ELECTRONICS DEPARTMENT

ROBOTICS TECHNOLOGY CENTER

Introduction To Embedded System

By: Mustafa M. Shiple

Quote of the Day

إذا سألت نفسك في كل أسبوع مرة : ماذا قرأت في هذا الأسبوع , فقد ضمنت لنفسك أعظم مستقبل علمي

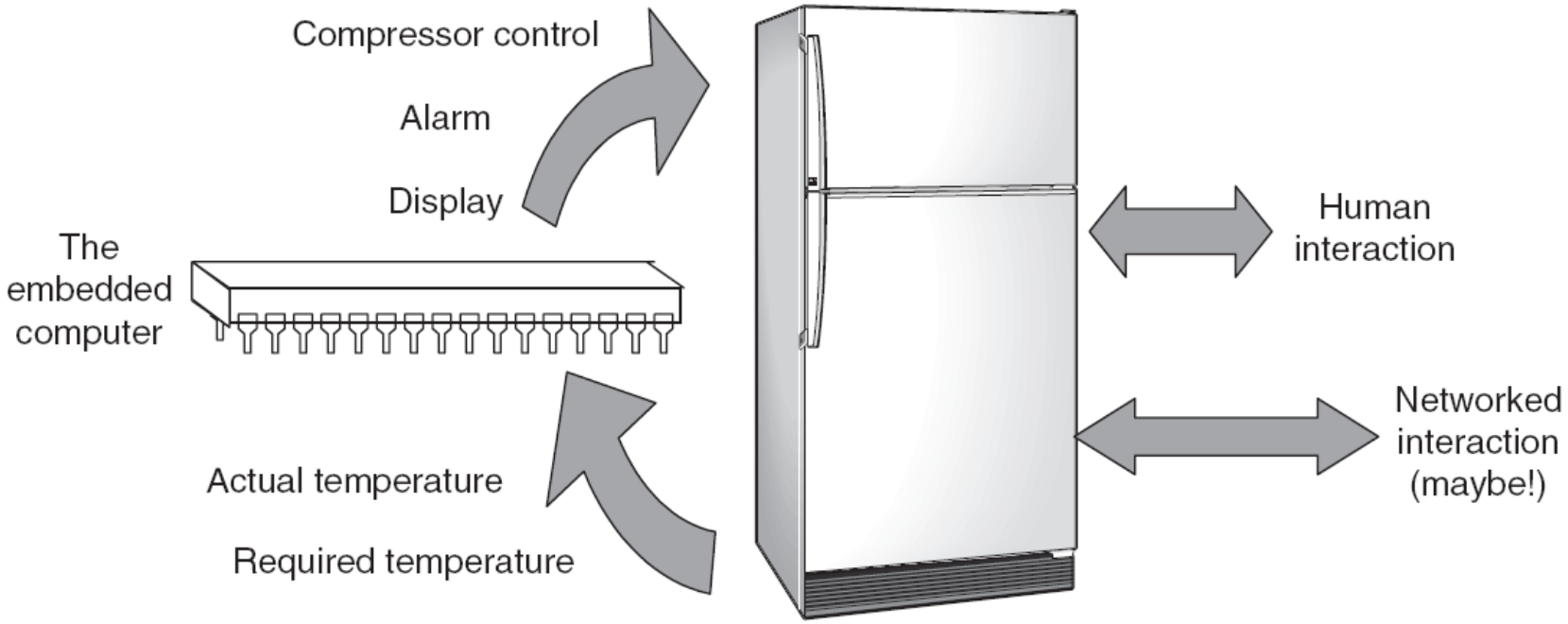
The empires of the future are the empires of the mind.

– *Winston Churchill*

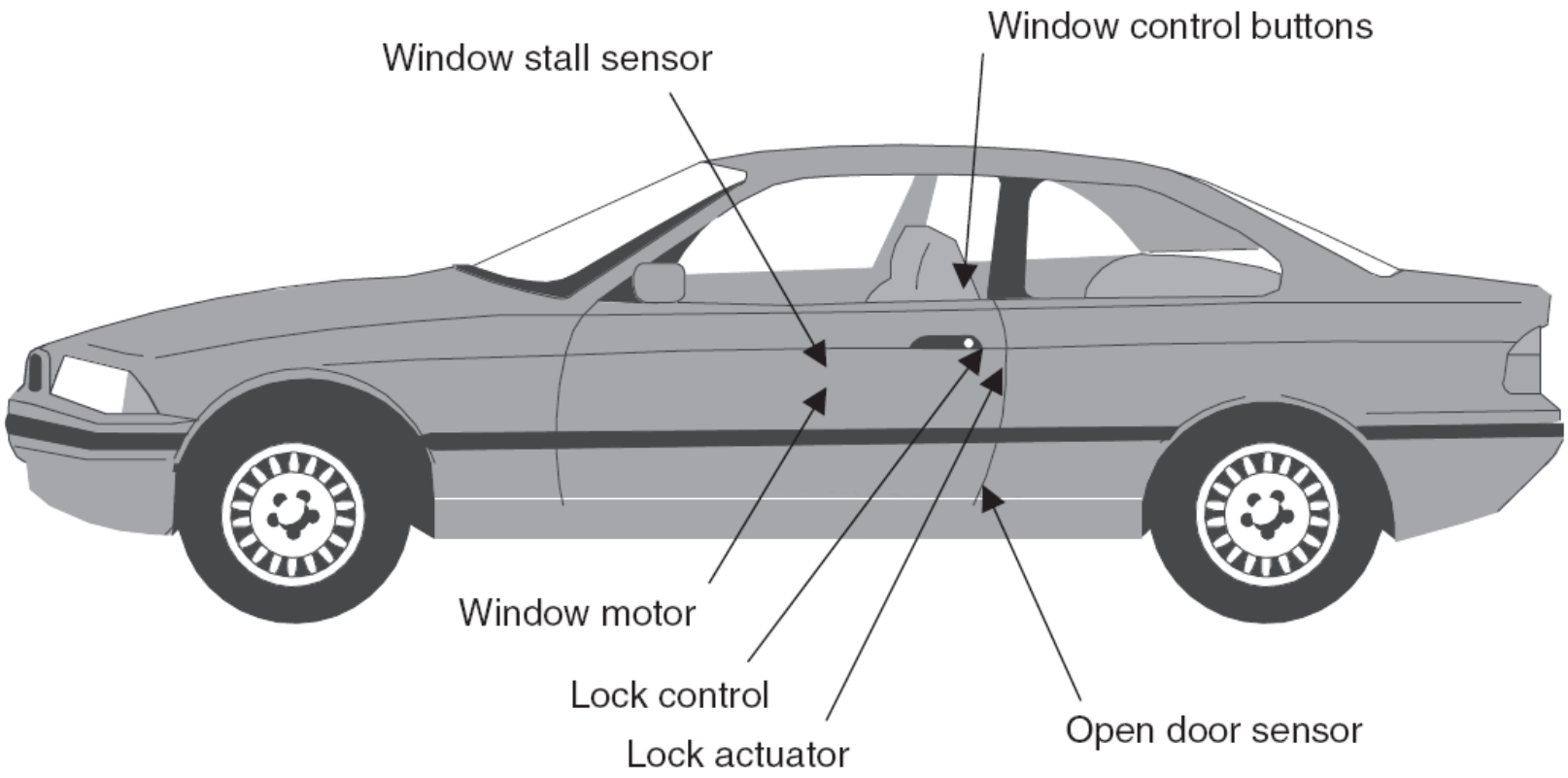
Embedded Systems

- **Embedded system**: is a system that principal function is not computational, but which is controlled by a computer embedded within it.

Examples: Refrigerator

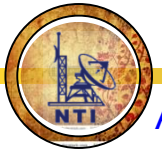


Examples: Car Door

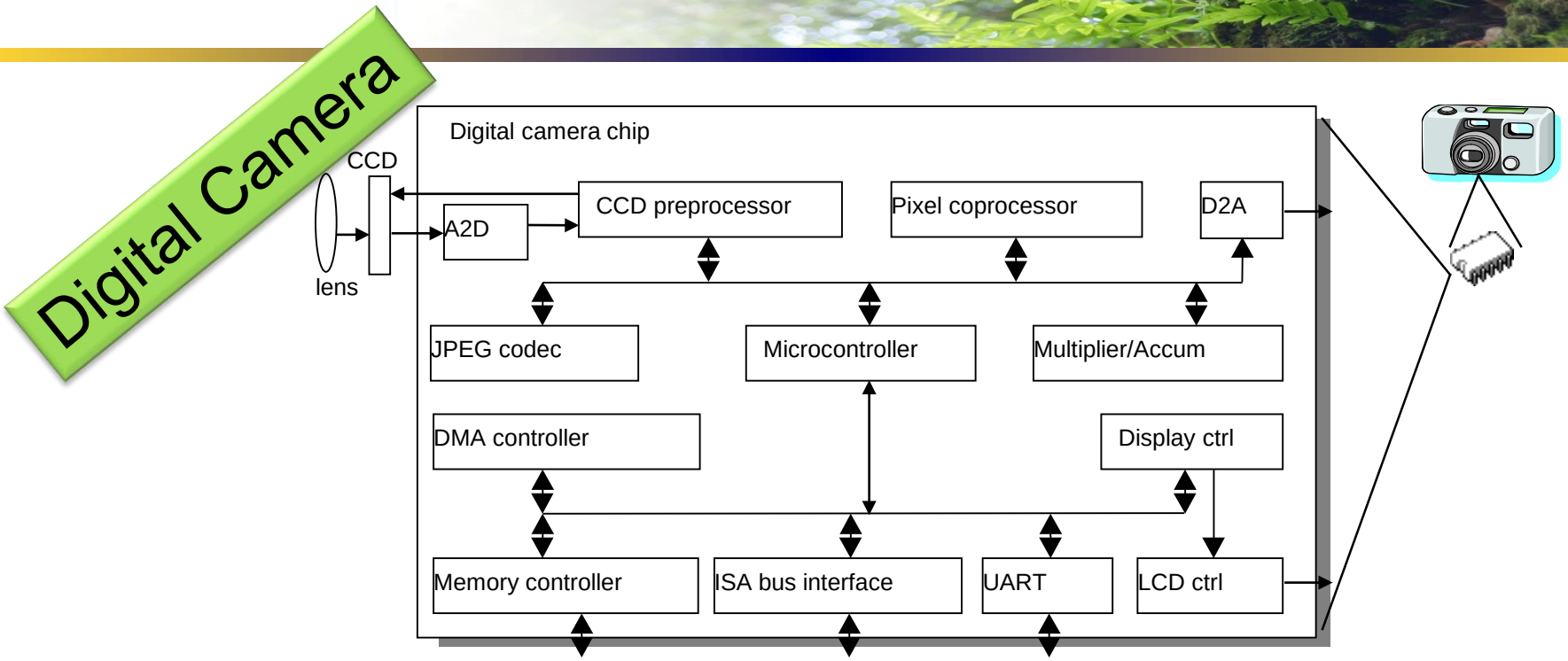


Some common characteristics

- Single-functioned
  - Executes a single program, repeatedly
- Tightly-constrained
  - Low cost, low power, small, fast, etc.
- Reactive and real-time
  - Continually reacts to changes in the system's environment
  - Must compute certain results in real-time without delay



ES example

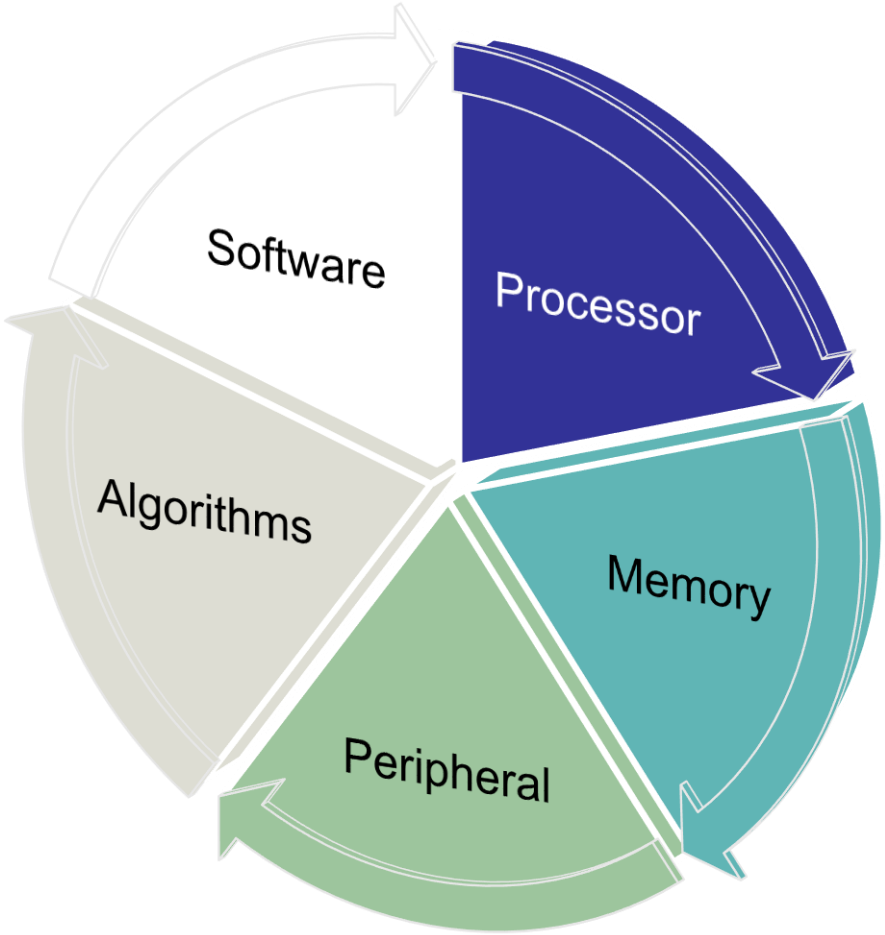


- Single-functioned -- always a digital camera
- Tightly-constrained -- Low cost, low power, small, fast
- Reactive and real-time -- only to a small extent

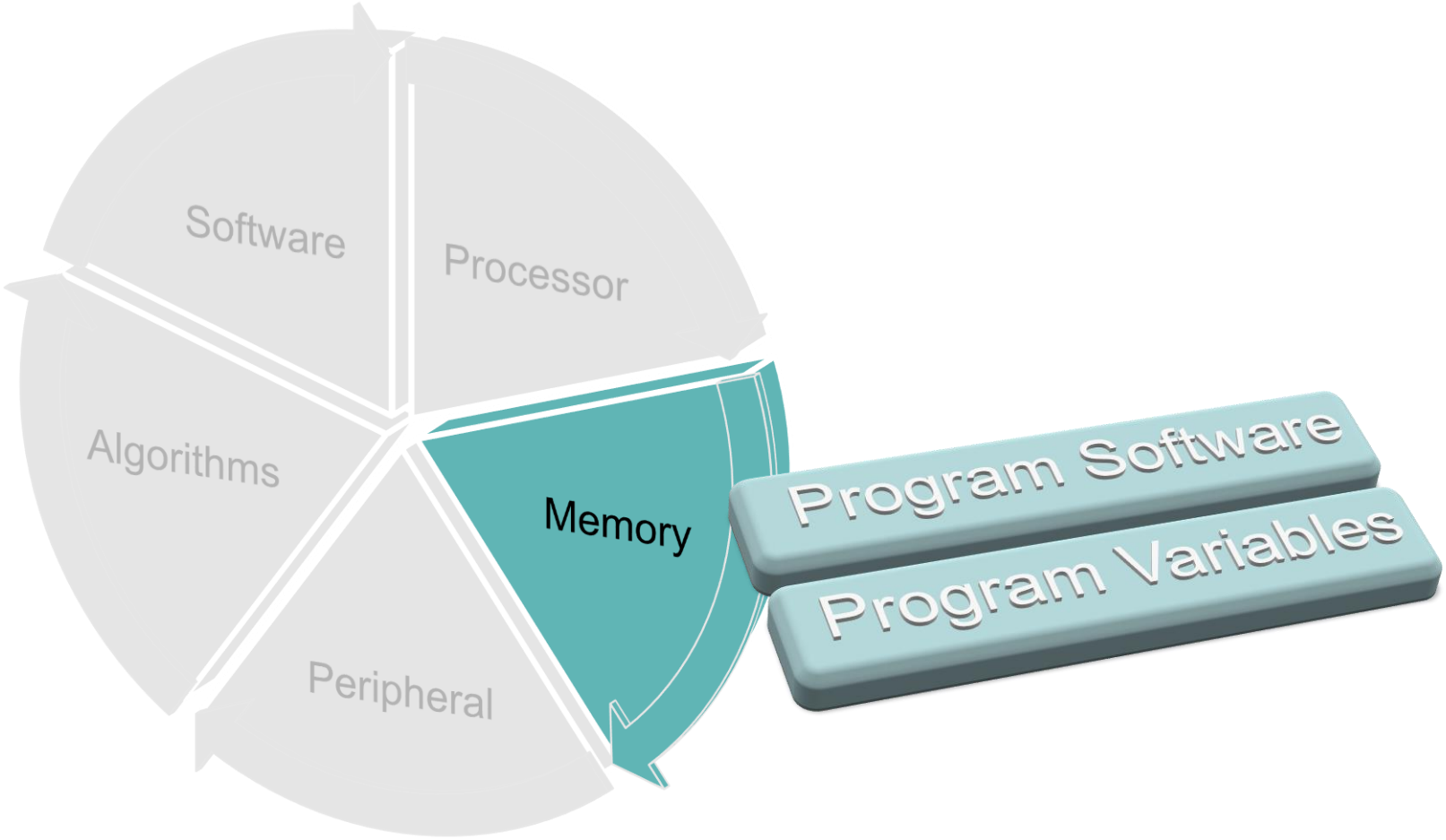
Why Embedded Systems

- Replacement for discrete logic-based circuits
- Provide functional upgrades
- Provide easy maintenance upgrades
- Improves mechanical performance
- Protection of intellectual property
- Replacement for analogue circuits

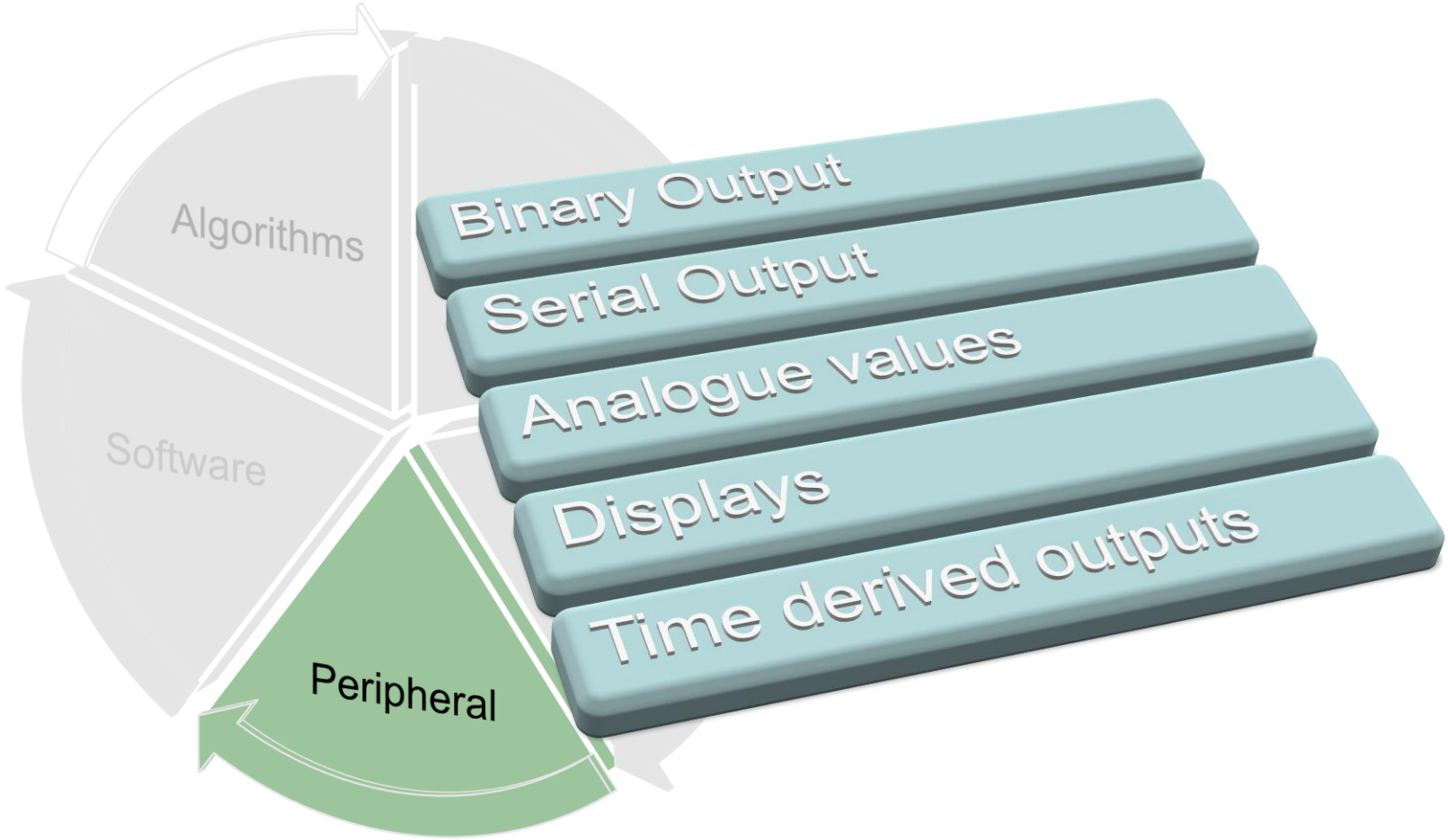
ES components



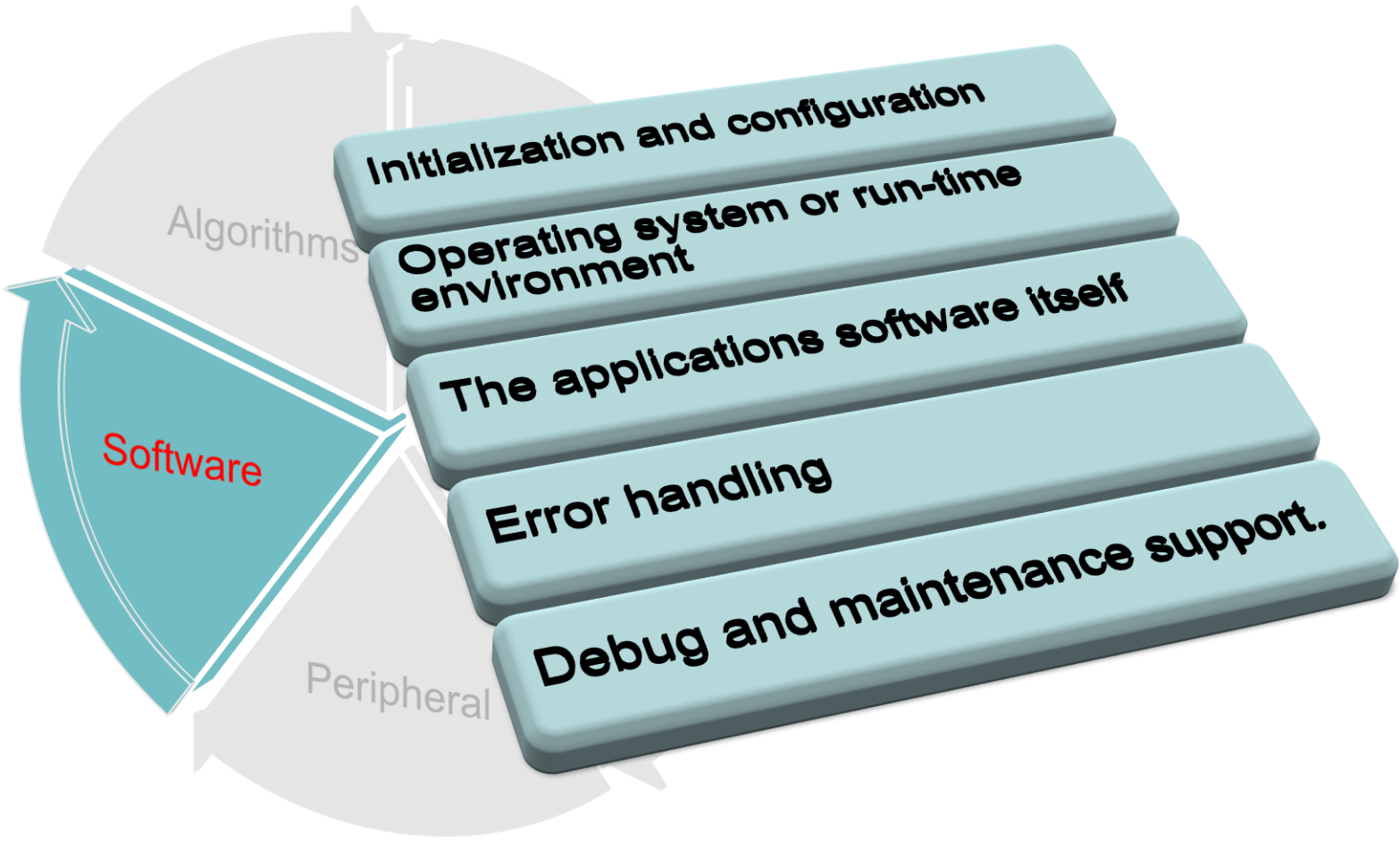
ES components



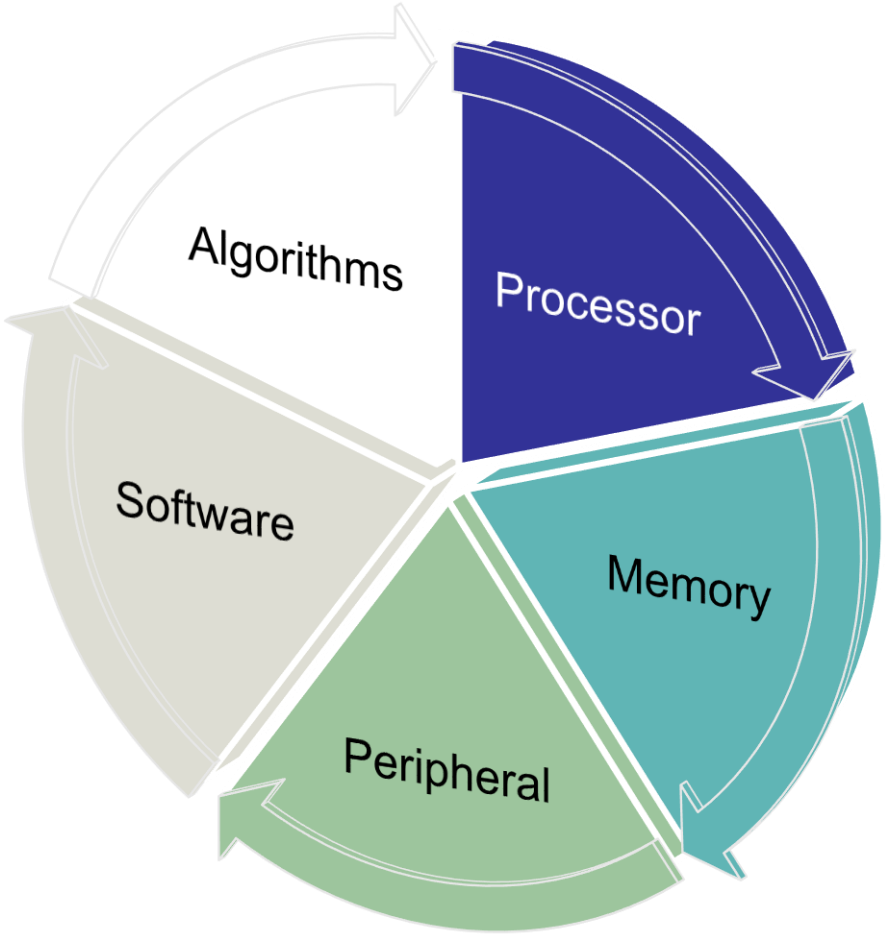
ES components



ES components



ES components



Introduction

FPGA
Field Programmable
Gate Array

ASIC
Application Specified
Integrated Circuit

Non ALU Chips

Hardware Platforms

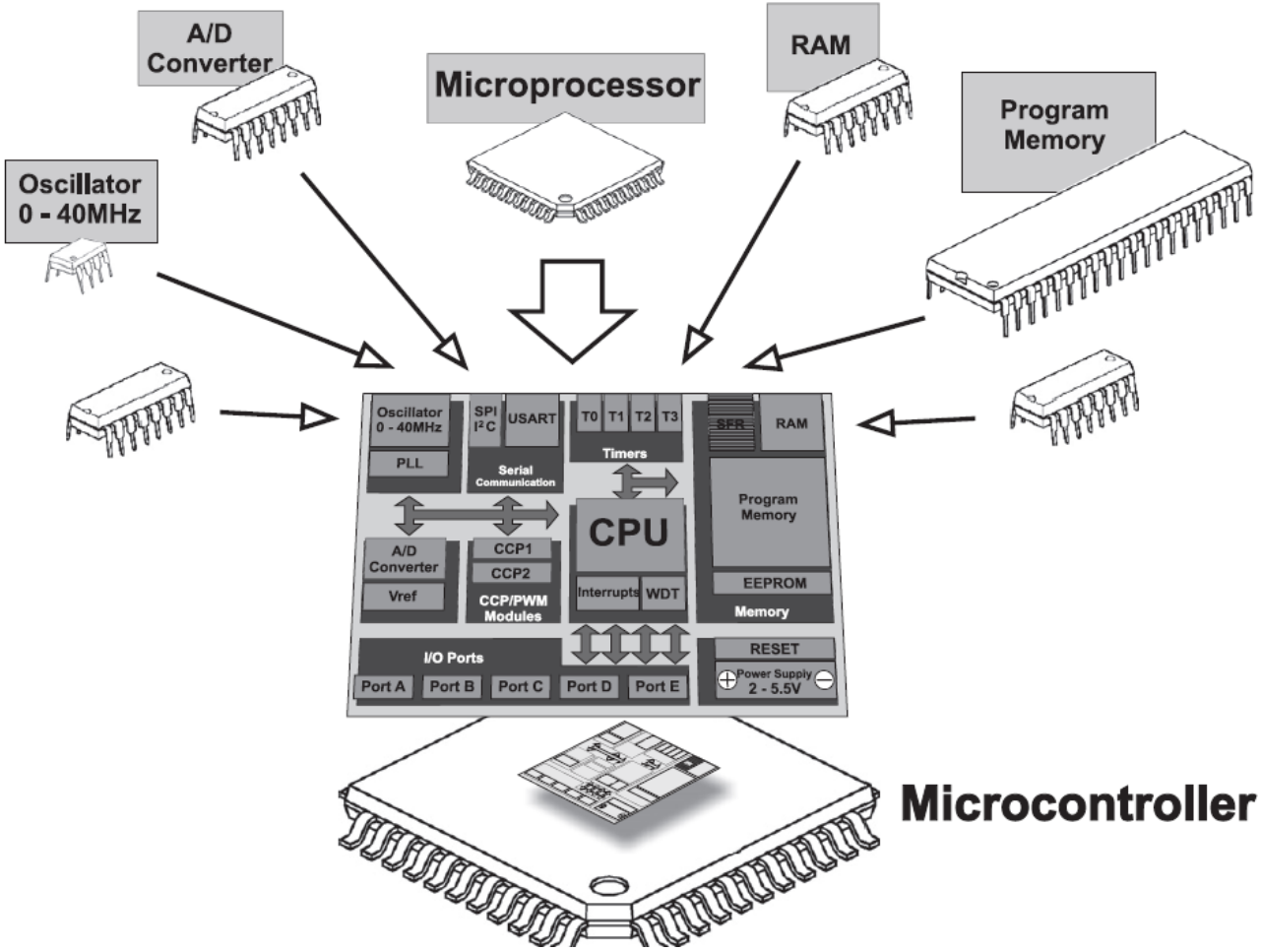
ALU Chips

Microprocessor

Microcontroller

DSP
Digital Signal PROCESSORS

GPP vs UC


A/D Converter
Microprocessor
RAM
Program Memory
Oscillator 0 - 40MHz
Oscillator 0 - 40MHz
PLL
SPI I²C
USART
Serial Communication
T0 T1 T2 T3
Timers
SFR
RAM
Program Memory
CPU
A/D Converter
Vref
CCP1
CCP2
CCP/PWM Modules
Interrupts
WDT
EEPROM
Memory
I/O Ports
Port A
Port B
Port C
Port D
Port E
RESET
Power Supply 2 - 5.5V
Microcontroller

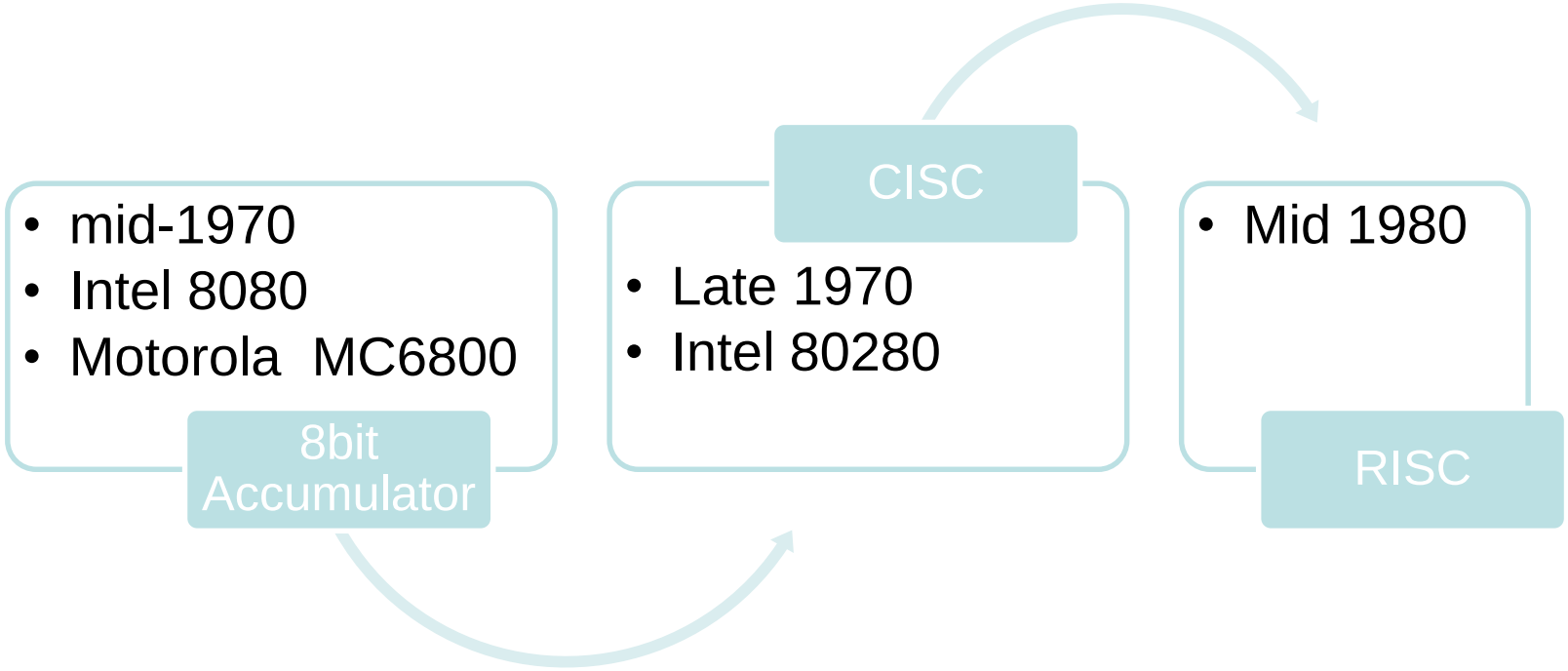
Introduction

- Which Core we have to use (DSP/μC/μP/IC)???
  - Instruction decoder style
  - Memory architecture
  - Mathematic Unit
  - Extra flavors

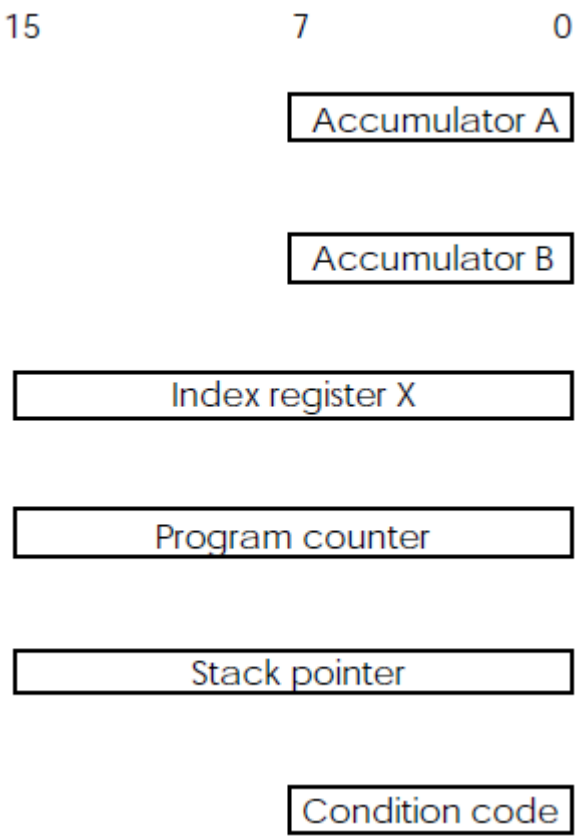
Embedded processors

Challenges and Opportunities

3 Architectures



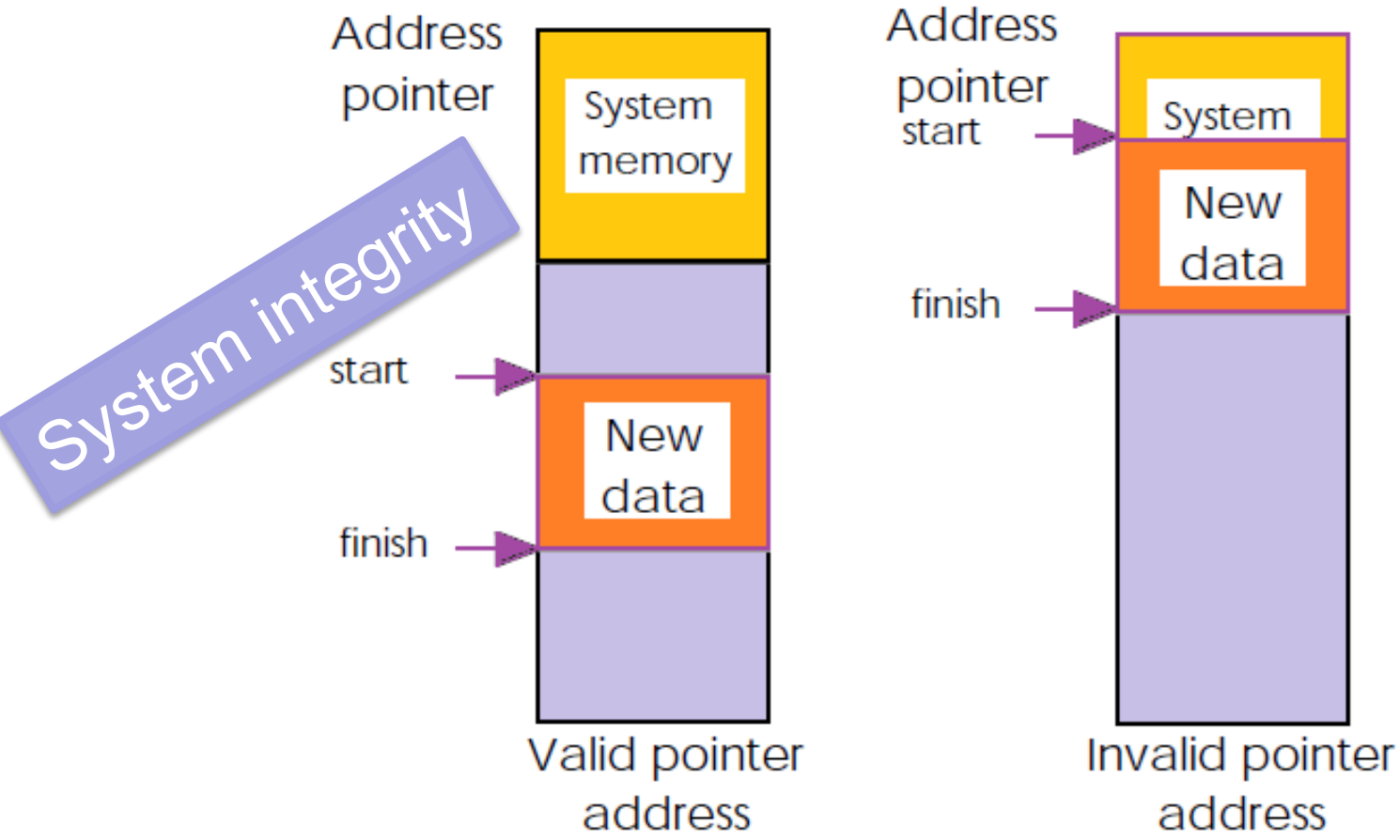
8 bit Architecture



Summary of 8Bit

- logical operations, can operate on direct memory using the index register to act as pointer
  - •Extra memory cycles
  - •Wider registers
  - •Unsigned resolution of only 256 bits
  - •Bank switching and program overlays

Summary of 8Bit



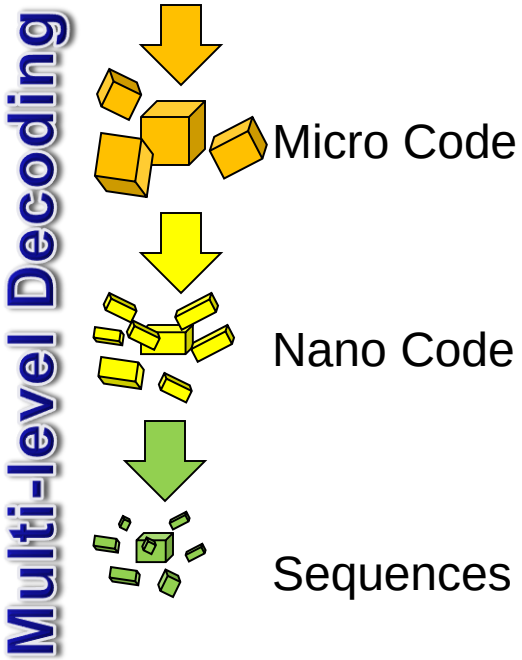
- ✓ The larger data size was needed to support higher precision arithmetic.
- ✓ The increased address space was needed to support bigger blocks of memory for larger programs
- ✓ The more complex the instruction, the less needed for a particular function and therefore the less memory that the system needed.

Survival Developers

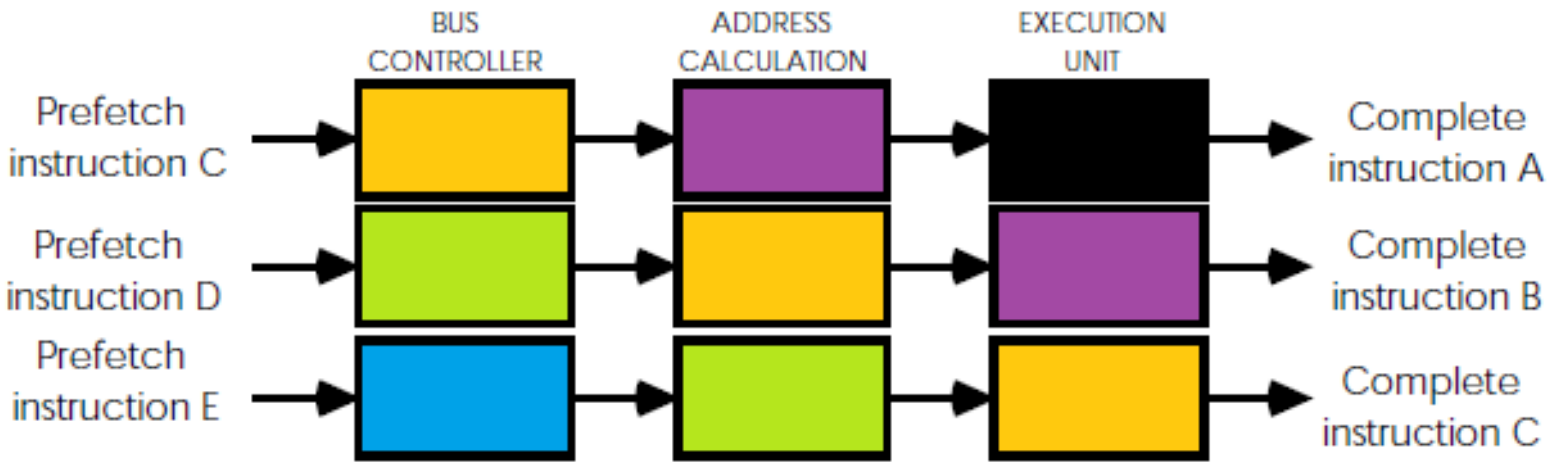
CISC

Complex Instruction Set Computers

Opcode 16 bits	operands 16 bits	operands 16 bits



Pipeline

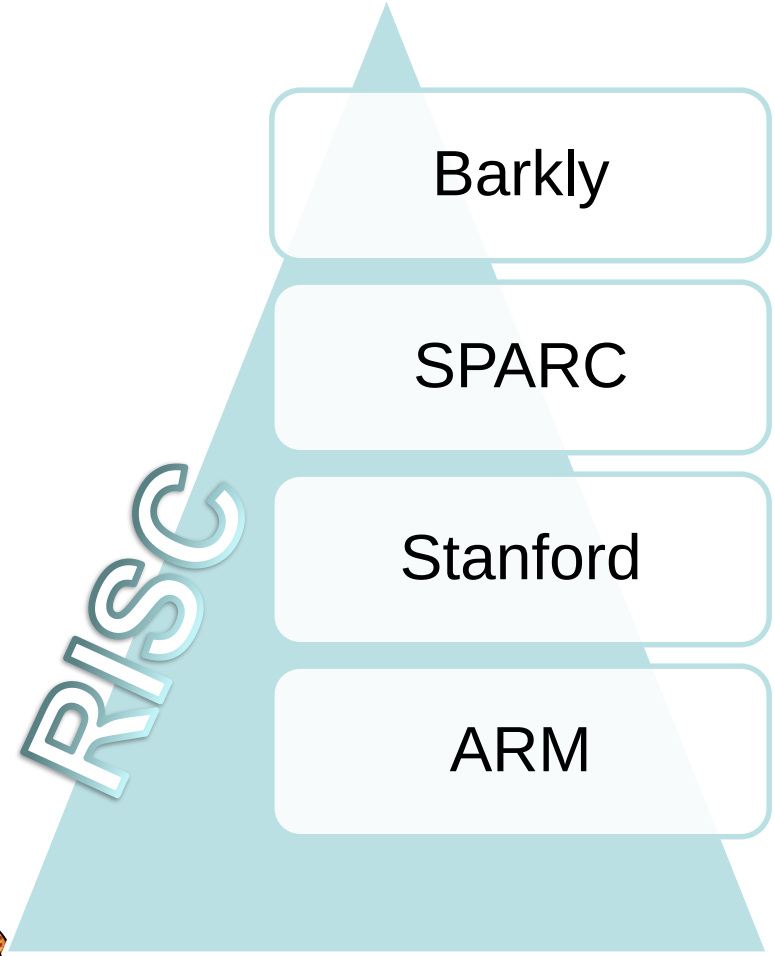


- Consumes real estate
- Increases pipeline delays

Gain higher speeds by reducing the amount of work done in each stage

RISC

- All instructions will be executed in a single cycle (op-code must be of a fixed width).
- Memory will only be accessed via load and store instructions (No memory manipulation).
- no micro-coding.


RISC
Barkly
SPARC
Stanford
ARM

CISC vs. RISC

Consider the program fragments:

CISC

```
mov ax, 10
mov bx, 5
mul bx, ax
```

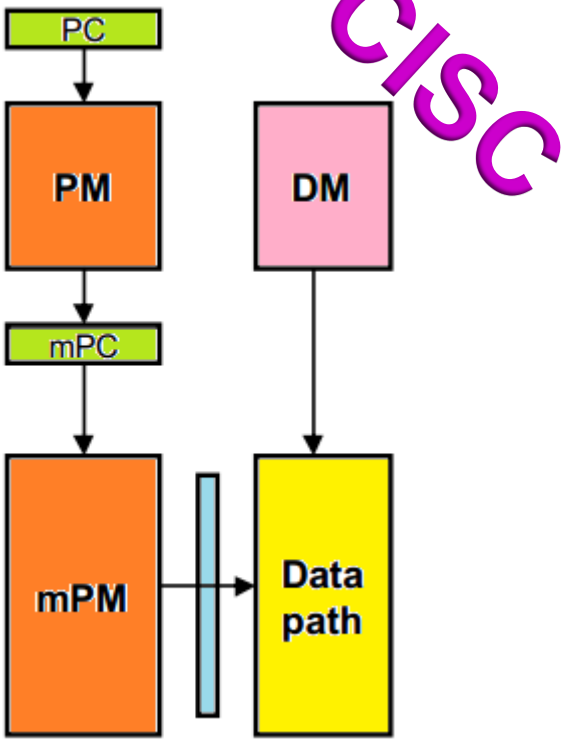
RISC

```
        mov ax, 0
        mov bx, 10
        mov cx, 5
Begin   add ax, bx
        loop Begin
```

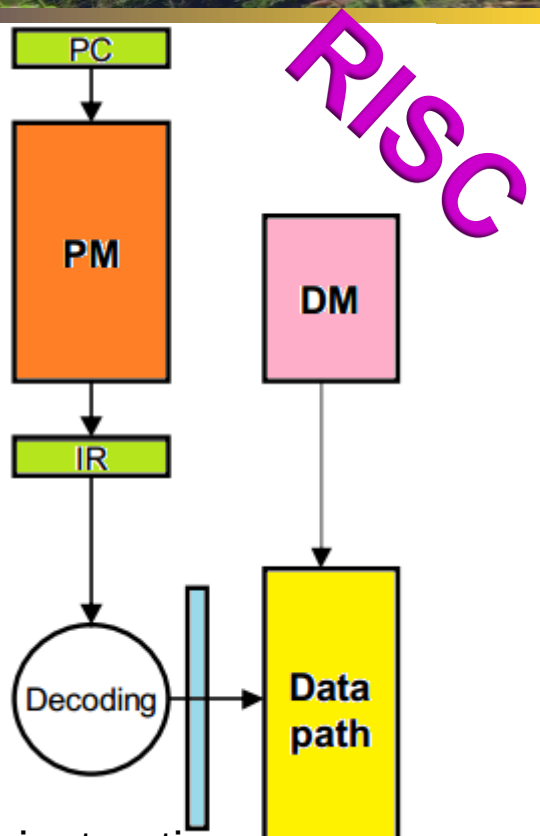
CISC clock cycles = (2 movs × 1 cycle) + (1 mul × 30 cycles)
= 32 cycles

RISC clock cycles = (3 movs × 1 cycle) + (5 adds × 1 cycle) +
(5 loops × 1 cycle)
= 13 cycles

CISC vs. RISC



Complex instructions possible
1 Instruction = n microinstructions



Simple instructions
No microprogramming
RISC PM = 2X CISC

CISC vs. RISC

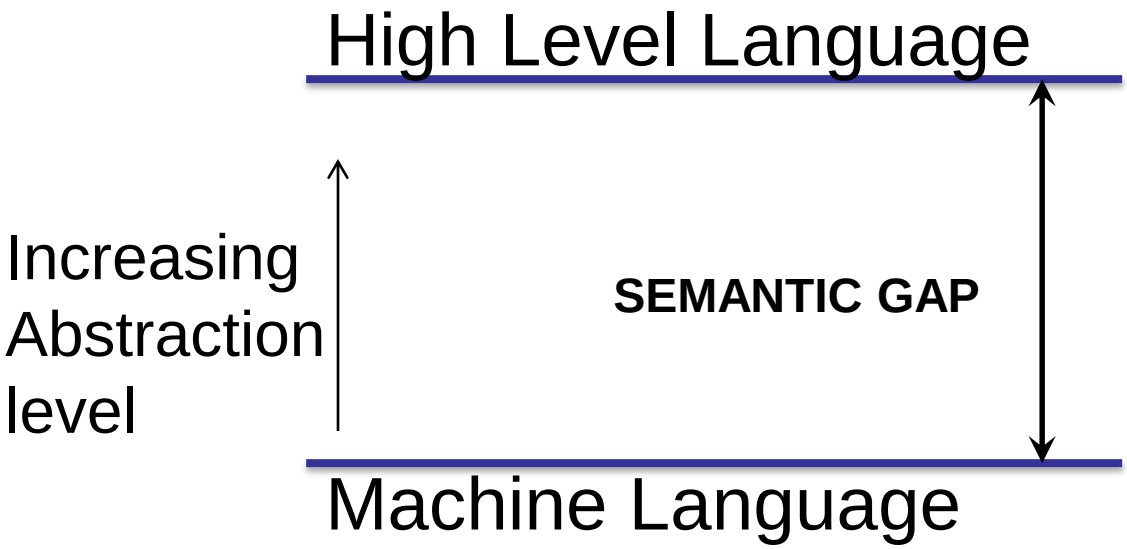
RISC

- Simple instructions, few in Number
- Fixed length instructions
- Complexity in compiler
- Only LOAD/STORE instructions access memory
- Few addressing modes

CISC

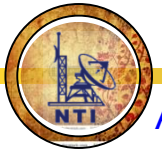
- Many complex instructions
- Variable length instructions
- Complexity in microcode
- Many instructions can access memory
- Many addressing modes

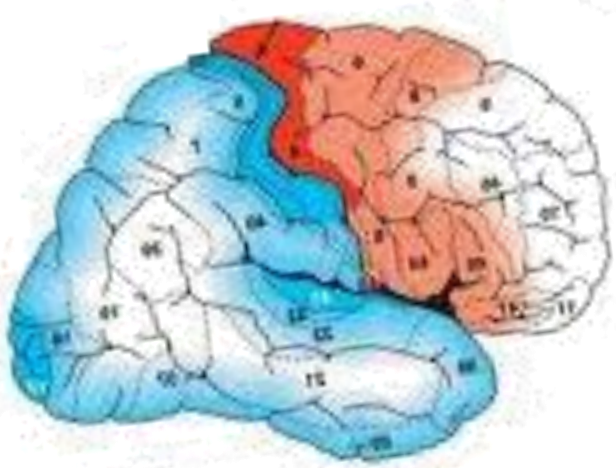
CISC vs. RISC


High Level Language
Increasing
Abstraction
level
SEMANTIC GAP
Machine Language

Recall

- Which Core we have to use (DSP/μC/μP/IC)???
  - Instruction decoder style
  - Memory architecture
  - Mathematic Unit
  - Extra flavors





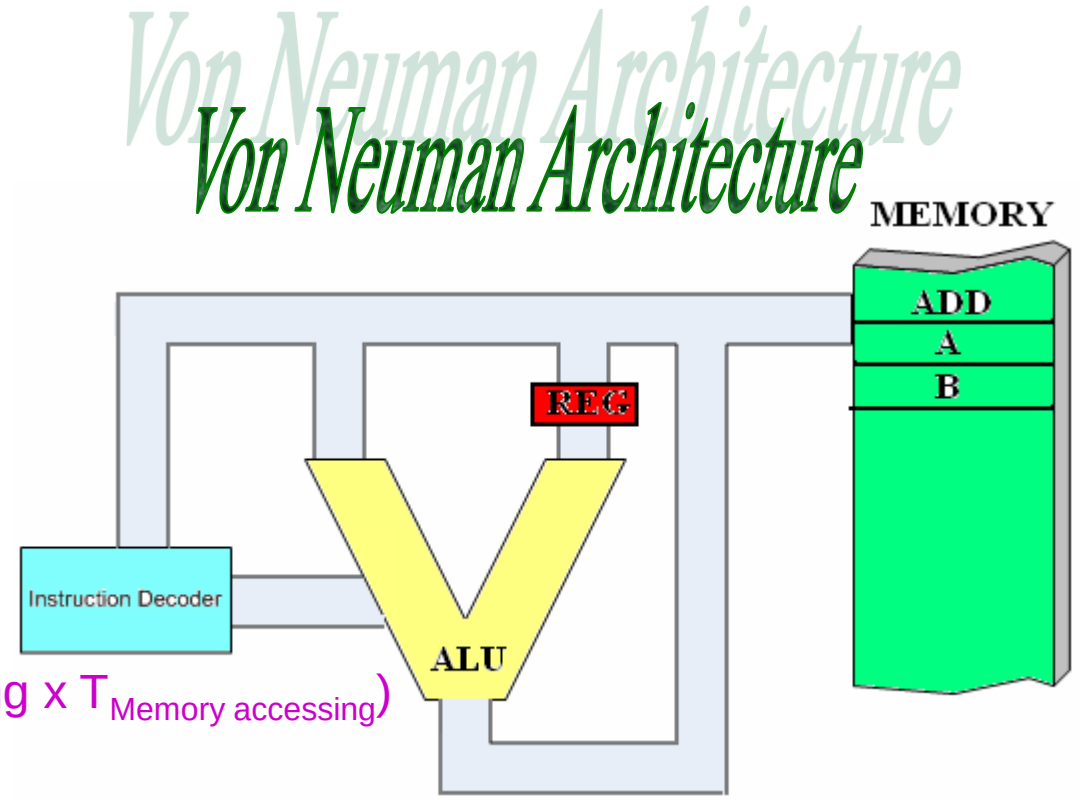
MEMORY ORGANIZATION

Microprocessor
Revision

- Programming example
  ADD A,B

- How many times does the system access the memory?

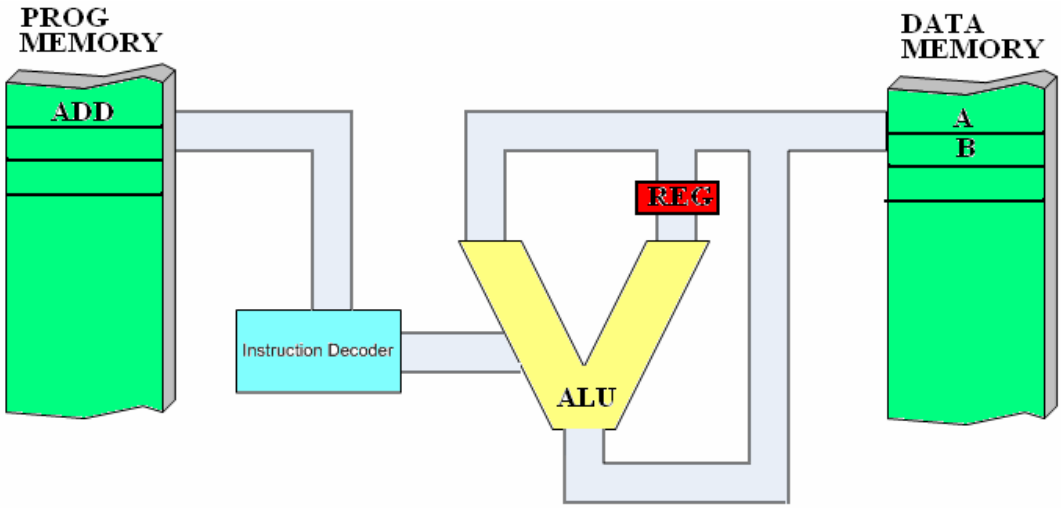
- $T_{total}=T_{ALU} + (\#\ memory\ accessing \times T_{Memory\ accessing})$



Microprocessor
Revision

- Programming example
  ADD A,B
- How many times does the system access the memory?

Time saving ratio?



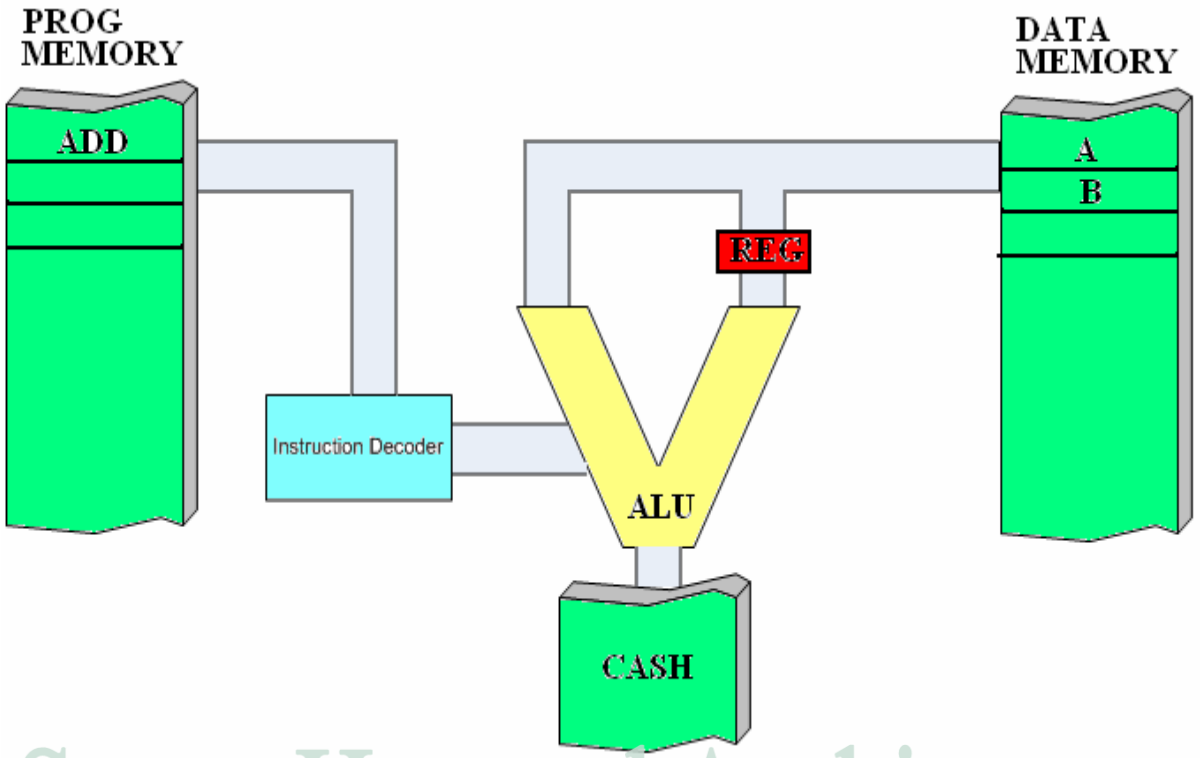
Harvard Architecture

Harvard Architecture

Microprocessor

Revision

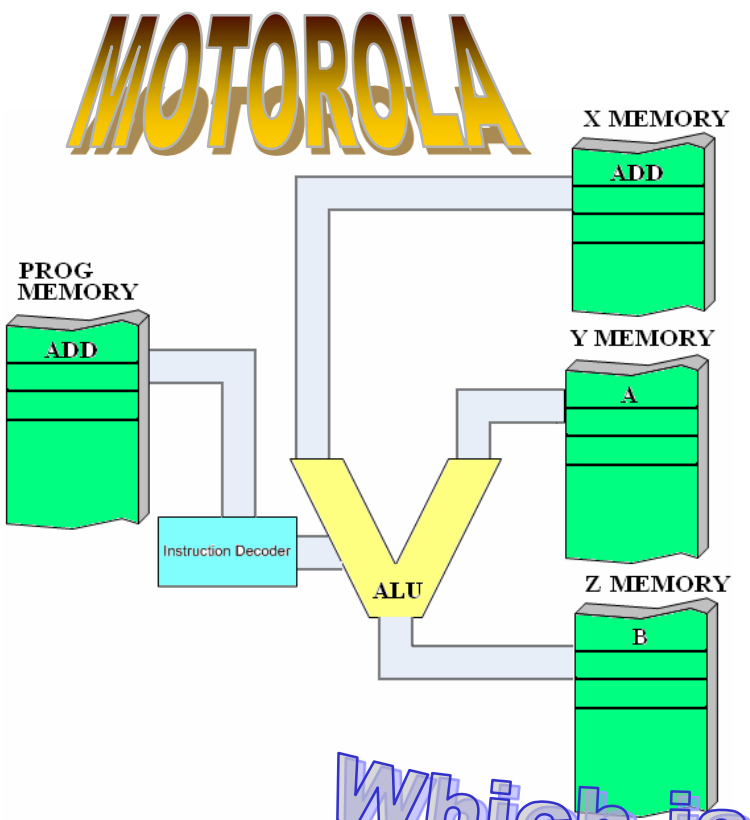
- Programming example
  ADD A,B
- How many times does the system access the memory?
- Time saving ratio?

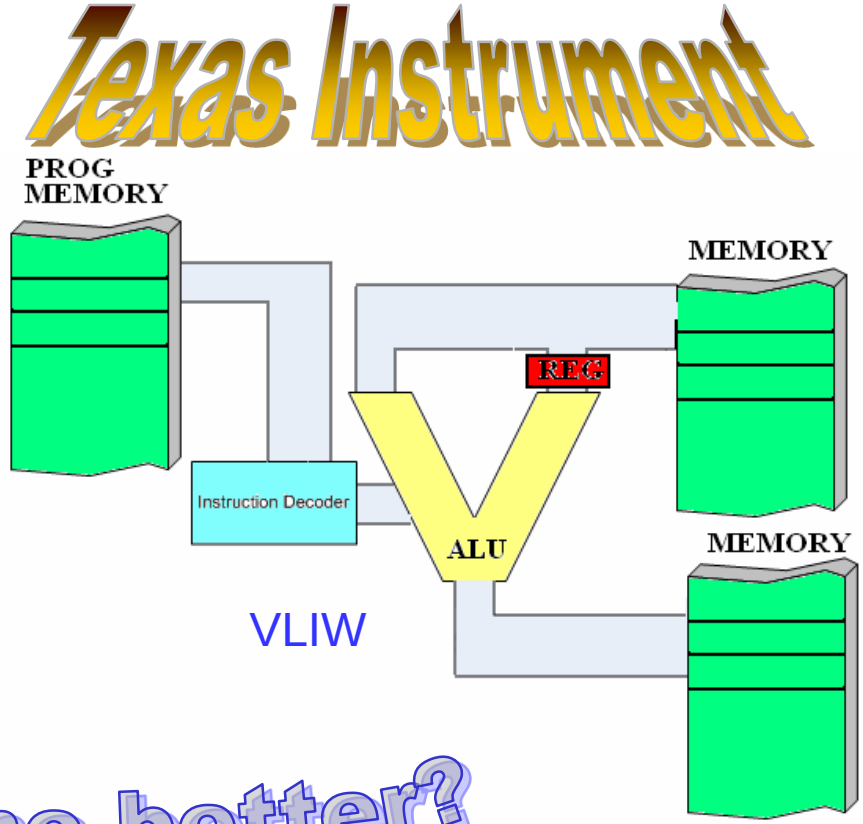


Super Harvard Architecture

Super Harvard Architecture

Thrive to DSP



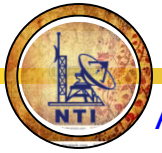


Which is the better?

Recall

- Which Core we have to use (DSP/μC/μP/IC)???
  - Instruction decoder style
  - Memory architecture
  - Mathematic Unit
  - Extra flavors

Mathematical Unit

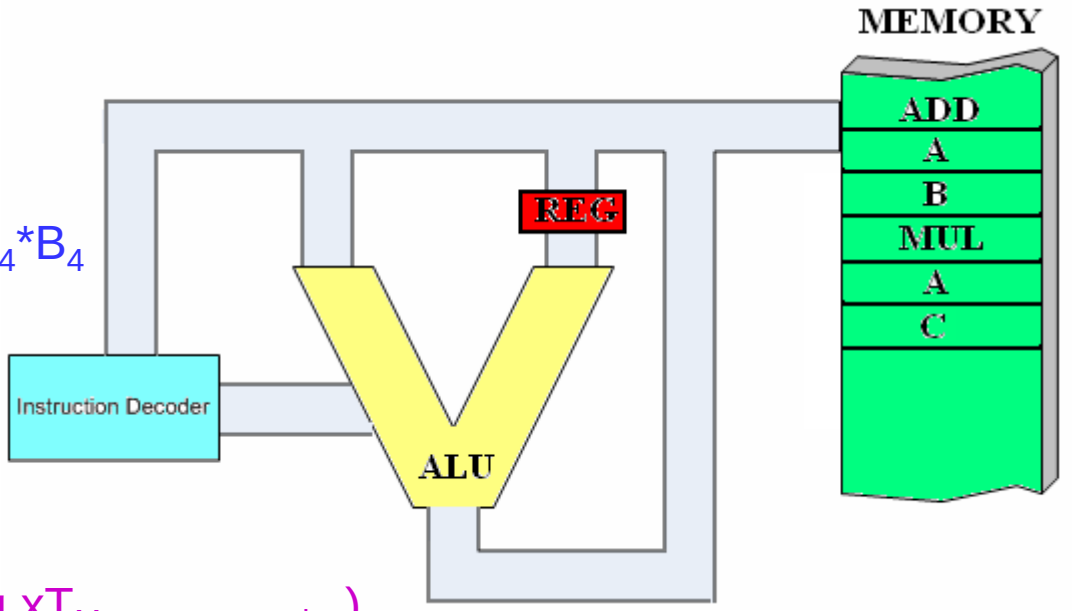


Mathematics

Example

Calculate the required time to execute the previous

Result = $A_1*B_1+A_2*B_2+A_3*B_3+A_4*B_4$



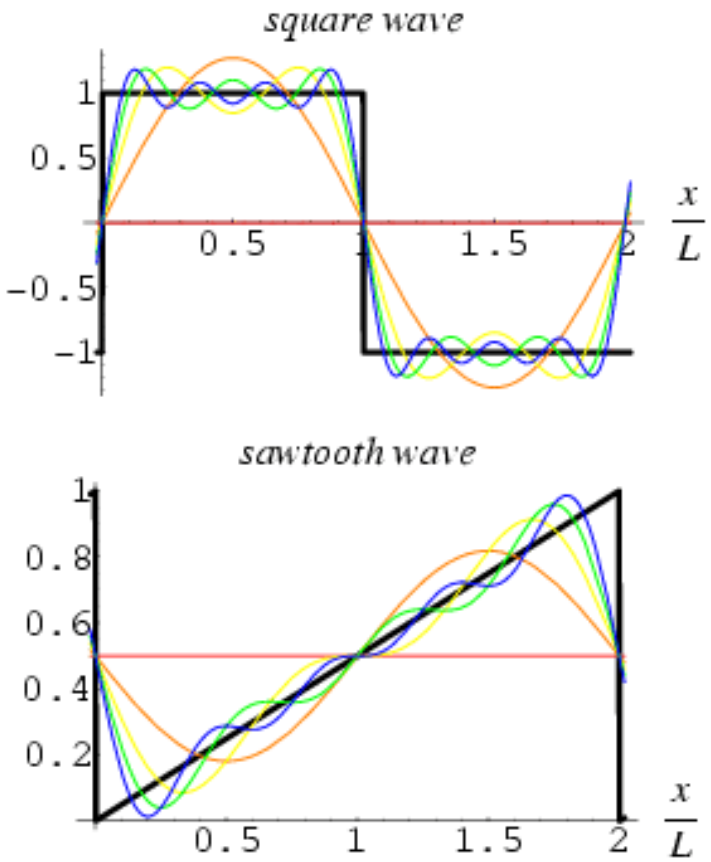
$T_{total}=T_{ALU}$ + (# memory accessing x$T_{Memory\ accessing}$)

ALU

Mathematics

Fourier series shine!!!

$$f(x) = \frac{1}{2} a_0 + \sum_{n=1}^{\infty} a_n \cos(n x) + \sum_{n=1}^{\infty} b_n \sin(n x),$$



Mathematics

Fourier Transform shine!!!

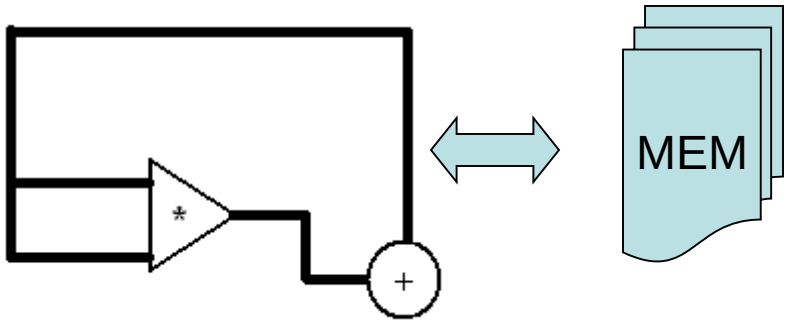
$$f(x) = \frac{1}{2} a_0 + \sum_{n=1}^{\infty} a_n \cos(n x) + \sum_{n=1}^{\infty} b_n \sin(n x),$$

Example:

Calculate the required time to execute the previous example

Result = $A_1*B_1+A_2*B_2+A_3*B_3+A_4*B_4$

Time saving ratio?



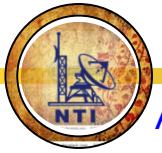
MAC

MAC

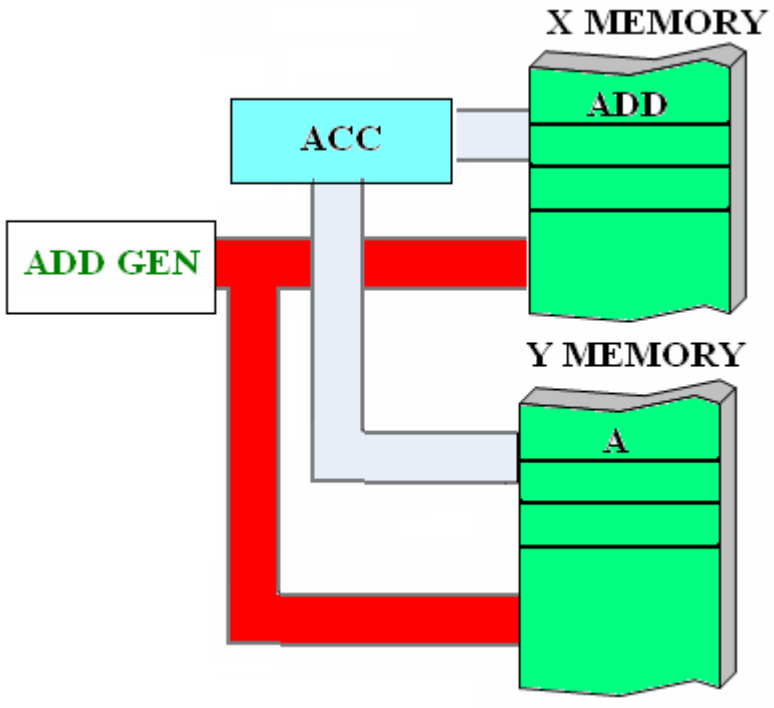
Multiply and Accumulate

Recall

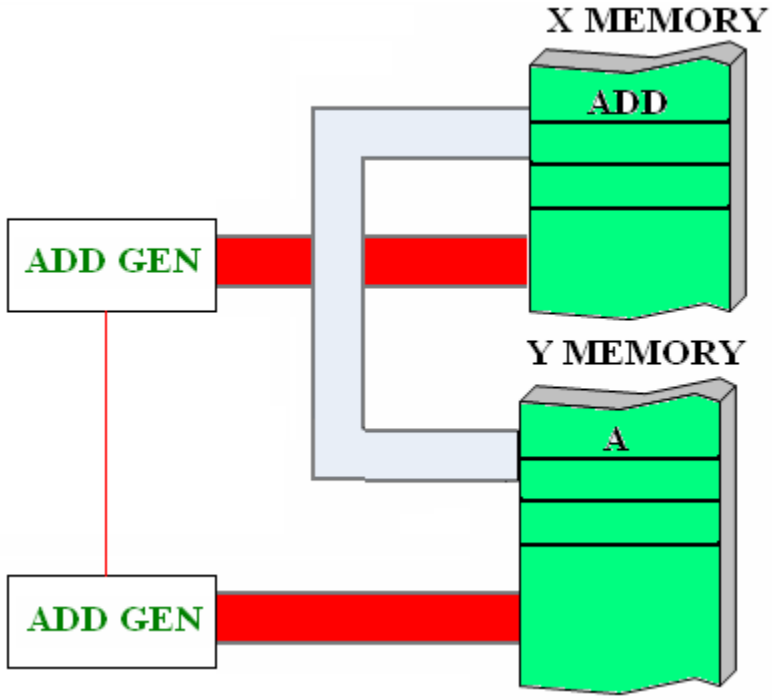
- Which Core we have to use (DSP/μC/μP/IC)???
  - Instruction decoder style
  - Memory architecture
  - Mathematic Unit
  - Extra flavors



Array handling

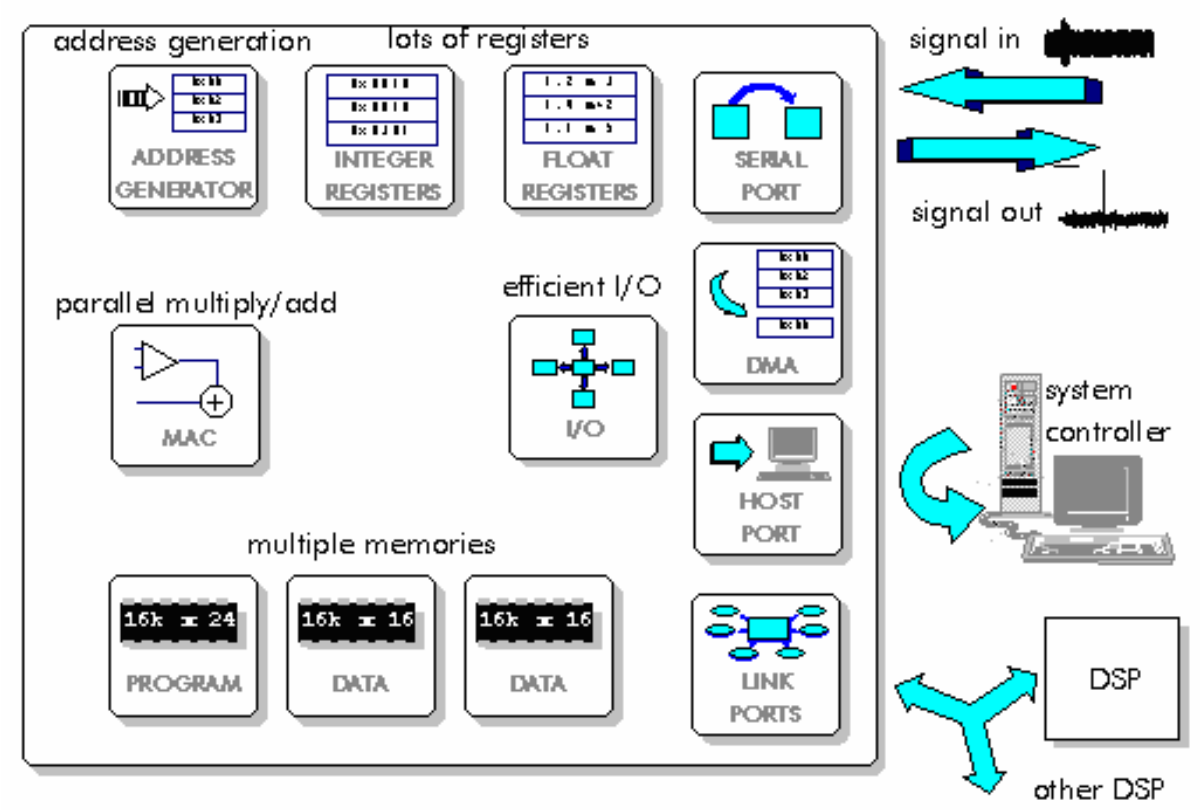


μP



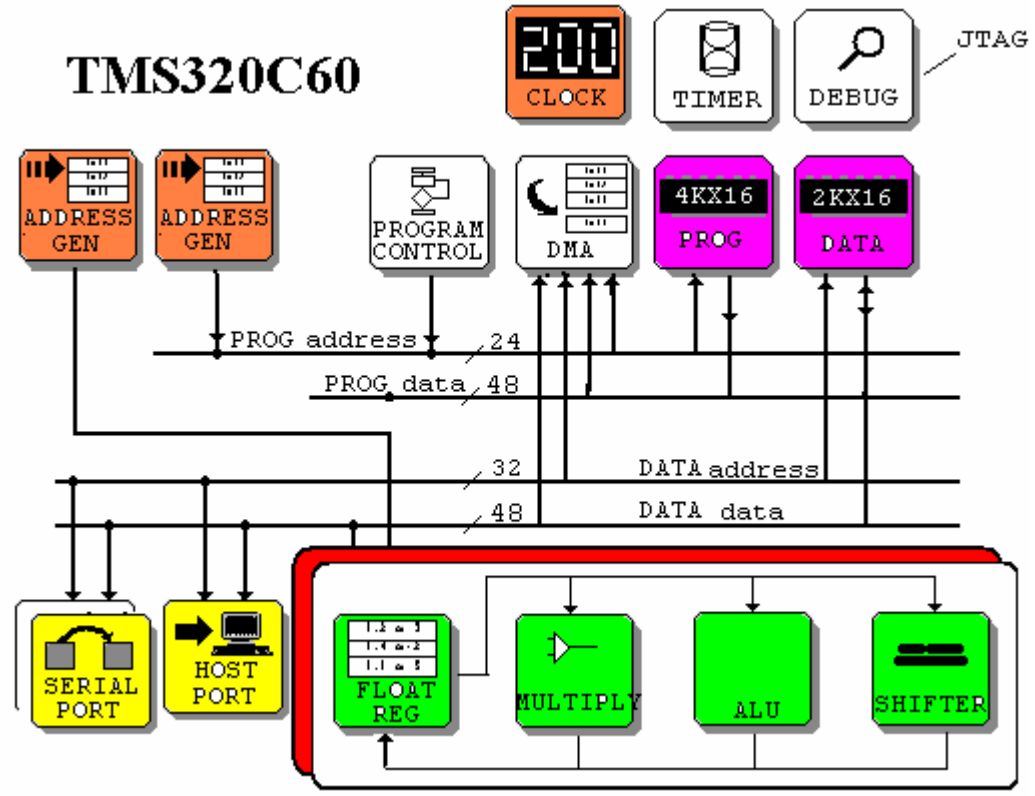
DSP

Welcome to world

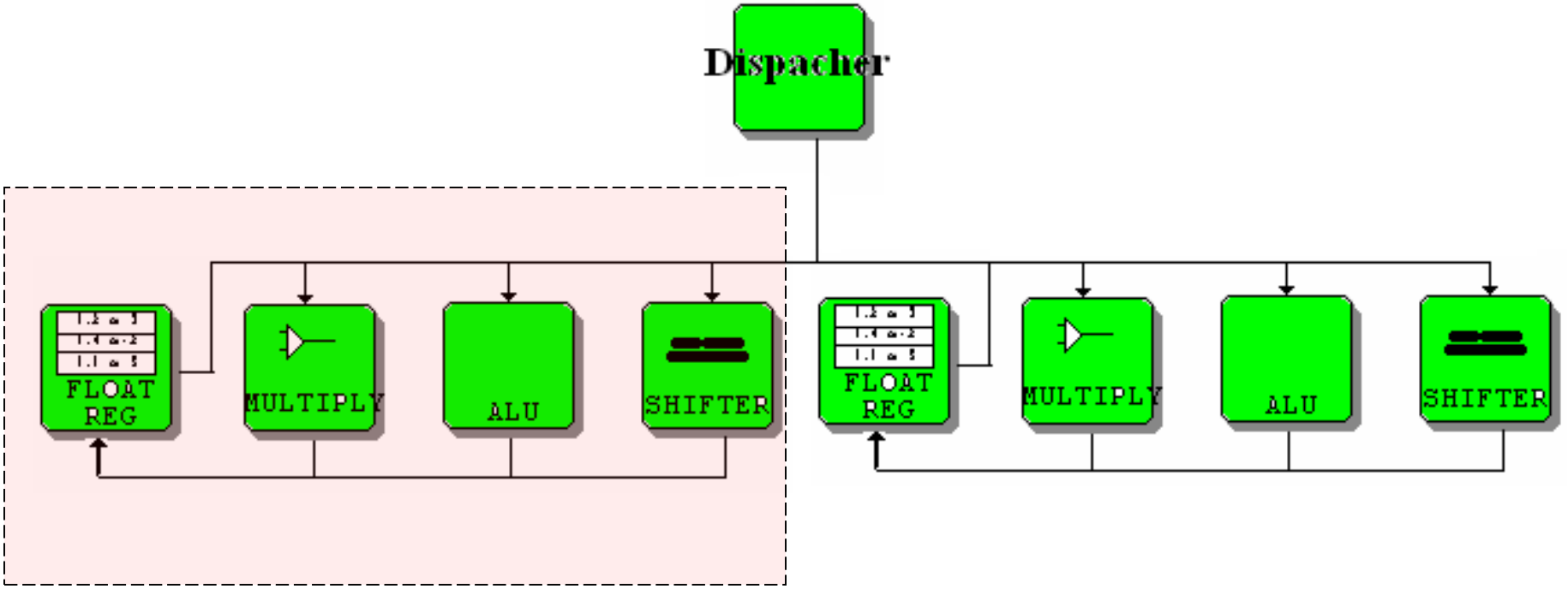


TMS320C6x

1600 MIPS



Parallel Processing



Recall

FPGA
Field Programmable
Gate Array

ASIC
Application Specified
Integrated Circuit

Non ALU Chips

Hardware Platforms

ALU Chips

Microprocessor

Microcontroller

DSP
Digital Signal PROCESSORS

Evaluation

Kindly on a piece of paper, evaluate this presentation according to

1. Information adequate.
2. Speed of presentation.
3. Suggestions

Note : DO NOT WRITE YOUR NAME

Thank you!!!