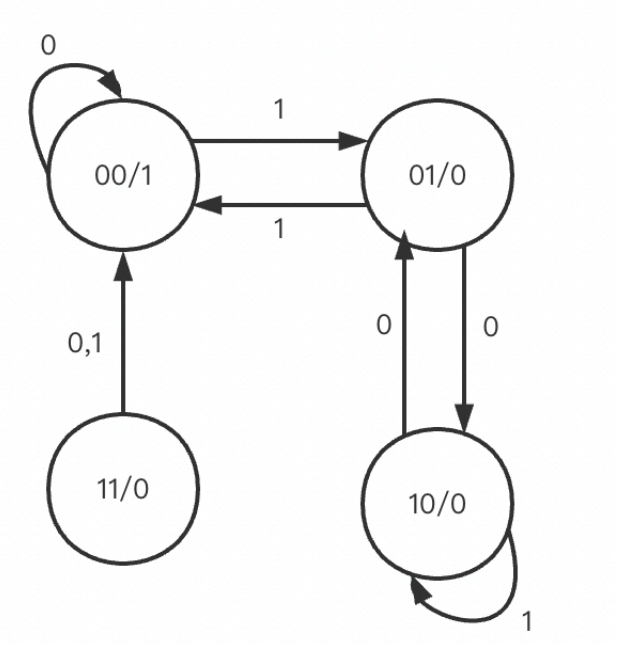
HW3 反馈

T1

(a).

S_1	S_0	X	Z	S_1'	S_0'
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	0	0

(b).



这里状态是 S_1S_0, 不是 S_1S_2 (我一开始也写错了) 还有状态机是要画的, 有同学直接没画.

T2

R1 + R2 is zero or positive.

T3

(a) opcode

(b) operand

T4

FETCH: 1 + 100 + 1 = 102

DECODE: 1 (memory access is not required)

EVALUATE ADDRESS: 0 (this phase is not required)

FETCH OPERANDS: 1 (memory access is not required)

EXECUTE: 1 (memory access is not required)

STORE RESULT: 0 (this phase is not required)

> FETCH 看英文版书第 135 页 Figure 4.4, 3 个 state, 对应 130-131 页的 3 个 Step, 其中第二个 state (Step) 是要 access memory 的, 用 100 周期, 其余是 1 个周期.
>
> 这里 STORE RESULT 应该是有争议的, 按照书 132 页 4.3.2.6 的说法, 对于 ADD 指令, "A seperate STORE RESULT phase is not needed", 因为这里考的是第四章, 我在答案里给出的是不含这个 phase, 但是第五章数据通路那里又确实有这个 phase, 所以 0 和 1 应该都算对. 这种如果考试里出出来助教肯定会好好讨论尽量给大家分的.
>
> 同时之前也有不少人问过第四章的 FETCH OPERANDS 阶段是不是跟第五章讲的有冲突. 确实是有冲突, 因为第四章没有 implementation, 第五章有具体的数据通路, 所以肯定是以第五章为准的. 第四章就看个概念.
>
> 有同学是只数了每 state, 没有加起来, 这肯定不行的.

T5

opcode 有 56 条，则需要 6bit 才能表示完整。

register 有 64 个，也需要 6bit 才能表示完整。

则 IMM 可用位数为：32 - 6 - 6 - 6 = 14bit

则表示范围：$-2^{13} \leq$ IMM $\leq 2^{13} - 1$

T6

(a)

ADD(0001) 和 AND(0101) 有了更大的立即数范围。

NOT(1001) 没有获益。

(b)

load 和 store 能有更多的一位去寻址。

(c)

branch 中没有寄存器，因此没有变化。

> 注意 c, BR 里面确实有立即数寻址, 但是里面没有寄存器, 寄存器位数变小了也不能让偏移量更大

T7

ADD 的 opcode: 0001

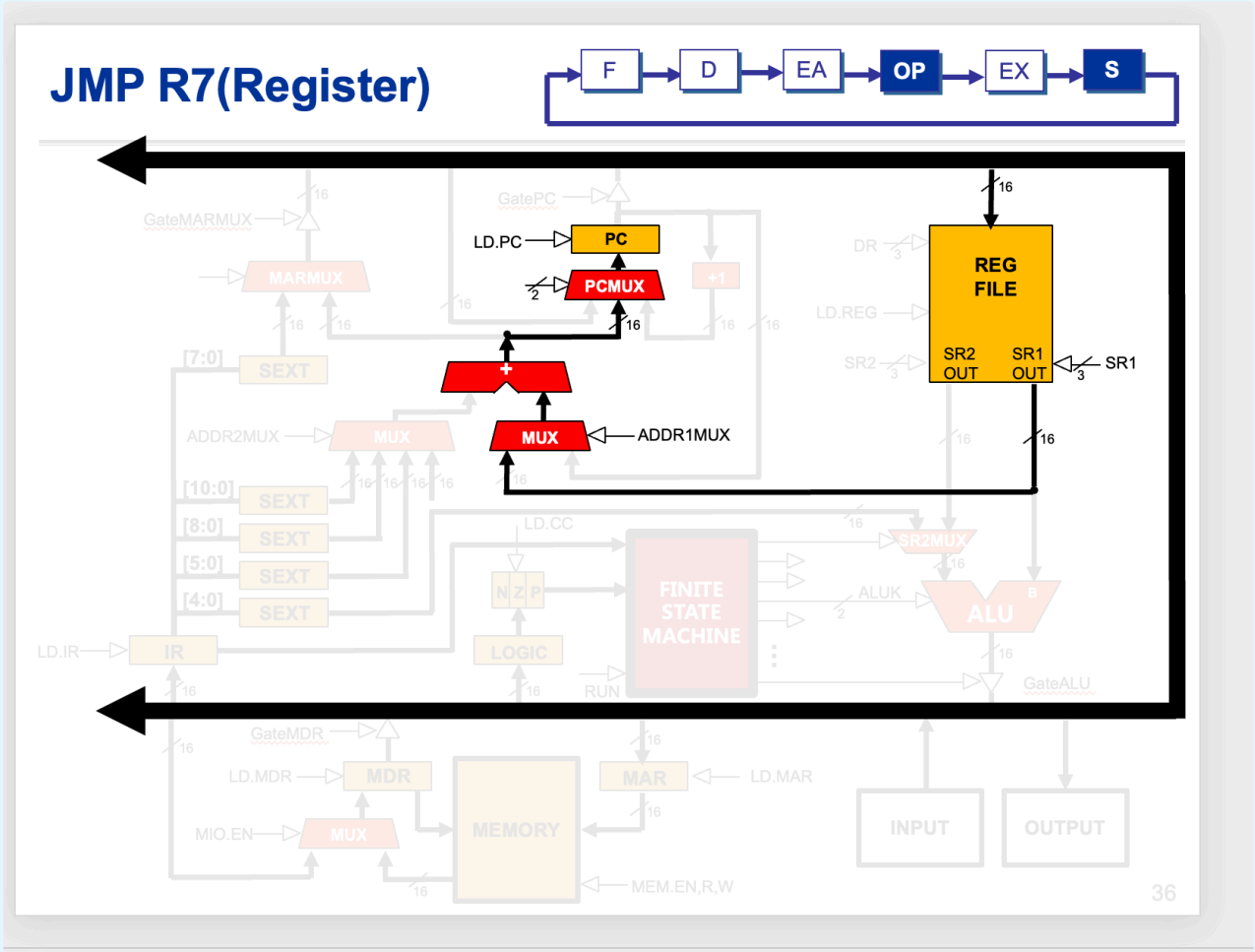
STR 的 opcode: 0111

JMP 的 opcode: 1100

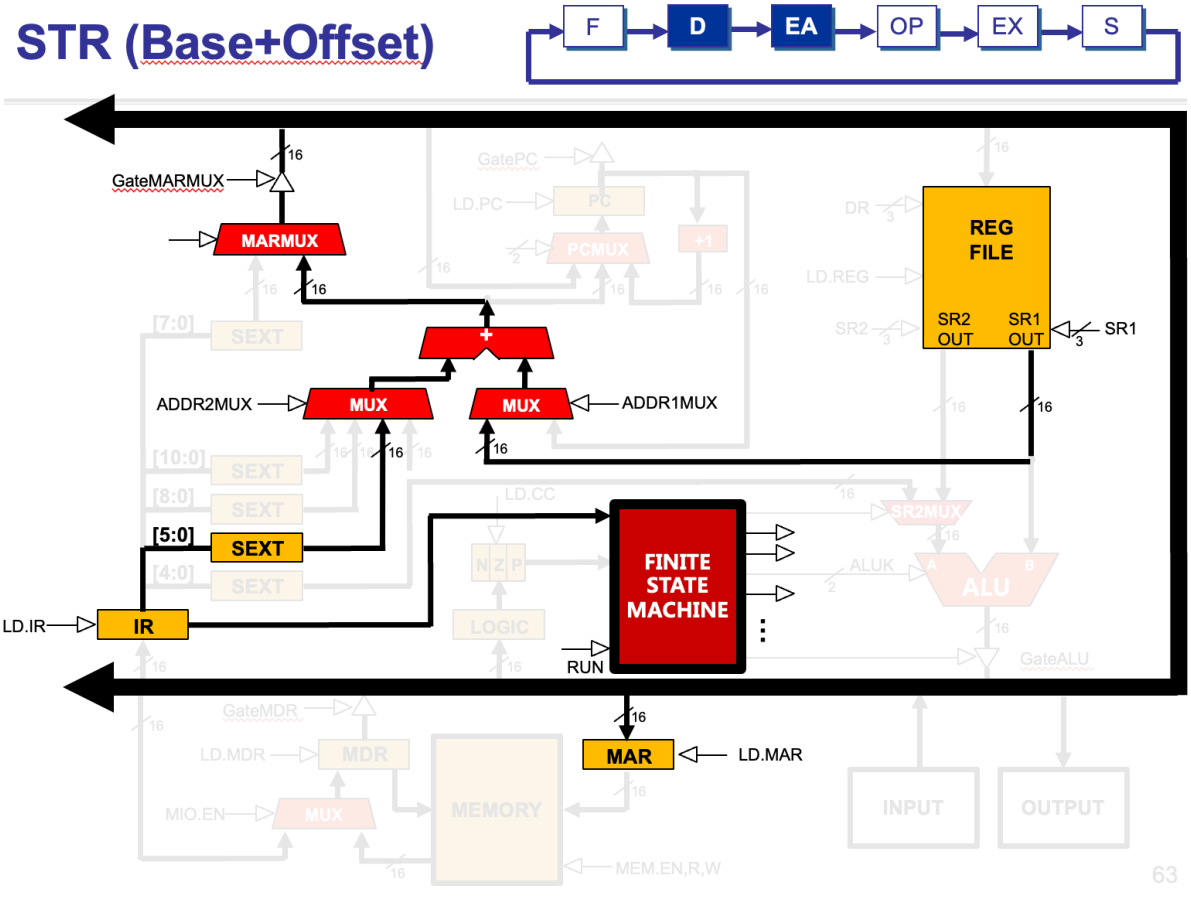
	fetch instruction	decode	evaluate address	fetch data	execute	store result
PC	0001，0111，1100					1100
IR	0001，0111，1100					
MAR	0001，0111，1100		0111			
MDR	0001，0111，1100			0111		

(这其实已经考到第五章了)

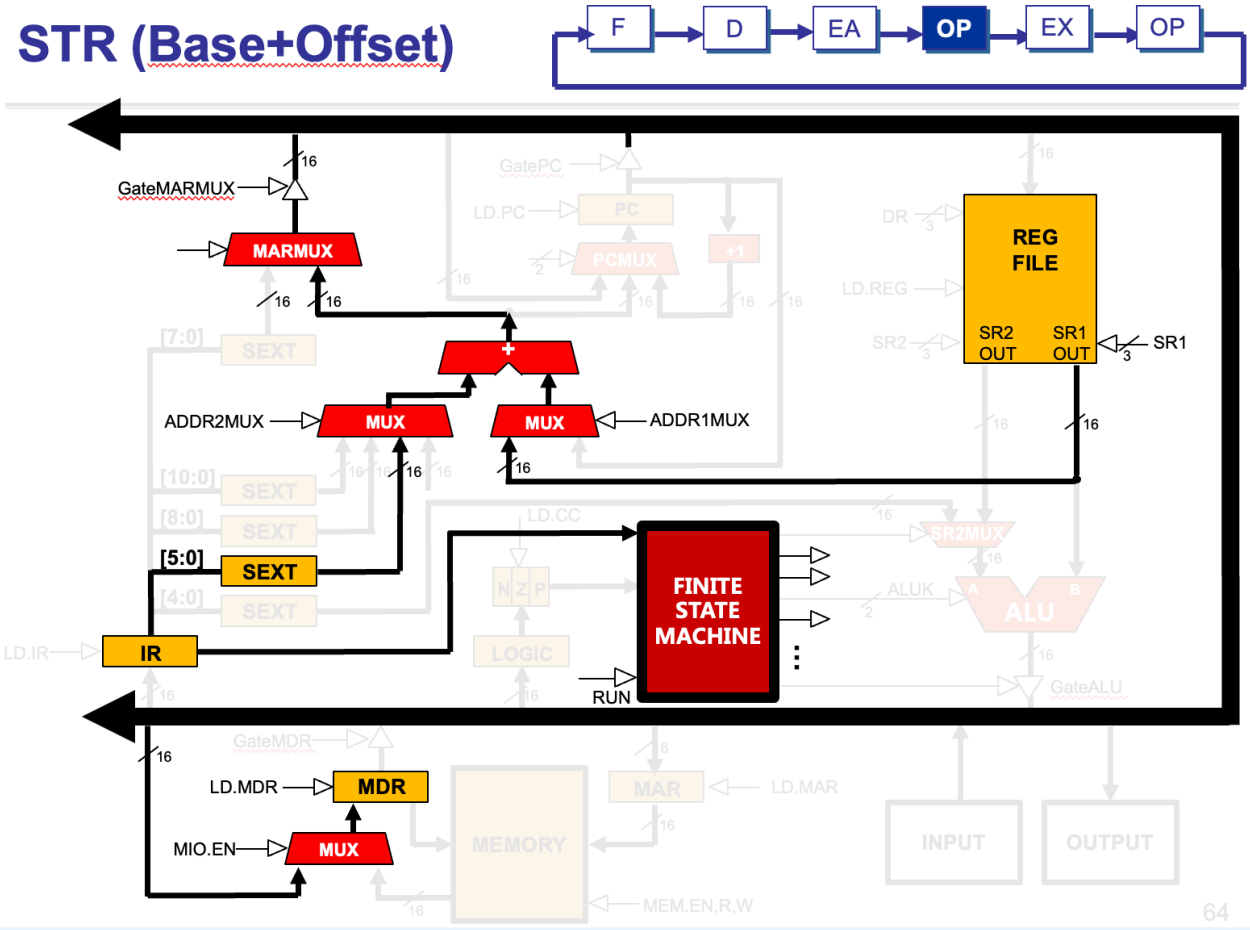
所有指令在 FETCH 阶段都是有 MAR -< PC, PC <- PC + 1, MDR <- M[MAR], IR <- MDR 的, 所以四个寄存器都被写, 第一列填上所有 opcode, 之后 JMP 会写 PC, STR 会写 MAR 和 MDR, 参见下面的数据通路:



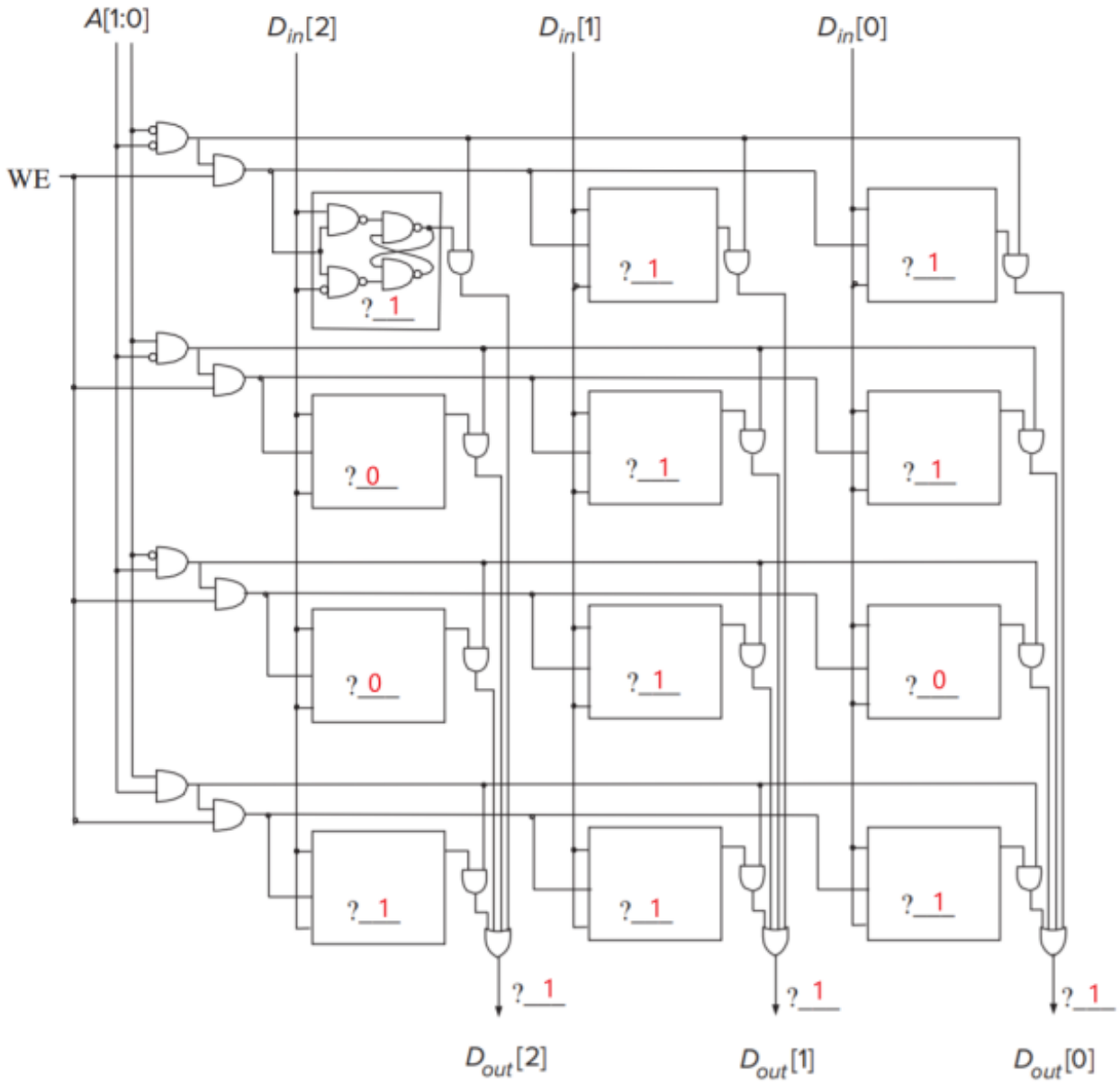
STR (Base+Offset)



STR (Base+Offset)



T8



只要看每个地址最后一次被写时 D_{in} 的值即可, 则地址 00 应该是 cycle 6 写的 111, 01 是 cycle 4 写的 011, 10 是 cycle 3 写的 010, 11 是 cycle 7 写的 111, D_{out} 看最后一个 cycle, 在读 11 处的值, 所以读出 111.

T9

这里题目没写清楚, 如果认为第一周期 MDR 还没被写入, 答案:

(a)

MAR: 000

MDR: 00011001

(b)

01100010

如果认为已经写入:

(a)

MAR: 001

MDR: 00110000

(b)

00010101

两种都算对.

> 推导过程: 这里其实有点像链表, 拿这 MDR 的高 3 位作为 MAR 再去取值, 所以对于没写入的情况, MAR 的变化过程是:
>
> 000 -> 001 -> 111 -> 011
>
> 这里 MAR 是 MDR 的高位, 所以对应 MDR 是
>
> 00011001 -> 00110000 -> 11110001 -> 01100010
>
> 对于已经写入的情况也是同理, 只是多推一步, 也就是 011 -> 000

T10

为了叙述方便, 下面会简称 Operation i 为 i, 表 1 指代 Operations on Memory, 表 2, 3, 4 指代 Memory before Access 1, Memory after Access 3, Memory after Access 5. 同时建议大家在看的同时按照我的推导填表.

因为

> The data in the MDR must come from a previous read (load).

而 Operation 3 的 MDR 与 1 有不同, 说明 Operation 2 是 Read. 而 Memory 在 Access 3 后与 Access 1 前 x4000 和 x4001 都发生了变化, 说明 Operation 1 和 3 写了这两处地址, 假设 1 写了 x4001, 与第二位是 0 矛盾, 因此 1 写了 x4000, 3 写了 x4001, 同时该步的 MDR 是 10xx0, 比对表 2, 知道应该来自 x4003. 到这里填出了表 1 前 3 行和表 2, 3 除了 x4002 外的行.

之后发现 x4003 在 Access 5 由 10110 变成了 01101, 发生了 Write, 说明 4 是 R, 5 是 W, 5 的 MDR 是 01101, 考虑到表 3 在上一步填完后没有 01101, 说明 x4002 处值为 01101, 得解, 最终答案如下:

Operations on Memory

	R/W	MAR	MDR				
Operation 1	W	x4000	1	1	1	1	0
Operation 2	R	x4003	1	0	1	1	0
Operation 3	W	x4001	1	0	1	1	0
Operation 4	R	x4002	0	1	1	0	1
Operation 5	W	x4003	0	1	1	0	1

Memory before Access 1

x4000	0	1	1	0	1
x4001	1	1	0	1	0
x4002	0	1	1	0	1
x4003	1	0	1	1	0
x4004	1	1	1	1	0

Memory after Access 3

x4000	1	1	1	1	0
x4001	1	0	1	1	0
x4002	0	1	1	0	1
x4003	1	0	1	1	0
x4004	1	1	1	1	0

Memory after Access 5

x4000	1	1	1	1	0
x4001	1	0	1	1	0
x4002	0	1	1	0	1
x4003	0	1	1	0	1
x4004	1	1	1	1	0

解答写的比较详细了, 一步步跟着推就行, 不爱推也没事, 习题课也会再讲.

T11

a. OPCODE 有 225 条，需要 8 位才能表示完整

b. 寄存器有 120 个，需要 7 位才能表示完整

c. 32 - 7 * 3 - 8 = 3，最多有 3 位 UNUSED

T12

a. 5×10^8

b. $5 \times 10^8 \ / \ 8 = 6.25 \times 10^7$

c. 由于流水线，每条指令执行时间相当于只需要 1 个 cycle，所以每秒可以执行约 5×10^8条指令（流水线启动和退出的耗时导致不能完全达到最大值，但是可以忽略）

a b 是送分的, c 超纲, 考试不会考.

T13

Fetch：将 PC 放入 MAR，PC++，将 M[MAR] 放入MDR，将 MDR 的内容放入IR

Decode：译码

Evaluate Address：地址计算（比如地址值等于 R3 内容和立即数 6 之和）

Fetch Data：读取指令处理所需要的源操作数（来自寄存器或者内存）

Execute：执行操作

Store Result：将执行结果写入目的寄存器或内存.