Custom Chip/ Card Design System

by A. M. Barone
J. K. Morrell

The Custom Chip/Card Design System (CCDS) is a set of software applications, tied together via a common data interchange, that is used for the design, analysis, and checking of custom electronic circuits. CCDS is intended to unite the separate electrical, logical, and physical design phases into a single design process. The underlying principle of the system rests on the idea of describing the product as a generalized network, with multiple overlapping views that correspond to the different design phases. In addition to the applications contained within CCDS, there are also links to other software tools for such things as circuit simulation.

Introduction and overview

Custom design is a time-consuming method of designing an electronic product. The process has been made more difficult for the designer because he must usually deal with an assortment of noncommunicating software tools that have each been developed separately. Within our own DA organization we found it difficult to integrate the custom programs. Unlike the standard masterslice system developed earlier, the custom applications always seemed to require special approaches that pulled the applications away from one another. System integration defied all of our early attempts.

The Custom Chip/Card Design System (CCDS) was developed to ease the task of custom design by interfacing an improved set of applications to a common data interchange.

CCDS can be used for the design, analysis, and checking of custom electronic circuits on any packaging level. The system is a component of the Engineering Design System (EDS) widely used throughout IBM. CCDS is intended to unite the separate electrical, logical, and physical design phases into a single design process. The underlying principle of the system is to view the product as a generalized network, with multiple overlapping "views" that correspond to the different design phases. This generalized network description supports the definition of abstract network data, as well as all graphic data necessary for schematic documentation and for physical manufacture. Application-dependent data can also be associated with any of the network or graphic data items.

The applications within CCDS operate under a common executive and include applications for schematic design, physical design, ground rule checking, logical-to-physical verification, and electrical layout analysis. Also supported are links to other software tools outside CCDS for plotting, circuit simulation, and mask generation. A system diagram is shown in **Figure 1**.

CCDS operates on System/370 hardware under the MVS operating system, although many of the applications will also operate under VM. The user interacts with the applications via 3250 graphic displays and 3270 alphanumeric terminals.

Design data reside on a direct-access data set that supports random updates and maintains history entries to undo any sequence of operations. All updates are applied so that hardware failures result only in the loss of one user transaction at most. Applications can keep private data on the common file or in other private data sets that are organized for specific application needs. Completed designs can be stored for later use on a partitioned dataset via a library mechanism.

Executive

The CCDS executive is an interactive function which provides access to all of the interactive applications available

in CCDS. As such, it supplies the structure within which CCDS can be developed as a system rather than a collection of disjoint applications.

The executive performs all data set allocations required by the applications and maintains them through the design session so that switching among the various applications is not adversely affected. The executive also provides error interception and recovery capabilities to prevent the loss of user data because of system or application problems, and automatically enlarges the available file space for the data interchange when an overflow condition occurs. These processes make extensive use of dynamic allocation and catalog management and employ sophisticated reconstruction and data management techniques to eliminate the need for user involvement.

Interapplication communications are handled by the executive and provide the means for more closely coupling the applications and enforcing specific design methodologies. CCDS has also developed a rather extensive set of services to eliminate their duplication among the various applications. Access to both this set of services and the subroutine package used to program the graphics display is provided by the executive.

During the implementation of the executive, a conscious effort was made to isolate the applications as much as possible from the external system environment. The result is a relatively easy migration path due to changes in this environment, including development of a VM version of CCDS, even though the original base was MVS.

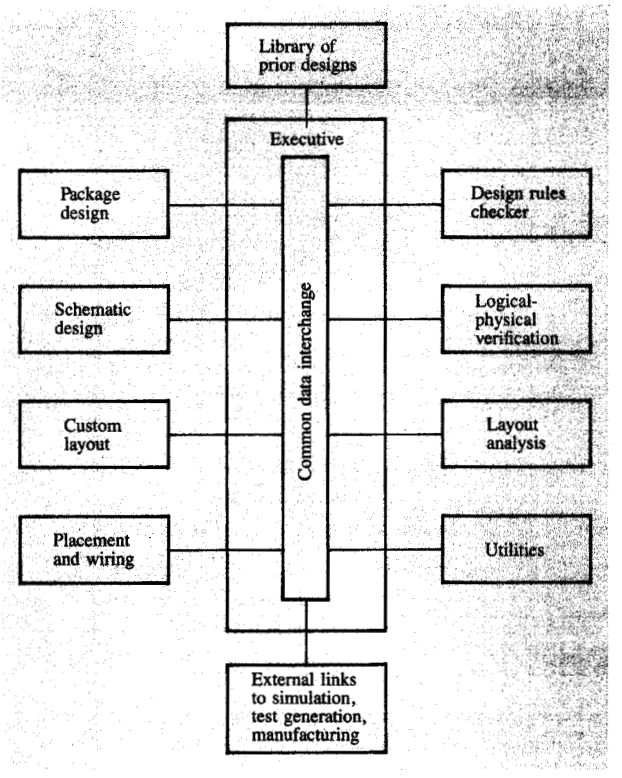

Figure 1

Custom chip/card design subsystem.

Technology rules applications

To maximize both designer efficiency and the integrity of the total design process, a design system must provide facilities for defining, controlling, and distributing technology information concerning the basic elements of the design. This information normally includes descriptions of legal packages, the corresponding sets of basic components, and the design ground rules which must be followed to ensure a manufacturable result.

• *Package definition*

The definition of a package within CCDS is stored within two categories of data. The first category contains information related to the specific manufacturing process being used and is collected for use within CCDS by a separate interactive package definition application. This application supports the specification of the number and order of the planes required, how each plane may be used (i.e., wiring, via, special reference, etc.), and how data are subdivided within a plane. Ground rules (minimum space, minimum width, etc.) and electrical parameters (resistance, capacitance, etc.) associated with each plane, so that any resultant designs using this technology are ensured to be manufacturable, are also specified using the package definition program. Information such as legal placement and wiring grids, allowable wire and via sizes, etc., can be optionally specified. In addition, a complete description of a package includes the manner in which it may be released to manufacturing, since certain packages may actually be released to the manufacturing facility in parts. For example, a card with two signal planes and two internal power planes, fully populated with a set of components, may actually result in the release of the power planes first, followed by the wiring data, with the component assembly data coming last. Similarly, it is desirable with gate array designs to release the fixed diffusion layers separately from the personality layers.

The second category of package data includes any graphical data required to describe the package. This may include such data as the package I/O pattern, outline, power distribution system, mounting holes, kerf information, and identifying PN/EC alphanumeric data. These data are provided by the technology designer using the normal custom physical design capabilities available within CCDS.

• *Component definition*

The definition of components within CCDS is handled by a separate component library function. This function allows the physical component descriptions to be related to their corresponding schematic or logic symbols so that complete integrity can be maintained between the design entry and physical design phases of the overall design process. This relationship may be simple (a one-to-one relationship), in which both the physical and symbol data are contained within a single model definition, or it may be complex (one-to-many), in which the physical and symbol data are defined in separate models and their relationship established by a third model. The graphical data required to display both the symbol and physical models are normally defined using the CCDS custom design capabilities, so their descriptions are limited only by the user's imagination and/or the technology limits. When the simple case exists that the symbol is rectangular, the component library function itself may be used to define the symbol. This was added primarily to define the "black box" logical descriptions of macros being used in a hierarchical design. Further, because a primary function of schematic or logic diagrams is to provide documentation for the design, extensive labeling capabilities are available as part of the component definition so that the final design documents are more easily understood.

• *Library*

The control and distribution of these package and component definitions are handled by the CCDS library. Once defined, they may be stored in the library, protected, and level-controlled so that no one other than the owner may modify them, and any new versions may be identified. Technology may then distribute this library to other design sites with the assurance that they will not be locally modified. In turn, when any resulting design reaches manufacturing, it is easy to determine that an "approved" set of elements has been used. This greatly eases the entry of designs into manufacturing.

Storing information in the CCDS library and retrieving it utilizes some interesting data management techniques. The internal format of the library is basically the same as the data interchange format used during designs. However, for transportability and concatenation abilities, the library uses the partitioned data set organization instead of the direct organization used for designs. Normally, a partitioned data set organization would require each package and component to be stored as a complete entity (including any nested data). Storing the elements as complete entities would greatly expand the size of the libraries and complicate the library update process (especially for components where multiple usage of symbol and physical models is common).

An alternative approach was therefore required. Instead of storing the elements as complete entities, CCDS has provided a library store function which separates the elements into their individual pieces (i.e., the model being stored and its nested models). These pieces are then stored separately as individual members of the library. The link among them is retained by maintaining a consistent set of model indices. As new elements are added, any duplicate models may be ignored, so that the data volume is kept to a minimum. In order to circumvent an eight-character limit for member names, a separate member is constructed which records a name-to-index correspondence, and the models are actually stored using their index. The corresponding library extract function reconstructs requested models into their original form.

A further consideration was how to interactively and dynamically provide a concatenation of libraries without impacting a large number of users when the concatenation order changed, as with a new release of the elements. Therefore, the library employs a linking mechanism where each library points to the next library in the chain. This allows the concatenation order to be predefined and changed, without impacting individual users, merely by changing the pointer in a base library (presumably maintained by a site-wide support group).

Design applications

The design process is normally divided into two phases: design entry and physical design. The design entry phase involves the definition of the design intent. On the basis of a specific package and the allowable set of components, a designer defines what components are required and how they are to be interconnected within the package to perform the desired function. This process usually requires the use of a verification tool (logic or analog/circuit simulator) to ensure that the design intent is at least nominally correct before it is committed to physical design. During the physical design phase, the designer places and wires the desired components on the basis of design intent, to provide the data necessary to drive manufacturing. In some cases, where predefined components are not available, this process also includes the design of the components themselves.

• *Design entry*

Since the majority of the designers currently using CCDS require schematics as their form of documentation, CCDS has been identified as a "schematic design system." As such, the primary simulation capability is oriented towards circuit or analog designs through a link to ASTAP [1].

CCDS design entry works from a library of symbols provided by the component definitions discussed previously. Since a major purpose of any design entry application is to provide design documentation, the data are organized into "sheets" which may then be plotted to form a set of manageable documents. Therefore, the design entry process is one of selecting the desired symbols from the library, placing them on the appropriate sheets, and identifying how

they are to be interconnected by means of a network definition process. In order to more easily define networks which span multiple sheets or to copy portions of the design from one sheet to another, up to four sheets may be displayed and interacted with simultaneously. Extensive text and labeling capabilities (including certain physical design information) are provided for more complete documentation. And, through direct access to the component definition function, both top-down and bottom-up hierarchical designs are supported.

Another major function of design entry is to prepare the data for physical design. This is provided by allowing the designer to associate specific physical components with the schematic symbols. On the basis of the component definitions, the networks are then expanded to include these components. For those designs where predefined components do not exist, the designer normally bypasses this step. Although he then loses the inherent integrity during physical design, he gains the flexibility of being able to design devices "on the fly."

Access to the circuit simulator is provided by translating the schematic design into the format required by the simulator. Additional device and execution parameters may be added to the schematic data in support of this so that the resulting output is sufficient to do a detailed circuit simulation. Any required modifications may be easily performed and the results re-analyzed until a satisfactory result is achieved. This capability is supported both before and after physical component assignment, although including the physical information provides more accurate results since the precise device characteristics are then known.

- *Physical design*

CCDS currently provides two applications for performing physical design. The first addresses custom designs through its generalized shape and data structuring capabilities. Although this approach may be the less efficient of the two, it provides the maximum flexibility. The second addresses those designs which only require the placement and wiring of predefined components. This application is totally dependent upon the design entry data structuring, and wiring is limited to stick figures, but it is clearly the more efficient way of handling this type of problem. Although both of these applications are primarily manual, plans are underway to access advanced automatic placement and wiring functions to increase the designer's efficiency even further.

Custom layout

The CCDS custom layout application is an outgrowth of an earlier program [2] named IGS which has been successfully used for detailed physical design. While the predecessor program was adequate from an interactive and graphic point of view, the major deficiency of this program and most other custom layout tools is that there is no direct connection to the schematic design entry phase. The improved CCDS layout program allows a designer to utilize the connection data from the schematic program to assist in the physical layout. Multiple windows depict area views of the design as well as a context- and net-oriented view.

The designer is permitted to nest any design within another, and each design can be fully annotated to show logical significance within the physical layout. Screen verification techniques allow a designer to interrogate an incomplete layout to highlight incomplete connections. The designer can also interrogate any screen element to list its associated parameters or have the program window around a named component, net, terminal, etc. that has been chosen from a list. Use of these features means that a designer is never "lost" within a complicated graphic image. All physical graphic details are now correlated to their logical or electrical counterparts. Fortunately, all these features have been provided using graphic techniques similar to those that were introduced in the predecessor layout program. The graphic primitives that are now supported include

- Points,
- Lines with width,
- Rectangles,
- Generalized polygons,
- Circles,
- Text operations.

Placement and wiring

The CCDS placement and wiring application utilizes information gathered during package and component definition to provide a more efficient means for performing physical design. That is, the information concerning the plane structure, placement and wiring grids, wire and via sizes, and physical ground rules is used in conjunction with the graphical descriptions of the package and required components to eliminate much of the physical design detail from the designer's concern.

Placement grids provide a more effective way to perform component placement, while the graphical information prevents a designer from placing components so that they interfere either with one another or with the package itself. During placement, the designer is graphically presented with component interconnection data and can at any time request a global, Manhattan-distance graph of the resulting wiring density. Aids are also available which provide an initial placement as well as individual selection of components based upon the connectivity information.

The predefined data for wiring grids, wire and via sizes, and the physical ground rules reduce the wiring problem to stick-figure routing of point-to-point connections. The designer graphically selects an overflow, assigns a specific wire and via size from the allowable set, and then routes the

connection on the desired wiring grid. The application automatically assigns the correct width and plane information and ensures that no shorts or spacing violations are introduced. Wiring may be performed on a horizontal, vertical, or 45-degree axis, and the designer may change the axis at any time (the wiring axis is automatically rotated 90 degrees when the current segment end-point is fixed). Changes in wiring planes are accomplished either by placing a via at the end of a wire segment or by explicitly requesting a plane change.

In addition to the actual design capabilities, this application provides a number of utilities to perform a variety of functions such as open/short checking and finding all nets having a specified span. There is also an automatic wiring capability which may be invoked on a single net, a group of nets, or all nets in the design. Although this function does not use as sophisticated an algorithm as some available programs, it still provides a significant advantage over complete manual design.

Since a design performed entirely within this application should be completely error-free, productivity gains are not only realized through the more efficient placement and wiring techniques employed, but significantly less time is required to successfully perform complete logical-to-physical and ground rule checking. The combination of these effects results in a major reduction of the total time spent in physical design.

Checking and layout analysis applications

The ground rules checker and layout analysis programs have been described before [3–5]. However, the front-end processors for these programs have been modified to operate in the CCDS environment, and the programs have been enhanced to support VLSI data volumes.

In the pre-CCDS environment, the checker constructed network relations as part of the ground rules check, prior to running the logical-to-physical verification step. Strictly speaking, this is no longer necessary within CCDS, where network information is preserved throughout the logical-to-physical transition. However, in order to allow designers the freedom to diverge from a strict network approach, and to ensure that no invalid assumptions are made, the checker disregards most structural data and reconstructs all network relationships during the checking phase.

The ground rules checking program has been enhanced over the years to handle larger designs more efficiently. In particular, to support VLSI, the program now has the ability to use alternate symbolic substitutes for previously designed and checked macros. This dramatically cuts down on the number of shape-to-shape comparisons and obviously reduces the CPU time to complete a check. Other improvements allow ground rules checking to be run in the foreground using the shapes data seen on the display screen of the layout program.

The layout analysis programs have been extended so that they now will electrically model the resistance and capacitance associated with a multiplanar conducting network in a single pass. The programs use a finite-element method as well as simpler heuristic techniques described in the references.

Utilities

The utility application is responsible for a host of functions to manipulate the design data in a variety of ways. For example, there are operations that

1. Merge separate designs.
2. Transfer portions of a design file to another file.
3. List the design data.
4. Delete portions of the design data.
5. Export and import data to/from external systems.

In addition, the utility application manages the library facility, which allows different designers to share common components throughout the IBM custom design community. There are over fifty functions available from several menu pages within the utility program. These functions are accessible from the executive or directly from the other applications.

User exits

In the development of a generalized design system such as CCDS, it is sometimes difficult to satisfy unique designer requirements, either because they are very technology-dependent or they do not have widespread applicability. This is especially true in custom designs. Therefore, CCDS has provided user exit capability within the technology rules, design, and checking applications. Interested users thus provide these missing functions so that they appear to be an integral part of CCDS. The user exit interface from the interactive (technology rules and design) applications provides access to the graphics display and the data interchange and allows for both textual and corresponding graphical feedback of results. The interface from batch checking applications provides extensive data management and manipulation capabilities so that users can extend the basic set of functions with relative ease.

Release

Since the final goal of any design is one of manufacturing, a design system is not complete until it addresses the question of release. The current set of CCDS users releases designs to both external vendors and internal IBM manufacturing. CCDS provides extensive documentation, in conjunction with a description of the physical design data [6], to support the vendor release process. This same physical design description can also be used, with user-provided administrative information, to drive internal manufacturing.

Both of these processes must be totally managed by the designer, which presents many possibilities for error. Therefore, CCDS is currently developing a separate release application utilizing a formal data interface, defined jointly by CCDS and IBM manufacturing, which will eliminate these errors and support both vendor and internal direct release requirements.

Conclusions

The use of CCDS has resulted in a measurable improvement in productivity for custom designs. While a great portion of the improvement can be attributed to better applications, it is obvious from designers' comments that the integration of the applications under a system architecture is equally responsible for much of the productivity gains.

Acknowledgments

The authors wish to thank their many colleagues who have contributed to the development of CCDS. Their efforts were truly outstanding. The work was accomplished under the managerial guidance of I. Saba, G. Hack, N. Santos, and L. Cesa.

References

1. *ASTAP—Advanced Statistical Analysis Program*, Order No. SH20-1118, available through IBM branch offices.
2. A. Barone, P. Carmody, J. Morrell, and C. Lovejoy, "An Interactive Graphics System for Large Scale Integration Design," *Proceedings of the International Conference on Interactive Techniques in Computer-Aided Design*, Bologna, Italy, September 1978.
3. C. McCaw, "Unified Shapes Checker—A Checking Tool for LSI," *ACM/IEEE Sixteenth Design Automation Conference Proceedings*, San Diego, CA, June 1979, pp. 81–87.
4. C. M. Sakkas, "Potential Distribution and Multi-Terminal DC Resistance Computations for LSI Technology," *IBM J. Res. Develop.* **23,** 640–651 (November 1979).
5. S. Newberry and P. Russell, "A Programable Checking Tool for LSI," *European Conference on Electronic Design Automation Proceedings*, Brighton, UK, September 1981.
6. D. Lambert, "Graphics Language/One—IBM Corporate-Wide Physical Design Data Format," *ACM/IEEE Eighteenth Design Automation Conference Proceedings*, Nashville, TN, June 1981, pp. 713–719.

Received January 6, 1984; revised April 30, 1984

A. M. Barone *IBM General Technology Division, East Fishkill facility, Hopewell Junction, New York 12533.* Mr. Barone is a senior engineer in the Engineering Design System organization of IBM located in East Fishkill. He is the recipient of divisional and corporate awards for the architecture and design of the Interactive Graphic System. He received the B.S. from City College of New York in 1965 and the M.S. from Syracuse University, New York, in 1972, both in electrical engineering.

J. K. Morrell *IBM General Technology Division, East Fishkill facility, Hopewell Junction, New York 12533.* Mr. Morrell is a senior engineer currently in the Engineering Design System organization of IBM located in East Fishkill. He has been primarily involved in the architecture and development of interactive design systems. Mr. Morrell received the B.S. in electrical engineering from Stevens Institute of Technology, Hoboken, New Jersey, in 1971.