بسم الله الرحمن الرحیم

ساختار کامپیوتر

امیر حسین رستمی

96101635

گزارش کار فاز دوم پروژه

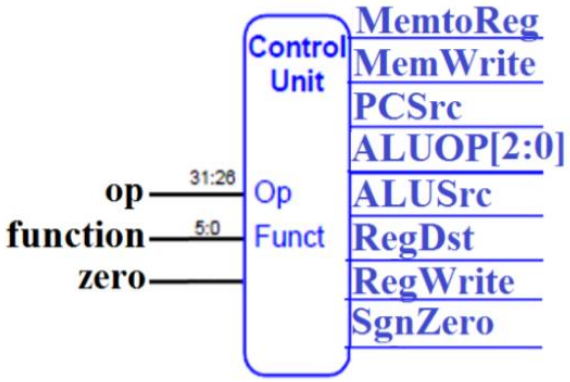
دکتر موحدیان عطار

دانشگاه صنعتی شریف بهار 98

گزارش فاز دوم :

داریم که در این قسمت به کمک بررسی آپکد و فانکشن کد دستورات نوشته شده به محاسبه ی خروجی های مدنظر می پردازیم.

- توجه : PCSrc در اصل and واحد branch و zero می باشد.



هدف:

هدف طراحی ماژولی با ورودی و خروجی های ذکر شده می باشد که با دریافت شش بیت بالا و پایین دستور و ورودی تک بیتی zero مربوط به صفر و یک بودن خروجی ALU، تمامی بیت های کنترلی را با توجه به نوع دستور مقدار دهی کند.

دستورات مورد نیاز برای پیاده سازی:

ADD - ADDU - SUB - SUBU - AND - OR - XOR - NOR - SLT - SLTU - LW - SW
BEQ - BNE - ANDI - ORI - XORI - ADDI - ADDIU - SLTI - SLTIU

قطعه کد نوشته شده برای این قسمت : بخش اول کد ....

```
module Controller(
input [5:0] op,
input [5:0] funct,
input zero,

output MemtoReg,
output MemWrite,
output PCSrc,
output [2:0]ALUOP,

output ALUSrc,
output RegDst,
output RegWrite,
output SgnZero

    );

reg [9:0] outs;

always@(*)
begin
casex(op)

  6'b100011: outs=10'b1_0_0_000_1_0_1_1;
  6'b101011: outs=10'b0_1_0_000_1_0_0_1;
  6'b000100: outs={2'b00,zero,7'b001_0101};
  6'b000101: outs={2'b00,~zero,7'b001_0101};
  6'b001100: outs=10'b0_0_0_010_1_0_1_0;
  6'b001101: outs=10'b0_0_0_011_1_0_1_0;
  6'b001110: outs=10'b0_0_0_100_1_0_1_0;
  6'b001000: outs=10'b0_0_0_000_1_0_1_1;
  6'b001001: outs=10'b0_0_0_000_1_0_1_0;
  6'b001010: outs=10'b0_0_0_110_1_0_1_1;
  6'b001011: outs=10'b0_0_0_111_1_0_1_0;

  6'b000000:
            casex(funct)
              12'b000000100000: outs=10'b0_0_0_000_0_1_1_0;
              12'b000000100001: outs=10'b0_0_0_000_0_1_1_0;
              12'b000000100010: outs=10'b0_0_0_001_0_1_1_0;
              12'b000000100011: outs=10'b0_0_0_001_0_1_1_0;
              12'b000000100100: outs=10'b0_0_0_010_0_1_1_0;
              12'b000000100101: outs=10'b0_0_0_011_0_1_1_0;
              12'b000000100110: outs=10'b0_0_0_100_0_1_1_0;
              12'b000000100111: outs=10'b0_0_0_101_0_1_1_0;
              12'b000000101010: outs=10'b0_0_0_110_0_1_1_0;
              12'b000000101011: outs=10'b0_0_0_111_0_1_1_0;
```

بخش دوم کد :....

```
                default:outs=10'b0_0_0_000_0_0_0_0;
                endcase
    default:outs=10'b0_0_0_000_0_0_0_0;
endcase
end

// extracting Data from register

assign MemtoReg = outs[9];

assign MemWrite = outs[8];

assign PCSrc = outs[7];

assign ALUOP = outs[6:4];

assign ALUSrc = outs[3];

assign RegDst = outs[2];

assign RegWrite = outs[1];

assign SgnZero = outs[0];

endmodule
```

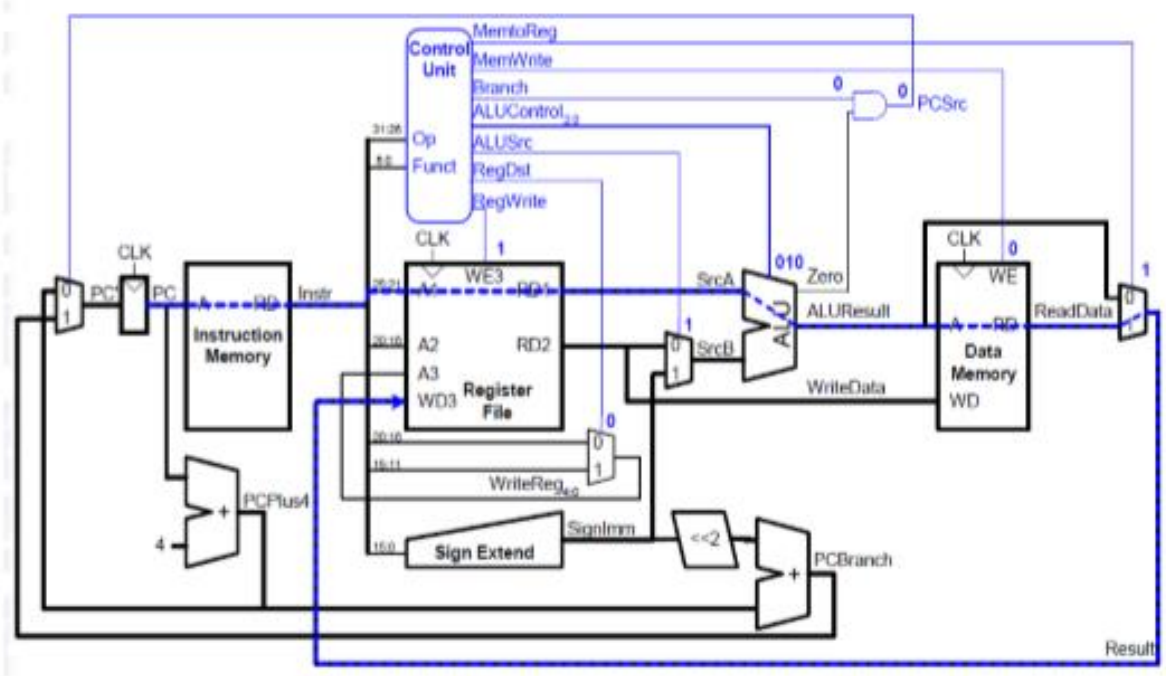
توجه کنید که اینکه آپکد و فانکشن کد چه خروجی هایی تعیین می کنند با جست و جو و بررسی در اینترنت به دست آمد.

ماژول DATA Path

برای طراحی این ماژول داریم که یک سری ماژول و mux و ... تعریف می کنیم.

ماژول های تعریفی الحاقی :

1- Sign extender
2- Shifter
3- ...



- نکته : بخش and شده ی zero و branch درونا داخل ماژول control Unit رخ میده و داریم که انگار ما این گیت را داخل Control unit می بریم و به جای branch پایانه ی PCSrc بیرون آوردیم ...

کد ماژول های زده شده :

```
module SignExtend(
input [15:0] gonnaSignExtend,
input sgnZero,
output [31:0] signExtended
    );

assign signExtended = (sgnZero == 1'b1 ) ? {signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            signExtended[15],
                                            gonnaSignExtend}
                                          : {16'b0000_0000_0000_0000,gonnaSignExtend};

endmodule


module Shifter(
input [31:0] in,
output [31:0] out
    );

assign out = in * 4;

endmodule
```

پس از ران کردن تست بنچ داریم که خروجی تماما صحیح و مطابق با خروجی correct Answer عه که همراه کد ارایه شده است.

Correct Output:

```
# ffff8a4f ff49a03e ed9232cf ed9232cf ec6c3298 ec6c3298 e6663185 e6663185 dddd8526 dbdb842d b6b6e9be b4b4e947 aac70a4f aaaaec60 a48e7be0 a48e7be0
# 9c7a4305 9c7a4305 99bd6c60 8bd654a6 8bd654a6 8894b185 878c0526 86b259c7 86b259c7 82846947 826e5d18 826e5d18 81da5ead 81da5ead 8183b298 8081042d
# 807f69be 802ab2cf 8019fbe0 8008c305 8007d4a6 8002d9c7 8001dd18 8000dead 8000203e 8000203e 7fff8a4f 7fff8a4f 7f49a03e 7f49a03e 6d9232cf 6c6c3298
# 66663185 5ddd8526 5ddd8526 5bdb842d 5bdb842d 36b6e9be 36b6e9be 34b4e947 34b4e947 2ac70a4f 2ac70a4f 2aaaec60 2aaaec60 248e7be0 1c7a4305 19bd6c60
# 19bd6c60 0bd654a6 0894b185 0894b185 078c0526 078c0526 06b259c7 02846947 02846947 026e5d18 01da5ead 0183b298 0183b298 0081042d 0081042d 007f69be
# 007f69be 002ab2cf 002ab2cf 0019fbe0 0019fbe0 0008c305 0008c305 0007d4a6 0007d4a6 0002d9c7 0002d9c7 0001dd18 0001dd18 0000dead 0000dead 0000203e
```

خروجی من : یکی اند !!!

```
ffff8a4f ff49a03e ed9232cf ed9232cf ec6c3298 ec6c3298 e6663185 e6663185 dddd8526 dbdb842d b6b6e9be b4b4e947 aac70a4f aaaaec60 a48e7be0 a48e7be0
9c7a4305 9c7a4305 99bd6c60 8bd654a6 8bd654a6 8894b185 878c0526 86b259c7 86b259c7 82846947 826e5d18 826e5d18 81da5ead 81da5ead 8183b298 8081042d
807f69be 802ab2cf 8019fbe0 8008c305 8007d4a6 8002d9c7 8001dd18 8000dead 8000203e 8000203e 7fff8a4f 7fff8a4f 7f49a03e 7f49a03e 6d9232cf 6c6c3298
66663185 5ddd8526 5ddd8526 5bdb842d 5bdb842d 36b6e9be 36b6e9be 34b4e947 34b4e947 2ac70a4f 2ac70a4f 2aaaec60 2aaaec60 248e7be0 1c7a4305 19bd6c60
19bd6c60 0bd654a6 0894b185 0894b185 078c0526 078c0526 06b259c7 02846947 02846947 026e5d18 01da5ead 0183b298 0183b298 0081042d 0081042d 007f69be
007f69be 002ab2cf 002ab2cf 0019fbe0 0019fbe0 0008c305 0008c305 0007d4a6 0007d4a6 0002d9c7 0002d9c7 0001dd18 0001dd18 0000dead 0000dead 0000203e
** Note: $stop    : C:/Users/user/Desktop/processor/SingleCycle__tb.v(69)
   Time: 258350 ns  Iteration: 2  Instance: /SingleCycle__tb
Break in Module SingleCycle__tb at C:/Users/user/Desktop/processor/SingleCycle__tb.v line 69
```

خروجی کد در اثر تست بنچ داده شده :