Linguagem de máquina - Representando instruções em binário

- ⇨ Como traduzir da linguagem de montagem para a linguagem de maquina?

Cada registrador possui um conjunto de bits associado (valor armazenado):
Assumindo que cada registrador tem 3 bits (8 registradores) e que a instrução tem 32 bits.

- 0=000
- 1=001
- 2=010
- 3=011
- 4=100
- 5=101
- 6=110
- 7=111

```
add  regA  regB destreg (addiciona regA com reg B e coloca em destreg)
```

```
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9  8  7  6  5  4  3  2  1  0
-----------------------------------------------------------------------------------------------
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
-----------------------------------------------------------------------------------------------
```

31-25 = 0
24-22=código de operação
21-19=regA
18-16=regB
15-3=0
2-0=destreg

```
add 1 2 5
```

31-25 = 0
24-22=código de operação=000
21-19 = regA = 1 = 001
18-16 = regB = 2 = 010
15-3 = 0
2-0 = destreg = 5 = 101

```
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9  8  7  6  5  4  3  2  1  0
-----------------------------------------------------------------------------------------------
| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 1| 0| 1| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 1| 0| 1|
-----------------------------------------------------------------------------------------------
```

```
lw regA regB deslocamento
(guarda em regB o valor armazenado na posição deslocamento do vetor
cujo endereço inicial está em regA)
```

```
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9  8  7  6  5  4  3  2  1  0
-----------------------------------------------------------------------------------------------
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
-----------------------------------------------------------------------------------------------
```

31-25 = 0
24-22 = código de operação = 010
21-19 = regA
18-16 = regB
15-0 = deslocamento expresso em Complemento a 2
(16 bits: (-2^{16-1}) -32768 a +32767$(-2^{16-1}-1)$)

Ex: *Programa em C:*
A[300]=h+A[300]

Programa em linguagem de montagem:
Supondo h alocada no registrador 2 e endereço base de A no registrador 6:

```
lw $6 $5 300
add $5 $2 $5
sw $6 $5 300 (sw: store word: guarda na posição 300 do vetor com
                  endereço inicial em $6 o valor do registrador $5)

lw $6 $5 300
31-25 = 0
24-22=código de operação=010
21-19=regA=6=110
18-16=regB=5=101
15-0=0000000100101100
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9  8  7  6  5  4  3  2  1  0
-----------------------------------------------------------------------------------------------
| 0| 0| 0| 0| 0| 0| 0| 0| 1| 0| 1| 1| 0| 1| 0| 1| 0| 0| 0| 0| 0| 0| 0| 1| 0| 0| 1| 0| 1| 1| 0| 0|
-----------------------------------------------------------------------------------------------

add 5 2 5
31-25 = 0
24-22=código de operação=000
21-19=regA=5=101
18-16=regB=2=010
15-3=0
2-0=destreg=5=101
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9  8  7  6  5  4  3  2  1  0
-----------------------------------------------------------------------------------------------
| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 1| 0| 1| 0| 1| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 0| 1| 0| 1|
-----------------------------------------------------------------------------------------------

sw 6 5 300
31-25 = 0
24-22=código de operação=011
21-19=regA=6=110
18-16=regB=5=101
15-0=0000000100101100
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
-----------------------------------------------------------------------------------------------
| 0| 0| 0| 0| 0| 0| 0| 0| 1| 1| 1| 1| 0| 1| 0| 1| 0| 0| 0| 0| 0| 0| 0| 1| 0| 0| 1| 0| 1| 1| 0| 0|
-----------------------------------------------------------------------------------------------
```

Instruções de desvio

Exemplo:
Programa em C:

```
if (x!=y)
  f=g+x;
else
  f=g+y;
```

Instrução de linguagem de montagem:

```
beq regA regB L1
(Caso conteúdo de regA seja igual ao conteúdo de regB vai para
instrução no endereço L1 (pulando para a instrução em L1), caso
contrário executa a próxima instrução.)
```

Supondo x alocada no registrador 1, y em 2, f em 3 e g em 4, teremos o seguinte código:

```
      beq $1 $2 ELSE
      add $4 $1 $3
      beq $1 $1 EXIT (usado para não executar o ELSE)
ELSE: add $4 $2 $3
EXIT:
```

Formato das instruções:

```
beq regA regB label
 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9  8  7  6  5  4  3  2  1  0
-------------------------------------------------------------------------------------------------
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
-------------------------------------------------------------------------------------------------
31-25 = 0
24-22=código de operação=100
21-19=regA
18-16=regB
15-0= lable = PC(program counter)+1+deslocamento (complemento a 2)
```