



PPA analysis overview

Version 1.0

Non-Confidential

Copyright © 2019 Arm Limited (or its affiliates).
All rights reserved.

Issue 01

102738_0100_01_en



PPA analysis overview

Copyright © 2019 Arm Limited (or its affiliates). All rights reserved.

Release information

Document history

| Issue | Date | Confidentiality | Change |
|---------|----------------|------------------|---------------|
| 0100-01 | 1 January 2019 | Non-Confidential | First release |

Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly

or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at <https://www.arm.com/company/policies/trademarks>.

Copyright © 2019 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349|version 21.0)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Feedback

Arm® welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on <https://support.developer.arm.com>

To provide feedback on the document, fill the following survey: <https://developer.arm.com/documentation-feedback-survey>.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Contents

| | |
|---|----|
| 1. Overview..... | 6 |
| 2. Power, performance, and area analysis..... | 7 |
| 3. Choosing the IP configuration..... | 10 |
| 4. Choosing the fab process..... | 15 |
| 5. Choosing the physical IP libraries..... | 18 |
| 6. Further choices that affect PPA implementation analysis..... | 24 |
| 7. How can higher performance increase the area of implemented IP?..... | 28 |
| 8. Performing PPA analysis for a system..... | 33 |

1. Overview

The intended audience for this material is system designers, of any level, or anyone else with an interest in gaining insight into how to choose individual pieces of IP and combine them into a custom SoC. Power, Performance, and Area (PPA) analysis collects figures relevant to each of the three categories. Along with cost, there is usually a tradeoff to be made between the three points, and when the tradeoff is made successfully, the custom SoC is much more likely to be fit for purpose. Interpreting the data, taking into consideration the context under which the data was obtained, and being aware of the limitations of comparing multiple sets of data are critical skills in this domain. This material aims to tutor you in those skills.

If your organization has signed up for the Arm Flexible Access program, then this material will enable you to interpret PPA analysis data for all the IP available. For organizations that are not members, it can still provide valuable tutoring on working with PPA data and, depending on the IP you are interested in, take you further towards deciding on which IP to license for your project.

When PPA analysis is performed, the analysis team must make many decisions, which are common to the decisions that are made when an SoC is manufactured. In this sense, the material has a secondary function. If you are interested in the steps taken to move IP selected for an SoC into a fully unified and realized piece of technology, this material can also provide valuable information.

2. Power, performance, and area analysis

Power, Performance, and Area (PPA) analysis is performed on an implementation of a piece of IP. At the point that the analysis is performed, the IP does not exist as a physical product. The purpose of the analysis is to gain a solid idea of how the implementation would be if it was realized. The importance of PPA analysis lies in this fact. PPA analysis gives interested parties the chance to evaluate implementations of a piece of IP before time and money is invested in producing the IP.

Before looking at any specific IP, let's look at what power, performance, and area mean in the context of a PPA analysis.

Power

With regard to power, the values of interest are measured in [watts](#). The bottom line is that a piece of IP could be very energy efficient, but the IP may still use too much power to be a viable choice for your project. There are two power readings of interest:

- Dynamic power
- Static power

Dynamic power

Dynamic power refers to the power that is consumed when the IP clock is running. In the PPA data provided for the Arm Flexible Access program, dynamic power is always expressed as mW/GHz to aid comparisons. With processors, you can assume that:

- A benchmark, for example [Dhrystone](#), was running when the measurements were taken.
- The processor was running at the maximum cycles possible for its clock setting.

Static power

Static power or leakage is the power that the IP uses when the clock is stopped but the IP is still powered. In the Arm Flexible Access program PPA data, static power is always expressed in mW to aid comparisons.

Performance

Performance refers to the maximum clock frequency that the IP can obtain, in a specific implementation. Performance is measured in MHz or, for more powerful processors, GHz. The value is also known as the target frequency in an implementation. Achieving higher levels of performance increases the area and power usage of a processor. This concept is explored further in [Exploring how higher performance can increase the area of the implemented IP](#).



In this overview, we use the term frequency to describe the performance relating to the clock frequency of a piece of IP. The term performance is used to describe other things, for example the data throughput in a cache.

Benchmarking

Independent benchmarks are also used as an indicator of performance.



Unlike PPA data, benchmark data is independent of implementation, and the figure can be calculated from the most abstract representation of a processor design. Benchmark data is useful for comparing IP without considering implementation details. Depending on the implementations, an earlier processor design can be, when realized, faster than a realized later design. The benchmarks, however, demonstrate that the later design is conceptually more powerful. This means that benchmarks give a different perspective to PPA frequency data.

For processors, three kinds of benchmark are used:

- [Dhrystone](#)
- [Coremark](#)
- [Spec2k](#)

Certain benchmarks are preferable for certain categories of processor. The Arm Flexible Access program PPA data is sensitive to this. If you are likely to compare two processors, you can be sure that the same benchmark was used for both sets of figures. For example, the Spec2k benchmark is used for A-series processors.

When running the benchmarks, Arm uses the industry standard Arm compilers, which our partners can also opt to use. These compilers are not special benchmark compilers designed to give favorable results. If you are comparing any Arm PPA data with data provided for another vendor's processors, check the compiler that is used to compile the benchmark code.

Benchmarking data example The following table looks at Dhrystone and Coremark values for the Arm Cortex-M processors that are available with the Arm Flexible Access program.

| Processor | Dhrystone (DMIPS/MHz) | CoreMarks/MHz |
|----------------|-----------------------|---------------|
| Arm Cortex-M0 | 0.89 | 2.33 |
| Arm Cortex-M0+ | 0.95 | 2.46 |
| Arm Cortex-M23 | 0.99 | 2.5 |
| Arm Cortex-M3 | 1.25 | 3.34 |
| Arm Cortex-M4 | 1.27 | 3.42 |
| Arm Cortex-M33 | 1.50 | 4.02 |
| Arm Cortex-M7 | 2.14 | 5.01 |

With both benchmarks, higher values mean higher performance. The data shows, by removing the implementation from the equation, that low-cost processors, for example the Arm Cortex-M0 or Arm Cortex-M23, are not as powerful as mainstream, rich processors for example the Arm Cortex-M33 or Arm Cortex-M7. As this material explores, the physical implementation of a piece of IP involves many factors. This means that IP with lower benchmarks can, depending on the implementation, achieve a higher target frequency than IP with a higher benchmark score.

For example, an Arm Cortex-M4 implementation can achieve the same target frequency as an Arm Cortex-M7 implementation. This is still the case when both processor implementations are set up to have similar capabilities, for example Floating Point Unit (FPU), Digital Signal Processor (DSP) extension, and no cache. However, the Arm Cortex-M4 implementation will have a much higher static power usage, which reflects the fact that an inherently less powerful design is being pushed. Pushing a piece of IP during implementation often makes meeting the required tradeoffs difficult. Better solutions can often be found by going back to the benchmarks, selecting a different piece of IP, and beginning to research what the IP can offer using PPA analysis data.

Area

With regard to size, the value of interest is measured in mm² and refers to the total area of silicon that is required to make a physical implementation of the IP. For processors, this includes the [logic gates](#), which are also called [standard cells](#), and the memories, for example the L1 caches.

Sometimes area readings are given as gate count figures. However, these figures are potentially misleading, and they can vary by a large margin depending on the fab process that is used and the maximum achievable frequency of the implementation. For this reason, this Arm Flexible Access program PPA data presents all area figures in mm².

3. Choosing the IP configuration

How a piece of IP is configured can have a big effect on a PPA analysis. In particular, the area of a realized piece of IP is affected by how many optional extra features are included. Potentially, the choice of options can have a bigger effect than channel length and track size. This means that becoming familiar with the configuration options for the different IP available, via the Arm Flexible Access program, and deciding whether an option is needed, is an important step. Ultimately, if an SoC design requires certain features, then the increase in size must be accepted. In this case, other ways of reducing the size, for example through physical IP library choice or using a more modern process, could be explored.

Running a PPA analysis requires that the analysis team choose how to configure the IP. If they decide to run an analysis multiple times, they may remove optional features during runs so that they can target, for example, minimal area or high performance.

The following sections look at the configuration options for processors as a starting point. Not all options are available for all processors. In addition, some options may be obligatory for some processors.

Caches

When looking at a PPA analysis, look at the size of the caches used in the trial implementation. If you are concerned about the area of the IP, you could reduce the size of the caches. Because smaller caches impact throughput performance, a tradeoff exists here. Also, be aware that larger caches increase performance up to a sweet spot beyond which further increases in cache size will only lead to marginal performance improvements.

Level 1 caches

On Arm Cortex-A series and Arm Cortex-R series processors, it is typical to have a level 1 cache for instructions, called an I-cache, and a level 1 cache for data, called a D-cache, for each core.



Note

It is possible to have a processor without any level 1 memories, and most Arm Cortex-M series processors do not have caches. Without a cache, the processor must fetch all data and instructions from the possibly slower main memory. If the main memory is slower, the system will be significantly slower even if the processor is clocked very fast.

Level 2 caches

Level 2 caches are only available for Arm Cortex-A series processors. All Arm Cortex-A series processors within the Arm Flexible Access program, except for the Arm Cortex-A5 processor, support an internal level 2 cache, which is shared between the cores on the processor. Be aware that the area for level 2 caches for Arm Cortex-A5 processors will still contribute to the overall area of the SoC. The area figure for Arm Cortex-A5 multi-core processor PPA analysis will include a level 2 cache controller (L2CC).



Some more advanced Arm architectures allow for each core in a multi-core processor to have a level 2 cache. However, these processors are not currently available within the Arm Flexible Access program.

Tightly Coupled Memory

Tightly Coupled Memory (TCM) ensures that critical code and data is always available. Compared with caches, which can introduce indeterminate delays, TCM supports the processing of data with single cycle access. Where applicable, data blocks can be moved into the TCM as a background task that is processed while within the TCM, and then written back to the main memory. One or more TCMs can increase performance where it is most needed, and these memories are very suitable for deterministic interrupt routine responses.

TCMs are an option for Arm Cortex-R series processors and the Arm Cortex-M7 processor. It is very common to have more than one TCM per core. For example, like level 1 caches, you can have a TCM A for instructions and a TCM B for data. TCMs can vary in size. Larger TCMs will have a bigger impact on the area of your realized IP.



When you look at PPA analysis data for the Arm Cortex-R5 and Arm Cortex-M7 processors, you will notice that only the interfaces are included. This is because the actual TCMs must be situated on a different block in the SoC than the core. This means that the PPA analysis data is only showing the area for the interface, which is negligible, and you must consider that TCMs will add to the final area of the SoC.



The Arm Cortex-R5 processor provides two interfaces for TCM B: B0 and B1. If you are using an [Advanced eXtensible Interface \(AXI\)](#) slave interface, you can make use of the second interface to TCM B. Although the increase in area from having two B interfaces is trivial, an increase in performance could be achieved by this configuration.

Fault protection

Two mechanisms can be employed to protect against faults occurring in data: Error Correction Code (ECC) and parity. Parity is cheaper in terms of area than ECC because only a single bit is used. In an SoC, both mechanisms can be employed, with ECC and parity protecting different data.

Although fault protection requires a larger memory footprint, because ECC and parity are built into the pipeline, there is a very minimal impact on performance.

Error Correction Code

Error Correction Code (EEC) is an optional feature that detects errors and, in the case of single-bit errors, automatically corrects them. This ensures data integrity in caches and TCMs. ECC achieves its function by storing code which describes the data present in memory. The code itself is stored in extra bits. For example, every 32 bits of memory might have 8 bits allocated for the code. When

the memory is written to, the codes are automatically created. When the memory is read from, any errors are detected and, where possible, corrected.

On Arm Cortex-R series processors, it is possible to have ECC on buses. When this configuration is used, it is noted in the PPA analysis data. This extra functionality does not affect the area of the processor IP beyond noise.

Parity

Parity is an optional feature that detects errors, and which involves the use of a parity bit. A parity bit is used to protect the data in the tag RAM of a cache controller. The tag RAM contains the address information for a cache.

Floating Point Unit

A Floating Point Unit (FPU) enables floating point calculations to be made on a processor. An FPU also significantly increases the area of a processor. FPUs can be single precision or double precision. Double precision FPUs use more area than single precision FPUs. Where the FPU is an option, for example for all Arm Cortex-R-series processors, all Arm Cortex-A-series processors, and the Arm Cortex-M23, Arm Cortex-M4, Arm Cortex-M33, and Arm Cortex-M7 processors, your decision regarding an FPU will ultimately be governed by the requirements of the software that you need to run on your SoC.

The Neon instruction set is reliant on an FPU being included in the processor design, and on some Arm Cortex-A-series processors, the FPU and Neon are coupled together as a single option.

Embedded Trace Macrocell

The Embedded Trace Macrocell (ETM) is a debug component that enables reconstruction of program execution. The ETM is designed to be a high-speed, low-power debug tool, which only supports instruction tracing. This ensures that the area that the ETM adds to any realized IP is minimal. The ETM is optional for the Arm Cortex-M23, Arm Cortex-M3, Arm Cortex-M4, Arm Cortex-M33, Arm Cortex-M7, Arm Cortex-R5, Arm Cortex-R8, and Arm Cortex-A5 processors. If you anticipate that processor tracing would have no use in your design, you could reduce the area of your design by excluding it. However, excluding the ETM may have a serious impact on the team developing software for your SoC, and might affect safety certification processes. We recommend that your software team is included in any discussion about the inclusion of an ETM.



The ETM option for the Arm Cortex-R5 processor is a separate component outside of the processor block. Although the ETM option cannot affect the PPA analysis of the Arm Cortex-R5 processor, if the option is included, then the ETM component will still add to the area of the SoC.

Memory Protection Unit

The Memory Protection Unit (MPU) is a hardware unit that controls a limited number of protection regions in memory. MPUs are a major component of the Arm Protected Memory System Architecture (PMSA), which is found on all the Arm Cortex-R-series processors and most of the Arm Cortex-M-series processors (Arm Cortex-M0+, Arm Cortex-M3, Arm Cortex-M4, Arm Cortex-M7, Arm Cortex-M23, and Arm Cortex-M33 processors) that are included in the Arm Flexible

Access program. In Arm Cortex-M-series processors and the Arm Cortex-R5 processor, MPUs are optional. In addition, you can specify how many MPU regions you want a core to support.

Neon

Neon is an advanced Single Instruction Multiple Data (SIMD) architecture extension for Arm Cortex-A series processors and the Arm Cortex-R52 processors. The technology is intended to improve the multimedia user experience by accelerating audio and video encoding and decoding, user interfaces, and 2D/3D graphics or gaming. Neon can also accelerate signal processing algorithms and functions. This means that Neon can speed up applications like audio and video processing, voice and facial recognition, computer vision and deep learning. Neon is optional on all Arm processors. If you do not require its capabilities for your software, you can reduce the area of your SoC design by excluding it.

The Neon instruction set is reliant on an FPU being included. On the Arm Cortex-A32, Arm Cortex-A34, Arm Cortex-A35, and Arm Cortex-A53 processors, Neon and the FPU are coupled together, so you need to include both or none.

Snoop Control Unit

A Snoop Control Unit (SCU) maintains data cache coherency between different cores and arbitrates between cores requesting level 2 access. The SCU is required for multi-core processor configurations and non-multi-core configurations where an ACP is present. The SCU is available on the Arm Cortex-R8 processor and Arm Cortex-A-series processors. The Arm Cortex-R5 processor has a Micro Snoop Control Unit (μ SCU).

SCUs significantly increase the area of a processor when they are included in the design. This tradeoff is necessary when a multi-core processor is required.

Accelerator Coherency Port

An Accelerator Coherency Port (ACP) is an [AXI](#) slave interface, which allows external, non-cached, intelligent peripherals, for example DMA controllers, companion DSPs, and Ethernet or Flexray interfaces, to access cacheable memory belonging to the core or cores of the processor.

To maintain cache coherency, access attempts are checked in all shared cached locations in the processor cluster. This data cache sharing typically boosts performance when the external memory access latency is long.

An ACP requires the presence of an SCU in a non-multi-core processor configuration. This technology is not an option for Arm Cortex-M-series processors, the Arm Cortex-R52 processor, or the Arm Cortex-A7 processor.

Interrupt controllers

Interrupt controllers manage interrupt requests (IRQs) and can be integrated or external. In relation to PPA analysis, integrated, built-in interrupt controllers will add to the area figure in the PPA analysis data. The more interrupts that are supported, the more space that is used. For example, an Arm Cortex-M0 processor supports 1-32 interrupts. However, an Arm Cortex-R52 processor can support 32-960 interrupts. When considering how many interrupts your processor needs to support, keep in mind that:

Each interrupt will contribute to the area of the SoC. External interrupt controllers contribute to the overall area of the SoC even if they do not contribute to PPA analysis area figures. Having more interrupts might make it harder for individual cores to achieve higher performance figures. All Arm Cortex-M series processors and the Arm Cortex-R52 processor have an integrated interrupt controller.

Arm Cortex-A5 multi-core processors and Arm Cortex-R8 processors have integrated interrupt controllers, but these can be set to support 0 IRQs, which effectively disables them and allows an external controller to be used. The Arm Cortex-A7 processor has the option of an integrated interrupt controller, but an external interrupt controller can be used instead.

The Arm Cortex-R5 processor and other Arm Cortex-A series processors available in the Arm Flexible Access program only support an external interrupt controller.



When looking at PPA analysis data, be aware of how the interrupt controllers have been set up. Are they integrated or external? How many IRQs were specified in the implementation? You might have plans for the setup of the interrupt controller that differ from the PPA implementation, and you should consider how your plans could affect the size of your final SoC design.

Low Latency Peripheral Port

A Low Latency Peripheral Port (LLPP) is a feature of Arm Cortex-R series processors which integrates latency-sensitive peripherals more tightly with the processor. LLPPs bypass the main AXI bus and ensure I/O access to the peripheral is not blocked by queued transactions.

The inclusion of an LLPP will not have any noticeable effect on the area figures in a PPA analysis, but it may offer a performance improvement for the SoC overall.

4. Choosing the fab process

Running a PPA analysis requires that the analysis team chooses a fab process that could potentially realize the IP. There are three choices to make, all of which can have a large effect on the results of the PPA analysis:

- The fab itself. Processes with identical names can vary considerably from fab to fab. Choosing the fab is as important as choosing the process.
- The fab process. Fabs generally offer a range of processes from older, slower, budget processes to new faster cutting-edge processes.
- The fab process option. Process options are design by the fab to optimize for specific properties, for example performance or power.



Unless specified otherwise, Arm PPA analysis data sets are for [TSMC](#) processes.

In this section, the significance of channel length, also called gate length, is explored in relation to processes. In addition to channel length, physical IP libraries specify other critical dimensions, which are discussed in [Choosing the physical IP libraries](#).

Introduction to fab process nomenclature

Fab processes were historically named after the minimum channel length, for example [μm](#), [250nm](#), [90nm](#), [32 nm](#), [22nm](#), or [10nm](#), that the process supported. The transistors in the cells span the channel. Smaller channel lengths result in faster performance, and therefore later processes have names which indicate a smaller minimum channel length. Older processes are named in micrometers and new processes are named in nanometers.

In modern processes, the minimum length of a channel no longer defines the name of the process. Instead, the name designates some other minimum feature size. However, what this feature size refers to varies from fab to fab, and therefore the drift between fabs regarding their processes has increased. Although channel length no longer defines the name of the process, process design rules still specify a minimal value for this critical dimension, even though it can now be a larger value than the process name. In some cases, a finite choice of channel length options is defined. Physical IP libraries that are designed for 40nm process and below incorporate different channel lengths including the minimum allowed.



Arm PPA analysis data sets specify the channel length of the physical IP library with which the analysis is performed. When reviewing PPA analysis data, remember that within the same process, you might have the option to move to a smaller channel length. Doing so will increase performance but at the cost of increased use of static power.

An example of fab process options

This section looks at the options which TSMC, the largest dedicated independent fab in the world, offers for their 28nm process:

| Process option | Description |
|----------------|---------------------------------|
| HP | High-performance |
| HPM | High-performance mobile |
| HPC | High-performance computing |
| HPL | High-performance, low-power |
| LP | Low-power |
| HPC+ | High-performance computing plus |
| ULP | Ultra-low power |

Imagine that the analysis team had settled on running a PPA analysis using the TSMC 28nm processes. They would now have to choose one of the process options above. In some cases, they might run more than one analysis, to give a broader view of what the IP could achieve in real situations.



Many of the PPA analysis data sets for processor IP offered as part of the Arm Flexible Access program were, in fact, calculated using the TSMC 28nm process. Arm PPA analysis data always specifies the process option chosen, regardless of which process was used. When reviewing PPA analysis data, remember that simply switching to another process option may get you closer to the ideal SoC implementation for your project.

How physical IP libraries fit with fab processes

Fab processes define a set of design rules. Physical IP libraries are designed for a specific process, giving options while at the same time adhering to the design rules of the process. For example, Arm physical IP libraries designed for the TSMC 28nm process support channel lengths of 38nm and 32nm. Choosing the appropriate library allows the analysis team to exercise judgement regarding the channel-length critical dimension, while ensuring that the physical implementation is correct for the process. [Choosing the physical IP libraries](#) contains more information on the options presented by physical IP libraries.

What does post-shrink mean in relation to a fab process?

As mentioned in [How physical IP libraries fit with fab processes](#), the industry moves forward to a new process about every two years. Between these full process nodes, advancements in lithography techniques allow fabs to develop what are called half nodes. The improvements in manufacturing allow the same circuit from a full node to occupy less silicon real estate than the preceding full node.

In these situations, the lengths and areas in the physical layout are considered pre-shrink. This means that the half-node optimizations have not yet been applied to the analysis. To calculate the area that the circuit will truly take up on silicon, optical shrink must be considered. 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 (0.9×0.9). The shrink factor for a half-node is not always 0.9 and is decided upon by the fab.



Arm PPA analysis data sets always present post-shrink area data if a half-node was chosen to perform the analysis.

5. Choosing the physical IP libraries

Once the process has been chosen, the analysis team must choose which physical IP library to use. Physical IP libraries are designed for specific processes and offer options in the following areas:

- Channel length
- Track height
- Voltage threshold
- High performance kits

Ultimately, the choice of physical IP library enables further adjustment of the manufacturing outcome beyond the choice of process option. For the remainder of this section, we refer to the Arm Standard Cell Libraries, which are production-quality physical IP libraries. We assume that the analysis team selected exclusively from these libraries for their implementation.



Unless specified otherwise, Arm Standard Cell Libraries were used to produce Arm PPA analysis data.

It is also possible to hybridize within the different options that the Arm Standard Cell Libraries provide. For example, it is possible to have 35% low voltage threshold cells and 65% high voltage threshold cells in an implementation. This is discussed in [Hybridizing the options given by Arm Standard Cell Libraries](#).

[Memory models](#) explores how memory on a piece of IP, for example a cache, is compiled into memory models.

How the target frequency determines physical IP library usage

The EDA tools decide which cells to use from the cell libraries that are available. The tools work to meet the specified target frequency and use hybridization as required to achieve this. Appropriate cell libraries, containing cells that can provide performance for critical paths on the silicon, must be available for the tools to use.

The target frequency must be chosen with care. Choosing a target frequency that is too high results in the area and power usage being higher than if a realistic value was chosen. In addition, the implementation will not achieve the overly high value.

Channel length

Channel length was discussed in [Introduction to fab process nomenclature](#). Although the minimum value for this critical dimension originally defined the name of a process, the option to vary this dimension exists in Arm Physical IP libraries for the newer processes, that is, those at 40nm processes and below. The minimum channel length remains a process rule to which libraries designed for a specific process must adhere.

Choosing a channel length that is longer than the minimum allowed does not usually increase the area of the IP. This statement might seem counterintuitive because channel length was, for many years, used to name processes. With every new advancement, more transistors could be fitted into a smaller area. However, each advancement represented an overall shift forward in the fundamental size of the process technology. This drive towards a reduction in fundamental size continues into the newer processes, for example 16nm, even though the minimum channel length for such processes is now larger than the naming size.

In fact, library cells for a process are often 100% footprint compatible. This means that the length of the cell is standardized, and the channel length is contained within the overall length of the cell. You can think of channel lengths that are greater than the minimum value being absorbed. For example, TSMC's 28nm process supports channel lengths of 31nm and 38nm. However, when footprint compatible cells are used, neither choice will result in an increase in the manufactured IPs area. Although the analysis team could, for some processes, choose very large channel lengths, this does not make economic sense. Arm Standard Cell Libraries provide a limited range of realistic channel lengths for a process, which ensures that the implemented cells have a footprint compatible size.

A good question is why the analysis team would choose a channel that is length greater than the minimum value allowed? In the previous example, using the shorter channel length of 31nm creates faster transistors but at a cost to static power. The 38nm channel length is slower but the transistors in the cell will have lower static power usage.

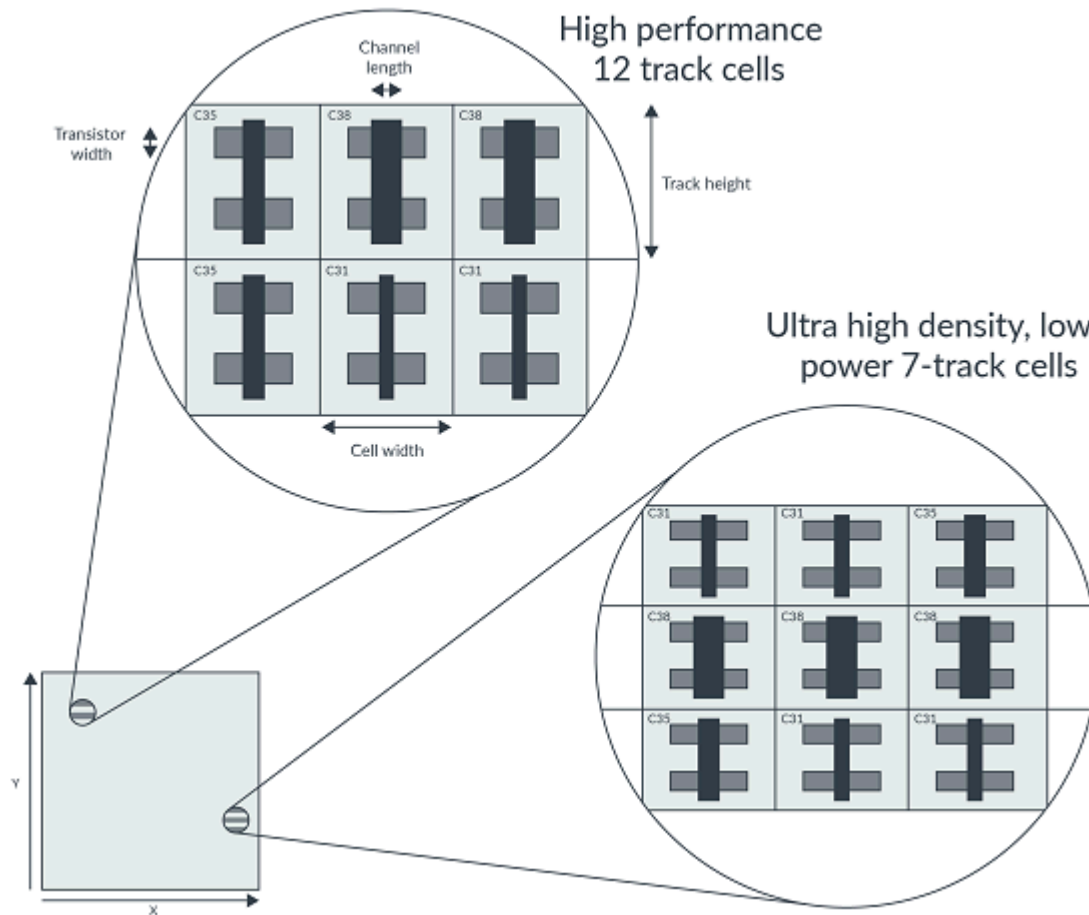
Note: Arm PPA analysis data records the channel length used, for example C38 or C31. When reviewing PPA analysis data, consider increasing the channel length in situations where the IP appears capable of providing excess performance for your requirement, but the static power consumption is too high.

Because the footprint of the cells is not affected by channel length, it is also possible to have cells of varying channel length in your implementation. [Hybridizing the options given by Arm Standard Cell Libraries](#) discusses further why varying channel lengths might be used.

Track height

Cells are sequentially placed along tracks, which are stacked one on top of the other on the silicon die. It is important to remember that track height in fact runs perpendicular to the channel length dimension and should not be thought of as a z dimension. Track height is itself a constant defined by the process design rules. One track constitutes the minimum allowable spacing between the Metal 1 routes in a process.

Cell heights are expressed in tracks. The height of the cells in an IP implementation can be varied if they adhere to the design rules of a process. When a cell is higher, it allows for a larger transistor width to be used. To confirm, the transistor width dimension also runs perpendicular to the channel length dimension. If you think of channel length running along the x dimension, and the transistors spanning the channel, then cell height and transistor width run along the y dimension. The following figure shows a comparison between the layout of 12-track footprint compatible cells and the layout of 7-track footprint compatible cells on a die:

Figure 5-1: Comparison

Wider transistor widths equate to higher performance but also increase the area of the implemented IP. When higher track heights are used, the increase in area is felt along the x and y dimensions, and the die remains a square.

Regarding track height, Arm Standard Cell Libraries can be split into the three main categories that are shown in the following table:

| Track | Characteristic | Usage |
|---------------|----------------------------------|--------------------------------|
| 7 or 8-track | Ultra-high density and low power | For cost critical applications |
| 9 or 10-track | High density | For mainstream applications |
| 12-track | High performance | For speed critical designs |



Arm PPA analysis data records the track height of the cells that are used: SC7, SC8, SC9, SC10, or SC12. When reviewing PPA analysis data, consider the effect of the track height on the results. Making changes to the track size could bring an IP implementation closer to your exact SoC requirements.

Voltage threshold

The voltage threshold of the cells in an IP implementation can be varied if they adhere to the design rules of a process. The voltage threshold has an impact on the switching speed of the transistors inside a cell, and therefore has a direct impact on performance. In addition, the voltage threshold also has an impact on static power consumption. High voltage threshold cells are slower, because they require more voltage to switch on, but consume less static power. Low voltage threshold cells are faster but consume more static power. Ultra-high and ultra-low variants, and a standard option, are available for cell voltage thresholds.



Arm PPA analysis data records the voltage threshold of the cells used: ULVt, LVT, SVt, HVT, and UHVt. The tradeoff between power and performance that sometimes needs to be made when designing an SoC is exemplified by the voltage threshold options that are available. If you need to make this tradeoff, keep the voltage thresholds of the cells in mind when reviewing PPA analysis data.

Add-on kits for the Arm Standard Cell Libraries

Depending on the process, Arm offers add-ons to the Arm Standard Cell Libraries, which contain additional selected cells and can be chosen instead of base cells. The add-on kits are tailored to a specific purpose, for example:

- Arm High Performance Kit (HPK)
- Arm Power Management Kit (PMK)
- Arm Low Power Kit (LPK)

The LPK and PMK implement low-power techniques like power gating and dynamic voltage scaling. The HPK employs advantageous circuit tuning practices inside the logic gates to achieve higher performance with a minimal impact on area and power.



Arm PPA analysis data records whether any of the add-on kits were used. The tradeoff between power and performance that sometimes needs to be made when designing an SoC is exemplified by the add-on kits available. If you need to make this tradeoff, keep in mind which add-on kits were used when you review PPA analysis data.

Hybridizing the options given by Arm Standard Cell Libraries

Each individual cell library represents a choice for each of the four options available for a cell: channel length, track height, voltage threshold, and add-on kit. Add-on kits extend the other three options over the add-on kit. Every combination of the options for a cell is available. This means that you can choose different combinations for different percentages of cells in the implementation.

In the example shown in the following table, a team carrying out an [AFAP analysis](#) on a processor chose to use four different libraries in the implementation. Each row in the table corresponds to an individual cell library. Two base libraries and two libraries from the Arm High Performance Kit were used:

| Percentage usage | Track size | Kit | Voltage threshold | Channel length |
|------------------|------------|------|-------------------|----------------|
| 60 | SC12 | Base | SVt | C31 |
| 25 | SC12 | Base | LVt | C31 |
| 5 | SC12 | HPK | SVt | C31 |
| 10 | SC12 | HPK | LVt | C31 |

The above choice is for the TSMC 28nm process. Because this is an AFAP implementation, a maximum track size and minimum channel length were chosen for all cells. This is expected, because both options maximize performance.

The other two performance boosts are used more sparingly. Only 10% of the cells are completely maximized for performance, because they come from the HPK and have a low voltage threshold. The HPK provides another 5% of the cells, but these cells have a standard voltage threshold. Another 25% of the cells have a low voltage threshold, but are selected from the base kit. In fact, it would waste power and area to use the HPK and a low voltage threshold for all the cells. The analysis team have used the EDA tools to save on power and area. The tools choose smaller and less power-hungry cells on paths that are not critical for timing. Even when an AFAP analysis runs, this additional area and power recovery step is performed in the implementation flow, specifically to improve the power and area results.



Note

Arm PPA analysis data records cell library hybridization. Some analyses, for example targeting minimum power and area usage, may use a single cell library. However, it is very likely that you would use hybridization, through the EDA tools, to save on power and area whatever the performance requirements of your SoC design. Although hybridization provides the ultimate flexibility in the tradeoff between power and area, cost, and performance, the design rules of some processes place limitations on this. For example, a process design rule may limit the amount of different voltage thresholds that can be combined on any given SoC.

Memory models

Arm PPA analysis uses production quality memory models with real timing. Memory compilers are used to create the memory models. For processors that include memory as part of the processor, for example level 1 and level 2 caches, or TCMs, the size and type of the memory included can have a major impact on PPA analysis.

Regarding 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 usage. Two 32KB caches will have a significantly higher area and power usage than two 16KB caches. However, it is important that a PPA analysis is realistic. Completely omitting caches from an analysis where caches are likely to be used in the real world is not helpful. Even when Arm runs [AFAP or Minimum analyses](#) for a processor, the size of the level 1 data and instruction caches is the same as for the Featured analysis. This enables these analyses to be more meaningfully compared.

Different compilers create memory with differing characteristics:

| Compiler | Description |
|----------------------------------|--|
| Arm Artisan RF High-Density SP | Lowest area and power usage |
| Arm Artisan RF High-Speed SP | Higher speed but higher area and power |
| Arm Artisan Fast Cache Instances | Optimized for L1 memories |



Arm PPA analysis data records the memory compiler that is used for the SoC memory. 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 at. However, faster memories take up more area and consume more power. You need to be aware of the tradeoff between processing frequency and power usage, and you need to use the smallest memories possible to achieve the performance that you require.

6. Further choices that affect PPA implementation analysis

The preceding three sections have provided a full understanding of the decisions that need to be made before performing a PPA analysis. We showed how an analysis team defines a physical implementation to analyze. This section explores further choices that the team must make when they run the analysis. Remember these decisions when you interpret PPA data. Also, be aware that some of the constraints applied at this part of the proceedings are hypothetical. For example, it is important to choose a temperature at which the IP is theoretically running. However:

- The value chosen might represent an extreme, undesirable situation that is unlikely to occur in the real world.
- PPA tools are used to simulate the environment, and the IP is never realized to test the findings.

Silicon speed, voltage, and temperature ranges

This section looks at the outlying values, and the median value, for three factors that affect the power and performance figures in PPA analysis data. The PPA tools take these factors into account when generating the data. However, it must be decided which values to use on each of the ranges when generating the power and performance figures. The IP is not expected to operate outside of the outliers.

The following table describes the three ranges:

| Range | Description |
|---------------|--|
| Silicon speed | <p>During integrated circuit semiconductor fabrication, variation from the nominal doping concentrations in the transistors on a silicon wafer can occur. This affects the carrier mobility (both electron and hole mobility) in the transistors. This variation can cause significant changes in the speed at which the digital signals transition from high to low and from low to high. Silicon speed is described by two-letter designators, with the first letter for the n-channel MOSFET speed and the second letter for the p-channel MOSFET speed.</p> <p>For example:</p> <ul style="list-style-type: none">• ss• tt• ff <p>The letter t refers to typical doping concentrations, in between the two outliers, and therefore a normal speed. The letter s refers to a carrier mobility that is slower than normal, which is reflected in the speed. The letter f, on the other hand, refers to a carrier mobility that allows for faster than normal performance.</p> <p>For example, if the ss outlier is selected, an assumption has been made that the analysis is for silicon which, because of its chemical makeup, has the slowest performance possible.</p> <p>Because of reduced process variability, more modern processes offer two new outliers: ssg (global slow) and ffg (global fast). These outliers represent high-yielding silicon, which can provide a 10-15% performance boost over more conservative silicon represented by ss and ff.</p> |

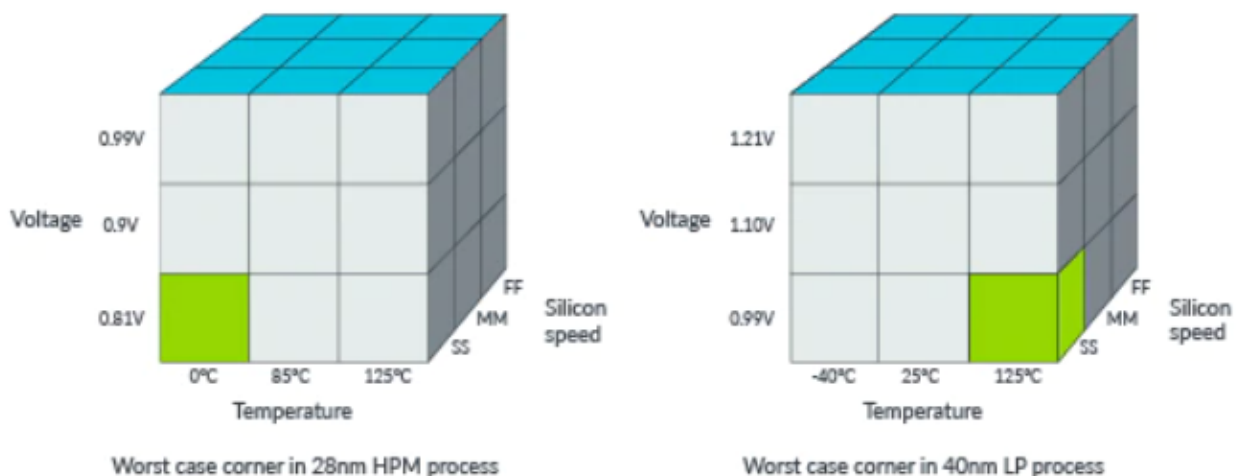
| Range | Description |
|-------------|--|
| Voltage | <p>The transistors in a piece of IP are expected to operate over an acceptable variance in voltage. Transistors operate faster at higher voltages and slower at lower voltage s. The worst-case slow outlier is assumed to be -10% from the typical voltage, and the best-case fast outlier is assumed to be +10% from the typical voltage.</p> <p>For example, if the typical, median voltage is 0.9V, then the best-case, high voltage outlier is 0.99V, and the worst case, low voltage outlier is 0.81V.</p> |
| Temperature | <p>The transistors in a piece of IP are expected to operate over an acceptable variance in temperature, which is typically -40°C to 125°C. It is very important to note that the worst-case outlier for a process of 40nm or above is 125°C, but this switches to 0°C for processes under 40nm.</p> |

For power and performance analysis, it is very important to know where on the process, voltage, and temperature range the analysis was carried out.

Worse case corner

Next we will look at the concept of a worst-case corner, which assumes the most pessimistic scenario for all the ranges. The following figure show a diagrammatic representation of worst-case corners for processes of 40nm or above, and for processes under 40nm:

Figure 6-1: worst-case corners



How power figures are affected

The power usage for a piece of IP is always calculated using typical values for silicon speed, voltage, and temperature as opposed to corner cases. Corner cases use extreme values that are unlikely to occur in a real-world situation and are not used when calculating power usage. This means that in a real-world situation, particularly if you take steps to ensure normal conditions, the power usage should be close to the calculated values if a similar IP setup is used. However, if you require a power usage below what a piece of IP can give under typical conditions, you should:

- Decrease the power usage of the IP by adjusting the physical IP library variants or by choosing a different process or process option.
- Select a piece of IP from the Arm Flexible Access program which has lower power usage.

How performance figures are affected

The frequency obtainable for a piece of IP is always calculated using the worst-case corner for silicon speed, voltage, and temperature. This frequency value has a direct effect on any benchmark values which are calculated during the analysis. Using an IP setup that is like the PPA analysis may give you better performance in a real situation, particularly if you take steps to ensure a consistent voltage or temperature. However, if you consistently require more performance than a piece of IP can give when the worst-case corners are used, you should:

- Increase the performance of the IP by adjusting the physical IP library variants or by choosing a different process or process option.
- Select a piece of IP from the Arm Flexible Access program which has better performance.

Margins and OCV derates

As we discussed previously, various factors affect the frequency, and therefore the achievable performance, of a piece of IP, for example:

- The options chosen for the IP
- The fab process and process options that are selected
- The physical IP libraries used in the implementation
- The worst-case corner for silicon speed, voltage, and temperature

Some other modifiers are discussed in the following subsections.

Margins

Fabs recommend setup and hold margins that provide a safety margin against potential failure in manufactured IP. When Arm produces physical implementations for the purpose of PPA analysis, these safety margins are added to all timing paths in the design. Incorporating the safety margins effectively reduces the performance while ensuring consistent operation. In addition to adhering to the foundry recommendations, the margins that Arm analysis teams add also consider clock period jitter. This is because the [PPLs](#) on which IP clocks rely might contain imperfections.

On-chip variation derates

An On-chip Variation (OCV) derate is a modification in performance which accounts for the effect of timing variation in the silicon that arises as part of the manufacturing process. As the feasible size of manufactured IP becomes smaller and smaller, because of more advanced processes, the significance of chip variation increases. Using an OCV derate attempts to model natural variation in the manufacturing process so that resulting PPA figures are more realistic.

Conclusions on the effect of margins and OCV derates

Effectively, taking margins and OCV derates into account gives an even more realistic view of the frequency obtainable by the IP than one derived for the worst-case corner. Final frequency figures generated for Arm PPA analysis include applied margins and OCV derates. In some cases, you will notice that a slightly more favorable frequency, which is for the worst-case corner with no further adjustment, is provided.

Using the fab-recommended margins on all timing paths makes physical implementations, generated by Arm, representative of IP on a realized SoC. However, the margins also have an

impact on area. For example, in the case of a 28nm fab process, up to 10% of the total area can be a result of:

- Hold fixing on functional and scan paths, because of large fab-recommended hold margins
- Application of high OCV derates to timing paths

When you compare pieces of IP, you need to check whether these precautionary measures have been applied to each PPA analysis. If these measures are neglected, the result might be a favorable analysis, which leads to an unacceptable final product when the implementation is realized.

7. How can higher performance increase the area of implemented IP?

This example examines how area is affected across implementations of a processor as a result of the target frequency setting, IP option selection, and physical IP library choice. The analyses for three implementations, [Minimum](#), [Featured](#), and [AFAP](#), are compared against each other. The comparison is interesting because, regarding fab processes, everything is the same for all three implementations. This means that the effect of the variants can be clearly shown.

The first three sections showcase the data of each of the implementations. Examine the data in these three sections before reading the breakdown of the results in the final section.

Minimum implementation example

The following tables contain PPA data that was obtained from a Minimum implementation, including data on the implementation decisions.

| Power | Performance | Area |
|--------------------|-----------------------------|------------------------------------|
| Dynamic 77.0mW/GHz | Maximum frequency 484.5 MHz | 0.330mm ² (Post-shrink) |
| Static 2.39 mW | | |

| Silicon process | - |
|-------------------|------------------|
| Fab | TSMC |
| Process | 28nm |
| Process option | HPM |
| Post-shrink scale | 0.81 (0.9 x 0.9) |

| Physical IP | - | - | - | - |
|------------------|--------------------------------|-----------------------------|------------|-------------------|
| Cell libraries | | Arm Standard Cell Libraries | | |
| % | Kit | Channel length | Track size | Voltage threshold |
| 100 | Base | C38 | SC9 | SVt |
| Memory libraries | Arm Artisan RF High-Density SP | | | |

| IP config | - |
|---------------------------------|---------------|
| CPU | Single core w |
| Level 1 caches | Yes |
| I-cache size | 32K |
| D-cache size | 32K |
| TCMs | No |
| I-TCM size | N/A |
| D-TCM size | N/A |
| Integrated interrupt controller | No |
| IRQs | N/A |

| IP config | - |
|-----------|-----|
| ACP | No |
| ECC | No |
| ETM | No |
| FPU | No |
| LLPP | No |
| MPU | No |
| Neon | N/A |
| SCU | No |

| Process conditions | - | - |
|--------------------|-------------------|-------------------------|
| | For power figures | For performance figures |
| Silicon speed | Typical (tt) | Slow (ss) |
| Voltage | 0.9V | 0.81V |
| Temperature | 85°C | 0°C |
| Margin | 50ps | |
| OCV | 8% | |

Featured implementation example

The following tables contain PPA data that was obtained from a Featured implementation, including data on the implementation decisions.

| Power | Performance | Area |
|----------------------|----------------------------|------------------------------------|
| Dynamic 123.2 mW/GHz | Maximum frequency 1000 MHz | 0.837mm ² (Post-shrink) |
| Static 13.051 mW | | |

| Silicon process | - |
|-------------------|------------------|
| Fab | TSMC |
| Process | 28nm |
| Process option | HPM |
| Post-shrink scale | 0.81 (0.9 x 0.9) |

| Physical IP | - | - | - | - |
|----------------|------|-----------------------------|------------|-------------------|
| Cell libraries | | Arm Standard Cell Libraries | | |
| % | Kit | Channel length | Track size | Voltage threshold |
| 16.5 | Base | C31 | SC9 | SVt |
| 8.8 | Base | C35 | SC9 | SVt |
| 28.5 | Base | C38 | SC9 | SVt |
| 33.7 | Base | C35 | SC9 | HVt |
| 3.4 | HPK | C31 | SC9 | SVt |
| 1.7 | HPK | C35 | SC9 | SVt |
| 4.2 | HPK | C38 | SC9 | SVt |
| 3.1 | HPK | C35 | SC9 | HVt |

| Physical IP | - | - | - | - |
|------------------|--------------------------------|---|---|---|
| Memory libraries | Arm Artisan RF High-Density SP | | | |

| IP config | - |
|---------------------------------|---------------|
| CPU | Single core w |
| Level 1 caches | Yes |
| I-cache size | 32K |
| D-cache size | 32K |
| TCMs | Yes |
| I-TCM size | 32K |
| D-TCM size | 32K |
| Integrated interrupt controller | Yes |
| IRQs | 32 |
| ACP | Yes |
| ECC | Yes |
| ETM | Yes |
| FPU | Yes |
| LLPP | No |
| MPU | No |
| Neon | N/A |
| SCU | No |

| Process conditions | - | - |
|--------------------|-------------------|-------------------------|
| | For power figures | For performance figures |
| Silicon speed | Typical (tt) | Slow (ss) |
| Voltage | 0.9V | 0.81V |
| Temperature | 85°C | 0°C |
| Margin | 50ps | |
| OCV | 8% | |

As Fast As Possible implementation example

The following tables contain PPA data that was obtained from an AFAP implementation, including data on the implementation decisions.

| Power | Performance | Area |
|----------------------|----------------------------|------------------------------------|
| Dynamic 143.0 mW/GHz | Maximum frequency 1538 MHz | 0.628 mm ² (Post-shrink |
| Static 92.6 mW | | |

| Silicon process | - |
|-------------------|------------------|
| Fab | TSMC |
| Process | 28nm |
| Process option | HPM |
| Post-shrink scale | 0.81 (0.9 x 0.9) |

| Physical IP | | - | - | - |
|------------------|----------------------------------|-----------------------------|------------|-------------------|
| Cell libraries | | Arm Standard Cell Libraries | | |
| % | Kit | Channel length | Track size | Voltage threshold |
| 60.5 | Base | C31 | SC12 | SVt |
| 23.5 | Base | C31 | SC12 | LVt |
| 5.2 | HPK | C31 | SC12 | SVt |
| 10.8 | HPK | C31 | SC12 | LVt |
| Memory libraries | Arm Artisan Fast Cache Instances | | | |

| IP config | - |
|---------------------------------|---------------|
| CPU | Single core w |
| Level 1 caches | Yes |
| I-cache size | 32K |
| D-cache size | 32K |
| TCMs | No |
| I-TCM size | N/A |
| D-TCM size | N/A |
| Integrated interrupt controller | no |
| IRQs | N/A |
| ACP | No |
| ECC | No |
| ETM | No |
| FPU | No |
| LLPP | No |
| MPU | No |
| Neon | N/A |
| SCU | No |

| Process conditions | - | - |
|--------------------|-------------------|-------------------------|
| | For power figures | For performance figures |
| Silicon speed | Typical (tt) | Slow (ss) |
| Voltage | 0.9V | 0.81V |
| Temperature | 85°C | 0°C |
| Margin | 50ps | |
| OCV | 8% | |

Breakdown of results

As expected, the Featured implementation has the largest area. The addition of IP options increases the number of cells that are required. However, the fact that the AFAP implementation is larger than the Minimum implementation may seem unusual, especially because both implementations use minimal IP options.

Notice that the Featured implementation takes some steps to allow for a certain level of performance above that of the Minimum implementation. For example, although not to the level of the AFAP, the Minimum implementation employs hybridization when selecting the cell libraries. By doing so, the Minimum implementation allows high performing cells to be selected for critical paths. In this sense, the Featured implementation is aiming for a configuration which could have a practical use.

The reason for the larger area observed in the AFAP implementation is that high performance cells require a higher area. The observation does not seem supported with regard to channel length. The Minimum implementation uses a longer cell channel length (C38) than the AFAP, which exclusively uses C31 cells. The longer channel length lowers the power consumption of the processor for the Minimum implementation. However, because channel length does not increase the size of footprint compatible cells, the fact that the AFAP uses a shorter channel length, for increased performance, does not result in an area reduction.

The Minimum implementation uses a single cell library across the board. This is because, if minimum power usage is a priority, there is no need to speed up any specific groups of cells. In contrast, the Featured and AFAP implementations use higher performing cells for critical paths. Regarding performance boosting, the AFAP goes further by:

- Exclusively using C31 cells
- Using cells with a track size of 12. Wider cells support high performance.
- Using cells with a low voltage threshold. Low voltage threshold cells are faster but consume more static power.

Two things are responsible for pushing up the areas of the AFAP implementation:

- The high track size
- The ambitious target frequency for the processor, which means that the EDA tools use larger, higher drive strength cells from the libraries

The Minimum implementation is best in terms of power usage. Higher channel length, smaller track size, and high voltage threshold cells all contribute to lower power usage. Also, when the demands for performance are less, higher density cells can be used, thereby reducing the area.

8. Performing PPA analysis for a system

There is nothing greatly different in running a PPA analysis for an entire system, for example an SoC, rather than a single piece of IP. However, analyzing a system does add more complexity. A single piece of IP usually contains one or two clocks. However, a system will contain many clocks with different modes of operation, and the PPA analysis team must decide what frequencies to set the clocks to. In addition, to drive meaningful power optimizations, the team must understand the power story of the system, and how the individual IPs perform in the context of a complete system. Different parts of the system might also run at different operating voltages.

A team that is tasked with generating system PPA data will have to repeat the following steps when pieces of IP are reconfigured and new IPs are added in:

1. Set up the input for the system. This process is usually more complicated than setting up the input for a single piece of IP.
2. Derive the timing constraints across the entire system.
3. Derive a representative floorplan.
4. Profile the system. The team assesses the need to tune frequency targets, adjusts bus widths, and looks for opportunities to insert and remove IPs, for example register slicing and asynchronous crossing. In this step, the team is tackling the challenges of implementation while maintaining the performance requirements of the system. By repeating the preceding steps, the team is making sure that the system is implementable at each stage in its development. With a large system, arriving at a complete, implemented system can take months, partly due to the length of time that it takes to validate larger PPA trial implementations. If the team is building from a preconfigured system, the process can be sped up. This is because the system will have been built with a fab process and floorplan in mind.