Programmable, Finite Impulse Response (FIR) Filter

Engineered by Bruno E. Gracia Villalobos

EE 4513

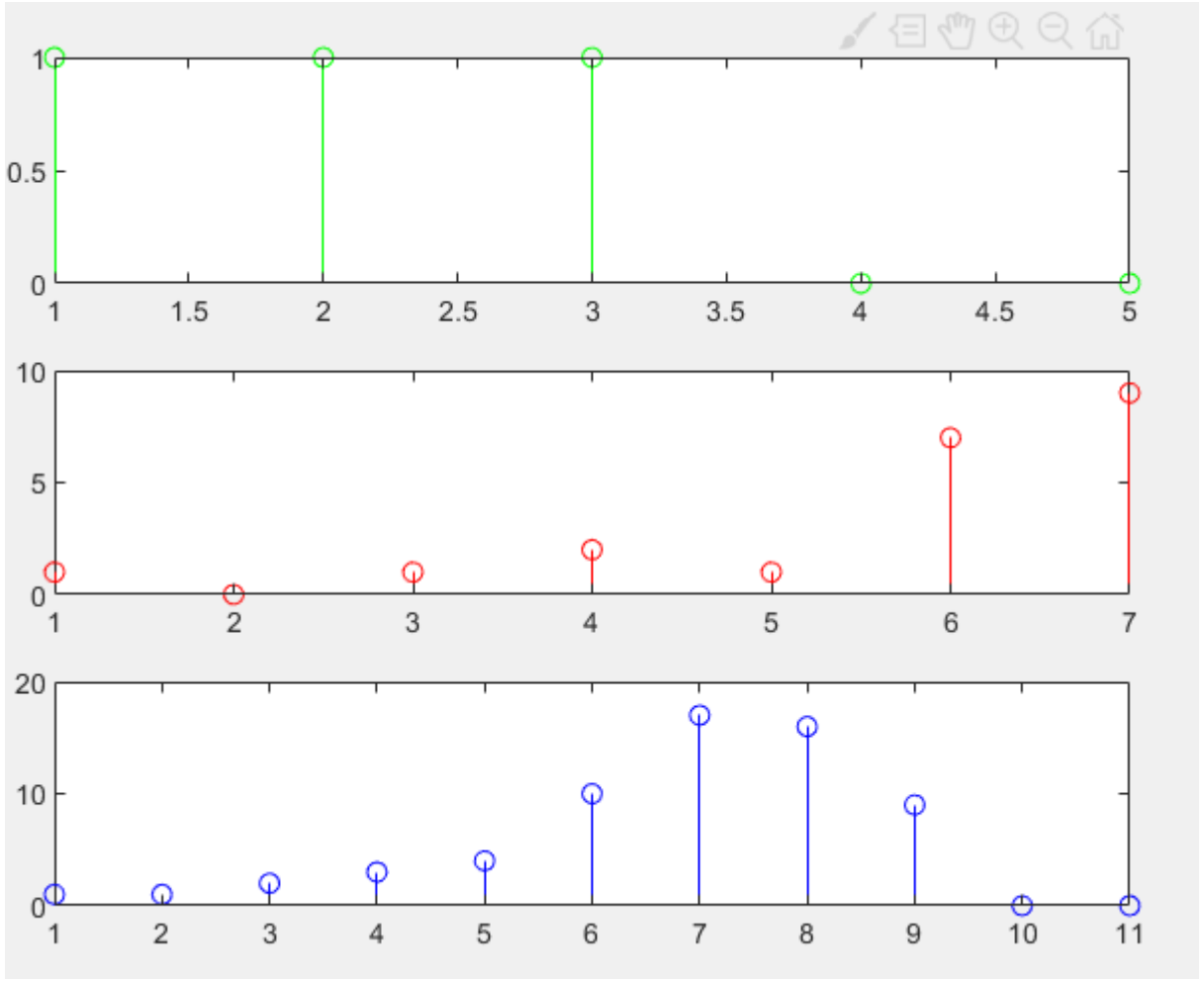
December 2019

WHAT IS A FILTER?

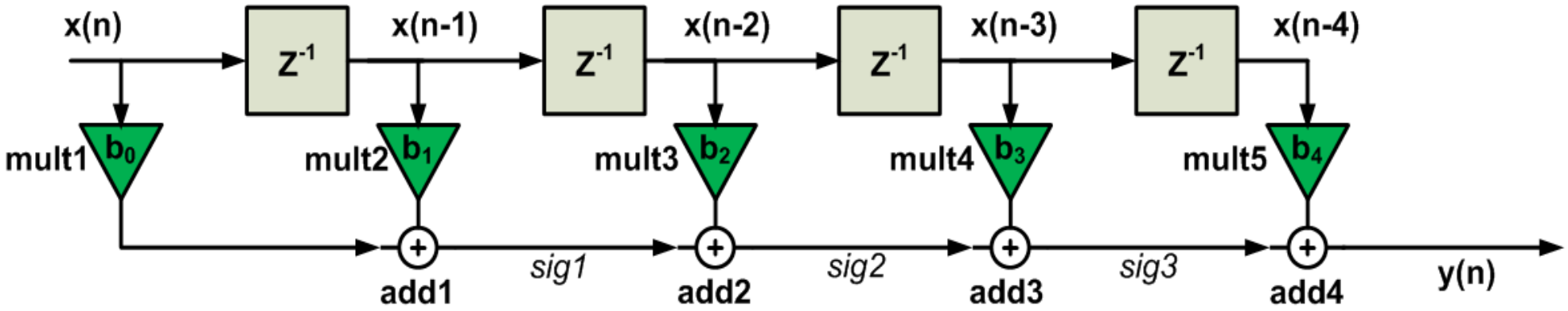
Filter

Signal

Output



DIRECT FORM I – 4th ORDER FIR FILTER

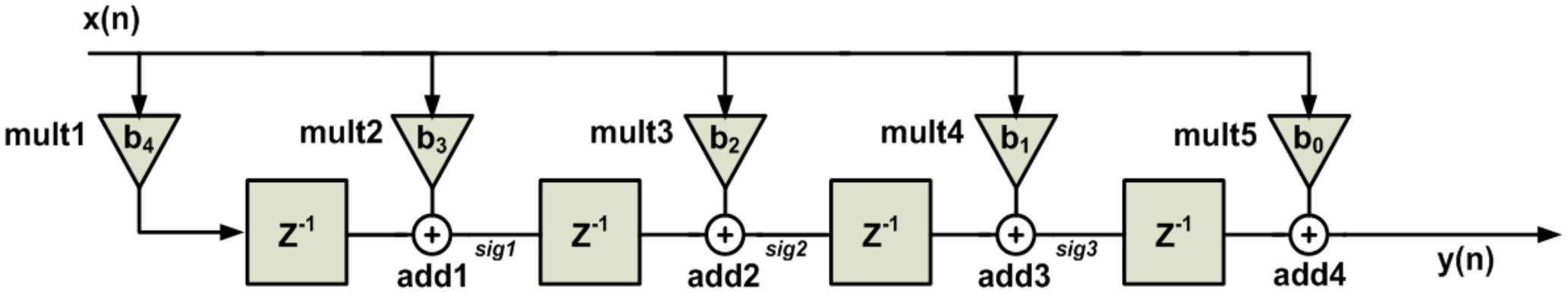


$$y[n] = \sum_{k=0}^{M} h[k]x[n-k]$$

$$y[n] = b4 * x[n-4] + b3 * x[n-3] + b2 * x[n-2] + b1 * x[n-1] + b0 * x[n]$$

$After\ four\ clock\ cycles, y[n]$ sees $\quad T = Tmult + 4Tadd$

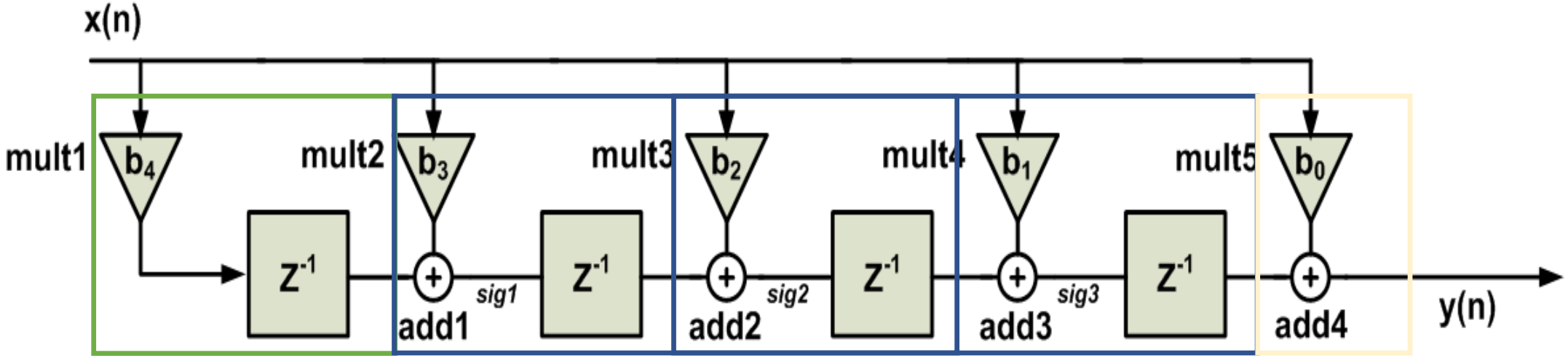
TRANSPOSED, DIRECT FORM I – 4th ORDER



Why?
Increase sampling rate… output y[n] "sees" $T = Tmult + Tadd$

How?
Reverse all branches without changing functionality.
Switch x[n] & y[n] placement.

PROGRAMMABLE….?

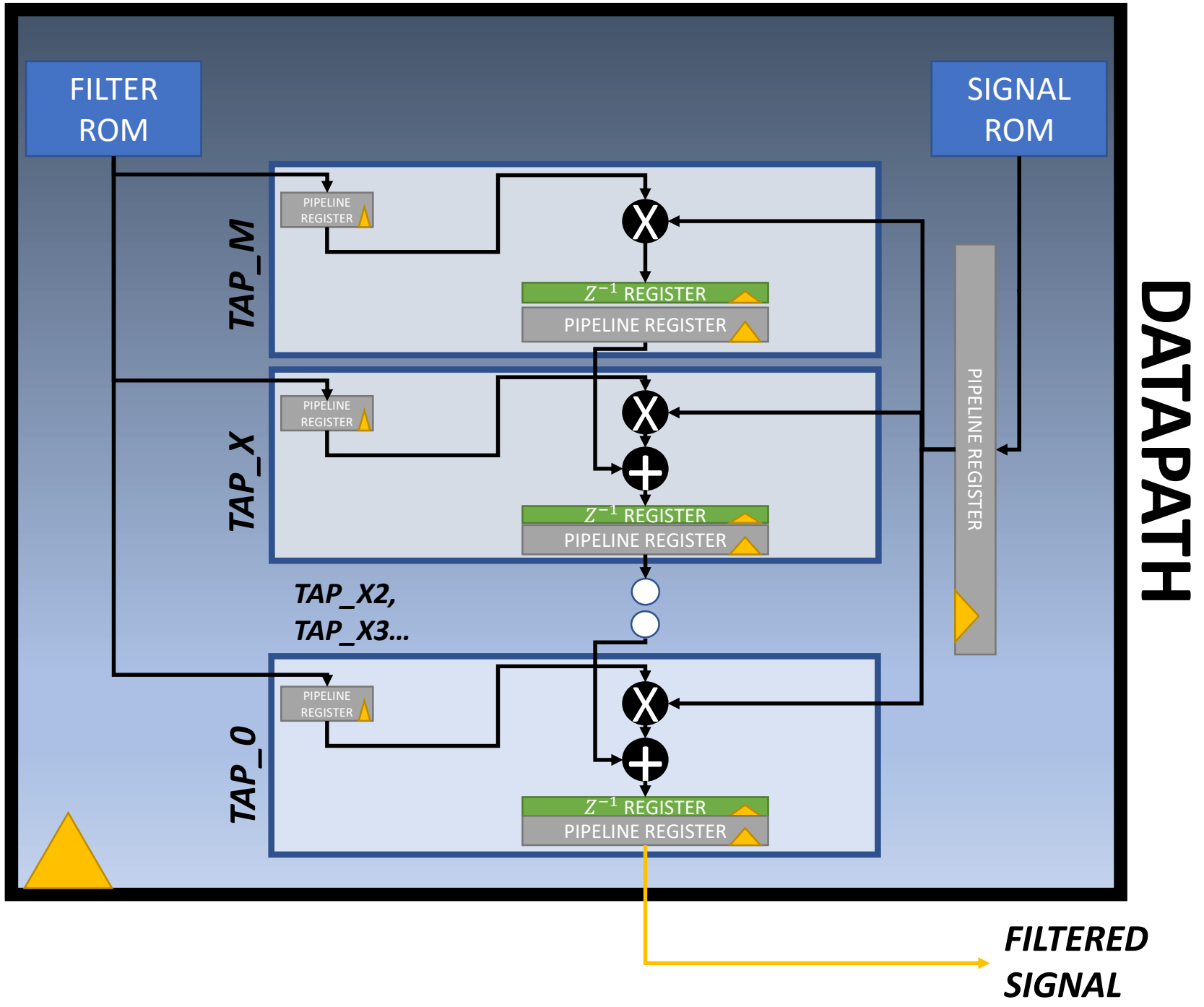


GREEN =
- 1 Multiplier
- 1 Register

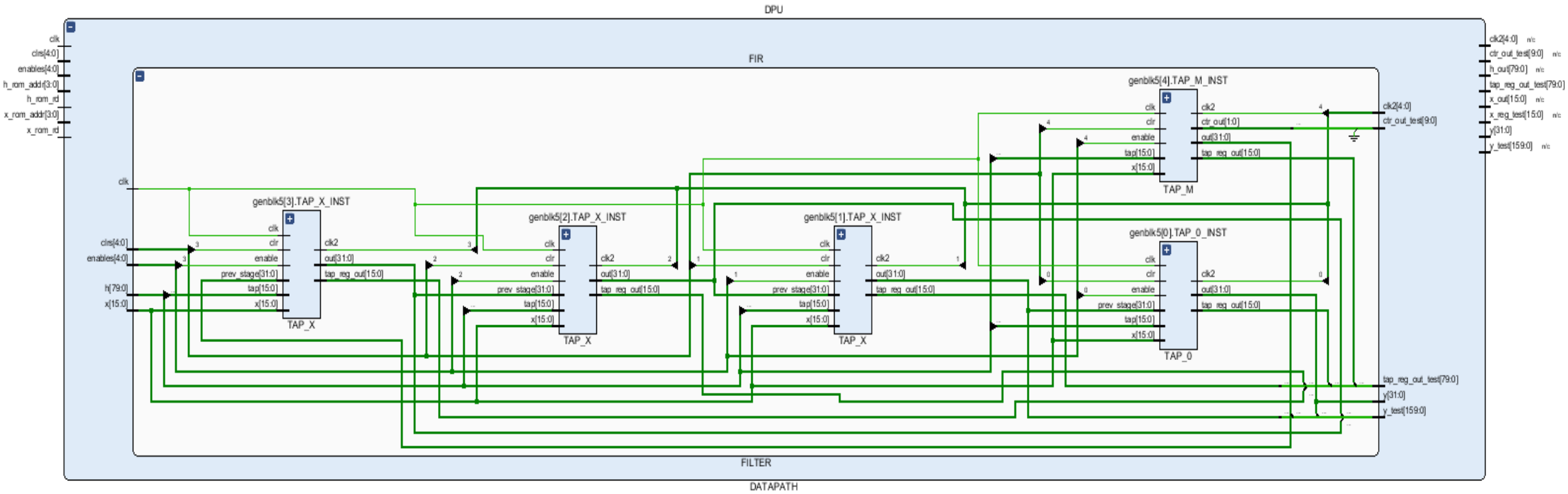
BLUE =
- 1 Multiplier
- 1 Adder
- 1 Register

YELLOW =
- 1 Multiplier
- 1 Adder

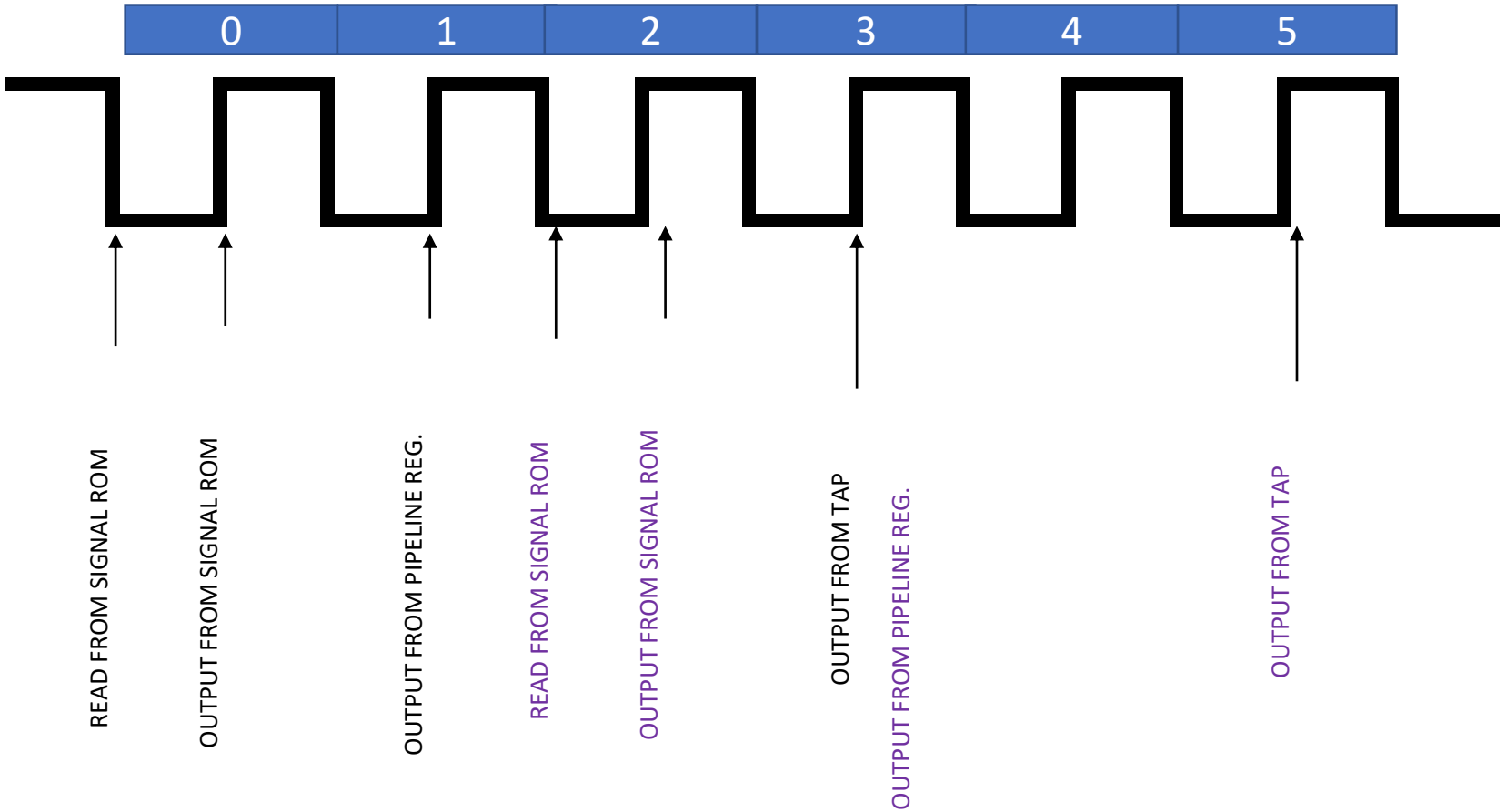
green = 1
yellow = 1
blue = M-1

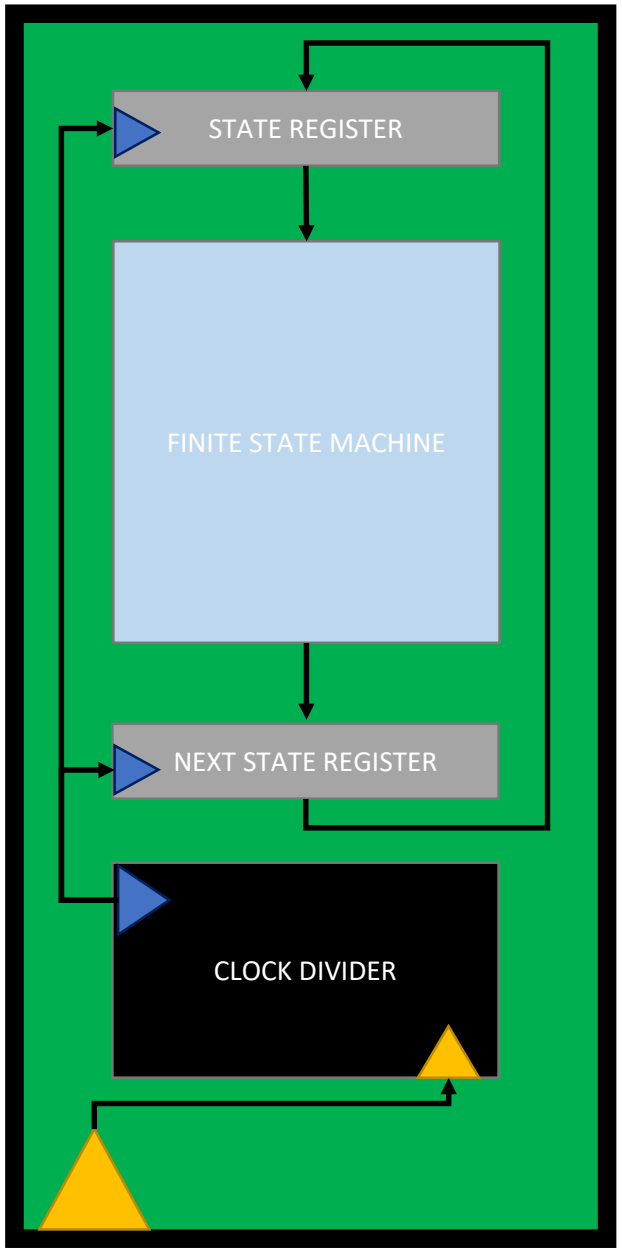

FILTER ROM
SIGNAL ROM
TAP_M
TAP_X
TAP_X2, TAP_X3...
TAP_0
PIPELINE REGISTER
Z^{-1} REGISTER
PIPELINE REGISTER
PIPELINE REGISTER
PIPELINE REGISTER
Z^{-1} REGISTER
PIPELINE REGISTER
PIPELINE REGISTER
PIPELINE REGISTER
Z^{-1} REGISTER
PIPELINE REGISTER
PIPELINE REGISTER
DATAPATH
FILTERED SIGNAL

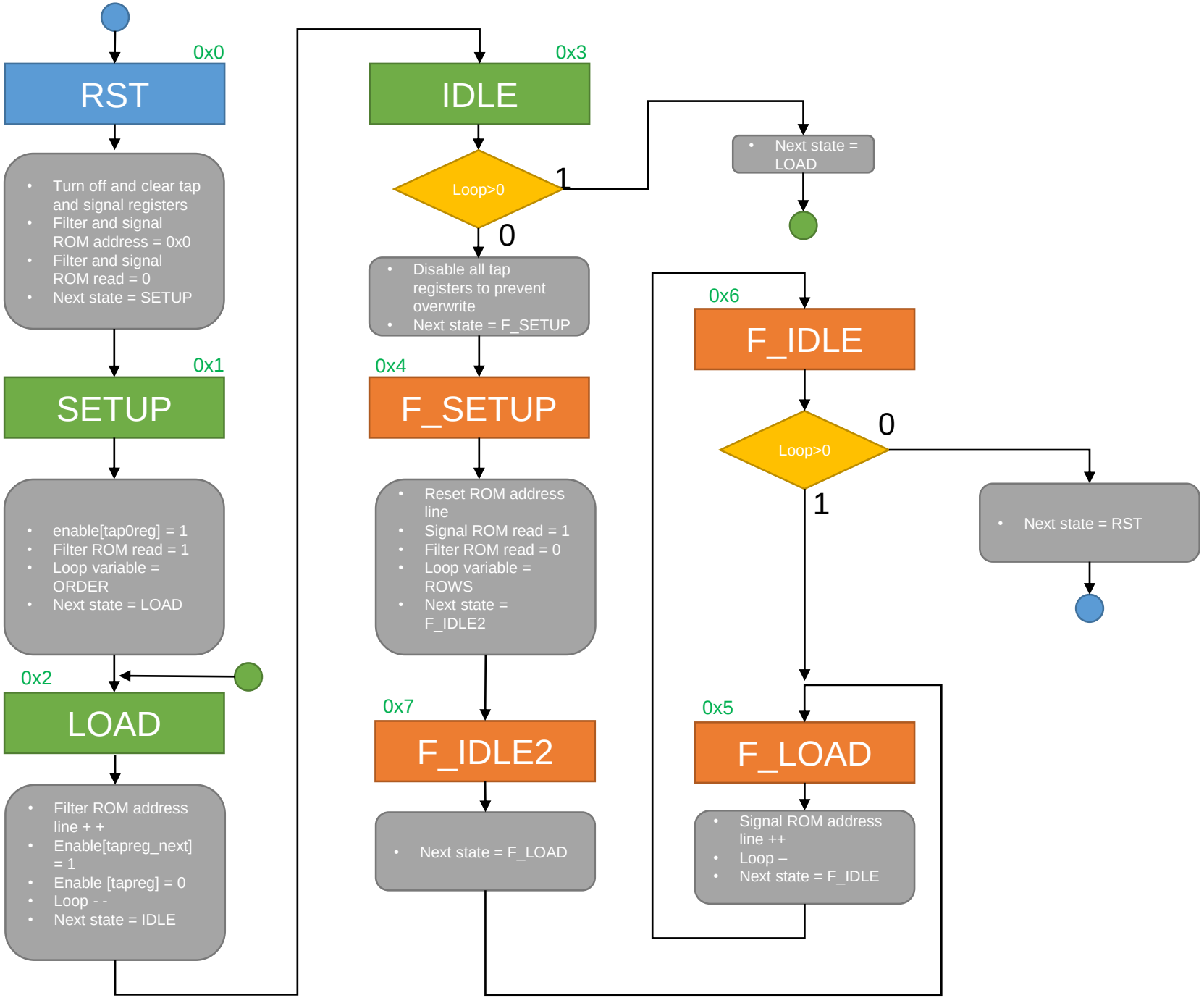
FILTER IN HDL. WHAT IS THE ORDER?

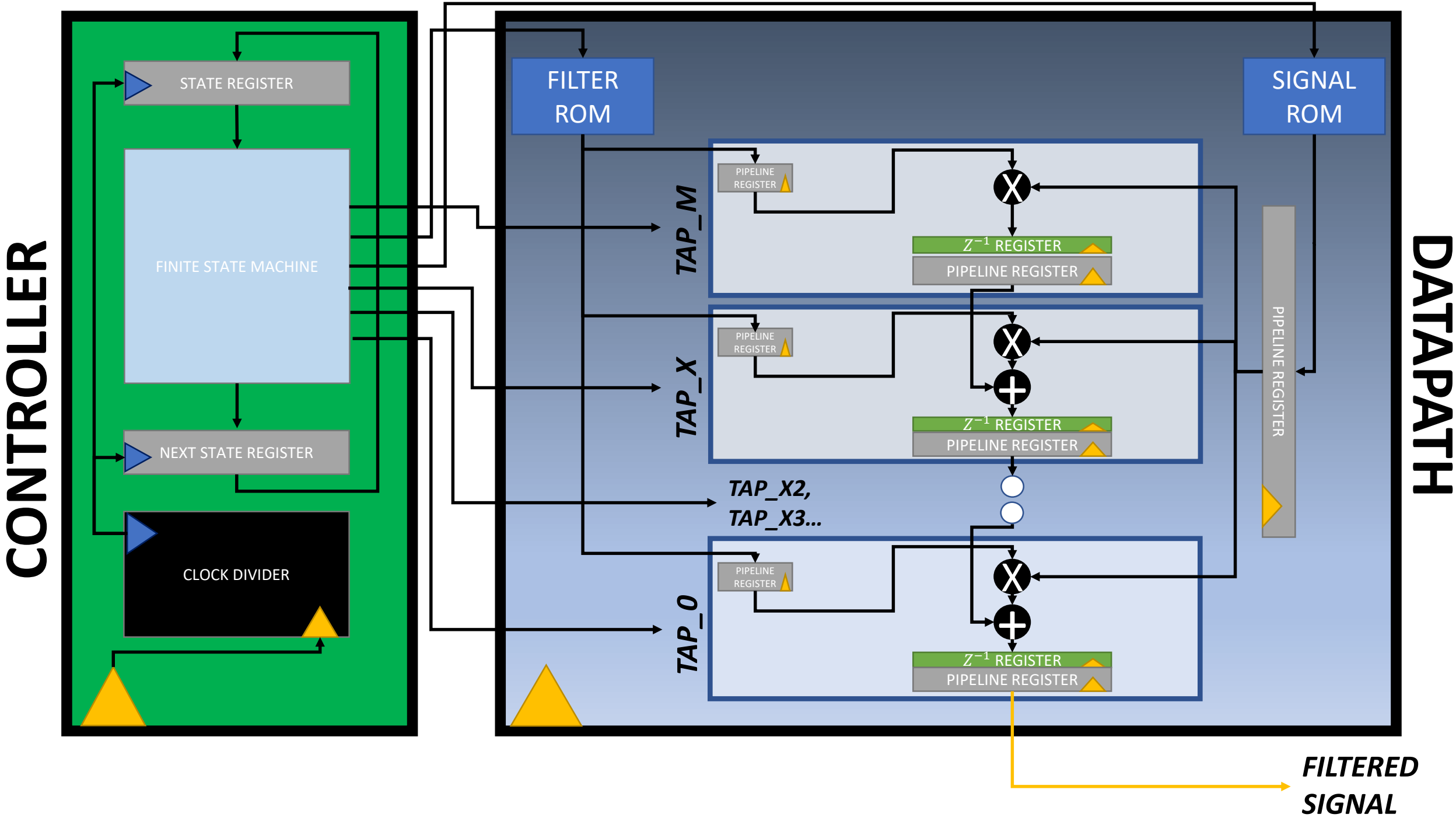


PIPELINE


0
1
2
3
4
5
READ FROM SIGNAL ROM
OUTPUT FROM SIGNAL ROM
OUTPUT FROM PIPELINE REG.
READ FROM SIGNAL ROM
OUTPUT FROM SIGNAL ROM
OUTPUT FROM TAP
OUTPUT FROM PIPELINE REG.
OUTPUT FROM TAP


CONTROLLER
STATE REGISTER
FINITE STATE MACHINE
NEXT STATE REGISTER
CLOCK DIVIDER


0x0
RST
Turn off and clear tap and signal registers
Filter and signal ROM address = 0x0
Filter and signal ROM read = 0
Next state = SETUP
0x1
SETUP
enable[tap0reg] = 1
Filter ROM read = 1
Loop variable = ORDER
Next state = LOAD
0x2
LOAD
Filter ROM address line + +
Enable[tapreg_next] = 1
Enable [tapreg] = 0
Loop - -
Next state = IDLE
0x3
IDLE
Loop>0
1
0
Next state = LOAD
Disable all tap registers to prevent overwrite
Next state = F_SETUP
0x4
F_SETUP
Reset ROM address line
Signal ROM read = 1
Filter ROM read = 0
Loop variable = ROWS
Next state = F_IDLE2
0x7
F_IDLE2
Next state = F_LOAD
0x6
F_IDLE
Loop>0
0
1
Next state = RST
0x5
F_LOAD
Signal ROM address line ++
Loop –
Next state = F_IDLE

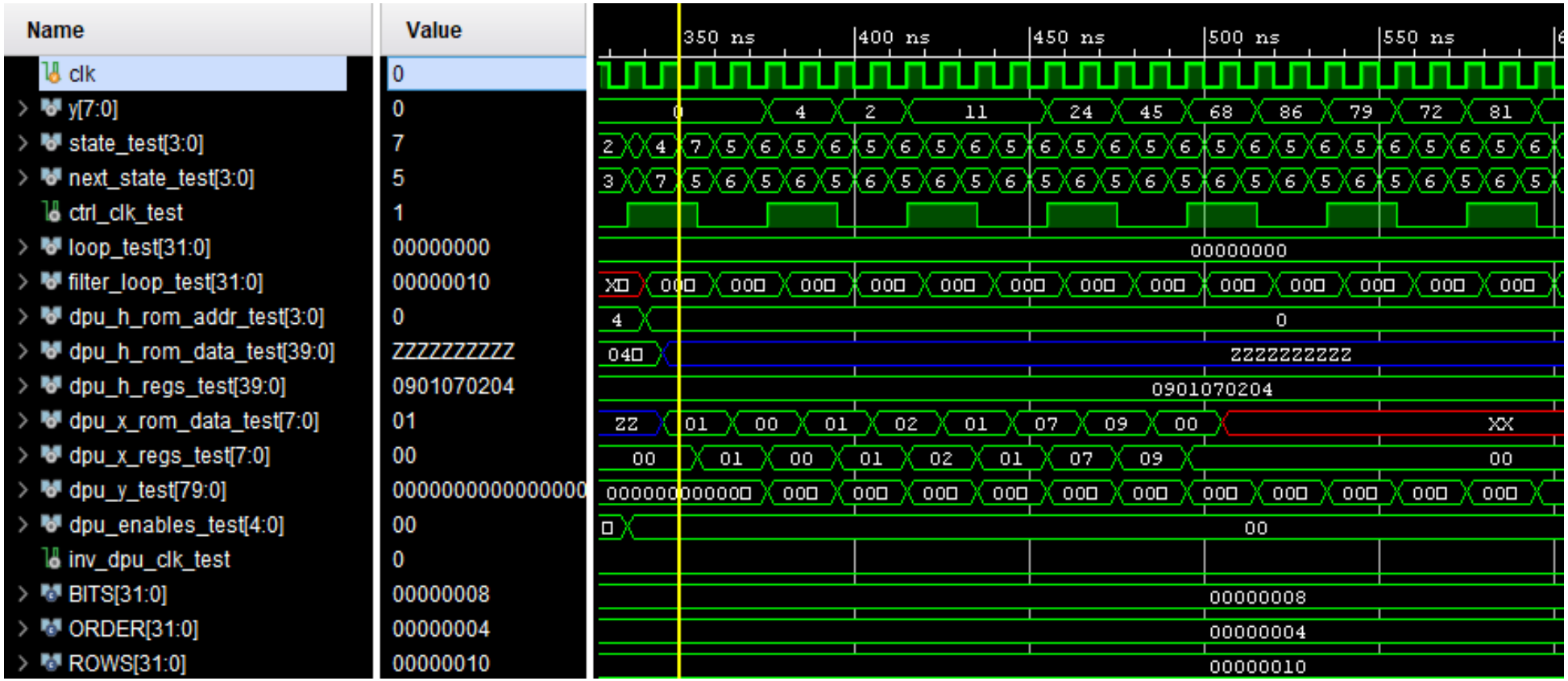

CONTROLLER
STATE REGISTER
FINITE STATE MACHINE
NEXT STATE REGISTER
CLOCK DIVIDER
DATAPATH
FILTER ROM
SIGNAL ROM
PIPELINE REGISTER
TAP_M
TAP_X
TAP_X2,
TAP_X3...
TAP_0
Z^{-1} REGISTER
PIPELINE REGISTER
FILTERED SIGNAL

COOL… DOES IT WORK?

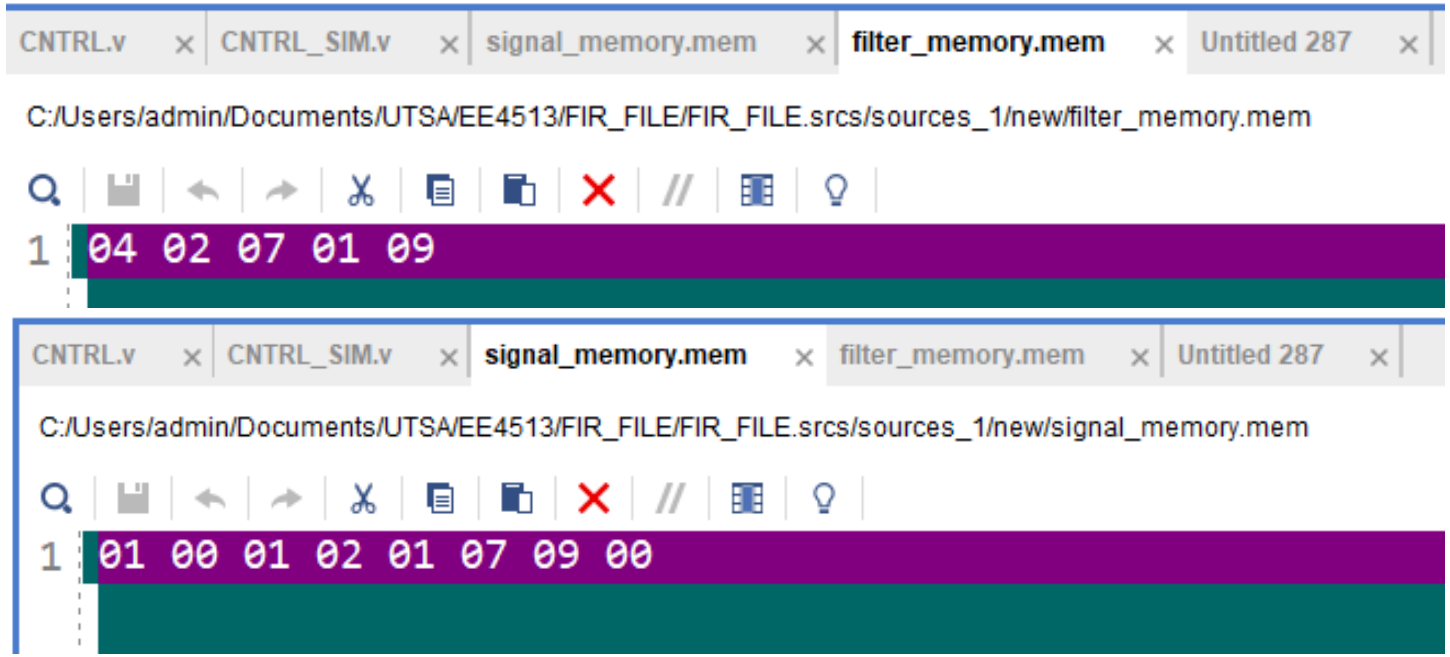
h = [4 2 7 1 9]

x = [1 0 1 2 1 7 9]

y = [4 2 11 11 24 45 68 86 79 72 81]

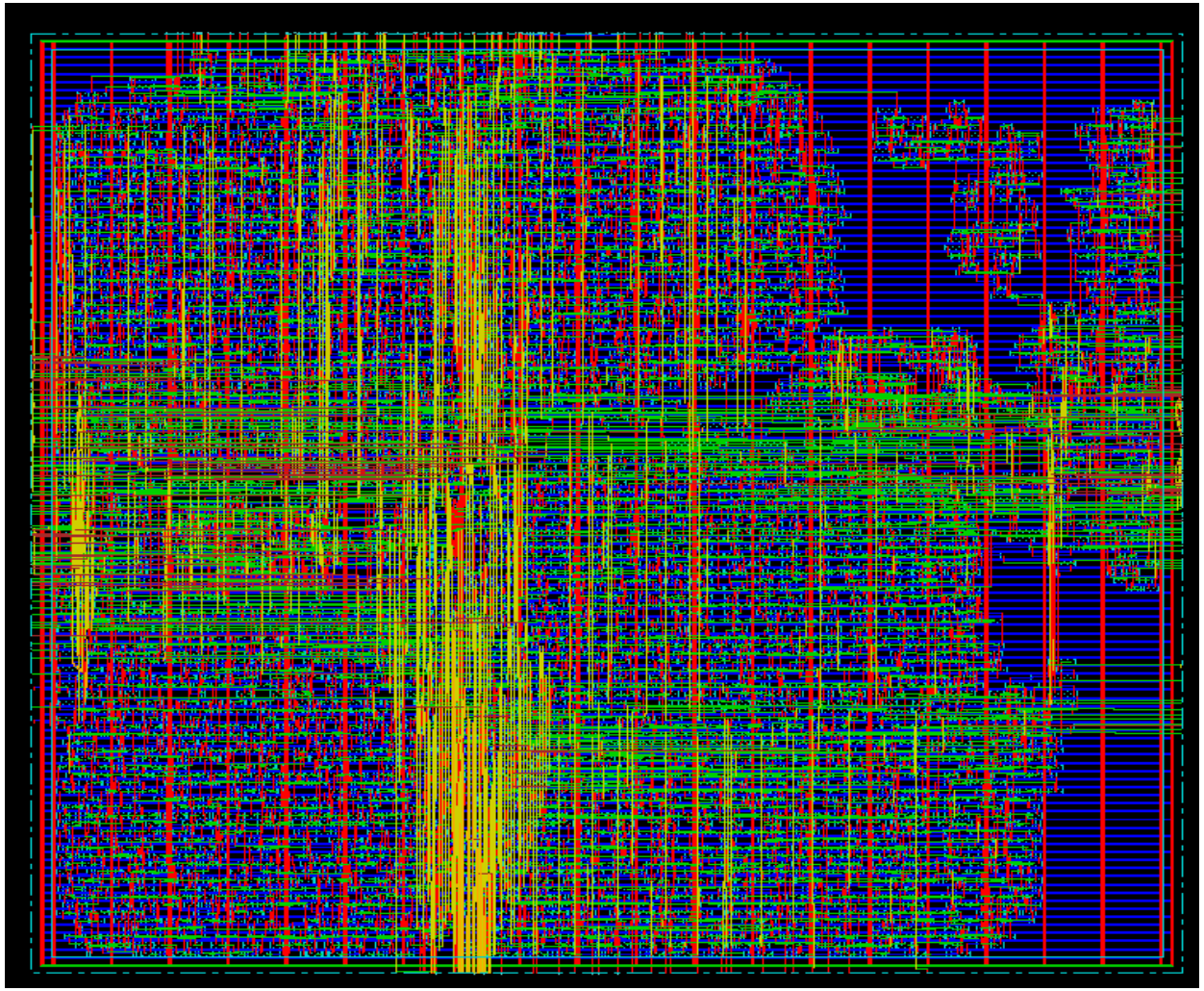


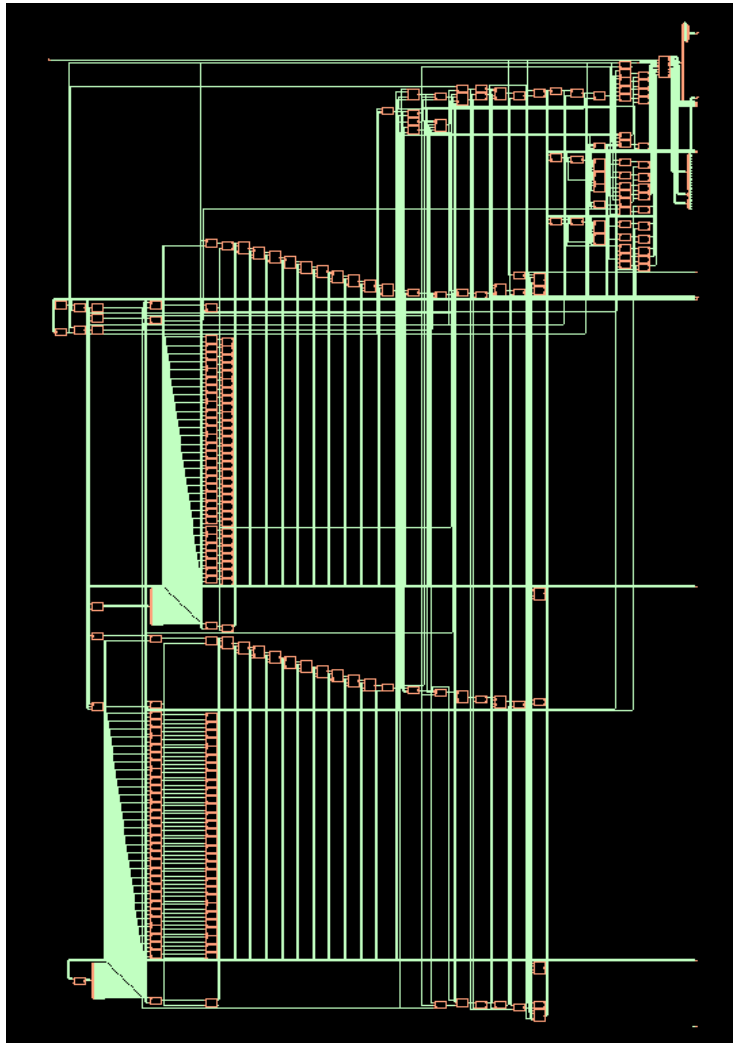
WHERE IS THE MEMORY COMING FROM?



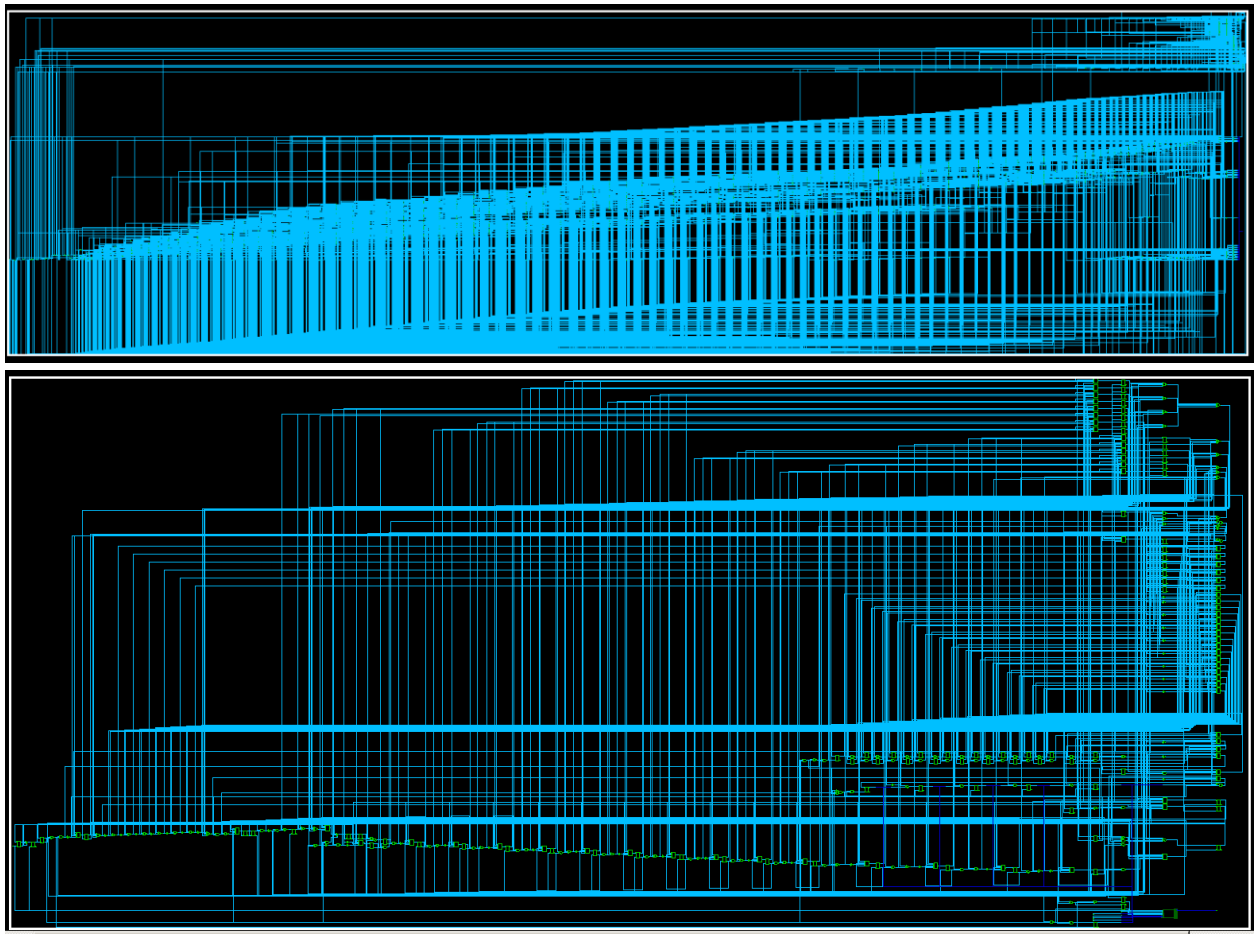

```
reg [BITS-1:0] rom [0:ROWS-1];

initial begin
    $readmemh(FILENAME, rom);
end
```

180NM





45 VS 90 NM



	45NM	90NM
TIMING	0.46f	0.61 f
TOTAL AREA	10948.894428	64264.217708
TOTAL POWER	3.8015e-02 mW	181.5616 uW

THANK YOU ALL