Sources and Destinations

- The generic form of an instruction is `<op> <src-opnds>, <dst>`.

- Where can source operands (rvalues) come from?
  - A constant value.
  - The contents of a GPR.
  - The contents of one or more memory cells.

- Where can results be placed (lvalues)?
  - A GPR.
  - One or more memory cells.

Immediate Operands

- This specification form is used for constants that are embedded directly within the instruction.
  - E.g., `$-577`, `$0x1F`.
  - Allowable range depends on the instruction.

- The concrete syntax is `$`X. It evaluates to X.

Register Operands

- This specification form is used for values that are the contents of a GPR.
    - E.g., `%rbx`, `%ecx`.
    - The width of the result (8, 16, 32, or 64 bits) is implicit in the register name.

- The concrete syntax is `r`. It evaluates to R[`r`].

Memory Operands

- This specification form is used for values that are the contents of one or more memory locations.
  - How wide? Encoded by the ASM suffix of the opcode. Call this width b (Bytes).
  - Which memory location? Too many choices, and may want to access different memory locations from the same piece of code (e.g., `A[i]`).
- Specified by a formula called the addressing mode (AM), which is encoded in the instruction.
- This formula is evaluated as part of instruction execution to provide a value called the effective address (EA). Call this value a.
- The EA is used to access memory bytes $M[a]$ through $M[a+b-1]$ (or $M_b[a]$ for brevity).

Memory Addressing Modes and EAs

Name	AM (Concrete syntax)	EA (Evaluates to)	Comments
Absolute	X	$M[X]$	Note difference from Immediate.
Indirect	(r_a)	$M\left[R[r_a]\right]$	r_b must be a 64-bit register.
Base + Displacement	$X(r_b)$	$M\left[X + R[r_b]\right]$	r_b must be a 64-bit register.
Indexed	$X(r_b, r_i)$	$M\left[X + R[r_b] + R[r_i]\right]$	r_b and r_i must be 64-bit registers.
Scaled Indexed	$X(r_b, r_i, s)$	$M[X + R[r_b] + R[r_i] \cdot s]$	r_b and r_i must be 64-bit registers. $s \in \{1, 2, 4, 8\}$.

Special Forms (Syntactic Sugar)

Variant of	AM	EA	Comments
Indexed	(r_b, r_i)	$M\big[R[r_b] + R[r_i]\big]$	*No displacement.* r_b and r_i must be 64-bit registers.
Scaled Indexed	(r_b, r_i, s)	$M[R[r_b] + R[r_i] \cdot s]$	*No displacement.* r_b and r_i must be 64-bit registers. $s \in \{1, 2, 4, 8\}$.
Scaled Indexed	$X(, r_i, s)$	$M[X + R[r_i] \cdot s]$	*No base register.* r_i must be a 64-bit register. $s \in \{1, 2, 4, 8\}$.
Scaled Indexed	$(, r_i, s)$	$M[R[r_i] \cdot s]$	*No displacement or base register.* r_i must be a 64-bit register. $s \in \{1, 2, 4, 8\}$.