

Embedded in Thin Slices

# Build an Embedded Systems Consulting Company (Part 6)

## Tradeoffs of Fixed-Price Contracts

COLUMNS

Continuing his "Building an Embedded Systems Consulting Company" article series, this month Bob gets into some important aspects of marketing your company. He explores the nature of contracts and how fixed price contracts can be an effective, albeit dangerous tool in marketing.

By  
**Bob Japenga**

One of the greatest challenges of starting a new embedded systems consulting company is marketing your skills. A lot of individuals do this by leveraging their previous employment contacts. They do consulting for the companies they left. This is an effective strategy early on in the life of the company. We did this a little and it helped bridge some gaps in our income early on. But eventually, every consulting company has to break new ground with new customers. Certainly, networking and Internet marketing are important to your future success. A lot of ink has been spilled about using these techniques to advance your company's future and we won't rehash all that.

But in this highly competitive world, it is not enough to have people click through to your web site using Google ad words. It is not even enough to be given a good referral from a supplier that has seen you bring successful products to market using their products. When a potential customer looks at your web site or talks to you on the phone what are they looking for? Certainly experience, longevity, stability and availability are all extremely important when you're being vetted. But there are a lot of other companies with all of those traits. What will cause this new potential customer to choose you?

One differentiator that we have found very useful in marketing our services is offering the customer a fixed-price contract for the services they have requested. They

come to us with a problem and we provide a solution for a known price and a schedule we are committed to. Let's look at the upsides and the downsides of this marketing tool in attracting new customers.

### **FIXED-PRICE UPSIDES**

From the customer's perspective, it is extremely attractive to begin a new project knowing upfront what your costs will be. Most engineering managers have experienced massive cost overruns when developing new embedded systems products. They have taken the heat from their investors or their bosses about how costly the new product has become. Coming to them and offering to complete their project at a fixed price helps alleviate those situations. In many respects, as consultants, it is our job to make our clients look good to their stake holders: bosses, investors and stock holders.

Another blessing of offering a fixed-price proposal to a potential customer is that it forces both you and the client to get a lot in writing up front. You can get in writing as much detail as possible about what you are going to do—a statement of work (SOW). And you can also put in writing what the product is going to do—a specification. Let's admit that most of us don't like writing these things down. I cannot tell you how many projects we have come upon with virtually none of that written down. Very often we get called in to clean up a mess created by another embedded systems consulting company. It staggers

my imagination that someone would hire a consulting firm and not have a SOW and a specification. But it happens. All the time. But if you are going to bid a fixed price contract for a complex embedded systems project, you darn well better know exactly what your customer is expecting you to do and what they are expecting the product you are designing will do. If not, you won't be in business very long. Those two documents—the SOW and the specification—need to be crafted carefully. We will talk more about how to write a statement of work and a specification next time.

## FIXED-PRICE DOWNSIDES

*You Can't Make Money on Fixed Price Projects:* Many years ago, I worked for a small aerospace firm which designed avionics for commercial and military aircraft. One of the things that the founder and president of that company told me was that you cannot make money in development. In the aerospace industry at time, cost-plus contracts long gone. A cost-plus contract is where get paid your cost plus a profit. You could make money on cost-plus development. But now the industry demands fixed price contracts. Our founder knew that you have to make your money in making the hardware. This was hammered home to me in the seven years that I led a team of engineers developing sophisticated avionics hardware and software. To illustrate how hard headed I am, I left that company to start MicroTools where we do fixed-price development for a living—knowing full well that you cannot make money doing development. So that is the first curse of doing fixed price contracts—you cannot make money at it over the long haul. MicroTools has a 29-year history of that.

*Increased Complexity in the ICs and Tools:* The second curse of fixed price development of embedded systems is brand new for us. The building blocks we're using for creating these products have become almost unmanageably complex. When we started out, we did a lot of development with the Intel 8051 (**Photo 1**). It was a great workhorse for us and we used it in a lot of products. The datasheet, the architectural document and programmer's guide / instruction set totaled 133 pages. In contrast, today's state-of-the-art devices are order of magnitudes more complex, and in turn so is their documentation. For example, the Microchip PIC32 microcontroller that we are using in one of our designs has a datasheet that is 736 pages long; a reference manual that is 1138 pages long; an architectural and instruction set document that is 244 pages long; and an errata that is 16 pages long. Oh, and did I mention that the IDE documentation is over 13,000 pages long?



**FIGURE 1**

The early Intel 8051 from the 1980s has a datasheet, an architectural document and programmer's guide / instruction set that totaled 133 pages. In contrast, today's state-of-the-art devices have documentation that is orders-of-magnitude longer.

Imagine that you have a small team of three engineers on your project. Let's say that they only read one half of the more than 15,000 pages of documentation. And let's say that they can read a page every 5 minutes. That is 625 hours per person or, with three people, almost a man-year just to read the documentation. And that's just for the microcontroller. If you added an Ethernet PHY chip, that has another 166 pages of documentation. And let me say from experience, you need to read this documentation. By not reading it, you put your entire project and possibly your company at risk.

*Increased Bugs in the Silicon:* Not only is the hardware becoming more complex, it has become much more bug-filled. These days, there are paths in the silicon that are not even tested. How can I say that? Both mathematically and from experience. For example, a microcontroller configuration tool we used allows the designer to select from a plethora of options. And the combinations and permutations of choices mathematically creates more combinations than are possible to test in a lifetime. From experience on a project this year, we found that in a particular configuration, when an internal reference voltage was turned on, it creates a low out-of-specification voltage on the internal bus. This causes the hardware to fetch an illegal instruction intermittently—it might happen in 5 minutes or it might happen in 5 days. This illegal instruction can happen at any address and was completely impossible to diagnose. We worked with the field applications engineer who worked with the factory. All in all, it took us over 500 hours to find out what was causing the problem and then create a work-around that we could live with. The budget for the entire hardware design? Five hundred hours! One bug ate the entire budget.

On another project this year, we used a Wi-Fi chip that did not have a stable Linux driver. Eventually vendor put this chip on the "Not recommended for new designs" list. But we spent hundreds of hours debugging





**FIGURE 2**

Knowing the total costs of a project ahead of time can backfire. The Apollo program cost over \$200 billion, according to NASA's 2009 assessment and adjusted for inflation. If the American people had known that at the time, would they have fully supported landing on the moon?

the software and hardware only to find that it was a defective design. Finally, another project this year used a chip whose compiler was changing so often that we could never count on stable code as bugs were fixed in the compiler. Fixing one bug would cause another problem to surface.

**Certification Demands:** When I first started out, I worked for large companies. I remember products going through Electromagnetic Compatibility (EMC) testing at the end of our projects. I was not involved in this phase at this time. I just remember products suffering delay after delay. I remember one consultant after another being called in to help us pass EMC. Cost overruns were the norm. This is a major risk in fixed-price projects. In this day of embedding cell modems in everything from trash cans to toilets, cell certification costs can be wildly different based on how many turns of the board you need to do; how many times you have to go back to the lab; and how long the schedule drags out.

**The Proposed Cost is Too High:** Most

of us drastically underestimate just about everything we do that has any amount of complexity. And we often wonder if we would have done the project if we knew it was going to cost this much. If the American people knew upfront that the Apollo program (**Photo 2**) was going to cost over \$200 billion dollars (according to NASA's 2009 assessment and adjusted for inflation) would they have supported landing a man on the moon so enthusiastically?

In the same way, if you calculated the total expenses perfectly of a project that is going to go on to be a wonderful success, most companies would not launch the project. This is probably the biggest downside of fixed price proposals. Your job is to leverage this to both your advantage and the customer's advantage. But let's face it. If it costs \$500,000 more to create a product that will be generating twenty million dollars in revenue each year, is it worth it?

### THREE STRATEGIES

Let me propose three strategies to help reduce sticker shock on a fixed price project:

**Incrementalism:** This is where you break the project up into manageable chunks, allowing the product to be test marketed in each successive phase. In one case, we deferred some of the large development costs of designing our own ARM9 board by incorporating off-the-shelf components that were more expensive. The customer had less margin on the initial 10,000 units. But by then he had broken into the industry and made a name for himself. He was now much more willing to pay the much larger cost of development to increase his margin. Would it have been more cost effective to design his own ARM9 board from the start? Absolutely! But I don't think he would have pulled the trigger on the high cost of developing and certifying his own proprietary board. Overall, this approach cost him more. But it got him into the game. This approach doesn't work on all projects but it can help get the client over the sticker shock of what it's actually going to cost him for development.

**Return-on-investment (ROI) analysis:** Of course, as a contractor, you cannot tell your client that his budget is too low. But if you are serious about this business, it will be your job to help educate your client as to what it actually costs to develop a product. I cannot tell you how many calls I get from individuals and companies who think they can develop their invention for under \$100,000. "Let's see: You want to sell about 100,000 of these per year for \$35. The injection molds alone will cost almost half of that \$100,000." That's the education part of it. But you can't just leave


the client discouraged and dejected. If you can get the client open to the discussion, this is where you help the client see how much they will be able to make once they start shipping 8,000 of these systems a month. If they haven't done the ROI analysis, do it for them.

*Eat it:* This is by far the most unpopular strategy. But we have used it over and over again. A product is going to cost one million dollars to design and develop. The company has a great idea that you can get behind. Your design is going to be incredible. You bid the project, not at what it's going to cost, but what the customer is willing to pay. You eat the rest.

"Bob, are you nuts?" I hope not. You have to do this only in cases where there is an opportunity for on-going work. And where you are confident that this client has the integrity and the product that you want to invest in. Let me give one example. We spent 9,000 man-hours developing a product for a company. At our normal going rate, we would have charged them more than \$1.3 million. We charged them \$350,000. I am convinced that they would never have turned on a \$1.3 million project. But in the last ten years they have generated more than \$6 million in revenue for us in on-going work. I am

convinced there are times and places where this is an effective but risky strategy.

## CONCLUSION

We have looked at some of the advantages of offering a fixed-price contract as a marketing strategy in running an embedded systems consulting firm. We have also looked at some of the very serious risks associated with this strategy. Next time we will look at some ways to make fixed-price contracts work. But of course, only in thin slices. 



## ABOUT THE AUTHOR

Bob Japenga has been designing embedded systems since 1973. In 1988, along with his best friend, he started MicroTools, which specializes in creating a variety of real-time embedded systems. MicroTools has a combined embedded systems experience base of more than 200 years. They love to tackle impossible problems together. Bob has been awarded 11 patents in many areas of embedded systems and motion control. You can reach him at [rjapenga@microtoolsinc.com](mailto:rjapenga@microtoolsinc.com).