have been implemented in this way have also been adaptive Ferguson meta-assemblers.

Meta-assemblers are a special case of meta-compilers. Theoretically, a meta-compiler can be used as a meta-assembler. However, it seems that there have been some problems when existing meta-compilers have been used this way. Besides these problems, meta-compilers are not widely available. Nonetheless, attempts in this direction have been reported. We should note that meta-assemblers can be used to complement compilers and meta-compilers. In this case, the high-level programming language is compiled into assembly language, which is then translated into machine code. The second step can be performed by a meta-assembler.

**Complementary tools**

For various reasons, the translation of assembly programs into machine code is done in three separate phases. In the first phase (assembly phase), an “assembler” transforms the assembly program into a form of code called “assembled code.” In the second phase (linking phase), a “linker” transforms the assembled code into another form of code called “linked code.” In the third phase (relocating phase), a “relocating loader” transforms the linked code into still another form of code called “loadable code.” The loadable code contains the machine code in absolute form; it can be loaded into the memory of the target computer with the help of another tool called the “absolute loader.” Note that the linker and relocating loader can be combined into one tool, the relocating linking loader.

Splitting the translation process into the subprocesses described above is desirable because it allows program modules from different language processors (e.g., assemblers, Fortran compilers) to be combined, and it greatly facilitates the development of modular programs.

Another translation scheme has the assembler directly produce loadable code, in which case no linkers and relocating loaders are required. With this scheme, however, we lose the advantages cited above.

If the assembler-linker-loader scheme is adopted, then a meta-assembler should be accompanied by a “meta-linker,” a “relocating meta-loader,” and an “absolute meta-loader” (or by a “combined meta-linker relocating meta-loader” and an “absolute meta-loader”). These tools facilitate the development of linkers, relocating loaders, and absolute loaders in the same way meta-assemblers facilitate the development of assemblers. Special meta-languages are required for this purpose, namely a “meta-linking language” for the meta-linker and a “meta-loading language” for the relocating meta-loader. The first is used for describing linkers to the meta-linker and the second for describing loaders to the relocating meta-loader, in the same way that a meta-assembly language is used for describing assemblers to the meta-assembler.

It has been demonstrated that the linking process is target-computer-independent. Hence, meta-linkers can be constructed as simple linkers and no meta-linking language is required.

---

**Summary of the meta-assemblers**

A list of existing meta-assemblers is presented in Table 1. Although I do not claim that the list is complete, I believe that it is a good approximation to a complete list.

The table presents general characteristics of meta-assemblers, such as category, intended usage, and portability. Also given are details about the construction of the meta-assemblers, such as who constructed them, on which host computer they were developed, and in which language they are coded. More important, the table is set up to provide answers to questions about the assemblers that can be implemented by each meta-assembler: What sort of assembly languages do they assemble? Can they be readily used or does the user have to develop complementary tools such as linkers and loaders?

The table is divided into five sections headed general, construction, assembly language, meta-assembly language, and complementary tools.

The general section is divided into five columns. The first gives the name of the meta-assembler. The second gives the class of the meta-assembler according to the classification scheme shown in Figure 4. The third column indicates the meta-assembler’s usage as intended by its constructor—that is, the kind of assemblers that it can implement: software assemblers, firmware assemblers, or both. The fourth column gives a “yes/no” answer to the question of the portability of the meta-assembler.

---

**PROCESS/PRODUCT ENGINEER**

**...Thin Film Vapor Deposition**

CarboMedics is an internationally respected manufacturer of sophisticated components for medical implant devices. Our Austin, Texas facility has an immediate opening for a process/product engineer who can assume responsibility for thin film coating processes used in the manufacture of selected components.

Product duties will include interaction with customers and vendors, GMP compliance, product cost control and long range planning. Process duties involve thin film process characterization to identify process boundaries with emphasis on continuous operation during production. An MS or PhD in Materials, Physics, Physical Chemistry or Chemical Engineering and 3 to 5 years experience in thin film vapor deposition manufacturing is required.

CarboMedics can offer a competitive salary, extensive benefits and the potential for professional advancement in the beautiful Texas Hill Country! Please forward a resume to our Personnel Department.

Ms. Carol Vaughn
CARBOMEDICS, INC.
1300-D East Anderson Lane
Austin, Texas 78752
(512) 837-9911

Equal Opportunity Employer M/F

---

April 1983