Document Numbers: X3J16/96-0107
WG21/N0925
Date: May 29, 1996
Reply To: Bill Gibbons
bill@gibbons.org

# Generated Copy Assignment Base Class Copy Ordering

## Introduction

The ARM specified that:

> Objects representing virtual base classes will be initialized only once by a generated copy constructor. Objects representing virtual base classes will be assigned only once by a generated assignment operator.

For copy constructors, virtual base classes were (and still are) specified to be copied before nonvirtual base classes, in depth-first left-to-right order of first occurrence in the inheritance hierarchy. No such ordering was specified for generated copy assignment operators; but the same ordering was allowed and was used by some implementations.

The restriction that virtual base class subobjects may be assigned only once has since been removed, and the ordering for copy assignment base classes has been specified. The current specification is:

> The implicitly-defined copy assignment operator for class `X` performs memberwise assignment of its subobjects. The direct base classes of `X` are assigned first, in the order of their declaration in the *base-specifier-list*, and then the immediate nonstatic data members of `X` are assigned, in the order in which they were declared in the class definition. Each subobject is assigned in the manner appropriate to its type:
> — if the subobject is of class type, the copy assignment operator for the class is used;
> — if the subobject is an array, each element is assigned, in the manner appropriate to the element type;
> — if the subobject is of scalar type, the built-in assignment operator is used.
> It is unspecified whether subobjects representing virtual base classes are assigned more than once by the implicitly-defined copy assignment operator.

The new specification does not allow the copy constructor ordering, or any ordering in which a virtual base is copied early relative to the non-virtual bases. This is seen as overly restrictive.

## The Problem

The working paper still gives compiler the latitude to eliminate redundant copies of virtual base classes. But the simplest way to implement that is to follow the same ordering as for copy constructors - by collecting all the virtual base copying together, it becomes possible to reduce the overhead to the testing of a single flag. Unfortunately this ordering is no longer allowed by the working paper.

Consider this example:

```
struct V { void operator=(const V &); };
struct A { void operator=(const A &); };
struct B : V { };
struct C : A, B { };
```

Under the kind of ordering used by copy constructors, the bases would be copied in the order (V,A,B). But the current working paper specifies the order (A,V,B). Honoring this ordering requirement, while eliminating redundant virtual base subobject copies, requires more object code than would be needed to follow the copy constructor ordering.

### Requirements for Safe Copying

Any ordering scheme must follow this basic principle:

> The copying of the non-base-class portion of a subobject will not begin until all of its base class subobjects have been completely copied.

Both the copy constructor ordering (copy all virtual bases first) and the current copy assignment ordering (copy virtual bases as they are encountered in a walk of the hierarchy) meet this requirement. But in some sense the two orderings are extremes: for copy constructors virtual base subobjects are copied as soon as possible, while for copy assignment functions they are copied as late as possible.

### Proposal

I propose relaxing the ordering requirements to allow generated copy assignment operators to copy virtual base subobjects early, as long as the basic ordering requirements are met.

Replace the paragraph "It is unspecified whether subobjects representing virtual base classes are assigned more than once by the implicitly-defined copy assignment operator." with the following text:

> Copying of virtual base class subobjects need not follow the exact ordering given above:
>
> - The assignment of a subobject representing a virtual base class may be omitted if it has already been assigned.
>
> - The assignment of a subobject representing a virtual base class may be done earlier than specified by the above ordering, as long as the ordering between the first instance of the assignment of each virtual base subobject is preserved and no derived class subobject is assigned before any of its direct or indirect base class subobjects.
>
> It is unspecified whether any assignments of subobjects representing virtual base classes are omitted or performed early.

### Summary

The current specification for the order of copying of virtual base subobjects is overly restrictive, which is probably a mistake since part of the intent is to make implementation easier. Any ordering which obeys the basic requirements and which copies virtual base subobjects somewhere between the (early as possible) copy constructor ordering and the (late as possible) current ordering should be allowed.