DSP com FPGAs

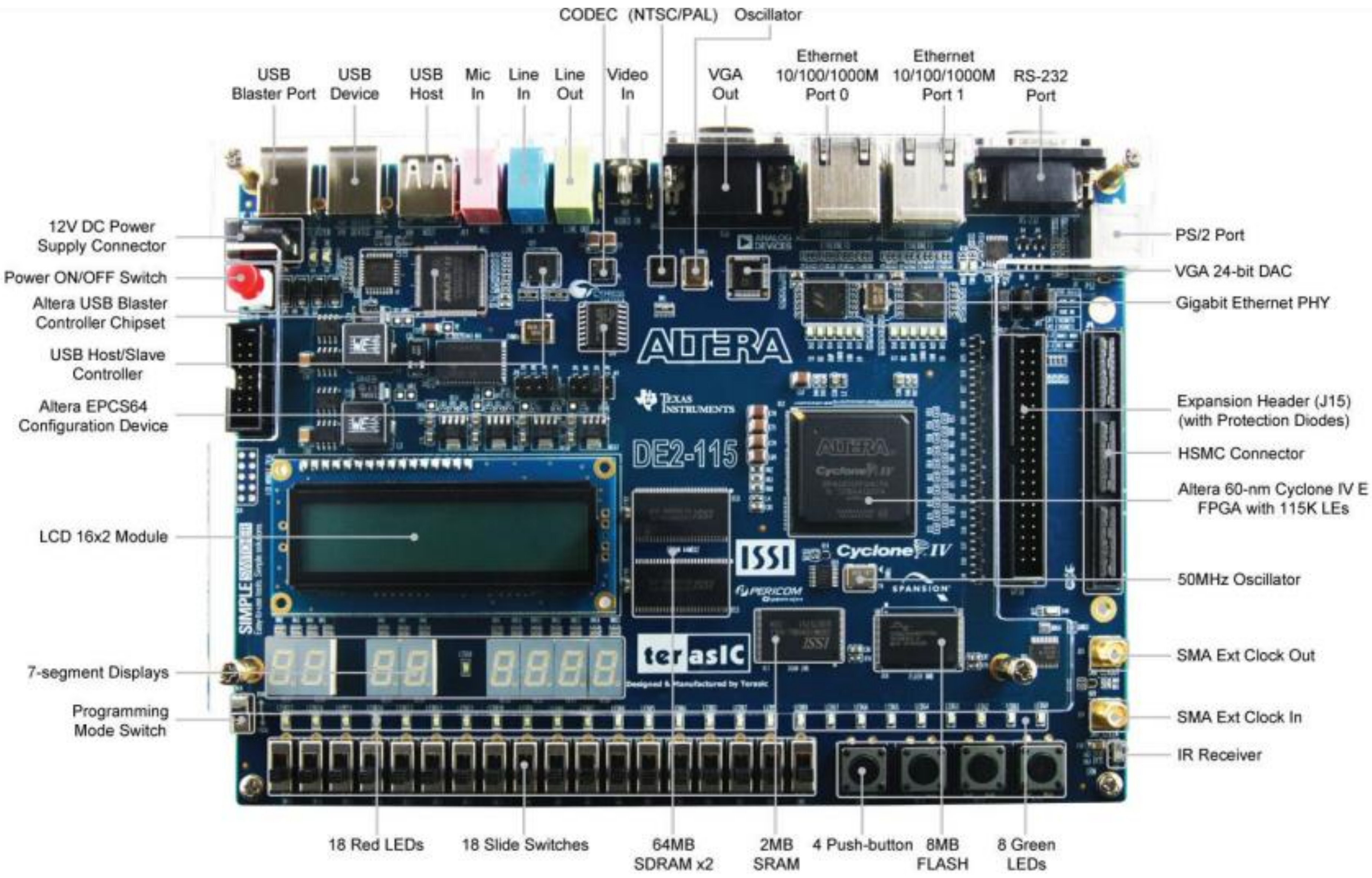
PROF. MAURÍCIO ACCONCIA DIAS

15 SEMANA DA ENGENHARIA - FHO

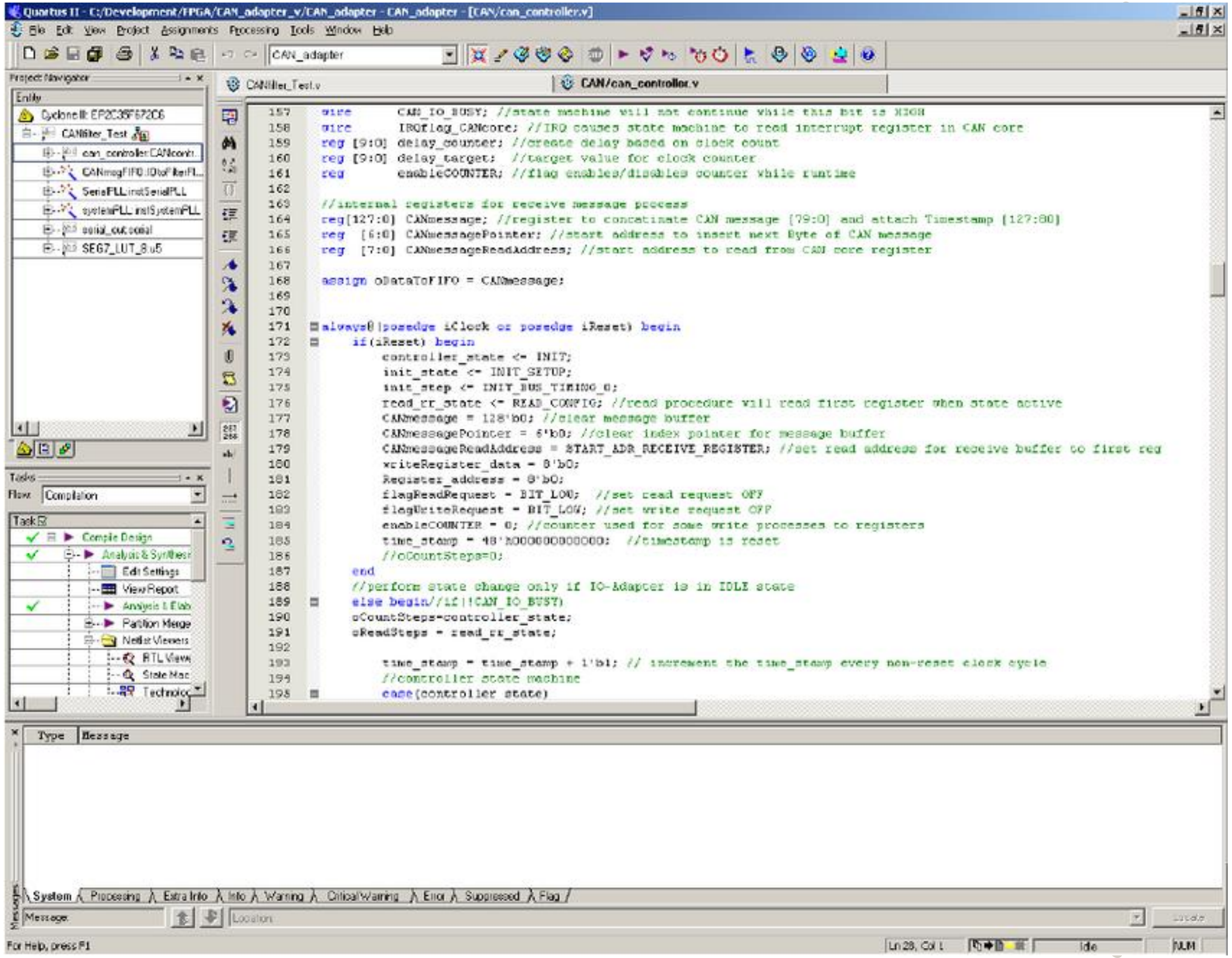
Tópicos do minicurso

- Materiais e métodos
- FPGA
- DSP
- Implementação – Análise dos módulos
- Execução

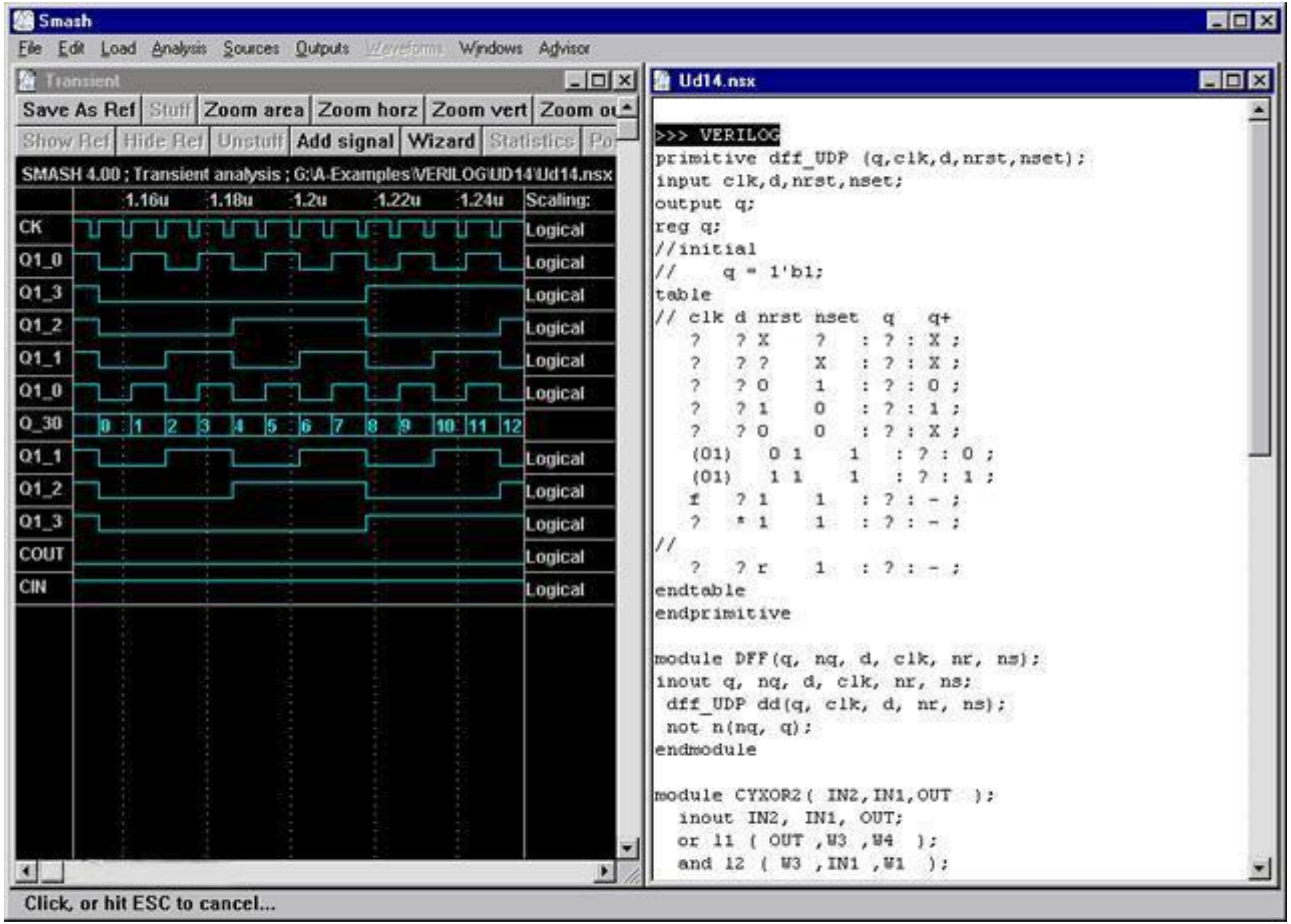
Materiais e métodos

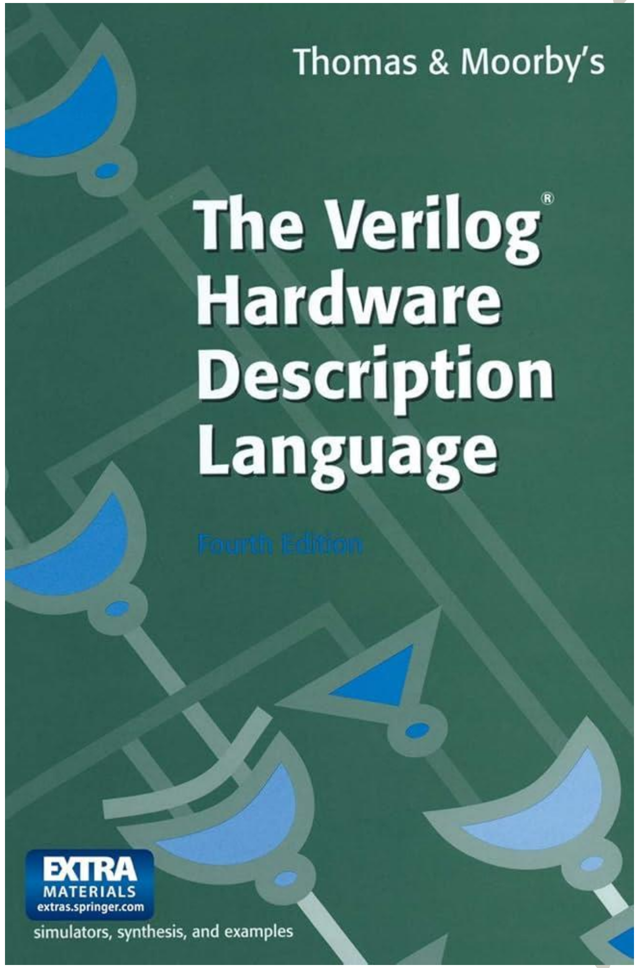


Materiais e métodos

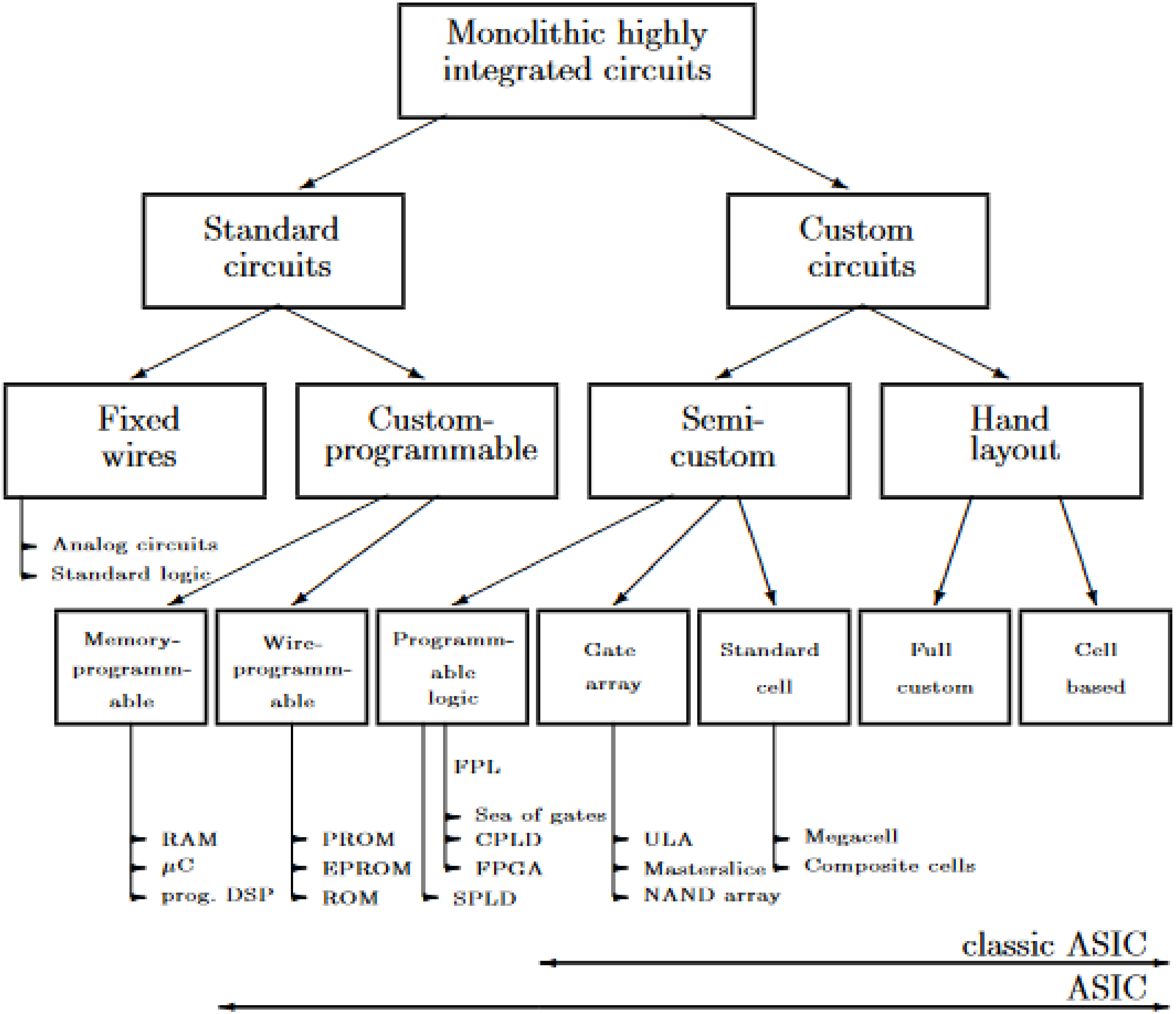


Materiais e métodos





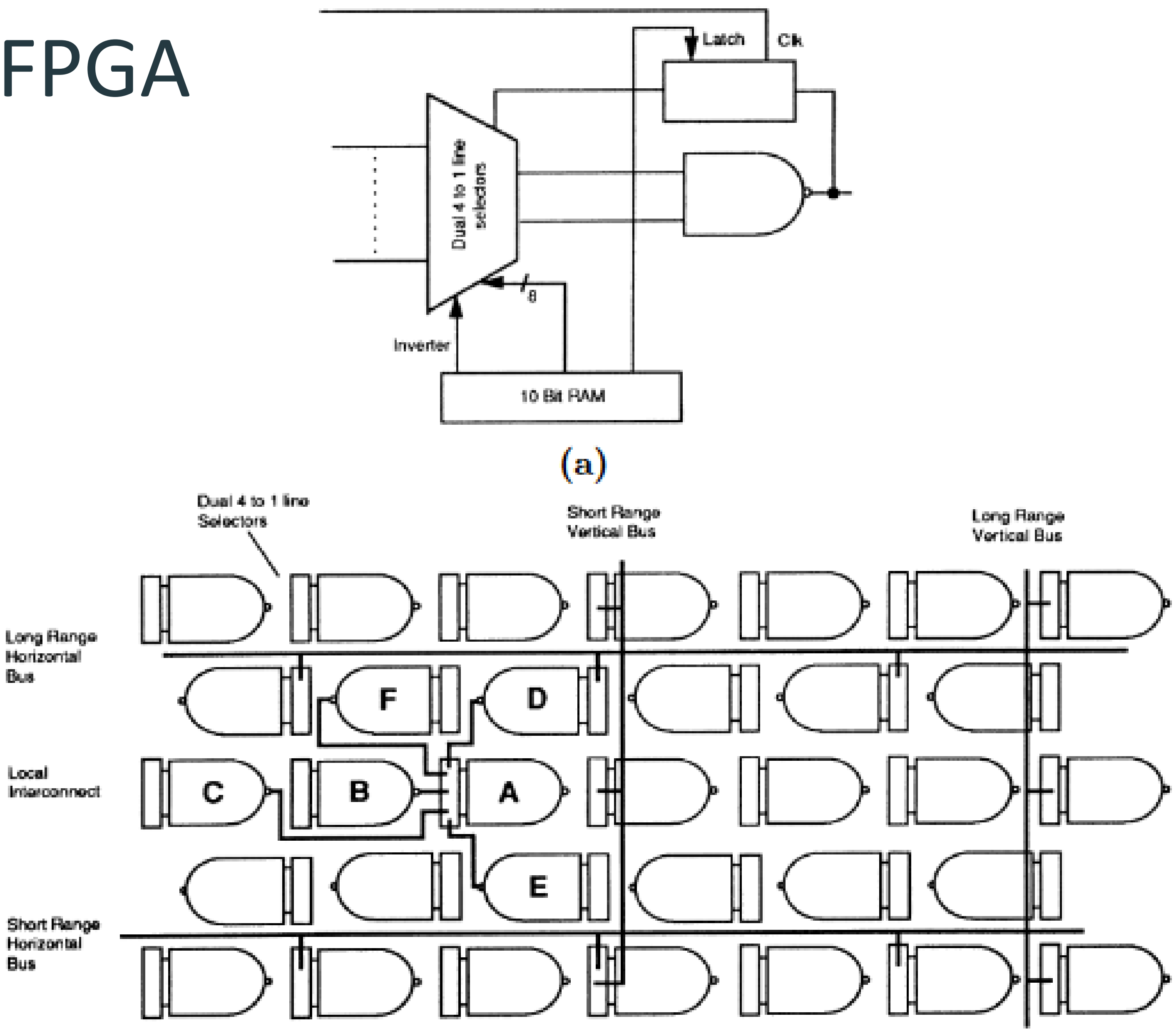
Técnicas de implantação

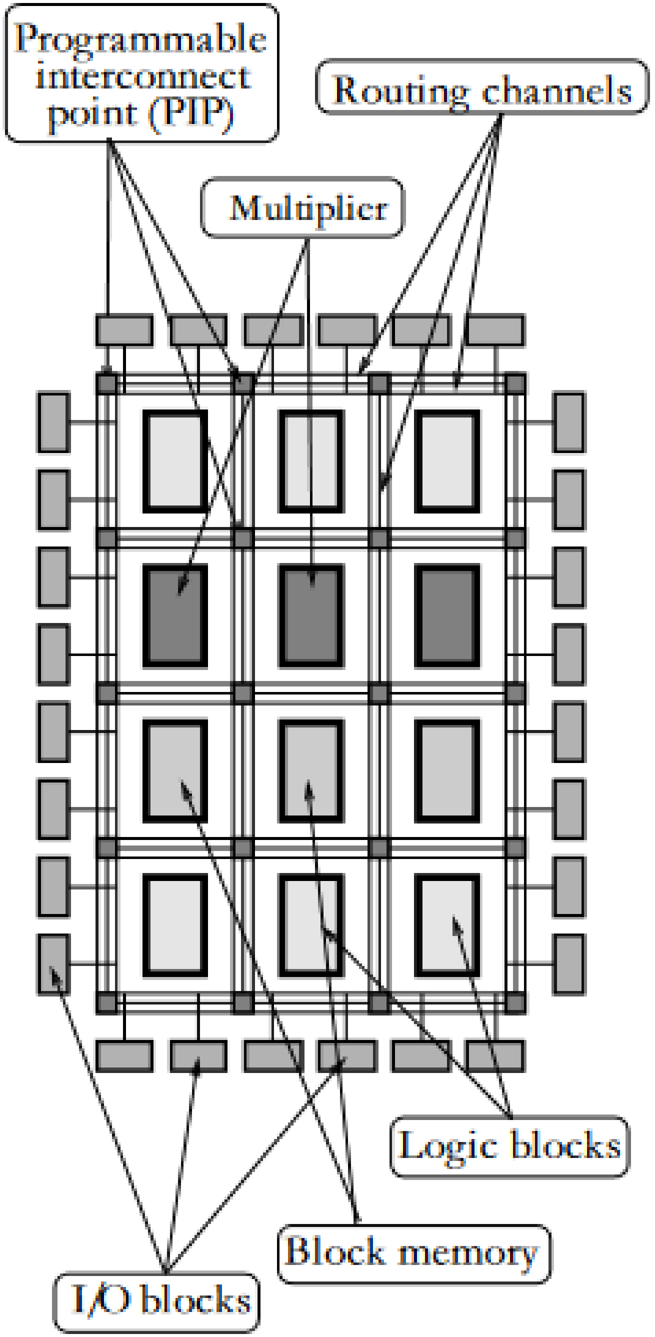


FPGA

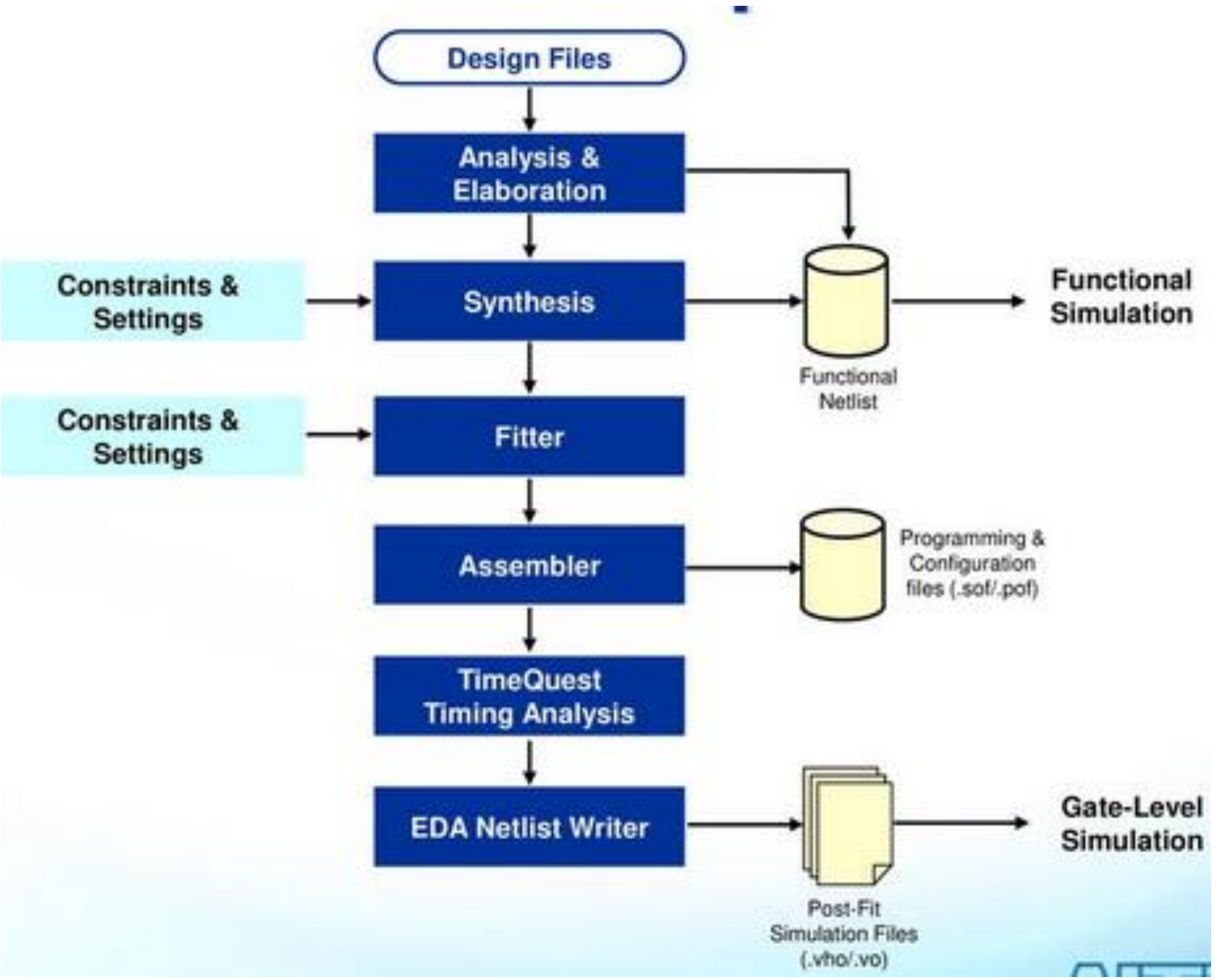
```
EP4CE115F29C7
 |    | | |||
 |    | | |||--> Speed grade 7 (lower is faster)
 |    | | ||---> Commercial temperature 0-85 Celsius
 |    | | |----> Package with 780 pins
 |    | |------> Package FineLine BGA
 |    |--------> LEs in 1000
 |-------------> Device family: Cyclone IV E
```

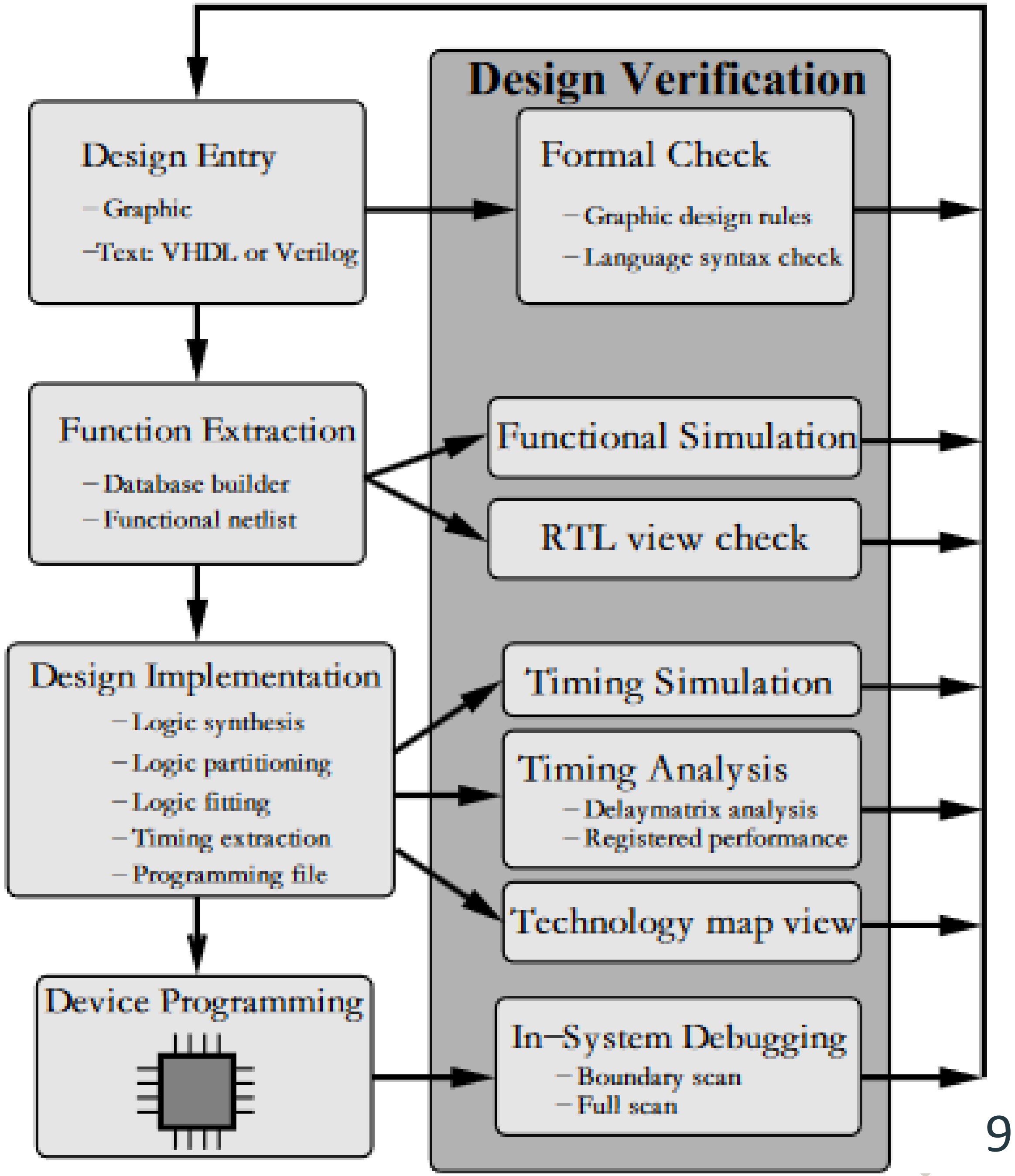
FPGA


Latch
Clk
Dual 4 to 1 line selectors
8
Inverter
10 Bit RAM
(a)
Dual 4 to 1 line Selectors
Short Range Vertical Bus
Long Range Vertical Bus
Long Range Horizontal Bus
F
D
Local Interconnect
C
B
A
E
Short Range Horizontal Bus

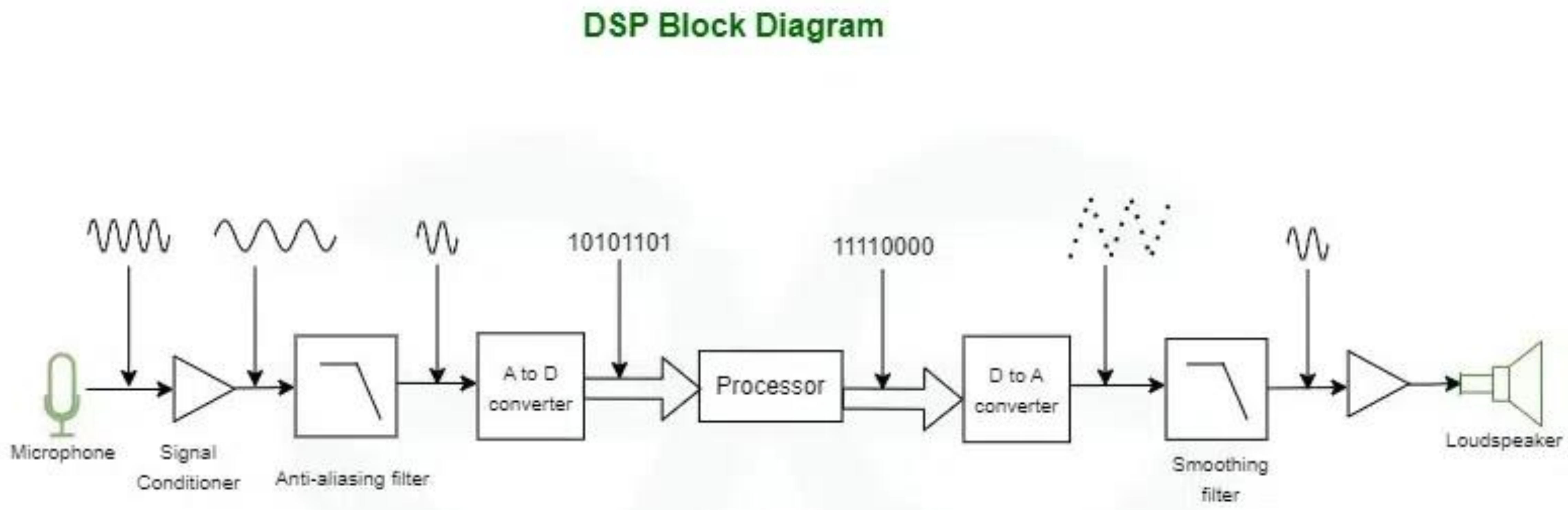

Programmable interconnect point (PIP)
Routing channels
Multiplier
Logic blocks
Block memory
I/O blocks

Design no FPGA





DSP



Fonte: https://www.geeksforgeeks.org/what-is-digital-signal-processing/

DSP

- **Os fundamentos do Processamento de Sinais Digitais incluem termos importantes que ajudam a entender a manipulação de sinais:**
  - **Amostragem: É o processo que amostra o sinal analógico contínuo em um sinal digital.**
  - **Quantização: É o processo que atribui números digitais ao sinal analógico calculado. Ele transforma um grupo de valores medidos em um conjunto finito.**
  - **Transformada de Fourier Discreta (DFT): Este método transforma um sinal de tempo discreto em seu domínio de frequência. Ele ajuda a entender as várias frequências presentes em um sinal.**
  - **Transformada Rápida de Fourier (FFT): Este algoritmo é bastante eficiente e realiza a DFT rapidamente. Além disso, é uma técnica avançada da DFT que auxilia na exploração de sinais de forma rápida e mais produtiva.**

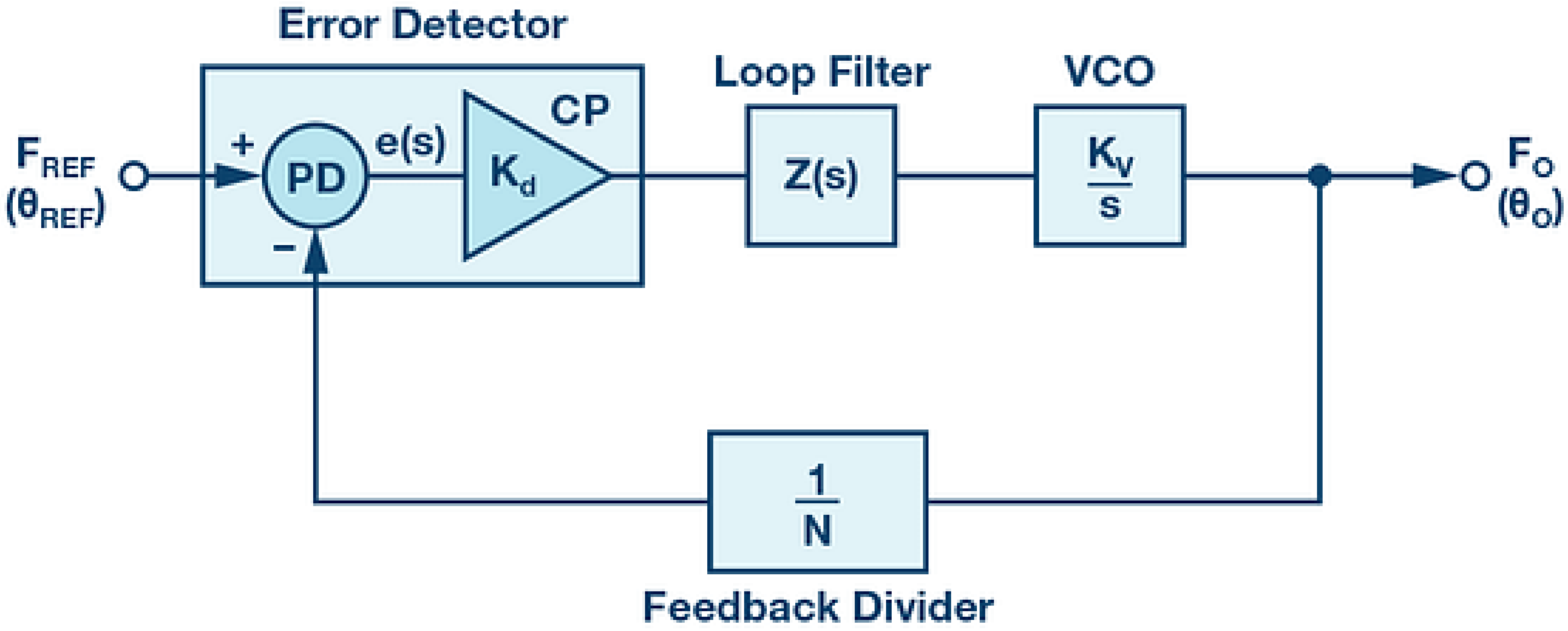
Clock e sincronismo

- O hardware de um DSP deve ser corretamente sincronizado para que seja possível aplicar cada uma das etapas corretamente e para que a saída seja obtida da forma correta

- A forma de sincronizar diversos blocos de hardware em um circuito é utilizando um sinal de clock comum

- Porém iremos precisar de diversos clocks diferentes para cada um dos blocos. Isso é obtido através de um circuito chamado de PLL – Phase Locked Loop

PLL



Fonte: https://medium.com/for-engineers/image-for-analog-devices-82d372e11cb6

FAST FOURIER TRANSFORM

$$F_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi kn}{N}}$$

$k: 0, \dots, \#\ of\ samples$ $n: 0, \dots, \#\ of\ samples$

(1 operation) 1 sample: $F_0 = x_0$*exponential

(4 operations) 2 samples: $F_0 = x_0$*exponential + x_1*exponential
$F_1 = x_0$*exponential + x_1*exponential

(9 operations) 3 samples: $F_0 = x_0$*exponential + x_1*exponential + x_2*exponential
$F_1 = x_0$*exponential + x_1*exponential + x_2*exponential
$F_2 = x_0$*exponential + x_1*exponential + x_2*exponential

$$F_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi kn}{N}}$$

$$F_k = \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-\frac{j2\pi k(2m)}{N}} + \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-\frac{j2\pi k(2m+1)}{N}}$$

$x_0, x_2, x_4, \ldots$

EVEN INDEX

$x_1, x_3, x_5, \ldots$

ODD INDEX

$$F_k = \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-\frac{j2\pi km}{N/2}} + C_k \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-\frac{j2\pi km}{N/2}}$$

$$\downarrow$$

$$\underbrace{\cos\left(\frac{-2\pi km}{N/2}\right)} + j\sin\left(\frac{-2\pi km}{N/2}\right) \qquad k: 0, \ldots, N \ (integers)$$

$$\underbrace{\cos\left(-2\pi m - \frac{2\pi mr}{N/2}\right)} \qquad m: 0, \ldots, N/2 \ (integers)$$

$$\cos\left(-\frac{2\pi mr}{N/2}\right) = \cos\left(-\frac{2\pi m(N/2 + r)}{N/2}\right)$$

$$F_k = \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-\frac{j2\pi km}{N/2}} + C_k \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-\frac{j2\pi km}{N/2}}$$

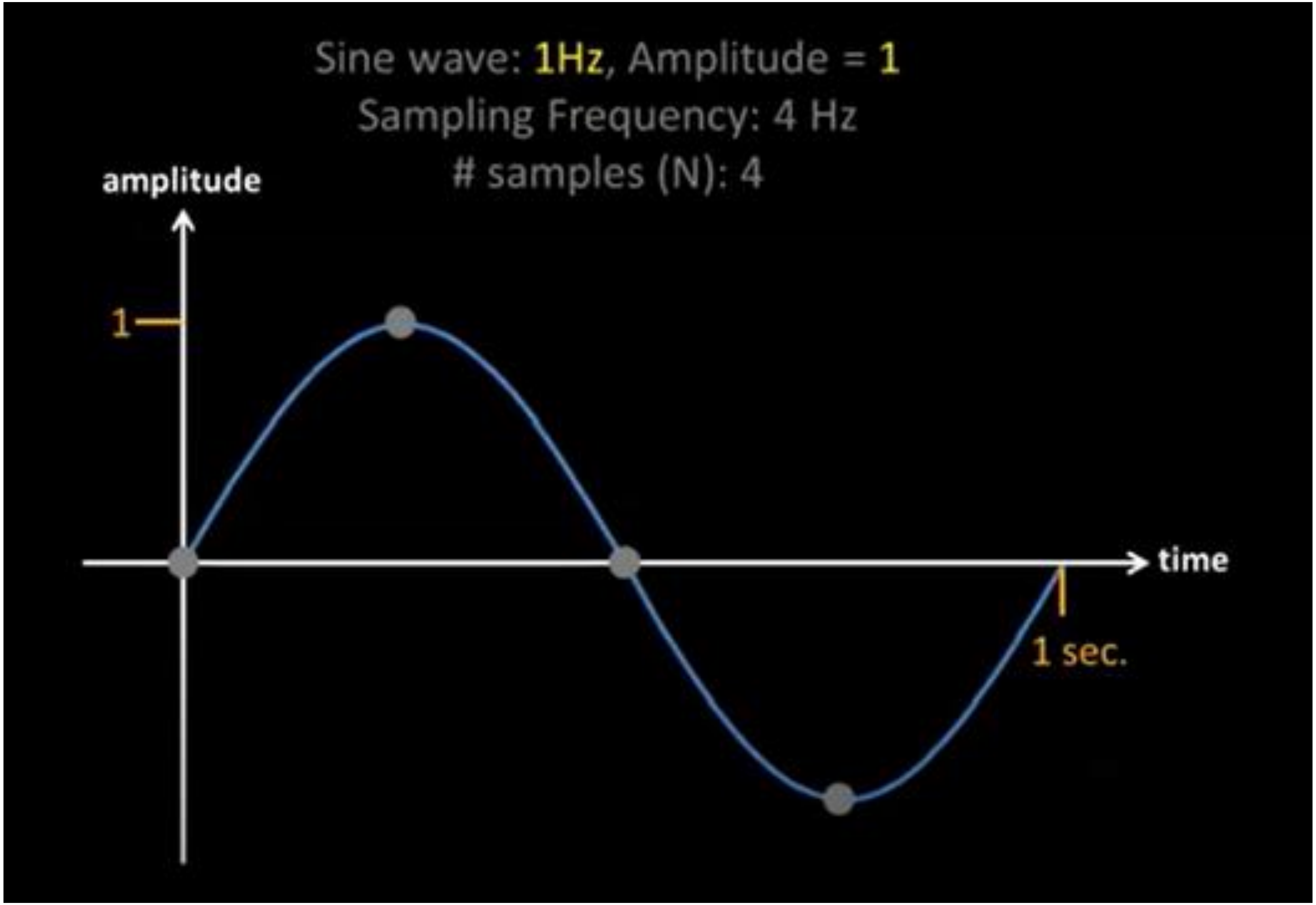
SYMMETRY IDENTITY

$$\cos\left(-\frac{2\pi km}{N/2}\right) = \cos\left(-\frac{2\pi(N/2+k)m}{N/2}\right)$$

$$\sin\left(-\frac{2\pi km}{N/2}\right) = \sin\left(-\frac{2\pi(N/2+k)m}{N/2}\right)$$

$$e^{-\frac{j2\pi km}{N/2}} = e^{-\frac{j2\pi(N/2+k)m}{N/2}}$$

$k: 0, \dots, N$


Sine wave: 1Hz, Amplitude = 1
Sampling Frequency: 4 Hz
samples (N): 4
amplitude
1
time
1 sec.

$$F_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi kn}{N}}$$

$$\sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-\frac{j2\pi km}{N/2}} \qquad e^{\frac{-j2\pi k}{N}} \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-\frac{j2\pi km}{N/2}}$$

$$\boxed{\sum_{m=0}^{N/4-1} x_{4m} \cdot e^{-\frac{j2\pi km}{N/4}}} \; e^{\frac{-4\pi k}{N}} \boxed{\sum_{m=0}^{N/4-1} x_{4m+2} \cdot e^{-\frac{j2\pi km}{N/4}}} \qquad \boxed{\sum_{m=0}^{N/4-1} x_{4m+1} \cdot e^{-\frac{j2\pi km}{N/4}}} \; e^{\frac{-j4\pi k}{N}} \boxed{\sum_{m=0}^{N/4-1} x_{4m+3} \cdot e^{-\frac{j2\pi km}{N/4}}}$$

$$F_0 = x_0 + x_2 + x_1 + x_3 = \underbrace{0 + 1}_{F_0^{e} = 1} + \underbrace{0 + (-1)}_{F_0^{o} = -1} = 0$$

$$F_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi kn}{N}}$$

$$\sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-\frac{j2\pi km}{N/2}} \qquad e^{\frac{-j2\pi k}{N}} \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-\frac{j2\pi km}{N/2}}$$

$$\sum_{m=0}^{N/4-1} x_{4m} \cdot e^{-\frac{j2\pi km}{N/4}} \quad e^{\frac{-4\pi k}{N}} \sum_{m=0}^{N/4-1} x_{4m+2} \cdot e^{-\frac{j2\pi km}{N/4}} \qquad \sum_{m=0}^{N/4-1} x_{4m+1} \cdot e^{-\frac{j2\pi km}{N/4}} \quad e^{\frac{-j4\pi k}{N}} \sum_{m=0}^{N/4-1} x_{4m+3} \cdot e^{-\frac{j2\pi km}{N/4}}$$

$$F_1 = x_0 + e^{\frac{-j4\pi k}{N}} x_2 + e^{\frac{-j2\pi k}{N}} \left(x_1 + e^{\frac{-j4\pi k}{N}} x_3 \right)$$

$$= \underbrace{0 + 1(0)}_{F_1^{\mathrm{e}} = 0} + (-j)\underbrace{(1 + (-1)(-1))}_{F_1^{\mathrm{o}} = 2} = -2j$$

Completed:
$F_0 = 0$
$F_1 = -2j$

$$F_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi kn}{N}}$$

$$\boxed{\sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-\frac{j2\pi km}{N/2}}} \qquad e^{\frac{-j2\pi k}{N}} \boxed{\sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-\frac{j2\pi km}{N/2}}}$$

$$F_0^e = 1 = F_2^e \qquad F_0^o = -1 = F_2^o$$

$$F_1^e = 0 = F_3^e \qquad F_1^o = 2 = F_3^o$$

$$F_2 = 1 + (1)(-1) = 0$$

$$F_3 = 0 + j(2) = 2j$$

Completed:

$F_0 = 0$

$F_1 = -2j$

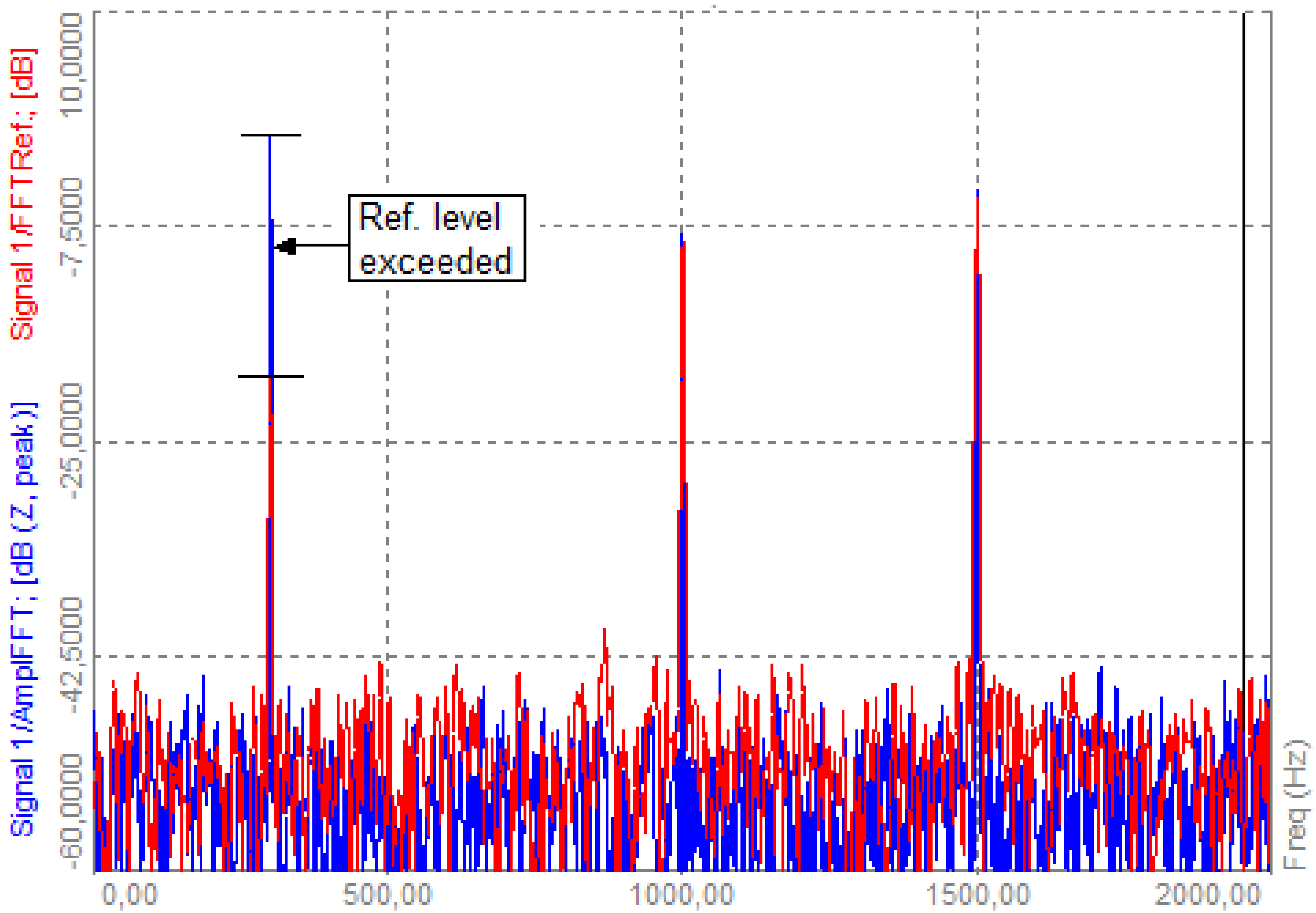
$F_2 = 0$

$F_3 = 2j$

Uso da FFT em áudio

- A "Transformada Rápida de Fourier" (FFT) é um método de medição importante na ciência de áudio e medição acústica.
- Ela converte um sinal em componentes espectrais individuais, fornecendo assim informações de frequência sobre o sinal. Facilita a filtragem de ruídos e a melhoria da qualidade do sinal em aplicações de áudio e comunicação.
- As FFTs são usadas para análise de falhas, controle de qualidade e monitoramento de condição de máquinas ou sistemas.

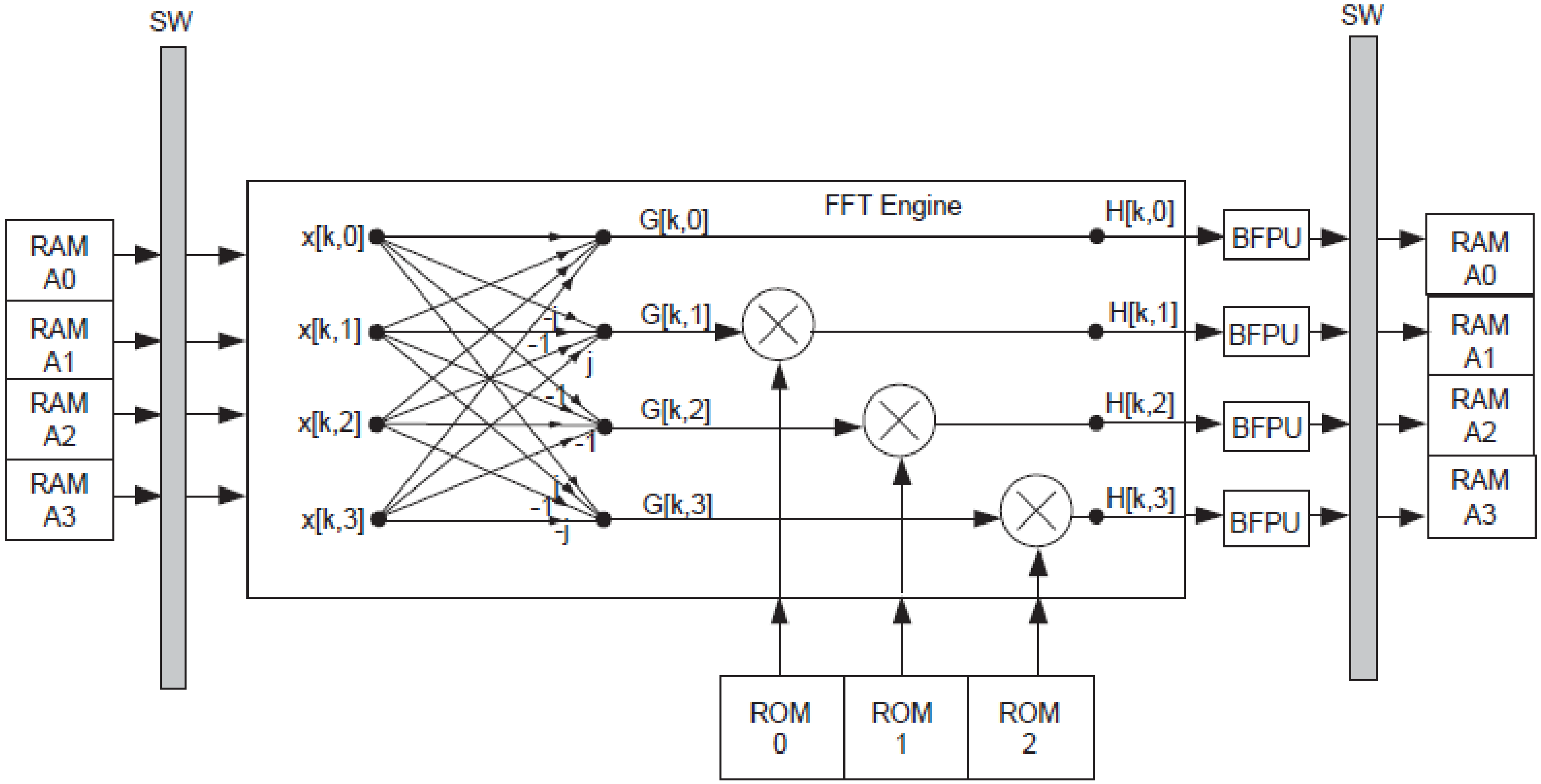
Espectro FFT



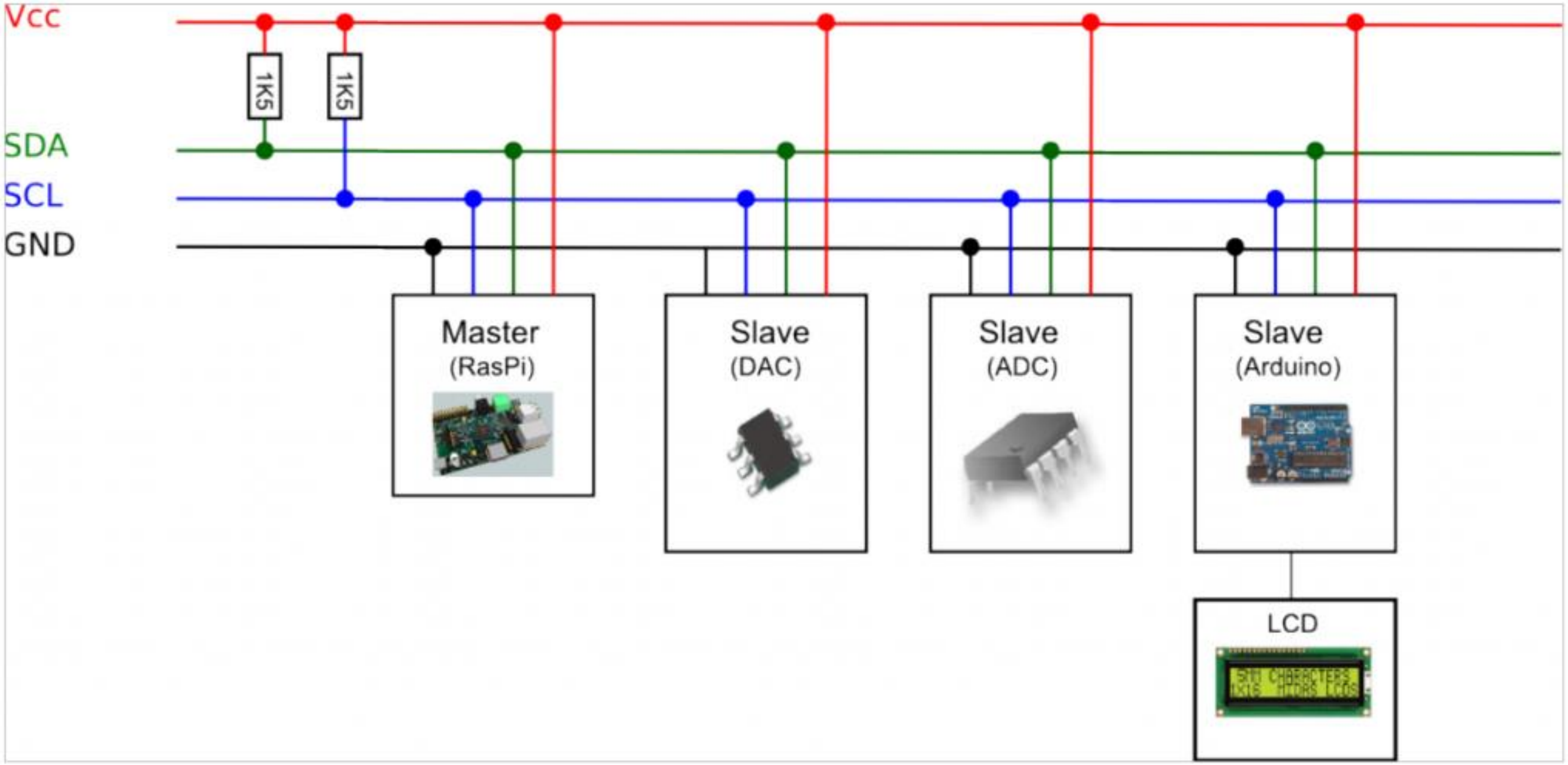
Altera Megacore Functions

- As funções MegaCore® são blocos reutilizáveis de propriedade intelectual (IP) que você pode personalizar para seu design. Use esses blocos de IP parametrizados para reduzir tanto o tempo de teste quanto o tempo de design.
- Essas funções pré-testadas são otimizadas para dispositivos Intel. Modelos de simulação funcional de IP em VHDL e Verilog HDL estão incluídos para design e depuração com ferramentas de simulação EDA suportadas pela Intel.

FFT Megacore altera



Componentes do design – i²c

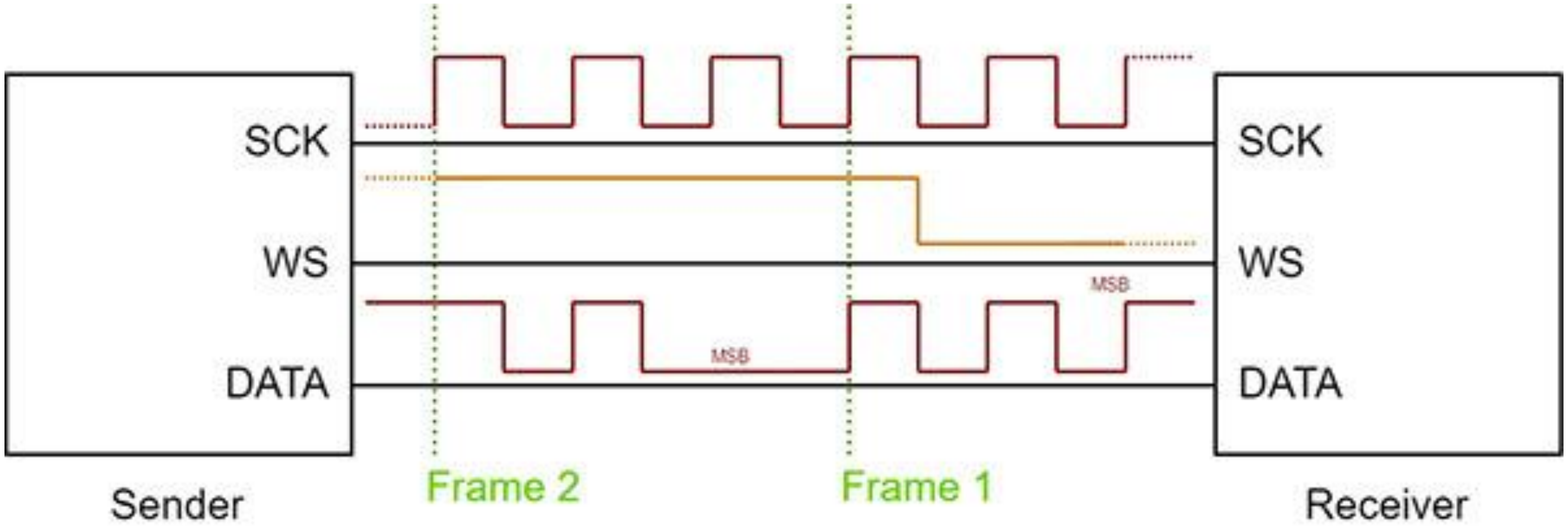


Componentes do design – i²c

- O protocolo I2C adota o tipo de comunicação mestre-escravo, sendo o mestre, responsável por efetuar a requisição de dados/tarefas dos escravos.

Componentes do design – i²s

- I2S é um protocolo de comunicação serial e síncrono para a troca de dados de áudio digital entre dispositivos de processamento de som, como microcontroladores, codecs de áudio ou processadores de sinal digital.



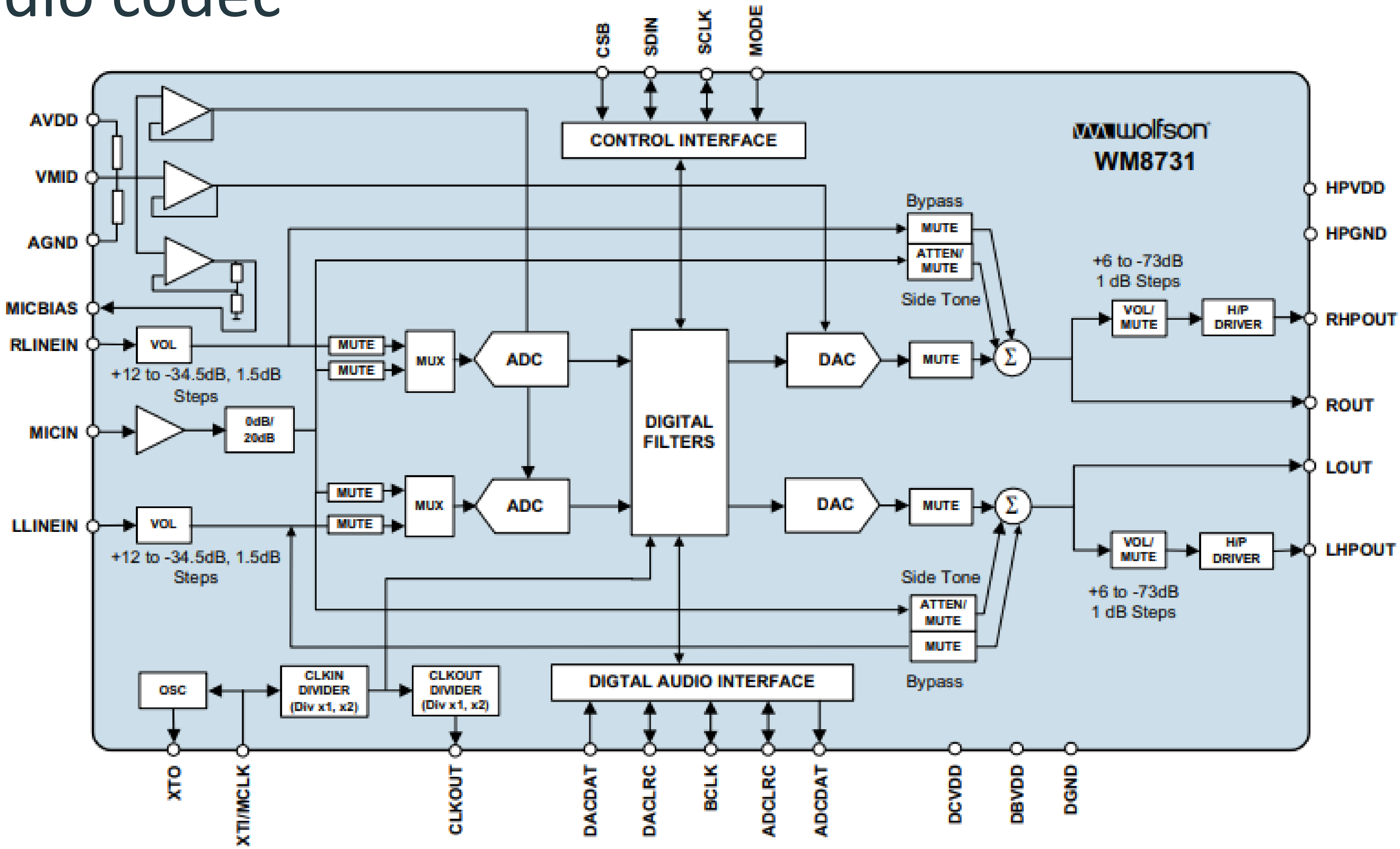
Componentes do design – i²s

- O sinal de clock funciona continuamente, e os bits que representam os dados de áudio também são colocados continuamente na única linha de dados.

- O transmissor ativo envia quaisquer dados disponíveis para transmissão, enquanto o receptor lê quantos bits puder — ou quantos espera receber. O remetente também pode enviar os bits de dados na borda de descida ou de subida do sinal SCK, mas os receptores devem ler os dados na borda de subida do sinal.

- A única limitação é que o remetente deve transmitir os dados com o bit mais significativo (MSB) primeiro.

Audio codec

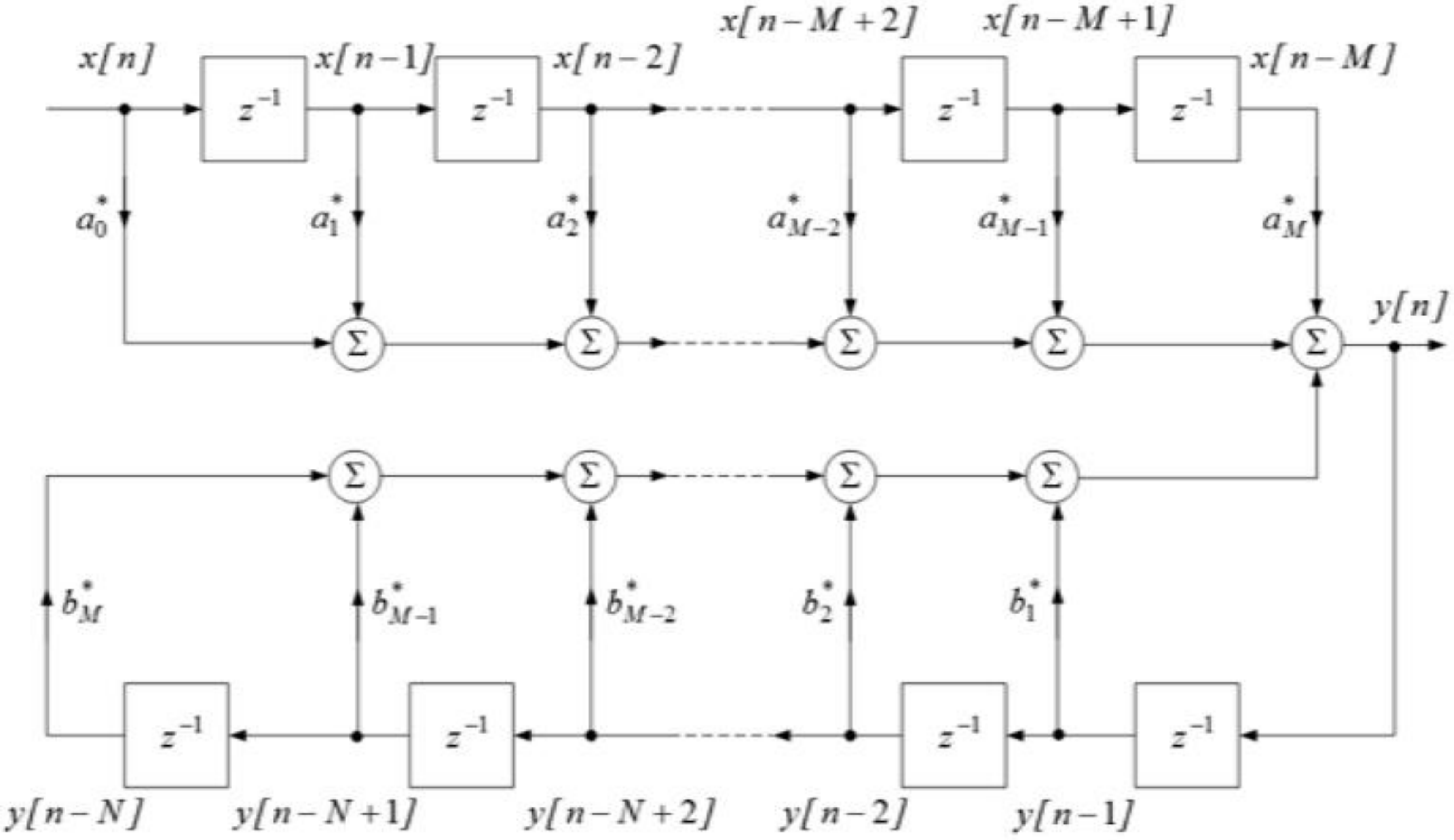


CSB
SDIN
SCLK
MODE
CONTROL INTERFACE
wolfson
WM8731
AVDD
VMID
AGND
MICBIAS
RLINEIN
VOL
+12 to -34.5dB, 1.5dB Steps
MICIN
0dB/ 20dB
LLINEIN
VOL
+12 to -34.5dB, 1.5dB Steps
MUTE
MUTE
MUX
ADC
MUTE
MUTE
MUX
ADC
DIGITAL FILTERS
DAC
DAC
MUTE
MUTE
Bypass
MUTE
ATTEN/ MUTE
Side Tone
Side Tone
ATTEN/ MUTE
MUTE
Bypass
+6 to -73dB 1 dB Steps
VOL/ MUTE
H/P DRIVER
VOL/ MUTE
H/P DRIVER
+6 to -73dB 1 dB Steps
HPVDD
HPGND
RHPOUT
ROUT
LOUT
LHPOUT
OSC
CLKIN DIVIDER (Div x1, x2)
CLKOUT DIVIDER (Div x1, x2)
DIGTAL AUDIO INTERFACE
XTO
XTI/MCLK
CLKOUT
DACDAT
DACLRC
BCLK
ADCLRC
ADCDAT
DCVDD
DBVDD
DGND

Componentes do design – Filtro IIR

- **Infinite Impulse Response Filter**
- Este tipo de filtro digital calcula a saída baseado não apenas nos sinais de entrada, mas também com base nas saídas anteriores. Ou seja, há um componente recursivo neste tipo de filtro.
- Por conta disso, ele pode gerar uma saída mesmo na ausência de sinal de entrada, daí a qualidade "infinita". Um filtro IIR pode funcionar como oscilador ou gerador de sinais, por exemplo. Essa resposta é composta por senoides com amplitude que decaem exponencialmente.

Componentes do design – Filtro IIR



Componentes do design – Filtro IIR

$$y[k] = a_k x[k] + a_{k-1} x[k-1] + a_{k-2} x[k-2] + \ldots$$
$$+ b_{k-1} y[k-1] + b_{k-2} y[k-2] + \ldots$$

⇨ Representação de um filtro **recursivo IIR**

Os coeficiente dos filtros IIR são desenvolvidos baseados na transformada-z.

Componentes do design – Filtro IIR

- Um exemplo prático de uso de um filtro IIR (Infinite Impulse Response) em áudio é na equalização de som em sistemas de áudio, como em mixers ou softwares de edição de áudio.

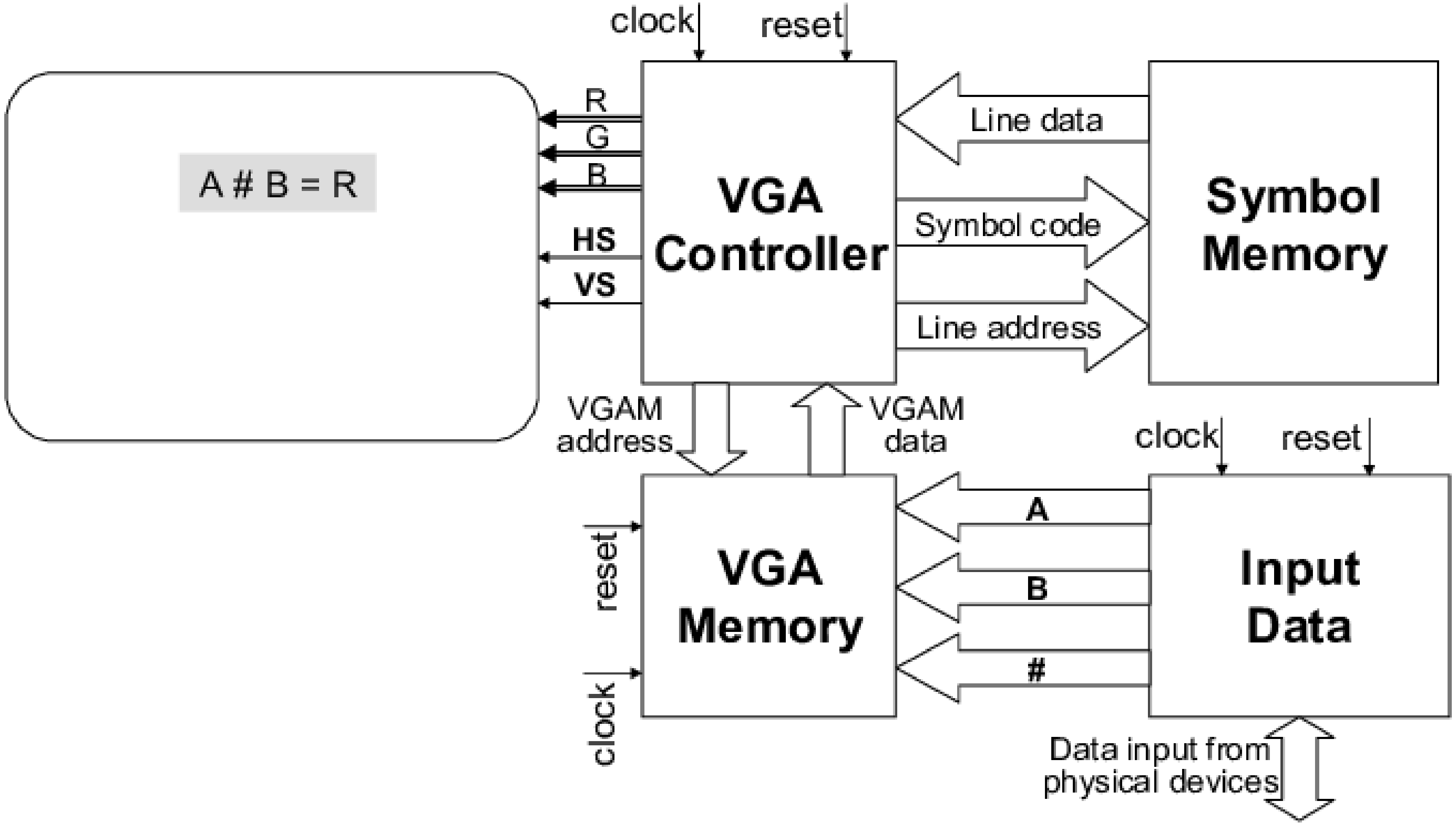
- Imagine que você está mixando uma faixa de música e percebe que a voz do cantor está um pouco abafada. Para resolver isso, você pode usar um filtro IIR para realçar as frequências médias e altas, que são importantes para a clareza da voz.

Exemplos

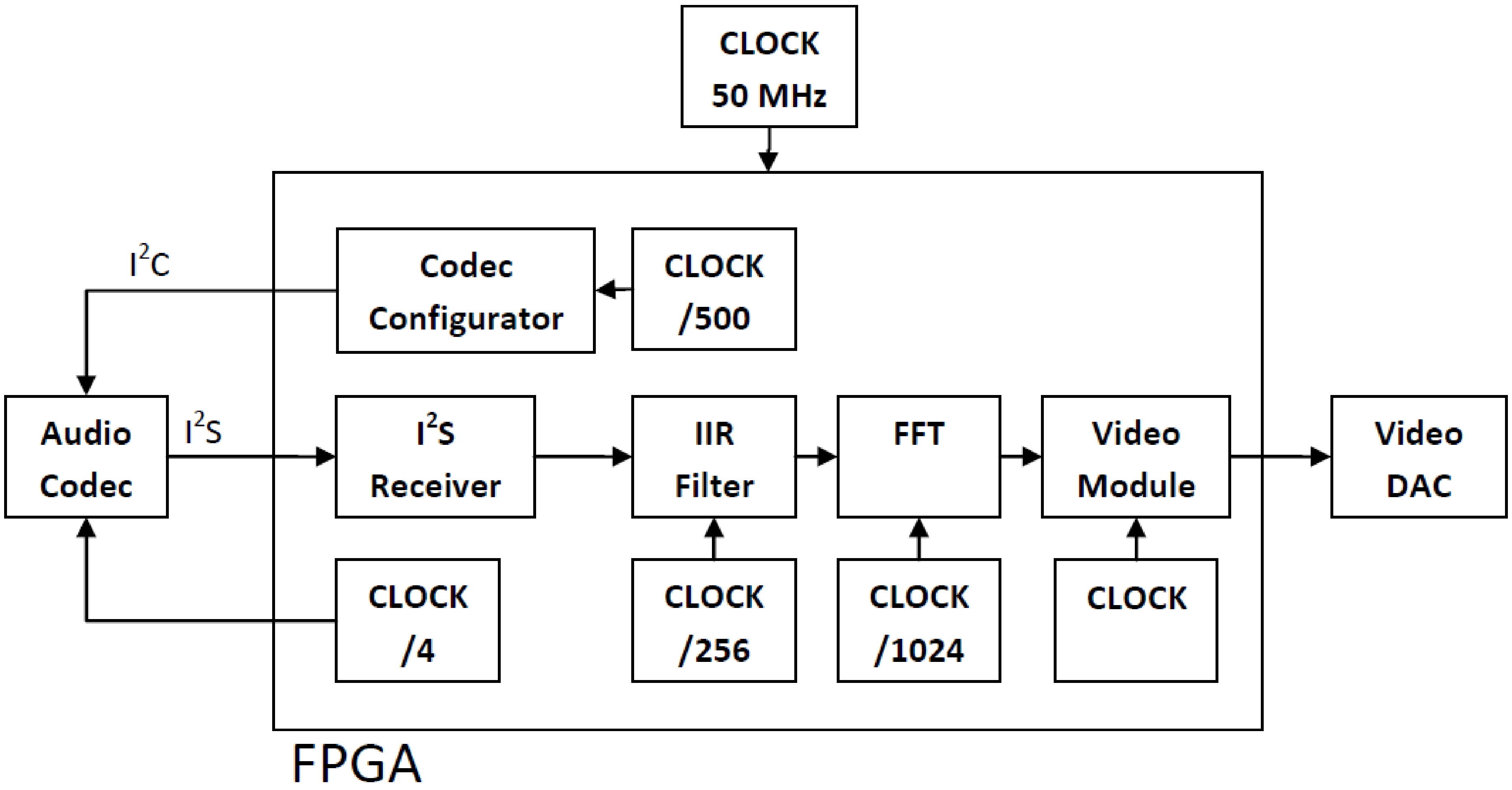
```
# IIR filter coefficients

freq = 2000 # Hz

r = 0.98

a1 = -2.0 * r * math.cos(freq / (SAMPLE_RATE / 2.0) * math.pi)

a2 = r * r

filter = [a1, a2]
```

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + b_4 z^{-4} + b_5 z^{-5} + b_6 z^{-6}}{1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3} + a_4 z^{-4} + a_5 z^{-5} + a_6 z^{-6}}$$

Controlador VGA



Finalmente...



CLOCK
50 MHz
I²C
Codec
Configurator
CLOCK
/500
Audio
Codec
I²S
I²S
Receiver
IIR
Filter
FFT
Video
Module
Video
DAC
CLOCK
/4
CLOCK
/256
CLOCK
/1024
CLOCK
FPGA

Vamos para a implementação

- Baixem a pasta toda do github
- Descompatem
- Abram o projeto do quartus e vamos sintetizar o hardware...

Bibliografia

- Artigos e manuais na pasta do github
- https://epxx.co/artigos/iirfilter_pt.html
- https://www.ece.ufrgs.br/~eng04006/aulas/aula24.pdf
- https://www.intel.com/content/www/us/en/programmable/quartushelp/17.0/reference/glossary/def_megacore.htm
- https://www.youtube.com/watch?v=htCj9exbGo0

DSP com FPGAs

PROF. MAURÍCIO ACCONCIA DIAS

15 SEMANA DA ENGENHARIA - FHO