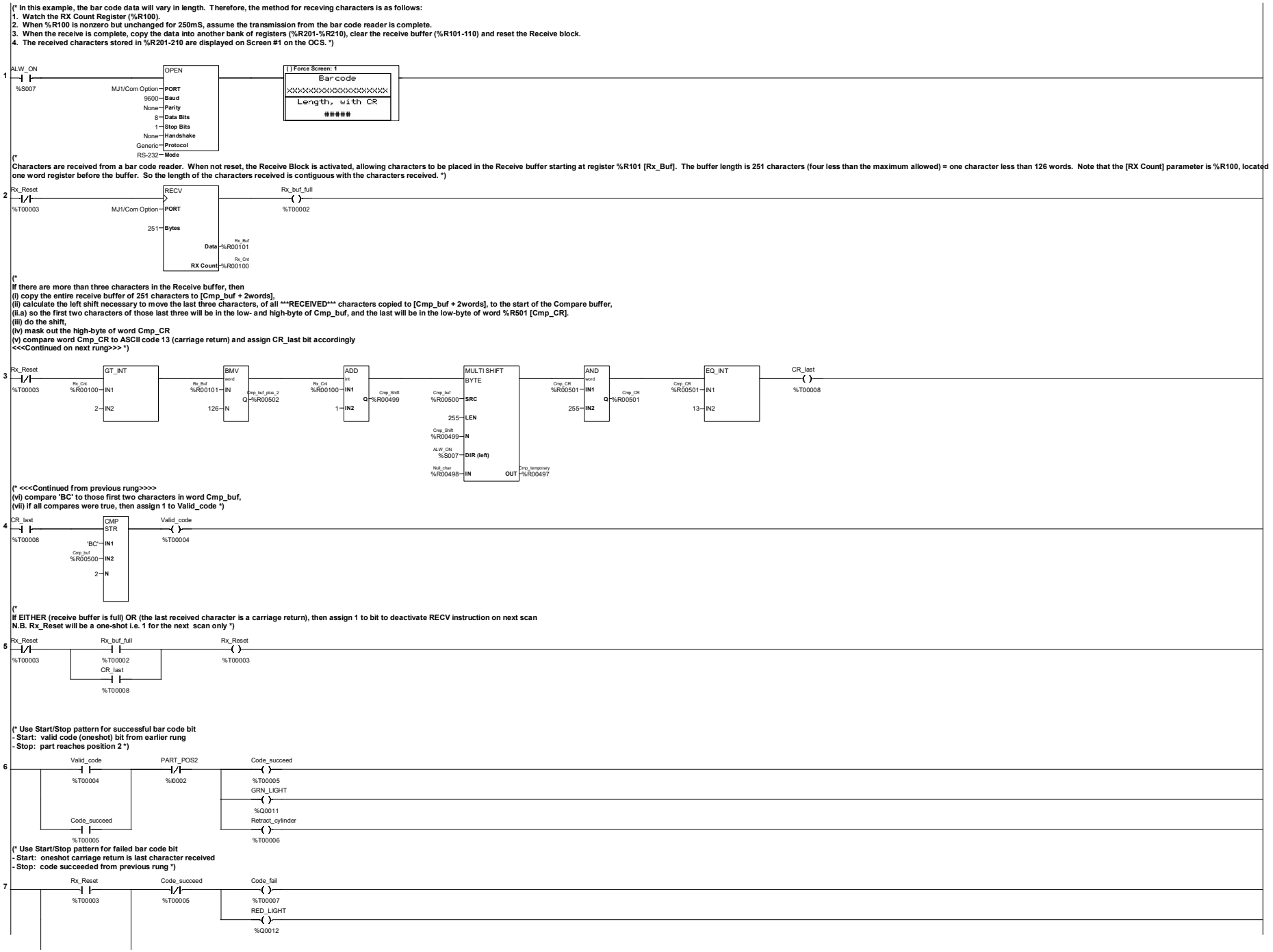
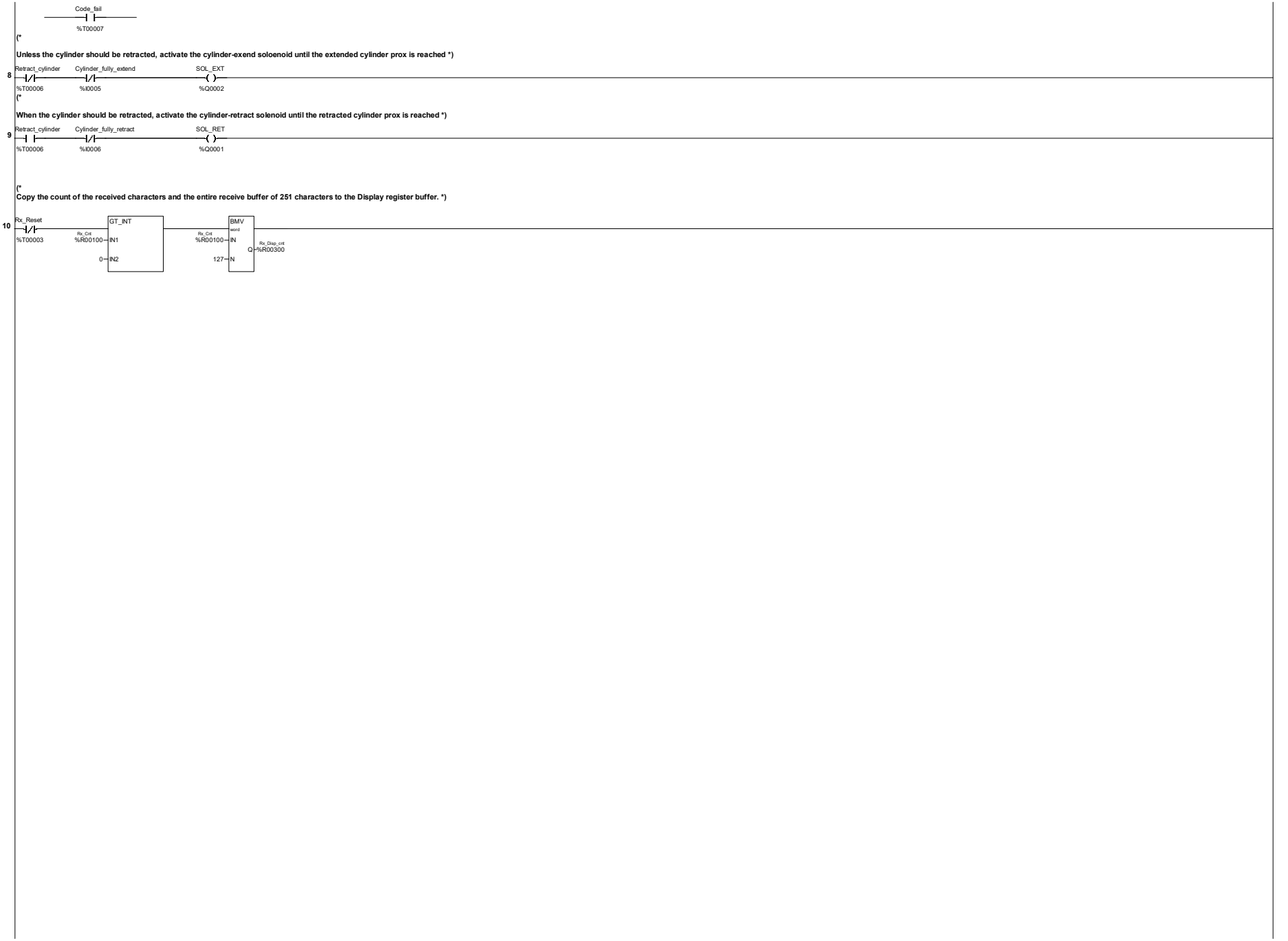

(* In this example, the bar code data will vary in length. Therefore, the method for receving characters is as follows:
1. Watch the RX Count Register (%R100).
2. When %R100 is nonzero but unchanged for 250mS, assume the transmission from the bar code reader is complete.
3. When the receive is complete, 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
ALW_ON
%S007
OPEN
MJ1/Com Option
PORT
9600
Baud
None
Parity
8
Data Bits
1
Stop Bits
None
Handshake
Generic
Protocol
RS-232
Mode
( ) Force Screen: 1
Barcode
Length, with CR
#####
(*
Characters are received from a bar code reader. When not reset, the Receive Block is activated, allowing characters to be placed in the Receive buffer starting at register %R101 [Rx_Buf]. The buffer length is 251 characters (four less than the maximum allowed) = one character less than 126 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
Rx_Reset
%T00003
RECV
MJ1/Com Option
PORT
251
Bytes
Data
Rx_Buf
%R00101
RX Count
Rx_Cnt
%R00100
Rx_buf_full
%T00002
(*
If there are more than three characters in the Receive buffer, then
(i) copy the entire receive buffer of 251 characters to [Cmp_buf + 2words],
(ii) calculate the left shift necessary to move the last three characters, of all ***RECEIVED*** characters copied to [Cmp_buf + 2words], to the start of the Compare buffer,
(ii.a) so the first two characters of those last three will be in the low- and high-byte of Cmp_buf, and the last will be in the low-byte of word %R501 [Cmp_CR].
(iii) do the shift,
(iv) mask out the high-byte of word Cmp_CR
(v) compare word Cmp_CR to ASCII code 13 (carriage return) and assign CR_last bit accordingly
<<<Continued on next rung>>> *)
3
Rx_Reset
%T00003
GT_INT
Rx_Cnt
%R00100
IN1
2
IN2
BMV
word
Rx_Buf
%R00101
IN
126
N
Cmp_buf_plus_2
%R00502
ADD
int
Rx_Cnt
%R00100
IN1
1
IN2
Cmp_Shift
%R00499
MULTI SHIFT
BYTE
Cmp_buf
%R00500
SRC
255
LEN
Cmp_Shift
%R00499
N
ALW_ON
%S007
DIR (left)
Null_char
%R00498
IN
OUT
Cmp_temporary
%R00497
AND
word
Cmp_CR
%R00501
IN1
255
IN2
Cmp_CR
%R00501
EQ_INT
Cmp_CR
%R00501
IN1
13
IN2
CR_last
%T00008
(* <<<Continued from previous rung>>>>
(vi) compare 'BC' to those first two characters in word Cmp_buf,
(vii) if all compares were true, then assign 1 to Valid_code *)
4
CR_last
%T00008
CMP
STR
'BC'
IN1
Cmp_buf
%R00500
IN2
2
N
Valid_code
%T00004
(*
If EITHER (receive buffer is full) OR (the last received character is a carriage return), then assign 1 to bit to deactivate RECV instruction on next scan
N.B. Rx_Reset will be a one-shot i.e. 1 for the next scan only *)
5
Rx_Reset
%T00003
Rx_buf_full
%T00002
CR_last
%T00008
Rx_Reset
%T00003
(* Use Start/Stop pattern for successful bar code bit
- Start: valid code (oneshot) bit from earlier rung
- Stop: part reaches position 2 *)
6
Valid_code
%T00004
PART_POS2
%I0002
Code_succeed
%T00005
GRN_LIGHT
%Q0011
Code_succeed
%T00005
Retract_cylinder
%T00006
(* Use Start/Stop pattern for failed bar code bit
- Start: oneshot carriage return is last character received
- Stop: code succeeded from previous rung *)
7
Rx_Reset
%T00003
Code_succeed
%T00005
Code_fail
%T00007
RED_LIGHT
%Q0012


Code_fail
%T00007
(*
Unless the cylinder should be retracted, activate the cylinder-exend soloenoid until the extended cylinder prox is reached *)
8
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 *)
9
Retract_cylinder
%T00006
Cylinder_fully_retract
%I0006
SOL_RET
%Q0001
(*
Copy the count of the received characters and the entire receive buffer of 251 characters to the Display register buffer. *)
10
Rx_Reset
%T00003
GT_INT
Rx_Cnt
%R00100
IN1
0
IN2
BMV
word
Rx_Cnt
%R00100
IN
Rx_Disp_cnt
%R00300
Q
127
N