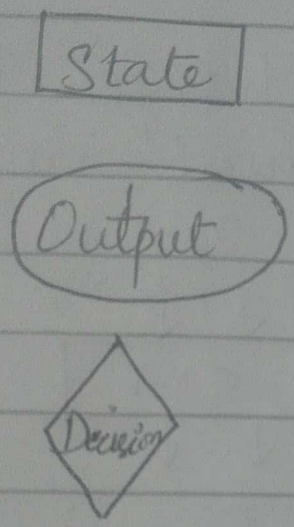
FINALS

Chap 9

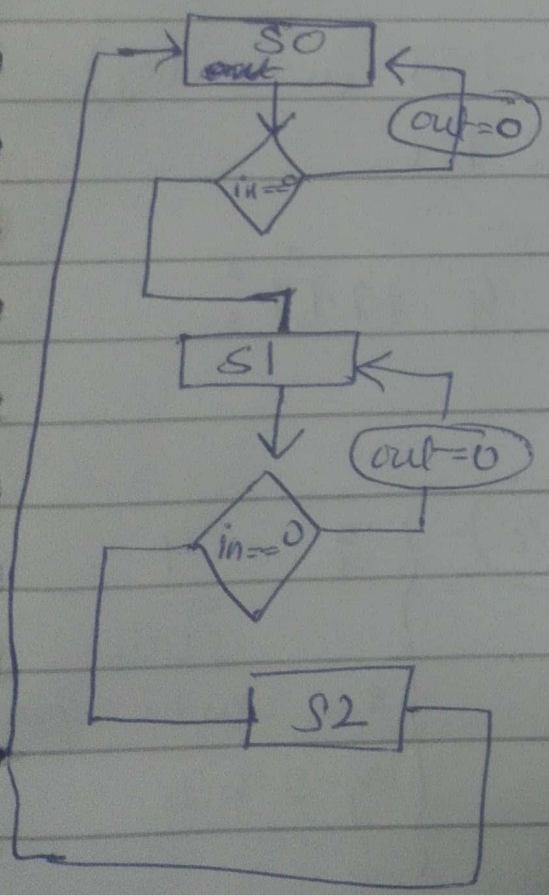
Algorithmic State machine (ASM chart)

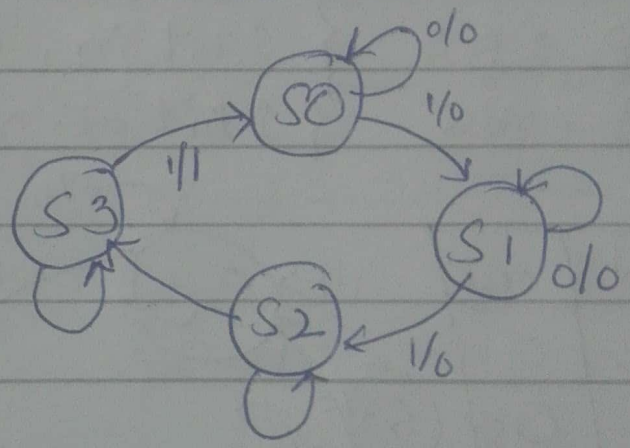


alternate way to represent State machine

Bubble Diagram

figurative way of representing state machines

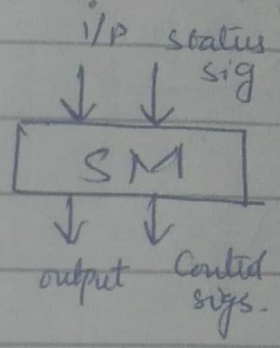




* has more readability

Chap 10 MicroCoded/ MicroProgrammed SM

Crux: We do not desire to change the hardware of our data path implem in any case where states change.

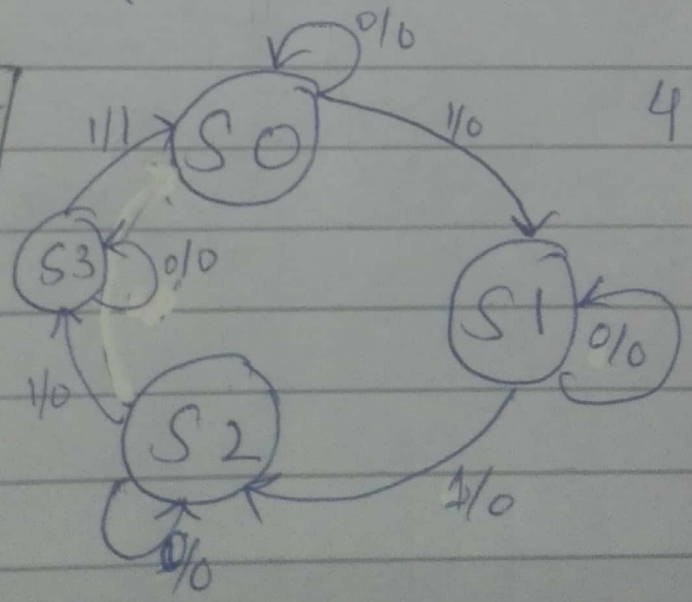


Truth table

i/p	CS	o/p	NS

Example i/p , CS { 1 bit in, CS 2 bit (w.r.t diag)
o/p , NS { 1 bit out, NS 2 bit

CS	in	NS	out
00	0	00	0
00	1	01	0
01	0	01	0
01	1	10	0
10	0	10	0
10	1	11	0
11	0	11	0
11	1	00	1

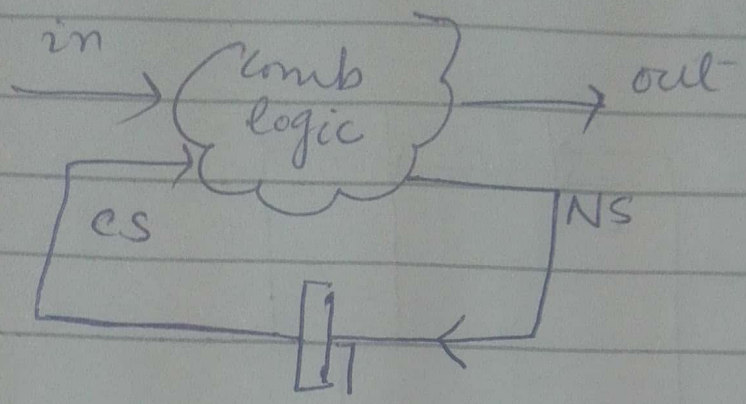


4 '1111's.

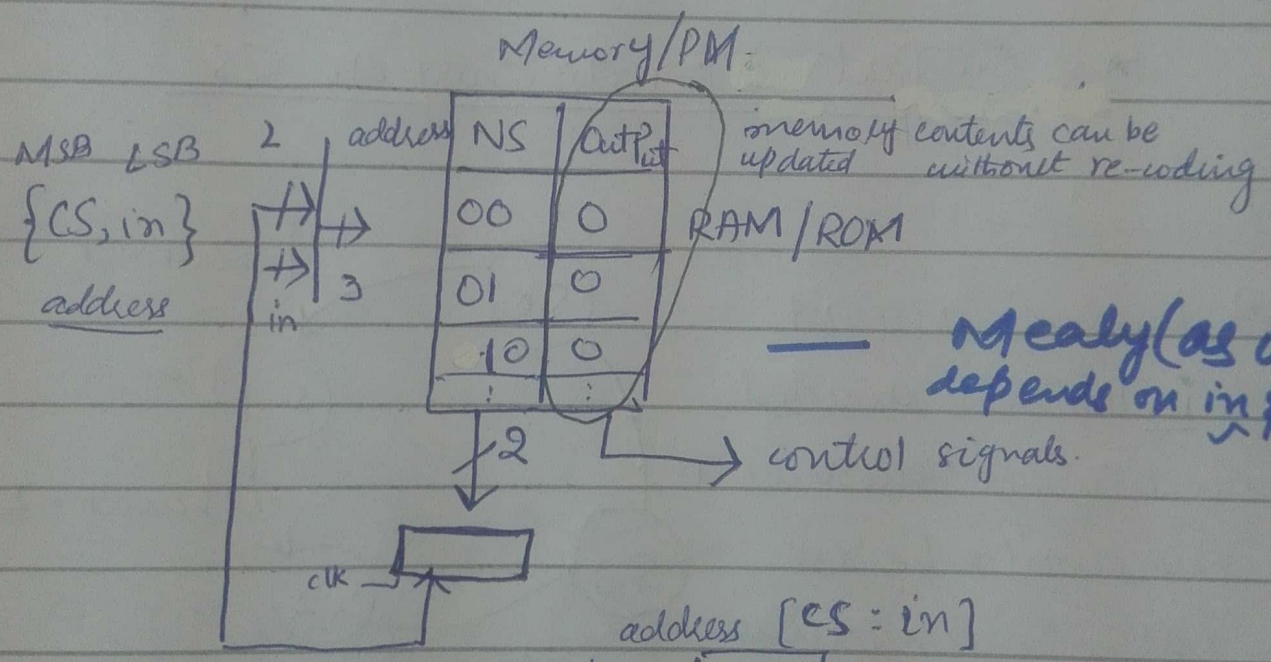
$2^2 = 4$ state

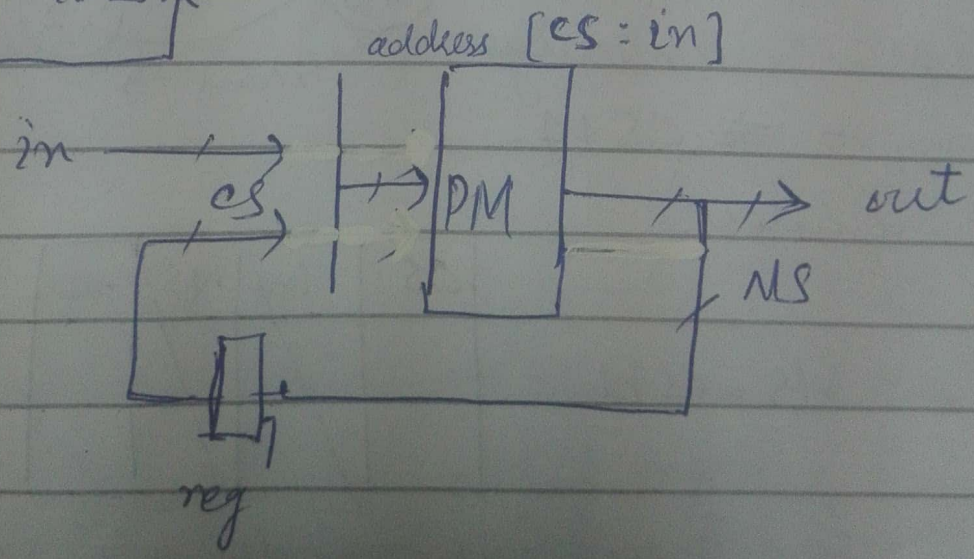
thus can be repre by 2 bits

We know

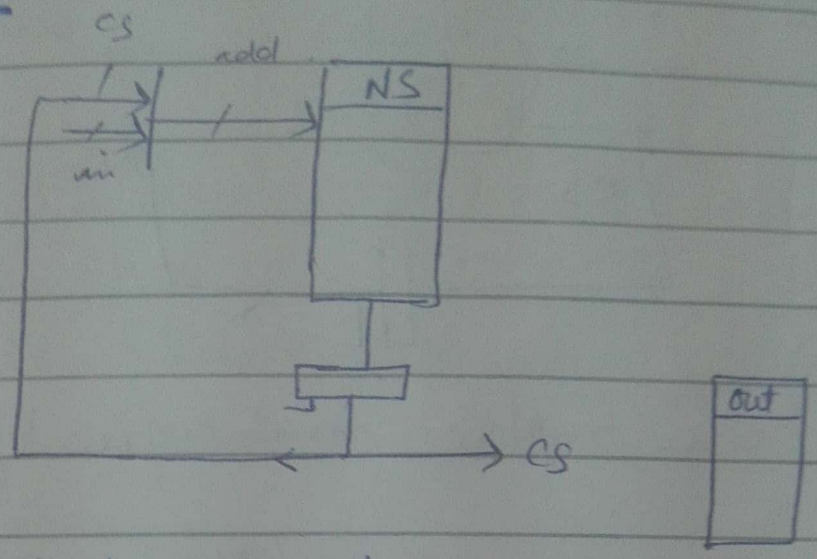


We can replace comb logic by the truth table



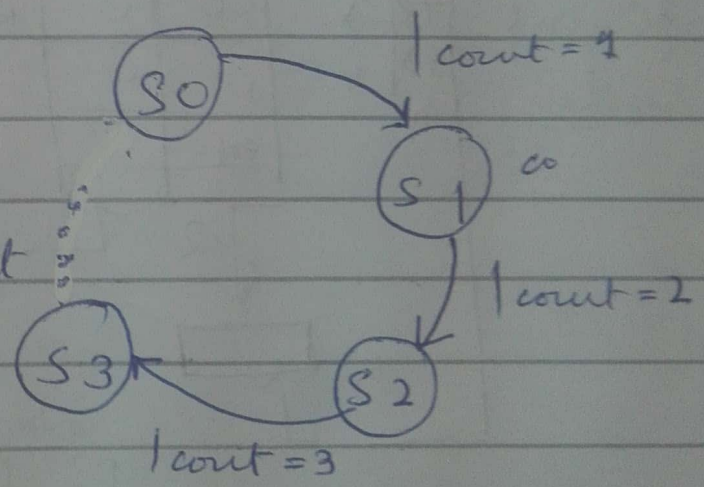


Moore



★ Output depends only on CS

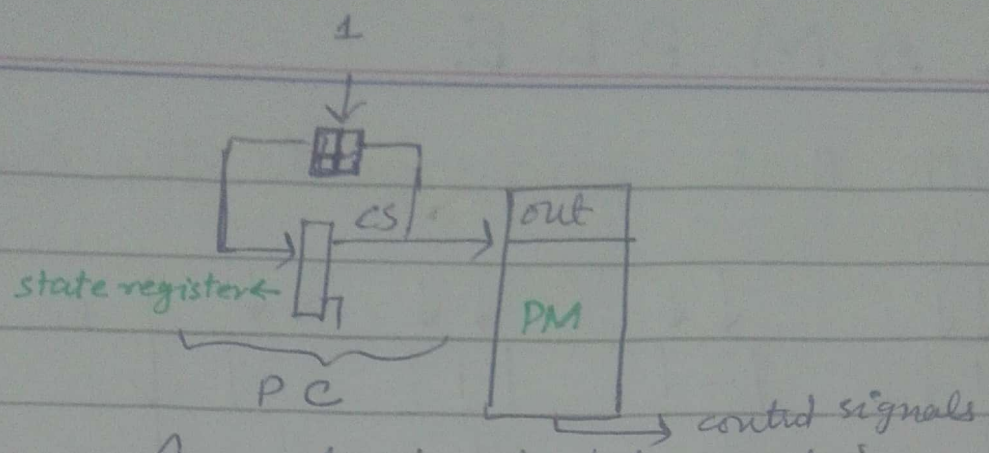
A kind of state M where in has been avoided so we always know what the NS would be as this S.M is continuously moving to CS+1. on a counter.



NS = CS+1.

CS	in	NS	out

Thus our model is simplified.



A counter based state machine

Custom Processor Design

(Counter based SM)

A Process uses min 3 cycles.

* Instr. fetch
* Instr. decode
* Instr. execute

With any Processor is linked "ISA"

i.e what type of instr. it executes

(1) ALU based (add, sub, mul ...)

(2) Control flow (if/else, loop, branch, goto)

EXAMPLE

- **ALU based (Sequential 1 by 1)**

General Instructions

$R_i = R_j + R_k$

$R_i = R_j - R_k$

$R_i = R_j \mid R_k$

$R_i = R_j \,\&\, R_k$

$R_i = R_j$ (MOV)

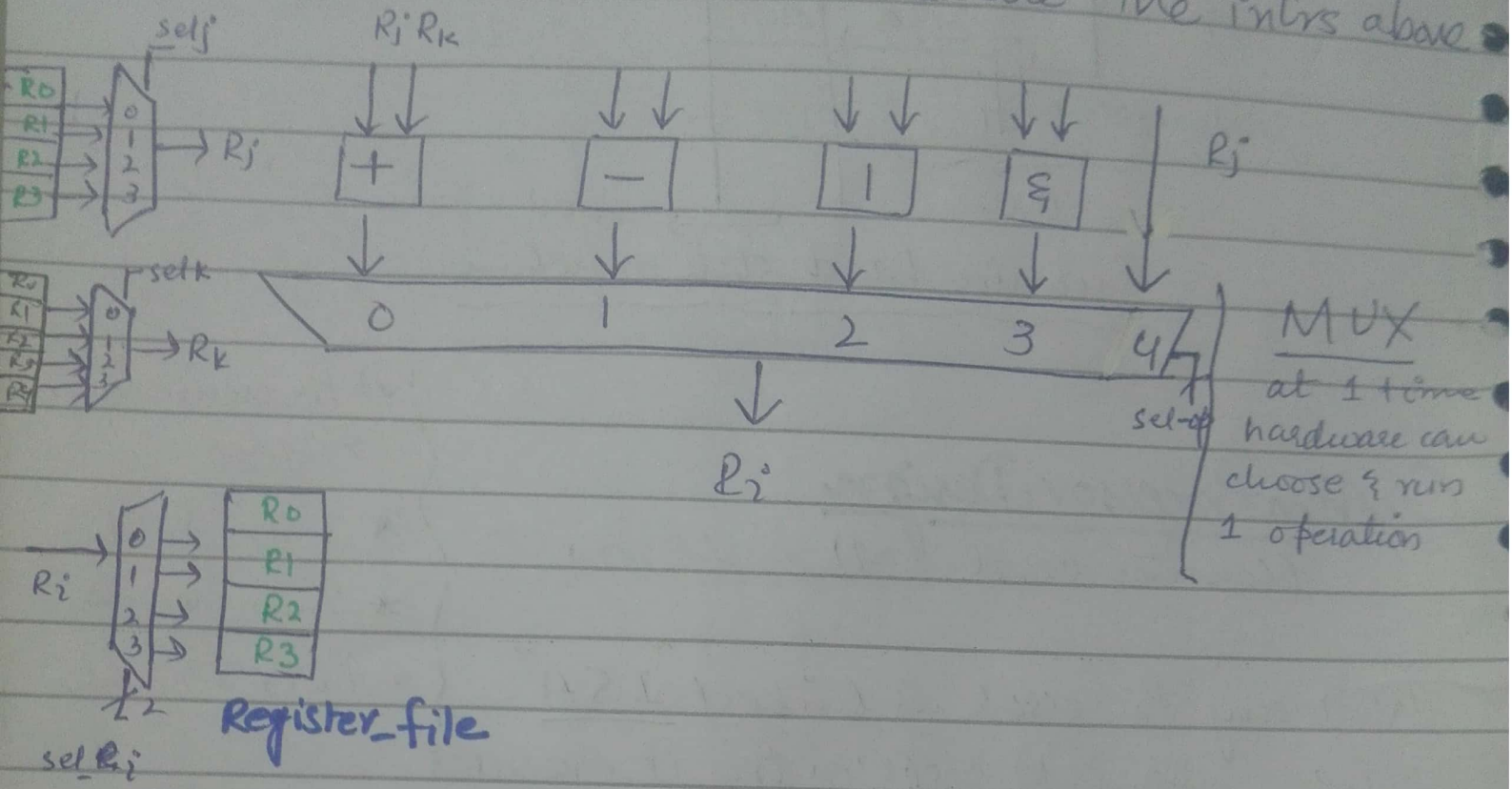
$R \rightarrow$ file register

$i, j, k \in \{0, 1, 2, 3\}$.

* all registers are 8 bit wide

EXAMPLE

★ DataPath that would execute the intrs above

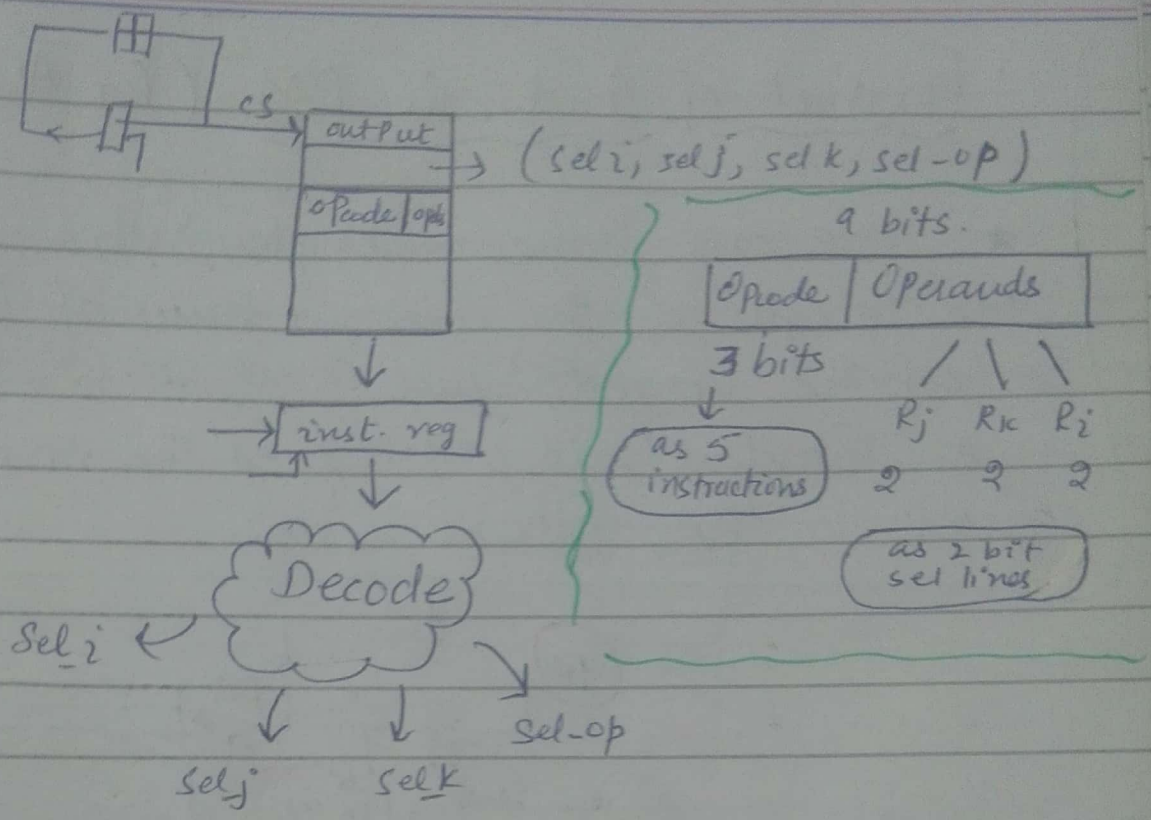


Control Signal of Datapath ↑

Sel i , sel j , sel k , sel-op

As this datapath is executing sequentially so we need a counter based control unit for it

$R_n = R_i + R_j + R_k + R_m;$ →



EXAMLE II

General Inst's :-

1. $R_i = R_j \oplus R_k$
2. $R_i = R_j \ominus R_k$
3. $R_i = R_j \uparrow R_k$
4. $R_i = R_j \& R_k$
5. $R_i = R_j$

} Controls DataPath

ALU Based Inst's (Sequential manner)

1 inst will always execute in in one cycle.

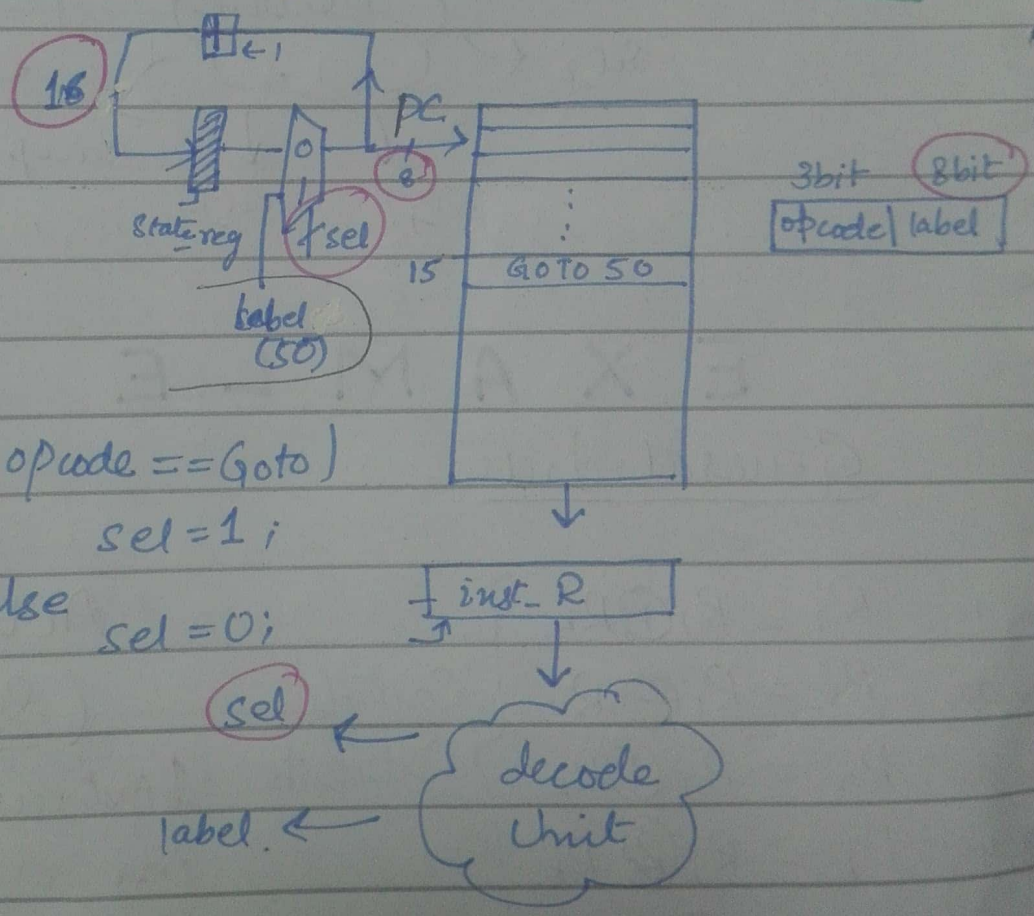
$op \in (+, -, *, |, \&)$

6. Goto label (Unconditional jump) } Control Unit Based.

Format of Instr. in PM (for ALU & Goto)

★ We can have multiple instrs. in PM. So the length of the Inst in terms of bits would be of the largest inst.

★ Constant will be assigned the no of bits equal to the bits of system.



How?

```
if(opcode == Goto)
    sel = 1;
else
    sel = 0;
```

★ Instruction

op	R_1	R_2	50	

?

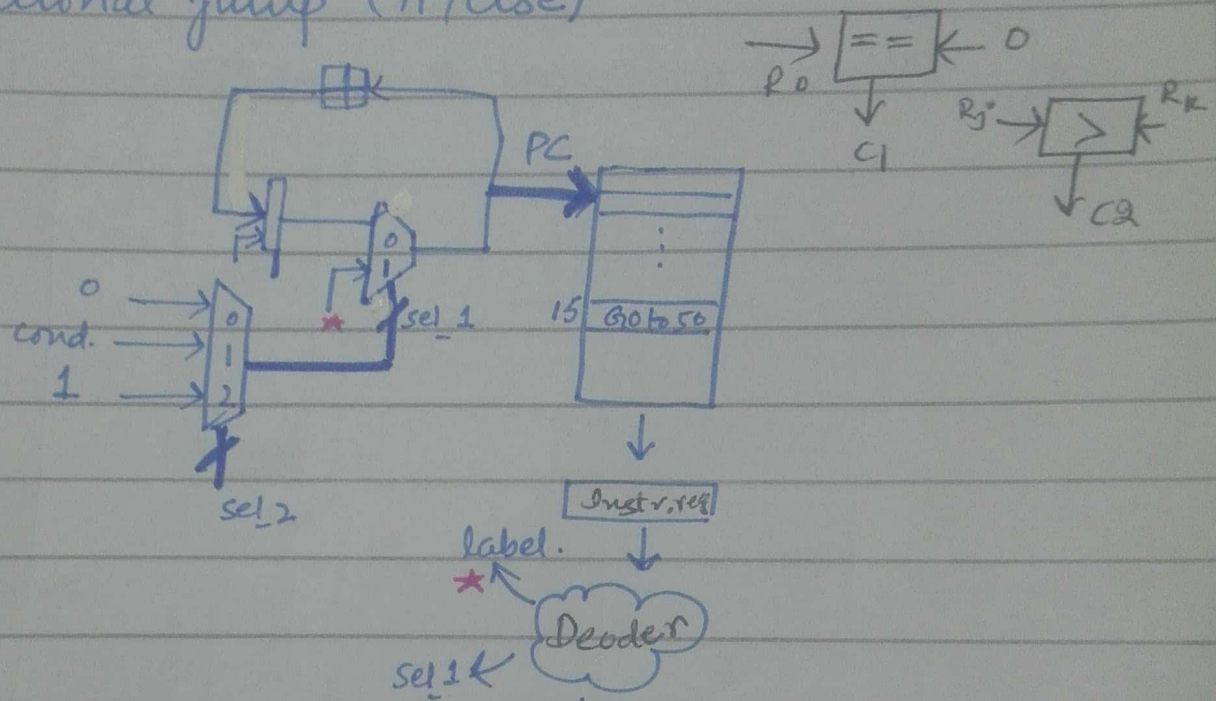
Bits? Binary?

★ What would be label?

EXAMPLE III

General Inst's.

1. ⋮
2. ⋮
6. ⋮
7. Conditional jump (if/else)



sel_2 = 0 ALU
sel_2 = 2 Un.cond jump
sel_2 = 1 if/else

sel_2 → values will be written in decode insts.

Format of Instr in PM (for Condi. jump)

opcode	Ri	add/label

if (Ri == 0)
else normal

opcode	Ri	Rj	label.

if (Ri > Rj)
else normal.

if your if/else condition is fixed, do not accommodate 2 bits.

if (R0 == 0)

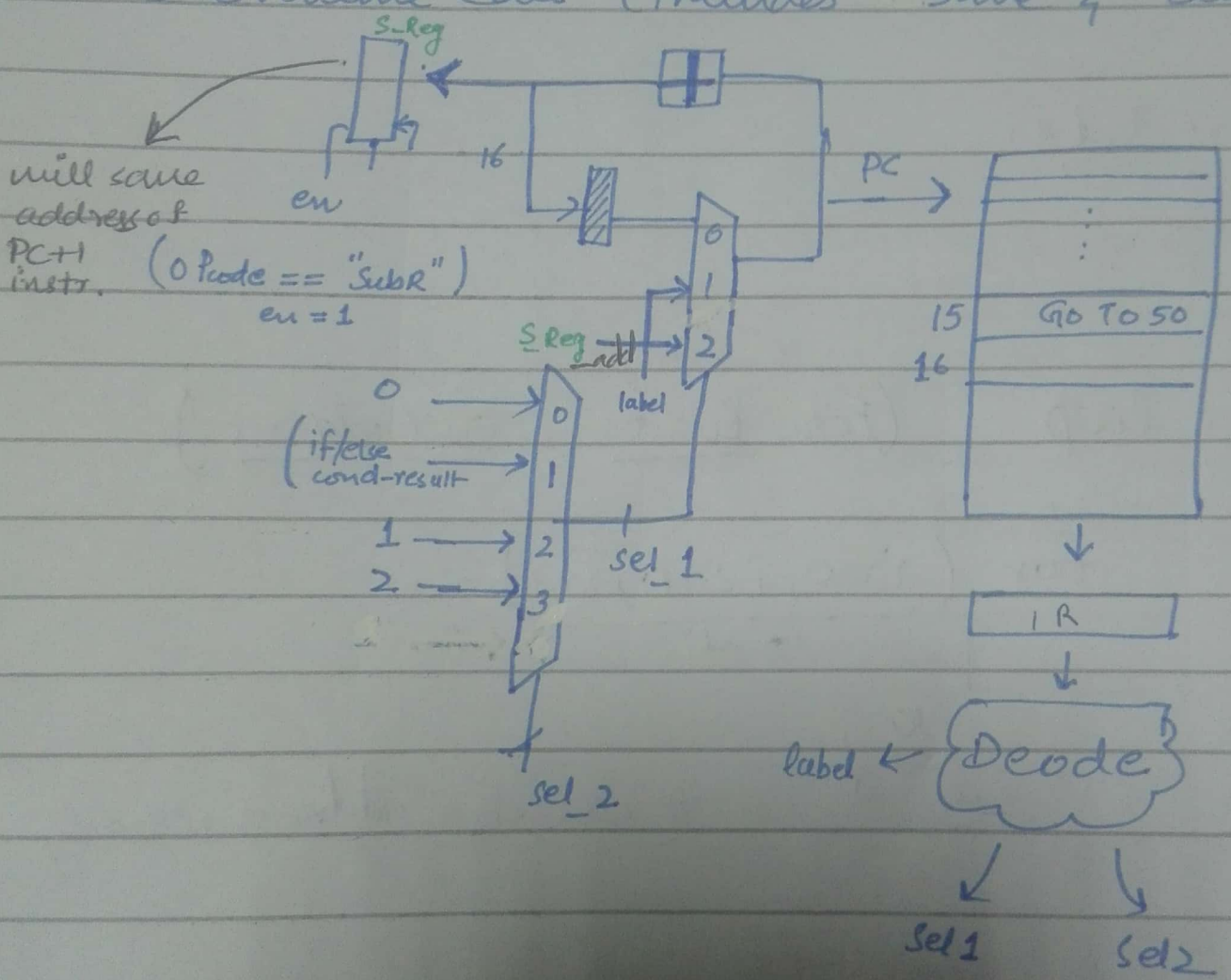
op	~~Rc~~	add

When we want to execute both if & else

EXAMPLE IV

General Insts.

1
2
⋮
7
8 Subroutine Call (includes ~~ri~~ save & return)



* While { if (cond) call sub
else normal }

if | cond jump add
add | subroutine

do it as a single instr of If.

Format of Instr in PM (subroutine & call)

Sub	opcode	add/label
return	opcode	—

General Instr:

1 :
2 :
:
:

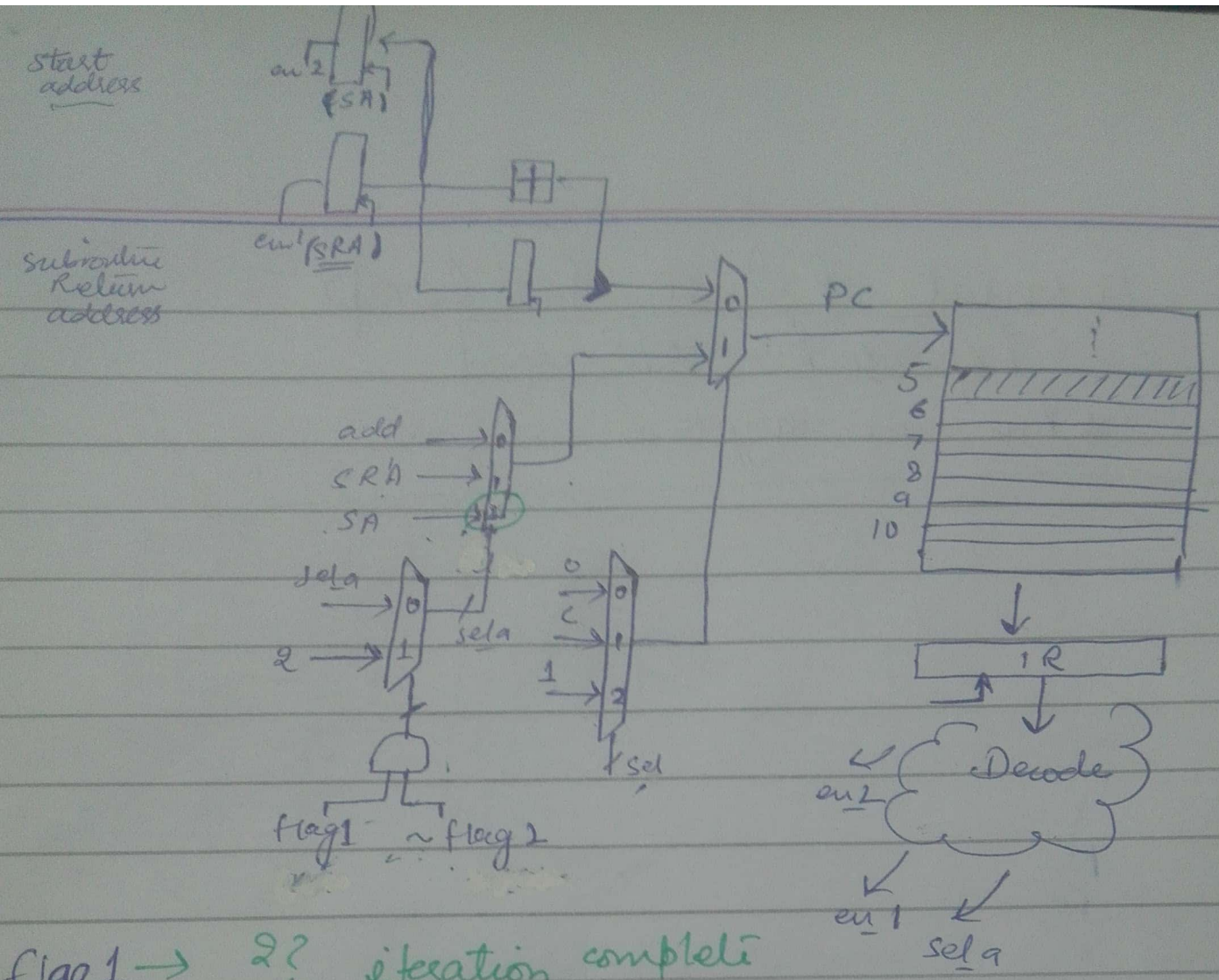
9 For loop (For count ; end address)

```
for (i=0, i<10, i++)
{
    }
```

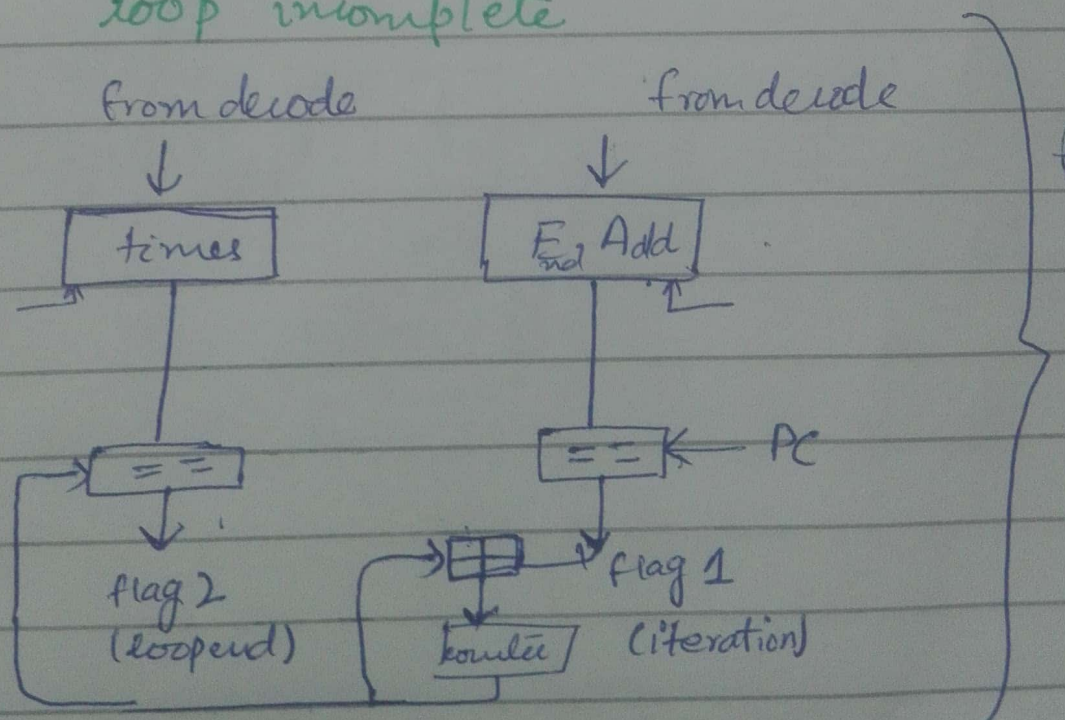
count

10 — end address

PC has start address already



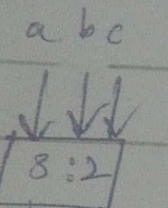
flag 1 → 2? iteration complete
loop incomplete



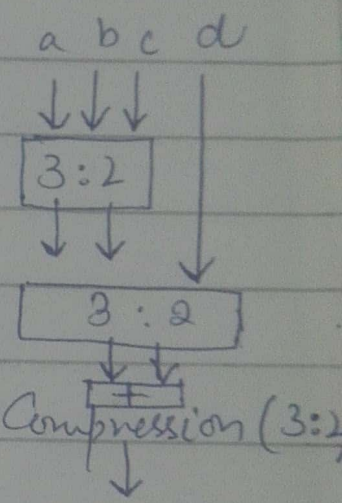
15 Jan 2018

Fast Multipliers / Multi Operand Addition

Carry Save Adder



→ Takes 3 input numbers
→ Generates 2 o/p's
→ delay of 1 full adder.
→ For more than 3 inputs → 2nd iteration



Example

1 F.A in Parallel.	0	1	0	0
	0	0	1	1
+	0	0	1	0
On this, simple adder	0	1	0	1
	0	0	1	0

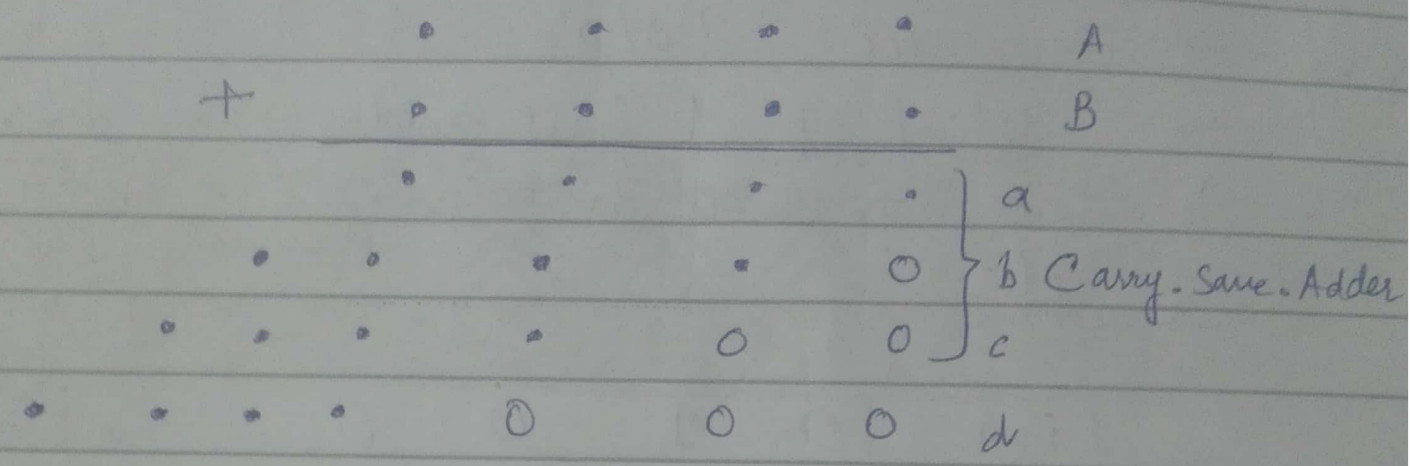
Sum
Carry

This step will take delay of a least a normal adder.

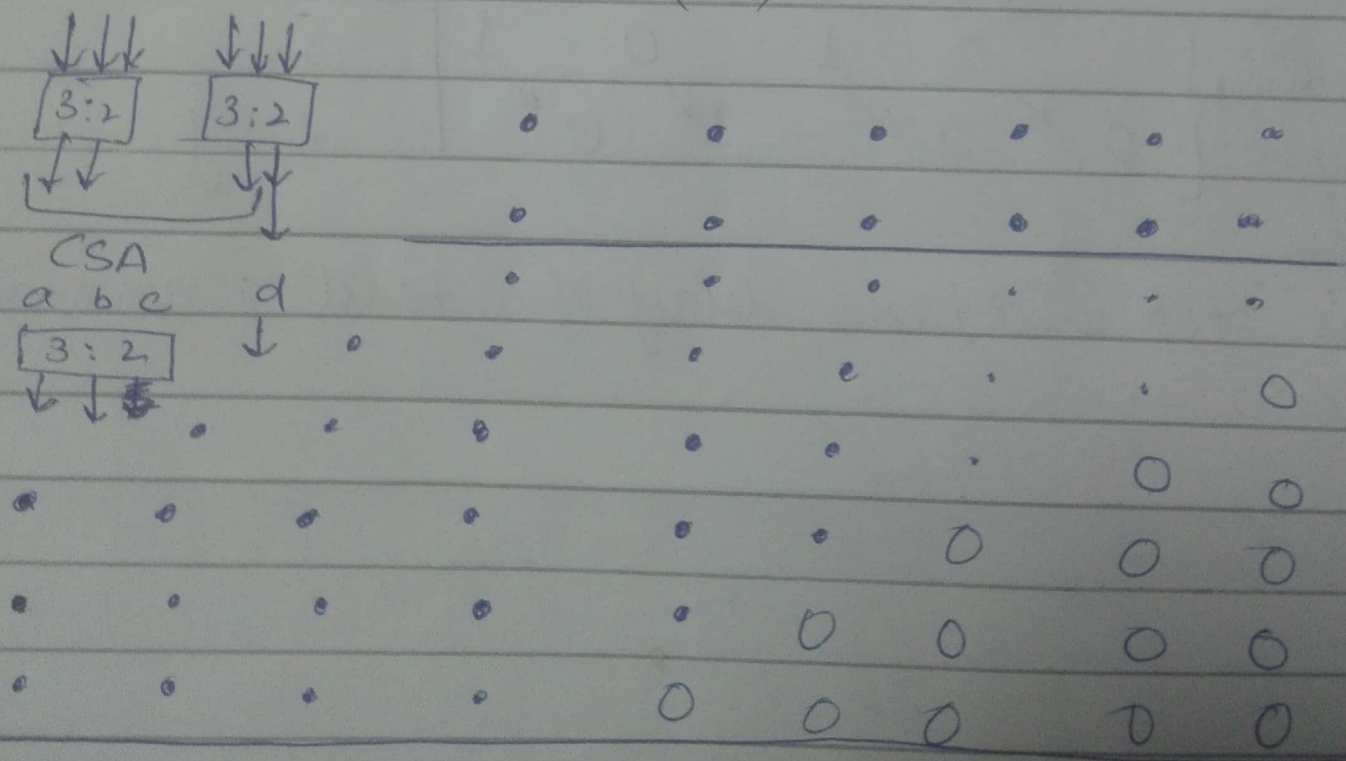
Do..

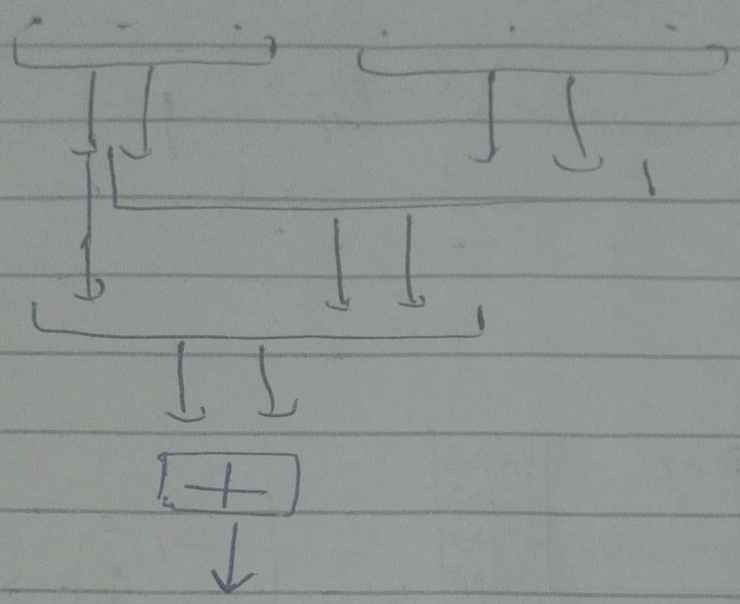
1) Carry Save Reduction — 1 CSA
2) Dual Save Reduction — 2 CSA
3) Wallace Tree — mor CSA

(1)



(2)





→ Solve Numerically from Book.

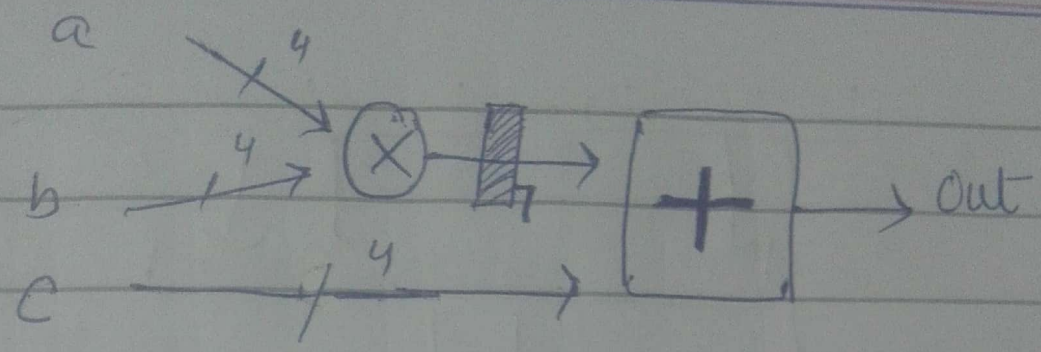
Figuratively Example

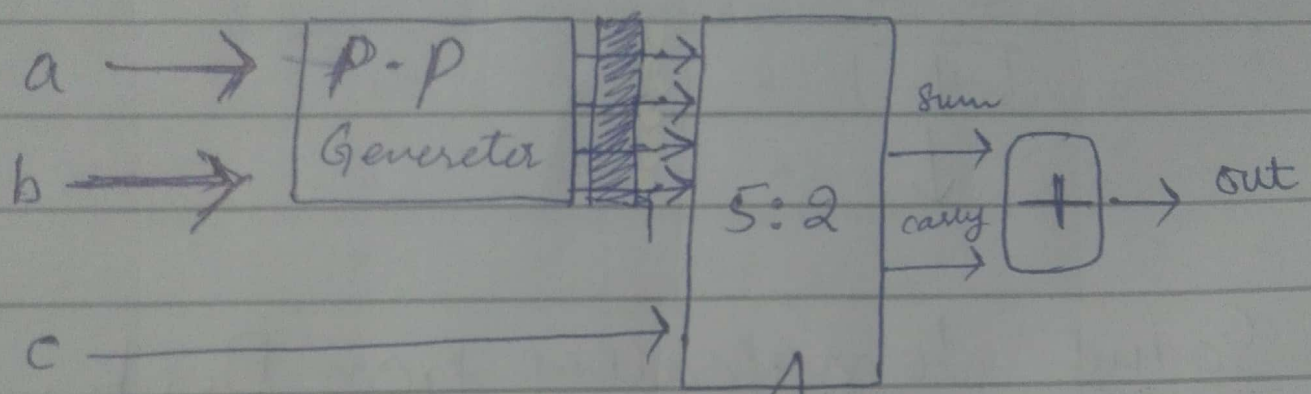
- Transform following using compression trees such that it uses a single CRA

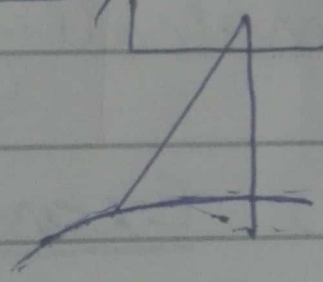
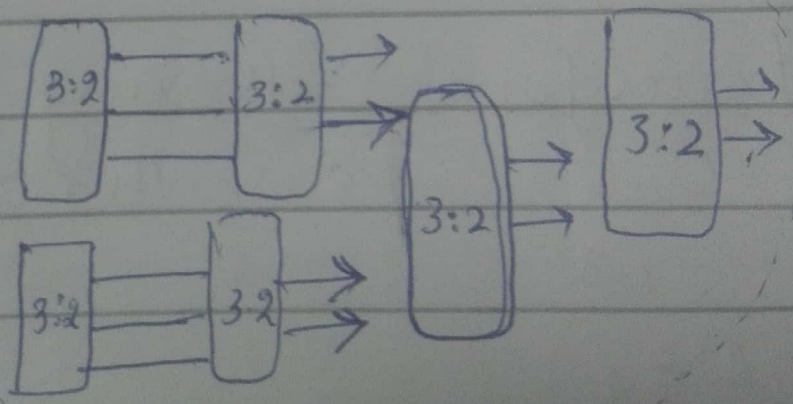
a $Q_{1.3}$

b $Q_{1.3}$

c $Q_{2.2}$







if 6:2.