MIGRATING

FROM

8/16-BIT TO 32-BIT

ARM

MICROCONTROLLER

Embeded
Expert IO

Introduction

Purpose

To Provide Information Outlining Migration From 8-Bit/16-Bit Microcontroller To A 32-Bit ARM Platform.

Objectives

01 Compare traditional 8/16bit cores to ARM

02 Provides and overview of ARM7 architecture.

03 Discuss the similarities between 8-bit/16-bit and ARM MCUs

Embedded
Expert IO

ARM Archicture

01 The STM32F4 Is Based On Armv7e-M Architecture.

02 3-Stage Pipeline With Branch Speculation.

03. Instruction Set

- i Thumb-1
- ii Thumb-2
- iii 32-Bit Hardware Intergern Multiply With 32-Bit Or 64-Bit Results, Signed Or Unsigned.
- iv 32-Bit Hardware Integer Divider (2-12 Cycles).

04 1 to 240 interrupts, plus NMI (Non-maskable Interrupt).

05 12 Cycle Interrupt latency.

06 Integrated sleep modes.

3-Stage Instruction Pipeline

ARM	Thumb	
PC	PC	Fetch
PC-4	PC-2	Decode
PC-8	PC-4	Execute

Fetch

Instruction Fetched From Memory

Decode

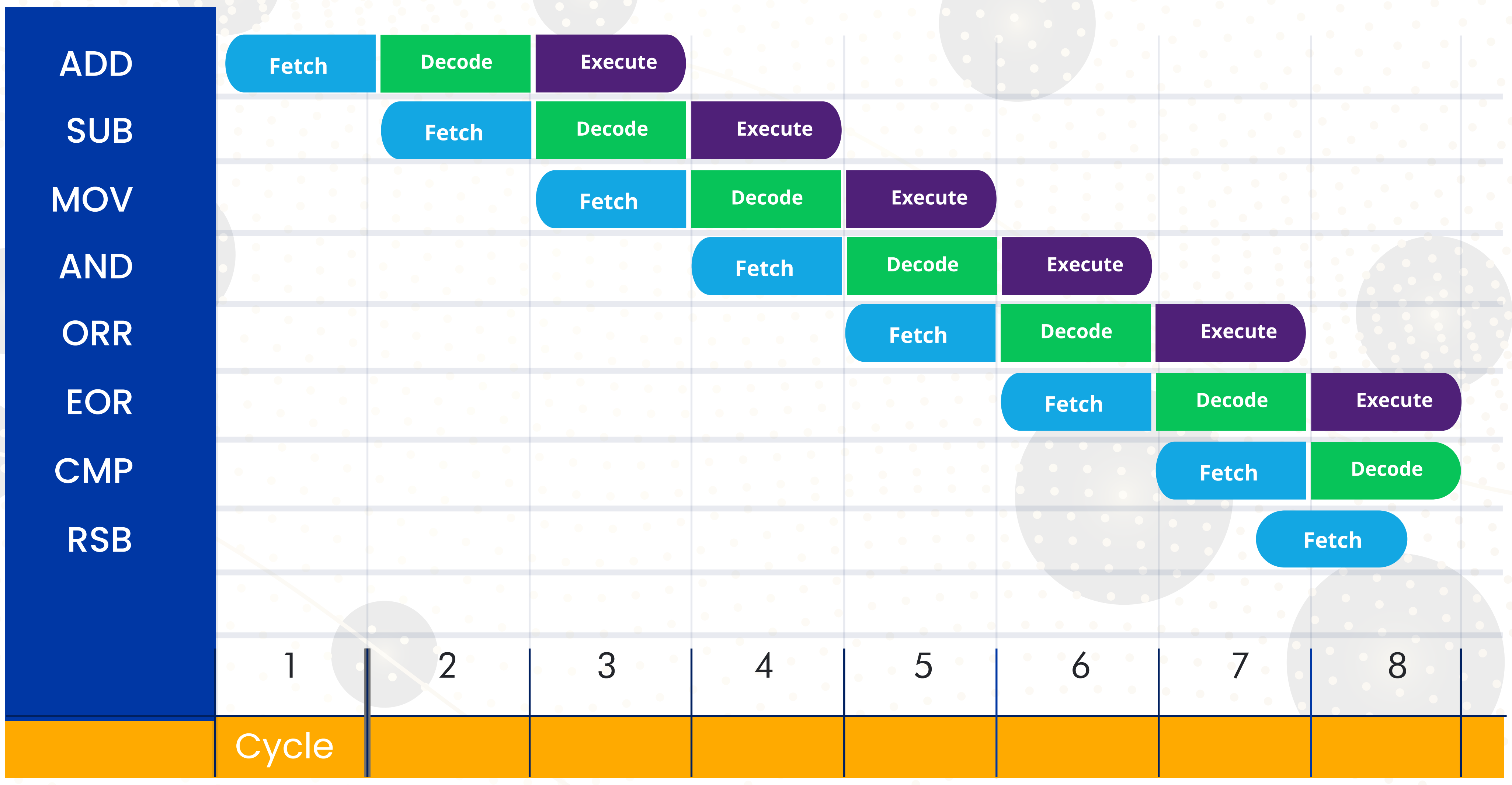
Thumb Only: Thumb Instruction decompressed To ARM instruction.
Instruction decoded.

Execute

Registers read from Register Bank, Shift and ALU operations performed, Registers written back to register bank.

Optimal Pipeline

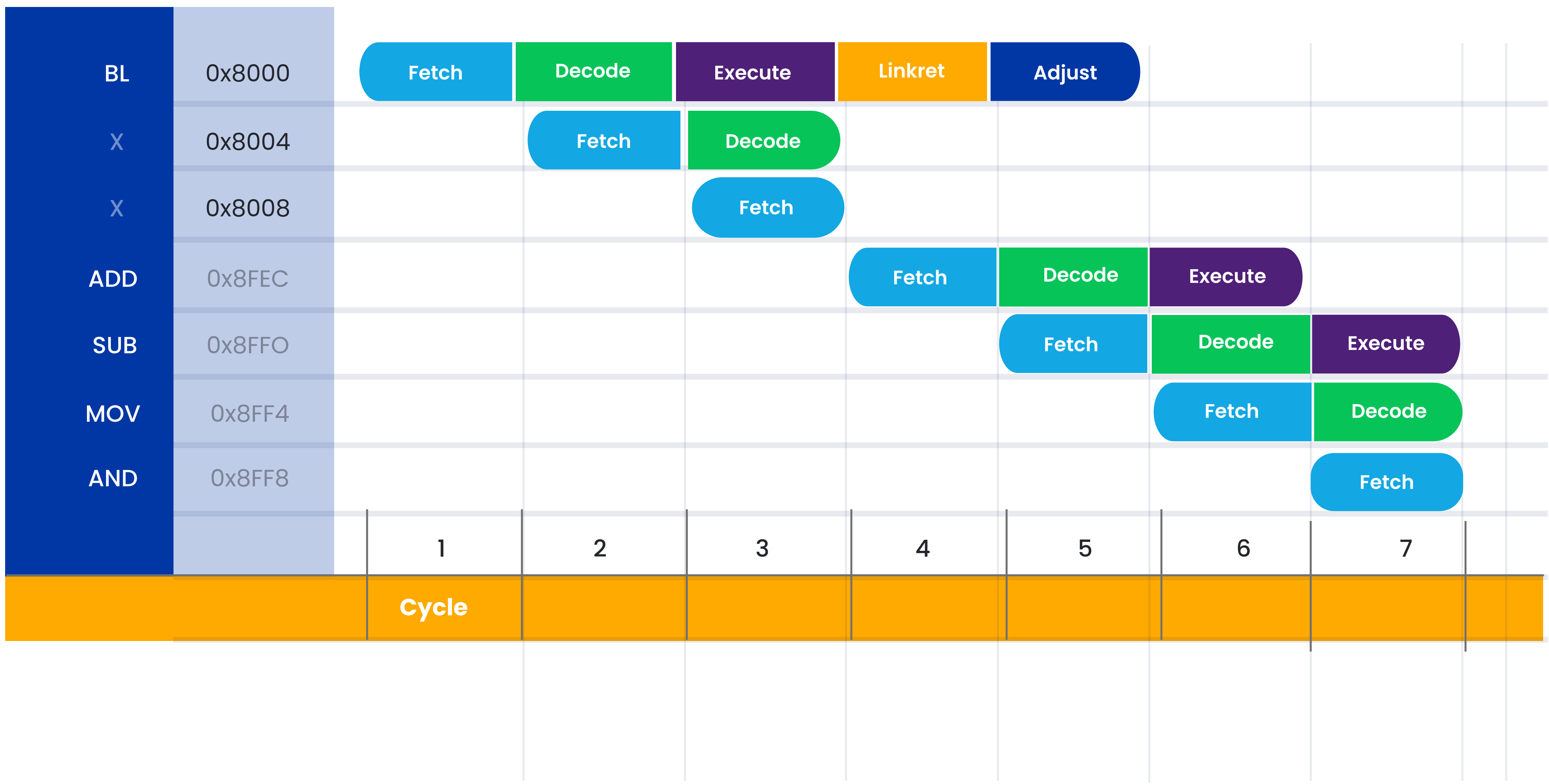
In this example it takes 6 clock cycles to execute 6 instructions.
All operations are in registers (single cycle instructions).
Clock cycles per instruction (CPI)=1



Cycle	1	2	3	4	5	6	7	8
ADD	Fetch	Decode	Execute					
SUB		Fetch	Decode	Execute				
MOV			Fetch	Decode	Execute			
AND				Fetch	Decode	Execute		
ORR					Fetch	Decode	Execute	
EOR						Fetch	Decode	Execute
CMP							Fetch	Decode
RSB								Fetch

Branch Pipeline Example

- Branches break the pipeline
- Example in ARM state



		1	2	3	4	5	6	7
BL	0x8000	Fetch	Decode	Execute	Linkret	Adjust		
X	0x8004		Fetch	Decode				
X	0x8008			Fetch				
ADD	0x8FEC				Fetch	Decode	Execute	
SUB	0x8FFO					Fetch	Decode	Execute
MOV	0x8FF4						Fetch	Decode
AND	0x8FF8							Fetch

Cycle

Typical 8/16 Bit Vs ARM

8/16-bit (CISC)	ARM (RISC)
No Instruction pipeline.	Instruction pipeline: ARM7-> 3stage ARM9 -> 5 stage ARM11-> 9 stages
One operating mode.	Multiple operating mode: User, FIQ, IRQ, Supervisor, Abort, Undef and System.
Single Instruction Type.	Multiple Instruction types: ARM and Thumb mode.
Simpler register organization.	More sophisticated register organization with shadow registers for various modes.

ARM Processor Modes

ARM has seven operating modes

i User: unprivileged mode under which most applications run.

ii FIR: entered, when a high priority (fast) interrupt.

iii IRQ: general purpose interrupt handling.

iv Supervisor: protected mode for the operating system.

v System: prevailed mode using same registers as user mode.

vi Abort: used to handle memory access violations.

vii Undefined: used to handle undefined functions.

Many 8/16-bit processors don't implement different modes of operation

ARM Instruction Set

01 All instruction are 32-bit long.

02 Many instructions execute in single cycle.

03 Most of the ARM instruction can be conditionally executed.

04 Could be divided into six broad classes of instruction:

- i Branch instructions.
- ii Data processing Instructions.
- iii Status register transfer instructions.
- iv Load and Store instructions.
- v Coprocessor instructions.
- vi Exception-generating instructions.

ARM7 Assembly

ARITHMETIC

-Add.
-Subtract.
-Negate.
-Multiply.
-Compare.

LOGICAL

-AND
-OR
-XOR
-Move Not
-Bit Clear

BRANCH

-Unconditional
-Conditional
-Branch and Exchange.
-Branch with Link and Exchange.

MOVE

-Not
-PSR to Register
-Register to PSR
-Immediate to Register.

LOAD/STORE

-Word
-Byte
-Halfword
-Doubleword
-Multiple

PUSH/POP

-Push with link
-Pop and return
-Pop and return with exchange

Thumb State

01 ARM uses a 32-bit architecture with a subset of 16-bit instructions, still using 32-bit data and registers.

02 Set of instructions re-coded into 16-bits.

- i Improve code desity by ~30%.
- ii Saveing program memory space.

03 In Thumb state only the program code is 16-bit wide.

- i After fetching the 16-bit instruction from memory, they are decompressed to 32-bit instruction before they are decoded and executed.
- ii All operation are still 32-bit operation.

Embedded
Expert IO

ARM Thumb Instruction Set

ARITHMETIC

-Add : ADD, ADC (low/high, immediate, to SP, w/ carry)

-Subtract : SUB, SBC (Immediate W/Carry)

-Negate : NEG

-Multiply :MUL

-Compare : CMP, CMN (low/ high, immediate, negative)

LOGICAL

Test : TST

AND : AND

OR : ORR

XOR : EOR

Bit clear : BIC

SOFTWARE INTERRUPT(SWI)

BRANCHING

-Branch (unconditional) : B

-Branch and Exchange : BX

-Branch with Link : BL

-Conditional (14 variants)

PUSH (PUSH)

Registers onto stack (LR, too)

POP (POP)

Registers from stack (PC, too)

MOVE

MOV, MVN

- Immediate

-Low/ High

- Move Not

LOAD*/ STORE

Word (immediate/register offset)

Byte (signed*, immediate/ register offset)

Halfword (signed*/immediate register offset)

Conditional (14 variants)

Multiple

Relative (PC* or SP)

Address* (using PC or SP)

SHIFT

LSL, LSR, ASR

Left/Right

Arithmetic/right

Rotate(ROR)

Right

Registers In Thumb State

01 The thumb state register set is a subset of the ARM state set. The programmer has direct access to :-

Eight General Registers	R0-R7
The Program Counter	PC
Stack Pointer	LC
Link Register	LR
Current Program Status Register	CPSR

02 In thumb state, the higher registers (r8-r12) are not part of the standard register set. The assembly language

programmer has limited access to them, but can use them for fast temporary storage.

Interrupt Handler Consideration

8/16-BIT (CISC)

- Normally, only one interrupt "type"

- Limited or no interrupt priorization

- No change in code type to service ISRs

ARM (RISC)

-Three types of interrupt
- Fast Interrupt
- Interrupt Request
- Non-Vectored Interrupt

-Detailed control over priority.
- FIQ always executed first
- IRQ priority levels (0-15)
- Default interrupts lowest priority

-Interrupt Service Routines must be written in ARM mode.

Vectored Interrupt Controll

01 ARM Processor.

02 32 interrupts request inputs.

03 16 IRQ interrupt can be auto-vectored.

04 Single instruction vectoring to ISR.

05 Dynamic software priority assignment.

06 16 Non-vectored interrupts.

07 Software Interrupts.

08 FIQ- Fast interrupt.

VIC-FIQ (Fast Interrupt) Details

01 FIQ have higher priority than IRQs

- i Service first
- ii FIQs disable IRQ

02 FIQ vector is last vector table (Allow handler to run sequentially from that address).

03 FIQ mode has 5 extra banked registers r8-r12 (interrupt must always preserve non-banked registers).

Is It Time To Move To ARM?

01 Small 48-pin and 64-pin QFN packages available.

02 Prices competitive with 8 or 16-bit architectures.

03 Low cost development tools are available.

Microcontroller Users Want

01 Many customers are using **8/16-bit** device today want :

i Easy upgrade path with minimal relearning.

ii Single chip solution:

- On chip flash
- Timers, PWM, UART, SPI, I2C
- ADC
- Predicable, deterministic performance
- No cache
- Same performance between flash and memory

02 Low power for battery operated devices.

03 Fast IO.

04 High performance, good quality and high reliable flash memory.

05 Hence, ARM is suitable choice in those scenarios.

Typical 8/16 Bit Vs ARM

Parameters	ATMEGA328p	STM32F103C8T6
GPIO	23	37
U(S)ART	1	3
SPI	1	2
I2C	1	2
16-bit timer	1	3
ADC	1 8-channels	2-9 Channels
RTC	0	1
Maximum Clock speed	20MHZ	72MHz
PWM	6	16
Interrupt pins	2	16
Others	-	USB, CAN, DMA 5V tolarence IO

Embedded
Expert IO

This report is brought to you by EmbeddedExpertIO. EmbeddedExpertIO is an online embedded systems school focused on professional embedded systems software programming.

If you are new to embedded systems programming our community provides step-by-step courses that will take you from "blinky" to "build your own Rtos".

If you are an embedded systems developer who wants to specialize in some specific aspect of embedded systems programming, We also provide a wide range of specialization courses to help you master different aspects of embedded firmware development.
We look forward to welcoming you to EmbeddedExpertIO.
Visit us at : https://study.embeddedexpert.io/

Thank You

Embedded
Expert IO