

Sequence Diagram

Design Guidelines

S7-1500

Table of Contents

1. GENERAL INFORMATION 3

2. MULTIPLE COMMANDS..... 5

3. BRANCHING..... 5

4. TRANSFER SEQUENCES HAVING MULTIPLE SOURCES/DESTINATIONS..... 7

5. OPERATOR RESPONSE..... 10

6. REQUEST OF ANOTHER UNIT 11

7. PID CONTROLLERS..... 11

8. ABNORMAL CONDITIONS 12

9. REVISION HISTORY 13

1. General Information

The function charts are called sequence diagrams and they are "drawn" with an Excel spreadsheet. Using a spreadsheet opens up the possibility of automatic code generation. That is a program can be written to automatically generate the PLC code as a library that can be imported into the programming software.

The general layout of a sequence is shown in Figure 1 with the various parts identified.

- A tag is used to track the step number and its variable/tag/symbol is shown next to the initial (zero) step.
- The step number for a particular step is shown within a box.
- Each transition is identified by a text statement describing the transition and by the logic that implements the transition. The description goes above the horizontal line and the logic with the variables/tags/symbols goes below the horizontal line. Use “.and.”; “.or.” between multiple transition parts.
- The action of each step has three parts:
 - The first part describes the action and contains the message displayed to the operator when the step is active.
 - The second part contains the commands (open/close valve, start/stop motor, etc.) Each different command occupies one row. The case of multiple commands for a step is shown in Figure 2.
 - The third part contains the message number. The message number for a particular step must be unique for the unit, so the message number is the step number plus an offset. To allow for steps to be added to a sequence in the future, message numbers are allocated for a unit in blocks of 20, 40, 60, and so on. If the first sequence in a unit has considerably less than 20 steps, then the message numbers for the second sequence should start at 21 and be equal to the step number plus 20. If the first sequence has between 10 and 30 steps, then the message numbers for the second sequence should start at 41.
- Every third step (average) should be a spare step. Spare steps have a transition condition of "1" shown below the horizontal line.
- When the sequence reaches the last step, it remains in the last step until another sequence starts, which is indicated by the transition out of the last step. The example sequence in Figure 1 initiates another sequence in its last step.

The Soda Ash unit sequence is provided as an example. Use the same format (column widths, etc.)

Other things about the sequence diagrams:

- Each sequence is in a separate worksheet (with the same name as the sequence) and in the same order as they are listed in narrative. Headers and footers are on print-outs and should match worksheet name.
- Follow the standard variable names consistent with the processor programming standards.
- In general, do not combine steps unless the narrative so states.
- No "-" in tags. For example, an equipment tagged as P-1000 is referenced as "P1000."

- Shutdown – Do not combine stopping of pumps, agitator, valves, etc. Each command should be in a separate step.
- The last sheet is named “Abnormal Conditions” and documents the abnormal conditions that cause a sequence auto-start and alarms.

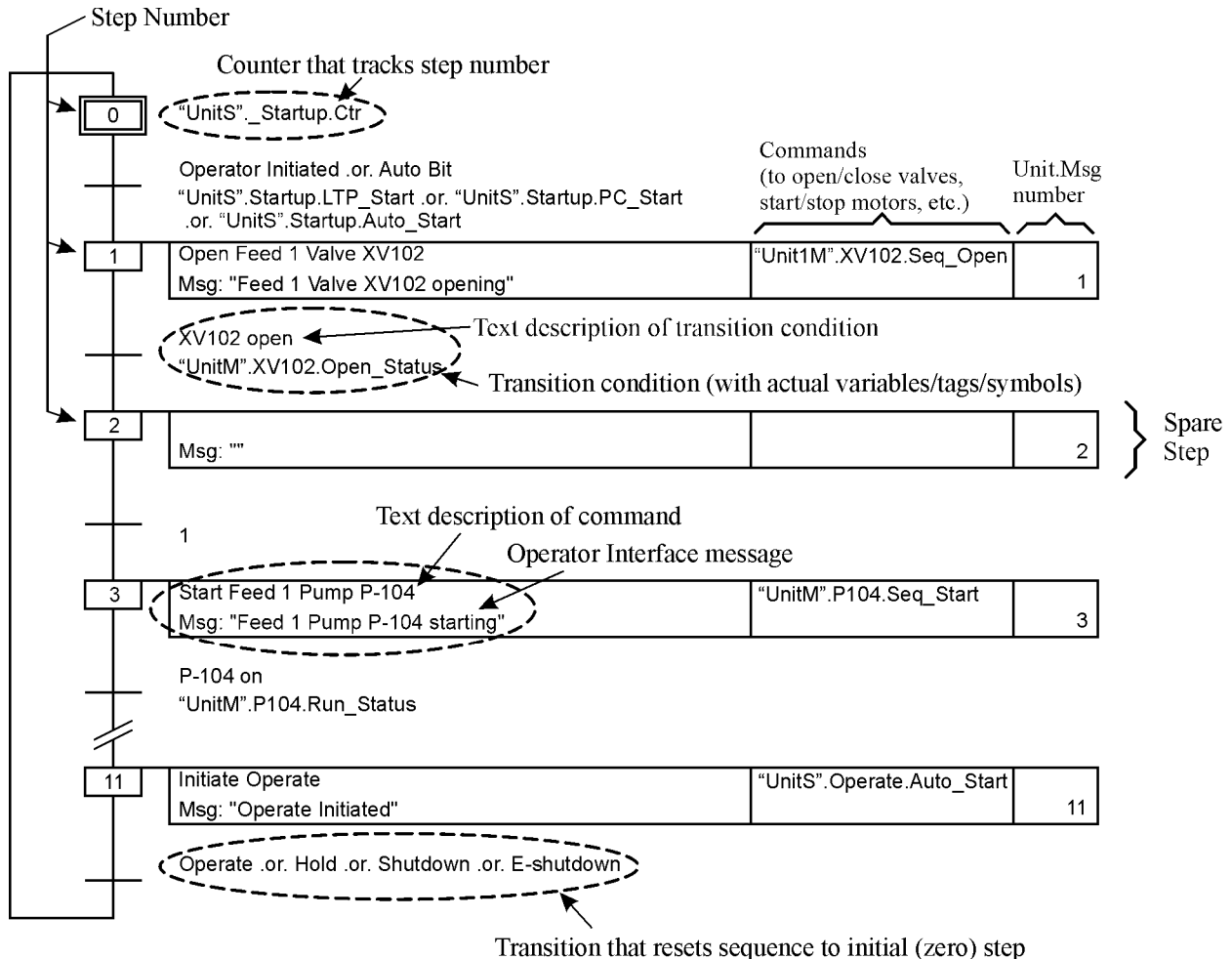


Figure 1. General format of sequence diagram.

The E-Shutdown sequence operates differently from the others. Since its purpose is to immediately shut down the process, the transitions do not wait for any confirmation that the commands have been executed. The next-to-last step needs to have a 15 second timer to pause before reaching last step. This delay allows the equipment that was stopped/closed to complete the action before the last step of the sequence is reached at which the operator is granted the privilege to manipulate the devices.

The formats for multiple commands, branching, soliciting operator input and sending a request to another unit are described in the following sections.

2. Multiple Commands

When multiple commands must be executed on a step, the command box is expanded as shown in Figure 2. Note that the individual commands are shown on successive rows.

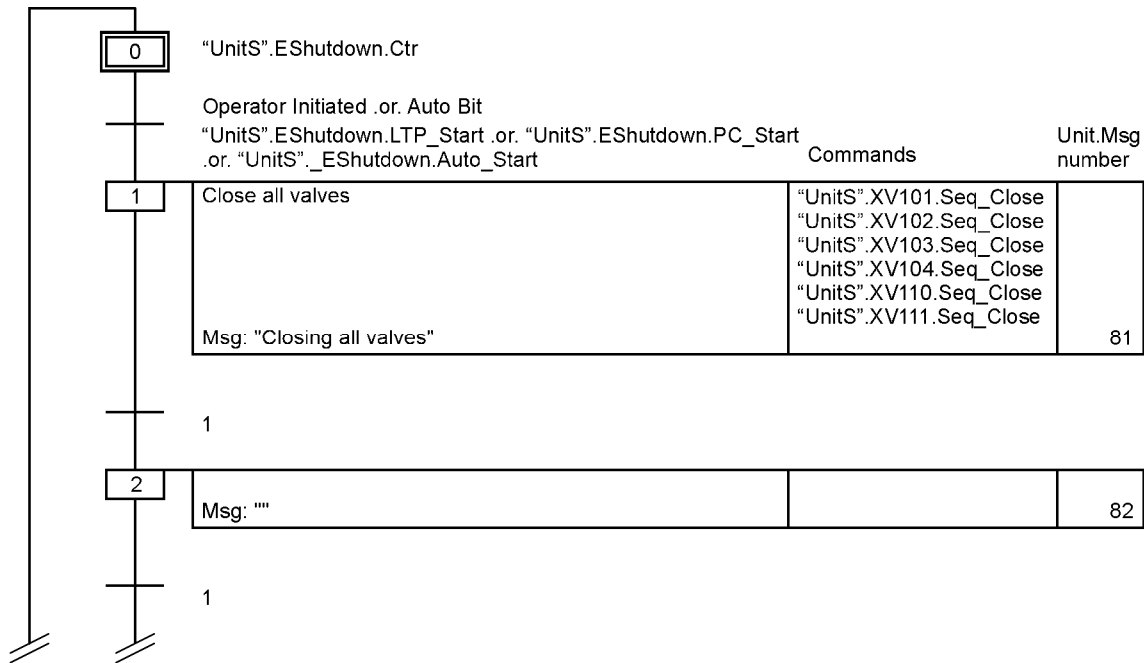


Figure 2. Multiple commands for a step.

3. Branching

Only "or" branching can be accommodated. A skip of sequence steps is done as shown in Figure 3 and a general branch is shown in Figure 4. Note the step numbering in Figure 4. The step numbers of the branch steps are incorporated into the step numbers of the sequence. If the sequence needs to branch immediately after it is started, insert a spare step at the start. Do not attempt to incorporate the branch logic in with the start logic.

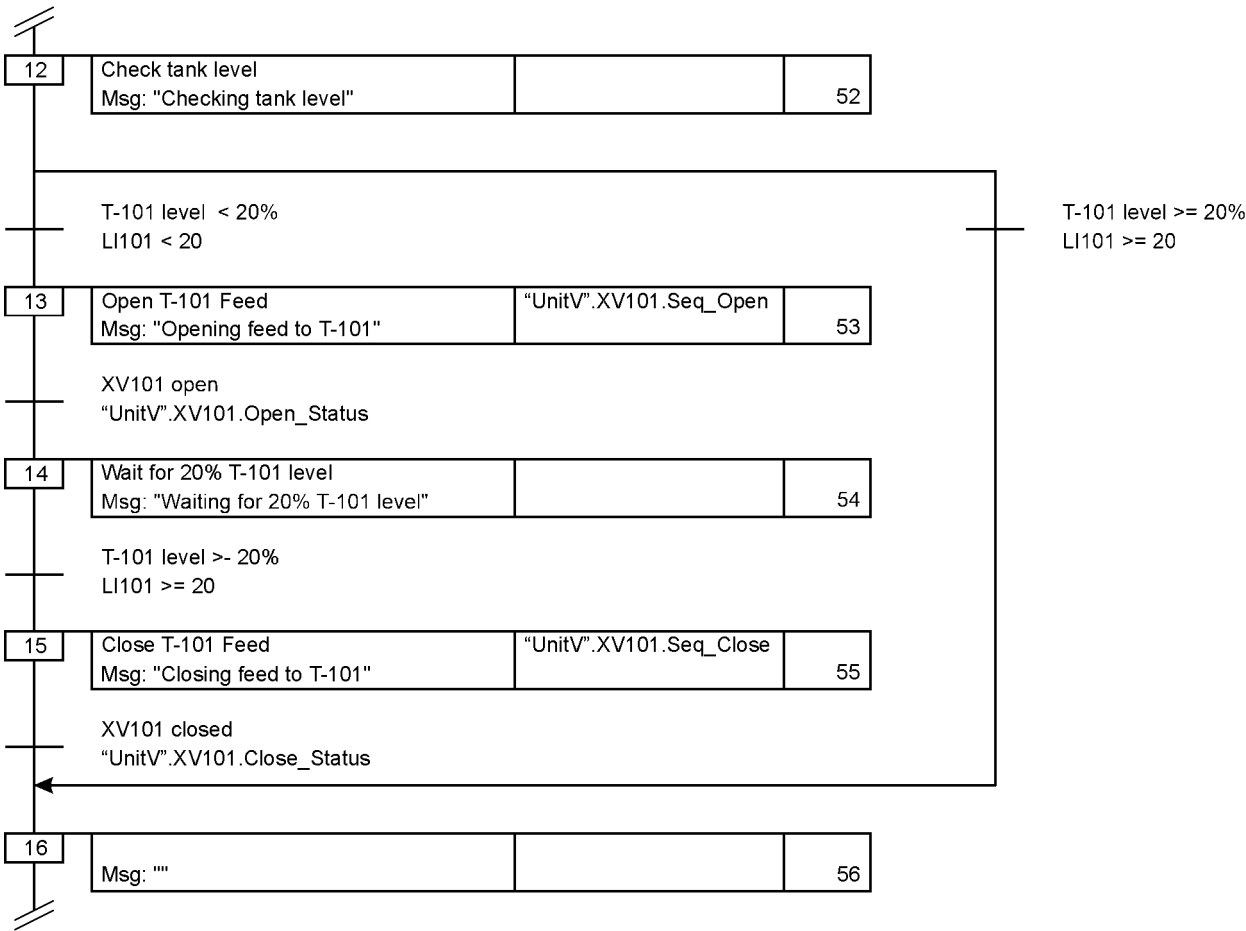


Figure 3. Sequence skip.

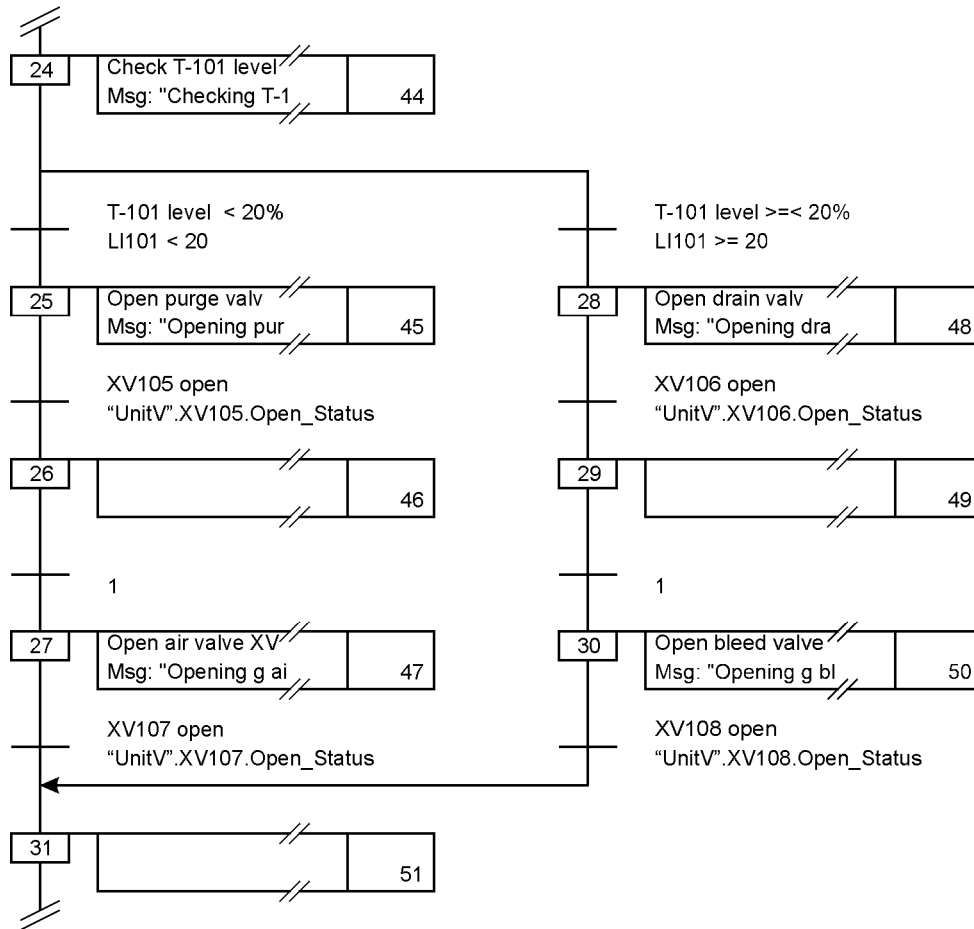


Figure 4. Sequence branching.

4. Transfer Sequences Having Multiple Sources/Destinations

For sequences that transfer material

1. from one source to multiple destinations, or
2. from multiple sources to one destination

formulate one sequence and use logic for those steps where a different valve is opened/closed. These sequences generally share most of the steps. There are two ways to handle these sequences: (1) one start button for the transfer, with the source/destination selected with a separate operator response, or (2) separate start buttons, one for each source/destination. The second method is used when there is only one source or only one destination. If there are multiple sources and destinations, the first method is used.

An example for the first method is shown in Figure 5. There is one start. Note that an operator start is ignored if a transfer is in progress. The fifth step opens the appropriate valve, depending on the destination tank selected by the operator, presumably from an operator response sometime during steps 1 – 4. The operator has selected a particular tank by pressing the appropriate button icon, which placed a value (1 to 3) in the Unit_Dest integer.

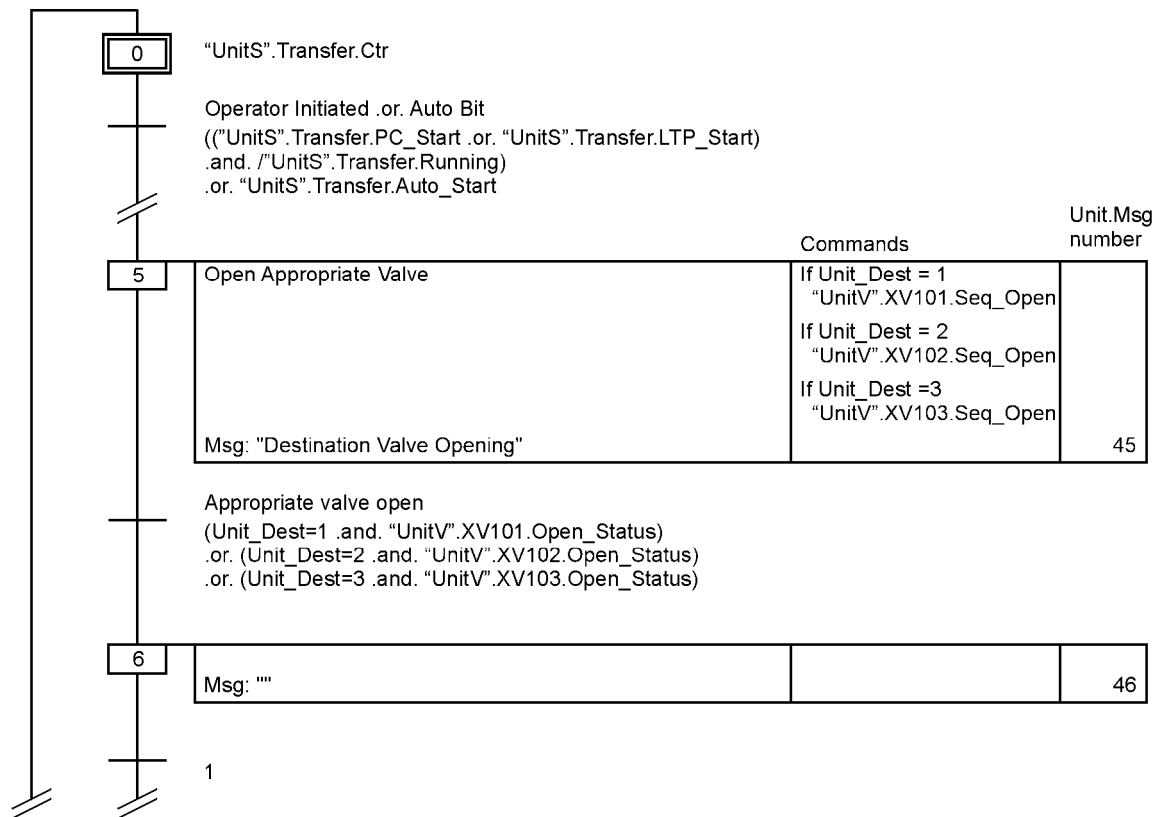


Figure 5. Transfer sequence with one start.

An example for the second method is shown in Figure 6. There is one start for each tank destination (or source). Note that an operator start is ignored if any transfer is in progress. There is a “running” indication for each transfer, which is used to determine the proper command and transition indication for step 5.

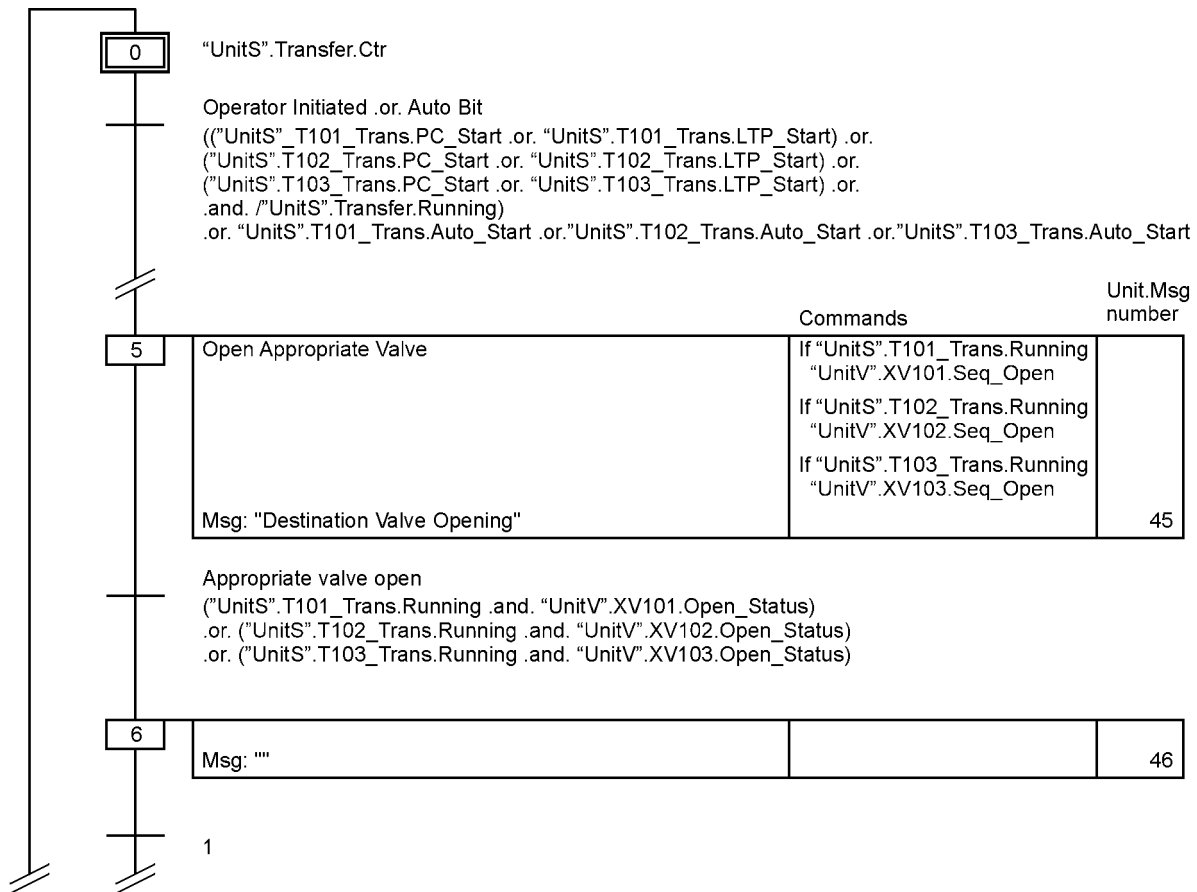


Figure 6. Transfer sequence with multiple starts.

5. Operator Response

When soliciting operator input, generally one will either solicit a value or one of a number of choices (for example, select source tank). For the first cases, the operator will need to enter a number. The sequence clears the value and then sets a Boolean that is the signal to the OI to display the prompt and the value field. When the operator has entered a value, the sequence transitions to the next step. The implementation is shown in Figure 7.

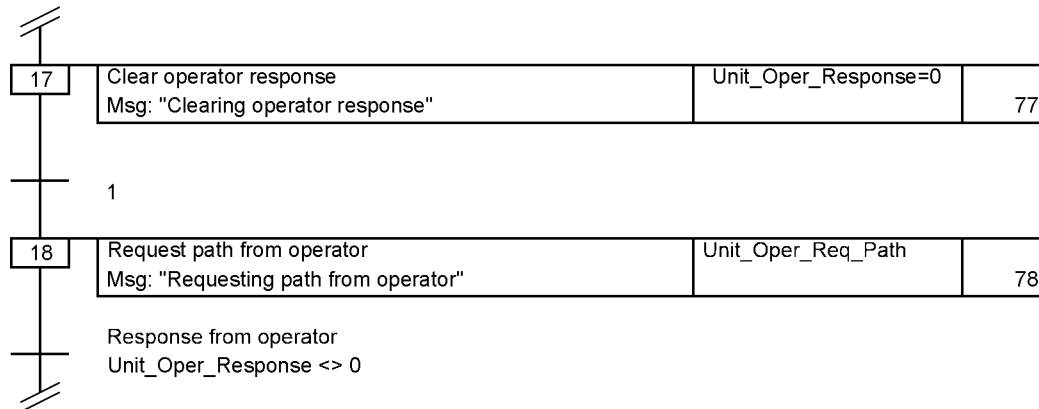


Figure 7. Soliciting and receiving operator value response.

When asking for the operator to verify a selection (for example, that a proper source tank is currently selected), the sequence sets a Boolean that is the signal to the OI to display the prompt and make the "Verify" button visible. The animation for the verify button must be configured so that when the operator presses the verify button, the Unit_Oper_Resp_Verify Boolean is turned on (**not set or reset**), which causes the sequence to transition to the next step. The implementation is shown in Figure 8.

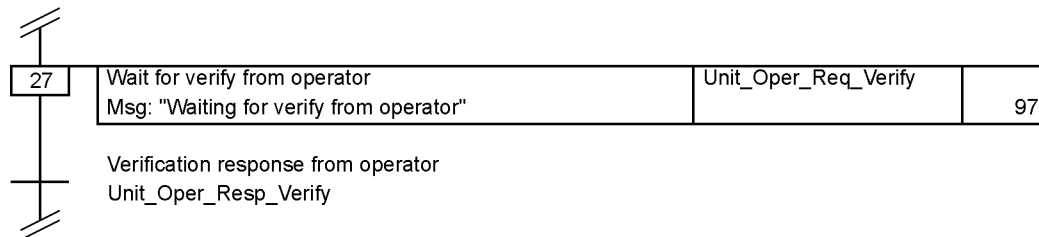


Figure 8. Soliciting and receiving operator verification response.

6. Request of Another Unit

When coordination among units is needed, one unit needs to send a request to the other unit and then wait for acknowledgement that the action has been accomplished. The implementation of this is shown in Figure 9. Whether or not the other unit is implemented in the same PLC makes no difference on the sequence diagram. If the other unit is in another PLC, then there will be some communication occurring in the background to send the request to the other unit and to receive the response. Typically, there is more than one piece of information that must be communicated on a periodic basis and the request/response is just a piece of it.

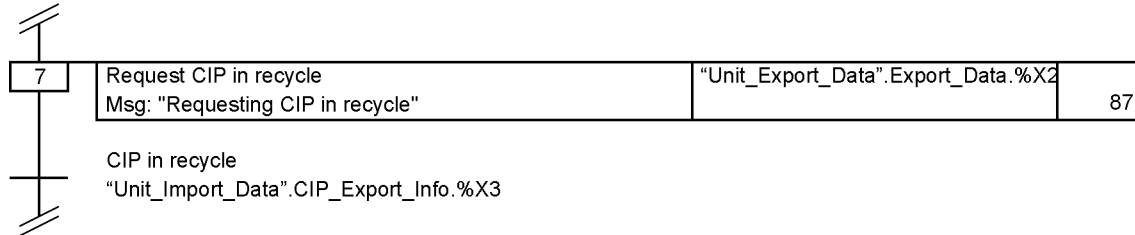


Figure 9. Sending request to another unit.

7. PID Controllers

Sequences control the following parameters of a PID loop:

- Mode

- Manual output (when in manual mode)

- Setpoint (when in automatic mode)

Sample commands to set the mode and the setpoint or output are shown in Figure 10. Note that the transition condition is "1." The particular commands in Figure 10 are for a ControlLogix PID block.

Since the ladder logic will set these parameters, it is not necessary to check that they are actually set. However, one may wish to check that the actual controller output matches the manual output when setting the mode to manual to verify that the PID controller actually executed and set the output before transitioning to the next step. If doing this check, the transition out of step 21 in Figure 10 would be "FIC101.OUT=0."

Flow totalization is used when one needs to know how much material is being transferred. The only sequence command is resetting the totalizer. In this case the command is the totalizer tag set equal to zero (for example, FQI101 = 0.0).

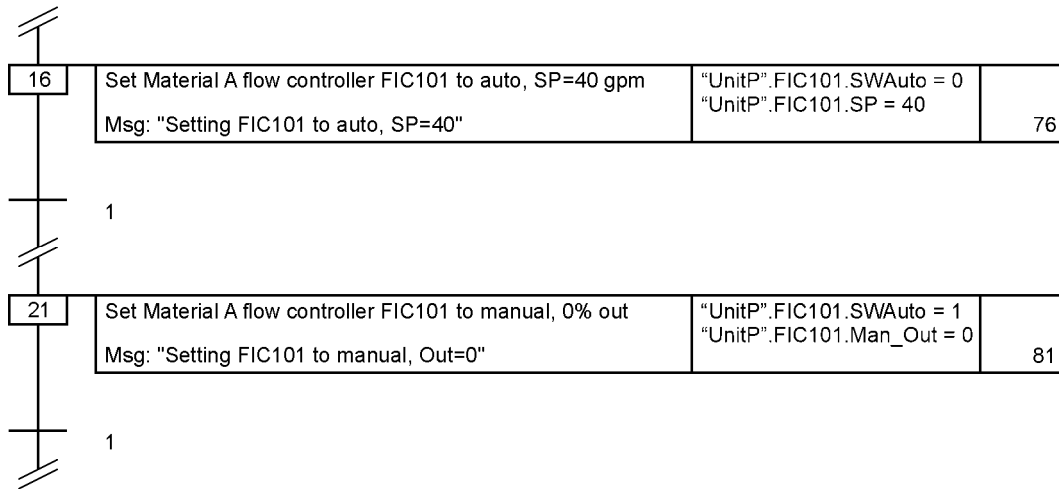


Figure 10. Sample PID loop commands.

8. Abnormal Conditions

The last sheet documents the abnormal conditions that cause the sequence auto-starts and alarms. The information is arranged in a table as shown below. The first column describes the condition and the second column shows the more detailed symbols and logic to detect the condition. The fourth column indicates the consequence of the action (usually the name of a sequence that will be started or “Alarm”). The third column is the Boolean turned on as a consequence of the condition. When the action is an alarm, the request bit is the output of a rung that implements the condition. When the action is a sequence, the request bit is a sequence auto-start bit and the condition is one of the triggers for the auto-start bit.

Description	Condition	Request Bit	Action
L-6100 fails for 1 sec.	L6100_Any_Fail for L6100_FailSdwn_Tmr (1 sec)	Soda_Ash_Shutdown.Auto_Start	Shutdown
LSL6100 indicates low bin level for 2 sec.	LSL6100 for L6100_Level_Alarm_Tmr (2sec)	LSL6100_Alarm	Alarm

9. Revision History

Rev 0	15Apr2005	First release
Rev 1	6Sep2007	Revised to use UDT of CLogix Rev 16
Rev 2	15Jan2008	Revised to add second method for transfer sequences. Use status from device UDTs instead of valve LS's or motor aux input.
Rev 3	14Jan2024	Started with Rev 4 of ControlLogix version. Changed tags to match S7 naming convention