

---

MODULE *AddSeq*

---

EXTENDS *Integers, Sequences, TLC*

CONSTANT *DigitSeq*

RECURSIVE *Val*(-)  
 $Val(seq) \triangleq \text{IF } seq = \langle \rangle \text{ THEN } 0 \text{ ELSE } seq[1] + 10 * Val(Tail(seq))$

$Digits \triangleq 0 \dots 9$

$Fix(seq) \triangleq \text{IF } seq = \langle \rangle \text{ THEN } \langle 0 \rangle \text{ ELSE } seq$

RECURSIVE *SAdd*(-, -, -)  
 $SAdd(s1, s2, c) \triangleq$   
     IF  $s1 \circ s2 = \langle \rangle$   
         THEN IF  $c = 1$  THEN  $\langle 1 \rangle$  ELSE  $\langle \rangle$   
         ELSE LET  $dig \triangleq Fix(s1)[1] + Fix(s2)[1] + c$   
             IN  $\langle dig \% 10 \rangle \circ SAdd(Tail(Fix(s1)), Tail(Fix(s2)), dig \div 10)$

$s1 \oplus s2 \triangleq SAdd(s1, s2, 0)$

VARIABLES *pc, u, v, sum, carry*

$vars \triangleq \langle u, v, sum, carry, pc \rangle$

*Init*  $\triangleq$  Global variables  
      $\wedge u \in DigitSeq$   
      $\wedge v \in DigitSeq$   
      $\wedge sum = \langle \rangle$   
      $\wedge carry = 0$   
      $\wedge pc = \text{"a"}$

The  $\div$  operator in this definition is defined so that  $m \div n$  is the smallest integer  $d$  such that  $d * n$  is less than or equal to  $m$ , for any integers  $m$  and  $n$  with  $n > 0$ .

*Next*  $\triangleq$   $\wedge pc = \text{"a"}$   
      $\wedge \text{IF } (u \neq \langle \rangle) \vee (v \neq \langle \rangle) \vee carry \neq 0$   
         THEN  $\wedge \text{LET } digit \triangleq Fix(u)[1] + Fix(v)[1] + carry$  IN  
              $\wedge sum' = Append(sum, digit \% 10)$   
              $\wedge carry' = (digit \div 10)$   
              $\wedge u' = Tail(Fix(u))$   
              $\wedge v' = Tail(Fix(v))$   
              $\wedge pc' = \text{"a"}$   
         ELSE  $\wedge carry' = 0$   
              $\wedge pc' = \text{"done"}$   
              $\wedge \text{UNCHANGED } \langle u, v, sum \rangle$

$AddSeq \triangleq Init \wedge \Box [Next]_{vars}$

This is the correct version of the inductive invariant that checks that the final value of sum is  $u \oplus v$  for the initial values of  $u$  and  $v$ .

$$\begin{aligned}
Inv \triangleq & \wedge u \in Seq(Digits) \\
& \wedge v \in Seq(Digits) \\
& \wedge sum \in Seq(Digits) \\
& \wedge carry \in \{0, 1\} \\
& \wedge pc \in \{\text{"a"}, \text{"done"}\} \\
& \wedge (pc = \text{"Done"}) \Rightarrow (u = \langle \rangle) \wedge (v = \langle \rangle) \wedge (carry = 0) \\
& \wedge (pc = \text{"Done"}) \Rightarrow (u = \langle \rangle) \wedge (v = \langle \rangle) \wedge (carry = 0)
\end{aligned}$$


---

The following asserts that *AddSeq* implements *AddSpec* under a suitable refinement mapping.

$$\begin{aligned}
endBar & \triangleq sum \neq \langle \rangle \\
xBar & \triangleq \text{IF } endBar \text{ THEN } 0 \text{ ELSE } Val(u) \\
yBar & \triangleq \text{IF } endBar \text{ THEN } 0 \text{ ELSE } Val(v) \\
zBar & \triangleq Val(sum) + (10^{Len(sum)}) * (carry + Val(u) + Val(v)) \\
A & \triangleq \text{INSTANCE } Add \text{ WITH } x \leftarrow xBar, y \leftarrow yBar, end \leftarrow endBar, z \leftarrow zBar
\end{aligned}$$


---



---

\ \* Modification History  
\ \* Last modified *Wed Oct 23 16:25:52 CEST 2024* by *lamport*  
\ \* Created *Mon Apr 10 10:43:22 PDT 2023* by *lamport*