LumiNet: Prototyping Organic Physical Networks

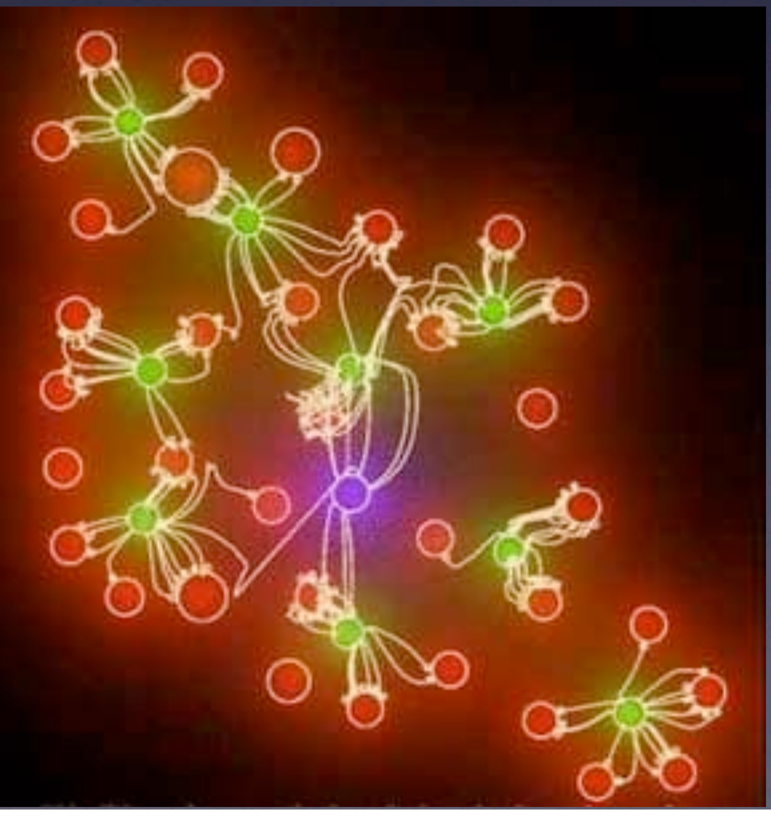
(and hacking Arduino in the process)

Jan Borchers and René Bohne

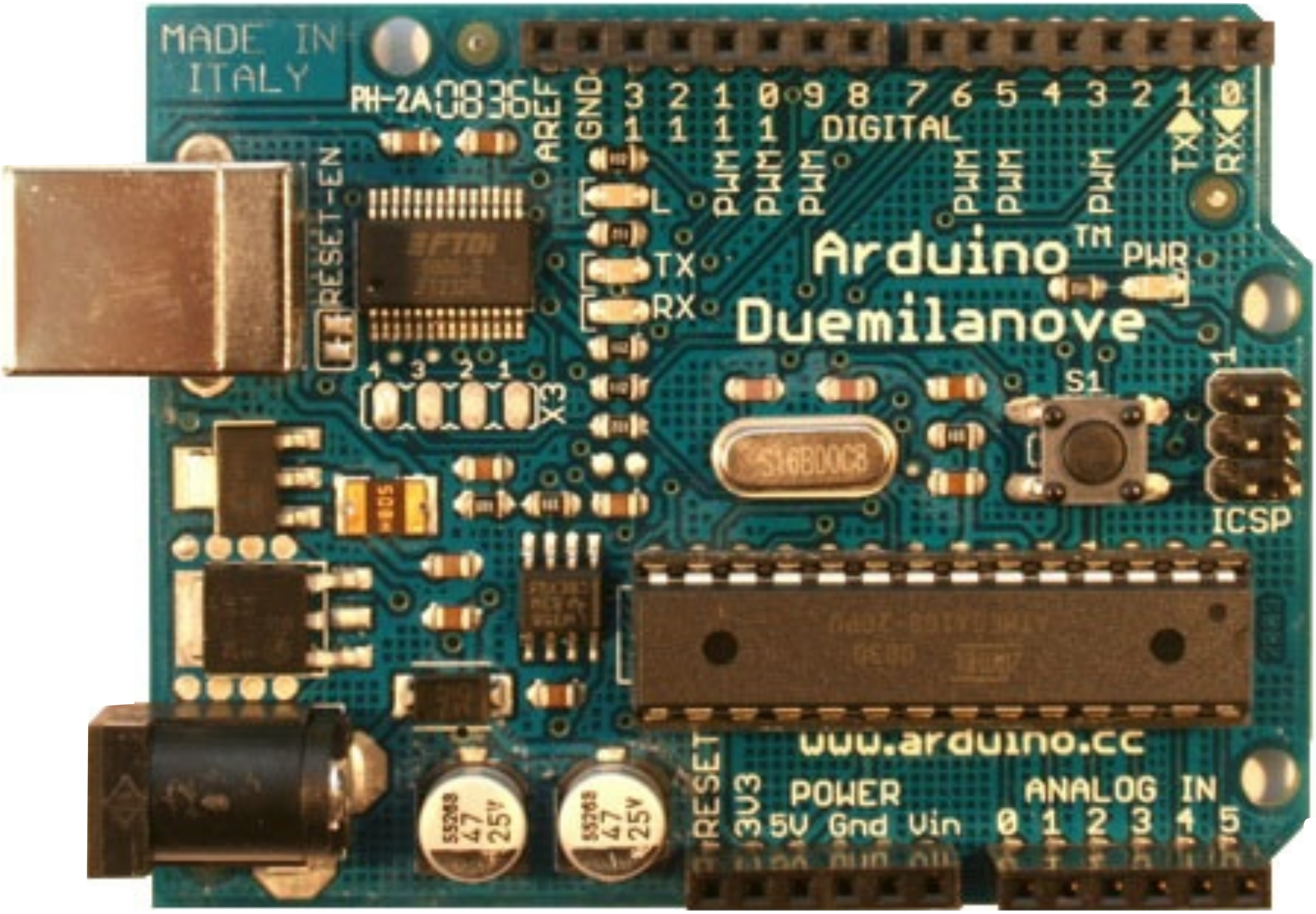
Media Computing Group
RWTH Aachen University, Germany

Sketching in Hardware
London, July 17, 2009

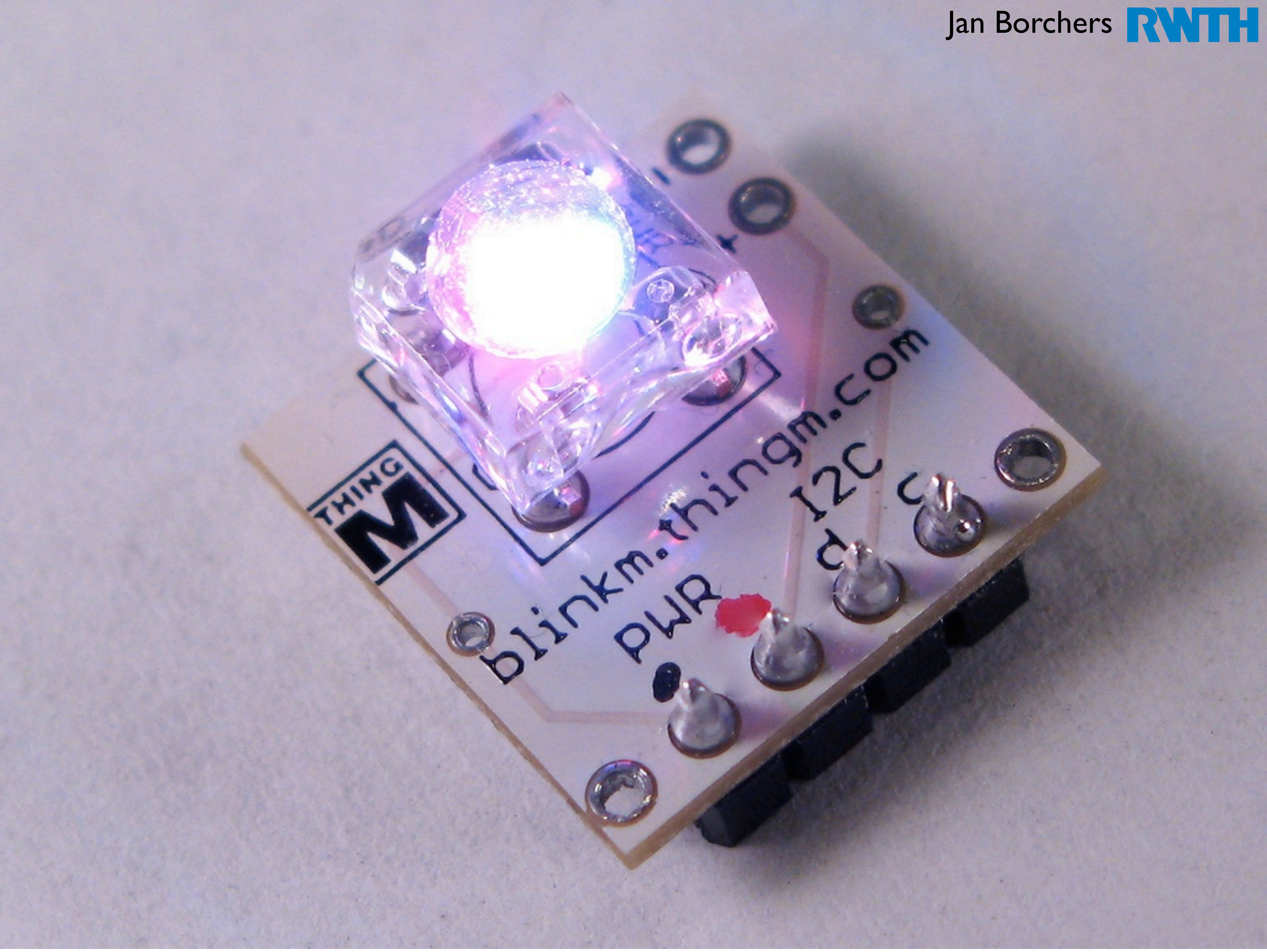
www.luminet.cc



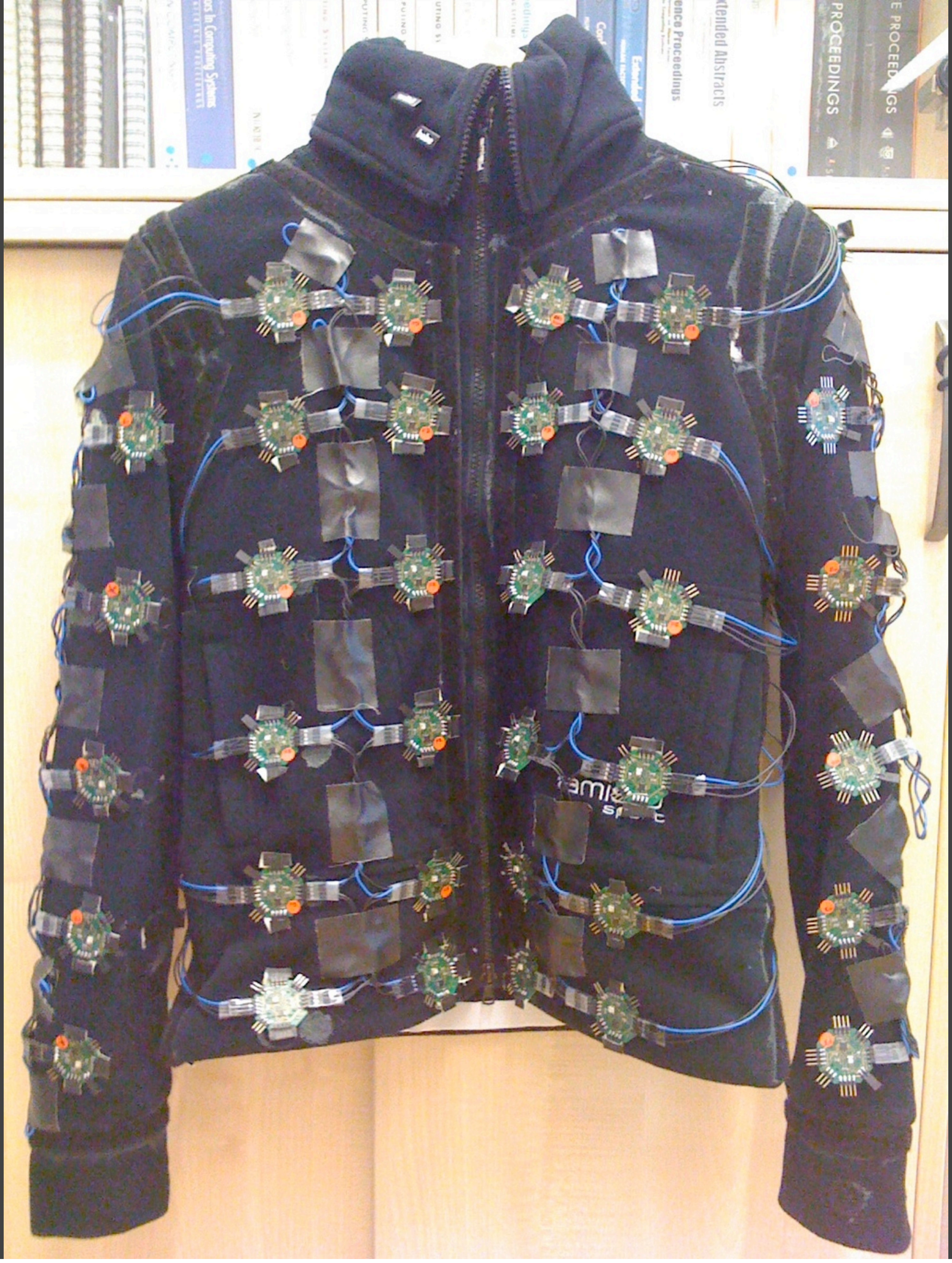
Computing and networking have been inspired by biology
“Natural” user interfaces have been a good choice in the past
Can we directly use more organic concepts in the UI? Physicality (BumpTop,...), Continuity (iPhoto), Humane presentation forms (associations)? — but still all virtual/on-screen
We want physical organic interface, and a networked one



Arduino limits:
Project with 100s of units?
Create emergent swarming behavior?
Lightup clothing with independent nodes?
Give away to people for studies, at events?
Arduino is too expensive and too large for these ideas
More importantly: Arduino supports centralized architecture — one central controller, many sensors and actuators (like LEDs)
This makes developing complex, high-performance networked projects hard

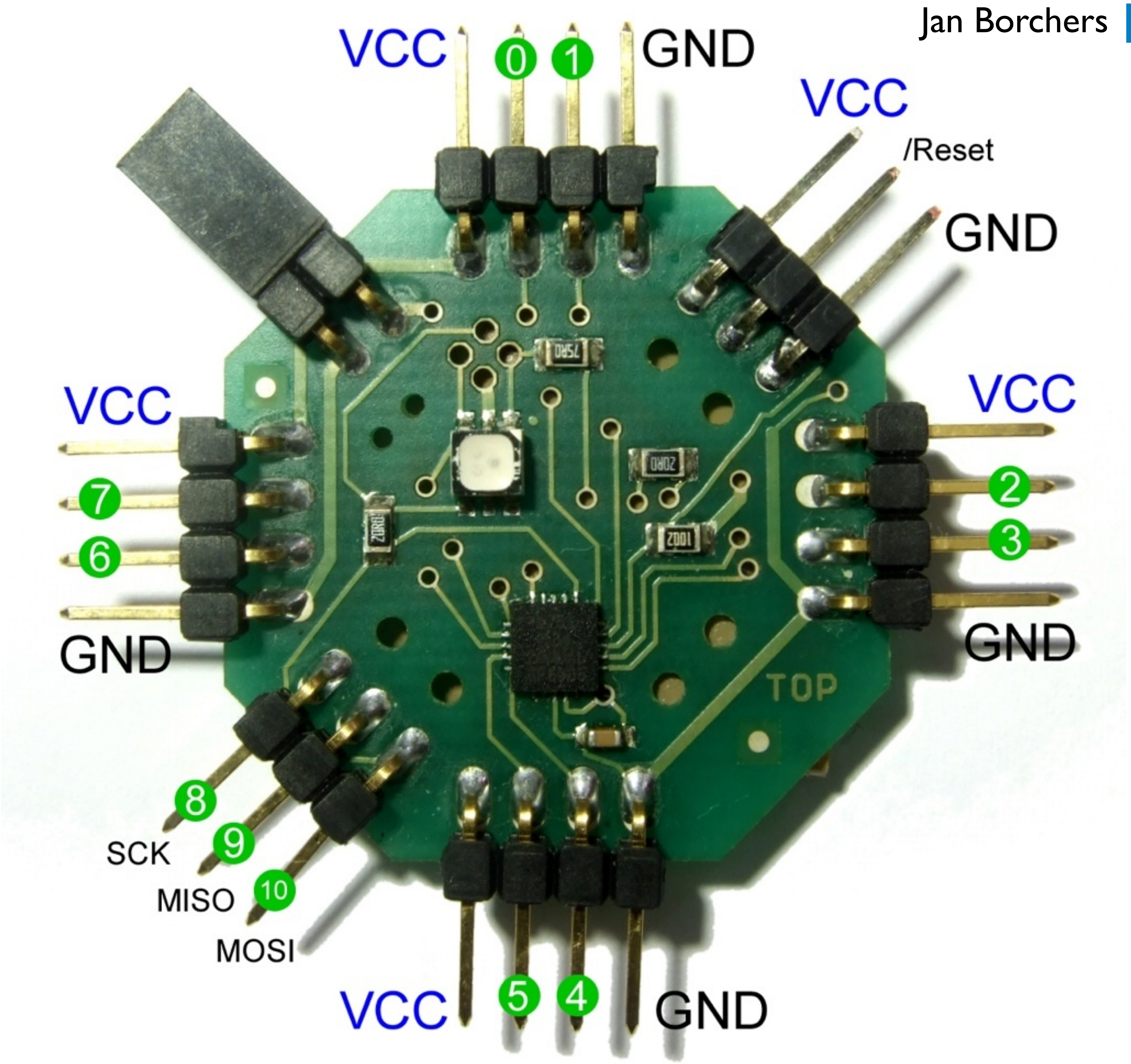


ThingM's BlinkM Smart LED:
- right hardware
- but missing communication links
- and: still designed for central control (e.g., through an Arduino)



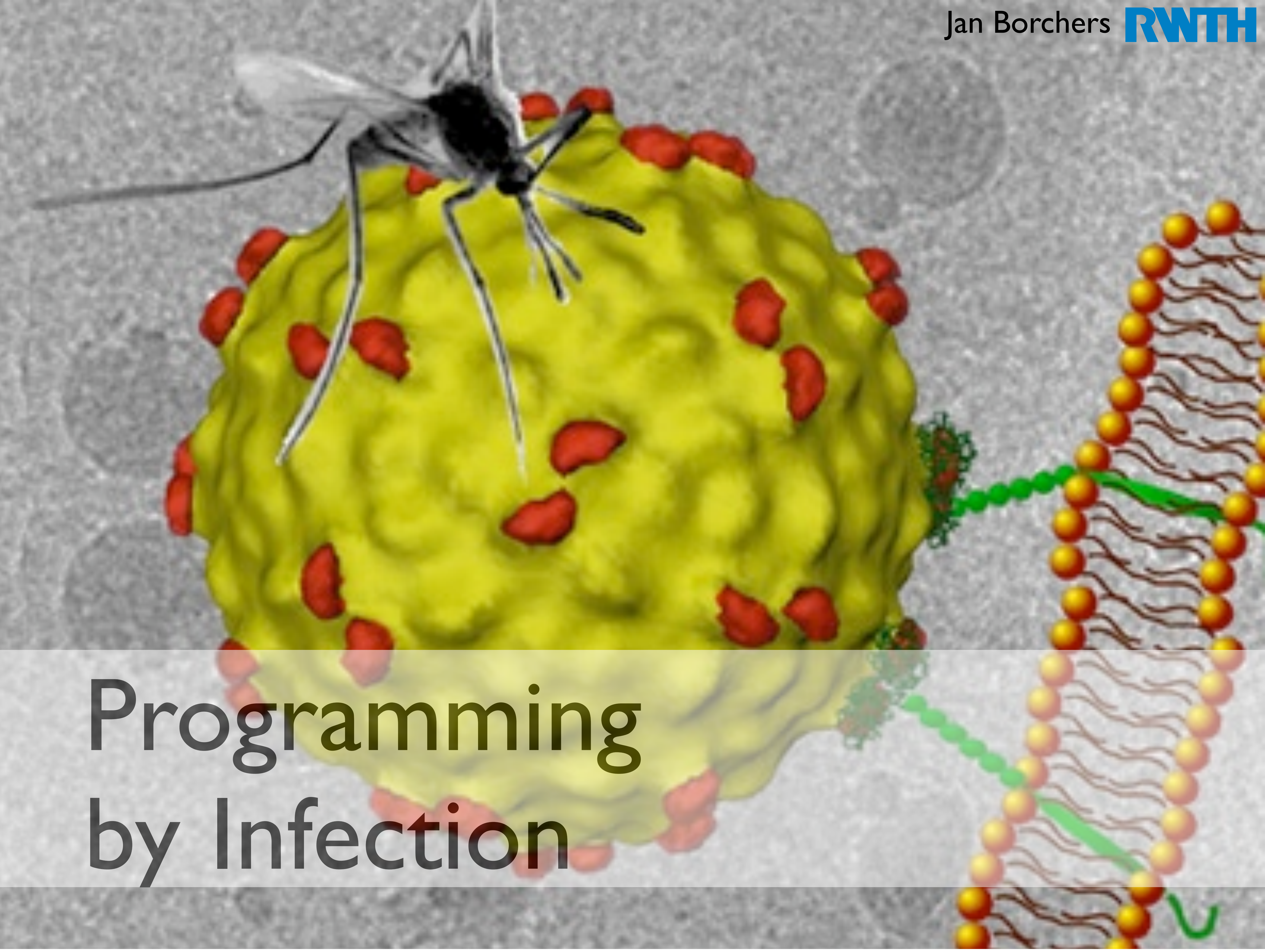
Our goal:
- Wired network of intelligent pixels
- under $10
- Organic network
- No central control
- Naturally highly parallel & robust
- Reconfigurable
- Supports decentralized stimuli from sensors at any node

E.g., this jacket contains 80 LumiNet nodes (two of these were made in 2008)



1.2” PCBs
ATtiny84
The LumiNet design (board revision 3.6):
- Bright OSRAM RGB SMD LED
- 4 bidirectional communication lines (numbers are Arduino-like pins)
- Separate one-time ISP connectors
- Separate one-time programming connectors
- Sensor dongles can inject data anywhere into network
- E.g., the “LightRing” program (see video) for the jacket has pushbuttons attached to nodes near the wrists for input grounding one of that node’s data inputs
- Jumper disconnects LED to make sure ISP programming works for initial bootloader programming (a hack)

Ideas for using LumiNet:
- Tangible Education & Simulation of tree algorithms and swarm behavior
- Wearable Computing projects (lightup clothing)



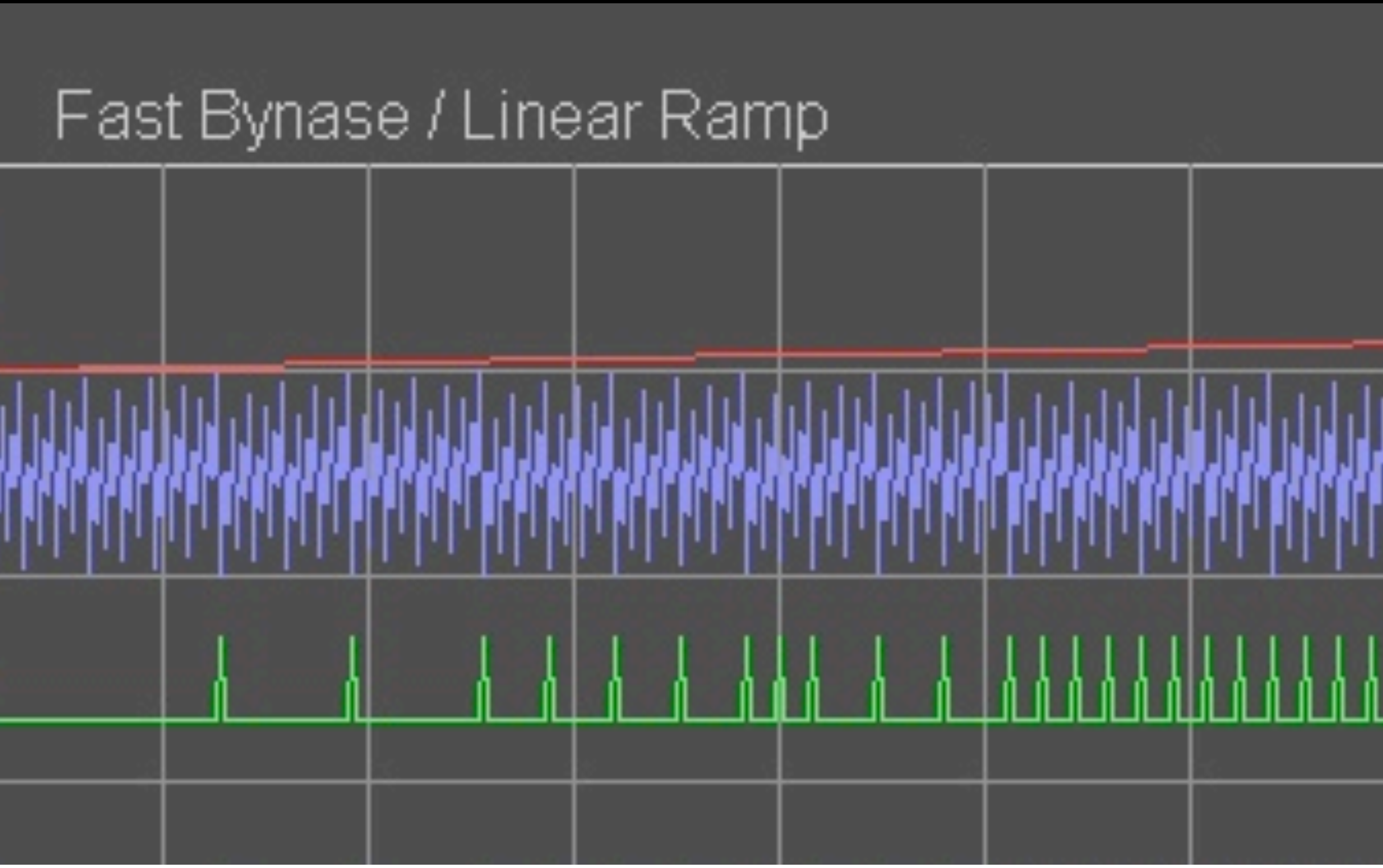
Programming by Infection

Problem: Individually reprogramming every single node for new behavior is tedious

Solution:
- Physical dongle (also LumiNet board) inserts program anywhere as viral vector
- Bootloaders on nodes self-modify themselves and spread vector

Issues with this: Fully meshed networks require sending priorities; developed software serial protocol in assembler for performance and robustness

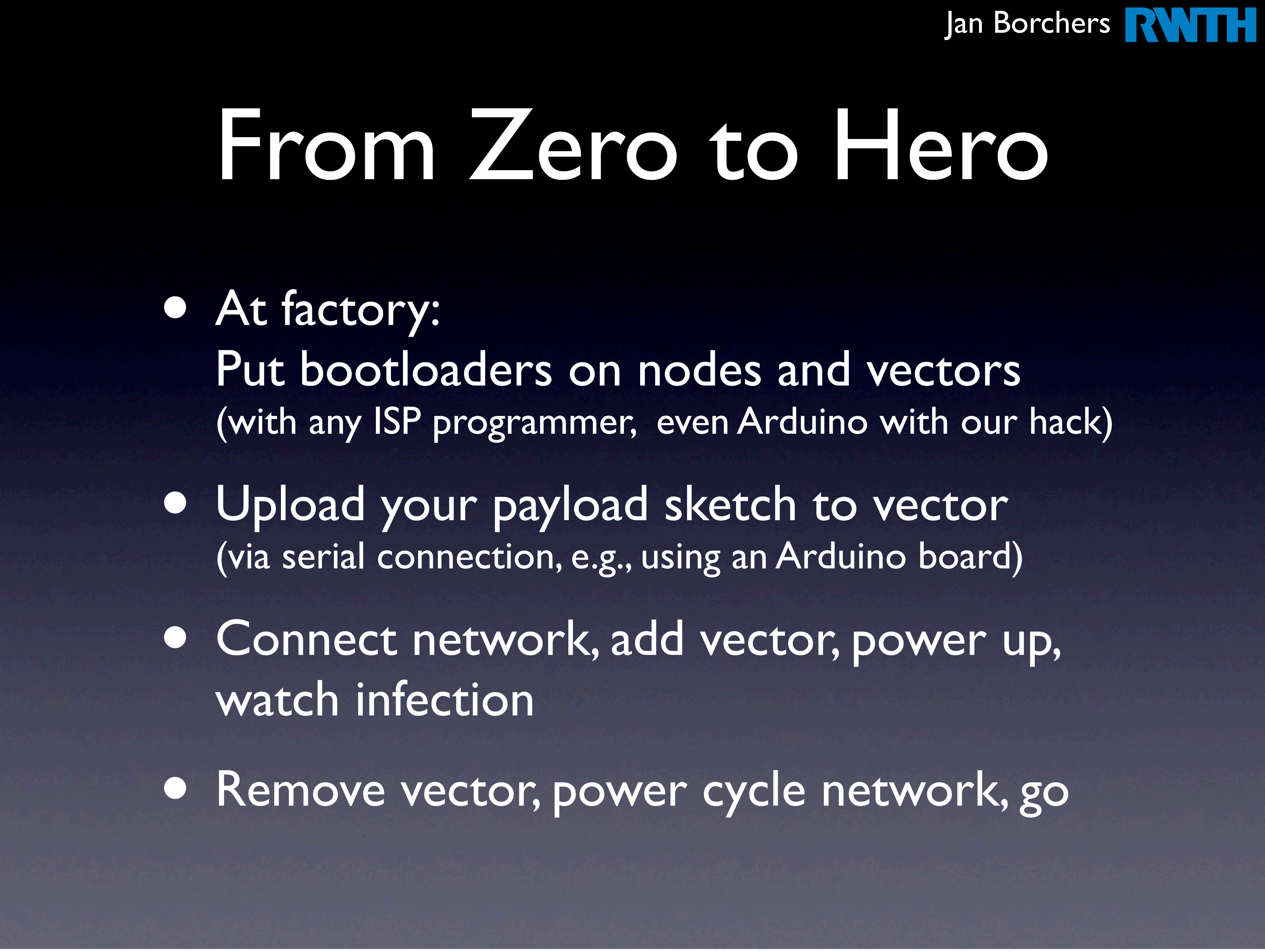
BYNASE



How should nodes communicate once an app is running?

Goals: Make noise your friend, no precise syncing between nodes required

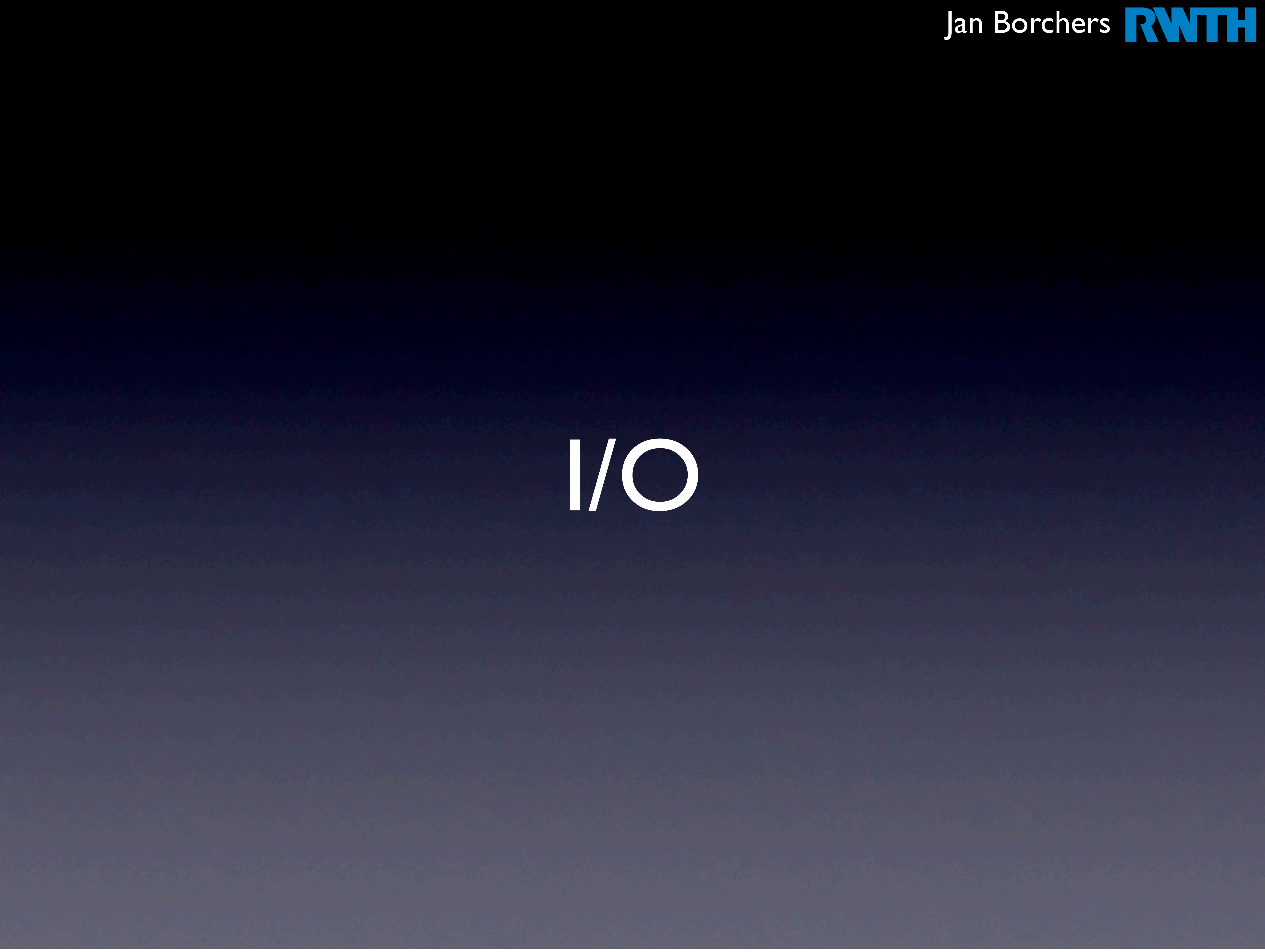
Solution: Cunningham's BYNASE (great for LumiNet LED brightness and color)
- Sender puts pin high statistically depending on value to send
- Receiver samples pin statistically (should happen more often than sender frequency)
- Missing quartz actually helps because it avoids synchronicity between nodes!



From Zero to Hero

- At factory:
  Put bootloaders on nodes and vectors
  (with any ISP programmer, even Arduino with our hack)
- Upload your payload sketch to vector
  (via serial connection, e.g., using an Arduino board)
- Connect network, add vector, power up, watch infection
- Remove vector, power cycle network, go

How to use LumiNet if you know Arduino
Hints:
- Remove jumper on LumiNet board while ISP programming the bootloaders
- Add North_RX (pin 1) to GND jumper while uploading new payload sketches to vector



I/O

LumiNet Sensor Nodes:

- Specially programmed nodes
- Don’t participate in infection
- Like constant periphery
- Inject sensory stimuli into network

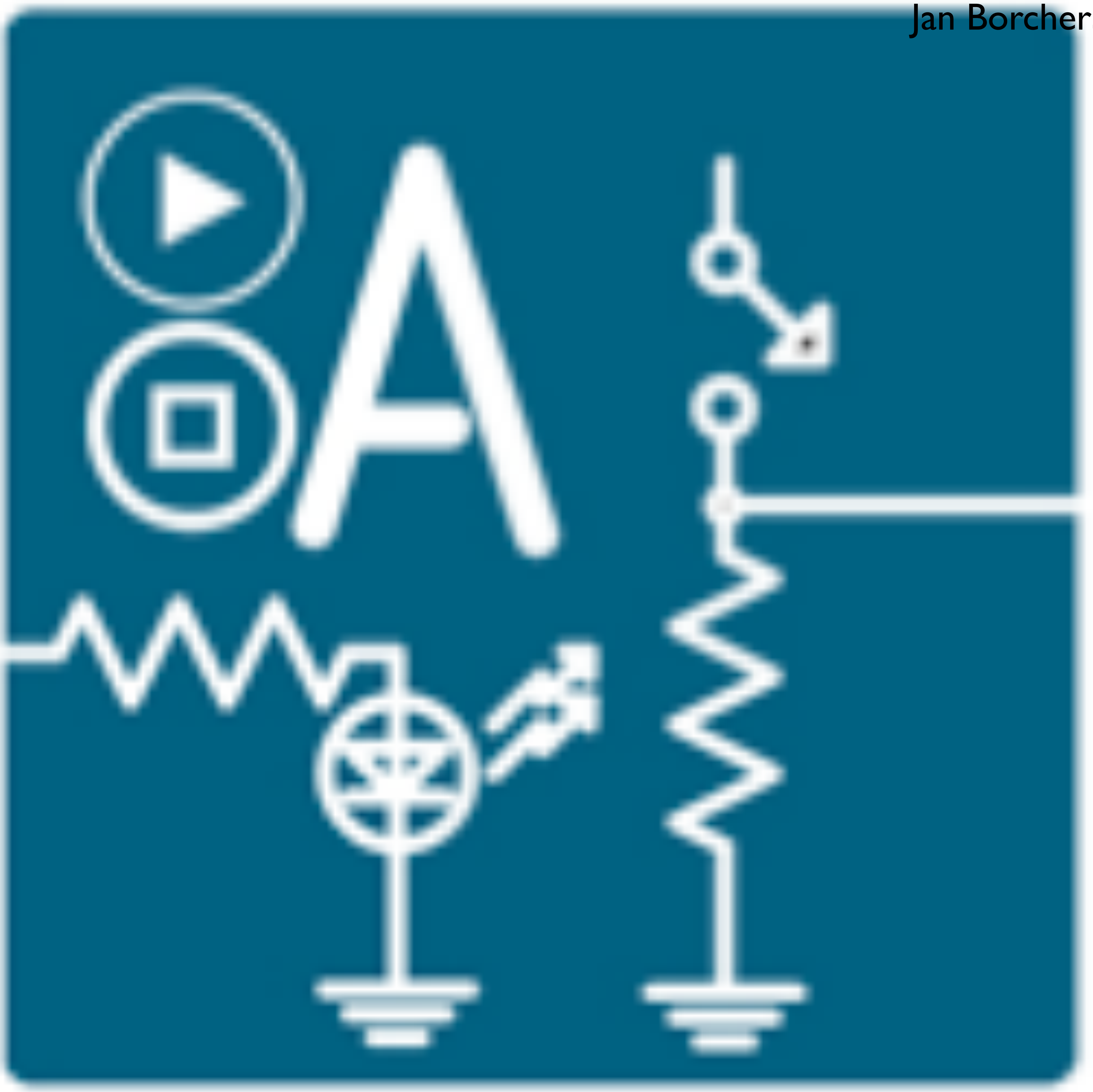
Send/Receive Nodes
- Also constant periphery
- Connect multiple people wearing LumiNet networks
- Bridge the air, e.g., using infrared LEDs and IR sensors

Demo!

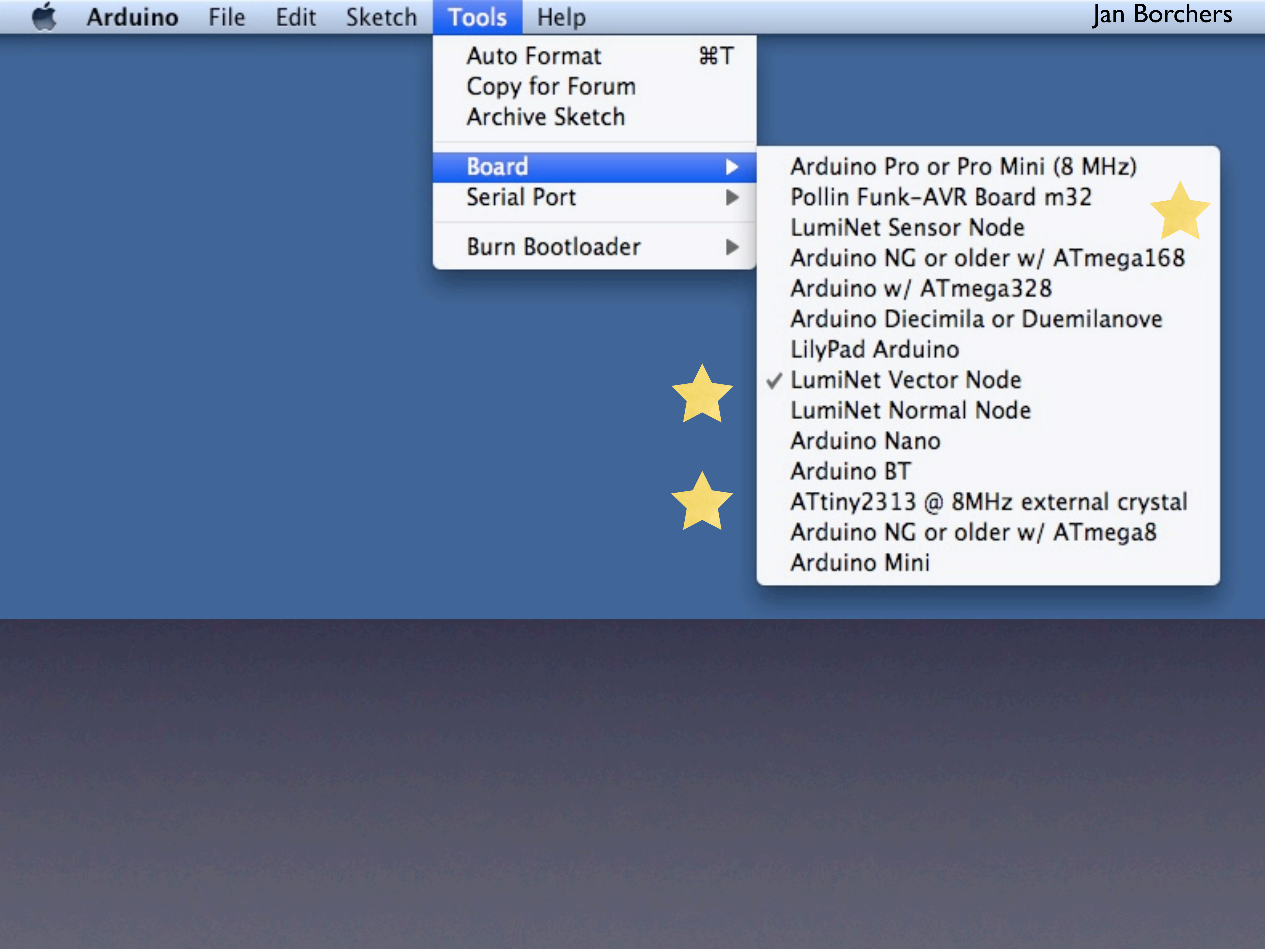
Showing jacket running LightRing code, pressing buttons to trigger ripples
- Nicely shows advantages of separate CPUs on each node (long ripple signals propagate naturally, with very simple code)
- IR sender & receiver on each index finger for propagating the signal down one arm and from the hand across to the other hand and back up the other arm

4x4 node network on the wall:
- Initally runs blue fading pattern
—> shows time drift w/o crystal
- Reprogram by infection, by attaching a viral vector infecting the net with Ant code

Additional demos if time:
- Accelerometer (injects SparkFun accelerometer data into LumiNet network)
- Light Sensor (which uses the LumiNet LED in reverse mode to detect ambient light and thus touch)

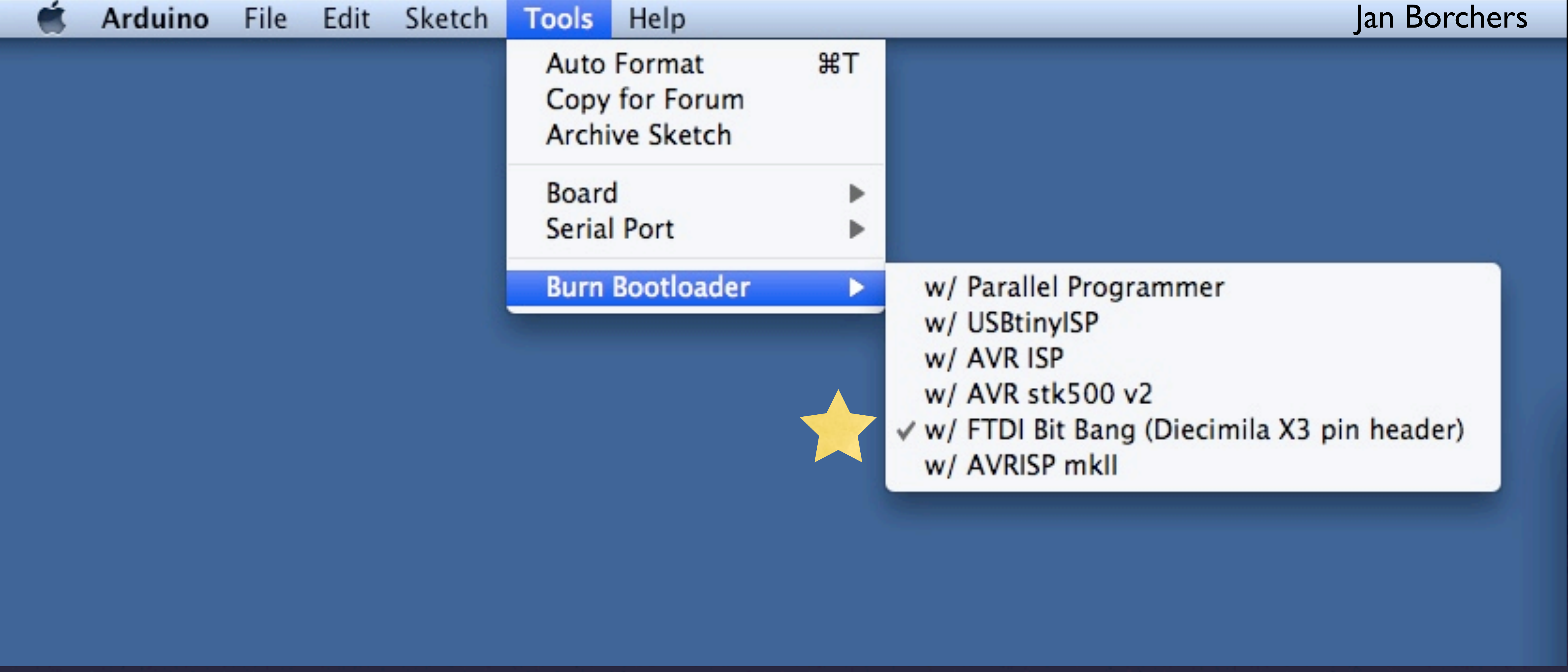


Our changes to Arduino:

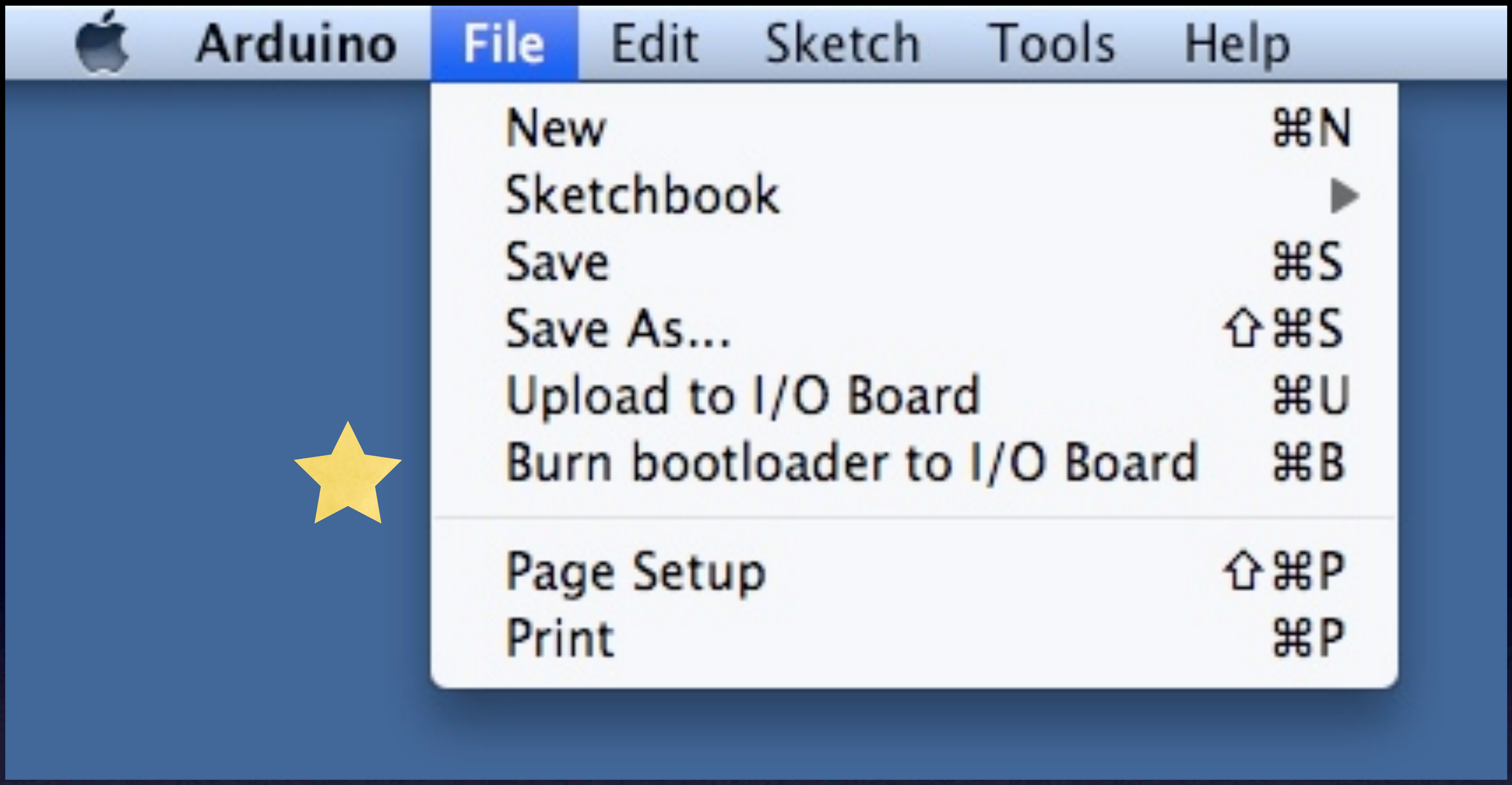


New board definitions in IDE

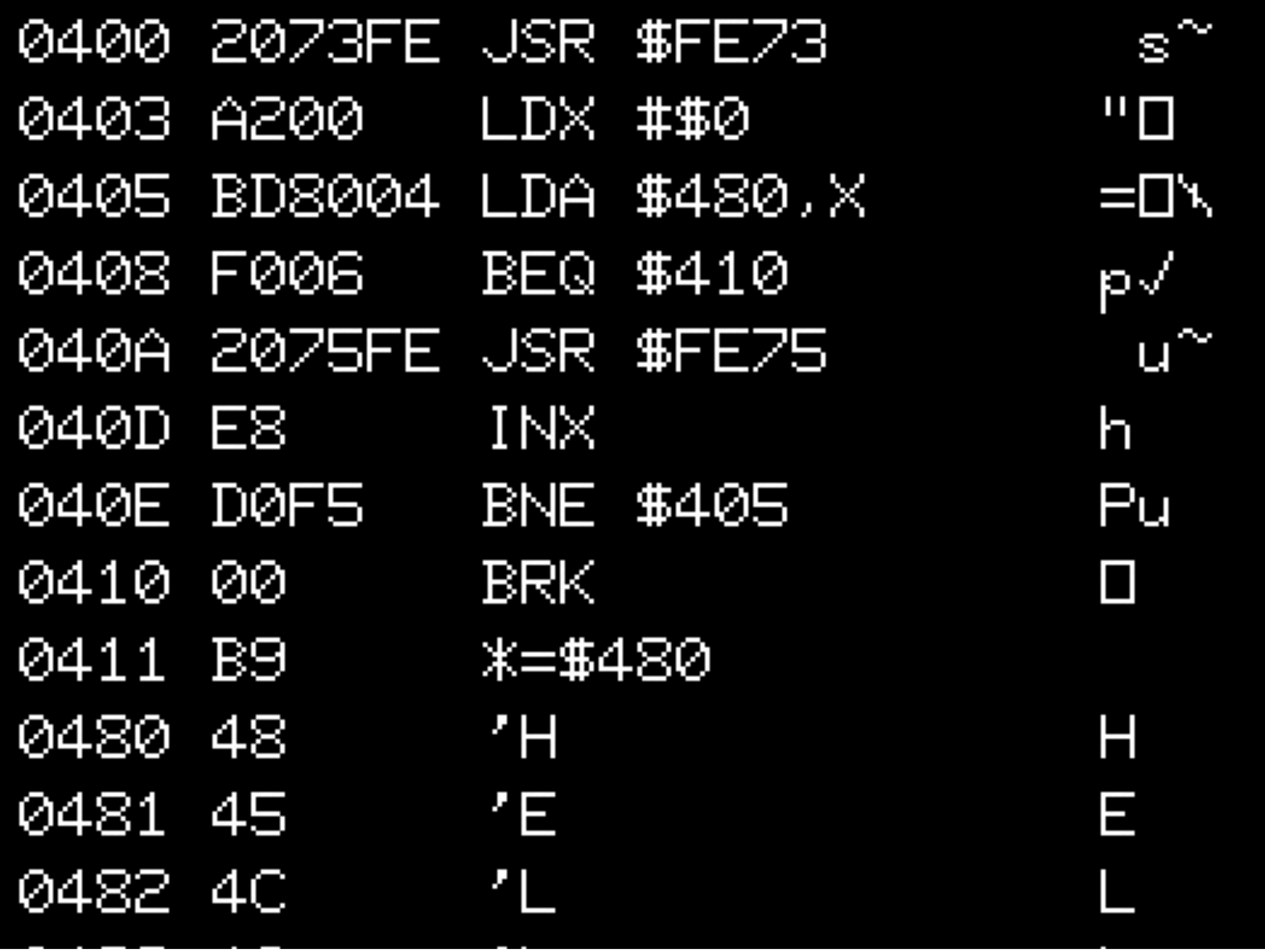
- to put bootloader onto normal LumiNet node or vector node (usually only needed at factory)
- to upload your viral LumiNet sketch to a vector node
- to program new LumiNet sensor nodes



Included an option in the IDE to use your Arduino board as your ISP programmer (for LumiNet bootloaders) if you don’t have one

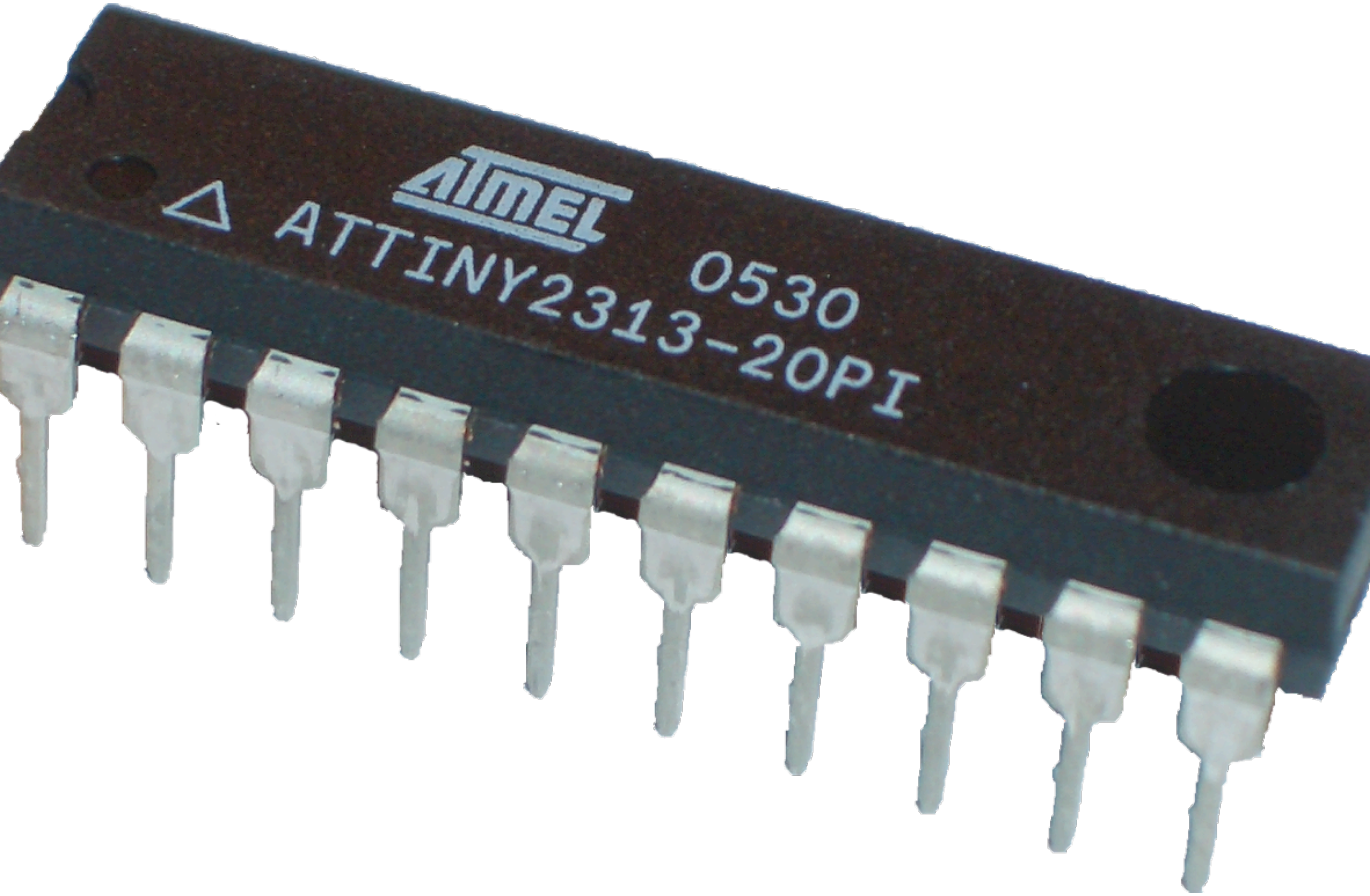


This option makes it easier to just press Cmd-B to put bootloaders on a lot of blank LumiNet nodes (at factory)

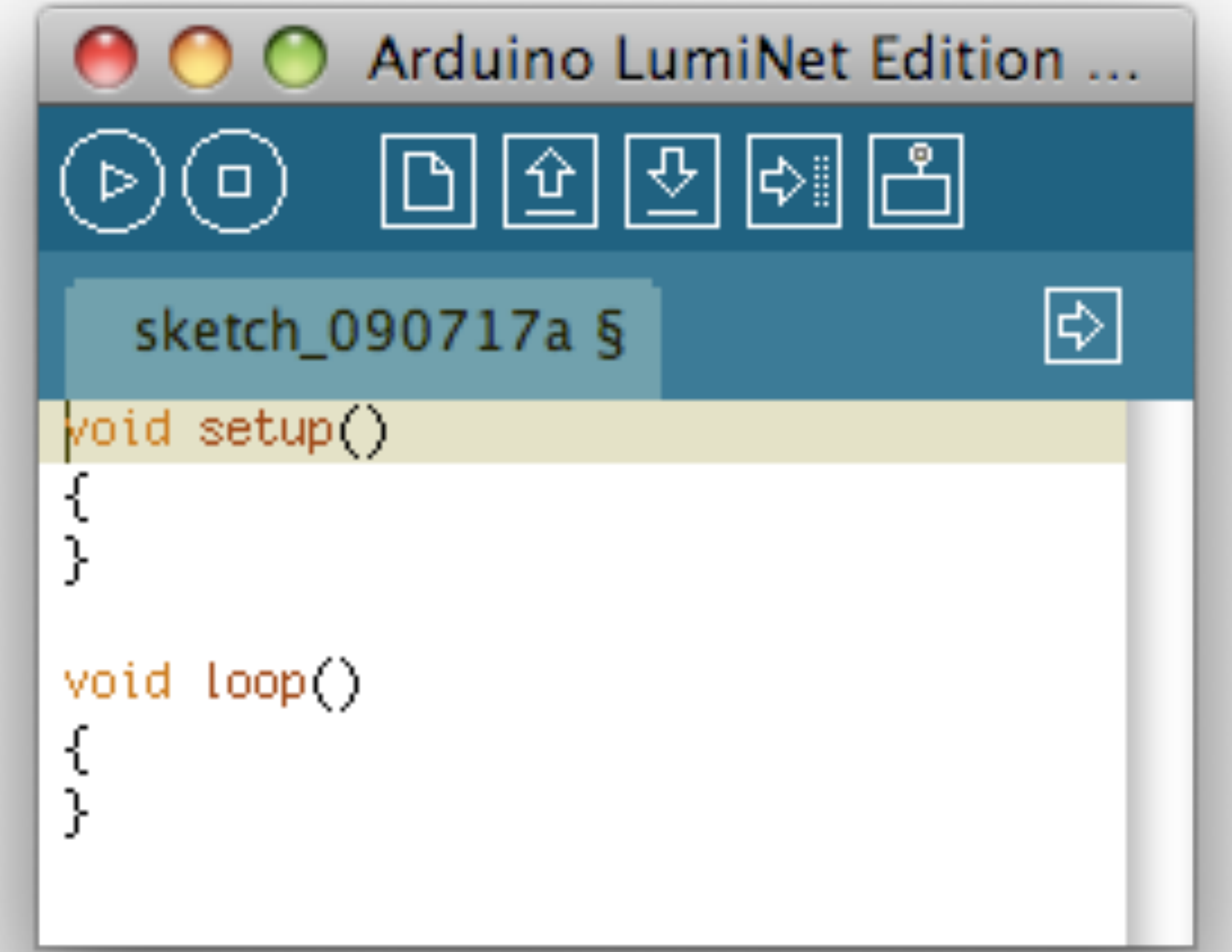


```
0400 2073FE JSR $FE73         s~
0403 A200   LDX #$0          "□
0405 BD8004 LDA $480,X       =□\
0408 F006   BEQ $410         p√
040A 2075FE JSR $FE75         u~
040D E8     INX              h
040E D0F5   BNE $405         Pu
0410 00     BRK              □
0411 B9     *=$480
0480 48     'H               H
0481 45     'E               E
0482 4C     'L               L
```

Assembler support for Arduino cores (and limited for sketches)



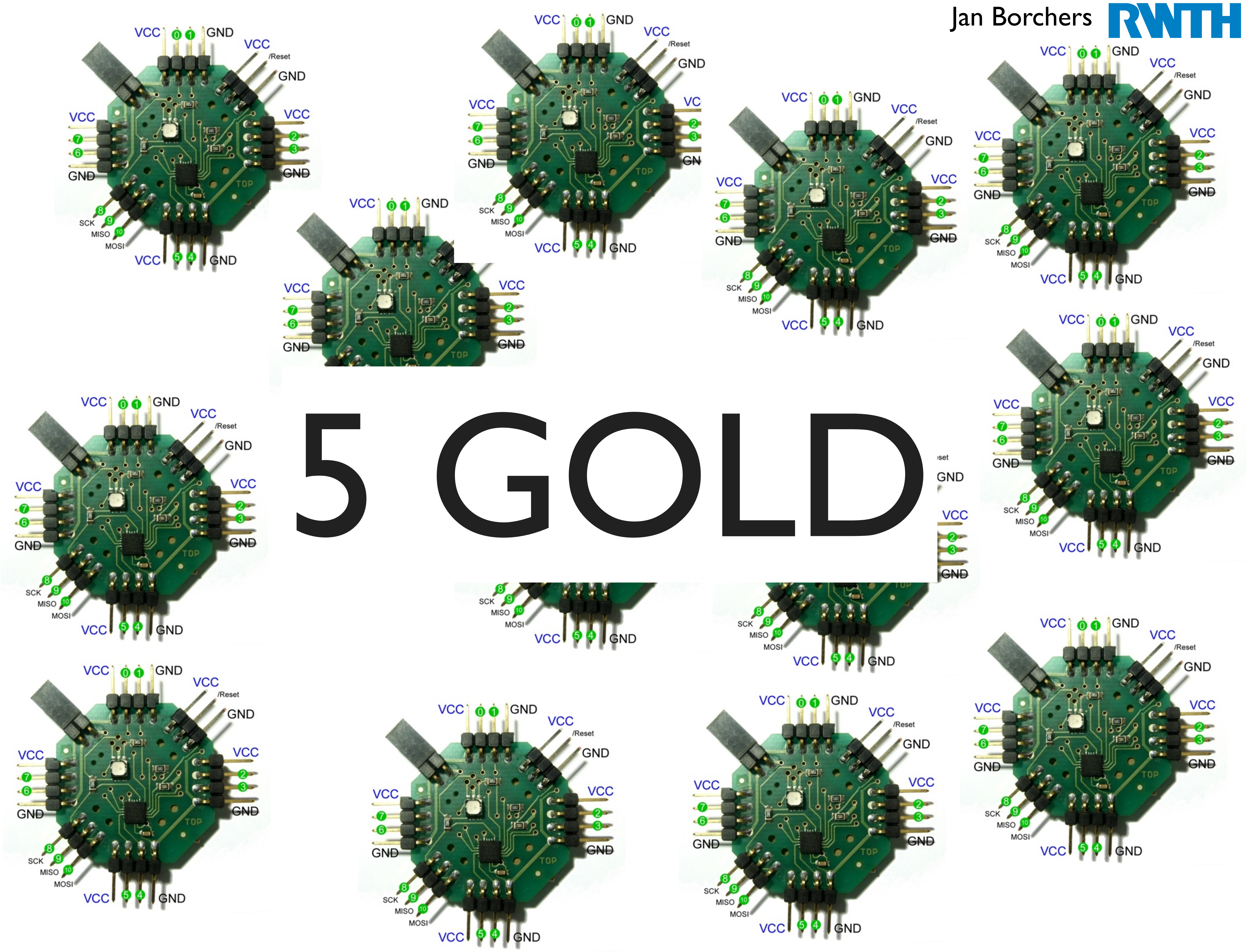
Added ATtiny support to Arduino, including ATtiny84, ATtiny2313; others easy to add
Had to change AVR uploader (avrdude)
Included sensible mappings of Arduino pinouts and features to these less powerful chips



```
void setup()
{
}

void loop()
{
}
```

Also new: An empty template is automatically created for new sketches. :)



5 GOLD

The non-capitalist approach:
- Initial production run (400 boards) was around 8.50 € per node incl. cables
- Rerun would be around 6 €
- Selling them at this conference at 5 € / $ / £ / whatever to keep things simple
- Selling below cost to get the word out and get people interested, rather than keep them to ourselves

Jan Borchers RWTH

Next?

Next steps:
- Write more cool vector code samples with interesting emerging behavior
- Make workflow easier, removing need for jumpers
- Maybe add on-screen simulation to preview parallel behavior of code

Hardware evolution:
- maybe add a crystal for easier deterministic communication
- But then ATtiny runs out of pins
- So move to ATmega (if it doesn’t make it too expensive) - maybe ATmega88
- Reduce to BlinkM board size, use both sides for components, maybe multilayer PCB
- Add a transistor to replace LED jumper
- Use built-in hardware PWM for all 3 LED colors

luminet.cc

Check out this URL for
- free download of our modified Arduino IDE
- Information on the hardware design
- Look for info on buying nodes coming up, or email René