計算機科学実験及び演習 3 HW 導入課題

橘大佑[1]

2021 年 4 月 14 日

[1]京都大学工学部情報学科 3 回計算機科学コース

工学部情報学科計算機科学コース
1029-31-6811
橘 大佑

導入課題

計算機科学実験及演習 3 HW
2021 年 4 月 14 日

課題 1

1. 回路の仕様の決定
   下の図 1 の 7SEG LED の名前の配置をもとに点灯させる場所の名前をしるす。

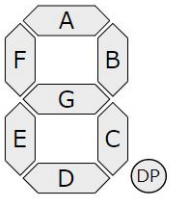

図 1: 7SEG LED の名前の配置

- ”0”の表現方法 · · · A,B,C,D,E,F
- ”1”の表現方法 · · · B,C
- ”2”の表現方法 · · · A,B,D,E,G
- ”3”の表現方法 · · · A,B,C,D,G
- ”4”の表現方法 · · · B,C,F,G
- ”5”の表現方法 · · · A,C,D,F,G
- ”6”の表現方法 · · · A,C,D,E,F,G
- ”7”の表現方法 · · · A,B,C,F
- ”8”の表現方法 · · · A,B,C,D,E,F,G
- ”9”の表現方法 · · · A,B,C,D,F,G
- ”A”の表現方法 · · · A,B,C,E,F,G
- ”B”の表現方法 · · · C,D,E,F,G
- ”C”の表現方法 · · · A,D,E,F
- ”D”の表現方法 · · · B,C,D,E,G
- ”E”の表現方法 · · · A,D,E,F,G
- ”F”の表現方法 · · · A,E,F,G

2. 回路の論理設計
   4 ビットで表される 1 桁の 16 進数を入力として、7SEG LED を駆動する 8 ビットの信号 LED を出力する組合せ回路を設計する。4 ビットの 2 進数 data を 7SEG LED に表示するためのデコーダを作成するに、まず上の海路の設計の仕様に基づいて 7SEG LED が点灯する場合は 1, そうでない場合は 0 となるようにデコーダの真理値表を作成した。7SEG LED の名前の配置と出力はそれぞれ A,B,C,D,E,F,G,DP が out6,out5,out4,out3,out2,out1,out0,out7 に順番に対応している。次に、組み合わせ回路を assign 文と関数で記述し、条件分岐には case 構文を用いることで、16 種類の入力からそれぞれ異なる 7SEG LED が点灯するように設計を行った。

3. CAD 上での設計
   ボード上での 4 ビットの入力を受け取るための data、7SEG LED を駆動する 8 ビットの信号を出力するための LED、そして出力する場所を定める temp を部品として用いた。また、7SEG LED を駆動する 8 ビットの信号を出力するためにボードの 7 セグメント LED と 4 ビットの入力をを与えるために Dip SW A0,A1,A2,A3 を用いた。LED の出力には下位ビットからそれぞれ PIN A4,PIN B3,PIN B4,PIN A5,PIN A6,PIN B6,PIN A3,PIN B5 を用いた。また、LED の表示させる場所 temp は下位ビットから PIN C3,PIN C4,PIN E5,PIN E6 を用いた。そして、ボード上での 4 ビットの入力 data を与えるために下位ビットからそれぞれ PIN D10,PIN C10,PIN B10,PIN A10 を指定した。

4. 動作結果の検証
   以下に入力データがそれぞれ 0000,0001,0010,0011,0100,0101,0110,0111,1000,1001,1010,1011,1100,1101,1110,1111 の順番に 50ns ごとに変化する状態のシミュレーション結果を表示する。図 2 ではそれぞれ出力は 01111110,00110000,01101101,01111001,00110011,01011011,01011111,01110010 となり、想定通りの値となっている。また、図 2 でもそれぞれ出力は 01111111,01110011,01110111,00011111,01001110,00111101,01001111,01000111,00000000 となっており、このシミュレーション結果から正しく動作していることが分かる。また、回路を FPGA にダウンロードした場合も，正しく動作することが確認できた。

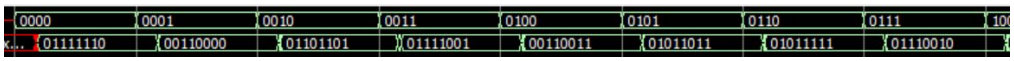

図 2: 0000 のときのシミュレーション結果

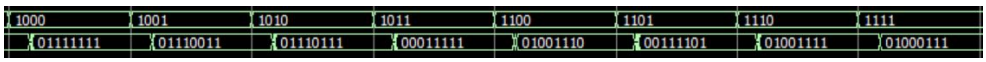

図 3: 0000 のときのシミュレーション結果

5. 性能評価
   使用した論理要素、ピンそれぞれは合計 7,16 個であった。動作可能速度については入力から出力までの最大値を採用して 9.776ns となった。この結果については十分に高速であると考えられる。改善点としては、今回は function と assign 構文を用いて組み合わせ回路を記述したが、この他にも always 文を用いれば記述量を少なくすることができると考えられ、改善が見込まれる。

工学部情報学科計算機科学コース
1029-31-6811
橘 大佑

計算機科学実験及演習 3 HW

導入課題

2021 年 4 月 14 日

課題 2,3

1. 回路の仕様の決定
   課題 1 で設計した 4 ビットのデータ入力から 1 桁の 16 進数を表示する回路を改良し、その回路を用いて 10 進数 4 桁の数字を表示して 1 クロックで 1 ずつカウントアップする回路を設計した。課題 2 と 3 の違いはカウントアップを停止させるボタンを押すか押さないか、だけあり設計は全て等しいため、レポートはふたつを統合して書く。また、ボード上で 4 桁の数字を表示させる場所が課題 1 で文字を表示させた場所と異なるため、課題 1 とはどの場所を表示させるか、という 4 ビットから 8 ビットの割り当ては資料をもとに変更した。

2. 回路の論理設計
   1 ビットの入力 clock と 1 ビットの入力 stop から 4 桁のカウンターを表示させるために下位ビットから LEDA,LEDB,LEDC,LEDD のそれぞれ 8 ビットを出力させた。今回は 10 進のカウンターであるために 0 から F までの文字のうち A から F の英字は使わず、0 から 9 の数字のみが表示されるカウンターであるため、それぞれ 4 ビットの dataA,dataB,dataC,dataD を定義して、クロックが立ち上がったときに dataA に 1 を足していくように設計し、dataA,dataB,dataC,dataD が 4 ビットの 2 進表現で 1001、つまり 10 進では 9 のときに、上位の data に桁上げを行う、例えば dataA が 1001 の状態から桁上げがあれば dataB に 1 を追加し、dataA には 0000 を代入する、ことで 4 桁 10 進のカウンターを設計した。課題 3 のカウントを停止させる stop については、停止ボタンが働いているとき、つまり stop==1 のときはカウントを行わず、そのままの状態を保つことでカウントの一時停止を実現した。

3. CAD 上での設計
   ボード上でクロック clock, 停止ボタン stop の入力ふたつを受け取り、4 桁の 7SEG LED を駆動する 8 ビットの信号を出力するための LEDA,LEDB,LEDC,LEDD、そしてボード上で出力する場所を定める loc を部品として用いた。クロック、停止ボタンのピンの割り当てはそれぞれクロック用ロータリースイッチ SW2 と Dip SW A3 を用いた。はじめはクロックの入力のために Dip SW スイッチを用いていたが、そうするとチャタリングが起こりカウントが 1 づつではなく 2,3 づつカウントされたりなどしたので、クロック用のスイッチを変更した。クロック用ロータリースイッチはつまみを左側に回していくとカウントのスピードが速くなった。点灯させる場所の指定には SEG SEL0 の 8 つの LED を点灯させた。また、4 桁の LED の出力には MU500-7SEG キットの左上の 4 つを割り当てた。ここで SEG SEL0 の右 4 つの LED も 8 ビット全て点灯してしまったが、右 4 つの LED でのカウントには影響しないためそのまま実験を進めた。これら右 4 つの LED を点灯させないようにするには SEG SEL1 での点灯または消灯をコードに含めて、ピン割り当てをする必要がある。

4. 動作結果の検証
   タイミング制約の設定に関してはクロックの周期と同じく 10ns ごとに変化するように設定した。ベンチマークでは最初にクロックとストップに 0 を割り当て、10ns ごとにクロックが変化するように設定し、シミュレーションを行った。

図 4: クロックの動作回数が 0 から 9 のときのシミュレーション結果

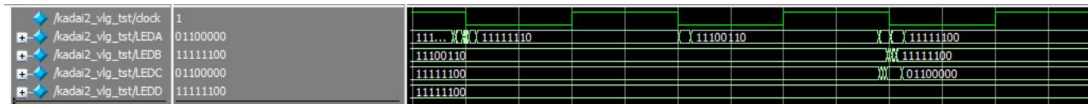

図 5: クロックの動作回数が 99 から 100 のときのシミュレーション結果

図 4 では 1 の位の LEDA が課題 1 で設計したデコーダの数字通りに表示されており、残りの LED3 つは 0 を表すビットのまま変化していないことが分かる。

図 5 ではカウントが 99 のときには 1 の位の LEDA と 10 の位の LEDB が課題 1 で設計したデコーダの数字通りに 9 を意味する 8 ビットが表示されており、残りの LEDC と LEDD は 0 を表している。つぎにカウントされるときには LEDA と LEDB は 0 を表す 8 ビットになっており、LEDC が 1 を表す 8 ビットをしめしている。これで 99 から 100 へのカウントも正しくされていることが分かる。

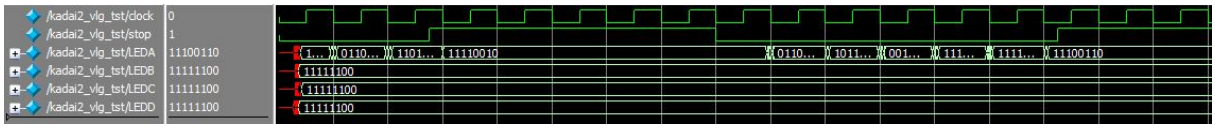

図 6: 停止スイッチを押してカウントを停止させたときのシミュレーション結果

図 6 は課題 3 に関して行ったシミュレーション結果である。最初にクロックとストップに 0 を割り当て、10ns ごとにクロックが変化するように設定したのは共通である。開始してから 55ns 後に stop に 1 を代入、そこから 105ns 後に stop に 0 を代入、最後にそこから 125ns 後に再び stop に 1 を代入している。最初に stop に 1 が代入されてカウントが停止するまでは課題 2 と同じようにカウントが行われている。そこから次に stop に 0 が代入されるまではカウントが停止し、LEDA,LEDB,LEDC,LEDD の値は変化していないことが分かる。そこから再び stop が 0 になるとカウントが再開し、再び stop が 1 になるとカウントが停止していることが分かる。

5. 性能評価
使用した論理要素、レジスター、ピンそれぞれは合計 54,16,35 個であった。また、クロック最大周波数は 173.91MHz であった。この結果については十分に高速であると考えられる。

課題 1 の HDL(kadai1.v)

```
module kadai1 (data, LED, temp);
  input [3:0] data;
  output [7:0] LED;
  output [3:0] temp;

  wire [3:0] data;
  wire [7:0] LED;
  //wire [3:0] temp;

  assign LED = dec(data);

  assign temp = 4'b0111;

  function [7:0] dec;
  input [3:0] din;

  case(din)
    4'b0000 : dec = 8'b0111_1110;
 4'b0001 : dec = 8'b0011_0000;
 4'b0010 : dec = 8'b0110_1101;
 4'b0011 : dec = 8'b0111_1001;
 4'b0100 : dec = 8'b0011_0011;
 4'b0101 : dec = 8'b0101_1011;
 4'b0110 : dec = 8'b0101_1111;
 4'b0111 : dec = 8'b0111_0010;
 4'b1000 : dec = 8'b0111_1111;
 4'b1001 : dec = 8'b0111_0011;
 4'b1010 : dec = 8'b0111_0111;
 4'b1011 : dec = 8'b0001_1111;
 4'b1100 : dec = 8'b0100_1110;
 4'b1101 : dec = 8'b0011_1101;
 4'b1110 : dec = 8'b0100_1111;
 4'b1111 : dec = 8'b0100_0111;
```

```
 default : dec = 8'b0000_0000;
 endcase

endfunction
endmodule
```

課題 2 の HDL(kadai1 と kadai2)

```
module kadai1 (data, LED, loc);
  input [3:0] data;
  output [7:0] LED;
  output loc;

  wire [3:0] data;
  wire [7:0] LED;
  //wire [3:0] loc;

  assign LED = dec(data);

  function [7:0] dec;
  input [3:0] din;

  case(din)          //7654_3210
    4'b0000 : dec = 8'b1111_1100;//0
 4'b0001 : dec = 8'b0110_0000;//1
 4'b0010 : dec = 8'b1101_1010;//2
 4'b0011 : dec = 8'b1111_0010;//3
 4'b0100 : dec = 8'b0110_0110;//4
 4'b0101 : dec = 8'b1011_0110;//5
 4'b0110 : dec = 8'b0011_1110;//6
 4'b0111 : dec = 8'b1110_0100;//7
 4'b1000 : dec = 8'b1111_1110;//8
 4'b1001 : dec = 8'b1110_0110;//9
 4'b1010 : dec = 8'b0111_0111;//A
 4'b1011 : dec = 8'b0001_1111;//B
 4'b1100 : dec = 8'b0100_1110;//C
 4'b1101 : dec = 8'b0011_1101;//D
 4'b1110 : dec = 8'b0100_1111;//E
 4'b1111 : dec = 8'b0100_0111;//F
```

```
 default : dec = 8'b0000_0000;
 endcase

endfunction
endmodule


module kadai2 (
    input stop, clock,
    output reg [7:0] LEDA,
 output reg [7:0] LEDB,
 output reg [7:0] LEDC,
 output reg [7:0] LEDD,
 output reg loc ) ;

 reg [3:0] dataA;
 reg [3:0] dataB;
 reg [3:0] dataC;
 reg [3:0] dataD;
 reg locA;
 reg locB;
 reg locC;
 reg locD;

 kadai1 led0 (.data(dataA), .LED(LEDA), .loc(locA));
 kadai1 led1 (.data(dataB), .LED(LEDB), .loc(locB));
 kadai1 led2 (.data(dataC), .LED(LEDC), .loc(locC));
 kadai1 led3 (.data(dataD), .LED(LEDD), .loc(locD));

always @( posedge clock ) begin
 loc <= 1'b1;
    if ( stop == 1 ) begin
        dataA <= dataA ;
  dataB <= dataB ;
  dataC <= dataC ;
  dataD <= dataD ;
    end else if (dataA == 4'b1001) begin
     if (dataB == 4'b1001) begin
      if (dataC == 4'b1001) begin
```

```
    if (dataD == 4'b1001) begin
     dataA <= 4'b0000;
  dataB <= 4'b0000;
  dataC <= 4'b0000;
  dataD <= 4'b0000;
 end else begin
     dataA <= 4'b0000;
  dataB <= 4'b0000;
  dataC <= 4'b0000;
     dataD <= dataD + 1;
 end
end else begin
    dataA <= 4'b0000;
    dataB <= 4'b0000;
    dataC <= dataC + 1;
end
  end else begin
      dataA <= 4'b0000;
         dataB <= dataB + 1;
  end
 end else begin
     dataA <= dataA + 1;
    end

end
endmodule
```

工学部情報学科計算機科学コース
1029-31-6811
橘 大佑

導入課題

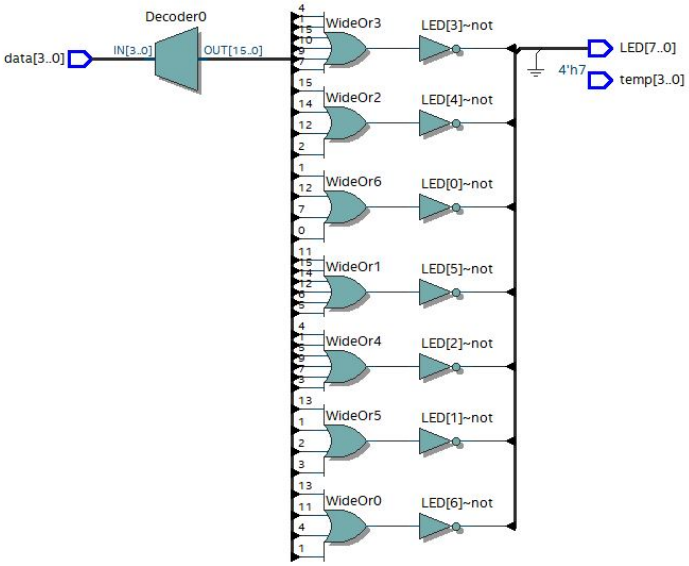

図 7: 課題 1 の回路図

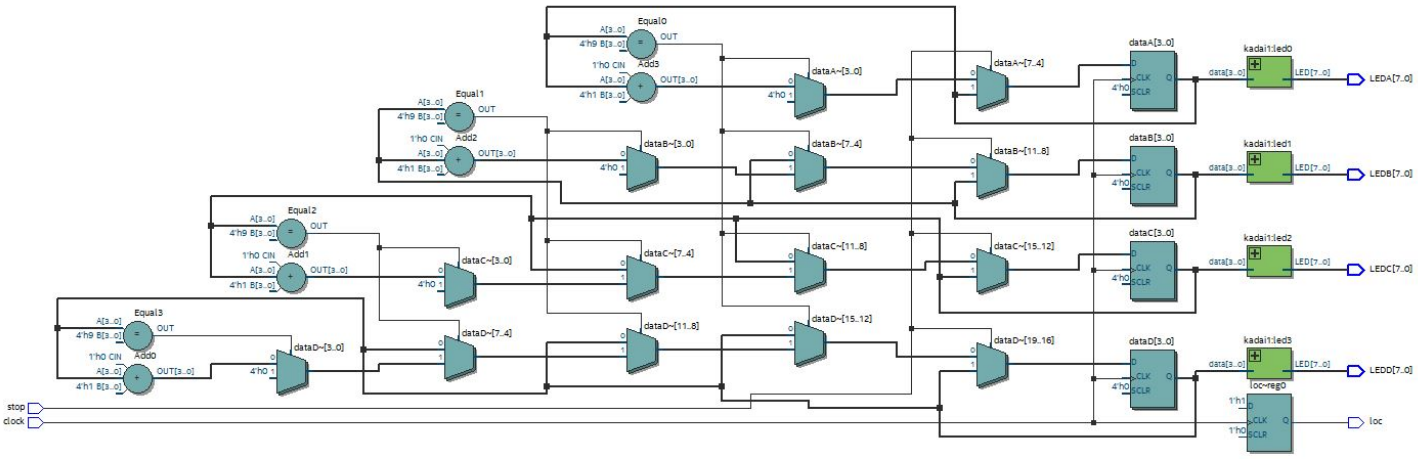

図 8: 課題 2,3 の回路図

工学部情報学科計算機科学コース
1029-31-6811
橘 大佑

導入課題

図 9: 0 から F の 16 種類の文字をボード上で点灯させたときの様子