



Intel® Agilex™ FPGA

10nm  
FPGA Fabric

EMIB

PCIe Gen 5

Output  
chipset

## Diseño Digital Avanzado

### Unidad 2 - Implementación de Filtros

**Dr. Ariel L. Pola**

[apola@fundacionfulgor.org.ar](mailto:apola@fundacionfulgor.org.ar)

September 10, 2022

# Tabla de Contenidos

## 1. Implementación de Filtros

# Implementación de Filtros



# Implementación de Filtros

## Simulación de Filtros FIR e IIR

### Efectos de cuantización

- Ejecutar los script de python con el objetivo de comprender los efectos de cuantización de los coeficientes.
  - fir\_filter\_direct\_form.ipynb
  - iir\_filter\_direct\_form.ipynb
  - IIR\_Filter\_Design.ipynb
- Instanciar y ejecutar el testbench de cada uno de los filtros.
  - filtro\_fir.v, tb\_filtro\_fir.v
  - iir.v, filter\_tb.v
  - iir\_top.v, iir\_filter.v, coeffSec1.v, coeffSec2.v, coeffSec3.v, tb\_iir\_filter.v

# Implementación de Filtros

## Implementación Filtro FIR en FPGA

### Primer modelo

- Implementar en FPGA el filtro FIR según los siguientes archivos
  - top\_design.v
  - signal\_generator.v
  - filtro\_fir.
  - SarTruncFP.v
- Agregar los IPs VIO e ILA para controlar en forma remota el diseño

# Implementación de Filtros

## Implementación Filtro FIR en FPGA

### Laboratorio

- Considerar un sistema de transmisión compuesto por una señal senoidal y un filtro pasa bajo con las siguientes características:
  - Señal senoidal compuesta por dos frecuencias  $f_1 = 17kHz$  ( $A = 0.5$ ) y  $f_2 = 1.5kHz$  ( $A = 1.0$ )
  - Frecuencia de muestreo  $f_s = 48kHz$
  - Filtro pasa bajo con frecuencia de corte  $f_{cut} = 8kHz$

# Implementación de Filtros

## Implementación Filtro FIR en FPGA

### Laboratorio

#### ■ Desarrollo del modelo

- 1 Utilizando el scripts de Python `coeff.ipynb`, determinar los coeficientes del filtro para una frecuencia de corte de  $f_{cut} = 8\text{khz}$ . El filtro debe tener una longitud de 15 coeficientes.
- 2 Realizar el diagrama en bloques del filtro.
- 3 Generar un proyecto con los archivos entregador por la cátedra con la herramienta Vivado.
- 4 Configurar el archivo `mem.hex` con las señales senoidales especificadas previamente utilizando el script `genmem.py`.
- 5 Generar los coeficientes del filtro utilizando el script `coeff.ipynb` para los siguientes valores de frecuencias de corte  $f_{cut} = 0.5\text{khz}, 8\text{khz}, 18\text{khz}$ .
- 6 Configurar el filtro en verilog con los valores de los coeficientes cuantizados (sintetizar cada filtro por separado).
- 7 Implementar en FPGA y graficar las señales senoidales pre y pos filtradas.