USB on RAILS

Cheap & Driver-less USB for Smart Objects

Tod E. Kurt • Sketching in Hardware 1 • June 24 2006

USB is Cool

- Simple protocol (compared to Ethernet, firewire)
- Pervasive
- Powers your gadget (500mA)
- Standardized device classes
  - storage, audio, video, human i/o
- Wireless USB soon

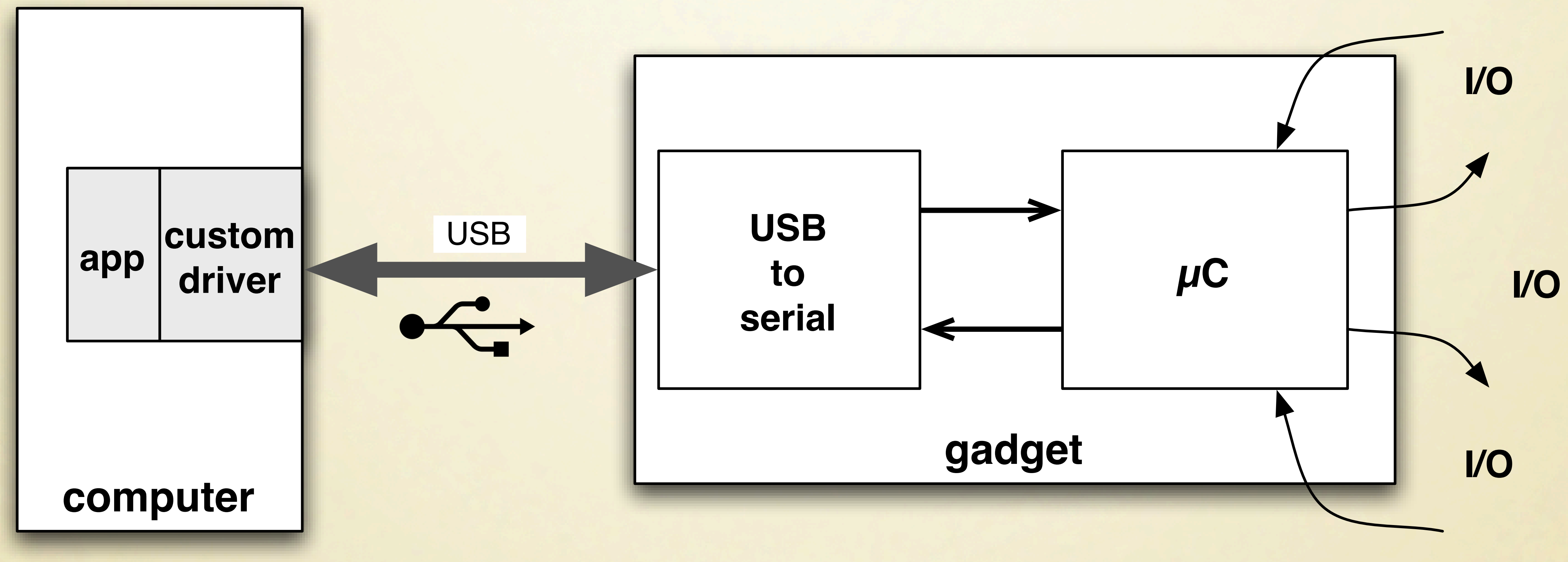
Hopefully all of our USB expertise extends to wireless USB

But USB is Crap

- Complex protocol (compared to serial or parallel)
- Device-specific drivers, one for each platform
- Drivers are often kernel-level, so reboot
- Driver updates lag OS updates
- Such a buzz-kill
- But there's a better way...

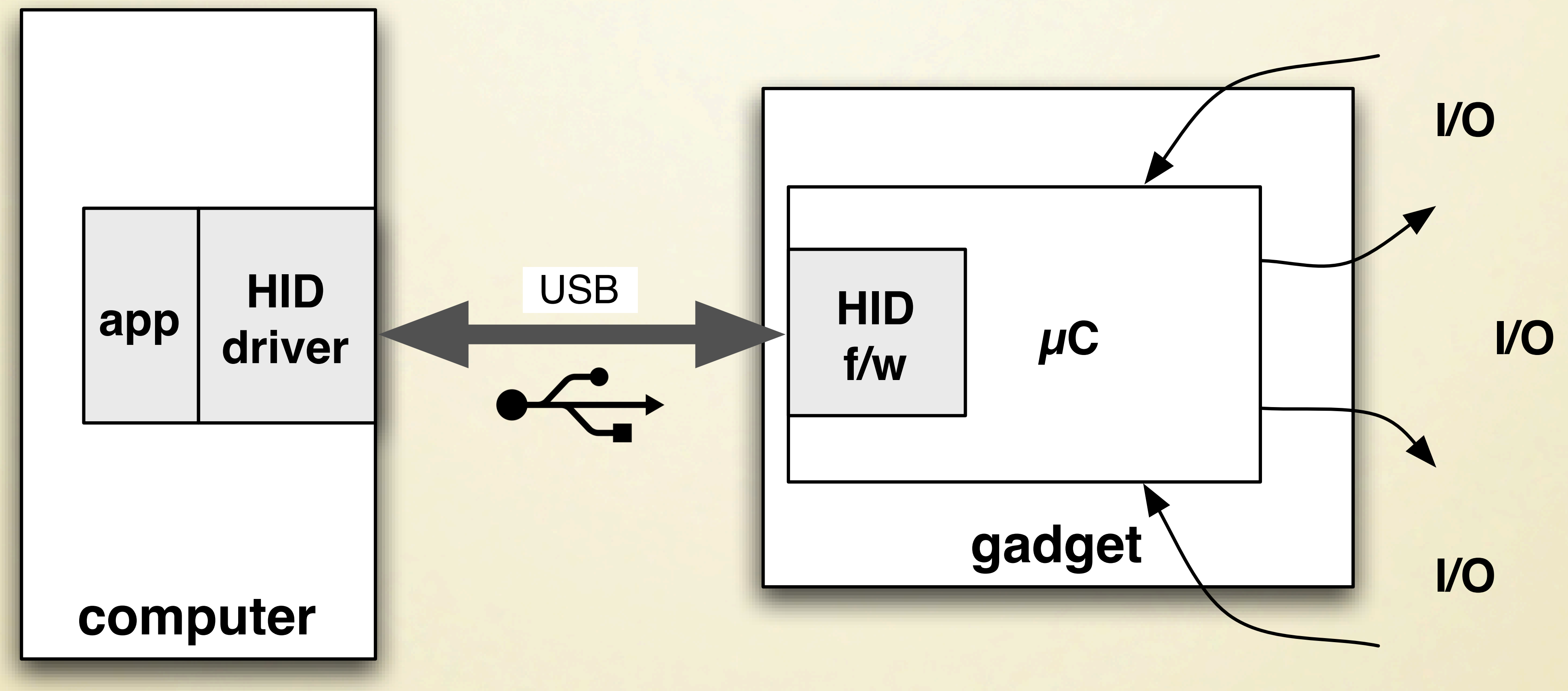
MIDI is serial, like RS-232
Basic Stamp doesn't have USB, so USB is invisible to hobbyists
buzz kill: for giving demos, for giving away 'built' demos

Typical Solution



pros: everyone knows how to do serial input/output
cons: custom driver, extra chip, protocol creation between computer & gadget

A Different Approach



pros: fewer parts, cheaper, no reboot!, higher-level data exchange
cons: gotta deal with HID now

HID: Only Humans Allowed

- Mouse, keyboards, gamepads, force-feedback, haptics
- Generalized periodic reporting of arbitrary data
  - Get_Report – get data structure from device
  - Set_Report – send data structure to device
- Driver already exists in all OSs
- Bluetooth has a HID class too
- Lots of jargon: "endpoint", "usage table", "descriptor"

"HID" is a misnomer: not just input, but output too
all HID data exchange built from Get_Report & Set_Report
periodic data exchange at the USB layer -> no more polling

Hack HID: Embrace & Subvert

- HID device class is perfect for smart objects
- HID solves gadget protocol, just pass data structs
- Use Rails approach: solve common case simply
- Build re-usable snippets:
    - Button, Knob, Slider, Display, LED, etc.
    - HID usage table snippets
    - Code snippets on PC & μC

usage table == the description of how data is exchanged with a HID device

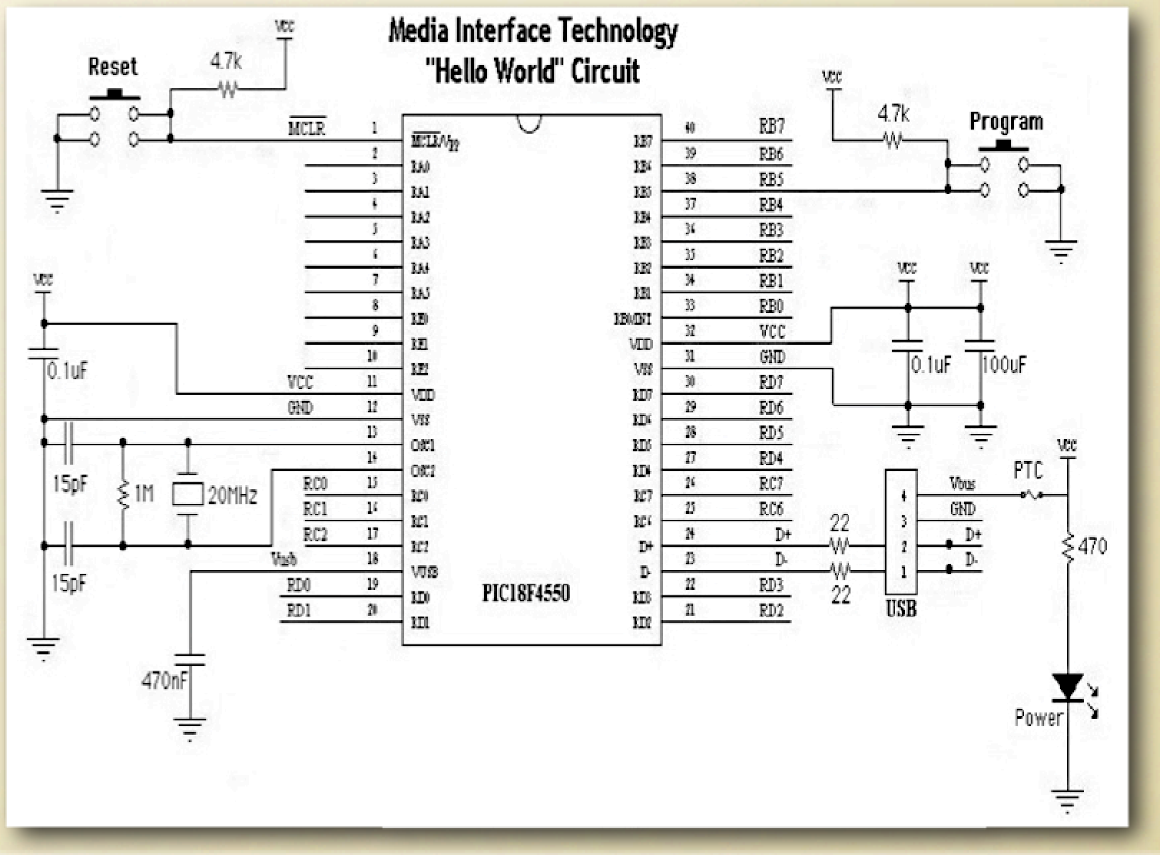
USB µControllers

- Once only special-purpose (mouse, flash drive)
- Now on general microcontrollers, cheaply
- Examples: (from common hacker chip families)
  - Microchip PIC 184550 family
  - Atmel AVR with AVR-USB
- A bunch of others too (Cypress,TI, Silicon Labs)

AVR-USB isn’t a “USB µC” per se, but it plays one on TV

PIC18F4550

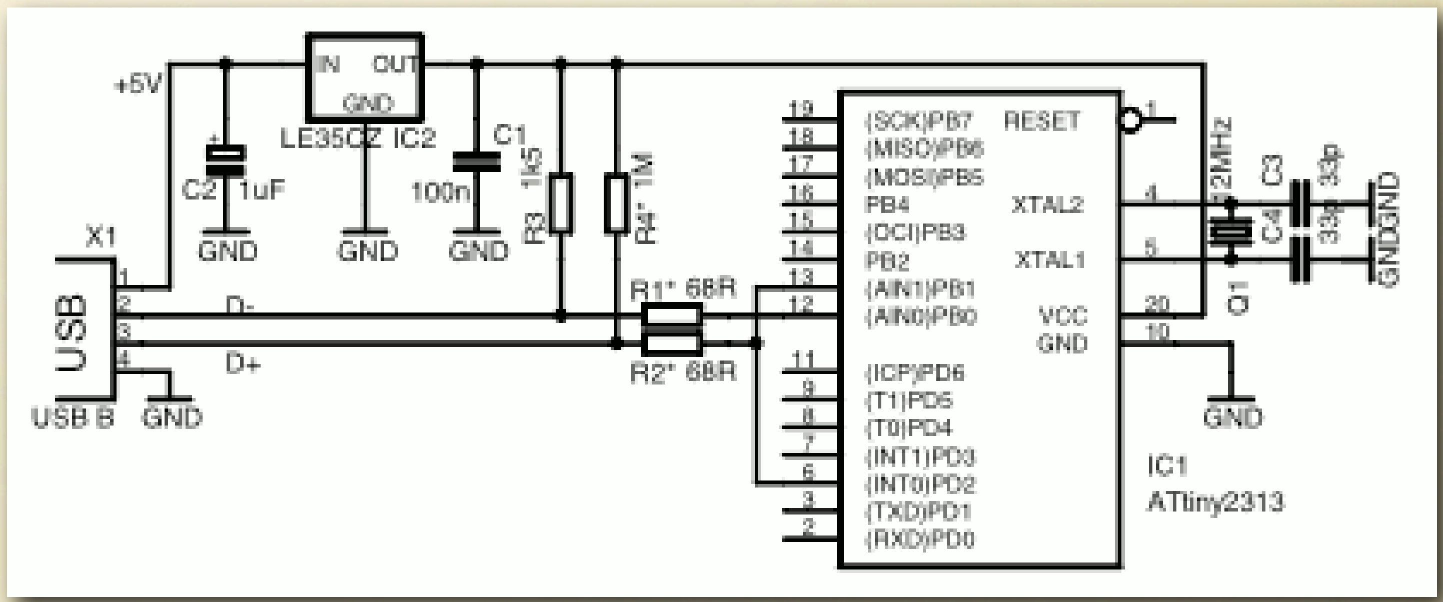
- Built-in USB interface
- Fast data transfers, USB 2.0
- Microchip C compiler (not free, Windows, etc)
- Lots of I/O
- Cheap: $10 in singles
- SparkFun sells them



I personally am put off of PICs now because of the very Windows-centric development community and the dearth of free high-level tools

AVR-USB

- USB in software
- Uses standard AVR µcontrollers
- Really Cheap: $2 for ATtiny2313
- Open-source (avr-gcc)
- Simple
- Yup, SparkFun

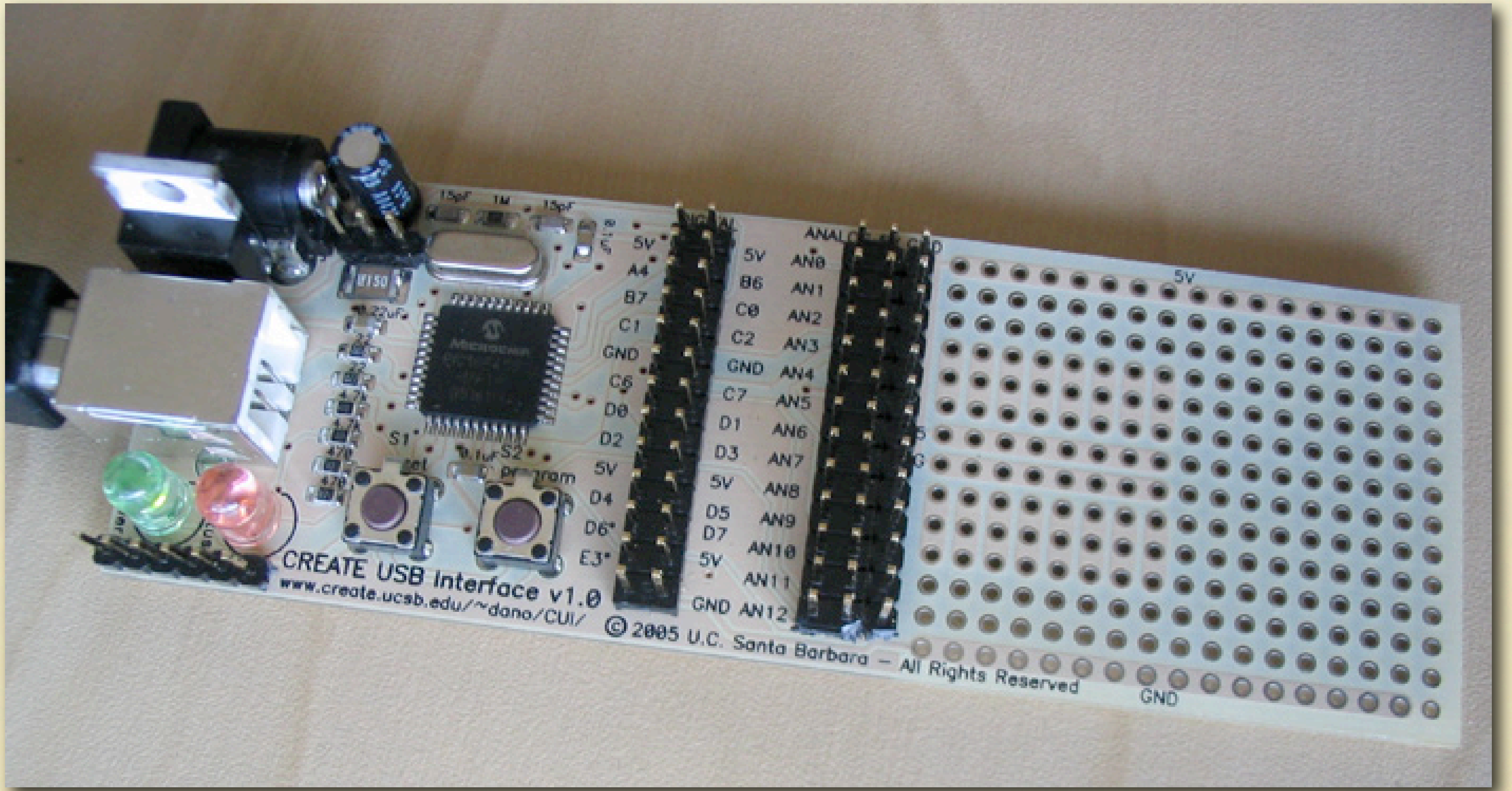


http://www.obdev.at/products/avrusb/

At $2 (even cheaper in quantity from Digikey), USB becomes almost free.
Circuit is smaller than USB fobs

PIC18F4550

CREATE USB Interface – Dan Overholt @ UCSB

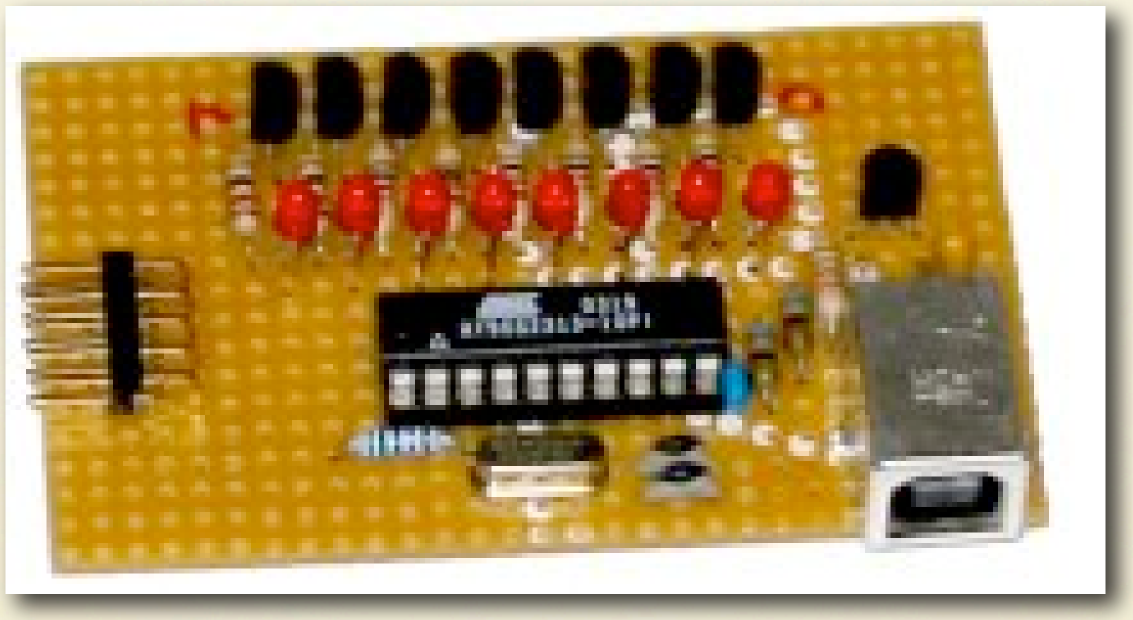


http://www.create.ucsb.edu/~dano/CUI/

Focuses on input part of HID, but can be used for other things
His “boing boing” project: accelerometers & touch sensors make music
Pretty cheap: $50 direct from him
Almost fob-sized

AVR-USB

PowerSwitch



LEDLoad

PowerSwitch: turn on/off up to 8 devices
LEDLoad: tiny board with multi-color LED output for PC stated

Help Out

- Prefer USB HID, eschew USB-to-Serial
- Help create tools for easy HID development
  - Windows, Mac OS X, Linux programmers
  - PIC & AVR aficionerdos
- Let's subvert HID for sketching purposes

Code snippets implemented as macros or code libraries
I'm focussing on AVR and Mac OS X, probably Arduino

End of Line

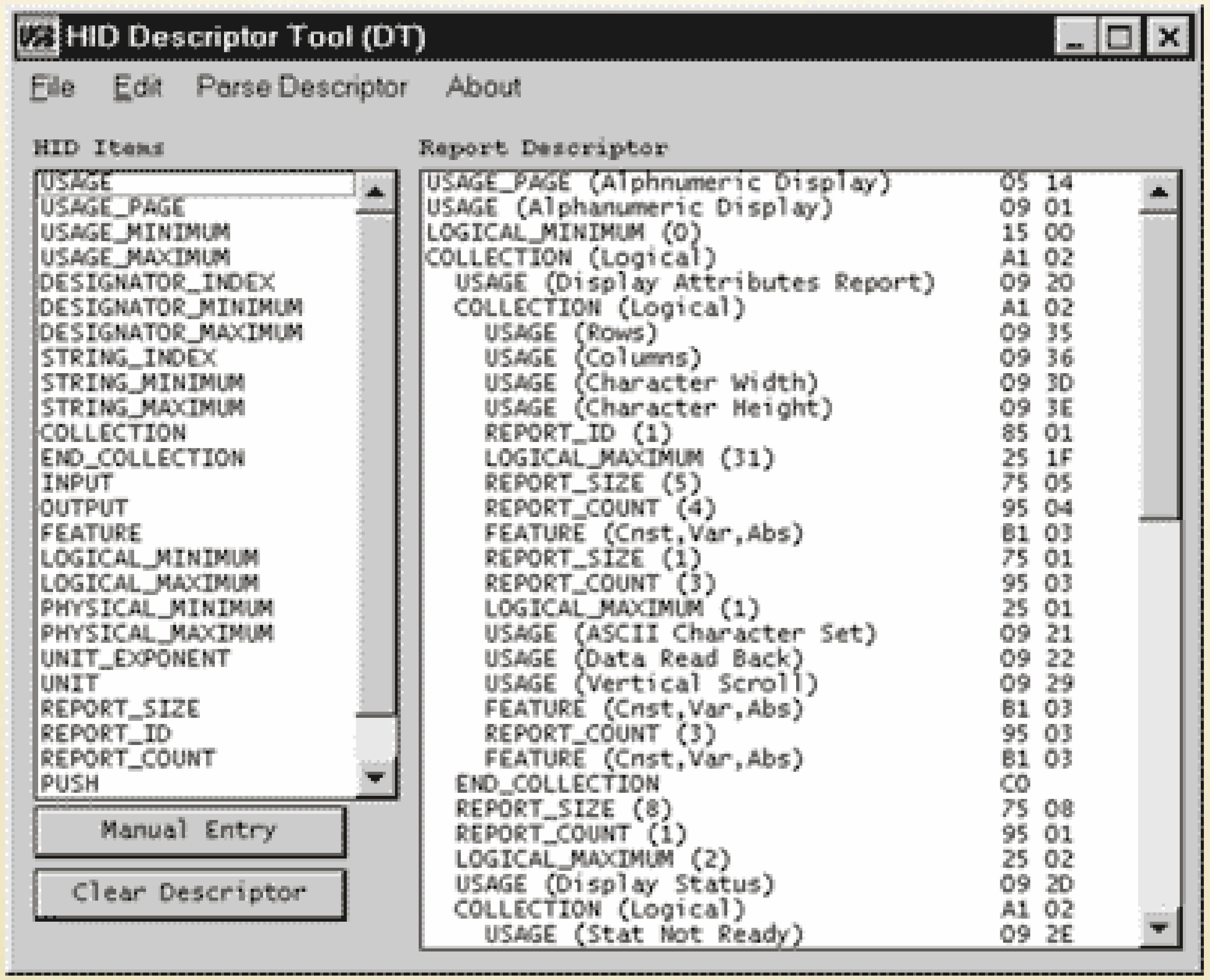
http://thingm.com/hidhacking/

Tod E. Kurt
tod@todbot.com
http://todbot.com/

http://thingm.com/

Let’s put USB into everything. Even rubber duckies.

HID HELL



```
USAGE
USAGE_PAGE
USAGE_MINIMUM
USAGE_MAXIMUM
DESIGNATOR_INDEX
DESIGNATOR_MINIMUM
DESIGNATOR_MAXIMUM
STRING_INDEX
STRING_MINIMUM
STRING_MAXIMUM
COLLECTION
END_COLLECTION
INPUT
OUTPUT
FEATURE
LOGICAL_MINIMUM
LOGICAL_MAXIMUM
PHYSICAL_MINIMUM
PHYSICAL_MAXIMUM
UNIT_EXPONENT
UNIT
REPORT_SIZE
REPORT_ID
REPORT_COUNT
PUSH
```

```
USAGE_PAGE (Alphnumeric Display)            05 14
USAGE (Alphanumeric Display)                09 01
LOGICAL_MINIMUM (0)                         15 00
COLLECTION (Logical)                        A1 02
  USAGE (Display Attributes Report)         09 20
  COLLECTION (Logical)                      A1 02
    USAGE (Rows)                            09 35
    USAGE (Columns)                         09 36
    USAGE (Character Width)                 09 3D
    USAGE (Character Height)                09 3E
    REPORT_ID (1)                           85 01
    LOGICAL_MAXIMUM (31)                    25 1F
    REPORT_SIZE (5)                         75 05
    REPORT_COUNT (4)                        95 04
    FEATURE (Cnst,Var,Abs)                  B1 03
    REPORT_SIZE (1)                         75 01
    REPORT_COUNT (3)                        95 03
    LOGICAL_MAXIMUM (1)                     25 01
    USAGE (ASCII Character Set)             09 21
    USAGE (Data Read Back)                  09 22
    USAGE (Vertical Scroll)                 09 29
    FEATURE (Cnst,Var,Abs)                  B1 03
    REPORT_COUNT (3)                        95 03
    FEATURE (Cnst,Var,Abs)                  B1 03
  END_COLLECTION                            C0
  REPORT_SIZE (8)                           75 08
  REPORT_COUNT (1)                          95 01
  LOGICAL_MAXIMUM (2)                       25 02
  USAGE (Display Status)                    09 2D
  COLLECTION (Logical)                      A1 02
    USAGE (Stat Not Ready)                  09 2E
```

This is how you configure HID descriptors currently. yuck.