VINETIC

VINETIC-4VIP PEB 3324
VINETIC-4M PEB 3314
VINETIC-2VIP PEB 3322
VINETIC-0 PEB 3320

VINETIC Packet Voice (RTP) Applications

Host Software Implementation Guidelines

Wired Communications

Infineon technologies

Never stop thinking.

Application Note
CONFIDENTIAL

Revision History:	**2003-07-07**	DS1.0
Previous Version:	Application Note, DS0.1, 2003-06-02	

Page	**Subjects (major changes since last revision)**
	Review Feedback incorporated

Edition 2003-07-07

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
81669 München, Germany

Attention please!

The information herein is given to describe certain components and shall not be considered as warranted characteristics.

Terms of delivery and rights to technical change reserved.

We hereby disclaim any and all warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

Infineon Technologies is an approved CECC manufacturer.

Information

For further information on technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies Office in Germany or our Infineon Technologies Representatives worldwide (**www.infineon.com**).

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

CONFIDENTIAL

Preface

The VINETIC (VIP and M versions) provides packet voice support for VoIP and VoATM implementations. To build a complete system solution, a Host controller is used to interface to VINETIC device(s) and is responsible for configuring, controlling and managing the packet voice streams. In terms of the software partitioning to achieve VoIP or VoATM, there is some functionality that needs to be implemented on the Host controller. This application note addresses this topic and is intended as a starting guide for software developers who wish to use the VINETIC along with the Host controller for VoIP or in other words RTP over IP.

Table of Contents **Page**

List of Figures

Page

List of Tables **Page**

1 Introduction

The VINETIC (VIP and M versions) provides packet voice support for VoIP and VoATM implementations. The enhanced DSP (EDSP) within VINETIC provides packetization support for RTP/RTCP per RFC 1889 or AAL2 per ITU-T I.366.2 depending on the firmware downloaded during initialization. It should be noted that this does not imply that the entire protocol (RTP/RTCP or AAL2) is implemented within the VINETIC. The VINETIC just makes it easier for handling the RTP/RTCP or AAL2 by providing preprocessed packet voice, events and statistics support conforming to these protocols. The host controller interfacing to the VINETIC is still responsible for creating/closing, maintaining the sessions and other subtleties of the protocol(s). It should also take care of some additional handling of the processed information provided by the VINETIC to complete the protocol requirements. The biggest advantage herein is the fact that VINETIC provides support for packet voice with complete RTP/RTCP header information along with the voice payload. The data is formatted exactly as per the standard. In other words, the host controller is relieved of the effort of creating the headers and formatting the data for packet voice. This helps in conserving precious horsepower on the controller which can be then utilized for other purposes. This note attempts to identify what the host Controller needs to implement in terms of the software requirements to complete a VoIP system.

Attention: This note does not cover implementation details for "Mixers" and "Translators" as specified by RFC 1889. It is assumed that the VINETIC will be used in endpoint implementations, typically for Line cards, IADs, and low end Gateways, where this functionality is not required. Also when conferencing is required, VINETIC provides the support on chip. With this assumption in the background, there are a number of feature support requirements per RFC 1889 which need not be implemented on the host controller for Packet Voice applications using VINETIC. It is assumed that the reader of this note is familiar with Voice over Packet technology in general and is upto speed on the programming details for the VINETIC device. All description in this note refers to a 4 channel VINETIC device.

2 Software Architecture

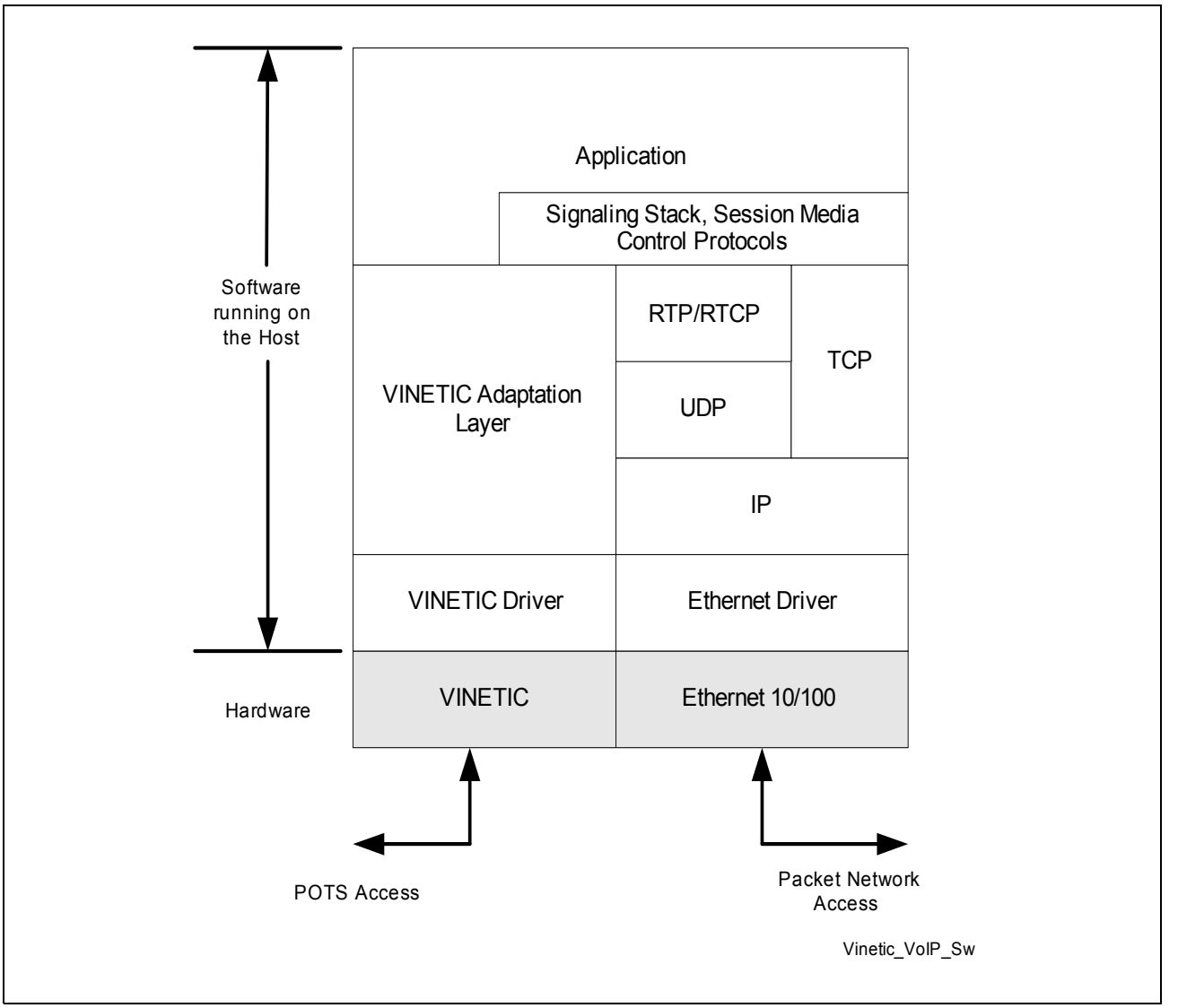

Figure 1 Functional Software Block Diagram for VoIP

Figure 1 provides a very high level overview of the various software blocks involved in the implementation of a VoIP system using VINETIC. The picture attempts to highlight in a simple manner the general partitioning of the various functional blocks. It should be noted however that this could differ based on customer specific implementation(s).

- **VINETIC:** This is the hardware which provides POTS access. The VINETIC device enables the user to bridge Analog POTS access to TDM or Packet Network. Specifically in the context of this note, it should be noted that the hardware provides packet preprocessing support for RTP/RTCP. Hence it should not be misconstrued from the above diagram that the RTP/RTCP implementation is entirely the responsibility of the host controller. There is a split of the functionality and is described in more detail in the next few sections of this note.

- **Ethernet 10/100:** This is the hardware which provides the packet network access. The hardware is capable of sending and receiving Ethernet frames.
- **VINETIC Driver:** This is the low level software driver for the VINETIC device. It provides low level access to the internal registers of the device for control and configuration. The driver supports Vinetic device Short commands and Mailbox commands for EOP, SOP, COP and IOP operations. It also provides support for handling packet in and packet out mailbox operations for VoIP scenario.
- **Ethernet Driver:** This is the low level software driver for the Ethernet device. It provides low level access to the internal registers, FIFOs for control, configuration and transmission/reception of IP packets.
- **VINETIC Adaptation Layer:** This software layer provides the high level API to the Application hiding all the nitty gritty details of the low level device driver access to VINETIC. The Adaptation layer provides functionality to control/configure POTS access, POTS signaling apart from using the capabilities of the chip set for packetization of voice using compressed voice.
- **TCP/UDP/IP:** Typically, this is the transport layer protocol stack responsible for the TCP and UDP data transfers. In most cases, the protocol stack is an integrated part of the Real Time Operating system. In some cases, customers may use home grown protocol stack or license it from other vendors. It is also very common to have this protocol stack provide a standard Socket Layer interface for the Application to interface to it.
- **RTP/RTCP:** This software module is responsible for providing end to end network transport functions suitable for applications transmitting real time data such as audio and video. The RTP is responsible for services related to payload type identification, sequence numbering and time stamping. The RTCP is responsible for delivery monitoring. Both RTP and RTCP typically use UDP transport and the interface is usually achieved through the standard Socket Layer Interface. The RTP/RTCP module however needs to provide an API for the Application to manage the various real time streams and coordinate appropriate actions based on the statistics received from the RTCP. In the specific context of our discussion where Vinetic is used in the system, some of the above mentioned functionality is supported on the hardware. Details of the functionality split are described in the next few sections of this note.
- **Signaling Stack and Session Media Control Protocol(s):** This is one of the most important software subsystems required to achieve packet voice functionality. The signaling stack is basically responsible for setting up, tearing down sessions and working in conjunction with Session media description protocols to establish voice over packet connections. The signaling stack in a broader sense need to bridge the signaling events coming out of the POTS access side (onhook, offhook, DTMF digits etc) and translate them into equivalent Network signaling messages to be sent to the peer on TCP. The reverse is also true where the signaling events received on the packet side need to be interpreted and corresponding states activated for the telephony. Typically, simple state machines are run for the Telephony side signaling to detect on-hook, off-hook, DTMF, tones and Dial pulses. On the Network signaling

front, there are multiple choices available today. Systems are implemented using H.323, SIP, MGCP or MEGACO. They work in close conjunction with the RTP/RTCP module to establish audio/video sessions with peer(s). In specific case like that of SIP, additional protocols like Session Description Protocol (SDP) are also involved. It is not the intention of this note to expand on the details of these protocols. The one point however which needs to be noted is that the Signalling protocol stacks interface with the underlying transport protocol (usually TCP) using the Socket layer interface and additionally with the RTP/RTCP module (UDP).

3 RTP Support

RFC 1889 specifies Real time Transport Protocol (RTP). The specification details mechanisms to provide end to end delivery services for data with real-time characteristics such as audio. Typically RTP runs on top of UDP/IP.

3.1 Implementation Of RTP Software Module in General

Typically designed to satisfy the needs of multi participant multimedia conferences, RTP is not limited to that particular application. For the implementation(s) we talk about here, RTP will be used to define mechanisms to transmit and receive data with real time properties (Audio) between endpoints (typically connections between IADs, Gateways and so on). Sessions are controlled by Signaling components and the voice traffic is carried over UDP/IP transport. For such applications, implementation of “Translators” as defined in RFC 1889 is not mandatory. As far as the “Mixer” functionality is concerned, it is essential when support for “conferencing” is required. With this context in mind, the implementations need to address the following:

- Setup and Tear down of RTP sessions. Maintain a mapping of the Synchronization source with the active voice channel on the POTS chip set and the transport layer port(s).
- Ensure that the information received from the Signaling component on the session media description is used for creation of the RTP Header along with the audio payload.
- Track the Payload type during a session by working closely with the Signaling component. This is relevant only if there happens to be a change in the payload type (codec) during the course of a call.
- Jitter buffering of the received data to ensure smooth playback. This is optional on the host controller because in some systems, the buffering is achieved on the DSP itself.

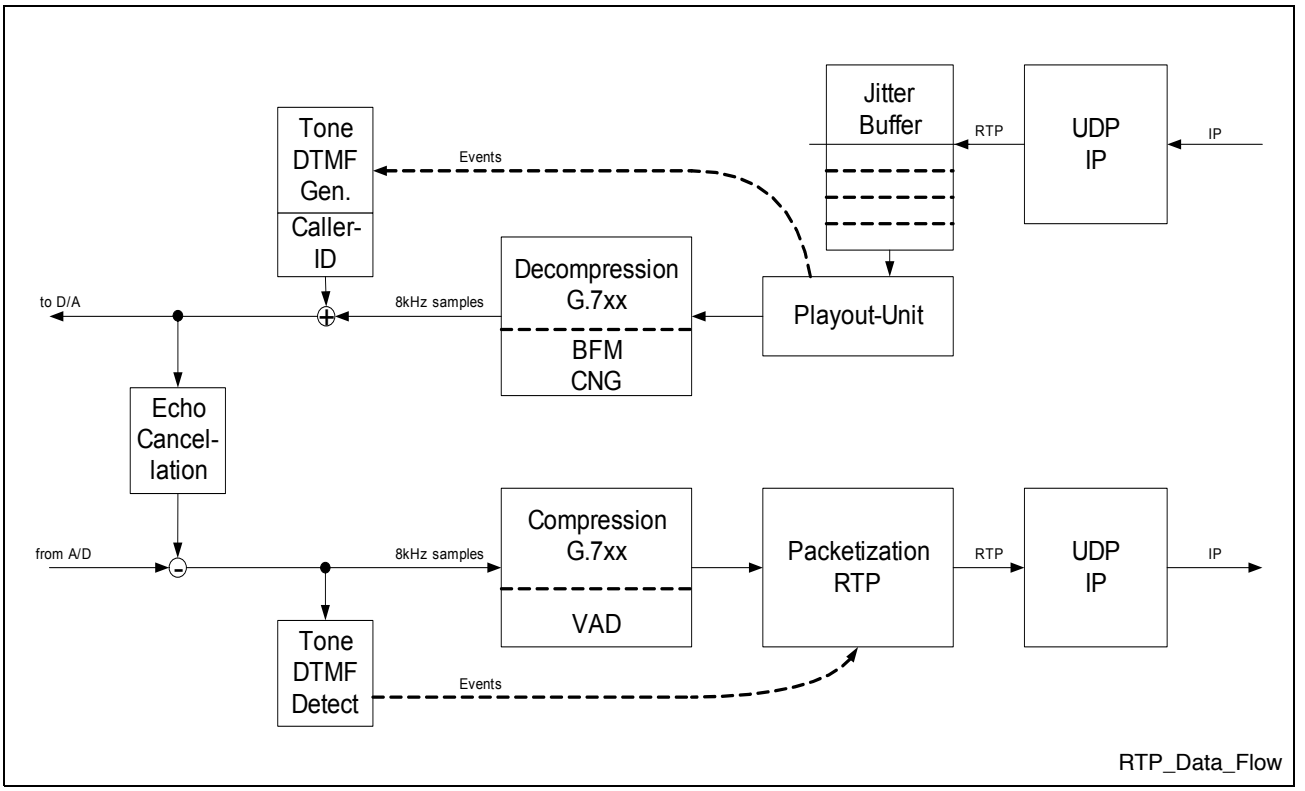

Figure 2 Data Flow for Packet Voice (RTP)

3.2 VINETIC Specific Implementation

In this section, an attempt is made to highlight the functionality that needs to be built and run on the host controller interfacing with the VINETIC for packet voice applications using RTP/RTCP. A high level overview of what a typical API implementation would be is provided with explanation. This by no means should be treated as a software design but should serve as a guideline to decide what partitioning needs to be done between the host controller and the VINETIC. Also provided for each functionality are the details of what commands need to be executed on the VINETIC if relevant.

At a minimum, the host controller needs to implement the following functionality.

3.2.1 RTPStartSession

This function should create a Session Object with a unique identifier for all future references. It will identify an RTP session to all the other functions of the module. A Synchronization source identifier and the VINETIC POTS channel in consideration should be specified while invoking this function. It is believed that in the specific case of the applications discussed herein, there will be one synchronization source only for each RTP session on a per active voice channel of VINETIC. Also the POTS channel number corresponds to the available free channel on the VINETIC(s) which is most likely tracked

by an onboard Resource Manager Software module. All other operations on this RTP session can be invoked only after the creation of this Session Object.

Attention: The concept of having independent SSRC identifier for each POTS channel on a Vinetic device as mentioned above can be debated because the clock used is the same for all channels. This being the case, one could argue that the SSRC identifier needs to be same. IThe choice is implementation specific although it does not create any problem in using an independent SSRC identifier for each POTS channel on the VINETIC device. As long as the host controller maintains a mapping of the SSRC identifier with the POTS channel and transport address for every Vinetic device, systems would be interoperable.

This API would be typically called by the Application when it is ready to establish an RTP session with its peer endpoint. It is assumed that at this point, the Signaling component running on the local endpoint has already negotiated with the peer endpoint on the capabilities and agreed upon the transport layer addresses for receive/transmit (UDP port numbers). It will be the responsibility of the Application to generate a random SSRC for the session. The hints provided in the Appendix A.6 of RFC 1889 should serve as a starting point. It should be noted that since we are not addressing multicast sessions with participants joining in and leaving out autonomously, the probability of having SSRC collisions would be very low as long as the guidelines of RFC 1889 are followed. Upon the successful call of this function, the Application should have a mapping of the SSRC Identifier and POTS channel number on the VINETIC(s) with a Unique Session Object Identifier which can be used for all further references or function calls of the module related to this session.

Attention: For this functionality the VINETIC does not need any interaction with the host controller. It is assumed that the host controller as a part of the initial boot up would have already activated the CODER module of the VINETIC. No other operations on the CODER module can be performed without activating it. Refer to the Preliminary User's Manual (Software Description). The relevant EOP command is "Coder Control". Apart from this, it is also required to configure the Coder for RTP support during initial boot up. This is done using the EOP command "Coder Configuration (RTP Support). Refer to the Preliminary User's Manual (Software Description). Please note that this command needs to be executed only if the host controller wants the channels to start with a specified Time stamp value. If this is not necessary then the "Coder Configuration (RTP Support) command can be ignored. Also important is the support for signaling events like DTMF and ATD. The VINETIC needs to be configured to generate RTP events for DTMF, AT detection and also support DTMF generation when RTP events for this case are received. This can also be done as a part of the initial boot up where the

SIGNALING module of the VINETIC has to be activated. Refer to the EOP commands "Signaling Control", "Signaling Channel" "DTMF/AT Generator" and "DTMF Receiver" detailed in the Preliminary User's Manual (Software Description).

3.2.2 RTPStopSession

This function should delete the Session Object created previously. One should close the RTP connection established **(*RTPCloseConnection*)** for the different Addresses in the session (In our case point to point) before invoking this function. The Session Object Identifier should be specified while invoking this function.

This functionality is just the opposite of ***RTPStartSession***. From the host controller's perspective, this would entail removing of all the mappings it maintained earlier on the SSRC identifier, POTS channel number on the VINETIC(s) and the unique Session Object Identifier. These would now be available for reuse for future sessions.

Attention: For this functionality the VINETIC does not need any interaction with the host controller.

3.2.3 RTPOpenConnection

This function will establish a connection to all addresses (In our case point to point) which have been added to the session by ***RTPAddAddress***. RTP packets can be sent only after the connection is established. The Session Object Identifier should be specified while invoking this function.

This function can be called only after ***RTPAddAddress*** establishes a logical mapping between the unique Session Object Identifier, POTS channel number on VINETIC, Local Host IP Address, Local Host UDP port, Remote Host IP Address and Remote Host UDP port. It should be noted that the Signaling component running on the local host would provide information on the Remote addresses. For the local address, the UDP port could be taken up as the first available even port starting from a default. The next port (always odd numbered) is assigned for the RTCP. Other mechanisms of local port assignment may also be implemented on the host controller. The host controller will have to establish bindings with the underlying transport layer (UDP/IP) to enable transmit and receive or RTP data. This would typically entail calling of the socket layer APIs which most TCP/UDP/IP stacks provide to open socket connections for RTP and RTCP, bind and start listening on the ports for receive information.

For this API functionality, there are a few interactions required on the part of the host controller with the VINETIC. They are detailed in the specific order they have to be called:

1. **EOP Command "Coder Channel Configuration (RTP Support)"**
   Refer to the Preliminary User's Manual (Software Description). This command is on a per POTS channel basis and has to be called before activating the Coder Channel.

The User's Manual defines the various bit definitions for this command. It should be noted however that the POTS Channel number ("CHAN" bits) and Synchronization source number ("SSRC" bits) related to the RTP Session Object Identifier is determined from the mapping stored in the host controller and used for this command. The host controller is also required to provide a starting Sequence Number ("SEQ-NR" bits) for this command.

2. **EOP Command "Coder Channel JB Configuration"**
   Refer to the Preliminary User's Manual (Software Description). This command is on a per POTS channel basis and has to be called before activating the Coder Channel. The User's Manual defines the various bit definitions for this command. The POTS channel number of the VINETIC which is related to the RTP Session Object Identifier is determined from the mapping stored in the host controller and used for the "CHAN" bits in this EOP command (First Command word).
3. **EOP Command "Signaling Channel Configuration (RTP Support)"**
   Refer to the Preliminary User's Manual (Software Description). This command is on a per POTS channel basis and has to be called before activating the Coder Channel. The User's Manual defines the various bit definitions for this command. The POTS channel number of the VINETIC which is related to the RTP Session Object Identifier is determined from the mapping stored in the host controller and used for the "CHAN" bits in this EOP command (First Command word). It should be noted that this command is necessary only if the user wants the support for RFC 2833 implemented for transmission of DTMF events as RTP packets and also receipt of the same from the peer endpoint.
4. **EOP Command "Coder Channel"**
   Refer to the Preliminary User's Manual (Software Description). This command is on a per POTS channel basis. It is assumed that the Signaling component on the local host at this point in time has all the negotiated parameters for the media session from its peer. This would include Encoder type, Packet frame time, and support for Silence suppression, CNG. The User's Manual defines the various bit definitions for this command. It should be noted however that the "EN" bit (Bit 15 of the 3rd Command Word) is responsible for activating or deactivating the coder channel with the rest of the programmed parameters. As a part of the ***RTPOpenConnection*** API functionality this "EN" bit shall be turned ON. The POTS channel number of the VINETIC which is related to the RTP Session Object Identifier is determined from the mapping stored in the host controller and used for the "CHAN" bits in this EOP command (First Command word). The Signal array addresses are configured based on the system requirements. For example ALM Output 1 connected to CODER Input 1 for Channel 1.

3.2.4 RTPCloseConnection

This function will close down all connections to the addresses of an RTP session. This function should be called before ***RTPStopSession***. The Session Object Identifier should

be specified while invoking this function. This function just closes an active RTP connection on the Local host. It should be noted that all the other information related to the RTP session: Local Address, Remote Address, Media session parameters, POTS channel mapping are still persistent. In other words, the Coder channel is just deactivated to stop transmitting and receiving RTP packets. Typically this functionality is called when the Signaling component on the host controller tears down an active voice channel (either initiated from the local or remote end: Phone going ON HOOK). For this API functionality, there is just one interaction required on the part of the host controller with the VINETIC. This is related to deactivating the specific POTS Coder Channel resource associated with the RTP Session.

1. **EOP Command "Coder Channel"**
   Refer to the Preliminary User's Manual (Software Description). This command is on a per POTS channel basis. The User's Manual defines the various bit definitions for this command. It should be noted however that the "EN" bit (Bit 15 of the 3rd Command Word) is responsible for activating or deactivating the coder channel with the rest of the programmed parameters. As a part of this API functionality the "EN" bit shall be turned OFF. The POTS channel number of the VINETIC which is related to the RTP Session Object Identifier is determined from the mapping stored in the host controller and used for the "CHAN" bits in this EOP command (First Command word). In order not to disturb any other settings on this voice channel as configured earlier, the host can initiate an EOP read of this Coder Channel command and toggle the "EN" bit before writing it back again to the VINETIC.

3.2.5 RTPAddAddress

This function will add a transport address to the RTP session created earlier by ***RTPStartSession***. The Session Object Identifier, Remote host connection port information and Local host connection port information should be specified while invoking this function. This function can also be extended to take as input POTS endpoint to support the on chip conferencing feature of VINETIC. In our context, there will be only point to point transfer of audio and hence there will be only one call of ***RTPAddAddress*** required for operations on a single RTP session.

Attention: VINETIC provides support for multiparty conferencing on the chip. The hardware and firmware architecture is highly modular. It is possible to support multiparty conferencing (up to 6 parties) with any combination of POTS and packet endpoints. This is done using the Signal Array module on the chip which is programmable. Hence an implementation using VINETIC does not require any additional RTP/RTCP handling support for "Mixing" on the host controller. Low level driver APIs can be executed to configure the chip for conferencing existing media streams. The host controller will however have to keep track of what

streams are being 'mixed'. The above API can be extended to included support for the on chip conferencing if required.

Attention:

3.2.6 RTPRemoveAddress

This function will remove a transport address from the RTP session created earlier by ***RTPStartSession***. The Session Object Identifier, Remote host connection port information and Local host connection port information should be specified while invoking this function. If the concept of on chip conferencing is supported in ***RTPAddAddress***, then ***RTPRemoveAddress*** should implement the functionality of excluding endpoints from a conferencing session. In our context, there will be only point to point transfer of audio and hence it is not anticipated that there will be any requirement for the implementation of this function. However it should serve as a placeholder for future expansion of capabilities.

Attention: For this functionality the VINETIC does not need any interaction with the host controller.

3.2.7 RTPSendPacket

This function will send an RTP packet to all the addresses of an RTP session (In our case only one destination address). The Session Object Identifier and the RTP packet complete with header information and voice payload should be specified while invoking this function.

Depending on how the host controller has the VINETIC(s) configured, once the POTS channels are activated for RTP support, packetized voice for upstream (To the Network) is available every packet frame interval in the Packet out Mailbox. If the VINETIC is configured to generate an Interrupt, host can service this using the short commands to determine the length of the packet available in the mailbox and then reading it out in a loop every interrupt. If a polling mechanism is used, the host controller can determine when the Packet out Mail box has relevant data and extract it using the same mechanism detailed above. The details of this mechanism and the exact sequence of short commands to be used are detailed in the User's Manual (Software Description).

The data retrieved from the Packet Out Mailbox of the VINETIC shall have the VINETIC specific information in the first two words. For a specific active POTS VINETIC channel, the Packet Out Mailbox can provide VOP packets, Event packets (EVT) or SID packets. It is the responsibility of the host controller to verify the Channel number and the Length fields and correlate the RTP packet information starting from the 3rd word to an established Transport layer address (Socket). The host controller is then responsible for calling the lower layer socket interface ("sendto") to send the RTP packet over UDP/IP to the destination

3.2.8 RTPSetPayloadType

This function will set the payload type of the packets that will be sent out. The Session Object Identifier and the Payload type should be specified while invoking this function. This function can be called during the start of the session and can also be subsequently invoked during run time to change the payload type if required.

Typically the payload type is set during the start of the call after the session media parameters are negotiated by the Signaling component. This is programmed for the VINETIC as a part of the EOP Command “Coder Channel” described in ***RTPOpenConnection***. However it is possible that the Payload type changes during the course of a call. In this case the host controller can dynamically request the VINETIC to adapt to the new Payload type on the active voice channel. It is recommended that the host controller attempts an EOP read operation “Coder Channel” for the active channel, modifies the payload type bit field and writes it back (EOP write operation “Coder Channel”) to the VINETIC. The active POTS channel number for VINETIC is derived from the mapping of the unique Session Object Identifier maintained by the host controller.

Attention: It is however not recommended to change the Payload type during the course of a call (run time).

3.2.9 RTPReceivedPacketCallback

The host controller should implement the functionality of registering a callback function with the lower layer Transport interface (UDP/IP stack) to receive incoming RTP packets. The host is responsible for correlating the incoming RTP packets on a specific socket to the unique Session Object Identifier and determining the VINETIC POTS channel for which it is meant to be played downstream (From the Network). Once this mapping is located, the host should identify the type of the RTP packet received and classify it as a Voice packet/SID packet or an EVT packet. This can be done by examining the Payload type. Once this classification is done, the host should use either a VOP or EVT write command to the VINETIC. Care should be taken to determine if there is enough space in the Packet in box by checking with a short command **rFIBXMS** before using the VOP or EVT commands. Details of this are described in the User’s Manual (Software Description).

4 RTCP Support

RFC 1889 specifies RTP Control Protocol (RTCP). It is based on the periodic transmission of control packets to all the participants in the session, using the same distribution mechanism as the data packets. The underlying protocol must however provide multiplexing of the data and control packets, for example using separate port numbers with UDP.

4.1 Implementation Of RTCP Software Module in General

A major part of the RTCP software module is normally capitalized within the RTP software module and hence hidden from the user. In other words, whenever an RTP session is established, it is mandatory to open an RTCP session as well. The host typically uses the information received from the Signaling stack (H.323/SIP/MGCP) to configure the UDP ports for voice on the local end. Per RFC 1889, the voice UDP port is always even numbered used for RTP. The next UDP port (always Odd numbered) is reserved for the RTCP data for the specific voice session. Hence the creation/deletion of the RTCP session implicitly happens whenever the host creates/deletes an RTP session.

The responsibility of the RTCP software module running on the host controller is then restricted to the following:

- Ensuring that the RTCP statistics on a per voice channel (session) are sent out at periodic intervals as defined by RFC 1889. This could be “Sender Reports” if the endpoint is an active sender or “Receiver Reports” if it is not an active sender.
- Ensuring that the received RTCP statistics for a particular voice channel (session) are made available to a higher level Management Entity. The information could be used by the Management entity to monitor the condition of the network for transmission and reception of voice packets and accordingly request change in media session parameters dynamically under Application control. Typical scenario could be to request the signaling component to switch to a low compression codec from a high compression codec previously negotiated if there is an indication of ample bandwidth availability. Of course, there has to be a close interaction of all the software modules with the Signaling stack and session media control protocols.
- Sending and receiving of SDES and BYE packets.
- Setting of parameters like CNAME, NAME, EMAIL, PHONE, LOC, TOOL, NOTE and PRIV items of SDES packets for a voice session per RFC 1889.
- Getting parameter details like CNAME, NAME, EMAIL, PHONE, LOC, TOOL, NOTE and PRIV items of SDES packets for a voice session per RFC 1889.

4.2 VINETIC Specific Implementation

As mentioned earlier, it is anticipated that the VINETIC would be used in applications requiring packet voice between endpoints (point to point) with a tight control on session negotiation. In other words, the chip set will be designed into POTS Line cards, IADs, and low end gateways where the endpoint is always an “active sender”. In these scenarios, it is not expected to support features for multicasting, participation of “receivers only” in the sessions and usage of “loosely controlled” sessions where participants enter and leave without membership control or parameter negotiation. In this case, the RTCP packets which are relevant are:

- SR Sender report, for transmission and reception statistics
- SDES Source description items (Optional)
- BYE Indicates end of participation (Optional)

4.2.1 RTCP Sender Report

For every voice channel established on the VINETIC, the chip set provides a pre prepared RTCP Sender Report formatted according to RFC 1889 on query by the host. The RTCP statistics for an active voice channel can be extracted from the VINETIC by the host controller using an “EOP” command. Refer to the Preliminary User’s Manual (Software Description). The Command “Coder Channel Statistics” provides a prepared RTCP Sender report as specified in RFC 1889. The only fields missing in the report are the RTCP Sender report header, 32 bit “SSRC of the Sender”, 32 bit “NTP Time stamp”, 32 bit “Last SR Time stamp”, and 32 bit “Delay since Last SR”. These have to be affixed by the host controller to complete the report before transmitting it out onto the network.

It should be noted that the VINETIC provides the RTCP statistics as a “Sender Report” only. A pre prepared “Receiver Report” is not available. Since the VINETIC is always used in an “Active Sender” endpoint, the requirement of generating “Receiver Report” is not valid.

Attention: It is possible for the host to transmit a “Receiver Report” also by tailoring the information received from the VINETIC. However, as mentioned above, the usage of VINETIC in the applications described above does not warrant the transmission of a “Receiver Report”.

In terms of the actual breakdown of the host’s responsibility to send the SR:

- Host maintains mapping of the RTP sessions established for the active voice channels on the VINETIC. When an RTP session is established the RTCP part is also set up. Similarly on the teardown of an RTP session, the RTCP part is also deinitialized.
- Host is responsible for scheduling the transmission of the SR periodically for an active voice channel. The periodicity is governed by the rules specified in the RFC 1889. A simplistic method however could be to schedule it every 5 seconds. It is

however advisable to vary the interval between the transmission of RTCP packets randomly as specified in RFC 1889.

- Host does not have to support compound RTCP packets for transmission but must be able to receive and decode them.
- Host needs to have a mechanism of generating an NTP time stamp. It is permissible to use the sampling clock to estimate the elapsed wall clock time and use it for the NTP time stamp.
- For every scheduled transmission of the SR, the host queries the VINETIC for the RTCP statistics using the EOP command mentioned earlier for the active channel. The details of the command is given below

Table 1 RTCP Support: Commands sent to VINETIC by the HOST

Command	Description
0x8600	EOP Read Command for Channel 0. Least significant four bits represent channel number
0x730E	Command Coder Channel Statistics (RTCP), Length 0x0E words

Table 2 RTCP Support: Response received by the HOST from VINETIC

Command	Description
0x8600	First command word sent by host earlier
0x730E	Second command word sent by host earlier
WORD1	Lower 16 bits of the RTP Time stamp
WORD2	Higher 16 bits of the RTP Time stamp
WORD3	Lower 16 bits of Sender’s packet count
WORD4	Higher 16 bits of Sender’s packet count
WORD5	Lower 16 bits of Sender’s octet count
WORD6	Higher 16 bits of Sender’s octet count
WORD7	Lower 16 bits of the Receiver’s SSRC
WORD8	Higher 16 bits of the Receiver’s SSRC
WORD9	8 bits of Receiver’s fraction lost and 8 bits of Receiver’s packets lost
WORD10	Higher 16 bits of Receiver’s packets lost
WORD11	Lower 16 bits of extended highest sequence number received
WORD12	Higher 16 bits of extended highest sequence number received
WORD13	Lower 16 bits of Interarrival Jitter
WORD14	Higher16 bits of Interarrival Jitter

As seen above from the response received from the VINETIC, the format is in line with the RTCP Sender Report specified in RFC 1889 but for the fields related to the RTCP Sender report header, 32 bit “SSRC of the Sender”, 32 bit “NTP Time stamp”, 32 bit “Last SR Time stamp”, and 32 bit “Delay since Last SR”. The host controller needs to complete this information before it requests the underlying transport protocol (UDP/IP) to send the information out to the network.

A write command will reset the entire statistics for the channel. This means that all counters maintained within the VINETIC for the channel would be reset. This would be under the host control. Another situation when the statistics for a specific active channel could get reset is when the SSRC value for incoming packets gets changed or the Sender SSRC is modified by the explicit control of the host.

4.2.2 SDES Packet

For the type of applications mentioned in this note where VINETIC is used, it is not anticipated to have multicast sessions or multiple applications operating in concert using cross-application binding through some means for each participant. In this context, it becomes entirely **optional** to transmit SDES RTCP packets.

If it is decided to implement the support for transmission of SDES packets, then it is recommended to support just the CNAME item. All other protocol requirements in terms of the periodicity of transmission of the SDES packet, maintaining the mapping of the CNAME for each session with the active voice channel would be the responsibility of the host.

Incoming SDES packets could also be ignored.

4.2.3 BYE Packet

For the type of applications described in this note, there will be a Signaling component which is responsible for setting up and tearing down the sessions. Hence the possibility of having a “loosely controlled membership” does not exist. In this context, the support for BYE RTCP packet is also entirely **optional.**

CONFIDENTIAL

Terminology

A	
AAL2	ATM Adaptation Layer 2
ATD	Answering Tone Detection
C	
CAS	Channel Associated Signaling
CCS	Common Channel Signaling
CNG	Comfort Noise Generation
COP	Coefficient OPeration
D_0	
DSP	Digital Signal Processor
DTMF	Dual Tone Multi Frequency
E	
EOP	Enhanced DSP OPeration
EVT	Event Transmission Operation
I	
IAD	Integrated Access Device
IOP	Interface OPeration
IP	Internet Protocol
P	
POTS	Plain Old Telephone System
R	
RFC	Request For Comments
RTCP	Real time Transport Control Protocol
RTP	Real time Transport Protocol
S	
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SOP	Status OPeration
T	
TCP	Transport Control Protocol

CONFIDENTIAL

U

UDP	User Datagram Protocol

V

VOP	VOice Packet operation
VoATM	Voice over Asynchronous Transfer Mode protocol
VoIP	Voice over Internet Protocol

References

[1] RFC 1889: RTP, A Transport Protocol for Real Time Applications.

[2] RFC 2833: RTP Payload for DTMF digits, Telephony tones and Telephony signals

[3] RFC 3261: SIP: Session Initiation Protocol

[4] VINETIC-4VIP Preliminary Users Manual DS1, Oct 2002 Software Description Document.

http://www.infineon.com

Published by Infineon Technologies AG