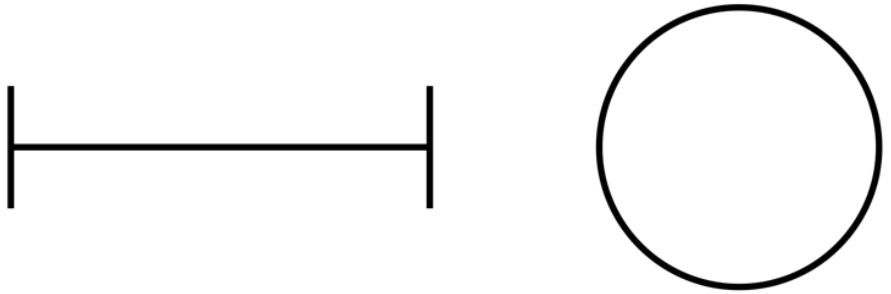
Sequential Ladder Logic

What is Sequential Logic

- Most of the time when working with PLCs and ladder logic we want a series of things to happen in an order
- For instance, waiting for a sensor to activate before we fire an output ect.
- We can use a sequential methodology to write out this code.

Cyclic vs Linear

- Cyclic programs return to the start after they finish meaning they repeat indefinitely until stopped
- Linear programs run once and then stops in an idle state waiting for an input

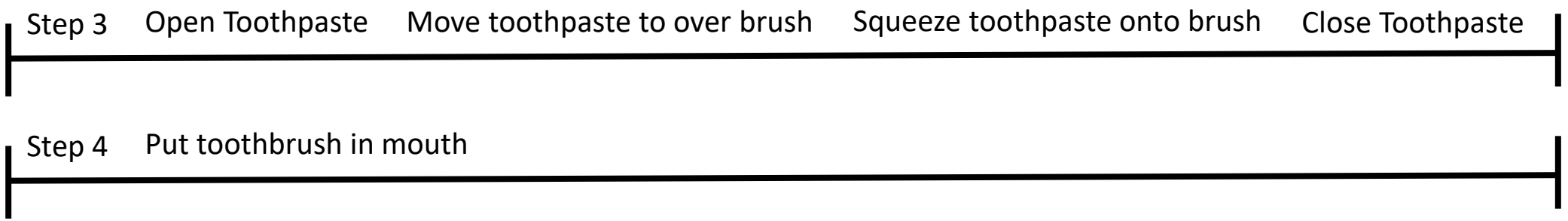


4 Key Parts of a PLC Sequence

- **The sequence start and stop**
  - Input conditions (e.g., start/stop buttons) that initiate or terminate the entire sequential process.
- **The sequence steps**
  - Defined stages or actions that occur in a specific order to complete the process.
- **Sequence Step Transition Conditions**
  - Logical conditions (e.g., sensor signals, timers, or counters) that dictate when to move from one step to the next.
- **Outputs That Initiate Actions**
  - The actuators or devices (e.g., motors, solenoids, lights) triggered by the logic to perform physical tasks in each step.

Example with toothpaste

- What are the steps to brushing your teeth in the morning?
- Can any of these steps be broken down into more detailed steps?
- Is the process cyclic or linear?
- What do we have to measure and change to make this process change? Can we write out a
- Can you write out a basic logic diagram for this process I.E:

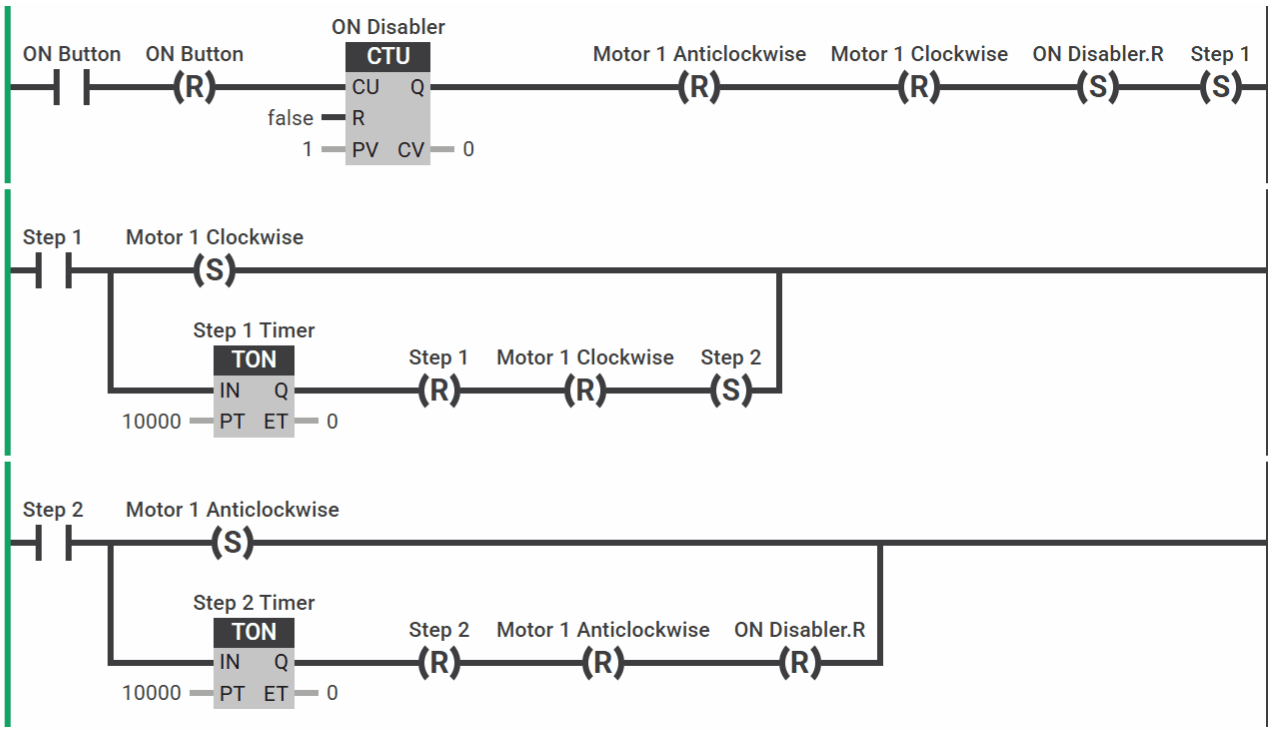


Example with 3 Cups of Tea (teams of 3)

- Can you do the same for making 3 cups of tea?
  - Think about how many times you do each thing
  - How can we count how many times that has been done
  - Think if you can break down each step into any further detail

Example of a sequential ladder logic

Name	Type	Value
ON Button	Bool	False
Step 1	Bool	False
Motor 1 Clockwise	Bool	False
Motor 1 Anticlockwise	Bool	False
Step 1 Timer	Timer	
ON Disabler	Counter	
Step 2	Bool	False
Step 2 Timer	Timer	



This line controls the turning on of the program enabling Step 1 at the end

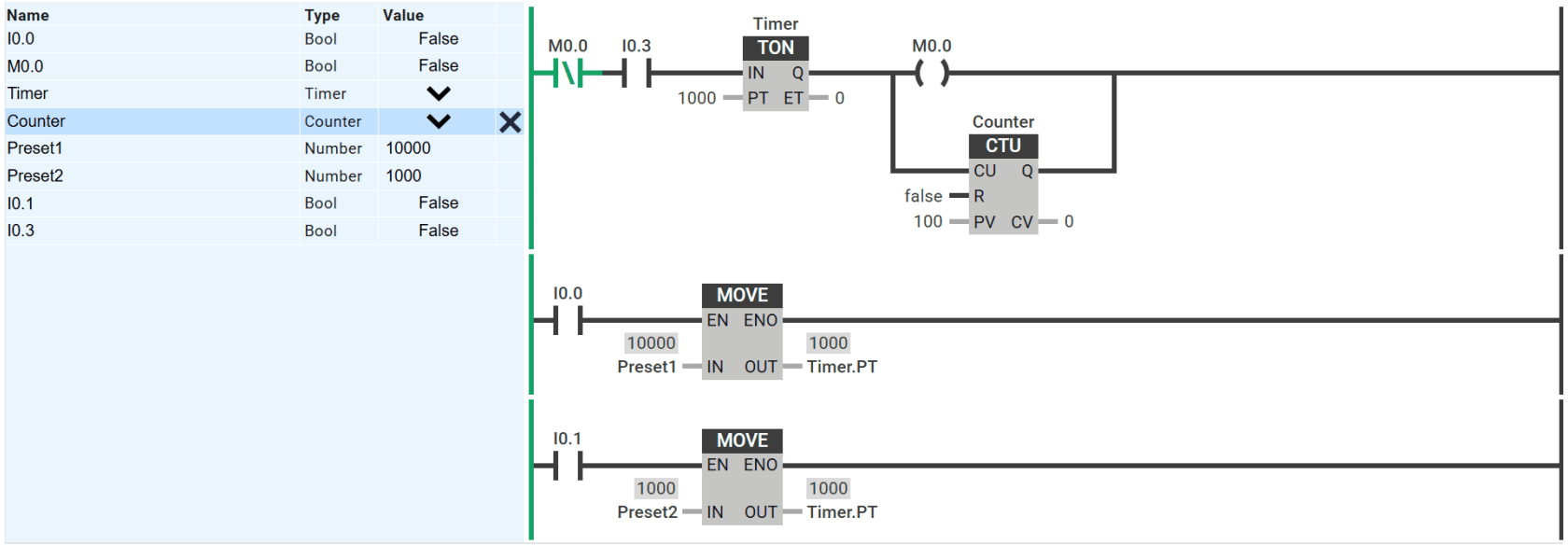
This line is step 1 for this step we have the motor moving Clockwise for 10 seconds

This line is step 1 for this step we have the motor moving Anticlockwise for 10 seconds reverting to its original position

On this slide there is a bit of bad ladder logic programming can you work out which bit?

What are Move Functions

- Move functions allow us to move a numerical value into a memory location
- This means we can move values into our timers and counters too



Comparators

Equals

Name	Type	Value
First Number	Number	1
Second Number	Number	1
q0.0	Bool	True

GRT

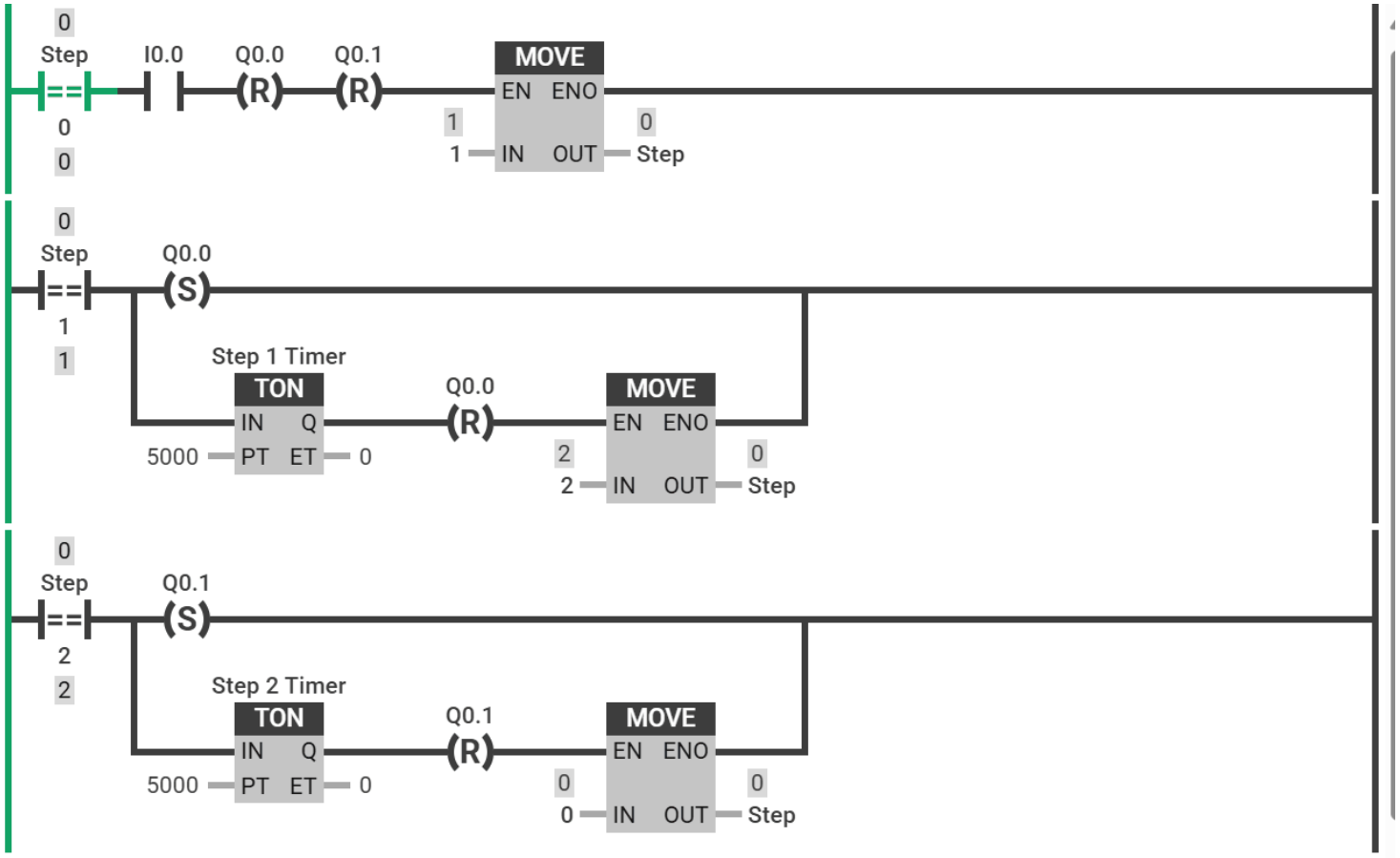
Name	Type	Value
First Number	Number	1
Second Number	Number	1
q0.0	Bool	False

LSR

Name	Type	Value
First Number	Number	1
Second Number	Number	1
q0.0	Bool	False

Using comparators

Name	Type	Value
I0.0	Bool	False
Q0.0	Bool	False
Q0.1	Bool	False
Step	Number	0
1	Number	1
2	Number	2
Step 1 Timer	Timer	
Step 2 Timer	Timer	
0	Number	0



Sequential Notation

- **Structured Representation**: Uses a step-based approach (A+, B+, A-, B-) to describe actuator movements in a logical sequence.
- **Time-Controlled Actions**: Time delays (T1, T2, etc.) ensure proper execution and prevent overlap of operations.
- **Simplifies Ladder Logic Design**: Helps in developing and debugging PLC programs by providing a clear, stepwise breakdown of the automation process.

Example of sequential notation

- Let's put our example ladder logic from earlier into notation:
- We have motor 1 which goes clockwise (+) or anticlockwise(-)
- We can write the notation for this action as [M1+, M1-]
- However, this doesn’t account for the time the motor spins for (10s)
- So we can write [M1+,10s,M1-,10s]

Sequential notation with actuators

- With items like actuators where there is a set full extension and retraction, we don’t necessarily have to write out the time they run for
- So, for an actuator fully extending then fully retracting we can write: [A1+,A1-]
- We can add the times if we want it to extend for a set time
- Bear in mind in your PLC software this will be done by a gate taking a reading of when the actuator is extended

Sequential Notation Delays

- We can write delays into our program using T(number), during this delay nothing will be happening
- So, for example if we want an actuator to fully extend and then stay extended for 5 seconds whilst nothing else happens, then retract we can put this:
- [A1+, T10, A1-]

Sequential notation with repeats

- For repeating actions, we can use exponentials
- For example, if we have an actuator we want to fully extend, wait for 3 seconds and then fully retract 3 times we can write this:
- $[(A1+, T3, A1-)^3]$
- If we put it to n that means it repeats indefinitely

Parallel Sequential Notation

- If we want two actions to happen in parallel, we can write one action above the other
- So, for instance if we have 2 actuators both extending at the same time we can write:
- $\begin{bmatrix} A1+ \\ A2+ \end{bmatrix}$

Understanding a sequential notion

- Say we have the notation: $[\binom{A1+}{A2+}, T2, \binom{A1-}{A2-}^{3}, M1+, 10s, T1]$
- So, in this notation we have 3 outputs:
  - 2 actuators
  - 1 motor
- So, both actuators extend, wait 2 seconds then retract 3 times then a motor moves for 10 seconds.
- This could be used in a conveyor system where two actuators are pressing down on lids

Your Turn

- Can you please make these sequences in your PLC software and then screenshot and explain what the code is doing:

- [B+, A+, A-, B-]
- $[(A+, 2s, A-, 2s)^3, T2, (B+, B-)^2, T2]$
- $[(A+,B+,T2,A-,B-)^n]$

Advanced Task

Can you make a PLC program for a **Coffee Machine**

- There are 3 sizes of cups, normal, small and espresso
- There are several components:
  - A coffee grinder with a motor
  - A pump with a motor which pumps the coffee