# Simularea si optimizarea arhitecturilor de calcul

**Autori:**

Miron Horia Andrei

**Semigrupa:**

244/1

**Coordonatori stiintifici:**

Florea Adrian

Stoisor Melisa Cristina

2022 - 2023

# Teorie Pentru proiect

Principalii parametri ai arhitecturii sunt:

**FR (rata de fetch):**

- specifica numarul de instructiuni citite simultan din cache sau memorie intr-un cilu de tact.
- Poate lua valori de 4, 8 sau 16 instructiuni.

**Irmax (issue rate maxim):**

- Numarul maxim de instructiuni, lansate in executie simultan intr-un ciclu de executie.
- Poate lua valorile: 2, 4, 8, 16 instructiuni.

**IBS(Instruction Buffer Size):**

- Dimensiunea buffer-ului este masurata in numar de instructiuni.
- Plaja de valori: 4, 8, 16, 32.
- Buffer-ul de prefetch lucreaza dupa principiul FIFO(first in first out).
- O intrare in buffer contine campurile:
  OPCODE – codul operatiei executata de instructiunea respectiva
  PC_crt- adresa instructiunii curente
  DATE/INSTR- adresa la care se citesc sau se scriu date din sau in memorie.

**Latenta:**

- Reprezinta numarul de ciclii necesari executiei intrsuctiunilor aritmetice, de salt si cele cu referire la memorie. Deobicei(initial) are valoarea 1.

**Memoria Cache ( IC si DC )**

- Sunt cache-uri mapate direct. Datele vor fi memorate in acelasi loc de fiecare data cand sunt accesate => stim ce data va fi evacuata din cache.
- Incarcarea si evacuarea datelor in cache se face la nivel de bloc si nu la nivel de locatie.

**BLOC_SIZE:**

- Dimensiunea in locatii a blocului din cache-ul de date al intrsuctiunii.

**SIZE_IC & SIZE_DC:**

- Dimensiunea cache-urilor de instructiuni respectiv de date au plaja de valori de la 64 locatii(128, 256, ..) pana la peste 8000 de locatii.

Tipuri arhitecturale utilizate in proiectare:

1. **Mapare directa:** fiecare bloc are doar un loc unde poate aparea in cache

   (Adresa blocului) % (Nr bloc. In cache)

2. **Complet asociativa:** blocul poate fi plasat oriunde in Cache.
3. **Semi-ascociativa:** blocul poate fi plasat numai intr-un set predeterminat, dar oriunde in acest set. Un set este un grup de blocuri intr-un Cache.

   (Adresa blocului) % (Nr. de seturi in Cache)

**Write-Through:**

- Informatia e scrisa in cache & Memoria principala
- Nu implica evacuare – AVANTAJ
- Exista penalitati datorita scrierii in MP – DEZAVANTAJ

**Write-Back:**

- Informatia e scrisa doar in Cache
- Viteza de scriere este cea a memoriei cache – AVANTAJ
- Implica evacuare in MP – DEZAVANTAJ

**Data Write Buffer (DWB):**

Mod de lucru:

1. Preia toate instructiunile LD, ST;
2. Le serializeaza
3. Anunta CPU-ul ca acesta nu mai are treaba cu aceste instructiuni LD, ST
4. DWB se va ocupa de bypassing, scrierea, citirea din Cache.

```
private void buttonChooseFile_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        sSourceFileName = openFileDialog.FileName;
    }
}
```

- Butonul de care ne ajutam sa deschidem file-ul respectiv pentru a putea alege trace-urile ('Trc').

```
public void OneCycleCalculations()
{
    int numberOneCycle = 0;
    int pc = 0;
    bool firstRead = true;
    bool branchFlag = true;
    int ct = 0;

    foreach (trace trc in traces)
    {
        ct++;
        if (trc.traceType == "S") // store
        {
            result.store++;
            result.dataCacheAccesses++;
        }
        else if (trc.traceType == "L") //load
        {
            result.load++;
            result.dataCacheAccesses++;
        }
        else
        {
            result.branch++;
        }

        if (firstRead)
        {
            numberOneCycle += trc.currentAdress - pc;
            firstRead = false;
        }
        else
        {
            if ((trc.currentAdress != pc) && (branchFlag == true))
            {
                numberOneCycle += trc.currentAdress - pc;
            }
            else if (branchFlag == true)
            {
                numberOneCycle += trc.currentAdress - pc;
            }
            else
            {
                numberOneCycle += trc.currentAdress - pc - 1;
            }
        }
```

- Calcula diferenta intre salturi, adica diferentele intre adresele curente si cele finale pentru fiecare trace

```
private void comboBoxFetchRate_SelectedIndexChanged(object sender, EventArgs e)
{
    fetchRate = Int32.Parse(comboBoxFetchRate.SelectedItem.ToString());
}

private void comboBoxIRMax_SelectedIndexChanged(object sender, EventArgs e)
{
    issueRMax = Int32.Parse(comboBoxIRMax.SelectedItem.ToString());
}

private void comboBoxIBS_SelectedIndexChanged(object sender, EventArgs e)
{
    instructionBS = Int32.Parse(comboBoxIBS.SelectedItem.ToString());
}

private void numericUpDownLatency_ValueChanged(object sender, EventArgs e)
{
    latency = (int)numericUpDownLatency.Value;
}

private void comboBoxNPen_SelectedIndexChanged(object sender, EventArgs e)
{
    nPen = Int32.Parse(comboBoxNPen.SelectedItem.ToString());
}

private void comboBoxICblockSize_SelectedIndexChanged(object sender, EventArgs e)
{
    blockSizeICache = Int32.Parse(comboBoxICblockSize.SelectedItem.ToString());
}

private void comboBoxSizeIC_SelectedIndexChanged(object sender, EventArgs e)
{
    sizeICache = Int32.Parse(comboBoxSizeIC.SelectedItem.ToString());
}
```

- Aceste functii sunt folosite pentru a retine in memeorie valorea pe care le-o insusim cand rulam programul.