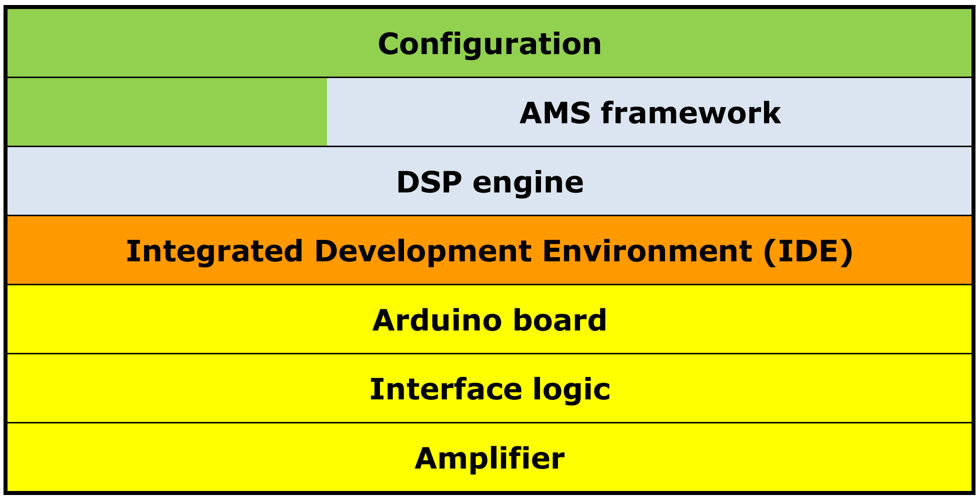
Amplifier Management System (AMS)

Product Description



Version 1.0

December 2015

© Steen Frank Jensen 2015

Contents

© Steen Frank Jensen 2015

1 Introduction

What is Amplifier Management System (AMS)

Amplifier Management System (AMS) is an Arduino based software solution designed to monitor, control and protect power amplifiers and loudspeakers.

With AMS you get full access to source code and documentation and AMS comes with a preconfigured setup that can be used as a starting point and a source of inspiration.

AMS is a software solution directed to professional amplifier developers and amplifier manufacturers. AMS does not include hardware.

By default, AMS measures power supply voltages and output voltages or any other analog value 1,000 times per second and calculate the operating conditions for the amplifier. The operating conditions are tested against predefined limits and if signs of catastrophic faults are detected it closes down within milliseconds.

AMS can also detect less critical over load conditions and turn down volume or close down the amplifier for a limited time. The possible reactions will depend on available features in the amplifier.

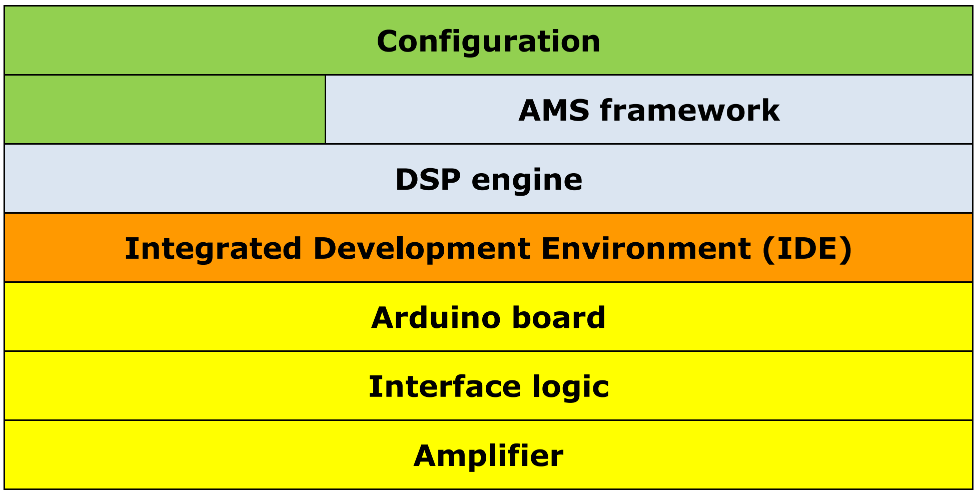
AMS can perform tests even during power up for early detection of faults like defective output transistors or shorted outputs.

AMS is a framework that is highly configurable and with full access to source code it can be extended with new features in order to support individual needs. The final solution is the sum of the framework, the configuration and individual extensions. With the right configuration and the right amplifier, it can be rich in features and facilities.

2 Building blocks

2.1 Block overview

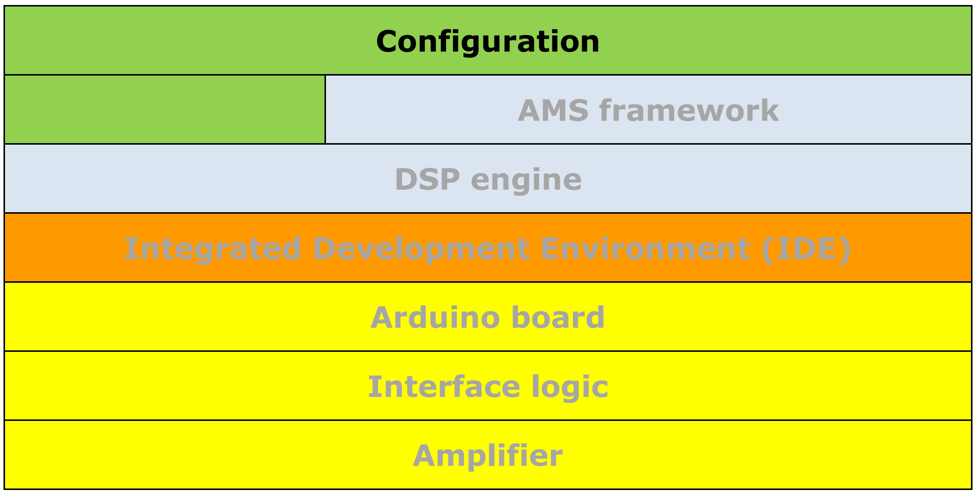
A solution with AMS consists of the following building blocks.



The blue blocks represent AMS.

In the following pages, we will look deeper into the building blocks.

2.2 Configuration



The configuration is the settings, calculation rules and programming statements that makes it possible to adapt AMS to individual needs.

The configuration stands on top of the AMS framework and if needed you have full access to the DSP-engine too.

Preconfiguration

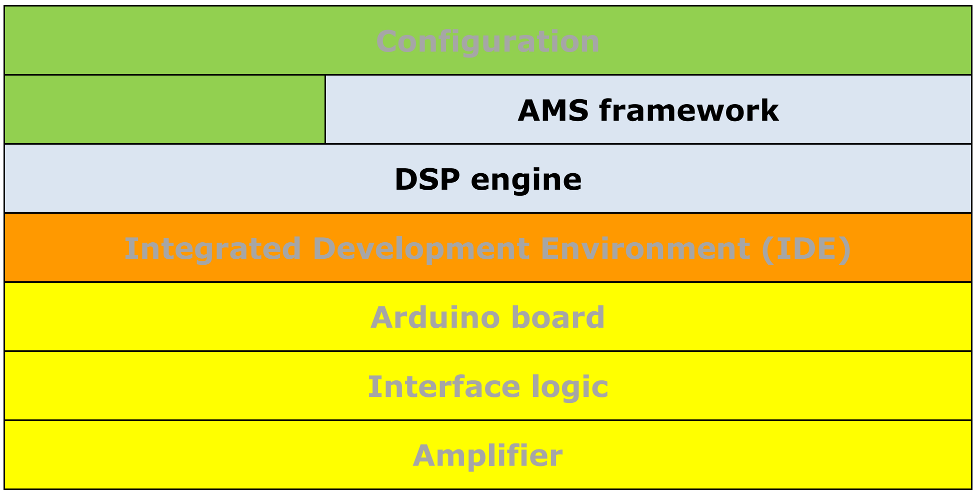
AMS comes preconfigured with a solution that is optimized for a power amplifier with single ended power supply and balanced output in a H-bridge construction.

The preconfiguration is a result of a comprehensive work with balanced power amplifiers. Faults and symptoms have been analyzed and calculations and filters developed in order to detect faults as early as possible during power up.

This preconfiguration is rich in features and can be used as a starting point and a source of inspiration.

You may come a long way if you start with the preconfigured solution and remove what you do not need and what is not possible with your amplifier.

2.3 Amplifier Management System (AMS)



Amplifier Management System (AMS) consists of the blue blocks.

2.3.1 DSP engine

The DSP engine is a software layer that makes it easy to implement Digital Signal Processing (DSP). This includes:

Fast sampling of analog input

DSP requires fast and efficient sampling of analog input which is optimized with interrupt driven AD-conversion.

Timers

In DSP most activities must be performed periodically, controlled by timers. The following timers are available:

1mS, 10mS, 100mS, 1S, 10S, 1Min

Idle loop

The idle loop contains activities that can be performed when the CPU is not busy with other tasks. This is the case most of the time, therefor you may place high priority activities in the idle loop.

The idle loop is executed at least every millisecond.

Real time clock

The real time clock (RTC) measures time since last reset of the CPU.

Events and responses are timestamped as an important feature to support test and debug.

DSP examples

The framework contains low pass filters and de-bounce filters implemented as IIR-filters. For more information, see Resources.

2.3.2 AMS framework

The AMS framework sits on top of the general DSP engine.

The AMS framework is an application specific framework with detailed knowledge of amplifiers. It is the result of a long development period where the program structure has been optimized for speed and flexibility and detailed knowledge of amplifier faults and symptoms has been incorporated.

It is supplemented with support facilities like print routines, LED drivers etc.

The behavior of the framework depends heavily on the configuration that adapts the solution to individual needs. This is done with a mixture of parameters and well-defined containers for individual program code.

Here are some examples of definitions (from the configuration) and aspects that are handled by the framework:

- Power up and Power state
  Definition of power states with names and durations.
  Definition of outputs that turn on relays and other control signals.
  Definition of tests and other activities for each power state in order to detect faults as early as possible.

- Inputs and outputs
  Definition of all inputs and outputs.

- AD-conversion
  Control of AD-conversion including formulas to convert to a relevant physical value like a floating point Voltage.

- Operating parameters like
  Voltage, ampere, power and energy
  Current value, peak value, RMS value
  Measurement of clipping time and over load (saturated transistors)

- Formulas
  Calculation rules for operating parameters.
  Rules can be expressed as simple arithmetic expressions.
  Complicated calculations may include program logic (if..then..else).

- Filters
  Low pass filters for DC detection and de-bounce.

- Auto zero
  Auto zero facilities to improve measurement precision.

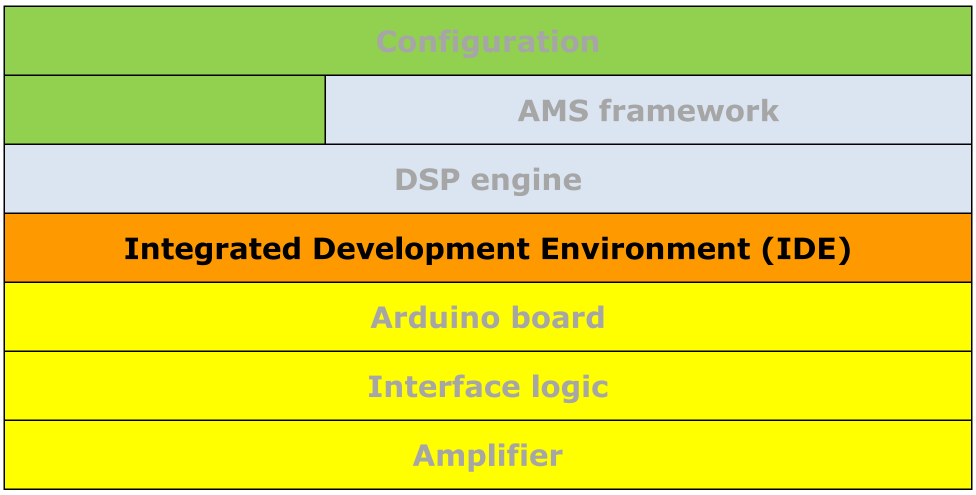
- Limits
  Definition of limits for all operating parameters like maximum allowed power or maximum allowed DC offset.

- Errors
  Definition of errors and error response.

- LED
  Visualization of status via light intensity and blink.

- Print routines
  Several print routines to support test and debug.

2.4 Integrated Development Environment



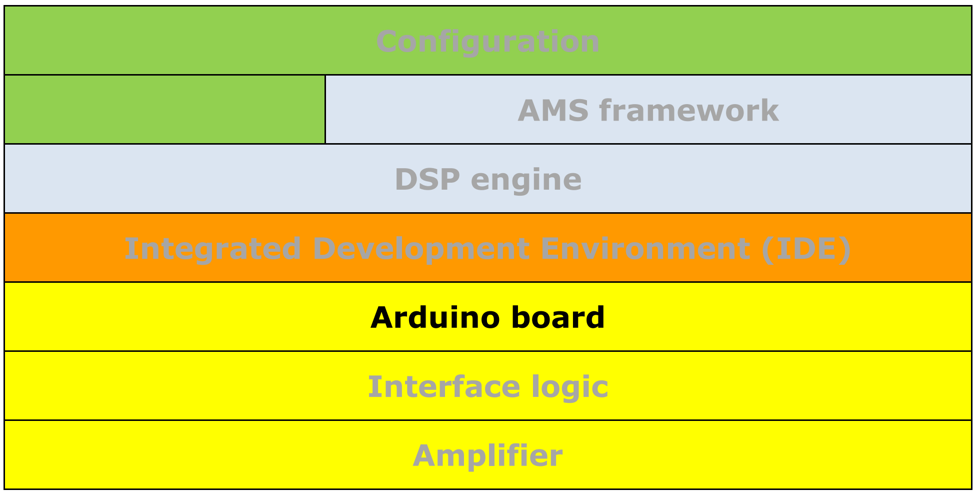
You need a PC with an Integrated Development Environment (IDE) connected to your Arduino board.

The IDE makes it possible to edit the configuration, download software to the Arduino board and test and debug the solution.

The IDE can be the native Arduino IDE or another IDE like Microsoft Visual Studio with Visual Micro. Both have been used during development of AMS.

AMS has been kept as simple as possible but it is not a small system so a minimum of programming experience is required.

2.5 Arduino boards



Arduino is a family of small stand-alone CPU boards with analog and digital inputs and outputs. They are useful as controllers for electronic devices.

There is a number of available Arduino boards with different specifications. Variations include speed, memory size and number of inputs and outputs.

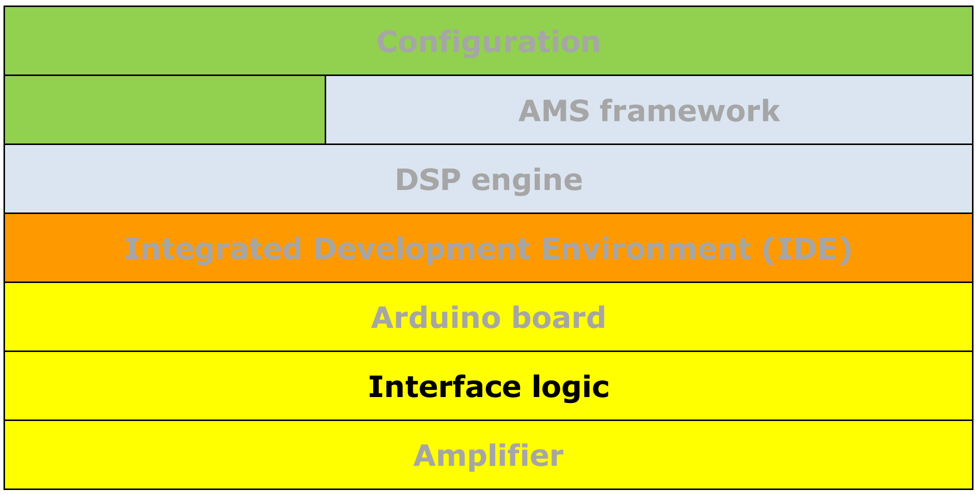
Pls. observe that some boards use 5V logic and some boards use 3.3 V logic. Pls. also observe different input voltages to the AD-converter.

Select a board that is compatible with your requirements.

AMS has been developed and tested with an Arduino UNO R3.

For more information on Arduino and Arduino boards, see Resources.

2.6 Interface logic



Interface logic is the hardware that interfaces between the Arduino board and the rest of the system. This may include:

- Analog input signal conditioning
  Analog voltages or other analog signals can be measured with the build in AD-converters. This will normally require appropriate signal conditioning or attenuation in order to comply with the measurement range of the AD-converter.
  Pls. observe that the Arduino boards have an input multiplexer that represents a capacitive load on the attenuator. To get precise measurements the output impedance of the signal conditioning circuit must comply with Arduino recommendations for the selected board.

- Low pass filters for analog input signals
  By default the AD-converter samples data 1,000 times per second.
  This means that it is unable to handle periodic signals with frequencies above 500 Hz (the Nyquist frequency). For best reliability, all frequencies above 500 Hz should be removed.
  On the other hand AMS is not designed to process periodic signals, instead it is designed to detect faults (aperiodic signals) as quickly as possible, so we want a high fc.
  In praxis, I have had good results with a simple first order low pass filter with fc=1,000 Hz but this is not guaranteed as a general solution.
  For more information on Nyquist and folding, see Resources.

- Analog output conditioning
  Some of the digital output pins can support Pulse With Modulation (PWM) as a simple way to generate analog output.
  This is used to control light intensity for one of the LEDs in the

preconfigured solution.
If your amplifier volume or any other parameter can be controlled by an analog voltage this could be useful. It would typical require some kind of conditioning and filtering.

- Digital inputs
  Digital inputs must be conditioned to the logical levels of the selected Arduino board.

- De-bounce logic
  Some digital inputs like switches or external inputs can be noisy and may need to be de-bounced.
  De-bounce can be done in hardware with a low pass filter and a Schmitt-trigger or it can be done in software.
  AMS is preconfigured with two inputs with software de-bounce.

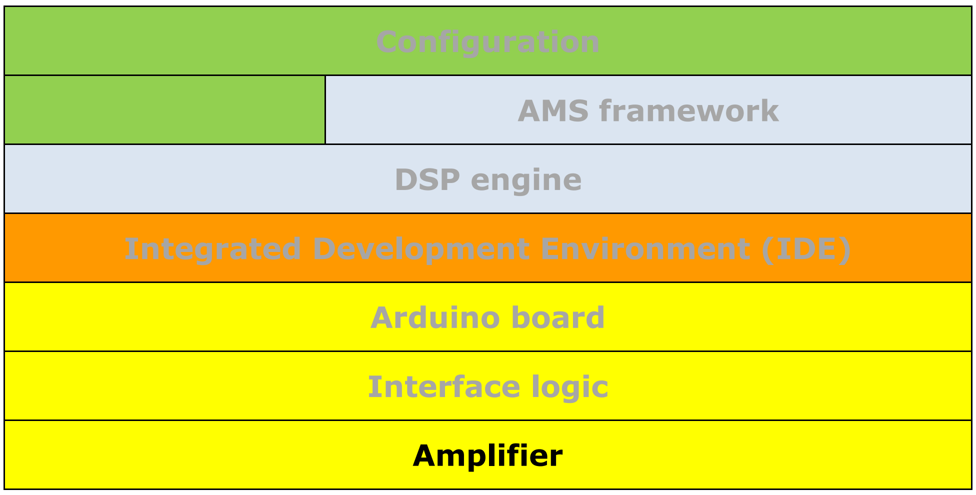
- Digital output
  Digital output can be used to drive relays or other external devices. This may require a driver that converts between the logical levels on the board and the external devices. This can be discrete transistors or integrated circuits.
  Pls. observe that some drivers like common emitter transistors invert the signal.

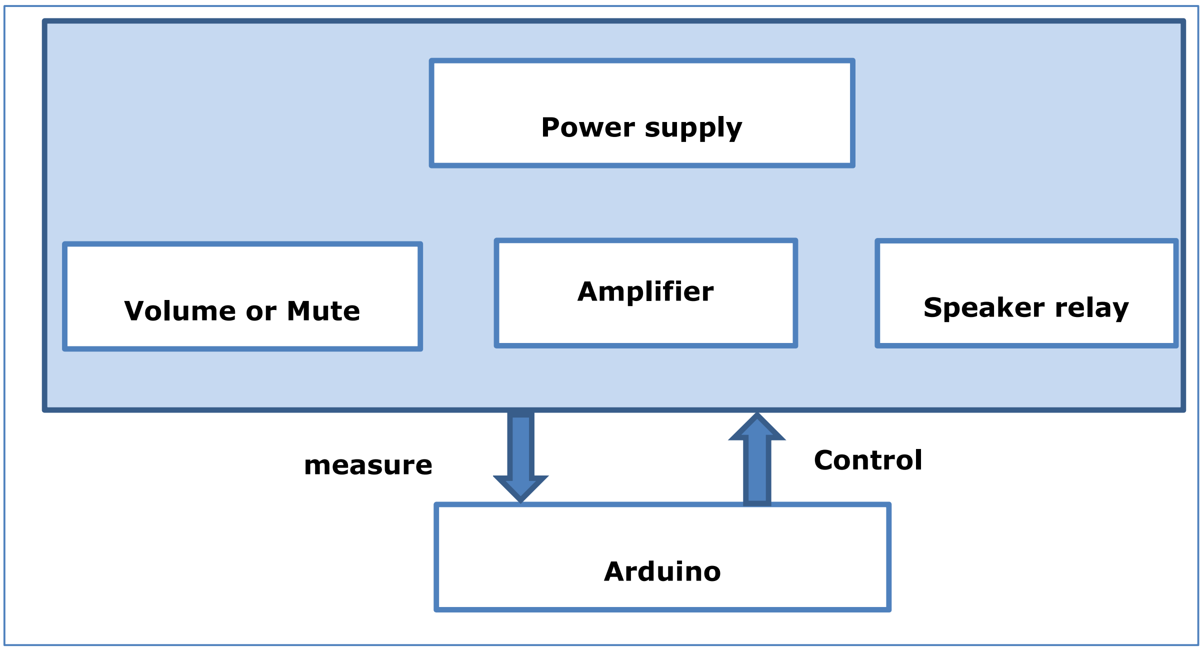
- LED drivers
  Some Arduino boards can drive a LED directly through a resistor.

- Fast reaction
  With analog input your reaction time will be limited by the sampling rate (1,000 times per second). If you want faster reaction to a critical signal like over current you may use a digital input.
  The total reaction time will depend on the amplifier components. Relays will take 10-15 mS to respond, electronic devices are much faster. The ICEpower amplifier module used during development can be closed down in just 1 uS.

- Active high or active low
  For all digital signals to and from the Arduino board you should specify if the signal is active high or active low. Active high is default.
  If a signal is active low, this can be compensated for in the software.

2.7 Amplifier





This figure illustrate the amplifier and some of its capabilities. Here are some important aspects of the amplifier to consider:

- Amplifier type
  Amplifiers can have balanced or unbalanced output and they can have symmetric or single ended power supplies.
  The behavior of the amplifier especially during power up varies with the amplifier type. Detailed understanding may be important.

- Early detection
  In general you should design a power up sequence and tests during power up that gives the best protection in all states and that detects faults as early as possible.

- Mute
  If input can be muted you will know that output is a result of the amplifier itself and not a result of the input signal. This can be used for easy detection of faults during power up.

This is just a few design considerations, more information can be provided as part of AMS.

2.8 Programming standards

AMS is based on the following programming standards:

Programming language	AMS is written in Arduino C, which should be portable across the Arduino boards. The solution has been tested on Arduino UNO R3.
Programming guidelines	AMS is programmed according to Arduino guidelines with focus on a simple programming style and clear documentation.
Optimized AD-conversion	AD-conversion is a slow process with the Arduino standard input function. It has been optimized with interrupt driven AD-conversion. This makes it possible to sample analog inputs 1,000 times per second with a minimum of CPU-load. Interrupt driven AD-conversion (50 lines of code) is bound to a specific CPU (ATmega328P) and is not portable.

3 Preconfigured solution

AMS comes with a preconfigured solution that is optimized for a power amplifier with single ended power supply and balanced output in a H-bridge configuration.

The amplifier can be muted which is important for early detection of faults.

The configuration is rich in features and can be used as a starting point and inspiration.

Pins for input/output

Preconfigured with

- 5 analog inputs
  Vpp
  4 Speaker terminals

- 4 digital inputs
  Extern mute
  Extern trigger
  Over current (L+R)

- 8 digital outputs
  4 Power control relays
  Mute
  Standby
  2 LED

Operating parameters

The following operating parameters are measured and calculated every mS and monitored with limits and appropriate error responses.

- 11 Operating parameters are monitored with current and peak value.

- Energy is calculated per
  Second
  10 seconds
  Minute

- Clipping detectors (saturated output)
  Duration of a single clipping event.
  Percentage of time clipped the last second.
  Percentage of time clipped the last minute.

- First and second order low pass filtered difference voltages across speaker terminals.

- First order low pass filtered sum of voltages across speaker terminals.

Power states

Power states are designed to support early detection of faults and include an auto zero state for improved precision.

- This required 7 power states.
- Each power state controls 5 digital outputs.
- Each power state includes one or more tests.

LEDs

Status is visualized with LEDs:

- Blue LED visualizes power state with increasing intensity.
- Red LED visualizes errors with blinks.

AD-conversion

5 voltages are sampled every mS.

- Vpp
- 4 Speaker terminals

De-bounce

Two external inputs are de-bounced:

- Mute
- Trigger

Auto zero

Preconfigured to optimize measurement of voltages across speaker terminals.

Idle loop

Idle loop gives the quickest response. It is used to monitor over current

© Steen Frank Jensen 2015

4 Resources

Links

Wikipedia – Digital signal processing (DSP)
https://en.wikipedia.org/wiki/Digital_signal_processing

Wikipedia – Infinite impulse response (IIR)
https://en.wikipedia.org/wiki/Infinite_impulse_response

Nyquist frequency and folding frequency
https://en.wikipedia.org/wiki/Nyquist_frequency

Arduino homepage
https://www.arduino.cc/en/reference/homePage

Arduino products (USA only)
https://www.arduino.cc/en/Main/Products

Arduino boards (Genuino) (outside USA)
https://www.arduino.cc/en/Main/GenuinoProducts

Arduino Uno R3.
https://www.arduino.cc/en/Main/ArduinoBoardUno

Arduino Integrated Development Environment (IDE)
https://www.arduino.cc/en/Main/Software

Microsoft Visual Studio with Visual Micro (IDE)
https://visualstudiogallery.msdn.microsoft.com/069a905d-387d-4415-bc37-665a5ac9caba