ARCAM HDMI ARC Messaging

Draft Version v0.9

Please Note:

This document is intended as the starting point for a fully defined API between ARCAM devices and the ARC MCU firmware.

- Anything in RED is for discussion and is to be removed or addressed in the final version...

ARCAM
The West Wing
Stirling House
Waterbeach
Cambridge
CB25 9PB
United Kingdom

Version

Version	Date	By	Description
0.1	09/04/2018	Chris Mann	First draft based on a set of requirements supplied by Pete's email 05/04/2019
0.2	29/05/2019	Chris Mann	Data types section added, BIN16 data type replaces N-Bytes
0.3	05/09/2019	Chris Mann	Update following a meeting with Fiona and Ethan from Hualu • Little-Endian use for multi-byte numbers • Handshaking section added • Changes to Power in Feedback section to handle Sleep/Idle (Network Standby) states • Optional changes to the Upgrade Start message to include an MD5 checksum for the whole upgrade data file. • Changes to the Upgrade Data message to include a checksum of the packet upgrade data, and to fix a size for the upgrade data (Question outstanding)
0.4	06/09/2019	Chris Mann	• Upgrade Data packet checksum change to use checksum provided by Hualu • Upgrade Data packet size fixed at 512 bytes. • Removal of Upgrade Start MD5 Checksum as not required by Hualu
0.5	11/10/2019	Chris Mann	• Changed to use a 16 bit checksum in Upgrade Data
0.6	18/10/2019	Chris Mann	• Reverted to 8bit Checksum
0.7	07/11/2019	Chris Mann	• Addition of ARC Firmware Version message
0.8	18/11/2019	Chris Mann	• Creation of a new messages for handling Transmitter version request/send, and removal of transmitter version from ARC Version Number message.
0.9	22/11/2019	Chris Mann	• Changes to ARC Transmitter Version Number to add a build version number and reverse the order that the numbers are sent.

Contents

Please Note: 1
Version 2
Overview 4
Terms 4
References 4
API Scope 5
General principles 6
Communication 6
Messaging 6

Message flow 6
Message structure 6
Data Types 6
HDMI ARC Commands 8
Handshaking 9
System Ready 9
ARC Version Number 10
Control 10
Power 11
Volume 11
Mute 12
ARC Format 12
ARC Stream 12
Feedback 14
Power 14
Volume 15
Mute 16
ARC Stream 16
CEC Name 17
ARC Transmitter Version Number 17
Upgrade 18
Upgrade Start 18
Upgrade Data 19
Upgrade End 20

Overview

This document describes API that will be used to pass messages between ARCAM products and the HDMI ARC MCU.

At the time of writing the ARCAM products that will use the API are as follows:

- SA30
- ST60

The document splits the API into areas of functionality, and provides a brief overview of each functional area.

Terms

The following terms are used through the document:

Term	Description
ARC MCU	The chip on the HDMI ARC board that is used to communicate with the Device.
Device	The ARCAM device that is communication with the HDMI Module.
Message	The mechanism by which information is transferred between the device and the ARC MCU and visa-versa.
Sender	The MCU that initiates a message.
Receiver	The MCU that receives a message.

References

Title	Location

API Scope

For this document the API is split into a number of functional areas:

Functional Area	Description
Control	The ARC MCU sends the Device messages that cause the device to change its state.
Feedback	The Device sends messages top the ARC MCU to inform it of changes in the Device status.
Upgrade	Messages exchanged between the Device and the ARC MCU to control the upgrade of the ARC MCU.

General principles

The following general principles apply across all the API's functional areas.

Communication

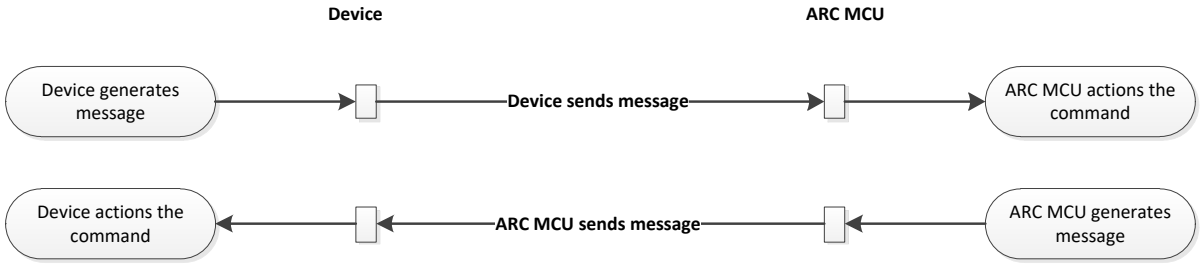
Communication between the ARC MCU and the Device is over RS232.

- Baud rate: 38400
- Data bits: 8
- Parity: None
- Stop bits: 1

Messaging

Message flow

All communication between the Device and the ARC MCU follows the sequence below.



Message structure

All messages will have the following structure:

Byte	Name	Description
0	Start of message (LSB)	0x23
1-2	Message Length	The length of the entire message including the Start and End of Message bytes, Message Length, and Command Type bytes. (N+2 bytes). This will be a UINT16 field.
3	Command Type	The command that is being sent in the message
4..N	Command Data	Data associated with the command
N+1	End of Message	0x0D

Message structure

Assumptions

- All messages will contain a single command
- The maximum message size will be 65536 bytes, although it is not expected that it will be this large.
- All messages will begin with a Start of Message byte (0x23) and end with an End of Message byte (0x0D)
- Numbers are transferred in little-endian format.

Data Types

The following data types are used in this document.

Note: All multi-byte numbers are in little-endian format.

Data Type	Description
UINT8	A 1-byte unsigned integer
UINT16	A 2-byte unsigned integer
UINT32	A 4-byte unsigned integer
BIN16	Used to transfer bulk data and strings. • The first 2 bytes contain the length of the following data bytes as a UINT16. • The remaining bytes contain the data. e.g. To send the string “Test” the data would be 0x04 0x00 0x54 0x65 0x73 0x74

HDMI ARC Commands

The commands that can exchanged between the Device and the ARC MCU are as follows:

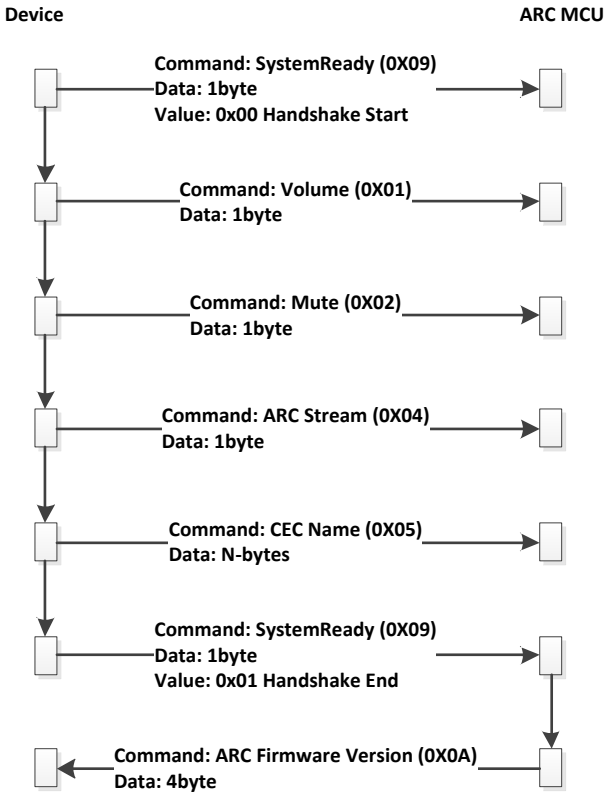
Command ID	Description
0x00	Power
0x01	Volume
0x02	Mute
0x03	ARC Format
0x04	ARC Stream
0x05	CEC Name
0x06	Upgrade Start
0x07	Upgrade Data
0x08	Upgrade End
0x09	System Ready
0x0A	ARC Firmware Version
0x0B	ARC Transmitter Version

Questions / Notes
Note: The same Command ID will be used to identify an action irrespective of whether it is generated by the Device or the ARC MCU.

Handshaking

It is anticipated that the ARC MCU will be ready to receive message from the Device before the Device is ready to send messages, hence the Handshaking process will involve the Device sending a series of messages to the ARC MCU to inform it that it is ready to receive messages, and of its current state.

The message sequence will be in the order show in the diagram below. A System Ready message (0x09) will be sent with a value HANDSHAKING_START (0x00), followed by a series of messages identifying the current state of the Device. Finally a second System Ready message (0x09) will be sent with a value HANDSHAKING_END (0x01) to identify that any subsequent messages from the Device will be commands.



On completion of the handshaking sequence the ARC module will send its current firmware versions to the device using the 0x0A ARC Firmware Version message.

System Ready

<table>
<tr><td>Command</td><td>System Ready</td></tr>
<tr><td>Direction</td><td>Device -> ARC MCU</td></tr>
<tr><td></td><td>The Device informs the ARC MCU the Device that it is performing Handshaking, and either about to send or complete sending status messages.

<table>
<tr><td colspan="3">Message : ARC MCU -> Device</td></tr>
<tr><td>Command ID</td><td>0x09</td><td>System Ready</td></tr>
<tr><td>Command Data</td><td>UINT8</td><td>The value supplied identifies the Handshaking state</td></tr>
</table>
</td></tr>
</table>

	<table><tr><td></td><td></td><td>• 0x00 Handshake Start • 0x01 Handshake End</td></tr></table>
Notes	• 0x01 Handshake End indicates that all status messages have been sent, and that the device is ready to receive messages.

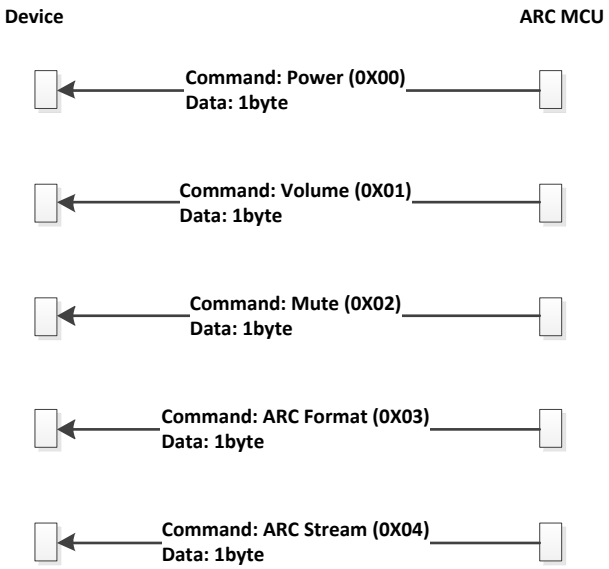
ARC Version Number

Command	ARC Version Number
Direction	ARC MCU->Device
	The ARC MCU informs the Device of its current firmware versions (for display purposes) **Message :** ARC MCU -> Device Command ID — 0x0A — ARC Firmware Version Major Version — UINT8 — The Firmware Major version Minor Version — UINT8 — The Firmware Minor Version
Notes	The ARC Version numbers will be used for display only.

Questions / Notes
Power State ON is assumed. Status messages are identical to the messages described in the Feedback Section.

Control

The ARC MCU will send the Device commands that will cause it to change its current state.



Power

<table>
<tr><td>Command</td><td>Power</td></tr>
<tr><td>Direction</td><td>ARC MCU -> Device</td></tr>
<tr><td></td><td>The ARC MCU informs the Device that it should change its power state.

<table><tr><td colspan="3">Message : ARC MCU -> Device</td></tr><tr><td>Command ID</td><td>0x00</td><td>Power</td></tr><tr><td>Command Data</td><td>UINT8</td><td>The value supplied identifies the Power status of the Device

• 0x00 Power Off
• 0x02 Power On</td></tr></table></td></tr>
<tr><td>Notes</td><td>• 0x00 Power Off – The Device will enter Standby state. This will put the device into Sleep or Idle mode dependent on the settings (Network Standby)
• 0x01 Power On – If in Standby state the Device will enter running state.</td></tr>
</table>

Volume

<table>
<tr><td>Command</td><td>Volume</td></tr>
<tr><td>Direction</td><td>ARC MCU -> Device</td></tr>
<tr><td></td><td>The ARC MCU informs the Device of a change in the Volume.

<table><tr><td colspan="3">Message : ARC MCU -> Device</td></tr><tr><td>Command ID</td><td>0x01</td><td>Volume</td></tr></table></td></tr>
</table>

	Command Data	UINT8	The value supplied will be: • A value between 0 and 99 • 0xF1 – Increment Volume • 0xF2 – Decrement Volume
Notes	ARCAM devices store volume in the range 0-99. If required a mechanism for translating this into another range can be supplied.		

Mute

Command	Mute		
Direction	ARC MCU -> Device		
	The ARC MCU informs the Device of a change in the Mute status.		
	Message : ARC MCU -> Device		
	Command ID	0x02	Mute
	Command Data	UINT8	The value supplied instructs the Device to change its Mute status: • 0x00 Muted • 0x01 Unmuted
Notes			

ARC Format

Command	ARC Format		
Direction	ARC MCU -> Device		
	The ARC MCU informs the Device of its current ARC format.		
	Message : ARC MCU -> Device		
	Command ID	0x03	ARC Format
	Command Data	UINT8	The following formats are available: • 0x00 ARC • 0x01 eARC
Notes			

ARC Stream

Command	ARC Stream
Direction	ARC MCU -> Device

<table>
<tr><td></td><td>The ARC MCU informs the Device of a change in the ARC Stream output status.

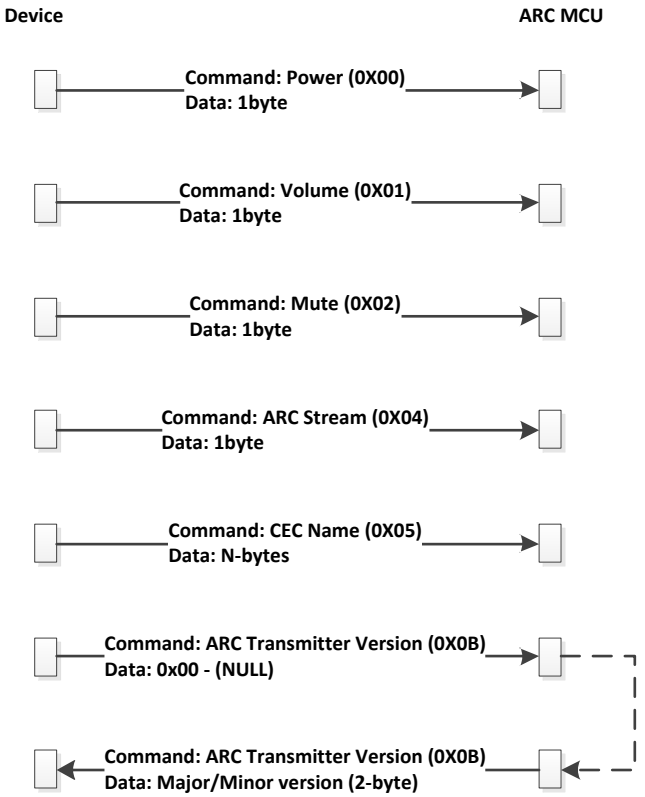
<table>
<tr><th colspan="3">Message : ARC MCU -> Device</th></tr>
<tr><td>Command ID</td><td>0x04</td><td>ARC Stream</td></tr>
<tr><td>Command Data</td><td>UINT8</td><td>The value supplied informs the device about the presence of an ARC Stream:

• 0x00 In-Active
• 0x01 Active</td></tr>
</table>
</td></tr>
<tr><td>Notes</td><td>• 0x00 In-Active – The Device will change its input to the default input TBD?
• 0x01 Active – The Device will change its input to the ARC Stream</td></tr>
</table>

Questions / Notes

Feedback

The Device will send the ARC MCU commands that inform it of the Devices current state.



Power

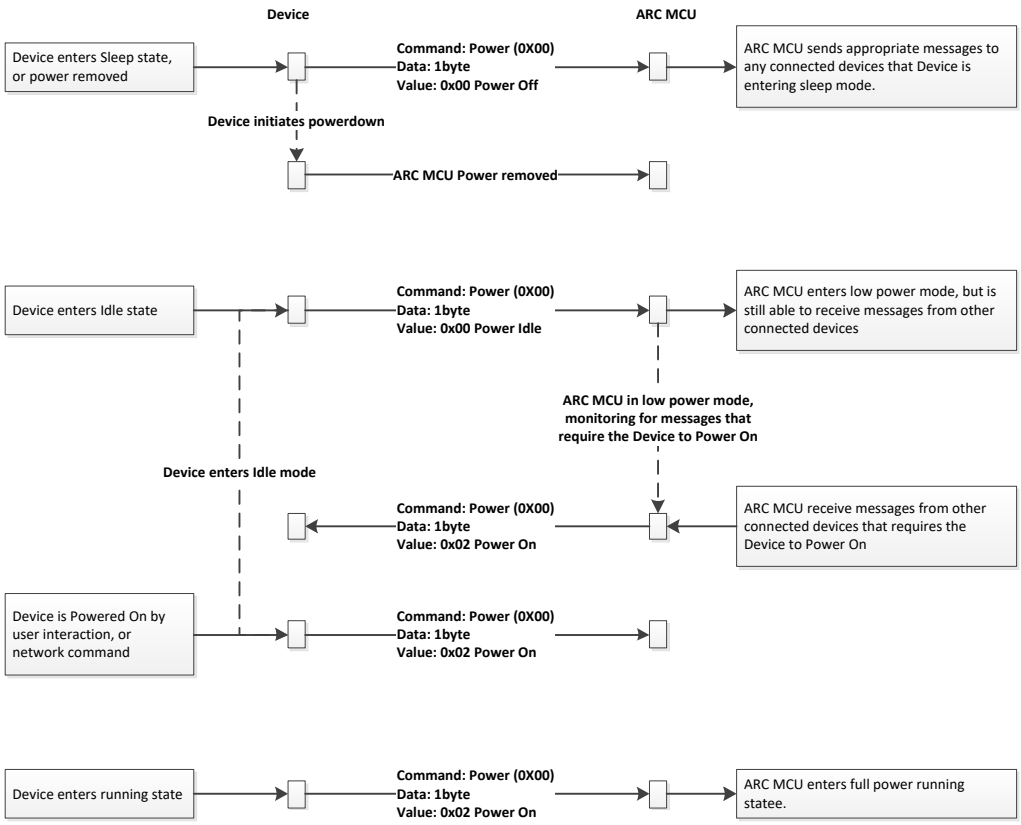
<table>
<tr><td>Command</td><td>Power</td></tr>
<tr><td>Direction</td><td>Device -> ARC MCU</td></tr>
<tr><td></td><td>The Device informs the ARC MCU that its power state change changed. This will be triggered:

• When the Device is powered up
• When the Device enters the running state from a standby state
• When the Device enters a standby state
• When the Device is powered down.

<table>
<tr><td colspan="3">Message : Device->ARC MCU</td></tr>
<tr><td>Command ID</td><td>0x00</td><td>Power</td></tr>
<tr><td>Command Data</td><td>UINT8</td><td>The value supplied identifies the Power status of the Device

• 0x00 Power Off
• 0x01 Power Idle
• 0x02 Power On</td></tr>
</table>
</td></tr>
<tr><td>Notes</td><td>Below is a description of when Power messages will be sent to the ARC MCU</td></tr>
</table>

- Power Off – This is sent prior to power to the ARC MCU being removed. The ARC MCU should attempt to inform any other connected devices as appropriate.
- Power Idle – This is sent prior to the Device entering idle mode. The ARC MCU should put itself into the lowest power state required to enable it to still receive messages from other devices that may require the Device to be brought out of its sleep state. Upon receipt of an appropriate message it should bring itself out of any low power state and send a Power On message to the Device.
- Power ON – The ARC MCU should enter normal running mode.



An attempt will be made to send a message on power-down, however it is not guaranteed that it will be successfully transferred before the Device loses power.

Volume

Command	Volume
Direction	Device -> ARC MCU
	The Device informs the ARC MCU of a change in the Volume.

Message : Device->ARC MCU		
Command ID	0x01	Volume
Command Data	UINT8	The value supplied will be

			• A value between 0 and 99
Notes	ARCAM devices store volume in the range 0-99. If required a mechanism for translating this into another range can be supplied.		

Mute

Command	Mute		
Direction	Device -> ARC MCU		
	The Device informs the ARC MCU of a change in the Mute status.		
	Message : Device->ARC MCU		
	Command ID	0x02	Mute
	Command Data	UINT8	The value supplied identifies the mute status of the Device: • 0x00 Muted • 0x01 Unmuted
Notes			

ARC Stream

Command	ARC Stream		
Direction	Device -> ARC MCU		
	The Device informs the ARC MCU of a change in the ARC Stream output status.		
	Message : Device->ARC MCU		
	Command ID	0x04	ARC Stream
	Command Data	UINT8	The value supplied identifies the whether the ARC input is active or not: • 0x00 In-Active • 0x01 Active
Notes	• 0x00 In-Active – The Device is NOT currently outputting the ARC Stream (A different input has been selected) • 0x01 Active – The Device is currently outputting the ARC Stream (The ARC input has been selected)		

CEC Name

<table>
<tr><td>Command</td><td>CEC Name</td></tr>
<tr><td>Direction</td><td>Device -> ARC MCU</td></tr>
<tr><td></td><td>The Device provides a unique name to the ARC MCU.

<table>
<tr><th colspan="3">Message : Device->ARC MCU</th></tr>
<tr><td>Command ID</td><td>0x05</td><td>CEC Name</td></tr>
<tr><td>Command Data</td><td>BIN16</td><td>The CEC Name that uniquely identifies the Device.</td></tr>
</table></td></tr>
<tr><td>Notes</td><td>In general this will be the Device model name (e.g. SA30)</td></tr>
</table>

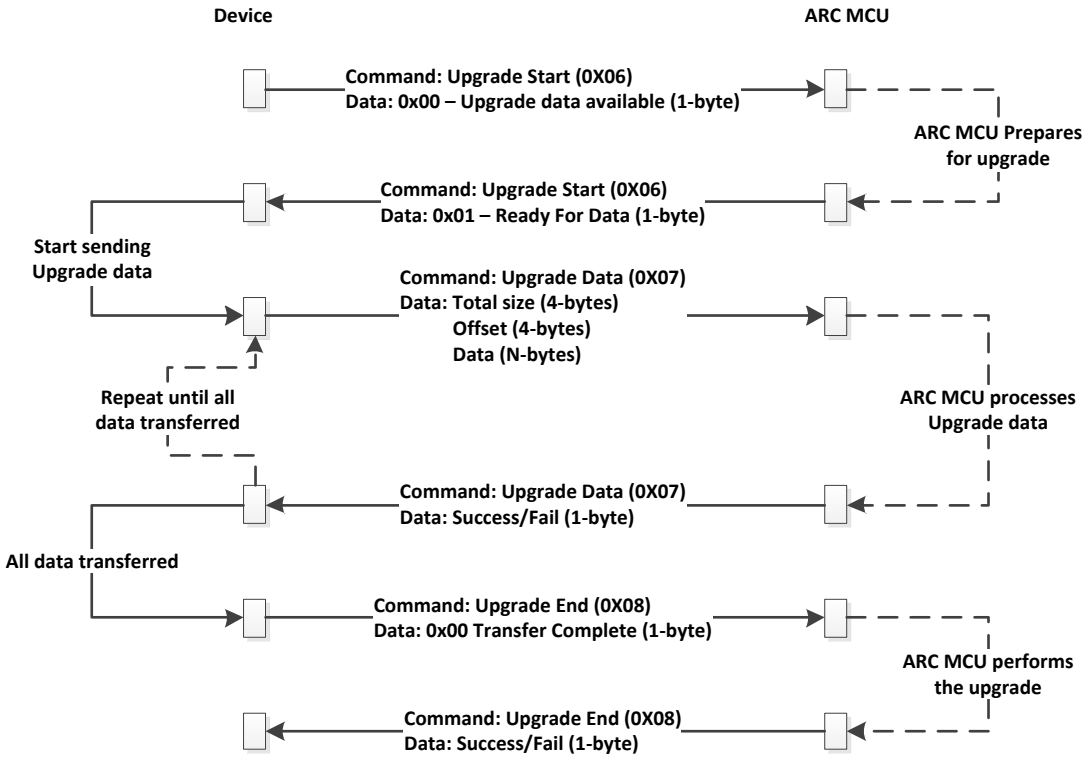
ARC Transmitter Version Number

<table>
<tr><td>Command</td><td>ARC Transmitter Version Number</td></tr>
<tr><td>Direction</td><td>ARC MCU->Device</td></tr>
<tr><td></td><td>The Devices requests the current transmitter firmware version (for display purposes)

<table>
<tr><th colspan="3">Message : Device -> ARC MCU</th></tr>
<tr><td>Command ID</td><td>0x0B</td><td>ARC Transmitter Firmware Version</td></tr>
</table>
<table>
<tr><th colspan="3">Message : ARC MCU -> Device</th></tr>
<tr><td>Command ID</td><td>0x0B</td><td>ARC Transmitter Firmware Version</td></tr>
<tr><td>Transmitter Build Version</td><td>UINT8</td><td>The HDMI Transmitter Build Version</td></tr>
<tr><td>Transmitter Minor Version</td><td>UINT8</td><td>The HDMI Transmitter Minor Version</td></tr>
<tr><td>Transmitter Major Version</td><td>UINT8</td><td>The HDMI Transmitter Major Version</td></tr>
</table></td></tr>
<tr><td>Notes</td><td>The ARC Transmitter Version numbers will be used for display only.</td></tr>
</table>

Upgrade

When an upgrade for the ARC MCU firmware is available the device will transfer the upgrade image to the ARC MCU. The ARC MCU will perform the upgrade of its firmware, and inform the Device when it is complete.



Upgrade Start

Command	Upgrade Start		
Direction	Device -> ARC MCU / ARC MCU -> Device		
	The Upgrade Start message is used by the Device to inform the ACR MCU that upgrade data is available. The ARC MCU then uses the same message to inform the Device that it is ready to receive the upgrade data.		
	Message : Device -> ARC MCU		
	Command ID	0x06	Upgrade Start
	Command Data	UINT8	The value indicates that the Device has Upgrade data for the ARC MCU • 0x00 Upgrade data available
	Message : ARC MCU -> Device		
	Command ID	0x06	Upgrade Start
	Command Data	UINT8	The value indicates that the ARC MCU is ready to receive the Upgrade data

			• 0x01 Ready For Data
Notes			

Upgrade Data

Command	Upgrade Data		
Direction	Device -> ARC MCU / ARC MCU -> Device		
	The Upgrade Data message is used by the Device to transfer the upgrade data to the ARC MCU. As it is expected that the Upgrade data will be too large to transfer in a single message the Upgrade Data message will be called repeatedly until all the data has been transferred. It will be possible to identify the position of the data packet in the overall Upgrade data from the Offset value. After transferring an Upgrade Data packet, the Device will wait for an Upgrade Data acknowledgement message from the ARC MCU before transferring the next packet.		
	Message : Device -> ARC MCU		
	Command ID	0x07	Upgrade Data
	Command Data	UINT32	The total size of the Upgrade data.
		UINT32	The offset in the upgrade data of the data contained in the packet.

		UINT8	Checksum – This will be a Checksum of the upgrade data contained in the current packet calculated using the following function
			
			`static uint8_t getChecksum8( const uint8_t *msg, uint32_t size )`
			`{`
			`  uint8_t sum1 = 0;`
			`  uint8_t sum2 = 0;`
			
			`  while( size-- )`
			`  {`
			`  sum1 += *msg++;`
			`  sum2 += sum1;`
			`  }`
			
			`  return sum1 ^ sum2 ;`
			`}`

		BIN16	The upgrade data.

	Message : ARC MCU -> Device Command ID: 0x07 — Upgrade Data Command Data: UINT8 — The value indicates that the ARC MCU has received the Upgrade data successfully • 0x00 Data transfer failed • 0x01 Data transfer success
Notes	The size of the whole packet will be 512 meaning that the Upload data will be 496bytes (512 – 5 bytes header/tail – 2bytes for BIN16 – 1 byte for checksum – 8 bytes for 2 UINT32 fields) If the Upgrade data transfer fails then the packet will be re-sent up-to 3 times (in total).

Upgrade End

Command	Upgrade End
Direction	Device -> ARC MCU / ARC MCU -> Device
	The Upgrade End message is used by the Device to inform the ACR MCU that the transfer of upgrade data is complete The ARC MCU then performs the upgrade and informs the Device that the upgrade is complete. **Message :** Device -> ARC MCU Command ID: 0x08 — Upgrade End Command Data: UINT8 — The value indicates that the Device has finished sending the Upgrade data to the ARC MCU • 0x00 Upgrade data transfer complete **Message :** ARC MCU -> Device Command ID: 0x08 — Upgrade End Command Data: UINT8 — The value indicates the completed status of the ARC MCU upgrade. • 0x01 Upgrade completed successfully • 0x02 Upgrade failed.
Notes	It is expected that the ARC MCU will respond after it has completed the upgrade process.

	When the ARC MCU has completed its upgrade it will reboot. During the reboot it may send spurious messages which should be ignored.