MANCHEstER 1824

The University of Manchester

Coprocessors

- Outline:
  - the ARM coprocessor interface
  - floating-point support
  - MOVE coprocessor
  - CP15, CP14

  ☞ hands-on: system software - semaphores

MANCHEstER 1824

The University of Manchester

Coprocessors

- Outline:
  - ➜ **the ARM coprocessor interface**
  - floating-point support
  - MOVE coprocessor
  - CP15, CP14

  - ☞ hands-on: system software - semaphores

Coprocessors

- ARM supports a generic extension of its instruction set through coprocessors
  - coprocessors have:
    - private registers and data types
    - their own interpretation of instructions
  - Example coprocessors:
    - hardware floating point - the VFP10
    - on-chip cache and MMU control
    - application specific (e.g. MOVE®)

MANCHEstER 1824

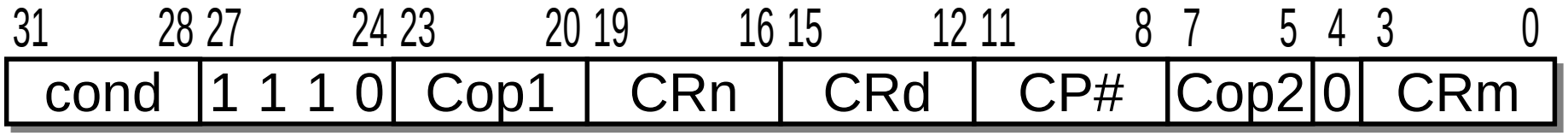
The University of Manchester

Coprocessor instructions

- follow the ARM load/store model:
  - coprocessor data processing instructions
    - operate on values in coprocessor registers
  - coprocessor data transfer instructions
    - move values between memory and coprocessor registers
- and in addition:
  - coprocessor register transfers
    - move values between ARM and coprocessor registers

Coprocessor instructions

- Coprocessor data processing instructions

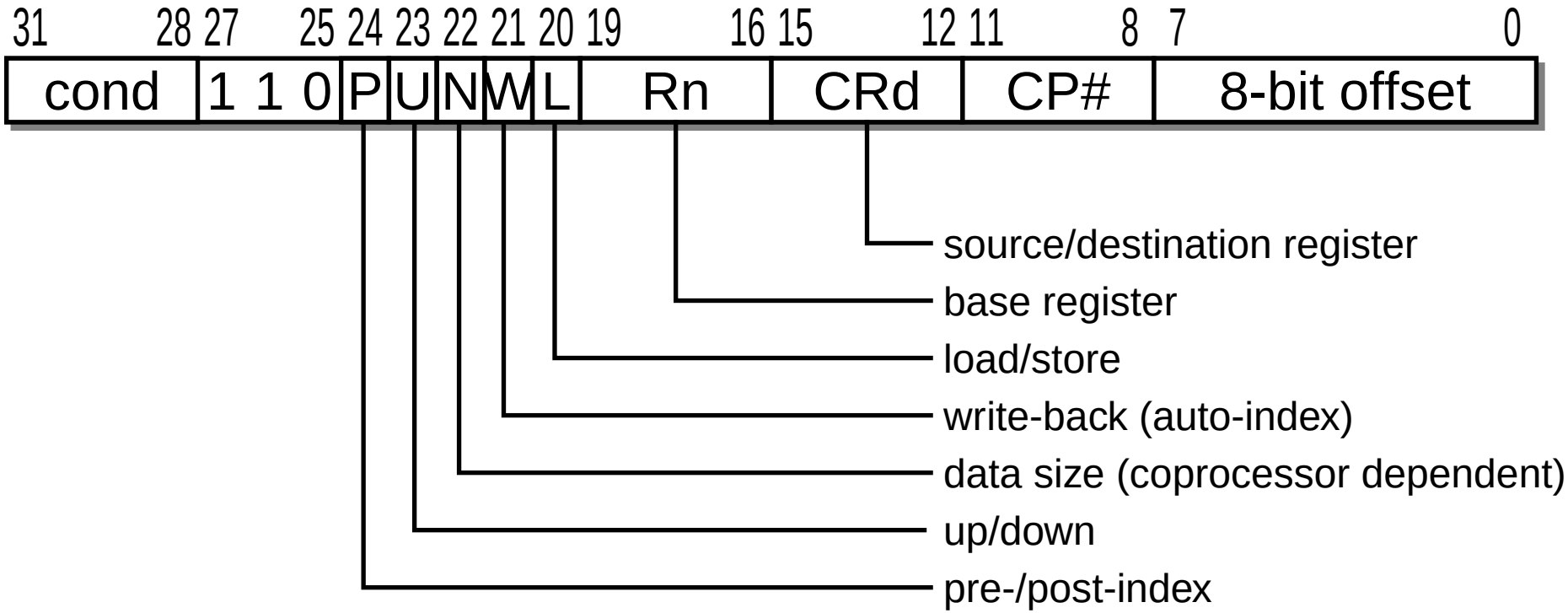



```
CDP{<cond>} Pcp, Cop1, CRd, CRn, CRm, Cop2
```

- CP# specifies the coprocessor number:
  - it performs the operation specified by Cop1 and Cop2 on data in CRn and CRm, putting the result in CRd
  - other interpretations are possible!

Coprocessor instructions

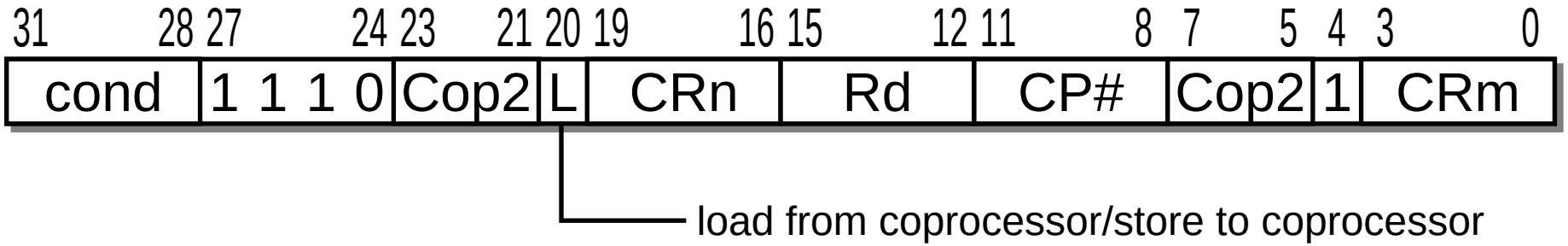
- Coprocessor data transfer instructions




```
LDC{<cond>}{L} Pcp, CRd, <addressing mode>
STC{<cond>}{L} Pcp, CRd, <addressing mode>
```

Coprocessor instructions

- Coprocessor register transfer instructions

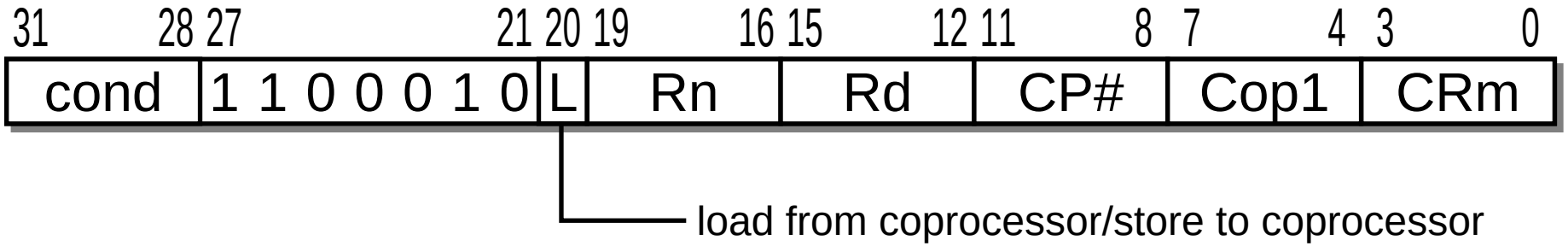



```
MRC{<cond>} Pcp, Cop1, Rd, CRn, CRm{, Cop2}
MRC{<cond>} Pcp, Cop1, Rd, CRn, CRm{, Cop2}
```

  - move a 32-bit value between the coprocessor and ARM
    - e.g.: floating-point FIX, FLOAT and compare
    - if Rd = r15, load is to CPSR (flags only)

Coprocessor instructions

- Coprocessor register transfer instructions



  - move a 64-bit value between the coprocessor and ARM
  - e.g. `FMDRR Dm, Rd, Rn`
    - Floating point 64-bit move Rd := lower half of Dm
      Rn := upper half of Dm
      - note: registers are specified independently

Coprocessor instructions

- Later ARMs (from v5) also have:
  - CDP2
  - LDC2/STC2
  - MCR2/MCR2
  - MCRR2/MRCC2
- These are the same as above except:
  - they use the former ‘NV’ (1111) condition
  - they are always, unconditionally, executed

MANCHEstER 1824

The University of Manchester

Coprocessor mnemonics

- Generic coprocessor mnemonics specify the fields in the instructions

e.g.
```
MCR    p10, 0, R1, CR2, 0      ;
```

- Sometimes more informative forms exist!

```
FMSR   S4, R1                  ; FP Move R1 to S2
```

  - p10 is the (single precision) floating point coprocessor

- Other classes or operations have similar syntax

MANCHEstER 1824

The University of Manchester

Coprocessor interface

- Coprocessors are attached to the ARM memory bus. They:
  - watch the instruction traffic on the bus
  - copy instructions into a pipeline
    - which mimics ARMs instruction pipeline
  - execute those instructions with the right CP#
    - though they may also decline to do so

Coprocessor interface

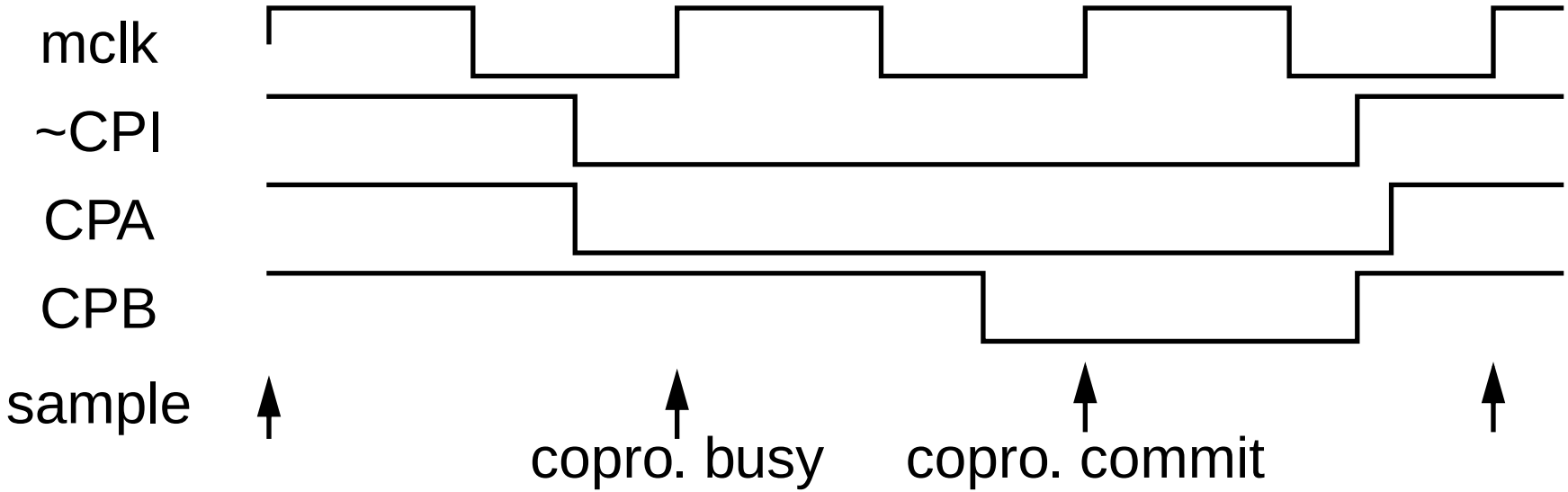
- Issues:
  - not all instructions entering the ARM pipeline are executed
    - those following a branch are not
  - coprocessor instructions are conditionally executed
  - the coprocessor may be absent
    - if present, it may be busy
  - the coprocessor controls the data types

Coprocessor interface

- ~CPI - from ARM to all coprocessors
  - coprocessor instruction
  - ARM has identified a coprocessor instruction and wishes to execute it
- CPA - from the coprocessor(s) to ARM
  - coprocessor absent
  - no coprocessor present can execute it
- CPB - from the coprocessor(s) to ARM
  - coprocessor busy
  - a coprocessor can execute it, but not yet

Coprocessor interface

- Interface timing
  - shows coprocessor busy, then available



~CPI	CPA	CPB	Meaning	Action
1	–	–	Not a (taken) coprocessor operation.	Do nothing
0	1	–	No coprocessor recognises this operation	Illegal instruction trap
0	0	1	Coprocessor may accept instruction in future	Stall pipeline
0	0	0	Coprocessor committed to operation	Coprocessor operation

MANCHEstER 1824

The University of Manchester

Coprocessors

- Outline:
  - the ARM coprocessor interface
  - ➜ **floating-point support**
  - MOVE coprocessor
  - CP15, CP14

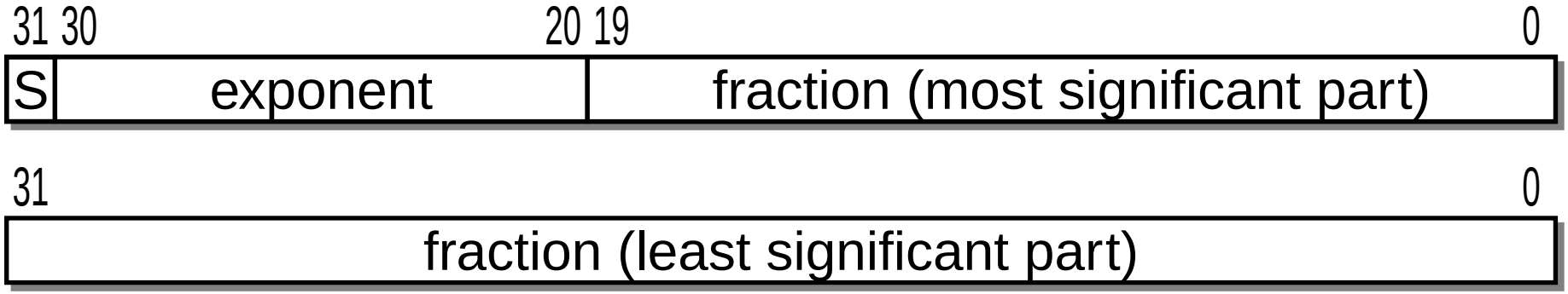
  - ☞ hands-on: system software - semaphores

Floating-point data types

- single precision:

  - value = $(-1)^S$ x 1.fraction x $2^{(exponent-127)}$

- double precision:



MANCHEstER 1824

The University of Manchester

Floating point support

- Floating-point instructions
  - map into the coprocessor instruction space
  - will use coprocessor if present …
  - … otherwise trap into software emulator
- Floating-point library
  - will not use coprocessor even if present
  - faster than software emulator
  - can be called from Thumb code
    - Thumb has no coprocessor instructions

Floating point support

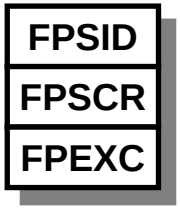
- VFP10
  - vector floating-point unit
    - includes vector instructions that perform multiple operations
  - exploits ARM1020Es 64-bit cache interface
    - and later …
  - can deliver 800 MFLOPS at 400 MHz
    - one load/store and one arithmetic operation per clock cycle (in vector mode)

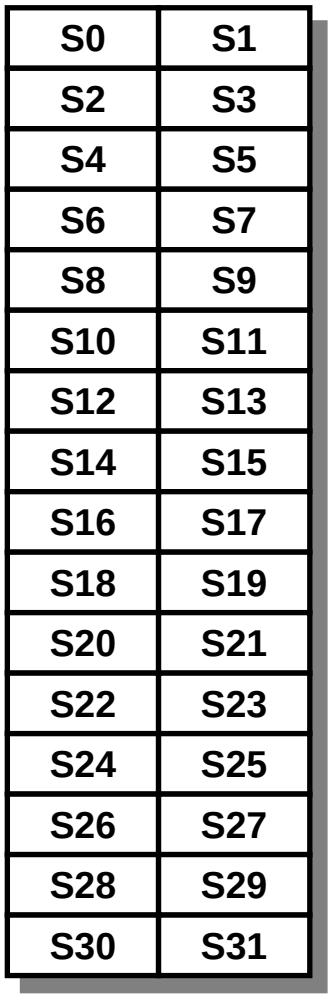
Floating point support

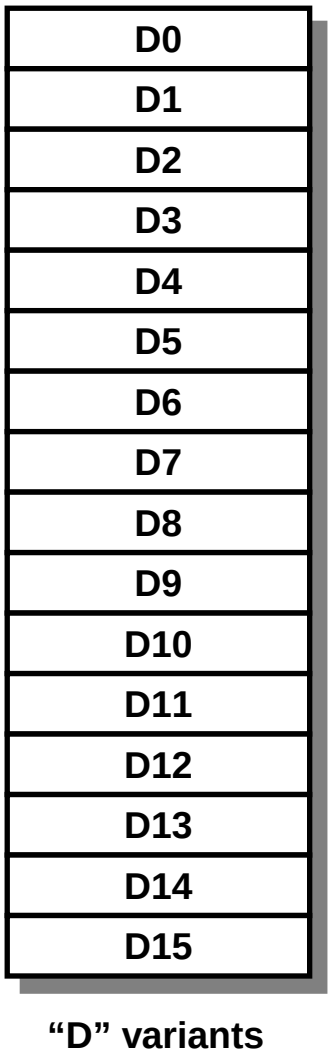
- VFP10 is coprocessor number 10
  - also CP11 if double precision implemented
- IEEE 754 subset
  - supports single (32-) and (possibly) double (64-) bit fp formats
  - most functions in hardware
    - does not support
      - remainder
      - binary ⇔ decimal
      - round-to-integer

VFP architecture

- 32 single precision registers
  - may overlap 16 double precision registers (bit mappings vary)
- Each ‘S’ register can hold: a single precision float or a 32-bit integer
- Can process short vectors as well as scalars
  - up to 8 single precision
  - up to 4 double precision



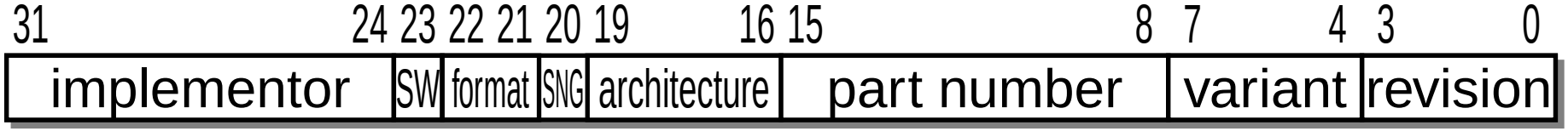




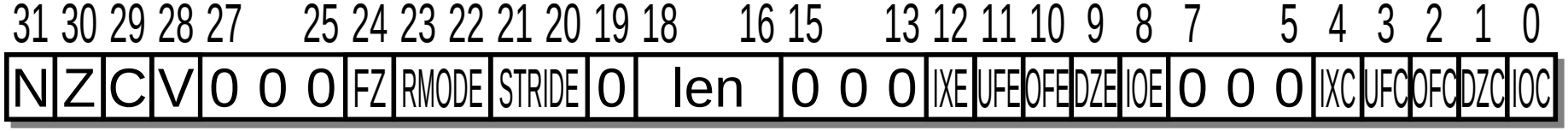
“D” variants

VFP system registers

- Floating point system ID register (FPSID)



- Floating point status and control register (FPSCR)



  - flags, rounding, vector length, exception control …

- Floating point exception register (FPEXC)

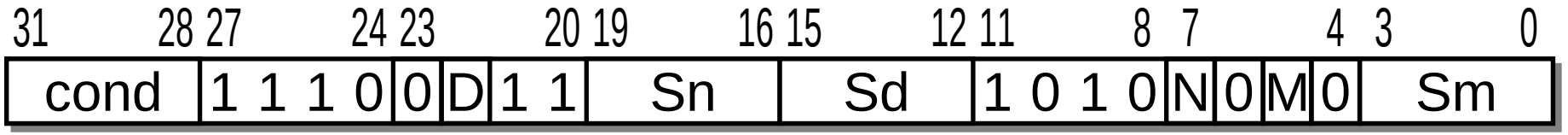
Example VFP instructions

- Load and store from/to memory
  - includes some multiple register moves
- Transfers from/to ARM registers
- Copy/negate/absolute value
  Add/subtract/multiply/divide/square root
  - single values or short vectors
- Comparisons
- Floating point/integer conversion

Example VFP instructions

- Floating point Add, Single precision

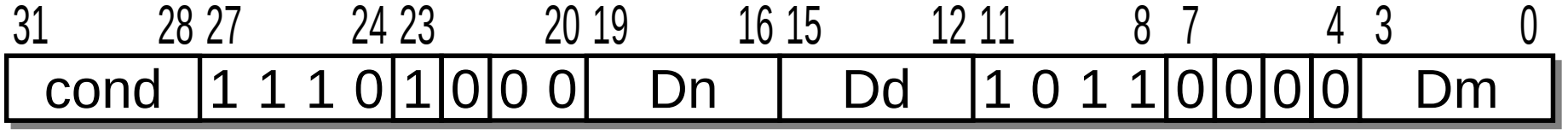
```
FADDS{<cond>}  <Sd>, <Sn>, <Sm>
```



  - {D, N, M} are LSBs of register specifiers

- Floating point Divide, Double precision

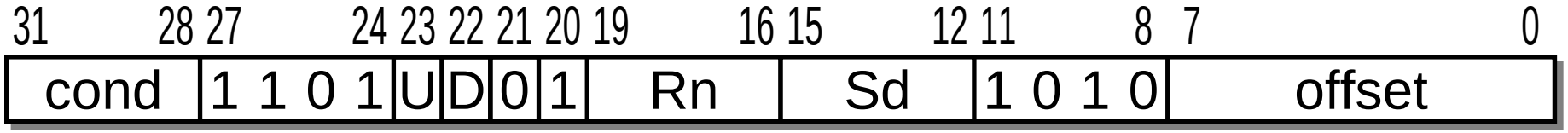
```
FDIVD{<cond>}  <Dd>, <Dn>, <Dm>
```



Example VFP instructions

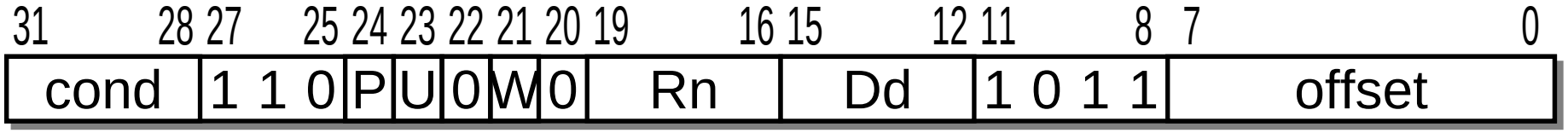
- Floating point Load, Single precision

```
FLDS{<cond>}  <Sd>, [<Rn>, #offset*4]
```



- Floating point Load Multiple, Double precision

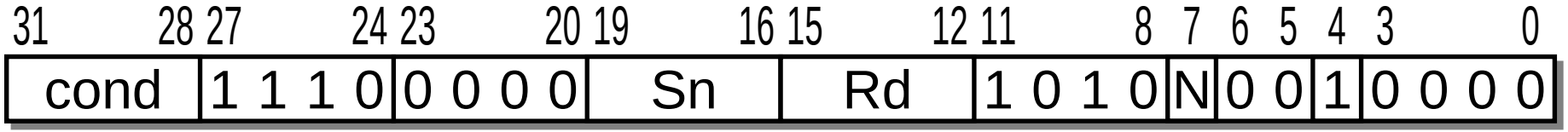
```
FLDM<mode>D{<cond>}  Rn{!}, <registers>
```



Example VFP instructions

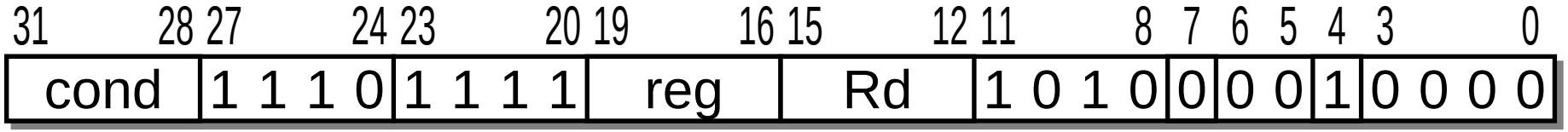
- Floating point Move, Single precision from Register

```
FMSR{<cond>}   <Sn>, <Rd>
```



- Floating point Move to Register from System Register

```
FMRX{<cond>}   <Rd>, <reg>
```



  - <reg> = {FPSID, FPSCR, FPEXC}

MANCHEstER 1824

The University of Manchester

Coprocessors

- Outline:
  - the ARM coprocessor interface
  - floating-point support
  - ➜ **MOVE coprocessor**
  - CP15, CP14

  - ☞ hands-on: system software - semaphores

MANCHEstER 1824

The University of Manchester

MOVE® coprocessor

- A video encoding acceleration coprocessor
  - to accelerate Motion Estimation (e.g. MPEG)
  - implements an 8 x 8 byte ‘block buffer’
  - major function is SAD (Sum of Absolute Differences)
    - compare 8 x 8 pixel blocks
  - has own mnemonics (starting with ‘U’)

MANCHEstER 1824

The University of Manchester

Coprocessors

- Outline:
  - the ARM coprocessor interface
  - floating-point support
  - MOVE coprocessor
  - ➜ **CP15, CP14**

  - ☞ hands-on: system software - semaphores

CP15, CP14

- These two ‘coprocessors’ are interfaces to system functions
  - outside the memory space
- **CP15** is the **system control** coprocessor
  - cache, MMU, paging, etc. control and status
  - more details in ‘Memory Hierarchy Support’ section
- **CP14** is the **debug** coprocessor
  - breakpoint, watchpoint, etc. control and status
  - more details in ‘System Development’ section

MANCHEstER 1824

The University of Manchester

Hands-on: System Software Semaphores

- Look further into ARM system software issues
  - Use a semaphore to perform atomic I/O

  ☞ Follow the ‘Hands-on’ instructions