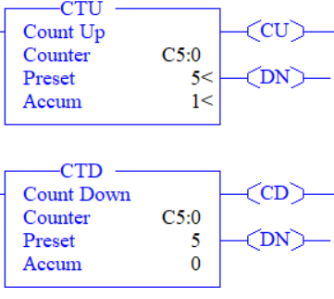
Title: **Counters** Handout: 4

Course: Introduction to Automation Unit: Introduction of PLC CLO: 4

**Objectives**

1. Student shall develop a knowledge of the Count Up (CTU) output instruction.
2. Student shall develop a knowledge of the Count Down (CTD) output instruction.
3. Student shall reinforce their knowledge of the decimal to binary conversion.

**Theory**

Count functions are output instructions that either count up (CTU) or count down (CTD) for each false-to-true rung continuity transition. Each count function produces an output when the accumulated value reaches the preset value. Rung transitions might be triggered by a pushbutton, limit switch or by parts passing by a photo-eye. Each count is retained when the rung loses logical continuity. The count is retained until a reset instruction with the same address as the counter is enabled, or if another instruction in the program overwrites the accumulated value. The accumulated value is retained after the CTU or CTD instruction goes false, and when power is removed from and then restored to the processor. Also, the on or off status of counter done, overflow, and underflow bits is retentive. The accumulated value and control bits are reset when a RES is enabled. Counter variables (found in the C5 table) use three words per element.

11 = UN (count down underflow) bit

12 = OV (count up overflow) bit

13 = DN (done) bit

14 = CD (count down enable) bit

15 = CU (count up enable) bit

The CD bits are always set prior to entering the REM Run or REM Test modes.

**Graphics**

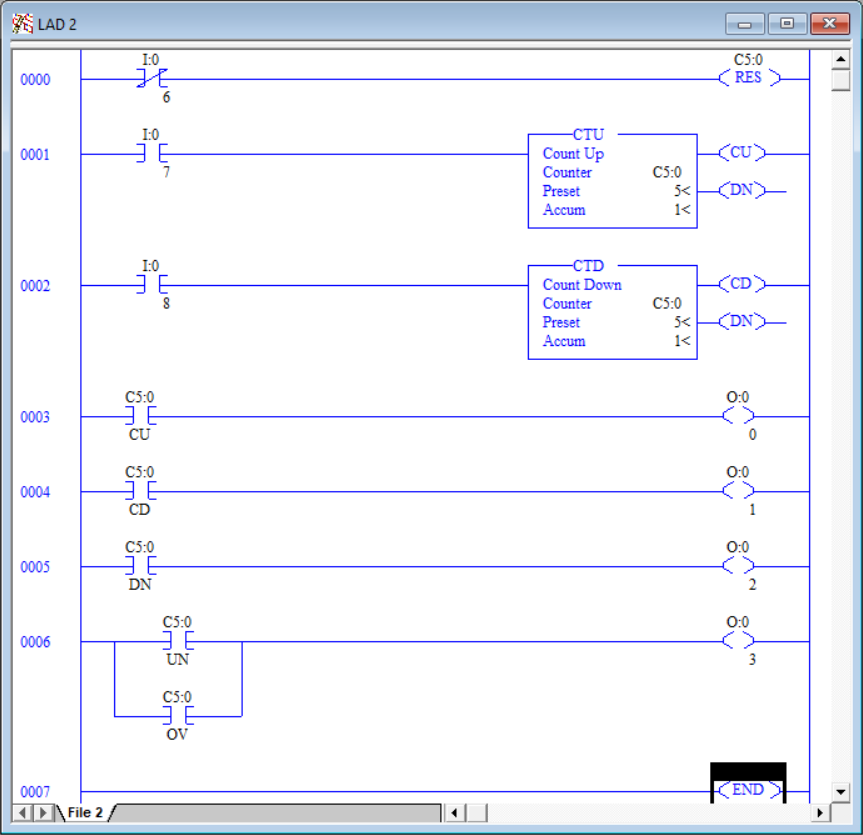
|  |  |
| --- | --- |
| CTU | CTD |
|  |  |

**Devices**

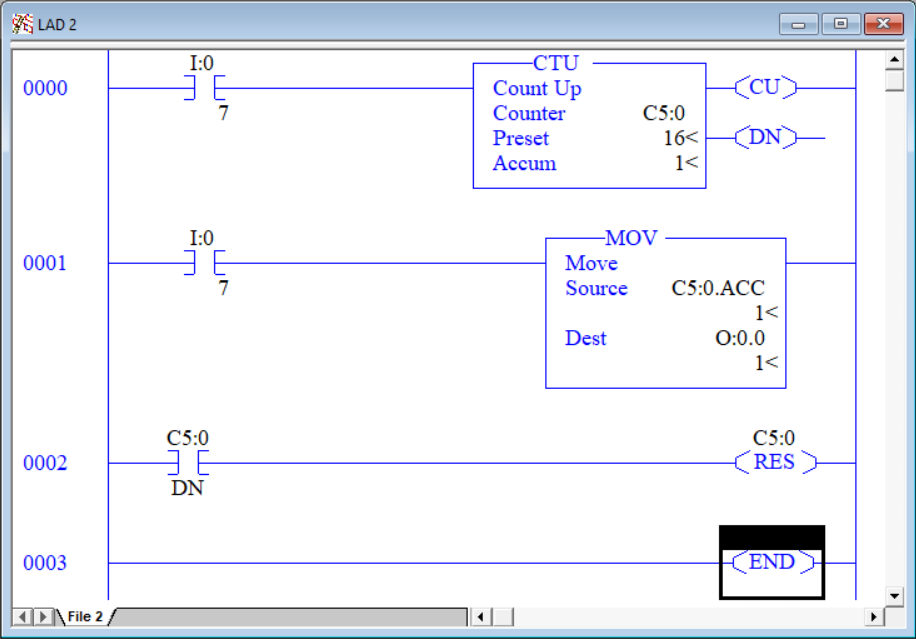
|  |  |  |
| --- | --- | --- |
| Inputs | | |
| *Device* | *Description* | *Symbol* |
| NC Pushbutton (PB1) | Reset Counter | CNT\_RST |
| NO Pushbutton (PB2) | Count Up | CNT\_UP |
| NO Pushbutton (PB3) | Count Down | CNT\_DW |
| Outputs | | |
| *Device* | *Description* | *Symbol* |
| Green Pilot Light | Count Up Enabled | CTU\_EN |
| Yellow Pilot Light | Count Down Enabled | CTD\_EN |
| Red Pilot Light | Count Done | CNT\_DN |
| Blue Pilot Light | Count Under/Overflow | UN\_OV |

**Instructions**

Program the logic should below.



1. Download the program and observe the logic online.
2. Press and release PB2. Notice that the green light comes on and then went off with the release of the button.
3. The *Accum* of the Count Up function went from 0 to 1.
4. Press and hold down PB2. Notice that the green light stays on, but the count only increased by 1. Holding down PB2 will not continually count while the button is pressed. It is necessary to release the button and press it again to increment the *Accum* once again.
5. Press and release PB3. Observe the yellow light illuminates while the button is pressed. Notice that the value decreases by 1. This happens because both the CTU and CTD have C5:0 assigned to them. The CTU will increase the *Accum* by 1 while the CTD will decrease the *Accum* by 1.
6. Pres and release PB2 until the *Accum* is exactly 5. The definition of a timer being done is when the Accum = Preset. When this condition occurs, the done (DN) bit is set to 1. The red light is indicating that the counter has finished.
7. Press and release PB2 two more times. Notice that the Accum is continuing the increase even though the counter is finished. The done (DN) bit remains on.
8. Press and release PB3 three times. Since the *Accum* < *Preset* the done (DN) bit is reset to 0.
9. Press and release PB3 until the Accum is less than zero. The CTD function decrements the Accum by one regardless of its value, so it is very possible to obtain large negative counts as well.
10. While still running online, edit the *Accum* value to -32,768. It is possible the change the *Accum* values of both timers and counters while online. This is the largest negative number the *Accum* word can represent.
11. Press and release PB3 button one time to force the count to exceed the *Accum* lower limit. This caused what is termed an underflow. The underflow (UN) bit is set which turns on the blue light. Notice that the Accum is now 32,767, which is the largest number that can be represented. The number “rolled over” much like an old time odometer would “roll-over” to 0 after reaching 99,999.9 miles.
12. The only way to clear the underflow bit (or the overflow bit), is the rest the counter. Rung 0000 has an output instruction that will not only set the Accum of the assigned counter variable to 0 but reset the underflow and overflow bits as well.
13. Press the PB1 button to reset the counter. Notice that the Accum and underflow bit where both reset to 0. No lights are on at this time.
14. While still running online, edit the *Accum* value to 32,767. This is the largest positive number the *Accum* word can represent.
15. The done bit is set since *Accum* >= *Preset* and the red light turns on.
16. Press and release PB2 once. The blue light illuminates again since the overflow (OV) bit is set. Notice the Accum “rolled-over” to the largest negative number that it can represent. Use the UN and/or OV bits to safeguard against this issue in production base programs.
17. Press the PB1 button to reset the counter. Notice that the *Accum* and overflow bit where both reset to 0. No lights are on at this time.
18. Program the follow logic.



*Accum* Integer Value

Bit Number

|  |  |  |  |
| --- | --- | --- | --- |
| Blue | Red | Yellow | Green |
| 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 2 |
| 0 | 0 | 1 | 1 | 3 |
| 0 | 1 | 0 | 0 | 4 |
| 0 | 1 | 0 | 1 | 5 |
| 0 | 1 | 1 | 0 | 6 |
| 0 | 1 | 1 | 1 | 7 |
| 1 | 0 | 0 | 0 | 8 |
| 1 | 0 | 0 | 1 | 9 |
| 1 | 0 | 1 | 0 | 10 |
| 1 | 0 | 1 | 1 | 11 |
| 1 | 1 | 0 | 0 | 12 |
| 1 | 1 | 0 | 1 | 13 |
| 1 | 1 | 1 | 0 | 14 |
| 1 | 1 | 1 | 1 | 15 |

1. Press and release the PB2 button to increment the *Accum* variable. The counter will automatically reset itself when it reaches 16. That is the maximum integer that our four pilot lights can represent ().