Introduction

This document describes the serial and network control commands for the Datasat AP20 and AP25 Audio Processors. The AP20 and AP25 supports a command set for remote control and automation. These commands can be transmitted via either the Ethernet or the serial interface.

This document is intended to be used by any system integrator who needs to control the AP20 or AP25 remotely. Example of an application where these commands could be used may be an automation unit, a remote control unit, or a remote software interface.

It assumes that the reader is familiar with standard serial and network TCP/IP concepts.

Serial Control

The remote serial control device must be connected to the RS232 ”Control” connector on the back of the AP20 or AP25. To configure the serial port, go to the menu System -> Automation ->Serial. Select the desired baud rate. Also set Serial Command Mode to AP20.

For test purposes you may connect to the AP20 or AP25 using PuTTY or any similar serial communications program. Connection from a standard PC to the AP20 or AP25 is a straight-through cable.

Once the serial connection is made and setup the user can execute any of the commands listed within this document.

Ethernet connection to AP20 or AP25

The RJ45 connector labeled Ethernet on the back of the AP20 or AP25 can be connected to a network switch or router. Once the network parameters are properly set the IP addresses for the AP20 or AP25 can be found in the Network screen in the IP Address box.

The AP20 or AP25 may also be connected directly to another network device using crossover cable, or a straight cable if the device supports auto-MDIX.

The client initiates the communication session with the AP20 or AP25 IP address at port 14500. Once connected the client may send commands as described in this document to set or read the AP20 or AP25 configuration. The configuration changes happen as soon as they are received. For example, you should see the Fader volume change immediately after receiving a command to set the fader.

For test purposes you may use PuTTY or any similar communications program to make a TCP/IP connection to the AP20 or AP25.

Using PuTTY

Open PuTTY in the configuration Session and set the following: Host Name: (enter the AP20 or AP25 IP address) Port: 14500 Configuration type: Raw. Select the Open button. Once the network connection is made the user may type in commands listed within this document and read the response.

Using HyperTerminal

In HyperTerminal, select “Connect Using: TCP/IP (WinSock). Then enter the AP20 or AP25 IP address under “Host address:”, and 14500 for “Port number: ”.

Using Telnet

To use Telnet, enter the IP address and port number in the command line, for example:

```
# telnet 10.1.1.78 14500
```

Application Programming

Custom programs can be written to communicate with the AP20 or AP25 using standard TCP/IP communications. An example C language interface is included at the end of this document for reference.

Password Protection

The AP20 or AP25 may be protected from unauthorized access by setting a password under the **System > Access Control** screen. There are two levels of password protection in the AP20 or AP25, labeled **NetCmd Password** and **Setup Password**.

NetCmd Password

The NetCmd Password is used to prevent unauthorized access to the AP20 or AP25 through a network connection. The NetCmd Password does not prevent access through the serial command.

Setup Password

The Setup Password prevents unauthorized access to any AP20 or AP25 setup commands through the AP20 or AP25 local front panel or remotely through serial or Ethernet connections. This does not affect the operator level commands that are used in this document.

Authentication Command

The AUTH command must be sent to the AP20 or AP25 before sending a password protected command. If this is not done, then the command results in no action and the AP20 or AP25 returns the string “SECERR”. Sending the correct password enables all network commands for the password level for as long as the network connection is maintained.

Not all network commands require a password. Inquiry commands such as SYSTEM and IDENTIFY will operate without a password.

Command Format

The general command format for all configuration commands is listed below:

```
@COMMAND [ARG1] [ARG2] <CR>
```

Each **COMMAND** and its arguments are defined in this document. Whether or not **[ARG1]** and/or [ARG2] are used depends on the command. Square brackets **[]** around the arguments in this document indicate that the argument is optional.

The command is terminated by a **<CR>**. The response returns ASCII text and is also terminated by **<CR>** character at the end. The **<CR>** represents an ASCII character with the value 0x0D. How to enter this character in the command is entirely dependant on the remote program or interface used. On a terminal interface, it is added by pressing ENTER on the keyboard. In some GUI interfaces it is represented by “**\r**”, and for XML it may be **&#10**.

Important: If you are having problems with executing a simple command to the AP20 or AP25, check that the command string starts with ‘**@**’ and properly sends the carriage return at the end.

Some commands are characterized as “Read” and are used only to read status or information from the AP20 or AP25. Commands that are “Read/Write” can be used to set the specific configuration item, or just read it.

For “Read/Write” commands the last argument is the value to write to the configuration. Omit the final argument in order to read the configuration item without changing it.

General Commands

1. System Information

Returns system versions and MAC address

Command:	@SYSTEM <cr>	Operation
Response:	VER<space>[Version]<LF> VERDATE<space>[Date]<LF> MAC<space>[Mac Address]<CR><\0>	Read

Parameters

Version	Software version number
Date	Software date/time
Mac Address	AP20 or AP25 MAC address

2. Identify

Get system identify information. Mostly used in discovery protocol.

Command:	@IDENTIFY<cr>	Operation
Response:	AP20<space> [IP]<space> [Circuit]<space> [Theater]<space> [Screen]<cr>	Read

Parameters

AP20	Confirms AP20 or AP25 is connected at this address
[IP]	IP address (useful after broadcast command)
[Circuit]	Circuit information
[Theater]	Theater information
[Screen]	Screen information

3. Health

Enquiry for system health data.

Command:	@HEALTH [*SUB_CMD*]<cr>	Operation
Response:	(depends on SUB_CMD)	Read

SUB_CMD

TEMPERATURE

Returns t1, t2, t3 Celsius temperatures where:
t1: H331 board temperature
t2: H332 board temperature
t3: H335 board temperature

Example response:
HEALTH TEMPERATURE 34,29,25

H331VOLTS

Returns voltages sensed on H331 board,
<vok>,<ref>,<5v>,<+15v>,<-15v>,<-5V>

Example response:
HEALTH H331VOLTS 1,3.18,4.99,15.0,-15.0-,-5.0
<vok> is 1 if voltages are all within limits, else 0

H332VOLTS

Returns voltages sensed on H332 board,
<vok>,<ref>,<5v>,<+15v>,<-15v>,<-5V>

Example response:
HEALTH H332VOLTS 1,3.18,4.99,15.0,-15.0-,-5.0
<vok> is 1 if voltages are all within limits, else 0.
If H332 board is not present, response will be
HEALTH H332VOLTS NA

H335VOLTS

Returns voltages sensed on H335 board, <vok>,<ref>,<1.3v>

Example response:
HEALTH H335VOLTS 1,3.13,1.32
<vok> is 1 if voltages are all within limits, else 0

H336VOLTS

Returns voltages sensed on H336 board,
<vok>,<ref>,<+5V>,<+15V>,<-15V>,<48V>,<vcpu>

Example response:
HEALTH H336VOLTS 1,3.39,5.10,15.0,-14.4,0.0,1
<vok> is 1 if voltages are all within limits, else 0
<48V> is mic phantom power, will be 0 if phantom power off
<vcpu> will be 1 if CPU power in limits, else 0

H338VOLTS

Returns voltages sensed on H338 board,
<vok>,<ref>,<5v>,<+10V>,<-10V>

Example response:
HEALTH H338VOLTS 0,3.18,5.02,10.56,-10.48
<vok> is 1 if voltages are all within limits, else 0
If H338 board is not present, response will be:
HEALTH H338VOLTS NA

4. Board Information

Command: @**BOARDINFO**<cr>	**Operation**
Response: **H331,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** **H332,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** **H335,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** **H337In,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** **H337Out,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** **H338,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** **HDMI,[*ID*],[*AD*],[*R*],[*V*],[*CS*],[*FW*],[*FCS*],** <cr>	**Read**

Returns a list of boards, present and their hardware and PIC f/w versions.

Arguments

None

Board IDs:

H331	H331 Board
H332	H332 Board
H335	DSP/Motherboard
H337in	H337In
H337in	H337Out
HDMI	HDMI Interface board

Parameters:

[ID]	Board ID
[AD]	Board Slot Address
[R]	Hardware revision
[V]	Loader version
[CS]	Loader Checksum
[FW]	Firmware version
[FCS]	Firmware Checksum

5. Authorization

Command:	@**AUTH**<space>**[Password]**<cr>	**Operation**
Response:	**AUTH**<space>***[SETUP|OP|SECERR]***<cr>	**Read**

Give a password to allow usage of restricted commands. The authorization is required for many commands if access the AP20 or AP25 is configured with a Password. The **AUTH** must be issued before issuing any password protected commands, and is valid only for the duration of the TCP/IP connection.

There are two levels of password protection in the AP20 or AP25. Both levels are set in the **System > Access Control** screen on the AP20 or AP25. The top password labeled **NetCmd Password** will allow access to the AP20 or AP25 for Operator level type commands. The bottom password labeled **Setup Password** allows access to setup and configuration level commands.

The **AUTH** may be used for either the Operator or Setup level password.

Parameters:

[Password]	Operator level or Setup level password. The AP20 or AP25 compares this first with setup level password and gives Setup Level authorization if it matches. Otherwise, it compares it to the Operator (NetCmd) password and authorizes operator commands if it matches.
SETUP	**The AP20 or AP25** returns this value when Setup Level authorization has been granted.
OP	**The AP20 or AP25** returns this value when Operator Level authorization has been granted.
SECERR	**The AP20 or AP25** returns this value if neither Setup nor Operator level authorization has been granted.

6. Serial Number

Command:	@**SERIALNO**<CR>	**Operation**
Response:	**SERIALNO** <space>**[SN]**<CR>	**Read**

Reads the AP20 or AP25 serial number.

Parameters

[SN]	This value is the serial number that has been programmed into the unit during the manufacturing process.

7. MAC Address

Command:	@**MAC**<CR>	**Operation**
Response:	**MAC**<space>***[Mac adr]***<CR>	**Read**

Reads the AP20 or AP25 or AP25 network MAC address.

Parameters

[Mac adr] This is the 12 digit AP20 or AP25 network interface MAC address.

Example

Send: MAC
Receive: MAC 080077124578

Format Commands

8. Format Selection

Command:	@**FORMAT**<space>***[New Format]***<CR>	**Operation**
Response:	**FORMAT**<space>***[Format]***<CR>	**Read/Write**

This is used to select a new format, or view the current format.

Parameters

[New Format] This is the format name to select. The name must match exactly the format name on AP20 or AP25.
Note: Spaces may be used within the name.

[Format] This is the current format name.

Example

Set the format to Digital Cinema
Send: FORMAT Digital Cinema
Receive: FORMAT Digital Cinema

Automation

9. Execute an AP20 or AP25 Macro

Command:	@**RUNMACRO**<space>***[Macro]***<CR>	**Operation**
Response:	**[OK** or **ERR no macro]**<CR>	**Write**

This is used to execute a macro defined in the AP20 or AP25.

Parameters

[Macro]	This is the macro name to execute. The name must match exactly the macro name on AP20 or AP25. Note: Spaces may be used within the name.
OK	Response after macro is found and executed.
ERR no macro	Response if macro does not exist on the AP20 or AP25.

Example

Run Macro named Auto1

Send: RUNMACRO Auto1
Receive: OK

Level Commands

10. Master Fader Level

Command:	@**FADER**<space>***[New Level]***<CR>	**Operation**
Response:	**FADER**<space>***[Level]***<CR>	**Read/Write**

This is used to set or read the fader level.

Parameters

[New Level]	Value to set the fader in tenths. Omit this argument to only read the fader value.
[Level]	Current master fader level in tenths**.**

Example

Set the master fader to 7.0

Send: FADER 70
Receive: FADER 70

11. Master Fader Mute

Command:	@**MUTED**<space>***[New Value]***<CR>	**Operation**
Response:	**MUTED**<space>***[Value]***<CR>	**Read/Write**

Mute or Unmute the AP20 or AP25 output.

Parameters

[New Value] 1 to mute, 0 to unmute.

[Value] Current mute value.

Example

Mute

Send: MUTED 1
Receive: MUTED 1

12. Monitor Level

Command:	@**MONITORLEVEL**<space>***[New Value]***<CR>	**Operation**
Response:	**MONITORLEVEL**<space>***[Value]***<CR>	**Read/Write**

Set or read the AP20 or AP25 monitor level.

Parameters

[New Value] 0 (minimum) to 100 (maximum).

[Value] Current monitor level value.

Example

MONITORLEVEL

Send: MONITORLEVEL 70
Receive: MONITORLEVEL 70

13. Monitor Mute

Command:	@**MONITORMUTE**<space>***[New Value]***<CR>	**Operation**
Response:	**MONITORMUTE**<space>***[Value]***<CR>	**Read/Write**

Set or read the AP20 or AP25 monitor level.

Parameters

[New Value] 0 (unmute) or 1 (mute).

[Value] Current mutevalue.

Example

Mute the monitor.

Send: MONITORMUTE 1
Receive: MONITORMUTE 1

Sample Network Control Program

```
/* ====================================================================== **
 *
 *  Module:    Ap20NetCmd.cpp
 *
 *  Project:   AP20 or AP25 Ethernet Control Program
 *
 * ====================================================================== **
 *
 * CONFIDENTIAL: CONTAINS CONFIDENTIAL PROPRIETARY INFORMATION OWNED BY
 * DATASAT DIGITAL ENTERTAINMENT, INCLUDING BUT NOT LIMITED TO TRADE SECRETS,
 * KNOW-HOW, TECHNICAL AND BUSINESS INFORMATION. UNAUTHORIZED DISCLOSURE IS A
 * VIOLATION OF STATE, FEDERAL, AND INTERNATIONAL LAWS.
 *
 * DO NOT DUPLICATE.   COPYRIGHT 2009, DATASAT DIGITAL ENTERTAINMENT,
 * UNAUTHORIZED DUPLICATION IS A VIOLATION OF STATE, FEDERAL AND INTERNATIONAL
 * LAWS.
 *
 * USE OF THE SOFTWARE IS AT USER'S SOLE RISK. THE SOFTWARE AND RELATED
 * DOCUMENTATION ARE PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND AND
 * DDE EXPRESSLY DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING,
 * BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
 * FOR A PARTICULAR PURPOSE. DDE DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED
 * IN THE SOFTWARE WILL MEET USER.S REQUIREMENTS, OR THAT THE OPERATION OF
 * THE SOFTWARE WILL BE UNINTERRUPTED OR ERROR-FREE, OR THAT DEFECTS IN THE
 * SOFTWARE WILL BE CORRECTED.  UNDER NO CIRCUMSTANCES, INCLUDING NEGLIGENCE,
 * SHALL DDE, OR ITS DIRECTORS, OFFICERS, EMPLOYEES OR AGENTS, BE LIABLE
 * TO USER FOR ANY INCIDENTAL, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES
 * (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION,
 * LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE, MISUSE
 * OR INABILITY TO USE THE SOFTWARE OR RELATED DOCUMENTATION.
 *
 * ====================================================================== */

/* ====================================================================== **
 *   Compile Options
 * ====================================================================== */

#define StrAp20Ip        "10.1.1.78"
#define StrAp20Password  "xyz"

/* ====================================================================== **
 *   Include Files
 * ====================================================================== */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <arpa/inet.h>

/* ====================================================================== **
 *   Definitions
 * ====================================================================== */

#define AP20_PORT_NUM    14500

/* ====================================================================== **
 *  Data
 * ====================================================================== */

#define RX_BUF_SIZE 2048
char rxBuf[ RX_BUF_SIZE + 1 ];
```

```
/* ========================================================================== **
 *  Prototypes
 * ========================================================================== */

int AP20Command( char *strAp20_IpAddress, char *StrCmd, char *StrPassword );
int Send( int fd , char *StrCmd );
int ReadResponse( int fd , char *StrCmd );

/* ========================================================================== **
 *  Functions
 * ========================================================================== */

/* -------------------------------------------------------------------------- **
 *  Function:  main
 *
 *  Picks up the Command from the command line arguments.
 *  In this example the AP20 IP address and AP20 Setup password is hardcoded.
 * -------------------------------------------------------------------------- */

int main (int argc, char **argv)
{
  char StrCmd[256];
  int cnt;

  if ( argc < 2)
    {
      printf ("Usage: Ap20NetCmd arg1 ... arg\n");
      exit(1);
    }

  // collect args
  int firstarg=1;
  snprintf( StrCmd, sizeof(StrCmd), "%s", argv[firstarg++] );
  for ( cnt = firstarg; cnt < argc; cnt++ )
    {
      strcat ( StrCmd , " ");
      strcat ( StrCmd , argv[cnt]);
    }

  AP20Command( StrAp20Ip, StrCmd, StrAp20Password );
}


/* -------------------------------------------------------------------------- **
 *  Function:  AP20Command
 *
 *  - A socket connection to the AP20 is established to the AP20 IP
 *    address using port 14500.
 *  - Send the AUTH command if the AP20 has a password defined.
 * -------------------------------------------------------------------------- */

int AP20Command( char *strAp20_IpAddress, char *StrCmd, char *StrPassword )
{

  int fd;
  struct sockaddr_in MySocket;       // sender main socket

  memset(&MySocket, 0, sizeof(MySocket));
  MySocket.sin_addr.s_addr = inet_addr( "127.0.0.1" );
  MySocket.sin_addr.s_addr = inet_addr( strAp20_IpAddress );    // Set the AP20 IP address here
  MySocket.sin_family      = AF_INET;
  MySocket.sin_port        = htons( AP20_PORT_NUM );            // Set the AP20 Port address

  // Get a file descriptor for the socket
  if ((fd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
      printf("socket() failed\n");
      return -1;
    }
```

```
  // Connect to the AP20
  if( ::connect( fd, (struct sockaddr *)&MySocket, sizeof( MySocket ) ) != 0 )
    {
      printf("connect() fail\n");
      close( fd );
      fd = -1;
      return -1;
    }

  printf( "Connection to %s:%d OK\n", strAp20_IpAddress, AP20_PORT_NUM );

  // Send passord only if AP20 has Setup Password defined

  char StrAuth[100];
  sprintf( StrAuth, "AUTH %s", StrPassword );
  if( strlen( StrPassword ) )
    {
      Send( fd, StrAuth );
      ReadResponse( fd, StrAuth );
      if( !strcmp( rxBuf, "SETUP" ) )
        {
          printf( "Wrong Password\n" );
          return( -1 );
        }
      else
        printf( "Password OK\n" );
    }

  Send( fd, StrCmd );
  ReadResponse( fd, StrCmd );

  close( fd );                                  // Close connection
  fd = -1;
  return( 0 );
}


/* ---------------------------------------------------------------------- **
 *  Function:  Send
 *
 *  Sends a command string the AP20.
 * ---------------------------------------------------------------------- */

int Send( int fd , char *StrCmd )
{
  char strAP20Cmd[1024];

  // Command starts with '@' and ends with CR

  strcpy( strAP20Cmd, "@"    );
  strcat( strAP20Cmd, StrCmd );
  strcat( strAP20Cmd, "\r"   );

  printf( "Sending NetCmd to AP20: %s\n", strAP20Cmd );

  int ret = write(fd,&strAP20Cmd,strlen(strAP20Cmd) );
  if ( ret < 0 )
    {
      fprintf(stderr,"write fail\n");
      close(fd);
      fd = -1;
    }
  return 0;
}

/* ---------------------------------------------------------------------- **
 *  Function:  ReadResponse
 *
```

```
 *  Reads data received from the AP20 until CR is received.
 * ------------------------------------------------------------------------ */

int ReadResponse( int fd , char *StrCmd )
{
  char c;
  int count=0;

  for( int i=0; i<RX_BUF_SIZE; i++ )
    {
      // read 1-by-1.
      int ret = read(fd, &c, 1);
      if ( ret < 0 )
        {
          printf( "Count=%d, Error\n", count );
          close(fd);
          fd = -1;
        }

      if (ret == 1)
        {
          // add to buffer
          rxBuf[count++] = c;
          if( c == '\r' )                     // End of response
            {
              rxBuf[count] = 0;
              printf( "%s\n", rxBuf );
              break;
            }
        }
    }
  return 0;
}

/* ======================================================================== **
 *            # # #   End of Ap20NetCmd.cpp     # # #
 * ======================================================================== */
```

Confidentiality Statement

CONFIDENTIAL: Contains confidential proprietary information owned by Datasat Digital Entertainment, including but not limited to trade secrets, know-how, technical and business information. Unauthorized disclosure is a violation of State, Federal, and International laws. Use limited to licensees of Datasat Digital Entertainment.

Copyright Protection

UNDER NO CIRCUMSTANCES, INCLUDING, WITHOUT LIMITATION, NEGLIGENCE, SHALL DATASAT DIGITAL ENTERTAINMENT OR ITS DIRECTORS, OFFICERS, EMPLOYEES OR AGENTS BE LIABLE FOR PERSONAL INJURY, OR ANY INCIDENTAL, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, LOSS OF DATA, BUSINESS INTERRUPTION OR ANY OTHER COMMERCIAL DAMAGES OR LOSSES, ARISING OUT OF OR RELATED TO YOUR USE OF THE DATASAT DIGITAL ENTERTAINMENT SOFTWARE, HOWEVER CAUSED, REGARDLESS OF THE THEORTY OF LIABILITY AND EVEN IF DATASAT DIGITAL ENTERTAINMENT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL DATASAT DIGITAL ENTERTAINMENT'S LIABILITY EXCEED THE AMOUNT PAID TO DATASAT DIGITAL ENTERTAINMENT. SOME STATES DO NOT ALLOW THE LIMITATION AND/OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

Datasat Digital Entertainment Technical Support

Email: techsupport@ati-amp.com
Tel outside USA: +1.818.401.4253
Tel in USA: 888.428.2268
Fax: +1.805.306.8029
www.datasatdigital.com