I spent the summer after I graduated from University researching for Bryce Himebaugh, a Clinical Assistant Professor of Computer Science at Indiana University in Bloomington. He was the Professor in one of my embedded systems classes, and he developed all of the course work for the systems classes I took at IU. He’s an incredibly experienced and knowledgeable guy, and he was nice enough to let me interview him for this site!

Question: What is his background before landing in his current role as a Clinical Professor at Indiana University?

During his Undergraduate degree in computer engineering he did a Co-op for Intel, which he highly recommends because of the practical work/engineering experience that you gain while you’re still a student. During this time working for intel he spent time supporting Integrated Circuits that were being sold to Delco, which is an automotive electronics design and manufacturing subsidiary of General Motors as well as other companies that consumed many intel chips. He mentioned that Co-op work is not very glamorous, but this time is where he was introduced to people doing interesting things in the embedded systems space. Ultimately it led him to his first full time industry job for Cummins who designs power generation products, mostly engines.

He started his full-time career with Cummins electronics, a Cummins subsidiary, as a hardware designer for the module that manages the fuel flow to the cylinders in an engine. This job entailed helping write the specifications, designing the hardware, implementing the board through all of the stages of production, and delivering a module to the main engine company. From an embedded systems perspective after the specifications were written he would design and verify the board, as well as write software to ensure that it was working properly. The software he wrote was to bring up the board for the first time and ensure it was working as expected. He would write tests to simulate vehicle speed and verify the module measured it correctly, fire injectors and ensure the wave forms looked correct, and write tests to ensure the hardware (such as solder joints) continued to function as they stress tested the module by running them through heat cycles. After this position he did another year of hardware development for another engine team before joining an engine development team where he was in a systems engineer type role. He was responsible for the module, as well as its harness, and sensors. This job is more wholistic because he was responsible for everything that went into the engine program, as it combines more elements of engineering at a higher level of abstraction.

After his experience at Cummins he came to Indiana University where he joined the university as a designer. At the time they needed engineers to design systems for student projects and for academic research. He was creating systems that would be used in classes similar to the systems classes I took at IU where we developed software for hardware, the hardware systems themselves, and analog computing devices to be used for various research purposes. He mentioned that he always finds his way back to doing development because that’s what he enjoys the most, and it shows because he did this for 10 years. This lead to an opportunity to lead a merger of the IT, between the Informatics and Computer Science departments which he led, while he continued to do research on modes of analog computation and low power neural networks.

The research that he did on analog computation and low power neural networks, led him to patent some ideas and receive some capital to launch a startup venture. He cofounded **Analog Computing Solutions, Inc**. which developed an integrated circuit for fast low-power signal processing for the medical industry to be used with prosthetics, hearing aids, and other applications. They ran this company for two years before running out of capital, while he continued to teach systems classes at IU before he landed in his current role as a Clinical Professor for Computer Science and now Intelligent Systems engineering, the new school at IU.

Question: In your time doing development, you’ve seen the advent of The Internet of Things(IoT) and the revolution of embedded systems. What was it like when it started? And where do you see it going now that wireless technology is a lot more prevalent?

Then:

Bryce mentioned that when he started it was before the advent of flash memory. This means that they had to burn EPROM just to get a program onto a microcontroller, which severely limited the kinds of systems that could be developed. For those of us who are young programmers/engineers/makers: **EPROM** stands for erasable programable read-only memory. It is a form of non-volatile memory that can be erased and rewritten by exposing it to ultraviolet light. You can recognize these kinds of chips because of the fused quartz window on the package that is needed so the UV light can reprogram the chip. It is much more cantankerous than plugging in a board and typing make download, compile, or even compared to loading a program with a debugger and a JTAG if I do say so myself! When he moved to the automotive world designing modules for engine control units they had flash, but it was external. This means that it wasn’t on the die with the chip like it is today. Because this memory was external, they had to go to the trouble of mapping the external memory to the chip which is a lot of extra work.

Now:

Currently, we have processors and communication peripherals that dwarf what used to be used when it comes to power and usability. Bryce looks to see low power communication protocols being integrated in the way that protocols such as SPI and I2C, or even ethernet are being more commonly integrated on boards today. He expects wireless transceivers for these low power protocols to be integrated on boards which will have the affect of being cheaper and lowering the power constraints of doing IoT designs and builds. He hopes that a network that is IoT available grows to become as prevalent and integrated into our infrastructure as Cellular. Today we could be anywhere in the everyday world and expect to be able to send a text, make a call, or query Google. This infrastructure is important since Cellular is not feasible for IoT because its use is predicated on a direct financial relationship with the provider. He says, “it’s intractable to have 1000 sim cards for each of your sensors in the field.” He gave **SIGFOX** in france as an example of his desire to see a more feasible IoT infrastructure, “It is a low bandwith IoT specific network that’s subsumed into the infrastructure similar to [ethernet] or the power grid.” He did mention that he realizes someone, conceivably tax payers, foot the bill for the implementation of such infrastructure, but in removing the direct financial relationship with the carrier it makes placing many new kinds of autonomous sensors in the field feasible on a scale that is not currently possible.

Question: What words of wisdom do you have for the person who is just getting started playing with hardware?

Bryce recommended joining one of the learning communities based on a popular platform such as Arduino, Mbed, or Raspberry pi. He said, “The cost of these development kits have become very affordable, the impetus to buy a few hundred dollar development kit is out of the way.” Using these platforms particularly Arduino and mbed you can compile, download, and debug for free. He said, I’d start in these ecosystems and leverage the community as much as I can to both learn how to program and build some real systems.“ These communities and platforms are so powerful and have so much open source knowledge that you can build a real system almost immediately. He recommends this approach because that is where you are going to learn the most. Try it, actually do it. You’ll run into all kinds of problems, and it’s those problems where you’ll learn the most. Being a beginner developing hardware, the worst-case situation is where you are in a custom platform where you cannot separate whether the issue is your hardware that you built or the software that you wrote. The ambiguity between those two problems require a lot of experience to work through. A major reason why it is better to be on one of these popular platforms because the probability that someone has encountered your problem or a similar one is high. One of the powers of leveraging these platforms is that you will probably be able to figure out what you’re doing wrong on a forum, and if not, there will be someone with the experience you’re lacking to help you diagnose your mistakes.

Question: You’ve seen the startup realm. You and I have talked a lot about the value proposition and how that is above all the most important part of the startup game (particularly when it comes to getting funded). What other skills or information is critical for people eager to join this domain?

When questioning him I mentioned a value proposition. This was a topic that he would often come back to when we would discuss his time in the startup community. Bryce and his partner developed a sophisticated integrated circuit for the applications they were targeting, and he would often times tell a story from a conference when this issue cropped up. Him and his partner in the startup met a guy at a startup conference for some Venture capital function when they met a guy who created a prototype that turned off your stove when you had left the house or moved a certain distance away from the kitchen. In reality his hardware was little more than a simple system you could throw together with an Arduino and a Bluetooth module with a web app, but because he had an iron clad value proposition describing what the customer could expect their product to do and why they need it he got the funding he needed and then some.

On this occasion the advice that Bryce shared the importance of having a minimum viable prototype. This concept entails having enough of a model of your system to probe the design constraint space in which you are working.Many times you could have identified a problem or misinformed assumption that arises while creating or using this prototype very early and pivot to a different, more suitable direction. This provides you the capacity to invest your design and development time, whether it be software or hardware, elsewhere. Design and development time is precious in the startup world because you are almost always constrained by resources. Additionally, don’t be afraid of creating a startup because it might fail, and also don’t be afraid of starting a subsequent startup because you’ve had one fail. You will learn so many lessons from the process irrespective of the ultimate outcome!