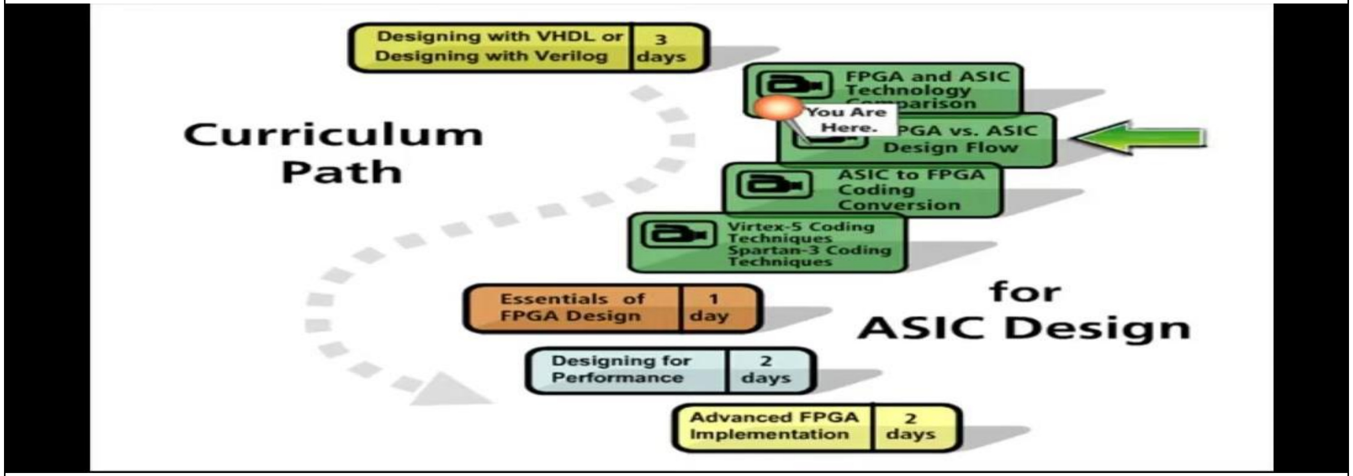
DAILY ASSESSMENT FORMAT

Date:	**01-06-2020**	**Name:**	**Sahana S R**
Course:	**Digital design using HDL**	**USN:**	**4al17ec083**
Topic:	**•Industry Application of FPGA** **•FPGA Business Fundamental** **•FPGA vs FPGA design flow** **•FPGA basics A look under the hood**	**Semester & Section:**	**6th sem** **B sec**
Github Repository:	**sahanasr-course**		

FORENOON SESSION DETAILS
Image of session  Designing with VHDL or Designing with Verilog 3 days Curriculum Path FPGA and ASIC Technology Comparison You Are Here. FPGA vs. ASIC Design Flow ASIC to FPGA Coding Conversion Virtex-5 Coding Techniques Spartan-3 Coding Techniques Essentials of FPGA Design 1 day for ASIC Design Designing for Performance 2 days Advanced FPGA Implementation 2 days

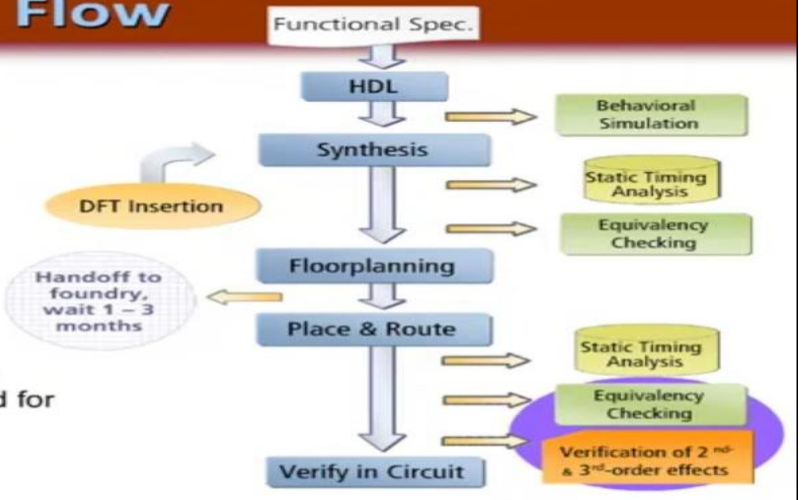
Design Flow

- ASIC and FPGA design and implementation methodologies differ somewhat
  - Xilinx FPGAs provide for reduced design time and later bug fixes
    - No design for test logic is required
    - Deep sub-micron verification is done
    - No waiting for prototypes
- Coding style
  - For high-performance designs, FPGAs may require some pipelining
  - When retargeting code from an ASIC to an FPGA, the code usually requires optimization (instantiation)

ASIC Design Flow

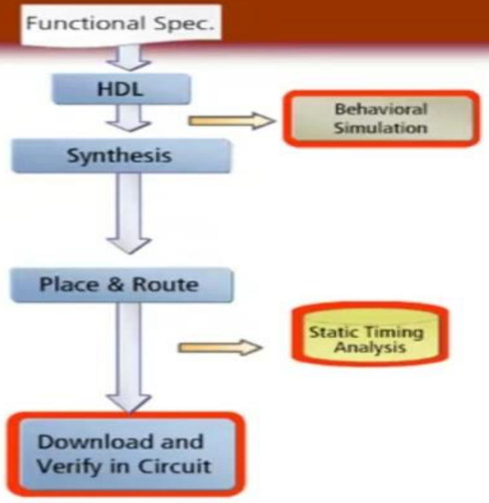
- ASIC tools are generally driven by scripts
- Post-synthesis static timing analysis and equivalency checking are musts for sign-off to foundry
- Verification of deep sub-micron effects is required for ASICs

  Internal, deep sub-micron effects are already verified for Xilinx FPGAs

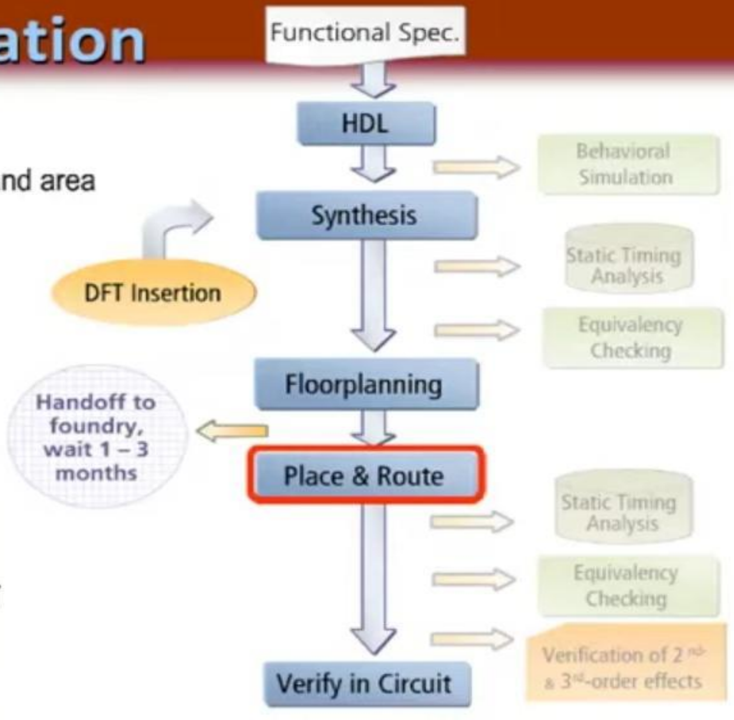


FPGA Design Flow

- FPGA tools are generally GUI-driven, pushbutton flows
  - FPGA tools also have scripting capabilities
- After the design passes behavioral simulation and static timing analysis, verification is completed most efficiently by verifying in circuit
  - Fast turnaround times
  - Static timing analysis is used to verify timing of the design
  - Timing simulation is supported
  - This is a simplified/typical design flow



ASIC Implementation



Create HDL

Optimized for ASIC technology and area

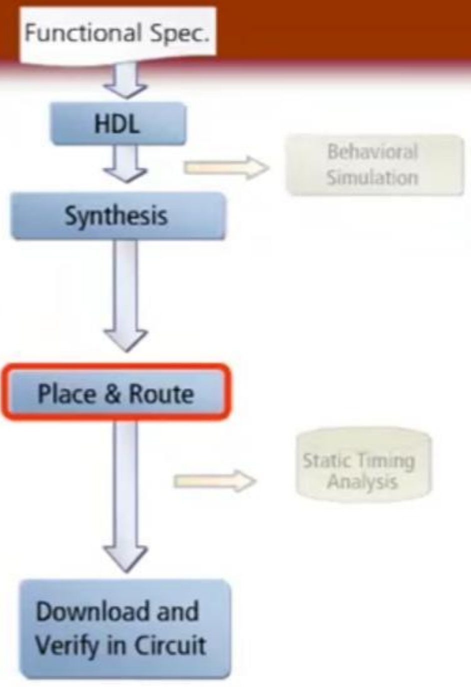
Synthesis

- Primarily driven by scripts
- Synopsys design compile
- Design for test logic insertion (BIST, Scan, and JTAG)

Place & route

Foundry tools, Cadence, AVANT

FPGA Implementation



Create HDL

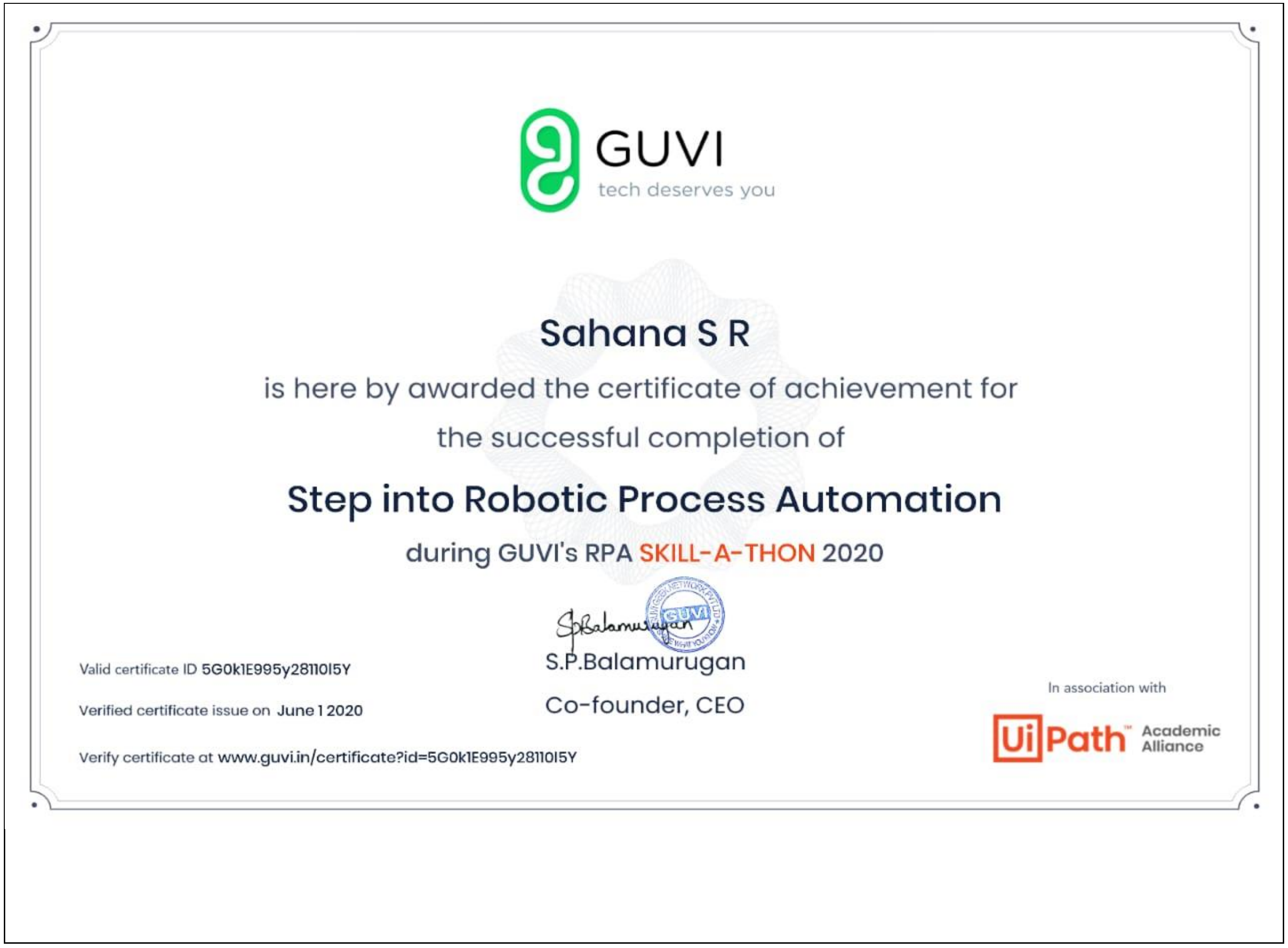
- Optimized for Xilinx FPGAs and performance

Synthesis

- Synopsys, Mentor, XST
- Pushbutton flow with scripting capabilities

Place & route

- Completed by the user
- Xilinx implementation tools – ISE® software
- Pushbutton flow, scripting capabilities



GUVI

tech deserves you

Sahana S R

is here by awarded the certificate of achievement for

the successful completion of

Step into Robotic Process Automation

during GUVI's RPA SKILL-A-THON 2020

GUVI GEEK NETWORK PVT LTD GUVI

S.P.Balamurugan

Co-founder, CEO

Valid certificate ID 5G0k1E995y28110I5Y

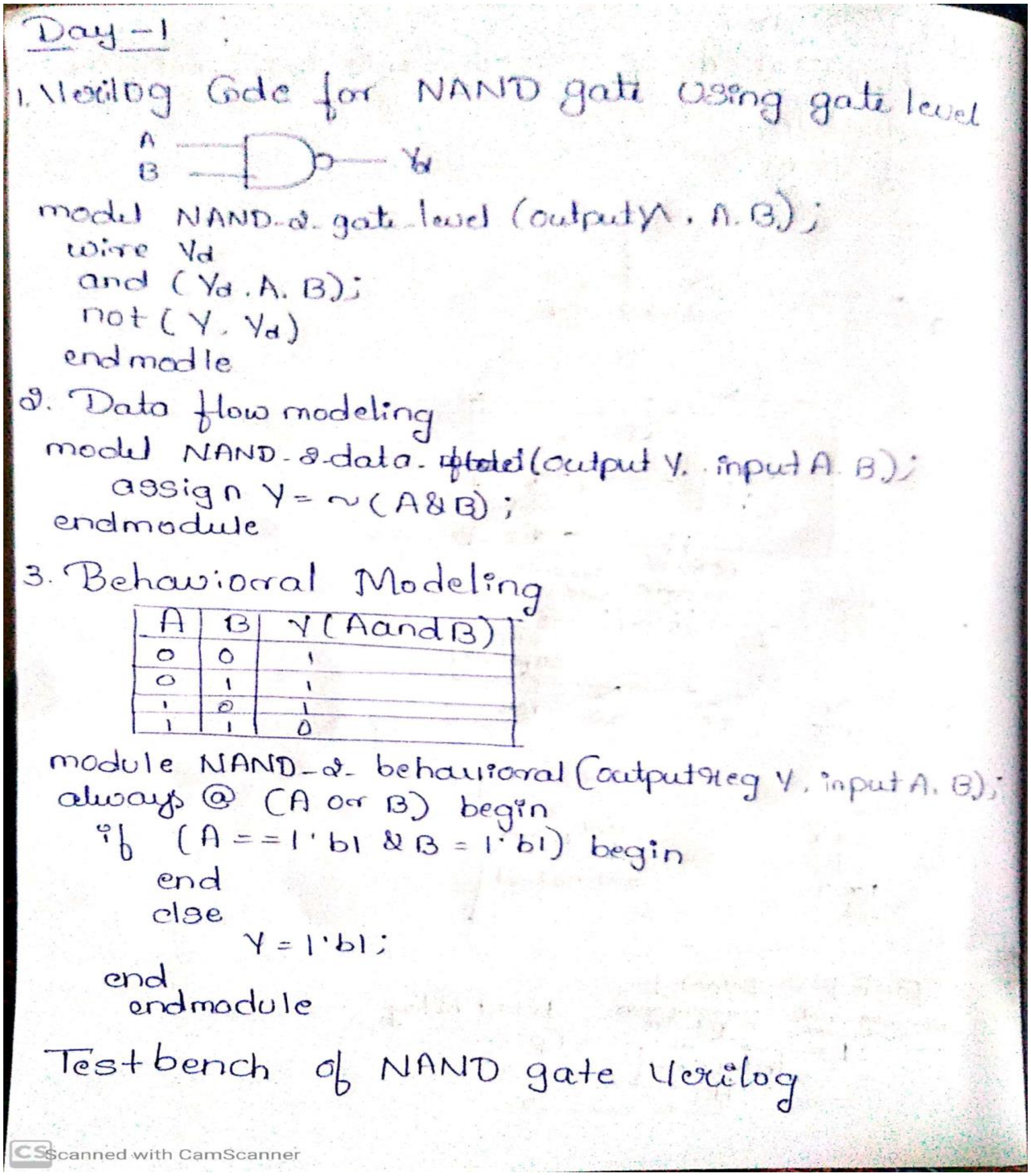
Verified certificate issue on June 1 2020

Verify certificate at www.guvi.in/certificate?id=5G0k1E995y28110I5Y

In association with

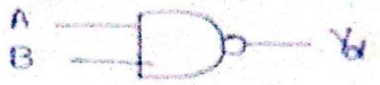
UiPath™ Academic Alliance

Report – Report can be typed or hand written for up to two pages.



Day - 1

1. Verilog Code for NAND gate using gate level



```
model NAND-2-gate-level (outputY, A, B);
  wire Yd
  and ( Yd . A. B);
  not ( Y . Yd )
 endmodle
```

2. Data flow modeling

```
model NAND-2-data-~~flow~~ (output Y. input A. B);
   assign Y = ~(A&B);
 endmodule
```

3. Behavioral Modeling

A	B	Y (A and B)
0	0	1
0	1	1
1	0	1
1	1	0

```
module NAND-2- behavioral (outputreg Y, input A, B);
always @ (A or B) begin
  if (A == 1'b1 & B = 1'b1) begin
      end
      else
           Y = 1'b1;
  end
   endmodule
```

Test bench of NAND gate Verilog

```
AND_2_behavioral Indtance0(Y,A,B)
initial begin
  A=0; B=0;
#1 A=0; B=1;
#1 A=1; B=0;
#1 A=1; B=1;
end
initial begin
 $monitor ("%t |A = %d |B = %d | Y=%d", $time,
              A,B,Y);
 $dumpfile ("dump.vcd");
 $dumpvars();
end
endmodule
```

Data flow modeling is a higher level of abstration. This helps as gate-level modeling becomes very Complicated for large circuits.