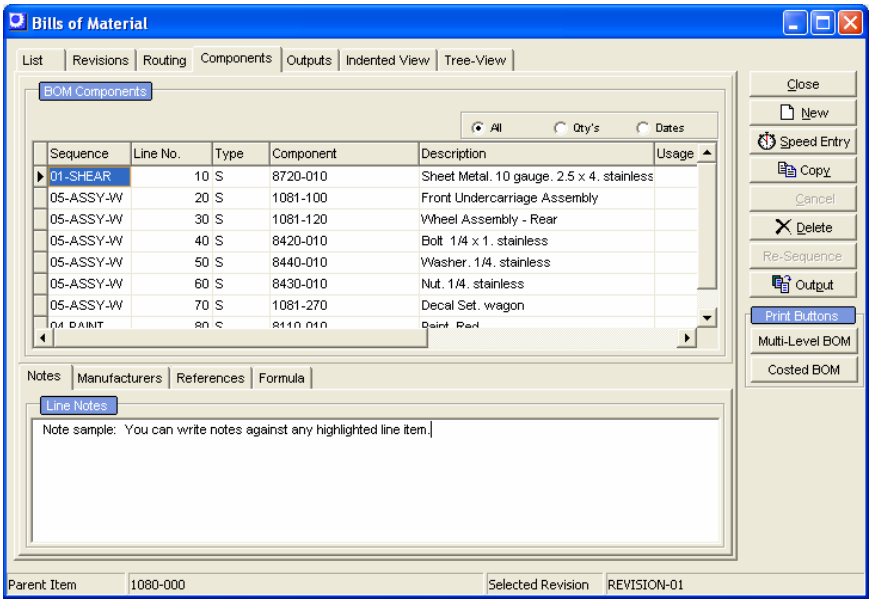
White Paper : BOM Structures

What is a bill of material?

A bill of material is a set of components that comprise an “assembly.”

What is the difference between a “product” and an “assembly?”

A “product” can consist of multiple assemblies and therefore, multiple BOM’s. An “assembly”, on the other hand, always refers to a single BOM.



BOM Components Entry Screen

“Building Block” analogy

A good way to visualize a product is to see it as a collection of building blocks. Each block is an assembly. Blocks are stacked in layers, like a pyramid. The final block at the top of the pyramid is the final product.

Each block needs a part number

In a manufacturing system, each block in the pyramid requires a parent part number to identify it. Each block’s components are then assigned to its parent part number in a separate BOM.

If you are a make to order company that has never used BOM's before, separate BOM's for each block may be a new concept. Many companies using spreadsheets for product specifications include all the components in the entire building block pyramid in a single list, perhaps using heading titles to identify groups of components.

Instead of heading titles, you now have to use parent part numbers. Instead of a single list of components for the entire pyramid, each block will now be defined with its own BOM and set of components.

What is a "subassembly?"

A "subassembly" is simply a term that refers to an assembly that is a component in another assembly. Technically, there is no difference between an assembly and subassembly – it's just a description.

The final product at the top of the pyramid is technically just another assembly. Among its BOM components may be one or more subassemblies, each of which has its own components and BOM. Within any given subassembly, its components can also be comprised of one or more subassemblies, forming another layer of building blocks at the base of the pyramid.

These layers of building blocks are referred to as BOM "levels." The building block pyramid is referred to as the "product structure."

Each assembly is given its own BOM

Each assembly is given its own parent part number and BOM. You do not concern yourself with what products the assembly might go into. If the assembly contains one or more subassemblies as components, you do not concern yourself with the components that comprise those subassemblies.

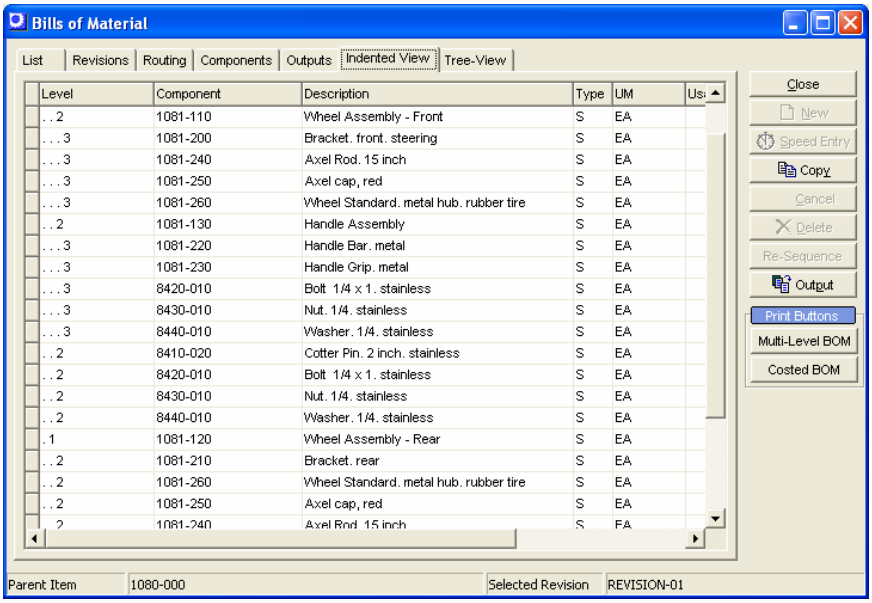
Once you create a BOM for an assembly, you can use its parent part number as a component in all BOM's that require this assembly as a subassembly.

What puts all the levels together?

Even though the final product can be a collection of multi-level BOM's, you can view it as if it were a single BOM through screens and reports. In DBA the "indented view" and "tree-view" inquiries and the *Multi-Level BOM* report show you an exploded view of all the assemblies that comprise the product structure, identified by level.

Levels work like this. Level '1' lists the BOM components for the final product. Level '2' lists the components associated with any subassemblies located in level '1'. Level '3' lists the components associated with any subassemblies located in level '2', and so forth.

When it comes to product costing, the *Cost Rollup* calculates a single *Estimated Cost* for the final product based on a rollup of costs among all the assemblies that comprise the product structure. Cost details for the product structure are listed on the *Costed BOM* report.



BOM – Indented View

How does the product structure get manufactured?

Let's say that you've defined a multi-level product that is a collection of BOM's for all the subassemblies involved. How does that product get manufactured?

There are two basic styles for manufacturing multi-level products.

The first and most common style is to make each subassembly on its own job and then issue the finished assembly to the next job(s) that requires it as a component.

The other style of manufacturing is to make all the subassemblies within a single job.

Why make each subassembly on its own job?

When an assembly is used as a component in many products, it is usually more efficient to give it its own job and then issue it to those jobs that need it as a component.

When you make an assembly on its job, you have the option of making it for stock so that it can be issued to jobs when needed without always having to be manufactured each time.

If an assembly goes through distinctly different work centers and processes than other assemblies within the product structure, it is best to make it on its own job so that it can be given its own routing and can be separately scheduled through the shop.

When is it desirable to make assemblies in a single job?

Making the entire product structure within a single job is only suitable under these conditions.

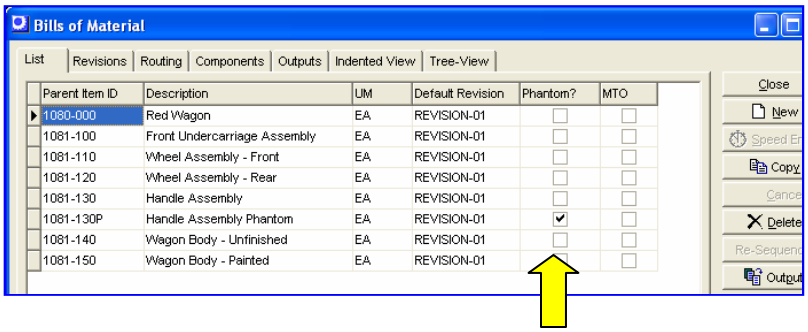
- Subassemblies are never stocked and are always made at the same time and always in proportional quantity to what is required by the final product.
- Subassemblies are manufactured primarily as “assembly” operations that don’t have to be scheduled separately though unique work centers and processes.

How do separate subassembly jobs get generated?

MRP automatically plans and suggests subassembly jobs based on a comparison of stock on hand with the demand from jobs that use the subassembly as a component. Subassembly jobs are backward scheduled from their required dates.

How does a single job for all subassemblies get generated?

If you want to manufacture a product’s subassemblies within a single job, those subassembly BOM’s must be flagged as a *Phantom* assembly in the *List* tab of the *Bills of Material* screen.



A “phantom” assembly is never stocked and does not have its own routing. When a job is created for a BOM that contains phantom assemblies, each phantom assembly’s components get pulled into that job. If a phantom assembly contains a component that also happens to be a phantom assembly, its components get pulled into the job as well. Thus you can have phantoms within phantoms.

When you use a single job like this, only one routing is used, which is the routing defined against the BOM of the final product. We suggest that each phantom assembly component be assigned to its own routing sequence so that each phantom assembly’s components are separately listed on the job traveler.

Which style is best?

In general, making each subassembly on its own job is the best way to go. It provides you with much more control over scheduling and work in process and gives you the option of stocking commonly used assemblies.

Making subassemblies within a single job is much less flexible, but can work well in pure “assemble to order” environments.