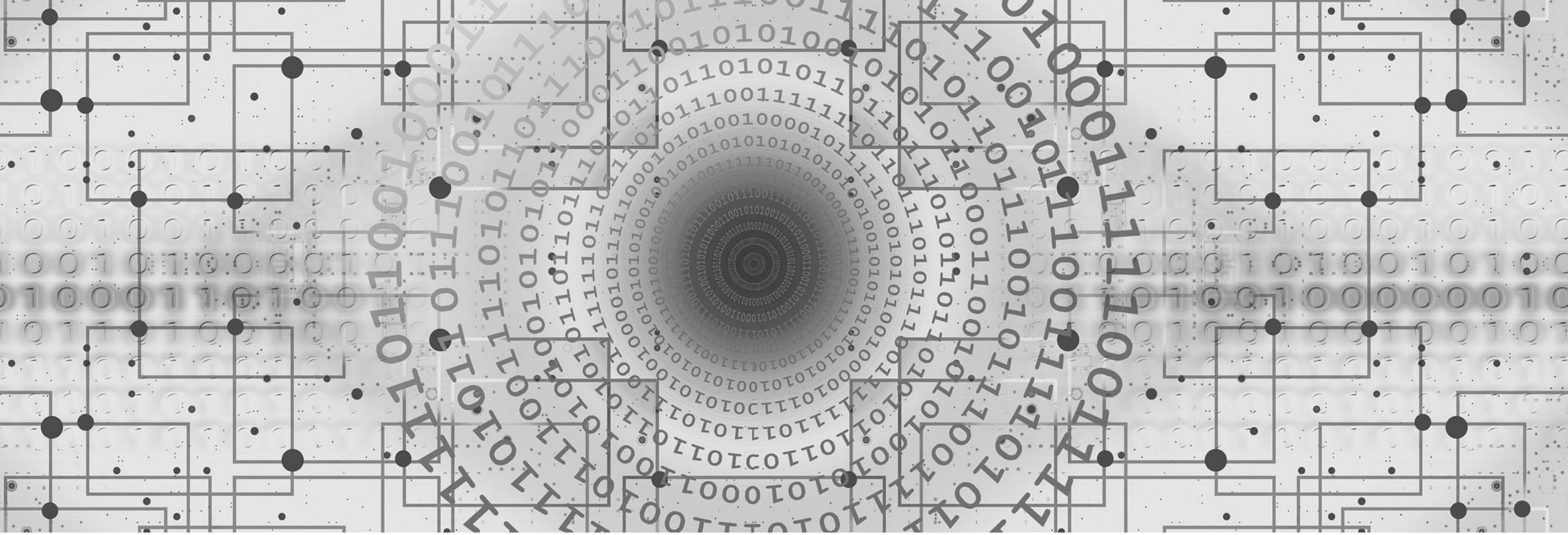


Homework-5a: Gray Code to Binary Code Converter

ECE-111 Advanced Digital Design Project

Vishal Karna

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering

Outline

- **What is Gray Code**

- **First understand Binary to Gray Code Conversion for reference purpose**
  - Overview of how to perform Binary code to Gray code conversion
  - Review of SystemVerilog code for Binary to Gray code conversion

- **Homework Requirements for Gray to Binary Code Conversion**
  - Overview of how to perform Gray Code to Binary Code Conversion

What is Gray Code

- **Gray code** named after Frank Gray, is an ordering of the binary numeral system such that two successive numbers differ in only one bit
  - Gray code was originally designed to prevent spurious output from electromechanical switches
  - Gray codes are widely used to facilitate error correction in digital communication applications

Decimal Value	Binary Value				Gray Code Value			
	B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

From previous to next value, only 1-bit changes at a time

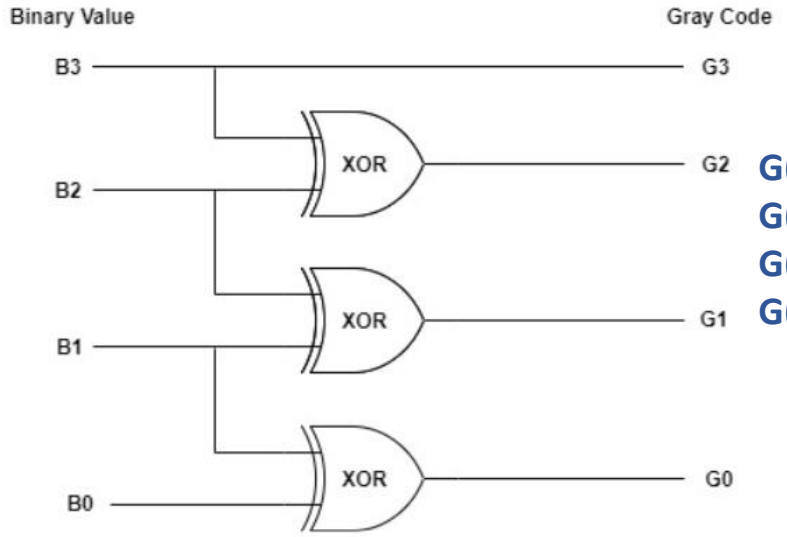
Binary Code to Gray Code Conversion Method

❑ Binary to Gray conversion :

- MSB of the gray code is always equal to the MSB of the given binary code.
- Other bits of the output gray code can be obtained by XORing binary code bit at that index and previous index.

Binary Value				Gray Code Value			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

From previous to next value, only 1-bit changes at a time



G(3) = B(3)
G(2) = B(2) xor B(3)
G(1) = B(1) xor B(2)
G(0) = B(0) xor B(1)

Binary to Gray Conversion using Function

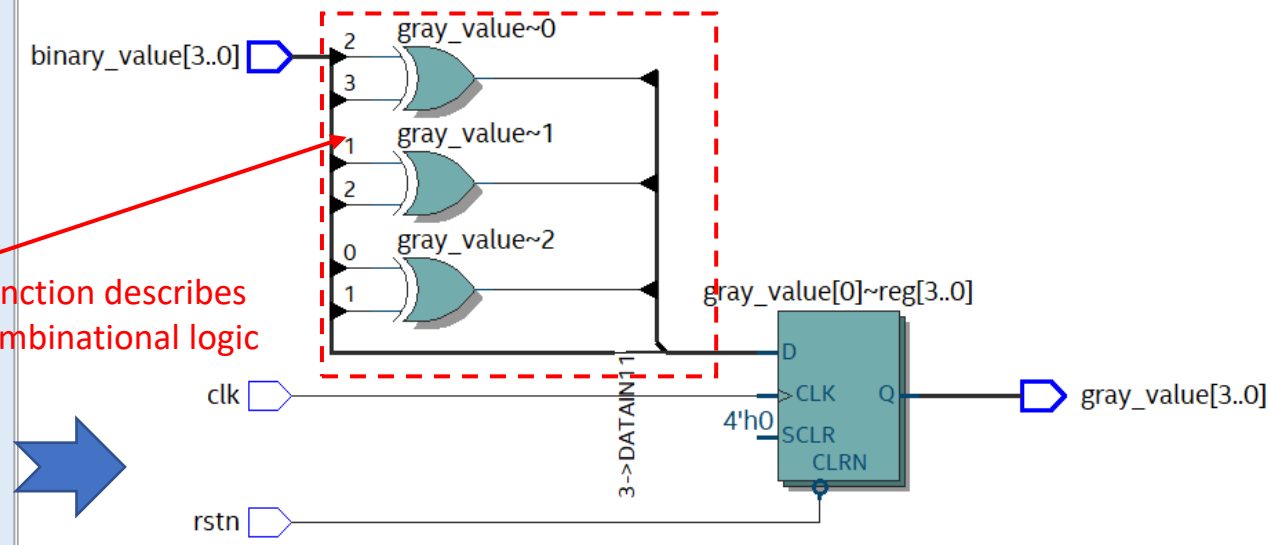
```
module binary_to_gray_conv  #(parameter N = 4)(
 input logic clk, rstn,
 input logic[N-1:0] binary_value,
 output logic[N-1:0] gray_value);

// Function to convert binary to gray value
function automatic [N-1:0] binary_to_gray(logic [N-1:0] value);
  begin
    binary_to_gray[N-1] = value[N-1];
    for(int i=N-1; i>0; i = i - 1)
      binary_to_gray[i-1] = value[i] ^ value[i - 1];
   end
endfunction

// Store binary2gray output in a register
always_ff@(posedge clk or negedge rstn) begin
  if (!rstn) begin
    gray_value <= 0;
  end
  else begin
    gray_value <= binary_to_gray(binary_value);
  end
end
endmodule: binary_to_gray_conv
```

default return type is logic if not specified

return value assigned to a automatic declared variable with same name as a function name

call to a function with input value passed

Function describes combinational logic



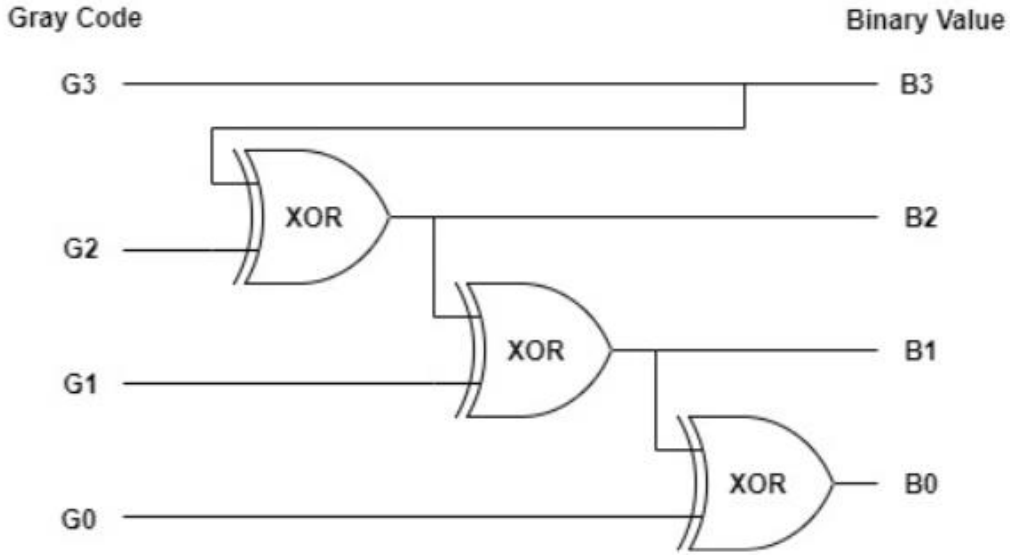
Gray Code to Binary Code Conversion Method

❑ Gray Code to Binary Conversion :

- MSB of the binary code is always equal to the MSB of the given binary number.
- Other bits of the output binary value can be obtained by XOR'ing gray code bit at that index and binary bit at next index.
- Use below mentioned 4-bit Gray code to Binary conversion equation, circuit and truth table as a reference for SystemVerilog code development.

Note : when simulating gray to binary code, ensure binary values for given gray code in waveform is reflected as per the truth table above.

Gray Code Value				Binary Value			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1



B(3) = G(3)
B(2) = G(2) xor B(3)
B(1) = G(1) xor B(2)
B(0) = G(0) xor B(1)

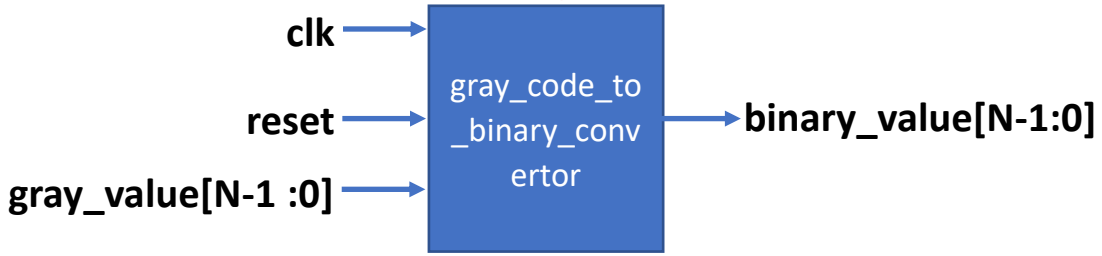
Homework Assignment-5a : Gray to Binary Code Converter

❑ Develop SystemVerilog RTL model for N-bit Gray to Binary Code Converter

- Use function to implement gray to binary code conversion and add a flipflop at the output the converter
- Synthesize design and run simulation using testbench provided
- Review synthesis results (resource usage and RTL netlist/schematic)
- Testbench is provided for 4-bit gray to binary value conversion.
- Review input and output signals in simulation waveform.
- Assume below mentioned primary port names and SystemVerilog RTL module name **gray_code_to_binary_convertor module.**

❑ Primary Ports for gray_code_to_binary_convertor module

- **Input clk :** posedge clock
- **Input rstn :** reset should be asynchronous negedge reset
- **Input gray_value :** N-bit input gray code signal
- **Output binary_value :** N-bit output converted binary value signal
  - **Note : Add a flipflop at the output of gray to binary code convertor logic**



Homework Assignment-5a : Gray to Binary Code Converter

- **Report should include :**
  - SystemVerilog design code
  - Synthesis resource usage and schematic generated from RTL netlist viewer
  - Simulation snapshot and explain simulation result to confirm RTL model developed works as a gray to binary code converter
  - Post-Mapping schematic is optional to submit.
  - Explanation of FPGA resource usage in report is not required.

- **Lab5 folder includes :**
  - Template code for design : gray_code_to_binary_convertor.sv
  - Full testbench code for gray to binary code converter is provided. For learning purpose, student can change the stimulus in initial block in testbench file.