Power-efficient memory and caches

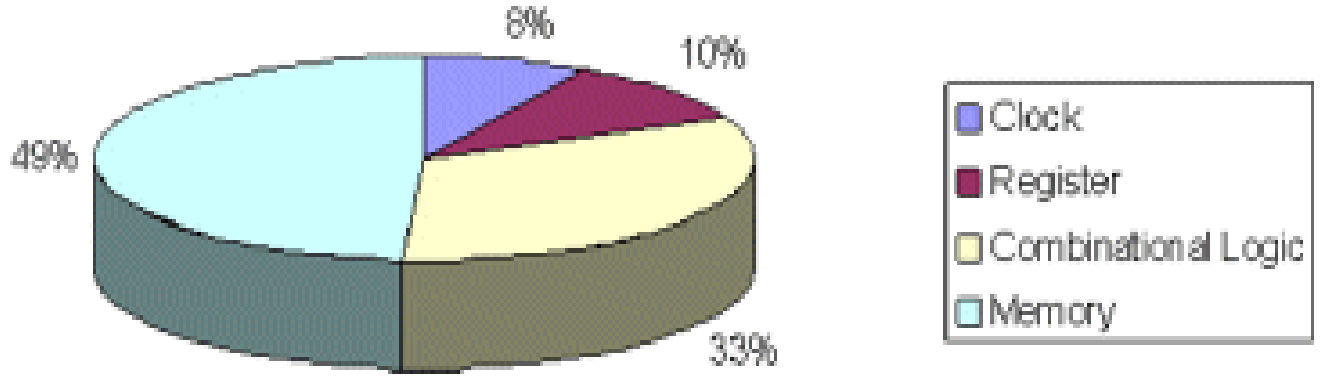
Jothini Sritharan

Green Computing

Introduction

Motivation

- Memory Centric



Power-consumption distribution in a SoC Design (Integrated Circuit)

Content

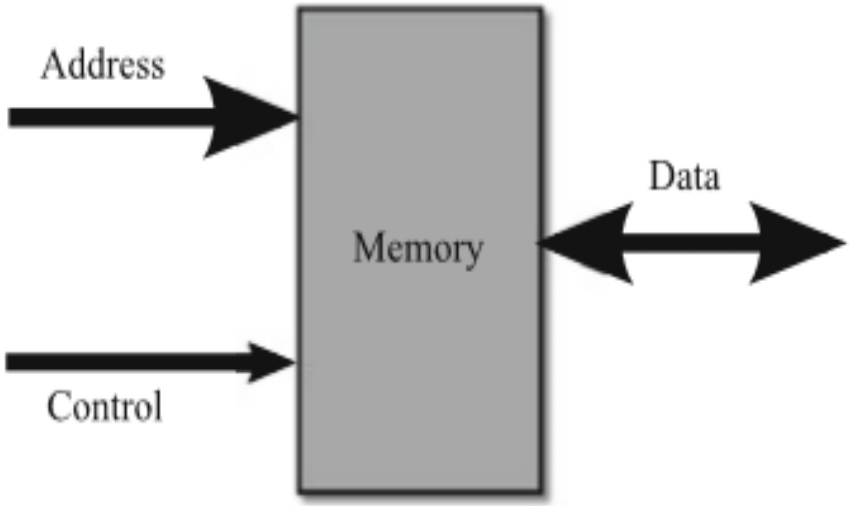
- Revision: Memory organization and caches
- Energy consumption concerning memory access
- Ideas of reducing power consumption
- Low power memory design

1. Memory organization and caches

- **Input:**

  → address: location of memory which is accessed

  → control: indicates important data; signals kind of operation



- **Bidirectional bus:**

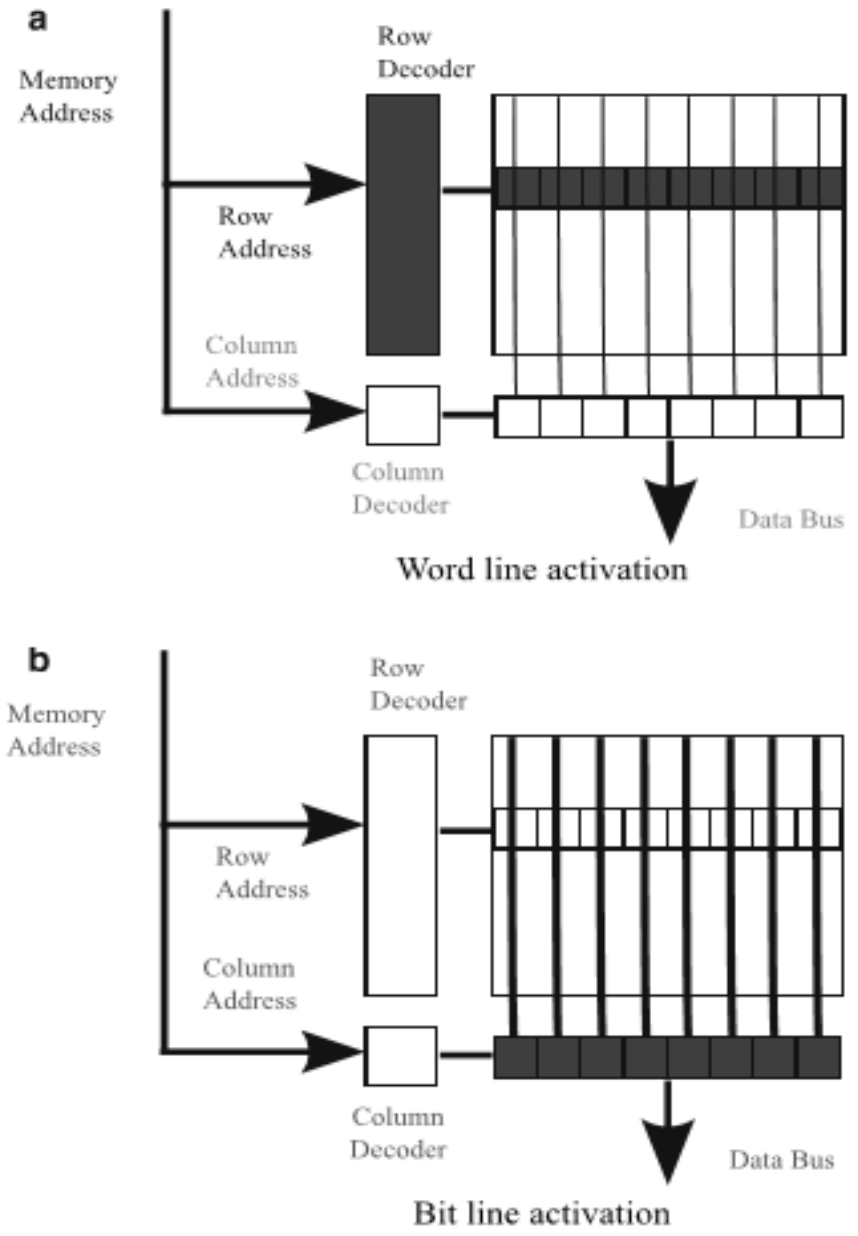
  → data: depending on operation it serves as input or output

- **Read operation:**

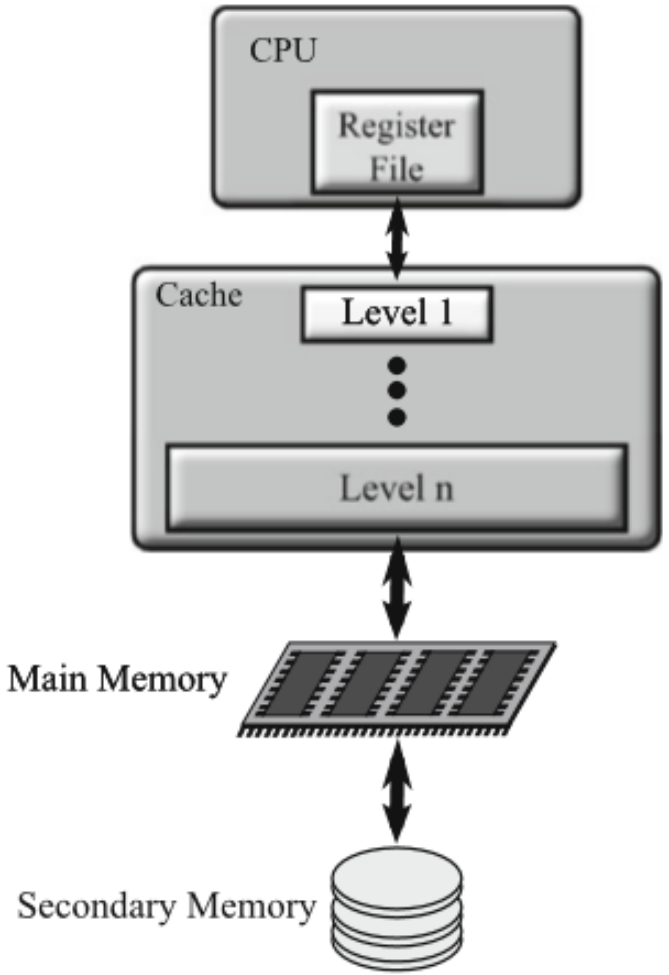
1. row address decoded
   → activates word line

2. selection of a row of cells
   → transfer of data between cells
   and bit lines

3. column decoder chooses bits in right column and transfers them to data bus

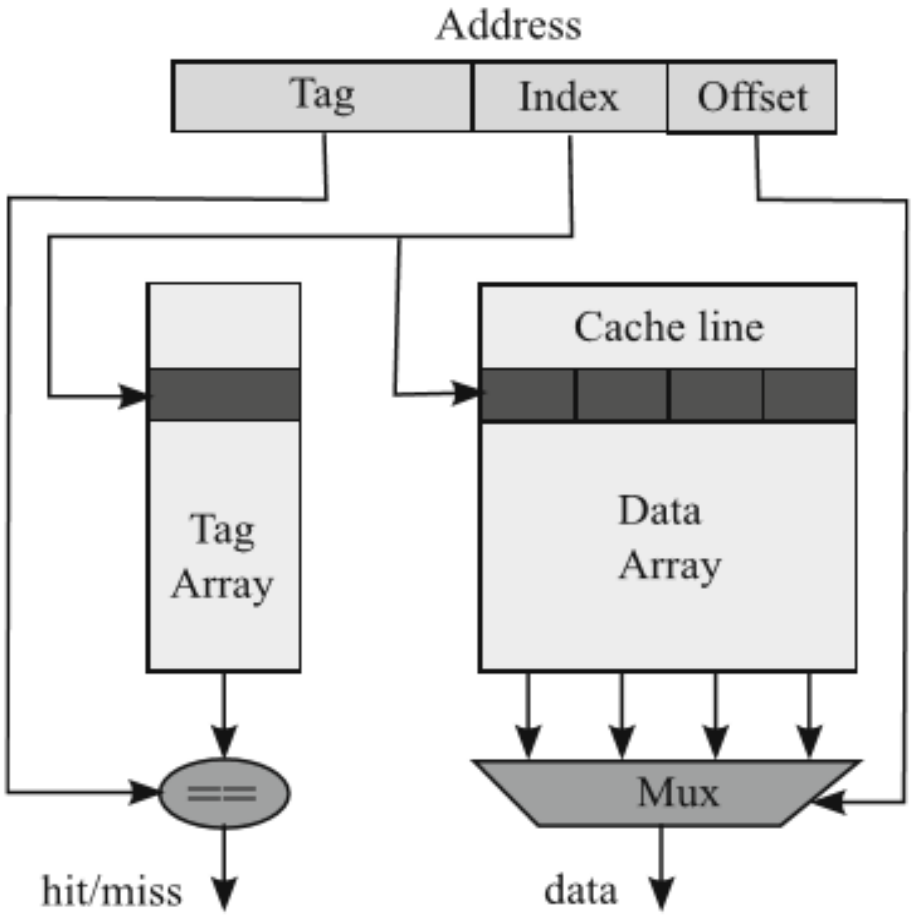


Cache



- „Cache hit“ → succesful finding of data
- „Cache missing“:
  - → compulsory miss
  - → capacity miss
  - → conflict miss
- Cache miss ratio = $\frac{number\ of\ Cache\ misses}{number\ of\ Cache\ accesses}$

Cache architecture



Cache hit:

- tag matches to an element of tag array
- Index collects corresponding tag and cache line
- Offset chooses right data from selected line

2. Energy consumption concerning memory access

The following three main components basically determine the energy consumption:

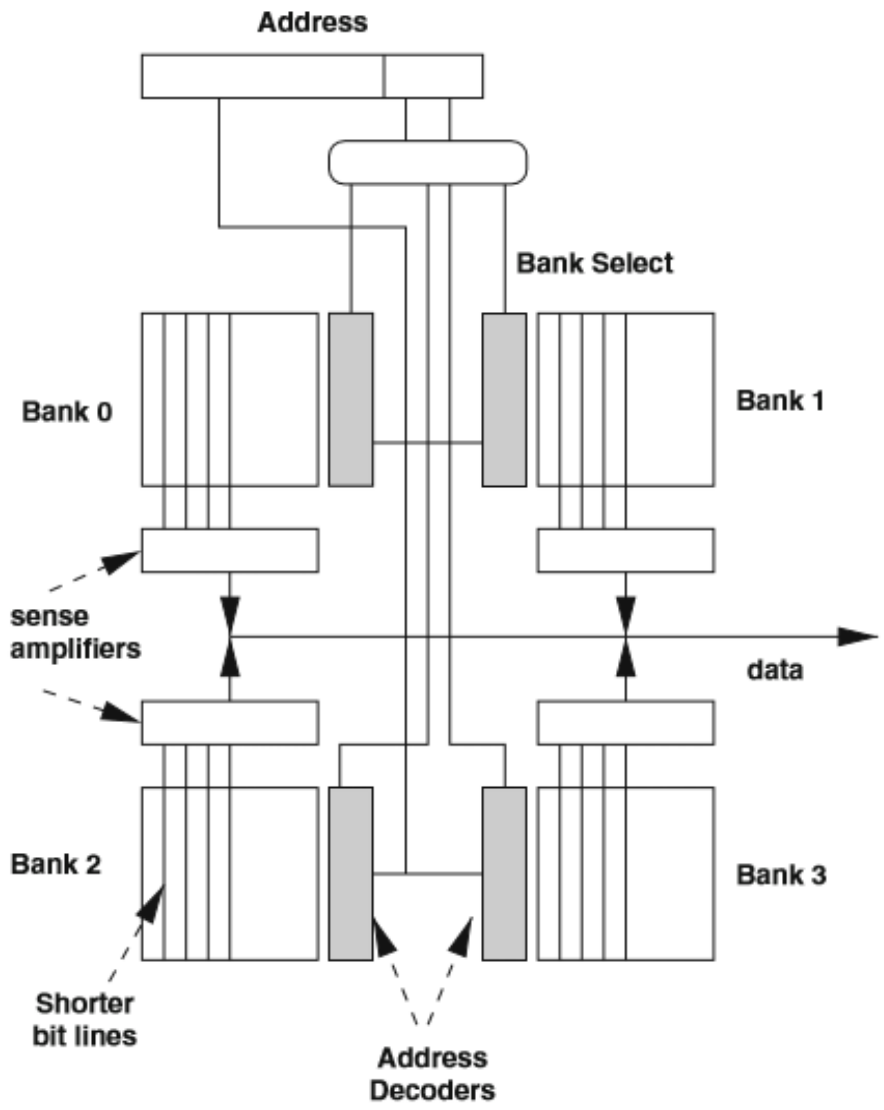
- 1. address decoders and word lines
- 2. data array, sense amplifiers and the bit lines
- 3. the data and address buses leading to the memory

3. Ideas of reducing power consumption

3.1 Power-efficient memory Architectures

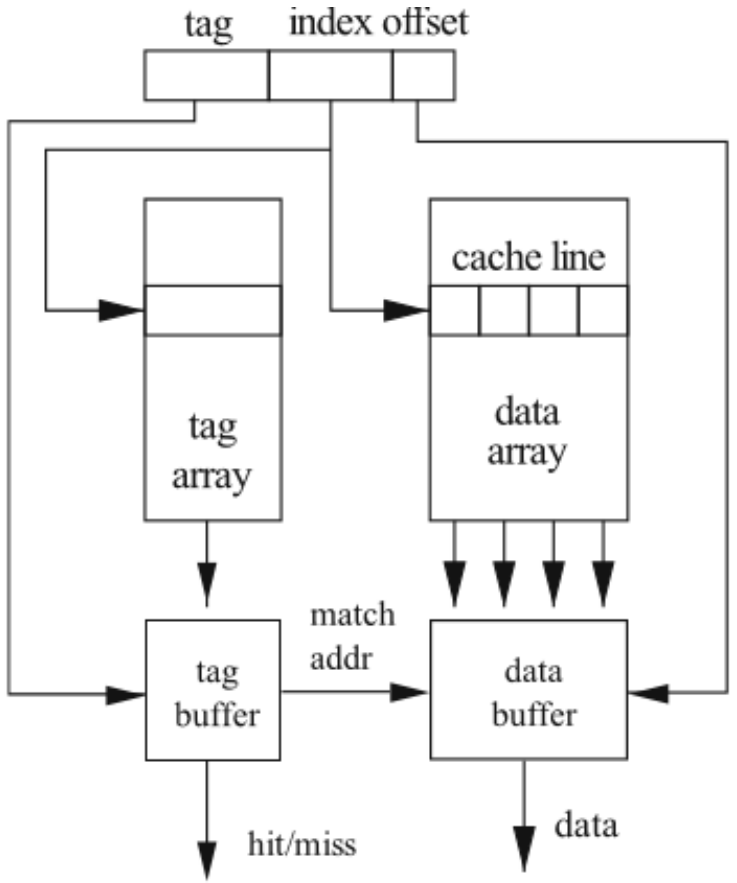
- **Partitioning memory and caches**

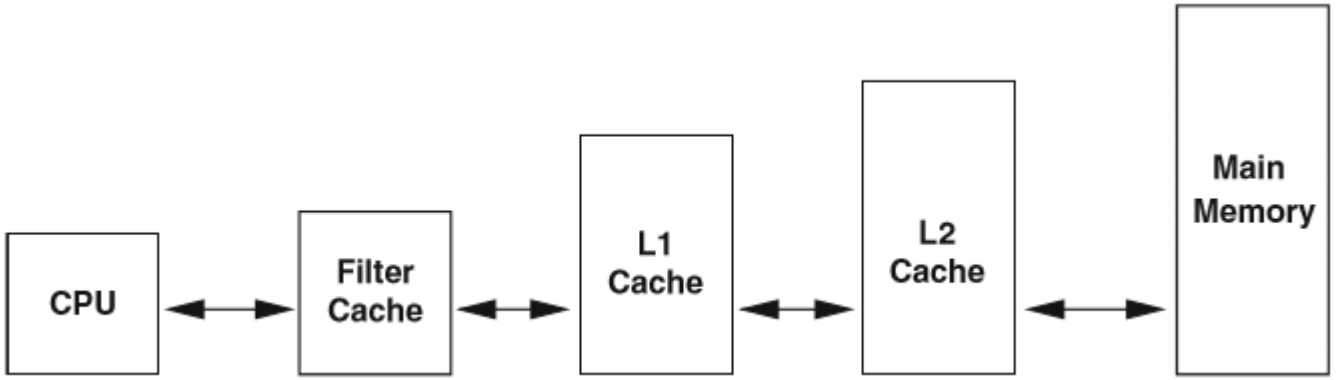
➢ memory array is divided into several banks → shorter transferring path because of smaller bit lines



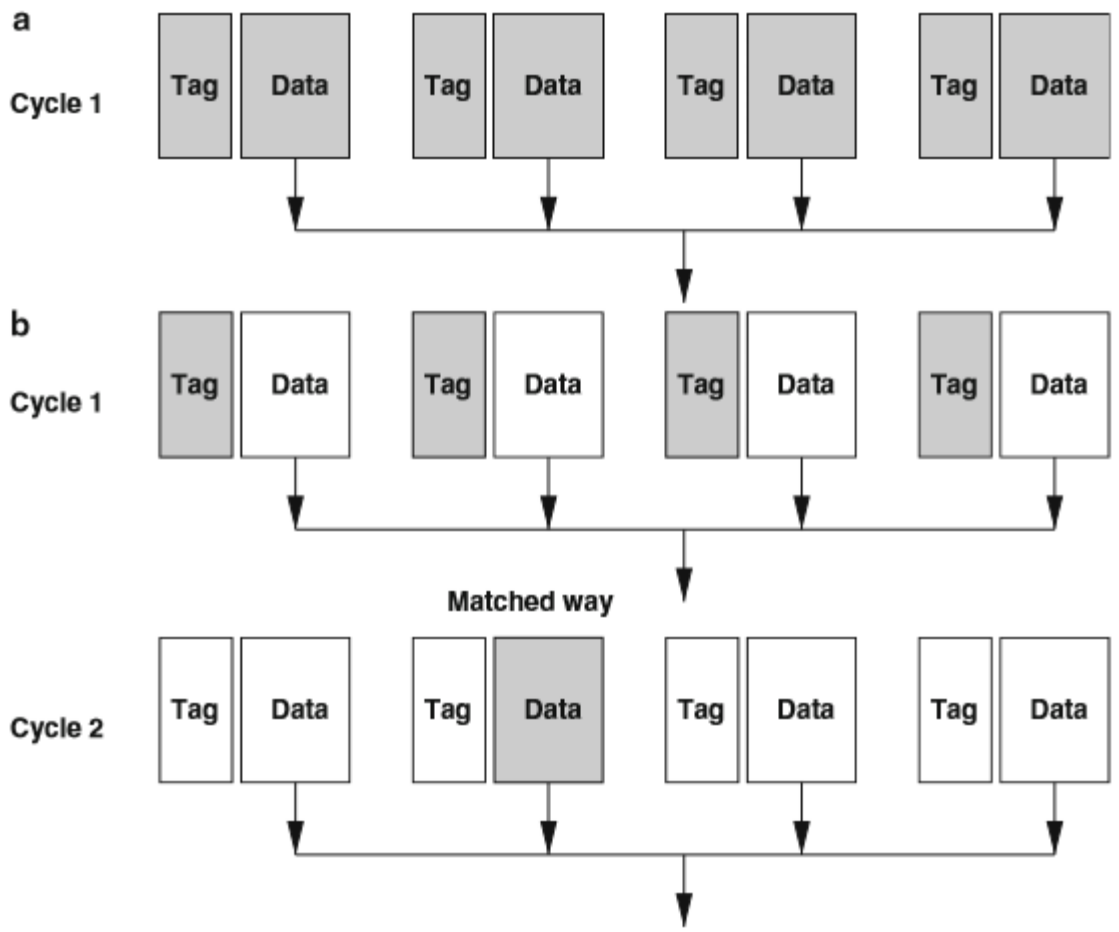
- **Additional memory**

➢ a small extra cache, which stores recently used data, is searched first and may prevent browsing through the whole main cache

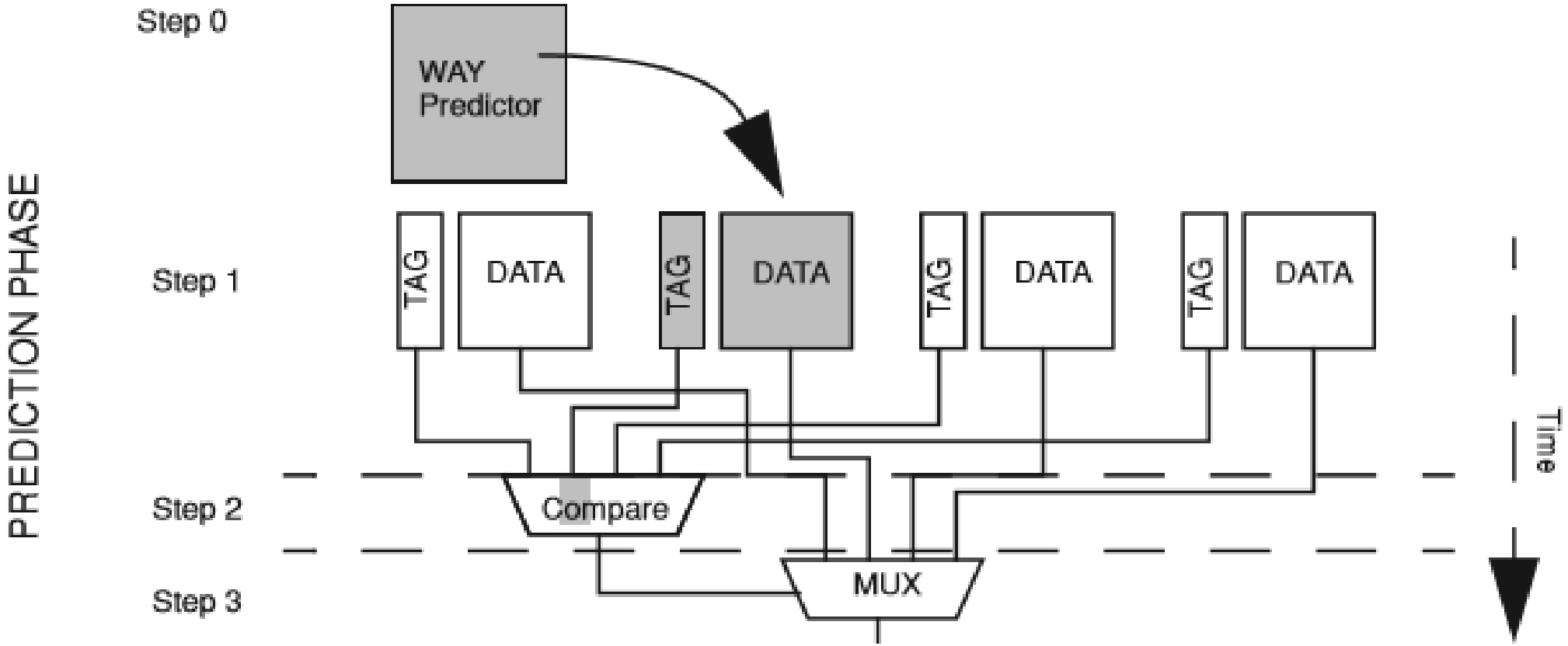


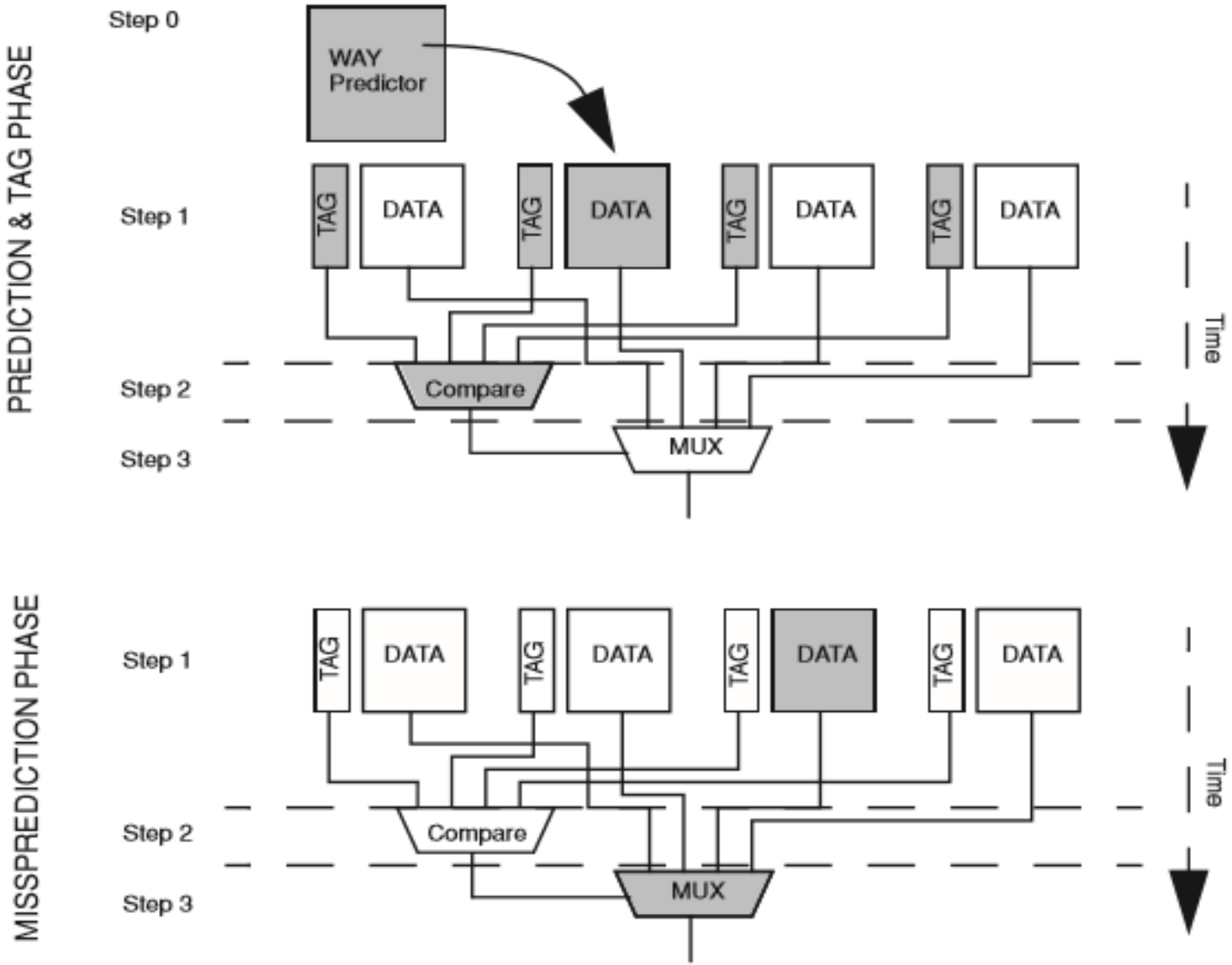


- **Reducing Tag and Data Array Fetches**



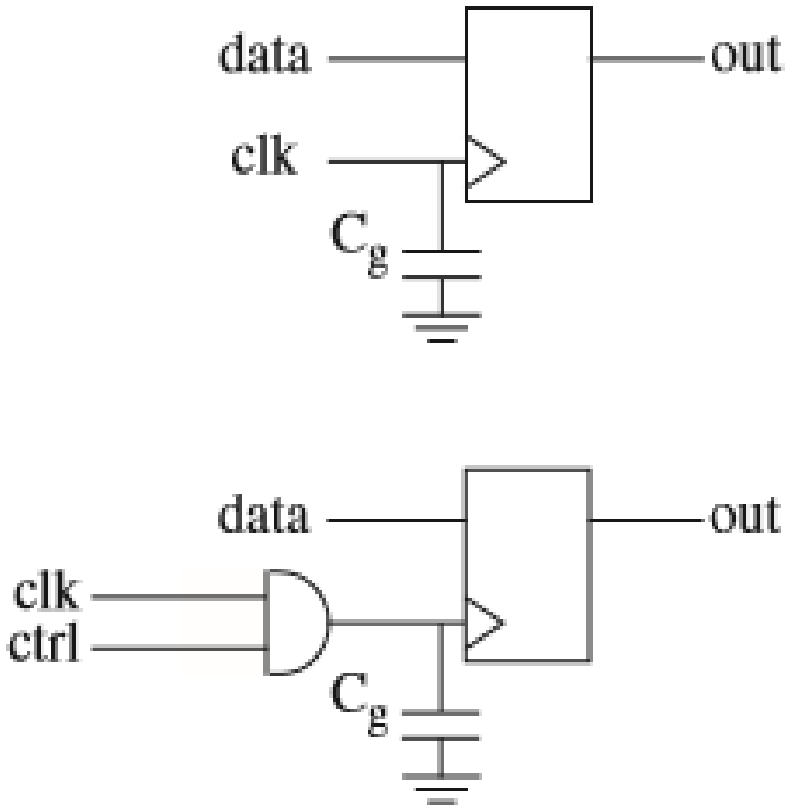
- Reducing number of accesses by only searching the tags without data first
- Then picking only the data of the right tag


PREDICTION PHASE
Step 0
WAY Predictor
Step 1
TAG
DATA
TAG
DATA
TAG
DATA
TAG
DATA
Step 2
Compare
Step 3
MUX
Time


PREDICTION & TAG PHASE
Step 0
WAY Predictor
Step 1
TAG
DATA
TAG
DATA
TAG
DATA
TAG
DATA
Step 2
Compare
Step 3
MUX
Time
MISSPREDICTION PHASE
Step 1
TAG
DATA
TAG
DATA
TAG
DATA
TAG
DATA
Step 2
Compare
Step 3
MUX
Time

3.2 Clock-Gating

- By connecting the clock with a control signal by an AND Gatter, the capacitance is only charged and discharged when control is on TRUE



3.3 Cacheable switching activity

- Recurring instructions are stored in a cache

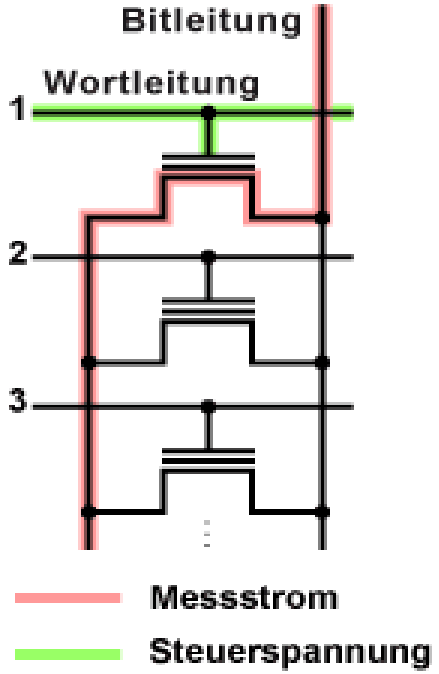
→ calculation has to be done only once

- Example: For loops

→ But only applicable when same input

4 Low power memory design

4.1 **NOR-Flash-Memory**



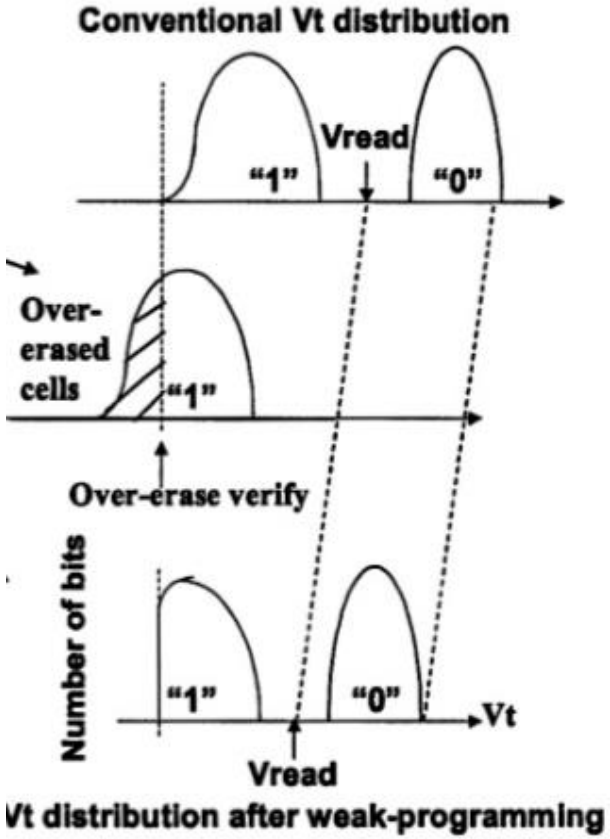
- Fast random access since a transistor can be individually adressed due to the parallel circuit

- **Bit-by-bit-weak program**

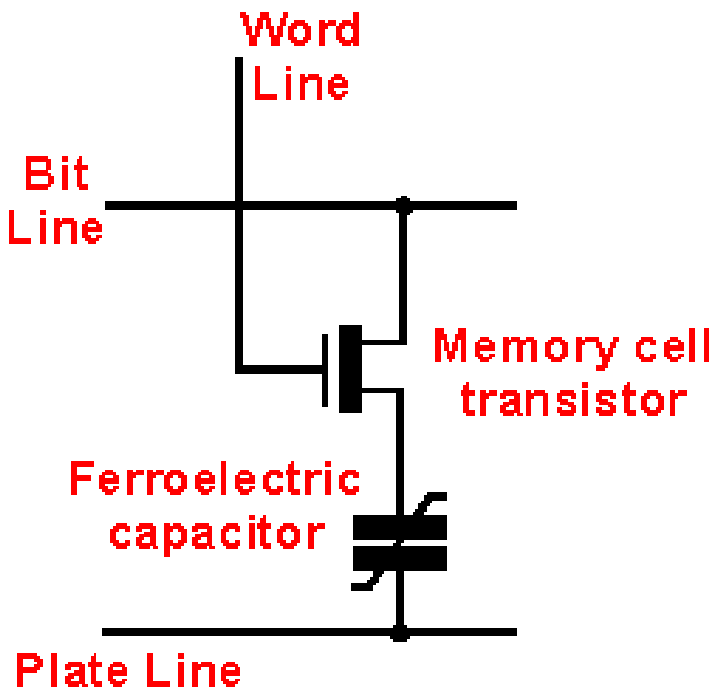
➢Applying lower voltage than ordinary

➢Offset of whole curve

→ Lower voltage is needed

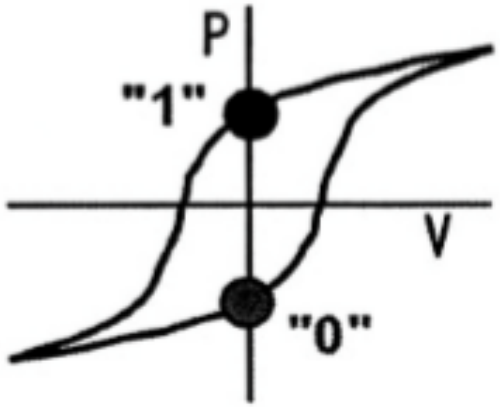


4.2 Ferroelectric Random Access Memory (FeRAM)

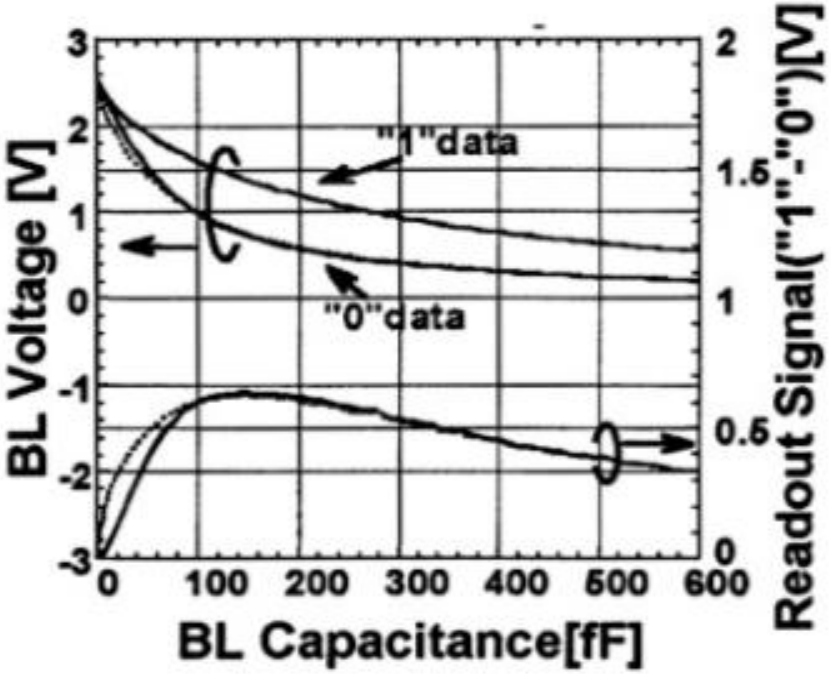


- Selecting transistor
- Capacitor saves information in form of polarization

- Reading: another voltage impulse leads to either a change or a constancy in polarization



- Two stable states at the voltage of 0V



- Readout-signal depends on used capacitance
  → has a maximum

Summary

- Memory and caches are responsible for a large share of power-consumption
  → necessity for power-efficient caches and memories

<u>General solutions:</u>

- Optimization of Cache architecture
- Additional memory with most recently used data
- Adaption of components like transistors and capacitances