The LDR Instruction

The High Level

The **Load Relative** instruction is a type of **Data Movement Instruction** that, given a **Base Register** and an **Offset**, loads MEM[BaseRegister + Offset] into a register.

To understand the content of this tutorial, you should know what a **PCOffset9** is, as well as how the **LD** instruction works.

(You may note that the LDR instruction is equal to the composite use of LEA and LDI instructions.)

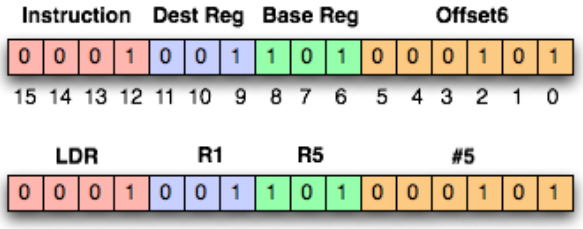

Figure 1: The LDR Instruction (details)

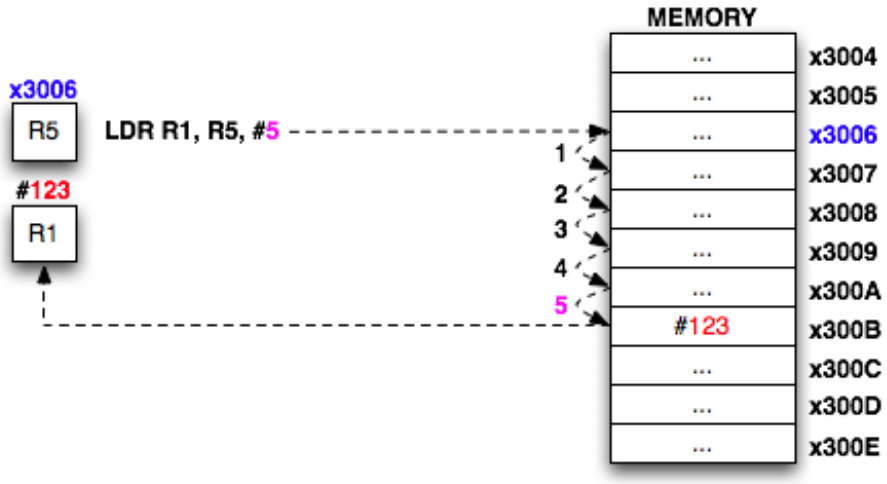

Figure 2: The LDR Instruction – Visual Execution

The Breakdown:

- Bits [15-12] specify what instruction to execute (0110 is LDI)
- Bits [11-9] specify which register to store the value in
- Bits [8-6] specify a Base Register
- Bits [5-0] specify a Two's Compliment 6-bit offset.

The Examples!

```
.orig x3000
   ;---------------
   ; Instructions
   ;---------------
   LD R5, ADDR_1       ; R5 <-- x4000
   LDR R1, R5, #0      ; R1 <-- MEM[R5 + #0] == MEM[R5] == MEM[x4000] == #55
   HALT

   ;---------
   ; Data
   ;---------
   ADDR_1  .FILL  x4000

; Hard-code the value #55 at memory location x4000
.orig x4000
   .FILL  #55

.end
```

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

```
LDR #5, R1      ; (ERROR: Order must be: LDR [Base Reg], [Offset])
LDR R1, x4000   ; (ERROR: You must use a label,not a literal memory address)
LDR R1, #32     ; (ERROR: Overflows Two's Compliment 6-bit field)
```

The first example pitfall code above isi incorrect because the order of operands should have been: LDR R1, #5

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

The third example pitfall code above is incorrect because it overflows a Two's Compliment 6-bit field. Since 6 bits can represent only numbers in the range $[-32, 31]$, the number #32 is too big to be expressed with 6 bits. Hence, the error.