Number System

- Sign-and-Magnitude

1-bit sign: 0 for +, 1 for –; n-1 bit magnitude.
Largest Value: $2^{n-1}-1$; Smallest Value: $-2^{n-1}+1$; Zeros: $+0, -0$

- r-1's complement

For positive number, complement is itself.
For negative, n-bit integer, m-bit fraction:

$$-x = r^n - r^m - x$$

构造一个全是$(r-1)$的数(eg. 111...111)，然后减掉对应的正数，得到负数补码。对二进制是各位取反。

- r's complement

For positive number, complement is itself.
For negative, n-bit integer, m-bit fraction:

$$-x = r^n - x$$

构造一个全是 0 的数，然后减掉对应的正数，得到补码。
Smallest: -2^n(1000...000)，only one 0
If there is a carry out of MSB, ***add 1*** to result.

- Excess Notion

Excess-7: 0000 map to -7，平移过去

- Floating Point

sign | exponent | mantissa
single-precision(32-bit): 1-8-23, excess-127, hidden-1
double-precision(64-bit): 1-11-52, excess-1023, hidden-1

exponent	fraction = 0	fraction ≠ 0	Equation
$00_H = 0000\ 0000_2$	±0	denomalize number	$(-1)^{sign} \times 2^{-126} \times 0.fraction$
$01_H, \ldots, FE_H$	normal	normal	$(-1)^{sign} \times 2^{exponent-127} \times 1.fraction$
FF_H	±∞	NaN	

- Tips:

1. round off(四舍五入，比如 0b0.001 得到 0b.01)
2. Overflow: pos + pos = neg; neg + neg = pos;

MIPS

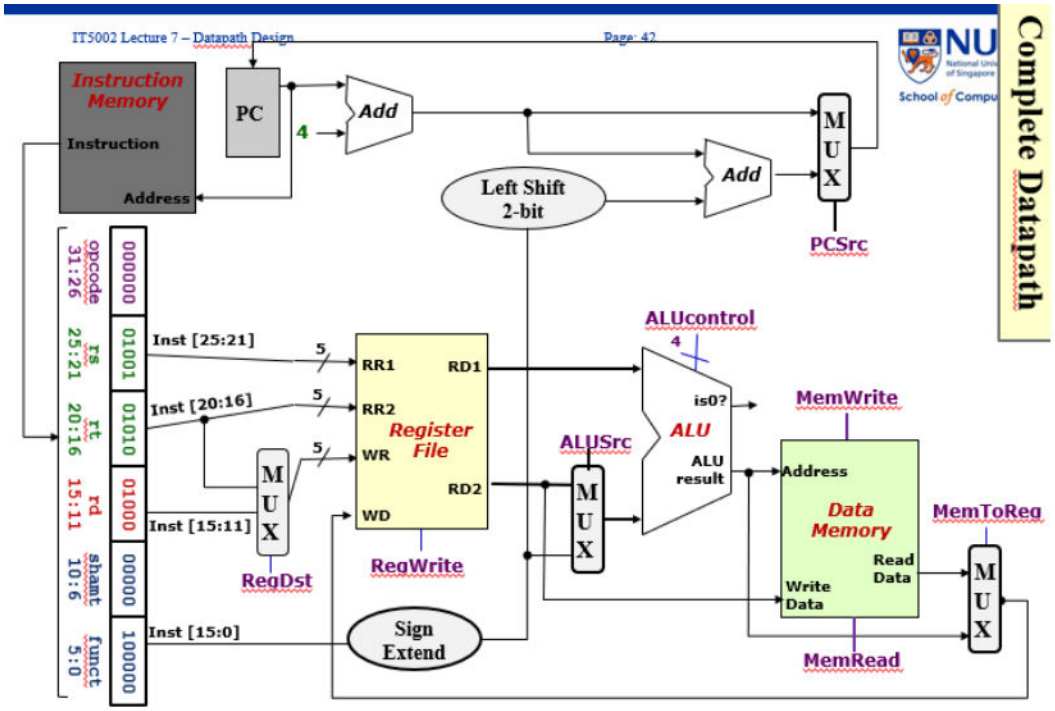
R-Format: opcode(6,all 0)+ rs(5), rt(5), rd(5), shamt(5), funct(6)
add, sub, ***slt***, ***sll***(sll rd, rt, shamt).
I-Format: opcode(6) + rs(5) + rt(5) + immedate(16)(Signed Number)
Branch(bne, beq)

$$PC + 4 + (im \times 4) \ / \ PC + 4$$

Branch range: $\pm 2^{15}$ words or $\pm 2^{17}$ bytes.
J-Format: opcode(6) + target add(26)(Unsigned)
Address

$$Add = 4bit(MSB)\ of\ (PC+4) + target(26bit) + 00(2bit)$$

Jump range: 256MB, 2^{28}bytes



ALUcontrol Signal:

5. Generating ALUcontrol Signal

Opcode	ALUop	Instruction Operation	Funct field	ALU action	ALU control
lw	00	load word	xxxxxx	add	0010
sw	00	store word	xxxxxx	add	0010
beq	01	branch equal	xxxxxx	subtract	0110
R-type	10	add	10 0000	add	0010
R-type	10	subtract	10 0010	subtract	0110
R-type	10	AND	10 0100	AND	0000
R-type	10	OR	10 0101	OR	0001
R-type	10	set on less than	10 1010	set on less than	0111

Instruction Type	ALUop
lw / sw	00
beq	01
R-type	10

ALUcontrol	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	slt
1100	NOR

Generation of 2-bit ALUop signal will be discussed later

Signal Output:

	RegDst	ALUSrc	MemTo Reg	Reg Write	Mem Read	Mem Write	Branch	ALUop	
								op1	op0
R-type	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

Signal Input:

	Opcode (Op[5:0] == Inst[31:26])						
	Op5	Op4	Op3	Op2	Op1	Op0	Value in Hexadecimal
R-type	0	0	0	0	0	0	0
lw	1	0	0	0	1	1	23
sw	1	0	1	0	1	1	2B
beq	0	0	0	1	0	0	4

Pipeline

Instruction	IF	ID	ALU	MEM	WB
Arithmetic	X	X	X		X
Branch	X	X	X		
Load	X	X	X	X	X
Store	X	X	X	X	

➢ Single cycle:

Cycle time: $\sum_{k=1}^{N} T_k$

Time : 每条指令花费时间的和

➢ Multi cycle:

1. 计算 cycle time: 每个 stage 中时间最长的为 cycle time.
2. 计算每条指令的时间，根据 cycle time * (#stage)
3. 计算 CPI，平均时间
4. Total time: #Instructions * CPI

➢ Pipeline:

1. 计算 cycle time: (max(每个 stage 时间) + delay time)
2. Total time: (I+N-1) * cycle time; (N-1)是 fill time

Speedup: single cycle or Multi-cycle 花费时间/ pipetime 时间

引入额外的寄存器:

IF/ID:

- PC+4(Branch), rs, rt, rd, shamt, func, imm(16)

ID/EX:

- PC+4, RD1, RD2, rt, rd(存疑，我觉得此处可以决定 DstReg 了), Imm(32)
- MtoR, RegWr, MemR, MemW, Branch, RegDst, ALUsrc, ALUop

EX/MEM:

- BrcTgt(PC+4+4*Imm), isZero, ALUres, RD2, DstReg(rt or rd)
- MtoR, RegWr, MemR, MemW, Branch

MEM/WB:

- MemRes, ALUres, DstReg
- MtoR, RegWr

Cache

Hit & Miss:

$$AvgAccTime = \text{Hit rate} \times \text{Hit time} + (1\text{-Hit rate}) \times \text{Miss penalty}$$

Hit time: Time to access cache.

Miss penalty: Hit time + time to memory

Types of Miss:

- Cold/Compulsory Miss: on the first time access to a block. (如果 cache block 从来没有装载过)。First reference miss, cold start miss。
- Conflit Miss: 曾经装载过，但是换出了。Collision miss, interference miss. Fully Associative Cache 不会发生。
- Capacity Miss: 全部 cache 容量满了。

Policy:

- Write Policy: Cache hit
  - Write-Through: 同时写入 Cache 和 MEM, using write buffer.
  - Write-Back: add a dirty bit, only write to memory when kicked out.
- Write Miss Policy: Cache miss
  - Write-Allocate: Load block to cache. 变成了 cache hit 情况.
  - Write-Around: Directly write to MEM.
- Replacement:
  - Least Recently Used(LRU): 需要记录最近什么时候用过
  - FIFO:先进先出
  - Random Replacement
  - Least Frequently Used:使用频率最低

N-Way Set Associative Cache: N-way: 一个 set 里有 n 路(可以放 n 个不同的 tag)

1. Block Size: 2^N bytes, N-bit
2. Number of **Cache Set**: cache 总容量/(n*block-size) = 2^M, M-bit
3. Tag: 32 – (N+M) bits

Fully Associative Cache: N=Block Number, a block can be stored in anywhere.

No Set Index

(Directly mapped: 一列; N-way: n 列, set index 行; Fully: N 列，1 行)