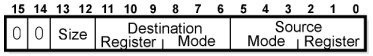
Immediate Value to Data Register

- In previous examples we have referred to either a **memory location** or a **data register**.
  - -> A location where the data is stored.
- Sometimes we need to specify a given value known as an **immediate value**.
  - -> Use immediate addressing mode.

Example: Move Value 10 into D2

-> Source mode = immediate mode = 111
-> Source register = immediate mode = 100



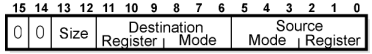
0 0 11 010 000 111 100 -> $343C

, PPHGLDWH 9D0XHV 6\PER0

- We distinguish immediate values from addresses using **#** sysmbol:
  - **#**1000 = Decimal value 1000
  - **#**$1000 = Hexadecimal value 1000
  - 1000 = Address 1000 (decimal)
  - $1000 = Address 1000 (hexadecimal)

0QHPRQLFV

Template for a move instruction.



- Diffecult to remember template for each instruction.
- We'll replace this with the sysmbol (mnemonic):
  - move
- Replace the size feld with the b, w and l subscripts
  - move.b -> move a byte
  - move.w -> move a word
  - move.l -> move a long word

Encode Operands and the Destination/Source Modes

- The operands and the destination /source modes are encoded as arguments following the symbol(**src**, **dest**):
  - **move**.b d0,d1
  - **move**.w $2000,d5
  - **move**.l $2000, $200A
  - **move**.l #10, d2
- Assembler sysntax for the move command is:
  - move <ea> <ea>
- <ea> effective address = an expression which, when evaluated, produces a value.

0RYH

MOVE **Move Data from Source to Destination** **MOVE**

Operation: Source → Destination

Assembler Syntax: MOVE ⟨ea⟩, ⟨ea⟩

Attributes: Size = (Byte, Word, Long)

Description: Moves the data at the source to the destination location, and sets the condition codes according to the data.

Condition Codes:

X	N	Z	V	C
—	*	*	0	0

X Not affected.
N Set if the result is negative. Cleared otherwise.
Z Set if the result is zero. Cleared otherwise.
V Always cleared.
C Always cleared.

' HIDX0WV

- In order to simplify coding of assembly language, certain defaults are assumed:
  - move -> move.w (i.e.: word size is default)
  - move 10,d2 -> absolute addressing
  - 10 -> decimal 10

$GGUHVVLQJ 5HVWULFWLRQV

Only Bytes may be read from ODD memory addresses. Words and Long-words must be read from EVEN memory addresses!

- This was a design decision to simplify memory interfacing.
- Accessing words or long-words on odd boundaries causes the 68000 to terminate the F->D->E cycle.
- -> An address error occurs

Which of these instruction is legal?

a) move.w $4001, d0 ✗
b) move.b $4004, d1 ✓
c) move.b $4003, d3 ✓
d) move.w $4001, $4003 ✗
e) move.b $4000, $4001 ✓