Control Unit:

برای طراحی این بخش ابتدا تمام سیگنال های کنترلی مورد نیاز برای بخش های مختلف cpu مان را بدست می آوریم.

سیگنال های مورد نیاز عبارتند از:

1) MemRead :

با یک شدن این سیگنال مقداری که در خانه با آدرس ورودی قرار دارد از حافظه دستور خوانده می شود.

2) RegWrite :

با یک شدن این سیگنال مقدار موجود در ثباتی که بیت های 10 تا 14 (ReadReg1)و نیز ثباتی که بیت های 5 تا 9 (ReadReg2) نشان می دهند از رجیستر فایل خوانده می شوند و به ترتیب در ReadData1 و ReadData2 قرار می گیرد.

3) CinSrc :

این سیگنال مربوط به سیگنال کنترلی مالتی پلکسری است که در مسیر ورودی Cin ، ALU قرار دارد که در صورت 1 شدن بیت Cin از دستور وارد بخش Cin مالتی پلکسر می شود و در صورت 0 شدن مقدار 0 در Cin در ALU قرار می گیرد.

4) AluSrcA :

سیگنال کنترلی ماکسی است که در ورودی اول ALU قرار دارد که در صورت 1 شدن آن مقدار ثبات A که همان ReadData1 است وارد ALU می شود و در صورت 0 شدن مقدار PC در آن قرار می گیرد.

5) AluSrcB :

سیگنال کنترلی ماکسی است که در ورودی دوم ALU قرار دارد و در صورت 1 شدن مقدار 4 در ALU قرار میگیرد( برای جمع شدن مقدار فعلی PC با 4 و تولید مقدار PC جدید ) و در صورت 0 شدن مقدار ثبات Bکه همان ReadData2 است وارد ALU می شود.

6) AluOp :

این سیگنال نیز 3 بیتی است و به ALU می رود تا نوع عملی را که قرار است بر روی عملوند ها قرار بگیرد را تعیین کند و چون 7 نوع عمل در ISA ما موجود است پس 7 عمل نیاز به 3 بیت برای تفکیک دارند.

این پردازنده که طراحی آن به صورت مالتی سایکل است هر دستور آن برای اجرا به 4 کلاک نیاز دارد:

کلاک اول(IF):

این کلاک در تمامی دستورات مشترک است و 2 عمل در این بخش انجام میگیرند :

1) InstructionBus(IB) <= Mem[PC]

2) PC <= PC + 4

کلاک دوم (ID & RR):

این کلاک نیز در تمامی دستورات مشترک است و 3 عمل در این بخش انجام میگیرد:

1) A <= GPR[IB[14-10]

2) B <= GPR[IB[9-5]

3) AluOut <= 22{0} + IB[14-5]

کلاک سوم:

در این کلاک با توجه به نوع دستورات عملیات خاصی انجام میگیرد:

1) دستوراتی که به in1 و in2 نیاز دارند:

AluOut <= in1 op in2

2)دستوراتی که به in1 و shamt نیاز دارند:

AluOut <= in1 (>> or <<) shamt

3)دستور ذخیره مقدار shamt در بانک ثبات :

GPR[IB[4-0]] <= AluOut

این نوع از دستورات در کلاک سوم تمام می شوند.

کلاک چهارم:

این کلاک نیز در دستورات بخش 1 و 2 کلاک سوم مشترک است و عمل زیر انجام میگیرد:

GPR[IB[4-0]] <= AluOut

شکل FSM آن به صورت زیر خواهد بود و چون 5 حالت موجود است پس به 3 بیت برای state نیاز خواهیم داشت و در نهایت یک ROM طراحی میکنیم که opcode و state را ورودی میگیرد (4+3 = 7 )و سیگنال های کنترلی و next state را خروجی می دهد (8+3 = 11).

این ROM 7 بیت ورودی و 11 بیت خروجی دارد که حجم آن برابر است با:

11 \* 27 = 1.375 KB



