Subject: Re: SSE2(/AVX) and alignment issues
Posted by tojocky on Sun, 30 Jan 2011 10:34:25 GMT
View Forum Message <> Reply to Message

mirek wrote on Sat, 29 January 2011 21:29tojocky wrote on Sat, 29 January 2011 03:23mirek wrote on Sat, 29 January 2011 01:03

This is not a question. The question is whether _regular_ 'new' should return 16-byte aligned values or not. (And later, with AVX, 32, then maybe in 4 more years 64 etc...)

As long as we agree that allocating SSE2 stuff with 'new' is not a regular thing, we are at option 2..

Mirek

Mirek,

Can you give us an example of "new" realization and "allocator" realization?

```
struct AvxSomething {
  _m256 x;
};

Array<AvxSomething> foo;

foo.Add(new AvxSomthing); // not supported in option2. Actually, not even supported by any compiler today

foo.Add<AvxSomething>(); // supported in both options
```

Option2 could also support e.g.:

```
foo = New<AvxSomething>();
foo = new (Aligned<AvxSomething>) AvxSomething;
foo = NEW(AvxSomething);

Delete(foo);
```

(The crucial problem is that we need to know the type in new/delete).

Quote:

Why you are not agree with new realization (option 1)?

Because it wastes memory. It means that every single block allocated by 'new' must be a multiple 16 bytes allocated for SSE2.

Then 32 bytes for AVX and AVX is supposed to grow in width, so it can be easily 64 bytes etc.. So even if you request 24 bytes from new, you would waste 32 bytes (if we are 32 bytes aligned).

Moreover, U++ allocator today greatly benefits from the fact that alignment requirement is only 8 bytes. It would be possible to overcome this, but only at the price of quite a lot of wasted memory (or speed).

Still undecided. But if I consider that the issue does not stop at 32 bytes...

Mirek

What about to implement something like?

```
void* operator new( size_t size, size_t alignment ){
    return __aligned_malloc( size, alignment );
}
```

and in code:

```
AlignedData* pData = new( 16 ) AlignedData;
```

or

```
AlignedData* pData = new( 32 ) AlignedData;or
AlignedData* pData = new( 64 ) AlignedData;
```

or

```
AlignedData* pData = new( 128 ) AlignedData;
```

?

User need to know when he uses sse2/3/4 data

First option I saw in the source code by this link. I do now agree with this because it is waste of space and speed.

How do you want to implement the second method?