DMP3 serial communication v1.20

1.1 System design

Through the serial port of the DMP3 mp3 player several external devices can be connected to the system, which according to the implemented function can be:

Controller	sends such button and jog control commands to the DMP3 that are similar to that of the user interface (E.g. External keypad or IR/RF remote controller)
Display	receives and displays messages of the DMP3
Terminal	has both controller and display functions E.g. Satellite Display or Cd-Changer simulator
Port	has only receiver function, sometimes processes special packets E.g. Fm Modulator

The serial communication port can be found on the DMP3 drawer's 50 pins Centronics connector, on the CarHost frame's Car Port (8 pol.) and the RJ11 6/6 connectors.

Setup of the system and plugging the connectors is further explained in Appendix #1.

1.2 The serial line

The format of the serial line is RS232, 19200 baud 8N2 (8 bit, no parity, 2 stop bit). Contrary to the standard RS232 we apply a 5V signal level, in inactive (logical 0) state the line level is 5V, in active (logical 1) state 0V.

Leading of the serial line between the units is such, that all units receive the outgoing messages of the DMP3 (TXD broadcast), while the DMP3 is able to receive the message of any unit (RXD). The serial outputs of the peripheral units are Open Collector type and all are connected to the input of the DMP3 RXD (See Appendix #1).

Installing several peripheral units the resistors protecting the TXD and RXD lines and the parallelly connecting pull-up resistors have to be considered. These forming a divider may reduce the signal level and by this make communication impossible. In order to avoid this, in systems consisting of many external units signal regenerators and hubs have to be installed.

Because of the above described design, in case of Broadcast data sending, the DMP3 cannot get response about the reception of messages, because in case of more units the answers would collide.

Likewise, the spontaneously incoming data packets from different units can collide as well. In the system there is no distinct collision detection, in case of colliding, the checksum of the communication packets is damaged and the receiver side of the DMP3 drops the corrupted messages.

The capacity of the DMP3 serial sending and receiving buffer us 128-128 byte. The overflow of the receiving buffer may lead to data loss or to the overwriting of already buffered data. It is expedient to have on the receiving side of the peripherial units an at least 128 byte capacity receiving buffer or process the received data as soon as possible.

In addition, the DMP3 implements a point-to-point serial communication protocol of which the functioning is discussed in a separate document. Packets of this type have to be discharged by the peripheral unit.

1.3 Awakening the DMP3 system from standby state

The DMP3 in standby state monitors its own buttons, the InUse input and the level of the RXD line. If it perceives an InUse=12V level, or for at least 100msec 0 level on the RXD line or that a button is pushed, it gets to active state.

When getting to active state it gives 100 msec 0 level on the TXD line and with this wakes up the units found in the system. Valid serial line communication can only begin after this.

2. Data communication

Units belonging to the DMP3 system run communication channels on the serial line interface independent from each other and not excluding each other.

The length of a message can be maximum 517 byte. In order to be easily detected and corrected, messages when being transferred are broken down to shorter, a maximum of 16 byte packets.

Packets of the communication channels (broadcast messages, two-way protocol, FM Modulator messages) can be mixed, keeping the integrity of the packets.

2.1.Packet format

Serial packet format (SERCOM_PACKET_S structure)		
Type	**Field name**	**Description**
BYTE	header	Packet header character, the value is always 0xAA
BYTE	chksum	Check sum: except for the header character, the sum of the whole packet's byte's negate
BYTE	id	bit 0-5: serial number of the packet in the message bit 6: in case of broadcast packet, its value is 1 bit 7: in case of the last packet, its value is 1
BYTE	len	Number of data bytes in the packet: 0 <= len <= 16.
BYTE	data[0] .. data[len]	Packet data, variable length, maximum 16 byte

The lower 6 bits of the id field contains the serial number of the packet, the first packet starts from 0. If the receiver (the DMP3 also) gets a 0 serial number packet, it starts to assemble the message. Based on the id, each packet can be identified, a repeated packet can be discharged on the reception side, or replace the corrupted packet received earlier.

When a packet with out of sequence number (and not the retransmission of a previous one) is received, processing of the packet is aborted. Reception of the packet is finished when the last packet is finished; then bit 7 = 1 in the id.

In case of id bit 6 = 0 a two-way communication packet arrives, packets like this have to be discharged on the peripherial side.

Example for a 17 byte long Broadcast message (hex data):
Message: 30 31 32 33 34 35 36 37 38 39 3A 3B 3D 3E 3F 40

Packet #1: AA 87 40 10 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E
Packet #2: AA FD C1 01 40

3. Broadcast messages

To the DMP3 more externally controlled devices (keypad and terminal type units) can be connected at the same time. Controllers send button and jog (boradcast) messages to the DMP3 (KEYEVENT, JOGEVENT) and can receive SETCTRL and STANDBY messages.

Other messages of the DMP3 are sent to the display type units. Not the image element's parameters, but that of the drawing items displayed on the LCD display (strings, logo, sand-glass, VUmeter) are to be sent, drawing is managed by the displaying unit.

The first two words of the Broadcast message (2 byte) contains the type of the message. Format and length of further data depends on the type of the message. The DMP3 does not check the content of the message; the check-sum excludes the processing most of damaged or truncated messages.

Broadcast message types			
Type	**Value** (first WORD of the message)	**Source**	**Description, function**
KEYEVENT	0x0000	Controller	Message describing button events
JOGEVENT	0x0001	Controller	Message describing jog button event s
SETCTRL	0x0002	DMP3	Message enabling/disabling the controller
LCD	0x0003	DMP3	Message forwarding the content of the LCD display, text lines and LCD control
VUMETER	0x0004	DMP3	Message forwarding the status of the VUmeter
WAITANI	0x0005	DMP3	Waiting animation (sand-glass) controlling message. Not used.
STANDBY	0x0006	DMP3	Standby (turn-off) command to the peripherial units
DEFLOGO	0x0007	DMP3	Activation of the default logo on the display
MENULOCAL	0x0008	DMP3	Activation of the local menu on the terminal
UPDATE_POLL	0x0080	DMP3	Polling of SW update request
File transmission	0x00F0 + code of the file transmission	DMP3	User logo (0xF0) and SW update file (0xF1) transmission

3.1 Secure Broadcast protokol

Since there is no acknowledgment about the correct transmission of the messages, to forward important messages (file transfer, standby state), the DMP3 uses the SECURE broadcast transmission.

In case of secure boradcast, the packets of the message are sent twice (0-0, 1-1…), so in case a packet is damaged, the repetition following it can be accepted.

The secure broadcast message always consists of minimum two packets. Shorter messages have to be expanded with dummy data (otherwise they could not be separated from messages sent twice after each other).

Controllers usually send short messages to the DMP3, to minimize probability of collision. Secure broadcast can also be used, but practically it is not necessary. (Usually messages are generated by the user, who on the user interface gets some kind of response, and if the message is lost, repeats the action).

3.2 SETCTRL - Message enabling/disabling the controller (from the DMP3)

Type	Value	Description
WORD	0x02	SETCTRL – permission/prohibition of the controller
DWORD	SerId	The 4 byte identificator of the enabled controller (serial number)

Each controller sends messages identified by its serial number to the DMP3 (SerId, 4 byte). (In case of a non DENSION unit, the serial number domain has to be registered to the given device.) The DMP3 receives and stores the current controller's id and sends out a SETCTRL controller enabling/disabling message. The SETCTRL broadcast message provides the id of the controller, which further on will "posses" control. Based on this, other controllers, if receive a different SETCTRL message different from their code can prohibit data sending, and this may appear on its display (if it has one).

While a given controller posseses the user interface, messages arriving with different id are discharged by the DMP3. In every case the DMP3's own user interface can be used, events arriving from this do not modify the current controller id.

The default controller id stored in the DMP3 is 0. This means, that any controller may register in. If no message arrives from a controller for 30 seconds, or the DMP3 gets back to player display mode (or stays in it), the value of the Controller's id will be 0 again. In case of a change, the DMP3 sends SETCTRL message, which again permits all units for data sending.

Special identifierts, which do not change the current controller:

`0x00000000`	UNLOCK	the controller sends a message but does not take over the user interface
`0xFFFFFFFF`	FORCE	data by any means will be accepted

Implementation of controller enabling message in case of simpler controllers (keypad) is not an essential requirement, correct functioining can be still implemented.

3.3 JOGEVENT - Jog message received from a controller

Type	Value	Description
WORD	0x01	JOGEVENT– description of a jog event
DWORD	SerId	4-byte id of the controller (serial number)
int	Code of jog event	Signed number of jog turn steps during a time interval Positive value is a turning clock-wise

3.4 KEYEVENT - Button messages arriving from a controller

Type	Value	Description
WORD	0x00	KEYEVENT – description of key press
DWORD	SerId	The 4-byte identifier of the controller (serial number)
int	Code of button event	Button codes different from the ones given here cannot be used, their effect is not defined (decimal value) **Pushing buttons for short**. Message is created when the button is released 19 NEXT, 20 BACK, 21 CIRCLE, 22 SQUARE, 23 JOG, **Pushing buttons for long**. Message is generated every 400msec while pushing the button 27 NEXT_L, 28 BACK_L, 29 CIRCLE_L, 30 SQUARE_L, 31 JOG_L. **Events generated by external keypad or CD-Changer simulator** **Configurable pushbutton codes**: 64 KEY_EXT01, 65 KEY_EXT02, 66 KEY_EXT03, 67 KEY_EXT04, The effect of the KEY_EXT## codes can be defined in a .CFG file **Special button event codes:** 129 KEYEVENT_PAUSE, 130 KEYEVENT_POWEROFF, 131 KEYEVENT_PLAY_NEXT, 132 KEYEVENT_PLAY_BACK, 133 KEYEVENT_NEXTDIR steps to the beginning of the next playing directory 134 KEYEVENT_BACKDIR steps to the beginning of the preivous playing directory 135 KEYEVENT_PLAYALBUM album play.

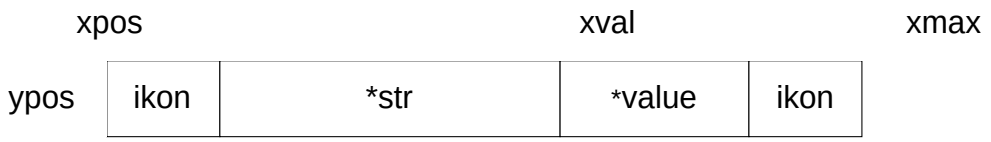
When pushing buttons for long, the DMP3 must detect when a button is released (e.g. fast forward/rewind, playing a song). In order this after each pushbutton message is accepted, the DMP3 prohibits to disables its own buttons for 800msec. If a long-pus h message arrives during this, then the button hasn't been released. If no button message arrives, the DMP3 generates the button release event by itself.

3.5 LCD - Messages forwarding the content of the display

Description of the LINE structure

The image to be displayed is described by a LINE structure. The actual (graphical) image is put together based on this and the unit's own abilites.

Elements of the LINE structure	
xpos	x position of the LINE's left top corner
xval	x position dividing the LINE into string and value fields
ypos	y position of the LINE's left top corner
attr	LINE characteristics: adjustment, font type, scroll, etc
xmax	x position of the LINE's right top corner
drawpos	Position of the first displayed character in the string pointed by ***str**
***str**	Pointer to the string to be displayed on the left side of the LINE
***value**	Pointer to the string to be displayed on the right side of the LINE
likon	Icon to be displayed on the left side of the LINE
rikon	Icon to be displayed on the right side of the LINE



The [0,0] coordinate is the left top corner of the display. The size of the display is 128x56 pixel.

Each LINE can be considered a window, of which the left top corner is given by the **xpos** and **ypos** fields, and the width is determined by the **xmax** field. Height of the window depends on the font used, which is described by the **attr** field.

The window might start and end with an ikon, which are determinded by the **likon** and **rikon** fields.

A LINE can be divided into two with the **xval** field. To the part before the **xval** the string defined by the ***str** pointer gets, while after it the string defined by the ***value** pointer gets.

The ***str** and ***value** pointers can point to an internal memory address, or to a text put together in a RAM; so in this format they can only be interpreted by the DMP3. Messages sent to the peripherial units are so completed with the content of the actual strings.

Interpretation of the attr field		
D[15]	StringId	If the value is 0 there is a playdisp type item, otherwise menu or browser
D[14 - 10]	StringInfo	Code of the string information (time, artist …etc. see later)
D[9]	N.A.	Not used
D[8]	Autoscroll	Ths string to be displayed has to be scrolled, if its value is 1
D[7 - 6]	Font type	0x00 – 8 pixel high fonts 0x11 – 12 pixel high fonts other values are not used
D[5]	Center	String has to be displayed aligning in the middle
D[4]	Left	String has to be displayed aligning on the left. If the alignment of the string is not obvious, then it has to be displayed aligning to the left.
D[3]	Right	String has to be displayed aligning on the right
D[2]	Inverz	The LINE has to be displayed in inverse
D[1]	N.A.	Not used
D[0]	DispEnab	Enabling drawing, if 1, the LINE is not to be displayed (it is possible to send not displayed messages as well)

StringInfo code (bit 14-10 in the attr field) interpretation	
D[14-10]	**Leírás**
0x01	File name
0x02	File name extension. File name aligned on the left, if there is not enough space, normal file name.
0x04	ID3, Title field.
0x05	ID3, Artist field.
0x06	ID3 Album field.
0x07	ID3 Year field.
0x08	ID3 Comment field.
0x09	ID3 Track field.
0x0A	ID3 Genre field.
0x0B	MPEG and Layer type information (e.g.: „MP1L3”).
0x0C	Bitrate information (e.g.: „128kbps”).
0x0D	Sampling rate (e.g.: „44kHz”).
0x0E	Mono / stereo information. („m” / „∞”) (one character)
0x10	Total playing time of the current title
0x11	Remaining playing time of the current title
0x12	Elapsed playing time of the current title
0x14	Displays playing state (”◀◀”,”▶▶”,”⏮”,”⏭”,”⏸”,”▶”,”■”).
0x15	Repeat mode, abbreviated (”Off”,”Rep”,”Rnd”,”Trk”).
0x16	Play mode, abbreviated (”Disk”,”Ins”,”Prg”,”Mrd”).
0x17	Play and Repeat mode together (e.g.: ”Jukebox”,”Ins Rnd”,MyRadio file name,….etc).
0x18	Number of all elements in the program
0x19	Serial number of the currently played number in the program

Messages forwarding LINE structure informtation

The above described LINE structure is sent together with the strings to be displayed to the peripherial units (SERIALLINE structure).

There can be a maximum of 16 LINEs on the display at the same time, these are identified by the LINENUM. The DMP3 sends the SERIALLINE structures continuously, which are stored by the display unit, and reproduces for the appropriate commands.

Forwarding of SERIALLINE message on the serial line		
Type	**Value**	**Description**
WORD	0x03	BROADCASTTYPE_LCD, sending the description of a/one LINE
WORD	LCDCTRL	If 0, the LINT to be displayed arrives, otherwise some kind of command to the dsiplay (see later)
WORD	LINENUM	Position of LINE in an array. The ones arriving with the same serial number overwrite each other. The value is 0x00 – 0x0F. if the value is 0x10 then it is a message window and is to displayed immediately. (the message window is always preceeded by a LCDCTRLSAVEMSGBACKGROUND command, and when the message becomes invalid, it is closed down by a LCDCTRLCLEARMSGLINE)
WORD	xpos	x position of the LINE's left top corner
WORD	xval	x position dividing the LINE into two
WORD	ypos	y position of the LINE's left top corner
WORD	attr	LINE characteristics: alignment, font type, scroll, etc
WORD	xmax	x position of the LINE's right top corner
WORD	drawpos	Position of first displayed character in the string. Concerns only the str.
DWORD	*str	Inside pointer, cannot be interpreted for the peripheral device.
DWORD	*value	Inside pointer, cannot be interpreted for the peripheral device.
BYTE	likon	Character code of the icon appearing on the left side of the LINE
BYTE	rikon	Character code of the icon appearing on the right side of the LINE
BYTE	str	Characters of the str string, + closing zero*
BYTE	value	Characters of thevalue string, + closing zero*

* length of the **str** and **value** strings can together be maximum 520 Byte

LCDCTRL – description of comnads controlling display		
Value	**Érték**	**Effect of command**
LCDLINE	0x00	LINE to be displayed arrives
LCDCTRLCLRLCD	0x02	Deletion of display, deletion of all LINES arriving before the command
LCDCTRLREFLCD	0x03	Assembly and refreshment of the display based on the sent LINEs
LCDCTRLREFDISP	0x04	Refreshment of the dsiplay
LCDCTRLCLEARMSGLINE	0x06	Deletion of message window, restoration of the previously saved image information. (See. LCDCTRLSAVEMSGBACKGROUND)
LCDCTRLLCDON	0x07	Turning on the display
LCDCTRLLCDOFF	0x08	Turning off the display
LCDCTRLLOGO	0x09	Display the default logo stored in the iinternal ROM or sent previously on the serial line
LCDCTRLSAVEMSGBACKGROUND	0x0A	Save the background and draw the frame of the message window before displaying the message

		window

3.6 VUMETER - Message forwarding the state of the VUmeter

If the display type unit gets a message of this type, it draws the VUmeter with the given parameters, and refreshes the content of the LCD. Size of VUmeter is the same as DMP3's VUmeter; style can be individual (continuous line, no peek, etc.)

Type	Value	Description
WORD	0x04	BROADCASTTYPE_VUMETER
WORD	VOL_LEFT	Left channel current volume (0-48)
WORD	VOL_RIGHT	Right channel current volume (0-48)
WORD	PEAK_LEFT	Left channel peak value (0-48)
WORD	PEAK_RIGHT	Right channel peak value (0-48)

3.7 WAITANI - Message controlling the waiting animation (sand-glass)

If the display type unit gets a message of this type, then after saving the background the waiting symbol is drawn may be animated.

Type	Value	Description
WORD	0x05	BROADCASTTYPE_WAITANI

3.8 DEFLOGO - Activation of default logo on the terminal's display

Type	Value	Description
WORD	0x07	BROADCASTTYPE_DEFLOGO

As an effect of the DEFLOGO the terminal delets the logo received earlier and activates the device's default logo. There is no display of logo in this case.

3.9 MENULOCAL - Activation of local menus on Terminal type units

Type	Value	Description
WORD	0x08	BROADCASTTYPE_MENULOCAL – activation of local menu
WORD	addr	The Eeprom address of the parameter set by a local menu item

In case of certain hardware parameters it is not necessary to change concrete settings of an external terminal concerning the DMP3 drawer. Such types are the CONTRAST, BACKLIGHT, INVERSE and ROTATE values.

In case of trying to enter to such menus the DMP3 does not take into consideration of pushing the JOG (does not enters into its own menu), but sends a MENULOCAL broadcast message to the controller. This command orders the Terminal to open its own local menu in the given place of the **addr** part of the message. The addr is the EEPROM address of the value of the parameter to be set, from this the terminal can identify the tasks concerning its own local menu.

MENULOCAL menu item	EEPROM addr
Contrast	0x28
Backlight	0x2A
Inverse Mode	0x32
Rotate Display	0x34

3.10 STANDBY - turning off peripherial units

Type	Value	Description
WORD	0x06	BROADCASTTYPE_STANDBY

Implementing message receiving in case of simpler controllers (keypad) is not an essential requirement, correct functioining can be still implemented. The DMP3 sends this message with secure broadcast protocol.

3.11 UPDATE_POLL - Polling for SW update request

Type	Value	Description
WORD	0x80	BROADCASTTYPE_UPDATE_POLL
char[9]	header	First 9 bytes of the header of software update file (EXTUPD.DAT) The header contains the unique identifier and version of each unit. E.g. in case of the Satellite Display "P3SAT1.20" (+closing 0)
BYTE	0x00	Closing 0

The DMP3 uses the described software uptade mode from 1.20 version up. (earlier versions just sent out the SATUPD.DAT file, if there was such)

When turning on power the DMP3 searches in the /DMP3 directory the EXTUPD.DAT file. If it exists, then it sends the header from the first 9 characters of the file in the UPDATE_POLL message. The units decide based on the header and the version wether they want the given file transfer.

Possible answers of the units to the UPDATE_POLL message:

- accepting the Update: for 500msec sending the '0' character in every 5 msec
- rejection of the Update: immediate stops the serial transmission for 500msec

This message has to be implemented in each unit, or the suspension of the serial transmission must be guaranteed in a different way in the critical period.

3.12 File transfer – Transfer of user logo and SW update files

The file transfer messages make it possible for the SW update, user logo and other large data to be transferred to the units. In case of file transfer the DMP3 always uses the secure broadcast protocol.

Structure of file transfer message		
Type	**Value**	**Description**
WORD	0x00F0 + the code of transfer	File transfer code: 0x00F0 – user logo 0x00F1 – SW update file.
WORD	addr	Position of the transferred data block in the file
WORD	len	bit0-14: length of the data block in byte 0 <= len <= 128 bit15: compressed transfer if each byte of the data block has the same value, then there is a compressed transfer
BYTE	data[128]	Len length data block. In case of compressed átvitel, the length independent of len is 1 byte

The DMP3 may initiate file transfer at any time. The transfer happens strictly in a continuous way, from the addr=0 position till the end of the file. To avoid addressing out of allowed area, protection is needed in the programs of the peripherial units.

If each byte of the data block has the same value, then there is a compressed transfer. Then there is only 1 byte to be transferred, the other bytes of the block are reconstructed by the unit. Compressed transfer is indicated by the uppermost bit of the len field.

In case of a user logo the length of the file is always 1024 byte, its format is the same as that of the .LCE extension logo file format, the description of which can be found in a separate document.

The length of the SW Update data is variable depending on peripherial unit type, but it can be a maximum of 65536 byte. The first 9 bytes of the header of the update file tells which unit the Sw update concerns, and from this the given unit also knows the length. Units different from this type alsp accept the update file, but at the end of transfer (or in case of buffer overlow) they discharge it.

At the end of SW update transfer - if the update is to the given unit – the unit executes the burn-in of its own firmware. During this the device has to protected against loss of power and noises.

3.13 Format of the software update file

Programs of the external units can be refreshed with the help of the \DMP3\EXTUPD.DAT file. Since the name of the file is fixed, one type unit's program can be refreshed at once.

Length of the file is changing by units, it can be maximum 65523 byte. The first 16 bytes of the EXTUPD.DAT file contain a header.

Format of the software update file's header		
Type	**Value**	**Description**
char	header[5]	Update identifier string, e.g. "P3SAT". This tells whch unit the update concerns
char	version[4]	Version identifier string
BYTE	0×00	Closing 0
BYTE	chksum	Check sum of the update file (ones complement of the sum of all the bytes). If the check-sum is wrong, the DMP3 skips the file
BYTE	force_flag	If the value differs from 0, refreshment will take place independent from the user's request
BYTE	reserved[4]	Not used bytes, their value must be 0

4.1. The FM Modulator commands

Message controlling the choice of the FM modulator frequency. Because the functioning of the FM Modulator is more or less simple, this a sepecial message with a distinctive header (not broadcast).

The simpler Fm Modulator (V1) is only capable to set one of the beforhand fixed frequencies.this is given by the FMTR_STATE value. The more developed FM Modulator (V2) is suitable to set arbitrary frequency, the value of which is determined by the REEQ_STR. The string is a value determined in the .MGS file (more precisly in the Default of it).

Type	**Field name**	**Value**	**Description**
BYTE	SERIAL_FMTR_HEADER	0xBB	Header of the FMTR packet
BYTE	FM_CHANNEL		FM channel code 0...15.
BYTE	~FM_CHANNEL		FMTR_STATE byte negate per bits
BYTE	FREQ_STR_CHKSUM		The check sum of FREQ_STR (negate of the sum of bytes per bits).
char[5]	FREQ_STR		The first five characters of the frequency identifing string ASCI. The value of the string is determinmded by the .MGS file loaded in the SW Setup Messages. If the string is shorter, then it is closed down by a 0 byte.

Example: turning on the FM Modulator and choosing the 1st FM channel, or the 87.1 MHz (hexa data):

FMTR message: BB 01 FE 31 38 37 2E 31 00

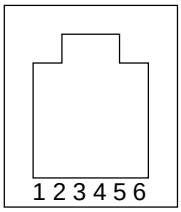
Appendix #1 – Design of the DMP3 system

Based on the experiences aquired while producing the DMP3 the mechanical and electronic design of the CarHost Frame has been modified. This basicly means compatibility between old and new devices and also for the peripherial units, but their distinction is still necessary.

Version V1
The first production series, made with metal CarHost frame, separated Satellite Display and FM Modulator connectors.

Satellite Display

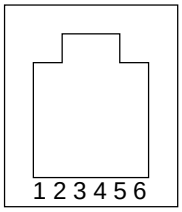
RJ11_6/6 Front view



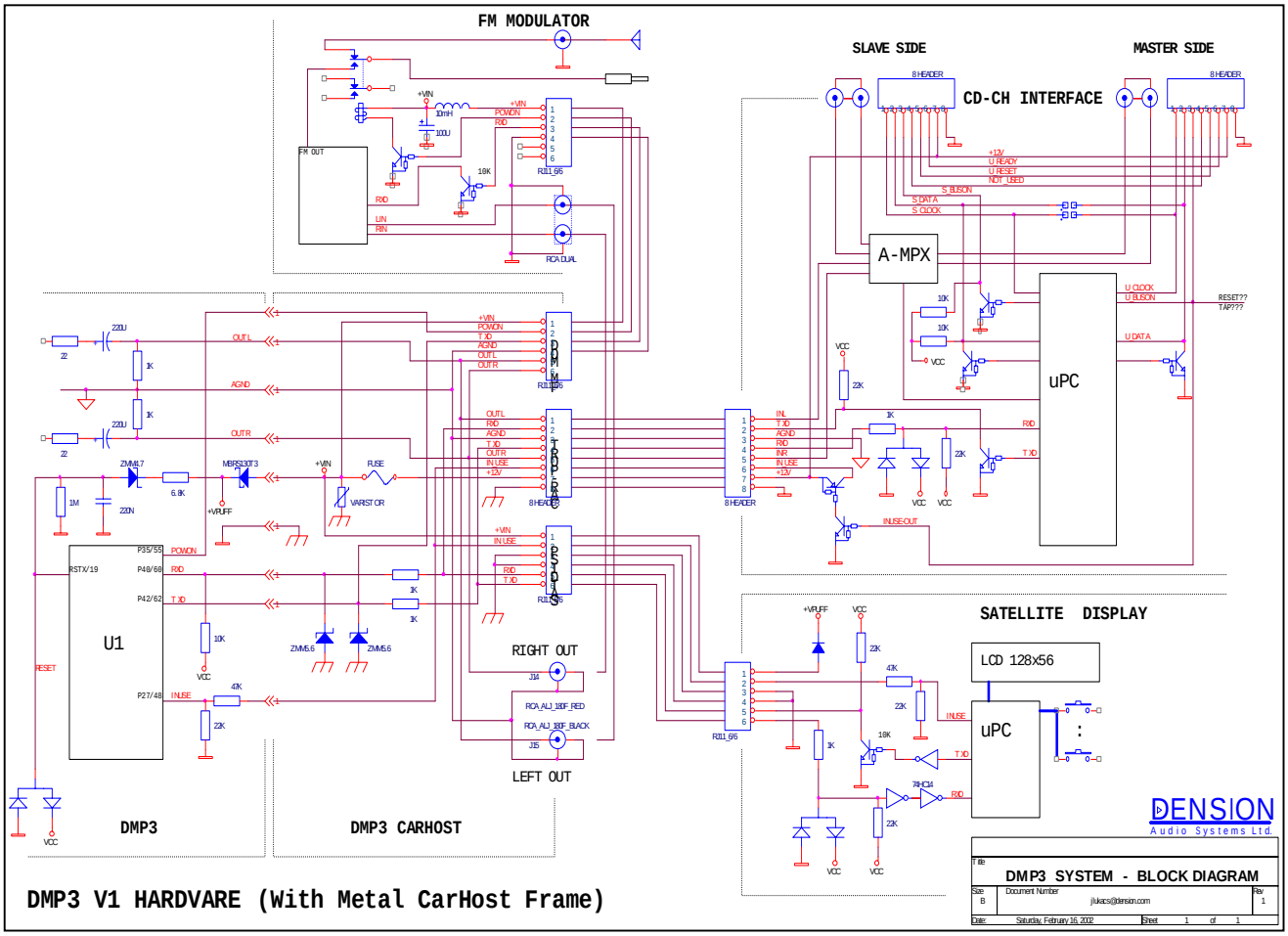
1 +Vin
2 INUSE
3 GND
4 GND
5 RXD
6 TXD

FM modulator

RJ11_6/6 Front view



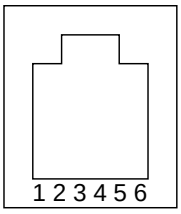
1 +Vin
2 POWON
3 RXD
4 AGND
5 OUTL
6 OUTR



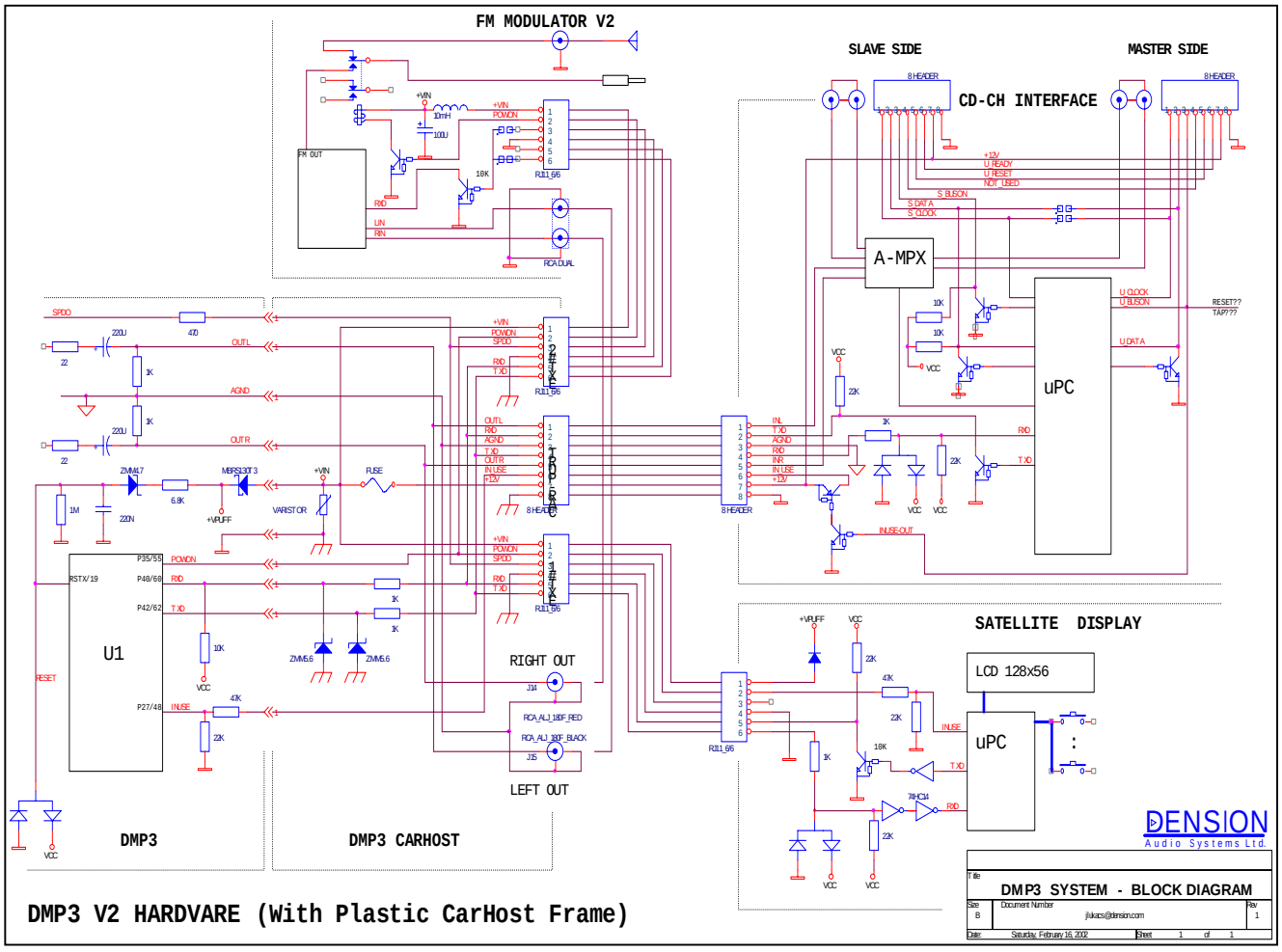
Version V2
Is made with plastic CarHost frame, and two unified EXT# connectors.

EXT#1-2

RJ11_6/6 Front view



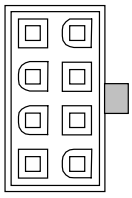
1 +Vin
2 POWON
3 SPDO
4 GND
5 RXD
6 TXD



Identical in the V1 and V2 versions

CarPort connector

Blue	GND		+12 V	Red
Purple	InUse		Out R	Gray
Green	TxD		AGND	Black
Brown	RxD		Out L	White



GND	föld of the power (- pole)
+12 V	Permanent +12V
InUse	Switched +12V (from ignition switch, or from the car Hi-Fi).
RxD	RX data
TxD	TX data
OUTL	Left side sound output (identical with the RCA connector)
OUTR	Right side sound output (identical with the RCA connector)
+Vin	Permanent +12V for the units
POWON	Power On logical output. If the DMP3 is turned on, it is +5V, otherwise 0V (max. 1 mA load)
SPDO	SPDIF digital sound output

Modifications:
- during file transfer information concerning the format of the EXTUPD file are added