

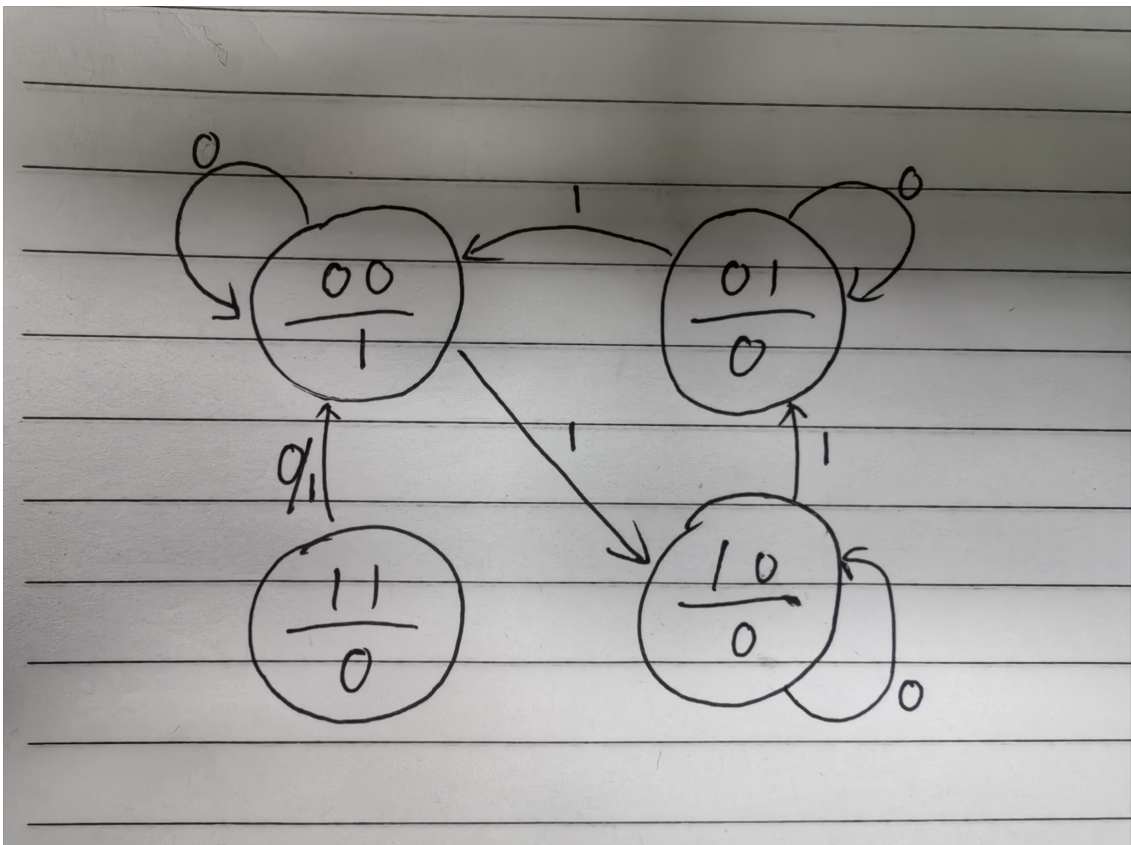
## Hw3

### T1

1.

$S_1$	$S_0$	$X$	$Z$	$S'_0$	$S'_1$
0	0	0	1	0	0
0	0	1	1	1	0
0	1	0	0	0	1
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	0	0	1
1	1	0	0	0	0
1	1	1	0	0	0

2.



### T2

•	$X$	$Halt?$	$R_0$
	000000010	yes!	2
	000000001	yes!	3
	000000000	yes!	6
	111111111	no!	-
	111111110	no!	-

### T3

- 内存取码：100；
- 译码+寄存器读取：1；
- 执行加法：1；
- 写入：1
- $100 + 1 + 1 + 1 = 103$

### T4

- 对于 **NOT** 指令，即使寄存器占用位数变少，也并无更多可用的地方，故没好处；
  - 对于 **ADD**、**AND** 的立即数相关模式，可使用的立即数从 **imm5** 到 **imm7** 范围变大，灵活性更高，有好处；
- 有好处；
  - 对于 **LD**、**ST**，寄存器字段占用位数减少，于是可扩展偏移量字段，使得可访问更远的内存地址，提高了代码效率；
- 无好处；
  - 条件分支指令并无寄存器字段，因而不会对该指令产生影响。

### T5

1.  $2^7 < 225 < 2^8$ , 故为 8；
2.  $2^6 < 120 < 2^7$ , 故为 7；
3.  $32 - 8 - 7 \times 3 = 3$ , 即为 3。

### T6

1	1	1
0	1	1
0	1	0
1	1	1
1	1	1

### T7

•	<b>Operation No.</b>	<b>R/W</b>	<b>MAR</b>	<b>MDR</b>
	1	W	x4000	11110
	2	R	x4003	10110
	3	W	x4001	10110
	4	R	x4002	01101
	5	W	x4003	01101

•	<b>Address</b>	<b>Before Access 1</b>	<b>After Access 3</b>	<b>After Access 5</b>
	x4000	01101	11110	11110
	x4001	11010	10110	10110
	x4002	01101	01101	01101
	x4003	10110	10110	01101
	x4004	11110	11110	11110

## T8

Address	Instruction
x3000	1001 111 001 111111
x3001	1001 110 010 111111
x3002	0101 101 111 000 010
x3003	0101 100 110 000 001
x3004	1001 001 101 111111
x3005	1001 010 100 111111
x3006	0101 000 001 000 010
x3007	1001 011 000 111111

## T9

1. 不能;  $R_2 = R_1 + \#2$
2. 能;
3. 不能; 若条件码为 N 或 P , 则分支到  $PC + 4$  , 否则顺序进行。
4. 不能;  $R_2 = NOT R_7$
5. 不能; *TRAP 23*: 打印提示到控制台, 从键盘读取和回显字符。

## T10

- **限制：**BR 指令使用 PC 相对寻址，只能跳转到当前 PC 附近的位置 ( $PC - 256 \sim PC + 255$ )，而无法访问较远的内存位置；
- **JMP 优势：**JMP 通过使用寄存器内存储的 16 位地址来跳转，使得该指令基本可访问内存中任意地址。
- **例子：**从 x3000 跳转到 x4000。
  - 对于 BR 指令，由于  $x4000 - x3010 = x1000$ ，转换为十进制为  $16^3 = 4096 \gg 255$ ，超过了 BR 指令能访问的范围，需进行多次跳转；
  - 对于 JMP 指令，可预先将 x4000 存入寄存器  $R_1$ ，进而用 JMP 跳转实现访问。