Ty Madsen

ECEN 220

30 October 2013

Lab 7 Register File

1 Bit Register

Verilog:

module Register1Bit(

input Din,

input Write,

input Clk,

input Clr,

output Dout

);

wire d,q;

fd\_c flipflop(q,Clk,d,Clr);

MUX21 mux(d,Write,q,Din);

assign Dout=q;

endmodule

TCL:

wave add / -radix hex

isim force add Clr 1

isim force add Clk 0 -time 0 -value 1 -time 10ns -repeat 20ns

isim force add Din 1

isim force add Write 0

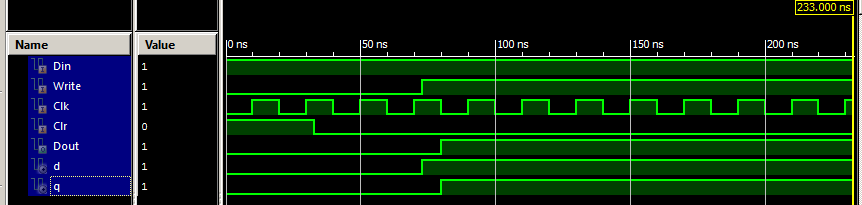
run 33ns

isim force add Clr 0

run 40ns

isim force add Write 1

run 160ns



4 Bit Register:

Verilog:

module Register4Bit(

input [3:0] Din,

input Write,

input Clk,

input Clr,

output [3:0] Dout

);

Register1Bit r0(Din[0], Write, Clk, Clr, Dout[0]);

Register1Bit r1(Din[1], Write, Clk, Clr, Dout[1]);

Register1Bit r2(Din[2], Write, Clk, Clr, Dout[2]);

Register1Bit r3(Din[3], Write, Clk, Clr, Dout[3]);

endmodule

TCL:

wave add / -radix hex

isim force add Clr 1

isim force add Clk 0 -time 0 -value 1 -time 10ns -repeat 20ns

isim force add Din 1111

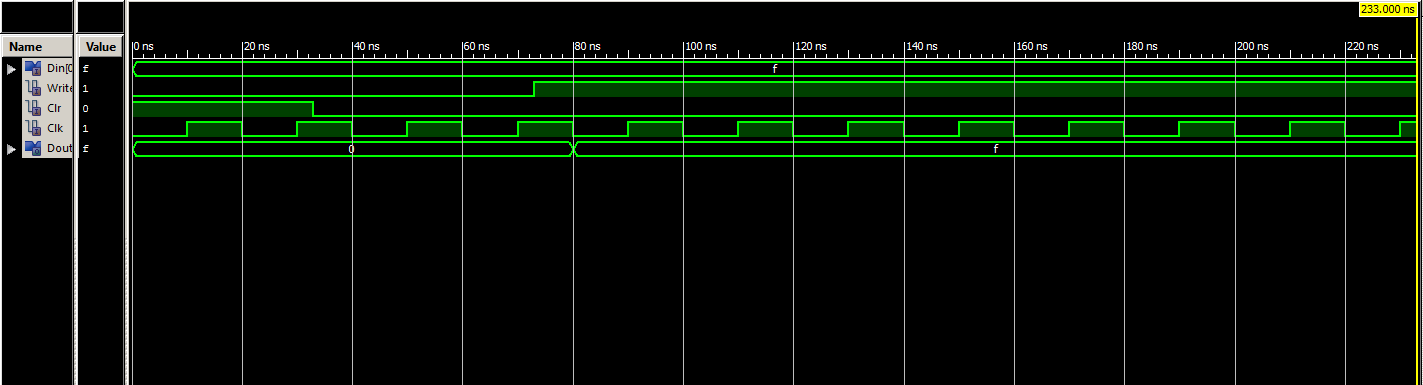
isim force add Write 0

run 33ns

isim force add Clr 0

run 40ns

isim force add Write 1

run 160ns

2 to 4 Decoder:

Verilog:

module Decoder2\_4(

input Addr0,

input Addr1,

output Sel0,

output Sel1,

output Sel2,

output Sel3

);

wire nAddr0, nAddr1;

not(nAddr0,Addr0);

not(nAddr1, Addr1);

and(Sel3,Addr0,Addr1);

and(Sel2,nAddr0,Addr1);

and(Sel1,Addr0,nAddr1);

and(Sel0,nAddr0,nAddr1);

endmodule

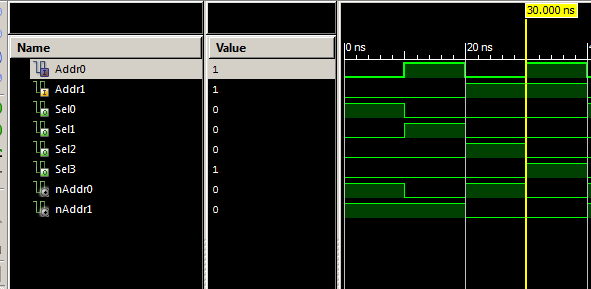
TCL:

wave add / -radix hex

isim force add Addr0 0 -time 0 -value 1 -time 10ns -repeat 20ns

isim force add Addr1 0 -time 0 -value 1 -time 20ns -repeat 40ns

run 160ns



4 Bit 4 Word:

Verilog:

module Register4Word(

input [0:3] Din,

input Write,

input Clk,

input Clr,

input Addr0,

input Addr1,

output [3:0] Reg\_0,

output [3:0] Reg\_1,

output [3:0] Reg\_2,

output [3:0] Reg\_3

);

wire wReg0, wReg1, wReg2, wReg3, Sel0, Sel1, Sel2, Sel3;

Decoder2\_4 decoder(Addr0, Addr1, Sel0, Sel1, Sel2, Sel3);

and(wReg3, Write, Sel3);

and(wReg2, Write, Sel2);

and(wReg1, Write, Sel1);

and(wReg0, Write, Sel0);

Register4Bit reg3(Din,wReg3,Clk,Clr,Reg\_3);

Register4Bit reg2(Din,wReg2,Clk,Clr,Reg\_2);

Register4Bit reg1(Din,wReg1,Clk,Clr,Reg\_1);

Register4Bit reg0(Din,wReg0,Clk,Clr,Reg\_0);

endmodule

TCL:

wave add / -radix hex

isim force add Clr 1

isim force add Din 1111

isim force add Write 0

isim force add Addr0 0

isim force add Addr1 0

run 33ns

isim force add Clr 0

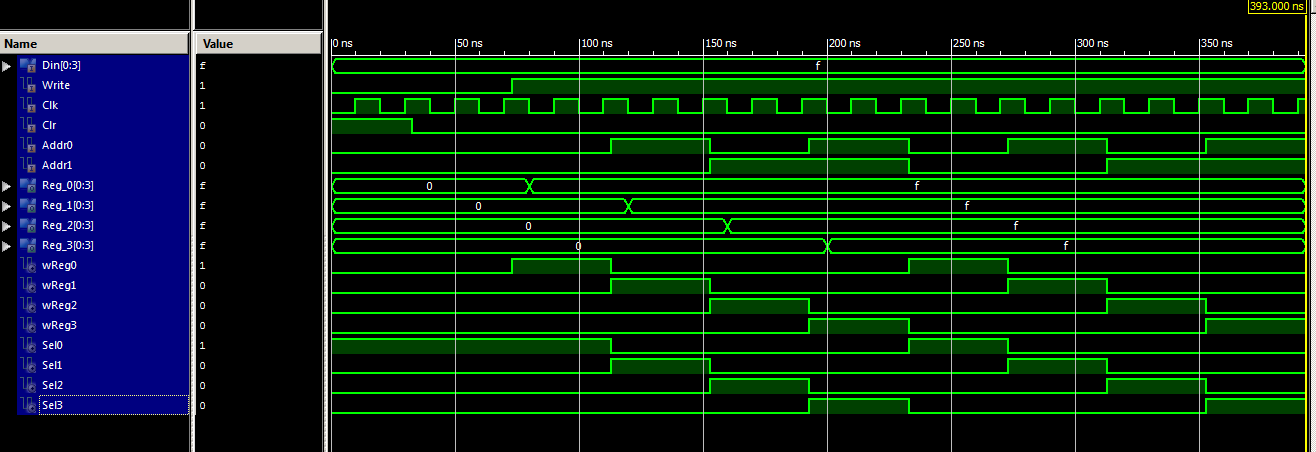
run 40ns

isim force add Write 1

isim force add Addr0 0 -time 0 -value 1 -time 40ns -repeat 80ns

isim force add Addr1 0 -time 0 -value 1 -time 80ns -repeat 160ns

run 320ns



4 to 1 and 16 to 4 Bit MUX:

Verilog:

module mux4\_1(result, sel, Din);

input[1:0] sel;

input[3:0] Din;

output result;

wire r0,r1;

mux2\_1 m0(r0, sel[0], Din[0], Din[1]);

mux2\_1 m1(r1, sel[0], Din[2], Din[3]);

mux2\_1 m2(result, sel[1], r0, r1);

endmodule

module mux16\_4(result, sel, Din3, Din2, Din1, Din0);

input[0:1] sel; // 2 select line

input[3:0] Din3, Din2, Din1, Din0; // 4 4-bit values come in as inputs

output[3:0] result; // 4-bit output

// Select which bits of which input gets passed to which of the 4-bit muxes

mux4\_1 M3(result[3], sel, {Din3[3], Din2[3], Din1[3], Din0[3]});

mux4\_1 M2(result[2], sel, {Din3[2], Din2[2], Din1[2], Din0[2]});

mux4\_1 M1(result[1], sel, {Din3[1], Din2[1], Din1[1], Din0[1]});

mux4\_1 M0(result[0], sel, {Din3[0], Din2[0], Din1[0], Din0[0]});

endmodule

TCL:

wave add / -radix hex

isim force add Din3 1010

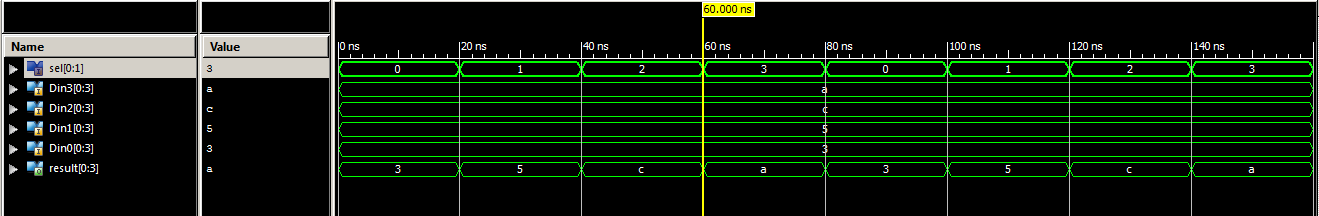
isim force add Din2 1100

isim force add Din1 0101

isim force add Din0 0011

isim force add sel 00 -time 0 -value 01 -time 20ns -value 10 -time 40ns -value 11 -time 60ns -repeat 80ns

run 160ns



Complete Register:

Verilog:

module Register4Word4Bit(

input [3:0] Din,

input Write,

input Clk,

input Clr,

input Addr0,

input Addr1,

input [1:0] ReadAddr,

output [3:0] RegFile\_out,

output [3:0] Din\_led

);

wire [3:0] Reg\_0, Reg\_1, Reg\_2, Reg\_3;

Register4Word fourWordReg(Din, Write, Clk, Clr, Addr0, Addr1, Reg\_0, Reg\_1, Reg\_2, Reg\_3);

mux16\_4 mux(RegFile\_out, ReadAddr, Reg\_3, Reg\_2, Reg\_1, Reg\_0);

buf(Din\_led[3],Din[3]);

buf(Din\_led[2],Din[2]);

buf(Din\_led[1],Din[1]);

buf(Din\_led[0],Din[0]);

endmodule

UCF:

## clock pin for Nexys 2 Board

NET Clk LOC = "B8"; # Bank = 0, Pin name = IP\_L13P\_0/GCLK8, Type = GCLK, Sch name = GCLK0

## Leds

NET RegFile\_out<0> LOC = "J14"; # Bank = 1, Pin name = IO\_L14N\_1/A3/RHCLK7, Type = RHCLK/DUAL, Sch name = JD10/LD0

NET RegFile\_out<1> LOC = "J15"; # Bank = 1, Pin name = IO\_L14P\_1/A4/RHCLK6, Type = RHCLK/DUAL, Sch name = JD9/LD1

NET RegFile\_out<2> LOC = "K15"; # Bank = 1, Pin name = IO\_L12P\_1/A8/RHCLK2, Type = RHCLK/DUAL, Sch name = JD8/LD2

NET RegFile\_out<3> LOC = "K14"; # Bank = 1, Pin name = IO\_L12N\_1/A7/RHCLK3/TRDY1, Type = RHCLK/DUAL, Sch name = JD7/LD3

NET Din\_led<0> LOC = "E17"; # Bank = 1, Pin name = IO, Type = I/O, Sch name = LD4

NET Din\_led<1> LOC = "P15"; # Bank = 1, Pin name = IO, Type = I/O, Sch name = LD5

NET Din\_led<2> LOC = "F4"; # Bank = 3, Pin name = IO, Type = I/O, Sch name = LD6

NET Din\_led<3> LOC = "R4"; # Bank = 3, Pin name = IO/VREF\_3, Type = VREF, Sch name = LD7

## Switches

NET ReadAddr<0> LOC = "G18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW0

NET ReadAddr<1> LOC = "H18"; # Bank = 1, Pin name = IP/VREF\_1, Type = VREF, Sch name = SW1

NET Addr0 LOC = "K18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW2

NET Addr1 LOC = "K17"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW3

NET Din<0> LOC = "L14"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW4

NET Din<1> LOC = "L13"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW5

NET Din<2> LOC = "N17"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW6

NET Din<3> LOC = "R17"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW7

## Buttons

NET Clr LOC = "B18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = BTN0

#NET "btn<1>" LOC = "D18"; # Bank = 1, Pin name = IP/VREF\_1, Type = VREF, Sch name = BTN1

#NET "btn<2>" LOC = "E18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = BTN2

NET Write LOC = "H13"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = BTN3