Module 4

Programmable Logic Control Systems

Lesson 20

Formal Modelling of Sequence Control Specifications and Structured RLL Programming

Instructional Objectives

After learning the lesson students should be able to

A. Describe motivations for formal modelling in the design of sequence control programs for an industrial control problem.

B. Describe the major steps in the design of a sequence control program for an industrial control problem.

C. Develop a Finite State machine model for simple industrial control problems

D. Develop a sequence control program for a Finite State Machine model

Motivation for Formal Modelling

To be convinced about the need for developing formal models for control problems in a systematic way, consider the example shown below.

Industrial Logic Control Example Revisited

Consider the industrial logic control example of the stamping process presented in Lesson 18. Let us recall the description of the process given in the lesson. For convenience of reference the description and a pictorial representation of the process is reproduced below.

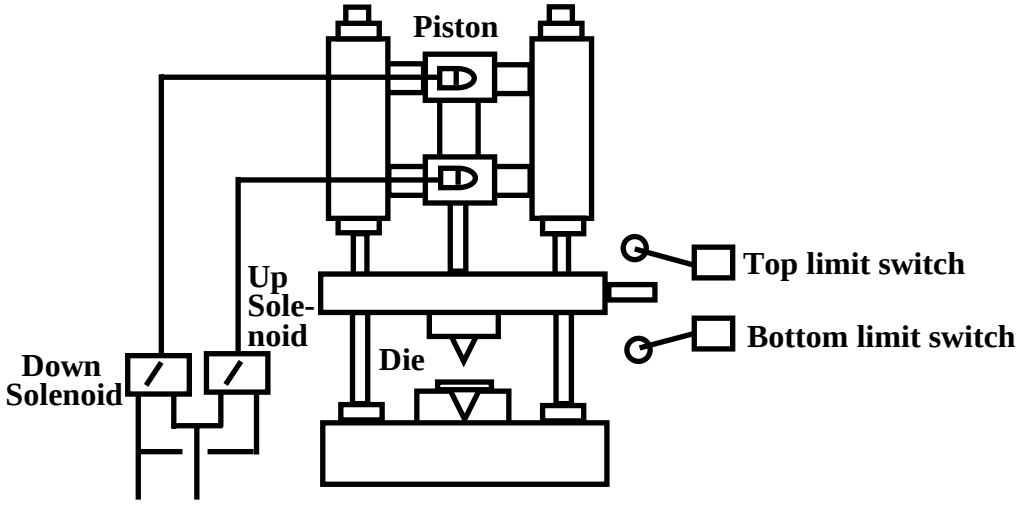

Fig 20.1 An Industrial Logic Control Example

Linguistic description of the industrial stamping process

The die stamping process is shown in figure 20.1. This process consists of a metal stamping die fixed to the end of a piston. The piston is extended to stamp a work piece and retracted to allow the work piece to be removed. The process has 2 actuators: an up solenoid and a down solenoid, which respectively control the electro-hydraulic direction control valves for the extension and retraction of the stamping piston and die. The process also has 2 sensors: an upper limit switch

that indicates when the piston is fully retracted and a lower limit switch that indicates when the piston is fully extended. Lastly, the process has a master switch which is used to start the process and to shut it down.
The control computer for the process has 3 inputs (2 from the limit sensors and 1 from the master switch) and controls 2 outputs (1 to each actuator solenoid).
The desired control algorithm for the process is simply as follows. When the master switch is turned on, the die-stamping piston is to reciprocate between the extended and retracted positions, stamping parts that have been placed in the extended piston machine. When the master switch is switched off, the piston is to return to a shutdown configuration with the actuators off and the piston fully retracted.

At first, let us consider an Relay Ladder Logic program that has been written directly from the linguistic description and assess it for suitability of operations.

The first version of sequence control program for the industrial stamping process

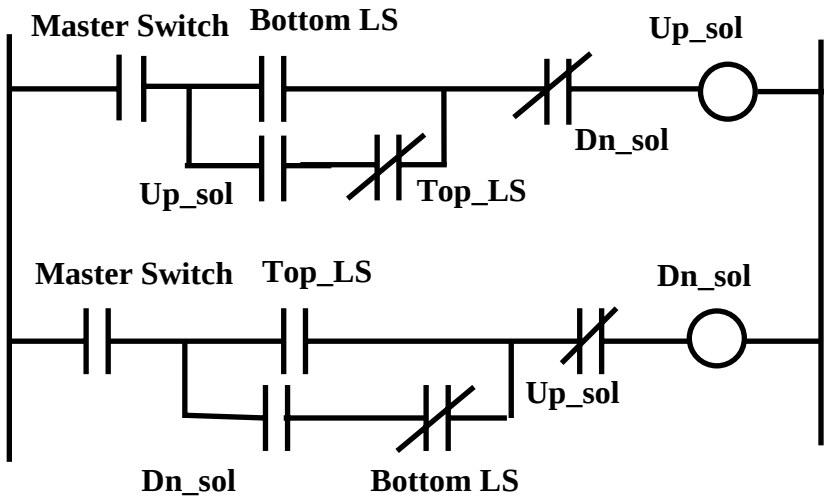

Fig. 20.2 An RLL program for the industrial stamping process

A hastily constructed RLL program for the above process may look like the one given in Figure 20.2. The above program logic indicates that the Up solenoid output becomes activated when the Master Switch is on and the bottom Limit Switch is on. Also there is interlock provided, so that when the Down solenoid is on, the Up solenoid cannot be on. Further, once the Up solenoid is on, the output is latched by an auxiliary contact, so that it remains on till the bottom LS is made on, when it turns off. A similar logic has been implemented for the activation of the Down solenoid.
However, on closer examination, several problems may be discovered with the above program. Some of these are discussed below.

- For example, there is no provision for a Master stop switch to stop the press from stopping in an emergency, except by turning the Master Switch off. This would indeed stop the process, however, if the press stops midway, both bottom and top limit switches would be off. Now the process would not start, even if the Master switch is turned on again. Therefore, either a manual jogging control needs to be provided, so that the

operator can return the piston to the up position by manually operating the hydraulics, or special auto mode logic should be designed to perform this.

- As a second example, note that this process does not have a part detect sensor. This implies that the moment the Master switch is on the press would start going up and down at its own travel speed, regardless of whether a part has been placed for pressing or not. Apart from wastage of energy, this could be safety hazard for an operator who has to place the part on the machine between the interval of a cycle of operation.

The above discussion clearly indicates the need for a systematic approach towards the development of RLL programs for industrial logic control problems. This is all the more true since industrial process control is critical application domain where control errors can lead to loss of production or operator safety. Therefore, in this chapter, we discuss a systematic approach towards the design of RLL programs.

Point to Ponder: 1

A. *Before going through the rest of the system description attempt to modify the RLL program shown in Fig. 20.2 to ensure that the process always moves to the up position first, and then from there, resumes its press cycle*

B. *Before going through the rest of the system description attempt to modify the RLL program shown in Fig. 20.2 to include a part detect sensor*

C. *Attempt to merge the above two solutions so that the new program is free from both the above problems*

Steps in Sequence Control Design

The approach to Sequence control Design presented below is derived from the basic principles of modern software engineering practices. An interested student is referred to standard software engineering references for a more detailed discussion on these.

A brief description of the steps follows:

The broad initial steps are

- Requirement Analysis
- For Modelling of the process
- Design
- Design verification and validation

A. Requirements Analysis

This is a very important initial step. Errors in this step may be discovered very late during commissioning of the control system and can result in loss of significant amount of man-hours. Note that, since the control engineer is not necessarily an expert in plant operations. Therefore it is quite natural that he may not understand requirements and characteristics of process operations fully, without conscious and significant effort.

Therefore, this phase is to be carried out in close consultation with the plant engineers of the user organization. It basically involves studying the system behavioral aspects of the system to:

i) identify the feedback inputs from sensors and the external operator inputs from Man Machine Interface (MMI),

ii) identify the controller outputs to actuators and the outputs to the indicators/ MMI,

iii) study and document the sequence of actions and events under the various operational 'modes'. This should not only indicate the 'normal' modes, but also 'emergency', 'start-up', 'shut-down' and other special modes operation that may not be occurring very frequently but important all the same.

iv) study the effects of possible failures in the process as well as sensors and identify possible means of recovery. Although a failure may be very rare in occurrence, unless they are considered, some of them may have devastating consequences when they occur. Failures in a process can occur in the sensors, the actuators, or some time the process equipment itself.

v) examine need for manual override control, additional sensors, indicators, and alarms for maintenance, operational efficiency or safety.

Initially, often the above information may be collected in the form of informal linguistic descriptions by direct discussion with plant personnel. Further, it must be remembered that software development is often an iterative process and therefore, the above analysis steps may have to be repeated a number of times for an actual design exercise.

Requirements Analysis for the Stamping Process

As given above the first step in requirement analysis is to identify the Process Control Inputs that need to be sensed and the Process Control outputs that need to be actuated.

Process Control Inputs

- *Part sensor*: A position switch that detects when a part has been placed. In cases where proper positioning of the part can take time. One may also consider using manual switch to be operated by the operator once he is satisfied that the part is properly placed and ready to be stamped. Here an automated part detect sensor has been assumed.
- *Auto PB* : A push button that indicates that machine is ready to stamp parts one after the other in the 'automatic mode'
- *Stop PB*: A push button that the operator can use to stop the machine any time during the time that the piston is moving down. This is needed to avoid stamping a part if any last second error is discovered by the operator regarding, say, the placement of the part.
- *Reset PB*: In case the piston has been stopped due to some error condition, it is desired that the operator explicitly presses this push button to indicate that the error has been taken care of, and the machine is ready to return to stamping in the auto mode.
- *Bottom LS*: This sensor indicates when the piston has reached the bottom position.
- *Top LS*: This sensor indicates when the piston has reached the top position.

Process Control Outputs

- *Up Solenoid*: Control output that drives the Up solenoid of the electro-hydraulic direction control valve which in turn drives the piston up.
- *Down Solenoid*: Control output that drives the Up solenoid of the electro-hydraulic direction control valve which in turn drives the piston up.
- *Auto Mode Indicator*: An indicator lamp that indicates that the machine is in 'Auto' mode.
- *Part Hold*: A gripping actuator that holds the part firmly to avoid movements during stamping

Point to Ponder: 2

A. *Can we use corresponding toggle switches, instead for the Auto, Stop and Reset PBs? What difficulties may be encountered?*

B. *For the limit switches, and the part detect sensor, would you prefer mechanical switches over photo switches for this application? Justify.*

C. *Propose at least one additional each of sensors, indicators and actuators for the above application and mention their benefits.*

Sequence of Events and Actions

A. The "Auto" PB turns the Auto Indicator Lamp on

B. When a part is detected, the press ram advances down to the bottom limit switch

C. The press then retracts up to top limit switch and stops

D. A "Stop" PB, if pressed, stops the press only when it is going down

E. If the "Stop" PB is pressed, the "Reset" PB must be pressed before the "Auto" PB can be pressed

F. After retracting, the press waits till the part is removed and the next part is detected

Point to Ponder: 3

A. *Note that in step F above, it is important to detect that the part is removed. What would happen, if this is not detected?*

B. *What would happen if after Stop PB is pressed, Reset PB and Auto PB are pressed in that sequence, even if the piston has not been taken to the top position manually?*

Effects of failures

Among possible failures for this process are drops in hydraulic pump pressure, failures in top and bottom limit switches etc. The exact nature of the failures and its impact need to be understood for the application context. While this should be done, it requires domain knowledge for the

control engineer. This is therefore not attempted here for reasons of conciseness. However, the learner is encouraged to augment the control logic with additional logic to detect such failures rapidly and initiate activities for fault tolerance.

Point to Ponder: 4

A. *What would happen in the process controlled by the program shown in Fig. 20.2 if top/bottom Limit switch is stuck closed/open?*

B. *What would happen in the process controlled by the program shown in Fig. 20.2 if the hydraulic pump pressure becomes too low to move the RAM?*

Formal process modelling

Once the requirements have been ascertained, formal process modelling can be undertaken. In this step the informal linguistic descriptions have to be rigorously checked for ambiguity, inconsistency or incompleteness. This is best achieved by converting linguistic descriptions into formal process models. Initially one may use intermediate forms like list of operations, flowchart etc. Eventually and before developing the control programs, these are to be converted into mathematically unambiguous and consistent description using a formal modeling framework such as a Finite State Machine (FSM). It is the experience of practical engineers that modelling paradigms that can be represented pictorially are particularly suited to human beings.
For formal modelling, a process often can be viewed as a Discrete Event System (DES). Many formalisms for creating timed or untimed models of DESs exist (e.g. Petri Nets). A detailed description of these is beyond the scope of this lesson. An interested reader is referred to literature on real-time systems for a more detailed discussion on these. In this lesson, it is shown how the process dynamics can be modeled as a Finite State Machine. The following facts which are very important to modelling are mentioned.

A. An FSM is a simple formalism for DES in which, at any time, the system exists in any one discrete-state of a finite set of such states.

B. A state is basically an assignment of values to the set of variables of the system. For a discrete event system, the process variables are assumed to take only a finite set of values. For example, the limit switches can only take two values each, namely, either ON or OFF.

C. Further, the set of the process variables have to be chosen in such a manner that, the future behaviour of the process would be determined solely based on the values of the chosen set of variables at the present time. For example, for the stamping press example, the set of process variables would include the values for the Top and Bottom limit switches. However, based only on these the behaviour of the process cannot be determined. This is because from these it cannot be determined whether the piston is moving up or moving down. Therefore one would have to add the state of the motion as a state variable. The set of values that this variable can take are: ‘going up’, ‘going down’ and ‘stationary’. In this case, one would also have to add another variable, namely the value of the part detect sensor output to be able to distinguish between the behavioral difference between the case when it is ON and when it is OFF, when the piston is at the top position.

D. Some of the state variables may be measured physically with sensors. Others may not be.

E. The choice of state variables can be subjective and different designers might pick others. The choice also depends on the nature of control actions that one would like to take. Thus, the choice of states is specific to the machine and its operation.

F. During its life cycle, the process moves from state to state over time. Thus it spends most of its time in the states. Occasionally however, it makes a transition from one state to another. The occurrence of a transition depends entirely on the occurrence of discrete events. Such events are names given to conditions involving states, some of which may change due to external factors, such as operator inputs, or due to internal factors, such passage of time. On occurrence of such an event, mechanisms causing state transitions are triggered. State transitions or events are generally considered instantaneous and thus, the system spends time only in the various states. State variables are modified by the occurrence of transitions. In fact, it is this change in the values of the state variables, which is taken to be a transition from one state to another.

G. All possible combinations of state variables may not be valid state assignments for a system. In other words, the system can have only some of all the possible combinations of state variables. These are said to be the combinations that are 'reachable' by the system.

H. One of the states is generally taken to be an 'initial state'. The system, when it starts its life cycle, that is, at the time from which its behaviour is described by the State Diagram, is supposed to be at the initial state.

I. At each state, a set of outputs are exercised. This is described by an output table, where the values for each output variable at each of the states is shown. The output table for the stamping press is shown in Figure 20.x.

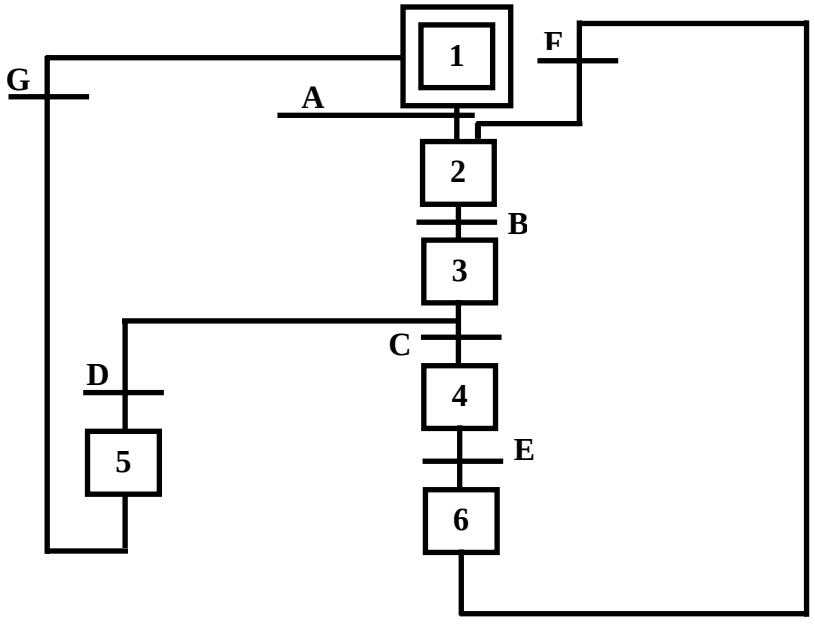

Figure 20.3 The State Diagram for the industrial stamping press

State No. / O/P	1	2	3	4	5	6
Auto Indicator	0	1	1	1	0	1
Part Hold	0	0	1	1	0	0
Up Sol	0	0	0	1	0	0
Down Sol	0	0	1	0	0	0

Figure 20.4 The Output Table for the industrial stamping press

From the State Diagram in Fig. 20.3, the following may be noted.

A. The states are marked numerically using the numerals 1-6. At each state, one has a unique set of values for the process state variables. For example, in state 3, both the limit switches are OFF, the part detect switch in ON, the motion state variable of the piston is 'going down'. State 1 is marked as the initial state with a double square.

B. The transitions are marked alphabetically using the letters A-G. Each transition has an associated condition under which it occurs. For example, the condition for transition B may be simply stated as "when it in state 2 and part detect goes ON'. Note that the condition described by the phrase 'in state 2' can be further explained in terms of the values of the state variable corresponding to state 2.

C. The outputs exercised at each state are described by the Output Table in Fig. 20.4.

Design of RLL Program

Based on the formal model, the sequence control program can be developed systematically. In fact, one of the main advantages of formal process modelling is that the process of development of the control program becomes mechanical. Thus it can be done quickly and with a much reduced chance of error. In the case of FSM models one has to write the RLL program such that over the scan cycles it executes the state machine itself. The outputs of the state machine go to the process, and since the state machine is nothing but a behavioral model for the process, the process also executes the transitions of the machine, as desired. The realization of a state machine by an RLL program involves computation of the process transitions, in terms of the inputs and the internal state variables of the program followed by computation of the new states and finally, the outputs corresponding to the states. This method is demonstrated here for RLL programming using the above example of the industrial stamping process.

The ladder logic begins with a section to initialize the states and transitions to a single value, corresponding to the initial state. Some PLCs programming languages provide special instructions for such initialization. In this case, however, it is assumed that all auxiliary variables representing the states are set to zero initially. Logic is provided such that in the first scan the auxiliary state variable corresponding to the initial state would be set to 1.
The next section of the ladder logic considers the transitions and then checks for transition conditions. Each transition condition contains an auxiliary NO contact corresponding to the source state from which it is defined. For example, note that the logic for the rung corresponding to transition A contains a contact corresponding to state 1, which is the source state for transition A. Further, it contains the other logical terms corresponding to the input state variables as well as

timer outputs, if applicable. In the case of transition A, the external condition is simply the pressing of the Auto PB. Note that, in any scan cycle, at most one transition can be enabled.

The next block of rungs constitutes the state logic. If the transition logic for any transition is satisfied, the following state logic which is the destination state for the enabled transition is to be turned on and the state logic which is the source state for the enabled transition is to be turned off. Therefore each of the rungs corresponding to a state contains one auxiliary NO contact corresponding to the transition for which that state is a destination state. Similarly, each of the rungs corresponding to a state contains one auxiliary NC contact corresponding to the transition for which that state is a source state in series with the above NO contact. If there is more than one transition for which the state is a destination, all the auxiliary NO contacts corresponding to these transitions should be put in parallel. Similarly, if there is more than one transition, for which the present state is a source, all the auxiliary NC contacts corresponding to these transitions are to be put in series. This occurs in the same scan cycle in which the transition logic is turned on, since the state logic rungs follow those corresponding to transition logic. Note that at the end of every scan cycle, there is only one state logic that is enabled.

Now that the state logic has changed, in the next scan cycle the transition that was enabled, turns off and the system stays in that state, till the next transition logic gets enabled. So that the state logic remains turned on, even if a transition for which this state is a source, turns off, an NO auxiliary contact corresponding to the state that latches the state logic is to be provided in parallel with the parallel block of all the auxiliary NO contacts for each transition for which the present state is a destination.

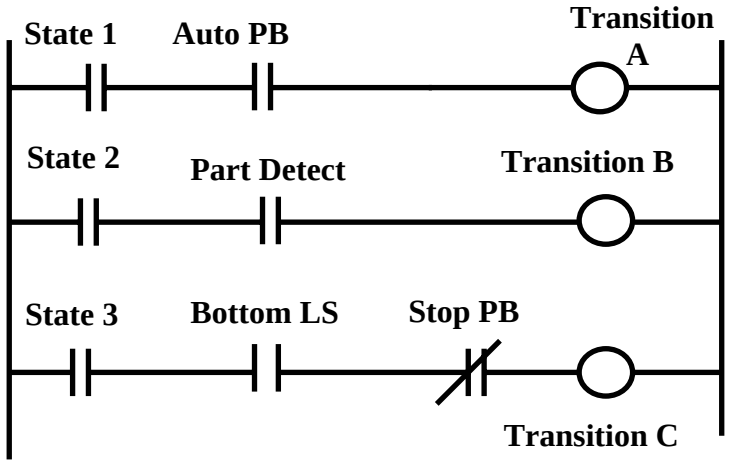

Fig. 20.5 State Transition Logic

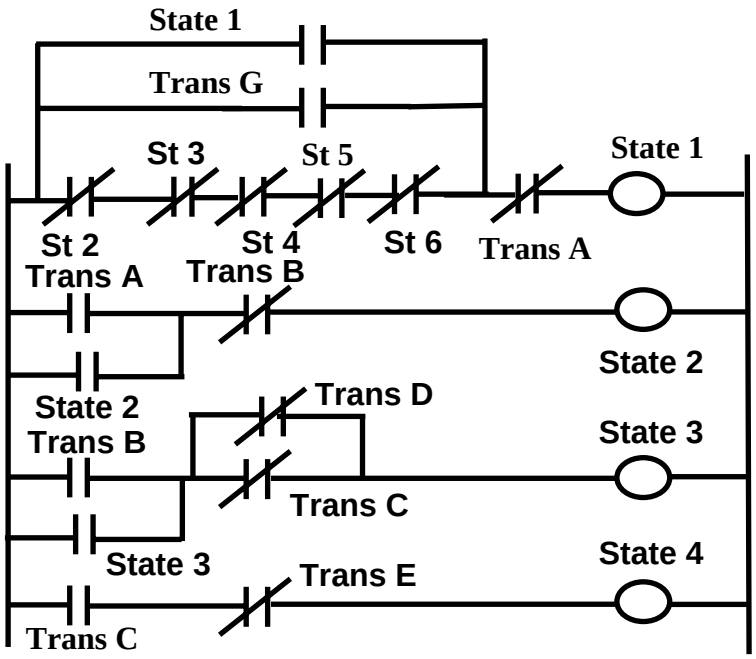

Fig. 20.6 State Logic

This is followed by ladder logic to turn on outputs as requires by the steps. This section of ladder logic corresponds to the actions for each step. The rung for each output therefore contains one NO auxiliary contact corresponding to the state in which it is enabled. If an output is enabled at more than one state, the auxiliary NO contacts corresponding to those states would be connected in parallel. Similarly, if Manual switch or PB contacts are required, they also have to be put in parallel with the contacts for the states.

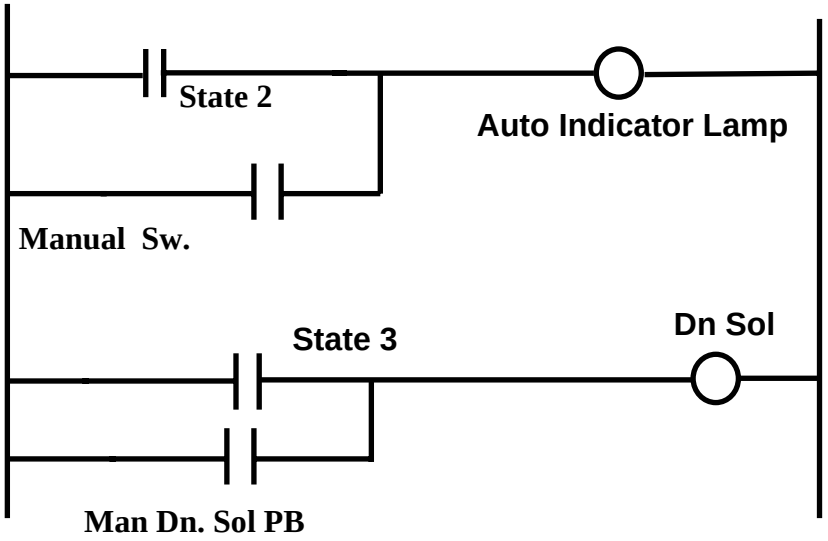

Fig. 20.7 Output Logic

Some of the rungs corresponding to state, transition and output logics of the industrial stamping process are shown in Fig. 20.5-20.7. These are mostly self explanatory. The reader is advised to check the correctness of these rungs.

Conclusion

Different methods can be used to design different controllers. The most basic controllers can be developed using simple flowcharts. More complex control problems should be solved with state diagrams. It is also possible to make a concurrent system using two or more state diagrams. However, for such concurrent processes the Sequential Function Chart formalism discussed in the next lesson is recommended.

Point to Ponder: 5

A. *What would happen if the order of the transition, state and output logic blocks is changed in an RLL program?*

B. *Mention any one disadvantage of a formal modelling approach, if you can think of it.*

C. *Mention any one advantage of a formal modelling approach, apart from the reduced risk of programming errors.*

Answers, Remarks and Hints to Points to Ponder

Point to Ponder: 1

A. Before going through the rest of the system attempt to modify the RLL program shown in Fig. 20.2 to ensure that the process always moves to the up position first, and then from there, resumes its press cycle

Ans: Without the state transition diagram this would be a considerable effort. However, with the state diagram approach of design discussed in this lesson, one only needs add a new state 7 between transition G and state 1 and then a new transition H between state 7 and state 1. This would imply that one new rung for state 7 and a new rung for transition H needs to be added. Transition G would now be a destination state for state 7 and not state 1. Similarly, Top LS would be the new enabling condition for transition H while the system would be in state 7. While in state 7, the UP solenoid would be on and therefore a new NO contact corresponding to state 7 needs to be added to the logic for UP solenoid in parallel with that for state 6.

D. Before going through the rest of the system attempt to modify the RLL program shown in Fig. 20.2 to include a part detect sensor

Ans: Already done in the lesson.

E. Attempt to merge the above two solutions so that the new program is free from both the above problems

Ans: Straightforward. Left as an exercise for the learner.

Point to Ponder: 2

A. Can we use corresponding switches, instead for the Auto, Stop and Reset PBs?

Ans: The only difference between PBs and switches is that a PB is assumed to be released automatically, while the switch is not. However, since the transition logic does not remain on for more than one scan cycle, in this case, it does not make a difference whether these are PBs or switches.

B. For the limit switches, and the part detect sensor, would you prefer mechanical switches over photo switches for this application? Justify.

Ans: Mechanical switches would be preferred here, since these are much more rugged. Turning these on and off does not cause any problem for the ram which is hydraulically powered.

C. Propose at least one more each of sensors, indicators and actuators for the above application and mention their benefits.

Ans: A hydraulic pressure sensor may be used for sensing the pump pressure before turning the machine into auto mode. This would be useful, particularly if the machine is run in auto mode for long intervals with robotic material handing equipment for part placement and removal. A fault indicator is also useful to indicate any deviation from normal cyclic operation. In this example, the electric power switch is always assumed to be on. One can add a contactor (actuator) to switch the power on, from state 1.

Point to Ponder: 3

A. *Note that in step F above it is important to detect that the part is removed. What would happen, if this is not detected?*

Ans: The same part may be stamped many times, before it is removed.

B. *What would happen if after Stop PB is pressed, Reset PB and Auto PB are pressed in that sequence, even if the piston has not been taken to the top position manually?*

Ans: If reset PB is pressed the process would move to State 1. However, pressing the auto PB would not take it to State 2, since UP Limit switch would not be made. At this point the process would deadlock unless the piston is taken up manually. After it moves up, if the Auto PB is pressed, the machine would move to State 2.

Point to Ponder: 4

A. *What would happen in the process controlled by the program shown in Fig. 20.2 if top/bottom Limit switch is stuck closed/open ?*

Ans: If the top LS is stuck closed, the Down solenoid would not be would off even if the piston comes to the bottom position. The piston would push the part to be stamped, but would not go up as desired during normal operation. Similarly, if the bottom LS is stuck closed, the piston would never come to the bottom position, since, the moment it would come to state 3, it would exit to state 4 and thus the UP solenoid would be on. One can similarly argue for the stuck open case.

B. *What would happen in the process controlled by the program shown in Fig. 20.2 if the hydraulic pump pressure becomes too low to move the RAM?*

Ans: The Up solenoid would be switched on, but the piston would not move. Thus the system would be deadlocked in state 4.

Point to Ponder: 5

A. *What would happen if the order of the transition, state and output logic blocks is changed in an RLL program?*

Ans: If the transition block is not put first, there would be an unnecessary delay of one scan cycle in turning on the state. If the output block is put before the state logic there would be a similar delay.

B. Mention any one disadvantage of a formal modelling approach, if you can think of it.

Ans: The formal modelling may lead to more number of rungs in the RLL program compared to one that does not use it. Consequently the memory and execution time requirements would be higher. However, these are not significant drawbacks for present day PLC speed and memory sizes.

C. Mention any one advantage of a formal modelling approach, apart from the reduced risk of programming errors.

Ans: It is much easier to modify an RLL developed following the formal method.

Source:http://www.nptel.ac.in/courses/Webcourse-contents/IIT%20Kharagpur/Industrial%20Automation%20control/pdf/L-20(SM)%20(IA&C)%20((EE)NPTEL).pdf