## ISA Binary Encoding (5A)

Copyright (c) 2014-2019 Young W. Lim.
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.
This document was produced by using LibreOffice.

## Based on

ARM System-on-Chip Architecture, $2^{\text {nd }}$ ed, Steve Furber

| Rn | $1^{\text {st }}$ Operand Reg / Base Reg |
| :--- | :--- |
| Rm | $2^{\text {nd }}$ Operand Reg / Operand Reg / Offset Reg / Source Reg |
| Rd | Left most Reg : Source / Destination Reg |
| S | Set Condition Codes / Signed / Restore PSR and force user bit |
| P | Pre/Post Index |
| $\mathbf{U}$ | Up/Down |
| B | Unsigned Byte/Word |
| H | Half-word Address |
| L | Link / Load/Store |
| W | Write-back (auto-index) |
| Opcode | 4-bit op codes |
| Sh | Shift type |
| R | CPSR/SPSR |


| Rn | $1^{\text {st }}$ Operand Reg / Base Reg |
| :--- | :--- |
| Rm | $2^{\text {nd }}$ Operand Reg / Operand Reg / Offset Reg / Source Reg |
| Rd | Source / Destination Reg |
| $\mathbf{S}$ | Set Condition Codes / Signed / Restore PSR and force user bit |

B Unsigned Byte/Word
H Half-word Address
L Link / Load/Store

T Selects the user view in the non-usermodes

| <cond> | Conditions for conditional execution |
| :--- | :--- |
| <shift> | Shift type and the shift amount (except RRX) |
| CPSR | Current Program Status Register |
| SPSR | Saved Program Status Register |
| RdHi | The most significant 32-bits |
| RdLo | The least significant 32-bits |

## ARM

| <CP\#> | Coprocessor number |
| :--- | :--- |
| <Cop1> | Coprocessor operation 1 |
| <Cop2> | Coprocessor operation 2 |
| CRd | Coprocessor Rd |
| CRn | Coprocessor Rn |
| CRm | Coprocessor Rm |

## All listings (1)



ISA (5A)
Binary Encoding

## All listings (2)

(1) Data Processing / PSR Transfer
(2) Multiply
(3) Multiply Long
(4) Single Data Swap
(5) Branch and Exchange
(6) Halfword Data Transfer: register offset
(8) Single Data Transfer
(9) Undefined
(10) Block Data Transfer
(11) Branch
(12) Coprocessor Data Transfer
(13) Coprocessor Data Operation
(14) Coprocessor Register Transfer
(15) Software Interrupt

| Rd := Rn <op> operand2 (shifted) |
| :---: |
| $\mathbf{R d}$ := Rm * Rs + Rn |
| RdHi : RdLo := Rm * Rs + RdHi : RdLo |
| Rd := [Rn]; [Rn]:= Rm |
| PC : $=\mathrm{Rn}$; ( $\mathrm{Rn}[0]=1$ Thumb, else ARM) |
| Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm |
| Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset |
| Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset |
| [Rn, i] :=: \{Register List\} |
| R14 := PC+8; PC := Offset |
| CRd :=: [Rn,Offset]; CRd :=: [Rn], Offset |
| CRd :=: CRn <CP Opc, CP> CRm |
| Rd :=: CRn <CP Opc, CP> CRm |

Rd := Rn <op> operand2 (shifted)
Rd:= Rm * Rs + Rn
RdHi : RdLo := Rm *Rs + RdHi : RdLo
Rd := [Rn]; [Rn]:= Rm
PC := Rn; (Rn[0]=1 Thumb, else ARM)
Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset
[Rn, i] :=: \{Register List\}
R14 := PC+8; PC := Offset
CRd :=: [Rn,Offset]; CRd :=: [Rn], Offset
CRd :=: CRn <CP Opc, CP> CRm
Rd :=: CRn <CP Opc, CP> CRm

## All listings (3)

(1) Data Processing / PSR Transfer
(2) Multiply
(3)

Multiply Long
Single Data Swap
Branch and Exchange
Halfword Data Transfer: register offset
Halfword Data Transfer: immediate offset
Single Data Transfer
Undefined
Block Data Transfer
Branch
Coprocessor Data Transfer
Coprocessor Data Operation
Coprocessor Register Transfer
Software Interrupt

I, Opcode, S, Rn, Rd, Operand2
A, S, Rd, Rn, Rs, Rm
U, A, S, RdHi, RdLo, Rs, Rm
B, Rn, Rd, Rm
Rm
P,U,W, L, Rn, Rd, S, H, Rm
P,U, W, L, Rn, Rd, Offset, S, H, Offset
P, U, B, W, L, Rn, Rd, Offset

P, U, S, W, L, Rn, Register List
L, Offset
P, U, N, W, L, Rn , CRd, CP\#, Offset
CP Opc, CRn, CRd, CP\#, CP, CRm
CP Opc, L, CRn, Rd, CP\# CP, CRm
Software Interrupt

## 1. Data Processing Instructions



Data Processing / PSR Transfer Rd:= Rn <op>operand2 (shifted)

| cond | 0 | 0 | 0 | 0 | 0 | 0 | A | S | Rd | Rn | Rs | 1 | 0 | 0 | 1 | Rm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Multiply $\quad$ Rd := Rm * Rs + Rn |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| cond | 0 | 0 | 0 | 0 | 1 | U | A | S | RdHi | RdLo | Rs | 1 | 0 | 0 | 1 | Rm |

\# Immediate Operand
S Set Condition Codes
A Accumulate
U Unsigned
B Unsigned Byte/Word
$\mathbf{R n} \quad$ Operand (1 $1^{\text {st }} /$ Adder) / Base Reg
Rm Operand (Multiplicand) / Source Reg
Rs Operand (Multiplier) Reg
Rd Destination Reg
RdHi Destination Reg
RdLo Destination Reg

## 2. Data Transfer Instructions



Single Data Transfer
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset
$\left.\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|}\hline \text { cond } & 0 & 0 & 0 & 1 & 0 & \mathbf{B} & \mathbf{0} & \mathbf{0} & \mathbf{R n} & \mathbf{R d} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & \text { Rm } \\ \hline \text { Single Data Swap }\end{array}\right]=[\mathrm{Rn}] ;[\mathrm{Rn}]:=\mathrm{Rm}$

Single Data Swap
Rd := [Rn]; [Rn]:= Rm
$\square$Rn

| Rd | 0 | 0 | 0 | 0 | 1 | $\mathbf{S}$ | $\mathbf{H}$ | 1 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Rm
Halfword Data Transfer: register offset
Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm

| cond | 0 | 0 | 0 | $\mathbf{P}$ | $\mathbf{U}$ | 1 | $\mathbf{W}$ | $\mathbf{L}$ | Rn | Rd | offsetH | 1 | $\mathbf{S}$ | $\mathbf{H}$ | 1 | offsetL |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Halfword Data Transfer: immediate offset

> Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

| cond | 1 | 0 | 0 | P | U | S | W |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Rn
Register list

Block Data Transfer

P Pre/Post Index
U Up/Down
B Unsigned Byte/Word
w Write-back (auto-index)
L Load/Store
S PSR \& force user mode
[Rn, i] :=: \{Register List\}

Rn Base Reg
Rm Offset Reg
Rd Source/Destination Reg
S Signed
H Half-word Address

## 3. Branch and Branch Exchange



[^0]
## 4. Coprocessor Instructions



Coprocessor Data Transfer
CRd :=: [Rn,Offset]; CRd :=: [Rn], Offset

| cond | 1 | 1 | 1 | 0 | CP Opc | CRn | CRd | CP\# | CP | 0 | CRm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Coprocessor Data Operation

Coprocessor Data Operation
CRd :=: CRn <CP Opc, CP> CRm

| cond | 1 | 1 | 1 | 0 | CP Opc | L | CRn | Rd | CP\# | CP | 1 | CRm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Coprocessor Register Transfer
Rd :=: CRn <CP Opc, CP> CRm

| P | Pre/Post Index |
| :--- | :--- |
| $\mathbf{U}$ | Up/Down |
| $\mathbf{N}$ | Transfer Length |
| $\mathbf{W}$ | Write-back (auto-index) |
| $\mathbf{L}$ | Load/Store |


| <CP\#> | Coprocessor number |
| :--- | :--- |
| <Cop1> | Coprocessor operation 1 |
| <Cop2> | Coprocessor operation 2 |
| CRd | Coprocessor Rd |
| CRn | Coprocessor Rn |
| CRm | Coprocessor Rm |
| Rn | Base Reg |
| Rd | Destination Reg |

## 5. Miscellaneous Instructions

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| cond | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 | 1 |  |  |  |  |  |  |

Undefined

cond | 1 | 1 | 1 | 1 | Ignored by processor |
| :--- | :--- | :--- | :--- | :--- | :--- |

Software Interrupt

## Condition Codes

```
31
```

```
    cond
```

| 0 | 0 | 0 | 0 | EQ | EQual / EQuals zero | Z ¢1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 0 | 0 | 0 | 1 | NE | Not Equal | $\mathrm{Z} \leftarrow 0$ |
| 0 | 0 | 1 | 0 | CS/HS | Carry Set / unsigned High or Same | $\mathrm{C} \leftarrow 1$ |
| 0 | 0 | 1 | 1 | CC/LO | Carry Clear / unsigned Lower | $\mathrm{C} \leftarrow 0$ |
| 0 | 1 | 0 | 0 | MI | MInus / negative | $\mathrm{N} \leftarrow 1$ |
| 0 | 1 | 0 | 1 | PL | PLus / positive or zero | $\mathrm{N} \leftarrow 0$ |
| 0 | 1 | 1 | 0 | VS | oVerflow Set | $V \leftarrow 1$ |
| 0 | 1 | 1 | 1 | VC | oVerflow Clear | $\mathrm{V} \leftarrow 0$ |
| 1 | 0 | 0 | 0 | HI | unsigned HIgher | $C \leftarrow 1, z \leftarrow 0$ |
| 1 | 0 | 0 | 1 | LS | unsigned Lower or Same | $\mathrm{C} \leftarrow 0, \mathrm{z} \leftarrow 1$ |
| 1 | 0 | 1 | 0 | GE | signed Greater than or Equal | $\mathrm{N}==\mathrm{V}$ |
| 1 | 0 | 1 | 1 | LT | signed Less Than | N ! = V |
| 1 | 1 | 0 | 0 | GT | signed Greater Than | $\mathrm{Z} \leftarrow 0, \mathrm{~N}==\mathrm{V}$ |
| 1 | 1 | 0 | 1 | LE | signed Less than or Equal | $\mathrm{Z} \leftarrow 1, \mathrm{~N}!=\mathrm{V}$ |
| 1 | 1 | 1 | 0 | AL | ALways | any |
| 1 | 1 | 1 | 1 | NV | NeVer (do not use?) | none |

## Branch and Branch with Link ( $\mathrm{B}, \mathrm{BL}$ )



| cond | 1 | 0 | 1 | $\mathbf{0}$ | 24-bit signed word offset |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| Branch | B $\langle<$ cond $>\}<$ target address> | PC := Offset |  |  |  |


| cond | 1 | 0 | 1 | $\mathbf{1}$ | 24-bit signed word offset |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Branch with Link | BL\{<cond>\} <target_address> |  | R14 := PC+8; PC := Offset |  |  |

## Branch, Branch with Link and eXchange (BX, BLX)



| cond | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | Rn |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :---: |
| Branch and Exchange |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |


| cond | 0 |  | 0 | 0 | 1 | 0 | 0 | $0{ }^{0} 1$ | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | Rn |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| ch with |  |  |  |  | an |  |  |  | X\{c | cond | d>\} | Rn |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |


| $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{H}$ |  | Offset |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| Branch with Link and Exchange |  | BLX\{<cond $>\}$ <address> | PC : $:=$ Offset |  |  |  |  |  |  |

H Half Word Address

## Data Processing Instructions



## Data Processing Instructions

| 31 | $30 \quad 29$ | 28 | 27 | 26 | 25 | 24 | $23 \quad 22$ | 2120 | 19 | 1817 | 16 | 15 | 1413 | 12 | 11 | 10 | 9 | 8 | 7 | , | 5 | 4 | 3 | 2 | 1 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | cond |  | 0 | 0 | \# |  | opcode | S |  | Rn |  |  | Rd |  |  |  |  |  |  | er | nd |  |  |  |  |  |
| Data Processing / PSR Transfer |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1
Data Processing with an immediate operand

0
Data Processing with a shifted operand - shift amount

0
Data Processing with a shifted operand - shift register

| \# | Immediate Operand | Rn | $1^{\text {st }}$ Operand Reg |
| :--- | :--- | :--- | :--- |
| S | Set Condition Codes | $\mathbf{R m}$ | $2^{\text {nd }}$ Operand Reg |
| rot\# | Shift applied to the 8-bit immeidate | Rs | Shift (amount) Reg |
| \#shift | Shift amount (5-bit immediate) | Rd | Destination Reg |
| Sh | Shift Type |  |  |

## Data Processing Instructions - a shifted operand



0
Data Processing with a shifted operand - shift amount

0

Data Processing with a shifted operand - shift register
<Sh> \{cond\}\{S\} Rd, Rm, Rs
<Sh> \{cond\}\{S\} Rd, Rm, \#n
RRX \{cond\}\{S\} Rd, Rm

| \#shift | Sh | 0 | Rm |
| :---: | :---: | :---: | :---: |


| Rs | $\mathbf{0}$ | Sh | 1 | Rm |
| :---: | :---: | :---: | :---: | :---: |


| Sh | Shift Type |
| :---: | :--- |
| $\mathbf{0 0}$ | Logical Left |
| $\mathbf{0 1}$ | Logical Right |
| $\mathbf{1 0}$ | Arithmetic Right |
| $\mathbf{1 1}$ | Rotate Right |

## Data Processing Instructions - operand2 examples



Data Processing with an immediate operand
0xA_05 // rotate 0x05 right by 20 (=2 * 0xA)
0xB_14 // rotate 0x14 right by 22 (=2 * 0xB) $0 x C \_50 / /$ rotate $0 \times 50$ right by 24 ( $=2$ * 0xC)

| cond | 0 | 0 | 0 | opcode | S | Rn | Rd | Shift | Rm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

0
Data Processing with a shifted operand - shift amount

Data Processing with a shifted operand - shift register

ADD r3, r2, r1, LSL \#3 ; r3 := r2 + r1*2^3
MOV r0, r0, LSR \#2 ; rO := r0/2^2

ADD r5, r5, r3, LSL r2
; r5 := r5 + r3*2^r2

## Data Processing Opcode

| cond | 0 | 0 | I | opcode | S |
| :---: | :---: | :---: | :---: | :---: | :---: |
| cond | 0 | 0 | \# | 0000 | S |
| cond | 0 | 0 | \# | 0001 | S |
| cond | 0 | 0 | \# | 0010 | S |
| cond | 0 | 0 | \# | 0011 | S |
| cond | 0 | 0 | \# | 0100 | S |
| cond | 0 | 0 | \# | 0101 | S |
| cond | 0 | 0 | \# | 0110 | S |
| cond | 0 | 0 | \# | 0111 | S |
| cond | 0 | 0 | \# | 1000 | S |
| cond | 0 | 0 | \# | 1001 | S |
| cond | 0 | 0 | \# | 1010 | S |
| cond | 0 | 0 | \# | 1011 | S |
| cond | 0 | 0 | \# | 1100 | S |
| cond | 0 | 0 | \# | 1101 | S |
| cond | 0 | 0 | \# | 1110 | S |
| cond | 0 | 0 | \# | 1111 | S |


| AND | Logical bit-wise AND | Rd:= Rn AND Op2 |
| :---: | :---: | :---: |
| EOR | Logical bit-wise XOR | Rd := Rn EOR Op2 |
| SUB | Subtract | Rd:= Rn - Op2 |
| RSB | Reverse Subtract | Rd := Op2 - Rn |
| ADD | Add | Rd := Rn + Op2 |
| ADC | Add with carry | $\mathrm{Rd}:=\mathrm{Rn}+\mathrm{Op} 2+\mathrm{C}$ |
| SBC | Subtract with carry | $\mathrm{Rd}:=\mathrm{Rn}-\mathrm{Op} 2+\mathrm{C}-1$ |
| RSC | Reverse subtract with carry | $\mathrm{Rd}:=\mathrm{Op} 2-\mathrm{Rn}+\mathrm{C}-1$ |
| TST | Test | Set CC on Rn AND Op2 |
| TEQ | Test equivalence | Set CC on Rn EOR Op2 |
| CMP | Compare | Set CC on Rn - Op2 |
| CMN | Compare negated | Set CC on Rn + Op2 |
| ORR | Logical bit-wise OR | Rd := Rn OR Op2 |
| MOV | Move | Rd : = Op2 |
| BIC | Bit clear | Rd := Rn AND NOT Op2 |
| MVN | Nive begated | Rd := NOT Op2 |

## PSR Transfer (MRS, MSR)



MRS (Transfer PSR contents to a register)

| cond $\mathbf{0}$ $\mathbf{0}$ $\mathbf{0}$ $\mathbf{1}$ $\mathbf{0}$ $\mathbf{P d}$ $\mathbf{1}$ 0 1 0 0 1 1 1 1 1 0 0 0 0 $\mathbf{0}$ $\mathbf{0}$ $\mathbf{0}$ $\mathbf{0}$ $\mathbf{R m}$ |
| :--- |
| MSR (Transfer register contents to PSR) |


| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\#$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P d}$ | $\mathbf{1}$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | Source Operand |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

MSR (Transfer register contents or immediate value to PSR flag bits only)

| 0 | 0 | 0 | 0 | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{R m}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

## PSR Transfer (MRS, MSR) - decoding

 MRS (Transfer PSR contents to a register)

| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P d}$ | $\mathbf{1}$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | Rm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| MSR (Transfer register contents to PSR) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\#$ | $\#$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P d}$ | $\mathbf{1}$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |  |  |  |  | Source Operand |  |  |  | MSR (Transfer register contents or immediate value to PSR flag bits only)



## Status register to general register transfer instructions

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 14 | 12 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| cond | $\mathbf{O}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P s}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ |  | $\mathbf{R d}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ |  |  |  |  |

MRS (Transfer PSR contents to a register)

## General register to status register transfer instructions



MSR (Transfer register contents to PSR)

| cond | 0 | 0 | \# | 1 | 0 | Pd | 1 | 0 | 1 |  |  | 1 | 1 | 1 | 1 | 1 | Source Operand |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |


| 0 | 0 | 0 | 0 | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{R m}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

## Multiply Instructions

| $31 \quad 3029$ |  | 27 | 26 | 25 | 24 |  | 23 |  | 21 | 20 | 19 | 1817 |  | 15 | 141312 | 11 | 10 | 8 | 7 | 6 | 5 | 4 | 3 | 1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| cond |  | 0 | 0 | 0 |  | 0 | 0 | 0 | A | S |  | Rd |  |  | Rn |  | Rs |  | 1 | 0 | 0 | 1 |  | Rm |
| Multiply | <mul> \{<cond>\}SS\} Rd, Rm, Rs $\{$, Rn\} |  |  |  |  |  |  |  |  |  |  |  |  |  | Rd := Rm * Rs + Rn |  |  |  |  |  |  |  |  |  |
| cond |  | 0 | 0 | 0 |  | 0 | 1 | U | A | S |  | RdHi |  |  | RdLo |  | Rs |  | 1 | 0 | 0 | 1 |  | Rm |
| Multiply Lo | ng |  | <mul> \{<cond>\}\{S\} RdHi, RdLo, Rm, Rs |  |  |  |  |  |  |  |  |  |  |  | RdHi : RdLo := Rm * Rs + RdHi : RdLo |  |  |  |  |  |  |  |  |  |

S Set Condition Codes
Rn Accumulator Operand
A Accumulate
Rm Operand (Multiplicand)
U Unsigned
Rs Operand (Multiplier)
Rd Destination
RdHi Destination (Hi Word)
RdLo Destination (Lo Word)

## Multiply Instruction Opcode



Multiply $\quad$ Rd := Rm * Rs + Rn

| cond | 0 | 0 | 0 | 0 | 1 | U | A | S | RdHi | RdLo | Rs | 1 | 0 | 0 | 1 | Rm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| ultiply Long |  |  |  |  |  |  |  |  |  | RdHi : | * | Rdr | : |  |  |  |

Multiply Long

| 0 | 0 | 0 | MUL | Multiply (32-bit result) | $\mathrm{Rd}:=(\mathrm{Rm} * \mathrm{Rs})[31: 0]$ |
| :--- | :--- | :--- | :--- | :--- | :--- |
| 0 | 0 | 1 | MLA | Multiply-accumulate (32-bit result) | $\mathrm{Rd}:=(\mathrm{Rm} * \mathrm{Rs}+\mathrm{Rn})[31: 0]$ |

## CLZ (Count leading zeros)



## Data Transfer Instructions



Single Data Transfer Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

| cond | 0 | 0 | 0 | 1 | 0 | $\mathbf{B}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{R n}$ | $\mathbf{R d}$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | $\mathbf{R m}$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Single Data Swap
Rd := [Rn]; [Rn]:= Rm


Halfword Data Transfer: register offset
Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm
cond

| $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{P}$ | $\mathbf{U}$ | $\mathbf{1}$ | $\mathbf{W}$ | $\mathbf{L}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Rn


| Rd | offsetH | 1 | $\mathbf{S}$ | $\mathbf{H}$ | 1 | offsetL |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Halfword Data Transfer: immediate offset

> Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

cond $\quad 1 . |$|  | 0 | 0 | $\mathbf{P}$ | $\mathbf{U}$ | $\mathbf{S}$ | $\mathbf{W}$ | $\mathbf{L}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Register list
Block Data Transfer
[Rn, i] :=: \{Register List\}

| P | Pre/Post Index | Rn | Base Reg |
| :--- | :--- | :--- | :--- |
| U | Up/Down | Rm | Offset Reg |
| B | Unsigned Byte/Word | Rd | Source/Destination Reg |
| W | Write-back (auto-index) | S | Signed |
| L | Load/Store | H | Half-word Address |
| S | PSR \& force user mode |  |  |

## Data Transfer Instructions



Single Data Transfer
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

| $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{B}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{R n}$ | $\mathbf{R d}$ | 0 | 0 | 0 | 0 | 1 | $\mathbf{0}$ | $\mathbf{0}$ | 1 | $\mathbf{R m}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Single Data Swap


Halfword Data Transfer: register offset

$$
\begin{array}{|l|l|l|}
\hline 0 & 0 & 0 \\
\hline
\end{array}
$$

Halfword Data Transfer: immediate offset

$$
\begin{array}{|l|l|l|}
\hline 1 & 0 & 0 \\
\hline
\end{array}
$$

Block Data Transfer

P Pre/Post Index
U Up/Down
B Unsigned Byte/Word
W Write-back (auto-index)
L Load/Store
S PSR \& force user mode
Rd := [Rn]; [Rn]:= Rm

| Rd | 0 | 0 | 0 | 0 | 1 | $\mathbf{S}$ | $\mathbf{H}$ | 1 | $\mathbf{R m}$ |
| :---: | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm

| Rd | offsetH | 1 | $\mathbf{S}$ | $\mathbf{H}$ | 1 | offsetL |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

## Register list

[Rn, i] :=: \{Register List\}

Rn Base Reg
Rm Offset Reg
Rd Source/Destination Reg
S Signed
H Half-word Address

## Single word data transfer instructions

| 31 | 3029 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | $18 \quad 17$ | 16 | 15 | 1413 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | cond |  | 0 | 1 | \# | P | U | B | W | L |  | Rn |  |  | Rd |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Single Data Transfer |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Rd :=: [R | , | Off | t]; | d |  | n] | Off |  |  |  |  |  |  |

Single Data Transfer $\quad$ Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

| cond $\mathbf{0}$ $\mathbf{1}$ $\mathbf{0}$ $\mathbf{P}$ $\mathbf{U}$ $\mathbf{B}$ $\mathbf{W}$ $\mathbf{L}$ $\mathbf{R n}$ $\mathbf{R d}$ Immediate offset |
| :--- |



## Half-word data transfer instructions



Halfword Data Transfer: register offset

Halfword Data Transfer: immediate offset
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

## Swap memory and register instruction (SWP)



Single Data Swap
Rd := [Rn]; [Rn]:= Rm

Halfword Data Transfer: register offset

$$
\begin{array}{c|l|l|l|l|l|l|l|c}
\mathbf{R d} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{S} & \mathbf{H} & \mathbf{1}  \tag{6}\\
\mathrm{Rd}:=:[\mathrm{Rn}, \mathrm{Rm}] ; & \mathrm{Rd}:=:[\mathrm{Rn}], \mathrm{Rm}
\end{array}
$$

1 Rm

| SH |  |
| :--- | :--- |
| 00 | SWP instruction |
| 01 | Unsigned Halfwords |
| 10 | Signed Byte |
| 11 | Signed Halfwords |

## Block data transfer instruction



Block Data Transfer
[Rn, i] :=: \{Register List\}

| $\mathbf{P}=\mathbf{1}:$ before | $\mathbf{U}=\mathbf{1}:$ Increment | $\mathbf{W}=\mathbf{1}:$ auto index | $\mathbf{L}=\mathbf{1}:$ LDM |
| :--- | :--- | :--- | :--- |
| $\mathbf{P}=\mathbf{0}:$ after | $\mathbf{U}=\mathbf{0}:$ Decrement | $\mathbf{W}=\mathbf{0}:$ no auto index | $\mathbf{L}=\mathbf{0}: \mathbf{S T M}$ |

P: Pre/Post U: Up/Down W:Write-back (auto index) S: PSR \& force user mode

| IB | PU=11 | DA | PU=00 |
| :--- | :--- | :--- | :--- |
| IA | PU=01 | IA | $P U=01$ |
| DB | PU=10 | DB | $P U=10$ |
| DA | $P U=00$ | IB | $P U=11$ |


| Name | Stack | Block | L | P | U |
| :--- | :--- | :--- | :--- | :--- | :--- |
| Pre-Increment Load | LDMED | LDMIB | 1 | 1 | 1 |
| Post-Increment Load | LDMFD | LDMIA | 1 | 0 | 1 |
| Pre-Decrement Load | LDMEA | LDMDB | 1 | 1 | 0 |
| Post-Decrement Load | LDMFA | LDMDA | 1 | 0 | 0 |
| Pre-Increment Store | STMFA | STMIB | 0 | 1 | 1 |
| Post-Increment Store | STMEA | STMIA | 0 | 0 | 1 |
| Pre-Decrement Store | STMFD | STMDB | 0 | 1 | 0 |
| Post-Decrement Store | STMED | STMDA | 0 | 0 | 0 |

## Block data transfer instruction examples



| IB | PU=11 |
| :--- | :--- |
| IA | PU=01 |
| DB | PU=10 |
| DA | PU=00 |

```
STMIB R8! {R0,R1,R4} LDMDA R8! {R0,R1,R4}
STMFA R8! {R0,R1,R4} LDMFA R8! {R0,R1,R4}
STMIA R8! {R0,R1,R4} LDMDB R8! {R0,R1,R4}
STMEA R8! {R0,R1,R4} LDMEA R8! {R0,R1,R4}
STMDB R8! {R0,R1,R4} LDMIA R8! {R0,R1,R4}
STMFD R8! {R0,R1,R4} LDMFD R8! {R0,R1,R4}
STMDA R8! {R0,R1,R4} LDMIB R8! {R0,R1,R4}
STMED R8! {R0,R1,R4} LDMED R8! {R0,R1,R4}
```


## Coprocessor Instructions



Coprocessor Data Transfer
CRd :=: [Rn,Offset]; CRd :=: [Rn], Offset

| cond | 1 | 1 | 1 | 0 | CP Opc | CRn | CRd | CP\# | CP | 0 | CRm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Coprocessor Data Operation

Coprocessor Data Operation
CRd :=: CRn <CP Opc, CP> CRm

| cond | 1 | 1 | 1 | 0 | CP Opc | $\mathbf{L}$ | CRn | Rd | CP\# | CP | 1 | CRm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Coprocessor Register Transfer

```
Rd :=: CRn <CP Opc, CP> CRm
```

P Pre/Post Index
U Up/Down
N Transfer Length
W Write-back (auto-index)
L Load/Store

| <CP\#> | Coprocessor number |
| :--- | :--- |
| <Cop1> | Coprocessor operation 1 |
| <Cop2> | Coprocessor operation 2 |
| CRd | Coprocessor Rd |
| CRn | Coprocessor Rn |
| CRm | Coprocessor Rm |
| Rn | Base Reg |
| Rd | Destination Reg |

## Coprocessor data operations



## Coprocessor data transfers



## Coprocessor register transfer



## Breakpoint instruction (BKPT)



## SWI (Software Interrupt)



Software Interrupt SWI \{<cond>\}<24-bit immediate>

## Undefined instrucitons



Undefined


## Unused arithmetic instructions




Multiply <mul> \{<cond>\}\{S\} Rd, Rm, Rs \{, Rn\} Rd:= Rm *Rs + Rn


## Unused control instructions

| 29 |  |  | 26 | 25 | 24 | 23 | $22 \quad 21$ | 20 | 19 | 18 | 16 | 15 | 14 | 12 | 11 | 10 |  |  | 6 |  |  | 3 | 21 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| cond |  |  | 0 | 0 | 1 | 0 | op1 | 0 |  | Rn |  |  | Rd |  |  | Rs |  |  | p2 |  | 0 |  | Rm |


| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | op 1 | $\mathbf{0}$ | Rn | Rd | Rs | $\mathbf{0}$ | op 2 | $\mathbf{1}$ | Rm |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | op1 | $\mathbf{0}$ | Rn | Rd | \#rot | 8-bit immediate |  |  |  |



Branch and Exchange PC:=Rn; (Rn[0]=1 Thumb, else ARM)

| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P s}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{1}$ | $\mathbf{R d}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | MRS (Transfer PSR contents to a register)


| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P d}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{1}$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{R m}$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

MSR (Transfer register contents to PSR)

| cond | $\mathbf{0}$ | $\mathbf{0}$ | $\#$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{P d}$ | $\mathbf{1}$ | $\mathbf{0}$ | $\mathbf{1}$ | 0 | 0 | 1 | 1 | 1 | 1 | $\mathbf{1}$ | Source Operand |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |

MSR (Transfer register contents or immediate value to PSR flag bits only)

## Unused load/store instructions

| $31 \quad 30 \quad 29$ | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 65 | 4 | 3 | 21 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| cond |  | 0 | 0 | 0 | P | U | B | W | L |  | R | n |  |  | R |  |  |  |  |  |  | 1 | op1 | 1 |  | Rm |  |



Single Data Swap

$$
\begin{equation*}
\mathrm{Rd}:=[\mathrm{Rn}] ;[\mathrm{Rn}]:=\mathrm{Rm} \tag{4}
\end{equation*}
$$

cond

| $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{P}$ | $\mathbf{U}$ | $\mathbf{0}$ | $\mathbf{W}$ | $\mathbf{L}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | Rn

Rd | 0 | 0 | 0 | 0 | $\mathbf{1}$ | $\mathbf{S}$ | $\mathbf{H}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | 1

Rm
Halfword Data Transfer: register offset
Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm
cond

| $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{P}$ | $\mathbf{U}$ | $\mathbf{1}$ | $\mathbf{W}$ | $\mathbf{L}$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

Rn
Rd
offset $\square$ 1 S H
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

## Unused coprocessor instructions



| $31 \quad 30 \quad 29$ | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 1817 | 16 | 15 | $14 \quad 13$ | 12 | 11 | 109 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| cond |  | 1 | 1 | 0 | P | U | N | W | L |  | Rn |  |  | CRd |  |  | CP\# |  |  |  |  |  |  |  |  |  |

Coprocessor Data Transfer

```
CRd :=: [Rn,Offset]; CRd :=: [Rn], Offset
```


## Undefined instruction space



Undefined


## Decoding Instructions (1)


(1) Data Processing / PSR Transfer
(2) Multiply
(3) Multiply Long
(4) Single Data Swap
(5) Branch and Exchange
(6) Halfword Data Transfer: register offset
(7) Halfword Data Transfer: immediate offset

Rd := Rn <op> operand2 (shifted)
Rd:= Rm * Rs + Rn
RdHi : RdLo := Rm *Rs + RdHi : RdLo
Rd:= [Rn]; [Rn]:= Rm
PC := Rn; (Rn[0]=1 Thumb, else ARM)
Rd :=: [Rn, Rm]; Rd :=: [Rn], Rm
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset

## Decoding Instructions (2)

| $\begin{array}{llll}31 & 30 & 29 & 28\end{array}$ | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 1817 | 16 | 15 | 1413 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 4 | 3 | 2 | 1 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| cond | 0 | 1 | \# | P | U | B | W | L |  | Rn |  |  | Rd |  |  |  |  |  |  | off |  |  |  |  |  |
| cond | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |

(8) Single Data Transfer
Rd :=: [Rn, Offset]; Rd :=: [Rn], Offset
(9) Undefined

## Decoding Instructions (3)


(10) Block Data Transfer
(11) Branch
[Rn, i] :=: \{Register List\}
R14 := PC+8; PC := Offset

## Decoding Instructions (4)


(12) Coprocessor Data Transfer

$$
\begin{aligned}
& \text { CRd :=: [Rn,Offset]; CRd :=: [Rn], Offset } \\
& \text { CRd :=: CRn <CP Opc, CP> CRm } \\
& \text { Rd :=: CRn <CP Opc, CP> CRm }
\end{aligned}
$$

(15) Software Interrupt

## Multiple register transfer instruction



## ARM Exception Handling

## References

[1] ftp://ftp.geoinfo.tuwien.ac.at/navratil/HaskellTutorial.pdf
[2] https://www.umiacs.umd.edu/~hal/docs/daume02yaht.pdf


[^0]:    L Link
    Rn Operand Reg

