10/17/16 Block Diagrams For FIR filters

Objective: Towards realization of FIR filters in digital hardware.

Recall: FIR filter compute

$$y[n] = \sum_{k=0}^{M-1} b_k \cdot x[n-k]$$

Number of operations:

per sample: M multiplication
M additions

E.g.: $M = 250$

$f_s = 80$ MSamples/s

$\Rightarrow 0.25 \cdot 10^3 \cdot 80 \cdot 10^6 =$

$20 \cdot 10^9 \frac{\text{operations}}{\text{sec}}$

This kind of computational rate can only be realized in HW.

10/17/16 # Block Diagrams For FIR filters

Objective: Towards realization of FIR filters in digital hardware.

Recall: FIR filter compute

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

Number of operations:

per sample: M multiplication
M additions

E.g.: $M = 250$

$f_s = 80$ MSamples/s

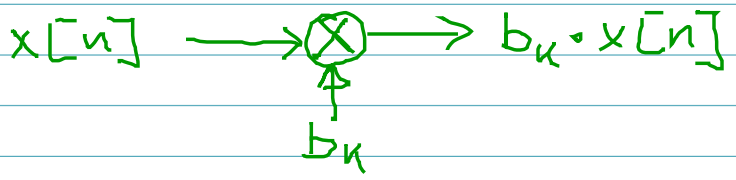
$\Rightarrow 0.25 \cdot 10^3 \cdot 80 \cdot 10^6 =$

$20 \cdot 10^9 \frac{\text{operations}}{\text{sec}}$

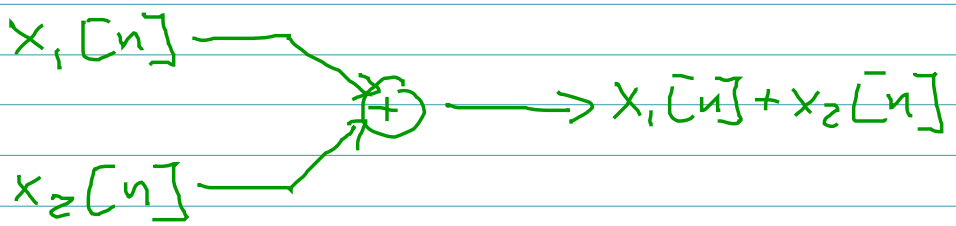
This kind of computational rate can only be realized in HW.

Building blocks:

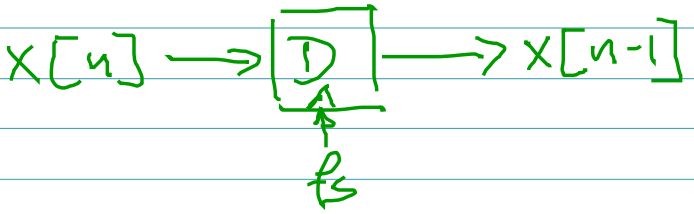
Multipliers:



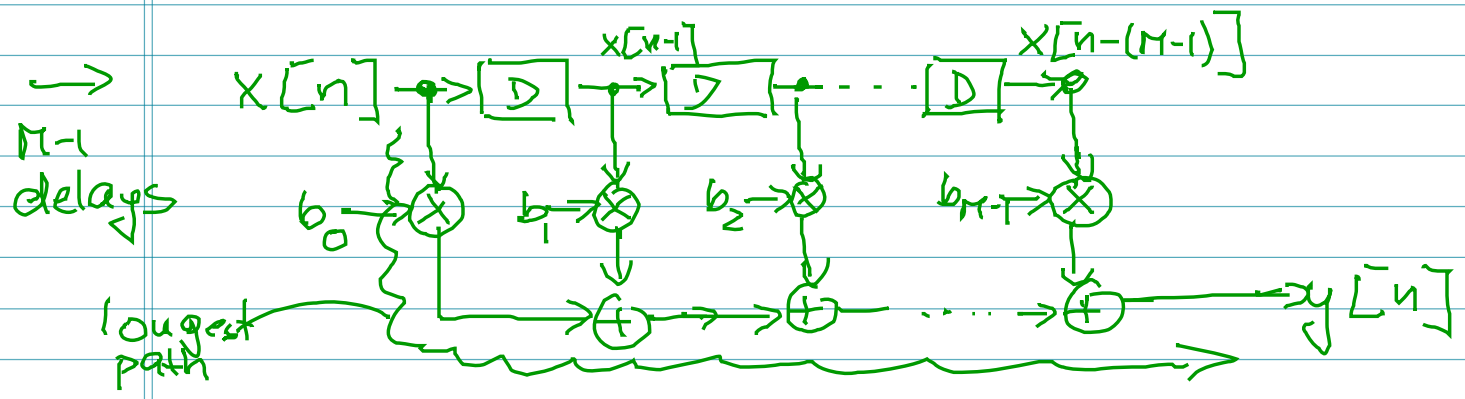
Adders:



Unit Delay:



Want: $y[n] = \sum_{k=0}^{M-1} b_k \, x[n-k]$



"Direct Form" Implementation

How long does it take to compute $y[n]$ if

- a multiplier takes 10 ns
- an adder takes 2 ns

before its output "settles"

Longest path between $x[n]$ and $y[n]$ without intervening delays has

1 multiplier → 10 ns

$(M-1)$ adders → $(M-1) \cdot 2$ ns

E.g.: if $M = 251$ ⇒ total delay

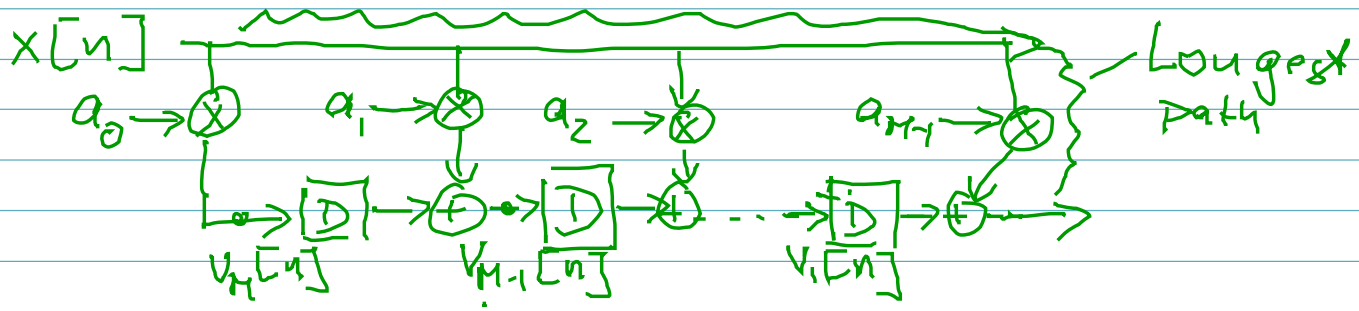
$10 \text{ ns} + 250 \cdot 2 \text{ ns} = 510 \text{ ns}$

$\frac{1}{510 \text{ ns}} \approx \frac{1}{500 \text{ ns}} = \frac{1}{0.5 \cdot 10^{-6}} = 2 \cdot 10^6$

$= 2 \text{ M samples/sec}$

max. rate

This is a much better circuit



Note: Longest path between input and output without intervening delays.

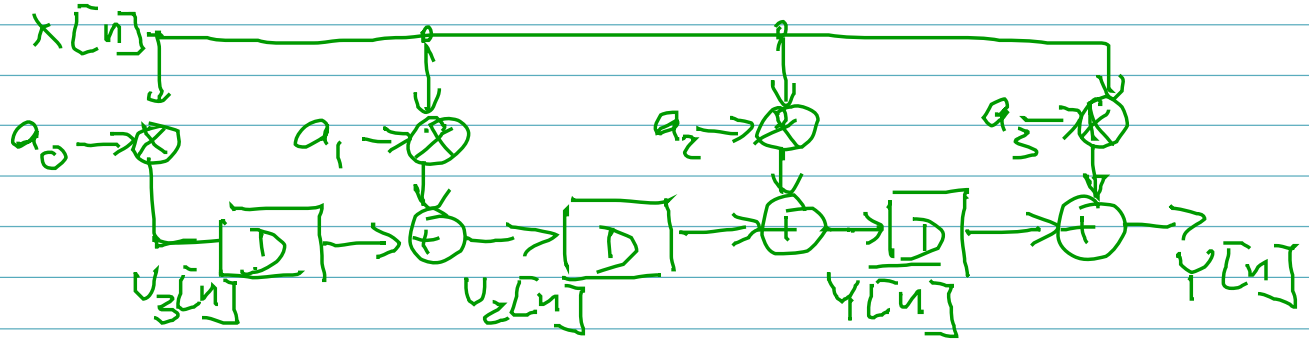
| multiplier → 10ns
| adder → 2ns
12ns

$\frac{1}{12ns} \simeq \frac{80M\text{samples}}{sec}$ Much higher rate than direct form

Q: What does this system do?
→ Find difference equation for this system.

Method:
1) Name intermediate signal
- each signal at input of delay gets a name
2) Write a signal for each named signal and output signal.
e.g. $v_{M-1}[n] = v_M[n-1] + a_1 x[n]$
3) Solve system of equations for $y[n]$
- Eliminate all intermediate signals

Example:



2. Four signal equations:

1) $y[n] = a_3 \cdot x[n] + v_1[n-1]$

2) $v_1[n] = a_2 \cdot x[n] + v_2[n-1]$.

3) $v_2[n] = a_1 \cdot x[n] + v_3[n-1]$

4.) $v_3[n] = a_0 \cdot x[n]$

3. Solve for $y[n]$

4.) & 3): 5) $v_2[n] = a_1 x[n] + \underbrace{a_0 x[n-1]}_{= v_3[n-1]}$

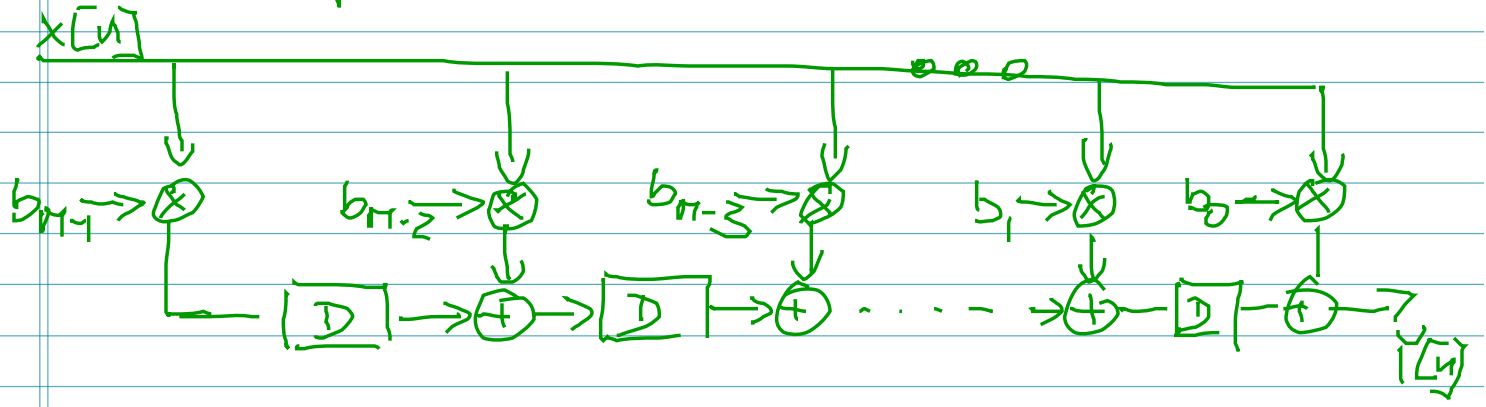
5.) & 2): 6) $v_1[n] = a_2 x[n] + \underbrace{a_1 x[n-1] + a_0 x[n-2]}_{= v_2[n-1]}$

6.) & 1): $y[n] = a_3 x[n] + \underbrace{a_2 x[n-1] + a_1 x[n-2] + a_0 x[n-3]}_{= v_1[n-1]}$

$y[n] = a_3 x[n] + a_2 x[n-1] + a_1 x[n-2] + a_0 x[n$

Want: $y[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] + b_3 x[n-3]$

"Transposed form" FIR filter



This structure realizes:

$$y[n] = \sum_{k=0}^{M-1} b_k \, x[n-k]$$

and avoids long chains of compute elements (adders, multipliers)