Instruction Set Architecture

Instructor: Saraju P. Mohanty

PART2

- Instruction Set Architectures
- Data Transfer Instructions
- Data Manipulation Instructions
- Program Control Instructions
- Program Interrupt

Sources

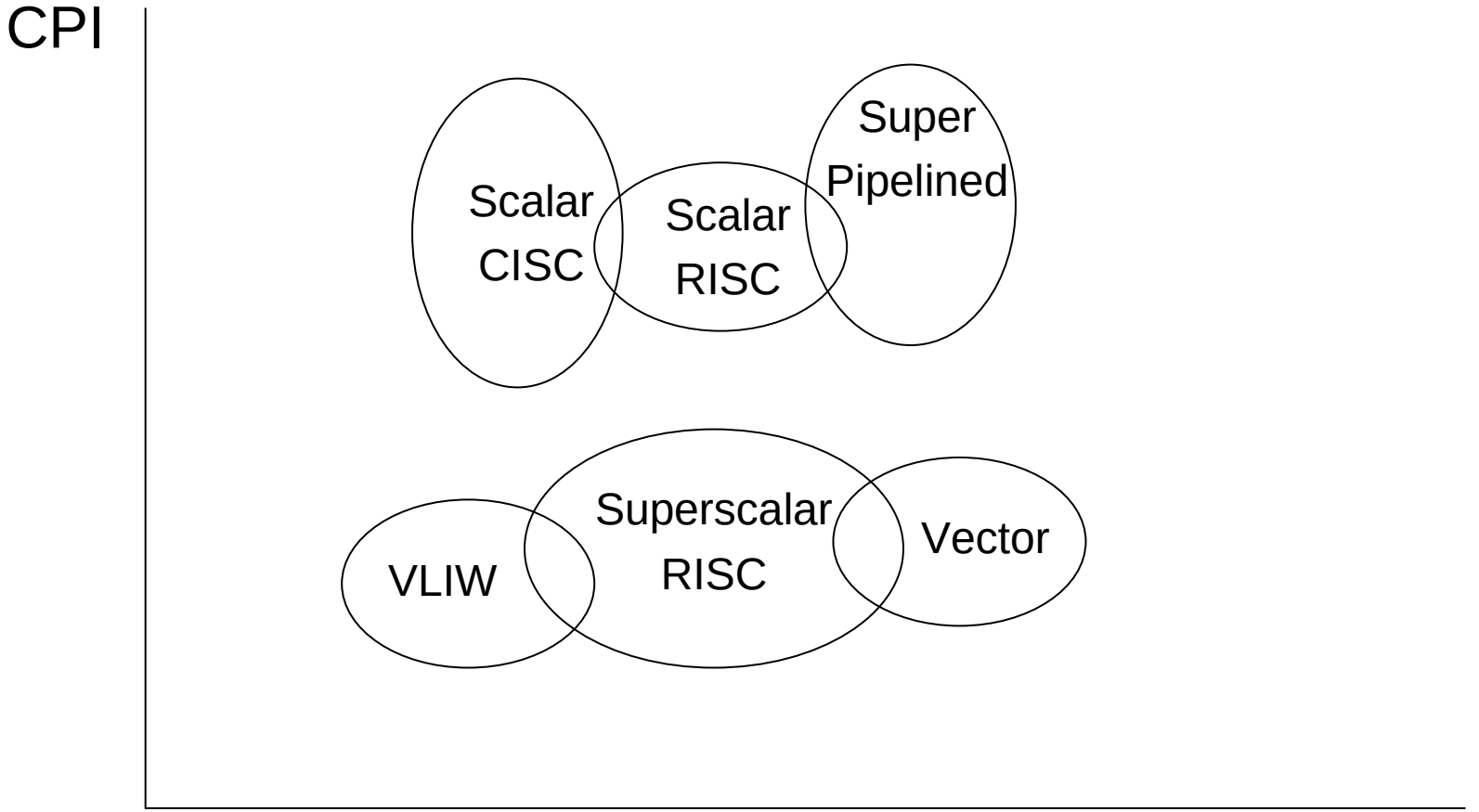
- Logic and Computer Design Fundamentals by M. M. Mano and C. R. Kime.
- Computer Organization and Design: The Hardware/Software Interface by David A. Patterson and John L. Hennessy.
- Dr. Valavanis lectures

Few Processor Families

Various processor families are:

- CISC (Complex Instruction Set Computer)
- RISC (Reduced Instruction Set Computer)
- Superscalar (Issues multiple instructions in one clock cycle)
- Superpipelined processor
- VLIW (Very Long Instruction Width)
- Vector / Parallel Processors
- Symbolic Processor

Design Space of Modern Processor Families



NOTE: CPI is cycle per instruction

Instruction Set Architectures

CISC architecture	RISC architecture
Large and Complex instruction set.	Small and Simple instruction set.
Small number of general purpose registers (GPRs) (8 to 24 in number).	Large number of general purpose registers (32 to 192).
Addressing modes are large in number (12 to 24).	Addressing modes are limited in number (3 to 5).
Instruction formats are of different lengths.	Instruction formats are all of the same length.
Instructions perform both elementary operations and complex operations.	Instructions perform elementary operations.
Memory access is directly available to most types of instructions.	Memory accesses are restricted to load or store.
Unified or Split Cache	Split Cache
Low Clock Rate	High Clock Rate
Microcoded CPU control	Hardwired CPU control

Elementary Computer Instructions

- Actual instruction set architectures range between those which are purely RISC and those which are purely CISC.
- There is a basic set of instructions that most of the computers include, irrespective RISC or CISC.
- Three major categories of elementary instructions:
  - Data transfer instructions
  - Data manipulation instructions
  - Program control instructions

Data Transfer Instructions

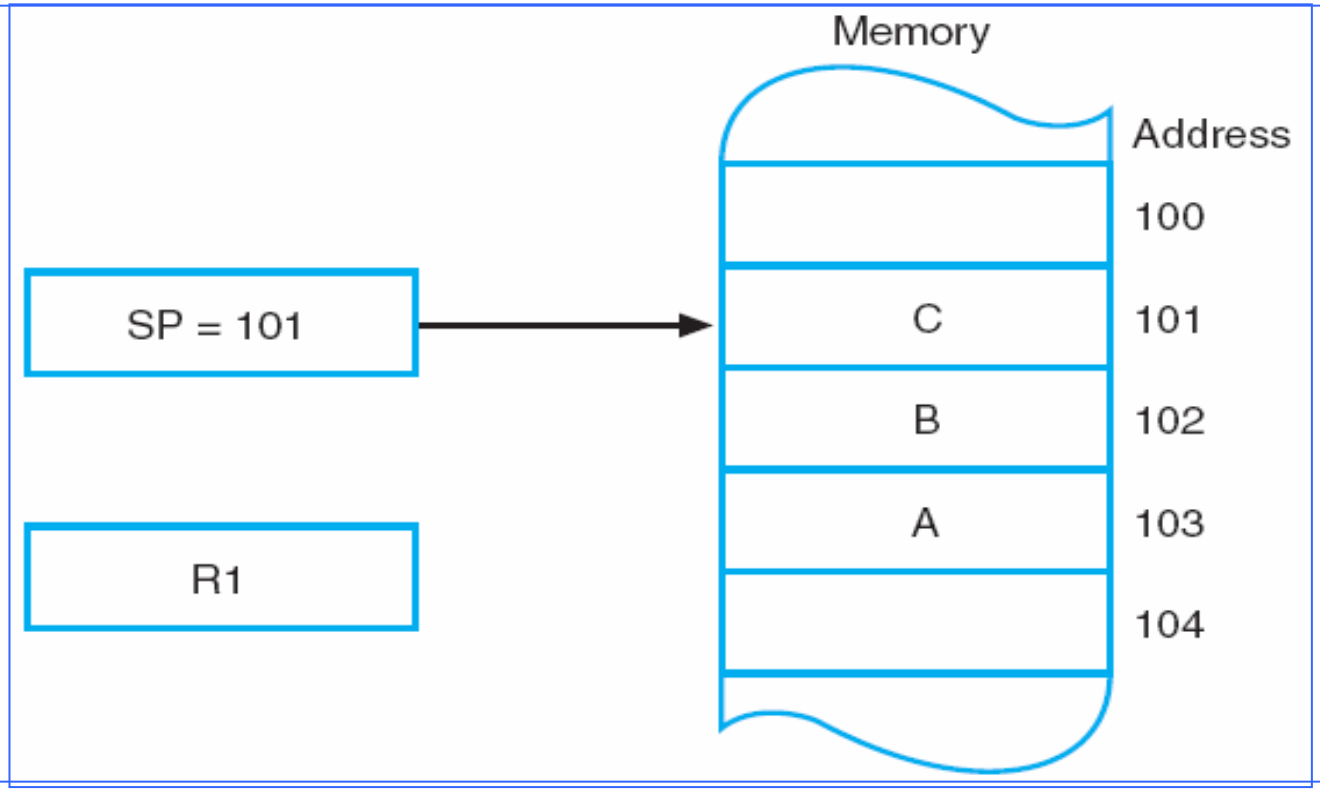
- Data Transfer instructions move data from one place in the computer to another without changing the content of the data.
- Mostly data transfers are between memory and processor registers, between processor registers and input and output registers and among the processor registers.

Name	Mnemonic
Load	LD
Store	ST
Move	MOVE
Exchange	XCH
Push	PUSH
Pop	POP
Input	IN
Output	OUT

Table 9-2 Typical Data Transfer Instructions

Data Transfer Instructions: Stack Instructions

- The stack instructions push and pop transfer data between a memory stack and a processor register or memory.
- The push operation places a new item on the top of the stack, while the pop operation removes one item fro the stack so that the stack pops up.



Stored in stack A, B, C. Operations like: SP←SP-1 M[SP]←R1 R1←M[SP] SP ← SP+1

Data Transfer Instructions: I/O Instructions

- I/O instructions transfer data between processor registers and input and output devices.
- Similar to load/store instructions transfer is between CPU register and external register, unlike load/store which is between CPU register and memory.
- Computers have several input/output ports dedicated for communication.
- A port is typically a register with input output lines.
- A particular port is chosen by an address, similar to the way an address selects a word in memory.
- I/O instructions include an address field in their format for specifying a particular port.
- Port addresses are specified in two ways:
  - Independent I/O: address ranges are assigned to memory and I/O ports are independent of each other.
  - Memory-mapped I/O: a sub-range of memory address is used for addressing I/O ports.

Data Manipulation Instructions

- Data manipulation instructions perform operations on data and provide the computational capabilities of the computer.
- **Three basic types**:
  - Arithmetic instructions
  - Logical and bit manipulation instructions
  - Shift instructions
- An instruction is typically processed by executing a sequence of one or more microinstructions.
- A microinstruction is an elementary operation executed by the hardware of the computer under the control of the control unit.
- An instruction may involve several elementary operations that fetch the instruction, bring the operands from appropriate processor registers, and store the result in the specified location.

Data Manipulation Instructions: Arithmetic

Four basic arithmetic instructions are addition, subtraction, multiplication and division.

Name	Mnemonic
Increment	INC
Decrement	DEC
Add	ADD
Subtract	SUB
Multiply	MUL
Divide	DIV
Add with carry	ADDC
Subtract with borrow	SUBB
Subtract reverse	SUBR
Negate	NEG

If only the addition and subtraction operations are provided by the computer, multiplication and division must be carried out by means of programs.

Data Manipulation Instructions: Logical

Logical instructions perform binary operations on words stored in registers or memory words.

Useful for manipulating individual bits or a group of bits that represent binary-coded information.

Name	Mnemonic
Clear	CLR
Set	SET
Complement	NOT
AND	AND
OR	OR
Exclusive-OR	XOR
Clear carry	CLRC
Set carry	SETC
Complement carry	COMC

Data Manipulation Instructions: Shift

Instructions to shift the content of an operand are provided in several varieties.

The table shows different kinds of shift operations provided.

Name	Mnemonic
Logical shift right	SHR
Logical shift left	SHL
Arithmetic shift right	SHRA
Arithmetic shift left	SHLA
Rotate right	ROR
Rotate left	ROL
Rotate right with carry	RORC
Rotate left with carry	ROLC

Program Control instructions

- A control instruction is one, which, when executed, may change the address value in the PC and cause the flow of control to be altered.
- They provide control over the flow of program execution and a capability of branching to different program segments, depending on the previous computations.

Typical Program Control Instructions

Name	Mnemonic
Branch	BR
Jump	JMP
Skip next instruction	SKP
Call procedure	CALL
Return from procedure	RET
Compare (by subtraction)	CMP
Test (by ANDing)	TEST

Program Control instructions: Conditional branch

- A branch instruction that may or may not cause transfer of control, depending on the value of stored bits in the PSR.
- Test for a different combination of status bits for a condition – if true, control is transferred to a new address, else, the program continues with the next instruction.

Instructions Relating to Status bits in the PSR

Branch condition	Mnemonic	Test condition
Branch if zero	BZ	$Z = 1$
Branch if not zero	BNZ	$Z = 0$
Branch if carry	BC	$C = 1$
Branch if no carry	BNC	$C = 0$
Branch if minus	BN	$N = 1$
Branch if plus	BNN	$N = 0$
Branch if overflow	BV	$V = 1$
Branch if no overflow	BNV	$V = 0$

Program Control instructions: Conditional branch

Instructions Relating for Unsigned Numbers

Branch condition	Mnemonic	Condition	Status bits*
Branch if higher	BH	$A > B$	$C + Z = 0$
Branch if higher or equal	BHE	$A \geq B$	$C = 0$
Branch if lower	BL	$A < B$	$C = 1$
Branch if lower or equal	BLE	$A \leq B$	$C + Z = 1$
Branch if equal	BE	$A = B$	$Z = 1$
Branch if not equal	BNE	$A \neq B$	$Z = 0$

*Note that C here is a borrow bit.

Instructions Relating for Signed Numbers

Branch condition	Mnemonic	Condition	Status bits
Branch if greater	BG	$A > B$	$(N \oplus V) + Z = 0$
Branch if greater or equal	BGE	$A \geq B$	$N \oplus V = 0$
Branch if less	BL	$A < B$	$N \oplus V = 1$
Branch if less or equal	BLE	$A \leq B$	$(N \oplus V) + Z = 1$

Program Control instructions: Procedure call and return

- A procedure is a self-contained sequence of instructions that performs a given computational task. Also known as subroutine.
- During execution of a program, a procedure may be called several times to perform its function through branching to the beginning of the procedure.
- The instruction that transfers control to a procedure is known as:
  - Call procedure or
  - Call subroutine or
  - Jump to subroutine or
  - Branch to subroutine or
  - Branch and link
- The instruction has one field and it performs to operations:
  - Stores the value of the PC in a temporary location
  - The address of the first instruction in the procedure is loaded into PC
- The final instruction in every procedure must be a return.

Program Interrupt

- Used to handle a variety of situations that require a departure from the normal program sequence.
- Transfers control from a program that is currently running to another service program as a result of an externally- or internally- generated request.
- The interrupt is usually initiated at an unpredictable point in the program by an external or internal signal.
- The address of the service program that processes the interrupt is determined by a hardware procedure.
- In response to a program interrupt, it is necessary to store information that defines all or part of the contents of the register set.

Program Interrupt: Types of Interrupts

The three major types of interrupts that cause a break in the normal execution of a program are as follows:

- External Interrupts: come from input or output devices, from timing devices, from a circuit monitoring the power supply, or from any other external source.
- Internal interrupts: come from the illegal or erroneous use of an instruction or data. These are also called as traps.

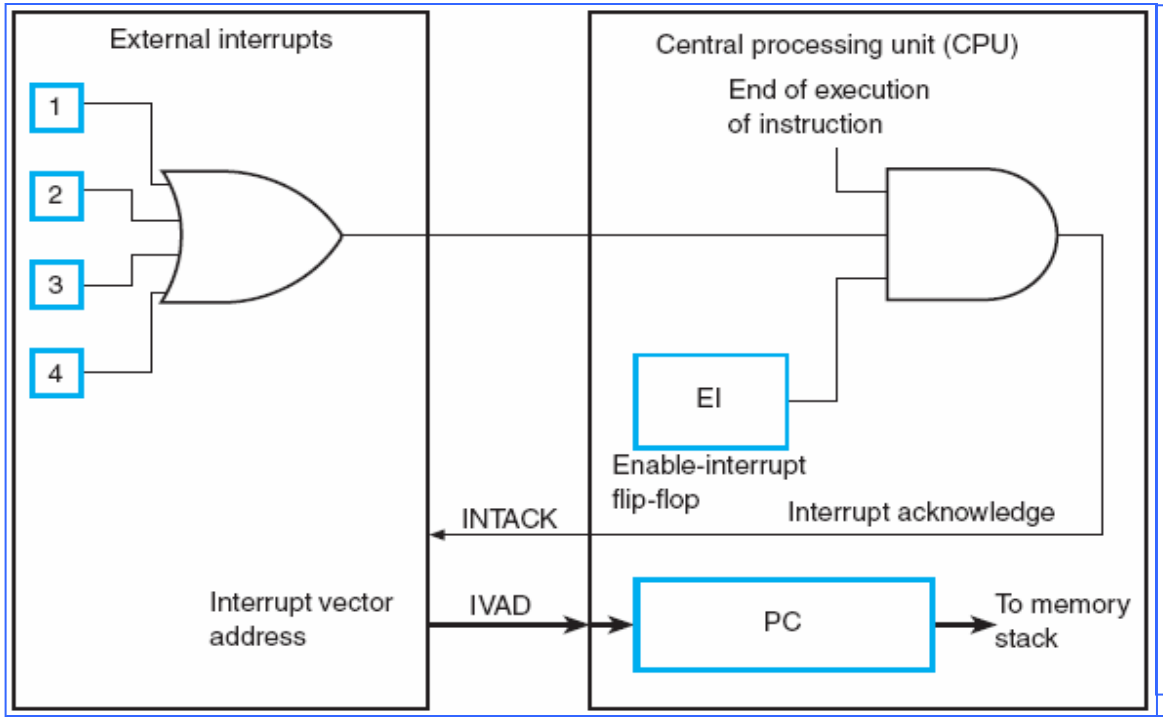
  e.g.:- arithmetic overflow, attempt to divide by zero.
- Software interrupts: initiated by executing an instruction. Can be used by the programmer to initiate an interrupt procedure at any desired point in the program.

NOTE: Interrupts are also known as exceptions.

Program Interrupt: Processing External Interrupts

- External interrupts may have single or multiple interrupt lines.
- In case of multiple interrupt sources, the sources are ORed to form a common line.
- An interrupt may be originated at any time during program execution.
- The computer usually acknowledges the interrupt only after the execution of the current instruction is completed (only if the state of the processor warrants it) to ensure no loss of information.

Program Interrupt: Processing External Interrupts



- Example of simplified external configuration
- Four interrupts are ORed
- EI flip-flop with programming instructions, enable interrupt (ENI) and disable interrupt (DSI).
- EI = 0, interrupt signal is neglected.
- EI =1, the CPU acknowledges interrupts by enabling INTACK.

- The interrupt source responds to INTACK by providing the interrupt vector address IVAD to the CPU.
- The program controlled EI allows the programmer to decide whether to use interrupt facility.
- A DSI instruction in a program means programmer does not want the program to be interrupted.
- A ENI instruction in a program means interrupt facility will be active while program is running.