What can you do with ADD, AND, NOT:

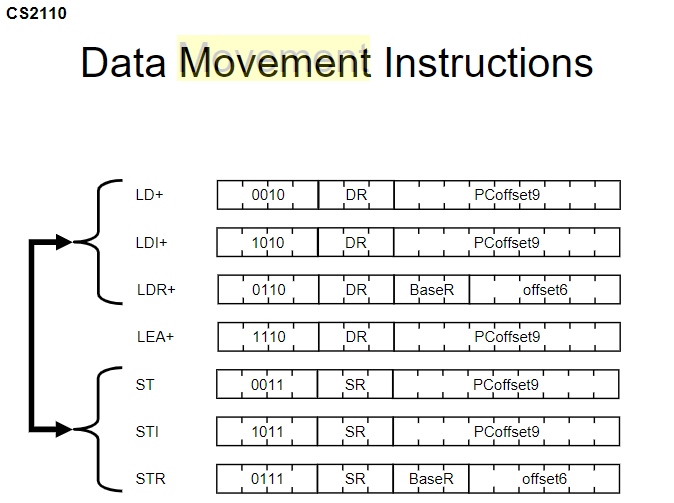
* Add
* And
* Not
* Subtract
* Or

|  |  |  |  |
| --- | --- | --- | --- |
| NOT | R2 | R3 |  |
| NOT | R3 | R3 |  |
| AND | R1 | R2 | R3 |
| NOT | R1 | R1 |  |

* Clear a register
* Copy from one register to another
* Increment a register by n, -16 < n < 15

Data Movement Instructions

* Should be read from the standpoint of the registers
  + LOAD something into this register
  + STORE this register into memory
* STORE instructions are read BACKWARDS (Left to right)



Loads

* LEA: Puts the address represented by some label into memory
* LD: Puts the contents of some label (address) into a register
* LDI: Puts the contents of memory location whose address is stored in a memory at some label into a register
* LDR: Puts the contents of the effective address into a register. The effective address is computed by adding the base register to the offset which is the offset6 immediate operand

Stores

* ST: Puts the contents of a register into an address specified by a label (address)
* STI: Puts the contents of a register into a memory location whose address is stored in memory at some label
* STR: Puts the contents of the source register into memory at some effective address. The effective is calculated by adding the base register to the offset6 immediate operand. 