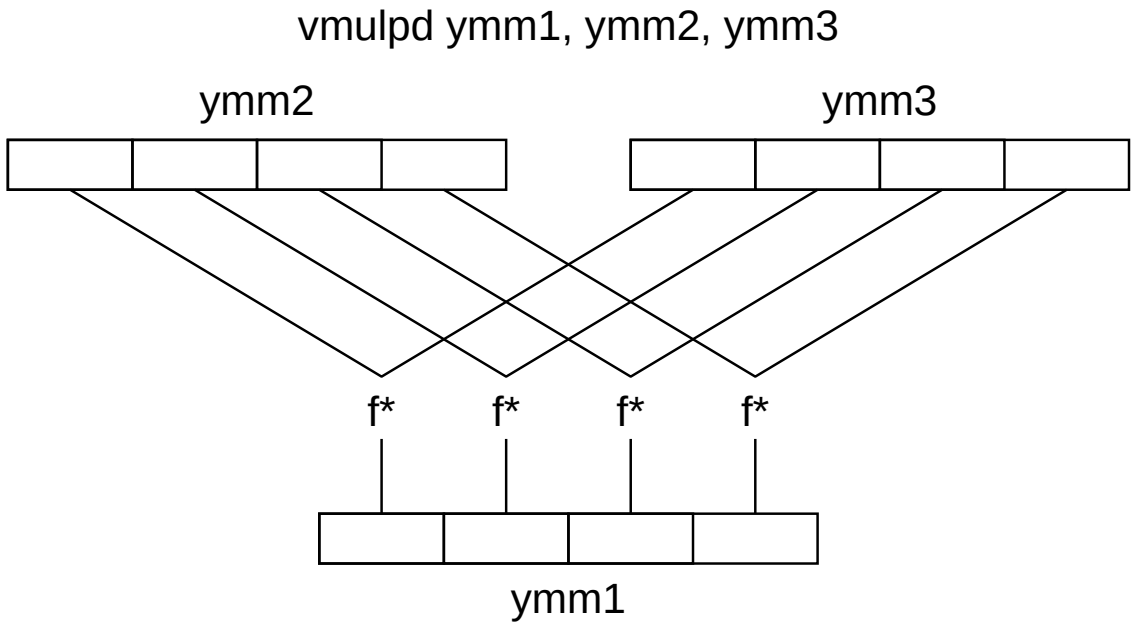
Gedanken zu SIMD und Vektorisierung

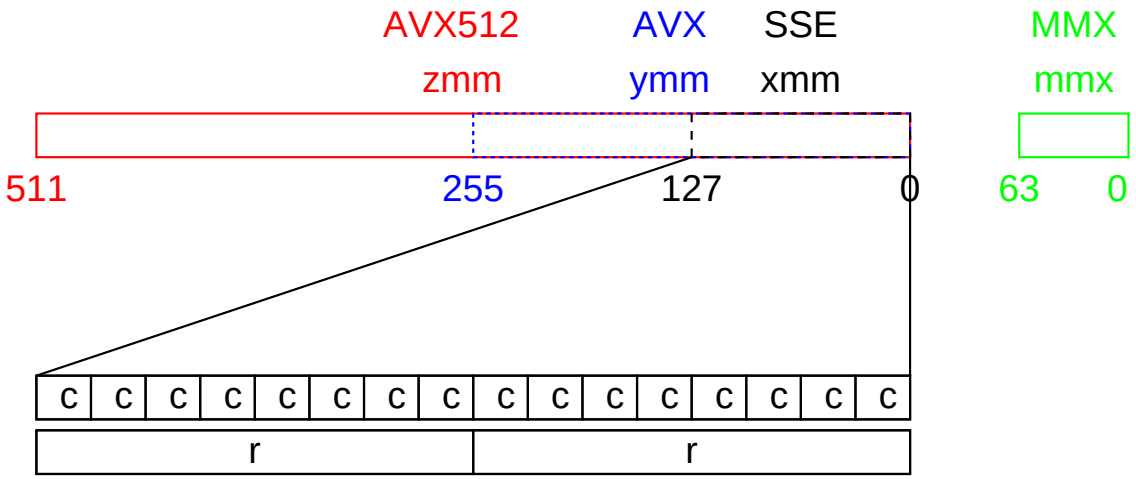
M. Anton Ertl, TU Wien

SIMD: Single Instruction, Multiple Data



- Cray-1
- Intel, AMD: MMX, 3DNow, SSE, SSE2-4, AVX, AVX512
- PowerPC: Altivec
- ARM Neon

SIMD-Register und Operationen



- Parallele Bearbeitung
- Anordnung ändern (Shuffle)
- Werte zusammenfassen oder verteilen
- Maskierung
- Spezialbefehle

Programmiersprachen

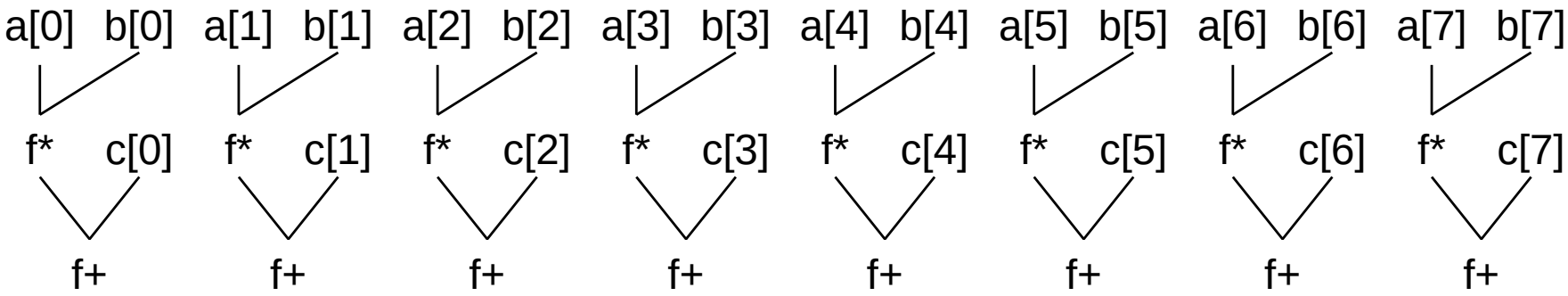
```
/* Intrinsics */
__m256d c _mm_mul_pd(a, b);

/* Auto-Vektorisierung */
for (i=0; i<4; i++)
  c[i] = a[i] * b[i];

/* GCC Vector Extensions */
typedef double v4d __attribute__ ((vector_size (32)));
c = a*b

!Fortran Array language
REAL, DIMENSION(4) :: a,b,c
c = a*b;
```

Und in Forth?

- Vektor-Stack
  Vektorlänge dynamisch
  Operationen zwischen gleich langen Vektoren
  Elementtypen nicht gecheckt (it's Forth!)

- Vorteile
  Elemente unabhängig
  Leicht parallelisierbar

a 8 floats v@ b 8 floats v@ vf* c 8 floats v@ vf+



```
a 8 floats v@ b 8 floats v@ vf* c 8 floats v@ vf+
```

```
    Einfach:
    ... setup ...
L1: vmovupd ymm0, [rbx]
    vmovupd [rdx], ymm0
    update rbx, rdx
    loop L1
    ... setup ...
L2: vmovupd ymm0, [rbx]
    vmovupd [rdx], ymm0
    update rbx, rdx
    loop L2
    ... setup
L3: vmovupd ymm0, [rbx]
    vmulpd ymm0, ymm0, [rdx]
    vmovupd [rbx], ymm0
    update rbx, rdx
    loop L3
    ....
```

```
    Effizient:
    ... Setup ...
L1: vmovupd ymm0, [rbx]
    vmulpd ymm0, ymm0, [rdx]
    vaddpd ymm0, ymm0, [rsi]
    vmovupd [rdi], ymm0
    update rbx, rdx, rsi, rdi
    loop L1
```

Und der Rest von Forth?

- Vektoren in einem separaten Bereich
- aber leben nur auf dem Vektor-Stack
- Aus dem Speicher lesen (z.B. v@)
- In den Speicher schreiben
  ```
  a 8 floats v@ b 8 floats v@ vf* c 8 floats v@ vf+ d 8 floats v!
  ```
- Problem: Mögliche Überlappung
  Laufzeittest auf Überlappung?
  Schreiben mit Zwischenablage
- Neu allozierter Speicher?
  ```
  a 8 floats v@ b 8 floats v@ vf* c 8 floats v@ vf+ valloc dp !
  ```

Zusammenfassung

- SIMD-Befehle existieren
- Sollten von Forth aus nutzbar sein
- Vektor-Stack
  Maschinen-unabhängig
  Keine unnötigen Abhängigkeiten $\Rightarrow$ optimierbarer Code
- ! problematisch

Ausblick?

- GPU
- BLAS
- Matrizen