A BASIC listing of a program which shows a spinning bar is given on page 20 of the CHEMUTIL-2 documentation. It works by printing a series of images of the bar. It uses the same principle that was used to simulate the flame but with a significant difference in application. The flame printed characters directly over one another, and it did not matter in which order they were chosen. The bar uses two characters printed side by side and the order is important.

To show a half turn of the bar requires the definition of ten new characters. These are then printed in pairs rapidly. The pairs are shown in Figure 1.

A full turn of the bar requires rapid printing in place of each pair in the order A B C D E D C B A. A continuing loop like that gives the illusion of a rotating stirrer bar. The CHEMUTIL-2 program is not quite as sophisticated as this in that it actually loops through A B C D E A. You can see that if you add a delay to that program so that each image stays on the screen for a longer time(2). The result of that economy of code is that the bar has a small hitch in its rotation which actually helps the illusion in that rotating magnetic bars often have a small kick in an otherwise smooth movement.

We need to rewrite the routine so that it can be successfully integrated with the flame routine and the yet-to-be developed falling drop code. The same logistical problems must be solved as were encountered with the flame. An appropriate program loop would be:-

Locate bar position : Print bar pair(A) : Delay : Locate bar position : Print bar pair(B) : Delay : etc.

However, we must arrange to have the full nine pair sequence for a full turn rather than the truncated five pair sequence of the CHEMUTIL-2 example. The reason is that we cannot be sure that the bar will rotate fast enough so that the hitch is acceptable when the flame and falling drop animation are integrated into it. We can rotate the bar slower and maintain the illusion when the nine pair sequence is in place. One way to code this would be to have two loops. The first loop prints images A to E and the second from D to B. However execution time can be reduced by defining six more characters which are duplicates of pairs D, C and B respectively. Then successive full sweeps through the eight pairs of characters will constitute a series of full turns of the stirrer bar. The additional six characters lowing the ten characters already available in CHEMUTIL-2 starting at $7390.

Listing 1 will add the duplicated characters to CHEMUTIL-2 and Listing 2 is the BASIC code for the rotating bar. Notice the similarity of Listing 2 to the program used to show the flickering flame. The ability to stop the animation by pressing the S-key has been included.

Listing 1

```
10 FOR I = 1 to 48
20 Read X : POKE (29599 + I),X
30 NEXT I
40 DATA 0, 0, 0, 120, 124, 126, 124, 120
50 DATA 0, 0, 0, 15, 31, 63, 31, 15
60 DATA 0, 0, 0, 112, 120, 126, 120, 112
70 DATA 0, 0, 0, 7, 15, 31, 15, 7,
80 DATA O, O, O, 96, 112, 120, 112, 96,
90 DATA O, O, O, 3, 7, 15, 7, 3,
```

Listing 2

```
10 CALL 25042 : PRINT "&" : REM
ENABLE CHEMUTIL-2 SET 2
20 FOR I = 106 TO 121 STEP 2
30 VTAB 10 : HTAB 10
40 PRINT CHR$ (I) + CHR$ (I + 1)
50 X = PEEK (-16368), O : IF X = 211
THEN I = 122 : GOTO 80
60 NEXT I
70 GOTO 20
80 VTAB 20 : END
```

The source of the code is given for the machine language version. Enter it into your Apple with an assembler as via the monitor. Then, after BLOADing CHEMUTIL-2 and running Listing 1,

```
RUN  10 CALL 25042
       20 CALL 31760
       30 IF PEEK (31746) = 16 THEN
END
          40 GOTO 20
```

Notice that the resulting stirrer is very jerky. That is because the delay, at Line 71 of the source code, is too short so that several bar images are drawn while the monitor screenis being refreshed. Try poking other values at 31861 ($7C75) and seeing the effect on the bar.

The machine language version was written so that the program terminates when the S-key is pressed or after 256 bar images have been shown. The latter is controlled by the TOTSPN counter. Its effect can be negated by removing Line 54 of the source code or by poking 31832, 234 : 31833, 234 and 31834, 234. It is useful when your overall program requires that the animation cease when the S-key is pressed or after a set time by a loop like FOR I = 1 TO 200 : CALL 31760 : NEXT I.

The next part of this series will discuss how the two separate routines for stirrer and flame can be integrated so that they appear to be executing simultaneously.Only machine language source code will be given since BASIC coding would execute too slowly for a realistic illusion.

An Animation of Distillation
Part III
Combined Heating and Stirring :

The two earlier parts of this series showed how to write routines for displaying a flickering flame and a spinning stirrer bar on the Apple graphics screen using CHEMUTIL-2 (1) as the character generator. In this article, I shall describe how the routine can be combined so that the flame and stirrer appear to be activated simultaneously. Since a BASIC program would execute too slowly for effective animation only the machine language source code is given.

The overall algorithm is simple in principle.Show flame (1), show stirrer (1), delay, show flame (2), show stirrer (2), delay, etc. As we have previously indicated, the delay is present because it will allow us to control the overall speed of the illusion and also leaves room for the introduction of the falling drop routine later.

This algorithm would work but is too limited because the bar would rotate at the same speed as the flame flickered. the simple alternation of flame and stirrer means that whatever affects the speed of one affects the speed of the other. A rapidly flickering flame with a slow stirrer would be impossible. Fortunately, we have already allowed for the latter possibility when we wrote the separate routines because they have separate delay routines incorporated into each. More importantly, we included the counter TOTCNT and TOTSPN which allow an exit from either routine when the counters are zeroed by the executing programs. this is the true significance of Line 48 and. Line 54 in the FLAME and SPINNER

```
SOURCE FILE: SPINNER
------ NEXT OBJECT FILE NAME IS SPINNER.OBJ0
7C00:            1          ORG  $7C00
0000:            2          OBJ  $7C00
7C00:            3 *
7C00:            4 ********************
7C00:            5 * SPINNER
7C00:            6 ********************
7C00:            7 *
0025:            8 VTAB     EQU  $25
0024:            9 HTAB     EQU  $24
6200:           10 SET1     EQU  $6200      ;REGULAR CHARACTER SET
6206:           11 SET2     EQU  $6206      ;ALTERNATE CHARACTER SET
7C02:           12 FLAG     EQU  $7C02      ;STOP FLAG
7C06:           13 SPINLOC  EQU  $7C06      ;VTAB AND HTAB LOCATION IN
7C00:           14 *           THESE TWO BYTES
7C08:           15 TOTSPN   EQU  $7C08      ;A COUNTER -WHEN TOTSPN IS ZERO
7C00:           16 *                           THE ROUTINE EXITS
7C0A:           17 SPNCNT   EQU  $7C0A      ;BAR IMAGE TO BE DISPLAYED
7C00:           18 *
7C00:           19 *
7C00:           20          DS   16
7C10:           21 *
7C10:A9 0B      22          LDA  #11
7C12:8D 06 7C   23          STA  SPINLOC    ;VTAB OF BAR LOCATION
7C15:A9 09      24          LDA  #9
7C17:8D 07 7C   25          STA  SPINLOC+1  ;HTAB OF BAR LOCATION
7C1A:A9 00      26          LDA  #$00       ;ZERO THE STOP FLAG
7C1C:8D 02 7C   27          STA  FLAG
7C1F:8D 08 7C   28          STA  TOTSPN
7C22:A9 EA      29          LDA  #$EA
7C24:8D 0A 7C   30          STA  SPNCNT     ;LOAD FIRST BAR IMAGE
7C27:A5 25      31          LDA  VTAB       ;SAVE LOCATION FROM WHICH
7C29:48         32          PHA             ;THE ROUTINE WAS CALLED
7C2A:A5 24      33          LDA  HTAB
7C2C:48         34          PHA
7C2D:           35 *
7C2D:           36 *
7C2D:AD 06 7C   37 SPINST   LDA  SPINLOC    ;SET TABS TO BAR LOCATION
7C30:85 25      38          STA  VTAB
7C32:AD 07 7C   39          LDA  SPINLOC+1
7C35:85 24      40          STA  HTAB
7C37:           41 *
7C37:20 06 62   42 SPINIT   JSR  SET2
7C3A:AC 0A 7C   43          LDA  SPNCNT     ;LOAD CURRENT FIRST BAR IMAGE
7C3D:C9 FA      44          CMP  #$FA       ;THIS IS THE LAST BAR IMAGE
7C3F:D0 05      45          BNE  RO
7C41:A9 EA      46          LDA  #$EA       ;RESTART BAR IMAGES
7C43:8D 0A 7C   47          STA  SPNCNT
7C46:20 ED FD   48 RO       JSR  $FDED      ;PRINT LEFT BAR IMAGE
7C49:EE 0A 7C   49          INC  SPNCNT
7C4C:AD 0A 7C   50          LDA  SPNCNT
7C4F:20 ED FD   51          JSR  $FDED      ;PRINT RIGHT BAR IMAGE
7C52:20 74 7C   52          JSR  DELAY
7C55:EE 0A 7C   53          INC  SPNCNT     ;GOTO NEXT BAR PICTURE
7C58:EE 08 7C   54          INC  TOTSPN
7C5B:D0 0A      55          BNE  RCO
7C5D:           56 *
7C5D:68         57 EXIT     PLA             ;RETURN TO CALLING LOCATION
7C5E:85 24      58          STA  HTAB
7C60:68         59          PLA
7C61:85 25      60          STA  VTAB
7C63:20 00 62   61          JSR  SET1       ;REVERT TO NORMAL CHARACTER SET
7C66:60         62          RTS
7C67:           63 *
7C67:20 7A 7C   64 RCO      JSR  KYBD       ;CHECK KEYBOARD
7C6A:AD 02 7C   65          LDA  FLAG
7C6D:C9 10      66          CMP  #$10       ;IS FLAG SET
7C6F:F0 EC      67          BEQ  EXIT
7C71:4C 2D 7C   68          JMP  SPINST
7C74:           69 *
7C74:           70 *
7C74:A9 01      71 DELAY    LDA  #$01       ;LEAVE ROOM FOR MORE
7C76:20 A8 FC   72          JSR  $FCA8
7C79:60         73          RTS
7C7A:           74 *
7C7A:AD 00 C0   75 KYBD     LDA  $C000      ;CHECK FOR KEY PRESS
7C7D:2C 10 C0   76          BIT  $C010      ;CLEAR KEYBOARD STROBE
7C80:C9 D3      77          CMP  #$D3       ;LOOK FOR S KEY
7C82:F0 01      78          BEQ  STOP
7C84:60         79          RTS
7C85:           80 *
7C85:A9 10      81 STOP     LDA  #$10       ;SET FLAG IF S WAS PRESSED
7C87:8D 02 7C   82          STA  FLAG
7C8A:60         83          RTS
7C8B:           84 *

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

source codes.

The improved algorithm becomes:
(A) Show a series of flame images with a delay between them until the TOTCNT counter becomes zero then ?
(B) Show a series of stirrer images with a delay between them until the TOTSPN counter becomes zero then
(C) Loop back to (A).

The flame delay and the stirrer delay need not be the same length and one (or both) of them can be used to check the keyboard periodically. There are now four locations which can be used to adjust the speed of the overall animation or the flame and stirrer separately. They correspond to the lengths of the two delay loops and the initial values chosen for TOTCNT and TOTSPN.

The source code for the combined routines shows how simple it ?was to combine them. Since the separate routines wee debugged separately, we can be certain that if the combination does not function the interface is at fault. In my experience the most likely errors at this point are made by not initializing variables and counters as the switch is made from one working routine to the other.

To execute the code it must be entered at $7C10 and CHEMUTIL-2 BLOADed. Listing 1 from part II must have been run to supply the extra stirrer images. Then the following brief BASIC program is run:

```
10 CALL 25042 : PRINT "&"
20 CALL 31760 C
```

You will now see a flickering flame with a rotating bar over it. The bar has been placed over the flame by Lines 33-36 of the source code. You can easily alter its relative position if you desire but this program puts it over the flame because it is the correct relative position for a distillation which is our ultimate goal.

In this code as written there are 32 flame images between every stirrer image. i.e. There are 256 flame images for every full turn of the stirrer bar. It is instructive to see the effects of introducing other values from 1-255 into TOTSPN, TOTCNT and the two delay intervals. Try poking values into locations 31850 (TOTSPN), 131855 (TOTCNT), 31891 (DELAY) and 31963 (DELSPN).

An advantage to writing the separate routines and then interfacing them is that they can still be used separately when needed. The equivalent combination in BASIC is a loop like GOSUB IFLAME : GOSUB STIRRER. A loop like this will show only the flame if the first line of the stirrer routine is RETURN. So by poking the first byte of the SPINST routine in the machine language code with the equivalent of RETURN only the flame image will appear. Try Poke 31913, 96 and see the result.

The disadvantage of interfacing separate routines in this way is that the combination will execute slower than a specially designed combination program. This is seldom of real consequence for machine language because

the length of the delay loops can be adjusted to compensate, but in BASIC programs it can be devastating. Remember that when you do animation in machine language you usually must deliberately slow it down but in BASIC the animation will be too slow from the first.

The fourth part of this series will discuss the falling drop and accumulating liquid animation.

(1) Bendall, V. "CHEMUTIL-2, A Chemistry Programming Utility"; Project SERAPHIM, NSF Science Education; Department of Chemistry, Eastern Michigan University, Ypsilanti, MI 48197, 1985.

(2) For example, add the following line to the CHEMUTIL-2 listing. 75 FOR K = 1 to 100 : NEXT K and RUN the program again.

```
SOURCE FILE  SPINBURN
------ NEXT OBJECT FILE NAME IS SPINBURN.OBJ0
7C00:             1         ORG  $7C00
0000.             2         OBJ  $7C00
7C00:             3 *
7C00:             4 ********************
7C00:             5 * SPINBURN
7C00:             6 ********************
7C00:             7 *
0025:             8 VTAB    EQU  $25
0024:             9 HTAB    EQU  $24
6200:            10 SET1    EQU  $6200
6206:            11 SET2    EQU  $6206
7C00:            12 FLIKLOC EQU  $7C00    ;VTAB, HTAB OF BURNER FLAME
7C02:            13 FLAG    EQU  $7C02    ;SETS TO $10 WHEN S IS PRESSED
7C03:            14 TOTCNT  EQU  $7C03    ;NUMBER OF TIMES THROUGH FLAME CYCLE
7C05:            15 BURNCT  EQU  $7C05    ;FLAME SYMBOL BEING DRAWN $E1-$E9
7C06:            16 SPINLOC EQU  $7C06    ;VTAB, HTAB OF STIRRER BAR
7C08:            17 TOTSPN  EQU  $7C08    ;NUMBER OF TIMES THROUGH SPIN CYCLE
7C0A:            18 SPNCNT  EQU  $7C0A    ;SPIN SYMBOL BEING DRAWN  $EA-$F3
7C00:            19 *
7C00:            20         DS   16
7C10:            21 *
7C10:A9 0E       22 START   LDA  #11      ;SET VTAB OF FLAME
7C12:8D 00 7C    23         STA  FLIKLOC
7C15:A9 09       24         LDA  #9       ;SET HTAB OF FLAME
7C17:8D 01 7C    25         STA  FLIKLOC+1
7C1A:A9 00       26         LDA  #$00     ;ZERO KYBD FLAG
7C1C:8D 02 7C    27         STA  FLAG
7C1F:A9 FF       28         LDA  #$FF
7C21:8D 03 7C    29         STA  TOTCNT
7C24:8D 08 7C    30         STA  TOTSPN
7C27:A9 EA       31         LDA  #$EA
7C29:8D 0A 7C    32         STA  SPNCNT
7C2C:AD 00 7C    33         LDA  FLIKLOC   ;STIRRER BAR TWO SPACES ABOVE FLAME
7C2F:38          34         SEC
7C30:E9 02       35         SBC  #2
7C32:8D 06 7C    36         STA  SPINLOC
7C35:AD 01 7C    37         LDA  FLIKLOC+1
7C38:8D 07 7C    38         STA  SPINLOC+1
7C3B:A9 E2       39         LDA  #$E2
7C3D:8D 05 7C    40         STA  BURNCT
7C40:A5 25       41         LDA  VTAB     ;SAVE CURRENT CURSOR LOCATION
7C42:48          42         PHA
7C43:A5 24       43         LDA  HTAB
7C45:48          44         PHA
7C46:            45 *
7C46:20 E8 7C    46 ROUND   JSR  LOCFL
7C49:            47 *
7C49:20 36 62    48 FLIKER  JSR  SET2
7C4C:AD 05 7C    49         LDA  BURNCT
7C4F:C9 EA       50         CMP  #$EA
7C51:D0 05       51         BNE  RO

7C53:A9 E1       52         LDA  #$E1
7C55:8D 05 7C    53         STA  BURNCT
7C58:20 ED FD    54 RO      JSR  $FDED
7C5B:20 92 7C    55         JSR  DELAY
7C5E:EE 05 7C    56         INC  BURNCT
7C61:EE 03 7C    57         INC  TOTCNT
7C64:D0 0D       58         BNE  ROO
7C66:20 A9 7C    59         JSR  SPINST
7C69:A9 FF       60         LDA  #$FF     ;SPINS BETWEEN FLICKER = 256-TOTSPN
7C6B:8D 08 7C    61         STA  TOTSPN
```

```
7C6E:A9 FF       62         LDA  #$FF     ;FLICKERS BETWEEN SPIN = 256-TOTSPN
7C70:8D 03 7C    63         STA  TOTCNT
7C73:20 98 7C    64 ROO     JSR  DELAY
7C76:AD 02 7C    65         LDA  FLAG
7C79:C9 10       66         CMP  #$10
7C7B:F0 03       67         BEQ  EXIT
7C7D:4C 46 7C    68         JMP  ROUND
7C80:            69 *
7C80:20 E8 7C    70 EXIT    JSR  LOCFL
7C83:A9 A0       71         LDA  #$A0     ;PUT OUT FLAME OF BURNER
7C85:20 ED FD    72         JSR  $FDED
7C88:68          73         PLA           ;GET ORIGINAL CURSOR LOCATION
7C89:85 24       74         STA  HTAB
7C8B:68          75         PLA
7C8C:85 25       76         STA  VTAB
7C8E:20 00 62    77         JSR  SET1     ;RETURN TO UPPER CASE
7C91:60          78         RTS
7C92:            79 *
7C92:A9 01       80 DELAY   LDA  #$01     ;DELAY BETWEEN FLAME IMAGES
7C94:20 A8 FC    81         JSR  $FCA8
7C97:60          82         RTS
7C98:            83 *
7C98:AD 00 C0    84 KYBD    LDA  $C000
7C9B:2C 10 C0    85         BIT  $C010
7C9E:C9 D3       86         CMP  #$D3     ;CHECK FOR S
7CA0:F0 01       87         BEQ  STOP
7CA2:60          88         RTS
7CA3:A9 10       89 STOP    LDA  #$10
7CA5:8D 02 7C    90         STA  FLAG
7CA8:60          91         RTS
7CA9:            92 *
7CA9:AD 06 7C    93 SPINST  LDA  SPINLOC  ;SET TABS TO STIRRER LOCATION
7CAC:85 25       94         STA  VTAB
7CAE:AD 07 7C    95         LDA  SPINLOC+1
7CB1:85 24       96         STA  HTAB
7CB3:            97 *
7CB3:20 06 62    98 SPINIT  JSR  SET2
7CB6:AD 0A 7C    99         LDA  SPNCNT
7CB9:C9 FA      100         CMP  #$FA
7CBB:D0 05      101         BNE  RO1
7CBD:A9 EA      102         LDA  #$EA
7CBF:8D 0A 7C   103         STA  SPNCNT
7CC2:20 ED FD   104 RO1     JSR  $FDED

7CC5:EE 0A 7C   105         INC  SPNCNT
7CC8:AD 0A 7C   106         LDA  SPNCNT
7CCB:20 ED FD   107         JSR  $FDED
7CCE:20 DA 7C   108         JSR  DELSPN
7CD1:EE 0A 7C   109         INC  SPNCNT
7CD4:EE 08 7C   110         INC  TOTSPN
7CD7:D0 D0      111         BNE  SPINST
7CD9:60         112         RTS
7CDA:           113 *
7CDA:A9 01      114 DELSPN  LDA  #$01     ;DELAY BETWEEN STIRRER IMAGES
7CDC:20 A8 FC   115         JSR  $FCA8
7CDF:60         116         RTS
7CE0:           117 *
7CE0:AD 00 7C   118 LOCFL   LDA  FLIKLOC  ;SET TABS TO FLAME LOCATION
7CE3:85 25      119         STA  VTAB
7CE5:AD 01 7C   120         LDA  FLIKLOC+1
7CE8:85 24      121         STA  HTAB
7CEA:60         122         RTS
7CEB:           123 *

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```