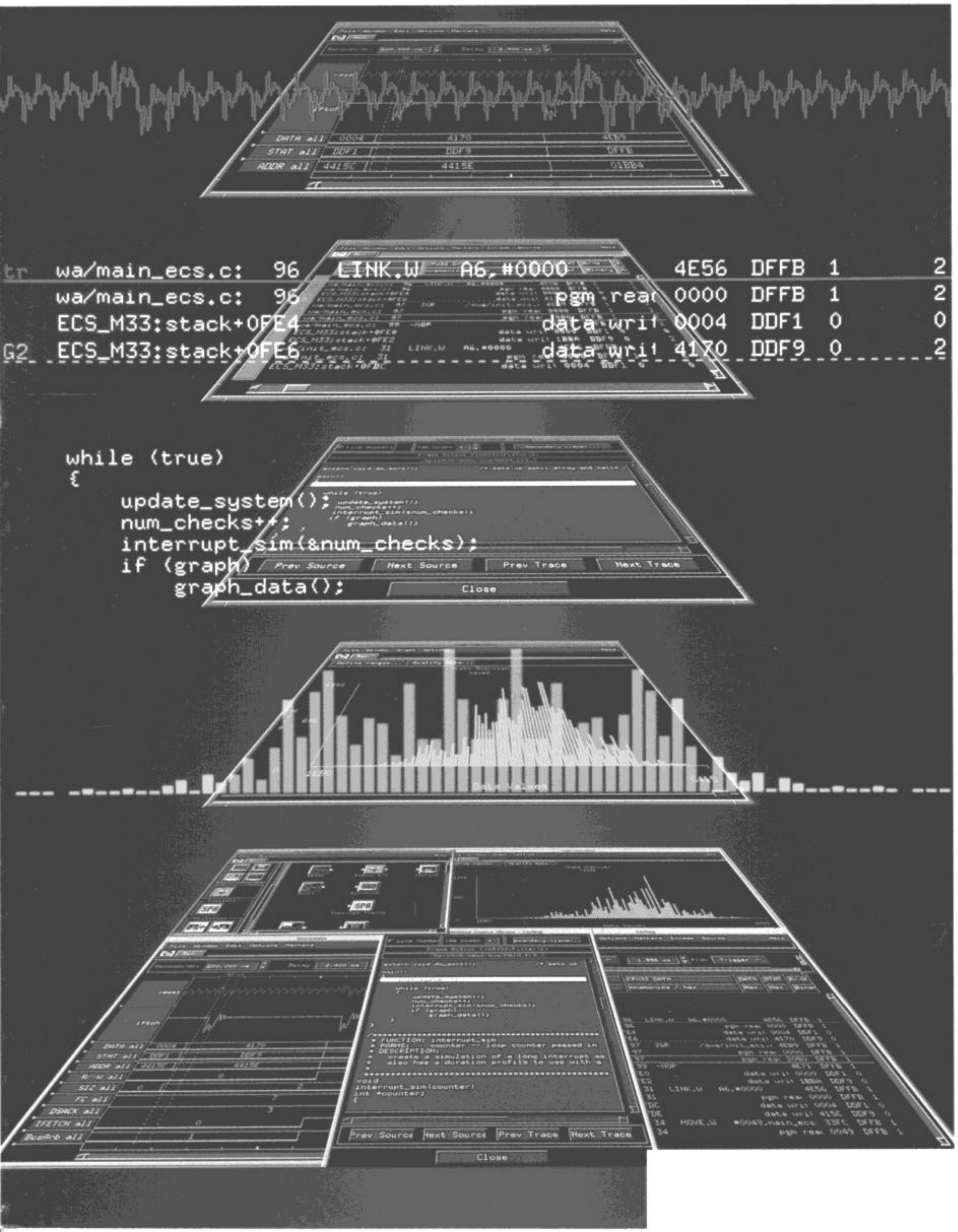
HEWLETT-PACKARD JOURNAL

June 1996



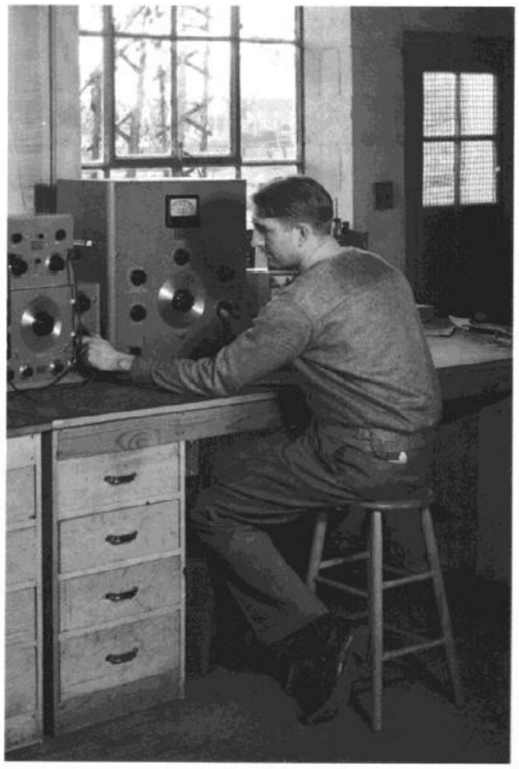
```
tr  wa/main_ecs.c:   96  LINK.W   A6,#0000        4E56  DFFB  1    2
    wa/main_ecs.c:   96              pgm rea(  0000  DFFB  1    2
    ECS_M33:stack+0FE4              data wri!  0004  DDF1  0    0
G2  ECS_M33:stack+0FE6              data wri!  4170  DDF9  0    2
```

```
while (true)
{
    update_system();
    num_checks++;
    interrupt_sim(&num_checks);
    if (graph)
        graph_data();
```

HEWLETT PACKARD

In Memoriam

Dave Packard
1912-1996



Dave Packard's idea for the HP Journal originated with a publication called the Experimenter, published by the General Radio Company. "I remember drooling over the Experimenter when I was in high school," Dave told Karen Lewis, HP's archivist and historian. Featuring an article entitled "A New Amplifier for Milli-Microsecond Pulses," the HP Journal made its debut in September, 1949.

Packard believed the Journal's role was to describe the technology used in the development of important HP products, and that it should be a vehicle for sharing such information with the larger engineering community. He also felt the Journal should recognize the contributions of individual HP engineers to the company and the electronics industry. Forty-seven years later, on the occasion of Dave Packard's passing, we at the Journal are still dedicated to those goals.

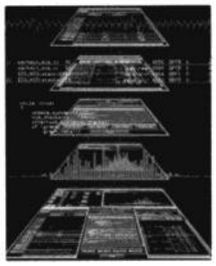
HEWLETT-PACKARD JOURNAL

June 1996 Volume 47 • Number 3

Articles

Executive Editor, Steve Beitler • **Managing Editor,** Charles L. Leath • **Senior Editor,** Richard P. Dolan • **Assistant Editor,** Robin Everest
Publication Production Manager, Susan E. Wright • **Graphic Design Support,** Renée D. Pighini • **Typography/Layout/Illustration,** John Nicoara

In this Issue



The most unpredictable and time-consuming part of developing a complex digital system is the system integration phase. During system integration, the various hardware and software subsystems finally come together for the first time. Although the interdependencies between the various subsystems have been simulated, emulated, and unit tested before system integration, there are always surprises. These surprises, or "hard problems," (page 6) dramatically affect the time-to-market goals for the product.

To find the root causes for these hard problems, engineers need to see what is going on in several subsystems (e.g., operating system, memory, I/O, etc.) at once. The article on page 15 describes a tool that assists engineers in the search for root causes. The tool is called the HP 16505A prototype analyzer. The HP 16505A is an X11/Motif application that runs on an HP 9000 Model 712 workstation connected to an HP 16500 logic analyzer. This configuration provides a graphical user interface (GUI) that enables users to create a measurement setup using icons that represent HP 16500 hardware modules and HP 16505A software modules. The software modules display or list the time-correlated data from the instruments in a variety of formats. To deal with the variety of data types from the instruments and the data uniformity requirements of the analysis and display modules, data is normalized immediately after acquisition and made available to all the output modules in a standard format (page 30). The selection of the encapsulated measurement server architecture of the HP 16505A (page 22) was based on feedback from an analysis of customer needs and requirements.

Allowing targeted users to participate in the design of a product is the best way to focus product goals and hopefully produce a successful product. The HP 38G graphing calculator (page 45) is a calculator designed for precalculus students and their teachers. In addition to the typical HP design team, an Educational Advisory Committee consisting of high school, community college, and university teachers was formed to help in designing this product. The HP 38G is built on the same software platform as the HP 48G graphing calculator, so it has many of the capabilities of the HP 48G but with a simpler user interface and a capability called *aplets*. An aplet (page 59) is a small application that focuses on a particular problem. Thus, teachers can keep students focused on a particular scientific or mathematical concept (e.g., polygons) by creating and downloading a set of aplets for students to work on.

HP OmniBook computer users are used to the idea of having a small mobile computer with some of the same capabilities and applications as their desktop machines. However, with features such as color displays, larger hard drives, and faster processors appearing in the HP OmniBook family of notebook computers, the expectation is to have a full-featured notebook computer with the same capabilities as desktop computers (of course not with the same performance as the high-performance desktop models). The article on page 38 describes the HP OmniBook 5000, which is a full-featured notebook computer based on the Pentium® processor. This product is designed to satisfy the demand for a notebook computer with complete desktop computer capabilities.

Today's mobile paging systems allow people to keep in 24-hour contact. One group of people who especially need this kind of service are physicians caring for critically ill patients. They need access to all the clinical patient information (graphical and textual) no matter where they are. The HP PalmVue system (page 64) fulfills this need by allowing the transmission of clinical patient information via conventional alphanumeric paging systems. The system integrates computer networks, palmtop computers, paging systems, and physicians who use HP monitoring systems and cardiographs to deliver high-quality patient data to mobile physicians.

HP divisions are always trying to find tools and processes to increase productivity and improve the quality of products. The last two articles in this issue describe efforts aimed at these quality and productivity goals. The first article (page 70) describes a tool that helps system administrators deal with installing and maintaining applications in an environment where there are hundreds of workstations and servers. The second article (page 76) describes software translators that facilitate the development of programs consisting of high-level C-language modules and low-level assembly-language modules. Such dual-language programs are common when old software modules are reused.

C.L. Leath
Managing Editor

Pentium is a U.S. trademark of Intel Corporation.

OSF, Motif, and Open Software Foundation are trademarks of the Open Software Foundation in the U.S.A. and other countries.

A Graphing Calculator for Mathematics and Science Classes

The HP 38G calculator allows teachers to direct students and keep them focused while they explore mathematical and scientific concepts. It features aplets, which are small applications that focus on a particular area of the curriculum and can be easily distributed from the teacher's calculator to the students'.

by Ted W. Beers, Diana K. Byrne, James A. Donnelly, Robert W. Jones, and Feng Yuan

The HP 38G calculator is a graphing calculator for students and teachers in mathematics and science classes. It features aplets, which are small applications that focus on a particular area of the curriculum and can be easily distributed from calculator to calculator. This allows the teacher to send an electronic story problem to each student in the class.

The HP 38G is built on the same software platform as the HP 48G family of graphing calculators,[1] but has a simpler user interface and feature set. Equations are entered using algebraic format rather than the reverse Polish notation (RPN) found in most HP calculators. The features of the HP 38G include:

- Graphical user interface
- Function, polar, parametric, stairstep, cobweb, histogram, scatter, and box and whisker plots
- Side-by-side split screen
- Tables
- Unlimited, scrollable history stack
- Symbolic equations
- HP Solve numeric root finder
- EquationWriter display
- Statistics functions
- Matrix operations
- User programming.

The hardware platform of the HP 38G is very similar to that of the HP 48G: they both have 32K bytes of RAM, 512K bytes of ROM, the same CPU and the same display (131 by 64 pixels). They both have a two-way infrared link for sending information to a printer and for transferring information between two calculators, and an RS-232 link for calculator-to-computer communications. An accessory that allows the calculator screen to be displayed with an overhead projector works with both the HP 48G and the HP 38G, but the HP 38G cable connector has been modified so that the overhead display accessory works with every HP 38G; no special handset is required.

The HP 48 case was redesigned to include a sliding plastic cover to make the HP 38G more durable for use by younger students. Also, two keys were removed to give visual emphasis to the navigation keys and to make the keyboard look less complicated.

Designed for and with Teachers

We set out to design a graphing calculator for precalculus students and teachers. To help us do this, we formed an Education Advisory Committee consisting of eight high school, community college, and university teachers. We met with the teachers every few months, and between meetings we kept in touch by email.

The teachers told us that they want to allow students to explore mathematical and scientific concepts, and at the same time they need to direct the students and keep them focused. In our first meeting with the teachers, we compared this to the idea of a child's sandbox: the child is given toys for playing and exploration, but within a protected, specialized environment. Thus, one of our main goals with the calculator software design was to encourage exploration by limiting choices. This led us to the concept of *aplets*: an aplet is a small application that focuses on a particular problem.

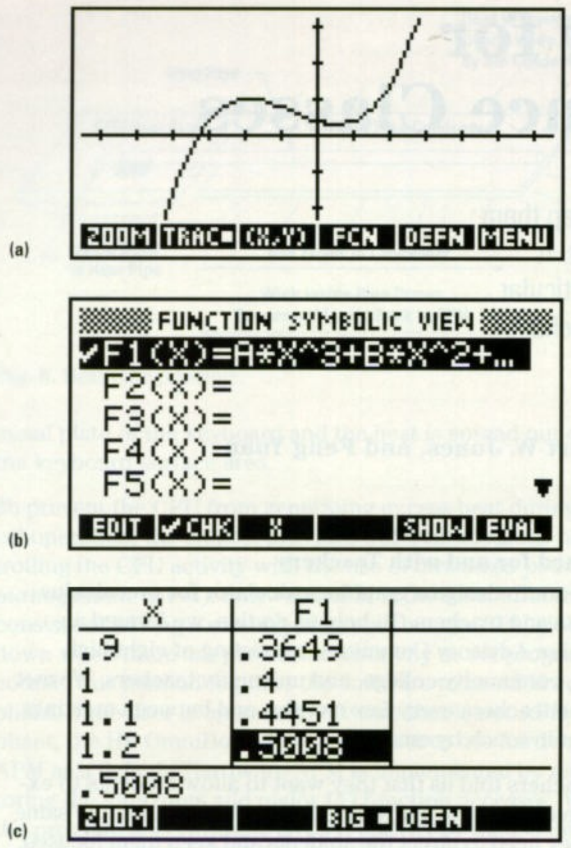
HP 38G Aplets and Views

We based the design of aplets on the National Council of Teachers of Mathematics "three views": graphic, symbolic, and numeric. Each problem can be explored using these different representations. For example, a mathematical function can be expressed as a graph (Fig. 1a), in symbolic form (Fig. 1b), or using a table of numbers (Fig. 1c).

Aplets can be created by teachers (either directly on the HP 38G or with the assistance of a computer) and then "beamed" to the students' calculators using infrared transfer. This way, a whole classroom full of students can have their calculators programmed identically at the beginning of a lesson. Then, the students can explore within the aplet on their own calculators.

Each aplet packages the formulas, settings, and other information associated with a particular problem. If the user wants to switch from one aplet to another, this can be done without disturbing any individual aplet's settings, since they are compartmentalized.

Several aplets are built into the HP 38G. When the calculator is first turned on, these built-in aplets are empty. The user



✓F1(X)=A*X^3+B*X^2+...
F2(X)=
F3(X)=
F4(X)=
F5(X)=

X	F1
.9	.3649
1	.4
1.1	.4451
1.2	.5008

.5008

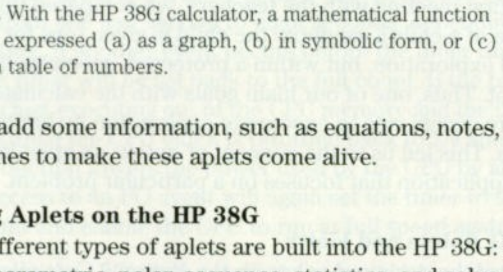
Fig. 1. With the HP 38G calculator, a mathematical function can be expressed (a) as a graph, (b) in symbolic form, or (c) using a table of numbers.

must add some information, such as equations, notes, or sketches to make these aplets come alive.

Using Aplets on the HP 38G

Six different types of aplets are built into the HP 38G: function, parametric, polar, sequence, statistics, and solve. Typically, a teacher will start with one of these aplet types, then customize it by adding particular functions that define a certain problem, together with settings, pictures, and text directions.

To start using a particular aplet that is already in the calculator, the student chooses it from the HP 38G library by pressing the **LIB** key. Continuing to explore the problem, the student may want to view the problem in different ways, and can do this by pressing the different view keys. **PLOT**, **SYMB**, and **NUM** display the graphic view, symbolic view, and numeric view, respectively. Additional views, such as split-screen views (Fig. 2), can be found by pressing the **VIEWS** key.



X	F1
-.60938	.634452
-.40625	.548178
-.20313	.468351
0	.4
.203125	.348153
.40625	.317838
.609375	.314083

Fig. 2. The plot-table view is a typical split-screen view.

The student or teacher can customize the graphical, numeric, and symbolic views for a specific problem by using the setup screens. It is also possible to annotate the problem by typing some text into the note view or by adding a sketch to the sketch view.

The teacher and student can easily generate custom aplets that present new examples based on the built-in aplet types. An aplet is created by working with an aplet type and adding all of the customizing information that relates to a particular problem (see article, page 59).

Main Components

The keyboard (Fig. 3) reflects the seven main components of the HP 38G's functionality (from top to bottom on the keyboard):

- Softkeys for accessing custom behavior defined by softkey labels along the bottom of the display
- View keys for moving among the possible representations of aplet data
- Library key for selecting the current topic of exploration with aplets
- Arrow keys for screen and menu navigation
- Home calculator keys for numeric entry and basic calculator operations
- Alpha keys for access to alphabetic characters
- Editing environments for creating, editing, and managing objects such as programs, matrices, lists, and notes.

Four of the key groupings are related as follows: first, a topic is chosen with the library key. Then, a way of looking

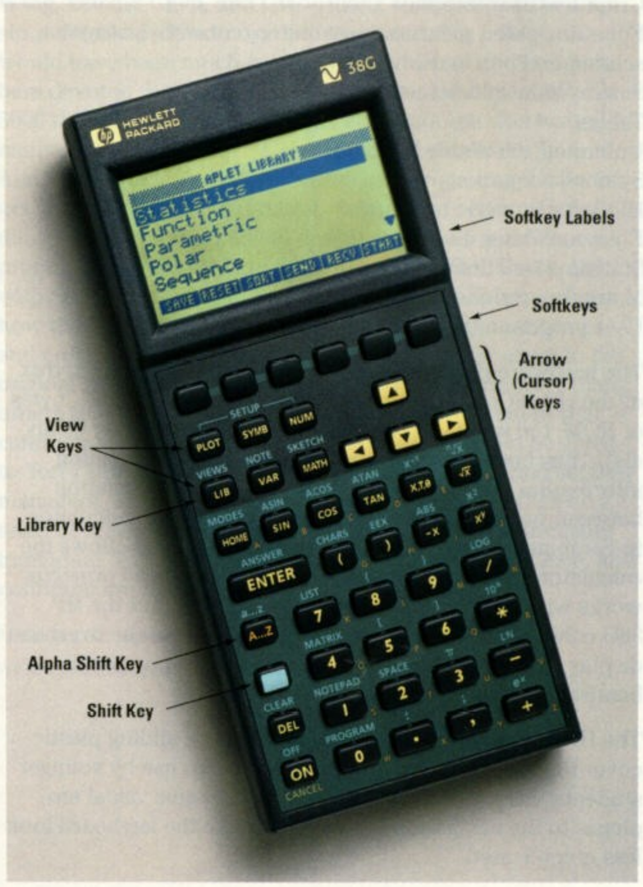


Fig. 3. HP 38G calculator.

at the problem is chosen with the view selection keys. Finally, the peculiarities of the problem view are manipulated with the softkeys and the arrow keys.

Softkeys

The softkeys are the top row of unlabeled keys, which are associated with the dynamic menu labels displayed along the bottom of the screen. They give access to context-specific custom behavior. Unlike the HP 48G, there is no "next-row" key since all HP 38G softkey sets are limited to six items for the sake of simplicity.

View Keys

The view keys provide one-key access to the three National Council of Teachers of Mathematics representations of aplet data: graphical, numerical (tabular), and symbolic, plus annotation and sketch views for documenting a topic. Keys for setting up view parameters and managing split-screen views are also included.

- The plot view gives a conventional mathematical graphical representation. In most cases it looks like some flavor of plot output in the HP 48G.
- The numeric view is generally a table of sampled values. It is a derived form of the mathematical object (except for statistics, where it is the defining view).
- The symbolic view is generally an expression representing the mathematical object of interest. The variables are defined by the aplet. This view is the defining form of the mathematical object (except for statistics, where it is derived).
- The note view is a mini word processor that allows the user to create, edit, and view a text document associated with the aplet.
- The sketch view is a set of standard-sized GROBs (graphic objects in HP 48G terminology) that explain the "story" of the aplet. The user can create, edit, view, and animate pictures. Editing capabilities include lines, boxes, circles, text labels, and Etch-a-Sketch-style drawing.

Moving from view to view is the same as task switching, which means the user can always go on to the next task, but there is no concept of going back to somewhere, since tasks are not being nested.

The **LIB** key invokes the library, which is the aplet selection and management environment, in which different aplets (including those that are not built-in) can be started, exchanged with others, and created by saving the state.

The **VAR** key provides access to variables and the **MATH** key provides access to scientific functions and other operations and commands. These variables and operations are available whenever the user is using an edit line.

Invoking the **VAR** or **MATH** menu starts a subtask, which must be completed or cancelled, at which point the calculator is back where it was when the subtask was started.

The Library Environment

This environment gives high-level access to aplets. The user can select an aplet and manage the current collection of aplets. From within the library, the user can take a snapshot of a built-in aplet, giving it a name and a directory of its own. This is how aplets are generated for dissemination and how users show their work. Also, the user can import or export aplets from the library to another HP 38G or to a computer.

Arrow Keys

The arrow keys are used for all direction-oriented operations such as tracing a function plot, moving among fields in an input form, and selecting commands from a pop-up menu.

The shift key can be used as a modifier for the arrow keys that signals motion "all the way" in the direction indicated.

Home Calculator Keys

The home calculator environment gives a familiar tool with a nice graphical interface. The user invokes it by pressing the **HOME** key. Inputs are displayed on the left side of a line, with the results displayed on the right side of the next line.

The calculator keys are used to type numbers and to access basic scientific calculator functions. The **ENTER** key is used to terminate data entry, to select operations from menus, and in general, to make things happen. Some mathematical operations are available on the keyboard and other operations and commands are available through the **MATH** pop-up menu.

The **ANSWER** shifted key gives access to the variable called Ans, which always contains the last result.

Alpha Keys

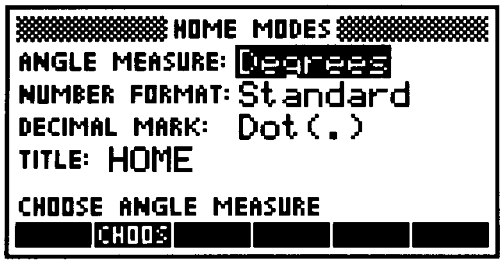
The alpha shift key, **A...Z**, provides access to the alphabetic characters, which are labeled on the keyboard overlay. Pressing the **CHARS** shifted key invokes the character browser, which provides access to characters that are not on the keyboard.

Unlike the HP 48G, there is no alpha lock key to confuse the user. The user can either press and release the alpha shift key before pressing each alpha character key, or hold the alpha shift key down while pressing as many alpha character keys as desired. However, an alpha lock toggle softkey is provided in some editing environments.

Editing Environments

The HP 38G has specialized environments for managing programs, matrices, lists, and notes. When the user invokes one of the editing environments, a scrolling choose list of all the existing objects of that type appears, together with a softkey menu. From this environment, objects can be transferred between the HP 38G and a computer or another HP 38G.

- The program environment provides tools for creating, editing, storing, sending, receiving, and running programs. Variables can be accessed through the **VAR** pop-up menu. Mathematical operations can be accessed from the keyboard or through the **MATH** pop-up menu, and programming commands can be accessed through the commands section of the **MATH** pop-up menu.
- The matrix environment provides a two-dimensional matrix editor for creating, editing, viewing, sending, and receiving matrices.
- The list environment provides a list editor for creating, editing, viewing, sending, and receiving lists.
- The notepad environment provides an environment for creating, editing, viewing, saving, sending, and receiving



```
HOME MODES
ANGLE MEASURE: Degrees
NUMBER FORMAT: Standard
DECIMAL MARK:  Dot(.)
TITLE: HOME

CHOOSE ANGLE MEASURE
     CHOOS
```

Fig. 4. A typical input form.

text documents. The tools for editing notes in the notepad environment are identical to the editor for the note view. The documents created in the notepad environment are not bundled with an aplet as they are in the note view. The notepad environment can be used for tasks such as creating, storing, and viewing lists or notes.

User Interface

One of our goals for the HP 38G project was to leverage software from the HP 48G/GX calculator platform. For the HP 48G/GX we developed two primary general-purpose user interface tools: input forms and choose boxes.[1]

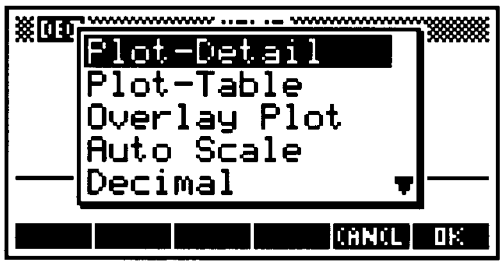
Input forms and choose boxes are interactive environments that are used to gather user input for a task. Input forms (see Fig. 4) are flexible, screen-sized dialog boxes similar to those in other graphical user interfaces. The appearance and use of input form fields resemble spreadsheet programs. Choose boxes (see Fig. 5) are either pop-up or screen-sized boxes optimized for selecting or working with one or more items from an arbitrarily long list. Input forms and choose boxes, along with some other user interface constructs, contribute to the improved ease of use that distinguishes the HP 48G/GX from its predecessor, the HP 48S/SX.

In the HP 38G, we found that input forms and choose boxes were a good match for the needs of the utility environments and most of the nine views available for working with aplets.

Graphical User Interface Applications

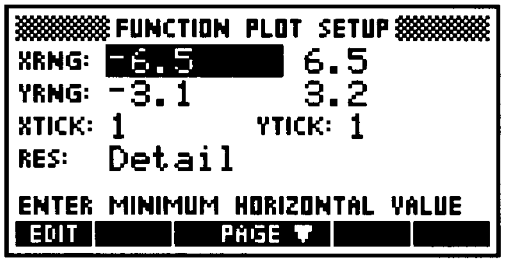
Input forms are used in most aplet views for entering a mix of settings and values. They're also used for gathering input to complete a task. For example:

- The function aplet's plot setup view is an input form in which modes and parameters are specified (see Fig. 6).
- The solve aplet's numeric view is an input form whose appearance varies according to the equation to be solved (see Fig. 7). The flexible layout and labeling of input form fields



```
Plot-Detail
Plot-Table
Overlay Plot
Auto Scale
Decimal        ▼

                CANCL  OK
```

Fig. 5. A typical choose box.



```
FUNCTION PLOT SETUP
XRNG:  -6.5      6.5
YRNG:  -3.1      3.2
XTICK: 1     YTICK: 1
RES:   Detail

ENTER MINIMUM HORIZONTAL VALUE
EDIT        PAGE ▼
```

Fig. 6. The function aplet's plot setup view input form.

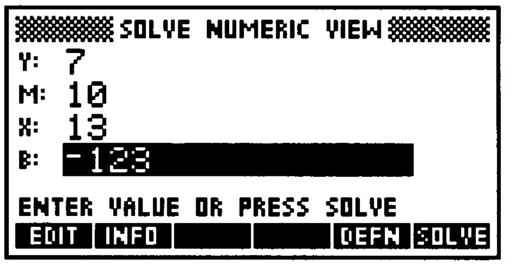
are employed to generate a custom input form each time the view is entered.

- When an aplet is to be saved, a simple input form is displayed to get the new aplet's name (see Fig. 8).
- The statistics symbolic view is a highly customized input form that scrolls to show more information than fits on one screen (see Fig. 9). Choose box elements, like the up and down arrows indicating more information is available off-screen, help suggest the operation of this hybrid view.

Choose boxes are used in a variety of views and utility environments wherever a list of related items needs to be managed. Here are a few examples of choose boxes in the HP 38G:

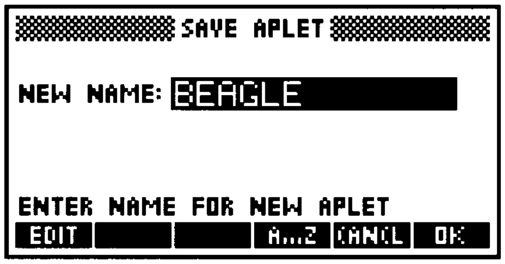
- The aplet library (see Fig. 10) is actually an elaborate choose box.
- Almost all symbolic views, such as the function aplet's symbolic view (see Fig. 11), are choose boxes.
- Choose boxes pop up within input forms like the **MODES** input form (see Fig. 12).
- The **VAR** and **MATH** menus (see Fig. 13) are two-column choose boxes that show categories of items plus the items in the selected category.

As these examples illustrate, the look and feel of HP 48G/GX input forms and choose boxes remain largely unchanged in



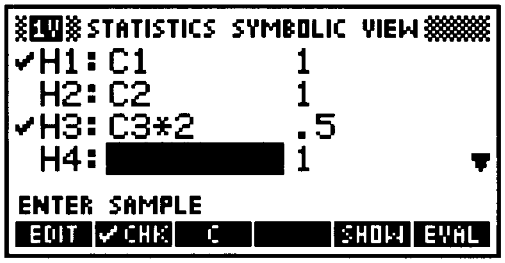
```
SOLVE NUMERIC VIEW
Y: 7
M: 10
X: 13
B: -123

ENTER VALUE OR PRESS SOLVE
EDIT INFO           DEFN SOLVE
```

Fig. 7. The solve aplet's numeric view, a dynamically generated input form.



```
SAVE APLET

NEW NAME: BEAGLE


ENTER NAME FOR NEW APLET
EDIT           A...Z CANCL OK
```

Fig. 8. The aplet library's save-aplet input form.



```
STATISTICS SYMBOLIC VIEW
✓H1: C1        1
 H2: C2        1
✓H3: C3*2      .5
 H4:           1
ENTER SAMPLE
EDIT ✓CHK C  SHOW EVAL
```

Fig. 9. The statistics aplet's symbolic view, a custom scrolling input form.

the HP 38G. However, substantial reengineering of the underpinnings of these tools was required to match other aspects of the HP 38G's use model, as discussed in the next section.

Topic Outer Loop

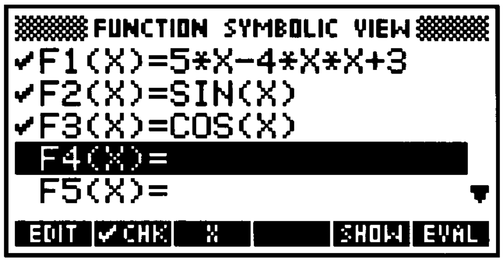
Many of the custom interfaces developed for the HP 48S/SX used an RPL-language tool we developed called the parameterized outer loop.[2] Parameterized outer loop applications depend on the parameterized outer loop for routine key and error handling and display management. The graphical user interface (GUI) elements introduced in the HP 48G/GX are also parameterized outer loop applications that automate routine matters of input entry, selection of options, and presentation of output.

In both the HP 48S/SX and the HP 48G/GX, the user interface was based on the notion of having a central environment (the user stack) from which other applications are launched and to which applications return when completed. All applications on these platforms, including parameterized outer loop applications like the GUI tools, are based on this "function call" model: they start, run for a while, then end, returning the flow of control to where they started. We call such applications *modal*.

New Model, New Tool

When we were investigating the use model for the HP 38G, it became apparent that the function call approach to application management would not suffice. The HP 38G is a tool for exploration, so we wanted to provide an environment that promotes wandering from one subject area to another, or in HP 38G terms, from one view or aplet to another.

To attain this goal we quickly determined that the modal nature of the parameterized outer loop and the applications based on it was too constraining, yet we weren't prepared to discard the wealth of useful tools and concepts we had built up from the parameterized outer loop foundation. Furthermore, we knew there were still plenty of times when the modal call-and-return interface would still apply, such as



```
FUNCTION SYMBOLIC VIEW
✓F1(X)=5*X-4*X*X+3
✓F2(X)=SIN(X)
✓F3(X)=COS(X)
 F4(X)=
 F5(X)=
EDIT ✓CHK X  SHOW EVAL
```

Fig. 11. The function aplet's symbolic view.

when pausing to get further input from the user before proceeding with a task. To accommodate all these needs, we developed the new *topic outer loop*.

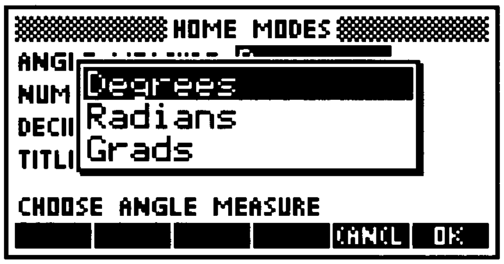
Topic Outer Loop Overview

Where parameterized outer loop applications are designed to preserve the environment from which they're launched and later restore that environment, topic outer loop topics are optimized for rapidly setting up and switching from one topic to the next. Except with regard to the home environment from which the topic outer loop is originally launched and to which it ultimately returns—after running many topics, perhaps—the topic outer loop doesn't preserve or restore a previous user interface since there is none to go back to.

The most obvious examples of topic outer loop topics in the HP 38G are aplets, but many other environments with similar behavior are also topic outer loop topics—for example, the aplet library and the user program catalog (see Fig. 14).

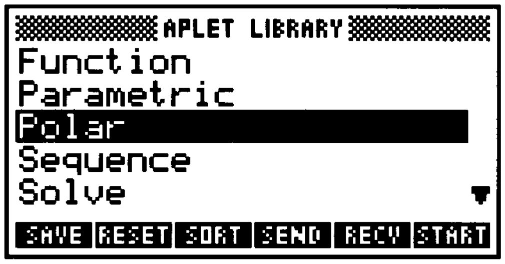
Like the parameterized outer loop on which it's based, the topic outer loop is launched from the calculator's system outer loop and temporarily redefines the current user interface until some exit condition is met. By design, the topic outer loop operates very similarly to the parameterized outer loop, but differs from the parameterized outer loop in two fundamental ways:

- To better support the standard two-tiered structure of HP 38G topics, the topic outer loop manages two nested user



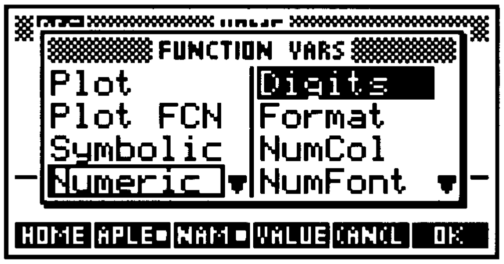
```
HOME MODES
ANGL
NUM  Degrees
DECI Radians
TITL Grads
CHOOSE ANGLE MEASURE
                 CANCL OK
```

Fig. 12. A pop-up choose box in the **MODES** input form.



```
APLET LIBRARY
Function
Parametric
Polar
Sequence
Solve
SAVE RESET SORT SEND RECV START
```

Fig. 10. The aplet library, an elaborate choose box.



```
FUNCTION VARS
Plot       Digits
Plot FCN   Format
Symbolic   NumCol
Numeric    NumFont
HOME APLET NAME VALUE CANCL OK
```

Fig. 13. The **VAR** menu, a two-column choose box.

interface levels. The parameterized outer loop manages just one.

- The topic outer loop fully supports organized and efficient switching from one topic to another. The parameterized outer loop is designed to shut down completely before launching another application.

The operation of the topic outer loop for starting a topic can be summarized as follows:

```
If a topic outer loop is already running {

  Evaluate the old view exit handler
  Evaluate the old topic exit handler
  Set the topic entry and exit handlers
  Evaluate the topic entry handler
  Set the view entry and exit handlers
  Evaluate the view entry handler
  Set the remainder of the view user interface

}

Else {

  Save the home user interface

  If error in {

    Set the topic entry and exit handlers
    Evaluate the topic entry handler
    Set the view entry and exit handlers
    Evaluate the view entry handler
    Set the remainder of the view user interface

    If error in {

      While not done with the topic outer loop {
        Evaluate the view display object
        Read a key and get its custom definition
        If error in
          Evaluate the key definition
        Then
          Evaluate the error handler object
        }

      Evaluate the view exit handler
      Evaluate the topic exit handler

      }
    Then {
      Evaluate the view exit handler
      Evaluate the topic exit handler
      Error

      }

    }

  Then {

    Restore the saved home user interface
    Error

    }

  Restore the saved home user interface

}
```

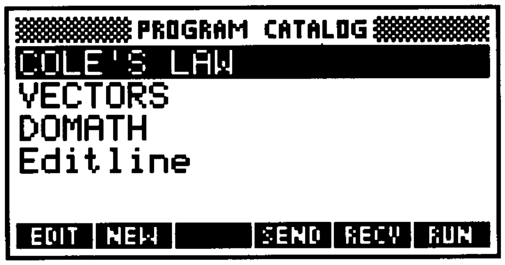

PROGRAM CATALOG
COLE'S LAW
VECTORS
DOMATH
Editline

Fig. 14. The user program catalog, a topic outer loop utility environment.

The code setting up the topic specifies the user interface and other environment elements unique to the topic, such as the topic entry handler and the view display object, when it runs the topic outer loop. This is how the behavior of the topic outer loop is customized. The topic outer loop is responsible for the key-display loop, low-level error handling, and juggling the topic and view entry and exit handlers and the saved home user interface.

When an event occurs that calls for running the topic outer loop, the topic outer loop may or may not already be running. As the first section of the topic outer loop overview illustrates, if a topic outer loop is already running, switching to a new topic is quick yet still gives the exiting topic an opportunity to shut down in an orderly fashion. Since it's common with the HP 38G to move from topic to topic without first returning home, this efficiency translates to faster performance.

Switching among views within the same topic is also common, and involves a similarly efficient set of operations:

```
Evaluate the old view exit handler
Set the view entry and exit handlers
Evaluate the view entry handler
Set the remainder of the view user interface
```

Reengineering the GUI Tools

Although very similar in specifications and use to the standard modal input form and choose box environments, versions of these environments based on the topic outer loop, which we call *modeless* environments, differ in the following ways:

- OK and cancel keys are nonfunctional.
- The default softkey set does not include OK and cancel keys.
- Task-switching keys are processed normally to allow task switching.
- The results returned by the input form or choose box engine always consist of the confirmed exit values. No flag indicating canceled or normal exit is returned.

To support modeless views and utility environments based on HP 48G/GX GUI tools, we adapted the input form and choose box engines to use the topic outer loop. However, modal input forms and choose boxes are also employed by the HP 38G. Rather than simply switch the engines from using the parameterized outer loop to using the topic outer loop, we modularized the components of the engines to enable the use of either. We then repackaged the modal versions of the engines to ensure backwards compatibility for existing code using them. To make modeless input form and choose box programming as straightforward as possible for programmers familiar with the modal versions, and yet still meet the requirements of the topic outer loop, we developed tools to translate traditional modal input form and choose box arguments and results to and from the specifications required for topic outer loop applications. This greatly simplified the reengineering of existing user interface code to make use of new modeless input forms and choose boxes. The process was largely mechanical, requiring only that the developer follow a few well-documented steps.

Aplets and Views

One of the key ideas of aplets is that they provide different ways of looking at a problem. For example, when exploring a story problem about speed and distance, the student can look at a symbolic expression, a table of numbers, a graph of the distance function, or even a diagram showing a tortoise and a hare. These different ways of looking at an aplet are called *views*. The topic outer loop manages the transitions between the views of an aplet. The views are implemented using the graphical user interface tools plus aplet data. The data associated with an aplet is encapsulated in a directory structure inherited from the HP 48G/GX.[1]

Aplet Structure

Associated with each aplet is a standard set of information. The topic outer loop uses this aplet information for aplet directory checking, topic switching, resetting, and so on. It's also used by the **VAR** menu and the **VIEWS** choose box. The standard information is:

- Topic ID
- Initial view
- Topic name
- Special views data
- Aplet-specific variables
- Topic entry procedure pointer
- Topic exit procedure pointer
- Topic reset procedure pointer
- Aplet directory checker procedure pointer.

An HP 38G aplet is a collection of aplet data and aplet views. Aplet data includes the aplet name, which is used in the library, real variables like Xmin and Xmax, which are set via the plot setup view, symbolic expressions, note text, and sketches. An aplet usually defines at least eight views: plot, symbolic, numeric, note, sketch, plot setup, symbolic setup, and numeric setup. The generic aplet contains items such as:

- Aplet name
- Aplet topic
- Attached library
- Plot view procedure pointer
- Symbolic view procedure pointer
- Numeric view procedure pointer
- Plot setup view procedure pointer
- Symbolic setup view procedure pointer
- Numeric setup view procedure pointer
- Note view procedure pointer
- Sketch view procedure pointer
- Xmin variable
- Ymin variable
- Other shared variables
- Aplet-specific variables
- List of symbolic expressions
- Array of independent values for numeric view
- List of strings for custom views
- Note text
- List of graphics objects for the sketch view.

For aplets built into the HP 38G, pointers like the plot view procedure pointer refer to code built into the 512K-byte ROM. New aplet types can include a RAM-based support library containing new view procedures. All aplets must contain a basic set of variables, but specific aplet types may implement additional optional variables.

View Structure

Each aplet view is managed by the topic outer loop, typically with the help of graphical user interface (GUI) tools like input forms. To customize its behavior, a view (or its GUI helper) specifies objects that are used while the view is visible. These include the view entry and exit handlers, the display handler, and the key handler, among others:

- Initialization procedure pointer
- Exit procedure pointer
- Display procedure pointer
- Key handler procedure pointer
- Non-view-specific-key-allowed flag
- Softkey menu description
- Error handler.

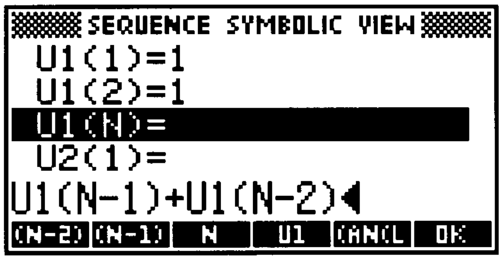
Symbolic View

Normally, the symbolic view is the defining view for an aplet. When the user starts an aplet, its symbolic view will be shown so the user can enter symbolic expressions or equations to be used in the aplet.

The symbolic view is the generalization of the HP 48G/GX EQ list. The EQ variable on the HP 48G is a list of unnamed symbolic expressions, which are plotted and traced together. The HP 38G breaks this into individual named symbolic expressions that have independent check marks. Expressions for a parametric function are further broken into real and imaginary parts. The expression for a sequence is broken into two initial terms and a recursive relation. This simplifies editing and selection of symbolic expressions. Giving expressions names in the symbolic view allows them to be reused in other expressions, home calculations, and programming.

Expressions entered in the symbolic view are checked for syntax errors and to a limited extent for semantic errors. Expressions defining a sequence are further classified and transformed into an internal form for cache-based iterative evaluation, saving both time and run-time RAM space. An EVAL menu key is provided in the symbolic view for constant expression evaluation, expression simplification, and function unfolding. Fig. 15 shows how the Fibonacci sequence can be defined and checked using ten keystrokes. (The EVAL menu key is shown when a command line is not active. It appears where OK appears in Fig. 15.)

The generic symbolic view is based on the choose box engine, which takes over display handling to maintain check marks on the left of the screen and takes over key mapping for dynamic menu changes and editing of expressions.


SEQUENCE SYMBOLIC VIEW

U1(1)=1

U1(2)=1

U1(N)=

U2(1)=

U1(N-1)+U1(N-2)

Fig. 15. Using the symbolic view of the sequence aplet, the Fibonacci sequence can be defined and checked using ten keystrokes.

Because of the different requirements for different aplets, the generic symbolic view is implemented as a derived instance of the symbolic view with several data fields and virtual routines to be overridden. The information for a symbolic view includes:

- Combine factor
- Total expression
- Group size
- Single-pick flag
- Softkey menu description
- Edit menu description
- Move focus procedure pointer
- Special initialization procedure pointer
- Edit terminator procedure pointer
- Expression checker
- Poststore procedure pointer.

For example, the sequence aplet combine factor is 3 (three terms define one sequence). The total expression is 30 (up to 10 sequences allowed). The group size is 3 (every three terms will share one check mark). The edit terminator procedure transforms definitions into internal form. The move focus procedure updates menu softkeys with the current sequence name. The expression checker rejects list or matrix expressions and initial terms that depend on the sequence independent variable n.

The symbolic view for the statistics aplet posed a new problem because we decided to show two expressions on one line, one for data and one for frequency, which is not supported by the choose box engine. The statistics symbolic view is implemented by customizing the input form to mimic a scrollable choose box with a check mark and two columns of data.

Setup Views

After expressions are set up, setup views are the natural places to go for setting the parameters for further explorations. The plot setup view is the main setup view, similar to the plot dialog box on the HP 48G, but enhanced with wider fields and a double-page design. The plot setup view, the symbolic setup view, and the numeric setup view are all based on the input form engine.

Plot View

The plot view is the most complicated of all aplet views. Students will spend most of their time here exploring the behavior of curves graphically and interactively.

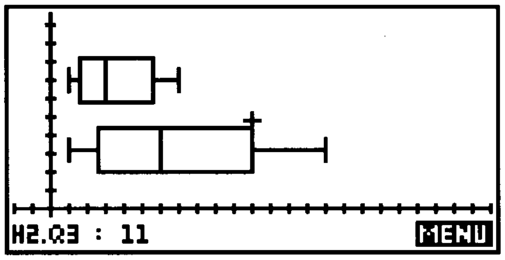


Fig. 16. Box and whisker plot.

The HP 38G takes the DRAW command in the HP 48G/GX plot window and improves it by implementing a "smart redraw." When switching back from other views, the picture, cursor position, and display mode are restored to the same state instantly, except when the defining parameters are changed. Plotting can be stopped and resumed later. When the user tries to move the cursor beyond the screen boundary, the graph shifts and redraws the scrolled-in portion. Zoom options are put into a choose box with more descriptive names. Instead of taking the current point as the first point, the box zoom prompts the user to select the first point. Tracing is improved and extended to statistics plots. Fig. 16 shows tracing on box and whisker plots.

For the function aplet, more areas of exploration are supported through the FCN menu key, which displays a choose box with choices of root, intersection, slope, area, and extremum. Fig. 17 shows the display for an area computation.

The plot view has overridable routines for curve drawing, curve tracing, FCN key handling, DEFN key handing, and other functions. For example, the function, parametric, and polar aplets share the same plot loop with different preprocessing, but the sequence aplet uses another plot loop for handling the discrete independent variable n.

For tracing, the sequence aplet implements four-way scrolling: the screen will scroll when the cursor is moved offscreen in all directions. The scatter plot overrides the FCN menu key to calculate and display a data-fitting curve. The information for a plot view includes:

- Draw procedure
- Independent variable ID
- Softkey menu description
- Display procedure pointer
- Key handler procedure pointer
- Pointer display procedure
- Draw axes flag
- Draw grid flag

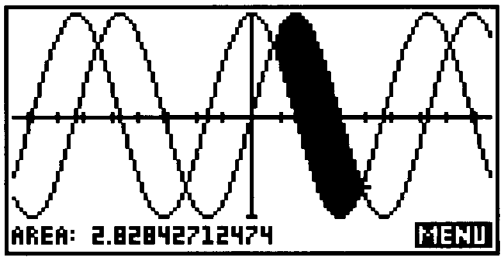
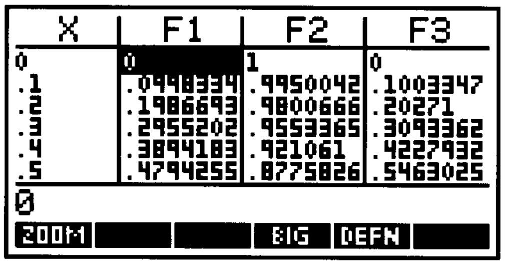


Fig. 17. Area computation accessible via the FCN softkey.



X	F1	F2	F3
0	0	1	0
.1	.0998334	.9950042	.1003347
.2	.1986693	.9800666	.20271
.3	.2955202	.9553365	.3093362
.4	.3894183	.921061	.4227932
.5	.4794255	.8775826	.5463025

0

Fig. 18. Automatic numeric view.

- Axis labels
- Display coordinate procedure
- Tracing procedures
- Coordinate display procedures
- Equation display procedure.

Numeric View

The numeric view lets a student explore the functions defined in the symbolic view in numeric form. The leftmost column displays values of the independent variable (except for statistics) and adjacent columns represent function results. There are two basic forms of the numeric view: automatic (Fig. 18) and build-your-own (Fig. 19). The automatic view displays a series of independent values with a starting value and a step specified by either the numeric setup view or the variables NumStart and NumStep.

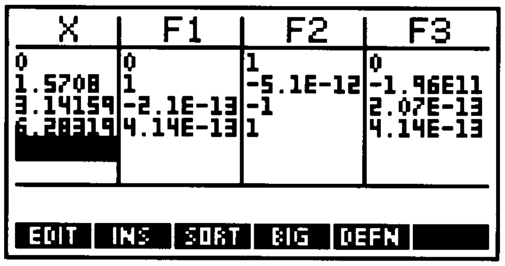
The BIG menu key lets the student display the numbers using a larger font. The ZOOM key provides a series of options for changing the start and step values for the independent variable display.

When the cursor is moved to the upper or lower boundaries of the display the table scrolls to show adjacent values. The table can also be reset by entering a new value for the independent variable in the leftmost column.

The build-your-own numeric view is useful for creating a table of interesting values. The values for the independent variable column in the build-your-own numeric view are accessible via the NumIndep variable.

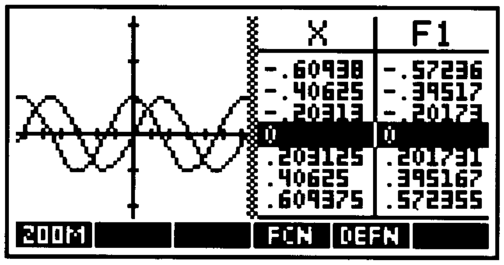
The information for a numeric view includes:

- Initialization procedure pointer
- Numeric zoom choices
- Softkey menu description
- Display procedure pointer
- Key handler procedure pointer
- Split plot-table configuration information.



X	F1	F2	F3
0	0	1	0
1.5708	1	-5.1E-12	-1.96E11
3.14159	-2.1E-13	-1	2.07E-13
6.28319	4.14E-13	1	4.14E-13

Fig. 19. Build-your-own numeric view.



X	F1
-.60938	-.57236
-.40625	-.39517
-.20313	-.20173
0	0
.203125	.201731
.40625	.395167
.609375	.572355

Fig. 20. Plot-table view.

Plot-Table View

The split plot-table view allows a student to combine the plot and numeric views (Fig. 20).

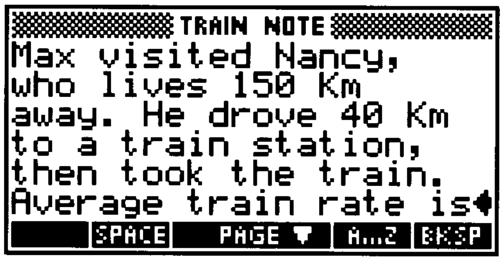
This view is implemented primarily as a plot view, with the right side of the display being a small numeric view that updates to reflect the position of the plot cursor. As the student moves the cursor from one function to another, the right side of the table changes to reflect the function being traced. The DEFN menu key displays the current function at the bottom of the display. This lets the student display the symbolic, plot, and numeric views of a function all at once.

Note View

Besides main aplet views like plot view, symbolic view, and so on, auxiliary views like the note view and sketch view are provided to add textual and pictorial descriptions to an aplet. The note view (Fig. 21) can be used to edit and display a text string attached to an aplet. The note can provide information about the aplet's subject, a suggested sequence of exploration, or the supported keys. The note view is basically a word-wrapping text editor with a 6-line-by-22-character display.

Although the original HP 48G/GX edit line supports multiple-line editing, individual lines are handled independently. When more than 22 characters are inserted into a line, that line gets scrolled to the left without affecting the rest of the lines. The HP 38G note view is implemented independently from the edit line code. Direct display routines are coded to show a character or a string at a specified location without generating intermediate GROBs. System-level keyboard handling code is modified to implement a general blinking cursor display scheme. Besides the text string to be edited, the note editor maintains a linewidth array for word-wrapping bookkeeping.

(continued on page 55)



TRAIN NOTE

Max visited Nancy,
who lives 150 Km
away. He drove 40 Km
to a train station,
then took the train.
Average train rate is

Fig. 21. Sample note view screen.

Distributed Software Team

The HP 38G calculator was the first product to be developed by Hewlett-Packard's Asia Pacific Personal Computer Division after the charter for handheld products was transferred there from Corvallis, Oregon. The software team was split between Oregon and Singapore, so we learned to make good use of communication technology, including Internet tools.

We had to figure out how to overcome a separation of over 9,000 miles and eight time zones. At the end of normal working hours in Oregon, the Singapore workday has just begun. Our specific communication needs included same-time, different-place meetings and documents that could be accessed anywhere, anytime.

Same-Time, Different-Place Meetings

Two of our Oregon team members were already experienced telecommuters, working at home a few days each week. They pioneered the idea of "virtual team meetings." Our first virtual meeting experiments were practiced with everyone in the office, sitting in our individual work areas with telephone headsets and a shared window running on all of our computers, so that we could all view and edit the same document at the same time.

We were able to work out the initial kinks in the process by standing up and shouting to each other if necessary. The next step was to link our two telecommuters from their Oregon homes. When we added our two Singapore engineers, we were already familiar with the procedures and etiquette for same-time, different-place meetings.

The team used HP 9000 workstations as well as computers at home that were connected to the workstations in the office using a LAN connection over 56-kbit/s frame relay lines. The applications used for sharing documents during meetings ran on our workstations and included Collage and HP SharedX. These applications allow a group of people to view and edit the same document simultaneously. However, our more typical way of sharing documents during meetings was by individually accessing our team's WorldWide Web pages.

Anywhere, Anytime Documents

Project documents are used to keep people informed of discussions and decisions and to provide product descriptions for the extended team members. Our challenge was to create and maintain project documents that would be accessible from many different locations. Our documentation process made extensive use of Internet technologies: hypertext documents, graphical browsers, and electronic mail.

Hypertext documents contain links to other documents. They can be easily created with any text editor using a special kind of formatting called Hypertext Markup Language (abbreviated HTML). Our team members were all able to get started with simple hypertext document creation after only a few minutes of study.

It became apparent to us that HTML was the best choice for developing and maintaining our project documentation, because:

- As a text-based source file format, HTML is compatible with our source code control system. This allowed us to maintain HTML documents with the same familiar tools that we used to maintain source code.
- A variety of HTML browsers such as Mosaic and Netscape are commonly available for multiple hardware platforms, which ensured that each team member could access our documentation.
- The hypertext nature of HTML documents provided a natural and extensible means to link together our rapidly growing documentation set.

We started with an HP 38G project home page. It had links to conventional software project documentation, such as our external reference specification (ERS) documents, which were also authored in HTML. But the power of linked documents to disseminate information also led us in new directions that supported the HP 38G project development process.

In past projects, we often regretted our inability to look back at topics discussed through less formal means like email, but with our HP 38G home page and tools to support HTML document generation, we finally had the enabling technologies to overcome this limitation. We began to archive our group email discussions and link them into our HP 38G project home page, indexed by date, author, and subject. On frequent occasions (sometimes as a group during a teleconference) we would refer back to these email archives to revisit a line of reasoning about design choices.

The success of the discussion archival technique led to the side benefit of a uniform, centrally maintained mailing list for software team members. Over time we expanded this concept to include multiple discussion groups, each with associated email archives and each specialized for a different audience.

Soon we were using the HP 38G home page to collect information that was previously scattered throughout paper documents or engineers' minds. For the first time, documents detailing the setup of our software debugger systems, EPROM burners, and more were readily available around the clock.

Other departments involved with the HP 38G began to seek out the resources we made available through the HP 38G home page, and it was quite rewarding for us to be able to direct them to our pages. The frustration of trying to coordinate delivery of information by hand was becoming a distant memory. Some departments, with our encouragement and support, began making information available through linked documents. These we gladly added to our project page, making it more valuable for all parties. For example, when our colleagues in Singapore had industrial design drawings to share, these were made available to all through the HP 38G home page, and the QA department's software test plans were added so that engineers could review and comment on them to ensure complete test coverage.

Conclusion

Our use of linked HTML documents has increased steadily since the HP 38G project. We now have more email discussion groups, our own Web server to simplify access for remote connections, a team home page with links to all of our project home pages, a dynamic team calendar of events, a "what's new" service for quickly learning what's changed, and a searching service for finding specific topics amongst our wealth of interconnected documents. We expect exciting new developments to further increase our productivity as we expand our knowledge and use of the Web.

Sketch View

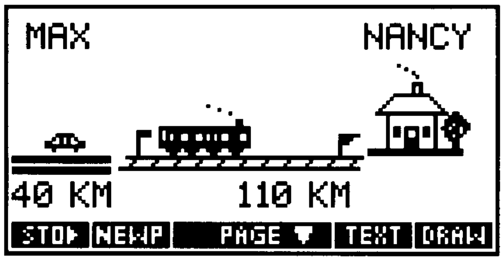
Some people believe that a picture is worth a thousand words, so the HP 38G generalizes the HP 48G/GX's bitmap editing features to form an aplet sketch view. With the sketch view, the user can edit and display a set of bitmaps attached to an aplet. Holding down the page-down or page-up key shows a prestored animation sequence (Fig. 22).

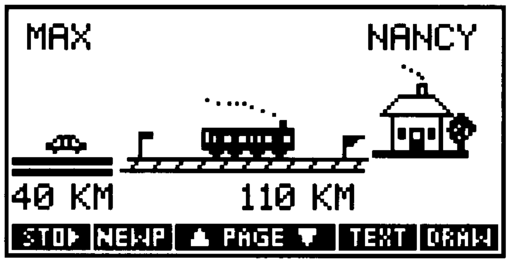
Compared with the HP 48G/GX, the HP 38G limits the bitmap editing features and improves the user interface and implementation. The HP 38G implements a rubber band algorithm when the user drags the second point to define a line, rectangle, or circle. The circle drawing code uses a fast integer-based iterative algorithm. The user can drag a text string in the small or medium font to any location. The HP 38G can store a selected portion of a screen into a GROB variable and recall it. When bitmaps are stored back into an aplet directory, they are first trimmed to the minimum enclosing rectangle to save RAM space.

Additional Views

Each aplet type defines additional views available in the **VIEWS** menu. For example, the function aplet offers some hybrid views and some preconfigured plot views. Fig. 23 shows the function **VIEWS** menu.

In addition, a user can define a new set of special views that provide high-level tools for an aplet. Such a custom interface makes the aplet easier to explore and hides details of the calculator's operation.

(a)


(b)


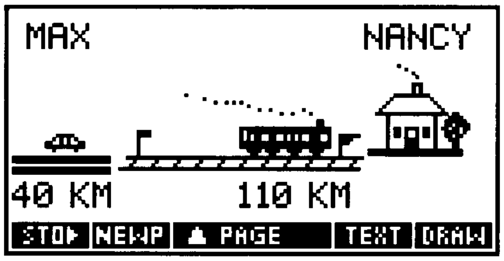
(c)


Fig. 22. An animation sequence in sketch view.

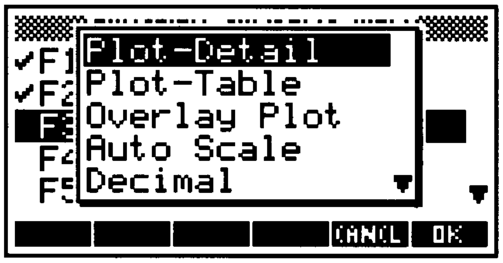


Fig. 23. Function **VIEWS** menu.

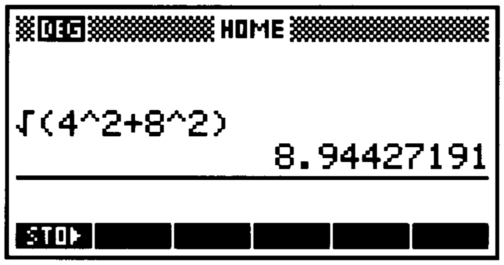
The Home Environment

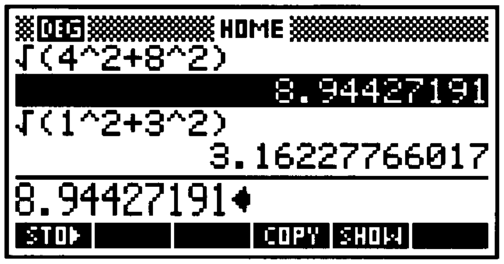
The freeform home environment fills the traditional calculator role of supporting quick calculations. The user enters expressions in algebraic form and the value of the expression (usually a number) is returned.

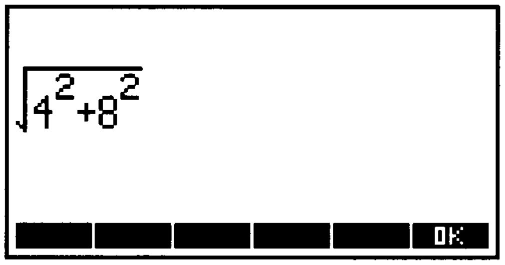
Unlike aplets, the home environment provides access to all calculator resources, including lists, matrices, graphics objects, and programs.

The History Stack

The text of the input and the result are stored on a history stack (Fig. 24a). The user can review the items on the history stack and reuse those items as parts of the current input (Fig. 24b). Expressions and equations on the history stack can be shown in two-dimensional mathematical format (Fig. 24c).

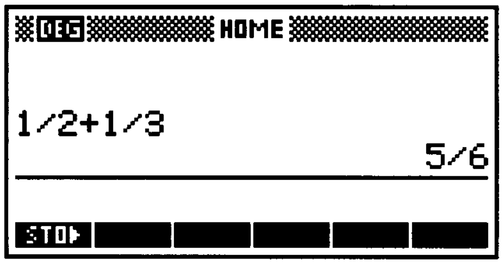
(a)

```
DEG HOME
√(4^2+8^2)
                8.94427191
STO▸
```

(b)

```
DEG HOME
√(4^2+8^2)
                8.94427191
√(1^2+3^2)
             3.16227766017
8.94427191
STO▸         COPY SHOW
```

(c)

$\sqrt{4^2+8^2}$

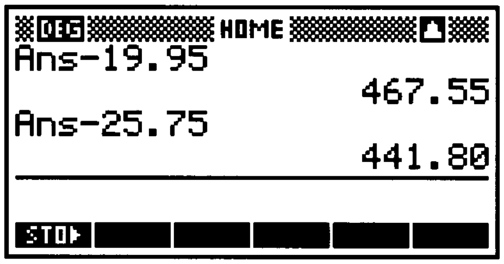
OK

Fig. 24. (a) History stack. (b) Previous result copied into the command line. (c) EquationWriter display of the same expression.



```
DEG  HOME

1/2+1/3
                 5/6
STO▸
```

Fig. 25. Fraction number format.



```
DEG  HOME
Ans-19.95
              467.55
Ans-25.75
              441.80
STO▸
```

Fig. 27. Checkbook calculations in the home environment.

The HP 38G includes special features to help beginners. To help students learn about fractions, the HP 38G has fraction number format, which uses continued fractions to convert results to rational form (Fig. 25). To help students unfamiliar with standard algebraic syntax, the HP 38G attempts to interpret ill-formed expressions as implied multiplication (Fig. 26).

The Variable Ans

Each time the user inputs an expression, the value of the expression is stored in a variable named Ans. This current value of Ans is placed on the history stack, and the name Ans can be used in the next calculation. Even when the user enters a command that performs some action but doesn't return a value, the current value of Ans is placed on the history stack.

If the user starts an input with an infix function such as +, –, *, or /, the calculator inserts the name Ans first. This saves keystrokes for tasks such as balancing a checkbook (Fig. 27). If the user presses **ENTER** without input, the previous input is repeated (Fig. 28). This saves keystrokes for iterative operations.

The VAR and MATH Menus

To organize its extensive resources, the HP 38G presents the most-used variables and functions on the keyboard. Additional resources are available in the **VAR** and **MATH** menus. These two-column menus offer specific items in the right column, categories of items in the left column, and broader choices on menu keys.

In the **VAR** menu (Fig. 29) the user can choose to examine variables either from the current aplet or from the shared home variables. The user also can choose either the name or the value of a variable.

In the HP 38G most variables are strongly typed, that is, for many variables the value must be a specific type. For example, the variable X must contain a real number, Z1 must contain a complex number, and M2 must contain a matrix. Several classes of variables contain exactly ten variables, such as the ten complex variables Z1, Z2, ..., Z9, Z0 (Fig. 30).

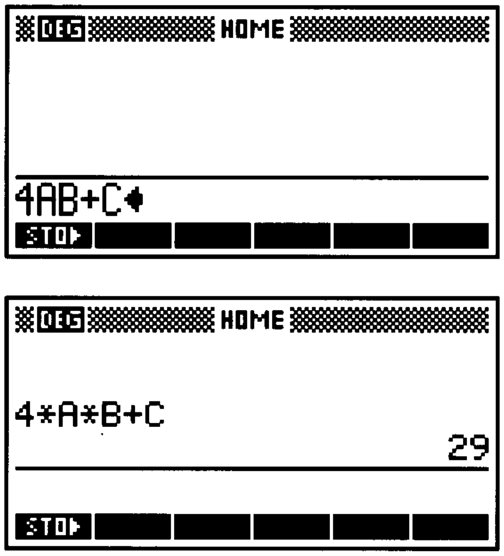
Variables are used not only for mathematical objects, but also for modes. For example, storing the constant Degrees (whose numerical value is 1) into the variable Angle selects degrees angle mode.

The classes of home variables include complex numbers (Z1 to Z0), graphics objects (G1 to G0), aplets (user-selected names), lists (L1 to L0), matrices (M1 to M0), modes (fixed descriptive names), notepad (user-selected names), programs (user-selected names), and real numbers (A to Z, θ).

The **MATH** menu (Fig. 31) offers additional functions, commands, and constants not available on the keyboard. The categories of functions are: calculus functions, complex-number functions, constants, hyperbolic functions, list functions, loop functions, matrix functions, functions of polynomials, probability functions, real-number functions, statistics functions, functions for symbolic manipulation, tests, and trigonometric functions.

Lists and Matrices

For these composite variable types there are catalogs that report the variables' sizes, along with special editors to enter and modify the elements. The catalogs and editors are tasks,



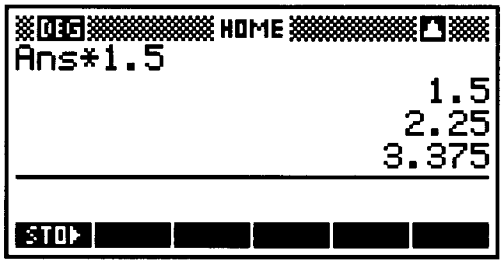
(a)

```
DEG  HOME

4AB+C◆
STO▸
```

(b)

```
DEG  HOME

4*A*B+C
                  29
STO▸
```

Fig. 26. (a) Implied multiply input. (b) Result.



```
DEG  HOME
Ans*1.5
                 1.5
                2.25
               3.375
STO▸
```

Fig. 28. If the user presses **ENTER** without input, the previous input is repeated. This saves keystrokes for iterative operations.

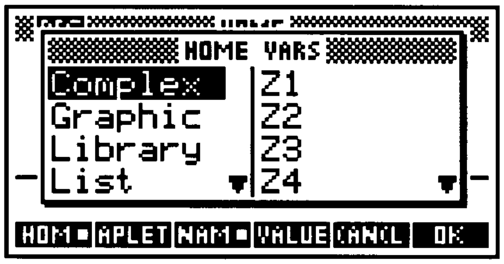


Fig. 29. **VAR** menu of the home environment.

so the user can easily move among aplets, home, catalogs, and editors. The list editor (Fig. 32) is one-dimensional. The matrix editor is two-dimensional (Fig. 33).

Lists and matrices can also be used as functions of an index value. For example, L1 is a list, while L1(2) is an expression whose value is the second element of L1.

Notes and Programs

For these text variable types there are catalogs that show the user-selected names and editors that allow freeform text input. The notepad holds simple text files such as phone lists (Fig. 34).

There is a program with the fixed name Editline, which holds the most recent input from the home environment. The user can choose to edit the most recent home input from within the program editor, or to execute the program Editline from the home environment by simply pressing **ENTER**.

Programs aren't parsed until the first time they're run. Because of this, and because the program editor is a task,

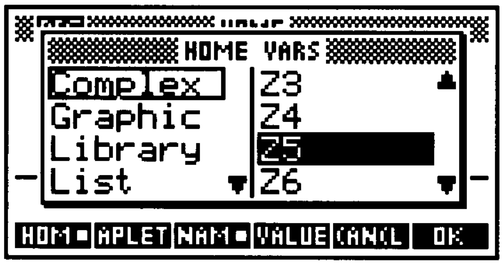


Fig. 30. Most variable values must be a specific type and several classes of variables contain exactly ten variables. For example, Z1 to Z0 are complex variables.

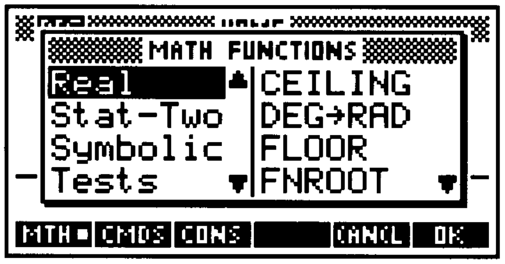
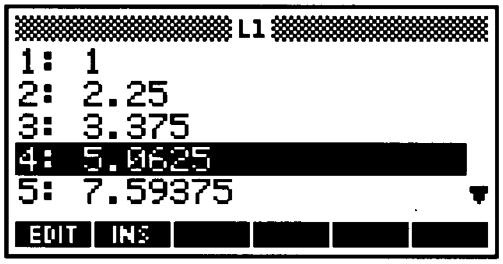
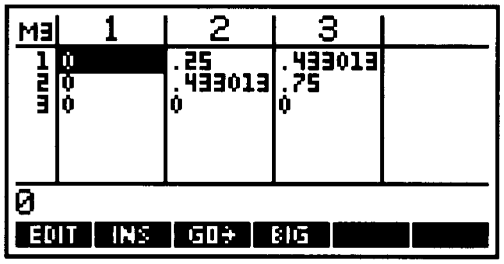


Fig. 31. **MATH** menu.



```
L1
1: 1
2: 2.25
3: 3.375
4: 5.0625
5: 7.59375
```

Fig. 32. The list editor is one-dimensional.



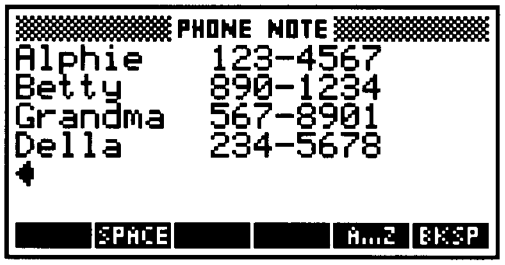
M3	1	2	3
1	0	.25	.433013
2	0	.433013	.75
3	0	0	0

```
0
```

Fig. 33. The matrix editor is two-dimensional.

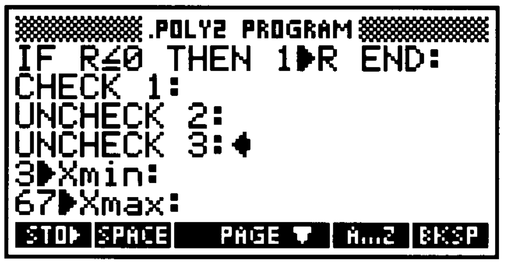
the user can write a program a little at a time, leaving the program in an invalid state between editing sessions. Fig. 35 shows part of a program.

Commands that perform some action and return no result can appear only within programs (which includes Editline). The categories of commands are: commands to control aplets, branch commands, commands for scaled drawings, commands to manipulate graphics objects, loop commands, matrix commands, printing commands, prompt commands for input and output, and statistics commands.



```
PHONE NOTE
Alphie    123-4567
Betty     890-1234
Grandma   567-8901
Della     234-5678
```

Fig. 34. The notepad holds simple text files such as phone lists.



```
.POLY2 PROGRAM
IF R≤0 THEN 1▶R END:
CHECK 1:
UNCHECK 2:
UNCHECK 3:
3▶Xmin:
67▶Xmax:
```

Fig. 35. Part of a program.

Programs that Support Aplets

The most important purpose of programs in the HP 38G is to support the user-defined entries within the **VIEWS** menu (Fig. 36). Using the SETVIEWS command, the user specifies a number of special views that represent the tools for manipulating the aplet. These tools can be presented at a high level, using terms relevant to the particular aplet and hiding the actual methods of the calculator from the aplet user.

The specification for each special view includes a prompt, which appears in the **VIEWS** menu. It also includes a program, which is run when the special view is selected, and a view specification, which determines the standard view (plot, symbolic, etc.) to be started when the program is done.

If an aplet has special views with start or reset prompts, pressing the menu keys START or RESET within the **LIB** catalog (Fig. 37) automatically selects the corresponding special view.

While the prompt category of commands enables programmatic interaction with the user, the main goal of HP 38G programs is to modify the configuration variables of the current aplet. After the program runs, the standard view specified in the **VIEWS** menu starts, operating with the configuration left by the program. This approach has the advantage that the interface to the standard view is familiar to the user.

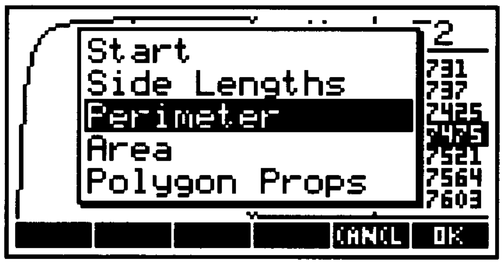


Fig. 36. **VIEWS** menu with user-specified special views.

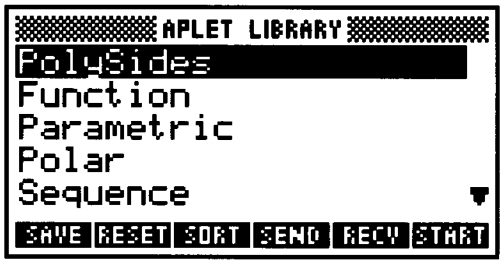


Fig. 37. **LIB** catalog.

All the components needed for special views are automatically managed by the HP 38G. The menu of special views is stored as a part of the aplet, and the programs used by the special views are automatically transferred with the aplet.

This makes aplets simple to use: a single request gets the aplet and associated programs, and the **VIEWS** menu offers the high-level tools that allow the user to focus more on the content of the aplet and less on the calculator.

Acknowledgments

We would like to acknowledge the rest of the calculator software team: Helen Choy, Gabe Eisenstein, and Charles Patton. Youwen Wu worked closely with us to ensure quality. Diana Roy (learning products) and Kevin Myers and Cary McCallister (technical support) provided many valuable user interface design suggestions. Ron Brooks from our marketing department contributed input from educators. Our education advisory committee provided valuable day-to-day input on our design decisions and usability issues.

References

1. D.K. Byrne, et al, "An Advanced Scientific Graphing Calculator," *Hewlett-Packard Journal*, Vol. 45, no. 4, August 1994, pp. 6-22.
2. T.W. Beers, et al, "HP 48SX Interfaces and Applications," *Hewlett-Packard Journal*, Vol. 42, no. 3, June 1991, pp. 13-21

Creating HP 38G Aplets

This article explores a simple aplet and shows how to construct an aplet called PolySides.

by James A. Donnelly

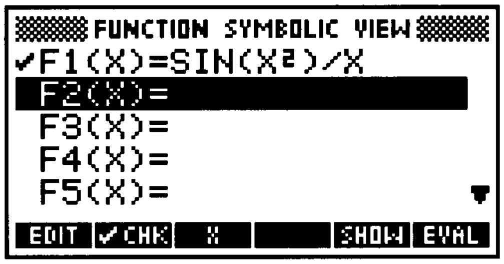
In designing an aplet for the HP 38G calculator, it's important to remember two guiding principles of the HP 38G design. First, the design recognizes that in a classroom setting there is (and should be) a large discrepancy between the amount of information entered into a graphing calculator and the amount of information produced by the calculator. We sought to minimize the calculator input required of the student and the teacher and maximize the returned information. Secondly, we sought to exclude irrelevant possibilities. By reducing irrelevant choices you focus the user's attention on the subject of interest and avoid distractions from things that don't matter. Many choices made during the design of the HP 38G were based on this goal.

Aplets contain information and have views. The information in an aplet consists of every major piece required to produce the views: the equations, setup information, mode information, sketch or text annotations, and attached libraries or programs. In this article we'll explore a simple aplet, then look at an aplet called PolySides and examine how it was constructed.

Using Built-in Aplets

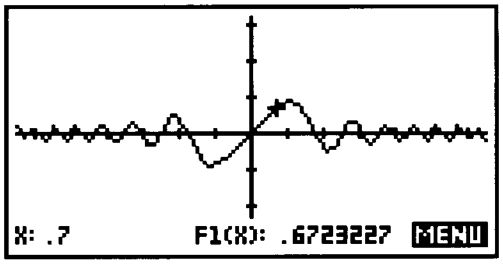
When the HP 38G is first turned on, the built-in aplets are empty. There's no equation, note, or sketch. When the user adds this information, these aplets come alive. You can save an aplet at any time, so it's easy to start one project, change in midstream to another, then come back to the first. (Because the appearance of the screens depends on the information you enter, the screens on your calculator may look different from the example screens in this article.)

A simple way to illustrate the aplet concept is to explore the equation SIN(X^2)/X. Select the function aplet, press **SYMB**, and enter the equation. ◆



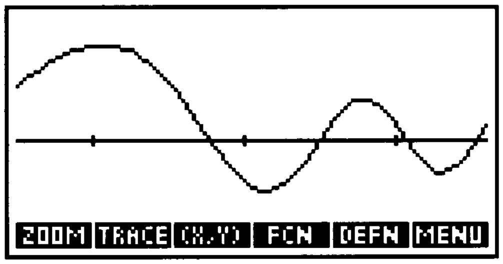
```
FUNCTION SYMBOLIC VIEW
✓F1(X)=SIN(X²)/X
 F2(X)=
 F3(X)=
 F4(X)=
 F5(X)=
```

Now press **PLOT** to see the plot using the default plot parameters. ◆

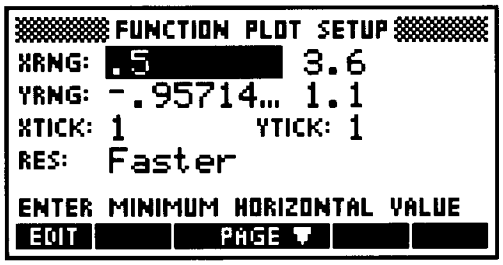


```
X: .7          F1(X): .6723227
```

Use the box option under **ZOOM** to look at a smaller area of the plot. ◆



You can see the plot scale by pressing [shift]**PLOT** to see the plot setup view. ◆

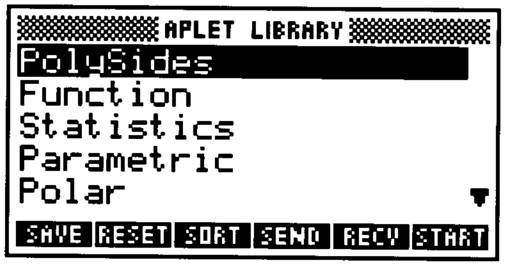


```
FUNCTION PLOT SETUP
XRNG: .5          3.6
YRNG: -.95714...  1.1
XTICK: 1       YTICK: 1
RES:  Faster
ENTER MINIMUM HORIZONTAL VALUE
```

At this point, you have an aplet that's completely dedicated to your interest in the function SIN(X^2)/X, and the aplet can be saved under a unique name or transmitted to another HP 38G or a computer. When the aplet is restarted on the original or another HP 38G, all modes and scales are set just the way they were saved.

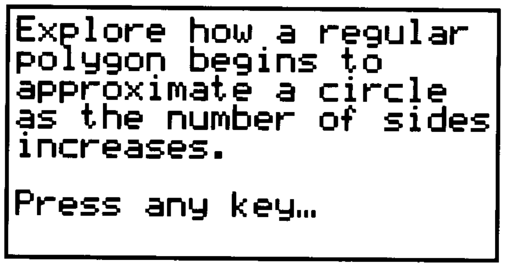
Exploring the PolySides Aplet

The PolySides aplet is designed to explore how a regular polygon can approximate a circle as the number of sides increases. PolySides can be loaded from a disk or another HP 38G in a single operation from the LIB catalog. Once loaded, PolySides appears in the aplet library. ➡



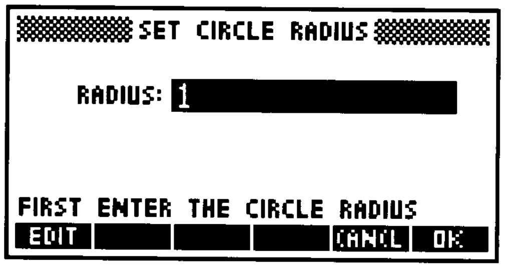
```
APLET LIBRARY
PolySides
Function
Statistics
Parametric
Polar
SAVE RESET SORT SEND RECV START
```

We are now just a single keystroke away from exploring the aplet. To begin the exploration, press START. The motivation for the lesson is displayed first. ➡



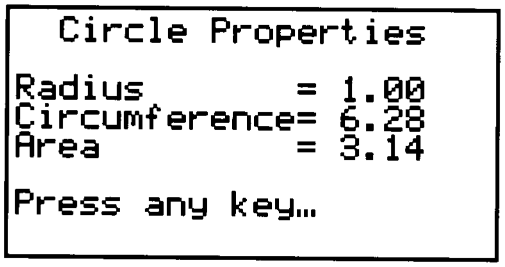
```
Explore how a regular
polygon begins to
approximate a circle
as the number of sides
increases.

Press any key...
```

After the introduction has been read and a key pressed, the next step is to enter the radius of the circle to be approximated. ➡



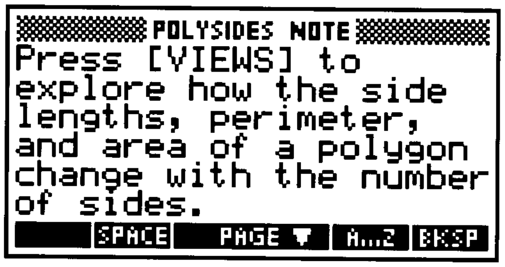
```
SET CIRCLE RADIUS
RADIUS: 1

FIRST ENTER THE CIRCLE RADIUS
EDIT CANCL OK
```

After the radius has been entered, the properties of the circle are displayed. ➡

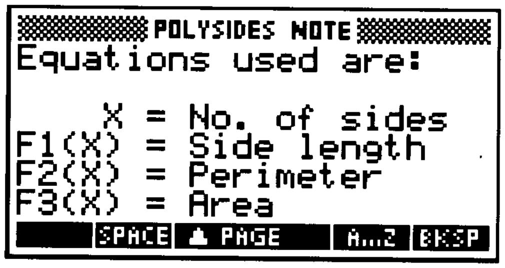


```
Circle Properties

Radius       = 1.00
Circumference= 6.28
Area         = 3.14

Press any key...
```

After the circle properties have been viewed, the note view of the aplet is displayed. The PAGE menu key switches between the pages of the note. ➡

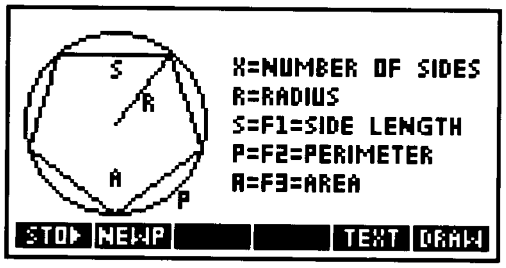


```
POLYSIDES NOTE
Press [VIEWS] to
explore how the side
lengths, perimeter,
and area of a polygon
change with the number
of sides.
SPACE PAGE ▼ A...Z BKSP
```



```
POLYSIDES NOTE
Equations used are:

    X = No. of sides
F1(X) = Side length
F2(X) = Perimeter
F3(X) = Area
SPACE ▲ PAGE A...Z BKSP
```

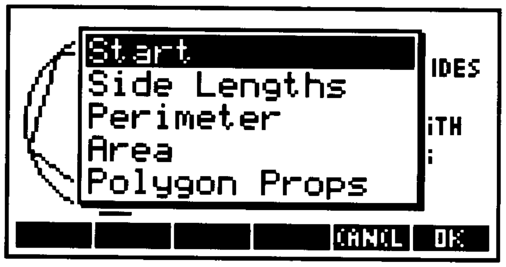
To see a sketch of the problem, press SKETCH. ➡



```
X=NUMBER OF SIDES
R=RADIUS
S=F1=SIDE LENGTH
P=F2=PERIMETER
A=F3=AREA
STO▸ NEWP TEXT DRAW
```

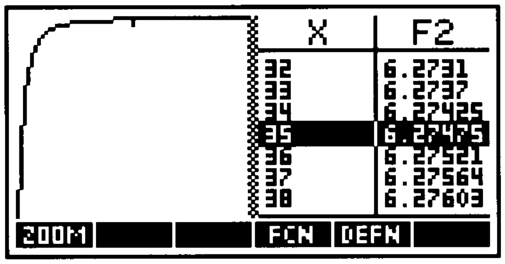
So far we have seen a pretty complete summary of the lesson with less than a dozen keystrokes. Now we can begin to explore how may sides are needed to approximate a circle.

Our central point of departure for PolySides is VIEWS (as mentioned in the note view). ➡



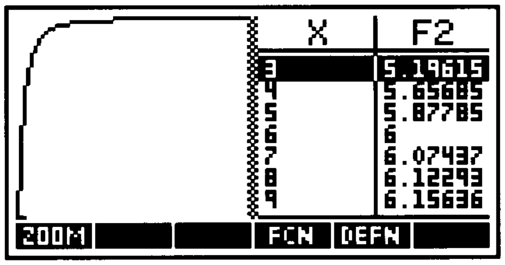
```
Start
Side Lengths
Perimeter
Area
Polygon Props
CANCL OK
```

These choices let you determine what aspect of a polygon approximation to a circle you'd like to explore. To explore how the perimeter of the polygon approximates the circle, move the highlight down to Perimeter and press OK. The plot-table view is presented automatically. ➡



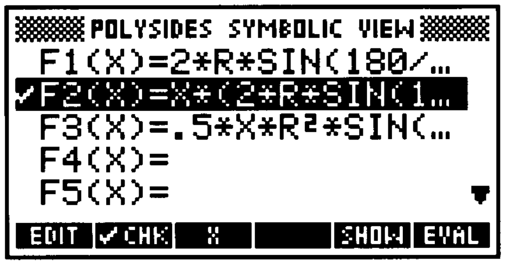
X	F2
32	6.2731
33	6.2737
34	6.27425
35	6.27475
36	6.27521
37	6.27564
38	6.27603

This is one of the HP 38G's split views, showing the plot and numeric views of the perimeter approximation at the same time. The variable X represents the number of sides, and the equation stored in F2 returns the perimeter as a function of X. Pressing the left or right arrow key moves the cursors for the plot and numeric views simultaneously. When the cursor is at the left edge of the plot, the table on the right shows the rapid change of the perimeter from a triangle to a nonagon. ▼



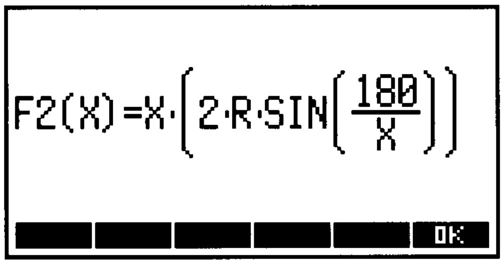
X	F2
3	5.19615
4	5.65685
5	5.87785
6	6
7	6.07437
8	6.12293
9	6.15636

To see the equation, just press **SYMB**. ▼



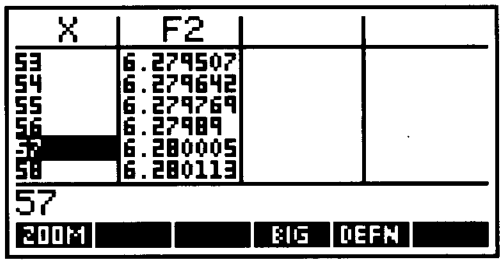
```
POLYSIDES SYMBOLIC VIEW
 F1(X)=2*R*SIN(180/…
✓F2(X)=X*(2*R*SIN(1…
 F3(X)=.5*X*R²*SIN(…
 F4(X)=
 F5(X)=
```

The symbolic view is where aplet equations are stored. The check mark beside F2 indicates that when a plot or numeric view is displayed, F2 will be used. Another view of the equation is available by pressing SHOW. ▼



$$F2(X)=X\cdot\left(2\cdot R\cdot SIN\left(\frac{180}{X}\right)\right)$$

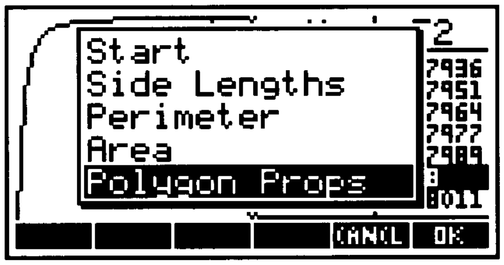
Notice that after the introduction there is no fixed order of events. You can change to any of the views at any time or press **HOME** at any time to do a short calculation. Part of the design philosophy of aplets is to let the student explore the lesson at will, without following any particular algorithm. Once you've explored the approximations to the perimeter and area, and perhaps explored the effect on side lengths, you may decide that a polygon with 57 sides yields a fair approximation to the circumference of a unit circle. ▼



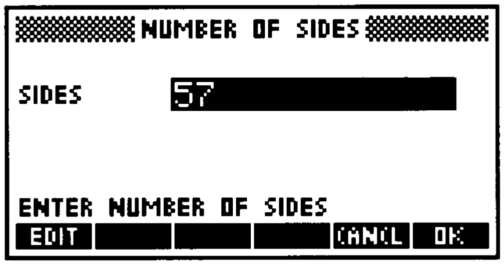
X	F2		
53	6.279507		
54	6.279642		
55	6.279769		
56	6.27989		
57	6.280005		
58	6.280113		

```
57
```

Now you can look at a summary of a 57-sided polygon by selecting Polygon Props (for polygon properties) from the **VIEWS** choices. ▼

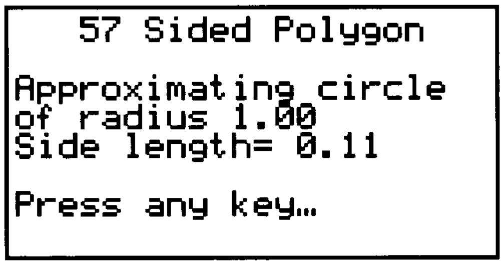


Press OK to select this choice. Since the cursor was last on X = 57, the number of sides defaults to 57. ▼

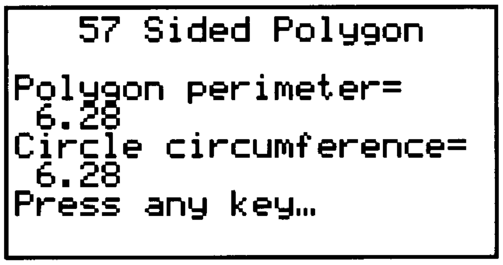


```
NUMBER OF SIDES

SIDES        57

ENTER NUMBER OF SIDES
```

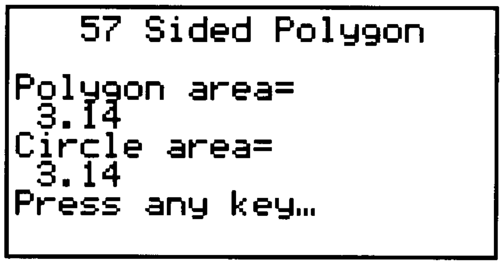
After the user accepts (or alters) the number of sides, the polygon data is displayed. ▼



```
   57 Sided Polygon

Approximating circle
of radius 1.00
Side length= 0.11

Press any key…
```



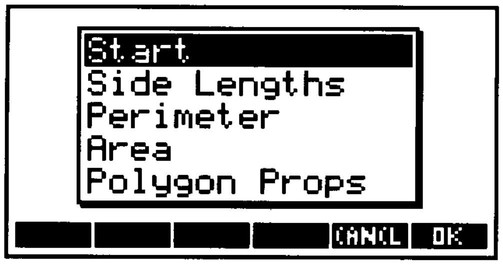
```
   57 Sided Polygon

Polygon perimeter=
 6.28
Circle circumference=
 6.28
Press any key…
```



```
 57 Sided Polygon
Polygon area=
 3.14
Circle area=
 3.14
Press any key…
```

Now we've observed that a 57-sided polygon does a fair job of approximating a unit circle. More explorations are possible. For instance, if the circle has a very large radius, how many more sides are required for a polygon to closely approximate the circle's area?

After the last polygon property screen has been acknowledged with a keystroke, the **VIEWS** menu is displayed. ➡



This lets you start with another circle or go back to find a different approximation.

Notice that the keystrokes involved for the entire exploration have been oriented to the views. We haven't entered a single bit of setup or scaling information beyond the radius of the circle being approximated. This is the goal of a well-formed aplet; more attention is directed to the lesson and less attention is directed to the mechanics of the calculator. You can imagine a teacher distributing this aplet in a classroom for part of a day's lesson.

If you're familiar with the HP 48G/GX calculators, you'll notice that variables like EQ and PPAR have never been mentioned, even though we changed the equation and plot scaling parameters several times. Furthermore, we never went near an input form to set the scale manually (although we could have done so by pressing [shift] **PLOT** to display the plot setup view).

Building the PolySides Aplet

PolySides was constructed using the function aplet and five user programs attached with the SETVIEWS command.

Beginning with a reset function aplet, the equations, note, and sketch were entered directly on the calculator. The equations are:

- Side Length `F1(X)=2*R*SIN(180/X)`
- Perimeter `F2(X)=X*2*R*SIN(180/X)`
- Area `F3(X)=.5*X*R²*SIN(360/X)`

Then five programs were entered, one for each entry in **VIEWS**. After the programs were entered, they were attached to the aplet with the SETVIEWS command. For convenience during aplet development, a separate program, SetPolyViews, was entered that contains the SETVIEWS command. SETVIEWS allows you to customize the **VIEWS** key with your own entries and selected entries from the default **VIEWS** choices. SETVIEWS takes triplets of arguments. Each triplet contains the prompt that will appear in the **VIEWS** list, a program name, and a number corresponding to the view that should be displayed after the program is executed. The SETVIEWS command for PolySides looks like this:

```
SETVIEWS
"Start";".Poly1";8;
"Side Lengths";".Poly2";16;
"Perimeter";".Poly3";16;
"Area";".Poly4";16;
"Polygon Props";".Poly5";7;
```

When Side Lengths is selected, the program .Poly2 is executed, then view number 16 is displayed. View number 7 is the **VIEWS** list, view number 8 is the notes view, and view number 16 is the split plot-table view. A SETVIEWS convention is that if a prompt is named Start or Reset, it is associated with the START or RESET menu key in the **LIB** catalog. The PolySides aplet takes advantage of this feature so that pressing START in the **LIB** catalog gets a student underway in a single keystroke.

The programs are listed below. (Note that the translation codes used are the same as translate code 3 on the HP 48G/GX.)

.Poly1 (Start)

```
ERASE:                              Clear the display
DISP 1;"Explore how a regular":     Display the
DISP 2;"polygon begins to":         motivation for
DISP 3;"approximate a circle":      the aplet
DISP 4;"as the number of sides":
DISP 5;"increases.":
DISP 7;"Press any key...":
FREEZE:
IF R\<=0 THEN 1\|>R END:
INPUT R;"SET CIRCLE RADIUS";        Prompt for the
" RADIUS:";                         circle radius
"FIRST ENTER THE CIRCLE RADIUS";R:
ERASE:                              Clear the display
2\|>Digits: Fixed\|>Format:         Set FIX 2 display
                                    mode
DISP 1;" Circle Properties":        Display the screen
                                    title
DISP 3;"Radius = " R:               Display each
                                    property
DISP 4;"Circumference = " 2*\pi*R:
DISP 5;"Area = " \pi*R^2:
DISP 7;"Press any key...":
FREEZE:                             Wait for a key
                                    press
Standard\|>Format:ERASE             Restore standard
                                    display format,
                                    clear the display
```

Programs .Poly2 through .Poly4 do similar jobs. Each validates the value for R (the radius), selects the appropriate equations in the symbolic view, calculates the plot scale parameters, and sets the initial values for the numeric view. Note that the plot scale is calculated to fit above the menu. The menu occupies 1/8 the display, so an adjustment of 1/8 the calculated vertical range is made to the vertical scale.

.Poly2 (Side Lengths)

```
IF R\<=0 THEN 1\|>R END:            Ensure a reason-
                                    able value for R
CHECK 1:                            Select equation F1
UNCHECK 2:
UNCHECK 3:
3\|>Xmin:                           Set the plot scale
67\|>Xmax:
F1(Xmax)\|>Ymin:
F1(Xmin)\|>Ymax:
Ymin-.125*(Ymax-Ymin)\|>Ymin:       Fit the plot above
                                    the menu
3\|>NumStart:                       Set the numeric
                                    view to begin at 1
                                    and increase in
1\|>NumStep:                        steps of 1
```

.Poly3 (Perimeter)

```
IF R\<=0 THEN 1\|>R END:
UNCHECK 1:
CHECK 2:                            Select equation F2
UNCHECK 3:
3\|>Xmin:
67\|>Xmax:
F2(Xmin)\|>Ymin:
F2(Xmax)\|>Ymax:
Ymin-.125*(Ymax-Ymin)\|>Ymin:
3\|>NumStart:
1\|>NumStep:
```

.Poly4 (Area)

```
IF R\<=0 THEN 1\|>R END:
UNCHECK 1:
UNCHECK 2:
CHECK 3:                            Select equation F3
3\|>Xmin:
67\|>Xmax:
F3(Xmin)\|>Ymin:
F3(Xmax)\|>Ymax:
Ymin-.125*(Ymax-Ymin)\|>Ymin:
3\|>NumStart:
1\|>NumStep:
```

.Poly5 (Polygon Properties)

```
INPUT X;"NUMBER OF SIDES";"SIDES";"ENTER NUMBER
OF SIDES";X:                        Sides prompt
ERASE:                              Clear the display
Standard\|>Format:                  Set standard
                                    format
DISP 1;" " X " Sided Polygon":      Display the screen
                                    title
2\|>Digits: Fixed\|>Format:         Set FIX 2 format
DISP 3;"Approximating circle":      Display circle
DISP 4;"of radius " R:              radius and poly-
DISP 5;"Side length= " F1(X):       gon side length
DISP 7;"Press any key...":
FREEZE:                             Wait for a key
                                    press
ERASE:                              Clear the display
Standard\|>Format:                  Set standard
                                    format
DISP 1;" " X " Sided Polygon":      Display the screen
                                    title
Fixed\|>Format:                     Set FIX 2 format
DISP 3;"Polygon perimeter=":        Display the
DISP 4;" " F2(X):                   perimeter and
DISP 5;"Circle circumference=":     circumference
DISP 6;" " s*\pi*R:
DISP 7;"Press any key...":
FREEZE:                             Wait for a key
                                    press
ERASE:                              Clear the display
Standard\|>Format:                  Set standard
                                    format
DISP 1;" " X " Sided Polygon":      Display the screen
                                    title
Fixed\|>Format:                     Set FIX 2 format
DISP 3;"Polygon area=":             Display the area
DISP 4;" " F3(X):
DISP 5;"Circle area=":
DISP 6;" " \pi*R^2:
DISP 7;"Press any key...":
FREEZE:                             Wait for a key
                                    press
Standard\|>Format: ERASE            Restore standard
                                    display format,
                                    clear the display
```

Building Your Own Aplets

Your own aplets can be as simple as the first example, somewhat more involved (like the PolySides aplet), or very involved, with more complex user programs. Creating a new base aplet that can be downloaded into the HP 38G requires the use of System RPL programming beyond the built-in user programming language.

Basic Steps. The basic steps for building an aplet are:

- Pick an appropriate built-in base aplet—function, polar, parametric, etc.
- In each of the views, enter the information that applies. For instance, you would put your equations in the symbolic view, plot parameters in the plot setup view, and so on.
- Put a sketch of the problem in the sketch view and a description of the problem in the note view.
- For a fancier aplet, use SETVIEWS to attach user programs to the **VIEWS** key. Remember that it may be handy to have a separate program that stores the SETVIEWS command while you're developing the aplet.

Flow of Control. In general, the user of the aplet should determine the flow of control, so that a problem can be explored freely. The PolySides aplet provides a little extra guidance by using the program .Poly1 attached to the start prompt in **VIEWS**. .Poly1 displays an initial screen, prompts for the circle radius, displays the circle's properties, and finally exits to the note view as directed by the SETVIEWS command. In general, the user should be able to navigate freely among the choices in **VIEWS** or the various views, or go home and come back.

Shared Variables. The global variables A to Z, Z1 to Z0, and so on are not stored within an aplet, so if an aplet depends on the value of a global variable it's a good idea to have the program associated with Start in **VIEWS** set these values.

Authors

June 1996

6 Digital System Integration

Patrick J. Byrne

As R&D manager at HP's Colorado Springs Division, Pat Byrne is responsible for logic analyzers, emulators, and digital system solutions. Born in Palo Alto, California, he earned a BSEE degree in 1982 from the University of California at Berkeley and an MSEE degree in 1988 from Stanford University. He joined HP's Integrated Circuits Business Division in 1983 and worked as an R&D project and section manager from 1986 to 1991 designing custom integrated circuits for HP printers, computers, instruments, and communications products. In 1991, he became an R&D section manager for logic analyzers and emulators at the Colorado Springs Division. He was the R&D section manager for the HP 16505A prototype analyzer. He is professionally interested in digital system development tools and processes and has authored two articles on logic analyzers. He is named as an inventor in a pending patent on configurable logic measurement systems. Pat is married and has three children. He enjoys outdoor sports such as golf, mountain biking, cross-country skiing, and soccer.

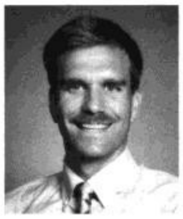
15 Prototype Analyzer Architecture

Jeffrey E. Roeca

Jeff Roeca is an R&D project manager at HP's Colorado Springs Division. He was a software contributor for the drag-and-drop GUI, fileout, and system level software for the HP 16505A prototype analyzer and currently manages the software development effort. Jeff joined HP in 1989 and has worked on the HP 16500 system software, the HP 1660 UI design (as principal designer), and the HP 16505A software design. He was awarded a BS degree in applied mathematics at California Polytechnic State University at San Luis Obispo. Before HP, he worked at ROLM Corporation on fault tolerant switching software, database integrity software, and telephony feature software. Born in Encino, California, Jeff is married and has three children. His hobbies include skiing, tennis, biking, hiking, and music.

22 Encapsulated Measurement Server

Gregory J. Peters



Born in Cedar Rapids, Iowa, Greg Peters earned a BSEE degree from Iowa State University in May 1984 and joined HP's Colorado Springs Division that June. He has provided technical support for HP pulse, pattern, IC test, and optical test products, assisted in defining the pattern generator and other measurement modules for the HP 16500 system, and participated in market investigation, definition, and introduction of products that now form the core of HP's digital prototyping solutions. Part of his responsibility was to act as the contact with HP's Workstation Division on sourcing issues. He is currently the lead product marketing engineer for the HP 16500 system including measurement modules and the HP 16505A prototype analyzer. He has authored four articles on logic analyzers and pattern generators. In 1991 he earned an MBA degree from the University of Colorado at Colorado Springs. In his free time, Greg enjoys mountain biking, skiing, golf, volleyball, and travel, especially international travel that mixes sightseeing and biking. He is also involved in local trail maintenance activities.

30 Normalized Data Library

Mark P. Schnaible

Mark Schnaible joined HP's Colorado Springs Division in 1988 after receiving a BSEE degree from Ohio State University. He is currently a software engineer in R&D working on the HP 16505A prototype analyzer software and assisting with other projects. He recently was responsible for the prototype analyzer's normalized data library and was a major contributor to the overall architecture. Previously he worked on the HP 54600 series of oscilloscopes and is named as the inventor in a patent related to the user interface software used in the oscilloscopes. Mark is married and has three sons. In his free time, he has taught object-oriented programming using C++ at the University of Colorado at Colorado Springs. He also enjoys softball, basketball, skiing, golf, and family life.

38 HP OmniBook 5000 Computer

Timothy F. Myers

Tim Myers is a hardware design engineer at HP's Mobile Computing Division. He is currently responsible for the electrical engineering of new products for the notebook computer market. Recently he was the lead system engineer for the HP OmniBook 5000 computer. He has also worked on the electrical engineering for the OmniBook 425 and HP 75C computers including IC design and system manufacturing support. He is a member of the IEEE and is a licensed physical engineer in the State of Oregon. His professional interests include mobile computing and spread-spectrum communications on which he has authored several articles. He received a BSEE degree in 1979 from Montana State University and an MSEE degree in 1992 from Oregon State University where he was a teaching assistant. He joined HP's Corvallis Division in 1979, then left, returning in 1992. He has also worked at Fortune Systems as an engineering manager and at Corvallis Microtechnology as a vice president for product strategy. Tim was born in Fort Collins, Colorado. He is married and has two sons. His hobbies include boating, reading, investing, fishing, and exploring Oregon's mountains and beaches.

45 HP 38G Calculator

Ted W. Beers

Ted Beers is a software R&D engineer in the Corvallis Design Center of HP's Asia Pacific Personal Computer Division and is responsible for the next-generation handheld products for education and for maintaining the group's growing set of Internet communication tools. He recently worked on the HP 38G calculator including graphical user interface enhancements, the topic outer loop, and the aplet library. He worked on the HP 48G/GX, developing the graphical user interface. He co-developed the Program Development Link product, a PC development environment for HP 48S/SX applications. He worked on the HP 48S/SX, focusing on the parameterized outer loop, interactive stack, key handing system, user customization, and high-level display management. He also contributed to the directory structure and user memory organization for the HP 28S and the testing for the HP 28C. Professionally interested in user interface design and implementation, he is named as an inventor in a patent involving key action and has written five papers on the HP 48G/GX, the HP 48S/SX, and the HP 41. Ted was born in West Lafayette, Indiana and was awarded a BS degree in computer and electrical engineering in 1984 from Purdue University. As a co-op student in college, he developed an HP 41 calculator well log analysis module for Petrophysical Data Consultants, an oil consulting firm. He joined the Handheld Computer and Calculator Operation at HP's Portable Computer Division in 1985. He enjoys hiking with his wife and two beagles, gardening, philosophy, animal welfare, vegetarian cooking, home improvement projects, and science fiction.

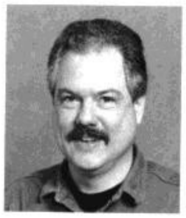
Diana K. Byrne

Diana Byrne is an R&D project manager in the Corvallis Design Center of HP's Asia Pacific Personal Computer Division. She is responsible for managing a team of engineers split between Oregon and Singapore who are working on handheld products for education. She was the project manager for the HP 38G graphing calculator. Previously she was the project manager for the HP 48G/GX graphing calculator and contributed to the plot and graphics code. She also worked as a software engineer on the HP 48SX scientific calculator for which she designed and implemented plot and graphics code and worked on the HP EquationWriter. She is professionally interested in the use of technology in education and has coauthored two papers on the HP 48G/GX and HP 48SX calculators. Diana was born in Portland, Oregon and earned a BS degree in mathematics with high honors in 1982 from Portland State University. She went on to receive an MS degree in mathematics in 1987 and an MS degree in computer science in 1988, both from the University of Oregon. Diana is currently enrolled in a Management of Technology masters program at the National Technological University. Before joining HP in 1988, she was a mathematics instructor at the University of Oregon and Portland State University. Diana has two sons. She commutes to work by bicycle, telephone, or airplane (traveling to the Corvallis or Singapore site). Her current hobby is studying the coffeehouse culture in the U.S. and Europe.

James A. Donnelly

Jim Donnelly joined HP's Corvallis Division in 1981 and contributed to HP 75 and HP 71 computer applications and the HP 14, 22, 32, and 42 calculators. He also worked on the HP 38G calculator and the HP Solve equation library card for the HP 48SX. Currently an R&D engineer in the Asia Pacific Personal Computer Division, he is responsible for the design of the next-generation handheld calculators and information appliances. Recently he worked on the HP 38G calculator focusing on the numeric views, matrix catalog, matrix editor, list catalog, list editor, I/O, and printing, among other areas. He has written three books about programming the HP 48 and a software simulator for Babbage's difference engine. Jim was awarded a BS degree in speech communication in 1979 from Oregon State University and worked on instrumentation software in the physics lab at the University of Oregon. Born in Chicago, Illinois, Jim is married and volunteers his time at the local community theater doing backstage work and set and prop construction. His hobbies include travel, photography, home machine shop, and gardening.

Robert W. Jones



Born in Mt. Vernon, Washington, Max Jones received a BS degree in mathematics from the University of Washington in 1976. He joined HP in 1982, writing documentation for the HP 41CX and HP 28 calculators. He also designed and implemented software for the HP 48SX and HP 48G/GX calculators. As a software design engineer, he is currently responsible for software development for handheld products. Recently he worked on the HP 38G home environment and user extensions to aplets. He is professionally interested in functional programming and rational approximation. He authored a paper on the HP 48SX calculator and is named as an inventor in a patent involving interface design using menu keys and keypress duration. He is a member of the Mathematical Association of America and the Society for Industrial and Applied Mathematics. Max is married and has two children. His hobbies include New Orleans rhythm & blues.

Feng Yuan

Born in Suzhou, Jiangsu in the People's Republic of China, Feng Yuan was awarded a BE degree in electronics in 1982 from the Shanghai University of Technology at Shanghai and a masters degree in computer science in 1984 and a PhD in computer engineering in 1987, both from Nanjing University at Nanjing. He became a lecturer at Nanjing University doing teaching and research, then moved to Singapore to work on computer-aided embroidery pattern design software. He joined HP's Asia Pacific Personal Computer Division in 1993. As an R&D engineer, he worked on the design and implementation of the HP 38G calculator, including the symbolic, note, and sketch views, sequence handling, statistical plots, external aplet examples, aplet development kit, and the Pascal compiler for the HP 48 and HP 38 calculators, on which he published an article. He also worked on the HP Palmtop FX computer, which is the Korean flash version of the HP 100LX computer. He is professionally interested in embedded system design, language processing and implementation, computer-aided design, and artificial intelligence. Feng is married and has one child. His hobbies include reading and playing the Chinese board game Go.

59 Creating HP 38G Aplets

James A. Donnelly

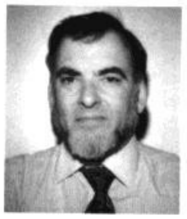
Author's biography appears elsewhere in this section.

64 HP PalmVue

Edward H. Schmuhl

Ted Schmuhl joined HP's Waltham Division in 1973. He is currently the program manager for the PalmVue product family at HP's Patient Monitoring Division (the successor to the Waltham Division). He is working with wireless systems that deliver medical information to clinicians via handheld computers. In his years at HP, Ted has held various positions within product development including software engineer, project leader, and manager. Some of his contributions have been the development of the architecture and software design for HP's first microprocessor-controlled patient monitor, management of the design and integration of HP's CareNet patient monitoring network, and project management of the HP PalmVue project. Before joining HP, he worked at RCA Corporation developing automated test systems for the military. He earned a BSEE degree from Tufts University in 1967 and an MSEE degree from Northwestern University in 1972. He is a member of the IEEE and is professionally interested in medical software and systems. Born in Summit, New Jersey, Ted is married and has one child. He is active in his church and community. An amateur musician, he plays trumpet in a local band. He also enjoys water sports such as sailing his 26-foot sailboat.

Allan P. Sherman



Born in Boston, Massachusetts, Al Sherman received a BSME degree in 1961 and an MSME degree in 1981, both from the Worcester Polytechnic Institute. He joined Sanborn Company, which was owned by HP, in 1964 and is currently a software development engineer at HP's Patient Monitoring Division. He was recently responsible for the palmtop software design and for the data handling and communication architecture for HP PalmVue. He continues to work on enhancements and extensions to the HP PalmVue system. Previously he worked on the mechanical design of electronic, recording, and analytical laboratory equipment. He developed HP's first computer-based patient monitoring system and designed the pressure processing algorithm used in these systems. He also provided software development for the early arrhythmia data management systems. He is named as the inventor in two patents on medical instrumentation and in three patents for blood pressure measurement techniques. He has authored two papers on arterial blood pressure measurement. Before HP, he was an engineering instructor at Western New England College in Springfield, Massachusetts. He is a member of the Boston Computer Society and teaches classes there. Al is married and has two children. His hobbies include computers, travel, and skiing.