基礎から学ぶ Verilog HDL & FPGA 設計

第11回
CPUの設計(2) Verilog HDLによる記述

中野浩嗣，伊藤靖朗

デバイスの記事　ビギナーズ

前回は，学習用CPU「TINYCPU」のアーキテクチャの詳細を検討した．今回はモジュールとそのポート，制御線，制御線のロジックなどを基に，TINYCPUをVerilog HDLで記述し，シミュレーションにより動作を確認する．　（筆者）

● 各種定数の定義

前回(2008年9月号，pp.149-154)は，学習用CPU「TINYCPU」のアーキテクチャの詳細を検討しました(**図1**)．後述の**表1**，**表2**，**表3**は，それぞれ，前回作成したモジュールとそのポート一覧，制御線一覧と制御線のロジックの一覧です．

今回はTINYCPUをいよいよVerilog HDLで記述します．まず，準備として，状態，算術論理演算，機械語命令を識別するための定数を，define文を用いて宣言しておきましょう．**リスト1**は，そのVerilog HDL記述です．このVerilog HDL記述を格納したファイルをdefs.vとします．この定数をほかのVerilog HDL記述で用いるには，その1行目に，以下の記述を加えます．

```
`include "defs.v"
```

この`includeはC言語の#includeと同様に，指定したファイルの中身がこの場所に挿入されることになります．算術論理演算回路alu.vとステート・マシンstate.vの中で記述されているdefine文も削除して，このinclude文に置き換えましょう．このようにしておくと，状態や機械語命令への定数の割り当てを変更する場合，defs.vだけ書きかえるだけでよく，バグが発生しにくくなります．

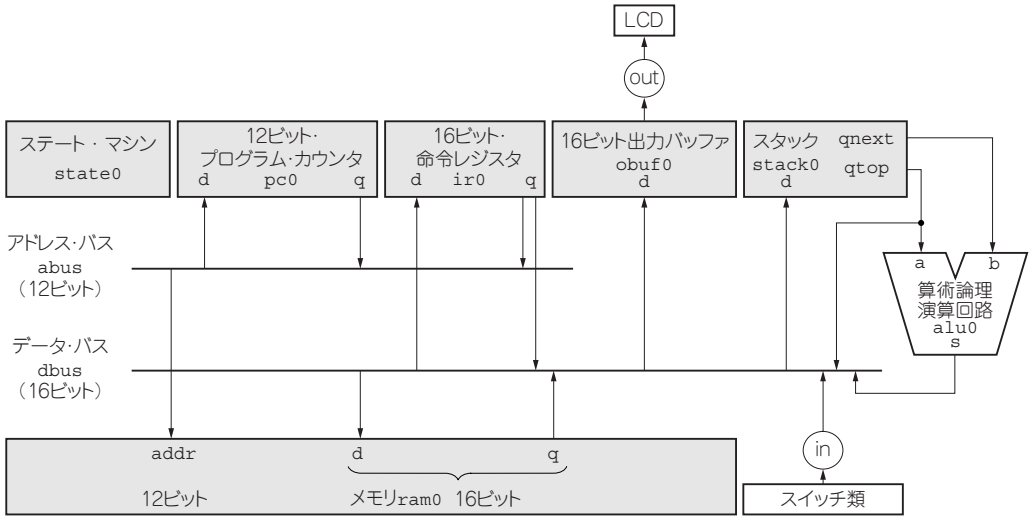

図1　TINYCPUのアーキテクチャ

Keyword　define，include，CPU，機械語プログラミング，算術論理演算回路，ステート・マシン，テストベンチ，シミュレーション

● 入力ポート，バス，制御線の宣言

次に，TINYCPUの本体を設計します．**リスト2**はそのVerilog HDL記述です．1行目でdefs.vを読み込みます．3行目～9行目で入出力ポートを宣言します．入力は1ビットのclk，reset，runと16ビットのinです．出力は，現在の状態csとプログラム・カウンタの値pcout，命令レジスタの値irout，スタック・トップの値qtop，アドレス・バスの値abus，データ・バスの値dbus，出力バッファの値outです．アドレス・バスabusは，後でalways文を用いたマルチプレクサとして設計するので，11行目でレジスタ型変数として定義します．12行目では，17本の制御線を1ビットのレジスタ型変数として定義します．これらの制御線は，後でalways文を用いた組み合わせ回路の出力となります．

リスト1　TINYCPUで用いる定数宣言のVerilog HDL記述defs.v

```
 1 `define IDLE   3'b000
 2 `define FETCHA 3'b001
 3 `define FETCHB 3'b010
 4 `define EXECA  3'b011
 5 `define EXECB  3'b100
 6
 7 `define  ADD  5'b00000
 8 `define  SUB  5'b00001
 9 `define  MUL  5'b00010
10 `define  SHL  5'b00011
11 `define  SHR  5'b00100
12 `define  BAND 5'b00101
13 `define  BOR  5'b00110
14 `define  BXOR 5'b00111
15 `define  AND  5'b01000
16 `define  OR   5'b01001
17 `define  EQ   5'b01010
18 `define  NE   5'b01011
19 `define  GE   5'b01100
20 `define  LE   5'b01101
21 `define  GT   5'b01110
22 `define  LT   5'b01111
23 `define  NEG  5'b10000
24 `define  BNOT 5'b10001
25 `define  NOT  5'b10010
26
27 `define HALT  4'b0000
28 `define PUSHI 4'b0001
29 `define PUSH  4'b0010
30 `define POP   4'b0011
31 `define JMP   4'b0100
32 `define JZ    4'b0101
33 `define JNZ   4'b0110
34 `define IN    4'b1101
35 `define OUT   4'b1110
36 `define OP    4'b1111
```

リスト2　TINYCPUのVerilog HDL記述tinycpu.v

```
 1 `include "defs.v"
 2
 3 module tinycpu(clk, reset, run, in, cs, pcout, irout, qtop, abus, dbus, out);
 4
 5   input clk,reset,run;
 6   input [15:0] in;
 7   output [2:0] cs;
 8   output [15:0] irout, qtop, dbus, out;
 9   output [11:0] pcout, abus;
10   wire [15:0] qnext, ramout, aluout;
11   reg [11:0] abus;
12   reg halt, cont, pcinc, push, pop, abus2pc, dbus2ir, dbus2qtop, dbus2ram,
          dbus2obuf,pc2abus, ir2abus, ir2dbus, qtop2dbus, alu2dbus, ram2dbus, in2dbus;
13
14   counter #(12) pc0(.clk(clk),.reset(reset),.load(abus2pc),.inc(pcinc),.d(abus),
.q(pcout));
15   counter #(16) ir0(.clk(clk),.reset(reset),.load(dbus2ir),.inc(0),.d(dbus),
                                                                    .q(irout));
16   state state0(.clk(clk),.reset(reset),.run(run),.cont(cont),.halt(halt),
                                                                     .cs(cs));
17   stack stack0(.clk(clk),.reset(reset),.load(dbus2qtop),.push(push),.pop(pop),
                                                .d(dbus),.qtop(qtop),.qnext(qnext));
18   alu alu0(.a(qtop),.b(qnext),.f(irout[4:0]),.s(aluout));
19   ram #(16,12,4096)  ram0(.clk(clk),.load(dbus2ram),.addr(abus),.d(dbus),
                                                                  .q(ramout));
20   counter #(16) obuf0(.clk(clk),.reset(reset),.load(dbus2obuf),.inc(0),
                                                               .d(dbus),.q(out));
21
22   always @(pc2abus or ir2abus or pcout or irout)
23     if(pc2abus) abus = pcout;
24     else if(ir2abus) abus = irout[11:0];
25     else abus = 12'hxxx;
26
27   assign dbus = ir2dbus ? {{4{irout[11]}},irout[11:0]} : 16'hzzzz;
28   assign dbus = qtop2dbus ? qtop : 16'hzzzz;
29   assign dbus = alu2dbus ? aluout : 16'hzzzz;
30   assign dbus = ram2dbus ? ramout : 16'hzzzz;
31   assign dbus = in2dbus ? in : 16'hzzzz;
32
33   always @(cs or irout or qtop)
34   begin
35     halt = 0; cont = 0; pcinc = 0; push = 0; pop = 0; abus2pc = 0; dbus2ir = 0;
            dbus2qtop = 0; dbus2ram = 0; dbus2obuf = 0; pc2abus = 0; ir2abus = 0;
             ir2dbus = 0; qtop2dbus = 0; alu2dbus = 0; ram2dbus = 0; in2dbus = 0;
36     if(cs == `FETCHA)
37       begin
38         pcinc = 1; pc2abus = 1;
39       end
40     else if(cs == `FETCHB)
41       begin
42         ram2dbus = 1; dbus2ir = 1;
43       end
44     else if(cs == `EXECA)
45       case(irout[15:12])
46       `PUSHI:
47         begin
48           ir2dbus = 1; dbus2qtop = 1; push = 1;
49         end
50       `PUSH:
51         begin
52           ir2abus = 1; cont = 1;
53         end
54       `POP:
55         begin
```