# The DIY Artificial Pancreas: Hacking Wetware with Open Source Software and Hardware

## Introduction

Diabetes, a disease that affects nearly 200 million people worldwide, is a deficiency in the pancreas to properly produce insulin (Type I) or a deficiency in the body to absorb it (Type II). Insulin is a hormone required to sustain human life, so people with this disease must obtain it from another source and administer it according to a complex formula involving carbohydrates, physical activity, sickness, and preexisting insulin levels to straddle a tenuous line between their blood sugar rising too high or dropping too low. Levels that are too high or remain high for too long through carbohydrate intake take years off your life – although extreme high levels can cause stroke, seizures, organ failure, brain damage, and death over a period of hours or days. Drop too low through exercise or insulin use and, after several minutes progressing through symptoms that include anxiety, tremors, sweating, dizziness, confusion, and heart palpitations, you die. Both of these scenarios are unfortunately very easy to achieve because throughout the day your body requires a combination of carbohydrates, which raise your blood sugar, and insulin, which lowers it. Your body gradually digests different carbohydrates at unpredictable speeds so if your body processes the appropriately matched amount of carbohydrates more slowly than the accompanying dose of insulin begins to work, that too can cause an unsafe drop in blood sugar. It’s a difficult balance to maintain consistently, so many patients choose to allow their blood sugar to remain higher than it should be over time.

Sensitivity to insulin can change over time, is different in every person, and the disease requires constant 24/7 management. It’s clear that this could be managed much more effectively with technology. A blood measurement called A1c is used by doctors to determine how high or low a patient’s blood sugar is on average over time and is a good key indicator we can use to determine the success or failure of our technology strategy. As a side benefit of constant data collection, we can also measure this against overall time in proper blood sugar range while using the system. Ordinarily this is not a measurement that is easily acquired so the A1c is used as a proxy, but because of a sensor discussed below this becomes easy for patients.

Some technology exists to make it easier to maintain some form of control over blood sugar, but they largely go toward easing the burden of monitoring and managing symptoms. None of these solutions take a decision-making role in management of the disease. Where people used to give themselves multiple injections of different kinds of insulin per day, people now have the ability to use an insulin pump to both slowly dose background insulin throughout the day and to give themselves larger (but riskier) doses required at meals. The insulin pump is worn for days at a time, eliminating the need for multiple painful injections throughout the day and reducing a lot of medical waste. Whereas most patients still must use some finger sticks to draw blood to check their blood sugar, giving them only a point-in-time measurement that lacks a lot of real-world context like trend direction, there now exist continuous glucose monitors (CGM) that stay attached to you for days at a time, taking a reading every five minutes and exposing trends to help you manage your disease. The CGM receiver will also alarm when your readings venture outside of pre-set parameters so you can do things like wake the patient up at night to take needed action to raise or lower their blood sugar back to safe levels.

## Hardware

It must be restated that none of these components take action on behalf of the patient and a human is still required to be in the loop, so we need to develop a solution that takes the data from the CGM, processes it, predicts future effects, and delivers a command to an insulin pump to do something other than deliver regular, unchanging background insulin. This requires us to overcome four technical challenges:

* We need a small, low-power and portable platform that the user can easily bring with them through their day to do all of this.
* A visualization solution that allows us to monitor the system both historically and as it works in real-time is required to monitor the status of the solution.
* We need to collect blood sugar readings from an FDA-certified CGM system.
* We need to be able to query and control an FDA-certified insulin pump.

Older pumps that have wireless connectivity allowed the manufacturer to use a proprietary protocol to communicate with the pump. After this was discovered and turned into a Black Hat talk in 2013 with an emphasis toward how this could be misused, manufacturers significantly limited how this functionality could be used on newer pumps. This wrinkle requires us to find and purchase older, used pumps and to use custom hardware and software.

The Raspberry Pi, particularly the Pi Zero, and Intel Edison are ultra-portable platforms that make phenomenal solutions on which to run our code. Further, each has built-in Bluetooth and Wi-Fi making it easy to connect them to known networks or tether them to a cell phone when not at home without needing extra attachments. We can collect CGM data over Bluetooth without having to plug the manufacturer-provided receiver into the board, although that remains an option with USB OTG. Our CGM and platform problems solved, the Open Source hardware community stepped in and created two solutions to the pump problem. The Explorer Board for the Intel Edison and Explorer HAT for the Raspberry Pi are 900 MHz radios that while advertised as general-purpose radios communicate with the insulin pumps using the proprietary protocol that was reverse engineered by a community member over the course of several years.

## Software and Safety

Our software solution will be a stack that includes Linux, Node, Go, Mongo, Python, and Bash. Nightscout is a purpose-built visualization and web-based data warehousing platform built on Mongo and Node designed specifically for remotely monitoring the well-being of a diabetic, whether yourself or those under your care. Each person builds their own Nightscout instance on Heroku or Azure and it works well within the free tier plans for those services but you can run it on private infrastructure if you have that option available to you. Nightscout is good for point-in-time information about how things are now, but because it also saves all of the data it receives it serves as a great reporting tool that can be used to generate charts and reports to bring to routine medical appointments. Nightscout will act as our monitoring mechanism but we can also use it to give our solution some insight into what we’re doing so it can use that information to make treatment decisions.

The Open Artificial Pancreas System, or OpenAPS, is a Node, Python, Go, and Bash suite of components that runs on Linux and is designed to acquire CGM data, process it using an algorithm that has over 10 million person-hours of use under its belt, and deliver a dosage command to an insulin pump. While patients can only be reasonably expected to do this process themselves several times per day, OpenAPS processes data and can potentially deliver a dose every five minutes – including, most importantly, while the patient is sleeping. Patients can normally be expected to wake up at least once or twice every night to alarms requiring action, but because OpenAPS works to keep the patient in range without intervention sleep is one particular area of life that improves dramatically almost immediately. Because a sleeping patient isn’t eating it’s easy for the system to maintain them within their blood sugar range and they get that in-range time on their A1c measurement without any work on their part as well.

While under development, users and maintainers have been building the car while driving it. Safety is a critical factor in the way changes and features are implemented because it’s important that this doesn’t cause someone to be hospitalized. Several checks and double checks are made before making dosing decisions, redundant equipment and fallback behaviors are defined, and features that may impact the algorithm are scrutinized heavily. Should the OpenAPS device fail due to being out of range of the CGM or pump or due to sudden power loss, the commands delivered to the pump are constructed in such a way that cause the pump to revert back to normal, factory defined behavior. If this happens during sleep, for example, the manufacturer-provided CGM receiver is used as a safety check to sound an alarm if the blood sugar levels go out of a defined range. One indirect safety factor that was discovered through use of the system makes sense when thought to its logical conclusion: unlike normal care routines that operate by making large drastic changes that can’t be undone in both carbohydrates and insulin intake, making very small changes every five minutes allows the system to self-correct after just five more minutes if the situation should suddenly change. This is key during meals: rather than take the entire insulin dose at the beginning of a meal and hoping the body doesn’t process the insulin faster than the carbohydrates and causing low blood sugar, OpenAPS mimics the pancreas’s normal function of slowly dosing insulin as the body processes the carbohydrates and keeps the patient’s blood sugar level. Because there is a lesser amount of insulin in a person’s system at any given time overall, the likelihood that they experience a potentially deadly blood sugar drop is lessened as well.

## Iterative Improvements, Results, and Live Demo

Over the last three years I’ve integrated the hardware components and software packages described above in several implementations that range in size from large, requiring a sizable bag modified to strap the components down securely to the inside, to small enough to fit unnoticed in a pant leg pocket. Each iteration and choice of component reduced the size, improved portability and battery life, and over time as I contributed to the OpenAPS project improved compatibility with FDA-certified devices and reduced friction around installing the system. For maximum flexibility, the systems work without requiring a connection to the Internet when one isn’t available. When Internet connectivity returns the device drops all of the data from the time it spent disconnected into Nightscout to aid in later analysis.

If my talk is accepted, I would describe all of the above and bring several of the devices I’ve built to demonstrate how single board computers, miniaturized as they have become over the years and available to the average consumer, enables anyone to design complex systems with significant life improving benefits. After identifying and explaining the different components of each iteration as they got smaller and smaller, I will walk through my tools live, what I do and don’t have to do anymore, the improved medical chart data achieved over this time with much less manual labor on my part, and show a live implementation that will have been working throughout the talk.

## Conclusion

Activity around this community hasn’t gone unnoticed by medical companies and the FDA, and now after years of stagnation some movement around new solutions has occurred. The FDA is getting ready to approve componentized, interchangeable diabetes devices that don’t need to go through the long and expensive recertification process with each potential provider company and in at least one case it is expected to approve a hybrid system that is the first and closest analog to what the community has built. There is still a long way to go in improvement to access to insulin and access to this technology in a way that is affordable and sustainable, but the hackers who were tired of waiting and who have both built this solution and spurred action by the government and medical companies deserve kudos for motivating them after many years of complacency.