```
──────────────── MODULE ReceiveModbus ────────────────
EXTENDS Sequences,
        Naturals,
        Modbus,
        TLC

LOCAL INSTANCE Hex
    WITH natValue ← 0, hexValue ← ⟨0⟩

MessagesToSerialPort ≜
    {⟨“:”, “J”, “G”, “P”, “9”, “4”, “3”, “2”, “J”, “3”, “9”, “J”, “G”, “W”, “I”, “R”, “W”⟩,
     ⟨“:”, “1”, “1”, “0”, “3”, “0”, “0”, “6”, “B”, “0”, “0”, “0”, “3”, “7”, “E”, “C”, “R”, “L”, “F”⟩,
     ⟨“:”, “1”, “1”, “0”, “3”, “0”, “0”, “6”, “B”, “0”, “0”, “0”, “3”, “7”, “E”, “C”, “R”, “L”, “1”, “0”, “3”, “
     ⟨“:”, “1”, “1”, “0”, “3”, “0”, “0”, “6”, “B”, “0”, “0”, “0”, “:”, “1”, “1”, “0”, “3”, “0”, “0”, “6”, “B”, “0”
     ⟨⟩,
     ⟨“ ”⟩,
⟨“!”⟩,
⟨“\””⟩,
⟨“#”⟩,
⟨“$”⟩,
⟨“%”⟩,
⟨“&”⟩,
⟨“’”⟩,
⟨“(”⟩,
⟨“)”⟩,
⟨“*”⟩,
⟨“+”⟩,
⟨“,”⟩,
⟨“-”⟩,
⟨“.”⟩,
⟨“/”⟩,
⟨“0”⟩,
⟨“1”⟩,
⟨“2”⟩,
⟨“3”⟩,
⟨“4”⟩,
⟨“5”⟩,
⟨“6”⟩,
⟨“7”⟩,
⟨“8”⟩,
⟨“9”⟩,
⟨“:”⟩,
⟨“;”⟩,
⟨“<”⟩,
⟨“=”⟩,
⟨“>”⟩,
⟨“?”⟩,
⟨“@”⟩,
⟨“A”⟩,
⟨“B”⟩,
⟨“C”⟩,
⟨“D”⟩,
⟨“E”⟩,
⟨“F”⟩,
⟨“G”⟩,
⟨“H”⟩,
⟨“I”⟩,
⟨“J”⟩,
⟨“K”⟩,
⟨“L”⟩,
⟨“M”⟩,
⟨“N”⟩,
⟨“O”⟩,
⟨“P”⟩,
⟨“Q”⟩,
⟨“R”⟩,
⟨“S”⟩,
⟨“T”⟩,
⟨“U”⟩,
⟨“V”⟩,
⟨“W”⟩,
⟨“X”⟩,
⟨“Y”⟩,
⟨“Z”⟩,
⟨“[”⟩,
⟨“\\”⟩,
⟨“]”⟩,
⟨“^”⟩,
⟨“_”⟩,
⟨“a”⟩,
⟨“b”⟩,
⟨“c”⟩,
⟨“d”⟩,
⟨“e”⟩,
⟨“f”⟩,
⟨“g”⟩,
⟨“h”⟩,
⟨“i”⟩,
⟨“j”⟩,
⟨“k”⟩,
```

$\langle$ "l" $\rangle$,
$\langle$ "m" $\rangle$,
$\langle$ "n" $\rangle$,
$\langle$ "o" $\rangle$,
$\langle$ "p" $\rangle$,
$\langle$ "q" $\rangle$,
$\langle$ "r" $\rangle$,
$\langle$ "s" $\rangle$,
$\langle$ "t" $\rangle$,
$\langle$ "u" $\rangle$,
$\langle$ "v" $\rangle$,
$\langle$ "w" $\rangle$,
$\langle$ "x" $\rangle$,
$\langle$ "y" $\rangle$,
$\langle$ "z" $\rangle$,
$\langle$ "{" $\rangle$,
$\langle$ "|" $\rangle$,
$\langle$ "}" $\rangle$,
$\langle$ "$\sim$" $\rangle$

}

}

BEGIN TRANSLATION

```
VARIABLES $rx$, $rxBuf$, $rxReg$, $incMessage$, $applicationBuffer$, $pc$

$vars \triangleq \langle rx, rxBuf, rxReg, incMessage, applicationBuffer, pc\rangle$

$Init \triangleq$ Global variables
          $\land rx = \text{FALSE}$
          $\land rxBuf = \langle\rangle$
          $\land rxReg = \langle\rangle$
          $\land incMessage \in MessagesToSerialPort$
          $\land applicationBuffer = \langle\rangle$
          $\land pc = $ "idle"

$idle \triangleq$ $\land pc = $ "idle"
          $\land$ IF $Len(incMessage) = 0$
                THEN $\land pc' = $ "start"
                     $\land rxReg' = \langle\rangle$
                     $\land$ UNCHANGED $\langle incMessage\rangle$
                ELSE $\land rxReg' = \langle Head(incMessage)\rangle$
                     $\land incMessage' = Tail(incMessage)$
                     $\land pc' = $ "start"
          $\land$ UNCHANGED $\langle rx, rxBuf, applicationBuffer\rangle$
```

```
$start \triangleq$  $\land pc = \text{"start"}$
                    $\land$ IF $rxReg \neq \langle\rangle \land Len(rxBuf) < MAXMODBUSSIZE$
                          THEN $\land pc' = \text{"receive"}$
                          ELSE $\land pc' = \text{"alldone"}$
                    $\land$ UNCHANGED $\langle rx, rxBuf, rxReg, incMessage, applicationBuffer\rangle$

$receive \triangleq$  $\land pc = \text{"receive"}$
                      $\land$ IF $Head(rxReg) = \text{":"}$
                            THEN $\land rxBuf' = \langle\rangle$
                            ELSE $\land$ TRUE
                                 $\land rxBuf' = rxBuf$
                      $\land pc' = \text{"r1"}$
                      $\land$ UNCHANGED $\langle rx, rxReg, incMessage, applicationBuffer\rangle$

$r1 \triangleq$  $\land pc = \text{"r1"}$
                 $\land rxBuf' = rxBuf \circ rxReg$
                 $\land pc' = \text{"r2"}$
                 $\land$ UNCHANGED $\langle rx, rxReg, incMessage, applicationBuffer\rangle$

$r2 \triangleq$  $\land pc = \text{"r2"}$
                 $\land rxReg' = \langle\rangle$
                 $\land pc' = \text{"check"}$
                 $\land$ UNCHANGED $\langle rx, rxBuf, incMessage, applicationBuffer\rangle$

$check \triangleq$  $\land pc = \text{"check"}$
                    $\land$ IF $IsModbus(rxBuf)$
                          THEN $\land rx' = \text{TRUE}$
                               $\land applicationBuffer' = rxBuf$
                               $\land pc' = \text{"alldone"}$
                          ELSE $\land pc' = \text{"idle"}$
                               $\land$ UNCHANGED $\langle rx, applicationBuffer\rangle$
                    $\land$ UNCHANGED $\langle rxBuf, rxReg, incMessage\rangle$

$alldone \triangleq$  $\land pc = \text{"alldone"}$
                      $\land rxBuf' = \langle\rangle$
                      $\land rxReg' = \langle\rangle$
                      $\land pc' = \text{"Done"}$
                      $\land$ UNCHANGED $\langle rx, incMessage, applicationBuffer\rangle$

$Next \triangleq$  $idle \lor start \lor receive \lor r1 \lor r2 \lor check \lor alldone$
                       $\lor$  Disjunct to prevent deadlock on termination
                         $(pc = \text{"Done"} \land \text{UNCHANGED } vars)$

$Spec \triangleq$  $\land Init \land \Box[Next]_{vars}$
                   $\land \text{WF}_{vars}(Next)$

$Termination \triangleq \Diamond(pc = \text{"Done"})$
```

END TRANSLATION

$SAFETYCHECK \triangleq$

receive buffer never overflows

$\land\ Len(rxBuf) \leq MAXMODBUSSIZE$

application buffer never overflows

$\land\ Len(applicationBuffer) \leq MAXMODBUSSIZE$

only valid modbus makes it to the app buffer

$\land\ IsModbus(applicationBuffer) \lor applicationBuffer = \langle\rangle$

flag is raised if and only if there is valid modbus in app buffer

$\land\ rx = \text{TRUE} \equiv IsModbus(applicationBuffer)$

$LIVELINESS \triangleq$

if the message is modbus then it gets to the app buffer

$\land\ IsModbus(incMessage) \leadsto IsModbus(applicationBuffer)$

if valid modbus comes through then it gets flagged for the application to consume

$\land\ IsModbus(incMessage) \leadsto rx = \text{TRUE}$

\* Modification History
\* Last modified *Mon* May 07 18:59:34 *EDT* 2018 by *SabraouM*
\* Created Sat May 05 11:36:54 *EDT* 2018 by *SabraouM*