ARM® Cortex®-R Series:

Power, Performance and Area Implementation Analysis.

Authors: Neil Werdmuller and Jatin Mistry, September 2014.

Summary:

Power, Performance and Area (PPA) implementation analysis is often used to compare different processors at a high-level. What a PPA analysis measures is:

1) **Power**: The power that is consumed by the processor. This is broken down into the dynamic power, the power consumed when running usually expressed as Watts/MHz, and the static power (leakage) when the clock is stopped but the silicon is still powered, expressed as Watts. The total power consumed by a running processor is the addition of both at the specified frequency, expressed in Watts.

2) **Performance**: The maximum attainable frequency of the clock driving the processor in this specific implementation. This frequency is often multiplied with an industry benchmark score such as Dhrystone MIPS or EEMBC CoreMark to provide a figure such as total DMIPS.

3) **Area**: How much silicon area the processor occupies in mm^2 (and typically the level 1 (L1) memories such as the cache) takes up. This can also be expressed as a gate count, though today gate count is approximated from the actual silicon area of the implementation.

However, there are ***many*** aspects of these analyses that mean comparing 'apples-to-apples' is very difficult. The goal of this document is to explain, for those without really deep processor implementation knowledge, the many variables that should be understood to get real value from any PPA data presented to enable an estimation of the real PPA of your own proposed processor implementation and also to make fair comparisons between processors, both from a single IP partner or between processors from different processor IP vendors.

1. What is a PPA analysis and how is it created?

To develop a PPA analysis, a complete trial physical implementation is required for a given processor in order to obtain a representative physical layout – A processor is used as an example throughout this document but the same applies to other ARM IP. A processor is initially defined in Register Transfer Level (RTL), typically Verilog or VHDL. The processor is firstly configured to a particular specification - this involves choosing what size the caches will be and which optional processor features will be included. Following this, a set of industry standard Electronic Design Automation (EDA) tools are then used to put the configured processor through a full synthesis and place and route flow. The synthesis step maps the high level RTL definition of the processor to a set of logical gates, e.g. NAND and NOR, from a chosen physical IP library such as ARM's standard cell library and memories. This step generates what is referred to as a 'netlist'. The place and route step takes the gates that form this netlist and then ensures that firstly they are correctly placed in a given core boundary, secondly the clock reaches all the registers in the design in a balanced manner and lastly adds the required routing between all of the blocks whilst adhering to any design rules.

There are three key inputs to the physical implementation: the first two are the RTL and choice of physical IP library as already described, the third is a set of constraints. These constraints include design rules that must be met such as the maximum allowable time taken for a low to high or high to low logic signal transition and maximum output load capacitance of a logic gate. However, the constraints also define the desired target clock frequency. At each step of the implementation flow a set of optimization techniques are used within the EDA tools in an attempt to achieve timing closure at the desired frequency whilst meeting the design rules to ensure the processor can run reliably.

To ensure the implemented design is representative of a real physical implementation, all PPA analyses are configured to include scan chains for Design for Test (DFT), include clock gating, use representative power grids and are fixed for hold timing against the semiconductor fabrication plant ("fab") recommended margins on all timing paths.

The output of the complete implementation flow is a physical layout of transistors, in the form of logical gates, and metal routing that could then be made into real silicon. This provides the total area of the silicon required for the given physical implementation as well as how much of that total area is occupied by just logical gates, or standard cells as they're more commonly known as, and memories, thus providing the ***Area*** of the PPA analysis.

To obtain the ***Performance*** of the final physical implementation, firstly an industry standard parasitic extraction tool is used to obtain all resistances and capacitances associated with the metal layers of the design. Typically the worst case extraction is used to obtain the most pessimistic analysis and the extraction performed will also consider coupling capacitance so that crosstalk effects - the effect between two simultaneously switching, adjacently placed signal routes - can also be considered. Secondly, the extracted parasitics and final netlist are passed to an industry standard Static Timing Analysis (STA) tool. The STA tool takes into account worst case process, voltage and temperature (PVT) variations and uses the extracted coupling capacitance values to check and propagate crosstalk effects which can degrade timing.

To obtain the ***Power*** of the PPA analysis, the execution of a benchmark such as Dhrystone is simulated on the final place and routed netlist. Real timing annotations provided by the Static Timing Analysis (STA) tool are used as part of the simulation and the switching activity inside the design during the simulation is captured. The switching activity can then be annotated back onto the design in an industry standard power analysis tool, along with a set of extracted parasitics to accurately model the power required to drive the signal routing.

Each run of these physical implementations and analyses, even on powerful server farms, takes hours or, for more complex designs, days to complete. A single detailed PPA analysis that is optimised for a particular target can take many weeks from starting the physical IP library selection to a final optimised analysis.

In the following sections, greater detail will be given on some of the aspects that make up and define ARM PPA results to aid understanding. Section 2 describes the ARM PPA results; Sections 3 explains the different configurations used for the Cortex-R series PPA analyses; Sections 4 and 5 explain the various choices that are available for standard cell libraries and memories; sections 6 – 11 provide deeper understanding of pre- and post- shrink area, standard cell utilization, how a gate count is

arrived at, PVT corners, margins and On Chip Variation (OCV) and the DMIPS performance. Finally, a conclusion is given in Section 12.

2. ARM PPA Results

The conclusion of a specific ARM Cortex-R PPA analysis results in the following data obtained from a single physical implementation to show exactly what power and area should be expected for a particular configuration and target frequency.

Typically these results include:

- A description of the PPA analysis' target, as examples: Speed Optimised, Power Optimised or a Typical implementation
- A description of the IPs configuration
- The maximum frequency for worst case silicon without OCV derate but including timing margins
- The maximum frequency for worst case silicon with OCV derate and timing margins
- The total die area post optical shrink.
- The process conditions.
- The standard cell libraries used.
- The memory libraries used.

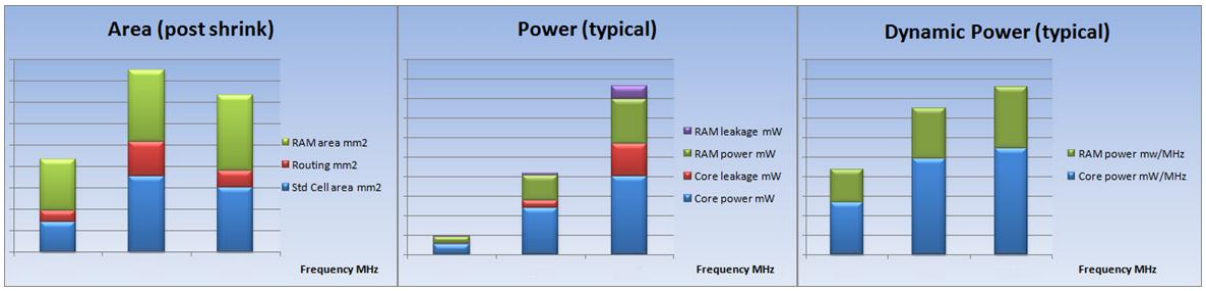
Configuration and Performance Section:

- Detail of the processor version and configuration.
- Frequency achieved.
- Macro dimensions and area breakdown of the processor core, memories plus utilization
- Power, both static and dynamic, of the processor core and the memories.

Floorplan Data Section:

- A diagram of the floorplan highlighting the main areas of the layout.
- Details of the metal layers used.
- The routing rules.

The details can be summarised into graphs that show the main outputs for the implementations analysed. These are: The area of the configurations, broken down into standard cell area, routing area and the area of the memories, total power broken down into core power and leakage and memory power and leakage, and dynamic power broken down into core and memory power.



3. Multiple PPA Analyses and Processor Configuration

The first decision to make when creating a PPA analysis, and the first information detailed in the results, is regarding the analyses target and processor configuration. The information that ARM silicon partners want to learn from a processor PPA analysis on a specific process is: How ***fast*** can the processor be run, how ***small*** and low power can it be configured and, most importantly, what are the results for a ***typical*** real-world implementation.

The silicon partner can then make estimations, based on the PPA requirements of their target implementation, where their implementation needs to fit within these bounds.

At ARM, for the Cortex-R Series processors, three main implementations are typically created:

- Minimum (MIN) – minimum usable processor targeting the lowest possible power and area.
- As Fast as Possible (AFAP) – targeting a processor with the highest possible clock speed.
- Featured (FEATURED) – a typical processor configuration and target frequency, which may not be the maximum attainable frequency in that configuration.

If the processor can also be used in other configurations, such as in multi-processor configurations, then these PPA analyses may also be developed.

An essential part of these multiple PPA analyses is the processor configuration. ARM processors enable many different configurations for different applications. This can include for example:

- The sizes of caches and other level 1 memories such as Tightly Coupled Memories (TCMs) if included inside the processor.
- How many master/slave ports are enabled for the processor. For example whether the Low Latency Peripheral Port (LLPP) or the Accelerator Coherency Port (ACP) is included.
- Is there a Floating Point Unit (FPU), and if so is it single precision or single and double precision?
- What error detection/correction (parity, ECC) and logging features are enabled?
- What debug features are enabled, how many watch and break points are allowed?
- How many Memory Protection Unit (MPU) regions are supported?
- How large are the branch prediction global history buffers and how big is the branch target address cache?

In multiprocessor systems the configuration is even more complex possibly with a Snoop Control Unit or with additional logic to support a lock-step (redundant core) implementation.

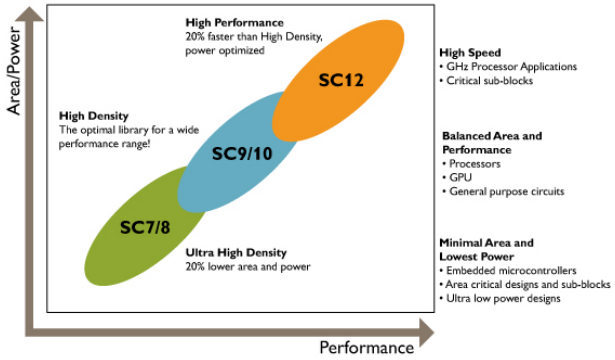
4. Process and Library Choices

The choice of process and library is a key input to the physical implementation and defines the transistors and logic gates that are used to implement the RTL design, which have a major impact on the final PPA results. The first decision is the silicon process that is being used, for example TSMC 28nm HPM or 40nm LP. This is specific to a silicon fabrication process and differs between silicon manufactures such as TSMC, UMC, Global Foundries etc.

Once the process has been selected then there is a choice of libraries that can be used to implement the design in silicon. ARM processors and other ARM IP use production quality ARM Standard Cell Libraries for PPA analyses. On top of the standard 'base' ARM standard cell library, there is a choice of a low power kit (LPK)/power management kit (PMK) for implementing low power techniques such as power gating and dynamic voltage scaling. Furthermore, at some process nodes ARM also offers a choice of a high performance kit (HPK), which employs advantageous circuit tuning practices inside the logic gates to achieve higher performance with minimal impact on area and power. This makes the HPK ideal for 'Featured' or 'AFAP' implementations.

Once the library provider has been chosen, there is then a large choice of specific library characteristics to cover a wide performance range for all types of designs. To begin with, the height of the standard cells in a logic gate can be varied to fit a number of different targets - Taller gates allow larger transistor widths to be used, equating to higher performance. The height of a standard cell library is expressed as a number of tracks, where 1 track is the minimum allowable spacing between the Metal 1 routes in a particular technology node. ARM standard cell libraries can be split into three main categories:

- Ultra high density/low power; 7 or 8-track (SC7/SC8) libraries for cost critical applications
- High density; 9 or 10-track (SC9/SC10) libraries for mainstream applications
- High performance; 12-track (SC12) libraries for speed critical designs



Following the choice of standard cell height is the option of which transistor threshold voltage, Vt, to use. The threshold voltage has an impact on the switching speed of the transistors inside a logic gate and therefore has a direct impact on performance. The threshold voltage does, however, also have a direct impact on static power (leakage). High threshold voltage (HVt) logic gates are slower but exhibit low static power whereas low threshold voltage (LVt) logic gates are fast but have high static power.

Finally, at 40nm and below, a choice of channel length is also available for ARM standard cells. The channel length choice can be treated in a similar way to the threshold voltage of the transistors.

Short channel lengths, e.g. C31 in TSMC 28HPM, provide the fastest transistor but at the cost of static power. Long channel lengths, e.g. C38 in TSMC 28HPM, are slower but have lower static power.

All Multi-threshold voltage and Multi-channel length standard cells in the ARM standard cell libraries are footprint compatible i.e. a NAND2 LVt C31 occupies the exact same area as a NAND2 HVt C38. This means that a fully Multi-Vt, Multi-C physical implementation can be employed to achieve best performance and power for a particular target configuration.

5. Memory Sizes and Types

Production quality RAMs with real timing are used during PPA analyses and are not instantiated as black boxes with ideal timing. For this reason, processors that include memory as part of the processor, for example the caches in the Cortex-R Series and the internal TCMs of the Cortex-R7, the size and type of memory that is included can have a major impact on the PPA analysis.

For caches, the size of the memories alone has a major impact on area and power consumption. Doubling the memory size can more than double the area and power so two 32KB caches will have significantly higher area and power than using two 16KB caches. However, 32KB is a more realistic configuration for a typical processor and is why all Cortex-R PPA analyses use 32KB caches.

Note also that how the memories are instantiated can also have a major impact. For the Cortex-R Series PPA analysis in 28nm HPM three main memory types were used:

- ARM Artisan RF High-Density SP Memory Compiler (lowest area and power)
- ARM Artisan RF High-Speed SP Memory Compiler (higher speed but higher area/power)
- ARM Artisan Fast Cache Instances (memories optimised for a processor)

The speed of the memories is often critical to the timing closure of the processor and has a major impact on the frequency that the processor can be clocked. However, faster memories take up more silicon area and consume more power. In physically implementing the processor, the smallest memories are used to achieve the speed and performance required by the design.

Typically for the Cortex-R Series the High-Density compiler was used for the MIN PPA analysis, the High-Speed compiler for the FEATURED PPA analysis, and Fast Cache Instances for the AFAP PPA analysis. As an example from the Cortex-R5 PPA analyses in 28nm HPM:

- 2x32KB caches instantiated with High-Density compiler: 0.118mm^2 (post shrink)
- 2x32KB caches instantiated with High-Speed compiler: 0.167mm^2 (post shrink)
- 2x32KB caches instantiated with Fast Cache Instances: 0.175mm^2 (post shrink)

The choice of memory also impacts the power consumption (dynamic and static) of the memory too:

- 2x32KB caches instantiated with High-Density compiler: 0.0171mW/MHz and 0.788mW
- 2x32KB caches instantiated with High-Speed compiler: 0.0269mW/MHz and 2.220mW
- 2x32KB caches instantiated with Fast Cache Instances: 0.0311mW/MHz and 13.70mW

6. What is pre and post-shrink area?

Post-shrink is often used in ARM PPA results but a common question is what does it mean and why is that? The chip manufacturing industry follows the silicon process roadmap set out by the

International Technology Roadmap for Semiconductors (ITRS - http://www.itrs.net/). About every two years the industry moves forward to the next manufacturing node, e.g. 90nm in 2004, 65nm in 2006, 45nm in 2008 etc.

Between the full nodes defined by the ITRS, advances in lithography techniques – the process used to manufacture the silicon - allow foundries to develop what are referred to as half nodes. The foundries are simplistically able to use the improved manufacturing process to allow the exact same circuit from a full node to occupy less silicon real estate. For this reason, the technique is often referred to as 'optical shrink'. Examples of full nodes defined by the ITRS are 65nm, 45nm and 32nm whilst examples of their equivalent half nodes are 55nm, 40nm and 28nm respectively.

As the shrink is achieved by optical methods, the circuit is still drawn by the EDA tools at the dimensions of the preceding full node when performing the physical layout. This means all lengths and areas in the physical layout are considered to be 'pre-shrink' and to calculate the area the circuit will truly take up on silicon, we must account for the optical shrink. Using the 28nm half-node as an example, the shrink from the preceding 32nm full node is 0.9. Consequently, all lengths in the physical layout are multiplied by 0.9 and all areas are multiplied by 0.81 (0.9 x 0.9). It must be noted that the shrink factor for a half-node is not always 0.9, and is decided upon by the fab.

7. Standard Cell Utilization

The ***Area*** in ARM PPA results gives the amount of silicon area - the die area - that would be required if the design were to be manufactured. However, not all of this area is occupied with gates, or standard cells as they're more commonly known. Some of the area is taken up by RAMs and placement blockage and so this area is intentionally unused for placing standard cells. The rest of the area could be used but some empty space will remain. The std. cell utilization quoted in ARM PPA results therefore refers to the percentage of the total die area that can be used for standard cell placement, which is actually occupied by standard cells. This can be calculated by the following:

$$Utilization = \frac{Std.\,Cell\ Area}{Total\ Die\ Area - Macro\ \&\ Blockage\ Area}$$

8. Calculating gate count from silicon area

A useful metric to understand the complexity of a particular design is the equivalent NAND2 gate count. This is because logic gates can take the form of very complex functions and so simply providing a true logical gate count can be deceiving. More complex logic gates tend to be larger in size and therefore yield higher numbers when converted to a number of NAND2 equivalent. This statistic is not shown in the ARM PPA results but is often included in combined comparison data, and is typically shown in the Cortex-R Series PPA summary PDF. To obtain the gate count, the following calculation is performed:

$$Gate\ count = \frac{Std.\,Cell\ Area}{Area\ of\ a\ standard\ X1\ drive\ NAND2}$$

The NAND2 equivalent gate count is also somewhat historic since it can be used to determine whether a particular processor could fit onto a programmable gate array.

Note it is important to check the size of the NAND2 gate used for the calculation. ARM uses standard X1 drive NAND2 but if a larger size of a high power NAND2 were used this would indicate a lower gate count.

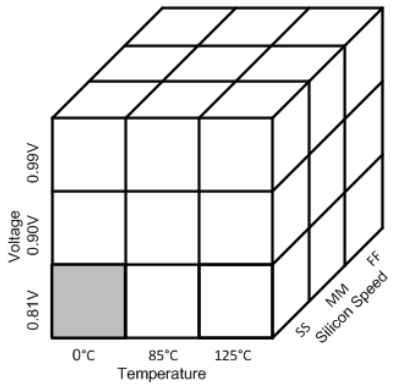
For example, on a 28nm HPM process with 9 track libraries a NAND2X1 is 0.486µm^2 and with 12 track libraries 0.648µm^2 and on 40nm LP process with 9 track libraries a NAND2X1 is 0.7182µm^2 and with 12 track libraries 0.9576µm^2 (all pre-shrink values). Similarly, comparisons of NAND2 equivalent gate counts should only be made between results from the same process and library. For example, it is not suggested to compare gate counts from a 40LP implementation with a 28HPM implementation due to difference in size of the NAND2 gates and the difference in gate variety.

9. Process, Voltage and Temperature Corners

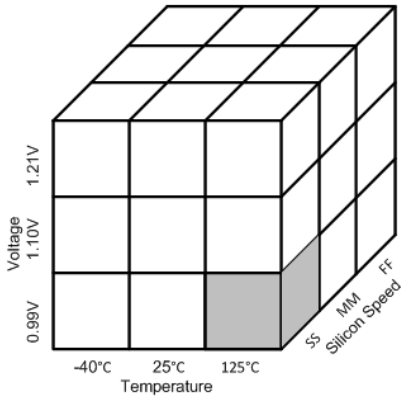
Process and Silicon Speed: In integrated circuit semiconductor fabrication, a process corner represents variation from nominal doping concentrations in transistors on a silicon wafer. This variation can cause significant changes in the speed at which digital signals transition from high to low and low to high. Silicon speed is described by two-letter designators, e,g, ss, tt or ff, where the first letter refers to the N-channel MOSFET (NMOS) corner, and the second letter refers to the P channel (PMOS) corner: **s**low**, t**ypical, and **f**ast. Fast and slow corners exhibit carrier mobility (both electron and hole mobility) speeds that are higher or lower than normal. Worst case silicon speed is therefore slow-slow.

Voltage: The device is expected to operate over an acceptable variation in voltage range. Transistors operate faster at higher voltages and slower at lower voltages. Therefore, the worst case 'slow' corner is assumed to be -10% from the typical voltage and the best is +10%. For example, if typical voltage is 0.9V then the best case, highest voltage is 0.99V and the worst case, lowest voltage is 0.81V.

Temperature: A device is designed to operate over a temperature range – typically -40°C to 125°C (though this can vary for some applications). At 40nm and larger geometries, the worst case temperature is 125C. At smaller process geometries this switches to be at 0C.



Worst Case Corner in 28nm HPM Process



Worst Case Corner in 40nm LP Process

For the Cortex-R Series PPA analyses, the frequency is always calculated for the worse case corner. That is under voltage in slow-slow silicon at 0°C in a 28nm process. This ensures that the target frequency can be reached in the worst case operating conditions. For the power calculations, typical conditions are assumed – at 28nm this is 85°C, typical voltage and typical-typical silicon to provide a typical power figure. It is important to confirm which corner was used for both the frequency and power calculations.

10. Margins and OCV derate

PPA analyses of ARM IP use fab recommended margins and On Chip Variation (OCV) derates. The setup and hold margins recommended by the fab are in place as a safety margin against potential failure in the manufactured silicon. All physical implementations done by ARM therefore fix to these margins for all timing paths in the design. It should be noted that the setup margin quoted in the ARM PPA results account for both the fab recommended setup margin and clock period jitter margin due to PLLs being imperfect. Similarly, for ARM IP such as the Cortex-R7 where negative edge triggered logic is present, a margin is also applied to account for clock duty-cycle jitter as the duty-cycle of a clock generated by a PLL is not always exactly 50/50.

OCV accounts for the effect of timing variation in the silicon that arises as part of the manufacturing process. As we move to smaller geometries, the significance of on chip variation becomes greater and greater. Simplistically, OCV assumes that non-common paths in the physical design can exhibit different behaviour in terms of speed. Therefore, an OCV derate is applied to the logic to either speed them up or slow them down during the timing analysis to consider the worst case timing between two points. This is most commonly used on the logic gates that feed the clock signal to the register from which the data is transmitting from and the register that the data signal is transmitting to.

The use of the fab's recommended settings on all timing paths makes ARM generated physical implementations representative of implementations done for a real SoC. However, their inclusion can have a significant impact on area and performance. At 28nm for example, up to 10% of the total area (more if you consider only the standard cell area) can be due to hold fixing on functional and scan paths due to the large hold margin and high OCV derates applied to timing paths.

11. Calculating Performance - DMIPS/MHz

When comparing Dhrystone DMIPS figures it is vital to understand whether the DMIPS value is a 'legal' DMIPS value or whether additional optimisations have been performed. The 'legal' DMIPS procedure prohibits the use of inlining and multifile compiler optimisations. Because these non-legal figures are sometimes quoted, ARM now provides the results for all three to enable fair comparisons, for example the Dhrystone DMIPS results are shown below:

DMIPS/MHz	Cortex-R4	Cortex-R5	Cortex-R7
Dhrystone	1.68	1.67	2.50
Dhrystone with inlining	2.03	2.02	2.90
Dhrystone with inlining and multifile	2.45	2.45	3.77

Dhrystone results all generated with ARM RVCT 4.0-728 compiler

The compiler that was used to generate the DMIPS data is also important. ARM typically uses the ARM RVCT compiler as it is an industry standard compiler that our partners can also use for their code compilation (and not a 'benchmark special' compiler).

12. Conclusion

PPA data is very complex. There are many configuration options for a processor itself and understanding the configuration that the PPA analysis supports is essential. It is possible to create a stripped down processor configuration that provides an attention grabbing headline area and power PPA analysis which could/would never be used in a real implementation. It is also possible to

configure a processor to be clocked very fast but, for example, without L1 memories it would have very poor performance having to fetch all data and instructions from slower main memory.

It is important to understand all of the variables to make any sensible PPA comparison between processors. The ARM Cortex-R Series PPA analyses use a representative physical implementation flow to make them as comparable to real final implementations as possible. Furthermore, the ARM PPA results provide complete details of the processor configuration, the libraries used and also the memory that is attached. It also details which corner was used, the OCV derate and margins considered.

It is often the case that ARM silicon partners, in their specific implementation, with their libraries and their optimisations can better the results that ARM PPA data indicates. PPA is useful data to feed into processor selection for a specific application. However, the PPA is just one factor in this selection process and in most cases the actual features and time-to-market considerations are much more critical.

When comparing PPA across processors always look beyond the headline numbers and check you are not comparing ‘apples-to-orangutans’!

