

Computer Architecture

Instruction Sets: Characteristics and Functions

ARASH HABIBI LASHKARI
(April- 2010)

What is an Instruction Set?

- The complete collection of instructions that are understood by a CPU
- Machine Code
- Binary
- Usually represented by assembly codes

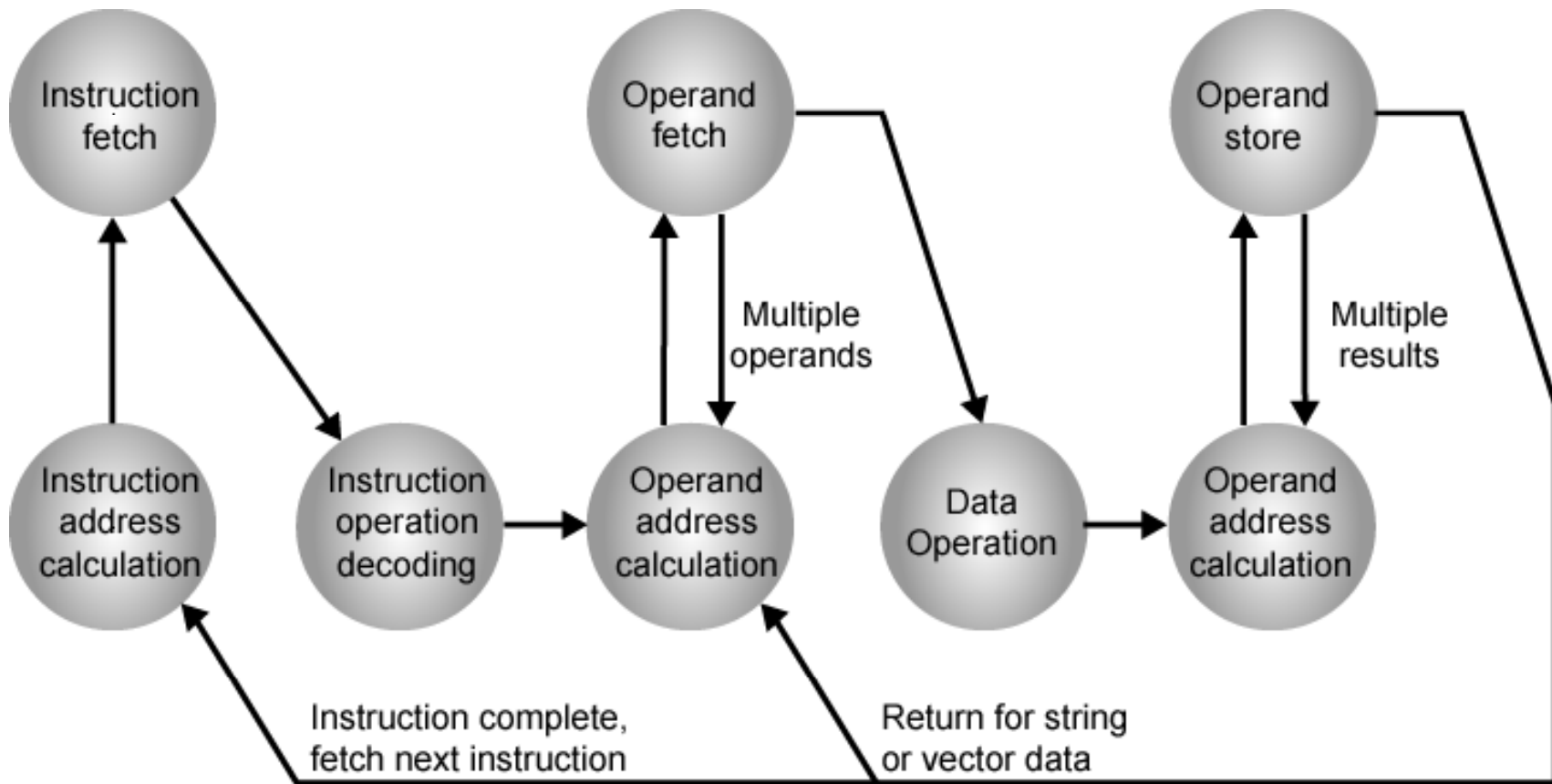
Elements of an Instruction

- Operation code (Op code)
 - Do this
- Source Operand reference
 - To this
- Result Operand reference
 - Put the answer here
- Next Instruction Reference
 - When you have done that, do this...

Where have all the Operands Gone?

- Long time passing....
- (If you don't understand, you're too young!)
- Main memory (or virtual memory or cache)
- CPU register
- I/O device

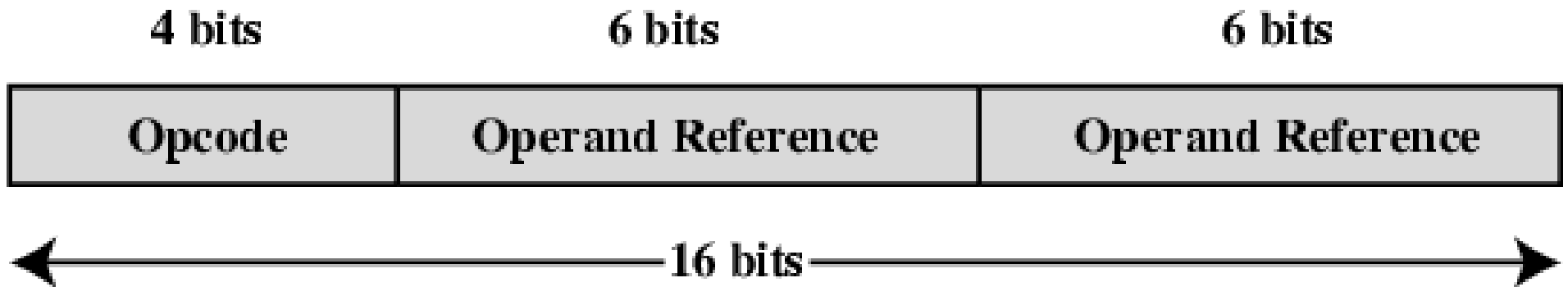
Instruction Cycle State Diagram



Instruction Representation

- In machine code each instruction has a unique bit pattern
- For human consumption (well, programmers anyway) a symbolic representation is used
 - e.g. ADD, SUB, LOAD
- Operands can also be represented in this way
 - ADD A,B

Simple Instruction Format



Instruction Types

- Data processing
- Data storage (main memory)
- Data movement (I/O)
- Program flow control

Number of Addresses (a)

- 3 addresses
 - Operand 1, Operand 2, Result
 - $a = b + c$;
 - May be a forth - next instruction (usually implicit)
 - Not common
 - Needs very long words to hold everything

Number of Addresses (b)

- 2 addresses
 - One address doubles as operand and result
 - $a = a + b$
 - Reduces length of instruction
 - Requires some extra work
 - Temporary storage to hold some results

Number of Addresses (c)

- 1 address
 - Implicit second address
 - Usually a register (accumulator)
 - Common on early machines

Number of Addresses (d)

- 0 (zero) addresses
 - All addresses implicit
 - Uses a stack
 - e.g. push a
 - push b
 - add
 - pop c

 - $c = a + b$

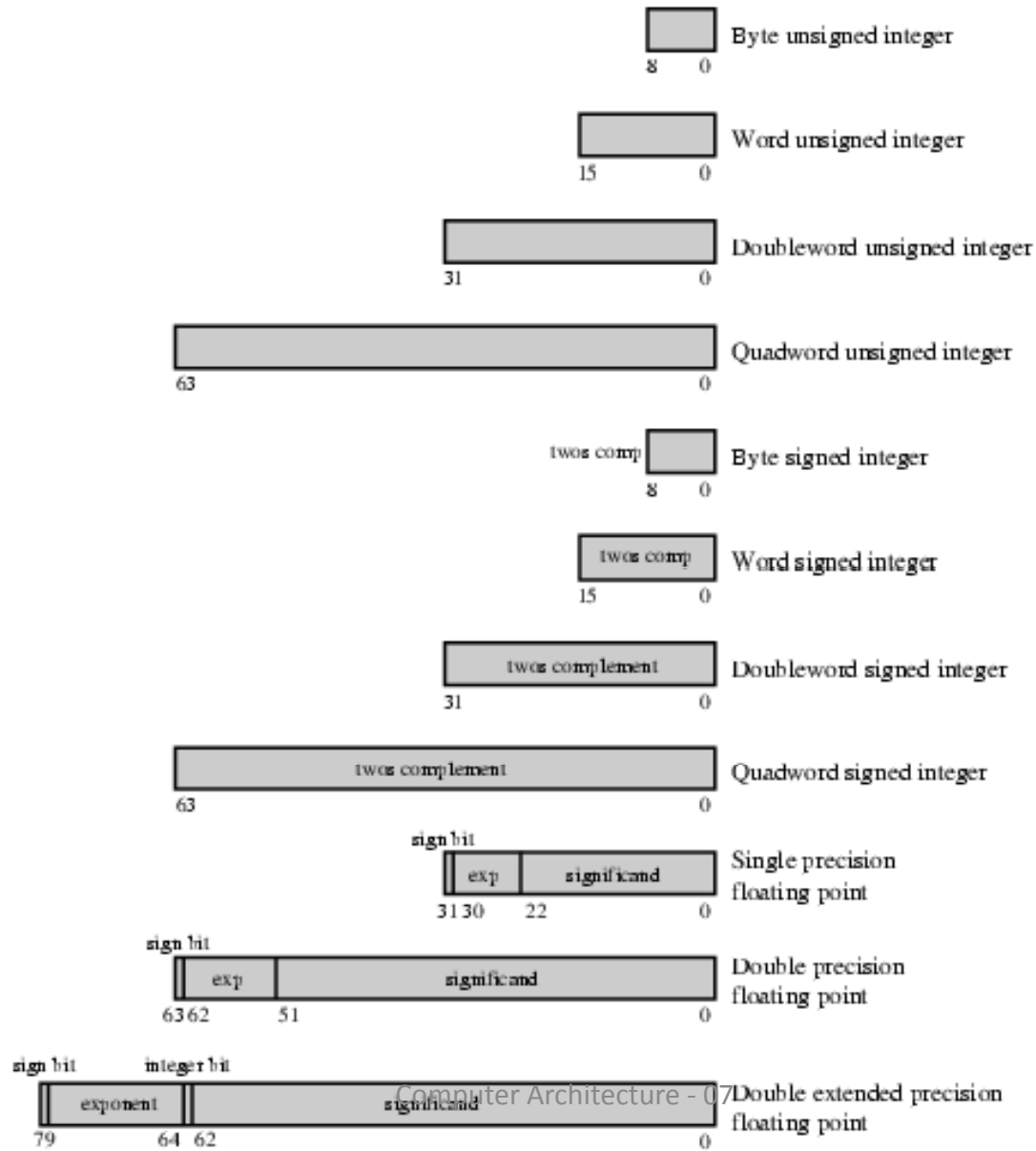
Types of Operand

- Addresses
- Numbers
 - Integer/floating point
- Characters
 - ASCII etc.
- Logical Data
 - Bits or flags
- (Aside: Is there any difference between numbers and characters? Ask a C programmer!)

Pentium Data Types

- 8 bit Byte
- 16 bit word
- 32 bit double word
- 64 bit quad word
- Addressing is by 8 bit unit
- A 32 bit double word is read at addresses divisible by 4

Pentium Numeric Data Formats



Types of Operation

- Data Transfer
- Arithmetic
- Logical
- Conversion
- I/O
- System Control
- Transfer of Control

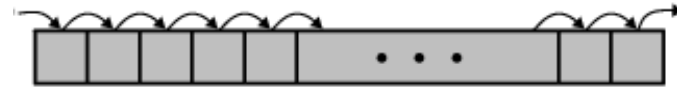
Data Transfer

- Specify
 - Source
 - Destination
 - Amount of data
- May be different instructions for different movements
 - e.g. IBM 370
- Or one instruction and different addresses
 - e.g. VAX

Arithmetic

- Add, Subtract, Multiply, Divide
- Signed Integer
- Floating point ?
- May include
 - Increment ($a++$)
 - Decrement ($a--$)
 - Negate ($-a$)

Shift and Rotate Operations



(a) Logical right shift



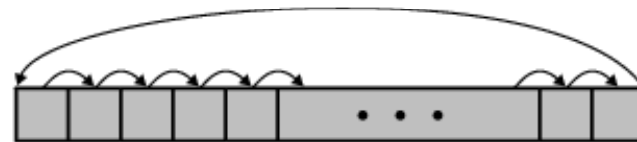
(b) Logical left shift



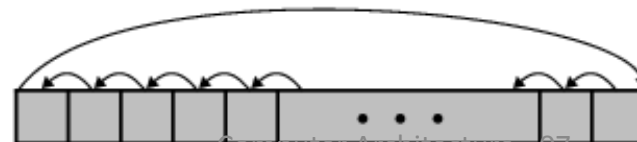
(c) Arithmetic right shift



(d) Arithmetic left shift



(e) Right rotate



(f) Left rotate

Logical

- Bitwise operations
- AND, OR, NOT

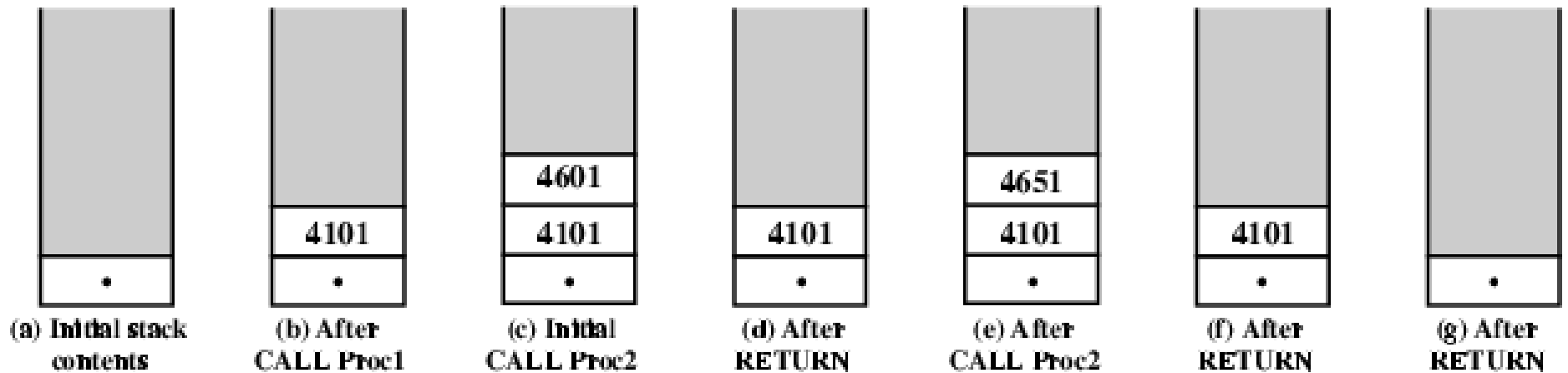
Conversion

- E.g. Binary to Decimal

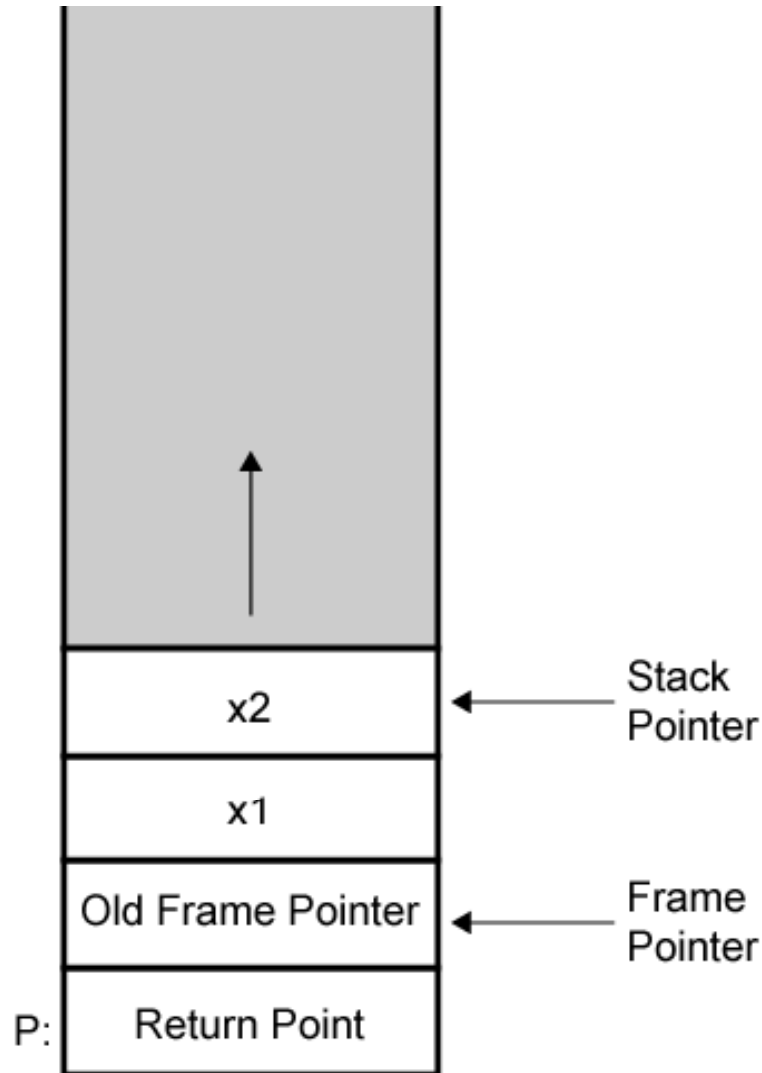
Input/Output

- May be specific instructions
- May be done using data movement instructions (memory mapped)
- May be done by a separate controller (DMA)

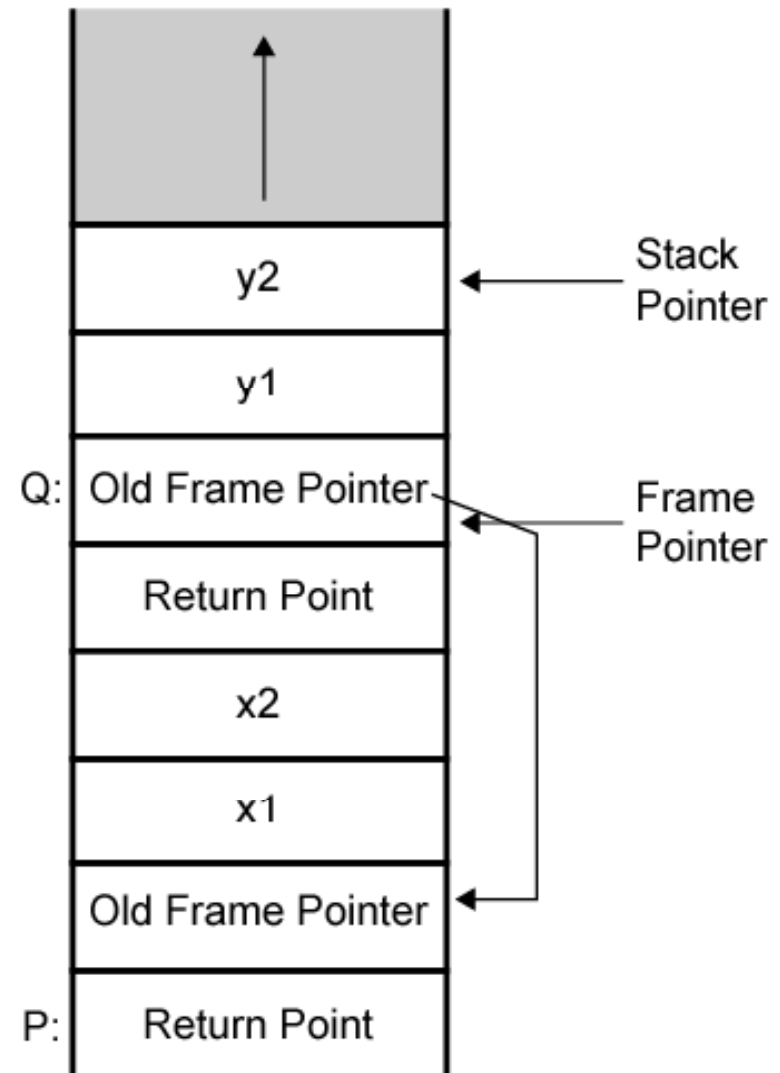
Use of Stack



Stack Frame Growth Using Sample Procedures P and Q



(a) P is active



(b) P has called Q

Questions

