CONFIDENTIAL

Project Codename "SSE*Plus*" Overview

January 24, 2008

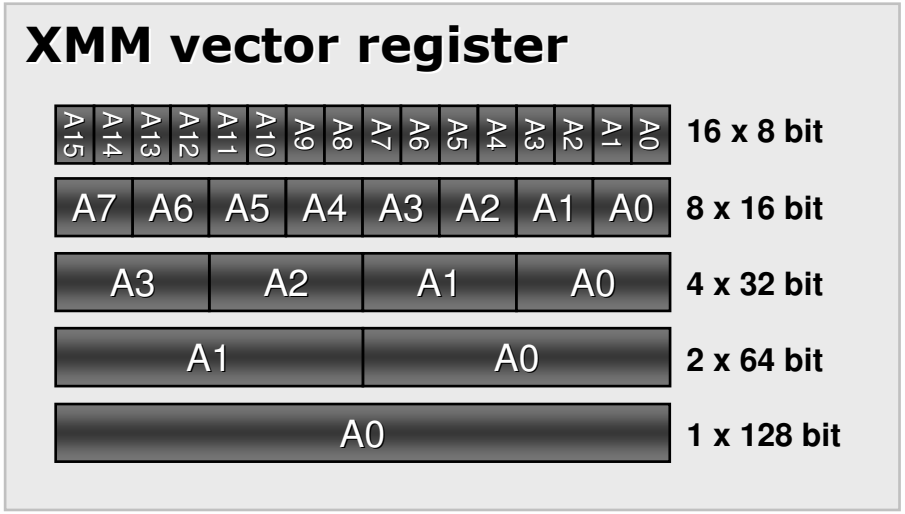
Streaming SIMD Extensions (SSE)

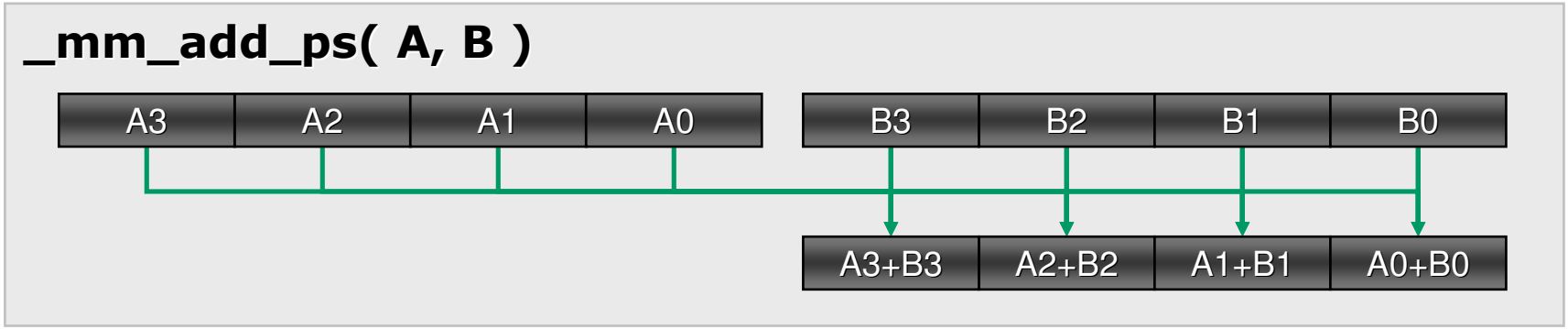
AMD Smarter Choice

Hardware instructions that operate on vector registers

Single instruction modifies register values in parallel

- 8 registers on 32 bit systems
- 16 registers on 64 bit systems
- 8 different SSE instruction sets
- Mixed support in hardware
- Accessible in most compilers through intrinsics (C function interface)





SIMD challenges

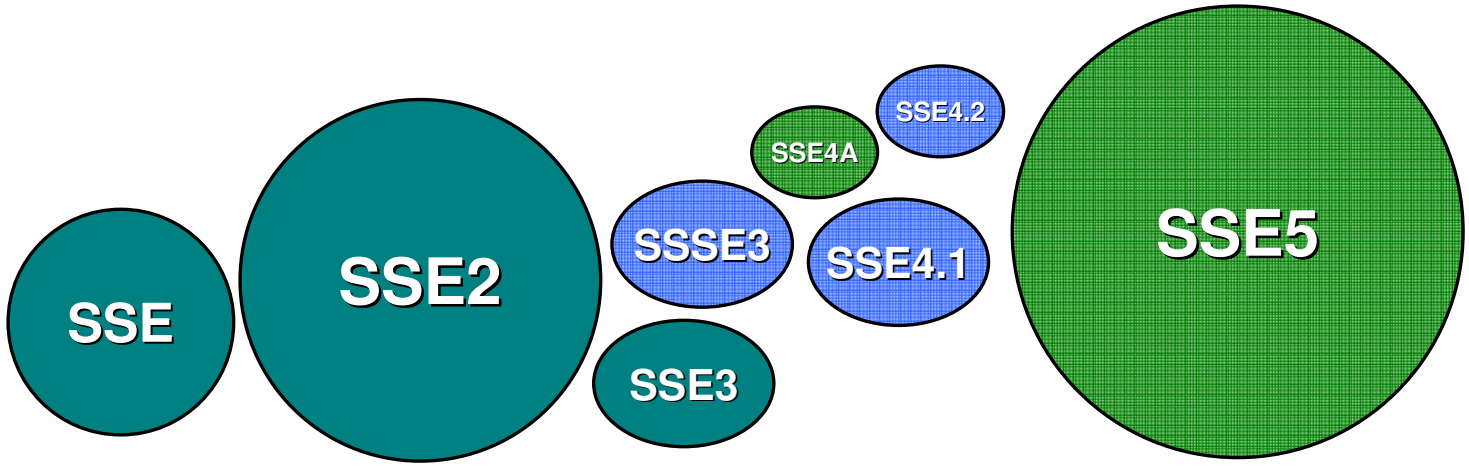
AMD
Smarter Choice

The 8 revisions of SSE (SSE*n*) have **471** instructions

Developers must diligently check CPUID

Developers write different functions for different SSE revisions

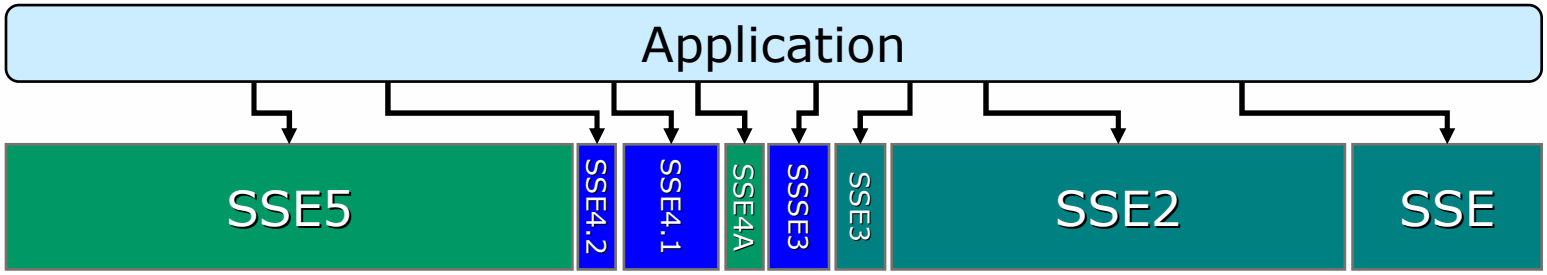
SSE*n* still has "missing instructions" (eg. 32bit integer divide)



SSE*n* Application Development

AMD
Smarter Choice

SSE*n* requires disparate code paths

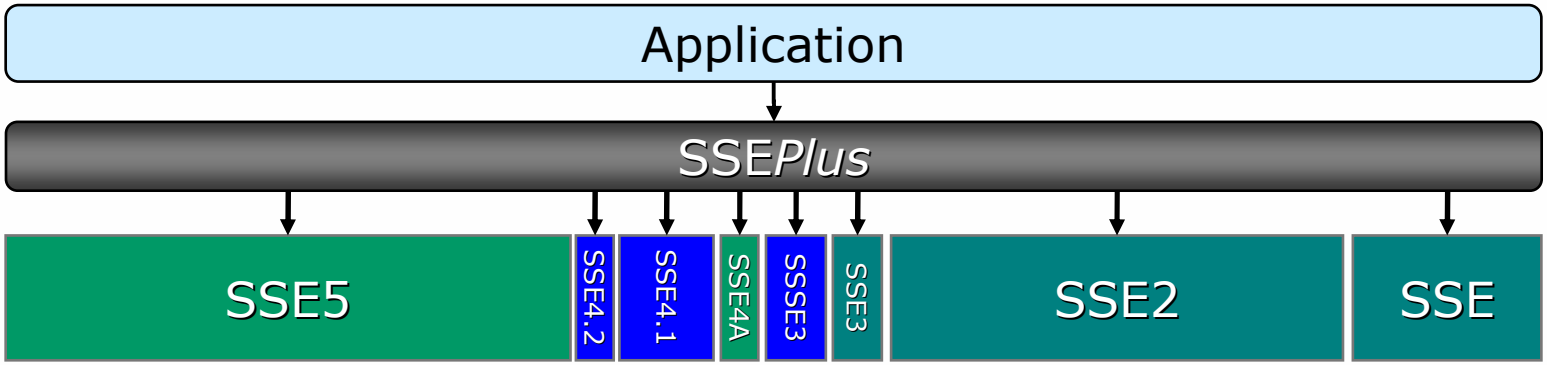



```
void Dispatch()
   switch( CPUID_CODE() )
      case ID_SSE2:    Run_v1()
      case ID_SSE3:    Run_v2()
      case ID_SSSE3:   Run_V2_sightly_differently()
      case ID_SSE4A:   Run_V2_with_insert()
      case ID_SSE4_1:  Run_V3()
      case ID_SSE4_2:  Run_v3_flavor_b()
      case ID_SSE5:    Run_v4()
      default:         Fall_back_to_reference()
```

"SSE*Plus*" Application Development

AMD Smarter Choice

SSE*Plus* enables **unified** code paths




```
void Dispatch()
   switch( CPUID_CODE() )
      case ID_SSE2:   Run_v1_SSE2 ()
      case ID_SSE3:   Run_v1_SSE3 ()
      case ID_SSSE3:  Run_v1_SSSE3()
      case ID_SSE4A:  Run_v1_SSE4A()
      case ID_SSE4_1: Run_v1_SSE41()
      case ID_SSE4_2: Run_v1_SSE42()
      case ID_SSE5:   Run_v1_SSE5 ()
      default:        Run_v1_REF  ()
```

"SSEPlus"

AMD Smarter Choice

Open Source library licensed under Apache v2.0

471 SSE*n* operations + 300 additional functions

C/C++ API similar to SSE*n* compiler intrinsics

```
__m128 _mm_hadd_ps( __m128 a, __m128 b )
// a and b are vectors of 4 floats
// Returns (b[3]+b[2],b[1]+b[0],…,a[1]+a[0])
```

Implementations optimized for multiple instruction sets

```
__m128 ssp_hadd_ps_SSE2( __m128 a, __m128 b )
// Optimized for SSE2
```

```
__m128 ssp_hadd_ps_SSE3( __m128 a, __m128 b )
// Optimized for SSE3
```

"SSE*Plus*" Continued

AMD
Smarter Choice

Developers can call targeted (*_SSE*n*) functions

```
void fn()
   …
   c = ssp_hadd_ps_SSE2( a, b )
   d = ssp_mul_ps_SSE2 ( c, a )
   …
```

Or combine generic functions with an architecture map file

```
#include "SSEPlus_MAP_AMD_F10h.h"

void fn()
   …
   c = ssp_hadd_ps( a, b )
   d = ssp_mul_ps ( c, a )
   …
```

"SSE*Plus*" Benefits

AMD
Smarter Choice

Develop with new instructions before hardware is available

Optimize once for target hardware, other platforms are easy

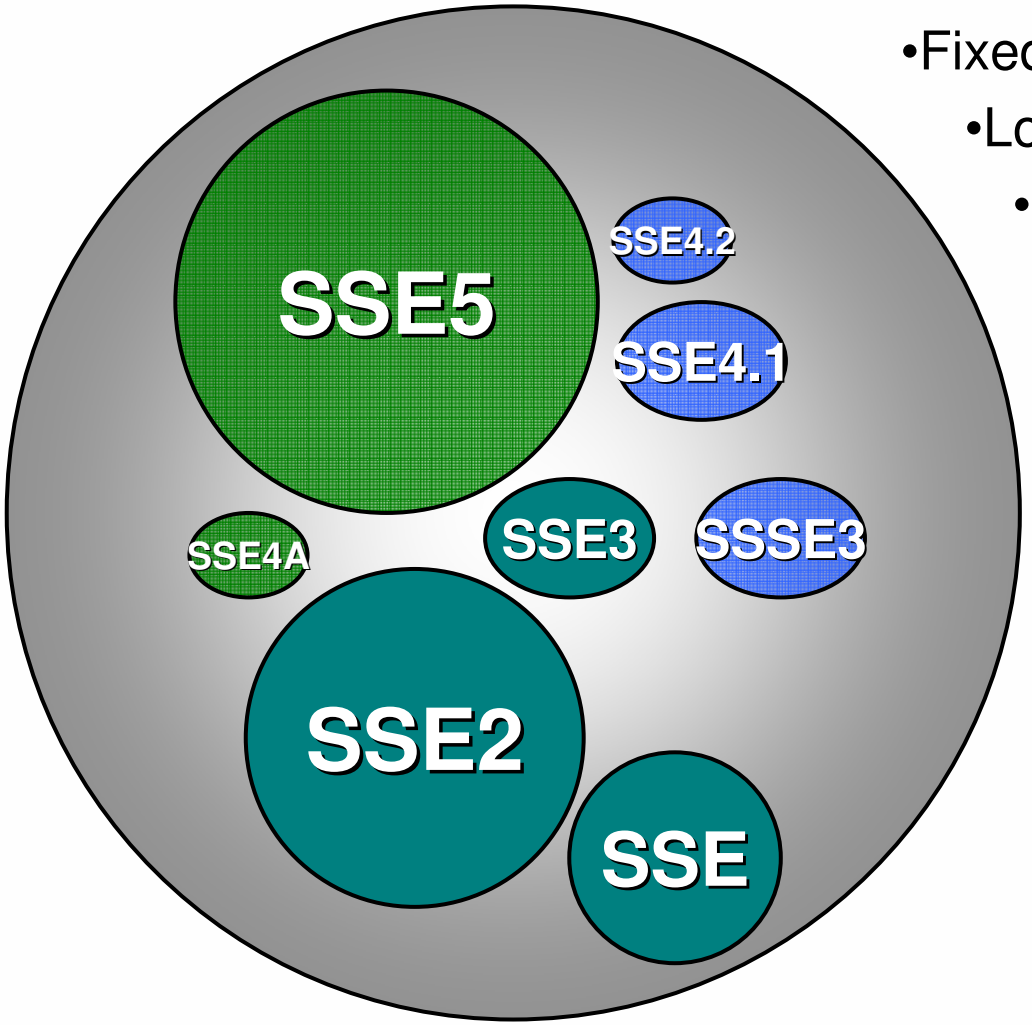
Ensure generated code conforms to target hardware

Stop worrying about instruction sets. Use instructions that match your algorithm

Open source: If a function is missing -> add it

Feedback loop: High value added functions may become hardware instructions

"SSEPlus"



300+ New Functions

- Arithmetic
- Fixed Accuracy
- Logical
- Pack / Unpack
- Trigonometry
- More

471 SSE Functions

- Simplified optimization
- Multi instruction compatibility

Open Source

- Immediate access to latest code

AMD
Smarter Choice

Trademark Attribution