Rocket Lander (under 3K bytes to run)

This low-altitude, vertical rocket lander program allows you to choose one of three primaries to land upon, and one of three sets of initial conditions. Note that velocity is negative downwards, positive upwards.

When prompted, enter the rocket fuel burn rate (zero for free-fall), a comma, then the number of seconds for that burn rate to be used. The burn rate may be a positive integer or decimal number, but is limited to a maximum of 100 kg./sec. Time must be an integer number of seconds; any fraction of a second is treated as a whole second.

Remember that the best way to land is to wait as long as possible before making a rocket burn, then burn at the maximum rate to slow down. If you run out of fuel, the ship automatically free-falls to impact. After surface impact, a message and the impact status are printed.

The program correctly allows for the decreasing mass of the ship due to the fuel that has been burned in calculating the change in velocity for the rocket burn. Line 310 of the program slows down the program operation to about one second for each ship's second. Line 330 prints the output headings again if they have been scrolled off the top of the screen.

The excess fuel supplied increases in the order: Moon, Mars, Earth, and also as the altitude increases. Therefore, what appears to be the easiest landing (Moon, low altitude) is actually quite close on fuel. Even on this one, a sucessful landing can be made with over 7 kilograms of fuel left over. Can you do it?

Delmer D. Hinrichs
2116 S. E. 377th Ave.
Washougal, Washington 98671

```
10 CLS : PRINT : PRINT CHR$(23) : PRINT TAB(8) "ROCKET LANDER"
20 PRINT TAB(6) "BY D. D. HINRICHS"
30 '               2116 S.E. 377TH AVE.
40 '               WASHOUGAL, WASHINGTON  98671
50 FOR I=1 TO 1000 : NEXT I                    ' DELAY LOOP
60 CLS : PRINT "DO YOU WANT TO LAND ON:"
70 PRINT "   1.  EARTH"
80 PRINT "   2.  MOON"
90 PRINT "   3.  MARS"
100 INPUT "ENTER NO. FOR DESIRED PRIMARY" ; P
110 ON P GOSUB 550, 560, 570
120 PRINT : PRINT "WHAT INITIAL CONDITIONS DO YOU WANT?"
130 PRINT"  1.  ALTITUDE = 1000 METERS, VELOCITY = -50 M/SEC"
140 PRINT"  2.  ALTITUDE = 10000 METERS, VELOCITY = -100 M/SEC"
150 PRINT"  3.  ALTITUDE = 20000 METERS, VELOCITY = -200 M/SEC"
160 INPUT "ENTER NO. FOR INITIAL CONDITIONS" ; C
170 ON C GOSUB 580, 590, 600
180 M=10000 : TM=M+FU          ' SHIP'S MASS, TOTAL MASS, & FUEL
190 CLS : GOSUB 610            ' PRINT HEADING
200 PRINT@ 64, ; : GOSUB 640   ' SHOW STATUS
210 L=L+1                      ' ENTRY FOR NEXT COMMAND
220 INPUT "ENTER BURN RATE (KG/SEC) ,   NO. OF SEC." ; B, N
230 IF B>100 PRINT "MAX. BURN RATE = 100" : B=100 : L=L+1
240 FOR T=1 TO N
250   NM=TM-B : IF NM<M THEN NM=M : B=TM-M     ' UPDATE MASS
260   NV=V+4000*LOG(TM/NM) ,                   ' UPDATE V FOR ROCKET BURN
270   NV=NV-G                                  ' UPDATE V FOR GRAVITY
280   NA=AL+(V+NV)/2                           ' UPDATE ALTITUDE
290   IF NA<=0 GOTO 380                        ' ON SURFACE?
300   TM=NM : V=NV : AL=NA : FU=TM-M : ET=ET+1   ' NO, UPDATE
310   FOR I=1 TO 300 : NEXT I                  ' DELAY LOOP
320   GOSUB 640                                ' SHOW STATUS
330   IF ET+L>12 GOSUB 610 : PRINT@ 960, ;
340   IF NM=M AND FF=0 THEN FF=1 :PRINT "FREE FALL, FUEL GONE"
350   IF FF=1 THEN B=0 : GOTO 270
360   NEXT T
370 GOTO 210                                   ' RETURN FOR NEXT COMMAND
380   ' INTERPOLATION OF EXACT IMPACT STATUS
390 VI=NV-(V-NV)*NA/(AL-NA)                    ' VELOCITY OF IMPACT
400 TT=ET+1+NA/(AL-NA)                         ' TOTAL TIME
410 RF=NM-M-B*NA/(AL-NA)                       ' REMAINING FUEL
420 PRINT
430 IF VI<-1 PRINT "********  YOU CRASHED  ********" : GOTO 450
440 PRINT "CONGRATULATIONS . . . . . YOU HAVE LANDED SAFELY!"
450 PRINT "YOUR IMPACT VELOCITY WAS" ; ABS(VI) ; "METERS/SEC."
460 PRINT "YOU HAVE" ; RF ; "KG. OF FUEL LEFT"
470 PRINT "IT TOOK YOU" ; TT ; "SECONDS"
480 GOSUB 610 : PRINT@ 960, ;
490 INPUT "DO YOU WANT TO TRY AGAIN (Y/N)" ; Z$
500 IF Z$="Y" RUN
510 CLS : IF VI<-1 PRINT "BETTER LUCK NEXT TIME"
520 PRINT "HAVE A GOOD DAY"
530 END
540   ' - - - ALL SUBROUTINES AFTER THIS - - -
550 G=9.81 : FU=1000 : H$="(EARTH)" : RETURN
560 G=1.62 : FU=200 : H$="(MOON)" : RETURN
570 G=3.74 : FU=500 : H$="(MARS)" : RETURN
580 AL=1000 : V=-50 : RETURN
590 AL=10000 : V=-100 : FU=FU*10 : RETURN
600 AL=20000 : V=-200 : FU=FU*20 : RETURN
610   ' HEADING
620 PRINT@ 0, "TIME, SEC.    ALTITUDE, M.   VELOCITY, M/SEC   ";
630 PRINT "FUEL, KG.    " ; H$ ; : RETURN
640   ' STATUS
650 PRINT USING "######    " ; ET ;
660 PRINT USING "############.###" ; AL, V, FU : RETURN
```

```
DO YOU WANT TO LAND ON:
   1.  EARTH
   2.  MOON
   3.  MARS
ENTER NO. FOR DESIRED PRIMARY? 2

WHAT INITIAL CONDITIONS DO YOU WANT?
  1.  ALTITUDE = 1000 METERS, VELOCITY = -50 M/SEC
  2.  ALTITUDE = 10000 METERS, VELOCITY = -100 M/SEC
  3.  ALTITUDE = 20000 METERS, VELOCITY = -200 M/SEC
ENTER NO. FOR INITIAL CONDITIONS? 1_

TIME, SEC.    ALTITUDE, M.   VELOCITY, M/SEC   FUEL, KG.    (MOON)
    0          1000.000        -50.000        200.000
ENTER BURN RATE (KG/SEC) ,   NO. OF SEC.? 5,5
    1           950.171        -49.659        195.000
    2           900.683        -49.316        190.000
    3           851.538        -48.973        185.000
    4           802.737        -48.629        180.000
    5           754.281        -48.284        175.000
ENTER BURN RATE (KG/SEC) ,   NO. OF SEC.?
    6           706.170        -47.938        170.000
    7           658.406        -47.590        165.000
    8           610.990        -47.242        160.000
    9           563.922        -46.893        155.000
   10           517.204        -46.543        150.000
ENTER BURN RATE (KG/SEC) ,   NO. OF SEC.? _
```

```
TIME, SEC.    ALTITUDE, M.   VELOCITY, M/SEC    FUEL, KG.     (MOON)
   15           288.892         -44.778          125.000
ENTER BURN RATE (KG/SEC) ,    NO. OF SEC. ?
   16           244.292         -44.422          120.000
   17           200.048         -44.065          115.000
   18           156.162         -43.708          110.000
   19           112.633         -43.350          105.000
   20            69.463         -42.991          100.000
ENTER BURN RATE (KG/SEC) ,    NO. OF SEC. ?
   21            26.652         -42.631           95.000

********  YOU CRASHED  ********
YOUR IMPACT VELOCITY WAS 42.4044 METERS/SEC.
YOU HAVE 91.8609 KG. OF FUEL LEFT
IT TOOK YOU 21.6278 SECONDS
DO YOU WANT TO TRY AGAIN (Y/N)? _
```

Orbitor (over 15K bytes to run)

This program allows you to simulate landing a rocket-ship at a specified site from orbit around any one of eight solar system primaries. It accurately simulates the celestial mechanics involved in landing from orbit.

The ship is initially set at a random position on an elliptical orbit, either clockwise or counterclockwise. To keep track of the ship's position accurately enough to allow a smooth landing, it is necessary to use double precision calculations. The two-dimensional ship's orbit is stored in rectangular coordinates (X, Y), but all displays are in polar coordinates (magnitude, angle) to make them easier to interpret.

The program allows you to save and recall the ship's status at any time on cassette tape. This is especially useful to save the status before trying a potentially "hazardous" maneuver.

You may select either a graphic output display (with only the current status shown numerically), or a tabular display (showing numerical data for several past time segments). When near to landing (altitude under 10 km.) the altitude display changes from km. to meters, and the velocity display changes from km./sec. to meters/sec. If the graphic display was selected, it changes from showing the entire primary and its surrounding space to show a 10 km. high by 20 km. wide area, centered on the landing site.

The ship's motion is calculated as a series of segments. The segment time may be set to any integer or decimal number up to 60 seconds. If no entry is made, the program uses 60 seconds. The number of segments to be calculated at one time may be set to any integer. If no entry is made, the program uses one segment. The rocket fuel burn rate may be set to any integer or decimal number up to 100 kg./sec. If no entry is made, the program uses zero burn rate (free-fall). Thus three quick presses of "ENTER" without entries will give one, 60 second free-fall segment. If the burn rate was not zero, a rocket thrust direction must be selected, from 0° to $\pm 180^{\circ}$. If no entry is made, the program uses 0° (straight up when landing at the landing site). If at any time during the calculation of a series of segments, you press any key, the program will stop, and prompt for your input. The status remains the last status displayed.

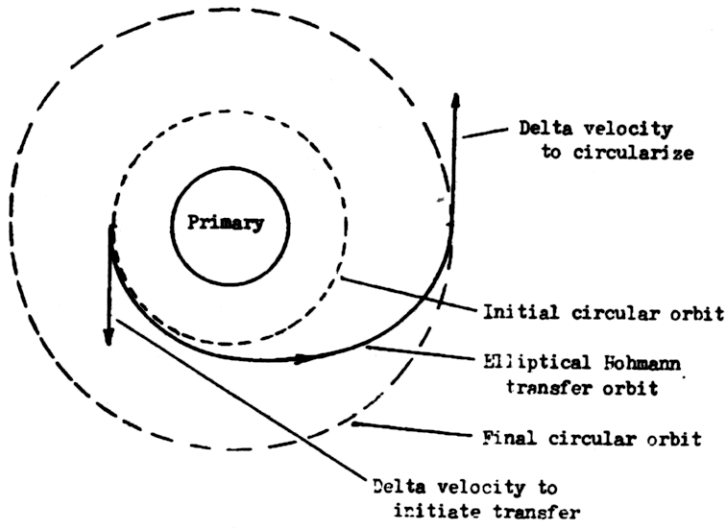
The ship has an "on-board computer", to calculate various things to help you to pilot the ship correctly. To use it, enter "99" as a segment time. (This also gives access to the routine to save the ship's status on tape). These calculations are pretty much self-explanatory except for the Hohmann transfer: A Hohmann transfer is a minimum-fuel-usage maneuver to go from one circular orbit to another circular orbit at a different altitude. It is half of an ellipse, tangent to one circular orbit at each end. See sketch. The delta velocities are the changes in the velocity required.

Delmer D. Hinrichs
2116 S. E. 377th Ave.
Washougal, Washington 98671

P.S.

This is a long program to key in. For $2.00 I'll record it on a cassette and send it to you.

D. D. H.



Typical Hohmann Transfer

```
10 CLS : RANDOM
20 PRINT"*   *   ROCKET-SHIP LANDING FROM RANDOM ORBIT   *   *"
30 '  (C) BY D. D. HINRICHS  1979
40 '          2116 S.E. 377TH AVE.
50 '          WASHOUGAL, WN.  98671
60 DEFINT I-N : DEFDBL P-Y : M=1E4 : PI=3.1415926535897932
70 Z$="" : PRINT : INPUT"DO YOU WANT INSTRUCTIONS (Y OR N)"; Z$
80 IF Z$="Y" GOSUB 3310 : GOTO 110
90 IF Z$="N" GOTO 110
100 PRINT "YOU MUST ANSWER 'Y' OR 'N'.      TRY AGAIN" : GOTO 70
110 PRINT : PRINT "SELECT THE PRIMARY THAT YOU WISH TO ORBIT."
120 PRINT "YOUR FUEL SUPPLY WILL BE ADJUSTED ACCORDINGLY."
130 PRINT "  1.  MERCURY"
140 PRINT "  2.  THE MOON"
150 PRINT "  3.  MARS"
160 PRINT "  4.  IO"
170 PRINT "  5.  EUROPA"
180 PRINT "  6.  GANYMEDE"
190 PRINT "  7.  CALLISTO"
```

```
200 PRINT "  8.  TITAN"
210 PRINT "  9.  RECALL PREVIOUS STATUS FROM CASSETTE TAPE"
220 N=0 : INPUT "WHICH DO YOU WANT TO TRY" ; N : CLS
230 IF N=9 GOTO 3220
240 IF N>0 AND N<9 GOTO 260
250 PRINT "YOUR ENTRY"; N ; "IS ILLEGAL.  TRY AGAIN" : GOTO 110
260 INPUT "DO YOU WANT GRAPHIC OR TABULAR DISPLAY (G OR T)"; Z$
270 IF Z$="G" OR Z$="T" GOTO 290
280 PRINT "YOU MUST ANSWER 'G' OR 'T'.    TRY AGAIN" : GOTO 260
290 CLS : PRINT "SELECTING RANDOM ORBIT"
300 RESTORE : FOR J=1 TO N*4-3 : READ H$ : NEXT J
310 READ H$, GC, SR, SM
320 AN=RND(0)*2*PI-PI : RD=SR+100+RND(0)*2400
330 XP=RD*COS(AN) : YP=RD*SIN(AN)   '  SHIP'S X & Y POSITION
340 VL=SQR(GC/RD)                   '  SHIP'S VELOCITY
350 IF RND(0)>.5 THEN AN=AN+PI/2 ELSE AN=AN-PI/2
360 XV=VL*COS(AN) : YV=VL*SIN(AN)   '  SHIP'S X & Y VELOCITY
370 AV=RND(0)*2*PI-PI : R=SR/RD     '  ANGLE FOR RANDOM DELTA V
380 DV=VL*(SQR(2*R/(1+R))-1)        '  DELTA VELOCITY TO SURFACE
390 XV=XV+RND(0)*DV*COS(AV) : YV=YV+RND(0)*DV*SIN(AV)
400 IF Z$="G" THEN IG=1 ELSE GOSUB 2790
410 GOSUB 2450
420 IF IG=0 INPUT "PRESS 'ENTER'" ; Z$ : CLS : L=0 : GOSUB 2450
430   '
440   '  REENTRY POINT, COMMAND FOR NEXT SET OF SEGMENTS
450 ST=0 : INPUT "SEGMENT TIME, SEC.  = " ; ST : GOSUB 3050
460 ST=ABS(ST) : IF ST=99 CLS : GOTO 1320   '  ON-BOARD COMPUTER
470 IF ST>60 PRINT "TIME REDUCED TO 60 SEC." :ST=60 :GOSUB 3020
480 IF ST=0 THEN ST=60              '  DEFAULT
490 NS=0 : INPUT "NO. OF SEGMENTS = " ; NS : GOSUB 3050
500 NS=ABS(NS) : IF NS=0 THEN NS=1  '  DEFAULT
510 RB=0 : INPUT "BURN RATE, KG./SEC.  = "; RB : GOSUB 3050
520 RB=ABS(RB) : IF RB=0 GOTO 580   '  DEFAULT, FREE-FALL
530 IF RB>100 PRINT"BURN RATE REDUCED TO 100":RB=100:GOSUB 3020
540 DT=0 : INPUT "THRUST DIRECTION, DEG.  = " ; DT : GOSUB 3050
550 IF ABS(DT)<=180 THEN DR=DT*PI/180 : GOTO 580
560 PRINT "ANGLE"; DT; "IS TOO GREAT.  TRY AGAIN" : GOSUB 3020
570 GOTO 540
580 FOR K=1 TO NS                   '  CALCULATE EACH SEG.
590   IF RB=0 THEN XR=XV :YR=YV :SN=SM :GOTO 640   '  FREE-FALL
600   SN=SM-ST*RB                   '  SHIP'S NEW MASS
610   IF SN<M PRINT "OUT OF FUEL" : SN=M : RB=0 : GOSUB 3020
620   DV=4*LOG(SM/SN)               '  BURN DELTA VELOCITY
630   XR=XV+DV*COS(DR) : YR=YV+DV*SIN(DR)   '  X & Y VEL AFTER B
640   XG=(XR+XV)/2 : YG=(YR+YV)/2   '  VEL. TO EST. GRAVITY
650   PX=XP+XG*ST/2 : PY=YP+YG*ST/2 '  POS. TO EST. GRAVITY
660   RG=SQR(PX*PX+PY*PY)           '  RADIUS FOR GRAVITY
670   RG=RG-GC*(ST/RG)*(ST/RG)/16   '  RADIUS CORRECTION
680   QA=ATN(PY/PX)                 '  ANGLE FOR GRAVITY
690   IF PX<0 THEN QA=QA+PI : IF PY<0 THEN QA=QA-2*PI
700   VG=GC/(RG*RG)*ST              '  GRAVITY DELTA VEL
710   VX=XR-VG*COS(QA) : VY=YR-VG*SIN(QA)   '  FINAL VELOCITY
720   PX=XP+(XV+VX)*ST/2 :PY=YP+(YV+VY)*ST/2   '  FINAL POSITION
730   P=PX*PX+PY*PY : R=SQR(P)      '  SHIP'S RADIUS
740   IF INKEY$<>"" GOTO 440        '  EMERGENCY STOP?
750   IF R-SR<10 GOTO 780           '  ALT. < 10 KM.?
760   IF LD=1 THEN LD=0 : L=0 : NG=0 : CLS : IF IG=0 THEN NG=1
770   GOTO 810
780   S=R : R=(P/S+S)/2 : IF S<>R GOTO 780   '  ADJUST RADIUS
790   IF LD=0 THEN LD=1 : L=0 : LG=0 : CLS : IF IG=0 THEN LG=1
800   IF R-SR<=0 GOTO 860           '  SURFACE IMPACT?
810   RD=R :XP=PX :YP=PY :XV=VX :YV=VY :SM=SN :TM=TM+ST/60
820   GOSUB 2450                    '  LIST NEW STATUS
830   NEXT K                        '  NEXT SEGMENT
840 GOTO 440                        '  NEXT COMMAND
850   '
860   '  IMPACT ROUTINE
870 Q=XP*XP+YP*YP : RI=SQR(Q)       '  INITIAL SHIP'S RADIUS
880 R=RI : RI=(Q/R+R)/2 : IF R<>RI GOTO 880   '  ADJUST RADIUS
890 Q=PX*PX+PY*PY : RF=SQR(Q)       '  FINAL SHIP'S RADIUS
900 R=RF : RF=(Q/R+R)/2 : IF R<>RF GOTO 900   '  ADJUST RADIUS
910 RA=(SR-RF)/(RI-RF)              '  RATIO, C/D
920 FV=SQR(VX*VX+VY*VY)             '  FINAL VELOCITY
930 XI=VX+RA*(XV-VX) : YI=VY+RA*(YV-VY)   '  X & Y IMPACT VEL.
940 GV=(FV+RA*(BV-FV))*1000         '  IMPACT VELOCITY
950 PRINT
960 IF GV>1 PRINT "********  YOU CRASHED  ******** "; :GOTO 980
970 PRINT "CONGRATULATIONS  -  -  -  YOU ARE DOWN SAFELY" ;
980 PRINT " ON " ; H$ ; "!!!!"
990 PRINT "YOUR IMPACT VELOCITY WAS " ; GV ; " METERS/SECOND"
1000 DI=ATN(YI/XI)*180/PI           '  DIRECTION OF IMPACT
1010 IF XI<0 THEN DI=DI+180 : IF YI<0 THEN DI=DI-360
1020 PRINT "YOUR DIRECTION OF IMPACT WAS " ; DI ; " DEGREES"
1030 AF=ATN(PY/PX)                  '  FINAL POSITION ANGLE
1040 IF PX<0 THEN AF=AF+PI : IF PY<0 THEN AF=AF-2*PI
1050 AI=ATN(YP/XP)                  '  INITIAL POSIT. ANGLE
1060 IF XP<0 THEN AI=AI+PI : IF YP<0 THEN AI=AI-2*PI
1070 DM=SR*(AF+RA*(AI-AF))*1000     '  MISS DISTANCE
1080 PRINT "YOU MISSED THE LANDING SITE BY " ; DM ; " METERS"
1090 FR=SN+RA*(SM-SN)-M             '  FUEL REMAINING
1100 PRINT "YOU HAVE " ; FR ; " KILOGRAMS OF FUEL LEFT"
1110 CT=TM+(ST-RA*ST)/60            '  CUM. TIME TO IMPACT
1120 PRINT "IT TOOK YOU " ; CT ; " MINUTES TO GET DOWN"
1130 Z$="" : PRINT
1140 IF IG=0 GOSUB 2360 : PRINT@ 960, ;
1150 INPUT "DO YOU WANT TO TRY ANOTHER LANDING (Y OR N)" ; Z$
1160 IF Z$= "Y" RUN
1170 PRINT : PRINT "UNTIL NEXT TIME, THEN"
1180 END
1190   '
1200   '  CONSTANTS FOR VARIOUS PRIMARYS
1210 DATA 0, "MERCURY", 21522, 2440, 40000
1220 DATA "THE MOON", 4901, 1739, 25000
1230 DATA "MARS", 43043, 3393, 60000
1240 DATA "IO", 5790, 1820, 25000
1250 DATA "EUROPA", 3190, 1525, 20000
1260 DATA "GANYMEDE", 9740, 2635, 30000
1270 DATA "CALLISTO", 6420, 2450, 25000
1280 DATA "TITAN", 9170, 2900, 30000
1290   '
1300   '  -  -  -  ALL SUBROUTINES AFTER THIS  -  -  -
1310   '
1320   '  THE "ON-BOARD COMPUTER" ROUTINE, TO ASSIST PILOTING
1330 PRINT : PRINT "WHAT DO YOU WANT TO CALCULATE?" ;
1340 PRINT "   (FOR LANDING ON " ; H$ ; ")"
1350 PRINT "  1.  VELOCITY TO MAINTAIN A CIRCULAR ORBIT"
1360 PRINT "  2.  HOHMANN TRANSFER TO A DIFFERENT ALTITUDE"
1370 PRINT "  3.  ORBITAL PERIOD"
1380 PRINT "  4.  DELTA VELOCITY TO CHANGE STATUS"
1390 PRINT "  5.  FUEL USAGE FOR GIVEN DELTA VELOCITY"
1400 PRINT "  6.  SAVE SHIP'S CURRENT STATUS ON CASSETTE TAPE"
1410 PRINT "  7.  END CALCULATIONS AND RETURN TO PILOTING"
1420 N=0 : INPUT "ENTER NO. OF DESIRED ROUTINE" ; N : CLS
1430 IF N=7 THEN L=0 : LG=0 : NG=0 : GOSUB 2450 : GOTO 440
1440 IF N>0 AND N<7 GOTO 1460
1450 PRINT "YOUR ENTRY", N;"IS ILLEGAL.  TRY AGAIN" : GOTO 1320
```

```
1460 ON N GOSUB 1490, 1580, 1830, 1940, 2160, 2150
1470 CLS : GOTO 1320
1480   '
1490 CLS:PRINT"*   *   *   CIRCULAR ORBIT VELOCITY   *   *   *"
1500 AL=0 : PRINT : INPUT "ENTER ALTITUDE IN KILOMETERS" ; AL
1510 IF AL<0 PRINT"ALTITUDE < 0 IS ILLEGAL.  TRY AGAIN":GOTO1500
1520 CV=SQR(GC/(AL+SR))
1530 PRINT :PRINT "CIRCULAR ORBIT VELOCITY = "; CV ; "KM./SEC."
1540 GOSUB 2300
1550 IF Z$="A" GOTO 1490
1560 IF Z$="R" RETURN
1570   '
1580 CLS : PRINT "*   *   HOHMANN TRANSFER CALCULATION   *   *"
1590 AI=0 : AF=0 : PRINT
1600 INPUT"ENTER INITIAL , FINAL ALTITUDES IN KILOMETERS"; AI, AF
1610 IF AI>=0 AND AF>=0 GOTO 1630
1620 PRINT "ALTITUDE < 0 IS ILLEGAL.  TRY AGAIN" : GOTO 1590
1630 R=(AF+SR)/(AI+SR)          '  RATIO OF INITIAL, FINAL RADII
1640 CV=SQR(GC/(AI+SR))         '  INITIAL CIRC. ORBIT VELOCITY
1650 DI=CV*(SQR(2*R/(1+R))-1)   '  DELTA V TO INITIATE TRANSFER
1660 CO=CV*(SQR(1/R)-SQR(2/(R*(1+R))))  ' TO CIRCULARIZE ORBIT
1670 FV=SQR(GC/(AF+SR))         '  FINAL CIRCULAR ORBIT VELOCITY
1680 A=(AI+AF)/2+SR             '  AVERAGE RADIUS
1690 HT=PI*SQR(A*A*A/GC)/60     '  HOHMANN TRANSFER TIME
1700 PRINT
1710 PRINT "INITIAL CIRCULAR ORBIT VELOCITY = "; CV; "KM./SEC."
1720 PRINT "DELTA VEL. TO INITIATE TRANSFER = "; DI; "KM./SEC."
1730 PRINT"INITIAL TRANSFER ORBIT VELOCITY = ";CV+DI;"KM./SEC."
1740 PRINT "HOHMANN TRANSFER TIME = "; HT ; "MINUTES"
1750 PRINT "FINAL TRANSFER ORBIT VELOCITY = ";FV-CO; "KM./SEC."
1760 PRINT "DELTA VEL. TO CIRCULARIZE ORBIT = "; CO; "KM./SEC."
1770 PRINT "FINAL CIRCULAR ORBIT VELOCITY = "; FV ; "KM./SEC."
1780 PRINT "TOTAL DELTA V FOR ORBIT CHANGE = ";DI+CO;"KM./SEC."
1790 GOSUB 2300
1800 IF Z$="A" GOTO 1580
1810 IF Z$="R" RETURN
1820   '
1830 CLS:PRINT"*   *   *   ORBIT PERIOD CALCULATION   *   *   *"
1840 AX=0 : AI=0 : PRINT
1850 INPUT "ENTER MAXIMUM , MINIMUM ALTITUDES IN KM." ; AX, AI
1860 IF AX<0 OR AI<0 PRINT"ALTITUDE < 0.   TRY AGAIN" :GOTO 1840
1870 A=(AX+AI)/2+SR                    '  AVERAGE RADIUS
1880 OP=2*PI*SQR(A*A*A/GC)/60          '  ORBITAL PERIOD
1890 PRINT : PRINT "PERIOD FOR ONE ORBIT = " ; OP ; "MINUTES"
1900 GOSUB 2300
1910 IF Z$="A" GOTO 1830
1920 IF Z$="R" RETURN
1930   '
1940 CLS:PRINT"*   *   DELTA VELOCITY TO CHANGE STATUS   *   *"
1950 AD=0 : AV=0 : DD=0 : DV=0 : PRINT
1960 INPUT "ENTER INITIAL DIRECTION, DEGREES" ; AD
1970 IF ABS(AD)>180 PRINT"ANGLE IS > 180.   TRY AGAIN":GOTO 1950
1980 INPUT "ENTER INITIAL VELOCITY, KM./SEC." ; AV
1990 IF AV<0 OR AV>9 PRINT AV;"IS ILLEGAL, TRY AGAIN":GOTO 1980
2000 INPUT "ENTER DESIRED DIRECTION, DEGREES" ; DD
2010 IF ABS(DD)>180 PRINT"ANGLE IS > 180.   TRY AGAIN":GOTO 2000
2020 INPUT "ENTER DESIRED VELOCITY, KM./SEC." ; DV
2030 IF DV<0 OR DV>9 PRINT DV;"IS ILLEGAL, TRY AGAIN":GOTO 2020
2040 AD=AD*PI/180 : DD=DD*PI/180        '  ANGLES IN RADIANS
2050 DX=DV*COS(DD)-AV*COS(AD)           '  DELTA X VELOCITY
2060 DY=DV*SIN(DD)-AV*SIN(AD)           '  DELTA Y VELOCITY
2070 DV=SQR(DX*DX+DY*DY)                '  DELTA VELOCITY
2080 DD=ATN(DY/DX)*180/PI               '  DIRECTION, DEGREES
2090 IF DX<0 THEN DD=DD+180 : IF DY<0 THEN DD=DD-360
2100 PRINT :PRINT"DIRECTION OF DELTA VELOCITY = ";DD; "DEGREES"
2110 PRINT "REQUIRED DELTA VELOCITY = " ; DV ; "KM./SEC."
2120 GOSUB 2300
2130 IF Z$="A" GOTO 1940
2140 IF Z$="R" RETURN
2150   '
2160 CLS :PRINT"FUEL USAGE REQUIRED FOR A GIVEN DELTA VELOCITY"
2170 PRINT "(ASSUMING A 4 KM./SEC. ROCKET EXHAUST VELOCITY)"
2180 DV=0 : H=0 : PRINT
2190 INPUT "DESIRED DELTA VELOCITY, KM./SEC." ; DV
2200 IF DV<0 OR DV>9 PRINT DV;"IS ILLEGAL, TRY AGAIN":GOTO 2180
2210 INPUT "SHIP'S TOTAL MASS, KG.  (DEFAULT, PRESENT MASS)"; H
2220 IF H<0 OR H>1E5 PRINT H;"IS ILLEGAL, TRY AGAIN" :GOTO 2210
2230 IF H=0 THEN H=SM
2240 F=H-H/EXP(DV/4)
2250 PRINT : PRINT "REQUIRED FUEL USAGE = " ; F ; "KILOGRAMS"
2260 GOSUB 2300
2270 IF Z$="A" GOTO 2160
2280 IF Z$="R" RETURN
2290   '
2300   '  'A' AND 'R' PRINT ROUTINES
2310 PRINT : Z$=""
2320 INPUT "ENTER 'A' FOR ANOTHER, 'R' TO RETURN TO MENU" ; Z$
2330 IF Z$="A" OR Z$="R" RETURN
2340 PRINT "YOU MUST ANSWER 'A' OR 'R'.   TRY AGAIN" :GOTO 2300
2350   '
2360   '  STATUS DISPLAY HEADING
2370 PRINT@ 0,"   TIME,     -----POSITION-----     -----" ;
2380 PRINT "VELOCITY-----  REMAINING"
2390 IF LD=1 GOTO 2420          '  NEAR LANDING?
2400 PRINT@ 64,"    MIN.     KM. ALT.    DEGREES    KM./SEC." ;
2410 PRINT "   DEGREES  FUEL, KG." ; : RETURN
2420 PRINT@ 64,"    MIN.     METER A.    DEGREES    METER/S." ;
2430 PRINT "   DEGREES  FUEL, KG." ; : RETURN
2440   '
2450   '  CALCULATE STATUS DISPLAY
2460 AL=RD-SR                   '  ALTITUDE FROM SURFACE
2470 DP=ATN(YP/XP)*180/PI       '  DIRECTION FROM PRIMARY, DEG
2480 IF XP<0 THEN DP=DP+180 : IF YP<0 THEN DP=DP-360
2490 BV=SQR(XV*XV+YV*YV) :CV=BV  '  SHIP'S NEW VELOCITY
2500 DV=ATN(YV/XV)*180/PI       '  SHIP'S VEL. DIRECTION, DEG
2510 IF XV<0 THEN DV=DV+180 : IF YV<0 THEN DV=DV-360
2520 FU=SM-M                    '  SHIP'S REMAINING FUEL
2530 IF L<>0 OR IG=0 GOTO 2560  '  NO GRAPHICS BACKGROUND?
2540 IF LD=0 AND NG=0 GOSUB 2790        '  ORBIT GRAPHICS?
2550 IF LD=1 AND LG=0 GOSUB 2960        '  LANDING GRAPHICS?
2560 IF LD=1 THEN AL=AL*1000 : CV=CV*1000  '  CONVERT TO METERS
2570 IF L=0 GOSUB 2360 : PRINT          '  PRINT HEADING?
2580   '  PRINT CURRENT STATUS
2590 IF IG=1 PRINT@ 128, ;
2600 PRINT USING "#####.###" ; CSNG(TM) ;
2610 PRINT USING "########.###" ; AL, DP, CV, DV, FU
2620 IF IG=0 AND JG=1 GOSUB 3050 : RETURN   '  NO GRAPHICS?
2630 IF LD=1 THEN L=1 : GOTO 2720       '  NEAR LANDING?
2640   '
2650   '  ORBITING GRAPHICS, SHIP
2660 JG=1 : IX=64-YP/SR*14
2670 IF IX<0 OR IX>127 RETURN
2680 IY=30-XP/SR*6
2690 IF IY<9 OR IY>47 RETURN
2700 SET(IX,IY) : SET(IX+1,IY) : RETURN
2710   '
2720   '  LANDING GRAPHICS, SHIP
2730 IX=64-YP*6.4
```

```
2740 IF IX<0 OR IX>127 RETURN
2750 IY=44-(XP-SR)*3.2
2760 IF IY<9 OR IY>44 RETURN
2770 SET(IX,IY) : RETURN
2780   '
2790   '  ORBIT GRAPHICS, BACKGROUND
2800 CLS
2810 PRINT@ 538, CHR$(176); CHR$(156); CHR$(134);
2815 PRINTSTRING$(7,CHR$(131));CHR$(140);CHR$(180);CHR$(144)
2820 PRINT@601,CHR$(190);CHR$(129);"           ";CHR$(171);CHR$(148)
2830 PRINT@665,CHR$(175);CHR$(144);"           ";CHR$(186);CHR$(133)
2840 PRINT@ 730, CHR$(131); CHR$(141); CHR$(164);
2845 PRINTSTRING$(7,CHR$(176));CHR$(140);CHR$(135);CHR$(129)
2850 FOR I=0 TO 9
2860   PRINT@ 352+(I*64), CHR$(191);
2870   NEXT I
2880 PRINT@ 982, "+180 DEG. " ; CHR$(191) ; " -180 DEG.";
2890 PRINT@ 576, "+90 DEG." ; : PRINT@ 632, "-90 DEG.";
2900 GOSUB 3120
2910 FOR I=0 TO 127 : SET(I,30) : NEXT I
2920 PRINT@ 604, H$ ;
2930 PRINT@ 960, "RADIUS =" ; SR ; "KM.";
2940 NG=1 : RETURN
2950   '
2960   '  LANDING GRAPHICS, BACKGROUND
2970 CLS
2980 PRINT@ 896, ; : FOR J=1 TO 64 : PRINT CHR$(176) ; : NEXT J
2990 PRINT@978,"LANDING SITE "; CHR$(190); CHR$(189);" ON ";H$;
3000 LG=1 : RETURN
3010   '
3020   '  DELAY LOOP
3030 FOR J=1 TO 1000 : NEXT J
3040   '
3050   '  LINE CHECK ROUTINE
3060 L=L+1 : IF L>13 AND IG=0 GOSUB 2360 : PRINT@ 960, ;
3070 IF IG=0 RETURN
3080   '
3090   '  CLEAR INPUT LINE AND REPLACE LEGEND
3100 PRINT@ 192, STRING$(36,CHR$(128))
3110 IF LD=1 GOTO 3130
3120 PRINT@ 288, CHR$(191) ; " 0 DEG.   (LANDING SITE)" ;
3130 PRINT@ 192, ; : RETURN
3140   '
3150   '  SAVE CURRENT STATUS ON TAPE
3160 CLS : O$="ORBIT"
3170 INPUT "POSITION TAPE, PRESS 'RECORD' THEN 'ENTER'"; Z$
3180 CLS : PRINT "SAVING SHIP'S CURRENT STATUS"
3190 PRINT#-1, O$,H$,GC,RD,SR,SM,TM,XP,YP,XV,YV,BV,IG,LD
3200 RETURN
3210   '
3220   '  GET PREVIOUS STATUS FROM TAPE
3230 O$=""
3240 INPUT "POSITION TAPE, PRESS 'PLAY' THEN 'ENTER'"; Z$
3250 CLS : PRINT "RESTORING SHIP'S PREVIOUS STATUS"
3260 INPUT#-1, O$,H$,GC,RD,SR,SM,TM,XP,YP,XV,YV,BV,IG,LD
3270 IF O$="ORBIT"THEN L=0:LG=0:JG=1:NG=0 :GOSUB 2450 :GOTO 440
3280 PRINT : PRINT "******  WRONG FILE  ******"
3290 PRINT "TRY AGAIN" : PRINT : GOTO 3240
3300   '
3310   '  INSTRUCTIONS
3320 CLS
3330 PRINT "YOU ARE THE PILOT OF A ROCKET SHIP IN A RANDOM " ;
3340 PRINT "ELLIPTICAL ORBIT"
3350 PRINT "AROUND ONE OF EIGHT NEARLY AIRLESS PRIMARIES.  " ;
3360 PRINT "YOU MUST CONTROL"
3370 PRINT "THE TIMING, DURATION, INTENSITY, AND DIRECTION " ;
3380 PRINT "OF THE ROCKET"
3390 PRINT "BURNS SO AS TO LAND GENTLY ( < 1 METER/SEC. ) AT " ;
3400 PRINT "THE TARGET SITE"
3410 PRINT "(AT 0 DEG.) WITHOUT RUNNING OUT OF FUEL.   YOU " ;
3420 PRINT "MAY RUN ANY"
3430 PRINT "NUMBER OF FREE-FALL OR ROCKET BURN SEGMENTS " ;
3440 PRINT "AUTOMATICALLY."
3450 PRINT "OPERATION OF THE ROCKET SHIP IS AS FOLLOWS:"
3460 PRINT "  1.  CHOOSE PRIMARY FOR LANDING.  (A RANDOM " ;
3470 PRINT "ORBIT IS SELECTED)"
3480 PRINT "  2.  ENTER SECONDS FOR EACH SEGMENT.  (MAXIMUM " ;
3490 PRINT "& DEFAULT = 60)"
3500 PRINT "      (ENTER '99' FOR HELP FROM THE 'ON-BOARD " ;
3510 PRINT "COMPUTER')"
3520 PRINT "  3.  ENTER NUMBER OF SEGMENTS.  (DEFAULT = 1)"
3530 PRINT "  4.  ENTER BURN RATE, KG./SEC.  (MAXIMUM = 100, ";
3540 PRINT "  DEFAULT = 0)"
3550 PRINT "  5.  IF BURN WAS MADE, ENTER THRUST DIRECTION.  ";
3560 PRINT "(TO + OR -180)"
3570 PRINT "  6.  EVALUATE DISPLAY, THEN RETURN TO STEP 2."
3580 PRINT "  7.  AFTER SURFACE IMPACT, LANDING STATUS IS " ;
3590 PRINT "INTERPOLATED."
3600 INPUT "TO CONTINUE, PRESS 'ENTER'" ; Z$
3610 CLS : RETURN
```

```
     *   *   ROCKET-SHIP LANDING FROM RANDOM ORBIT   *   *

     DO YOU WANT INSTRUCTIONS (Y OR N)? N

     SELECT THE PRIMARY THAT YOU WISH TO ORBIT.
     YOUR FUEL SUPPLY WILL BE ADJUSTED ACCORDINGLY.
       1.  MERCURY
       2.  THE MOON
       3.  MARS
       4   IO
       5.  EUROPA
       6.  GANYMEDE
       7.  CALLISTO
       8.  TITAN
       9.  RECALL PREVIOUS STATUS FROM CASSETTE TAPE
     WHICH DO YOU WANT TO TRY? 2_


    TIME,      -----POSITION-----      -----VELOCITY-----   REMAINING
     MIN.      KM. ALT.    DEGREES     KM./SEC.    DEGREES  FUEL, KG.
    0.000    1704.200      16.353       1.249     -75.355  15000.000
 SEGMENT TIME, SEC. = ? 25
 NO. OF SEGMENTS = ? 6
 BURN RATE, KG./SEC. = ? 25
 THRUST DIRECTION, DEG. = ? 5
    0.417    1704.520      15.830       1.269     -71.312  14375.000
    0.833    1707.400      15.299       1.297     -67.303  13750.000
    1.250    1712.910      14.741       1.331     -63.364  13125.000
    1.667    1721.140      14.213       1.372     -59.526  12500.000
    2.083    1732.160      13.671       1.421     -55.815  11875.000
    2.500    1746.070      13.120       1.476     -52.252  11250.000
 SEGMENT TIME, SEC. = ? _
```

```
1000 ' === GENERALIZED "MENU" PROGRAM FOR TRS-80 DISK BASIC ===
1010 ' STEVE MACGREGOR, 3701 W WETHERSFIELD, PHOENIX, AZ  85029
1020 ' ================================================
1030 '
1040 '                --- INSTRUCTIONS ---
1050 '
1060 ' PLACE A COPY OF THIS PROGRAM ON EACH DISK CONTAINING
1070 ' BASIC-LANGUAGE PROGRAMS, AND INSERT DATA STATEMENTS IN
1080 ' LINES 6000-6230 CONTAINING THE FILENAME AND A SHORT
1090 ' DESCRIPTION OF EACH PROGRAM.  (SEE LINES 5000- FOR THE
1100 ' FORMAT OF THE DATA STATEMENTS.)
1110 '
1120 ' KEEP THE LIST OF PROGRAMS CURRENT BY ADDING A DATA LINE
1130 ' FOR EACH NEW PROGRAM ADDED TO THE DISK, AND DELETING THE
1140 ' LINE FOR EACH PROGRAM DELETED.
1150 '
1160 ' WHENEVER THE BASIC COMES ON-LINE, TYPE //RUN "MENU"// TO
1170 ' LOAD AND EXECUTE THE MENU PROGRAM, WHICH WILL DISPLAY A
1180 ' LIST OF THE PROGRAMS ON THE DISK.  THEN DO ONE OF THE
1190 ' FOLLOWING:
1200 '
1210 ' TO LOAD AND EXECUTE A PARTICULAR PROGRAM FROM THE MENU,
1220 ' TYPE THE NUMBER OPPOSITE THE DESCRIPTION, AND PRESS THE
1230 ' //ENTER// KEY.
1240 '
1250 ' TO EXIT TO BASIC, PRESS THE //BREAK// KEY.
1260 '
1270 ' TO EXIT TO TRSDOS, TYPE A ZERO (0), AND PRESS //ENTER//.
1280 '
1290 ' IF THE DESIRED PROGRAM IS NOT ON THIS DISK, REMOVE THE
1300 ' DISK AND INSERT THE CORRECT ONE.  THEN PRESS //ENTER//
1310 ' TO LOAD THE "MENU" PROGRAM FROM THE NEW DISK.
1320 '
1330 ' WHEN A PROGRAM HAS COMPLETED, IT IS MOST CONVENIENT TO
1340 ' HAVE IT TERMINATE BY EXITING TO THE MENU INSTEAD OF TO
1350 ' BASIC, BY EXECUTING THE STATEMENT //RUN "MENU"// AS THE
1360 ' LAST LINE OF THE PROGRAM, INSTEAD OF //END//.
1370 '
1380 CLEAR 50: DIM P$(24)
2000 CLS: FOR I=24 TO 88 STEP 64: PRINT@I,;: FOR J=0 TO 11
2010 READ T: PRINT CHR$(128+T);: NEXT J,I
2020 DATA 29,24,21,55,51,1,29,16,21,21,0,21
2030 DATA 5,0,5,13,12,4,5,2,5,9,12,1
3000 P=0: L=128
3010 READ P$: IF P$="" THEN 4000
3020 P=P+1: IF P=13 THEN L=160
3030 P$(P)=P$: READ P$
3040 PRINT@L,USING "##) ";P,: PRINT P$;: L=L+64: GOTO 3010
4000 PRINT@896,CHR$(31);
4010 LINEINPUT "SELECTION";P$: IF P$="" THEN RUN "MENU"
4020 K=VAL(P$): IF K=0 THEN CMD "S"
4030 IF K<0 OR K>P THEN 4000
4040 RUN P$(K)
5000 ' -- DATA CONSISTS OF PROGRAM FILENAME AND DESCRIPTION
5010 '    THE MAXIMUM NUMBER OF PROGRAMS IS 24.  IF TWELVE OR
5020 '    FEWER PROGRAMS ARE LISTED, THE DESCRIPTION MAY BE
5030 '    UP TO 60 CHARACTERS LONG.  IF THIRTEEN OR MORE, THE
5040 '    DESCRIPTION MAY BE UP TO 26 CHARACTERS.
5060 ' -- FORMAT OF DATA LINE:
5070 '    6000 DATA FILENAME/BAS,"DESCRIPTION"
7000 DATA "": REM -- MUST BE LAST DATA LINE
```

> 16 GROUP

Hello, My name is Robin Lloyd and I am 15 years old, and going into the tenth grade. I am the editors daugter. I would like to start a little something for the young people who have fathers who are in the computer field and subscribe to the newsletter. I was thinking about starting a pen pal system. All you have to do is send me your name, age, address, hobbies and interests on a postcard or in a letter and I will have them printed up in the newsletter and if you find some one listed who has the same interests that you do then all you have to do is to write them and maybe you will meet some new friends. I would really like your cooperation on this project and I feel that it will help all of us to learn a little bit more about our parents' work and maybe it will help us to get interested in the same sort of job. If you have any questions about this project then feel free to write me and I will answer your questions as well as I can.

Send your information to:

Miss Robin Lloyd
7554 Southgate Rd.
Fayetteville, NC 28304

Thanks,

Robin

ROBIN HOPE LLOYD
AGE 15
7554 SOUTHGATE RD.
FAYETTEVILLE, NC 28304

HOBBIES

ROLLERSKATING
WATCHING TELEVISION
MUSIC
GOING TO THE MOVIES
SINGING
DISCO DANCING

INTEREST

COMPUTERS
LISTENING TO MUSIC FROM THE 1950's
ARTS AND CRAFTS
READING EDGAR ALLEN POE STORIES

T H E B I T P I C K E R ' S T O O L B O X

By Steve MacGregor, 3701 W Wethersfield, Phoenix, Arizona 85029

This machine-language routine will convert a binary number from ØØØØ to 1111 in the A register into an ASCII hex digit:

```
ADD   A,9ØH
DAA
ADC   A,4ØH
DAA
```

This routine was lifted from the resident monitor of the Poly-88 computer, and its operation is not at all obvious.

NOTES:

<N> IN THE OPERAND INDICATES ONE BYTE OF IMMEDIATE DATA PORT NUMBER, OR RELATIVE-JUMP OFFSET.

<NN> IN THE OPERAND INDICATES TWO BYTES OF IMMEDIATE DATA OR ABSOLUTE ADDRESS.

* BEFORE THE OP-CODE INDICATES THAT THE INSTRUCTION CAN BE PRECEEDED BY HEX 'DD' OR 'FD' TO INDICATE THAT THE IX OR IY REGISTER, RESPECTIVELY, IS TO BE ADDRESSED INSTEAD OF THE HL REGISTER.

BEFORE THE OP-CODE INDICATES THAT THE FIRST BYTE OF THE INSTRUCTION CAN BE PRECEEDED BY HEX 'DD' OR 'FD' (TO INDICATE INDEXING BY THE IX OR IY REGISTER, RESPECTIVELY, INSTEAD OF THE HL REGISTER), AND FOLLOWED BY ONE BYTE OF OFFSET (IN TWO'S-COMPLEMENT FORM).

OP	(--)	(CB)	(ED)
00	NOP	RLC B	
01	LD BC,<NN>	RLC C	
02	LD (BC),A	RLC D	
03	INC BC	RLC E	
04	INC B	RLC H	
05	DEC B	RLC L	
06	LD B,<N>	#RLC (HL)	
07	RLCA	RLC A	
08	EX AF,AF'	RRC B	
09	*ADD HL,BC	RRC C	
0A	LD A,(BC)	RRC D	
0B	DEC BC	RRC E	
0C	INC C	RRC H	
0D	DEC C	RRC L	
0E	LD C,<N>	#RRC (HL)	
0F	RRCA	RRC A	
10	DJNZ <N>	RL B	
11	LD DE,<NN>	RL C	
12	LD (DE),A	RL D	
13	INC DE	RL E	
14	INC D	RL H	
15	DEC D	RL L	
16	LD D,<N>	#RL (HL)	
17	RLA	RL A	
18	JR <N>	RR B	
19	*ADD HL,DE	RR C	
1A	LD A,(DE)	RR D	
1B	DEC DE	RR E	
1C	INC E	RR H	
1D	DEC E	RR L	
1E	LD E,<N>	#RR (HL)	
1F	RRA	RR A	
20	JR NZ,<N>	SLA B	
21	*LD HL,<NN>	SLA C	
22	*LD (<NN>),HL	SLA D	

OP	(--)	(CB)	(ED)
23	*INC HL	SLA E	
24	INC H	SLA H	
25	DEC H	SLA L	
26	LD H,<N>	#SLA (HL)	
27	DAA	SLA A	
28	JR Z,<N>	SRA B	
29	*ADD HL,HL	SRA C	
2A	*LD HL,(<NN>)	SRA D	
2B	*DEC HL	SRA E	
2C	INC L	SRA H	
2D	DEC L	SRA L	
2E	LD L,<N>	#SRA (HL)	
2F	CPL	SRA A	
30	JR NC,<N>		
31	LD SP,<NN>		
32	LD (<NN>),A		
33	INC SP		
34	*INC (HL)		
35	*DEC (HL)		
36	*LD (HL),<N>		
37	SCF		
38	JR C,<N>	SRL B	
39	*ADD HL,SP	SRL C	
3A	LD A,SP	SRL D	
3B	DEC SP	SRL E	
3C	INC A	SRL H	
3D	DEC A	SRL L	
3E	LD A,<N>	#SRL (HL)	
3F	CCF	SRL A	
40	LD B,B	BIT 0,B	IN B,(C)
41	LD B,C	BIT 0,C	OUT (C),B
42	LD B,D	BIT 0,D	SBC HL,BC
43	LD B,E	BIT 0,E	LD (<NN>),BC
44	LD B,H	BIT 0,H	NEG
45	LD B,L	BIT 0,L	RETN
46	#LD B,(HL)	#BIT 0,(HL)	IM 0
47	LD B,A	BIT 0,A	LD I,A
48	LD C,B	BIT 1,B	IN C,(C)
49	LD C,C	BIT 1,C	OUT (C),C
4A	LD C,D	BIT 1,D	ADC HL,BC
4B	LD C,E	BIT 1,E	LD BC,(<NN>)
4C	LD C,H	BIT 1,H	
4D	LD C,L	BIT 1,L	RETI
4E	#LD C,(HL)	#BIT 1,(HL)	
4F	LD C,A	BIT 1,A	LD R,A
50	LD D,B	BIT 2,B	IN D,(C)
51	LD D,C	BIT 2,C	OUT (C),D
52	LD D,D	BIT 2,D	SBC HL,DE
53	LD D,E	BIT 2,E	LD (<NN>),DE
54	LD D,H	BIT 2,H	
55	LD D,L	BIT 2,L	

OP	(--)	(CB)	(ED)
56	#LD D,(HL)	#BIT 2,(HL)	IM 1
57	LD D,A	BIT 2,A	LD A,I
58	LD E,B	BIT 3,B	IN E,(C)
59	LD E,C	BIT 3,C	OUT (C),E
5A	LD E,D	BIT 3,D	ADC HL,DE
5B	LD E,E	BIT 3,E	LD DE,(<NN>)
5C	LD E,H	BIT 3,H	
5D	LD E,L	BIT 3,L	
5E	#LD E,(HL)	#BIT 3,(HL)	IM 2
5F	LD E,A	BIT 3,A	LD A,R
60	LD H,B	BIT 4,B	IN H,(C)
61	LD H,C	BIT 4,C	OUT (C),H
62	LD H,D	BIT 4,D	SBC HL,HL
63	LD H,E	BIT 4,E	
64	LD H,H	BIT 4,H	
65	LD H,L	BIT 4,L	
66	#LD H,(HL)	#BIT 4,(HL)	
67	LD H,A	BIT 4,A	RRD
68	LD L,B	BIT 5,B	IN L,(C)
69	LD L,C	BIT 5,C	OUT (C),L
6A	LD L,D	BIT 5,D	ADC HL,HL
6B	LD L,E	BIT 5,E	
6C	LD L,H	BIT 5,H	
6D	LD L,L	BIT 5,L	
6E	#LD L,(HL)	#BIT 5,(HL)	
6F	LD L,A	BIT 5,A	RLD
70	#LD (HL),B	BIT 6,B	
71	#LD (HL),C	BIT 6,C	
72	#LD (HL),D	BIT 6,D	SBC HL,SP
73	#LD (HL),E	BIT 6,E	LD (<NN>),SP
74	#LD (HL),H	BIT 6,H	
75	#LD (HL),L	BIT 6,L	
76	HALT	#BIT 6,(HL)	
77	#LD (HL),A	BIT 6,A	
78	LD A,B	BIT 7,B	IN A,(C)
79	LD A,C	BIT 7,C	OUT (C),A
7A	LD A,D	BIT 7,D	ADC HL,SP
7B	LD A,E	BIT 7,E	LD SP,(<NN>)
7C	LD A,H	BIT 7,H	
7D	LD A,L	BIT 7,L	
7E	#LD A,(HL)	#BIT 7,(HL)	
7F	LD A,A	BIT 7,A	
80	ADD A,B	RES 0,B	
81	ADD A,C	RES 0,C	
82	ADD A,D	RES 0,D	
83	ADD A,E	RES 0,E	
84	ADD A,H	RES 0,H	
85	ADD A,L	RES 0,L	
86	#ADD A,(HL)	#RES 0,(HL)	
87	ADD A,A	RES 0,A	

OP	(--)	(CB)	(ED)
88	ADC A,B	RES 1,B	
89	ADC A,C	RES 1,C	
8A	ADC A,D	RES 1,D	
8B	ADC A,E	RES 1,E	
8C	ADC A,H	RES 1,H	
8D	ADC A,L	RES 1,L	
8E	#ADC A,(HL)	#RES 1,(HL)	
8F	ADC A,A	RES 1,A	
90	SUB B	RES 2,B	
91	SUB C	RES 2,C	
92	SUB D	RES 2,D	
93	SUB E	RES 2,E	
94	SUB H	RES 2,H	
95	SUB L	RES 2,L	
96	#SUB (HL)	#RES 2,(HL)	
97	SUB A	RES 2,A	
98	SBC B	RES 3,B	
99	SBC C	RES 3,C	
9A	SBC D	RES 3,D	
9B	SBC E	RES 3,E	
9C	SBC H	RES 3,H	
9D	SBC L	RES 3,L	
9E	#SBC (HL)	#RES 3,(HL)	
9F	SBC A	RES 3,A	
A0	AND B	RES 4,B	LDI
A1	AND C	RES 4,C	CPI
A2	AND D	RES 4,D	INI
A3	AND E	RES 4,E	OUTI
A4	AND H	RES 4,H	
A5	AND L	RES 4,L	
A6	#AND (HL)	#RES 4,(HL)	
A7	AND A	RES 4,A	
A8	XOR B	RES 5,B	LDD
A9	XOR C	RES 5,C	CPD
AA	XOR D	RES 5,D	IND
AB	XOR E	RES 5,E	OUTD
AC	XOR H	RES 5,H	
AD	XOR L	RES 5,L	
AE	#XOR (HL)	#RES 5,(HL)	
AF	XOR A	RES 5,A	
B0	OR B	RES 6,B	LDIR
B1	OR C	RES 6,C	CPIR
B2	OR D	RES 6,D	INIR
B3	OR E	RES 6,E	OTIR
B4	OR H	RES 6,H	
B5	OR L	RES 6,L	
B6	#OR (HL)	#RES 6,(HL)	
B7	OR A	RES 6,A	
B8	CP B	RES 7,B	LDDR
B9	CP C	RES 7,C	CPDR
BA	CP D	RES 7,D	INDR

OP	(--)	(CB)	(ED)
BB	CP E	RES 7,E	OTDR
BC	CP H	RES 7,H	
BD	CP L	RES 7,L	
BE	#CP (HL)	#RES 7,(HL)	
BF	CP A	RES 7,A	
C0	RET NZ	SET 0,B	
C1	POP BC	SET 0,C	
C2	JP NZ,<NN>	SET 0,D	
C3	JP <NN>	SET 0,E	
C4	CALL NZ,<NN>	SET 0,H	
C5	PUSH BC	SET 0,L	
C6	ADD A,<N>	#SET 0,(HL)	
C7	RST 0	SET 0,A	
C8	RET Z	SET 1,B	
C9	RET	SET 1,C	
CA	JP Z,<NN>	SET 1,D	
CB	== COLUMN 2 ==	SET 1,E	
CC	CALL Z,<NN>	SET 1,H	
CD	CALL <NN>	SET 1,L	
CE	ADC <N>	#SET 1,(HL)	
CF	RST 8	SET 1,A	
D0	RET NC	SET 2,B	
D1	POP DE	SET 2,C	
D2	JP NC,<NN>	SET 2,D	
D3	OUT (<N>),A	SET 2,E	
D4	CALL NC,<NN>	SET 2,H	
D5	PUSH DE	SET 2,L	
D6	SUB <N>	#SET 2,(HL)	
D7	RST 16	SET 2,A	
D8	RET C	SET 3,B	
D9	EXX	SET 3,C	
DA	JP C,<NN>	SET 3,D	
DB	IN A,(<N>)	SET 3,E	
DC	CALL C,<NN>	SET 3,H	
DD	== INDEX IX ==	SET 3,L	
DE	SBC <N>	#SET 3,(HL)	
DF	RST 24	SET 3,A	
E0	RET PO	SET 4,B	
E1	*POP HL	SET 4,C	
E2	JP PO,<NN>	SET 4,D	
E3	*EX (SP),HL	SET 4,E	
E4	CALL PO,<NN>	SET 4,H	
E5	*PUSH HL	SET 4,L	
E6	AND <N>	#SET 4,(HL)	
E7	RST 32	SET 4,A	
E8	RET PE	SET 5,B	
E9	*JP (HL)	SET 5,C	
EA	JP PE,<NN>	SET 5,D	
EB	EX DE,HL	SET 5,E	
EC	CALL PE,<NN>	SET 5,H	
ED	== COLUMN 3 ==	SET 5,L	

```
OP      (--)            (CB)            (ED)

EE      XOR  (N)        #SET  5,(HL)
EF      RST  40          SET  5,A

F0      RET  P           SET  6,B
F1      POP  AF          SET  6,C
F2      JP   P,(NN)      SET  6,D
F3      DI               SET  6,E
F4      CALL P,(NN)      SET  6,H
F5      PUSH AF          SET  6,L
F6      OR   (N)        #SET  6,(HL)
F7      RST  48          SET  6,A

F8      RET  M           SET  7,B
F9     *LD   SP,HL       SET  7,C
FA      JP   M,(NN)      SET  7,D
FB      DI               SET  7,E
FC      CALL M,(NN)      SET  7,H
FD      == INDEX IY ==   SET  7,L
FE      CP   (N)        #SET  7,(HL)
FF      RST  56          SET  7,A
```

NOT SO BASIC.

TIPS FOR TRICKS WITH LEVEL II BASIC
BY CHRIS GUNDLACH, 618 HAL GREER BOULEVARD, HUNTINGTON WVA

BREAK KEY OFF

This command turns off the !BREAK! key:

```
POKE 16396, 23
```

You can issue it as a direct command and any program that you then RUN will be unstoppable, except with the rear reset. You can also use it inside programs to transform the operation of the !BREAK! key to some other function. Example:

```
10  POKE 16396,23 : XX$=INKEY$ : IF XX$="" THEN 10 ELSE AX=ASC(XX$) :
    IF AX=2 THEN POKE 16396,25 : RETURN ELSE PRINT AX;: GOTO 10
```

This subroutine is handy for program development. When you GOSUB 10, the subprogram prints the ASCII code of any key hit on the keyboard. Hitting the !BREAK! key, however, executes a RETURN, and also restores the operation of the !BREAK! key itself to normal.

The subroutine works because when the !BREAK! key is turned off, its ASCII code becomes 2. It is usually 1.

You can use the same technique for other features. If you replace the underlined part of the subprogram above with:

```
IF AX=2THEN RUN ELSE PRINT AX;: GOTO 10
```

the !BREAK! key automatically restarts the resident program. Expand this feature into your longer programs and you can effectively make your programs have positive, complete control over the keyboard.

Expanding a little on the above, replace the underlined part of the subroutine with:

```
IFAX=96THEN RUN ELSE IF AX=2 THEN POKE 16396,25:RETURN
ELSE PRINT AX;: GOTO 10
```

Pressing the shift-@ key TWICE restarts your whole resident program, and pressing !BREAK! returns you from the subroutine. The "automatic secret restart code" — the two shift-@'s — is part of your TRS' normal logic. The first shift-@ stops the program, awaiting the touch of any key. Unless the "restart" key is another shift-@, the routine simply reads and prints the character. Expanding this logic to all input logic in your program lets you offer flexibility, yet tight control, in your user-oriented work.

SPECIAL-KEY NOTATION

One convention that makes for attractive programs is the display of keyboard instructions. This program initialization line sets up some string variables for display:

```
10 CLEAR 500: DEFSTR W : WL=CHR$(133) : WR=CHR$(138):
   WE=WL+"ENTER"+WR : WB=WL+"BREAK"+WR : WS = WL+"SHIFT"+WR :
   WC=WL+"CLEAR"+WR : WA=WS+" "+CHR$(125):
   WX=WE+" TO CONTINUE" : WW=CHR$(31) : WH=CHR$(28)
```

The following subroutine prints standard INPUT keyboard instructions, and returns the cursor to its last position:

```
60000 Q1=PEEK(16416):Q2=PEEK(16417):Q3=PEEK(16418):PRINT CHR$(15);
60010 PRINT@704,STRING$(64,143);WW;: PRINT@731," KEYBOARD↓
      ";TAB(8);CHR$(125);" TO BACKSPACE & CORRECT↓
      ";WA;" TO ERASE ENTIRE LINE ↓
      ";WB;" TO RESTART PROGRAM ↓
      ";WE;" AFTER TYPING YOUR ENTRY";
60020 POKE 16416, Q1 : POKE 16417, Q2 : POKE 16418, Q3 : RETURN
60030 Q1=PEEK(16416):Q2=PEEK(16417):Q3=PEEK(16418):PRINT@704,
      STRING$(64,143);WW;WX;
60040 WI=INKEY$ : IF WI="" THEN60040ELSE IF ASC(WI)<>13 THEN
      60040 ELSE PRINT CHR$(29);WW;: GOTO 60020
```

Calling GOSUB 60000 from a program prints the INPUT instructions at screen bottom. Calling GOSUB 60030 goes to a subroutine that waits for an !ENTER! to continue. Both routines will return the cursor to where it was before the call.

The down-arrow at the end of lines in 60010 mean just that — hit the down-arrow key. It becomes part of the text string.

Tapes with these subroutines are available from me, $3.00 each. Address is under "NOT SO BASIC" at the top.

THE BITPICKER'S TOOLBOX

By Steve MacGregor, 3701 W Wethersfield, Phoenix, Arizona 85029

The INT function of BASIC can be thought of as a "round-down" function, so that INT(X) is the greatest integer less than or equal to X. To do a "round-off", use INT(X+.5), and get the closest integer to X. If you need to "round-up", -INT(-I) will give the least integer greater than or equal to X.