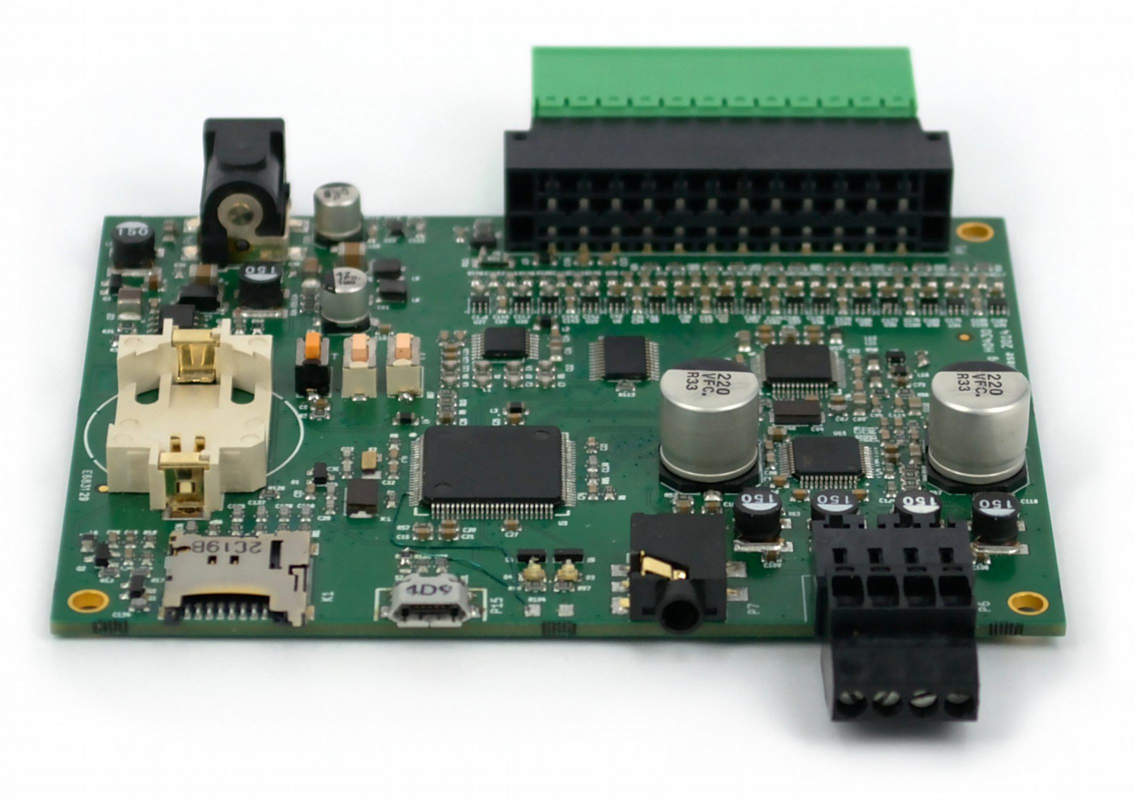
Starling H0430

Programmable Audio Controller with power amplifier



User Guide

ITB CompuPhase
Eerste Industriestraat 19-21
1401 VL Bussum, The Netherlands

Tel.: +31 (0)35 6939261
Fax.: +31 (0)35 6939293
info@compuphase.com
www.compuphase.com

2016-06-08

Contents

Preliminary

Anti-static handling precautions

Please observe anti-static handling precautions when handling the device, as it contains components that are sensitive to static discharges.

Power supply voltage range

Please verify the power supply voltage to be within the range that the device supports, and that the polarity is correct. See page 11 for the specifications and page 4 for the power connector. Voltages outside the appropriate range may cause malfunction. Voltages above the high limit or with inverted polarity may damage the device.

Legal disclaimer

ITB CompuPhase shall not be liable for the incidental or consequential losses or damage to tangible property, injury or death of a person in connection with the use of this device. Although the examples in this guide have been tested with care, they may contain errors and they are not guaranteed for any particular purpose.

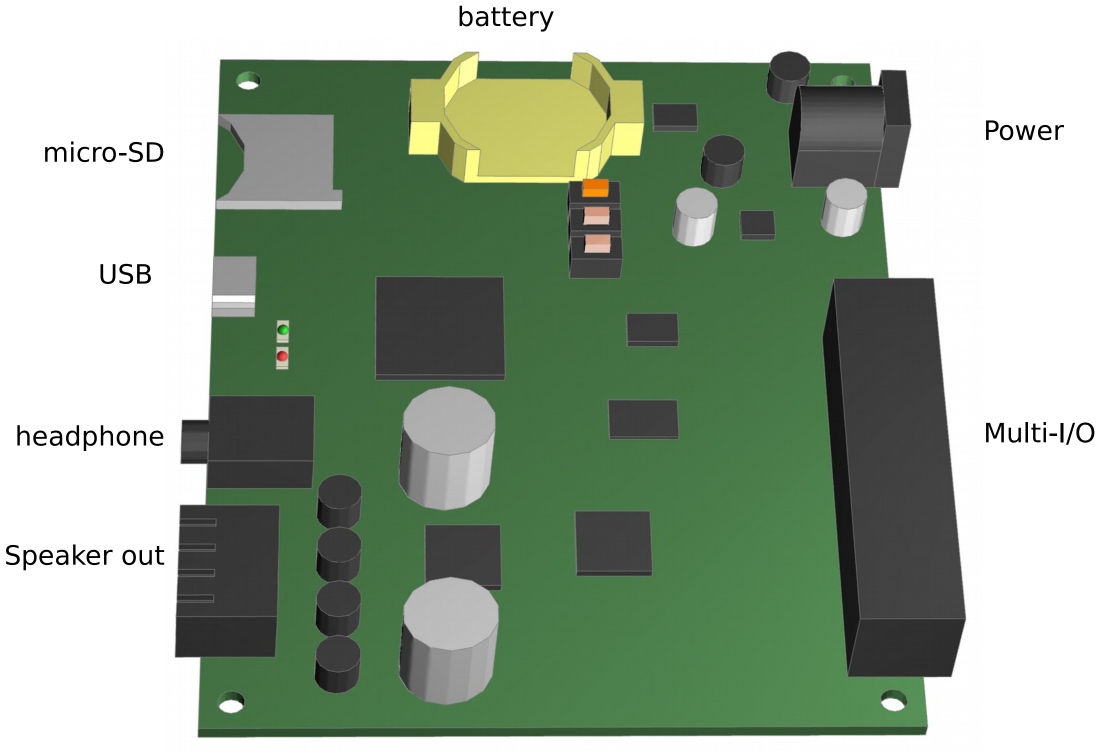
Overview of the Starling, model H0430

The Starling is a compact and solid state general purpose controller with a digital-audio decoder that runs under the control of a scripting language. The audio fragments and the script that controls the operation and behaviour of the audio player are stored on a micro-SD card. The audio outputs are a headphone output, also suitable for line-out level, and a 2 × 15 Watt speaker output. Auxiliary devices, such as (electronic) switches or data terminals may be attached through an 16-bit general purpose I/O bus, an SPI bus, or a dual RS232 line.

This booklet introduces the scripting capabilities of the Starling audio controller. For technical details and a programming reference, please see the documentation that is provided in Adobe Acrobat format on the CD-ROM that comes with the product.

The script controls the operation of the Starling controller and the peripherals. The script must be written in the PAWN language and be stored (in compiled form) on the micro-SD card. In absence of a script, the Starling plays all audio files on the micro-SD card. To react on switches or I/O pins and to communicate over the network interface, or drive other peripherals, a PAWN script is required.

Connecting the Starling, model H0430



Micro-SD card

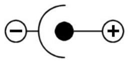
The micro-SD card connector is of the push/push type. To insert a card, push it in until it clicks and then release it. To remove a card, push the card again until it clicks and release it; it will then be ejected.

The memory card connector accepts SD cards, SD/HC cards and MMC cards. The memory cards must use the standard FAT file system (both FAT16 and FAT32 are supported). The firmware supports subdirectories, encrypted volumes (partitions) and encrypted audio tracks, using strong encryption with a 128-bit key.

The micro-SD card must contain all audio tracks. Typically, it will also contain a programmed “script” that determines how the device responds to various inputs —for example, by playing different audio tracks for each switch that is pressed.

Power connector

The power connector is a standard low voltage power connector with an outer diameter of 5.5 mm and a pin size of 2.1 mm. The pin is the plus pole.



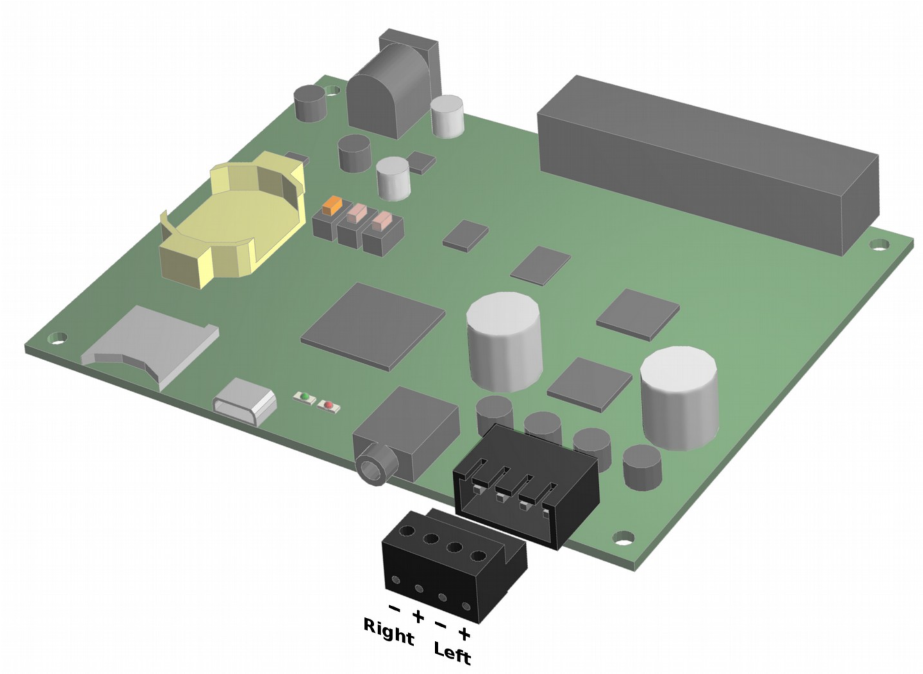
The operating voltage must be in the range of +7.5 V DC to +28 V DC.

Headphone / line-out output

Line-level audio output is present on a 3.5 mm TRS (“jack”) socket. It is also suitable for a headphone. The minimum impedance is 16 Ω. The recommended impedance is 30 Ω for headphones and 600 Ω for connection to line-in (e.g. of an amplifier).

Speaker output

A plugable terminal block provides stereo audio output with maximum 15 Watt per channel.



Minimum speaker impedance is 4 Ω. Adhere to the picture above for phase polarity.

USB connector

The Starling can be connected to a PC or workstation using a USB interface. The USB interface provides file transfer to and from the memory card, script debugging, and interfacing with applications on the PC.

The Starling identifies itself on the PC as a virtual serial device (CDC class, ACM subclass). The Starling can use the standard driver for virtual serial ports that most operating systems provide.

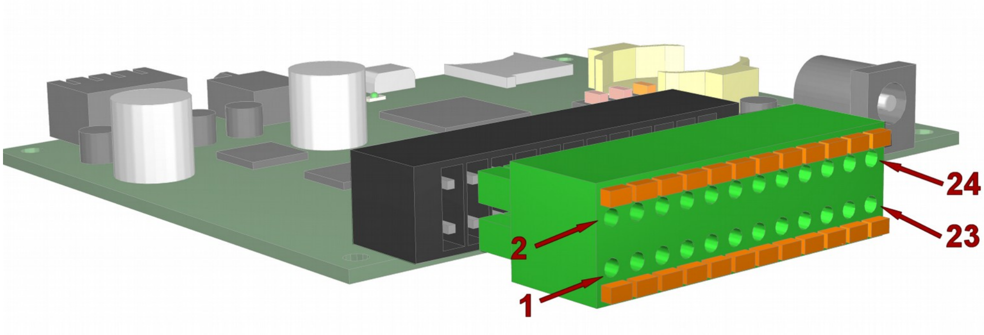
Battery connector

The battery connector uses a common 3 V CR2032 battery. The battery is used for keeping the time while the board is powered off and for powering a small region of memory, so that settings are remembered when the power is off.

If you do not need clock functions and/or storing settings in the device's memory, the battery is optional. Note that a script can also store device settings on the micro-SD card.

Multi-I/O connector.

The multi-I/O connector is a plugable terminal block with spring-cage wire terminals. The connector provides digital I/O pins for switches, relays, LEDs or TTL logic. In addition, an SPI serial bus and a dual RS232 port.



1	General I/O pin 0	2	General I/O pin 1
3	General I/O pin 2	4	General I/O pin 3
5	General I/O pin 4	6	General I/O pin 5
7	General I/O pin 6	8	General I/O pin 7
9	General I/O pin 8	10	General I/O pin 9
11	General I/O pin 12	12	General I/O pin 11
13	SPI MOSI / general I/O pin 14	14	SPI SSEL / general I/O pin 13
15	SPI MISO / general I/O pin 16	16	SPI SCK / general I/O pin 15
17	Analogue output	18	Ground
19	Power (fused at 400 mA)	20	Ground
21	RS232 port 2 TxD	22	RS232 port 2 RxD
23	RS232 port 1 TxD	24	RS232 port 1 RxD

General purpose I/O

The 16 general purpose digital I/O pins can be configured as input or output. The I/O pins are EMC and ESD filtered, and slew-rate limited to 1 MHz. I/O pins 12 through 15 are multiplexed with SPI functionality; they can be configured as either SPI or digital I/O.

When the I/O pins are configured as inputs, the pins are 5 V-tolerant. I/O pins 0 through 11 have a pull-up resistor of 10 kΩ; I/O pins 12 through 15 have no pull-up or pull-down. The pins 0 through 7 are suitable for low-latency digital I/O after disabling the debounce filter; the pins 8 through 15 have a fixed debounce filter (which cannot be disabled). When the debounce filter is active on a pin, the maximum input frequency is 50 Hz.

When the I/O pins are configured as outputs, the high-level voltage for the pins is the configurable external voltage, which is 3.3 V by default. Each output pin has a series resistor of 120 Ω for current limiting and short-circuit protection; they can directly drive a LED (or a similar load, such as an opto-coupler).

The analogue pin is output-only; it has a range from 0 V to the external voltage. The analogue output pin is limited to 100 kHz.

RS232 (serial communication)

Two serial ports are available on the multi-I/O connector. The signals on these pins have the standard voltage RS232 levels for serial communication. All common Baud rates are supported (1200, 2400, 4800, 9600, 19200, 28800, 38400, 57600 or 115200); special Baud rates, such as 31250 for MIDI, are supported as well. The number of data bits and stop bits, and the parity is configurable in the user script.

The RS232 interface supports the XON/XOFF protocol (software handshaking), but no hardware handshaking protocols. The use of a handshaking protocol is configurable.

SPI (serial communication)

The voltage levels of the SPI signal pins is the configurable external voltage, which is 3.3 V by default. The maximum SPI clock frequency is 8 MHz. The SPI lines are shared with the general purpose I/O pins 12 through 15.

Power out

The power output has a self-resetting fuse for short-circuit protection. The maximum current that can be drawn from the power pin is 400 mA. The power output must be enabled in the user script (it is disabled by default). The voltage (if enabled) is 3.3 V by default, but can be configured between 2.5 V and 5 V. This is the “external voltage” also used by the I/O pins and the SPI pins.

On-board switches, adjusting amplifier gain

The Starling model H0430 has three on-board switches: RESET, FUNC and AUX. The RESET switch is hardwired to reset the Starling; it is not available for any other purpose. The FUNC and AUX buttons adjust the amplifier gain, but they are also available for user scripts.

In a default configuration, the FUNC switch increases the gain of the power amplifier and the AUX switch decreases the gain of the amplifier. Note that this adjusts only the volume at the speaker outputs; the volume of the headphone/line-out is *not* changed.

The amplifier gain set by the FUNC and AUX switches is stored in non-volatile memory. Therefore, the gain settings are retained even after a power-cycle.

In case that the user script sets the amplifier gain, the FUNC and AUX switches no longer adjust the amplifier gain. The user script can set both amplifier gain and the overall volume (for headphone/line-out as well as speaker outputs) and it can react on the FUNC and AUX switches.

Mechanical montage

Four mounting holes with a diameter of 3.1 mm (suitable for M3 screws or 1/8” screws) are spaced at 93 mm horizontally and 73 mm vertically. Ensure at least 6 mm of clearance below the PCB. See the datasheet for details and drawings.

The mounting holes are electrically isolated from the digital and analogue ground of the device.

Development environment and user-scripts

A full development environment for user-scripts is freely available on the CompuPhase web site. It is also included on the CD-ROM that comes with the product. The development environment contains:

- Several ready-to-run and fully documented scripts that can be configured through an “INI” file and a “playlist” file. No programming is required to run these scripts.
- The PAWN development tools: the compiler, the debugger and configuration files.
- An assortment of example scripts in source code form, to get you starting with programming the Starling audio controller.
- Auxiliary utilities, such as a serial monitor program.
- Extensive documentation in Adobe Acrobat format: the PAWN language manual, the Starling programming reference, and the datasheet with detailed electrical and mechanical information.
- Example MP3 tracks, for testing purposes.

Installing the development environment

On Microsoft Windows, run the application called “Setup” on the CD-ROM or the self-installing setup downloaded from the CompuPhase web site. Then follow the directions of the installation utility. The installer allows you to choose various optional components for installation.

The PAWN compiler is also available for Linux, on the CD-ROM and on-line. For installation under Linux, please refer to the documentation of the development environment.

Using the ready-made scripts

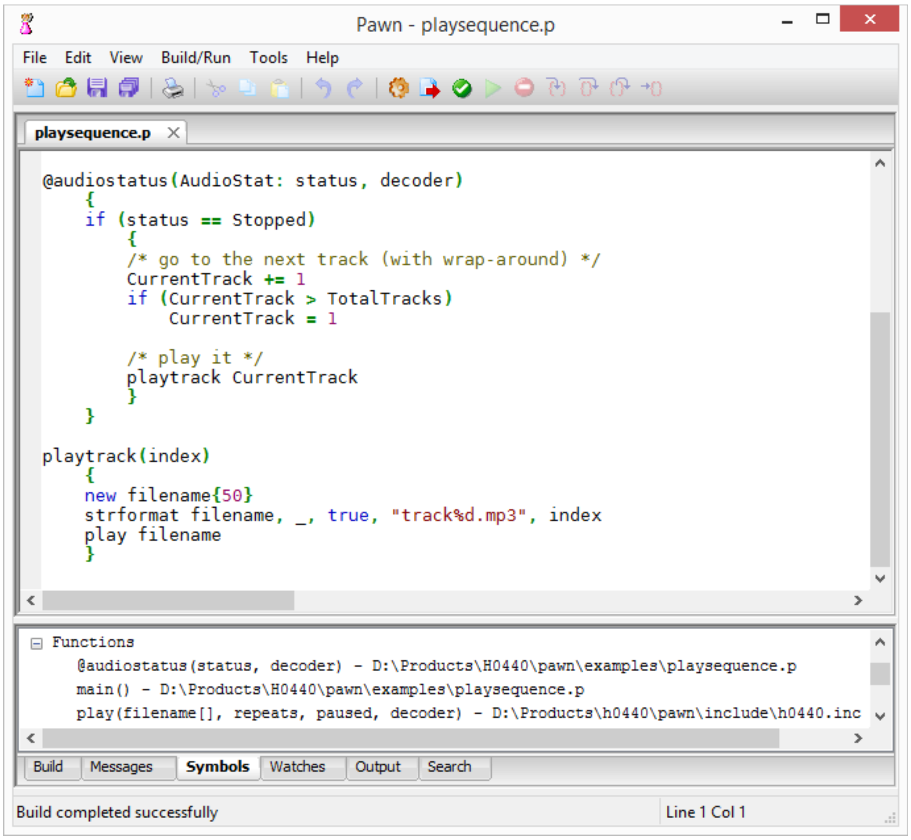
You may first want to try whether one of the standard, ready-made scripts fits your requirements. The ready-made scripts are documented in HTML pages, and links to these pages appear in the “Program Files” menu (after installing the development environment).

The ready-made scripts read the audio tracks to play from a playlist that conforms to the WinAmp M3U specification. The scripts may optionally be configured through a standard text file in “INI” file format. All configuration options are described in the aforementioned HTML pages.

The ready-made scripts may also serve as more advanced examples for coding your own scripts. Each of the ready-made scripts comes with the full and commented source code. To use the scripts, no programming or build step is required, because the scripts come in a form that is directly usable by the Starling controller (“pre-compiled”).

Writing scripts

Scripts for the Starling are in the PAWN language. To create the scripts, you use the “Integrated Development Environment” (IDE) that is part of the PAWN development tools. When saving the script's source code, we suggest the extension “.p” or “.pawn”.



```
@audiostatus(AudioStat: status, decoder)
    {
    if (status == Stopped)
        {
        /* go to the next track (with wrap-around) */
        CurrentTrack += 1
        if (CurrentTrack > TotalTracks)
            CurrentTrack = 1

        /* play it */
        playtrack CurrentTrack
        }
    }

playtrack(index)
    {
    new filename{50}
    strformat filename, _, true, "track%d.mp3", index
    play filename
    }
```

```
⊟ Functions
    @audiostatus(status, decoder) - D:\Products\H0440\pawn\examples\playsequence.p
    main() - D:\Products\H0440\pawn\examples\playsequence.p
    play(filename[], repeats, paused, decoder) - D:\Products\h0440\pawn\include\h0440.inc
```

Build completed successfully

Line 1 Col 1

The IDE for Pawn

A minimal script, that plays one file on power-on and then stops, is:

```
@reset()
    {
    play "track1.mp3"
    }
```

The Starling controller does not interpret the source code of the script directly, but it executes a compiled form of the script. You can compile directly from the IDE, or use the separate PAWN compiler in a console window or “DOS box”.

The PAWN compiler generates a binary file with the extension “`.amx`” from the input file. The next steps are to copy this “`.amx`” file to the micro-SD card and to insert this card into the Starling. In the IDE you can also adjust the output path so that the source file gets compiled directly onto the micro-SD card.

Typically, a script must do a lot more than play just a single file. At the very least, it should allow to re-start playing one or more audio files without needing to power-cycle the device. More commonly, the script must react on (user) input in the form of switches or digital levels from sensors. Please look at to the example scripts for various interfacing techniques.

As stated above, compiling the script creates a new file with the extension “`.amx`”. This file needs to be copied to the micro-SD card. It can be in one of two locations:

- a directory called “`system`” on the micro-SD card.
- the root directory of the micro-SD card

The Starling first checks the root directory and, if no compiled script is found in the root, then checks for a script in the “`system`” directory.

If you have multiple scripts on a card, the compiler script that must run on power-up (or after reset) should be called “`startup.amx`”. This script can also be in either the root directory or the “`system`” directory.

Audio tracks

The Starling supports the following audio file formats:

- MPEG version 1 layer 3 (“MP3”), as well as MPEG version 2 and 2.5 layer 3;
- Ogg Vorbis;
- Uncompressed “WAV” files (RIFF header), 8-bit and 16-bit;
- ADPCM files with a RIFF header;
- Optionally HE-AAC (AAC LC with SBR), level 3; this option requires a license.

Track duration & seeking to a position

Within the MP3 file format, there are the encoding varieties “constant bit rate” (CBR), “variable bit rate” (VBR) and “average bit rate” (ABR). VBR and ABR files must have a “Xing” header to support “direct jumps” to a specific position in the MP3 track. Without this header, jumps to a position may be inaccurate for VBR and ABR files and the reported track duration may be inaccurate as well. Reported durations and direct jumps are always accurate for CBR files.

Reported durations and direct jumps are also accurate for uncompressed PCM files (“WAV” files).

Ogg Vorbis files are always “variable bit rate” and no meta-header with information on the full file is defined. The Starling therefore relies on the APE tag for the duration of the track. Jumps to a position in an Ogg Vorbis track may not be accurate.

APE Tags, ID3 Tags and synchronized events

The scripting language gives the programmer access to the information in an APE or ID3 tag (version 2), with information on the artist, album & track, full title, copyright and cue/segue times.

The supported frames in the APE and ID3 tags are:

ID3 frame	APE frame	Description
TIT2	Title	Song title
TPE1	Artist	Performing artist
TALB	Album	Album title
COMM	Comment	Comment
TCOP	Copyright	Copyright
TSRC	ISRC	ISRC data, a code to identify sound/music recordings
TLEN	Duration	Duration of the audio track in milliseconds
SYLT		Synchronized lyrics
	Cue	Cue time in milliseconds (silence at the start of the track)
	Segue	Segue time in milliseconds (silence at the end of the track)

The APE and ID3 tags both contain similar data, but both also have fields that are absent from the other. The ID3 tag supports events to be synchronized to time-stamps inside the track (“synchronized lyrics”) and the APE tag provides fields that are useful for the purpose of cross-fading.

Adding “synchronized lyrics” to an MP3 file

In addition to the standard “authoring” information, the Starling supports the optional “synchronized lyrics” frame in the ID3 tag. The contents in this frame will cause scriptable events to be “fired” on the time stamps that are attached to each line of lyrics. Although the synchronized lyrics frame was originally designed for purposes of karaoke and sub-titling songs, another use would be to tag time-stamped cues or commands at specific positions in a song or audio fragment.

The first step in making a script react on synchronized events in an MP3 file is to attach an ID3 tag with the “SYLT” frame to that file. A suitable SYLT editor is freely available from the CompuPhase web site.

Synchronized lyrics is a set of plain text strings where each string has a time-stamp, in milliseconds from the start of the audio track. After having added the public function “`@synch()`” to your script, this function will be invoked (or called) at the precise time-stamps of each string and with the text of the string as a parameter.

For purposes of Karaoke, the text strings will indeed be the lyrics of the piece. For other applications, the time-stamped text may consist of strings like “Light #1 off”, or “Next slide”. It is up to your script to interpret the text strings that it receives.

Encrypted MP3 tracks

The Starling supports MP3 tracks that are encrypted with 128-bit device key, plus an optional 32-bit user key. The device key is hard-coded in the device; only Starling units with the correct key can play back the encrypted tracks. When ordering a series of MP3 players, a customer may request a unique device key for that series. In addition to the key, the customer also receives a programs for encrypting MP3 tracks.

ID3 tag information (version 2) is preserved and non-encrypted. However, the ID3 tag should not be modified after encrypting the file.

Troubleshooting

If the device does not run correctly, please first try these few tests.

As a first test, remove power, disconnect all cables and devices attached to the Starling and remove the micro-SD card. Then (re-)connect the power and observe the two LEDs.

- The green LED should stay on. If it remains off, please check the power and the polarity of the power. See page 11 for the voltage level and page 4 for the polarity. If the green LED blinks, there is a problem with the firmware, please contact us for a replacement or option to restore the firmware.
- The red LED should blink roughly twice per second.

After passing the above test, create a micro-SD card with only MP3 tracks in the root directory (delete all script files and other files with a “.amx” extension). It is best to format the card first and then put a few MP3 tracks on it (at least, verify that you can read this micro-SD card on a PC and that it uses the FAT16 or FAT32 file formats). Also connect a headphone to the 3.5 mm TRS socket.

- After inserting the card, audio should start playing. The red LED indicates read access to the micro-SD card, it typically blinks so fast that it appears full on, but dimmed. The green LED stays full on. Audio should sound through the headphone.

If these tests pass, you can now copy one of the pre-compiled scripts onto the micro-SD card. It is suggested that you first copy the “Play playlist” example, which is a complete example of a simple script that just plays tracks in the order that they are specified in the playlist file.

Specifications

General

Operating voltage	+7.5 V DC to +28 V DC.
Power consumption	1.35 Watt to 32 Watt, depending on peripheral use.
Dimensions	100 mm × 80 mm.
Weight	0.070kg.

Audio

Supported formats	MPEG layer III (“MP3”); Ogg Vorbis; 8-bit & 16-bit RIFF WAV; MIDI format 0; AAC (AAC support requires a license).
Frequency response	20 Hz to 20 kHz, ±0.5dB.
Dynamic range	90 dB (typical).
Distortion+Noise	THD+N < 0.13% at 1 kHz 1 Watt (power amplifier); THD+N < 0.1% at 1 kHz (headphone/line-out).
Noise	SNR 93 dB at 1kHz +0 dBu (typical).
Speaker impedance	4 Ω minimum, 8 Ω nominal.
Headphone impedance	16 Ω minimum, 30 Ω nominal for headphones, 600 Ω nominal for line-out.
Max. output level	2 × 15 Watt (power amplifier).

Integrated peripherals

Temperature sensor	Measures -40 °C to +125 °C; typical accuracy 0.5 °C.
Real-time clock	Inaccuracy < ±1.7 seconds per 24 hours.
Pseudo-random numbers	Cycle length 2^{32}, initialized with true a random seed.
RS232 interfaces	Two interfaces conforming to 3-wire RS232 ports; standard signal levels.
SPI interface	Standard SPI interface, with programmable voltage level and two slave select lines
USB	Full-speed device, identified as CDC/ACM class.
Digital I/O	16 general-purpose I/O pins, with programmable voltage level (2.5 V to 5.0 V); max. 40 mA source or sink; all pins are ESD/EMC filtered.
Analogue out	1 software-controlled analogue output pin.

Operating conditions

Temperature	Designed for -40 °C to +85 °C.
Humidity	5% to 95% non-condensing.
Vibration	Full solid-state device, no moving parts.

Conformity

EMC (emission/immunity)	Compliant with EU Directive 2004/108/EC: EN 55022 and EN 55024 + A1 (2001) + A2 (2003).
Electrical safety	Compliant with EU Directive 2006/95/EC: EN 60950-21
RoHS	Compliant with EU Directive 2002/95/EC.