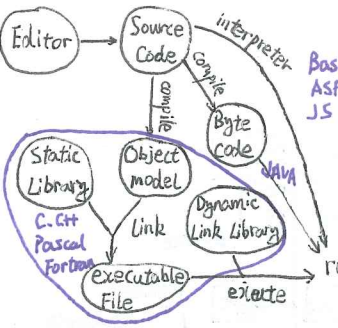
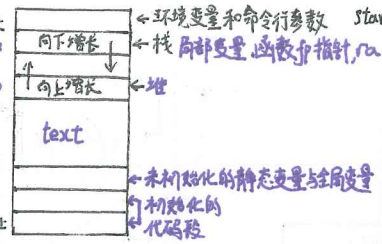
计算机组成

CHAP 01 Introduction





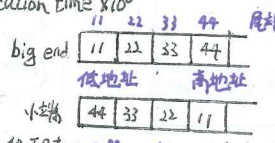
CPU time = Instructions * CPI * Clock cycle time
= Instructions * CPI / Clock rate
= Instructions / MIPS (Millions of Instructions Per Second)

CPI = CPU clock cycles for a program / Instruction Count

"Processor X is n times faster than Y" MIPS = Instruction count / Execution time x10^6

n = Execution time_Y / Execution time_X

speed up = Execution time without enhancement / Execution time with enhancement



移码表示 (offset binary) 相当于补码表示符号位取反

原码: 11111000 ← (-120)10
反码: 10000111
补码: 10001000
移码: 00001000

single precision.
31 | 30-23 | 22-0
sign | exponent | fraction
bias (-1)^sign * (1+Fraction) * 2^exp-bias

double precision
31 | 30-20 | 19-0 | 32-0
sign | exponent | fraction | fraction

Chap 2 MIPS instructions

R-type / R-format

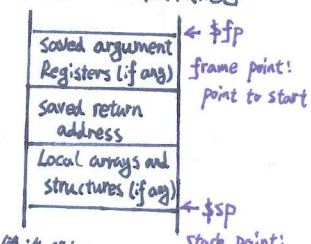
op	rs	rt	rd	shamt	funct
6	5	5	5	5	6 (bits)

I-type / I-format

op	rs	rt	address
6	5	5	16 (bits)

op: basic operation of instruction
rs: first register
rt: second register
rd: register destination
shamt: shift amount
funct: function (add, sub, and, or)

procedure call 时的栈帧



signed multiplication

booth 算法例:

假设: addition 与 subtraction 可用

booth 算法 (核心: 与乘数相乘化为 shift)

① 当乘数中某个1与其他1不连续时, 将其拆出

② 当乘数中有连续的1时, 将连续1拆出, 末位加1减1

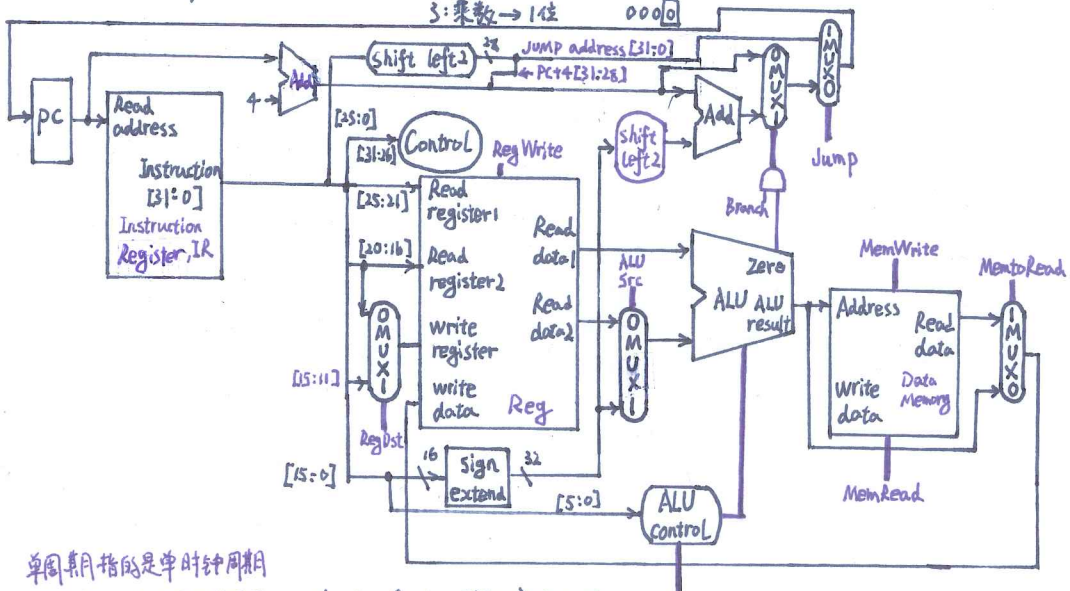
Z = y x 10 111100
= y x (10 000000 + 111100 + 100 - 100)
① ②
= y x (1x2^7 + 1x2^6 - 1x2^2)
然后将y左移7位/左移6位/左移2位的结果加减即可

分析连续两位 (看积的最后2位, 操作完成后右移积)

Action	
10	product 减去 multiplicand
11	不动
01	product 加上 multiplicand
00	不动

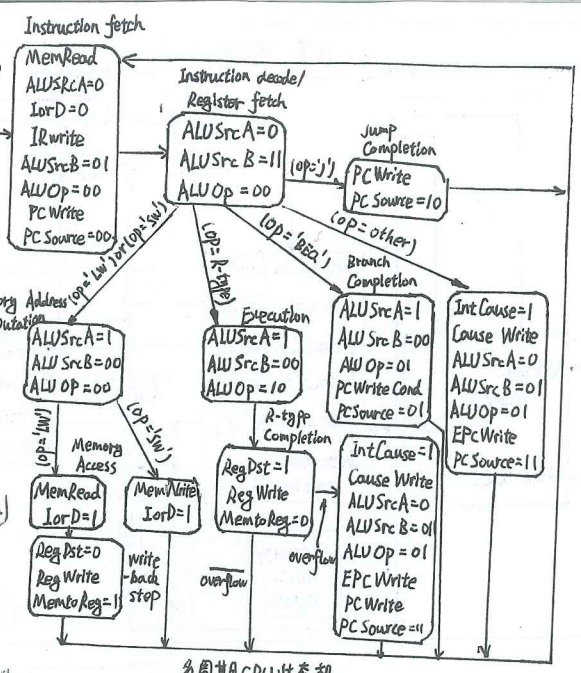
迭代次数	步骤	乘数	被乘数	积
0	初始值	0011	0000 0010	0000 0000
1	1a: 1→积=积+被乘数	0011	0000 0010	0000 0010
	2: 被乘数←1位		0000 0100	
	3: 乘数→1位	0001		
2	1a: 1→积=积+被乘数		0000 0100	0000 0110
	2: 被乘数←1位		0000 1000	
	3: 乘数→1位	0000		

由于 iteration 3、4 都因为乘数为0而无操作. 所以最终积为 0000 0110

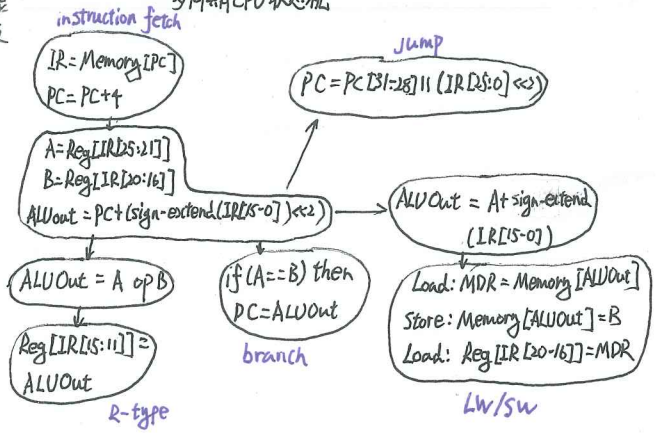


Single Cycle CPU Datapath

单周期指的是单时钟周期
由于单周期时钟频率上限需要照顾延迟最大的指令, 而很多指令不用这么大的时间间隔, 因此用多周期提高效率



多周期CPU状态机



DMA 实际上是一种低一级的调度器. CPU将如何调动 memory 与 I/O 设备间的数据的权限交给DMA此DMA管理. 数据直接由 Memory、I/O 设备间的数据通路传输. DMA 负责管理