**ARM cortex a series**

[**1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**종류**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1)

Cortex-A 시리즈가 적용된 명령어 셋은 ARMv7 부터다. 이전 ARMv6 명령어 셋으로 만들어진 아키텍쳐는 해당되지 않았다.

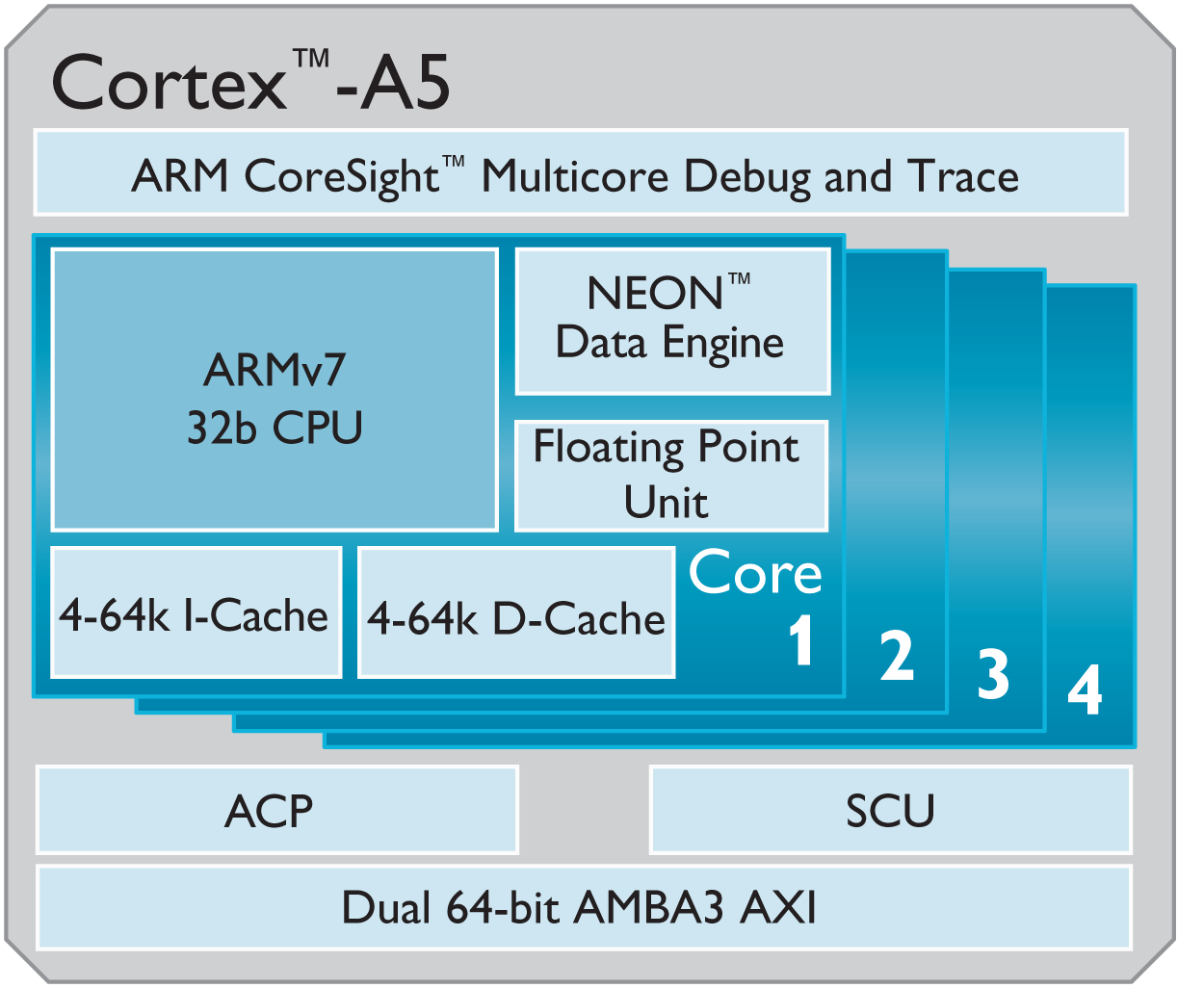
2013년 기준으로 현재 ARMv7의 후속 명령어 셋인 ARMv8이 공개되었고, 이 둘을 이용해서 Cortex-A 시리즈를 비롯해 직접 명령어 셋을 이용해서 제작하는 커스텀 아키텍쳐까지 활발히 만들어지고 있다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=2)]

[**1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**ARMv7**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=3)]

[**1.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A5**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.1)



[[PNG external image]](http://www.arm.com/images/Cortex_A5_large.png)

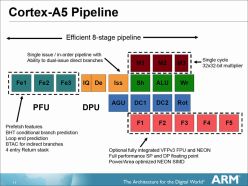
2009년 10월에 발표된 저사양 CPU 아키텍쳐다. ARMv7 명령어 셋으로 만들어진 CPU 아키텍쳐 중에서는 가장 성능이 낮다. 이는 Cortex-A5가 기존의 ARM11을 대체하는 아키텍쳐로 개발되었기 때문이다.

성능은 ARM11보다 약간 상회하는 정도지만, 전력 소모율은 ARM11 이전의 아키텍쳐인 ARM9와 비슷한 수준을 보여준다. 때문에 ARM에서는 아직도 흐르고 넘치는 100개 이상의 ARM11 및 ARM9 아키텍쳐 라이센스 취득사가 Cortex-A5를 대체재로 이용할 수 있도록 독려하고 있다.

또한 Cortex-A9처럼 멀티코어화가 가능하다. 때문에 임베디드 시스템 뿐만이 아니라 저사양 [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0)에도 종종 탑재되고 있으며 퀄컴 역시 스냅드래곤 200에 Cortex-A5 멀티코어 CPU를 사용하는 라인업을 잡아두었다. 다만, 쿼드코어 구성이라 하더라도 성능은 [옴니아 II](https://mirror.enha.kr/wiki/%EC%98%B4%EB%8B%88%EC%95%84%20II)에 들어간 ARM11을 네 개 붙여놓은 것과 비슷하기 때문에 많은 것을 기대하기는 어렵다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=4)]

[**1.1.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.1.1)



[[JPG external image]](http://ascii.jp/elem/000/000/579/579266/Photo12_248x.jpg)

* 8-pipeline
* 한 사이클에 1개의 명령어 디코더 (DE) - 64k
* NEON/vfp SIMD 유닛을 외장형태로 부착 가능
* issue명령 분배기 1개가 2~3개의 모듈에 파견

기본적인 연산 모듈은 2개

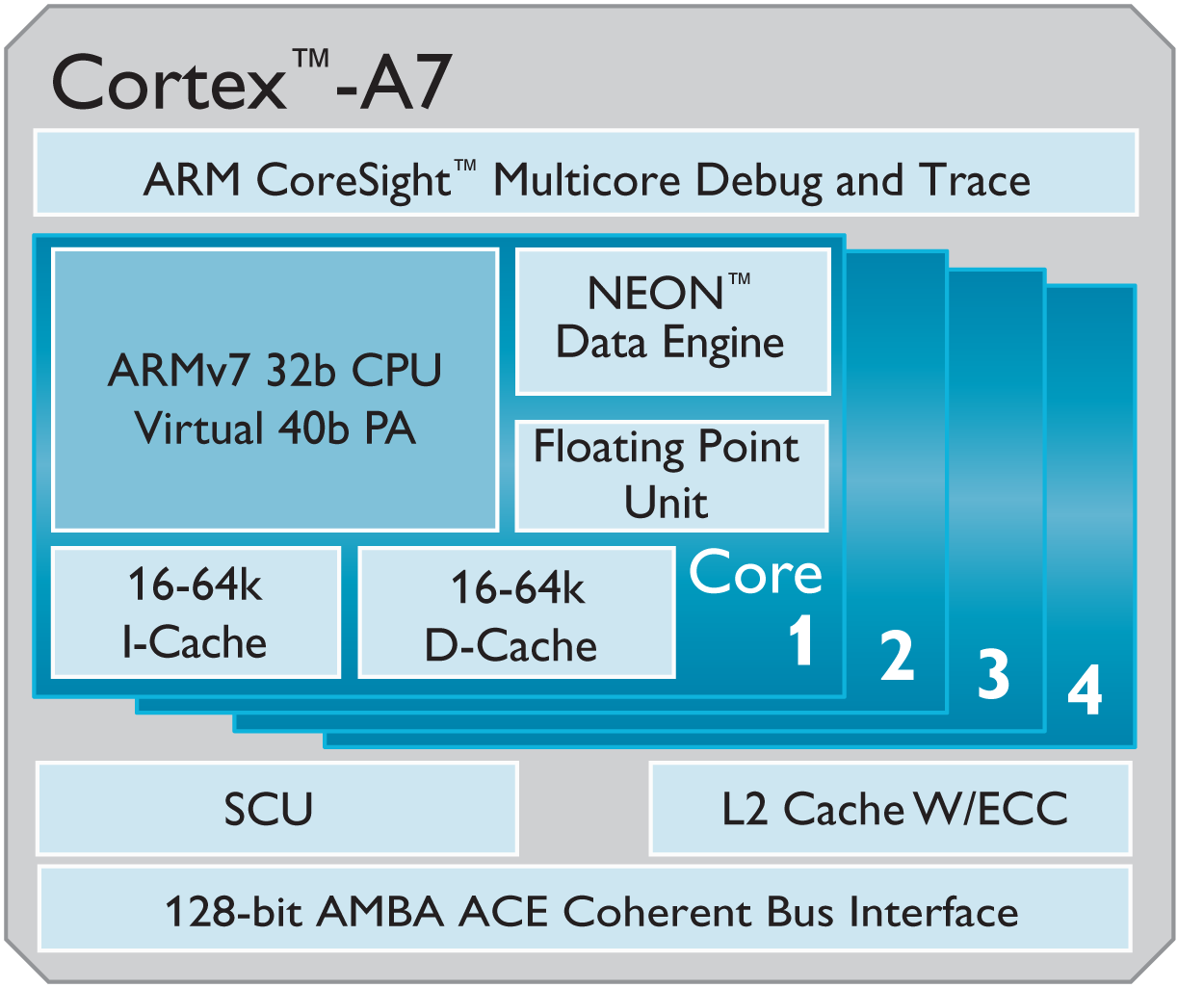
A-ALU  
B-Load/Store -AGU

C - Neon/vfp - 외장형 옵션

ARMv7 기반 아키텍처 중에서 유일하게 1개의 디코더를 가지고 있다. L2 Cache는 지원하지 않으며 Neon/vfp도 필요에 따라서 외장으로 따로 부착하는 형태이다. 다만 ARMv7 자체의 성능이 준수하기 때문에 아무리 저사양이라고 불리는 Cortex-A5의 성능은 ARM11의 120%정도.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=5)]

[**1.1.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A7**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.2)



[[PNG external image]](http://www.arm.com/images/Cortex_A7_large.png)

2011년 하반기에 발표된 엔트리급 CPU 마이크로 아키텍쳐다.

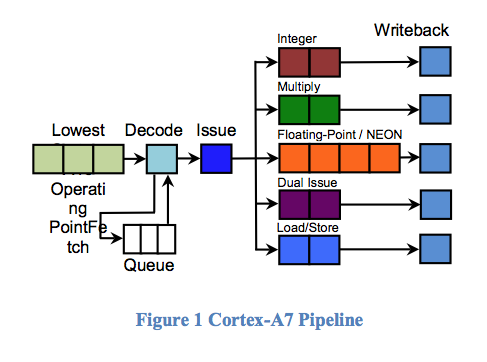
전체적으로 Cortex-A5의 연장선상에서 개발된 아키텍쳐다. 기술 진보로 인해 성능은 Cortex-A8과 비슷한 수준이지만 그 성능을 내기 위한 전력 소모는 5배 이상 적다[[1]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn1). ARM의 목표대로 2010년에 $500정도의 [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0) 성능을 2013년에 $100 ~ 200 정도의 [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0) 성능으로 만들어준 주역이다.

기본적으로 모든 유닛이 Cortex-A8과 같은 순차적 명령어 처리지만, Cortex-A9의 특징이었던 멀티코어 지원, NEON 및 L2 Cache 내장, ALU모듈의 나눗셈 지원 등이 추가되었다.

간단한 구조와 전력 소모가 적은 설계를 기반으로 고성능 CPU 아키텍쳐의 자비없는 전력 소모량을 보완하기 위한 big.LITTLE 솔루션에서 Cortex-A12와 Cortex-A15의 LITTLE 코어를 담당한다. 하지만 자체적인 성능역시 나쁘지 않고[[2]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn2) 설계와 양산도 쉽기 때문에 Cortex-A7로 이루어진 모바일 AP도 퀄컴과 미디어텍 그리고 여러 중국산 업체를 중심으로 활성화되고 있다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=6)]

[**1.1.2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.2.1)



[[PNG external image]](http://www.wired.com/insights/wp-content/uploads/2011/10/arma7-a7pipeline.png)

* 8-pipeline + Superscalar
* 한 사이클에 2개의 명령어 디코더(DE) - 128k
* 명령어 발행 어레이 개수 = 1
* Issue명령 분배기 1개가 5라인(in) 파견 5개
* L2 Cache 내장

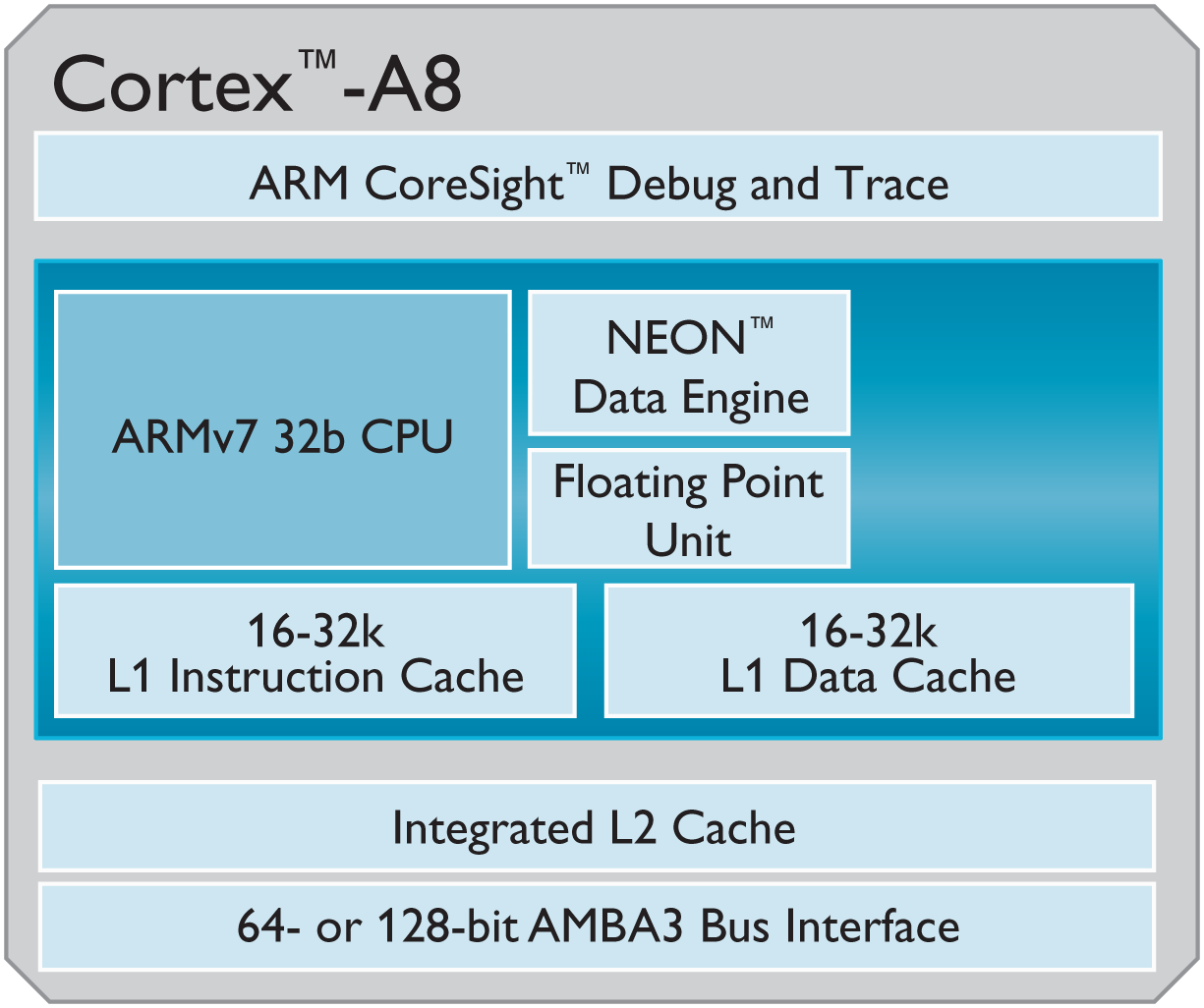
연산 모듈은 총 5개(A~E)

A - ALU(in)-정수연산 덧/뺄셈   
B - ALU(in)-정수연산 덧/뺄셈+곱/나눗셈  
C - NEONvfp   
D - Dual issue-Superscalar  
E - load/store

저전력을 목적으로 나온 코어답게 간단한 설계를 보여주고 있다. 기본적으로 모든 유닛은 순차적 명령어 처리 방식이지만, L2 Cache, NEON SIMD의 통합 등 진보적인 설계를 도입함에 따라 실제로는 Cortex-A8 이상의 성능을 보여주고 있다. Cortex-A7의 ALU는 Cortex-A12나 Cortex-A15의 ALU같이 나눗셈을 지원하며, 그 외의 다른 특징으로 Dual-Issue가 가능한 Supersclar 유닛을 추가하여 제한적으로 1사이클당 2개 명령어 처리가 가능하도록 되어 있고 기존 A8/A9와는 달리 pipeline구조 안으로 NEON 연산유닛을 통합함으로써 연산 성능을 향상시키고 있다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=7)]

[**1.1.3.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A8**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.3)



[[PNG external image]](http://www.arm.com/images/Cortex_A8_large.png)

2005년 중반기에 발표된 첫 ARMv7 기반의 CPU 아키텍쳐다.

이전 세대 ARMv6기반의 ARM11 아키텍처와 비교했을때 주요 변경점은 2issue in-order 슈퍼스칼라 처리구조가 도입되고, NEON SIMD가 기본으로 지원되면서 클럭당 처리능력, 멀티미디어 데이터 처리와 부동소수점 연산능력이 향상되었다.

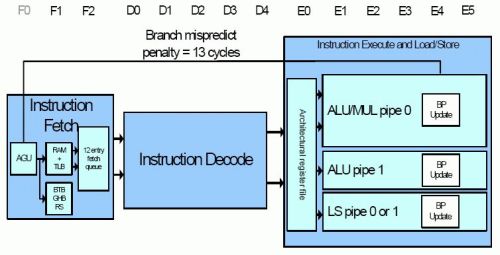
초기의 최대 클럭은 800 MHz로 이는 [ARM](https://mirror.enha.kr/wiki/ARM%20Holdings)이 [Texas Instruments](https://mirror.enha.kr/wiki/Texas%20Instruments)와 같이 설계했던 첫 Cortex-A8기반 AP인 [OMAP](https://mirror.enha.kr/wiki/OMAP) 34XX에서 최대 클럭을 800 MHz까지 밖에 끌어올리지 못했기 때문이다. 이후 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)가 코어 커스텀을 통해서 1 GHz를 돌파[[3]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn3)하고, ARM도 코어 리비전을 통해 1 GHz로 끌어 올림에 따라 보편적인 최대 클럭은 1 GHz가 되었다.

2005년에 발표되었지만, 통상 ARM제품군의 경우 아키텍쳐 발표에서 그 아키텍쳐를 적용하여 실제 제품군이 나오기까지 3~4년 정도 걸리기 때문에 A8코어를 적용한 제품들이 본격적으로 출시된 것은 2009-2010년 이었고 옴니아 HD같은 그나마 리소스 소모가 큰 제품에서 먼저 사용되었다. 이후 [애플](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%28%EA%B8%B0%EC%97%85%29)이 [아이폰 3GS](https://mirror.enha.kr/wiki/%EC%95%84%EC%9D%B4%ED%8F%B0%203GS)에 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)의 [S5PC100](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90%20%ED%97%88%EB%B0%8D%EB%B2%84%EB%93%9C) 600 MHz를 사용하면서 기존 ARM11을 사용한 [옴니아 II](https://mirror.enha.kr/wiki/%EC%98%B4%EB%8B%88%EC%95%84%20II)를 눌러버렸다. [옴니아 II](https://mirror.enha.kr/wiki/%EC%98%B4%EB%8B%88%EC%95%84%20II)의 CPU 클럭이 800 MHz로 당시 고클럭 제품인 것을 감안해도 체감 퍼포먼스의 차이는 [운영체제](https://mirror.enha.kr/wiki/%EC%9A%B4%EC%98%81%EC%B2%B4%EC%A0%9C)를 차이를 감안해도 극명하게 보여준 사례가 되었다.

이후 [안드로이드](https://mirror.enha.kr/wiki/%EC%95%88%EB%93%9C%EB%A1%9C%EC%9D%B4%EB%93%9C%28OS%29) 탑재 [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0)들이 시장에 활발히 출시되면서 Cortex-A8을 사용한 AP들도 덩달아서 활발히 출시되고, 동세대 비교대상이었던 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) S1 및 S2의 Scorpion 아키텍쳐보다 클럭대비 성능이 뛰어나다는 평가를 받는다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=8)]

[**1.1.3.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.3.1)



[[JPG external image]](http://noel.feld.cvut.cz/vyu/scs/prezentace2007/ARM-Cortex-A8/images/mikrostrukturaObr.jpg)

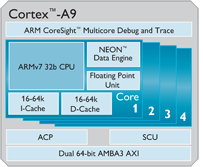
* 한 사이클에 2개의 명령어 디코더
* issue명령 분배기에서 3개의 연산모듈로 파견

3개의 연산 모듈

A - ALU/MUL : 정수 연산 모듈, 덧/뺄셈 및 곱/나눗셈 지원  
B - ALU  
C - LS - 로드/스토어 모듈

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=9)]

[**1.1.4.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A9**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.4)



[[PNG external image]](http://www.arm.com/images/Cortex_A9.png)

2007년 3월에 발표된 Cortex-A8의 후속작으로, Cortex-A8보다 많은 개선점이 존재한다.

첫번째로 멀티코어를 지원한다. 최대 쿼드코어까지 구성이 가능해졌다. 두번째로 NEON SMID의 강화로 Cortex-A8에서 첫번쨰로 사용된 NEON코어를 직접 연산 유닛에 내장화했다. 세번째는 [비순차적 명령어 처리](http://ko.wikipedia.org/wiki/%EB%B9%84%EC%88%9C%EC%B0%A8%EC%A0%81_%EB%AA%85%EB%A0%B9%EC%96%B4_%EC%B2%98%EB%A6%AC)를 도입했다. 이 외에 L2 Cache의 도입같은 변화점이 있다.

Cortex-A8와 마찬가지로 본격적인 시장 투입은 느렸다. 2011년 1월에 공개된 [nVIDIA Tegra](https://mirror.enha.kr/wiki/nVIDIA%20Tegra" \o "nVIDIA Tegra) 2가 첫 제품이었다.[[4]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn4)

더군다나 Cortex-A9의 멀티코어 구성으로 2010년부터 본격화된 [안드로이드](https://mirror.enha.kr/wiki/%EC%95%88%EB%93%9C%EB%A1%9C%EC%9D%B4%EB%93%9C%28OS%29) [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0) 열풍은 정점을 찍게된다. 특히 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)의 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 4210을 사용한 [갤럭시 S II](https://mirror.enha.kr/wiki/%EA%B0%A4%EB%9F%AD%EC%8B%9C%20S%20II)의 성공으로 더욱 가세되었기 때문이다. 이 뿐만이 아니라 Cortex-A8보다 진보한 아키텍처 구조로 클럭당 성능비도 증가했으며 멀티코어화로 인한 성능상승폭은 거의 코어수에 비례하기 때문에 단순이 코어 개선의 효과보다 큰 성능 향상이 있었다.

이러한 Cortex-A9 멀티코어에 대응하기 위해 퀄컴은 Scorpion 아키텍쳐를 멀티코어화 시키고 고클럭화를 목표로 파이프라인을 늘려 1.5 GHz의 클럭을 달성하지만 태생적으로 클럭대비 성능이 Cortex-A9보다 밀리기 때문에 성능은 성능대로 놓치고 발열은 발열대로 놓치게되었다. 이후 Krait 아키텍쳐가 나오지 전까지 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4)의 이미지가 하락하게된 원흉이 되었다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=10)]

[**1.1.4.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.4.1)

* 8~11-pipeline
* 한 사이클에 2개의 명령어 디코더 (DE) - 64k
* 명령어 발행 어레이 개수 = 1
* ISSUE명령 분배기 1개가 3+1라인(in) 한번에 파견은 최대 3개
* 외장 L2 Cache

연산 모듈은 총 4개

A1 - ALU(out) : 정수 연산 모듈, 덧/뺄셈 지원  
A2 - ALU(out추가) : 정수 연산 모듈, 덧/뺄셈+곱셈 지원  
B - Load/Store - AGU  
C - NEONvfp (in)

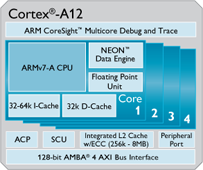
Cortex-A8의 순차적 명령어 처리방법에서 진보한 비 순차적 명령어 처리를 지원한다. 다만 위의 파이프라인 개요도에서 볼 수 있듯이 파란색의 비 순차적 처리를 지원하는 모듈은 Issue명령어 분배기와 ALU연산유닛 뿐이다. 내장화 된 Neon유닛과 AGU자체는 순차적으로 명령어를 처리한다.

이전 Cortex-A과 ALU유닛의 개수는 같지만 두번째 ALU유닛에서 정수 곱셈을 지원이 추가 되었다. 다만 이는 일종의 보조 유닛으로, Issue 분배기에서 한번에 보낼 수 있는 파견량은 3개로써 한번에 모든 연산모듈에 명령을 보낼수는 없다.

이를 만회하기 위해서 비교적 대용량의 L2 Cache를 코어에 외장으로 지원하며 [운영체제](https://mirror.enha.kr/wiki/%EC%9A%B4%EC%98%81%EC%B2%B4%EC%A0%9C)에서 지원이 필요하다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=11)]

[**1.1.5.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A12**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.5)



[[PNG external image]](http://www.arm.com/images/Cortex-A12_203px.png)

2013년 6월 2일에 발표된 ARMv7 기반 아키텍쳐.

전반적으로 Cortex-A9을 Cortex-A15의 설계를 혼용하여 전성비를 높이는데 주력한 아키텍쳐이다.

A15/A7==>A57/A53이라는 주력 라인업과는 다르게 좀 갑툭튀한 느낌이 있는데 사실 32비트 기반인 이 아키텍처가 제품화될 시기인 2015년에는 64비트 메인스트림인 A57/A53기반 제품들이 출시될 시기와 겹치게 될 상황이기 때문.

그런데 A12가 적용되는 타겟 공정은 2015년 시점에서는 저가양산라인으로 포지션하게 될 GF와 TSMC의 28nm가 될 예정이고, 반면 A57/A53기반의 하이엔드 제품들은 20/16/14nm공정 라인에서의 생산을 목표로 하고 있다. 이 관점에서 2015년 시점에서 생산단가가 떨어질대로 떨어지게 될 28nm 공정을 활용할 만한 아키텍쳐로는 기존에 개발된 A15는 해당 공정에서 전력소모 특성이 좋다고 말하기 힘들어 사용이 제한되고, 그것보다도 복잡한 A57은 더더욱 28nm수준에서는 부적합하다고 추정할 수 있는 상황. 그럴 경우 28nm 공정에 적용할 만한 제품은 저성능의 A7/A9/A53로 한정되면서 중간급 라인업이 비어버리는 문제가 생긴다. 즉 원래는 시기적으로 A15가 미들레인지로 내려오면서 채워넣었어야 하는 부분을 A12가 대신 채워넣어야 하는 상황이 된 것.

Cortex-A9와 비교했을때 명령 디코더가 3개로 늘어났고, Cortex-A9가 정수 연산 부분에서 부분적으로 비 순차적 명령어 처리를 한 것에 반해서 Cortex-A12는 모든 연산 유닛이 비 순차적 처리를 지원한다. 또한 외장으로 Cortex-A53의 분기 예측기(Branch Predictor)를 도입하고 ALU모듈에서 나눗셈을 지원, L2 Cache의 내장화 같은 개선점이 있다.

또한 미드레인지 라인업이지만, Cortex-A15와 같이 Cortex-A7 아키텍쳐와 조합해서 big.LITTLE을 지원한다고 한다.

전반적인 성능은 퀄컴 Krait와 애플의 Swift와 비슷할 것으로 예측되고 있다. 구조적으로는 Cortex-A15의 요소들을 최소한으로 잘라내어 Cortex-A9에 적용한 것으로 추측된다.

014년 10월 1일 [ARM 공식 홈페이지](http://community.arm.com/groups/processors/blog/2014/09/30/arm-cortex-a17-cortex-a12-processor-update)에서 Cortex-A12가 Cortex-A17로 통합되었다고 발표했다

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=12)]

[**1.1.5.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.5.1)

* 10~12-pipeline
* 한 사이클에 2개의 명령어 디코더(DE) - 128k
* 명령어 발행 어레이(Rename & Dispatch) = 2
* ISSUE명령 분배기 3개가[[5]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn5) 각 2라인씩 총6라인 파견6개
* 내장 L2 Cache

연산 모듈은 총 6개

A1- ALU(out) : 정수 연산 모듈, 덧/뺄/곱셈 + 나눗셈 지원  
A2- ALU(out) : 정수 연산 모듈, 덧/뺄/곱셈 + 나눗셈 지원

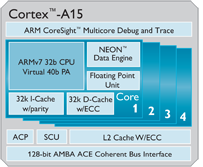
B1- load/store(out) - AGU  
B2- load/store(out)

C1-NEONvfp (out)  
C2-NEONvfp (out)

파란색이 비 순차적 명령어 처리 부분으로 ALU모듈에서만 비 순차적으로 명령어를 처리했던 Cortex-A9에 비해서 모든 유닛에서 지원하게 되었다. 또한 Cortex-A15와 마찬가지로 명령어 발행 어레이와 Issue분배기의 분리구조를 체택. 하나의 발행 어레이는 Neon/vfp SMID를 담당하며, 나머지 하나의 발행기가 ALU모듈과 AGU모듈을 통합해서 관리한다. 다만 Issue분배기 자체는 3개로써 각각 2개의 연산 모듈을 담당하며 한번에 최대 6개의 명령어를 발행할 수 있다. 이는 Cortex-A9에서의 Issue분배기가 한번에 최대 3개 까지 밖에 명령어를 발행하지 못했기 때문에 모든 연산 유닛을 활용할 수 없었던 것에 대한 개선점이다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=13)]

[**1.1.6.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A15**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.6)



[[PNG external image]](http://www.arm.com/images/Cortex_A15.png)

2010년 9월에 발표된 Cortex-A9의 후속작이다. 발표 시기를 보면 Cortex-A7과 Cortex-A12보다도 빠르지만 성능은 ARMv7기반의 모든 아키텍쳐 중에서 가장 높다.  사실, 이후에 발표된 Cortex-A7과 Cortex-A12는 Cortex-A15를 저전력화한 아키텍쳐라 보면 된다.

전작인 Cortex-A9보다 향상된 ALU유닛과 배수만을 따로 처리하는 ALU를 추가했고, 분기 예측기연산 모듈의 내장화를 실시했다. 또한 모든 유닛에 비 순차 처리방식 도입을 하는 등 ARMv7 기반의 아키텍쳐가 보여줄 수 있는 최대한의 성능을 보여준다. 때문에 처리능력에서는 코어 초기형 수준의 X86기반의 아키텍쳐들과도 견줄 수 있는 수준이다.

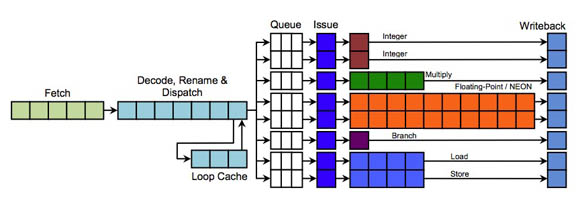
문제는, ARM이 추구하던 **저전력 코어로서의 이점을 상당히 잃었다**는 것인데...

부 구조상 성능을 올릴 수 있는 모든 것을 때려넣은 구조이기 때문에 이전의 ARM이 출시한 그 어떤 CPU 아키텍쳐보다 **자비없는 발열**과 **이성을 잃은 전력 소모율**을 보여준다. 때문에 크레이트같은 모바일 지향으로 설계된 아키텍쳐에 비해 발열문제를 처리하기 어려운 아키텍쳐가 되어버렸다. 32nm공정에서는 듀얼코어의 엑시노스 5250 정도가 태블릿 전용으로 탑재되었고, 2013년에 28nm공정을 사용한 5410에 와서야 스마트폰에 탑재하는데 성공. 다만 전력소모는 경쟁제품 대비 압도적인 모습을 보여주지는 못하고 있다.

다만 이러한 상황은 ARM사가 A15의 포지션을 모바일뿐만 아니라 서버 영역까지 확장하면서 벌어진 것이며 ARM은 이 문제를 해결하기 위해서 모바일 영역에서는 big.LITTLE 방식을 도입하게 되었다. 다만 big.LITTLE방식은 소프트웨어적인 부분에도 많이 의존하게 되는데 그에 대한 개발완료는 2013년 말이 될 예정.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=14)]

[**1.1.6.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.6.1)



[[JPG external image]](http://regmedia.co.uk/2011/10/20/arm_a15_pipeline_small.jpg)

* 정수-15 / 부동 소수점- 17~25 pipeline
* 한 사이클에 3개의 명령어 디코더 - 128k
* 명령어 발행 어레이 개수 = 5
* Issue명령 분배기 5개가 각 1~2개씩 총 8라인
* 내장 L2 Cache 통합, 분배식 입력

연산 모듈은 총 8개 (A~E)

A1 - 정수연산(덧/뺄셈)(out)  
A2 - 정수연산(덧/뺄셈)(out)

B - 정수 연산(곱/나눗샘)(out)

C1 - NEONvfp (out)  
C2 - NEONvfp (out)

D - branch(분기 예측)

E1 - load/store  
E2 - load/store

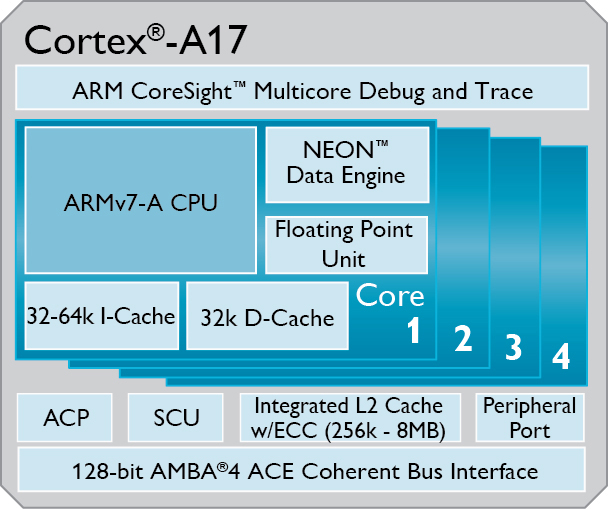
확실하게 기존 ARM의 CPU 아키텍쳐보다 파이프라인이 대폭 복잡해진 것을 알 수 있다. Cortex-A9 대비 명령어 디코더의 개수가 1개 더 늘어나서 3개를 가지고 있으며 각 연산 유닛의 파트마다 각각의 독립적인 발행 어레이와 명령 분배기를 가지는 구조로써 Cortex-A9 대비 2배로 연산 유닛이 증가하고 이 것을 효율적으로 이용하도록 되어있다. Cortex-A9가 최대 발행능력이 떨어져서 모든 모듈을 한번에 동작할 수 없었지만 Cortex-A15는 한번에 8개 유닛을 모두 사용할 수 있다고 한다.

기본적인 ALU의 성능이 강화되었고 곱셈과 나눗셈 전용의 ALU가 추가되었다. 또한 모든 연산 유닛의 비 순차적 처리지원과 분기 예측기(Branch Predictor)를 내장하고 있다.

L2 Cache를 ARMv7 명령어셋 기반 Cortex-A 시리즈 중 최초로 아키텍쳐 안에 내장하였으며 통합된 하나의 거대 L2 Cache는 각 코어로부터 직접적으로 리소스를 주고 받을 수 있다. 이는 Cortex-A9에서 아키텍쳐와 외장 L2 Cache가 컨트롤러를 통해 한단계 연결되었던 설계에 비해서 개선된 구조이다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=15)]

[**1.1.7.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A17**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.7)



[[JPG external image]](http://www.legitreviews.com/wp-content/uploads/2014/02/Cortex_A17.jpg)

2014년 2월에 발표된 마지막 ARMv7기반의 프로세서.   
네이밍 라인업상 Cortex-A15의 상위 라인업 이며, Cortex-A7 아키텍쳐와 조합해서 big.LITTLE을 지원한다고 한다. 전체적인 성능은 Cortex A9보다 60%향상.

현재 64bit 지원의 ARMv8로 넘어갈 시장의 상황에서 보자면, 미래를 위한 라인업으로써. 2014년 기준 하이엔드 성능이지만, 본격적으로 시장에 탑제 기기가 나올 2015년 기준으로 미들레인지의 성능을 내줄 것으로 예상하고 있다고 한다.

이는 ARMv8기반의 A57 프로세서가 분명 뛰어난 성능은 맞지만, Cortex A15의 전례처럼 초기 양산이 매우 힘들다면[[6]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn6), 중,소 SOC제작사들은 기존의 ARMv7을 사용할 수 밖에 없고[[7]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn7), 하이엔드 시장의 ARMv8과 비교한다면, 기존 A15로는 부족하다는 판단이 들었다고 보인다. 앞전의 Cortex A15의 여러가지[[8]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88" \l "fn8" \o "라인업상이던 제품상이던)문제가 Cortex A12의 출시로 어느정도 해소 되었다고 본다면, Cortex A17은 Cortex A57 세대를 준비하는 지금의 Cortex A12의 위치의 라인업이라고 볼 수 있다.

시장에 첫 출시되는 제품은 미디어텍의 MT6595로 Cortex-A7과 [ARM big.LITTLE 솔루션](https://mirror.enha.kr/wiki/ARM%20big.LITTLE%20%EC%86%94%EB%A3%A8%EC%85%98)으로 구성했다. 빠르면 2014년 2분기부터 시장에 투입될 예정이라고 한다.

2014년 10월 1일 [ARM 홈페이지](http://community.arm.com/groups/processors/blog/2014/09/30/arm-cortex-a17-cortex-a12-processor-update)에서 공식적으로 Cortex-A12를 Cortex-A17 라인으로 편입시켰다

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=16)]

[**1.1.7.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.1.7.1)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=17)]

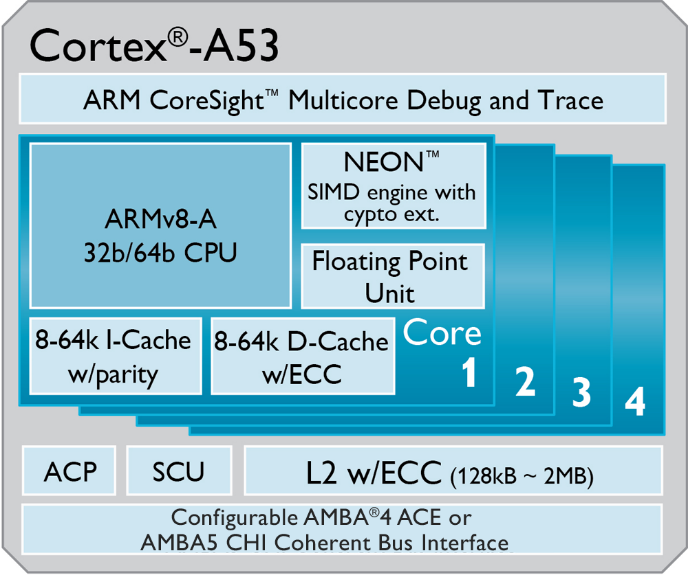
[**1.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**ARMv8**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.2)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=18)]

[**1.2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A50 시리즈**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.2.1)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=19)]

[**1.2.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A53**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.2.1.1)

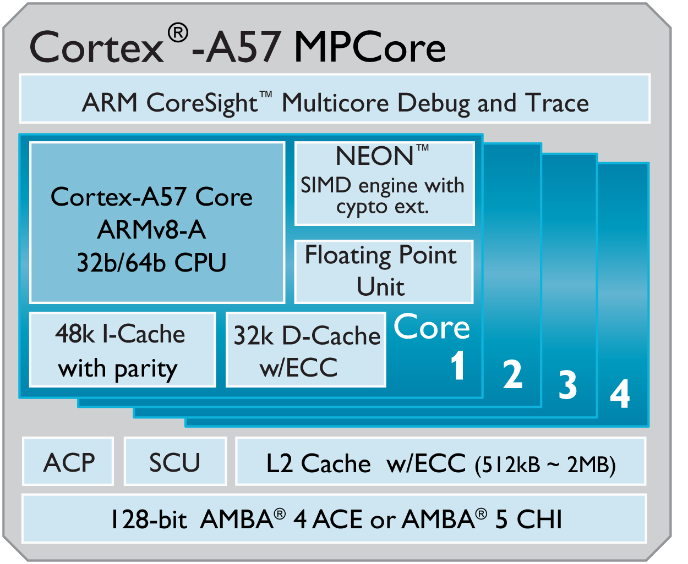


[[PNG external image]](http://www.arm.com/images/Cortex-A53-large.png)

ARMv8 명령어셋 기반의 CPU로 Cortex-A57과 [ARM big.LITTLE 솔루션](https://mirror.enha.kr/wiki/ARM%20big.LITTLE%20%EC%86%94%EB%A3%A8%EC%85%98)으로 구성이 가능하다. 대표적으로 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 808 MSM8992, [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 810 MSM8994, 삼성 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 7 Octa 5433 등이 있다. 또한, 자체적인 성능역시 중급형 CPU로 충분하기에 중급형 AP에 탑재되는 경우도 있다. 대표적으로 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 410 MSM8916, [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 615 MSM8939, [미디어텍](https://mirror.enha.kr/wiki/%EB%AF%B8%EB%94%94%EC%96%B4%ED%85%8D) MT6795 등이 있다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=20)]

[**1.2.1.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cortex-A57**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.2.1.2)



[[PNG external image]](http://www.arm.com/images/Cortex_A57_large.png)

ARMv8 명령어셋 기반의 CPU로 Cortex-A53과 [ARM big.LITTLE 솔루션](https://mirror.enha.kr/wiki/ARM%20big.LITTLE%20%EC%86%94%EB%A3%A8%EC%85%98)으로 구성이 가능하다. 대표적으로 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 808 MSM8992, [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 810 MSM8994, 삼성 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 7 Octa 5433 등이 있다.

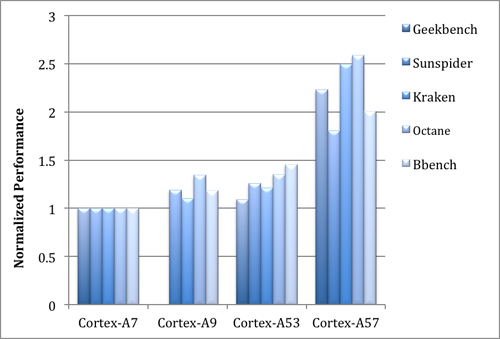
[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=21)]

[**1.2.1.3.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**공통사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.2.1.3)

* 64비트 지원(AArch64)
* 32비트 ARMv7 명령어와 100% 하위호환
* TrustZone® 보안 기술
* NEON™ Advanced SIMD
* DSP & SIMD 확장
* VFPv4 부동소숫점 연산
* 하드웨어 가상화 지원

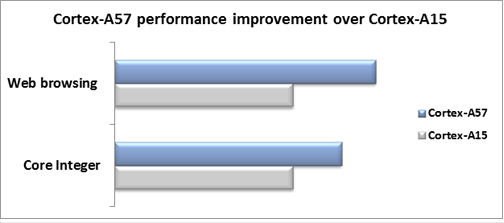
[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=22)]

[**1.2.1.4.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**성능**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-1.2.1.4)



[[PNG external image]](http://www.arm.com/images/Cortex-A53-performance-chart.png)

Cortex-A53는 전체적으로 Cortex-A9와 비슷하거나 향상된 성능을 보이며 Cortex-A57는 Cortex-A53의 약 1.5배에서 2배의 성능을 보인다.



[[PNG external image]](http://www.arm.com/images/A57performance_webpage.png)

Cortex-A57 프로세서와 Cortex-A15 프로세서의 비교 그래프.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=23)]

[**2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**커스텀 아키텍쳐**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2)

[ARM](https://mirror.enha.kr/wiki/ARM%28CPU%29)에서 직접 설계한 아키텍쳐가 아닌 명령어 셋을 직접 이용해서 제작한 일종의 커스텀 아키텍쳐 목록이다. ARM에게 IP라이센스를 취득해서 제작된다.

구조 상으로 Cortex-A 시리즈와 유사하나, 각 아키텍쳐 설계사가 필요에 따라서 취사선택해 개발했다는 것이 특징이다.[[9]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn9)

관련 커뮤니티에서 커스텀 아키텍쳐의 본류를 Cortex-A 시리즈에서 찾으려는 경향이 있다. 다만, 엄밀히 말해서 명령어 셋을 직접 이용해서 만드는 것이기 때문에 Cortex-A 시리즈에서 본류를 찾는 것은 적절하지 않다. 때문에 적절히 '비슷한 성능의 Cortex-A 시리즈가 이 것'이라는 사실만 알고 있는 정도로 생각해야 한다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=24)]

[**2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**ARMv7 기반**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=25)]

[**2.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**퀄컴**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.1)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=26)]

[**2.1.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Scorpion**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.1.1)

퀄컴이 자사 AP 브랜드인 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) S1부터 S3까지 사용한 커스텀 아키텍쳐.

본가 ARM 라인업에 대응하는 아키텍쳐로는 Cortex-A8과 Cortex-A9로 볼 수 있다. 2009년 말, 첫 시제품이 [HTC](https://mirror.enha.kr/wiki/HTC)의 [HD2](https://mirror.enha.kr/wiki/HD2)에 탑재, 주목을 받았다. ARM기반 CPU로는 최초로 **1 GHz에 도달**했기 때문이다. 특히, 당시 경쟁 CPU 아키텍쳐는 ARMv6 기반의 ARM11 아키텍쳐였기 때문에 더욱 부각되었다.[[10]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn10)

특히, 국내에서는 이 시기에 [옴니아 II](https://mirror.enha.kr/wiki/%EC%98%B4%EB%8B%88%EC%95%84%20II)와 [아이폰 3GS](https://mirror.enha.kr/wiki/%EC%95%84%EC%9D%B4%ED%8F%B0%203GS)의  양강구도를 가지고 있었다. 이와중에 국내의 소수 PPC[[11]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn11)유저들은 [HD2](https://mirror.enha.kr/wiki/HD2)를 주목했다. 1 GHz라는 당시 최고클럭의 CPU와 [Windows Mobile](https://mirror.enha.kr/wiki/Windows%20Mobile)을 가지고도고 충분한 퍼포먼스를 뿜어내는 [HTC](https://mirror.enha.kr/wiki/HTC)의 최적화 능력[[12]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn12)과 화려한 [hTC Sense](https://mirror.enha.kr/wiki/hTC%20Sense" \o "hTC Sense)는 [Windows Mobile](https://mirror.enha.kr/wiki/Windows%20Mobile) [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0)이라 볼 수 없는 퍼포먼스를 보여줬고, [옴니아 II](https://mirror.enha.kr/wiki/%EC%98%B4%EB%8B%88%EC%95%84%20II)와 대조되면서 초기 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4)의 이미지를 좋은 쪽으로 부각시켰다.

다만, 2010년 이후로 오픈 소스인 [안드로이드](https://mirror.enha.kr/wiki/%EC%95%88%EB%93%9C%EB%A1%9C%EC%9D%B4%EB%93%9C%28OS%29)를 사용한 [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0)이 주로 출시되었고, 시장의 파이가 커짐에 따라서 [ARM](https://mirror.enha.kr/wiki/ARM)에게 라이센스를 취득한 AP 개발업체가 증가, [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4)은 초기의 모습과는 달리 하락세를 걷기 시작했다.

Cortex-A8을 사용한 TI의 [OMAP](https://mirror.enha.kr/wiki/OMAP)과 [ARM](https://mirror.enha.kr/wiki/ARM)의 권고를 무시하고 코어 커스텀까지 강행해서 1 GHz의 클럭을 돌파[[13]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn13)한 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)의 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 3110같은 경쟁 AP에게 성능적으로 밀리기 시작했다. 이후, [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) S2에는 Scorpion 아키텍쳐를 일부 개선하고 CPU 클럭을 1.4 GHz까지로 상승시켰지만, Cortex-A8과의 경쟁에서는 완전히 패배해버린다.[[14]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn14)[[15]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn15)

2011년에는 이러한 현상이 정점을 찍어버렸다. Cortex-A8의 후속 아키텍쳐인 Cortex-A9 기반의 AP가 속속 출시. 문제는 Cortex-A9가 멀티코어를 지원하기 시작하면서 TI [OMAP](https://mirror.enha.kr/wiki/OMAP) 4 시리즈와 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)의 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 4210, 가 전부 Cortex-A9기반의 듀얼코어를 사용하기 시작했고. 퀄컴에서는 이에 대항하기 위해 Scorpion을 소폭 개량해 멀티코어를 지원하게 만들고, L2 Cache의 용량을 증가시켜 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) S3에 탑재했다. 파이프라인의 수를 늘려서 CPU 클럭도 1.5 GHz로 기존[[16]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn16)대비 약 50%정도 상승시켰다. 다만, IPC 상의 개선이나 공정 미세화가 동반되지 않는 상황에서 올리다보니  **발열의 스냅드레곤**, **스냅드레기**같은 명성을 확고히 하는데 일조하였다.

특히, 2011년 말에 우후죽순으로 출시된 1세대 [LTE](https://mirror.enha.kr/wiki/LTE) 지원 [스마트폰](https://mirror.enha.kr/wiki/%EC%8A%A4%EB%A7%88%ED%8A%B8%ED%8F%B0)의 경우, 당시 [LTE](https://mirror.enha.kr/wiki/LTE)를 지원하면서 **3G 음성통화**까지 지원하는 AP가 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 뿐[[17]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn17)인 상황에서 내부에 칩셋이 많아지니 기기가 커짐에 따라 낮은 해상도로는 경쟁력이 없다고 판단한 스마트폰 제조사들이 기존 WVGA(480 x 800), FWVGA(480 x 854)보다 고 해상도인 HD 720p(1280 x 720)을 채택하면서 **엄청난 발열의 시너지 효과**를 보여주었다.[[18]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn18) 때문에 Scorpion 아키텍쳐는 모바일 AP 시장에서 아키텍쳐의 개량에 뒤처지면 시장에서 어떠한 평가를 받는지를 생생히 보여주는 본보기가 되었다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=27)]

[**2.1.1.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.1.1.1)

* 10~12 pipeline
* 한 사이클에 2개의 명령어 디코더
* Issue 분배기가 3라인 파견
* 부분 비 순차 처리 지원
* Neon유닛이 연산 모듈에 포함
* 멀티코어 구성가능

Scorpion의 자세한 구조는 잘 알려지지 않았다. 다만 알려져 있는 정보상으로는  상단히 진보한 구조의 아키텍처로써 ARMv7기반의 첫 아키텍쳐인 Cortex-A8과 비교시에도 상당한 우위점이 많다.

Cortex-A8에서는 Neon/vfp SMID모듈을 외부에 옵션으로 부착하는 형태였는데, Scorpion에서는 처음부터 연산유닛화 해서 파이프라인에 내장화 했다. 연산모듈에 알맞게 명령어를 분배하는 Issue 분배기도 비 순차적 처리를 지원하며, 상당히 초기 모델부터소량의[[19]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88" \l "fn19" \o "S1부터 인지는 확실하지 않다.) L2 Cache를 내장했다. 그리고 처음부터 멀티코어화를 고려해서 개발했기 때문인지 아키텍처를 크게 변경 할 필요 없이 듀얼코어까지 지원한다. 이 모든 것이 Cortex-A 시리즈에서는 Cortex-A9부터 지원된 부분이다.

이 외에 Cortex-A9보다 나은 점도 존재한다. Cortex-A9의 Neon유닛은 64 KB기반인 반면, 이쪽은 128 KB기반이다. 이 때문에 멀티미디어 기능에 많이 사용되는 부동소수점 연산에서 상당한 강점으로 보인다. 다만 이것이 시장에서 실제 강점으로 크게 활용되지는 못했는데, SoC의 발전에 따라 AP다이 안에 추가로 GPU코어를 탑제하는 것이 일반화 되면서 CPU만 쓰기 보다는 GPU가속을 활용하는 것이 성능 향상이 컷기 때문이다.

결론적으로 매우 괜찮은 아키택쳐 였으나, 시장의 변화와 경쟁에서 밀리고 결국 오명을 다 뒤집어쓴 매우 대표적인 사례가 되었다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=28)]

[**2.1.1.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Krait**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.1.2)

퀄컴이 [스냅드래곤 S4](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4#s-3.4)부터 사용한 CPU 아키텍쳐. 이후 라인업이 S 라인업에서 X00 라인업으로 개편되면서 CPU 아키텍쳐도 기존 [스냅드래곤 S4](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4#s-3.4)에서 사용된 아키텍쳐는 Krait 200, [스냅드래곤 600](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4#s-4.3)이 사용하는 아키텍쳐는 Krait 300, [스냅드래곤 800](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4#s-4.4)이 사용하는 아키텍쳐는 Krait 400으로 개선되었다.

수의 의미는 순정 ARMv7 명령어 셋 기반의 아키텍쳐보다 리비전 된 정도를 의미한다. 200에서 300은 클럭당 10%에서 최대 30% 정도의 성능 향상이 있었으며 300에서 400의 경우에는 최대 클럭이 20% 더 상승되어서 모바일 AP 최초로 CPU 클럭이 2 GHz를 돌파하게 되었다.[[20]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn20)

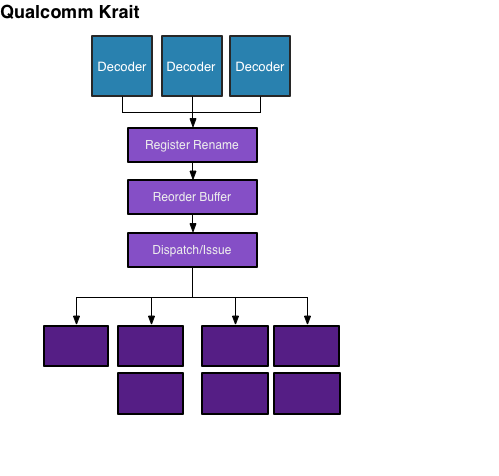
이전에는 '[ARM Cortex-A9](https://mirror.enha.kr/wiki/ARM%20Cortex-A9) 베이스 + [ARM Cortex-A15](https://mirror.enha.kr/wiki/ARM%20Cortex-A15) 명령어 추가'의 방식으로 만들어졌다고 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) 항목에 서술 되어 있었는데, 이건 반만 맞고 반은 틀린 설명이다. 굳이 꼭 집어서 [ARM Cortex-A9](https://mirror.enha.kr/wiki/ARM%20Cortex-A9)를 커스텀한거다, [ARM Cortex-A15](https://mirror.enha.kr/wiki/ARM%20Cortex-A15)를 커스텀한거다 말할 수는 없는데, [Qualcomm Scorpion](https://mirror.enha.kr/wiki/Qualcomm%20Scorpion)부터 시작된 퀄컴의 자체적인 주력 [CPU](https://mirror.enha.kr/wiki/CPU)는 죄다 아키텍처 라이센스를 딴 다음 ARMv7 명령어 세트 기반으로 커스텀한 아키텍쳐다. 또한 Krait와 [ARM Cortex-A15](https://mirror.enha.kr/wiki/ARM%20Cortex-A15)는 ARM 기반 프로세서의 발전 방향이 비슷하기 때문에 대략적인 구조가 비슷할 뿐이지 여러모로 차이가 많다.

굳이 파이프라인 레벨에서 비교해본다면 Krait는 A15보다는 A9에 비슷한 구조이다. A9대비 디코더의 개수와 연산 모듈은 늘었지만, Issue 분배기의 한계로 늘어난 연산 모듈을 한번에 활용하는것이 불가하며,이것도 A9에서 지적되었던 문제이다. A15와 비슷한점은 캐쉬를 코어 내에 내장했다는 것 정도다.

[삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)나 [엔비디아](https://mirror.enha.kr/wiki/%EC%97%94%EB%B9%84%EB%94%94%EC%95%84) 등 경쟁사들이 ARMv8기반의 Cortex-A53과 Cortex-A57을 이용한 차세대 AP를 2014년 상반기를 목표로 개발하고 있는 상황에서 생산사인 [TSMC](https://mirror.enha.kr/wiki/TSMC)의 미세공정의 이원화가 늦어지는 상황[[21]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn21)에다 과거 Scorpion 아키텍쳐의 전철을 밟아 Krait 아키텍쳐를 개선하고 클럭을 상승시키는 쪽으로 갈 것이라는 예상이 나오고 있다.[[22]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn22) 다만, 퀄컴이 이전의 사례를 답습했다면 Krait의 후속 아키텍쳐로 승부를 볼 것이라는 이야기도 있다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=29)]

[**2.1.1.2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.1.2.1)



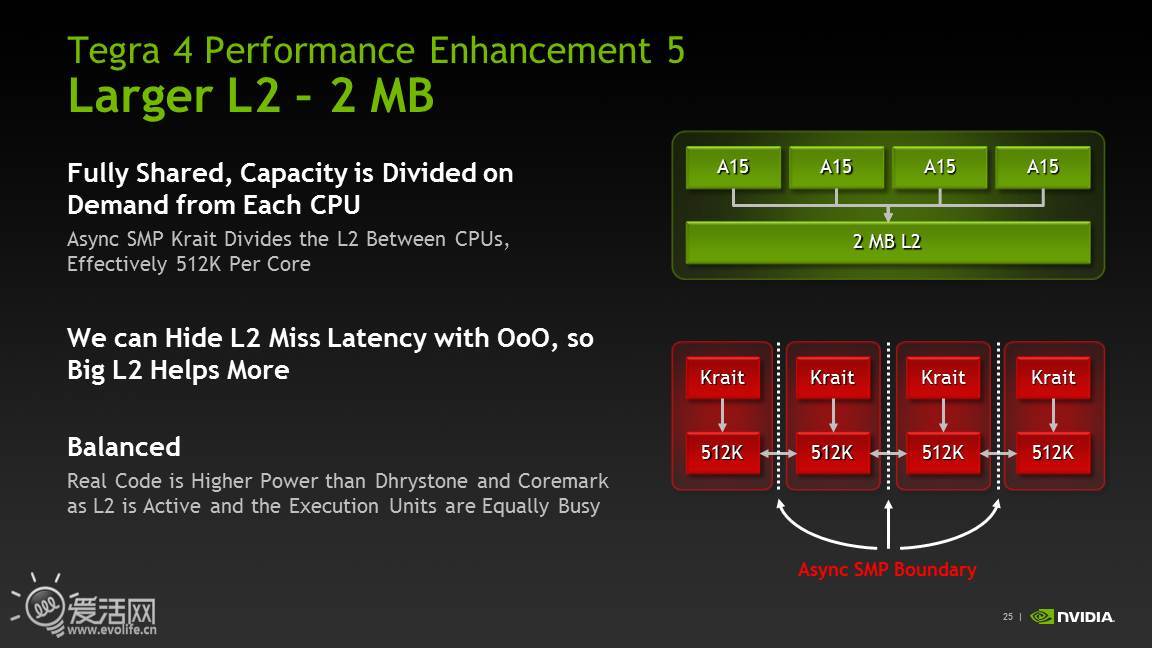
[[PNG external image]](http://i-cdn.phonearena.com/images/articles/71040-thumb/krait.png)

* 한 사이클의 3개의 명령어 디코더
* 명령어 발행 어레이 개수 = 1
* ISSUE명령 분배기 1개가 4라인 한번에 파견은 4개

연산 모듈은 총 7개

A1 - ?[[23]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn23)  
A2 - ?  
B1 - ?  
C1 - ?  
C2 - ?  
D1 - NEON/vfp (128k)  
D2 - NEON/vfp (128k)

동시대 경쟁 아키텍쳐인 Cortex-A9와 비교시에 명령어 디코더가 1개 늘었다. 다만 총 연산 모듈은 7개인 것에 반해 issue명령 분배기의 최대 파견량이 최대 4개로 Krait가 Cortex-A15보다 Cortex-A9에 가까운 아키텍쳐라는 것을 알 수 있다.[[24]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn24)



[[JPG external image]](http://gigglehd.com/zbxe/files/attach/images/251/989/604/010/719db68271dc505a6712d1ed70563ab2.jpg)

또한 Cortex-A9의 외장 L2 Cache에 비해서 Krait 아키텍쳐는 내장형의 L2 Cache를 가지고 있지만[[25]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn25), 각각의 코어가 비교적 소량의 L2 Cache를 가지고 있으며 하나의 대형 Cache를 가지고 있는 Cortex-A15나 Cortex-A12에 비해서 상황에 따른 효용성이 떨어진다. 즉 하나의 코어에 작업이 몰리더라도 한개의 코어는 제한적인 용량의 Cache만을 사용할 수 있고 이는 모든 코어가 하나의 대용량 Cache를 사용하는 방식에 비해서 불리하다. 이는 Krait 아키텍쳐 자체가 비동기식 구조를 가지고 있어서 각각의 코어가 별개로 운용되는것을 전제로 설계되었기에 나타나는 문제이다.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=30)]

[**2.1.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Apple**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.2)

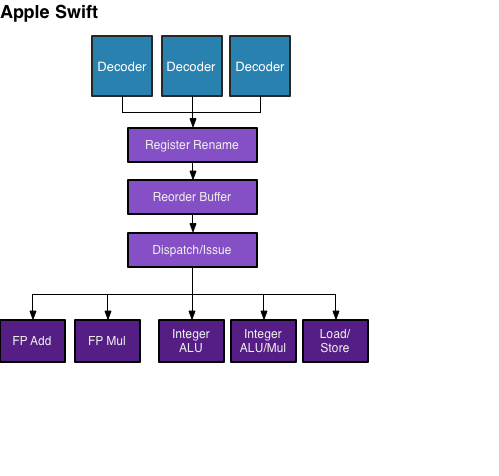
[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=31)]

[**2.1.2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Swift**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.2.1)

[애플](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%28%EA%B8%B0%EC%97%85%29)의 자체 AP인 [A6와 A6X](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%20A%20%EC%8B%9C%EB%A6%AC%EC%A6%88)에 사용된 ARMv7 기반의 커스텀 아키텍쳐. 자세한것은 항목 참조.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=32)]

[**2.1.2.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**사양**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.1.2.1.1)



[[PNG external image]](http://i-cdn.phonearena.com/images/articles/71041-thumb/swift.png)

* 한 사이클에 3개의 명령어 디코더
* 명령어 발행 어레이(dispatch) 개수 = 1
* ISSUE명령 분배기 1개가 5라인 한번에 파견은 5개

연산 모듈은 총 5개 (A~E)

A - ALU(out[[26]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#fn26))-정수연산 덧/뺄셈   
B - ALU(out)-정수연산 덧/뺄샘/곱/나눗셈  
C - NEON/vfp (?)  
D - NEON/vfp (?)  
E - load/store - AGU

A9대비 명령어 디코더의 개수와 연산 유닛의 수가 늘었다. 특히 Krait와 달리 A9때부터 문제시 되었던 Issue분배기의 최대 파견량이 총 연산 유닛의 수보다 적어서 효율이 떨어진다는 점이 해소되었다. 현재 전체적인 평가는 Krait와 비슷한 성능으로 귀결.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=33)]

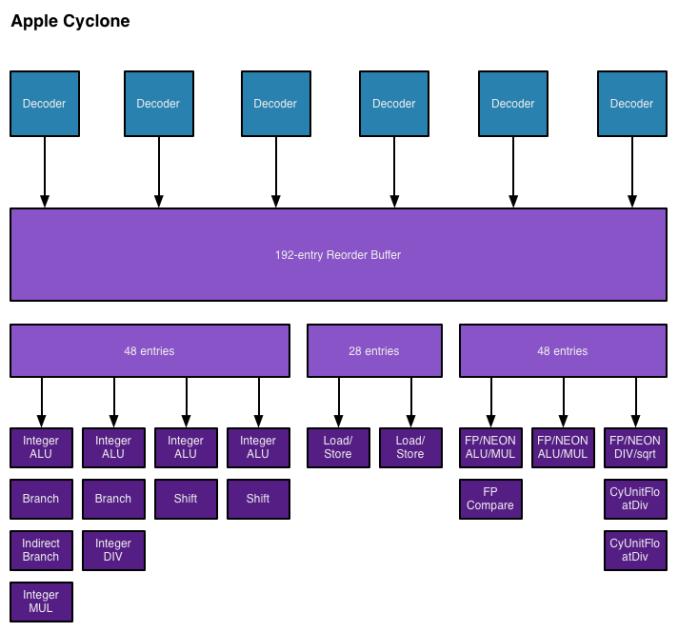
[**2.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**ARMv8 기반**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.2)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=34)]

[**2.2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Apple**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.2.1)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=35)]

[**2.2.1.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Cyclone**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.2.1.1)



[[PNG external image]](http://images.anandtech.com/doci/7910/Cyclone_575px.png)

[애플](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%28%EA%B8%B0%EC%97%85%29)의 자체 AP인 [A7과 A8, A8X](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%20A%20%EC%8B%9C%EB%A6%AC%EC%A6%88)에 사용된 ARMv8 기반의 커스텀 아키텍쳐. 자세한것은 항목 참조.

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=36)]

[**2.2.2.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**nVIDIA**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.2.2)

[[edit](http://rigvedawiki.net/r1/wiki.php/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88?action=edit&section=37)]

[**2.2.2.1.**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#toc)**Denver**[**¶**](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#s-2.2.2.1)

추가바람

----

* [[1]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn1) 다만 전력소모 절감은 마이크로 아키텍처 개선에 의한 것도 있지만 공정미세화에 의한 것도 있다.
* [[2]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn2) [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90) [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 5410의 LITTLE 코어와 [nVIDIA Tegra](https://mirror.enha.kr/wiki/nVIDIA%20Tegra" \o "nVIDIA Tegra) 3의 성능이 비슷한 결과가 있기도 했었다.
* [[3]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn3) [ARM](https://mirror.enha.kr/wiki/ARM%20Holdings)에서는 800 MHz이 한계라 보고 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)를 만류했었다.
* [[4]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn4) 다만, Cortex-A8이 시장에 본격적으로 사용되기 시작한 기간이 5년 이라는 것을 비교해 볼 때 시장 출시 주기가 점점 빨라지고 있다는 것을 알수 있다.
* [[5]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn5) 밑에서 각 A,B,C로 지칭
* [[6]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn6) A15의 대량 양산을 성공적으로 한 업체는 겨우 삼성전자와 엔비디아가 유일하다 둘 다 기존 반도체 시장에서 최대급 규모의 회사들이란 것을 보자면, 접근성으로의 Cortex A15는 매우 떨어진다.
* [[7]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn7) 라인업상 Cortex A15급의 성능이 필요했던 업체들은 여러가지 이유로 Cortex A15를 포기하고 Cortex A9 R4같은 기존 프로세서의 최신 리비전 버전을 사용하거나 Cortex A7을 쿼드코어 2모듈 구조로 8코어까지 묶는 등 어러가지 방법으로 해결하였다
* [[8]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn8) 라인업상이던 제품상이던
* [[9]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn9) 다만, 이는 일반적인 Cortex-A 시리즈를 사용하는 AP 설계사도 마찬가지다. 순수한 ARM 아키텍쳐를 사용하는 것이 아니다.
* [[10]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn10) 당시 Cortex-A8을 사용한 AP도 존재하기는 했었다. TI의 [OMAP](https://mirror.enha.kr/wiki/OMAP) 3 시리즈와 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)의 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 3110 초기 모델 뿐이었고 800 MHz내외의 클럭을 가지고 있었다.
* [[11]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn11) Pocket PC, 당시 일반인들에게는 생소한 WM기반의 PDA의 총칭한다.
* [[12]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn12) 이전부터 [HTC](https://mirror.enha.kr/wiki/HTC)는 [마이크로소프트](https://mirror.enha.kr/wiki/%EB%A7%88%EC%9D%B4%ED%81%AC%EB%A1%9C%EC%86%8C%ED%94%84%ED%8A%B8)의 허락과 협조를 받아 [Windows Mobile](https://mirror.enha.kr/wiki/Windows%20Mobile)을 자체적으로 빌드한 결과물을 탑재했다.
* [[13]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn13) 무식한 오버클럭은 아니다. IPC와 공정 개선도 동반된 클럭 상승은 전형적인 비메모리의 성능 개선 방법이다.
* [[14]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn14) 다만 당시의 기준의 성능 척도는 **게임**이 가장 큰 비중을 차지하고 있었고 당시 스냅드래곤에 들어간 Adreno GPU가 경쟁 AP에 들어간 imagination technologies의 PowerVR SGX5 시리즈에 비해서 성능과 최적화 모두가 부족했기 때문이였다.
* [[15]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn15) 이는 스마트폰 시장을 새로이 개척한 애플의 아이폰 시리즈의 AP가 전통적으로 PowerVR GPU를 사용하고 있었고, 초기 안드로이드에 대한 이해가 부족한 어플 개발자들이 모델링의 최적화 과정 없이 이식해 온 게임들의 대다수는 같은 PowerVR GPU를 사용하는 기기에서 원활이 돌아갔던것에 비해서 Adreno라는 자체 GPU를 사용했던 퀄컴의 AP에서는 원래 성능을 내기 어려웠다.
* [[16]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn16) 스마트폰에 들어가 출시된 모델
* [[17]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn17) **퀄컴의 AP만**[**LTE**](https://mirror.enha.kr/wiki/LTE)**를 지원하는 것이 아니다**. 현재 잘못 알려진 정보로, 2011년 당시 퀄컴에서조차 [LTE](https://mirror.enha.kr/wiki/LTE)를 지원하는 통신 모뎀은 **죄다 데이터 통신만 지원**하는 녀석들 뿐이었다. 거기에 이 통신 모뎀을 내장한 AP도 존재하지 않았다. 다만, 기본적으로 통신 모뎀이 내장되어 있지 않은 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) S3 APQ8060과 같이 사용할 경우, 원래 이론상 불가능한 [**GSM**](https://mirror.enha.kr/wiki/GSM)**과**[**WCDMA**](https://mirror.enha.kr/wiki/WCDMA)**음성통화가 지원**되었다. [CDMA](https://mirror.enha.kr/wiki/CDMA)의 경우 [CDMA](https://mirror.enha.kr/wiki/CDMA) 통신 모뎀이 내장된 [퀄컴 스냅드래곤](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4) S3 MSM8660과 같이 사용하여 통신을 따로 처리했다. 다만 약간의 예외로 CDMA+LTE조합의 네트워크에서 스냅드래곤을 사용하지 않는 퀄컴의 CDMA칩+삼성의 LTE칩 같은 조합이 소수 있었지만, 출시 기기들의 배터리 소비 문제가 심각하여 이후에는 볼 수 없게 되었다.
* [[18]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn18) 해상도가 높을수록 전력 소모율이 높아진다. 거기에 통신 모뎀과 AP가 분리된 상황이니 기존 통신 모뎀이 통합된 경우보다 전력 소모율이 높다. 하물며 이게 HD 720p를 지원하지만 감당하기 버거워한 [**퀄컴 스냅드래곤**](https://mirror.enha.kr/wiki/%ED%80%84%EC%BB%B4%20%EC%8A%A4%EB%83%85%EB%93%9C%EB%9E%98%EA%B3%A4)**S3**다. [더 이상의 자세한 설명은 생략한다](https://mirror.enha.kr/wiki/%EB%8D%94%20%EC%9D%B4%EC%83%81%EC%9D%98%20%EC%9E%90%EC%84%B8%ED%95%9C%20%EC%84%A4%EB%AA%85%EC%9D%80%20%EC%83%9D%EB%9E%B5%ED%95%9C%EB%8B%A4).
* [[19]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn19) S1부터 인지는 확실하지 않다.
* [[20]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn20) 여담으로, 최초로 2 GHz에 도달한 모바일 AP는 [삼성전자](https://mirror.enha.kr/wiki/%EC%82%BC%EC%84%B1%EC%A0%84%EC%9E%90)의 [엑시노스](https://mirror.enha.kr/wiki/%EC%97%91%EC%8B%9C%EB%85%B8%EC%8A%A4) 5250이다.
* [[21]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn21) 현재, 모바일 AP 파운더리 시장의 큰 손인 [애플](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%28%EA%B8%B0%EC%97%85%29)이 자사의 AP를 위해 [TSMC](https://mirror.enha.kr/wiki/TSMC)가 22nm 공정으로 전환하도록 자금 투입을 상당히 하고 있는 상황이다. 때문에 공정이 전환되어도 [애플](https://mirror.enha.kr/wiki/%EC%95%A0%ED%94%8C%28%EA%B8%B0%EC%97%85%29)에 몰아줄 가능성이 높다.
* [[22]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn22) 현재 지금까지 알려진 로드맵상에서는 이쪽이 유력하다.
* [[23]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn23) 총 연산 유닛의 수와 Neon의 경우는 비교적 명확하나 자세한 내용은 불명이다. 현재 ALU와 AGU가 있을 것으로 추측된다.
* [[24]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn24) Cortex-A15는 issue명령 분배기 자체가 5개, 한번의 파견량은 8개로써 8개의 연산 모듈을 모두 사용할 수 있다.
* [[25]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn25) 내장형 Cache가 효율이 더 좋다
* [[26]](https://mirror.enha.kr/wiki/ARM%20Cortex-A%20%EC%8B%9C%EB%A6%AC%EC%A6%88#rfn26) Out oder/비순차적 처리

﻿