UNIVERSITEIT iYUNIVESITHI STELLENBOSCH UNIVERSITY 100 1918 · 2018

forward together · saam vorentoe · masiye phambili

Computer Systems / Rekenaarstelsels 245

Lecture 25

Microcontroller Audio/ Mikrobeheerder klank

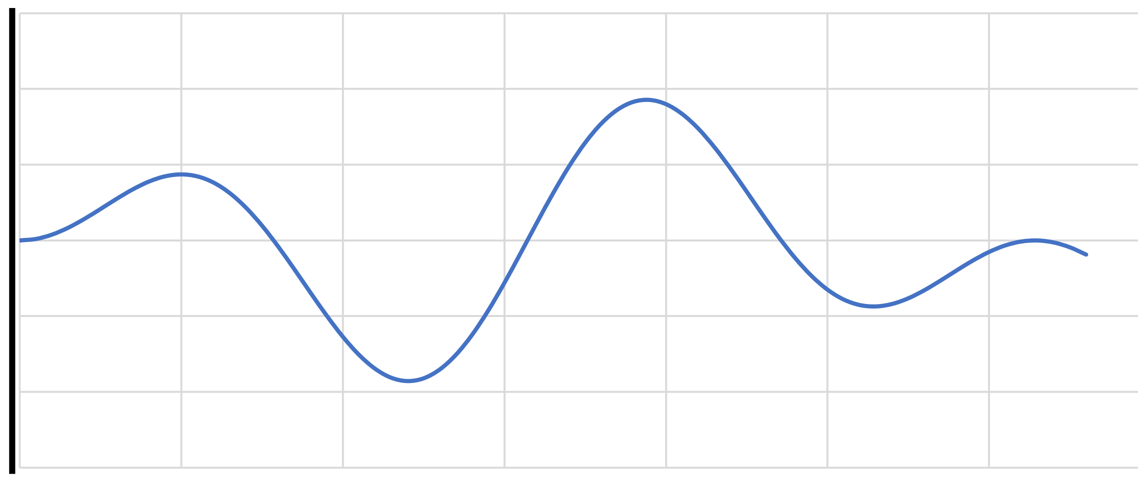
Dr Rensu Theart & Dr Lourens Visagie

Contents
Inhoud

- Audio signals and PCM
- Dev board audio capability
- Emulator audio capability
- Audacity
- Playing sounds in the emulator

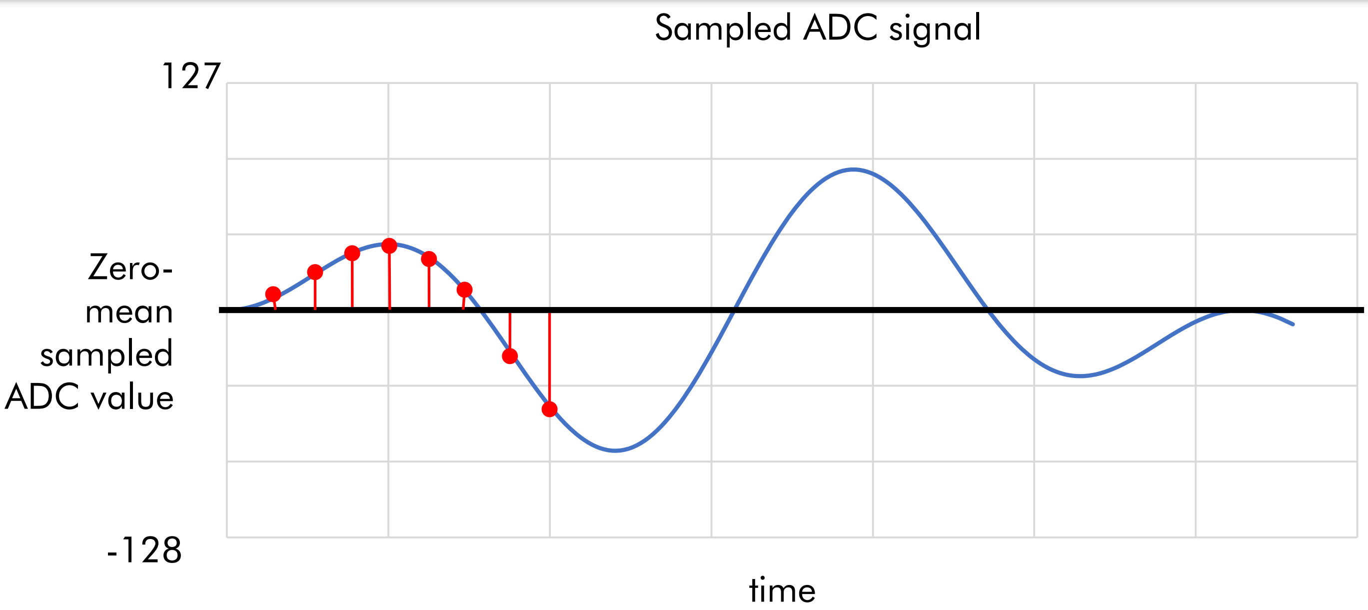
Audio signals
Klank siene

MICROPHONE



- Sound is vibration that propagates as a wave through a medium (i.e. the air around you)
- Audio signal is the representation of the sound as a level of analog voltage

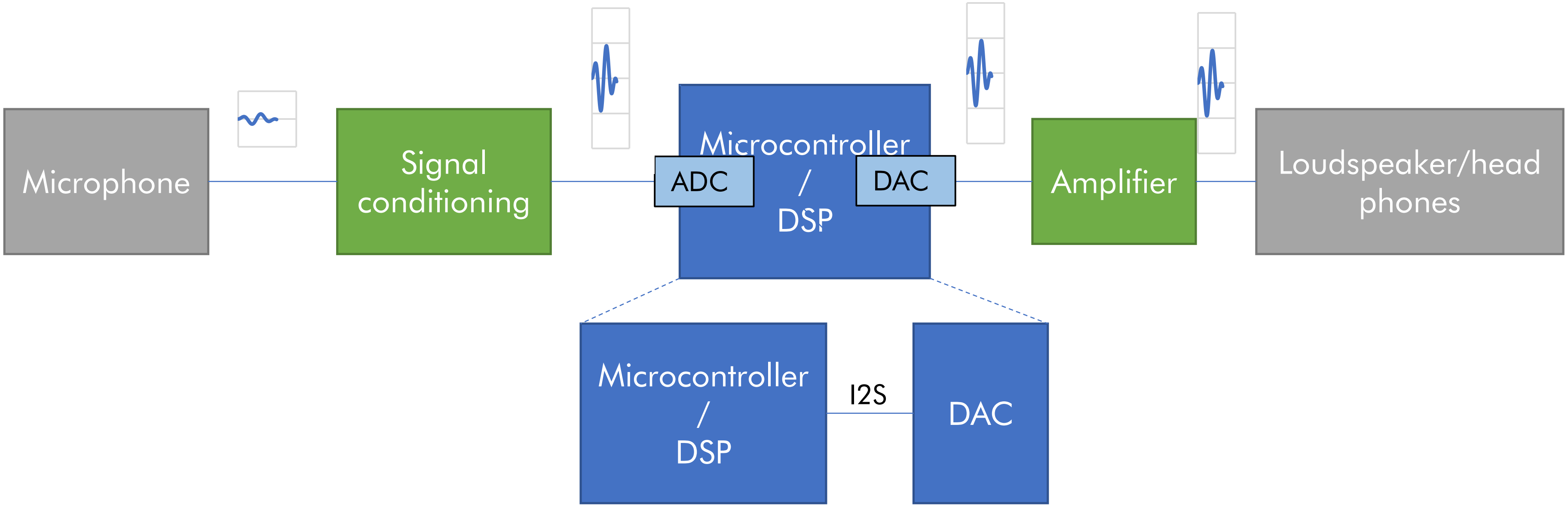
PCM Audio
PCM Klank



- PCM (Pulse Code Modulation) Audio: stream of ADC sampled audio data, with zero-mean.
- 0, 10, 25, 33, 40, 31, 8, -31, -61, …

- Audible audio signals have frequencies in the range 20 to 20kHz. Which is why good quality digital audio is sampled at rates >40kHz
- Usually there is multiple channels (mono = 1 channel, stereo = 2 channels (left and right). Within digital audio stream, left and right (PCM) sample values are interleaved

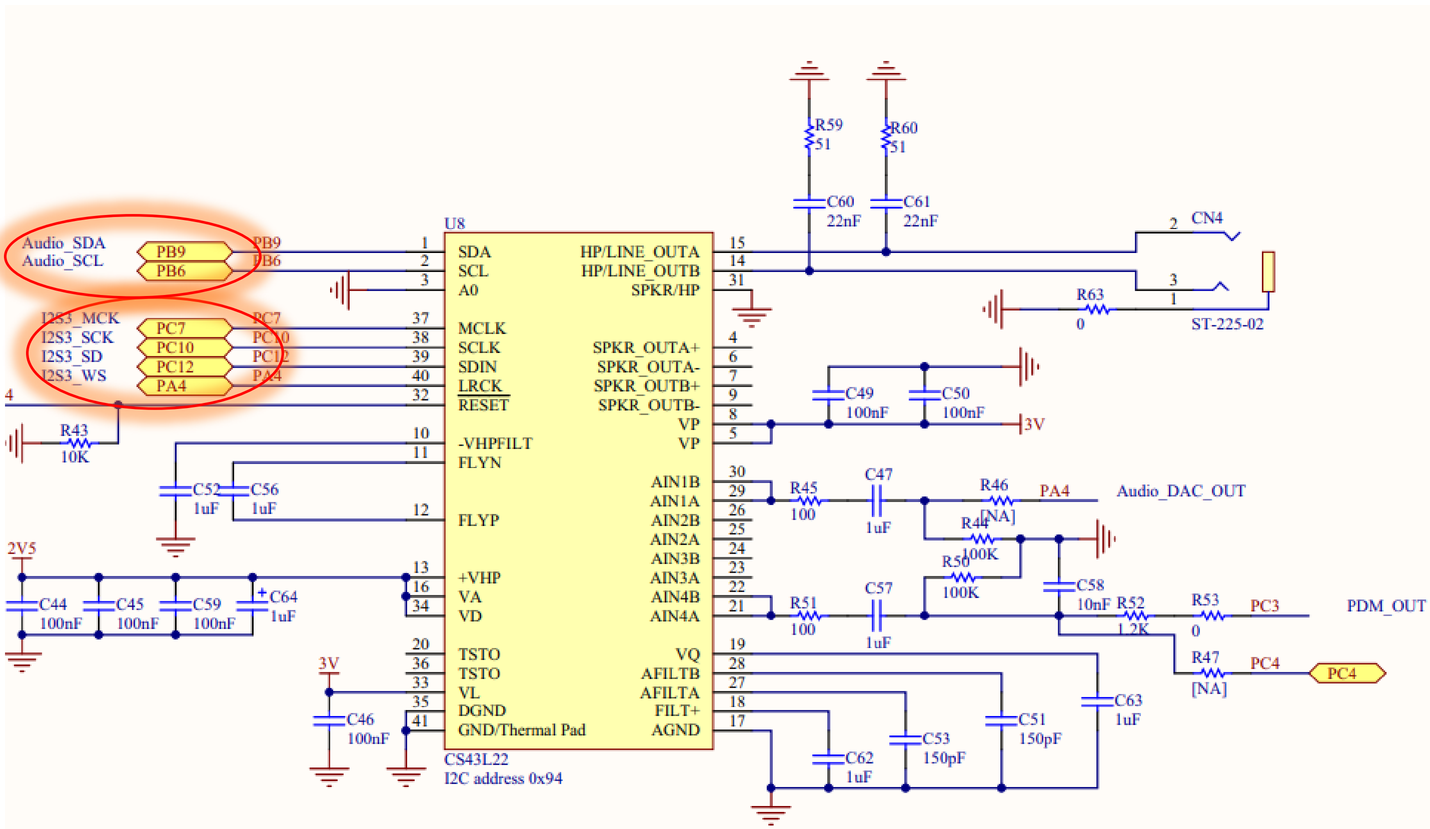
Audio signals
Klank siene



- Microphone audio signals have a very small voltage range (typically a few mV). They need amplification (signal conditioning – high gain) to be used with an ADC
- Speaker voltage range is not significantly large → Loudspeaker or headphones is a resistive load. It is more a case of supplying enough current to the loudspeaker/headphones

Audio on the STM32F411 development board
Klank op die STM32F411 ontwikkelingsbord

- External audio DAC connected to I2S3 peripheral (dual functionality with SPI3 peripheral)



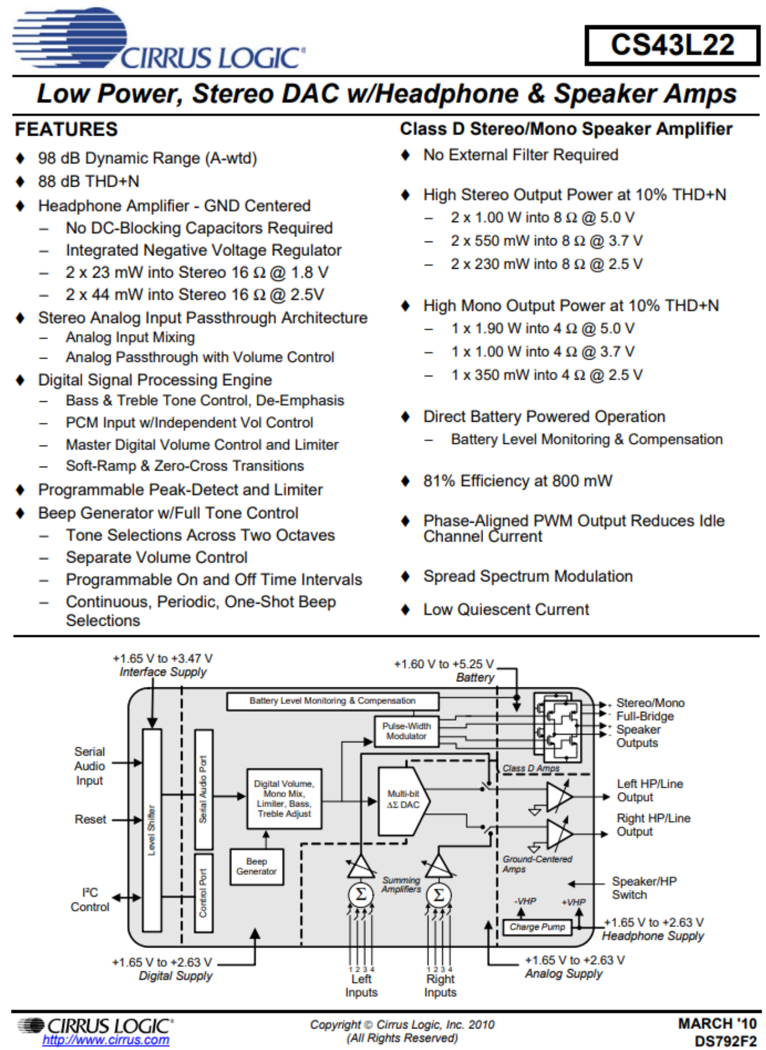
- But, the external DAC also needs to be setup using I2C

Audio on the STM32F411 development board
Klank op die STM32F411 ontwikkelingsbord

- CS43L22 External DAC
- Lots of options and settings
- Can select 16, 24, 32 bit I2S data
- I2C used for
  - Data type selection
  - volume
  - Bass, treble, filtering
  - Lots of stuff
- Use with STM provided libraries and drivers

- (Development board also has a microphone onboard)

- Too much to implement in the emulator…



CIRRUS LOGIC®

CS43L22

Low Power, Stereo DAC w/Headphone & Speaker Amps

FEATURES

- 98 dB Dynamic Range (A-wtd)
- 88 dB THD+N
- Headphone Amplifier - GND Centered
  - No DC-Blocking Capacitors Required
  - Integrated Negative Voltage Regulator
  - 2 x 23 mW into Stereo 16 Ω @ 1.8 V
  - 2 x 44 mW into Stereo 16 Ω @ 2.5V
- Stereo Analog Input Passthrough Architecture
  - Analog Input Mixing
  - Analog Passthrough with Volume Control
- Digital Signal Processing Engine
  - Bass & Treble Tone Control, De-Emphasis
  - PCM Input w/Independent Vol Control
  - Master Digital Volume Control and Limiter
  - Soft-Ramp & Zero-Cross Transitions
- Programmable Peak-Detect and Limiter
- Beep Generator w/Full Tone Control
  - Tone Selections Across Two Octaves
  - Separate Volume Control
  - Programmable On and Off Time Intervals
  - Continuous, Periodic, One-Shot Beep Selections

Class D Stereo/Mono Speaker Amplifier

- No External Filter Required
- High Stereo Output Power at 10% THD+N
  - 2 x 1.00 W into 8 Ω @ 5.0 V
  - 2 x 550 mW into 8 Ω @ 3.7 V
  - 2 x 230 mW into 8 Ω @ 2.5 V
- High Mono Output Power at 10% THD+N
  - 1 x 1.90 W into 4 Ω @ 5.0 V
  - 1 x 1.00 W into 4 Ω @ 3.7 V
  - 1 x 350 mW into 4 Ω @ 2.5 V
- Direct Battery Powered Operation
  - Battery Level Monitoring & Compensation
- 81% Efficiency at 800 mW
- Phase-Aligned PWM Output Reduces Idle Channel Current
- Spread Spectrum Modulation
- Low Quiescent Current

CIRRUS LOGIC®
http://www.cirrus.com

MARCH '10
DS792F2

Audio using the STM32F4 emulator
Klank met die STM32F4 emuleerder

- Audio player connected to I2S3
  - PA4 = I2S WS (channel select)
  - PC10 = I2S clock
  - PC12 = I2S data
- Audio player accepts single channel audio only, 16 bits per sample, with data rate of 22050 samples per second.
- The emulator CPU is too slow for real-time audio synthesis ☹
- Can only play back pre-recorded sounds from SRAM or Flash memory
- You have to use DMA
- Use HAL_I2S_Transmit_DMA to play sound from a buffer in memory

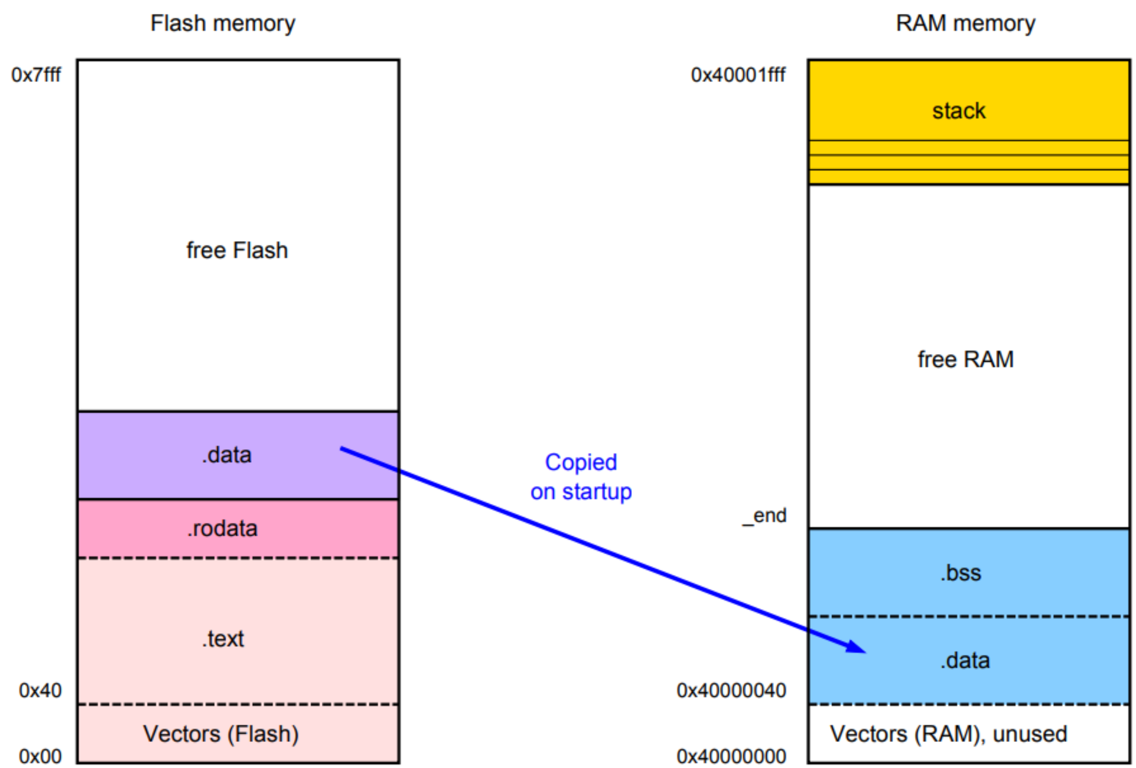
Audio using the STM32F4 emulator
Klank met die STM32F4 emuleerder

- Memory sizes for the STM32F411VE (and emulator):
  - SRAM: 128kB
  - Flash: 512kB
- You can play sounds from SRAM or Flash
- Maximum audio duration
  - SRAM: If you have the entire 128kB available (which you don't) ~6s
  - Flash: If you have the entire 512kB available (which you don't) ~23s
- Ideally you should have been able to load sounds from SD card:
  - Read initial buffer from file and start playing using DMA
  - While playing, read next block of data from SD card
  - After callback, play next chunk
  - Unfortunately emulated SD card+SPI is too slow ☹
- You can still load short sound samples from SD card in entirety to buffer in SRAM and play from there – but probably won't fit buffer larger than 3s in SRAM
- Or you can store samples as part of program code in flash memory
  - declare the array as `static const int16_t my_sound[] = {…};`

Where is my array data stored?
Waar word my skikking data gestor?

- The linker decides where to place variables, arrays, program code etc.
- The linker arranges your program into sections:
- **.text**: Program code. Read only
- **.rodata**: constants and strings (read only)
- **.data**: Initialised global and static variables (non-zero initial value)
- **.bss**: Uninitialised global and static variables (zero value on startup)

C compiler. Memory map. Program in Flash



Where is my array data stored?
Waar word my skikking data gestoor?

- After building your program, the linker outputs a .map file which tells you where everything is located

Preparing sound files for playing in the emulator
Voorbereiding van klank vir die emuleerder

- Audacity – Open source sound editing program: https://www.audacityteam.org/

- Free sound samples
  - Everywhere on the internet!
  - Remember, don't pirate music!

- Use audacity to record your own

- In order to play the sound through the emulator:
  - Re-sample to 22050Hz if needed
  - Convert to mono if needed
  - Export it as raw PCM binary data file (16-bit PCM)
  - Use the provided Audio2Source program to convert the binary PCM file to a text string with the array initialization
  - Copy the string into your program source code
  - Call HAL_I2S_Transmit_DMA to play the sound