Programação Paralela: das *threads* aos FPGAs

Processamento Vetorial (SIMD)

Prof. Ricardo Menotti
menotti@ufscar.br

Prof. Maurício Acconcia Dias
macccdias@gmail.com

Prof. Helio Crestana Guardia
helio.guardia@ufscar.br

Departamento de Computação
Universidade Federal de São Carlos

Atualizado em: 7 de maio de 2020

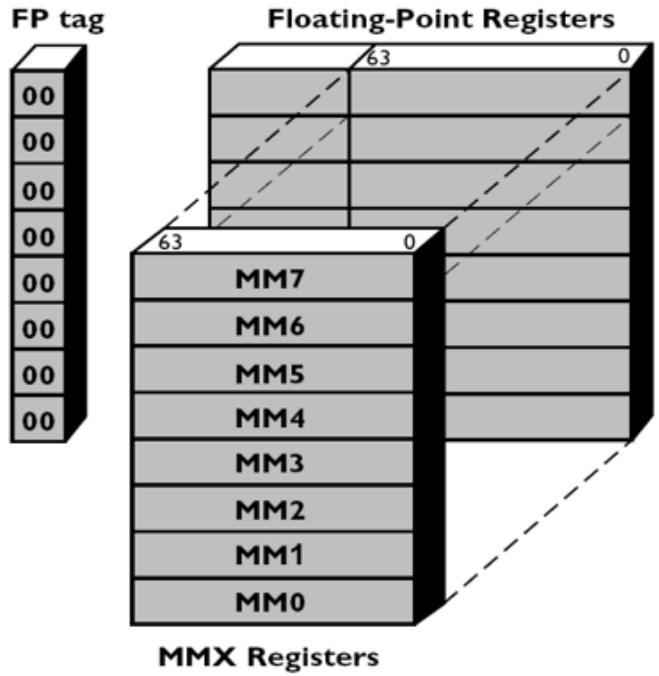
Roteiro

Histórico

- Na década de 90 as aplicações de multimedia (imagem, som, video) se popularizaram
  - Uso de tipos de dados pequenos
    - pixels: 8-bits
    - áudio: 16-bits
  - Repetição de operações
  - Paralelismo inerente
- Surgiram recursos de processamento SIMD em várias arquiteturas.
  - MMX – Intel
  - NEON – ARM
  - AltiVec – Apple/IBM/Motorola

Histórico: MMX

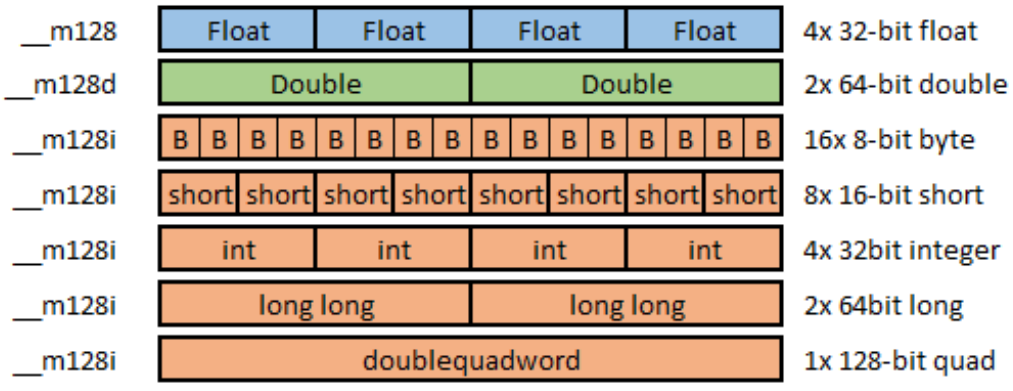
- Novo tipo de dados: 64-bit packed (empacotado).
- Operações com números inteiros apenas
- Reutiliza registradores de ponto flutuante
  - Não é possível misturar instruções MMX com operações de ponto flutuante (sem grande perda de desempenho).



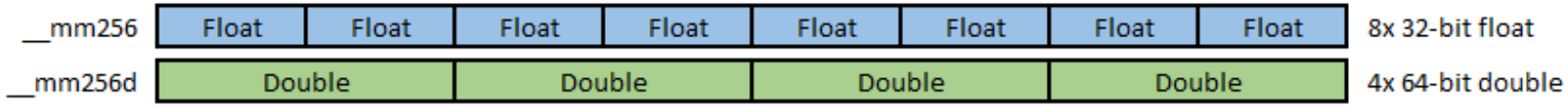
Processamento Vetorial

Registradores

SSE Data Types (16 XMM Registers)



AVX Data Types (16 YMM Registers)



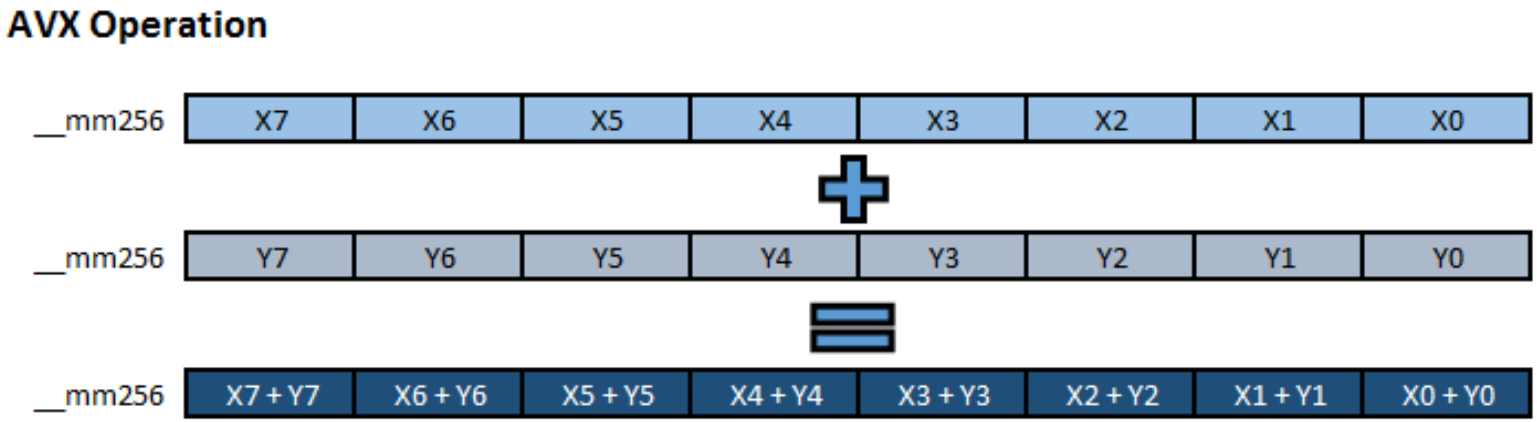
__mm256i *256-bit Integer registers. It behaves similarly to __m128i.Out of scope in AVX, useful on AVX2*

Processamento Vetorial

Funcionamento

Executam a mesma operação em múltiplos elementos de dados paralelamente:

```
vaddps ymm0, ymm1, ymm2
```

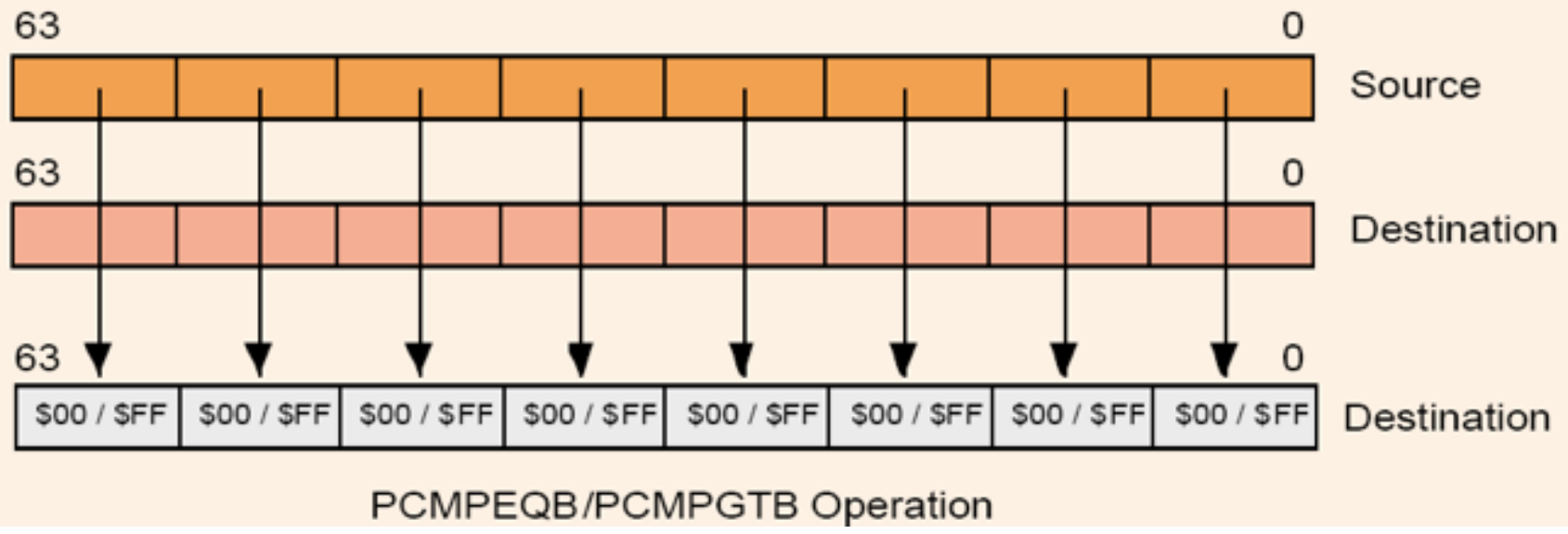


Processamento Vetorial

Comparações

Não modificam CFLAGS:

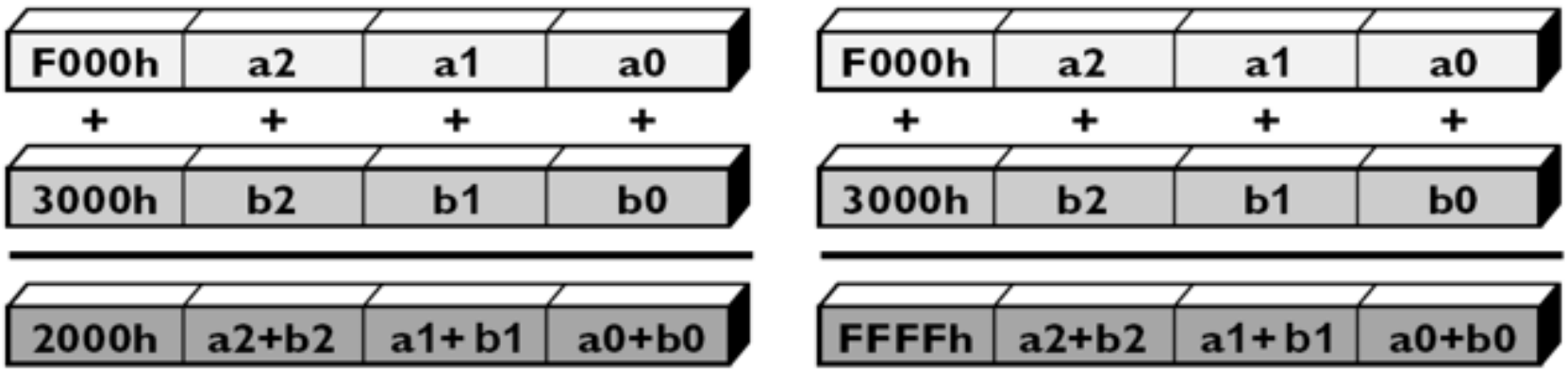
`pcmpeqb mm0, mm1`



Processamento Vetorial

Aritmética de Saturação

- Útil em aplicações gráficas
- Quando uma operação resulta em *under/overflow*
  - Resultado será o maior/menori inteiro representável
- Pode ser com ou sem sinal



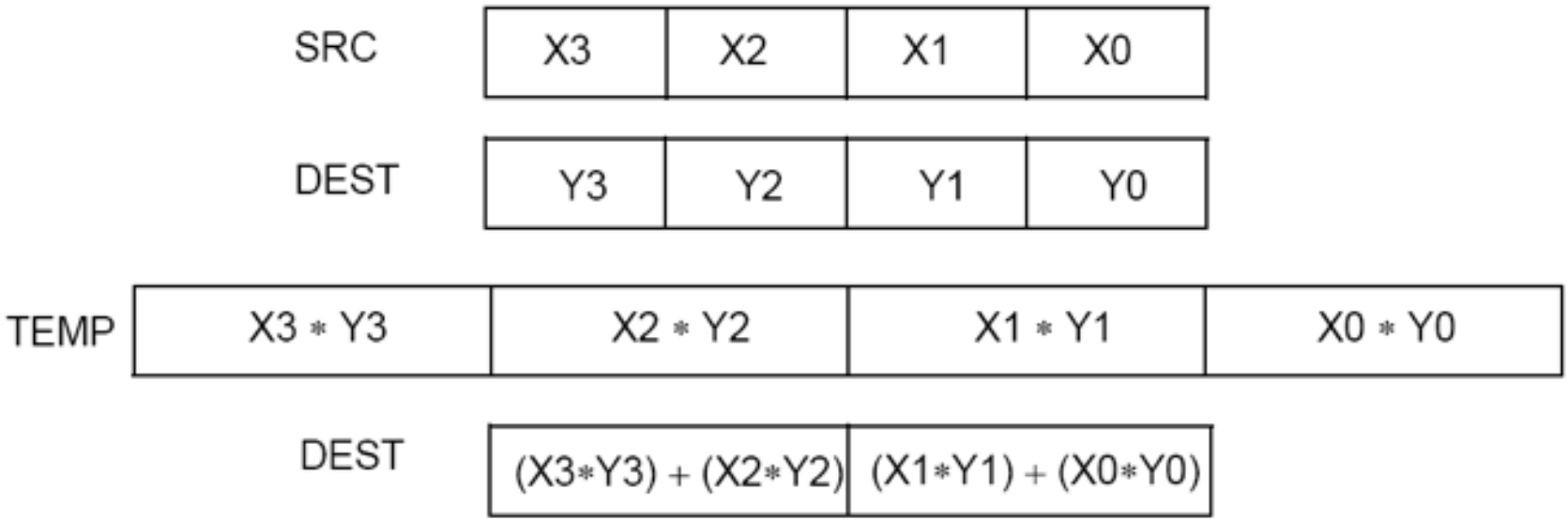
wrap-around *saturação*

Processamento Vetorial

Multiplica e Acumula

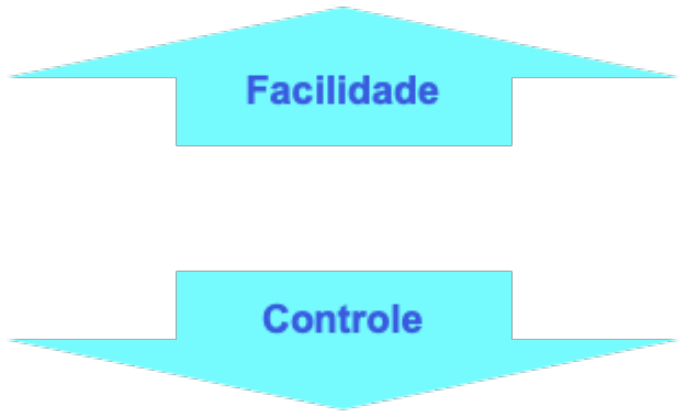
`pmaddwd xmm0, xmm1`

`pmaddwd ymm0, ymm1, ymm2`



Como usar?

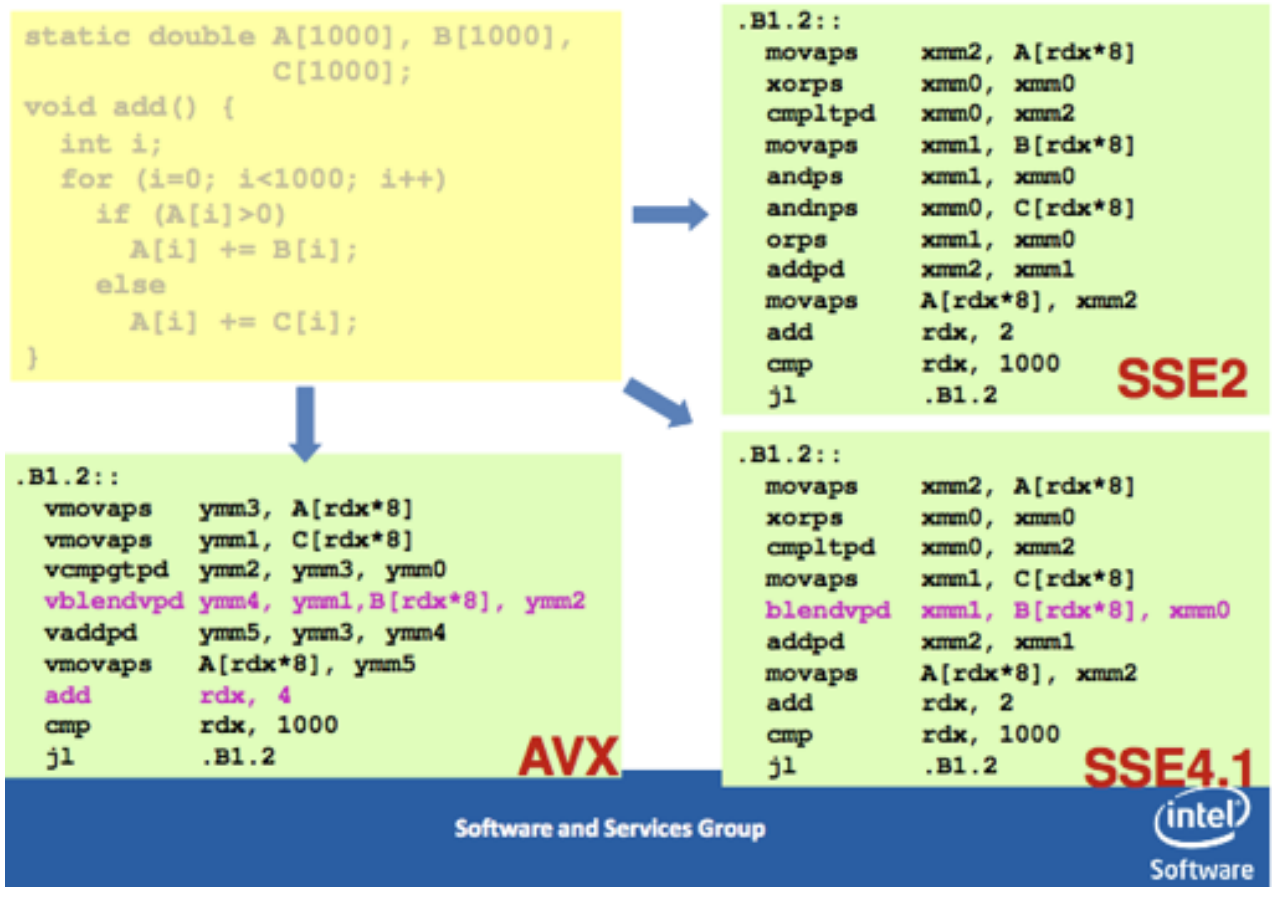
- Compilador (autovetorização)
- Bibliotecas (IPP, MKL, etc)
- Notação específica (Cilk Plus)
- Compilador (#pragma)
- Tipos/Funções intrínsecas
- Assembly[1]



[1] Exemplo

Como usar?

Autovetorização




```
cc add.c -mavx -S -o add.s
```

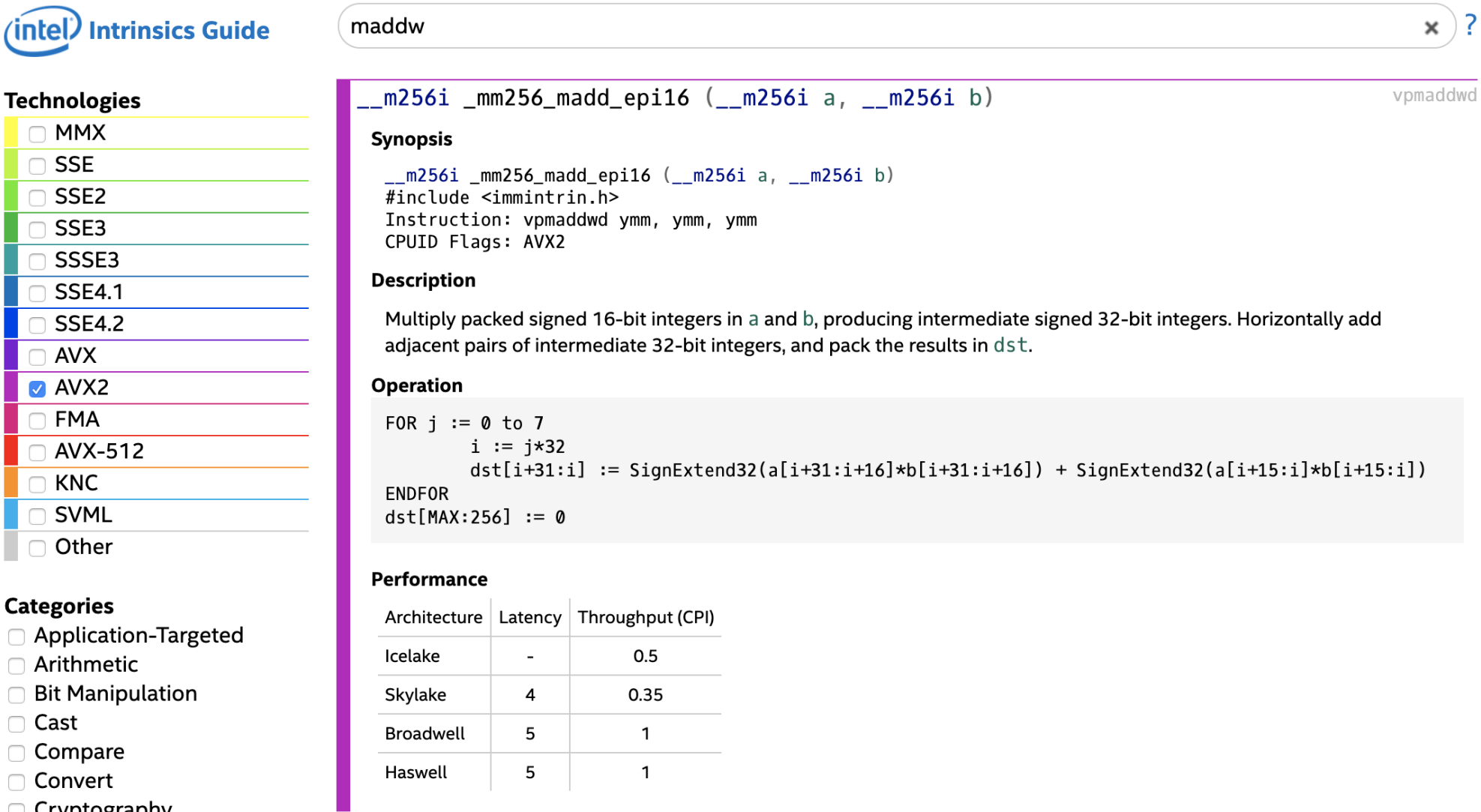
Como usar?

Autovetorização

- Pode encontrar problemas:
  - Dependências entre as iterações
  - Tipos de dados misturados
  - Condição muito complexa
  - Índice muito complexo
  - Vetorização ineficiente
  - Outros...

Como usar?

SIMD/Vector intrinsic



Como usar?

Compatibilidade

- Programas vetoriais são muito dependentes do ISA
- Onde o programa irá rodar?
  - Sempre na mesma máquina
  - Em duas máquinas diferentes
  - Em um cluster com máquinas diversas
  - Não tenho a menor ideia!
- CPU dispatch (`cpuid`)
  - Manual ou automático
  - Funciona em processadores antigos
  - Melhor desempenho em processadores novos

Como usar?

Pontos chave para programação

- Organização de dados eficiente (*data layout*)
- Eliminação de instruções de desvio (*if-then-else*)

- Kip R. Irvine, Assembly Language for x86 Processors, 7e, Pearson, 2015
- **Vetorização SSE & AVX**
- Intel® Intrinsics Guide
- x86 and amd64 instruction reference

Programação Paralela: das *threads* aos FPGAs

Processamento Vetorial (SIMD)

Prof. Ricardo Menotti
menotti@ufscar.br

Prof. Maurício Acconcia Dias
macccdias@gmail.com

Prof. Helio Crestana Guardia
helio.guardia@ufscar.br

Departamento de Computação
Universidade Federal de São Carlos

Atualizado em: 7 de maio de 2020