SAMSUNG

Accelerated SoC level Android Home Screen Bringup, System Software Development and Validation at Pre-Silicon with Advanced Hybrid Emulation Methodology

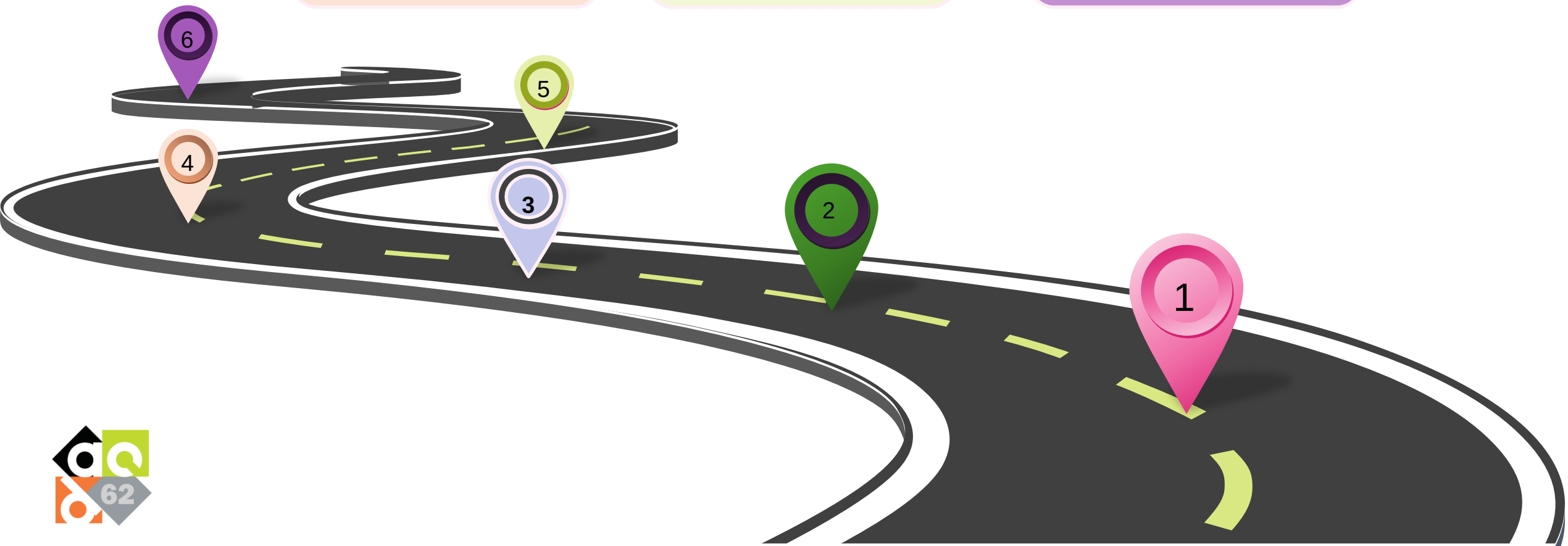
Samsung Semiconductor India Research

Sarang Kalbande (sarang.mk@samsung.com)
Rinkesh Yadav (rinkesh.y@samsung.com)
Garima Srivastava (s.garima@samsung.com)
Hyundon Kim (hyundon.kim@samsung.com)

62 THE CHIPS TO SYSTEMS CONFERENCE

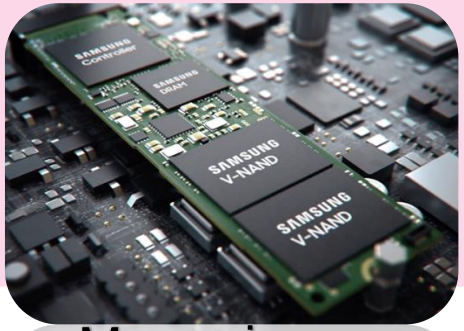
SPONSORED BY CEDA® IEEE Council on Electronic Design Automation SIGDA acm

1. Introduction & Motivation
2. Methodology
3. Techniques
4. Evidence
5. Results Summary
6. Future Scope

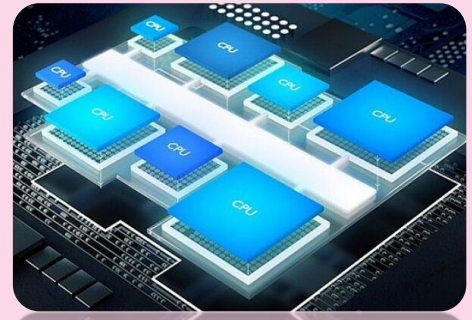


Introduction

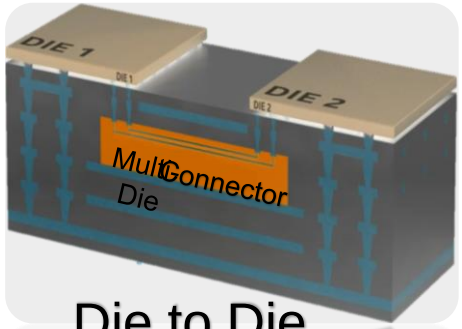
- Growth in Semiconductor Industry
- Rapidly increasing complexity in modern SoC designs



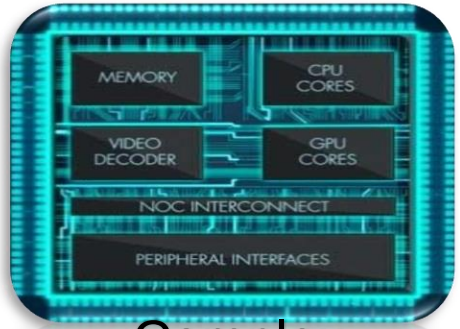
Memories



MultiCore
CPUs, GPUs,
NPUs, DSPs



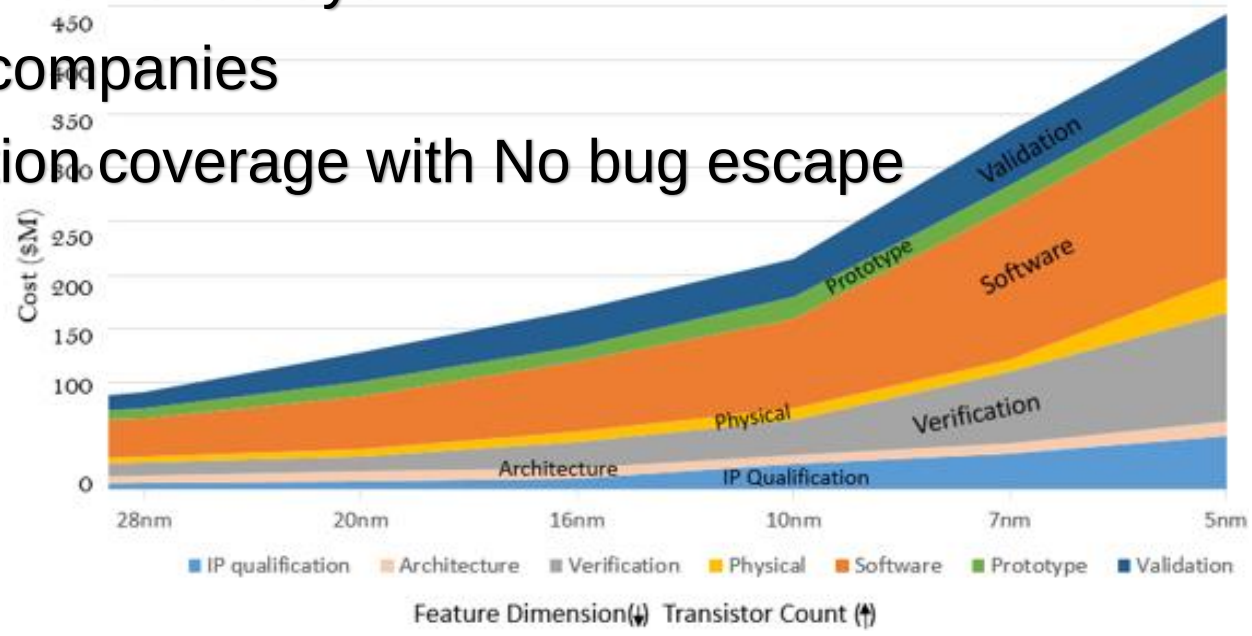
Die to Die



Complex
Subsystems

Introduction

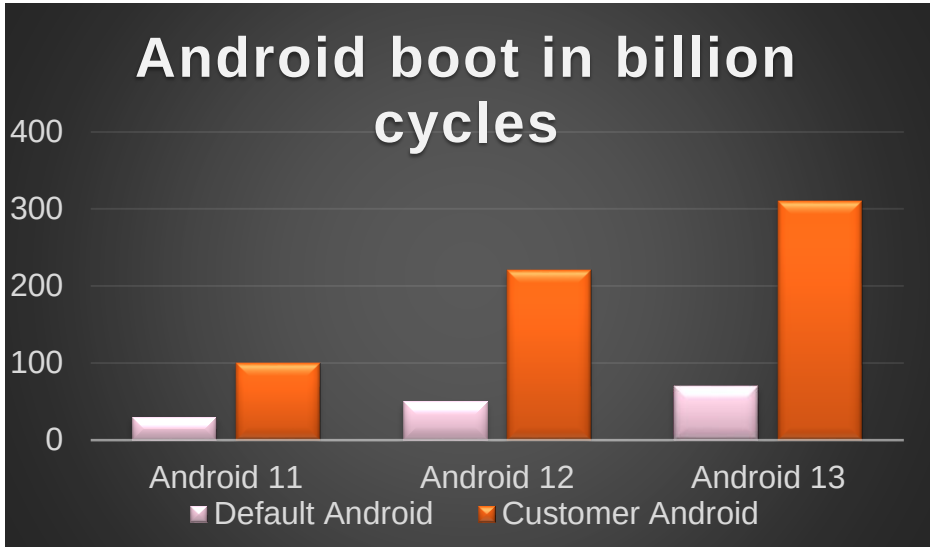
- Growth in Semiconductor Industry
- Rapidly increasing complexity in modern SoC designs
- Increase in the complexity and cost of software development and validation
- Higher risk of hardware and software bugs during silicon implementation
- Late-stage bug discovery leads to time-to-market delays
- Elevated risk of costly silicon re-spins for companies
- Ensuring maximum System Level verification coverage with No bug escape
- Challenge to Validate chip on time



Motivation

- Design re-spins and significant revenue losses
- New validation strategy ,an Advanced Hybrid methodology to address said limitations,
- Focused on:
    1) Early Android home screen bring-up with Acceleration
    2) Complete System software development before silicon readiness
- Benefits of this approach:
    1) Faster issue detection
    2) Improved confidence in design early at Pre-silicon
    3) Reduced risk of costly silicon re-spins
- Developed this methodology specifically for large-scale SoCs exceeding ~1.5 billion gates.

Challenge to bring-up software on Pure Emulation



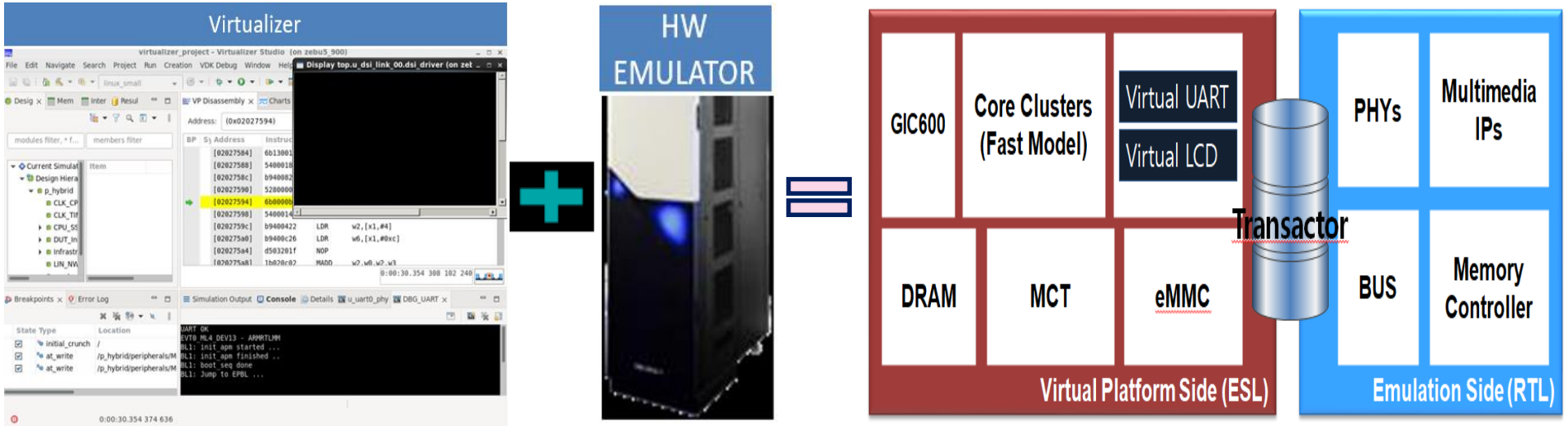
Android 13 boot time (cpu @3Mhz, 4cycles/ instruction):

(310Bx4/3 MHz)/3600s = 114 hours (~ 4 days)

- Pure Emulator driver clock frequency is not sufficient to bring-up Linux kernel, Android OS boot and to develop system level software.
- Takes ~114 hours to bring-up Android 13 where SoC is running on 3Mhz emulator driver clock frequency
- To overcome less emulator speed and to accelerate software development, Hybrid emulation is introduced.

What is Hybrid Emulation ?

- Hybrid emulation combines Virtual prototypes and Hardware emulator.
- One part of the SoC design is run at the emulator and the other part is run at virtual platform.
- Virtual prototypes are high performance, System C models of particular IP, a block or an entire SoC as per requirements.
- The task of virtual platform is to have enough accuracy to support the level of software being run on it. This is achieved by modelling the behaviour blocks and inter-block communication at transaction level(TLM), which makes these faster than equivalent cycle-accurate RTL.



Main Idea-Methodology

We developed several below Techniques and Customizations to accelerate the Android home screen boot process on Hybrid Emulation, including:

1. Migrating ARM Fast Model (AFM) to a QEMU-based virtual CPU subsystem.
2. Utilizing High-Speed smart virtual memories
3. Smart Design partitioning of SoC components between Hardware (RTL) and virtual environments
4. Introducing Advanced Clocking Techniques.
5. Hardware-Software Profiling.
6. Moving Boot Devices to Virtual Side.

Techniques and Customizations

1. **Migrating ARM Fast Model (AFM) to a QEMU-based virtual CPU subsystem**

- ARM Fast models are System C/TLM they run in single thread ,executing sequentially
- QEMU is C/C++ and multithread capable
- QEMU helped accelerating Android boot as it helped spawn multiple threads.

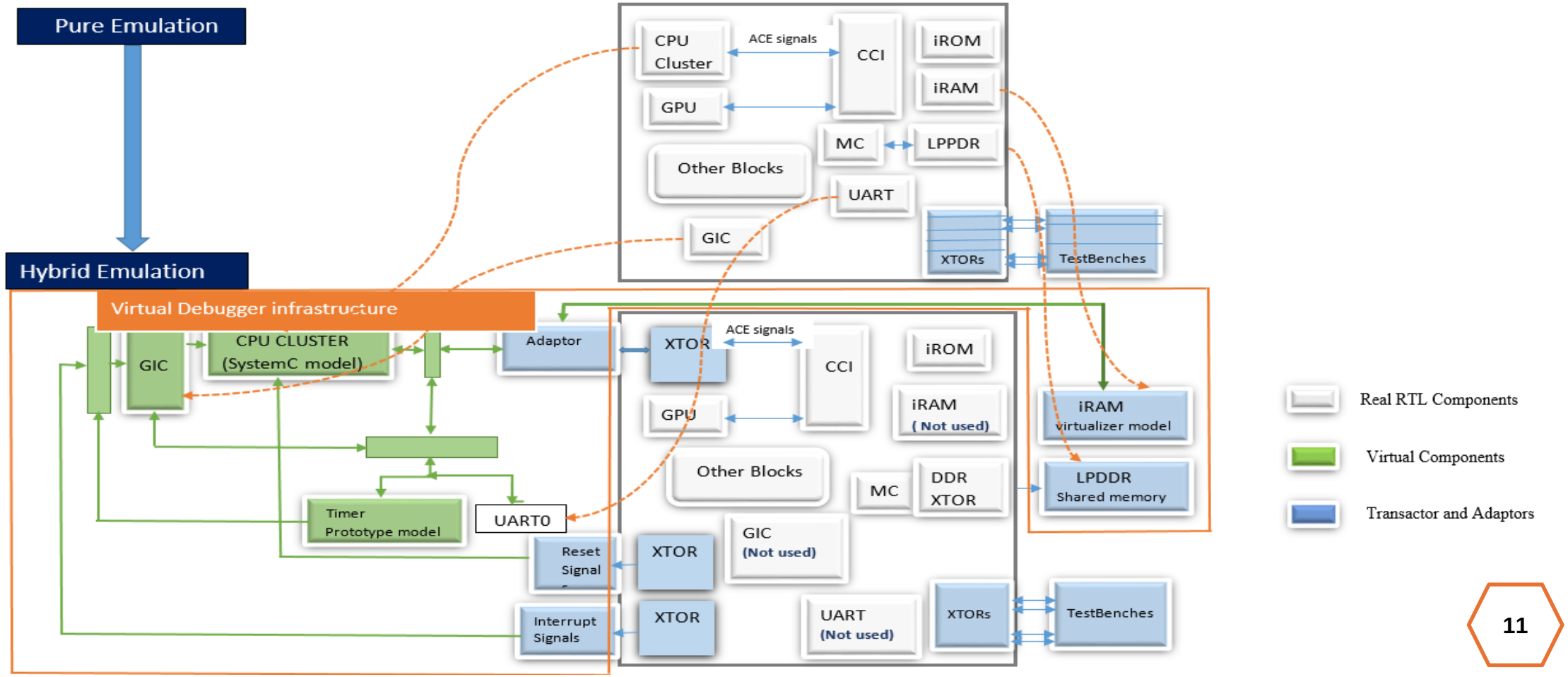
2. **Utilizing high-speed smart virtual memories**

- Different IRAM,IROM memories involved in SoC Bootflow have been migrated from Pure RTL to Virtual side for faster Boot

Techniques and Customizations_Cont..

3. Smart Design Partitioning of SoC components between Hardware (RTL) and virtual environments.

This partitioning is dependent on user bringup and verification need



Techniques and Customizations_Cont..

4. Introducing advanced clocking techniques

- RTL and Virtual side IPs (CPU,GPU,DPU,MIF) smart clocks synchronizations to gain acceleration
- Configuring Highest frequency design clk to half to increase emulator run time throughput.
- Modifying Virtual CPU System C model operating frequency in MHz

5. Hardware-software profiling

- Various Profiling tools and techniques used to speedup boot
- Software-Centric – collect runtime stats of Function call frequency,CPU cycles ,Cache hit/misses, Latencies combined with emulator hooks to track execution addresses, instruction counts and memory accesses
- Hardware-Centric – Performance monitors to see cache behaviour,TLB misses,Branch prediction,CPU cycles, Vendor specific switches
- Memory & I/O Access Tracing- TLM transaction level trace handlers
- Tools like **Perfetto** for system wide profiling

6. Moving Boot devices to Virtual Side

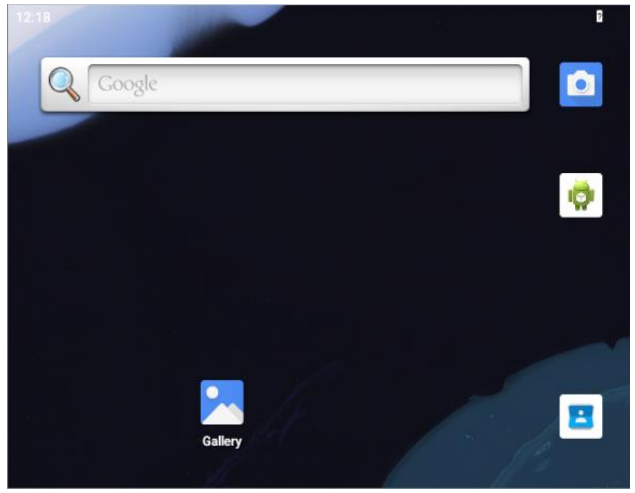
- Real boot devices like SDCard ,UFS etc have been migrated to Virtual side for faster boot.

Evidence

These are Snapshots of Android Logo and Android Home Screen images
captured on Hybrid Platform for Samsung Mobile Exynos SoC Design.

- Version : Android 16, Kernel 6.12
- Time to Home Screen : 20min
- Service count : 249
- Resolution : 640x480
- All s/w images were loaded from virtual UFS like silicon.

Android Logo



Android Home Screen

Results Summary

Android Home Screen Boot time comparison between different Pre-Silicon Conventional Methods

	Simulation	Pure Emulation	Hybrid Emulation (AFM)
Environment Initialization	4 Min	5 Min	5 Min
Kernel Boot-up (Prompt)	125,865 Min (2097 hrs)	400 Min (6 hrs)	6 Min
Android Logo	231,000 Min (3850 hrs)	1200 Min (18 hrs)	36 Min
Android Home Screen	510,517 Min (8508 hrs)	3200 Min(53 hrs)	143 Min
Total Consumed Time	867,384 Min (14456 hrs)	4805 Min (80 hrs)	190 Min (~3 hrs)
Clock Frequency	Few Hz	3 MHz	~ 59.9 MHz

x20

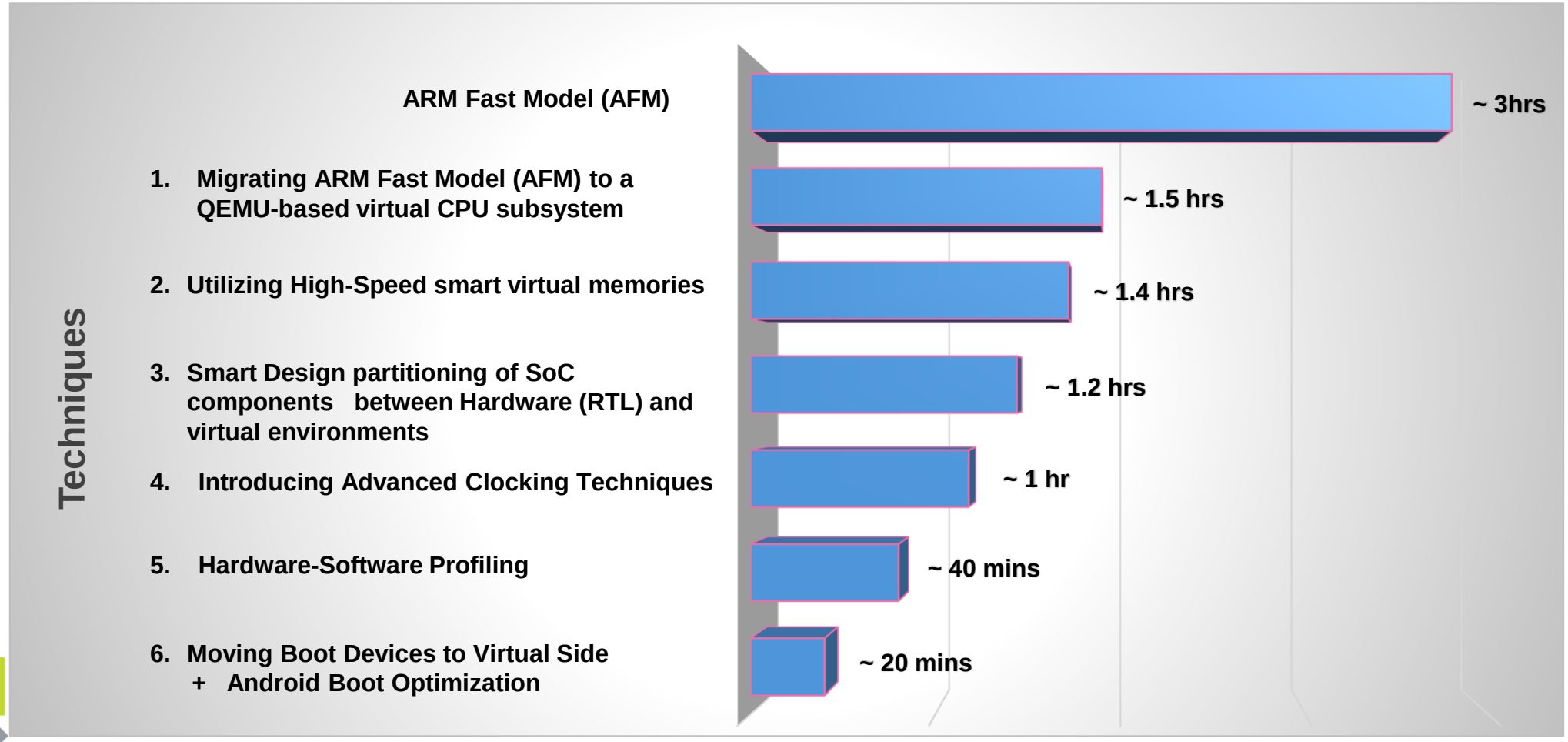
Main Idea-Methodology

We developed several below Techniques and Customizations to accelerate the Android home screen boot process on Hybrid Emulation, including:

1. Migrating ARM Fast Model (AFM) to a QEMU-based virtual CPU subsystem.
2. Utilizing High-Speed smart virtual memories
3. Smart Design partitioning of SoC components between Hardware (RTL) and virtual environments
4. Introducing Advanced Clocking Techniques.
5. Hardware-Software Profiling.
6. Moving Boot Devices to Virtual Side.

Results Summary-with Techniques

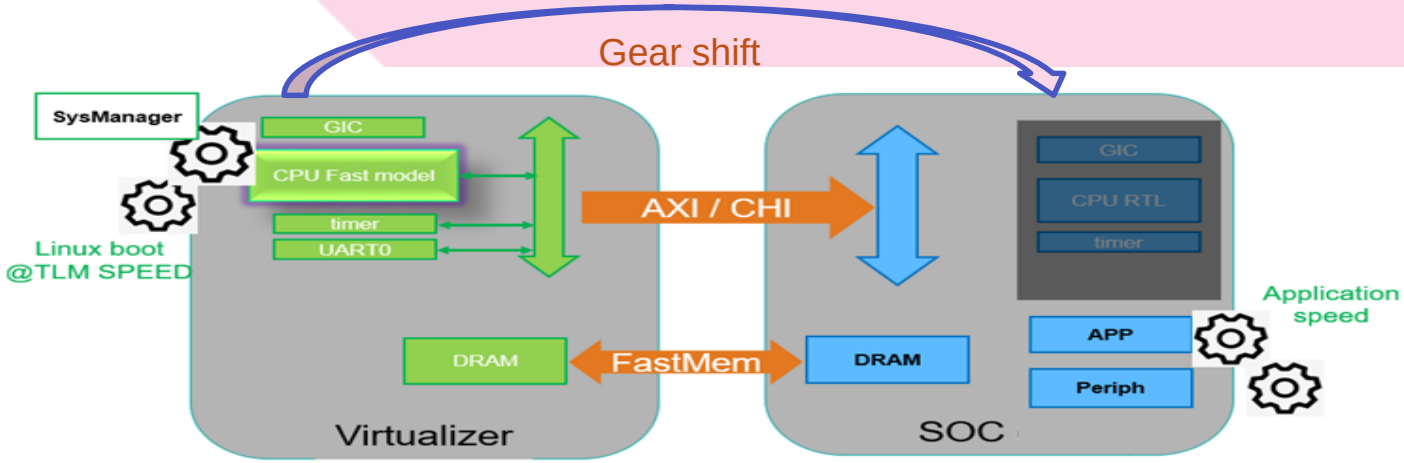
Android Home Screen boot time after applying different techniques



Future Scope

1) Gear-Shift Enablement

1) Once **Linux/Android** boot is done on Virtual CPU , **CPU STATE** will left shift from virtual to **RTL CPU**
2) Now This setup can be utilized to run real World System Benchmarks like for CPU (Geekbench),GPU (Antutu), NPU(MLPerf) etc.
3) App Launch Profiling of Performance and Power



2) ARMonARM Enablement

- Running ARM instructions on ARM native Host Processor instead of x86 to speed up execution

3) Real World Devices

- Enabling real world Virtual devices like sensors to give real inputs to software

Sarang Kalbande

Associate Director ,
Samsung Semiconductor India Research

SAMSUNG