Chapter 47

Using DAS Studio 3 to configure the KAD/EBM/102

TEC/NOT/075

This paper discusses the following topics:

47.1 Module overview

The KAD/EBM/102 is a single-channel Ethernet bus monitor, which can parse and classify traffic using up to eight frame fields such as MAC addresses, IP addresses, and IP length with optional masking.

Like other Acra KAM-500 bus monitors, the data within the parsed Ethernet frames is available to be distributed to other modules via the KAM-500 backplane. It can identify and parse 254 unique Ethernet flows, where a flow represents a stream of Ethernet frames from a specific source to a specific destination. Flows are tagged with timestamps and packet counts.

The KAD/EBM/102 can be used to monitor, record, diagnose, and troubleshoot Ethernet network traffic. It can also be configured as an Ethernet-to-PCM bridge by means of the Multi Chassis Scheduler (MCS) software, which is integrated within the compiler of DAS Studio 3. The module can be programmed to operate in a promiscuous mode, which allows it to observe all traffic on the link, not only packets that are addressed to it.

Network management traffic, such as Address Resolution Protocol (ARP) packets and Internet Control Message Protocol (ICMP) packets, is optionally discarded or monitored via the catchall parser slot ID 254.

The KAD/EBM/102 supports Ethernet packet sizes up to 1,500 bytes and also features valid and error frame counters and error detection on Ethernet and IP layers. Connection speed is programmable to operate at 10BaseT, 100BaseTX, 1000BaseT or auto-negotiation.

NOTE: The KAD/EBM/101 is a similar module to the KAD/EBM/102 in terms of set up (see *TEC/NOT/046 - Using the KAD/EBM/101)*, however it is limited to 100BaseTX.

47.2 Ethernet-to-PCM bridge overview

The MCS is system scheduling software that enables you to transparently transmit parameters from any available Data Acquisition Unit (DAU) in a networked system by means of an Ethernet bus monitoring module such as the KAD/EBM/102 located in an Ethernet-to-bridge chassis. This system is referred to as Scenario 1 throughout this document.

The MCS is in charge of automatically creating intra-chassis Ethernet packets that are captured by the KAD/EBM/102.

The following diagram shows a typical MCS over Ethernet scenario where PCM is used for real-time telemetry in an Ethernet system.

In this example, parameter **P1** comes from a module placed on a remote **DAU 1** and it is being transmitted over the network , then **P1** gets parsed by the KAD/EBM/102 and, once available in the KAM-500 backplane of the Ethernet-to-PCM bridge chassis, transmitted over PCM using a KAD/ENC/106.

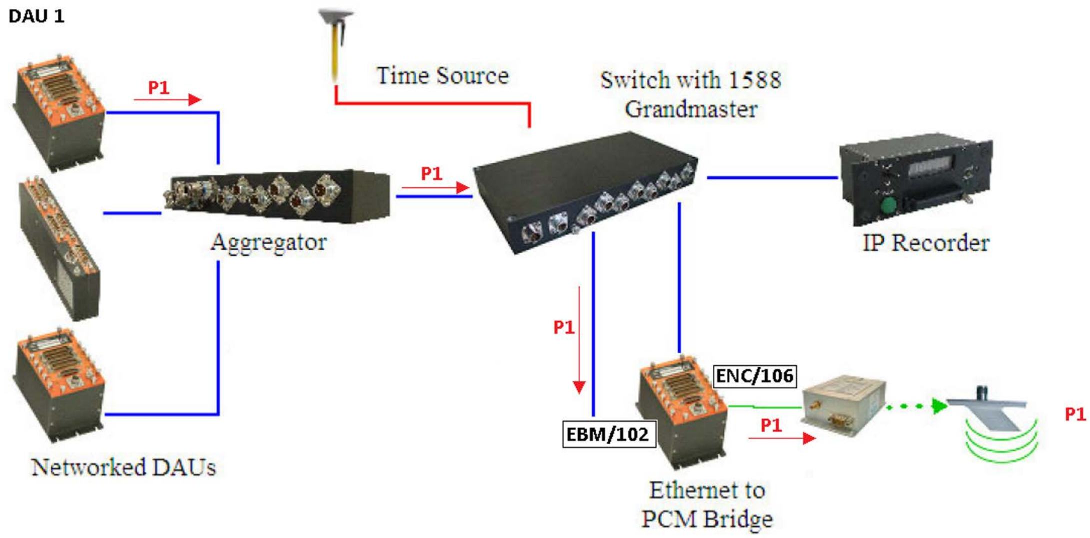
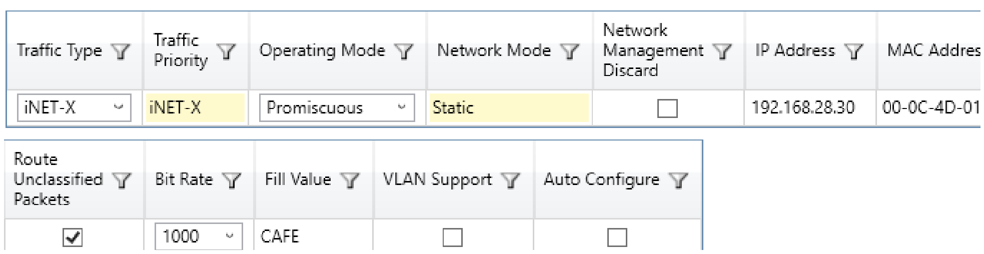


Figure 47-1: Example of MCS traffic showing parameter P1 from remote DAU 1

47.3 KAD/EBM/102 DAS Studio 3 settings

This technical note describes three scenarios (referenced as: *Scenario 1*; *Scenario 2*; *Scenario 3*) where DAS Studio 3 can be used to configure the KAD/EBM/102. Depending on the scenario, some of the default settings shown in the following table need to be modified.



Traffic Type	Traffic Priority	Operating Mode	Network Mode	Network Management Discard	IP Address	MAC Addres
iNET-X	iNET-X	Promiscuous	Static	☐	192.168.28.30	00-0C-4D-01

Route Unclassified Packets	Bit Rate	Fill Value	VLAN Support	Auto Configure
☑	1000	CAFE	☐	☐

Figure 47-2: DAS Studio 3 Settings tab for KAD/EBM/102

Table 47-1: KAD/EBM/102 settings in DAS Studio 3

Setting	Notes
Traffic Type	This setting defines the type of packet expected when configuring the KAD/EBM/102 as a bus monitor in different scenarios. In an Ethernet-to-PCM stream scenario (MCS), select **iNET-X** or **IENA**. (Scenario 1) In a generic Ethernet bus monitor scenario, select **Generic**. (Scenario 2) Select **All** when parsing both MCS packets and generic Ethernet packets. (Scenario 3) **NOTE:** Select either iNET-X or IENA. Do not mix traffic types; if you decide to use iNET-X packets for all other traffic, then the transport packets must also be iNET-X. **NOTE:** The MCS builds packets to transport parameters from DAUs to the PCM DAU at user-defined rates. It is necessary to separate MCS Ethernet packets from non-PCM bridge traffic (for more information, see “47.5.2 Filtering Ethernet packets” on page 5).

Table 47-1: KAD/EBM/102 settings in DAS Studio 3 (continued)

Setting	Notes
Traffic Priority	This advanced option is only available when parsing both MCS packets and generic Ethernet packets (Scenario 3). It defines the processing priority for package types when filtering rules overlap.
Operating Mode	By setting **Promiscuous**, the KAD/EBM/102 parses all traffic, even if not directed specifically at the KAD/EBM/102. That is, it parses all traffic regardless of the MAC address destinations. If you select **Non-promiscuous**, the KAD/EBM/102 parses only those Ethernet packets with a Destination IP address and a Destination MAC address matching those defined for the KAD/EBM/102. All other traffic is discarded, including broadcast traffic. **NOTE:** Promiscuous mode is the recommended setting as it covers the most common scenarios since users generally monitor traffic going to other devices.
Network Mode	Only Static IP addresses are supported by KAD/EBM/10x modules. This setting can not be modified. **NOTE:** Protocols for automating the task of assigning IP addresses such as Dynamic Host Configuration Protocol (DHCP) are not supported .
Network Management Discard	This setting enables or disables network management commands ARP and PING. When this check box is selected, all broadcast ARP/PING requests are discarded. When this check box is cleared, ARP/PING requests are processed and a response is sent back by the module. The default and recommended setting is to have the check box cleared allowing you to ping the KAD/EBM/102.
IP Address	Allows you to specify a unique IP Address for the KAD/EBM/102. **NOTE:** The KAD/EBM/102 requires an IP address primarily to work in non-promiscuous mode, that is, when data is being sent directly to it.
MAC Address	Allows you to specify a unique MAC Address for the KAD/EBM/102.The first three bytes for Curtiss-Wright MAC addresses are 00-0C-4D.
Route Unclassified Packets	When enabled, all unclassified traffic is routed to the Catchall-Parser (slot ID 254) By default, this check box is selected.
Bit Rate	The following speed options are available: 10, 100, 1000, or Auto-negotiate. The main factor in selecting one of these options is the speed of the output port of the connecting switch. We recommend using 1000 (1000BaseT) whenever possible, as this offers the greatest Ethernet bandwidth.
Fill Value	The default value is 0xAAAA. To assist with identifying Ethernet connection issues, we recommend using a readily identifiable hex word such as 0xCAFE.
VLAN Support	This is an advanced option. Enabling VLAN Support allows parser slots—when present—to skip over VLAN packet headers, allowing you to classify traffic on the basis of IP and UDP/TCP packet header fields. Refer to “47.8.10 VLAN” on page 16 for further information. By default, this check box is cleared.
Auto Configure	This option must be checked when the KAD/EBM/10x is required to parse both generic and MCS packets simultaneously. This is explained in Scenario 3 in this document and requires the **Traffic Type** field to be set to **All**. The default setting of the Traffic Priority field can be changed.

For further information, refer to the latest *KAD/EBM/102* data sheet.

47.4 Using DAS Studio 3 to set up configuration scenarios

DAS Studio 3 is used to create a configuration, which contains the various elements, which make up your data acquisition system. You then use this configuration file to manage and program these elements.

To see how hardware is represented in the DAS Studio 3 graphical user interface, see Figure 1 in the *DAS Studio 3 User Manual*.

47.5 Scenario 1: using the KAD/EBM/102 to transmit Ethernet data to PCM

As described in "47.2 Ethernet-to-PCM bridge overview" on page 1, in Ethernet systems where every DAU has an Ethernet controller and transmits data to a central switch, the KAD/EBM/102 can act as a bridge module for transmitting Ethernet-captured data into a Pulse Code Modulation (PCM) stream.

In this scenario, the KAD/EBM/102 is connected to the output of an aggregating switch, through which data from all other DAUs is available to the PCM DAU.

This scenario is illustrated in the following figure where two DAUs (DAU_0 and DAU_1) are connected to a network switch; and a PCM DAU (containing a KAD/EBM/102) is connected between the output of the network switch and a PCM transmitter. Connecting the system in this way makes all DAU parameters available to the PCM stream transmitted by the KAD/ENC/106.

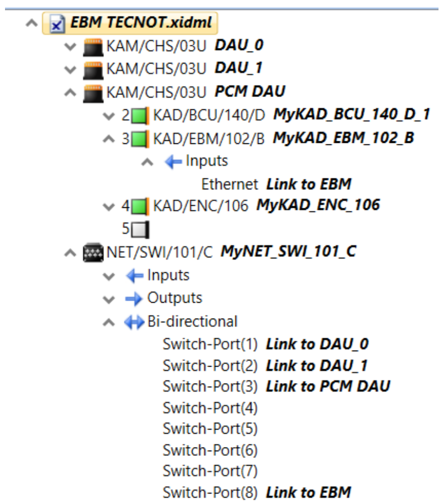


Figure 47-3: Scenario 1 – Ethernet to PCM bridge (MCS)

NOTE: This example assumes the network switch is correctly configured to route packets from a remote DAU to the KAD/EBM/102.

47.5.1 Configuring the KAD/EBM/102 to transmit Ethernet data into a PCM stream

The following figure shows the settings when the KAD/EBM/102 has been configured to act as a bridge module for transmitting Ethernet data and Traffic Type is set to iNET-X or IENA.

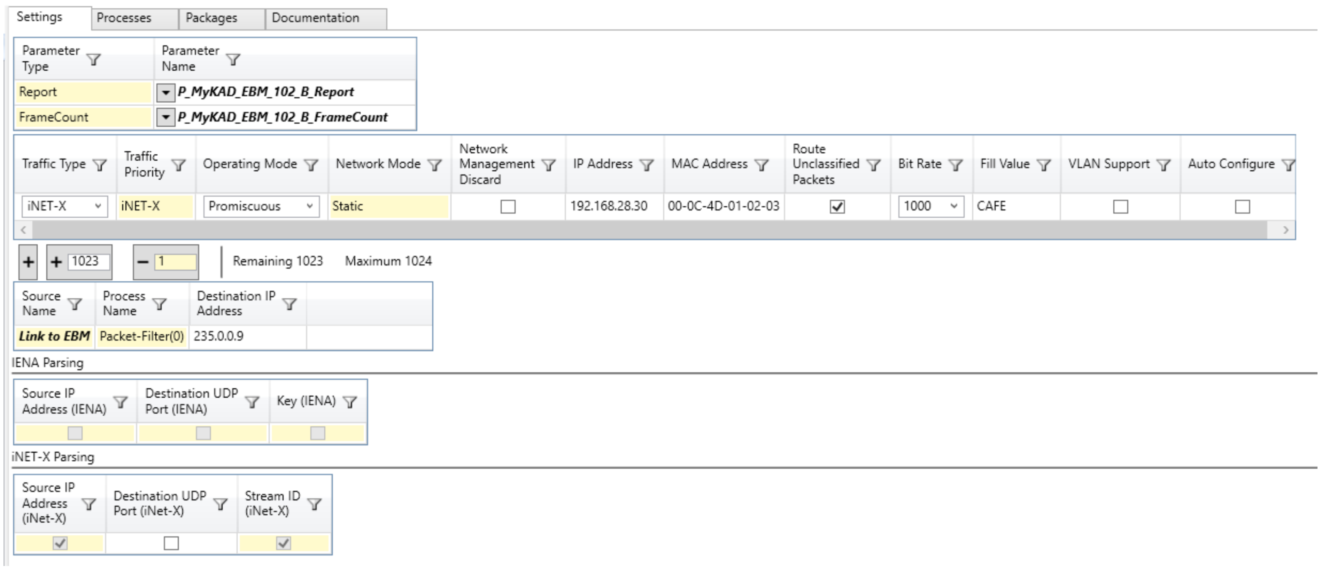


Figure 47-4: DAS Studio 3 settings when configuring the KAD/EBM/102 for MCS

The following settings need to be modified from the defaults.

Table 47-2: Settings for Scenario 1

Setting	Notes
Traffic Type	Select **iNET-X** or **IENA**.
Operating Mode	Select **Promiscuous**.
IP Address	Change to a unique IP address in the system.
MAC Address	Change to a unique MAC address in the system starting as 00-0C-4D.
Bit Rate	Optional. Can be changed to Auto-negotiate if required.
Fill Value	Optional. Recommended to use a readily identifiable hex word such as 0xCAFE.

47.5.2 Filtering Ethernet packets

Traffic from all connected DAUs is available to the KAD/EBM/102, so it is likely that non-PCM bridge traffic is also flowing across the same link. Therefore, the KAD/EBM/102 must be able to identify MCS packets. This is achieved by creating a filter for each KAD/EBM/102 input. To create such a filter, do the following.

1. Select the **KAD/EBM/102** in the Navigator pane and then click the **Settings** tab.
2. In the **Process Name** column, click the + button beside **<Create a new 'Packet-Filter' process on Ethernet>**.
   A **Packet-Filter** setting appears.
3. In the **Destination IP Address** field, insert a value for a multicast IP address.
   In the above example the Multicast IP address used is 235.0.0.9.
   NOTE: MCS also supports unicast packets for which the destination is the KAD/EBM/102 itself. In this case, in the **Destination IP Address** field, enter the KAD/EBM/102 IP address. So for the example above, type **192.168.28.30**.

47.5.3 Defining Ethernet packets to parse

In DAS Studio 3.4.16 onwards, when selecting the traffic type of the MCS packets, the settings which enable the KAD/EBM/102 to identify Ethernet packets for parsing are set automatically.

When Traffic Type IENA is selected, the IENA Parsing pane automatically selects Source IP Address (IENA) and Key (IENA).

When Traffic Type iNET-X is selected, the iNET-X Parsing pane automatically selects Source IP Address (iNET-X) and Stream ID (iNET-X).

NOTE: Select the **Destination UDP Port** check box if an Ethernet controller module is also used in the Ethernet system as the PTP Grandmaster. The reason is the Grandmaster generates PTP packets (UDP/IP packets), which could be parsed inadvertently by the KAD/EBM/102. Selecting the Destination UDP Port check box prevents from this happening.

47.5.4 Generating MCS packets

You can proceed to build a PCM frame once the routing of the switch (for example NET/SWI/101/C) and the module settings are configured correctly in DAS Studio 3. Once the PCM is created, click **Verify**; MCS packets are automatically built by DAS Studio 3 at the data source DAUs. This ensures that the data words to be sent over real-time PCM arrive at the data selector chassis in time for transmission. Refer to "47.8.12 Step-by-step instructions for generation of MCS packets (Scenario 1)" on page 17 for details on how MCS packets are automatically built.

47.6 Scenario 2: using the KAD/EBM/102 as a generic Ethernet parser

The KAD/EBM/102 can also be used as an Ethernet bus monitor to parse traffic from external sources, thereby making it available to the Acra KAM-500 backplane for use in other modules. In this scenario, all traffic to be parsed by the KAD/EBM/102 is considered generic, therefore you must use the Ethernet Builder application to parse Ethernet traffic.

For the KAD/EBM/102 to parse generic Ethernet traffic, you must set the **Traffic Type** field to **Generic** as shown in the following figure.

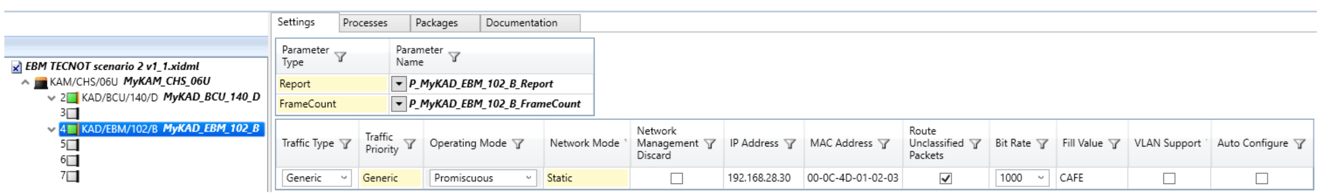


Figure 47-5: Scenario 2 - KAD/EBM/102 acting as a generic Ethernet parser in DAS Studio 3

The following settings need to be modified from the defaults.

Table 47-3: Settings for Scenario 2

Setting	Notes
Traffic Type	Select **Generic.**
IP Address	Change to a unique IP address in the system.
MAC Address	Change to a unique MAC address in the system starting as 00-0C-4D.
Bit Rate	Optional. Can be changed to Auto-negotiate if required.
Fill Value	Optional. Recommended to use a readily identifiable hex word such as 0xCAFE.
VLAN Support	Optional. Depending on application.

47.6.1 Using Ethernet Builder to configure the KAD/EBM/102

The Ethernet Builder application adds and configures Ethernet messages and parameters on Ethernet bus monitor modules, such as the KAD/EBM/102.

This section features a worked example of how to use Ethernet Builder to configure the KAD/EBM/102 to parse an Ethernet packet with a source IP of 10.11.12.13, a destination IP of 233.10.11.45, and a destination port of 1024.

In this worked example, words 100, 102, 104, and 106 are extracted from the parsed packet.

Knowing that the Word Offset is defined as per the example in the following table, Word Offset 0 is the start of the Ethernet Frame and is therefore the first 16 bits of the Destination MAC address.

Table 47-4: iNET-X, IENA, and VLAN word offset

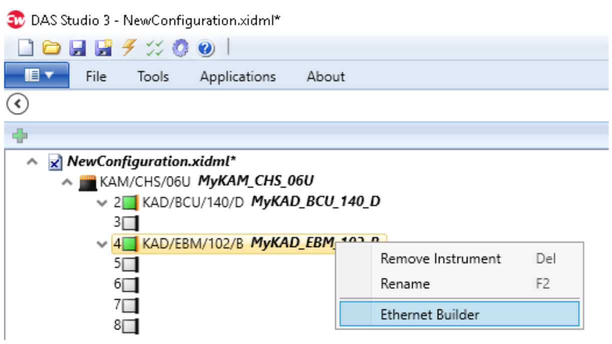
Field name	Values of offset index words	
	Offset If VLAN disabled	**Offset If VLAN enabled**
MAC header		
Destination MAC address, word 0 (MSW)	0	0
Destination MAC address, word 1	1	1
Destination MAC address, word 2 (LSW)	2	2
Source MAC address, word 0 (MSW)	3	3
Source MAC address, word 1	4	4
Source MAC address, word 2 (LSW)	5	5
Frame/Protocol type; 0x0*0 e.g. IP = 0800h	6	8
IP header		
VLAN priority/ID	N/A	7
IP version/IHL/ToS	7	9
IP packet size	8	10
IP ID	9	11
IP flags and fragment offset	10	12
IP TTL and Protocol; xx11h for UDP and xx06h for TCP	11	13
IP header checksum	12	14
Source IP address, word 0 (MSW)	13	15
Source IP address, word 1 (LSW)	14	16
Destination IP address, word 0 (MSW)	15	17
Destination IP address, word 1 (LSW)	16	18
Source port no.	17	19
Destination port no.	18	20
UDP Length; TCP seq word 0	19	21
UDP CSum; TCP seq word 1	20	22
Data payload		
IENA key or iNET Control Field 0 // TCP Ack word 0	21	23
IENA size or iNET Control Field 1 // TCP Ack word 1	22	24
iNET-X Stream ID word 0 or IENA date 0 // TCP offset/flags	23	25
iNET-X Stream ID word 1 or IENA date 1// TCP window	24	26

Table 47-4: iNET-X, IENA, and VLAN word offset (continued)

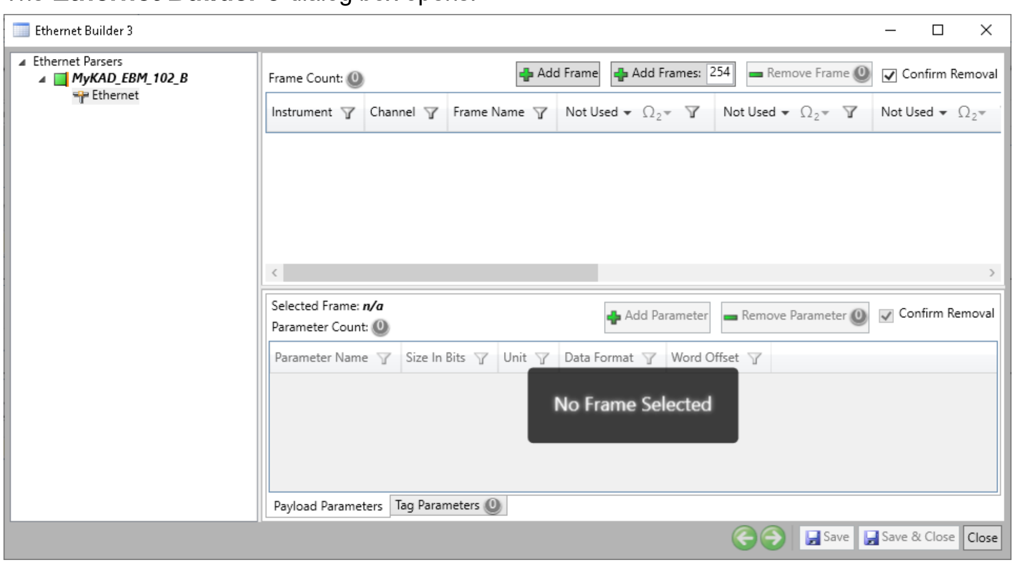
Field name	Values of offset index words	
	Offset If VLAN disabled	Offset If VLAN enabled
iNET-X Seq Number 0 or IENA date 2 // TCP Csum	25	27
iNET-X Seq Number 1 or IENA status // TCP urgent Ptr	26	28
iNET-X Pckt Length 0 or IENA Seq // TCP Options (or TCP Data 0)	27	29

To parse the Ethernet frame, do the following:

1. Before running this application, ensure that the KAD/EBM/102 has been added to the configuration, and that **Traffic Type** is set to **Generic** (see “47.6 Scenario 2: using the KAD/EBM/102 as a generic Ethernet parser” on page 6). Otherwise, a “No Supported Ethernet Parsers Found” message is displayed when you try to launch the Ethernet Builder application.
2. Right-click the KAD/EBM/102 module in the Navigator and then click **Ethernet Builder**. (You can also click the **Applications** menu and then click **Ethernet Builder**.)

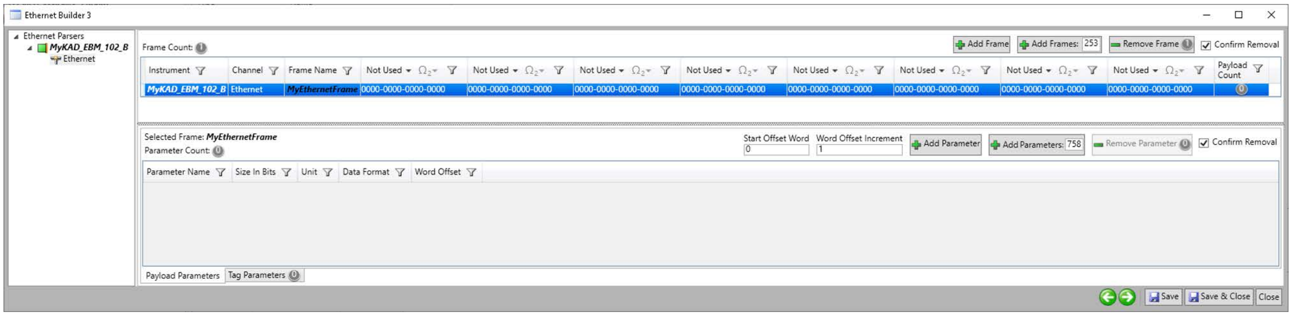


The **Ethernet Builder 3** dialog box opens.



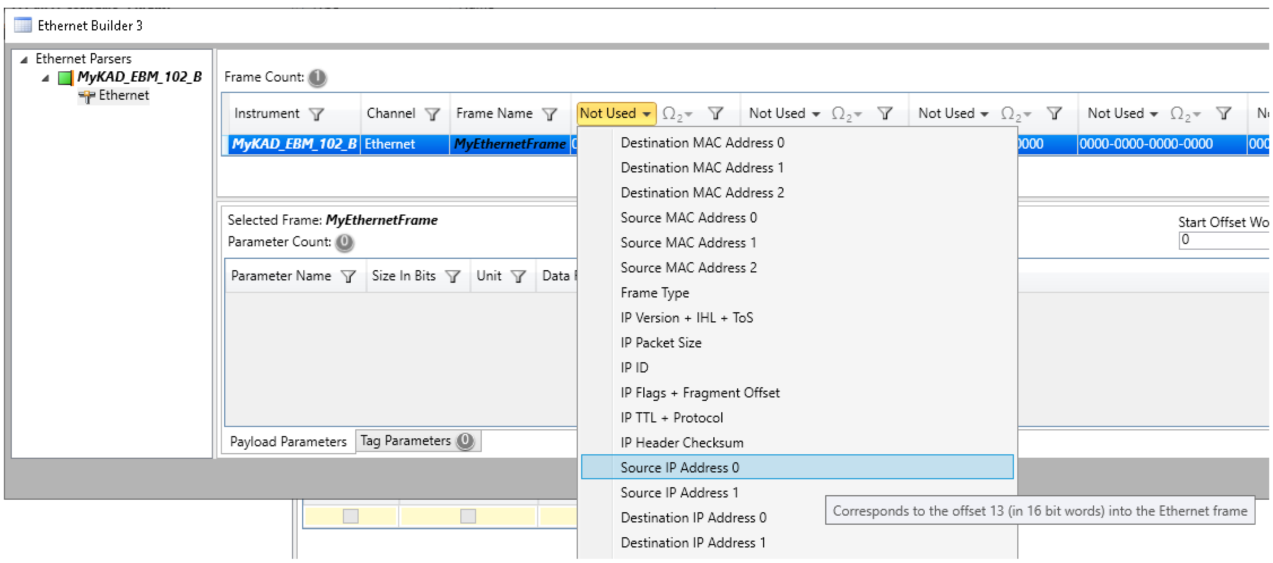
For dialog box navigation, see “Builder application GUI overview” in the *DAS Studio 3 User Manual*.

3. Add a single frame to the Ethernet link by selecting **Ethernet** in the Navigator pane and clicking **Add Frame**.

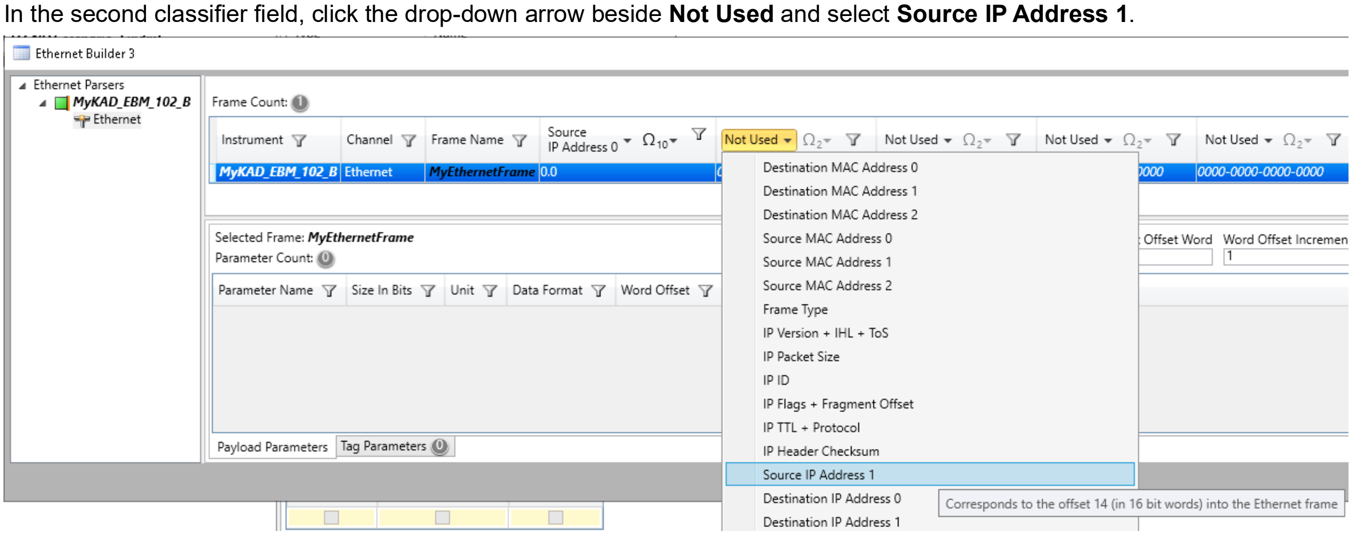


NOTE: In the **Frame Count** pane, there are eight 16-bit user-definable classifier fields (by default, these fields are labeled **Not Used**), which can be used to create the rules which help the KAD/EBM/102 identify Ethernet packets.
As the fields to be used for source IP and destination IP are both two 16-bit fields, two classifier rules for both the source IP and the destination IP are required.

4. In the first classifier field, click the drop-down arrow beside **Not Used** and select **Source IP Address 0**.



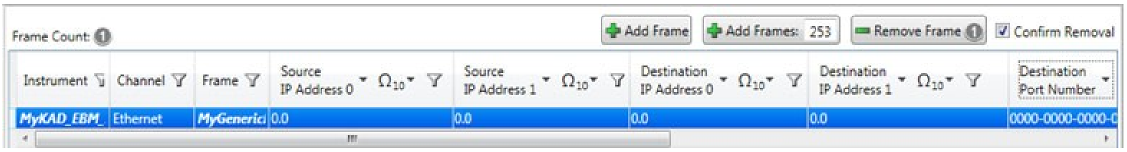
5. In the second classifier field, click the drop-down arrow beside **Not Used** and select **Source IP Address 1**.



NOTE: When Source IP Address 0 is in use, it is not available in the drop-down list.

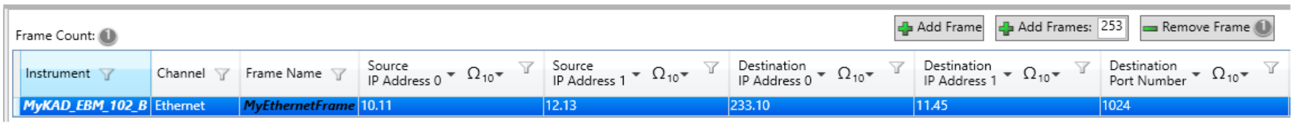
6. Set the remaining classifier fields by selecting **Destination IP** for the third and fourth classifier fields, and selecting **Destination Port Number** for the fifth classifier field.

Definitions for the required classifier fields are now set.



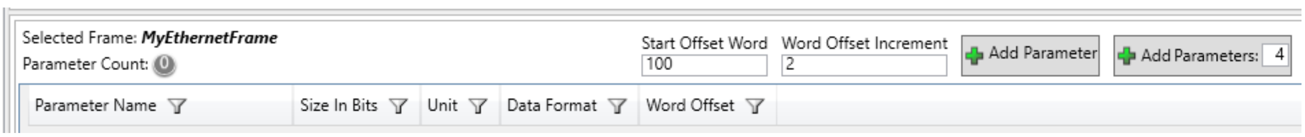
NOTE: Values for these classifier fields can be formatted using Decimal, Hex, Octal, or Binary format. To change a format, click the drop-down arrow by the Ω symbol for each classifier field and choose the desired format. In our working example, Decimal format suffices.

7. In the **Source IP Address 0** field, type **10.11**.
8. In the **Source IP Address 1** field, type **12.13**.
9. In the **Destination IP Address 0** field, type **233.10**.
10. In the **Destination IP Address 1** field, type **11.45**.
11. In the **Destination Port** field, type **1024**.
    Values for the classifiers fields are now set.

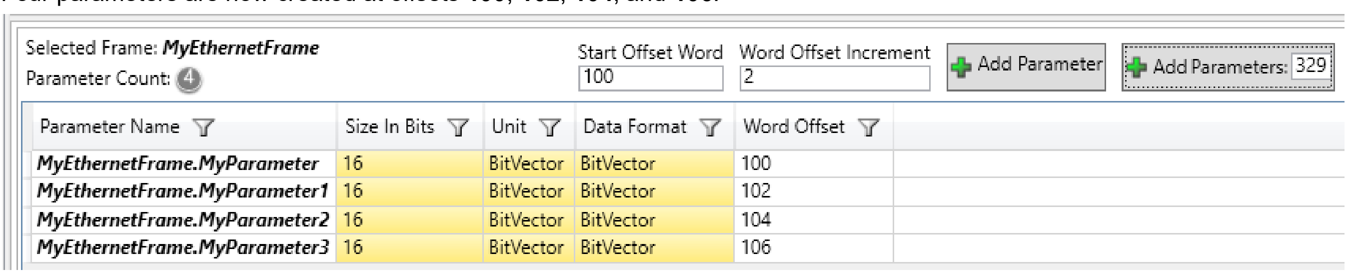


NOTE: Now that parsing classifiers have been defined, add the parameters to be extracted from the packet; specifically, words 100, 102, 104, and 106.

12. In the **Start Offset Word** field, type **100**.
13. In the **Word Offset Increment** field, type **2**.
14. In the field beside **Add Parameters**, type **4**.



15. Click **Add Parameters**.
    Four parameters are now created at offsets **100**, **102**, **104**, and **106**.



16. To save changes, click **Save & Close**.

47.7 Scenario 3: KAD/EBM/102 setup to parse Generic and MCS packets simultaneously

In this scenario the KAD/EBM/102 works as both MCS parser (Scenario 1) and generic Ethernet parser (Scenario 2).

The **Auto Configure** check box must be selected and **Traffic Type** must be set to **All** as shown in the following figure.

NOTE: The **Traffic Priority** field has to be set accordingly if the filtering rules overlap, which could happen in the unlikely case of having non-unique parsing rules on generic packets and IENA/iNET-X MCS packets.

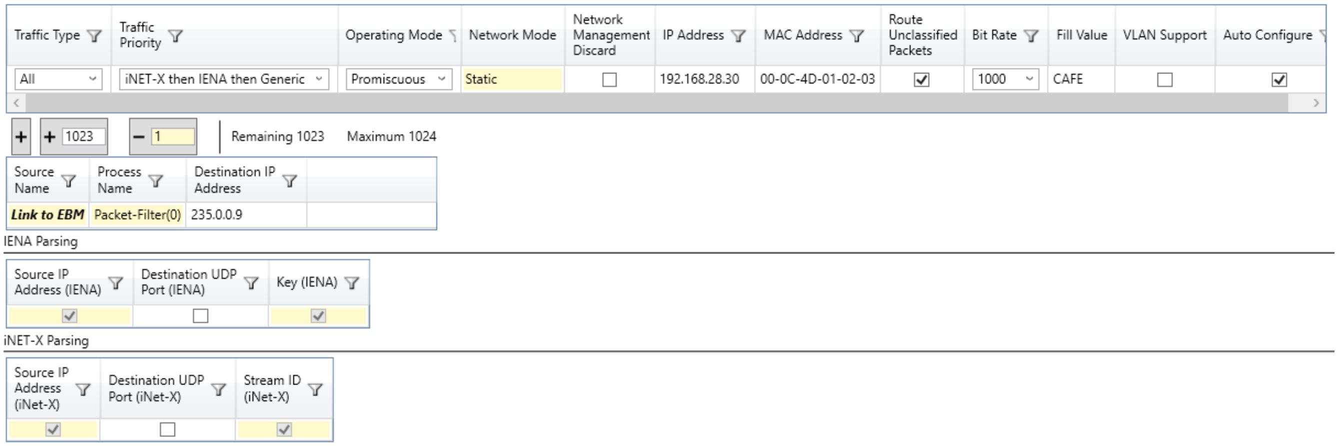


Figure 47-6: Setup Scenario 3 - KAD/EBM/102 acting as both MCS and generic Ethernet parser in DAS Studio 3

The following settings need to be modified from the defaults.

Table 47-5: Settings for Scenario 3

Setting	Notes
Traffic Type	Select **All**.
Traffic Priority	Filtering rules overlap. Set depending on application. Leave as default if unsure.
IP Address	Change to a unique IP address in the system.
MAC Address	Change to a unique MAC address in the system starting as 00-0C-4D.
Bit Rate	Optional. Can be changed to Auto-negotiate if required.
Fill Value	Optional. Recommended to use a readily identifiable hex word such as 0xCAFE.
VLAN Support	Optional. Depending on application.
Auto Configure	This option must be checked.

To add MCS packets, see “47.5 Scenario 1: using the KAD/EBM/102 to transmit Ethernet data to PCM” on page 4.

To add generic packets, see “47.6 Scenario 2: using the KAD/EBM/102 as a generic Ethernet parser” on page 6.

IMPORTANT: Once the parsing rules have been defined using Ethernet Builder, the parameters are only visible to an external sink such as PCM or visible again in Ethernet Builder after verification. This is because the MCS algorithm adds an internal field to the generic packet to prevent this packet from being filtered; this is done using the Packet-Filter Destination IP Address. See “47.8.13 Step-by-step instructions for adding generic parser rules (Scenario 3)” on page 19.

NOTE: It is possible to import parsing rules from another XidML file, however the new parsing rules only show up after verification.

47.8 Appendix

47.8.1 ARINC 664 Part 7 (A664P7) monitoring

The KAD/EBM/102 is an Ethernet packet parser and can therefore monitor ARINC 664 Part 7 (ARINC-664P7) messages, however the packets to be parsed must not be transmitted in burst and the parameters must be in a fixed location in the Ethernet packet. The KAD/EBM/103 can parse specific Rockwell Collins ARINC-664P7 messages and the KAD/EBM/104 can parse specific ARINC-664P7 messages from GE aviation.

A KAD/ARR/101 may be used to check the sequencing and remove the redundancy between the two source networks.

47.8.2 Triple buffer

For generic packets, the module uses the usual concept of triple buffer as per all Acra KAM-500 parser bus monitors. The following figure illustrates the triple buffering of data words (green) and time message tags (white) in the KAD/EBM/102 parser.

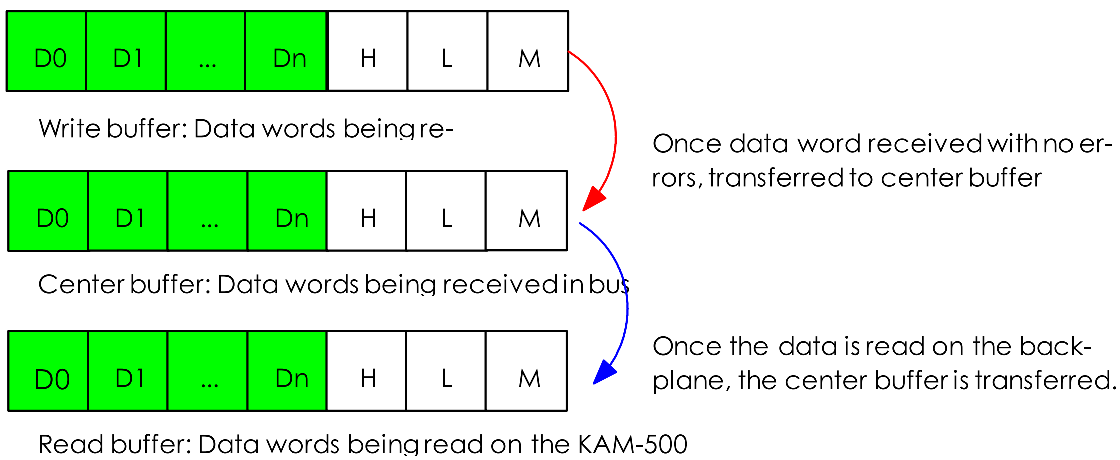


Figure 47-7: Triple buffering of traffic and associated message tags

D0, D1, D2, Dn in the previous figure corresponds to the Ethernet traffic data words with n < 758.

The time tags H, L, M correspond to High time, Low time, and Micro time—which is the time of the start of the first received bit of the message with a 1-µsec resolution.

The way triple buffering works is as follows:

Time message tags are added to each message received and stored in separate buffers. As soon as a message is received with no errors, the contents of the write buffer is transferred to the center buffer. If the data in the center buffer has not been transferred to a read buffer, a skipped flag is set.

As soon as the last parameter of interest has been read from the buffer being read by the backplane, the contents of the center buffer (if new) are transferred to the read buffer. If no new data word has been received, the stale flag is set. A center and read buffer exist for every message ID (parser slot). Skipped and stale bits can be found in the Message Info register to indicate whether messages are lost or repeated (undersampling or oversampling situations).

Additional tags such as a Message Count, Message Size, Message Status and Message Info registers are also available as additional information and can be added from the Ethernet Builder application as explained in “47.8.11 Using wildcards and tags in Ethernet Builder” on page 17. For further information regarding these registers, refer to the *KAD/EBM/102* data sheet.

NOTE: For MCS operation, the KAD/EBM/102 uses a single buffer operation.

47.8.3 Burst packets

The KAD/EBM/102 parser is triple buffered, therefore if a burst of more than two packets to be parsed is received, the triple buffer gets saturated and packets are skipped.

In a situation where burst traffic is expected and all messages are required to be parsed, then the parser data must be sampled at a greater rate than the burst rate. This is effectively an oversampling situation and the sampling rate must be set at the maximum expected frequency of the burst packet. In this scenario, the KAD/EBM/102 MessageInfo register generates lots of stale (repeated) packets when the incoming traffic is steady.

47.8.4 Fragmented packets

The KAD/EBM/102 supports fragmented packets, however it does not re-assemble the packet.

For classification, the Fragmentation Flags/Offset described in the Internet Protocol RFC, needs to be included.

As shown in the following figure, the Fragment Offset is a 13-bit number (13 LSBs of the 16-bit field) present in the IP header of the packet. Where the first received fragment has an offset of 0, the second fragment specifies an offset equal to the number of bytes in the first fragment divided by 8; the nth fragment specifies an offset equal to the sum of the number of bytes in the preceding n-1 packets divided by 8.

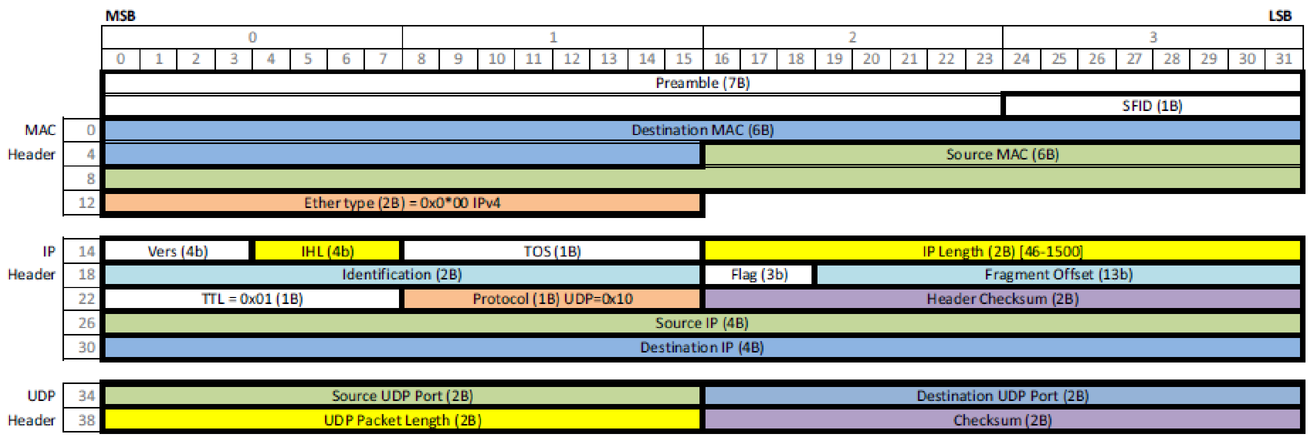


Figure 47-8: MAC/IP/UDP header showing Fragmentation Flags/Offset

Each fragment of an incoming packet must be treated as a separate flow of Ethernet frames, where the unique identifier for each flow is the IP fragmentation offset. For example, if you have a 64-k IP packet, fragmented into 1-k fragments, then 64 classifier slots for that IP packet must be set. The offset field is required to be set in the classifier fields in Ethernet Builder as **IP Flags + Fragment Offset** as shown in the following figure.

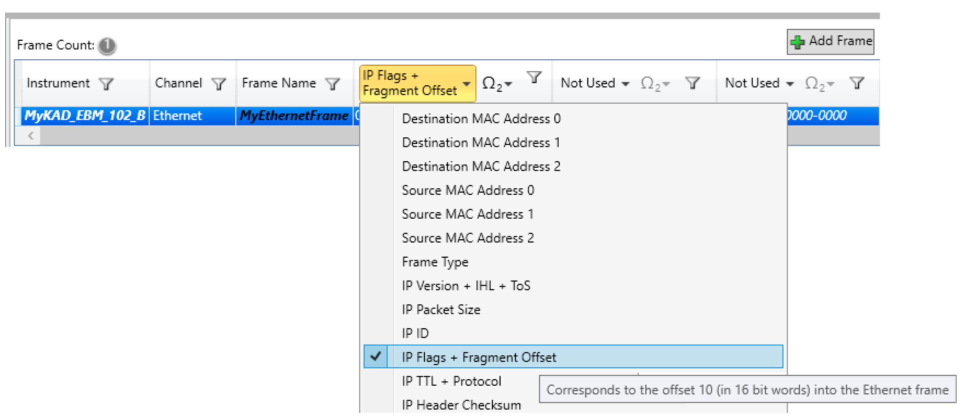


Figure 47-9: Ethernet builder showing classification to select to parse fragmented packets

47.8.5 Parsing non-fixed length Ethernet packets

Because the packet length is not fixed, you must specify the maximum data words (which is 758, the maximum for an Ethernet packet). You can decrease the number of data words if you know the maximum output from the device you are monitoring.

Discard the last data words if the packet length is not at its maximum. The data word 8 reports the IP size packet and can be used to discard the last data words after post-processing on the ground.

Table 47-6: Values of Offset Index Words

FIELD NAME	Values of Offset Index Words	
	OFFSET IF VLAN DISABLED	OFFSET IF VLAN ENABLED
Destination MAC address, word 0 (MSW)	0	0
Destination MAC address, word 1	1	1
Destination MAC address, word 2 (LSW)	2	2
Source MAC address, word 0 (MSW)	3	3
Source MAC address, word 1	4	4
Source MAC address, word 2 (LSW)	5	5
Frame / Protocol type ; 0x0*0 e.g. IP = 0800h	6	8
VLAN priority/ID	N/A	7
IP version/IHL/ToS	7	9
IP packet size	8	10

For example, if the packet is 700 data words, discard the last 58 data words because they will be random or previous data stored in RAM.

47.8.6 Parsing parameters with multiple occurrences in an Ethernet packet

The KAD/EBM/102 parses an Ethernet packet into 16-bit words. The KAD/EBM/102 parses an incoming Ethernet packet into 16-bit words, that is, the module splits the incoming Ethernet frames into N standalone 2-byte registers. The module's logic processes these registers as independent blocks of data without taking into account that they may be part of multiple words/occurrences. It is up to the real time or post-processing software to establish the relationship within the gathered data.

The following figure illustrates an example of a parameter P1 composed of two words in the same packet, which is a unique type of packet coming from an Ethernet device source IP 10.0.0.1. The first instance 1-P1 is at word offset 30; the second instance is at word offset 33.

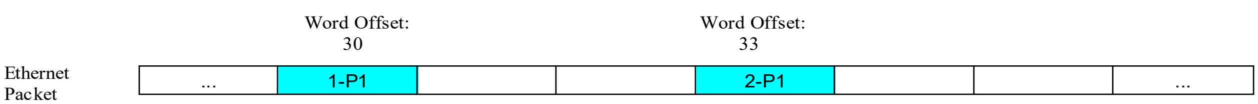


Figure 47-10: Ethernet packet showing a parameter with 2 words

Two distinct data words need to be set up in the Ethernet builder as shown in the following figure.

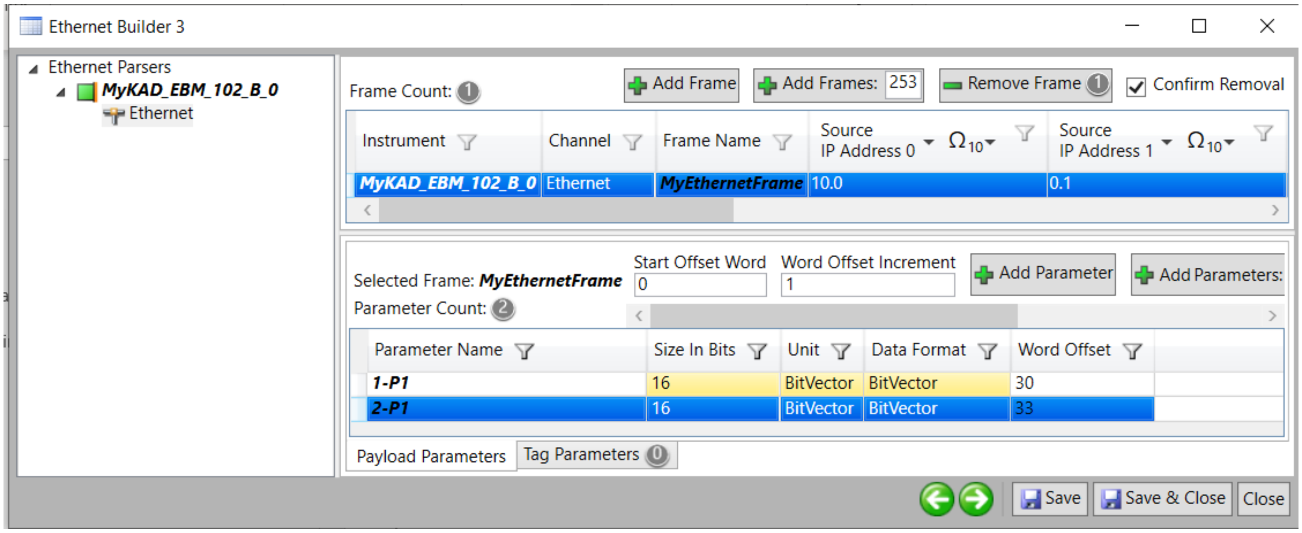


Figure 47-11: Ethernet builder showing that the 2 words must be distinctly defined

If the parameter needs to be placed, for example in PCM, the two samples composing the parameter P1 need to be commutated accordingly to match the PCM frame. In the following example it is set at 1:2.

SFID							Syncword
SFID					1-P1		Syncword
SFID							Syncword
SFID					2 -P1		Syncword

Figure 47-12: PCM showing the two samples composing the parameter P1

For the software decommutating the PCM, such as GS Works, a mimic PCM is required with one single parameter at 1:2; this indicates to the decom software that the two samples are coming from a single parameter as shown in the following figure.

SFID							Syncword
SFID					P1		Syncword
SFID							Syncword
SFID					P1		Syncword

Figure 47-13: Mimic PCM indicating to decom software that two samples belong to the same parameter P1

47.8.7 Latency

For a generic packet, the KAD/EBM/102 uses the triple buffer, therefore parameters extracted from it are 2 to 3 samples late. So for a packet at 100 Hz, the latency for a parameter extracted from it is between 20 ms to 30 ms.

For the MCS with a PCM, DAS Studio 3 delays the PCM in order to take into account all the delays associated with the different Ethernet devices. This latency is specified in the following PCM package definition in the XidML file. This latency does not take into account the delays associated with specific sources such as the filter delay on an analog module or the delay of the PCM decom.

```
<IRIG-106-Ch-4 Name="MyIRIG-106-Ch-4Package_4_256x16">
    <PackageRate>64</PackageRate>
    <Sequencing>
      <Offset_uS>7884.97924804688</Offset_uS>
    </Sequencing>
```

47.8.8 Auto-negotiation

Auto-negotiation can cause an interoperability problem. Two Ethernet devices linked to each other shall either be:

Both Auto

Both Force 100 Mbps

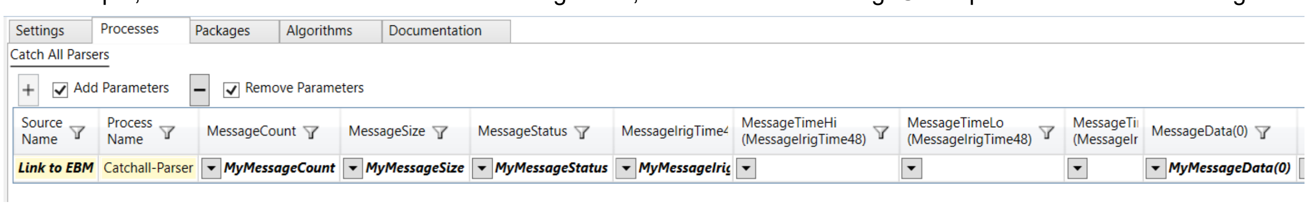
This interoperability is not a problem for a 1-Gbps link, that is, 1 Gbps and Auto can be mixed in the same link.

For details, request *TSD-AB-009 Auto-Negotiation issue.pdf* from Curtiss-Wright support (acra-support@curtisswright.com).

47.8.9 Not getting data – how to debug

Refer to the following if the KAD/EBM/102 is not getting data.

- Monitor the Report word of the KAD/EBM/102 if the link is connected to the Ethernet device for which the module is parsing data. Refer to the *KAD/EBM/102* data sheet for details of the Report word. If the Report word indicates that it's not connected, check the wiring.
- Use Wireshark on a PC directly connected to the Ethernet device to check if there's traffic and if it's as expected. If it's not, check the Ethernet device to be monitored by the KAD/EBM/102.
- Use catchall (remove all previously defined flows). With the KAD/EBM/102 selected, click the **Processes** tab. Select the **Add Parameters** check box and then click the **+** button. Data words and catchall tags are displayed as shown in the following figure.
  For example, to confirm that the module is monitoring traffic, check that the MessageCount parameter is incrementing.



  If there's traffic with the catch all, verify that the KAD/EBM/102 classifications with the Ethernet Builder are set up correctly.

If there's still no traffic after trying the above, the KAD/EBM/102 may be damaged, in which case contact Curtiss-Wright support (acra-support@curtisswright.com).

47.8.10 VLAN

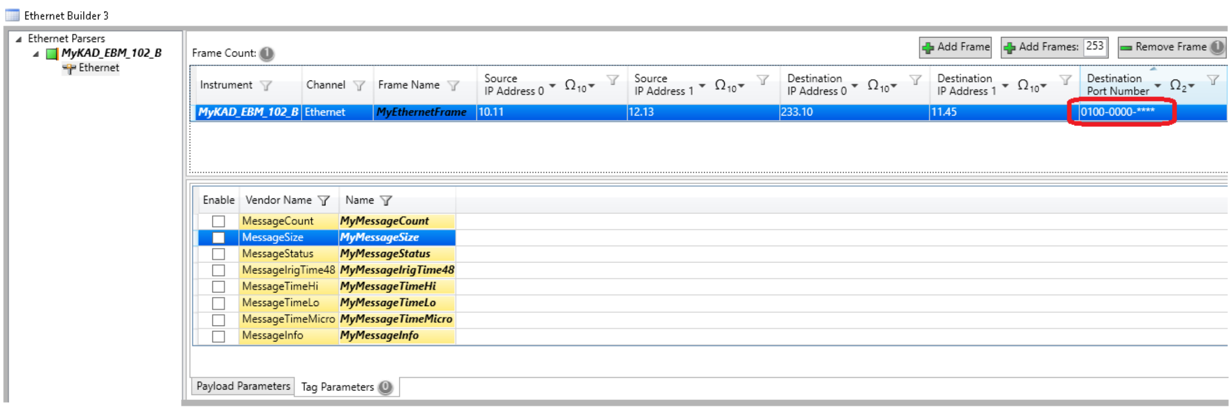
VLAN Tagging, also known as Frame Tagging, is a method developed by Cisco to help identify packets traveling through trunk links. When an Ethernet frame traverses a trunk link, a special VLAN tag is added to the frame and sent across the trunk link. A commercial switch maybe able to identify packets from different VLANs according to the information contained in its VLAN tags. IEEE 802.1Q adds a 4-byte VLAN tag between the Source/Destination MAC address and Length/Type fields of an Ethernet frame to identify the VLAN to which the frame belongs.

47.8.11 Using wildcards and tags in Ethernet Builder

On occasion, data may be required from a packet on the same link with the same source and destination IP addresses (but whose destination port may change). This data can be captured by using wildcards in the classifier settings. Wildcards are expressed as asterisks in Ethernet Builder, and can only be used when Binary format has been set for values in the classifier fields.

The following example illustrates how to set a wildcard in one of the filtering options.

1. Open **Ethernet Builder** and add a frame with the same settings as explained in “47.6.1 Using Ethernet Builder to configure the KAD/EBM/102” on page 6.
2. Select the **Destination Port Number** field in the new packet, and click the drop-down arrow next to the Ω symbol to change its format from **Decimal** to **Binary**.
   The 1024 value in the **Destination Port Number** field now reads **0100-0000-0000**.
3. To represent varying port numbers, insert wildcards (asterisks) where required. This forces the KAD/EBM/102 to capture this packet regardless of the destination port number.
   For example, 0100-0000-**** parses incoming packets with the same source and destination IP from port 1024 to 1039.
   NOTE: Wildcarding is only supported in binary mode. At this point, if the Binary format is changed back to Decimal (using the drop-down arrow beside the Ω symbol), the value for the **Destination Port Number** field (edited using wild-cards) does not change format.



4. Add parameters to the new packet as required.
5. Click the **Tag Parameters** tab to add additional message tag parameters such as **MessageTimeStamp** (timestamp of arrival of the message), **MessageCount**, **MessageSize** (bytes), **MessageStatus**, **MessageInfo** (stale / skipped message).
6. To save changes, click **Save & Close**.

47.8.12 Step-by-step instructions for generation of MCS packets (Scenario 1)

As explained in “47.5.4 Generating MCS packets” on page 6, MCS packets are created by the compiler. After settings are configured, you can proceed to build the PCM.

To generate MCS packets, on the **Tools** menu, click **Verify**. MCS packets are automatically built by DAS Studio 3 at the data source DAUs, this ensures that the data words to be sent over real-time PCM arrive at the data selector chassis in time for transmission.

In the following example, **P1** is a parameter sourced from an analog module (KAD/ADC/105/B) in slot 3 of **DAU_1**.

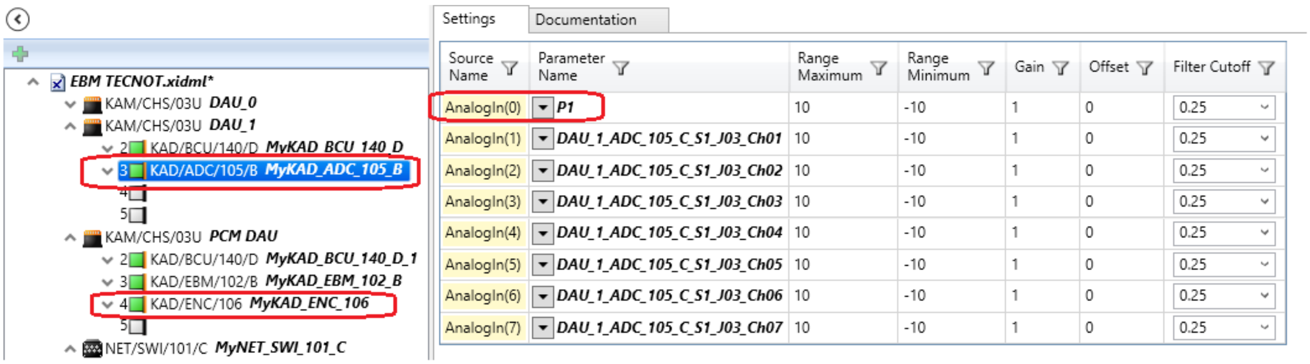


Figure 47-14: Parameter P1 from remote DAU_1

As shown in the following figure, **P1** is being transmitted over PCM at a rate of 512 sps.

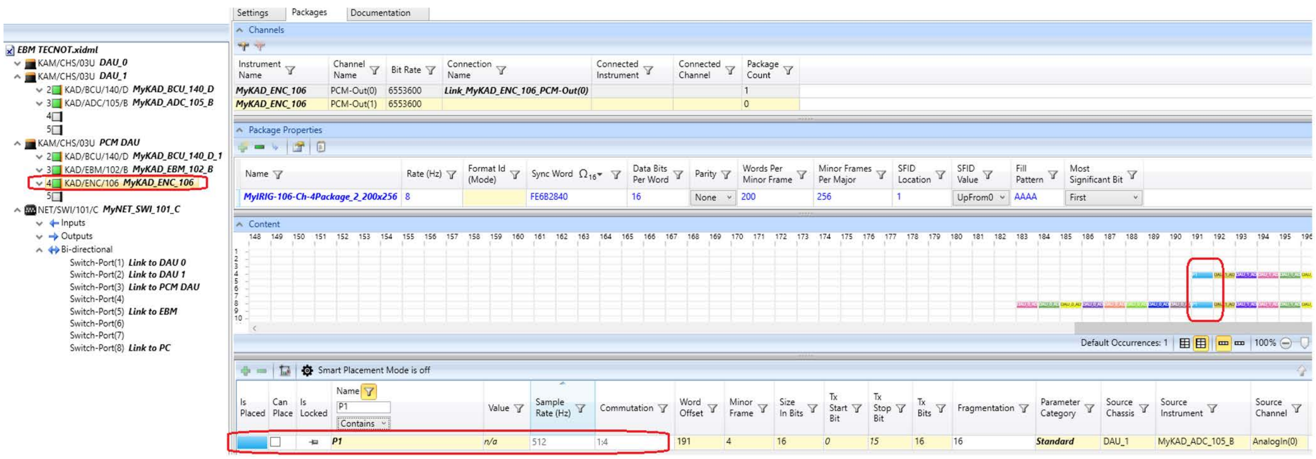


Figure 47-15: Parameter P1 transmitted over PCM at 512 Hz

After verification of the project, the compiler generates MCS packets (as shown in the following figure) with Multicast IP address 235.0.0.9 as per the KAD/EBM/102 setting in "47.5.4 Generating MCS packets" on page 6.

NOTE: The Stream ID is automatically generated and starts from 0xFFFF in the system. This value decrements by one for each new MCS packet required by DAS Studio 3.

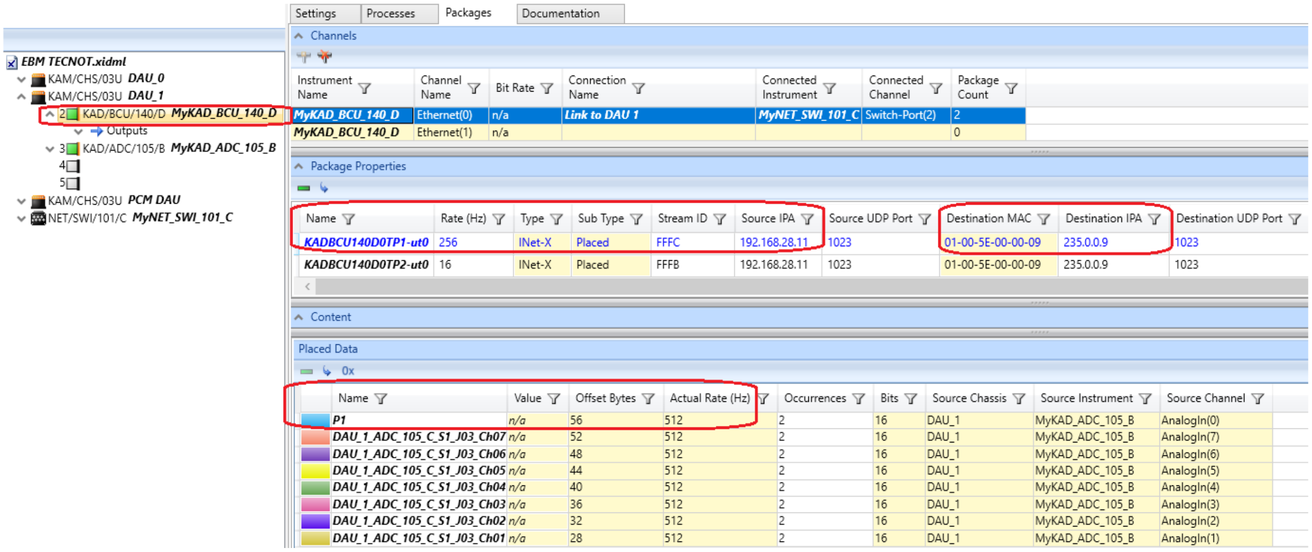


Figure 47-16: MCS packets generated at remote DAU_1

After verification of the project, the KAD/EBM/102 is also configured to automatically receive MCS packets.

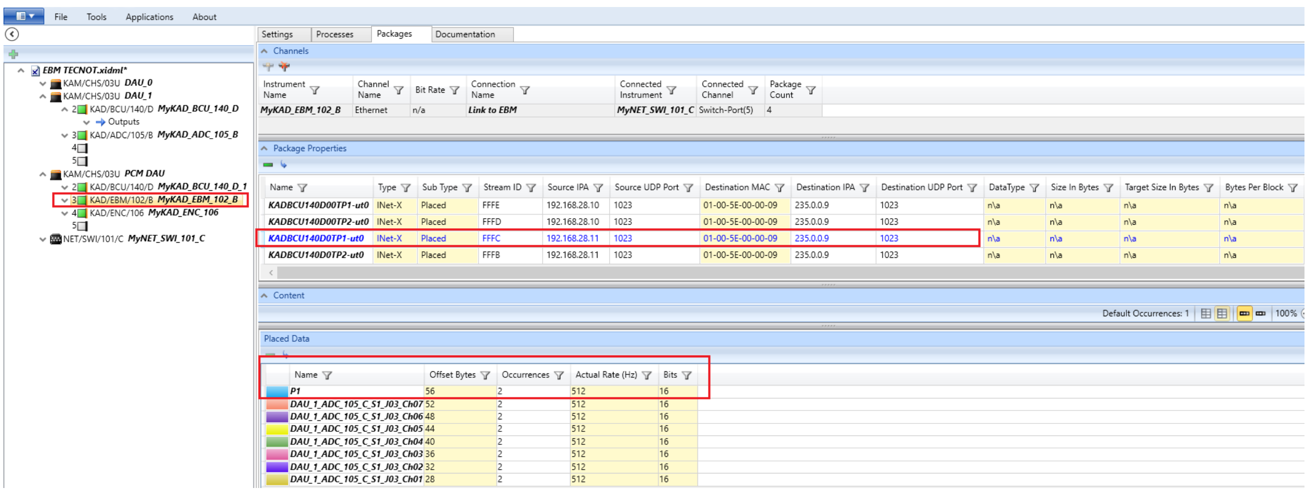


Figure 47-17: MCS packets created to parse MCS packets

NOTE: Verification of a file containing MCS scheduling requires writing new sections into the XidML file. That is, the original XidML file is modified by the compiler.

47.8.13 Step-by-step instructions for adding generic parser rules (Scenario 3)

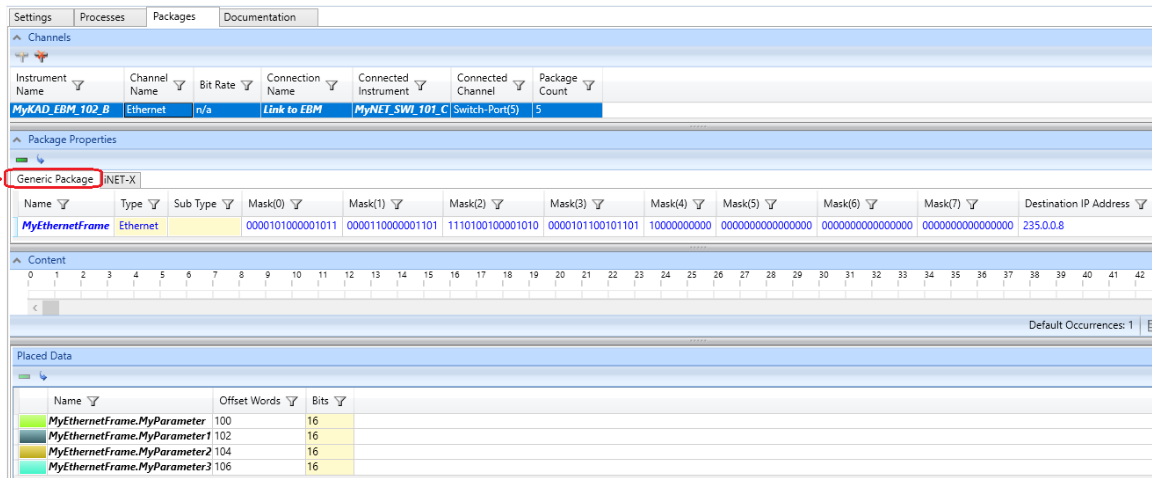
As explained in “47.7 Scenario 3: KAD/EBM/102 setup to parse Generic and MCS packets simultaneously” on page 10, the steps to add generic packets using the Ethernet Builder described in Scenario 2 can also be followed in Scenario 3.

The following example describes the procedure.

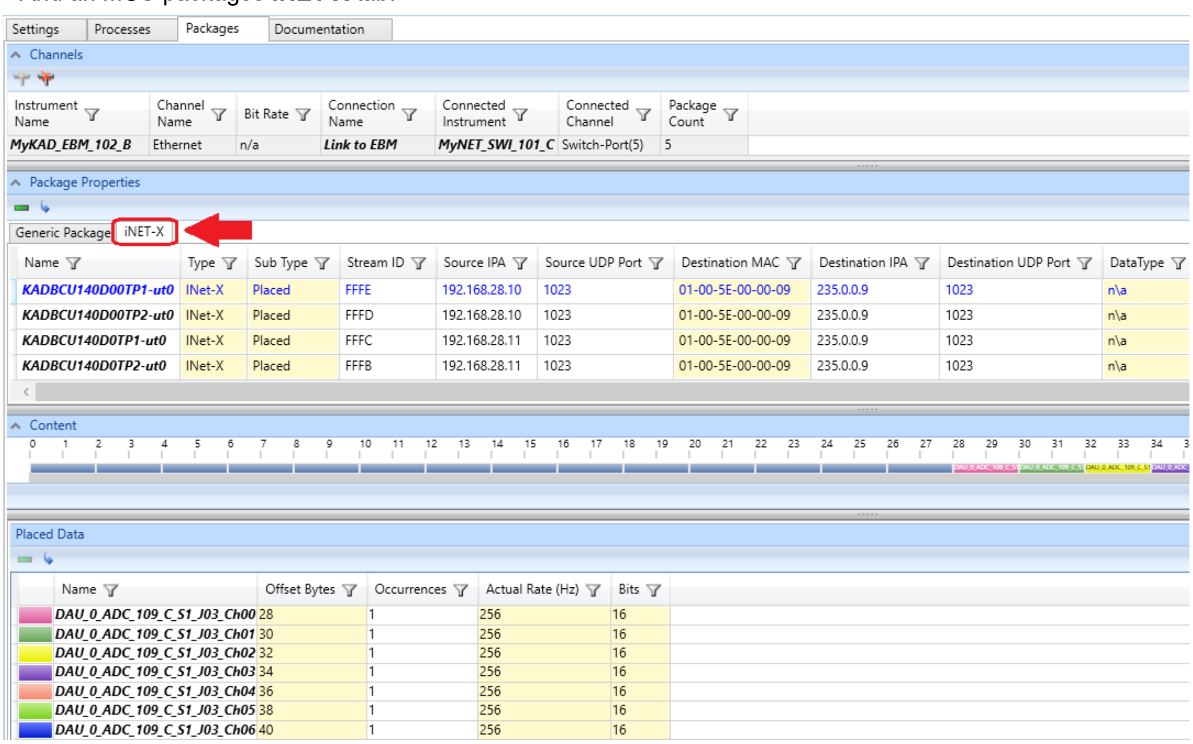
1. Ensure the KAD/EBM/102 is set up as per Scenario 3, that is, both Generic and MCS parser are set up.
   Note, at this stage of the configuration process:
   - The packages section of the KAD/EBM/102 is empty
   - Only one packet filter is set in the settings tab
   - When reopening Ethernet Builder, the parsing rules no longer appear. This applies to rules manually set and imported rules.

   NOTE: The above is expected as the project has not been verified yet.

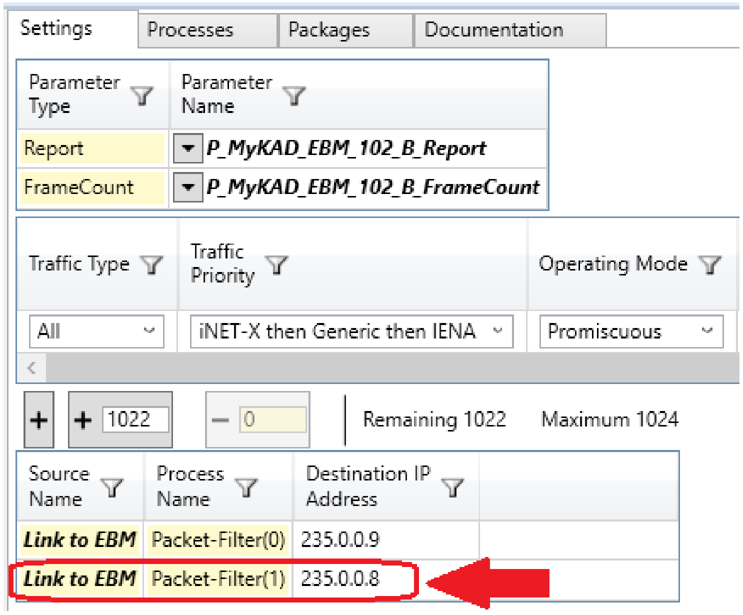
2. On the **Tools** menu, click **Verify**.
   Two tabs with packages appear.
   - A generic parser **Generic Package** tab.



   - And an MCS packages **iNET-X** tab.



   A new packet filter with destination IP **235.0.0.8** is added to the KAD/EBM/102 settings. This allows the module to differentiate between generic and MCS packets.



47.8.14 Related documentation

To better understand this paper, read the following documents.

Table 47-7: Data sheets

Document	Description
KAD/EBM/101/B	Ethernet bus monitor parser - 1ch
KAD/EBM/102/B	Gigabit Ethernet bus monitor parser - 1ch
KAD/ENC/106	IRIG-106 PCM encoder (PMF output)

Table 47-8: Data sheets

Document	Description
TEC/NOT/046	Using the KAD/EBM/101
TEC/NOT/067	IENA and iNET-X packet payload formats
TEC/NOT/068	Network MCS in KSM-500

Table 47-9: User manual

Document	Description
DOC/MAN/030	DAS Studio 3 User Manual

NOTE: Sample XidML files used to represent the scenarios in this technical note are available upon request from Curtiss-Wright support (acra-support@curtisswright.com).

This page is intentionally blank