorul de solicitare de la client poate fi alocat la portul [1] al zonei de memorie. Al doilea bloc de selectare a porturilor 932 include un prim detector al ultimului 1 (trailing one detector – TOD) 912-0, un al doilea TOD 912-1, un registru de mască 914, un deplasator de mască spre dreapta 916, un bloc de selectare 918 a câștigătorului portului [1] și un bloc de mascare cu logică ȘI 920. TOD 912 este configurat să primească un vector de solicitare de la client, care poate include o multitudine de biți, și să detecteze o poziție de bit, numărând de la dreapta la stânga, pe care vectorul devine pentru prima dată diferit de zero. Funcționarea celui de al doilea bloc de selectare 932 a porturilor este substanțial similară cu a primului bloc de selectare 930 a porturilor, cu excepția faptului că funcționează de la dreapta la stânga

5

vectorului de intrare, selectând ultimul 1 din vectorul de solicitare de intrare folosind un detector al ultimului 1 912-0.

10

[0108] Ieșirile blocurilor de selectare a câștigătorilor porturilor 908, 918 sunt comunicate și ele către același bloc de detectare 910 a câștigătorilor porturilor, care este configurat să determine dacă aceeași solicitare de acces la memorie a câștigat acces atât la portul [0] al zonei, cât și la portul [1] al zonei. Dacă același vector de solicitare de la client a câștigat acces atât la portul [0] al zonei, cât și la portul [1] al zonei, același bloc de detectare 910 a câștigătorilor selectează unul dintre porturile de zonă pentru a ruta solicitarea și alocă celălalt port solicitării imediat inferioare în grad din vectorul de intrare. Se evită astfel supraalocarea de resurse pentru o anumită solicitare, perfecționându-se astfel alocarea resurselor la

15

solicitanți concurenți.

20

[0109] Blocul de arbitrare a porturilor 900 funcționează astfel: începând din partea stângă a vectorului solicitării clientului pe 32 de biți și a LOD mascat 902-1, comunică poziția primului vector de solicitare mascat, dacă acest vector de solicitare mascat nu este depășit de o intrare cu prioritate mai ridicată sosită de la vectorii prioritari sau cu prioritate maximă; solicitantul care corespunde poziției LOD câștigă și primește acces la portul [0]. Poziția LOD este folosită și pentru a avansa poziția măștii cu ajutorul deplasatorului spre stânga 904 pe 32 de biți și este folosită, de asemenea, pentru comparația cu alocarea LOD la portul 1, pentru a verifica dacă același solicitant a primit acces la ambele porturi, caz în care numai unul dintre porturi este acordat, comutând un bistabil pentru a acorda acces alternativ între porturile 0 și

25

1 în cazul detectărilor succesive ale aceluiași câștigător. În cazul în care ieșirea LOD de la detectorul mascat 902-1 a primit prioritate printr-un bit 1 corespunzător din vectorul prioritar, clientul solicitant primește acces la portul 0 timp de 2 cicluri consecutive. În cazul în care nu există niciun prim 1 în vectorul mascat al solicitării clientului și nu există nicio solicitare cu

30

prioritate superioară, LOD nemascat câștigă și primește acces la portul 0. În oricare caz de mai sus, un bit 1 din vectorul cu prioritate maximă va avea întâietate față de orice solicitare anterioară și va acorda solicitantului acces nerestricționat la portul 0.

[0110] Logica din partea inferioară a diagramei începe de la dreapta vectorului de solicitare, în rest funcționând în același mod ca partea superioară, care începe din partea stângă a vectorului de solicitare. În acest caz, funcționarea blocului de arbitrare al portului 1 este identică cu porțiunea portului 0 a logicii din punctul de vedere al priorităților etc.

[0111] În unele implementări, un element de procesare 402 poate include o memorie-tampon pentru a reduce o latență a accesului la memorie datorată arbitrării accesului la memorie. FIG. 10 ilustrează un mecanism de reducere a latenței accesului la memorie datorate arbitrării accesului la memorie conform anumitor implementări. Într-un model tipic de arbitrare a accesului la memorie, blocul de arbitrare a accesului la memorie este constituit într-o bandă, ceea ce duce la o penalitate de arbitrare cu overhead fix în momentul alocării unei resurse partajate, cum ar fi un bloc RAM 502, la unul dintre multiplele elemente de procesare (de exemplu, solicitanții). De exemplu, când un solicitant 402 trimite o solicitare de acces la memorie către un bloc de arbitrare 608/702, durează cel puțin patru cicluri până când solicitantul 402 primește un mesaj de acordare a accesului, pentru că fiecare dintre etapele următoare durează cel puțin un ciclu: (1) analizarea solicitării de accesare a memoriei la codificatorul de adresă one-hot 602, (2) analizarea ieșirii codificatorului de adresă one-hot 602 la blocul de arbitrare 608/702, (3) trimiterea unui mesaj de acordare a accesului către codificatorul de adresă one-hot 602 din partea blocului de arbitrare 608/702 și (4) trimiterea mesajului de acordare a accesului către solicitant 402 din partea codificatorului de adresă one-hot 602. Ulterior, solicitantul 402 trebuie să trimită o solicitare de date de memorie către blocul RAM 502 și să recepționeze date de la blocul de memorie 502, fiecare dintre aceste etape durând cel puțin un ciclu. Prin urmare, o operație de accesare a memoriei are o latență de cel puțin cinci cicluri. Această penalitate fixă ar reduce lățimea de bandă a subsistemului de memorie.

[0112] Această problemă a latenței poate fi rezolvată folosind o memorie-tampon de solicitare a accesului 1002, care să fie păstrată în elementul de procesare 402. De exemplu, memoria-tampon de solicitare a accesului la memorie 1002 poate recepționa solicitări de acces la memorie din partea elementului de procesare la fiecare ciclu de procesor și le poate stoca până când acestea sunt gata să fie trimise la blocul de arbitrare a memoriei 608/702. Memoria-tampon 1002 activă sincronizează frecvența cu care sunt trimise solicitările de

memorie către blocul de arbitrare a memoriei 608/702 și frecvența la care se recepționează date din subsistemul de memorie. În unele implementări, memoria-tampon poate include o coadă. Numărul de elemente din memoria-tampon 1002 (de exemplu, profunzimea memoriei-tampon) poate fi mai mare decât numărul de cicluri pentru preluarea datelor din subsistemul de memorie. De exemplu, când latența de acces la RAM este de 6 cicluri, numărul de elemente din memoria-tampon 1002 poate fi 10. Memoria-tampon 1002 poate reduce penalitatea de latență rezultată din arbitrare, îmbunătățind debitul subsistemului de memorie. În principiu, folosind memoria-tampon a solicitărilor de acces la memorie, solicitanților li se poate aloca până la 100% din lățimea de bandă totală a memoriei.

10 [0113] Se va înțelege că o posibilă problemă legată de utilizarea mai multor instanțe RAM este că, permițând accesul simultan al mai multor elemente de procesare la subinstanțe din cadrul unui banc, poate rezulta o disputare a memoriei.

[0114] Documentul de față prezintă cel puțin două moduri de a aborda disputarea memoriei. În primul rând, se va avea grijă în momentul conceperii software-ului, după cum vom descrie ulterior, să se evite o disputare a memoriei și/sau un conflict de memorie prin dispunerea cu atenție a datelor în subsistemul de memorie, astfel încât să se reducă disputarea și/sau conflictele de memorie. În plus, instrumentele de dezvoltare de software asociate dispozitivului de procesare paralelă pot permite raportarea disputării memoriei sau a conflictului de memorie în faza de concepere a software-ului. Prin urmare, problemele de disputare a memoriei sau cele de conflict de memorie pot fi corectate prin perfecționarea dispunerii datelor ca reacție la disputarea memoriei sau la conflictul de memorie raportat(ă) în etapa de concepere a software-ului.

20 [0115] În al doilea rând, după cum vom descrie mai jos, blocul ISI din cadrul arhitecturii este configurat să detecteze conflictele de porturi (disputarea) în hardware și să întrerupă elementele de procesare cu grad redus de prioritate. De exemplu, blocul ISI este configurat să analizeze solicitările de acces la memorie din partea elementelor de procesare, să deservească succesiunea de solicitări de acces la memorie și să ruteze solicitările de acces la memorie conform ordinii de prioritate, astfel încât toate operațiile de citire sau scriere de date de la toate elementele de procesare să fie finalizate în ordinea priorității.

30 [0116] Ordinea de prioritate între elementele de procesare poate stabili în mai multe moduri. În unele implementări, ordinea de prioritate poate fi definită static în momentul proiectării sistemului. De exemplu, ordinea de prioritate poate fi codificată ca stare de resetare pentru regiștrii de sistem, astfel încât, la pornirea unui sistem, acesta să pornească cu

un set de priorități prealocate. În alte implementări, ordinea de prioritate se poate determina dinamic folosind regiștri programabili de către utilizator.

[0117] În anumite implementări, programatorii pot planifica dispunerea datelor pentru aplicațiile lor software, astfel încât să reducă disputarea subblocurilor de memorie partajate în cadrul unei zone de memorie. În unele cazuri, planificarea dispunerii datelor poate fi asistată de un bloc de arbitrare. De exemplu, blocul de arbitrare poate detecta o dispută a memoriei, poate permite, pe baza priorității, accesarea memoriei de către un element de procesare asociat activității cu gradul cel mai ridicat de prioritate, poate întrerupe alte elemente de procesare care își dispută memoria și poate derula disputa proces cu proces, până la rezolvarea acesteia.

[0118] FIG. 11 ilustrează o aplicație de software de planificare conform anumitor implementări. În această aplicație, software-ul de planificare poate coordona o implementare a unui filtru de încețoșare 3x3 în cadrul unei benzi de procesare. Software-ul de planificare poate, în faza de execuție, să determine o ordonare a operațiilor și să coordoneze operațiile elementelor de procesare. Un grafic de flux de date 1100 pentru banda de procesare include elementul 1 – elementul 5 1102-1110. Elementul 1 1102 poate include o memorie-tampon de intrare 1112, un bloc de procesare 1144 și o memorie-tampon de ieșire 1114. Memoria-tampon de intrare 1112 și memoria-tampon de ieșire 1114 pot fi implementate folosind un bistabil. În unele implementări, fiecare dintre celelalte elemente 1104-1110 poate avea o structură substanțial similară cu a elementului 1 1102.

[0119] În unele implementări, elementul 2 1104 poate include un element de procesare (de exemplu, un procesor vectorial sau un accelerator hardware) care poate filtra o intrare cu un filtru de încețoșare 3x3. Elementul 2 1104 poate fi configurat să primească date de intrare de la memoria-tampon partajată 1118, care păstrează temporar date de ieșire ale elementului 1 1102. Pentru a aplica un filtru de încețoșare 3x3 la o intrare, elementul 2 1104 poate recepționa cel puțin 3 linii de date de la memoria-tampon partajată 1118 înainte de a putea începe operația. Astfel, planificatorul software 1120, care poate rula pe un procesor RISC 1122, poate detecta că memoria-tampon partajată 1118 conține numărul corect de linii de date înainte de a semnală elementului 2 1104 că poate începe operația de filtrare.

[0120] După semnalul inițial că există 3 linii de date, planificatorul software 1120 poate fi configurat să semnalizeze elementului 2 1104 de fiecare dată când se adaugă o linie suplimentară nouă la memoria-tampon rulată de 3 linii 1118. În plus față de sincronizarea linie cu linie, se efectuează ciclul cu ciclul arbitrajul și sincronizarea pentru fiecare element

din banda de procesare. De exemplu, elementul 1102 poate include un accelerator hardware care produce un pixel complet de ieșire la fiecare ciclu. Pentru obținerea acestei ieșiri, acceleratorul hardware poate menține plină memoria-tampon 1112, astfel încât blocul de procesare 1114 să aibă suficiente date pentru a-și continua operațiile. În acest fel, blocul de procesare 1114 poate produce o ieșire suficientă pentru a menține debitul elementului 1102 cât mai ridicat.

[0121] În unele implementări, un lanț de instrumente software poate previziona conflictele de memorie analizând programul software care utilizează subsistemul de memorie. Lanțul de instrumente software poate include un mediu de dezvoltare integrat (IDE) bazat pe o interfață grafică de utilizator (GUI) (de exemplu, un IDE bazat pe Eclipse), din care programatorul să poată să editeze cod, să apeleze compilatorul, asamblorul și să efectueze depistarea erorilor la nivel de sursă când este necesar. Lanțul de instrumente software poate fi configurat să previzioneze conflictele de memorie printr-o analiză dinamică a programelor care rulează pe mai multe procesoare folosind un simulator de sistem care să emuleze întreaga procesare, magistrala, elementele de memorie și perifericele. De asemenea, lanțul de instrumente software poate fi configurat să înregistreze, într-un fișier jurnal sau pe un dispozitiv de afișare, dacă diferite programe care rulează pe diferite procesoare sau resurse hardware încearcă un acces concomitent la un anumit bloc al unei zone de memorie. Lanțul de instrumente software poate fi configurat să efectueze înregistrări ciclu cu ciclu.

[0122] În unele implementări, banda de procesare 1100 poate include și unul sau mai multe contoare hardware (de exemplu, un contor pentru fiecare instanță de memorie) care să fie incrementate de fiecare dată când se produce un conflict de memorie. Aceste contoare pot fi citite apoi de către un depanator (de exemplu, JTAG) și afișate pe un ecran sau înregistrate într-un fișier. Analiza ulterioară a fișierelor jurnal de către programatorul de sistem poate permite planificarea diferită a accesului la memorie, astfel încât să se reducă posibilitatea conflictelor la porturile de memorie.

[0123] O dificultate-cheie pentru programatorii de arhitectură IBM de tip CELL (ilustrată în FIG. 2) este planificarea prin program a transferurilor de date cu sute de cicluri înainte, astfel încât datele să poată fi controlate de DMA și stocate în stocarea locală (local storage – LS) înainte ca un procesor vectorial să acceseze datele. Unele implementări ale arhitecturii prezentate pot aborda această problemă realizând arbitrarea și planificarea accesurilor în hardware și înregistrând conflictele în contoare hardware care să poată fi citite de către

utilizator. În acest fel, arhitectura prezentată poate fi utilizată pentru a crea o bandă de înaltă performanță pentru procesarea video/de imagini.

[0124] FIG. 12 ilustrează o structură ierarhică a unui sistem care conține un dispozitiv de procesare paralelă conform anumitor implementări. Sistemul 1200 poate include un sistem de calcul paralel 1202 cu o multitudine de elemente de procesare, cum ar fi filtre, și o aplicație software 1204 care rulează pe sistemul de calcul paralel 1204, o interfață de programare a aplicațiilor (API) 1206 pentru interfața dintre aplicația 1204 și sistemul de calcul paralel 1202, un compilator 1208 care compilează aplicația software 1204 în vederea rulării în sistemul de calcul paralel 1202 și un planificator 1210 pentru controlul operațiilor elementelor de procesare din sistemul de calcul paralel 1202.

[0125] În unele implementări, dispozitivul de procesare paralelă prezentat poate fi configurat să funcționeze împreună cu un instrument de descriere a benzii de procesare (de exemplu, o aplicație software) 1204, care permite descrierea benzilor de procesare de imagine sub forma unui grafic de flux de date. Instrumentul de descriere a benzii de procesare 1204 este capabil să descrie benzile de procesare de imagini/video într-un mod flexibil, independent de platforma hardware/software de la bază. În particular, graficul de flux de date, utilizat de instrumentul de descriere a benzii de procesare, permite descrierea activităților independent de elementele de procesare (de exemplu, resursele procesorului și ale acceleratorului de filtre) care pot fi utilizate pentru implementarea graficului de flux de date. Datele de ieșire care rezultă din instrumentul de descriere a benzii de procesare pot include o descriere a graficului aciclic dirijat (DAG) sau a graficului de flux de date. Descrierea DAG sau a graficului de flux de date poate fi stocată într-un format adecvat, cum ar fi XML.

[0126] În unele implementări, descrierea DAG sau a graficului de flux de date poate fi accesibilă tuturor celorlalte instrumente din sistemul 1200 și poate fi folosită pentru controlarea operațiilor dispozitivului de procesare paralelă în conformitate cu DAG. FIG. 13 ilustrează cum se poate utiliza descrierea DAG sau a graficului de flux de date pentru a controla operațiile unui dispozitiv de procesare paralelă conform unor implementări.

[0127] Înainte de operația propriu-zisă a dispozitivului de calcul, un compilator 1208 pentru dispozitivul de procesare paralelă 1202 poate lua (1) o descriere a graficului de flux de date 1306 și (2) o descriere a resurselor disponibile 1302 și poate genera o listă de activități 1304 care indică modul în care se poate efectua DAG la nivelul mai multor elemente de procesare. De exemplu, atunci când o activitate nu poate fi realizată pe un singur element de procesare, compilatorul 1208 poate împărți activitatea la mai multe elemente de procesare;

126

atunci când activitatea se poate realiza pe un singur element de procesare, compilatorul 1208 poate alocă activitatea unui singur element de procesare.

[0128] În unele cazuri, când activitatea ar utiliza numai o parte a capacităților unui element de procesare, compilatorul 1208 poate fuziona și planifica executarea mai multor activități pe un singur element de procesare în mod secvențial, până la limita care poate fi suportată de un element de procesare. FIG. 14A ilustrează planificarea și emiterea de activități de către compilator și de către planificator conform unor implementări. Avantajul planificării activităților folosind un compilator și planificatorul este că compilatorul și planificatorul pot planifica automat activitățile pe baza operațiilor realizate de activități. Acesta este un mare avantaj față de situația anterioară, în care un programator trebuia să determine manual planificarea codului care rulează pe un element de procesare sau pe un grup de elemente de procesare care derulează o anumită activitate, inclusiv când anume să planifice transferurile de date realizate de DMA de la periferice la CMX, de la CMX la blocul CMX și de la CMX înapoi la periferice. Această muncă era dificilă și predispusă la erori, iar utilizarea DFG-urilor permite automatizarea acestui proces, economisind timp și măbind productivitatea.

[0129] În timpul execuției unui dispozitiv de calcul, planificatorul 1210 poate planifica dinamic activitatea la nivelul elementelor de procesare disponibile pe baza listei de activități 1304 generate de compilator 1208. Planificatorul 1210 poate funcționa pe procesorul-gazdă RISC 1306 într-un sistem multinucleu și poate planifica activitățile la nivelul elementelor de procesare, cum ar fi o multitudine de procesoare vectoriale, acceleratoare de filtru și unități de procesare pentru acces direct la memorie (DMA), folosind statisticile de la monitoarele de performanță hardware și de la cronometre 1308. În unele implementări, monitoarele de performanță hardware și cronometrele 1308 pot include contoare de întreruperi, contoare de conflicte CMX, contoare de cicluri de magistrală (ISI, APB și AXI) și contoare de cicluri, care pot fi citite de planificator 1210.

[0130] În unele implementări, planificatorul 1210 poate alocă activitățile la elemente de procesare disponibile pe baza statisticilor primite de la monitoarele de performanță hardware și de la cronometre 1308. Monitoarele de performanță hardware și cronometrele 1308 pot fi folosite pentru creșterea eficienței elementelor de procesare sau pentru efectuarea unei activități folosind mai puține elemente de procesare, astfel încât să se economisească energie sau să permită calcularea altor activități în paralel.

104

[0131] În acest scop, monitoarele de performanță hardware 1308 pot oferi o metrică a performanței. Metrica performanței poate fi un număr care indică nivelul de activitate al unui element de procesare. Metrica de performanță se poate utiliza pentru a controla numărul de elemente de procesare instanțiate pentru efectuarea unei activități. De exemplu, atunci când

5 metrica de performanță asociată unui anumit element de procesare este mai mare decât un prag predeterminat, planificatorul 1210 poate instanția un element de procesare suplimentar de același tip ca primul element de procesare, distribuind astfel activitatea la mai multe elemente de procesare. Un alt exemplu: atunci când metrica de performanță asociată unui

10 anumit element de procesare se situează sub un prag predeterminat, planificatorul 1210 poate scoate unul dintre elementele de procesare instanțiate de același tip ca primul element de procesare, reducând astfel numărul elementelor de procesare care realizează o anumită activitate.

[0132] În unele implementări, planificatorul 1210 poate prioritiza utilizarea elementelor de procesare. De exemplu, planificatorul 1210 poate fi configurat să determine dacă ar fi de

15 preferat ca activitatea să fie alocată unui procesor sau unui accelerator de filtru hardware.

[0133] În unele implementări, planificatorul 1210 poate fi configurat să schimbe dispunerea memoriei-tampon CMX din subsistemul de memorie, astfel încât sistemul să poată întruni criteriile de configurare de execuție. Criteriile de configurare a execuției pot include, de exemplu, debitul de procesare a imaginilor (cadre pe secundă), consumul de

20 energie, cantitatea de memorie folosită de sistem, numărul de procesoare aflate în funcțiune și/sau numărul de acceleratoare aflate în funcțiune.

[0134] Există mai multe moduri în care poate fi dispusă în memorie o memorie-tampon de ieșire. În unele cazuri, memoria-tampon de ieșire poate fi adiacentă fizic în memorie. În

alte cazuri, memoria-tampon de ieșire poate fi împărțită în „bucăți” sau „zone”. De exemplu,

25 memoria-tampon de ieșire poate fi împărțită în N fâșii verticale, unde N reprezintă numărul de procesoare alocate aplicației de procesare de imagini. Fiecare fâșie este situată într-o altă zonă CMX. Această dispunere poate favoriza procesoarele, deoarece fiecare procesor poate accesa local memoriile-tampon de intrare și de ieșire. Totuși, dispunerea aceasta poate fi nefavorabilă acceleratoarelor de filtru, putând cauza numeroase conflicte pentru acestea.

30 Acceleratoarele de filtru procesează adesea datele de la stânga la dreapta. Prin urmare, toate acceleratoarele de filtru și-ar iniția procesele accesând prima fâșie de imagine, ceea ce ar cauza multe conflicte de la început. În alte cazuri, memoria-tampon de ieșire poate fi împărțită întrețesută. De exemplu, memoria-tampon de ieșire poate fi împărțită pe toate cele

16 zone CMX, într-o întrețesere de mărime predeterminată. Această mărime predeterminată poate fi 128 de biți. Dispunerea întrețesută a memoriei-tampon de ieșire poate favoriza acceleratoarele de filtru, deoarece distribuirea accesurilor pe zonele CMX reduce riscul de conflicte.

- 5 [0135] În unele implementări, o memorie-tampon (cum ar fi una de intrare sau de ieșire) poate fi alocată în funcție de natura hardware și/sau software a producătorilor și a consumatorilor săi. Consumatorii sunt mai importanți, deoarece ei necesită, în general, mai multă lățime de bandă (filtrele citesc, de obicei, mai multe linii și produc o singură linie). Filtrele hardware sunt programate în funcție de dispunerea memoriilor-tampon (ele permit
- 10 adresarea memoriei adiacente, întrețesute și distribuite).
- [0136] FIG. 14B ilustrează un proces pentru planificarea automată a unei activități folosind compilatorul și planificatorul conform unor implementări. Compilatorul determină o listă de activități care trebuie efectuate de dispozitivul de procesare paralelă pe baza DAG. În etapa 1402, planificatorul este configurat să primească lista de activități și să păstreze lista de
- 15 activități în cozi separate. De exemplu, când lista de activități include (1) activități de efectuat de către DMA, (2) activități de efectuat de un procesor și (3) activități de efectuat de un filtru hardware, planificatorul poate stoca activitățile în trei cozi separate: de exemplu, o primă coadă pentru DMA, o a doua coadă pentru procesor și o a treia coadă pentru filtrul hardware.
- [0137] În etapele 1404-1408, compilatorul este configurat să emită activitățile către
- 20 componentele hardware asociate, pe măsură ce aceste componente devin disponibile pentru activități noi. De exemplu, în etapa 1404, când DMA devine disponibil pentru efectuarea unei activități, compilatorul de execuție este configurat să desfacă prima coadă pentru DMA și să comunice către DMA activitatea scoasă din coadă. Tot astfel, în etapa 1406, când procesorul devine disponibil pentru efectuarea unei activități, compilatorul de execuție este configurat să
- 25 desfacă a doua coadă pentru procesor și să comunice către procesor activitatea scoasă din coadă. De asemenea, în etapa 1408, când filtrul hardware devine disponibil pentru efectuarea unei activități, compilatorul de execuție este configurat să desfacă a treia coadă pentru filtrul hardware și să comunice către filtrul hardware activitatea scoasă din coadă.
- [0138] În unele implementări, planificatorul 1210 poate folosi valorile de contor de la
- 30 monitoarele de performanță hardware și cronometre 1308 pentru a regla utilizarea elementelor de procesare, în special în cazul în care rulează simultan mai multe benzi de procesare (de exemplu, o aplicație software 1204) în matricea de elemente de procesare, deoarece aceste benzi de procesare nu au fost neapărat proiectate împreună. De exemplu,

dacă lățimea de bandă efectivă alocată fiecărei benzi de procesare este mai sub valoarea preconizată și se produc numeroase conflicte apărute la accesarea memoriei CMX, planificatorul 1210 poate folosi aceste informații pentru a intercala executarea a 2 benzi de procesare, modificând ordinea în care sunt luate activitățile din cele 2 cozi ale benzilor de procesare și reducând astfel conflictele de memorie.

[0139] În unele implementări, compilatorul DAG poate funcționa în timp real (de exemplu, online). FIG. 15 ilustrează funcționarea unui compilator DAG în timp real conform anumitor implementări. Compilatorul DAG în timp real 1502 poate fi configurat să primească la intrare o descriere XML a DAG, descrierea elementelor de procesare disponibile și eventualele constrângeri definite de utilizator, cum ar fi numărul de procesoare, frecvența cadrelor, disiparea de putere avută în vedere etc. Apoi, compilatorul DAG în timp real 1502 poate fi configurat să planifice componentele DAG la nivelul elementelor de procesare, de exemplu, unitatea de procesare DMA, un procesor, un filtru hardware și o memorie, pentru a se asigura că DAG conform specificațiilor poate satisface constrângerile definite de utilizator când este mapat pe resursele de sistem. În unele implementări, compilatorul DAG în timp real 1502 poate determina dacă activitățile din DAG pot fi efectuate în paralel după un algoritm de parcurgere în lățime (breadth-first). Dacă lățimea DAG este mai mare decât numărul de elemente de procesare disponibile pentru efectuarea activității în paralel (de exemplu, cantitatea de putere de procesare disponibilă este mai mică decât paralelismul DAG), compilatorul DAG în timp real 1502 poate să „plieze” activitățile astfel încât acestea să fie efectuate secvențial pe elementele de procesare disponibile.

[0140] FIG. 16 compară o planificare generată de un planificator OpenCL cu o planificare generată de planificatorul DAG online propus conform anumitor implementări. Programul produs de planificatorul propus 1208/1502 poate elimina copiile redundante și transferurile DMA prezente într-o planificare de tip OpenCL tipică. Aceste transferuri de date sunt prezente într-o planificare OpenCL pentru că GPU-ul folosit pentru efectuarea procesării pe o activitate DAG este situat la distanță față de procesorul care execută planificarea. Într-un procesor de aplicație tipic folosit într-un dispozitiv mobil, se transferă blocuri mari de date înainte și înapoi între procesorul care execută planificarea și GPU-ul care face procesarea. În modelul propus, toate elementele de procesare împart același spațiu de memorie, nefiind astfel necesară copierea în ambele direcții și realizându-se astfel o economie considerabilă privind timpul, lățimea de bandă și disiparea de putere.

- [0141] În unele implementări, când o activitate ar utiliza numai o parte a capacităților unui element de procesare, planificatorul 1210 poate fi configurat să fuzioneze și să planifice executarea mai multor activități pe un singur element de procesare în mod secvențial, până la limita care poate fi suportată de un element de procesare, ca în FIG. 14.
- 5 [0142] Într-o aplicație de procesare de imagini, un planificator poate fi configurat să împartă procesarea activităților între procesoare, divizând o imagine în fâșii. De exemplu, imaginea poate fi divizată în fâșii verticale sau orizontale de lățime predeterminată.
- [0143] În unele implementări, planificatorul poate determina numărul de procesoare folosite pentru o anumită aplicație de procesare de imagine. Acest lucru permite
- 10 planificatorului să determine numărul de fâșii pentru imagine. În unele implementări se poate efectua o operație de filtrare de către procesoarele aflate în serie. De exemplu, când există 5 filtre software executate de aplicație, procesoarele 402 pot fi configurate fiecare să execute primul filtru software simultan într-un prim moment, al doilea filtru software simultan într-un al doilea moment etc. Acest lucru înseamnă că sarcina computațională este
- 15 echilibrată mai uniform între procesoarele alocate aplicației respective de procesare de imagini. Acest lucru se datorează faptului că procesoarele sunt configurate să execute simultan aceeași listă de filtre în aceeași ordine.
- [0144] Când sunt alocate prea multe procesoare la aplicația de procesare de imagini, procesoarele pot petrece mult timp în repaus, așteptând ca acceleratoarele de filtru hardware
- 20 să finalizeze activitățile. Pe de altă parte, când aplicației îi sunt alocate prea puține procesoare, acceleratoarele de filtru hardware pot petrece mult timp în repaus. În unele implementări, planificatorul 1210 poate fi configurat să detecteze aceste situații și să se adapteze în consecință. În alte implementări, planificatorul 1210 poate fi configurat să supraalocă procesoare aplicațiilor particulare de procesare de imagini și să permită
- 25 procesoarelor să își reducă puterea odată ce și-au finalizat activitatea înaintea acceleratoarelor de filtru hardware.
- [0145] În unele implementări, planificatorul poate folosi un mecanism barieră pentru a sincroniza elementele de procesare, cum ar fi acceleratoarele de filtru hardware și procesoarele. Datele de ieșire ale planificatorului pot include un flux de comenzi. Aceste
- 30 comenzi pot include (1) comenzi de pornire pentru elementele de procesare, cum ar fi acceleratoarele de filtru hardware și procesoarele, și (2) comenzi-barieră. O comandă-barieră arată că elementele de procesare trebuie să aștepte și să nu treacă la un set următor de comenzi decât după ce toate elementele de procesare din grup au ajuns la comanda-barieră,

chiar dacă unele elemente de procesare și-au finalizat practic activitatea. În unele implementări, planificatorul poate comunica această comandă-barieră pe baza dependențelor dintre activitățile efectuate de elementele de procesare.

[0146] FIG. 17 ilustrează un mecanism-barieră pentru sincronizarea elementelor de procesare conform anumitor implementări. Fluxul de comenzi include comenzi-barieră (1702, 1712) și comenzi de activitate (1704, 1706, 1708, 1710). Fiecare comandă de activitate poate fi asociată cu un element de procesare și, ca în graficul de mai jos, comenzile de activitate pot fi finalizate la momente diferite. Prin urmare, planificatorul poate include o comandă-barieră 1712, astfel încât elementele de procesare să nu treacă la activități viitoare decât după ștergerea comenzii-barieră 1712. Acest mecanism-barieră poate fi considerat ca fiind o organizare temporară a activităților paralele într-o bandă de procesare.

[0147] În unele implementări, mecanismul-barieră este implementat în hardware folosind semnale de întrerupere 1714. De exemplu, planificatorul poate programa o mască de biți, care specifică ce elemente de procesare aparțin unui grup. Pe măsură ce elementele de procesare finalizează activitățile alocate, se declară semnale de întrerupere asociate elementelor de procesare respective. Odată ce toate semnalele de întrerupere asociate elementelor de procesare din grup au fost declarate, controlerul elementelor de procesare poate primi un semnal de întrerupere global, care arată că toate elementele de procesare au ajuns la comanda-barieră.

20 Sursele de întrerupere pot include procesoare vectoriale SHAVE, procesoare RISC, filtre hardware sau evenimente externe. În special filtrele hardware permit numeroase moduri, inclusiv un mod de memorie-tampon necirculară, în care memoria-tampon de intrare/ieșire conține un cadru, iar filtrul poate fi configurat să emită o un singur semnal de întrerupere fie când a procesat întregul cadru de intrare, fie când a scris întregul cadru de ieșire

25 corespunzător. De asemenea, filtrele pot fi programate să opereze asupra liniilor, asupra corecțiilor sau asupra blocurilor din cadre folosind setări adecvate pentru dimensiunile imaginilor, pentru intervalul de adresă/linie de bază al memoriei-tampon etc.

[0148] O provocare importantă în cazul unui dispozitiv complex de procesare paralelă este modalitatea de a programa elementele de procesare din dispozitivul de procesare paralelă, în special pentru sistemele încorporate care sunt foarte sensibile la putere și au la dispoziție puține resurse (de exemplu, resurse computaționale și de memorie). Imagistica computațională și mai ales procesarea video și de imagini solicită foarte puternic sistemele

încorporate în ceea ce privește performanțele, deoarece dimensiunile și frecvențele cadrelor sunt foarte mari, ele crescând tot mai mult de la an la an.

[0149] Soluția la această problemă prezentată aici este de a asigura o interfață de programare pentru aplicații (API) 1206 care să permită scrierea aplicațiilor la nivel înalt de către un programator fără a necesita cunoștințe aprofundate privind detaliile arhitecturii procesoarelor multinucleu 1202. Folosind API software 1206, programatorul poate crea rapid noi benzi de procesare de imagini sau video fără a cunoaște în profunzime detaliile implementării, deoarece detaliile privind implementarea funcțiilor în software, pe procesoare programabile, sau în hardware sunt abstractizate, programatorul neconfruntându-se cu ele. De exemplu, o implementare a unui filtru de încetoșare este dată ca implementare de software de referință care rulează pe unul sau mai multe procesoare sau filtre de accelerator hardware. Programatorul poate folosi inițial o implementare cu filtru de încetoșare software, iar apoi poate trece la utilizarea unui filtru hardware fără a schimba în ansamblu implementarea benzii de procesare, deoarece nu programatorul, ci ISI, AMC și blocul de arbitrare CMX au rolul de a determina care procesor și resurse hardware obțin acces la blocurile de memorie fizice și în ce ordine.

[0150] În timp ce abordarea cu memorie cu porturi multiple descrisă mai sus este adecvată pentru partajarea memoriei în condiții de lățime mare de bandă și latență scăzută între procesoare identice, ea nu este ideală pentru partajarea lățimii de bandă cu alte dispozitive. Aceste alte dispozitive pot fi acceleratoare hardware și alte procesoare cu diferite cerințe de latență, în special la aplicațiile care necesită o lățime de bandă foarte mare, cum este procesarea computațională de video și de imagini.

[0151] Arhitectura prezentată poate fi folosită împreună cu un subsistem de memorie cu porturi multiple, pentru a asigura lățimea de bandă mai mare necesară mai multor accesări simultane din partea unei multitudini de procesoare VLIW programabile având cerințe de latență cu un grad de determinare ridicat, un grup mare de filtre hardware programabile pentru procesarea de imagini/video, precum și o interfață cu magistrala pentru a permite controlul și accesarea datelor de către un procesor-gazdă convențional și de către periferice. FIG. 18 ilustrează dispozitivul de procesare paralelă cu diferite tipuri de elemente de procesare conform anumitor implementări. Dispozitivul de procesare paralelă include o multitudine de procesoare 1802 și o multitudine de acceleratoare de filtru 1804, iar multitudinea de procesoare 1802 și multitudinea de acceleratoare de filtru 1804 pot fi cuplate

la subsistemul de memorie 412 prin ISI 410, respectiv prin controlerul de memorie al acceleratorului (AMC) 1806.

[0152] Subsistemul de AMC 1806 și subsistemul de memorie multinucleu (CMX) 412 asigură stocarea pe cip, facilitând procesarea semnalului digital de streaming de putere mică
5 la nivelul procesoarelor 1802, precum și al acceleratoarelor de filtru hardware 1804 pentru anumite aplicații de procesare de imagini/video. În unele implementări, memoria CMX 412 este organizată în 16 zone de 128 kB, organizate sub forma unor cuvinte pe 64 de biți (2 MB) în total. Fiecare procesor 1802 poate avea acces direct la o zonă din subsistemul de memorie 412 și acces indirect (cu latență mai mare) la toate celelalte zone ale subsistemului de
10 memorie 412. Procesoarele 1802 pot utiliza memoria CMX 412 pentru a stoca instrucțiuni sau date, iar acceleratoarele de filtru hardware 1804 folosesc memoria CMX 412 pentru a stoca date.

[0153] Pentru a facilita partajarea datelor între elementele de procesare eterogene, permițând procesoarelor intolerante la latență 1802 să atingă performanțe ridicate când
15 accesează o memorie CMX partajată 412 cu acceleratoare de filtru hardware 1804, acceleratoarele de filtru hardware 1804 sunt concepute să tolereze latența. Acest lucru se realizează prevăzând fiecare accelerator de filtru hardware (filtru) 1804 cu memorii FIFO locale, care conferă mai multă elasticitate sincronizării, precum și cu un comutator crossbar pentru a partaja accesul la CMX, ISI putând astfel să susțină comunicarea inter-SHAVE fără
20 dispute cu acceleratoarele de filtru hardware, ca în FIG. 10 conform anumitor implementări.

[0154] În plus față de conflictele la porturile de ieșire, este posibil și conflictul cu accesarea unui port de intrare ISI-410. Dacă mai multe zone externe încearcă să acceseze aceeași zonă de memorie în oricare ciclu, se poate produce un conflict de port. Maparea
portului LSU la portul de interconectare ISI este fixă, fiind astfel posibil ca SHAVE 0 1802-0
25 să acceseze zona 2 prin portul LSU 0 și ca SHAVE 11 (1702-11) să acceseze zona 2 prin portul LSU 1 fără să apară conflicte. Matricea ISI poate permite transferul la fiecare ciclu a 8 x 2 porturi x 64 biți de date. De exemplu, SHAVE N 1802 poate accesa zona N+1 prin portul LSU 0 și 1, iar toate cele 8 procesoare SHAVE pot accesa simultan fără nicio întrerupere.

[0155] În unele implementări, subsistemul de memorie 412 poate fi împărțit logic pe zone
30 (blocuri). FIG. 19 ilustrează subsistemul de memorie multinucleu propus conform anumitor implementări. FIG. 19 ilustrează o interconectare detaliată prin magistrală între AXI, AHB, SHAVE-uri, ISI și CMX, precum și acceleratoare de filtru, AMC și CMX. Diagrama arată două porturi de intrare AMC și două porturi de ieșire AMC, 2 porturi de intrare ISI și 2

porturi de ieșire ISI, conexiuni la L2 cache și excludere reciprocă (mutex), precum și arbitrarea internă a scrierii/citirii și multiplexarea sursei pentru adresarea celor 4 blocuri de memorie și a memoriei FIFO, precum și selectarea destinației de ieșire a celor 4 ieșiri ale blocului de memorie spre ISI și AMC.

- 5 [0156] Fiecare zonă se poate conecta la două din 16 surse de intrare ISI posibile, inclusiv 12 SHAVE-uri, DMA, unitatea de gestionare a texturii (Texture Management Unit – TMU) și interfață de magistrală AHB către procesorul-gazdă montat pe placă. Tot astfel, fiecare zonă are 2 porturi ISI de ieșire care permit unei zone să trimită date la 2 din 16 destinații posibile, inclusiv 12 SHAVE-uri, DMA, unitatea de gestionare a texturii (TMU) și interfețele de
- 10 magistrală AHB și AXI către procesorul-gazdă montat pe placă. În implementarea preferată, zona de memorie conține 4 blocuri RAM fizice cu bloc de arbitrare a intrării care, la rândul său, se conectează la procesorul SHAVE local (2 LSU-uri și 2 porturi de instrucțiuni pe 64 biți), 2 porturi de intrare ISI, 2 porturi de intrare AMC și FIFO folosită pentru mesageria inter-SHAVE, precum și o FIFO de mesagerie, L2 cache și blocuri de excludere reciprocă
- 15 (mutex).

- [0157] Pe calea de ieșire de la o zonă CMX, intrarea la blocul de selectare a destinației este conectată la cele 4 instanțe RAM, precum și la L2 cache și la blocurile hardware mutex. Ieșirile de la blocul de selectare a destinației, ilustrat în FIG. 20 ca blocul 2002, se conectează la cele 2 porturi LSU locale și la porturile de instrucțiuni (SP_1 și SP_0), precum și 2 porturi
- 20 de ieșire ISI și 2 porturi de ieșire AMC. Cele 2 porturi ISI permit conectarea unei zone locale la două destinații din cele 12 posibile procesoare, DMA, TMU AXI și magistralele gazdă AHB. Procesoarele sunt prevăzute cu acces la memorie prin interconectarea inter-SHAVE (ISI), care se conectează la cele 2 intrări ISI pe 64 biți și la 2 porturi de ieșire ISI pe 64 biți conținute într-o zonă a subsistemului de memorie multinucleu. Accesul determinist
- 25 caracterizat prin lățime de bandă mare și latență scăzută oferit de interconectarea ISI reduce întreruperile la nivelul procesoarelor și oferă un debit computațional ridicat.

- [0158] FIG. 21 ilustrează o arhitectură AMC crossbar conform anumitor implementări. AMC crossbar 1806 poate fi configurat să conecteze filtrele hardware de procesare de imagine 1804 la porturile AMC ale zonelor de memorie multinucleu CMX 412. AMC 1806
- 30 poate include unul sau mai multe controlere de port 2102 ale zonei, preferabil unul pentru fiecare bloc CMX 412. Controlerele porturilor zonei 2102 sunt, la rândul lor, conectate la filtrul de solicitări de adrese de zone (slice address request filter – SARF) 2104. La rândul său, SARF 2104 este conectat la clienții AMC (în această implementare, clienții AMC sunt

acceleratoarele hardware de procesare de imagine). SARF 2104 acceptă solicitări de citire/scriere de la acceleratoarele de filtru și le oferă acestora semnale de solicitare sau de permisiune, acceptând date și adrese de la porturile SIPP care au primit acces de scriere și furnizând date de citire celor care au primit acces de citire. În plus, SARF oferă mastering
5 AXI pe magistrala-gazdă AXI pentru procesorul-gazdă din sistem, ceea ce permite gazdei să interogheze (citire/scriere) memoria CMX prin comutatorul crossbar AMC.

[0159] În unele implementări, este prevăzut un controler al porturilor zonei 2102 în AMC 1806, care comunică cu cele 2 porturi de citire și cele 2 porturi de scriere din zona de memorie CMX asociată 412, ca în FIG. 21. Văzut dinspre partea de accelerator de filtru a
10 subsistemului de memorie CMX 412, fiecare filtru hardware este conectat la un port al comutatorului crossbar 1806 de la controlerul de memorie al acceleratorului (AMC). AMC 1806 are o pereche de porturi de citire pe 64 biți și o pereche de porturi de scriere pe 64 biți care îl conectează la fiecare zonă de memorie CMX 412 (există 16 zone, totalizând 2 MB în implementarea preferată). Conectarea acceleratoarelor hardware de procesare de imagine la
15 AMC 1806 prin interfețe client de citire sau de scriere și asigurarea unei memorii-tampon locale în cadrul acceleratoarelor permite relaxarea cerințelor de latență, lăsând disponibilă mai multă lățime de bandă pentru ISI și procesoare, cu o sincronizare cu grad ridicat de determinare care permite reducerea întreruperii procesoarelor.

[0160] FIG. 20 ilustrează o singură zonă a infrastructurii CMX conform anumitor
20 implementări. Zona conține un arbitru și multiplexare la sursă, care permite ca până la 4 din opt posibile surse pe 64 de biți să acceseze cele 4 blocuri SRAM fizice din zona CMX, L2 cache partajată, blocurile hardware mutex partajate pentru negocierea inter-procesor a excluderii reciproce în rândul firelor de proces, precum și o memorie FIFO pe 64 de biți folosită pentru mesageria inter-SHAVE cu lățime de bandă mică. Cele șase surse de intrare
25 sunt: AM Cout1 și AM Cout0, care sunt conectate la porturile corespunzătoare slice_port[1] și slice_port[0] din ACM, ca în FIG. 21; la care se adaugă 2 porturi ISI (ISlout1 și ISlout0); 2 porturi LSU (LSU_1 și LSU_0); și, în cele din urmă, 2 porturi de instrucțiuni (SP_1 și SP_0), care, combinate, permit citirea de instrucțiuni pe 128 de biți de la CMX. Arbitrul și multiplexarea sursei generează adresa de citire/scriere și datele pe 64 de biți pentru
30 controlarea celor 4 blocuri SRAM ca reacție la accesarea prioritară din cele 8 surse de intrare. În timp ce intrarea memoriei FIFO de comunicații inter-SHAVE pe 64 de biți este conectată la arbitru și la multiplexorul sursei, ieșirea poate fi citită numai de către procesorul local din zona CMX. În practică, fiecare procesor comunică cu memoria FIFO de mesagerie a altui

108

procesor prin porturile ISIout1 și ISIout0 și prin infrastructura ISI externă zonei CMX, care interconectează CMX, zonele și procesoarele.

[0161] În plus față de arbitrarea între 64 de solicitanți de la fiecare dintre cele 2 porturi AMC dintr-o zonă, cum se vede în FIG. 20, este prevăzut un arbitru suplimentar 2:1 care să arbitreze între porturile AMC 1 și 0. Scopul acestui arbitru 2:1 este să prevină saturarea de către unul sau celălalt dintre cele 2 porturi AMC a întregii lățimi de bandă a portului AMC, fapt ce ar duce la întreruperi excesive la unul dintre porturile solicitante. Această caracteristică suplimentară asigură o alocare mai echilibrată a resurselor în prezența mai multor solicitanți intensi ai lățimii de bandă a portului, menținând astfel un debit mai ridicat în întreaga arhitectură. Tot astfel, un arbitru 2:1 arbitrează între cele 2 porturi de procesor SP1 și SP0, din motive similare.

[0162] Logica de arbitrare și multiplexare controlează, de asemenea, accesul procesoarelor, fie direct, fie prin ISI, la o memorie L2 cache partajată, printr-un al doilea nivel de arbitrare care partajează accesul conform unui algoritm round robin strict între 16 surse posibile, un port pe 64 de biți fiind conectat între arbitrul de nivel doi și fiecare dintre cele 16 zone CMX. Tot astfel, aceeași logică permite accesul la cele 32 de mutexuri hardware care sunt folosite pentru negocierea între procesoare a excluderii reciproce în rândul firelor de proces care rulează pe cele 12 procesoare montate pe placă și cele 2 procesoare RISC pe 32 de biți (prin conexiunile de magistrală AHB și AXI de pe ISI).

[0163] Prioritatea în implementarea preferată este că SP_1 și SP_0 au gradul cel mai ridicat de prioritate, urmate de LSU_1 și LSU_0, apoi de ISIout1 și ISIout0, iar AM Cout1 și AM Cout0 și, în cele din urmă, FIFO au gradul de prioritate cel mai mic. Motivul acestei alocări a priorității este că SP_1 și SP_0 controlează accesul de program al procesorului la CMX, iar procesorul va intra imediat în întrerupere dacă nu este disponibilă următoarea instrucțiune, după care urmează LSU_1 și LSU_0, cauzând din nou întreruperea procesorului; tot astfel, ISIout1 și ISIout0 vin de la alte procesoare și vor face ca acestea să se întrerupă dacă datele nu sunt disponibile imediat. Porturile AM Cout1 și AM Cout0 au prioritatea cea mai redusă, ele având memorii FIFO integrate și putând astfel să tolereze un nivel ridicat de latență înainte de a se întrerupe. FIFO de la procesor este necesară numai pentru mesageria pe lățime de bandă mică dintre procesoare, având astfel prioritatea cea mai mică dintre toate.

[0164] Odată ce arbitrul a permis accesul a până la 4 surse la cele 4 blocuri SRAM, L2 cache, mutexuri și FIFO, sunt selectate datele de ieșire de la cele șase surse de date citite, inclusiv 4 blocuri SRAM, L2 cache și mutexuri, aceste date fiind dirijate spre până la 4 din 8

posibile porturi de destinație pe 64 de biți; 4 la procesorul asociat zonei de memorie (SP_1, SP_0, LSU_1 și LSU_0); 2 asociate ISI (ISlout1 și ISlout0) și, în cele din urmă, 2 asociate AMC (AMCout1 și AMCout0). Nu este necesară nicio prioritizare la multiplexorul de ieșire, numai 4 surse pe 64 de biți trebuind să fie distribuite la 8 porturi de destinație.

5 [0165] FIG. 22 ilustrează un controler AMC cu porturi crossbar conform anumitor implementări. Controlerul 2202 al portului crossbar de la AMC include un arbitru round robin 2204, care conectează controlerul portului 2202 la acceleratoarele 1804 ale căror
10 solicitări au fost filtrate prin procesor. După aceea, arbitrul poate transmite solicitările valide de la clienții AMC la memoria FIFO a controlerului de port. În cazul solicitărilor de citire, se transmite un răspuns la solicitare (ID-ul clientului de citire și indexul de linie) către FIFO TX ID de citire. Datele returnate de la portul zonei și semnalele valide sunt introduse în logica de citire a controlerului de port, care prezintă ID-ul de client de citire și indicii de linie din FIFO Rd TX ID și transmite datele de citire de la portul de zonă corespunzător și semnalele valide
15 spre FIFO de date Rd, de unde acestea pot fi citite de clientul AMC solicitant. Pe partea CMX a FIFO, logica de întrerupere a porturilor prezintă solicitările de la FIFO și asigură controlul porturilor de zonă pentru cele 2 porturi de intrare AMC din zona de memorie CMX asociată.

[0166] Numărul de interfețe client de citire și de scriere spre CMX sunt configurabile separat. Orice client se poate adresa oricărei (sau oricăror) zone din CMX. Având 16 zone de
20 memorie în CMX, 2 porturi la fiecare zonă și o frecvență de procesor de 600 MHz în sistem, lățimea maximă totală a memoriei de date care poate fi asigurată clienților este de 143 GB/s: Lățimea de bandă max. = 600 MHz * (64/8) * 2 * 16 = 1,536e11 B/s = 143 GB/s.

[0167] La frecvența superioară de 800 MHz a procesorului, banda de frecvență crește la
25 191 GB/sec. Controlerul AMC arbitrează accesările simultane ale interfețelor de citire/scriere din partea blocurilor de accelerator hardware conectate la el. Cel mult două adrese de citire/scriere din fiecare zonă de memorie pot fi alocate la fiecare ciclu de procesor, rezultând astfel o lățime de bandă de memorie maximă a zonei de 8,9 GB/s la o frecvență de 600 MHz a procesorului din sistem. Accesul clienților nu este restricționat la spațiul de adrese din CMX. Orice acces care iese din spațiul de adrese din CMX este transmis la programul master
30 al magistralei AXI de la AMC.

[0168] FIG. 23 ilustrează o operație de citire folosind un AMC 1806 conform anumitor implementări. În această ilustrație, 4 cuvinte de date sunt citite din intervalul de adrese A0-3. Clientul AMC (de exemplu, un accelerator de filtru 1804) declară mai întâi o solicitare la

intrarea controlerului de port. Controlerul de port 2202 răspunde emițând un semnal de permisiune (gnt) care, la rândul său, determină clientul să emită adresele A0, A1, A2 și, în cele din urmă, A3. Valorile rindex corespunzătoare apar pe arcul ascendent al ciclului de procesor (clk) care corespunde fiecărei permisiuni. Se poate vedea că sincronizarea poate fi
5 foarte elastică pe partea clientului în comparație cu datele și adresele de index, care sunt transmise de la zona CMX la controlerul de port. Sincronizarea deterministă pe partea CMX a controlerului de port permite accesul eficient la CMX partajat între clienții AMC și procesoare, care au o sensibilitate ridicată la latență, iar memoriile FIFO și stocarea locală de la clienții AMC permit o variabilitate ridicată a sincronizării în partea de client AMC (de
10 exemplu, acceleratorul de filtru) din subsistemul de memorie CMX 412.

[0169] FIG. 24 ilustrează o operație de scriere folosind un AMC 1806 conform anumitor implementări. În diagrama de sincronizare este ilustrat transferul a 4 cuvinte de date la CMX prin AMC. Clientul AMC emite o solicitare, iar la următorul arc ascendent al ciclului de procesor (clk), semnalul de permisiune (gnt) se ridică, transferând cuvântul de date D0
15 asociat adresei A0 prin AMC. Semnalul gnt coboară apoi timp de un ciclu de procesor, iar pe următorul arc clk ascendent, se ridică semnalul gnt timp de 2 cicluri de procesor, oferind acces pentru D1 și D2 la adresele A1, respectiv A2, înainte ca semnalul gnt să coboare din nou. La următorul arc clk ascendent, semnalul gnt se ridică din nou, permițând transferarea cuvântului de date D3 la adresa A3, după care semnalele req și gnt coboară la următorul arc
20 clk, în așteptarea următoarei solicitări de citire/scriere.

[0170] Cadrul software al benzii de procesare de imagine pentru streaming (SIPP) folosit împreună cu modelul din FIG. 12 asigură o abordare flexibilă a implementării benzilor de procesare de imagini folosind memoria CMX 412 pentru memoriile-tampon de linie de scanare, blocurile de cadre (subsecțiuni de cadre) sau chiar întregi cadre la rezoluție ridicată
25 cu ajutorul unei pastile DRAM externe într-un modul, conectate la un substrat de care este atașată matricea de procesare de imagini/video. Cadrul SIPP se ocupă de complexități precum gestionarea marginilor imaginii (replicarea pixelilor) și gestionarea memoriei-tampon ciclice de linii, făcând ca implementarea funcțiilor de ISP (procesare a semnalului de imagine) din software (de la nivelul procesorului) să devină mai simplă și mai generică.

30 [0171] FIG. 25 ilustrează dispozitivul de procesare paralelă 400 conform anumitor implementări. Dispozitivul de procesare paralelă 400 poate include un subsistem de memorie (CMX) 412, o multitudine de acceleratoare de filtru 1804 și o structură de magistrală 1806 pentru arbitrarea accesului la subsistemul de memorie 412. Subsistemul de memorie (CMX)

412 este construit astfel încât să permită unei multitudini de elemente de procesare 402 să acceseze, în paralel, memoria de date și de cod de program fără a cauza întreruperi. Aceste elemente de procesare 402 pot include, de exemplu, procesoare SHAVE (unitate de procesare vectorială cu arhitectură hibridă pentru streaming) procesoare VLIW (cuvânt de instrucțiune foarte lung) după caz, acces paralel la memoria de date și de cod de program fără întreruperi sau un accelerator de filtru. În plus, subsistemul de memorie (CMX) 412 poate prevedea ca un procesor-gazdă (neinclus în ilustrație) să acceseze subsistemul de memorie CMX 412 printr-o magistrală paralelă precum AXI (neinclusă în ilustrație). În anumite implementări, fiecare element de procesare 402 poate citi/scrie până la 128 de biți pe ciclu prin porturile sale LSU și poate citi cod de program de până la 128 biți pe ciclu prin portul său de instrucțiuni. În plus față de interfețele ISI și AMC pentru procesoare, respectiv pentru acceleratoarele de filtru, CMX 412 oferă acces simultan la memorie pentru citire/scriere prin interfețele de magistrală AHB și AXI. AHB și AXI sunt magistrale de interfață paralelă ARM standard, care permit conectarea unui procesor, a memoriei și a perifericelor folosind o infrastructură de magistrală partajată 1806. Subsistemul de memorie CMX 412 poate fi configurat să gestioneze un maxim de 18 accesuri de 128 biți la memorie la fiecare ciclu.

[0172] Acceleratoarele 1804 includ o colecție de filtre hardware de procesare de imagine care pot fi utilizate în cadrul software SIPP 1200. Acceleratoarele 1804 pot scuti elementele de procesare 1802 de o parte din funcționalitatea cea mai solicitantă din punct de vedere computațional. Diagrama arată cum se pot conecta o multitudine de acceleratoare de filtru 1804 la AMC 1804 care realizează filtrarea de adrese, arbitrarea și multiplexarea. La AMC 1804 mai pot fi conectate multiple interfețe seriale de cameră MIPI 2502, în implementarea preferată existând în total 12 culoare seriale MIPI conectate în 6 grupe de câte 2 culoare. AMC 1804 este conectat, de asemenea, la interfețele AXI și APB, pentru a permite celor 2 procesoare RISC ale sistemului din implementarea de referință să acceseze memoria CMX prin AMC. Elementul final al diagramei este CMX 412, a cărei accesare o arbitrează AMC 1804, permițând mai multor acceleratoare de filtru hardware 1804 acces simultan la instanțele RAM fizice din memoria CMX 412. Este înfățișat, de asemenea, un accelerator de filtru de referință 1804, în acest caz un filtru 5x5 2D, care conține o bandă de procesare aritmetică fp16 (format cu virgulă mobilă pe 16 biți ca IEEE754), un controler asociat al întreruperilor în banda de procesare, un client de citire al memoriei-tampon de linie pentru stocarea unei linii de intrare în banda de procesare fp16, o intrare de control al începutului de linie și client de scriere al memoriei-tampon de linie pentru stocarea datelor de ieșire ale benzii de

procesare fp16. Pentru ca acceleratoarele să se potrivească în cadrul SIPP, ele necesită acces la memoria CMX la o lățime de bandă ridicată, acest acces fiind asigurat de controlerul de memorie pentru acceleratoare (AMC).

5 [0173] În unele implementări, subsistemul de memorie CMX 412 poate fi împărțit în blocuri sau zone de 128 kB asociate cu elementul lor de procesare învecinat 402 pentru un acces de mare viteză, cu consum de putere scăzut. În cadrul unei zone, memoria este organizată sub forma unor blocuri mai mici, de exemplu blocuri SRAM independente 3x32 kB, 1x16 kB și 2x8 kB. Dimensiunea fizică a memoriei RAM poate fi aleasă ca un compromis între utilizarea spațiului și flexibilitatea configurației. Orice element de procesare 10 402 poate accesa memoria RAM fizică oriunde în subsistemul de memorie (CMX) 412 cu aceeași latență (3 cicluri), dar accesul în afara zonei locale a unui procesor este limitată ca lățime de bandă și va avea un consum de putere mai ridicat decât accesarea unei zone de memorie locale. În general, pentru a reduce consumul de putere și a spori performanțele, un element de procesare 402 poate stoca date local într-o zonă de memorie dedicată.

15 [0174] În unele implementări, fiecare memorie RAM fizică poate avea o lățime de 64 de biți. Dacă mai multe elemente de procesare 402 încearcă să acceseze aceeași memorie RAM fizică, se poate produce un conflict care duce la o întrerupere a procesorului. CMX va arbitra automat conflictele de port, asigurându-se că nu se pierd date. Pentru fiecare conflict de port, un element de procesare 402 este întrerupt timp de un ciclu, ceea ce reduce debitul. Printr-o 20 dispunere atentă a datelor (de către programator) în cadrul CMX 412 se pot evita conflictele de port și se pot valorifica mai bine ciclurile de procesor.

[0175] În unele implementări, o multitudine de procesoare sunt dotate cu acceleratoare și memorie CMX.

25 [0176] Se poate observa din arhitectura hardware de procesare de imagine din FIG. 25 că fiecare accelerator de filtru 1804 poate include cel puțin o interfață client AMC de citire și/sau de scriere pentru accesarea memoriei CMX 412. Numărul de interfețe client de citire/scriere ale AMC 1806 se poate configura după cum este necesar. AMC 1806 poate include o pereche de porturi pe 64 de biți în fiecare zonă de memorie CMX 412. AMC 1806 rutează solicitările de la clienții săi către zona CMX 412 adecvată (prin decodificarea parțială 30 a adresei). Solicitățile simultane din partea mai multor clienți pentru aceeași zonă de memorie pot fi arbitrate după un algoritm round robin. Datele de citire returnate de la CMX 412 sunt rutate înapoi la clienții de citire AMC solicitanți.

- 5 [0177] Clienții AMC (acceleratoarele) 1804 prezintă o adresă completă pe 32 de biți la AMC 1806. Accesurile din partea clienților care nu se mapează pe spațiul memoriei CMX sunt transmise la programul master al magistralei AXI de la AMC. Accesările simultane (din afara spațiului de memorie CMX) din partea diferitor clienți sunt arbitrate după un algoritm round robin.
- [0178] AMC 1806 nu se limitează la a oferi acces la CMX 412 pentru acceleratoarele de filtru 1804; orice accelerator hardware sau elemente terțe pot utiliza AMC 1806 pentru a accesa memoria CMX și spațiul mai larg de memorie al platformei dacă interfețele sale de memorie sunt adaptate în mod adecvat la interfețele client de citire/scriere ale AMC.
- 10 [0179] Banda hardware de procesare de imagine (SIPP) poate include acceleratoarele de filtru 1804, un bloc de arbitrare 1806, control MIPI 2502, interfețe APB și AXI și legături la memoria multiport CMX 412, precum și un filtru hardware 5x5 exemplar. Această configurație permite ca o multitudine de procesoare 1802 și acceleratoare de filtru hardware 1804 pentru aplicațiile de procesare de imagine să partajeze un subsistem de memorie 412
- 15 compus dintr-o multitudine de blocuri fizice RAM (memorie cu acces aleatoriu) cu un singur port.
- [0180] Utilizarea memoriilor cu un singur port crește eficiența energetică și spațială a subsistemului de memorie, dar limitează lățimea de bandă. Configurația propusă le permite acestor blocuri RAM să se comporte ca un subsistem de memorie virtual multiport, capabil să
- 20 deservească multiple solicitări de citire și scriere simultane primite din surse multiple (procesoare și blocuri hardware), folosind instanțe RAM fizice multiple și asigurând acces arbitrat la acestea pentru a deservi surse multiple.
- [0181] Utilizarea unei interfețe de programare pentru aplicații (API) și partiționarea datelor la nivel de aplicație sunt importante pentru a asigura reducerea concurenței în rândul
- 25 procesoarelor sau între procesoare și acceleratoarele de filtru pentru accesarea blocurilor RAM fizice, crescând astfel lățimea de bandă de date pentru procesoare și hardware în cazul unei configurații date a subsistemului de memorie.
- [0182] În unele implementări, dispozitivul de procesare paralelă 400 poate fi situat într-un dispozitiv electronic. FIG. 26 ilustrează un dispozitiv electronic care include un dispozitiv
- 30 de procesare paralelă conform anumitor implementări. dispozitivul electronic 2600 poate include un procesor 2602, memorie 2604, una sau mai multe interfețe 2606 și un dispozitiv de procesare paralelă 400.

- [0183] Dispozitivul electronic 2600 poate avea memorie 2604, cum ar fi un suport care poate fi citit de computer, memorie flash, o unitate de disc magnetic, o unitate optică, o memorie programabilă doar în citire (PROM) și/sau o memorie doar în citire (ROM). Dispozitivul electronic 2600 poate fi configurat cu unul sau mai multe procesoare 2602 care
- 5 procesează instrucțiuni și rulează software care poate fi stocat în memorie 2604. Procesorul 2602 poate comunica, de asemenea, cu memoria 2604 și cu interfețele 2606 pentru comunicarea cu alte dispozitive. Procesorul 2602 poate fi orice procesor aplicabil, cum ar fi un sistem pe cip care combină o unitate centrală de procesare, un procesor de aplicații și memorie flash sau un procesor de calcul cu set redus de instrucțiuni (RISC).
- 10 [0184] În unele implementări, compilatorul 1208 și planificatorul 1210 pot fi implementate în software stocat în memorie 2604 și rulează pe procesor 2602. Memoria 2604 poate fi un suport netranzitoriu care poate fi citit de un computer, memorie flash, o unitate de disc magnetic, o unitate optică, o memorie programabilă numai în citire (PROM), o memorie numai în citire (ROM) sau orice altă memorie sau combinație de memorii. Software-ul poate
- 15 rula pe un procesor capabil să execute instrucțiuni pentru computer sau cod pentru computer. De asemenea, procesorul ar putea fi implementat în hardware folosind un circuit integrat specific aplicației (application specific integrated circuit – ASIC), o matrice logică programabilă (programmable logic array - PLA), o matrice de porți logice reprogramabilă (field programmable gate array – FPGA) sau orice alt circuit integrat.
- 20 [0185] În unele implementări, compilatorul 1208 poate fi implementat într-un dispozitiv de calcul separat care comunică cu dispozitivul electronic 2600 prin interfața 2606. De exemplu, compilatorul 1208 poate funcționa într-un server comunicând cu dispozitivul electronic 2600.
- [0186] Interfețele 2606 pot fi implementate în hardware sau software. Interfețele 2606 se
- 25 pot utiliza pentru a recepționa date și informații de control din rețea precum și din surse locale, cum ar fi o telecomandă de televizor. Dispozitivul electronic poate pune la dispoziție o varietate de interfețe de utilizator, cum ar fi o tastatură, un ecran tactil, un trackball, un touchpad și/sau un mouse. De asemenea, dispozitivul electronic poate include difuzoare și un dispozitiv de afișare în anumite implementări.
- 30 [0187] În unele implementări, un element de procesare din dispozitivul de procesare paralelă 400 poate include un cip integrat capabil să execute instrucțiuni pentru computer sau cod pentru computer. De asemenea, procesorul ar putea fi implementat în hardware folosind un circuit integrat specific aplicației (application specific integrated circuit – ASIC), o

matrice logică programabilă (programmable logic array - PLA), o matrice de porți logice reprogramabilă (field programmable gate array – FPGA) sau orice alt circuit integrat.

[0188] În unele implementări, dispozitivul de procesare paralelă 400 poate fi implementat ca sistem pe cip (system on chip – SOC). În alte implementări, unul sau mai multe blocuri din

5 dispozitivul de procesare paralelă pot fi implementate sub forma unui cip separat, iar dispozitivul de procesare paralelă poate fi realizat ca sistem într-un singur modul (system in package – SIP). În unele implementări, dispozitivul de procesare paralelă 400 poate fi folosit pentru aplicații de procesare a datelor. Aplicațiile de procesare a datelor pot include aplicații de procesare de imagini și/sau aplicații de procesare video. Aplicațiile de procesare de

10 imagini pot include un proces de procesare de imagini, inclusiv o operație de filtrare de imagini; aplicațiile de procesare video pot include o operație de decodificare video, o operație de codificare video, o operație de analiză video pentru detectarea mișcării sau a obiectelor din datele video. Aplicațiile suplimentare ale invenției de față includ învățarea automată și clasificarea pe baza unei secvențe de imagini, obiecte sau date video și aplicații de realitate

15 augmentată, inclusiv cele în care o aplicație pentru jocuri extrage date geometrice din mai multe vizualizări cu camera, inclusiv camere de profunzime, și extrage date din multiplele vizualizări din care poate fi extrasă geometria wireframe (de exemplu, printr-un nor de puncte) pentru umbrirea ulterioară la vertex de către un GPU.

[0189] Dispozitivul electronic 2600 poate include un dispozitiv mobil, cum ar fi un

20 telefon celular. Dispozitivul mobil poate să comunice cu o multitudine de rețele de acces radio folosind o multitudine de tehnologii de acces, precum și cu rețele de comunicații cablate. Dispozitivul mobil poate fi un smartphone care oferă capacități avansate, cum ar fi editarea de texte, navigarea pe internet, jocuri, capacități de cititor de cărți în format electronic, precum și o tastatură completă. Dispozitivul mobil poate rula un sistem de operare

25 precum Symbian OS, iPhone OS, Blackberry de la RIM, Windows Mobile, Linux, Palm WebOS și Android. Ecranul poate fi un ecran tactil care se poate folosi pentru introducerea datelor în telefonul mobil, ecranul putând fi folosit în locul tastaturii complete. Dispozitivul mobil poate fi capabil să ruleze aplicații sau să comunice cu aplicații puse la dispoziție de servere din rețeaua de comunicații. Dispozitivul mobil poate primi actualizări și alte

30 informații de la aceste aplicații în rețea.

[0190] De asemenea, dispozitivul electronic 2600 poate include numeroase alte dispozitive, cum ar fi televizoare, videoproiectoare, receptoare TV sau unități de recepție TV, videorecordere digitale (DVR), computere, netbookuri, laptopuri, tablete și orice alte

echipamente audiovideo care pot să comunice cu o rețea. De asemenea, dispozitivul electronic poate păstra în stiva sau în memoria sa coordonate de poziționare pe glob, informații de profil sau alte informații privind locația.

5 [0191] Se va aprecia că, deși au fost descrise aici mai multe configurații diferite, caracteristicile fiecăreia pot fi combinate avantajos într-o varietate de forme, pentru a obține un avantaj.

10 [0192] În specificația anterioară, aplicația a fost descrisă făcându-se trimiteri la exemple specifice. Este evident însă că ea se poate modifica în diferite moduri fără să ne îndepărtăm de la spiritul și domeniul general al invenției definite în revendicările anexate. De exemplu, conexiunile pot fi orice tip de conexiune adecvată pentru transferarea de semnale de la sau la nodurile, unitățile sau dispozitivele respective, de exemplu, prin dispozitive intermediare. În consecință, în lipsa altor precizări sau sugestii, conexiunile pot fi, de exemplu, conexiuni directe sau conexiuni indirecte.

15 [0193] Trebuie înțeles faptul că arhitecturile înfățișate aici au doar rol de exemplu și că, de fapt, se pot implementa numeroase alte arhitecturi care să realizeze aceeași funcționalitate. În sens abstract, dar clar, orice configurație de componente menită să atingă aceeași funcționalitate este efectiv „asociată”, astfel încât să se atingă funcționalitatea dorită. Prin urmare, oricare două componente de aici combinate pentru a realiza o anumită funcționalitate pot fi văzute ca fiind „asociate” unul cu celălalt astfel încât să se atingă funcționalitatea

20 dorită, indiferent de arhitecturi sau de componentele intermediare. Tot astfel, oricare două componente astfel asociate pot fi privite ca fiind „conectate operabil” sau „cuplate operabil” unul la celălalt pentru a obține funcționalitatea dorită.

[0194] În plus, cei avizați vor recunoaște faptul că granițele dintre funcționalitatea operațiilor descrise mai sus au doar un caracter ilustrativ. Funcționalitatea mai multor operații

25 poate fi combinată într-o singură operație și/sau funcționalitatea unei singure operații poate fi distribuită în operații suplimentare. Mai mult, implementările alternative pot include mai multe instanțe ale unei anumite operații, iar ordinea operațiilor poate fi modificată în diferite alte implementări.

[0195] Sunt posibile însă și alte modificări, variații și alternative. Specificațiile și

30 desenele trebuie privite, în mod similar, ca având un sens ilustrativ, nu restrictiv.

[0196] În revendicări, orice semne de referință plasate între paranteze nu vor fi interpretate ca limitând revendicarea. Cuvintele „care cuprinde” nu exclud prezența altor elemente sau a altor etape decât cele menționate într-o revendicare. Mai mult, termenii „un”

- și „o” sunt utilizați aici cu sensul de unu sau mai mult de unu. De asemenea, utilizarea unor expresii introductive precum „cel puțin unul/una” și „unul/una sau mai multe” în revendicări nu trebuie interpretate ca sugerând că introducerea unui alt element de revendicare de către articolele nehotărâte „un” sau „o” limitează vreo revendicare anume care conține respectivul
- 5 element de revendicare introdus la invenții care conțin un singur asemenea element, chiar și atunci când aceeași revendicare include expresiile introductive „unul/una sau mai multe” sau „cel puțin un/o” și articolele nehotărâte precum „un” sau „o”. Același lucru este valabil și pentru utilizarea articolelor hotărâte. În lipsa altor precizări, termenii precum „primul” și „al
- 10 termenii respectivi. Astfel, acești termeni nu au neapărat scopul de a indica prioritizarea temporală sau de altă natură a elementelor respective. Simplul fapt că anumite măsuri sunt repetate în revendicări care diferă între ele nu indică faptul că o combinație a acestor metode nu poate fi utilizată în mod avantajos.

[0197] Revendicăm:

116

REVENDICĂRI

1. Un dispozitiv de procesare paralelă, dispozitivul de procesare cuprinzând:
o multitudine de elemente de procesare, fiecare configurat să execute instrucțiuni;
5 un subsistem de memorie care cuprinde o multitudine de zone de memorie, inclusiv o primă zonă de memorie asociată unuia dintre multiplele elemente de procesare, această primă zonă de memorie cuprinzând o multitudine de blocuri de memorie cu acces aleatoriu (RAM), fiecare bloc având porturi individuale de citire și de scriere; și
un sistem de interconectare configurat să cupleze multitudinea de elemente de
10 procesare și subsistemul de memorie, acest sistem de interconectare incluzând:
o interconectare locală configurată să cupleze prima zonă de memorie și unul dintre multiplele elemente de procesare; și
o interconectare globală configurată să cupleze prima zonă de memorie și restul multiplelor elemente de procesare.
15
2. Dispozitivul de procesare din revendicarea 1, în care unul dintre multiplele blocuri de memorie RAM este asociat unui bloc de arbitrare, acest bloc de arbitrare fiind configurat să primească solicitări de accesare a memoriei din partea unuia dintre multiplele elemente de procesare și să permită accesul unuia dintre multiplele elemente de procesare la unul dintre
20 multiplele blocuri de memorie RAM.
3. Dispozitivul de procesare din revendicarea 2, blocul de arbitrare fiind configurat să acorde acces la unul dintre multiplele blocuri de memorie RAM conform unui algoritm round robin (coadă circulară de priorități).
25
4. Dispozitivul de procesare din revendicarea 2, blocul de arbitrare cuprinzând un detector de conflicte configurat să monitorizeze solicitările de accesare a memoriei vizând unul dintre multiplele blocuri de memorie RAM și să determine dacă există două sau mai multe dintre multiplele elemente de procesare care încearcă să acceseze simultan același bloc
30 dintre multiplele blocuri de memorie RAM.

5. Dispozitivul de procesare din revendicarea 4, detectorul de conflicte fiind cuplat la o multitudine de decodificatoare de adrese, fiecare dintre aceste multiple decodificatoare de adrese fiind cuplat la unul dintre multiplele elemente de procesare și fiind configurat să determine dacă unul dintre multiplele elemente de procesare încearcă să acceseze unul dintre multiplele blocuri de memorie RAM asociate blocului de arbitrare.
6. Dispozitivul de procesare din revendicarea 1, multitudinea de elemente de procesare cuprinzând cel puțin un procesor vectorial și cel puțin un accelerator hardware.
7. Dispozitivul de procesare din revendicarea 6, care mai cuprinde în plus o multitudine de controlere ale zonelor de memorie, fiecare controler fiind configurat să asigure accesul la una dintre multiplele zone de memorie.
8. Dispozitivul de procesare din revendicarea 7, sistemul de interconectare cuprinzând o primă magistrală configurată să asigure comunicarea dintre cel puțin un procesor vectorial și subsistemul de memorie.
9. Dispozitivul de procesare din revendicarea 8, sistemul de interconectare cuprinzând o a doua magistrală configurată să asigure comunicarea dintre cel puțin un accelerator hardware și subsistemul de memorie.
10. Dispozitivul de procesare din revendicarea 9, al doilea sistem de magistrală cuprinzând un filtru de solicitări de adrese ale zonelor de memorie configurat să medieze comunicarea dintre cel puțin un accelerator hardware și subsistemul de memorie, recepționând o solicitare de accesare a memoriei din partea cel puțin unui accelerator hardware și permițând cel puțin unui accelerator hardware să acceseze subsistemul de memorie.
11. Dispozitivul de procesare din revendicarea 1, unul dintre multiplele dispozitive de procesare cuprinzând o memorie-tampon pentru a mări debitul sistemului de memorie, numărul de elemente din cadrul memoriei-tampon fiind mai mare decât numărul de cicluri necesare pentru preluarea datelor din subsistemul de memorie.

12. O metodă de operare a unui sistem de procesare paralelă, metoda cuprinzând:
- asigurarea unei multitudini de elemente de procesare, inclusiv un prim element de procesare și un al doilea element de procesare, fiecare dintre multiplele elemente de procesare
- 5 fiind configurat să execute instrucțiuni;
- asigurarea unui subsistem de memorie care cuprinde o multitudine de zone de memorie, inclusiv o primă zonă de memorie asociată primului element de procesare, această primă zonă de memorie cuprinzând o multitudine de blocuri de memorie cu acces aleatoriu (RAM), fiecare bloc având porturi individuale de citire și de scriere;
- 10 recepționarea – de către un bloc de arbitrare asociat unuia dintre multiplele blocuri de memorie RAM printr-o interconectare locală a unui sistem de interconectare – unei prime solicitări de accesare a memoriei din partea primului element de procesare; și
- trimiterea – de către blocul de arbitrare prin interconectarea globală – unui prim mesaj de autorizare către primul element de procesare, pentru a autoriza accesul primului
- 15 element de procesare la unul dintre multiplele blocuri de memorie RAM.
13. Metoda din revendicarea 12, cuprinzând în plus:
- recepționarea – de către blocul de arbitrare printr-o interconectare globală a sistemului de interconectare – unei a doua solicitări de accesare a memoriei din partea unui al doilea
- 20 element de procesare; și
- trimiterea – de către blocul de arbitrare prin interconectarea globală – unui al doilea mesaj de autorizare către al doilea element de procesare, pentru a autoriza accesul celui de al doilea element de procesare la unul dintre multiplele blocuri de memorie RAM.
- 25 14. Metoda din revendicarea 12, cuprinzând în plus trimiterea – de către blocul de arbitrare – unei multitudini de mesaje de autorizare către multitudinea de elemente de procesare, pentru a autoriza accesul la unul dintre multiplele blocuri de memorie RAM conform unui algoritm round robin.
- 30 15. Metoda din revendicarea 12, cuprinzând în plus:
- monitorizarea – de către un detector de conflicte din blocul de arbitrare – solicitărilor de accesare a memoriei vizând unul dintre multiplele blocuri RAM; și

identificarea situației în care două sau mai multe dintre multiplele elemente de procesare încearcă să acceseze simultan același bloc dintre multiplele blocuri de memorie RAM.

5 16. Metoda din revendicarea 12, multitudinea de elemente de procesare cuprinzând cel puțin un procesor vectorial și cel puțin un accelerator hardware.

10 17. Metoda din revendicarea 16, cuprinzând în plus asigurarea unei multitudini de controlere ale zonelor de memorie, fiecare controler fiind configurat să asigure accesul la una dintre multiplele zone de memorie.

15 18. Metoda din revendicarea 17, cuprinzând în plus asigurarea comunicării dintre cel puțin un procesor vectorial și subsistemul de memorie printr-un prim sistem de magistrală al sistemului de interconectare.

20 19. Metoda din revendicarea 18, cuprinzând în plus asigurarea comunicării dintre cel puțin un accelerator hardware și subsistemul de memorie printr-un al doilea sistem de magistrală al sistemului de interconectare.

25 20. Metoda din revendicarea 19, al doilea sistem de magistrală cuprinzând un filtru de solicitări de adrese ale zonelor de memorie configurat să medieze comunicarea dintre cel puțin un accelerator hardware și subsistemul de memorie, recepționând o solicitare de accesare a memoriei din partea cel puțin unui accelerator hardware și permițând cel puțin unui accelerator hardware să acceseze subsistemul de memorie.

30 21. Un dispozitiv electronic cuprinzând:
un dispozitiv de procesare paralelă cuprinzând:

- o multitudine de elemente de procesare, fiecare configurat să execute instrucțiuni;

35 - un subsistem de memorie care cuprinde o multitudine de zone de memorie, inclusiv o primă zonă de memorie asociată unuia dintre multiplele elemente de

procesare, această primă zonă de memorie cuprinzând o multitudine de blocuri de memorie cu acces aleatoriu (RAM), fiecare bloc având porturi individuale de citire și de scriere; și

5 un sistem de interconectare configurat să cupleze multitudine de elemente de procesare și subsistemul de memorie, acest sistem de interconectare incluzând:

o interconectare locală configurată să cupleze prima zonă de memorie și unul dintre multiplele elemente de procesare; și

o interconectare globală configurată să cupleze prima zonă de memorie și restul multiplelor elemente de procesare;

10 un procesor, comunicând cu dispozitivul de procesare paralelă, configurat să ruleze un modul stocat în memorie care este configurat:

să recepționeze un grafic de flux de date asociat unui proces de procesare de date, graficul de flux de date cuprinzând o multitudine de noduri și o multitudine de arce care conectează două sau mai multe dintre multitudine de noduri, fiecare nod
15 identificând o operație și fiecare arc identificând o relație dintre nodurile conectate; și

să aloce un prim nod din multitudine de noduri la un prim element de procesare al dispozitivului de procesare paralelă, iar un al doilea nod din multitudine de noduri la un al doilea element de procesare din dispozitivul de procesare paralelă, paralelizând astfel operațiile asociate cu primul nod și cu al doilea nod.

20

22. Dispozitivul electronic din revendicarea 21, graficul de flux de date fiind prezentat într-un format de limbaj extensibil de marcare (XML).

23. Dispozitivul electronic din revendicarea 21, modulul fiind configurat să aloce primul
25 nod din multitudine de noduri la primul element de procesare pe baza unei performanțe anterioare a unui subsistem de memorie din dispozitivul de procesare paralelă.

24. Dispozitivul electronic din revendicarea 23, subsistemul de memorie al dispozitivului de procesare paralelă cuprinzând un contor configurat să contorizeze un număr de conflicte
30 de memorie într-o perioadă de timp predeterminată, iar performanța anterioară a subsistemului de memorie cuprinzând numărul de conflicte de memorie măsurat de contor.

25. Dispozitivul electronic din revendicarea 21, modulul fiind configurat să aloce primul nod din multitudinea de noduri la primul element de procesare în timp ce dispozitivul de procesare paralelă operează cel puțin o porțiune a graficului de flux de date.
- 5 26. Dispozitivul electronic din revendicarea 21, modulul fiind configurat să recepționeze o multitudine de grafice de flux de date și să aloce toate operațiile asociate multitudinii de grafice de flux la un singur element de procesare din dispozitivul de procesare paralelă.
- 10 27. Dispozitivul electronic din revendicarea 21, modulul fiind configurat să intercaleze accesările memoriei de către elementele de procesare, pentru a reduce conflictele de memorie.
28. Dispozitivul electronic din revendicarea 21, dispozitivul electronic incluzând un dispozitiv mobil.
- 15 29. Dispozitivul electronic din revendicarea 21, graficul de flux de date fiind specificat folosind o interfață de programare a aplicațiilor (API) asociată dispozitivului de procesare paralelă.
- 20 30. Dispozitivul electronic din revendicarea 21, modulul fiind configurat să furnizeze date de intrare tip imagine către multitudinea de elemente de procesare astfel:
divizând datele de intrare de tip imagine într-o multitudine de fâșii; și
furnizând una dintre multiplele fâșii de date de intrare tip imagine către unul dintre multiplele elemente de procesare.
- 25 31. Dispozitivul electronic din revendicarea 30, numărul de fâșii de date de intrare de tip imagine fiind identic cu numărul de elemente dintre multiplele elemente de procesare.

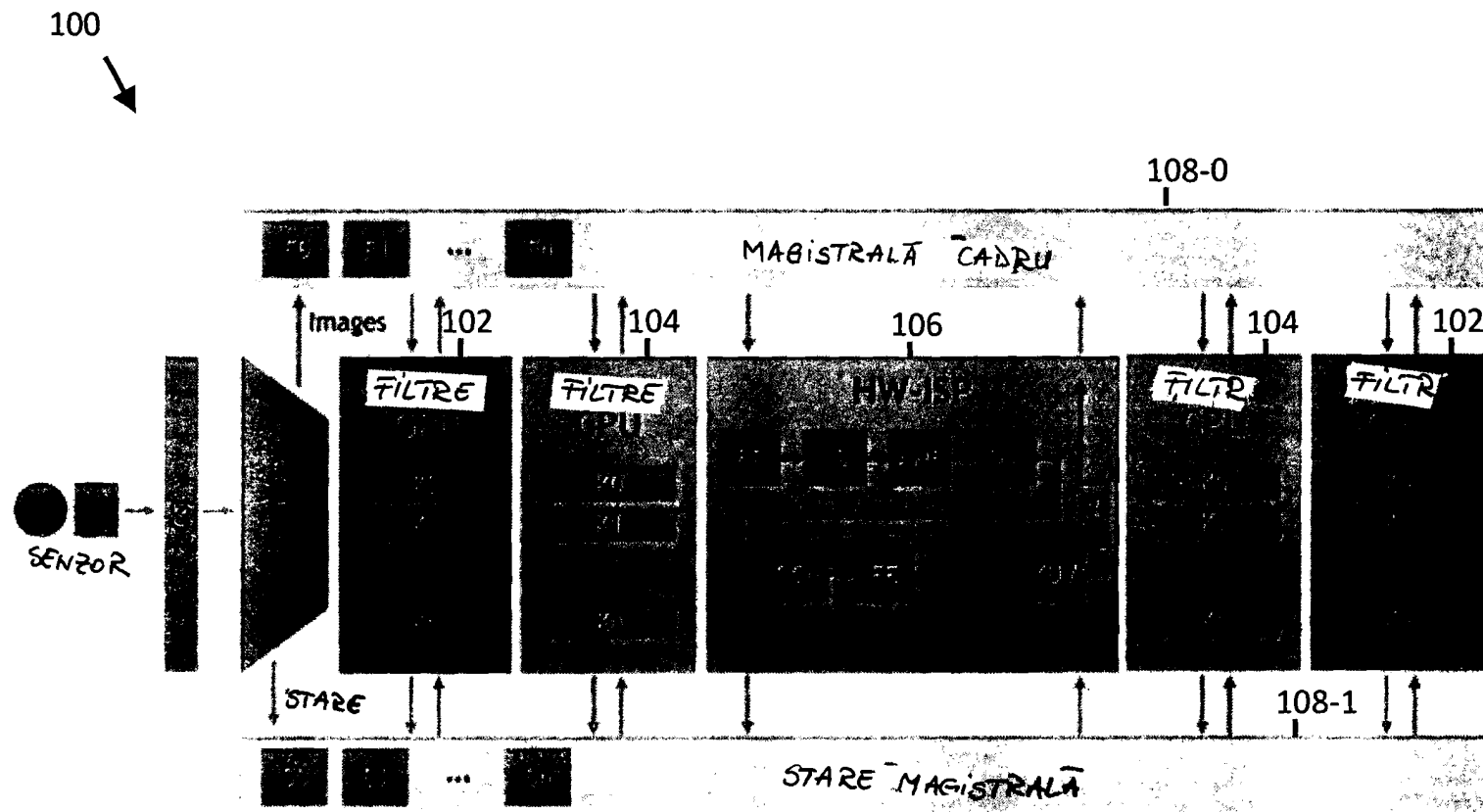
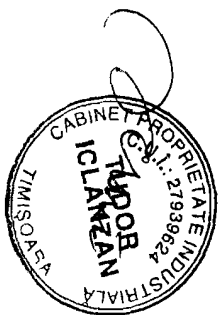


FIG. 1



2013-00812--
06-11-2013

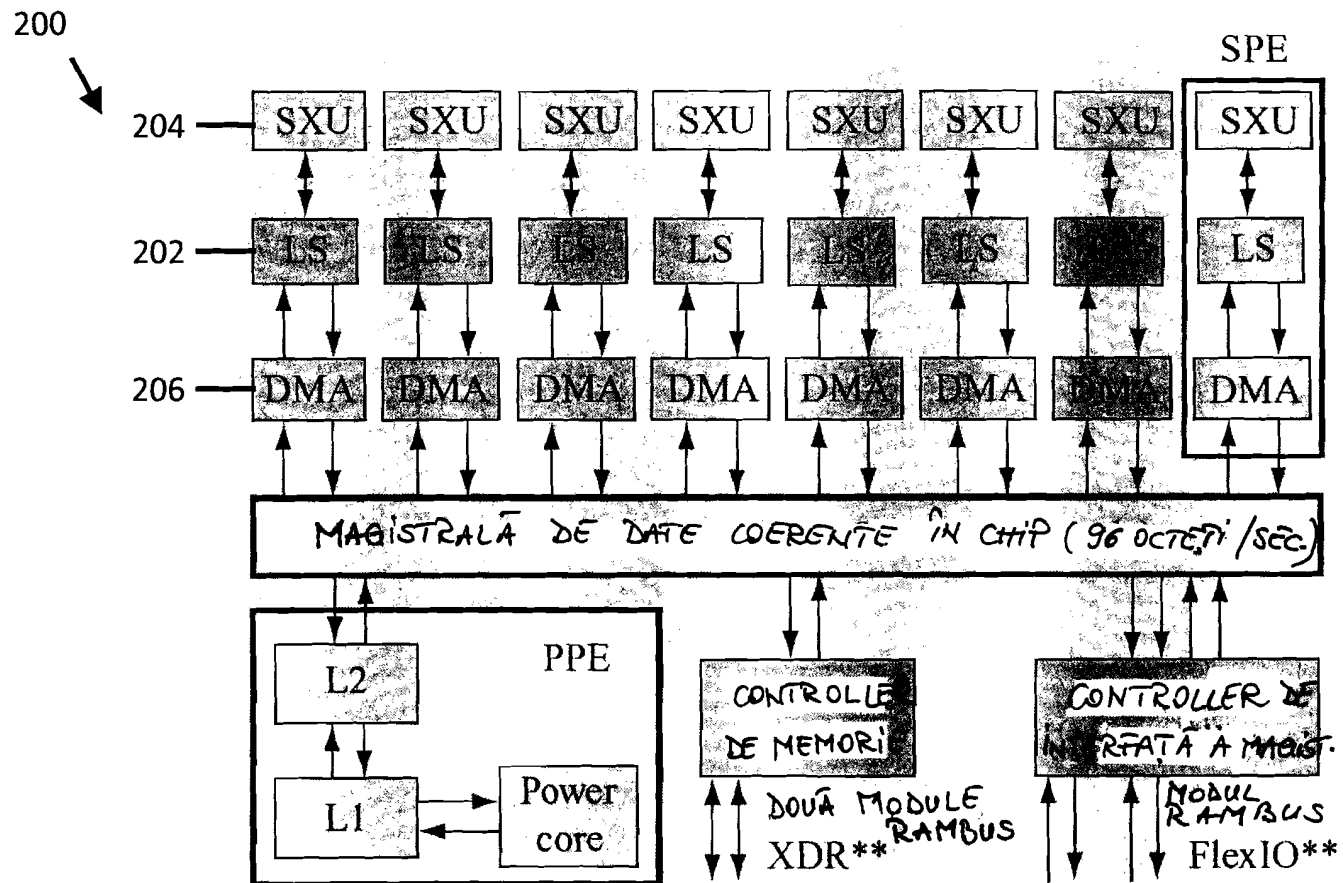
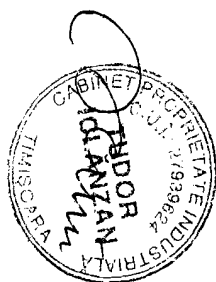


FIG. 2



9-2013-00812--
06-11-2013

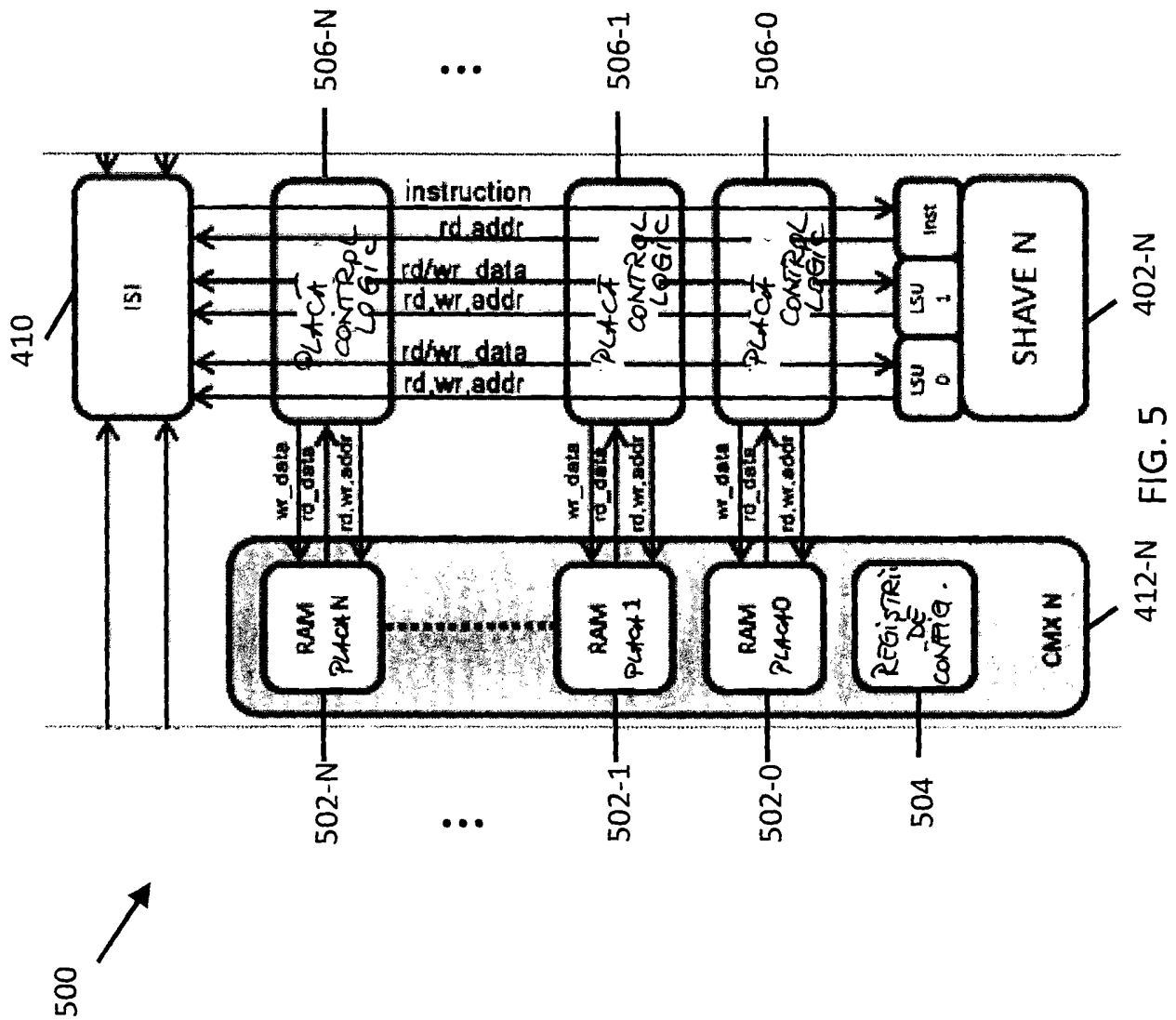


FIG. 5

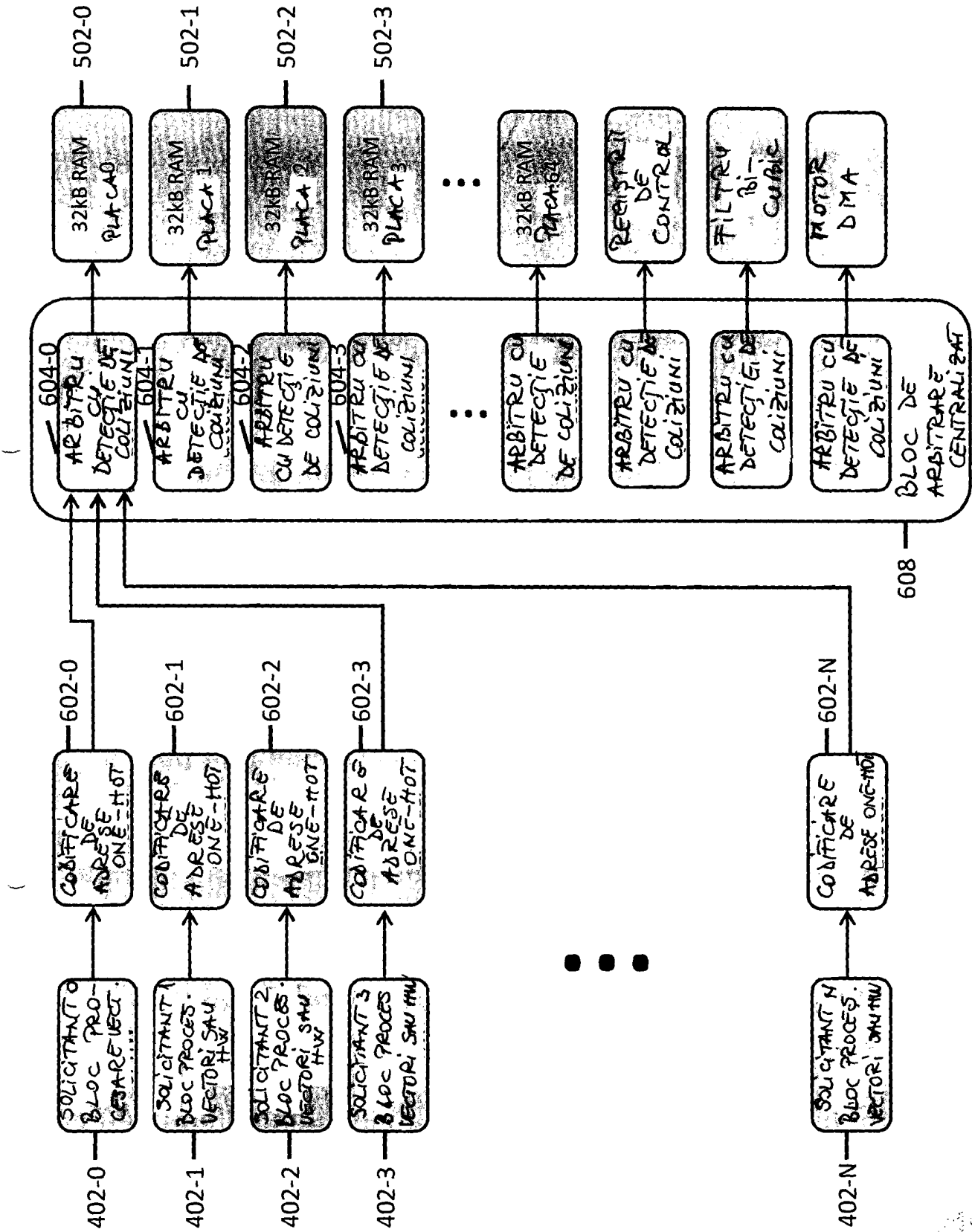


FIG. 6



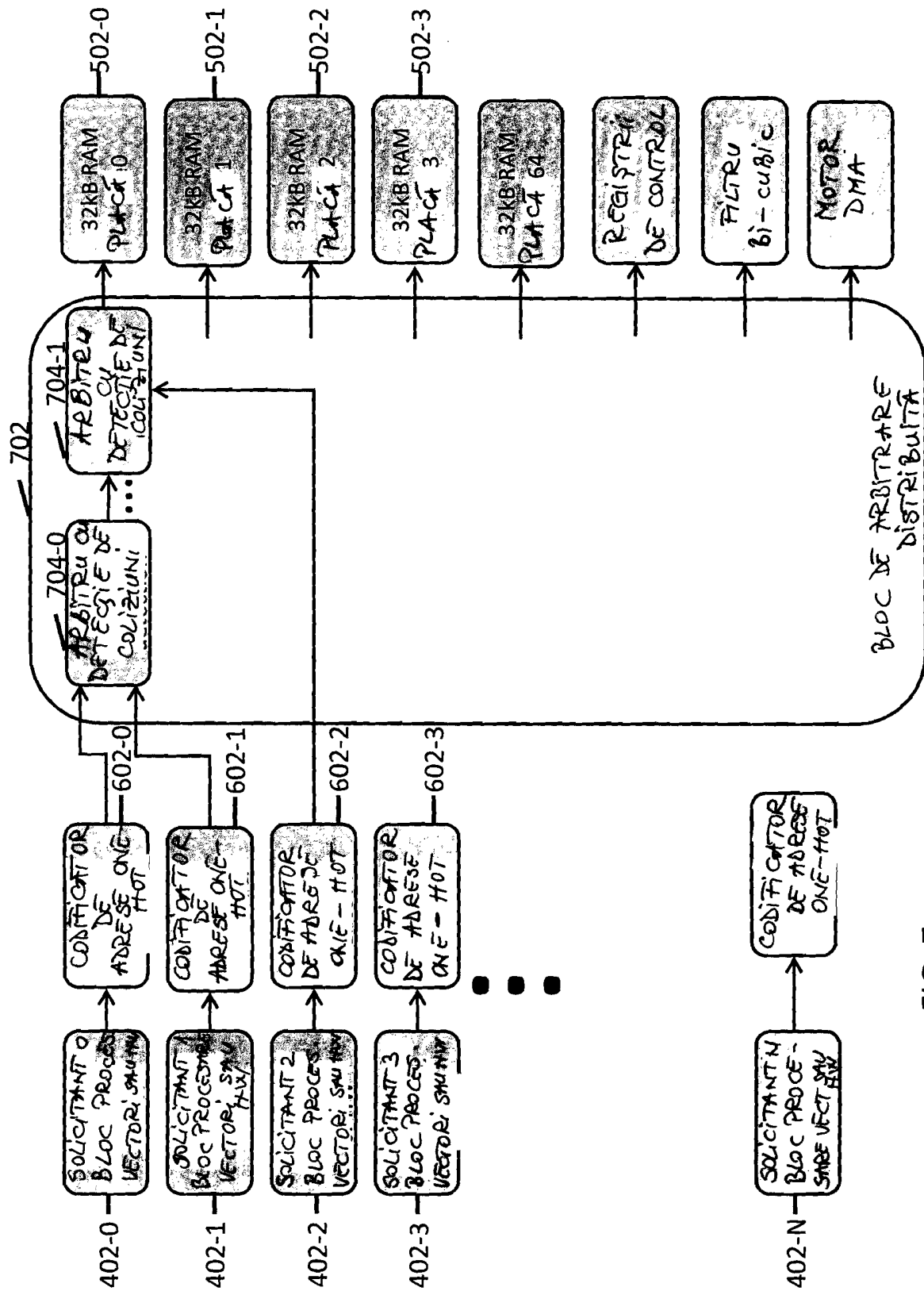


FIG. 7

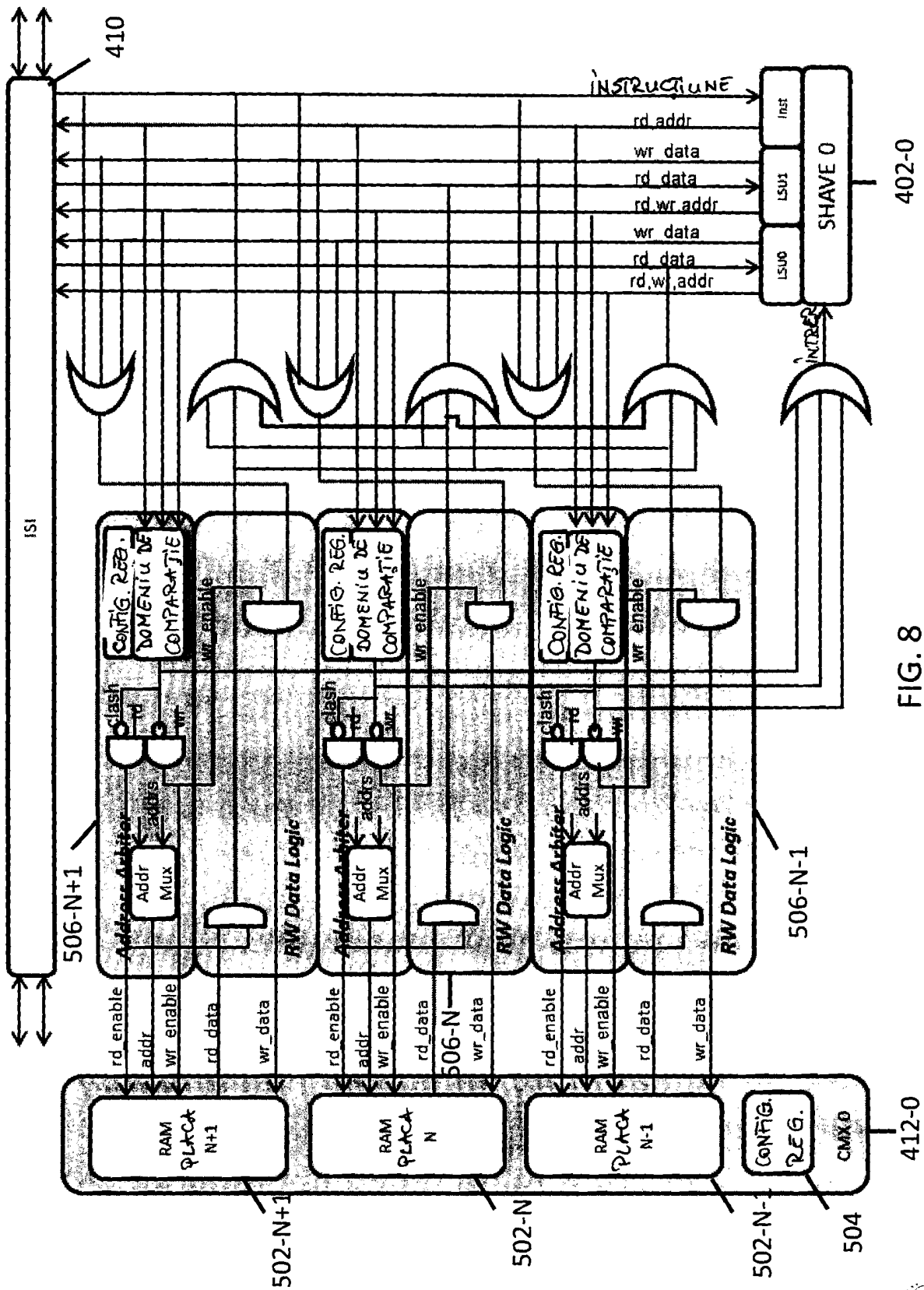
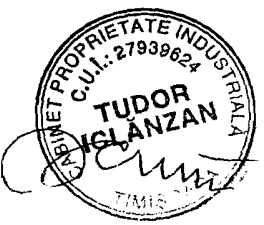
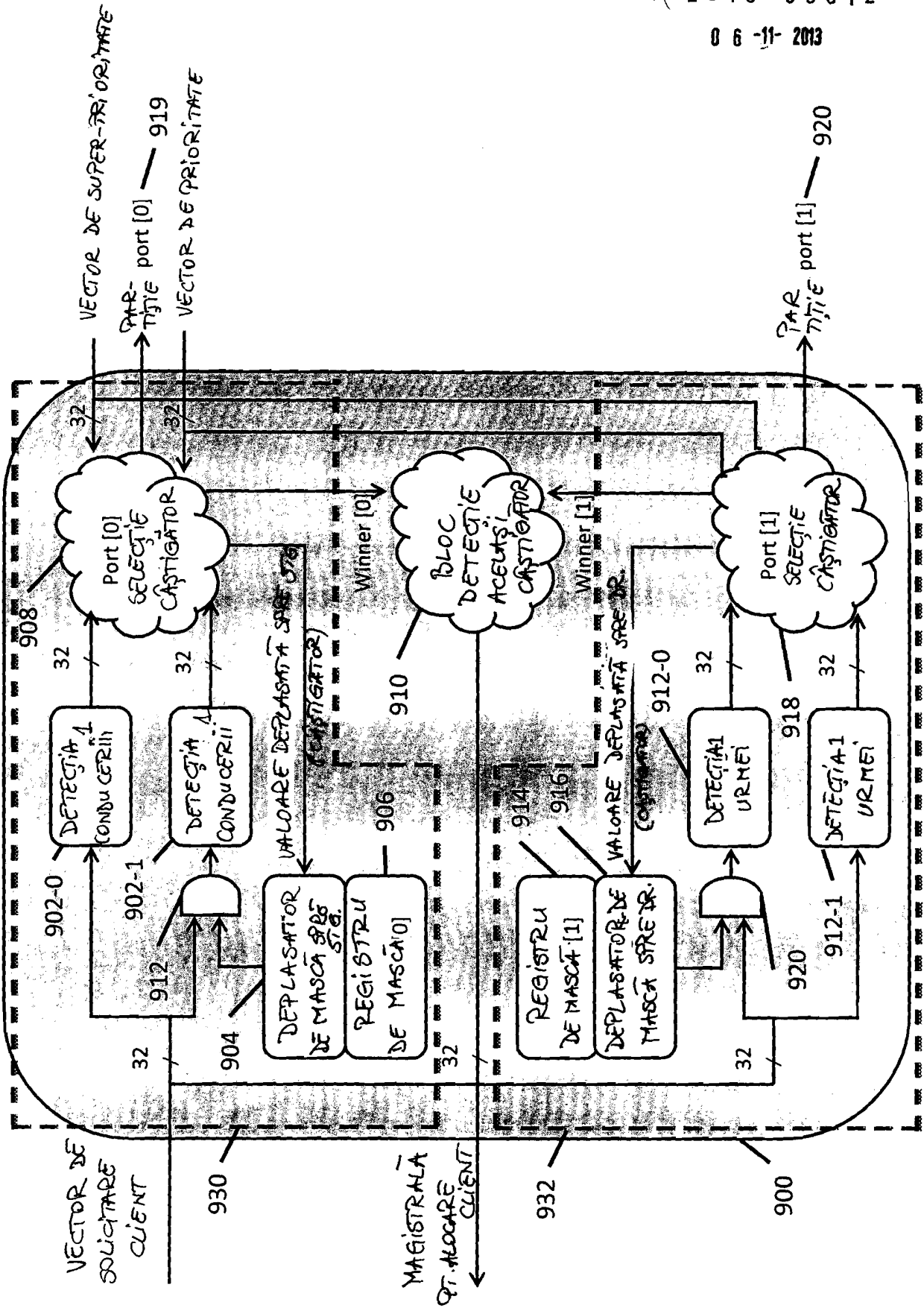


FIG. 8





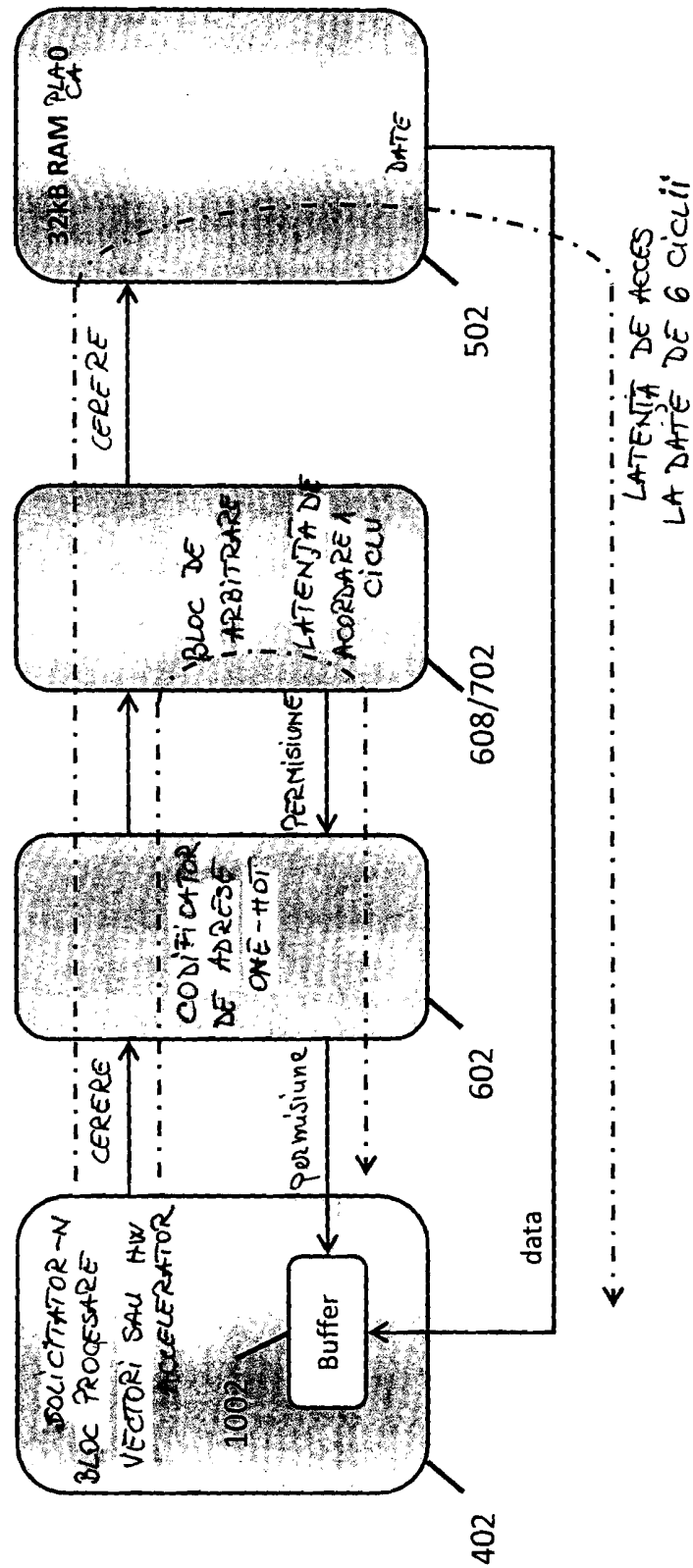
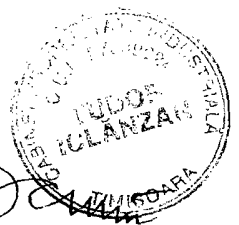


FIG. 10



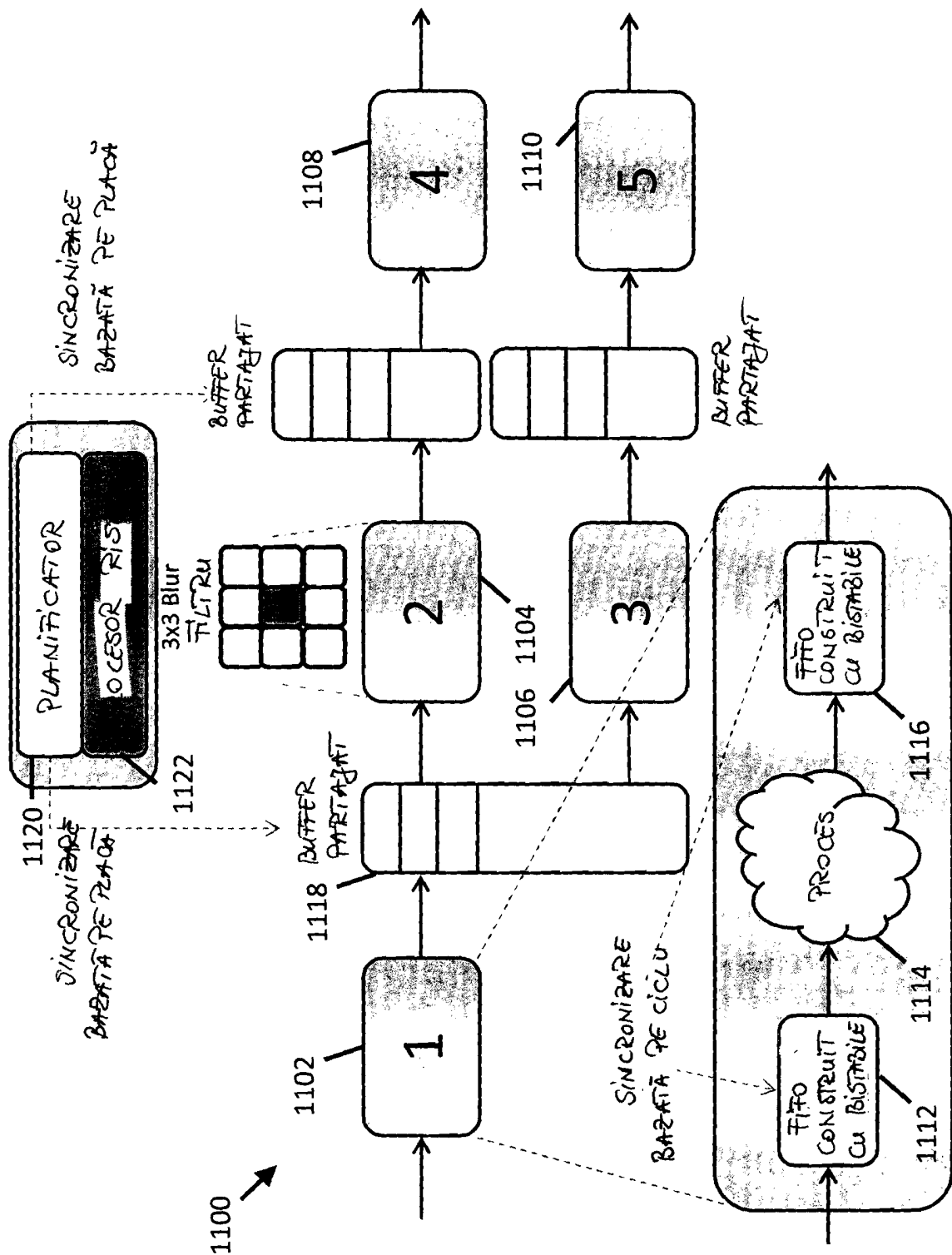


FIG. 11

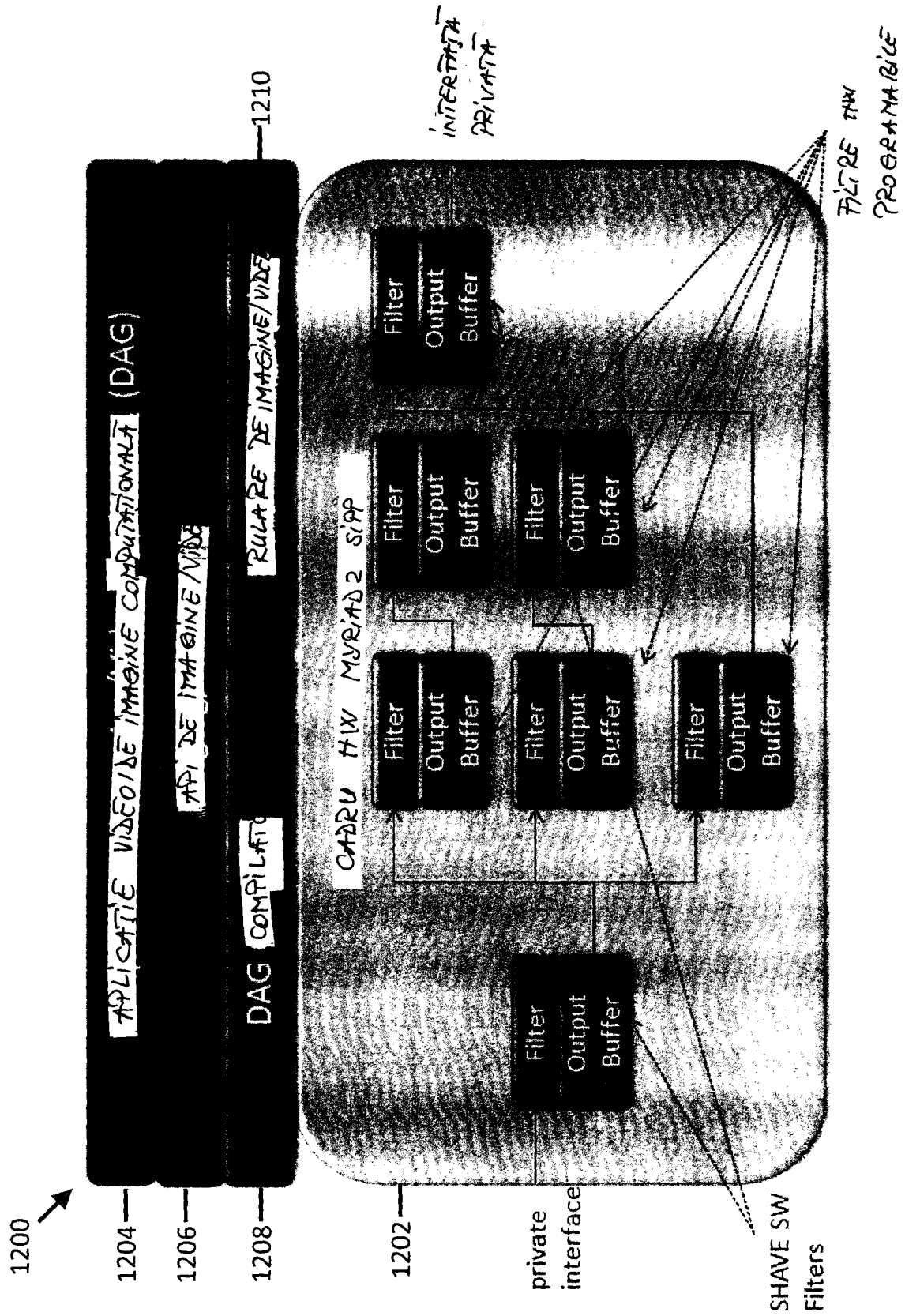


FIG. 12

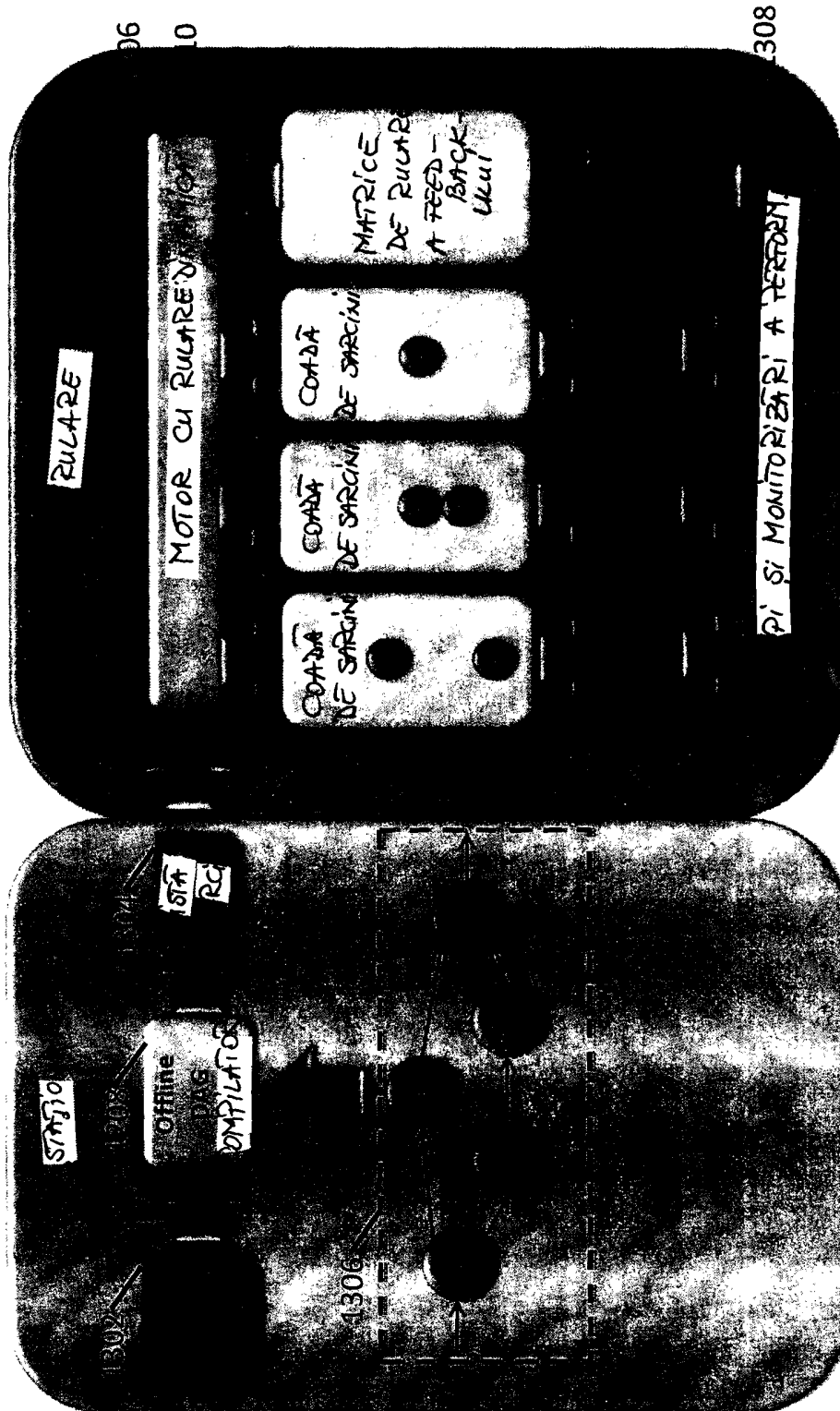


FIG. 13

COMPILATORUL SI PLANIFICATORUL CE RULEAZA
SPRÂJINĂ UNIREA SÂRCINILOR ÎN CONJUNCȚIE CU DIVIZIUNEA SÂRCINILOR PT A
MINIMIZA TRANSFERURILE DE DATE REDUNDANTE

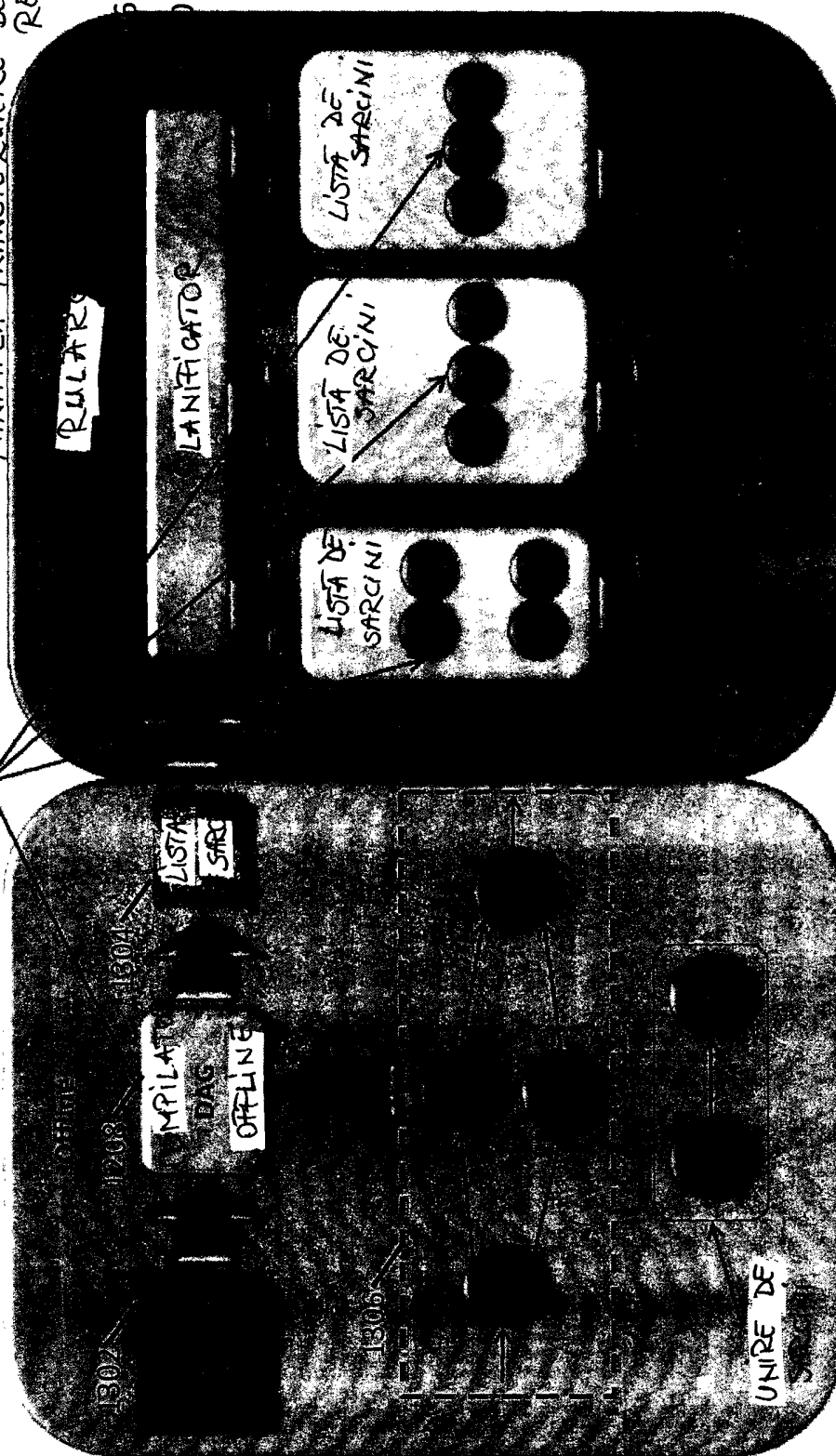


FIG. 14A

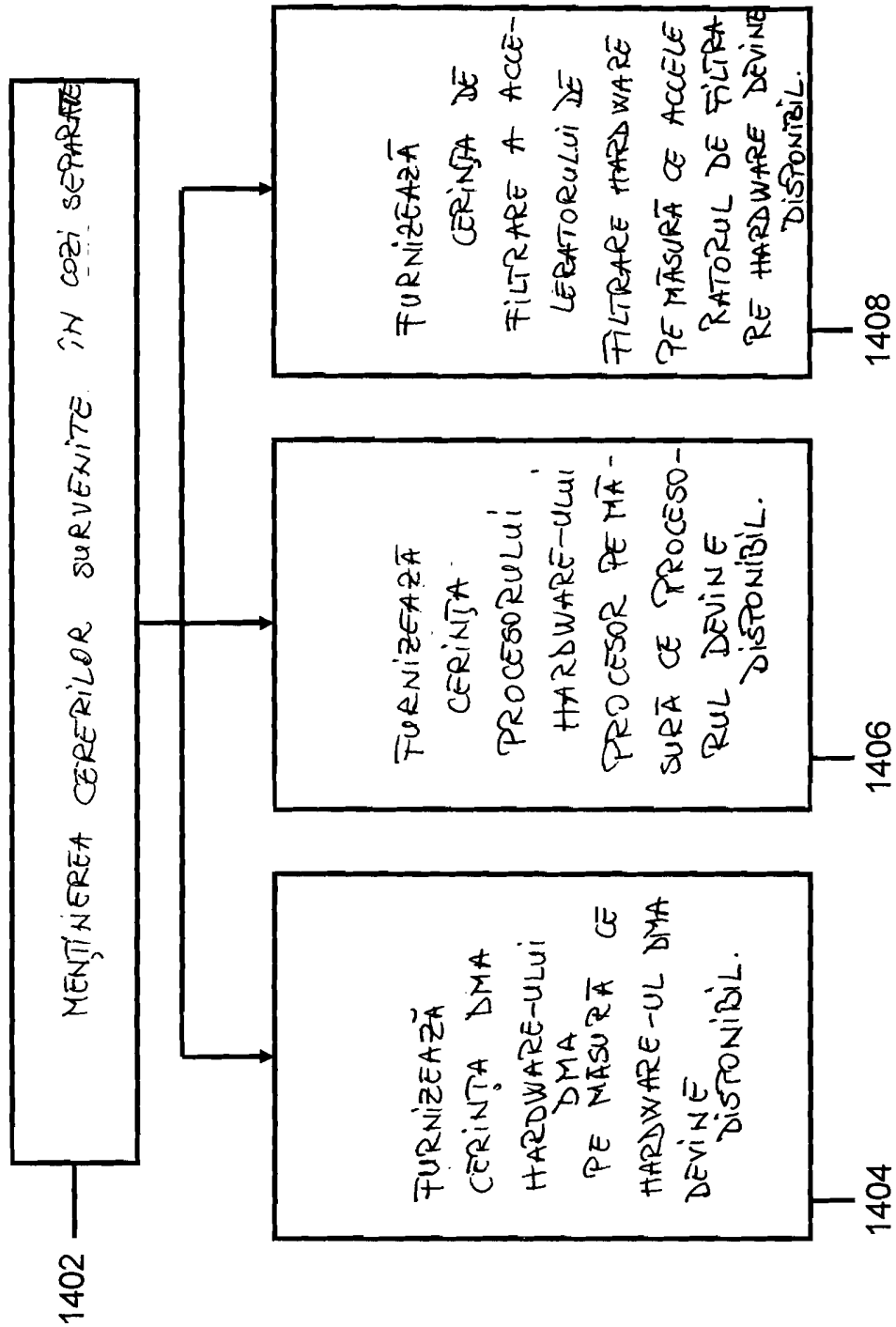
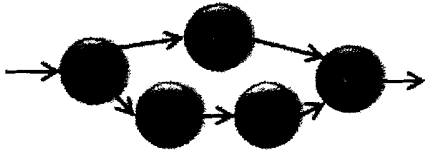
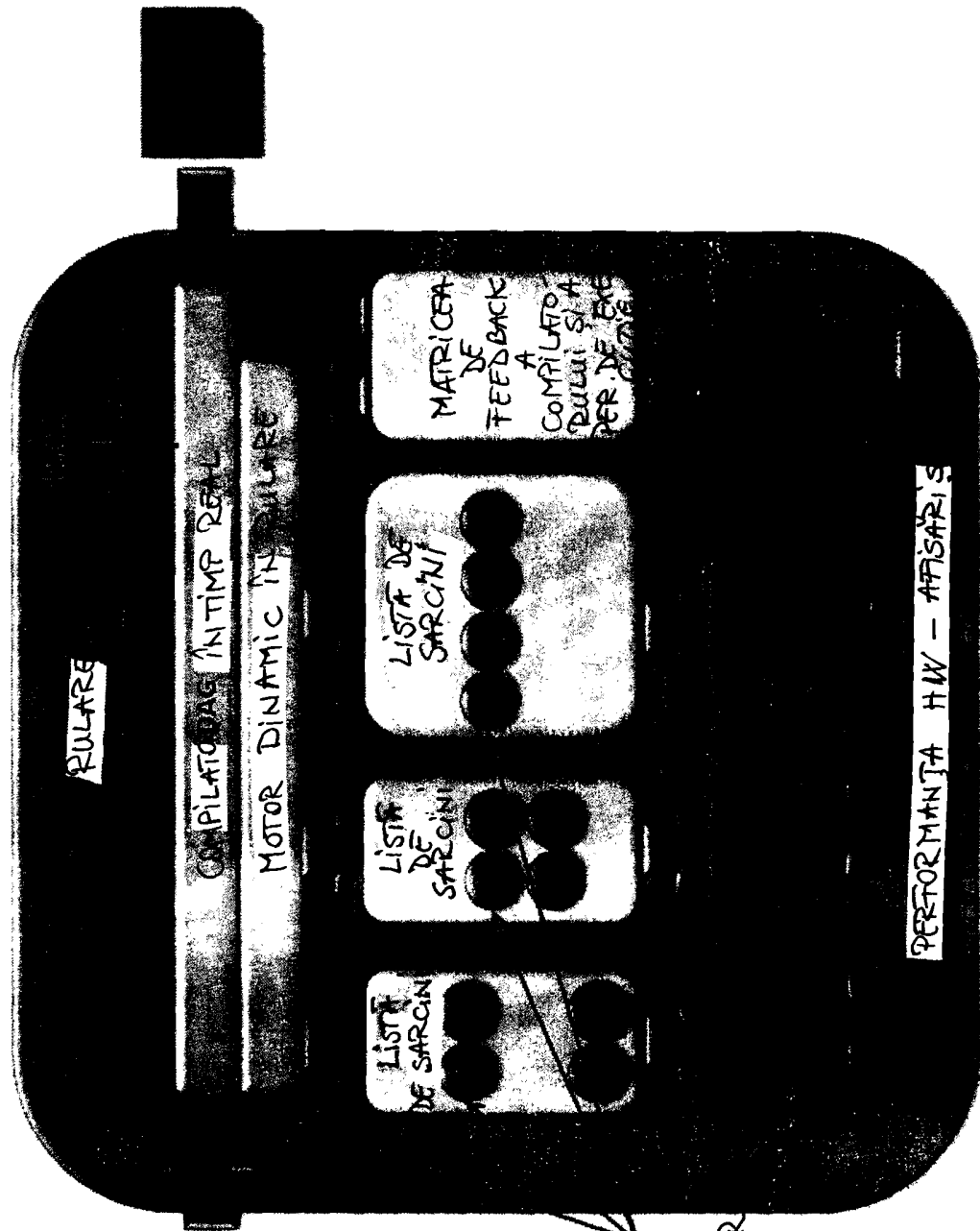


FIG. 14B





COMPILATORUL ȘI
MOTORUL DE RULARE
SPRIGINA ÎNTR-
ȚIREA SARCINILOR
PENTRU A EXPLOATA
PARALELISMUL DATELOR
DISPONIBILE

FIG. 15

Octavian

PLANIFICARE GENERATĂ DE
PLANIFICATOR ONLINE

PLANIFICARE GENERATĂ DE
PLANIFICATORUL DAG ONLINE

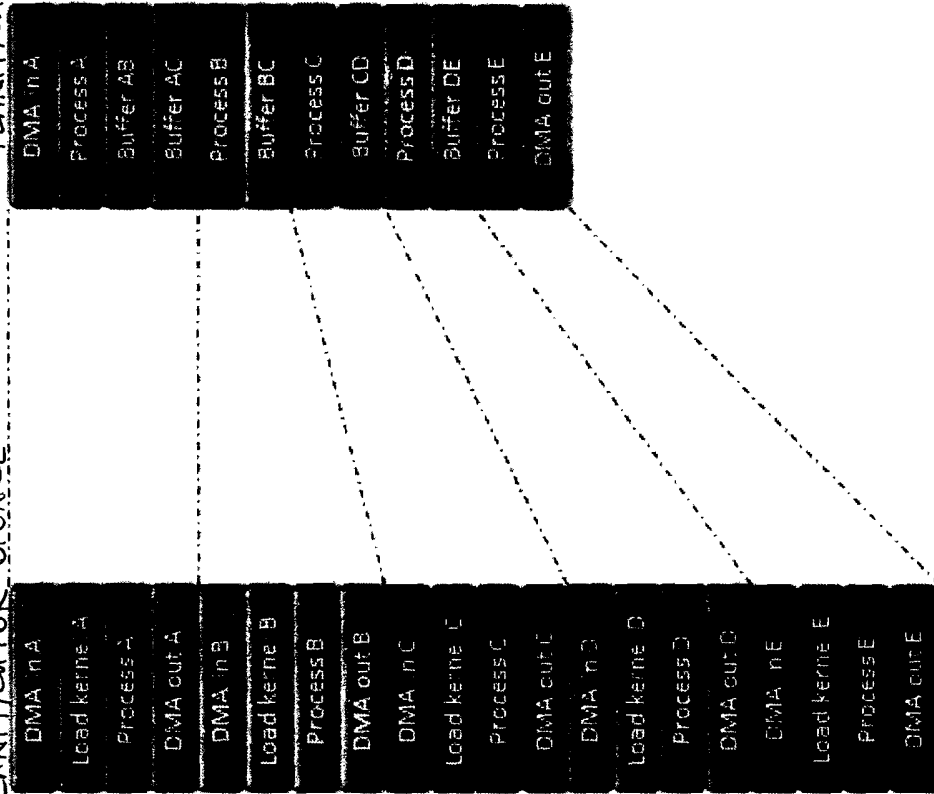
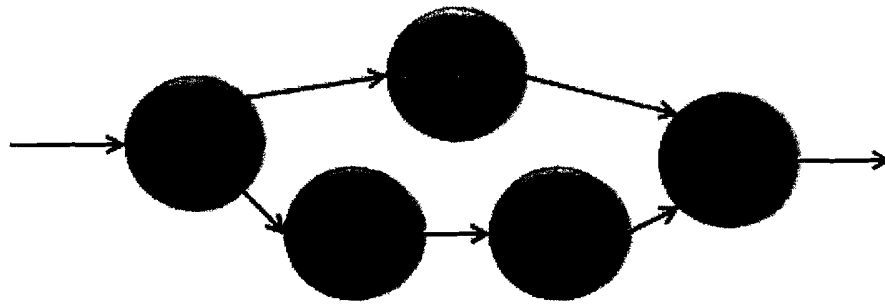


FIG. 16



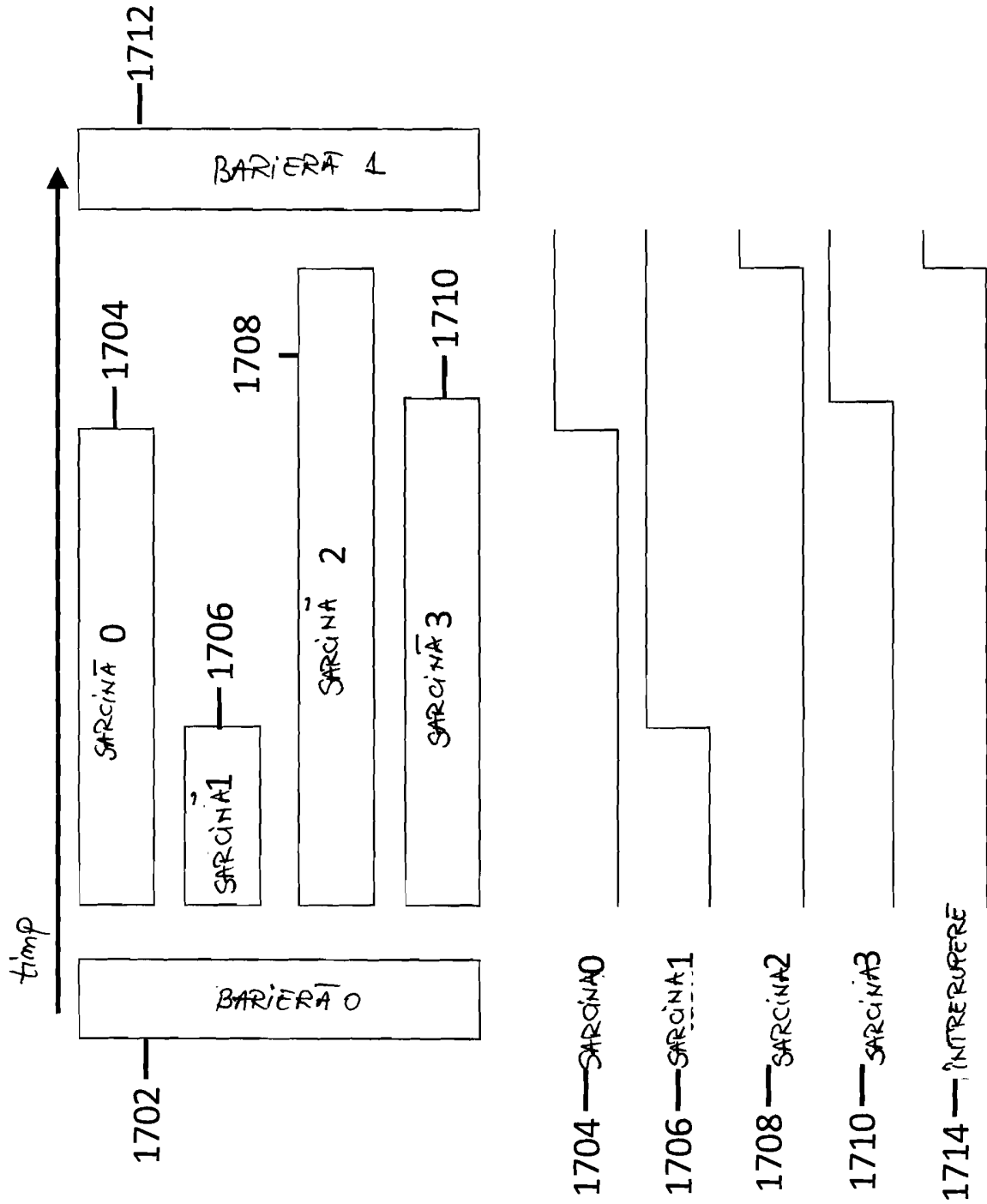


FIG. 17



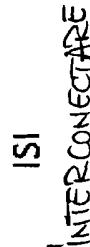


FIG. 19

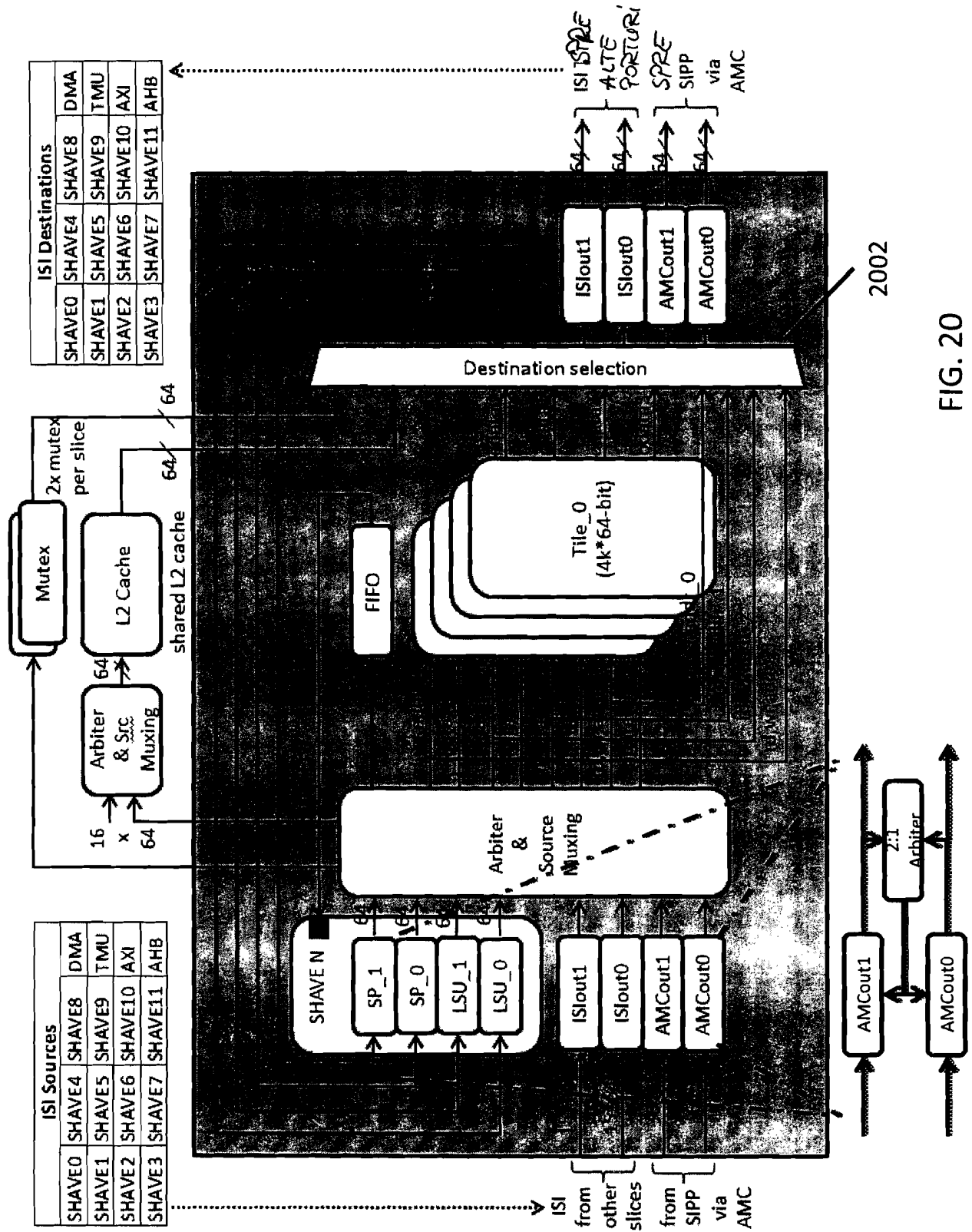
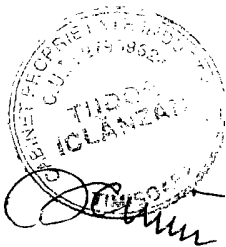


FIG. 20



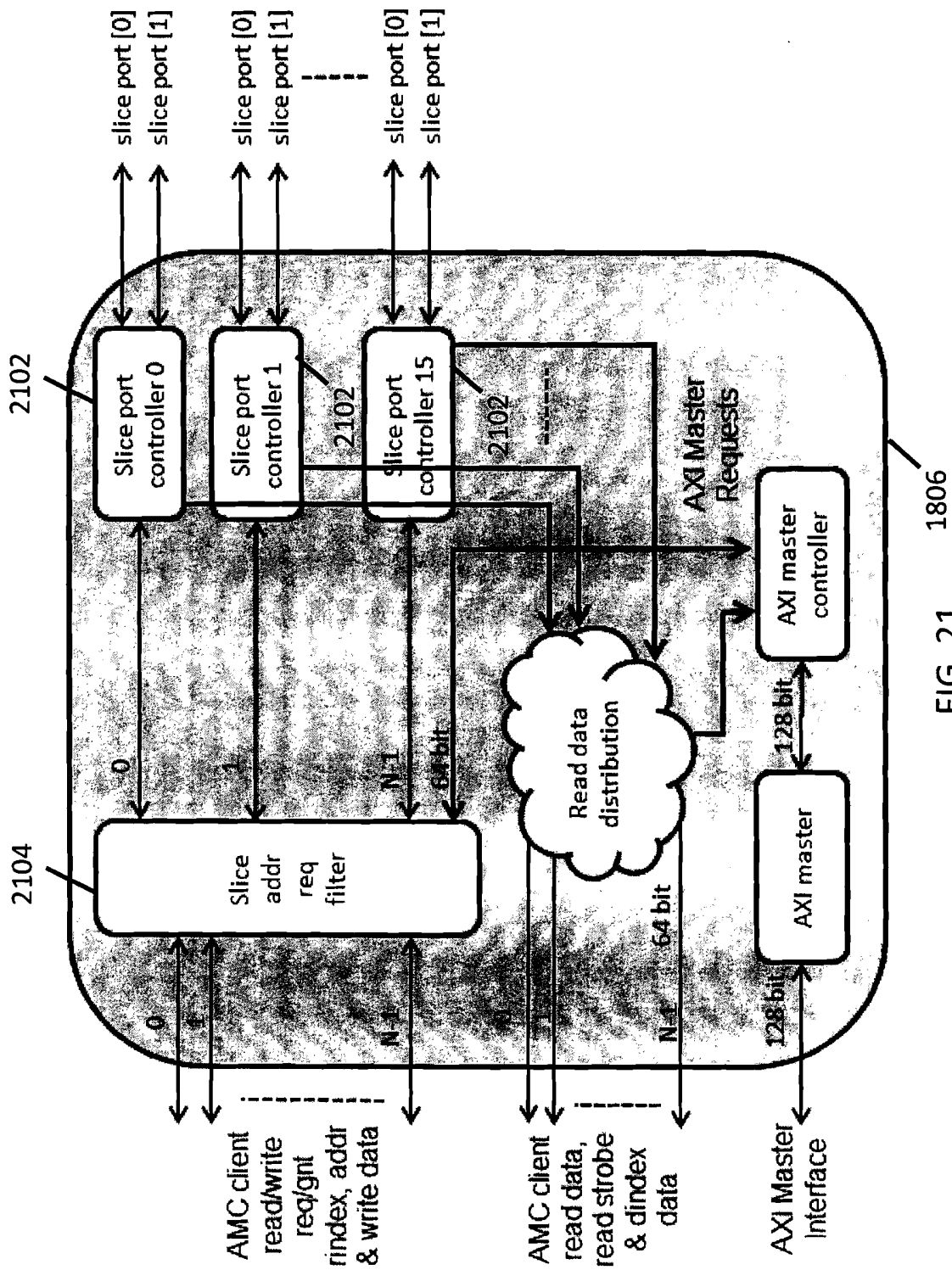


FIG. 21

Scanned

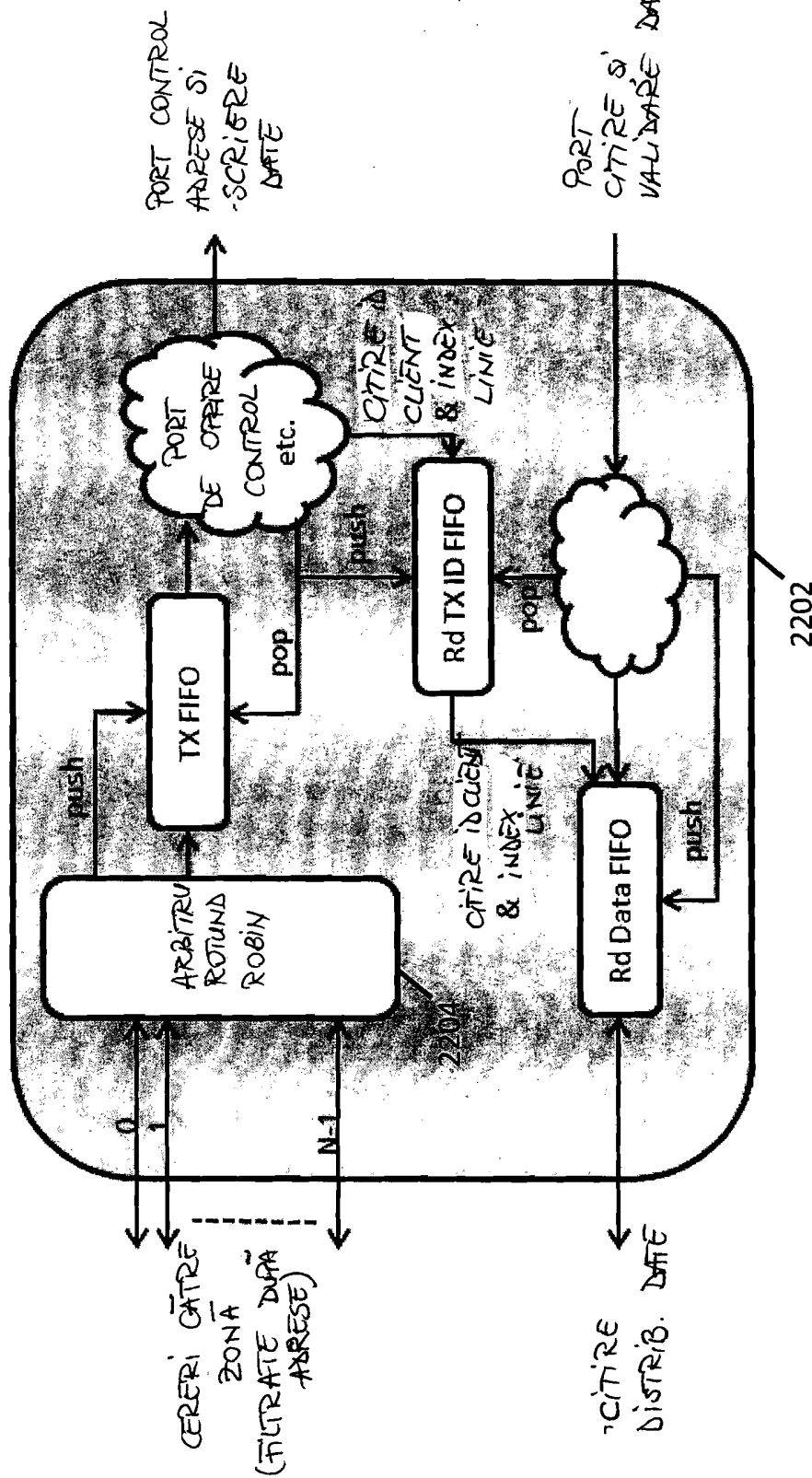
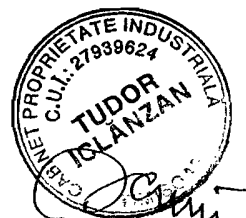


FIG. 22



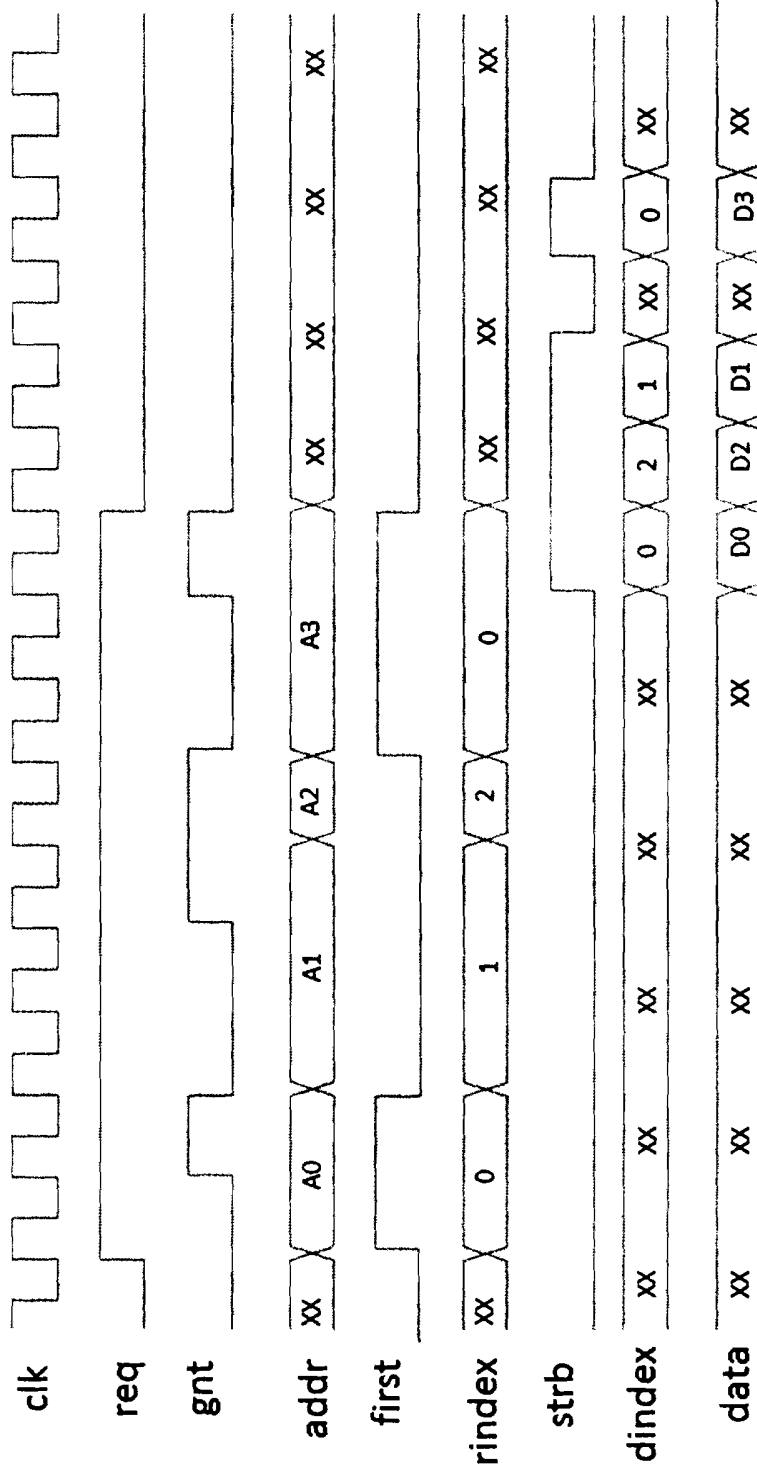
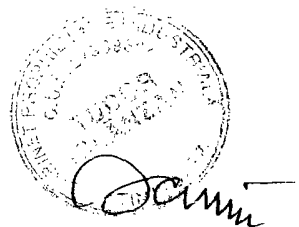


FIG. 23



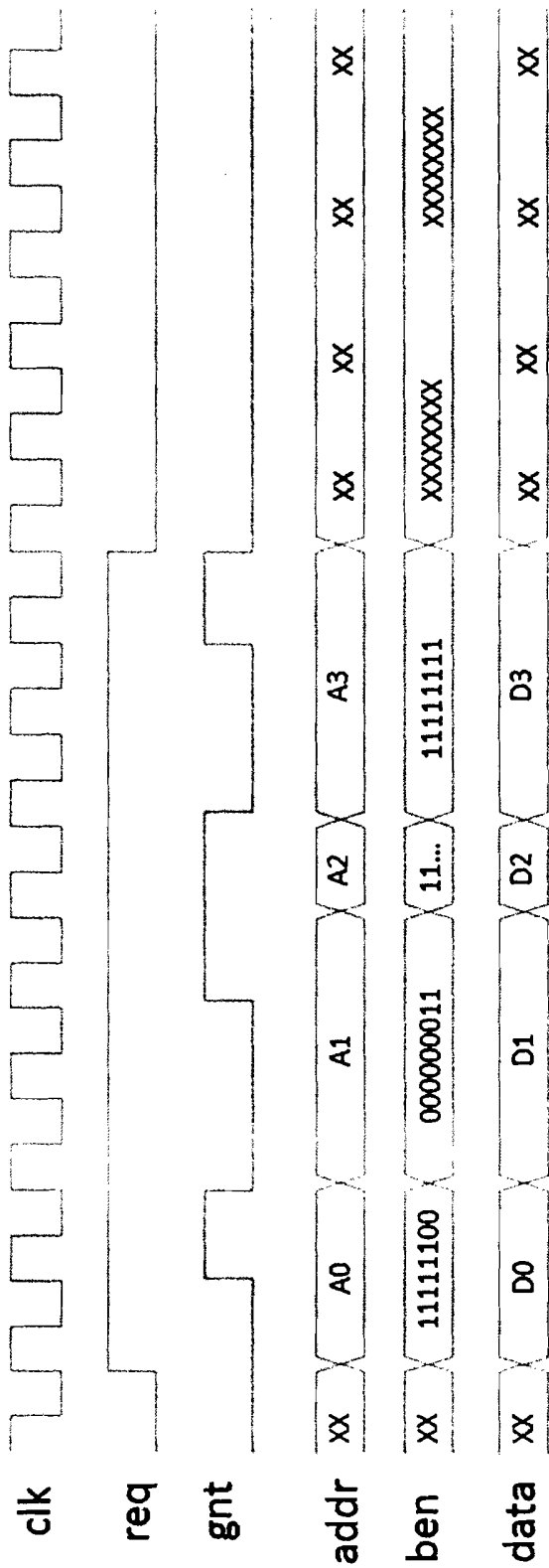
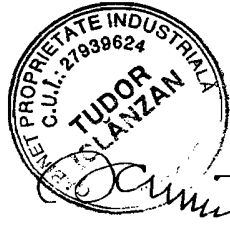


FIG. 24



1202

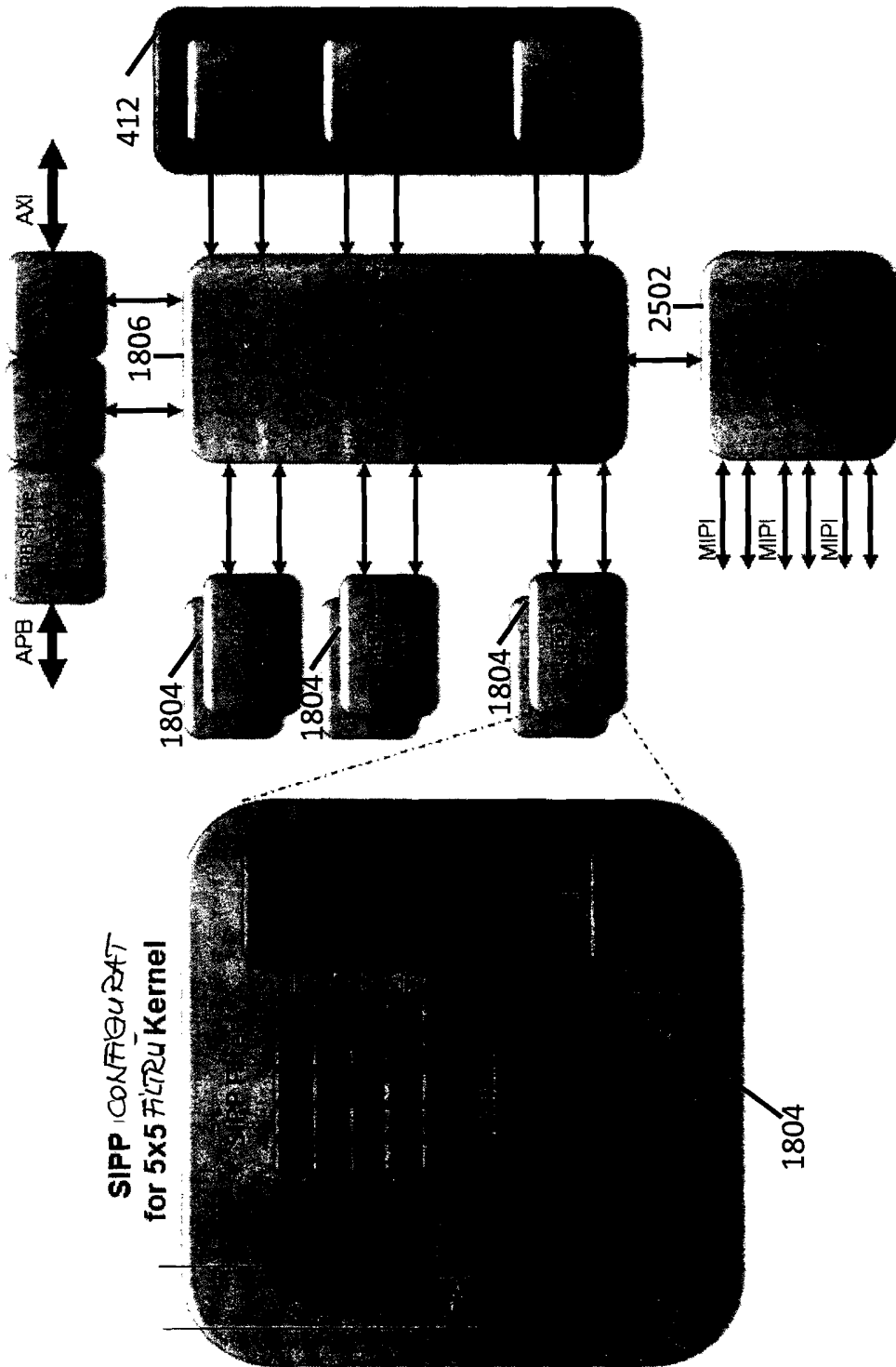
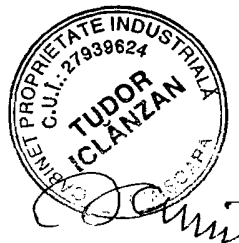


FIG. 25



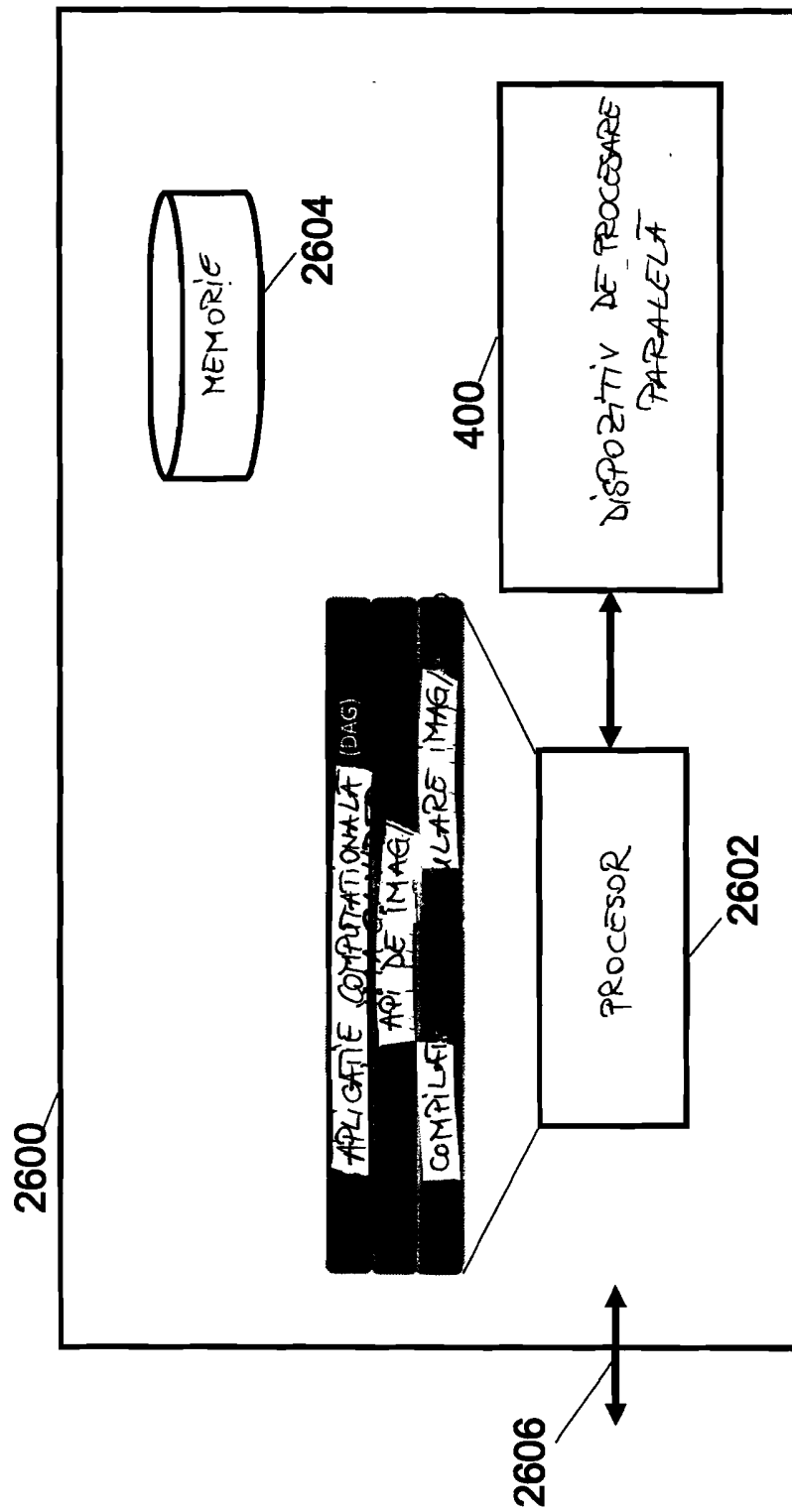


FIG. 26

