به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

ESL

گزارش پروژه چهارم

نام و نام خانوادگی	علی مومنی
شماره دانشجویی	810100215
تاریخ ارسال گزارش	

فهرست

توضیحات

برای انجام این پروژه نیز برای سهولت کار از repository خود chisel که در پروژه قبل استفاده شده بود استفاده می‌کنیم و تمامی فایل ها را در همان محل جایگذاری میکنیم. پوشه اصلی پروژه در محل chisel-tutorial\src\main\scala\CA4 ایجاد شده است. تمامی فایل هایی استفاده شده در مسیر داده نیز در این پوشه قرار داده شده اند بجز ALU که در محل پروژه قبلی بود و برای استفاده از آن، آن را در هرجا که نیاز بود با دستور روبه‌رو import می‌کنیم: import solutions.ALU

پس از نوشتن ماژول های مورد نیاز بصورت جداگانه، مسیرداده را با اینتستنس گرفتن از ماژول ها ایجاد میکنیم. برای اینکار ابتدا سیگنال های ورودی خروجی را متصل کرده و سپس بقیه مراحل را انجام می‌دهیم:

```
class DatapathIO extends Bundle {
  val sel_RegW = Input(Bool())
  val mem_WE = Input(Bool())
  val SrcBSel = Input(UInt(2.W))
  val AReg_out = Output(UInt(16.W))
  val sel_PC = Input(Bool())
  val RegWe = Input(Bool())
  val sel_PCAddr = Input(Bool())
  val AReg_en = Input(Bool())
  val ALUOp= Input(UInt(3.W))
  val Rom_inst_out = Output(UInt(8.W))
}
```

سپس ماژول کنترلر را با توجه به جدول داده شده در صورت سوال به کمک دستور switch/case مطابق چیزی که در پروژه سوم و ALU استفاده شده طراحی می‌کنیم:

```
package CA4

import chisel3._
import chisel3.util._

class Controller extends Module {
  val io = IO(Flipped(new DatapathIO))

  io.sel_PC := true.B
  io.sel_RegW := true.B
  io.sel_PCAddr := true.B
  io.SrcBSel := true.B
  io.mem_WE := false.B
  io.AReg_en := false.B
  io.ALUOp := 0.U
  io.RegWe := false.B

  val inst = io.Rom_inst_out
  val opcode = inst(7, 2)
  val imm = inst(0)

  switch(inst) {
    is("b00001000".U) { // Add
      io.ALUOp := Mux(imm === 0.U, 1.U, 1.U)
      io.SrcBSel := Mux(imm === 0.U, 2.U, 1.U)
    }
    is("b00001001".U) { // Addi
      io.ALUOp := Mux(imm === 0.U, 1.U, 1.U)
      io.SrcBSel := Mux(imm === 0.U, 1.U, 0.U) // Adjust SrcBSel for Addi
    }
    is("b00001100".U) { // Sub
      io.ALUOp := Mux(imm === 0.U, 2.U, 2.U)
      io.SrcBSel := Mux(imm === 0.U, 2.U, 1.U)
    }
```

در نهایت ماژول های مسیرداده و کنترلر را در top module که LoresProcessor16.scala نام دارد اینتنس میگیریم:

```
package CA4

import chisel3._
import chisel3.util._
import chisel3.iotesters.{PeekPokeTester, Drive

class LerosProcessor16 extends Module
{
  val io = IO(new Bundle {})
    val datapath = Module(new Datapath)
    val controller = Module(new Controller)

    controller.io <> datapath.io
}
```

برای تست کردن این پردازنده مطابق خواسته صورت پروژه ۵ عدد را در ممووری گذاشته و در نهایت آن ها را به کمک instruction های مورد نیاز درون رجیسترفایل های 0 تا 4 ریخته و در رجیسترفایل با ایندکس 5، مجموع آن ۵ عدد را محاسبه می‌کنیم. اعداد استفاده شده در این تست به شرح ذیل است:

حالت اول) 5, 7, 3, 42, 15

```
val defaultValues = VecInit(Seq(5.U, 7.U, 3.U, 42.U, 15.U))
```

حالت دوم) 1, 2, 3, 4, 5

```
val defaultValues = VecInit(Seq(1.U, 2.U, 3.U, 4.U, 5.U))
```

همچنین instruction های مورد نیاز برای این عمل به شرح ذیل می‌باشد:

```
rom(0) := "b00100000_00000000".U
rom(1) := "b01010000_0000000".U
rom(2) := "b01100000_00000000".U
rom(3) := "b00110000_00000000".U
rom(4) := "b01100000_00000001".U
rom(5) := "b00110000_00000001".U
rom(6) := "b01100000_00000010".U
rom(7) := "b00110000_00000010".U
rom(8) := "b01100000_00000011".U
rom(9) := "b00110000_00000011".U
rom(10) := "b01100000_00000100".U
rom(11) := "b00110000_00000100".U
rom(12) := "b00100000_00000000".U
rom(13) := "b00001000_00000001".U
rom(14) := "b00001000_00000010".U
rom(15) := "b00001000_00000011".U
rom(16) := "b00001000_00000100".U
rom(17) := "b00001000_00000101".U
rom(18) := "b00110000_00000101".U
```

همانطور که مشاهده می‌شود این کار توسط ۱۹ دستور در این پردازنده قابل انجام می‌باشد.

در حالت اول انتظار میرود خروجی در رجیستر با اندیس ۵ قرار بگیرد و مقدارش برابر با ۵ + ۷ + ۳ + ۴۲ + ۱۵ که برابر با ۷۲ است دیده شود. جهت دیده شدن خروجی قطعه کدی در رجیسترفایل نوشته شده که تنها اگر مقدار رجیستر با اندیس ۵ مخالف صفر بود (یعنی مقداری را گرفته بود)، تمامی مقادیر ۶ رجیستر نمایش داده شود (۵ رجیستر حاوی اعداد و رجیستر آخر حاوی حاصل جمع اعداد).

```
// Testing
when (registerFile(5) =/= 0.U) {
  for (i <- 0 until numRegisters) {
    printf(p"Regfile($i): ${registerFile(i.U)}\n")
  }
printf(p"\n")
}
```

برای اجرا کردن این پروژه تنها نیاز است دستور sbt run را در محل پروژه (chisel-tutorial) اجرا کنیم.

درستی سنجی طراحی

نتیجه تست حالت اول:

```
PS D:\UT\Semester 6\ESL\CAs\CA4\chisel-tutorial> sbt run
[info] Loading project definition from D:\UT\Semester 6\ESL\CAs\CA4\chisel-tutorial\project
[info] Loading settings for project chisel-tutorial from build.sbt ...
[info] Set current project to chisel-tutorial (in build file:/D:/UT/Semester%206/ESL/CAs/CA4/chisel-tutorial/)
[info] Compiling 1 Scala source to D:\UT\Semester 6\ESL\CAs\CA4\chisel-tutorial\target\scala-2.12\classes ...
[info] running CA4.LerosProcessor16
[info] [0.000] Elaborating design...
[info] [0.110] Done elaborating.
Computed transform order in: 138.5 ms
Total FIRRTL Compile Time: 559.7 ms
End of dependency graph
Circuit state created
[info] [0.000] SEED 1718557874909
Regfile(0): 56050
Regfile(1): 58054
Regfile(2): 22334
Regfile(3): 18044
Regfile(4): 23343
Regfile(5): 11158

Regfile(0): 5
Regfile(1): 7
Regfile(2): 3
Regfile(3): 42
Regfile(4): 15
Regfile(5): 72

test LerosProcessor16 Success: 0 tests passed in 26 cycles taking 0.051506 seconds
[info] [0.030] RAN 21 CYCLES PASSED
[success] Total time: 5 s, completed Jun 16, 2024 8:41:17 PM
```

علت اینکه بخش قرمز رنگ نمایش داده شده است این است که در لحظه اول تمامی رجیستر ها به دلیل عدم initialization مقداری غیرصفر بصورت رندوم دارند و چون مقدار ریجستر با اندیس۵ مخالف صفر است مقادیر تمام ریجستر ها نمایش داده شده است. بخش آبی رنگ بخش درست و نتیجه اصلی می‌باشد که مطابق انتظار، رجیسترها مقادیر یه آرایه ۵ تایی را به ترتیب گرفته و در خانه آخر مجموع آن ها نوشته شده است که برابر با ۷۲ می‌باشد.

نتیجه تست حالت دوم:

```
PS D:\UT\Semester 6\ESL\CAs\CA4\chisel-tutorial> sbt run
[info] Loading project definition from D:\UT\Semester 6\ESL\CAs\CA4\chisel-tutorial\project
[info] Loading settings for project chisel-tutorial from build.sbt ...
[info] Set current project to chisel-tutorial (in build file:/D:/UT/Semester%206/ESL/CAs/CA4/chisel-tutorial/)
[info] Compiling 1 Scala source to D:\UT\Semester 6\ESL\CAs\CA4\chisel-tutorial\target\scala-2.12\classes ...
[info] running CA4.LerosProcessor16
[info] [0.000] Elaborating design...
[info] [0.141] Done elaborating.
Computed transform order in: 171.1 ms
Total FIRRTL Compile Time: 619.4 ms
End of dependency graph
Circuit state created
[info] [0.000] SEED 1718558145856
Regfile(0): 8413
Regfile(1): 11536
Regfile(2): 42306
Regfile(3): 36782
Regfile(4): 61306
Regfile(5): 7849

Regfile(0): 1
Regfile(1): 2
Regfile(2): 3
Regfile(3): 4
Regfile(4): 5
Regfile(5): 15

test LerosProcessor16 Success: 0 tests passed in 26 cycles taking 0.077148 seconds
[info] [0.050] RAN 21 CYCLES PASSED
[success] Total time: 5 s, completed Jun 16, 2024 8:45:48 PM
```

در این حالت نیز انتظار میرفت که خروجی برابر با ۱ + ۲ + ۳ + ۴ + ۵ یا برابر با ۱۵ باشد که مطابق شکل این مقدار در رجیستر آخر ذخیره شده است.