Computer Architecture

Richard Röttger – SDU, Fall 2017

Lab 4

Lab 4 – Using SSE instructions

In this lab we will learn how to use the SSE extensions. When SSE was introduced, the CPUs received 8 additional 128 Bit registers `XMM0-7`. Each of these registers can act as the following set-up:

- 2 x 64-bit floating points (double precision) 2 x 64-bit integers
- 4 x 32-bit floating points (single-precision) 4 x 32-bit integers
- 8 x 16-bit integers
- 16 x 8-bit characters (bytes)

Generally speaking, in order to work with SSE instructions, first the registers need to be loaded with the appropriate data. For example, in order to load 4 single precision floating point numbers:

```
movaps     mem,%xmm0
```

where mem is a memory location of the floats in question. For loading different datatypes (e.g., 8 16-Bit integers), different mov-Instructions are used. The content of the entire register could now be seen as: xmm0 = [[float1],[float2],[float3],[float4]]

Afterwards, different operations can be performed, e.g.:

```
addps      %xmm0, %xmm1
```

Let us assume that xmm0 = [[float1],[float2],[float3],[float4]] and xmm1 = [[float5],[float6],[float7],[float8]]. After the operations, xmm1 holds the following data:

xmm1 = [[float1+float5], [float2+float6], [float3+float7], [float4+float8]]

Further information on using SSE instructions can be found here:
http://en.wikibooks.org/wiki/X86_Assembly/SSE

1. Calculate the Inner Product

The inner product of two vectors $u = (u_1, \dots, u_n)$ and $v = (v_1, \dots, v_n)$ is defined as $u \circ v = u_1 \cdot v_1 + \cdots + u_n \cdot v_n$.

Objectives:

- Store two vectors of single precision floats in the data section of your program
- Calculate the inner product using SSE instructions
- Output the result or inspect the result using gdb (see link above for instructions)

2. Load the Vectors from Files

Objectives:

- Assume each vector consisting only of integers
- Load two files containing one vector each
- Calculate the inner product of both vectors using SSE instructions
- Output the result on the command line

Hint:

- Use the snippets from the obligatory assignment in order to load and parse the vectors

Extra:

- Assume the vector now consists only of 32-Bit integers
- Modify your program and the parsing routines such they parse and store only 32-Bit integers and the SSE instructions use 4 32-Bit integers at the same time