Programmable System Design

Virendra Singh

Professor

Computer Architecture and Dependable Systems Lab

Department of Electrical Engineering

Indian Institute of Technology Bombay

http://www.ee.iitb.ac.in/~viren/

E-mail: viren@ee.iitb.ac.in

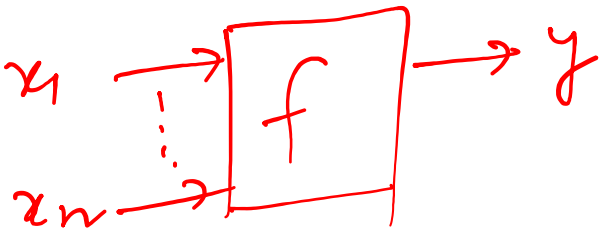
CS-226: Digital Logic Design

Lecture 28: 12 April 2021

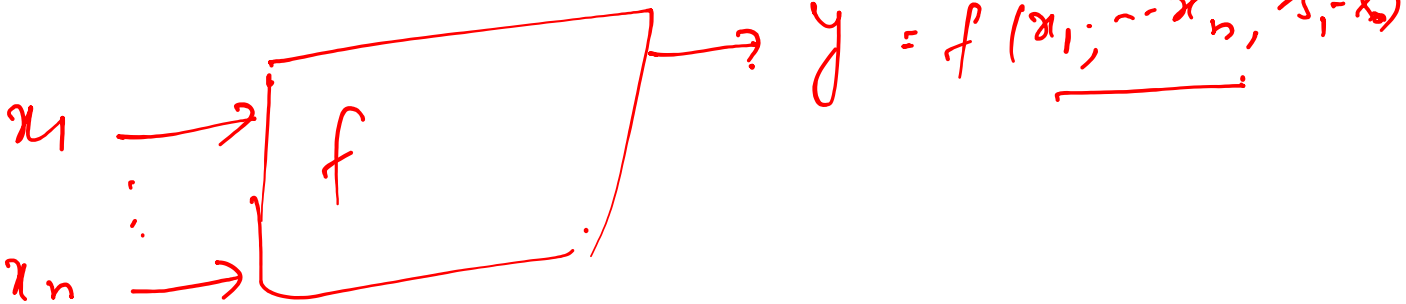
CADSL

Finite State Machine

✓ Combinational



✓ Sequential



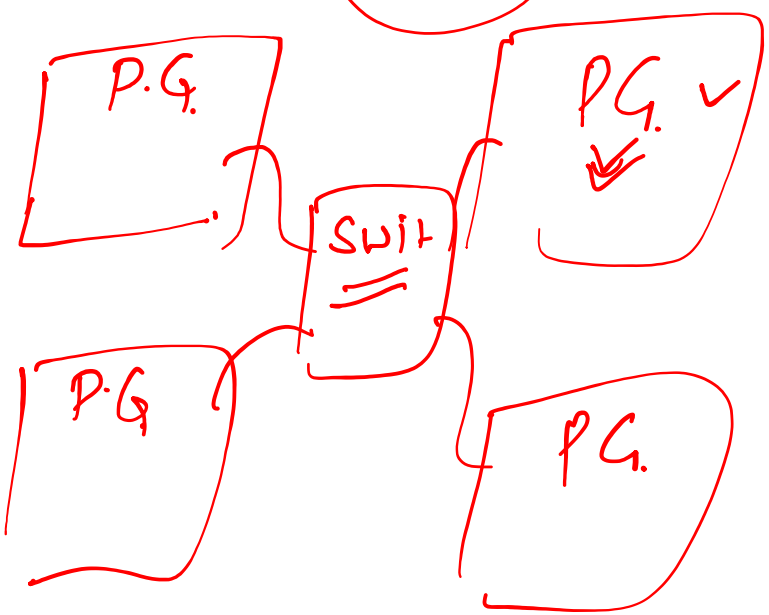
temporal behaviour

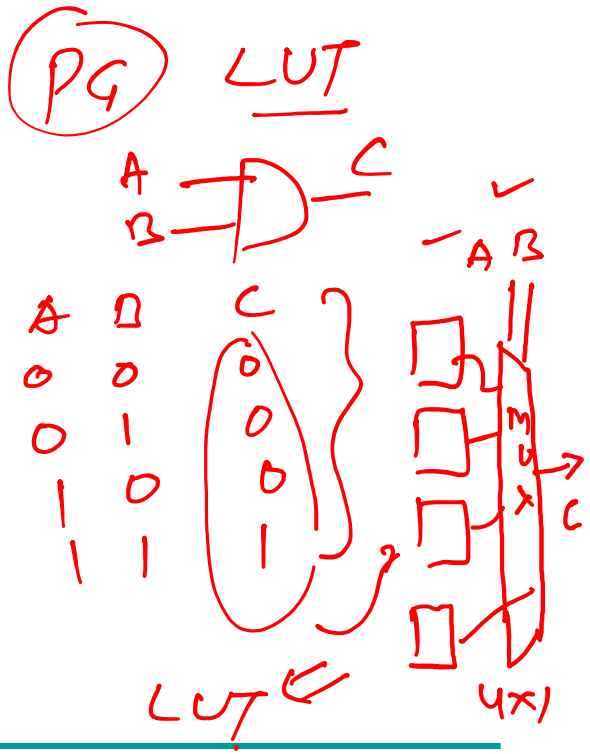
$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow \cdots \rightarrow S_n$

Flexible System

Implement using flexible fabric

FPGA (Field Programmable Gate Array)





5 input LUT & 6 input LUT.

$2^5 = 32 \rightarrow 2^{2^5} = 2^{32}$ $2^6 = 64$

$y_1 = x_1x_2 + x_3x_4 + x_4x_4$

$y_2 = x_1 + x_2x_4$

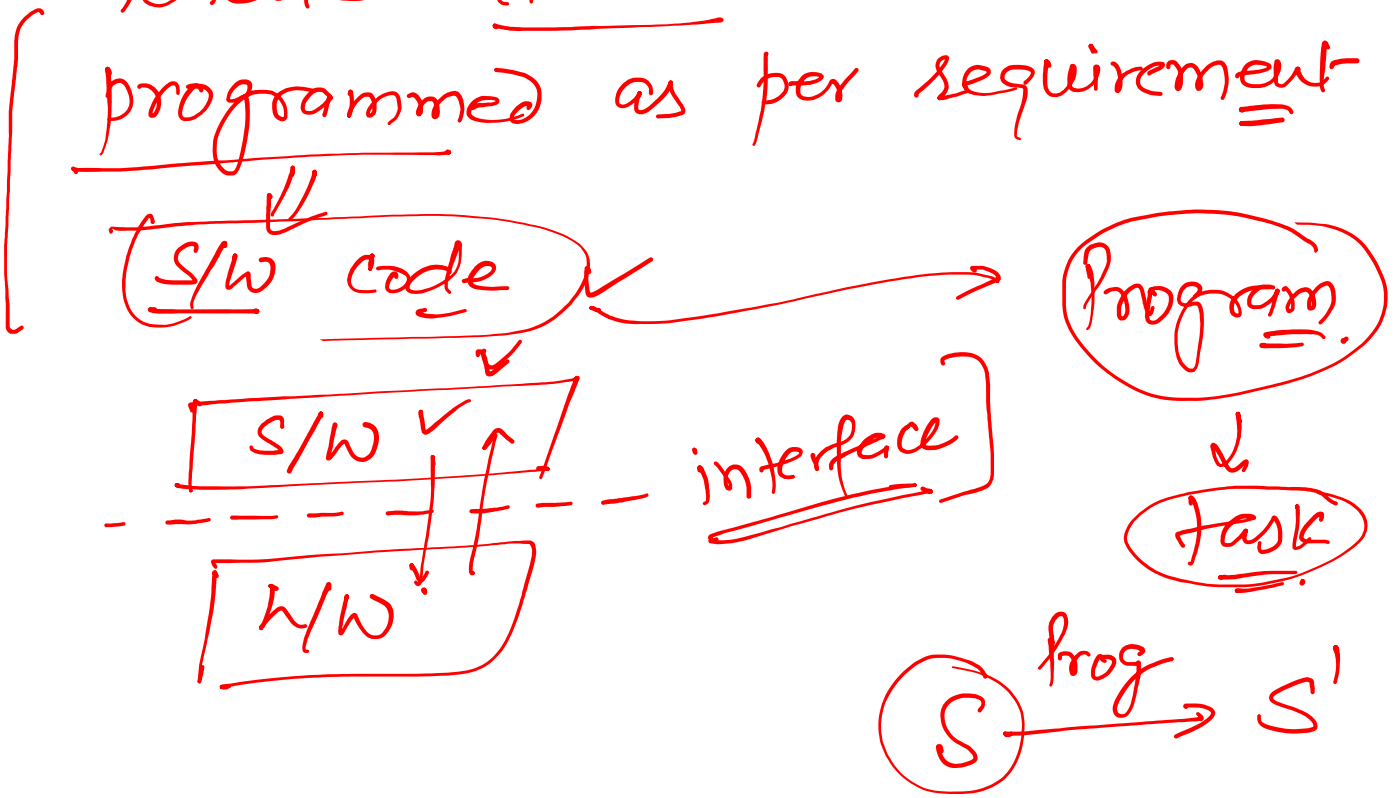
$y_3 = x_1x_3 + x_3x_4 + x_1$

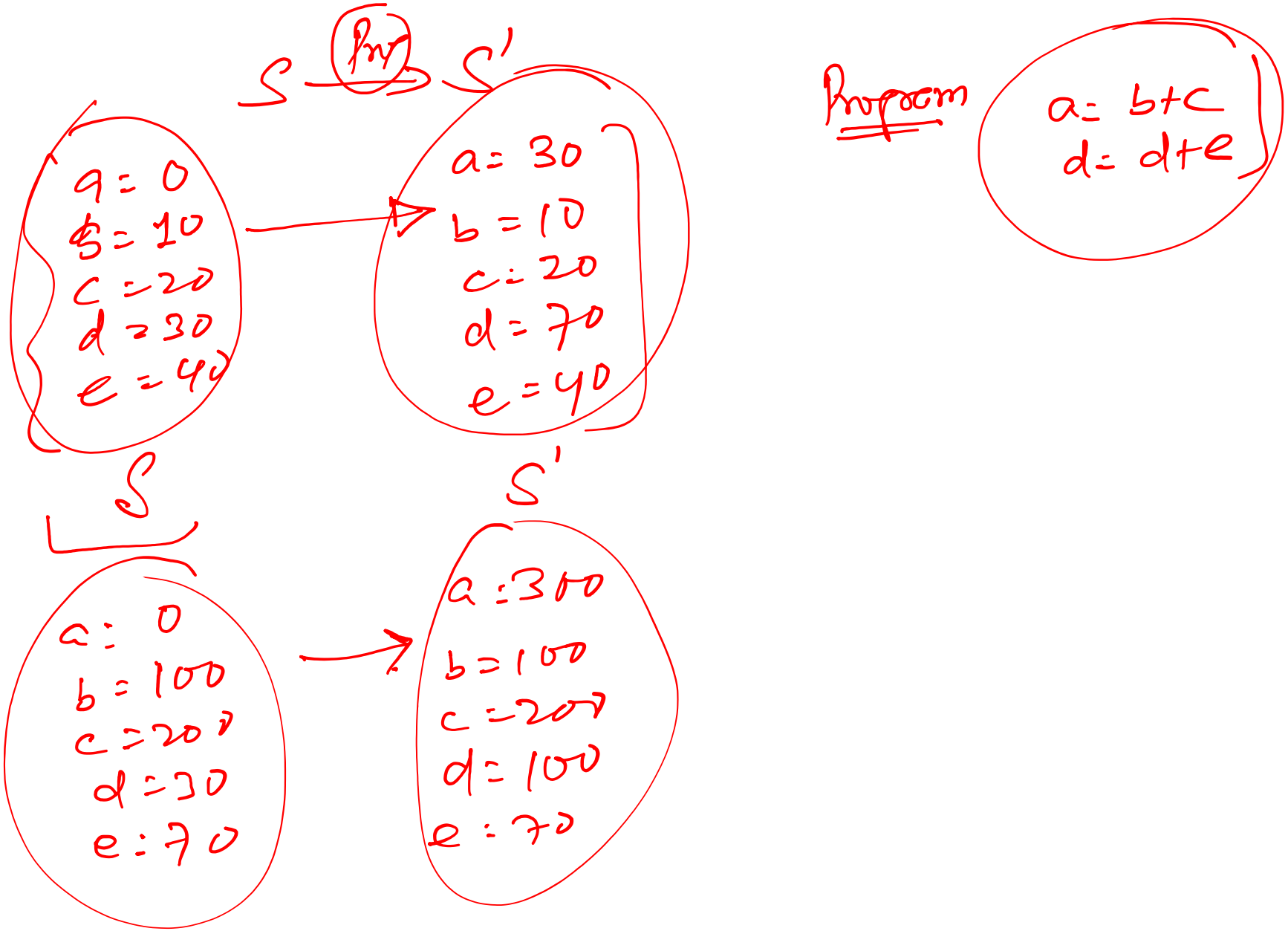
Redesign circuit → Map on FPGA.

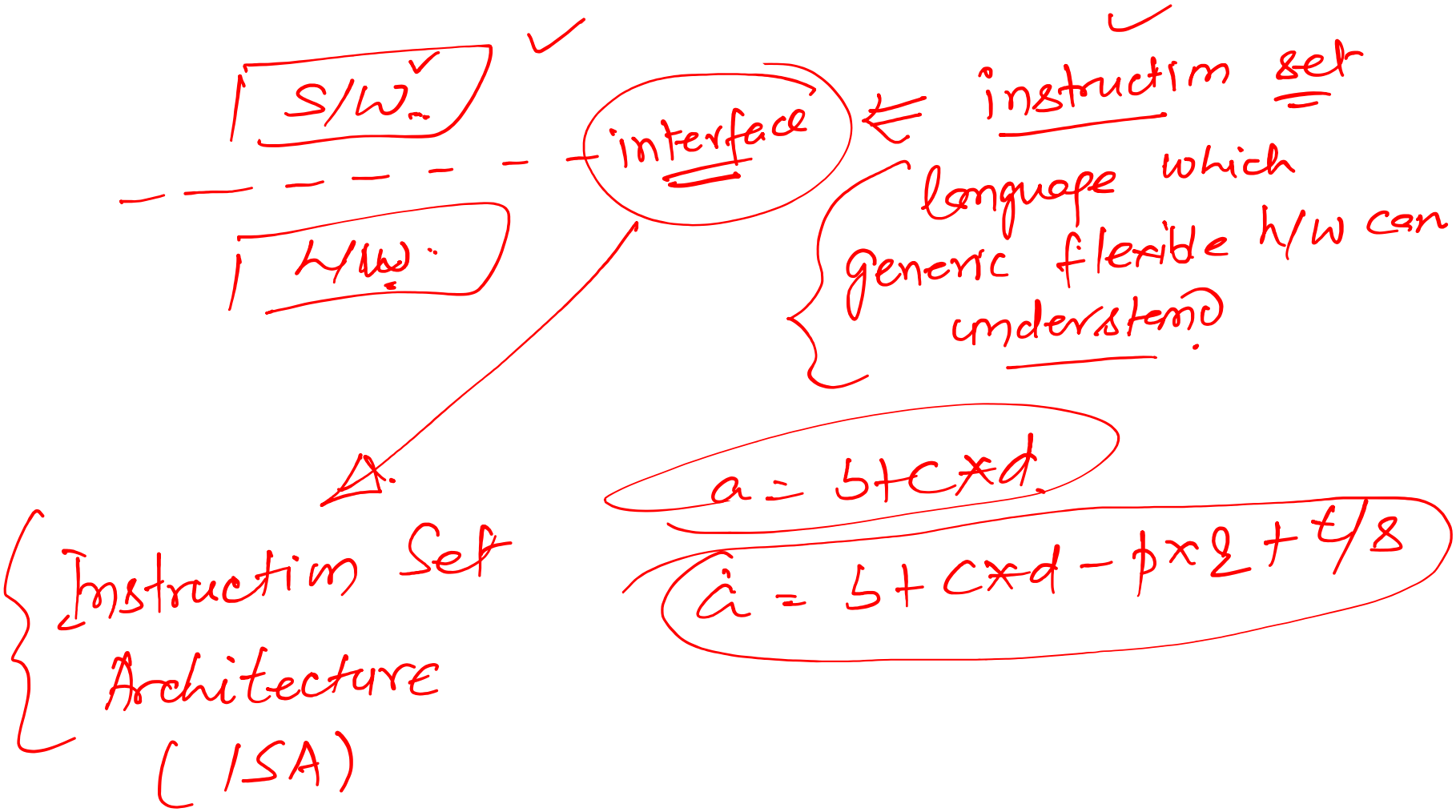
map logic on LUTs & interconnect

Programmable Fabric

Generic hardware which can be programmed as per requirement




S
Pw
S'
a = 0
b = 10
c = 20
d = 30
e = 40
a = 30
b = 10
c = 20
d = 70
e = 40
Pwprogram
a = b+c
d = d+e
S
S'
a : 0
b = 100
c = 20
d = 30
e : 70
a : 300
b = 100
c = 20
d = 100
e : 70

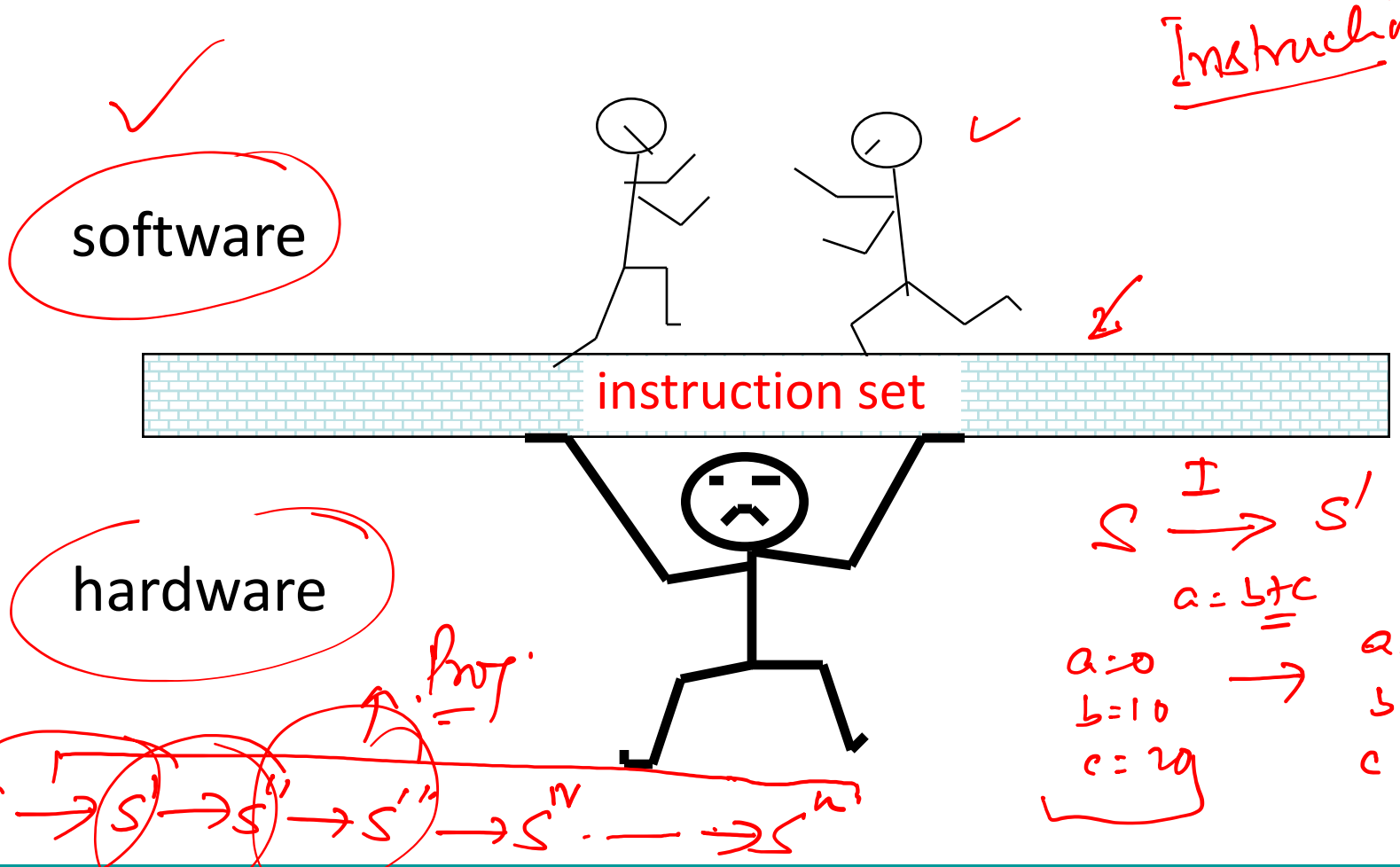

S/W
H/W
interface
instruction set
language which
generic flexible h/w can
understand
Instruction Set
Architecture
(ISA)
a = b+c*d
a = b+c*d-p*q+t/s

Instruction Set Architecture

- Instruction set architecture is the structure of a computer that a machine language programmer must understand to write a correct (timing independent) program for that machine.

- The instruction set architecture is also the machine description that a hardware designer must understand to design a correct implementation of the computer.

Instruction Set Architecture (ISA)



Instruction

- C Statement

  f = (g+h) – (i+j)

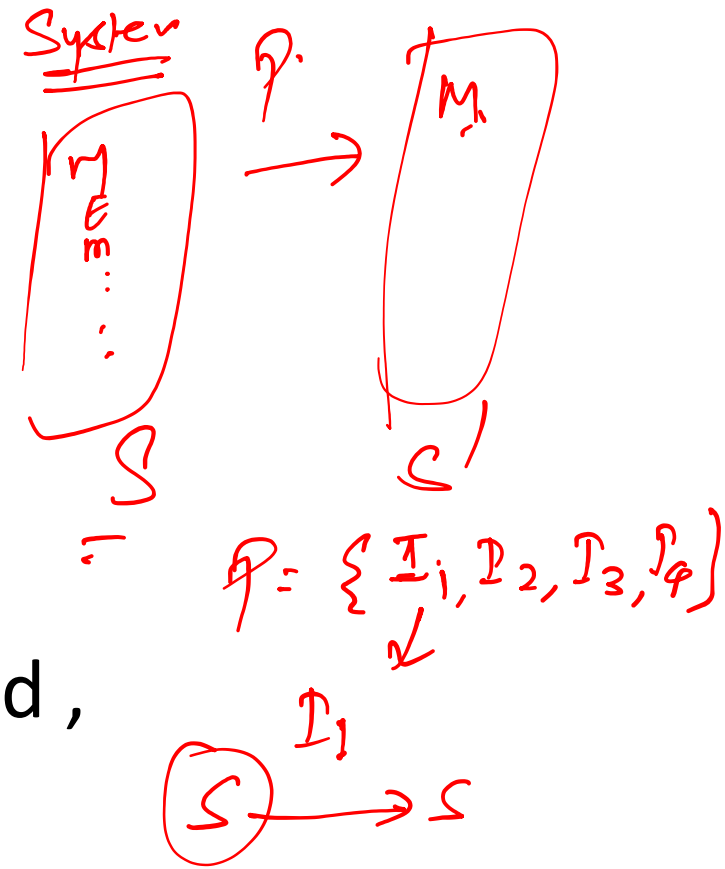
- Assembly instructions

  add t0, g, h

  add t1, I, j

  sub f, t0, t1

- Opcode/mnemonic, operand , source/destination



Why not Bigger Instructions?

- Why not "f = (g+h) – (i+j)" as one instruction?
- Church's thesis: A very primitive computer can compute anything that a fancy computer can compute – you need only logical functions, read and write to memory, and data dependent decisions
- Therefore, ISA selection is for practical reasons
  - Performance and cost not computability
- Regularity tends to improve both
  - E.g, H/W to handle arbitrary number of operands is complex and slow, and UNNECESSARY

Primitive computing System

Instruction

logical ¬, {∧, ∨} ± *

memory Read / Write

data dependent decision

a = b + c add b, c, a

Inp. →
Inp. →
→ .

I1, I2, I3, ... In

Thank You