ARM

ARM Low-power Processors and Architectures

Dan Millett
Verification Enablement
Processor Division

Agenda

- Introduction to ARM Ltd

ARM Architecture/Programmers Model

Data Path and Pipelines

System Design

Development Tools

ARM Ltd

- Founded in November 1990
  - Spun out of Acorn Computers
  - Initial funding from Apple, Acorn and VLSI
- Designs the ARM range of RISC processor cores
  - Licenses ARM core designs to semiconductor partners who fabricate and sell to their customers
  - ARM does not fabricate silicon itself
- Also develop technologies to assist with the design-in of the ARM architecture
  - Software tools, boards, debug hardware
  - Application software
  - Bus architectures
  - Peripherals, etc

ARM's Activities

Connected Community

Development Tools

Software IP

Processors

System Level IP:

Data Engines

Fabric

3D Graphics

Physical IP

ARM Connected Community – 700+

Huge Range of Applications

Tele-parking

Intelligent toys

Utility Meters

IR Fire Detector

Exercise Machines

Energy Efficient Appliances

Intelligent Vending

Equipment Adopting 32-bit ARM Microcontrollers

How many ARM's Do You Have?

Mobile phones

~100%
market share

Smartphones

3x 100%
market share

Mobile Computers

5x 100%
market share

Digital TVs

35%
market share

Disk Drives

~75%
market share

PC Peripherals

40%
market share

Cars

5x 50%
market share

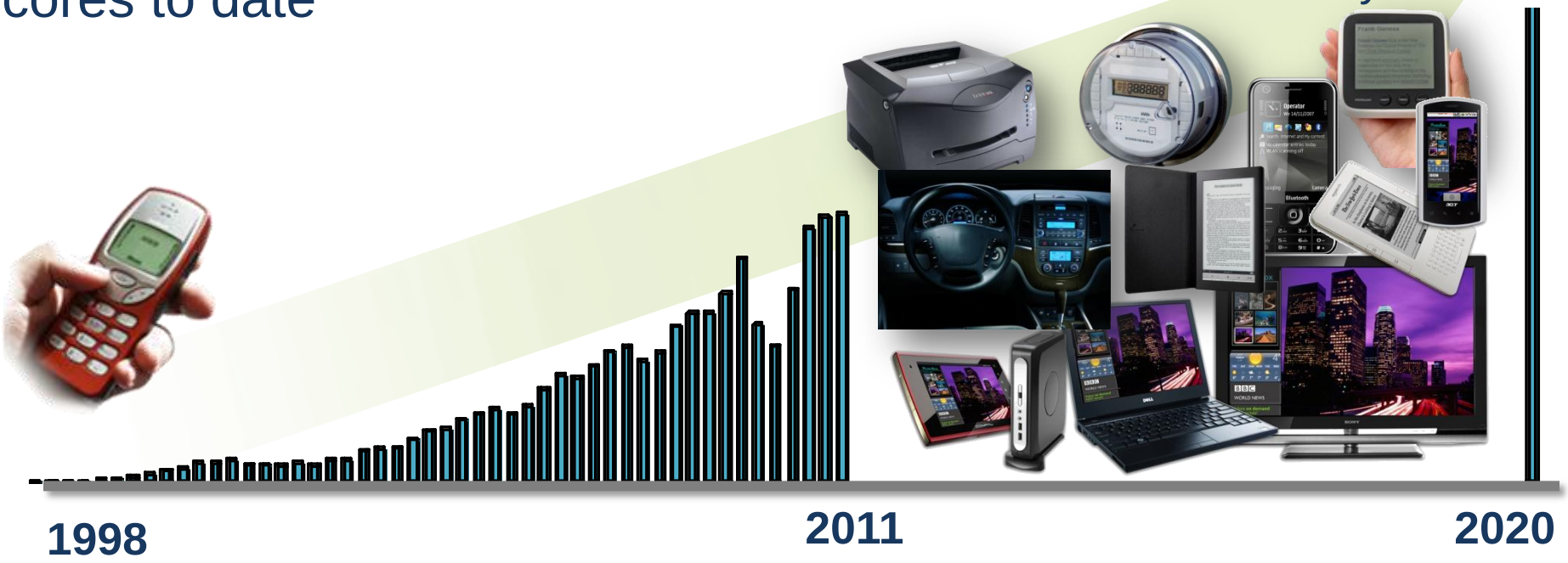
Microcontrollers

35%
market share

Huge Opportunity For ARM Technology

25+
billion
cores to date

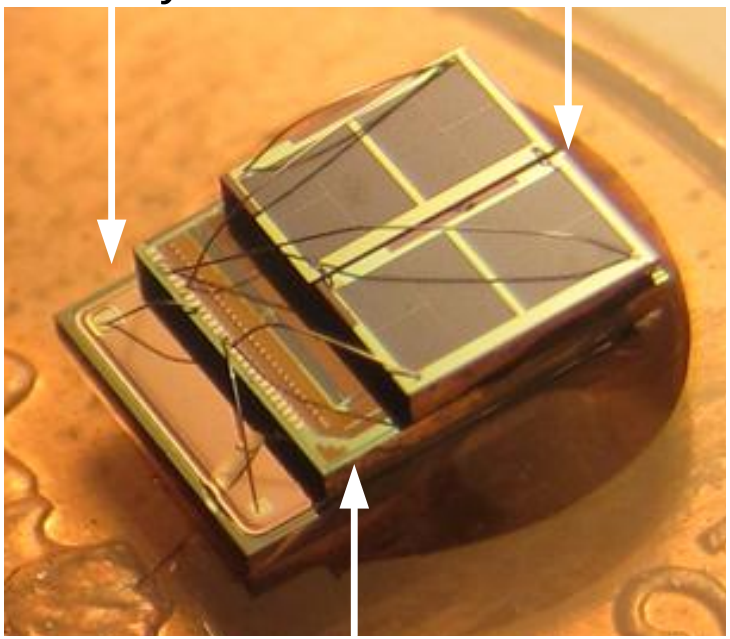
100+
billion cores accumulated after next 9 yrs



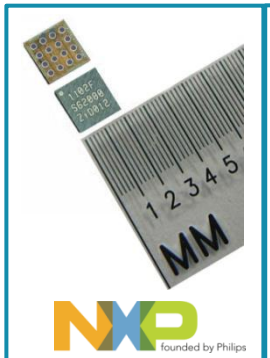
World's Smallest ARM Computer?

Wireless Sensor Network

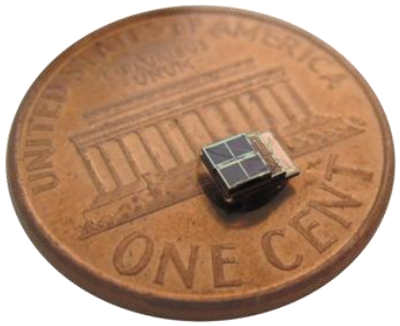
- **Sensors, timers**
- **Cortex-M0 +16KB RAM 65nm**
  UWB Radio antenna
- **10 kB Storage memory**
  ~3fW/bit
- **12µAh Li-ion Battery**



Processor, SRAM and PMU



Cortex-M0; 65¢



University of Michigan

Wirelessly networked into large scale sensor arrays

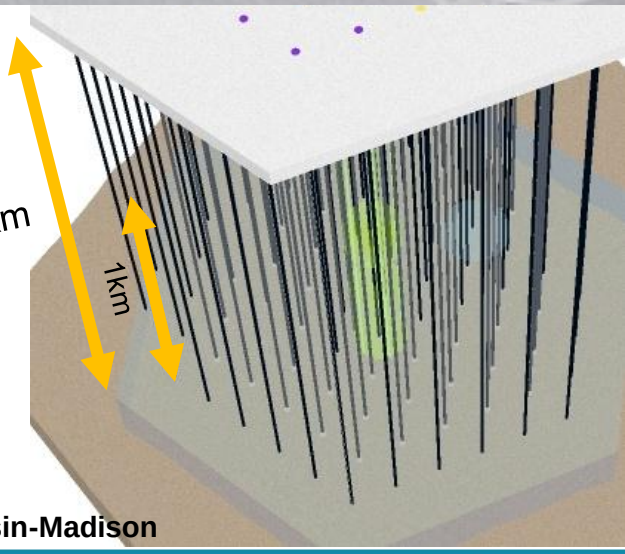
World's Largest ARM Computer?

4200 ARM powered Neutrino Detectors

70 bore holes 2.5km deep

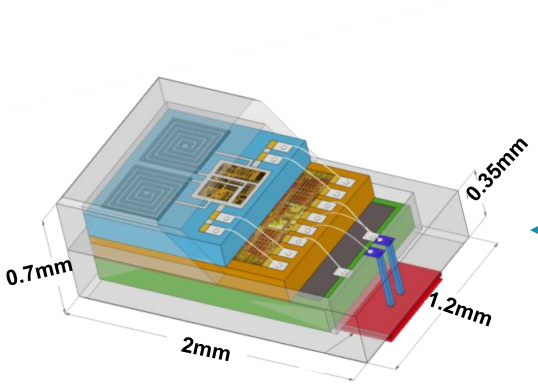
60 detectors per string starting 1.5km down

1km^3 of active telescope

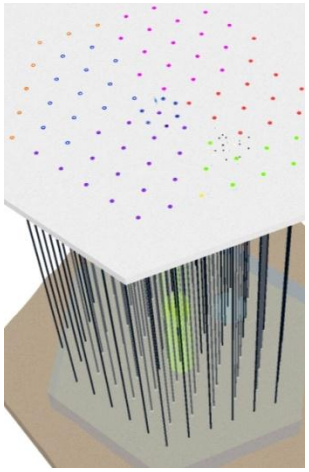


Work supported by the National Science Foundation and University of Wisconsin-Madison

From 1mm³ to 1km³



1mm³ ⟷ 1km³



10¢ ⟷ $1000

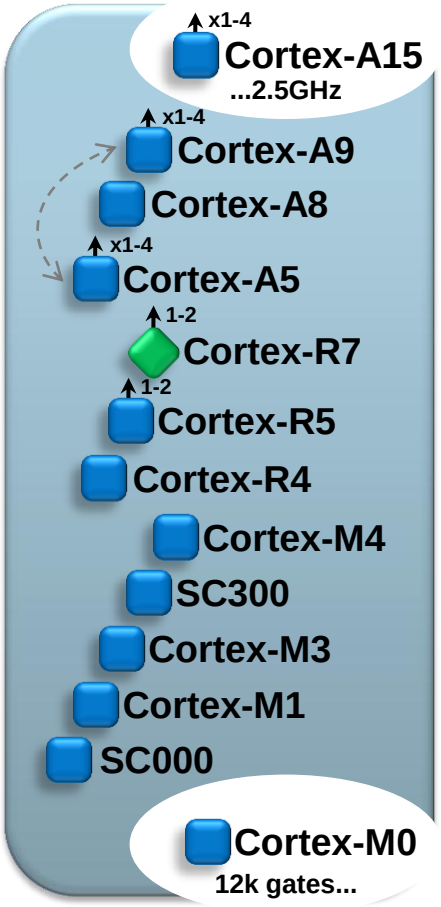
Embedded | Mobile | Consumer | Home | Enterprise | Mobile Computing | PC | Server | HPC

Agenda

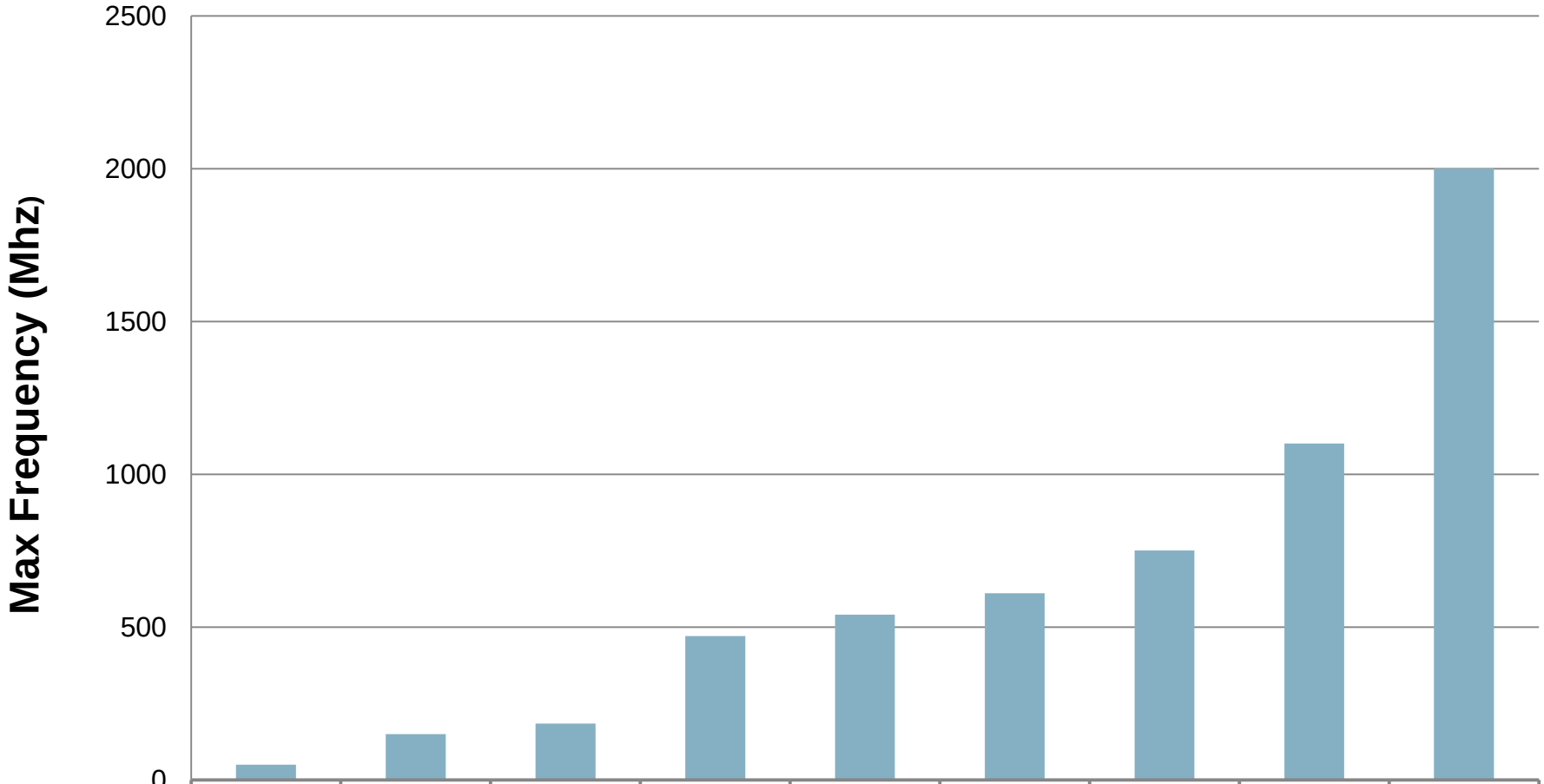
- Introduction to ARM Ltd
- ARM Architecture/Programmers Model
- Data Path and Pipelines
- System Design
- Development Tools

ARM Cortex Advanced Processors

- ARM Cortex-**A** family:
  - Applications processors
  - Targeted for OS's, graphics, demanding tasks
- ARM Cortex-**R** family:
  - Embedded processors
  - Real-time signal processing, control applications
- ARM Cortex-**M** family:
  - Microcontroller-oriented processors
  - MCU, ASSP, and SoC applications



Relative Performance*



	Cortex-M0	Cortex-M3	ARM7	ARM926	ARM1026	ARM1136	ARM1176	Cortex-A8	Cortex-A9 Dual-core
Max Freq (MHz)	50	150	184	470	540	610	750	1100	2000
Min Power (mW/MHz)	0.012	0.06	0.35	0.235	0.36	0.335	0.568	0.43	0.5

*Represents attainable speeds in 130, 90, 65, or 45nm processes

Cortex family

Cortex-A8

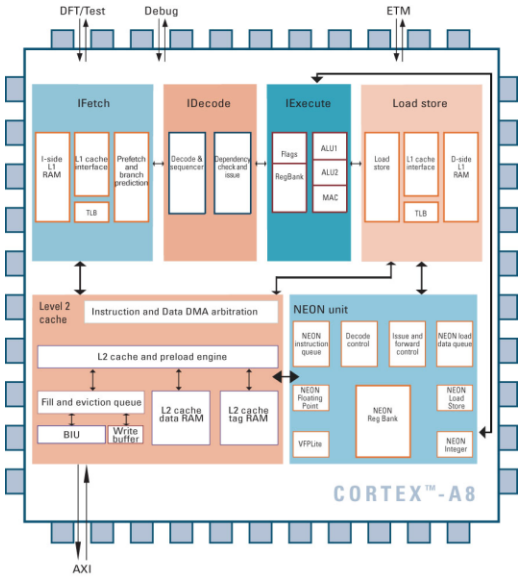
- Architecture v7A
- MMU
- AXI
- VFP & NEON support

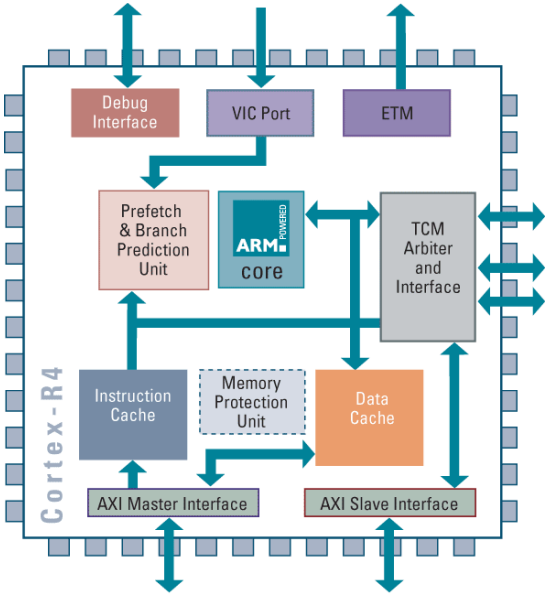
Cortex-R4

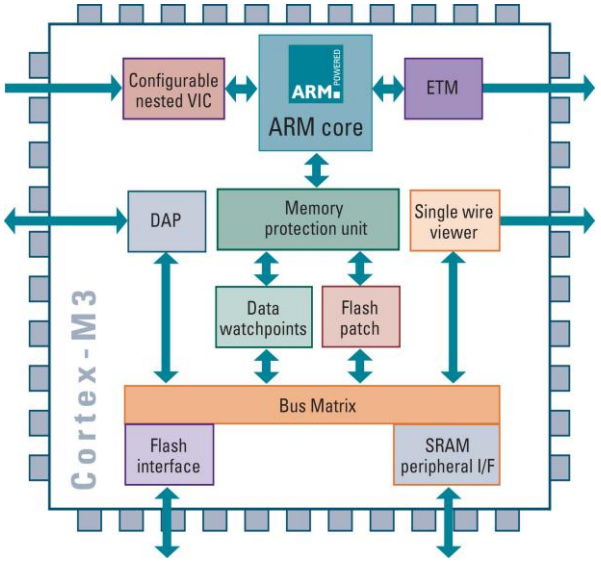
- Architecture v7R
- MPU (optional)
- AXI
- Dual Issue

Cortex-M3

- Architecture v7M
- MPU (optional)
- AHB Lite & APB



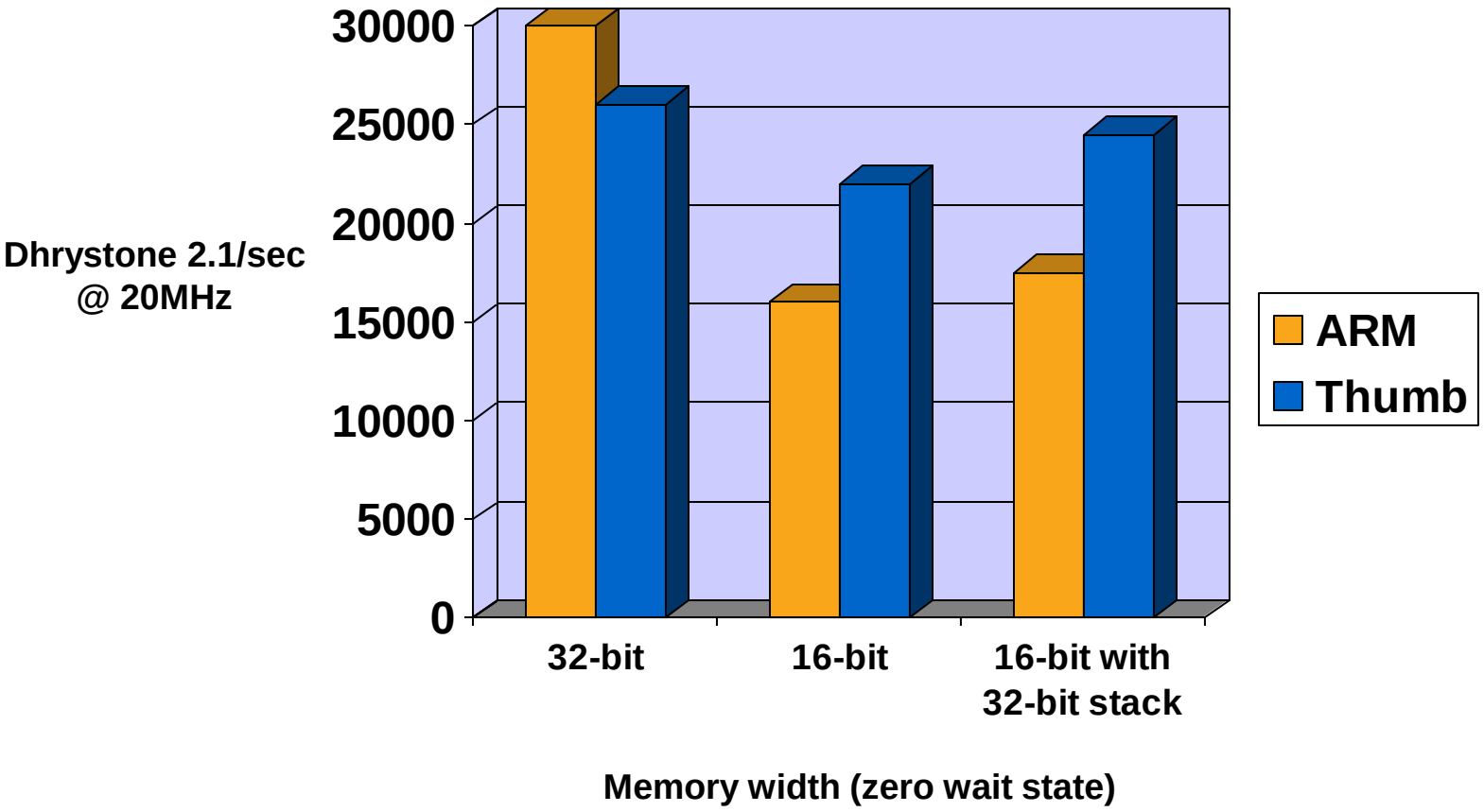




Data Sizes and Instruction Sets

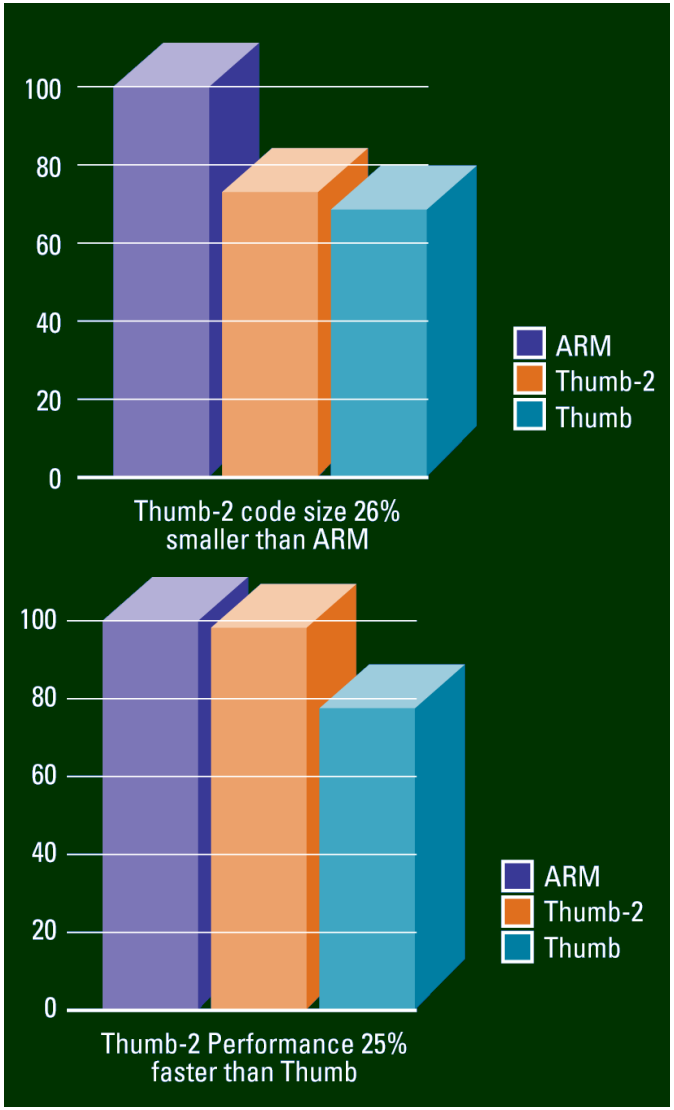
- The ARM is a 32-bit architecture.
- When used in relation to the ARM:
  - **Byte** means 8 bits
  - **Halfword** means 16 bits (two bytes)
  - **Word** means 32 bits (four bytes)
- Most ARM's implement two instruction sets
  - 32-bit ARM Instruction Set
  - 16-bit Thumb Instruction Set
- Jazelle cores can also execute Java bytecode

ARM and Thumb Performance



The Thumb-2 instruction set

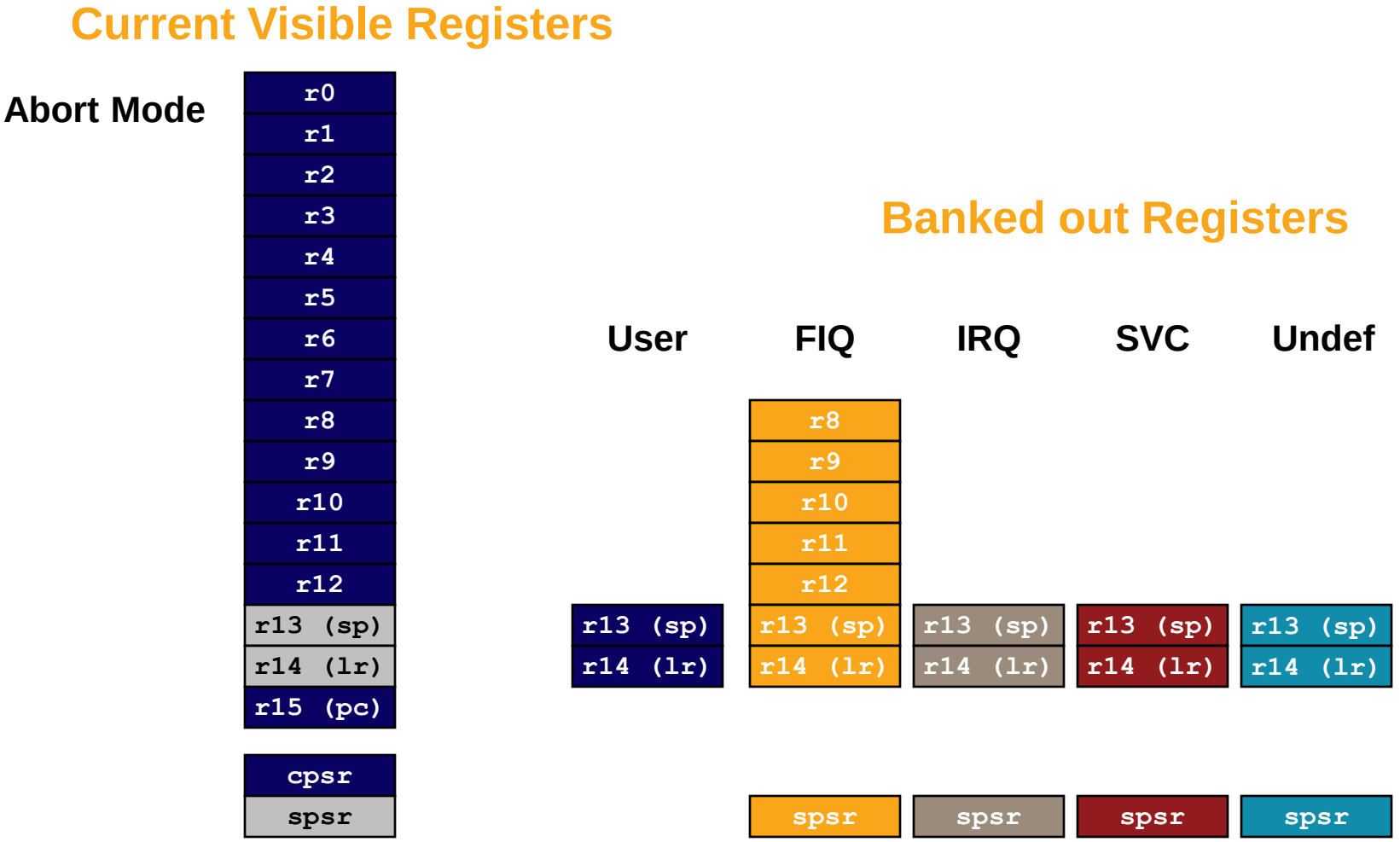
- Variable-length instructions
  - ARM instructions are a fixed length of 32 bits
  - Thumb instructions are a fixed length of 16 bits
  - Thumb-2 instructions can be either 16-bit or 32-bit
- Thumb-2 gives approximately 26% improvement in code density over ARM
- Thumb-2 gives approximately 25% improvement in performance over Thumb



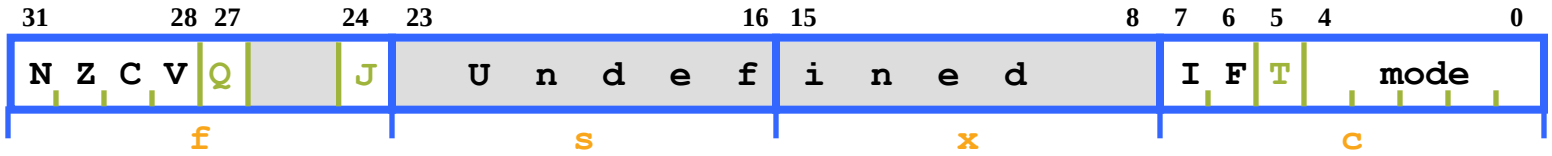
Processor Modes

- The ARM has seven basic operating modes:
  - **User** : unprivileged mode under which most tasks run
  - **FIQ** : entered when a high priority (fast) interrupt is raised
  - **IRQ** : entered when a low priority (normal) interrupt is raised
  - **Supervisor** : entered on reset and when a Software Interrupt instruction is executed
  - **Abort** : used to handle memory access violations
  - **Undef** : used to handle undefined instructions
  - **System** : privileged mode using the same registers as user mode

The ARM Register Set



Program Status Registers



- Condition code flags
  - N = Negative result from ALU
  - Z = Zero result from ALU
  - C = ALU operation Carried out
  - V = ALU operation oVerflowed
- Sticky Overflow flag - Q flag
  - Architecture 5TE/J only
  - Indicates if saturation has occurred
- J bit
  - Architecture 5TEJ only
  - J = 1: Processor in Jazelle state
- Interrupt Disable bits.
  - I = 1: Disables the IRQ.
  - F = 1: Disables the FIQ.
- T Bit
  - Architecture xT only
  - T = 0: Processor in ARM state
  - T = 1: Processor in Thumb state
- Mode bits
  - Specify the processor mode

Conditional Execution and Flags

- ARM instructions can be made to execute conditionally by postfixing them with the appropriate condition code field.
  - This improves code density *and* performance by reducing the number of forward branch instructions.

```
  CMP   r3,#0
  BEQ   skip
  ADD   r0,r1,r2
skip
```

```
CMP   r3,#0
ADDNE r0,r1,r2
```

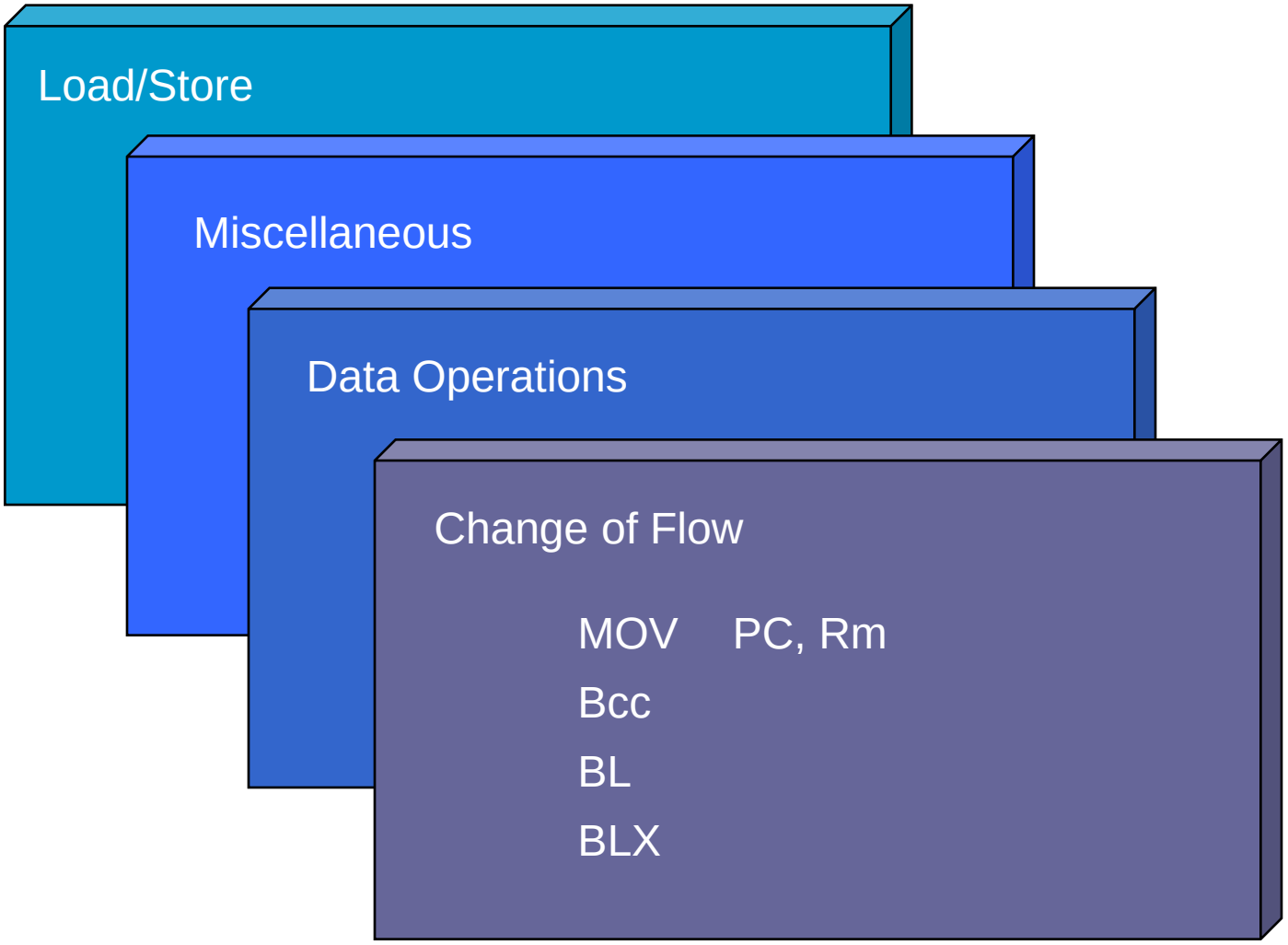
- By default, data processing instructions do not affect the condition code flags but the flags can be optionally set by using “S”. CMP does not need “S”.

```
loop
  …
  SUBS r1,r1,#1
  BNE loop
```

decrement r1 and set flags

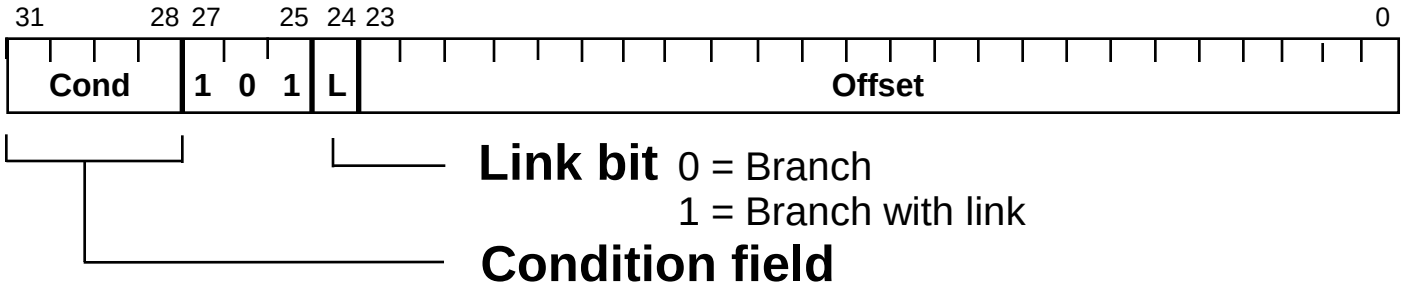
if Z flag clear then branch

Classes of Instructions



Branch instructions

- Branch : `B{<cond>} label`
- Branch with Link : `BL{<cond>} subroutine_label`



- The processor core shifts the offset field left by 2 positions, sign-extends it and adds it to the PC
  - ± 32 Mbyte range
  - How to perform longer branches?

Data processing Instructions

- Consist of :
  - Arithmetic: `ADD` `ADC` `SUB` `SBC` `RSB` `RSC`
  - Logical: `AND` `ORR` `EOR` `BIC`
  - Comparisons: `CMP` `CMN` `TST` `TEQ`
  - Data movement: `MOV` `MVN`

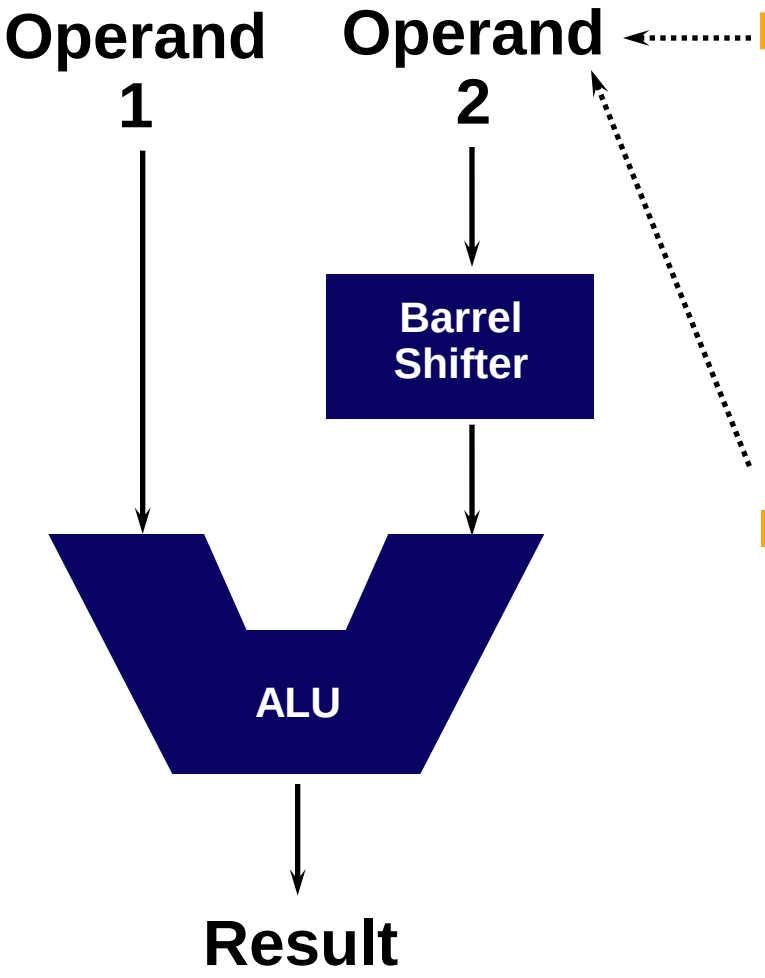
- These instructions only work on registers, NOT memory.

- Syntax:

```
<Operation>{<cond>}{S} Rd, Rn, Operand2
```

  - Comparisons set flags only - they do not specify Rd
  - Data movement does not specify Rn
  - Second operand is sent to the ALU via barrel shifter.

Using a Barrel Shifter:The 2nd Operand



Register, optionally with shift operation

- Shift value can be either be:
  - 5 bit unsigned integer
  - Specified in bottom byte of another register.
- Used for multiplication by constant

Immediate value

- 8 bit number, with a range of 0-255.
  - Rotated right through even number of positions
- Allows increased range of 32-bit constants to be loaded directly into registers

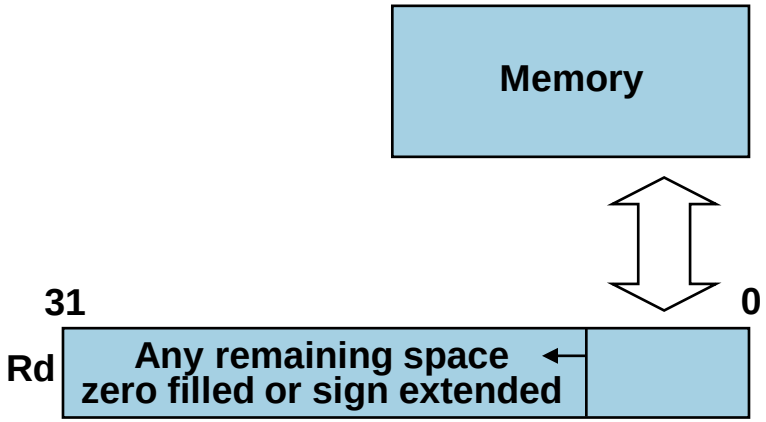
Data Processing Instruction Examples

```
MOV r3, r0          ; copies r0 into r3
MVN r6, r8          ; copies the complement of r8 into r6
ADD r0, r1, r2            ; r0 = r1 + r2
ADC r0, r1, r2            ; r0 = r1 + r2 + <carry flag>
SUB r3, r1, r7            ; r3 = r1 - r7
RSB r3, r1, r7            ; r3 = r7 - r1
SBC r3, r1, r7            ; r3 = r1 - (r7 + <carry flag>)

AND r0, r1, #0xA5   ; r0 = r1 & 0xA5
BIC r0, r1, #0xA5   ; r0 = r1 with bits 0,2,5,and 7 cleared
ORR r0, r1, #0xA5   ; r0 = r1 with bits 0,2,5,and 7 set

CMP r5, r9          ; same as SUBS, but only affects APSR
CMN r0, r1          ; same as ADDS, but only affects APSR
TST r0, r1          ; same as ANDS, but only affects APSR
TEQ r0, r1          ; same as EORS, but only affects APSR
```

Single / Double Register Data Transfer

- Use to move data between one or two registers and memory

LDRD	STRD	Doubleword
LDR	STR	Word
LDRB	STRB	Byte
LDRH	STRH	Halfword
LDRSB		Signed byte load
LDRSH		Signed halfword load

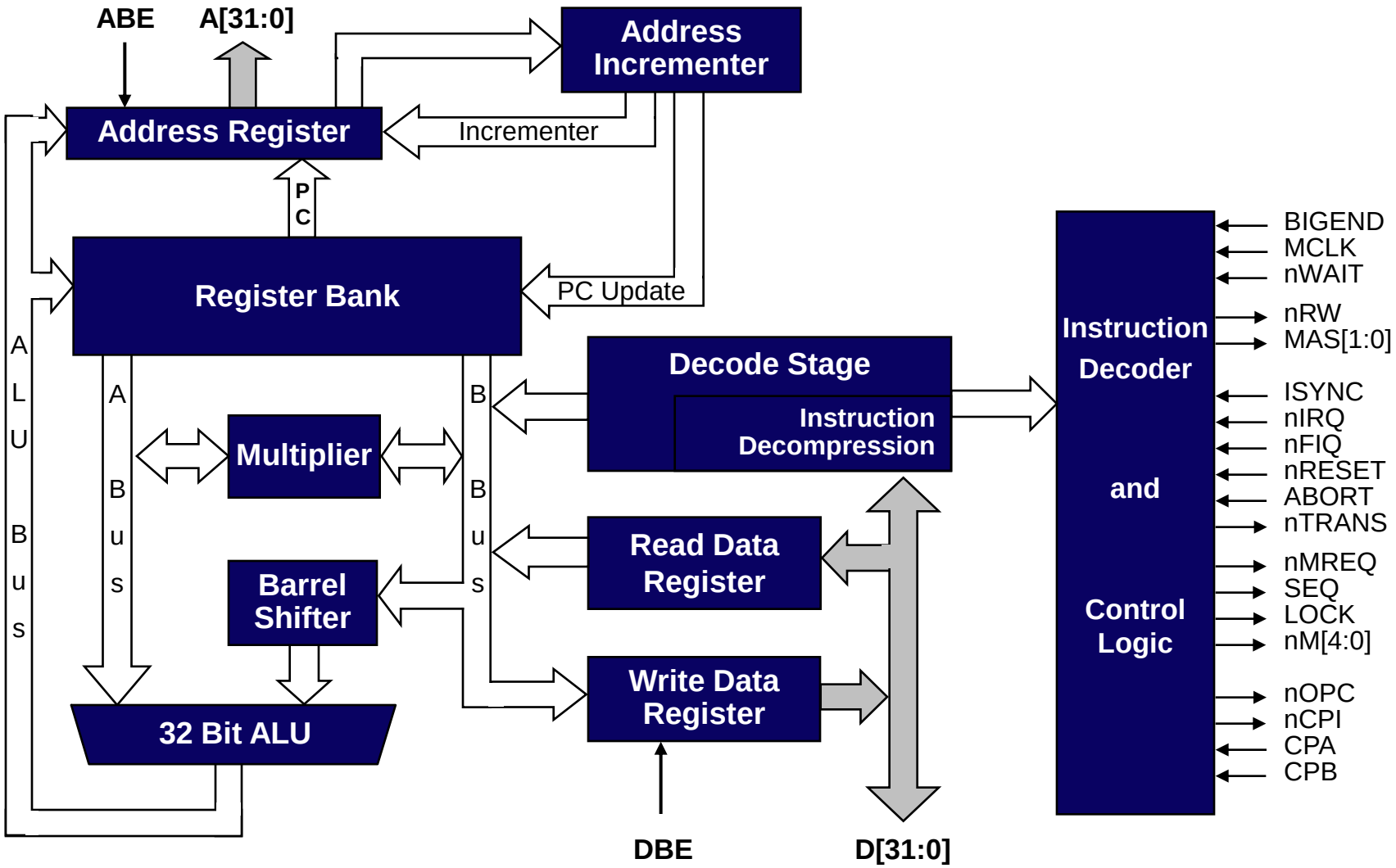


- Syntax
  - `LDR{<size>}{<cond>} Rd, <address>`
  - `STR{<size>}{<cond>} Rd, <address>`

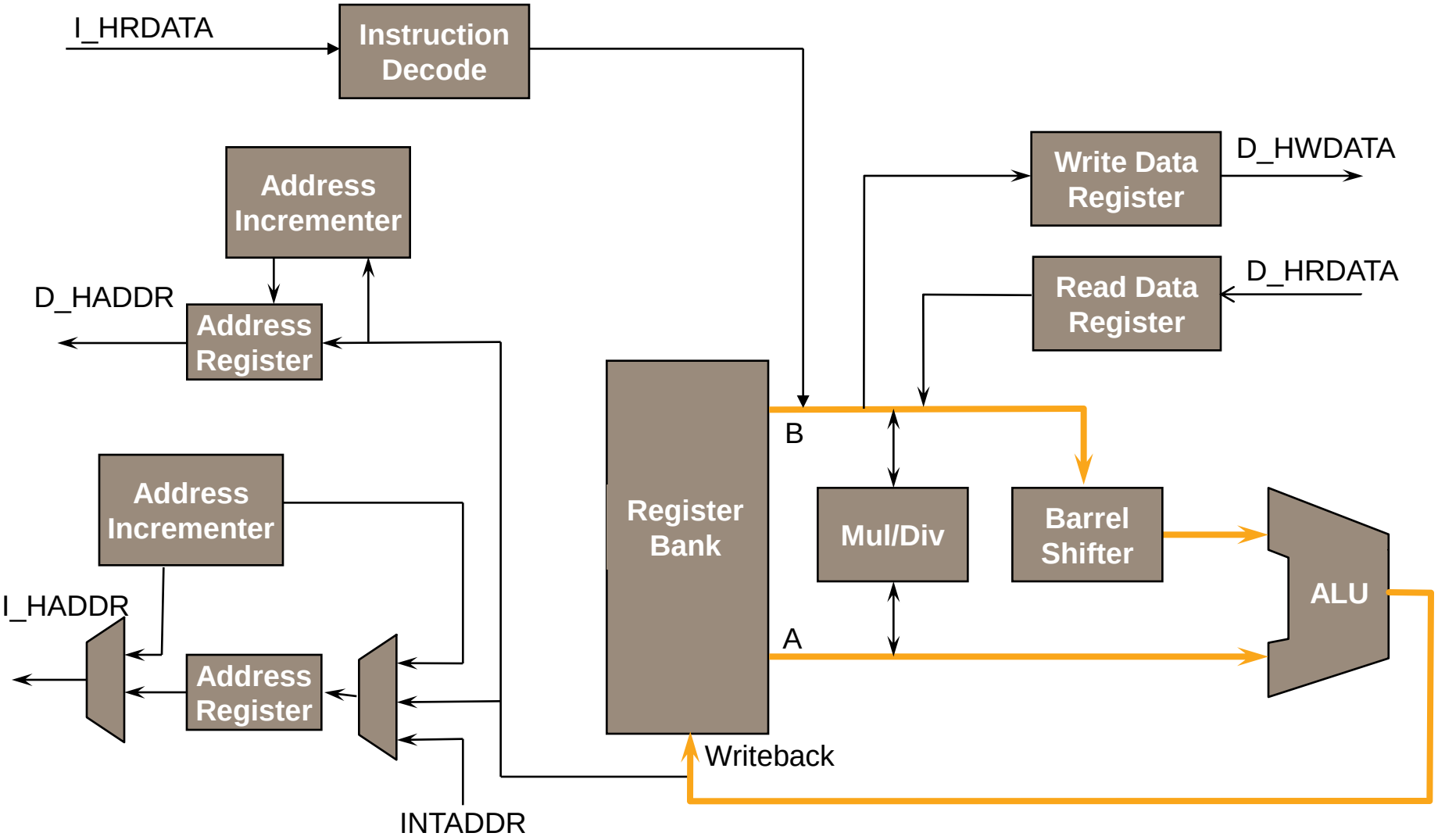
Agenda

- Introduction to ARM Ltd
- ARM Architecture/Programmers Model
- Data Path and Pipelines
- System Design
- Development Tools

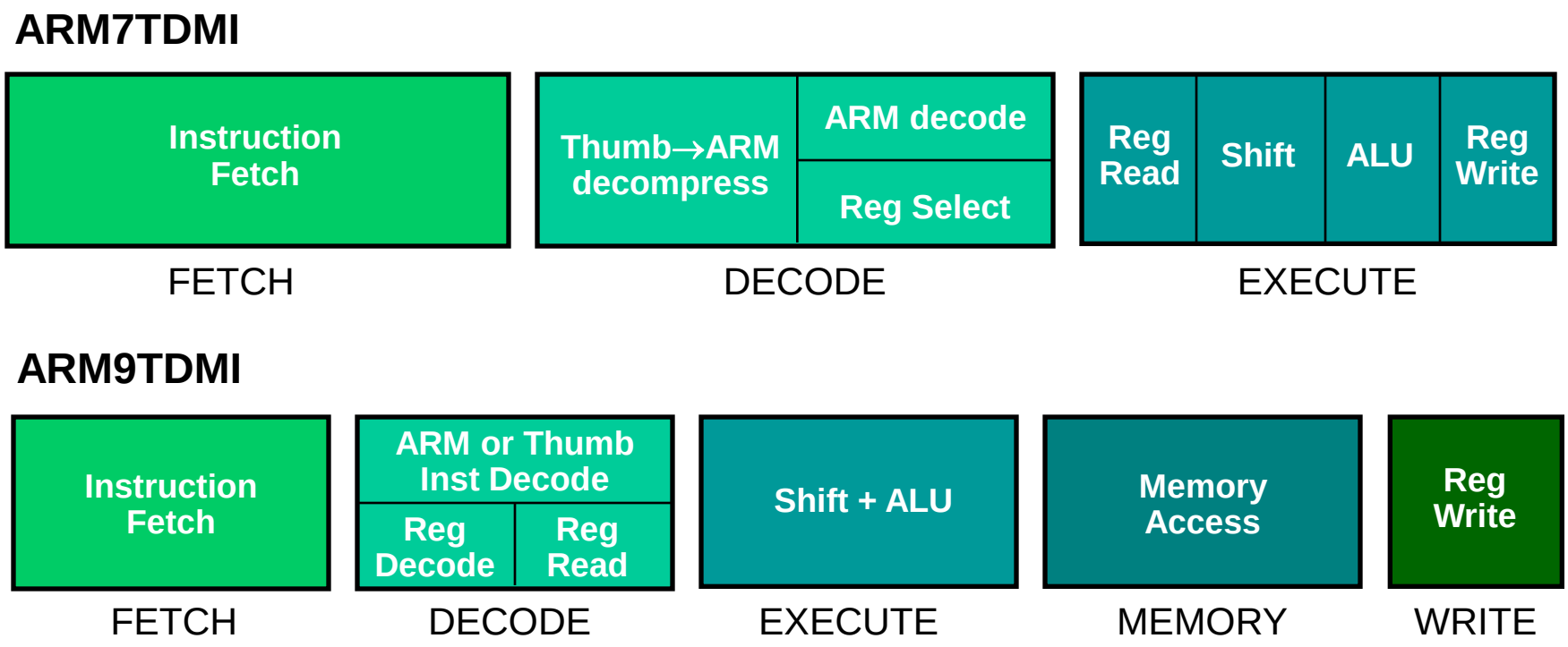
The ARM7TDM Core



Cortex-M3 Datapath

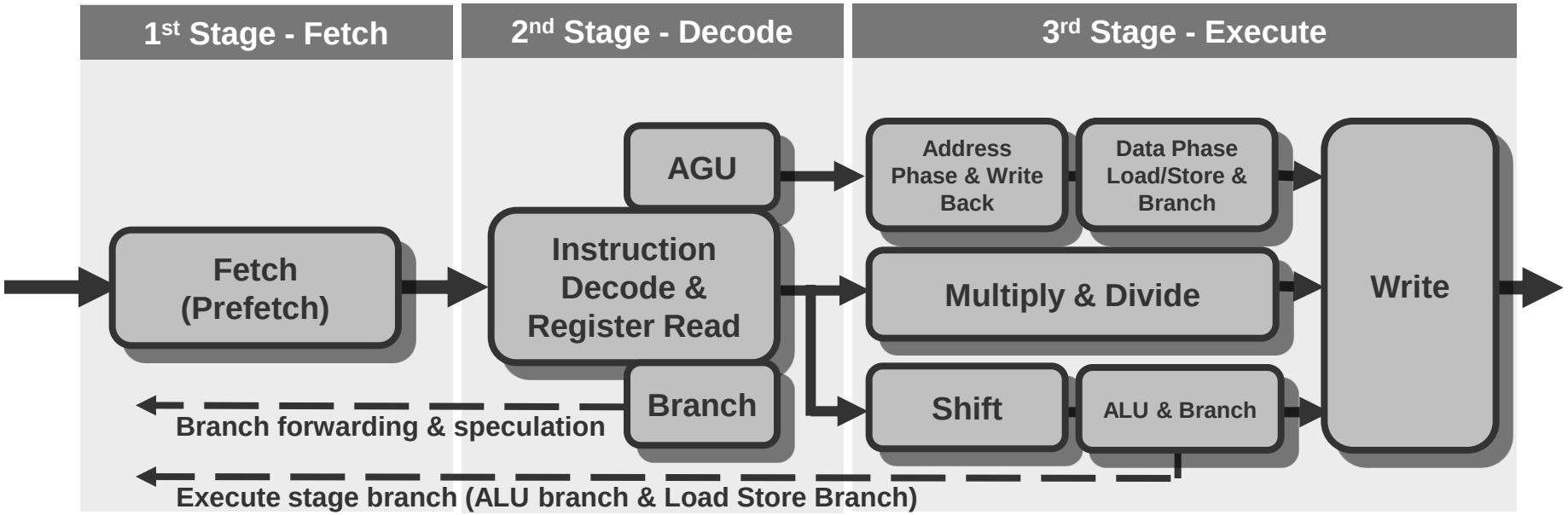


Pipeline changes for ARM9TDMI



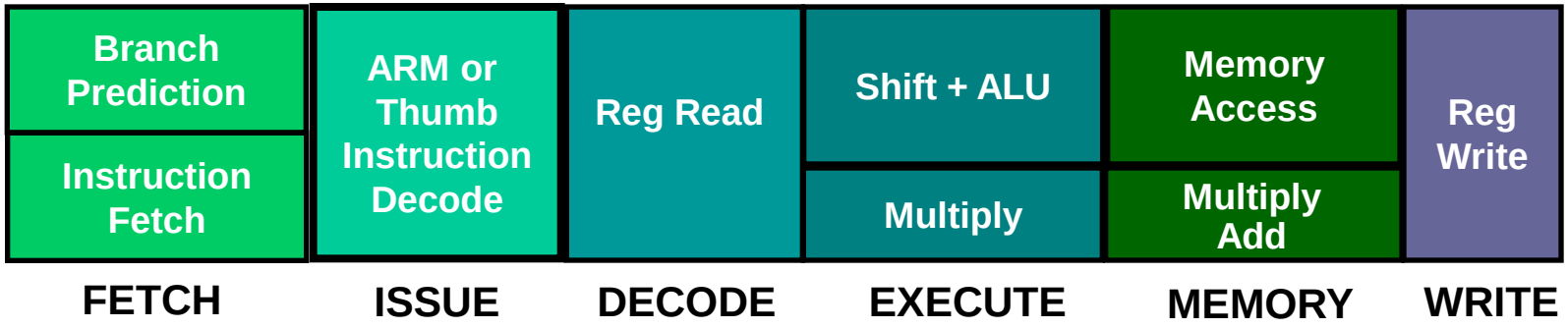
Cortex-M3 Pipeline

- Cortex-M3 has 3-stage fetch-decode-execute pipeline
  - Similar to ARM7
  - Cortex-M3 does more in each stage to increase overall performance

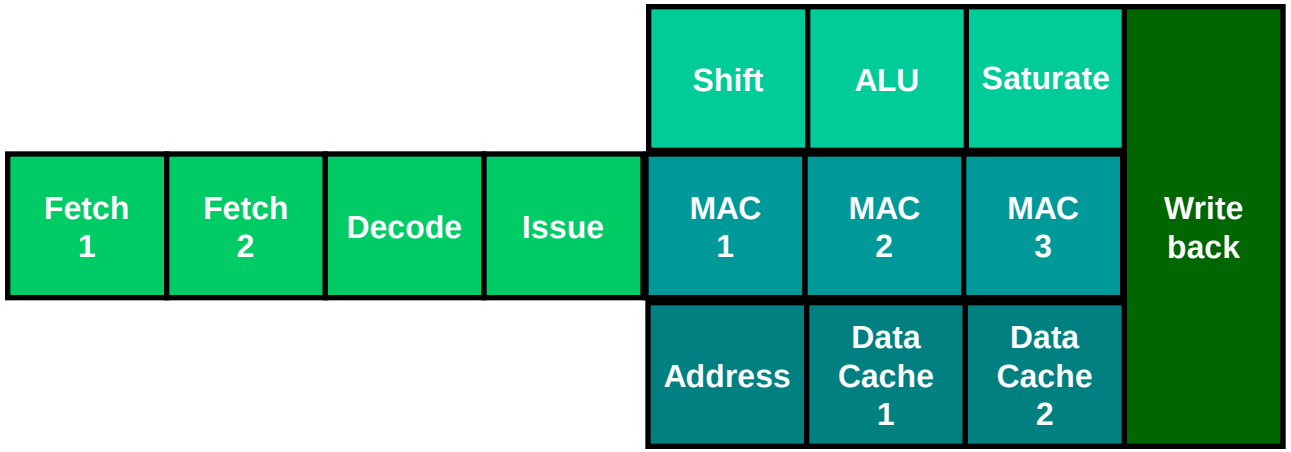


ARM10 vs. ARM11 Pipelines

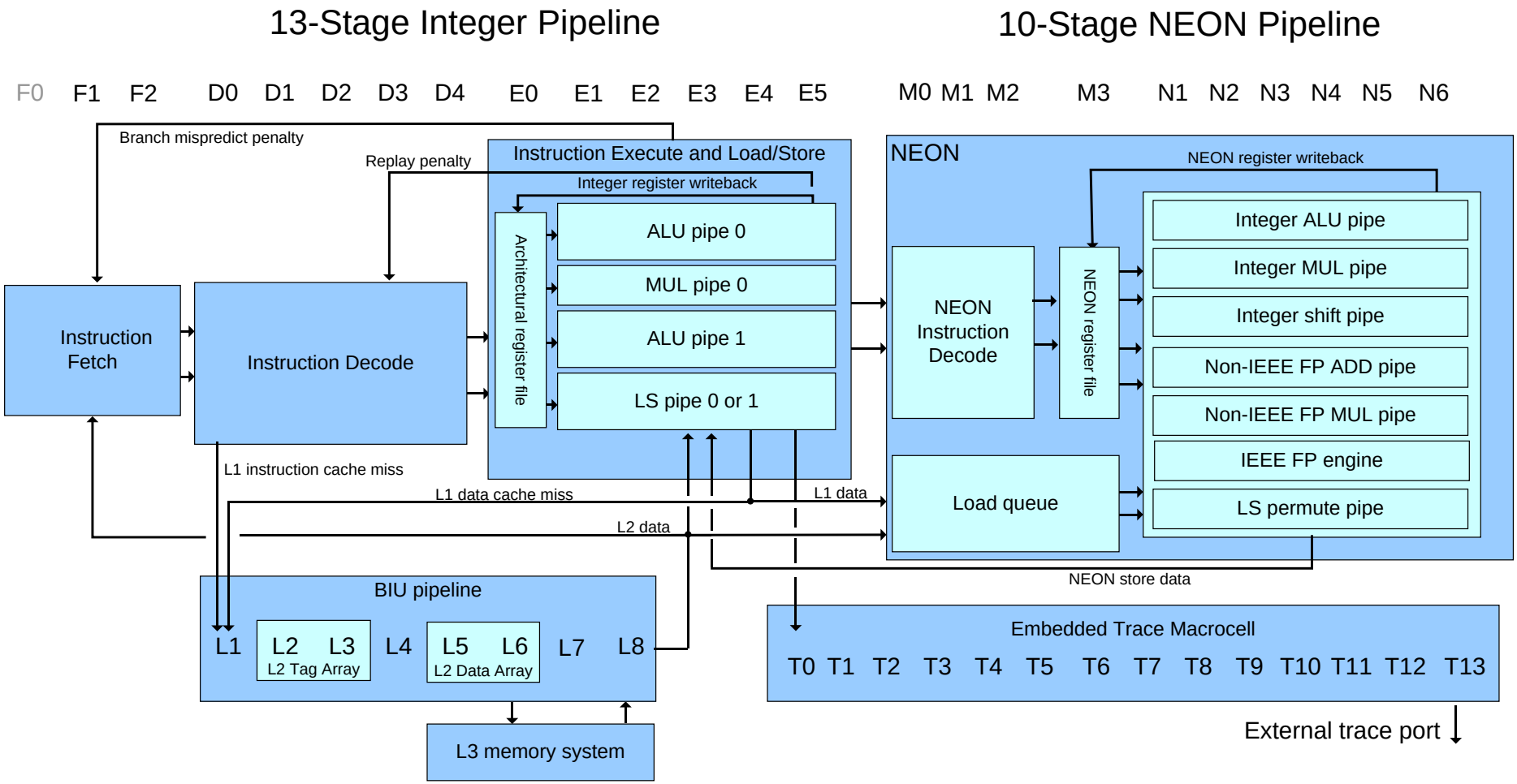
ARM10



ARM11



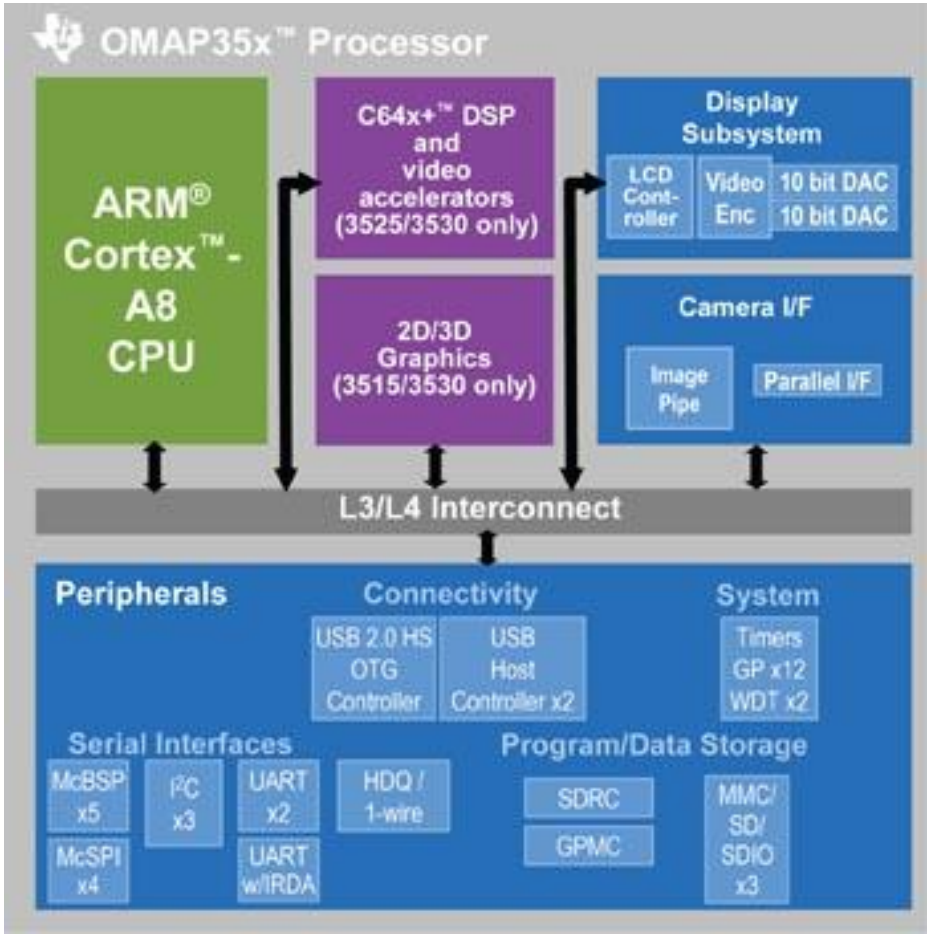
Full Cortex-A8 Pipeline Diagram



Agenda

- Introduction to ARM Ltd
- ARM Architecture/Programmers Model
- Data Path and Pipelines
- System Design
- Development Tools

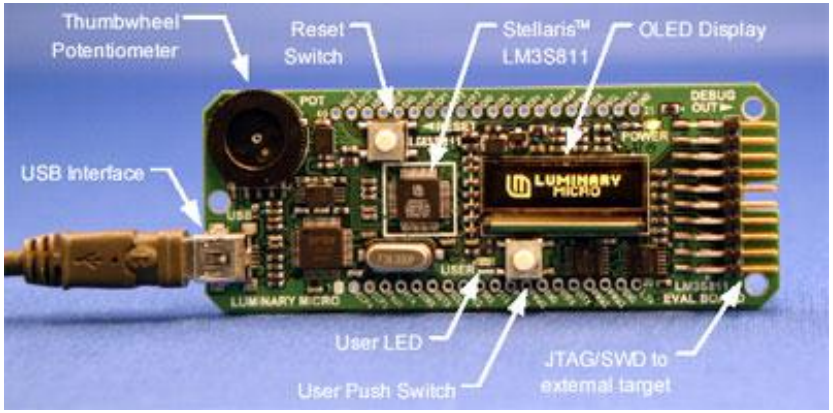
TI OMAP35X SoC

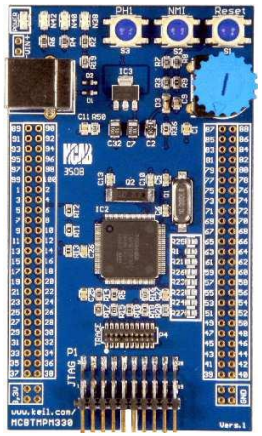


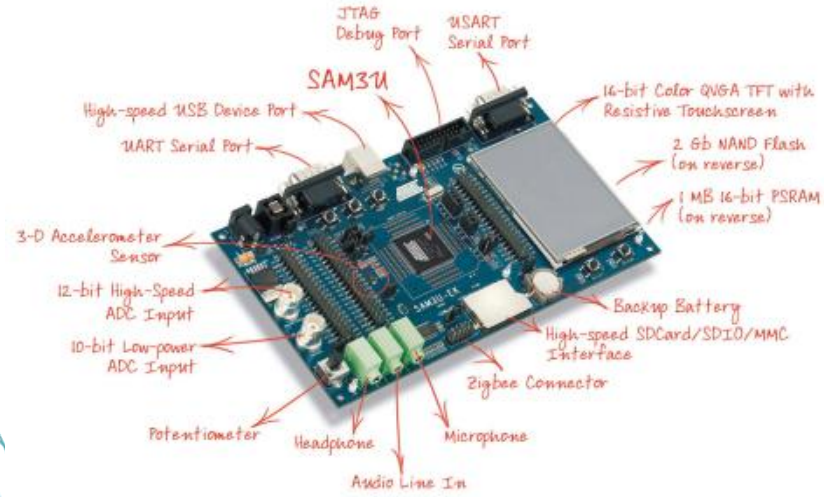
Agenda

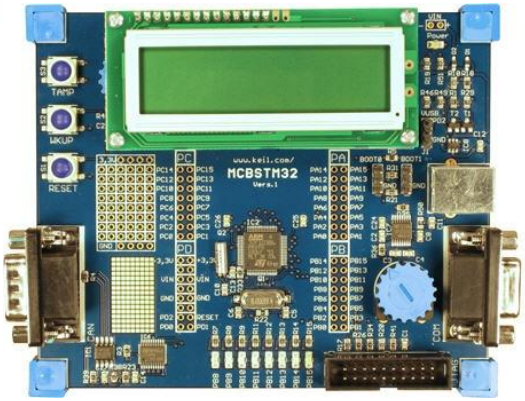
- Introduction to ARM Ltd
- ARM Architecture/Programmers Model
- Data Path and Pipelines
- System Design
- Development Tools

Development Platforms





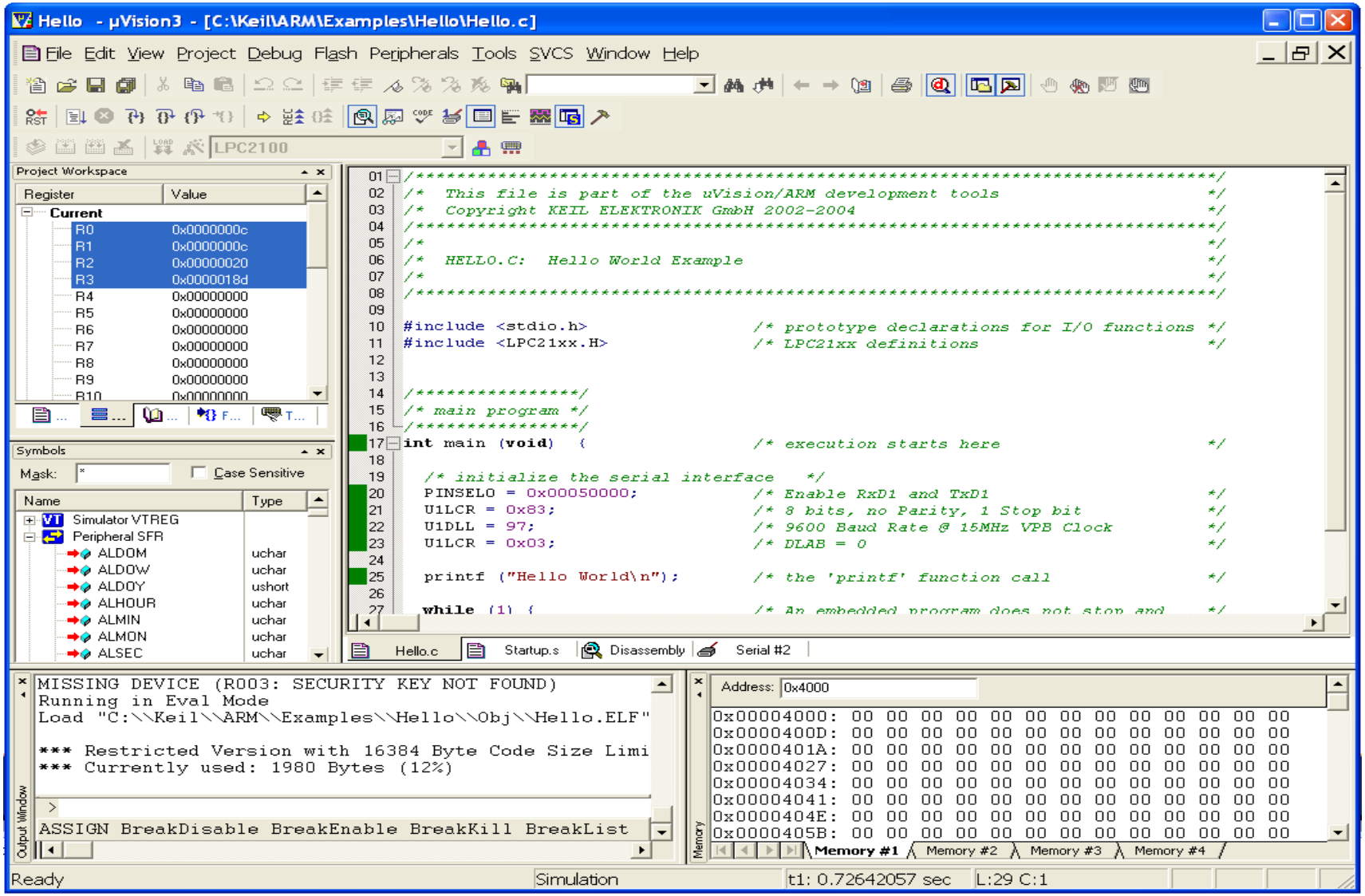




Keil Development Tools for ARM

- Includes ARM macro assembler, compilers (ARM RealView C/C++ Compiler, Keil CARM Compiler, or GNU compiler), ARM linker, Keil uVision Debugger and Keil uVision IDE
- Keil uVision Debugger accurately simulates on-chip peripherals (I^2C, CAN, UART, SPI, Interrupts, I/O Ports, A/D and D/A converters, PWM, etc.)
- Evaluation Limitations
  - 16K byte object code + 16K data limitation
  - Some linker restrictions such as base addresses for code/constants
  - GNU tools provided are not restricted in any way
- http://www.keil.com/demo/

Keil Development Tools for ARM



University Resources

- www.arm.com/university/
- University@arm.com

Your Future at ARM…

- ***Graduate and Internship/Co-op Opportunities***
  - Engineering: Memory, Validation, Performance, DFT, R&D, GPU and more!
  - Sales and Marketing: Corporate and Technical
  - Corporate: IT, Patents, Services (Training and Support), and Human Resources

- ***Incredible Culture and Comprehensive Benefit Package***
  - Competitive Reward
  - Work/Life Balance
  - Personal Development
  - Brilliant Minds and Innovative Solutions

- ***Keep in Touch!***
  - www.arm.com/about/careers

TI Panda Board

OMAP4430 Processor

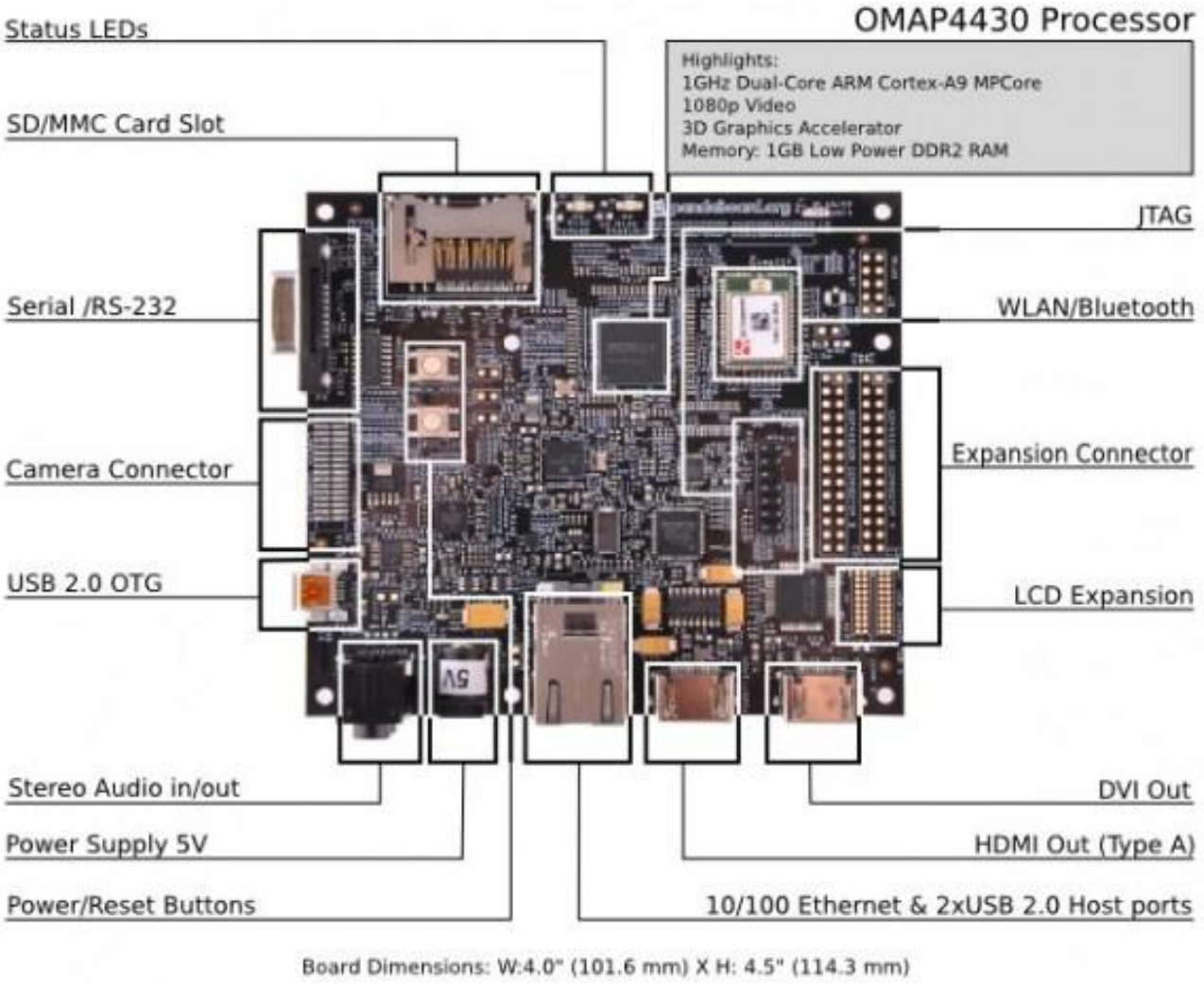
- 1 GHz Dual-core ARM Cortex-A9 (NEON+VFP)
- C64x+ DSP
- PowerVR SGX 3D GPU
- 1080p Video Support

POP Memory

- 1 GB LPDDR2 RAM

USB Powered

- < 4W max consumption (OMAP small % of that)
- Many adapter options (Car, wall, battery, solar, ..)



Fin

Nokia N95 Multimedia Computer

OMAP™ 2420 Applications Processor
ARM1136™ processor-based SoC, developed using Magma ®
Blast® family and winner of 2005 INSIGHT Award for 'Most Innovative SoC'

Symbian OS™ v9.2
Operating System supporting ARM processor-based mobile devices, developed using ARM® RealView® Compilation Tools

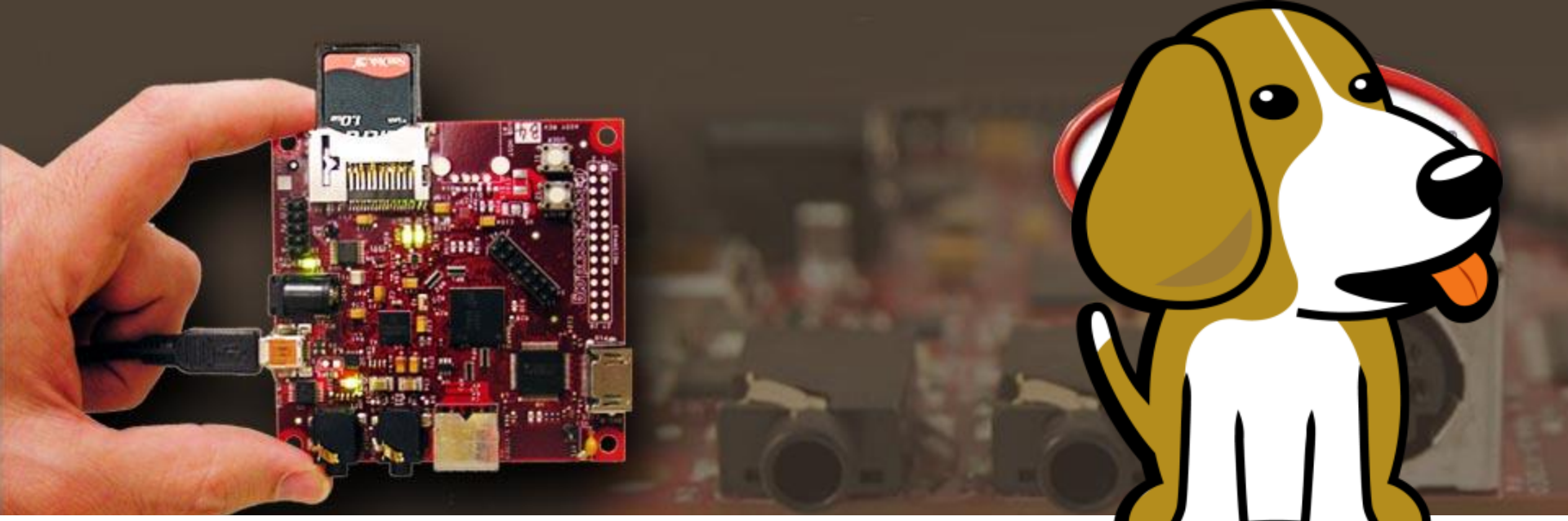
S60™ 3rd Edition
S60 Platform supporting ARM processor-based mobile devices

Mobiclip™ Video Codec
Software video codec for ARM processor-based mobile devices

ST WLAN Solution
Ultra-low power 802.11b/g WLAN chip with ARM9™ processor-based MAC

NOKIA
CONNECTING PEOPLE

Connect. Collaborate. Create.



Beagle Board

Targeting community development

$149

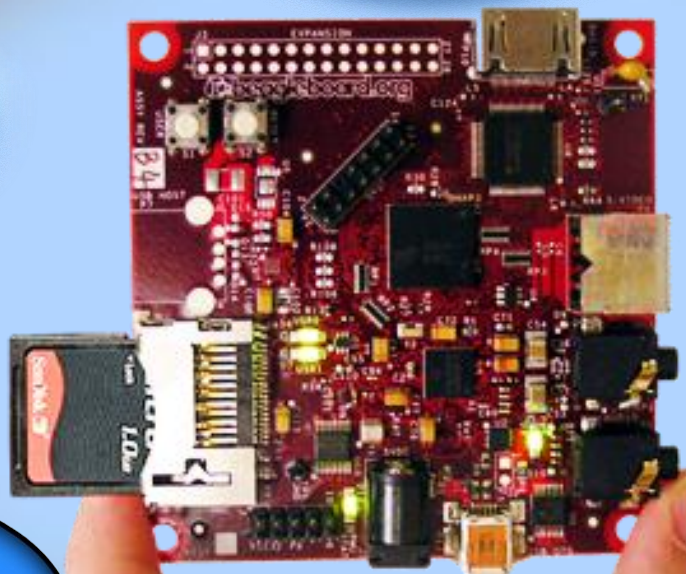
Personally affordable

Wikis, blogs, promotion of community activity

> 1000 participants and growing

Active & technical community

Freedom to innovate



Open access to hardware documentation

Instant access to >10 million lines of code

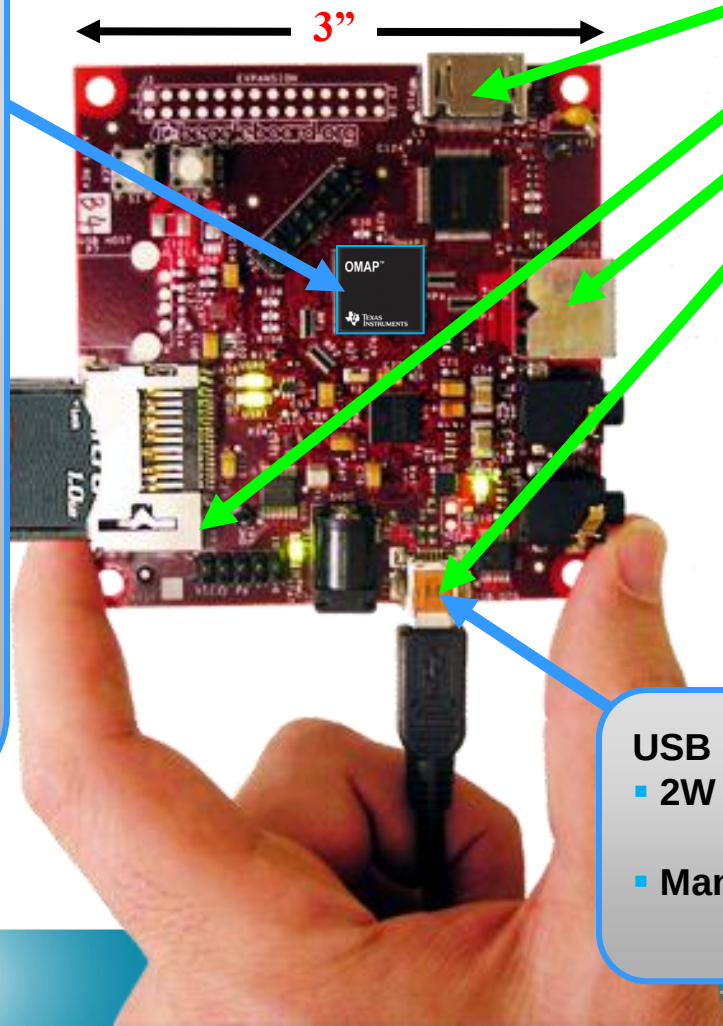
Opportunity to tinker and learn

Free software

Fast, low power, flexible expansion

OMAP3530 Processor
- 600MHz Cortex-A8
  - NEON+VFPv3
  - 16KB/16KB L1$
  - 256KB L2$
- 430MHz C64x+ DSP
  - 32K/32K L1$
  - 48K L1D
  - 32K L2
- PowerVR SGX GPU
- 64K on-chip RAM

POP Memory
- 128MB LPDDR RAM
- 256MB NAND flash



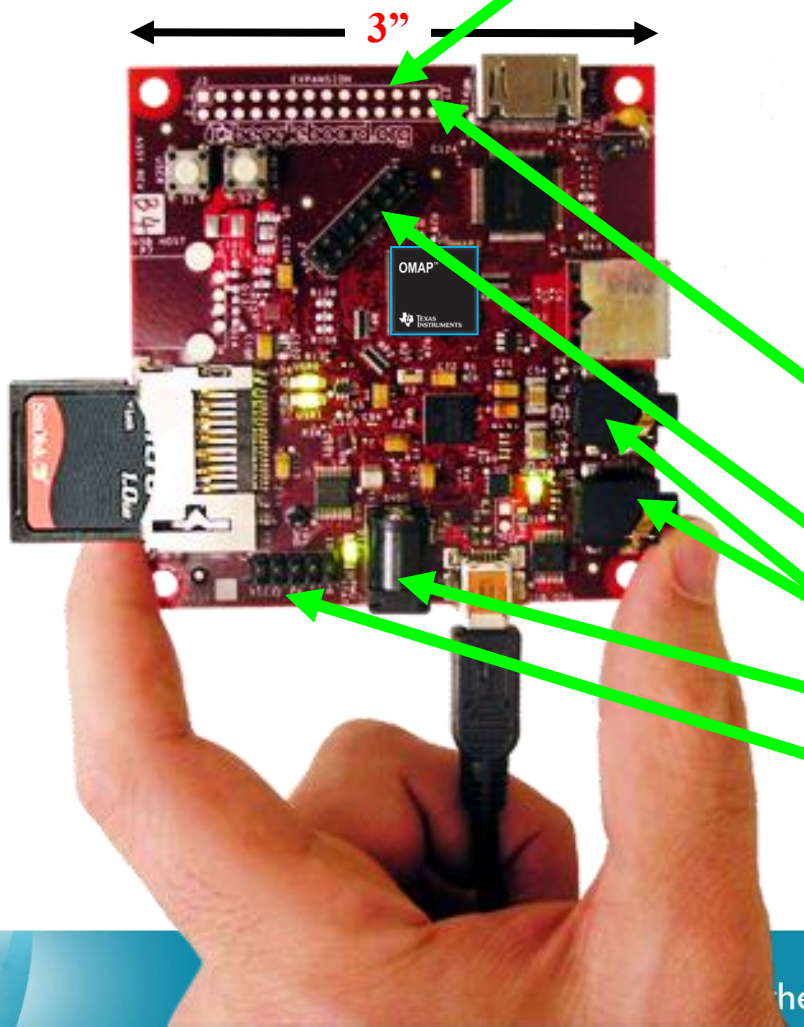
Peripheral I/O
- DVI-D video out
- SD/MMC+
- S-Video out
- USB 2.0 HS OTG
- I^2C, I^2S, SPI, MMC/SD
- JTAG
- Stereo in/out
- Alternate power
- RS-232 serial

USB Powered
- 2W maximum consumption
  - OMAP is small % of that
- Many adapter options
  - Car, wall, battery, solar, …

And more…

On-going collaboration at BeagleBoard.org
- Live chat via IRC for 24/7 community support
- Links to software projects to download

Other Features
- 4 LEDs
  - USR0
  - USR1
  - PMU_STAT
  - PWR
- 2 buttons
  - USER
  - RESET
- 4 boot sources
  - SD/MMC
  - NAND flash
  - USB
  - Serial



Peripheral I/O
- DVI-D video out
- SD/MMC+
- S-Video out
- USB HS OTG
- I^2C, I^2S, SPI, MMC/SD
- JTAG
- Stereo in/out
- Alternate power
- RS-232 serial

Project Ideas Using Beagle

- **OS Projects**
  - OS porting to ARM/Cortex (TI OMAP)
  - MythTV system
  - “Super-Beagle” – stack of Beagles as compute engine and task distribution
  - Linux applications

- **NEON Optimization Projects**
  - Codec optimization in ffmpeg (pick your favorite codec)
  - Voice and image recognition
  - Open-source Flash player optimizations (swfdec)