Pipeline

Complete Revision

- Deepak Poonia

(IISc Bangalore; *GATE AIR 53 & 67*)

Content of this Lecture:

1. Pipeline Introduction
2. Data Dependency & Hazards

GATE 2024 Complete Course Link:

https://www.goclasses.in/s/pages/gatecompletecourse

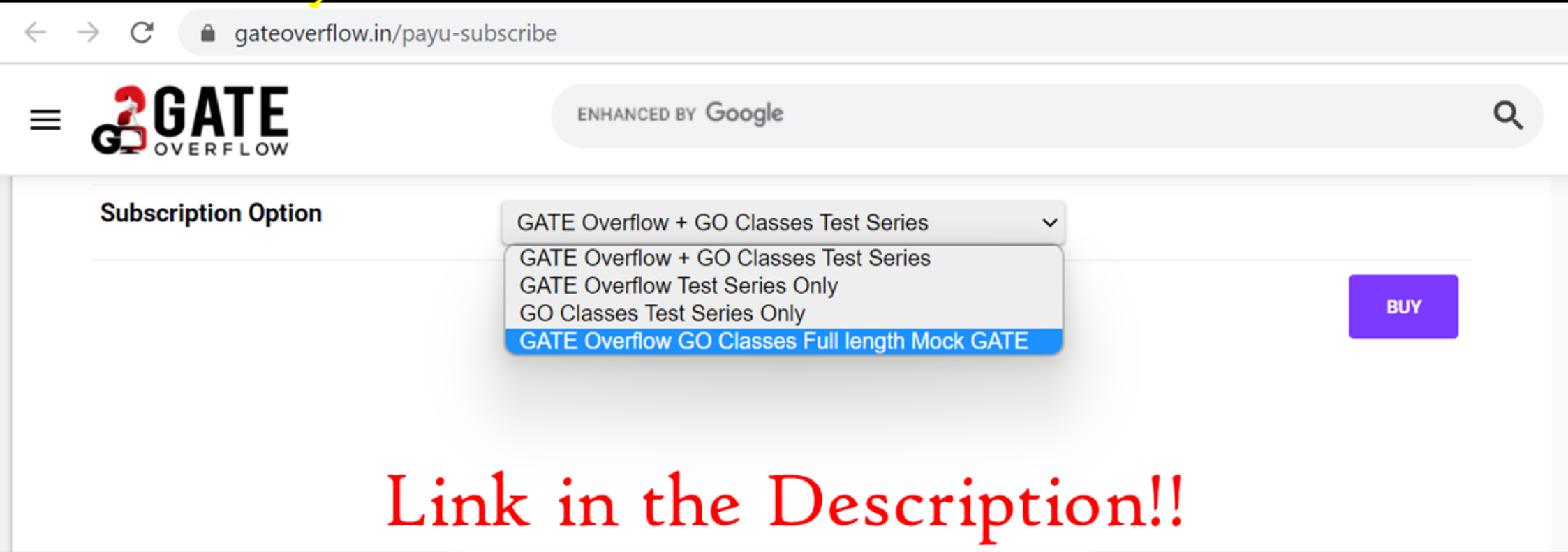
Instructor:

Deepak Poonia

IISc Bangalore

GATE CSE AIR 53; AIR 67; AIR 206; AIR 256

Subscribe for ALL 15 Mock Tests By GateOverflow + GO Classes



Link in the Description!!

GO CLASSES

GATE 2024
COMPLETE COURSE CS-IT

(1 YEAR)

GO CLASSES

GATE 2025

COMPLETE COURSE CS-IT

(2 YEARS)

2023 Discrete Mathematics

★ ★ ★ ★ ★ 5.0 (62 ratings)

Deepak Poonia (MTech IISc Bangal...

Free

2023 C Programming

★ ★ ★ ★ ★ 5.0 (59 ratings)

Sachin Mittal (MTech IISc Bangalor...

Free

On “GATE-Overflow” Website

GO CLASSES

GO Classes

GO CLASSES

GATE OVERFLOW

LIVE

GATE 2023

Live + Recorded Lectures

Daily Home Work + Solution

Watch Any Time + Any Number of Times

Summary Lectures For Every Topic

Practice Sets From Standard Resources

Enroll Now

+91 - 6302536274

www.goclasses.in

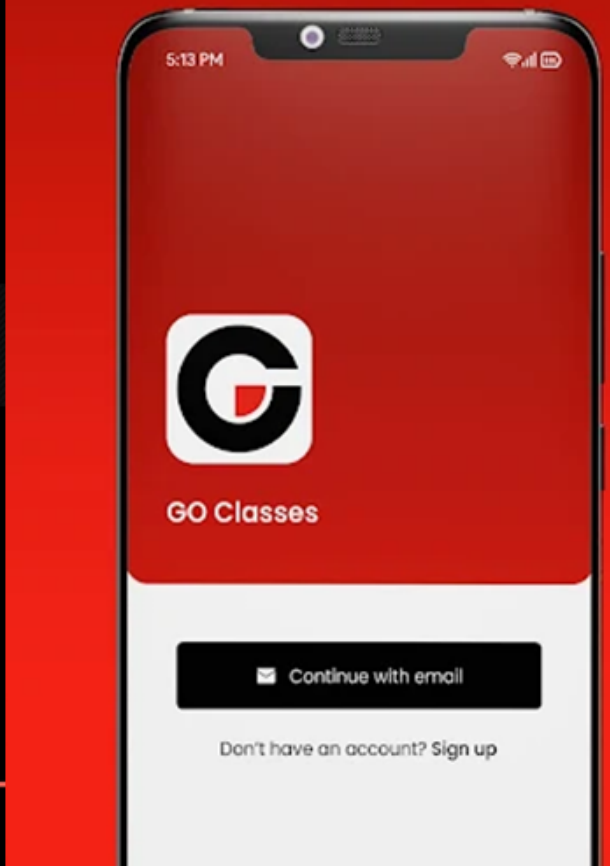
linkedin.com/company/go-classes

instagram.com/goclasses_cs

Download the GO Classes Android App:

https://play.google.com/store/apps/details?id=com.goclasses.courses

Search "GO Classes" on Play Store.



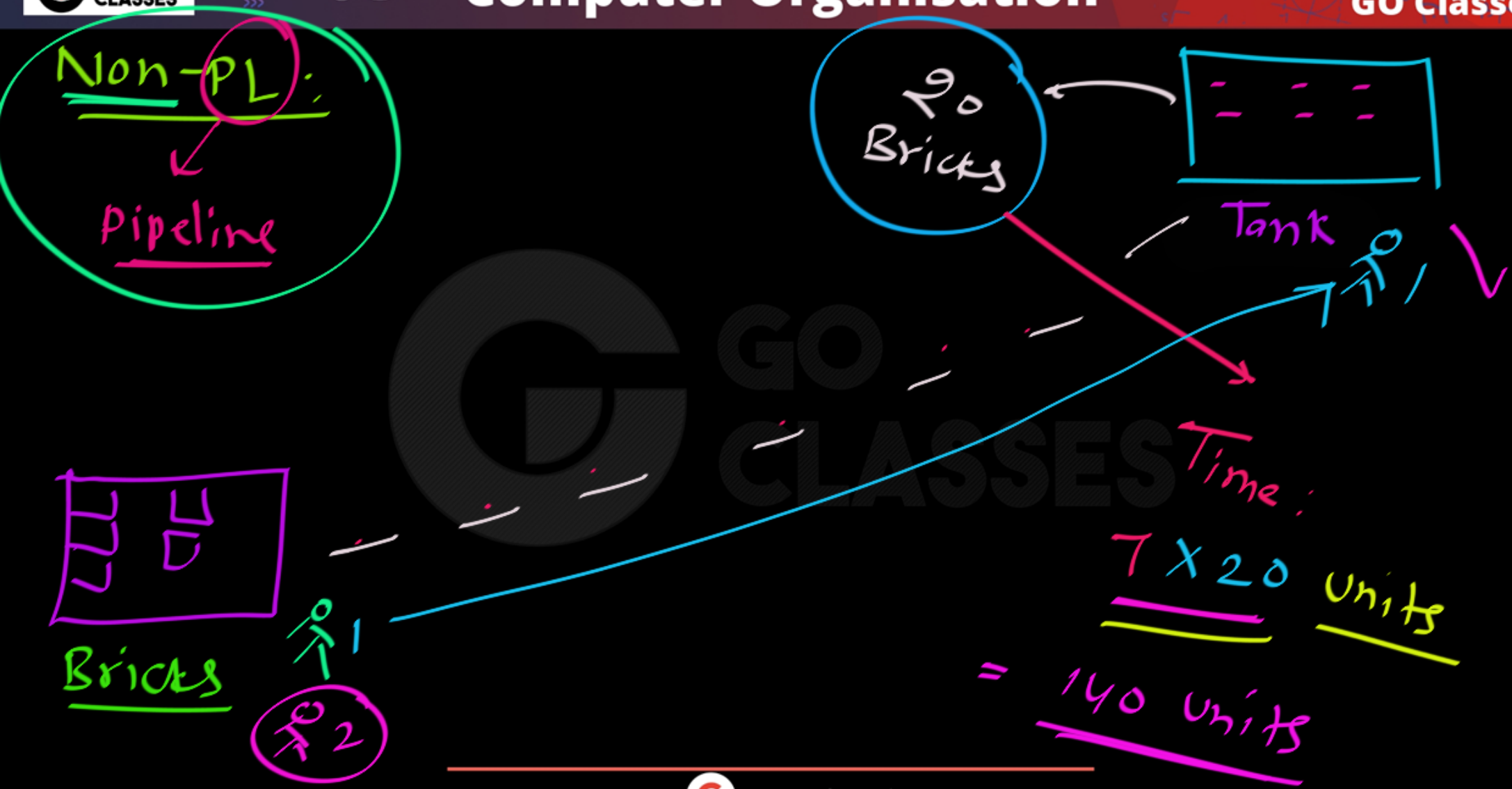
Pipeline

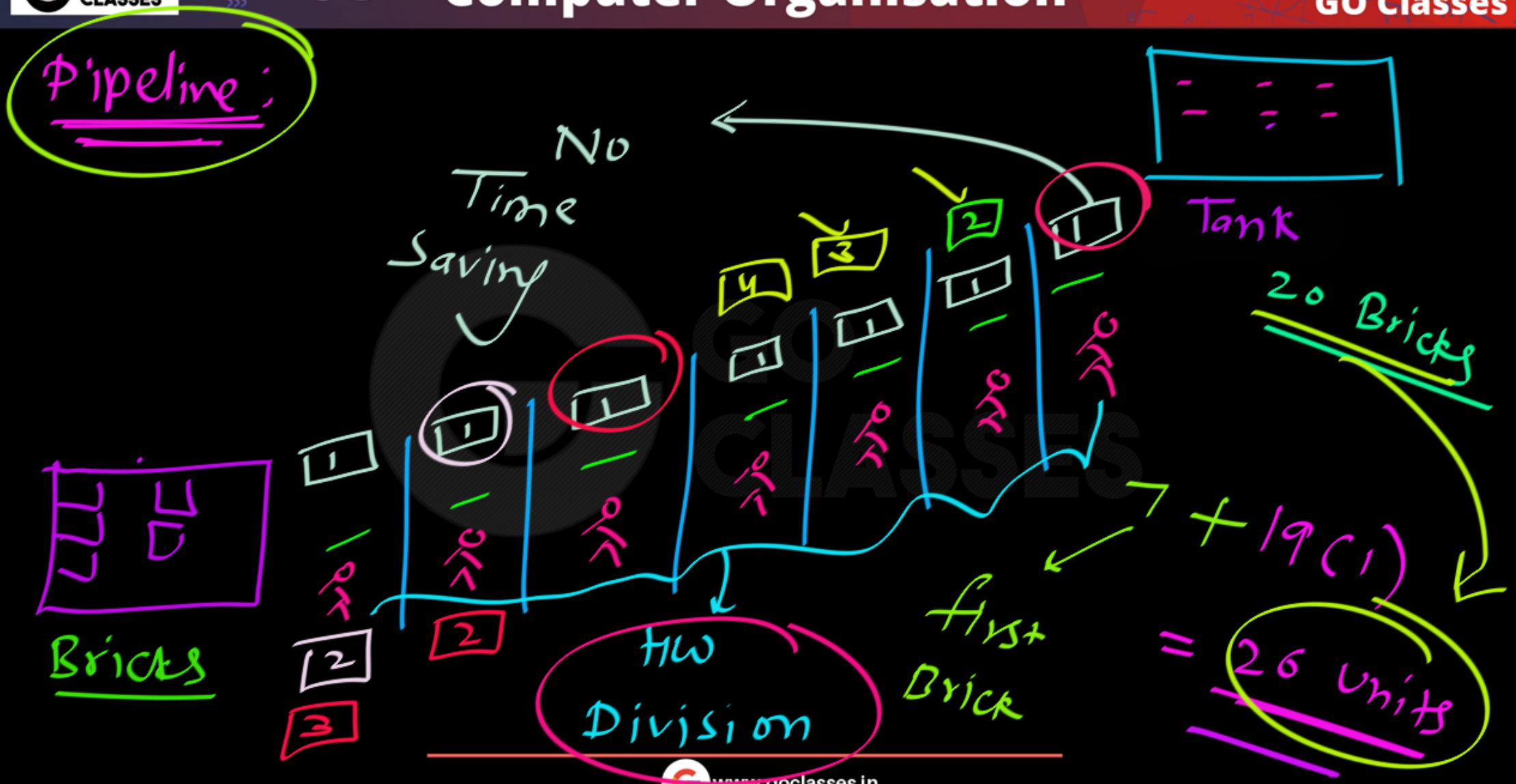
Pipeline:

You ALREADY Know & Understand Pipelining.

Scenario 1.

Pipelining in Construction


Non-PL :
Pipeline
90 Bricks
Tank
Bricks
P1
P2
Time :
7 X 20 Units
= 140 units

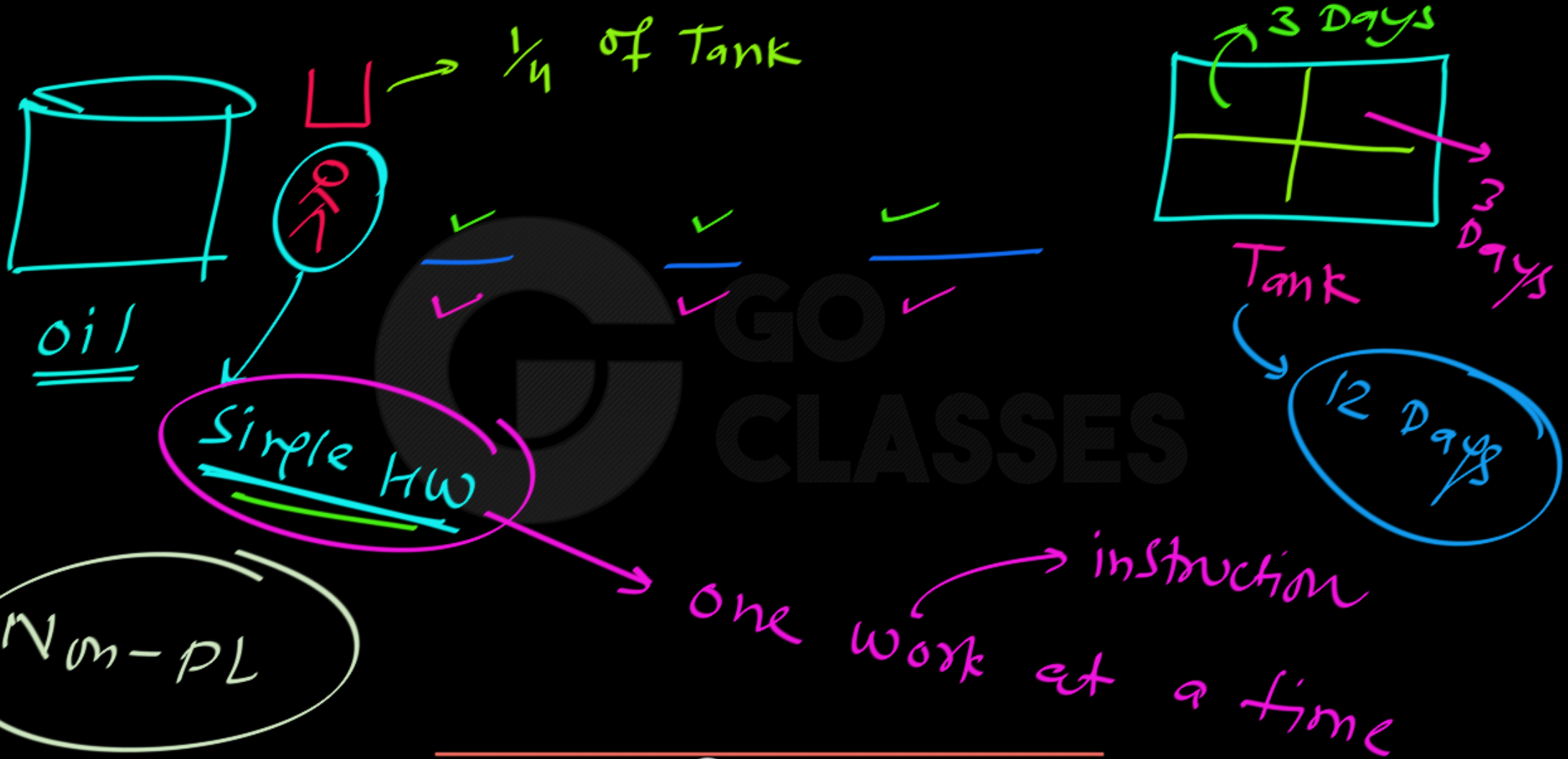

Pipeline :
No Time Saving
Tank
20 Bricks
Bricks
HW Division
First Brick
+19(1)
= 26 units

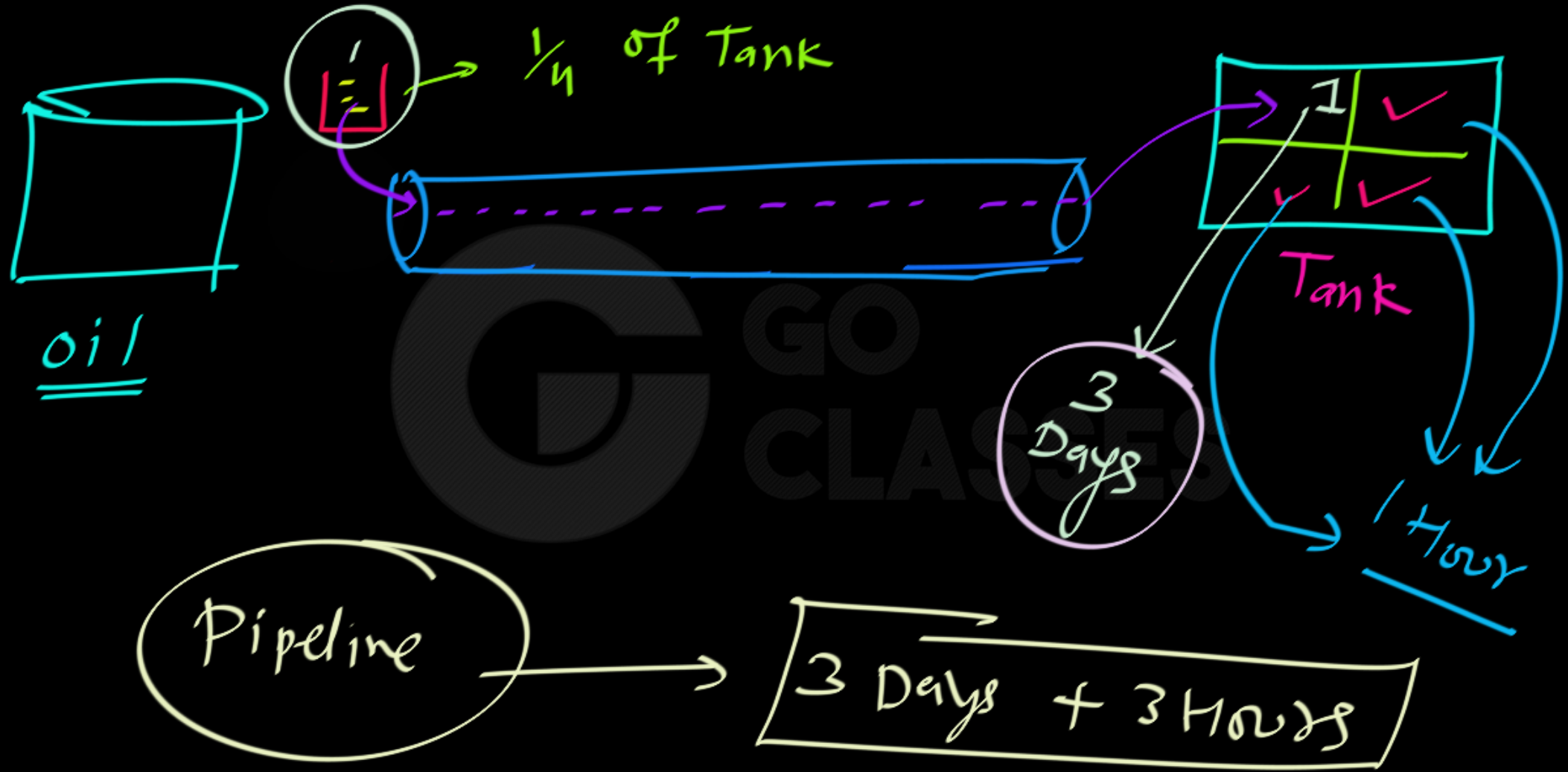
Scenario 2.

Oil Pipeline

GO CLASSES

GO Classes


1/4 of Tank
3 Days
3 Days
Tank
12 Days
oil
Simple HW
Non-PL
one work at a time
instruction


1/4 of Tank
Oil
Tank
3 Days
1 Hour
Pipeline
3 Days + 3 Hours

4.5 An Overview of Pipelining

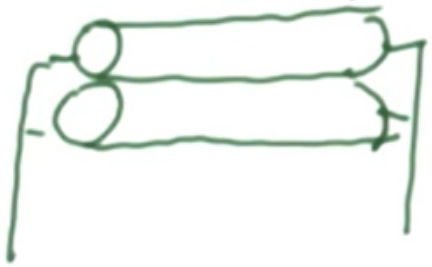
Never waste time.
American proverb

Pipelining is an implementation technique in which multiple instructions are overlapped in execution. Today, **pipelining** is nearly universal.

LAUNDRY PIPELINING QUIZ

1 HOUR

1 HOUR



1 HOUR

- 10 LOADS OF LAUNDRY

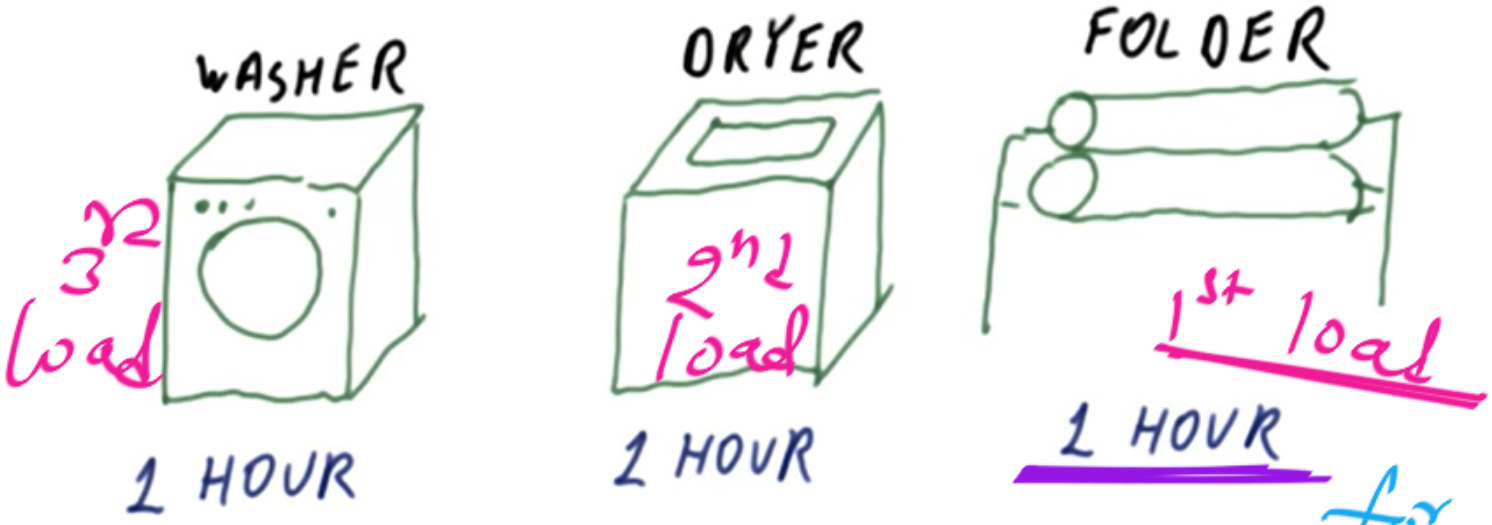
NO PIPELINING => _____ HOURS

WITH PIPELINING => _____ HOURS

Source: Google Tech Univ.

GO CLASSES

LAUNDRY PIPELINING QUIZ



- 10 LOADS OF LAUNDRY

NO PIPELINING => 10×3 HOURS (for one load of laundry)

WITH PIPELINING => $3 + 9(1)$ HOURS = 12 Hours

1st load

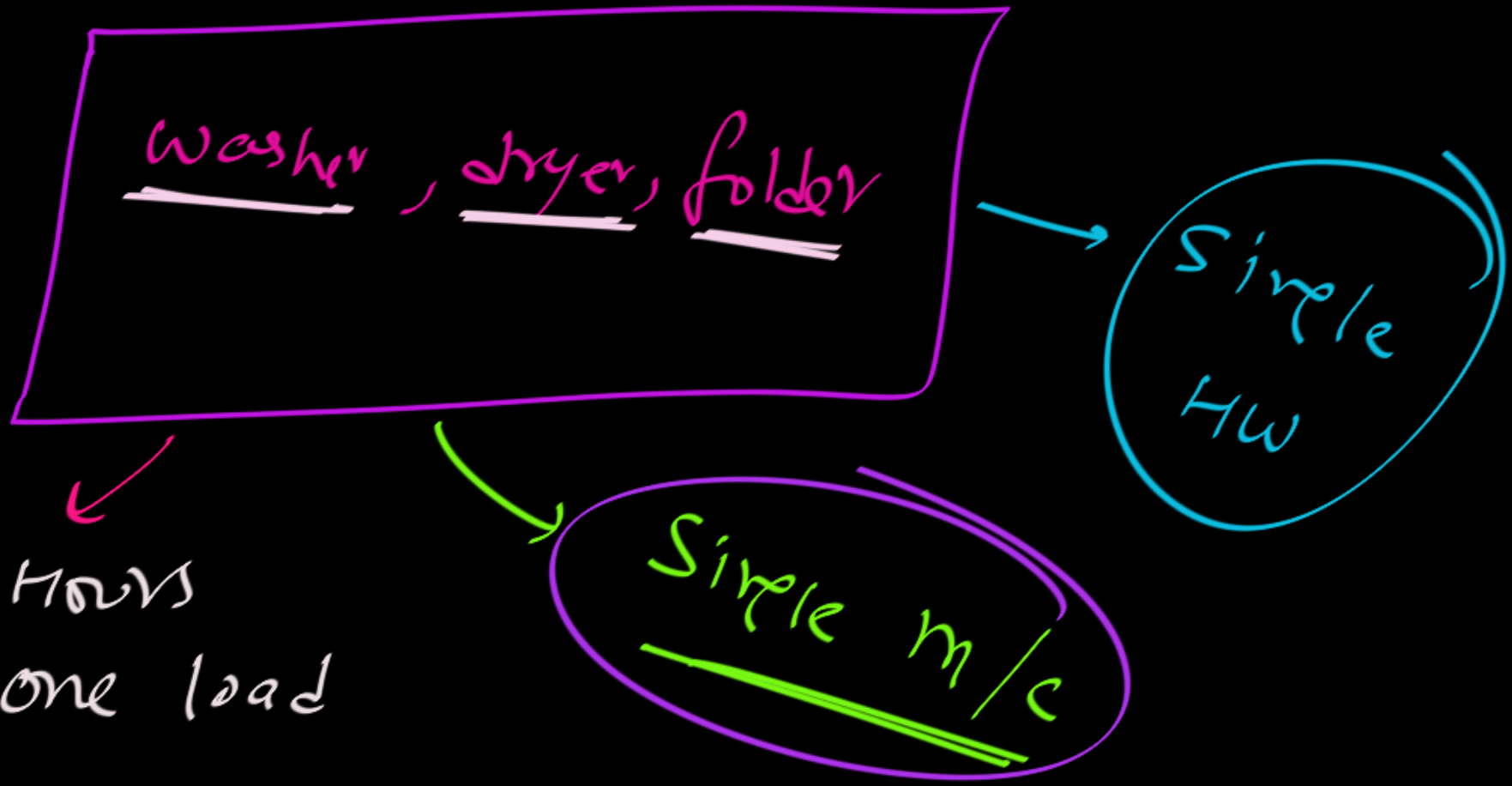
GO Classes

Speedup of Pipeline over Non-PL :

$$\frac{\text{NPL}}{\text{PL}} \quad \frac{30}{12}$$

for 10 loads of Laundry.

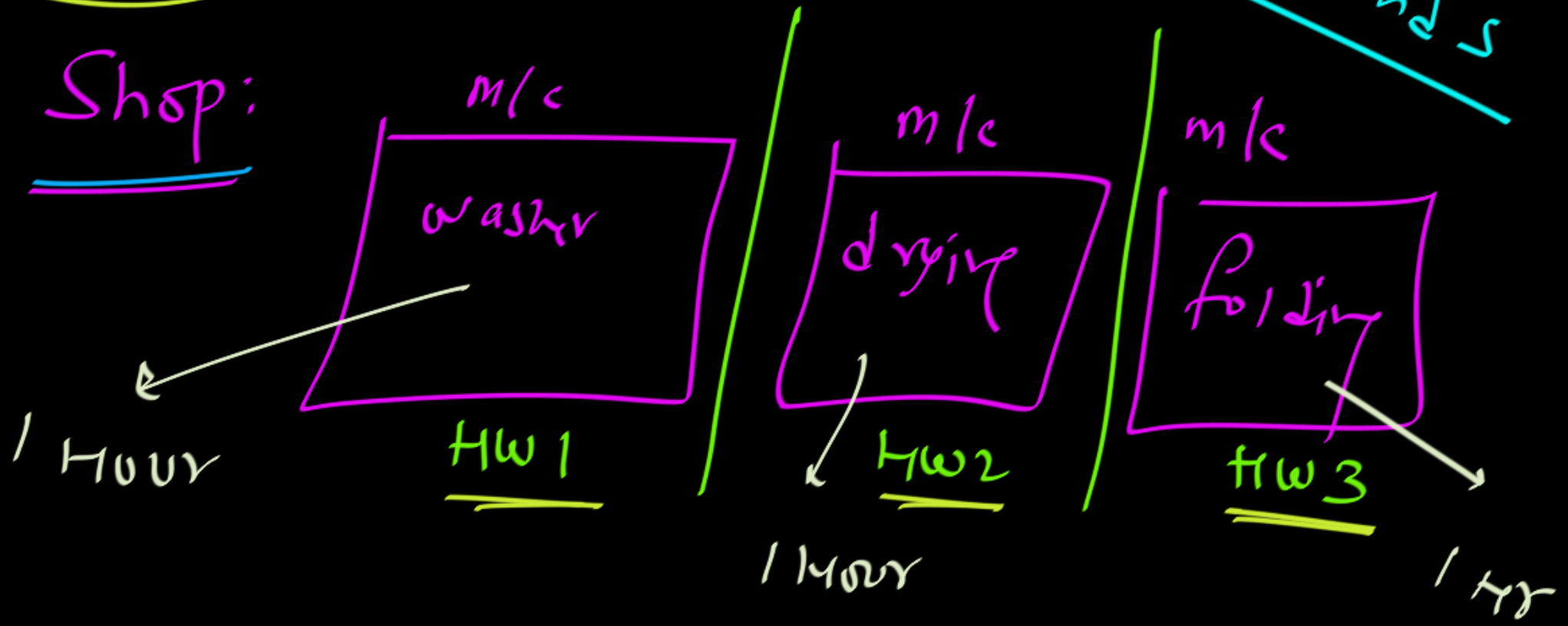
Non-Pipeline Laundry m/c :



Pipelined Laundry m/c:

Friends

Shop:



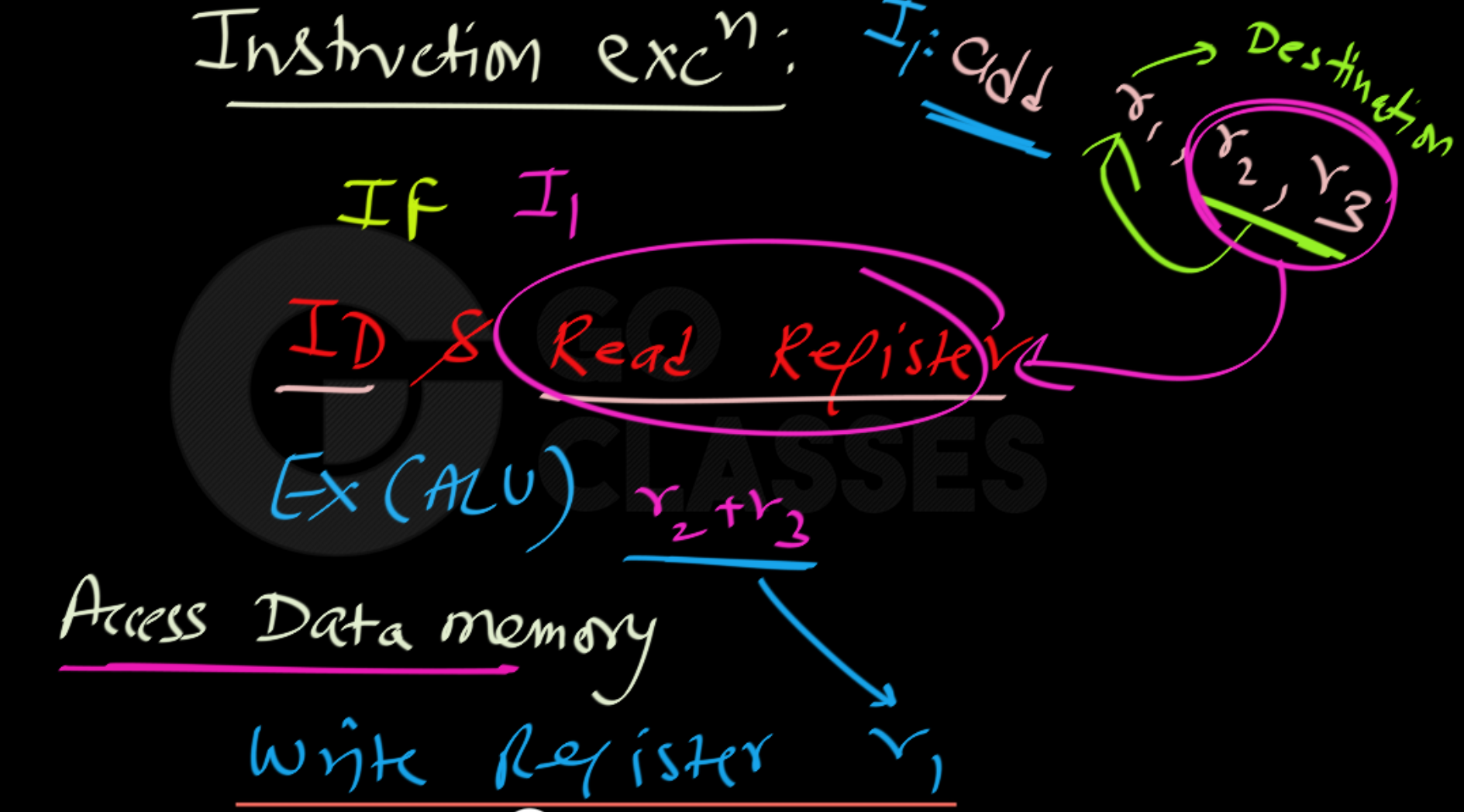
Pipeline:

Pipelining improves efficiency by executing multiple instructions simultaneously.

Pipeline:

Pipelining is an implementation technique in which multiple instructions are overlapped in execution.
Today, pipelining is nearly universal.

Main Course:
Pipelining in a Processor


Instruction exc^n:
I1: Add
Destination
R1
R2, R3
If I1
ID & Read Register
Ex (ALU) R2+R3
Access Data memory
Write Register R1

I_2: lw r_1, 2(r_2)

Load word

Desti. ↑ source

$r_1 \leftarrow M[2 + r_2]$

Data memory

Address of Operand (Data)

If I_2

ID, Read Reg r_2

ALU $r_2 + 2 \longrightarrow$ Address

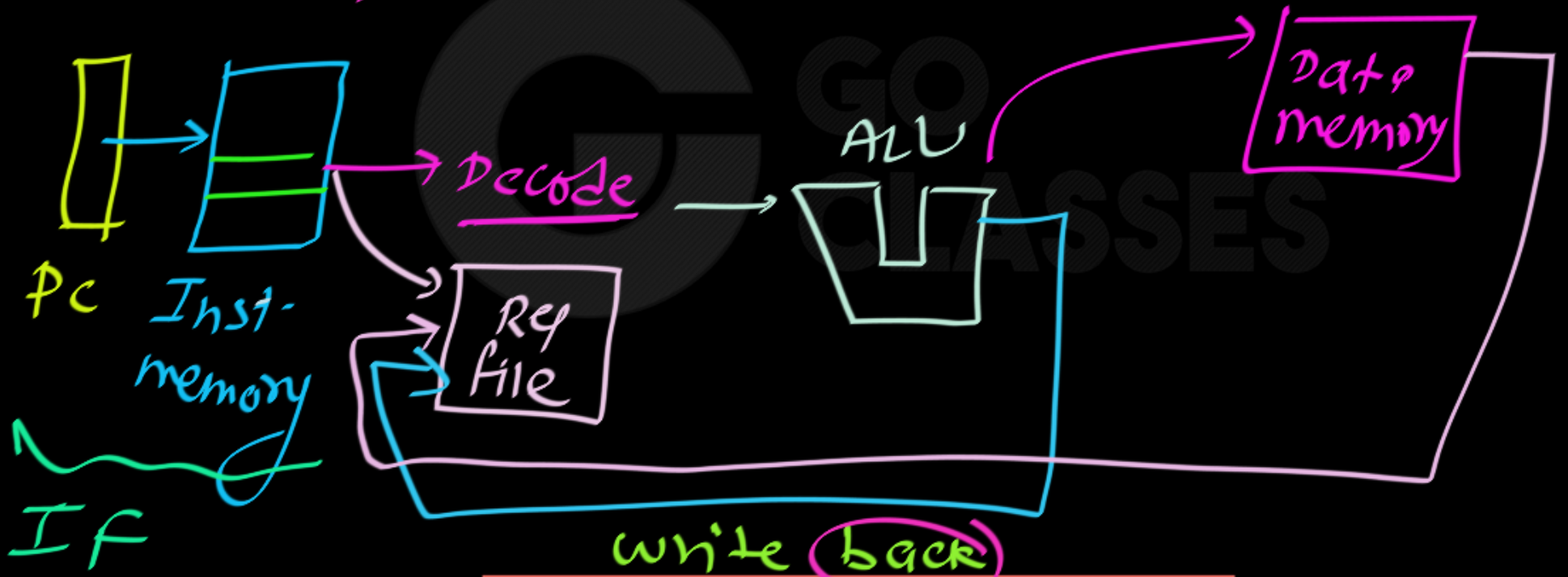
Access Data memory $\longrightarrow M[r_2 + 2]$

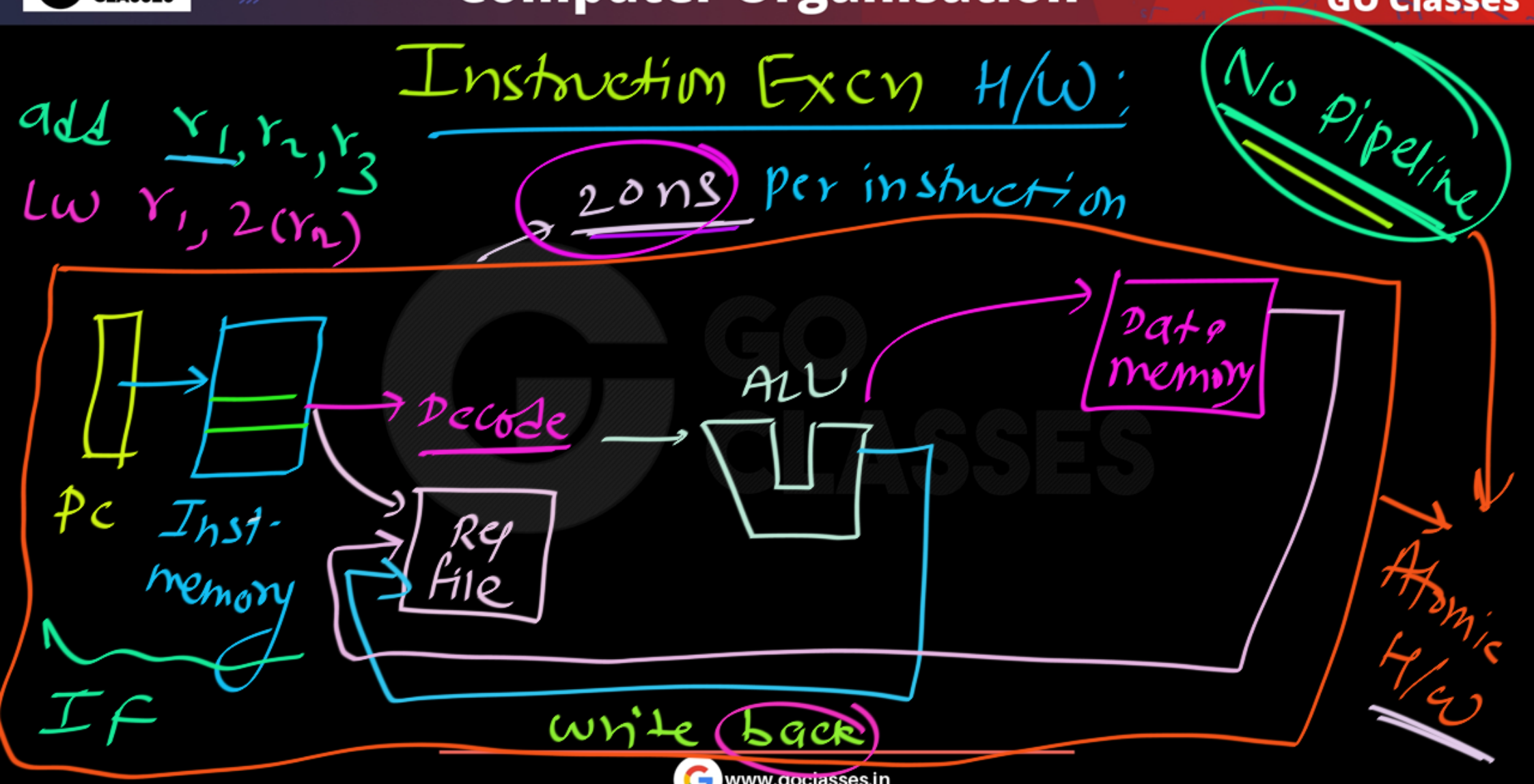
write back in Register (r_1)

Instruction Excn H/W:

add r_1, r_2, r_3

lw $r_1, 2(r_2)$




Instruction Excn H/W:
No pipeline
add r1, r2, r3
Lw r1, 2(r2)
20ns per instruction
Decode
ALU
Data memory
Reg file
Pc
Inst-memory
If
write back
Atomic H/W

Simple Cycle Execution

≡ Non-Pipeline Exᶜʸ

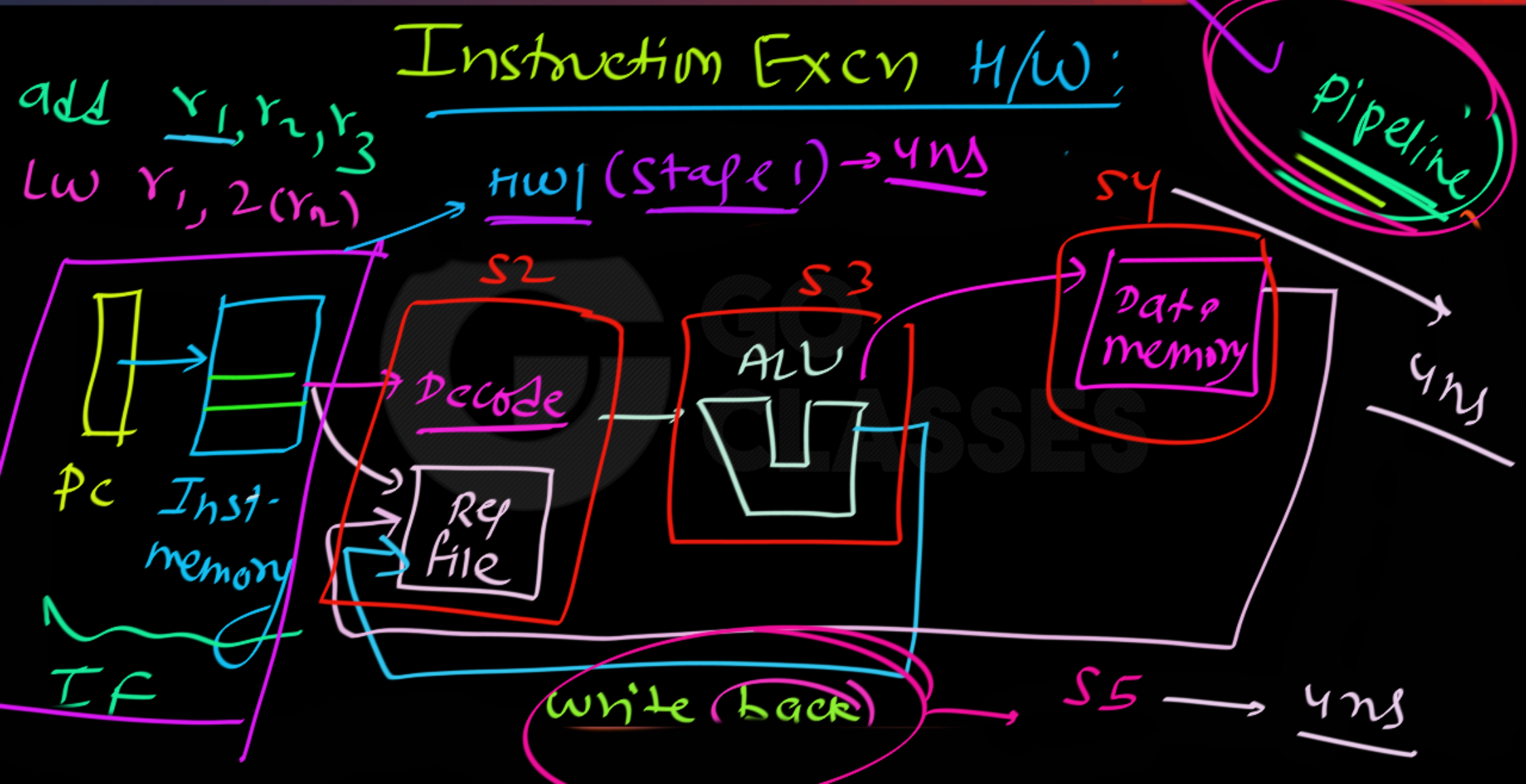
one cycle time : 20 ns

100 instructions

100 * 20 ns

= 2000 ns

GO CLASSES

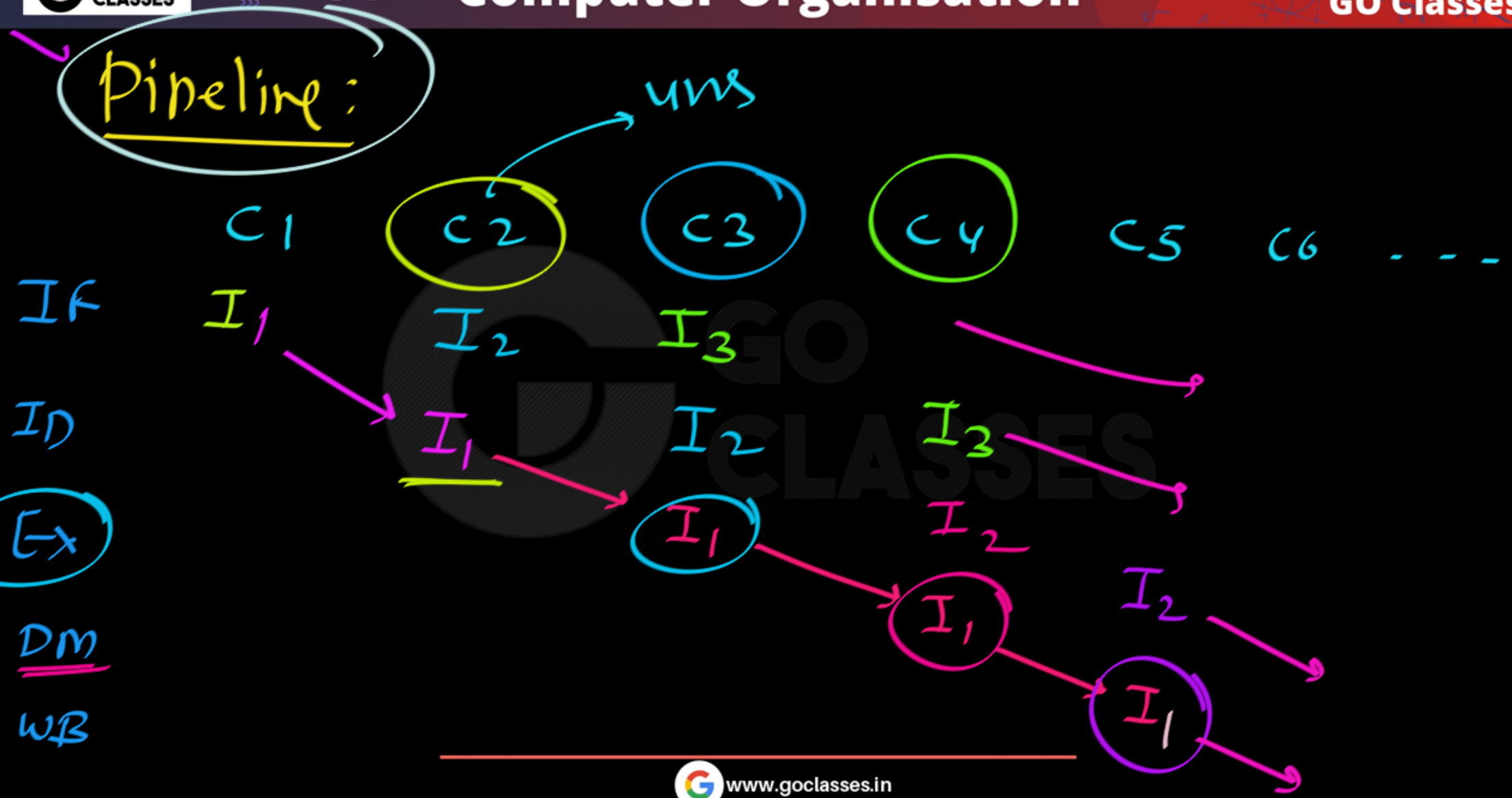

Instruction Excn H/W:
add r1, r2, r3
Lw r1, 2(r2)
HW1 (stage 1) → 4ns
S2
S3
S4
Decode
ALU
Data memory
Reg file
Pc
Inst-memory
IF
write back
S5 → 4ns
4ns
Pipeline

100 Inst;

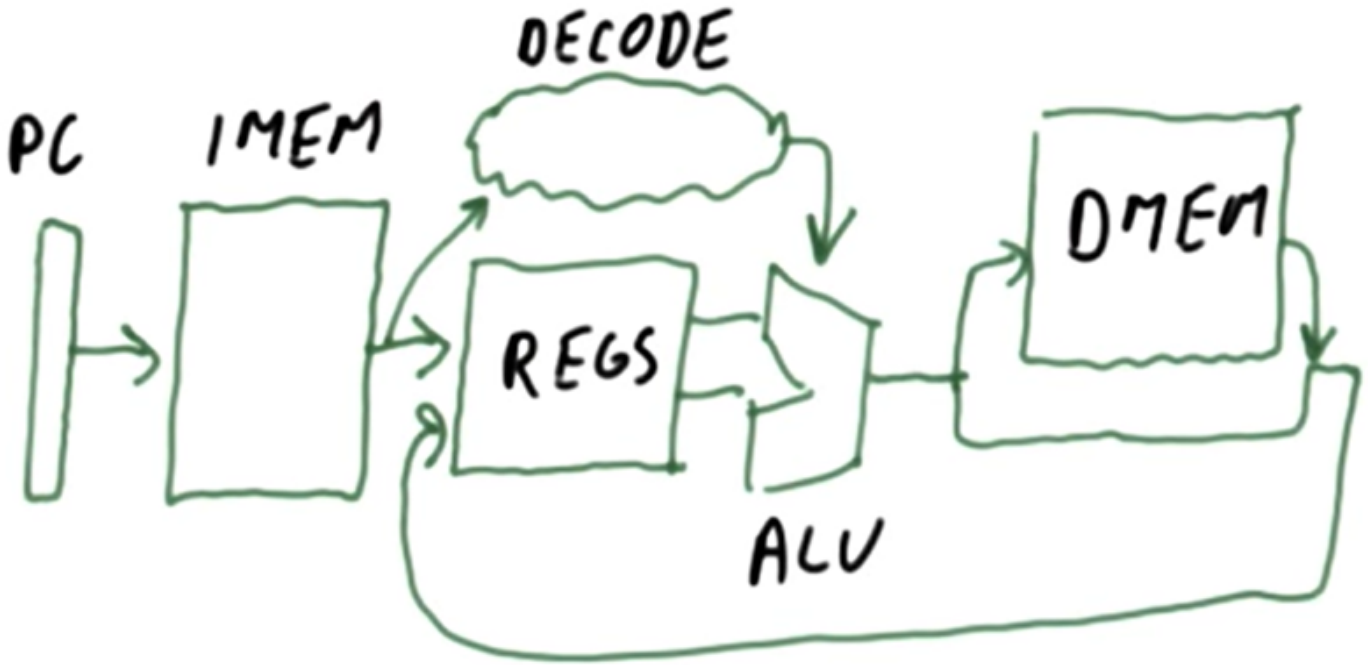
Pipelined Processor;

20 + 99 (4 ns) = 416 ns

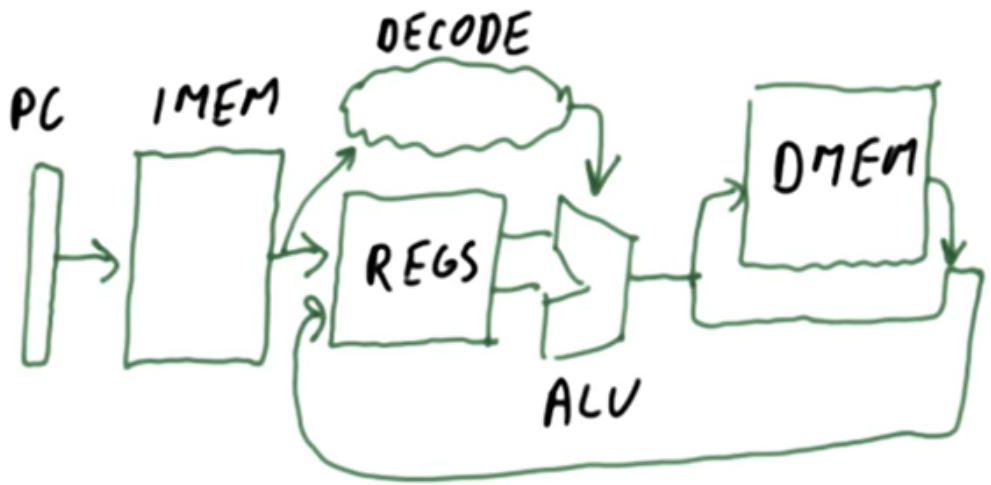
1st inst


Pipeline :
uns
C1
C2
C3
C4
C5
C6
- - -
IF
ID
Ex
DM
WB
I1
I2
I3
I1
I2
I3
I1
I2
I1
I2
I1

PIPELINING IN A PROCESSOR



PIPELINING IN A PROCESSOR



	F	D	A	M	W
C1	I1				
C2	I2	I1			
C3	I3	I2	I1		
C4	I4	I3	I2	I1	
	I5	I4	I3	I2	I1

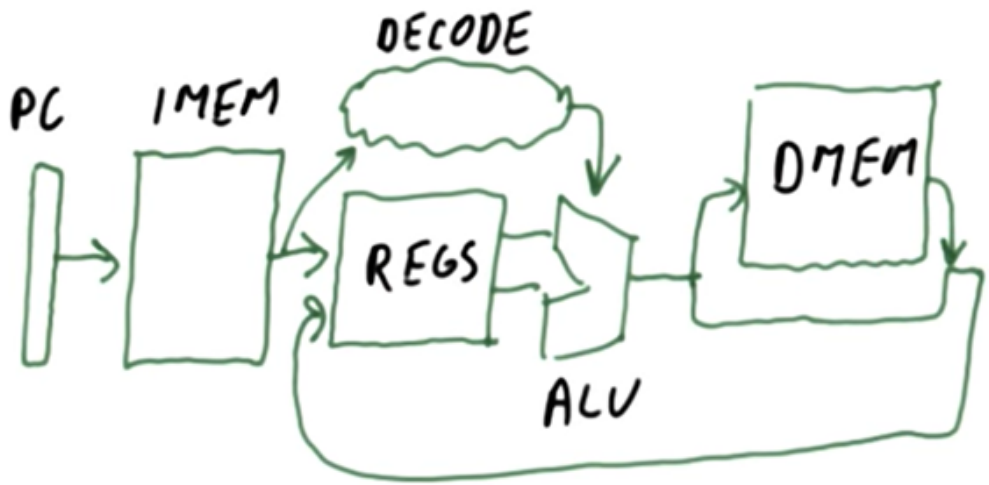
4ns

20ns

LATENCY : 20ns/INST

THROUGHPUT : $\frac{1\ \text{INST}}{4\text{ns}}$

PIPELINING IN A PROCESSOR



	F	D	A	M	W
C1	I1				
C2	I2	I1			
C3	I3	I2	I1		
C4	I4	I3	I2	I1	
	I5	I4	I3	I2	I1

20 ns

4ns

20ns

LATENCY : 20ns/INST

THROUGHPUT : $\frac{1\ \text{INST}}{4\text{ns}}$

INSTRUCTION PIPELINING QUIZ

- 5-STAGE PIPELINE (1 CLOCK CYCLE / STAGE)
- 10 INSTRUCTIONS IN OUR PROGRAM

NO PIPELINING : _____ CYCLES

WITH PIPELINING : _____ CYCLES

INSTRUCTION PIPELINING QUIZ

- 5-STAGE PIPELINE (1 CLOCK CYCLE / STAGE)
- 10 INSTRUCTIONS IN OUR PROGRAM

NO PIPELINING : 10×5 CYCLES

WITH PIPELINING : $5 + 9(1)$ CYCLES $= 14$ clock cycles

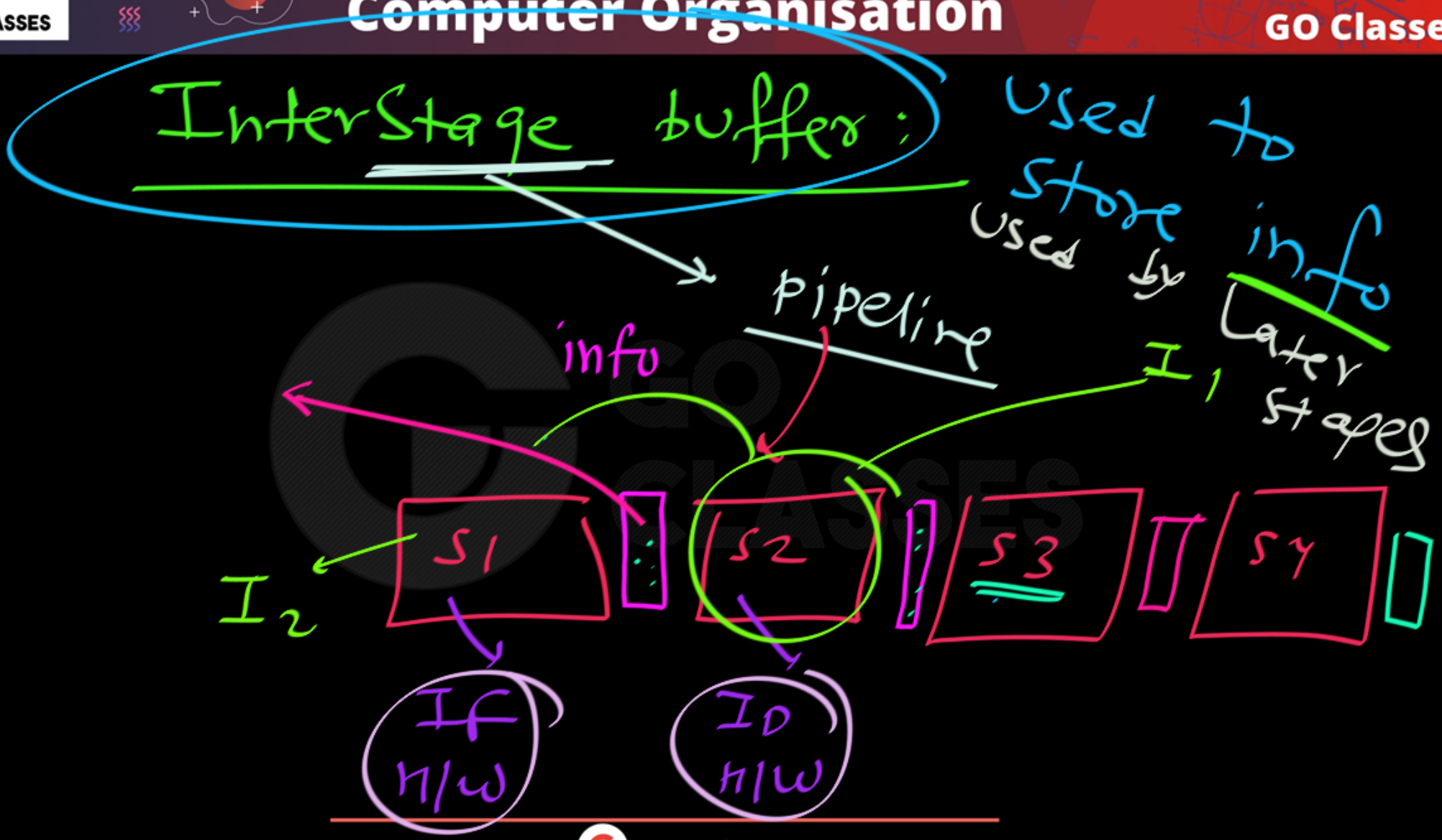
1st Inst

No Concept of Stages

Savings for 10 Inst Exen:

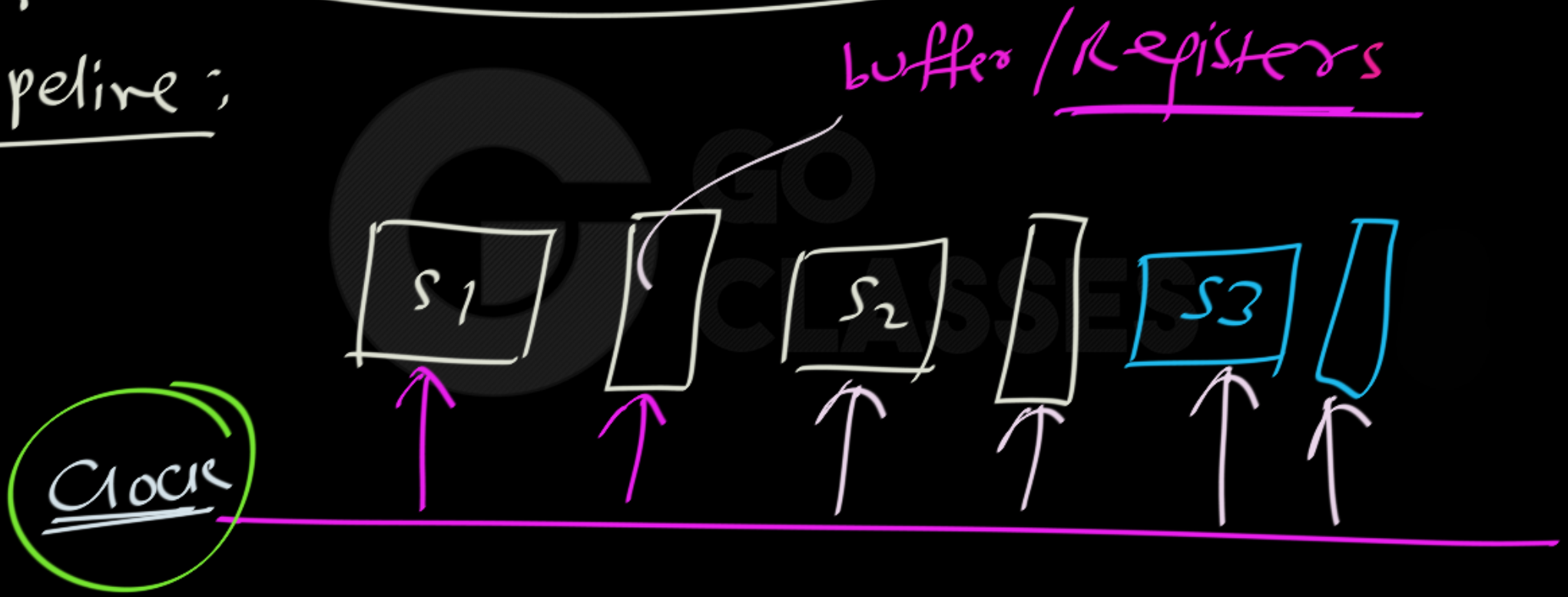
50 – 14 Clock cycles

GO CLASSES


InterStage buffer :
Used to store info Used by later stages
pipeline
info
I1
I2
S1
S2
S3
S4
If H/W
ID H/W

InterStage buffer :

Pipeline :

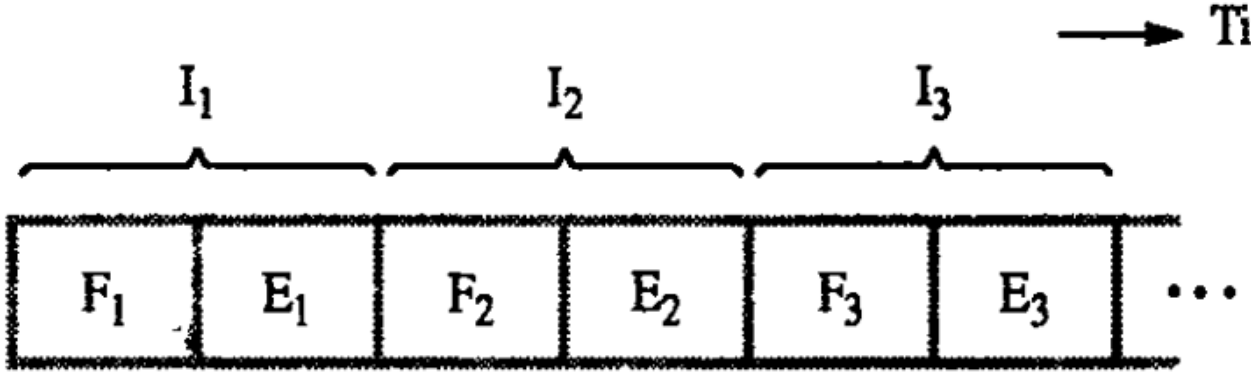


Next Example:

GO CLASSES

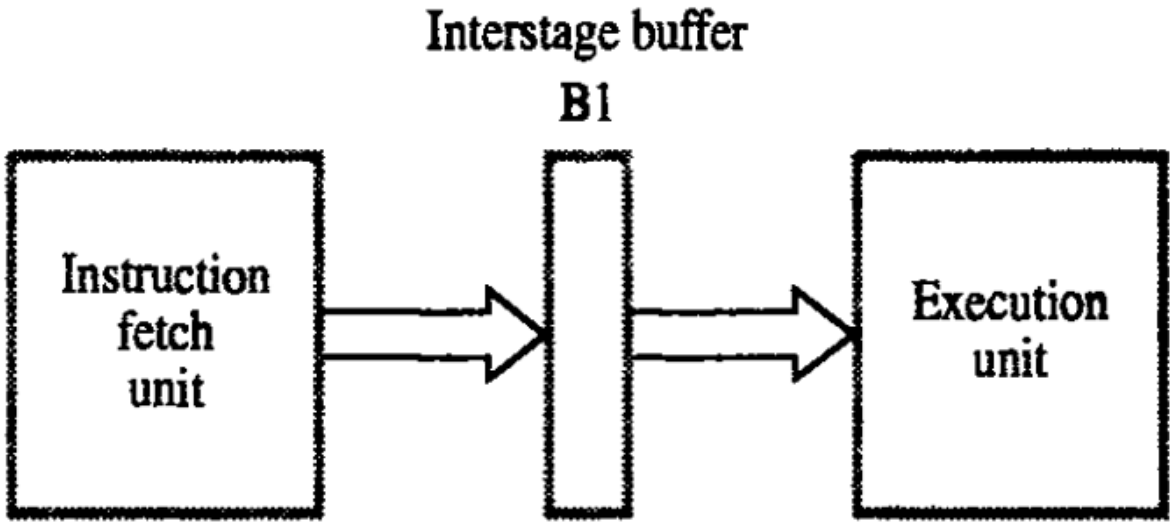
Consider how the idea of pipelining can be used in a computer. The processor executes a program by fetching and executing instructions, one after the other. Let F_i and E_i refer to the fetch and execute steps for instruction I_i. Execution of a program consists of a sequence of fetch and execute steps, as shown in Figure 8.1*a*.

Now consider a computer that has two separate hardware units, one for fetching instructions and another for executing them, as shown in Figure 8.1*b*. The instruction fetched by the fetch unit is deposited in an intermediate storage buffer, B1. This buffer is needed to enable the execution unit to execute the instruction while the fetch unit is fetching the next instruction. The results of execution are deposited in the destination



(a) Sequential execution

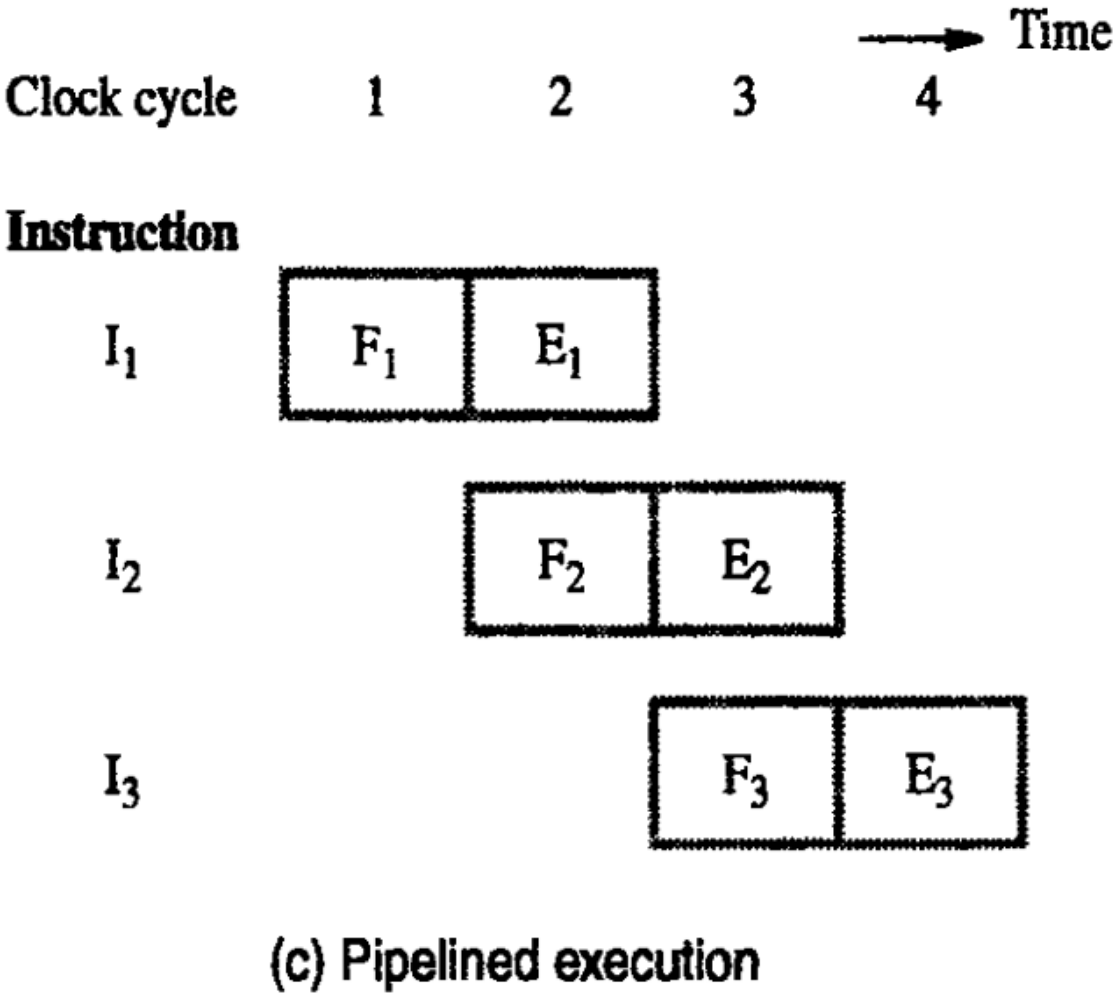
2 Stage Pipelining



(b) Hardware organization

GO CLASSES

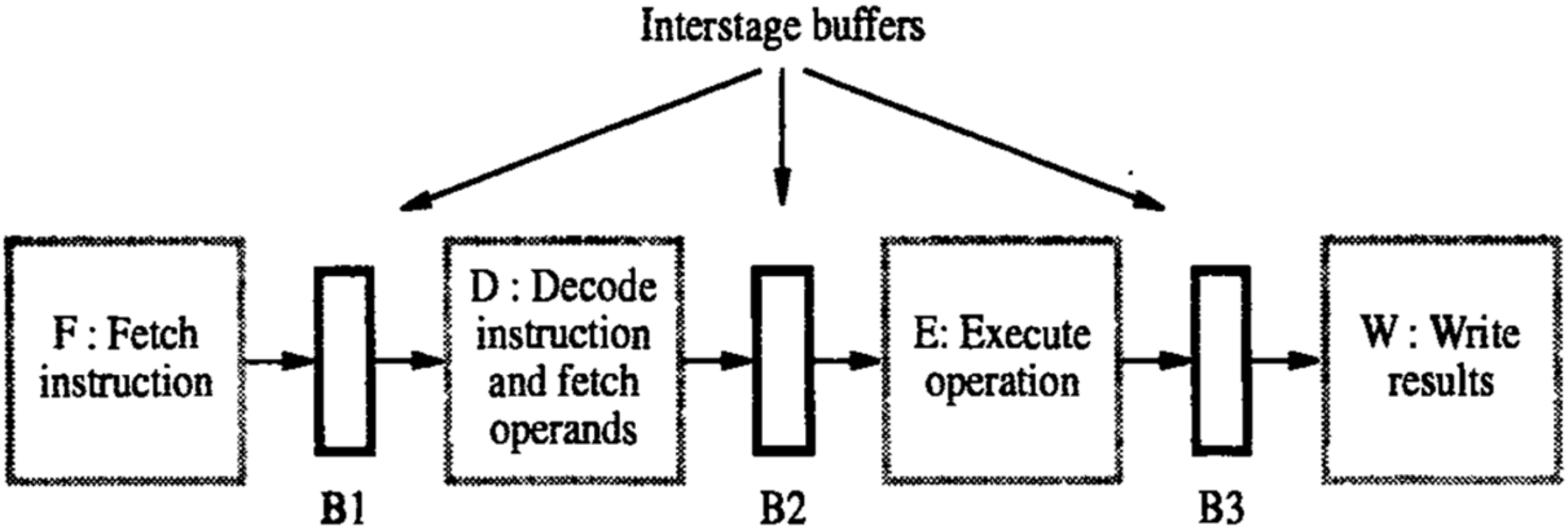
GO Classes



(c) Pipelined execution

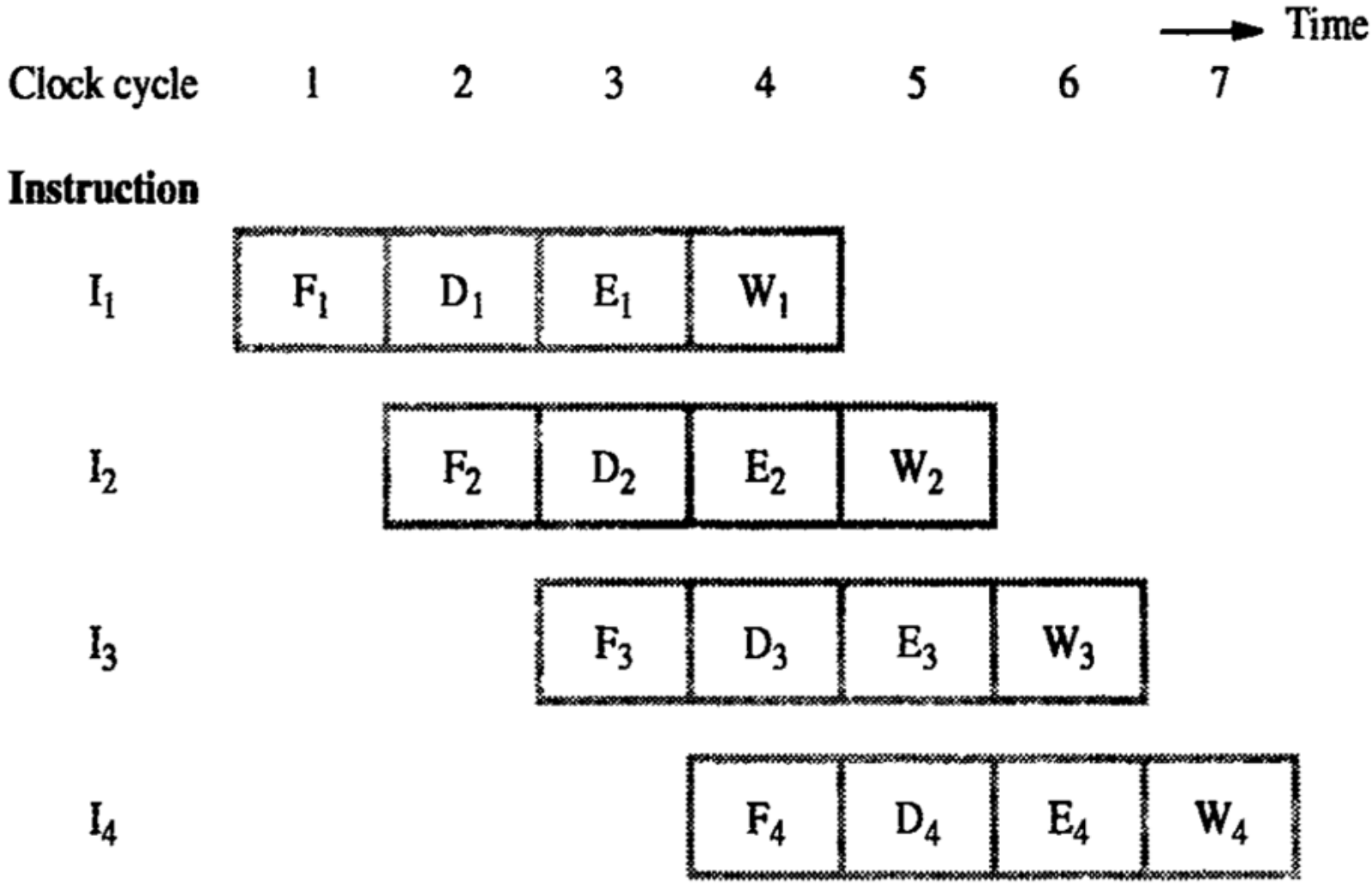
Figure 8.1 Basic idea of instruction pipelining.

Example:
4 Stage Pipelining



(b) Hardware organization

Figure 8.2 A 4-stage pipeline.



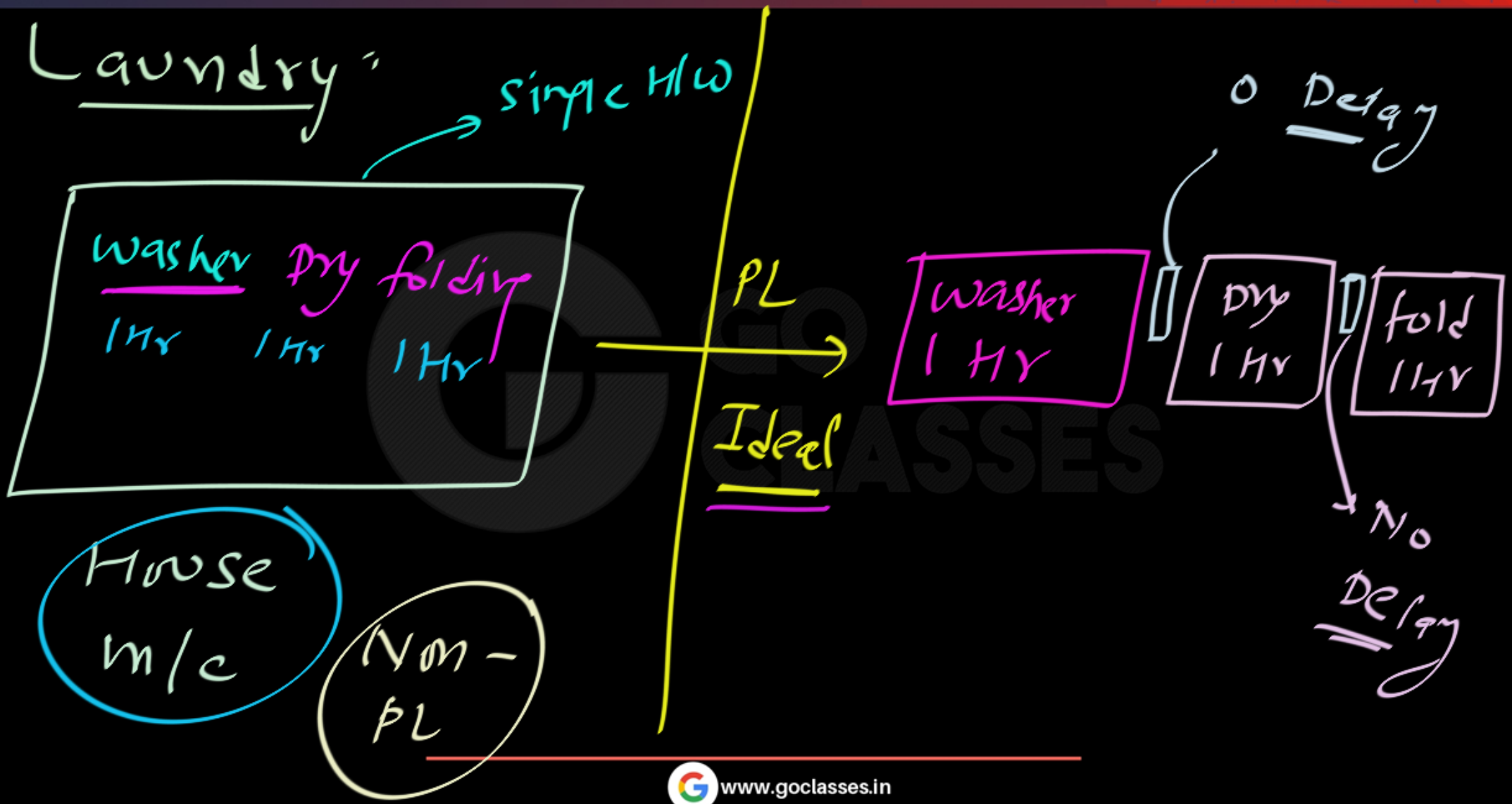
(a) Instruction execution divided into four steps

Performance Analysis of Pipeline:

~ 50% GATE PYQ

1. Performance:

In an IDEAL World


Laundry:
Single H/W
Washer
1 Hr
Dry
1 Hr
folding
1 Hr
PL
Ideal
Washer
1 Hr
Dry
1 Hr
fold
1 Hr
0 Delay
No
Delay
House
m/c
Non-
PL

Non-PL CPU ≡ Simple cycle execution of Instruction

IDEAL World:

Non-PL CPU:

t_n per instruction

non-PL

cycle time of Non-PL CPU

PL CPU → k stages;

$\frac{t_n}{k}$ time per stage

cycle time of PL CPU

1 Inst Exec^n time :

Non-PL : t_n

PL : $\left(\frac{t_n}{k}\right) k = t_n$ } Same (Ideal world)

y Inst Exc^n time :

Non-PL : $y * t_n$ → per instruction

PL : $\underbrace{k}_{\text{for } k \text{ cycles inst}} + \underbrace{y-1}_{\text{Remaining inst.}} * \frac{t_n}{k}$ → cycle time

GO CLASSES

A lot of intructions :

Speedup ; $\frac{NPL}{PL} = \frac{y\,t_n}{(k+y-1)\frac{t_n}{k}}$

$= k$

k → #Stages

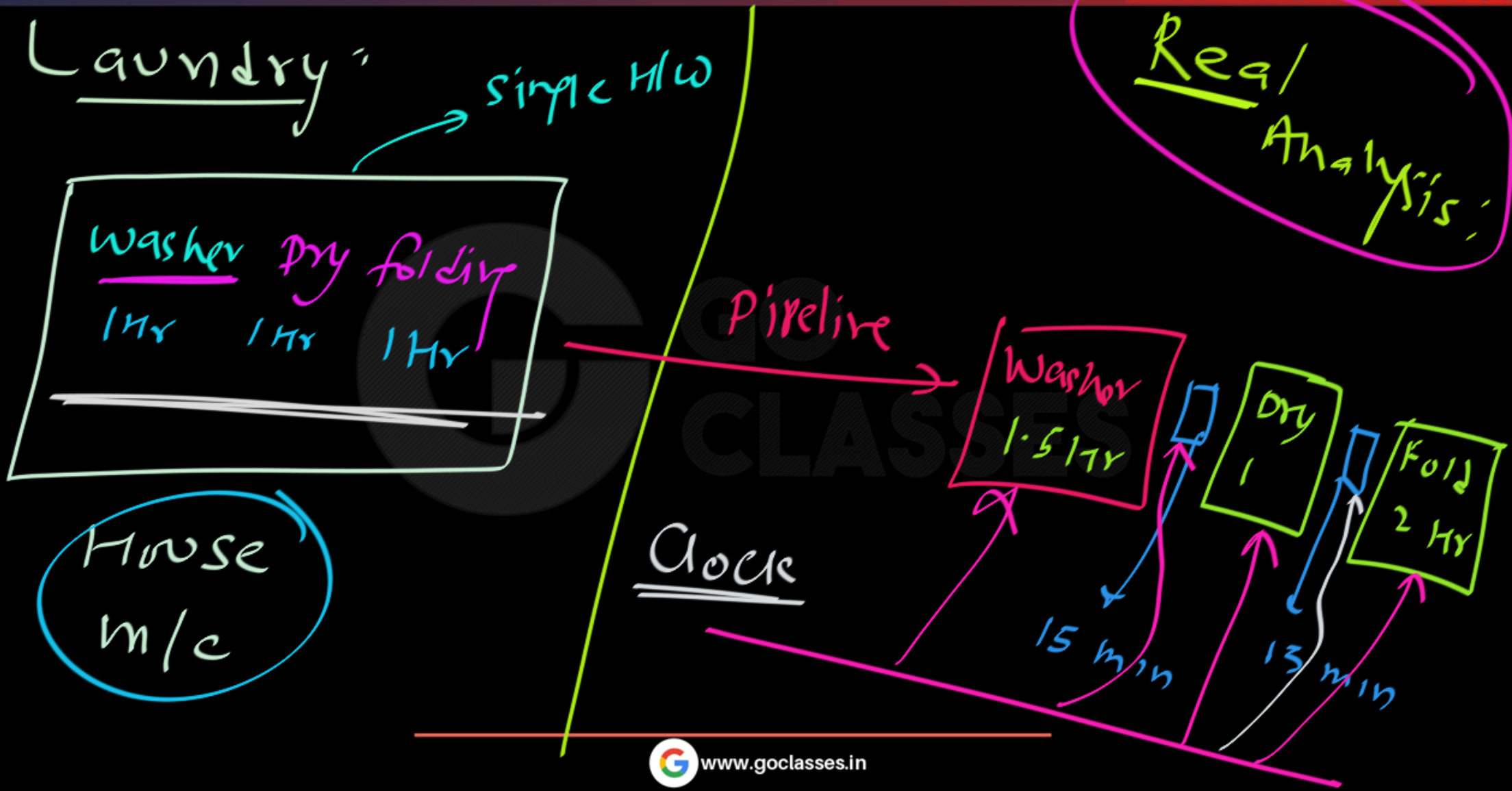
Hope

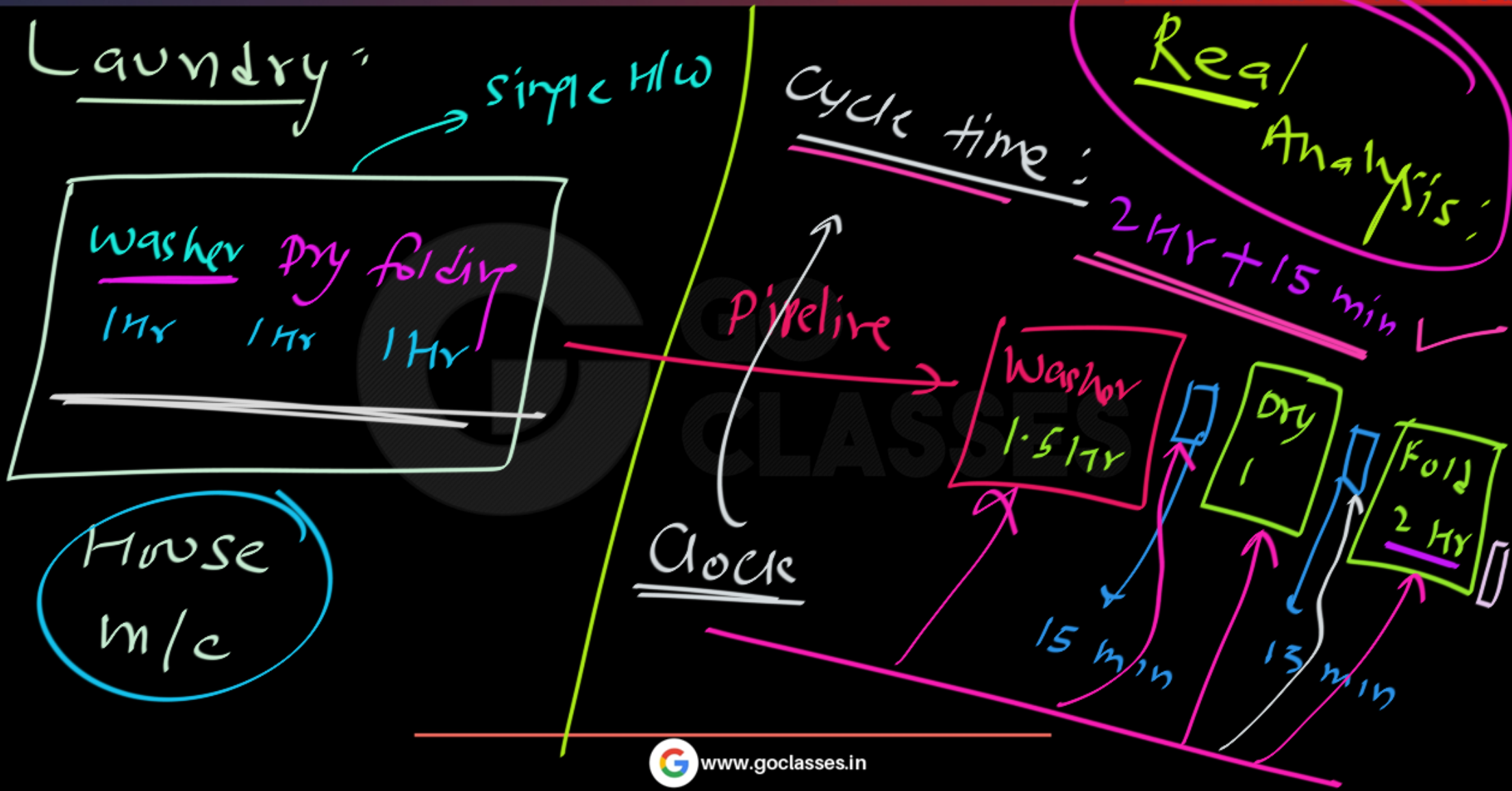
Note: Maximum Speedup we Can achieve with Pipeline over Non-PL = #stages in pipeline

IDEAL

Note:

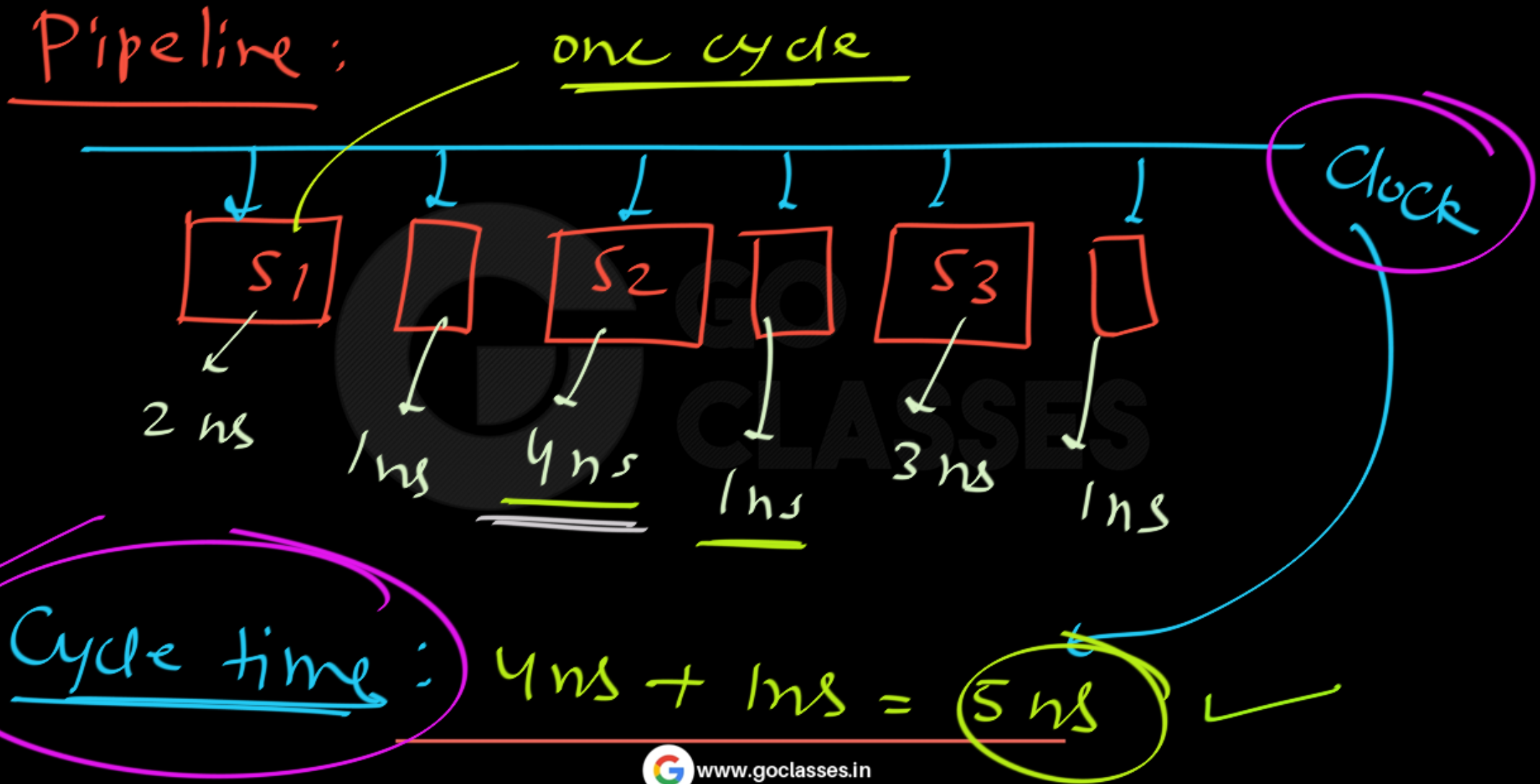
Speedup, by default, find for a large no. of instructions.


Laundry:
Single H/W
Washer
Dry
folding
1 Hr
1 Hr
1 Hr
House m/c
Real Analysis:
Pipeline
Clock
Washer
1.5 Hr
Dry
1
Fold
2 Hr
15 min
13 min


Laundry:
Simple H/W
Washer
Dry
folding
1 Hr
1 Hr
1 Hr
House
M/c
Cycle time:
2 Hr + 15 min
Real
Analysis:
Pipeline
Clock
Washer
1.5 Hr
Dry
1
Fold
2 Hr
15 min
15 min

2. Performance:

In Slightly Real World


Pipeline :
one cycle
Clock
S1
S2
S3
2 ns
1 ns
4 ns
1 ns
3 ns
1 ns
Cycle time :
4 ns + 1 ns = 5 ns

1.18.7 Pipelining: GATE CSE 2004 | Question: 69 top

https://gateoverflow.in/1063

A 4-stage pipeline has the stage delays as 150, 120, 160 and 140 *nanoseconds*, respectively. Registers that are used between the stages have a delay of 5 *nanoseconds* each. Assuming constant clocking rate, the total time taken to process 1000 data items on this pipeline will be:

A. 120.4 microseconds

B. 160.5 microseconds

C. 165.5 microseconds

D. 590.0 microseconds

gate2004-cse | co-and-architecture | pipelining | normal

goclasses.in
tests.gatecse.in

Per stage time

1.18.7 Pipelining: GATE CSE 2004 | Question: 69

https://gateoverflow.in/1003

A 4-stage pipeline has the stage delays as 150, 120, 160 and 140 *nanoseconds*, respectively. Registers that are used between the stages have a delay of 5 *nanoseconds* each. Assuming constant clocking rate, the total time taken to process 1000 data items on this pipeline will be:

A. 120.4 microseconds
B. 160.5 microseconds
C. 165.5 microseconds
D. 590.0 microseconds

cycle time : 160 + 5 = 165 ns

1000 inst : 4 cycles + 999 (1 cycle)

1st Inst

Remaining

gate2004-cse | co-and-architecture | pipelining | normal

goclasses.in
tests.gateose.in

4 + 999 cycles needed for 1000 inst.

time: 1003×165 ns

$$= \frac{10003 \times 165 \times \mu s}{1000}$$

courses.engr.illinois.edu/cs433/fa2020/assignments/cs433-fa20-hw2-solutions.pdf
cs433-fa20-hw2-solutions.pdf 1 / 6 153%

Problem 1 [5 points]

Consider two different machines. The first has a single cycle datapath (i.e., a single stage, non-pipelined machine) with a cycle time of 15 ns. The second is a pipelined machine with 5 pipeline stages and a cycle time of 3ns.

Part (A) [1 point]

What is the speedup of the pipelined machine versus the single cycle machine assuming there are no stalls?

courses.engr.illinois.edu/cs433/fa2020/assignments/cs433-fa20-hw2-solutions.pdf

cs433-fa20-hw2-solutions.pdf 1 / 6 153%

Problem 1 [5 points]

1 instruction

Consider two different machines. The first has a single cycle datapath (i.e., a single stage, non-pipelined machine) with a cycle time of 15 ns. The second is a pipelined machine with 5 pipeline stages and a cycle time of 3ns.

Part (A) [1 point]

What is the speedup of the pipelined machine versus the single cycle machine assuming there are no stalls?

IDEAL

n instructions :

Non PL : $n \times 15$ ns

PL :

$$(5 \text{ cycles} + (n-1) \text{ 1 cycle}) * 3 \text{ ns}$$

5 cycles → 1st inst

$(n-1)$ → Remaining

Speedup

$$= \frac{\text{NPL exec}^n \text{ time}}{\text{PL exec}^n \text{ time}}$$

$$= \frac{n \times 15 \text{ ns}}{(5 + n - 1)\, 3 \text{ ns}}$$

Speedup $= 5$

large no. of instr

Huge

courses.engr.illinois.edu/cs433/fa2020/assignments/cs433-fa20-hw2-solutions.pdf

Problem 1 [5 points]

Consider two different machines. The first has a single cycle datapath (i.e., a single stage, non-pipelined machine) with a cycle time of 15 ns. The second is a pipelined machine with 5 pipeline stages and a cycle time of 3ns.

Part (A) [1 point]

What is the speedup of the pipelined machine versus the single cycle machine assuming there are no stalls?

Solution: The speedup is $^{15\ ns}/_{3\ ns} = 5$.

ideal

Part (C) [2 points]

Now consider a 4 stage pipeline machine with a cycle time of 3.1 ns. Again, assuming no stalls, is this implementation faster or slower than the original 5 stage pipeline? Explain your answer.

www.goclasses.in

PL 1

5 stages

3 ns cycle time

better

PL 2

4 stages

3.1 cycle time

for one instruction :

PL1 : 15 ns

PL2 : 12.4 ns

PL 1

5 stages

3 ns cycle time

← better

PL 2

4 stages

3.1 cycle time

A lot of instructions :

PL 1 : 3 ns per inst

PL 2 : 3.1 ns per inst

courses.engr.illinois.edu/cs433/fa2020/assignments/cs433-fa20-hw2-solutions.pdf

cs433-fa20-hw2-solutions.pdf 1 / 6 153%

Problem 1 [5 points]

Consider two different machines. The first has a single cycle datapath (i.e., a single stage, non-pipelined machine) with a cycle time of 15 ns. The second is a pipelined machine with 5 pipeline stages and a cycle time of 3ns.

Part (A) [1 point]

What is the speedup of the pipelined machine versus the single cycle machine assuming there are no stalls?

Part (C) [2 points]

Now consider a 4 stage pipeline machine with a cycle time of 3.1 ns. Again, assuming no stalls, is this implementation faster or slower than the original 5 stage pipeline? Explain your answer.

Solution: The 5 stage machine is faster. This is because it has a smaller cycle time, which results in a faster overall execution time (since there are no stalls, they both have the same CPI).

33. Two processors, M-5 and M-7, implement the same instruction set. Processor M-5 uses a 5-stage pipeline and a clock cycle of 10 nanoseconds. Processor M-7 uses a 7-stage pipeline and a clock cycle of 7.5 nanoseconds. Which of the following is (are) true?

I. M-7's pipeline has better maximum throughput than M-5's pipeline.

II. The latency of a single instruction is shorter on M-7's pipeline than on M-5's pipeline.

III. Programs executing on M-7 will always run faster than programs executing on M-5.

(A) I only (B) II only (C) I and III only (D) II and III only (E) I, II, and III

GRE CS

For large no. of inst.

33. Two processors, M-5 and M-7, implement the same instruction set. Processor M-5 uses a 5-stage pipeline and a clock cycle of 10 nanoseconds. Processor M-7 uses a 7-stage pipeline and a clock cycle of 7.5 nanoseconds. Which of the following is (are) true?

I. M-7's pipeline has better maximum throughput than M-5's pipeline.

II. The latency of a single instruction is shorter on M-7's pipeline than on M-5's pipeline.

III. Programs executing on M-7 will always run faster than programs executing on M-5.

(A) I only (B) II only (C) I and III only (D) II and III only (E) I, II, and III

M-5

5 stages

10 ns cycle time

better

M-7

7 stages

7.5 ns cycle time

One inst Latency :

M-5 : 5 x 10 ns = 50 ns

M-7 : 7 x 7.5 ns > 50 ns

Throughput: no. of work done per unit of time.

Insts executes per unit of time.

M-5

5 stages

10 ns cycle time

M-7

7 stages

7.5 ns cycle time

n inst:

$(5+n-1)10$ ns

$(7+n-1)7.5$ ns

Throughput: $\frac{n}{(5+n-1)10}$

Throughput: $\frac{n}{(7+n-1)7.5}$

www.goclasses.in

m-5
Throughput

$$\frac{n}{(5+n-1)\,10}$$

$\frac{1}{10}$

m-7
Throughput ✓

$$\frac{n}{(7+n-1)\,7.5}$$

Large no. of inst

$\frac{1}{7.5}$

GO CLASSES