**به نام زیباترین دوست**

# ARM architecture

معماری arm که پیشین با نام های advanced RISK Machine و Acron RISK شناخته میشد(مترجم: RISK یک گونه از معماری میباشد که در AVR هم بکار رفته است،معماری است که از لحاظ تعداد دستورات کاهش یافته است و در مقابل معماری CISK قرار دارد،در اینگونه معماری بیشتر بر روی قسمت برنامه نویسی کارها انجام میشود تا سخت افزار)و دارای معماری پردازنده 32 بیتی میباشد و توسط شرکت ARM Limited ساخته شده است و هم اکنون به صورت بسیار وسیع در وسایل تعبیه شده بکار میرود.به دلیل صرفه جویی در مصرف انرژی،پردازنده های ARM در وسایل همراه همچون PDA، cell phone و هر وسیله ای که هدف اصلی آن مصرف کمتر انرژی میباشد استفاده میشود.
امروزه خانواده ARM بیشتر از75 درصد سهم بازار پردازنده های 32 بیتی از نوع ریسک را گرفته اند و ARM به بزرگترین معماری 32 بیتی بدل شده است.امروزه ARM در بیشتر وسایل الکترونیکی از جمله وسایل قابل حمل گرفته(PDA،موبایل،وسایل پخش کننده موسیقی،کنسول های بازی دستی و ماشین حساب ها) تا وسایل جانبی رایانه(دیسک سخت،مسیریاب های رایانه) پیدا میشود; گرچه ARM هنوز به صورت جدی وارد پردازنده های رایانه های رومیزی یا ابر رایانه ها نشده است. دو شاخه مهم در این خانواده ،سری های بسیار مشهور XScale از شرکت Marvell و OMAP از شرکت Texas Instruments هستند.

## تاریخچه

طراحی ARM برای نخستین بار در سال 1983 در قالب یک پروژه ای تحقیقاتی در شرکت Acron Computer Ltd به منظور ساخت پردازنده های فشرده شده از نوع RISK و به سرپرستی Sophie Wilson و Steve Furber با هدف رسیدن به تاخیر ورودی و خروجی کم(interrupt) ساخته شد. و میخواستند چیزی شبیه به MOS Technology 6502 که در طراحی های Acron وجود داشت را بسازند.معماری دسترسی حافظه ی 6502 (the 6502's memory access architecture) به توسعه دهندگان امکان ساخت وسایلی سریع بدون نیاز به استفاده از سخت افزارهای گران قیمت برای دسترسی مستقیم به حافظه را می داد.اولین نمونه ساخت این تیم در آپریل سال 1985 به نام ARM1 ساخته شد و اولین محصول جدی شرکت به نام ARM2 در همان سال ساخته شد.

یک پردازنده Conexant ARM
که عمدتا در روترها بکار میرود

ویژگی های ARM2:

- یک دیتا باس 32 بیتی
- اندازه آدرس دهی 26 بیتی(MB64)
- 16 رجیستر 32 بیتی.
- کدهای برنامه نویسی در حافظه 64 مگابایتی قرار میگیرند
- شمارنده برنامه-program counter- به 26 بیت محدود است به این دلیل که 6 بیت از 32 بیت رجیستر به پرچم های وضعیت status flag -– از پیش رزو شده اند.

شاید بتوان گفت که ARM2 ساده ترین و کارآمدترین میکروپردازنده ای 32 بیتی در جهان است،با تنها 30,000 ترانزیستور (در مقایسه با موتورلا مدل 68000 با حدود 70,000 ترانزیستور).بیشتر این سادگی به دلیل نداشتن میکروکد(چیزی حدود 1/4 از 1/3 مدل 68000 میباشد) و مانند بیشتر پردازنده های روز در آن حافضه نهان -cache– استفاده نشده است(مترجم: داشتن کش قوی یک مزیت بزرگ میباشد به این دلیل که 90 درصد نیازهای پردازنده با مراجعه به کش خودش برطرف میشود و نیاز کمتری به خواندن از I/O با کلاک پایین، نیاز میشود!!!!!!).این سادگی باعث مصرف پایین انرژی این وسیله شده است،گرچه کارایی بالاتری نسبت به intel 80286 دارا میباشد.جانشین بعدی آن،سری ARM3 میباشد که با چهار کیلو بایت کش تولید شد و باعث ارتقای کارایی این سری شد.
بعدها در سال 1980 اپل و VLSI Tchnologhy شروع کار با شرکت Acron در زمینه تولید نسخه جدید هسته ARM کردند.این کار آنقدر برای شرکت Acron مهم بود که در سال 1990 تیم طراحی خود را در قالب یک شرکت جدید به نام Advanse RISK Machines Ltd در آورد. به همین دلیل گاهی وقت ها ARM با نام Advance RISK Machine بجای Acorn RISK Machine خوانده میشد.
بعد ها شرکت Advanced RISK Machines به شرکت ARM Ltd تبدیل شد و این تغییر نام هنگامی صورت گرفت که شرکت مادرش به نام ARM Holding plc در سال 1998 در بازار بورس لندن و NASDAQ به وجود آمد.
کار جدید شرکت اپل سرانجام به ARM6 به سرانجام رسید،نخستین نسخه اش در سال 1991 عرضه شد.اپل از ARM6 به عنوان پایه ARM610 در PDA های خودش که به نام Apple Newtone بود استفاده کرد.در سال 1994 شرکت Acron از ARM610 به عنوان پردازنده اصلی در رایانه های مخصوص خودش که برپایه RISK بودند استفاده کرد.DEC به معماری ARM6 مجوز استفاده از نامStrongARM را داد (به دلیلی که باعث برخی سردرگمی ها شده بود به این دلیل که این شرکت ها اغلب محصولشان را به اسم DEC Alpha تولید میکردند) . ای پردازنده ها در فرکانس کاری 233 مگاهرتز فقط یک وات انرژی مصرف میکنند(خیلی از نسخه های جدید مصرفشان خیلی کمتر از این مقدار شده است)بعدها اینتل از فرصت استفاده کرد و در اقدامی تدافعی خط تولید سری i960 خود را با StrongARM ترکیب کرد.بعد ها اینتل پیاده سازی کارامد خود را با نام XScale توسعه داد و در ادامه امتیاز این فناوری را به شرکت Marvell فروخت.
هسته ARM در طی این همه تغییرات هنوز سادگی خود را حفظ کرده بودند.ARM2 ها دارای 30000 ترانزیستور بودند در حالی که تعداد ترانزیستورهای ARM6 فقط به حدود 35000 رشد پیدا کردند.تجارت ARM بیشتر از طریق فروش هسته های IP صورت میگرفت و مجوز استفاده از این هسته ها را برای ساختن میکروکنترلر ها و پردازنده ها را به دیگران میداد.میتوان گفت موفقترین این پیاده سازی ها ARM7TDMI میباشد که صدها میلیون از آن در غالب بسیاری از تجزیهات میکروکنترل ها فروخته شد.نظر به اینست که

Original Design Manufacturer ها، هسته های ARM را با بسیاری از گزینه های اختیاری مشتریها ادغام کنند تا یک پردازنده کامل را تولید کنند، یکی از کارهای ODM ها را میتواند گسترش هسته های ARM بر روی semiconductor fabs های قدیمی دانست و این کار تا به امروز هم برای وسایلی که میخواهند با قیمت پایین تولید شوند، بسیار پاسخگو می باشد.در ژانویه 2008 میلادی بیشتر از میلیاردها هسته ARM فروخته شد و iSuppli پیش بینی کرد که این رقم در سال 2011 به 5 میلیارد خواهد رسید.معماری رایجی که در موبایل های هوشمند و PDA ها بکار برده شده است ARM نسخه چهار این پردازنده ها است. پرازنده های XScale و ARM926 به نام ARMv5TE وجود دارند و هم اکنون از آنها بیشتر در StrongARM,ARM925T و ARM7TDMI که بر پایه ARM نسخه چهار استفاده میشود و این پردازنده ها بیشتر در وسایل با تکنولوژی بالا دیده میشود.

| خانواده | نسخه معماری | هسته | ویژگی | حافظه نهان(Cache) بر حسب I/D)/MMU ( | MIPS @ MHz | وسایلی که از این فناوری استفاده کرده اند |
|---|---|---|---|---|---|---|
| ARM1 | ARMv1 | ARM1 | | None | | ARM Evaluation System second processor for BBC Micro |
| ARM2 | ARMv2 | ARM2 | Architecture 2 added the MUL (multiply) instruction | None | 4 MIPS @ 8 MHz 0.33 D MIPS/MHz | Acorn Archimedes, Chessmachine |
| ARM2 | ARMv2a | ARM250 | Integrated MEMC (MMU), Graphics and IO processor. Architec | None, MEMC1a | 7 MIPS @ 12 MHz | Acorn Archimedes |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | ture 2a added the SWP and SWPB (swap) instructi ons. | | | |
| ARM3 | ARMv 2a | ARM 2a | First use of a process or cache on the ARM. | 4K unified | 12 MIPS @ 25 MHz 0.50 D MIPS/M Hz | Acorn Archimed es |
| ARM6 | ARMv 3 | ARM 60 | v3 architec ture first to support address ing 32 bits of memory (as oppose d to 26 bits) | None | 10 MIPS @ 12 MHz | DO 3 Interactiv e Multiplay er, Zarlink GPS Receiver |
| ARM6 | ARMv 3 | ARM 600 | Cache and coproce ssor bus (for FPA10 floating-point unit). | 4K unified | 28 MIPS @ 33 MHz | |
| ARM6 | ARMv 3 | ARM 610 | Cache, no coproce ssor bus. | 4K unified | 17 MIPS @ 20 MHz 0.65 D | Acorn Risc PC 600, Apple Newton |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | MIPS/MHz | 100 series |
| ARM7 | ARMv3 | ARM 700 | | 8KB unified | 40 MHz | Acorn Risc PC prototype CPU card |
| ARM7 | ARMv3 | ARM 710 | | 8KB unified | 40 MHz | Acorn Risc PC 700 |
| ARM7 | ARMv3 | ARM 710a | | 8 KB unified | 40 MHz 0.68 DMIPS/MHz | Acorn Risc PC 700, Apple eMate 300 |
| ARM7 | ARMv3 | ARM 7100 | Integrated SoC. | 8 KB unified | 18 MHz | Psion Series 5 |
| ARM7 | ARMv3 | ARM 7500 | Integrated SoC. | 4 KB unified | 40 MHz | Acorn A7000 |
| ARM7 | ARMv3 | ARM 7500 FE | Integrated SoC. "FE" Added FPA and EDO memory controller. | 4 KB unified | 56 MHz 0.73 DMIPS/MHz | Acorn +A7000 |
| ARM7TDMI | ARMv4T | ARM 7TDMI(-S) | 3-stage pipeline, Thumb | none | 15 MIPS @ 16.8 MHz | Game Boy Advance, Nintendo DS, iPod, Lego NXT, Atmel AT91SAM7, Juice Box |
| ARM7TDMI | ARMv4T | ARM 710T | | 8 KB unified, MMU | 36 MIPS @ | Psion Series 5mx, |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | 40 MHz | Psion Revo/Revo Plus/Diamond Mako |
| ARM7TDMI | ARMv4T | ARM720T | | 8 KB unified, MMU | 60 MIPS @ 59.8 MHz | Zipit Wireless Messenger |
| ARM7TDMI | ARMv4T | ARM740T | | MPU | | |
| ARM7TDMI | ARMv5TEJ | ARM7EJ-S | Jazelle DBX, Enhanced DSP instructions, 5-stage pipeline | none | | |
| StrongARM | ARMv4 | SA-110 | | 16 KB/16 KB, MMU | 203 MHz 1.0 DMIPS/MHz | Apple Newton 2x00 series, Acorn Risc PC, Rebel/Corel Netwinder, Chalice CATS, Psion Netbook |
| StrongARM | ARMv4 | SA-1110 | | 16 KB/16 KB, MMU | 233 MHz | LART, Intel Assabet, Ipaq H36x0, Balloon2, Zaurus SL-5x00, HP |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | Jornada 7xx, Jornada 560 series, Palm Zire 31 |
| ARM8 | ARMv4 | ARM 810 | 5-stage pipeline, static branch prediction, double-bandwidth memory | 8 KB unified, MMU | 84 MIPS @ 72 MHz 1.16 DMIPS/MHz | Acorn Risc PC prototype CPU card |
| ARM9TDMI | ARMv4T | ARM 9TDMI | 5-stage pipeline | none | | |
| ARM9TDMI | ARMv4T | ARM 920T | | 16 KB/16 KB, MMU | 200 MIPS @ 180 MHz | Armadillo, GP32,GP2X (first core), Tapwave Zodiac (Motorola i. MX1), Hewlet Packard HP-49/50 Calculators, Sun SPOT, [Cirrus Logic EP9315], Samsung s3c2442 (HTC TyTN, FIC Neo |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | FreeRun ([ner[8 |
| ARM9 TDMI | ARMv 4T | ARM 922T | | 8 KB/8 KB, MMU | | |
| ARM9 TDMI | ARMv 4T | ARM 940T | | 4 KB/4 KB, MPU | | GP2X (second core), Meizu M6 Mini Player[9 |
| ARM9 E | ARMv 5TE | ARM 946E -S | Enhanced DSP instructions | | variable, tightly coupled memories, MPU | Nintendo DS, Nokia N-Gage, Conexant 802.11 chips |
| ARM9 E | ARMv 5TE | ARM 966E -S | | no cache, TCMs | | ST Micro STR91xF, includes Ethernet |
| ARM9 E | ARMv 5TE | ARM 968E -S | | no cache, TCMs | | |
| ARM9 E | ARMv 5TEJ | ARM 926E J-S | Jazelle DBX, Enhanced DSP instructions | variable, TCMs, MMU | 220 MIPS @ 200 MHz, | Mobile phones: Sony Ericsson (K, W series); Siemens and Benq (x65 series and newer); Texas Instruments OMAP1710, OMAP1610, |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | OMAP1611, OMAP1612, OMAP-L137; Qualcomm MSM6100, MSM6125, MSM6225, MSM6245, MSM6250, MSM6255A, MSM6260, MSM6275, MSM6280, MSM6300, MSM6500, MSM6800; Freescale i.MX21, i.MX27, Atmel AT91SAM9, GPH Wiz, Marvell Feroceon |
| ARM9E | ARMv5TE | ARM996H | Clockless | no caches, | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | S | processor, Enhanced DSP instructions | TCMs, MPU | | |
| ARM10E | ARMv5TE | ARM1020E | (VFP), 6-stage pipeline, Enhanced DSP instructions | 32 KB/32 KB, MMU | | |
| ARM10E | ARMv5TE | ARM1022E | (VFP) | 16 KB/16 KB, MMU | | |
| ARM10E | ARMv5TEJ | ARM1026EJ-S | Jazelle DBX, Enhanced DSP instructions | variable, MMU or MPU | | |
| XScale | ARMv5TE | 80200/IOP310/IOP315 | I/O Processor, Enhanced DSP instructions | | | |
| XScale | ARMv5TE | 80219 | | | 400/600 MHz | Thecus N2100 |
| XScale | ARMv5TE | IOP321 | | | BogoMips 600 @ 600 MHz | Iyonix |
| XScale | ARMv5TE | IOP33x | | | | |
| XScale | ARMv5TE | IOP34x | 1-2 core, RAID Acceleration | 32K/32K L1, 512K L2, MMU | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| XScale | ARMv5TE | PXA210/PXA250 | Applications processor, 7-stage pipeline | | PXA210: 133 and 200 MHz, PXA250: 200, 300, and 400 MHz | Zaurus SL-5600, iPAQ H3900, Sony CLIÉ NX60, NX70V, NZ90 |
| XScale | ARMv5TE | PXA255 | | 32KB/32KB, MMU | 400 BogoMips @ 400 MHz | Gumstix basix & connex, Palm Tungsten E2,Mentor Ranger & Stryder, iRex ILiad |
| XScale | ARMv5TE | PXA263 | | | 200, 300 and 400 MHz | Sony CLIÉ NX73V, NX80V |
| XScale | ARMv5TE | PXA26x | | | default 400 MHz, up to 624 MHz | Palm Tungsten T3 |
| XScale | ARMv5TE | PXA27x | Applications processor | Kb/32 32 Kb, MMU | 800 MIPS @ 624 MHz | Gumstix verdex, HTC Universal, HP hx4700, Zaurus SL-C1000, 3000, 3100, 3200, |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | Dell Axim x30, x50, and x51 series, Motorola Q, Balloon3, Trolltech Greenphone, Palm TX, Motorola Ezx Platform A728, A780, A910, A1200, E680, E680i, E680g, E690, E895, Rokr E2, Rokr E6, Fujitsu Siemens LOOX N560, Toshiba Portégé G500, Trēo 650-755p, Zipit Z2 |
| XScale | ARMv5TE | PXA800(E)F | | | | |
| XScale | ARMv5TE | Monahans | | 32KB/32KB L1, TCM, MMU | 1000 MIPS @ 1.25 GHz | |

| | | | | | | |
|---|---|---|---|---|---|---|
| XScale | ARMv5TE | PXA900 | | | | Blackberry 8700, Blackberry Pearl (8100) |
| XScale | ARMv5TE | IXC1100 | Control Plane Processor | | | |
| XScale | ARMv5TE | IXP2400/IXP2800 | | | | |
| XScale | ARMv5TE | IXP2850 | | | | |
| XScale | ARMv5TE | IXP2325/IXP2350 | | | | |
| XScale | ARMv5TE | IXP42x | | | | NSLU2 |
| IXP460/IXP465 | | | | | | NSLU2 |
| ARM11 | ARMv6 | ARM1136J(F)-S | SIMD, Jazelle DBX, (VFP), 8-stage pipeline | variable, MMU | 740 @ 532-665 MHz (i.MX31 SoC), 400-528 MHz | Texas Instruments OMAP2420 (Nokia E90, Nokia N93, Nokia N95, Nokia N82), Zune, BUGbase, Nokia N800, Nokia N810, Qualcom |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | m MSM7200 (with integrated ARM926EJ-S Coprocessor@274 MHz, used in Eten Glofiish, HTC TyTN II, HTC Nike), Freescale i.MX31 (which was used in the original Zune 30gb and Toshiba Gigabeat .(S |
| ARM11 | ARMv6T2 | ARM1156T2(F)-S | SIMD, Thumb-2, (VFP), 9-stage pipeline | variable, MPU | | |
| ARM11 | ARMv6KZ | ARM1176JZ(F)-S | , SIMD Jazelle DBX, (VFP | variable, MMU+TrustZone | | Apple iPhone, Apple iPod touch, Conexant CX2427X , Motorola |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | RIZR Z8, Motorola RIZR Z10 |
| ARM11 | ARMv6K | ARM11 MPCore | 1-4 core SMP , SIMD Jazelle DBX, (VFP) | variable, MMU | | Nvidia APX 2500 |
| Cortex | ARMv7-A | Cortex-A8 | Application profile, VFP, NEON, Jazelle RCT, Thumb-2, 13-stage superscalar pipeline | variable (L1+L2), MMU+TrustZone | up to 2000 (2.0 DMIPS/MHz in speed from 600 MHz to greater than 1 GHz) | Texas Instruments [[OMAP\|OMAP3430][SBM\|SBM7000],\|Gumstix Overo Earth, Pandora, Archos 5 |
| Cortex | ARMv7-A | Cortex-A9 | Application profile, (VFP), (NEON), Jazelle RCT and DBX, Thumb-2, Out-of-order speculative issue superscalar | MMU+TrustZone | 2.0 DMIPS/MHz | |
| Cortex | ARMv7-A | Cortex-A9 MPC | As Cortex-A9, 1-4 core | MMU+TrustZone | 2.0 DMIPS/MHz | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | ore | SMP | | | |
| Cortex | ARMv7-R | Cortex-R4(F) | Embedded profile, (FPU) | variable cache, MPU optional | DMIPS 600 | Broadcom is a user, TMS570 from Texas Instruments |
| Cortex | ARMv7-M | Cortex-M3 | Microcontroller profile, Thumb-2 only | no cache, (MPU) | 125 DMIPS @ 100 MHz | Energy Micro's EFM32, Luminary Micro microcontroller family, ST Microelectronics STM32 |
| Cortex | ARMv6-M | Cortex-M1 | FPGA targeted, Microcontroller profile, Thumb-2 (BL, MRS, MSR, ISB, DSB, and DMB) | None, tightly coupled memory optional. | Up to 136 DMIPS @ 170 MHz (0.8 DMIPS/MHz, MHz achievable FPGA-dependent) | "Actel ProASIC 3 and Actel Fusion PSC devices will sample in [Q3 2007" |

برای تمیز نگه داشتن طراحی و ساده و سریع بودن در سیمکشی آن ها از میکروکد استفاده نشده است،خیلی ساده تر از پردازنده های 8 بیتی(مترجم:AVR هم 8 بیتی است)6502 که در سری پیشین میکرو کامپیوتر های شرکت Acron استفاده میشد.
معماری ARM شامل قابلیت های طراحی ریسک زیر است:

- Load/store architecture

- No support for misaligned memory accesses (now supported in ARMv6 cores, with some exceptions related to load/store multiple word instructions)
- Uniform 16 × 32-bit register file
- Fixed instruction width of 32 bits to ease decoding and pipelining, at the cost of decreased code density. (Later "Thumb mode" ,increased code density.)
- Mostly single-cycle execution

برای رسیدن به این سادگی نسبت به معاصران عصر خودش همچون intel 80286 و Motorola 68020 بعضی از ویژگی های منحصر بفردی درطراحیش بکار رفته است.همچون:

- اجراهای شرطی بیشتر دستورها،باعث کمتر شدن مقدار پردازش شده است و همچنین جبران فقدان branch predicator شده است.
- عملیات ریاضی فقط زمانی بر روی کدهای شرطی اثر میگذارد،که نیاز باشد.
- Barrel shifter های 32 بیتی که میتوانند بدون کاهش دادن کارایی در بیشتر عملیات ریاضی و همچنین محاسبات آدرس دهی استفاده بشوند.
- شاخص گذاری قوی ماژول های آدرس دهی
- لینک رجیستر برای فراخوانی سریعتر توابع
- 2، سطح اولویت ساده،ولی سریع وقفه—interrupt- زیرسیستم که وظیفه سویچ کردن بانک های رجیستر را بر عهده دارد.

یکی از ویژگی های جالب در طراحی ARM،وجود کدهای شرطی 4 بیتی در ابتدای هر عملیاتی است،به این معنی که اجرای هر دستور،تابع یک شرط اختیاری هست(مترجم:شاید به این معنیست که از ابتدا تا انتهای تمام کدهای صفحه رایکجا پردازش نمیکند،بلکه بنابر نیاز هنگامی که به هر دستور رسید آنرا اجرا میکند);در دیگر طراحی های پردازنده دستورات شرطی فقط در هر شاخه عملیات ها قرار دارند.
این تدابیر تاثیر قابل ملاحضه ای را در کاهش کد گذاری بیتها یی که میخواهند درون حافضه مقیم شوند، میگذارد،ولی از لحاظ دیگر،اجازه نمیدهد که کدهای شرطی کوچک همانند if…statement ها به طور کامل در حافظه قرار بگیرند.یک مثال استاندارد برای این موضوع، الگوریتم اقلیدسی میباشد:
در زبان برنامه نویسی c دستور حلقه به صورت زیر میباشد:

```
While ( i != j)
{
    If (i > j)
        i-=j ;
    else
      j-=i ;
}
```

دستور اسمبلی حلقه در ARM بگونه زیر میباشد :

```
Loop CMP Ri , Rj
SUBGT Ri,Ri,Rj
SUBLT Rj,Rj,Ri
BNE loop
```

معماری ARM در این مثال مانع شاخه های که بین then سپس else میشود.
(مترجم: دوستان توجه کنند که then سپس else شبیه syntax زبان VB میباشد.که اینگونه است:

```
If I > j
Then I -= j
Else j -=i
EndIf
```

نویسنده به این دلیل که زبان VB به زبان انسان نزدیکتر است اینگونه گفته است!!!!!!!!!)

یکی دیگر از ویژگی های یکتا که درون مجموعه دستورات هست قابلیت قرار دادن دستورات شیفت درون دیگر دستورات ریاضی،منطقی و جابجایی رجیستری است،برای مثال به نمونه کدی که به زبان c نوشته شده است دقت کنید:

```
A += (j << 2 );
```

میتواند مانند یک دستور در ARM پردازش بشود.

```
ADD Ra, Ra, Rj, LSL #2
```

مثال بالا حاصل پردازش دستور در یک ARM بود،اینگونه دستورات فشرده میشوند و به همین دلیل نیاز کمتری به دسترسی به حافظه پیدا میکنند،بنابرین از خط لوله-pipeline- انتقالی داده ها به طور موثر تری استفاده میشود.گرچه اینگونه پردازش ها در ARM دربسیاری حالات خیلی خوب عمل میکند ولی هیچگاه نمیتواند رقیب خوبی برای پردازنده هایی با معماری پیچیده باشد(مترجم:به این دلیل میشود اینگونه پردازش های literal را در ARM انجام داد چون ترانزیستورهای موجود در ARM حدود 35000 عدد میباشد و معماری ساده ای هم دارد،ولی نمیتوان اینگونه پردازش ها را برای پردازنده های 4 هسته ای هم بکار برد)
همچنین ARM دارای خصوصیتیست که به ندرت در بقیه معماری های RISK دیده میشود،مانندPC-relative addressing (pc همان شمارنده برنامه یا program counter میباشد)-در حقیقت شمارنده برنامه-program counter- در ARM یکی از همان 16 رجیستر از پیش رزو شده میباشد، و همچنین مدل آدرس دهی pre-and post- increment در ARM وجود دارد.
یکی دیگر از مطالبی که باید آنرا یادآوری کنیم اینکه ARM مدت زیادی نیست که تولید شده است،ولی در طول مدت حضورش همیشه تعداد دستورعمل هایی که میشود با آن اجرا کرد بیشتر شده است.برخی از ARM های اولیه (قبل از ARM7TDMI)،برای مثال قادر نبودند که یک کمیت 2 بایتی را ذخیره کنند بنابرین گفتن اینکه ARM قادر نیست که داده های C را ذخیره کند کمی مشکل است(مترجم:مثلا داده های float یا decimal که برای ایجادشان بیشتر از 4 بایت نیاز دارند) "ویکی پدیا ذکر کرده است که این جمله نیاز به مدرک دارد".
ARM7 و طراحی های پیشین های،سه pipeline متفاوت داشتند که عبارت بودند از واکشیfetch، رمز گشایی decode و اجراییexecute
طراحی های با کارایی بالاتر مانند ARM9 از 5 نوع pipeline استفاده کردندکه دوتای اضافه شده عبارت بودند از: faster adder و بیشتر کردن شاخه های prediction logic ( مترجم : که زمان اجرای هر فرایندی را طبق الگوهایی حدث میزند) .
معماری ARM امکان گسترش دادن دستور العمل -instruction- را با استفاده از پردازننده کمکی-coprocessors- را مهیا ساخته است.که میتوان با دستورات MCR,MRC,MRRC و MCRR آنها را از سمت نرم افزار آدرس دهی کنیم.فضای پردازنده کمکی به صورت منتقی به 16 پردازنده کمکی با اندیس هایی از صفر تا 15 تقسیم

میشوند.کمک پردازنده ی شماره پانزده(cp15) توسط بعضی از دستورات کنترلی مانند کنترل کش و دستورات MMU(در پردازنده ای که این قابلیت را دارند) از قبل رزو میشود. در ماشین هایی که بر پایه ARM هستند،وسایل جانبی اغلب بوسیله نگاشت رجیسترهای فیزیکیشان در فضای حافظه ARM یا بوسیله فضای پردازنده جانبی یا بوسیله ارتباطشان به دیگر وسایل(بوسیله گذرگاه باس) که خود آنها قبلا به پردازنده وصل شده اند(مترجم: روش آخر مثل روش وصل کردن USB میباشد!)، میتوان آنها را به دستگاه اصلی وصل کرد. دستیابی به پردازنده های کمکی اغلب با زمان تاخیر همراه هست به همین دلیل برخی وسایل جانبی(مانند کنترلگر وقفه XScale) به گونه ای طراحی شده اند که میتوانند هم از طریق حافظه و هم از طریق پردازنده کمکی به دستگاه وصل شوند.

## Thumb

برای بیشتر شدن کارایی سیستم،میشود کدها را از لحاظ حجمی،چگال تر کرد،پردازنده های ARM7TDMI مدی به نام Thumb دارند.وقتی پردازنده در این مد است،پردازنده دستورالعمل ها را به صورت 16 بیتی اجرا میکند(مترجم:در حالت عادی دستورات در مد 32 بیتی اجرا میشوند). بیشتر این دستورهایی 16 بیتی میتوانند به طور مستقیم بر روی دستورات 32 بیتی ARM نگاشت شوند. این صرفه جویی در فضای حافظه در غیاث با full mode از راه تبدیلات ضمنی عملوندها و تا جای امکان محدود کردن آنها بدست می آید. در مد Thumb کدهای عملگر-opcodes- توانایی کمتر را دارند.برای مثال فقط شاخه ها میتوانند شرطی باشند(مترجم:مباحث مربوط به درخت های ساختمان داده) و بیشتر کدهای عملگر محدود به استفاده از نیمی از رجیسترهای عمومی پردازنده هستند.در کل، کدهای عملگر کوچکتر باعث چگالتر شدن کدها میشوند،گرچه برخی از این عملگرها نیازمند دستور العمل های بیشتری هستند.در این شرایط که پورت های حافظه یا پهنای گذرگاه ها به کمتر از 32 بیت محدود میشود،کدهای عملگر کوچک شده به بیشتر شدن کارایی کمک میکنند و با فضای باند محدود شده، تعداد کمتری کدهای برنامه نویسی در پردازنده بارگیری-load- میشوند.
سخت افزارهای تعبیه شده ای مانند Game Boy Advance یک مقدار بسیار کمی را برای دسترسی به RAM را در کانال 32 بیتی تخصیص میدهند و در بیشتر زمان ها داده ها فقط اجازه دارند از کانال 16 بیتی یا یک کانال ثانویه باریک تری عبور کنند.این عبارت، این مفهوم را میرساند که برای کامپایل در روش Thum فقط بخشی از داده ها که برای پردازنده نقش حیاتی را دارند اجازه استفاده از فضای کامل باند 32 بیتی رم را دارند وفقط باید این دستور العمل های حیاتی را از گذرگاه 32 بیتی عبور دهند.
اولین پردازنده با رمزگشای Thumb در ARM7TDMI بکار رفت.تمام سری های ARM9 و سری های بعدی آن و همچنین XScale همگی دارای دستورالعمل رمزگشای Tumb هستند.

## دستورات اضافه شده برای پردازش سیگنال های دیجیتالی (DSP)

برای بهبود معماری ARM در ضمینه پردازش سیگنال های دیجیتالی و برنامه های چندرسانه ای، چند دستورالعمل جدید به مجموعه دستورالعمل ها اضافه شده است.برای نشان دادن وجود این فناوری در پردازنده ها ،علامت"E" را در نام میکرو مینویسند، یک نمونه اش معماری ARMv5TE و ARMv5TEJ.

## Jazelle

فناوری که به نام Jazelle DBX (اجرای مستقیم بایت کدها) نامیده میشود به ARM های جدید این امکان را میدهد که برخی بایت کدهای جاوا را در سخت افزار به عنوان یک برنامه سوم درکنار ARM های موجود و در کنار حالت Thumb اجرا بشود.
بیشتر سود استفاده از Jazzelle عاید شرکت های تلفن همراه میشود که میتوانند سرعت اجرای بازی ها و برنامه های کاربردی که تحت جاوا هستند را بالا ببرند."ویکی پدیا گفته است که این گفته نیاز به منبع دارد".
Jazelle همچون ماشین مجازی جاوا (JVM) میتوانند بایت کدهای جاوا را در سخت افزار اجرا کنند. اجرای آنها بوسیله نرم افزار باعث پیچیده تر شدن آنها و همچنین محدود شدن عملگر های آنها میشود.
ARM ادعا کرده است که بیش از 95 درصد بایت کدهای جاوا قابلیت اجرای مستقیم در پردازنده را دارند.
فناوری Jazelle در معماری ARMv5TEJ وجود دارد و ARM926EJ-S اولین پردازنده ای بود که با فناوری Jazelle معرفی شد،
Jazelle با آمدن حرف J در نام پردازنده ها مشخص میشود.
انتشار مشخصات این فناوری بسیار ناقص بود و فقط به همین بسنده کرده بود که کدهای سیستم عاملی بوسیله Jazell میتوانند از JVM پشتیبانی کنند. و همچنین بیان کرده بود که فقط JVM احتیاج دارد(یا اجازه دارد) که به لایه سخت افزار وابسته باشد.و این ارتباط قوی بین سخت افزار و JVM بدون هیچ خللی در کار بقیه نرم افزار ها باعث راحت تر شدن کار میشود.در حقیقت، این فناوری، یک سیاست هوشمندانه ای است در جهت کنترل داشتن بر روی هر JVM است.

## پیاده سازی

اکستنشن Jazelle یک مرحله اضافه شده ما بین مرحله وارسی-fetch- و رمزگشایی-decode- درون pipeline پردازنده میباشد.در این فناوری بایت کدهای جاوا به یک یا چند دستور محلی-native- ARM تبدیل-convert- میشوند.
Jazelle اکثر دستورات ساده JVM را برای سخت افزار ترجمه میکند.این پیاده سازی هزینه سنگین ترجمه را کاهش میدهد.علاوه بر این نیاز به داشتن JIT یا دیگر ماشین های مجازی را کاهش میدهد.دستور های بایت کدی که در پیاده سازی Jazelle وجود ندارد به یک دستور معمولی سخت افزاری تبدیل میشوند تا خللی در اجرا بوجود نیاید و دستورهای دیگر فراخوانی شوند.جزییات این پیاده سازی منتشر نشده است.
مد Jazelle بوسیله دستور BXJ فعال میشود. پیاده سازی سخت افزاری Jazelle فقط یک سری از بایت کدهای JVM را پشتیبانی میکند; برای بایت کدهای غیر قابل کنترل یا پنهان شده توسط سیستم عامل ، سخت افزار ARM،نرم افزار JVM را فراخوانی میکند.نیازی نیست که ماشین مجازی بداند که کدام بایت کد درون نرم افزار پیاده سازی شده است و کدام یک درون سخت افزار،خواه درون نرم افزار باشد یا سخت افزار، JVM به صورت کامل دستور العمل های بایت کد را اجرا میکند.

ادامه دارد....

اعضای گروه:

1. آقای رضا منصوری
2. آقای امید ادیبان
3. آقای بابک اهری
4. آقای داریوش عباسی
5. آقای وحید سلیمانزاده
6. آقای محمد امین شریفی

منبع: ویکی پدیای انگلیسی

نکته:تیم توسعه دهنده ARM هنوز در حال عضو گیری است

**به نام زیباترین دوست**