

Add a GUI coprocessor

to improve GUI design efficiency

By Nicole Coleman
Marketing Manager, Amulet Technologies

The majority of us are touched by graphic displays on a daily basis. Our lives would be very different without these easy-to-use displays. In the very near future, it will be difficult to use a washing machine, microwave, home lighting system, or automobile without using some sort of graphic display. In addition, consumers are now accustomed to high-end graphics for their consumer electronics, thereby compounding GUI development complexity.

Due to development process inefficiencies, the addition of engineers to a GUI development program does not necessarily mean it will be easily completed. As a solution, embedded managers are now partitioning their GUI development programs by adding additional processors that replace one dedicated graphics processor. This allows the manager to split up their product developers into teams grouped by individual processor. In this article, Nicole advocates this useful GUI development method.

Development inefficiencies

Without proper partitioning, projects get so large that programmers have difficulties following the code, and therefore errors proliferate and debugging becomes inefficient. Studies have shown that as a project grows and additional engineers are added, productivity plummets. The problem is that for each person added, the personal communication channels increase logarithmically. This means that with only six engineers on a team there can be as many as fifteen channels of communication (Figure 1).

Fred Brooks, in his book *Mythical Man-Month*, discusses his studies at IBM and how the IBM mainframe System/360 OS project grew from 150 to over 1,000 employees and productivity soon plummeted. Time was quickly filled with meetings, memos, e-mails, and problem solving. Too many interruptions can kill development or prolong it for months. It is easy to see why about 80 percent of embedded systems projects are delivered late.

With products becoming more complex, there are thousands and thousands of additional lines of code to write. For this reason, embedded designers are integrating additional graphics microprocessors to increase design efficiency. By partitioning, project managers can split the developers into small teams where each team is responsible for their own graphics microprocessor.

GUI coprocessor solution

GUI coprocessors help partition the GUI development from the rest of the application, which greatly improves the efficiency of the overall development cycle. In addition, the dedicated GUI coprocessor incorporates a built-in graphical operating system.

With this approach, the developer has a complete set of tools specifically designed to manage the GUI, interact with the user, and control the LCD. This approach effectively splits the work load, while leaving the reliable legacy 8-bit microprocessor in place, fully leveraged, and virtually unmodified. In effect, the GUI functionality is bolted onto the existing code.

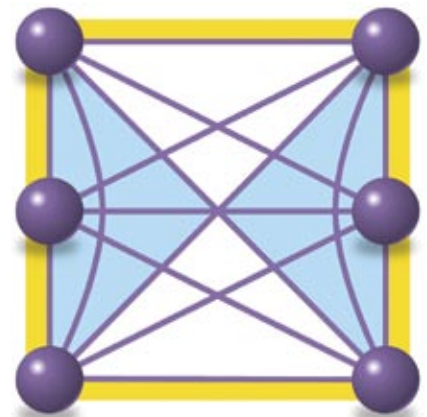


Figure 1

It is also well known that as user interface projects grow, GUI processes burden the host microprocessor by as much as 60 percent. Thus, the developer is forced to find a more powerful (and costly) processor to run the application and the GUI. By partitioning the GUI, the developer is able to use a smaller (and less expensive) host microprocessor to run their application. An example is shown in Figure 2 (on back).

Another huge added benefit is the possibility to develop the GUI concurrently with the rest of the product development versus having to wrestle with the GUI development at the tail end.

Amulet GUI coprocessor

The Amulet GUI chip is a combination LCD controller and user interface manager, handling all communications with the UI and the LCD. The Amulet CPU has task-specific opcodes for graphics rendering, I/O processing, and general purpose computing that enables the GUI kernel firmware to implement a highly efficient task scheduler. It is optimized to execute Amulet's GUI kernel and component-based GUI firmware.

The chip renders GUI pages containing graphic images, widgets, and other UI objects directly to the LCD, eliminating the need for complex code to draw each pixel on the display. It also allows for tactile interaction, which can enhance the consumer experience with the end product. Images and text are created in HTML, which is then compiled into highly compact micro-HTML that uses much less memory.

Increased efficiency

Smaller teams significantly reduce the number of communication channels. Imagine a team of six people working on the GUI design, software communication, and hardware. As shown in Figure 1, this results in fifteen channels of communication.

By partitioning the GUI using a coprocessor, the GUI design team can be reduced to two people. This reduces the lines of communication to only one channel, leaving the other four engineers with only six channels of communication for the software/hardware integration (Figure 3). We therefore have only seven total channels of communication as opposed to fifteen.

Increased collaboration

This approach to GUI partitioning also allows marketing managers, usability specialists, and graphic designers to play a direct role in the GUI design. Designing the user interface using HTML authoring software tools now enables non-engineers to assist in the design for the final product.

As the demand for visually appealing user interfaces increases, the complexity of the code in projects grows significantly, pushing out product launches for months. Now the marketing team is able to collaborate with the engineers to shorten the design cycle and integrate a user interface that is visually appealing, stands out from the competition, and is aligned with the look and feel of their other product lines.

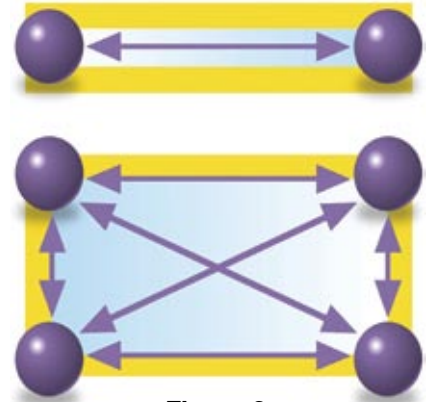


Figure 3

Rapid iterations

Simplifying the design cycle in this way allows for rapid iterations. The graphic designers can now make changes to the GUI immediately and have the ability to create multiple final designs to use on focus groups. The ability to refine the final UI design multiple times reduces the risk of customers not catching on with the product once it is in the marketplace. Both the marketing and engineering teams can go through GUI testing seamlessly.

Without proper partitioning using a GUI coprocessor, debugging becomes a huge issue. Each time a change is made, engineers have to start over with the process and devour loads of development time. Even a simple change may invalidate all previous testing. Not only does partitioning the GUI leave room for less errors and provide straightforward debugging, but iterations to the GUI design do not slow down the development process as when the host microprocessor is responsible for both the GUI and main application.

Summary

At some point in time during the development process, the host microprocessor may no longer be able to handle the GUI and the application. If this occurs, system costs will grow out of reach for the end customer to afford. So, the next time you are out evaluating displays and microprocessors, remember the many benefits of GUI development partitioning.

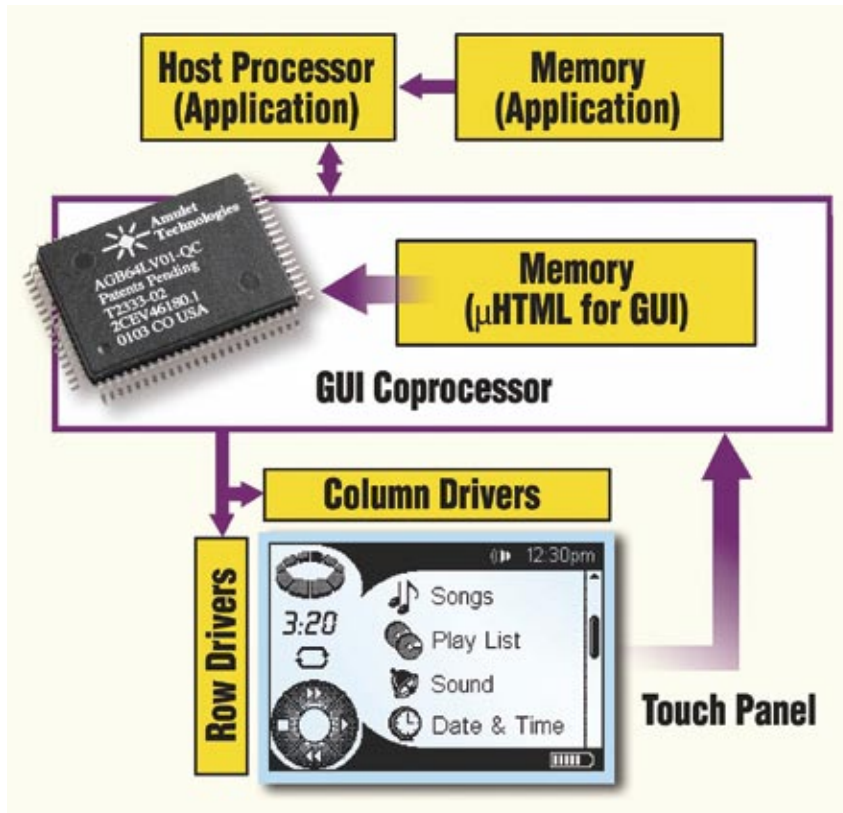


Figure 2


Amulet Technologies
 Empowering Flat Panel Displays

Amulet Technologies
 275 Saratoga Ave., Suite 230
 Santa Clara, CA 95050
 Tel: 408-244-0363 • Fax: 408-243-5457
 E-mail: Sales@AmuletTechnologies.com
 Website: www.AmuletTechnologies.com