MANCHEStER
1824

The University of Manchester

System Development

- Outline:
  - system modelling
  - on-chip debug
  - AMBA
  - rapid silicon prototyping
  - embedded ARM cores

☞ hands-on: system modelling

MANCHEStER 1824

The University of Manchester

System Development

- ❑ Outline:
  - ➜ **system modelling**
  - ❍ on-chip debug
  - ❍ AMBA
  - ❍ rapid silicon prototyping
  - ❍ embedded ARM cores

☞ hands-on: system modelling

MANCHEstER
1824

The University
of Manchester

System Modelling

- From prototype environment ...
  - Undefined resources: Core, Memory, Cache
  - C library use of hardware
  - Debug libraries
- ... to final product
  - Standalone embedded application
  - Specific memory layout of the target hardware
  - Initialisation sequence

MANCHEStER 1824

The University of Manchester

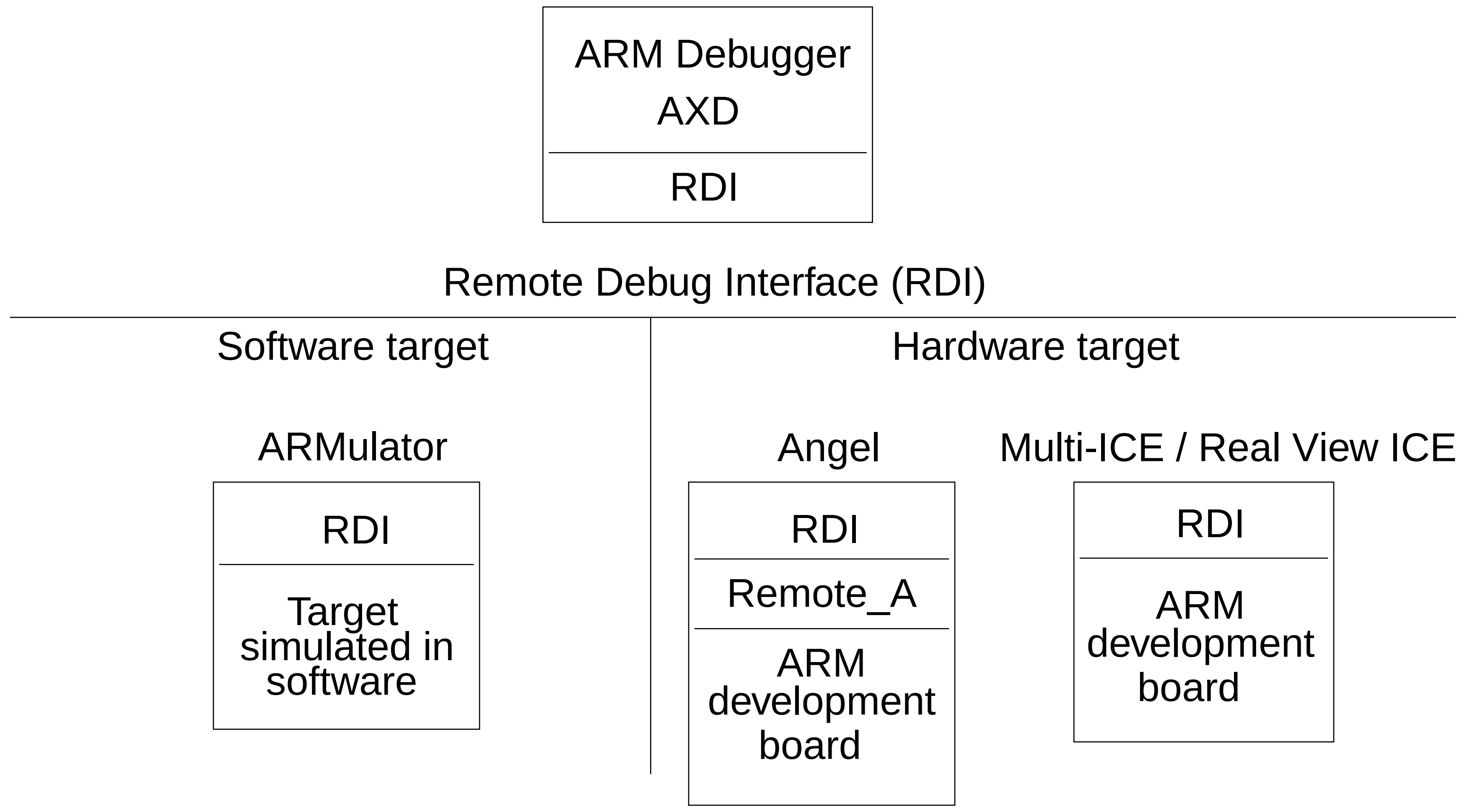
System development & debugging

- A common set of debugger front-ends
  - armsd, AXD, RVD
- Same code can be debugged on:
  - Software simulation targets
    - ARMulator
  - Hardware targets
    - RealView ICE, Multi-ICE, RV Trace, MultiTrace, Angel

MANCHEStER 1824

The University of Manchester

Debugger-target interface



Software debug: the ARMulator

- A software model of an ARM core with:
  - support for Thumb instructions
  - a programmable memory interface
    - for modelling the target memory system
    - various rapid prototyping tools are supplied
  - a coprocessor interface
    - supporting custom coprocessor models
  - an operating system interface
    - system calls handled by host or emulation

The ARMulator

- The core of a complete system model
  - clock-cycle accurate
  - inspect registers and memory
  - set breakpoints and watchpoints
- Supports software development
  - concurrently with hardware development
  - higher performance than detailed hardware models

MANCHEStER 1824

The University of Manchester

From ARMulator to on-chip Debug

- Important to understand simulator's default behaviour
- Default build needs to be tailored to specific needs:
  - Uses of ADS/RVDS C library
    - semi-hosted SWI calls
  - Memory map and Linker placement rules
  - Reset and initialisation

ADS/RVDS C Library

- Avoiding C library semihosting
  - import __use_no_semihosting_swi (in C: #pragma import)
  - linker reports any remaining SWI call
- Retargeting C library calls
  - example: retargeting the `printf()` family of function to print out to a hardware UART

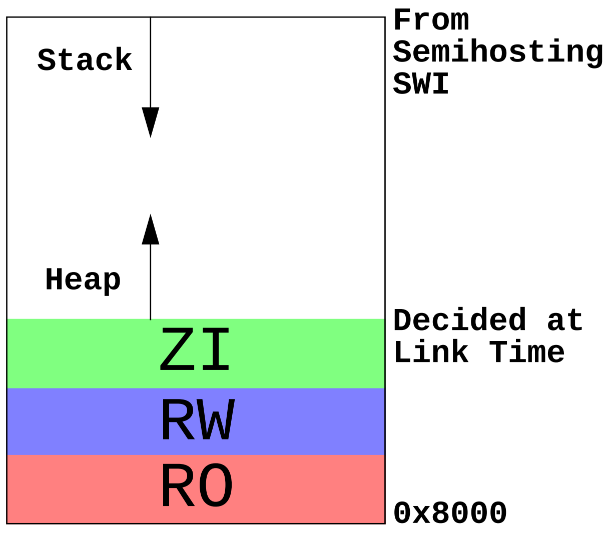
```
extern void sendchar (char *c); /* UART communications */
int fputc (int c, FILE *f)
{ /* redirect a char to the UART */
    sendchar (c);
    return c;
}
```

MANCHEStER 1824

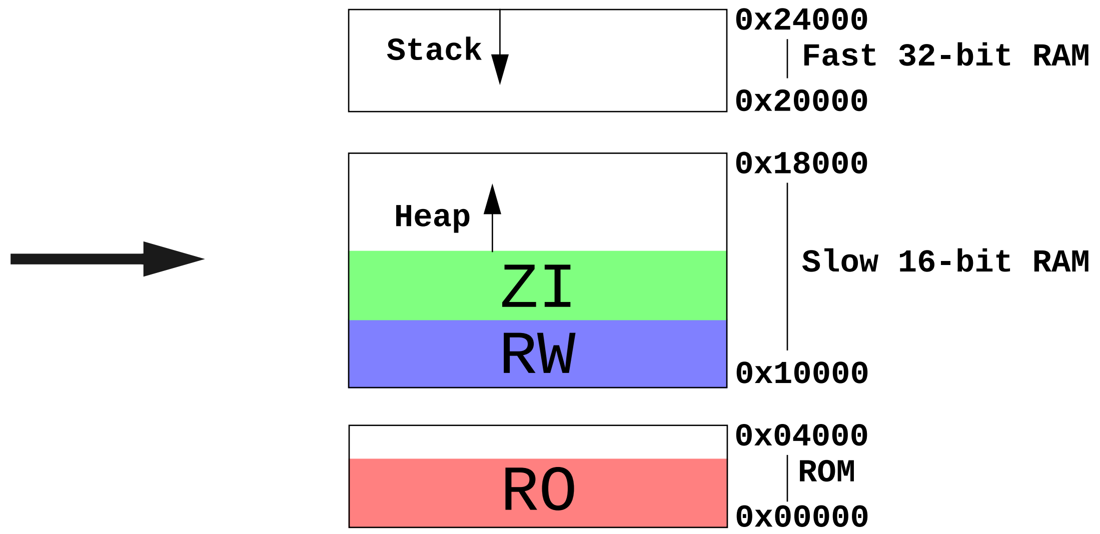
The University of Manchester

Image memory map

- Target hardware usually has several memory devices at different address ranges
- Scatterloading
  - describes memory location of code&data at load&run-time
  - `armlink -scatter scatfile.scf file1.o file2.o`



Default memory map



Simple scatterloading example

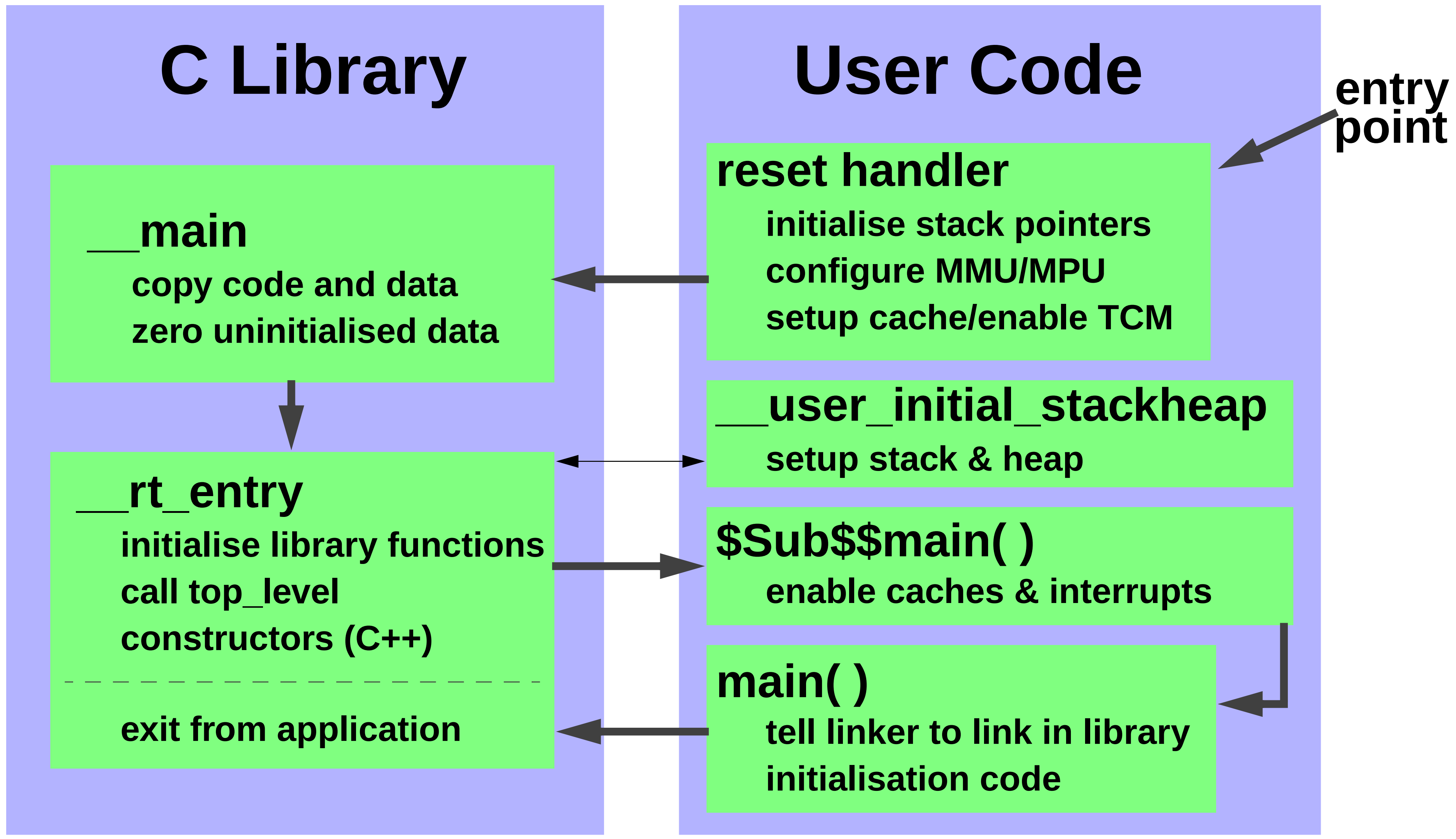
Reset and Initialisation

- Usually need to initialise:
  - vector table
  - stack pointers in IRQ/FIQ modes
  - MMU/MPU
  - other hardware

MANCHEstER 1824

The University of Manchester

Initialisation sequence example



System Development

❑ Outline:

- ❍ system modelling
- ➜ **on-chip debug**
- ❍ AMBA
- ❍ rapid silicon prototyping
- ❍ embedded ARM cores

☞ hands-on: system modelling

MANCHEstER 1824

The University of Manchester

On-chip debug

- Debug monitor: Angel
  - runs on target hardware with the application
  - requires target resources (memory, exception vectors, …)
- Integrated on-chip debug: Multi-ICE / RealView ICE
  - non intrusive, requires almost no resources
  - instead, uses additional debug hardware within the core
    - ARM processor debug extension signals (main ones: BREAKPT, DBGRQ, DBGACK)
    - EmbeddedICE, Embedded Trace

EmbeddedICE

- Hardware registers controlled through:
  - JTAG boundary scan
  - Debug coprocessor
- Two possible actions:
  - Halt debug-mode debugging
    - processor halts at debug events
    - unsuitable for real-time systems
  - Monitor debug-mode debugging
    - debug events generate exceptions (aborts)
    - non-intrusive mode, for debugging real-time systems

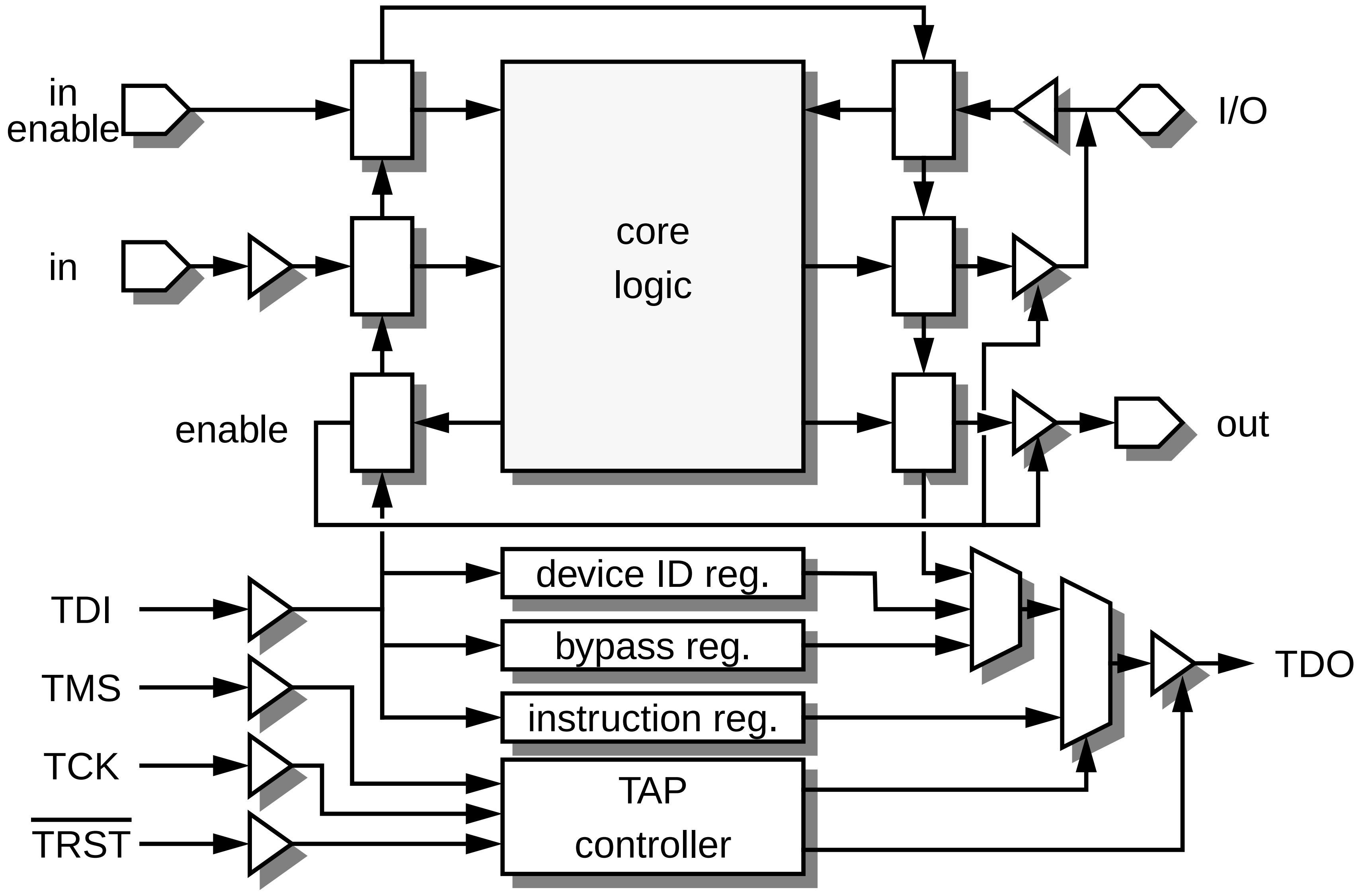
JTAG Test Access Port

- Joint Test Action Group
  - looked especially at PCB production test
    - surface mount defeats bed of nails approach
  - on-chip scan path gives access to pins
    - so chip to chip paths can be tested
  - other uses are a side benefit:
    - in-circuit testing of the chip core logic
    - chip debug support, e.g. EmbeddedICE
  - Note: not primarily for VLSI production test!

MANCHEsTER 1824

The University of Manchester

JTAG Boundary Scan Organization



EmbeddedICE

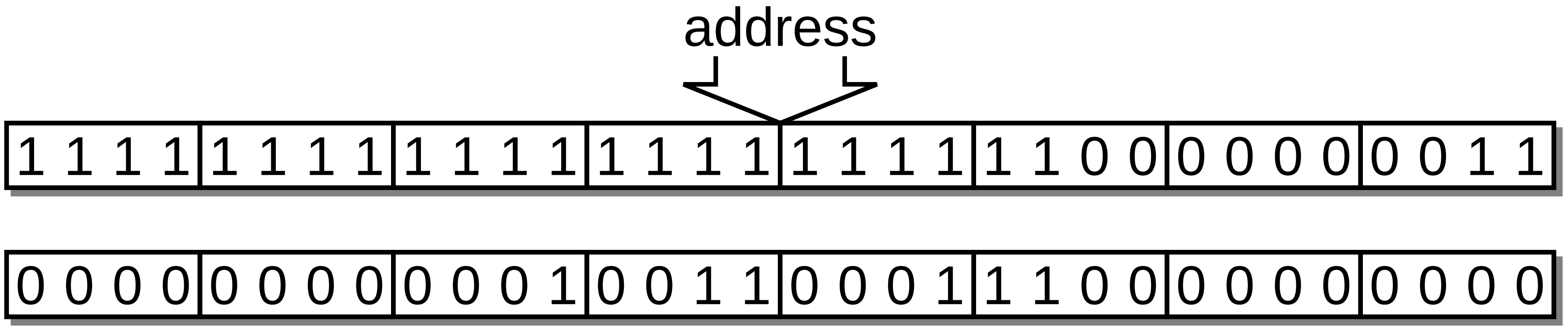
- ICE functions:
  - breakpoints, watchpoints
    - generate an event at a particular instruction/data access
    - hardware can easily be included on chip
    - N.B. ROM breakpoints require hardware!
  - trace buffer
    - retains interface state before and after trigger
    - Embedded Trace Macrocell now supported
    - uses hardware compression to reduce pin requirement

Breakpoints and Watchpoints

- Breakpoint
  - if this memory address is fetched as an instruction an exception occurs
    - may be inserted as an instruction (BKPT)
    - may be detected in hardware
- Watchpoint
  - if this memory address is accessed by a load or store an exception occurs
    - must be detected in hardware

Breakpoints and Watchpoints

- ARM break- watchpoint hardware
  - mask and pattern
  - trap if selected bits match desired pattern
  - example:

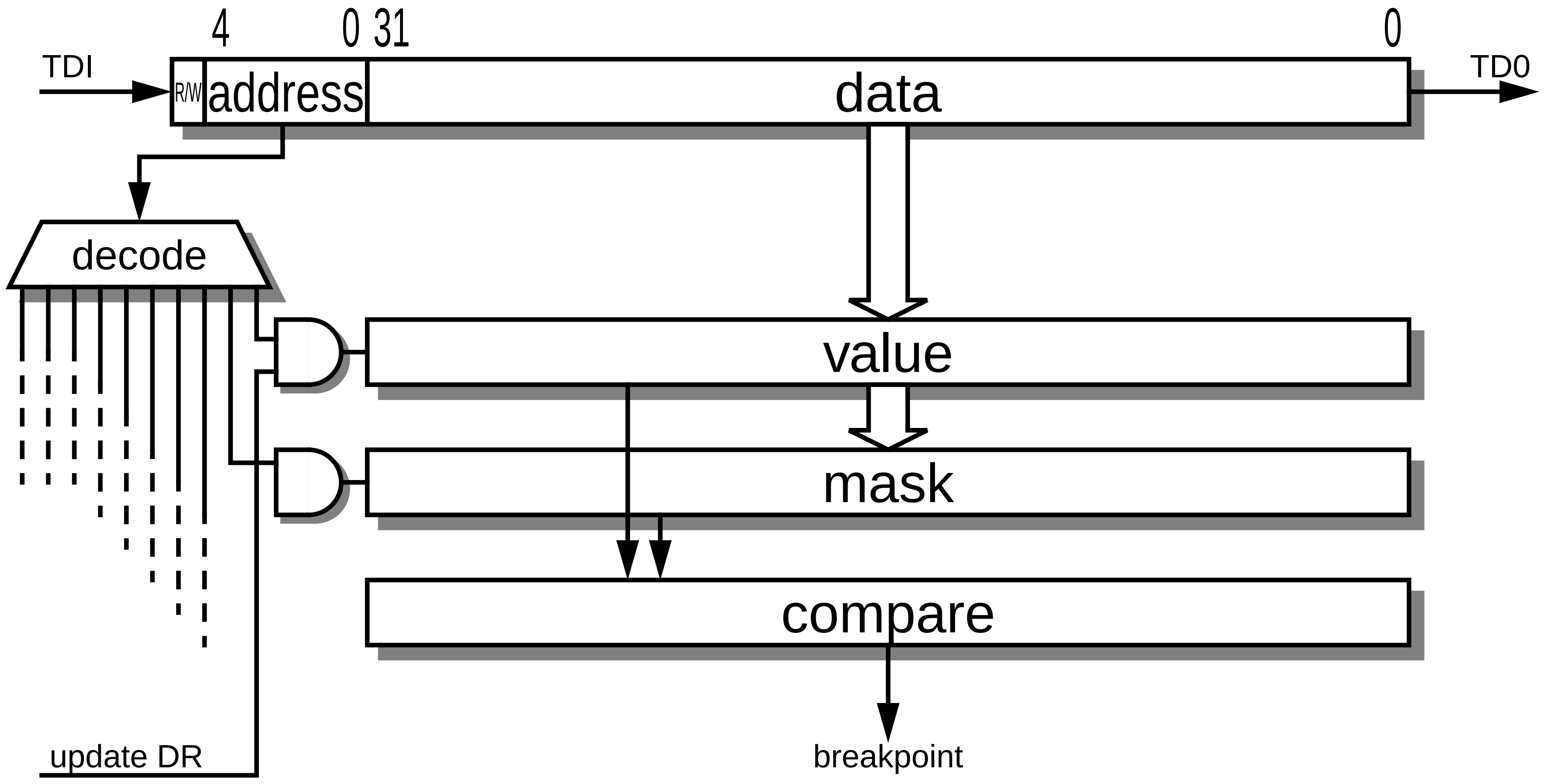


breaks on word addresses 0x00131C00 – 0x00131CFC

MANCHEstER 1824

The University of Manchester

EmbeddedICE register read and write structure



- Registers accessed via **scan chain**

MANCHEStER
1824

The University
of Manchester

EmbeddedICE register mapping

Address	Width	Function
00000	3	Debug control
00001	5	Debug status
00100	6	Debug comms control register
00101	32	Debug comms data register
01000	32	Watchpoint 0 address value
01001	32	Watchpoint 0 address mask
01010	32	Watchpoint 0 data value
01011	32	Watchpoint 0 data mask
01100	9	Watchpoint 0 control value
01101	8	Watchpoint 0 control mask
10000	32	Watchpoint 1 address value
10001	32	Watchpoint 1 address mask
10010	32	Watchpoint 1 data value
10011	32	Watchpoint 1 data mask
10100	9	Watchpoint 1 control value
10101	8	Watchpoint 1 control mask

MANCHEstER 1824

The University of Manchester

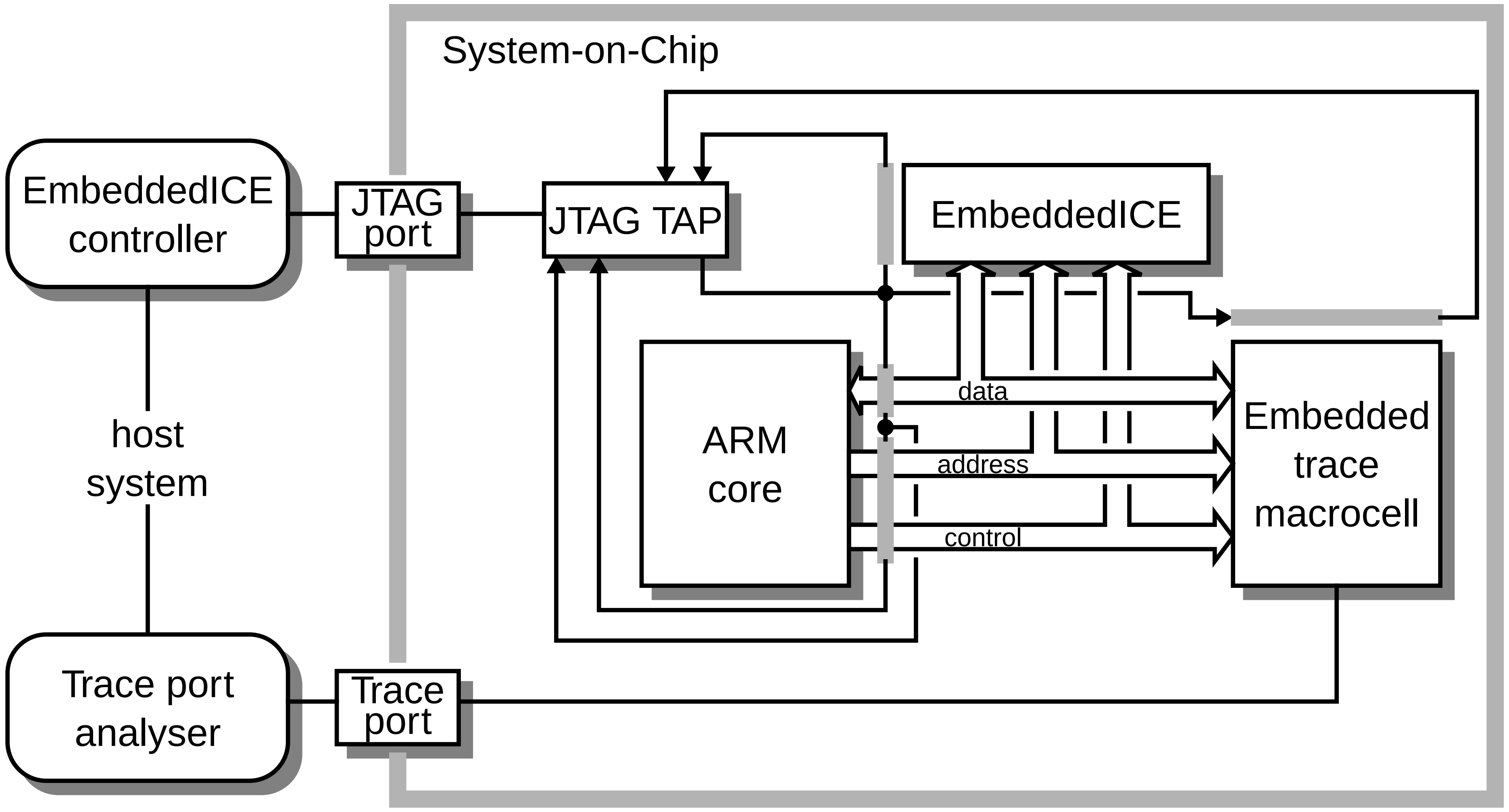
Embedded Trace

- The **embedded trace macrocell** (ETM) comprises:
  - trace port – outputs processor signals
  - filtering/triggering – allows capture of wanted data
    - triggering allows capture from selected code
    - filtering disregards unwanted data – saves storage/bandwitdh
- these can make the processor behaviour *observable*
  - signals available at trace port

MANCHEStER 1824

The University of Manchester

Embedded Trace



Embedded Trace

- A **trace buffer** can be added to store trace signals
  - essential at high speeds!
- Comprises:
  - trace interface
  - JTAG interface
  - AHB bus interface
- Needs:
  - RAM to store traces

Debug Unit

- Programmable through CP14 or scan chains
- Characteristics
  - instruction address comparators for triggering breakpoints
  - data address comparators for triggering watchpoints
  - bidirectional Debug Communication Channel
  - ability to disable caches and TLBs
  - mode for debugging real-time systems
    (e.g. servo mechanisms)

MANCHEStER 1824

The University of Manchester

Debug Unit

- Halt debug-mode debugging
  - processor halts at debug events (breakpoints, ...)
  - when halted, external host can examine and modify its state using the DBGTAP pin
  - unsuitable for real-time systems
  - requires external hardware to control DBGTAP
- Monitor debug-mode debugging
  - debug events generate exceptions
  - handler can program new debug events through CP14

MANCHEstER 1824

The University of Manchester

CP14 Registers

Register Opcode2:CRm	Abbreviation	Name
0	DIDR	Debug ID Register
1	DSCR	Debug Status and Control Register
2-4	–	Reserved
5	DTR	Data Transfer Register
6	WFAR	Watchpoint Fault Address Register
7	VCR	Vector Catch Register
8-9	–	Reserved
10	DSCCR	Debug State Cache Control Register
11	DSMCR	Debug State MMU Control Register
12-63	–	Reserved
64-69	BVR_N	Breakpoint Value Registers
70-79	–	Reserved
80-85	BCR_N	Breakpoint Control Registers
86-95	–	Reserved
96-97	WVR_N	Watchpoint Value Registers
98-111	–	Reserved
112-113	BVR_N	Watchpoint Control Registers
114-127	–	Reserved

MANCHEStER
1824

The University
of Manchester

System Performance Monitoring

- A small collection of counters, triggered by ‘events’
  - e.g. cache miss, TLB miss, dependency stall, branch mispredicted, …
  - configurable
  - can cause interrupts after a preset number of events
  - introduced in ARM11
- Can be used for code profiling
- Accessible via CP15

MANCHEStER
1824

The University
of Manchester

System Development

❑ Outline:

- ❍ system modelling
- ❍ on-chip debug
- ➜ **AMBA**
- ❍ rapid silicon prototyping
- ❍ embedded ARM cores

☞ hands-on: system modelling

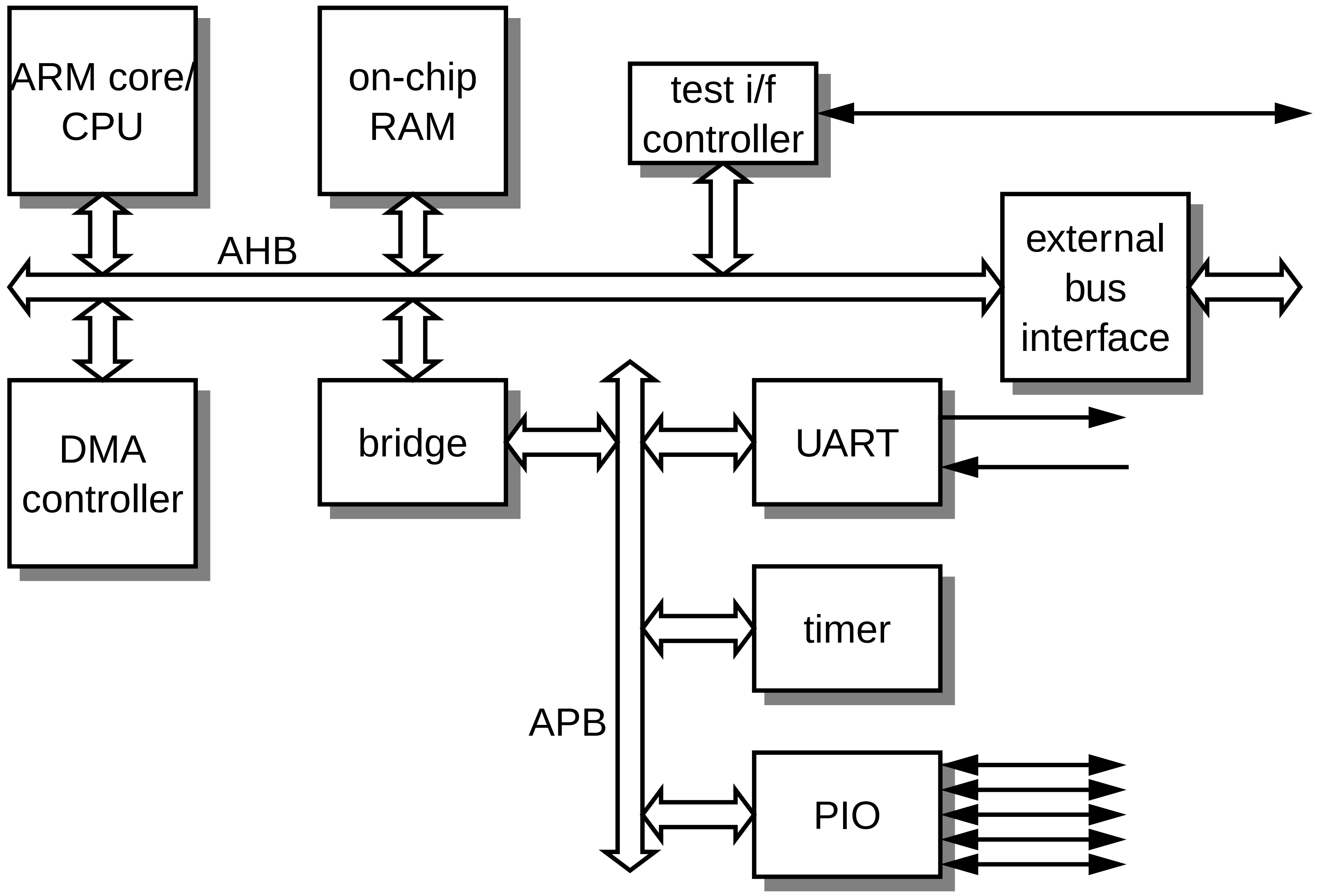
AMBA

- Advanced Microprocessor Bus Architecture
  - a systematic solution to assembling macrocell-based systems
  - ARM Ltd's attempt to establish an on-chip bus standard
- AMBA structure:
  - Advanced High-performance Bus (AHB)
    - high-performance, multi-master
  - Advanced Peripheral Bus (APB)
    - interface for low performance peripherals
  - Advanced eXtensible Interface (AXI) (new)

MANCHEstER 1824

The University of Manchester

A typical AMBA-based System



MANCHEstER 1824

The University of Manchester

AMBA Test Interface

- VLSI production test is an economically important issue
  - macrocell based designs present problems
    - how can each macrocell be systematically tested?
  - AMBA offers a standardised solution
    - based on 32-bit parallel access, via the bus, to test registers

AMBA Standards

Bus	Master	Performance	pipelined/split transactions	Other
AHB	multi	high	yes	32- to 1024-bit data bus
APB	single	low	no	used to reduce main bus load
AXI	multi	high	yes	separate data buses out-of-order completion

- AXI is intended as a replacement for the AHB bus
  - used for future designs
  - some components already developed:
    - L220 level-2 cache controller
    - PL300 configurable interconnect
    - PL340 SDRAM controller

MANCHEStER 1824

The University of Manchester

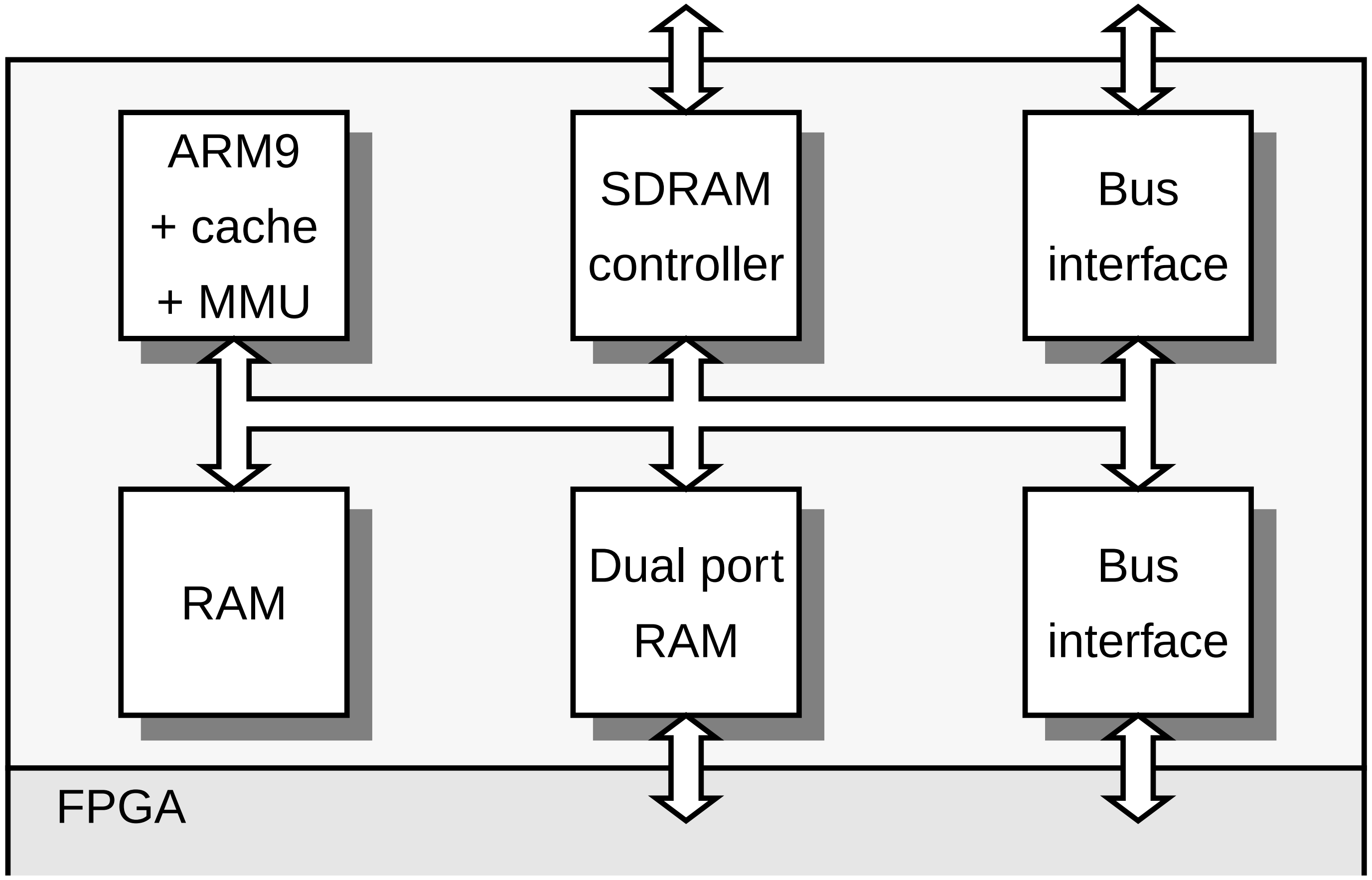
System Development

- Outline:
  - system modelling
  - on-chip debug
  - AMBA
  - ➜ **rapid silicon prototyping**
  - embedded ARM cores

☞ hands-on: system modelling

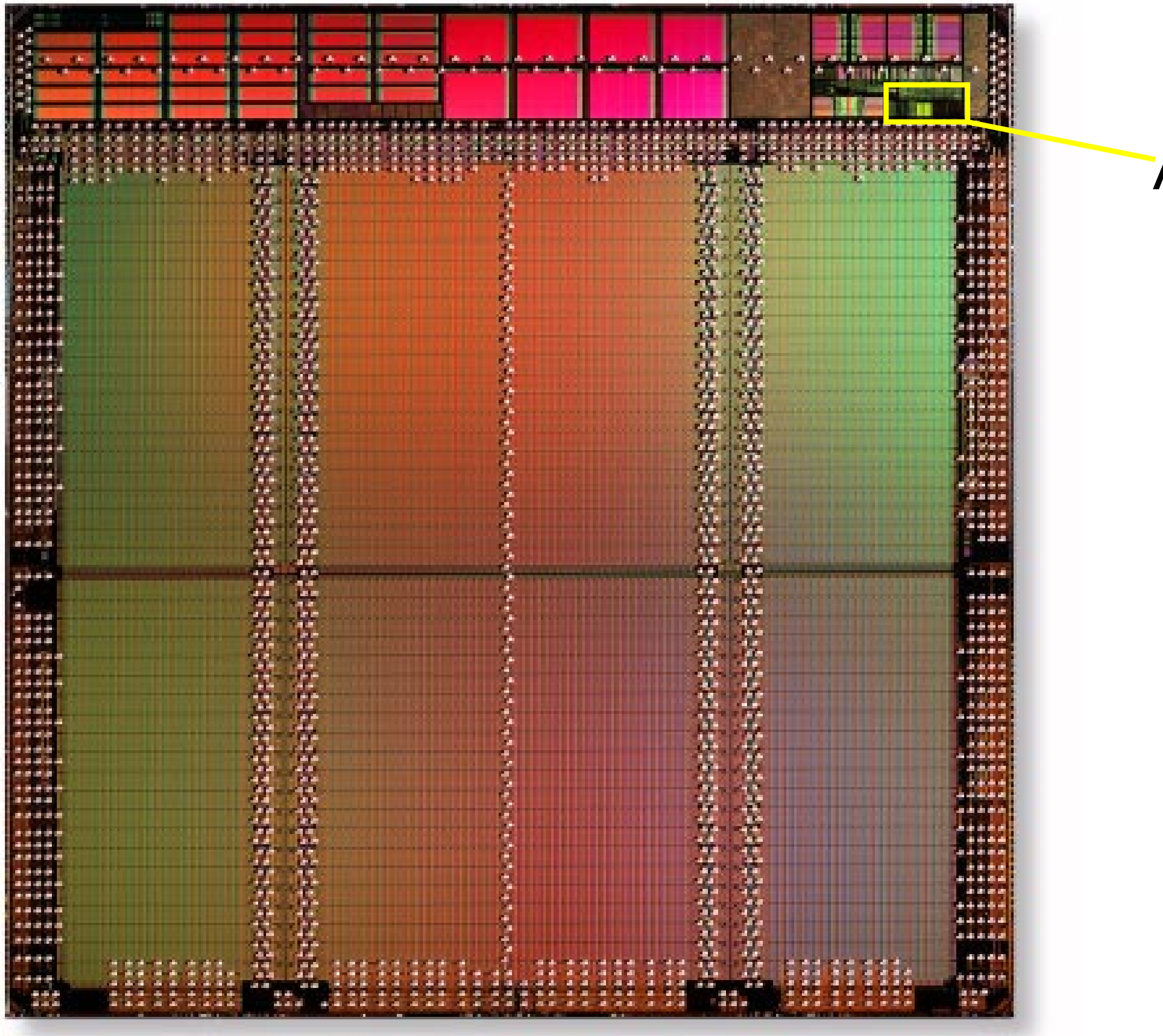
Excalibur

- Rapid prototyping system



  - ARM-based computer …
  - … plus LOTS of uncommitted gates

Excalibur



MANCHEStER
1824

The University
of Manchester

System Development

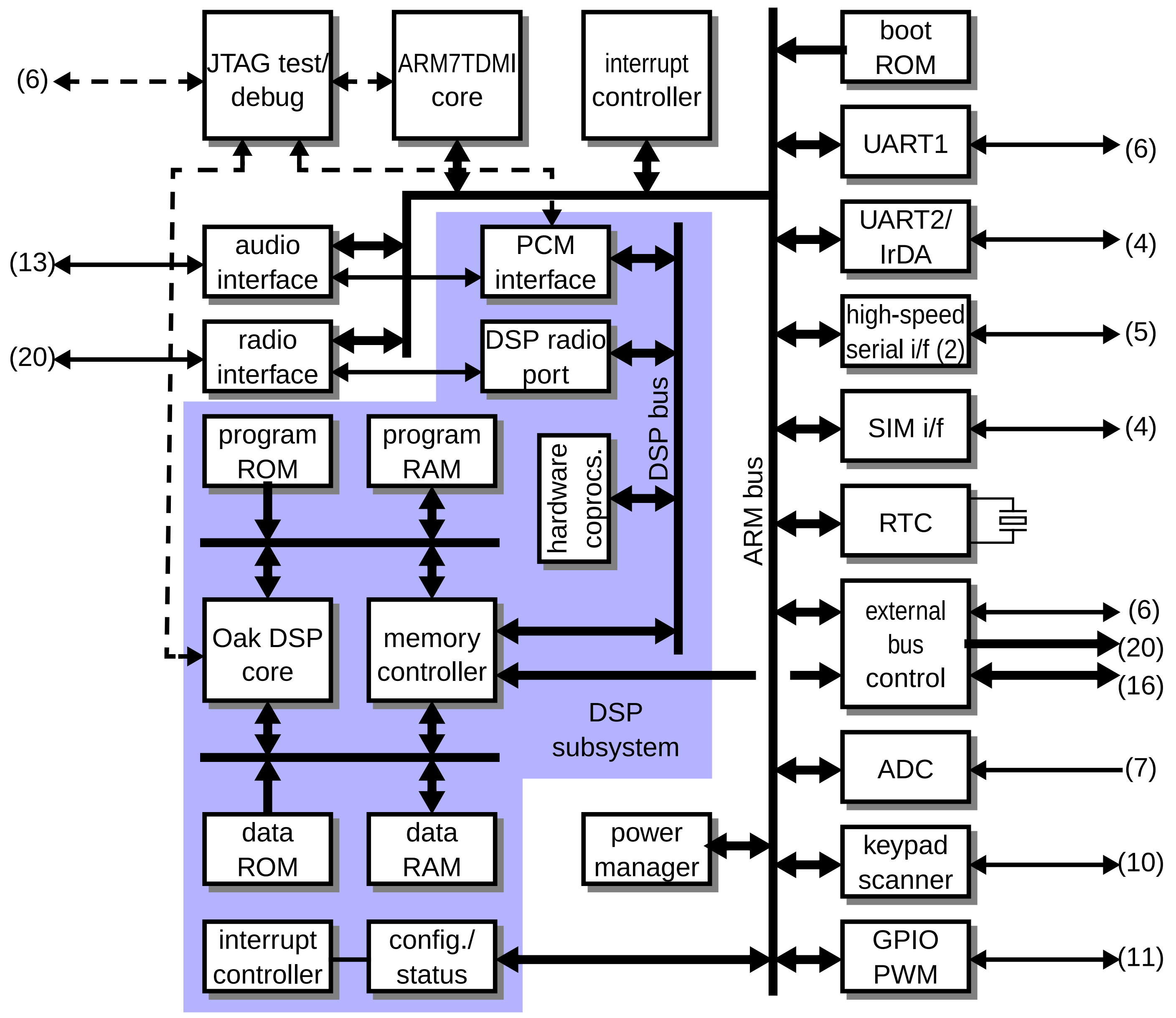
- Outline:
  - system modelling
  - on-chip debug
  - AMBA
  - rapid silicon prototyping
  - ➜ **embedded ARM cores**

☞ hands-on: system modelling

MANCHEStER
1824

The University
of Manchester

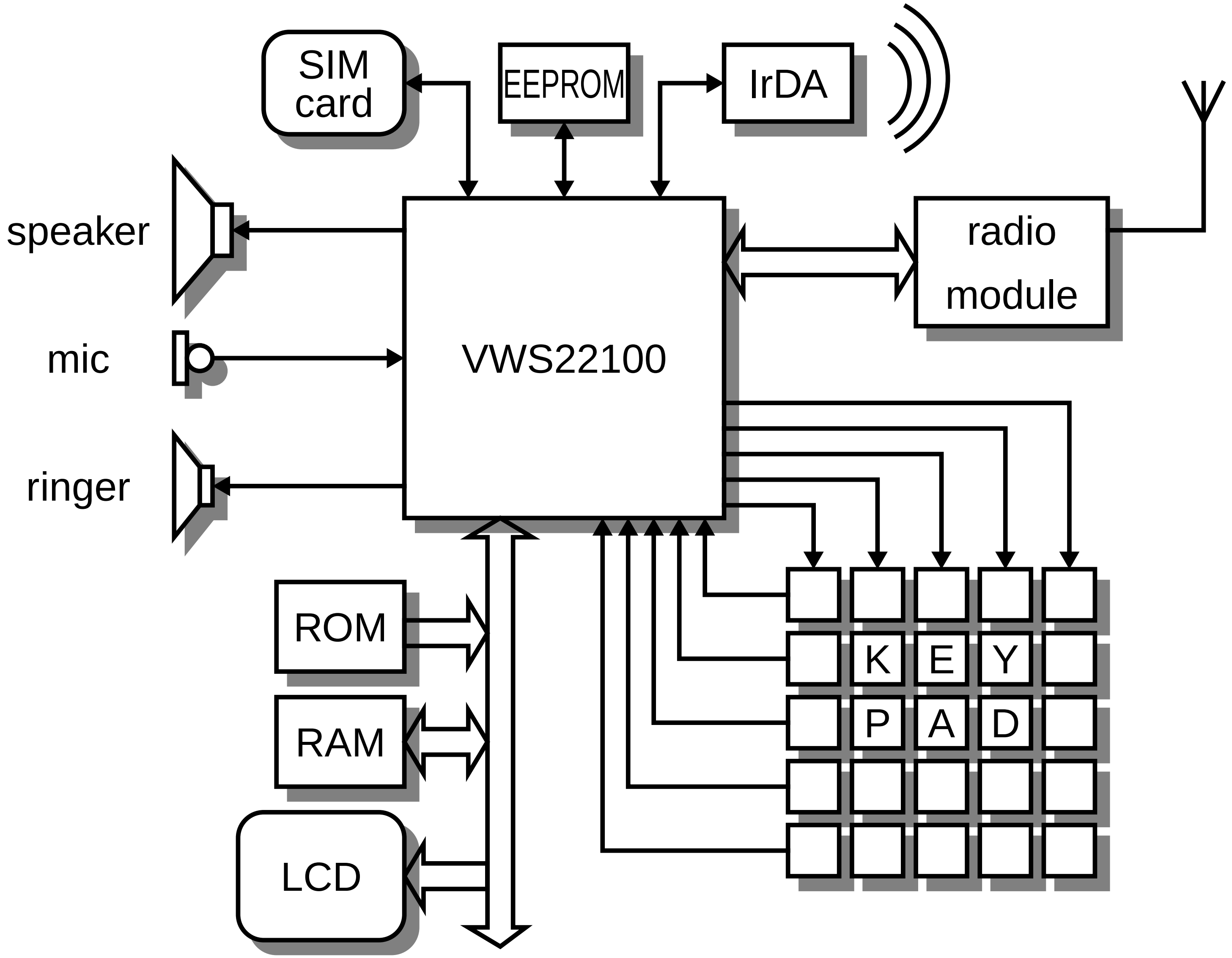
VLSI OneC GSM chip



MANCHEStER 1824

The University of Manchester

Typical OneC system configuration



DRACO

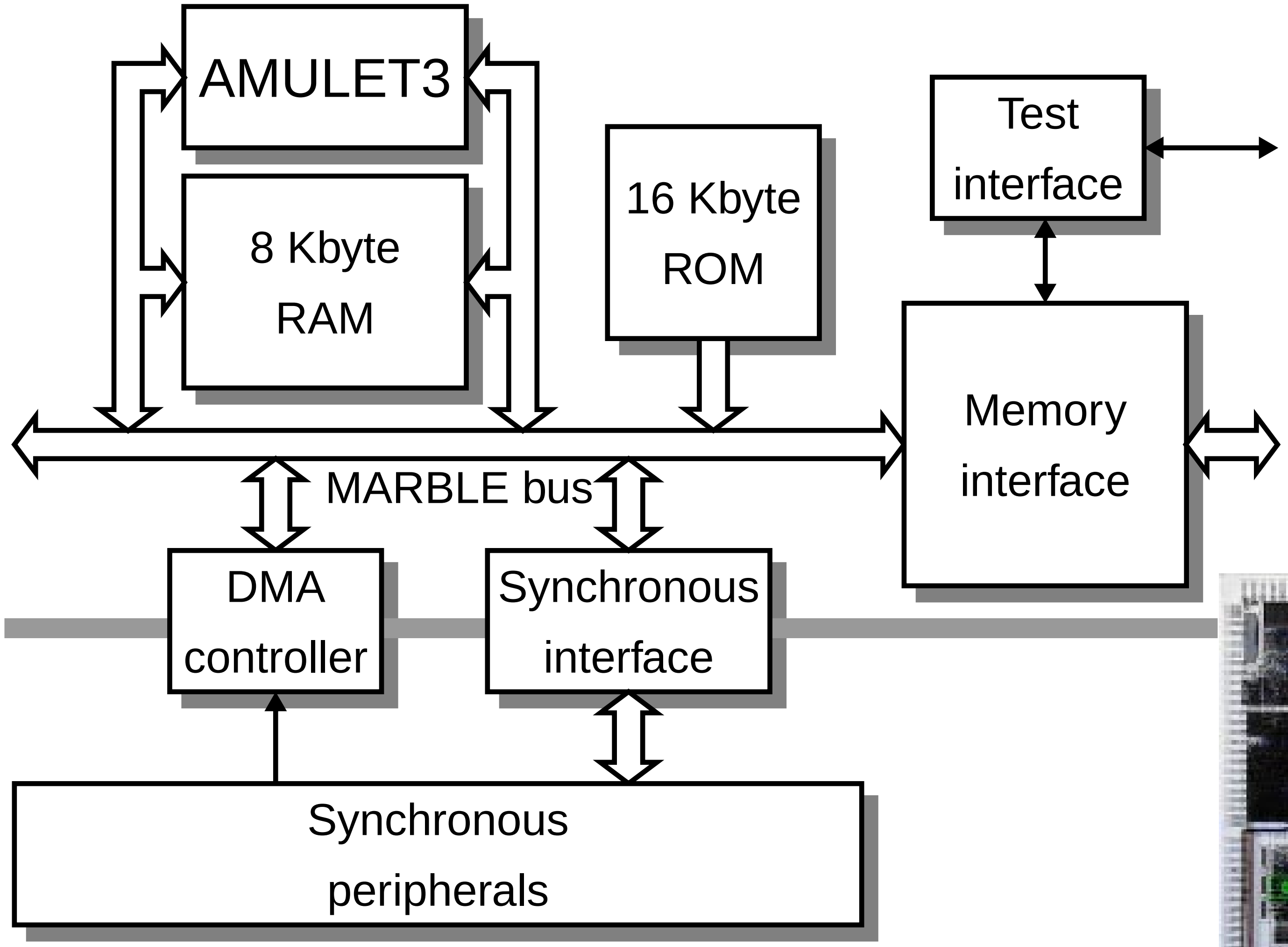
- **D**ECT **Ra**dio **Co**mmunications Controller
  - in collaboration with Hagenuk GmbH
  - combines ISDN and DECT telecommunications systems
  - world’s first “commercial” 32-bit asynchronous SoC product
    - … would have been …

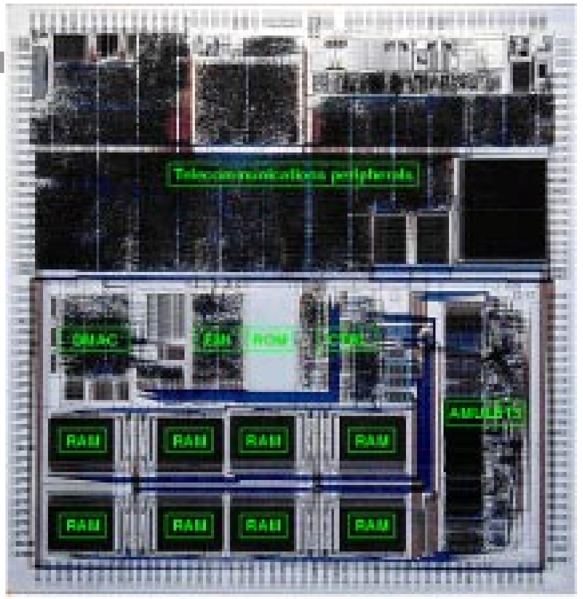
Process 0.35 µm	Transistors 825,000	MIPS 100
Metal layers 3	Die area 21 mm^2	Power 215 mW
Vdd 3.3 V	Clock none	MIPS/W 465

MANCHEstER
1824

The University
of Manchester

DRACO





Fabricated 2000

MANCHEstER
1824

The University
of Manchester

Hands-on: system modelling

- ❑ Using the ARMulator
  - ❍ to generate address traces
  - ❍ to get performance estimates
    - – using the memory map facility
  - ❍ advanced configuration
    - – adding your own system models

☞ Follow the 'Hands-on' instructions