Semestrální projekt č. 1 z předmětu AD4M36PAP, školní rok 2012-2013

Téma: Návrh jednoduchého výpočetního systému

ČVUT FEL, obor VT, 1. ročník, e-mail: valiclud@fel.cvut.cz

Jméno: Ludvík Valíček

Datum: 5. prosince 2012

1. Zadání

Navrhněte a popište v jazyce Verilog jednoduchý výpočetní systém pozůstávající z procesoru, oddělené instrukční a datové paměti. Procesor bude podporovat instrukce add, sub, and, or, slt, addi, lw, sw a beq ve formátu instrukční sady MIPS. Součástí procesoru musí být řídicí jednotka a aritmeticko-logická jednotka. Funkčnost Vašeho návrhu demonstrujte simulací níže uvedeného programu.

```
if(a<=b)
    a = b-a;
else
    for(int i=0; i!=c; i++)
        a += i;
a = a & 3;
while(1);
```

Předpokládejte, že proměnné a, b a c jsou typu int (4B) a jsou již uloženy v paměti na adresách 0x0010, 0x0014 a 0x0018 datové paměti, proměnná c je větší než 0, a program je již uložen v instrukční paměti od adresy 0x0000 (tj. není potřeba řešit zavádění programu). Procesor po resetu (po spuštění) začíná vykonávat instrukce od adresy 0x0000.

2. Řešení

2.1. Přepis programu do jazyka symbolických adres

Pos	Kod programu	Kod programu hexa
1	lw, $2, 0($0)	8c020000
2	lw, $3, 4($0)	8c030004
3	lw, $4, 8($0)	8c040008
4	slt $5, $3, $2	0062282a
5	beq $5, $0, 06	10a00005
6	addi $6, $0, 0	20060000
7	beq $6, $4, 05	10c40004
8	add $2, $2, $6	461020
9	addi $6, $6, 1	20c60001
10	beq $0, $0, -3	1000fffd
11	sub $2, $2, $3	.00431022
12	addi $7, $0, 3	20070003
13	and $2, $2, $7	.00471024
14	addi $8, $0, 1	20080001
15	sw, $2, 0($0)	ac020000
16	slt $9, $8, $0	.0100482a
17	beq $9, $0, -1	1120ffff

Ukázka kódování dvou instrukcí:

Kod programu	OpCode	RS	RT	Immediate
lw, $2, 0($0)	100011	00000	00010	0000000000000000
beq $5, $0, 06	000100	00101	00000	0000000000000110

2.2. Popis navrženého systému v jazyce Verilog

Systém se skládá z oddělené datové paměti (DataMemory.v) a instrukční paměti (instMemory.v) a z řídící jednotky (controllerOnly). Řídící jednotka v sobě zahrnuje čítač instrukcí (PC), registry (registr []) a jednotlivé cykly zpracování instrukcí. Instrukce jsou zpracovávány v cyklech:

S0 – čtení instrukce z instrukční paměti a její uložení do registru instData
S1 – zvýšení čítače o 1, dekódování instrukce a zpracování (výpočet) instrukce
S2 – načtení dat z oddělené datové paměti
S3 – uložení dat do oddělené datové paměti

Pro snadnější odladění programu jsou použity pomocné proměnné OpCode, IR, result, regS, regT, regD, regI, které monitorují běh programu.

Modul datové paměti:

```
module DataMemory( clk, dataAddress, writeData, memWrite, memRead, memData );

	input [31:0] dataAddress, writeData;
	input memRead, memWrite, clk;
	output [31:0] memData;
	wire [31:0] memData;
	reg [31:0] register [512:0];

	always @ (posedge clk) begin
		if(memWrite==1'b1)begin
		register[dataAddress] <= writeData;
		end
	end

	assign memData = (memRead==1'b1)? register[dataAddress] : 0;

	initial begin //storing input data
	register[32'd0] <= 32'd4; //a
	register[32'd4] <= 32'd2; //b
	register[32'd8] <= 32'd6; //c
	register[32'd12] <= 32'd0; //i
	end
endmodule
```

Modul instrukční paměti:

```
module instMemory( clk, address, instRead, instData );

  input [31:0] address;
       input instRead, clk;
       output [31:0] instData;
       wire [31:0] instData;
       reg [31:0] register [512:0];

       assign instData = (instRead==1'b1)? register[address] : 0;
       initial begin //storing program instruction data

       register[0] <= 32'h8c020000;
       register[1] <= 32'h8c030004;
       register[2] <= 32'h8c040008;
       register[3] <= 32'h0062282a;
       register[4] <= 32'h10a00005;
       register[5] <= 32'h20060000;
       register[6] <= 32'h10c40006;
       register[7] <= 32'h00461020;
       register[8] <= 32'h20c60001;
       register[9] <= 32'h1000fffc;
       register[10] <= 32'h00431022;
       register[11] <= 32'h20070003;
       register[12] <= 32'h00471024;
       register[13] <= 32'h20080001;
       register[14] <= 32'hac020000;
       register[15] <= 32'h0100482a;
       register[16] <= 32'h1120ffff;
end
endmodule
```

Modul řídící a výpočetní jednotky:

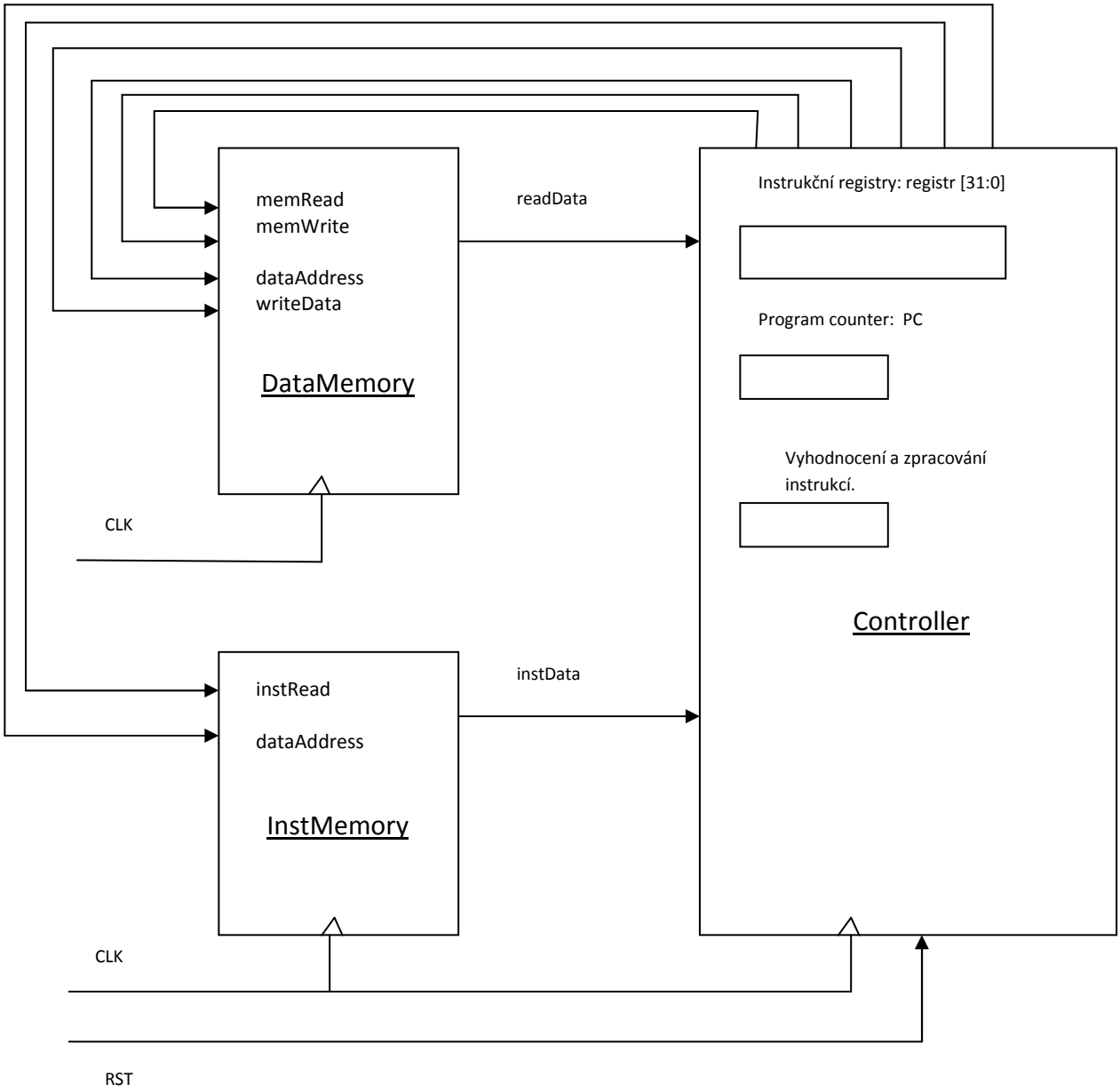
```
module controllerOnly( clk, reset, opCode, PC, dataWrite,
   dataRead, instRead, instData,  dataAddress, memData, readData, IR, result, regS,
regT, regD, regI);

   input clk, reset;
       input [31:0] instData, readData;
       output [5:0] opCode;
   output dataWrite, dataRead, instRead;
   output [31:0] memData, dataAddress, IR, result;
       output [15:0] PC, regI;
       output [4:0] regS, regT, regD;

       reg dataWrite, dataRead, instRead;
       reg  [31:0] memData,  dataAddress;
       reg [4:0] state =0, nextstate, regS, regT, regD;
       reg [5:0] opCode, funct, i;
       reg [31:0] registr [31:0];
```

```
 reg [31:0] IR, result;
 reg  [15:0] PC = 0;
 reg  [15:0] regI ;

 parameter S0=0; //fetch
 parameter S1=1; //decode
 parameter S2=2; //loading data from memory
 parameter S3=3; //storing word to memory

initial begin
        for (i=0;i<32;i=i+1)
        registr [i] = 32'd0;
end

always@(posedge clk) begin
                        state=nextstate;
                end

always @(state,reset) begin

                 if(reset) begin
                          state = S0;
                          PC = 0;
                          end
                 else
                          begin

        case(state)

        S0: begin //fetch instruction, inst is in regist IR
          dataAddress = PC;
                instRead=1'b1;
                dataRead=1'b0;
                dataWrite = 1'b0;
                nextstate=S1;
        end
        S1: begin //decode, instruction is in registr IR, increment PC
                IR = instData;
                instRead=1'b0;
                PC = PC + 1; // incrementing programme counter
                opCode = instData [31:26];
                regS = instData [25:21];
                regT = instData [20:16];
                regI = instData [15:0];
                regD = instData [15:11];

                if((opCode==6'b101011)) begin // op code is sw
                dataAddress = IR[25:21] + IR[15:0];
                dataWrite=1'b1;
                nextstate = S3;
                end

                if(opCode==6'b100011) begin // op code is lw
                dataAddress = regS + IR[15:0];
                dataRead=1'b1;
```

```
        nextstate = S2;
        end

        if(opCode==6'b000000) begin // R type instruction

                if(11'b00000100000 == IR[10:0]) begin // R type ADD
                result = registr [regS] + registr [regT];
                registr[ IR[15:11]] = result;
                nextstate=S0;
                end
                if(11'b00000100100 == IR[10:0]) begin // R type AND
                result = registr [regS] & registr [regT];
                registr[ IR[15:11]] = result;
                nextstate=S0;
                end
                if(11'b00000100101 == IR[10:0]) begin // R type OR
                result = registr [regS] | registr [regT];
                registr[ IR[15:11]] = result;
                nextstate=S0;
                end
                if(11'b00000101010 == IR[10:0]) begin // R type SLT

                if ( registr [regS] < registr [regT] )
                                result = 31'd1;
                else
                                result = 31'd0;
                registr[ regD ] = result;
                nextstate=S0;
                end
                if(11'b00000100010 == IR[10:0]) begin // R type SUB
                result = registr [regT] - registr [regS];
                registr[ regD] = result;
                nextstate=S0;
                end
end //end R if

                if(opCode==6'b001000) begin // ADDItype instruction
                result =  registr [regS] + regI;
                registr[ regT] = result;
                nextstate = S0;
                end

                if(opCode==6'b000100) begin // beq instruction
                result = registr [ regT ] - registr [ regS ] ;
                if ( registr [ regS ] == registr [ regT ] )
                                PC = PC + regI;
                nextstate = S0;
                end
        end // end S1
S2: begin // lw
        registr [ IR [20:16] ] = readData;
        result = registr[ IR[20:16]];
        nextstate=S0;
        end
S3: begin //sw
```

```
                memData = registr [ regT ];
                result = registr[ regT ];
                nextstate=S0;
                end
            endcase
            end
            end
endmodule
```

2.3. Blokové schéma navrženého systému



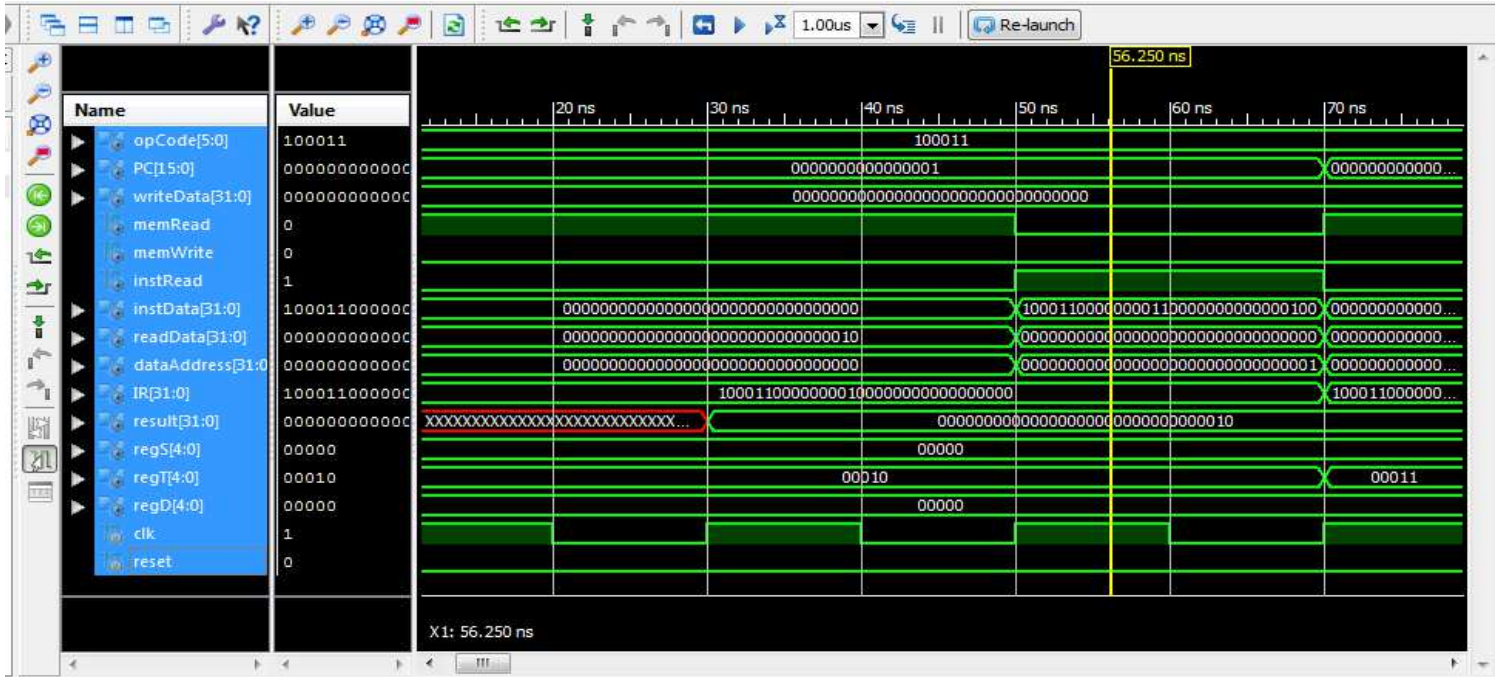
2.4. Výsledky simulace

2.4.1. Simulace pro a <= b

Vstupní hodnoty: a = 2, b = 4, c = 6

Konečný výsledek před operací &: a = 2 & 3;
a = 00 .. 0010

Ukázka začátku simulace:



Podrobné výsledky simulace:

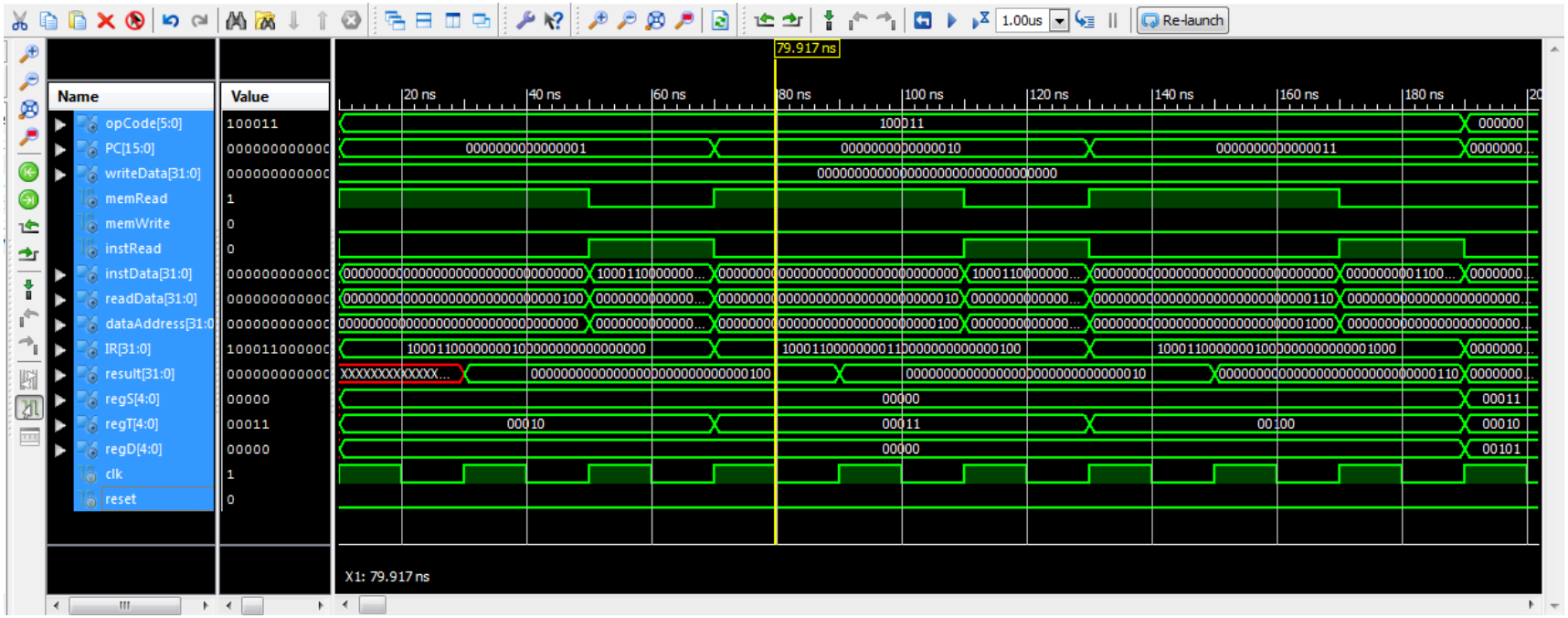
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
PC : 0; memRead : 0; readData : 00000000; instRead : 1; IR : xxxxxxxx; dataAddress : 00000000; result : xxxxxxxx; regS : xxxxx; regT : xxxxx; regD : xxxxx;
PC : 1; memRead : 1; readData : 00000002; instRead : 0; IR : 8c020000; dataAddress : 00000000; result : xxxxxxxx; regS : 00000; regT : 00010; regD : 00000;
PC : 1; memRead : 1; readData : 00000002; instRead : 0; IR : 8c020000; dataAddress : 00000000; result : 00000002; regS : 00000; regT : 00010; regD : 00000;
PC : 1; memRead : 0; readData : 00000000; instRead : 1; IR : 8c020000; dataAddress : 00000001; result : 00000002; regS : 00000; regT : 00010; regD : 00000;
PC : 2; memRead : 1; readData : 00000004; instRead : 0; IR : 8c030004; dataAddress : 00000004; result : 00000002; regS : 00000; regT : 00011; regD : 00000;
PC : 2; memRead : 1; readData : 00000004; instRead : 0; IR : 8c030004; dataAddress : 00000004; result : 00000004; regS : 00000; regT : 00011; regD : 00000;
PC : 2; memRead : 0; readData : 00000000; instRead : 1; IR : 8c030004; dataAddress : 00000002; result : 00000004; regS : 00000; regT : 00011; regD : 00000;
PC : 3; memRead : 1; readData : 00000006; instRead : 0; IR : 8c040008; dataAddress : 00000008; result : 00000004; regS : 00000; regT : 00100; regD : 00000;
PC : 3; memRead : 1; readData : 00000006; instRead : 0; IR : 8c040008; dataAddress : 00000008; result : 00000006; regS : 00000; regT : 00100; regD : 00000;
PC : 3; memRead : 0; readData : 00000000; instRead : 1; IR : 8c040008; dataAddress : 00000003; result : 00000006; regS : 00000; regT : 00100; regD : 00000;
PC : 4; memRead : 0; readData : 00000000; instRead : 0; IR : 0062282a; dataAddress : 00000003; result : 00000000; regS : 00011; regT : 00010; regD : 00101;
PC : 4; memRead : 0; readData : 00000000; instRead : 1; IR : 0062282a; dataAddress : 00000004; result : 00000000; regS : 00011; regT : 00010; regD : 00101;
PC : 10; memRead : 0; readData : 00000000; instRead : 0; IR : 10a00005; dataAddress : 00000004; result : 00000000; regS : 00101; regT : 00000; regD : 00000;
PC : 10; memRead : 0; readData : 00000000; instRead : 1; IR : 10a00005; dataAddress : 0000000a; result : 00000000; regS : 00101; regT : 00000; regD : 00000;
PC : 11; memRead : 0; readData : 00000000; instRead : 0; IR : 00431022; dataAddress : 0000000a; result : 00000002; regS : 00010; regT : 00011; regD : 00010;
PC : 11; memRead : 0; readData : 00000000; instRead : 1; IR : 00431022; dataAddress : 0000000b; result : 00000002; regS : 00010; regT : 00011; regD : 00010;
PC : 12; memRead : 0; readData : 00000000; instRead : 0; IR : 20070003; dataAddress : 0000000b; result : 00000003; regS : 00000; regT : 00111; regD : 00000;
PC : 12; memRead : 0; readData : 00000000; instRead : 1; IR : 20070003; dataAddress : 0000000c; result : 00000003; regS : 00000; regT : 00111; regD : 00000;
PC : 13; memRead : 0; readData : 00000000; instRead : 0; IR : 00471024; dataAddress : 0000000c; result : 00000002; regS : 00010; regT : 00111; regD : 00010;
PC : 13; memRead : 0; readData : 00000000; instRead : 1; IR : 00471024; dataAddress : 0000000d; result : 00000002; regS : 00010; regT : 00111; regD : 00010;
PC : 14; memRead : 0; readData : 00000000; instRead : 0; IR : 20080001; dataAddress : 0000000d; result : 00000001; regS : 00000; regT : 01000; regD : 00000;
PC : 14; memRead : 0; readData : 00000000; instRead : 1; IR : 20080001; dataAddress : 0000000e; result : 00000001; regS : 00000; regT : 01000; regD : 00000;
PC : 15; memRead : 0; readData : 00000000; instRead : 0; IR : ac020000; dataAddress : 00000000; result : 00000001; regS : 00000; regT : 00010; regD : 00000;
PC : 15; memRead : 0; readData : 00000000; instRead : 0; IR : ac020000; dataAddress : 00000000; result : 00000002; regS : 00000; regT : 00010; regD : 00000;
PC : 15; memRead : 0; readData : 00000000; instRead : 1; IR : ac020000; dataAddress : 0000000f; result : 00000002; regS : 00000; regT : 00010; regD : 00000;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 0100482a; dataAddress : 0000000f; result : 00000000; regS : 01000; regT : 00000; regD : 01001;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 0100482a; dataAddress : 00000010; result : 00000000; regS : 01000; regT : 00000; regD : 01001;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;

2.4.2. Simulace pro a>b

Vstupní hodnoty: a = 4, b = 2, c = 6,

Výsledek před operací &: a = 19 & 3;
a = 00 … 0011

Ukázka začátku simulace:



Podrobné výsledky simulace:

Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
PC : 0; memRead : 0; readData : 00000000; instRead : 1; IR : xxxxxxxx; dataAddress : 00000000; result : xxxxxxxx; regS : xxxxx; regT : xxxxx; regD : xxxxx;
PC : 1; memRead : 1; readData : 00000004; instRead : 0; IR : 8c020000; dataAddress : 00000000; result : xxxxxxxx; regS : 00000; regT : 00010; regD : 00000;
PC : 1; memRead : 1; readData : 00000004; instRead : 0; IR : 8c020000; dataAddress : 00000000; result : 00000004; regS : 00000; regT : 00010; regD : 00000;
PC : 1; memRead : 0; readData : 00000000; instRead : 1; IR : 8c020000; dataAddress : 00000001; result : 00000004; regS : 00000; regT : 00010; regD : 00000;
PC : 2; memRead : 1; readData : 00000002; instRead : 0; IR : 8c030004; dataAddress : 00000004; result : 00000004; regS : 00000; regT : 00011; regD : 00000;
PC : 2; memRead : 1; readData : 00000002; instRead : 0; IR : 8c030004; dataAddress : 00000004; result : 00000002; regS : 00000; regT : 00011; regD : 00000;
PC : 2; memRead : 0; readData : 00000000; instRead : 1; IR : 8c030004; dataAddress : 00000002; result : 00000002; regS : 00000; regT : 00011; regD : 00000;
PC : 3; memRead : 1; readData : 00000006; instRead : 0; IR : 8c040008; dataAddress : 00000008; result : 00000002; regS : 00000; regT : 00100; regD : 00000;
PC : 3; memRead : 1; readData : 00000006; instRead : 0; IR : 8c040008; dataAddress : 00000008; result : 00000006; regS : 00000; regT : 00100; regD : 00000;
PC : 3; memRead : 0; readData : 00000000; instRead : 1; IR : 8c040008; dataAddress : 00000003; result : 00000006; regS : 00000; regT : 00100; regD : 00000;
PC : 4; memRead : 0; readData : 00000000; instRead : 0; IR : 0062282a; dataAddress : 00000003; result : 00000001; regS : 00011; regT : 00010; regD : 00101;
PC : 4; memRead : 0; readData : 00000000; instRead : 1; IR : 0062282a; dataAddress : 00000004; result : 00000001; regS : 00011; regT : 00010; regD : 00101;
PC : 5; memRead : 0; readData : 00000000; instRead : 0; IR : 10a00005; dataAddress : 00000004; result : ffffffff; regS : 00101; regT : 00000; regD : 00000;
PC : 5; memRead : 0; readData : 00000000; instRead : 1; IR : 10a00005; dataAddress : 00000005; result : ffffffff; regS : 00101; regT : 00000; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 20060000; dataAddress : 00000005; result : 00000000; regS : 00000; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 20060000; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00110; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000006; regS : 00110; regT : 00100; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 00000007; result : 00000006; regS : 00110; regT : 00100; regD : 00000;
PC : 8; memRead : 0; readData : 00000000; instRead : 0; IR : 00461020; dataAddress : 00000007; result : 00000004; regS : 00010; regT : 00110; regD : 00010;
PC : 8; memRead : 0; readData : 00000000; instRead : 1; IR : 00461020; dataAddress : 00000008; result : 00000004; regS : 00010; regT : 00110; regD : 00010;
PC : 9; memRead : 0; readData : 00000000; instRead : 0; IR : 20c60001; dataAddress : 00000008; result : 00000001; regS : 00110; regT : 00110; regD : 00000;
PC : 9; memRead : 0; readData : 00000000; instRead : 1; IR : 20c60001; dataAddress : 00000009; result : 00000001; regS : 00110; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 1000fffc; dataAddress : 00000009; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 1000fffc; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 7; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000005; regS : 00110; regT : 00100; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 00000007; result : 00000005; regS : 00110; regT : 00100; regD : 00000;
PC : 8; memRead : 0; readData : 00000000; instRead : 0; IR : 00461020; dataAddress : 00000007; result : 00000005; regS : 00010; regT : 00110; regD : 00010;
PC : 8; memRead : 0; readData : 00000000; instRead : 1; IR : 00461020; dataAddress : 00000008; result : 00000005; regS : 00010; regT : 00110; regD : 00010;
PC : 9; memRead : 0; readData : 00000000; instRead : 0; IR : 20c60001; dataAddress : 00000008; result : 00000002; regS : 00110; regT : 00110; regD : 00000;
PC : 9; memRead : 0; readData : 00000000; instRead : 1; IR : 20c60001; dataAddress : 00000009; result : 00000002; regS : 00110; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 1000fffc; dataAddress : 00000009; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 1000fffc; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 7; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000004; regS : 00110; regT : 00100; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 00000007; result : 00000004; regS : 00110; regT : 00100; regD : 00000;
PC : 8; memRead : 0; readData : 00000000; instRead : 0; IR : 00461020; dataAddress : 00000007; result : 00000007; regS : 00010; regT : 00110; regD : 00010;
PC : 8; memRead : 0; readData : 00000000; instRead : 1; IR : 00461020; dataAddress : 00000008; result : 00000007; regS : 00010; regT : 00110; regD : 00010;
PC : 9; memRead : 0; readData : 00000000; instRead : 0; IR : 20c60001; dataAddress : 00000008; result : 00000003; regS : 00110; regT : 00110; regD : 00000;
PC : 9; memRead : 0; readData : 00000000; instRead : 1; IR : 20c60001; dataAddress : 00000009; result : 00000003; regS : 00110; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 1000fffc; dataAddress : 00000009; result : 00000000; regS : 00000; regT : 00000; regD : 11111;

PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 1000fffc; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 7; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000003; regS : 00110; regT : 00100; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 00000007; result : 00000003; regS : 00110; regT : 00100; regD : 00000;
PC : 8; memRead : 0; readData : 00000000; instRead : 0; IR : 00461020; dataAddress : 00000007; result : 0000000a; regS : 00010; regT : 00110; regD : 00010;
PC : 8; memRead : 0; readData : 00000000; instRead : 1; IR : 00461020; dataAddress : 00000008; result : 0000000a; regS : 00010; regT : 00110; regD : 00010;
PC : 9; memRead : 0; readData : 00000000; instRead : 0; IR : 20c60001; dataAddress : 00000008; result : 00000004; regS : 00110; regT : 00110; regD : 00000;
PC : 9; memRead : 0; readData : 00000000; instRead : 1; IR : 20c60001; dataAddress : 00000009; result : 00000004; regS : 00110; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 1000fffc; dataAddress : 00000009; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 1000fffc; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 7; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000002; regS : 00110; regT : 00100; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 00000007; result : 00000002; regS : 00110; regT : 00100; regD : 00000;
PC : 8; memRead : 0; readData : 00000000; instRead : 0; IR : 00461020; dataAddress : 00000007; result : 0000000e; regS : 00010; regT : 00110; regD : 00010;
ISim>
run all
PC : 8; memRead : 0; readData : 00000000; instRead : 1; IR : 00461020; dataAddress : 00000008; result : 0000000e; regS : 00010; regT : 00110; regD : 00010;
PC : 9; memRead : 0; readData : 00000000; instRead : 0; IR : 20c60001; dataAddress : 00000008; result : 00000005; regS : 00110; regT : 00110; regD : 00000;
PC : 9; memRead : 0; readData : 00000000; instRead : 1; IR : 20c60001; dataAddress : 00000009; result : 00000005; regS : 00110; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 1000fffc; dataAddress : 00000009; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 1000fffc; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 7; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000001; regS : 00110; regT : 00100; regD : 00000;
PC : 7; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 00000007; result : 00000001; regS : 00110; regT : 00100; regD : 00000;
PC : 8; memRead : 0; readData : 00000000; instRead : 0; IR : 00461020; dataAddress : 00000007; result : 00000013; regS : 00010; regT : 00110; regD : 00010;
PC : 8; memRead : 0; readData : 00000000; instRead : 1; IR : 00461020; dataAddress : 00000008; result : 00000013; regS : 00010; regT : 00110; regD : 00010;
PC : 9; memRead : 0; readData : 00000000; instRead : 0; IR : 20c60001; dataAddress : 00000008; result : 00000006; regS : 00110; regT : 00110; regD : 00000;
PC : 9; memRead : 0; readData : 00000000; instRead : 1; IR : 20c60001; dataAddress : 00000009; result : 00000006; regS : 00110; regT : 00110; regD : 00000;
PC : 6; memRead : 0; readData : 00000000; instRead : 0; IR : 1000fffc; dataAddress : 00000009; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 6; memRead : 0; readData : 00000000; instRead : 1; IR : 1000fffc; dataAddress : 00000006; result : 00000000; regS : 00000; regT : 00000; regD : 11111;
PC : 11; memRead : 0; readData : 00000000; instRead : 0; IR : 10c40004; dataAddress : 00000006; result : 00000000; regS : 00110; regT : 00100; regD : 00000;
PC : 11; memRead : 0; readData : 00000000; instRead : 1; IR : 10c40004; dataAddress : 0000000b; result : 00000000; regS : 00110; regT : 00100; regD : 00000;
PC : 12; memRead : 0; readData : 00000000; instRead : 0; IR : 20070003; dataAddress : 0000000b; result : 00000003; regS : 00000; regT : 00111; regD : 00000;
PC : 12; memRead : 0; readData : 00000000; instRead : 1; IR : 20070003; dataAddress : 0000000c; result : 00000003; regS : 00000; regT : 00111; regD : 00000;
PC : 13; memRead : 0; readData : 00000000; instRead : 0; IR : 00471024; dataAddress : 0000000c; result : 00000003; regS : 00010; regT : 00111; regD : 00010;
PC : 13; memRead : 0; readData : 00000000; instRead : 1; IR : 00471024; dataAddress : 0000000d; result : 00000003; regS : 00010; regT : 00111; regD : 00010;
PC : 14; memRead : 0; readData : 00000000; instRead : 0; IR : 20080001; dataAddress : 0000000d; result : 00000001; regS : 00000; regT : 01000; regD : 00000;
PC : 14; memRead : 0; readData : 00000000; instRead : 1; IR : 20080001; dataAddress : 0000000e; result : 00000001; regS : 00000; regT : 01000; regD : 00000;
PC : 15; memRead : 0; readData : 00000000; instRead : 0; IR : ac020000; dataAddress : 00000000; result : 00000001; regS : 00000; regT : 00010; regD : 00000;
PC : 15; memRead : 0; readData : 00000000; instRead : 0; IR : ac020000; dataAddress : 00000000; result : 00000003; regS : 00000; regT : 00010; regD : 00000;
PC : 15; memRead : 0; readData : 00000000; instRead : 1; IR : ac020000; dataAddress : 0000000f; result : 00000003; regS : 00000; regT : 00010; regD : 00000;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 0100482a; dataAddress : 0000000f; result : 00000000; regS : 01000; regT : 00000; regD : 01001;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 0100482a; dataAddress : 00000010; result : 00000000; regS : 01000; regT : 00000; regD : 01001;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 0; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;
PC : 16; memRead : 0; readData : 00000000; instRead : 1; IR : 1120ffff; dataAddress : 00000010; result : 00000000; regS : 01001; regT : 00000; regD : 11111;

3. Závěr

V této semestrální práci jsem si důkladně procvičil architekturu mikroprocesoru a instrukcí MIPS. Práci jsem ze začátku koncipoval jako „multicycle“ mikroprocesor, což nebylo zadáním této práce a také „multicycle“ mikroprocesor je složitější na implementaci. Teprve po konzultacích jsem takto navržený mikroprocesor upravil a výsledkem je výše uvedený funkční Verilog kód. Vypracování úlohy pro mě bylo přínosem a pomohlo mi lépe pochopit základy architektury mikroprocesoru a jeho programování.