Timer and Counter Instructions

MET 382
Controls & Instrumentation
for Automation

Spring '08
T.E. Kostek

Topics

- Introduction to timers and counters
- Timer and counter applications
- ControlLogix Timers
- ControlLogix Counters

2

Introduction to Timers and Counters

Introduction to timers and counters

- Timers and counters:
  - Are fundamental PLC instructions common to all PLCs
  - Function like traditional hardwired timers and counters
  - Function as output instructions in a PLC program

4

Introduction to timers and counters

- Timers and counters generate both word level (DINT) and bit level (BOOL) data and status
- Bit level (**BOOL**) examples:
  - Done (DN) bit
  - Enable (EN) bit
- Word level (**DINTs**) examples:
  - Accumulated (ACC) value
  - Preset (PRE) value

5

Timer and Counter Applications

Timer Applications

- PLC timers often provide a necessary delay in operation while controlling a sequential machine, routing pallets on a conveyor system, etc.
- With respect to a conveyor application, a timer could be used to cause an on-delay so that a pallet traveling down the conveyor settles into a desired position. After the delay, the pallet might be:
  - *Raised* by a shot pin device into a fixed position
  - *Clamped* into a fixed position
  - *Pushed* by a ram in order to loop the pallet around the end of a conveyor

Continued on next slide 7

Timer Applications

- Timers are used when **event driven** programming is not possible
  - For instance, suppose a pneumatic cylinder does not have sensors which indicate when the cylinder is fully extended or fully retracted.
  - In this **time driven** application, a timer could be implemented such that the cylinder is extended for 5 seconds and then retracted back to its home position.

Continued on next slide 8

Timer Applications

- When event driven programming is implemented, a timer can be used to determine if an event fails to occur (i.e., Error Trapping).
- Example: Suppose a pneumatic cylinder has sensors which indicate when the cylinder is fully extended or fully retracted.
  - When the PLCs output is switched on (in order to extend the cylinder) a timer is enabled and starts timing.
  - When the cylinders extend sensor switches on (indicating the cylinder is fully extended):
    - The timer is reset (it is not allowed to time out)
    - The PLCs output will be switched off
    - The cylinder then retracts to its home position
  - If, however, there is a problem and the cylinder fails to fully extend, the timer will time out and an error will be generated.

9

Counter Applications

- Counters count events, such as the number of:
  - Parts passing a certain point on a conveyor system
  - Good parts/bad parts manufactured during a particular shift
  - The number of times a machine cycles through it's operation (for example, a machine may require preventative maintenance after 5,000 cycles)

Continued on next slide

10

Counter Applications

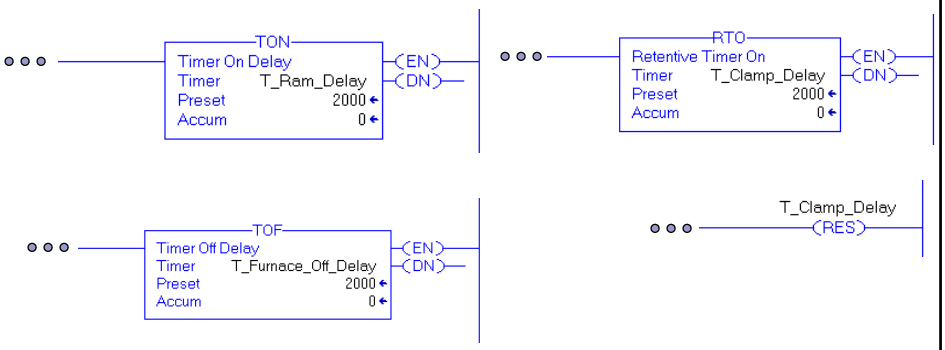
- For troubleshooting purposes, a counter can be used to monitor the steps of a sequential machine (e.g., a pneumatic robot).
  - At the start of the sequence, the counters accumulated value is reset to zero.
  - Each time the robot completes a step in its sequence, the counters accumulated value is incremented by one.
  - The counters accumulated value is reset each time the robot successfully completes its sequence.
  - If the robot fails to complete its sequence, the counters accumulated value will display the specific step at which the robot failed.
  - Knowing where (what step) the robot failed simplifies the troubleshooting process.

11

ControlLogix Timers

ControlLogix Timers

- Timer On-Delay (TON)
- Timer Off-Delay (TOF)
- Retentive Timer On (RTO) & Reset (RES)

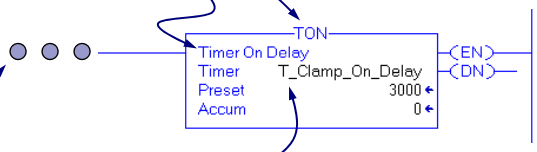


Timers are output instructions!

13

Timer Terminology

Type of timer instruction



Input side of rung intentionally not shown

Tag name

→ If you want to use a timer, you must create a tag of type timer.

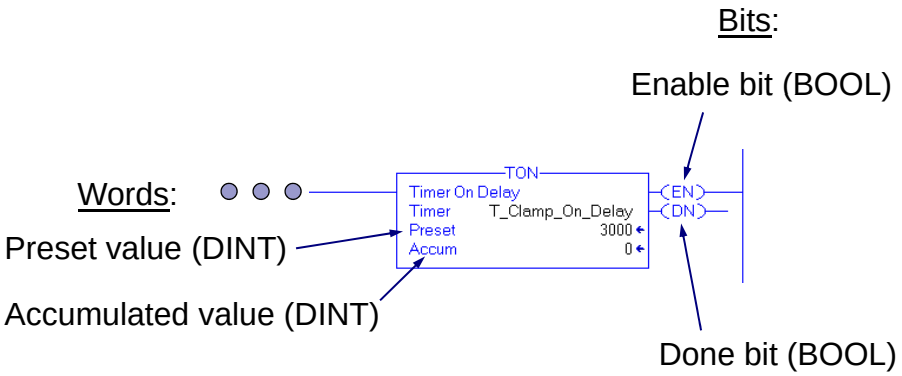
→In this class, the name of all timer tags will start with a "T_"

→ When entering the instruction, this tag must be defined before the preset and accumulated values can be entered.

→This tag is typically defined "on the fly"

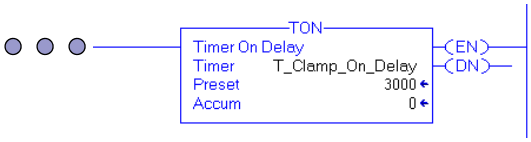
14

Timer Terminology



15

Timer Terminology



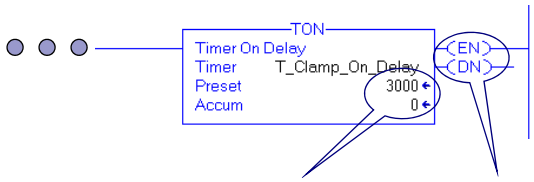
The time base = 1 ms for all ControlLogix timers, therefore:

The **Preset** value in seconds = “Preset” * 0.001
The **Accumulated** value in seconds = “Accum” * 0.001

The above example illustrates a 3 second on-delay timer:
3000 * 1 ms = 3000 * 0.001 = 3 seconds

16

Timer Terminology

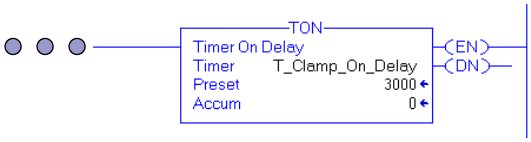


When online and in the run mode, these **word** values are displayed and updated in real time.

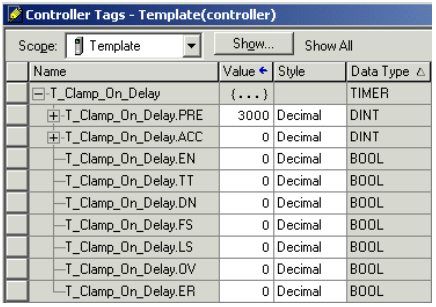
When online and in the run mode, these **bit** values are displayed and updated in real time:
→Highlighted green = “1”
→Not highlighted = “0”

17

Timer Terminology



Timer status can also be monitored via the tag database



18

Timer Terminology – Words (DINTs)

- **Preset** (PRE) Value
  - Specifies the value (in milliseconds) which the timer must reach before the done (DN) bit changes state.
  - Stored as a binary number (DINT)
  - A DINT stores a 32-bit signed integer ranging from: -2,147,483,648 to +2,147,483,647
- **Accumulated** (ACC) Value
  - Is the number of milliseconds the instruction has been enabled.
  - Stops changing when ACC value = PRE value.

19

Timer Terminology – Time Base

- Time Base (1 ms)
  - 1 ms x **PRE** value = actual preset time
  - 1 ms x **ACC** value = actual accumulated time

Key Points:

→The time base dictates the accuracy/resolution of the timer.

→Some PLCs allow the user to select the time base (e.g., 1 s, 0.1 s, etc.) on an individual basis for each timer instruction.

20

Timer Terminology – Bits (BOOL)

- **EN** - Enable Bit
  - On when the timer rung is true, hence the timer is enabled
- **TT** - Timer-Timing Bit
  - On when accumulated value is **changing**, hence timer is timing
- **DN** - Done Bit
  - Changes state when the accumulated (ACC) value is equal to the preset (PRE) value.
  - Its actual operation depends on the type of timer (on-delay, off-delay, etc.)

21

Timer Terminology – Tag Database

Controller Tags - Template(controller)

Scope: Template | Show... | Show All

Name	Value	Style	Data Type
T_Clamp_On_Delay	{...}		TIMER
T_Clamp_On_Delay.PRE	3000	Decimal	DINT
T_Clamp_On_Delay.ACC	0	Decimal	DINT
T_Clamp_On_Delay.EN	0	Decimal	BOOL
T_Clamp_On_Delay.TT	0	Decimal	BOOL
T_Clamp_On_Delay.DN	0	Decimal	BOOL
T_Clamp_On_Delay.FS	0	Decimal	BOOL
T_Clamp_On_Delay.LS	0	Decimal	BOOL
T_Clamp_On_Delay.OV	0	Decimal	BOOL
T_Clamp_On_Delay.ER	0	Decimal	BOOL

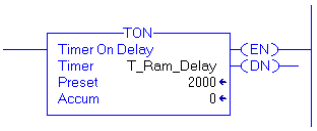
PRE = Preset, **ACC** = Accumulated

EN = Enable, **TT** = Timer Timing, **DN** = Done

(FS, LS, OV, ER → Not implemented at this time)

22

Timer On-Delay (TON)



- When the rung which contains the TON instruction goes **TRUE**:
  - The accumulated value starts incrementing (usually from 0)
  - EN = 1 (since the rung is true)
  - TT = 1 (since the accumulated value is changing)
  - DN = 0 (since the accumulated value does not yet = the preset value)
- When **ACC = PRE**:
  - The accumulated value stops incrementing
  - EN stays on for as long as the rung remains true
  - TT = 0 (since the accumulated value is not changing)
  - DN = 1 (since ACC = PRE)

23

Timer On-Delay (TON)

- Whenever the rung which contains the TON instruction goes **FALSE**:
  - The ACC value is reset to 0
  - The EN, TT, and DN bits are reset to 0

Key points:

→The TON instruction is a self resetting timer. When the rung goes false, the timer is automatically reset. A reset instruction can be used, but it is usually not.

→Timers are “level” based instructions not “transition” based instructions.

24

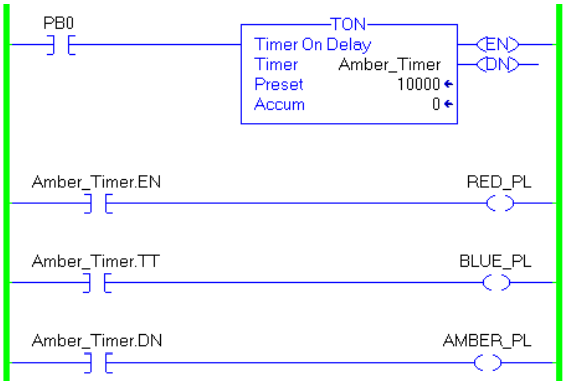
Timer On-Delay (TON)

- The accumulated value of a TON instruction is automatically reset:
  - When the rung goes false
  - When returning to the run mode after a power failure
  - When returning to the run mode after the processor was placed in the program mode

The TON instruction is not retentive!

25

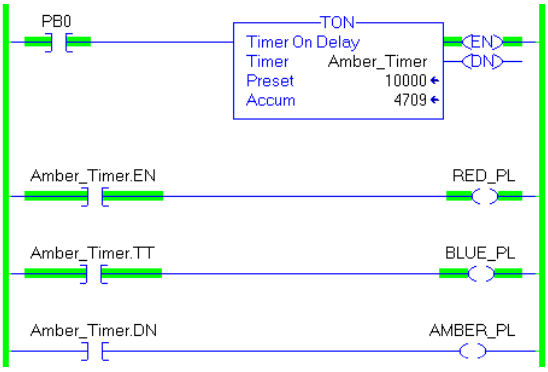
TON Example – State 1: PB0 not pushed



Tag Name	Value	Style	Type
Amber_Timer	{...}		TIMER
Amber_Timer.PRE	10000	Decimal	DINT
Amber_Timer.ACC	0	Decimal	DINT
Amber_Timer.EN	0	Decimal	BOOL
Amber_Timer.TT	0	Decimal	BOOL
Amber_Timer.DN	0	Decimal	BOOL

26

TON Example – State 2: PB0 being pushed



Tag Name	Value	Style	Type
Amber_Timer	{...}		TIMER
Amber_Timer.PRE	10000	Decimal	DINT
Amber_Timer.ACC	4709	Decimal	DINT
Amber_Timer.EN	1	Decimal	BOOL
Amber_Timer.TT	1	Decimal	BOOL
Amber_Timer.DN	0	Decimal	BOOL

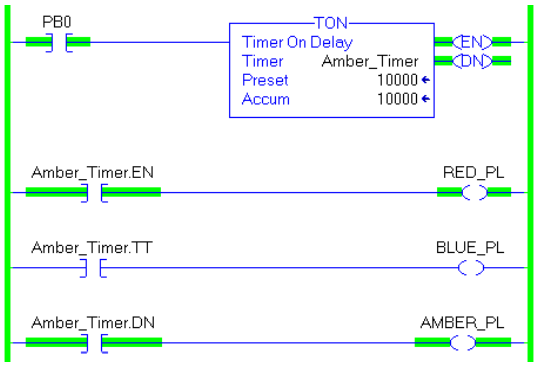
Notes:

PB0 is being pressed.

PB0 has been pushed for 4.709 seconds

27

TON Example – State 3: PB0 being pushed



Tag Name	Value	Style	Type
Amber_Timer	{...}		TIMER
Amber_Timer.PRE	10000	Decimal	DINT
Amber_Timer.ACC	10000	Decimal	DINT
Amber_Timer.EN	1	Decimal	BOOL
Amber_Timer.TT	0	Decimal	BOOL
Amber_Timer.DN	1	Decimal	BOOL

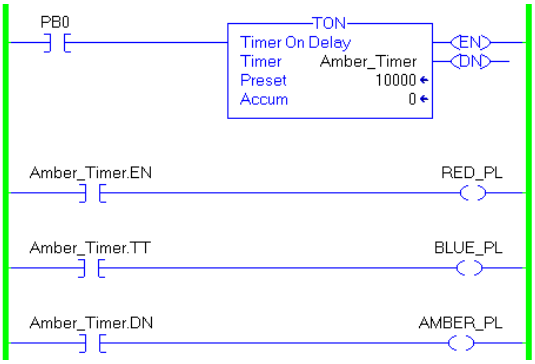
Notes:

PB0 is being pressed.

PB0 has been pushed for at lease 10 seconds.

28

TON Example – State 4: PB0 released



Tag Name	Value	Style	Type
Amber_Timer	{...}		TIMER
Amber_Timer.PRE	10000	Decimal	DINT
Amber_Timer.ACC	0	Decimal	DINT
Amber_Timer.EN	0	Decimal	BOOL
Amber_Timer.TT	0	Decimal	BOOL
Amber_Timer.DN	0	Decimal	BOOL

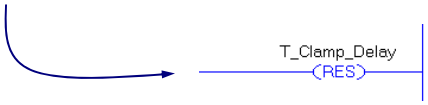
Notes:

PB0 is no longer being pushed.

29

Retentive Timer On (RTO)

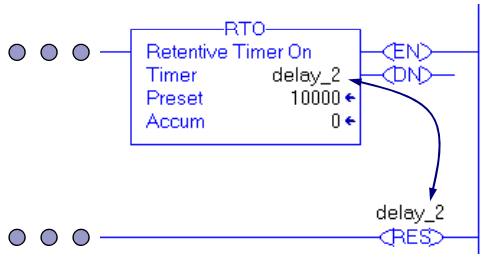
- Same as Timer On-Delay (TON), EXCEPT:
  - A retentive timer retains (remembers) it's ACC value even if the:
    - The rung goes false, or
    - The processor is placed in the program mode, or
    - The processor faults. or
    - The processor loses power (due to battery backup)
  - A RESET (RES) instruction must be used to reset the accumulated value of a retentive timer.



30

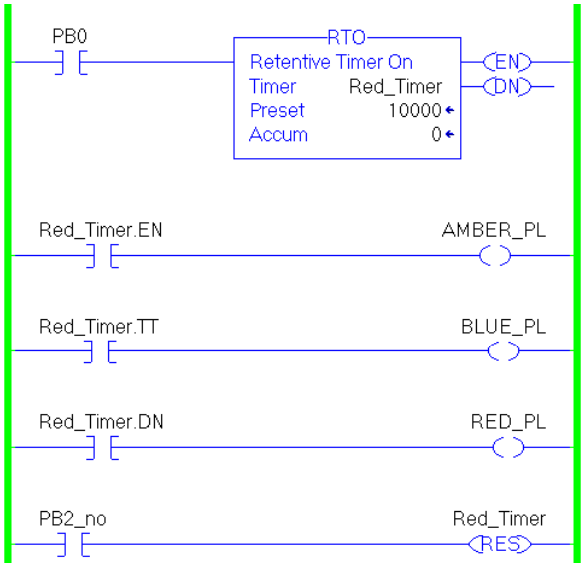
Retentive Timer On (RTO)

- A RESET (RES) instruction must be used to reset a retentive timer
- The RES instruction must have the same tag name as the timer you want to reset:



31

RTO Example – State 1: PB0 not pushed

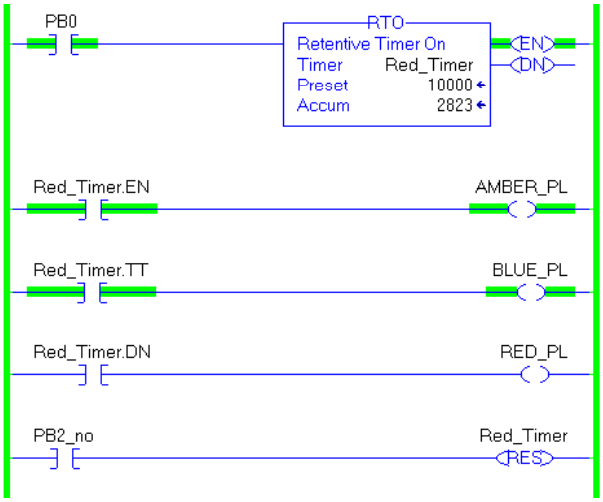


Notes:

PB0 is not being pushed

32

RTO Example – State 2: PB0 pushed



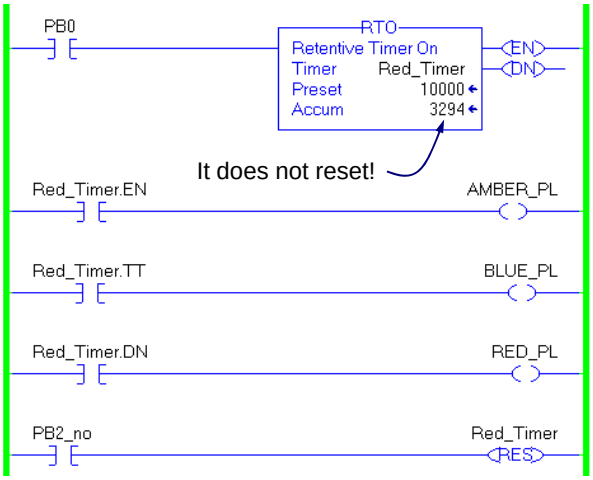
Notes:

PB0 is being pushed.

PB0 has been pushed for 2.823 seconds.

33

RTO Example – State 3: PB0 released



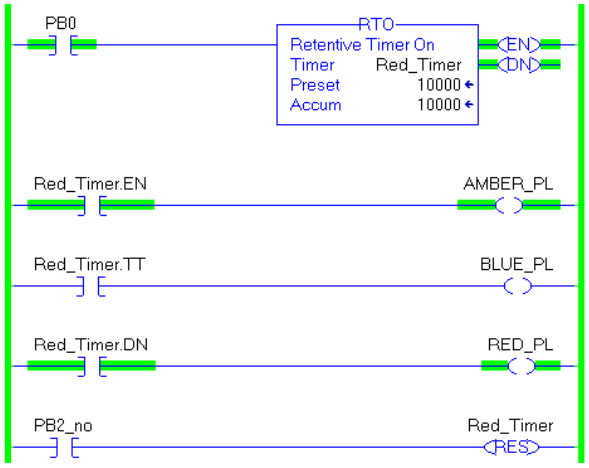
Notes:

PB0 was released before the timer timed out.

PB0 was pushed for 3.294 seconds and then released.

34

RTO Example – State 4: PB0 pushed again



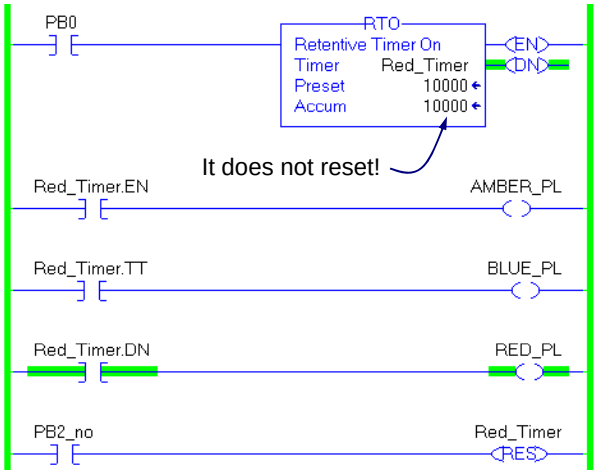
Notes:

PB0 is pushed again until the timer times out.

PB0 is still being pushed.

35

RTO Example – State 5: PB0 Released

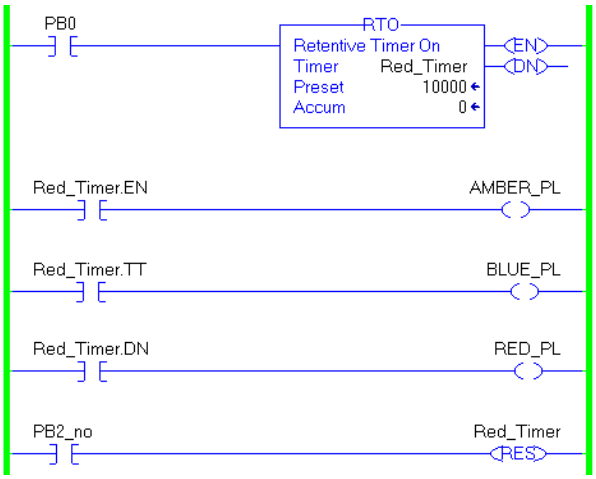


Notes:

PB0 is released after the timer times out.

36

RTO Example – State 6: Timer is Reset



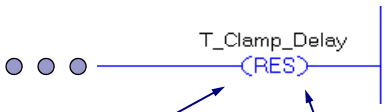
Notes:

PB0 is not being pushed.

PB2_no was pushed and released in order to reset the timer.

37

The RESET (RES) Instruction



A reset instruction must be used with an RTO instruction and all counter instructions

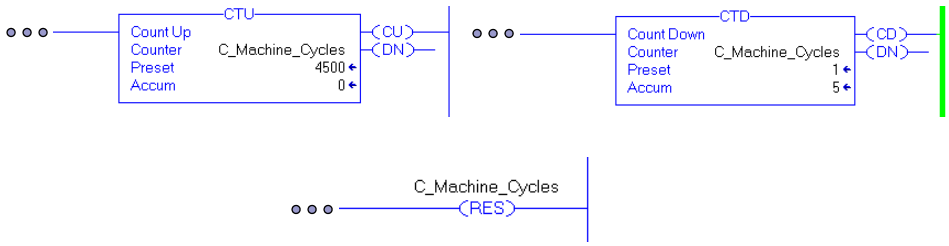
A reset instruction is generally NOT used with TON and TOFF instructions (but it can be)

38

ControlLogix Counters

ControlLogix Counters

- Count Up (CTU)
- Count Down (CTD)
- Counter (and Timer) Reset (RES)

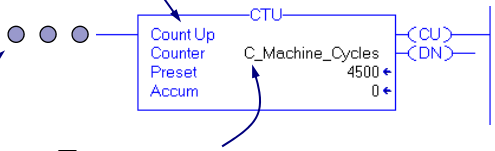


Counters are output instructions!

40

Counter Terminology

Type of counter instruction



Input side of rung intentionally not shown

Tag name

→ If you want to use a counter, you must create a tag of type counter.

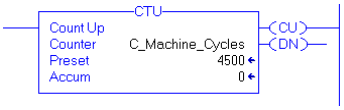
→In this class, the name of all counter tags will start with a "C_"

→ When entering the instruction, this tag must be defined before the preset and accumulated values can be entered.

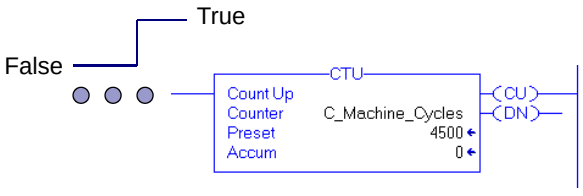
→This tag is typically defined "on the fly"

41

Count Up Counter (CTU)



- When the rung transitions from false-to-true:
  - The Accumulated (ACC) value increments by one



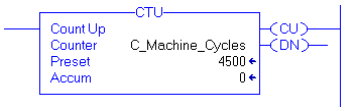
Note:

→ Counters are "transition" based instructions not "level" based instructions.

→With respect to a counter, it does NOT matter how long the rung stays true or false – its only the transition that "counts"

42

Count Up Counter (CTU)



- When the rung transitions from false-to-true:
  - The Accumulated (ACC) value increments by one
- When the Accumulated value = Preset (PRE) value:
  - The done (DN) bit is set to a 1
  - The accumulated value continues to increment on each false-to-true transition
- The reset (RES) instruction will reset the Done bit and reset the Accumulated value to zero

43

Count Up Counter (CTU)

- Preset (PRE) Value
  - Specifies the value the counter must reach before the done (DN) bit turns ON
  - Stored as a 32-bit DINT
  - A DINT stores a 32-bit signed integer ranging from: -2,147,483,648 to +2,147,483,647
- Accumulated (ACC) Value
  - Is the number of false-to-true transitions
  - Is reset to zero when a reset (RES) instruction (of the same counter address) is executed

44

Count Up Counter (CTU)

- **CU** **Count Up Enable Bit**
  > Is set when the rung goes true

  > Is reset when rung goes false or Reset instruction occurs

- **DN** **Count Up Done Bit**
  > Set when ACC value is ≥ PRE value

  > Reset by RES instruction

45

Count Up Counter (CTU)

- **OV** **Count Up Overflow Bit**
  > Is set when ACC > +2,147,483,647

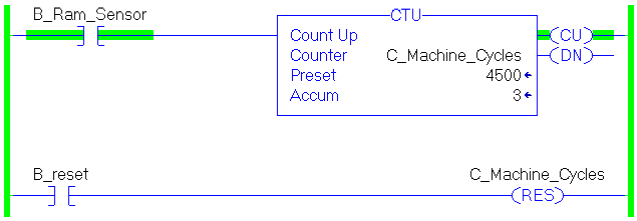
  > Is reset when RESET instruction is executed

Accumulated value keeps incrementing even after the ACC value = PRE value

46

CTU Example

→ Accumulated value = 3, rung is true:



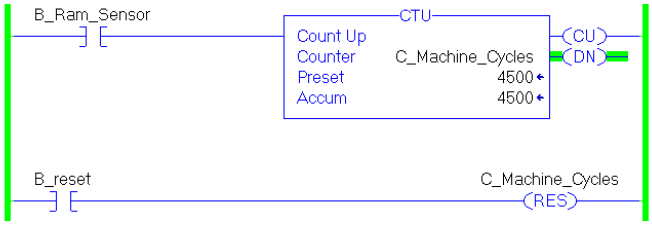
Controller Tags - Template(controller)

Scope: Template | Show... | Show All

Name	Value	Style	Data Type
C_Machine_Cycles	{...}		COUNTER
C_Machine_Cycles.PRE	4500	Decimal	DINT
C_Machine_Cycles.ACC	3	Decimal	DINT
C_Machine_Cycles.CU	1	Decimal	BOOL
C_Machine_Cycles.CD	0	Decimal	BOOL
C_Machine_Cycles.DN	0	Decimal	BOOL
C_Machine_Cycles.OV	0	Decimal	BOOL
C_Machine_Cycles.UN	0	Decimal	BOOL

CTU Example (cont'd)

→ ACC = PRE, the done bit is set:



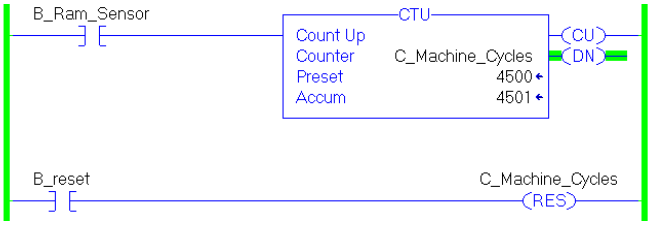
Controller Tags - Template(controller)

Scope: Template | Show... | Show All

Name	Value	Style	Data Type
C_Machine_Cycles	{...}		COUNTER
C_Machine_Cycles.PRE	4500	Decimal	DINT
C_Machine_Cycles.ACC	4500	Decimal	DINT
C_Machine_Cycles.CU	0	Decimal	BOOL
C_Machine_Cycles.CD	0	Decimal	BOOL
C_Machine_Cycles.DN	1	Decimal	BOOL
C_Machine_Cycles.OV	0	Decimal	BOOL
C_Machine_Cycles.UN	0	Decimal	BOOL

CTU Example (cont'd)

→ The accumulated value keeps incrementing each time the rung transitions from false-to-true even after the done bit has been set:



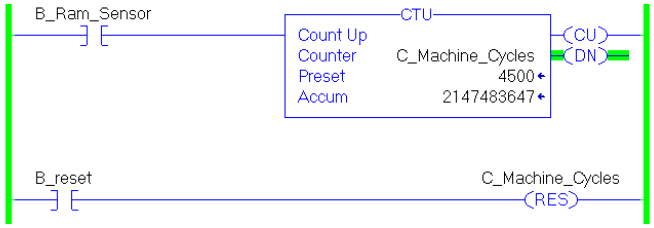
Controller Tags - Template(controller)

Scope: Template | Show... | Show All

Name	Value	Style	Data Type
C_Machine_Cycles	{...}		COUNTER
C_Machine_Cycles.PRE	4500	Decimal	DINT
C_Machine_Cycles.ACC	4501	Decimal	DINT
C_Machine_Cycles.CU	0	Decimal	BOOL
C_Machine_Cycles.CD	0	Decimal	BOOL
C_Machine_Cycles.DN	1	Decimal	BOOL
C_Machine_Cycles.OV	0	Decimal	BOOL
C_Machine_Cycles.UN	0	Decimal	BOOL

CTU Example (cont'd)

→ After many false-to-true transitions, the accumulated value reaches 2,147,483,647:



Since the accumulated value is a 32 bit word, its value ranges from: -2,147,483,648 to +2,147,483,647

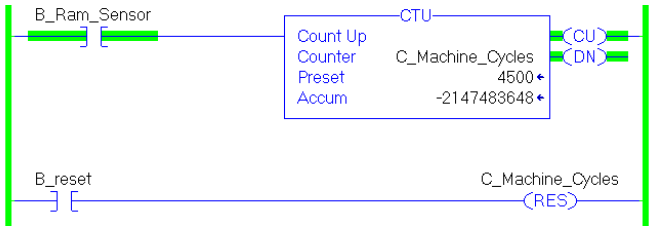
Controller Tags - Template(controller)

Scope: Template | Show... | Show All

Name	Value	Style	Data Type
C_Machine_Cycles	{...}		COUNTER
C_Machine_Cycles.PRE	4500	Decimal	DINT
C_Machine_Cycles.ACC	2147483647	Decimal	DINT
C_Machine_Cycles.CU	0	Decimal	BOOL
C_Machine_Cycles.CD	0	Decimal	BOOL
C_Machine_Cycles.DN	1	Decimal	BOOL
C_Machine_Cycles.OV	0	Decimal	BOOL
C_Machine_Cycles.UN	0	Decimal	BOOL

CTU Example

→ After the next false-to-true transition, the accumulated value becomes -2,147,483,648 and the OV bit is set:



Controller Tags - Template(controller)

Scope: Template | Show... | Show All

Name	Value	Style	Data Type
C_Machine_Cycles	{...}		COUNTER
C_Machine_Cycles.PRE	4500	Decimal	DINT
C_Machine_Cycles.ACC	-2147483648	Decimal	DINT
C_Machine_Cycles.CU	1	Decimal	BOOL
C_Machine_Cycles.CD	0	Decimal	BOOL
C_Machine_Cycles.DN	1	Decimal	BOOL
C_Machine_Cycles.OV	1	Decimal	BOOL
C_Machine_Cycles.UN	0	Decimal	BOOL

Counters are retentive

- The accumulated value of any counter is retained even during a power failure.
- The on/off status of the counter done, overflow, and underflow bits are retentive as well.

52