DLX

**Generalitati:**

* DLX-ul este un microprocesor didactic care a fost conceput pe baza unor teste executate pe o serie de microprocesoare comerciale cu o filosofie asemanatoare cu cea a DLX-ului.(de ex. MIPS)
* Procesor RISC, 32 de registrii de uz general .Set relativ redus de instructiuni simple, majoritatea fara referire la memorie si cu putine moduri de adresare.( arhitectura tip LOAD / STORE)
* Simulatorul DLX descrie modul de functionare al procesorului DLX.
* Utilizarea tehnicilor de procesare pipeline a instructiunilor, ceea ce implica o rata teoretica de executie de o instructiune / ciclu, pe modelele de procesoare care pot lansa în executie la un moment dat o singura instructiune (procesoare scalare).
* Procesorul DLX are cinci nivele distincte în procesarea instructiunilor :IF, ID,EX,MEM,WB.
* Tipuri de date: Datele sunt pe 8 biti, 16 biti (semicuvânt) si pe 32 de biti (cuvânt)pentru operanzi întregi si 32 de biti simpla precizie si 64 de biti dubla precizie pentru operanzi în virgula mobila.

**Registrii micoprocesorului DLX:**

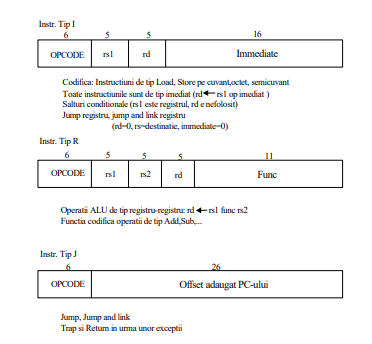
* Procesor RISC, 32 de registrii de uz general (R0,R1...R31), aditional mai exista si un set de registri de virgula flotanta (FPR) care pot fi folositi registri pe 32 de biti. Registrii pentru virgula flotanta pe 64 de biti sunt denumiti F0, F1… F30.
* Valoarea lui R0 este întotdeauna 0 (cablat la masa – caracteristica de altfel comuna tuturor microprocesoarelor RISC)

**Moduri de adresare:**

* imediat si respectiv indexat, ambele cu deplasamentul pe 16 biti.

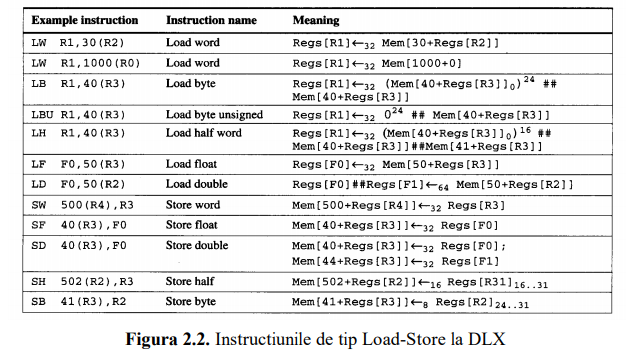
**Formatul instructiunii la DLX:**

* Formatul instructiunii este proiectat în asa fel încât sa asigure o decodificare optima si o functionare rapida a structurii pipeline. Setul de instructiuni este ortogonal pe 32 de biti cu un OPCODE primar de 6 biti. Acest format permite un deplasament de 16 biti folosit ca index, constanta imediata sau adresa relativa (la PC ) de salt.

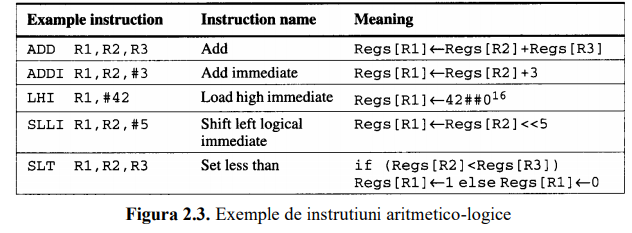


**Instructiuni DLX:**

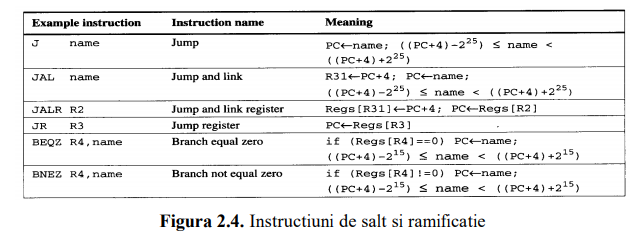
**a) Instructiuni Load-Store:**

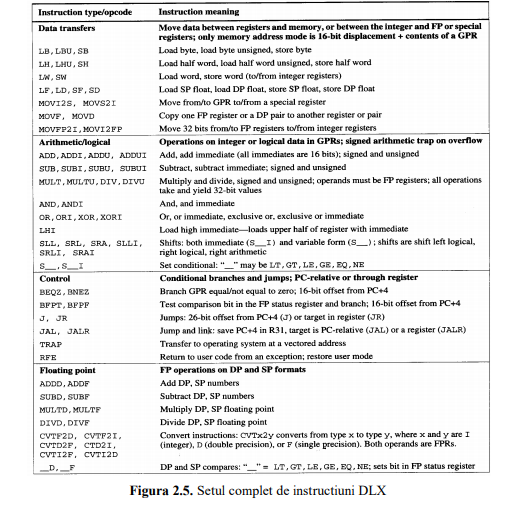


**b)Instructiuni ALU:**



**c)Instructiuni de salt si Ramificatie:**





**Implementare DLX/Implementarea Pipline:**

* Dlx se poate implementa in cel mult 5 cicli:

**1. Instruction Fetch(IF)** -> se calculeaza adresa grupului de instructiuni ce trebuiesc citite din memoria principala. Obtine PC-ul si aduce instructiunea respectiva din memorie înregistrul IR; incrementeaza PC-ul cu 4

**2. Instruction Decode(ID)** ->decodifica instructiunile aduse,se citesc operanzii din setul de registri generali, se calculeaza adresa de salt (pentru instructiunile de ramificatie).

**3.Execution(EX)** -> se executa operatii aritmetico-logice, de deplasare si rotire asupra operanzilor care pot fi numere întregi sau flotante, se calculeaza adresa de acces la memorie (pentru instructiunile LOAD sau STORE).

**4.Memory Acces(MEM)** -> se acceseaza / scrierea datelor din /in memoria principala, prin instructiunile LOAD sau STORE.

**5.Write Back(WB)** -> Scrie rezultatul în setul de registri fie ca acesta provine din memorie sau din iesirile ALU.

***Implementarea Pipline:***

