Introduction to LADSIM

This Demo Copy contains many of the features of the full program, the main exceptions being Save and Print.

LADSIM has four main features, namely:-

1. It enables the PC to simulate a PLC such that Ladder Logic PLC programs can be built up on the screen in a simple 'Drag and Drop' manner.
2. These PLC programs can then be used to control one of eight different 'on screen' simulations of control problems including a Car Park and an Elevator. Of course, if the program that the student has written is incorrect then the simulation will not work properly! (Note only four simulations are present in the Demo version).
3. After the student has become familiar with the programming techniques used for a PLC by controlling the simulations s/he can then move on to control real external items of equipment via a Bytronic Interface if required. Bytronic manufacture a range of products designed for this purpose including conveyor systems, traffic control units, rotary transfer units and many more. Please contact your supplier for more details.
4. Students can move on to programming real PLCs and connect them to LADSIM via a Bytronic Interface to control the simulations. In this way students can transfer their knowledge to manufacturers specific conventions while learning on problems they have already tackled.

In order to illustrate the potential of LADSIM the procedure for building up a Ladder Diagram to control the Car Park is given over the next few pages. It is not suggested that the solution given is the best one but it serves as a good example. If you follow the procedure you will become familiar with many of the features of the program.

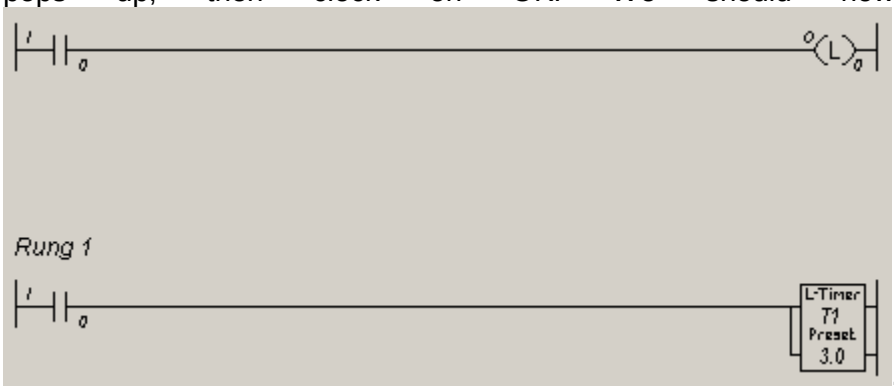
Procedure for producing a Ladder Logic Diagram for the Car Park Simulation

1. To start the program, select the LADSIM icon under Bytronic in the Start Menu.
2. Click on the Simulations menu and then on Car Park. You will be presented with the car park simulation screen for which you are about to develop some ladder code. The simulation's objectives and identification of the various input and output signals used can be found in LADSIM's help facility. You can quickly access this by pressing the F1 key.After you have finished viewing the help screen return to the simulation. You will also see the I/O browser and part of the Ladder Diagram, but we will consider these later.
3. At the top middle of the screen are controls similar to those of a Video Recorder. These control the execution of the Ladder Program. Click on the Play button. Click on the car and two pointing hands will be displayed. Click on the left hand and the car will go left and try and get into the car park. It will break the input barrier beam (IP0) but the barrier will not lift. The first part of our program must therefore make the barrier open when the beam is broken. Now click on the Stop button to stop execution of the ladder diagram.
4. The input barrier beam is a **N**ormally **O**pen **C**ontact (**NOC**) in this particular application. If you move the mouse cursor slowly over the symbols in the toolbar at the top left of the screen you will see that Normally Open Contact is displayed over the top left symbol (the one next to the arrow). Hold down the mouse button and drag the symbol to the left hand side of Rung 0 and release it. You will then be asked via the **Inputs** dialog box to select the identification of this input. We want IP0, so click OK and this will be displayed on the ladder. When IP0 is closed we want the barrier to open so drag the symbol for an output to the right hand side of the rung and identify it as OP0 in the **Outputs** dialog box and then clocking OK. We now have the first rung of our ladder as shown:

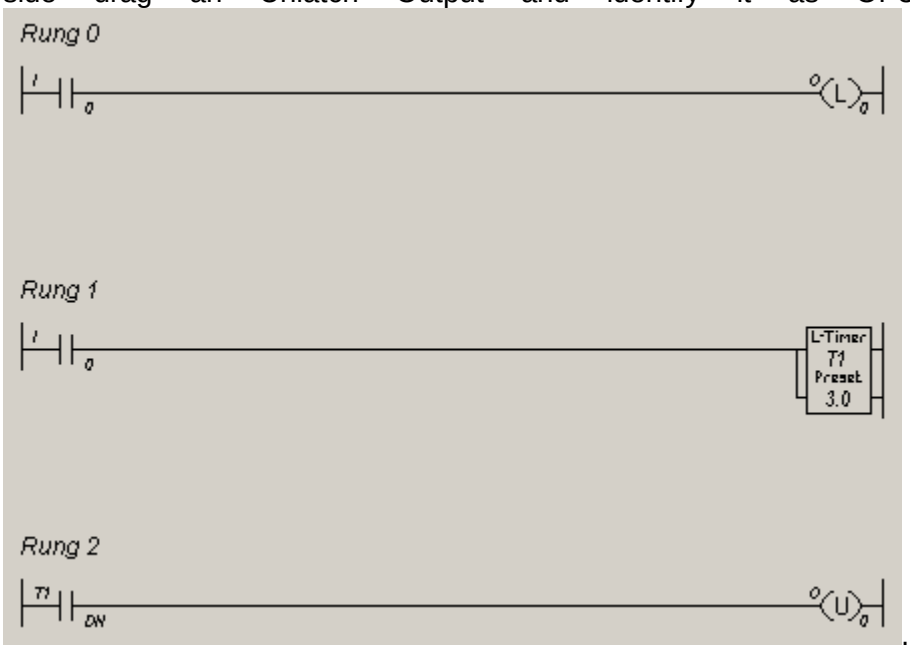
. Let's try it and see by following the simple step given in (5).
5. Click the Play button again and send the car into the car park. You should see that the car goes quite happily into the car park but unfortunately the barrier closes before the car has entered properly and the car is wrecked. Obviously we can't allow this so we need to 'latch' the barrier open. Thus, return to the ladder editor by clicking the 'stop' button as before and, from the **Edit** menu, click on Delete Control. Follow the instructions at the bottom of the screen and delete OP0. Replace it with a Latched Output. Your ladder diagram should now look like this:

.
6. Now try this version. You will see that the barrier stays open and the car is not damaged. However, following cars can now get in free of charge and cars can get out of the entrance! Click on a car that is in the car park to see what happens. This we need to add a timer to control the amount of time the barrier stays latched open. As an intermediate step we will add a timer, see how it works and then use it to control the barrier. (Stop the diagram execution). Click on the **Add Rung** button to get a second rung. We want the timer to start when the IP0 beam is broken so drag IP0 to the second rung, as with the first, and drag a Timer to the right hand side. Choose T1, Latched On

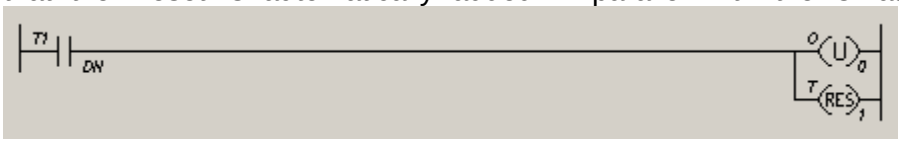
Delay Timer and Preset the time delay to 3 seconds in the dialog box that pops up, then clock on OK. We should now have:



7. Run the simulation as before but select the Timers card on the I/O Monitor. When a car breaks the beam, notice the Accumulator (Acc) display for the timer increase in 0.1 second steps and, after 3 seconds, the DN bit is set. We now have a timer which starts timing when the IP0 beam is broken and the T1/DN bit is set 3 seconds later. We can thus add a rung which will unlatch OP0 when the T1/DN bit is set. So, add a rung as before and drag a NOC to the left hand side. Identify it as T1/DN from the dialog box. On the right hand side drag an Unlatch Output and identify it as OP3 thus:



8. Try the new program and you should see that it works OK, but only for the first car. Why is this? You can get some idea by looking at the ladder diagram while it is executing. You will notice that active elements are highlighted and, in particular, you will notice that the T1/DN input is not highlighted until 3 seconds after the first car goes through. The T1/DN is then permanently set and no other car is allowed through. In order to overcome this problem it is necessary to Reset the timer at the same time that the barrier is unlatched. Thus, add a Reset to the right of rung 2 and identify it as T1. You will notice that the Reset is automatically added in parallel with the Unlatch thus:

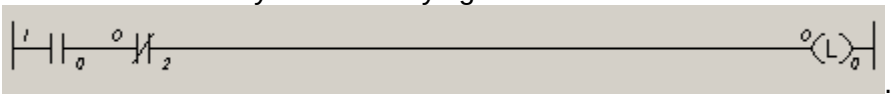


9. Try this and you should see that it works well. Congratulations, you've noew solved the problem of getting cars into the car park. The problem now, however, is that cars are trapped in the car park! Click on one and see what happens. This can be solved by adding a similar bit of program for the output barrier. If you now add this you should now have a working car park, with a bit of luck!
10. You should have noticed that the car park has a number of displays describing various aspects of the car park's functions. The next problem is to connect these Car Park displays. The Full and Spaces lights will come on when OP2 and OP3 are energised, respectively.
11. Thus, if we use Counter 1 (C1) in the 'count up' mode such that every time the entrance barrier beam is broken it triggers the counter, and we set the preset to 6 to represent the capacity of the car park, when the C1/UP bit is set, i.e. 6 is reached, we can turn on the Full light (OP2). Conversely, this same bit (or flag) can be used to control OP3 such that, when it is not set, the Spaces light is turned on.
12. To implement the above, add a rung and put IP0 on the left and a counter, identified as C1 on the right. C1 should have a preset of 6 and be an UP counter. Add another rung with a NOC on the left identified as C1/UP and OP2 on the right. Finally, add another rung with a **NCC** on the left identified as C1/UP, and OP3 (Spaces) on the right:

13. If you try this you will find that the Full and Spaces displays work but only when the cars are entering the car park. When a car exits, the count is not changed. This can be rectified by using the exit beam to trigger the same counter but this time in the 'count down' mode:

14. When you try this you will find that the program works well until we allow 7 cars into the car park. The last one cannot find a space so it exits and the counter counts down to 5. This is obviously wrong as there are still 6 cars in the car park.
15. To overcome this problem we must prevent cars entering the car park when it is full. Thus we could add a condition to the first rung that the Full display was

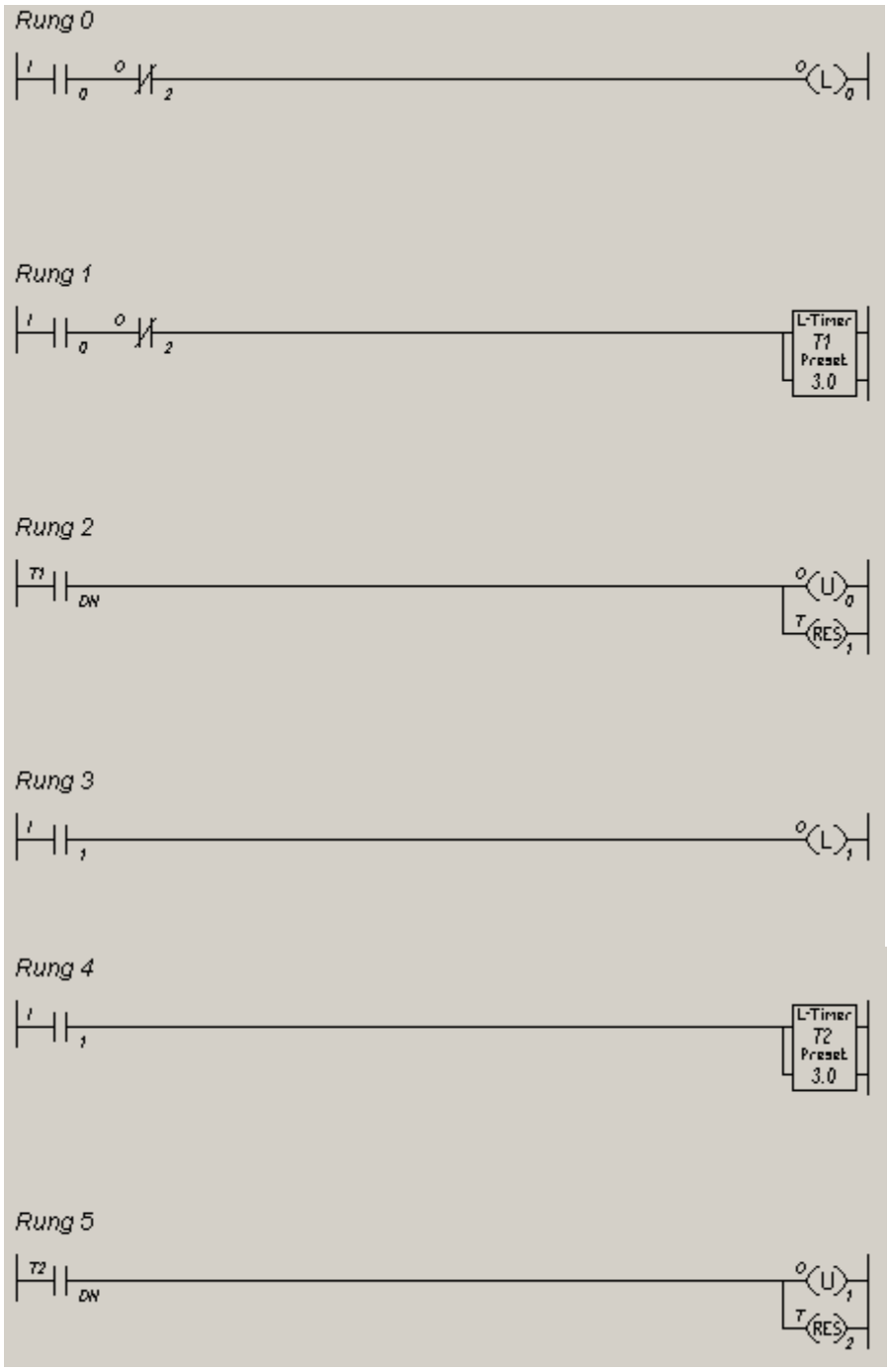
not on by modifying it as shown below:

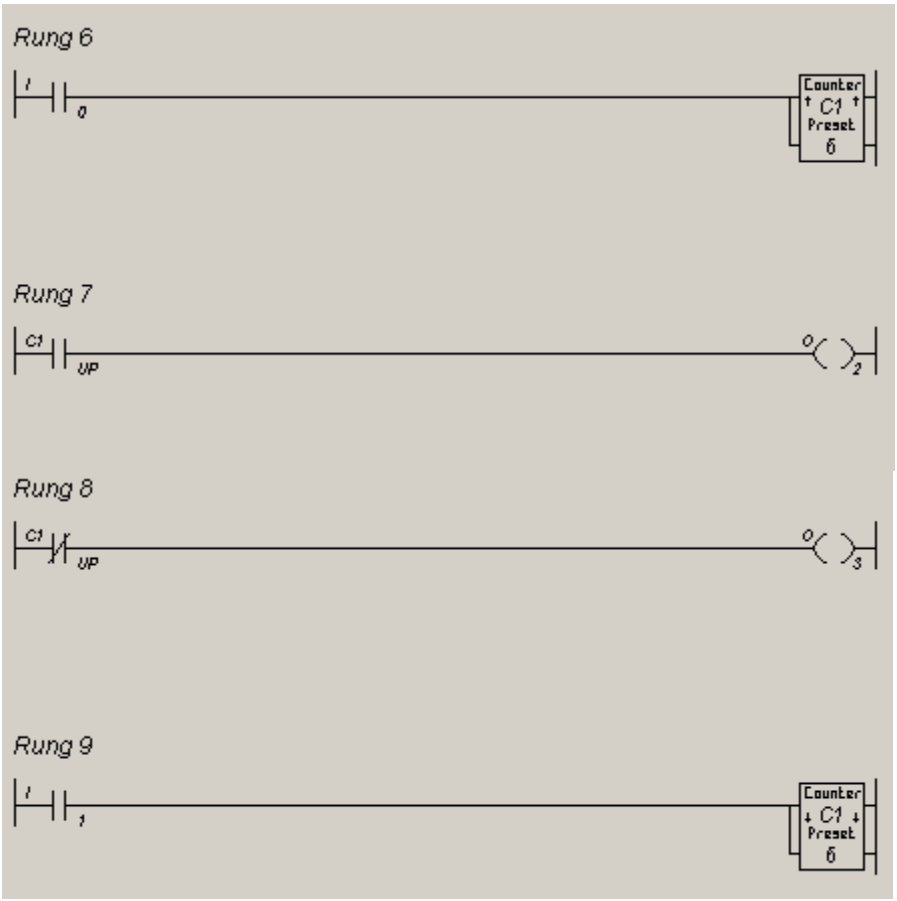


.

16. We must also modify the second rung (rung 1) in a similar way.
17. You have now completed a ladder program to control the car park simulation, The final version of the program is given at the end of this document. We hope you have found this demonstration useful. We think it demonstrates many of LADSIM's features. There are, however, many other features of LADSIM that we have not covered. For example:
    a. Comments can be added to every rung allowing you to clearly document your program – click Show Comments in the Edit menu.
    b. Programs can be debugged using the **Debugger** before attempting control of simulations or external devices – the debugger is found on the **Control** menu.
    c. LADSIM also includes the use of **Flags** and **Shift Registers**.
    d. Your ladder diagrams can be printed with and without comments.
    e. LADSIM includes another 7 simulations that will test your students' ability to write control programs.
    f. LADSIM includes a direct link to the Bytronic range of interface cards that will allow you to control real external plant via the interface card.
    g. PLCs can be connected to LADSIM via the Bytronic range of interface cards to control the simulations.

LADSIM – Suggested Solution for Car Park Simulation




Rung 6
I
0
Counter
↑ C1 ↑
Preset
6
Rung 7
C1
UP
O
2
Rung 8
C1
UP
O
3
Rung 9
I
1
Counter
↓ C1 ↓
Preset
6