Other Reduced Instruction Set Computers (RISC) and Complex Instruction Set Computers (CISC)

Adam Boult (www.bou.lt)

February 11, 2024

Contents

Part I

Other Reduced Instruction Set Computers (RISC)

Chapter 1

Reduced Instruction Set Computer (RISC)

1.1 Introduction

1.1.1 Introduction

Chapter 2

MIX and MMIX

2.1 Introduction

2.1.1 Introduction

Chapter 3

ARM

3.1 Introduction

3.1.1 Introduction

Chapter 4

PowerPC

4.1 Introduction

4.1.1 Introduction

li r3, 4 (move 4 into register 3)

r0 is always zero

r1 is stack pointer

r3 is return value

4.1.2 Arithmetic

add r1, r2, r3 (add r2 and r3 and put in r1)

addi r1, r2, 3 (add r2 and immediate value of 3 and put in r1)

can do "move" by adding zero to something and putting it in destination

4.1.3 Branching

b label (branch to label)

cmp 0, 0, r1, r2 (compare two values into condition register 0)

blt label (go to branch if previous comparison was less than) ble beq bge bgt bne

4.1.4 RAM

accessing RAM stw r1, 8(r2) (store r1 in location indicated by r2 offset by 8) lwz r1, 8(r2) (replace r1 with data in location indicated by r2 offset by 8)

4.1.5 Functions

functions: bl label (branch and link - saves return address in link register) blr (branch to link register - ending a function)

4.1.6 Floating point

floating point f0, f1, etc are floating point registers lfs (load float single) lfd stfs stfd (store float dobule)

fadd (floating add) fadds (floating add single precision) fmul fmadd fdiv

Chapter 5

Microprocessor without Interlocked Pipelined Stages (MIPS)

5.1 Introduction

5.1.1 Introduction

registers

```
access with $t0 etc

li load immediate
li $t1, 8 (loads 8 into t1)

add $t0, $t0, $t1 (add t1 and t0 and put in t0)

NA: move, li, la and others and pseudo instructions
```

Chapter 6

The Java Virtual Machine (JVM)

6.1 Introduction

6.1.1 Introduction

JVM has stack and registers.

Chapter 7

Web Assembly (wasm)

7.1 Introduction

7.1.1 Introduction

web assembly is load store (RISC?) stack. has registers too?

Part II

Complex Instruction Set Computers (CISC)

Chapter 8

Complex Instruction Set Computer (CISC)

8.1 Introduction

8.1.1 Introduction

Chapter 9

IBM System/360

9.1 Introduction

9.1.1 Introduction

Chapter 10

Motorola 6800 and Motorola 6502

10.1 Motorola 6800

10.1.1 Introduction

10.2 Motorola 6502

10.2.1 Introduction

```
general convention:
+ %00000001 for binary
+ $FA for eg hex of 8 bit (single byte requires 2&8=8*8)
+ 123 for decimal

Registers:
+ A (accumulator)
+ X
+ Y
+ PC (program counter)
+ S (stack pointer)
+ P (status)

status flags:
+ N (negative)
+ V (overflow)
+ B (break)
+ D (decimal)
+ I (interrupt disable)
```

```
+ Z (zero)
+ C (carry)

instructions
+ LDA (load accumulator with memory)
  * 16  bit address space, so $0000 format
  * LDA $1002 (load value at memory $1002 into accumulator)
  * LDA $02 (load value at memory $0002 into accumulator) (defaults to 00 if missing - zero
  * LDA #$02 (load hex $02 into accumulator)
+ LDX (load X with memory)
+ LDY (load Y with memory)
+ STA (store accumulator in memory)
+ STX (store X in memory)
+ STY (store y in memory)

arithmetic:
+ ADC (add memory to accumulator with carry)
+ SBC (substract memory from accumulator with borrow)
+ INC (increment memory)
+ INX (incrememt index X)
+ INY (increment index Y)
+ DEC (decrement memory)
+ DEX (decrement index X)
+ DEY (decrement index Y)

+ ASL (arithmetic shift left)
+ ASR (arithmetic shift right)
+ ROL (rotate left)
+ ROR (rotate right)
+ AND (and memory with accumulator)
+ OR (or memory with accumulator)
+ EOR (exclusive or memory with accumulator)
```

Chapter 11

Motorola 68000

11.1 Introduction

11.1.1 Introduction

Chapter 12

Intel 8080 and Zilog Z80

12.1 Intel 8080

12.1.1 Introduction

12.2 Zilog Z80

12.2.1 Introduction

Based on Intel 8080.

Different mnemonics, but binary compatible?

ld a, 5

add a, b + adds b to accumulator a

sub adc sbc

use of memory: ld [Output], a (writes accumulatoro to memory)

jumps: jp CC, Target jr CC, Target

stack push af (pushes accumulator and flag (because together are 16 bit) pop

functions

12.2.2 Gameboy

gameboy based on this, ppu separate.

Chapter 13

Intel 8086 (x86)

13.1 Introduction

13.1.1 Introduction

13.1.2 x86 accumulators

standard register names (accumulators)

ax (accumulator, 16 bit?)

al is the lower 8 bit part of ax

ah is the higher 8 bit part of ax

eax (Extended accumulator. 32 bit?)

rax (64 bit accumulator)

13.1.3 mov instruction

mov eax, 1 (move the value 1 into the register eax

13.1.4 Other registers

register names for bases + bx (16 bit) * bl * bh + ebx (32 bit)

count register + cx (16 bit) * cl * ch + ecx (32 bit)

data registers + dx (16 bit) * dl * dh + edx (32 bit)

13.1.5 x86 extension: MMX

13.1.6 x86 extension: SSE

13.1.7 x86 extension: AVX