

# ARM

## RISC Architecture Introduction

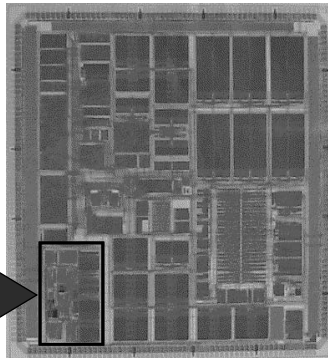
Case study: ARM7TDMI Processor Core

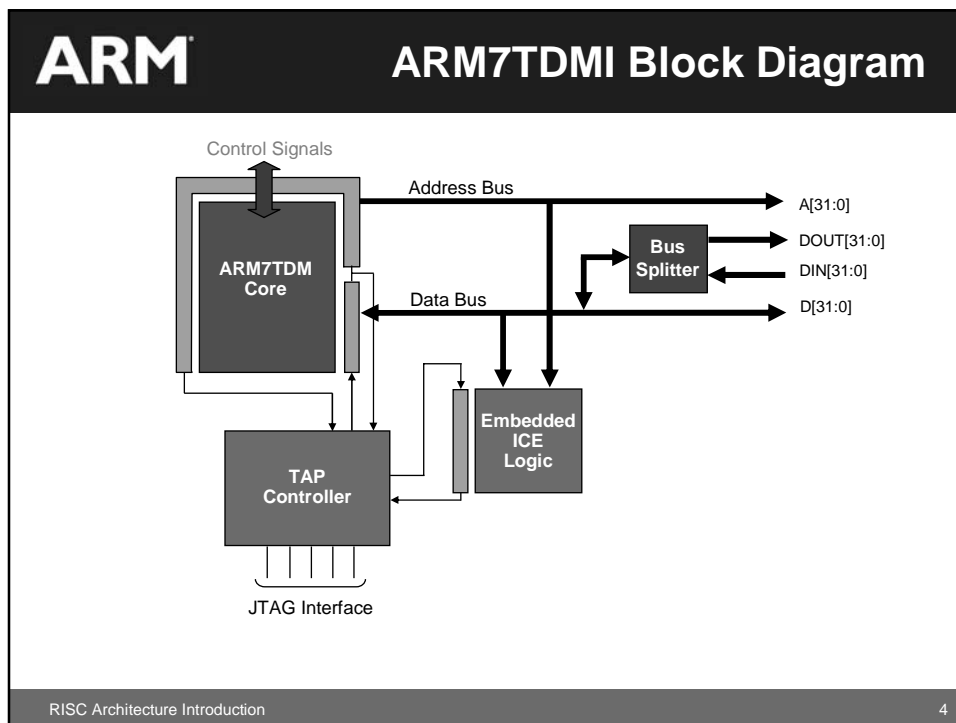
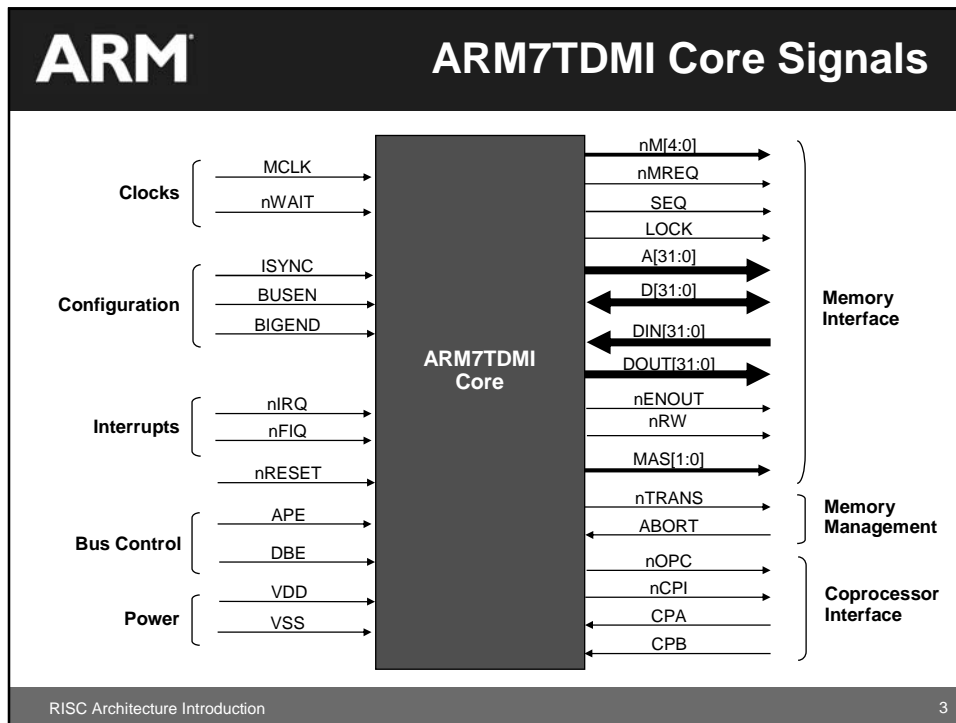
T H E A R C H I T E C T U R E F O R T H E D I G I T A L W O R L D

# ARM

## Embedded microprocessors

Microprocessor







## 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.**



## Single register data transfer

<b>LDR</b>	<b>STR</b>	Word
<b>LDRB</b>	<b>STRB</b>	Byte
<b>LDRH</b>	<b>STRH</b>	Halfword
<b>LDRSB</b>		Signed byte load
<b>LDRSH</b>		Signed halfword load

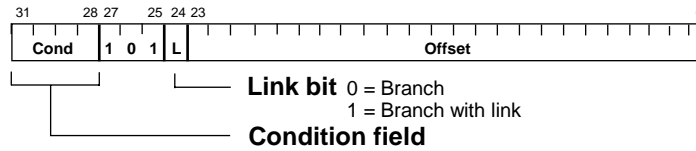
- **Memory system must support all access sizes**
- **Syntax:**
  - **LDR{<cond>}{<size>} Rd, <address>**
  - **STR{<cond>}{<size>} Rd, <address>**

e.g. LDREQB

# ARM

## Branch instructions

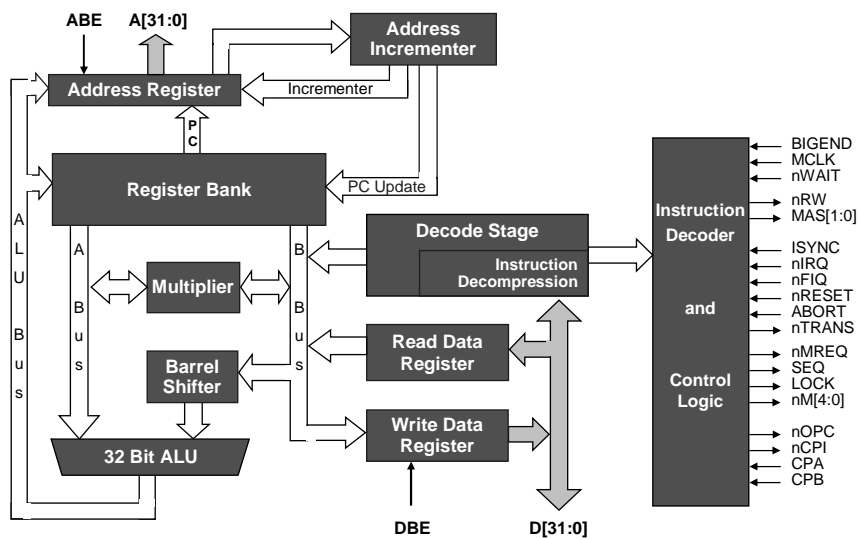
- Branch : B{<cond>} label
- Branch with Link : BL{<cond>} subroutine\_label



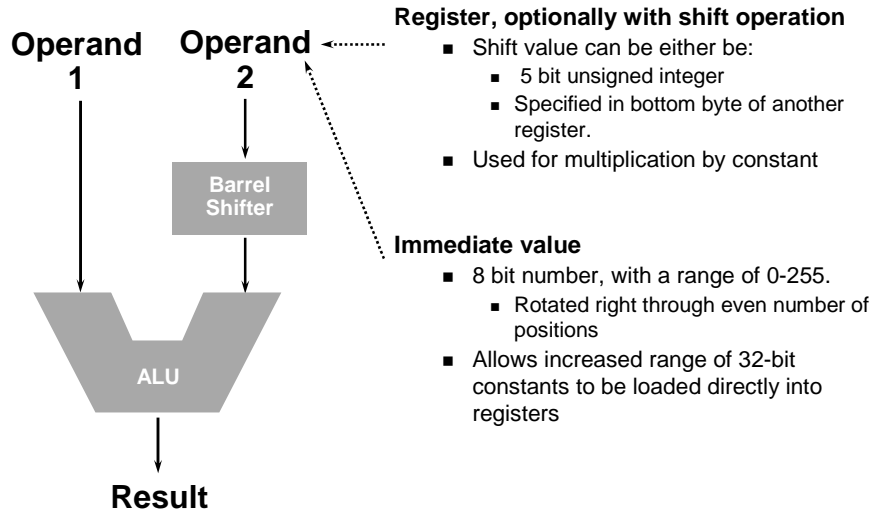
- The processor core shifts the offset field left by 2 positions, sign-extends it and adds it to the PC
  - ± 32 Mbyte range
  - How to perform longer branches?

# ARM

## The ARM7TDM Core



## Using the Barrel Shifter: The Second Operand



## Exception Handling

- When an exception occurs, the ARM:**
  - Copies CPSR into SPSR\_<mode>
  - Sets appropriate CPSR bits
    - Change to ARM state
    - Change to exception mode
    - Disable interrupts (if appropriate)
  - Stores the return address in LR\_<mode>
  - Sets PC to vector address
- To return, exception handler needs to:**
  - Restore CPSR from SPSR\_<mode>
  - Restore PC from LR\_<mode>

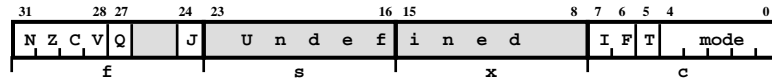
**This can only be done in ARM state.**

	⋮
0x1C	FIQ
0x18	IRQ
0x14	(Reserved)
0x10	Data Abort
0x0C	Prefetch Abort
0x08	Software Interrupt
0x04	Undefined Instruction
0x00	Reset

**Vector Table**

Vector table can be at  
**0xFFFF0000** on ARM720T  
 and on ARM9/10 family devices

# ARM 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

# ARM 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
    
```

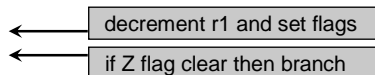
```

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".

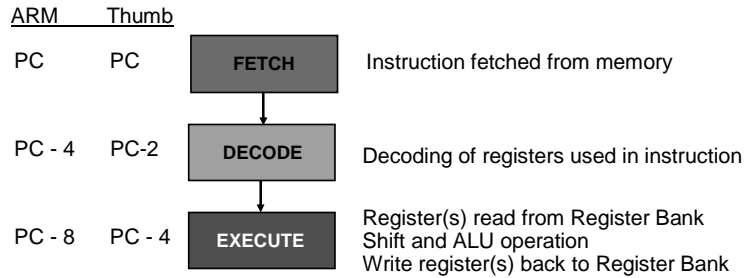
```

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



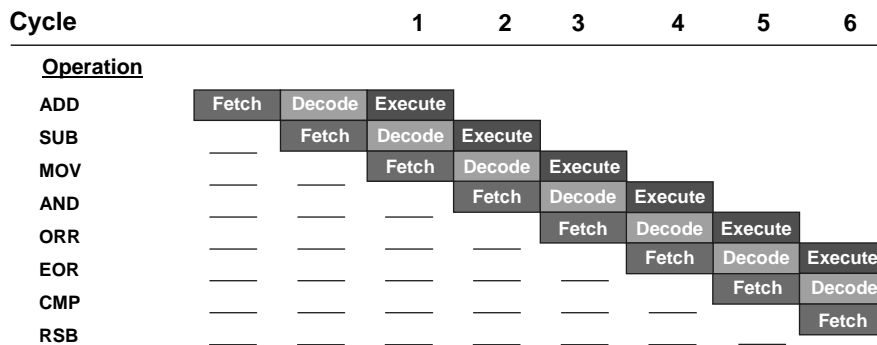
# ARM The Instruction Pipeline

- The ARM7 family uses a 3 stage pipeline in order to increase the speed of the flow of instructions to the processor.
  - Allows several operations to be undertaken simultaneously, rather than serially.



- The PC points to the instruction being fetched, not the instruction being executed.

# ARM Optimal Pipelining



- In this example it takes 6 clock cycles to execute 6 instructions
- All operations here are on registers ( single cycle execution )
- Clock cycles per Instruction (CPI) = 1



## LDR Pipeline Example

Cycle	1	2	3	4	5	6
<u>Operation</u>						
ADD	Fetch	Decode	Execute			
SUB		Fetch	Decode	Execute		
LDR			Fetch	Decode	Execute	Data Writeback
MOV				Fetch	Decode	
AND					Fetch	
ORR						Fetch

- In this example it takes 6 clock cycles to execute 4 instructions
- Clock cycles per Instruction (CPI) = 1.5



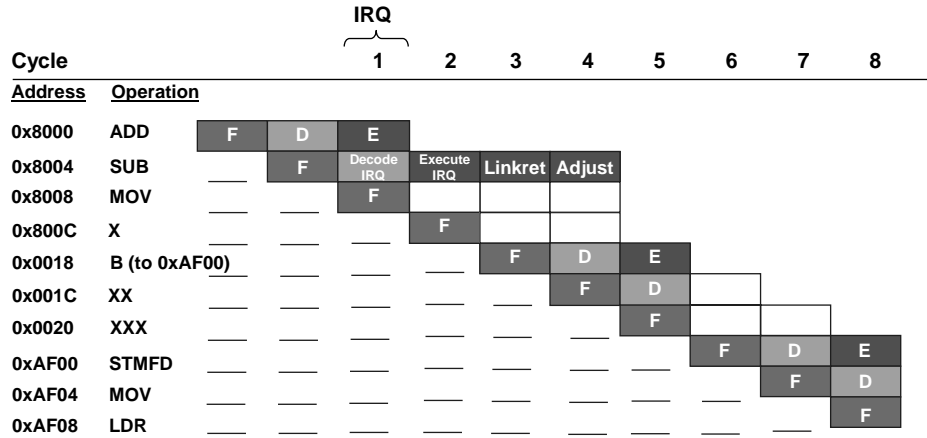
## Branch Pipeline Example

Cycle	1	2	3	4	5
<u>Address</u> <u>Operation</u>					
0x8000   BL	Fetch	Decode	Execute	Linkret	Adjust
0x8004   X		Fetch	Decode		
0x8008   XX			Fetch		
0x8FEC   ADD				Fetch	Decode
0x8FF0   SUB					Fetch
0x8FF4   MOV					

- Breaking the pipeline
- Note that the core is executing in ARM state

# ARM

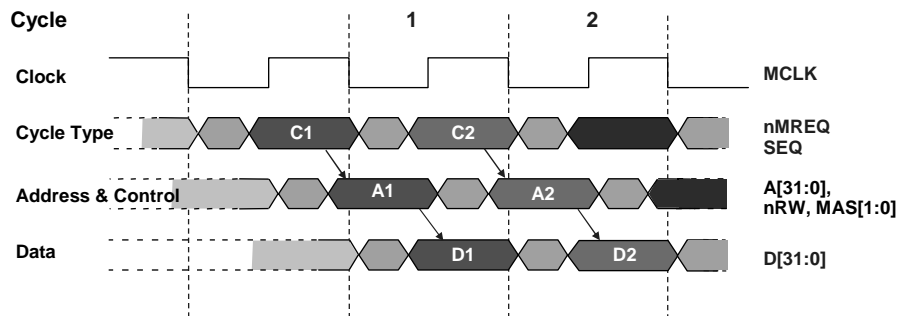
## Interrupt Pipeline Example



- IRQ interrupt minimum latency = 7 cycles

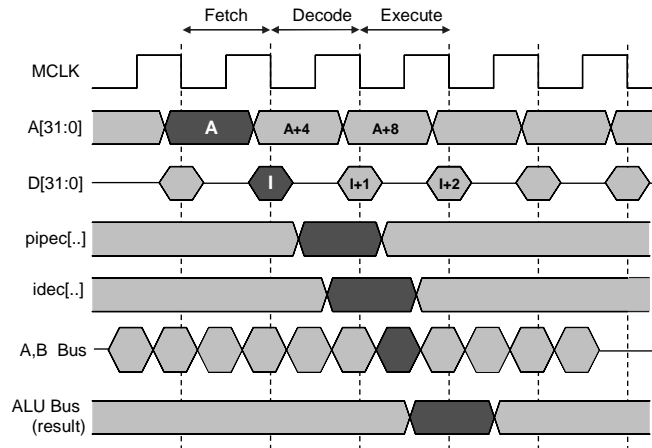
# ARM

## ARM Bus Pipelining



ARM

## Execution Example



ARM

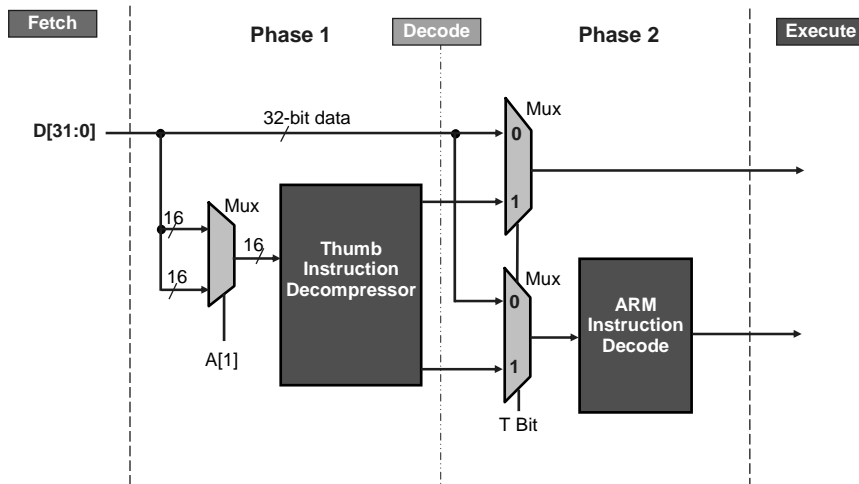
## ARM Memory Interface

The ARM memory interface has four basic cycle types:

- **Sequential.**
  - The ARM core requests a transfer to or from an address which is either the same, or one word or one half-word greater than the preceding address.
- **Non-sequential.**
  - The ARM core requests a transfer to or from an address which is unrelated to the address used in the preceding cycle.
- **Internal.**
  - The ARM core does not require a transfer, as it is performing an internal function, and no useful prefetching can be performed at the same time.
- **Coprocessor register transfer.**
  - The ARM core wishes to use the data bus to communicate with a coprocessor, but does not require any action by the memory system.

**ARM**

## The Effect of the T-bit



**ARM**

## Longer Pipelining

Cycle		1	2	3	4	5	6	7	8	9
Operation										
ADD R1, R1, R2		F	D	E	W					
SUB R3, R4, R1			F	D	E	W				
LDR R4, [R7]			F	D	E	M	W			
AND R6, R3, R1				F	D	E	W			
ORR R8, R3, R4					F	D	E	W		
EOR R3, R1, R2						F	D	E	W	

F - Fetch D - Decode E - Execute I - Interlock M - Memory W - Writeback

- In this example it takes 6 cycles to execute 6 instructions, CPI of 1.
- The LDR instruction does not cause the pipeline to interlock

**ARM**

**Questions or comments:  
university@arm.com**