Scalable TDM Media Gateways With FreeSwitch & OpenZAP

Presented by: Nenad Corbic, B.Eng
Chief Software Engineer
Software Manager
Sangoma Technologies

Introduction

- Voice API Building Blocks

- TDM Architectural Limitations & Bottlenecks
- Asterisk/Zaptel FreeSwitch/OpenZap Architecture
  Zaptel (Channelized) Model vs TDM API (Span) model

- Unified TDM Voice API – libsangoma
- Abstracted Signaling Stacks – API - libsigboost
- Sangoma & FreeSwitch

- FreeSwitch OpenZap Architecture
- Features and Benefits of OpenZap
- OpenZap API
- OpenZap and Woomera

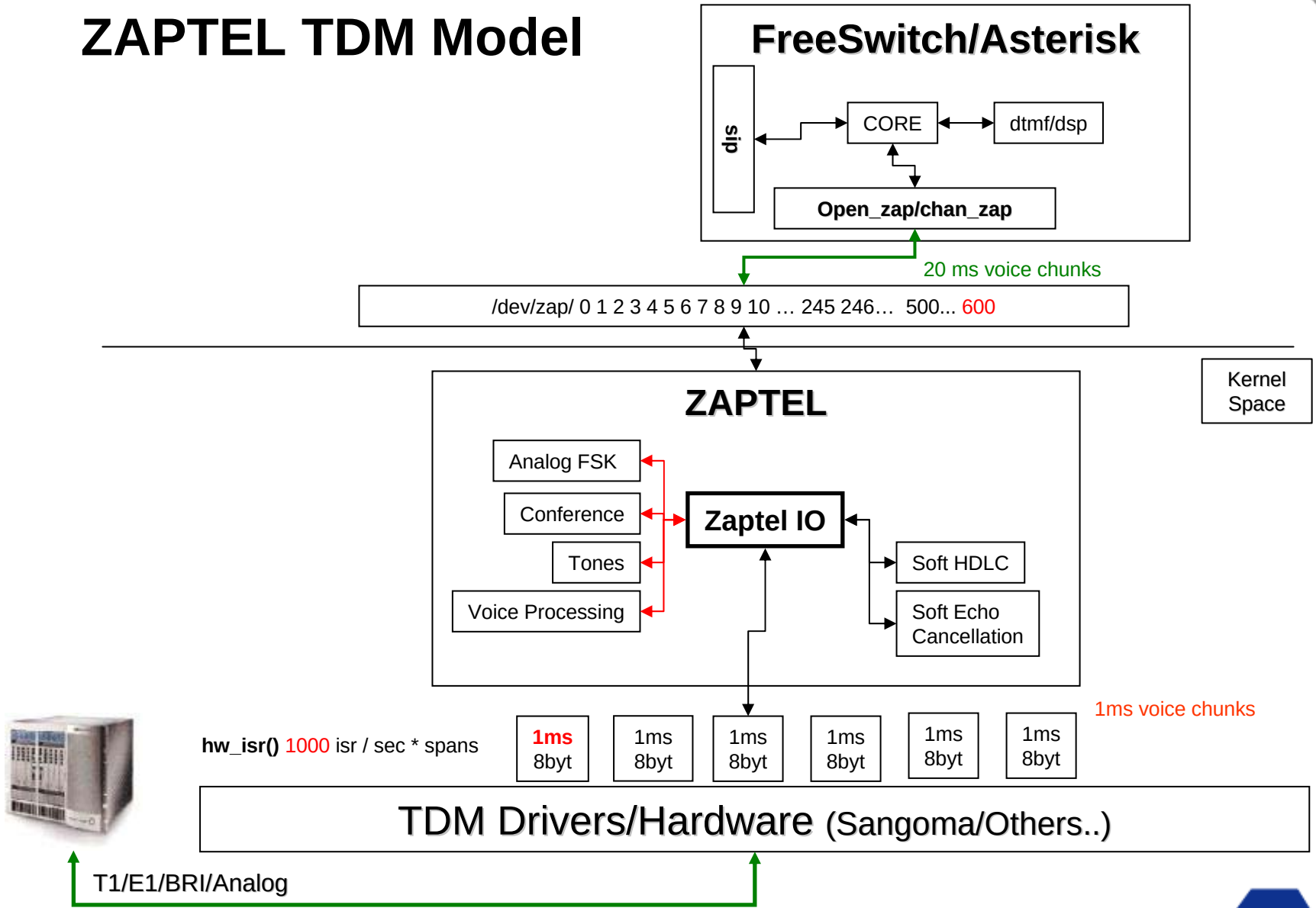
- Whats new at Sangoma: Software & Hardware

TDM Architectural Limitations & Bottlenecks

- Monolithic/Single Server vs Distributed Architecture.
- TDM Bottleneck (Zaptel/TDM Hardware)
- Interrupt Frequency / Real Time
- Interrupt Latency / Real Time
- Kernel / User Space Context Penalty
- TDM Clustering Solutions limited and hard to configure.
- Software Codec (g729)
- Software Echo Cancellation & DTMF & D-Chan HDLC

- Goal super scalable gateway ~ 1000 Voip to PSNT calls.

ZAPTEL TDM Model



Channelized Architecture

- **Device per channel**
- **T1 = 24 channels = 24 devices**

- **Interrupt per Card**
  Interrupt depends on chunk size
- **Chunk/Interrupts**
  1ms == 1000 intr/sec
  2ms == 500 intr/sec
  5ms == 200 intr/sec
  10ms == 100 intr/sec

Interrupt Frequency

- Generating interrupt every 8 bytes – 1000 interrupts per sec.
- Ok for single card.
- However adding multiple cards
- Exponential Drop Off in performance
- Solution is to use higher chunks.

Interrupt Latency

- System can to delay a hardware interrupt by 15 to 20ms
- Need for large hardware buffering to avoid overrun.

1 channel: 20ms = 160 bytes

A108 - 8 Port E1 card = 248 channels

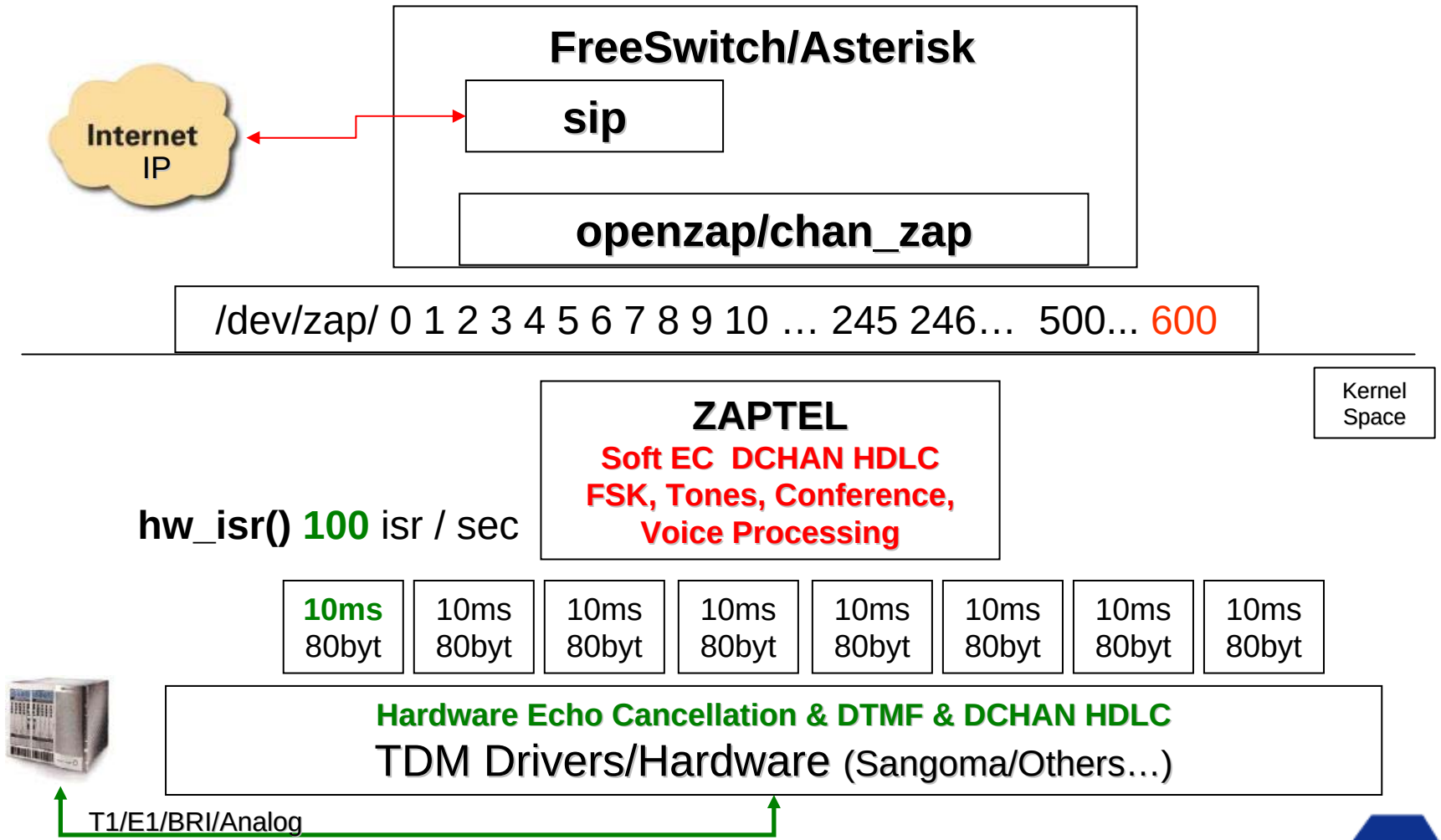
Buffering for Rx = 248ch * 160 bytes ~= 40KB

Buffering for Tx = 248ch * 160 bytes ~= 40KB

Total = 80KB of memory in hardware

- Cost vs Functionality

HW Optimizations: EC & DTMF & DCHAN HDLC



Channelized: Chunk Size Performance

- System: 16 E1 - 496 Channels - Quad Core / 4Gig Memory

- Chunk Sizes:

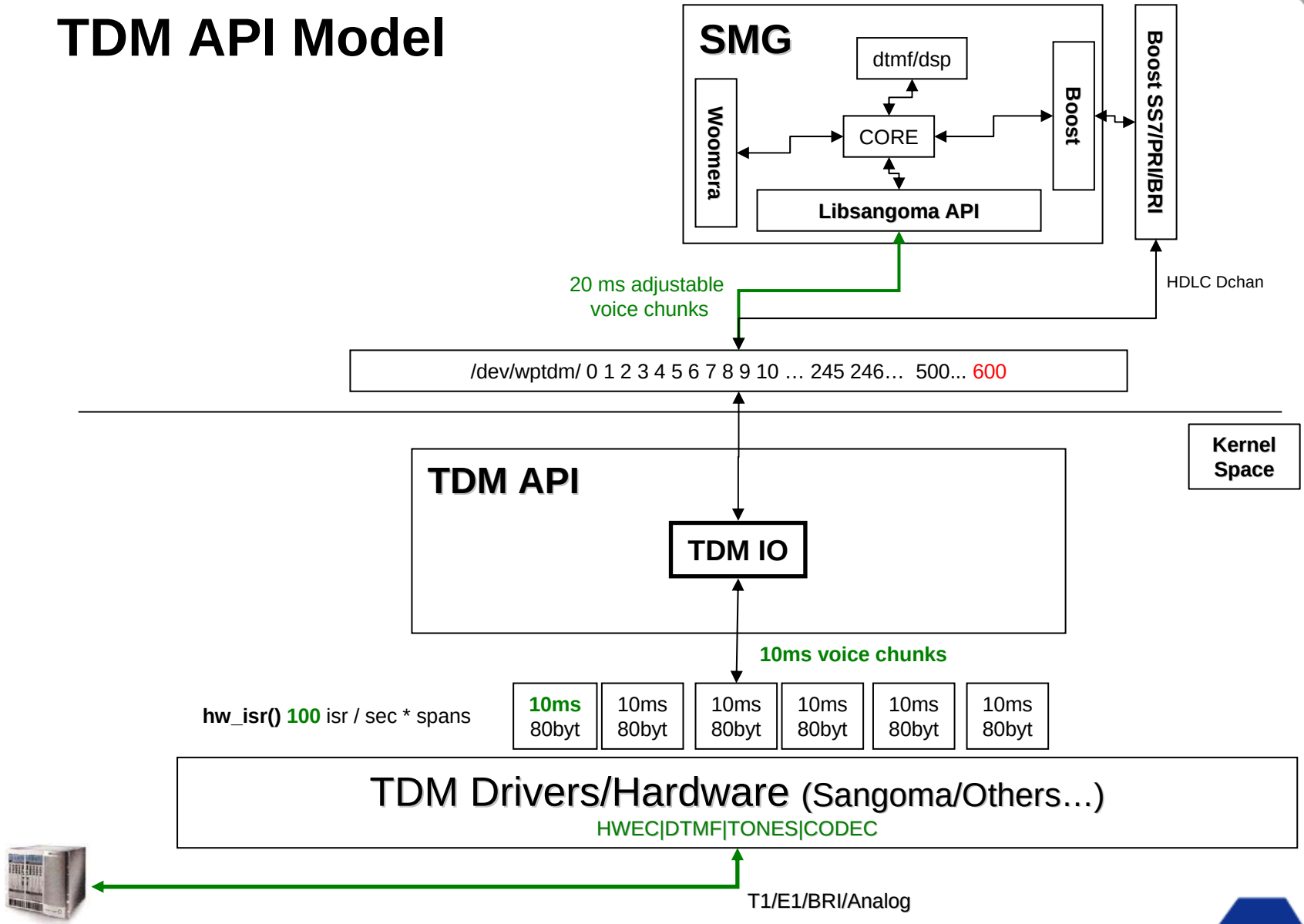
- 8 bytes -- 1ms interrupt == 1000 interrupts/sec
- 16 bytes -- 2ms interrupt == 500 interrupts/sec
- 40 bytes -- 5ms interrupt == 200 interrupts/sec
- 80 bytes -- 10ms interrupt == 100 interrupts/sec

- Results:

Chunk	Inter/sec	Sy Load: Idle	Sy Load: 496 channels
8	1000	15%	26%
16	500	6%	16%
40	200	2%	9%
80	100	1%	7%

```
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu------
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs  us sy  id wa st
15  0      0 705904 591500 1377936    0    0     0   172 1285 45546 13  7  80  0  0
```

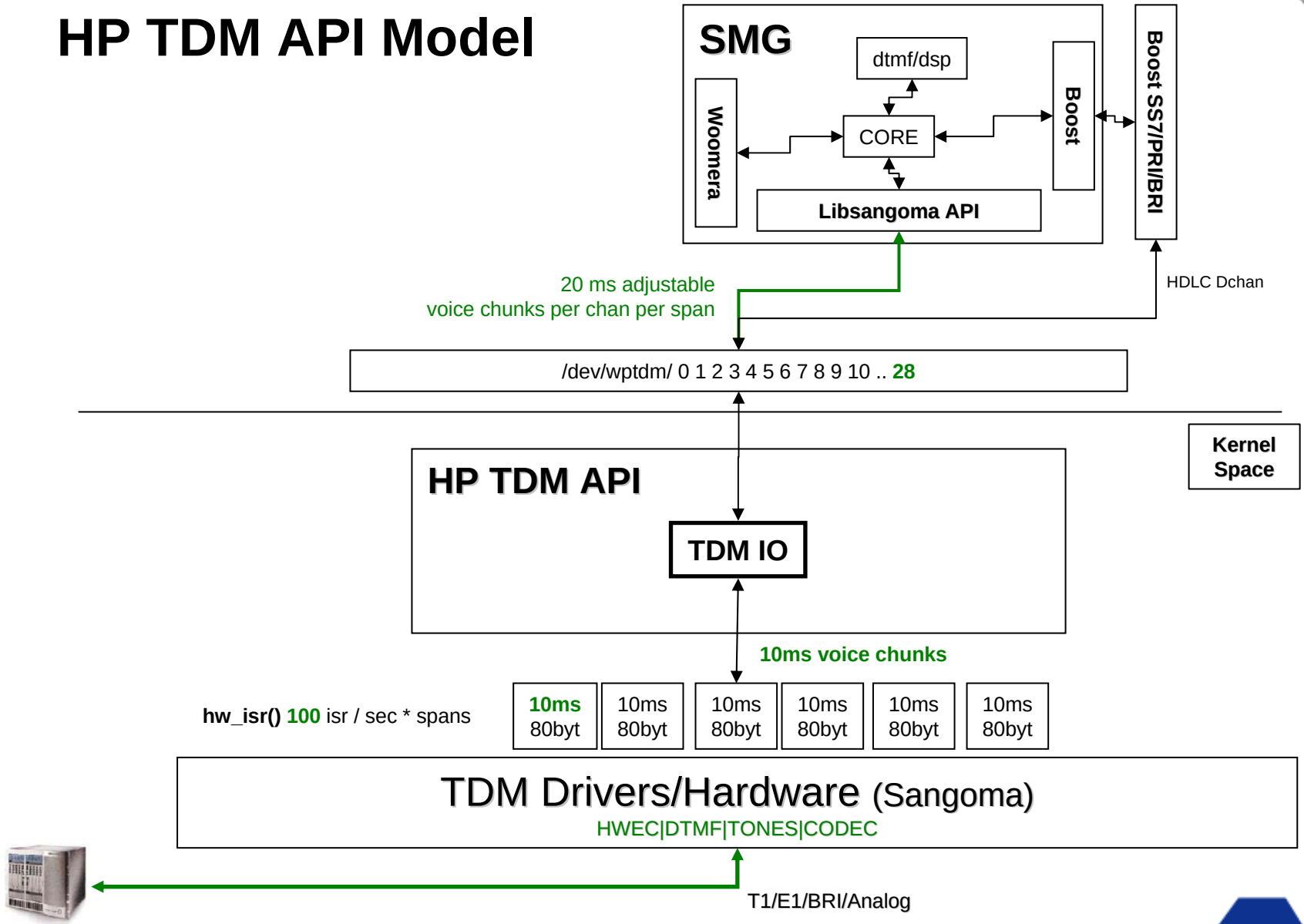
TDM API Model



Kernel Context Penalty

- Greater the number of kernel devices, the greater the context penalty.
- System doesn’t scale over 500 individual channels.
- Where each channel receives its own kernel device.
- Solution is to move to a per span kernel device.
- This way all channels inside a span are read in a same kernel context.
- Stage two solution is to memory map user space into kernel space and have zero copy transfers from kernel to user space.

HP TDM API Model



SPAN Mode (HP TDM API) Performance

- System: 16 E1 - 496 Channels - Quad Core / 4Gig Memory

- Chunk Sizes:

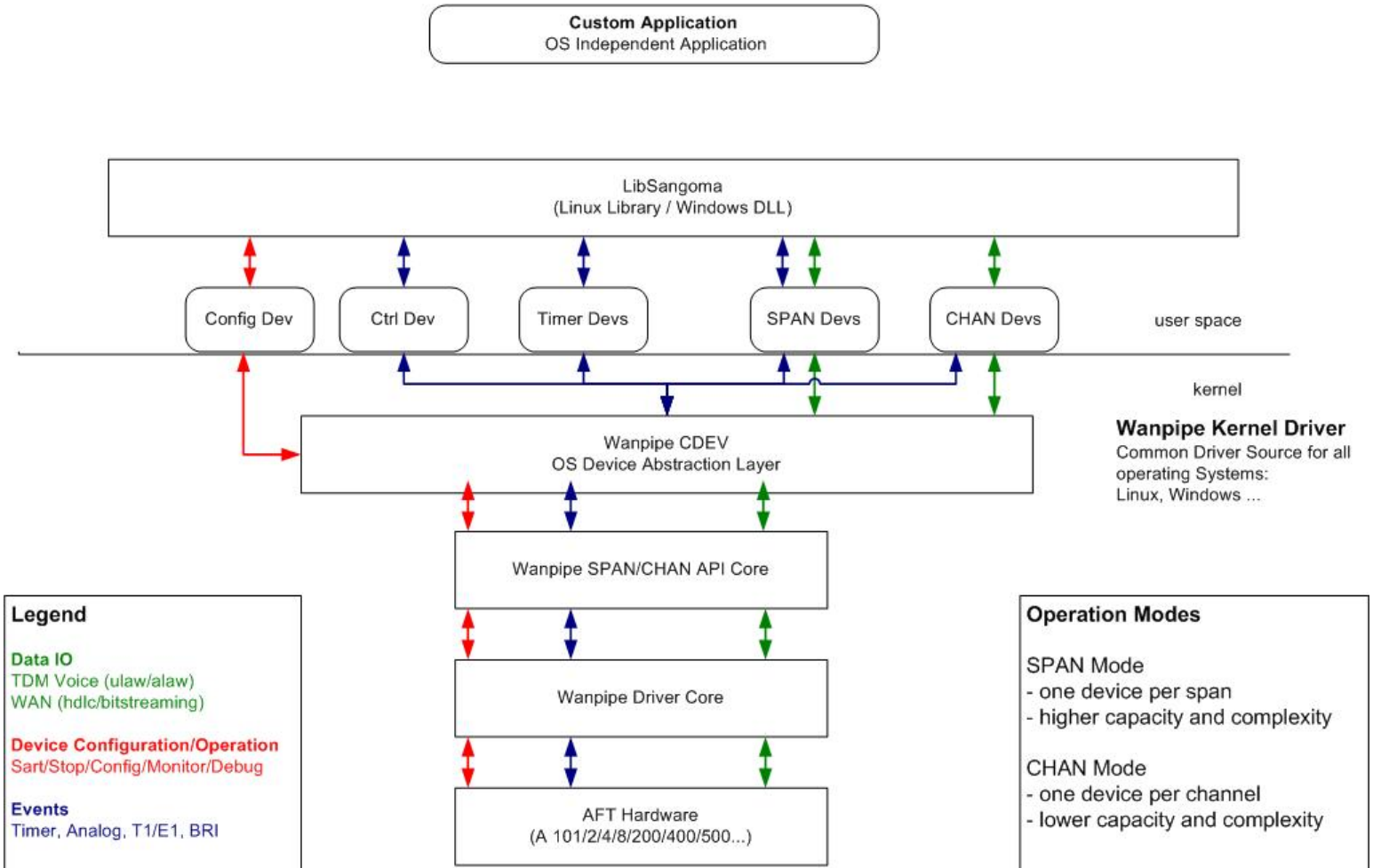
- 8 bytes -- 1ms interrupt == 1000 interrupts/sec
- 16 bytes -- 2ms interrupt == 500 interrupts/sec
- 40 bytes -- 5ms interrupt == 200 interrupts/sec
- 80 bytes -- 10ms interrupt == 100 interrupts/sec

- Results:

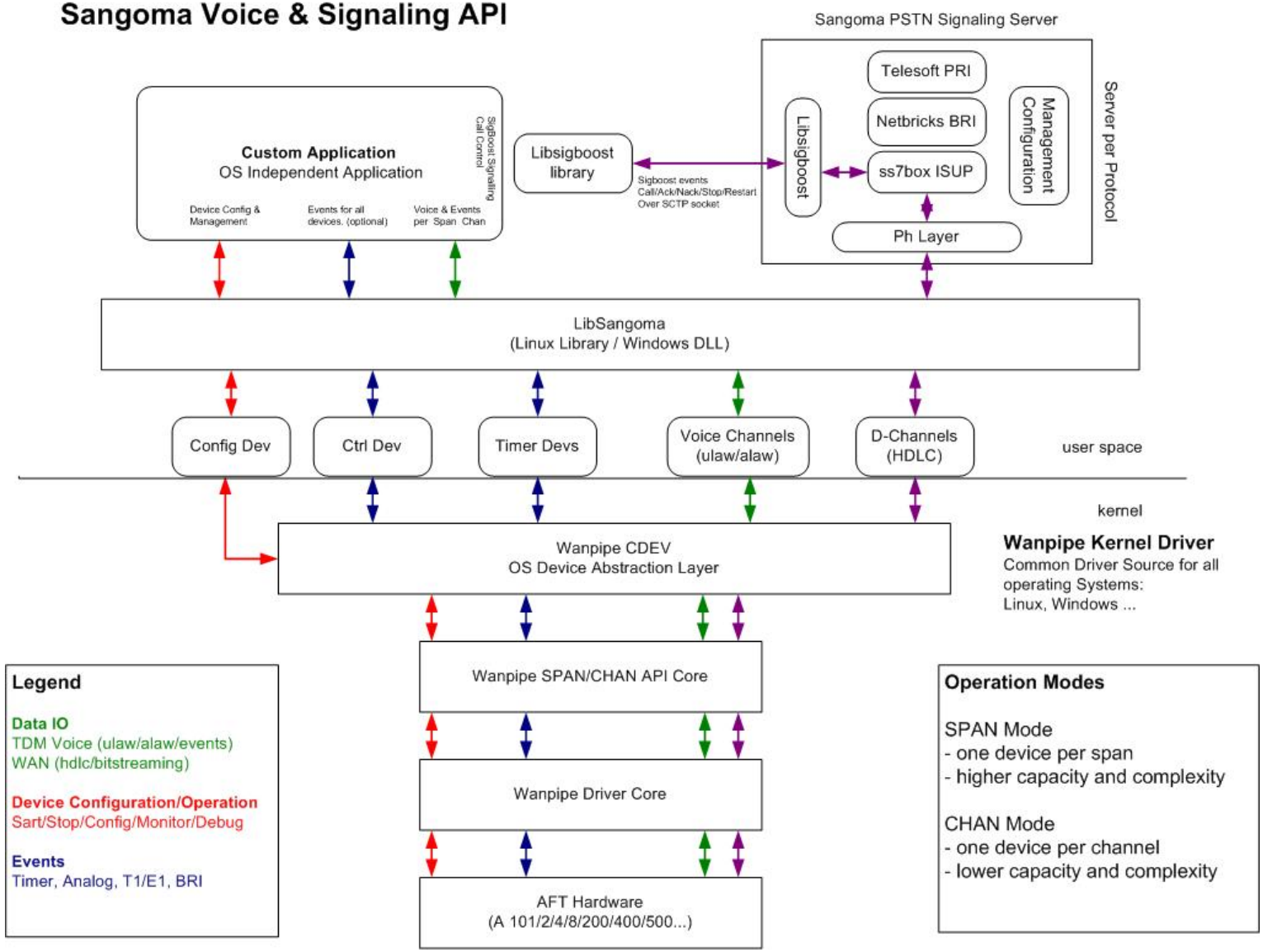
Chunk	Inter/sec	Sy Load: Idle	Sy Load: 496 channels
80	100	1%	**5%**

procs		memory				swap		io			system		cpu				
r	b	swpd	free	buff	cache	si	so	bi	bo		in	cs	us	sy	id	wa	st
0	0	0	407764	519728	1855568	0	0	0	0		3744	**6250**	12	**5**	83	0	0

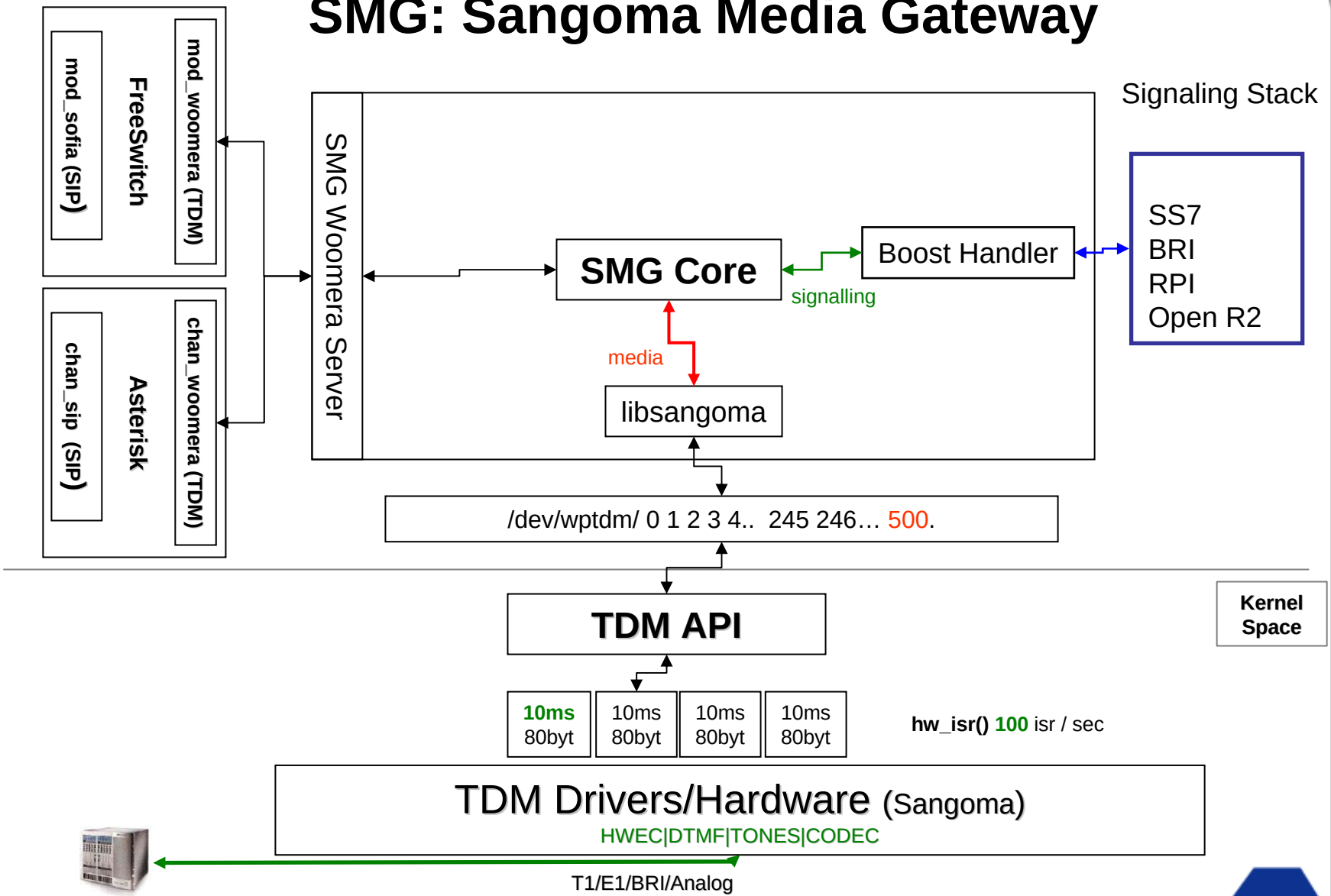
Sangoma Unified Wanpipe API



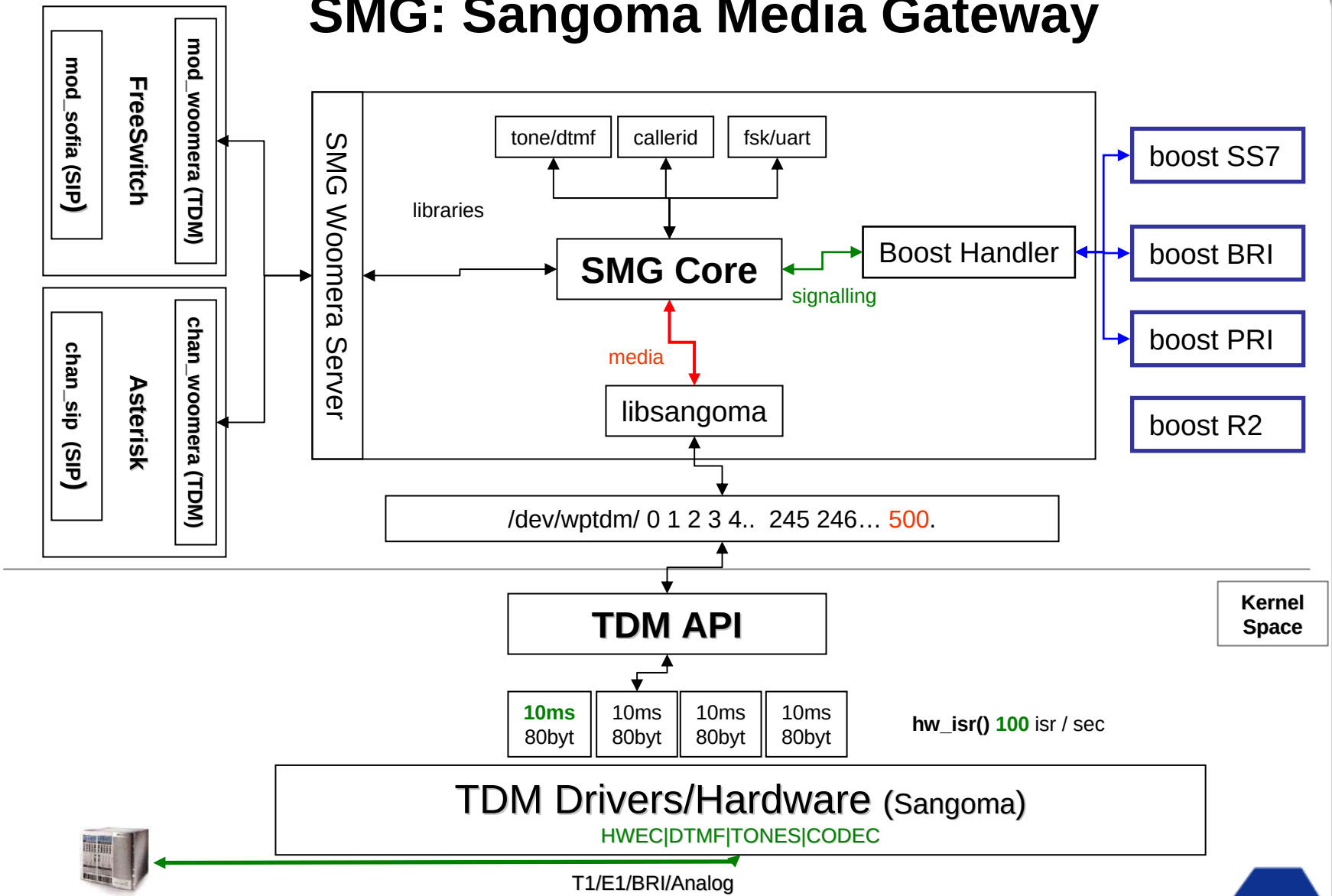
Sangoma Voice & Signaling API



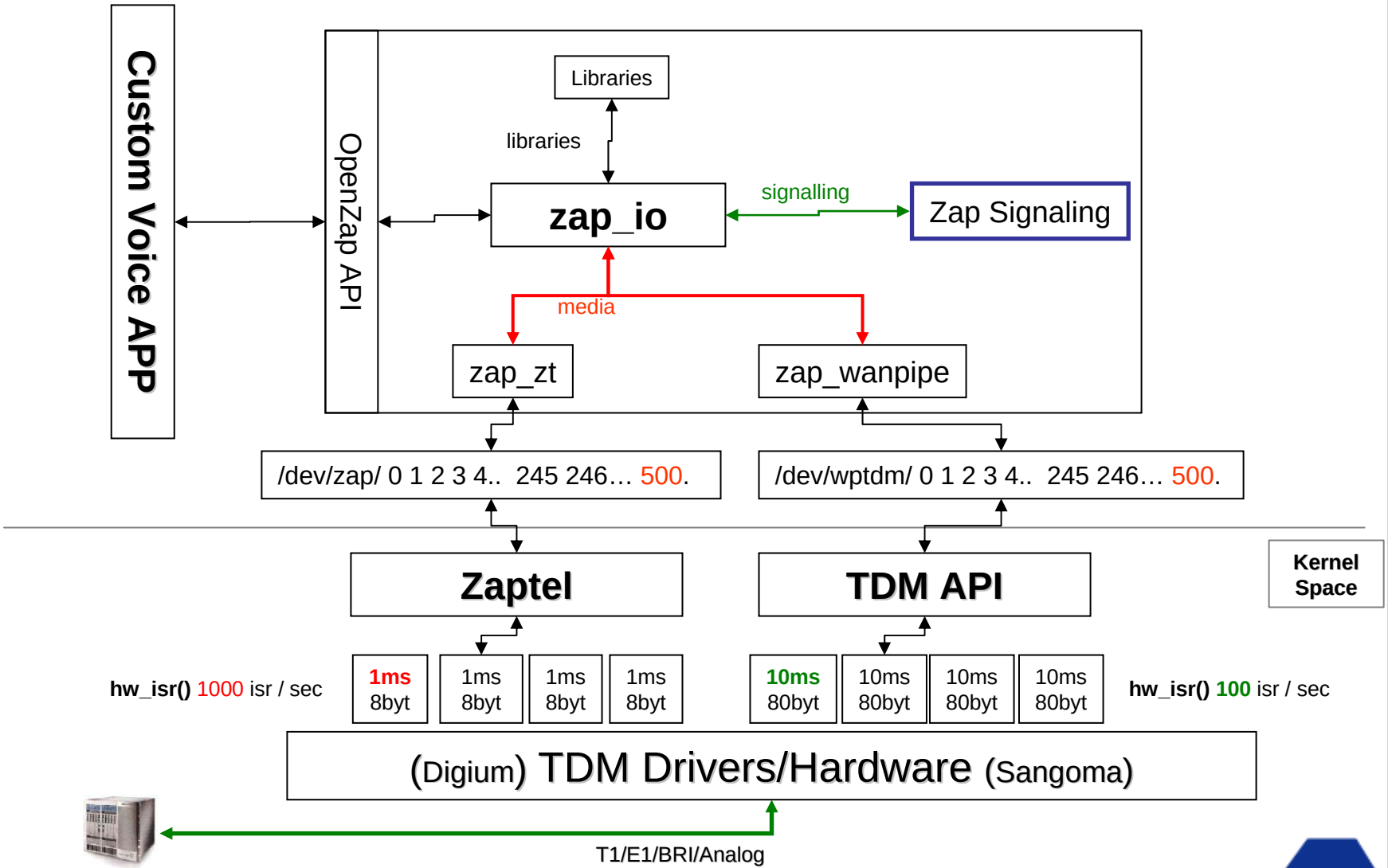
SMG: Sangoma Media Gateway



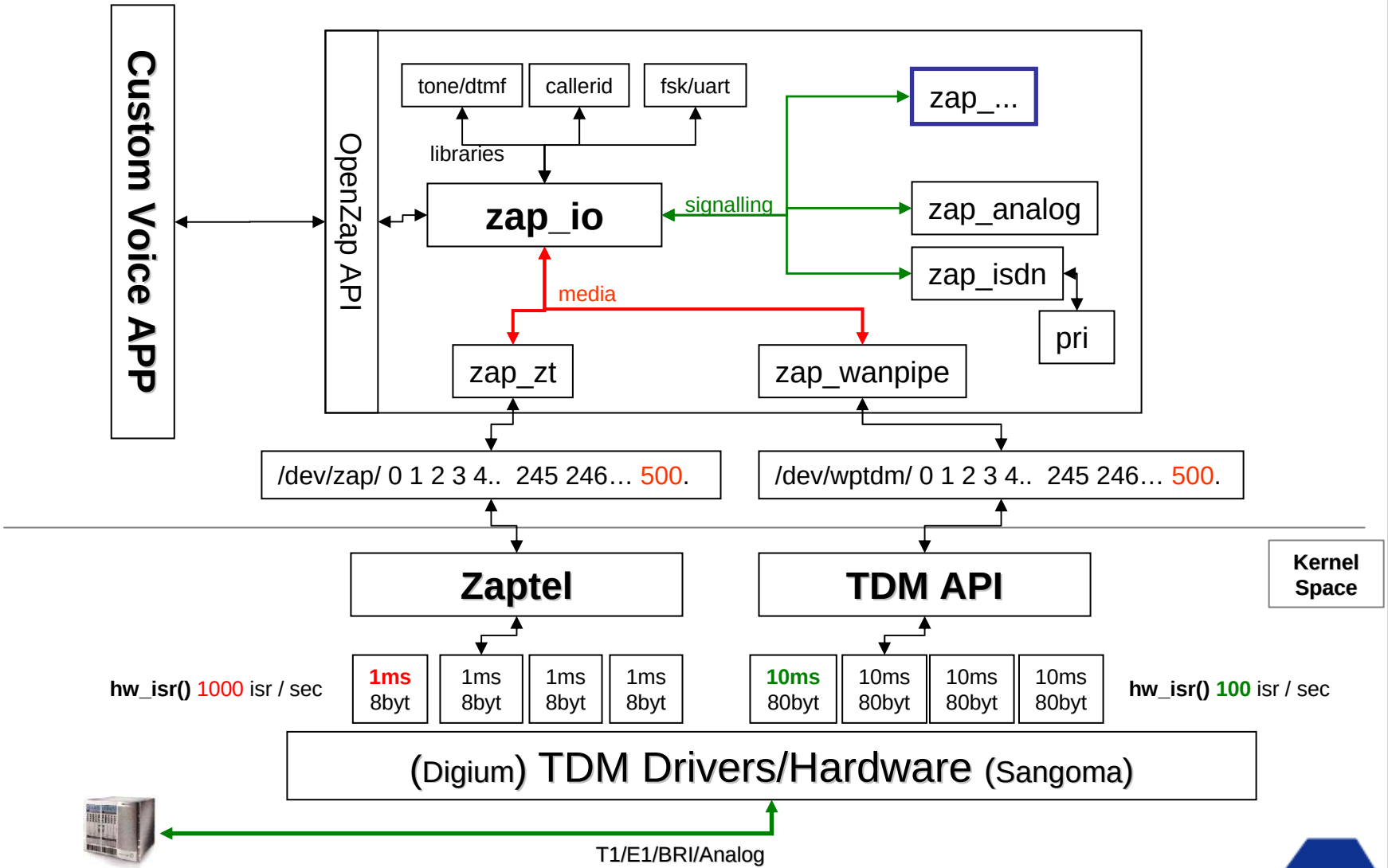
SMG: Sangoma Media Gateway



TDM Voice & Signalling API: OpenZap



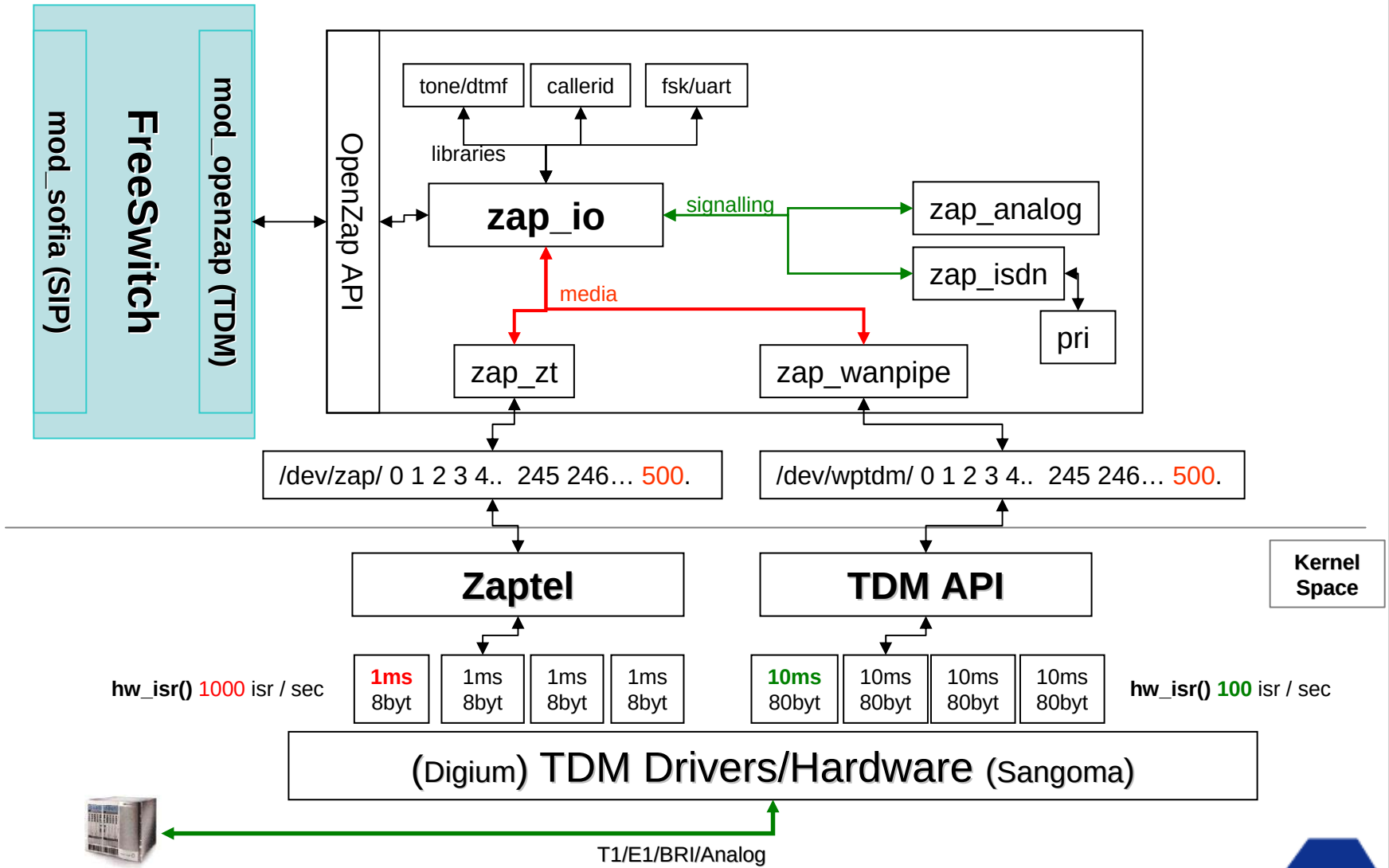
TDM Voice & Signalling API: OpenZap



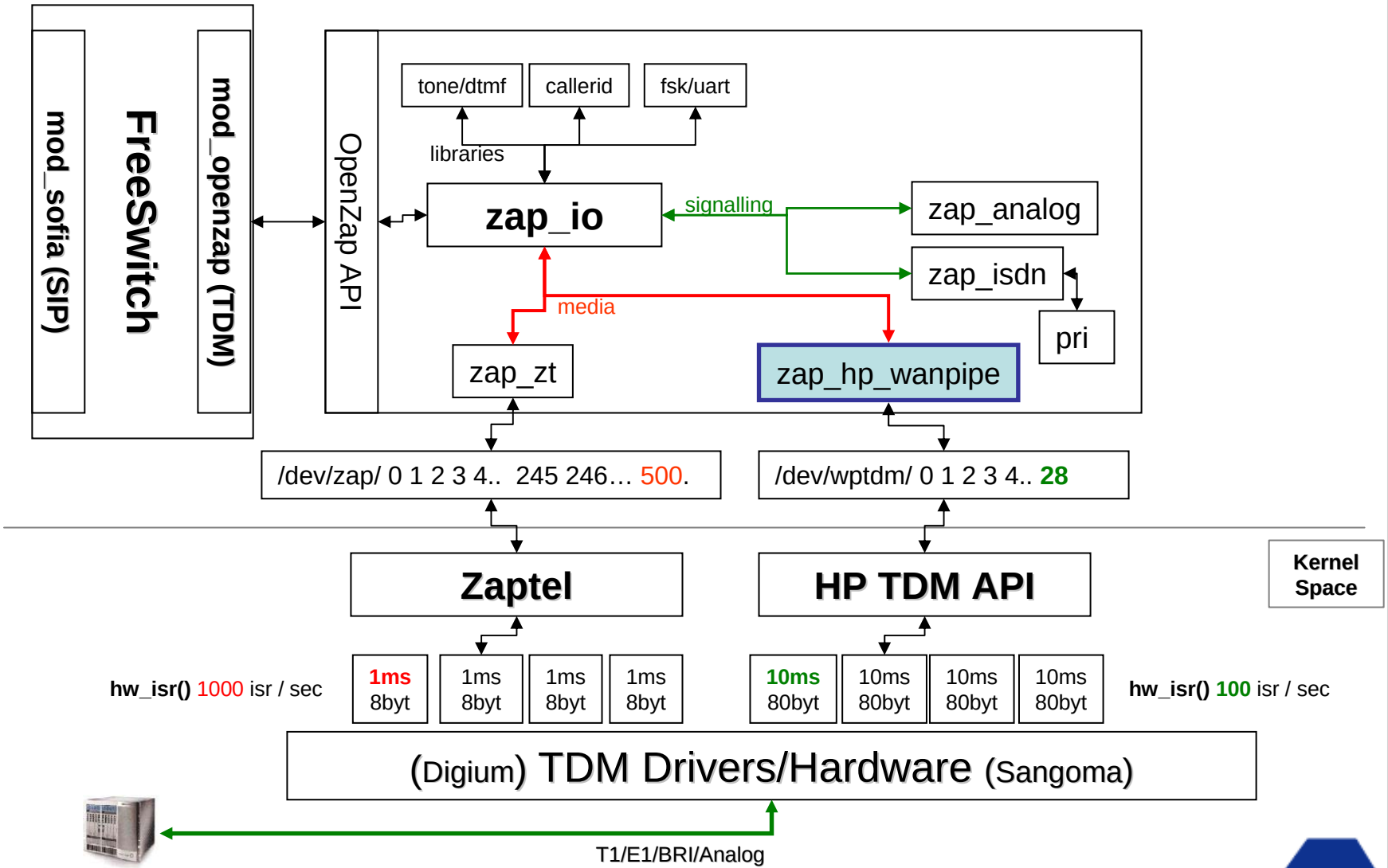
Open Zap Architecture

- **Unified, User Space, C API for both Voice and Signalling**
- **Object Oriented Voice Architecture**
- **Clearly defined abstraction between:**
  Media, Signalling, Voice Processing and Hardware.
- **Unification of hardware vendors:**
  Zaptel -> Digium, TDM API -> Sangoma, etc…
- **Easy reuse of existing stacks! Easy addition of new stacks and or voice processing libraries.**
- **Operating System Independent: Linux, BSD, Solaris, Windows …**

FreeSwitch & OpenZap



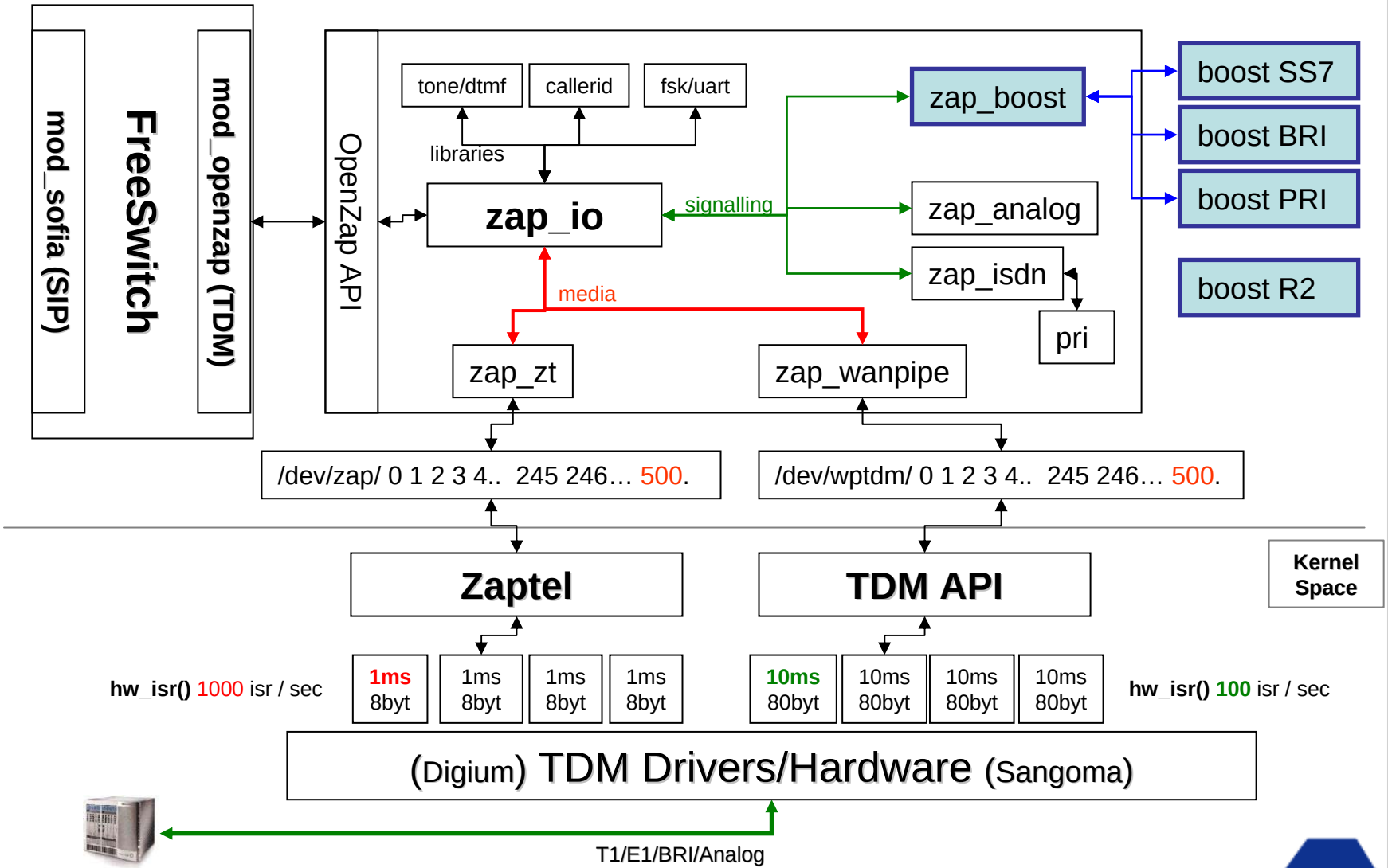
FreeSwitch HP OpenZap



Open Zap & Boost (PRI/BRI/SS7) Protocols

- Unified API to TDM Signaling Stacks (libsigboost)
- Independent development paths.
- Increased Stability due to Socket Interface.
- Proven, production ready, field tested stacks.
- Stability and functionality Inheritance.

FreeSwitch & OpenZap & Boost Protocols



FreeSwitch OpenZap Configuration

- **Open Zap Config File**
  -> /etc/openzap/openzap.conf
  [span wanpipe]
  name => OpenZAP
  number => 1
  trunk_type => e1
  b-channel => 1:1-15
  ….
  (Equivalent to /etc/zaptel.conf)

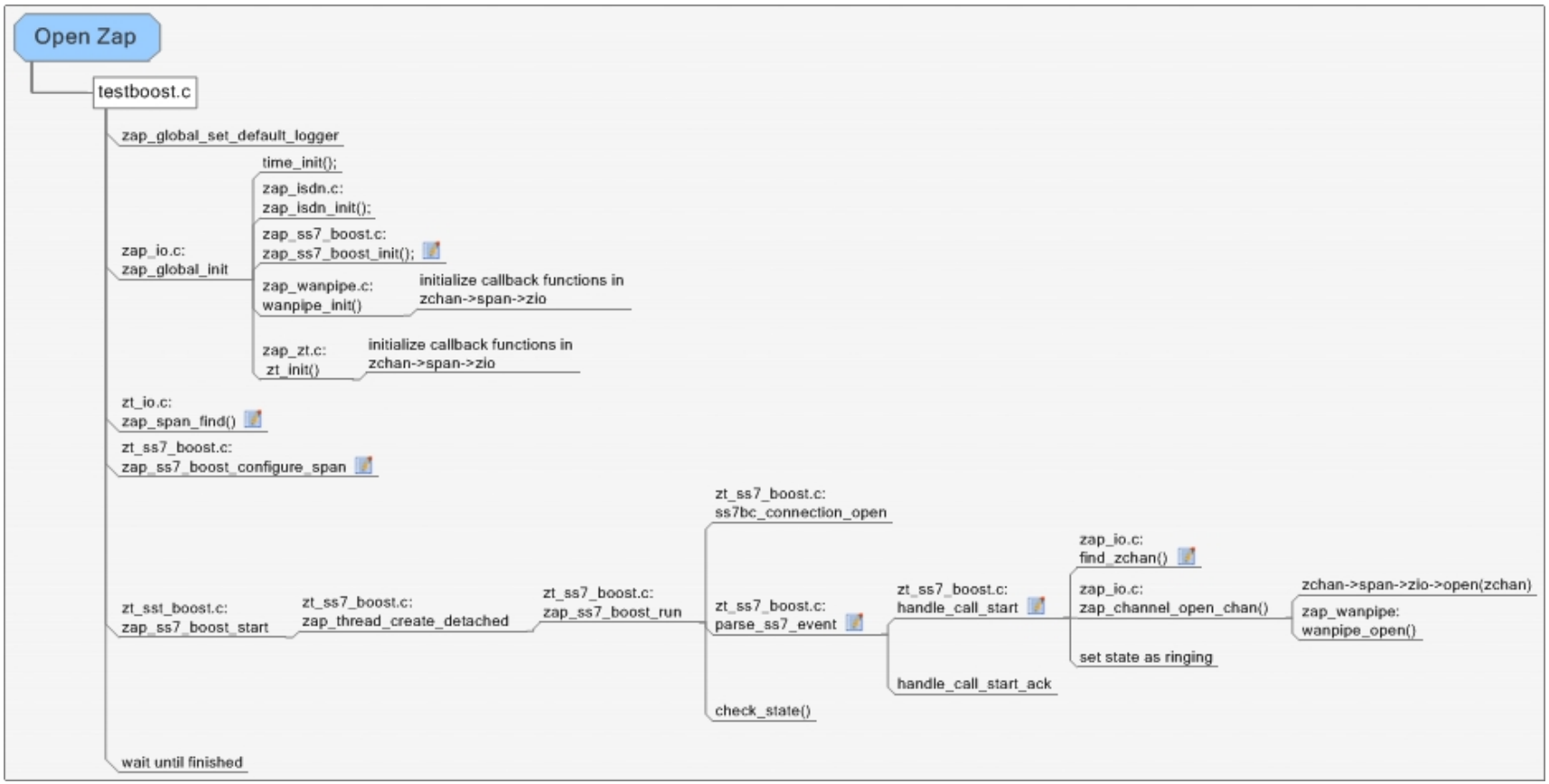
- **FreeSwitch OpenZap Config File**
  -> /usr/local/freeswitch/conf/autoload_configs/openzap.conf.xml
  <configuration name="openzap.conf" description="OpenZAP Configuration">
  <boost_spans>

  ….
  (Equivalent to /etc/asterisk/zapata.conf)

- **FreeSwitch Extension/DialPlan File**
  -> /usr/local/freeswitch/conf/dialplan/default.xml
  …
  (Equivalent to /etc/asterisk/extensions.conf)

- **wancfg_fs**
  (similar to wancfg_zaptel)

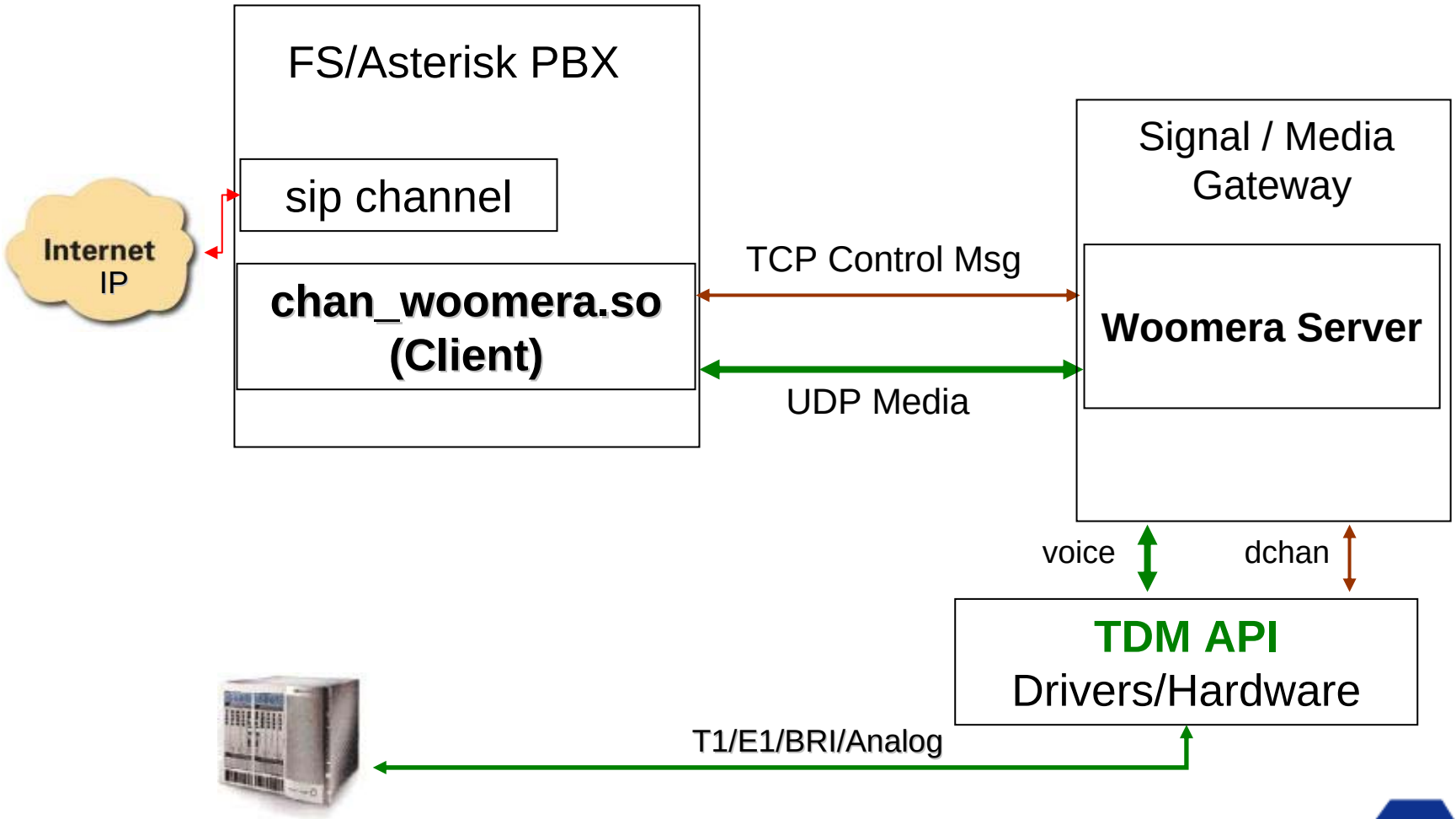
OpenZAP API


Open Zap
testboost.c
zap_global_set_default_logger
time_init();
zap_isdn.c:
zap_isdn_init();
zap_ss7_boost.c:
zap_ss7_boost_init();
zap_io.c:
zap_global_init
zap_wanpipe.c:
wanpipe_init()
initialize callback functions in
zchan->span->zio
zap_zt.c:
zt_init()
initialize callback functions in
zchan->span->zio
zt_io.c:
zap_span_find()
zt_ss7_boost.c:
zap_ss7_boost_configure_span
zt_ss7_boost.c:
ss7bc_connection_open
zap_io.c:
find_zchan()
zt_sst_boost.c:
zap_ss7_boost_start
zt_ss7_boost.c:
zap_thread_create_detached
zt_ss7_boost.c:
zap_ss7_boost_run
zt_ss7_boost.c:
parse_ss7_event
zt_ss7_boost.c:
handle_call_start
zap_io.c:
zap_channel_open_chan()
zchan->span->zio->open(zchan)
zap_wanpipe:
wanpipe_open()
set state as ringing
handle_call_start_ack
check_state()
wait until finished

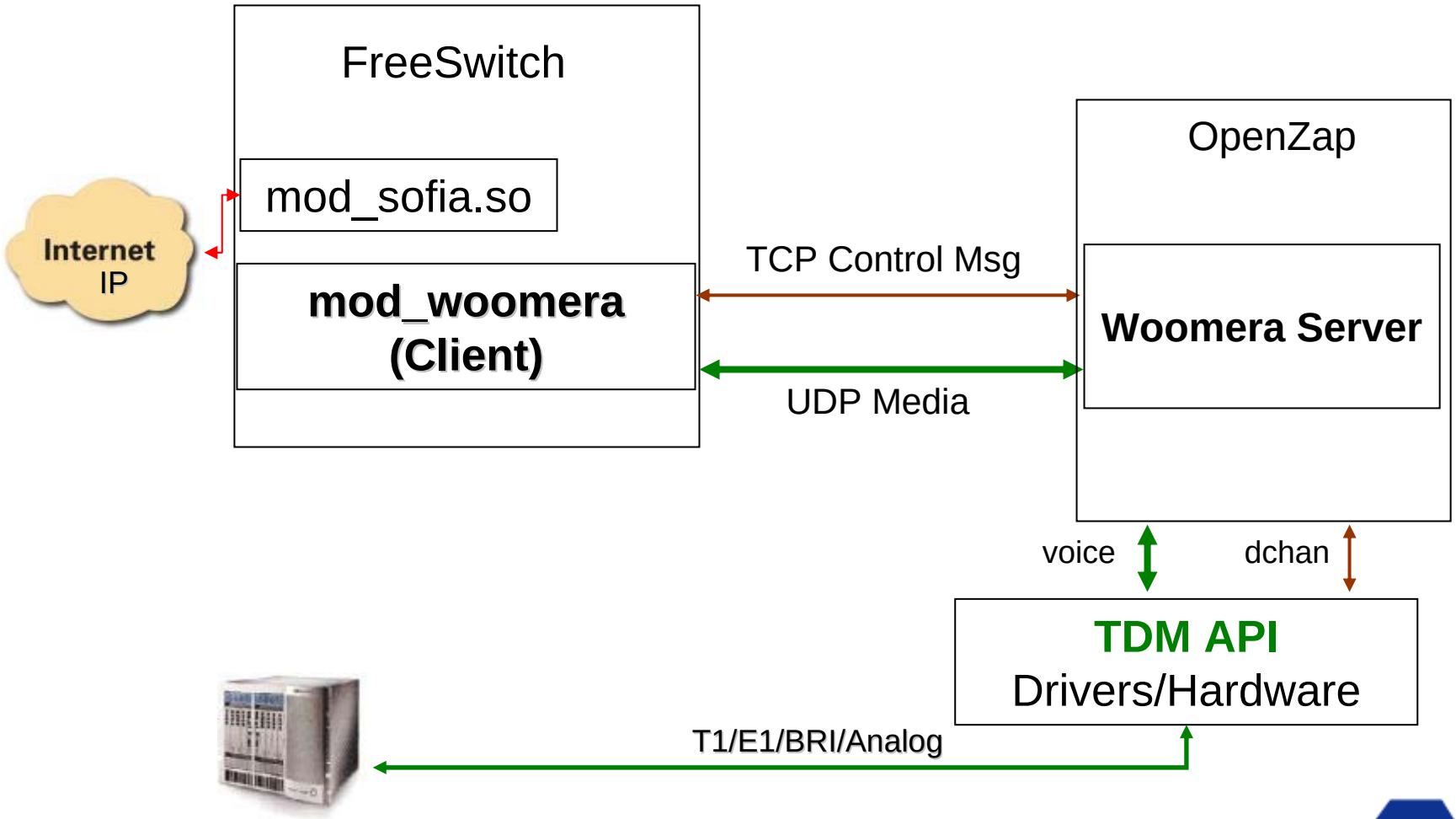
The Distributed TDM Architecture: Woomera

- Started as a response to a business need
- Providing a carrier-grade SS7 interface to Asterisk and others.
- A need to support 16 + E1 lines.
- Woomera Protocol: TCP Control Socket + UDP Media Socket (Woomera offers leverage)
- Design a Generic Channel Driver using Woomera for Asterisk.

Woomera FS/Asterisk PBX Model



Woomera FS/Asterisk PBX Model



Woomera Control Protocol

- TEXT based Call Control Messages (carried over TCP)
- HELLO, CALL, HANGUP, LISTEN, ACCEPT, ANSWER, DTMF, BYE/QUIT
- Each command is transmitted in ASCII text format

```
EVENT HELLO
Version: 1.0
Supported-Protocols:
h323,sip,iax.ss7,pri,bri
```

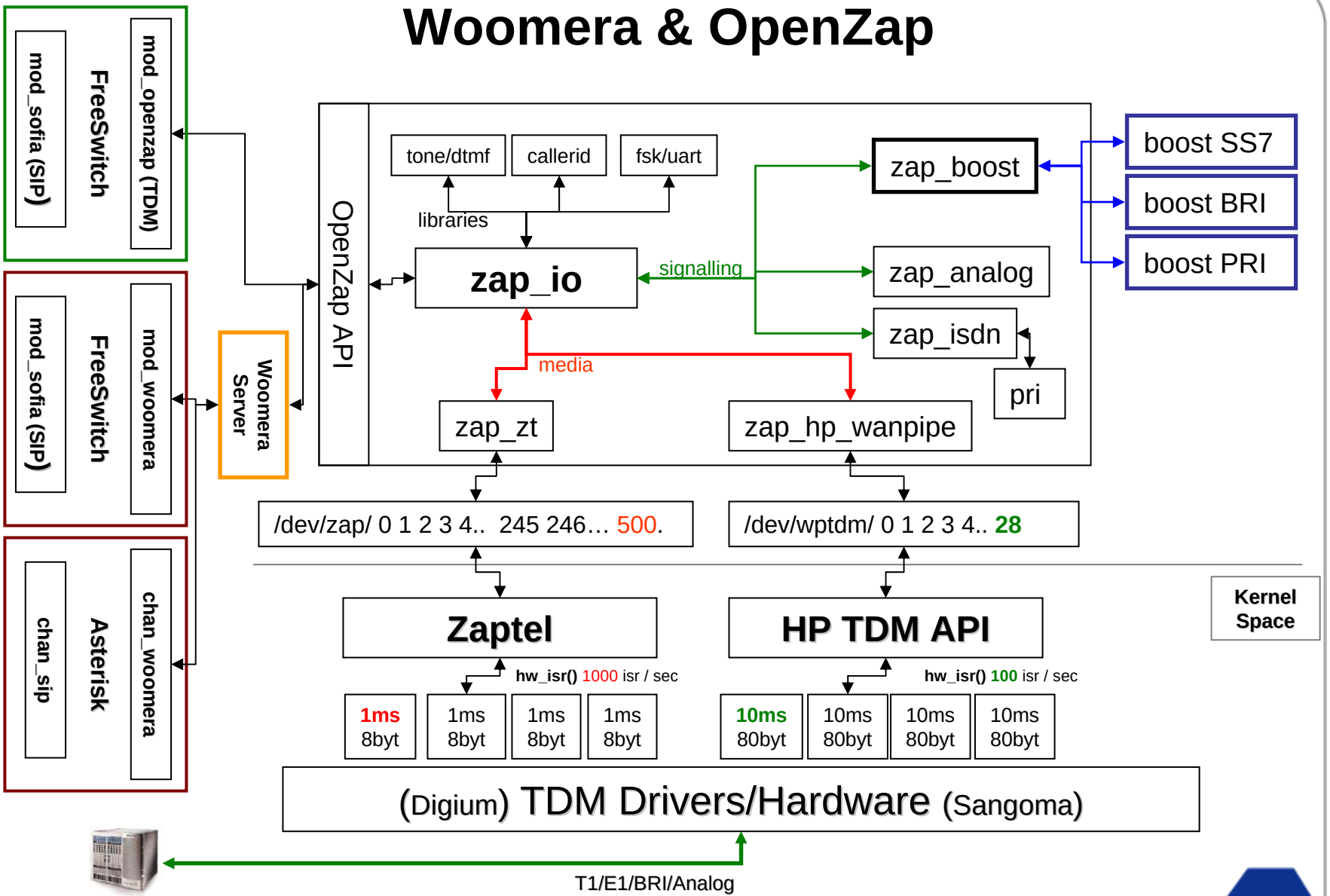
```
CALL
bri:g1/9054741990
Raw-Audio: 192.168.1.1:9000
Request-Audio: raw
```

```
ACCEPT
Unique-Call-Id: id1
```

```
EVENT MEDIA
Unique-Call-Id: id1
Raw-Audio:192.168.1.1:9000
```

- UDP based: Media (ulaw,alaw,PMC-16…)
- RFC in progress…

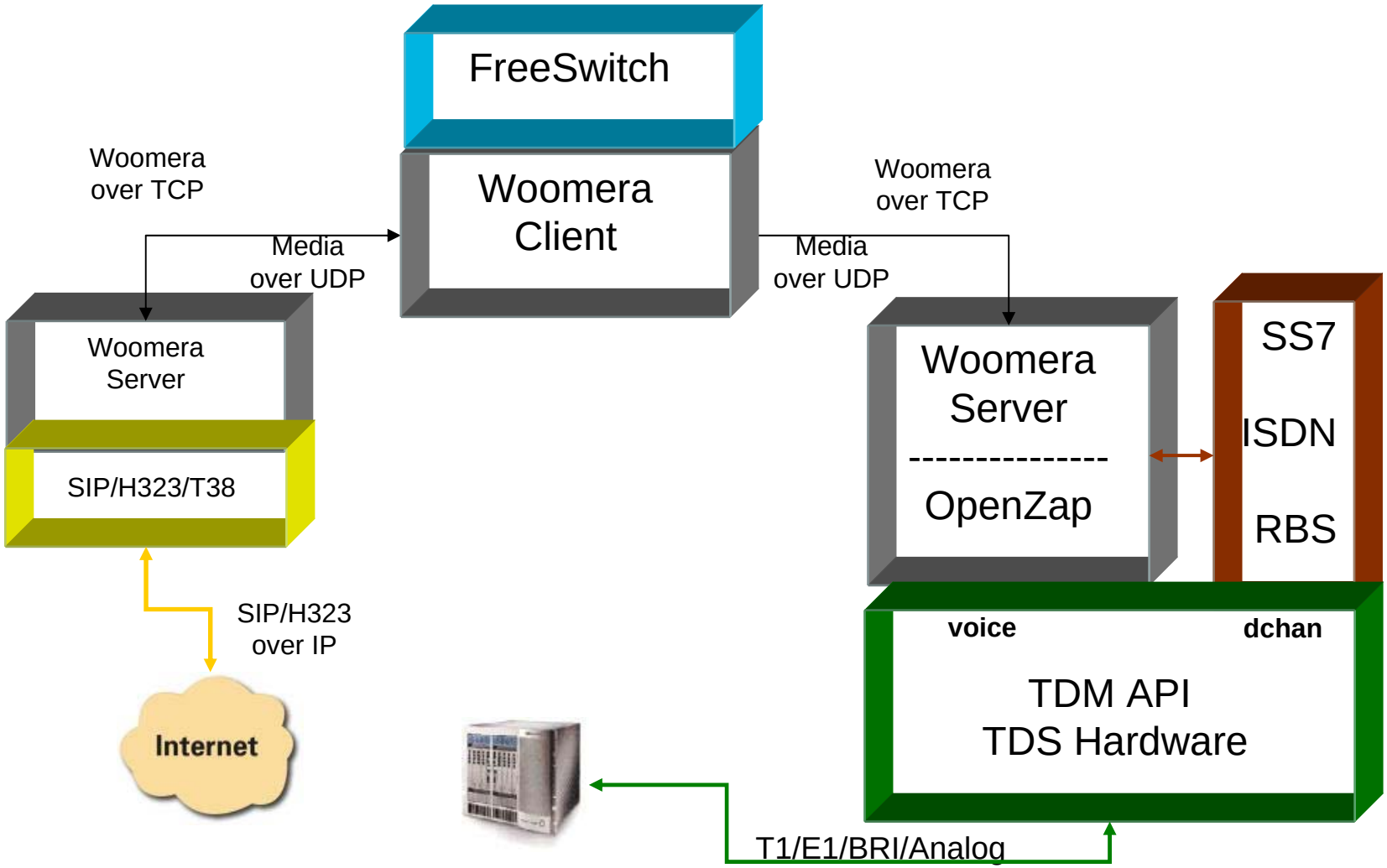
Woomera & OpenZap



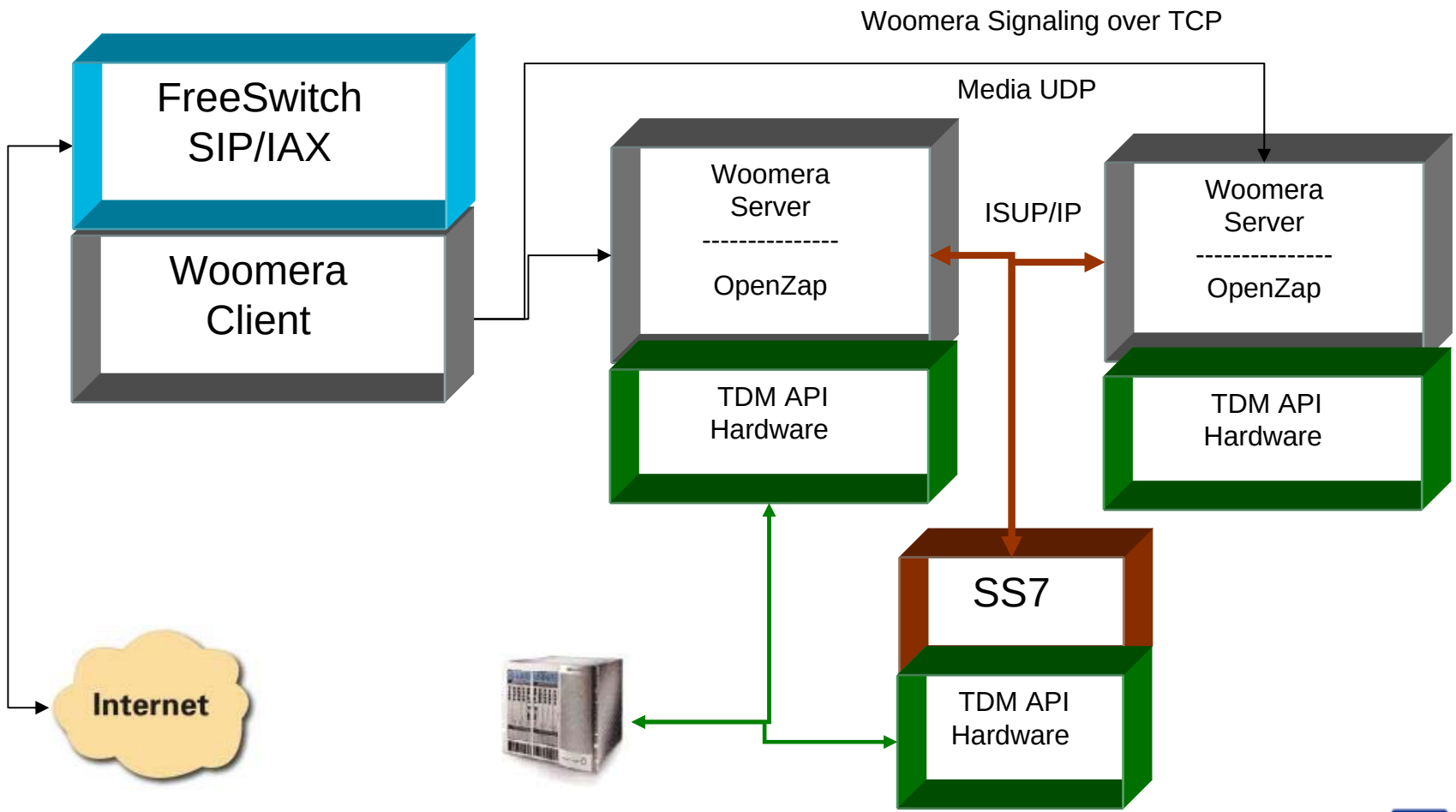
Object Oriented Telephony Design

- Each piece is well defined
- Reuse of debugged modules
- Reduced Bugs
- Increased Stability
- Increased Asterisk Stability

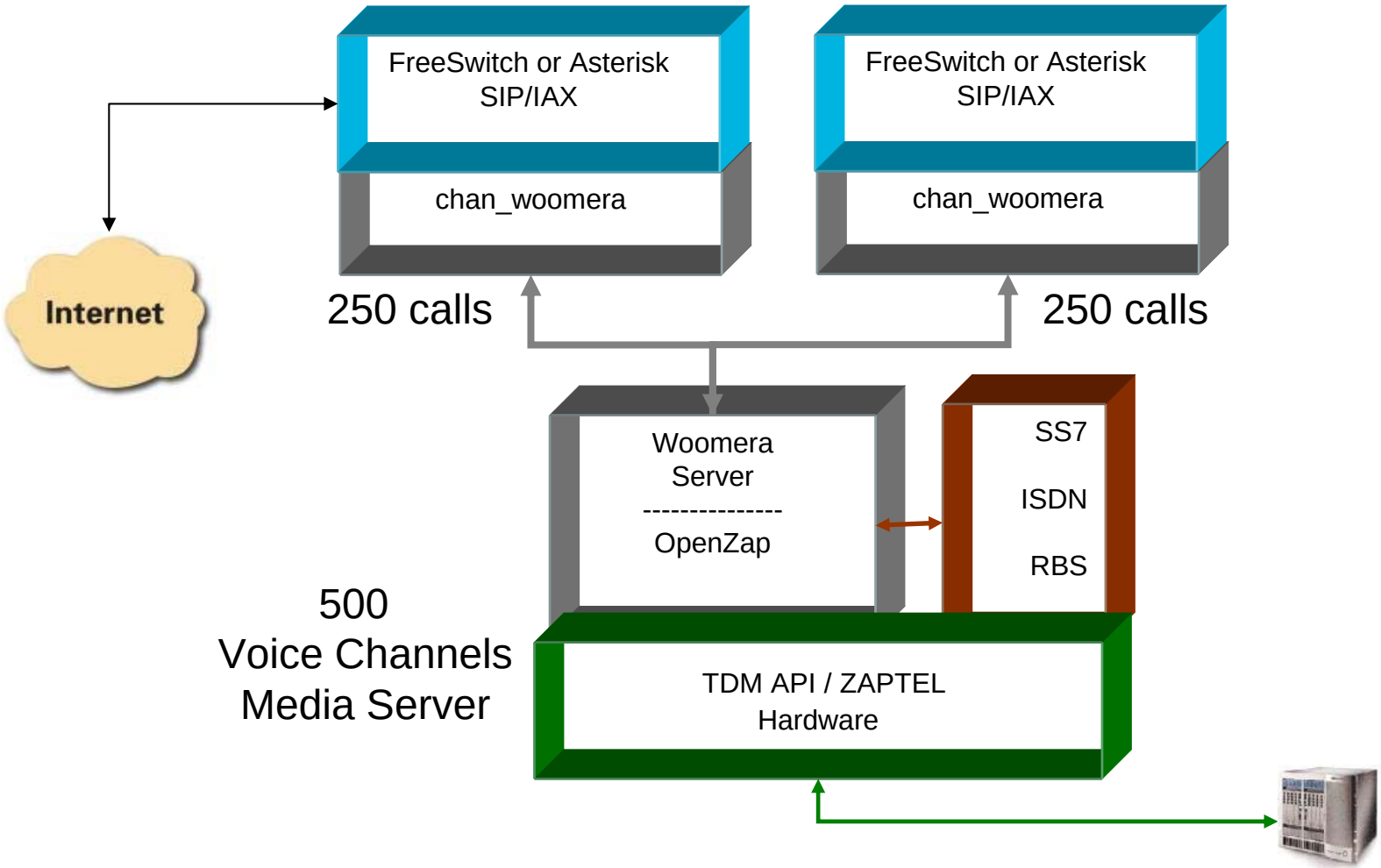
Distributed Woomera TDM Architecture



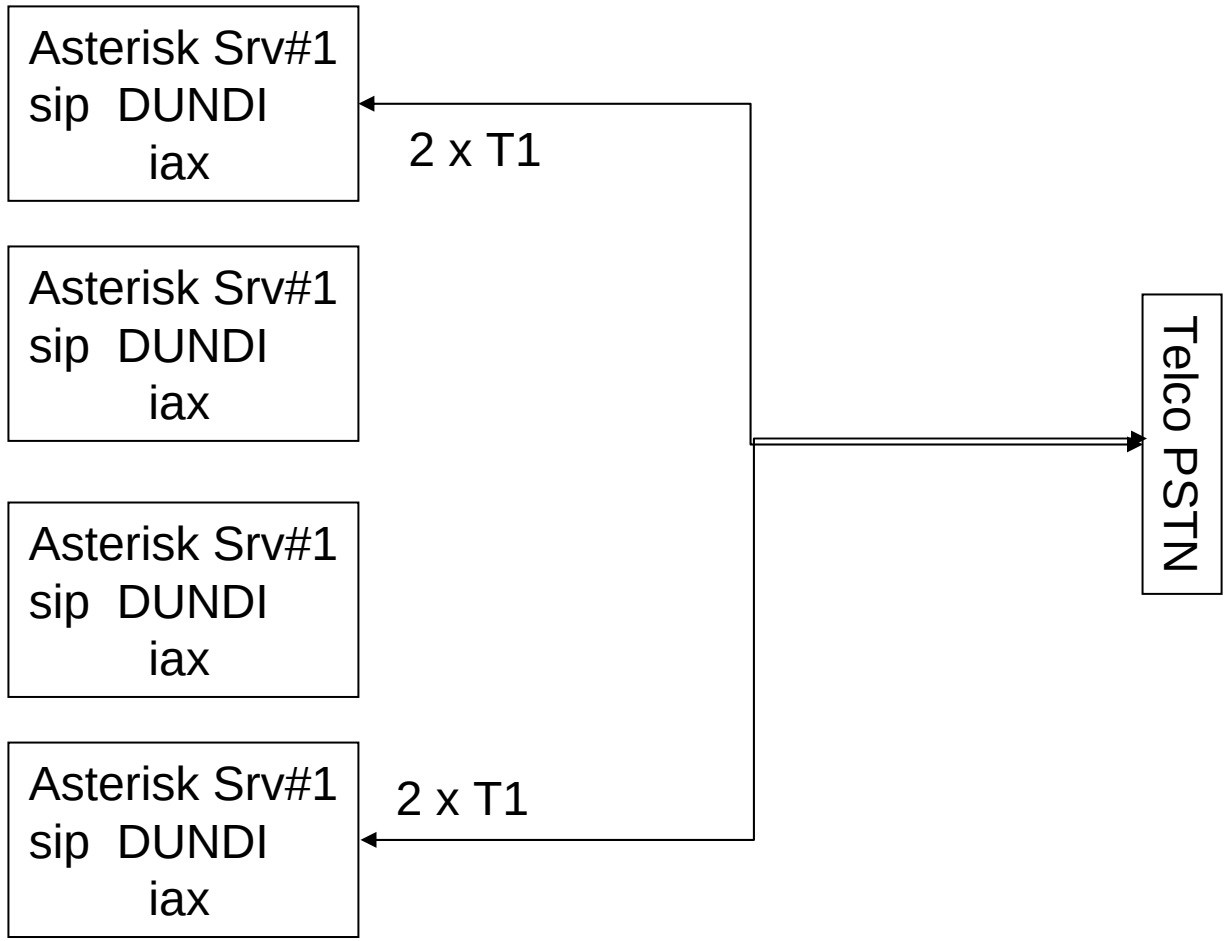
Distributed Clustering Woomera TDM Architecture



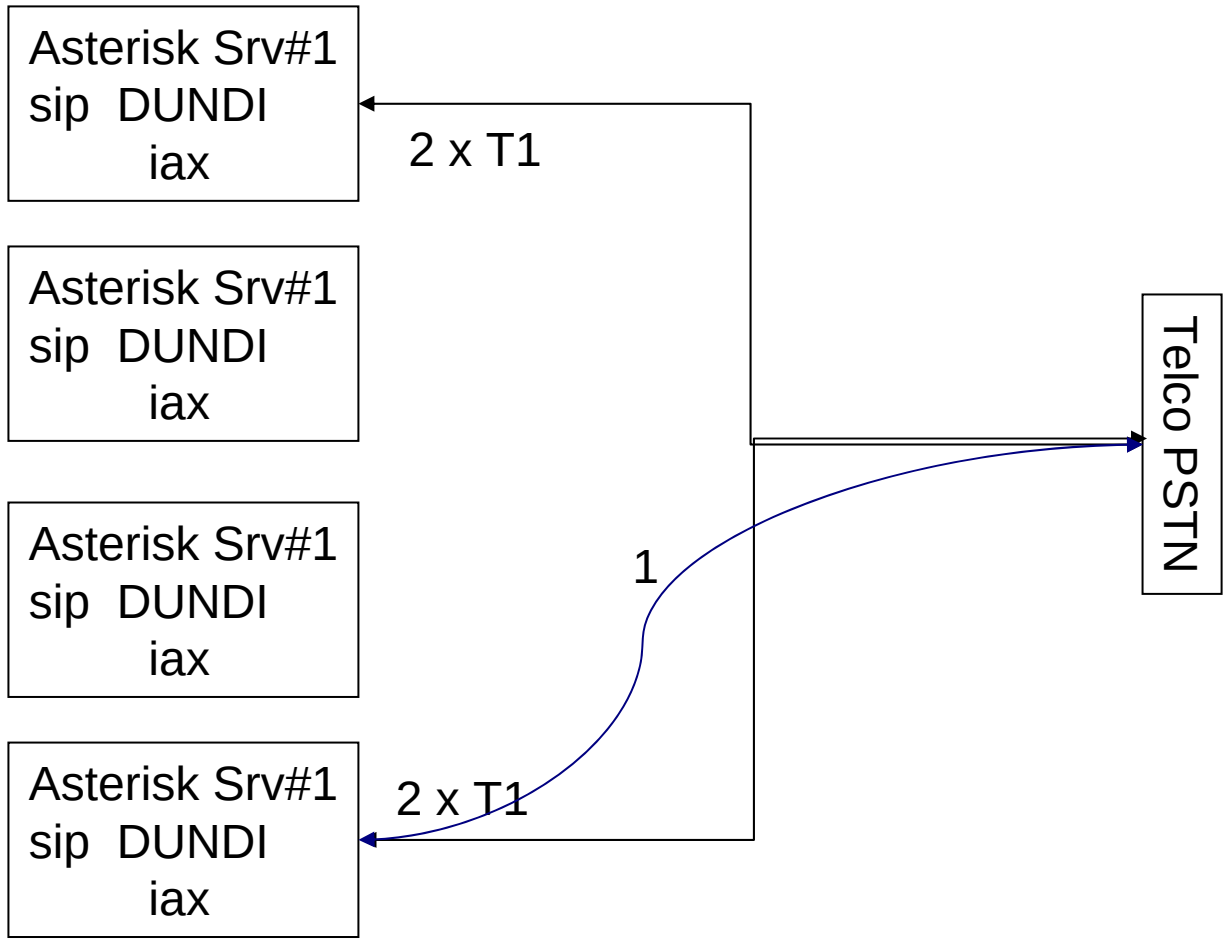
Auto FreeSwitch Load Balancing Scaling



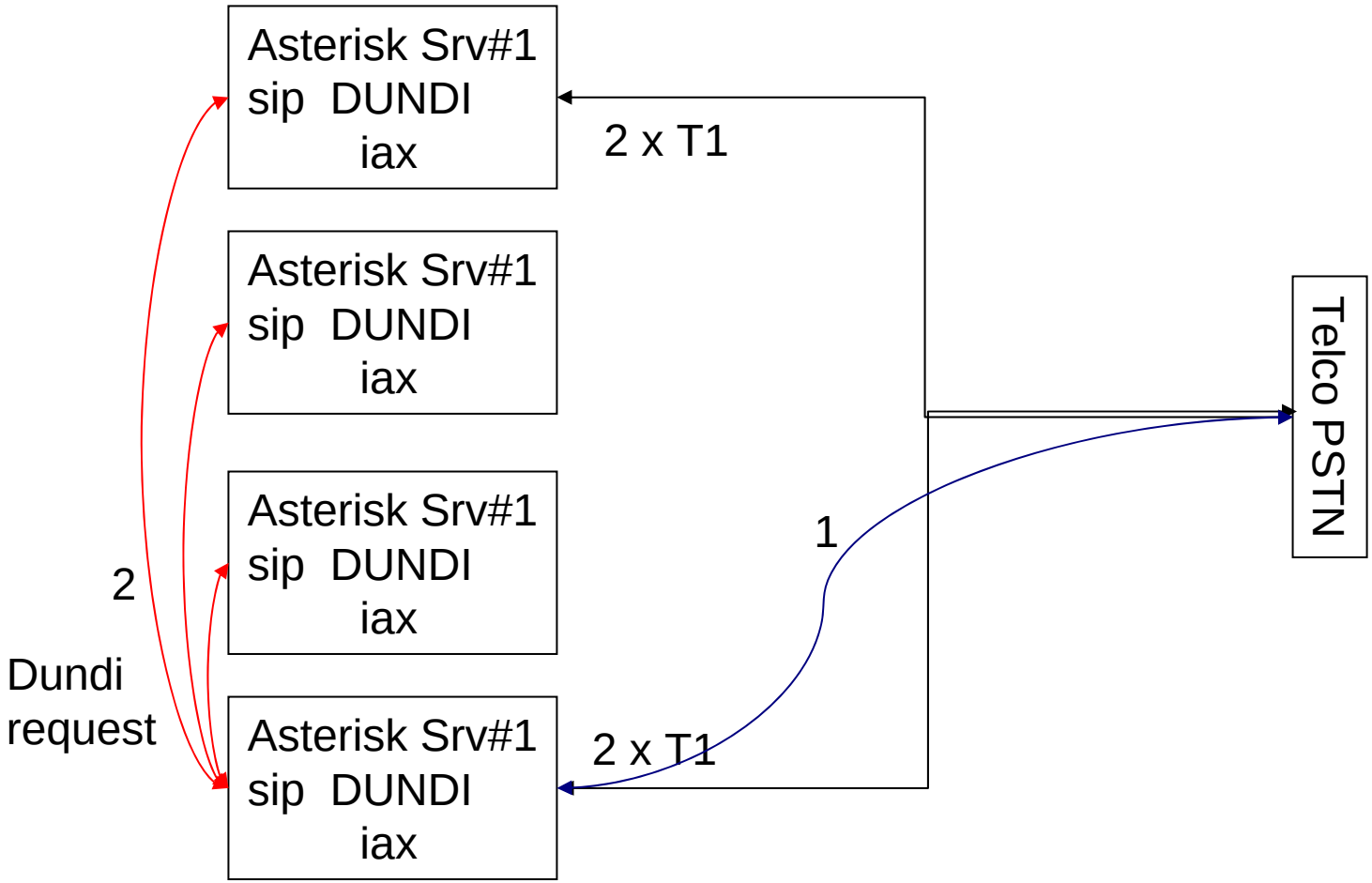
DUNDi Clustering



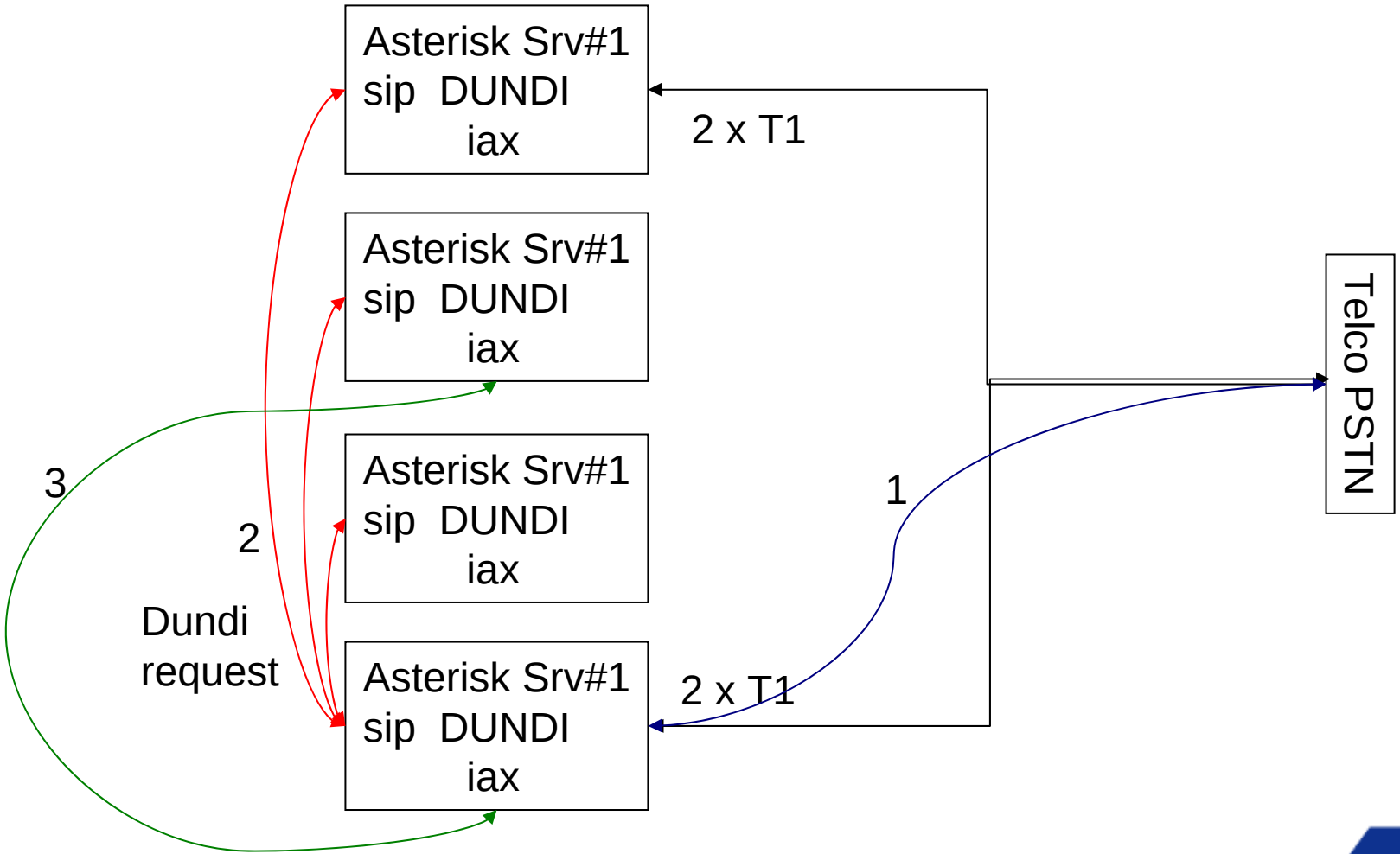
DUNDi Clustering



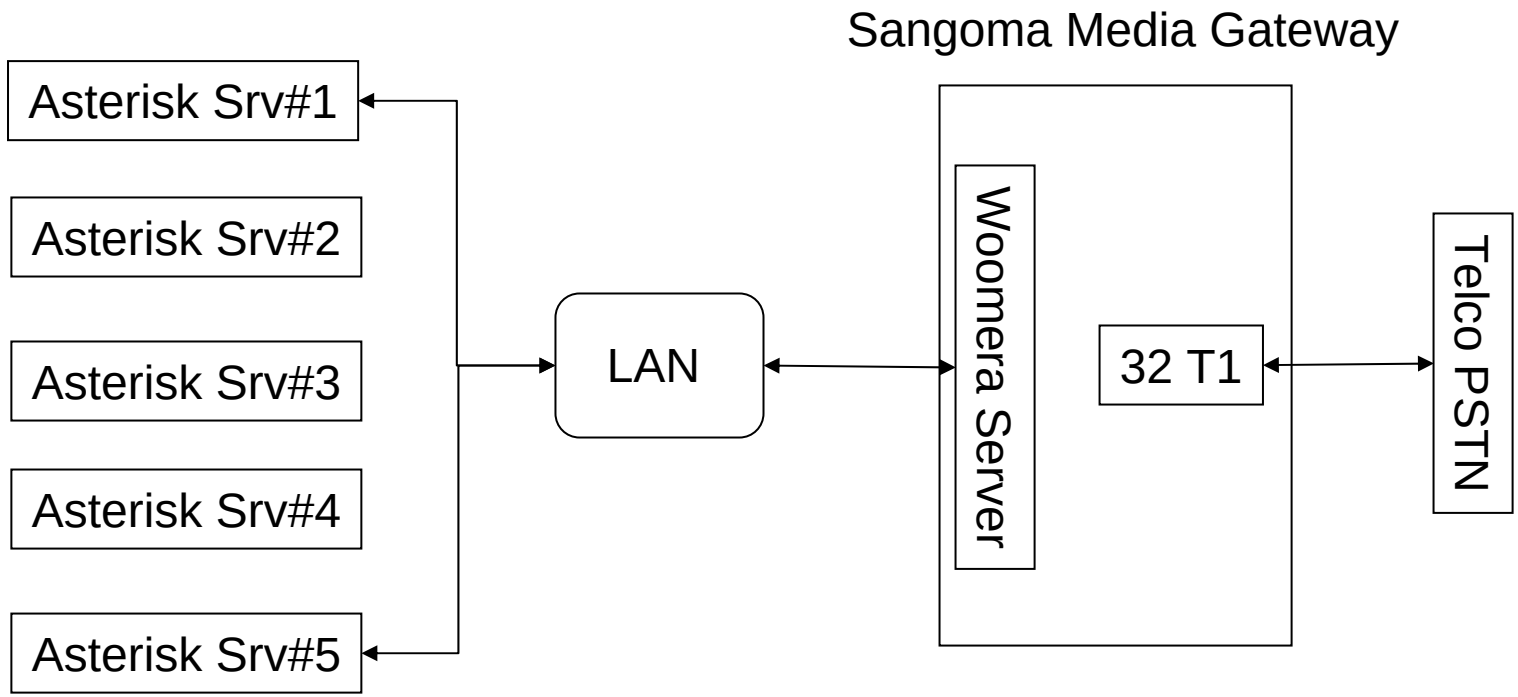
DUNDi Clustering



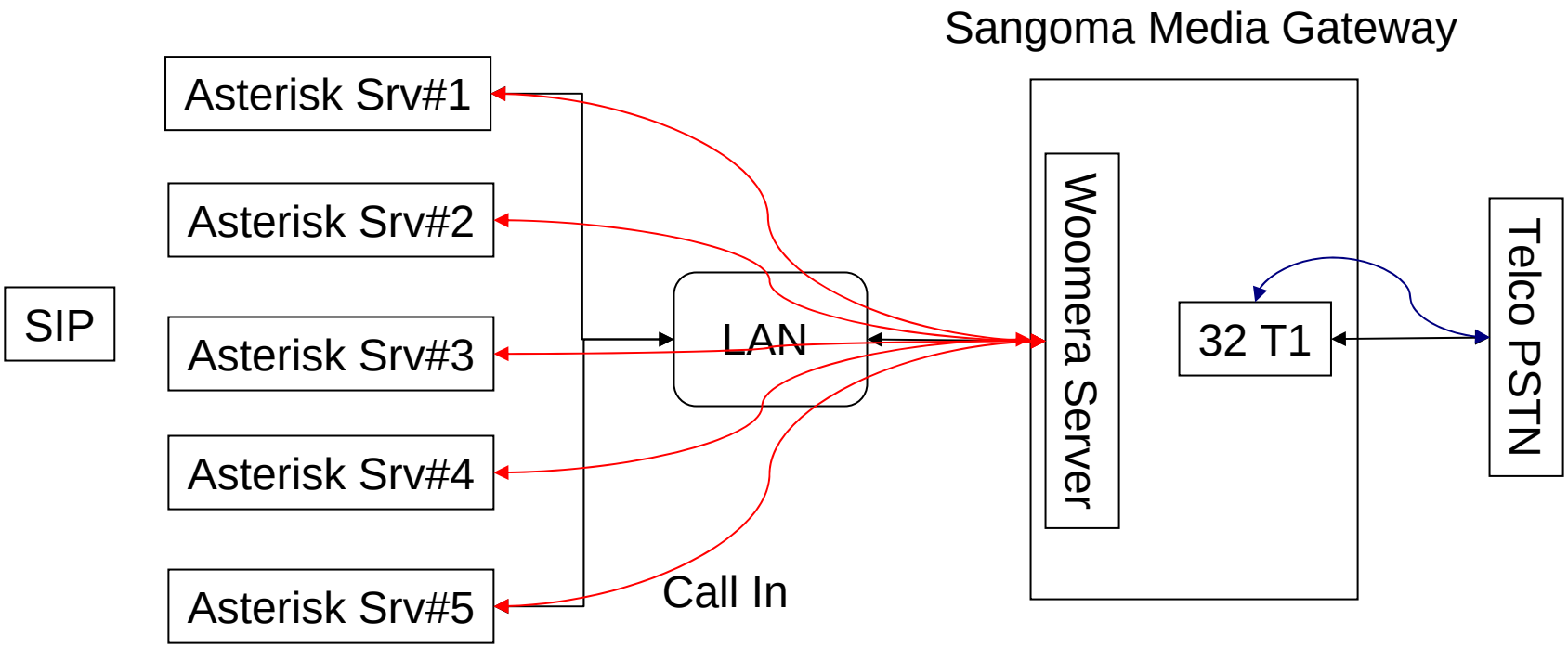
DUNDi Clustering



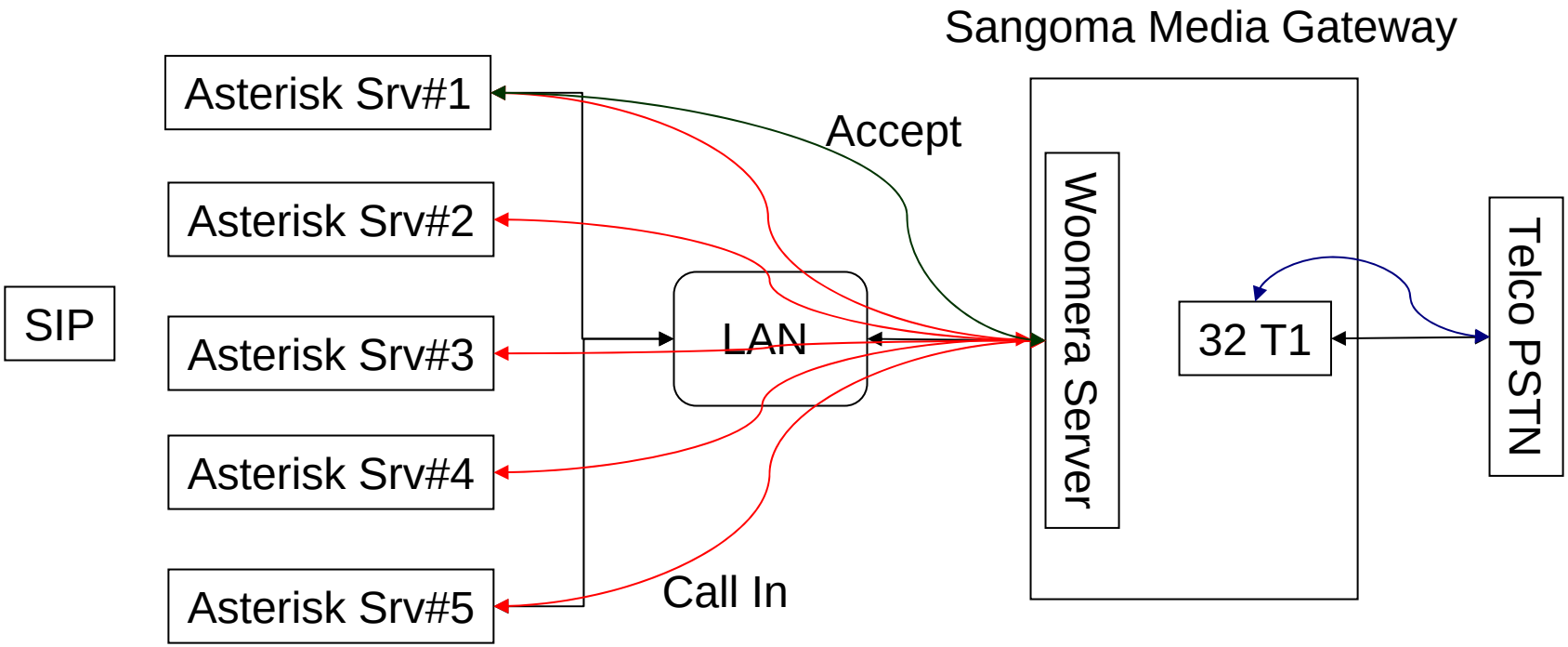
Woomera Clustering



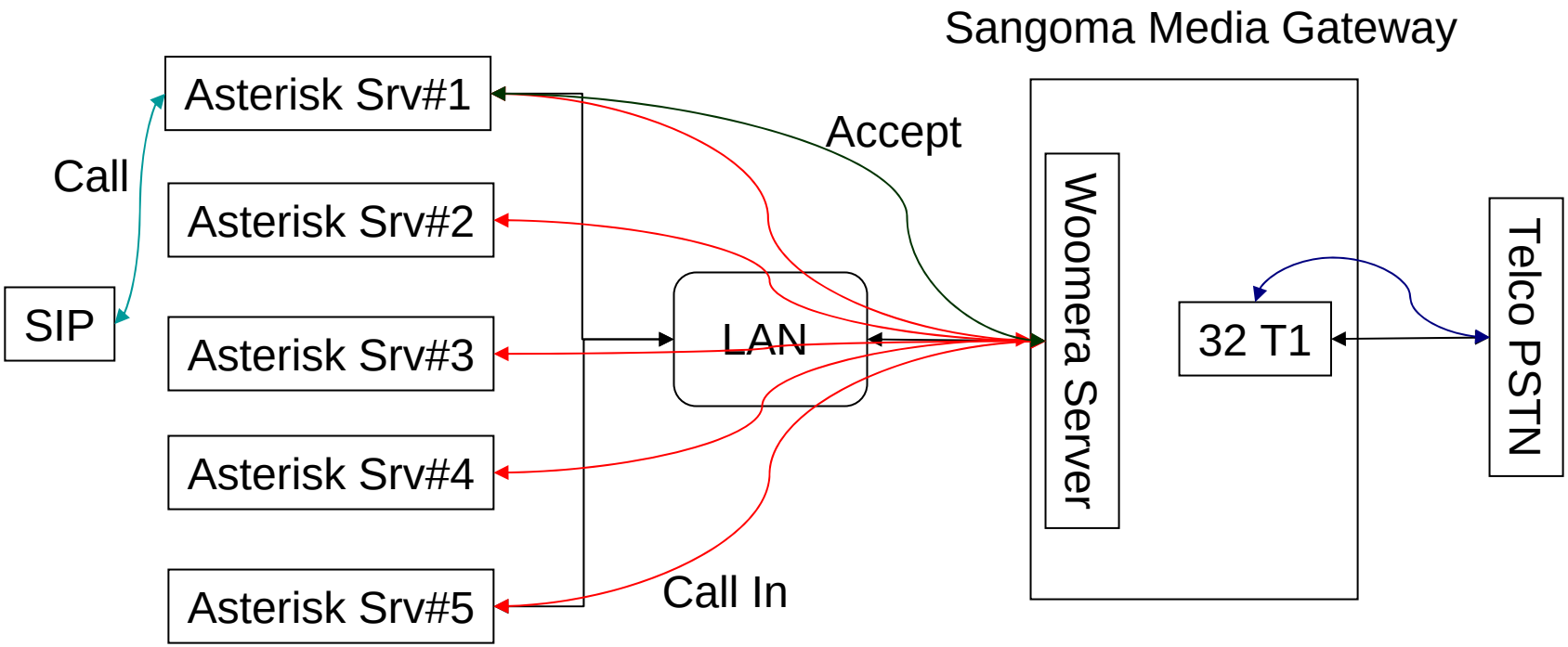
Woomera Clustering



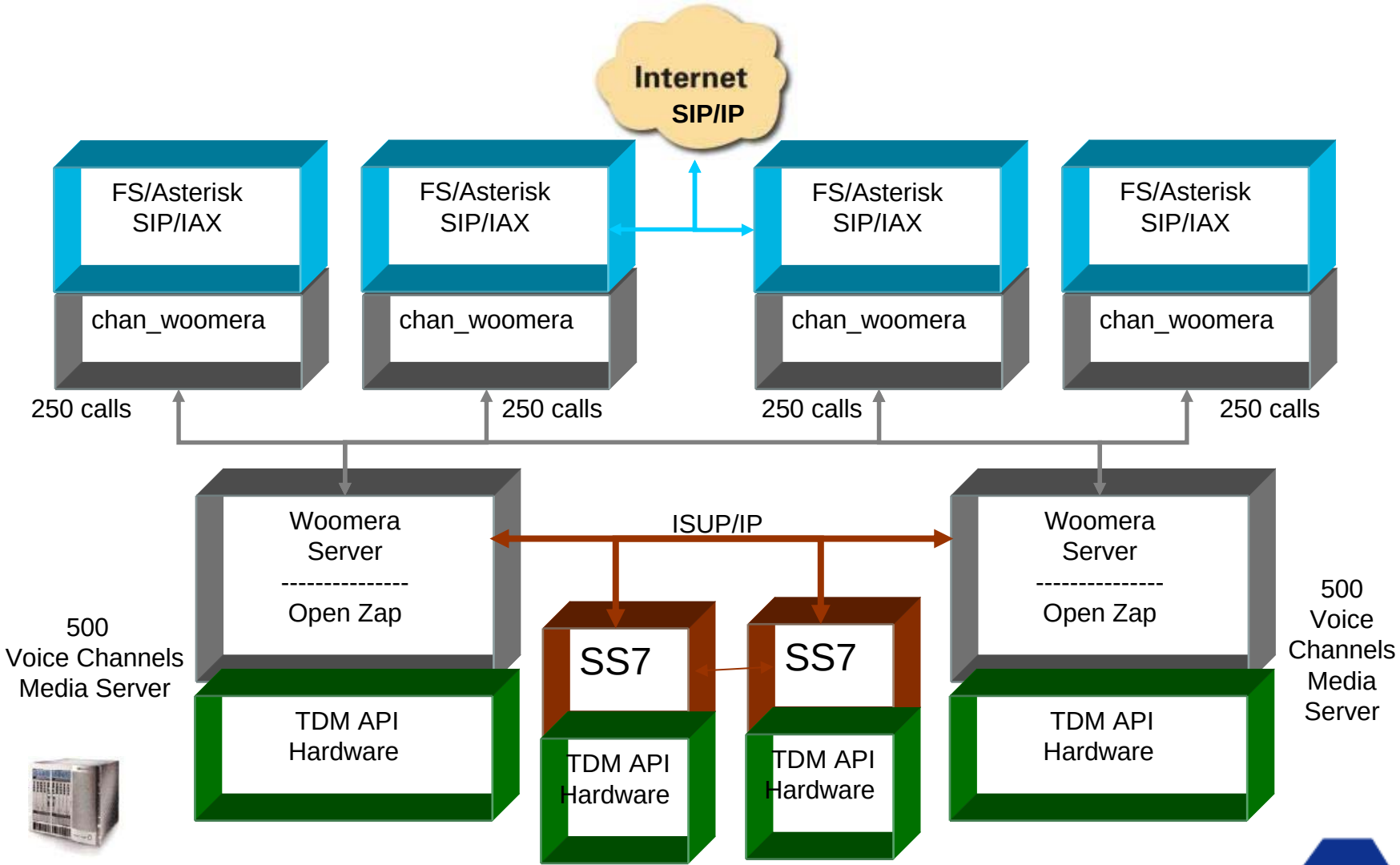
Woomera Clustering



Woomera Clustering



Distributed Clustered Scaled Asterisk & SS7



Distributed Performance

- **FreeSwitch/Asterisk can handle 1000+ SIP-SIP calls**
- **Woomera channels are lighter than SIP/RTP**
- **FreeSwitch/Asterisk should be able to handle 1000+ distributed Woomera TDM to TDM or SIP to TDM calls.**
- **Lab Tests – Single System:**
  sipp → Asterisk -> OpenZap -> 16E1
  Total Calls: 495
  Calls Per Sec: 100
  System: Quad Core 2.4Ghz, 2GB Memory
  Performance: 60% idle

Future of Open Source PBX/Gateways

- **Demand is growing!**
- **Need for 32 + E1 media gateways.**
- **Distributed, Object model works**
- **Open Source is the answer: FreeSwitch and/or Asterisk**
- **Telco Grade Design!**

Whats new at Sangoma: Software

- **Netborder SIP-SIP & SIP-PSTN Gateway**
  Linux & Windows
  Fully Integrated Product / Web GUI
  Highly Scalable – 32E1s – 1000+ call gateway
  Single Quad Core – 50% usage on 1000 SIP to PSTN calls.

- **Abstracted Sangoma Signaling Stacks for Asterisk & FreeSwitch**
  PRI – Telesoft
  BRI – Netbricks
  SS7 – ss7box
  OpenR2 – libopenR2

- **OpenR2 For Asterisk & FreeSwitch**
  libopenr2 has been integrated and committed into Asterisk tree.

- **Sangoma Unified TDM API (LibSangoma)**
  SPAN Mode - (NetBorder, Yate, FreeSwitch)
  CHAN Mode - (Asterisk, FreeSwitch)
  Common API for Linux & Windows

- **MTP2 API**
  Mobile SS7 Applications through third party vendors

- **Woomera Library (LibWoomera)**
  Standardized Woomera Implementation added to chan_woomera.

Whats new at Sangoma: Hardware

- **B700 – Mixed BRI & Analog**
  - with octasic hwec
- **B600 – Low cost 4FXO/1FXS Analog**
  - with octasic hwec
- **B601 – Mixed T1/E1 & Analog**
  - with octasic hwec
- **USB FXO**
  - with microchip hwec
- **VoiceTime USB Timing Stick**
  - Asterisk timing using VoiceTime
- **All Hardware Supports Multi Chunk Sizes**
  - much more efficient interrupt usage
- **Infinite Buffering**
  - zero data drop on high latency interrupts


Thank You!
Questions and Comments?