Single cycle processor: Each instruction requires only 1 cycle but each cycle has a longer period. Even complex instructions can be executed in that 1 cycle so its period is long. Means that your processor must be slower.

For now, assume that we execute only 1 instruction in 1 cycle.

The Big Picture

The five classic components of a computer:

Diagram

Description automatically generated

Control unit is mostly made up of finite state machines. There are lots of complexities to design the control unit. However, in our single cycle processor, control unit will be made up of only combinational circuit. We will not need any memory element inside our control unit or we will not have any state because single cycle processor is simple.

For now, we want to design the datapath of single cycle processor.

The Big Picture: The Performance Perspective

Diagram, shape

Description automatically generatedPerformance of a machine is determined by:

* instruction count
* clock cycle time
* clock cycles per instruction

Processor design (datapath and control) will determine:

* clock cycle time
* clock cycles per instruction

Single cycle processor – one clock cycle per instruction

* Advantages: simple design, low CPI (fixed to 1)
* Disadvantages: long cycle time, which is limited by the slowest instruction

How to Design a Processor: step-by-step

Analyze instruction set 🡪 datapath requirements

* the meaning of each instruction is given by register transfers:
  + R[rd] <- R[rs] + R[rt]; -----> RTL representation for add instruction
* datapath must include storage element for ISA registers
* datapath must support each register transfer

Select set of datapath components and establish clocking methodology

Design datapath to meet the requirements

* Designing datapath means to meet requirements of each instruction by inserting them one by one

Analyze implementation of each instruction to determine setting of control points that effects the register transfer

Design the control logic

Review: MIPS Instruction Formats

Table

Description automatically generated

**STEP 1A: THE MIPS SUBSET FOR TODAY**

Timeline

Description automatically generated

Register Transfer Logic (RTL)

RTL gives the meaning of the instructions

All instructions start by fetching the instruction

Text, letter

Description automatically generated

Step 1: Requirements of the Instruction Set

2 distinct memories:

* Instruction & data
  + 2 distinct in order to read instruction and data at the same time (1 cycle)
  + Harward Archtitecture

Registers (32 x 32)

* read rs
* read rt
* write rt or rd

PC

Sign extender

Add and sub register or extended immediate

Add 4 and/or shifted extended immediate to PC

**STEP 2: COMPONENTS OF THE DATAPATH**

Diagram, schematic

Description automatically generated

Diagram

Description automatically generatedStorage Element: Register (Basic Building Blocks)

Register

* Similar to the D Flip Flop except
  + N-bit input and output
  + Write enable input
* Write Enable:
  + negated (0): Data Out will not change
  + asserted (1): Data Out will become Data In on the falling edge of the clock

Üçgen, içeriğin sadece edge’de değişebileceğini gösterir.  
Clk’ta baloncuk olması negative edge triggered anlamına gelir. Yani register content can only change in negative edges:

Diagram

Description automatically generated

We can read data anytime.

Yazma sadece edgelerde yapılır. Negative edge triggered olduğu için bunda negative edgelerde yapılıyor.

Diagram, schematic

Description automatically generatedStorage Element: Register File

Register File consists of 32 registers:

* Two 32-bit output busses: busA and busB
* One 32-bit input bus: busW

Register is selected by:

* RA (number) selects the register to put on busA (data)
* RB (number) selects the register to put on busB (data)
* RW (number) selects the register to be written via busW (data) when Write Enable is 1

Clock input (CLK)

* The CLK input is a factor ONLY during write operation
* During read operation, behaves as a combinational logic block:
  + RA or RB valid 🡪 busA or busB valid after “access time”

Read from Register File

Diagram, schematic

Description automatically generated



32 32-bit register olduğunu düşünelim yani n=32.

2 tane 32x1 MUX

Paralel olarak 2 data okuyabilmek için 2 mux var. Select inputlar 5 bit.

Write to Register File

Diagram

Description automatically generated

Diagram

Description automatically generatedStorage Element: Memory

Memory

* One input bus: Data in
* One output bus: Data out



Memory word is selected by:

* Address selects the word to put on Data Out
* Write Enable = 1: address selects the memory word to be written via the Data In Bus

Clock input (CLK)

* The CLK input is a factor ONLY during write operation
* During read operation, memory behaves as a combinational logic block:
  + Address valid 🡪 Data Out valid after “access time”

230 words = 4 GB

We don’t need to write and read in parallel in our memory. If instruction is lw, we read; if sw, we write.

For register, at the same cycle, for example we have to read 2 register contents and write the result to another register. We have to perform reading 2 register contents and writing to register in parallel.

**STEP 3**

Register Transfer Requirements 🡪 Datapath Design

* Instruction Fetch (Reading instruction from memory)
* Decode instructions and Read Operands
* Execute Operation
* Write back the result

3a: Overview of the Instruction Fetch Unit

Diagram

Description automatically generatedThe common RTL operations

* Fetch the Instruction: mem[PC]
* Update the program counter:
  + Sequential Code: PC 🡨 PC + 4
  + Branch and Jump: PC 🡨 “something else”

PC has instruction address.

Next Address Logic computes the next PC value. Usually PC+4 other than jump instructions.

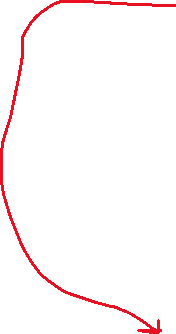
3b: R-Type Instructions

R[rd] 🡨 R[rs] op R[rt] Example: add rd, rs, rt

* Ra, Rb, and Rw come from instruction’s rs, rt, and rd fields
* ALUctr and RegWr: control logic after decoding the instruction

Diagram

Description automatically generated



3c: Load Operations

Diagram, schematic

Description automatically generated



Sign extender:

Diagram

Description automatically generated

imm16[15] is MSB of 16 bit number.

Yaptığımız concatenation (bus’ı büyütüyoruz). Normalde 2 kablo bir araya gelemez.

ALUctr, ALUSrc, RegWr… 🡪 bunlar hep control unitin outputları.

MemtoReg is 0 for R-type instructions. It directly puts output of ALU to busW.  
MemtoReg is 1 for lw.

Diagram, schematic

Description automatically generated3d: Store Operations

sw geldiğinde RegWr 0 olmalı ki busW’deki alakasız bir data, alakasız bir register adresine yazılmasın.

3e: The Branch Instruction

Graphical user interface, application

Description automatically generated

**Datapath for Branch Operations:**

Diagram, schematic

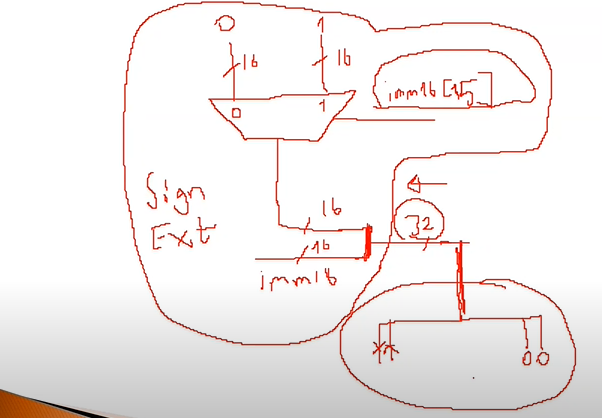
Description automatically generated

beq’da immediate field’da kaç instruction ileri ya da geri gideceğini taşıyoruz.

ALU’muz rs ile rt contentleri eşit mi (rs-rt =? 0) diye baktığı için adderlar eklemek zorunda kaldık.

Branch input comes from the control unit. It becomes 1 if instruction is beq.

PC extender 🡪 sign extend the 16 bit immediate number and multiplies it by 4 (shift left).





**PUTTING IT ALL TOGETHER: A SINGLE CYCLE DATAPATH**

Diagram, schematic

Description automatically generated

Diagram, schematic

Description automatically generatedDifferent View:

Control’deki RegDst oku ters bakıyor.

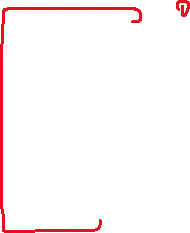
Control’ün tek inputu OPCODE. R typelarda ALU Control’e function field gelir. Tek fark ALU’nun yapacağı işlem değişebilir. Onun için ALU control’e function field gelir.

We can insert new instructions to our datapath.

For example:

* Diagram, schematic

  Description automatically generatedaddi $rt, $rs, imm16

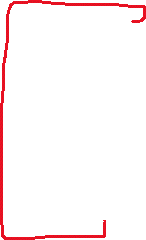


lw $rt, imm16($rs) <-------> R[rt] 🡨 M[$rs + SignExt(imm)]

* rs ve rt’nin okunması gerekir. Bunlar her cycle’da otomatik olarak okunuyor.
* rs içeriğiyle signextendimmediate’ı topla, bir adres bulacaksın. Bu adres ile memory’ye git. Memory sana content verecek, bu contenti rt’ye koy.

Diagram, schematic

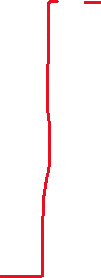
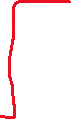
Description automatically generated



beq

Diagram, schematic

Description automatically generated



ALU çıkarma yapacak. Eğer zero biti 1 olursa eşit.

bne

beq’ten tek farkı bne geldiğinde eğer çıkarma sonrası ALU’da zero bit 0’sa, yani eşit değillerse branch edecek.

Diagram

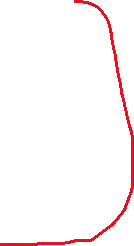
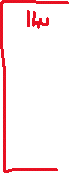
Description automatically generated

lhu $rt, imm($rs)

load halfword unsigned 🡪 memory’den çıkan sayının least sig. 16 bitini alır ve başına 16 tane 0 koyup rt registerına yükler.

Diagram, schematic

Description automatically generated



jal L

L labelındaki adrese atlıyor ve kendisinden bir sonraki instructionın adresini (PC+4) 31 nolu registerda saklıyor.

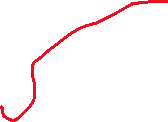
J instructionıdır. 26 bitlik adresin soluna PC’ın most significant 4 biti (PC[31:28]), sağına 2 sıfır eklenir.

Diagram, schematic

Description automatically generated

Diagram

Description automatically generated



**STEP 4: GIVEN DATAPATH RTL 🡪 CONTROL**

Diagram, timeline

Description automatically generated

Bizde control unit ikiye ayrılmış:

* Birisi sadece opcode’a bakıyor
* Birisi ALU Control, function field’a bakıyor

Diagram, schematic

Description automatically generatedMeaning of the Control Signals

SUMMARY

5 steps to design a processor

1. Analyze instruction set 🡪 datapath requirements
2. Select set of datapath components & establish clock methodology
3. Design datapath meeting the requirements
4. Analyze implementation of each instruction to determine setting of control points that effects the register transfer.
5. Design the control logic

MIPS makes it easier

* Instructions same size
* Source registers always in same place
* Immediates same size (16-bit), location
* Operations always on registers/immediates

Singe cycle datapath ---> CPI = 1 , CCT = long

Her şey 1 cycle içerisinde gerçekleştiği için her blok tek bir şey ile uğraşabilir. Örneğin ALU rs ile signextendimmediate’ı topluyorsa bu instruction için aynı anda başka bir iş yapamaz.

Multicyclelarda bir instruction birkaç cycle sürebilir. Instructionın ilk cycleında rs ve rt registerlarını okurken, bir yandan da PC+4’ü ALU’ya yaptırabilirdik. Dolayısıyla sol üstteki addera ihtiyaç olmazdı. Yani donanım azalır.

Ayrıca multicyclelarda her instruction en uzun instruction kadar zaman almak zorunda kalmazdı.

Multicycleın dezavantajı control unitin tasarımını zorlaştırmasıdır. Control unit finite state machine’e dönüşür.

Pipelining multicycledan daha iyi bir çözüm. Pipeliningin yapısı single cycle processore daha benzer.