Microprocessor & Computer Architecture (µpCA)

UE19CS252

Dr. D. C. Kiran

Department of
Computer Science and Engineering

Microprocessor & Computer Architecture (µpCA)

Instruction Encoding

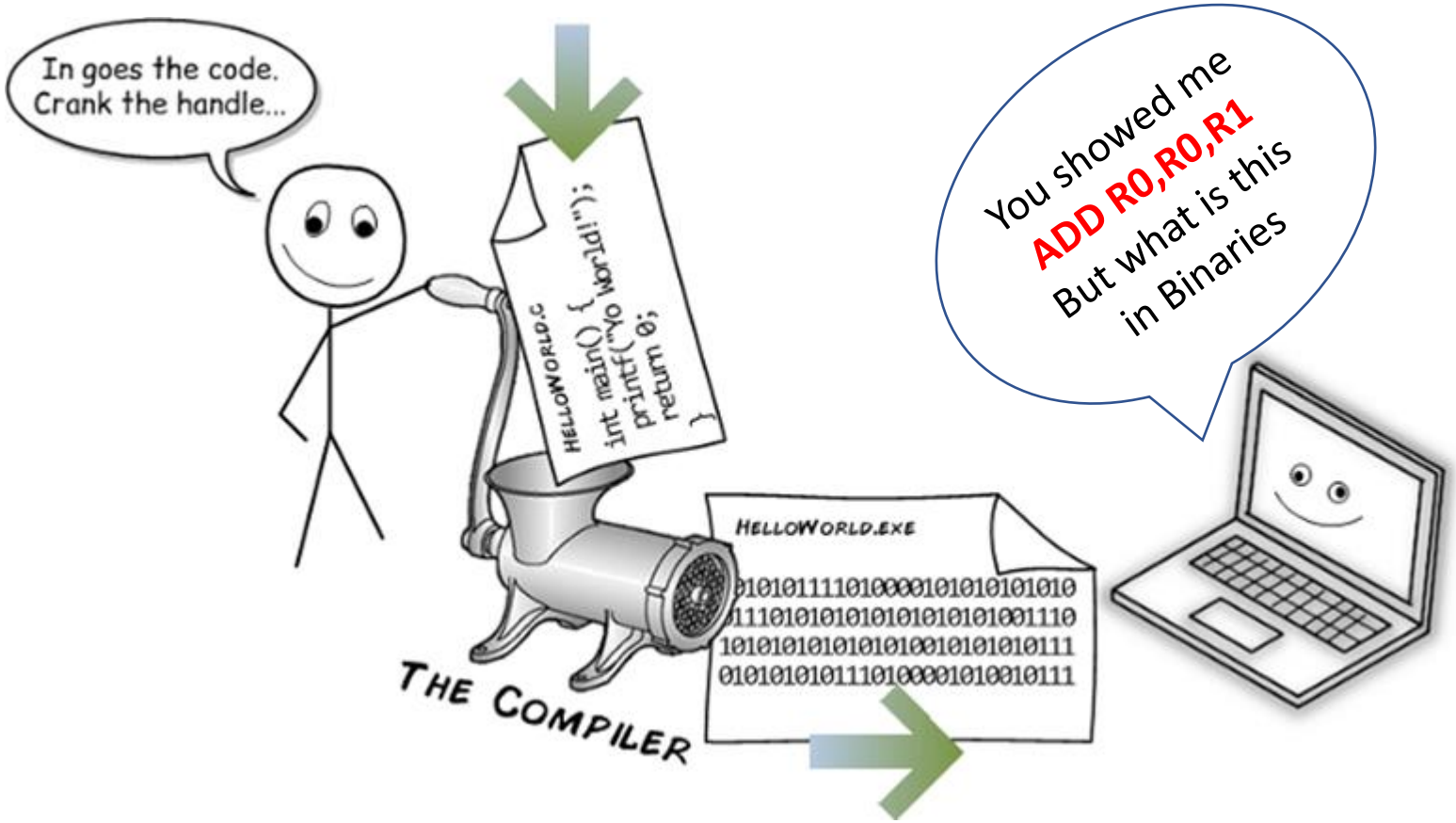
Dr. D. C. Kiran
Department of Computer Science and Engineering

Microprocessor & Computer Architecture (µpCA)

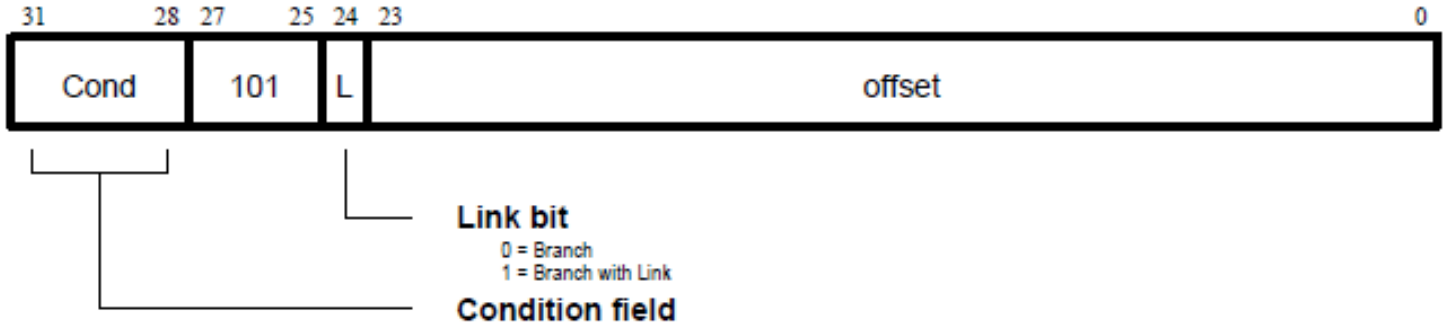
Syllabus

Unit 1: Basic Processor Architecture and Design

- ~~Microprocessor Overview~~
- ~~CISC VS RISC~~
- ~~Introduction to ARM Processor & Applications~~
- ~~ARM Architecture Overview~~
- ~~Different ARM processor Modes~~
- ~~Register Bank~~
- ~~ARM Program structure~~
- ~~ARM Instruction Format~~
- ~~ARM INSTRUCTION SET~~
- **Instruction Encoding**
  - ~~✓ Instruction Layout~~
  - ~~✓ Data Processing Instruction~~
  - ✓ **Branch Instruction**
  - ✓ **Data Transfer Instruction**
  - ✓ **Multiplication Instruction**


In goes the code.
Crank the handle...
HELLOWORLD.C
int main() {
printf("Yo world!");
return 0;
}
THE COMPILER
HELLOWORLD.EXE
You showed me
ADD R0,R0,R1
But what is this
in Binaries

BRANCH INSTRUCTIONS

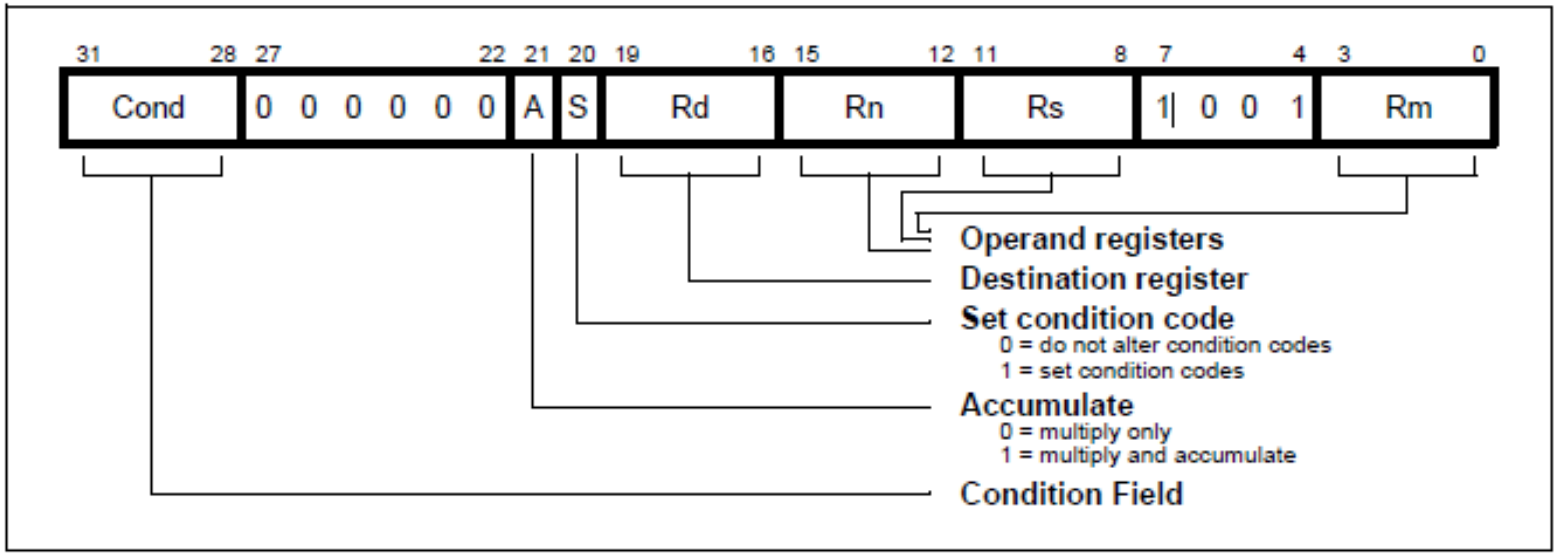


Ex 1: B LOOP

Ex 2: BL SUBROUTINE

Microprocessor & Computer Architecture (µpCA)

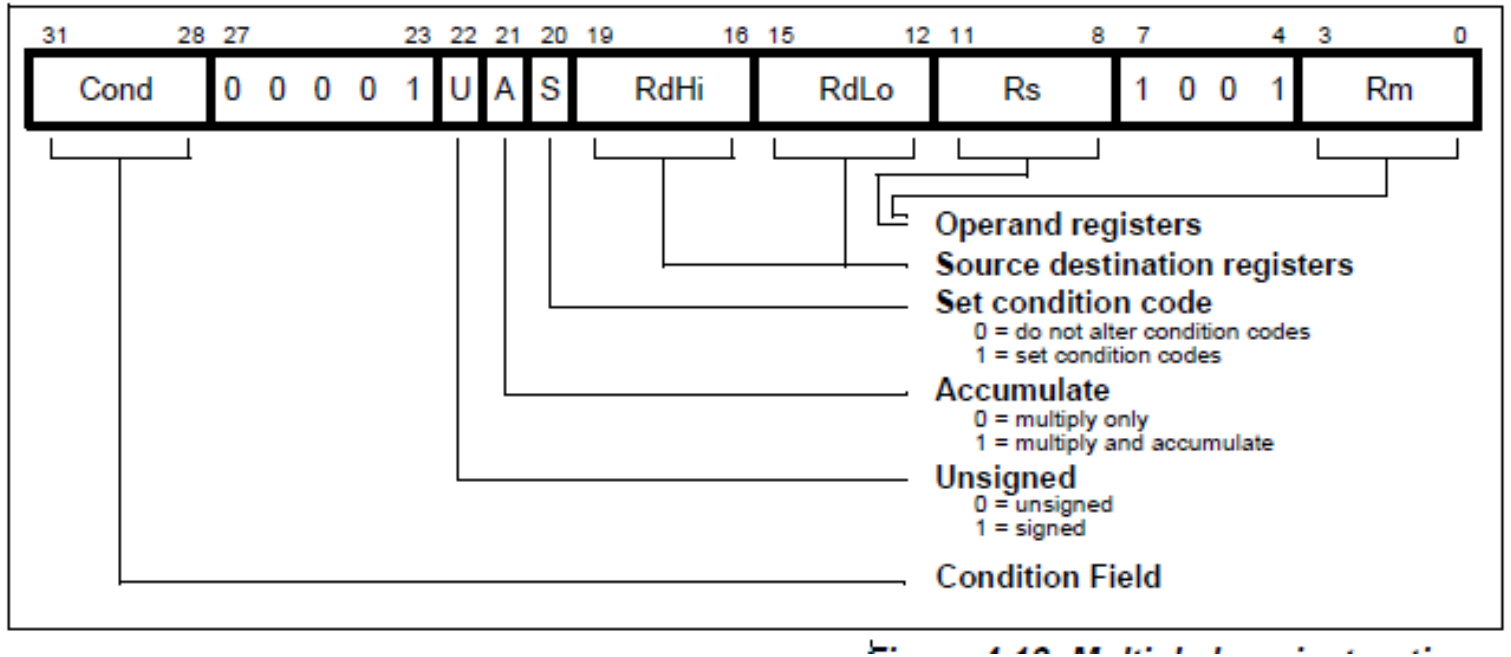
MULTIPLICATION INSTRUCTIONS



Ex 1: MUL R0, R1, R2

Ex 2: MLA R0, R1, R2 , R3

MULTIPLICATION INSTRUCTIONS

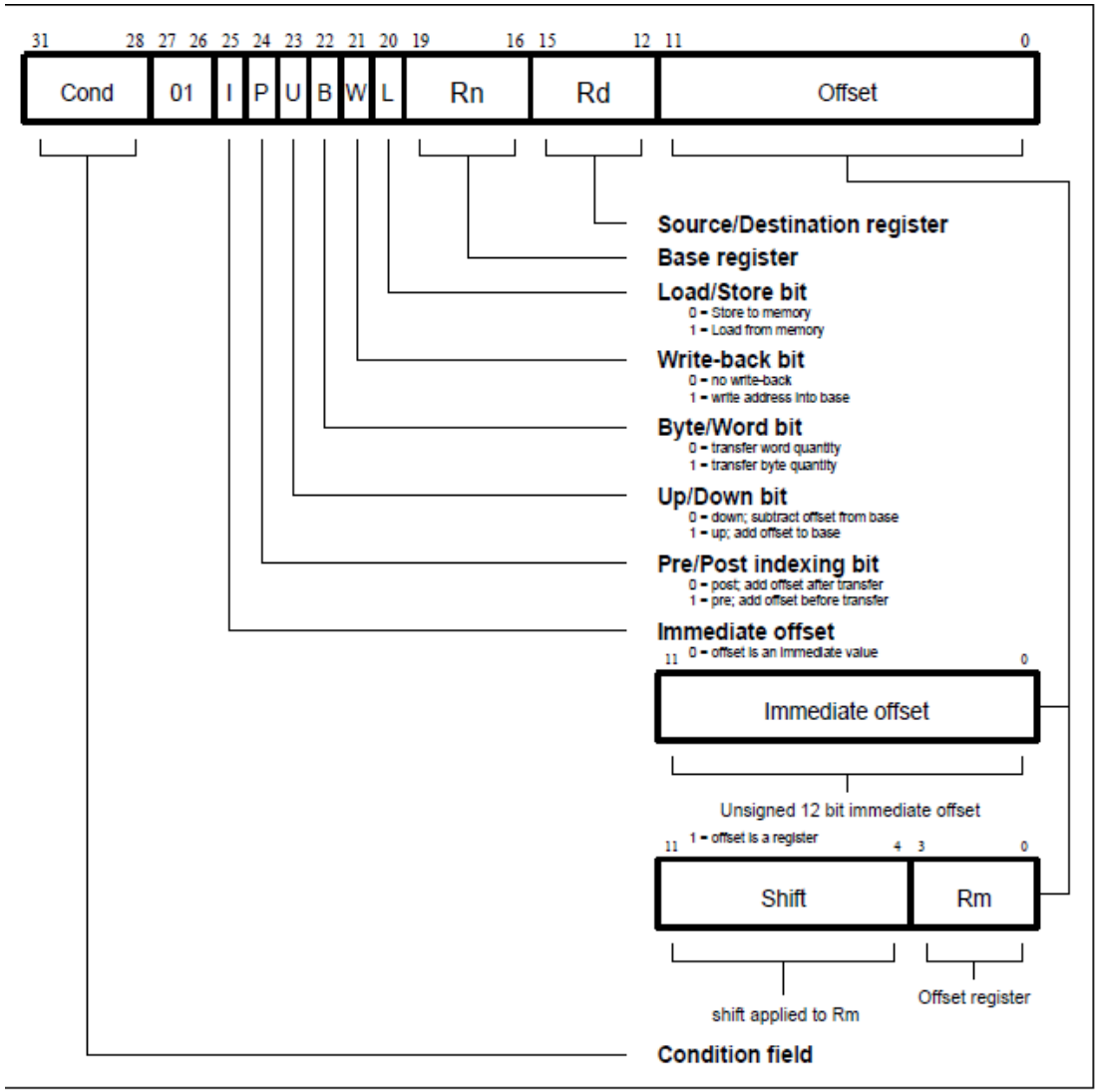


Ex 3: SMULL R0, R1, R2 , R3

Ex 4: UMLAL R0, R1, R2 , R3

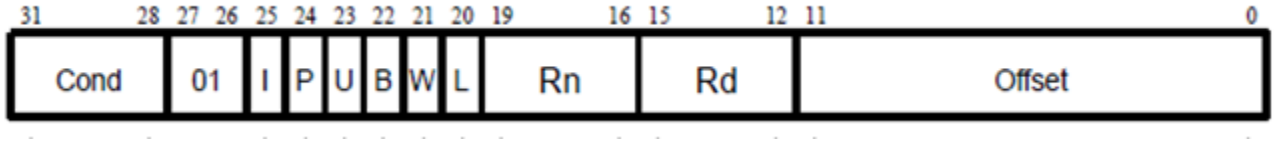
Microprocessor & Computer Architecture (µpCA)

Data Transfer Instruction



Data Transfer Instruction: STR

■Ex:



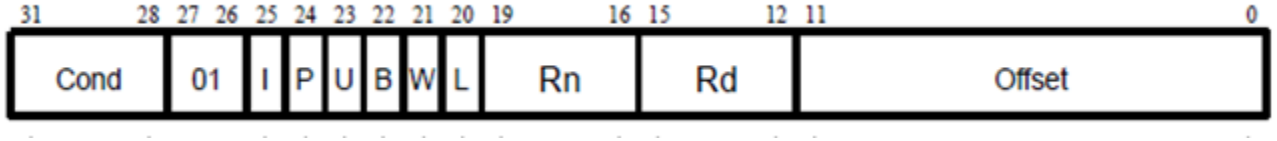
STR R0, [R1]

1110 01 0 0 1 0 1 0 0001 0000 000000000000

1110 01 I P U B W L 0001 0000 000000000000

E4B10000

Data Transfer Instruction: LDR



■ Ex:

LDR R0, [R1], R2

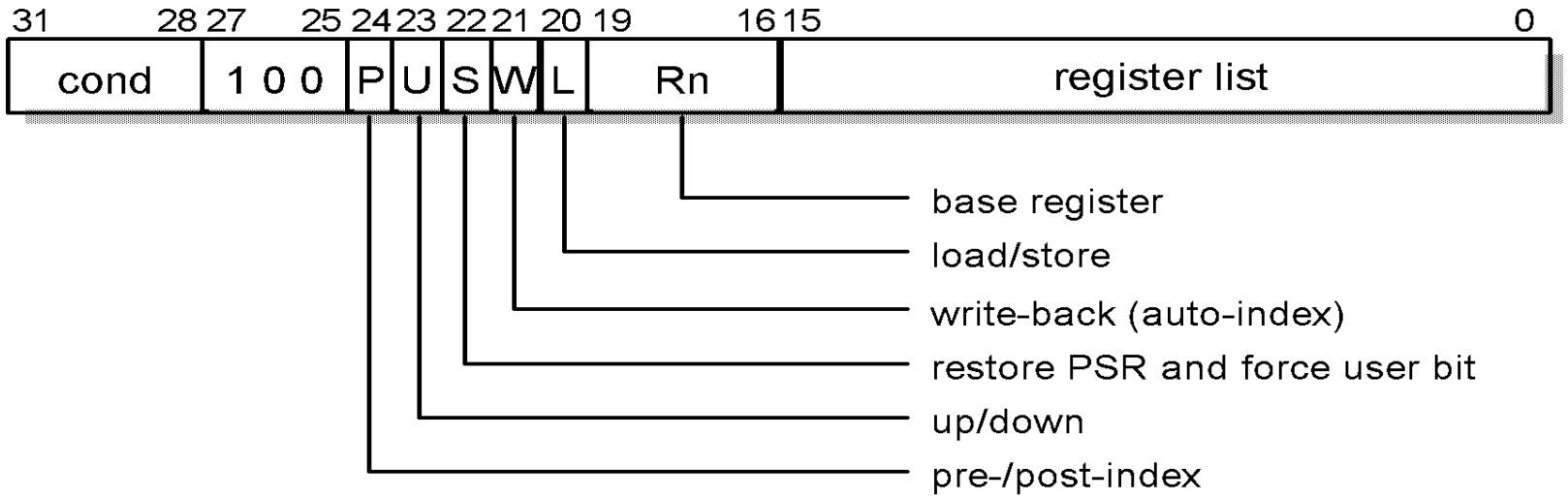
1110 01 1 0 1 0 1 1 0001 0000 00000000 0010

1110 01 I P U B W L 0001 0000 00000000 0010

1110 0110 1011 0001 0000 0000 0000 0010

E6B10002

Block Transfer Instruction



Ex 1: LDMIA R13! , { R0, R5 - R8, R11}

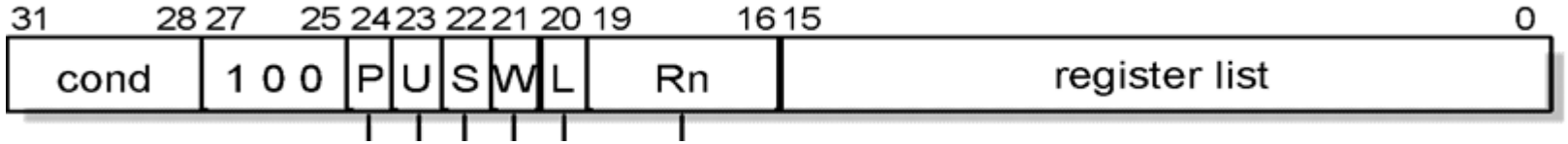
Ex 2: STMIA R13! , { R8, R4- R6, R12}

Block Transfer Instruction: Addressing Mode

Name	Stack	Other	L bit	P bit	U bit
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

Microprocessor & Computer Architecture (µpCA)

Block Transfer Instruction: LDM



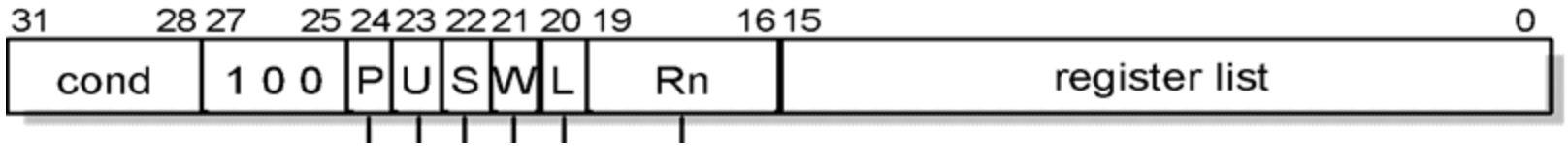
- **LDMIA R13! , { R0, R5 - R8, R11}**

1110 100 **0** 1 **0** **1** **1** 1101 0000 **1** 00**1** **1** **1** **1** 0000**1**

1110 100 **P** **U** **S** **W** **L** 1101 **0000** **R11** **00R8R7R6R5** **0000R0**

E8BD09E1

Block Transfer Instruction: STM



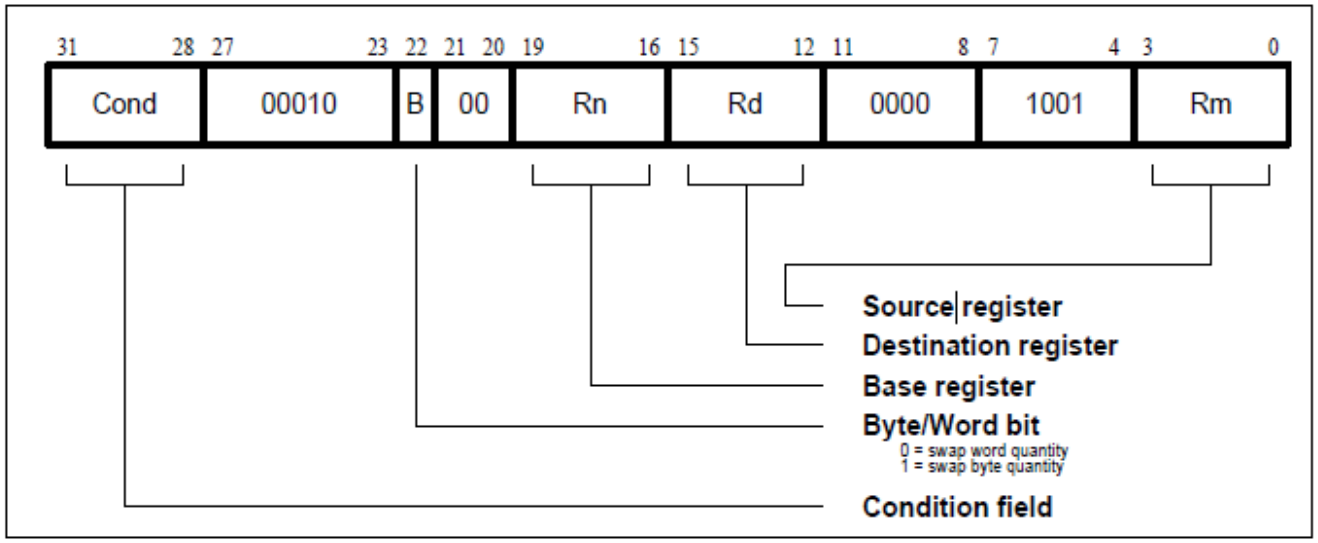
- **STMIB R13! , { R8, R4- R6, R12}**

1110 100 **1 0 0 1 0** 1101 00010 00101110000

1110 100 **P U S W L** 1101 000**R12**0 00**R8** 0**R6R5R4** 0000

E92D1170

SWAP


31 28 27 23 22 21 20 19 16 15 12 11 8 7 4 3 0
Cond
00010
B
00
Rn
Rd
0000
1001
Rm
Source register
Destination register
Base register
Byte/Word bit
0 = swap word quantity
1 = swap byte quantity
Condition field

Next Session

Unit 2
Pipeline Processor

THANK YOU

Dr. D. C. Kiran
Department of Computer Science and Engineering

dckiran@pes.edu

9829935135