Vorlesung 7

Aufgabe 1

Explain three vectorization clauses of your choice that can be used with #pragma omp simd

1) aligned – damit kann man dem Compiler signalisieren, dass Arrays an bestimmten Speichergrenzen ausgerichtet sind. Wenn Compiler weiß, dass die Daten korrekt ausgerichtet sind – kann er den Code effizienter vektorisieren
2) safelen – maximale sichere Vektorbreite (es ist sicher, jeweils bis zu n Elemente gleichzeitig zu verarbeiten)
3) reduction – damit kann man dem Compiler mitteilen, wie er mehrere parallele Berechnungen (wie eine Summe zum Beispiel) zum Endwert kombiniert

Aufgabe 3

Read the article Intel MMX for Multimedia PCs.
Discuss two things you find particularly interesting

1. Ich fand interessant, dass 32-Bit-CPUs vor MMX diese 32bits nicht vollständig genutzt haben obwohl sie das Potenzial dazu hatten. Es scheint mir eine zu offensichtliche Verbesserung zu sein. Aber es ist auch leicht zu reden, wenn etwas bereits entwickelt wurde :)

2. Das Beispiel mit dem Cursor fand ich interessant. Ich habe nie darüber nachgedacht, welche Operationen dahinter stecken.

Ich habe erfahren, dass es ein Beispiel von Data-Dependent Computation ist, da jeder Pixel davon abhängt, ob er Teil des Sprites oder des Hintergrunds ist. Dass Sprite in einem 2D-Array gespeichert wird und dass um ihn über eine Szene zu legen, überprüft das System für jeden Pixel, ob er dem „clear color“ entspricht (quasi transparent) – wenn ja, bleibt der ursprüngliche Bild erhalten (1), wenn nicht wird der Pixel in das Ausgabebild geschrieben(0).

Dann wird die Maske umgekehrt – die sichtbaren Sprite Pixel werden 1. Danach wird das ursprüngliche Bild bearbeitet, sodass die überdeckten Bereiche entfernt werden (dort wo Sprite 1 ist, wird 0 geschrieben). So entstehen 2 „Bilder“ – Sprite mit nur sichtbaren Pixeln und Szene ohne überdeckte Teile und anschließend werden sie mit OR-Operation kombiniert. So einfach, nur mit compare, and, not, or und ohne pixelweise if-else Vergleiche geht es mit MMX.