

# Software 35.00

## Release Notes

### API

#### C

**EtherSpaceLink\_open** does not check the result code of the asynchronous connect call correctly and will try and negotiate on a connection which is not connected resulting in a somewhat incorrect error code being returned. You will still get an error, but with this correction or more applicable error.

#### C

Introduction of **EtherSpaceLink\_request\_link\_status\_port** which requests port status without switching active link on DSI/MSR and 3U devices. On ESL devices it will switch the active port.

#### C++/ python

**request\_link\_status\_port** method added which performs the same as the function above for the connection class

The timetag callbacks on Windows would report incorrect wall times thus producing incorrect pcap packet timestamps.

### msr

Corrected usage of the /f parameter so as not to error.

The msr program has the ability to break large captures into smaller configurable chunks. Currently they are incrementally numbered, the file name is now the start and end times of the capture. If you want the chunks to be incrementally file numbered set the environment variable **ESL\_FILE\_INDEX**

#### PCAP capture

The offset from UTC in the PCAP header is now populated ensuring consistent timing when reading pcap files in different timezones..



## Software 34.98

### Release Notes

#### **New Linux Platform**

We now support **Ubuntu 20.04** with this release comes for support for both python 2.7 and python 3.

The default is for python 2.7 (as this is the install default) but if you wish to use python 3 then set the environment variable **ESL\_PYTHON\_VER** to **3**

## Software 34.96

### Release Notes

#### **msr capture program**

The msr capture program allows users to capture traffic on specific ports, currently it captures traffic on all ports irrespective of this parameter. This has now been rectified.

# Software 34.94

## Release Notes

### **spwio Documentation**

The PDF for spwio was incorrectly printed in landscape as opposed to portrait.

### **esl\_srscfg**

We have introduced the **esl\_srscfg** utility in order to make configuring SRS routers simpler.

### **esl\_spw**

We have introduced **esl\_rspw** a utility which allows one to send and receive RMAP/CCSDS and raw messages without resort to complex programming.

### **spwio RMAP plugin**

This plugin displayed “...” when either more than 8 bytes of data were received or if the increment address flag was set, this caused a degree of confusion. Instead, if the increment address flag is set the plugin displays the token “AI”

# Software 34.87

## Release Notes

### **MSR Capture**

When requested to capture NULLS (ESC-FCT) the msr capture program generates the appropriate records whilst recording to PCAP formatted files.

### **Java Read Program**

The Java read program now displays NULLS (ESC FCT) when encountered.

### **Wireshark Plugin**

The wireshark plugins now display ESC FCT (NULL) and FCT frame data.

### **Spw program**

We now allow a spin up period in the average throughput calculations

### **SRS Configuration**

We have improved error reporting when configuring SRS routers.

# Software 34.86

## Release Notes

### **CentOS-8**

We now provide support for CentOS-8

### **Apple Mac 19.0.0**

We now provide support for MacOS Catalina (19.0.0)

### **API Change**

#### **EtherSpaceLink\_dump\_status**

This function provides a means of dumping given context to stderr for help in debugging applications.

# Software 34.83

## Release Notes

### **Wireshark / PCAP Integration**

When using packet aggregation with the final EOP being preserved (ESL\_PCAP\_RECORD\_EOP\_TS) we didn't preserve the final EOP if short timestamps were sent by the msr. This is now rectified.

### **API Change**

The EthersSpaceLink\_close, EthersSpaceLink\_shutdown, EthersSpaceLink\_abort calls now shutdown the spacewire ports automatically, user applications no longer have to.

### **Documentation**

The documentation for the 3U was missing, this is now included in the doc package.



Software 34.82

Release Notes

**Graphical Capture program**

When running capture recording to multiple files, the file size limit is now respected.

# Software 34.81

## Release Notes

### **Version Number change**

We have now added patch release numbers to the version identification.

The version numbering is now of the format

<ReleaseNumber>.<VersionNumber><PatchLevel>

The first patch to this version will be known as 34.82.

### **C++ RMAP**

The C++ API for RMAP has been corrected with respect to the endianness of the command byte, previously the bit positions of command flags were reversed causing corrupt RMAP messages.

### **READ program**

The java read program did not load the RMAP plugin correctly.

### **RMAP plugin**

Support for Read Modify Write verb has been added

### **MSR capture program**

The msr capture program places waveforms in the current directory if the environment variable ESL\_WF\_DIR is not set. ESL\_WF\_DIR is an environment variable that controls where waveform files are saved.

### **Waveform files**

When a waveform file is saved by the msr program, the name of the file is printed to stdout and the waveform file is prefixed with the epoch time. Allowing easy identification of the capture file in any logs.

### **SVG Waveform files**

These are now scaled correctly, so there are no longer have large blocks of unreadable color.

### **Python 3.7 Support on Windows**

Support for Python 3.7 on windows. If python 3.7 is required, set ESL\_PYTHON\_VER to 3.7 prior to running 4linksvars.

### **Python timecode example**

examples/python/timecode is an example of how to send and listen for Spacewire timecodes.

### **Graphical Capture program on Windows**

The graphical capture program uses GTK, however the GTK runtime ships with a DLL used by other packages and this can stop the program starting. The environment program 4linksvars.bat puts the GTK libraries in the path before other libraries.

### **Change of License for example programs**

All examples are now shipped with the BSD license making copying and reuse of the examples easier.

### **Graphical Capture program / Wireshark Integration**

It is now possible to capture directly into Wireshark from within the graphical capture program

### **Improved PCAP buffering**

The PCAP file buffer has been made an optimal size.

## API Changes

### Return Values

Throughout a return value of `-1` was used to indicate an error, this has been changed. If a function returns a value `< 0` an error has occurred, the returned value being the error code as described in `EtherSpaceLink_Constants.h`. This change does not affect the C++ or Python bindings as error handling is now by exception.

### `EtherSpaceLink_read_packet_full`

When called with `EtherSpaceLink_REPORT_SPECIAL_DATA` and `EtherSpaceLink_REPORT_EXTENSION_DATA` this function used to return `-2 - <number of bytes>`. This conflicts with the error handling as defined above. Instead the returned flags indicate whether a size is being returned. If the returned flags are `EtherSpaceLink_SPECIAL_SIZE` or `EtherSpaceLink_EXTENSION_SIZE` then the returned value is the number of special or extension bytes.

Example:

```
unsigned char rxbuf[4096];
int bytes_received, extension, ii;
int flags;
int active_port;

EtherSpaceLink esldev = EtherSpaceLink_open("1.1.1.1:1234");
if (!esldev)
{
    printf("Unable to connect\n");
    return 1;
}

bytes_received = EtherSpaceLink_read_packet_full ( esldev,
                                                rxbuf,
                                                sizeof(rxbuf),
                                                &flags,
                                                EtherSpaceLink_REPORT_EXTENSION_DATA
                                                | EtherSpaceLink_REPORT_SPECIAL_DATA
                                                );

if (bytes_received < 0)
{
    // Error condition
    fprintf (stderr, "Error %d \n", bytes_received);
    return 1;
}
switch (flags)
{
    case EtherSpaceLink_SPECIAL_SIZE:
    case EtherSpaceLink_EXTENSION_SIZE:
        fprintf (stdout, "%d bytes of special data\n", bytes_received);
}
}
```

## **receivePacket (java)**

This function no longer reports the size of extension or special data as a negative number. Instead one must check the context of the return value by calling **get\_terminator**. If this function returns **SPECIAL\_SIZE** or **EXTENSION\_SIZE** then the size returned is the extension size. Note that if **enable\_callbacks** is called the application developer does not have to perform analysis of special and extension\_data, and instead you get high level callbacks (e.g. link changed, error, or waveform data);

## **EtherspaceLink\_fastclose**

Introduced new API verb which shuts down the connection as soon as possible using SO\_LINGER with a timeout of 0.

### Language Implementations

C++ method fastclose

Java method fastclose

Python method fastclose

## **EtherspaceLink\_sendtimecode**

Explicit function to send a Spacewire timecode.

### Language Implementations

C++ method send\_timecode(timecode)

Java method send\_timecode(timecode)

Python method send\_timecode(timecode)

## **EtherspaceLink\_get\_error**

This returns the last error to occur in the **current thread** in the EtherSpaceLink library, mostly it should not be required to call this as functions return the error code directly. This change ensures that error codes are thread safe between receiver and transmit threads.

## **EtherspaceLink\_get\_error\_text**

This returns a string description of the error last error to occur in the **current thread**.

This change ensures that error codes are thread safe between receiver and transmit threads.

