Up Counter (S_CU)

Figure 12 shows the symbol of S_CU up counter.

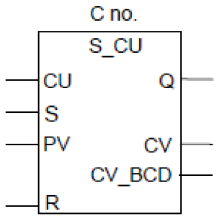

Figure 12: Up counter symbol

Description: S_CU (Up Counter) is preset with the value at input PV if there is a positive edge at input S. The counter is reset if there is a "1" at input R and the count value is then set to zero. The counter is incremented by one if the signal state at input CU changes from "0" to "1" and the value of the counter is less than "999". If the counter is set and if RLO = 1 at the inputs CU, the counter will count accordingly in the next scan cycle, even if there was no change from a positive to a negative edge or vice versa. The signal state at output Q is "1" if the count is greater than zero and "0" if the count is equal to zero.

Table 4 illustrates the function of each parameter of S_CU (Up Counter):

Table 4: S_CU (Up Counter) Parameters

Parameter	Data type	Description
C no.	Counter	Counter identification number; range depends of CPU
CU	Bool	Count up input
S	Bool	Start input for presetting counter
PV	Word	Value for presetting counter Enter counter value as C#<value> in the range from 0 to 999
R	Bool	Reset input
CV	Word	Current counter value, hexadecimal number
CV_BCD	Word	Current counter value, BCD coded
Q	Bool	Status of the counter

The following example illustrates the operation of S_CU (Up Counter), see figure 13:

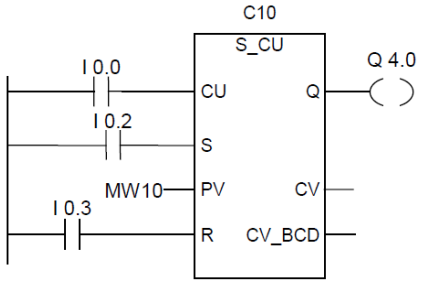

Figure 13: S_CU (Up Counter) Example

If I0.2 changes from "0" to "1", the counter is preset with the value of MW10. If the signal state of I0.0 changes from "0" to "1", the value of counter C10 will be incremented by one - unless the value of C10 is equal to "999". Q4.0 is "1" if C10 is not equal to zero.

Compare Integer (CMP ? I)

Figure 14 shows different comparison operations symbols:

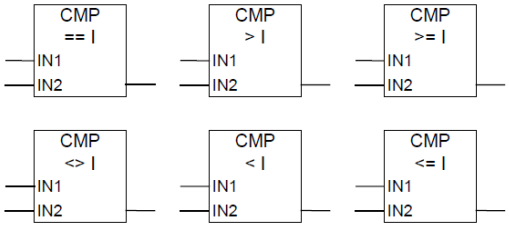

Figure 14: Six comparison operations symbols

Description: CMP ? I (Compare Integer) can be used like a normal contact. It can be located at any position where a normal contact could be placed. IN1 and IN2 are compared according to the type of comparison you choose. If the comparison is true, the RLO of the function is "1". It is linked to the RLO of the whole rung by AND if the box is used in series, or by OR if the box is used in parallel.

IN1 and IN2 are compared according to the type of comparison you choose:

- ==: IN1 is equal to IN2
- <>: IN1 is not equal to IN2
- >: IN1 is greater than IN2
- <: IN1 is less than IN2
- >=: IN1 is greater than or equal to IN2
- <=: IN1 is less than or equal to IN2

Table 5 illustrates the function of each parameter of comparison instructions:

Table 5: Comparison instructions parameters

Parameter	Data type	Description
box input	Bool	Result of the previous logic operation
box output	Bool	Result of the comparison, is only processed further if the RLO at the box input = 1
IN1	INT	First value to compare
IN2	INT	Second value to compare

The following example illustrates the operation of comparison instructions, see figure 15:

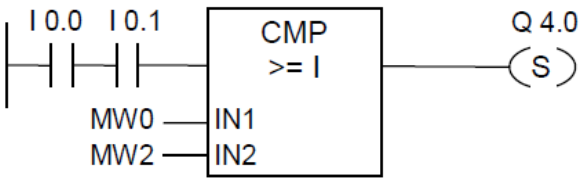

Figure 15: Comparison example

Output Q4.0 is set if the following conditions exist:

- There is a signal state of "1" at inputs I0.0 and at I0.1
- AND MW0 >= MW2.

Assign a Value (MOVE)

Figure 16 shows the symbol of move instruction:

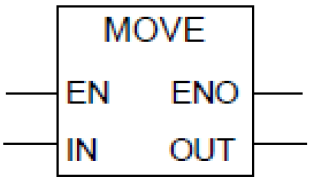

Figure 16: The symbol of move instruction

Description: MOVE (Assign a Value) is activated by the Enable EN Input. The value specified at the IN input is copied to the address specified at the OUT output. ENO has the same logic state as EN. MOVE can copy only BYTE, WORD, or DWORD data objects.

Table 6 illustrates the function of each parameter of MOVE instruction:

Table 6: MOVE instruction parameters

Parameter	Data type	Description
EN	Bool	Enable input
ENO	Bool	Enable output
IN	All elementary data types with a length of 8, 16, or 32 bits	Source value
OUT	All elementary data types with a length of 8, 16, or 32 bits	Destination address

The following example illustrates the operation of MOVE instruction, see figure 17:

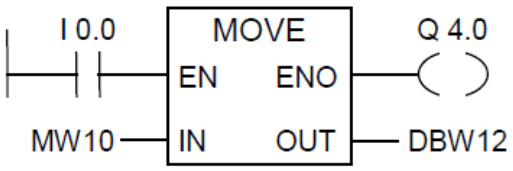

Figure 17: MOVE instruction example

The instruction is executed if I0.0 is "1". The content of MW10 is copied to data word 12 of the currently open DB. Q4.0 is "1" if the instruction is executed.

Equipment

1- Computer with SIMATIC software.
2- Siemens S7-400 PLC
3- LEDs.
4- Switches.
5- Connecting wires.

Procedures

Part 1: Generating a square wave signal

1- Connect one LED to the digital output module.
2- Write down the ladder diagram to generate a square wave signal with a period of 10 seconds.
3- Download the program to the PLC.

Part 2: Using MOVE instruction to define a value for a timer

1- Connect three switches to the digital input module; the first one is used to send a signal to the MOVE instruction, the second to start the timer and the third to reset the timer.
2- Connect one LED to the digital output module.
3- Write down the ladder diagram to switch on the LED after 6 seconds from pressing on the start switch.
4- Download the program to the PLC.

Part 3: Counting how many times a switch was pressed

1- Connect three switches to the digital input module; the first one is used to start the counter, the second to reset the counter and the third is the switch that you are required to know how many times was pressed. Preset the counter with the value of 5.
2- Connect one LED to the digital output module.
3- Write down the ladder diagram to turn the LED on when the counted value reaches 5; the LED will be off for any other value.
4- Download the program to the PLC.

Part 4: Generating a square wave signal for a finite number of cycles

1- Connect one LED to the digital output module.
2- Write down the ladder diagram to generate a square wave signal with 6 cycles each of period of 10 seconds.
3- Download the program to the PLC.

Part 5: Implementing a toggle function

In this example we have one button and a light. First time we press on the button the light will go on and the second time we press the button; in other word we want this button to work exactly like a toggle button.

1- Connect one switch to the digital input module.
2- Connect one LED to the digital output module.
3- Write down the ladder diagram to implement this function using SIMATIC Manager software.
4- Download the program to the PLC.

Part 6: Traffic light

The following represents the sequence of operation of the traffic light:

- Red light on
  Yellow light off
  Green light off
  Delay 20 seconds
- Red light off
  Yellow light on
  Green light off
  Delay 5 seconds
- Red light off
  Yellow light off
  Green light on
  Delay 20 seconds
- Red light off
  Yellow light on
  Green light off
  Delay 5 seconds
- Repeat

1- Draw the hard wiring diagram then connect the required hardware (LEDs, Switches …).
2- Write down the ladder diagram to implement this function using SIMATIC Manager software.
3- Download the program to the PLC.

Discussion and Conclusions

1) Mention four different applications where the counters and timers can be used in PLCs?
2) What is(are) the modification(s) on part 1 to generate a PWM signal with a period of 20 seconds and a duty cycle of 70%?
3) What is(are) the modification(s) on part 3 to turn the LED on for any value greater than 5?
4) What is(are) the modification(s) on part 3 to turn the LED on for any value except 5?

5) Refer to SIMATIC Ladder Logic (LAD) for S7-300 and S7-400 Programming Reference Manual and illustrate the function of S_OFFDT Off-Delay S5 Timer? Compare it with S_ODT On-Delay S5 Timer?