ISE

VHDL

VIVADO

HLS

PETA linux

ZYNQ

HDL

**HDL Overview**

You can use a Hardware Description Language (HDL), such as VHDL or Verilog, for your top-level or lower-level design files. HDL files describe the behavior and structure of system and circuit designs. Using HDLs allows you to do the following:

* Use a synthesis engine to translate your design to gates

Synthesis decreases design time by eliminating the need to define every gate. In addition, the synthesis tool can apply automation, such as machine encoding styles or automatic I/O insertion during optimization, resulting in greater efficiency.

* Run functional simulation early in the design cycle

You can verify your design functionality early in the flow by simulating the HDL description. Testing your design at the Register Transfer Level (RTL) or gate level before the design is implemented allows you to make changes early in the design process.

* Retarget your code to different architectures

You can use the same HDL design for new architectures with a minimum of recoding. This works especially well if you inferred, rather than instantiated, components.

Vivado High-Level Synthesis

Accelerates IP Creation by Enabling C, C++ and System C Specifications

پردازش سیگنال: انجام عملیات ریاضی روی سیگنال

Fixed-point

* Unsigned
* Signed

Floating-point

پیاده سازی عملگرهای جبری

* جمع کننده، ضرب کننده، تقسیم کننده، جذر، توابع مثلثاتی
* Sign & magnitude
* One’s complement
* Two’s complement

With two’s complement if we know the addition of many numbers does not have overflow, we do not need to be worried about internal overflow that may occur.

پیاده سازی در VHDL

* Unsigned
* Signed

use ieee.numeric\_std.all;

signal X: signed (7 downto 0) :=( others=>’0’);

نمایش اعداد اعشاری

* Floating-Point
* Fixed-Point

مفهوم یک عدد باینری، به تفسیری که از آن میشود بستگی دارد.

Quantization

Fixed-Point system does not have any standard.

Use s.m.n approach to show fixed-point numbers.

S: sign

M: number of integer digits

N: number of fraction digits

Fixed-point addition:

The number of fraction digits (N) of two numbers must be equal by appending zero in the right.

The number of integer digits (M) of two numbers must be equal by sign-extending the number with smaller M.

Fixed-point multiplication

(m1, n1)\*(m2, n2) => (m1+m2, n1+n2)

(s, m1, n1)\*(s, m2, n2) => (s, m1+m2+1, n1+n2)

Fixed-Point Quantization

Quantization noise (error)

Lesson2

(s, m, n) => (1, m+n+1, n) in Matlab

(s, 2, 8) => (1, 11, 8)

1. Create floating-point model in Simulink application
2. Create Fixed-Point model based on floating-point model
3. Quantize variables and compare floating-point output with fixed-point output and calculate quantization error
4. Save Simulink input and output data in two files
5. Write Vhdl code from Simulink Fixed-Point model
6. Write a test bench and feed input data from step 4 in vhdl model and save vhdl output data
7. Compare output data from Simulink model and FPGA implementation.

Quantization process

1. Quantizing out of system input signals
2. Quantizing in system input signals
3. Quantizing operands

DDS IP for creating sin waves in FPGA

Lesson 3

Filters & FFT

FIR and IIR Filters

Scaling

* Real scaling
* Virtual scaling

FIR Filters

* Simple to implement
* Linear phase
* Higher magnitude

IIR Filters

* Complicate to implement
* Non-linear phase
* Lower magnitude

FIR implementation

* Direct Form FIR
* Transposed Structure (Better for implementation)

Delay Line

* Delay line can be implemented by registers
* Delay line can be implemented by memory block

Lesson 4

Matlab FDA tool