Course Code	Course Title	L	T	P	C
1152EC101	VLSI SIGNAL PROCESSING	3	0	0	3

a) Course Category

Program elective

b) Preamble

This Course provides the basic and design knowledge about VLSI Signal Processing which involves DSP Technology ,Algorithmic and Numeric strength reduction and pipelining and parallel processing.

c) Prerequisite

Digital Electronics, Digital Signal Processing and VLSI design

d) Related Courses

Low Power VLSI

e) Course educational objectives

i) To understand the various VLSI architectures for digital signal processing.
ii) To introduce techniques for altering the existing DSP structures to suit VLSI implementations.
iii) To explains how to design high-speed, low-area, and low-power VLSI systems for a broad range of DSP applications.

f) Course Outcomes

Upon the successful completion of the course, students will be able to:

CO Nos.	Course Outcomes	Knowledge Level (Based on Revised Bloom's Taxonomy)
CO1	Illustrate design architectures for DSP algorithms.	K3
CO2	Apply retiming and algorithmic strength reduction technique optimize design parameters	K3
CO3	Apply high level algorithm transformation to optimize design parameters.	K3
CO4	Apply various Bit-level arithmetic architecture to design the multipliers	K3
CO5	Apply numeric strength reduction to reduce area and power in digital	K3

	filters	

e)	Correlation of COs with POs													
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	H	M	L	L	-	-	L	-	-	-	-	-	-	-
CO2	M	H	M	M	L	-	-	-	-	-	-	-	-	-
CO3	M	H	M	M	M	-	-	-	L	-	-	-	L	-
CO4	H	M	M	M	H	-	-	-	L	L	-	M	M	-
CO5	M	H	M	-H	M	L	-	L	L	L	L	M	-	-

f) Course Content

UNIT I INTRODUCTION TO DSP SYSTEMS **9**

Introduction to DSP systems – Typical DSP algorithms, Data flow and Dependence graphs – critical path, Loop bound, iteration bound, Longest path matrix algorithm. Introduction to pipelining and parallel processing.

UNIT II RETIMING, ALGORITHMIC STRENGTH REDUCTION **9**

Retiming – definitions and properties, Unfolding – an algorithm for unfolding, properties of unfolding, sample period reduction -Algorithmic strength reduction in filters and transforms – 2-parallel FIR filter, 2-parallel fast FIR filter

UNIT III FAST CONVOLUTION **9**

Fast convolution – Cook-Toom algorithm, modified Cook-Toom algorithm – Winograd Algorithm-cyclic convolution – design of fast convolution algorithm by inspection

UNIT IV PIPELINIG AND PARALLEL RECURSIVE AND ADDAPTIVE FILTERS **9**

Introduction – pipelined interleaving in digital filters –pipelining in 1^{st} order IIR digital filters and higher order IIR digital filters –parallel processing for IIR filter low power IIR filter design using pipelineand parallel processing – pipelined Adaptive digital filters

UNIT V NUMERICAL STRENGTH REDUCTION **9**

Numerical strength reduction – subexpression elimination, multiple constant multiplication, subexpression

sharing in digital filters – Additive and multiplicative number splitting

Total 45 Hrs

g) Learning Resources

Text Books

1. Keshab K.Parhi, "VLSI Digital Signal Processing Systems, Design and Implementation", John Wiley, Indian Reprint, 2007.
2. S.Y.Kuang, H.J. White house, T. Kailath, "VLSI and Modern Signal Processing", Prentice Hall, 1995

Reference Books

1. U. Meyer –Baese, "Digital Signal Processing with Field Programmable Arrays", Springer, Second Edition, Indian Reprint, 2007

Online Resource

1. https://books.google.co.in/books?isbn=8126510986
2. **http://nptel.iitg.ernet.in/**

Practical Aspects.

After implementing design and analyzing implementation results, the following methods to improve design performance prior to programming and configuring your device:

Optimize timing performance, using any of the following methods:

1. Use synthesis techniques, such as proper coding, as described in Using Synthesis Techniques to Improve Timing Performance.
2. Use timing constraints, as described in Optimizing Design Constraints.
3. Floorplan your design, as described in Floorplanning with PlanAhead™ Software.

Experiment with implementation options, also known as process properties, using any of the following methods:

1. Modify individual process properties, as described in Design Performance Techniques for FPGAs.
2. Use predefined Design Goals and Strategies provided to modify sets of process properties, as described in Using Design Goals and Strategies.
3. Use SmartXplorer to run multiple implementation flows using different sets of process properties, as described in Using SmartXplorer.
4. Use FPGA Editor to make modifications to your FPGA design.
5. You can use FPGA Editor to check that your design was implemented as expected, and then use it to fine-tune your design, as described in Implementation Strategies using FPGA Editor.

Use techniques to reduce area utilization, power consumption, memory use, and runtime and to preserve design results as follows:

1. Use coding techniques to reduce area utilization and power consumption, as described in Using RTL Coding and Synthesis Techniques to Reduce Area Utilization and Power Consumption.
2. Use constraints and process properties to reduce memory use and runtime, as described in Memory Use and Runtime Strategies for FPGAs.
3. Use SmartGuide™ technology to use results from a previous implementation to guide the next

implementation. This helps to reduce runtime, preserve logic, and meet timing, as described in Using SmartGuide Technology.

4. Use Partitions to reuse or preserve certain modules in your design during implementation, as described in the Partitions Overview.