NASA CR 185701 1 of 1

FINAL REPORT

REGENERABLE BIOCIDE DELIVERY UNIT

VOLUME 2

SEPTEMBER 1992

Prepared by

James E. Atwater
Richard R. Wheeler, Jr.

Contract No. NAS9-18361

Lyndon B. Johnson Space Center
National Aeronautics and Space Administration
Houston, TX 77058

UMPQUA RESEARCH COMPANY
AEROSPACE DIVISION
P.O. BOX 791 - 125 VOLUNTEER WAY
MYRTLE CREEK, OR 97457
TELE: (503) 863-7770
FAX: (503) 863-7775

URC 80356

FINAL REPORT

REGENERABLE BIOCIDE DELIVERY UNIT

VOLUME 2

SEPTEMBER 1992

Prepared by

James E. Atwater
Richard R. Wheeler, Jr.

Contract No. NAS9-18361

Lyndon B. Johnson Space Center
National Aeronautics and Space Administration
Houston, TX 77058

UMPQUA RESEARCH COMPANY
AEROSPACE DIVISION
P.O. BOX 791 - 125 VOLUNTEER WAY
MYRTLE CREEK, OR 97457
TELE: (503) 863-7770
FAX: (503) 863-7775

URC 80356

APPENDIX I

PROJECT SOFTWARE

CONTENTS

```
' PROGRAM : RMCV.BAS
' Author: James E. Atwater - Umpqua Research Company.

DECLARE SUB SETINTEGRATIONTIME (SECONDS!)
DECLARE SUB SETIODINEMODE ()
DECLARE SUB TOGGLE (REL!)
DECLARE SUB MONITORIODINEONSCREEN ()
DECLARE SUB SETBLANK ()

DECLARE SUB PASSIVALUES (IODIDE, TRIIODIDE, IODINE, BASELINE)
DECLARE SUB LAMPON ()
DECLARE SUB GR (J!, I!, AGRAPH!, IODINE, IODIDE, TRIIODIDE, XX!,
RELAY())
DECLARE SUB GETTEMP (TEMP)
DECLARE SUB READCYCLE (IFAC(), I2FAC(), IFUDGE(), Y$, YY$(),
STREAMS(), AGRAPH, LAST$)

DECLARE SUB RECALIBRATE (IFAC(), I2FAC(), IFUDGE())
DECLARE SUB CALIBRATE (IFAC(), I2FAC(), IFUDGE())
DECLARE SUB INITIALIZE (STREAMS())
DECLARE SUB NEWTIME (SETTIME)


'$INCLUDE: 'C:\QB45\UV_SUBS\52SETUP.BAS'

DIM STREAMS(8)
DIM IFAC(4), I2FAC(4), IFUDGE(4)
DIM YY$(2)
DIM RELAYS%(8)

   RELSTATE% = 0
   RELAYS%(1) = 1
   FOR I = 1 TO 7
        RELAYS%(I + 1) = RELAYS%(I) * 2
   NEXT I

   CWORD% = &H8B
   PORTA% = &H300
   CADDR% = &H303
   OUT CADDR%, CWORD%
   OUT PORTA%, RELSTATE%

  FOR I = 1 TO 8
   TOGGLE (I)
```

```
NEXT I

 SCREEN 12
COLOR 15
AGRAPH = 1
LAMPON
SETINTEGRATIONTIME (2)
SETIODINEMODE

FOR K = 1 TO 4
  STREAMS(K) = 0
NEXT K

CALL INITIALIZE(STREAMS())
CLS

LOCATE 10, 10
PRINT " 1. USE CANNED RESPONSE FACTORS "
LOCATE 12, 10
PRINT " 2. INPUT RESPONSE FACTORS MANUALLY "
LOCATE 14, 10
PRINT " 3. DERIVE RESPONSE FACTORS USING STANDARDS "
LOCATE 16, 10
PRINT " 4. EXIT THE PROGRAM "

LOCATE 20, 10
INPUT " How do you want to calibrate the instrument "; MENU

 CLS
 BEEP
 LOCATE 16, 10
 INPUT " Do you want data to be logged into a file (Y/N) "; YY$(1)

 CLS
 BEEP
 LOCATE 16, 10
 INPUT " Do you want data to be output on PRINTER (Y/N) "; YY$(2)

SETBLANK
Y$ = "y"
IF YY$(1) = "y" THEN YY$(1) = "Y"
IF YY$(2) = "y" THEN YY$(2) = "Y"

IODIDE = 1!
TRIIODIDE = 2!
```

```
 IODINE = 3!
 BASELINE = 4!
 I2228 = 0!

 FOR I = 1 TO 4
  IFAC(I) = .8
  I2FAC(I) = 41.5
  IFUDGE(I) = .148
 NEXT I

 IFUDGE(1) = .18
 I2FAC(2) = I2FAC(2) * .65
 I2FAC(3) = I2FAC(3) * .55
 I2FAC(4) = I2FAC(4) * .55

 CLS

 LAST$ = TIME$

SELECT CASE MENU
 CASE 1
    CALL READCYCLE(IFAC(), I2FAC(), IFUDGE(), Y$, YY$(), STREAMS(),
AGRAPH, LAST$)
 CASE 2
    CLS

   FOR I = 1 TO 4
        SELECT CASE STREAMS(I)
        CASE 1
        PRINT I
                INPUT " What is the value of I2 calibration factor ", I2FAC(I)
                PRINT I
        INPUT " What is the value of I- calibration factor ", IFAC(I)
        PRINT I
        INPUT " What is the value of the fudge factor (228nm I2) ", IFUDGE(I)
      CASE 0
      END SELECT
   NEXT I

   CALL READCYCLE(IFAC(), I2FAC(), IFUDGE(), Y$, YY$(), STREAMS(),
AGRAPH, LAST$)

 CASE 3
    CLS
    CALL CALIBRATE(IFAC(), I2FAC(), IFUDGE())
    CLS
    CALL READCYCLE(IFAC(), I2FAC(), IFUDGE(), Y$, YY$(), STREAMS(),
AGRAPH, LAST$)

CASE 4
END SELECT

END   ' This is the end of the program



'       Subroutines written for RMCV.BAS follow:


SUB CALIBRATE (IFAC(), I2FAC(), IFUDGE())
   CLS
   Y$ = "y"
  WHILE (Y$ = "y") OR (Y$ = "Y")
   BEEP
   INPUT " Which Stream (1-4) is being calibrated "; AGRAPH
   LOCATE 22, 10
   INPUT " What is the value of the IODINE (I2) Standard (mg/L) ", I2STD
   LOCATE 2, 10
   INPUT " PRESS <ENTER> WHEN SAMPLE IS LOADED "; Y

   CALL PASSIVALUES(IODIDE, TRIIODIDE, IODINE, BASELINE)
   CALL MONITORIODINEONSCREEN
   I2FAC(AGRAPH) = I2STD / (IODINE - BASELINE)
   IFUDGE(AGRAPH) = (IODIDE - BASELINE) / I2STD

   BEEP
   LOCATE 23, 10
   BEEP
   INPUT " What is the value of the IodiDe (I-) Standard (mg/L) ", ISTD
   CLS

   LOCATE 23, 10
   INPUT " PRESS <ENTER> WHEN SAMPLE IS LOADED "; Y
   CALL PASSIVALUES(IODIDE, TRIIODIDE, IODINE, BASELINE)
   CALL MONITORIODINEONSCREEN
   IFAC(AGRAPH) = ISTD / (IODIDE - BASELINE)
   INPUT " CARRIAGE RETURN TO CONTINUE ", Y

   CLS
   LOCATE 10, 10
  PRINT " THE VALUE OF THE IODINE FACTOR IS ", I2FAC(AGRAPH)
  LOCATE 11, 10
  PRINT " THE VALUE OF THE IODIDE FACTOR IS ", IFAC(AGRAPH)
  LOCATE 12, 10
  PRINT " THE VALUE OF THE I2 @ 228 nm (FUDGE) FACTOR IS ",
IFUDGE(AGRAPH)
  LOCATE 23, 10
  INPUT " Do you want to do another one (Y/N) "; Y$
 WEND

END SUB     ' End of Subroutine CALIBRATE


SUB GETTEMP (TEMP)
 '1200 baud, odd parity 7-bit data, 2 stop bits, 32-byte input buffer

  OPEN "com2:1200,O,7,2" FOR RANDOM AS #1 LEN = 32
  WHILE LOC(1) < 14                'wait for 14 characters to pass
  WEND
  MODEMINPUT$ = INPUT$(LOC(1), #1)  'grab all characters in buffer
  I = 1                          'search for the +
  WHILE MID$(MODEMINPUT$, I, 1) <> "+"
    I = I + 1
  WEND

  TEMP = VAL(MID$(MODEMINPUT$, I + 2, 3))
  CLOSE #1
END SUB     ' End of Subroutine GETTEMP


SUB GR (J, I, AGRAPH, IODINE, IODIDE, TRIIODIDE, XX, RELAY()) STATIC
      '       Graphics Subroutine

DIM NUM1(0 TO 256)  AS INTEGER
DIM NUM2(0 TO 256)  AS INTEGER
DIM NUM3(0 TO 256)  AS INTEGER
DIM NUM4(0 TO 256)  AS INTEGER
DIM NUM5(0 TO 256)  AS INTEGER

DIM NUM6(0 TO 256)  AS INTEGER
DIM NUM7(0 TO 256)  AS INTEGER
DIM NUM8(0 TO 256)  AS INTEGER
DIM NUM9(0 TO 256)  AS INTEGER
DIM NUM10(0 TO 256) AS INTEGER
DIM NUM0(0 TO 256)   AS INTEGER
DIM HC(0 TO 1024)  AS INTEGER
DIM UD(0 TO 1024)  AS INTEGER
DIM RH(0 TO 1024)  AS INTEGER
DIM RP(0 TO 1024)  AS INTEGER

DIM RR(0 TO 1024)  AS INTEGER
DIM IR(0 TO 1024)  AS INTEGER
DIM I2L(0 TO 1024)  AS INTEGER
DIM IL(0 TO 1024)  AS INTEGER
DIM TL(0 TO 1024)  AS INTEGER

DIM MGL(0 TO 1024)  AS INTEGER
DIM RMCV1(0 TO 1024)  AS INTEGER
DIM RMCV2(0 TO 1024)  AS INTEGER
DIM RMCV3(0 TO 1024)  AS INTEGER
DIM RMCV4(0 TO 1024)  AS INTEGER

IF J = 0 THEN
    SCREEN 12
    COLOR 8
    VIEW PRINT 1 TO 30
    LOCATE 1, 1
    PRINT "HUMIDITY CONDENSATE"
    GET (0, 0)-(160, 12), HC
    CLS
    LOCATE 1, 1
    PRINT "URINE DISTILLATE"

    GET (0, 0)-(160, 12), UD
    CLS
    LOCATE 1, 1
    PRINT "RECLAIMED HYGIENE WATER"
    GET (0, 0)-(200, 12), RH

    CLS
    LOCATE 1, 1
    PRINT "RECLAIMED POTABLE WATER"
    GET (0, 0)-(200, 12), RP
    CLS

     COLOR 10
     LOCATE 1, 1
    PRINT "REGENERATION RELAYS"
    GET (0, 0)-(200, 12), RR
```

```
CLS

   LOCATE 1, 1
PRINT "IODINE MONITOR RELAYS"
GET (0, 0)-(200, 12), IR
CLS

LOCATE 1, 1
PRINT "Iodine"
GET (0, 0)-(60, 12), I2L
CLS

LOCATE 1, 1
COLOR 14
PRINT "Iodide"
GET (0, 0)-(60, 12), IL
CLS

COLOR 15
LOCATE 1, 1
PRINT "Triiodide"
GET (0, 0)-(80, 12), TL
CLS

COLOR 14
LOCATE 1, 1
PRINT " mg/L"
GET (0, 0)-(60, 16), MGL
CLS

COLOR 8
LOCATE 1, 1
PRINT "RMCV 1"
GET (0, 0)-(70, 12), RMCV1
CLS

LOCATE 1, 1
PRINT "RMCV 2"
GET (0, 0)-(70, 12), RMCV2
CLS

LOCATE 1, 1
PRINT "RMCV 3"
GET (0, 0)-(70, 12), RMCV3
CLS
```

```
 LOCATE 1, 1
 PRINT "RMCV 4"
 GET (0, 0)-(70, 12), RMCV4
 CLS
 COLOR 14

FOR II = 0 TO 10
 LOCATE 1, 1
 SELECT CASE II
   CASE 0
    PRINT "0"
    GET (0, 0)-(6, 12), NUM0
   CASE 1
    PRINT "1"
    GET (0, 0)-(6, 12), NUM1

   CASE 2
    PRINT "2"
    GET (0, 0)-(6, 12), NUM2
   CASE 3
    PRINT "3"
    GET (0, 0)-(6, 12), NUM3

   CASE 4
    PRINT "4"
    GET (0, 0)-(6, 12), NUM4
   CASE 5
    PRINT "5"
    GET (0, 0)-(6, 12), NUM5

   CASE 6
    PRINT "6"
    GET (0, 0)-(6, 12), NUM6
   CASE 7
    PRINT "7"
    GET (0, 0)-(6, 12), NUM7

   CASE 8
    PRINT "8"
    GET (0, 0)-(6, 12), NUM8
   CASE 9
    PRINT "9"
    GET (0, 0)-(6, 12), NUM9
```

```
        CASE 10
          PRINT "10"
          GET (0, 0)-(6, 12), NUM10
      END SELECT
    NEXT II
END IF

TEMPGRAPH = AGRAPH
IF (I = 1) THEN SCREEN 12
IF (I = 1) THEN PAINT (50, 50), 1
VIEW
WINDOW

IF RELAY(1) = 0 THEN LINE (0, 440)-(64, 460), 2, BF
IF RELAY(1) = 1 THEN LINE (0, 440)-(64, 460), 4, BF
IF RELAY(2) = 0 THEN LINE (70, 440)-(134, 460), 2, BF
IF RELAY(2) = 1 THEN LINE (70, 440)-(134, 460), 4, BF

IF RELAY(3) = 0 THEN LINE (140, 440)-(204, 460), 2, BF
IF RELAY(3) = 1 THEN LINE (140, 440)-(204, 460), 4, BF
IF RELAY(4) = 0 THEN LINE (210, 440)-(274, 460), 2, BF
IF RELAY(4) = 1 THEN LINE (210, 440)-(274, 460), 4, BF

IF RELAY(5) = 0 THEN LINE (350, 440)-(414, 460), 2, BF
IF RELAY(5) = 1 THEN LINE (350, 440)-(414, 460), 4, BF
IF RELAY(6) = 0 THEN LINE (420, 440)-(484, 460), 2, BF
IF RELAY(6) = 1 THEN LINE (420, 440)-(484, 460), 4, BF

IF RELAY(7) = 0 THEN LINE (490, 440)-(554, 460), 2, BF
IF RELAY(7) = 1 THEN LINE (490, 440)-(554, 460), 4, BF
IF RELAY(8) = 0 THEN LINE (560, 440)-(624, 460), 2, BF
IF RELAY(8) = 1 THEN LINE (560, 440)-(624, 460), 4, BF

PUT (10, 444), RMCV1
PUT (80, 444), RMCV2
PUT (150, 444), RMCV3
PUT (220, 444), RMCV4

PUT (360, 444), RMCV1
PUT (430, 444), RMCV2
PUT (500, 444), RMCV3
PUT (568, 444), RMCV4

PUT (55, 426), IR
PUT (410, 426), RR
```

```
WHILE (XX = 0)
  VIEW
  WINDOW
  PAINT (50, 50), 1
 LINE (0, 0)-(275, 200), 7, BF
 LINE (0, 0)-(275, 200), 12, B

 LINE (168, 205)-(258, 220), 5, BF
 PUT (188, 206), I2L
 LINE (268, 205)-(358, 220), 9, BF
 PUT (288, 206), IL
 LINE (368, 205)-(458, 220), 0, BF
 PUT (378, 206), TL

 LINE (350, 0)-(625, 200), 7, BF
 LINE (350, 0)-(625, 200), 12, B
 LINE (0, 225)-(275, 425), 7, BF
 LINE (0, 225)-(275, 425), 12, B
 LINE (350, 225)-(625, 425), 7, BF
 LINE (350, 225)-(625, 425), 12, B

 x = 25

 WHILE (x < 275)
   LINE (x, 0)-(x, 200), 1
   LINE (x, 225)-(x, 425), 1
  x = x + 25
 WEND

 x = 375

 WHILE (x < 625)
  LINE (x, 0)-(x, 200), 1
  LINE (x, 225)-(x, 425), 1
  x = x + 25
 WEND

 Y = 20

 WHILE (Y < 200)
  LINE (0, Y)-(275, Y), 1
  LINE (276, Y)-(280, Y), 15
  LINE (350, Y)-(625, Y), 1
  LINE (345, Y)-(349, Y), 15
  IF Y = 100 THEN LINE (327, Y)-(344, Y), 15
  IF Y = 100 THEN LINE (281, Y)-(300, Y), 15
  IF Y = 100 THEN PUT (311, 93), NUM5
  IF Y = 100 THEN PUT (292, 110), MGL
  Y = Y + 20
 WEND

 Y = 225

 WHILE (Y < 425)
  LINE (0, Y)-(275, Y), 1
  LINE (276, Y)-(280, Y), 15
  IF Y = 325 THEN LINE (281, Y)-(300, Y), 15
  LINE (350, Y)-(625, Y), 1
  LINE (345, Y)-(349, Y), 15

  IF Y = 325 THEN LINE (327, Y)-(344, Y), 15
  IF Y = 325 THEN PUT (311, 318), NUM5
  IF Y = 325 THEN PUT (292, 335), MGL
  Y = Y + 20
 WEND

 PUT (118, 5), RMCV1
 PUT (45, 23), RP
 PUT (468, 5), RMCV2
 PUT (400, 23), RH

 PUT (118, 230), RMCV3
 PUT (60, 248), HC
 PUT (468, 230), RMCV4
 PUT (427, 248), UD
 XX = XX + 5
WEND

SELECT CASE AGRAPH
    CASE 1
       VIEW (0, 0)-(275, 200)
       WINDOW (0, 100)-(110, 0)
    CASE 2
       VIEW (350, 0)-(625, 200)
       WINDOW (0, 100)-(110, 0)
    CASE 3
       VIEW (0, 225)-(275, 425)
       WINDOW (0, 100)-(110, 0)
```

```
        CASE 4
            VIEW (350, 225)-(625, 425)
            WINDOW (0, 100)-(110, 0)
        END SELECT

        IIODINE = IODINE * 10
        IIODIDE = IODIDE * 10
        TTRIIODIDE = TRIIODIDE * 10
        LINE (XX - 1, 0)-(XX + 1, IIODINE), 5, BF
        LINE (XX - 1, IIODINE)-(XX + 1, IIODINE + IIODIDE), 9, BF
        LINE (XX - 1, IIODINE + IIODIDE)-(XX + 1, IIODINE + IIODIDE +
TTRIIODIDE), 0, BF
        J = J + 1
END SUB      ' End of Graphics Subroutine


SUB INITIALIZE (STREAMS()) STATIC
   FOR K = 1 TO 4
        CLS
        BEEP
        LOCATE 23, 10

        PRINT " STREAM NO."; K; "  ---- FLOWING ? (1=YES  0 = NO) "
        LOCATE 25, 40
        INPUT STREAMS(K)
   NEXT K
   CLS
END SUB      ' End of Subroutine INITIALIZE


SUB LAMPOFF
    x = LAMPSTATUS
    IF x <> 0 THEN LAMP 0!
END SUB      ' End of LAMPOFF Subroutine


SUB LAMPON
    x = LAMPSTATUS
    IF x = 0 THEN LAMP 1!
END SUB      ' End of Subroutine LAMPON


SUB MONITORIODINEONSCREEN STATIC
  MEASURE
   FOR I% = 1 TO 4
```

```
    PRINT USING "WAVE : ###    ABS: #.####"; RAW(I%, 1); RAW(I%, 2)
  NEXT I%
END SUB      'End of Subroutine MONITORIODINEONSCREEN


SUB NEWTIME (SETTIME)
  VIEW PRINT 30 TO 30
  FOR I = 1 TO 100
   BEEP
  NEXT I
  INPUT " ENTER NEW TIME BETWEEN SAMPLES  <RETURN> ", SETTIME
END SUB      'End of Subroutine NEWTIME


SUB PASSIVALUES (IODIDE, TRIIODIDE, IODINE, BASELINE)
 MEASURE
 IODIDE = RAW(1, 2)
 TRIIODIDE = RAW(2, 2)
 IODINE = RAW(3, 2)
 BASELINE = RAW(4, 2)
END SUB      'End of Subroutine PASSIVALUES


SUB READCYCLE (IFAC(), I2FAC(), IFUDGE(), Y$, YY$(), STREAMS(), AGRAPH,
LAST$) STATIC

FOR I = 1 TO 4
 REGEN(I) = 0
NEXT I

RINDEX = 0
CLS
VIEW PRINT 1 TO 25
LOCATE 20, 20

PRINT " PRESS Q KEY TO EXIT PROGRAM "
INPUT " ENTER THE DESIRED NUMBER OF MINUTES BETWEEN SAMPLES ",
SETTIME
CLS
INPUT " GET FLOW CELL WORKING THEN <RETURN> TO INITIATE DATA
LOGGING ", R
LAST$ = TIME$

 J = 0
 FLOWNUM = 0
```

```
RESETI = 0
I = 0
XX = 0

 FOR m = 1 TO 8
  RELAY(m) = 0
  IF (m < 5) AND (STREAMS(m) = 1) THEN FLOWNUM = FLOWNUM + 1
 NEXT m

 m = 1

  WHILE m <= 4
    SELECT CASE STREAMS(m)
     CASE 1
          AGRAPH = m
          m = 5
     CASE 0
          m = m + 1
    END SELECT
  WEND

    TOGGLE (AGRAPH)
    RELAY(AGRAPH) = 1

IF (REGEN(AGRAPH) = 1) AND (RELAY(AGRAPH + 4) = 0) THEN TOGGLE
(AGRAPH + 4)
    IF REGEN(AGRAPH) = 1 THEN RELAY(AGRAPH + 4) = 1

WHILE (((Y$ = "y") OR (Y$ = "Y")) AND ((INKEY$ <> "Q") AND (INKEY$ <>
"q")))
  IF (INKEY$ = "b") OR (INKEY$ = "B") THEN CALL SETBLANK
IF (INKEY$ = "C") OR (INKEY$ = "c") THEN CALL RECALIBRATE(IFAC(),
I2FAC(), IFUDGE())
IF (INKEY$ = "T") OR (INKEY$ = "t") THEN CALL NEWTIME(SETTIME)

IF (I = FLOWNUM) THEN RESETI = 1 ELSE RESETI = 0
IF (I <> FLOWNUM) THEN I = I + 1
IF RESETI = 1 THEN I = 1

  S$ = TIME$
  T$ = MID$(S$, 4, 2)
  TIME = VAL(T$)
  INITTIME = TIME
  NETTIME = 0
```

```
IF REGEN(AGRAPH) = 1 THEN NETTIME = TIME + 5
IF REGEN(AGRAPH) = 0 THEN NETTIME = TIME + SETTIME
VIEW PRINT 30 TO 30
LOCATE 30, 1

WHILE (NETTIME - TIME) > 0
     S$ = TIME$
   PRINT "PRESENT TIME: "; S$; " LAST SAMPLE TIME: "; LAST$; "           ."
      T$ = MID$(S$, 4, 2)
    TIME = VAL(T$)

    PRINT "PRESENT TIME: "; S$; " LAST SAMPLE TIME: "; LAST$; "           ."
    IF (INKEY$ = "q") OR (INKEY$ = "Q") THEN EXIT SUB
    IF TIME < INITTIME THEN TIME = TIME + 60
    PRINT "PRESENT TIME: "; S$; " LAST SAMPLE TIME: "; LAST$; "           ."
WEND

LAST$ = TIME$
BEEP
CALL PASSIVALUES(IODIDE, TRIIODIDE, IODINE, BASELINE)
I228 = IODIDE
I350 = TRIIODIDE

I464 = IODINE
I600 = BASELINE

 IF AGRAPH = 3 THEN IODINE = (IODINE - BASELINE) * I2FAC(AGRAPH) * 2.0
- 2.0
 IF AGRAPH = 4 THEN IODINE = (IODINE - BASELINE) * I2FAC(AGRAPH) * 1.7
- 1.5
 IF AGRAPH = 2 THEN IODINE = (IODINE - BASELINE) * I2FAC(AGRAPH) * 1.7
 IF AGRAPH = 1 THEN IODINE = (IODINE - BASELINE) * I2FAC(AGRAPH) *
1.227 - .7848

TRIIODIDE = (TRIIODIDE - BASELINE) * .021
IF TRIIODIDE < 0 THEN TRIIODIDE = 0
IODIDE = IODIDE - BASELINE
I2228 = IFUDGE(AGRAPH) * IODINE
IODIDE = IODIDE - I2228

IODIDE = IFAC(AGRAPH) * IODIDE
IF IODINE < 0 THEN IODINE = 0
IF IODIDE < 0 THEN IODIDE = 0
CALL GR(J, I, AGRAPH, IODINE, IODIDE, TRIIODIDE, XX, RELAY())
S$ = TIME$
```

```
  IF IODINE < 2 THEN REGEN(AGRAPH) = 1
  IF (REGEN(AGRAPH) = 1) AND (RELAY(AGRAPH + 4) = 0) THEN TOGGLE
(AGRAPH + 4)
  IF REGEN(AGRAPH) = 1 THEN RELAY(AGRAPH + 4) = 1

 SELECT CASE AGRAPH
  CASE 1
     RTIME = 7
  CASE 2
     RTIME = 9
  CASE 3
     RTIME = 9
  CASE 4
     RTIME = 11
 END SELECT


IF (RINDEX < RTIME) THEN RSTOP = 999
 IF (RINDEX >= RTIME) THEN
  SELECT CASE AGRAPH
   CASE 1
     RSTOP = 10
   CASE 2
     RSTOP = 10
   CASE 3
    RSTOP = 10
   CASE 4
     RSTOP = 7
  END SELECT
 END IF

IF (REGEN(AGRAPH) = 1) THEN RINDEX = RINDEX + 1
IF RINDEX >= 30 THEN RSTOP = 0
IF (REGEN(AGRAPH) = 1) AND (IODINE > RSTOP) THEN RINDEX = 0
IF (REGEN(AGRAPH) = 1) AND (IODINE > RSTOP) THEN REGEN(AGRAPH) = 0

IF (REGEN(AGRAPH) = 0) AND (RELAY(AGRAPH + 4) = 1) THEN TOGGLE
(AGRAPH + 4)
IF (REGEN(AGRAPH) = 0) AND (RELAY(AGRAPH + 4) = 1) THEN
RELAY(AGRAPH + 4) = 0
VIEW
WINDOW

IF RELAY(1) = 0 THEN LINE (0, 440)-(64, 460), 2, BF
IF RELAY(1) = 1 THEN LINE (0, 440)-(64, 460), 4, BF
IF RELAY(2) = 0 THEN LINE (70, 440)-(134, 460), 2, BF
IF RELAY(2) = 1 THEN LINE (70, 440)-(134, 460), 4, BF

IF RELAY(3) = 0 THEN LINE (140, 440)-(204, 460), 2, BF
IF RELAY(3) = 1 THEN LINE (140, 440)-(204, 460), 4, BF
IF RELAY(4) = 0 THEN LINE (210, 440)-(274, 460), 2, BF
IF RELAY(4) = 1 THEN LINE (210, 440)-(274, 460), 4, BF

IF RELAY(5) = 0 THEN LINE (350, 440)-(414, 460), 2, BF
IF RELAY(5) = 1 THEN LINE (350, 440)-(414, 460), 4, BF
IF RELAY(6) = 0 THEN LINE (420, 440)-(484, 460), 2, BF
IF RELAY(6) = 1 THEN LINE (420, 440)-(484, 460), 4, BF

IF RELAY(7) = 0 THEN LINE (490, 440)-(554, 460), 2, BF
IF RELAY(7) = 1 THEN LINE (490, 440)-(554, 460), 4, BF
IF RELAY(8) = 0 THEN LINE (560, 440)-(624, 460), 2, BF
IF RELAY(8) = 1 THEN LINE (560, 440)-(624, 460), 4, BF


IF YY$(2) = "Y" THEN N = 0 ELSE N = 5

WHILE N < 1
 LPRINT "No."; AGRAPH; " "; S$; " I2 = "; IODINE; " I- = "; IODIDE; " I3- = ";
TRIIODIDE
 N = N + 1
WEND

IF REGEN(AGRAPH) = 1 THEN REGEN$ = "  REGENERATION CYCLE ON"
IF REGEN(AGRAPH) = 0 THEN REGEN$ = " REGENERATION CYCLE OFF"

   D$ = DATE$
   T$ = TIME$
   T$ = T$ + " "
   DD$ = LEFT$(D$, 5) + "." + RIGHT$(D$, 2)
   F$ = "R-" + DD$
   FB$ = "B:" + F$

CALL GETTEMP(TEMP)

IF YY$(1) = "Y" THEN L = 7
                 WHILE L = 7
                  OPEN F$ FOR APPEND AS #1
                  OPEN FB$ FOR APPEND AS #2
                  WRITE #1, AGRAPH, T$, I228, I350, I464, I600
```

```
                WRITE #1, IODINE, IODIDE, TEMP, REGEN$

                WRITE #2, AGRAPH, T$, I228, I350, I464, I600
                WRITE #2, IODINE, IODIDE, TEMP, REGEN$
                L = L + 1
                CLOSE #1
                CLOSE #2
               WEND


     WHILE (((XX = 110) AND (I = FLOWNUM)) OR (XX > 110))
      VIEW
      WINDOW
      CLS
      PAINT (30, 30), 1
      XX = 0
      CALL GR(J, I, AGRAPH, IODINE, IODIDE, TRIIODIDE, XX, RELAY())
     WEND

     IF (I = FLOWNUM) OR (REGEN(AGRAPH) = 1) THEN XX = XX + 5

     YY = 0
     RESETI = 0
     IF (FLOWNUM = 1) OR (REGEN(AGRAPH) = 1) THEN GOTO 1
     TOGGLE (AGRAPH)
     RELAY(AGRAPH) = 0

     IF (REGEN(AGRAPH) = 0) AND (RELAY(AGRAPH + 4) = 1) THEN TOGGLE
(AGRAPH + 4)
    IF (REGEN(AGRAPH) = 0) AND (RELAY(AGRAPH + 4) = 1) THEN
RELAY(AGRAPH + 4) = 0
    VIEW
    WINDOW

IF RELAY(1) = 0 THEN LINE (0, 440)-(64, 460), 2, BF
IF RELAY(1) = 1 THEN LINE (0, 440)-(64, 460), 4, BF
IF RELAY(2) = 0 THEN LINE (70, 440)-(134, 460), 2, BF
IF RELAY(2) = 1 THEN LINE (70, 440)-(134, 460), 4, BF

IF RELAY(3) = 0 THEN LINE (140, 440)-(204, 460), 2, BF
IF RELAY(3) = 1 THEN LINE (140, 440)-(204, 460), 4, BF
IF RELAY(4) = 0 THEN LINE (210, 440)-(274, 460), 2, BF
IF RELAY(4) = 1 THEN LINE (210, 440)-(274, 460), 4, BF

IF RELAY(5) = 0 THEN LINE (350, 440)-(414, 460), 2, BF
IF RELAY(5) = 1 THEN LINE (350, 440)-(414, 460), 4, BF
IF RELAY(6) = 0 THEN LINE (420, 440)-(484, 460), 2, BF
IF RELAY(6) = 1 THEN LINE (420, 440)-(484, 460), 4, BF

IF RELAY(7) = 0 THEN LINE (490, 440)-(554, 460), 2, BF
IF RELAY(7) = 1 THEN LINE (490, 440)-(554, 460), 4, BF
IF RELAY(8) = 0 THEN LINE (560, 440)-(624, 460), 2, BF
IF RELAY(8) = 1 THEN LINE (560, 440)-(624, 460), 4, BF

 WHILE (YY = 0)
   TEMPGRAPH = AGRAPH

   SELECT CASE AGRAPH
    CASE 1
        IF STREAMS(2) = 1 THEN TEMPGRAPH = 2
        IF (STREAMS(3) = 1) AND (STREAMS(2) = 0) THEN TEMPGRAPH = 3
IF ((STREAMS(4) = 1) AND (STREAMS(3) = 0) AND (STREAMS(2) = 0)) THEN
TEMPGRAPH = 4

    CASE 2
        IF STREAMS(3) = 1 THEN TEMPGRAPH = 3
        IF (STREAMS(3) = 0) AND (STREAMS(4) = 1) THEN TEMPGRAPH = 4
IF ((STREAMS(1) = 1) AND (STREAMS(4) = 0) AND (STREAMS(3) = 0)) THEN
TEMPGRAPH = 1

    CASE 3
        IF STREAMS(4) = 1 THEN TEMPGRAPH = 4
        IF (STREAMS(4) = 0) AND (STREAMS(1) = 1) THEN TEMPGRAPH = 1
IF ((STREAMS(2) = 1) AND (STREAMS(1) = 0) AND (STREAMS(4) = 0)) THEN
TEMPGRAPH = 2

    CASE 4
        IF STREAMS(1) = 1 THEN TEMPGRAPH = 1
        IF (STREAMS(1) = 0) AND (STREAMS(2) = 1) THEN TEMPGRAPH = 2
IF ((STREAMS(3) = 1) AND (STREAMS(2) = 0) AND (STREAMS(1) = 0)) THEN
TEMPGRAPH = 3
   END SELECT

   AGRAPH = TEMPGRAPH
   TOGGLE (AGRAPH)
   RELAY(AGRAPH) = 1
  IF (REGEN(AGRAPH) = 1) AND (RELAY(AGRAPH + 4) = 0) THEN TOGGLE
(AGRAPH + 4)
  IF REGEN(AGRAPH) = 1 THEN RELAY(AGRAPH + 4) = 1
```

```
   VIEW
   WINDOW

IF RELAY(1) = 0 THEN LINE (0, 440)-(64, 460), 2, BF
IF RELAY(1) = 1 THEN LINE (0, 440)-(64, 460), 4, BF
IF RELAY(2) = 0 THEN LINE (70, 440)-(134, 460), 2, BF
IF RELAY(2) = 1 THEN LINE (70, 440)-(134, 460), 4, BF

IF RELAY(3) = 0 THEN LINE (140, 440)-(204, 460), 2, BF
IF RELAY(3) = 1 THEN LINE (140, 440)-(204, 460), 4, BF
IF RELAY(4) = 0 THEN LINE (210, 440)-(274, 460), 2, BF
IF RELAY(4) = 1 THEN LINE (210, 440)-(274, 460), 4, BF

IF RELAY(5) = 0 THEN LINE (350, 440)-(414, 460), 2, BF
IF RELAY(5) = 1 THEN LINE (350, 440)-(414, 460), 4, BF
IF RELAY(6) = 0 THEN LINE (420, 440)-(484, 460), 2, BF
IF RELAY(6) = 1 THEN LINE (420, 440)-(484, 460), 4, BF

IF RELAY(7) = 0 THEN LINE (490, 440)-(554, 460), 2, BF
IF RELAY(7) = 1 THEN LINE (490, 440)-(554, 460), 4, BF
IF RELAY(8) = 0 THEN LINE (560, 440)-(624, 460), 2, BF
IF RELAY(8) = 1 THEN LINE (560, 440)-(624, 460), 4, BF

YY = 1
WEND

1     YY = 0

      FOR J = 1 TO 10000
      NEXT J
WEND

END SUB       'End Subroutine READCYCLE


SUB RECALIBRATE (IFAC(), I2FAC(), IFUDGE()) STATIC
  VIEW PRINT 30 TO 30

   FOR I = 1 TO 100
     BEEP
   NEXT I

   FLAG$ = "Y"

  WHILE (FLAG$ = "y") OR (FLAG$ = "Y")
```

```
    INPUT " WHICH STREAM DO YOU WISH TO RECALIBRATE "; N
    BEEP
    INPUT " Enter the new value for I2factor "; TFACTOR
    I2FAC(N) = I2FAC(N) * TFACTOR

    BEEP
    INPUT " Enter the new value for I-factor "; TFACTOR
    IFAC(N) = IFAC(N) * TFACTOR
    BEEP
    INPUT " Enter the new value for Fudge "; TFACTOR

    IFUDGE(N) = IFUDGE(N) * TFACTOR
    BEEP
    INPUT " DO YOU WANT TO RECALIBRATE ANOTHER STREAM (Y/N) ";
FLAG$
    BEEP
  WEND

  PRINT
END SUB      'End Subroutine RECALIBRATE


SUB SETBLANK
   VIEW PRINT 30 TO 30
   INPUT " PUT BLANK IN CELL AND PRESS <ENTER> TO READ IN BLANK
VALUE "; R
   BLANK
   BEEP
END SUB      'End Subroutine SETBLANK

SUB SETINTEGRATIONTIME (SECONDS) STATIC
 INTEG.TIME = SECONDS
END SUB      'End Subroutine SETINTEGRATIONTIME


SUB SETIODINEMODE STATIC
        WAVEMODE = 1
        NUMWAVES = 4
        WAVELIST(1) = 228
        WAVELIST(2) = 350
        WAVELIST(3) = 464
        WAVELIST(4) = 600
END SUB      'End Subroutine SETIODINEMODE
```

```
SUB TOGGLE (REL) STATIC
 DIM RELAYS%(8)

   RELAYS%(1) = 1
   FOR I = 1 TO 7
        RELAYS%(I + 1) = RELAYS%(I) * 2
   NEXT I

   CWORD% = &H8B
   PORTA% = &H300
   CADDR% = &H303
   RELSTATE% = RELSTATE% XOR RELAYS%(REL)
   OUT PORTA%, RELSTATE%
END SUB      ' End of Subroutine TOGGLE
```

```
/*    ispecies.C  iodine equilibrium program */
/*    version 1.0 - James E. Atwater  */

#include <stdlib.h>
#include <stdio.h>
#include <graphics.h>
#include <string.h>
#include <math.h>
#include <conio.h>
#include <io.h>
#include <fcntl.h>
#include <sys\stat.h>

main()
{
double io3, i, i2, hoi,i3, it, h, k1, k2, k3,ph;
double oi,hio3,i2oh,oh,i2o,h2oi,dh;
double convergence, criterion, root,h6;
double  nexti2guess, itcalc,k1acalc,k4,k5,k6,k7,k8;
double  i2ratio,iratio,i3ratio,hoiratio,io3ratio;

int  j,k,n,m;
char *file_name;
FILE *file_variable;

   k1  = 1.25e-47;
   k2 = 4.83e-13;

   for (m = 1; m <= 11;)
   {
    switch (m)
    {
        case 1: i2  = 1.0;file_name ="b:io3equil.1";break;
        case 2: i2  = 2.0;file_name ="b:io3equil.2";break;
        case 3: i2  = 3.0;file_name ="b:io3equil.3";break;
        case 4: i2  = 4.0;file_name ="b:io3equil.4";break;
        case 5: i2  = 5.0;file_name ="b:io3equil.5";break;

        case 6: i2  =10.0;file_name ="b:io3equil.10";break;
        case 7: i2  =20.0;file_name ="b:io3equil.20";break;
        case 8: i2  =50.0;file_name ="b:io3equil.50";break;
        case 9: i2  =100.0;file_name ="b:io3equil.100";break;
        case 10: i2  =200.0;file_name="b:io3equil.200";break;
        case 11: i2  =300.0;file_name ="b:io3equil.300";break;
```

```
  }

file_variable = fopen(file_name, "wt");
i2  =i2 / 126900.0 / 2.0;
it  =i2 * 2.0;
root  =1.0 / 6.00;
criterion  =it * 0.000001;

for (k = 0; k < 141;)
{
    ph  = k * 0.10;
    ph  = -1.0 * ph;
    h  = pow(10.0,ph);
    ph  =-1.0 * ph;
    oh  = 1.0000000000e-14 / h;

    convergence  = 1.0;
    nexti2guess  = 1.0;
    j =0;

    while (convergence > criterion)
    {
      j++;
      i2  = nexti2guess * i2;
      io3  = k1 / pow(h,6) / 3125.0 * pow(i2,3);
    io3  = pow(io3, root);
    i  = 5.0 * io3;

      i3  = 736.0 * i * i2;
      hoi  = k2 * i2 / i / h;
      oi  =2.300e-11 * hoi / h;
      hio3  =io3 * h / 0.164;
      i2oh  =0.13 / oh * i * oi;

      i2o  =0.045 * oi * i;
      h2oi  =hoi * h / 0.03;
    itcalc  =io3+2*i2+i+3*i3+hoi+oi+hio3+2*i2oh+2*i2o+h2oi;
      clrscr();

      printf("i2 = %g\n",i2);
      printf("io3 = %g\n",io3);
      printf("i-= %g\n",i);
      printf("i3 = %g\n",i3);
      printf("hoi = %g\n",hoi);
      printf("oi- = %g\n",oi);
    printf("i2oh- = %g\n",i2oh);
    printf("i2o-- = %g\n",i2o);
    printf("h2oi+ = %g\n",h2oi);
    printf("itcalc = %g\n",itcalc);

    printf("itotal = %g\n",it);
    printf("iteration  %d\n",j);
    printf("convergence = %g\n",convergence);
    printf(" m = %d\n",m);
    printf(" k = %d\n",k);
    nexti2guess  = it / itcalc;
    convergence  = fabs(it-itcalc);
  }

k1  = io3/pow(i2,3)*pow(h,6)*pow(i,5);
k3  = i3 / i2 / i;
k2  = hoi * i * h / i2;
k4  = oi / hoi * h;
k5  = io3 * h / hio3;

k6  = i2oh * oh / oi / i;
k7  = i2o / oi / i;
k8  =hoi * h / h2oi;
dh  = io3 * 6 + hoi - hio3;
dh  = - log10(dh);

dh  = ph + dh;
i2  = i2 * 126900.0 * 2.0;
i  = i * 126900.0;
i3  = i3 * 126900.0 * 3.0;
hoi = 126900.0 * hoi;

oi  = oi * 126900.0;
io3  = io3 * 126900.0;
hio3  = hio3 * 126900.0;
i2oh = i2oh * 2.0 * 126900.0;
i2o  = i2o * 2.0 * 126900.0;

h2oi = h2oi * 126900.0;
it  = it * 126900.0;
itcalc = itcalc * 126900.0;
clrscr();
printf("i2 = %g\n",i2);
printf("io3 = %g",io3); printf("    k1 = %g\n",k1);
printf("i-= %g\n",i);
```

```
            printf("i3 = %g",i3);   printf("     k3= %g\n",k3);
            printf("hoi = %g",hoi); printf("     k2= %g\n",k2);
            printf("oi- = %g",oi);  printf("     k4= %g\n",k4);
            printf("hio3 = %g", hio3); printf("      k5= %g\n",k5);

            printf("i2oh- = %g",i2oh); printf("      k6= %g\n",k6);
            printf("i2o-- = %g",i2o); printf("      k7= %g\n",k7);
            printf("h2oi+ = %g",h2oi);printf("      k8= %g\n",k8);
            printf("final pH = %g",ph);printf(" init ph = %g\n",dh);
            printf("itcalc = %g\n",itcalc);

            printf("itotal = %g\n",it);
            printf("iteration %d\n",j);
            printf("convergence %g\n",convergence);
            fprintf(file_variable,"%g %g %g %g %g %g %g %g %g %g
                 %g\n",ph,i2,i,i3,hoi,io3,oi,hio3,i2oh,i2o,h2oi);

            it  = it / 126900.0;
            k++;
      }     /* end of for k loop */

      fclose(file_variable);
      m++;
      }   /*  end of for m loop */

} /*  of main */         /*   End of program ispecies  */
```

```
/*    ipseudo.C  iodine equilibrium program */
/*    version 3.0 - Author: James E. Atwater  */



#include <stdlib.h>
#include <stdio.h>
#include <graphics.h>
#include <string.h>
#include <math.h>
#include <conio.h>
#include <io.h>
#include <fcntl.h>
#include <sys\stat.h>


main()
{
double  i, i2, hoi,i3, it, h, k1, k2, k3,ph, t_hoi, next_hoi_guess;
double oi,i2oh,oh,i2o,h2oi,dh,hoi_calc,h_convergence,h_criterion;
double convergence, criterion, root,h6,next_h_guess,h_calc;
double  nexti2guess, itcalc,k1acalc,k4,k5,k6,k7,k8;
double  i2ratio,iratio,i3ratio,hoiratio,io3ratio;

int  j,k,n,m;
char *file_name;
FILE *file_variable;

   k1  = 1.25e-47;
   k2 = 4.83e-13;

   for (m = 1; m <= 11;)
   {
    switch (m)
    {
        case 1: i2  = 1.0;file_name ="b:hoieqcor.1";break;
        case 2: i2  = 2.0;file_name ="b:hoieqcor.2";break;
        case 3: i2  = 3.0;file_name ="b:hoieqcor.3";break;
        case 4: i2  = 4.0;file_name ="b:hoieqcor.4";break;

        case 5: i2  = 5.0;file_name ="b:hoieqcor.5";break;
        case 6: i2  =10.0;file_name ="b:hoieqcor.10";break;
        case 7: i2  =20.0;file_name ="b:hoieqcor.20";break;
        case 8: i2  =50.0;file_name ="b:hoieqcor.50";break;
        case 9: i2  =100.0;file_name ="b:hoieqcor.100";break;
        case 10: i2  =200.0;file_name="b:hoieqcor.200";break;
        case 11: i2  =300.0;file_name ="b:hoieqcor.300";break;
    }

  file_variable = fopen(file_name, "wt");
  i2  =i2 / 126900.0 / 2.0;
  it  =i2 * 2.0;
  root  =1.0 / 6.00;
  criterion  =it * 0.00000001;

  for (k = 0; k < 71;)
  {
      ph  = k * 0.20;
      ph  = -1.0 * ph;
      h  = pow(10.0,ph);
      ph  =-1.0 * ph;
      oh  = 1.0000000000e-14 / h;

      convergence  = 1.0;
      nexti2guess  = 1.0;
      j  =0;

      while (convergence > criterion)
      {
        j++;
        i2  = nexti2guess * i2;
        hoi  = k2 * i2 / h;
        hoi = pow(hoi,0.5);
        i = hoi;
        i3  = 736.0 * i * i2;
        oi = 2.3e-11 * hoi/h;

       if (oi > i)
       {
        oi = i;
       }

         i2oh  =0.13 / oh * i * oi;
         i2o  =0.045 * oi * i;
         h2oi  =hoi * h / 0.03;
        itcalc  =2*i2+i+3*i3+hoi;
        clrscr();

        printf("i2 = %g\n",i2);
        printf("i-= %g\n",i);
        printf("i3 = %g\n",i3);
        printf("hoi = %g\n",hoi);
        printf("oi- = %g\n",oi);

        printf("i2oh- = %g\n",i2oh);
        printf("i2o-- = %g\n",i2o);
        printf("h2oi+ = %g\n",h2oi);
        printf("itcalc = %g\n",itcalc);
        printf("itotal = %g\n",it);

        printf("iteration  %d\n",j);
        printf("convergence = %g\n",convergence);
        printf(" m = %d\n",m);
        printf(" k = %d\n",k);
        nexti2guess  = it / itcalc;
        convergence  = fabs(it-itcalc);

      }  /* end of First While  */

     h_convergence = hoi;
     h_criterion = hoi/1.0e6;
     t_hoi = hoi;
     next_h_guess = 1;

     while(h_convergence > h_criterion)
     {
      hoi = next_h_guess * hoi;
    oi = 2.3e-11 * hoi/h;
      i2oh  =0.13 / oh * i * oi;
      i2o  =0.045 * oi * i;
      h2oi  =hoi * h / 0.03;

      h_calc = hoi + oi + 2* i2oh + 2* i2o + h2oi;
      next_h_guess = t_hoi/h_calc;
      h_convergence = fabs(t_hoi-h_calc);
     }
     k3  = i3 / i2 / i;
     k2  = hoi * i * h / i2;
     k4  = oi / hoi * h;
     k6  = i2oh * oh / oi / i;
     k7  = i2o / oi / i;

     k8  =hoi * h / h2oi; */
     i2  = i2 * 126900.0 * 2.0;
```

```
            i  = i * 126900.0;
            i3  = i3 * 126900.0 * 3.0;
            hoi = 126900.0 * hoi;
            oi  = oi * 126900.0;
            i2oh = i2oh * 2.0 * 126900.0;

            i2o  = i2o * 2.0 * 126900.0;
            h2oi = h2oi * 126900.0;
            it  = it * 126900.0;
            itcalc = itcalc * 126900.0;
            clrscr();

            printf("i2 = %g\n",i2);
            printf("i-= %g\n",i);
            printf("i3 = %g",i3);  printf("     k3= %g\n",k3);
            printf("hoi = %g",hoi); printf("     k2= %g\n",k2);
            printf("oi- = %g",oi);  printf("     k4= %g\n",k4);

            printf("i2oh- = %g",i2oh); printf("       k6= %g\n",k6);
            printf("i2o-- = %g",i2o); printf("        k7= %g\n",k7);
            printf("h2oi+ = %g",h2oi);printf("        k8= %g\n",k8);
            printf(" init ph = %g\n",dh);
            printf("itcalc = %g\n",itcalc);

            printf("itotal = %g\n",it);
            printf("iteration %d\n",j);
            printf("convergence %g\n",convergence);
            fprintf(file_variable,"%g %g %g %g %g %g %g %g
               %g\n",ph,i2,i,i3,hoi,oi,i2oh,i2o,h2oi);

            it  = it / 126900.0;
            k++;
        }      /* end of for k loop */

        fclose(file_variable);

      m++;
      }   /*  end of for m loop */

    } /*  of main */
```

```
/*    iifast.C  iodine equilibrium program  with iodine initially present */
/*    version 3.0 - Author: James E. Atwater  */

#include <stdlib.h>
#include <stdio.h>
#include <graphics.h>
#include <string.h>

#include <math.h>
#include <conio.h>
#include <io.h>
#include <fcntl.h>
#include <sys\stat.h>

main()
{
double  ii,i, i2, hoi,i3, it, h, k1, k2, k3,ph, t_hoi, next_hoi_guess;
double oi,i2oh,oh,i2o,h2oi,dh,hoi_calc,h_convergence,h_criterion;
double convergence, criterion, root,h6,next_h_guess,h_calc;
double  nexti2guess, itcalc,k1acalc,k4,k5,k6,k7,k8;
double  i2ratio,iratio,i3ratio,hoiratio,io3ratio;

double o_guess_1, o_guess_2, iit, iitcalc, temp_i;
int  j,k,n,m;
char *file_name;
FILE *file_variable;

    k1  = 1.25e-47;
    k2 = 4.83e-13;

    for (m = 1; m <= 11;)
    {
     switch (m)
     {    case 1: i2  = 1.0;file_name ="b:ihoieq1.1";break;
          case 2: i2  = 2.0;file_name ="b:ihoieq1.2";break;
          case 3: i2  = 3.0;file_name ="b:ihoieq1.3";break;
          case 4: i2  = 4.0;file_name ="b:ihoieq1.4";break;
          case 5: i2  = 5.0;file_name ="b:ihoieq1.5";break;

          case 6: i2  =10.0;file_name ="b:ihoieq1.10";break;
          case 7: i2  =20.0;file_name ="b:ihoieq1.20";break;
          case 8: i2  =50.0;file_name ="b:ihoieq1.50";break;
          case 9: i2  =100.0;file_name ="b:ihoieq1.100";break;
          case 10: i2  =200.0;file_name="b:ihoieq1.200";break;
          case 11: i2  =300.0;file_name ="b:ihoieq1.300";break;
```

```
        }

    file_variable = fopen(file_name, "wt");

    i2  =i2 / 126900.0 / 2.0;
    it  =i2 * 2.0;
    ii = 1.0 * it;
    iit = ii + it;

    root  =1.0 / 6.00;
    criterion  =it * 0.0000001;

    for (k = 0; k < 71;)
    {
        ph  = k * 0.20;
        ph  = -1.0 * ph;
        h  = pow(10.0,ph);
        ph  =-1.0 * ph;
        oh  = 1.0000000000e-14 / h;

        convergence  = 1.0;
        nexti2guess  = 1.0;
        j  =0;
        hoi = i2 * k2/h/ii;
        oi = 0.0;

        i3 = 0.0;
        i2oh = 0.0;
        i2o = 0.0;
        h2oi = 0.0;

        while (convergence > criterion)
        {
            j++;
         if (ph < 7.50)
         {
            i2 = i2 * nexti2guess;
            i = ii + (hoi+oi+2*i2oh+2*i2o+h2oi-i3);
            hoi = i2 * k2/h/i;
            i3 = 736.0 * i * i2;

            oi = 2.3e-11 * hoi/h;
            i2oh  =0.13 / oh * i * oi;
            i2o  =0.045 * oi * i;
            h2oi  =hoi * h / 0.03;
```

```
}

else
{
   hoi = hoi * nexti2guess;
   i2 = hoi * h * i/k2;
   i = ii + hoi;
   i3 = 736.0 * i * i2;

  oi = 2.3e-11 * hoi/h;
  i2oh  =0.13 / oh * i * oi;
  i2o  =0.045 * oi * i;
  h2oi  =hoi * h / 0.03;
 }
 n=0;
 while ((hoi+oi+2*i2oh +2*i2o+h2oi) > (0.5 * it))
   {
      n++;
      iratio = it / 2*(hoi+oi+2*i2oh +2*i2o+h2oi);
      hoi = hoi* iratio;
      oi = oi* iratio;
      i2oh = i2oh* iratio;

      i2o  = i2o* iratio;
      h2oi  =h2oi* iratio;
      hoi_calc = (hoi+oi+2*i2oh +2*i2o+h2oi);
      printf(" it = %g ",it);printf("  n= %d ",n);
      printf("hoispecies = %g\n", hoi_calc);
   }

 temp_i = i - ii + (ii/i*i3);
 itcalc  =2*i2+temp_i+3*i3+hoi+oi+2*i2oh+2*i2o+h2oi;

 printf("i2 = %g\n",i2);
 printf("i-= %g\n",i);
 printf("i3 = %g\n",i3);
 printf("hoi = %g\n",hoi);
 printf("oi- = %g\n",oi);

 printf("i2oh- = %g\n",i2oh);
 printf("i2o-- = %g\n",i2o);
 printf("h2oi+ = %g\n",h2oi);
 printf("pH = %g\n",ph);
 printf("itcalc = %g\n",itcalc);
```

```
    printf("itotal = %g\n",it);
    printf("iteration  %d\n",j);
    printf("convergence = %g\n",convergence);
    printf(" m = %d\n",m);
    printf(" k = %d\n",k);
    nexti2guess = it / itcalc;
    convergence = fabs(it-itcalc);
  } /* end of First While  */

 iitcalc = 2*i2+i+3*i3+hoi+oi+2*i2oh+2*i2o+h2oi;

k3  = i3 / i2 / i;
k2  = hoi * i * h / i2;
k4  = oi / hoi * h;
k6  = i2oh * oh / oi / i;
k7  = i2o / oi / i;

k8  =hoi * h / h2oi;
i2  = i2 * 126900.0 * 2.0;
i  = i * 126900.0;
i3  = i3 * 126900.0 * 3.0;
hoi = 126900.0 * hoi;

oi  = oi * 126900.0;
i2oh = i2oh * 2.0 * 126900.0;
i2o  = i2o * 2.0 * 126900.0;
h2oi = h2oi * 126900.0;

it  = it * 126900.0;
itcalc = itcalc * 126900.0;
iit = iit * 126900.0;
iitcalc = iitcalc * 126900.0;

printf("i2 = %g\n",i2);
printf("i-= %g\n",i);
printf("i3 = %g",i3);  printf("     k3= %g\n",k3);
printf("hoi = %g",hoi);  printf("     k2= %g\n",k2);
printf("oi- = %g",oi);  printf("     k4= %g\n",k4);

printf("i2oh- = %g",i2oh); printf("      k6= %g\n",k6);
printf("i2o-- = %g",i2o); printf("       k7= %g\n",k7);
printf("h2oi+ = %g",h2oi);printf("       k8= %g\n",k8);
printf(" init ph = %g\n",dh);
printf("itcalc = %g\n",itcalc);
```

```
            printf("itotal = %g\n",it);
            printf("iit = %g\n",iit);
            printf("iitcalc = %g\n",iitcalc);
            printf("iteration %d\n",j);
            printf("convergence %g\n",convergence);
            fprintf(file_variable,"%g %g %g %g %g %g %g %g
                    %g\n",ph,i2,i,i3,hoi,oi,i2oh,i2o,h2oi);

            it  = it / 126900.0;
            iit = iit/126900.0;
            iitcalc = iitcalc/126900.0;
            itcalc = itcalc/126900.0;
            printf("pH = %g\n",ph);

            k++;
      }     /* end of for k loop */

      fclose(file_variable);
      m++;
      }   /*  end of for m loop */

  } /*  of main */
```

```
/*      Program: IKINETIC

                  Author: James E. Atwater      */


        Clear;
        ph = 7.0
        h = 10^(-ph)
        initi2 = 4.0
        initi2 = initi2/126900.0 / 2.0
        initi = 0.0

        initi = initi/126900.0
        oi = 0.0
        endt = 10000.0
        k1 = 3.0
        kb1 = 4.4 10^12

        k20 = 250.0
        k21 = 120.0
        kb2 = 3.0 10^8
        filename = "b:p7i2x4.m"


        NDSolve[{i2'[t] == -3.0 i2[t] + 4.4 10^12 i[t] hoi[t]  h,
                i'[t] == 3.0 i2[t] - 4.4 10^12 i[t] hoi[t] h + 2.0/3.0 250.0 hoi[t]^2
                 - 2.0 3.0 10^8 io3[t] i[t]^2 h^2,
                hoi'[t] == 3.0 i2[t] - 4.4 10^12 i[t] hoi[t]  h

                - 250.0 hoi[t]^2 - 120.0 hoi[t] oi + 3.0 3.0 10^8 io3[t] i[t]^2 h^2,
                io3'[t] == 1.0/3.0 250.0 hoi[t]^2 + 1.0/3.0 120.0 hoi[t] oi
                - 3.0 10^8 io3[t] i[t]^2 h^2 ,i2[0] == initi2, i[0] == initi,
                hoi[0] == 0.0, io3[0]== 0.0 },{i2,i,hoi,io3},{t,0,endt}]

        a = Table[{t, Evaluate[i2[t] /. %],Evaluate[i[t] /. %],
                Evaluate[hoi[t] /. %], Evaluate[io3[t] /. %]},
                 {t, 0, endt, endt/50}]

        outfil = OpenWrite[filename]
        index = 0

        While[index <=50,{
                index = index + 1;
                t = a[[index,1]];
```

```
          i2 = a[[index,2]] 126900.0 2.0;


          i = a[[index,3]] 126900.0;
          hoi = a[[index,4]] 126900.0;
          io3 = a[[index,5]] 126900.0;
          Write[outfil,CForm[t],CForm[i2],CForm[i],CForm[hoi],CForm[io3]];
          }]

OutputStream["b:p4i2x2l.m", 3]
Close[filename]
```

```
/*      Program: I2OKINHC                    */
/*      Author: James E. Atwater             */



        Clear[]
        filename ="b:i2_hc_2o.10f"
        outfil = OpenWrite[filename]

        ihco2h = 0.010/46.2
        ii2 = 0.004/253.8
        delta = N[ii2-ihco2h]
        k = 0.37
        t = 0

        index = 0

        While[index<=100,{
                t= 20 index;
                ratio = N[ii2/ihco2h Exp[delta k t] ];
                a = NSolve[{i2 == hco2h ratio,
                        hco2h - i2 + ii2 == ihco2h},{i2, hco2h}];
                I2 = i2 /. a[[1,1]];

                I2 = I2*253800;
                Print[I2];
                Write[outfil,,CForm[t],,CForm[I2]];
                index = index + 1;

        }]
Close[filename]
```

```
'Program : BOOTRMCV.BAS
' Author: Richard R. Wheeler, Jr.
'
120 ' I2 DATA MUST BE VALID THE FIRST TIME THROUGH AT STARTUP,
OTHERWISE
125 ' DUE TO 'DATA VALIDITY CHECKING' THE NEW I2 VALUE MAY
ALWAYS BE SET
130 ' TO THE BAD ORIGINAL I2 VALUE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'
'
'
140 CONFIG KEYPAD$ 20,16,66
142 CONFIG DISPLAY &40,6,0
144 CLEAR COM$ 2
146 CLEAR COM$ 1
148 CONFIG COM$ 2,26,6,0,0
150 CONFIG BAUD 2,6,4,0
152 CONFIG COM$ 1,13,0,0,0
154 CONFIG BAUD 1,6,4,0
156 '
158 '
'
167 'Initiallize the keys to specific ASCII characters. <ENTER> ,ASCII 13 must
168 'be poked in seperately.  the keyboard 'D' is defined to be '-'.
170 FOR X=0 TO 15
172 READ A$
174 POKE SYS(12)+X,ASC(A$)
176 NEXT X
178 DATA 1,2,3,A,4,5,6,B,7,8,9,C,'<ENTER>',0,.,-
180 POKE SYS(12)+12,13

'initialization section.........
'
182 DIM a(3)
184 N6 = (8 * 8001)
186 VSFLAG = PEEK(N6,1)
188 IF VSFLAG = 2 THEN GOSUB ..RETREIVE_VARIABLES
'
190 'the following variables are reset whether or not a power outage occurs.
'
192 'Initial flag for data validity checks in ..com.
194 EE = 0
196 RGEN$= "REGENERATE"
198 WASH$= "...washout"
200 SI$= ".....SuPeRiOdInAtIoN"
```

```
202 XZTIME= 30.0
204 RMAXTIME= 600
206 N5= 1
208 DIM B$(10)
'
210 DIM CNTRDAY(3)
215 DIM LDAT$(3)
220 DIM T0(3)
225 'these above arrays are initiallized later in the program.
230 '
232 BDFLAG$ = "GOOD_DATA"
'
'
240 IF VSFLAG = 2 THEN GOTO ..RECOVER
242 'otherwise the default variable values will be assigned as below.
244 '
'
250 'Initial values. Unless otherwise needed, pi is the debugging default.
'
255 PFLAG = 5
260 POKE N6+480,PFLAG,1
'
265 LRTHRU = 0
267 POKE! N6+570,LRTHRU,1
'
270 ECFLAG$ = "DISABLED"
272 POKE$ (N6+600),ECFLAG$,1
290 '
300 I2$= "%%%%%"
303 POKE$ (N6+60),I2$,1
305 'Flowrate in liters per minute.
310 Q= 0.120
312 POKE! (N6+150),Q,1
315 'Crystal column iodine concentration in grams per liter.
320 CRYSI2= 0.240
323 POKE! N6+180,CRYSI2,1
325 '
330 '
345 'Cumulative washout Iodine mass in grams.
350 MASSI2 = 0.0
355 POKE! (N6+90),MASSI2,1
360 '
365 'the com2 flag can be set to 'E'nable or 'D'isable com2 checking.
370 FLAGCOM2$= "E"
373 POKE$ N6+210,FLAGCOM2$,1
```

```
395 'Cumulative washout volume(THROUGHPUT) to pass through MCV in liters.
400 THRU= 0.0
405 POKE! (N6+120),THRU,1
410 N= 0
413 POKE! N6+360,N,1
415 'the upperlimit is a safety check that will terminate a regeneration.
420 UPLIMI2= 7.50
423 POKE! N6+270,UPLIMI2,1
425 'the regeneration initiation trigger is the variable BOTLIMI2.
430 BOTLIMI2= 2.00
435 POKE! N6+300,BOTLIMI2,1
440 'default MCV Bed Volume in cc.
442 BV = 100
444 POKE! (N6+630),BV,1
450 '
504 ZXTIME = 300
510 POKE! N6+240,ZXTIME,1
519 'initiallize the recalibration array; DIM sets all to zero(DONE EARLIER).
525 a(1)= 1
527 FOR N7 = 0 TO 2
529 POKE! N6+390+(N7*30),a(N7),1
531 NEXT N7
535 'the iodine data is collected from COM2 about every V0TIME minutes.
540 V0TIME= 15
543 POKE! N6+330,V0TIME,1
545 'the following arrays are used in the subroutine ..TOTAL_ELLAPSED_TIME
550 'to calculate ellapsed time, even if it transcends days.  So far only
551 'three functions have been identified as needing this abillity:
552 '(0)the check ..com2 timeout, (1)DLTATHRU and MASSI2 calculation,
553 ' and (2)the regen. timeup.
555 'DIM CNTRDAY(3)
560 'DIM LDAT$(3)
565 'DIM T0(3)
'
575 'make sure all relays are off.
576 '0-main regen. relay;1-teflon regen. relay;2-superiodination relay.
577 BIT 65,0,0
578 BIT 65,1,0
579 BIT 65,2,0
580 R$ = WASH$
590 POKE$ (N6+30),R$,1
'
'
600 'initiallize times.
605 GOSUB ..TIMEINMINUTES
```

```
610 T= T
615 '
'
620 RW$ = DATE$(0)
622 FOR G = 0 TO 2
624 CNTRDAY(G) = 0
626 LDAT$(G) = RW$
628 T0(G) = T
630 NEXT G
632 '
'
650 VSFLAG = 2
660 POKE N6,VSFLAG,1
665 GOTO ..START1
670 ..RECOVER

675 'This is an important part of the power outage recovery code.
'
680 IF R$ <> WASH$ THEN GOTO ..SKIP1
' merely continue with washout and sample after 15 minutes(thus reset cntrs).
682 GOSUB ..TIMEINMINUTES
683 RW$ = DATE$(0)
684 FOR G = 0 TO 2
685 CNTRDAY(G) = 0
686 LDAT$(G) = RW$
687 T0(G) = T
688 NEXT G
690 GOTO ..START1
695 '
'
700 ..SKIP1
704 IF R$ <> SI$ THEN GOTO ..SKIP2
' merely go to washout and sample after 15 minutes. SUPERIODINATION ENDED.
706 GOSUB ..TIMEINMINUTES
708 RW$ = DATE$(0)
709 FOR G = 0 TO 2
710 CNTRDAY(G) = 0
711 LDAT$(G) = RW$
712 T0(G) = T
713 NEXT G
714 R$ = WASH$
715 POKE$ (N6+30),R$,1
716 GOTO ..START1
718 '
720 ..SKIP2
```

```
725 IF R$ <> RGEN$ THEN GOTO ..SKIP3
730 'first recover the last recorded total ellapsed regeneration time.
732 T1 = PEEK!(N6+510,1)
734 'calculate the remaining regeneration time, set duration equal to this
736 'time and initiate regeneration.
738 ZXTIME = ZXTIME - T1
739 'above duration saved in battery backed RAM below(line 788) after relay on.
740 'do not reinitiallize the displaced iodine mass to zero.
742 'MASSI2= 0.0
744 '
750 'initiallize all times and counters.
752 GOSUB ..TIMEINMINUTES
754 '
756 RW$ = DATE$(0)
758 FOR G = 0 TO 2
760 CNTRDAY(G) = 0
762 LDAT$(G) = RW$
764 T0(G) = T
766 NEXT G
'
770 'turn relays on.
771 BIT 65,0,1
772 BIT 65,1,1
774 R$ = RGEN$
776 POKE$ (N6+30),R$,1
'
782 'set reg. sample time to 5min.; first save current sample period to V3TIME.
784 V3TIME = V0TIME
786 V0TIME = 5
'
787 'Save the new regen. duration in battery backed RAM.
788 POKE! N6+240,ZXTIME,1
796 ..SKIP3
800 ..START1
'
802 'This section was included for 'second hand' calculated variables.
803 'Note that the program start-up always comes through here last!!!
'
804 'Minimum THRUput(1L/cc*BED VOLUME) must flow through RMCV before a
second
805 'auto-regen. is allowed.
806 MTHRU = (1*BV)
810 'The maximum regeneration time allowed is 200% of the nominal regen. time
811 'at that particular MCV residence time.
812 'The nominal regeneration time for a flow rate of 120cc/min and an MCV
```

```
813 'bed volume of 100cc is roughly 300 minutes.
815 RMAXTIME = ((300)/(100/120))*(BV/(Q*1000))*2.0
817 'The minimum regen. time allowed is 10% of the nominal regen. time.
818 XZTIME = ((300)/(100/120))*(BV/(Q*1000))*.1
819 'Therefore for 100cc bed and 120cc/min, RMAXTIME=600min and
XZTIME=30min.
'
820 ..START
823 'the main program flow begins and comes back through here.
826 'DISPLAY (0,0) "                    ";
829 DISPLAY (0,0) "UMPQUA RESEARCH RMCV";
832 'DISPLAY (1,0) "                    ";
835 DISPLAY (1,0) " ";DATE$(0);"  ";TIME$(0);" ";
838 'DISPLAY (2,0) "                    ";
840 DISPLAY (2,0) "  status= ";R$;
842 IF FLAGCOM2$ = "D" THEN GOTO 870
844 IF BDFLAG$ = "GOOD_DATA" THEN GOTO 858
846 DISPLAY (3,0) "RECEIV'n BAD I2 DATA";
850 GOTO ..MARK1
856 'DISPLAY (3,0) "Last [I2]=          ";
858 DISPLAY (3,0) "Last [I2]=";I2$;" ppm";
860 GOTO ..MARK1
870 DISPLAY (3,0) "COM2 read 'D'isabled";
880 ..MARK1

900 'Program spends most of its time in this section waiting for [I2] data
910 'and/or a keyboard interrupt.
915 ON KEYPAD$ GOSUB ..KEYPAD
'
'
950 ..superloop
952 ON KEYPAD$ GOSUB ..KEYPAD
954 'Superiodination timing loop/ manual termination.
956 IF R$ <> SI$ THEN GOTO ..leave
960 GOSUB ..TIMEINMINUTES
965 IF T >= SUTIME THEN GOSUB ..SUPERIODINATION
967 IF R$ = WASH$ THEN GOTO ..leave
'
968 FOR X = 1 TO 1000
969 NEXT X
'
970 'display manual termination message.
975 DISPLAY (0,0) "Press any key to    ";
980 DISPLAY (1,0) "terminate           ";
985 DISPLAY (2,0) "      SUPER-        ";
```

```
990 DISPLAY (3,0) "    IODINATION    ";
'
995 GOTO ..superloop
997 ..leave
'
1019 'if the COM2 flag has been 'D'isabled, then do not read [I2] from COM2.
1020 IF FLAGCOM2$ = "D" THEN GOTO ..label1
1025 ' the index P is used in the following subroutine.
1030 P = 0
1035 GOSUB ..TOTAL_ELLAPSED_TIME
1040 TT = TT
1050 CDAT$=CDAT$
'
1055 'iodine value collected from COM2 with period >= V0TIME(15min. initially).
1060 IF TT < V0TIME THEN GOTO ..label1
1065 '
1070 'T0(0) is the last time that COM2 was read.
1075 T0(0) = T
1080 'LDAT$(0) is the last date that COM2 was read.
1085 LDAT$(0) = CDAT$
1087 'initiallize the day counter.
1088 CNTRDAY(0) = 0
1090 'the variables T and CDAT$ are from ..TOTAL_ELLAPSED_TIME.
'
1095 IF R$ = SI$ THEN GOTO ..superloop
1096 GOSUB ..COM2
'
1100 ..label1
'
1235 '
1240 'see if regeneration is in process.
1245 IF R$ <> RGEN$ THEN GOTO ..label2
'
1250 GOSUB ..CHECK_REG_TIME_UP
1255 ..label2
'
1260 'This section of code included in the main program exists so that
1265 'variables may be written frequently to battery backed RAM.
1270 'This section is only to be entered every 100 times through the main loop
1275 'with the exception of the 'total ellapsed time during regen.' variable.
1280 IF R$ <> RGEN$ THEN GOTO ..label3
1285 'Ellapsed regeneration time needs to be written frequently so that a
1290 'power down time that occurs during regeneration can be recovered from.
1292 'TT comes from line 1250 (GOSUB ..CHECK_REG_TIME_UP).
1295 POKE! N6+510,TT,1
```

```
1300 ..label3
'
1340 IF N5 < 100 THEN GOTO ..label4
1345 '
1350 'Variables will be saved first, followed by current-time/up-time pairs.
1355 'The last 190 (8-byte) blocks have been reserved for variables.
1356 'This corresponds to locations (8*8001) through (8*8190)= 65,520.
1357 '30 bytes is more than enough memory for each variable(8*190 =1520 bytes),
1358 'therefore so as to minimize the chance of overwriting, I will simply
1359 'place the variables sequentially in 30 byte sections instead of
1360 'placing the first character of the (i+1)th variable directly after the
1361 'last character of the ith variable.
'
1590 N5 = 1
1600 ..label4
2000 IF R$ = SI$ THEN GOTO ..superloop
2010 GOTO ..START

2500 END
'
'
2700 ..COM2
'
2703 IF R$ = SI$ THEN GOTO ..RETCOM
'2705 'disable interrupts.
'2710 'the ITROFF placed here hangs program up if no valid data in com2 buffer.
2715 'clear buffer and receive fresh(current) data.
2720 CLEAR COM$ 2
2725 CONFIG COM$ 2,26,6,0,0
2730 'wait 1 sec. for buffer to fill.
2732 DELAY 1
2735 '
2740 II2$= "99"
'
'
2742 V = 1
2744 '
2745 'search for <cntrl s> character(ASCII 19). Characters erased from
2746 'buffer as read. ASTRO mon. data format:
<cntrls>x.xxx<CR><cntrls>x.xxx<CR><cntrls>x.xxx<CR> ...
2750 ..SEARCH
2755 S$= INKEY$(2)
'the below three lines were added so that the program would not hang up.
2760 ON KEYPAD$ GOSUB ..KEYPAD
2770 DISPLAY (3,0) "Wait'n for COM2 data";
```

```
2780 IF FLAGCOM2$ = "D" THEN GOTO ..RETCOM
2785 IF R$ = SI$ THEN GOTO ..RETCOM
'
2790 IF R$ <> RGEN$ THEN GOTO 2845
2794 'check reg. time up only every 20th time through loop.
2795 IF V < 20 THEN GOTO 2840
2800 GOSUB ..CHECK_REG_TIME_UP
2810 V = 0
'
2840 V = V + 1
2845 IF S$ = CHR$(19) THEN GOTO ..FOUND
2850 GOTO ..SEARCH
'
2855 ..FOUND
2860 'if <cntrl s> character exists in buffer then keyboard interrupt disabled.
2865 ITROFF
2867 FOR K1 = 0 TO 4
2870 B$(K1)= INKEY$(2)
2875 II2$= II2$ + B$(K1)
2880 NEXT K1
2885 '
2890 'collect the five character number 'x.xxx' from II2$
2895 II2$= MID$(II2$,3,5)
2900 II2= VAL(II2$)
'
2925 'correct the [I2] with the recalibration coefficients.
2930 'the initial values are a(0)=0, a(1)=1, a(2)=0 ; i.e. no correction.
2935 I2= a(0) + a(1)*II2 + a(2)*II2*II2
2940 I2$= STR$(I2)
'
2943 IF I2 <= 9.9 AND I2 >= 0.1 THEN GOTO ..HOP2
2944 IF EE = 0 THEN BDFLAG$ = "BAD_DATA"
2945 IF EE = 0 THEN GOTO ..RETCOM
2947 ..HOP2
2948 BDFLAG$ = "GOOD_DATA"
'
2950 'this TT may not be the same as for ..com2 timeout if nothing in buffer.
2955 P = 1
2960 GOSUB ..TOTAL_ELLAPSED_TIME
2965 'This routine returns the total ellapsed time TT for function P = 1; as
2970 'well as the current date CDAT$ and current time T.
2972 'TT is used in the deltathroughput calculation below.
'
2975 'the variable T0(1) is the last actual sample time.
2976 T0(1)= T
```

```
2977 'the variable LDAT$(1) is the last actual sample date; CDAT$ from
..TOTAL_ELLAPSED_TIME.
2978 LDAT$(1)= CDAT$
2979 'initialize the day counter to zero.
2980 CNTRDAY(1)= 0
'
'Data validity check; If in error then use the last valid I2 value.
'A difference of 1.0ppm should never occur between two successive I2 samples
'during washout; during regeneration this may happen!!
2990 IF I2 > 9.9 OR I2 < 0.1 THEN GOTO ..skip4
3000 '
'
3004 'For program startup and when a power outage/ restart occurs.
3005 IF EE = 0 THEN LSTI2 = I2
3007 EE = 1
3008 '
'
3009 IF R$ = RGEN$ THEN GOTO 3020
3010 IF ABS((I2-LSTI2)) > 1.0 THEN GOTO ..skip4
3020 GOTO ..skip5
'
3030 ..skip4
3032 BDFLAG$ = "BAD_DATA"
3034 GOTO ..RETCOM
'
3040 ..skip5
3043 BDFLAG$ = "GOOD_DATA"
3045 POKE$ (N6+60),I2$,1
'
3050 'TT comes from the function P = 1. See line 2950.
3055 DLTATHRU= Q*(TT)
3060 THRU= THRU + DLTATHRU
3065 POKE! (N6+120),THRU,1
'
3085 MASSI2= MASSI2 + DLTATHRU * (0.5) *(LSTI2 + I2)/1000
3087 POKE! (N6+90),MASSI2,1
3090 'This cumulative iodine mass in grams will be used
3095 'to determine regeneration time.
'
3120 'Keep track of the maximum(peak) [I2] value during washout.
3125 IF R$ = RGEN$ THEN GOTO ..HOP1
3130 IF PFLAG <> 7 THEN PKI2 = I2
3135 IF PFLAG = 7 THEN GOTO 3150
3140 PFLAG = 7
3145 POKE N6+480,PFLAG,1
```

```
3150 IF PKI2 < I2 THEN PKI2 = I2
3155 POKE! N6+540,PKI2,1
3160 ..HOP1

3200 'Store pertinent data in memory(up to 91Kbytes) and upload to
3202 'IBM on com1 whether or not link is actually present.
3204 'If the RMCV is in regeneration then store data as negative numbers.
3210 M1THRU = THRU
3212 M2I2 = I2
3215 IF R$ = WASH$ THEN GOTO 3230
3220 M1THRU = (-1) * THRU
3225 M2I2 = (-1) * I2
'
3230 PRINT #1,M2I2
3235 PRINT #1,M1THRU
'
3240 'Thruput is stored in floating point format in four consecutive locations.
3250 'Iodine concentration is stored in the same manner. The
3260 'Throughput in liters and the [I2] in ppm are thus in eight consec. loc.
3270 'The 0 location in segment 1 has an absolute address of &10000.
3280 'The maximum user RAM address is 65,535(&26BFF). N=0,1,...,8190
3290 'The last 190 blocks have been set aside for battery backed varialbes.

3300 POKE! (N*8),M2I2,1
3310 POKE! ((N*8)+4),M1THRU,1
3320 N= N+1
3330 'When this memory is filled then any new data overwites the oldest.
3340 IF N > 8000 THEN N=0
3350 POKE! N6+360,N,1
'
3440 'Make sure regeneration is on before checking upperlimit.
3445 'At this point in the program there are only two possible
3447 'states for R$ (WASH$ and RGEN$).
3450 IF R$ = WASH$ THEN GOTO ..READY?
3460 IF I2 >= UPLIMI2 THEN GOSUB ..TERMINATE_REGENERATION
3470 GOTO ..RETCOM
'
3480 ..READY?
3482 'See if MCV is ready to regenerate. Trigger regeneration only if
3484 'two consecutive iodine readings are below trigger point.
3486 IF (I2 > BOTLIMI2 OR LSTI2 > BOTLIMI2) THEN GOTO ..RETCOM
'
3490 'If minimum thruput after a regen. has'nt occurred then regen. disallowed.
3495 IF (THRU - LRTHRU) < MTHRU THEN GOTO ..RETCOM
'
```

```
3500 'check to see if Peak Error Correction function is enabled.
3502 IF ECFLAG$ = "ENABLED" THEN GOTO ..HERE1
3504 EC = 0
3506 GOTO ..HERE2
3508 ..HERE1
'
3515 'calculate the peak error correction factor(self regulation function
3516 'that attempts to make the peak value during a washout exactly 4ppm).
3517 'This cumulative mass correction is in grams.  Note that a typical
3518 'amount iodine mass replaced at beginning of a 100cc MCV life is 10,000mg.
3520 EC = (4 - PKI2) * 2.35
3522 'bounds need to be set on the magnitude of the EC parameter(in grams).
3524 IF PKI2 < 2 THEN EC = 4.7
3525 IF PKI2 > 6 THEN EC = -4.7
'
3527 'determine regeneration duration. If the calculated time is below a
3528 'reasonable range then regenerate for the minimum time.
3529 'if calc. time is greater than the upper sensibillity check, then use
3530 'the duration of the last regeneration and allow regeneration to occur.
'
3535 ..HERE2
3540 AZXTIME = (MASSI2 + EC)/(CRYSI2 * Q)
3545 IF AZXTIME <= XZTIME THEN ZXTIME = XZTIME
3550 IF AZXTIME >= RMAXTIME THEN ZXTIME = ZXTIME
3553 IF AZXTIME > XZTIME AND AZXTIME < RMAXTIME THEN ZXTIME =
AZXTIME
3556 POKE! N6+240,ZXTIME,1
3558 '
3560 'reinitiallize the displaced iodine mass to zero.
3565 MASSI2= 0.0
3570 POKE! (N6+90),MASSI2,1
'
3615 GOSUB ..TIMEINMINUTES
3620 T = T
3622 '
3624 'T0(2) is the regeneration beginning time. This should be
3626 T0(2) = T
3628 'LDAT$(2) is the start of regen. date.
3630 LDAT$(2) = DATE$(0)
3632 'Initialize day counter to zero.
3633 CNTRDAY(2) = 0
'
3635 'turn relays on.
3640 BIT 65,0,1
3642 BIT 65,1,1
```

```
3645 R$ = RGEN$
3647 POKE$ (N6+30),R$,1
'
3650 'set reg. sample time to 5min.; first save current sample period to V3TIME.
3653 V3TIME = V0TIME
3655 V0TIME = 5
'
3710 GOTO ..RETCOM
'
3900 ..RETCOM
3901 DISPLAY (3,0) "Last [I2]=          ";
3902 IF BDFLAG$ = "GOOD_DATA" THEN LSTI2 = I2
3903 're-enable interrupts before returning.
3905 ITRON
3910 RETURN
'
'
'
4000 ..KEYPAD
'
4004 IF R$ = SI$ THEN GOTO ..SI
4006 'Change display configuration so that cursor displayed and char's echoed.
4008 CONFIG DISPLAY &40,6,1
'
4009 BB$ = KEYPAD$(0)
4010 ..HERE
4015 BB$ = "%"
4020 CC$ = ""
'
4030 DISPLAY (0,0) "    <select option:";
4032 DISPLAY (1,0) "'A*' TO RECALIBRATE ";
4034 DISPLAY (2,0) "'B*' CHANGE PARAMTRS";
4036 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
4040 ..AGAIN1
4041 ITRON
4042 DELAY .08
4043 AGFLAG$ = "ANYTHING"
4044 ON KEYPAD$ GOSUB ..WHAT?
4045 FOR U=1 TO 10000
4046     IF AGFLAG$ = "KEY" THEN GOTO ..CONT1
4047 NEXT U
4048 GOTO ..MARK2
4049 ..WHAT?
4050      AA$ = KEYPAD$(0)
```

```
4051      AGFLAG$ = "KEY"
4052 RETURN
4053 ..CONT1
4054 ITROFF
4055 IF AA$ = CHR$(13) THEN GOTO ..OKAY1
4056 BB$ = BB$ + AA$
4057 CC$ = MID$(BB$,2)
4058 DISPLAY (0,1) "    ";
4059 DISPLAY (0,1) CC$;
4060 ITRON
4061 GOTO ..AGAIN1
4062 ..OKAY1
4065 'check to see if CC$ is valid.
'
4072 IF CC$ = "A" THEN GOSUB ..RECALIBRATE
4074 IF CC$ = "A" THEN GOTO ..MARK2
'
4076 IF CC$ = "B" THEN GOSUB ..CHANGE_PARAMETERS
4078 IF CC$ = "B" THEN GOTO ..MARK2
'
4080 IF CC$ <> "C" THEN GOTO 4010
4090 DISPLAY (0,0) "    <select option:";
4100 DISPLAY (1,0) "'A*' DUMP DATA COM1 ";
4110 DISPLAY (2,0) "'B*' INITIATE REGEN.";
4120 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
4150 INPUT KEYPAD$ 10,A$
'
4160 IF A$ = "A" THEN GOSUB ..DOWNLOAD_DATA
4165 IF A$ = "A" THEN GOTO ..MARK2
4170 IF A$ = "B" THEN GOSUB ..INITIATE_REGENERATION
4175 IF A$ = "B" THEN GOTO ..MARK2
'
4180 IF A$ <> "C" THEN GOTO 4090
4190 DISPLAY (0,0) "    <select option:";
4200 DISPLAY (1,0) "'A*' INITIALIZE RAM ";
4210 DISPLAY (2,0) "'B*' VIEW REGEN TIME";
4220 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
4250 INPUT KEYPAD$ 10,A$
'
4260 IF A$ = "A" THEN GOSUB ..INITIALIZE_RAM
4265 IF A$ = "A" THEN GOTO ..MARK2
4270 IF A$ = "B" THEN GOSUB ..VIEW_REGEN_TIME
4275 IF A$ = "B" THEN GOTO ..MARK2
```

```
'
4280 IF A$ <> "C" THEN GOTO 4190
'
4290 DISPLAY (0,0) "    <select option:";
4300 DISPLAY (1,0) "'A*' END RGENERATION";
4310 DISPLAY (2,0) "'B*' check COM2 y/n ";
4320 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
4350 INPUT KEYPAD$ 10,A$
'
'
4360 IF A$ = "A" THEN GOTO 4363
4362 GOTO 4370
4363 IF R$ <> RGEN$ THEN GOTO 4370
4364 GOSUB ..TERMINATE_REGENERATION
4365 IF A$ = "A" THEN GOTO ..MARK2
4370 IF A$ = "B" THEN GOSUB ..CHECK_COM2_?
4375 IF A$ = "B" THEN GOTO ..MARK2
'
4380 IF A$ <> "C" THEN GOTO 4290
'
4390 DISPLAY (0,0) "    <select option:";
4400 DISPLAY (1,0) "'A*' PEC enabled y/n";
4410 DISPLAY (2,0) "'B*' ..unused      ";
4420 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
4450 INPUT KEYPAD$ 10,A$
'
4460 IF A$ = "A" THEN GOSUB ..PEC
4465 IF A$ = "A" THEN GOTO ..MARK2
'
4480 IF A$ <> "C" THEN GOTO 4390
'
4490 DISPLAY (0,0) "    <select option:";
4500 DISPLAY (1,0) "'A*' TO RETURN CNTRL";
4510 DISPLAY (2,0) "'#*' SUPERIODINATION";
4520 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
4550 INPUT KEYPAD$ 10,A$
4560 IF A$ = "." THEN GOSUB ..SUPERIODINATION
4565 IF A$ = "." THEN GOTO ..MARK2
4570 IF A$ = "A" THEN GOTO ..MARK2
'
'
4580 IF A$ <> "C" THEN GOTO 4490
```

```
4590 DISPLAY (0,0) "     <select option:";
4600 DISPLAY (1,0) "'A*' TO RETURN CNTRL";
4610 DISPLAY (2,0) "'#*' TERMINATE PROG.";
4620 DISPLAY (3,0) "     OPERATION.     ";
'
4650 INPUT KEYPAD$ 10,A$
4660 IF A$ = "." THEN GOSUB ..QUIT
4665 IF A$ = "." THEN GOTO 20000
4670 IF A$ = "A" THEN GOTO ..MARK2
'
4680 GOTO 4590
4830 GOTO 4860
4840 ..SI
4850 GOSUB ..SUPERIODINATION
4855 GOTO ..MARK2
4860 '
'
4900 ..MARK2
4903 ITRON
4905 CONFIG DISPLAY &40,6,0
4910 RETURN
'
5500 ..TIMEINMINUTES
5510 ITROFF
5520 T$= TIME$(0)
5530 HH$= MID$(T$,1,2)
5540 HH= VAL(HH$)
5550 MM$= MID$(T$,4,2)
5560 MM= VAL(MM$)
5570 SS$= MID$(T$,7,2)
5580 SS= VAL(SS$)
5590 T= 60*HH+MM+SS/60
5600 '
5610 ITRON
5615 RETURN
'
'
'
6000 ..TOTAL_ELLAPSED_TIME
'
6020 ITROFF
'
6030 CDAT$ = DATE$(0)
'
6040 IF CDAT$ = LDAT$(P) THEN GOTO 6080
```

```
6050 CNTRDAY(P) = CNTRDAY(P) + 1
6060 LDAT$(P) = CDAT$
6070 '
'
6080 GOSUB ..TIMEINMINUTES
6090 'T=T
6100 '
'
6110 IF CNTRDAY(P) = 0 THEN GOTO 6170
'
6120 W1 = 24*60*(CNTRDAY(P) - 1)
6130 W2 = 24*60 -T0(P)
6140 TT = W1 + W2 + T
6150 '
'
6170 IF CNTRDAY(P) <> 0 THEN GOTO 6200
6180 TT = T - T0(P)
6200 ITRON
6205 RETURN
''
6500 ..TERMINATE_REGENERATION
6510 '
6515 ITROFF
''
6550 'Turn relays off, and set status to ...WASHOUT.
'
6615 BIT 65,0,0
6617 BIT 65,1,0
6620 R$ = WASH$
6625 POKE$ (N6+30),R$,1
'
'set sample time back to what is was before regeneration started.
6630 V0TIME = V3TIME
6640 A$= "A"
6645 'set the washout peak detection flag to indicate that the next sample is
6646 'the first sample during the next washout. See line 3120.
6650 PFLAG = 5
6655 POKE N6+480,PFLAG,1
'
'Last Regeneration THRUput saved.
6660 LRTHRU = THRU
6665 POKE! N6+570,LRTHRU,1
'
6680 ITRON
6685 RETURN
```

```
7000 ..RECALIBRATE
'
'
7050 DISPLAY (0,0) "ENTER COEFFICIENTS  ";
7060 DISPLAY (1,0) "FOR A QUADRATIC    ";
7070 DISPLAY (2,0) "CORRECTION:        ";
7075 DISPLAY (3,0) "                   ";
7080 DISPLAY (3,0) "'C*' to cont... ";
'
7090 INPUT KEYPAD$ 10,A$
'
7100 IF A$ <> "C" THEN GOTO 7500
'
7110 DISPLAY (0,0) " a+b[I2]+c[I2]^^2   ";
7115 DISPLAY (1,0) "                    ";
7120 DISPLAY (1,0) "a=";a(0);"b=";a(1);"c=";a(2);
7125 DISPLAY (2,0) "                    ";
7130 DISPLAY (2,0) " '#'='.'   'D'='-' ";
7135 DISPLAY (3,0) "                    ";
7140 DISPLAY (3,0) "enter 'a*'>";
'
7145 INPUT KEYPAD$ 10,AA$
'
7150 AA= VAL(AA$)
7155 'Check to see that the value is a number in the proper range.
7160 IF AA <= 9.9 THEN GOTO 7170
7165 GOTO 7110
7170 IF AA > -9.9 THEN GOTO 7180
7175 GOTO 7110
7180 a(0) = AA
'
7210 DISPLAY (0,0) " a+b[I2]+c[I2]^^2   ";
7215 DISPLAY (1,0) "                    ";
7220 DISPLAY (1,0) "a=";a(0);"b=";a(1);"c=";a(2);
7225 DISPLAY (2,0) "                    ";
7230 DISPLAY (2,0) " '#'='.'   'D'='-' ";
7235 DISPLAY (3,0) "                    ";
7240 DISPLAY (3,0) "enter 'b*'>";
'
7245 INPUT KEYPAD$ 10,AA$
'
7250 AA= VAL(AA$)
7255 'Check to see that the value is a number in the proper range.
7260 IF AA <= 9.9 THEN GOTO 7270
7265 GOTO 7210
```

```
7270 IF AA > -9.9 THEN GOTO 7280
7275 GOTO 7210
7280 a(1) = AA
'
7310 DISPLAY (0,0) " a+b[I2]+c[I2]^^2   ";
7315 DISPLAY (1,0) "                    ";
7320 DISPLAY (1,0) "a=";a(0);"b=";a(1);"c=";a(2);
7325 DISPLAY (2,0) "                    ";
7330 DISPLAY (2,0) " '#'='.'   'D'='-' ";
7335 DISPLAY (3,0) "                    ";
7340 DISPLAY (3,0) "enter 'c*'>";
'
7345 INPUT KEYPAD$ 10,AA$
'
7350 AA= VAL(AA$)
7355 'Check to see that the value is a number in the proper range.
7360 IF AA <= 9.9 THEN GOTO 7370
7365 GOTO 7310
7370 IF AA > -9.9 THEN GOTO 7380
7375 GOTO 7310
7380 a(2) = AA
'
7500 CC$ = "A"
7505 FOR N7 = 0 TO 2
7510 POKE! N6+390+(N7*30),a(N7),1
7515 NEXT N7   } 7517  EE = 0
7520 RETURN
'
7600 ..CHECK_COM2_?
'
7620 DISPLAY (0,0) "The reading of COM2,";
7630 DISPLAY (1,0) "for [I2] data, can  ";
7640 DISPLAY (2,0) "be 'disabled'= 'C*'. ";
7650 DISPLAY (3,0) "                    ";
7660 DISPLAY (3,0) "or 'enabled'= 'A*'";
'
7680 INPUT KEYPAD$ 10,A$
'
7690 IF A$ = "A" THEN FLAGCOM2$ = "E"
7700 IF A$ = "A" THEN GOTO 7770
7710 IF A$ = "C" THEN FLAGCOM2$ = "D"
7720 IF A$ = "C" THEN GOTO 7770
7740 GOTO 7600
'
7770 A$ = "B"
```

```
7775 POKE$ N6+210,FLAGCOM2$,1
7780 RETURN
'
'
7800 ..PEC
'
7805 DISPLAY (0,0) "The Peak Error Cor- ";
7807 DISPLAY (1,0) "rection function is ";
7809 DISPLAY (2,0) "                    ";
7811 DISPLAY (2,0) "currently ";ECFLAG$;
7813 DISPLAY (3,0) "DEFAULT IS: DISABLED";
7815 DELAY 5
'
7820 DISPLAY (0,0) "The Peak Error Cor- ";
7830 DISPLAY (1,0) "rection function can";
7840 DISPLAY (2,0) "be 'disabled'='C*'  ";
7850 DISPLAY (3,0) "                    ";
7860 DISPLAY (3,0) "or 'enabled'='A*'";
'
7880 INPUT KEYPAD$ 10,A$
'
7890 IF A$ = "C" THEN ECFLAG$ = "DISABLED"
7900 IF A$ = "C" THEN GOTO 7970
7910 IF A$ = "A" THEN ECFLAG$ = "ENABLED"
7915 IF A$ <> "A" THEN GOTO ..PEC
'
7920 DISPLAY (0,0) "Input greatest [I2] ";
7922 DISPLAY (1,0) "observed thusfar in ";
7924 DISPLAY (2,0) "the current washout:";
7926 DISPLAY (3,0) "                    ";
7928 DISPLAY (3,0) "x.xxx* (ppm)>";
'
7930 INPUT KEYPAD$ 10,AA$
'
7940 AA= VAL(AA$)
7942 'Check to see that the value is a number in the proper range.
7944 IF AA <= UPLIMI2 THEN GOTO 7948
7946 GOTO 7920
7948 IF AA > BOTLIMI2 THEN GOTO 7952
7950 GOTO 7920
7952 PKI2 = AA
7954 POKE! N6+540,PKI2,1
'
7960 GOTO 7970
'
```

```
'
'
7970 A$ = "A"
7975 POKE$ N6+600,ECFLAG$,1
7980 RETURN
'

'
8000 ..DOWNLOAD_DATA
'
'
8140 'Thruput is stored in floating point format in four consecutive locations.
8150 'Iodine concentration is stored in the same manner. The
8160 'Throughput in liters and the [I2] in ppm are thus in eight consec. loc.
8170 'The 0 location in segment 1 has an absolute address of &10000.
8175 'The maximum user RAM address is 65,535(&26BFF). N=0,1,...,8190
8180 'The last 190 blocks have been set aside for battery backed varialbes.
8185 N3= 0
8187 N4= 1
8190 FOR N2 = 0 TO 8000
8200 IODINE = PEEK!((N2*8),1)
8210 LITERS = PEEK!(((N2*8)+4),1)
8220 'Download to IBM on com1 whether or not link is actually present.
8230 PRINT #1,IODINE
8240 PRINT #1,LITERS
8250 N3 = N3+1
8260 IF N3 < 100 THEN GOTO 8350
8270 DISPLAY (0,0) "     ";TIME$(0);"     ";
8275 DISPLAY (1,0) "                    ";
8280 DISPLAY (2,0) "downloading >COM1...";
8285 DISPLAY (3,0) "                    ";
8290 DISPLAY (3,0) " ";N2+1;" of ";" 8001 ";
8300 N3= 0
8310 '
8350 IF N4 < 1000 THEN GOTO 8450
8360 DISPLAY (0,0) " waiting for target ";
8370 DISPLAY (1,0) "computer to write   ";
8380 DISPLAY (2,0) "data to harddisk    ";
8390 DISPLAY (3,0) "and/or floppy drive.";
8395 'will delay for 10 seconds.
8400 DELAY 10
8410 N4 = 0
8420 '
8440 '
8450 N4 = N4+1
```

```
8460 NEXT N2
'
8500 RETURN
'
'
8600 ..INITIATE_REGENERATION
'
8603 IF R$ <> WASH$ THEN GOTO 8900
'
'
8610 DISPLAY (0,0) "Enter regeneration  ";
8615 DISPLAY (1,0) "time in minutes:    ";
8620 DISPLAY (2,0) "time> xxxx.xx       ";
8625 DISPLAY (3,0) "                    ";
8630 DISPLAY (3,0) "time> ";
'
8635 INPUT KEYPAD$ 10,AA$
'
8640 AA= VAL(AA$)
8645 'Check to see that the value is a number in the proper range.
8650 IF AA <= 1000 THEN GOTO 8660
8655 GOTO 8610
8660 IF AA > 0 THEN GOTO 8670
8665 GOTO 8610
8670 ZXTIME = AA
8672 'see line 8850.
'
'
8705 'This subroutine is used only to
8707 'FORCE the RMCV into regeneration, the duration 'ZXTIME' will be set
8710 'manually through keyboard input.
8713 'Normal regen. is entered elsewhere in the prog.
8715 GOSUB ..TIMEINMINUTES
8720 'T = T
8740 '
8750 'T0(2) is the regeneration beginning time. This should be
8760 T0(2) = T
8770 'LDAT$(2) is the start of regen. date.
8780 LDAT$(2) = DATE$(0)
8790 'initialize the day counter to zero.
8800 CNTRDAY(2)= 0
'
8825 'turn relayS on.
8830 BIT 65,0,1
8835 BIT 65,1,1
```

```
8840 R$ = RGEN$
8845 POKE$ (N6+30),R$,1
8850 'regen. duration saved after the relay is actually turned on.
8855 POKE! N6+240,ZXTIME,1
'
8860 'set reg. sample time to 5min.; first save current sample period to V3TIME.
8865 V3TIME = V0TIME
8870 V0TIME = 5
'
8895 MASSI2= 0.0
8897 POKE! (N6+90),MASSI2,1
'
8900 A$ = "B"
8905 RETURN
'
'
9000 ..CHANGE_PARAMETERS
'
'
9010 DISPLAY (0,0) "    <select change:";
9020 DISPLAY (1,0) "'A*' FLOW RATE      ";
9030 DISPLAY (2,0) "'B*' regen. upperlim";
9040 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
9050 INPUT KEYPAD$ 10,A$
'
9060 IF A$ = "A" THEN GOSUB ..CHANGE_FLOWRATE
9065 IF A$ = "A" THEN GOTO ..MARK3
'
9070 IF A$ = "B" THEN GOSUB ..CHANGE_REGEN_UPPER_LIMIT
9075 IF A$ = "B" THEN GOTO ..MARK3
'
9080 IF A$ <> "C" THEN GOTO 9010
9090 DISPLAY (0,0) "    <select change:";
9100 DISPLAY (1,0) "'A*' ..unused       ";
9110 DISPLAY (2,0) "'B*' ..unused       ";
9120 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
9150 INPUT KEYPAD$ 10,A$
'
'9160 IF A$ = "A" THEN GOSUB ..??
'9165 IF A$ = "A" THEN GOTO ..MARK3
'9170 IF A$ = "B" THEN GOSUB ..??
'9175 IF A$ = "B" THEN GOTO ..MARK3
'
```

```
9180 IF A$ <> "C" THEN GOTO 9090
9190 DISPLAY (0,0) "    <select change:";
9200 DISPLAY (1,0) "'A*' I2COL eff. [I2]";
9210 DISPLAY (2,0) "'B*' regen. trigger ";
9220 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
9250 INPUT KEYPAD$ 10,A$
'
9360 IF A$ = "A" THEN GOSUB ..CHANGE_I2COL_EFFLUENT
9365 IF A$ = "A" THEN GOTO ..MARK3
9370 IF A$ = "B" THEN GOSUB ..CHANGE_REGEN_TRIGGER
9375 IF A$ = "B" THEN GOTO ..MARK3
'
9380 IF A$ <> "C" THEN GOTO 9190
9390 DISPLAY (0,0) "    <select change:";
9400 DISPLAY (1,0) "'A*' Cumulative I2mg";
9410 DISPLAY (2,0) "'B*' current THRUPUT";
9420 DISPLAY (3,0) "'C*' MORE OPTIONS...";
'
9450 INPUT KEYPAD$ 10,A$
'
9460 IF A$ = "A" THEN GOSUB ..CHANGE_CUMULATIVE_I2MASS
9465 IF A$ = "A" THEN GOTO ..MARK3
9470 IF A$ = "B" THEN GOSUB ..CHANGE_CURRENT_THRUPUT
9475 IF A$ = "B" THEN GOTO ..MARK3
'
9480 IF A$ <> "C" THEN GOTO 9390
9490 DISPLAY (0,0) "    <select change:";
9500 DISPLAY (1,0) "'A*' COM2 scan freq ";
9510 DISPLAY (2,0) "'B*' MCV bed volume ";
9520 DISPLAY (3,0) "'C*' TO RETURN CNTRL";
'
9550 INPUT KEYPAD$ 10,A$
'
9560 IF A$ = "A" THEN GOSUB ..CHANGE_COM2_SCAN_PERIOD
9565 IF A$ = "A" THEN GOTO ..MARK3
9570 IF A$ = "B" THEN GOSUB ..CHANGE_MCV_BED_VOLUME
9575 IF A$ = "B" THEN GOTO ..MARK3
'
9580 IF A$ <> "C" THEN GOTO 9490
'
9590 GOTO ..MARK3'
9700 ..MARK3
9705 CC$ = "B"
9710 RETURN
```

```
'11000 ..CHANGE_FLOWRATE
'
11100 'Change the flow rate to its observed(actual) value.  This value is used
11105 'with the I2COL eff.[I2] and washout history to calc. regen. duration.
11110 DISPLAY (0,0) "The current flowrate";
11120 DISPLAY (1,0) "                    ";
11130 DISPLAY (1,0) "is ";Q*1000;" cc/min.";
11140 DISPLAY (2,0) "enter>  'xxx.x*'    ";
11150 DISPLAY (3,0) "                    ";
11160 DISPLAY (3,0) "new rate> ";
'
11170 INPUT KEYPAD$ 10,QQ$
11180 QQ = VAL(QQ$)
'
11185 'Check to see that the value is a real number in the proper range.
11190 IF QQ <= 999.9 THEN GOTO 11210
11200 GOTO ..CHANGE_FLOWRATE
11210 IF QQ >= 0.0 THEN GOTO 11230
11220 GOTO ..CHANGE_FLOWRATE
11230 Q = QQ/1000
11240 POKE! (N6+150),Q,1
'
11500 A$ = "A"
11510 RETURN
'
'
12000 ..CHANGE_REGEN_UPPER_LIMIT
'
12100 'Change upper safety limit on regeneration(initially 7.5ppm).
12110 DISPLAY (0,0) "The upper safety    ";
12120 DISPLAY (1,0) "                    ";
12130 DISPLAY (1,0) "limit is ";UPLIMI2;" ppm";
12140 DISPLAY (2,0) "enter>  'xxx.x*' -  ";
12150 DISPLAY (3,0) "                    ";
12160 DISPLAY (3,0) "new value> ";
'
12170 INPUT KEYPAD$ 10,QQ$
12180 QQ = VAL(QQ$)
'
12185 'Check to see that the value is a real number in the proper range.
12190 IF QQ <= 19.9 THEN GOTO 12210
12200 GOTO 12000
12210 IF QQ > 0.0 THEN GOTO 12230
12220 GOTO 12000
12230 UPLIMI2 = QQ
```

```
12240 POKE! N6+270,UPLIMI2,1
'
'
12500 A$ = "B"
12510 RETURN
'
13000 ..CHANGE_MCV_BED_VOLUME
'
'
13100 'Change MCV bed volume.
13110 DISPLAY (0,0) "The current MCV bed ";
13120 DISPLAY (1,0) "                    ";
13130 DISPLAY (1,0) "volume is ";BV;" cc";
13140 DISPLAY (2,0) "enter>  'xxxx.x*'   ";
13150 DISPLAY (3,0) "                    ";
13160 DISPLAY (3,0) "new value> ";
'
13170 INPUT KEYPAD$ 10,QQ$
13180 QQ = VAL(QQ$)
'
13185 'Check to see that the value is a real number in the proper range.
13190 IF QQ <= 10000 THEN GOTO 13210
13200 GOTO 13000
13210 IF QQ > 0.0 THEN GOTO 13230
13220 GOTO 13000
13230 BV = QQ
13240 POKE! N6+630,BV,1
'
13500 A$ = "B"
13510 RETURN
'
'
15000 ..CHANGE_I2COL_EFFLUENT
'
15100 'Change I2COL [I2] eff. level to its observed(actual) value.  This value
15105 'is used with the flowrate and washout history to calc. regen. duration.
15110 DISPLAY (0,0) "The I2COL [I2] eff. ";
15120 DISPLAY (1,0) "                    ";
15130 DISPLAY (1,0) "expected is";CRYSI2*1000;" ppm";
15140 DISPLAY (2,0) "enter> 'xxx.x*'(ppm)";
15150 DISPLAY (3,0) "                    ";
15160 DISPLAY (3,0) "actual value>";
'
15170 INPUT KEYPAD$ 10,QQ$
15180 QQ = VAL(QQ$)
```

```
'
15185 'Check to see that the value is a real number in the proper range.
15190 IF QQ <= 999.9 THEN GOTO 15210
15200 GOTO 15000
15210 IF QQ > 0.0 THEN GOTO 15230
15220 GOTO 15000
15230 CRYSI2 = QQ/1000
15240 POKE! N6+180,CRYSI2,1
'
15500 A$ = "A"
15510 RETURN
'
'
16000 ..CHANGE_REGEN_TRIGGER
'
16100 'Change regeneration trigger level(initially 2ppm).
16110 DISPLAY (0,0) "The regen. trigger  ";
16120 DISPLAY (1,0) "                    ";
16130 DISPLAY (1,0) "level is ";BOTLIMI2;" ppm";
16140 DISPLAY (2,0) "enter> 'xxx.x*'(ppm)";
16150 DISPLAY (3,0) "                    ";
16160 DISPLAY (3,0) "new trigger> ";
'
16170 INPUT KEYPAD$ 10,QQ$
16180 QQ = VAL(QQ$)
'
16185 'Check to see that the value is a real number in the proper range.
16190 IF QQ <= 9.9 THEN GOTO 16210
16200 GOTO 16000
16210 IF QQ > 0.0 THEN GOTO 16230
16220 GOTO 16000
16230 BOTLIMI2 = QQ
16240 POKE! N6+300,BOTLIMI2,1
'
16500 A$ = "B"
16510 RETURN
16600 ..CHANGE_CUMULATIVE_I2MASS
'
16700 'Change the current washout cumulative I2 mass value.
16710 DISPLAY (0,0) "The cumulative I2   ";
16720 DISPLAY (1,0) "                    ";
16730 DISPLAY (1,0) "mass is";MASSI2*1000;" mg";
16740 DISPLAY (2,0) "enter> 'xxxx.x*'(mg)";
16750 DISPLAY (3,0) "                    ";
16760 DISPLAY (3,0) "new mass>";
```

```
'
16770 INPUT KEYPAD$ 10,QQ$
16780 QQ = VAL(QQ$)
'
16785 'Check to see that the value is a real number in the proper range.
16790 IF QQ <= 99999.9 THEN GOTO 16810
16800 GOTO 16600
16810 IF QQ >= 0.0 THEN GOTO 16830
16820 GOTO 16600
16830 MASSI2 = QQ/1000
16840 POKE! (N6+90),MASSI2,1
'
16900 A$ = "A"
16910 RETURN
'

17000 ..CHANGE_CURRENT_THRUPUT
'
17100 'Change the total throughput(liters) as of the last COM2 scan.
17110 DISPLAY (0,0) "The current through-";
17120 DISPLAY (1,0) "                    ";
17130 DISPLAY (1,0) "put is";THRU;" L";
17140 DISPLAY (2,0) "enter> 'xxxx.x*'(L) ";
17150 DISPLAY (3,0) "                    ";
17160 DISPLAY (3,0) "new thruput>";
'
17170 INPUT KEYPAD$ 10,QQ$
17180 QQ = VAL(QQ$)
'
17185 'Check to see that the value is a real number in the proper range.
17190 IF QQ <= 500000 THEN GOTO 17210
17200 GOTO 17000
17210 IF QQ >= 0.0 THEN GOTO 17230
17220 GOTO 17000
17230 THRU = QQ
17240 POKE! (N6+120),THRU,1
'
17500 A$ = "B"
17510 RETURN
'
17600 ..CHANGE_COM2_SCAN_PERIOD
'
17630 IF R$ = WASH$ THEN V4TIME = V0TIME
17640 IF R$ = RGEN$ THEN V4TIME = V3TIME
'
```

```
17700 'Change the WASHOUT scan frequency of COM2(initially checked every 15min.).
17710 DISPLAY (0,0) "Washout COM2 scan   ";
17720 DISPLAY (1,0) "                    ";
17730 DISPLAY (1,0) "time is ";V4TIME;" min.";
17740 DISPLAY (2,0) "enter> 'xxx.x*'(min)";
17750 DISPLAY (3,0) "                    ";
17760 DISPLAY (3,0) "new period>";
'
17770 INPUT KEYPAD$ 10,QQ$
17780 QQ = VAL(QQ$)
'
17785 'Check to see that the value is a real number in the proper range.
17790 IF QQ <= 999.9 THEN GOTO 17810
17800 GOTO 17600
17810 IF QQ >= 0.0 THEN GOTO 17830
17820 GOTO 17600
17830 IF R$ = WASH$ THEN V0TIME = QQ
17840 IF R$ = RGEN$ THEN V3TIME = QQ
'
17900 A$ = "A"
17903 'Save the washout COM2 scan frequency in battery backed RAM.
17905 POKE! N6+330,QQ,1
17910 RETURN
''
18000 ..CHECK_REG_TIME_UP
'
18010 ITROFF
18130 'see if regeneration time is up and terminate regen. if it is.
18140 'the index P is used in the following subroutine.
18145 P = 2
18150 GOSUB ..TOTAL_ELLAPSED_TIME
18160 'the value of TT is calculated in the above subroutine.
18165 'the value of CDAT$ determined in the above subroutine.
18170 '
'
18200 IF TT < ZXTIME THEN GOTO 18290
18210 '
18270 GOSUB ..TERMINATE_REGENERATION
'
18290 ITRON
18295 RETURN
'
'
'
'
```

```
18400 ..INITIALIZE_RAM
'
18405 'Initiallize the user RAM for data storage. All locations 0.0 initially.
18410 'Thruput is stored in floating point format in four consecutive locations.
18420 'Iodine concentration is stored in the same manner. The
18430 'Throughput in liters and the [I2] in ppm are thus in eight consec. loc.
18440 'The 0 location in segment 1 has an absolute address of &10000.
18443 'The maximum user RAM address is 65,535(&26BFF). N=0,1,...,8190(=65,520/8)
18445 'Last 190 memory blocks have been set aside for battery backed variables.
18447 N3= -1
18450 FOR N1 = 0 TO 8000
18460 POKE! (N1*8),0.0,1
18470 POKE! ((N1*8)+4),0.0,1

18475 N3 = N3+1
18480 IF N3 < 100 THEN GOTO 18495
18485 DISPLAY (0,0) "     ";TIME$(0);"     ";
18486 DISPLAY (1,0) "                    ";
18488 DISPLAY (2,0) "initializing memory ";
18489 DISPLAY (3,0) "                    ";
18490 DISPLAY (3,0) " ";N1;" of ";" 8001 ";
18493 N3= 0
18495 NEXT N1
18500 RETURN
'
'
18600 ..VIEW_REGEN_TIME
'
18605 'View the current or last regeneration duration; assuming regeneration
18606 'was not manually terminated.
18610 DISPLAY (0,0) "The last regener-   ";
18620 DISPLAY (1,0) "ation duration was  ";
18630 DISPLAY (2,0) "                    ";
18640 DISPLAY (2,0) " ";ZXTIME;"minutes";
18650 DISPLAY (3,0) "if timeout occurred.";
'
18660 DELAY 5
18680 RETURN
'
'
18700 ..RETREIVE_VARIABLES
'
18720 VSFLAG = PEEK(N6,1)
18725 '
18730 R$ = PEEK$((N6+30),1)
```

```
18740 I2$ = PEEK$((N6+60),1)
18750 MASSI2 = PEEK!((N6+90),1)
18760 THRU = PEEK!((N6+120),1)
18770 Q = PEEK!((N6+150),1)
18775 '
18780 CRYSI2 = PEEK!(N6+180,1)
18790 FLAGCOM2$ = PEEK$(N6+210,1)
18800 ZXTIME = PEEK!(N6+240,1)
18810 UPLIMI2 = PEEK!(N6+270,1)
18820 BOTLIMI2 = PEEK!(N6+300,1)

18830 V0TIME = PEEK!(N6+330,1)
18840 N = PEEK!(N6+360,1)
18850 'Retrieve the recalibration coefficients.
18855 FOR N7 = 0 TO 2
18860 a(N7) = PEEK!(N6+390+(N7*30),1)
18865 NEXT N7
18875 PFLAG = PEEK(N6+480,1)

18885 PKI2 = PEEK!(N6+540,1)
18890 '
18895 LRTHRU = PEEK!(N6+570,1)
18900 '
18905 ECFLAG$ = PEEK$((N6+600),1)
18910 '
18915 BV = PEEK!(N6+630,1)
'
18950 RETURN
'
'
19000 ..SUPERIODINATION
'
19005 'The superiodination function allows for sterilization of tank(s) and
19010 'plumbing downstream of the RMCV iodine monitor. This will not be allowed
19020 'if regeneration is in progress. An upper time limit of 30minutes will
19030 'be the maximum allowable biocide shock sterilization time. This
19040 'superiodination(S.I.) can be manually terminated before 30 minute limit.
19050 'The first iodine scan following a S.I. should not occur immediately
19055 'since there has been no flow through the MCV.
19060 'The time keeping function for the iodine scan rate will thus be reset.
'
19070 IF R$ = RGEN$ THEN GOTO 19400
19072 IF R$ = SI$ THEN GOTO 19300
19074 IF R$ <> WASH$ THEN GOTO 19500
'
```

```
19080 'Input the superiodination time; up to 30 minutes.
19090 DISPLAY (0,0) "Enter duration of   ";
19100 DISPLAY (1,0) "SUPERIODINATION:    ";
19110 DISPLAY (2,0) "(time <= 30 minutes)";
19120 DISPLAY (3,0) "                    ";
19130 DISPLAY (3,0) "duration(min.)>";
'
19150 INPUT KEYPAD$ 10,QQ$
19160 QQ = VAL(QQ$)
'
19170 'Check to see that the value is a real number in the proper range.
19180 IF QQ <= 30 THEN GOTO 19200
19190 GOTO 19080
19200 IF QQ > 0.0 THEN GOTO 19220
19210 GOTO 19080
19230 GOSUB ..TIMEINMINUTES
19240 '
19250 SUTIME = T + QQ
19260 '
19262 'Turn relays on and begin superiodination.
19264 BIT 65,0,1
19266 BIT 65,2,1
19270 R$ = SI$
19275 POKE$ (N6+30),R$,1
19280 '
19290 GOTO 19500
'
19300 'Turn off both valves and resume washout.
'
19310 BIT 65,0,0
19320 BIT 65,2,0
19325 R$ = WASH$
19327 POKE$ (N6+30),R$,1
'
19330 'THE FOLLOWING SECTION RESETS THE LAST ..COM2 SCAN TIME TO
PRESENT TIME.
' the index P is used in the following subroutine.
19340 P = 0
19350 GOSUB ..TOTAL_ELLAPSED_TIME
'
19360 'iodine value collected from COM2 with period >= V0TIME(15min. initially).
19389 'T0(0) is the last time that COM2 was read.
19390 T0(0) = T
19391 'LDAT$(0) is the last date that COM2 was read.
19392 LDAT$(0) = CDAT$
```

```
19393 'initiallize the day counter.
19394 CNTRDAY(0) = 0
19395 'the variables T and CDAT$ are from ..TOTAL_ELLAPSED_TIME.
19398 GOTO 19500

19400 DISPLAY (0,0) "                    ";
19410 DISPLAY (1,0) "SUPERIODINATION IS  ";
19420 DISPLAY (2,0) "NOT ALLOWED DURING A";
19430 DISPLAY (3,0) "  REGENERATION!!!!!!";
19435 DELAY 5

19500 A$ = "."
19520 RETURN
'
'
19750 ..QUIT
19760 '
19780 'Input whether or not current program variables are to be saved.
19790 DISPLAY (0,0) "Are program values  ";
19800 DISPLAY (1,0) "to be saved?  i.e.  ";
19810 DISPLAY (2,0) "MASSI2,THRU,Q,...   ";
19820 DISPLAY (3,0) "                    ";
19830 DISPLAY (3,0) "(A*=yes/B*=no)>";
'
19850 INPUT KEYPAD$ 10,QQ$
'
19870 'Check to see that the value is valid.
19880 IF QQ$ = "A" THEN GOTO 19920
19900 IF QQ$ = "B" THEN GOTO 19940
19910 GOTO 19780
19920 VSFLAG = 2
19930 GOTO 19960
19940 VSFLAG = 4
19950 GOTO 19960
'
19960 A$ = "."
19970 POKE N6,VSFLAG,1
19980 RETURN
'
```

```
20000 DISPLAY (0,0) "UMPQUA RESEARCH RMCV";
20010 DISPLAY (1,0) " ";DATE$(0);"  ";TIME$(0);" ";
20020 DISPLAY (2,0) "                    ";
20030 DISPLAY (3,0) " program terminated ";

20035 'make sure ALL relayS ARE off.
20040 BIT 65,0,0
20050 BIT 65,1,0
20060 BIT 65,2,0
20100 END
□
```

```
PROGRAM: COM1DUMP.BAS
AUTHOR: RICHARD R. WHEELER, JR.

DIM I2$(1000)
DIM THRU$(1000)

OPEN "COM1:9600,N,8,1,RS,CS0,DS,CD" FOR RANDOM AS #1 LEN = 256
MM$ = MID$(DATE$, 1, 2)
F$ = "DUMPRMCV." + MM$

FOR L = 1 TO 8
        CLS
        PRINT "READING COM1..."
        FOR K = 0 TO 999
                INPUT #1, I2$(K)
                INPUT #1, THRU$(K)
        NEXT K

        PRINT "WRITING TO HARDDISK..."
        OPEN F$ FOR APPEND AS #2

        FOR J = 0 TO 999
                WRITE #2, I2$(J), THRU$(J)
        NEXT J
        CLOSE #2
NEXT L
CLS
PRINT "READING COM1..."
FOR K = 0 TO 0
        INPUT #1, I2$(K)
        INPUT #1, THRU$(K)
NEXT K

PRINT "WRITING REMAINDER TO HARDDISK..."
OPEN F$ FOR APPEND AS #2

FOR J = 0 TO 0
        WRITE #2, I2$(J), THRU$(J)
NEXT J
CLOSE #1
CLOSE #2

END
```

PROGRAM: COM1FLY.BAS
AUTHOR: RICHARD R. WHEELER, JR.

```
OPEN "COM1:9600,N,8,1,RS,CS0,DS,CD" FOR RANDOM AS #1 LEN = 256
CLS

LABEL1:

FOR n1 = 1 TO 10000
    IF INKEY$ = "q" OR INKEY$ = "Q" THEN GOTO 500
NEXT n1

PRINT "WAITING FOR DATA FROM RMCV CONTROLLER OVER COM1..."
INPUT #1, I2$
INPUT #1, THRU$

PRINT "AT TIME "; TIME$
PRINT "[I2](in ppm) = "; I2$
PRINT "TOTAL THROUGHPUT(in liters) = "; THRU$
PRINT ""

' WRITING TO HARDDISK...
D$ = DATE$
YY$ = MID$(D$, 9, 2)
DD$ = MID$(D$, 4, 2)
MM$ = MID$(D$, 1, 2)
F$ = "PR-" + MM$ + "-" + DD$ + "." + YY$

OPEN F$ FOR APPEND AS #2
WRITE #2, TIME$, I2$, THRU$
CLOSE #2

GOTO LABEL1:

500 PRINT " "
PRINT "COM1FLY.exe terminated"

CLOSE #1

END
```

APPENDIX II

PROTOTYPE RMCV OPERATIONS MANUAL

TABLE OF CONTENTS

I. INTRODUCTION

This manual is intended to familiarize the user with the operation of the Regenerative Microbial Check Valve(RMCV) prototype unit developed for NASA-JSC under contract NAS9-18361, entitled REGENERABLE BIOCIDE DELIVERY UNIT. The basic function of the RMCV unit is the same as that of an expendable MCV, to provide significant contact kill of the microbial population and to passively deliver residual iodine into a process stream at a level between 4 and 2 mg/L I_2. In addition the self controlled RMCV periodically regenerates the MCV resin with iodine, thus effectively increasing the life of the MCV.

II. HARDWARE DESCRIPTION

1. GENERAL PHYSICAL LAYOUT

The physical layout of the RMCV prototype is shown in Figure 1. Key pieces of hardware are the MCV and the I_2-crystal beds each mounted interior to the 15 inch cubic frame, the ASTRO IODINE BENCH mounted on the side of the frame, the Quick Disconnect panel mounted on the lower right side of the RMCV, and the electronics enclosure mounted on top of the frame.

2. USER INTERFACES

The hardware user interfaces are mounted on the Quick Disconnect (QD) panel. The interfaces are clearly labeled and include both inlet and outlet QD's for the process water stream. In addition to the integrated iodine monitor, provided by ASTRO International Corporation, the capability to use an external iodine monitor was incorporated into the design. An external monitor can be used in series with, or as a placement for, the ASTRO IODINE BENCH. Two QD's are provided to allow for an effluent split stream to be diverted through the external monitor and then back into the RMCV plumbing. The volume of flow through the external monitor can be continuously adjusted from 0 to full flow by varying a needle valve mounted on the QD panel. Turned fully clockwise the entire the entire flow is routed through the external monitor. If the full flow is to go through an external monitor, the minimum orifice size in the external stream path should not be less than 1/8 inch I.D. to minimize back pressure. A maximum RMCV operating pressure of 5 psig is allowed. The needle valve should be fully open (fully counter-clockwise) when an external monitor is not connected to the QD's. Additional fittings and plumbing are provided with the RMCV prototype to facilitate connection to 1/4" and 1/8" o.d. plumbing.

III. SOFTWARE TECHNICAL DESCRIPTION

1. CONTROL FLOW CHART

A flow chart showing the overall algorithm logic flow is shown in Figure 2.

2. AUTONOMOUS CONTROL

The RMCV control program is designed to operate continuously, controlling the RMCV operation automatically with no user interaction required. The ability to change many operational parameters has, however, been provided to allow for the flexibility.

3. USER INTERFACE

The user interfaces to the RMCV control system are the serial ports, located on the side of the electronics enclosure, the LCD/Keypad interface mounted on the face of the electronics enclosure, and the ON/OFF power switch located on the right side of the electronics enclosure.

3.1 DISPLAYED OPTIONS

This section will give a screen by screen description of the displayed options shown on the RMCV prototype's Liquid Crystal Display(LCD) when a keypad interrupt occurs. Each key on the keypad has been defined to have a specific function. Most of the keys display on the LCD the value that is shown on their surface. The exceptions are the '*' key, the '#' key, and the 'D' key; each of these have been redefined to have the respective functions '<enter>', '.', and '-' respectively during program operation. A screen by screen explanation follows:

screen #1 (Invoked by pressing any key)

'A*' TO RECALIBRATE

'B*' TO CHANGE PARAMETERS

'C*' MORE OPTIONS...

For the first screen, a timeout function has been implemented so that in the case of an inadvertent key entry the program will not remain in the menu mode indefinitely. If, however, a second screen is entered beyond this screen then the operator must complete keypad entry until return to the normal program control is achieved.

Pressing the key 'A' followed immediately by the key '*' will take the user to a screen that will allow recalibration of the iodine data received from the iodine monitor.

Pressing the key 'B' followed by the key '*' will take the user to a screen that will allow change of RMCV operating parameters, such as flow rate, COM2 scan rate and so forth.

Pressing the key 'C' followed by the key '*' will take the user to the next high level option display window. The menu that is displayed is as follows:

screen #2

'A*' DUMP DATA COM1

'B*' INITIATE REGEN.

'C*' MORE OPTIONS...

Pressing the key 'A' followed immediately by the key '*' will initiate a 'data dump' to the RMCV serial port labelled "COM1 DATA OUT"(see section 3.2 for a description of use).

Pressing the key 'B' followed by the key '*' will take the user to a screen that will allow manual initiation of regeneration for an amount of time entered by the user.

Pressing the key 'C' followed by the key '*' will take the user to the next high level option display window. The menu that is displayed is as follows:

screen #3

'A*' INITIALIZE RAM

'B*' VIEW REGEN TIME

'C*' MORE OPTIONS...

Pressing the key 'A' followed immediately by the key '*' will initiate a subroutine that initializes the area in battery backed RAM in which 8000 (throughput, iodine) pairs of data can be stored. A zero value is stored in each location during initialization.

Pressing the key 'B' followed by the key '*' will allow the user to view the time that elapsed during the last regeneration.

Pressing the key 'C' followed by the key '*' will take the user to the next high level option display window. The menu that is displayed is as follows:

screen #4

'A*' END REGENERATION

'B*' check COM2 y/n

'C*' MORE OPTIONS...

Pressing the key 'A' followed immediately by the key '*' will terminate a regeneration if one is currently in progress.

Pressing the key 'B' followed by the key '*' will take the user to a screen that will allows the user to select whether or not data from the system iodine monitor is read.

Pressing the key 'C' followed by the key '*' will take the user to the next high level option display window. The menu that is displayed is as follows:

screen #5

'A*' PEC enabled y/n

'B*' ..unused

'C*' MORE OPTIONS...

<u>Pressing the key 'A'</u> followed immediately by the key '*' will take the user to a screen that will allow Peak Error Correction function described in section 4 to be enabled or disabled.

<u>Pressing the key 'B'</u> followed by the key '*' will leave the user in the current screen- this is merely a place holder.

<u>Pressing the key 'C'</u> followed by the key '*' will take the user to the next high level option display window. The menu that is displayed is as follows:

<u>screen #6</u>

'A*' TO RETURN CNTRL

'#*' SUPERIODINATION

'C*' MORE OPTIONS...

<u>Pressing the key 'A'</u> followed immediately by the key '*' will return program control to normal operation.

<u>Pressing the key '#'</u> followed by the key '*' will take the user to a screen that will allow initiation of a SUPERIODINATION, for a user entered period of time to a maximum of 30 minutes. The superiodinated effluent stream bypasses the RMCV iodine monitor(s).

<u>Pressing the key 'C'</u> followed by the key '*' will take the user to the next high level option display window. The menu that is displayed is as follows:

<u>screen #7</u>

'A*' TO RETURN CNTRL

'#*' TERMINATE PROG. OPERATION

Pressing the key 'A' followed immediately by the key '*' will return control to normal operation.

Pressing the key '#' followed by the key '*' will take the user to a screen that will allow program termination. Whether or not the program variables are to be saved and used when the program is next restarted is user specified. If the variables are not to be saved, then the next time the control program starts up it will use the default values for these variables. Default variable values follow:

flow rate = 120 cc/min

crystal bed effluent iodine level = 240 mg/L

regeneration trigger point = 2.0 mg/L

regeneration upper safety MCV effluent iodine level = 7.5 mg/L

Peak Error Correction function - disabled

Iodine monitor (COM2 scan) data read period = 15 minutes

Following are the lower level menu descriptions:

screen #1A (TO RECALIBRATE)

```
a+b[I2]+c[I2]^^2            (quadratic calibration equation)
a=a,b=b,c=c                       (3 coefficients to equation)
'#'='.' 'D'='-'
enter 'a*'>
```

Each coefficient (a, b, and c) is entered using a similar screen. Once the desired coefficients have been entered, all subsequent iodine data received from the system iodine monitor will be corrected as follows:

$[I_2]_c = a+b\,([I_2]_r)+c\,([I_2]_r^2)$, where $[I_2]_c$ is the corrected iodine concentration and $[I_2]_r$ is the value received from the iodine monitor.

screen #1B.1 (CHANGE PARAMTRS)

'A*' FLOW RATE

'B*' regen. upperlim

'C*' MORE OPTIONS...

Pressing the key 'A' followed immediately by the key '*' will take the user to a screen that will allow entry of a new flow rate.

Pressing the key 'B' followed by the key '*' will take the user to a screen that will allow entry of a new regeneration upper safety limit as described in section 4.

Pressing the key 'C' followed by the key '*' will take the user to the next second level (CHANGE PARAMTRS) option display window. The menu that is displayed is as follows:

screen #1B.2 (CHANGE PARAMTRS)

'A*' ..unused

'B*' ..unused

'C*' MORE OPTIONS...

Pressing the key 'A' or 'B' followed immediately by the key '*' will keep the display in this screen.

Pressing the key 'C' followed by the key '*' will take the user to the next second level (CHANGE PARAMTRS) option display window. The menu that is displayed is as follows:

screen #1B.3 (CHANGE PARAMTRS)

'A*' I2COL eff. [I2]

'B*' regen. trigger

'C*' MORE OPTIONS...

Pressing the key 'A' followed immediately by the key '*' will take the user to a screen that will allow entry of a new Crystal Bed column effluent iodine concentration level (this is 240 mg/L [I_2] by default).

Pressing the key 'B' followed by the key '*' will take the user to a screen that will allow entry of a new regeneration trigger I_2 level. This is the RMCV effluent level at which regeneration will be auto-initiated. By default this value is 2.0 mg/L.

Pressing the key 'C' followed by the key '*' will take the user to the next second level (CHANGE PARAMTRS) option display window. The menu that is displayed is as follows:

screen #1B.4 (CHANGE PARAMTRS)

'A*' Cumulative I2mg

'B*' current THRUPUT

'C*' MORE OPTIONS...

Pressing the key 'A' followed immediately by the key '*' will take the user to a screen that will allow entry of a corrected cumulative mass of iodine depletion for the cycle currently in progress.

Pressing the key 'B' followed by the key '*' will take the user to a screen that will allow entry of a corrected total cumulative throughput volume since initiation of flow. Pressing the key 'C' followed by the key '*' will take the user to the next second level (CHANGE PARAMTRS) option display window. The menu that is displayed is as follows:

screen #1B.5 (CHANGE PARAMTRS)

'A*' COM2 scan freq

'B*' unused param.

'C*' TO RETURN CNTRL

Pressing the key 'A' followed immediately by the key '*' will take the user to a screen that will allow entry of a new scan frequency for reading iodine data from the system iodine monitor.

Pressing the key 'B' followed by the key '*' will return program control to normal operation mode.

Pressing the key 'C' followed by the key '*' will return program control to the normal operation mode.

3.2 SERIAL PORT INTERFACES

There are two serial interfaces for the RMCV prototype. The port labeled "1 DATA OUT" is a RS232 D-sub 9 pin interface that may be used to download MCV effluent iodine data and the corresponding total throughput data, which is stored in the 5080 microcontroller's battery backed RAM, to the COM1 port on an IBM compatible PC. Initiation of the routine COM1DUMP.EXE on the PC must occur before the data dump is initiated from the RMCV keypad, otherwise incomplete data transfer will occur. This same port may also be used to transfer data to a PC's COM1 as it is generated by the RMCV. This is accomplished by executing the program RMCVTRAK.EXE resident on the C: drive of the PC delivered to NASA with the prototype. Every time the microcontroller collects an iodine value from the iodine monitor, a data pair is written to RAM on the controller, and the same data pair is sent out to the COM1 port of the microcontroller whether or not it is connected to a PC.

The second port labelled "IODINE DATA IN" is used to receive iodine data from an external iodine monitor. In this case the system iodine monitor is ignored and the external I_2 values are used for system control. A simple connector change must be made on the microcontroller unit housed within the electronics enclosure. The connector labeled "From Iodine Bench" must be disconnected from the microcontroller and the connector labeled "from EXT. Monitor" must be connected in its place.

4. IMPORTANT FEATURES

There are several key features to the control algorithm that are listed below with a short explanation of each:

<u>i. Auto-boot start on powering up.</u>

This is a feature of the microcontroller. This feature is utilized by the 'power outage recover function'.

<u>ii. Power outage recovery function.</u>

This function assumes that both the process stream pump and the RMCV both lose power for the same time period. If this is true then the program will, when power is reapplied, pick up control where it left off; i.e. if in regeneration then regeneration will be continue to completion. The advantage is that no operator action is required.

If, however, the flow through the RMCV continues while power to the RMCV is off, then the program will record incorrect total throughput volumes and incorrect total iodine depletion of the MCV resin for that washout. <u>This will result in an insufficient regeneration time to replace iodine mass lost during the washout!</u> This can be compensated for by user action by one of the following means:

a) make a rough calculation of throughput and iodine mass lost during the power outage period and simply increment these parameters through keypad entry when the power is reapplied, or

b) following the next regeneration, manually initiate a regeneration through keypad entry, for a length of time that will replace the iodine mass lost during the power outage.

<u>iii. Peak Error Correction (PEC) function.</u>

This function self regulates the RMCV such that over the life of the MCV, a peak iodine value of 4.0 mg/L is targeted during each successive washout. For example, if the peak iodine value that is reached during a particular washout is 3.6 mg/L when the next regeneration occurs the program will add to the calculated washout replacement iodine mass, a small correction mass. Similarly, if the peak iodine value recorded by the program during washout is slightly over 4.0 mg/L, then the program will subtract a small correction mass for regeneration. The PEC function is by default disabled, and must be enabled by the operator after start-up through the keypad interrupt menu if it is desired. Note that a power outage will not disable this function once it has been enabled, only through the keypad interrupt menu or a controlled shutdown may it be disabled.

<u>iv. 'On the Fly' recalibration of iodine monitor.</u>

As an integral part of the RMCV prototype unit, the ASTRO IODINE BENCH cannot be directly recalibrated using the recalibration procedure shown in the manufacturer's operations manual. However, a programmatic recalibration function allowing for up to a quadratic correction of the iodine data received from the IODINE BENCH is available via keypad input. Use of this function is explained in more detail in the screen by screen menu descriptions found in section 3.1.

<u>v. Entry of Operating Parameters</u>

The prototype RMCV unit was initially equipped with a 100cm^3 bed volume MCV, and a 175cm^3 (initial volume) iodine crystal bed. It was designed to operate at a nominal flow rate of 120cm^3/min. If the RMCV is required to operate at a higher flow rate, the control algorithm will still regenerate the MCV resin properly as long as certain criteria are met. A correspondingly larger MCV and a larger crystal bed must be designed according to the minimum residence time requirements listed in Section 10 of the final report for this contract. In addition, the new flow rate must be manually entered via the keypad interrupt menus. If, on the other hand, it is desired to operate the RMCV at a lower flow rate than 120cm^3 min, the MCV and the I_2-crystal beds do not need to be replaced, but the new flow rate must still be manually entered.

The ability to change many other parameters was incorporated into the control algorithm. These are discussed in more detail in the 'screen by screen' description in section 3.1.

vi. Program safety features and error checking.

Upper and lower safety limits were placed on the length of regeneration time allowed. These are 600 minutes and 30 minutes respectively at a flowrate of 120 cm^3 min and an MCV bed volume of 100 cm^3; these will vary with different flow rates and MCV bed volumes. Note that these are hard coded parameters- i.e. they are not allowed to be changed during program operation. Another safety feature is the disallowing of successive auto-regenerations unless a minimum amount of 100 Liters of throughput has passed through the RMCV in the interim. This feature was intended to guard against inadvertent overregeneration of the MCV resin.

Error checking of the iodine data received from the IODINE BENCH is performed prior to using this data for calculations. The algorithm first verifies that the number received is in the proper range for the monitor (between 9.9 and 0.10 mg/L I_2). Next the algorithm verifies that no large changes in the received iodine value have occurred since the last data was accepted. If a change of more than 1.0 mg/L has occurred, then the last iodine value received is used for the present calculation. If bad data is being received as determined by the above error checking then the program displays a message stating that 'Bad iodine data is being received' and continues with operation using the last valid number until the iodine data received comes back within range. The second error checking function described above ($\Delta[I_2] > 1.0$) is disabled during regeneration, since it has been observed that this magnitude of change may naturally occur during regeneration if the COM2 scan period is sufficiently large. Also during regeneration a safety upper limit has been placed on the level of RMCV I_2 effluent that will be allowed. This value is 7.5 mg/L by default, but can be changed through the keypad interrupt menus. When this value is exceeded during a regeneration, the regeneration will be automatically terminated.

IV. START-UP PROCEDURE

The general start-up procedure is as follows:

i) Plumb the RMCV prototype in-line with the process stream.

ii) Establish flow through the RMCV.

iii) Plug the RMCV electronics into a 120vac, 60Hz outlet.

iv) Turn the RMCV power button on.

From this point on the RMCV will operate fully automatically. Time, data, current effluent I_2 value and cyclic status (regeneration or washout) is indicated on the LCD display.

V. SHUT-DOWN PROCEDURE

The control algorithm for the RMCV has a power outage handling capability which necessitates a special shut-down procedure. If the RMCV is to be shut-down and it is desired that the operational parameters be reset to default values, then this option must be selected programmatically as described in section 3.1. If, however, it is desired that the current operation values are to be retained, then the RMCV can be shut-down by either simply removing power, or by selecting the proper option as described in section 3.1.

For the RMCV prototype to function properly, flow must be first established through the RMCV and then immediately followed by program execution (within a few hours of establishing flow). If the program is not executing, then flow should not be established through the RMCV and conversely if there is no flow, then the RMCV program should not be invoked. The underlying reason for these requirements is the method in which regeneration times and total throughputs are determined. Incorrect values will be used by the controller if flow is not subsequently followed by program execution.

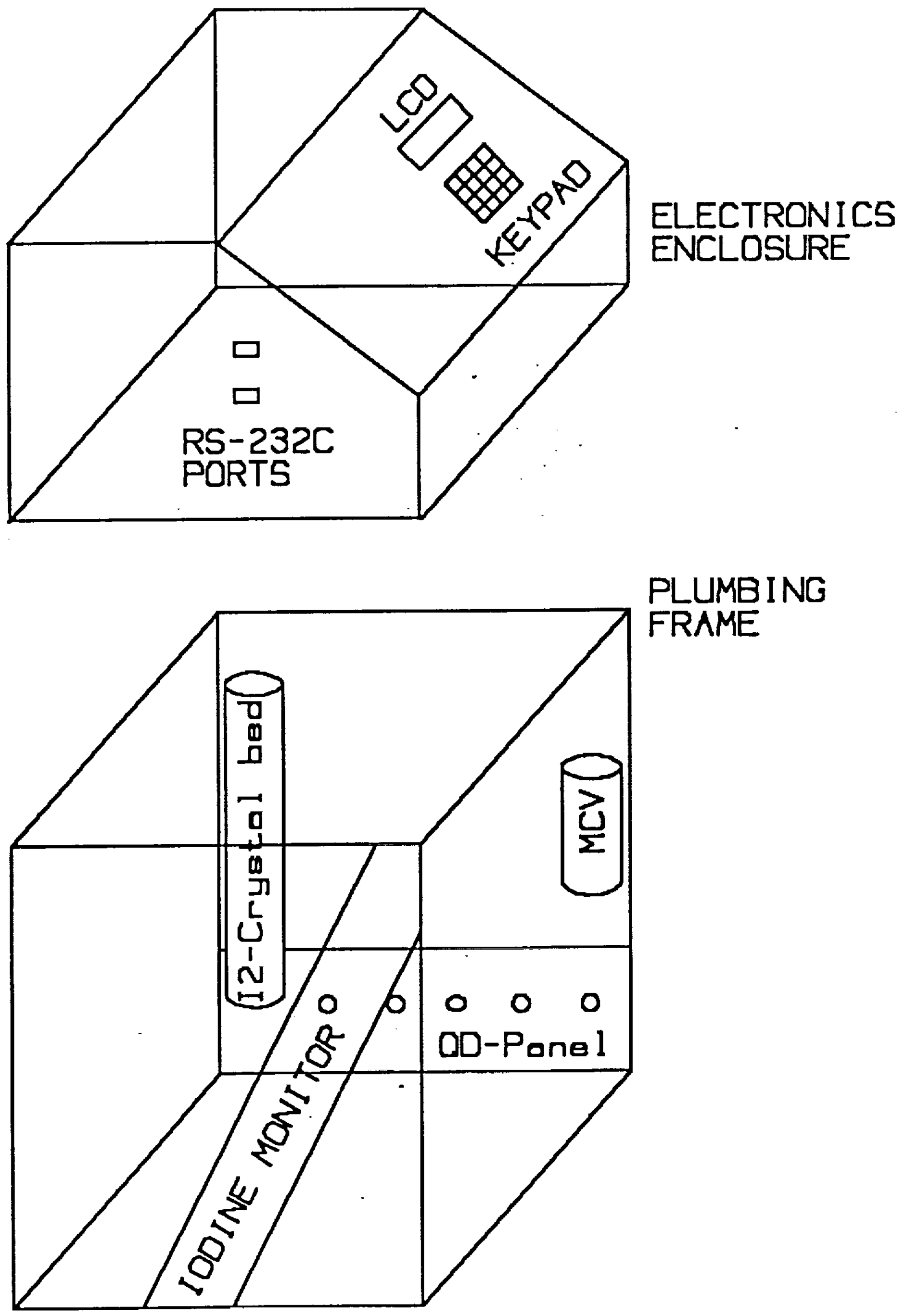

Figure 1. Prototype RMCV Prototype Layout.

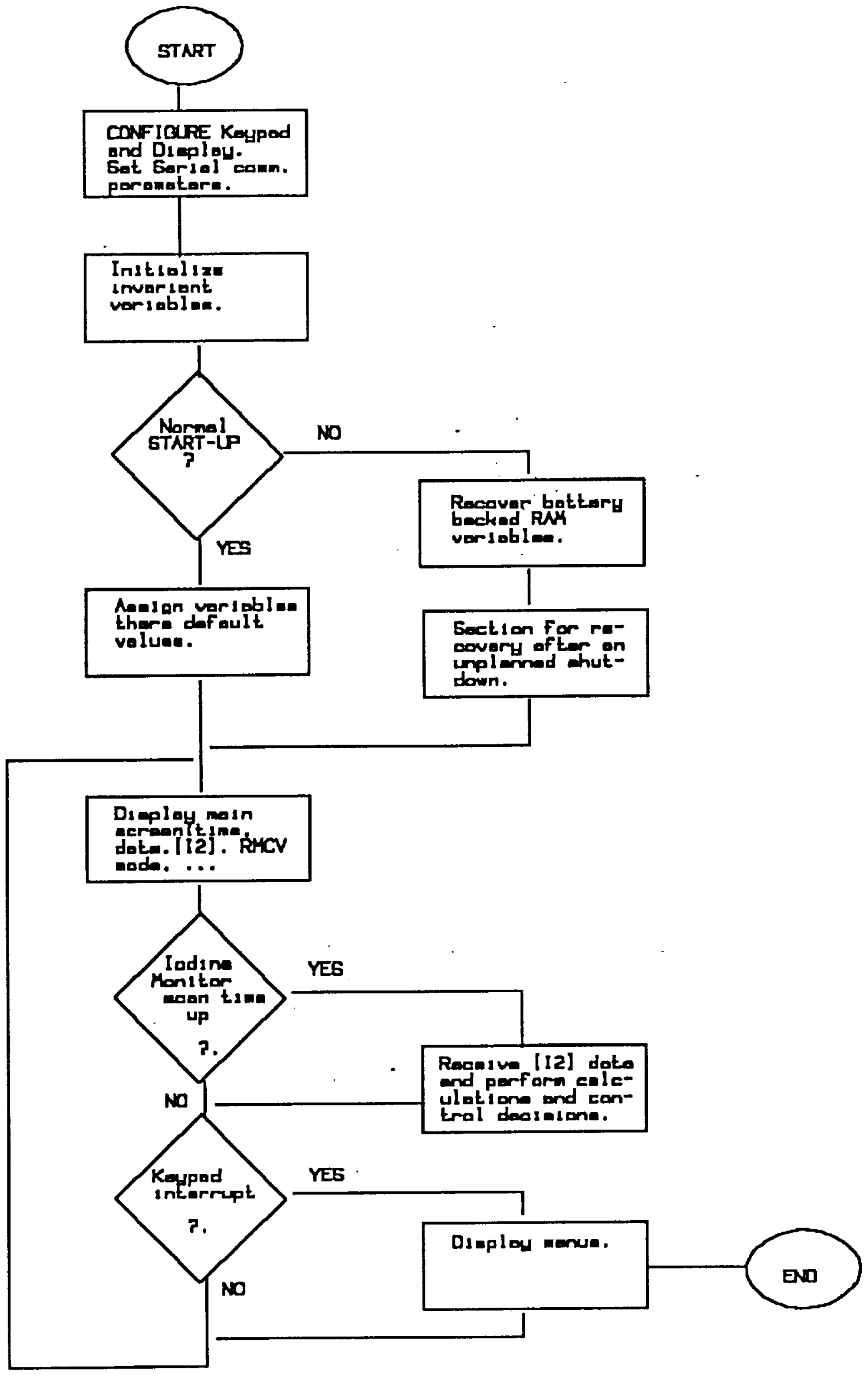
START
CONFIGURE Keypad and Display. Set Serial comm. parameters.
Initialize invariant variables.
Normal START-UP ?
NO
YES
Recover battery backed RAM variables.
Section for recovery after an unplanned shutdown.
Assign variables there default values.
Display main screen(time, date, [I2], RMCV mode, ...
Iodine Monitor scan time up ?.
YES
NO
Receive [I2] data and perform calculations and control decisions.
Keypad interrupt ?.
YES
NO
Display menus.
END

Figure 2. RMCV Control Flowchart.