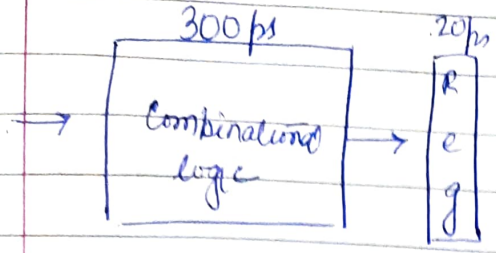
Measure of throughput Giga Instructions Per



Latency = 320 ps

Throughput 320×10^{-12} secs — 1 instruction

1 sec — $\frac{1 \times 10^{12}}{320}$

$= \frac{100}{32} \times 10^9$

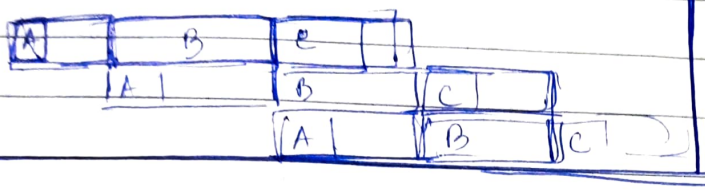
$= 3.12 \times 10^9$ instructions

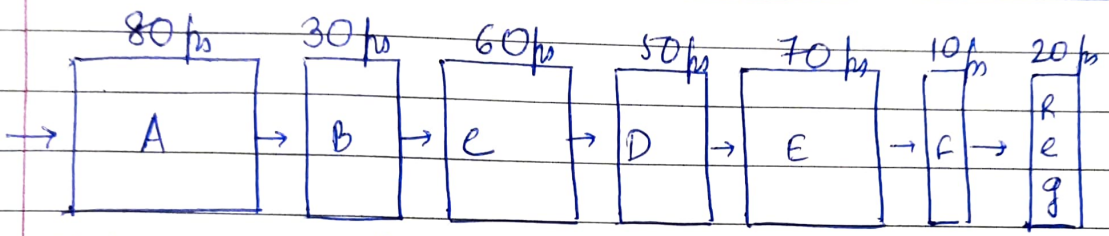
= 3.12 GIPS

(A)

Balanced pipeline

(C)





Pipeline register has a delay of 20 ps.

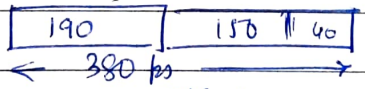
Where do we insert a single pipeline register to have a 2 stage pipeline? ABC [Reg] → DEF [Reg]

(170+20) (130) (20)

2 stage pipeline

(B)

latency



latency 380ps

Throughput 190×10^{-12} pico second — 1 instruction

1 second — $\frac{10^{12}}{190} = \frac{100 \times 10^9}{19}$

$= 5.26$ GIPS

3 stage pipeline

AB [R] CD [R] EF [R]

(110+20) (110+20) (100)

130ps 130ps

390 ps latency

$\frac{10^{12}}{130} = \frac{100}{13} = 7.6$

4 stage pipeline

A BC D EF

100 110 70 110

440ps $\frac{100}{11} = 9.09$

① balanced pipeline
② diminishing return of balanced pipeline

	A	B	C	D	EF
5 stage pipeline	100	50	80	70	100

500 ps

$\frac{10^{12}}{100}$ $\frac{10^9 \times 10}{}$ 10 GIPS

bounded by the slowest stage

$\frac{300}{k}$ each pipeline register 20 ps

(D) Latency $k \times \left(\frac{300}{k} + 20\right)$

$k \to \infty$ each pipeline stage $\left(\frac{300}{k} + 20\right)$ ps

infinite latency $\frac{10^{12}}{\frac{300}{k} + 20}$ $\frac{10^{12}}{20}$ $\frac{10^9 \times 100}{2}$

50 GIPS

(E)

Operation	Integer Latency	Issue	Capacity	Floating Point Latency	Issue	Capacity
Addition	1	1	4	3	1	1
Multiplication	3	1	1	5	1	2
Division	3-30	3-30	1	3-15	3-15	1

Intel core i7

one to process exponent value
1 to add fractions
1 to round the result

(F)

$a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots a_n x^n$

```
double poly (double a[], double x, long degree)
{ long i;
  double result = a[0];
  double xpwr = x;
  for(i=1; i<=degree; i++) {
     result += a[i] * xpwr
     xpwr = x * xpwr;
  }
  return result.
```

n multiplications
n additions

~~$a_0 + a_1 (x + a_2 (x$~~

$a_0 + x\,(a_1 + x\,(a_2 + x\,(a_3 + x\,(a_4 + \ldots x\,(a_{n-1} + x a_n$

```
double poly (double a[], double x, long degree)
{ long i;
  double result = a[degree];
  for (i = degree-1; i>=0; i--)
     result = a[i] + x* result;
  return result.
}
```