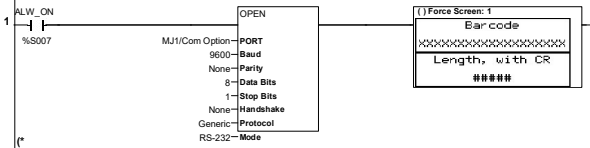
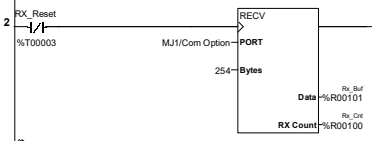
(* In this example, the bar code data will vary in length. Therefore, the method for receving characters is as follows:
1. Watch the RX Count Register (%R100).
2. When %R100 is nonzero but unchanged for 250mS, assume the transmission from the bar code reader is complete.
3. When the receive is complete, copy the data into another bank of registers (%R201-%R210), clear the receive buffer (%R101-110) and reset the Receive block.
4. The received characters stored in %R201-210 are displayed on Screen #1 on the OCS. *)

1



(*
Characters are received from a bar code reader. When not reset, the Receive Block is enabled, allowing characters to be placed in the Receive buffer starting at register %R101 [Rx_Buf]. The buffer length is 254 characters (one less than the maximum allowed), or 127 words. Note that the [RX Count] parameter is %R100, located one word register before the buffer. So the length of the characters received is contiguous with the characters received. *)

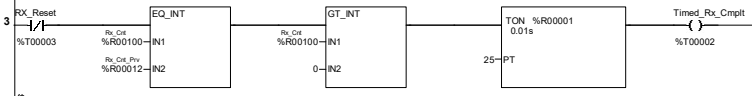
2



(*
When not reset, check to see that characters have been received, but RX_Cnt is not changing; this is assumed to be an indication that transmission by the bar code reader is complete. To do this, compare RX_Cnt with RX_Cnt_Prv, which is the value of RX_Cnt from the previous scan. Also check RX_Cnt to see that it is greater than zero, indicating characters have been received. Also, since Rx_Cnt is greater than zero it cannot be -1, which would indicate that the port is not open.

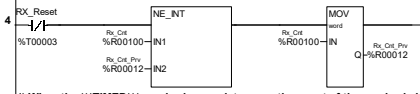
If those conditions are all true, trigger the timer; if the timer completes, assign a 1 to the ***TIMED*** receive complete bit, indicating the bar code read into the buffer so far is a complete barcode. *)

3



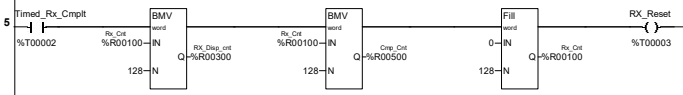
(*
When not reset, if the current received count of characters is different than the value from the previous scan,, then copy the former to the the latter, for comparison in the previous rung on the next scan. *)

4



(* When the ***TIMED*** receive is complete, copy the count of the received characters and the receive buffer characters to Display and Compare register buffers, then clear the received buffer count to 0 along with the receive buffer (128 words = 1 word + 254 characters). Then assign a 1 to RX_Reset bit to make the next ***ONE*** scan a reset scan, which removes power from the Receive Block; the scan after that, the RECV instruction will start a new read. *)

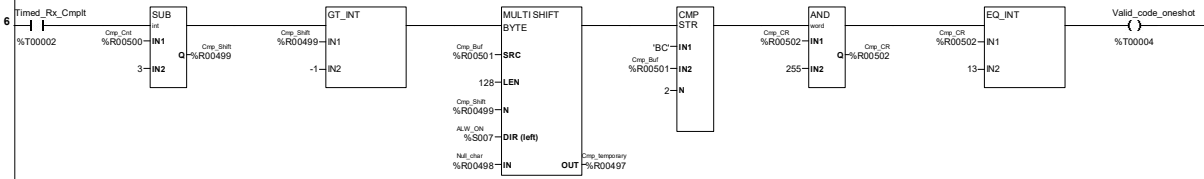
5



(*

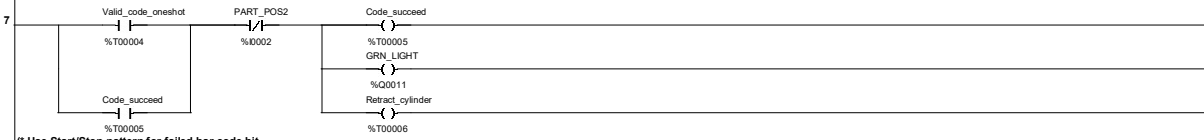
Calculate the left shift necessary to move the last three characters in the Compare buffer to the start of the Compare buffer, so the first two of those last three will be in the low- and high-byte of %R501, and the last byte will be in the low-byte of %R502 [Cmp_CR]. Then do the shift, compare 'BC' to those first two bytes, mask out the high-byte of Cmp_CR and compare Cmp_CR to ASCII code 13 (carriage return), and finally assign 1 to Valid_code if all compares were true *)

6



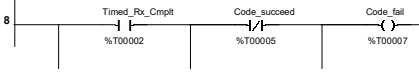
(* Use Start/Stop pattern for successful bar code bit
- Start: valid code oneshot bit from previous rung
- Stop: part reaches position 2 *)

7



(* Use Start/Stop pattern for failed bar code bit
- Start: oneshot when ***TIMED*** receive completes
- Stop: code succeeded from previous rung *)

8




RED_LIGHT
%Q0012
Code_fail
%T00007
(*
Unless the cylinder should be retracted, actiivate the cylinder-exend soloenoid until the extended cylinder prox is reached *)
9
Retract_cylinder
%T00006
Cylinder_fully_extend
%I0005
SOL_EXT
%Q0002
(*
When the cylinder should be retracted, activate the cylinder-retract solenoid until the retracted cylinder prox is reached *)
10
Retract_cylinder
%T00006
Cylinder_fully_retract
%I0006
SOL_RET
%Q0001