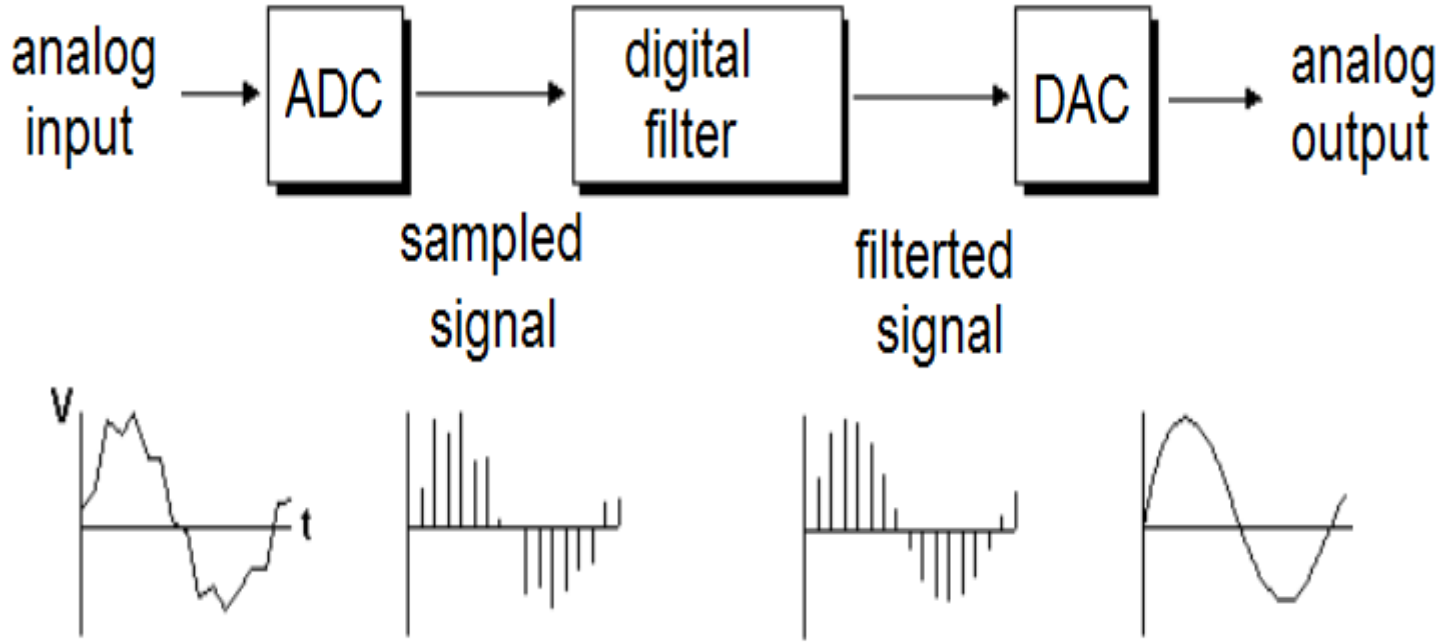
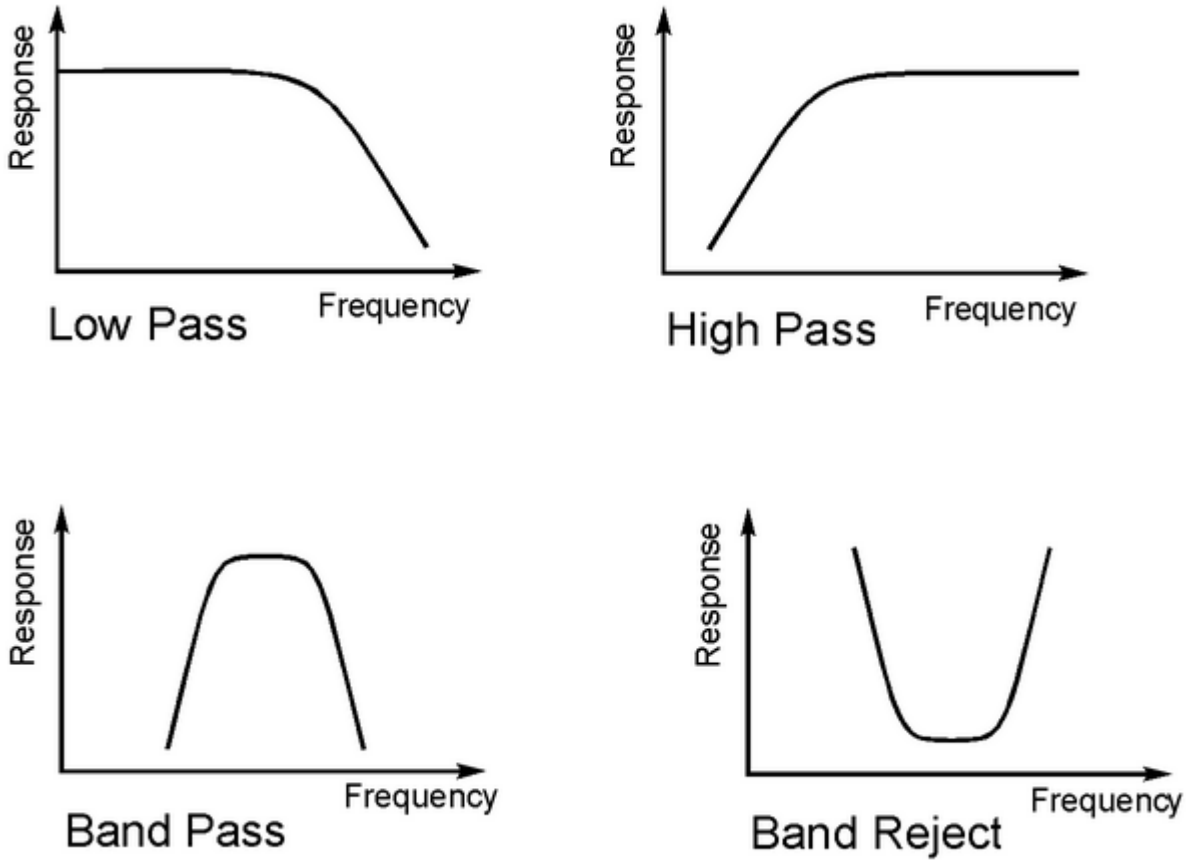
Digital Filters

- FIR & IIR Filters
- Moving Average Filter
- Hardware Structures
- Physical Implementation


analog
input
ADC
digital
filter
DAC
analog
output
sampled
signal
filterted
signal
V
t

Digital filters are used to remove some frequencies from a signal and let other frequencies pass through to the output.



- Recursive filters use past output values y([n – i] to form the current output.

  e.g. $y[n] = 0.25y[n - 1] + 0.25x[n]$

  Infinite Impulse Response (**IIR**)Filter

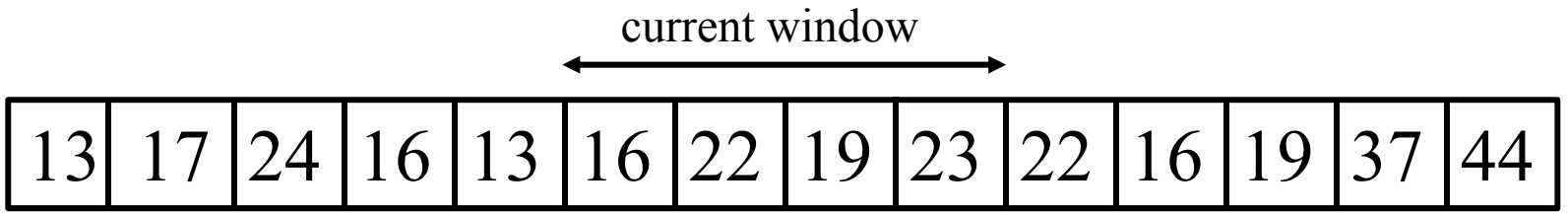
- Non-Recursive filters only use input values.

  e.g. $y[n] = 0.25x[n - 1] + 0.25x[n]$

  Finite Impulse Response (**FIR**) Filter

Moving Average Filter

Consider the case of a 4-point averaging window:



$$y[n] = \tfrac{1}{4}\left(x[n] + x[n-1] + x[n-2] + x[n-3]\right)$$

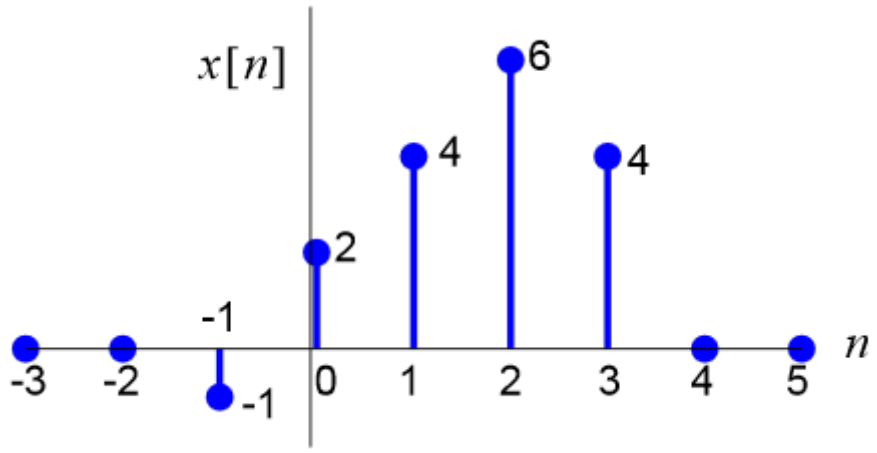
$$\text{current output} = (16 + 22 + 19 + 23) / 4 = 20$$

This is a causal filter. The present output is formed from only past and present signal values.

Consider the following filter:

$$y[n] = \frac{1}{3}\left(x[n] + x[n-1] + x[n-2]\right)$$

With a finite-length input sequence:

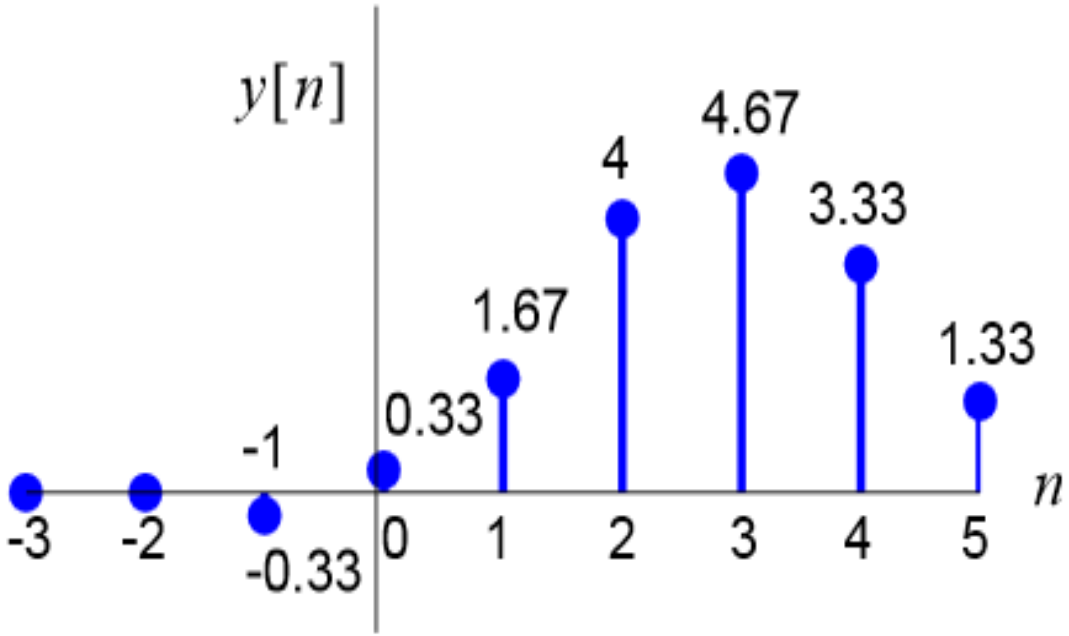


Output values can be calculated as follows:

$$y[0] = \frac{1}{3}(x[0] + x[-1] + x[-2])$$

$$= \frac{1}{3}(2 - 1 + 0) = \frac{1}{3} = 0.333$$

$$y[2] = \frac{1}{3}(6 + 4 + 2) = \frac{12}{3} = 4$$



- The action of the filter has been to ‘smooth out’ the waveform.
- This filter could operate in real time as only past and present values are used.
- The system is ‘causal’.

The General FIR Filter

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

For the 4-point moving average filter:

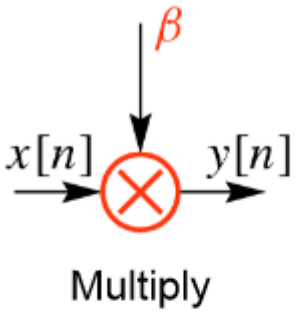
$M = 3$ and $b_0 = b_1 = b_2 = b_3 = 0.25$

For a hardware implementation of an FIR filter we require:

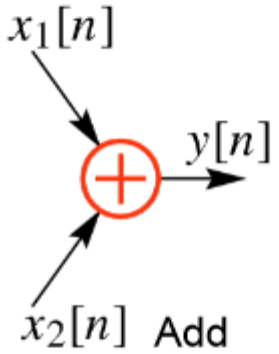
- multiplication
- addition
- signal delay

Building Blocks

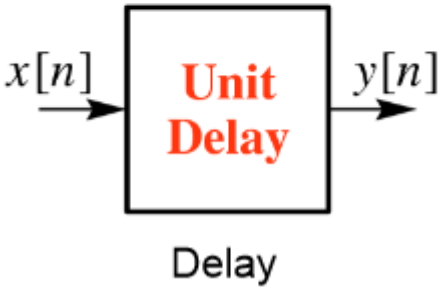
Multiplier: y[n] = βx[n]
Must be fast and have desired precision. There may be many coefficients. (256 is not uncommon)



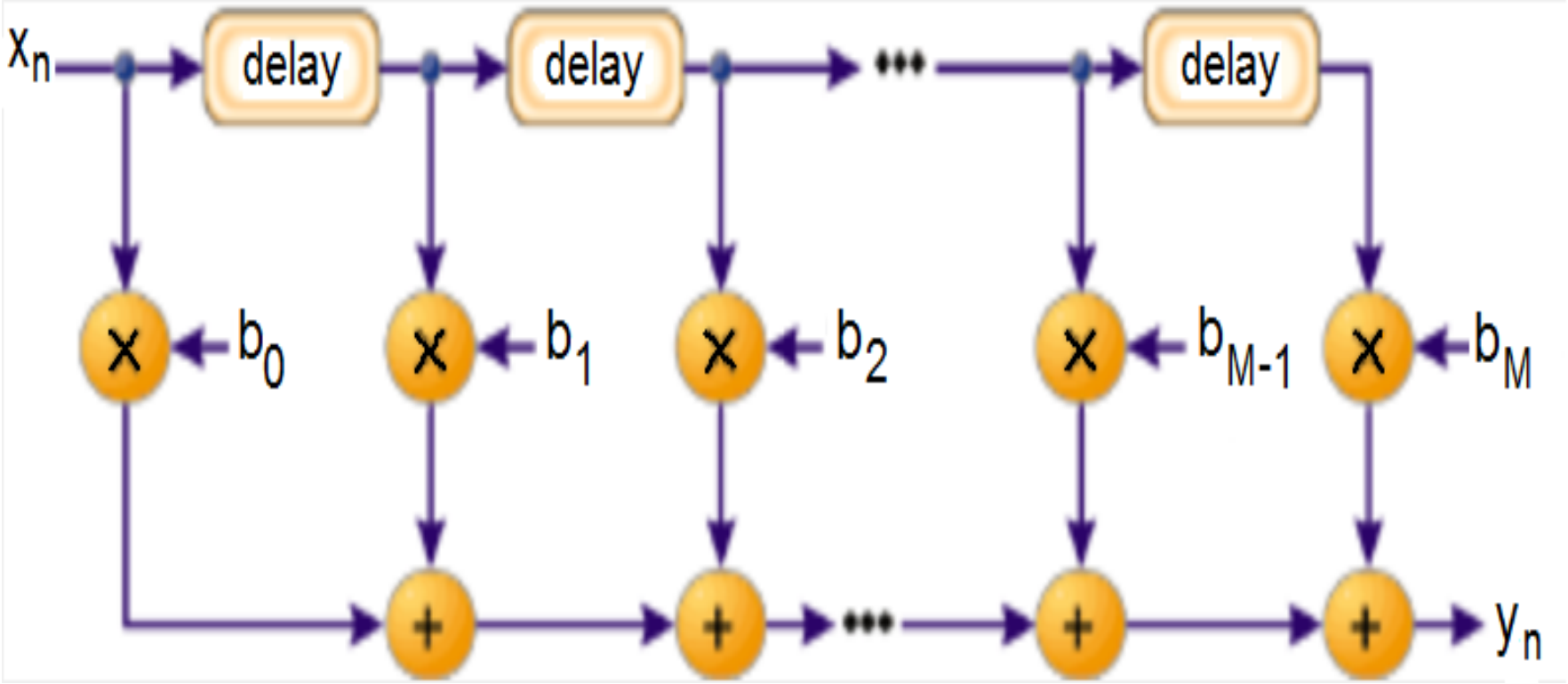
Adder: $y[n] = x_1[n] + x_2[n]$
Additions are required along with multiplications. DSP devices contain dedicated multiply-accumulate units.



Unit Delay: y[n] = x[n - 1]
Provides a one sample delay, The sample is stored in memory. (register)



Block Diagram



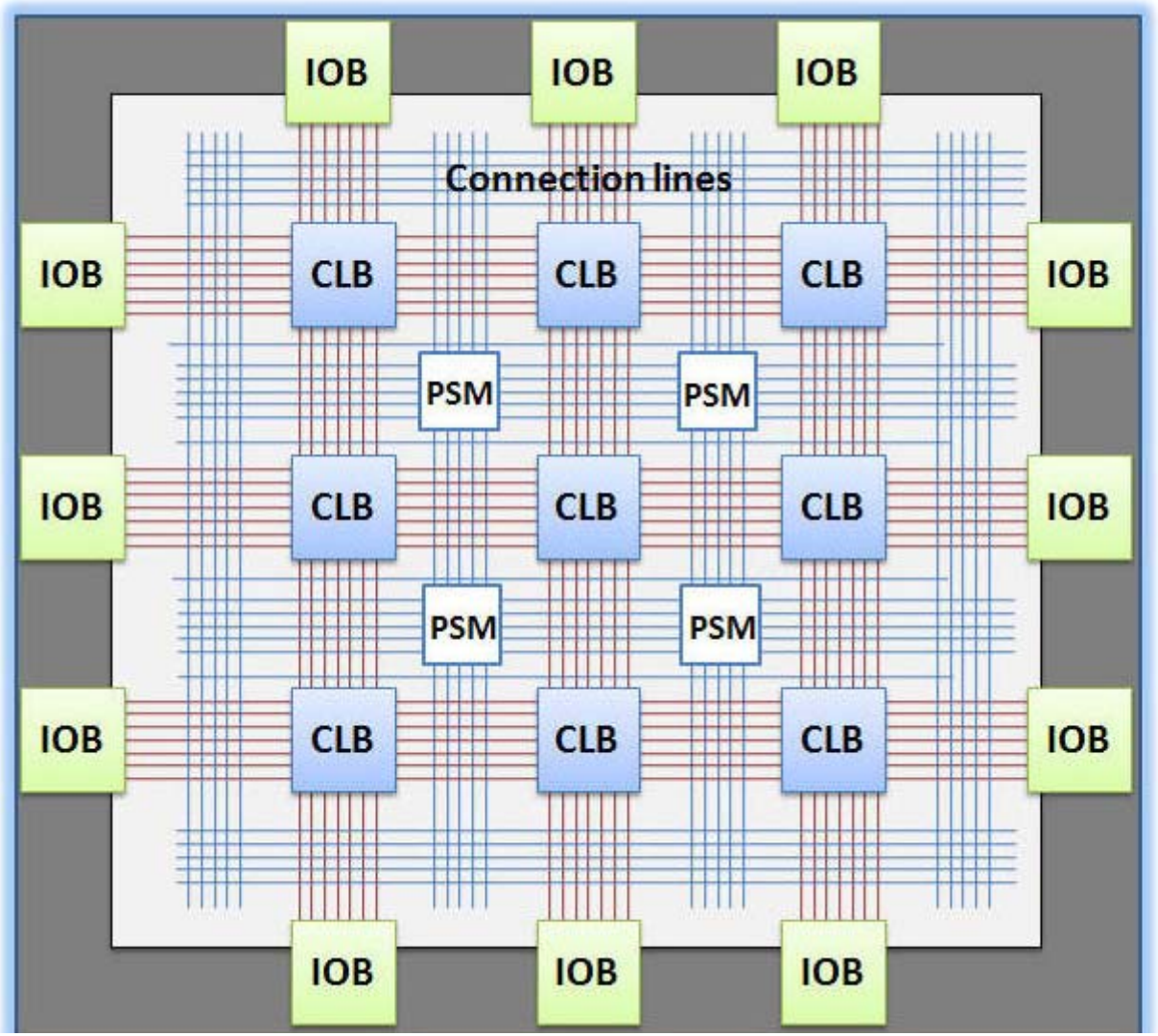
Verilog Code for 8th Order Lowpass FIR

```
module FIR_Lowpass #(parameter
order = 8,
word_size_in = 8,
Word_size_out = 2*word_size_in + 2,
b0 = 8'd7,
b1 = 8'd17,
b2 = 8'd32,
b3 = 8'd46,
b4 = 8'd52,
b5 = 8'd46,
b6 = 8'd32,
b7 = 8'd17,
b8 = 8'd7)(
output [word_size_out – 1 : 0]   Data_out,
input [word_size_in – 1 : 0]     Data_in,
input                            clock, reset
);
reg [word_size_in – 1 : 0]       Samples[1 : order];
integer                          k;
```

```
assign Data_out   =    b0 * Data_in
                     + b1 * Samples[1]
                     + b2 * Samples[2]
                     + b3 * Samples[3]
                     + b4 * Samples[4]
                     + b5 * Samples[5]
                     + b6 * Samples[6]
                     + b7 * Samples[7]
                     + b8 * Samples[8];

always @ (posedge clock)
  if (reset == 1) begin for (k = 1; k <= order; k = k + 1) Samples[k] <= 0; end
  else begin
    Samples[1] <= Data_in;
    for (k = 2; k <= order; k = k + 1) Samples[k] <= Samples[k – 1];
    end
endmodule
```

Xilinx Spartan 6 Device



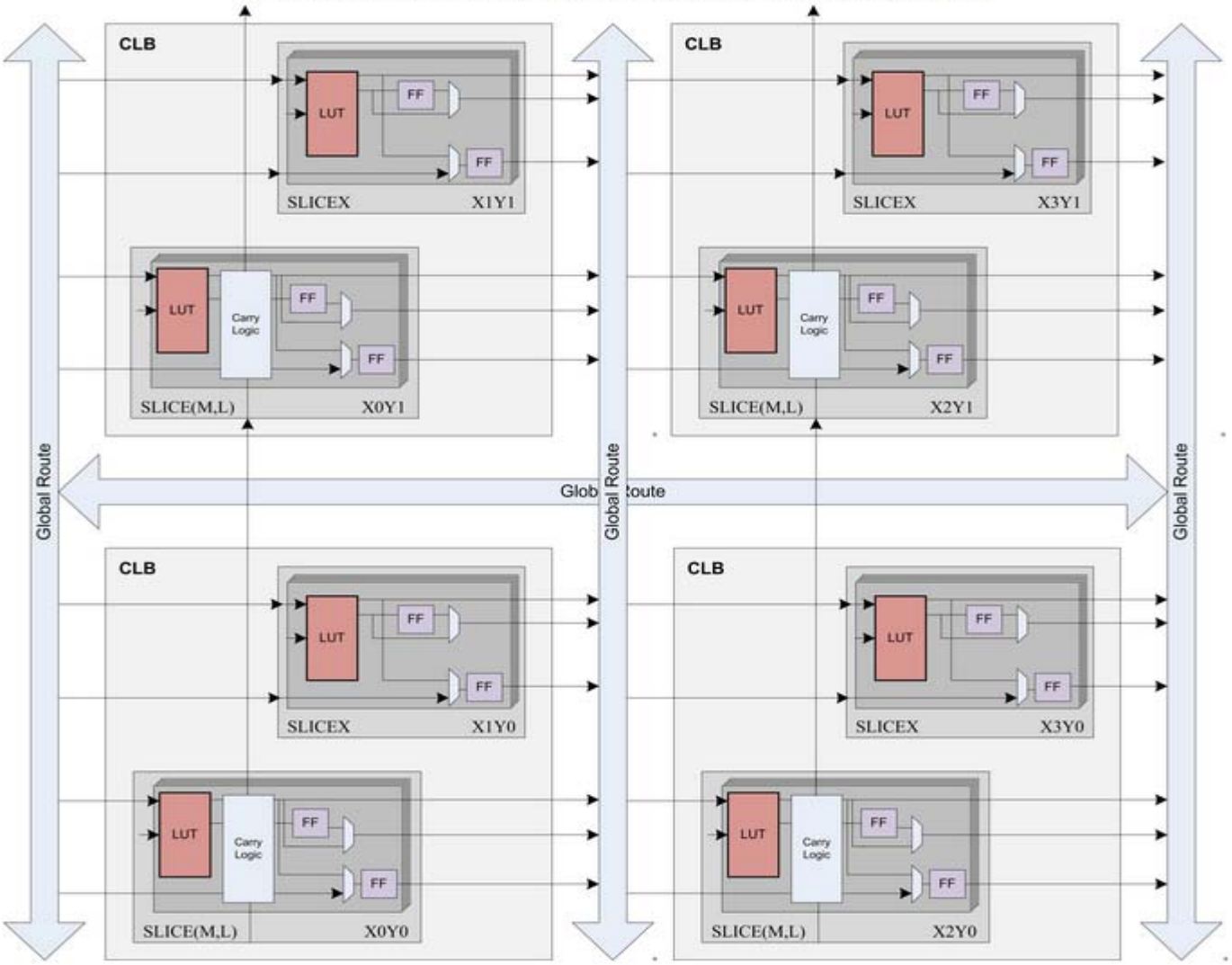
IOB
Input Output Block

CLB
Configurable
Logic Block

PSM
Programable
Switch Matrix

Connection lines
Single, Long
Double, Direct

Figure 1: CLB, Slice and LUT organization in the Spartan6 FPGAs



In addition, RAM & up to 180 DSP48A1 Slices - Each slice contains a fast 18 x 18 multiplier and a 48-bit accumulator capable of operating at 250MHz.

DSP48A1 Slice