The LD Instruction

The High Level
The **Load Direct** instruction is a type of **Data Movement Instruction** that takes a value directly from a memory location denoted by a **label** and places it into a register.

To do so, it uses a **PC Offset**, which is a count of how many memory locations ahead of the **Program Counter** (aka: the PC) the desired load value is. In order words, the PCOffset9 is like saying, "how many spaces ahead of the PC do I have to go to get to the thing I want to load?"

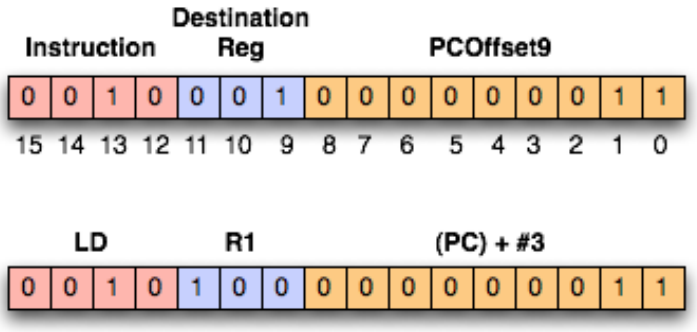

Figure 1: The LD Instruction (details)

The Breakdown:

- Bits [15-12] specify what instruction to execute (0010 is LD)
- Bits [11-9] specify which register to store the loaded value in
- Bits [8-0] specify a 9-bit PCOffset, which is the number of memory locations (i.e. "spaces") between the PC and the value you want to load

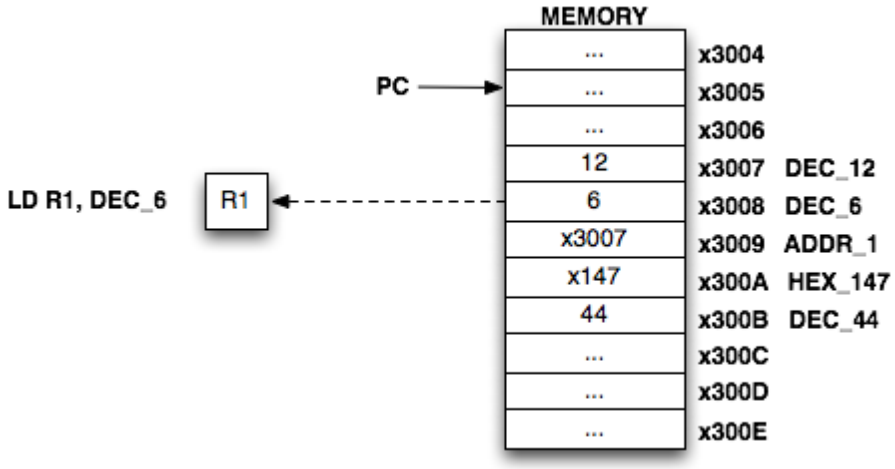

Figure 2: The LD Instruction – Visual Execution

The Examples!

```
.orig x3000            ; program begins here
   ;---------------
   ; Instructions
   ;---------------
   LD R5, DEC_123      ; Put the value #123 into R5
   LD R2, ADDR_1       ; Put the value x4000 into R2
   LD R1, ASCII_0      ; Put #48 (ASCII for the character '0') into R1
   HALT                ; Stop execution of program

   ;---------
   ; Data
   ;---------
   DEC_123    .FILL     #123
   ADDR_1     .FILL     x4000
   ASCII_0    .FILL     #48
.end                   ; tell assembler not to look for any more code
```

Pitfalls... (aka: Erroneous code makes baby bunnies cry)
The example below is erroneous. Please do NOT try to code this way!

```
   LD R1, x4000  ; (ERROR: You must use a label,not a literal memory address)
   LD R2, LabelThatIsReallyFarAway ; (ERROR: Overflows 9-bit PCOffset9 field)
```

The first example pitfall code above is incorrect because you have to use a **label** whenever you use the LD instruction. You cannot give the instruction an address. It's just not built that way.

The second example pitfall code above is incorrect and classically causes problems for **many** students. Since this instruction uses a 9-bit **PC Offset**, which means the value you want to load must be no more than [Range of 9 bits] memory locations away from the PC (Note: The range of a Two's Compliment 9-bit field is ±[Range of 8 bits] == [-256, 255].

Further Illustrated erroneous example:
If you have a program that begins at x3000, is 500 lines of code long, has as its first instruction a LD instruction, and has the **Data Area** at the end of the **Instruction Area** (as it should), then when the program tries to issue the LD instruction, it will find that it can't "reach" the desired value to load because it is "too far away" (i.e. cannot be reached with Two's Compliment 9-bits of range)

```
;-------------------------------------------------------------
; ERRONEOUS CODE EXAMPLE:
; Example of a very common error that students run into.
; This program will result in an "Overflows 9-bit PCOffset" error
;-------------------------------------------------------------
.orig x3000
   ;---------------
   ; Instructions
   ;---------------
   LD R1, DEC_17

   ...
   ... [256 or more lines of code, thus pushing the Data Area
   ...  beyond the reach of the PCOffset9 that the LD instruction
   ...  needs to use]

   HALT
   ;-------
   ; Data
   ;-------
   DEC_17     .FILL     #17
.end
```