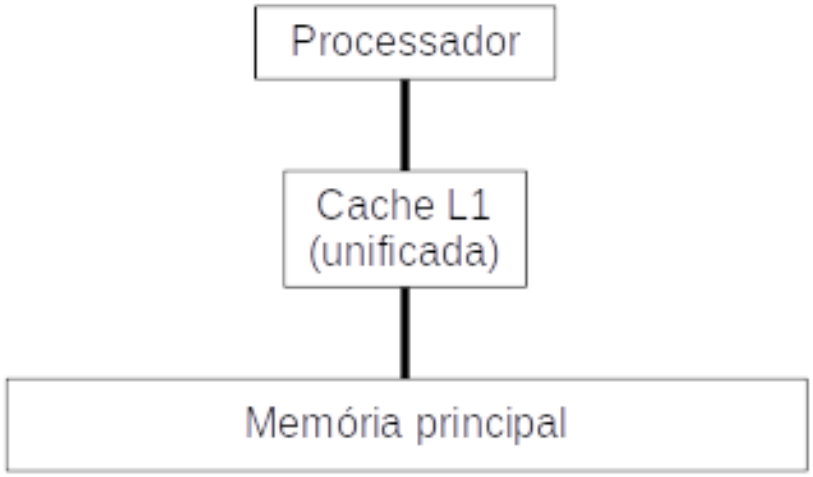
1o. Trabalho: Simulação e Avaliação de Caches

- Parte I: **Desenvolvimento de simuladores de diferentes organizações de caches**

- Parte II: **Avaliação de desempenho das diferentes organizações**
  - Realizar experimentos com os simuladores
  - Obter resultados de desempenho
  - Analisar resultados
  - Responder questionário

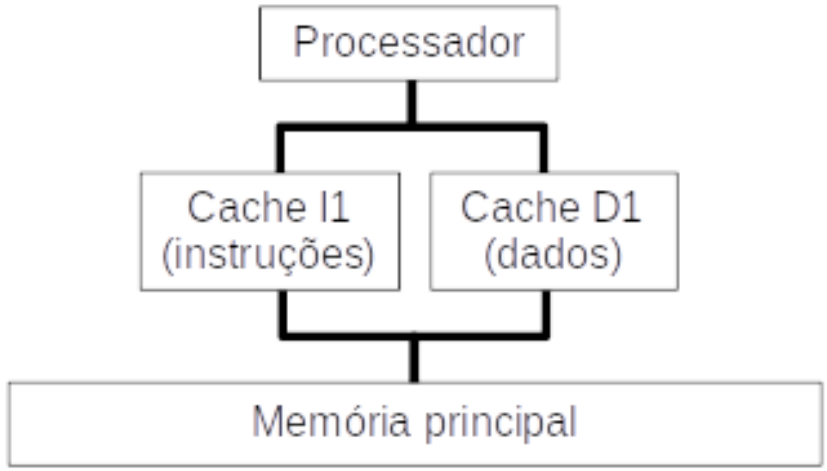
Parte I: Desenvolvimento de Simuladores de Caches

- **Módulo cache**: Implementa cache genérica
  - Fornecido pronto em C
  - Entender o código
- **Simulador simbasica**:
  - Um único nível de cache, unificada para instruções e dados: **cache L1**
  - Fornecido pronto em C
  - Entender o código

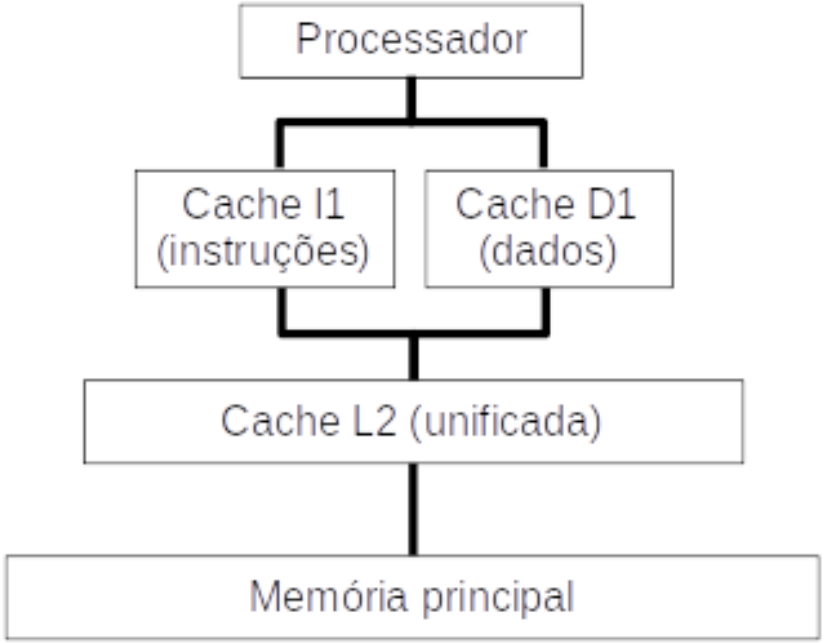


Parte I: Desenvolvimento de Simuladores de Caches

- **Simulador simsplit**:
  - Um único nível de cache split, separada para instruções e dados: **caches I1 e D1**
  - Implementar, baseando-se no simbasica

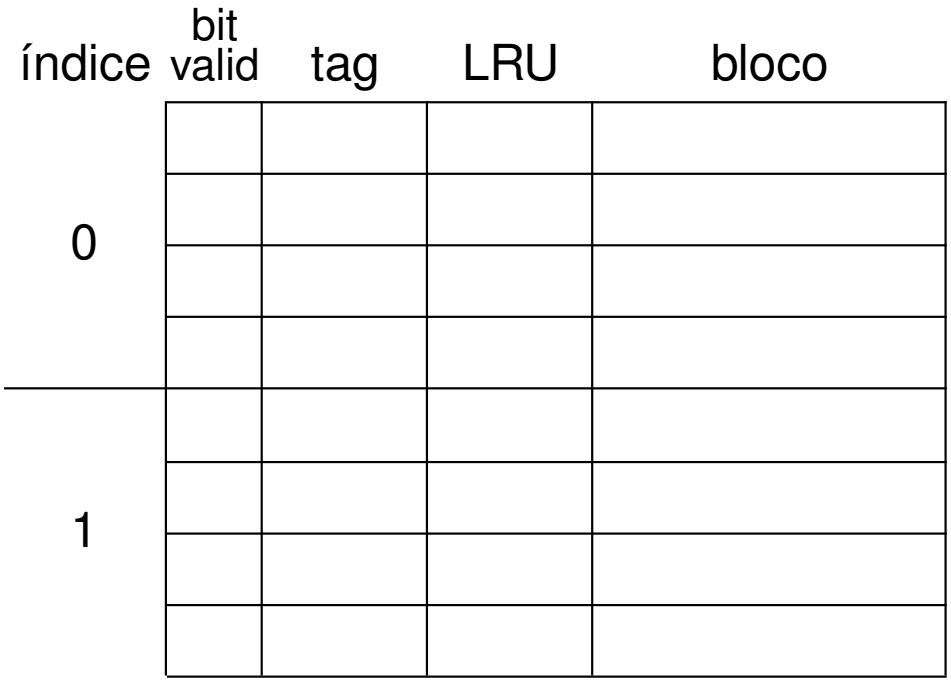


- **Simulador simniveis**:
  - Dois níveis de cache, split no nível 1 e unificada no nível 2: **caches I1, D1 e L2**
  - Implementar, baseando-se no simsplit



Módulo cache: Implementa Cache Genérica (fornecido pronto)

- **Configuração da cache**: fornecida como entrada
  - Nº total de blocos, associatividade, nº de palavras por bloco
- **Estrutura de dados da cache**:
  - **Cache**: vetor de conjuntos (nº de conjuntos = nº total de blocos / associatividade)
  - **Cada conjunto**: vetor de vias (nº de vias = associatividade)
  - **Cada bloco**: bit de validade, tag, LRU (não precisa de campo bloco)
- **Exemplo**:



nº total de blocos = 8

associatividade = 4

nº de palavras por bloco =

Módulo cache: Implementa Cache Genérica (fornecido pronto)

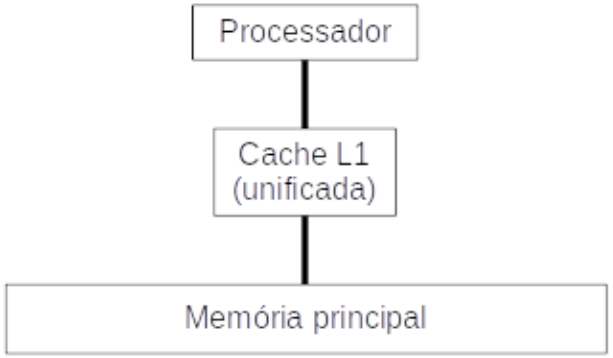
- **Função alocaCache (..., nBlocos, associatividade, nPalavrasBloco)**:
  - Recebe como entrada **configuração da cache**
  - Configuração determina:
    - Nº de conjuntos da cache
    - Se cache é mapeada diretamente, associativa por conjunto ou totalmente associativa
    - Se cache explora localidade espacial ou não
  - Aloca estrutura de dados para cache
  - Inicializa cache vazia: bit de validade = 0, para todos as posições
- **Função liberaCache (...)**:
  - Libera estrutura de dados alocada para cache

Módulo cache: Implementa Cache Genérica (fornecido pronto)

- **Função buscaInsereCache (..., endereco)**:
  - Recebe como entrada: **endereço de memória acessado**
  - Divide os bits do endereço em campos: tag, índice (se houver)
  - Procura dado na cache, usando índice,
    testando bits de validade e comparando tags
  - Obtém acerto ou falha no acesso a cache
  - Caso ocorra falha:
    - "Insere bloco" acessado na cache
    - Se necessário, realiza substituição na cache ("retira um bloco" da cache), usando política de substituição **LRU** (**exato**), se necessário
  - **Retorna**:
    - 0: acesso causou acerto
    - 1: acesso causou falha (SEM substituição)
    - 2: acesso causou falha (COM substituição)

Simulador simbasica (fornecido pronto)

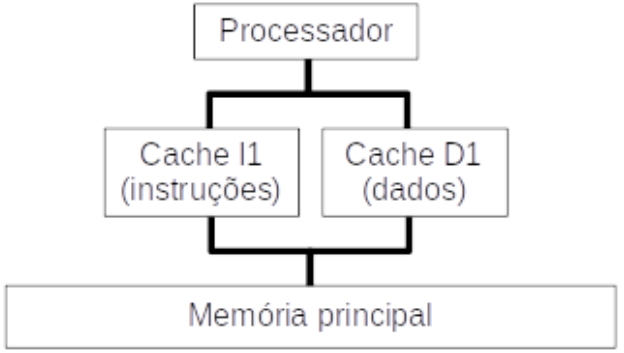
- **Um único nível de cache, unificada para instruções e dados: cache L1**
  - **Inicializa**:
    - Recebe por linha de comando:
      - Arquivo com configuração da cache **L1**
      - Arquivo de trace
    - Inicializa medidas de desempenho
    - Aloca cache **L1** com configuração lida, chamando função **alocaCache**



  - **Laço principal**:
    - Enquanto não processou todos os endereços do trace:
      - Lê um acesso do trace (com endereço de memória acessado)
      - Simula acesso a esse endereço na cache **L1**, chamando função **buscaInsereCache** para cache **L1**
      - Atualiza medidas de desempenho
  - **Finaliza**:
    - Imprime medidas de desempenho
    - Libera cache **L1**, chamando função **liberaCache**

Simulador simsplit (implementar)

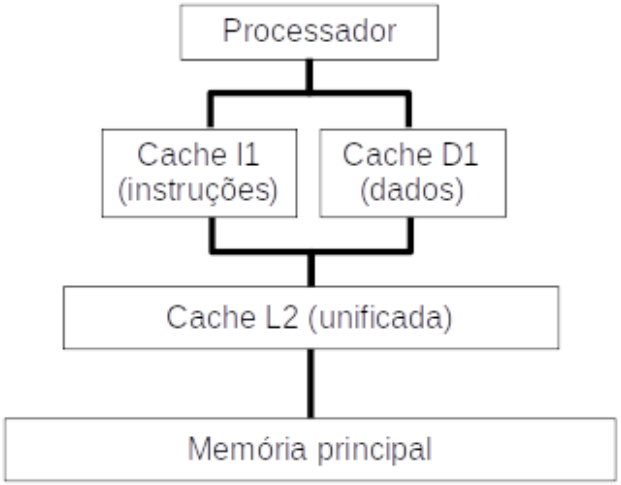
- **Um único nível de cache split, separada para instruções e dados: caches I1 e D1**
  - **Inicializa**:
    - Recebe por linha de comando:
      - Arquivo com configuração das caches **I1** e **D1**
      - Arquivo de trace
    - Inicializa medidas de desempenho
    - Aloca **I1** e **D1**, chamando função **alocaCache** para cada uma



  - **Laço principal**:
    - Enquanto não processou todos os endereços do trace:
      - Lê um acesso do trace
      - Se acesso é a instrução:
        * Simula acesso a esse endereço na cache **I1**, chamando função **buscaInsereCache** para cache **I1**
      - Senão (se acesso é a dado):
        * Simula acesso a esse endereço na cache **D1**, chamando função **buscaInsereCache** para cache **D1**
      - Atualiza medidas de desempenho
  - **Finaliza**:
    - Imprime medidas de desempenho
    - Libera caches **I1** e **D1**, chamando função **liberaCache** para cada uma

Simulador simniveis (implementar)

- **Dois níveis de cache, split no nível 1 e unificada no nível 2: caches I1, D1 e L2**
  - **Inicializa**:
    - Recebe por linha de comando:
      - Arquivo com configuração de **I1**, **D1** e **L2** e arquivo de trace
    - Inicializa medidas de desempenho
    - Aloca **I1**, **D1** e **L2** com configuração lida, chamando função **alocaCache** para cada
  - **Laço principal**:
    - Enquanto não processou todos os endereços do trace:
      - Lê um acesso do trace
      - Se acesso é a instrução:
        * Simula acesso a esse endereço na cache **I1**,
        chamando função **buscaInsereCache** para cache **I1**
      - Senão (se acesso é a dado):
        * Simula acesso a esse endereço na cache **D1**,
        chamando função **buscaInsereCache** para cache **D1**
      - Se ocorreu falha em **I1** ou **D1**:
        * Simula acesso a esse endereço na **L2**,
        * chamando função **buscaInsereCache** para **L2**
      - Atualiza medidas de desempenho
  - **Finaliza**: Imprime medidas de desempenho e libera **I1**, **D1** e **L2**



Simuladores e Interface de Execução

- **Simuladores simsplit, simniveis e simpre**:
  - **Desenvolver**: em C, c++, java ou Python
- **Interface de execução dos simuladores**:
  - **Entradas**:
    - Arquivo com configuração da(s) cache(s)
    - Arquivo de trace
  - **Saídas**: Medidas de desempenho impressas na tela
  - **Execução por linha de comando**:
    - `simbasica arquivoConfiguracao arquivoTrace`
  - **Não modificar interface de execução dos simuladores**

Entrada dos Simuladores: Arquivo de Configuração de Cache

- **Arquivo texto**:
  - Possui informações da configuração das caches, **nesta ordem**:
    - Nº total de blocos da cache
    - Associatividade da cache (nº de blocos por conjunto)
    - Nº de palavras por bloco da cache
  - **Uma linha para cada cache**
- **Valores da configuração são sempre potência de 2**
- **Exemplos**:
  - **simbasica**:

32	2	2

  - **simsplit**:

32	2	1
32	1	4

  - **simniveis**:

32	2	1
32	1	4
256	4	8

Entrada dos Simuladores: Arquivo de Trace de Endereços

- Cada trace corresponde à execução de um programa, com uma determinada entrada de dados
- **Arquivo texto**:
  - Contém todos os **acessos à memória** realizados pelo programa (acessos a instruções e dados)
  - Possui, **para cada acesso**: (uma linha para cada acesso)
    - Tipo de acesso: 'I', 'L' ou 'S'

      [leitura de instrução, leitura de dado (load) ou escrita de dado (store)]
    - Endereço de memória acessado
- **Exemplo**:

```
I  12288
I  12292
I  12296
I  12320
L  0
I  12324
L  32
I  12328
S  4
```

Entrega do Trabalho

- **Grupos**: 2 ou 3 alunos
- **Data de entrega**:
- **Submissão pelo AVA**