Task

5

Traffic Light Experiment

A. Purpose

The purpose of the traffic light experiment is to give a real world example of how a PLC might be used. This experiment also gives an example of a program that incorporates all the basic logical operators. Throughout this experiment the students with be exposed to inputs, outputs, timers, and the compare option.

B. Sample Program Code (Implementing a Timer Circuit)

Recall from Task 3 that timers have internal digital outputs and an analog reading. Also, the logics involving cumulative measurement effects for a particular output and for the ladder can be computed by mathematical functions like “addition” and “subtraction” in the logic software. Advance functions like the “PD & PID” controllers along with digital and analog device controls are also possible by the direct implementation of the instructions. Particularly, the gains in these controllers are very much easy to control by the end user and easily adjustable in the program mode; even after downloading the program. Advance High-speed devices are controlled with the “HSC” instructions in the menu where the switching speed is increased. File editing like addition of subroutines or macros is also possible to avoid monotonous repetitions of the ladder parts.

Before working with the TON timer (Figure 5.1), there are a few terms that the student must familiarize themselves with:

a) TON (Timer On): This instruction will time as long as the instructions preceding it on the rung are true and the accumulated value is less than the preset.

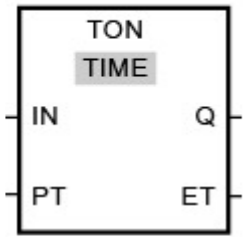
b) IN (Input): activate the timer when the rung conditions preceding the TON are true; otherwise, disable the timer.

c) PT (Preset time or target time): T#10s means the preset or target time is 10 second.

d) RT (Reset): This instruction writes a "0" to the EN (Enable) bit, TT (Timer Timing) bit, and DN (Done) bits.

e) ET (elapsed time): when calling #timer.ET, you will get a 32- bit sized number that gives a value in [ms]. It can be saved to an %MD time address.

f) Q (output): output 1 if ET=PT, output 0 if IN=0 or ET<PT; when Q=1, ET stops and hold, as shown in Figure 5.2.



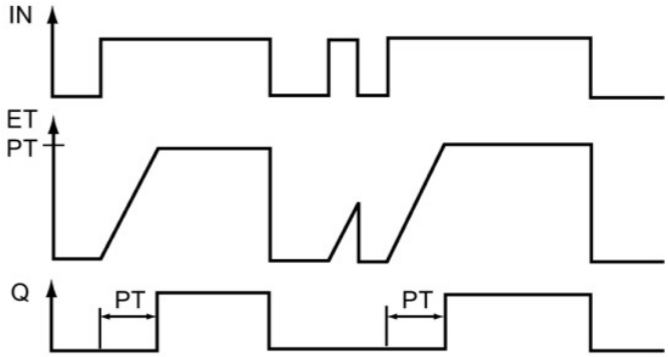


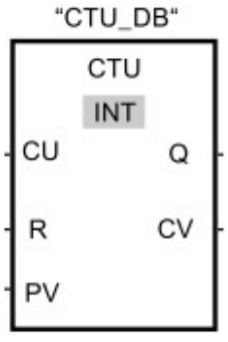
Figure 5.1: TON Timer Figure 5.2: Sequence diagram for TON Timer

The student must familiarize themselves with the CTU up counter (refer to Figure 5.3):

a) Whenever CU changes from "0" to "1", CV increases by 1 and it can be saved to an %MD address.

b) when CV = PV, Q outputs "1", and thereafter every time CU changes from "0" to "1", Q keeps outputting "1", The CV continues to increment by 1 until the maximum value of the integer type specified by the counter is reached.

c) At any time, as long as R is "1", Q outputs "0", and the CV immediately stops counting and returns to 0, shown in Figure 5.4.



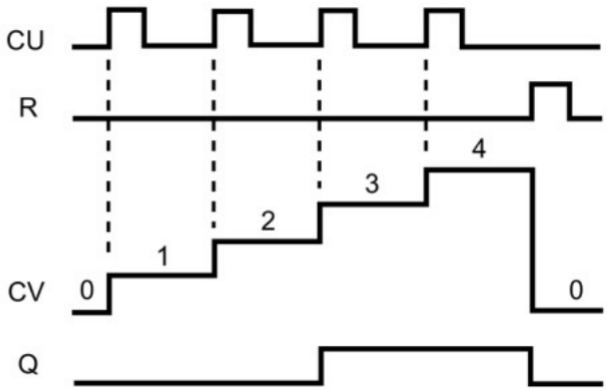


Figure 5.3: CTU Up Counter Figure 5.4: Sequence diagram for CTU Counter

To gain experience with this function in TIA Portal, build the program based on Task 5a, 5b, and 5c, shown in Figure 5.5, 5.6, 5.7: *(Note to students: Verify sample program I/O addresses with those from Table 4.2 on page 4.5)*

C. Exercises

Perform the following tasks:

1) Write a program using the stack lights to simulate traffic lights at an intersection. It is important to notice that the green and yellow lights from each stack should never be illuminated simultaneously.

2) Write a program using the stack lights that simulates the timing lights at a drag strip. Use the following sequence: Red Lights; Amber Lights; Green Lights; Buzzer.

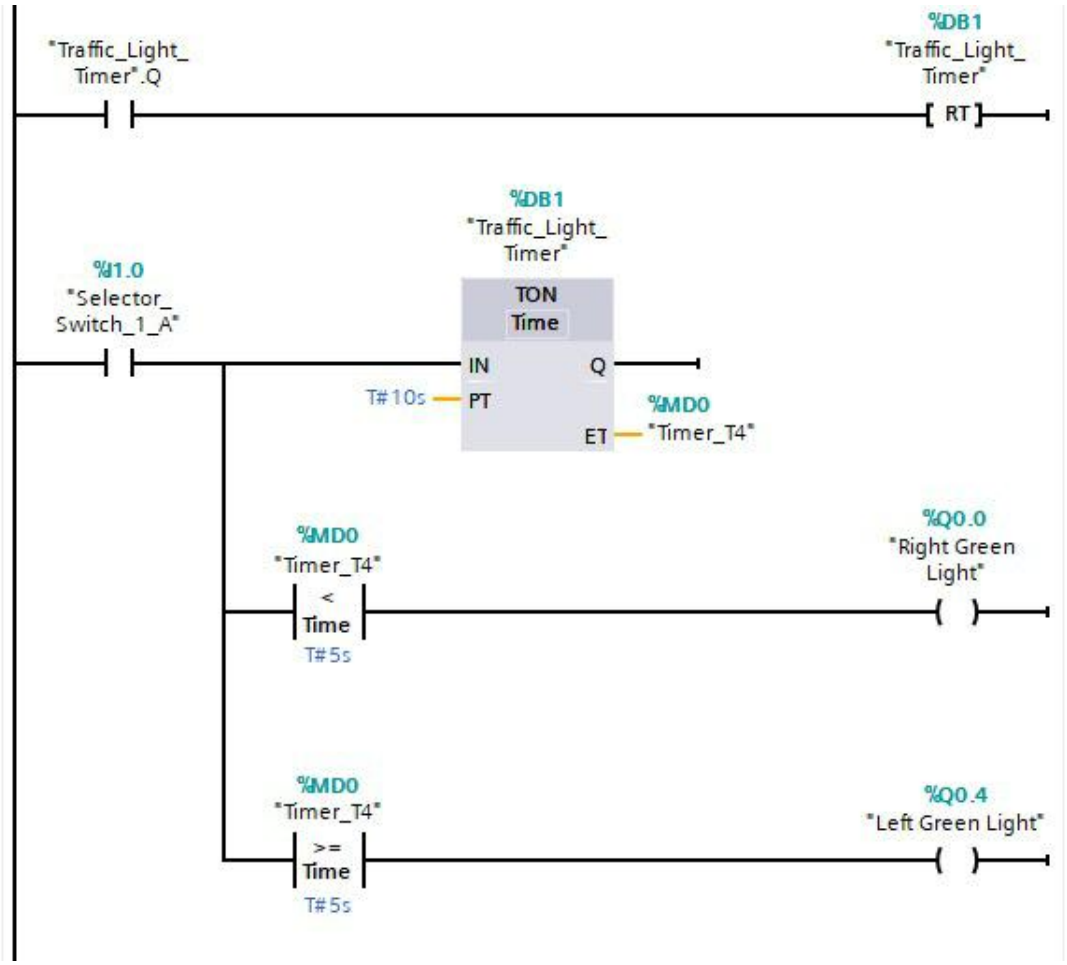


Figure 5.5: Task 5a Traffic Light with a TON Timer

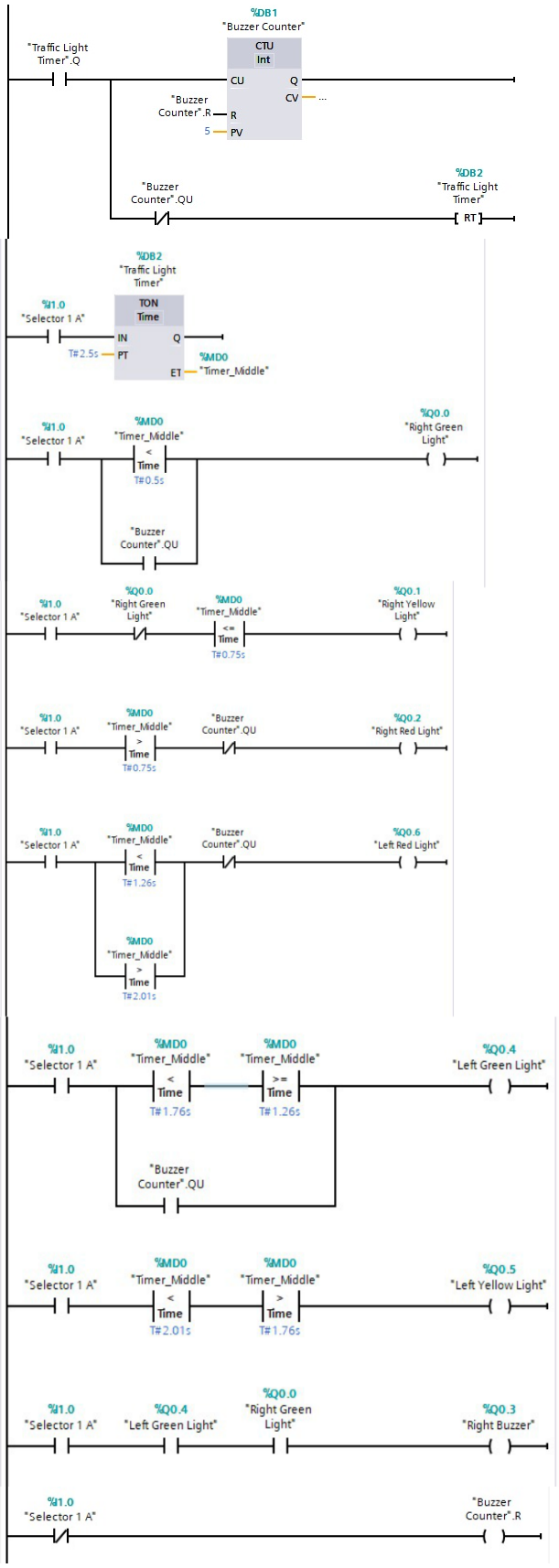


Figure 5.6: Task 5b Traffic Light with a TON Timer and a CTU Counter

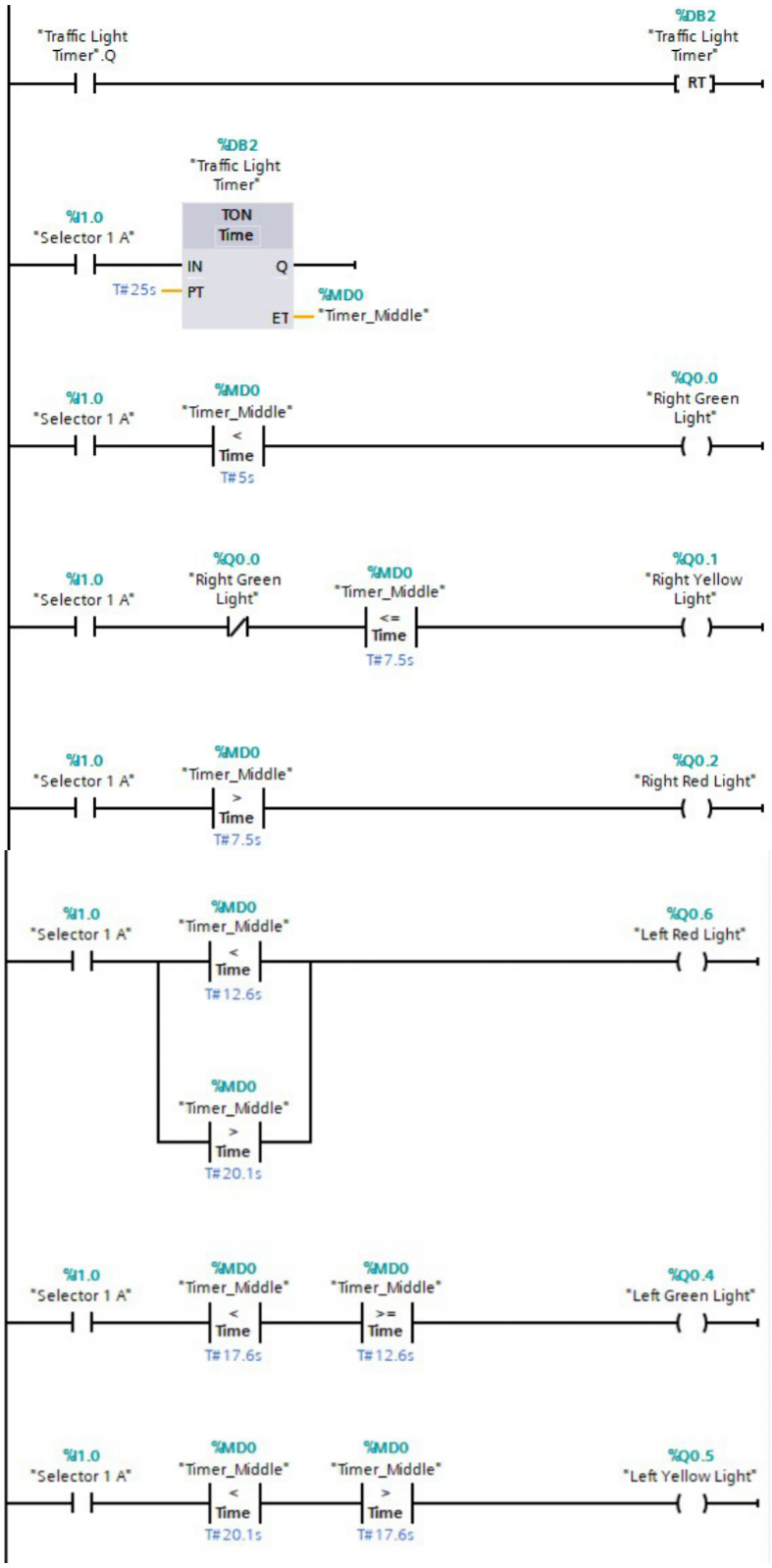


Figure 5.7: Task 5c Traffic Light with a TON Timer and a longer time interval