AtlasScientific
Biology•Technology

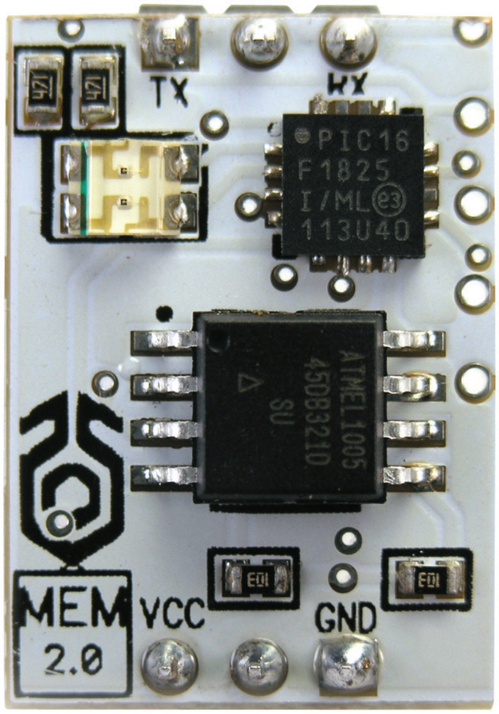
ENV-32X

Hardware V1.6
Firmware V2.0

Micro footprint Embedded Data Logging subsystem

Features

- 32Mb flash memory
- No drivers required
- Specifically designed to hold strings of up to 128 characters long
- Read/ Write/Erase 32,760 individual strings
- Fast operation read/write a 128 byte string in just 3ms
- Write to memory confirmation
- Two different organized read back methods
- 20 year data retention
- Up to 3,276,000,000 program/erase cycles
- Simple asynchronous serial connectivity (voltage swing 0-VCC)
- Simple instruction set consisting of only 7 commands
- Debugging LED's
- 3.3 volt operation
- Low power consumption
- Well suited for airborne applications weighing only 1.53 grams



4.5mA active mode*
3.5mA quiescent mode*
*LED's off

Description

Reading and storing data from sensors on a regular basis is a common practice within the scientific community. However, storing data has always been somewhat problematic. Often times an MMC/SD card is used. These card systems require complex drivers to operate and a DOS emulation system to put the data on the card, as well as a fare amount of hardware, including moving parts.

The Atlas Scientific ENV-32X Embedded Data Logging Subsystem has been specifically designed to hold the hundreds to thousands of short data strings that are so commonly used when logging sensor data. Read/Write/Erase operations are done using simple one character commands in asynchronous serial format. A full 128 byte character string is written to memory in just 3 milliseconds.

Contents

System Overview

The ENV-32X consists of 32,760 memory pages. Each page is 128 bytes long. An ASCII string of between 1-128 bytes can be sent to the ENV-32X for storage into memory simply by transmitting a '*' followed by the data to be stored such as "*1234" "*AbCd".

Page incrementing is done automatically starting at page 0. Data read back can be output in two different forms.

1) A comma separated string such as 1234,AbCd<CR>.

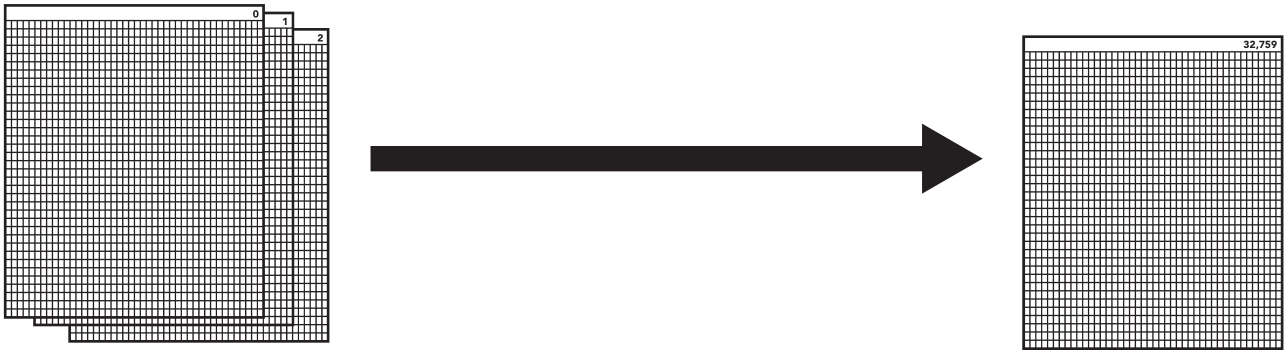
2) A numbered string ending with a <CR> such as:

0:1234<CR>
1:AbCd<CR>

Memory space available and other diagnostic information is also available.
The data stored on the ENV-32X will not be lost in the event of a power failure.

Memory structure in-depth explanation:

As it has been stated previously, the ENV-32X consists of 32,760 memory pages. To help understand this terminology one could simply imagine a lab notebook that has 32,760 pages in it. This lab notebook starts at page 0 and ends at page 32,759.



Each page of memory has the ability to hold 128 characters. If we were to imagine each page of the lab notebook being just big enough to hold 128 characters you can easily see how the memory is organized.

Including spaces, this alpha - numeric string is 128 characters long.
It is actually much longer than one may initially realize.

Memory writing in-depth explanation:

Each time data is written to the memory, a new page is used regardless of the length of the data written.

As an example, we will write 3 strings to memory.

The "*" indicates to the ENV-32X that a memory write is about to occur. The "*" is not actually written to the memory. A <CR> indicates the end of the string to be written to memory. The <CR> is not actually written to memory.

String 0 "*ABC123"<CR>

String 1 "***time= 14:34:56,pH=4.5,ORP=234,D.O=4.7,E.C=211µs, battery power= 89%, i= 11mA,battery charging=on**. <CR>

String 2 "*Including spaces, this is a string that is 128 characters long. It is actually longer than one may initially think. Space filler#" <CR>

In this example string 0 is written to page 0.
String 1 is written to page 1.
String 2 is written to page 2.

In order to maintain data integrity, the following rules govern the data writing protocol:

- Page incrimination is done automatically.
- It is not possible to write data to a specific page.
- Data written to the memory that is less than 128 bytes will consume a whole page regardless of its length.
- Data sent to the memory that is greater than 128 characters will not be written to the memory.
- Data sent to the memory that is 0 characters long will not be written to the memory.

Pin Out

GND Return from the DC power supply. GND (& Vcc) must be ripple and noise free for best operation.

Vcc 2.5 - 3.6 volts. **Voltages above 3.6 volts will damage the ENV-32X**

TX TX output delivers asynchronous serial data in TTL RS232 format, except voltages are 0-Vcc. All data output ends with a carriage return (ASCII 13).

Example:

```
1234<CR>
AbCd<CR>
```

The baud rate is: **38400, 8 bits, no parity, with one stop bit.**
The voltage swing 0-VCC, **not** +/- 12 volts
If standard voltage level RS232 is desired, connect an RS232 converter such as a MAX232.

RX TTL RS-232 receive pin

Absolute Maximum Ratings*

Parameter	MIN	TYP	MAX	Units
Storage temperature	-40		125	C°
Operational temperature	-40	25	85	C°
VCC	2.5	3.3	3.6	V

*Note: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. Exposure to maximum rating conditions for extended periods may affect device reliability

Device operation

When the ENV-32X is connected to a power supply (2.5v to 3.6v) the **green** "power on" indicator LED will be lit. The device will immediately enter standby mode and wait for a command.

There are a total of 7 different commands that can be given to the ENV-32X.
All commands must be followed by a carriage return <CR>.
Commands are not case sensitive.

Command list Quick reference

Command	Function	Default state
L[0,1]	L1: Enables debugging LEDs L0: Disables debugging LEDs	Enabled
* (ASCII 42)	Write to memory	N/A
D	When this command is sent twice in a row within 1 millisecond, all data written to the memory will be deleted	N/A
R[0,1]	Read back stored data. Where R0 will set the ENV-32X to output all stored data in a single CSV string. R1 will set the ENV-32X to output all stored data in an individual ascending numbered string.	N/A
E	Ends the data outputting event that is initiated by the R[0,1] command. The ENV-32X will enter standby/quiescent mode.	N/A
M	Transmit the current status of the memory. Free pages, free space in bytes, pages used	N/A
I	Information: Type of device | firmware version | firmware creation date	N/A

Command Definitions

L1 This will enable both debugging LED's.

The ENV-32X has two LED's
Green LED............................Power indicator
Red LED................................Instruction received/data outputting

By default, the LED's are enabled.
These LED's are designed to help the user determine that the ENV-32X is operating properly.

Changes to this setting are written to EEPROM memory and therefore will be retained even if the power is cut.

Keeping the LEDs on will consume an additional 30 mA

Full proper syntax: l1<cr> or L1<CR>

L0 This will disable both debugging LED's.

Changes to this setting are written to EEPROM memory and therefore will be retained even if the power is cut.

Full proper syntax: l0<cr> or L0<CR>

***** (ascii 42) This indicates to the ENV-32X that the string following the asterisk is to be written to memory.

Full proper syntax: *AbC123<CR>

Improper syntax:

***<CR> This would be read as string of 0 bytes.**
AbC123<CR> This would be ignored.

If the data has been successfully stored to memory the ENV-32X will respond:

Y<CR>

If the data has NOT been successfully stored to memory the ENV-32X will respond:

N<CR>

This may occur if:

- The data string sent is longer than 128 bytes.
- The data string sent is less than 1 byte.
- The memory is full.

D Instructs the ENV-32X to delete all stored data.

This instruction must be given twice in a row.
Once this command is given there is no way to undo this operation.

Full proper syntax: **D<CR> or d<CR>**
D<CR> or d<CR

The ENV-32X will respond:

X<CR>

Both LEDs will turn on, even if they have been disabled.
This operation takes approximately 20 seconds to complete depending on the amount of data to be deleted.

R[0,1] Reading back the data stored in the ENV-32X is done by simply transmitting the R command followed by a 0 or 1. The R0 command will instruct the ENV-32X to output all the stored data in a CSV format - where the comma separates the data page by page.

Example:

If the data stored was:
*123
*ABC
*!@#

The data retrieved using the R0<CR> command would be:
123,ABC,!@#<CR>

The R1 command will instruct the ENV-32X to output all the stored data in numerical order based on the page number the data was written on. Each string of page data is terminated with a <CR>.

Example:

If the data stored was:
*123
*ABC
*!@#

The data retrieved using the R1<CR> command would be:
123<CR>
ABC<CR>
!@#<CR>

E This instructs the ENV-32X to end stored data read back. This instruction can be used after an R[0,1] command has been issued. It will stop the ENV-32X from continuing to output all the data that is stored in its memory.

Delivering the "E" (END) instruction when it is not actively outputting what has previously been stored to memory will have no effect on the ENV-32X.

Full proper syntax: e<CR> or E<CR>

The ENV-32X will respond by ceasing data transmission.
There is no ASCII response to this instruction.

M Instructs the ENV-32X to return a CSV string containing the current space available
The output string is a comma separated value containing:
Free pages, free space in bytes, pages used

Full proper syntax: m<cr> or M<CR>

The ENV-32X will respond:

32760, 4193280,0<CR>
Free pages, free space in bytes, pages used

I Instructs the ENV-32X to transmit its version number.

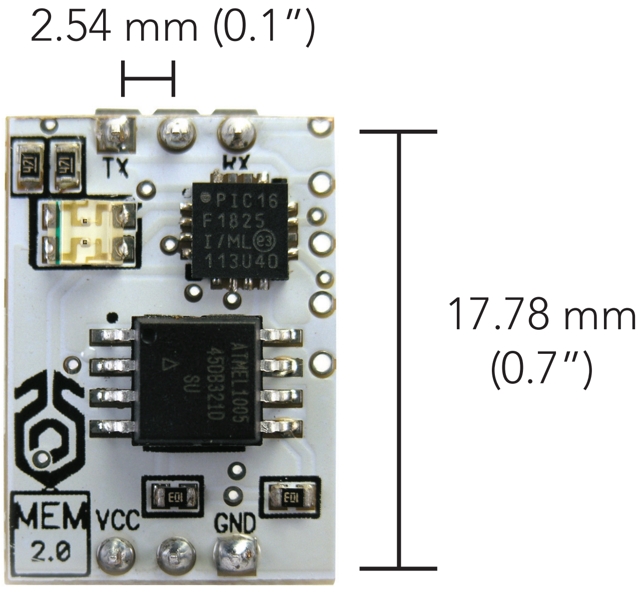
A CSV string will be transmitted that will contain 3 values.

1. The type of device:	"M" (for Memory)
2 The firmware version number:	"V2.0"
3 The firmware version date:	"2/12" (Feb / 2012)

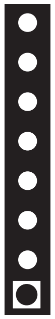
Full proper syntax: i<cr> or I<CR>

The ENV-32X will respond:
M,V2.0,2/12<CR>

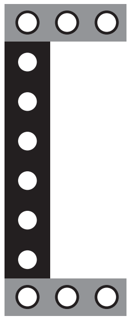
How to make a Footprint for the Atlas Scientific ENV-32X Circuit



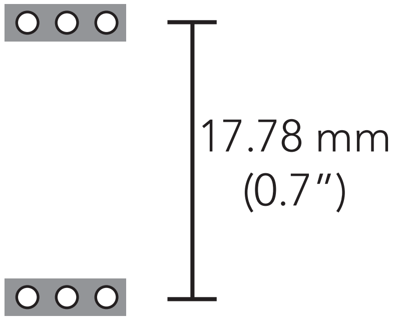
1. In your CAD software place an 8 position header.



2. Place a 3 position header at both top and bottom of the 8 position header as shown.



3. Once this is done you can delete the 8 position header. Make sure that the two 3 position headers are 17.78mm (0.7") apart from each other.



Warranty

Atlas Scientific warranty's the ENV-32X Circuit to be free of defect during the debugging phase of device implementation, or 30 days after receiving the ENV-32X Circuit (which ever comes first).

The debugging phase

The debugging phase is defined by Atlas Scientific as the time period when the ENV-32X Circuit is inserted into a bread board or shield and is connected to a microcontroller according to this wiring diagram. Reference this wiring diagram for a connection to USB debugging device, or if a shield is being used, when it is connected to its carrier board.

If the ENV-32X Circuit is being debugged in a bread board, the bread board must be devoid of other components. If the ENV-32X Circuit is being connected to a microcontroller, the microcontroller must be running code that has been designed to drive the ENV-32X Circuit exclusively and output the ENV-32X Circuit's data as a serial string.

It is important for the embedded systems engineer to keep in mind that the following activities will void the ENV-32X Circuit's warranty:

- **Soldering any part of the ENV-32X Circuit**
- **Running any code that does not exclusively drive the ENV-32X Circuit and output its data in a serial string**
- **Embedding the ENV-32X Circuit into a custom made device**
- **Removing any potting compound**

Reasoning behind this warranty

Because Atlas Scientific does not sell consumer electronics; once the device has been embedded into a custom made system, Atlas Scientific cannot possibly warranty the ENV-32X Circuit against the thousands of possible variables that may cause the ENV-32X Circuit to no longer function properly.

Please keep this in mind:

1. **All Atlas Scientific devices have been designed to be embedded into a custom made system by you, the embedded systems engineer.**
2. **All Atlas Scientific devices have been designed to run indefinitely without failure in the field.**
3. **All Atlas Scientific devices can be soldered into place.**

*****Atlas Scientific is simply stating that once the device is being used in your application, Atlas Scientific can no longer take responsibility for the ENV-32X Circuit continued operation. This is because that would be equivalent to Atlas Scientific taking responsibility over the correct operation of your entire device.