Multiplicação e Divisão *bitwise*

MSc. Joao Rossi

IDP

joao.borba@idp.edu.br

github.com/joaorossi15

Multiplicação: Algoritmo

1. Inicializar $P \leftarrow 0$, $count \leftarrow 0$;
2. Verificar se $LSB(Q) == 1$;
3. Se sim, $P \leftarrow P + M$;
4. Deslocamento logico de um bit à esquerda em M («= 1) ;
5. Deslocamento logico de um bit à direita em Q (»= 1);
6. Verificar se $count == 32$;
7. Se sim, finalize o algoritmo;
8. Se não, count++ e retorne ao inicio.

Multiplicação: Como Otimizar?

1. Podemos otimizar o algoritmo básico:
   - E se, ao invés de deslocar M à esquerda, deslocarmos o produto para a direita?
   - E se salvarmos o multiplicador na porcão menos significativa do produto?
2. Com isso:
   - Não precisamos de outro local para salvar Q;
   - M precisa de apenas 32 bits.

Multiplicação: Algoritmo Otimizado

1. $P[63..32] \leftarrow 0$;
2. $P[31..0] \leftarrow Q$;
3. Se $P[0] == 1$, $P[63..32] \leftarrow P[63..32] + M$;
4. Deslocamento de 1 bit à direita em P utilizando *carry* da soma;
5. Verificar se $count == 32$;
6. Se sim, finalize o algoritmo;
7. Se não, $count++$ e retorne ao inicio.

Divisão: Algoritmo

1. Inicializar $count \leftarrow 0$, $R[31..0] \leftarrow$ Dividendo;
2. Deslocar à esquerda R em 1 bit;
3. $R[63..32] \leftarrow R[63..32] - D$
4. Deslocar à esquerda Q em 1 bit;
5. Se $R \geqslant 0$, então $LSB(Q) \leftarrow 1$;
6. Caso contrário, restaure o valor original dos 32 bits mais significativos de R ($R[63..32] \leftarrow R[63..32] + D$);
7. Verificar se $count == 32$;
8. Se não, $count++$ e retornar ao passo 2.

Divisão: Como Otimizar?

1. Ambos o quociente quanto o resto recebem deslocamentos à esquerda;
2. Podemos combinar o quociente nos bits menos significativos do resto:
   - Com isso, o resto receberá um deslocamento à esquerda a mais;
   - Para corrigir: um *shift* à direita.

Divisão: Algoritmo

1. Inicializar $count \leftarrow 0$, $R[31..0] \leftarrow$ Dividendo;
2. Deslocar à esquerda R em 1 bit;
3. $R[63..32] \leftarrow R[63..32] - D$
4. Se $R \geqslant 0$, então deslocar R à esquerda e $LSB(R) \leftarrow 1$;
5. Caso contrário, restaure o valor original dos 32 bits mais significativos de R ($R[63..32] \leftarrow R[63..32] + D$) e deslocar R à esquerda;
6. Verificar se $count == 32$;
7. Se não, $count++$ e retornar ao passo 3;
8. Descolar à direita os 32 *hi bits* de R.

Atualmente

- Processadores modernos utilizam unidades de multiplicação e divisão dedicadas, reduzindo as operações em poucos ciclos de *clock*

Atualmente

- Processadores modernos utilizam unidades de multiplicação e divisão dedicadas, reduzindo as operações em poucos ciclos de *clock*
- Em arquiteturas mais simples ainda podem existir algoritmos iterativos, mas com otimizadores

Atualmente

- Processadores modernos utilizam unidades de multiplicação e divisão dedicadas, reduzindo as operações em poucos ciclos de *clock*
- Em arquiteturas mais simples ainda podem existir algoritmos iterativos, mas com otimizadores
- Mesmo com otimizações, a divisão costuma levar mais ciclos que a multiplicação devido à sua maior complexidade de implementação e paralelização, muitas vezes necessitando de algoritmos de aproximação (float)