Building Smart Consumer Devices

Preparing for the Challenges

Ramesh N Raghavan,

Chief Architect, Product Engineering Solutions,

Wipro Technogies

Smart Device Characteristics

- Multi-functional
- “name” more based on form factor or promoted functionality
- Built on a fairly powerful computing engine, with rich Operating system, UI layer, and multiple connectivity and I/O Options.
- Field or Auto upgradeable to next generation firmware/software
- Fast power on and switching between roles.
- Rapid commoditization, hence window to monetize the design is very small.

Smart Device Engineering – Silicon to System

Custom UI

Telematic Appls

Multimedia Apps (MP3 Player, PVR, ...)

Mobile Apps (Embd Browser)

Network & Remote Mgmt

Application Program Interface, Industry specific Middleware

DSP Optimization / Performance Tuning

Operating System, Real Time Kernel, Porting to target platform

Board support packages & device drivers

Hardware platforms
VLSI, SOC Design, LCD Ctrl, Peripherals Implementation, Interface Logic, DSP Integration

Layered View of a Smart Device

STB Solutions (ATSC, DVB)

Linux Phone, Hands Free Telephony

MP3, MPEG2/4, H.323, H.264, WMA

IEEE 1394
USB, UART
Wireless LAN,

Re-use Components

System Integration

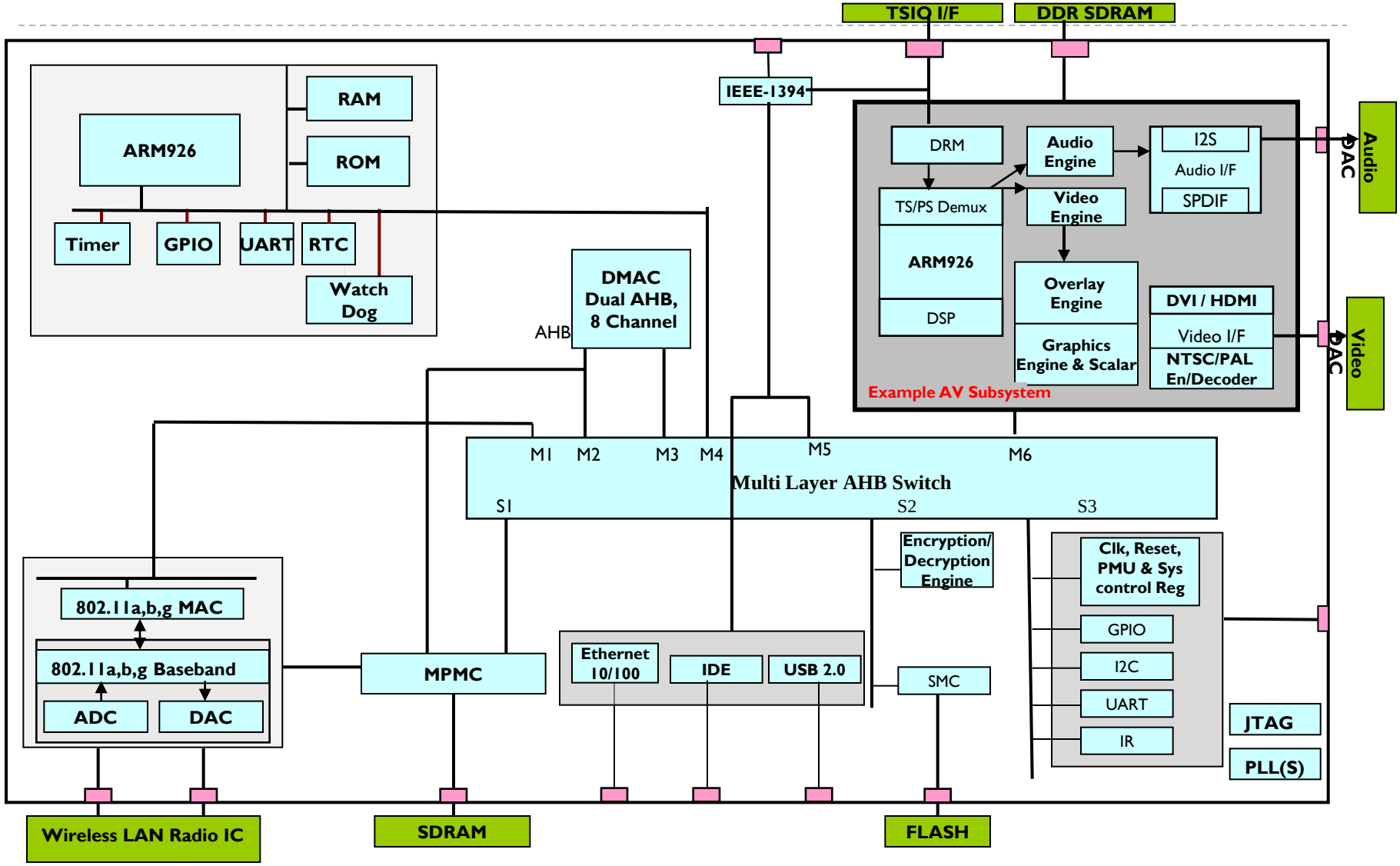
Architecting, Evaluation, Implementation, Testing, Certification

Engineering

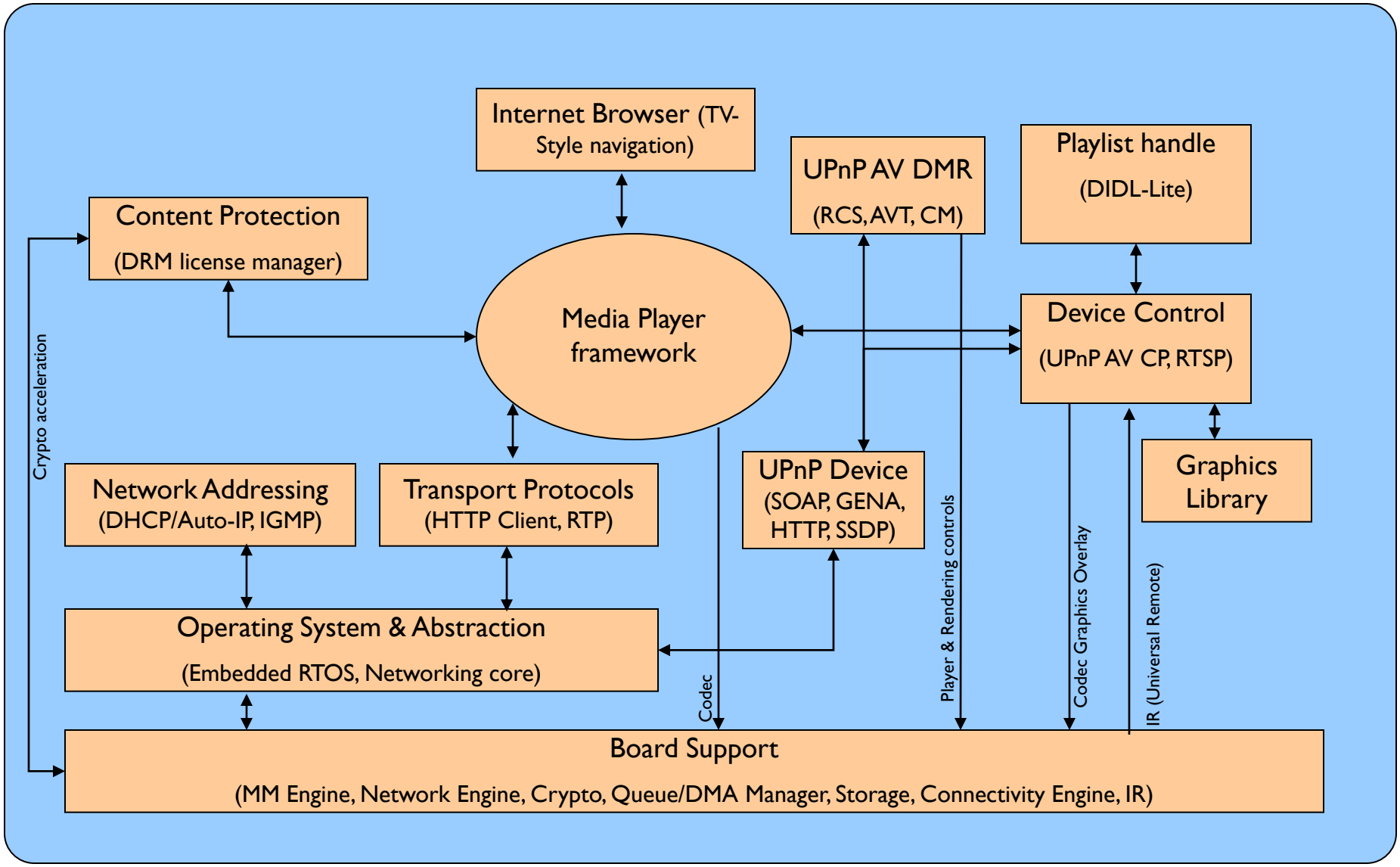
Smart Device Engg Challenges

- Significant increase in HW and SW complexity
  - Product typically built around a complex SoC with at least a couple of processor Cores, DSPs, HW accelerator blocks for Graphics, Video, security, with multi level buses on chip.
  - Runs a powerful OS such as Linux, Android or equivalent providing a lot of flexibility, but comes with added complexity in ensuring correctness, robustness, and predictable performance.
  - Modeling and estimating the performance of the system, identifying bottlenecks, HW-SW partitioning to get the optimal architecture.

Block Diagram for a typical SoC



Sample Software Architecture – Wireless Streaming Device



Smart Device Engg Challenges.. (contd)

- Integration of a large number of 3rd party SW/FW components is the norm for realizing the required functionality.
  - Need to have the skills to quickly understand and tie all the pieces together for realizing the desired functionality.
- Building a sophisticated and refined UI
  - Often the killer feature that differentiates the product in a crowded market place
  - Requires both UI Design expertise as well as engineering skills to realize the same.
- Compliance to a plethora of standards, and certification testing for those
  - BlueTooth, WiFi, Audio standards, Connectivity standards etc.

Smart Devices Engg Challenges..(contd)

- Continuous churn in requirements driven by a very dynamic market environment – Escalating customer expectations, cost and competitive pressure.
- Stability on continuous usage, with infrequent power cycling – Becomes a challenge with the exploding amount of software on the device.
- Low power mode implementation and rapid restore to full functionality, and overall power management for battery operated devices.
- Triaging issues when they crop up – Multiple SW components, HW blocks, drivers, Virtual machines (java) – needs expertise across different blocks and a system level view.

Getting Prepared – as a manager

- Structure teams with specialized expertise areas, but ensure very strong communication, group work among them.
  - Traditional isolated models of VLSI, SW, Board teams need to converge – Need to have system level architects who have sufficient understanding of all the areas and who can co-ordinate across the groups
- Drive performance modeling to the extent possible with the available processor models, device models, virtual platforms and available software base to get a better understanding.
- Organize multiple sessions to understand the end product usage scenarios thoroughly.
  - Identify the interaction between different sub-systems by walking through the use cases – often throws up surprising interdependencies between modules, handling state information etc.
  - With increasing number of SW modules, often coming from several 3rd parties, it becomes crucial to understand what the different modules assume about the usage scenarios.
  - Crucial to get the right domain experts involved in this stage.

Getting prepared – as a manager..(contd)

- Focus on the overall SW architecture and identify portions that can be independently verified thoroughly.
  - Triaging issues becomes difficult when you have doubts about the correctness of several components in the system.
  - Identify test beds for individual modules that can be driven with the use cases identified from a system level perspective.
- Explore the possibility of simulating the entire SW on a platform like a PC, helps in identifying integration issues early.
- Adopt the right engineering model such as Agile which allows for multiple iterations, as it reflects the continuous churn in requirements that is the norm in this domain.
- Strengthen steps like peer reviews for code changes, design changes as often the complexity of the entire system is challenging for a single person.

Getting prepared – as an Engineer

- Scale up your technical skills
  - Strong programming skills and understanding of complex data structures, object oriented design principles is essential to perform well in such programs.
  - Most of these systems use multiple cores, and SW is typically multi-threaded – Need to understand concurrent programming models, race conditions, locking well.
  - Understand how to use the available tools like simulators, debuggers very effectively – Time spent in learning these well will have a huge payoff during project execution – Usage often falls short of potential.
  - As a embedded SW engineer, should gain sufficient understanding of the HW architecture (SoC/Board) to be able to appreciate issues when they arise – Need to develop skills to be able to read data sheets/programmers guide for large complex devices efficiently – Comes with experience and also ability to abstract what is not relevant and focus on the relevant subsystem.
  - As a SoC Designer, or Board designer, appreciation of the issues SW/FW engineers will have while bringing up the system, debugging support needed is crucial to be able to come up with a good design that allows for sufficient tracing and logging as require.
  - Try to draw parallels between different interfaces to be able to understand them faster.

Getting prepared – as an Engineer

- Get an understanding of the end user scenarios for the product you are designing – Helps in coming up with a good architecture and design of the various components.
  - Often the design undergoes multiple changes as newer use cases are discovered late in the cycle.
- Understand how to use tools like Bus analyzers, network/media stream analyzers, logic analyzers, scopes effectively – can be crucial to debug difficult issues.
- Learn to use system profiling tools to understand performance issues.
- Technology is constantly evolving, and so one can never stop learning to be able to perform well.

Q & A