CONCEPTUAL DESIGN ON A MICROCOMPUTER

by

Charles F. Morgan
Associate Professor
Department of Architecture
University of Florida

with

Martin Gundersen
David S. Moore
Tony Ambrose
John F. Alexander

October, 1986
Introduction:

As most computer users in the profession of Architecture recognize, computer usage in the profession, while increasingly widespread, is limited in scope. Architects may use the computer for word processing, estimating, office management and drawing production, but for the work most central to their profession, building design, the computer is used very little. While computer software and hardware vendors try to maintain the utility of their wares in this area, their misunderstanding of the differences among engineering, CAD/CAM, and Architecture shows in the inappropriateness of most systems for conceptual architectural design. While some software and systems address basic architectural design, most are incapable of the transition to design development, with its need for more symbolic information.

Educators in Architecture are quite a bit more clear in their understanding of the differences between professions and the need for different forms of the same information; they understand that the difference is conceptual in nature. Those educators who attempt to use computers further recognize that currently available software and hardware very seldom can be used at a conceptual level in architecture, especially since there is a need to translate that information from abstract diagrammatic forms to literal visualization forms to symbolic "production" forms. Only at relatively great cost can conceptual issues in
architectural design be addressed with computers, so most offices and educational institutions settle for production-oriented software and hardware in their initial computer purchases. The best of this software and hardware is capable of addressing design issues somewhat beyond conceptual design in the typical building design process, often called "design development." Several educational institutions, but very few offices, are pursuing conceptual architectural design on computers, usually with large research budgets and a cadre of very dedicated architectural faculty and students who happen to have extensive computer expertise.

It is clear that a need exists for less expensive software and hardware that can be used for conceptual design. Educational institutions, of course, spend the greatest portion of their time teaching conceptual design, but even offices must spend a significant fraction of their resources in generating the basic building solutions for which they will produce production documents.

Of course, it is rather difficult to come up with a definition of conceptual design with which most architects and/or architectural educators would agree. The process of architectural design is iterative, with detail decisions affecting and forcing the rethinking of previous general decisions. Conceptual decisions, while generally made at the beginning of the process, are still subject to modification and/or refinement. However, while a strict definition is difficult, certain ideas are common to most accepted notions about conceptual design. Such ideas include: definition of user needs (both explicit and implicit), the evocative/emotive nature of the design, appropriate ordering
systems that carry out the design concept, efficiency in structural concept, massing of the building design and the relationship between positive and negative space, among many others. Any conceptual software/hardware system must aid the designer in making basic decisions in these areas if it is to have utility in conceptual design. If these decisions are made apart from the computer, before the computer is involved in the design, then the computer system cannot be called a conceptual design tool.

Conceptual Design Education and Computers:

While the specific coursework and curriculum form differs between the many architectural education programs, all spend a great deal of time teaching conceptual design. Moreover, all programs address ideas of "ordering" or "structure" within conceptual design, as one of the bases for making further design decisions. Within many programs, the idea of the "grid" is central to the teaching of architectural ordering systems. The word grid implies to most people the ideas of regularity, formality, and repetition, among others. Most students, when first introduced to the concept, begin their investigations with very mundane, rectangular and rigid experiments. Their instructors spend a great deal of time "forcing" the students to expand their explorations into more complex, yet still ordered solutions.

On the other hand, the very regularity of grids implies that computers might be very useful in their manipulation. Until recently, however, the graphic shortcomings of most computers has made it difficult to use them in the education of graphically-
oriented individuals, even within those realms for which the computer has great potency. Again, the cost of workstations with suitable graphic capabilities has been beyond the budget of most educational institutions, especially given the number of students to whom conceptual design is taught. With the appearance of powerful microcomputers with substantial graphic capabilities, a tie between grids, computers, and student designers can finally be made, as can other issues intrinsic to conceptual design.

There are problems, of course. As mentioned, beginning students tend toward solutions that are either completely unordered or ordered with such rigidity as to be worthless. Many students will have very limited computer skills, so using a computer program for investigating grids may be more trouble than the results merit. Likewise, students will have developed graphic skills of their own, and the computer will have to have both the graphic quality and the speed to justify the student rejecting his own skills. Finally, many ordering systems or grids are not so easily manipulated with a computer. For example, a quadrille grid (made up of aligned squares) or proportional grid (aligned rectangles), or some types of rhythmical grids can be easily set up in programmed matrices; non-rectilinear grids and others (e.g. the "ken" grid) are very difficult.

![Figure 1: Four Grid Types](image)

93
To be sure, grids are not used alone. In beginning design courses they are most often used first in the composition of figure-ground studies by students, involving relatively simple diagrams set within an ordering system. They might also be combined with color and/or figure-ground reversal studies. Any of these exercises are very labor intensive, involving a great deal of time in the presentation of the solutions, either with cut-out pieces of paper, marking pens, or a multitude of other media. Also, students will be studying and investigating a great number of non-formal issues (culture, environment and behavior, etc.) implicit in conceptual design.

GRIDIT - Conceptual Design on a Microcomputer:

Over the last few years, the University of Florida has had a contract with the IBM Corporation through its Academic Information Systems Division. As part of this contract, two professors in the College of Architecture have been funded and supplied with equipment to investigate the possibilities for teaching a set of introductory and conceptual design issues to beginning architecture students through the use of computers. In particular, IBM is interested in the application of its high-resolution, high-color Professional Graphics Display and Adapter in a field that demands advanced graphic capabilities of its professionals. The contract calls for two educational "deliverables," the first being a two-dimensional conceptual design tool, the second a three-dimensional conceptual design tool. At this point, the first of the packages, called GRIDIT, has been completed and is
being tested in beginning design classes at the University of Florida. The following discussion will address the instructional concept, classroom use, and other issues related to GRIDIT. The second package, MODLIT, is still in production.

The first courses in architecture programs generally take one of two forms: In the first, students are trained in basic graphic techniques, and begin formal design education in later courses. In the second, used at the University of Florida, formal design education is taught from the very beginning, at the same time as basic graphic techniques. Students in these courses are beginners in many ways. Many have not had any introduction to computers and/or keyboard skills. They have not had college-level practice at trying to absorb concepts simply through reading and lectures. On the other hand, they have not developed habits, especially graphic habits, that may bias their attitude toward computer tools.

Very early in the first course, students are presented with information about architectural ordering systems, and quickly begin exercises involving the generation and manipulation of grids and figure-ground diagrams. A great deal of time is spent in the production of the diagrams, with a variety of media, and there is a concentration on the quality of the production. The investigations must not only be thorough; they must also be presented with strong graphic quality. Obviously, many students have a hard time performing well in both areas.

GRIDIT is intended to address many of these issues: It is intended for a beginning design student who has few, if any, key-
board and computer skills. It provides graphics of sufficiently high quality to satisfy both the student and the instructor. It addresses issues that are of prime concern to beginning design students in ways that are instructive and which save time so that more thorough investigations can be pursued. On the other hand, GRIDIT is enjoyable and useful, even for more advanced students working out conceptual designs. While it does not have the ability to address complex grid formats, it addresses adequately those of most common usage in architecture. Also, it allows the student to bring non-formal conceptual design issues into his work, for it does not constrain the content of his diagrams.

The GRIDIT program is organized into several packages, each with a different instructional intent. The first package is a simple tutorial describing the variety of grid types, not all of which can be manipulated in the program. A sample follows:

![Figure 2: A Typical Grid Introduction Screen](image-url)
The second GRIDIT package allows the student to create one of three types of grids, a simple quadrille grid, a proportional rectangular grid, or a rectangular grid based on rhythms entered by the student. The student may return to this package at any time, even after creating a diagram; he may then see the changes that occur in the diagram as a result of structural manipulation.

The third package is used to create figure-ground diagrams, with a number of constraints. The student is limited to certain shaped objects in his diagram (though an irregular polygon of up to fifty sides can be created), and the placement of the objects is constrained by the grid that forms the background of the diagram. The program also limits the student to sixteen colors, imposed by the specific programmatic approach taken in GRIDIT (the Display is capable of 256 colors at a time, out of a palette of 4096). Figure 3 is a typical diagram with the drawing menu:

Figure 3: Typical Diagram with Drawing Menu
The final package is really more a set of editing capabilities in GRIDIT. After creating the initial grid and a figure-ground diagram, the student may manipulate the grid which orders the diagram, and so modify the formal quality of the diagram. He may also investigate the figure-ground relationships explicitly by going into a two-color mode that emphasizes the relationships between "positive" and "negative" spaces. Figure 4 shows the same diagram after the ordering grid has been modified. On the next page, Figure 5 shows the diagram with the figure-ground relationship reversed.

![Diagram with Modified Grid](image)

Figure 4: Diagram with Modified Grid

The student may use any of the above packages repeatedly. That is, he may continue to change his diagram by adding or deleting objects, and he may continue to manipulate or redefine the grid on which the diagram lies, until he is satisfied with
the overall solution. He may also save his solution to a diskette file, if he is constrained by time, or if he is required to submit the solution to his instructor.

Figure 5: Diagram with Figure-Ground Reversed

Ideally, the use of this program would occur in the design studio, so that solutions and their processes could be shared, and so that instructors could provide feedback useful to all students. This situation would require, of course, a reasonable number of workstations whose cost is non-trivial (the Professional Graphics Display adds approximately $4000 to the cost of each workstation). At the University of Florida we anticipate equipping a sixteen-station computer laboratory with the appropriate equipment, allowing the students to be "cycled" through GRIDIT and other software appropriate to that level of student. Even this modest plan will severely tax our program's resources.
The GRIDIT package was completed at the end of Spring Semester, 1986. It was demonstrated at that time to a number of faculty and graduate students. The faculty were particularly interested in the program, especially those assigned to teach the introductory design courses in the Fall Semester. The interested faculty included several with very low levels of computer literacy and one faculty member who has, up to this point, been very negative about the use of computers in design studios. Both the ease of use of GRIDIT, and its ability to be a real help in the design process were repeatedly mentioned by the faculty as strengths of the program.

In order to better prepare for the use of GRIDIT this fall, GRIDIT was tested on a few students this summer. The students were high school juniors and seniors taking a course in "Design Exploration," but their level of computer and design expertise seems to mirror that of the freshman college students who enroll in the program. Because of the limited number of workstations, only those students showing an interest were allowed to work with the computer, though all were introduced to GRIDIT, as well as AUTOCAD and MegaCADD. The limitation of workstations, as well as time constraints imposed by a three-week program, resulted in a poor test of the software. For the most part, only those students with previous computer experience worked with any of the software. While they enjoyed playing with GRIDIT, their main interest seemed to be in showing off their previously-acquired abilities with AUTOCAD.
If we are able to purchase a few more workstations, we anticipate that a much better test of the package will occur in the fall, when it will be possible to formalize assignments for students to perform with GRIDIT. Likewise, a greater proportion of those students should understand the conceptual design value of GRIDIT and not be swayed by the raw drawing capabilities of AUTOCAD and MegaCADD.

In any case, there are several minor weaknesses with the GRIDIT package that have been identified during informal testing, but none that greatly affect its utility in class: First, GRIDIT currently does not provide any possibility for hard-copy output. It is unlikely that the situation of limited resources will ever allow hard-copy equipment that is of adequate color quality. In any case, the ability to save diagrams on diskette provides an alternate solution for turning work into an instructor.

Second, the limitations in drawing possibilities are bothersome to some users. The main advantage of GRIDIT, its simplicity of use, is probably its main disadvantage. Adding additional drawing capabilities, while useful to more advanced students, may do a disservice to those students most in need of a more simple, limited tool. Likewise, the editing capabilities could be expanded, also adding to the overall complexity of the package. We will continue to consider alternatives in these areas.

Third, the solely keyboard input, while simple, is bothersome after a relatively short time. We hope to implement "joystick" input in the near future.
Fourth, the color limitations are not consistent with the capabilities of the expensive display monitor. Early programmatic decisions affected this capability; given time we hope to implement full 256-color capability. Alternatively, we plan to create a companion package that students can use for color investigation. Because of the 4096-color palette, such a package could be very exciting.

Fifth, it is clear that GRIDIT has at least one instructional weakness in that it may leave the user with the impression that grids need to be rectilinear. Even though the beginning tutorial makes reference to other grid types, GRIDIT is limited in its ability to manipulate the many varieties of grids. The modular organization of GRIDIT may allow further expansion in the area, if the necessary grid data structures can be integrated with the rest of the system.

In summary, the current weaknesses in GRIDIT are not great. The program has great utility for beginning design students in allowing them to investigate a basic ordering system in conceptual design, the grid. While it does not allow full flexibility in drawing, it shows students some of the constraints that an ordering system may impose on any design. It is also a useful tool for student investigations into figure-ground relationships within ordered systems. Finally, through all of these activities, GRIDIT helps teach design, but does not control it. As it must be, the student, not the computer, is the final arbiter of good design.