Session #1
Introduction to Verilog

Advantages of working in Digital Domain

- Continuous signals has infinite range of values. Digital signals has finite range of values which makes them predictable and simpler to process and manipulate.
- The device used in Digital Circuits generally operate in one of the two states known as ON and OFF resulting in a very simple operation
- It is easier to store digital values for future use.
- Transmitting digital signals is easier, even if they face deformations during transmission, errors can predicted.
- Transforming digital signals from one representation to another is very easy to achieve.
- Easy controllability and stability.
- Reliability, accuracy and speed
- Lower cost per function.
- Versatile circuits can be designed and realized on a single chip. For e.g., the microprocessor can do unlimited computational functions.

Arriving to Digital Domain

Conversion of Analog signals to discrete samples and giving them a value is the primary requirement to process the signals in digital domain.

Example: Digital Sound Processing

Sound is produced by the vibration of a media like air or water. Audio refers to the sound within the range of human hearing. Naturally, a sound signal is analog, which is continuous in both time and amplitude.

To store and process sound information in a computer or to transmit it through a computer network, we must first convert the analog signal to digital form using an analog-to-digital converter (ADC)

The conversion involves two steps: (1) sampling, and (2) quantization.

ADC

- Analog to Digital Converter

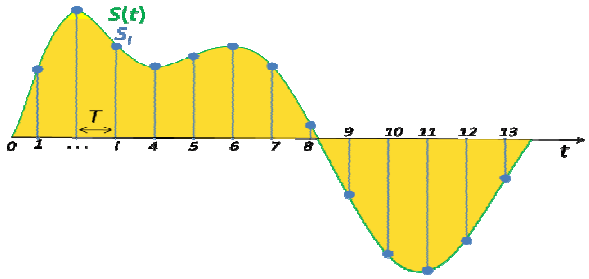
  A device which converts continuous signals to discrete signal numbers.

  The digital output value of ADC is proportional to the analog input voltage or current

  Digital output may use different coding schemes like: Binary, Grey, etc.

Sampling

- Sampling is the reduction of a continuous signal to a discrete signal.



Nyquist Criterion

Bandlimited analog signal that has been sampled can be perfectly reconstructed from an infinite sequence of samples if the sampling rate exceeds $2B$ samples per second, where B is the highest frequency in the original signal.

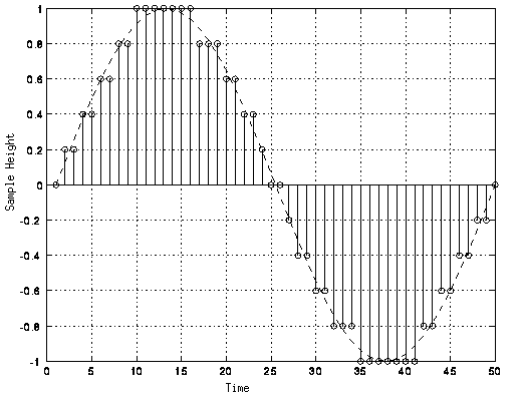
An ideal sampler which extracts samples from the continuous signals should obey the criterion.

The samples obtained from the sampler obeying the criterion, can be used to reconstruct the continuous signal

Quantization

- **Quantization** is the process of approximating ("mapping") a continuous range of values by a relatively small ("finite") set of discrete symbols or integer values.
- In other words, Quantization is the process of limiting the value of a sample of a continuous function to one of a predetermined number of allowed values, which can then be represented by a finite number of bits in the digital world.
- After quantization we can do encoding using binary techniques

Quantization



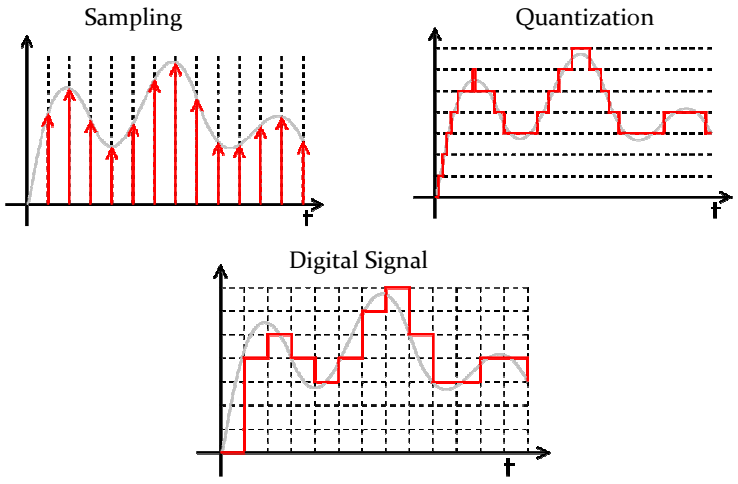
Encoding

Each quantized level can be given a specific symbol or a value.

There are different encoding schemes e.g. Binary, Grey, etc.

Processing is done on these values by manipulating them according to the outputs we require.

Analog to Digital Conversion



ADC Resolution

$$Q = \frac{E_{FSR}}{2^M - 1} = \frac{E_{FSR}}{N}$$

Where:

Q is resolution in volts per step (volts per output codes less one),
EFSR is the full scale voltage range = *VRefHi − VRefLow*,
M is the ADC's resolution in bits.
N is the number of intervals

ADC Resolution

Resolution

The resolution of the converter indicates the number of discrete values it can produce over the range of analog values.

The values are usually stored electronically in binary form, so the resolution is usually expressed in bits.

In consequence, the number of discrete values available, or "levels", is usually a power of two.

For example, an ADC with a resolution of 8 bits can encode an analog input to one in 256 different levels, since $2^8 = 256$.

DAC

Digital to Analog Converter

DACs are used to convert finite precision time series data to a continually varying physical signal.

By the Nyquist–Shannon sampling theorem, sampled data can be reconstructed perfectly provided that it meets certain requirements.

However, even with an ideal reconstruction filter, digital sampling introduces quantization error that makes perfect reconstruction practically impossible.

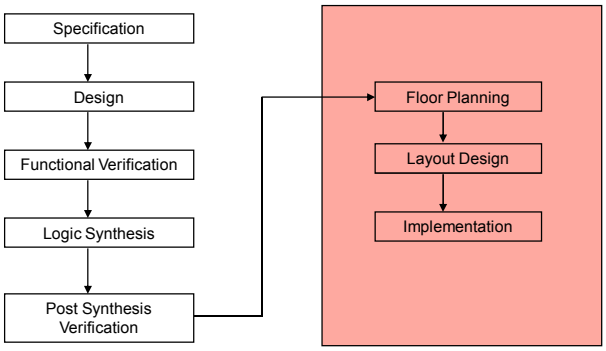
Increasing the number of bits used in each sample (resolution) can be used to reduce the quantization error

Digital Design Summary

- Truth Tables
- K-Map simplification
- Implementing a circuit
- Testing a circuit
- Limitations of K-Map
- Other methods

Design Flow

The question is: how to realize a digital circuit?



What is a HDL?

- HDL is an acronym for Hardware Description Language.
- As the name suggests, it is a software language which supports constructs which can describe a digital hardware.
- Examples:
  - VHDL
  - Verilog
  - System Verilog
  - System C

Need for HDL

Traditionally, digital design was a manual process of designing and capturing digital circuits using schematic entry tools.

Lot of disadvantages and constantly being replaced by different methods.

System designers are always competing to build highly cost effective products as fast as possible in a highly competitive environment.

Top down design methodologies using HDL and synthesis along with simulation and verification

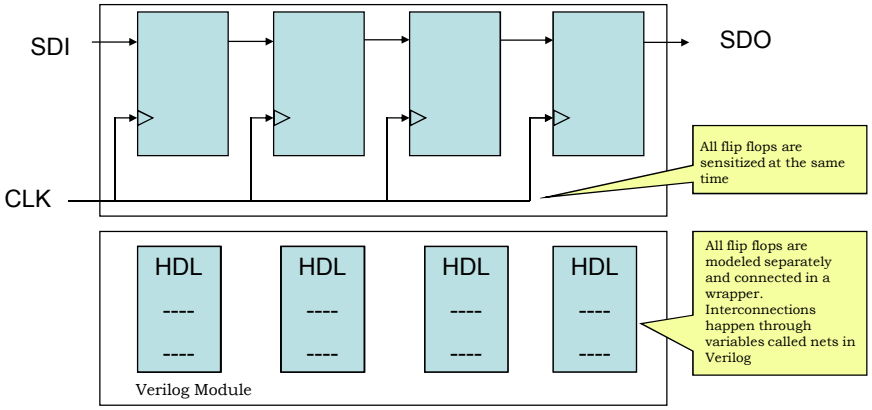
Since time to market is a concern -

"ASAP" becomes the key word in todays world!

Key advantages of using HDLs

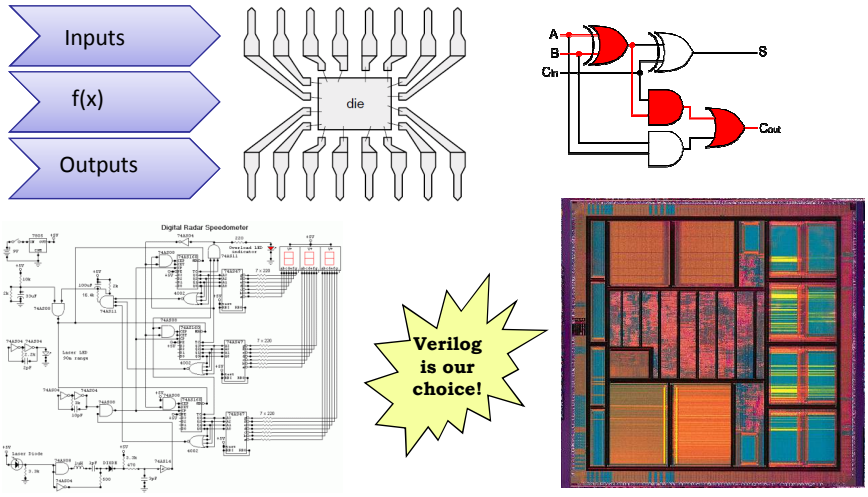
- Increased productivity yields shorter development cycles with more product features and reduced time to market.
- Reduced Non-Recurring Engineering costs.
- Design re-use enabled.
- Increased flexibility in design changes.
- Faster exploration of alternative architectures.
- Faster exploration of alternative technology libraries.
- Enables use of synthesis to rapidly sweep the design space of area and timing, and to automatically generate testable circuits
- Better and easier design auditing and verification

Difference between HDLs and other software programming languages



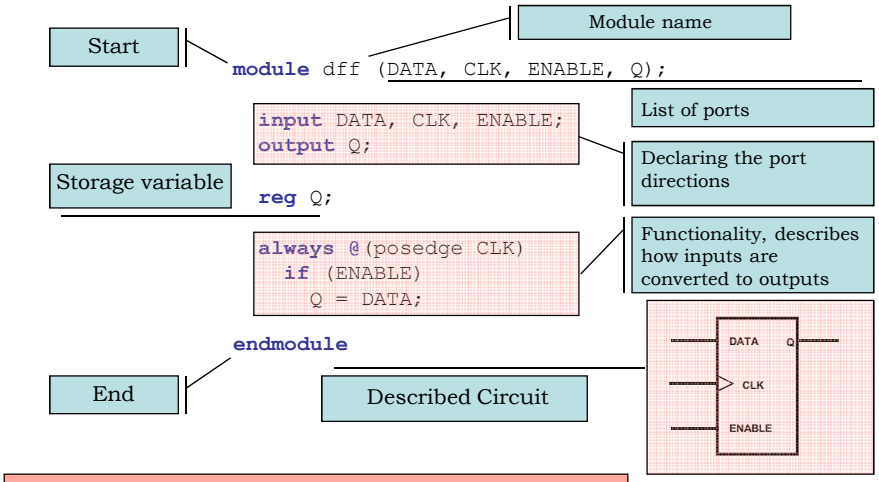
- Generally, hardware operates in a concurrent fashion. Hence, any language to be qualified as a HDL, it should support concurrent constructs which can be used to reflect the hardware.
- Software programming languages are sequential in nature; meaning the code executes one step at a time. They are targeted for processors which execute instructions one by one.

Basic Layout of a Digital Circuit



All the details of the circuit should be captured by the HDL description of that circuit.

Description of Digital Circuits



HDL description should exactly mirror the digital circuit in concern.

A D-Flip Flop

History of Verilog

- ***Beginning***
  - Verilog was invented by Phil Moorby and Prabhu Goel during 1983-84 at Automated Integrated Design Systems (renamed to Gateway Design Automation in 1985) as a hardware modeling language.
  - GDA was brought by Cadence Design Systems in 1990.
  - Cadence now has full proprietary rights to Gateway's Verilog and Verilog – XL simulators.
- ***Verilog – 95***
  - Cadence decided to make the language available for open standardization, they transferred Verilog into public domain under Open Verilog International (now known as Accellera) organization.
  - Verilog was later submitted to IEEE and became IEEE standard 1364-1995, commonly referred to as Verilog-95

History continued….

- ***Verilog 2001***
  - Extentions to Verilog-95 were submitted back to IEEE to cover the deficiencies that users had found in the original Verilog standard. These extensions became IEEE standard 1364-2001 known as Verilog-2001.
  - Verilog-2001 is the dominant flavor of Verilog supported by majority of commercial EDA software packages.
- ***Verilog-2005***
  - Verilog-2005(IEEE standard 1364-2005) consists of minor corrections, spec clarifications and a few new language features like uwire keyword.
- ***System Verilog***
  - System Verilog is a superset of Verilog-2005 with many new features and capabilities to aid design verification and design modelling.

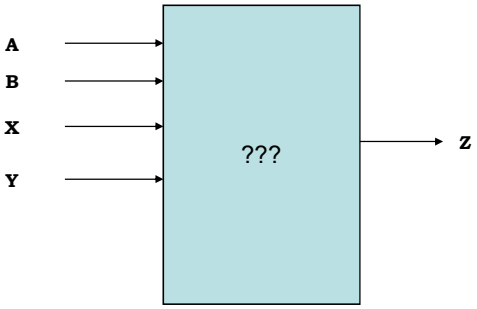
Language Reference Manual

- The Verilog Language Reference Manual is a document which describes various features of the Verilog HDL and the usage information.
- Refer to the Verilog LRM provided to you.

Welcome to the world of Verilog!

Exercise 1

- Design the following circuit



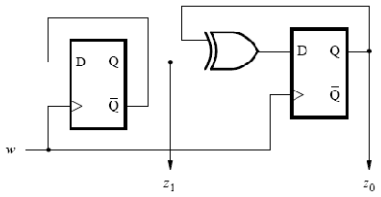
Specifications:

A	B	Z
0	0	X&Y
0	1	X|Y
1	0	!X
1	1	!Y

20 Minutes

Exercise 2

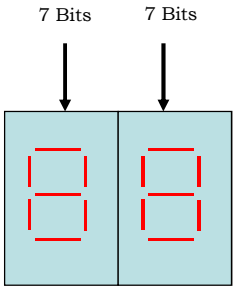
- Analyze the circuit and develop the specifications along with waveforms:



20 Minutes

Exercise 3

- How do you design the logic for a seven segment display given below to count from 00 to 99?



30 Minutes