# Artistic Licence



# Art-Net mx

## Specification for the
## Art-Net mx Ethernet Communication Standard

Release V1.4  Document Revision AA 1/4/04

**Comments on Revision P:**

1. Editorial Changes.

**Comments on Revision Q:**

There are a number of subtle changes in this revision that may require some minor code changes in products that implement Art-Net. The following is a list of pointers to areas of the document that have changed.

1. The secondary IP address has been changed to a 10.x.x.x. This is an un-routable address.
2. The method of calculating the IP address from the MAC address has been improved.
3. Each packet definition now contains a detailed description of allowed implementation and private versus broadcast reply.
4. Programmable IP addresses added.
5. RDM Support Added
6. Server reply added

**Comments on Revision R:**

1. Type in ArtAddress->Swout corrected.

**Comments on Revision S:**

1. Typographic corrections.

**Comments on Revision AA:**

1. Addition of mx extensions.

**Overview:**

Art-Net is an Ethernet protocol based on the TCP/IP protocol suite. Its purpose is to allow transfer of large amounts of DMX512 data over a wide area using standard networking technology. Art-Net is primarily intended to operate over 10BaseT networks.

The latest revision of the protocol implements a number of new features and also simplifies the data transfer mechanism. The changes are all based on feed back from manufacturers who are using the protocol.

**Limitations:**

A theoretical limit of 255 universes of DMX512 exists in this specification. However a simplistic data rate comparison (DMX runs at 250KBaud, 10BaseT at 10MBaud) suggests a maximum of 40 universes of DMX is the limit. Art-Net uses a simple delta transmission compression technique that will provide about 60 universes. If an installation of more than say 20 universes is contemplated, then a 100BaseT backbone should be considered. If this is done, then the universe limit occurs only between the node and the backbone up-converter. i.e. there would be no problem.

**Terminology:**

- Node: A device that translates DMX512 to or from Art-Net is referred to as a Node.

- Universe: A single DMX512 frame of 512 channels is referred to as a Universe.

- Sub-Net: A group of 16 consecutive ports on the limited broadcast address is referred to as a sub-net. (Not to be confused with the subnet mask).

- A central controller or monitoring device (lighting console) is referred to as a Server.

- IP: The IP is the Internet protocol address. It is expressed in either a long word format (0x12345678) or dot format (2.255.255.255). Convention is that the former is hexadecimal and the latter is decimal. The IP uniquely identifies any Nodes or Servers on a network.

- Subnet Mask: Defines which part of the IP represents the Network address and which part represents the Node address. All Art-Net implementations require a Sub-Net mask of 255.0.0.0. This means that the first byte of the IP is the network address and the remaining three bytes are the Node address.

- Port: Actual data transmission on Art-Net uses the UDP protocol that operates 'on top of' the TCP/IP protocol. UDP data transfer operates by transferring data from a specific IP:Port address on a Node or Server to a second specific IP:Port address on a second Node or Server. Art-Net uses only one port address of 0x1936.

- Limited Broadcast: When a network first connects, the Server does not know the number of Nodes on the network, nor does it know their IP addresses. The Limited broadcast address allows the Server to send an ArtPoll to all Nodes on the network.

- Server: A generic term describing an Art-Net device with the primary task of generating control data. For example, a lighting console.

- Node: A generic term describing an Art-Net device with the primary task of receiving control data. For example, a dimmer or an Ethernet to DMX gateway.

- Media Server: A generic term describing an Art-Net device capable of generating control data based on the 'mx' Media Extensions to Art-Net.

## Ethernet Implementation:

### General Notes:

- All communication is UDP. Each packet format defined in this document form the Data field of an enclosing UDP packet.

- Packet formats are specified in a manner similar to C-language structures, in which all data items are considered to be unsigned integers of type INT8, INT16 or INT32 according to the number of bits. There are no hidden padding bytes, except at the very end of a packet, which may be rounded up to a multiple of 2 or 4 bytes. Extra bytes at the end of a valid received packet are ignored.

- The protocols are generalised for handling future versions with increased numbers of ports.

- Many bit data fields contain unused positions. These may be used in future versions of the protocol. They should be transmitted as zero and not tested by receivers.

- All packet definitions are designed such that their length can be increased in future revisions, whilst retaining compatibility. For this reason, only minimum packet length is checked in this protocol.

### Protocol Operation:

A Node operates in one mode, each Node having a unique IP address derived from its Ethernet MAC address. The UDP port used as sources and destinations is 0x1936.

### IP address configuration

The Art-Net protocol, by default, uses a Class A IP address scheme. This allows Art-Net products to communicate directly and without the need for a DHCP server to be connected to the network. The use of Class A addressing is allowed within a closed network. It is important to ensure that Art-Net data is not routed onto the Internet.

Products implementing Art-Net should default to the Primary IP address of 2.?.?.?.

The IP address consists of a 32 bit number designated as A.B.C.D. The lower the bytes B.C.D is calculated from the MAC address. The high byte 'A' is set to one of two values as shown in the following table.

The MAC address is a 48 bit number designated u:v:w:x:y:z. This is a globally unique number. The upper three bytes 'u.v.w' are registered to a specific organisation. The lower three bytes 'x.y.z' are assigned by that organisation. In order to ensure that there is minimal possibility of IP address conflicts between different manufacturers supporting Art-Net, the product OEM code is added to the MAC address.

The 'B' field of the IP address is calculated by adding the high byte of the OEM code with the low byte of the OEM code and the 'x' field of the MAC address.

On power up, the Node checks its configuration for IP addressing mode. If it has been programmed to use a custom IP address, the following procedure is not used.

| | IP Address A.B.C.D | | | | Subnet Mask |
|---|---|---|---|---|---|
| **'roduct Switch Settings** | **A** | **B** | **C** | **D** | |
| :ustom IP Programmed | As Programmed | | | | As Programmed |
| Jetwork Switch Off | 2 | x+OEM | y | z | 255.0.0.0 |
| Jetwork Switch On | 10 | x+OEM | y | z | 255.0.0.0 |

The sub-net mask is always initialised to 255.0.0.0, unless a custom IP address is in use. This means that the network address is the most significant 8 bits and the Node address is the least significant 24 bits of the IP address. This is a Class A network address and for this reason care must be exercised when connecting to other networks. If an installation requires connection of an Art-Net network to another network that has Internet access, then the connection must be implemented via a router that filters out the Class A addresses.

IP address Example

Given the following settings, the IP address calculation will be as follows:

Network Switch = Off
MAC address = 12.45.78.98.34.76
OEM code = 0x0010

Calculation:

IP Address A = 2 (Because Network switch is off).
IP Address B = 114 (98 + 0 + 16).
IP Address C = 34 (from MAC address).
IP Address D = 76 (from MAC address).

IP Address = 2.114.34.76.

Server Default Poll

By default a Server should poll both the primary and secondary Art-Net addresses:

- 2.255.255.255:0x1936 Primary Art-Net Address
- 10.255.255.255:0x1936 Secondary Art-Net Address

Network Topology:

Art-Net allows two network topologies to operate simultaneously:

1. Peer to Peer: This is an unmanaged network where multiple Nodes transfer data without the intervention of a server. All data transfer uses ArtDmx packets. All data is broadcast. This is the power on mode of operation for all Art-Net compliant products. In Peer to Peer mode, all IP packets are sent to the limited broadcast address 2.255.255.255 (or 10.255.255.255 depending upon the Network Switch setting), and are therefore received by all Nodes on the same local network.

2. Server to Peer: This is the most sophisticated implementation whereby one or more Nodes communicate with one or more central servers (lighting consoles). This mode of operation is programmable using the ArtPoll data packet. In Server to Peer mode, data transfer can be programmed to operate as broadcast or private.

The Universe Address of each DMX512 Universe is encoded as an 8-bit number. The high nibble is referred to as the Sub-net address and is set to a single value for each Node. The low nibble is used to define the individual DMX512 Universe within the Node.

This means that any Node must have:

1. One front panel "Sub-net" switch.
2. One front panel "Universe" switch for each implemented DMX512 input or output.

**Operation**

All UDP packets accepted by the Node conform to the Art-Net protocol specification as defined below. Any other packets are ignored.

**ArtPoll:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | Send ArtPollReply. |
| | Private Transmit | Server transmits this packet to a specific Server or Node IP address when a single device response is required. |
| | Broadcast | Server broadcasts this packet to poll all Servers and Nodes on the network. |
| Node | Receive | Send ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Send ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

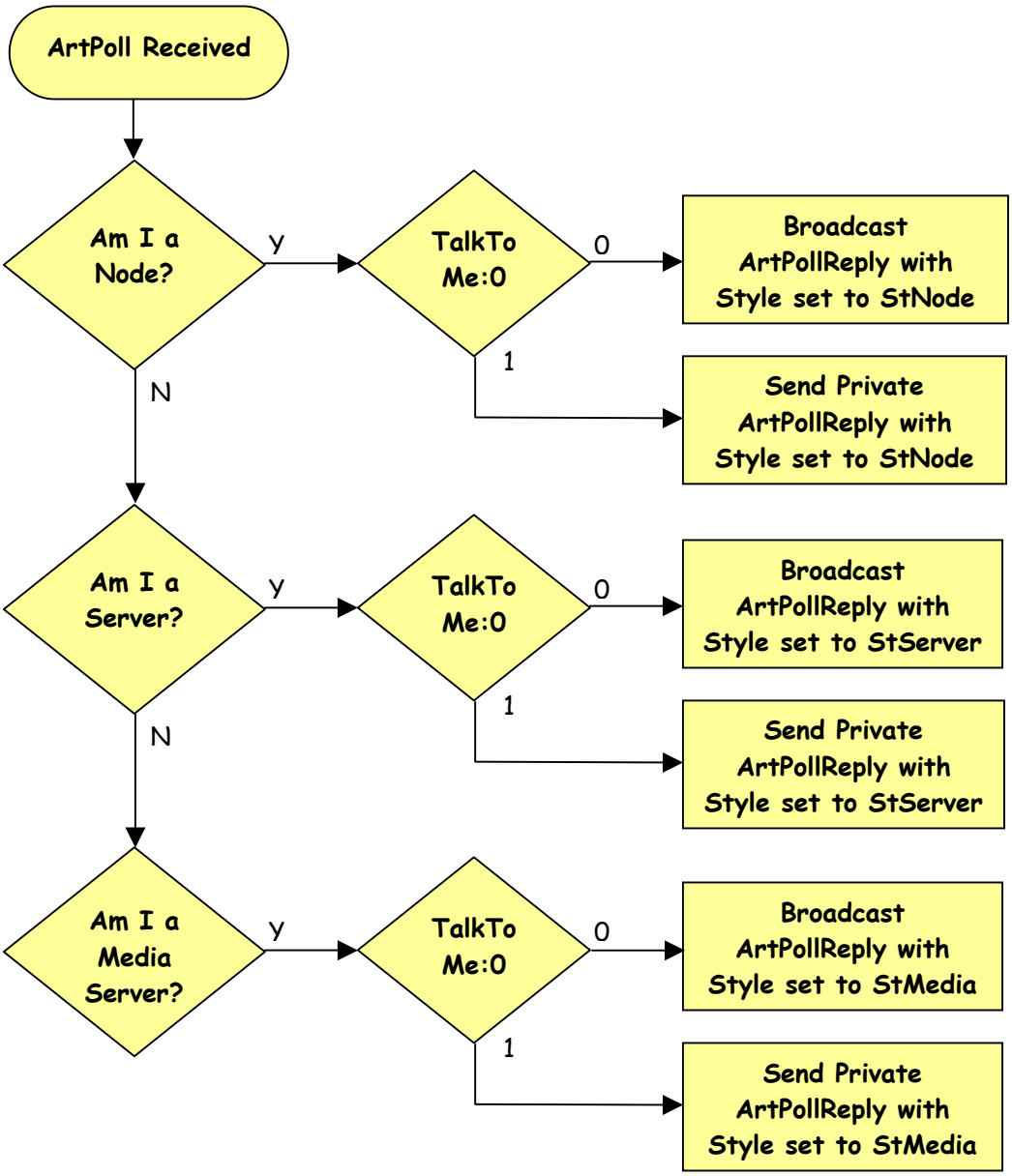
The ArtPoll packet is used to discover the presence of other Servers, Nodes and Media Servers. The ArtPoll packet is only sent by a Server. Both Servers and Nodes respond to the packet.

A Server broadcasts an ArtPoll packet to IP address 2.255.255.255 (sub-net mask 255.0.0.0) at UDP port 0x1936. This is the limited broadcast address:

The Server initially broadcasts ArtPoll in order to discover the presence of other Servers and Nodes on the network. The Server can then choose whether to continue using broadcast, or communicate privately with the detected network devices. (ArtPollReply includes the replier's IP address).

The Server may assume a maximum timeout of 3 seconds between sending ArtPoll and receiving all ArtPollReply packets.

The Server that broadcasts an ArtPoll should also reply to it's own message with an ArtPollReply. This ensures that any other Servers listening to the network will detect all devices without the need for all Servers connected to the network to send ArtPoll packets.


ArtPoll Received
Am I a Node?
y
TalkTo Me:0
0
Broadcast ArtPollReply with Style set to StNode
1
Send Private ArtPollReply with Style set to StNode
N
Am I a Server?
y
TalkTo Me:0
0
Broadcast ArtPollReply with Style set to StServer
1
Send Private ArtPollReply with Style set to StServer
N
Am I a Media Server?
y
TalkTo Me:0
0
Broadcast ArtPollReply with Style set to StMedia
1
Send Private ArtPollReply with Style set to StMedia

| ArtPoll | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int 16 | - | The OpCode defines the class of data following ArtPoll within this UDP packet.<br>Transmitted low byte first. See Table 1 for the OpCode listing.<br>Set to OpPoll. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14. Servers should ignore communication with nodes using a protocol version lower than 14. |
| 5 | TalkToMe | Int8 | - | Set behaviour of Node |
| | | | 7-2 | Unused, transmit as zero, do not test upon receipt. |
| | | | 1 | 0 = Only send ArtPollReply in response to an ArtPoll or ArtAddress. |
| | | | | 1 = Send ArtPollReply whenever Node conditions change. This selection allows the Server to be informed of changes without the need to continuously poll. |
| | | | 0 | 0 = Broadcast all further ArtPollReplys. |
| | | | | 1 = Send all future ArtPollReplys to the sender of this packet. |
| 6 | Pad | Int8 | - | Filler byte to make packet length even. |

**Table 1 - OpCodes:**

The following table details the legal OpCode values used in Art-Net packets:

| Opcodes | | |
|---|---|---|
| **Name** | **Value** | **Definition** |
| **OpPoll** | 0x2000 | This is an ArtPoll packet, no other data is contained in this UDP packet. |
| **OpPollReply** | 0x2100 | This is an ArtPollReply Packet. It contains device status information. |
| **OpOutput** | 0x5000 | This is an ArtDmx data packet. It contains DMX512 information for a single Universe. |
| **OpAddress** | 0x6000 | This is an ArtAddress packet. It contains remote programming information for a Node. |
| **OpInput** | 0x7000 | This is an ArtInput packet. It contains enable – disable data for DMX inputs. |
| **OpTodRequest** | 0x8000 | This is an ArtTodRequest packet. It is used to request a Table of Devices (ToD) for RDM discovery. |
| **OpTodData** | 0x8100 | This is an ArtTodData packet. It is used to send a Table of Devices (ToD) for RDM discovery. |
| **OpTodControl** | 0x8200 | This is an ArtTodControl packet. It is used to send RDM discovery control messages. |
| **OpRdm** | 0x8300 | This is an ArtRdm packet. It is used to send all non discovery RDM messages. |
| **OpVideoSetup** | 0xa010 | This is an ArtVideoSetup packet. It contains video screen setup information for nodes that implement the extended video features. |
| **OpVideoPalette** | 0xa020 | This is an ArtVideoPalette packet. It contains colour palette setup information for nodes that implement the extended video features. |
| **OpVideoData** | 0xa040 | This is an ArtVideoData packet. It contains display data for nodes that implement the extended video features. |

| Opcodes | | |
|---|---|---|
| **Name** | **Value** | **Definition** |
| **OpMacMaster** | 0xf000 | This is an ArtMacMaster packet. It is used to program the Node's MAC address, Oem device type and ESTA manufacturer code. This is for factory initialisation of a Node. It is not to be used by applications. |
| **OpMacSlave** | 0xf100 | This is an ArtMacSlave packet. It is returned by the node to acknowledge receipt of an ArtMacMaster packet. |
| **OpFirmwareMaster** | 0xf200 | This is an ArtFirmwareMaster packet. It is used to upload new firmware or firmware extensions to the Node. |
| **OpFirmwareReply** | 0xf300 | This is an ArtFirmwareReply packet. It is returned by the node to acknowledge receipt of an ArtFirmwareMaster packet. |
| **OpIpProg** | 0xf800 | This is an ArtIpProg packet. It is used to reprogramme the IP, Mask and Port address of the Node. |
| **OpIpProgReply** | 0xf900 | This is an ArtIpProgReply packet. It is returned by the node to acknowledge receipt of an ArtIpProg packet. |
| **OpMedia** | 0x9000 | This is an ArtMedia packet. It is Unicast by a Media Server and acted upon by a Server. |
| **OpMediaPatch** | 0x9100 | This is an ArtMediaPatch packet. It is Unicast by a Server and acted upon by a Media Server. |
| **OpMediaControl** | 0x9200 | This is an ArtMediaControl packet. It is Unicast by a Server and acted upon by a Media Server. |
| **OpMediaContrlReply** | 0x9300 | This is an ArtMediaControlReply packet. It is Unicast by a Media Server and acted upon by a Server. |

**Table 2 - OemCode:**

The following table details the registered OEM codes. The OEM code defines a specific manufacturer's product type. The OemCode is returned in the ArtPollReply:

| Value | Manufacturer | Product | Notes | RDM |
|---|---|---|---|---|
| 0x0000 | Artistic Licence | DMX-Hub | 4x DMX in, 4x DMX out | ✖ |
| 0x0001 | ADB | Netgate | 4x DMX in, 4x DMX out | ✖ |
| 0x0002 | MA Lighting | DMX-Hub | 4x DMX in, 4x DMX out | ✖ |
| 0x0003 | Artistic Licence | Ether-Lynx | 2x DMX in, 4x DMX out | ✓ |
| 0x0004 | LewLight | Capture v2 | TBA | ✖ |
| 0x0005 | High End | TBA | TBA | ✖ |
| 0x0006 | Avolites | TBA | TBA | ✖ |
| 0x0010 | Artistic Licence | Down-Lynx | 2x DMX out. Wall Panel. | ✖ |
| 0x0011 | Artistic Licence | Up-Lynx | 2x DMX in. Wall Panel. | ✖ |
| 0x0014 | Artistic Licence | Net-Lynx O/P | 2x DMX out. Boxed Product. | ✖ |
| 0x0015 | Artistic Licence | Net-Lynx I/P | 2x DMX in. Boxed Product. | ✖ |
| 0x0030 | Doug Fleenor Design | TBA | 2x DMX out. | ✖ |
| 0x0031 | Doug Fleenor Design | TBA | 2x DMX in. | ✖ |
| 0x0050 | Goddard Design | DMX-Link ™ O/P | 2x DMX out. | ✖ |
| 0x0051 | Goddard Design | DMX-Link ™ I/P | 2x DMX in. | ✖ |
| 0x0070 | ADB | Net-Port O/P | 2x DMX out. | ✖ |
| 0x0071 | ADB | Net-Port I/P | 2x DMX in. | ✖ |
| 0x0072-0x007f | ADB | Reserved | | |
| 0x008c | Zero 88 | TBA | 2x DMX out. | ✖ |
| 0x008d | Zero 88 | TBA | 2x DMX in. | ✖ |
| 0x008e | Flying Pig | TBA | 2x DMX out. | ✖ |
| 0x008f | Flying Pig | TBA | 2x DMX in. | ✖ |
| 0x0090 | ELC | ELC 2 | 2x DMX out. | ✖ |

| | | | | |
|---|---|---|---|---|
| 0x0091 | ELC | ELC 4 | 4x DMX in. 4x DMX out. | ✘ |

| Value | Manufacturer | Product | Notes | RDM |
|---|---|---|---|---|
| 0x0180 | Martin | Maxxyz | 4x DMX in. 4x DMX out. | ✖ |
| 0x0190 - 0x019f | Enttec | Reserved | | |
| 0x01a0 | IES | PBX | 1 x DMX in. 1 x DMX out. | ✖ |
| 0x01a1 | IES | Executive | 2 x DMX in. 2 x DMX out. | ✖ |
| 0x01a2 | IES | Matrix | 2 x DMX in. 2 x DMX out. | ✖ |
| 0x01a3 - 0x01af | IES | Reserved | | |
| 0x01b0 | EDI | Edig | 4x DMX in. 4x DMX out. | ✖ |
| 0x01c0 | Nondim Enterprises | OpenLux | 4x DMX in. 4x DMX out. | ✖ |
| 0x01d0 | Green Hippo | Hippotizer | Emulates 1x DMX in. | ✖ |
| 0x01e0 | VNR | Merger-Booster | 4x DMX in. 4x DMX out. | ✖ |
| 0x01f0 | Robe | ILE | 1x DMX in. 1x DMX out. | ✖ |
| 0x01f1 | Robe | ILE Controller | 4x DMX in. 4x DMX out. | ✖ |
| 0x0210 | Artistic Licence | Down-Lynx RDM | 2x DMX out. Wall Panel. | ✓ |
| 0x0211 | Artistic Licence | Up-Lynx RDM | 2x DMX in. Wall Panel. | ✓ |
| 0x0214 | Artistic Licence | Net-Lynx O/P RDM | 2x DMX out. Boxed Product. | ✓ |
| 0x0215 | Artistic Licence | Net-Lynx I/P RDM | 2x DMX in. Boxed Product. | ✓ |
| 0x0230 | Doug Fleenor Design | TBA | 2x DMX out. | ✓ |
| 0x0231 | Doug Fleenor Design | TBA | 2x DMX in. | ✓ |
| 0x0250 | Goddard Design | DMX-Link ™ O/P | 2x DMX out. | ✓ |
| 0x0251 | Goddard Design | DMX-Link ™ I/P | 2x DMX in. | ✓ |
| 0x0270 | ADB | Net-Port O/P | 2x DMX out. | ✓ |
| 0x0271 | ADB | Net-Port I/P | 2x DMX in. | ✓ |
| 0x0280 | LSC | Down-Lynx | 2x DMX out. | ✓ |
| 0x0281 | LSC | Up-Lynx | 2x DMX in. | ✓ |

| Value | Manufacturer | Product | Notes | RDM |
|---|---|---|---|---|
| 0x0300 | Golden Stage | DMX-net/O | 2x DMX out. | ✘ |
| 0x0301 | Golden Stage | DMX-net/I | 2x DMX in. | ✘ |
| 0x0303 – 0x030f | Golden Stage | Reserved | | ✘ |
| 0x8000 | ADB | Netgate XT | Video output and trigger inputs | ✘ |
| 0x8001 | Artistic Licence | Net-Patch | TBA | |
| 0x8002 | Artistic Licence | DMX-Hub XT | Video output and trigger inputs | ✘ |
| 0x8003 | Artistic Licence | Four-Play | Real time data record - playback | ✘ |
| 0x00ff | --- | --- | Development code. Not to be used in production products. | |

**Table 3 – NodeReport Codes:**

The following table details the NodeReport codes. TheNodeReport code defines generic error, advisory and status messages for both Nodes and Servers. The NodeReport is returned in both the ArtPollReply and ArtPollServerReply:

| Code | Mnemonic | Description |
|---|---|---|
| 0x0000 | RcDebug | Booted in debug mode (Only used in development) |
| 0x0001 | RcPowerOk | Power On Tests successful |
| 0x0002 | RcPowerFail | Hardware tests failed at Power On |
| 0x0003 | RcSocketWr1 | Last UDP from Node failed due to truncated length, Most likely caused by a collision. |
| 0x0004 | RcParseFail | Unable to identify last UDP transmission. Check OpCode and packet length. |
| 0x0005 | RcUdpFail | Unable to open Udp Socket in last transmission attempt |
| 0x0006 | RcShNameOk | Confirms that Short Name programming via ArtAddress, was successful. |
| 0x0007 | RcLoNameOk | Confirms that Long Name programming via ArtAddress, was successful. |
| 0x0008 | RcDmxError | DMX512 receive errors detected. |
| 0x0009 | RcDmxUdpFull | Ran out of internal DMX transmit buffers. |
| 0x000a | RcDmxRxFull | Ran out of internal DMX Rx buffers. |
| 0x000b | RcSwitchErr | Rx Universe switches conflict. |
| 0x000c | RcConfigErr | Product configuration does not match firmware. |
| 0x000d | RcDmxShort | DMX output short detected. See GoodOutput field. |
| 0x000e | RcFirmwareFail | Last attempt to upload new firmware failed. |
| 0x000f | RcUserFail | User changed switch settings when address locked by remote programming. User changes ignored. |

**Table 4 – Style Codes:**

The following table details the Style codes. The Style code defines the general functionality of a Server. The Style code is returned in ArtPollReply.

| Code | Mnemonic | Description |
|---|---|---|
| 0x00 | StNode | A DMX to / from Art-Net device |
| 0x01 | StServer | A lighting console. |
| 0x02 | StMedia | A Media Server. |
| 0x03 | StRoute | A network routing device. |
| 0x04 | StBackup | A backup device. |
| 0x05 | StConfig | A configuration or diagnostic tool. |

**ArtPollReply:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| All devices | Receive | No Art-Net action. |
| | Private Transmit | Transmits this packet to a specific Server IP address, in response to an ArtPoll, if the TalkToMe field is '1'. |
| | Broadcast | Broadcasts this packet in response to an ArtPoll if the TalkToMe field is '0'. |

A device, in response to a Server's ArtPoll, sends the ArtPollReply. The 'TalkToMe' field in the ArtPoll packet can modify this default mode of operation. This packet is also broadcast by all Art-Net devices on power up.

| ArtPollReply | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpPollReply<br>Transmitted low byte first. |
| 3 | IP Address[4] | Int8 | - | Array containing the Node's IP address. First array entry is most significant byte of address. |
| 4 | Port | Int16 | - | The Port is always 0x1936<br>Transmitted low byte first. |
| 5 | VersInfoH | Int8 | - | High byte of Node's firmware revision number. The Server should only use this field to decide if a firmware update should proceed. The convention is that a higher number is a more recent release of firmware. |
| 6 | VersInfo | Int8 | - | Low byte of Node's firmware revision number. |
| 7 | SubSwitchH | Int8 | - | The high byte of the Node's Subnet Address. This field is currently unused and set to zero. It is provided to allow future expansion. |
| 8 | SubSwitch | Int8 | - | The low byte of the Node's Sub-net Address. This is the variable that addresses a Node within Art-Net.<br>In the Ether-Lynx and Netgate products, the front panel Sub-net 'switch' sets this field. |

| Field | Name | Size | Bit | Description | |
|---|---|---|---|---|---|
| 9 | OemHi | Int8 | - | The high byte of the Oem value. | |
| 10 | Oem | Int8 | - | The low byte of the Oem value.<br>The Oem word describes the equipment vendor and the feature set available. Bit 15 high indicates extended features available. Current registered codes are defined in Table 2. | |
| 11 | Ubea Version | Int8 | - | This field contains the firmware version of the User Bios Extension Area (UBEA). If the UBEA is not programmed, this field contains zero. | |
| 12 | Status | Int8 | - | General Status register containing bit fields as follows. | |
| | | | 7-6 | Indicator state. | |
| | | | | 00 | Indicator state unknown. |
| | | | | 01 | Indicators in Locate Mode. |
| | | | | 10 | Indicators in Mute Mode. |
| | | | | 11 | Indicators in Normal Mode. |
| | | | 5-4 | Universe Address Programming Authority | |
| | | | | 00 | Universe Programming Authority unknown. |
| | | | | 01 | Set by front panel controls. |
| | | | | 10 | Programmed by network. |
| | | | | 11 | Not used. |
| | | | 3 | Not implemented, transmit as zero, receivers do not test. | |
| | | | 2 | 0 = Normal firmware boot (from flash). Nodes that do not support dual boot, clear this field to zero. | |
| | | | | 1 = Booted from ROM. | |
| | | | 1 | 0 = Not capable of Remote Device Management (RDM). | |
| | | | | 1 = Capable of Remote Device Management (RDM). | |
| | | | 0 | 0 = UBEA not present or corrupt | |
| | | | | 1 = UBEA present | |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 13 | EstaMan | Int16 | - | The ESTA manufacturer code. These codes are used to represent equipment manufacturer. They are assigned by ESTA. This field can be interpreted as two ASCII bytes representing the manufacturer initials. |
| 14 | ShortName [18] | Int8 | - | The array represents a null terminated short name for the Node. The Server uses the ArtAddress packet to program this string. Max length is 17 characters plus the null. This is a fixed length field, although the string it contains can be shorter than the field. |
| 15 | LongName [64] | Int8 | - | The array represents a null terminated long name for the Node. The Server uses the ArtAddress packet to program this string. Max length is 63 characters plus the null. This is a fixed length field, although the string it contains can be shorter than the field. |
| 16 | NodeReport [64] | Int8 | - | The array is a textual report of the Node's operating status or operational errors. It is primarily intended for 'engineering' data rather than 'end user' data. The field is formatted as: "#xxxx [yyyy..] zzzzz…"<br>xxxx is a hex status code as defined in Table 3. yyyy is a decimal counter that increments every time the Node sends an ArtPollResponse that is not responding to an ArtPoll.<br>This allows the server to monitor event changes in the Node.<br>zzzz is an English text string defining the status.<br>This is a fixed length field, although the string it contains can be shorter than the field. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 17 | NumPortsH | Int8 | - | The high byte of the word describing the number of input or output ports. The high byte is for future expansion and is currently zero. |
| 18 | NumPorts | Int8 | - | The low byte of the word describing the number of input or output ports. If number of inputs is not equal to number of outputs, the largest value is taken. Zero is an illegal value. The maximum value is 4. |
| 19 | PortTypes [4] | Int8 | - | This array defines the operation and protocol of each channel. (Ether-Lynx example = 0xc0, 0xc0, 0xc0, 0xc0). The array length is fixed, independent of the number of inputs or outputs physically available on the Node. |
| | | | 7 | Set is this channel can output data from the Art-Net Network. |
| | | | 6 | Set if this channel can input onto the Art-NetNetwork. |
| | | | 5-0 | 00000 = DMX512<br>00001 = MIDI<br>00010 = Avab<br>00011 = Colortran CMX<br>00100 = ADB 62.5<br>00101 = Art-Net |
| 20 | GoodInput [4] | Int8 | - | This array defines input status of the node. |
| | | | 7 | Set – Data received. |
| | | | 6 | Set – Channel includes DMX512 test packets. |
| | | | 5 | Set – Channel includes DMX512 SIP's. |
| | | | 4 | Set – Channel includes DMX512 text packets. |
| | | | 3 | Set – Input is disabled. |
| | | | 2 | Set – Receive errors detected. |
| | | | 1-0 | Unused and transmitted as zero. |
| 21 | GoodOutput [4] | Int8 | - | This array defines output status of the node. |
| | | | 7 | Set – Data is being transmitted. |
| | | | 6 | Set – Channel includes DMX512 test packets. |
| | | | 5 | Set – Channel includes DMX512 SIP's. |
| | | | 4 | Set – Channel includes DMX512 text packets. |
| | | | 3 | Set – Output is merging ArtNet data. |
| | | | 2 | Set – DMX output short detected on power up |
| | | | 1 | Set – Merge Mode is LTP. |
| | | | 0 | Unused and transmitted as zero. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 22 | Swin [4] | Int8 | - | This array defines the 8 bit Universe address of the available input channels. In DMX-Hub and Netgate, the high nibble is identical to the data held in the low nibble of Subswitch. The low nibble corresponds to the front panel selector for each channel. |
| 23 | Swout [4] | Int8 | - | This array defines the 8 bit Universe address of the available output channels. In DMX-Hub and Netgate, the high nibble is identical to the data held in the low nibble of Subswitch. The low nibble corresponds to the front panel selector for each channel. |
| 24 | SwVideo | Int8 | - | Set to 00 when video display is showing local data. Set to 01 when video is showing ethernet data. |
| 25 | SwMacro | Int8 | - | If the Node supports macro key inputs, this byte represents the trigger values. The Node is responsible for 'debouncing' inputs. When the ArtPollReply is set to transmit automatically, (TalkToMe Bit 1), the ArtPollReply will be sent on both key down and key up events. However, the Server should not assume that only one bit position has changed.<br>The Macro inputs are used for remote event triggering or cueing.<br>Bit fields are active high. |
| | | | 7 | Set – Macro 8 active. |
| | | | 6 | Set – Macro 7 active. |
| | | | 5 | Set – Macro 6 active. |
| | | | 4 | Set – Macro 5 active. |
| | | | 3 | Set – Macro 4 active. |
| | | | 2 | Set – Macro 3 active. |
| | | | 1 | Set – Macro 2 active. |
| | | | 0 | Set – Macro 1 active. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 26 | SwRemote | Int8 | - | If the Node supports remote trigger inputs, this byte represents the trigger values. The Node is responsible for 'debouncing' inputs. When the ArtPollReply is set to transmit automatically, (TalkToMe Bit 1), the ArtPollReply will be sent on both key down and key up events. However, the Server should not assume that only one bit position has changed.<br>The Remote inputs are used for remote event triggering or cueing.<br>Bit fields are active high. |
| | | | 7 | Set – Remote 8 active. |
| | | | 6 | Set – Remote 7 active. |
| | | | 5 | Set – Remote 6 active. |
| | | | 4 | Set – Remote 5 active. |
| | | | 3 | Set – Remote 4 active. |
| | | | 2 | Set – Remote 3 active. |
| | | | 1 | Set – Remote 2 active. |
| | | | 0 | Set – Remote 1 active. |
| 27 | Spare | Int8 | | Not used, set to zero |
| 28 | Spare | Int8 | | Not used, set to zero |
| 29 | Spare | Int8 | | Not used, set to zero |
| 30 | Style | Int8 | | The Style code defines the equipment style of the device. See Table 4 for current Style codes. |
| 31 | MAC Hi | Int8 | | MAC Address Hi Byte. Set to zero if node cannot supply this information. |
| 32 | MAC | Int8 | | MAC Address |
| 33 | MAC | Int8 | | MAC Address |
| 34 | MAC | Int8 | | MAC Address |
| 35 | MAC | Int8 | | MAC Address |
| 36 | MAC Lo | Int8 | | MAC Address Lo Byte |
| 37 | Filler | 32 x 8 | | Transmit as zero. For future expansion. |

**ArtIpProg:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with ArtIpProgReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Reply with ArtIpProgReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

The ArtIpProg packet allows the IP settings of a Node to be reprogrammed.

The ArtIpProg packet is sent by a Server to the private address of a Node. If the Node supports remote programming of IP address, it will respond with an ArtIpProgReply packet.

In all scenarios, the ArtIpProgReply is sent to the private address of the sender.

| ArtIpProg | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpIpProg<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 7 | Command | Int8 | - | Action this packet as follows: |
| | | | - | Defines the how this packet is processed. If all bits are clear, this is an enquiry only. |
| | | | 7 | Set to enable any programming. |
| | | | 6-4 | Not used, transmit as zero |
| | | | 3 | Set to return all three parameters to default |
| | | | 2 | Program IP Address |
| | | | 1 | Program Subnet Mask |
| | | | 0 | Program Port |
| 8 | Filler4 | Int8 | | Set to zero. Pads data structure for word alignment. |
| 9 | ProgIpHi | Int8 | | IP Address to be programmed into Node if enabled by Command Field |
| 10 | ProgIp2 | Int8 | | |
| 11 | ProgIp1 | Int8 | | |
| 12 | ProgIpLo | Int8 | | |
| 13 | ProgSmHi | Int8 | | Subnet mask to be programmed into Node if enabled by Command Field |
| 14 | ProgSm2 | Int8 | | |
| 15 | ProgSm1 | Int8 | | |
| 16 | ProgSmLo | Int8 | | |
| 17 | ProgPort Hi | Int8 | | PortAddress to be programmed into Node if enabled by Command Field |
| 18 | ProgPort Lo | Int8 | | |
| 19 | Spare1 | Int8 | | Transmit as zero, receivers don't test. |
| 20 | Spare2 | Int8 | | Transmit as zero, receivers don't test. |
| 21 | Spare3 | Int8 | | Transmit as zero, receivers don't test. |
| 22 | Spare4 | Int8 | | Transmit as zero, receivers don't test. |
| 23 | Spare5 | Int8 | | Transmit as zero, receivers don't test. |
| 24 | Spare6 | Int8 | | Transmit as zero, receivers don't test. |
| 25 | Spare7 | Int8 | | Transmit as zero, receivers don't test. |
| 26 | Spare8 | Int8 | | Transmit as zero, receivers don't test. |

**ArtIpProgReply:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Node | Receive | No Action. |
| | Private Transmit | Transmits to specific Server IP address. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action |
| | Private Transmit | Transmits to specific Server IP address. |
| | Broadcast | Not Allowed. |

The ArtIpProgReply packet is issued by a Node in response to an ArtIpProg packet.

Nodes that do not supports remote programming of IP address do not reply to ArtIpProg packets.

In all scenarios, the ArtIpProgReply is sent to the private address of the sender.

| ArtIpProgReply | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpIpProgReply<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Filler3 | Int8 | - | Pad length to match ArtIpProg. |
| 8 | Filler4 | Int8 | | Pad length to match ArtIpProg. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 9 | ProgIpHi | Int8 | | IP Address of Node. |
| 10 | ProgIp2 | Int8 | | |
| 11 | ProgIp1 | Int8 | | |
| 12 | ProgIpLo | Int8 | | |
| 13 | ProgSmHi | Int8 | | Subnet mask of Node. |
| 14 | ProgSm2 | Int8 | | |
| 15 | ProgSm1 | Int8 | | |
| 16 | ProgSmLo | Int8 | | |
| 17 | ProgPort Hi | Int8 | | Port Address of Node. |
| 18 | ProgPort Lo | Int8 | | |
| 19 | Spare1 | Int8 | | Transmit as zero, receivers don't test. |
| 20 | Spare2 | Int8 | | Transmit as zero, receivers don't test. |
| 21 | Spare3 | Int8 | | Transmit as zero, receivers don't test. |
| 22 | Spare4 | Int8 | | Transmit as zero, receivers don't test. |
| 23 | Spare5 | Int8 | | Transmit as zero, receivers don't test. |
| 24 | Spare6 | Int8 | | Transmit as zero, receivers don't test. |
| 25 | Spare7 | Int8 | | Transmit as zero, receivers don't test. |
| 26 | Spare8 | Int8 | | Transmit as zero, receivers don't test. |

**IP Address Override:**

All Art-Net compatible devices provide a temporary override facility that defeats any non-standard IP address programming.

Setting the Sub-Net switch and all available Universe switches to 'F' enables IP override. In IP override mode, the Node ignores all parameters programmed by ArtIpProg commands.

**ArtAddress:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

A Server or monitoring device on the network can reprogram numerous controls of a node remotely. This, for example, would allow the lighting console to re-route DMX512 data at remote locations. This is achieved by sending an ArtAddress packet to the Node's IP address. (The IP address is returned in the ArtPoll packet). The node replies with an ArtPollReply packet.

Fields 7 to 13 contain the data that will be programmed into the node.

| ArtAddress | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpAddress<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 7 | Short Name [18] | Int8 | - | The array represents a null terminated short name for the Node. The Server uses the ArtAddress packet to program this string. Max length is 17 characters plus the null. The Node will ignore this value if the string is null.<br>This is a fixed length field, although the string it contains can be shorter than the field. |
| 8 | Long Name [64] | Int8 | - | The array represents a null terminated long name for the Node. The Server uses the ArtAddress packet to program this string. Max length is 63 characters plus the null. The Node will ignore this value if the string is null.<br>This is a fixed length field, although the string it contains can be shorter than the field. |
| 9 | Swin [4] | Int8 | - | This array defines the low nibble of the Universe address for the available input channels. This corresponds to the front panel selector for each channel. This value is ignored unless bit 7 is high. i.e. to program a switch to value 0x07, send the value as 0x87.<br>Send 0x00 to reset this value to the physical switch setting.<br>Use value 0x7f for no change. |
| 10 | Swout [4] | Int8 | - | This array defines the low nibble of the Universe address for the available output channels. This corresponds to the front panel selector for each channel. This value is ignored unless bit 7 is high. i.e. to program a switch to value 0x07, send the value as 0x87.<br>Send 0x00 to reset this value to the physical switch setting.<br>Use value 0x7f for no change. |
| 11 | SubSwitch | Int8 | - | The low nibble of the Node's Sub-net Address. This is the variable that addresses a Node within Art-Net. This is the DMX-Hub Sub-net switch. This value is ignored unless bit 7 is high. i.e. to program a switch to value 0x07, send the value as 0x87.<br>Send 0x00 to reset this value to the physical switch setting.<br>Use value 0x7f for no change. |

| Field | Name | Size | Bit | Description | | |
|---|---|---|---|---|---|---|
| 12 | SwVideo | Int8 | - | Reserved. | | |
| 13 | Command | Int8 | - | Node configuration commands: | | |
| | | | | Value | Mnemonic | Action |
| | | | | 0x00 | AcNone | No action |
| | | | | 0x01 | AcCancelMerge | If Node is currently in merge mode, cancel merge mode upon receipt of next ArtDmx packet. See discussion of merge mode operation. |
| | | | | 0x02 | AcLedNormal | The front panel indicators of the Node operate normally. |
| | | | | 0x03 | AcLedMute | The front panel indicators of the Node are disabled and switched off. |
| | | | | 0x04 | AcLedLocate | Rapid flashing of the Node's front panel indicators. It is intended as an outlet locator for large installations. |
| | | | | 0x05 | AcResetRxFlags | Resets the Node's Sip, Text, Test and data error flags. If an output short is being flagged, forces the test to re-run. |

| Field | Name | Size | Bit | Description | | |
|---|---|---|---|---|---|---|
| 13 cont/ | Command | Int8 | - | Node configuration commands: Note that Ltp / Htp settings should be retained by the node during power cycling. | | |
| | | | | Value | Mnemonic | Action |
| | | | | 0x10 | AcMergeLtp0 | Set DMX Port 0 to Merge in LTP mode. |
| | | | | 0x11 | AcMergeLtp1 | Set DMX Port 1 to Merge in LTP mode. |
| | | | | 0x12 | AcMergeLtp2 | Set DMX Port 2 to Merge in LTP mode. |
| | | | | 0x13 | AcMergeLtp3 | Set DMX Port 3 to Merge in LTP mode. |
| | | | | 0x50 | AcMergeHtp0 | Set DMX Port 0 to Merge in HTP (default) mode. |
| | | | | 0x51 | AcMergeHtp1 | Set DMX Port 1 to Merge in HTP (default) mode. |
| | | | | 0x52 | AcMergeHtp2 | Set DMX Port 2 to Merge in HTP (default) mode. |
| | | | | 0x53 | AcMergeHtp3 | Set DMX Port 3 to Merge in HTP (default) mode. |
| | | | | 0x90 | AcClearOp0 | Clear DMX Output buffer for Port 0 |
| | | | | 0x91 | AcClearOp1 | Clear DMX Output buffer for Port 1 |
| | | | | 0x92 | AcClearOp2 | Clear DMX Output buffer for Port 2 |
| | | | | 0x93 | AcClearOp3 | Clear DMX Output buffer for Port 3 |

**ArtDmx:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Server broadcasts its real time data (Console output). |
| Node | Receive | No Action |
| | Private Transmit | Not Allowed. |
| | Broadcast | Node broadcasts its real time data. |
| Media Server | Receive | No Action |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

ArtDmx is the data packet used to transfer DMX512 data. The format is identical for Node to Server, Node to Node and Server to Node.

The Node will always transmit ArtDmx on the broadcast address. This ensures that peer to peer operation can always occur, even when the network contains a server.

Consider for example, an installation with a single lighting console (Server) and two Ether-Lynx (Nodes). The Server would poll the Nodes at power on to identify their IP addresses. From this point onwards the Server can then conduct private communication with each Node. However, in order to use the Nodes to transfer data from Node to Node or to add a second Server to the network, the Nodes must broadcast the DMX data.

The Data is output through the DMX O/P port corresponding to the Universe setting. In the absence of received ArtDmx packets, each DMX O/P port re-transmits the same frame continuously.

The first complete DMX frame received at each input port is placed in an ArtDmx packet as above and broadcast as an ArtDmx packet containing the relevant Universe parameter. Each subsequent DMX frame containing **new data** (different length or different contents) is placed is also broadcast as an ArtDmx packet.

Nodes do not broadcast ArtDmx for DMX512 inputs that have not received data since power on.

However, an input that is active but not changing, will re-broadcast the last valid ArtDmx packet at approximately 4-second intervals.

A DMX input that fails, will not continue to send ArtDmx data.

| ArtDmx | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpOutput<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Sequence | Int8 | - | The sequence number is used to ensure that ArtDmx packets are used in the correct order. When Art-Net is carried over a medium such as the Internet, it is possible that ArtDmx packets will reach the receiver out of order.<br>This field is incremented in the range 0x01 to 0xff to allow the receiving node to resequence packets.<br>The Sequence field is set to 0x00 to disable this feature. |
| 6 | Physical | Int8 | - | The physical input port from which DMX512 data was input. This field is for information only. Use Universe for data routing. |
| 7 | Universe | Int16 | - | The high byte is currently set to zero. The low byte is the address of this Universe of data. In DMX-Hub, the high nibble is the Sub-net switch and the low Nibble is the Universe address switch.<br>Transmitted low byte first. |
| 8 | LengthHi | Int8 | - | The length of the DMX512 data array. This value should be an even number in the range 2 – 512.<br>It represents the number of DMX512 channels received.<br>High Byte. |
| 9 | Length | Int8 | - | The length of the DMX512 data array. This value should be an even number in the range 2 – 512.<br>It represents the number of DMX512 channels received.<br>Low Byte. |
| 10 | Data [Length] | Int8 | - | An array of DMX512 lighting data. |

**Refresh Rate:**

The ArtDmx packet is intended to transfer DMX512 data. For this reason, the ArtDmx packet for a specific IP Address should not be transmitted at a repeat rate faster than the maximum repeat rate of a DMX packet containing 512 data slots.

**Data Merging:**

The Art-Net protocol allows multiple nodes or servers to transmit ArtDmx data to the same universe.

A node can detect this situation by comparing the IP addresses of received ArtDmx packets. If ArtDmx packets addressed to the same Universe are received from different IP addresses, a potential conflict exists.

The Node can legitimately handle this situation using one of two methods:

1. Consider this to be an error condition and await user intervention.
2. Automatically merge the data.

Nodes should document the approach that is implemented in the product user guide. The Merge option is preferred as it provides a higher level of functionality.

Merge is implemented in either LTP or HTP mode as specified by the ArtAddress packet.

Merge mode is implemented as follows:

1. If ArtDmx is received from differing IP addresses, the data is HTP merged to the DMX output. In this situation, *ArtPollReply-GoodOutput-Bit3* is set. If *Art-Poll-TalkToMe Bit 1* is set, an ArtPollReply should be transmitted when merging commences.
2. Exit from Merge mode is handled as follows:
   - If *ArtAddress AcCancelMerge* is received, the Next ArtDmx message received ends Merge mode. The Node then discards any ArtDmx packets received from an IP address that does not match the IP address of the ArtDmx packet that terminated Merge mode.
   - If either (but not both) sources of ArtDmx stop, the failed source is held in the merge buffer for 10 seconds. If, during the 10 second timeout, the failed source returns, Merge mode continues. If the failed source does not recover, at the end of the timeout period, the Node exits Merge mode.
   - If both sources of ArtDmx fail, the output holds the last merge result.

Merging is limited to two sources, any additional sources will be ignored by the Node.

The Merge implementation allows for the following two key modes of operation.

1. Combined Control: Two Servers (Consoles) can operate on a network and merge data to multiple Nodes.
2. Backup: One Server (Console) can monitor the network for a failure of the primary Server. If a failure occurs, it can use the *ArtAddress AcCancelMerge* command to take instant control of the network.

When a node provides multiple DMX512 inputs, it is the responsibility of the Node to handle merging of data. This is because the Node will have only one IP address. If this were not handled at the Node, ArtDmx packets with identical IP addresses and identical universe numbers, but conflicting level data would be transmitted to the network.

**ArtInput:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

A Server or monitoring device on the network can enable or disable individual DMX512 inputs on any of the network nodes. This allows the Server to directly control network traffic and ensures that unused inputs are disabled and therefore not wasting bandwidth.

All nodes power on with all inputs enabled.

Caution should be exercised when implementing this function in the server. Keep in mind that some network traffic may be operating on a node to node basis.

| ArtInput | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpInput<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | NumPorts H | Int8 | - | The high byte of the word describing the number of input or output ports. The high byte is for future expansion and is currently zero. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 8 | NumPorts | Int8 | - | The low byte of the word describing the number of input or output ports. If number of inputs is not equal to number of outputs, the largest value is taken. Zero is an illegal value. The maximum value is 4. |
| 9 | Input [4] | Int8 | - | This array defines input disable status of each channel. (DMX-Hub example = 0x01, 0x00, 0x01, 0x00 to disable first and third inputs) |
| | | | 7-1 | Not currently used |
| | | | 0 | Set to disable this input. |

**Firmware and UBEA upgrades:**

This section defines the packets used to send firmware revisions to a node. In all instances, communication is private. Under no circumstances should the broadcast address be used.

The transaction involves the server sending multiple ArtFirmwareMaster packets to a Node's IP address. Each packet is acknowledged by the Node with an ArtFirmwareReply.

The server allows a 20 second maximum delay for reception of the ArtFirmwareReply. If the reply is not received in this time, the server aborts the transaction. The large time period is to allow for Nodes that are writing directly to slow non-volatile memory.

The Node allows a 20 second delay between sending an ArtFirmwareReply and receipt of the next consecutive ArtFirmwareMaster. If the next consecutive block is not received within this time, the Node aborts the transaction. In this instance the Node returns to it's previous operating system and sets ArtPollReply->Status and ArtPollReply ->NodeReport accordingly.

The firmware update file contains a header that defines the Node OEM values that are valid for this update. The Server must check this value before sending to a Node. The Node also checks this data on receipt of the first packet. If the Node receives a packet with an invalid code, it sends an error response.

The UBEA is the User Bios Expansion Area. This is a limited firmware upload mechanism that allows third party firmware extensions to be added to a Node.

Manufacturers who implement this feature must document the software interface requirements.

**ArtFirmwareMaster**:

| **Implementation** | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with OpFirmwareReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Reply with OpFirmwareReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

| ArtFirmwareMaster | | | | | | |
|---|---|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** | | |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 | | |
| 2 | OpCode | Int16 | - | OpFirmwareMaster.<br>Transmitted low byte first. | | |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. | | |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 | | |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. | | |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. | | |
| 7 | Type | Int8 | - | Defines the packet contents as follows: | | |
| | | | | Value | Mnemonic | Function |
| | | | | 0x00 | FirmFirst | The first packet of a firmware upload. |
| | | | | 0x01 | FirmCont | A consecutive continuation packet of a firmware upload. |
| | | | | 0x02 | FirmLast | The last packet of a firmware upload. |
| | | | | 0x03 | UbeaFirst | The first packet of a UBEA upload. |
| | | | | 0x04 | UbeaCont | A consecutive continuation packet of a UBEA upload. |
| | | | | 0x05 | UbeaLast | The last packet of a UBEA upload. |
| 8 | BlockId | Int8 | - | Counts the consecutive blocks of firmware upload. Starting at 0x00 for the FirmFirst or UbeaFirst packet. | | |
| 9 | Firmware Length3 | Int8 | - | The total number of words (Int32) in the firmware upload plus the firmware header size. Eg a 32K word upload plus 530 words of header information == 0x00008212. This value is also the file size (in words) of the file to be uploaded. | | |
| 10 | Firmware Length2 | Int8 | - | | | |
| 11 | Firmware Length1 | Int8 | - | | | |
| 12 | Firmware Length0 | Int8 | - | LSB | | |
| 13 | Spare[20] | Int8 | - | Server sets to zero, Node does not test. | | |
| 14 | Data[512] | Int16 | - | This array contains the firmware or UBEA data block. The order is hi byte first. The interpretation of this data is manufacturer specific. | | |

**ArtFirmwareReply**:

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | Send next OpFirmwareMaster. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Node | Receive | No Action. |
| | Private Transmit | Node transmits to a specific Server IP address. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action. |
| | Private Transmit | Node transmits to a specific Server IP address. |
| | Broadcast | Not Allowed. |

This packet is sent by the Node to the Server in acknowledgement of each OpFirmwareMaster packet.

| ArtFirmwareReply | | | | | | |
|---|---|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** | | |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 | | |
| 2 | OpCode | Int16 | - | OpFirmwareReply.<br>Transmitted low byte first. | | |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. | | |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 | | |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. | | |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. | | |
| 7 | Type | Int8 | - | Defines the packet contents as follows. Codes are used for both firmware and UBEA. | | |
| | | | | Value | Mnemonic | Function |
| | | | | 0x00 | FirmBlock Good | Last packet received successfully. |
| | | | | 0x01 | FirmAll Good | All firmware received successfully. |
| | | | | 0xff | FirmFail | Firmware upload failed. (All error conditions). |
| 8 | Spare[21] | Int8 | - | Node sets to zero, Server does not test. | | |

**Firmware File Format**:

All firmware and UBEA upload files should be of the following format.

The firmware file extension is .alf.
The UBEA file extension is .alu.

| Byte | Name | Description |
|---|---|---|
| 1 | ChecksumHi | This is a 16 bit, one's compliment checksum of the firmware data area. |
| 2 | ChecksumLo | LSB of above |
| 3 | VersInfoHi | High byte of Node's firmware revision number. The Server should only use this field to decide if a firmware update should proceed. The convention is that a higher number is a more recent release of firmware. |
| 4 | VersInfoLo | LSB of above |
| 5-34 | UserName | 30 byte field of user name information. This information is not checked by the Node. It is purely for display by the Server. It should contain a human readable description of file and version number. Whilst this is a fixed length field, it must contain a null termination. |
| 35-546 | Oem[256] | An array of 256 words. Each word is hi byte first and represents an Oem code for which this file is valid. Unused entries must be filled with 0xffff. |
| 547-1056 | Spare[254] | An array of 254 words. Currently unused and should be set to zero. |
| 1057 | Length3 | The total length in words of the firmware information following this field. |
| 1058 | Length2 | |
| 1059 | Length1 | |
| 1060 | Length0 | LSB |
| 1061 | Data[] | The firmware data as an array of 16 bit values ordered hi byte first. The actual data is manufacturer specific. |

## RDM Support:

This section defines the packet structure used to gate the Remote Device Management (RDM) protocol across Art-Net. It is assumed that the reader is familiar with the RDM document.

Art-Net devices support RDM as follows:

1. All RDM discovery commands are proxied; Art-Net devices hold local RDM device lists and conduct their own discovery.
2. All RDM Get / Set commands are non-proxied; they are passed to end devices for response.

This document defines the following terms:

**Input Gateway:** A device that inputs DMX512 onto the Art-Net network (e.g. Up-Lynx).

**Output Gateway:** A device that outputs DMX512 from the Art-Net network (e.g. Down-Lynx)

**Table of Devices (TOD):** The list of RDM devices maintained by both Input and Output Gateways.

## RDM Discovery

### Output Gateway Operation

Output Gateways perform RDM discovery independent of network operation. This includes full discovery upon power-on and incremental discovery as a background task. The Output Gateway informs the network about its TOD as follows:

1. Upon receipt of an ArtTodRequest packet, the Output Gateway broadcasts an ArtTodData packet containing the entire TOD. All Input Gateways parse the ArtTodData packets. If the Sub-Net and Universe fields match, the Input Gateway adds the TOD contents to their own internal TOD. This allows Input Gateways to respond to any RDM discovery commands they receive.
2. Upon completion of initial RDM discovery, Output Gateways broadcast their TOD in an ArtTodData packet.
3. When an RDM device is added to or removed from the Output Gateway's TOD (during incremental discovery), an ArtTodData packet is broadcast automatically.

**Input Gateway Operation**

Input Gateways generate a TOD by monitoring Art-Net traffic. The TOD is then used to reply to RDM discovery commands by proxy. Operation is as follows:

1. Upon power-on, Input Gateways broadcast an ArtTodRequest packet.
2. The network is monitored for ArtTodData packets. If the Sub-Net and Universe fields match, the Input Gateway adds the TOD contents to its own internal TOD. This allows Input Gateways to respond to any RDM discovery commands they receive.
3. Input Gateways do not transmit any RDM discovery messages to the network.

**Server Operation**

Servers emulate the operation of Input Gateways.

**ArtTodRequest**:

This packet is used to request the Table of RDM Devices (TOD). A Node receiving this packet must not interpret it as forcing full discovery. Full discovery is only initiated at power on or when an ArtTodControl.AtcFlush is received.

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Server broadcasts to all nodes. |
| Node Output Gateway | Receive | Reply with ArtTodData. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Node Input Gateway | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Input Gateway broadcasts to all nodes. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

| ArtTodRequest | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpTodRequest.<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 7 | Spare1 | Int8 | - | Transmit as zero, receivers don't test. |
| 8 | Spare2 | Int8 | - | Transmit as zero, receivers don't test. |
| 9 | Spare3 | Int8 | - | Transmit as zero, receivers don't test. |
| 10 | Spare4 | Int8 | - | Transmit as zero, receivers don't test. |
| 11 | Spare5 | Int8 | - | Transmit as zero, receivers don't test. |
| 12 | Spare6 | Int8 | - | Transmit as zero, receivers don't test. |
| 13 | Spare7 | Int8 | - | Transmit as zero, receivers don't test. |
| 14 | Spare8 | Int8 | - | Transmit as zero, receivers don't test. |
| 15 | Command | Int8 | - | |
| | | | | Value / Mnemonic / Function |
| | | | | 0x00 / TodFull / Send the entire TOD. |
| 16 | AdCount | Int8 | - | The array size of the Address field. Max value is 32. |
| 17 | Address [AdCount] | Int8 | - | This array defines the 8 bit Universe address of the Output Gateway nodes that must respond to this packet. The high nibble is the Sub-Net switch. The low nibble corresponds to the Universe. |

**ArtTodData**:

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Node Output Gateway | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Output Gateway always broadcasts this packet. |
| Node Input Gateway | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

| ArtTodData | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpTodData.<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Port | Int8 | - | Physical Port. Range 1-4. |
| 7 | Spare1 | Int8 | - | Transmit as zero, receivers don't test. |
| 8 | Spare2 | Int8 | - | Transmit as zero, receivers don't test. |
| 9 | Spare3 | Int8 | - | Transmit as zero, receivers don't test. |
| 10 | Spare4 | Int8 | - | Transmit as zero, receivers don't test. |
| 11 | Spare5 | Int8 | - | Transmit as zero, receivers don't test. |
| 12 | Spare6 | Int8 | - | Transmit as zero, receivers don't test. |
| 13 | Spare7 | Int8 | - | Transmit as zero, receivers don't test. |
| 14 | Spare8 | Int8 | - | Transmit as zero, receivers don't test. |

| Field | Name | Size | Bit | Description | | |
|---|---|---|---|---|---|---|
| 15 | Command Response | Int8 | - | Defines the packet contents as follows. The TodFull command should be used with full RDM discovery. The TodAdd and TodSubtract commands should be used with incremental discovery. | | |
| | | | | Value | Mnemonic | Function |
| | | | | 0x00 | TodFull | The packet contains the entire TOD or is the first packet in a sequence of packets that contains the entire TOD. |
| | | | | 0xff | TodNak | The TOD is not available. |
| 16 | Address | Int8 | - | The 8 bit Universe address of the Output Gateway DMX Port that generated this packet. The high nibble is the Sub-Net switch. The low nibble corresponds to the Universe. | | |
| 17 | UidTotalHi | Int8 | - | The total number of RDM devices discovered by this Universe. | | |
| 18 | UidTotalLo | Int8 | - | | | |
| 19 | BlockCount | Int8 | - | The index number of this packet. When UidTotal exceeds 200, multiple ArtTodData packets are used. BlockCount is set to zero for the first packet, and incremented for each subsequent packet containing blocks of TOD information. | | |
| 20 | UidCount | Int8 | - | The number of UIDs encoded in this packet. This is the index of the following array. | | |
| 21 | TOD [UidCount] | 48 bit | - | An array of RDM UID. | | |

**ArtTodControl**:

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Server broadcasts to all nodes. |
| Node Output Gateway | Receive | Reply with ArtTodData. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Node Input Gateway | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Input Gateway broadcasts to all nodes. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

The ArtTodControl packet is used to send RDM control parameters over Art-Net. The response is ArtTodData.

| ArtTodControl | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpTodControl.<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Spare1 | Int8 | - | Transmit as zero, receivers don't test. |
| 8 | Spare2 | Int8 | - | Transmit as zero, receivers don't test. |
| 9 | Spare3 | Int8 | - | Transmit as zero, receivers don't test. |
| 10 | Spare4 | Int8 | - | Transmit as zero, receivers don't test. |
| 11 | Spare5 | Int8 | - | Transmit as zero, receivers don't test. |
| 12 | Spare6 | Int8 | - | Transmit as zero, receivers don't test. |
| 13 | Spare7 | Int8 | - | Transmit as zero, receivers don't test. |
| 14 | Spare8 | Int8 | - | Transmit as zero, receivers don't test. |

| Field | Name | Size | Bit | Description | | |
|---|---|---|---|---|---|---|
| 15 | Command | Int8 | - | Defines the packet action. | | |
| | | | | Value | Mnemonic | Function |
| | | | | 0x00 | AtcNone | No action. |
| | | | | 0x01 | AtcFlush | The node flushes it's TOD and instigates full discovery. |
| 16 | Address | Int8 | - | The 8 bit Universe address of the DMX Port that should action this command. | | |

**ArtRdm**:

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Server broadcasts to all nodes. |
| Node Output Gateway | Receive | No Action |
| | Private Transmit | Not Allowed. |
| | Broadcast | Node broadcasts to all. |
| Node Input Gateway | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Node broadcasts to all. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

The ArtRdm packet is used to transport all non-discovery RDM messages over Art-Net.

| ArtRdm | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpRdm.<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Spare1 | Int8 | - | Transmit as zero, receivers don't test. |
| 8 | Spare2 | Int8 | - | Transmit as zero, receivers don't test. |
| 9 | Spare3 | Int8 | - | Transmit as zero, receivers don't test. |
| 10 | Spare4 | Int8 | - | Transmit as zero, receivers don't test. |
| 11 | Spare5 | Int8 | - | Transmit as zero, receivers don't test. |
| 12 | Spare6 | Int8 | - | Transmit as zero, receivers don't test. |
| 13 | Spare7 | Int8 | - | Transmit as zero, receivers don't test. |
| 14 | Spare8 | Int8 | - | Transmit as zero, receivers don't test. |

| Field | Name | Size | Bit | Description | | |
|---|---|---|---|---|---|---|
| 15 | Command | Int8 | - | Defines the packet action. | | |
| | | | | Value | Mnemonic | Function |
| | | | | 0x00 | ArProcess | Process RDM Packet. |
| 16 | Address | Int8 | - | The 8 bit Universe address of the DMX Port that should action this command. | | |
| 17 | RdmPacket | Int8 [Vari] | - | The RDM data packet including the StartCode. | | |

**Display of status:**

Most Art-Net compliant equipment will provide some level of status indication. The following format is suggested:

| Name | Mnemonic | Colour | Function | |
|---|---|---|---|---|
| Power | Pow | Red | Normally on, flashes if fault detected. | |
| Communication | Com | Amber | On if any Art-Net packets detected on network, timeout after 6 seconds. | |
| DMX512 | DMX x | Green | DMX Input | On if good DMX received, flashing if errors detected. Alternative Start Codes are not errors! |
| | | | DMX Output | On if receiving ArtDmx for this output. Timeout after 6 seconds. |

**Data Integrity:**

Art-Net receivers should check two items:

- Compare the ID[8] field
- Check packet length is 'greater than or equal to' the specification length. Do not make an 'equal to' test, as packet lengths may increase in future revisions to the specification.

## Extended Features:

Nodes that implement the extended video features use the following data structures. These features are primarily intended for implementation of remote video repeater monitors for lighting consoles. They are not part of the Media Extensions.

## ArtVideoSetup:

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

A Server or monitoring device on the network can remotely configure the video output of a node that supports video output. All nodes supporting this feature should power on with network video disabled. Due to buffer size constraints, a maximum of 63 fonts may be programmed in one packet. This means that it is necessary to send four separate packets to redefine the entire font map.

| ArtVideoSetup | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpVideoSetup<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Filler3 | Int8 | - | Reserved. Transmit as zero. |
| 8 | Filler4 | Int8 | - | Reserved. Transmit as zero. |

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 9 | Control | Int8 | - | |
| | | | 7-1 | Not currently used |
| | | | 0 | Set to enable network video. Clear to return video control to node's local control. |
| 10 | FontHeight | Int8 | - | Set the number of scan lines to be used by each character. This also indirectly sets the total number of text lines that are visible. Supported values are 8-16 inclusive. |
| 11 | FirstFont | Int8 | - | The number (0-255) of the first font to be reprogrammed. |
| 12 | LastFont | Int8 | - | The total number (1-63) of fonts to be reprogrammed. Set to zero to inhibit font reprogramming. |
| 13 | WinFont Name[64] | Int8 | - | The name of the nearest matching Windows font. This is used by network data monitors to improve the readability of emulated video displays. The string is null terminated. Set to a null string to disable this feature. |
| 14 | FontData[] | Int8 | - | Array of font data. The size is FontHeight*FontRange. The data structure is identical to that required by PC Bios call used for font reprogramming. |

**ArtVideoPalette:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

A Server or monitoring device on the network can remotely configure the video output colour palette of a node that supports video output. All nodes supporting this feature should power on with network video disabled.

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpVideoPalette<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | ColourRed [17] | Int8 | - | Each array entry is in the range 0-0x3f. This array defines the amount of red in each of the 17 available colours. |
| 8 | ColourGreen[17] | Int8 | - | Each array entry is in the range 0-0x3f. This array defines the amount of green in each of the 17 available colours. |
| 9 | ColourBlue [17] | Int8 | - | Each array entry is in the range 0-0x3f. This array defines the amount of blue in each of the 17 available colours. |

**ArtVideoData:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server transmits to a specific node IP address. |
| | Broadcast | Not Allowed. |
| Node | Receive | Reply with ArtPollReply. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |

This packet is used to send test video data to the node. Start position and size parameters are included to allow partial screen updates.
Due to buffer size constraints, a maximum of one quarter screen may be programmed in one packet. This means that it is necessary to send four separate packets to refresh the entire screen.

| Field | Name | Size | Bit | Description |
|---|---|---|---|---|
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpVideoData. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | PosX | Int8 | - | The start column of the text update. Range 0 – 79. |
| 8 | PosY | Int8 | - | The start row of the text update. Range 0 – 49. |
| 9 | LenX | Int8 | - | The number of columns to update. Value = (End Column) - PosX. |
| 10 | LenY | Int8 | - | The number of rows to update. Value = (End Row) - PosY. |
| 11 | Data[] | Int16 | - | Array of text starting top left of text area and working left to right, then top to bottom. High byte is ASCII character. Low byte is VGA attribute byte. |

**Media Extensions Art-Net mx**

The Media Extensions packets are used to control the streaming of data from media servers.

This is primarily aimed at the routing of video data over Art-Net for post processing by a lighting controller or pseudo video wall controller.

The Server uses ArtPoll to detect the existence of Media Servers on the network. It then interrogates the Media Server, using ArtMediaControl, to find it's capabilities.

The Server is responsible for sending patch information to the Media Server using the ArtMediaPatch packet. This consists of a list of virtual coordinates that define the video pixel positions that the Server requires to be streamed. In most instances, these virtual coordinates will be scaled to a video image by the Media Server. Other implementations are also possible, for example, by relating the coordinates to positions on an audio spectrum analyser.

The Server can send virtual patch information at any time, it is not limited to setup and configuration. This allows the Server to perform split screen effects in real time. The Server also sends 'transport' commands for start, stop etc using ArtMediaControl.

The Media Server is responsible for streaming the required media data using the ArtMedia packet. Note that this packet is always Unicast. This allows the use of a 100/10BaseT Switch to segment the Server area of the network from the Node area.

**ArtMedia:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | Use data as required. No response required |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Node | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action. |
| | Private Transmit | Media Server streams data in Unicast. |
| | Broadcast | Not Allowed. |

The ArtMedia packet is used to stream visual information on the network. The primary purpose is to allow a Media Server to send data to a Lighting Server that represents the requested RGB value for a video pixel. The coordinates of each video pixel are set using the ArtMediaPatch packet.

The ArtMedia packet contains packed, run length encoded data. The Media Server only transmits this packet when a change of source data occurs.

Multiple packets may be sent when the number of pixel changes exceeds the maximum that can be carried by in a single packet.

Implementers are encouraged to make the maximum use of the global commands and run length encoding in order to conserve bandwidth.

| ArtMedia | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpMedia<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Filler3 | Int8 | - | Reserved. Transmit as zero. |
| 8 | Filler4 | Int8 | - | Reserved. Transmit as zero. |
| 9 | Stream | Int8 | - | Video stream number. Each scream is allocated a patch. |
| 10 | Command | Int8 | - | See Table 5 |

| | | | | |
|---|---|---|---|---|
| 11 | ComData1 | Int8 | - | Red |
| 12 | ComData2 | Int8 | - | Green |
| 13 | ComData3 | Int8 | - | Blue |
| 14 | Pack[] | 6xInt8 | - | Variable length array as defined below. At least one entry exists as Pack[]->LampHi==0xff terminates the list. Max number of entries is 240. |

| **ArtMedia->Pack** | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | LampHi | Int8 | - | The Lamp Index defined in the ArtMediaPatch table. Set to 0xff to define end of list. |
| 2 | LampLo | Int8 | - | Low byte of above. |
| 3 | Count | Int8 | - | Number of consecutive triplets. 1 = Set this Lamp number to red, green, blue value. 2 = Set this Lamp and also next consecutive Lamp Number to red, green, blue value. Zero value not allowed. |
| 4 | Red | Int8 | - | Red data value. |
| 5 | Green | Int8 | - | Green data value. |
| 6 | Blue | Int8 | - | Blue data value. |

**Table 5 – Command Codes for ArtMedia:**

These codes are used in the ArtMedia packet to provide global commands.

| **Code** | **Mnemonic** | **Description** |
|---|---|---|
| 0x00 | MedComStd | No global commands, parse Pack[] for data. |
| 0x01 | MedComLev | Set all pixels to RGB combination in ComData1-3. Then parse Pack[] as normal. |

**ArtMediaPatch:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action |
| | Private Transmit | Server sends multiple packets to Media Server, so defining pixel list. |
| | Broadcast | Not Allowed. |
| Node | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Parse data and add to list. Respond with ArtMediaControlReply |
| | Private Transmit | Not Allowed |
| | Broadcast | Not Allowed. |

The ArtMediaPatch packet is used to configure the MediaServer. It contains a packed list of virtual screen coordinates. This list defines the position of the pixel that will be streamed using the ArtMedia packet.

The ArtMediaPacket can be sent at any time in order to implement real time effects such as zoom.

Multiple packets may be sent when the number of pixel coordinates exceeds the maximum that can be carried by in a single packet.

Implementers are encouraged to make the maximum use of the global commands and run length encoding in order to conserve bandwidth.

| ArtMediaPatch | | | |
|---|---|---|---|
| **Field** | **Name** | **Size** | **Description** |
| 1 | ID[8] | Int8 | Array of 8 characters, the final character is a null termination. Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | OpMediaPatch Transmitted low byte first. |
| 3 | ProtVerH | Int8 | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | Pad length to match ArtPoll. |
| 7 | Filler3 | Int8 | Reserved. Transmit as zero. |
| 8 | Filler4 | Int8 | Reserved. Transmit as zero. |
| 9 | Stream | Int8 | Video stream number. Each scream is allocated a patch. |
| 10 | PatchCommand | Int8 | See Table 6 |

| | | | |
|---|---|---|---|
| 11 | VirtualDeltaXHi | Int8 | Maximum width used in Coord[] |
| 12 | VirtualDeltaXLo | Int8 | Low byte of above |
| 13 | VirtualDeltaYHi | Int8 | Maximum height used in Coord[] |
| 14 | VirtualDeltaYLo | Int8 | Low byte of above |
| 15 | CoordCountHi | Int8 | Total number of coordinates in list. This is not necessarily the array size of Coord[] as multiple packets can be used to build the list. The MediaServer uses this value to allocate the memory storage. If this value changes between packets, the MediaServer can assume a soft reset and clear the list. |
| 16 | CoordCountLo | Int8 | Low byte of above |
| 17 | Aperture | Int8 | The equation used to generate the average pixel colour. |
| 18 | Diameter | Int8 | The diameter of the circle or rectangle around the pixel that is used to average the colour value. |
| 19 | Coord[] | 6xInt8 | Variable length array as defined below. At least one entry exists as Coord[]->LampHi==0xff terminates the list.<br>The Index of this array is used by ArtMedia->LampHi/Lo.<br>Max number of entries is 240. |

| **ArtMedia->Coord** | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | LampHi | Int8 | - | The Lamp Index defined in the ArtMedia table. |
| 2 | LampLo | Int16 | - | Low byte of above. |
| 3 | XHi | Int8 | - | The X ordinate of this pixel. |
| 4 | XLo | Int8 | - | Low byte of above |
| 5 | Yhi | Int8 | - | The Y ordinate of this pixel. |
| 6 | Ylo | Int8 | - | Low byte of above |

**Table 6 – Command Codes for ArtMediaPatch:**

These codes are used in the ArtMediaPatch packet to provide global commands.

| **Code** | **Mnemonic** | **Description** |
|---|---|---|
| 0x00 | MedPatchStd | No global commands, parse Coord[] for data. |
| 0x01 | MedPatchRes | Clear entire coordinate list. |

**ArtMediaControl:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Server sends packet to Media Server to control video and or request information. |
| | Broadcast | Not Allowed. |
| Node | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | Parse data and add to list. Respond with ArtMediaControlReply |
| | Private Transmit | Not Allowed |
| | Broadcast | Not Allowed. |

The ArtMediaControl packet is used to remote control the Media Server and also retrieve information from the Media Server.

| ArtMediaControl | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpMediaControl<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Filler3 | Int8 | - | Reserved. Transmit as zero. |
| 8 | Filler4 | Int8 | - | Reserved. Transmit as zero. |
| 9 | Stream | Int8 | - | Video stream number. Each scream is allocated a patch. |
| 10 | CtrlCommand | Int8 | - | See Table 7 |
| 11 | ComData1 | Int8 | - | Data depends on content of above field |
| 12 | ComData2 | Int8 | - | " |
| 13 | ComData3 | Int8 | - | " |
| 14 | ComData4 | Int8 | - | " |
| 15 | ComData5 | Int8 | - | " |
| 16 | ComData6 | Int8 | - | " |

**Table 7 – CtrlCommand Codes for ArtMediaControl:**

These codes are used in the ArtMedia packet to provide global commands.

| Code | Mnemonic | Description | |
|---|---|---|---|
| 0x00 | MccNone | There is no explicit command. | |
| 0x01 | MccSelectClip | Select the active Clip. The Clip number meaning is manufacturer specific. It can be a video file for playback or an effects macro to be applied to live video. Rewind media and stop. | |
| | | ComData1 | High byte of clip number. |
| | | ComData2 | Low byte of clip number. |
| | | ComData3 | 1 = Stream one frame. 0 = quiet change. |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |
| 0x02 | MccStop | Stop streaming. Freeze live image. | |
| | | ComData1 | 1 = Rewind to start. 0 = hold position. |
| | | ComData2 | =0 |
| | | ComData3 | =0 |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |
| 0x03 | MccPlay | Start streaming from current position in video. If already playing, ignore. If frozen, thaw. | |
| | | ComData1 | Direction. 1 = Rev. 0 = Fwd. |
| | | ComData2 | =0 |
| | | ComData3 | =0 |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |
| 0x04 | MccGoto | Stop playback. Search to specified position. Stream one frame. If position out of range, goto end of media. | |
| | | ComData1 | = Days (0-255) or FrameNumberHi |
| | | ComData2 | = Hours (0-23) or FrameNumber3 |
| | | ComData3 | = Minutes (0-59) or FrameNumber2 |
| | | ComData4 | = Seconds (0-59) or FrameNumber1 |
| | | ComData5 | = Frames or FrameNumberLo |
| | | ComData6 | Time code mode:<br>0 = frame number<br>1 = 24 fps (Film) timecode<br>2 = 25 fps (EBU) timecode<br>3 = 29.97 fps (DF) timecode<br>4 = 30 fps (SMPTE) timecode |

| | | | |
|---|---|---|---|
| 0x05 | MccTranScan | Start playback fwd from current position dropping every 'X' frames. Alternatively, freeze video and stream one frame every 'X' video frames. | |
| | | ComData1 | Direction. 1 = Rev. 0 = Fwd. |
| | | ComData2 | Number of frames to drop. |
| | | ComData3 | =0 |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |
| 0x06 | MccGetClip Name | Send reply or multiple replies containing text name of clip. See Table 8 for data format. | |
| | | ComData1 | ClipNumberHi |
| | | ComData2 | ClipNumberLo |
| | | ComData3 | Multiple:<br>0 = send name of clip defined above.<br>1 = send names of all clips available in this stream.<br>2 = send names of all clips for all streams. |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |
| 0x07 | MccGetStream Name | Send reply or multiple replies containing text name of stream. See Table 8 for data format. | |
| | | ComData1 | Multiple:<br>0 = send name of this stream.<br>1 = send names of all streams. |
| | | ComData2 | =0 |
| | | ComData3 | =0 |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |
| 0x07 | MccGetModel Name | Send reply containing text name of model of Media Server. This should be human readable. See Table 8 for data format. | |
| | | ComData1 | Multiple:<br>0 = send name of this stream.<br>1 = send names of all streams. |
| | | ComData2 | =0 |
| | | ComData3 | =0 |
| | | ComData4 | =0 |
| | | ComData5 | =0 |
| | | ComData6 | =0 |

**ArtMediaControlReply:**

| Implementation | | |
|---|---|---|
| **Entity** | **Direction** | **Action** |
| Server | Receive | No Action. |
| | Private Transmit | Not Allowed |
| | Broadcast | Not Allowed. |
| Node | Receive | No Action. |
| | Private Transmit | Not Allowed. |
| | Broadcast | Not Allowed. |
| Media Server | Receive | No Action |
| | Private Transmit | Server sends packet to Media Server to control video and or request information. |
| | Broadcast | Not Allowed. |

The ArtMediaControlReply packet is used by a Media Server in reply to ArtMediaControl. The packet is also sent by the Media Server on power on or reset and when any relevant data is changed by user input to the media server such as a clip name change or transport control change.

| ArtMediaControlReply | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | ID[8] | Int8 | - | Array of 8 characters, the final character is a null termination.<br>Value = 'A' 'r' 't' '-' 'N' 'e' 't' 0x00 |
| 2 | OpCode | Int16 | - | OpMediaControl<br>Transmitted low byte first. |
| 3 | ProtVerH | Int8 | - | High byte of the Art-Net protocol revision number. |
| 4 | ProtVer | Int8 | - | Low byte of the Art-Net protocol revision number. Current value 14 |
| 5 | Filler1 | Int8 | - | Pad length to match ArtPoll. |
| 6 | Filler2 | Int8 | - | Pad length to match ArtPoll. |
| 7 | Filler3 | Int8 | - | Reserved. Transmit as zero. |
| 8 | Filler4 | Int8 | - | Reserved. Transmit as zero. |
| 9 | Stream | Int8 | - | Video stream number. |
| | | | | |
| 11 | MaxClipsHi | Int8 | - | The maximum number of clips available per stream |
| 12 | MaxClipsLo | Int8 | - | " |

| ArtMediaControlReply | | | | |
|---|---|---|---|---|
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 13 | TcDay | Int8 | - | Current position of media.<br>Days (0-255) or FrameNumberHi |
| 14 | TcHour | Int8 | - | Hours (0-23) or FrameNumber3 |
| 15 | TcMinute | Int8 | - | Minutes (0-59) or FrameNumber2 |
| 16 | TcSecond | Int8 | - | Seconds (0-59) or FrameNumber1 |
| 17 | TcFrames | Int8 | - | Frames or FrameNumberLo |
| 18 | TcMode | Int8 | - | Defines the time code data required for position commands and also the format of the above fields. |
| 19 | Status1 | Int8 | - | Status of the playback stream |
| | | | 7 | 1 = playing. 0 = stop. (1 = thaw. 0 = freeze) |
| | | | 6 | 1 = scan fwd. 0 = not scanning. |
| | | | 5 | 1 = scan rev. 0 = not scanning |
| | | | 4 | 1 = end of media reached |
| | | | 3 | =0 |
| | | | 2 | =0 |
| | | | 1 | =0 |
| | | | 0 | =0 |
| 20 | Status2 | Int8 | - | =0 |
| 21 | Status3 | Int8 | - | =0 |
| 22 | Status4 | Int8 | - | =0 |
| 23 | Data[128] | 1-128 | | Content defined by CtrlCommand. See Tables 7 & 8. |

**Table 8 – ArtMediaControlReply->Data format:**

The format of the Data field is defined by CtrlCommand. See Table 7. The field is variable length from 1 to 128. If the field is not required, the length is 1 byte. This entry is then set to MccNone.

| ArtMediaControlReply->Data | | | | |
|---|---|---|---|---|
| CtrlCommand == MccGetClipName | | | | |
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | CtrlCommand | Int8 | - | Command to which this packet replies. |
| 2 | Stream | Int8 | - | Stream Number to which name refers. |
| 3 | ClipHi | Int8 | | Clip Number to which name refers. |
| 4 | ClipLo | Int8 | - | =0 |
| 5 | Aux4 | Int8 | - | =0 |
| 6 | Aux3 | Int8 | - | =0 |
| 7 | Aux2 | Int8 | - | =0 |
| 8 | Aux1 | Int8 | - | =0 |

| | | | | |
|---|---|---|---|---|
| 9-128 | Text | Int8 | - | Requested text in null terminated ASCII. |

| **ArtMediaControlReply->Data** | | | | |
|---|---|---|---|---|
| **CtrlCommand == MccGetStreamName** | | | | |
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | CtrlCommand | Int8 | - | Command to which this packet replies. |
| 2 | Stream | Int8 | - | Stream number |
| 3 | Aux6 | Int8 | - | =0 |
| 4 | Aux5 | Int8 | - | =0 |
| 5 | Aux4 | Int8 | - | =0 |
| 6 | Aux3 | Int8 | - | =0 |
| 7 | Aux2 | Int8 | - | =0 |
| 8 | Aux1 | Int8 | - | =0 |
| 9-128 | Text | Int8 | - | Requested text in null terminated ASCII. |

| **ArtMediaControlReply->Data** | | | | |
|---|---|---|---|---|
| **CtrlCommand == MccGetModelName** | | | | |
| **Field** | **Name** | **Size** | **Bit** | **Description** |
| 1 | CtrlCommand | Int8 | - | Command to which this packet replies. |
| 2 | Aux7 | Int8 | - | =0 |
| 3 | Aux6 | Int8 | - | =0 |
| 4 | Aux5 | Int8 | - | =0 |
| 5 | Aux4 | Int8 | - | =0 |
| 6 | Aux3 | Int8 | - | =0 |
| 7 | Aux2 | Int8 | - | =0 |
| 8 | Aux1 | Int8 | - | =0 |
| 9-128 | Text | Int8 | - | Requested text in null terminated ASCII. |

## Art-Net Viewer:

The Art-New viewer software is available free of charge from the Artistic Licence web site. The software allows a total of six nodes to be interrogated and configured.

The software has been tested on all five Windows platforms. The software is intended for developers. Please use DMX-Workshop for network administration and configuration.

### Installation:

To install, please download the file and execute in a new folder. This will decompress the installation files. Next, run the setup.exe file.

After installation, confirm that:

1. You have a 10BaseT Ethernet card installed.
2. The PC's IP address is set to either:
   - 2.0.0.1 or
   - 10.0.0.1
3. The PC's sub-net mask is set to 255.0.0.0.

If you need to change the IP or sub-net settings of the PC, you should then re-boot. You will require a cross-wired (red) RJ45 cable to connect the PC directly to the DMX-Hub. If using an Ethernet Hub to allow connection of several DMX-Hubs, you will require straight wired (yellow) RJ45 cables.

## Status Screen:

The status screen provides a scrolling display of all Art-Net packets. The packets are decoded and displayed in a C-like form.
The scrolling display can be modified as follows:

1. Ignore non-matching IP's: If checked, only packets from an IP address matching the 'Node IP' (top left) will be displayed.
2. Ignore ArtPoll: If checked, ArtPoll messages will be filtered out.
3. Ignore ArtPollReply: If checked, ArtPollReply messages will be filtered out.
4. Ignore ArtDmx: If checked, ArtDmx messages will be filtered out.
5. Ignore ArtInput: If checked, ArtInput messages will be filtered out.

The 'Clear List' button clears the list contents.

### Polling:

Nodes should be at IP addresses of either 2.x.x.x or 10.x.x.x. The two buttons 'Scan2.255.255.255' and 'Scan 10.255.255.255' broadcast ArtPoll to the relevant limited broadcast address.
If Nodes are present, they will reply with ArtPollReply packets. These replies are displayed in the scrolling window. The software also decodes the replies and fills in the IP address of each Node that replied in the group of six buttons.

The six button group can now be used to send ArtPoll packets direct ot each Node's IP address (as opposed to broadcasting).

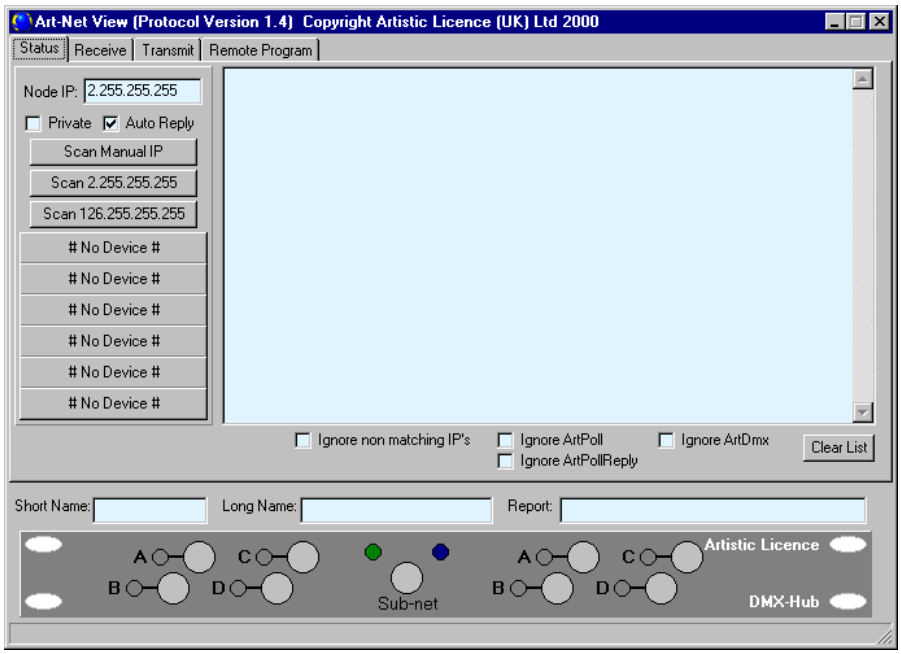
TalkToMe

The two check boxes 'Private' and 'Auto-Reply' modify the TalkToMe field in all ArtPoll packets that are sent.

If 'Private' is ticked, the Node is instructed to send ArtPollReply (but not ArtDmx) packets to the PC's IP address.

If 'Auto Reply' is ticked, the Node will send an ArtPollReply whenever it's status changes.

The 'Scan Manual IP' button sends an ArtPoll to the address typed into the 'Node IP' box.



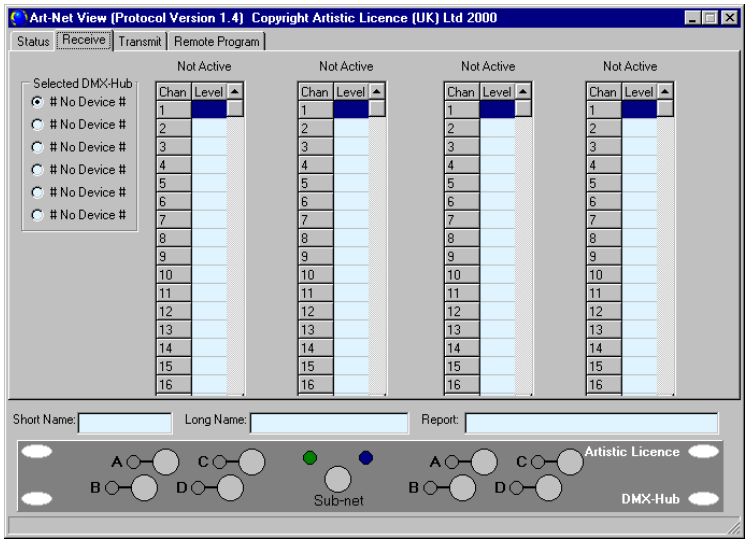

Lower Screen

The Lower screen section displays the following:

1. Node's Short Name. This is decoded from received ArtPollReply messages.
2. Node's Long Name. This is decoded from received ArtPollReply messages.
3. Node's Report. This is decoded from received ArtPollReply messages.
4. Display of a DMX-Hub or Netgate front panel. The controls of this display are updated to reflect the ArtPollReply settings.

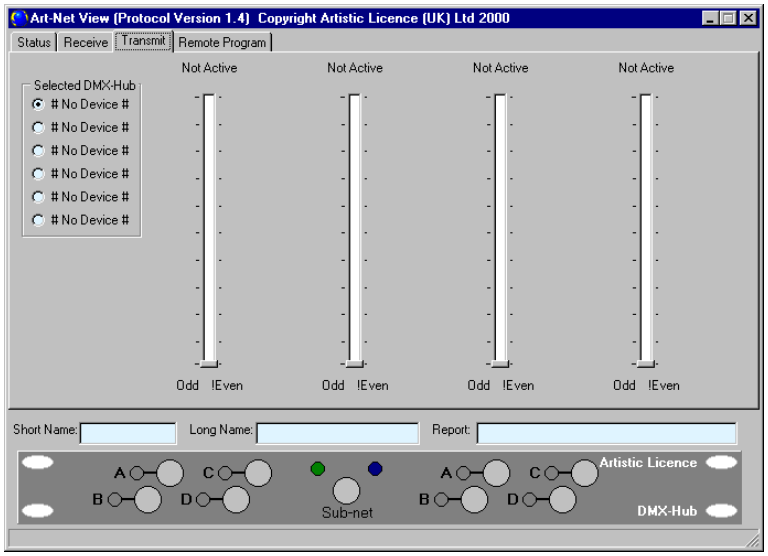
## Receive Screen:

The Receive screen displays four windows of DMX512 data decoded from ArtDmx packets. The title of each display shows the 8 bit Universe setting for each.
The Radio Button list allows one of six possible Nodes to be displayed.



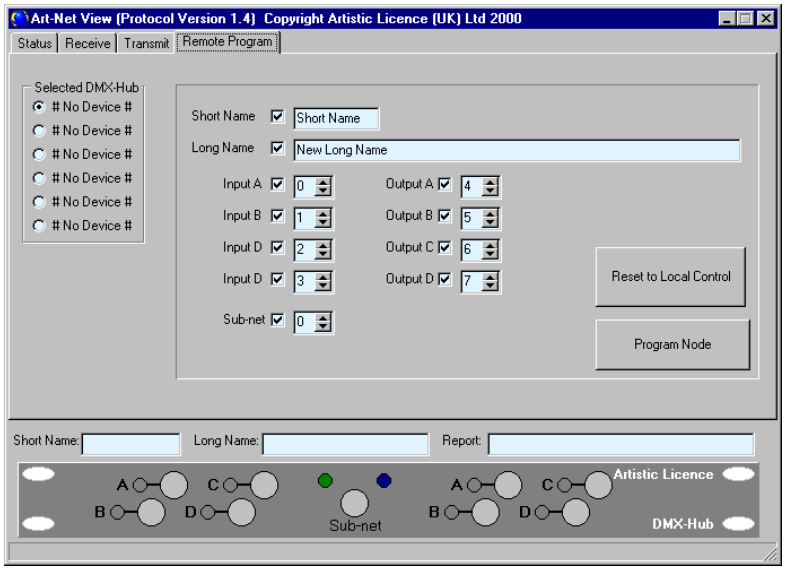

## Transmit Screen:

The Transmit provides four faders that set the output DMX levels of each Universe on the selected Node. The faders operate such that all odd numbered channels receive the fader value and all even numbered channels receive the inverted value.




The title of each display shows the 8 bit Universe setting for each.
The Radio Button list allows one of six possible Nodes to be displayed.

## Remote Program Screen:




The Remote Program screen allows all parameters of ArtAddress to be modified to re-program the selected Node.
The check box must be ticked in order for the value to be used.

## Known Bugs:

1. In heavy network activity situations the status screen can ignore user input. Work-around: Disconnect network cable until screen refresh stops, then tick the 'Ignore ArtDmx' check box.

Please report any problems to Support@ArtisticLicence.com

Wayne Howell 11/08/01

**Artistic Licence Product Notes:**

1. DMX-Hub:
   - The Network Switch is marked Switch 4
   - Merge functionality is not implemented
2. Down-Lynx:
   - Merge functionality is implemented
3. Ether-Lynx:
   - Merge functionality is implemented

***Artistic Licence***

B1 & B3 Livingstone Court
Peel Road
Harrow
Middlesex
England
HA3 7QT
Tel: +44 (0)20 88 63 45 15
Fax: +44 (0)20 84 26 05 51