NAME

ADDACC − add a REAL*8 scalar to a REAL*8 two-part accumulator

SYNOPSIS

CALL ADDACC(S, ACC)

S	is the REAL*8 scalar to be added to the accumulator
ACC(2)	is the REAL*8 two-part accumulator

DESCRIPTION

On input ACC contains zero or some quantity that has been accumulated previously, with the most significant part in ACC(1) and the least in ACC(2).

The routine begins by setting U=ACC(1) and V=S (if |ACC(1)|>|S|) or U=S and V=ACC(1) (otherwise). Next it finds the sum Z=U+V, which will usually differ from the true sum due to cancellation error. The error that was made is approximately ZZ=(U−Z)+V. To this the routine adds the least significant part of the accumulator, to produce the total correction ZZ=ZZ+ACC(2). Then the true sum is ACC(1)=Z+ZZ. The error in this calculation is approximately (Z−ACC(1))+ZZ, and this correction is stored in ACC(2) to be accumulated on the next invocation of the routine.

WARNING

The arithmetic operations described above must be performed in that order so that the errors in them are correctly estimated and corrected for. It is therefore essential that this routine be compiled with optimization turned off, to prevent the compiler from "simplifying" the calculations in a way that defeats their purpose.

SEE ALSO

MPYACC, which adds the product of two scalars to an accumulator

LINKAGE

gfortran source.f −L${HOME}/lib −lmisc

AUTHOR

Michael Kupferschmid

REFERENCES

Kupferschmid, Michael, Classical Fortran: Programming for Engineering and Scientific Applications, CRC Press (2009), additional section 18.4+.

EXAMPLE

```
      REAL*8 S,ACC(2)
      ACC(1)=1.D+16
      ACC(2)=0.D0
      S=0.01D0
      CALL ADDACC(S, ACC )
      WRITE(6,901) 1.D16+0.01D0,ACC(1),ACC(2)
  901 FORMAT(1PD22.16/'or more precisely ',1PD22.16,' + '1PD22.16)
      STOP
      END
```

This example produced the following output:

```
unix[1] a.out
1.0000000000000000D+16
or more precisely 1.0000000000000000D+16 + 1.0000000000000000D-02
unix[2]
```