Vezba 1

UVOD

**ARM Load/Store Instrukcije**

* ARM je Load/Store arhitektura:
  + Samo load i store instrukcije mogu pristupati memoriji
  + Nema podršku za memory <-> memory obradu podataka
  + Mora preneti podatke u registre pre korišćenja istih
* ARM ima tri seta instrukcija koje mogu da interaguju sa glavnom memorijom i to su:

– Single register data transfer (LDR/STR)

– Block data transfer (LDM/STM)

– Single Data Swap (SWP)

• Osnovne load/store instrukcije:

|  |  |  |
| --- | --- | --- |
| LDR | STR | Word |
| LDRB | STRB | Byte |
| LDRH | STRH | Halfword |
| LDRSB |  | Signed byte load |
| LDRSH |  | Signed halfword load |

Da prenesite 4bita podataka treba da specificiramo dve stvari:

–Registar: r0 - r15

–Memorisku adresu:

• Razmisljajte o memoriji kao jednom jednodimenzionalnom nizu, tako da mozemo mu pristupiti davanjem pointera na memorisku adresu.

• Postoje trenuci kad zelimo da koristimo adresu sa pomeranjem

Postoje dva tipa moda adresiranja koje se koriste u ARM-u

• Pre-indexed addressing: Adresa generisana se koristi momentalno

• Post-indexed addressing: Generisana adresa se koristi kasnije se upisuje u base registar

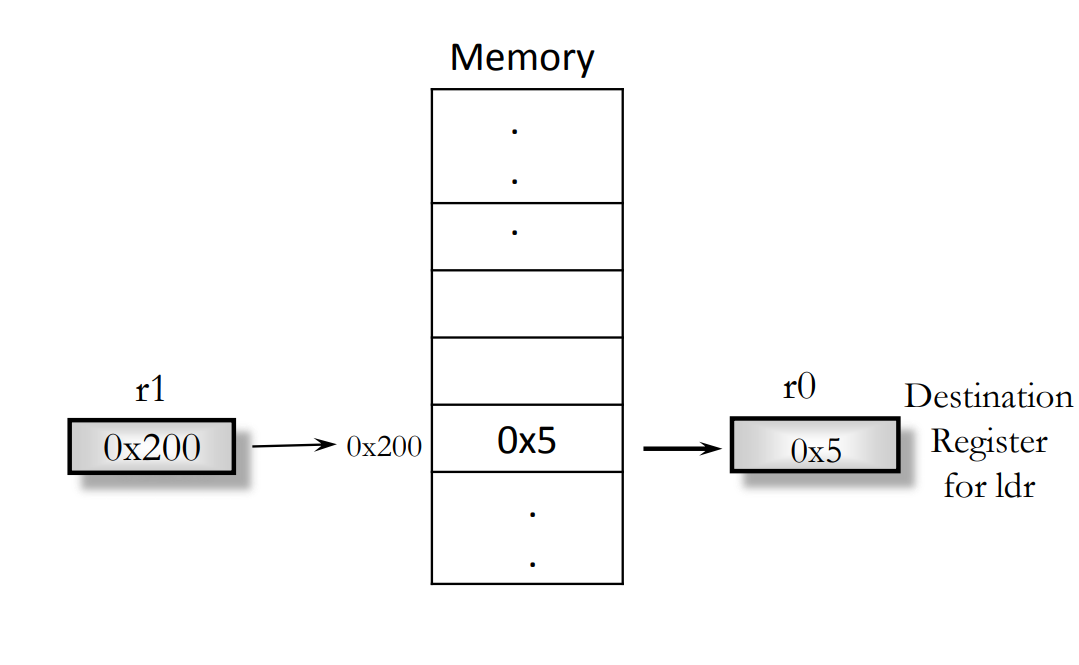
**[Rn]**

Register

Address accessed is value found in Rn.

Primer:

LDR r0, [r1] @ r0 \*r1



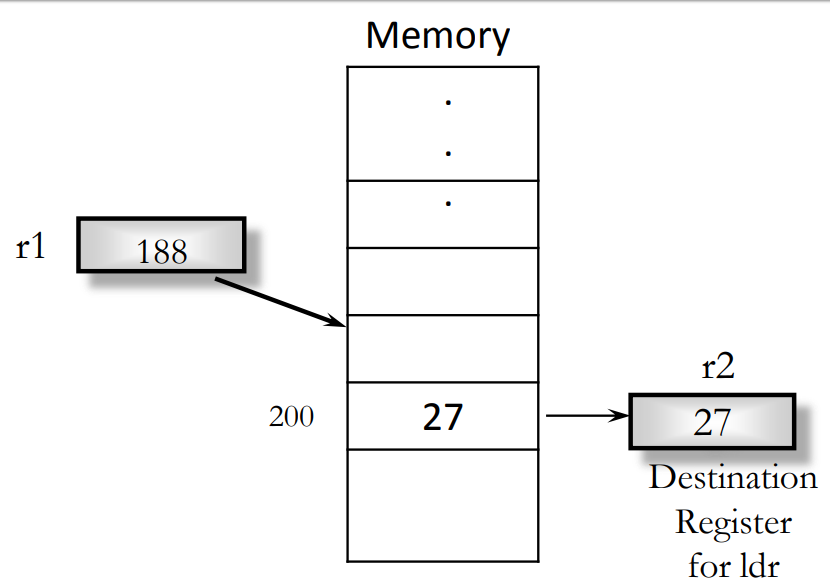
**[Rn, #±imm]**

Immediate offset

Adresa kojoj se pristupa je imm vise/manje nego adresa koja se nalazi u Rn. Rn se **ne** **menja**.

Primer:

LDR r2, [r1, #12] @ r2 ← \*(r1 + 12)



**[Rn, ±Rm]**

Register offset

Adresa kojoj se pristupa ima vrednost Rn ± vradnost u Rm. Rn i Rm **ne menjaju** vrednosti.

Primer:

LDR r2, [r0, r1] @ r2 ← \*(r0 + r1)

**[Rn, ±Rm, shift]**

Scaled register offset

Adresa kojoj se pristupa je vrednost u Rn ± Rm pomeren u zavisnosti od shift. Rn i Rm **ne menjaju** vrednosti.

Primer:

LDR r0, [r1, r2, lsl #2] @ r0 ← \*(r1 + r2\*4)

**[Rn, #±imm]!**

Immediate pre-indexed w\update

Adresa kojoj se pristupa sa immediate offset modom, ali vrednost u registru Rn postaje adresa kojoj se pristupa

Primer:

LDR r2, [r1, #12]! @ r1 ← r1 + 12 onda r2 ← \*r1

**[Rn, ±Rm]!**

Register pre-indexed w\update

Adresa kojoj se pristupa je sa register offset modom, ali Rn vrednost postaje adresa kojoj se pristupa.

Primer:

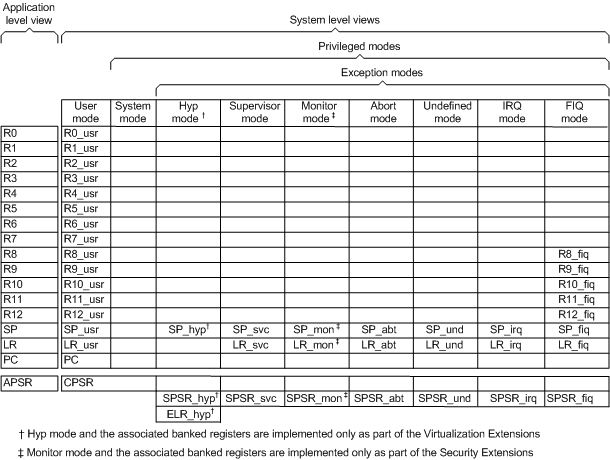
LDR r2, [r0, r1]! @ r0 ← r0 + r1 onda r2 ← \*r0

**ARM registri**

ARM procesori imaju registre opšte namene i registre specialne namene. Dodatni registri se mogu koristiti u privilegovanom modu.

U svim ARM procesorima, sledeci registri su dostupni i može im se pristupiti u svim procesorskim modovima:

* 13 registara opšte namene R0-R12.
* Jedan *Stack Pointer* (SP).
* Jedan *Link Registar* (LR).
* Jedan *Program Counter* (PC).
* Jedan *Application Program Status Register* (APSR).



--------------------------------------------Deo materijala je vucen sa ovih adresa------------------------------------------

https://people.cs.clemson.edu/~rlowe/cs2310/notes/ln\_arm\_load\_store\_plus\_multiple\_transfers.pdf