Matrox Imaging Library (MIL) 10 Update 34
Release Notes (milGenTL)
August 2017

This document outlines what is new with Matrox support for GenTL and explains the current limitations and particularities.

It also presents last-minute information that did not make it into the manual or on-line help. Note that this help file serves to complement your manual. The information found in this file overrides your formally documented material.

Contents

1. MIL Driver for GenTL

The Matrox GenTL Consumer driver allows you to use a third-party GenTL library (GenTL Producer) with MIL (GenTL Consumer).

1.1 What's new in MIL 10 Update 34

1.1.1 Standards compliance

The MIL driver for GenTL supports the following standards:

- GenICam™ GenAPI Standard Version 2.4.1.
- GenICam™ GenTL Standard Version 1.5.

1.1.2 Summary of new features

The following features are new for this release:

- New API to enumerate installed GenTL producer libraries.
- New M_SYSTEM_GENTL system type.
- New API to access multiple GenICam XML description files.
- New API to enumerate GenTL modules.

- New API to display the feature browser.
- New API to hook a MIL callback to a GenICam feature change.
- New API to announce M_GRAB buffers prior to grabbing.
- New API to allocate M_CONTAINER buffers in order to grab from multi-component devices such as a 3D camera.
- New hardware-specific examples:
    - CameraEvents, located in Examples\BoardSpecific\CameraEvents\C++. This program demonstrates new MIL features for managing GenICam™ devices. This program focuses on hooking a MIL handler to asynchronous camera events.
    - ChunkMode, located in Examples\BoardSpecific\ChunkMode\C++. This program shows the use of the MdigProcess() function to perform real-time acquisition. It also enables your GenICam™ device in chunk mode (if supported).
    - EnumFeatures, located in Examples\BoardSpecific\EnumFeatures\C++. This example shows how to use MIL in order to enumerate all the features in your GenICam™ compliant device.
    - FeatureChangeHook, located in Examples\BoardSpecific\FeatureChangeHook\C++. This example shows how to use MIL in order to hook a MIL callback function to GenICam™ feature change events.
    - GenTL, located in Examples\BoardSpecific\GenTL\C++. This program demonstrates new MIL features for managing GenICam GenTL devices using the Matrox GenTL consumer system.
    - MultiComponentGrab, located in Examples\BoardSpecific\MultiComponentGrab\C++. This program shows the use of the MbufAllocDefault() and MdigProcess() function to perform real-time multi-component acquisition using M_CONTAINER buffers.

1.1.3 API enhancements

Additions to MappInquire()

- New Inquire types:
    - M_GENTL_ PRODUCER_COUNT. Specifies the number of installed GenTL producer libraries available.
      UserVarPtr data type info: address of a MIL_INT.
    - M_GENTL_PRODUCER_DESCRIPTOR + n. Specifies the selected GenTL producer libraries' path and filename, where n is the index number of the GenTL producer library.
      UserVarPtr data type info: MIL_TEXT_PTR
    - M_GENTL_PRODUCER_DESCRIPTOR_SIZE + n. Specifies the size of the selected GenTL producer libraries' description, where n is the index number of the GenTL producer library +1.
      UserVarPtr data type info: address of a MIL_INT.

Additions to MsysAlloc()

- New system type:
    - M_SYSTEM_GENTL. Allocates a MIL GenTL system.
- New combination constant for the SystemNum parameter.
    - M_GENTL_PRODUCER(n). Specifies that the n^{th} GenTL producer library will be used by the MIL system allocation, where n can be a number from 0 to 127.
- New InitFlags

- M_MIXED. Specifies to initialize any available transport layer type interfaces to work with the Matrox GenTL system.
- M_CL. Specifies to initialize Camera Link® type interfaces to work with the Matrox GenTL system.
- M_CLHS. Specifies to initialize Camera Link HS™ type interfaces to work with the Matrox GenTL system.
- M_CXP. Specifies to initialize CoaXPress® type interfaces to work with the Matrox GenTL system.
- M_GEV. Specifies to initialize GigE Vision® type interfaces to work with the Matrox GenTL system.
- M_U3V. Specifies to initialize USB 3 Vision™ type interfaces to work with the Matrox GenTL system.

Additions to MsysInquire():

- New Inquire type:
  - M_GENTL_INTERFACE_COUNT. Inquires the number of initialized GenTL interfaces available to the GenTL producer. Note that this result is limited by the GenTL producer loaded during MsysAlloc() and the setting of the InitFlag parameter.
    UserVarPtr data type: address of a MIL_INT.

Additions to MsysControl(),

- New ControlTypes and associated ControlValues:
  - M_GC_FEATURE_BROWSER: Sets whether to display the GenTL system and interface configuration information interactively. You can specify a combination value with one of the following:
    - + M_GENTL_SYSTEM. Specifies to display the GenTL system configuration information.
    - + M_GENTL_INTERFACE_NUMBER(n). Specifies to displays the n^{th} GenTL interface configuration information, where n is the n^{th} GenTL interface, and n can be a number from 0 to M_GENTL_INTERFACE_COUNT-1.
  - M_GC_FEATURE_BROWSER's control values are as follows:
    - M_DEFAULT. Same as M_OPEN.
    - M_OPEN. Opens the feature browser. You must specify a combination value from the following:
      - + M_ASYNCHRONOUS. Specifies that this function returns immediately once the feature browser window opens.
      - + M_SYNCHRONOUS. Specifies that this function is blocked until the feature browser window closes.
    - M_CLOSE. Closes the Feature Browser.

New MsysControlFeature() / MsysInquireFeature()

- Same API as MdigControlFeature() / MdigInquireFeature()
- Combination values for the values listed in the *For specifying the type of information about the feature to set and the data type returned* table:
  - M_GENTL_SYSTEM. Specifies to target the GenTL system XML.

- M_GENTL_INTERFACE_NUMBER(n). Specifies to targets the nth GenTL interface XML, where n can be a number from 0 to M_GENTL_INTERFACE_COUNT-1.

- M_FEATURE_ENUM_ENTRY_DISPLAY_NAME + n. Inquires the internally-used name of the specified enumeration entry of the feature, where n is the index into the enumerated list.

- M_FEATURE_USER_ARRAY_SIZE(). Specifies that the feature value is expressed as a string of a specified size. The M_FEATURE_USER_ARRAY_SIZE() macro passes the size of the user-allocated buffer (first passed to the MsysInquireFeature's UserVarPtr parameter). See M_FEATURE_USER_ARRAY_SIZE() in the MIL documentation.

- M_FEATURE_CHANGE_HOOK. Sets whether to enable an event to occur when the value of the specified feature changes. Once enabled, use MsysHookFunction() with M_FEATURE_CHANGE to hook a specified function to the feature change event. Repeat for each feature that you want to enable a feature change event. Alternatively, to enable an event to occur when the value of any feature changes, use MsysHookFunction() with M_FEATURE_CHANGE + M_ALL.

- Within the hook function, use MsysGetHookInfo() with M_GC_FEATURE_CHANGE_NAME to retrieve the name of the feature that caused the event.

Additions to MsysHookFunction():

- M_FEATURE_CHANGE. Hooks the function to the event that occurs when the value of a GenICam GenTL feature changes. The list of potential feature change events is set using MsysControlFeature() with M_FEATURE_CHANGE_HOOK.

  Combination values for M_FEATURE_CHANGE

  You can add the following value to M_FEATURE_CHANGE to specify to hook the specified function to all feature change events.

  - M_ALL. Specifies to hook the specified function to all feature change events.

  - M_GENTL_SYSTEM: Specifies to hook the specified function to a GenTL system feature change callback.

  - M_GENTL_INTERFACE_NUMBER(n): Specifies to hook the specified function to a n^{th} GenTL interface feature change callback, where n can be a number from 0 to M_GENTL_INTERFACE_COUNT-1.

Additions to MsysGetHookInfo():

The following allows you to retrieve information from a GenTL feature change event. These information types are only available if MsysGetHookInfo() was called from a function hooked to GenTL feature change event using MsysHookFunction() with M_FEATURE_CHANGE.

- M_GC_FEATURE_CHANGE_NAME: Retrieves the name of the GenTL feature that changed.
  UserVarPtr data type: array of type MIL_TEXT_CHAR. Required array size: MsysGetHookInfo() with M_GC_FEATURE_CHANGE_NAME_SIZE

- M_GC_FEATURE_CHANGE_NAME_SIZE: Retrieves the string length of the feature name that changed.
  UserVarPtr data type: address of a MIL_INT.

Additions to MdigInquire():

- M_GENTL_INTERFACE_INDEX. Inquires the number of GenICam GenTL camera interface that is associated to the specified digitizer.
  UserVarPtr data type: address of a MIL_INT.

- M_GENTL_STREAM_COUNT. Inquires the number of GenTL streams associated to the specified digitizer.
  UserVarPtr data type: address of a MIL_INT.

Additions to MdigControl():

- M_GC_FEATURE_BROWSER: Displays the Matrox feature browser containing the GenTL device, remote device, and stream XMLs. You can specify a combination value with one of the following:
    - M_GENTL_DEVICE. Displays the GenTL device's configuration information.
    - M_GENTL_REMOTE_DEVICE. Displays the GenTL remote device (camera) configuration information.
    - M_GENTL_STREAM_NUMBER(n). Displays the n^{th} GenTL stream configuration information, where n is a number from 0 to M_GENTL_STREAM_COUNT-1.
- M_GENTL_ANNOUNCE_BUFFER. Sets the buffer to announce to the GenTL producer, before starting the acquisition with MdigGrab(). Note that it is not necessary to announce buffers used with MdigGrabContinuous() or MdigProcess().
    - MIL buffer identifier. Specifies the buffer to announce to the GenTL producer.
- M_GENTL_REVOKE_BUFFER. Revokes a previously announced buffer. Note that acquisition must be stopped before revoking a buffer.
    - MIL buffer identifier. Specifies the buffer to revoke from the GenTL producer. Note that this buffer must have been previously announced using M_GENTL_ANNOUNCE_BUFFER.

Additions to MdigControlFeature() / MdigInquireFeature():

- New combination constants for the ControlType parameter:
    - M_GENTL_DEVICE. Specifies the GenTL device's configuration information.
    - M_GENTL_REMOTE_DEVICE (Default). Specifies the GenTL remote device's configuration information (such as, the camera).
    - M_GENTL_STREAM_NUMBER(n). Specifies the nth GenTL stream's configuration information.
- The FeatureType parameter has been changed to UserVarType. This was done to simplify writing code with MdigControl/InquireFeature(). UserVarType must always reflect the type of the pointer passed to the UserVarPtr parameter. Legacy code is transparently supported, but we recommend you update your code. Note that M_TYPE_REGISTER now becomes M_TYPE_UINT8, M_TYPE_ENUMERATION now becomes M_TYPE_INT64 or M_TYPE_STRING, and M_TYPE_COMMAND now becomes M_DEFAULT. Data type conversions are made, whenever possible, in cases where the feature's "native" data type is different than the UserVarType supplied. Regardless of a feature's "native" data type it can always be read as a string. See hardware-specific examples for details.

  The following is a list of example calls using the new UserVatType:

    - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"), M_TYPE_INT64, &Int64Var)
    - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Gain"), M_TYPE_DOUBLE, &DoubleVar)
    - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("ReverseX"), M_TYPE_BOOLEAN, &BoolVar)
    - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("PixelFormat"), M_TYPE_STRING, MIL_TEXT("Mono8"))
    - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("LUTValueAll"), M_TYPE_UINT8, Uint8Array)

        - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("AcquisitionStart"), M_DEFAULT, M_NULL)
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"), M_TYPE_INT64, &Int64Var)
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Gain"), M_TYPE_DOUBLE, &DoubleVar)
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("ReverseX"), M_TYPE_BOOLEAN, &BoolVar)
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE + M_STRING_SIZE, MIL_TEXT ("PixelFormat"), M_TYPE_MIL_INT, &MilIntVar)
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("PixelFormat"), M_TYPE_STRING, MilTextCharArray)
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("LUTValueAll"), M_TYPE_UINT8, Uint8Array)

- M_FEATURE_USER_ARRAY_SIZE() can now be used with MdigInquireFeature when the data type returned is a string or an array of bytes (register). The M_FEATURE_USER_ARRAY_SIZE() macro is used to pass the size of the user-allocated buffer passed to MdigInquireFeature's UserVarPtr parameter. M_FEATURE_USER_ARRAY_SIZE() is passed using the UserVarType parameter. See GenTL hardware-specific example for sample usage.

  The following is a list of example calls using M_FEATURE_USER_ARRAY_SIZE():

    - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("PixelFormat"), M_TYPE_STRING + M_FEATURE_USER_ARRAY_SIZE(N), MilTextCharArray); N being equal to the number of MIL_TEXT_CHAR in the MilTextCharArray.
    - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("LUTValueAll"), M_TYPE_UINT8 + M_FEATURE_USER_ARRAY_SIZE(N), Uint8Array); N being equal to the number of Uint8 in the Uint8Array.

- M_FEATURE_ENUM_ENTRY_DISPLAY_NAME can now be used to inquire possible enumeration string entry to use for display purposes. See M_FEATURE_ENUM_ENTRY_NAME in the MIL documentation.

- M_FEATURE_VALUE_AS_STRING is now deprecated.

    - To read a feature's value as a string and get the required string length use:
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE + M_STRING_SIZE, MIL_TEXT ("Width"), M_TYPE_MIL_INT, &MilIntVar);
    - To read a feature's value as a string use:
        - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"), M_TYPE_STRING+M_FEATURE_USER_ARRAY_SIZE(ArraySize), MilTextCharArray);
    - To write a feature's value from a string use:
        - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"), M_TYPE_STRING, MIL_TEXT("1024"));

- M_FEATURE_CHANGE_HOOK. Identifies the specified FeatureName to trigger the M_FEATURE_CHANGE hook callback. You must be hooked to the M_FEATURE_CHANGE hook type using MdigHookFunction().

Additions to MdigHookFunction():

- M_FEATURE_CHANGE. Hooks the function to the event that occurs when a specified feature changes. The list of potential feature change events is set using MdigControlFeature() with M_FEATURE_CHANGE_HOOK.

  You can add the following value to the above-mentioned value to specify to hook the specified function to all featue change events.

  - M_ALL. Specifies to hook the specified function to all feature change events.
  - M_GENTL_DEVICE: Specifies to hook the specified function to a GenTL device feature change event.
  - M_GENTL_REMOTE_DEVICE: Specifies to hook the specified feature change to a GenTL remote device feature change event.
  - M_GENTL_STREAM_NUMBER(n): Specifies to hook the specified feature change to the n^{th} GenTL stream feature change event.

Additions to MdigGetHookInfo():

The following allows you to retrieve information from a GenICam feature change event. These information types are only available if MdigGetHookInfo() was called from a function hooked to GenICam feature change event using MdigHookFunction() with M_FEATURE_CHANGE.

- M_GC_FEATURE_CHANGE_NAME: Retrieves the name of the GenICam feature that changed.
  UserVarPtr data type: array of type MIL_TEXT_CHAR
  Required array size: MdigGetHookInfo() with M_GC_FEATURE_CHANGE_NAME_SIZE.
- M_GC_FEATURE_CHANGE_NAME_SIZE: Retrieves the string length of the feature name that changed.
  UserVarPtr data type: *address of a MIL_INT.*

New function:
MbufAllocDefault(MIL_ID SystemId, MIL_ID ReferenceId, MIL_INT64 Attribute, MIL_INT64 ControlFlag, MIL_INT64 ControlValue, MIL_ID* BufIdVarPtr):

This function allocates a MIL data buffer on the specified system.

- SystemId: Specifies the MIL system on which to allocate the buffer. This parameter should be set to a MIL GenTL system.
- ReferenceId: Specifies the source from which retrieve additional information about the buffer to allocate (such as, the size of the buffer and number of bands). The ReferenceId can be a buffer, a digitizer, or a display. M_DEFAULT will retrieve buffer information from the MilConfig utility.
- Attribute: Specifies usage of the buffer.
  - M_CONTAINER: Specifies a container to store either multiple component containers, or a list of related MIL buffers.
    - M_3D_SCENE: Specifies a multi-component container. This container can be used to store images representing data coming from a 3D camera.

      You can use one or more of the following values in combination with each other and with the above-mentioned values, to specify the container's usage.

      - M_DISP: Specifies a container that can be displayed.
      - M_GRAB: Specifies a container into which to grab data.
      - M_PROC: Specifies a container that can be processed.
- ControlFlag: Specifies the operation to perform.

    - M_DEFAULT: Specifies to perform buffer allocation using the information specified. The ControlValue must be set to M_DEFAULT.

- ControlValue: Parameter value associated to the ControlFlag parameter. See ControlFlag parameter for more details.
- BufIdVarPtr: Specifies the address of the variable in which to write the buffer or container identifier.
- Return value: The returned value is the buffer or container identifier. If allocation fails, M_NULL is returned.

Additions to MbufControl():

- M_GC_FEATURE_BROWSER: Displays the Matrox feature browser containing the GenTL buffer configuration information. You can specify a combination value with the following:
    - + M_GENTL_BUFFER. Displays the GenTL buffer configuration information.

Additions to MbufInquire() for M_CONTAINER buffers:

- M_COMPONENT_COUNT: Inquires the number of components in the container. UserVarPtr Data type: address of a MIL_INT.
- M_COMPONENT_ID(MIL INT ComponentType): Inquires the identifier of the component in the container.

  Parameters
    - ComponentType. This parameter specifies the type of buffer for which the identifier must be returned. ComponentType can take one of the following MIL pre defined constants. However if the camera supports the ComponentIDValue SNFC feature then the ComponentType must be set to the camera's ComponentIDValue associated to the component to get.
        - M_INTENSITY: Returns the MIL_ID of the buffer representing the intensity of the image.
        - M_INFRARED: Returns the MIL_ID of the buffer representing an image acquired in the infrared band.
        - M_ULTRAVIOLET: Returns the MIL_ID of the buffer representing an image acquired in the ultraviolet band.
        - M_RANGE: Returns the MIL_ID of the buffer representing 3D range information.
        - M_DISPARITY: Returns the MIL_ID of the buffer representing the disparity information of a stereoscopic 3D camera.
        - M_CONFIDENCE: Returns the MIL_ID of the buffer representing to the confidence map data of a 3D camera.
        - M_SCATTER: Returns the MIL_ID of the buffer associated to the scatter data. This represents how much light is scattered around the reflected light.
        - M_REFLECTANCE: Returns the MIL_ID of the buffer associated to the reflected intensity of the image.
- M_COMPONENT_ID_BY_INDEX(MIL_INT ComponentCount): Inquires the identifier of the component in the container at a specified index.

  Parameters
    - ComponentCount. This parameter specifies the index of the component in the container. This value can be from 0 to M_COMPONENT_COUNT-1.
- M_DATA_INFO_TYPE: Inquires the type of information stored in the buffer: UserVarPtr should be set

to the address of a MIL_INT.

- For M_CONTAINER buffers:
  - M_3D_SCENE: Specifies that the container is a multi-component buffer representing 3D data.
- For M_IMAGE buffers:
  - M_NULL: Specifies that the image buffer is not part of a container buffer.
  - M_INTENSITY: Specifies that the buffer is part of a container buffer and contains the intensity of the image.
  - M_INFRARED: Specifies that the buffer is part of a container buffer and contains an image acquired in the infrared band.
  - M_ULTRAVIOLET: Specifies that the buffer is part of a container buffer and contains an image acquired in the ultraviolet band.
  - M_RANGE: Specifies that the buffer is part of a container buffer and contains 3D range information.
  - M_DISPARITY: Specifies that the buffer is part of a container buffer and contains the disparity information of a stereoscopic 3D camera.
  - M_CONFIDENCE: Specifies that the buffer is part of a container buffer and contains the confidence map data of a 3D camera.
  - M_SCATTER Specifies that the buffer is part of a container buffer and contains the associated to the scatter data.
  - M_REFLECTANCE: Specifies that the buffer is part of a container buffer and contains the reflected intensity of the image.

New MbufControlFeature() / MbufInquireFeature()

- Same API as MdigControlFeature() / MdigInquireFeature()
- Combination constants for the ControlType / InquireType parameter

  The following value might be combined with the ControlType / InquireType values to specify to control/Inquire the GenTL buffer configuration information.

  - M_GENTL_BUFFER. This is the default value. Specifies to control/inquire the GenTL buffer configuration information.
- M_FEATURE_CHANGE_HOOK. Identifies the specified FeatureName to trigger the M_FEATURE_CHANGE hook callback. You must be hooked to the M_FEATURE_CHANGE hook type using MbufHookFunction().

Additions to MbufHookFunction():

- M_FEATURE_CHANGE Hooks the function to the event that occurs when the value of a GenTL buffer feature changes. To enable an event to occur when the value of a specific feature changes, use MbufControlFeature() with M_FEATURE_CHANGE_HOOK set to M_ENABLE. Repeat for each feature that you want to enable a feature change event. To enable an event to occur when the value of any feature changes, use M_FEATURE_CHANGE + M_ALL.

  Within the hook function, use MbufGetHookInfo() with M_GC_FEATURE_CHANGE_NAME to retrieve the name of the feature that caused the event.

  You can add the following value to the above-mentioned value to specify to hook the specified function to all featue change events.

- M_ALL. Specifies to hook the specified function to all feature change events.
- M_GENTL_BUFFER: Specifies to hook the specified function to GenTL buffer feature change events.

Additions to MbufGetHookInfo():

The following allows you to retrieve information from a GenTL feature change event. These information types are only available if MbufGetHookInfo() was called from a function hooked to GenTL feature change event using MbufHookFunction() with M_FEATURE_CHANGE.

- M_GC_FEATURE_CHANGE_NAME: Retrieves the name of the GenICam feature that changed.
  UserVarPtr data type: array of type MIL_TEXT_CHAR.
  Required array size: MdigGetHookInfo() with M_GC_FEATURE_CHANGE_NAME_SIZE
- M_GC_FEATURE_CHANGE_NAME_SIZE: Retrieves the string length of the feature name that changed.
  UserVarPtr data type: array of a MIL_INT.

---

2. Supported operating systems

This section lists all the operating systems that the Matrox GenTL system supports.

- 32-bit Windows® 7.
- 64-bit Windows® 7.
- 32-bit Windows® 8.1.
- 64-bit Windows® 8.1.
- 32-bit Windows® 10.
- 64-bit Windows® 10.

---

3. Example locations (in the help file)

In the help file, the location information written at the top of examples might not be up-to-date. Use MIL Example Launcher to find an example on disk.

---

4. Last minute information

A problem has been found when MIL 10 Processing Pack 1 is installed after MIL 10 Update 34. The MIL 10 Processing Pack 1 installer will overwrite MIL import libraries that are provided by MIL 10 Update 34. The following functions will therefore no longer link:

- MsysControlFeature
- MsysInquireFeature

- MbufControlFeature
- MbufInquireFeature

To resolve this issue, re-install MIL 10 Update 34 or make sure to install MIL 10 Processing Pack 1 before installing MIL 10 Update 34.