

## **Appendix 3 (Autonomous Drifter)**

### **USF TECHNICAL REPORT**

#### **AUTONOMOUS DRIFTER**

USF obtained the fiberglass shell for the autonomous drifter from Technocean, it is the same shell used in the MMS study of oil tracking surface drifters in the Norwegian Sea in 1991. The almost spherical shell is approx 12 inches in diameter (30 cm) with a small symmetric keel on the bottom that extends down 2 inches (4-5cm) from the diameter of the sphere, Figure 28. The shell is approximately 1/4 to 1/8th inch (.3 to .6 cm) thick, fiberglass made in two hemispheres with a 3/8 inch (1cm) lip on both halves extending radially inward toward the sphere center and Technocean fiber glasses the two halves together for their customers, after populating the interior with electronics. USF needed a reusable drifter, and so incorporated two mating right circular cylinders one glassed to the bottom shell and one to the top shell, the bottom cylinder lip extends 4 inches above the lip on the bottom shell thus acting as a water barrier and its close fit into the top cylinder together with a gasket makes for a water tight fit between the shell halves, while allowing the halves to be separated Figure 29.

Approximately 8 ounces of lead shot were glassed into the small keel at the bottom of the lower shell half to keep the drifter stable and provide the proper water line. A 12 amp hour 12 volt Power Sonic gell cell was used as a power source and was set at the bottom center of the bottom shell. Two reed switches were

Appendix 3 (Continued)

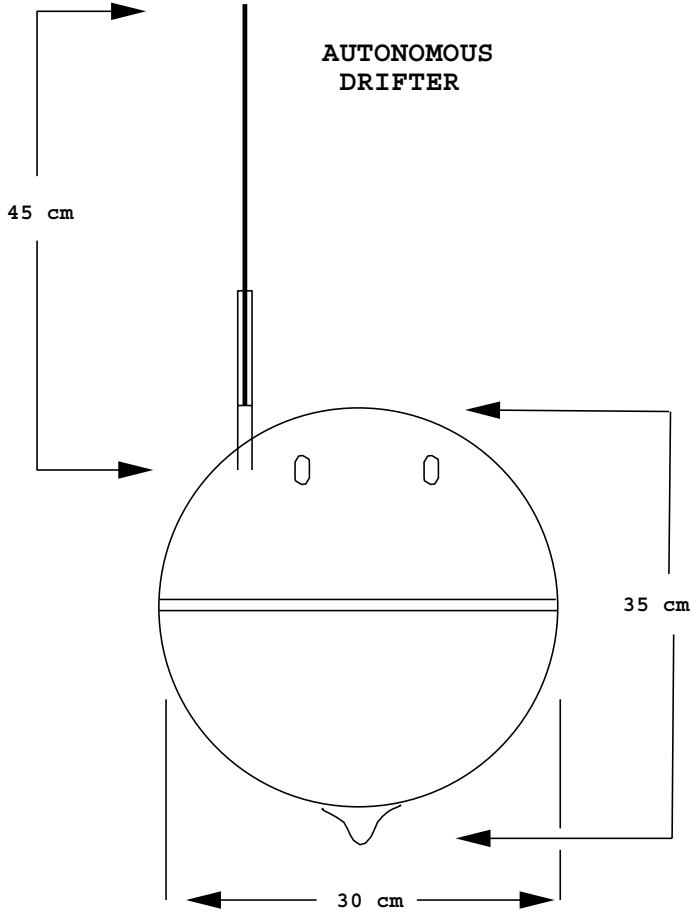


FIG. 44. Autonomous Drifter Specifications.

### Appendix 3 (Continued)

affixed to the bottom shell to provide exterior control of power application to the radio and gps and a separate power switch to the data logger.

The GPS used was a Garmin 36 TracPac which is a GPS and antenna with a NEMA output in a very small 2.23 x 3.796 x 1.047 inches (5.664 x 9.642 x 2.66 cm) light weight 4.4 oz (124.5 g) package. The Garmin 36 tracks up to 12 satellites draws 150 mA average at 12 volts and has 15 meter RMS position accuracy.

The communications to the drifter were via Freewave spread spectrum radio (see Appendix 2), which required the use of a small external whip antenna, (see Figure 28) which was the major difference between the Technocean drifter and the USF drifter. The data logger employed in the drifter was JK Microsystems Flashlite control and data acquisition computer, the Flashlite has 2 serial ports (one for the gps and one for communications) and on-board flash memory to store the gps data. This allowed for the ability to back up data on-board the drifter when communications were down or ineffective.

Appendix 3 (Continued)

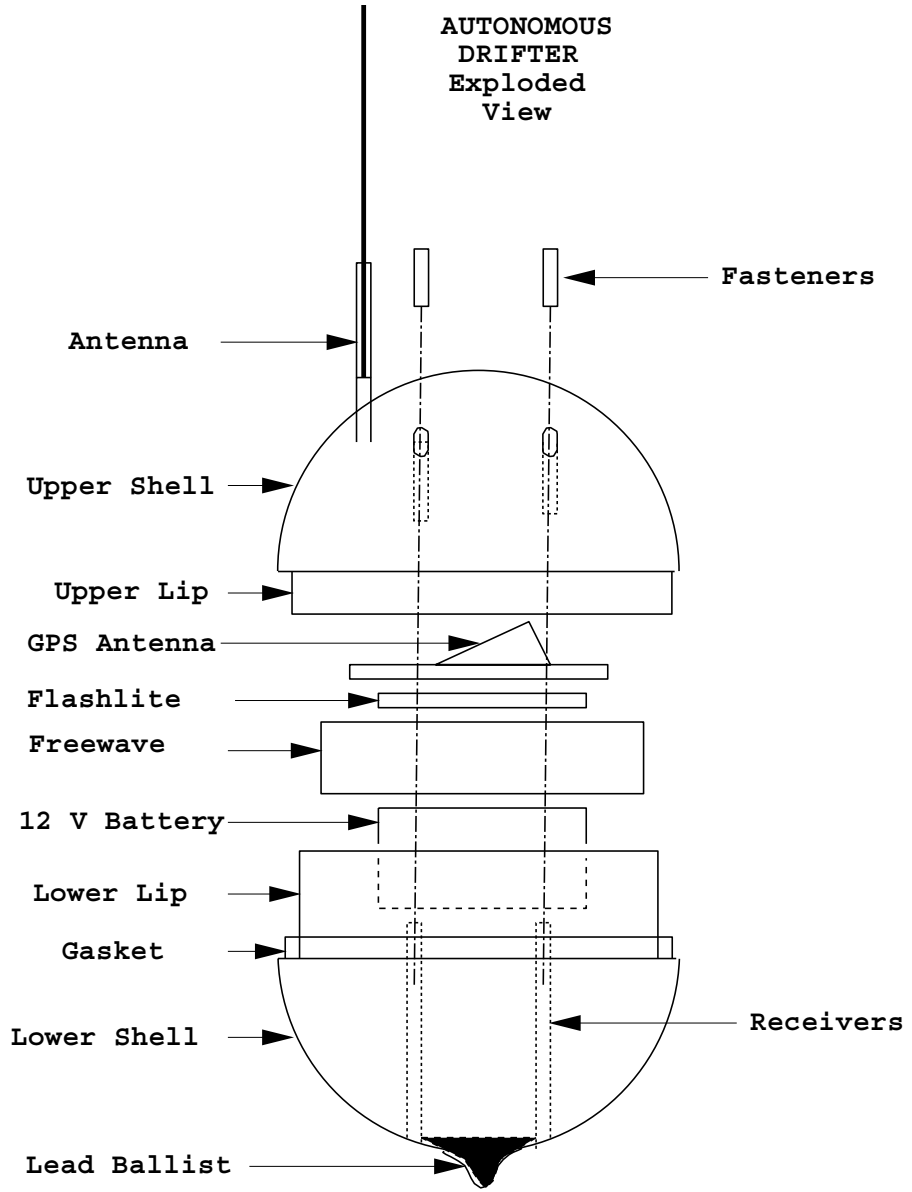


FIG. 45. Autonomous Drifter Exploded View.

### Appendix 3 (Continued)

The Flashlite software was modified from the on-board serial communications program in qbasic. The code used is provided below:

```
OPEN "o", 1, "cons:" ' open screen output
PRINT #1, "Drifter code modified from"
PRINT #1, "Flashlite Terminal Program Version 1.1"
PRINT #1,
ON ERROR GOTO DiskError ' error trapping on
OPEN "i", 2, "term.dat" ' open the comm parameter file
ON ERROR GOTO 0 ' error trapping off
PRINT #1, "Loading default line parameters...." GoTerm:
INPUT #2, baud% ' input baud rate
CALL setbaud(baud%) ' set it
INPUT #2, parity% ' input parity
CALL setparity(parity%) ' set it
INPUT #2, databits% ' input data bits
CALL setstop(databits%) ' set it
INPUT #2, stopbits% ' input stop bits
CALL setbit(stopbits%) ' set it
CALL StartRx(version%) ' start the receive routine, get vers.
PRINT #1, "Line parameters loaded, Comm routine version "; (version% / 10)
PRINT #1, "Press ESC to end, erase TERM.DAT and rerun to configure."
```

### Appendix 3 (Continued)

```
PRINT #1,  
DEF SEG = &HF000 ' segment of V-25 I/O  
POKE &HFF01, &HFC ' make PM0 bits 0 & 1 low (output)  
POKE &HFF00, 0 ' make P0 bits 0 & 1 low, true DSR  
  
n = 0 'set loop counters for prints  
  
m = 0 'and  
  
TD$ = "" 'initialize string var  
  
cnt = 720 'set internal loop size  
  
OPEN "o", 3, "pos.dat" 'set output file  
  
PRINT #3, "This is the output file for"  
  
PRINT #3, "USF Drifter Buoy #1, the ASCII"  
  
PRINT #3, "strings have the format; "  
  
PRINT #3, "$GPRMC,1,2,3,4,5,6,7,8,9,10,11hh<CR><LF>"  
  
PRINT #3, "1 is UTC time of position fix"  
  
PRINT #3, "2 is Status, A=valid pos, V=NAV reciever warning"  
  
PRINT #3, "3 is Lat, dmmm.mmmm"  
  
PRINT #3, "4 is Lat hemisphere, N or S"  
  
PRINT #3, "5 is Lon, dddmm.mmmm"  
  
PRINT #3, "6 is Lon hemisphere, E or W"  
  
PRINT #3, "7 is Speed over ground knots (240 sec filter)"  
  
PRINT #3, "8 is Course over ground, 000.0 to 359.9"
```

### Appendix 3 (Continued)

```
PRINT #3, "9 is UTC date of pos fix, ddmmyy"

PRINT #3, "10 is Magnetic variation 000.0 to 180.0"

PRINT #3, "11 is Magnetic variation direction E or W"

CLOSE #3

main:

GOSUB receive ' do we need to receive any chars?

GOSUB send ' do we need to send any chars?

GOTO main ' do it again

receive:

CALL getcharbuf(charcount%) ' any incoming chars?

IF charcount% = 0 THEN RETURN ' no, return

CALL getchar(char%) ' yes, get the char

b$ = CHR$(char%)

IF char% = 10 THEN n = n + 1 'loop until cnt LFs

IF n = cnt + 1 THEN GOSUB writes ' after cnt LFs reset counter 'do hour count

IF n = cnt THEN GOSUB prints 'print until next LF ' PRINT #1, CHR$(char%);

' and print it

RETURN
```

### Appendix 3 (Continued)

prints:

```
PRINT #1, b$; 'print until cnt + 1 LFs sub
```

```
TD$ = TD$ + b$ 'save string waiting for disk print
```

```
RETURN
```

writes:

```
n = 0 'reset LF counter
```

```
m = m + 1 'index hour counter
```

```
IF m = 1 THEN TD1$ = TD$ 'save 12 min strings
```

```
IF m = 2 THEN TD2$ = TD$
```

```
IF m = 3 THEN TD3$ = TD$
```

```
IF m = 4 THEN TD4$ = TD$
```

```
IF m = 5 THEN GOSUB diskpr 'if hour then print
```

```
TD$ = "" 'reset dummy string
```

```
RETURN
```

diskpr:

```
TD5$ = TD$
```

```
OPEN "a", 3, "pos.dat" 'open output file ans append
```

```
PRINT #3, TD1$ 'saved strings to output file
```

```
PRINT #3, TD2$
```

```
PRINT #3, TD3$
```

```
PRINT #3, TD4$
```

### Appendix 3 (Continued)

```
PRINT #3, TD5$

CLOSE #3 'close output file

m = 0 'reset hour counter

TD1$ = "" 'reset save strings

TD2$ = ""

TD3$ = ""

TD4$ = ""

TD5$ = ""

RETURN

send:

a$ = INKEY$ ' is a key hit?

IF a$ = "" THEN RETURN ' no, return

CALL putchar(ASC(a$)) ' yes, send it

IF ASC(a$) = 27 THEN END ' if it was ESC then end

RETURN

EchoOn:

DEF SEG = &H40 ' turn on bios console echp

POKE &H8A, 1 ' see User Manual "Programming Examples"

RETURN

EchoOff:
```

### Appendix 3 (Continued)

```
DEF SEG = &H40 ' turn off bios console echo

POKE &H8A, 0 ' see User Manual "Programming Examples"

RETURN

DiskError:

IF ERR <> 53 THEN GOTO UnknownErr

PRINT #1, "Comm parameter file not found"

GetBaud:

PRINT #1,

PRINT #1,

PRINT #1, "Enter baud rate:"

PRINT #1, "1 - 110 baud 6 - 4800 baud"

PRINT #1, "2 - 300 baud 7 - 9600 baud"

PRINT #1, "3 - 600 baud 8 - 19.2k baud"

PRINT #1, "4 - 1200 baud 9 - 38.4k baud"

PRINT #1, "5 - 2400 baud 10 - 115.2k baud"

PRINT #1,

PRINT #1, ">";

GOSUB EchoOn

INPUT baud%

GOSUB EchoOff

IF 0 < baud% < 11 THEN GOTO GetDataBits
```

### Appendix 3 (Continued)

```
PRINT #1, "Invalid baud rate"

GOTO GetBaud

GetDataBits:

PRINT #1,

PRINT #1,

PRINT #1, "Enter data bits"

PRINT #1, "7 - 7 data bits"

PRINT #1, "8 - 8 data bits"

PRINT #1,

PRINT #1, ">";

GOSUB EchoOn

INPUT databits%

GOSUB EchoOff

IF 6 < databits% < 9 THEN GOTO GetStopBits

PRINT #1, "Invalid data bits"

GOTO GetDataBits

GetStopBits:

PRINT #1,

PRINT #1,

PRINT #1, "Enter stop bits"

PRINT #1, "1 - 1 stop bit"
```

### Appendix 3 (Continued)

```
PRINT #1, "2 - 2 stop bits"

PRINT #1,

PRINT #1, ">";

GOSUB EchoOn

INPUT stopbits%

GOSUB EchoOff

IF 6 < stopbits% < 9 THEN GOTO GetParity

PRINT #1, "Invalid stop bits"

GOTO GetStopBits

GetParity:

PRINT #1,

PRINT #1,

PRINT #1, "Enter parity"

PRINT #1, "0 - none"

PRINT #1, "1 - odd"

PRINT #1, "2 - even"

PRINT #1,

PRINT #1, ">";

GOSUB EchoOn

INPUT parity%

GOSUB EchoOff
```

### Appendix 3 (Continued)

```
IF parity% < 3 THEN GOTO WriteParam
```

```
PRINT #1, "Invalid parity"
```

```
GOTO GetParity
```

```
WriteParam:
```

```
PRINT #1,
```

```
PRINT #1,
```

```
OPEN "o", 2, "term.dat"
```

```
PRINT #2, baud%
```

```
PRINT #2, parity%
```

```
PRINT #2, databits%
```

```
PRINT #2, stopbits%
```

```
CLOSE 2
```

```
OPEN "i", 2, "term.dat"
```

```
GOTO GoTerm
```

```
UnknownErr:
```

```
PRINT #1, "Disk error "; ERR
```

```
END
```

These modifications allowed for internal logging as well as data feed via the spread spectrum radios. As can be seen the data is gathered every 12 minutes, sent via radio and then logged every hour (5, 12 minute samples). The Garmin output string contains date,time, status, and velocity as well as latitude and longitude.