AN-X2-AB-DRV Unscheduled Message and NS LED Update

The AN-X2-AB-DRV-04.v4.7.x.qtf and later firmware includes two functionality updates.

LED Ignore Options

In some cases, the AN-X module may receive errors from other racks or slave devices on the RIO network. These errors do not affect the operation of the AN-X and when they cause the NS LED to blink or stay RED, it can be a nuisance. Generally we expect only Abort and Noise errors in this category. Use discretion when disabling any NS LED error indications.

The diagnostic counters will always increment to indicate the given errors; there is no way to mask their behavior.

The following options may be added to the top of the main configuration file. Remove the semicolons to prevent the NS LED from turning red when the given error occurs.

```
; LedIgnNoise   ; Do not make NS LED Red for RIO Noise    Errors
; LedIgnAbort   ; Do not make NS LED Red for RIO Abort    Errors
; LedIgnCrc     ; Do not make NS LED Red for RIO CRC      Errors
; LedIgnTout    ; Do not make NS LED Red for RIO Timeout  Errors
; LedIgnPrcl    ; Do not make NS LED Red for RIO Protocol Errors
```

Current Unscheduled MSG

With the current Unscheduled MSG scheme the Service, Class, Instance, Attribute and Request Len are defined in the EthDev file for the drive.

The following example reads the Product Code from the Identity Object:

```
UnSchMsg         ; Start of Unscheduled MSG definition
Service   0x0e  ; CIP Service Code
Class     0x01  ; CIP Class
Instance  0x01  ; CIP Instance (0 if not defined)
Attribute 0x03  ; CIP Attribute (0 if not defined)
ReqLen      0   ; CIP Data length in bytes (BTW Ofs 0)
EndUnSchMsg     ; End of Unscheduled MSG definition
```

The data portion of the CIP Response is copied into BTR Ofs 0.

BT Defined Unscheduled MSG

With the BT Defined Unscheduled MSG scheme, the Service, Class, Instance, Attribute and Request Len are taken from Ofs 0 to 3 of the BTW Data. This allows multiple types of messages to be sent using separate block transfers.

It is defined in the EthDev files as follows:

```
UnSchMsgBT      ; Enable BT Defined Unscheduled MSG
```

The CIP MSG Request arguments are mapped as follows:

```
BTW[0] Low Byte - CIP Data Length in bytes (BTW Ofs 4)
BTW[0] Hi  Byte - CIP Service Code
BTW[1]          - CIP Class
BTW[2]          - CIP Instance
BTW[3]          - CIP Attribute
```

The CIP MSG Response arguments are mapped as follows:

```
BTR[0] - GenSTS (0xffff=Busy, 0x0000=NoError)
BTR[1] - ExtSTS if GenSTS!=0, RspDataLen if GenSTS=0
BTR[2..63] -Data Portion of CIP Response
```

Define BTW and BTR Locations

The BTW and BTR locations are usually defined in the main configuration file as follows:

```
Rack, 0o01,1,4
btw,  7, 0
btr,  7, 1
```

The Rack number is taken from the Rack definition preceding the BTW and BTR. BTW and BTR definitions are followed by I/OGroup and Module number.

The BTW length in the PLC must be long enough to include the data required to build the UnschdMSG.

For example, reading the ID object using BT Defined UnschdMSG, the BTW must be at least 4 words to get the CIP arguments to the AN-X.

Any CIP Data write data required has to be included as well. For example an Attribute write command that has 4 bytes of CIP data would require 2 extra words (6 words in total) for a BT Defined UnschdMSG.

The BTR length in the PLC must be long enough to read the CIP Status header for BT Defined UnschdMSG (2 words) plus the CIP data bytes in the reply.

In the example of reading the Identity Object, the return size is variable so to be safe we read extra data.

Debugging

A new keyword 'UnschMsgDebug' can be added to the top of the main config file to help debug UschdMSGs.

For example:

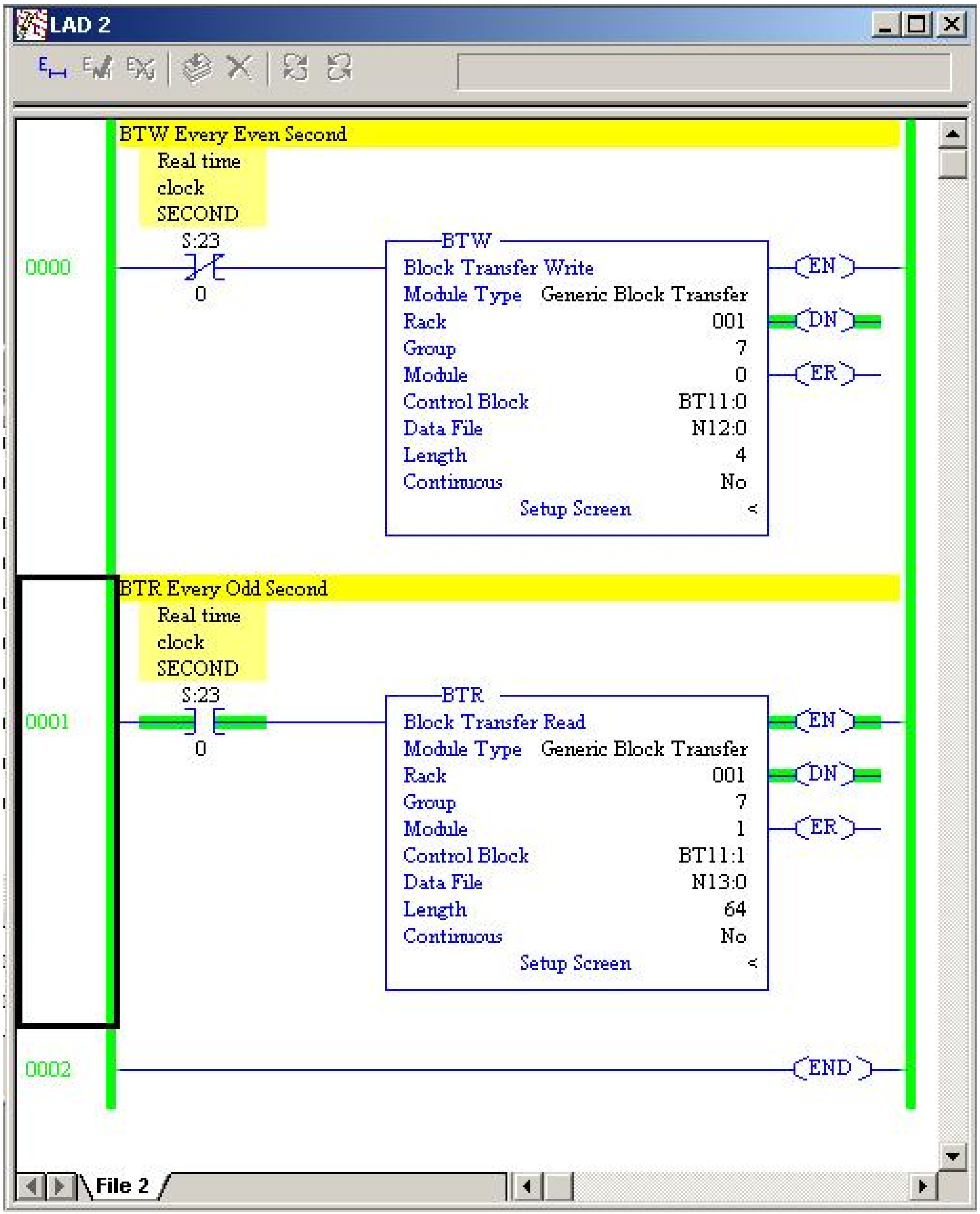
```
Baud, 230k
;MapReport
FaultRioRacks
UnschMsgDebug
```

With this option, the contents of the UnschdMSG Requests and Responses are shown in the Ethernet/IP log. This can be helpful for confirming exactly what the AN-X is sending and what we're getting back from the target.

For Example:

```
00:28.939 342 :UnSchd Req: 01 02 20 01 24 01
00:28.947 708 :UnSchd Rsp: 81 00 00 00 40 03 0c 00 - 51 00 01 01 01 00
01 00 00 00 0b 41 4e 2d 58 2d - 41 42 2d 53 43 41
00:30.650 270 :UnSchd Req: 01 02 20 01 24 01
00:30.658 620 :UnSchd Rsp: 81 00 00 00 40 03 0c 00 - 51 00 01 01 01 00
01 00 00 00 0b 41 4e 2d 58 2d - 41 42 2d 53 43 41
```

Sample PLC-5 Ladder



0000

BTW Every Even Second

Real time clock SECOND

S:23
0

BTW
Block Transfer Write
Module Type Generic Block Transfer
Rack 001
Group 7
Module 0
Control Block BT11:0
Data File N12:0
Length 4
Continuous No
Setup Screen <

(EN)
(DN)
(ER)

0001

BTR Every Odd Second

Real time clock SECOND

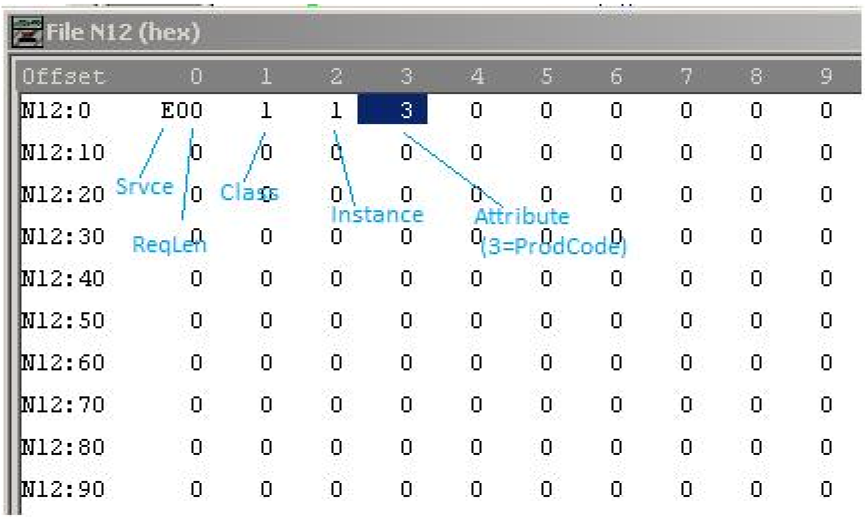
S:23
0

BTR
Block Transfer Read
Module Type Generic Block Transfer
Rack 001
Group 7
Module 1
Control Block BT11:1
Data File N13:0
Length 64
Continuous No
Setup Screen <

(EN)
(DN)
(ER)

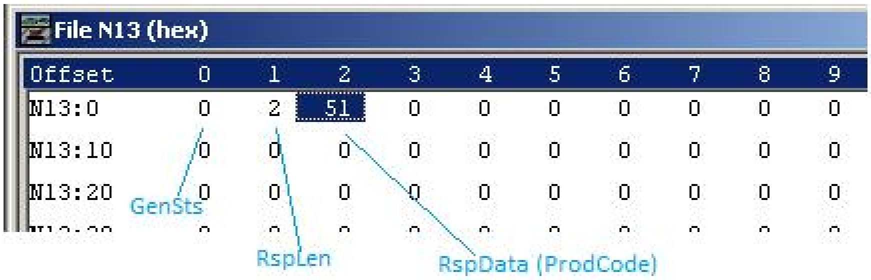
0002

(END)



File N12 (hex)

Offset	0	1	2	3	4	5	6	7	8	9
N12:0	E00	1	1	3	0	0	0	0	0	0
N12:10	0	0	0	0	0	0	0	0	0	0
N12:20	0	0	0	0	0	0	0	0	0	0
N12:30	0	0	0	0	0	0	0	0	0	0
N12:40	0	0	0	0	0	0	0	0	0	0
N12:50	0	0	0	0	0	0	0	0	0	0
N12:60	0	0	0	0	0	0	0	0	0	0
N12:70	0	0	0	0	0	0	0	0	0	0
N12:80	0	0	0	0	0	0	0	0	0	0
N12:90	0	0	0	0	0	0	0	0	0	0



File N13 (hex)

Offset	0	1	2	3	4	5	6	7	8	9
N13:0	0	2	51	0	0	0	0	0	0	0
N13:10	0	0	0	0	0	0	0	0	0	0
N13:20	0	0	0	0	0	0	0	0	0	0
N13:30	0	0	0	0	0	0	0	0	0	0