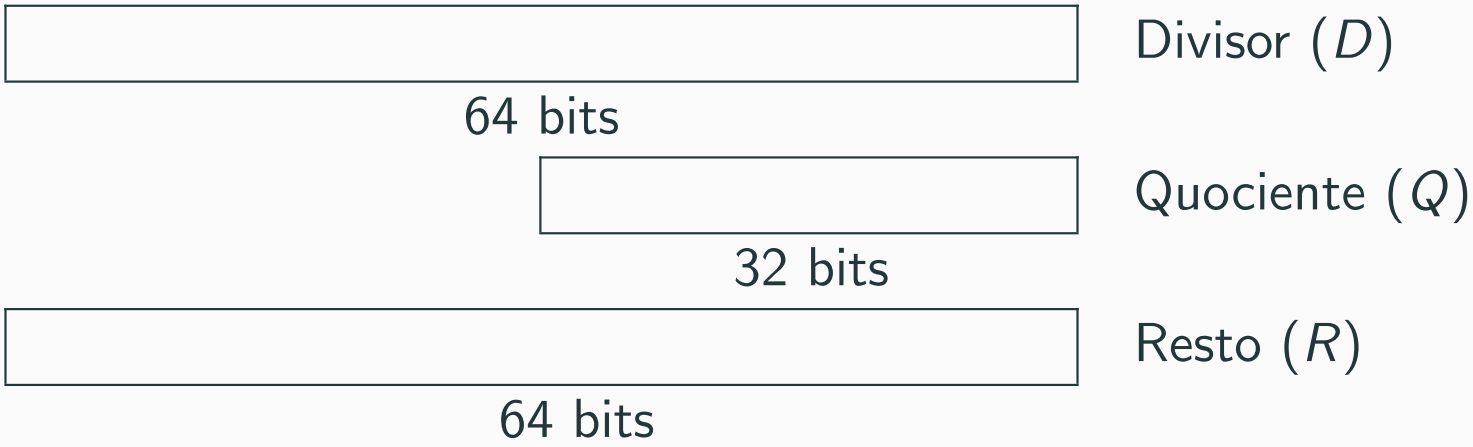
Divisão de inteiros

Fundamentos de Arquitetura de Computadores

Prof. John Lenon C. Gardenghi

Engenharia de Software
Faculdade do Gama
Universidade de Brasília

Divisão de inteiros sem sinal



1. Defina contador $= 1$ e
   - salve o Dividendo na parte *menos significativa* de R e
   - salve o Divisor na parte *mais significativa* de D.
2. $R = R - D$.
3. Faça um deslocamento à esquerda de 1 bit em Q.
   3.1 Se $R \geq 0$, defina o bit menos significativo de Q como sendo 1.
   3.2 Se $R < 0$, restaure o valor original de R ($R = R + D$).
4. Faça um deslocamento à direita de 1 bit no D.
5. Se contador < 33, faça contador $=$ contador$+1$ e volte ao Passo 2.

Exemplo: $\dfrac{7_{\text{dec}}}{2_{\text{dec}}} = \dfrac{0111_{\text{bin}}}{0010_{\text{bin}}}$

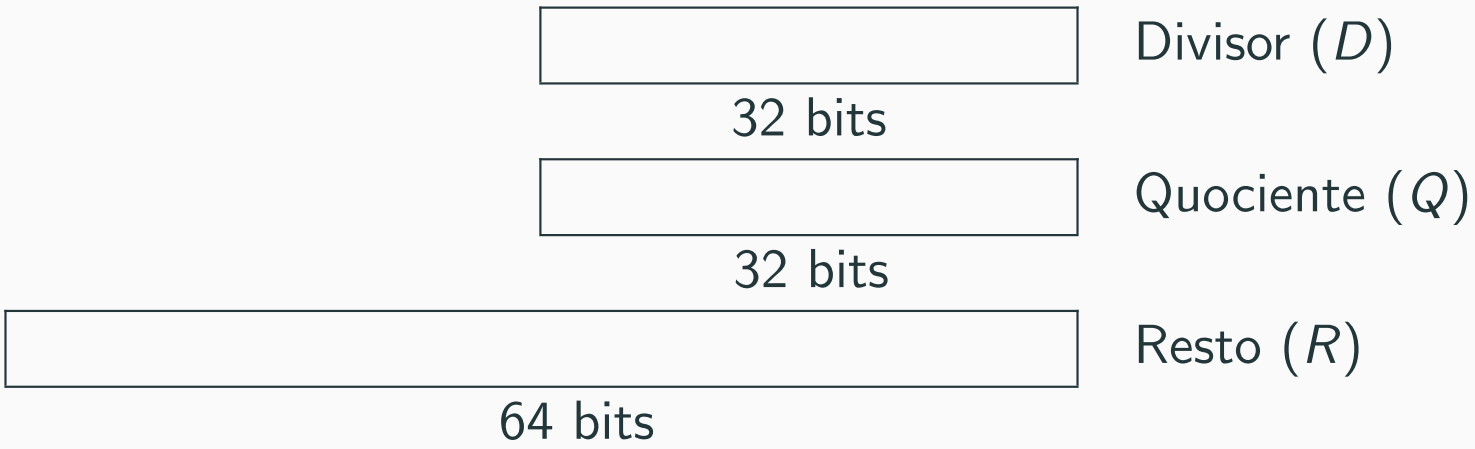
Primeira otimização

- Na primeira versão do algoritmo,
  - metade dos bits do divisor são nulos e
  - o primeiro passo nunca produzirá 1 em Q.

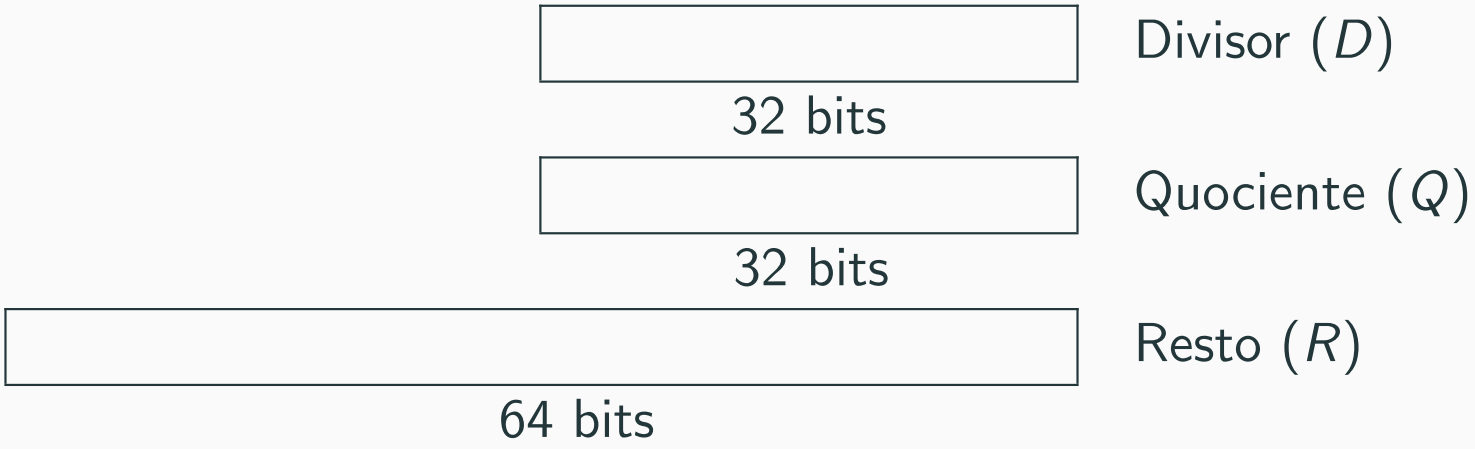
Primeira otimização

- Na primeira versão do algoritmo,
  - metade dos bits do divisor são nulos e
  - o primeiro passo nunca produzirá 1 em Q.

- Dessa forma:
  - usar D com 32 bits,
  - deslocar R à esquerda (ao invés de D à direita) e
  - operar D e os 32 bits mais significativos de R.



Primeira otimização: algoritmo



1. Defina contador $= 1$ e salve o Dividendo na parte *menos significativa* de R.
2. Faça um deslocamento à esquerda de 1 bit em R.
3. $R[63..32] = R[63..32] - D$.
4. Faça um deslocamento à esquerda de 1 bit em Q.
   - 4.1 Se $R \geq 0$, defina o bit menos significativo de Q como 1.
   - 4.2 Se $R < 0$, restaure o valor original dos 32 bits mais significativos de R ($R[63..32] = R[63..32] + D$).
5. Se contador < 32, faça contador=contador+1 e volte ao Passo 2.

Exemplo: $\frac{7_{\text{dec}}}{2_{\text{dec}}} = \frac{0111_{\text{bin}}}{0010_{\text{bin}}}$

Segunda otimização

- Na segunda versão do algoritmo,
  - ambos o quociente quanto o resto recebem deslocamentos à esquerda.

- Podemos combinar o **quociente** nos bits menos significativos do **resto**.
  - Com isso, o resto receberá um deslocamento à esquerda a mais.
  - Por isso, é necessário um deslocamento à direita para correção ao final.

Segunda otimização: algoritmo

1. Salve o Dividendo na parte *menos significativa* do Resto e defina contador $= 1$.
2. Faça um deslocamento à esquerda de 1 bit em R.
3. $R[63..32] = R[63..32] - D$.
4. 4.1 Se $R \geq 0$, faça um deslocamento à esquerda de 1 bit no Resto e defina o bit menos significativo do Resto como sendo 1.
   4.2 Se $R < 0$, restaure o valor original dos 32 bits mais significativos do Resto ($R[63..32] = R[63..32] + D$) e faça um deslocamento à esquerda de 1 bit no Resto.
5. Se contador < 32, faça contador=contador+1 e volte ao Passo 3.
6. Faça um deslocamento à direita dos 32 bits mais significativos do Resto.

- **Resultado**: Resto na parte mais significa de R, e quociente na menos significativa.

Exemplo: $\frac{7_{\text{dec}}}{2_{\text{dec}}} = \frac{0111_{\text{bin}}}{0010_{\text{bin}}}$

Divisão de inteiros com sinal

1. Salve o sinal do dividendo e do divisor, e torne-os positivos.
2. Faça a divisão usando o algoritmo.
3. Faça com que o resto tenha o mesmo sinal que o dividendo tinha ao início do procedimento.
4. Negue o quociente se o sinal do divisor era diferente do sinal do dividendo ao início do procedimento.

Exemplo

$-7_{\text{dec}} \div 2_{\text{dec}} = -3_{\text{dec}}$, resto -1_{dec}.

Instruções no assembly MIPS

- Dois registradores de 32
  - `HI`: 32 bits mais significativos
  - `LO`: 32 bits menos significativos

Instruções no assembly MIPS

- Dois registradores de 32
  - `HI`: 32 bits mais significativos
  - `LO`: 32 bits menos significativos

- Instruções
  - `div rs, rt` / `divu rs, rt`
    - resultado nos registradores `hi` (*resto*) e `lo` (*quociente*)
    - não há verificação automática de divisão por zero

Instruções no assembly MIPS

- Dois registradores de 32
  - `HI`: 32 bits mais significativos
  - `LO`: 32 bits menos significativos

- Instruções
  - `div rs, rt` / `divu rs, rt`
    - resultado nos registradores `hi` (*resto*) e `lo` (*quociente*)
    - não há verificação automática de divisão por zero
  - `mfhi rd` / `mflo rd`
    - move dos registradores `HI` e `LO` para o `rd`