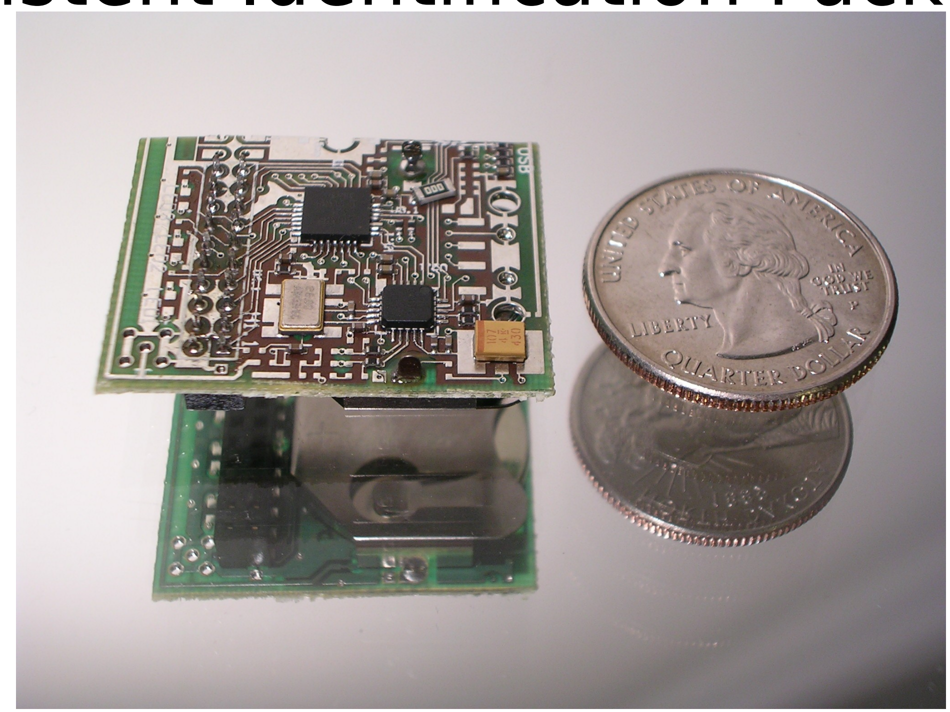
Experimenting with the PIP Radio Platform

Presented by Ben Firner

A PIP is a small radio device

- Affectionately called a “Pipsqueak”
- Runs on a coin cell battery
- PIP = Persistent Identification Packets



PIPs are a flexible test platform

- There is a very low barrier to entry
  - Code is in C (assembly can be used as well)
- We have already written the hard parts
  - USB code on the PIP and on Linux readers
- Well set up to test power consumption
- The best way to learn about radio is to use it

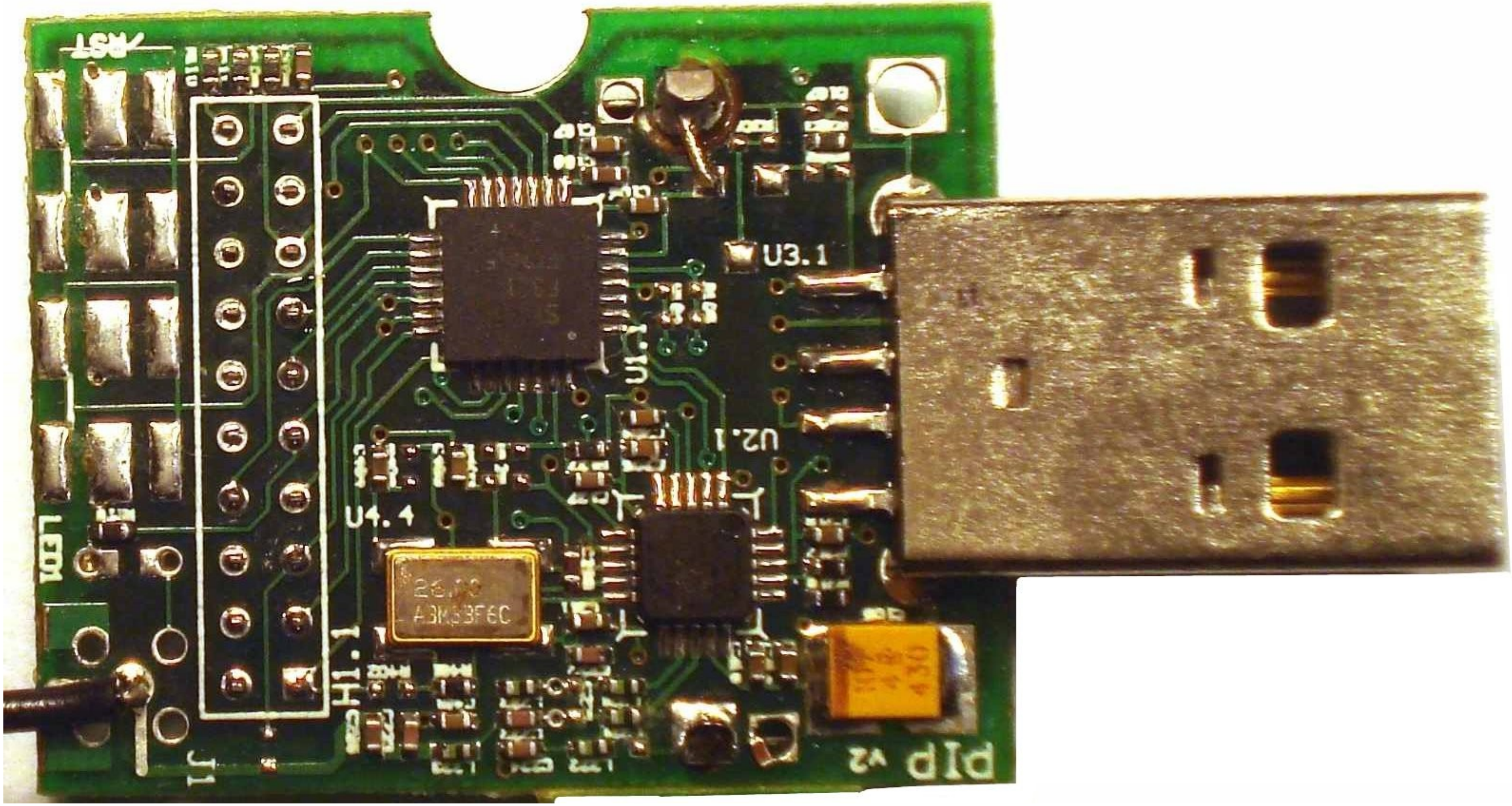
Things you can observe directly

- RSSI
- Power consumption using the oscilloscope
- View packet formats and spectrum on the network analyzer
  - CC1100 does FSK, GFSK, MSK, and ASK/OOK
- Use the vector signal analyzer to get raw packet data

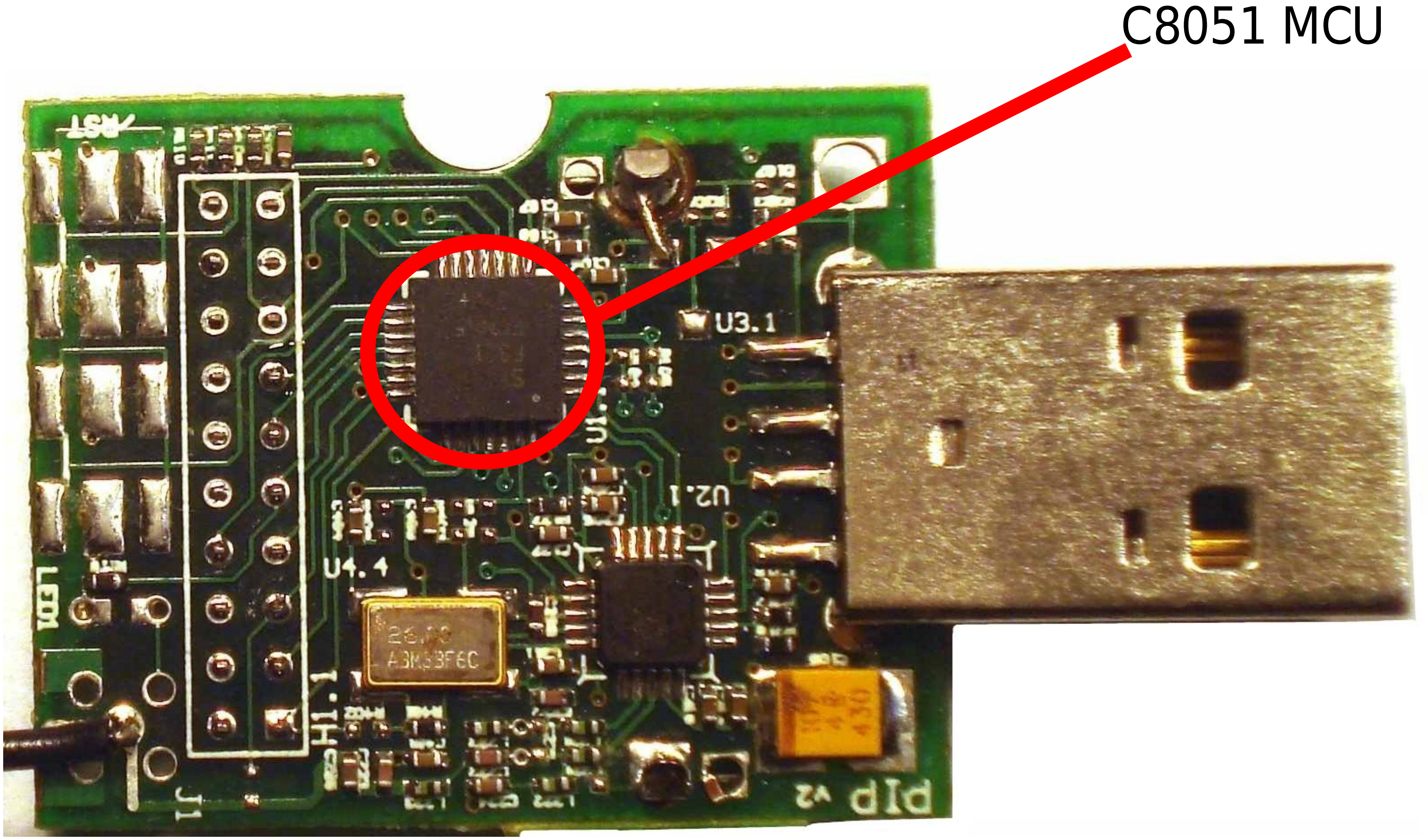
Statistics you can gather

- Packet loss
  - How well does radio work from car to car?
- Interference
  - Should you have your router next to your microwave?
- Signal attenuation
  - Are tinfoil hats effective?

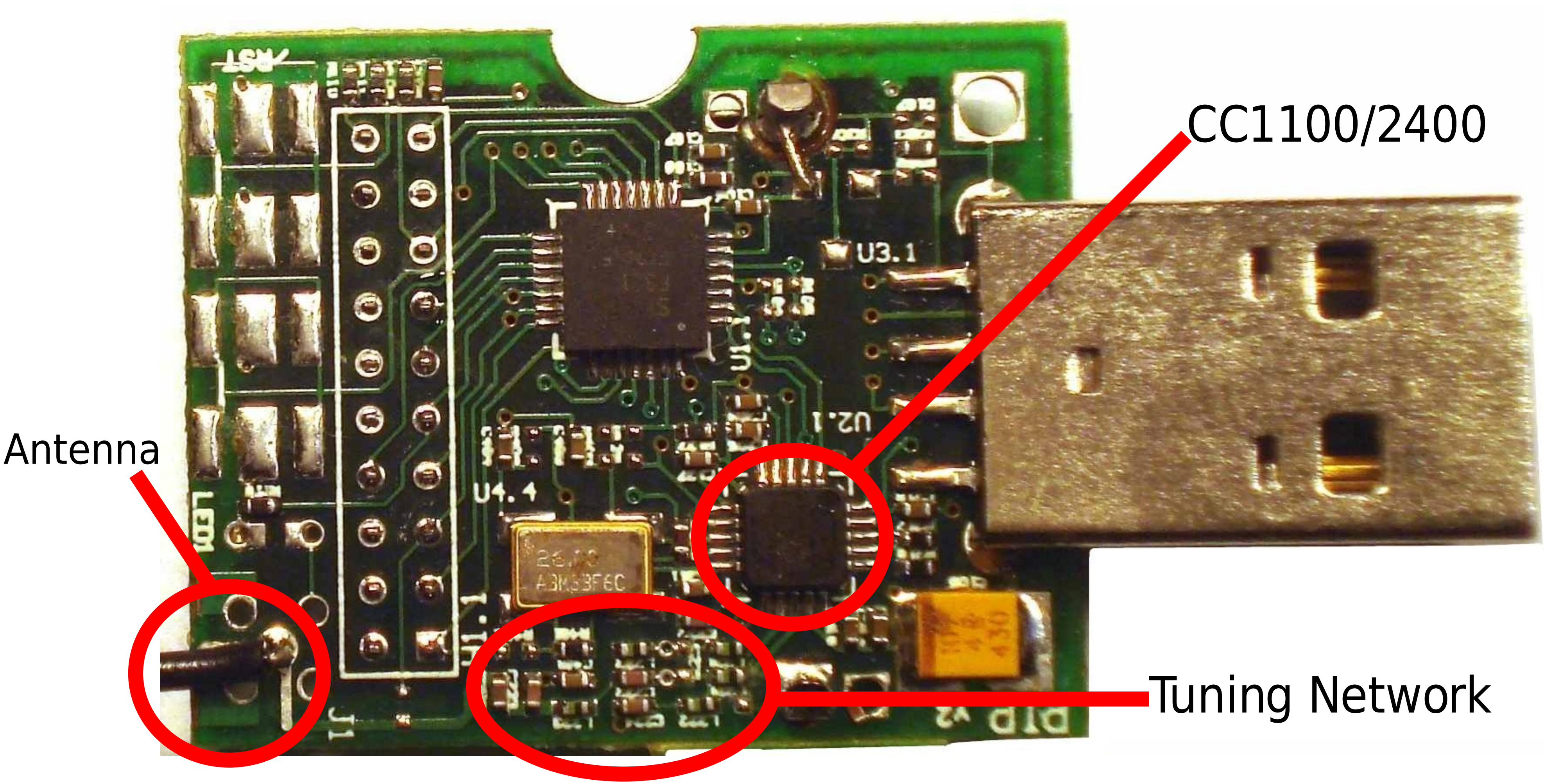
PIP Hardware



PIP Hardware

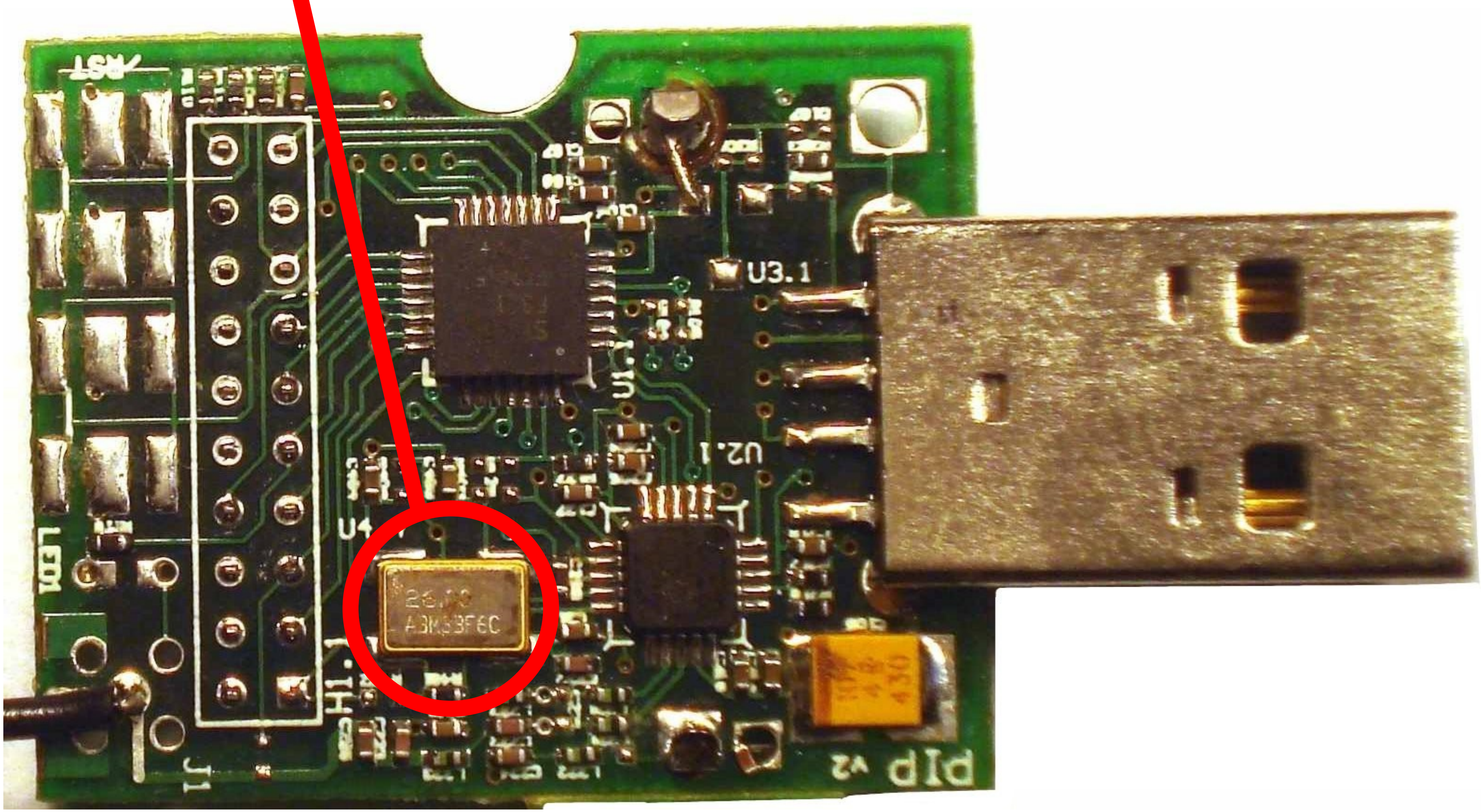


PIP Hardware

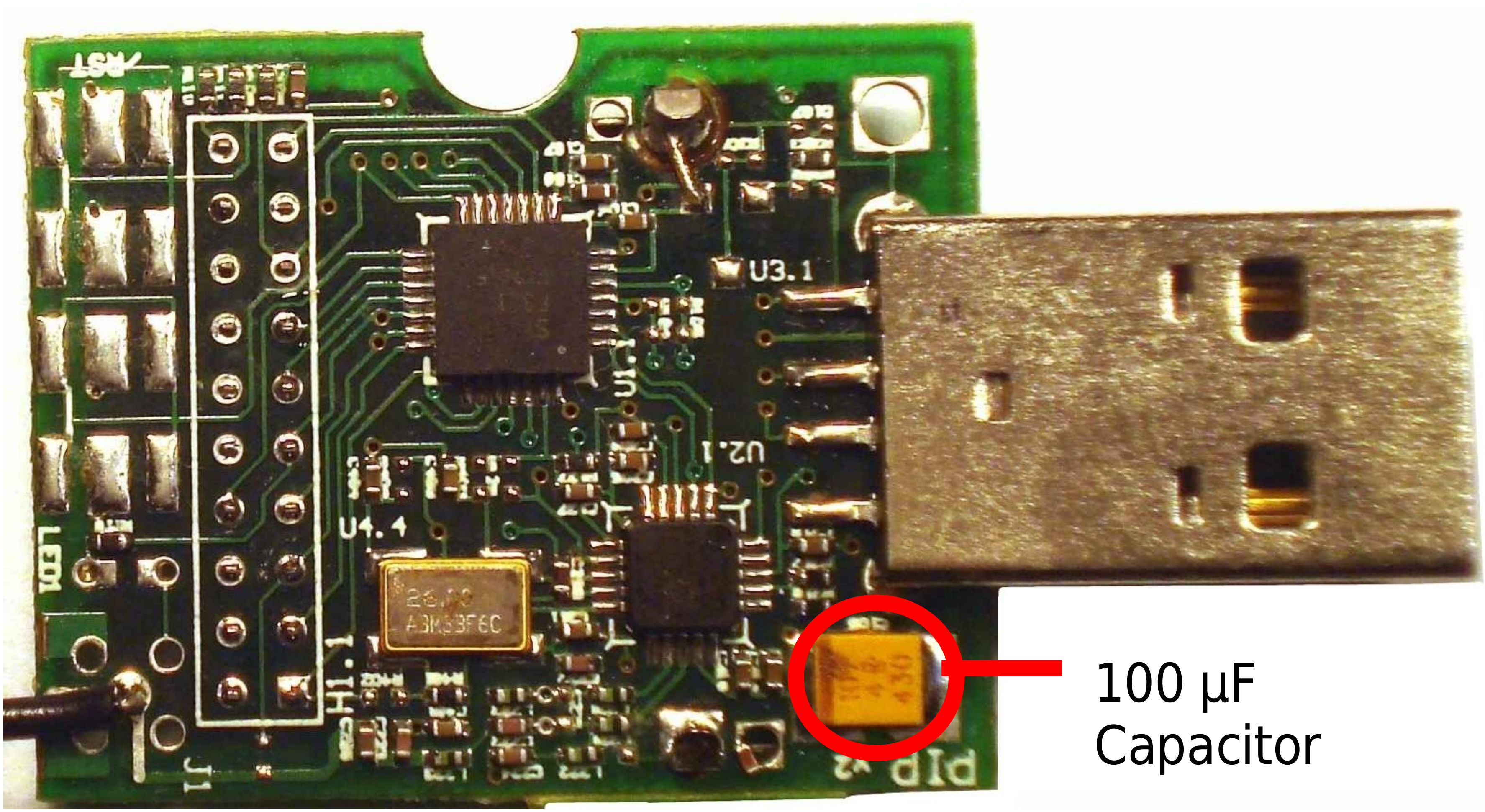


PIP Hardware

26MHz Oscillator (MCU clock)

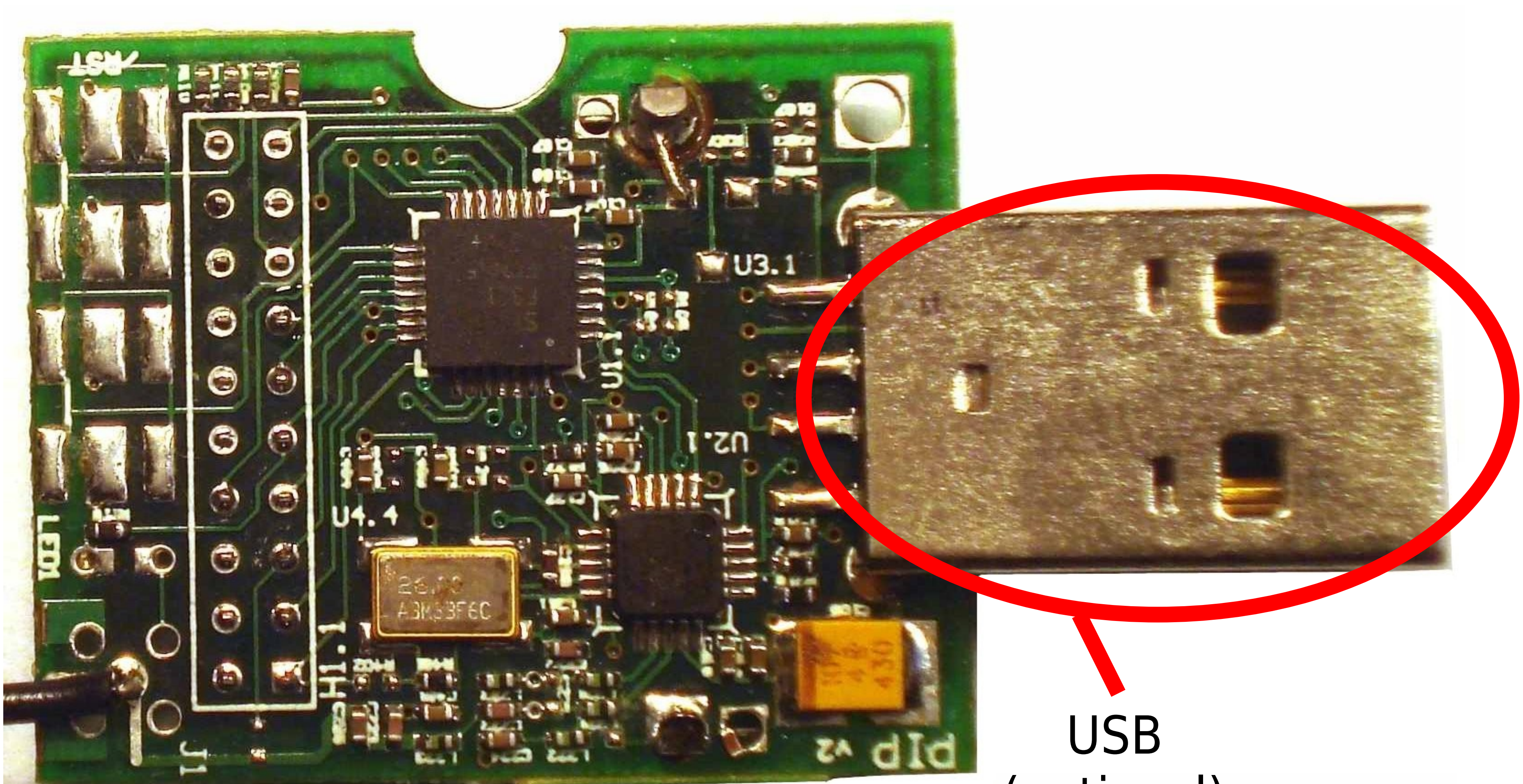


PIP Hardware



100 µF
Capacitor

PIP Hardware



USB
(optional)

Comparison to Berkeley Motes

PIPs

- Coin cell battery
  - Can be run with AA batteries using a special device.
- CC1100/2400 radio
- Programmed in C, use Keil µVision to flash.

Berkeley Motes

- Many models
- AA batteries
- CC1100/2400 radio
- Programmed with TinyOS and NesC

PIP Software

- Streamcollect
  - Our data collecting code. This receives data from multiple PIPs attached to a computer via USB. Data is stored in a sqlite3 database and can be printed to stdout.
- Analyze2
  - Analysis code that gives many statistics about a data set collected with Streamcollect.
- PIP codebase

The existing codebase is useful

- SendBeacon
  - Sleep/Transmit/Sleep cycle with optional frequency hopping
- Listen4Beacons
  - Receives packets and prints them via USB.
- Jamming code
  - Continuous transmission of random bits
- Code for ACK and CS protocols also exists

Rolling your own is easy

- Many parameters can be selected by changing a register or calling a function
  - Modulation format, Rx/Tx frequency, transmission power, FEC, Data Whitening, transmission duty cycle, etc
- It is easiest to start with existing code and change it to suite your needs

How PIP programming works

- All of the default register values for the CC1100 are set in RegSettings_X.c
- Change default register values or call halSpiWriteReg to set register values.
- Look at the CC1100/2400 data sheets to see what registers do:
  - focus.ti.com/lit/ds/symlink/cc1100.pdf
  - focus.ti.com/lit/ds/symlink/cc2400.pdf

Changing the frequency

- #include <Rollcall\tuning.h>
- ....
- setFreq(902100000.0)

WINLAB
WIRELESS INFORMATION NETWORK LABORATORY

Writing data to the USB FIFO

- #include “queue.h”
- #include “F32x_USB_Structs.h”
- #include “usb_init.h”
- ....
- //Initialize global USB queue
- host_queue_p = queue_init();
- ....
- queue_insert(host_queue_p, str, sizeof(str));

Sending variable length packets

- #include <Rollcall\rfsuite.h>
- ....
- //Data whitened, variable packet length
- halSpiWriteReg(CCxxx0_PKTCTRL0, 0x41);
- BYTE packet[] = {length, ...};
- rfSendPacketNonblock(packet, sizeof(packet));

Receiving variable length packets

- #include <Rollcall\rfsuite.h>
- ....
- //Data whitened, variable packet length
- halSpiWriteReg(CCxxx0_PKTCTRL0, 0x41);
- BYTE rx_buffer[psize];
- BYTE status[2];
- UINT8 len;
- rfReceive(rx_buffer, &len, status);

WINLAB WIRELESS INFORMATION NETWORK LABORATORY

Transmitting random data

- #include <Rollcall\rfsuite.h>
- ....
- //Infinite length transmission with random data
- halSpiWriteReg(CCxxx0_PKTCTRL0, 0x22);
- halSpiStrobe(CCxxx0_STX);
- //Infinite loop so the packet never ends
- while(1);

Keep the PIP in mind

- The PIPs are perfect to doing quick tests
- A few measurements can verify your assumptions before you do a large simulation or test
- PIPs are also a great tool to use when trying to gain experience using are measurement tools