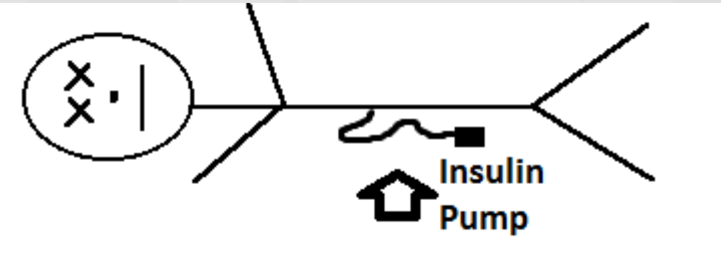
Hacking Medical Devices for Fun and Insulin: Breaking the Human SCADA System

Jay Radcliffe
jay.radcliffe@gmail.com
@jradcliffe02

Can “evil” people hack medical devices?



Me (AKA Dad)

Why Diabetic Devices?

- On my 22nd birthday I was diagnosed with diabetes
- Being a geek, I have a huge assortment of devices to "help" me with my condition
- Defcon 2009 – Parking Meter hacking

Type I Diabetes

- When a person loses the ability to produce insulin
- Rather then the pancreas producing insulin, a person has to administer a synthetic insulin as replacement
- Sounds simple, but there is no magic formula
- Infinite number of variables (Stress, Time of Day, Physical Activity, Illness, Fiber, Fat content)

How Diabetes Works

- Non-PWDs have a blood sugar between 90 – 120
- The liver and pancreas work together to control these levels
- Pancreas produces insulin, which is used to process the sugar into energy (for use or storage)
- Liver holds a sugar (glucose) reserve that can be used if levels get too low

Normal Sugar Relationship

- Normal person eats a Snickers bar (32g Carbs)
- As that sugar enters the blood stream, pancreas produces insulin to match that quantity of sugar to allow the conversion to energy
- Sugar levels might jump 20 “points” as insulin takes effect

Abnormal Sugar Relationship

- PWD eats a Snickers Bar (32g Carbs)
- Diabetics have an equation Amount of Insulin per Grams of Carbs (Mine is 1U Insulin / 10g Carbs)
- Ideally, Insulin is given at the perfect time and mimics human insulin, keeping sugar levels stable.

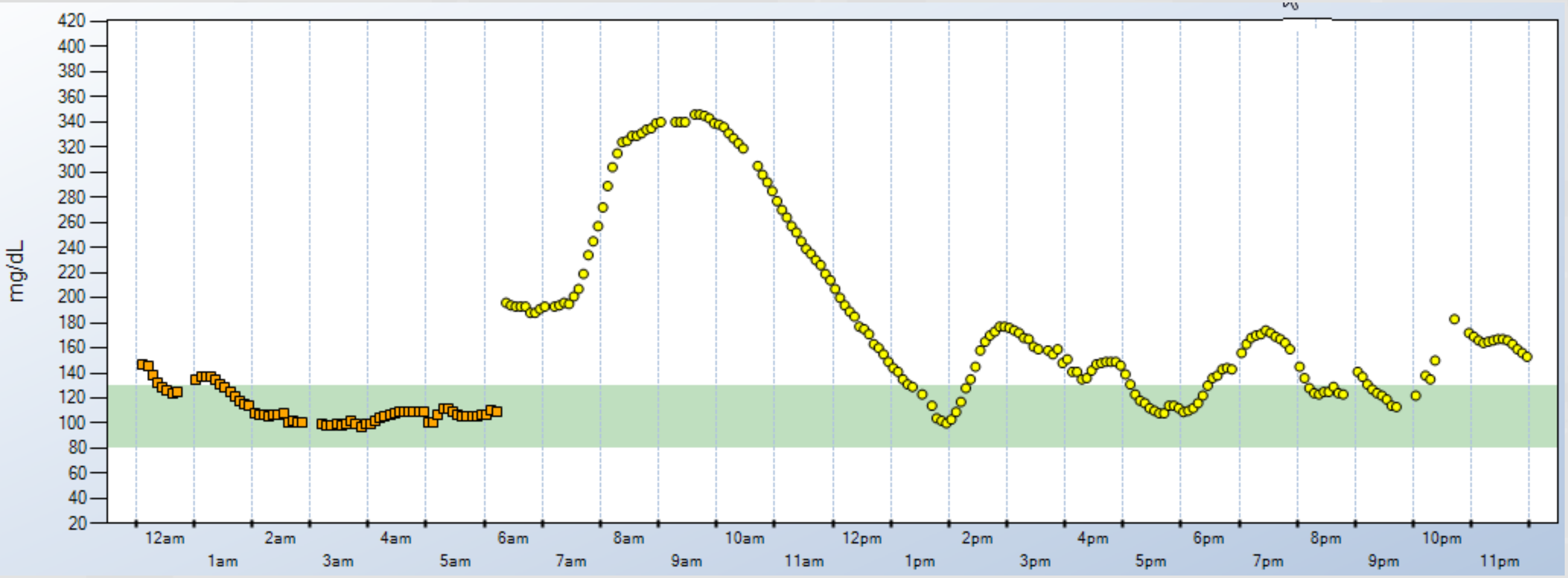
Abnormal Sugar Relationship – No Insulin

- If no Insulin is administered, blood sugar has a huge spike (In my case, 200+ points within 40 minutes)
- Sugar can not be processed into energy, body does two things
  - Filters sugar out through the kidneys. Very stressful to kidneys. Extreme Thirst.
  - Body switched to fat for energy. Also very stressful, causes ketosis potentially ketoacidosis.
- Headaches, blurry vision, long term kidney damage

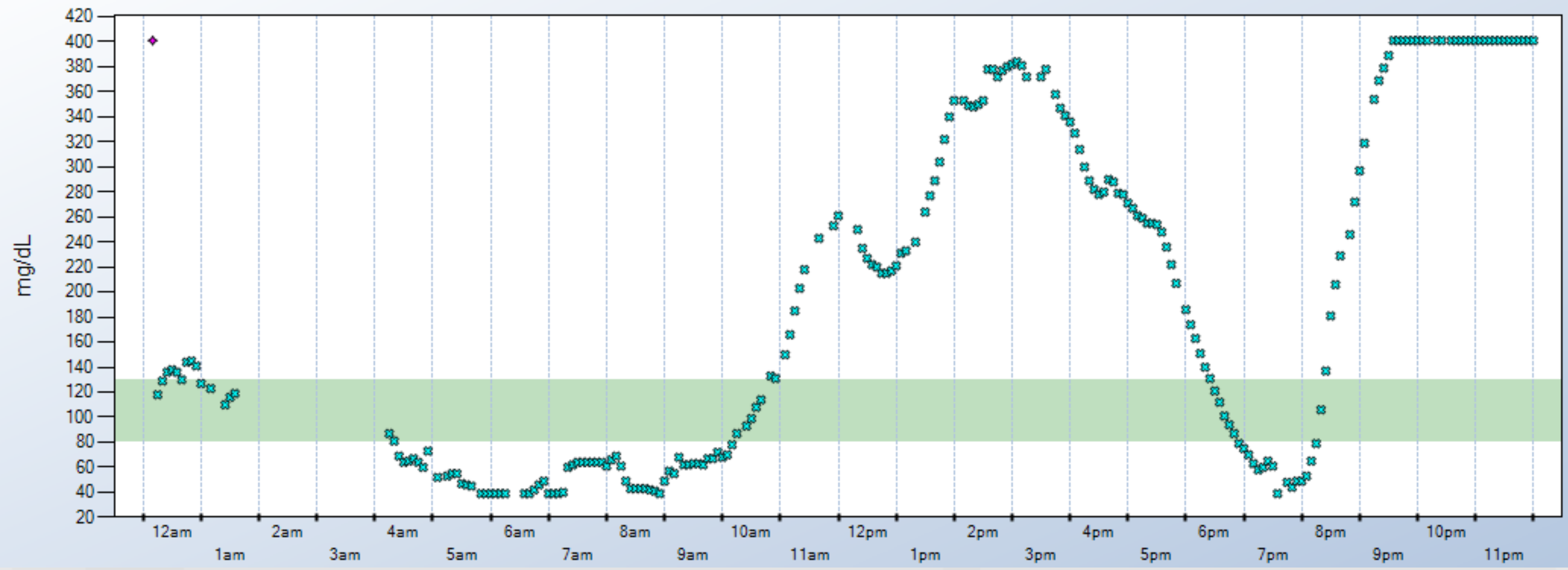
Abnormal Sugar Relationship – Overdose of Insulin

- If too much Insulin is given, blood sugar can crash to under 60
  - Heart and Brain run on sugar exclusively
  - Body starts to shutdown, conserving available sugar to respiration and heart
  - Starts with: Sweating, loss of fine motor control, shaking hands, overly drunk feeling
  - Uncorrected leads to coma, respiratory failure and death
- Some diabetics lose the ability to feel these symptoms

A Good Day

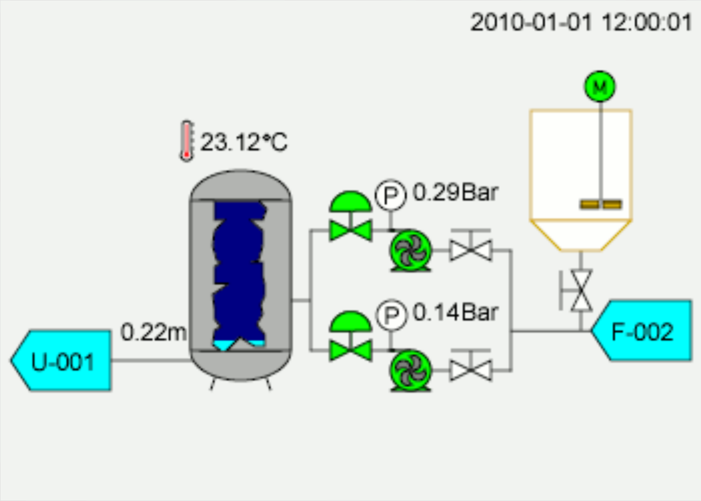

mg/dL
420
400
380
360
340
320
300
280
260
240
220
200
180
160
140
120
100
80
60
40
20
12am
1am
2am
3am
4am
5am
6am
7am
8am
9am
10am
11am
12pm
1pm
2pm
3pm
4pm
5pm
6pm
7pm
8pm
9pm
10pm
11pm

Bad Day


mg/dL
420
400
380
360
340
320
300
280
260
240
220
200
180
160
140
120
100
80
60
40
20
12am
1am
2am
3am
4am
5am
6am
7am
8am
9am
10am
11am
12pm
1pm
2pm
3pm
4pm
5pm
6pm
7pm
8pm
9pm
10pm
11pm

Human Chemical Plant

- Body is like a complex chemical plant
- Relationship between pressure and temperature of chemicals just like insulin and sugar
- The SCADA system monitors the pressure, and adds or removes heating to keep pressure constant.
- Pressure gets too high = BOOM
- Pressure gets too low = water delivery failure



Human SCADA System

- Similar to water, diabetics monitor sugar levels and adjust insulin and food intake to control levels
- Sugar too low? Drink fruit juice or other sugar foods
  - Hard to precisely measure amount of carbs/sugar consumed
  - Could take hours to process total amount of sugar
- Sugar too high? Adjust insulin or wait
  - Fast acting Insulin lasts 3-4 hours in human system, can not be removed. Not easily adjusted
- Frustrating never-ending manual process

Two Technologies – One Purpose

- Continuous Glucose Monitors (CGM)
  - New Technology
  - Small wire in tissue to measure electrical elements of fluid
  - Graphs sugar values over time
  - Huge leap forward

- Insulin Pump
  - Delivers insulin in 2 ways
    - Basel: Every 3 minutes
    - Meal: At Mealtime
  - Delivered through tubing attached to body
  - Tubing replaced every 3 days

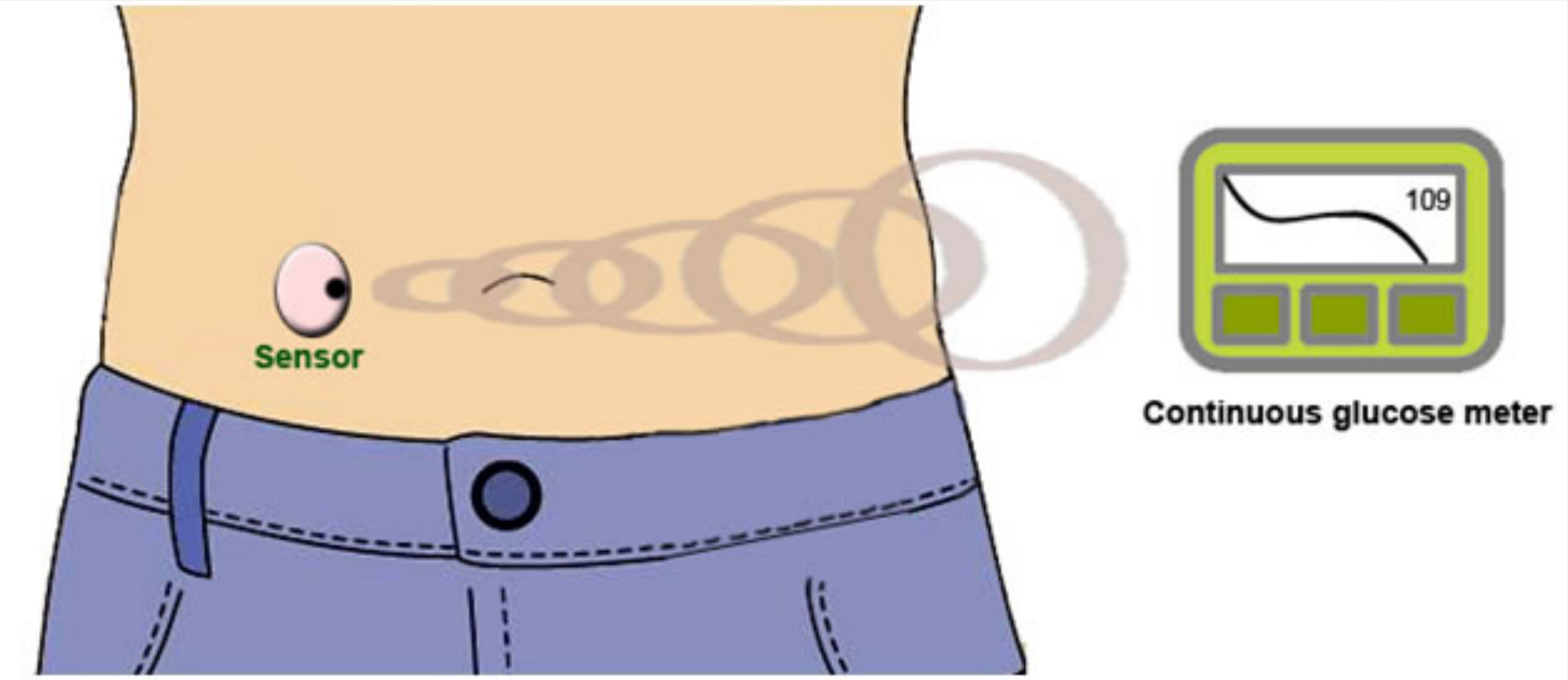
Diabetic Pressure Gauge

- Pre-Tech: Urine Tasting (Yes, for real)
  - Very imprecise, gross, no synthetic insulin
- Early 80's Home Test Kit
  - Blood test, poke finger, get value
  - Live Demonstration!
  - Accuracy varies (10-15%) Cost = $0.75 - $1.25 per test
  - No contextual information (direction/history)
- Still most common used method

Deeper into CGM systems

- Mid-2000's advent of Continuous Glucose Meters
- Measures resistive value of interstital fluid to measure sugar levels
- Wireless Sensor attached to special wire inserted into tissue
- Needs blood testing every 12 hours to calibrate, sensor lasts 7 days (Per FDA regulations) $40-70 per sensor

CGM



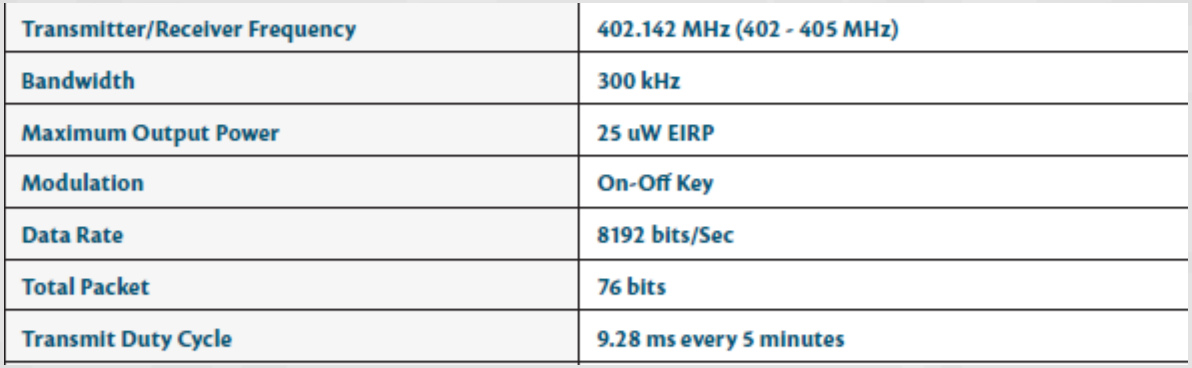
Picture from Armozyme.com

CGM Recon Work

- Hypothesis: CGM wireless results are transmitted with little to no security. These results can be vulnerable.
  - Sensor runs on 1.5v “watch” battery for 2 years. Crypto would require more horsepower (200k+ transmits)
  - Non-bidirectional communication. Sensor has no knowledge of what is receiving the data
  - Sensor is unaware of time or sequence numbers
- How do we verify this?

CGM Recon Research - Manual

- First, read the manual
  - RTFM: FCC Disclosure



Transmitter/Receiver Frequency	402.142 MHz (402 - 405 MHz)
Bandwidth	300 kHz
Maximum Output Power	25 uW EIRP
Modulation	On-Off Key
Data Rate	8192 bits/Sec
Total Packet	76 bits
Transmit Duty Cycle	9.28 ms every 5 minutes

  - Small Transmission (9ms, 76 bit packet)
  - Sounds like: 🔈
  - No ACK back, confirms beacon

CGM Recon Research - FCC

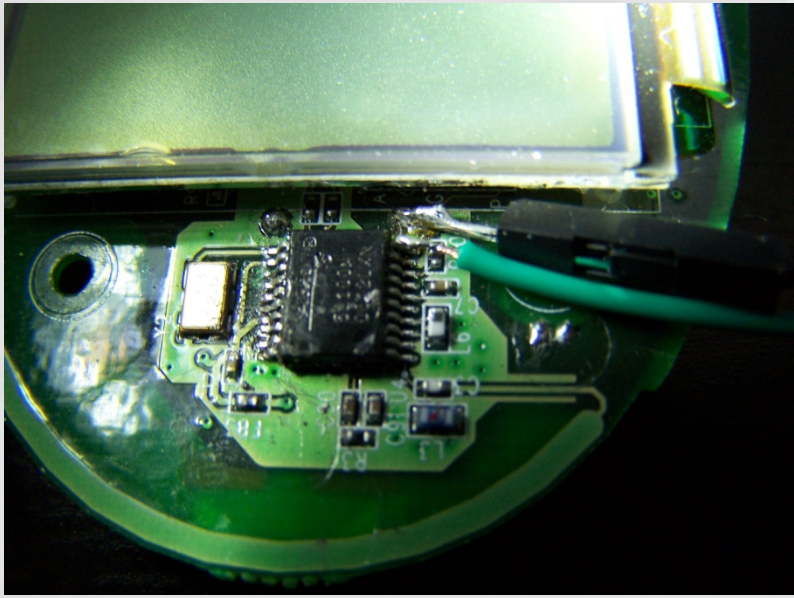
- All RF Transmitting devices go through FCC testing and verification
- FCC issues a TX ID for all devices
- Very Detailed Report. Screen Caps from Spectrum Analyzers, Oscilloscopes
- http://transition.fcc.gov/oet/ea/fccid/

CGM Recon Research - Patent

- When companies file a patent, documents are published that show how the device is made and it's function
- http://www.freepatentsonline.com
- Very detailed on operation of devices

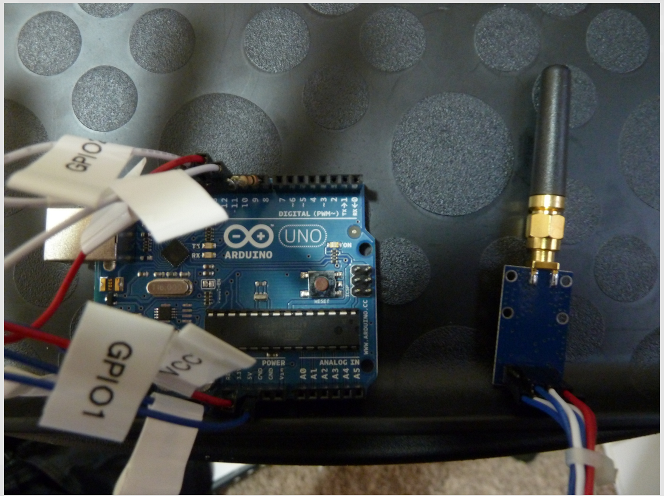
CGM Recon Research – Unboxing

- Taking a CGM Apart
- AMIS 52100M Chip
- Antenna Visible
- Out Of Production Chip
- Datasheet has good hints
- Same chip used in ICS environments (ICS/SCADA)



CGM – HandsOn How to Listen

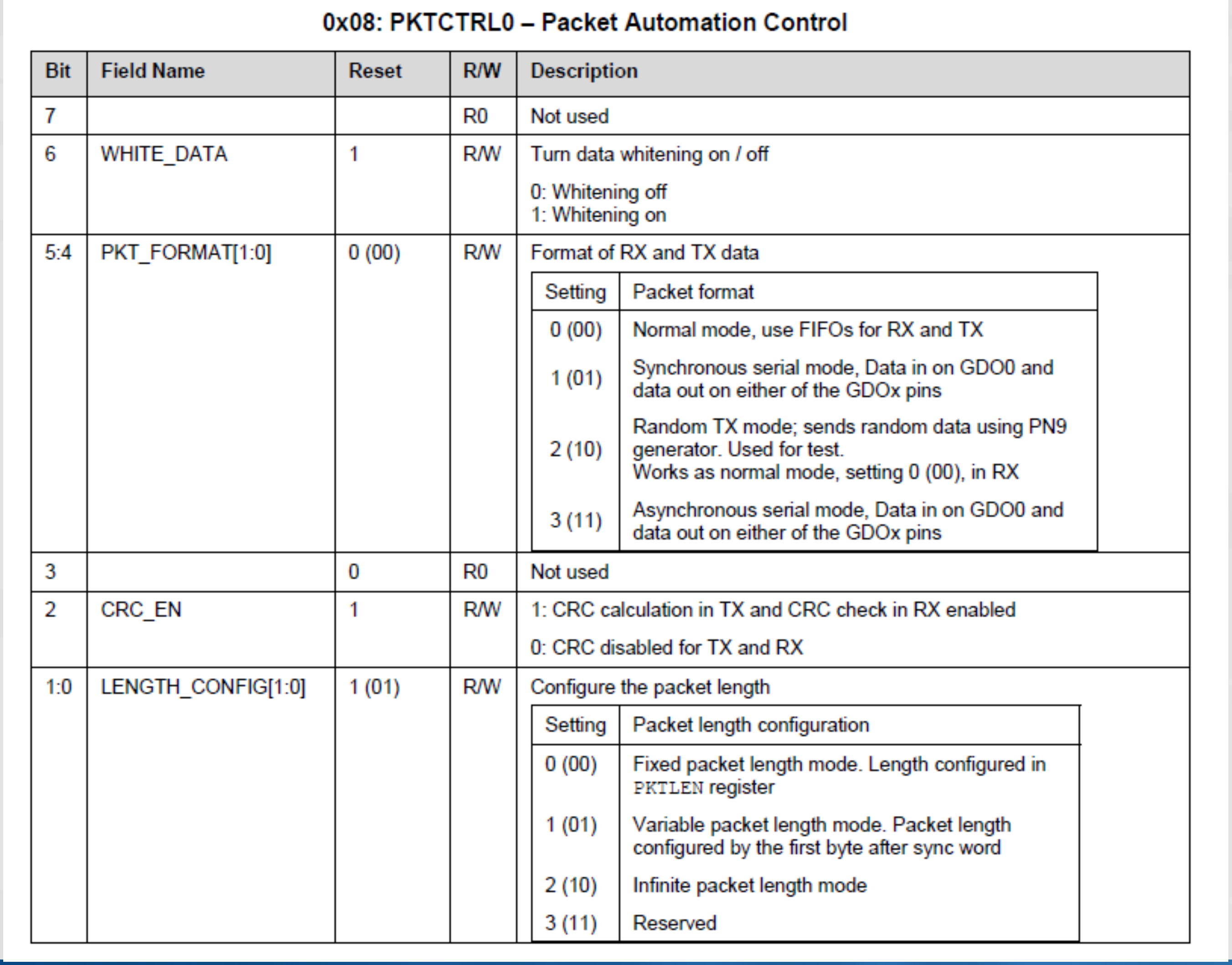
- Ardunio Based Solution
  - Ardunio is a hardware based platform that has RF modules that it can use
  - RFM22B by HopeRF /CC1101 by TI
  - Cover 300mhz – 900mhz (sub-1ghz)



CGM - HandsOn Ardunio Problems

- First, hard to program. Registers have to be set according to the manual, all in binary/hex notation.
- Example: Register 0x08 Packet Control
- 8 bits of data in the register

CC1101 Register Example



0x08: PKTCTRL0 – Packet Automation Control

<table>
<tr><th>Bit</th><th>Field Name</th><th>Reset</th><th>R/W</th><th>Description</th></tr>
<tr><td>7</td><td></td><td></td><td>R0</td><td>Not used</td></tr>
<tr><td>6</td><td>WHITE_DATA</td><td>1</td><td>R/W</td><td>Turn data whitening on / off
0: Whitening off
1: Whitening on</td></tr>
<tr><td>5:4</td><td>PKT_FORMAT[1:0]</td><td>0 (00)</td><td>R/W</td><td>Format of RX and TX data
<table>
<tr><th>Setting</th><th>Packet format</th></tr>
<tr><td>0 (00)</td><td>Normal mode, use FIFOs for RX and TX</td></tr>
<tr><td>1 (01)</td><td>Synchronous serial mode, Data in on GDO0 and data out on either of the GDOx pins</td></tr>
<tr><td>2 (10)</td><td>Random TX mode; sends random data using PN9 generator. Used for test.
Works as normal mode, setting 0 (00), in RX</td></tr>
<tr><td>3 (11)</td><td>Asynchronous serial mode, Data in on GDO0 and data out on either of the GDOx pins</td></tr>
</table></td></tr>
<tr><td>3</td><td></td><td>0</td><td>R0</td><td>Not used</td></tr>
<tr><td>2</td><td>CRC_EN</td><td>1</td><td>R/W</td><td>1: CRC calculation in TX and CRC check in RX enabled
0: CRC disabled for TX and RX</td></tr>
<tr><td>1:0</td><td>LENGTH_CONFIG[1:0]</td><td>1 (01)</td><td>R/W</td><td>Configure the packet length
<table>
<tr><th>Setting</th><th>Packet length configuration</th></tr>
<tr><td>0 (00)</td><td>Fixed packet length mode. Length configured in PKTLEN register</td></tr>
<tr><td>1 (01)</td><td>Variable packet length mode. Packet length configured by the first byte after sync word</td></tr>
<tr><td>2 (10)</td><td>Infinite packet length mode</td></tr>
<tr><td>3 (11)</td><td>Reserved</td></tr>
</table></td></tr>
</table>

CGM - HandsOn Ardunio Problems

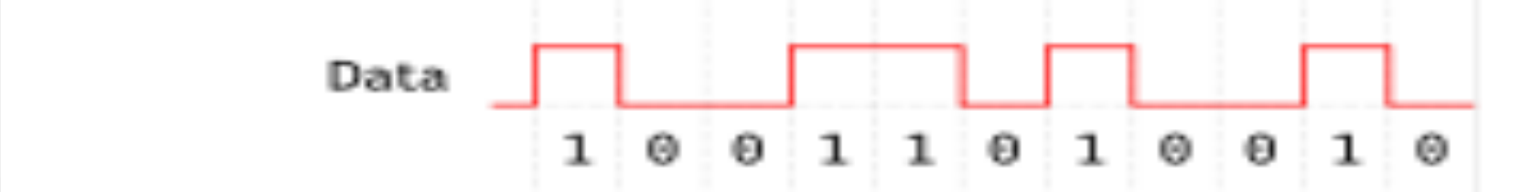
- Even after you determine the register settings, you have to set them
- Little to no verification that value has taken
- Lost 2 weeks to this
- Thought I was writing to register, turns out none of the register values were being changed
- Zero indication of that

CGM – HandsOn Hardware is Different

- First Real difference between Systems/Computer world and Hardware World
- Hardware is very concerned with cycles, so much of the hardware code I saw did little to no verification of it's actions. If I had this issue with a perl program or shell script I would have gotten an error.
- Note: We see a lot of exploits and vulnerabilities based on this concept in software. Buffer overflows due to not verifying boundaries (strcpy). Can this be the case in hardware land?

CGM – Signal Dissection

- What is On-Off Keying? (AKA OOK)
- Simplest form of RF Modulation



- Pure Binary, no signal = 0, signal = 1. Very fast
- 8192 bits / sec * 9ms = ~ 76 bits

CGM – Signal Dissection

- Next problem: The RF module wants to know certain parameters of the transmission
  - Preamble: This is a series of binary 1s and 0s used to indicate that data is going to be coming shortly
    - Used to “warm-up” the RF gain settings and to wake from a power saving mode
  - Sync Word: Think of this as the secret word. Set of characters that assure that the transmission format is correct.
  - CRC/CRC Location: This is usually 8 bits at the end that are used to make sure there is data integrity.

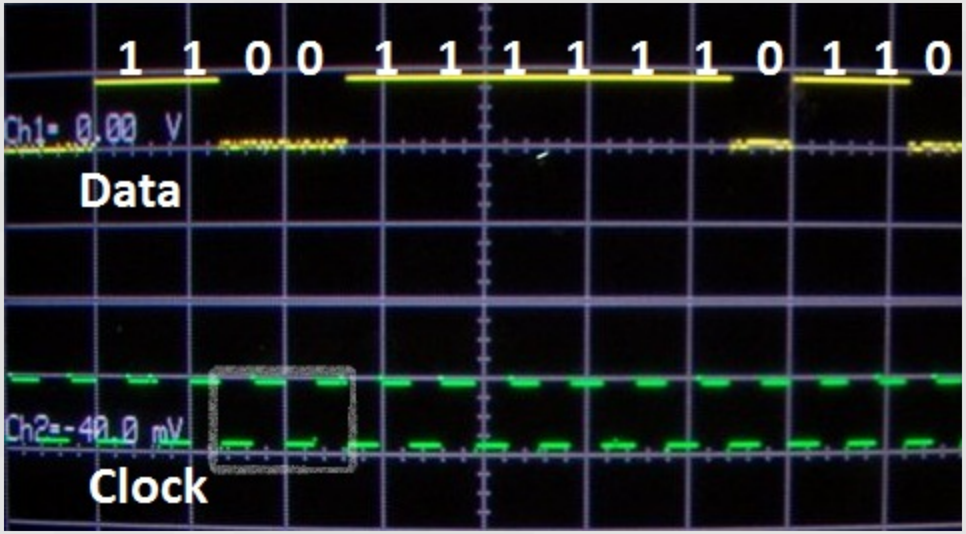
CGM – Signal Dissection Example

- TESTING TESTING 31337 12345 15
- |______Preamble_____||_SYNC__||__Data__||_CRC_|

- If 31337 is not received, RF Module ignores it
- If 15 is not the CRC (assume CRC is 1+2+3+4+5) RF module ignores it
- Guess what, I have no idea the format!
- AMIS Data Sheet indicated that it doesn’t use Preamble, only sync word, which is set in the by the manufacturer

CGM – Signal Dissection Example

- Direct mode is a configuration for the RF module that allows you to “see” all the signals on a given frequency
- Only way to view is with an oscilloscope or logic analyzer



CGM - Signal Dissection

- Here's what is known:
  - 76 bit transmission
  - CRC exists(Patent docs mention it)
  - There is a transmitter ID
    - 5 Characters
    - First char is 0 or 1, last 4 are [0-9,A-Z] (From Manual)
  - There is a Sync word of unknown length and value
  - There is some numerical data for the electrical resistance

CGM – Signal Catching

- Took a couple days to get some things figured out
  - Mid-80’s borrowed oscilloscopes – Manual not so friendly
  - RF module settings way too sensitive (AGC)
  - More register battles
- Eventually captured two 9.3ms transmission exactly 5 min apart!

CGM – Signal Transcription

- Collected 10 transmissions and decoded with paper and pen
- Looks like total jibberish – not what I was expecting
- I expected TCPdump like precision

CGM – Signal Transcription

- Was expecting a preamble per my research:
  - 10101010 = Research Preamble (8 bit)
- What I saw:
  - 1111 up to 11111111
- Re-read AMIS documents
- “RF Sense”
  - Chip expects a “wake-up” transmission
  - Series of 1s make sense!
  - Variance makes sense, RF module wakeup/setteling

CGM – Signal Transcription

- Think like a cryptographer
  - Known values in "plain-text" = last 4 of TX ID (CTA3)
  - Most of the transmission is identical every time (Sync, transmitter ID)
  - Data will change little in 5 min intervals
  - Patterns in "crypt-text"?

CGM – Signal Transcription

- Without changing any bits, only alignment I see something!
- Of 40 captured Transmissions 80% had this same series of 24 bits, all starting after Preamble/RF Wakeup
- Tried Inversion (AMIS chip option) - No luck
- Reached out to TI for help
  - Clueless. Obscure way to use this chip.
  - Got questioned on the ethics of my work

CGM – Signal Transcription

- Just too many combinations of settings, all impact how the direct mode behaves
- Zero real documentation, Zero users experienced
- Way beyond intended purpose (Definition of Hacking)

CGM – Security Risks

- Replay Attacks
  - Method: Capture and repeat packet
  - Impact: Incorrect Values or DoS
  - Limitations: Physical Range, Can’t manipulate values (yet)
- Denial of Service
  - Method: “Jamming” legitimate signal
  - Impact: User would get no values from CGM
  - Limitations: Physical Range, Non-Critical functions

CGM – Security Risks

- Injection
  - Method: If you can reverse the format, you can construct a sensor transmission. Listen and catch TX ID, then retransmit with fake data portion
  - Impact: User inputs incorrect values into insulin equation. Too much/too little insulin.
  - Limitations: Human Intelligence, Gut Feeling, Experience. Currently unknown data format.

Two Technologies – One Purpose

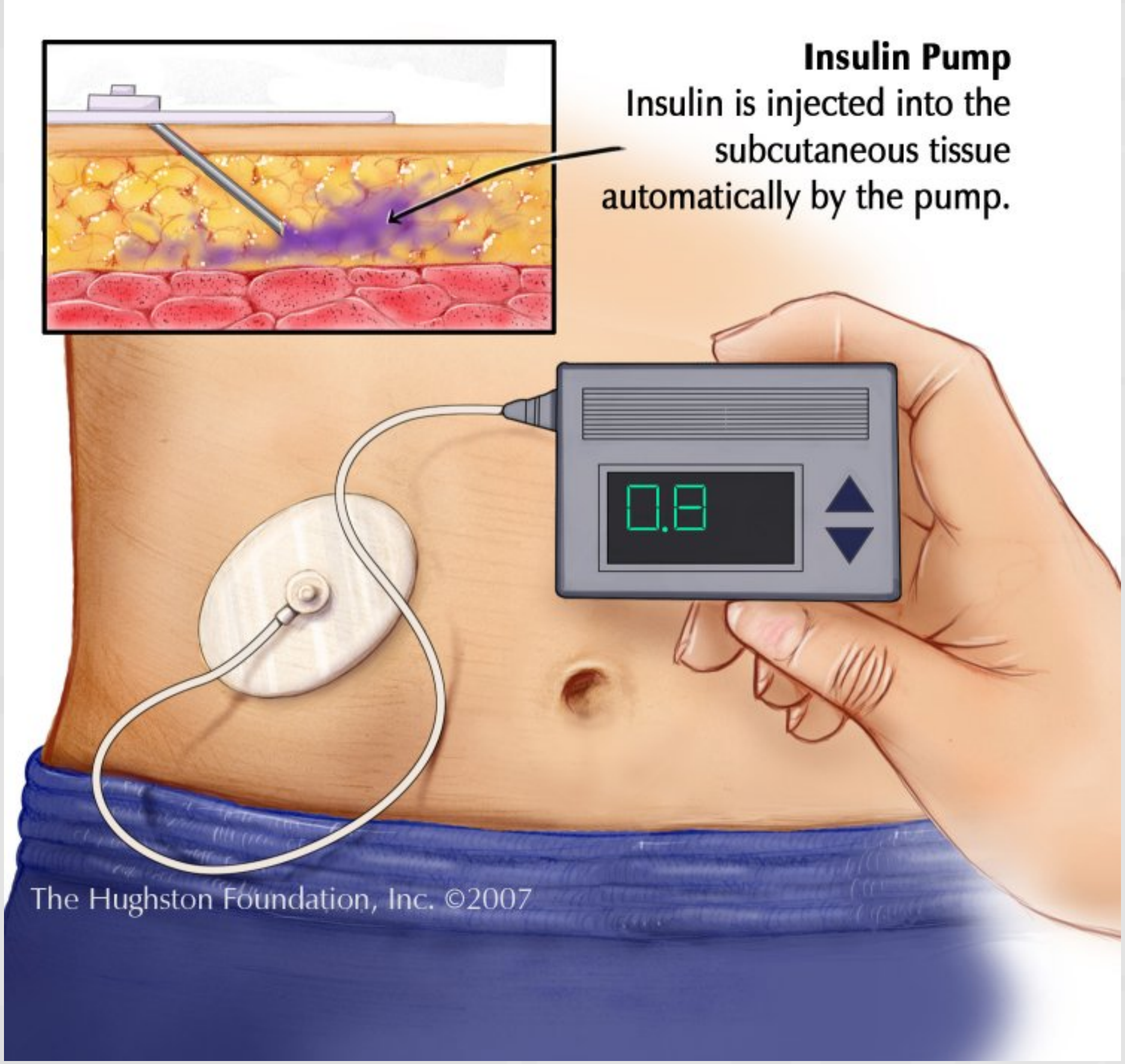
- Continuous Glucose Monitors (CGM)
  - New Technology
  - Small wire in tissue to measure electrical elements of fluid
  - Graphs sugar values over time
  - Huge leap forward

- Insulin Pump
  - Delivers insulin in 2 ways
    - Basel: Every 3 minutes
    - Meal: At Mealtime
  - Delivered through tubing attached to body
  - Tubing replaced every 3 days

Pump

- Insulin Pumps are used to delivery insulin to patients, hooked to a person via tubing 24/7
- Blood Meters can send measurements wirelessly to Insulin Pump for easier user experience
- Special USB dongles used to program Insulin Pumps and download history data
- Special wireless remotes used to deliver insulin

Pump



Insulin Pump
Insulin is injected into the subcutaneous tissue automatically by the pump.
0.8
The Hughston Foundation, Inc. ©2007

Pump – Recon

- Hypothesis: Wireless communication with insulin pumps are not secured and can be subject to attacks
  - Communication is more complex, probably bi-directional
  - Ancient windows programs used for config (will not install on anything above XP) indicate lack of knowledge
  - Devices not designed to be updated. No way of patching. 5+ year life span.

Pump – Recon

- Java Based Config program
  - Set logging from NONE to HIGH
  - BAM! Shows full packets, command structure, ACK responses, everything.

    -
      ```
      INFO: XXXXXX Command-sendCommand: SENDING CMD 0x5A (Set RF Power On-command packet)
      ```

    -
      ```
      INFO: XXXXXX Command-encode: about to encode bytes = <0xA7 0x31 0x33 0x70 0x5A 0x00 0xA8>
      ```

    -
      ```
      INFO: XXXXXX SerialPort-write(int buffer[])(20MS): writing <0x0A 0x0B 0xA8 0x6D 0x16 0x8E 0x39 0xB2 0x94 0xB5 0x55 0xA9 0xA5>
      ```

Pump – Signal Decoding

- Encoding?!
  - Encoding makes the message longer, but not double. Wonder how?
- Jar file
  - Not obfuscated, shows full encoding method
  - Not crypto complicated
  - Shows *all* commands and packet structure.

Pump – Signal Transmission

- Jar/Log Analysis
  - Talks directly to serial port (USB-to-Serial)
  - Only unique piece of information needed is Serial Number of pump
- Pump analysis
  - No ability to stop/block receiver
  - No verification step on pump
  - Does exactly what it’s told, no questions asked

Pump – Other Wireless Commands

- Remote Controls
  - All have optional remotes with unique IDs
  - Pump has to be configured to allow that ID
- Blood Meter
  - Blood Meter has Unique ID, sends beacon out with value + ID
  - Pump has to be configured to allow that ID
- Data Download
  - Has all historical data

Pump – Security Risks

- Hardware Needed
  - RF Serial/USB device are easy to get. $100 New, as low as $20 on eBay. No restrictions.
  - Remotes for pumps: Also trivial to acquire.
- Information Needed
  - Serial Number: Can be socially engineered.
  - Serial Number can be scanned for. Six digits number. Very feasible.

Pump – Security Risks

- Full Remote Control
  - Method: Send command to pump to allow Remote Control ID 12345.
  - Impact: Full meal insulin delivery control.
  - Limitations: Physical Range (100ft, more with mods), Pump Notification of Delivery
  - Very scary. Applies to any configurable setting. Including the variables on how much insulin to deliver.
  - “root” access to the device (and technically your body)

Future Potential

- JDRF Artificial Pancreas project
  - Links CGM and Insulin pump together
  - Eliminate User Intervention!
  - CGM data will be used to act without the user, *very* dangerous.
  - Makes security of CGM transmission much more important

Next Gen CGM

- New RF range (2.4ghz) bluetooth?
- Some already using bluetooth in pumps, partnering with CGM on new pump features
- Bluetooth better or worse? Maybe both
  - Tools for research
  - Tools for exploits (Metasploit module for Insulin pumps? AHHHHH!)

Suggestions

- New RF chips have crypto on board, use it
- Use IR rather then RF – Painful, but more secure
- Verify New Config
- Setting a Passcode
- Keep range limited
  - One pump uses 13mhz OOK. Near 20 meter ham band where 1 watt transmissions can be global.
- Blocking
  - Researchers are working on RF blocking necklaces for stopping RF OOK Pacemakers from malicious interference

Applying to Other Worlds

- Same Hardware RF Chips used in ICS/SCADA environments
- Older SCADA wireless uses OOK wireless in sub-1ghz bands
- Same techniques, different targets
- Harder to replace, more costly

Hardware Hacking Research

- Huge value, more should be done
  - Everything becoming wireless or connected
  - There is always a threat lurking, shouldn’t be ignored
  - Don’t hide behind obscurity, way too many smart people, it always fails
- It’s really hard
  - Think of trying to transcribe TCP packets on oscilloscope
  - More tools needed, more interest needed.

Feedback

- Please Remember to Complete Your Feedback Form!
- Questions? Comments?
- jay.radcliffe@gmail.com
- Twitter: @jradcliffe02