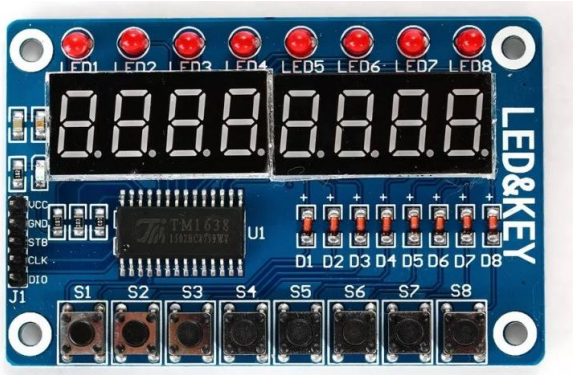
Chương 4
GIAO TIẾP MODULE MỞ RỘNG LED 7 ĐOẠN: TM1638

1. Giới thiệu

TM1638 là mô đun mở rộng bao gồm 8 led đơn, 8 led 7 đoạn và 8 nút nhấn. Vi mạch TM1638 điều khiển led đơn, quét led 7 đoạn và phím nhấn, sử dụng giao tiếp nối tiếp với thiết bên ngoài. Các led đơn và led 7 đoạn được điều khiển bằng phương pháp quét, do đó có thể thay đổi thông số quét để thay đổi độ sáng của LED.

Trong phần thực hành này, sinh viên học cách giao tiếp truyền dữ liệu 1 chiều từ FPGA sang mô đun mở rộng TM1638 để điều khiển LED đơn, LED 7 đoạn. Sinh viên xây dựng các chương trình đếm, đồng hồ số hiển thị trên LED 7 đoạn nhằm mở rộng các ứng dụng cho board Spartan 3E. Sinh viên cũng học cách chuyển đổi từ số hex sang số bcd, giải mã led 7 đoạn sử dụng hàm trong Verilog để hiển thị giá trị số trên led 7 đoạn

Module TM1638 có kết nối chân khá tương thích với Spartan 3E, sử dụng một bus 5 dây giao tiếp.



Hình : Mô đun mở rộng ngoại vị (Led đơn, led 7 đoạn và nút nhấn) TM1638

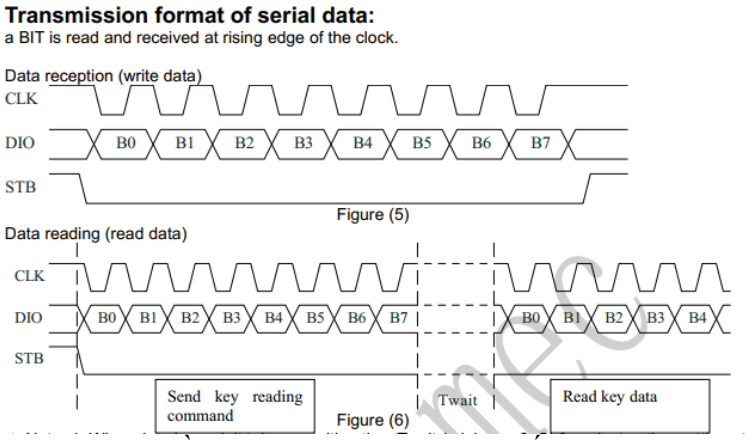
Các tín hiệu giao tiếp bao gồm VCC (3.3V), GND, STB, CLK, DIO
Các tín hiệu được kết nối đến Pinheader mở rộng của Spartan 3 (J4) như sau **(sinh viên lưu ý đấu nối dây chính xác để không làm hỏng mạch điện trên board). Khi kết nối đúng, led nguồn trên mô đun TM1638 sẽ sáng.**

Bảng 1. Sơ đồ kết nối Spartan 3 và TM1638

TM1638	Spartan 3E (J4)
VCC	VCC
GND	GND
STB	E8
CLK	F8
DIO	C7

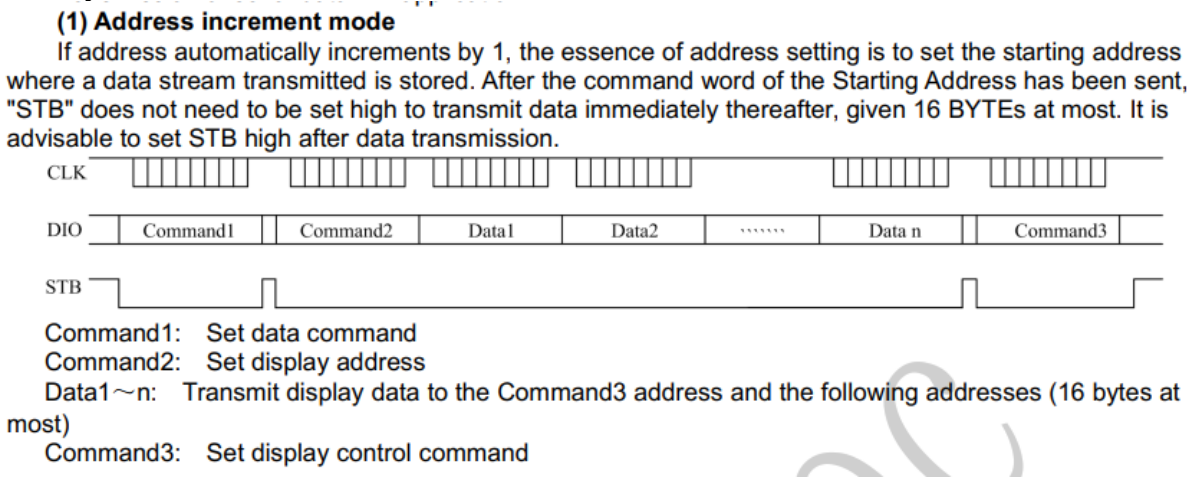
TM1638 có 8 led đơn, giá trị điều khiển mỗi led được lấy từ 1 ô nhớ 8 bit tại địa chỉ 1, 3, 5, 7, 9, 11, 13, 15. 8 led 7 đoạn cathode chung được lấy dữ liệu từ 8 ô nhớ ở địa chỉ thứ 0, 2, 4, 6, 8,10 ,12 ,14 (sinh viên đọc kỹ tài liệu kỹ thuật TM1638 để nắm rõ) . Như vậy có tất cả 16 ô nhớ để điều khiển 16 led. Để giao tiếp với board TM1638, sinh viên cần đọc chi tiết tài liệu kỹ thuật

của board, trong đó trình bày chi tiết các mã lệnh giao tiếp cũng như địa chỉ vùng nhớ. Giao tiếp với TM1638 được tóm tắt như sau



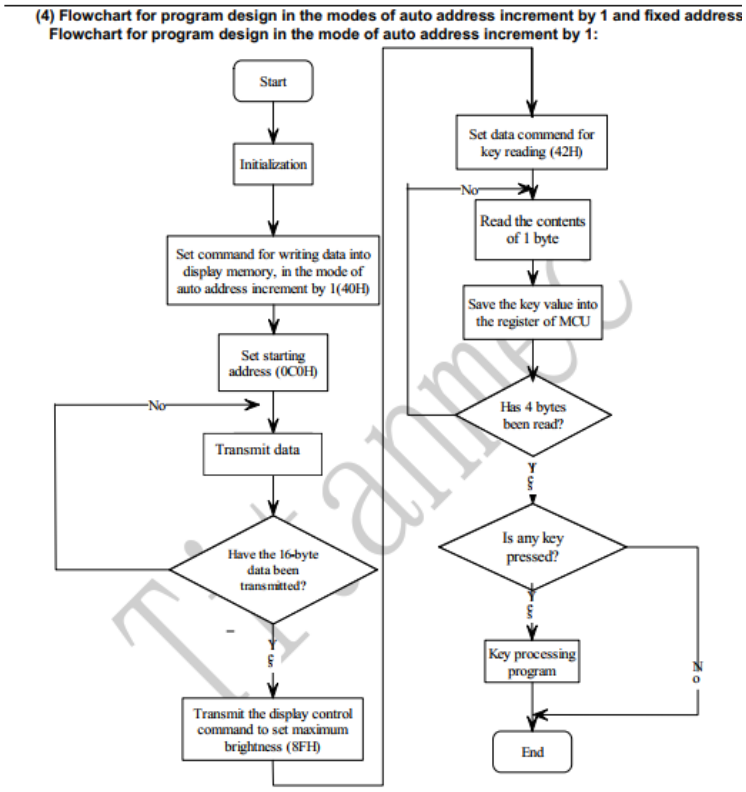
Hình 2. Sơ đồ dạng sóng tín hiệu giao tiếp với TM1638

TM1638 đọc dữ liệu tại cạnh lên xung CLK sau khi STB được kéo xuống 0. Byte đầu tiên là phải là mã lệnh, các byte tiếp theo có thể là mã lệnh hoặc dữ liệu. TM1638 hỗ trợ truyền dữ liệu theo từng địa chỉ ô nhớ hoặc truyền liên tục. Trong bài thực hành này, sinh viên có thể xây dựng chương trình chuyển dữ liệu liên tục 16 byte cho TM1638. Giao thức truyền liên tục dữ liệu được trình bày bên dưới



Hình 3. Giao thức truyền dữ liệu liên tục

Truyền liên tục dữ liệu và đọc giá trị nút nhấn được thực hiện theo lưu đồ sau:

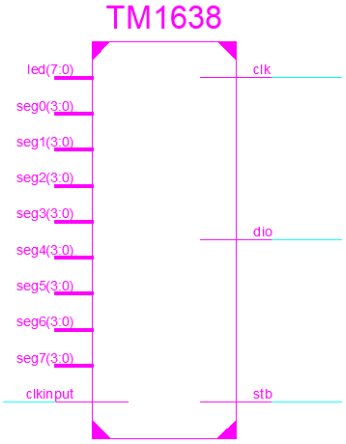


2. Thực hành

Thiết kế Mô đun giao tiếp TM1638 để điều khiển 8 led 7 đoạn và 8 led đơn. Mô đun được thiết kế với ngõ vào 1 byte cho 8 led đơn, các giá trị cho mỗi led 7 đoạn được sử dụng 4 bit, ngõ vào xung clk có tần số khoảng vài trăm khz. Ngõ ra stb, clk, dio giao tiếp TM1638

2.1 Xây dựng mô đun điều khiển TM1638

Sơ đồ khối module giao tiếp



Hình 4. Sơ đồ mô đun giao tiếp TM1638

Code Verilog cho module giao tiếp TM618

```
/* TM1638 driver
Author: Mr. Son
input clock: tested for 200KHz  ( 50H - free counter -> bit 5
*/

module TM1638(
   input wire [7:0] led , // 8 leds
         input wire [3:0] seg7,seg6,seg5,seg4,seg3,seg2,seg1,seg0 ,//4 bit data for cathode
commond LED

         input clkinput,
   output reg clk,
   output reg stb,
   output reg dio
   );

/* Hex-Digit to seven segment LED decoder
 Author: Mr. Son
*/
function [7:0] sseg;
   input  [3:0] hex;
   begin
      case (hex)
         4'h0: sseg[7:0] = 8'b0111111;
         4'h1: sseg[7:0] = 8'b0000110;
         4'h2: sseg[7:0] = 8'b1011011;
         4'h3: sseg[7:0] = 8'b1001111;
         4'h4: sseg[7:0] = 8'b1100110;
         4'h5: sseg[7:0] = 8'b1101101;
         4'h6: sseg[7:0] = 8'b1111101;
         4'h7: sseg[7:0] = 8'b0000111;
         4'h8: sseg[7:0] = 8'b1111111;
         4'h9: sseg[7:0] = 8'b1101111;
         4'hA: sseg[7:0] = 8'b1110111;
         4'hB: sseg[7:0] = 8'b1111100;
         4'hC: sseg[7:0] = 8'b1011000;
         4'hD: sseg[7:0] = 8'b1011110;
         4'hE: sseg[7:0] = 8'b1111001;
         default : sseg[7:0] = 8'b0000000; // 4'hF
      endcase
   end
endfunction

         integer cs = 0;
         integer i ;
```

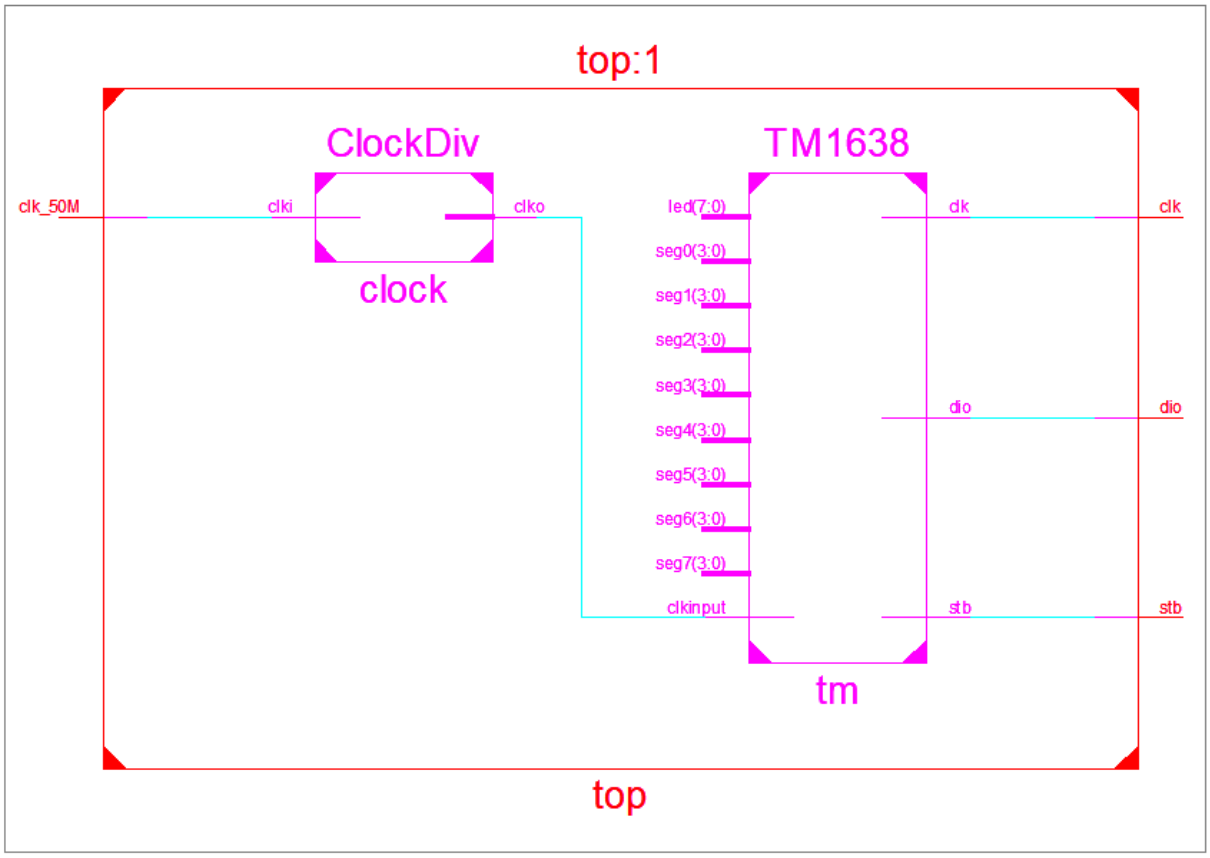
```
reg [7:0] command1 =8'h40, command2 =8'hC0,command3 =8'h8F;
wire [127:0] leddata; // 1,3,5,7,9,11,13,15: single led; 0,2,4,6,8,10,12,14: seg LED (common
cathode)
reg [127:0] leddatahold;

        assign leddata[0*8+7:0*8+0] = sseg(seg0);
        assign leddata[2*8+7:2*8+0] = sseg(seg1);
        assign leddata[4*8+7:4*8+0] = sseg(seg2);
        assign leddata[6*8+7:6*8+0] = sseg(seg3);
        assign leddata[8*8+7:8*8+0] = sseg(seg4);
        assign leddata[10*8+7:10*8+0] = sseg(seg5);
        assign leddata[12*8+7:12*8+0] = sseg(seg6);
        assign leddata[14*8+7:14*8+0] = sseg(seg7);


        assign leddata[1*8+7:1*8+0] = led[0] ;
        assign leddata[3*8+7:3*8+0] = led[1] ;
        assign leddata[5*8+7:5*8+0] = led[2] ;
        assign leddata[7*8+7:7*8+0] = led[3] ;
        assign leddata[9*8+7:9*8+0] = led[4] ;
        assign leddata[11*8+7:11*8+0] = led[5] ;
        assign leddata[13*8+7:13*8+0] = led[6] ;
        assign leddata[15*8+7:15*8+0] = led[7] ;


initial
begin

        clk = 1 ;
        stb = 1 ;
  dio = 0 ;

end


always @(posedge clkinput)
begin
if (cs==0)
        begin
        stb = 0; // initial tm1638
  command1 =8'h40; command2 =8'hC0;command3 =8'h8F;
        leddatahold=leddata ;
        end

else if ((cs >=1)&&(cs<=16))
        begin
```

```
        dio = command1[0];
        clk = ~clk ;
        if (clk) command1=command1>>1 ;
        end
else if (cs==17)
        stb = 1; // stop tm1638

else if (cs==18)
        stb = 0; // ready to send the second command
// send second command
else if ((cs >=19)&&(cs<=34))
        begin
        dio = command2[0];
        clk = ~clk ;
        if (clk) command2=command2>>1 ;
        end

else if ((cs >=35)&&(cs<=290))
        begin
        dio = leddatahold[0];
        clk = ~clk ;
        if (clk) leddatahold=leddatahold>>1 ;
        end

else if (cs==291)
        stb = 1; // stop tm1638 for end of data

else if (cs==292)
        stb = 0; // ready to send the third command
// send last command
else if ((cs >=293)&&(cs<=308))
        begin
        dio = command3[0];
        clk = ~clk ;
        if (clk) command3=command3>>1 ;
        end

else if (cs==309)
        stb = 1; // End
else if (cs==310)
        cs = -1 ;  //repeat

// update cs
  cs=cs+1;
end
endmodule
```

2.2.Xây dựng mô đun test mô đun TM1638

Viết chương trình điều khiển LED đơn và hiển thị số từ 0 đến 7 trên led 7 đoạn để test mô đun TM1638

Sinh viên phân tích schematic và code mô tả hệ thống sau, tiến hành thực nghiệm để kiểm tra kết quả



Verilog code

Top module
module top(input clk_50M, output wire clk, output wire stb, output wire dio); wire clko; ClockDiv clock (clk_50M, clko) ; wire [4:0] seg [7:0];

```
TM1638 tm (8'b01010101,0,1,2,3,4,5,6,7,
          clko,
    clk,
    stb,
    dio
    );

endmodule
```

Clock divide

```
module ClockDiv(
input wire clki,
output wire  clko
   );
wire [26:0] r_next ;
reg [26:0] r_reg;
initial r_reg =0 ;
always @(posedge clki)
r_reg = r_next;
assign r_next = r_reg + 1 ;
assign clko=r_reg[5];
endmodule
```

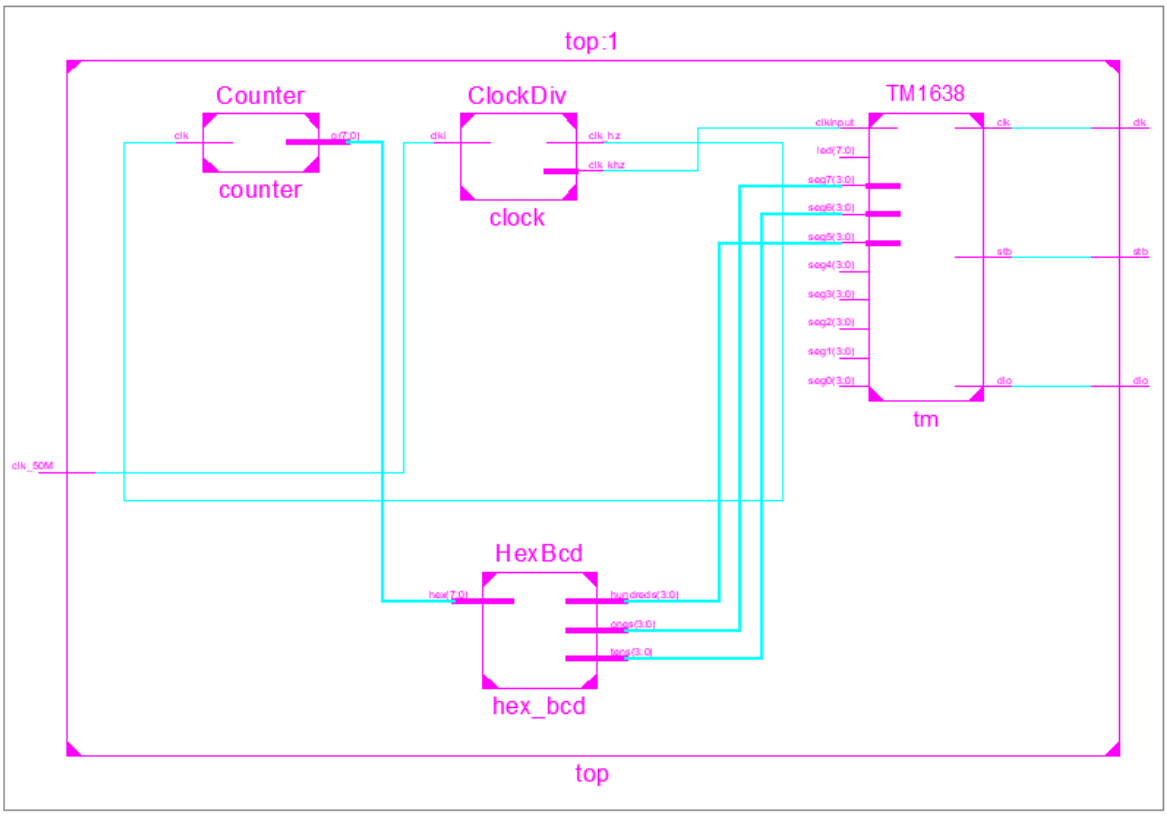
pcf file

```
# PlanAhead Generated physical constraints
NET "clk_50M" LOC = C9;
NET "stb" LOC = E8;
NET "clk" LOC = F8;
NET "dio" LOC = C7;

# PlanAhead Generated IO constraints

NET "clk_50M" IOSTANDARD = LVCMOS33;
NET "clk" IOSTANDARD = LVCMOS33;
NET "stb" IOSTANDARD = LVCMOS33;
NET "dio" IOSTANDARD = LVCMOS33;
```

3. Phát triển ứng dụng với TM1638

3.1.Viết chương trình đếm hiển thị trên led 7 đoan

Sinh viên phân tích schematic và code mô tả hệ thống sau, tiến hành thực nghiệm để kiểm tra kết quả




```
module top(input clk_50M,
           output wire clk,
    output wire stb,
    output wire dio
    );
wire clk_khz,clk_hz;
wire [7:0] q;
wire [3:0] ones,tens,hundreds ;
ClockDiv clock (clk_50M, clk_khz,clk_hz) ;
Counter counter (clk_hz,q ) ;
HexBcd hex_bcd (q,ones,tens,hundreds);
wire [4:0] seg [7:0];
TM1638 tm (8'b1,ones,tens,hundreds,15,15,15,15,15,
           clk_khz,
    clk,
    stb,
    dio
    );
endmodule
```

```
module ClockDiv(
input wire clki,
output wire  clk_khz,clk_hz
```

```
    );
wire [26:0] r_next ;
reg [26:0] r_reg;
initial begin r_reg =0 ;end
always @(posedge clki)
r_reg = r_next;
assign r_next =(r_reg==50000000)?0: r_reg + 1 ;
assign clk_khz=r_reg[5];  /*781.250 Khz*/
assign clk_hz=(r_reg<=50000000/2)?0:1;  /*781.250 Khz*/

endmodule
```

```
module Counter(
input wire clk,
output wire [7:0] q
   );
wire [7:0] r_next ;
reg [7:0] r_reg;
initial r_reg =0 ;
always @(posedge clk)
r_reg = r_next;
assign r_next = r_reg + 1 ;
assign q=r_reg;

endmodule
```

```
module add3(in,out);
input [3:0] in;
output [3:0] out;
reg [3:0] out;
always @ (in)
case (in)
4'b0000: out <= 4'b0000;
4'b0001: out <= 4'b0001;
4'b0010: out <= 4'b0010;
4'b0011: out <= 4'b0011;
4'b0100: out <= 4'b0100;
4'b0101: out <= 4'b1000;
4'b0110: out <= 4'b1001;
4'b0111: out <= 4'b1010;
4'b1000: out <= 4'b1011;
4'b1001: out <= 4'b1100;
default: out <= 4'b0000;
endcase
endmodule


module HexBcd(
```

```
        input      [7:0] hex,
   output  wire [3:0]  ones,
   output  wire [3:0]  tens,
   output  wire [3:0]  hundreds

   );

wire [3:0] c1,c2,c3,c4,c5,c6,c7;
wire [3:0] d1,d2,d3,d4,d5,d6,d7;
assign d1 = {1'b0,hex[7:5]};
assign d2 = {c1[2:0],hex[4]};
assign d3 = {c2[2:0],hex[3]};
assign d4 = {c3[2:0],hex[2]};
assign d5 = {c4[2:0],hex[1]};
assign d6 = {1'b0,c1[3],c2[3],c3[3]};
assign d7 = {c6[2:0],c4[3]};
add3 m1(d1,c1);
add3 m2(d2,c2);
add3 m3(d3,c3);
add3 m4(d4,c4);
add3 m5(d5,c5);
add3 m6(d6,c6);
add3 m7(d7,c7);
assign ones = {c5[2:0],hex[0]};
assign tens = {c7[2:0],c5[3]};
assign hundreds = {c6[3],c7[3]};

endmodule
```

```
# PlanAhead Generated physical constraints
NET "clk_50M" LOC = C9;
NET "stb" LOC = E8;
NET "clk" LOC = F8;
NET "dio" LOC = C7;

# PlanAhead Generated IO constraints

NET "clk_50M" IOSTANDARD = LVCMOS33;
NET "clk" IOSTANDARD = LVCMOS33;
NET "stb" IOSTANDARD = LVCMOS33;
NET "dio" IOSTANDARD = LVCMOS33;
```

3.2.Các ứng dụng phát triển trên mô đun TM1638

3.2.1. Viết chương trình điều khiển 8 led sáng dần trên TM1638

3.2.2. Viết chương trình điều khiển 8 led sáng dần từ trong ra ngoài trên TM1638

3.2.3. Viết chương trình đếm giờ phút giây hiển thị trên TM1638

4. Các ứng dụng nâng cao

4.1. Viết chương trình đếm từ 000 đến 999 hiển thị trên TM1638

4.2. Viết chương trình đếm từ 0 đến 999, tốc độ đếm điều điều chỉnh nhanh/chậm bằng công tắc xoay trên board FPGA Spartan 3.

4.3. Viết chương trình điều khiển 8 led trên board TM1638. led thực hiện 4 chương trình: chop tắt, 1 led sáng chạy từ trái sang phải và ngược lại, led sáng dần từ trái sang phải, led sáng dần từ phải sang trái. Chương trình được lựa chọn bằng 1 nút nhấn, tốc độ led được điều chỉnh bằng 1 nút nhấn.