MODULE *LRC*

LOCAL INSTANCE *Naturals*
LOCAL INSTANCE *Sequences*
LOCAL INSTANCE *TLC*

RECURSIVE $addSeq(\_)$
$addSeq(seq) \triangleq$
  IF $seq = \langle\rangle$
   THEN 0
   ELSE $Head(seq) + addSeq(Tail(seq))$

Takes a decimal number and creates an 8-bit binary sequence. 8-bit is all that is necessary for *LRC* so that is all we do here

$$\begin{aligned} DecimalToBinarySeq(num) \triangleq\ & \langle num \div 128\%2\rangle \\ \circ\ & \langle num \div 64\%2\rangle \\ \circ\ & \langle num \div 32\%2\rangle \\ \circ\ & \langle num \div 16\%2\rangle \\ \circ\ & \langle num \div 8\%2\rangle \\ \circ\ & \langle num \div 4\%2\rangle \\ \circ\ & \langle num \div 2\%2\rangle \\ \circ\ & \langle num\%2\rangle \end{aligned}$$

B2D and *BinarySeqToDecimal* operate in tandem to transform a sequence of 1's and 0's of arbitrary length to a decimal numb

RECURSIVE $B2D(\_, \_)$
LOCAL $B2D(num, seq) \triangleq$
  IF $seq = \langle\rangle$
   THEN 0
   ELSE $(seq[Len(seq)] * num) + B2D(2 * num, SubSeq(seq, 1, Len(seq) - 1))$

$BinarySeqToDecimal(seq) \triangleq B2D(1, seq)$

adds 1 to a binary sequence the "I don't feel like doing this in binary" way

$BinaryAdd1(seq) \triangleq DecimalToBinarySeq(BinarySeqToDecimal(seq) + 1)$

XOR for 1's and 0's because TLA+ *XOR* only operates on TRUE/FALSE

$$\begin{aligned} XOR(a, b) \triangleq \text{CASE}\ & a = 1 \land b = 1 \rightarrow 0 \\ \Box\ & a = 1 \land b = 0 \rightarrow 1 \\ \Box\ & a = 0 \land b = 1 \rightarrow 1 \\ \Box\ & \text{OTHER} \rightarrow 0 \end{aligned}$$

takes a binary sequence and produces the $2s$ compliment

$TwosComp(seq) \triangleq BinaryAdd1([x \in \text{DOMAIN } seq \mapsto XOR(seq[x], 1)])$

the reason this module exists

$CalculateLRC(seq) \triangleq BinarySeqToDecimal(TwosComp(DecimalToBinarySeq((addSeq(seq)\%256))))$ negative

\ * Modification History
\ * Last modified *Thu Jun* 13 12:27:36 *EDT* 2019 by *mehdi*
\ * Last modified *Wed* May 02 18:01:26 *EDT* 2018 by *SabraouM*
\ * Created *Wed* May 02 17:41:39 *EDT* 2018 by *SabraouM*