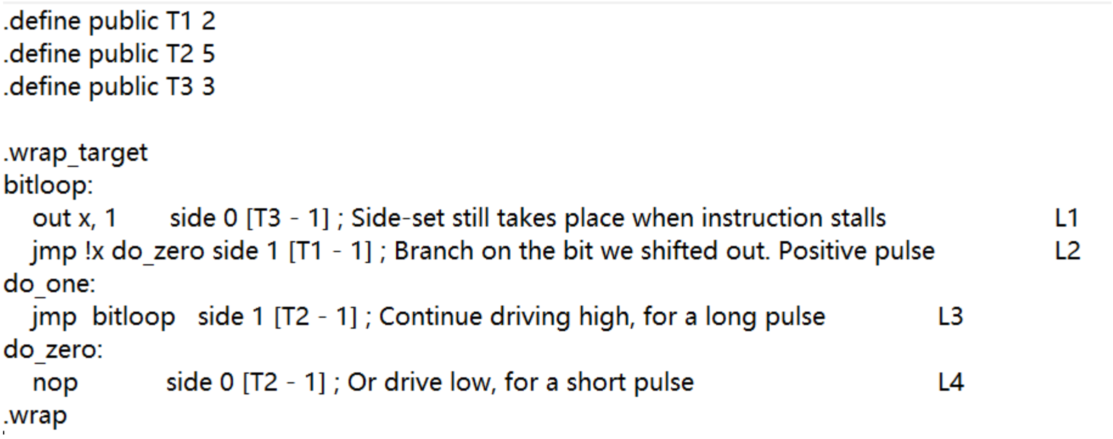
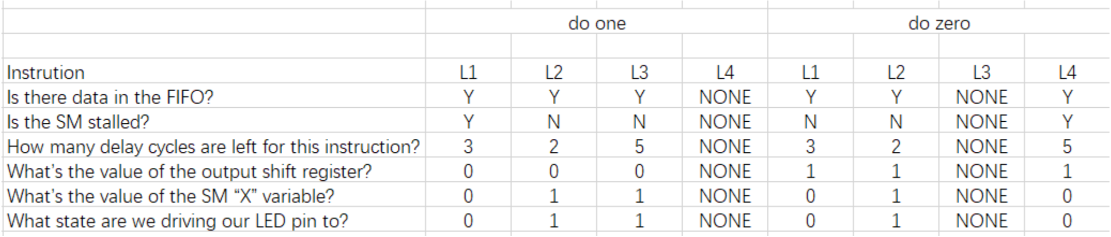
We can write SM from the code

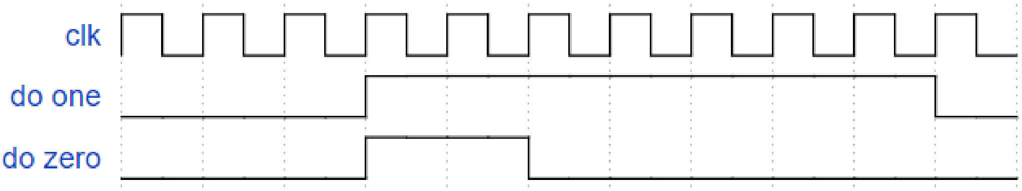


```
.define public T1 2
.define public T2 5
.define public T3 3

.wrap_target
bitloop:
    out x, 1       side 0 [T3 - 1] ; Side-set still takes place when instruction stalls           L1
    jmp !x do_zero side 1 [T1 - 1] ; Branch on the bit we shifted out. Positive pulse              L2
do_one:
    jmp  bitloop   side 1 [T2 - 1] ; Continue driving high, for a long pulse              L3
do_zero:
    nop            side 0 [T2 - 1] ; Or drive low, for a short pulse                       L4
.wrap
```



| | do one | | | | do zero | | | |
|---|---|---|---|---|---|---|---|---|
| Instrution | L1 | L2 | L3 | L4 | L1 | L2 | L3 | L4 |
| Is there data in the FIFO? | Y | Y | Y | NONE | Y | Y | NONE | Y |
| Is the SM stalled? | Y | N | N | NONE | N | N | NONE | Y |
| How many delay cycles are left for this instruction? | 3 | 2 | 5 | NONE | 3 | 2 | NONE | 5 |
| What's the value of the output shift register? | 0 | 0 | 0 | NONE | 1 | 1 | NONE | 1 |
| What's the value of the SM "X" variable? | 0 | 1 | 1 | NONE | 0 | 1 | NONE | 0 |
| What state are we driving our LED pin to? | 0 | 1 | 1 | NONE | 0 | 1 | NONE | 0 |


clk
do one
do zero