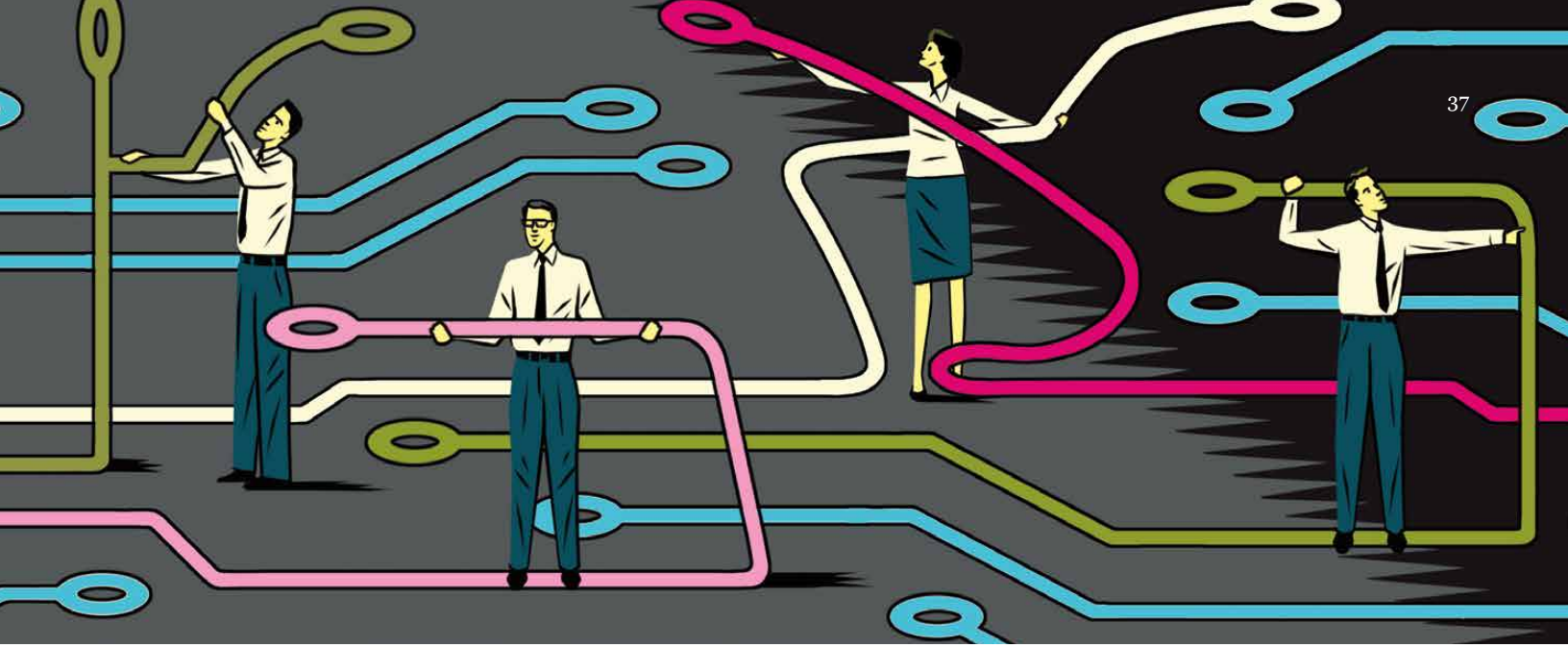

Bill Butcher

How semiconductor companies can get better at managing software development

They may want to consider adopting one of four basic organizational structures.

Gang Liang, Christopher Thomas, and Bill Wiseman

Software isn't just for purchasing or outsourcing anymore. Increasingly, semiconductor companies are exploring in-house software development as a way to reduce costs, improve time to market, and differentiate themselves from competitors. Organizations that have traditionally been focused only on hardware (silicon wafers and circuits) are hiring more software engineers to support the development of the integrated circuits that are at the heart of most consumer electronics, medical devices, automobiles, appliances, and pieces of heavy equipment in use today. In the late 1990s, it was common for chip makers to invest in one software engineer for every ten hardware engineers; today the ratio is closer to 1:1 or, in some cases, 1:1.5.

Chip makers are pursuing software development primarily to meet customers' growing demands for more sophistication in and more support for the components they buy. Indeed, the leaders in various sectors of the semiconductor market—Intel, MediaTek, and Qualcomm, among others—now routinely earmark significant portions of their R&D budgets for software development, and some are already providing end-to-end software-based products for customers. MediaTek, for instance, provides both software and reference designs with its chip sets so that customers can create their own branded mobile phones with basic features. Its end-to-end offering has helped customers cut their own time to market and R&D costs and has allowed them to focus more

on strategies for branding and differentiating their products.[1]

But while some market leaders have been at this for a while, other players are only just starting to take a closer look at how they build, use, and manage software (see sidebar, "What are they building?"). They face a number of challenges: software resources at hardware-oriented companies tend to be limited, and engineering talent can be scarce and hard to acquire and retain. Additionally, selling silicon is the main business, so efforts to divert scarce resources toward software development may meet internal resistance.

Semiconductor companies that choose to pursue a rigorous software-development program will need to have the right organizational structure in place—one that enables companies to motivate talent, control the R&D budget, launch products more quickly, and meet customers' expectations. Some semiconductor companies have established a central software organization to support all of their business units, while others are struggling to keep up with "rogue" development efforts happening within various business units, each of which has its own software team. Our work points to four potential organizational structures that software-minded semiconductor companies may want to consider adopting to get the most from their existing development efforts and to make it easier to pursue new software R&D initiatives: completely decentralized, completely centralized, hybrid, and leveraged. There are advantages and potential pitfalls associated with each, and the appropriate structure will differ for every company based on its existing talent, resources, and overall business objectives.

Four ways to organize

We have seen 1,000-person companies make the transformation from one organizational structure to another within 12 months, but a years-long effort is much more typical, particularly if the company is starting from scratch or having to make hard decisions about which groups to merge. In either scenario, a change in metrics and mind-set will be required. Executives will need to develop mechanisms for tracking the productivity gains from their software R&D, and they will need to foster engagement and commitment to software development across the company.

Completely decentralized. Semiconductor companies with a number of different business units that have little or no business or technical crossover likely would find it easiest to pursue a completely decentralized software organization: each of the business units funds and manages its own software group, and the unit's general manager retains the autonomy to deploy software resources where needed. In the 1990s, Intel's architecture business unit boasted a dedicated software organization that created homegrown development tools to support its x86 systems. Even today, the software group works closely with a number of third-party software vendors—Oracle and SAP among them—to optimize those applications for every generation of its central processing units. Intel also had separate software groups dedicated to its NOR Flash and i960 businesses. The NOR Flash software team built up a strong capability in device drivers, and the i960 software team focused on enabling Intel silicon to work well with third-party software and applications. There was almost no overlap in customer bases or operations among those business units.

The completely decentralized model works well so long as the units' businesses and technologies remain independent—which, in today's semiconductor environment, is quite rare. If, for instance, units are combined or new businesses emerge and need the same fundamental software and technologies being developed and managed in other groups, it makes less sense (operationally and financially) to duplicate efforts. One large integrated-chip manufacturer, for instance, had created separate handset and tablet business units as each of those technologies emerged in the marketplace. There were separate software-development groups for each unit, but the company eventually realized that development teams in both used the same system on a chip, which meant the company was wasting its resources and needlessly creating conflict and competition between two groups of engineers.

Completely centralized. For semiconductor companies whose business units rely on all the same technologies, a completely centralized software organization will be most efficient and effective. Under this organizational structure, software-development and technological expertise radiates from a central group—one that reports to the C-suite—removing potential redundancies and significantly reducing resource and development costs. That is the case for one large US components vendor. It provides integrated circuits to manufacturers of a range of consumer electronics, including laptop computers, mobile phones, tablets, and other devices. The underlying graphical processing unit and other technologies embedded in its chips are standard, so one version optimized for Android is suitable for all its customers. Having one centralized software group allows the company to better manage all its licensees and reduce development costs.

What are they building?

At the start of the shift toward in-house software development, many semiconductor companies were focused primarily on developing their own firmware—software embedded in their integrated circuits that would dictate how the chips would function. Over the past few years, some have started working directly with operating-system vendors to make sure their device drivers will work seamlessly and their processors will perform optimally in those environments; still others began to release software tools (compilers, debuggers, tuners, and the like), plus common libraries and middleware, so third parties could create optimized applications for their company's chips. Most recently, semiconductor companies have started to create end-to-end, embedded software products for original-equipment and original-device manufacturers.

A completely centralized model also confers upon semiconductor companies other benefits, including a consistent approach to R&D planning, a standard set of software-development and management tools, a common software-development process and methodology, and comprehensive rules and standards for assuring quality and appropriately managing source code. By establishing this level of consistency across all business units, chip makers can reduce their R&D costs and accelerate growth in new and critical businesses that may not otherwise have the funding or technical capabilities to pursue software development as a complement to their existing work. This centralized structure also may facilitate offshore expansion or development outsourcing—the strategic moves favored by many semiconductor companies these days—by making it easier for them to manage global engineering resources or maintain relationships with vendors.

There are a few drawbacks, however. For instance, the funding model for this approach can be complicated. In many companies that use a completely centralized model, the business units pay a "tax" based on their needs, financial strength, and other criteria. This can be a headache for the finance team, which has to calculate the difference between projected needs and actual demand for each business unit—each of which would obviously want to pay as little of this tax as possible. Additionally, under the centralized model, the business units would have less control over software development as a resource. Often, the objectives of the centralized software group and the business unit will not be completely aligned; the units may have unique requirements that a centralized organization simply may not be aware of. For example, video playback on a mobile device and videoconferences over the Internet both use the same H.264 video codec (or compression software), but the implementation of H.264 in each case is quite different. A centralized software group could easily deliver the common video codec but likely would not have the technological expertise to support its implementation on both mobile and Internet platforms.

It is critical for companies that adopt a centralized model to pay attention to process, metrics, and collaboration—for instance, convening a small team that represents the interests of each of the business units and the centralized software group. The team would meet regularly to analyze software priorities and rank them according to business units' needs and the impact of certain projects on the company overall. It is also good practice to establish service-level agreements between the centralized software group and business units to help clarify roles and responsibilities—and to preserve some level of control for the general managers involved.

Hybrid structure. This organizational approach combines the financial and technological efficiencies provided by a centralized software group with the greater flexibility and controls that a decentralized structure may offer the business units. At first glance, it seems to present the best of both worlds. In reality, there are significant funding and operational challenges to address. Under the hybrid model, the technologies and software capabilities that are common to all business units become the property of a centralized group, while the technologies and software that are unique to a particular business unit are maintained and developed separately. One leading

maker of mobile chip sets, for instance, has a centralized software group that is charged with enabling and optimizing the Android operating system for use with the company's generic system-on-a-chip architecture. But each of the company's business units "owns" a version of this technology that it uses in ways that are specific to its group.

Besides just holding on to the common software, the centralized group should also establish best practices for its use and encourage sharing among all the other software teams within the company. To that end, a joint committee should be convened to manage common software-development priorities, and service-level agreements should be drawn up. But as with the completely centralized model, a charge-back process must be established; the use of common technologies would be subject to a tax based on revenues, profits, or other criteria, and each business unit's software organization would be required to fund its unique development initiatives separately.

Leveraged structure. Many semiconductor companies have a core business and a number of units that are derivative of the core. For instance, one chip manufacturer's core business is in microcontrollers and microprocessors, primarily for the automotive market, but increasingly its technologies are also being used in medical and consumer applications. For it and companies like it that are exploring market expansion, a leveraged software organization may make the most sense. Under this structure, the software group would report to the core business unit rather than to a centralized corporate team. As with the hybrid model, the software organization would own the completed software components and resources but would deliver them to the rest of the company. For instance, the software team in the chip manufacturer's consumer-products business unit could take technologies developed by the software team in the company's automotive unit and modify them to suit its and the market's needs. As with the centralized model, the core business's software group would need to establish best practices in software development and encourage sharing across the organization, but the other business units would have to fund their own unique development initiatives.

Which model?

To determine which of these structures is best, companies need to consider their existing software capabilities—that is, the type of software R&D they are currently undertaking (if any), their overarching objectives relating to software, and the funding and other resources at their disposal. They should also consider their competitors' software capabilities.

It is unlikely that many companies would pursue a completely decentralized model; this type of structure just isn't the norm in today's semiconductor environment. But the companies that already have lots of software R&D experience, or that have a core business unit with several businesses feeding off of it, will want to explore hybrid or leveraged models. The individual business units would immediately benefit from software technologies that are already in hand (managed by the centralized software organization), but they would retain the flexibility to create unit-specific products based on their unique technical and business needs. Such companies could realize less duplication of effort and waste. By contrast, the companies that have limited software R&D experience may want to set up a centralized software organization focused on just one business or a few business units at first—starting narrow

to ensure that success is within reach but establishing best practices that can be rolled out more broadly as software-development initiatives gain momentum.

These decisions won't necessarily be permanent; as semiconductor companies move from a single-minded focus on developing silicon components to a broader focus on delivering end-to-end offerings built around their integrated circuits, their software organizations will need to change as well. In the transition from one model to another, executives may need to introduce key performance indicators and other metrics to help the software organization (however it is structured) quantify the impact of its development efforts and to help project leaders set and meet personal targets. Because of the global scarcity of technical talent, semiconductor executives also may need to adjust some of their human-resources practices—for instance, providing attractive, high-profile assignments in which software experts actively participate in product design and planning, or letting software engineers lead higher-level strategy discussions. Most important, executives who are bringing software R&D in house will need to become steeped in basic software terminology and concepts. They don't have to be experts, but gaining at least a rudimentary understanding of what the software can and can't do may help them achieve their business objectives in the long run.

• • •

The software-development function in most semiconductor companies typically flies under the radar—until growth slows and executives with cost cutting in mind notice the large cadres of engineers they've acquired over the years or until a new business opportunity emerges and executives notice how few engineers they have on staff. We believe executives need to be more proactive; they need to recognize the complexity and collaboration associated with software development and react accordingly.

[1] *Android Authority*, "MediaTek is riding high, how far can it go?," blog entry by Simon Hill, April 25, 2014, androidauthority.com.

The authors would like to thank Harald Bauer and Ondrej Burkacky for their contributions to this article.

Gang Liang (Gang_Liang@McKinsey.com) is a senior expert in McKinsey's Boston office, **Christopher Thomas** (Christopher_Thomas@McKinsey.com) is an associate principal in the Beijing office, and **Bill Wiseman** (Bill_Wiseman@McKinsey.com) is a director in the Taipei office.