MANCHEstER
1824

The University
of Manchester

ARM integer cores

- Outline:
  - the ARM 3-stage pipeline
  - the ARM7TDMI core
  - the ARM 5-stage pipeline
  - the ARM9TDMI core
  - the ARM10TDMI core
  - StrongARM & XScale
  - the ARM11 core
  - Cortex
  - ☞ hands-on: system software – interrupts

ARM integer cores

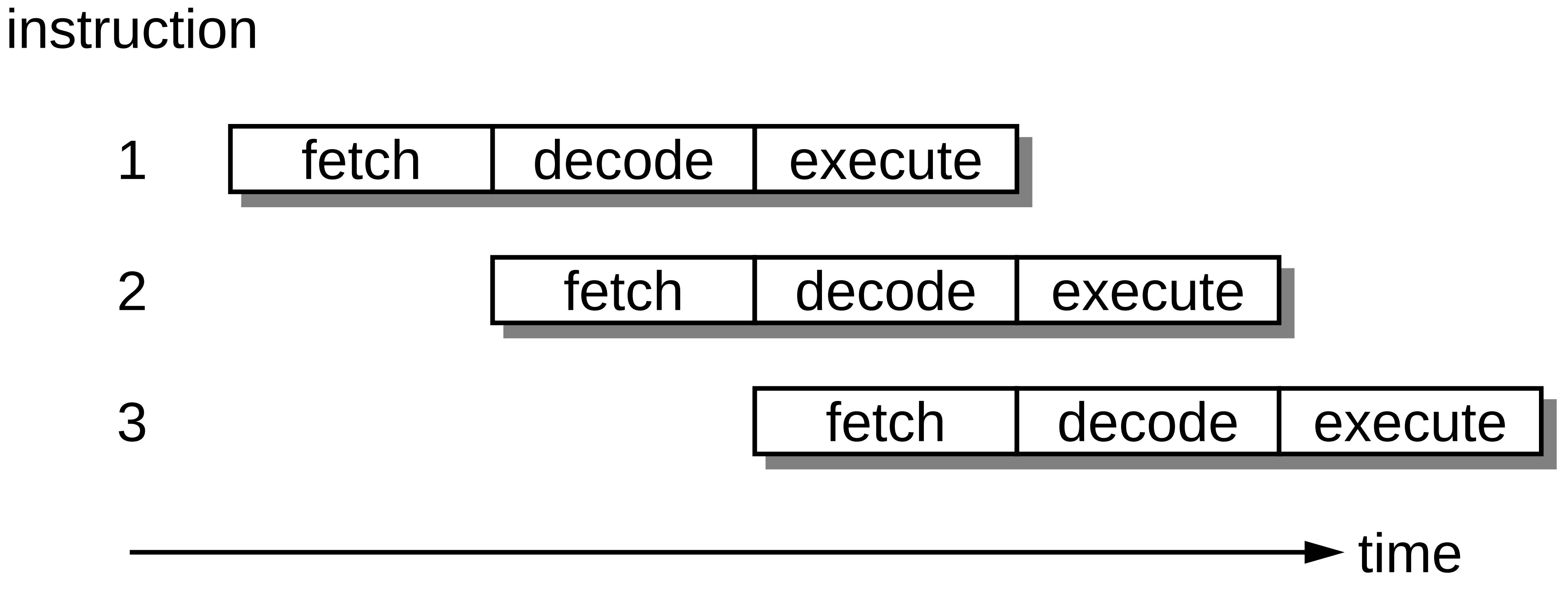
- Outline:
  - ➜ **the ARM 3-stage pipeline**
  - ❍ the ARM7TDMI core
  - ❍ the ARM 5-stage pipeline
  - ❍ the ARM9TDMI core
  - ❍ the ARM10TDMI core
  - ❍ StrongARM & XScale
  - ❍ the ARM11 core
  - ❍ Cortex
  - ☞ hands-on: system software – interrupts

The 3-stage ARM pipeline

(The original architecture which has affected on the instruction set.)

- fetch
  - the instruction is fetched from memory
- decode
  - the instruction is decoded and the datapath control signals prepared for the next cycle
- execute
  - the operands are read from the register bank, shifted, combined in the ALU and the result written back

The 3-stage ARM pipeline



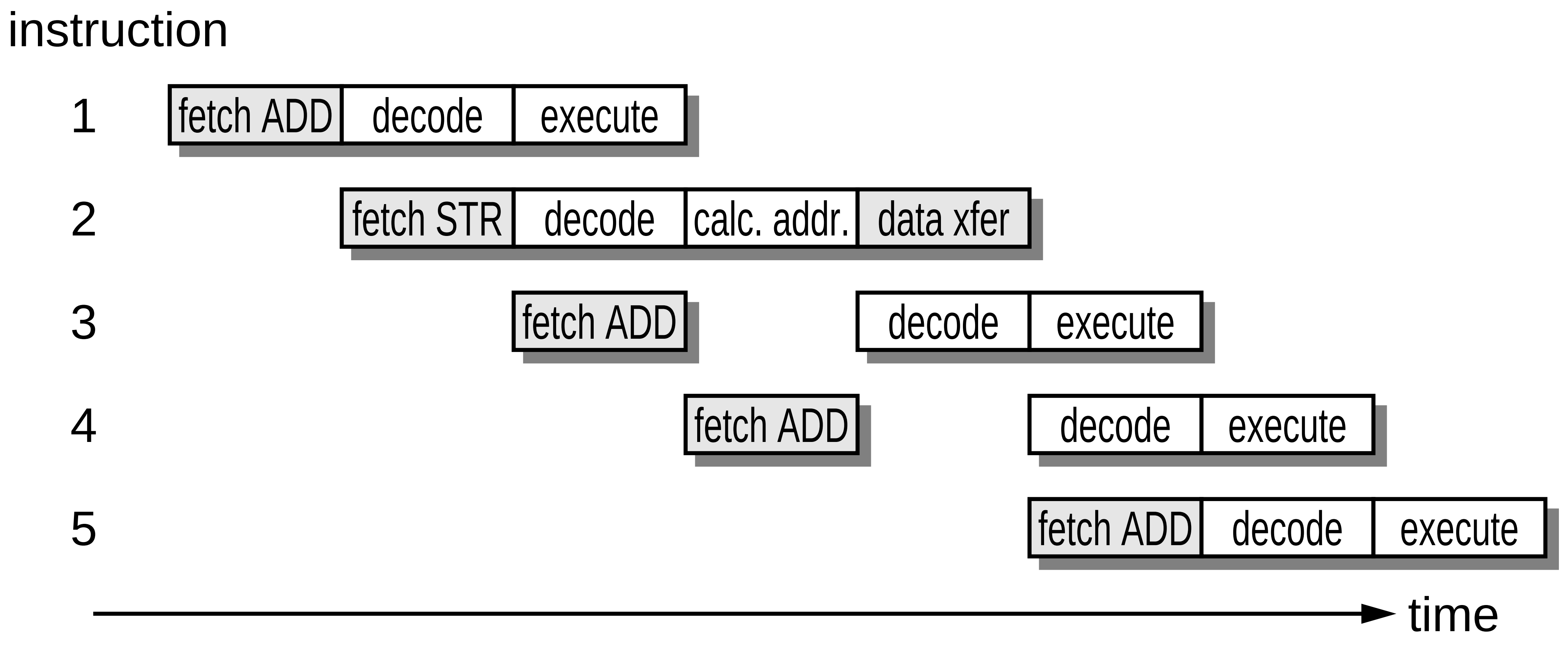
- Single cycle instructions
  - complete at a rate of one per clock cycle
  - intended to use (a single) memory efficiently

MANCHEstER 1824

The University of Manchester

The 3-stage ARM pipeline

- More complex instructions:



  - ‘STR’ causes a stall while transfer occurs

MANCHEStER 1824

The University of Manchester

The 3-stage ARM pipeline

- PC behaviour
  - r15 increments twice before an instruction executes
    - due to pipeline operation
  - therefore r15 = address of instruction + 8
    - (+12 if used after first cycle, though this is architecturally undefined)
    - in Thumb code the offset is +4
  - normally the assembler makes the necessary adjustments, e.g. in branches

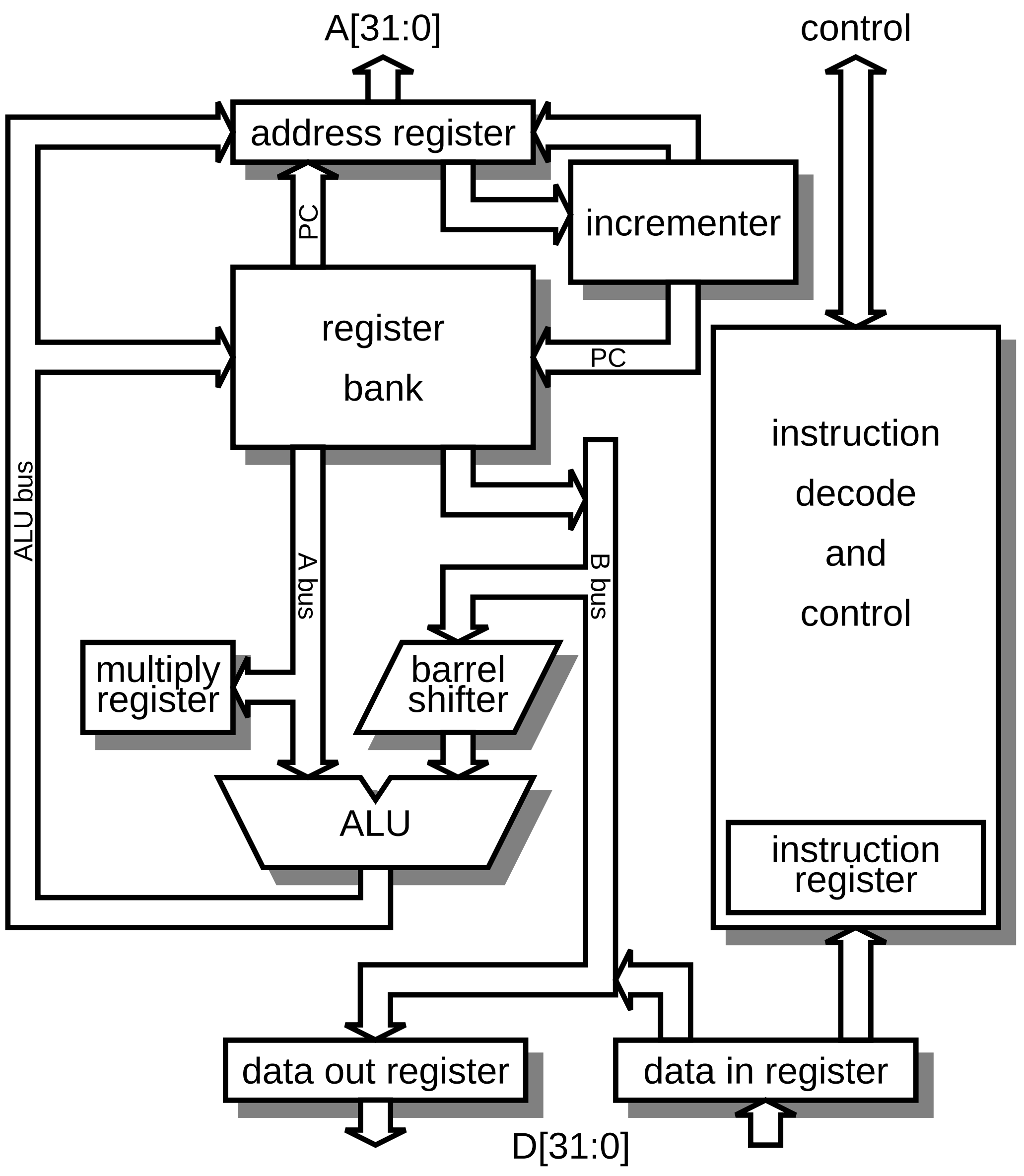
This behaviour is **consistent for all ARMs**, although the pipeline structures may vary.

3-stage ARM organization

- ARM components:
  - register bank
    - 2 read ports, 1 write port
    - plus additional read and write ports for r15
  - barrel shifter
  - ALU
  - address register and incrementer
  - memory data registers
  - instruction decoder and control

3-stage ARM organization

- Separate address incrementer
- Two register read ports
- Barrel shifter in series with ALU



Why does this matter?

The internal structure of the processor can result in pipeline stalls ($\Rightarrow$ loss of performance) in some circumstances.

- Instruction dependencies
  - pipeline needs flushing and refilling when a branch is taken
  - alleviated by branch prediction
- Data dependencies
  - instruction may have to wait for the result from a previous one
  - alleviated by forwarding
  - particular problem with loads (memory is long latency)
    - code reordering can help

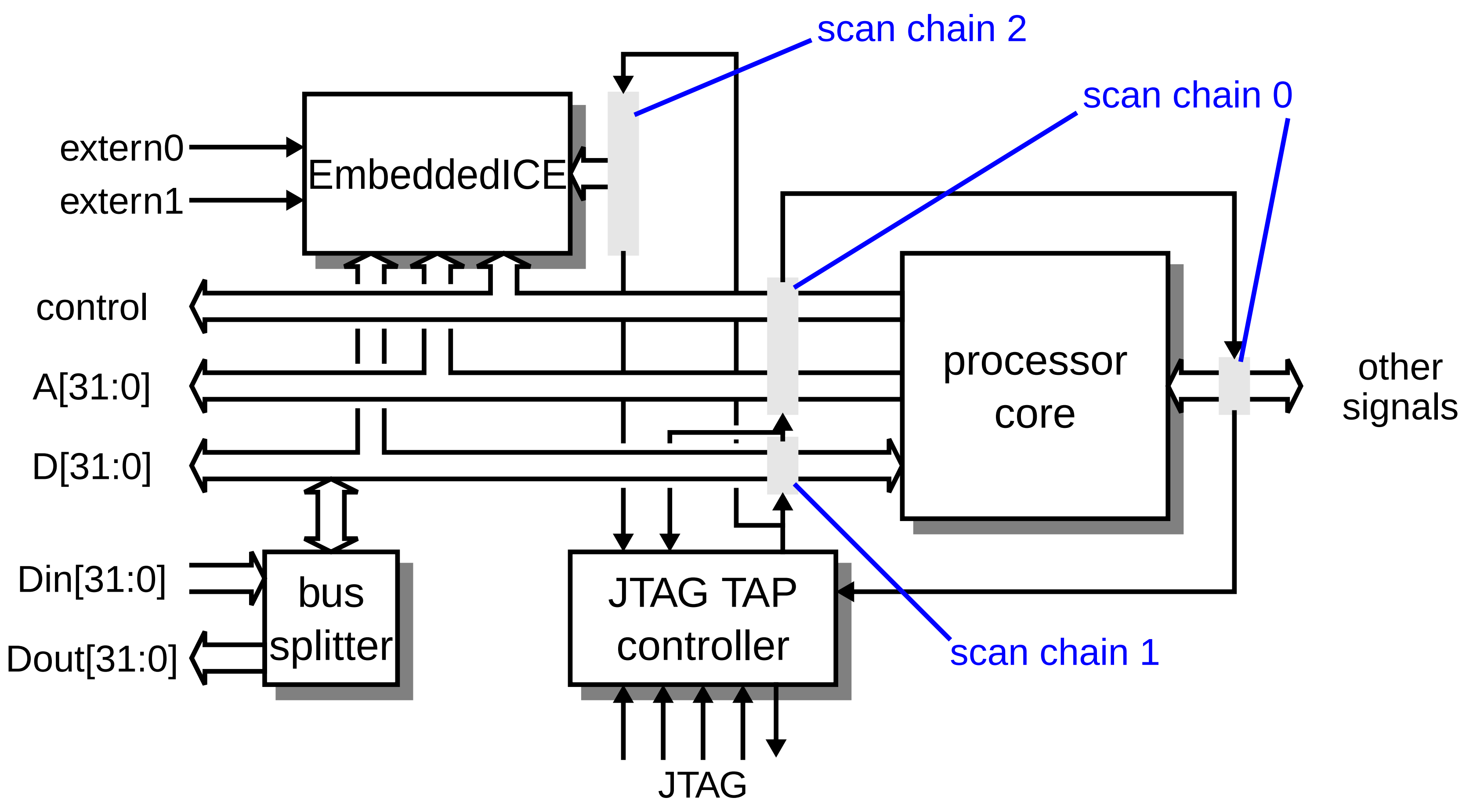
ARM integer cores

- Outline:
  - the ARM 3-stage pipeline
  - ➜ **the ARM7TDMI core**
  - the ARM 5-stage pipeline
  - the ARM9TDMI core
  - the ARM10TDMI core
  - StrongARM & XScale
  - the ARM11 core
  - Cortex
  - ☞ hands-on: system software – interrupts

The ARM7TDMI

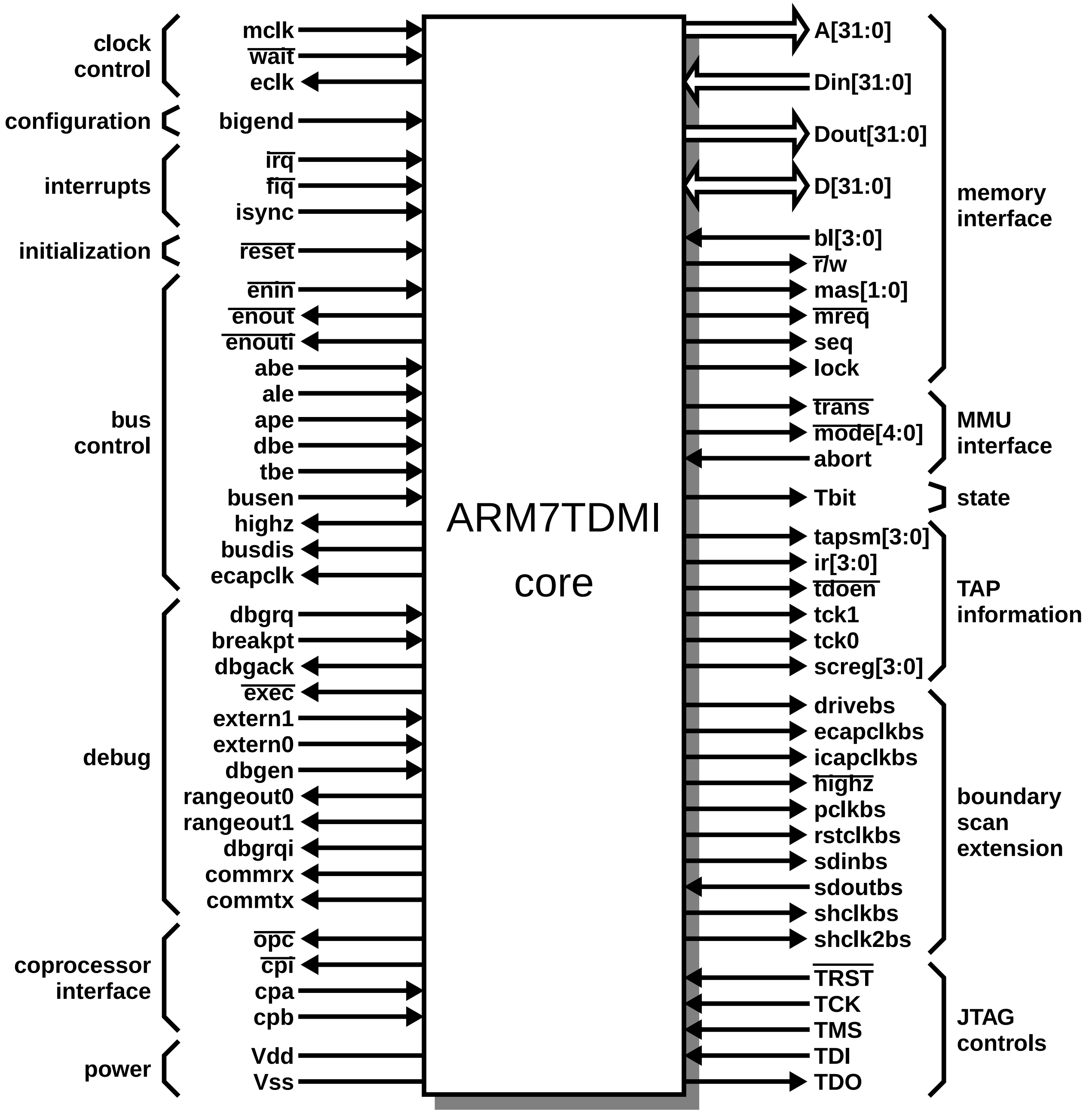
- The ARM7TDMI is …
  - an **ARM7** 3-stage pipeline core, with
  - **T** - support for the Thumb instruction set
  - **D** - support for debug
    - the processor can stop on a debug event
  - **M** - support for long multiplies
  - **I** - the EmbeddedICE macrocell
    - provides breakpoint and watchpoint hardware
      - described later

ARM7TDMI organization



- Scan chains provide access to signals for debugging

The ARM7TDMI core interface signals



ARM7TDMI core interface signals

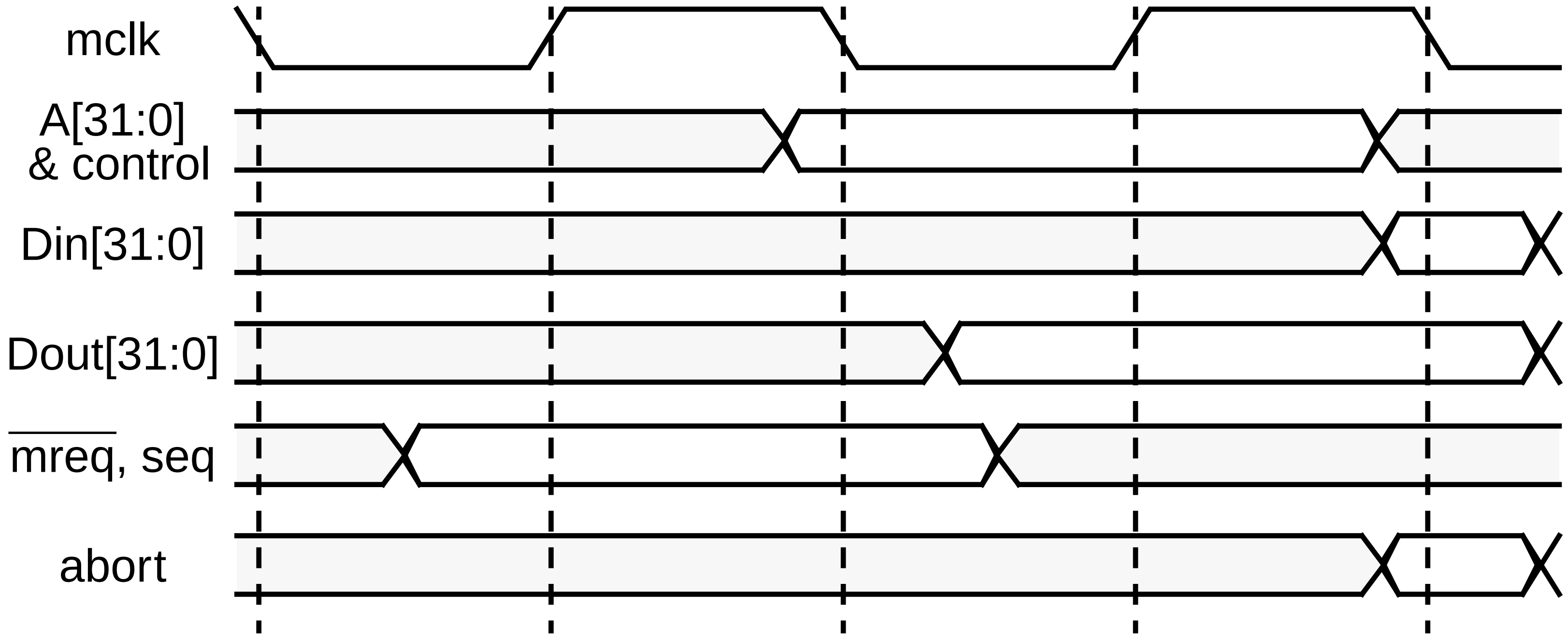
- Memory interface
  - ~mreq - memory request
  - seq - sequential address
    - together these signals indicate the sort of bus cycle which will happen next
    - they are pipelined ahead to aid memory design

mreq	seq	Cycle	Use
0	0	N	Non-sequential memory access
0	1	S	Sequential memory access
1	0	I	Internal cycle bus and memory inactive
1	1	C	Coprocessor register transfer memory inactive

MANCHEstER 1824

The University of Manchester

ARM7TDMI core interface signals



- Notes:
  - request for memory is in clock cycle before transfer occurs
  - wait state insertion is possible

ARM7TDMI

- ARM7TDMI debug support
  - the EmbeddedICE module
  - supports breakpoints and watchpoints
    - controlled via the JTAG test access port
    - EmbeddedICE & JTAG are covered later
- ARM7TDMI characteristics:

Process 0.35 μm	Transistors 74,209	MIPS 60
Metal layers 3	Core area 2.1 mm^2	Power 87 mW
Vdd 3.3V	Clock 0 to 66 MHz	MIPS/W 690

A ‘modern’ ARM7 (0.18μm) is < $\frac{1}{2}mm^2$

MANCHEstER 1824

The University of Manchester

ARM integer cores

- Outline:
  - the ARM 3-stage pipeline
  - the ARM7TDMI core
  - ➜ **the ARM 5-stage pipeline**
  - the ARM9TDMI core
  - the ARM10TDMI core
  - StrongARM & XScale
  - the ARM11 core
  - Cortex
  - ☞ hands-on: system software – interrupts

MANCHEstER
1824

The University
of Manchester

Getting higher performance

- Increase the clock rate
  - the clock rate is limited by the slowest pipeline stage
    - decrease the logic complexity per stage
    - increase the pipeline depth (number of stages)
- improve the CPI (clocks per instruction)
  - fewer wasted cycles
    - better memory bandwidth

The 5-stage ARM pipeline

- Fetch
- Decode
  - instruction decode and register read
- Execute
  - shift and ALU
- Memory
  - data memory access
- Write-back

The 5-stage ARM pipeline

- Reducing the CPI
  - ARM7 uses the memory on nearly every clock cycle
    - for either instruction fetch or data transfer
  - therefore a reduced CPI requires more than one memory access per clock cycle
- Possible solutions are:
  - separate instruction and data memories
  - double-bandwidth memory (e.g. ARM8)

MANCHEstER
1824

The University
of Manchester

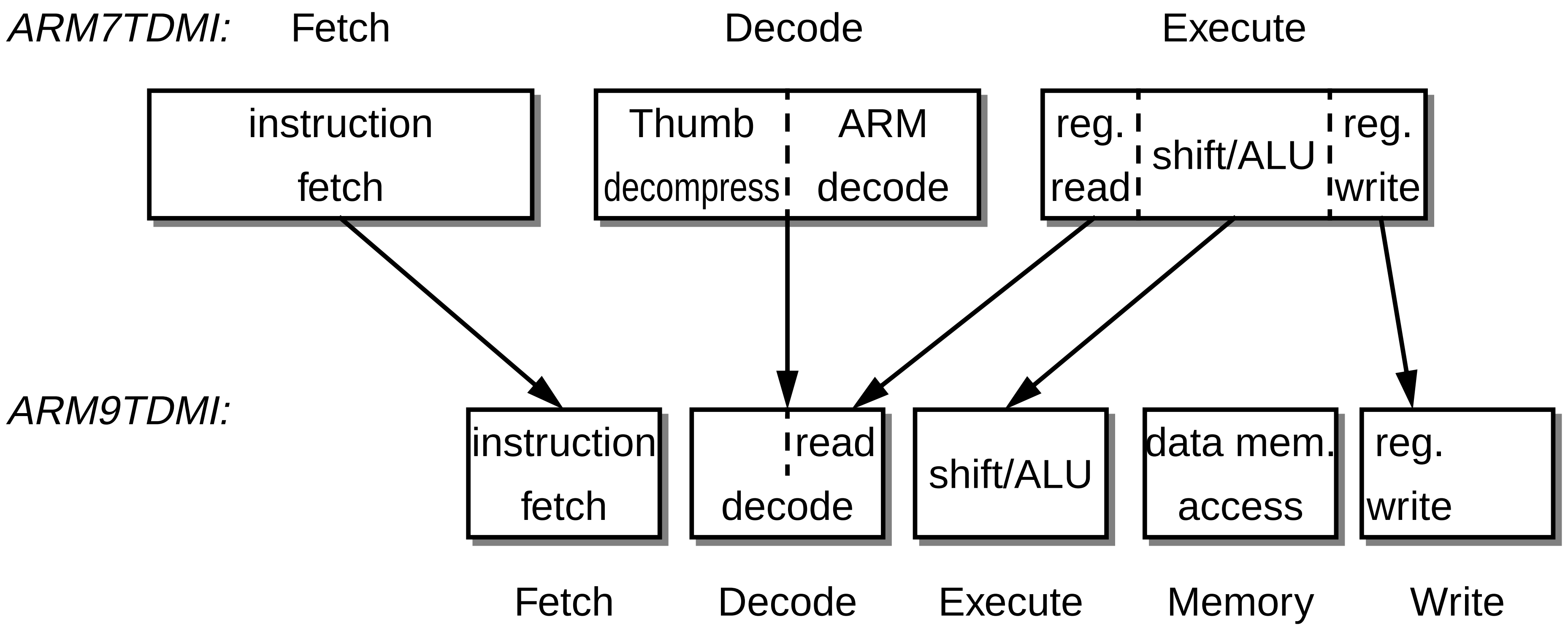
ARM integer cores

- Outline:
  - the ARM 3-stage pipeline
  - the ARM7TDMI core
  - the ARM 5-stage pipeline
  - ➜ **the ARM9TDMI core**
  - the ARM10TDMI core
  - StrongARM & XScale
  - the ARM11 core
  - Cortex
  - ☞ hands-on: system software – interrupts

ARM9TDMI

- The ARM9TDMI is …
  - a ‘classic’ Harvard architecture 5-stage pipeline
    - separate instruction and data memory ports
  - with full support for Thumb and EmbeddedICE debug
  - aimed at significantly higher performance than the ARM7TDMI
    - enhanced pipeline (then) operated at 100-200 MHz
    - (now) up to 250MHz

ARM9TDMI pipeline



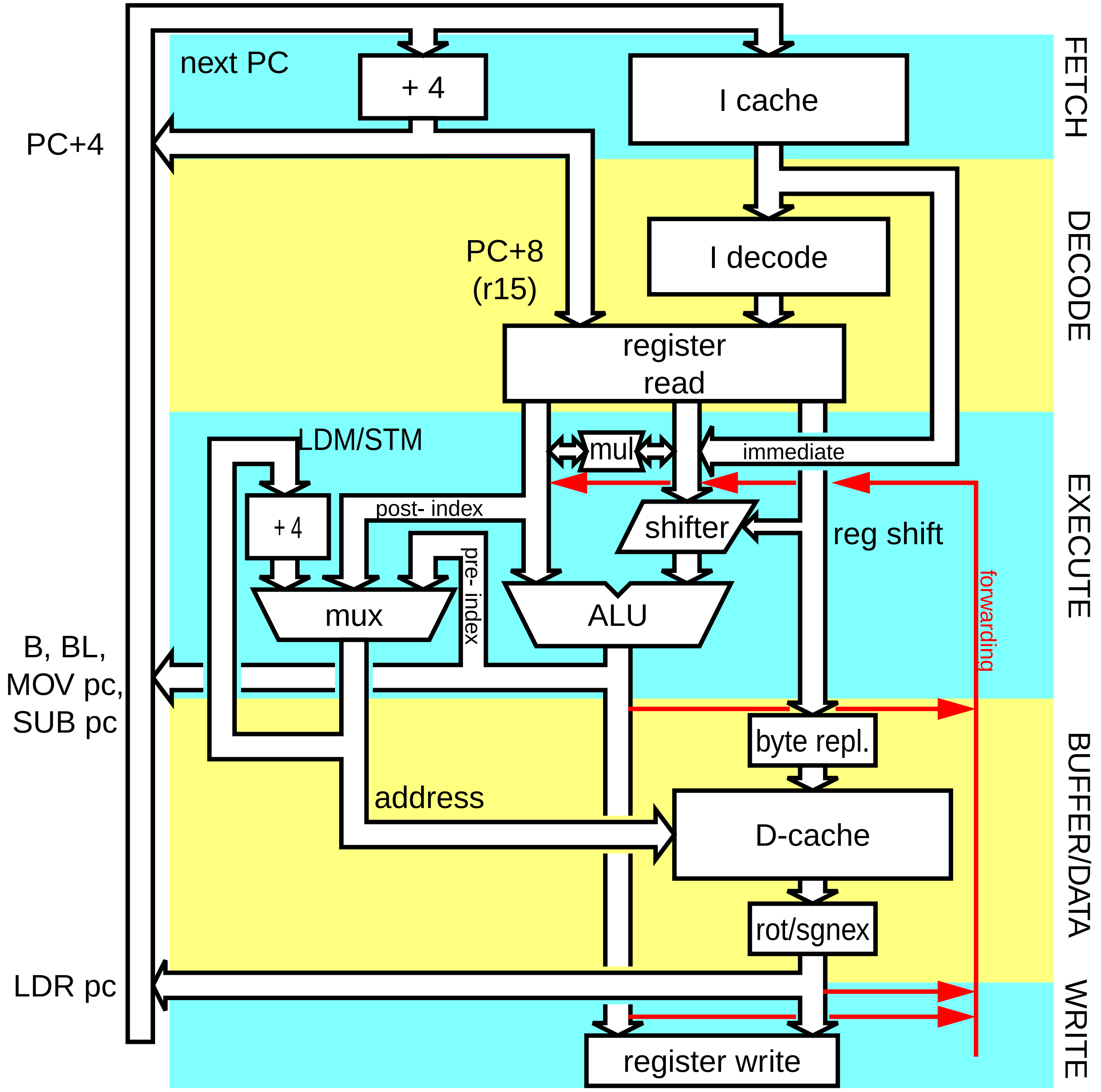
- Thumb instructions are decoded directly

MANCHEstER
1824

The University
of Manchester

ARM9TDMI pipeline

- Pipeline introduces more dependencies
- alleviated with forwarding paths



MANCHEStER
1824

The University
of Manchester

ARM9TDMI

- EmbeddedICE
  - as ARM7TDMI, plus:
    - hardware single-stepping
    - breakpoints on exceptions
- On-chip coprocessor support
  - for floating-point, DSP, and so on

Process 0.25 µm	Transistors 111,000	MIPS 220
Metal layers 3	Core area 2.1 mm^2	Power 150 mW
Vdd 2.5 V	Clock 0-200 MHz	MIPS/W 1,500

ARM integer cores

- Outline:
  - the ARM 3-stage pipeline
  - the ARM7TDMI core
  - the ARM 5-stage pipeline
  - the ARM9TDMI core
  - ➜ **the ARM10TDMI core**
  - StrongARM & XScale
  - the ARM11 core
  - Cortex
  - ☞ hands-on: system software – interrupts

MANCHEstER
1824

The University
of Manchester

ARM10TDMI

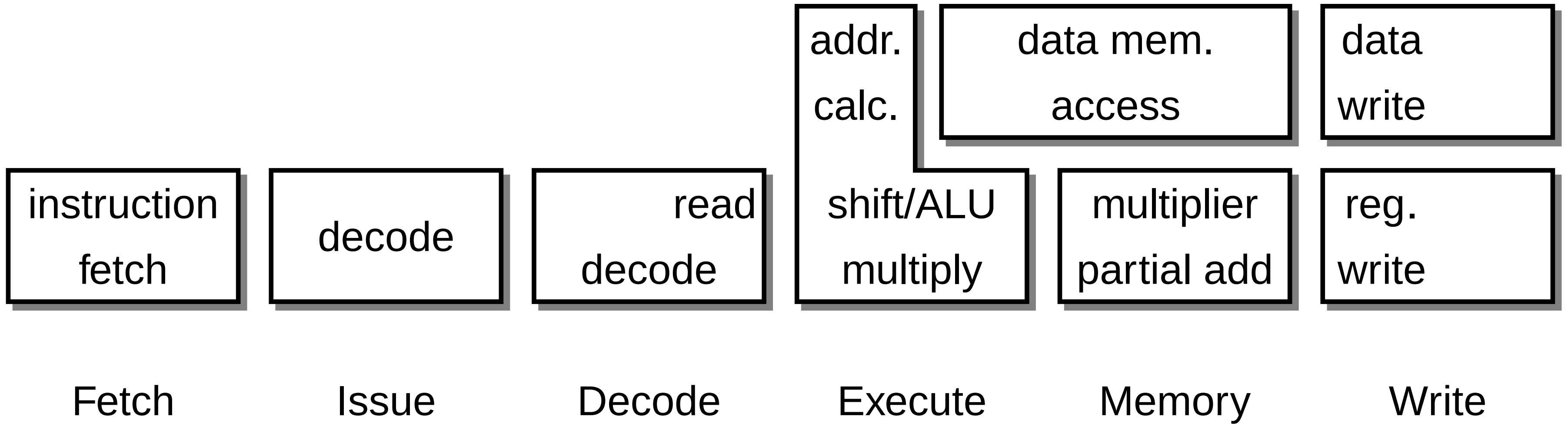
- The ARM10TDMI is …
  - aimed at significantly higher performance than the ARM9TDMI
  - achieved through use of:
    - higher clock rate
    - 64-bit I- and D-memory buses
    - branch prediction
    - hit-under-miss D-memory interface

MANCHEStER 1824

The University of Manchester

ARM10TDMI pipeline

- 6-stage pipeline



- Additional time allowed for
  - I- and D-memory accesses
  - instruction decode

MANCHEstER
1824

The University
of Manchester

ARM integer cores

- Outline:
  - the ARM 3-stage pipeline
  - the ARM7TDMI core
  - the ARM 5-stage pipeline
  - the ARM9TDMI core
  - the ARM10TDMI core
  - ➜ **StrongARM & XScale**
  - the ARM11 core
  - Cortex
  - ☞ hands-on: system software – interrupts

StrongARM & XScale

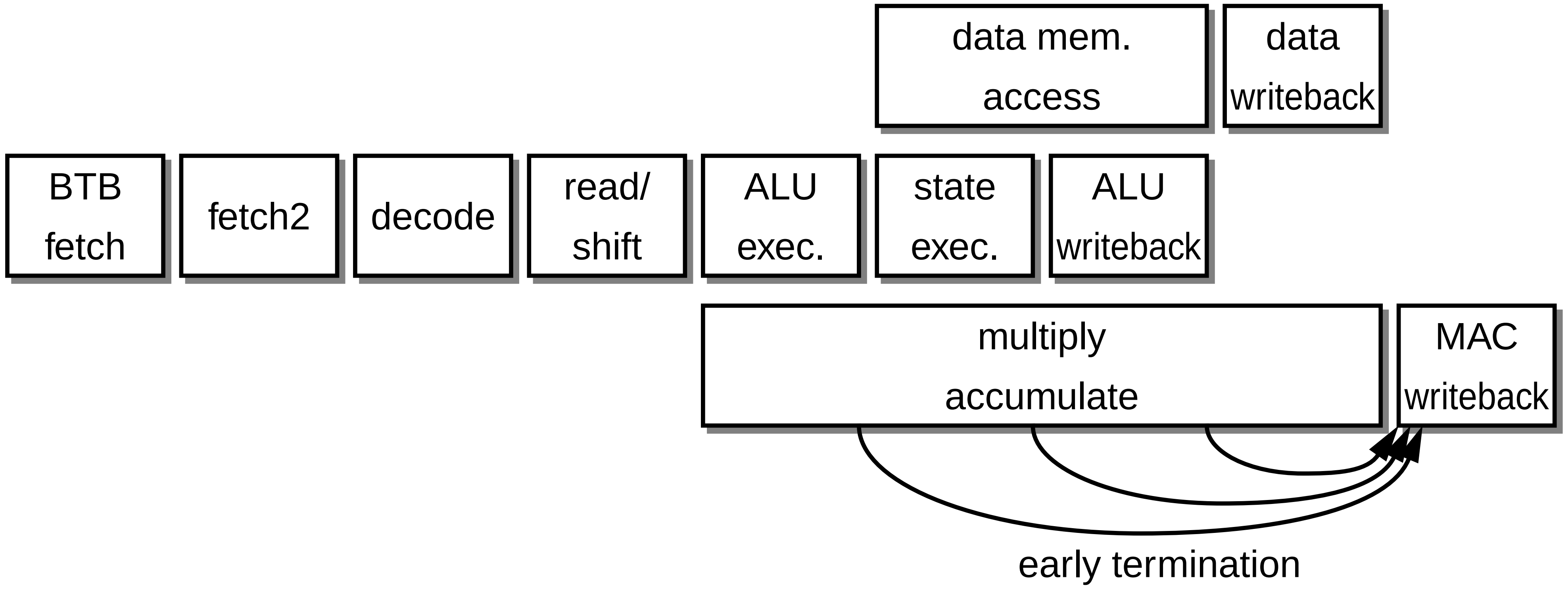
- Most ARM implementations are designed by ARM Ltd.
- Two notable exceptions:
  - StrongARM
    - designed by Digital, later bought by Intel
    - now largely obsolete
  - XScale
    - designed by Intel
    - current
    - up to 800 MHz

Both of these were designed primarily for high performance
Used for proprietary devices

MANCHEstER
1824

The University
of Manchester

XScale™ pipeline



- another deep pipeline
- (unpredicted) branches are expensive
- data (memory) dependencies cause stalls

MANCHEStER
1824

The University
of Manchester

ARM integer cores

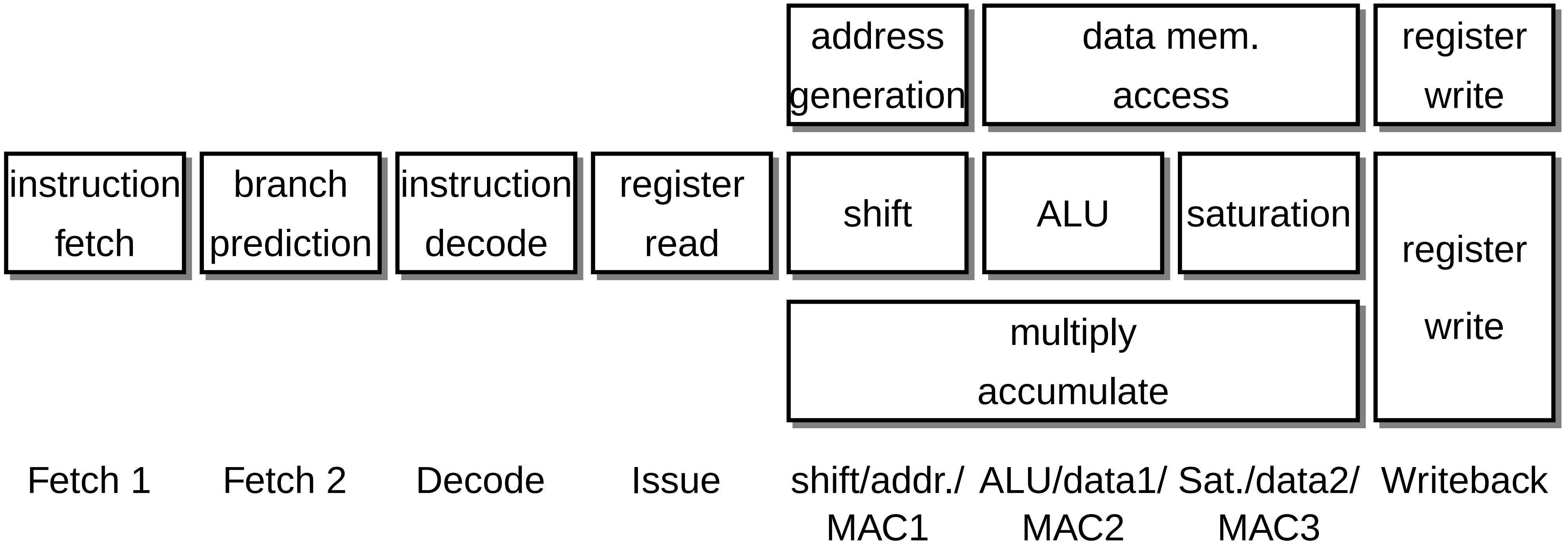
- Outline:
  - the ARM 3-stage pipeline
  - the ARM7TDMI core
  - the ARM 5-stage pipeline
  - the ARM9TDMI core
  - the ARM10TDMI core
  - StrongARM & XScale
  - ➜ **the ARM11 core**
  - Cortex
  - ☞ hands-on: system software – interrupts

ARM11

- Most recent *available* ARM core (from ARM Ltd.)
- process portable
- high speed
  - faster clock (~550MHz)
  - deeper pipeline
    - also to accommodate DSP operations
  - improved branch prediction

ARM11 pipeline

- 8-stage pipeline



  - Parallel ALU and data access (especially during LDM/STM)
  - ‘Hit under Miss’ in ‘data1’ alleviates cache miss stalls

ARM 11 branch prediction

- Two-level dynamic branch prediction
  - constant offset branches
  - 128 entry, direct mapped (addr$_{[9:3]}$)
  - cost: 1 or 0 cycles if successful (taken/not taken)
- Static branch prediction
  - constant offset branches …
  - … that miss the dynamic predictor
  - cost: 4 cycles if successful
- Return stack
  - three entries
  - Pushes on BL or BLX (inc. BLX Rn)
  - Pops on: {`BX lr; MOV pc, lr; LDR pc, [sp], #n; LDMIA sp!, {…,pc}`}
  - cost: 4 cycles if successful

ARM 11 dependencies

- Data operations forward results
- Load operations impose an extra two cycle penalty (cache hit)
- Memory operations require their address registers early

Thus:

```
ADD    r2, r1, r0      ; produce R2
ADD    r4, r3, r2      ; consume R2 - no stall

LDR    r2, [r1]        ; load r2
ADD    r4, r3, r2      ; lose 2 cycles waiting

ADD    r2, r1, r0      ; produce R2
LDR    r4, [r2]        ; lose one cycle

LDR    r2, [r1]        ;
LDR    r4, [r2]        ; lose 3 cycles in total
```

– a few other dependencies may be seen from the pipeline figure

MANCHEstER 1824

The University of Manchester

ARM integer cores

- ❑ Outline:
  - ❍ the ARM 3-stage pipeline
  - ❍ the ARM7TDMI core
  - ❍ the ARM 5-stage pipeline
  - ❍ the ARM9TDMI core
  - ❍ the ARM10TDMI core
  - ❍ StrongARM & XScale
  - ❍ the ARM11 core
  - ➜ **Cortex**
  - ☞ hands-on: system software – interrupts

Cortex™

- Three ‘flavours’:
  - Cortex-A Series
    - applications processors
    - ARM, Thumb and Thumb-2 instruction sets
  - Cortex-R Series
    - embedded processors for real-time systems.
    - ARM, Thumb, and Thumb-2 instruction sets
  - Cortex-M Series
    - deeply embedded/cost sensitive processors
    - Thumb-2 instruction set only

Cortex-M3

- First of the line
  - Thumb-2 only
  - new design
    - 3-stage pipeline
    - Harvard core
  - small core
    - ~$\frac{1}{2}$mm^2 (excluding cache)
  - performance (0.18µm):
    - ~100 MHz
    - ~8000 MIPS/W

Hands-on: system software – interrupts

- Look further into ARM system software issues
  - Write an interrupt handler
  - Use it to trigger a context switch

☞ Follow the ‘Hands-on’ instructions