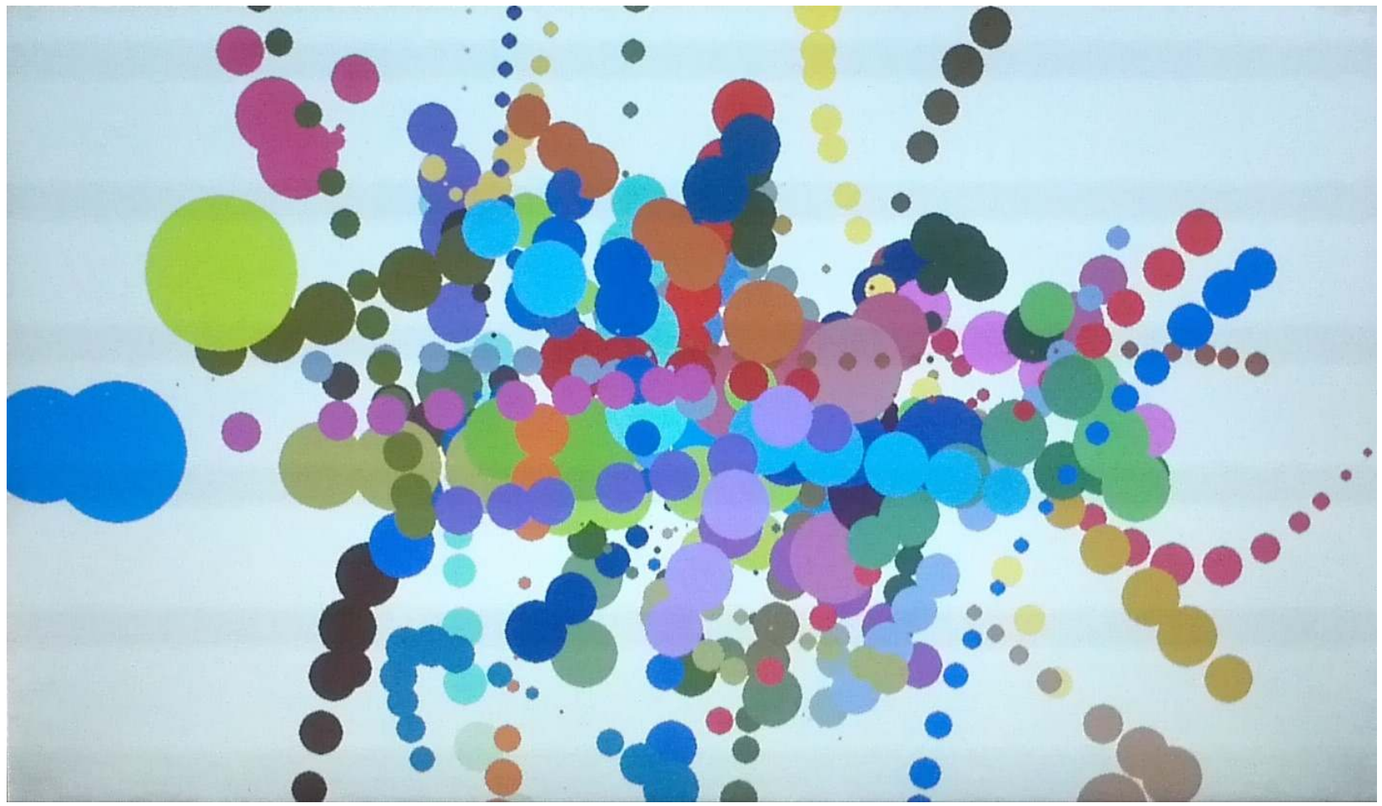
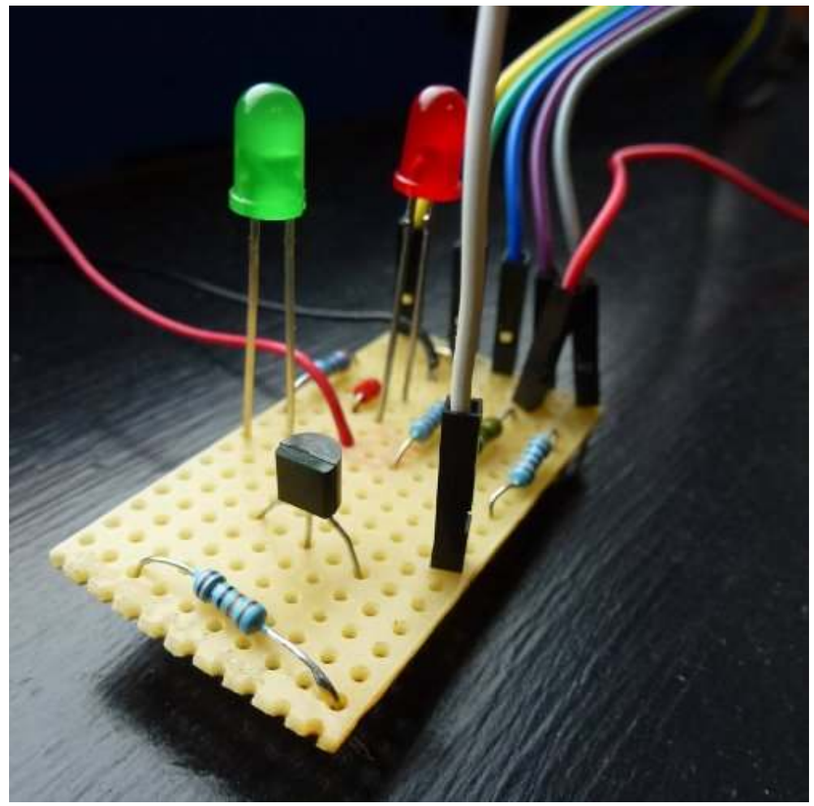
Open { *culture, schools, arts & sciences* } Tech

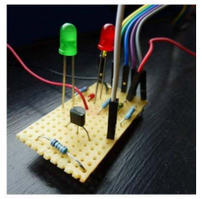




http://Pi.GATE.ac.uk/

Open {*Culture*} Tech

http://Pi.GATE.ac.uk/



inexpensive hardware for interactive arts and sciences

After the success of the Raspberry Pi, more low-cost, low-power computers are arriving. We looked at some of them...

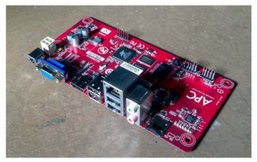
Turns out there are a lot of these things out there. Quite a few predating the Pi. We got our hands on what we could and tried them out. We installed either Debian or Ubuntu Linux then ran through a series of benchmarks.

The full review is available online at http://pi.gate.ac.uk/

Great.

What can I do with them?

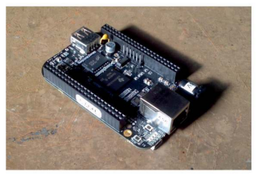
A wide variety of free and open software exists supporting digital art and science. Let's take a look...



APC8750

Big. Lots of ports. VGA output is a nice touch. Pretty snappy response when using it. Kind of a desktop replacement in the end. Me? I'd put it in the kitchen for listening to music.

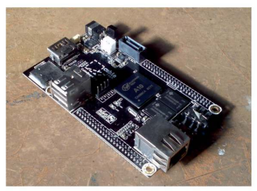
800 MHz ARMv6, 512 MB RAM. $49.



BeagleBone Black

A hacker's board. More GPIO than you can shake a stick at. Really pretty flashing LEDs. Neither of which fix to the fundamental flaw. It's a hacker's board. You need the patience of a saint to get anywhere.

1 GHz ARMv7, 512 MB RAM. $45.



Cubieboard

A lot has been jammed in. There's SATA and infrared. It's a quick setup and works well enough. One word: NAS. Network attached storage. No one that sees this will think of anything else, although there is GPIO.

1 GHz ARMv7, 512 MB RAM. $49.

DreamPlug

So polished, and that's reflected in the price. Still, it's perfect for the next step up from an OpenWRT router. It's got dual-gigabit Ethernet. And eSATA. And audio. And Wi-fi. (It's a bit bigger too.)

1.2 GHz ARMv5, 512 MB RAM. $149.

MK802+ Mini PC

For every computer that's huge there's one that's tiny. This is it. Probably the best responsiveness out of any system tested. Tiny comes at a price though, there's a huge power brick.

1 GHz ARMv7, 1 GB RAM. $59.

Pi Model B

What to say about what's already a classic? The best bit of Raspberry Pi is community. Meet up with other locals interested at a Raspberry Jam. Walk down to Maplin and buy one. It's hard to beat.

700 MHz ARMv6, 512 MB RAM. $35.

Pi Model A

The hardware hacker's Raspberry Pi. Lower power and fewer ports. If you really need those ports it's cheaper to get a model B, but otherwise the A is it's smaller cousin.

700 MHz ARMv6, 256 MB RAM. $25.

TonidoPlug2

This is really an appliance and in Piano black looks it. It takes a SATA hard drive and is a NAS. That you can install your own software onto. Beware though - there's nothing but SATA, USB, and Wi-fi.

800 MHz ARMv5, 512 MB RAM. $119.

GIMP

The GNU Image Manipulation Program. Name says it all. Draw to your heart's desire.

Inkscape

Inkscape: Draw Freely. Vector (scalable) graphics. This poster was made with this!

Processing

Visual arts programming. Used here at the university in the Music department!

Firefox

Mozilla Firefox Web Browser. Based on Netscape. Remember that?

Audacity

Audacity: Free Audio Editor and Recorder. Also available to music students here!

Python

Python Programming Langague. A simple but powerful way to work.

Octave

GNU Octave. An advanced interactive calculator.

LibreOffice

LibreOffice from the Document Foundation. A complete office suite, based on Sun's StarOffice.

Linux

Linux is becoming more main mainstream. Maybe you've heard of these?

Ubuntu

Android

Open {*Schools*} Tech

(School's ***In***!)

Program or be programmed:

- the world is saturated with computer code: everything from fridges to singing birthday cards
- the people who write our code, create our world
- shouldn't everyone have the chance to create their own world, and not simply wander through the creations of others?

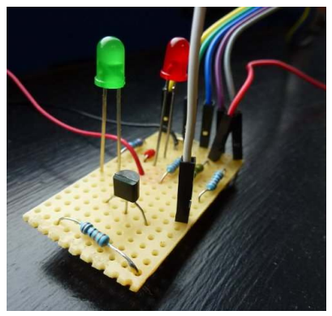
Computers in the UK Curriculum

- draft curriculum for schools: programming is back!
- good news for other subject areas: thinking logically, finding patterns in data can open our eyes to the world around us — to how birds flock, or populations grow, or economic crises play out
- develop our creativity, our concentration and our artistry
- asks schools to build 'a critical understanding of technology's impact on the individual and society'.

The Rasberry Pi

- a computer that schools don't need to worry about kids breaking, or abusing, or using to endanger their safety — providing they're careful with the soldering iron!
- designed to spend much of its life in pieces
- runs a free, open source operating system (Linux)
- not a finished, out-of-the-box product — it's a project, or a thousand projects — a robot remote control, a programmable camera, a parent alarm, a media center, an electronic paint brush

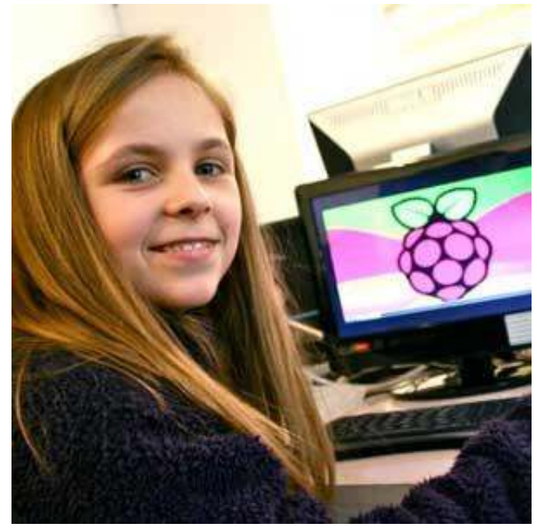
Google donated $1,000,000
for Pi hardware in schools:



http://Pi.GATE.ac.uk/

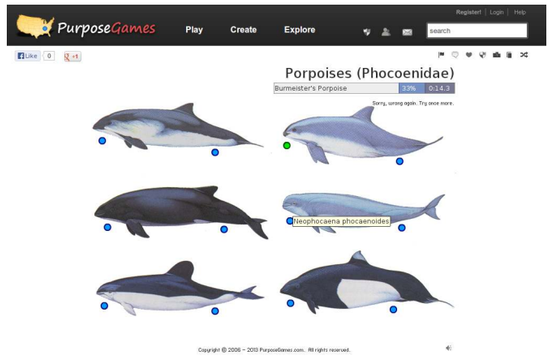
"At Boots science is for boys and pink princess toys are for girls"

Amy Mather (14-year-old Pi hacker) at Wuthering Bytes, September 14th 2013.



What to do?

- Barbie the programmer?! Princess Pi? Pink Electronics?
- Games with a porpoise?



Wanted:

- Better role models
- Better opportunities
- End discrimination

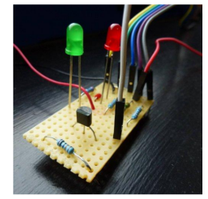
Get involved:

STEMNET
Science, Technology, Engineering and Mathematics Network

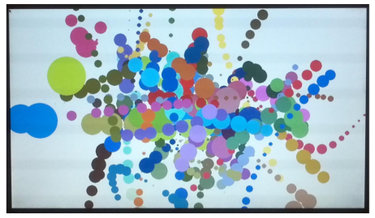
{code club}

Open {Art} Tech

Pi Brush: Python Source Code



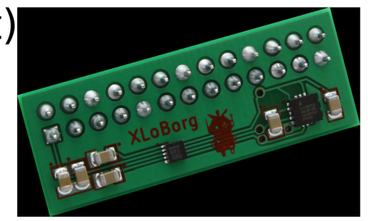
Display (Server)



mid-development photograph

```
import socket
import select
import pygame
import time
import numpy
import math
import datetime
import random

# ==============
# initialization
# ==============

# port to listen on
port = 5005

# setup networking
sock = socket.socket(socket.AF_INET,
        socket.SOCK_DGRAM) # UDP
sock.bind(("0.0.0.0", port))
sock.setblocking(0)

# screen solution
XRES = 1920
YRES = 1080

# setup display
pygame.init()
screen = pygame.display.set_mode((XRES, YRES))
# reset to white
screen.fill((255, 255, 255))
# push white
pygame.display.flip()

# length of moving average array
AL = 20

# accelerometer storage for moving average
AXa = numpy.zeros((AL, 1))
AYa = numpy.zeros((AL, 1))
AZa = numpy.ones((AL, 1))

# array index for accelerometer data
Ai = 0

# store gravity when fast is detected..
GX = 0
GY = 0
GZ = 1

# polar gravity
PGR = 0
PGA = -math.pi/2
PGB = 0

# screen gravity
PSGR = 1
PSGA = -math.pi/2
PSGB = 0

# accelerometer values
AX = 0
AY = 0
AZ = 0

# rotated for screen accelerometer values
GAX = 0
GAY = 0
GAZ = 0
last_G = 0

# timing information
last_time = time.time();

# brush info
BX = 0 # position
BY = 0
VX = 0 # velocity
VY = 0
P = 0 # amount of paint on brush
S = 0 # distance brush has traveled
last_stroke = 0

# seed random number generator
random.seed(time.time())

# =========
# functions
# =========

def polar(X, Y, Z):
    x = numpy.linalg.norm([X, Y, Z])
    if (x > 0):
        y = -math.atan2(Z, X)
        z = math.asin(Y / x)
    else:
        y = 0
        z = 0
    return (x, y, z)

def cartesian(X, A, B):
    x = 0 # don't bother - isn't used
    y = X * math.sin(B) * math.sin(A)
    z = X * math.cos(B)
    return (x, y, z)


# ============
# main program
# ============

fast = 0
notfast = 0
running = 1
while running:

    # move time forward
    dt = time.time() - last_time
    last_time = time.time()

    # no changes made to the screen so far
    draw = 0

    # check for keyboard input
    event = pygame.event.poll()
    if event.type == pygame.QUIT:
        running = 0
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_q:
            running = 0
        elif event.key == pygame.K_r:
            screen.fill((255, 255, 255))
            draw = 1
        elif event.key == pygame.K_s:
            filename = "%i.png" % time.time()
            pygame.image.save(screen, filename)


    # ==================
    # networking & sensor
    # ==================

    result = select.select([sock], [], [], 0)
    if len(result[0]) > 0:
        # read in data
        data = result[0][0].recvfrom(1024)
        a = data[0].split(",")
        AXa[Ai] = float(a[0])
        AYa[Ai] = float(a[1])
        AZa[Ai] = float(a[2])
        Ai = Ai + 1
        if Ai == AL:
            Ai = 0

        # moving averages for acceleration
        AX = numpy.sum(AXa) / AL
        AY = numpy.sum(AYa) / AL
        AZ = numpy.sum(AZa) / AL

        # combined acceleration for
        # working out resting gravity
        A = math.fabs(numpy.linalg.norm([AX,
                AY, AZ]) - 1)

        # in a slow moment store most recent
        # direction of the gravitational field
        if A < 0.02 and (last_time - last_G) > 0.12:
            GX = AX
            GY = AY
            GZ = AZ
            (PGR, PGA, PGB) = polar(GX, GY, GZ)
            last_G = last_time

        # rotate to screen coordinates
        # and subtract gravity
        (PAR, PAA, PAB) = polar(AX, AY, AZ)
        (GAX, GAY, GAZ) = cartesian(PAR,
                PAA - PGA + PSGA, PAB - PGB + PSGB)
        GAZ = GAZ - PGR


    # ==================
    # paintbrush physics
    # ==================

    # acceleration detection for paint strokes
    A = numpy.linalg.norm([GAY, GAZ])

    # detect moving quickly
    if A > 0.4 and fast != 1 and \
            last_time - last_stroke > 0.5:
        fast = 1
        notfast = 0
        scale = random.random() * 0.5 + 0.5
        BX = YRES * GAY * scale + XRES / 2 + \
                random.randint(-XRES/4, XRES/4)
        BY = YRES * GAZ * scale + YRES / 2 + \
                random.randint(-YRES/7, YRES/7)
        VX = 0
        VY = 0
        P = 100
        COLR = random.randint(0, 255)
        COLG = random.randint(0, 255)
        COLB = random.randint(0, 255)

    # detect stopping
    if fast == 1 and (A < 0.1 or ((BX > (XRES + 200) \
                or BX < -200) and (BY > (YRES + 200) \
                or BY < -200)) or P <= 0):
        notfast = notfast + dt
        if notfast >= 0.12:
            fast = 0
            BX = 0
            BY = 0
            last_stroke = last_time

    if fast == 1:
        # accelerate the paint brush
        VX = VX - GAY * dt * 170
        VY = VY - GAZ * dt * 170
        BX = BX + VX * dt * 120
        BY = BY + VY * dt * 120

        # add splotches.... high velocity big
        # splotches far apart, low small close
        if P > 0:
            V = numpy.linalg.norm([VX, VY])
            S = S + V
            d = A * random.randint(3, 5) * 25 + V
            if S > d:
                S = S - d
                P = P - pow(A*4, 2) * math.pi
                pygame.draw.circle(screen, (COLR, COLG,
                        COLB), (int(BX), int(BY)), int(A*45))
                draw = 1

        # push updates to the screen
        if draw == 1:
            pygame.display.flip()

pygame.quit()
```

Remote (Client)



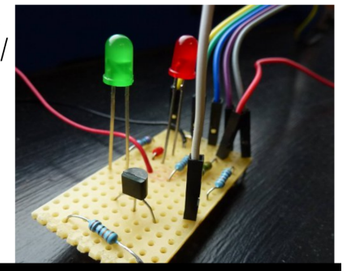
accelerometer sensor unit

```
import socket
import XLoBorg
import time

# network stufff
server = "modelb"
port = 5005

# setup for the accelerometer
XLoBorg.printFunction = XLoBorg.NoPrint
XLoBorg.Init()

# make the socket connection
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
while True:
    message = '%+01.4f,%+01.4f,%+01.4f' \
            % XLoBorg.ReadAccelerometer()
    sock.sendto(message, (server, port))
    time.sleep(0.005)
```

http://Pi.GATE.ac.uk/

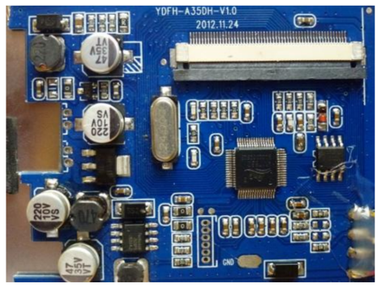
Open {*Photography*} Tech

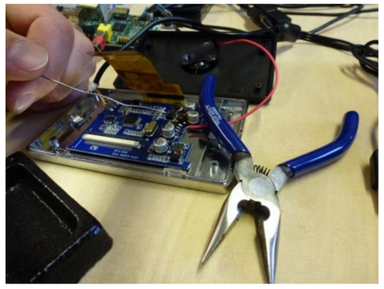
http://Pi.GATE.ac.uk/



It's your data, but do you own it?!

- all your email is bcc. BoredYoungSpies@nsa.gov
- you're a Facebook friend of NoSenseOfHumour@gchq.gov.uk
- anything the NSA has, Mossad has

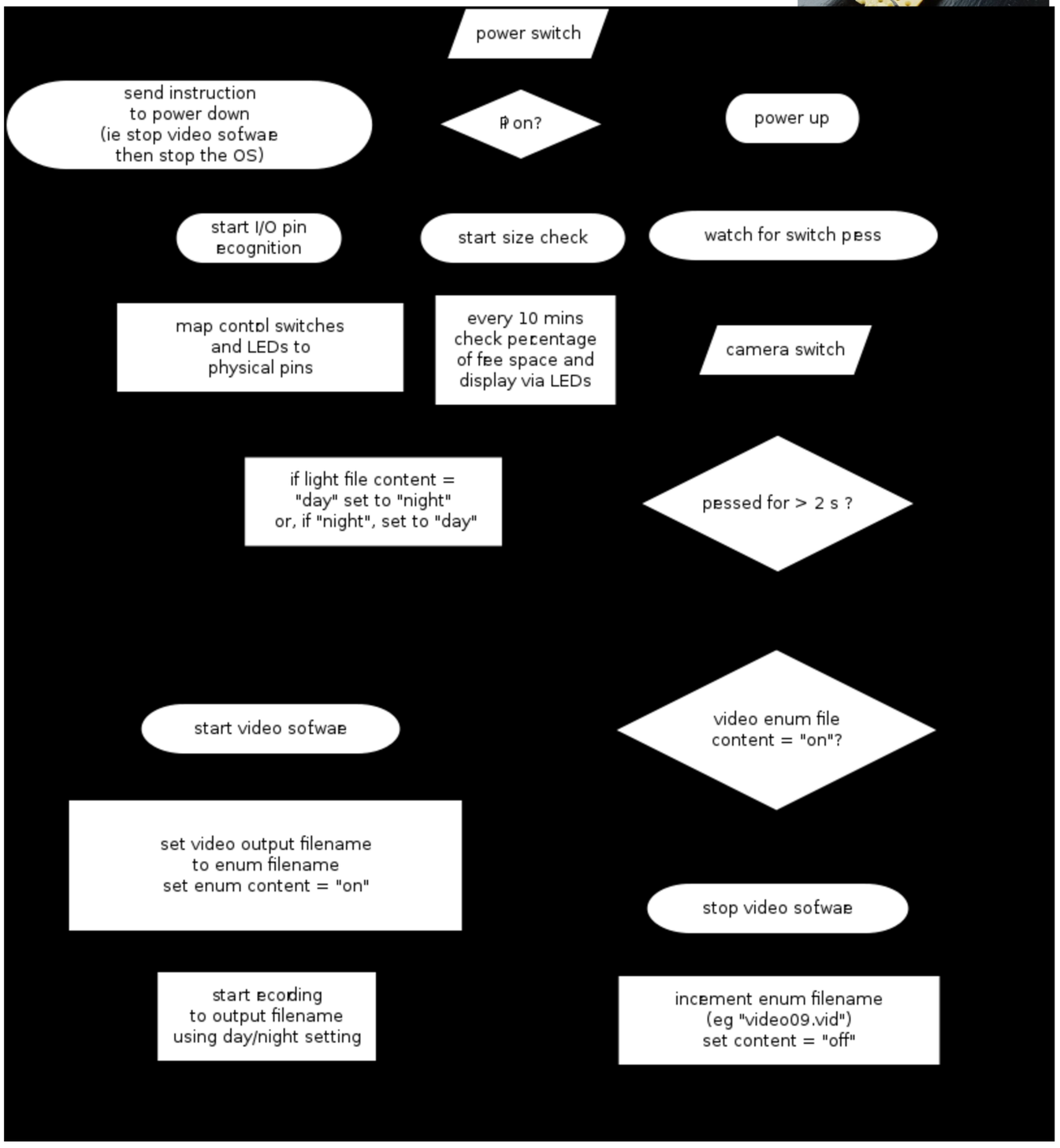




SnowCam: a **Pi-based camera** that is:

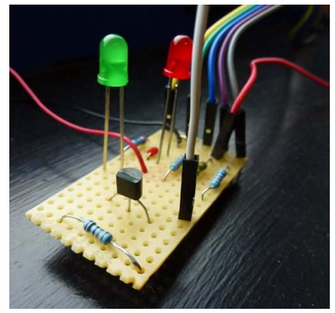
- programmable
- schools-friendly
- eco-friendly
- DRM-free
- NSA, GCHQ un-friendly

Thinking algorithmically...



Open Tech... *Future Making*

http://Pi.GATE.ac.uk/



(an exploration of engineered art and social science with open-source hard/soft/etc-ware)

Future Making will be a rolling event over two days with around a dozen exhibits plus an hourly short stage show. The target audience is anyone curious, with secondary schools invited to attend if the timing of the event permits. The performers are expected to be engineers, musicians, social scientists, dancers, hackers and visual artists.

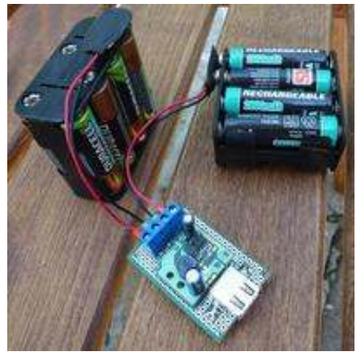
There is a new conjunction emerging around open hardware, maker culture, and art. Parallels exist with the explosive change in pop culture pressaged by punk in the late 1970s: from the Homebrew Computer Club trading circuits and information; from Iain M. Banks' legacy of utopian technofutures, blending electro-augmented humanity with benevolent AI; from the wearable electronics kits (e.g. from Adafruit.com — echoing Lady Ada Lovelace, inventor of the computer algorithm); to the latest rounds of sampling and remix.

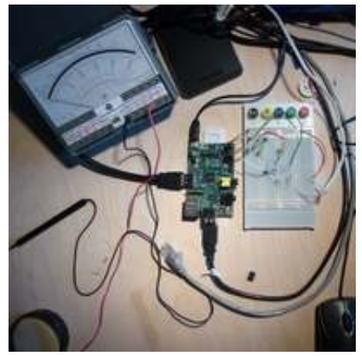
Open-source hardware allows people to make their own robots, cameras, electrocardiagraph machines and even full computers by downloading schematics and building — incorporating any changes they need — and, typically, free open-source software is available to run these projects. 3D printers have helped this adoption of the open-everything ethos.

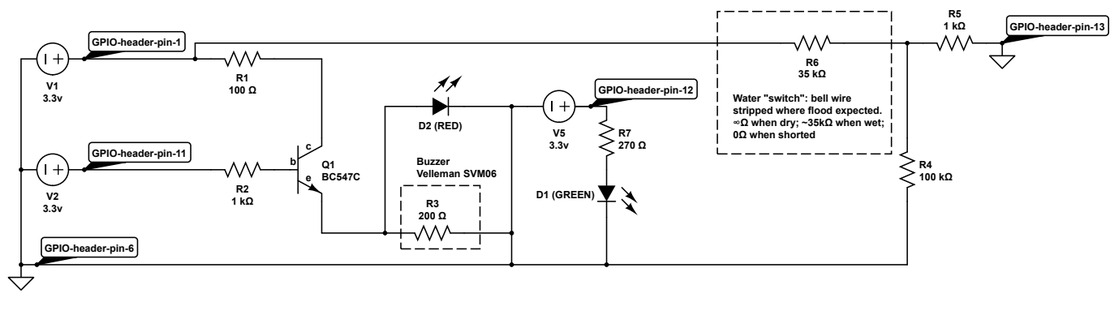
As RaspberryPi.org and many other sites testify, there are stacks of DIY projects based on the Raspberry Pi, and the flood shows no sign of slowing. The Pi is a small cheap computer which is very easy to cobble together with other devices. It has a large community behind it and is one of the drivers behind proposed changes in the schools ICT curriculum.

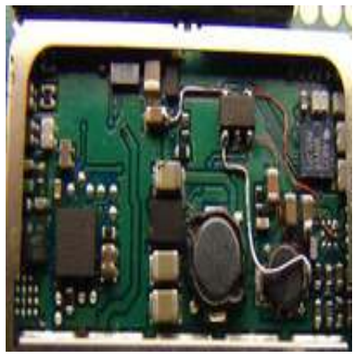
As part of a trend to open hardware and maker culture the Pi is also a vehicle for localist responses to climate change and peak oil: the more we manufacture locally the more secure we are (the Pi is made in Wales); the less energy we use the less carbon we pump into the atmosphere (the Pi is low power).

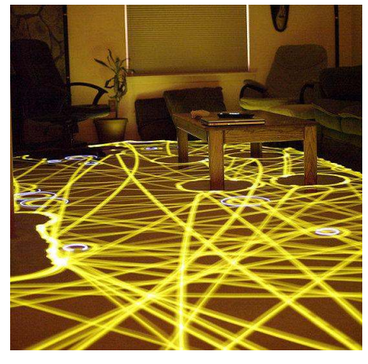
The cases of Edward Snowden, Chelsea Manning and Julian Assange, however, readily attest to questions around ownership of data, and 'patent-trolling' is slowing down the adoption of cheaper industrial processes — we will engage in an age-appropriate exploration of this topic in the context of the international 'broadcast' nature of the internet.

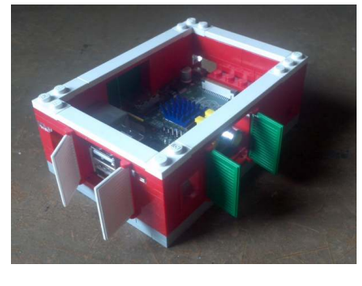


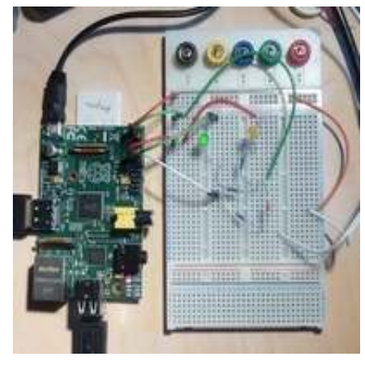












To be continued...

<< paste your project here >>