Assignment Project Exam Help

Computer architecture: processors

https://powcoder.com

Add WeChat powcoder

EEE8087

Dr Fei Xia and Dr Alex Bystrov

Introduction to processors

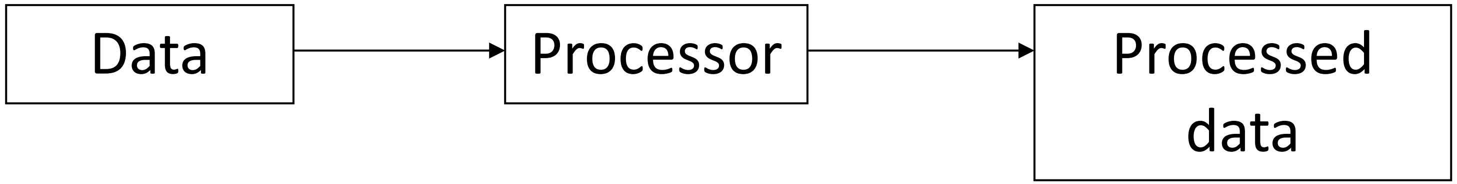
- The brain of the computing system, meant to carry out the intended functionality, as and when needed.

Assignment Project Exam Help

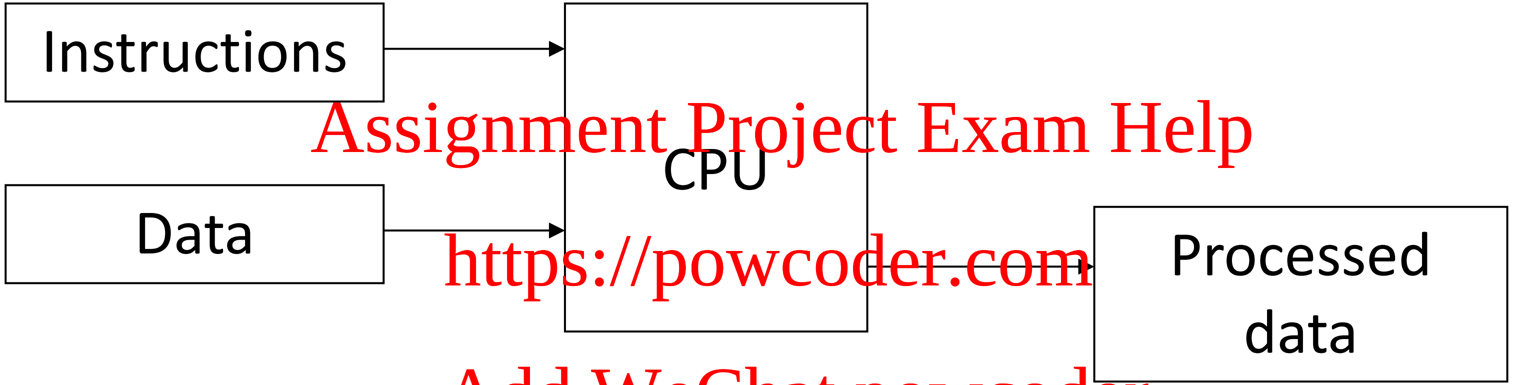
https://powcoder.com

Add WeChat powcoder

A simplified view v1.0



Simplified View v2.0 – data types



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Simplified example of an *instruction:*

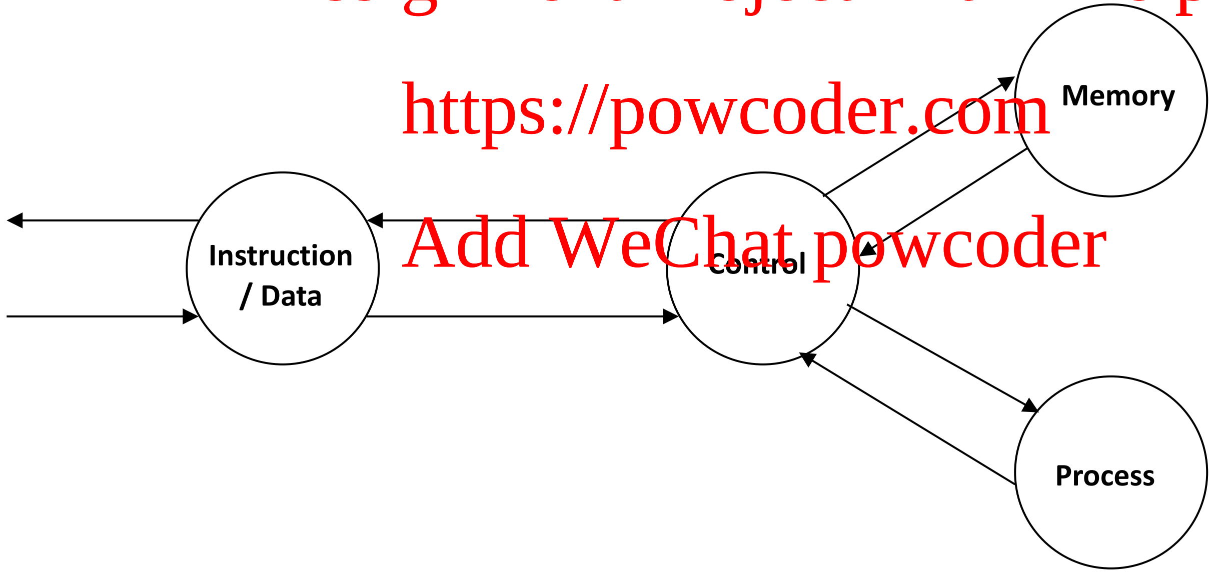
Opcode	Mode	register	Address

Functional view

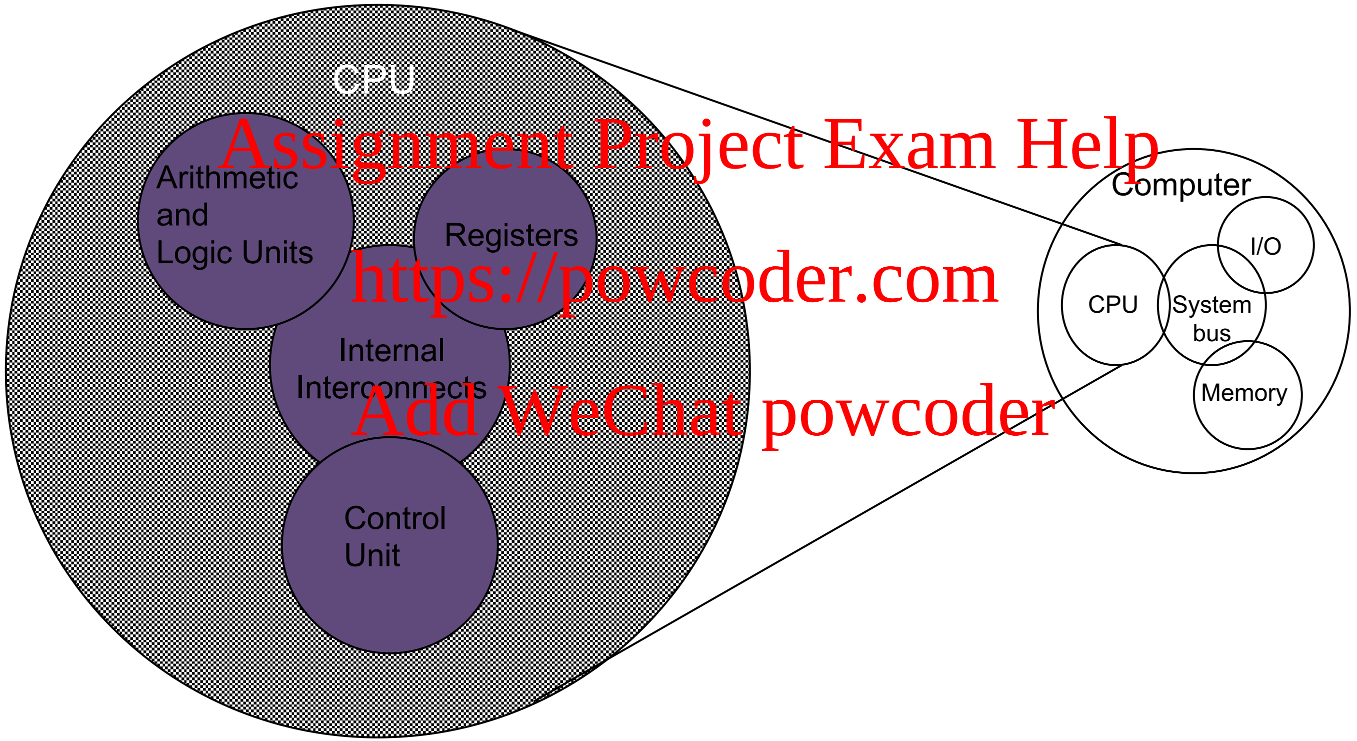
Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

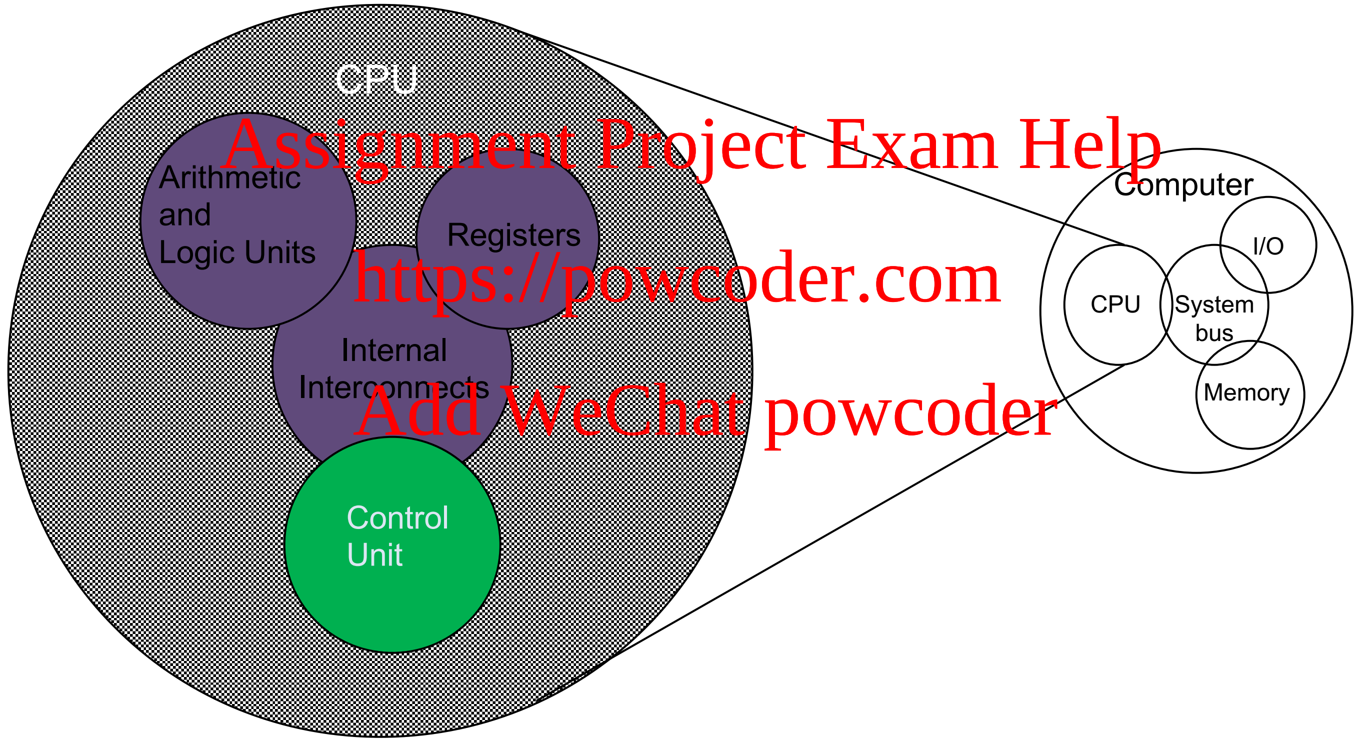


CPU Structure



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Control unit: data flow

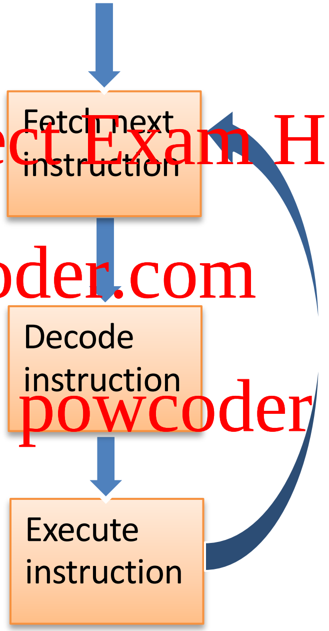


Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

CPU control steps: data flow

- Fetch instructions
- Interpret instructions
- Fetch data
- Process data
- Write data

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder



Simplified view

Data flow: execute

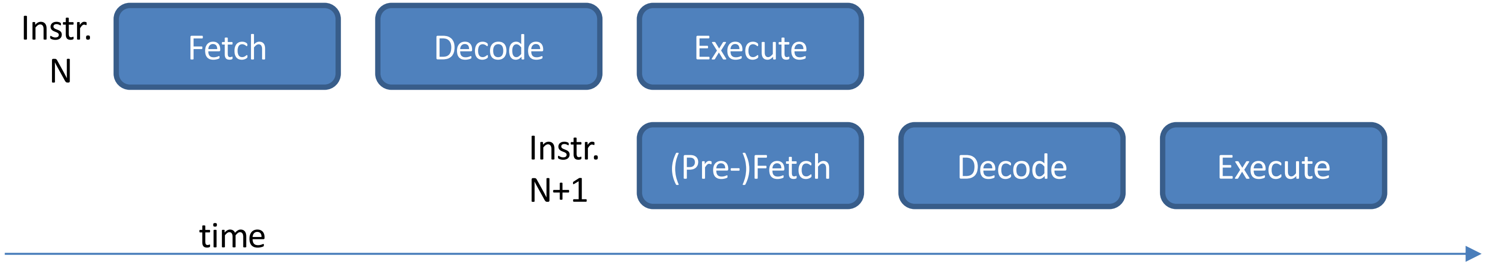
- **Fetch** and **Decode** are very common in all CPU architectures; however, **Execute** flow may take many forms
- Depends on instruction being executed
- May include
  - Memory read/write
  - Input/Output
  - Register transfers
  - ALU operations

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Prefetch

- Some architectures have this additional step to improve performance
- Can fetch next instruction during execution of current instruction (pipelining)
- Called instruction prefetch
- Prefetch can require accessing main memory

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder



Improved Performance through Prefetch

- Prefetch offers good performance as it reduces the latency between CPU and the main memory
- But performance is not doubled:
  - Fetch usually shorter than execution
    - Prefetch more than one instruction?
  - Any jump or branch means that prefetched instructions are not the required instructions
- Add more stages or time multiplex the stages to improve performance

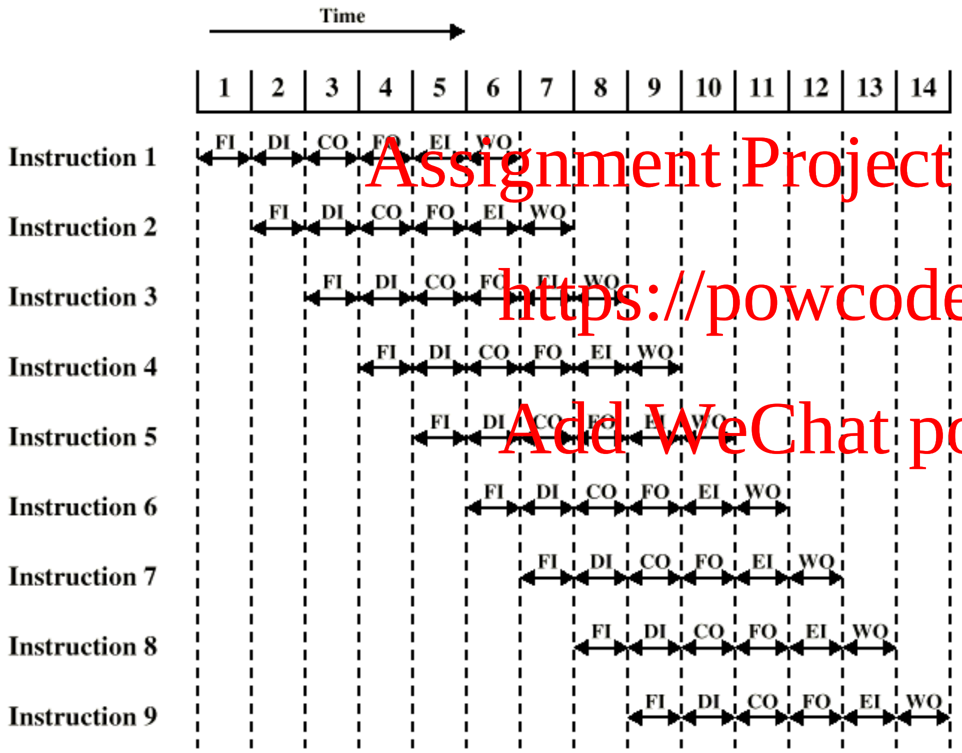
Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Pipelining

- Detailed data flow
  - Fetch instruction
  - Decode instruction
  - Control operand addresses
  - Fetch operands
  - Execute instructions
  - Write result
- Overlap these operations

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Timing of Pipeline – 6 stages



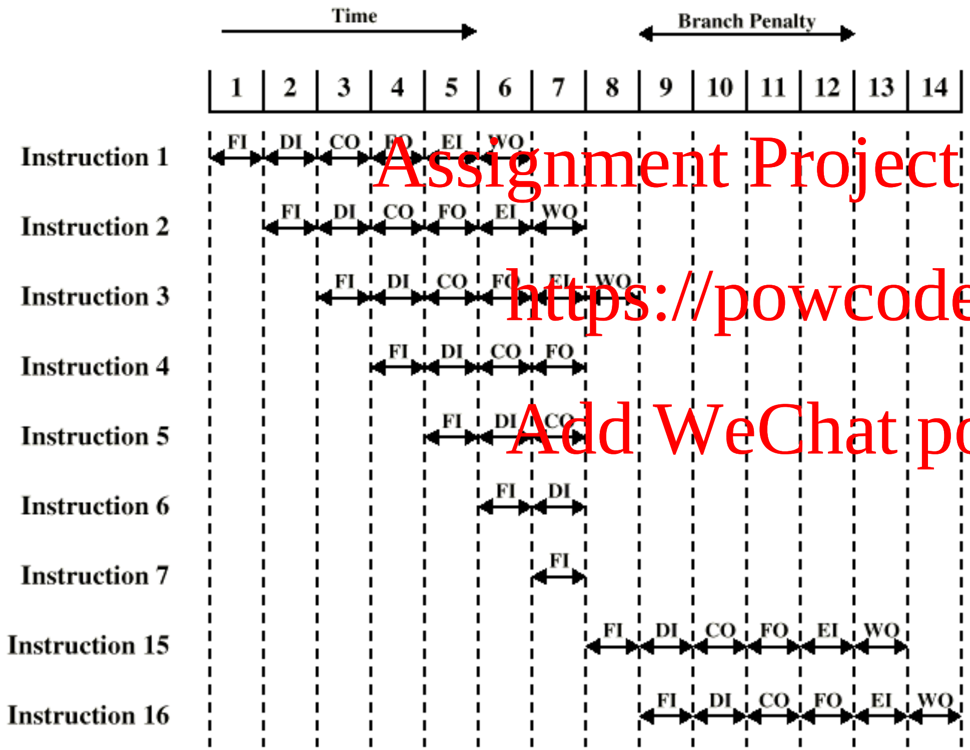
Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- FI: fetch instr.
- DI: decode instr.
- CO: control operand address
- FO: fetch operands
- EI: execute instructions
- WO: write-back operands

Branch in a Pipeline



Assignment Project Exam Help

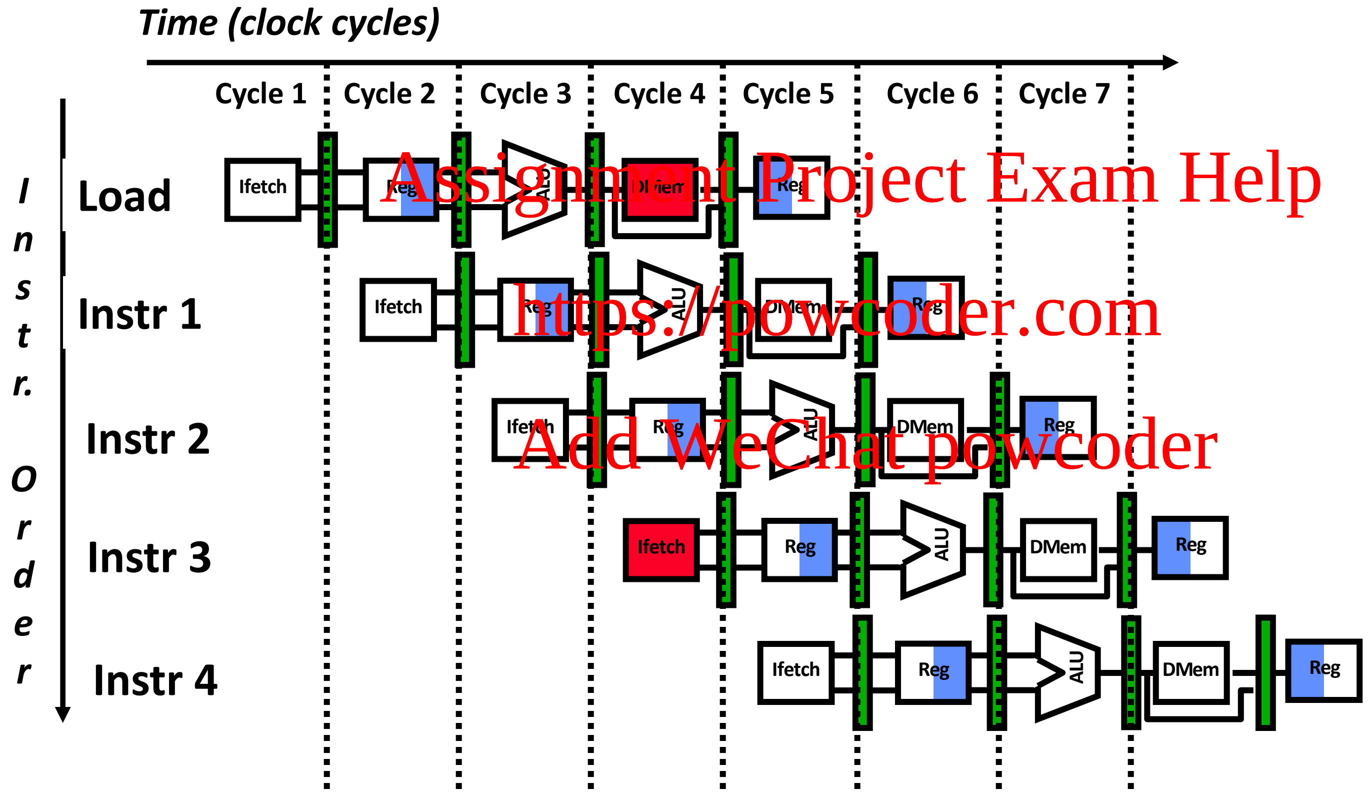
https://powcoder.com

Add WeChat powcoder

Instruction 3 caused a branch to 15

Instructions 4-7 have **stalls**

Resource conflict stalls



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Apart from branching, it is possible to have stalls because of resource conflicts

Needs careful processor pipeline design with appropriate arbitration between streams

(eg. skip the cycle 4)

Dealing with Branches

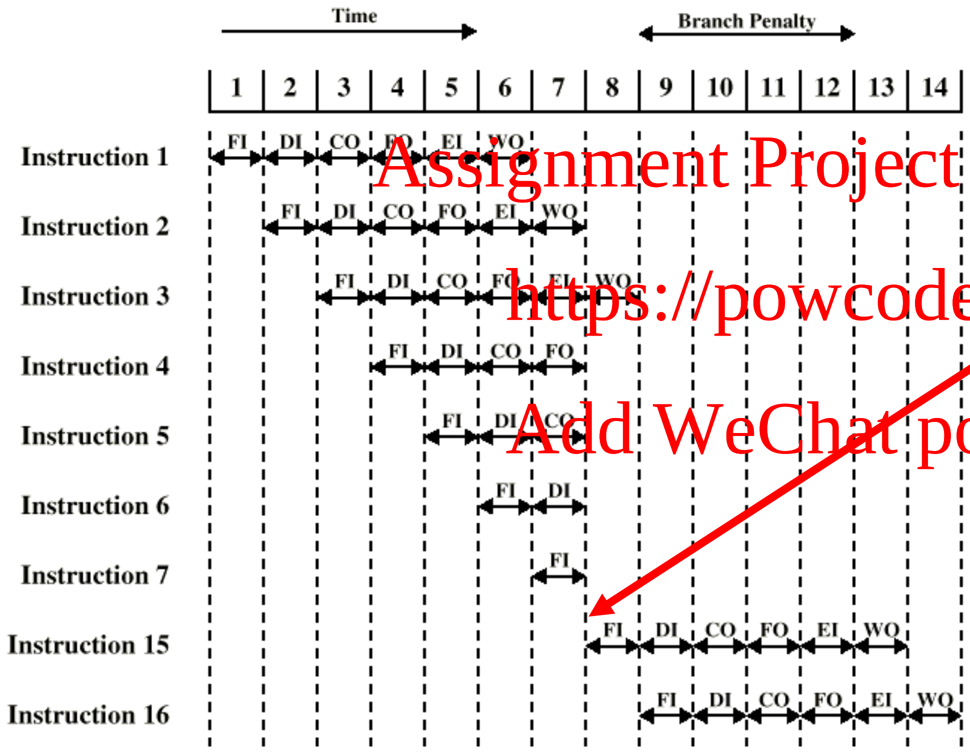
- Multiple Streams
- Prefetch Branch Target
- Loop buffer
- Branch prediction
- Delayed branching

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Prefetching branching target



Assignment Project Exam Help

Prefetch the branch instructions and store somewhere non-conflicting

https://powcoder.com

Add WeChat powcoder

Target of branch is prefetched in addition to instructions following branch

Keep target until branch is executed

Used as far back as the IBM 360/91

Loop Buffer

- Often jump targets are a loop with sequence of instructions
- Very fast memory (IRs) stores these N Instructions in sequence
- The instructions in the loop can be pipelined
- Maintained by fetch stage of pipeline
- Check buffer before fetching from memory
- Very good for small loops or jumps
- Used by CRAY-1

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Branch Prediction (1)

- Predict never taken (pessimistic)
  - Assume that jump will not happen
  - Always fetch next instruction
  - Examples: 68020 & VAX 11/780 (manufactured by DEC)
  - Do not prefetch after branch
- Predict always taken (optimistic)
  - Assume that jump will happen during fetch
  - Next fetch the branch target instruction

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

BP (1)

```
main()
{
int a,b,c[50];
b = 2;
for( a= 0; a < 50; a++)
        c[a] = a * b;
}
```

```
        mov     r3, #2
        str     r3, [fp, #-16]
        mov     r3, #0
        str     r3, [fp, #-20]
        b       .L2
.L3:
        ldr     r1, [fp, #-20]
        ldr     r2, [fp, #-20]
        ldr     r3, [fp, #-16]
        mul     r0, r3, r2
        mvn     r2, #207
        mov     r3, r1, asl #2
        sub     r1, fp, #12
        add     r3, r3, r1
        add     r3, r3, r2
        str     r0, [r3, #0]
        ldr     r3, [fp, #-20]
        add     r3, r3, #1
        str     r3, [fp, #-20]
.L2:
        ldr     r3, [fp, #-20]
        cmp     r3, #49
        ble     .L3
        sub     sp, fp, #12
        ldmfd   sp, {fp, sp, pc}
```

Predict always jump has a 49/50 success rate and predict never jump has a 1/50 success rate

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Branch Prediction (2)

- Predict by Opcode
  - Some instructions are more likely to result in a jump than others
  - For example COMPARE instructions
  - Can get up to 75% success
- Taken/Not taken switch
  - Based on previous history (machine learning aided)
  - Good for loops
- Delayed Branch
  - Do not take jump until you have to
  - Do all current in sequence until the jump instruction
  - Rearrange instructions

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Speedup from pipelining

- Ideally should equal to the number of pipelined stages (pipeline depth)
  - Without pipelining, CPI is equal to the number of stages in Data Flow;
    assuming each stage requires 1 cycle (= Ideal CPI x Pipeline depth)
  - CPI = clocks per instruction, ideally = 1

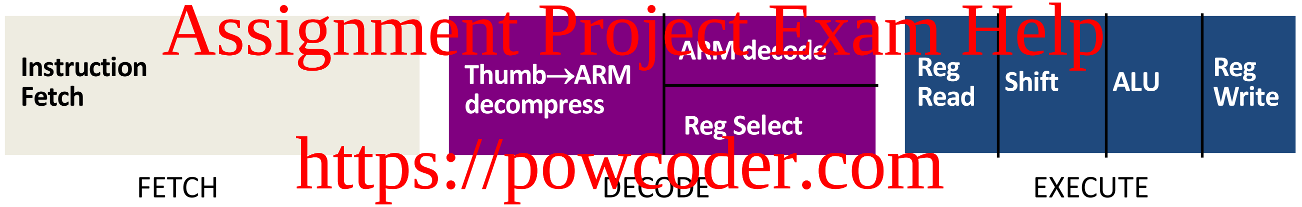
Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

$$\mathrm{CPI}_{\mathrm{pipelined}} = \text{Ideal CPI} + \text{Average Stall cycles per Inst}$$

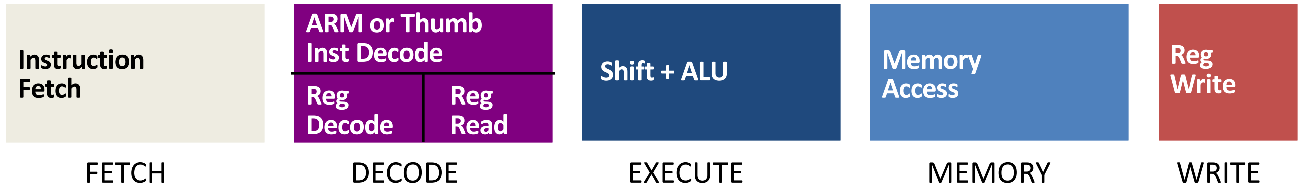
$$\text{Speedup} = \frac{\text{Ideal CPI} \times \text{Pipeline depth}}{\text{Ideal CPI} + \text{Pipeline stall CPI}} \times \frac{\text{Cycle Time}_{\mathrm{unpipelined}}}{\text{Cycle Time}_{\mathrm{pipelined}}}$$

Pipelined architecture examples

ARM7TDMI – 3 stage pipeline

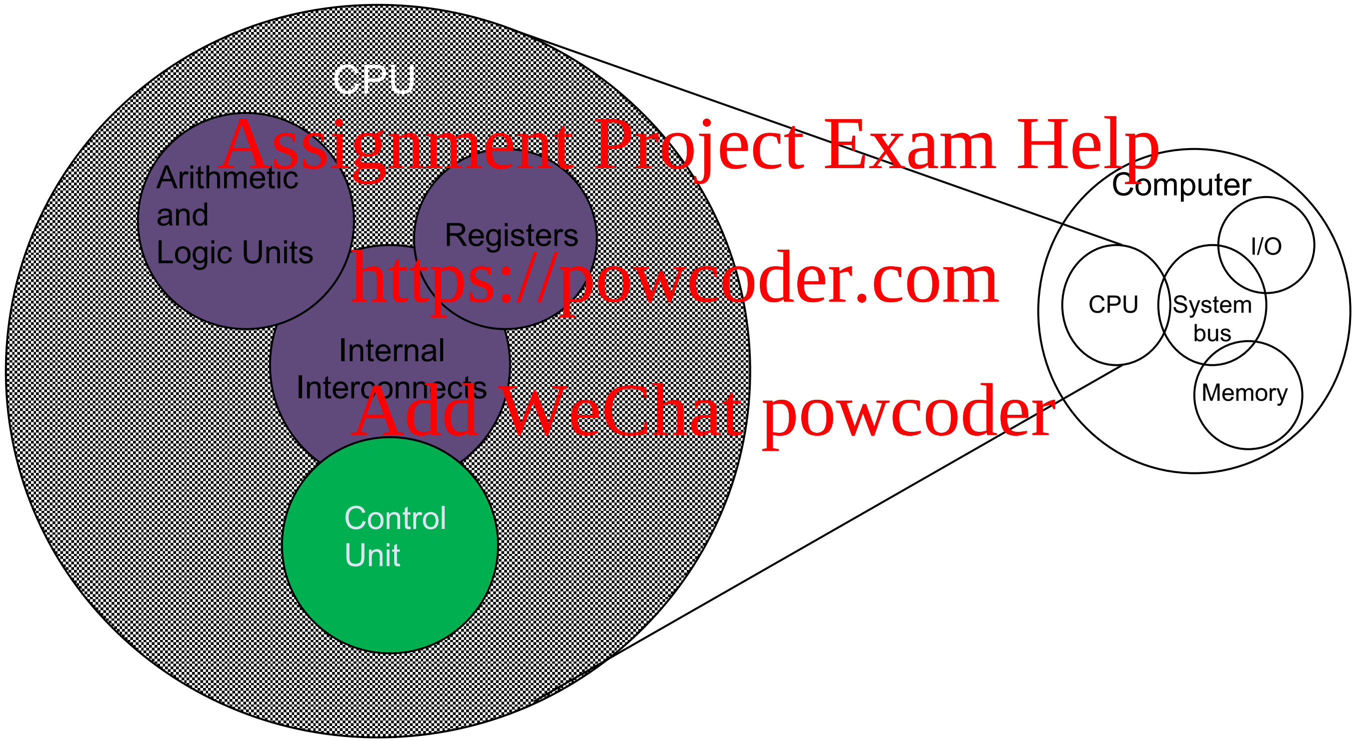


ARM9TDMI – 5 stage pipeline



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Control unit: CPU types



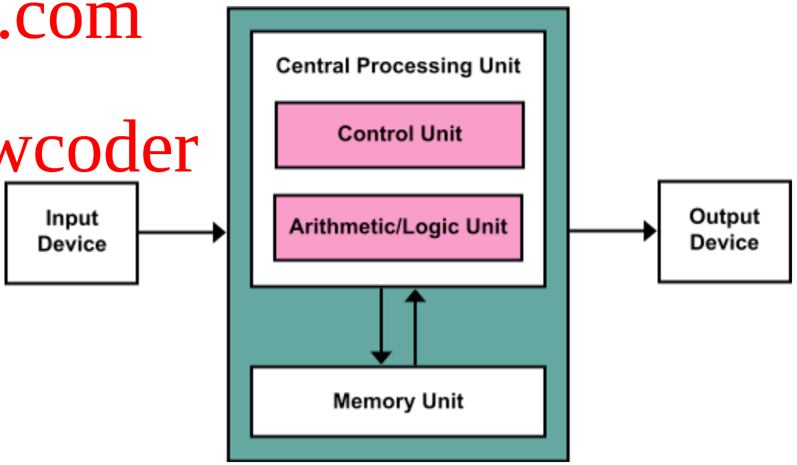
Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Von Neumann
Harvard
CISC
RISC

Von Neumann architecture

- "Princeton architecture"
- Data and instructions share the same memory and memory interface with the CPU
- Input and output may be on separate interconnects
- Usually simplified to using a single bus for all data/instructions transfer
- Most of classical and current systems belong to this to some degree

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

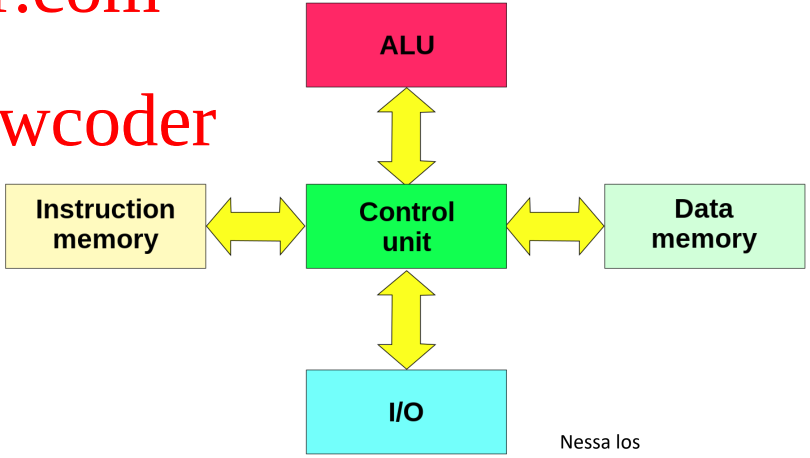


Source: Kapooht

Harvard architecture

- Separate instruction and data memories connected to the processor's control unit using separate interconnects
- I/O share the same interconnect

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

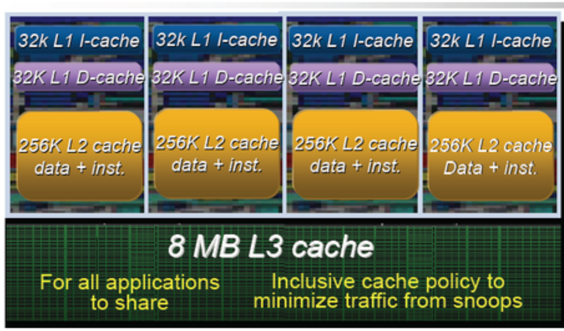


Nessa los

A bit of both

- Modified Harvard architecture
  - Or sometimes called “almost Von Neumann architecture”
  - Memories inside and close to CPU are divided into instruction and data
    - Instruction registers and data registers
    - Instruction cache and data cache (usually L1 cache)
    - Connected with separate interconnects
  - Memories further away from CPU are organized in Von Neumann fashion
  - ARM and Intel current tech processors use this
  - Review pipeline stalls when fetch clashes with data store (slide 13)

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder


32k L1 I-cache
32K L1 D-cache
256K L2 cache data + inst.
32k L1 I-cache
32K L1 D-cache
256K L2 cache data + inst.
32k L1 I-cache
32K L1 D-cache
256K L2 cache data + inst.
32k L1 I-cache
32K L1 D-cache
256K L2 cache Data + inst.
8 MB L3 cache
For all applications to share
Inclusive cache policy to minimize traffic from snoops

CISC and RISC

- CISC: complex instruction set computer
- RISC: reduced instruction set computer

Assignment Project Exam Help

- Berkeley group coined the term RISC and made a CPU called RISC 1, soon after Stanford made a similar CPU called MIPS

https://powcoder.com

- SPARC also emerged from SUN

Add WeChat powcoder

- ARM has a range of RISC architectures
- Early RISC CPUs had about 50 instructions compared to 200-300 common for CISC
  - The aim was to simplify CPU to process (and start) instructions faster

RISC philosophy

- Instructions of fixed length executing in a single clock cycle
- Pipelines to achieve one-instruction-per-one-clock-cycle throughput (need to predict branches in program flow in advance)
- Simple control logic to increase clock speed, no micro-code
- Operations performed on internal registers only; only LOAD and STORE instructions access external memory

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

MIPS example: add \$rd, \$rs, \$rt

B_{31-26}	B_{25-21}	B_{20-16}	B_{15-11}	B_{10-6}	B_{5-0}
opcode	register s	register t	register d	shift amount	function

CISC characteristics

- Binary compatibility
  - Old binary code can run on newer versions
- Complex control logic to support many instructions
- Use of micro-code
  - One program instruction can execute in many cycles
- Variable-length instructions to save program memory
- Small internal register sets compared with RISC
- Complex addressing modes, operands can reside in external memory or internal registers

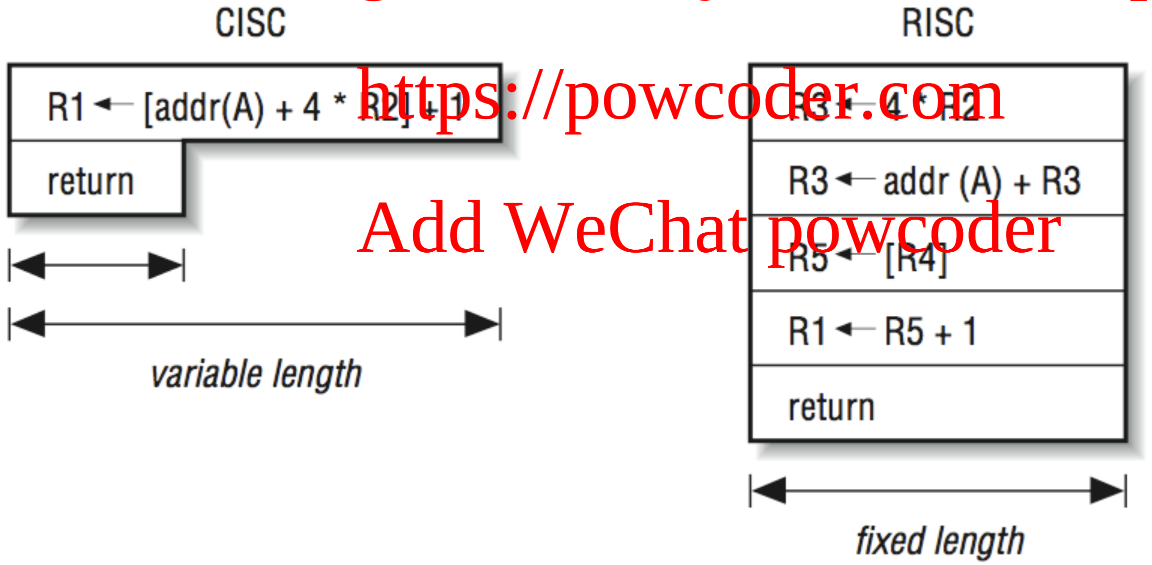
Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

A CISC versus RISC example

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder



One way of looking at it…

- Runtime = clock-period x CPI x N_{instr}
- CISC tries to reduce the number of instructions
  - Fewer instructions to do more
  - Increased CPI
  - Complex CPU design (multi-mode registers, and multi-cycle executions)
- RISC tries to reduce the clock cycles per instruction
  - less cycles-per-instr
  - more instructions
  - simpler CPU design
- Obvious trade-offs can be seen!

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Another way of looking at it

- CISC assembler code may be easier for human programmers to handle
  - When manually coding
- But is this advantage really relevant these days?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder