86-6-127

United States
Department of
Agriculture

Forest Service

**Pacific Southwest
Forest and Range
Experiment Station**

1960 Addison Street
Berkeley
California 94704

Research Note
PSW-376

October 1985

Controlling Suspended Sediment Samplers by Programmable Calculator and Interface Circuitry

Rand E. Eads **Mark R. Boolootian**

Eads, Rand E.; Boolootian, Mark R. *Controlling suspended sediment samplers by programmable calculator and interface circuitry.* Res. Note PSW-376. Berkeley, CA: Pacific Southwest Forest and Range Experiment Station, Forest Service, U.S. Department of Agriculture; 1985. 8 p.

A programmable calculator connected to an interface circuit can control automatic samplers and record streamflow data. The circuit converts a voltage representing water stage to a digital signal. The sampling program logs streamflow data when there is a predefined deviation from a linear trend in the water elevation. The calculator estimates suspended sediment discharge from rating coefficients for the gauging site. When a threshold value of accumulated suspended sediment discharge is reached, the calculator sends a signal to the interface circuit that activates a pumping sampler. The sampling program is easily updated, and data are transferred to a computer by using a digital cassette recorder. This system increases sampling flexibility and efficiency.

Retrieval Terms: automatic pumping samplers, streamflow, suspended sediment, sampling, water quality, data logger, programmable calculator

Gauging sites located in steep terrain are often subjected to short-duration runoff events. These events reduce the probability of collecting samples during high flow conditions. Automatic pumping samplers can improve the collection of suspended sediment data when operated at fixed time intervals. Although such intervals can range from minutes to hours, long intervals are often used to limit the number of samples and reduce laboratory expense. As a result, high flow conditions are missed or undersampled. Sampling frequencies adequate for uncommon large runoff events, however, oversample low flow conditions.

A programmable calculator can improve the quality of sampling by skewing it towards important events. Moreover, the sampling program is easily updated, and field records, containing streamflow data and sampling times, can be transferred directly to a computer. Commercial controller/data loggers have been available for several years. They are often preprogrammed for a specific site, however, and generally emphasize data logging, instead of handling complex equations. Also, the cost of these devices may be prohibitive.

This note describes a system that can control the collection of pumped suspended sediment samples and record streamflow data, and includes detailed wiring schematic, component list, and program listing. The general theory of operation and software development is described in an earlier publication.[1]

METHODS

Hardware

Two commercial calculators[2]—the Hewlett Packard HP-41CV and HP-41CX—can activate pumping samplers and log streamflow data under program control. In addition to the calculator, two plug-in modules and a general purpose interface are necessary to communicate with the external interface circuitry and devices (*fig. 1*). The HP-IL provides a physical two-wire link between the calculator and the general purpose interface. The HP-IL Module codes and decodes messages and commands between the HP-IL and the calculator. The Extended I/O Module supplies additional instructions that enhance the input/output functions of the HP-IL Module. The HP-IL Converter is a general purpose interface that provides communication between the HP-IL and external circuitry supplied by the user. In addition, it provides two eight-bit parallel data buses, a bidirectional data buffer, and data transfer logic for "handshaking." Input 1 supplies a 0- to 5-VDC signal to the analog-to-digital converter that sends an eight-bit representation of stage to data bus A (*fig. 2*). Data bus B supplies output signals through the calculator and HP-IL that control the start of analog-to-digital conversion, and activate the pumping sampler. The circuit board can control two samplers: outputs 7 and 8 control sampler A, and outputs 9 and 10 control sampler B. In addition, output 3 provides a 12-VDC signal for an event marker. The handshake lines control the input from the analog-to-digital converter, power-up and -down circuitry, and latch the data in the interface circuit (Appendix A).

To extend battery life in remote areas, the circuit components that consume the most power are powered down when not in use (this includes the converter, analog-to-digital converter, and potentiometer which are connected to output 2). In our application, the HP-IL Converter uses negative control logic, which implies that all signals go true when power is removed. This complicates interface circuit design by requiring support circuitry to prevent false signals from triggering undesired events. The line for powering-up external circuitry (WKUP) is an output of the converter and functions under calculator control, even when power is off to the converter. WKUP pulses whenever an IL command is sent from the calculator, in-

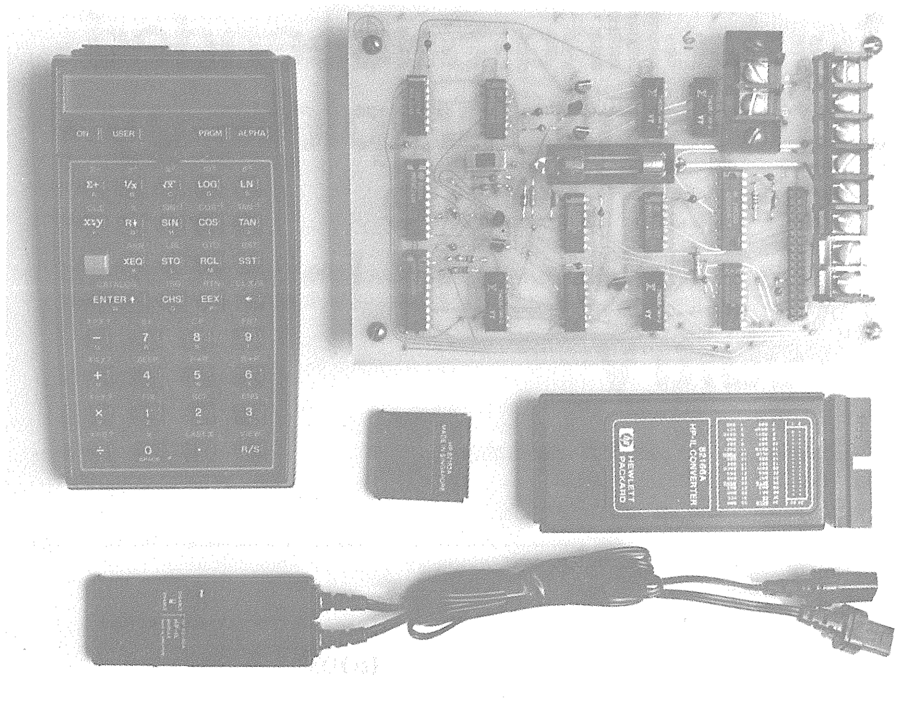

Figure 1—The sediment sampling system consists of a programmable calculator, plug-in modules, and general purpose interface to connect with the interface circuitry.

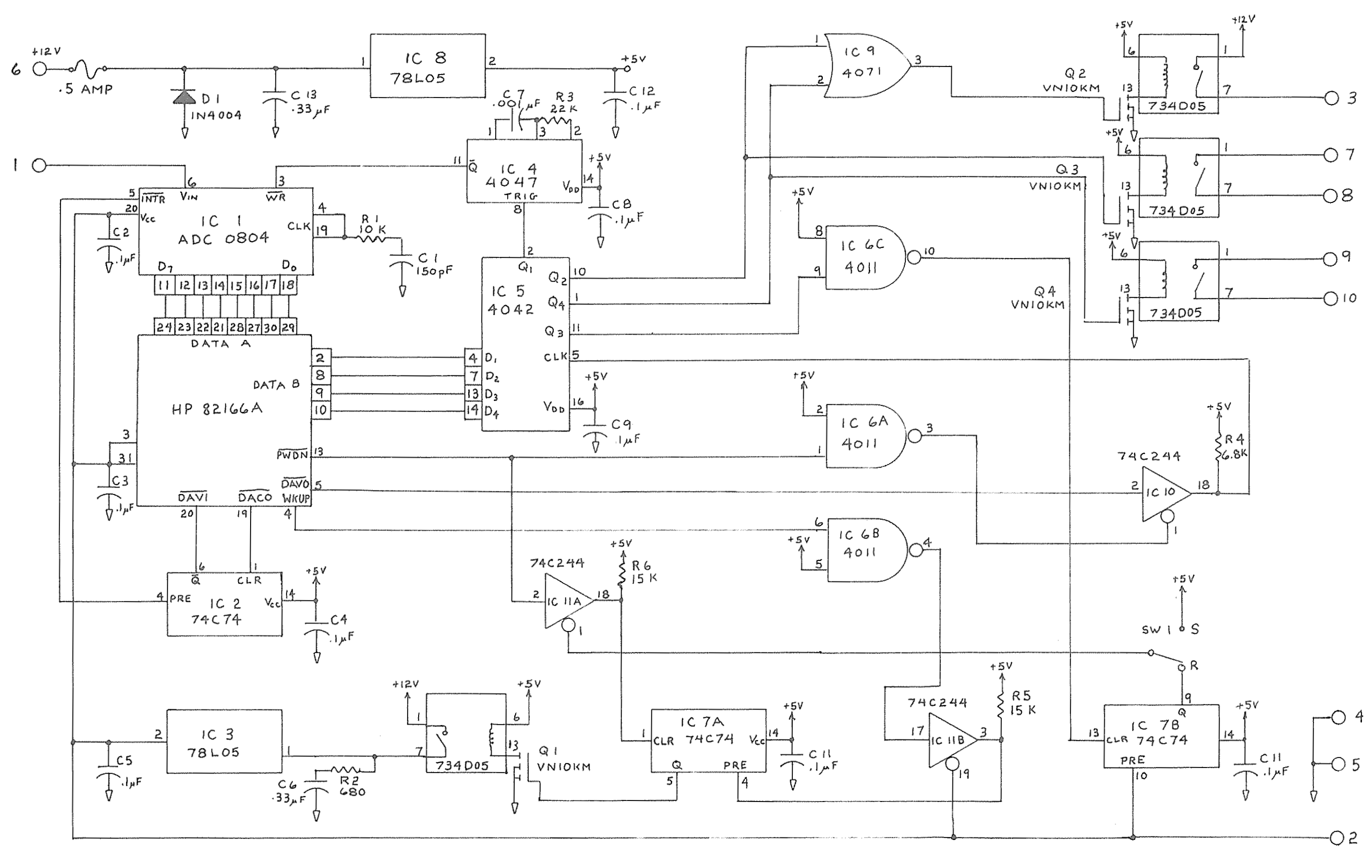

Figure 2—Electrical circuitry for the automatic sediment sampling system using a programmable calculator.

cluding the command for powering-down the external circuitry. Because the flip-flop (IC 7A) that switches power to output 2 has both the power-up and power-down lines as inputs, tri-stating the lines is necessary to prevent the signals from interfering with one another.

Data bus B on the converter is used for internal communication when it is not being used for output. Therefore, the data lines must be latched. A handshake line on the converter $\overline{DAVO}$ is used to latch the data. Since $\overline{DAVO}$ goes true when the converter is powered down, a tri-state is used with a pull-up resistor to prevent false latching.

Additionally, the circuit allows serial sampling using two pumping samplers. When the last bottle in the first sampler has been filled, the program changes control lines to permit sampling in the second sampler.

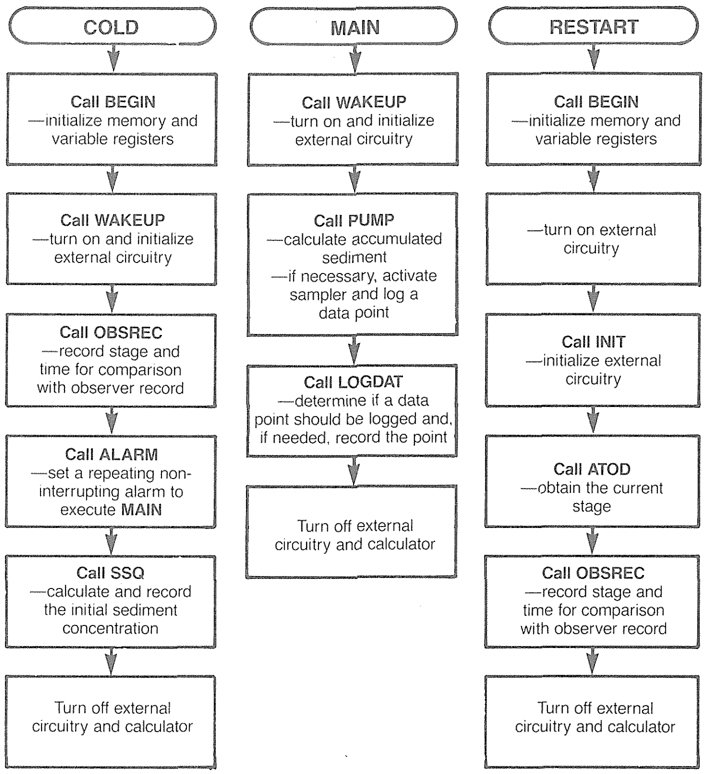

Figure 3—The routine COLD is executed when the programmable calculator begins operations. It subsequently activates the sampling routine MAIN. The RESTART routine is similar to COLD, but does not set up an alarm to execute MAIN.

Software

We developed the software to control a pumping sampler based on estimated suspended sediment discharge. Although specific to our application, modifications of the program for similar situations would not require extensive changes.

The software for the HP-41 system consists of three programs: (a) the sampling program (Appendix B), (b) the data dump routine, and (c) the program for transferring data from the calculator to the computer.

The sampling program controls the pumping sampler and event marker, performs the necessary calculations, and records streamflow data. The system is fully automated and requires no operator intervention, except for the initial calculator installation and performing data dumps. The sampling program, along with the registers containing equation coefficients and constants, is stored on a cassette tape. Using the HP Digital Cassette Drive, the program and register contents are loaded into the calculator at the office or in the field.

When a calculator is first installed at a site, the subroutine COLD is executed (*fig. 3*). It blanks memory and sets the variable data registers to appropriate values. The subroutine then applies power to the external circuitry, initializes the converter, and records the stage and time. It also sets an alarm in the calculator that will execute the sampling program (MAIN). In our application, the alarm is set to repeat once every 10 minutes (the interval can be any integral divisor of 60). Finally, COLD turns off external circuitry and then turns off the calculator.

The subroutine RESTART is executed each time a data dump is performed. RESTART executes essentially the same operations as COLD, with one exception: it does not set an alarm. The alarm set by COLD continues operating. An alarm will not activate if the calculator is on; thus it is possible to perform data dumps and service the sampler without interruption. When RESTART has completed the initialization it turns off both the external circuitry and the calculator.

When the alarm time is due, the calculator turns on and executes the subroutine MAIN. MAIN estimates the current rate of sediment discharge and accumulates the estimated amount of sediment which passed the gauging station since the last wake-up period. After a threshold amount of sediment has been exceeded, MAIN activates the pumping sampler.

MAIN also determines if a data point should be logged. A data point, consisting of time and stage, requires two bytes of memory. It is recorded if the current stage deviates by more than a predetermined tolerance from the line defined by the two previously recorded data points. Stage is read from the analog-to-digital converter and time is the number of wake-up intervals since the last recorded data point. The maximum number of wake-up intervals between recorded data points is never allowed to exceed 255 (without recording a point). A total of 408 data points can be stored.

The data dump program is a short series of instructions that retrieve data from the calculator and write them onto a cassette tape. Performing a data dump does not interfere with normal operation except for delaying (by one or two wake-up intervals) pumping sampler activation or data logging. The cassette tape is then taken to the office for analysis.

At the office, data from the cassette are reloaded onto a calculator for transfer to a computer. Data transfer, from calculator to computer, takes place under calculator program control. The calculator is connected to the computer through the HP-IL/RS-232-C Interface. The interface is programmable and can be configured to "talk" to any computer that uses the RS-232-C protocol. Although data are stored in binary format in the calculator, the program transfers them in ASCII format.

DISCUSSION

Automatic collection of suspended sediment samples can be improved by using a programmable calculator capable of controlling a pumping sampler. This system provides the flexibility to update program coefficients and perform data dumps in the field by persons with minimal training. Data logging occurs only when the stage hydrograph deviates from a linear trend. Data are removed from the calculator in the field by using a digital cassette recorder. Records are then transferred to a computer without intermediate data reduction or manual entry. Reliability, low cost (less than $1000), and flexibility make this system attractive for many types of environmental sampling studies.

Implementation of this system will require several levels of knowledge. A computer programmer would be required to write programs for removing data from the digital cassette and transferring it to the computer. Interface circuit building, modification, and repair would require a basic understanding of digital electronics.

We are developing and testing a new sampling system that incorporates the HP-71B portable computer (programmable in Basic with 17.5- to 33.5-k of available memory). The interface hardware, developed for the HP-41CX, is compatible and requires no modification for similar sampling requirements.

ACKNOWLEDGMENT:

Steve Hankin, formerly with the Station staff and now with the Pacific Marine Environmental Laboratory, U.S. Department of Commerce, Seattle, Washington, proposed the original concept, and assisted in the design and development of the software and interface circuitry.

END NOTES AND REFERENCES

[1]Eads, R. E.; Hankin, S. C.; Boolootian, M. R. *A programmable caculator improves automatic sampling of suspended sediment.* Water Resour. Res. (In press.)

[2]Trade names and commercial enterprises or products are mentioned for information only. No endorsement by the U.S. Department of Agriculture is implied.

[3]This device has been discontinued and replaced by the HP 82166C HP-IL Interface kit. This kit contains the individual components necessary to construct the converter.

APPENDIX A—Components to Construct System

The commercial components necessary to construct this system are manufactured by Hewlett-Packard and include:

- HP-41 CV or HP-41 CX calculator
- HP-IL Module (HP 82160A)
- Time Module (HP 82182A)—HP-41 CV only
- Extended I/O Module (HP 82183A)
- HP-IL Converter (HP 82166A)[3]
- Digital Cassette Drive (HP 82161A)
- HP-IL/RS-232-C Interface (HP 82164A)

Battery exchanges for the interface circuit, using a 12-VDC, 9.5 Ah gelled electrolyte battery, and the calculator, using 1.5-VDC, size N alkaline batteries, are necessary at about 8-week intervals, depending on temperature, when using a 10-minute wake-up interval.

The temperature limits for operating the calculator are 0 to 45 °C. Temperatures below 0 °C may affect program execution, resulting in temporary system failure which requires the assistance of a field technician. The use of this system outside of the operating limits may require insulation, or in extreme conditions, heating or cooling.

Information on circuit construction and program is available from the authors at Pacific Southwest Forest and Range Experiment Station, 1700 Bayview Drive, Arcata, California 95521.

APPENDIX B—System Software

```
LBL AL          ;ALARM
                  ;set a repeating non-interrupting
                  ;alarm that will execute the Main program
RCL 06          ;get wakeup interval
100
/               ;make into repeat interval
ENTER
ENTER           ;place it in the Z-reg
RCL 13          ;get number of wakeup intervals since truncated restart
2
+
*               ;calculate number minutes past the hour for the first
                  alarm
RCL 07          ;get time of restart
INT             ;truncate it
+               ;starting time of first alarm
HR
HMS             ;fix time if it's incorrect
0               ;start today
X<>Y            ;put into proper register
"MA"            ;program to execute on alarm
XYZALM
RTN
LBL AT          ;ATOD
                  ;read in value from A to D and
                  ;leave in the X-reg

0
```

```
OUTXB
1               ;fire the A to D
OUTXB           ;needs a rising pulse
INXB            ;input data point
RTN
LBL BE          BEGIN
                  ;initialize memory and variable registers
TIME
STO 28          ;save time
INT             ;hours only
XEQ DE
ST- 09          ;time of last wakeup after a restart
RCL 28          ;get time
STO 07          ;save as time of restart
DATE            ;get today's date
STO 08          ;save as date of restart
144             ;max number of bytes for pumped samples
STO 11          ;save
-9              ;force logging of next two data points
STO 12          ;save false stage
0               ;initialize selected variables
STO 23          ;number of intervals since last logging
STO 24          ;slope at last logging
STO 25          ;stage at last logging
STO 26          ;number samples taken
30.17001
STO 27          ;blank memory: reg 30-170
CLA
LBL 01          ;top of loop
ASTO IND 27
ISG 27          ;increment and skip if greater
GTO 01          ;bottom of loop
FIX 4           ;4 places to right of decimal point
RTN
LBL CO          ;COLD
                  ;initial program run upon installation.
                  ;set up calculator environment and set alarm
TIME            ;DATE and TIME executed
STO 07          ;in order to prevent
DATE            ;blow-ups during
STO 08          ;a COLD start
XEQ BE
RCL 07          ;get time of restart
XEQ WA
XEQ OR
STO 29          ;save record
RCL 13          ;get number wakeup intervals since truncated restart
1
+               ;increment
STO 09          ;save as time of last wakeup
XEQ AL
0
STO 22          ;accumulated sediment is 0
XEQ SS
STO 10          ;save sediment concentration for next wakeup
4               ;prepare for the PWRDN command
OUTXB
0
OUTXB
PWRDN           ;turn off peripherals
OFF             ;put the calculator to sleep
END
LBL DE          ;DELTIME
                  ;calculate the number of wakeup
                  ;intervals since the truncated restart
                  ;on entry: X-reg = current time
RCL 08          ;get DATE of restart
DATE            ;get today's date
DDAYS           ;compute difference
1440
*               ;convert to minutes
X<>Y            ;place current time back in X-reg
RCL 07          ;get time of restart
INT             ;calculate difference between truncated time of restart
                  and
                  ;current exact time
HR              ;convert to decimal hours
X<>Y
HR              ;convert to decimal hours
-               ;get difference
HMS             ;back to clock form (hours, minutes, seconds)
STO 27          ;save it
INT             ;integer part is hours
60
*               ;in minutes
RCL 27          ;fractional part is minutes
FRC
100             ;move decimal point
*
+               ;minutes and seconds
HMS+            ;total minutes
STO 27          ;save it
RCL 06          ;get number minutes in a wakeup interval
MOD             ;compute error (how far from wakeup interval are we)
CF 09           ;clear 'badtme' flag
.03             ;allowed error is 3 seconds
X<=Y?           ;is error <=3 seconds
SF 09           ;yes! set 'badtime' flag
RCL 27          ;get difference in time
RCL 06          ;get wakeup interval
/               ;calculate number of wakeup intervals
INT             ;whole intervals only
STO 13          ;save
RTN             ;X-reg contains number of wakeup
                  ;intervals since last restart
LBL IN          ;INIT
                  ;perform converter set up
CLRLOOP         ;clear devices on loop
64
FINDAID         ;get converter ID
SELECT          ;make primary device
CLA             ;load alpha register with ASCII values for converter
                  setup
50
XTOAR
64
XTOAR
0
XTOAR
90
XTOAR
ADROFF          ;turn off auto addressing
1
LAD             ;make converter a listener
0               ;ready for control register setup
DDL
3
OUTAN           ;set register
UNL             ;unlisten converter
ADRON           ;auto addressing back on
XEQ AT
2
DEVL            ;clear transfer buffer
RTN
LBL LO          ;LOGDAT
                  ;determine if we should log a data point
```

```
RCL 11          ;get number bytes stored so far
6
/
RCL 05          ;get maximum register number we can store in
INT             ;make number bytes into a register number
X>Y?            ;is memory full
RTN             ;yes! return
144             ;put max number intervals allowed between loggings in
                  X-reg
RCL 13          ;get number wakeup intervals since trunc restart
RCL 23          ;get number wakeup intervals since last logged point
-               ;compute difference
STO 28          ;save
X>Y?            ;exceeded number allowed intervals
GTO 01          ;yes! force a logging
FS? C 08        ;no! has a pumped sample been taken
GTO 01          ;yes! force a logging
RCL 25          ;get stage at last logging
RCL 28          ;get number wakeup intervals since last logging
RCL 24          ;get slope at last logging
*
+               ;calculate a predicted value for current stage
RCL 12          ;get actual current stage
-               ;compute difference
ABS             ;absolute value
RCL 03          ;get allowed tolerance
X>Y?            ;is difference between predicted and actual less than
                  allowed
                  ;tolerance
RTN             ;yes! return
LBL 01          ;no! log a point
RCL 12          ;get current stage
11              ;register number containing number of bytes stored so
                  far
XEQ ST
RCL 28          ;get number of wakeup intervals since last logging
11              ;register number containing number of bytes stored so
                  far
XEQ ST
RCL 12          ;get current stage
RCL 25          ;get stage at last logging
-               ;compute difference
RCL 28          ;get number wakeup intervals since last logging
/               ;calculate slope between this stage and previous
STO 24          ;save slope
RCL 12          ;get current stage
STO 25          ;save as old stage
RCL 13          ;get number wakeup intervals since trunc restart
STO 23          ;save as number intervals at last logging since trunc
                  restart
RTN
LBL LR          ;LASTREC
                  ;make a calculator interpolating record
                  ;execute during a data dump
XEQ WA
XEQ OR
STO 28          ;save record
END
LBL MA          ;MAIN
                  ;main program—executed each wakeup
TIME
STO 28          ;save time
XEQ WA
FS? 09          ;is 'Badtime' flag set
GTO 01          ;yes! This is not a routine wakeup
XEQ PU
XEQ LO
LBL 01
4               ;prepare for PWRDN command
OUTXB
0
OUTXB
PWRDN           ;power down peripherals
OFF             ;put calculator to sleep
END
LBL OR          ;OBSREC
                  ;make a calculator interpolation record
                  ;format is SSSHHMM where SSS is stage in feet
                  ;and HHMM is time
CF 29           ;no commas in display
TIME            ;get current time
100
*
INT             ;convert time into 2400 hour integer form
STO 27          ;save
CLA
ARCL 27         ;recall time into alpha register
LBL 01
ATOXR           ;get rightmost char in alpha register
46
X=Y?            ;is it a period
GTO 02          ;yes!
GTO 01          ;no!—go back and keep looking
LBL 02
ALENGIO         ;length of string in alpha register
4
X=Y?            ;is it 4 characters
GTO 03          ;yes!
48              ;no!—insert a 0 in front of string
XTOAL
LBL 03          ;save string in alpha register
ASTO 27
CLA
RCL 12          ;get stage
XEQ TO          ;convert to meters
100
*               ;shift decimal point
ARCL X          ;move contents of X-reg into alpha register
LBL 04
ATOXR           ;get rightmost character in alpha register
46
X=Y?            ;is it a period
GTO 05          ;yes!
GTO 04          ;no!—keep looking
LBL 05
ALENGIO         ;get length of string in alpha register
3
X=Y?            ;is it 3 characters
GTO 06          ;yes!
48              ;no!—place a 0 in front of string
XTOAL
LBL 06
49              ;place a 1 in front of string
XTOAL
ARCL 27         ;get time and append to string in alpha register
ANUMDEL         ;move alpha string into X-reg
SF 29           ;turn commas on
RTN
LBL PU          ;PUMP
                  ;determine if a pumped sample should
                  ;be taken and do so
XEQ TS
ST+ 22          ;sum sediment
RCL 04          ;get threshold sediment value
RCL 22          ;get accumulated sediment value
X<=Y?           ;is accum. sed. value less than or equal to threshold
```

```
                value
RTN             ;yes! No sample yet
72              ;no!—max number samples in sampler A
RCL 26          ;get number samples taken
X<Y?            ;number samples taken less than number samples held
                  by sampler A
GTO 01          ;yes!—pump from pumping sampler A
144             ;Max number samples to be taken in sampler B
X<>Y            ;swap X and Y
X<Y?            ;number samples taken less than number samples held
                  by both
                  ;samplers
GTO 02          ;yes!—pump from pumping sampler B
RCL 04          ;no bottles left—get threshold value
ST- 22          ;subtract threshold value from accumulated value and
                  save
RTN
LBL 01          ;fire pumping sampler A
0
OUTXB
2
OUTXB           ;set data line high
PSE             ;hold high for a moment
0
OUTXB
GTO 03
LBL 02          ;fire pumping sampler B
0
OUTXB
8
OUTXB           ;set data line high
PSE             ;hold high for a moment
0
OUTXB
LBL 03
RCL 04          ;get threshold sed value
ST- 22          ;subtract threshold sed value from accumulated value
                  and save
RCL 12          ;get current stage
26              ;register number containing number pumped samples
                  taken so far
XEQ ST
RCL 13          ;get number wakeup intervals since trunc restart
256
MOD             ;(number intervals since trunc restart) modulo 256
26              ;reg. number containing number pumped samples taken
                  so far
XEQ ST
RCL 13          ;get number wakeup intervals
256
/
INT             ;INT (number wakeup intervals/256)
256
XEQ ST
SF 08           ;set flag indicating pumped sample was taken
                  ;(will force logging of data point)
XEQ LO
RTN
LBL RE          ;RESTART
                  ;restart program—performed after every data dump
                  ;resets calculator environment
XEQ BE
SF 25           ;ignore errors
PWRUP           ;turn on external circuitry
XEQ IN
XEQ AT
X=0?            ;can not have 0 stage
1
STO 12          ;save stage
```

```
XEQ OR
STO 29          ;save record
4               ;set up for power down
OUTXB
0
OUTXB
PWRDN           ;turn off external circuitry
OFF             ;turn off calculator
END
LBL SS          ;SSQ
                  ;compute sediment concentration
                  ;Q = AQ*(stage-KQ)**BQ
RCL 12          ;get current stage
XEQ TO          ;convert to meters
RCL 16          ;get KQ
-
X>0?            ;is stage—KQ > 0
GTO 01          ;yes!
0               ;no! return 0 for concentration
RTN
LBL 01
RCL 15          ;get BQ
Y^X             ;form (stage-KQ)**BQ
RCL 14          ;get AQ
*               ;= Q
ENTER
ENTER           ;place in Z-reg
                  ;C=AC*(Q-KC)**BC
RCL 19          ;get KC
X>0?            ;is Q - KC > 0
GTO 02          ;yes!
0               ;no! return 0 for concentration
RTN
LBL 02
RCL 18          ;get BC
Y^X             ;form ( Q - KC)**BC
RCL 17          ;get AC
*               ;=C
*               ;=QC
#########       ;constant to adjust units
*
RTN             ;Q*C in X-reg
LBL ST          ;STORE
                  ;store 1 byte in data array
                  ;on entry
                  ;X-reg = register number containing address
                  ;to store into
STO 27          ;save register number
RDN             ;rotate registers
RCL IND 27      ;get contents of register number
6               ;number of bytes per register
/
INT             ;discard fractional part
30              ;FIRSTREG—offset into array
+               ;register number into which we store
1
ST+ IND 27      ;increment count
RDN             ;rotate registers
STO 27          ;register to store into
RDN             ;rotate registers thus move data into X-reg
CLA
ARCL IND 27     ;get existing string
XTOAR           ;append character to ALPHA register
ASTO IND 27     ;put it back
RTN
LBL TO          ;TOMETER
                  ;convert stage from A to D units to meters
                  ;on entry: X-reg = STAGE
```

```
RCL 21     ;get slope
*          ;multiply by stage
RCL 20     ;get Y intercept
+
RTN        ;stage (meters) = A + B* stage (A to D)
LBL TS     ;TSED
             ;compute total accumulated sediment since last
             sample was taken
             ;we use the trapezoidal method
             ;result in X-reg.
XEQ SS     ;get current concentration
RCL 10     ;get sediment concentration at last wakeup
X<>Y
STO 10     ;save current sediment concentration
+
2
/          ;take the mean of the two concentrations
RCL 09     ;get number wakeup intervals since trunc restart at last
             wakeup
RCL 13     ;get number wakeup intervals since trunc restart
             currently
STO 09     ;save number wakeup intervals now for next time
X<>Y       ;swap X and Y registers
-          ;compute number intervals since now and last wakeup
             (usually 1)
*          ;multiply interval difference by concentration mean
RCL 06     ;get number minutes in a wakeup interval
*
60
*          ;to obtain CMS
RTN        ;leave value in X-reg
LBL WA     ;WAKEUP
             ;perform power up initializations
             ;for external circuitry
             ;on entry: X-reg contains time
SF 25      ;ignore PWRUP error
PWRUP      ;turn on peripherals
XEQ DE
XEQ IN     ;initialize circuit
XEQ AT     ;read A to D
X=0?
1          ;can not have 0 stage
STO 12     ;save stage
RTN
```

REGISTERS

00	program revision number
01	observer record
02	observer record
03	allowed error in stage prediction
04	threshold volume of sediment discharge
05	relative number of last useable data register
06	TIME interval (in minutes) to elapse between wakeups
07	TIME of restart
08	DATE of restart
09	numbers of wakeup intervals since truncated restart at last wakeup
10	sediment discharge rate at last wakeup
11	number of logged data points so far ($\times 2$)
12	current stage
13	number of wakeup intervals between truncated restart and this wakeup
14	discharge rating equation coefficient
15	discharge rating equation coefficient
16	discharge rating equation coefficient
17	concentration rating equation coefficient
18	concentration rating equation coefficient
19	concentration rating equation coefficient
20	conversion to meters coefficient
21	conversion to meters coefficient
22	accumulated sediment discharge
23	number of wakeup intervals since truncated restart and last logging
24	slope of hydrograph at last logging
25	stage at last logging
26	number of pumped samples taken so far ($\times 3$)
27	scratch register
28	scratch register
29	initial calculator record
30-53	pumped sample data
54-190	logged data

The Authors:
are assigned to the Station's research unit studying Pacific Coastal Forests, with headquarters at the Redwood Sciences Laboratory, Arcata, Calif. **RAND E. EADS** is a hydrologic technician. He has been a member of the staff since 1975. Native of Los Angeles, he earned a biology degree (1974) at Humboldt State University, Arcata. **MARK R. BOOLOOTIAN** is a computer programmer analyst. He earned a mathematics degree (1983) at Humboldt State University. He joined the Station staff in 1982.