Hacked The Badge

- Home
- How it works
- **The Story**
- Make your own
- The Team

Motivation

DEFCON is a wonderful thing. Perhaps the most interesting group of people I've ever meet. After a 16 hour drive to reach DEFCON, my friends and I got into the badge line and waited. Once we all had our badges it was off to staples to get memory cards.

The default functionality of the badge is cool and it was fun playing with it. When I found out what the badge did, I agreed it was a file transferring device, but not over IR. The IR is cool and its fun to screw with TV's but I wanted something more interesting. The idea for the DC16 Web-badge was born.

Our good friend Joe Grand was kind enough to drag TX/RX pins onto the 6 pin connector (SCI2), and connected to LED's 3 and 4 (SCI1). This small act is what makes the web badge possible as a completely stand alone solution. The original attempt made to create the web badge used the USB port. The main problem with with this solution is it tied the web badge to a computer to work, or writing complex USB drivers. SCI is an extremely simple interface that just about everything talks.

I attempted to create a working version of the web server while at DEFCON, but it proved to have too many pitfalls. Being silent while the awards were giving out I decided that I'd carry on and make the web badge a reality. Digging through my box of random electronics, I found some Digi ConnectMe from my old job. Now I had all the pieces I needed to make a completely stand alone solution. The work began.

The first limitation I set on myself was to require any DEFCON'er with a working badge to duplicate this solution at some level for less than $10 bucks. Although its not possible to create a fully stand alone version for that much money, it is very possible to use your computer as the serial-to-IP converter.

Since you have to pay for the full version of code warrior if you use more than 2^15 bytes, I required my code to be smaller than that. This proved to be very difficult. Most of the DC16 code had to be removed and the amount of logic that could be put into its place was limiting. Even so, the trimming process was completed and the whole web server solution was implemented in less than 500 lines.

Pit Falls

- SCI Baud Rate - When you get into simple chip's like the one on our DC16, you lose a few simplifications. Baud rates to me always consisted of: 9600, 38400, 115200 etc. With the DC16, the only way to create baud rates is through an equation based on the clock rate. To figure out the correct time I wrote a program that create a z pattern with the 8 bits. The clock rate increased as it printed out the pattern. Watching this on the computer with a known baud rate I found the correct

clock rate and that all the bits needed to be inverted. Luckily this chip has a built in feature for that.

- Power Supply Noise - The first wall wart I used to power the web badge created a huge amount of DC noise. So much that I had a 8% error margin, way way way too high. One of my good friends let me use his oscilloscope which uncovered the problem. The near term solution was a whole bunch of AA batteries and later, some better wall warts.
- Communication Hanging - The original code in the DC16 has a problematic line in the SPI interface. It is possible for the code to block until the end of time waiting for a reply that may never come. Also, on the SCI interface to the outside world the same problem was true. To fix this I implemented a simple but affective timeout solution that ran off one of the chip's interrupts, this combined with some simple code changes protected the code from hang ups.
- Communication D.O.S. - The first few versions of the IP-to-Serial code allowed for some very simple denial of service attacks. I had to completely revamp the code to create a queue system with time outs. D.O.S. is still possible but you have to want to take this guy down, instead of trying to honestly use it and taking it down like before.
- Bad solder joints - The badge I got from DEFCON had faulty solder joints on the SD memory card's holder. As you can imagine this cause a lot of wasted time. I finally found a low powered iron and some thin solder, retouched all the joints and solved the problem.
- Floating ground - Lets face it, computer science guys that play with electronics make some dumb mistakes. I made several. My crown bone head maneuver was to create a system with no common ground. During my testing I had the USB interface plugged into my computer and using that same computer to talk over the serial interface. This of course creates a common ground, everything worked fine. When I started testing things on the Digi, I had the USB plugged in, which gave me a float ground, but a ground reference. Without the USB plugged in, the thing wouldn't work at all. I chased this in code for literally days until one day I accidentally touch the badge to the Digi's ground wire and it worked for a second. I though longer about it, realized what I had done and rewired everything. This is proof that its always a hardware problem. Which brings me to one of my favorite jokes. How many software guys does it take to screw in a light bulb? None its a hardware problem.