# Specification of Kvaser Memorator Device configuration XML format



Printed Sunday 22nd May, 2016

(This page is intentionally left blank.)

# Contents

# 1 About this document

This document describes version 2.0 of the XML format used for configuring Kvaser Memorator (2nd generation) devices. A complete example of an XML configuration file can be found in Section 12.3, Sample XML setting, on Page 35. All content of attributes and elements are case sensitive.

This specification is applicable to the devices listed in Table 1.

| Device | Product Number (EAN) |
|---|---|
| Kvaser Eagle | 73-30130-00567-9 |
| Kvaser Memorator Pro 5xHS | 73-30130-00778-9 |
| Kvaser Memorator Pro 5xHS CB | 73-30130-00832-8 |
| Kvaser Memorator Pro 2xHS v2 | 73-30130-00819-9 |

Table 1: Kvaser Memorator (2nd generation) devices and their product numbers.

Throughout this document, we use the name Kvaser Memorator (2nd generation) to mean any one of the products listed above, unless otherwise noted.

# 2 Kvaser document start (KVASER)

The root element should be KVASER which does not have any attributes.

Element name: KVASER
Children: VERSION, BINARY_VERSION, SETTINGS, CAN_BUS, TRIGGERBLOCK, FILTERS, TRANSMIT_LISTS, MESSAGES, SCRIPTS

```
<KVASER>
  <VERSION />
  <BINARY_VERSION />

  <SETTINGS>
    <MODE />
    <CANPOWER />
    <COMMENT />
    <TARGET_EAN />
      ...
  </SETTINGS>

  <CAN_BUS>
    <PARAMETERS />
      ...
  </CAN_BUS>

  <TRIGGERBLOCK>
    <TRIGGERS>
      <TRIGGER_XXX />
        ...
    </TRIGGERS>

    <STATEMENTS>
      <STATEMENT>
        <EXPRESSION />
        <ACTIONS>
          <ACTION_XXX />
            ...
        </ACTIONS>
      </STATEMENT>
        ...
    </STATEMENTS>
  </TRIGGERBLOCK>

  <FILTERS>
    <XXX_PASS>
      <CHANNEL />
      ...
    </XXX_PASS>
      ...
    <XXX_STOP>
      <CHANNEL />
      ...
    </XXX_STOP>
```

```
    ...
  </FILTERS>

  <TRANSMIT_LISTS>
    <TRANSMIT_LIST>
      <TRANSMIT_MESSAGE />
      ...
    </TRANSMIT_LIST>
      ...
  </TRANSMIT_LISTS>

  <MESSAGES>
    <MESSAGE />
      ...
  </MESSAGES>

  <SCRIPTS>
    <SCRIPT>
      <FILENAME />
      <PATH />
    </SCRIPT>
    ...
  </SCRIPTS>

</KVASER>
```

The element `KVASER` does not have any attributes.

# 3 Configuration version (VERSION, BINARY_VERSION)

The VERSION element is used by the validation and conversion library. This document specifies version 2.0. Unknown elements are ignored.

Element name: VERSION
Children: None

```
<VERSION>2.0</VERSION>
```

The BINARY_VERSION element is used by the conversion library when creating the binary configuration for downloading to the device.

| Binary Version | Description |
|---|---|
| 5.0 | Base version for Kvaser Memorator 2nd Generation. |
| 6.0 | Supports CAN FD (needs support in Firmware). |

Table 2: Known Binary Versions.

Element name: BINARY_VERSION
Children: None

```
<BINARY_VERSION>5.0</BINARY_VERSION>
```

# 4 General settings (SETTINGS)

The SETTINGS element contains configuration settings such as afterburner and overriding logger modes.

Element name: SETTINGS
Children: MODE, CANPOWER, COMMENT, TARGET_EAN

```
<SETTINGS>
  <MODE />
  <CANPOWER />
  <COMMENT />
  <TARGET_EAN />
    ...
</SETTINGS>
```

## 4.1 Logger mode (MODE)

Logging is normally triggered by events specified as Trigger Conditions, see Section 6, Trigger Conditions (TRIGGERBLOCK, TRIGGERS), on Page 13. This can however be overrided by setting the attribute log_all to YES and thereby always log everything.

Element name: MODE
Children: None

```
<MODE
    log_all="YES"
    fifo_mode="NO"
    />
```

| Attribute name | Value | Description |
|---|---|---|
| log_all | YES, NO | Override TRIGGERBLOCK below and always log everything. |
| fifo_mode | YES, NO | Erase old data as needed. This means that all recorded messages are stored in one large circular buffer and only the oldest messages on the disk will be overwritten when needed. |

Table 3: Attributes for element MODE.

## 4.2 Afterburner timeout (CANPOWER)

The Kvaser Memorator (2nd generation) will continue to log data for a predetermined time after the loss of external power. This timeout can be set in the CANPOWER element.

Element name: CANPOWER
Children: None

```
<CANPOWER
    timeout="10000"
    />
```

| Attribute name | Value | Description |
|---|---|---|
| timeout | 0...30000 | The time in ms that Kvaser Memorator (2nd generation) will continue to log after the loss of external power. |

Table 4: Attributes for element CANPOWER.

## 4.3 Configuration comments (COMMENT)

You can store e.g. descriptions of the configuration in the `COMMENT` element.

Element name: `COMMENT`
Children: `None`

```
<COMMENT>This is my complete configuration.</COMMENT>
```

The element `COMMENT` is optional and does not have any attributes.

## 4.4 Target device (TARGET_EAN)

This is the EAN for the device that the configuration is written for. When validating the configuration, this target device capabilities are taken into consideration.

It is possible to specify multiple `TARGET_EAN` elements, in this case the configuration will be validated for each target.

Element name: `TARGET_EAN`
Children: `None`

```
<TARGET_EAN>73-30130-99010-4</TARGET_EAN>
```

The element `TARGET_EAN` does not have any attributes.

# 5 CAN Bus Parameters (CAN_BUS)

The element CAN_BUS contains one PARAMETERS or PARAMETERS_FD element for each channel on the device.

Element name: CAN_BUS
Children: PARAMETERS, PARAMETERS_FD

```
<CAN_BUS>
  <PARAMETERS />
    ...
</CAN_BUS>
```

## 5.1 Bus parameters (PARAMETERS)

The PARAMETERS element contains information about the CAN channel, such as bitrate, sample point and synchronization jump width. You must specify one PARAMETERS element for each channel on the device. Different devices have different number of channels so make sure you specify the correct amount.

While bitrate and sjw bus parameters can be provided as they are, sample point can be altered by specifying the length of first and second time segments. The first segment comprises all time quanta from, but not including, sync segment to the sampling point. The second time segment consists of time quanta from the sampling point to the end of the bit.

The conversion library helps you validate this through the use of TARGET_EAN, see Section 4.4, Target device (TARGET_EAN), on Page 10.

Element name: PARAMETERS
Children: None

```
<PARAMETERS
  channel="0"
  bitrate="1000000"
  tseg1="11"
  tseg2="4"
  sjw="1"
  silent="YES"
  />
```

| Attribute name | Value | Description |
|---|---|---|
| channel | uint8 | The CAN channel to set |
| bitrate | uint32 | Bitrate |
| tseg1 | uint8 | First time segment length, in timequanta |
| tseg2 | uint8 | Second time segment length, in timequanta |
| sjw | uint8 | Synchronization jump width |
| silent | YES, NO | Do not transmit anything on the CAN bus when logging |

Table 5: Attributes for element PARAMETERS

However, if your Kvaser Memorator (2nd generation) supports CAN-FD protocol and you want to communicate in CAN-FD, then more bus parameters need to be specified. Pay attention that allowed range for first and second time segment lengthes in arbitration phase are different. Moreover, some devices would allow you to run communication in non-ISO mode (required to use with controllers that had been manufactured before ISO standard was issued). Note that if you specify one of “data phase” attributes, or an “iso” attribute, then you are expected to provide all of the rest. The device will always converse in CAN-FD in this case.

Element name: `PARAMETERS`
Children: `None`

```
<PARAMETERS
    channel="1"
    bitrate="1000000"
    tseg1="11"
    tseg2="4"
    sjw="1"
    bitrate_brs="10000000"
    tseg1_brs="5"
    tseg2_brs="2"
    sjw_brs="1"
    silent="YES"
    iso="YES"
    />
```

| Attribute name | Value | Description |
|---|---|---|
| channel | uint8 | The CAN channel to set |
| bitrate | uint32 | Bitrate, arbitration phase |
| tseg1 | uint8 | First time segment length, in tq, arbitration phase |
| tseg2 | uint8 | Second time segment length, in tq, arbitration phase |
| sjw | uint8 | Synchronization jump width, arbitration phase |
| bitrate_brs | uint32 | Bitrate, data phase |
| tseg1_brs | uint8 | First time segment length, in tq, data phase |
| tseg2_brs | uint8 | Second time segment length, in tq, data phase |
| sjw_brs | uint8 | Synchronization jump width, data phase |
| silent | YES, NO | Do not transmit anything on the CAN bus when logging |
| iso | YES, NO | Message follows ISO standard for CAN FD |

Table 6: Attributes for element PARAMETERS

# 6 Trigger Conditions (TRIGGERBLOCK, TRIGGERS)

The trigger conditions describe how Kvaser Memorator (2nd generation) should capture and filter data. The possibilities range from simply logging everything to advanced combinations of triggers and filters.

Triggers are useful if you want Kvaser Memorator (2nd generation) to start/stop logging when something special happens. It can be a message with a certain identifier, a signal value, an error frame or an external trigger. In Kvaser Memorator (2nd generation) you can in addition to the above mentioned features also set a trigger that will be activated when the SD-card is full. Without the use of any triggers or filters in the configuration, the flash disk can be filled up relatively fast depending on the bus load.

The element `TRIGGERS` can hold max 16 triggers (this is device dependant, see also Section 12.1, Implementation limits, on Page 35).

Element name: `TRIGGERBLOCK`
Children: `TRIGGERS, STATEMENTS`

Element name: `TRIGGERS`
Children: `TRIGGER_MSG_ID, TRIGGER_MSG_DLC, TRIGGER_MSG_ERROR_FRAME, TRIGGER_SIGVAL, TRIGGER_EXTERNAL, TRIGGER_TIMER, TRIGGER_DISK_FULL, TRIGGER_STARTUP`

The elements `TRIGGER_XXX` are used to match CAN messages, timers, external triggers or other inputs.

```
<TRIGGERBLOCK>
  <TRIGGERS>
    <TRIGGER_XXX />
      ...
  </TRIGGERS>

  <STATEMENTS>
    <STATEMENT>
      <EXPRESSION />
      <ACTIONS>
        <ACTION_XXX />
          ...
      </ACTIONS>
    </STATEMENT>
      ...
  </STATEMENTS>
</TRIGGERBLOCK>
```

See Section 6 Trigger Conditions (TRIGGERBLOCK, TRIGGERS) for a description of the element `STATEMENTS`.

## 6.1 Message Id trigger (TRIGGER_MSG_ID)

Trigger on a messages within a specified range of message id.

The element TRIGGER_MSG_ID is a child to the TRIGGERS element.

Element name: TRIGGER_MSG_ID
Children: None

```
<TRIGGER_MSG_ID
    channel="1"
    name="My_first_id_trigger"
    timeout="0"
    msgid="6"
    msgid_min="2"
    can_ext="YES"
    can_fd="NO"
    protocol="NONE"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| channel | uint8 | CAN channel, note that the range is device dependant. |
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |
| timeout | 0...1000000000, -1 | The trigger will remain true for the time specified in ms, -1 means forever. |
| msgid | uint32 | Message identifier, upper limit. |
| msgid_min | uint32 | Message identifier, lower limit. Must be equal to, or lesser than msgid. |
| can_ext | YES, NO | Use CAN extended identifiers. |
| can_fd | YES, NO | Optional. Message in CAN FD format. |
| protocol | NONE, J1939 | Use a Higher Layer Protocol to interpret the message id. |
| msg_field | SRC, DST, PGN | How to interpret the msgid when using a higher level protocol. Only used with protocol J1939. May be a comma separated list of specified values. |

Table 7: Attributes for element TRIGGER_MSG_ID

## 6.2 Message length trigger (TRIGGER_MSG_DLC)

Trigger on messages within a range of data length codes.

The element TRIGGER_MSG_DLC is a child to the TRIGGERS element.

Element name: TRIGGER_MSG_DLC
Children: None

```
<TRIGGER_MSG_DLC
    channel="1"
    name="My_first_dlc_trigger"
```

```
        timeout="100"
        can_fd="NO"
        dlc="7"
        dlc_min="5"
        />
```

| Attribute name | Values | Description |
|---|---|---|
| channel | uint8 | CAN channel, note that the range is device dependant. |
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |
| timeout | 0...1000000000, -1 | The trigger will remain true for the time specified in ms, -1 means forever. |
| can_fd | YES, NO | Optional. Message in CAN FD format. |
| dlc | uint32 | Upper limit for a data length limit. |
| dlc_min | uint32 | Lower limit for a data length limit. Must be equal to, or lesser than dlc. |

Table 8: Attributes for element TRIGGER_MSG_DLC

## 6.3 Error frame trigger (TRIGGER_MSG_ERROR_FRAME)

Trigger on the error flag in a CAN message.

The element `TRIGGER_MSG_ERROR_FRAME` is a child to the `TRIGGERS` element.

Element name: `TRIGGER_MSG_ERROR_FRAME`
Children: `None`

```
<TRIGGER_MSG_ERROR_FRAME
    channel="1"
    name="My_first_flag_trigger"
    timeout="100"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| channel | uint8 | CAN channel, note that the range is device dependant. |
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |
| timeout | 0...1000000000, -1 | The trigger will remain true for the time specified in ms, -1 means forever. |

Table 9: Attributes for element TRIGGER_MSG_ERROR_FRAME

## 6.4 Signal value trigger (TRIGGER_SIGVAL)

Trigger on raw signal values.

The element `TRIGGER_SIGVAL` is a child to the `TRIGGERS` element.

Element name: TRIGGER_SIGVAL
Children: None

```
<TRIGGER_SIGVAL
    channel="1"
    name="My_first_sigval_trigger"
    timeout="-1"
    msgid="34"
    can_ext="YES"
    can_fd="NO"
    dlc="4"
    startbit="1"
    length="7"
    datatype="UNSIGNED"
    byteorder="BIG_ENDIAN"
    data="234"
    data_min="99"
    condition="ON_DATA_EQUAL_TO"
    protocol="J1939"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| channel | uint8 | CAN channel, note that the range is device dependant. |
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |
| timeout | 0...1000000000, -1 | The trigger will remain true for the time specified in ms, -1 means forever. |
| msgid | uint32 | Message identifier |
| can_ext | YES, NO | Use CAN extended identifiers. |
| can_fd | YES, NO | Optional. Message in CAN FD format. |
| dlc | uint8 | Optional. Data length code, length will be ignored if dlc is omitted. |
| startbit | uint8 | Signal starts at this bit. |
| length | uint8 | Signal length in bits. |
| datatype | UNSIGNED, SIGNED | Type of data in signal. |
| byteorder | BIG_ENDIAN, LITTLE_ENDIAN | Byte order of data. |
| protocol | NONE, J1939 | Use a Higher Layer Protocol to interpret the message id. |
| msg_field | SRC, DST, PGN | How to interpret the msgid when using a higher level protocol. Only used with protocol J1939. May be a comma separated list of specified values. |
| data | uint32, int32 | Raw value to match, upper limit. |
| data_min | uint32, int32 | Raw value to match, lower limit. |
| condition | ON_DATA_EQUAL_TO, | |
| | ON_DATA_NOT_EQUAL_TO, | |
| | ON_DATA_LARGER_THAN, | |
| | ON_DATA_SMALLER_THAN, | |
| | ON_DATA_CHANGE_TO, | |
| | ON_DATA_CHANGE_FROM | Condition that the raw value range should match. |

Table 10: Attributes for element TRIGGER_SIGVAL

## 6.5 External trigger (TRIGGER_EXTERNAL)

Trigger on an external event. See your device's userguide for more information on how to connect this.

The element `TRIGGER_EXTERNAL` is a child to the `TRIGGERS` element.

Element name: `TRIGGER_EXTERNAL`
Children: `None`

```
<TRIGGER_EXTERNAL
    channel="0"
    name="My_first_external_trigger"
```

```
timeout="0"
level="TRIG_EXTERNAL_LEVEL_LO_HI"
/>
```

| Attribute name | Values | Description |
|---|---|---|
| channel | uint8 | CAN channel, note that the range is device dependant. |
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |
| timeout | 0...1000000000, -1 | The trigger will remain true for the time specified in ms, -1 means forever. |
| level | TRIG_EXTERNAL_LEVEL_LO_HI, | |
| | TRIG_EXTERNAL_LEVEL_HI_LO | Trigger on positive or negative flank of the trigger. |

Table 11: Attributes for element TRIGGER_EXTERNAL

## 6.6 Timer trigger (TRIGGER_TIMER)

Trigger after a specific time have elapsed.

The element `TRIGGER_TIMER` is a child to the `TRIGGERS` element.

Element name: `TRIGGER_TIMER`
Children: `None`

```
<TRIGGER_TIMER
    name="My_first_timer_trigger"
    timeout="0"
    offset="5"
    repeat="NO"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |
| timeout | 0...1000000000, -1 | The trigger will remain true for the time specified in ms, -1 means forever. |
| offset | uint32 | Delay in seconds after which the trigger becomes true. |
| repeat | YES, NO | Trigger more than once. |

Table 12: Attributes for element TRIGGER_TIMER

## 6.7 Disk trigger (TRIGGER_DISK_FULL)

Trigger when the disk becomes full. Note that the disk will never become full in FIFO mode.

The element TRIGGER_DISK_FULL is a child to the TRIGGERS element.

Element name: TRIGGER_DISK_FULL
Children: None

```
<TRIGGER_DISK_FULL
    name="My_first_diskFull_trigger"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |

Table 13: Attributes for element TRIGGER_DISK_FULL

## 6.8 Start trigger (TRIGGER_STARTUP)

Trigger once at power on.

The element TRIGGER_STARTUP is a child to the TRIGGERS element.

Element name: TRIGGER_STARTUP
Children: None

```
<TRIGGER_STARTUP
    name="My_first_startup_trigger"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name is used to refer to this trigger from other blocks, e.g. from expression. Must not contain spaces. |

Table 14: Attributes for element TRIGGER_STARTUP

# 7 Trigger Conditions (TRIGGERBLOCK, STATEMENTS)

This section describes the element STATEMENTS inside the element TRIGGERBLOCK. See also Section 6, Trigger Conditions (TRIGGERBLOCK, TRIGGERS), on Page 13 for descriptions of the element TRIGGERS.

Element name: STATEMENTS
Children: STATEMENT

```
<TRIGGERBLOCK>
  <TRIGGERS>
    <TRIGGER_XXX />
      ...
  </TRIGGERS>

  <STATEMENTS>
    <STATEMENT>
      <EXPRESSION />
      <ACTIONS>
        <ACTION_XXX />
          ...
      </ACTIONS>
    </STATEMENT>
      ...
  </STATEMENTS>
</TRIGGERBLOCK>
```

## 7.1 Trigger statement (STATEMENT)

The element STATEMENT binds together triggers and actions, e.g. start logging when a specific CAN message is received.

In Kvaser Memorator (2nd generation) you can have up to 8 STATEMENT elements.

Element name: STATEMENT
Children: EXPRESSION, ACTION

```
<STATEMENT
    pretrigger ="0"
    posttrigger="500">
  <EXPRESSION>
    (My_first_dlc_trigger OR My_first_sigval_trigger) AND
       My_first_id_trigger
  </EXPRESSION>
  <ACTIONS>
    <ACTION_START_LOG/>
  </ACTIONS>
</STATEMENT>
```

| Attribute name | Values | Description |
|---|---|---|
| pretrigger | uint32 | The length of the pretrigger in ms. |
| posttrigger | uint32 | The length of the posttrigger in ms. |

Table 15: Attributes for element STATEMENT.

## 7.2 Trigger expression (EXPRESSION)

The trigger expression is formed from triggers, parantheses, and the logical operators `AND` and | `OR`. Expressions are evaluated from left to right, with `AND` and `OR` having the same priority, so that

```
  (My_first_dlc_trigger OR My_first_sigval_trigger) AND
     My_first_id_trigger
</EXPRESSION>
```

and

```
  My_first_dlc_trigger OR My_first_sigval_trigger AND
     My_first_id_trigger
```

are equivalent expressions. Please use parantheses to disambiguate complex expressions. The expression is written using strings that refer to the attribute `name` in each trigger variable.

The expression can contain up to 31 items, where triggers and operators each counts as one item.

Element name: `EXPRESSION`
Children: `None`

```
<EXPRESSION>
  (My_first_dlc_trigger OR My_first_sigval_trigger) AND
     My_first_id_trigger
</EXPRESSION>
```

The element `EXPRESSION` does not have any attributes.

## 7.3 Trigger actions (ACTIONS)

The `ACTIONS` element contains the actions that will be performed when the trigger expression becomes true.

With the help of `ACTIONS`, it is possible to use triggers as start and stop triggers. Add start triggers if you want the logger to start its logging when the specified trigger condition becomes true. Add stop triggers if you want Memorator to stop its logging when the specified trigger condition becomes true.

The `ACTIONS` element can contain up to 6 actions (`ACTION_XXX` elements).

Element name: ACTIONS
Children: ACTION_START_LOG, ACTION_STOP_LOG,
ACTION_STOP_LOG_COMPLETELY, EXTERNAL_PULSE
ACTION_ACTIVATE_AUTO_TRANSMIT_LIST,
ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST

```
<ACTIONS>
  <ACTION_XXX />
    ...
</ACTIONS>
```

## 7.4 Start log action (ACTION_START_LOG)

The element ACTION_START_LOG is used to start logging.

Element name: ACTION_START_LOG
Children: None

```
<ACTION_START_LOG/>
```

The element ACTION_START_LOG is optional and does not have any attributes.

## 7.5 Stop log action (ACTION_STOP_LOG)

The element STOP_LOG stops logging after the posttrigger is finished (attribute posttrigger in element STATEMENT).

Element name: ACTION_STOP_LOG
Children: None

```
<ACTION_STOP_LOG/>
```

The element ACTION_STOP_LOG is optional and does not have any attributes.

## 7.6 Stop log completely (ACTION_STOP_LOG_COMPLETELY)

The element STOP_LOG_COMPLETELY stops logging after posttrigger is finished, as ACTION_STOP_LOG, but it will also keep the Kvaser Memorator (2nd generation) from triggering again and go off bus as well. A power cycle is needed to restart the logging.

Element name: ACTION_STOP_LOG_COMPLETELY
Children: None

```
<ACTION_STOP_LOG_COMPLETELY/>
```

The element ACTION_STOP_LOG_COMPLETELY is optional and does not have any attributes.

## 7.7 Send pulse (ACTION_EXTERNAL_PULSE)

The element ACTION_EXTERNAL_PULSE sends out an external pulse, usually on the first CAN channel, see your device userguide for detaills.

Element name: ACTION_EXTERNAL_PULSE
Children: None

```
<ACTION_EXTERNAL_PULSE
    duration="1"
    />
```

The element ACTION_EXTERNAL_PULSE is optional.

| Attribute name | Values | Description |
|---|---|---|
| duration | uint32 | Length of pulse, in seconds. |

Table 16: Attributes from element ACTION_EXTERNAL_PULSE.

## 7.8 Activate Transmits (ACTION_ACTIVATE_AUTO_TRANSMIT_LIST)

The element ACTION_ACTIVATE_AUTO_TRANSMIT_LIST activates the transmit list with attribute name. See Section 9.1, List of messages to send (TRANSMIT_LIST), on Page 29.

Element name: ACTION_ACTIVATE_AUTO_TRANSMIT_LIST
Children: None

```
<ACTION_ACTIVATE_AUTO_TRANSMIT_LIST
    name="TransmitList1"
    />
```

The element ACTION_ACTIVATE_AUTO_TRANSMIT_LIST is optional.

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name of transmit list to activate. |

Table 17: Attributes from element ACTION_ACTIVATE_AUTO_TRANSMIT_LIST.

## 7.9 Deactivate Transmits (ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST)

The element ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST deactivates the transmit list with attribute name. See Section 9.1, List of messages to send (TRANSMIT_LIST), on Page 29.

Element name: ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST
Children: None

```
<ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST
    name="SecondList"
    />
```

The element `ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST` is optional.

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name of transmit list to deactivate. |

Table 18: Attributes from element ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST.

# 8 Filter block (FILTERS)

Filters specify conditions for to record messages. One can specify either ’pass’ or ’stop’ filter. ’Pass’ filter allow specifying what messages are going to be recorded Kvaser Memorator (2nd generation). ’Stop’ filters allow specifying what messages are *not* going to be recorded. Filters’ purpose is to limit the number of logged messages. Filters block is evaluated before the triggers, so be careful not to filter away CAN messages that can should be used by the triggers.

In the extremely rare occasion where ’pass’ and ’stop’ filters are needed to be used together, ’pass’ filters will always be evaluated ahead of ’stop’ filters. As for their internal ordering, filters of same type are evaluated in the relative order they were provided by the user. This means that you should sort the filters so that those that affect the higher-rate messages come earlier in the list.

Element name: `FILTERS`
Children: `MESSAGE_PASS, MESSAGE_STOP, MESSAGE_COUNTING_PASS, FLAGS_PASS, FLAGS_STOP, SIGNAL_PASS, SIGNAL_STOP`

```
<FILTERS>
  <XXX_PASS>
    <CHANNEL />
    ...
  </XXX_PASS>
    ...
  <XXX_STOP>
    <CHANNEL />
    ...
  </XXX_STOP>
    ...
</FILTERS>
```

## 8.1 Specifying channels (CHANNEL)

All filter elements need to specify what channel(s) they are active on. This is done by adding one or more `CHANNEL` elements. See the device user guide for information about how many CAN channels your Kvaser Memorator (2nd generation) has.

Element name: `CHANNEL`
Children: `None`

```
<CHANNEL>0</CHANNEL>
```

The element `CHANNEL` does not have any attributes.

## 8.2 Filter on message (MESSAGE_PASS and MESSAGE_STOP)

A message filter allow filtering on message content (e.g. identifier and data length).

Element name: MESSAGE_PASS, MESSAGE_STOP
Children: CHANNEL

The elements MESSAGE_PASS and MESSAGE_STOP are optional.

```
<MESSAGE_PASS
    protocol="NONE"
    msgid="34"
    msgid_min="34"
    can_ext="NO"
    can_fd="NO"
    dlc="8"
    >
  <CHANNEL>0</CHANNEL>
</MESSAGE_PASS>
```

| Attribute name | Values | Description |
|---|---|---|
| protocol | NONE, J1939 | Use a Higher Layer Protocol to interpret the message id. |
| msg_field | SRC, DST, PGN | How to interpret the msgid when using a higher level protocol. Only used with protocol J1939. May be a comma separated list of specified values. |
| msgid | uint32 | Message identifier, upper limit. |
| msgid_min | uint32 | Message identifier, lower limit. Must be equal to, or lesser than msgid. |
| can_ext | YES, NO | Use CAN extended identifiers. |
| can_fd | YES, NO | Optional. Message in CAN FD format. |
| dlc | uint8 | Optional. Data length code, length will be ignored if dlc is omitted. |

Table 19: Attributes for element MESSAGE_PASS and MESSAGE_STOP

## 8.3 Filter on signal (SIGNAL_PASS and SIGNAL_STOP)

A Signal filter allow filtering on the raw signal value in a CAN message.

Element name: SIGNAL_PASS, SIGNAL_STOP
Children: CHANNEL

The elements SIGNAL_PASS and SIGNAL_STOP are optional.

```
<SIGNAL_PASS
    protocol="J1939"
    msgid="34"
    msg_field="PGN,SRC,DST"
    can_ext="NO"
    can_fd="NO"
    dlc="8"
```

```
      startbit="2"
      length="5"
      datatype="UNSIGNED"
      byteorder="BIG_ENDIAN"
      data="0x2D"
      >
    <CHANNEL>0</CHANNEL>
    <CHANNEL>4</CHANNEL>
</SIGNAL_PASS>
```

| Attribute name | Values | Description |
|---|---|---|
| protocol | NONE, J1939 | Use a Higher Layer Protocol to interpret the message id. |
| msg_field | SRC, DST, PGN | How to interpret the msgid when using a higher level protocol. Only used with protocol J1939. May be a comma separated list of specified values. |
| msgid | uint32 | Message identifier |
| can_ext | YES, NO | Use CAN extended identifiers. |
| can_fd | YES, NO | Optional. Message in CAN FD format. |
| dlc | uint8 | Optional. Data length code, length will be ignored if dlc is omitted. |
| startbit | uint8 | Signal starts at this bit. |
| length | uint8 | Signal length in bits. |
| datatype | UNSIGNED, SIGNED | Type of data in signal. |
| byteorder | BIG_ENDIAN, LITTLE_ENDIAN | Byte order of data. |
| data | uint32 | Raw signal value to match. |

Table 20: Attributes for element SIGNAL_PASS and SIGNAL_STOP

## 8.4 Filter by flag (FLAG_PASS and FLAG_STOP)

A Flag filter will allow filtering on special types of messages (error frames, messages with standard or extended CAN identifiers). Note that only one type of flag may be used per filter.

Element name: `FLAG_PASS, FLAG_STOP`
Children: `CHANNEL`

The elements FLAG_PASS and FLAG_STOP are optional.

```
<FLAG_PASS
    flag_std="YES"
    flag_ext="YES"
    flag_errorframe="NO"
    >
  <CHANNEL>3</CHANNEL>
  <CHANNEL>4</CHANNEL>
</FLAG_PASS>
```

| Attribute name | Values | Description |
|---|---|---|
| flag_std | YES, NO | Filter on messages with standard CAN identifiers. |
| flag_ext | YES, NO | Filter on messages with extended CAN identifiers. |
| flag_errorframe | YES, NO | Filter on error frames. |

Table 21: Attributes for elements FLAG_PASS and FLAG_STOP

## 8.5 Counting Pass filters (MESSAGE_COUNTING_PASS, SIGNAL_COUNTING_PASS, FLAG_COUNTING_PASS)

A Counting Pass filter maintains a counter of occurrences of the specified message, signal or flag, and will either 'pass' or 'stop' matching CAN messages depending on current value of this counter. The user can allow (for example) every 25th message of the specified type to pass by setting `counter_threshold` attribute to `1` and `counter_max` attribute to 25. Counting Pass filters can be defined for Message, Signal and Flag filters and follow their respective attribute definitions, with the additions listed here.

Element name: `MESSAGE_COUNTING_PASS`, `SIGNAL_COUNTING_PASS`, `FLAG_COUNTING_PASS`
Children: `CHANNEL`

```
<MESSAGE_COUNTING_PASS
    protocol="NONE"
    msgid="34"
    can_ext="NO"
    can_fd="NO"
    counter_threshold="7"
    counter_max="16"
    >
  <CHANNEL>0</CHANNEL>
  <CHANNEL>2</CHANNEL>
</MESSAGE_COUNTING_PASS>
```

The elements `MESSAGE_COUNTING_PASS`, `SIGNAL_COUNTING_PASS` and `FLAG_COUNTING_PASS` are optional.

| | | |
|---|---|---|
| counter_threshold | uint16 | Filter is inactive when the number of counted messages is larger than this threshold. |
| counter_max | uint16 | The counting will restart on the first message count that is larger than this. |

Table 22: Additional attributes for elements MESSAGE_COUNTING_PASS, SIGNAL_COUNTING_PASS and FLAG_COUNTING_PASS

# 9 Transmit lists (TRANSMIT_LISTS)

A transmit list is a list of messages that can be sent on the bus when a specific expression is fulfilled. It is possible to specify up to 8 transmit lists.

Element name: TRANSMIT_LISTS
Children: TRANSMIT_LIST

```
<TRANSMIT_LISTS>
  <TRANSMIT_LIST>
    <TRANSMIT_MESSAGE />
    ...
  </TRANSMIT_LIST>
    ...
</TRANSMIT_LISTS>
```

## 9.1 List of messages to send (TRANSMIT_LIST)

A transmit list is a list of messages that can be sent on the bus when a specific expression is fulfilled. They will be sent in the order specified and can be sent both continuously or only once. It is also possible to set a delay time between every message. A transmit list is activated either through the action ACTION_ACTIVATE_AUTO_TRANSMIT_LIST (see subsection 7.8) or at power on.

Element name: TRANSMIT_LIST
Children: TRANSMIT_MESSAGE

The element TRANSMIT_LIST is optional (TRANSMIT_LISTS must be present but it may be empty).

```
<TRANSMIT_LIST
    name="TransmitList1"
    msg_delay="1500"
    cycle_delay="0"
    cyclic="NO"
    autostart="NO"
    >
</TRANSMIT_LIST>
```

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name is used to refer to this transmit list from other blocks, e.g. from statement block. Must not contain spaces. |
| msg_delay | uint32 | The delay between two messages. In 100 µs. |
| cycle_delay | uint32 | The time between the last and first message if the list is cyclic. In 100 µs |
| cyclic | YES, NO | Repeat the message sequence. |
| autostart | YES, NO | Begin att power on. |

Table 23: Attributes for element TRANSMIT_LIST.

## 9.2 Message to send (TRANSMIT_MESSAGE)

Transmit a CAN message or an error frame.

Element name: `TRANSMIT_MESSAGE`
Children: `None`

```
<TRANSMIT_MESSAGE
    name="Message2"
    channel="0"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name of message to send, refers to the Message attribute name. |
| channel | uint8 | CAN channel, note that the range is device dependant. |

Table 24: Attributes for element TRANSMIT_MESSAGE.

# 10 List of CAN Messages (MESSAGES)

The element MESSAGES holds a list of defined CAN messages for referencing in TRANSMIT_LISTs.

Element name: MESSAGES
Children: MESSAGE

The element MESSAGES is optional.

```
<MESSAGES>
  <MESSAGE />
    ...
</MESSAGES>
```

## 10.1 Defined CAN Message (MESSAGES)

The element MESSAGES defines the CAN messages that are refered to in the transmit lists.

Element name: MESSAGE
Children: None

```
<MESSAGE
    name="Message2"
    msgid="0x6e"
    dlc="8"
    can_ext="YES"
    can_fd="NO"
    can_fd_brs="NO"
    error_frame="NO"
    remote_frame="NO"
    b0="0x33"
    b1="0x33"
    b2="0x33"
    b3="0x33"
    b4="0x10"
    b5="0x0"
    b6="0x0"
    b7="0x0"
    />
```

| Attribute name | Values | Description |
|---|---|---|
| name | string | Name is used to refer to this message from other blocks, e.g. from transmit list. Must not contain spaces. |
| msgid | uint32 | Message identifier |
| can_ext | YES, NO | Use CAN extended identifiers. |
| can_fd | YES, NO | Optional. Message in CAN FD format. |
| can_fd_brs | YES, NO | Mandatory when `can_fd` is present, otherwise optional. Message is bit rate switched. |
| error_frame | YES, NO | Message is an error frame. |
| remote_frame | YES, NO | Message is a remote frame. |
| dlc | uint8 | Data length code. |
| b0 | uint8 | Data byte 0. |
| b1 | uint8 | Data byte 1. |
| b2 | uint8 | Data byte 2. |
| b3 | uint8 | Data byte 3. |
| b4 | uint8 | Data byte 4. |
| b5 | uint8 | Data byte 5. |
| b6 | uint8 | Data byte 6. |
| b7 | uint8 | Data byte 7. |

Table 25: Attributes for element MESSAGE.

# 11 List of Scripts (SCRIPTS)

Up to four compiled scripts can be added to the configuration.

Element name: SCRIPTS
Children: SCRIPT

The element SCRIPTS is optional.

```
<SCRIPTS>
  <SCRIPT>
    <FILENAME />
    <PATH />
  </SCRIPT>
  ...
</SCRIPTS>
```

## 11.1 Script (SCRIPT)

Definition of where to fetch the compiled t program (.txe) and other script related settings. Restriction using script_external="YES": Filename maximum 12 characters including the ending ".txe". Special leading characters (e.g. "F:/") may have device dependent interpretation.

If the t program is using any of the functions filterDropMessage, loggerStart or loggerStop, it needs to be loaded to slot 0 on the device. This is accomplished by setting the attribute primary to YES (at most one script can have this set).

Element name: SCRIPT
Children: FILENAME, PATH

```
<SCRIPT
    primary="YES"
    default_channel="0"
    script_external="NO"
    >
  <FILENAME>script_0.txe</FILENAME>
  <PATH>C:/tmp/scripts</PATH>
</SCRIPT>
```

| Attribute name | Values | Description |
|---|---|---|
| primary | YES, NO | The script that should be the primary script. |
| default_channel | uint8 | The default CAN channel that the script will use. |
| script_external | YES, NO | If NO (default), the script is embedded in the binary configuration. |

Table 26: Attributes for element SCRIPT

## 11.2 Script Filename (FILENAME)

The name of the file containing the compiled script (.txe). Note that whitespace is not removed from the element before being used.

The element `FILENAME` does not have any attributes.

Element name: `FILENAME`
Children: `None`

```
<FILENAME>script_0.txe</FILENAME>
```

## 11.3 Script Path (PATH)

The path to the file containing the compiled script. Note that whitespace is not removed from the element before being used.

The element `PATH` is optional and does not have any attributes.

Element name: `PATH`
Children: `None`

```
<PATH>C:/tmp/scripts</PATH>
```

# 12 Appendix

## 12.1 Implementation limits

The XML specification does not specify provide any attribute limits, each product has its own restrictions. They are given in user guides of the respective Kvaser Memorator (2nd generation) products.

## 12.2 Higher Layer Protocols, J1939

J1939 is a set of standards defined by SAE and used in heavy-duty vehicles. If the attribute protocol is set to J1939, extended identifiers must be used.

## 12.3 Sample XML setting

This is a more complex example: a complete configuration that uses the whole range of possible elements.

```
<?xml version="1.0"?>
<!-- ELEMENTS are written with all caps -->
<!-- attributes with lower case -->
<!-- Both use underscores to separate words -->

<KVASER>
  <VERSION>2.0</VERSION>
  <BINARY_VERSION>5.0</BINARY_VERSION>
  <SETTINGS>
    <MODE
        log_all="YES"
        fifo_mode="NO"
        />
    <CANPOWER
        timeout="10000"
        />
    <COMMENT>This is my complete configuration.</COMMENT>
    <TARGET_EAN>73-30130-99010-4</TARGET_EAN>
    <TARGET_EAN>73-30130-00567-9</TARGET_EAN>
  </SETTINGS>

  <BUSPARAMS>
    <PARAMETERS
      channel="0"
      bitrate="1000000"
      tseg1="11"
      tseg2="4"
      sjw="1"
      silent="YES"
      />
```

```
<PARAMETERS
    channel="1"
    bitrate="1000000"
    tseg1="11"
    tseg2="4"
    sjw="1"
    bitrate_brs="10000000"
    tseg1_brs="5"
    tseg2_brs="2"
    sjw_brs="1"
    silent="YES"
    iso="YES"
    />
</BUSPARAMS>

<TRIGGERBLOCK>
  <TRIGGERS>
    <TRIGGER_MSG_ID
        channel="1"
        name="My_first_id_trigger"
        timeout="0"
        msgid="6"
        msgid_min="2"
        can_ext="YES"
        can_fd="NO"
        protocol="NONE"
        />

    <TRIGGER_MSG_DLC
        channel="1"
        name="My_first_dlc_trigger"
        timeout="100"
        can_fd="NO"
        dlc="7"
        dlc_min="5"
        />

    <TRIGGER_MSG_ERROR_FRAME
        channel="1"
        name="My_first_flag_trigger"
        timeout="100"
        />

    <TRIGGER_SIGVAL
        channel="1"
        name="My_first_sigval_trigger"
        timeout="-1"
        msgid="34"
        can_ext="YES"
        can_fd="NO"
        dlc="4"
        startbit="1"
        length="7"
        datatype="UNSIGNED"
```

```
        byteorder="BIG_ENDIAN"
        data="234"
        data_min="99"
        condition="ON_DATA_EQUAL_TO"
        protocol="J1939"
        />

    <TRIGGER_EXTERNAL
        channel="0"
        name="My_first_external_trigger"
        timeout="0"
        level="TRIG_EXTERNAL_LEVEL_LO_HI"
        />

    <TRIGGER_TIMER
        name="My_first_timer_trigger"
        timeout="0"
        offset="5"
        repeat="NO"
        />

    <TRIGGER_DISK_FULL
        name="My_first_diskFull_trigger"
        />

    <TRIGGER_STARTUP
        name="My_first_startup_trigger"
        />

  </TRIGGERS>

  <STATEMENTS>
    <STATEMENT
        pretrigger ="0"
        posttrigger="500">
      <EXPRESSION>
        (My_first_dlc_trigger OR My_first_sigval_trigger) AND
           My_first_id_trigger
      </EXPRESSION>
      <ACTIONS>
        <ACTION_START_LOG/>
        <ACTION_EXTERNAL_PULSE
            duration="1"
            />
        <ACTION_ACTIVATE_AUTO_TRANSMIT_LIST
            name="TransmitList1"
            />
        <ACTION_STOP_LOG/>
      </ACTIONS>
    </STATEMENT>
    <STATEMENT
        pretrigger ="0"
        posttrigger="500">
      <EXPRESSION>
```

```
          My_first_dlc_trigger OR My_first_sigval_trigger AND
             My_first_id_trigger
        </EXPRESSION>
        <ACTIONS>
          <ACTION_STOP_LOG_COMPLETELY/>
          <ACTION_DEACTIVATE_AUTO_TRANSMIT_LIST
              name="SecondList"
              />
        </ACTIONS>
      </STATEMENT>
    </STATEMENTS>
  </TRIGGERBLOCK>

  <FILTERS>
    <MESSAGE_PASS
        protocol="NONE"
        msgid="34"
        msgid_min="34"
        can_ext="NO"
        can_fd="NO"
        dlc="8"
        >
      <CHANNEL>0</CHANNEL>
    </MESSAGE_PASS>

    <MESSAGE_STOP
        protocol="J1939"
        msg_field="PGN"
        msgid="34"
        can_ext="NO"
        can_fd="NO"
        >
      <CHANNEL>0</CHANNEL>
      <CHANNEL>1</CHANNEL>
    </MESSAGE_STOP>

    <MESSAGE_COUNTING_PASS
        protocol="NONE"
        msgid="34"
        can_ext="NO"
        can_fd="NO"
        counter_threshold="7"
        counter_max="16"
        >
      <CHANNEL>0</CHANNEL>
      <CHANNEL>2</CHANNEL>
    </MESSAGE_COUNTING_PASS>

    <SIGNAL_PASS
        protocol="J1939"
        msgid="34"
        msg_field="PGN,SRC,DST"
        can_ext="NO"
        can_fd="NO"
```

```
        dlc="8"
        startbit="2"
        length="5"
        datatype="UNSIGNED"
        byteorder="BIG_ENDIAN"
        data="0x2D"
        >
      <CHANNEL>0</CHANNEL>
      <CHANNEL>4</CHANNEL>
    </SIGNAL_PASS>

    <SIGNAL_STOP
        protocol="J1939"
        msgid="0x6"
        msg_field="PGN,SRC,DST"
        can_ext="NO"
        can_fd="NO"
        startbit="0"
        length="8"
        datatype="UNSIGNED"
        byteorder="BIG_ENDIAN"
        data="0x2D"
        >
      <CHANNEL>0</CHANNEL>
      <CHANNEL>1</CHANNEL>
    </SIGNAL_STOP>

    <FLAG_PASS
        flag_std="YES"
        flag_ext="YES"
        flag_errorframe="NO"
        >
      <CHANNEL>3</CHANNEL>
      <CHANNEL>4</CHANNEL>
    </FLAG_PASS>

    <FLAG_STOP
        flag_std="NO"
        flag_ext="NO"
        flag_errorframe="YES"
        >
      <CHANNEL>3</CHANNEL>
      <CHANNEL>4</CHANNEL>
    </FLAG_STOP>
  </FILTERS>

  <TRANSMIT_LISTS>
    <TRANSMIT_LIST
        name="TransmitList1"
        msg_delay="1500"
        cycle_delay="0"
        cyclic="NO"
        autostart="NO"
        >
```

```
      <TRANSMIT_MESSAGE
          name="Message1"
          channel="0"
          />
      <TRANSMIT_MESSAGE
          name="Message2"
          channel="0"
          />
    </TRANSMIT_LIST>
  </TRANSMIT_LISTS>

  <MESSAGES>
    <MESSAGE
        name="Message1"
        msgid="0x6f"
        flags="0x5x"
        dlc="8"
        b0="0x33"
        b1="0x33"
        b2="0x33"
        b3="0x33"
        b4="0x0"
        b5="0x0"
        b6="0x0"
        b7="0x0"
        />
    <MESSAGE
        name="Message2"
        msgid="0x6e"
        dlc="8"
        can_ext="YES"
        can_fd="NO"
        can_fd_brs="NO"
        error_frame="NO"
        remote_frame="NO"
        b0="0x33"
        b1="0x33"
        b2="0x33"
        b3="0x33"
        b4="0x10"
        b5="0x0"
        b6="0x0"
        b7="0x0"
        />
  </MESSAGES>

  <!-- Max number of scripts is four -->
  <SCRIPTS>
    <SCRIPT
        primary="YES"
        default_channel="0"
        script_external="NO"
        >
      <FILENAME>script_0.txe</FILENAME>
```

```
      <PATH>C:/tmp/scripts</PATH>
    </SCRIPT>
    <SCRIPT
        primary="NO"
        default_channel="1"
        >
      <FILENAME>script_0.txe</FILENAME>
      <PATH>.</PATH>
    </SCRIPT>
    <SCRIPT
        primary="NO"
        default_channel="3"
        >
      <FILENAME>script_0.txe</FILENAME>
    </SCRIPT>
  </SCRIPTS>

</KVASER>
```

# 13 Version History

Version history for document SP_98166_memo_configuration_xml_format:

| Revision | Date | Changes |
|---|---|---|
| 1.0 | 2015-09-15 | Initial release of XML 2.0 specification |
| 1.1 | 2015-10-13 | Corrected sjw values to follow CAN specification |
| 1.2 | 2015-12-09 | Added Kvaser Memorator Pro 5xHS CB |
| 1.3 | 2016-02-22 | Added Kvaser Memorator Pro 2xHS v2. Plus some minor corrections |