about 30%). Presumably, the latter tilt towards integer operands for "/" is the natural consequence of generalizing compression to replication in APL2, a generalization which allows an integer operand having no Boolean counterpart.

	A1000	A3000
INT ADD	381.0	67.5
FP ADD	8717.0	187.3
INT MULT	7134.3	238.0
FP MULT	13829.1	199.4
INDEX	126.4, 180.8	28.4, 42.4
CHAR COMPR	125.1, 178.0	29.5, 43.3
INT COMPR	119.0, 173.2	42.0, 27.9
INT +RED	25.0	4.0
INT ⌈RED	24.1	4.0
BOOL SCAN	67794.3, 100.0	17358.4, 18.0
MAT ROTAT	576.2	144.1
CHAR TRANS	753.8	171.5
INT TRANS	804.1	178.0
VEC OF VECS	1019.3	224.2
PARTITION	456.9	105.6
RHO EACH	630.1	157.8
VEC COMPAR	21.1	3.0
INT SRT	710.2	151.4
BOOL COMPAR	437.0, 185.1	108.6, 38.0
IOTA	1552.3	76.2

Although the numbers for A3000 are generally in the same ball park as those in the tabulation found in the Dyalog APL review, substantially better than the PS/2 Mod 70 in a few instances (the *BOOL SCAN* test with Boolean *BM*, and the dyadic *IOTA* test), but substantially poorer in many instances (e.g., *MAT ROTAT*, *CHAR* and *INT TRANS*, *INT SRT*, and *BOOL COMPAR*), its relative performance seems a bit weak to me, considering the potential strength of a 25 Mhz 68030/68882 combination. I can only speculate about the reasons —one contributing source possibly being the multitasking operating system of the Amiga, which makes it impossible for the system to devote its exclusive attention to an APL computing task.

Overall, I'm not really unhappy with these numbers. In a practical sense more power would simply be a luxury. I did find some evidence that the algorithm employed by APL.68000 for the interpretation of ⌹ may need some fine-tuning. The following comparison between A3000 with APL.68000 Level II and a PS/2 (20 Mhz 80386/80387) with APL2/PC uses two test expressions:

```
Q*0.5   ←→where Q←?40000⍴999999
⌹A      ←→where A←(¯50000+?40 40⍴99999)÷25000
```

the first to test the performance of the 25 Mhz 68882 relative to the 20 MHz 80387 (since both have a single instruction for square root), the second to compare execution times in inverting a large matrix. In all cases, in order to remove any dependence upon data, ⎕*RL* is initialized to 16807 before the use of "*?*".

```
     ∇ ID TEST X;C;Z
[1]    C←⊂(Z←'TEST',⍕ID),';⍙T;⍙S'
[2]    C←C,'⍙T←⎕AI[2]' ('⍙S←',X) '⎕AI[2]-⍙T'
[3]    ⍞←(⎕FX C),': '
[4]    ⍎Z
[5]    C←⎕EX Z
     ∇
```

Expression	25 Mhz A3000 Level II	20 Mhz PS/2 APL2/PC
1 *TEST* 'Q*0.5'	680	4510
2 *TEST* '⌹A'	13720	5220

Conclusions

Notwithstanding the few complaints(?) I've made in this review, APL.68000 Level II for the Amiga is an excellent product worth every penny of its very reasonable price. It provides a surprisingly large portion of APL2/370 features for an initial release. MicroAPL's proven track record lends credence to my belief that it won't be long before as much of APL2/370 as is appropriate to a personal desktop computing system will be fully incorporated into Level II. ■

CPCUG APL Lessons Now Available for Several Interpreters

Dick Holt
HRH Systems
Box 4496
Silver Spring, MD 20914
202-586-4449

A series of twenty-six interactive self-teaching on-disk lessons are now available for TryAPL2, IBM APL2, STSC APL★PLUS and Pocket APL, Sharp, and I-APL. Based on the work of Z. V. Jizba, these lessons were transferred to multiple APL formats by the APLSIG of the Capital PC User Group (CPCUG) for use in their 1991 APL classes. Lessons were edited to make them more generic, and to incorporate classroom experience.

CPCUG lessons are downloadable free from the BBS\APL: 301-384-3672, 300/1200/2400 baud, N-8-1, 24 hours a day. See File menu Y.

Lessons may be ordered by mail from HRH Systems at the above address, for US$19, postage-paid worldwide. Mail orders should specify disk size and APL version (.TRY, .ATF, .AWS, .SAW, or .IWS). Checks accepted in any national currency.

Lessons will be sent free upon request to Eastern Europe and the Soviet Union. CPCUG lessons were also a part of the APL91 Software Exchange. ■