Microprocessor & Computer Architecture (µpCA)

UE19CS252

Dr. D. C. Kiran

Department of
Computer Science and Engineering

PES UNIVERSITY ONLINE

Microprocessor & Computer Architecture (µpCA)

Data Transfer Instructions

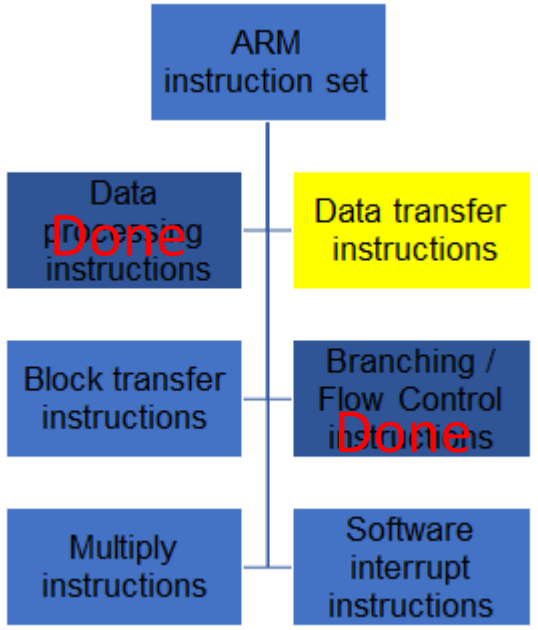
Dr. D. C. Kiran
Department of Computer Science and Engineering

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**
  - ~~Data Processing Instructions~~
  - ~~Flow Control Instructions~~
  - **Data Transfer Instructions**
    - ~~**LDR & STR**~~
    - **Indexing**



- Pre Indexing Without Write Back
- **Syntax:** LDR Rd, [Rn,OFFSET]
- Pre Indexing With Write Back

  Syntax: LDR Rd, [Rn,OFFSET]!
- Poste Indexing

  Syntax: LDR Rd, [Rn] ,OFFSET

- Memory is addressed by a register and an offset.

```
LDR   R0, [R1]  @ mem[R1]
```

- Three ways to specify offsets:
  - **Immediate**

```
LDR   R0,[R1,#4]      @ mem[R1+4]
```

  - **Register**

```
LDR   R0,[R1,R2]      @ mem[R1+R2]
```

  - **Scaled Register**

```
LDR   R0,[R1,R2,LSL #2] @ mem[R1+4*R2]
```

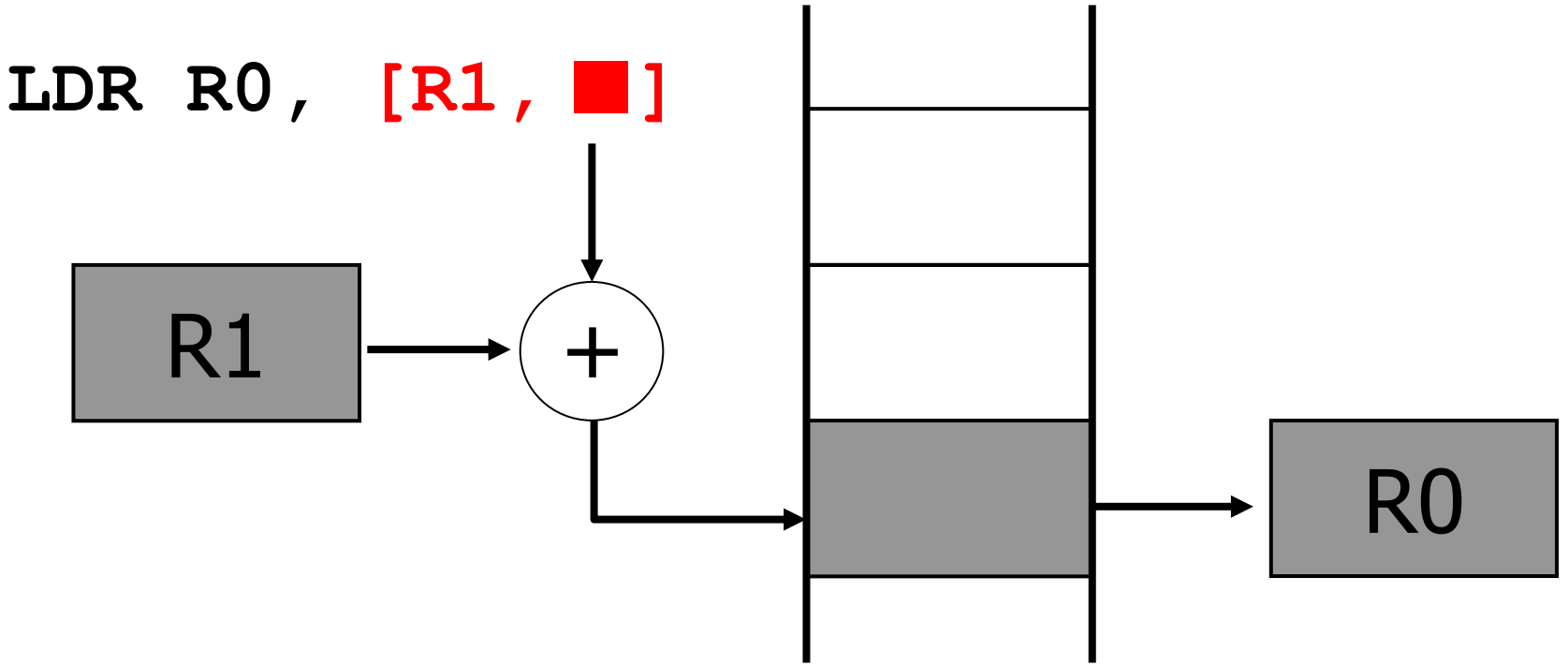
- Pre-index addressing (`LDR R0, [R1, #4]`)
  without a writeback.

- Auto-indexing addressing (`LDR R0, [R1, #4]!`)
  Pre-index with writeback
  calculation before accessing with a writeback

- Post-index addressing (`LDR R0, [R1], #4`)
  calculation after accessing with a writeback

Pre-index addressing

```
LDR   R0, [R1, #4]     @ R0=mem[R1+4]
                       @ R1 unchanged
```

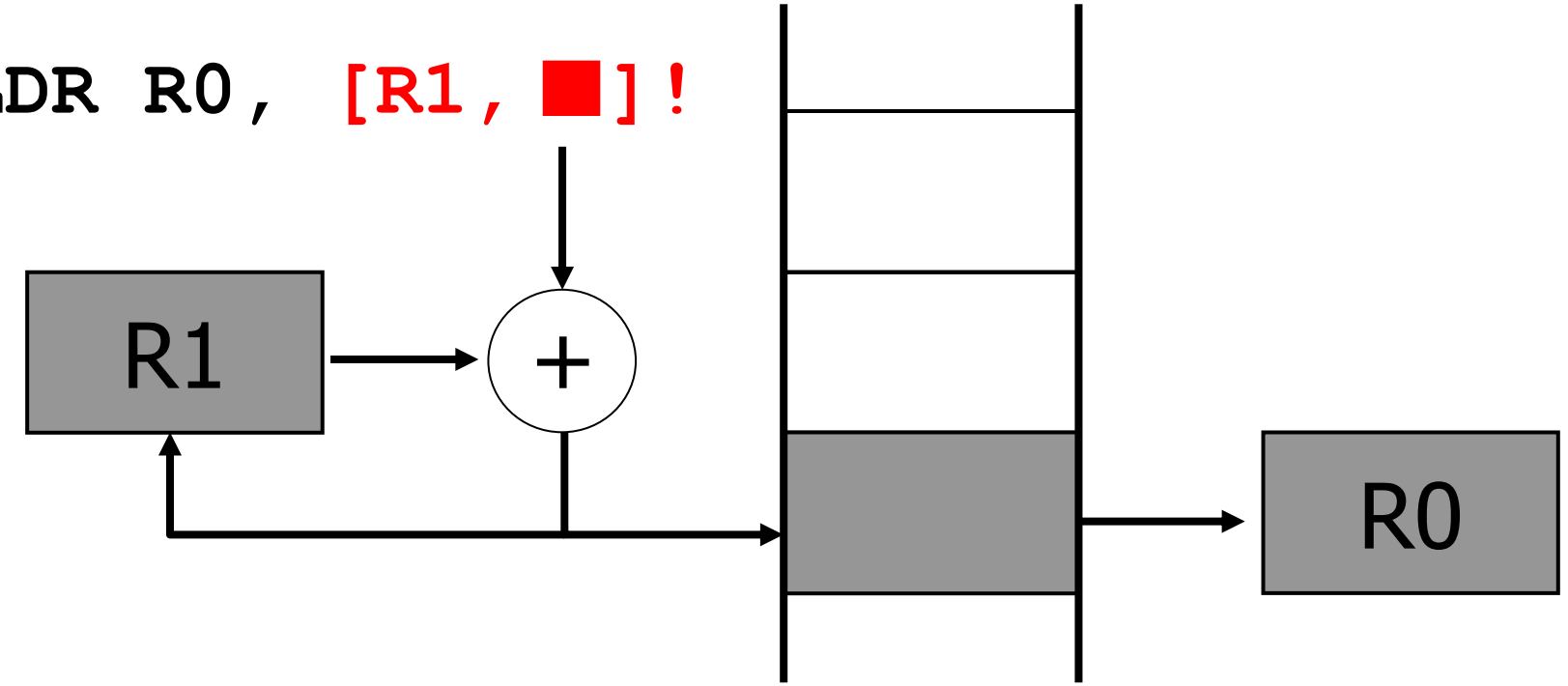


Auto-indexing addressing

```
LDR   R0, [R1, #4]!   @ R0=mem[R1+4]
                      @ R1=R1+4
```

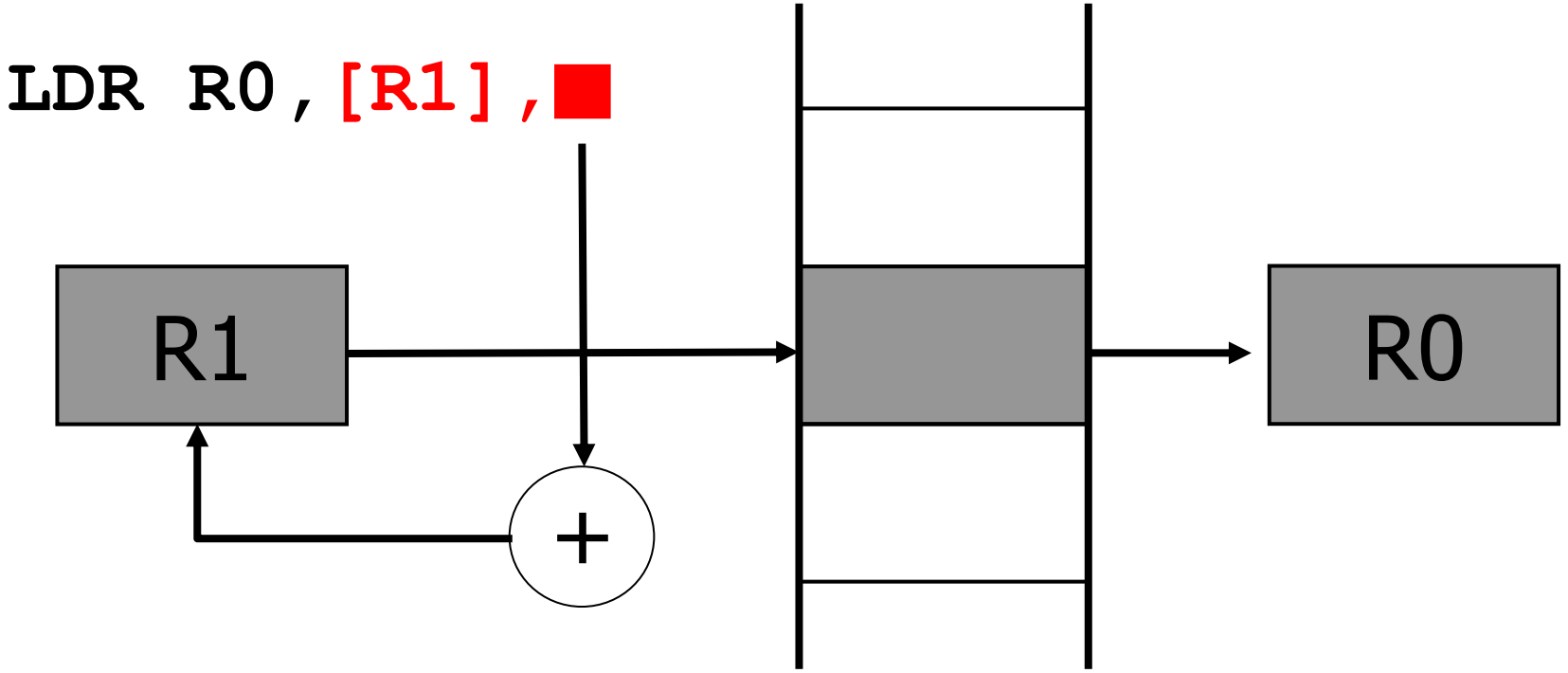
No extra time; Fast;

LDR R0, [R1, ■]!



Post-index addressing

```
LDR   R0, [R1], #4        @ R0=mem[R1]
                          @ R1=R1+4
```



Microprocessor & Computer Architecture (µpCA)

Pre-indexed addressing without Write Back or Auto Indexing

```
                        .text
00001000:E59F1000       LDR R1,=A
00001004:E5910004       LDR R0,[R1,#4]
                        .data
0000100C:               A: .word 10,20,30,40
```

R0 get the 0x100C+4th Value

R1 is not Updated

RegistersView

General Purpose | Floating Point

Hexadecimal | Unsigned Decimal | Signed Decimal

```
R0       :00000014
R1       :0000100c
R2       :00000000
R3       :00000000
R4       :00000000
R5       :00000000
R6       :00000000
R7       :00000000
R8       :00000000
R9       :00000000
R10 (sl) :00000000
R11 (fp) :00000000
R12 (ip) :00000000
R13 (sp) :00005400
R14 (lr) :00000000
R15 (pc) :00001008
```

MemoryView1

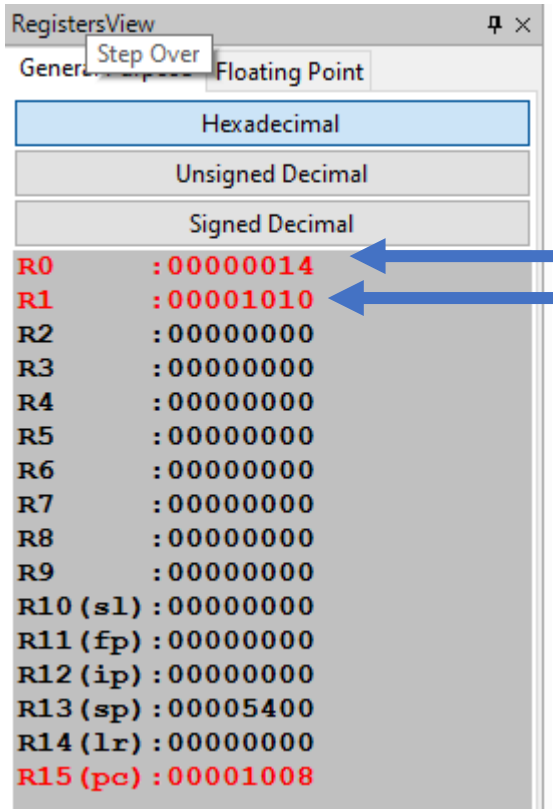
0000100C

```
0000100C  0000000A  00000014  0000001E  00000028
00001030  81818181  81818181  81818181  81818181
00001054  81818181  81818181  81818181  81818181
```

Microprocessor & Computer Architecture (µpCA)

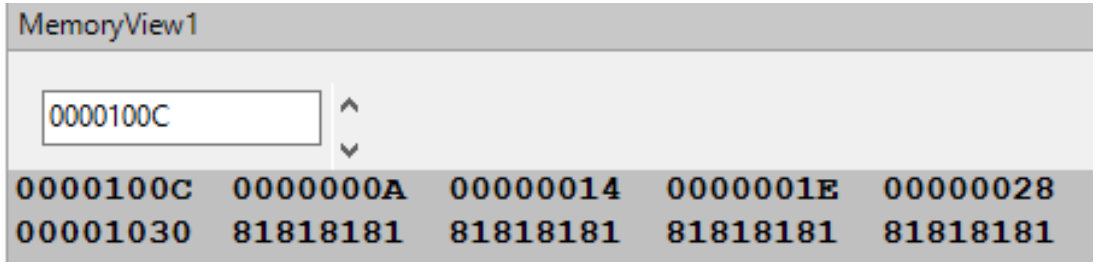
Pre-indexed addressing with Write Back or Auto Indexing-2

```
                        .text
00001000:E59F1000       LDR R1,=A
00001004:E5B10004       LDR R0,[R1,#4]!
                        .data
0000100C:               A: .word 10,20,30,40
```



R0 get the 0x100C+4th i.e 0x1010th Value

R1 is Updated to 0x1010



Microprocessor & Computer Architecture (µpCA)

Post-index addressing:2

```
                        .text
00001000:E59F1000       LDR R1,=A
00001004:E4910004       LDR R0,[R1],#4
                        .data
0000100C:               A: .word 10,20,30,40
```

RegistersView

General Purpose | Floating Point

Hexadecimal | Unsigned Decimal | Signed Decimal

```
R0      :0000000a
R1      :00001010
R2      :00000000
R3      :00000000
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
R10(sl) :00000000
R11(fp) :00000000
R12(ip) :00000000
R13(sp) :00005400
R14(lr) :00000000
R15(pc) :00001008
```

R0 get the 0x100Cth Value

R1 is Updated to 0x1010

MemoryView1

0000100C

```
0000100C  0000000A  00000014  0000001E  00000028
00001030  81818181  81818181  81818181  81818181
00001054  81818181  81818181  81818181  81818181
```

Next Session:

Block Transfer

- **LDRM**
- **STRM**

THANK YOU

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

dckiran@pes.edu

9829935135