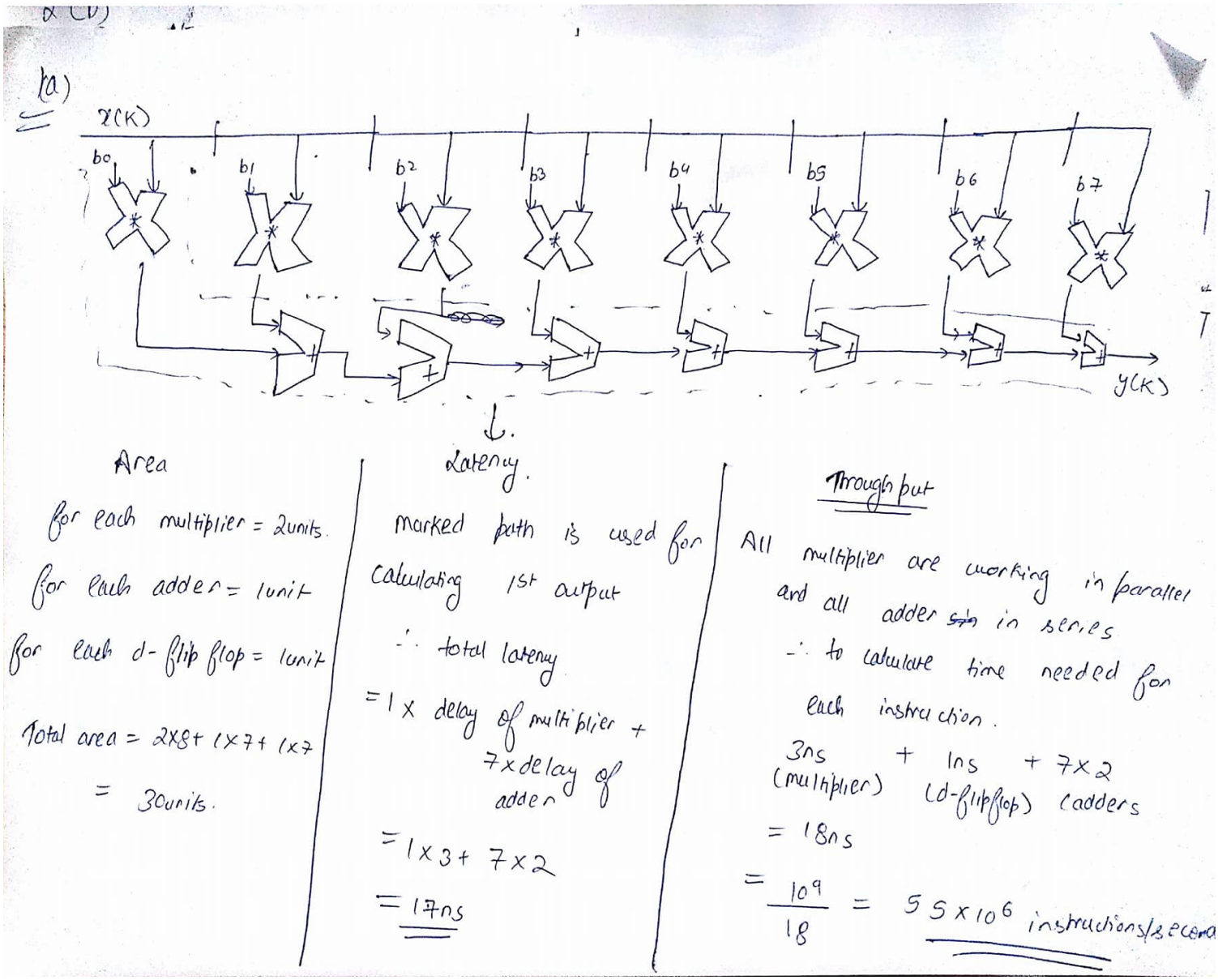
VLSI ASSIGNMENT – 3

Ayush Malpani(201530241)

1 a.)

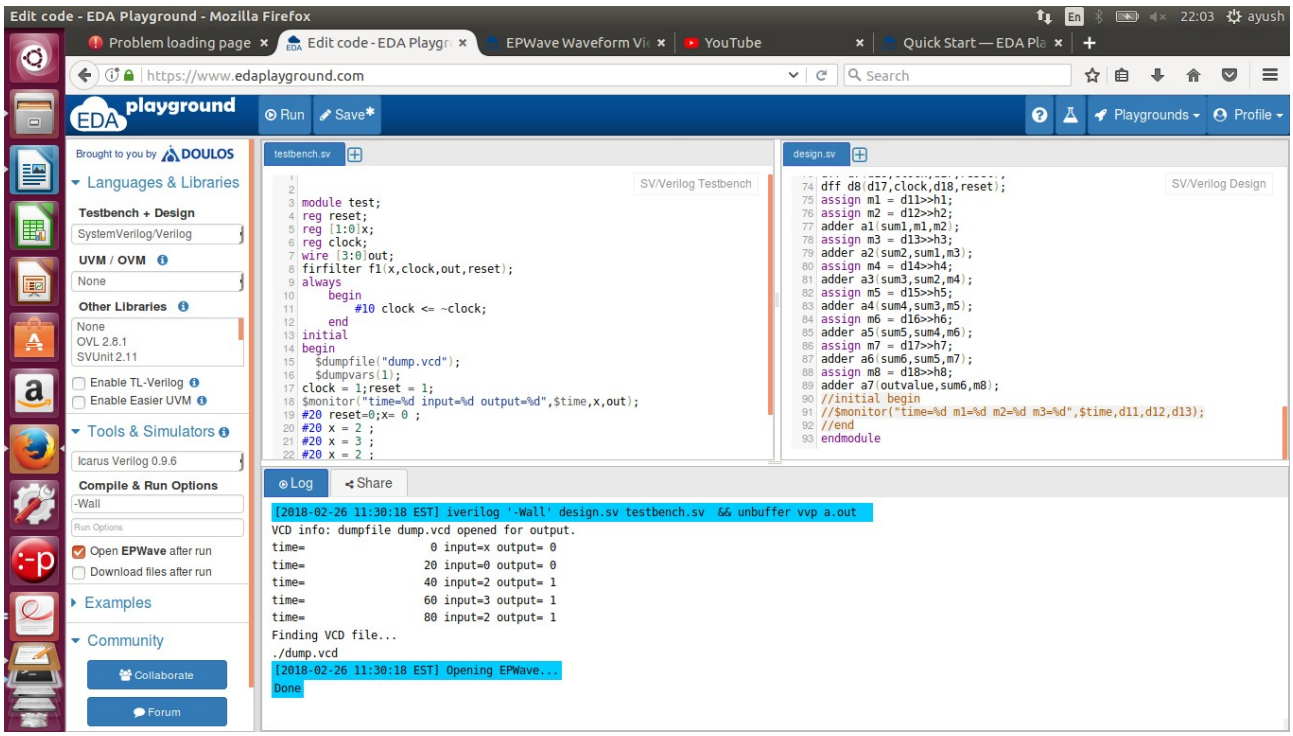
In this isomorphic architecture of FIR filter parallel adders and multipliers are used

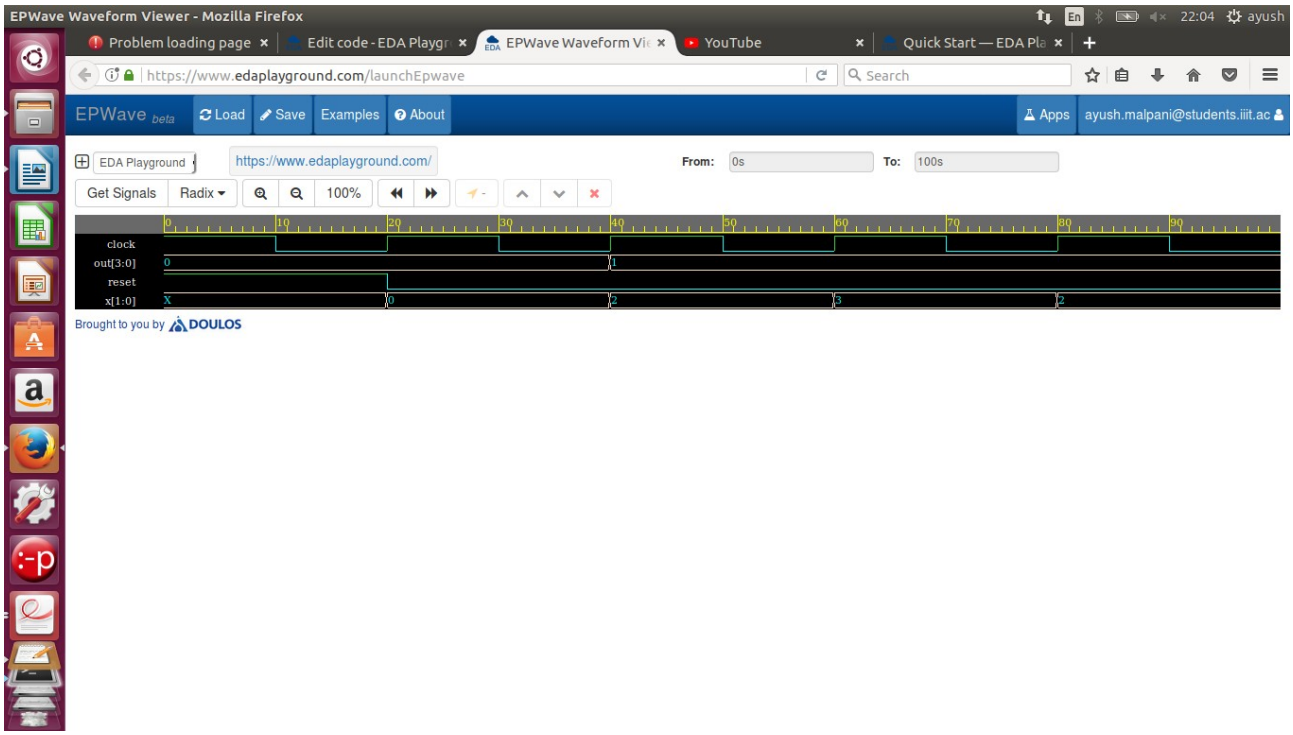


Total Area Req - 30 units

Latency – 17 ns

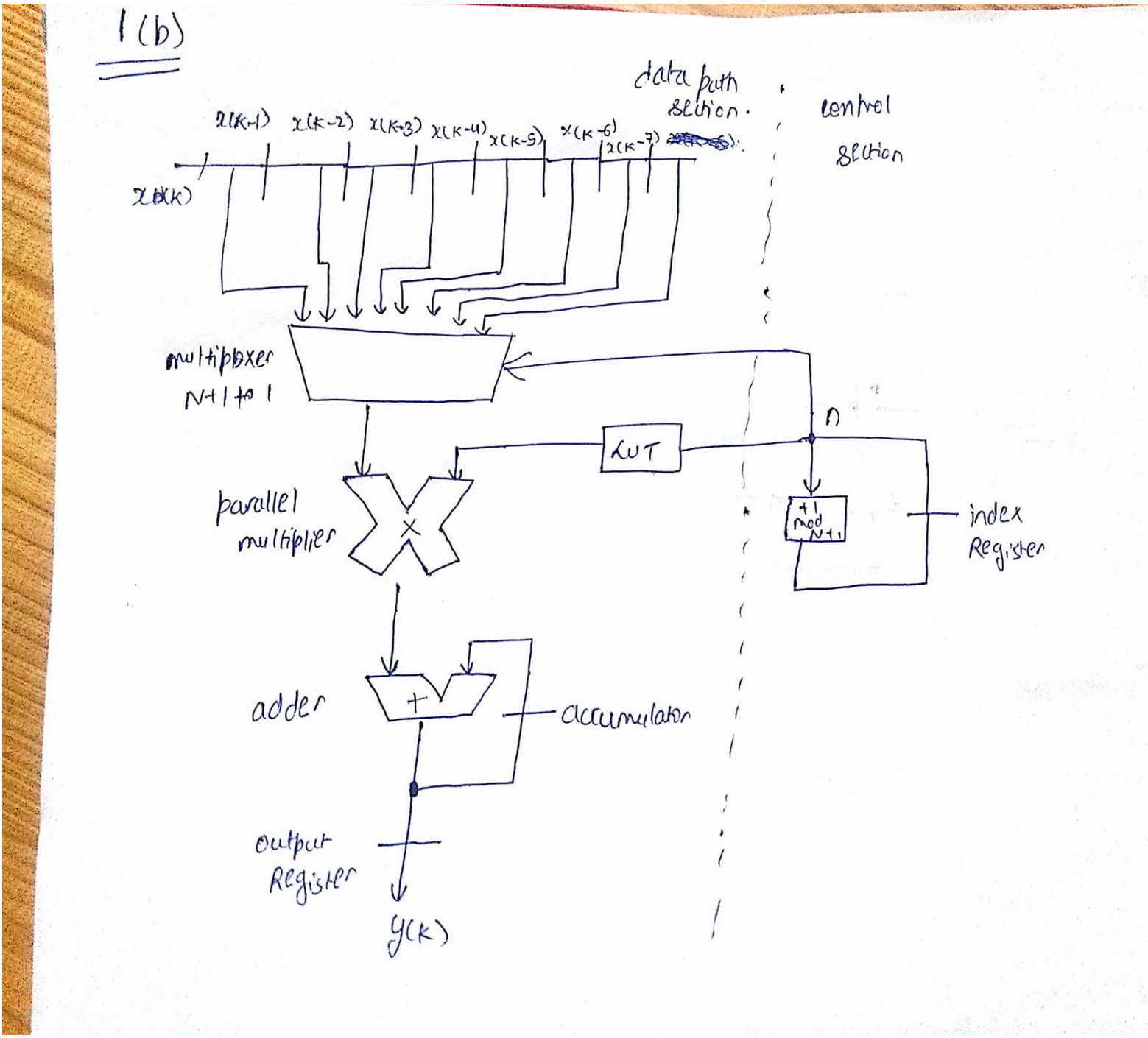
Throughput – 55 * 10^6 instructions/ second


Edit code - EDA Playground - Mozilla Firefox
Problem loading page
Edit code - EDA Playgr
EPWave Waveform Vie
YouTube
Quick Start — EDA Pla
https://www.edaplayground.com
Search
EDA playground
Run
Save*
Playgrounds
Profile
Brought to you by DOULOS
Languages & Libraries
Testbench + Design
SystemVerilog/Verilog
UVM / OVM
None
Other Libraries
None
OVL 2.8.1
SVUnit 2.11
Enable TL-Verilog
Enable Easier UVM
Tools & Simulators
Icarus Verilog 0.9.6
Compile & Run Options
-Wall
Run Options
Open EPWave after run
Download files after run
Examples
Community
Collaborate
Forum
testbench.sv
SV/Verilog Testbench
module test;
reg reset;
reg [1:0]x;
reg clock;
wire [3:0]out;
firfilter f1(x,clock,out,reset);
always
begin
#10 clock <= ~clock;
end
initial
begin
$dumpfile("dump.vcd");
$dumpvars(1);
clock = 1;reset = 1;
$monitor("time=%d input=%d output=%d",$time,x,out);
#20 reset=0;x= 0 ;
#20 x = 2 ;
#20 x = 3 ;
#20 x = 2 ;
design.sv
SV/Verilog Design
dff d8(d17,clock,d18,reset);
assign m1 = d11>>h1;
assign m2 = d12>>h2;
adder a1(sum1,m1,m2);
assign m3 = d13>>h3;
adder a2(sum2,sum1,m3);
assign m4 = d14>>h4;
adder a3(sum3,sum2,m4);
assign m5 = d15>>h5;
adder a4(sum4,sum3,m5);
assign m6 = d16>>h6;
adder a5(sum5,sum4,m6);
assign m7 = d17>>h7;
adder a6(sum6,sum5,m7);
assign m8 = d18>>h8;
adder a7(outvalue,sum6,m8);
//initial begin
//$monitor("time=%d m1=%d m2=%d m3=%d",$time,d11,d12,d13);
//end
endmodule
Log
Share
[2018-02-26 11:30:18 EST] iverilog '-Wall' design.sv testbench.sv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
time= 0 input=x output= 0
time= 20 input=0 output= 0
time= 40 input=2 output= 1
time= 60 input=3 output= 1
time= 80 input=2 output= 1
Finding VCD file...
./dump.vcd
[2018-02-26 11:30:18 EST] Opening EPWave...
Done


EPWave Waveform Viewer - Mozilla Firefox
Problem loading page
Edit code - EDA Playgr
EPWave Waveform Vie
YouTube
Quick Start — EDA Pla
https://www.edaplayground.com/launchEpwave
Search
EPWave beta
Load
Save
Examples
About
Apps
ayush.malpani@students.iiit.ac
EDA Playground
https://www.edaplayground.com/
From:
0s
To:
100s
Get Signals
Radix
100%
clock
out[3:0]
reset
x[1:0]
Brought to you by DOULOS

1 b.) Iteratively decomposed architecture with only one multiplier and one adder.

In this architecture mux is used to select one of the input and to give the required input to the multiplier and selecting the required multiplicand value from the lookup table.



Area

for mux = 1unit

for each multiplier = 2unit

for each adder = 1unit

for look up table = 2unit

for each mod circuit = 1unit

total area = 2 + 1 + 1 + 2 + 1

= 7units

Latency

for mux = 1ns

for multiplier = 3ns.

for adder = 2ns.

for rest components = 1ns.

= 8 × 1 + 8 × 3 + 8 × 2 + 8 × 1

(mux) (multiplier will be working in parallel) (+1 mod N+1)

= 8 + 24 + 16 + 8

= 48ns. + 8

= 56ns

Throughput

formux = 1ns

for multiplier = 3ns

for adder = 2ns.

for rest components = 1ns.

= 8 × 1 + 8 × 3 + 8 × 2 + 8 × 1

= 8 + 24 + 16 + 8

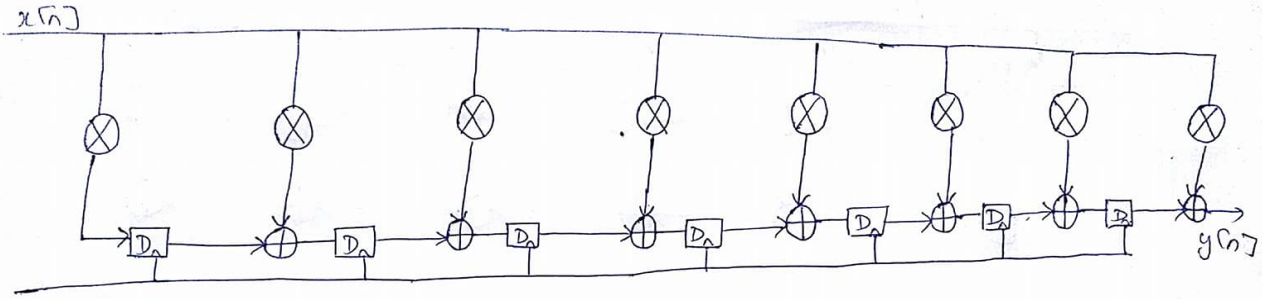
= 56ns

∴ Throughput = $\left(\frac{10^9}{56}\right)$

= 17.8×10^5 instructions / sec

1 c.)

(1)(c.)



clk → of period = 5ns

Area

for each multiplier = 2units

for each adder = 1unit

for each d-flipflop = 1unit

total area = 8 × 2 + 7 × 1 + 7 × 1

= 30units

Latency

for each multiplier = 3ns

for each adder = 2ns

Rest = 1ns.

Since for 1st output only last multiplier and adder are used.

∴ Latency = 3ns + 2ns

= 5ns

Throughput

for each calculation ~~instruction~~

d-flip and multiplier will act in parallel.

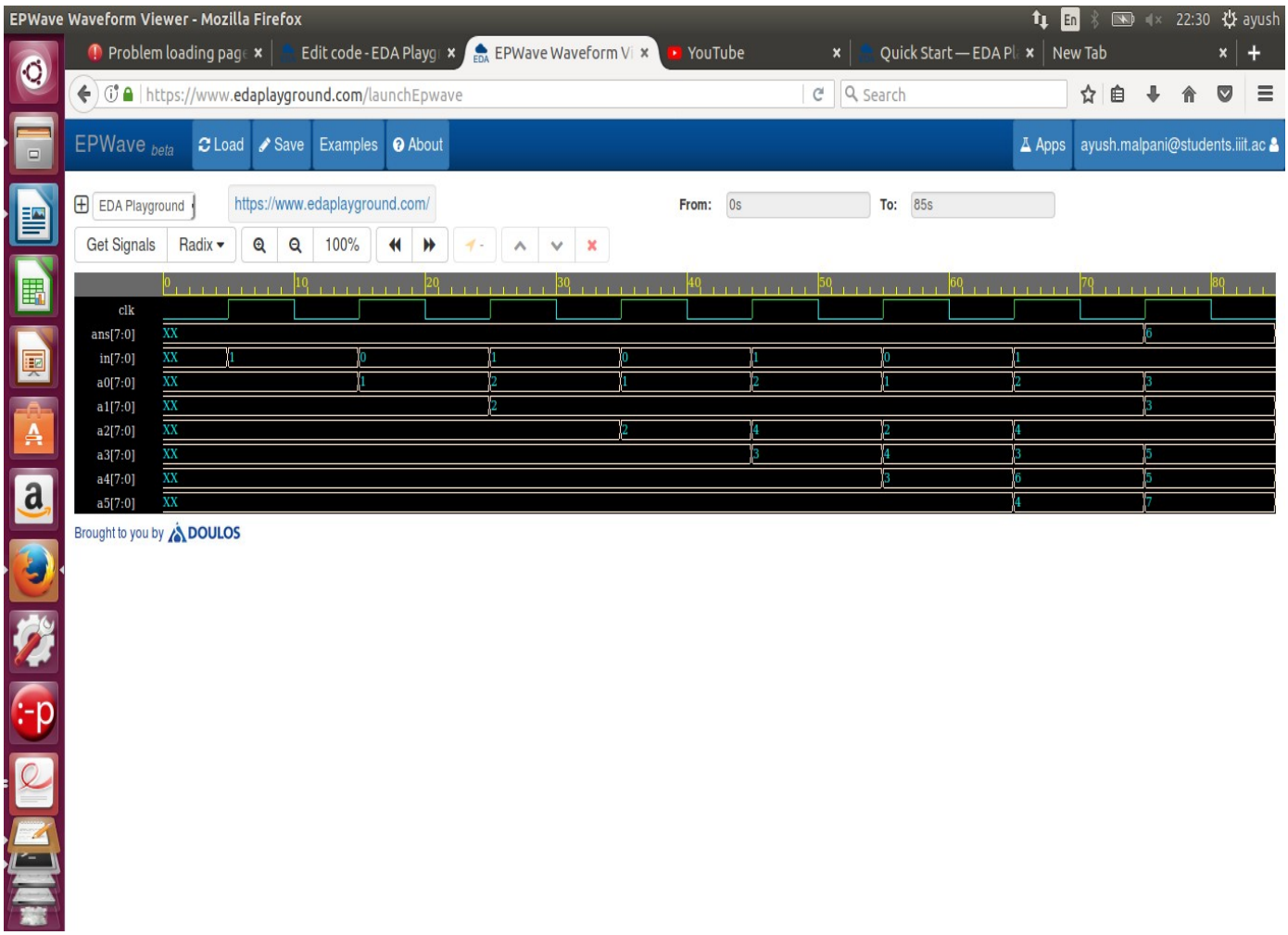
∴ for each calculation ~~instruction~~

one mult. and adder will be used.

∴ 5ns → 1 ~~ins~~ calculation

1s → $\frac{10^9}{5} = (2 \times 10^8)$

Area = 30 units
Latency = 5ns
Throughput = 2 * 10^8 instructions/second

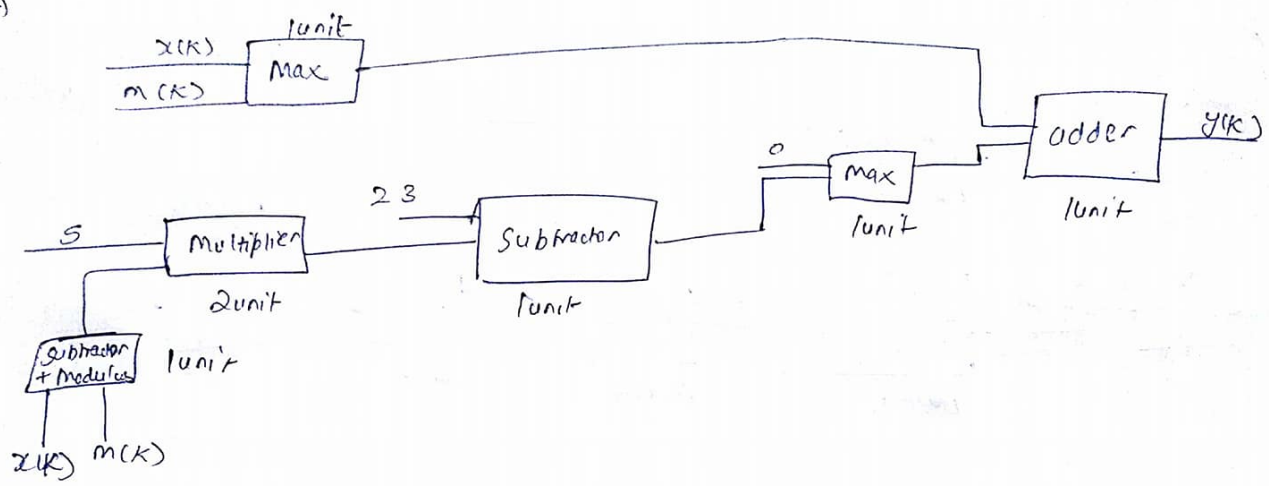


1 d.)

	Isomorphic architecture	With one multiplier adder	Pipelined
Area	30 units	7 units	30 units
Latency	17 ns	56 ns	5 ns
Throughput	55 * 10^6 instructions/sec	178 * 10^5 instructions/sec	2 * 10 ^8 instruction/sec

2 a.)

(2.) (a.)



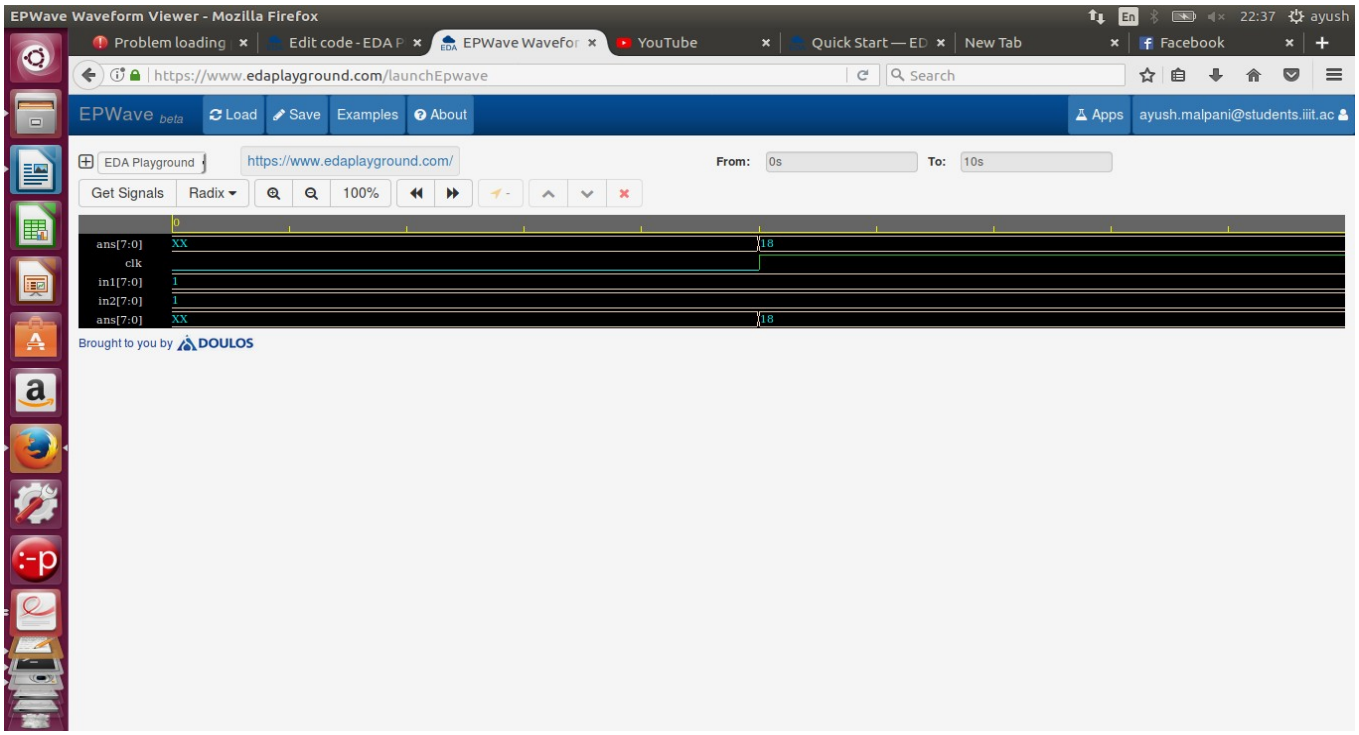
Total area = 1 + 2 + 1 + 1 + 1 + 1 = 7units.

~~Total~~ latency = 2ns (Subtraction + modulus). + 3ns (Multiplier) + 2ns (Subtraction) + 2ns (Max) + 2ns (adder)

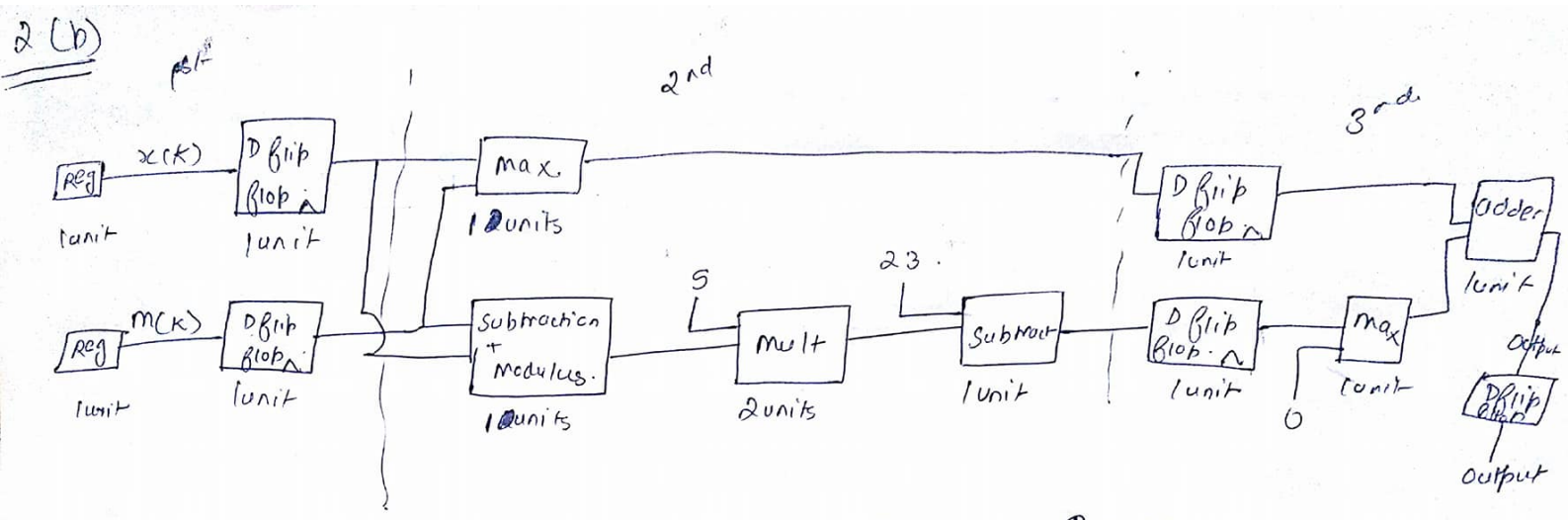
= 11ns.

Throughput = $\frac{1}{11\text{ns}}$ = 11ns.

$1s = \left(\frac{10^9}{11}\right) = 9 \times 10^7$ instructions / second.



2 (b)



Area.

for each mult = 2units.

for each adder = 1unit

Rest components = 1units.

= 11units.

Latency

20 clock cycles

= 7 units x 2

= 14 ns

7 units clock cycle because critical path is of 7 units in 2nd stage.

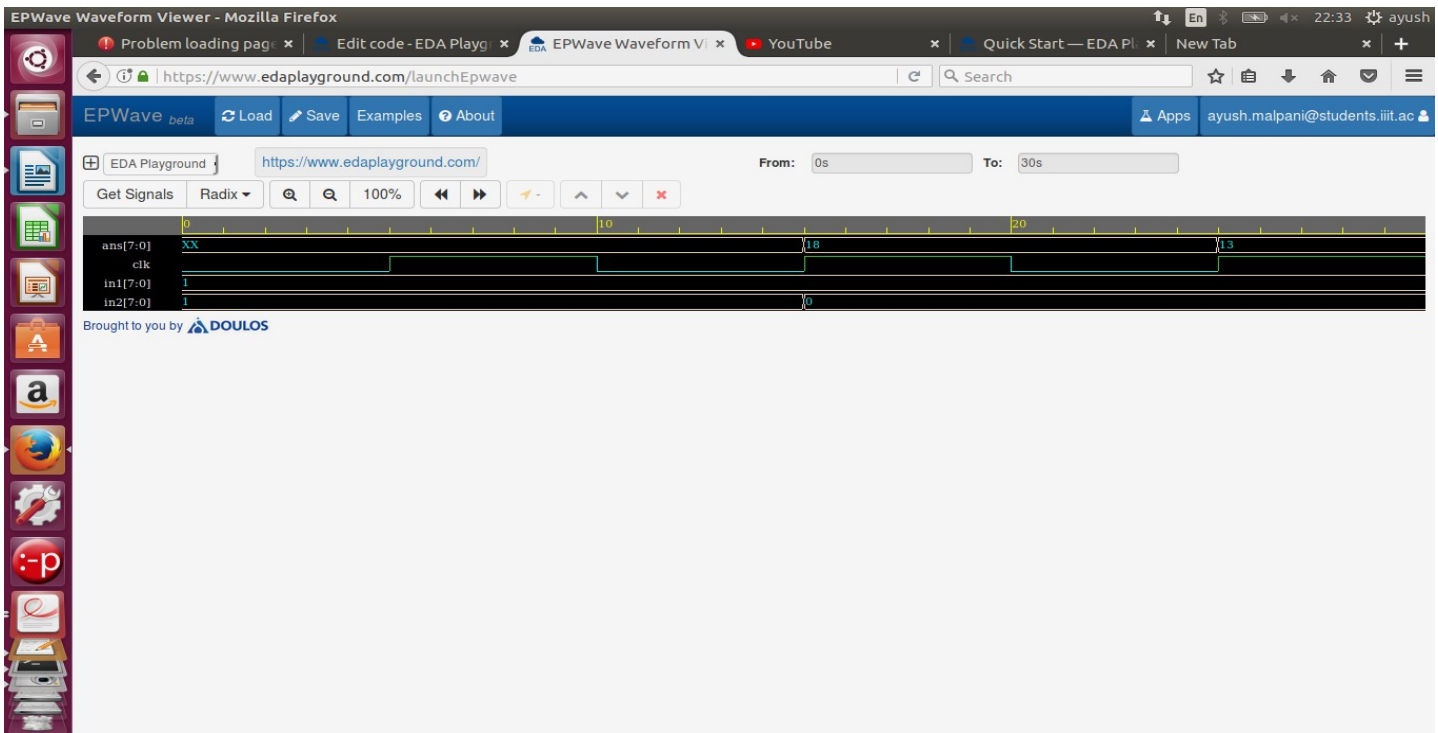
Throughput

Assuming unit to be

1 instruction = 7 ns.

∴ total throughput

$= \frac{10^9}{7} = \frac{14}{7} \times 10^7$ instructions/s



2c.)

	Pipelined	Without Pipelined
Area	11 units	7units
Latency	14 ns	11 ns
Throughput	14 * 10^7 instructions/second	9 * 10^7 instructions/second

2d.) The area in the first case i.e the non-pipelined case comes out to be 7 units while in the pipelined case it comes out to be 11 units. Therfore the non-pipeline architecture is more efficient in terms of area as compared to the pipeline architecture.

There is an increase in the throughput in the pipelined case.