

Achieving optimal scalability and voice quality in open source telephony

Konrad Hammel
Software Engineer
Sangoma Technologies



WHAT IS THE SECRET TO SCALABILITY AND QUALITY?

The background features a large, light-colored gear on the right side and a stylized, outlined letter 'A' on the left side, both rendered in a light blue or grey tone. The overall background is a light, neutral color with a subtle gradient.

USING THE BEST HARDWARE AND THE BEST SOFTWARE

Outline



- Sangoma Hardware
 - AFT-Series
 - B-Series
 - Other Hardware
- Intro to Asterisk Architecture
- Bottlenecks and Scalability Issues
 - Everything in Software
 - Chunk Size restriction of Dahdi
 - Channel Based
 - Monolithic Design
- FreeTDM + SMG + SIP/Woomera
- Questions???



SANGOMA HARDWARE

AFT Series



- Advanced Flexible Telephony
 - Award winning design from scratch
- OOP Design
 - Modular -> PCI/PCIe interface, telephony interface, DSP
 - Abstraction -> common base, Remora system
- Higher per card cost but lower maintenance and easier to stock

AFT Series - Continued



AFT Series - Features



- Octasic HWECC
 - Industry's 1st telco grade HWECC
 - Adjustable 128ms tail
 - Fully Independent...no fine tuning needed
 - Fax/Modem and DTMF detection
- Field Upgradable Firmware
 - Fix bugs and add new features on the fly
- Crash Proof Firmware
 - Recover the card after “act of god” accidents
- Industry first and only **LIFETIME WARRANTY**

AFT Series - Features

Fax Sync

- Reliable T1/E1/BRI to analog faxing
- Syncs clock from digital to analog card



Remora Expansion System

- Add more telephony ports without using PCI/PCIe slots
- Up to 24 ports per card



AFT Series - Analog

- A200
 - Low density modular
 - 2-24 port FXO/FXS
 - 2u, PCI/PCIe(E), half-length
 - Optional HWEC (D)
- A400
 - High density modular
 - 2-24 port FXO/FXS
 - 2u, PCI/PCIe(E), full length
 - Optional HWEC (D)



AFT Series – Digital T1/E1

- A10X line
 - A101, A102, A104,
and A108
- 2-8 T1/E1/J1 ports
- Channelized for voice and data
- 2u, PCI/PCIe (E) half-length
- Optional HWEC (D)



AFT Series – Digital BRI

- A500
- 2-6 port modular BRI, up to 24 with Remora
- 2u, PCI/PCIe (E), half-length
- Optional HWEC (D)



B-Series – Mix Mode

- B700
- Modular BRI and analog
- 2-4 BRI, 2 FXO/FXS
- 2u, PCI/PCIe(E) half-length
- Optional HWEC (D)
- 5 year warranty



B-Series – Mix Mode

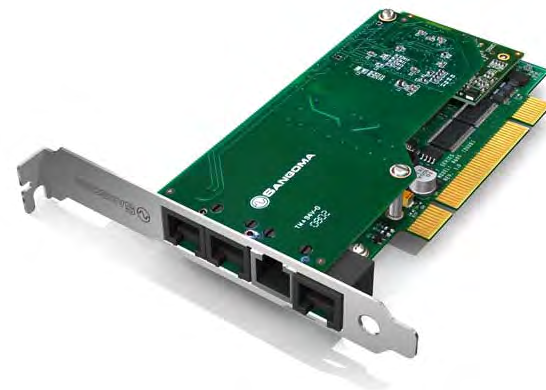
B600

- 4 FXO ports, 1 FXS port
- 2u, PCI/PCIe, half-length
- Optional HWEC (D)
- 5 year warranty



B601D

- 4 FXO, 1 FXS, 1 T1/E1/J1
- 2u PCI/PCIe , half-length
- Comes with HWEC
- 5 year warranty



Other Hardware

U100 (USBFXO)

- 2 port FXO interface via USB
- Comes with HWEC
- 5 year warranty



UT-50/UT-51

- Asterisk timing device
- USB (UT-50) and internal pin header (UT-51) interface
- 5 year warranty

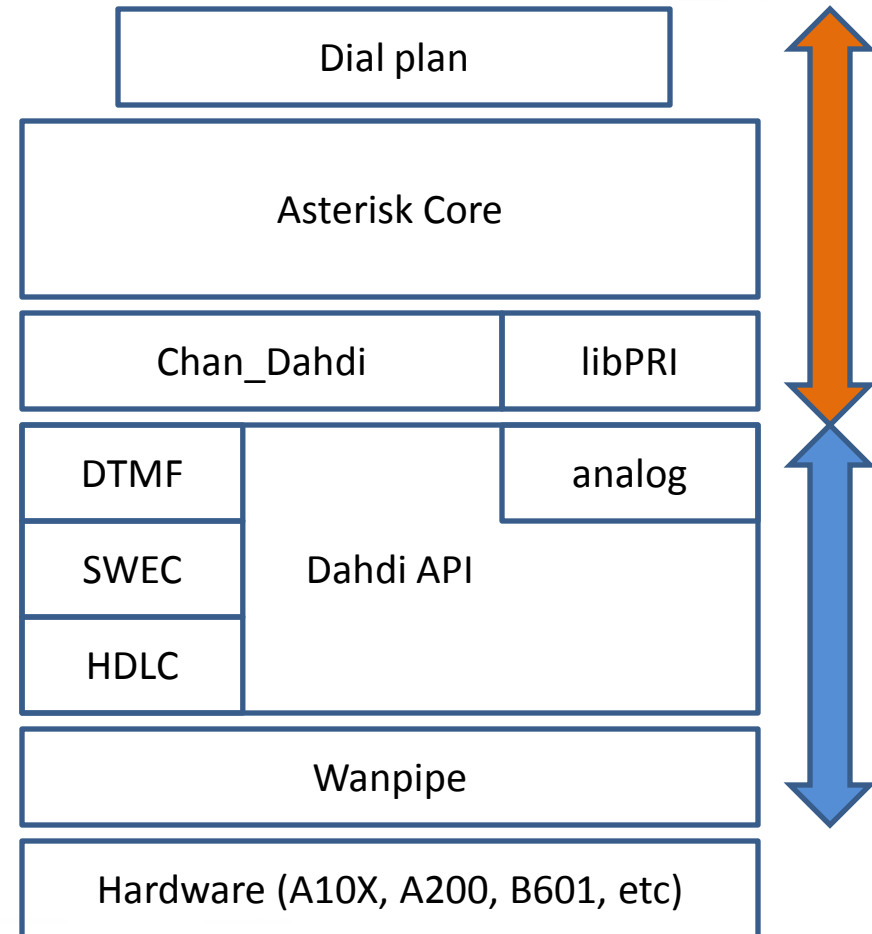




INTRODUCTION TO ASTERISK ARCHITECTURE

Asterisk Architecture

- Asterisk Core
- Channel Drivers like Chan_SIP and Chan_Dahdi
- Action Plan (dial plan)
- Dahdi API
- Hardware Drivers
- Hardware
- User Space
- Kernel Space



The background features a large, light-colored gear on the left side. Overlapping the gear is a stylized, blue-outlined 'W' shape. The overall background is a light blue gradient.

BOTTLENECKS AND SCALABILITY ISSUES

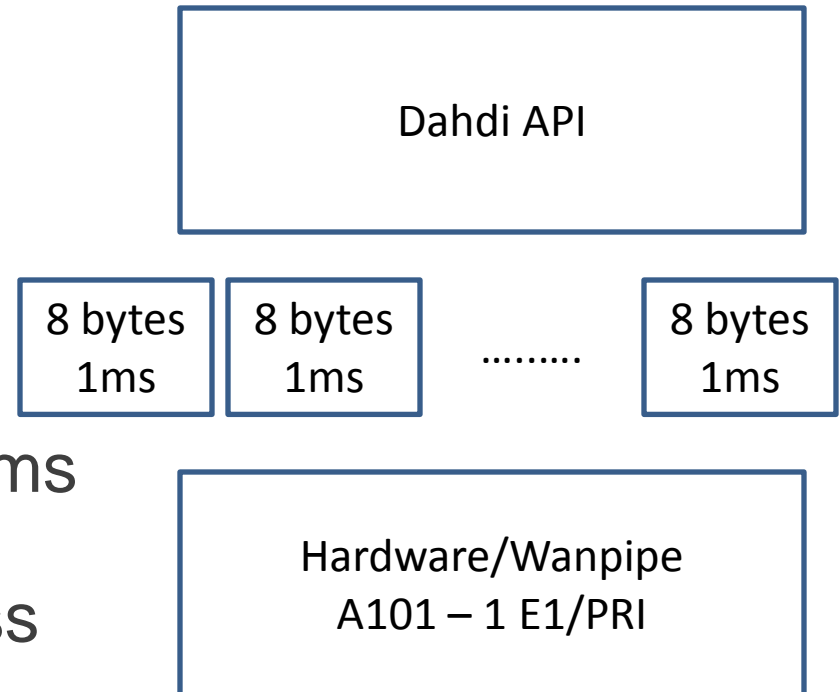
Software Everything



- HDLC encoding
 - Easy but still takes processing power
 - Very simple to do in FPGA based hardware
- Echo Cancelling
 - Extremely CPU intensive...complicated math
 - Audio glitches when not perfect
 - DSP designed to do math
- DTMF Detection
 - Like EC can be CPU intensive because it is math based
 - Hardware EC DSP can easily take care of this

Chunk Size

- Dahdi takes 1ms = 8 bytes
 - 1000 interrupts per second!
 - WHY???
 - Analog signaling
 - Software EC
 - Software DSP
- SOLUTION: Increase to 20ms chunks optimal for system performance (up to 70% less CPU load)
 - [How to Reduce Asterisk System load by 70%](#)



Channel Based



- PROBLEM: Channel Based API
 - Each voice channel gets a kernel device
 - Easy for user space...
 - 16 E1 ports = 496 devices
 - HUGE amount of context switches
- SOLUTIONS: Span Based API
 - Each span gets a device
 - A little more work in user space
 - Much less work done in time dependent kernel

Monolithic Design



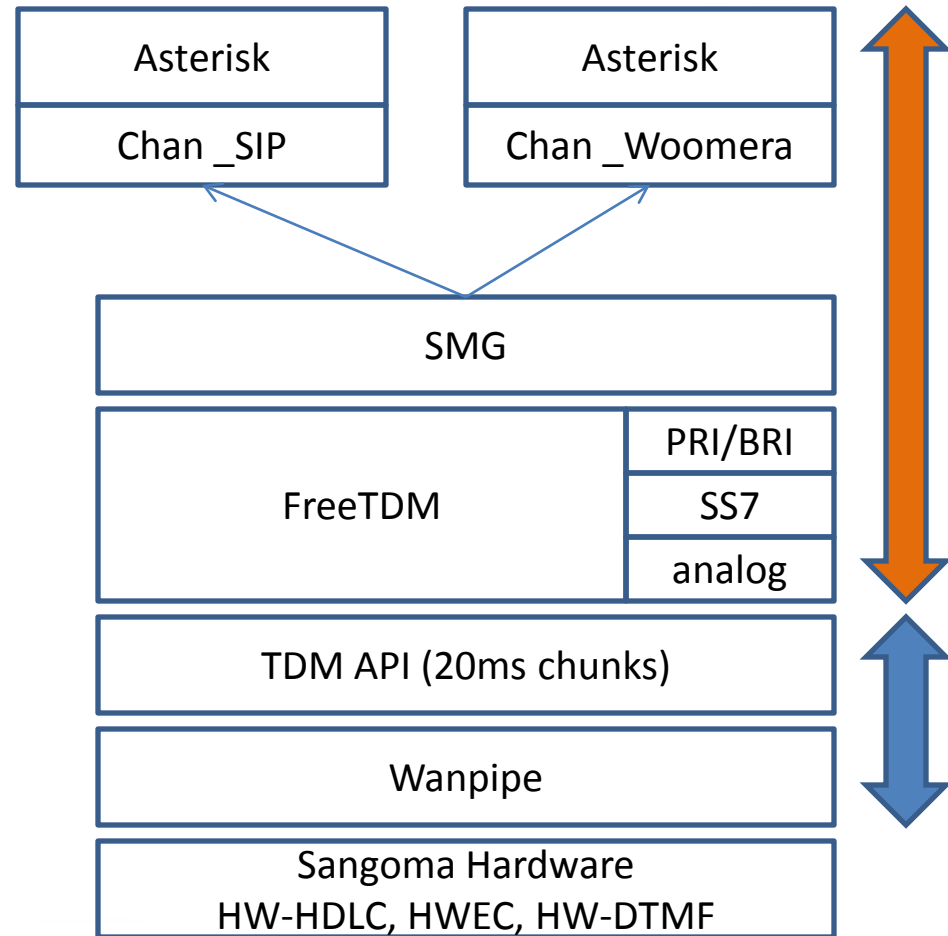
- PROBLEM: Dahdi, Chan_Dahdi, and Asterisk are linked directly
 - If one fails, the whole system fails
 - All load concentrate on one system
- SOLUTION: Woomera or SIP
 - Socket based connection to Asterisk
 - A crash on one side does not kill the other side
 - Client-Server Architecture
 - Allows for 1-to-Many connections (load balancing)
 - Asterisk registers into PSTN interface



FREETDM + SMG + WOOMERA/SIP

FreeTDM + SMG + SIP

- High Quality Sangoma Hardware
- Wanpipe Kernel drivers
- TDM API
- FreeTDM + Sig stacks
- SMG
- Chan_SIP or Chan_Woomera
- Asterisk Cores
- User vs. Kernel Space



Kernel Space



Sangoma Hardware

- Telco grade quality
- Hardware HDLC framing
- Hardware Echo Canceling
- Hardware DTMF Detection

TDM API

- Small, Open Source, kernel based API
- Runs at 20ms chunks
- Can run in channel mode or span mode
- No processing of any kind...just passes data
- OS independent

FreeTDM



- Open Source, User space, C based TDM/PSTN API
- Span based or Channel based
- Unified..handles voice and signaling
 - PRI, BRI, SS7, analog
 - DTMF detection and generation
 - Caller-id detection and generation
- Complete hardware abstraction allows any hardware to run
- “plug and play” stacks (open source and proprietary)
- Operating system independent: Linux and Windows

SMG

A decorative line graphic in the top right corner of the slide, consisting of a horizontal line that steps up, then down, then up again, and finally continues horizontally.

- Sangoma Media Gateway
- Open Source (always has been, always will be)
- Connects to FreeTDM and uses the FS core to access SIP or Woomera , transcoding (HW or SW), logging (unified hardware, TDM, stack logging), and a web front end interface
- Asterisk Channel bridging, SMG-to-SMG bridging
- OS independent: Linux and Windows

Conclusion

- Voice quality in Asterisk can be improved by:
 - Using telco grade hardware
 - Using telco grade HWEC
 - Optimizing for system load
- Asterisk scalability is achieved by:
 - Moving processor intensive tasks to hardware
 - Reducing system load by increasing data chunk size
 - Using a distributed architecture

