

Application Note

**Establishing a MONA 3G
Multimedia Session with
the Dialogic[®] 3G-324M API**

Establishing a MONA 3G Multimedia Session with the Dialogic® 3G-324M API

Application Note

Executive Summary

In 3G mobile networks, the protocol used to deliver real-time conversational video is the 3G-324M protocol. The 3G-324M protocol is used because it provides reliable end-to-end video communication in error-prone wireless networks. Because the 3G-324M protocol is a peer-to-peer protocol, a 3G-324M video endpoint, including a multimedia server, must “talk” the protocol language or pass through a 3G gateway that can break out the audio and video streams. Thus, developing a multimedia server for the 3G video telephony network requires a 3G gateway front-end server or endpoint capable of 3G multimedia services.

Although the Dialogic-enhanced 3G-324M API, called the Dialogic® 3G-324M API, can be used to create both 3G gateway and 3G multimedia server applications, this application note focuses on how the Dialogic 3G-324M API can be used to create a 3G multimedia application without an external gateway, which can result in savings on hardware resources and a reduced need for external gateway equipment.

This application note provides an overview of the 3G-324M standard, lists the Dialogic® components that can be used to create a 3G multimedia session, details the steps an application performs to establish a 3G multimedia session using the Dialogic® 3G-324M API, and provides procedures to enable a feature in the Dialogic 3G-324M API called Media Oriented Negotiation Acceleration (MONA), which can reduce the session setup time to successfully connect to a 3G-324M endpoint.

Establishing a MONA 3G Multimedia Session with the Dialogic® 3G-324M API

Application Note

Table of Contents

Introduction	3
The 3G-324M Specification	3
3G-324M	3
Call Signaling	4
H.324 Base Protocol	4
H.223 Multiplexer/Demultiplexer Protocol and H.223 Bitstream	4
H.245 Session Control Protocol	5
Media Components	6
3G Call Breakdown	6
H.324 Annex K - MONA	7
Media Preconfigured Channel	7
Accelerated Connection Procedure	7
Dialogic® 3G-324M API	7
Dialogic® 3G-324M (m3g) Library	7
Mux3G Firmware	8
Basics of a 3G-324M Session	8
Overview	8
Device Abstractions	9
Step 1. Initializing the Devices	11
- Initializing the m3g Device	11
- Enabling MONA Events	13
- Initializing the mm Device	13
Step 2. Connecting the Devices	14
- Connecting Networks	14
- Connecting Audio and Video Media Ports	16
Step 3. Starting a 3G Call — Establishing the Bearer Channel	18
Step 4. Establishing a 3G-324M Session	18
- Establishing a 3G-324M Session using MONA	18
- Establishing Standard H245 Open Logical Channels	19
Step 5. Exchanging Media	24
- StartMedia Approach	24
- ModifyMedia Approach	24
- Starting Multimedia Play/Record	25
Step 6. Terminating a 3G-324M Session	27
- Stopping Multimedia Play/Record	27
- StopH.245 - Terminate the H.245 Session	27

Establishing a MONA 3G Multimedia Session with the Dialogic® 3G-324M API

Application Note

Step 7. Ending a 3G Call - Disconnecting the Bearer Channel	28
- Loss of Session	28
Step 8. Disconnecting Devices	28
- Disconnecting Media Port Connections	28
- Disconnecting Network Device	29
Step 9. Closing Devices	29
Step 10. Exiting an Application	29
Summary	30
For More Information	30

Establishing a MONA 3G Multimedia Session with the Dialogic® 3G-324M API

Application Note

Introduction

Dialogic® 3G-324M video telephony-enabled products provide software resources that can be used to enable 3G video services. In 3G mobile networks, the protocol used to deliver real-time conversational video is the 3G-324M protocol, which provides reliable end-to-end video communication in error-prone wireless networks. The 3G-324M standard, defined by the 3rd Generation Partnership Project (3GPP), includes a number of sub-protocols that establish the video session over a circuit-switched connection. Because the 3G-324M protocol is a peer-to-peer protocol, a 3G video endpoint, including a multimedia server, must “talk” the protocol language or pass through a 3G gateway that can break out the audio and video streams. Thus, developing a multimedia server for the 3G video telephony network requires a 3G gateway front-end server or endpoint capable of 3G multimedia services. Although the Dialogic® 3G-324M API can be used to create both 3G gateway and 3G multimedia server applications, this application note focuses on how the Dialogic 3G-324M API can be used to create a 3G multimedia application without an external gateway.

This application note provides an overview of the 3G-324M standard, lists the Dialogic® components that can be used to create a 3G multimedia session, provides the steps an application performs to establish a 3G multimedia session using the Dialogic 3G-324M API, and provides procedures to enable a feature in the Dialogic 3G-324M API called Media Oriented Negotiation Acceleration (MONA), which can reduce the session setup time to successfully connect to a 3G-324M endpoint by providing techniques to establish media quickly.

Example code that demonstrates 3G-324M multimedia capability is included within the 3G-324M Multimedia Gateway Demo that is provided in the Dialogic® product release (see the *For More Information* section). The document that describes the operation and configuration of the 3G-324M Multimedia Gateway Demo to support this multimedia mode is a separate *Dialogic® 3G-324M Multimedia Gateway Demo Guide* (see the *For More Information* section).

The flexibility of Dialogic® 3G-324M video telephony products enables application developers to produce 3G multimedia applications in one system without the need for a separate 3G Gateway and IP Media Server in two independent servers. This can benefit developers who are seeking to save on hardware

resources, to reduce the need for external gateway equipment, or to embrace software control over aspects of a 3G multimedia system. In order to offer 3G multimedia services with one application, Dialogic provides the following APIs:

- Dialogic® 3G-324M API Library (m3g_ library)
- Dialogic® Multimedia API Library (mm_ library)
- Dialogic® Device Management API Library (dev_ library)

To set up a 3G multimedia session, the application establishes a 3G-324M session, and terminates the 3G multimedia data. This can be accomplished by using the m3g_ library API to set up a 3G session and the mm_ library API to play or record video. The dev_ library API is used to connect the m3g and mm devices so that 3G-324M multiplexed audio and video data can be terminated for play or record at the multimedia device.

The Dialogic 3G-324M API and system software can simplify 3G-324M session establishment for application developers, such as by reducing what is generally a protocol-intensive process into a more manageable procedure. That, in turn, can reduce the burden of an application designer to understand the lowest level protocol specifics.

Although this application note provides some insight into Dialogic® multimedia capabilities applicable to 3G devices, it does not go into detail on all capabilities of the Dialogic® multimedia devices. To learn more about Dialogic multimedia devices, refer to the *Dialogic® Multimedia Programming Guide and Multimedia API Library Reference* (see the *For More Information* section). Also, example code that demonstrates multimedia device capability is included in the Multimedia Demo that is provided in the Dialogic product release (see the *For More Information* section).

The 3G-324M Specification

3G-324M

The 3G-324M standard has been approved by both the 3GPP and 3GPP2 as a solution for multimedia communication over existing PSTN circuit switched networks. The 3G-324M specification is an umbrella of standards that enable 3G real-time multimedia services over PSTN bandwidths. A goal of the 3G-324M spec is to reuse the circuit switched network for cellular multimedia services with built-in error resiliency. This provides the ability for transmission of real-time conversational multimedia services between 3G-324M endpoints over wireless

links. A 3G-324M call involves a network call and a 3G-324M session. A number of different standards are used to establish a 3G-324M session, and the building blocks that make up a 3G endpoint are shown in Figure 1. A 3G endpoint includes a network interface, signaling channel, multiplexer, and media components.

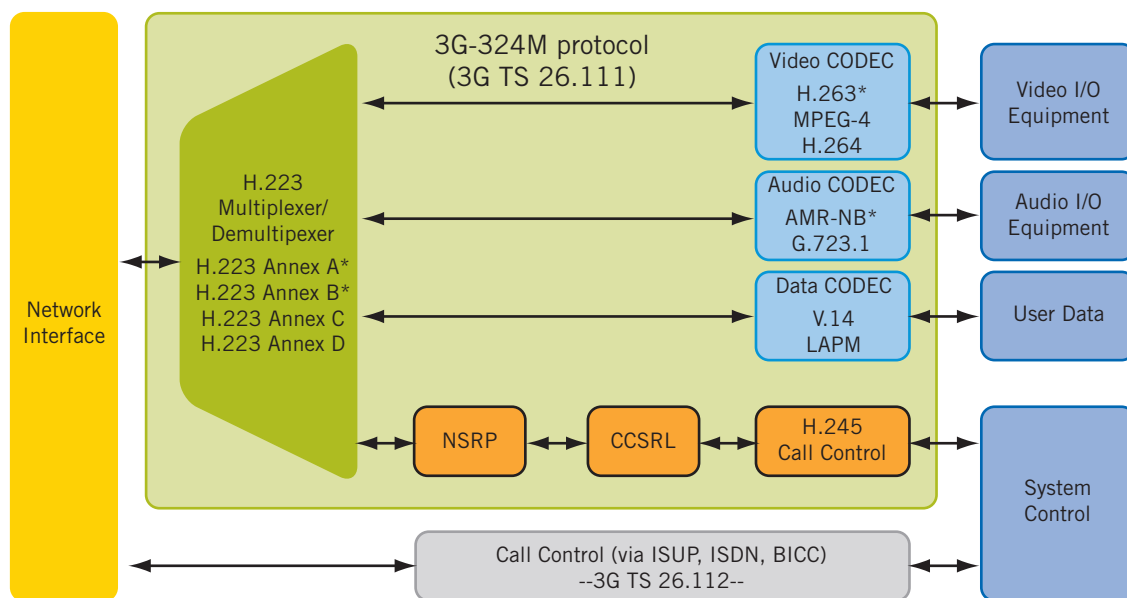


Figure 1. Standards Used to Establish 3G-324M Session

Call Signaling

Call signaling is done in the PSTN network to set up a digital (64 kbps) bearer channel between two 3G-324M endpoints. The 3G-324M specification defines a hierarchy of Releases that address next generation network infrastructure. Release 99 (R99) of the 3G-324M specification describes a 3G-324M session in a traditional TDM network. Release 4 (R4) of the spec describes a 3G-324M session in an all IP network. In a Release 99 PSTN network, the call setup of the bearer channel is normally accomplished using ISUP (SS7) or ISDN protocol. The digital bearer channel contains no proprietary framing of data (such as A-law or μ -law companding); it is a clear channel transparent link. In a Release 4 IP network, it is expected that the call will be established using Bearer Independent Call Control (BICC). The bearer channel in the Release 4 IP network is an Nb UP RTP connection. Following successful call signaling, the network call is connected, a transparent data bearer channel is established, and the flow of H.223 bitstream data begins the 3G-324M session (see Figure 2).

H.324 Base Protocol

H.324 is the base protocol standard for providing real-time multimedia video telephony. It includes the H.223 multiplex protocol and H.245 session control protocol. The H.324 standard was amended for mobile devices with the H324M (also known as Annex C) mobile extension. H.324 Annex K adds support for Media Oriented Negotiation Acceleration (MONA).

H.223 Multiplexer/Demultiplexer Protocol and H.223 Bitstream

The H.223 standard is an end-to-end low bit rate protocol in the form of a bitstream between two multimedia endpoints. It provides a method to combine multiple media channels in one physical connection with bit-error resiliency and retransmission built in. This is important for wireless applications where transmission errors are prevalent. The H.223 standard is used for multiplexing

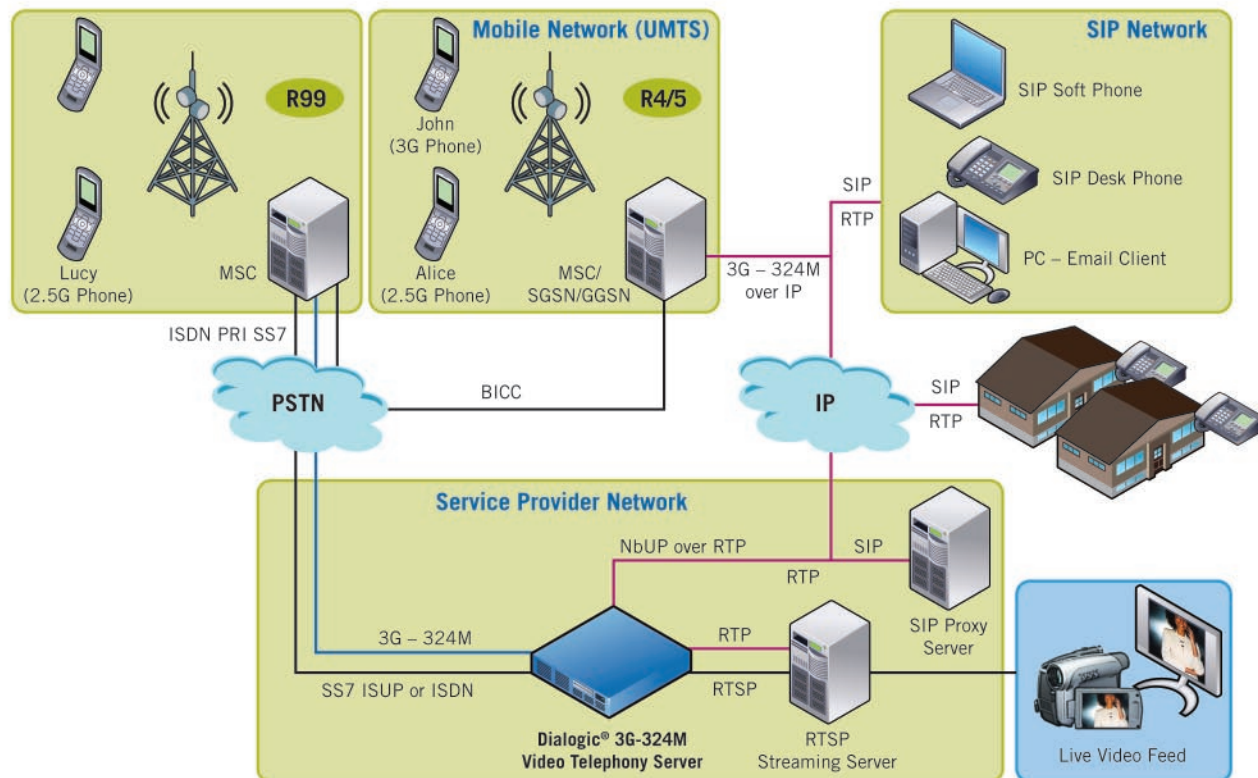


Figure 2. Call Signaling

and demultiplexing control, audio, and video logical channels of the 3G-324M session into a single bitstream. The bitstream can then be sent over a physical network IP or TDM bearer channel connection.

When the session is first connected, a mobile-level detection procedure is started that sets up the frame synchronization between endpoints. Each endpoint sends a frame synchronization flag pattern that corresponds to the highest multiplexer level at which it is capable of operating. At the same time, it detects the incoming flag pattern from its peer. If the synchronization flag pattern the endpoint receives represents a lower multiplex level than it is transmitting, the endpoint changes its transmitted multiplex level to the detected lower level. Each side retransmits its respective synchronization flag patterns until a level is reached that both endpoints support. This protocol helps set up the error resiliency level. Once the multiplexer layer is determined, the H.245 control channel (also known as Logical Channel 0) is opened automatically between endpoints.

H.245 Session Control Protocol

The H.245 standard is the protocol used to provide session control between 3G-324M endpoints. The H.245 protocol provides the method for endpoints to exchange media capabilities, to open and close logical channels for media, and to specify the content of the media channels when they are opened.

The H.245 protocol begins with Master Slave Determination (MSD) exchange and Terminal Capabilities Set (TCS) exchange. MSD exchange determines that the endpoint with the highest terminal type becomes the master. The endpoint deemed master is given priority in resolving conflicts during logical channel establishment. During TCS exchange, each endpoint specifies its receiver capabilities so that its remote peer may subsequently open logical media channels and transmit in a media and multiplexer format that is supported by both endpoints. Unidirectional logical channels are opened by the local endpoint (transmitter) based on the capabilities that the remote endpoint (receiver) provides in the capabilities exchange. Unidirectional logical channels are opened in the

forward direction, to specify audio or video media format from local to remote endpoint. Each endpoint opens one forward logical channel for transmitting audio and one forward logical channel for video. Logical channels identify the media capabilities that are used to specify CODEC type and format, and multiplexer capabilities of the particular media channel.

Media Components

The audio and video media types specified in the media channel are defined by the 3G-324M specification. For audio, the mandatory CODEC is AMR, which is a coder that dynamically adjusts the number of bits used for voice and error resiliency based on the network conditions. Also supported in the standards are G.723 and AMR-WB. For video, the H.263 CODEC is mandatory and must be supported by all endpoints. Also, a strongly preferred option is the MPEG4 coder that has better error resiliency and higher efficiency in low bandwidth applications. Gaining popularity, the H.264 CODEC has been added to the list of preferred options for video CODECs.

3G Call Breakdown

The 3G-324M session requires a few steps to set up a 3G-324M multimedia call (see Figure 3). The following steps highlight some of the exchanges that occur between endpoints to establish a media session:

1. A training phase determines the H.223 multiplexing level.
2. The MSD and TCS messages are exchanged.
3. Each endpoint opens video and audio logical channels in the forward direction that include the type of media that is defined for that channel.
4. Multiplex Table Entries are transferred to define how the logical channel data is organized into multiplexer frames.
5. Media is exchanged between the endpoints.

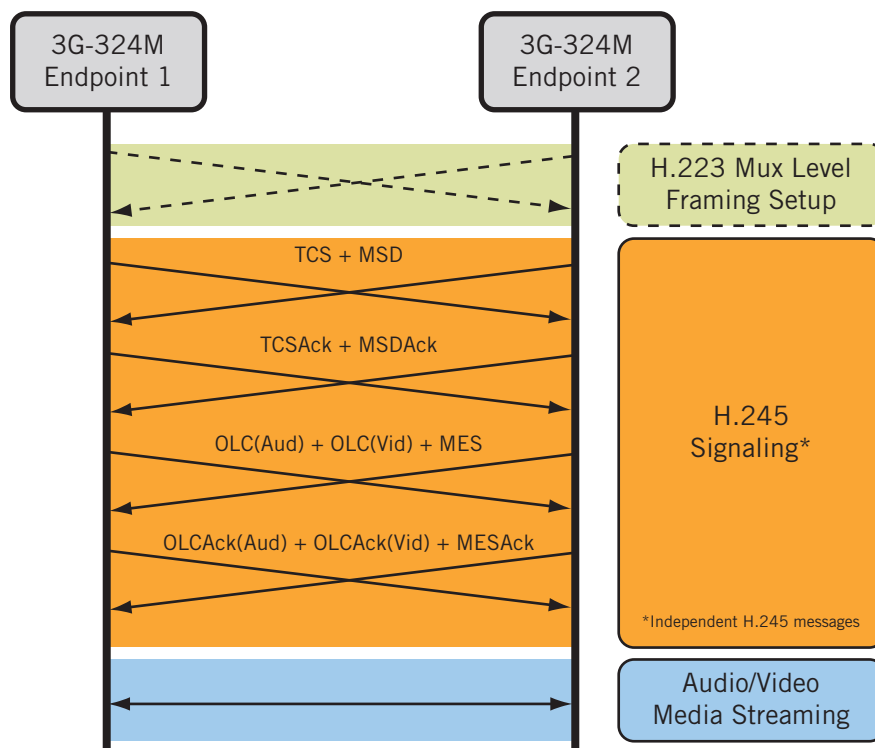


Figure 3. 3G Call Breakdown

H.324 Annex K - MONA

The Media Oriented Negotiation Acceleration (MONA) standard is a group of complementary procedures designed to significantly reduce delay in H.324 call setup time by establishing media as early as possible. (This can be in the form of an early exchange of media preference for immediate media establishment or by establishing media before protocol acknowledgements. As a result, the video caller experiences video connection in roughly the same time as a voice call without having to wait on full protocol exchange.)

The procedures include Media Preconfigured Channels (MPC), Accelerated Connect Procedure (ACP), and Signaling Preconfigured Channel (SPC). The MONA standard implemented by Dialogic uses MPC and ACP procedures, which classifies the Dialogic® 3G-324M software as a Class II MONA terminal. MONA provides a flexible accelerated channel setup method that depends on an initial exchange of preferences (Preference Messages [PM]) and the execution of a common inference algorithm. MONA also provides fast preconfigured channel setup mechanisms, which do not wait for standard H245 message acknowledgements, but provide a fallback if the initial media transmission attempts do not succeed. With MONA, media channels are typically ready for streaming in less than a second, compared with six to eight seconds using the standard H.245 Open Logical Channel establishment.

Media Preconfigured Channel

The Media Preconfigured Channel (MPC) procedure is used to establish media channels quickly within a 3G-324M session. Terminals indicate support for MONA MPC by inserting MONA Preference Messages (PMs) within special framing flags before the H.223 Mux level detection procedure frame synchronization. The MPC call setup procedure allows for media to be established as soon as the PM is sent. Media can be received on MPC channels as soon as the PM is received. Results from the PM exchange determine whether receiving and transmitting MPC channels are established and what type of media is supported by the 3G-324M endpoint. It is possible that all or a subset of the media channels are established using the MPC procedure. After media is established, MPCs are managed identically to LCs.

Accelerated Connection Procedure

The Accelerated Connection Procedure (ACP) allows media streaming to begin much earlier in the 3G-324M call. The ACP procedure uses H.245 to establish Logical Channels (LC) in the same way that is used in standard H.245 OLC establishment procedure; however, ACP procedures do not wait on acknowledgement messages (TCSAck, MSDAck) before initiating logical channels and the resulting media. OLCs and media may be transmitted once the TCS is received and before a TCSAck and MSDAck are received. Similarly, media may be initiated before receiving an OLCAck. The ACP procedure speeds the OLC procedure to establish media by not waiting on H245 roundtrip message delay.

Dialogic® 3G-324M API

The Dialogic 3G-324M API provides the ability to control and manage a collection of 3G-324M endpoints. The Dialogic 3G-324M API provides an abstraction of a 3G-324M multimedia multiplexer for wireless multimedia application development, allowing connection between multimedia on a circuit switched network and multimedia on a packet-switched network. It provides an application interface to the H.245 control session and a means to establish audio and video streams. The 3G-324M API enables the multiplexing and demultiplexing of the audio and video streams to/from a data bearer channel through its software component, the m3g device. The Dialogic 3G-324M software can be visualized in two parts (see Figure 4):

- **Dialogic 3G-324M (m3g) API library** — Provides software component device abstraction and application interface
- **Mux3G firmware** — Handles the low-level 3G-324M protocol exchange

Dialogic® 3G-324M (m3g) Library

The Dialogic 3G-324M (m3g) API Library is an R4 technology-specific library that abstracts the H.245 session control, an audio packet interface, and a video packet interface that exposes them as three separate R4 m3g devices. The m3g control device is used to represent the aggregate interface that connects to the network transport. The m3g audio and m3g video devices are used to control the audio and video media stream connections and to set the media capabilities. The

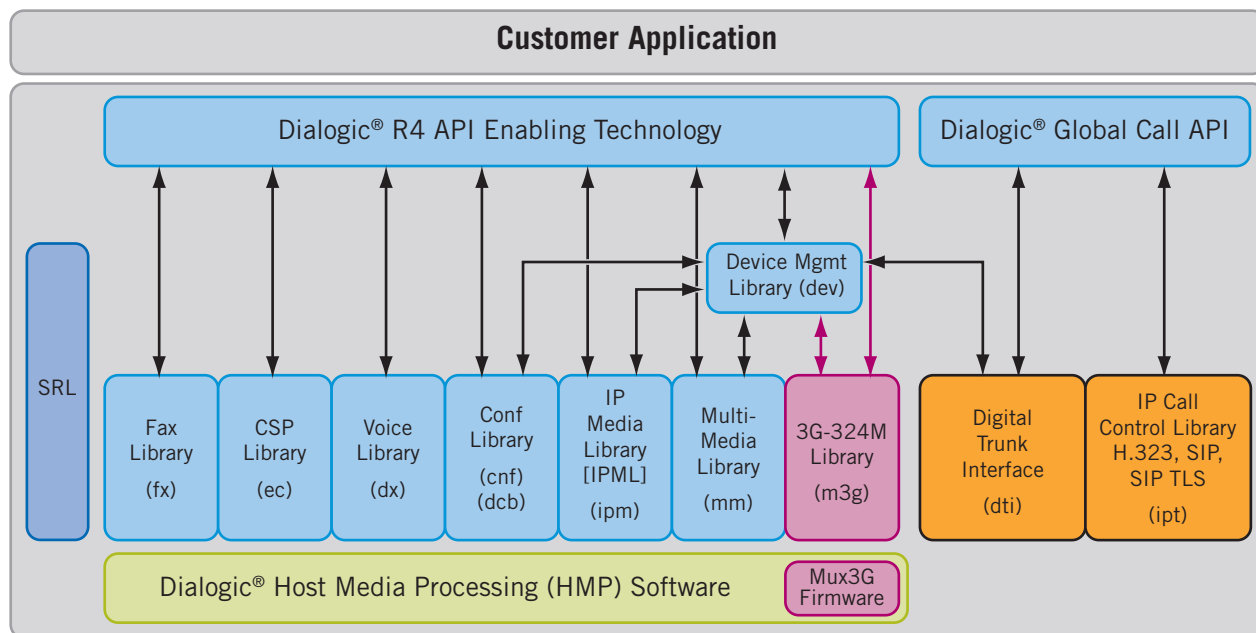


Figure 4. Dialogic 3G-324M Software

Dialogic 3G-324M API Library can be used along with other Dialogic® R4 API libraries, such as the Dialogic® Device Management API Library for device connection, the Dialogic® Multimedia API Library for audio/video multimedia Play/Record, and the Dialogic® IP Media Library API for IP Streaming.

Mux3G Firmware

The Mux3G firmware is the 3G session protocol and multiplexing/demultiplexing engine. It is the Dialogic® system software component that controls the 3G-324M protocol stack implementation. The Mux3G firmware performs the aspects of the low-level 3G-324M protocols while also providing user session input and session progress eventing through the Dialogic 3G-324M Library. The firmware carries out the protocol between the local device and the remote 3G-324M endpoint. It provides the minimum interface necessary for application 3G session control, without requiring the application developer to have detailed protocol-specific knowledge.

Basics of a 3G-324M Session

Overview

In order to establish a 3G multimedia session, an application must make the proper internal device connections, establish a 3G-324M session with the remote 3G endpoint, and terminate multimedia by playing the audio/video to the remote 3G endpoint or recording the audio/video from the remote 3G endpoint.

For purposes of this application note, the guideline for those opting to develop a complete 3G multimedia application, including initialization, session establishment, device cleanup, and session termination consists of ten steps. The first five steps initialize and establish a 3G multimedia session:

1. Initializing the Devices
 - Initializing the m3g Device
 - Enabling MONA Events
 - Initializing the mm Device
2. Connecting the Devices
 - Connecting Networks
 - Connecting PSTN Network
 - Connecting IP Network
 - Connecting Audio and Video Media Ports
 - Retrieving Device Media Ports
 - Connecting Media Ports between Devices
 - Transcoding versus “Native” Connection
3. Starting a 3G Call — Establishing the Bearer Channel
4. Establishing a 3G-324M Session
 - Establishing a 3G-324M Session using MONA
 - Establishing Standard H245 Open Logical Channel
5. Exchanging Media
 - StartMedia Approach
 - ModifyMedia Approach
 - Starting Multimedia Play/Record

The last five steps terminate and exit the 3G multimedia session:

6. Terminating a 3G-324M Session
 - Stopping Multimedia Play/Record
 - StopH.245 - Terminating the H.245 Session
7. Ending a 3G Call - Disconnecting the Bearer Channel
 - Loss of Session
8. Disconnecting Devices
 - Disconnecting Media Port Connections
 - Disconnecting Network Device
9. Closing Devices
10. Exiting an Application

Device Abstractions

The devices used to create a 3G multimedia session are the “Mux3G” device (m3g device) and the “multimedia” device (mm device). The following information on these devices’ abstractions is useful to review before starting step 1.

m3g Device Abstraction

The Dialogic 3G-324M API Library is an API that provides device handles to be used to control the aspects of the 3G-324M multiplexer endpoint. The device abstractions include board, control, audio, and video devices (see Figure 5).

An “m3g device” represents one instance of a 3G-324M multiplexer endpoint and terminates the 3G-324M peer-to-peer protocol. The m3g device provides a connection to the aggregate data on the 3G network bearer channel and internal connections for audio and video media streams. When the m3g device receives aggregate data from the 3G network bearer channel, it demultiplexes the data into H.245 control messages, and audio and video media streams. In the reverse direction, audio and video media streams are multiplexed together with H.245 control messages and sent out via the aggregate 3G network bearer channel to the remote 3G-324M endpoint.

- **Board Device (m3gBm)** — The board device handle can be used to set “board” level parameters for all m3g instances at one time. Currently, only one board (that is, m3gB1) is supported. The m3gB1 handle can be used with the m3g_SetParm() function to set parameters or the m3g_EnableEvents() function to enable unsolicited events.
- **Control Device (m3gBmTn)** —The control device handle is the primary handle for 3G-324M endpoint control. The control device is used to manage the H.245 control channel interface (logical channel 0) and to set up the capabilities of the H.223 multiplexer/demultiplexer. The control device is also used to control the ports that are connected to the 3G network aggregate data source.
- **Audio Device (m3gBmTn:AUDIO1)** —The audio device handle is used to represent the audio connection to/from the multiplexer. The audio device is used to designate the audio terminal capabilities of the m3g device and to control the audio media streaming ports.
- **Video Device (m3gBmTn:VIDEO1)** —The video device handle is used to represent the video connection to/from the multiplexer. The video device is used to designate the video terminal capabilities of the m3g device and to control the video media streaming ports.

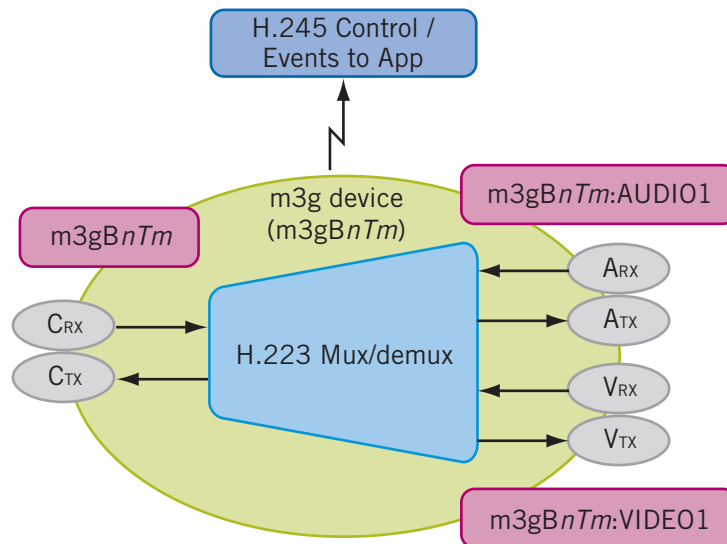


Figure 5. m3g Device Abstraction

mm Device Abstraction

The Dialogic Multimedia API Library is an API that provides a termination used to play and record digitized multimedia in support of applications providing video services, such as video mail, video color ring, video caller ID, and video location-based services (see Figure 6).

An “mm device” represents one instance of a multimedia endpoint. The mm device includes both audio and video play/record capabilities that can be used together, for example to maintain audio/video synchronization, or separately, for example to provide video plus an audio track. The mm device includes internal transmit (Tx) and receive (Rx) ports for audio and video connections to other Dialogic® multimedia software devices that provide transport to the appropriate network.

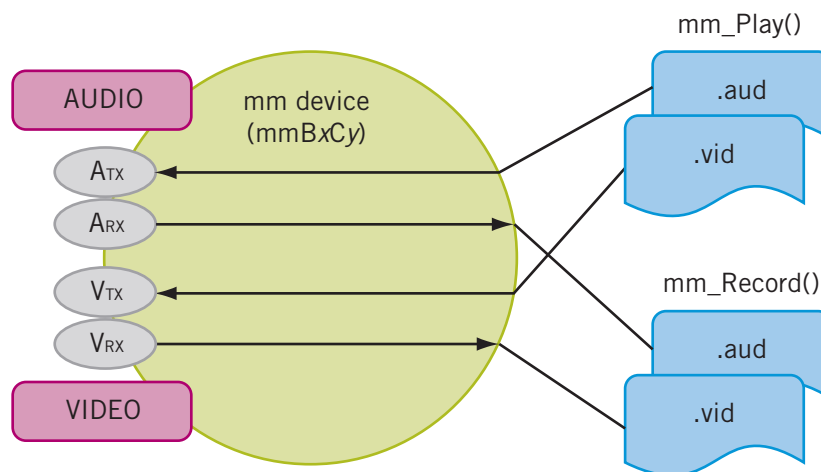


Figure 6. mm Device Abstraction

Step 1. Initializing the Devices

Initializing the m3g Device

The m3g initialization procedure includes starting the m3g library; opening the board, control, audio and video devices; and initializing the default application parameters and capabilities in each case (see Figure 7).

1. Before any other m3g function is called, an application makes a call to the m3g_Start() function to initialize the 3G-324M Library.
2. An application opens the board device and initializes the default system level parameters for the 3G-324M endpoints in the system. The board device can also be used to configure event reporting for optional events.
3. The m3g_OpenEx() function is called with the board device string m3g_OpenEx("m3gB1",...,EV_ASYNC). The m3g_SetParm() function and m3g_EnableEvents() function can be called with the board device handle to set system parameters and enable events, respectively.
4. Once the board system-level parameters are set, each 3G-324M channel control, audio, and video device should be opened and configured. The 3G-324M API device initialization is a process that requires the user to query the default Dialogic® system capabilities with the Get Local Capabilities method (that is, m3g_GetLocalCaps() function) and to set the application's preferred capabilities with the Set Terminal Capabilities method (that is, m3g_SetTSC() function).

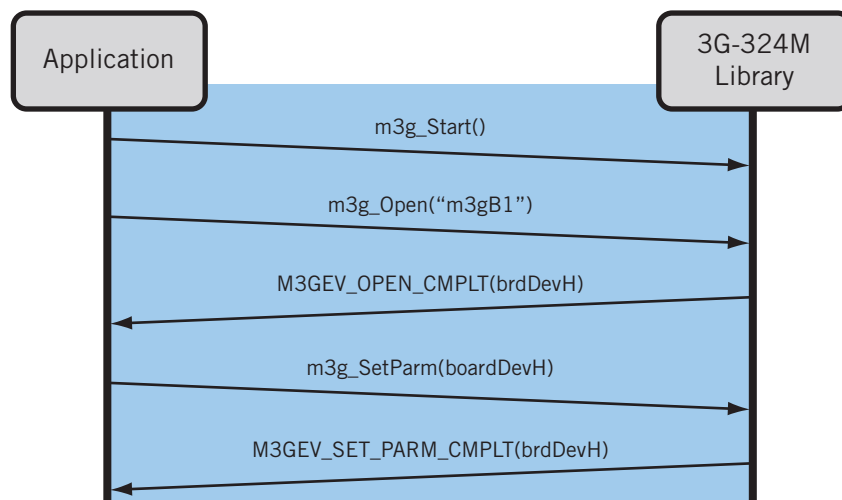


Figure 7. m3g Device Initialization

Run the following initialization procedure for each 3G-324M channel (see Figure 8). After this procedure, the system will be ready to negotiate a 3G session that will occur in *Step 4, Establishing a 3G-324M Session*.

For every 3G-324M channel being used by the application, the application will:

- **Open the m3g control device**
 - o The application calls the `m3g_OpenEx("m3gB1Tn", ..., ASYNC)` function using the m3g control device string to configure the H.223 multiplexer capabilities.
- **Reset the channel**
 - o It is good practice to call the `m3g_Reset()` function with the m3g control handle to reset the 3G-324M channel to an initialized state at the beginning of the application.
 - o The `m3g_Reset()` function is used so that the control, audio, and video devices that were not closed properly are returned to an initial state.
- **Get the system default H.223 capabilities**
 - o The application calls the `m3g_GetLocalCaps()` function with the m3g control handle to retrieve the default H.223 capabilities of the system.
 - o The default H.223 system capabilities are returned in the termination event `M3G_GET_LOCAL_CAPS_CMPLT`.
- **Open the m3g audio device**
 - o The application calls the `m3g_OpenEx("m3gB1Tn:AUDIO1", ..., ASYNC)` function using the m3g audio device string to configure the audio stream to/from the multiplexer/demultiplexer.
- **Get the system default audio capabilities**
 - o The application calls the `m3g_GetLocalCaps()` function with the m3g audio handle to retrieve the default audio capabilities of the system.
 - o The audio CODEC types supported by the system (for example, AMR, G723), as well as other audio capabilities, are returned in the termination event `M3G_GET_LOCAL_CAPS_CMPLT`.
- **Open the m3g video device**
 - o The application calls the `m3g_OpenEx("m3gB1Tn:VIDEO1", ..., ASYNC)` function using the m3g video device string to configure the video stream to/from the multiplexer.
- **Get the system default video capabilities**
 - o The application calls the `m3g_GetLocalCaps()` function with the m3g video handle to retrieve the default video capabilities of the system.
 - o The video CODEC types supported by the system (for example, H263, MPEG4), as well as other video capabilities, are returned in the termination event `M3G_GET_LOCAL_CAPS_CMPLT`.
- **Set the application-preferred capabilities**
 - o The application must cache the H.223, audio, and video capability structures returned from the `m3g_GetLocalCaps()` function and use these structures to make modifications to the default system capabilities.
 - o The application calls the `m3g_SetTCS()` function with the m3g control handle to set the local Terminal Capabilities Set for H.223, audio, and video. The local TCS can be the entire list of capabilities as received by the `m3g_GetLocalCaps()` function or modified values based on the application specification. For interoperability reasons, it is the generally preferred option that the H.223 and media capabilities specified should be left unchanged from those values returned via the `m3g_GetLocalCaps()` function. Unless there is a compelling reason, changing the default capability settings may increase risk of failures in subsequent logical channel establishment.
 - o After the `m3g_SetTCS()` function is called, the system is now ready to exchange preferred capabilities with a remote 3G-324M endpoint during the capabilities exchange of the 3G session.

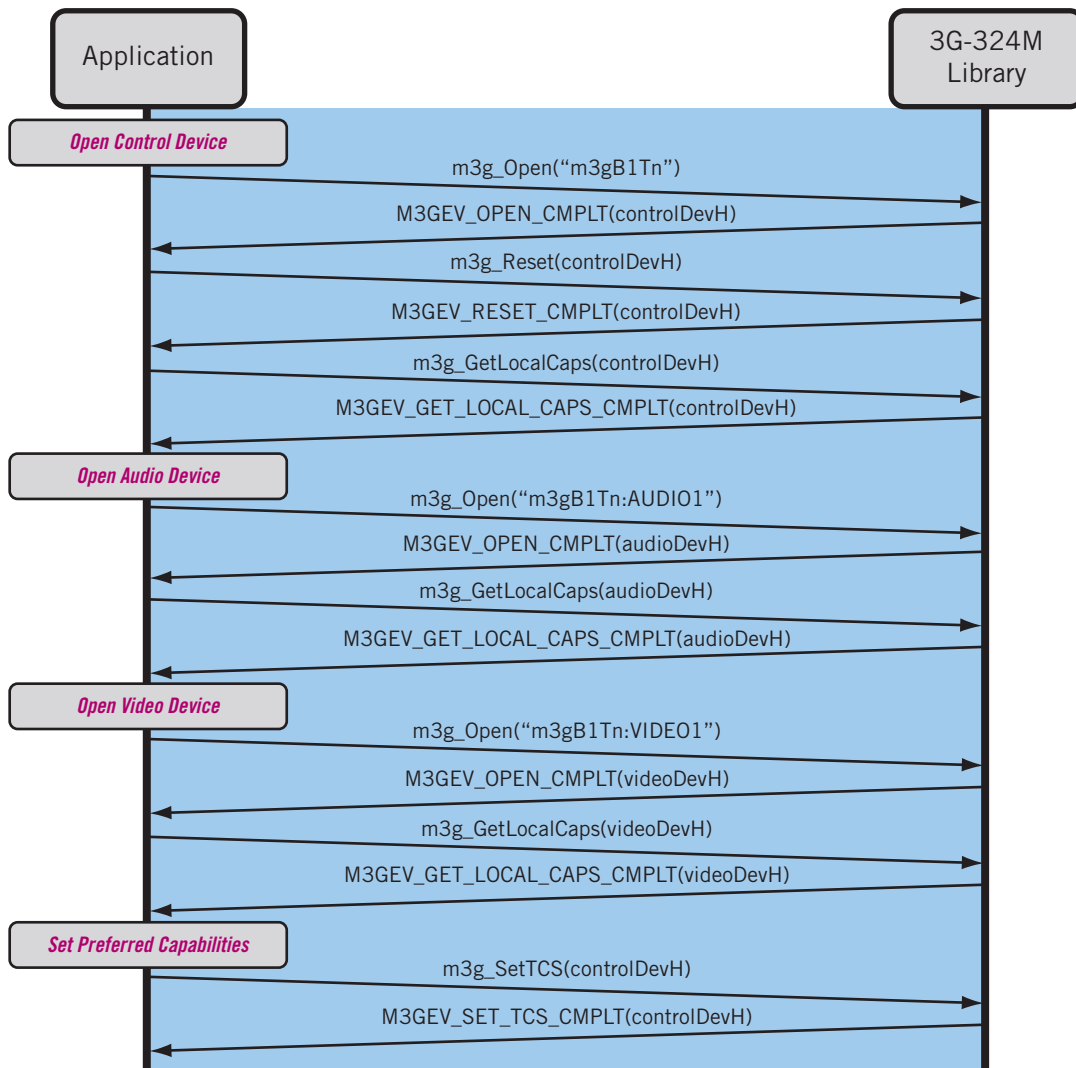


Figure 8. Initialization Procedure for Each 3G-324M Channel

Enabling MONA Events

Prior to establishing the 3G-324M Session with MONA, the application should enable unsolicited MONA events with `m3g_EnableEvents`. The MONA unsolicited events are `M3GEV_SEND_MONA_PREF_MSG` and `M3GEV_MONA_PREF_MSG_RCVD`. These events will indicate to the application that the MONA preference message was successfully sent or received as part of the MONA MPC procedure.

Initializing the mm Device

The multimedia termination for the audio and video is provided by the mm device. Before the 3G multimedia connections can be made between the m3g device and the mm device, the mm device must be initialized. To initialize the mm device, an application must open the mm device using the `mm_Open()` function and reset the devices using the `mm_Reset()` function (see Figure 9).

- **Open the mm device**
 - o The application calls the `mm_Open("mmBxCy",...)` function using the mm device string to get a unique handle to the device. The application must wait for an `MMEV_OPEN` event indicating the handle is valid.
- **Reset the channel**
 - o It is good practice to call the `mm_Reset()` function with the mm handle to reset the multimedia device to an idle state at the beginning of the application.

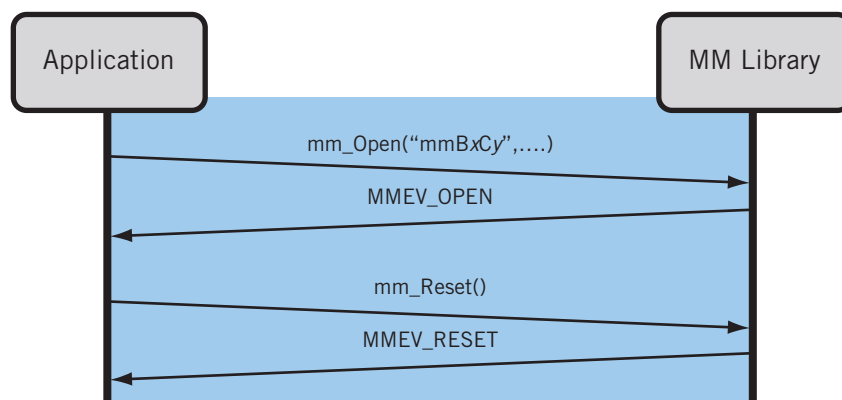


Figure 9. mm Device Initialization

Step 2. Connecting the Devices

Sometime before multiplexing starts and media is streaming in a 3G session, the m3g device is to be connected on one side to the network aggregate data and on the media side to audio and video media terminations. The m3g device does not terminate the audio and video streams, but performs the multiplexing/demultiplexing and provides an audio and video connection to other Dialogic® software components, such as the mm device. Relative to the m3g device, three full-duplex device connections (network, audio, and video) are to be made for achieving multimedia multiplexing and demultiplexing capability. The API used to connect devices is the Dialogic® Device Management API Library.

An application decides when the connection and disconnection between devices is made. Devices can be connected statically before call establishment or dynamically once the multimedia services are required. For the purpose of this document and to simplify the steps required, devices are connected after the *Step 1, Initializing the Devices* procedure, and remain connected throughout the 3G call. It is up to the application developers to handle dynamic allocation of device resources and the appropriate events to suit their application resource allocation.

Connecting Networks

On the network side, the m3g device is connected to an aggregate streaming line device. This is the multiplexer connection to the aggregate bitstream data provided by the bearer channel. The network connection can be PSTN (Release 99) or IP (Release 4).

Connecting PSTN Network

In a Release 99 PSTN Network, the m3g device is connected to a Dialogic® Global Call API Network dti device. The dti device provides the PSTN connection and represents the 64 kbps transparent clear channel or “unrestricted data” DS0 timeslot connection to the PSTN network. The dti device is put into non-companded, transparent mode so that the timeslot carries the modified bitstream (see Figure 10).

- An application calls the `dev_Connect()` function to connect the m3g device to the dti device.

Note: On some Dialogic® products, the transparent mode is enabled by calling the `gc_SetConfigData()` function and setting the `CCPARAM_TRANSPARENTMODE` parameter to `CCDM3FW_TRANSPARENTMODE_ENABLE`. For more information, refer to the product-specific release guide.

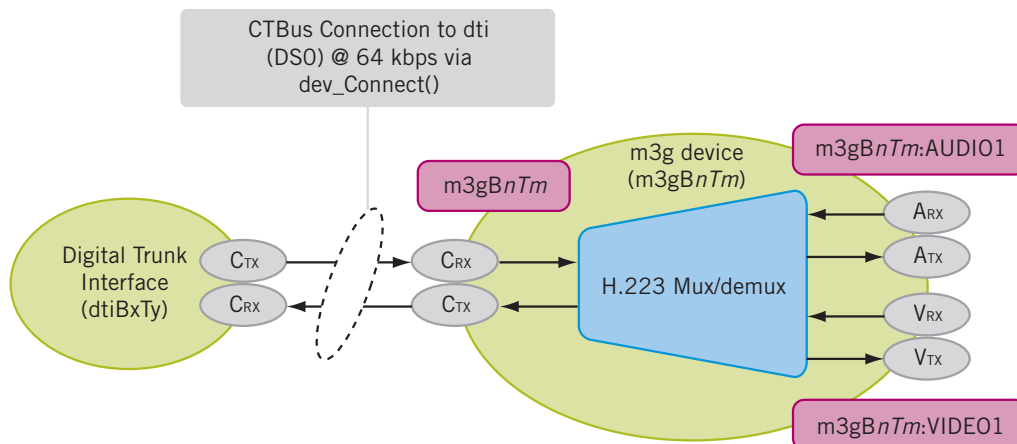


Figure 10. PSTN Network Connection

Connecting IP Network

In a Release 4 IP Network, the m3g device is connected to a Dialogic® IPMedia Streaming device using the Nb UP protocol extensions (see Figure 11). The IPMedia Streaming device, known as an “ipm device,” is used to control the IP media session over RTP. If used as a transport for 3G-324M bitstream, the ipm device uses a special Nb UP protocol along with special initialization messages to set up the RTP session. The resulting IP/RTP packets contain an Nb UP packet header and a payload carrying the aggregate bitstream.

- An application uses the `dev_PortConnect()` function to connect the m3g device to the ipm device. The `dev_PortConnect()` function use is described in the following *Connecting Audio and Video Media Ports* section.

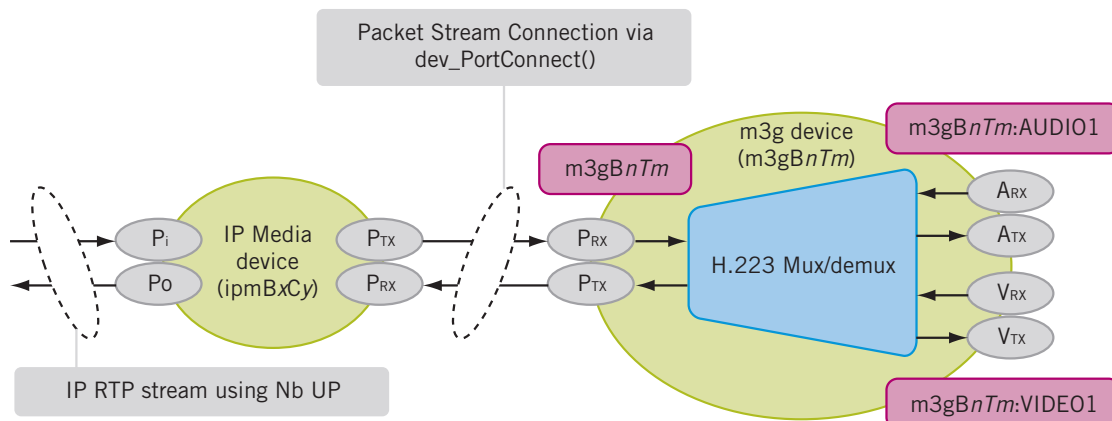


Figure 11. IP Network Connection

Connecting Audio and Video Media Ports

For a 3G multimedia session, media termination connections are made to an mm device using the Device Management API Library. The procedure to make port connections between devices involves retrieving device Tx media ports for audio and video, retrieving device Rx media ports for audio and video, and connecting appropriate Tx-Rx pairs between devices using the dev_PortConnect() function.

Retrieving Device Media Ports

In order to retrieve the device Rx and Tx Ports for audio and video, an application calls the dev_GetTransmitPortInfo() function to get the audio and video Tx ports and the dev_GetReceivePortInfo() function to get the audio and video Rx ports. The following steps correspond to the steps in Figure 12 and detail what an application does to get Tx and Rx media ports for the m3g and mm devices:

- a. Call the dev_GetTransmitPortInfo() function with the m3g audio and video handles to get the audio and video Tx Ports for the m3g device. The DM_PORT_INFO_LIST of Tx ports for the m3g device are returned in the asynchronous DMEV_GET_TX_PORT_INFO return event.
- b. Call the dev_GetReceivePortInfo() function with the m3g audio and video handles to get the audio and video Rx Ports for the m3g device. The DM_PORT_INFO_LIST of Rx ports for the m3g device are returned in the asynchronous DMEV_GET_RX_PORT_INFO return event.
- c. Call the dev_GetTransmitPortInfo() function with the mm handle to get the audio and video Tx Ports for the mm device. The DM_PORT_INFO_LIST of Tx ports for the mm device are returned in the asynchronous DMEV_GET_TX_PORT_INFO return event.
- d. Call the dev_GetReceivePortInfo() function with the mm handle to get the audio and video Rx Ports for the mm device. The DM_PORT_INFO_LIST of Rx ports for the m3g device are returned in the asynchronous DMEV_GET_TX_PORT_INFO return event.
- e. An application must cache the audio and video port information for later use when connecting devices.

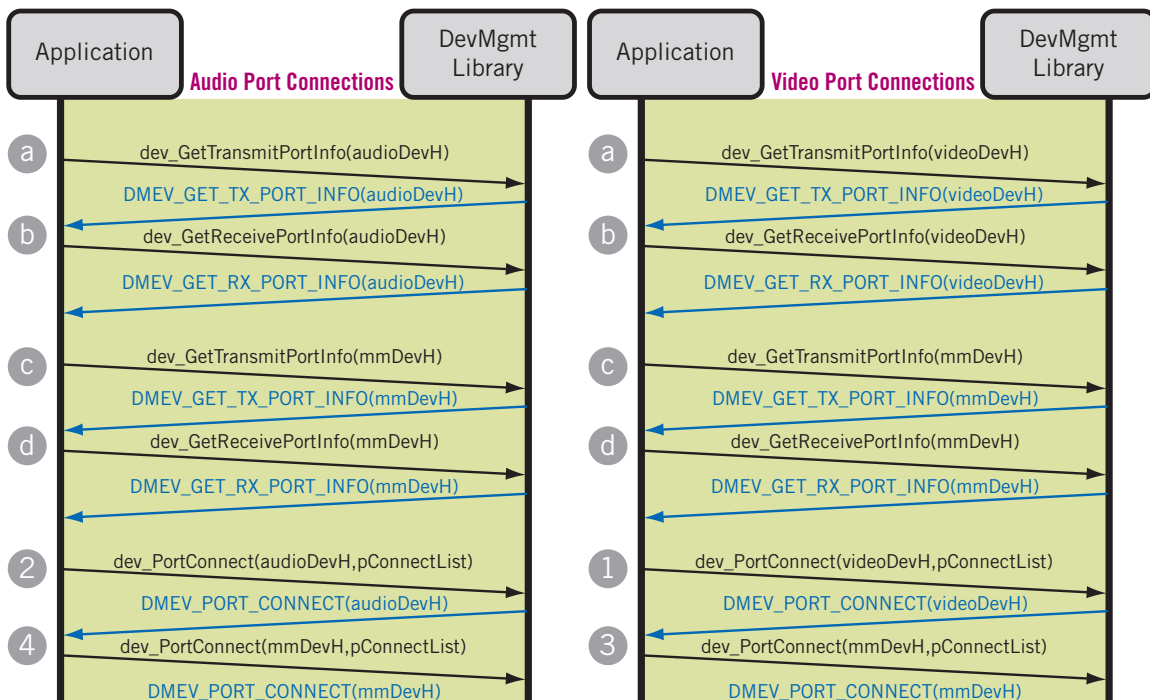


Figure 12. Retrieving Device Media Ports and Making Port Connections

Connecting Media Ports between Devices

In order to make full-duplex audio and video connections between the m3g device and the mm device, the dev_PortConnect() function must be called four times, once for each desired transmit media stream (see Figure 13). The dev_PortConnect() function creates a unidirectional packet connection between the transmit port of one device to the receive port of another device. In this case, two audio and two video transmit streams are created, one audio tx stream and one video tx stream for each device.

1. *Connect m3g device Tx Video Stream* – Call dev_PortConnect() function with the m3g Tx video port_info list and mm Rx video port_info list.
2. *Connect m3g device Tx Audio Stream* - Call dev_PortConnect() function with the m3g Tx audio port_info list and mm Rx audio port_info list.
3. *Connect mm device Tx Video Stream* - Call dev_PortConnect() function with the mm Tx video port_info list and m3g Rx video port_info list.
4. *Connect mm device Tx Audio Stream* - Call dev_PortConnect() function with the mm Tx audio port_info list and m3g Rx audio port_info list.
5. In each case above, enable transcoding or native (no transcoding) connection by specifying the unFlags field of the DM_PORT_CONNECT_INFO data structure.

Note: Tx and Rx directions above represent internal media stream direction relative to the device, not direction relative to the 3G network.

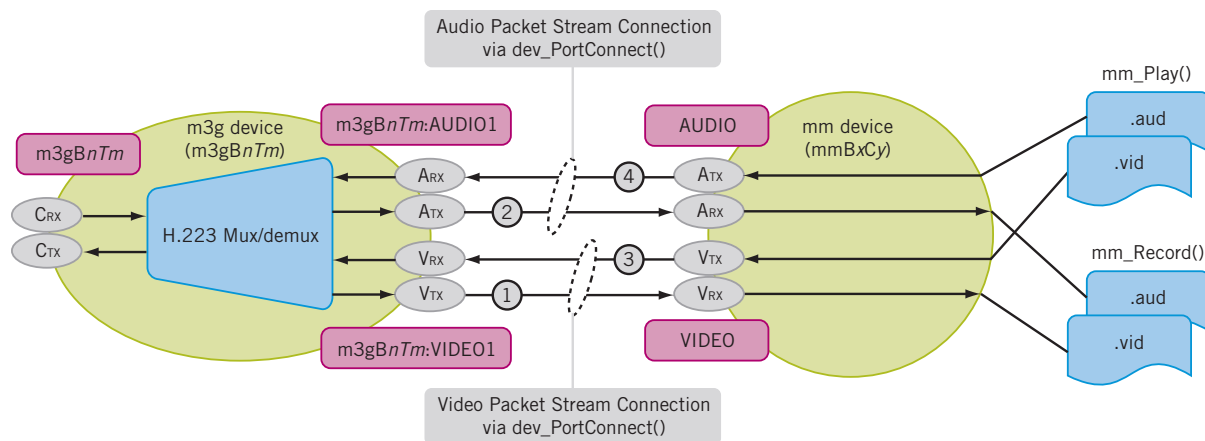


Figure 13. Connecting Media Ports between Devices

Transcoding versus “Native” Connection

An application can make connections to an m3g device in two modes, “native” (no transcoding) or with transcoding enabled. The native connection method is supported on all Dialogic® products. Native connections are where audio or video media is passed without modification from the 3G network to the multimedia file. This mode means that no audio or video transcoding is performed on the media from the 3G network. The media will be stored in a multimedia file using the 3G network CODECs. In most cases, the video CODEC will be stored as H.263 or MPEG4 for video and as AMR for audio. An application may want to employ this mode

to save on MIPS and increase density. A native unidirectional media stream connection can be made by calling `dev_PortConnect()` with the `unFlags=DMFL_TRANSCODE_NATIVE` in the `DM_PORT_CONNECT_INFO` structure.

Transcoding can be enabled to keep multimedia files in a singular format. Audio transcoding allows the network coder such as AMR to be stored as PCM data, for example. Video transcoding is a CPU-intensive process that provides translation between video coder types, such as H.263 and MPEG4, resolution types, bitrates, and framerates. Enabling video transcoding will also provide greater control over video I-Frame generation and stream manipulation, such as text overlay. The greater control over video transcoding comes at the expense of CPU and reduced density.

In products that support transcoding connections, such as Dialogic® Multimedia Platform for ATCA 2.0 (MMP 2.0) and Dialogic® Multimedia Kit for PCIe, the Dialogic® system software automatically takes care of translation between media formats when transcoding is enabled. An application can enable transcoding on a unidirectional media stream by calling `dev_PortConnect()` with the `unFlags=DMFL_TRANSCODE_ON` in the `DM_PORT_CONNECT_INFO` structure. This means that the `dev_PortConnect()` with `TRANSCODE_ON` will handle transcoding between the two media coder formats specified in each device's data structures. For transport over 3G, the media coder settings are specified for the m3g device in the `m3g_SetTCS()` function and the `m3g_OpenLC()` function data structure initialization. For multimedia Play/Record, the media coder settings are specified for the mm device in the `mm_Play()` function and `mm_Record()` function data structure initialization. Transcoding is then performed between devices by translating the media stream definitions at the m3g device to and from the media definitions at the mm device.

Step 3. Starting a 3G Call — Establishing the Bearer Channel

After *Step 1, Initializing Devices* and *Step 2, Connecting Devices*, the 3G application is ready to receive a 3G call from the network or make a 3G call to the network and establish a bearer channel. Across the bearer channel, the 3G-324M peer-to-peer protocol will establish a 3G-324M session.

Within the Release 99 PSTN network, the bearer channel is normally established using ISUP or ISDN protocol. The bearer

channel in the Release 99 PSTN network is a 64 kbps clear channel, “unrestricted data” DS0 timeslot.

Within the Release 4 IP network, it is expected that the call will be established using BICC. The bearer channel in the Release 4 IP network is an Nb User Plane (Nb UP) RTP connection.

Step 4. Establishing a 3G-324M Session

Once a call is connected within the network and a bearer channel is established, the 3G-324M peer-to-peer session protocol is started with the remote 3G-324M endpoint. The role of the 3G-324M API begins after the application receives indication that the bearer channel is established. It is at this time that a 3G-324M session can be established through 3G-324M protocol exchange.

The m3g device is the Dialogic® software component that handles the 3G-324M multiplexing protocol. An application walks through establishing a 3G session by waiting on protocol events from the Dialogic® Mux3G firmware and applying the proper methods to open media channels for media to start streaming.

An application can be written to enable MONA procedures during session establishment with any remote 3G-324M endpoint. The MONA procedures are intended not to interfere with 3G-324M endpoints that do not support MONA. The real advantage comes when connecting to a MONA capable endpoint where all media channels are set up using the MCP technique, significantly reducing time to establish media streaming. Fallback establishment procedures assure that media can be established even if MONA procedures do not succeed.

Establishing a 3G-324M Session using MONA

The MONA feature in the Dialogic 3G-324M API software can be selectively controlled per call. If MONA is disabled, the call proceeds using the standard logical channel establishment procedures (see “Establishing Standard H.245 Open Logical Channels”). To reduce setup time, the application can use this option to enable the MONA procedure as described and fall back to the standard H.245 OLC for media channels that are not established as MPCs. The MONA procedure will also attempt to use the ACP procedure to establish the media channels that were not started as MPCs.

While at the protocol level, the ACP procedure does not require waiting for message acknowledgements; from the application point of view, the ACP procedure and standard H.245 OLC establishment procedures are handled exactly the same. Underlying behavioral differences between ACP and standard H.245 OLC procedures are abstracted from the application allowing for common API behaviors, which require exchanging TCS, MSD, and OLC messages.

The following steps are performed if establishing a 3G-324M session with MONA (see Figure 14):

1. Start H.245 – MONA Preference Message Exchange — To start the 3G-324M session with MONA procedures, the application calls the `m3g_StartH245()` function with MONA enabled in the `M3G_H223_SESSION` structure. Prior to H.223 mux level framing, a MONA preference message is exchanged within the frame header to identify the local Tx and Rx Media Preconfigured Channels (MPCs) supported by each endpoint. The application should expect the unsolicited events `M3GEV_SEND_MONA_PREF_MSG` and `M3GEV_MONA_PREF_MSG_RCVD` to indicate that the MONA preference messages have been exchanged by the local and remote endpoints.

Note: In the MONA procedures, the preference message MPC-TX and MPC-RX bits are automatically set based on the capabilities specified in the `m3g_SetTCS()` function. If the specified terminal capabilities match the format of the MPCs defined in the MONA standard, the respective MPC-TX bits are automatically set. However, only one MPC-RX video bit may be set for a given call, with the priority given to H.263. Thus, if the default capabilities are unchanged in the call to the `m3g_SetTCS()` function, the MPC-TX bits for AMR, H.263 and MPEG4 are enabled, and the MPC-RX bits for AMR and H.263 are enabled.

2. Establish MPCs — Up to four Media Preconfigured Channels (MPCs) can be established based on the MONA preference message: audio Tx, audio Rx, video Tx, and video Rx. After exchange of the MONA preference messages, events will be returned to signal that the MPCs have been established and that media streaming can be started. The events returned are `M3GEV_TX_MPC_ESTABLISHED` and `M3GEV_RX_MPC_ESTABLISHED`. Each of these events can be returned twice, once for audio and once for video. It should be noted that the application may not receive

all four MPC-established events if the MPC negotiation is unsuccessful for any media channel direction. Whether the 3G endpoint does not support MONA or MONA preferences are incompatible, this is a normal situation and the fallback is for the media channels to be opened using the standard H.245 Open Logical Channel procedure (see “Establishing Standard H.245 Open Logical Channels”).

3. Establish Media for MPC with `m3g_ModifyMedia()` function

— Once the protocol establishes the MPCs, each media channel direction that was established can be started with the `m3g_ModifyMedia()` function. In *Step 5, Exchanging Media*, see “ModifyMedia Approach” for details on how the `m3g_ModifyMedia()` function is used to start media streaming.

4. Proceed with MONA ACP and Standard H.245 Logical Channel Establishment

— Once the application receives the `M3GEV_FRAMING_ESTABLISHED`, `M3GEV_MSD_ESTABLISHED`, `M3GEV_REMOTE_TCS_RCVD`, and `M3GEV_LOCAL_TCS_ACKD` events, the application will not receive further `MPC_ESTABLISHED` events. Media channels that were not established as MPCs must be established by retrieving matched capabilities and using the standard H.245 open logical channel procedure for forward or reverse logical channel establishment. As part of the MONA procedures, when MONA is enabled using the `m3g_StartH245()` function, the MONA Accelerated Connection Procedure (ACP) will be tried as part of the remaining protocol exchange. From the application point of view, the procedure to establish logical channels using ACP is the same as the standard H.245 OLC procedure described in “Establishing Standard H.245 Open Logical Channel”.

Establishing Standard H.245 Open Logical Channels

The standard H.245 open logical channel establishment procedure is the complete protocol method to creating logical channels used to establish media streaming. An application must be written to follow the standard open logical channel establishment whenever a media channel is not established as an MPC. If MONA is enabled using the `m3g_StartH245()` function, any media channels are not established using the MPC procedure will be established using ACP in the following steps. If MONA is not enabled during the `m3g_StartH245()` function, then media channels will be established as logical channels using the following steps.

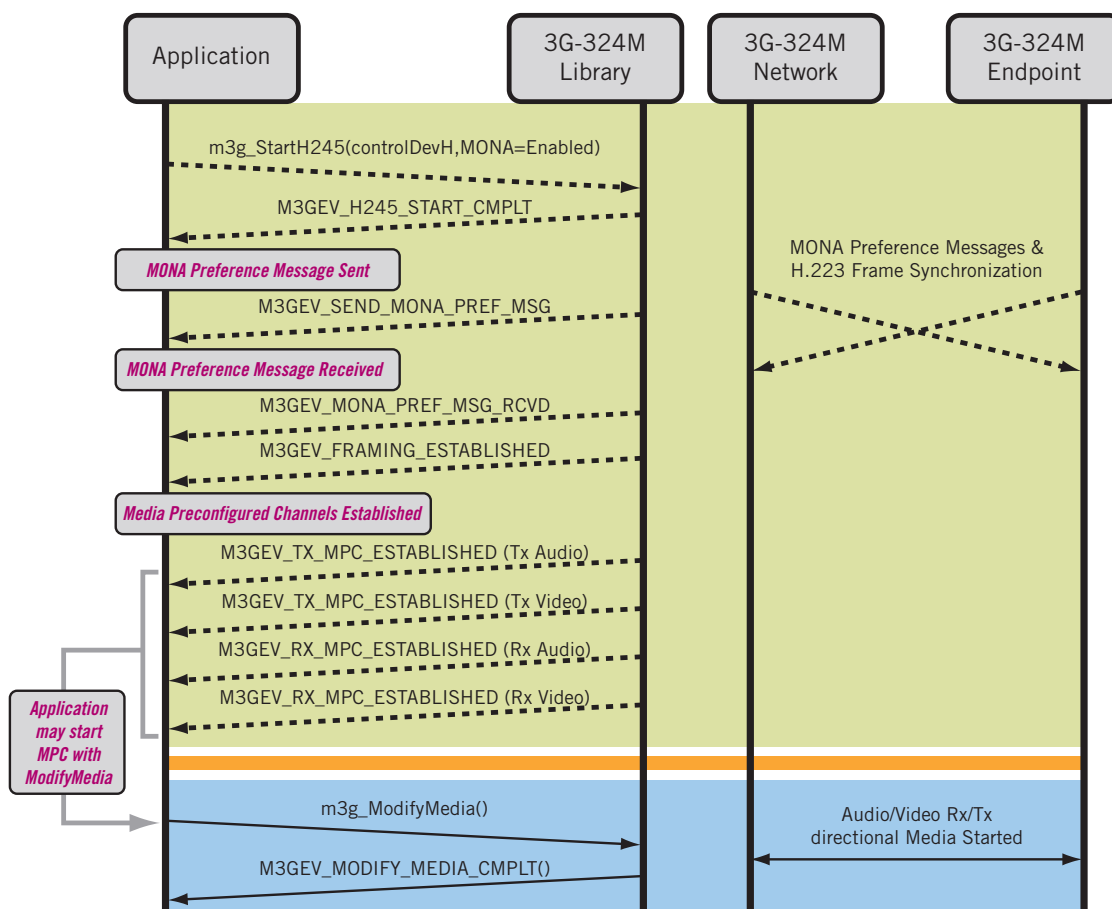


Figure 14. Establishing MONA MPCs

The following steps are performed if establishing a 3G-324M session:

1. Start H.245 — Physical Layer Frame Synchronization

- After MONA Preference Message Exchange, the application calls the `m3g_StartH245()` function to initialize the H.245 logical channel 0 control session. The Mux3G firmware starts the 3G-324M protocol by first synchronizing to the H.223 mux level framing. This training phase at the H.223 level is used to determine the error protection capabilities and multiplexing level using the H.223 capabilities structure provided by the application in the `m3g_SetTCS()` function. Once the Mux3G firmware establishes framing, the Physical Layer is established and a `M3GEV_FRAMING_ESTABLISHED` event is returned. The Mux3G firmware then immediately opens logical channel 0 for H.245 messaging. The initial H.245 exchange, including Master Slave Determination and Terminal Capabilities Exchange, is negotiated with values provided in the Device Initialization steps.
- **Terminal Capabilities Set (TCS) Exchange** — The Mux3G firmware automatically handles the Terminal Capabilities exchange with the remote 3G-324M endpoint. The Terminal Capabilities Exchange transactions are exchanged with the remote 3G-324M endpoint based on the settings provided by the application in the `m3g_SetTCS()` function. Local TCS acknowledgement and receipt of remote TCS events will be posted by the Mux3G firmware. The application is notified

that the local TCS was acknowledged with an M3GEV_LOCAL_TCS_ACKD event and that the remote TCS was received with a M3GEV_REMOTE_TCS_RCVD event. The receipt of these two messages signals that the Terminal Capabilities Set exchange was successful.

- **Master Slave Determination (MSD)** — The Mux3G firmware automatically handles the Master Slave Determination exchange with the remote 3G-324M endpoint. The Master Slave Determination transactions are exchanged with the remote 3G-324M endpoint using the application specified terminal type in the M3G_E_PRM_H245_TERMINAL_TYPE parameter setting. The application is notified of either designation as master or slave with a M3GEV_MSD_ESTABLISHED return event when MSD negotiation is complete.
- An application must wait on the low-level 3G-324M protocol exchange to complete before moving forward. The application can proceed to open audio and video logical channels only after it receives the following four events:
 1. M3GEVT_FRAMING_ESTABLISHED
 2. M3GEV_MSD_ESTABLISHED
 3. M3GEV_REMOTE_TCS_RCVD
 4. M3GEV_LOCAL_TCS_ACKD
- M3GEV_FRAMING_LOST — The M3GEV_FRAMING_LOST event will be returned if:
 - The frame synchronization pattern is not found in the bitstream. This condition can occur if the frame synchronization pattern is not detected within five seconds of calling m3g_StartH.245() function after call connection.
 - The call is lost or the remote endpoint drops the 3G-324M session without terminating logical channels.

If the M3GEV_FRAMING_LOST event is returned within five seconds of calling m3g_StartH.245(), this may suggest that the data on the aggregate is not proper 3G-324M encoded data, the m3g_StartH.245() function was called too early or too late and has timed out, or the data is not present in the bearer channel because the specified timeslot is incorrect.

2. Get Matched Capabilities

- Prior to opening logical channels, the application calls the m3g_GetMatchedCaps() function to query the system to determine the matching capabilities between the local and remote 3G-324M endpoints. The m3g_GetMatchedCaps() function is provided as a convenience to the application to determine the matched capabilities and priority of capabilities between the remote and local 3G-324M endpoints (see Figure 15). The capabilities are listed in decreasing order of preference by the endpoint deemed the master in the H.245 Master Slave Determination.
- An application can choose to use the matched capabilities provided, or to modify these capabilities to select which capabilities it will use to open the forward logical channels for audio and video.

Note: Any media or multiplexer format chosen by the local endpoint must be within the tolerances of media formats supported by the remote endpoint as communicated in the matched capability list. For example, the local transmitter should not open a video logical channel with a video maxBitRate value exceeding the bit rate value that the remote video receiver indicated it can receive for video. To improve interoperability among third party 3G vendors, the audio and video CODECs chosen should be those most preferred by the master. These CODECs are identified as being the first transmit audio and video CODECs within the matched capability list.

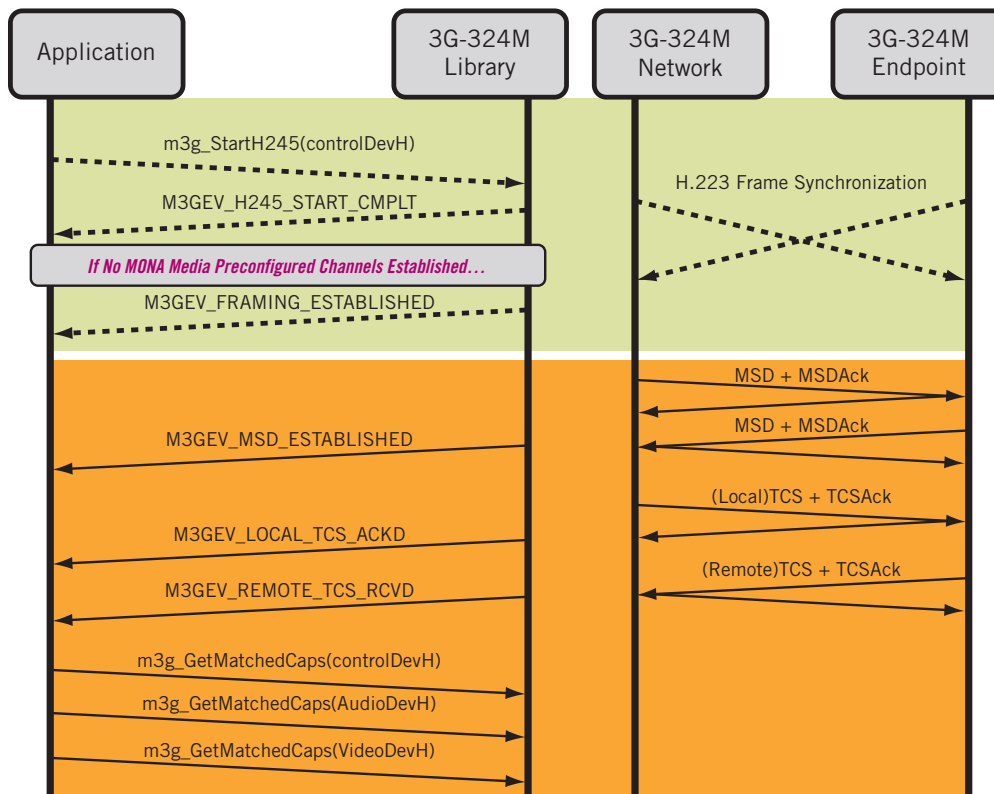


Figure 15. Start H.245 and Get Matched Capabilities

3. Creating Audio/Video Logical Channel

- After the low-level protocol has succeeded and an application determines the matched capabilities between 3G-324M endpoints, the application can begin opening the forward logical channels for audio and video. The application will also acknowledge or reject the reverse open logical channel requests from the remote endpoint.
- Forward Logical Channels** — Open Logical Channel (OLC) requests are sent from local to the remote 3G-324M endpoint to specify the H.223 multiplexed media channel, adaptation layer format and interleaving format for the Tx direction media stream. The remote endpoint acknowledges the OLC with an OLCAck. By acknowledging the OLC request, the remote endpoint verifies that the OLC request conforms to the negotiated TCS and that it will configure its multiplexer to receive the media channel with the specified parameters.
 - *Audio Forward Logical Channel*— The `m3g_OpenLC()` function is called with the control handle and `capabilityType=M3G_E_AUDIO_CAPABILITY` to open the audio forward logical channel. The OLC is used to request the specific audio media stream, in the TX direction that will be open from the local to the remote endpoint. The audio capability, including audio CODEC type of the channel, is specified in the `pMediaCapability` field, and the adaptation layer format of the audio channel is specified by the `pH223LCParameter` field. The audio logical channel number will be returned in the `OpenLogicalChannelAck` received from the remote 3G endpoint, indicated to the application by `M3GEV_OPEN_LC_CMPLT` event.

- *Video Forward Logical Channel*— The `m3g_OpenLC()` function is called with the control handle and `capabilityType=M3G_E_VIDEO_CAPABILITY` to open the video forward logical channel. The OLC is used to request the specific video media stream, in the TX direction that will be open from the local to the remote endpoint. The video capability, including video CODEC type of the channel, is specified in the `pMediaCapability` field, and the adaptation layer format of the video channel is specified by the `pH223LCParameter` field. The video logical channel number will be returned in the `OpenLogicalChannelAck` received from the remote 3G endpoint, indicated to the application by `M3GEV_OPEN_LC_CMPLT` event.
- **Reverse Logical Channels** — The Open Logical Channel (OLC) requests from a remote 3G-324M endpoint specify the media channels streaming from remote to local 3G-324M endpoint. A Reverse Logical channel is acknowledged or rejected by the application.
- **Audio and Video Reverse Logical Channels** — The remote 3G-324M endpoint will asynchronously request to open the reverse logical channels, specifying the capabilities and adaptation layer to use for the media stream. The OLC from the remote endpoint is used to request the specific media stream that will be open from the remote to the local endpoint. The reverse media logical channel is the RX direction relative to the local application. The application will get an indication from the Mux3G firmware that an OLC has been received that matches negotiated capabilities. The `M3GEV_REMOTE_OLC_RCVD` event indicates to the application that the open logical channel was received from the remote endpoint. The `m3g_RespondToOLC()` function is used to send the Acknowledgement or Rejection back to the remote 3G-324M endpoint (see Figure 16).
- Once the Logical Channel has been successfully opened, multimedia streaming can begin in that direction.

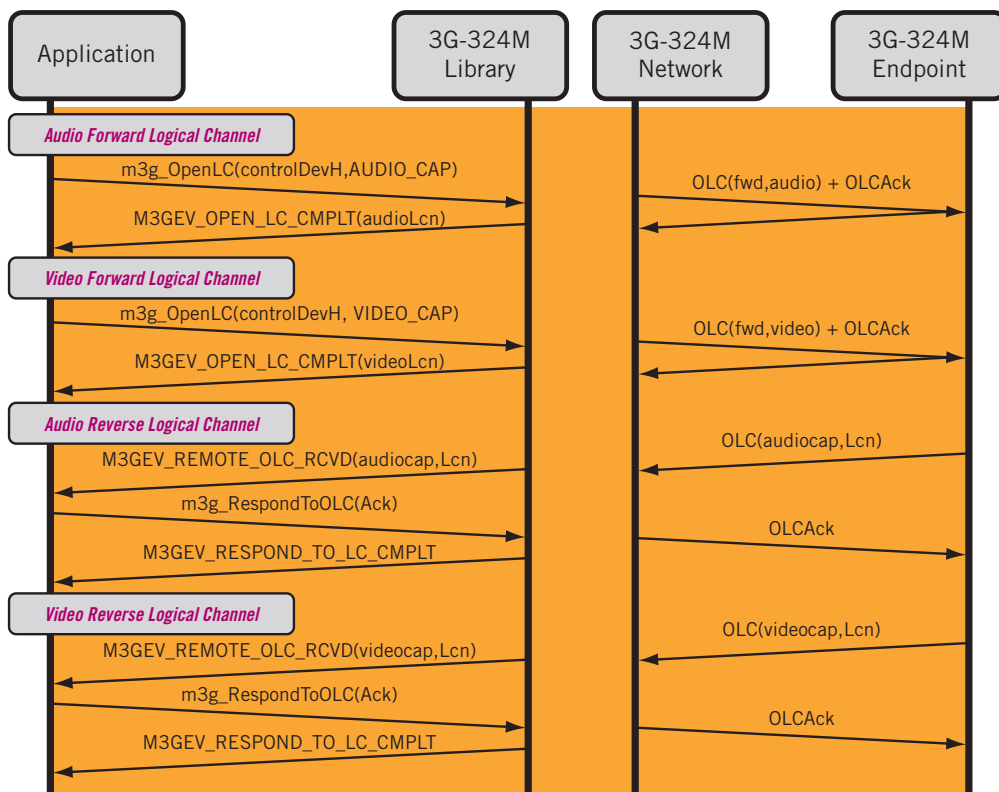


Figure 16. Audio/Video Logical Channel Creation

Step 5. Exchanging Media

Once the 3G-324M Session logical channel is open, media can flow in that direction. The Dialogic® 3G-324M API provides two approaches to starting media streams between the multimedia device and the H.223 multiplexer/demultiplexer:

- **StartMedia Approach** —The first method uses the `m3g_StartMedia()` function to start a full duplex media stream once both the forward and reverse logical channels are open between the 3G-324M local and remote endpoints.
- **ModifyMedia Approach** — The second method uses the `m3g_ModifyMedia()` function to start a unidirectional media stream. The `m3g_ModifyMedia()` function is used to start a media stream that corresponds to the direction of the individual forward or reverse logical channel that was successfully opened.

To simplify the start of bi-directional media streaming to a remote 3G endpoint, an application can take advantage of the StartMedia approach. Conversely, faster media start can be achieved if unidirectional media streams are started using the ModifyMedia approach after successfully establishing the Logical Channel.

StartMedia Approach

The `m3g_StartMedia()` function begins the multiplexing/demultiplexing on the H.223 data to enable media streaming (see Figure 17). The application can begin audio and video streaming by calling the `m3g_StartMedia()` function once both the forward and reverse logical channels are open for the media type. The `m3g_StartMedia()` function is called using the m3g audio handle to start audio streaming and using the m3g video handle to start video streaming between the H.223 multiplexer and the multimedia device. The application performs the following actions to start full-duplex media using the StartMedia approach:

- **Start full-duplex audio streaming to/from H.223 multiplexer** — Call the `m3g_StartMedia()` function with the m3g audio handle. The system returns the `M3GEV_START_MEDIA_CMPLT(audio)` event when streaming has been successfully initiated.
- **Start full-duplex video streaming to/from H.223 multiplexer** — Call the `m3g_StartMedia()` function with the m3g video handle. The system returns the `M3GEV_START_MEDIA_CMPLT(video)` event when streaming has been successfully initiated.

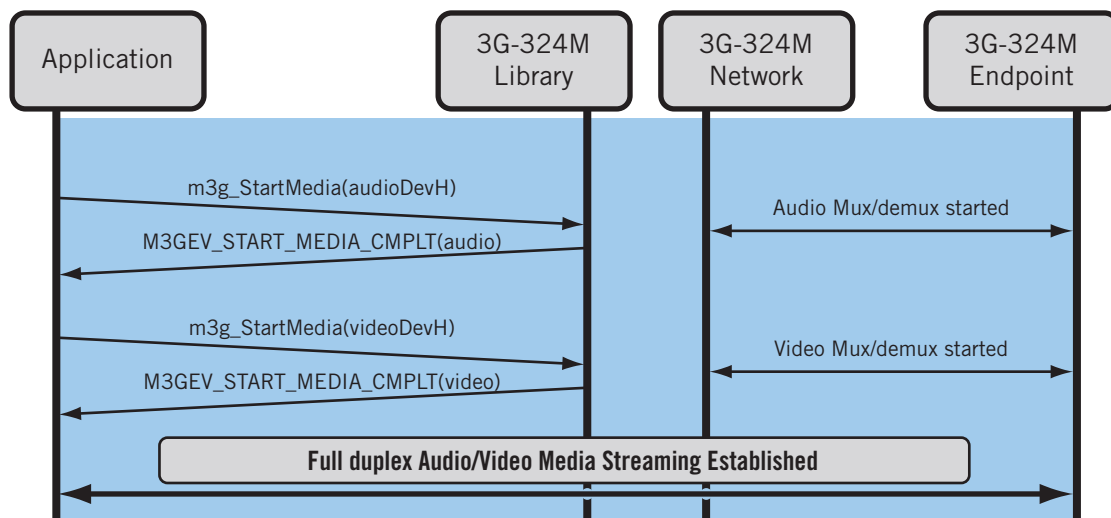


Figure 17. Start Media Approach

ModifyMedia Approach

The `m3g_ModifyMedia()` function begins the multiplexing/demultiplexing on the H.223 data to enable media streaming in a half-duplex direction (see Figure 18). The application can begin the half-duplex audio or video streaming immediately after the corresponding forward logical channel or reverse logical channel is open by calling the `m3g_ModifyMedia()` function.

The `m3g_ModifyMedia()` function is called using the `m3g` audio handle and a direction parameter to start audio streaming and using the `m3g` video handle and a directional parameter to start video streaming between the H.223 multiplexer and the multimedia device. The direction parameter allows the application to specify transmit only (`M3G_E_TX`), receive only (`M3G_E_RX`), transmit and receive (`M3G_E_RXTX`), or complete stop (`M3G_E_IDLE`). The direction specified in the `m3g_ModifyMedia()` function is the direction relative to the 3G network. This is opposite to the direction specified during port connections, which was relative to the media termination device.

As an example, the following steps could be performed to start full duplex audio and video media by calling the `m3g_ModifyMedia()` function to start unidirectional streaming corresponding to the OLC direction, once the logical channel is open. Note that the sequence in which logical channels may be opened is non-deterministic. Each audio and video logical channel is opened in parallel by both endpoints in any order. The following example provides one potential sequence of channels being opened:

- **Start forward logical channel audio Tx** — The application will call the `m3g_ModifyMedia()` function with the `m3g` audio handle and the direction `M3G_E_TX`. The system returns the `M3GEV_MODIFY_MEDIA_CMPLT` event when streaming has been successfully initiated.
- **Start reverse logical channel audio Rx** — The application will call the `m3g_ModifyMedia()` function with the `m3g` audio handle and the direction `M3G_E_RXTX` to enable full duplex audio. The system returns the `M3GEV_MODIFY_MEDIA_CMPLT` event when streaming has been successfully initiated.
- **Start reverse logical channel video Rx** — The application will call the `m3g_ModifyMedia()` function with the `m3g` video handle and the direction `M3G_E_RX`. The system returns the `M3GEV_MODIFY_MEDIA_CMPLT` event when streaming has been successfully initiated.
- **Start forward logical channel video Tx** — The application will call the `m3g_ModifyMedia()` function with the `m3g` video handle and the direction `M3G_E_RXTX` to enable full duplex video. The system returns the `M3GEV_MODIFY_MEDIA_CMPLT` event when streaming has been successfully initiated.

Note: Tx and Rx directions above represent the media stream direction relative to the 3G network.

Starting Multimedia Play/Record

After the media streams are started between the H.223 multiplexer/demultiplexer and the multimedia device, the `mm` device can be used to play or record multimedia files. The `mm` device must set up the play list with the proper audio and video files. The method of connection between devices, “native” or transcoding, dictates the media format that can be used to store media files.

On systems that are set up to provide Native connections, the multimedia files must be stored with AMR or G723 for audio and H263 QCIF for video with a video bit rate less than 40 kbps. If MPEG4 is supported by the Dialogic® software, the video can be stored as Native MPEG4 data as well.

On systems that support transcoding, the files can be stored in any supported format and the media will be transcoded to the proper format. If transcoding is enabled, the media characteristics for the 3G network are specified in the `m3g` device the `m3g_OpenLC()` function initialization structures and the media characteristics of the multimedia storage format are specified in the `mm_Play()` function or `mm_Record()` function data structure initialization.

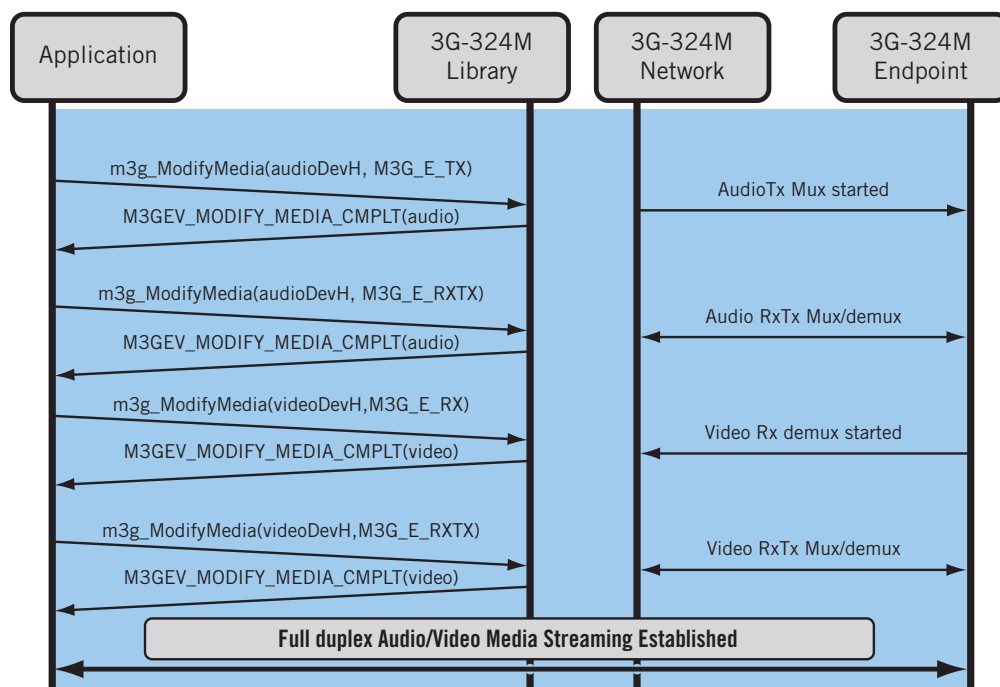


Figure 18. Modify Media Approach

H.245 UII Digit Detection — For mobile telephony applications, user input communication is often made through a DTMF digit sent from the remote 3G endpoint and received by the local application. In a 3G-324M session, the digit is sent via the H.245 UserInputIndication(UII) message. An application can use the Dialogic® software to detect a H.245 UII digit by waiting for the M3GEV_H.245UII_RCVD event. The application can wait to receive H.245UII events during a 3G multimedia session and take action on receipt of the event.

Fast Video Update Request Generation/Detection — In a 3G-324M session, an endpoint might need to request a full video frame update, called an I-Frame update, from the remote endpoint to provide a reference video frame to facilitate decoding or to refresh the video image. The endpoint requests an I-Frame from the remote terminal by sending an H.245 Miscellaneous Command Fast Video Update (FVU) message.

The Dialogic® 3G-324M software can be set to generate an H.245 FVU to a remote endpoint to request that the remote endpoint refresh the video by generating a new I-Frame. To send an FVU request to a remote endpoint, the application calls the `m3g_SendH245MiscCmd()` function with the `m3g` control device handle and the `h245MiscCmdType=M3G_E_FAST_UPDATE_PICTURE`. Requesting an FVU is a viable option at the start of a recording, because the `mm_Record()` function will wait up to 5 seconds for an I-Frame to start the recording so that the beginning of the video is not corrupt.

In the opposite direction, a remote endpoint can send the H245 FVU Request to the 3G multimedia application. An application can use the Dialogic® software to detect the H.245 FVU message by waiting for the `M3GEV_H245_MISC_CMD_RCVD` event with `h245MiscCmdType= M3G_E_FAST_UPDATE_PICTURE`. The Dialogic software can be set up to automatically generate an I-Frame by using the `m3g_SetParm()` function to set the parameter `M3G_E_PRM_RELAY_FASTUPDATE_TO_MEDIA_DEV` equal to true. Automatic I-Frame generation is only valid when a video transcoding connection is made between the `m3g` device and the `mm` device using the `dev_PortConnect()` function.

Step 6. Terminating a 3G-324M Session

Sometime after media streaming has been established, the 3G-324M session can be terminated through the 3G-324M protocol. The following steps are performed to terminate a 3G-324M call, to terminate the session, and to instruct the Dialogic® 3G-324M system software to clean up to the proper states and properly de-allocate resources.

Stopping Media Streaming

The application will call the `m3g_StopMedia()` function to independently stop audio and video streaming to/from the H.223 multiplexer/demultiplexer. The `m3g_StopMedia()` function is called with the `m3g` audio handle to terminate the audio multiplexing/demultiplexing session. Separately, the `m3g_StopMedia()` function is called with the `m3g` video device handle to terminate the video multiplexing/demultiplexing session. The `M3GEV_STOP_MEDIA_CMPLT` event is returned when media streaming termination is complete.

StopH.245 - Terminating the H.245 Session

After stopping audio and video media in both directions, the final step in terminating the 3G-324M session is to terminate the 3G-324M session using the `m3g_StopH245()` function (see Figure 19). The application can call the `m3g_StopH245()` function to terminate and release all H.245 resources. If an `EndSession` event was not received prior to this point from the remote endpoint, a call to the `m3g_StopH.245()` function triggers the Mux3G firmware to Close Logical Channels and send an `EndSession` command to the remote 3G endpoint indicating that the session has ended and will be terminated without further communication.

Receiving EndSession from remote terminal — An `M3GEV_ENDSESSION_RCVD` may be returned to the application to signify that the remote endpoint terminated the H.245 session. An `EndSession` command must be the last 3G-324M message sent from the remote endpoint signifying the H.245 session has ended and that no more communication should be expected. If an `EndSession` event is returned to the 3G-324M protocol stack, the application must call the `m3g_StopH245()` function to terminate the local session.

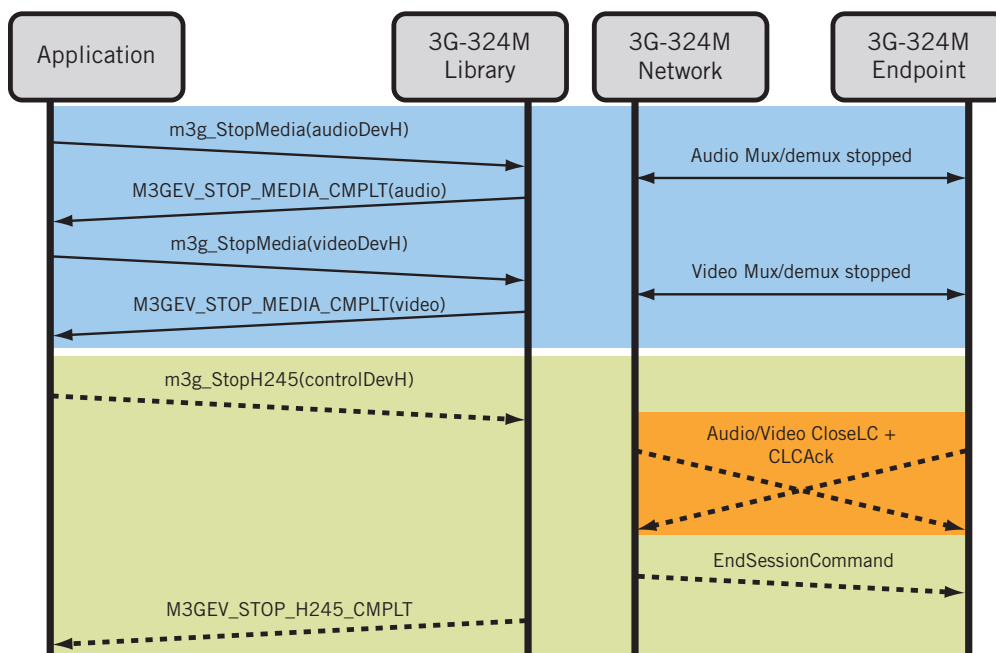


Figure 19. StopH.245 - Terminate the H.245 session

Step 7. Ending a 3G Call – Disconnecting the Bearer Channel

A 3G call is ended by disconnecting the bearer channel through the network. In a Release 99 PSTN network, the bearer channel containing the data stream is disconnected through normal ISDN or ISUP signaling methods. In a Release 4 IP network, it is expected that the call will be disconnected using BICC.

Loss of Session

If a bearer channel is disconnected, closed, or corrupted while a 3G-324M session is in progress, the Mux3G firmware will determine that the frame synchronization has been lost. Then, a M3GEV_FRAME_LOST event will be generated to the application. Subsequently, the Mux3G firmware will internally tear down active 3G-324M sessions and present the M3GEV_ENDSESSION_RCVD event to signal that the remote endpoint has terminated the session. No matter the disconnect method, it is expected that an application will follow the process in *Step 6, Terminating a 3G-324M Session*, to instruct the Dialogic® 3G-324M system software to clean up to the proper states and properly de-allocate resources.

Step 8. Disconnecting Devices

Connections can optionally be disconnected after the H.245 session is stopped, or can remain connected for the next 3G call established with a remote endpoint.

Disconnecting Media Port Connections

As in the case with the dev_PortConnect() function, in order to disconnect audio and video connections between the m3g device and the mm device, the dev_PortDisconnect() function must be called four times, once for each desired transmit media stream (see Figure 20). The dev_PortDisconnect() function removes the unidirectional packet connections between the one or more transmit ports of one device to the receive port of another device. In that case, two audio and two video transmit streams are disconnected, one audio Tx stream and one video Tx stream for each device, as follows:

1. **Disconnect m3g device Tx Video Stream** — Call dev_PortDisconnect() function with the m3g Tx video port_info list.
2. **Disconnect m3g device Tx Audio Stream** — Call dev_PortDisconnect() function with the m3g Tx audio port_info list.
3. **Disconnect mm device Tx Video Stream** — Call dev_PortDisconnect() function with the mm Tx video port_info list.
4. **Disconnect mm device Tx Audio Stream** — Call dev_PortDisconnect() function with the mm Tx audio port_info list.

Note: Tx and Rx directions above represent internal media stream direction relative to the device, not direction relative to the 3G network.

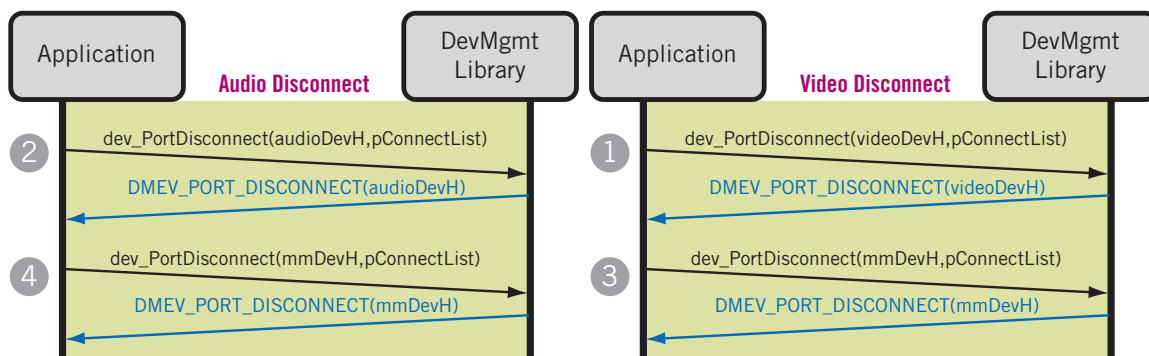


Figure 20. Disconnect Media Port Connections

Disconnecting Network Device

On the aggregate network side, the m3g device can be disconnected from an aggregate line device bearer channel to disconnect the multiplexer output. In a Release 99 PSTN Network, the m3g device will be disconnected from the Dialogic® Global Call Software Network dti device using the dev_Disconnect() function. The dev_Disconnect() function will be called once with the m3g control handle and once with the dti handle to disconnect the aggregate timeslot connection. In a Release 4 IP Network, the m3g device is disconnected from the ipm device using the dev_PortDisconnect() function. The dev_PortDisconnect function is called twice, once with the m3g control port list and once with the ipm port list, similar to the method described in the previous “Disconnecting Media Port Connections” section. Devices must be disconnected, and “disconnect complete” events received, prior to closing the devices and terminating the application.

Step 9. Closing Devices

Before an application is exited, the application must call the Close() function on the devices that were opened by the application. This terminates the queuing of the library events for the device, de-allocates the resources that were reserved for the device and returns the device to a known terminated state. The Close() function is only called after DMEV_DISCONNECT or DMEV_PORT_DISCONNECT events are received after disconnecting the devices.

For every 3G-324M channel being used by the application, the application must:

- **Close the m3g control device** — The application calls the m3g_Close() function using the m3g control device handle.
- **Close the m3g audio device** — The application calls the m3g_Close() function using the m3g audio device handle.
- **Close the m3g video device** — The application calls the m3g_Close() function using the m3g video device handle.
- **Close the mm devices** — The application calls the mm_Close() function using the mm device handle.

Once all m3g devices are closed, the application must:

- **Close the m3g board device** — The application calls the m3g_Close() function using the m3g board device handle, which may have been opened to configure systemwide settings for all 3G-324M endpoints in the system.

Step 10. Exiting an Application

The last 3G-324M call an application must make before the application is exited is the function m3g_Stop(). The m3g_Stop() function will stop the 3G-324M library and release all allocated resources. The m3g_Stop() function returns M3G_SUCCESS when the 3G-324M library was successfully stopped.

Establishing a MONA 3G Multimedia Session with the Dialogic® 3G-324M API

Application Note

Summary

In order to establish a 3G multimedia session, a multimedia server must pass through a 3G gateway that can demultiplex the audio and video streams or must directly support the 3G-324M protocol. The Dialogic® 3G-324M API, along with the Dialogic® Multimedia API, can provide a developer the option of reducing costs by providing a one unit solution for a 3G Multimedia server. The Dialogic® 3G-324M API provides a straightforward approach to session control by giving the application developer control of the 3G-324M session establishment without the complications introduced by the various sub-protocols of the 3G-324M standard. Support for H.324M Annex K, also known as MONA, provides that 3G-324M sessions established by the Dialogic® 3G-324M API connect quickly. A separate multimedia device, supported by the Dialogic® Multimedia API gives the developer rich media streaming capabilities to the 3G network. This application note provided an introduction to the 3G-324M standard and the Dialogic® implementation through the Dialogic® 3G-324M API session establishment, and provided steps to guide those who opt to use the Dialogic® API to provide 3G multimedia services with MONA support.

For More Information

Dialogic® 3G-324M Multimedia Gateway Demo Guide —
http://www.dialogic.com/manuals/docs/3g324m_demo_v1.pdf

Dialogic® 3G-324M API Library Reference —
http://www.dialogic.com/manuals/docs/3g324m_api_v4.pdf

Dialogic® Multimedia API Library Reference —
http://www.dialogic.com/manuals/docs/multimedia_api_v1.pdf

Dialogic® Device Management API Library Reference —
http://www.dialogic.com/manuals/docs/device_mgmt_api_v7.pdf

Dialogic® Multimedia Programming Guide —
http://www.dialogic.com/manuals/docs/multimedia_programming_lin_v2.pdf

Example code that demonstrates 3G-324M multimedia capability is included in the 3G-324M Multimedia Gateway Demo that is provided in the following Dialogic® product releases:

- Dialogic® Multimedia Software for AdvancedTCA Releases 1.1 and 2.0 (Note: Release 1.1 will not support MONA)
- Dialogic® Host Media Processing Software Release 3.1LIN
- Dialogic® Multimedia Kit Software Release 1.0 for PCIe

Example code for that demonstrates multimedia device capability is included in the Multimedia Demo that is provided in the following Dialogic® product releases:

- Dialogic® Multimedia Software for AdvancedTCA Releases 1.1 and 2.0 (Note: Release 1.1 will not support MONA)
- Dialogic® Host Media Processing Software Release 3.1LIN
- Dialogic® Multimedia Kit Software Release 1.0 for PCIe

www.dialogic.com

Dialogic Corporation
9800 Cavendish Blvd., 5th floor
Montreal, Quebec
CANADA H4M 2V9

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH PRODUCTS OF DIALOGIC CORPORATION OR ITS SUBSIDIARIES ("DIALOGIC"). NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN A SIGNED AGREEMENT BETWEEN YOU AND DIALOGIC, DIALOGIC ASSUMES NO LIABILITY WHATSOEVER, AND DIALOGIC DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF DIALOGIC PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHT OF A THIRD PARTY.

Dialogic products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Using the AMR-NB resource in connection with a Dialogic® product does not grant the right to practice the AMR-NB standard. To seek a patent license agreement to practice the standard, contact the VoiceAge Corporation at <http://www.voiceage.com/licensing.php>.

Dialogic may make changes to specifications, product descriptions, and plans at any time, without notice.

Dialogic is a registered trademark of Dialogic Corporation. Dialogic's trademarks may be used publicly only with permission from Dialogic. Such permission may only be granted by Dialogic's legal department at 9800 Cavendish Blvd., 5th Floor, Montreal, Quebec, Canada H4M 2V9. Any authorized use of Dialogic's trademarks will be subject to full respect of the trademark guidelines published by Dialogic from time to time and any use of Dialogic's trademarks requires proper acknowledgement.

The names of actual companies and products mentioned herein are the trademarks of their respective owners. Dialogic encourages all users of its products to procure all necessary intellectual property licenses required to implement their concepts or applications, which licenses may vary from country to country.

Any use case(s) shown and/or described herein represent one or more examples of the various ways, scenarios or environments in which Dialogic products can be used. Such use case(s) are non-limiting and do not represent recommendations of Dialogic as to whether or how to use Dialogic products.