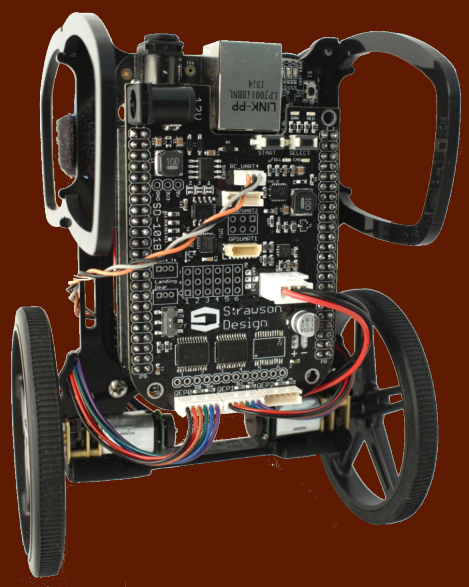


REAL-TIME PROGRAMMING WITH BEAGLEBONE PRUS

beagleboard.org

Jason Kridner, jkridner@beagleboard.org

Home info link

- http://elinux.org/Ti_AM33XX_PRUSSv2

What are PRUs

- “Programmable Real-time Units”
- 32-bit RISC processors at 200MHz with single-cycle pin access for hard real-time
- Optimized for packet processing/switching and software implementations of peripherals
- Part of the PRU-ICSS, “Industrial Communications SubSystem”

Why and when to use PRUs

- Free from running on an operating system, so can be dedicated to a function
- Real-time because it can't be interrupted from its given task by other tasks
  - Interrupts are simply registered into an event register
  - Operations scheduled in an event loop
- Low, low, low latency from input to output
  - Zero-depth pipeline
- You can't interface an external MCU to DDR memory so fast!

Examples usage

- Tight control loops
  - Driving motors in a mobile robot, CNC machine or 3D printer
- Custom protocols
  - WS28x LEDs, DMX512, ...
  - EtherCAT, ProfiBUS, ProfiNET, ...
- Soft peripherals
  - PWM, UART (LEGO), ...

Example projects (see wiki page)

- 6502 memory slave
- DMX512
- WS28xx LEDs (OLA, LEDscape)
- MachineKit (Madison, WI – June 28!!)
- GSoC: pruspeak, BeagleLogic
- GCC, Forth, ...

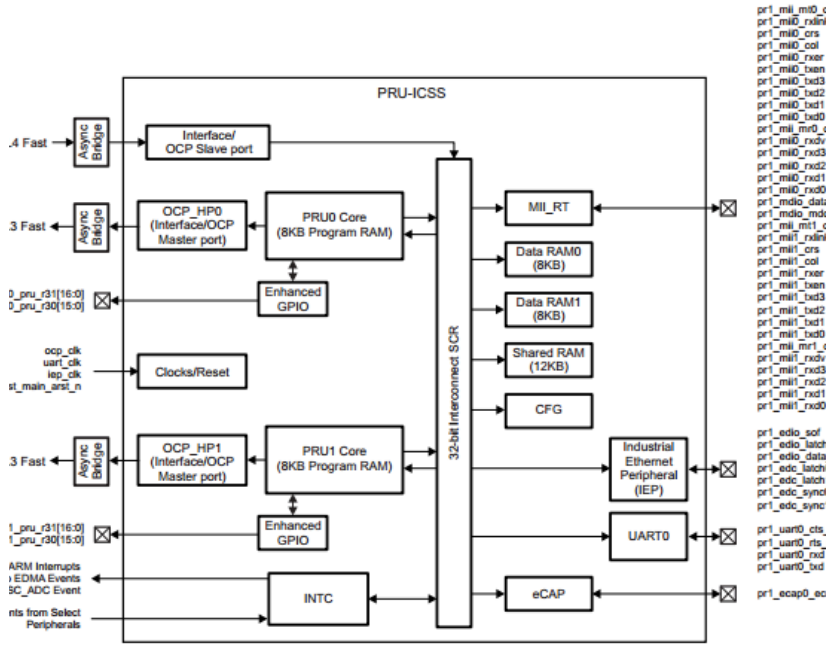
Why isn’t it more popular

- Like lots of amazing things, started as something very focused --- you can’t keep a good thing down
- Tools still being developed
  - C compiler
  - Linux drivers
- Libraries

PRUSS architecture details

- 2 cores at 200MHz each
- Memory
  - 8kB program each
  - 8kB data each
  - 12kB data shared
  - Access through L3 to external memory and peripherals

Figure 2. PRU-ICSS Integration



Extra peripherals in subsystem

- Shift serial capture/send
- Parallel capture/send

25 PRU low-latency I/Os

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
PRU0_7	25	26	PRU1_16 IN
PRU0_5	27	28	PRU0_3
PRU0_1	29	30	PRU0_2
PRU0_0	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
PRU0_6	41	42	PRU0_4
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
PRU0_15 OUT	11	12	PRU0_14 OUT
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	PRU1_13
PRU1_12	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
PRU1_8	27	28	PRU1_10
PRU1_9	29	30	PRU1_11
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
PRU1_6	39	40	PRU1_7
PRU1_4	41	42	PRU1_5
PRU1_2	43	44	PRU1_3
PRU1_0	45	46	PRU1_1

Accessing the other peripherals

- Yes, you can!
- The “L3” bus is exposed, so you can directly poke all of the peripheral registers
- Be careful! --- be sure the main CPU isn’t trying to access them at the same time, so you need to manually disable access to them on the main CPU

PRU Linux drivers

- uio_pruss – upstream
  - Memory mapped PRU control registers from userspace
  - Interfaces entirely in userspace library
- Various remote_proc implementations
  - “Proper” Linux abstraction of a processor
  - Lots of different “standard” interfaces

Why is Linux so painful

- Linux wants to abstract the hardware
- Is it really future-proof?
- Who is controls?

PRU tools – a work in progress

- TI C compiler
- Forth
- GCC
- pruspeak and remote_proc
- StarterWare (really?? Yes.)

PRU pin muxing

- Devicetree Overlays
- cape-universal

Questions!

- http://elinux.org/Ti_AM33XX_PRUSSv2
- jkridner@beagleboard.org (but, I don't answer questions if beagleboard@googlegroups.com isn't in copy)
- Follow @jadon and @beagleboardorg

What does your robot need to do?

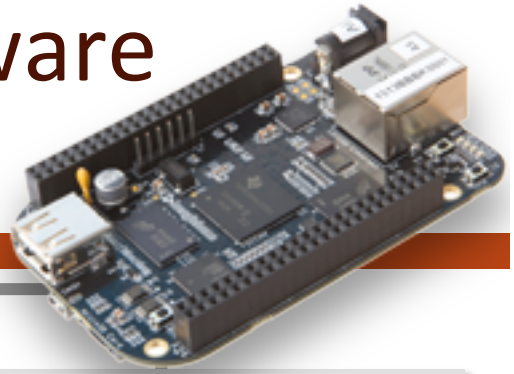
Basics (Microcontroller-like)

- Analog sensors
  - Range finders, controls
- Digital/serial sensors
  - Collision, controls
  - Motion, orientation
  - Wheel rotation
- Motors
  - Servo, DC, stepper

Extras (Computer-like)

- Networking
  - Web controls
  - Streaming data
  - Social media
- Heaving processing
  - Vision
- High level languages
  - Python, JavaScript, Ruby, ...

BeagleBone Black: Open hardware computer for makers



Truly flexible open hardware and software development platform

All you need is in the box

Proven ecosystem from prototype to product

BeagleBone Black

- Ready to use: $45
- 1 GHz performance
- On-board HDMI to connect directly to TVs and monitors
- More and faster memory now with 512MB DDR3
- On-board flash storage frees up the microSD card slot
- Support for existing Cape plug-in boards

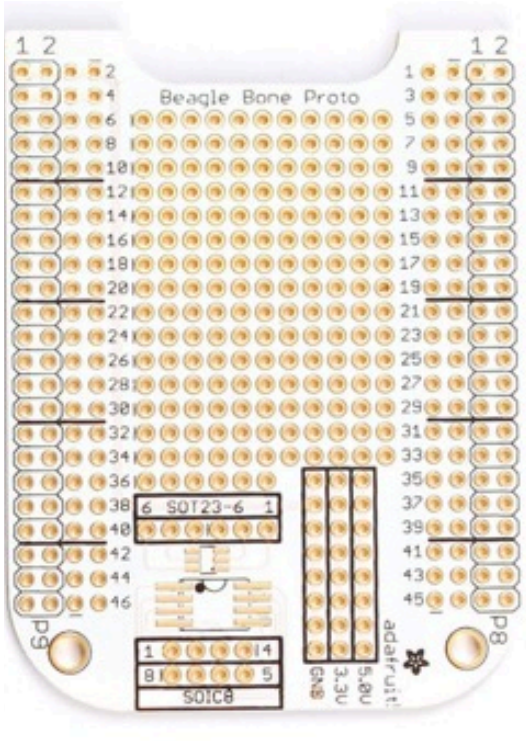
Most affordable and proven open hardware Linux platform available

beagleboard.org

BeagleBone Capes

http://beaglebonecapes.com

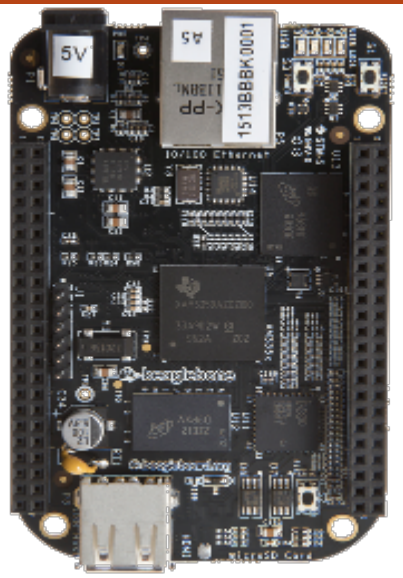
- Just another word for a daughterboard
- Many have a cape-like formfactor
- Up to 4 stacked, depending on resources used



Cape Expansion Headers

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
UART4_RXD	11	12	GPIO_60
UART4_TXD	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
SPI0_CS0	17	18	SPI0_D1
I2C2_SCL	19	20	I2C2_SDA
SPI0_D0	21	22	SPI0_SCLK
GPIO_49	23	24	UART1_TXD
GPIO_117	25	26	UART1_RXD
GPIO_115	27	28	SPI1_CS0
SPI1_D0	29	30	GPIO_122
SPI1_SCLK	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPPWM0
DGND	43	44	DGND
DGND	45	46	DGND

beagleboard.org

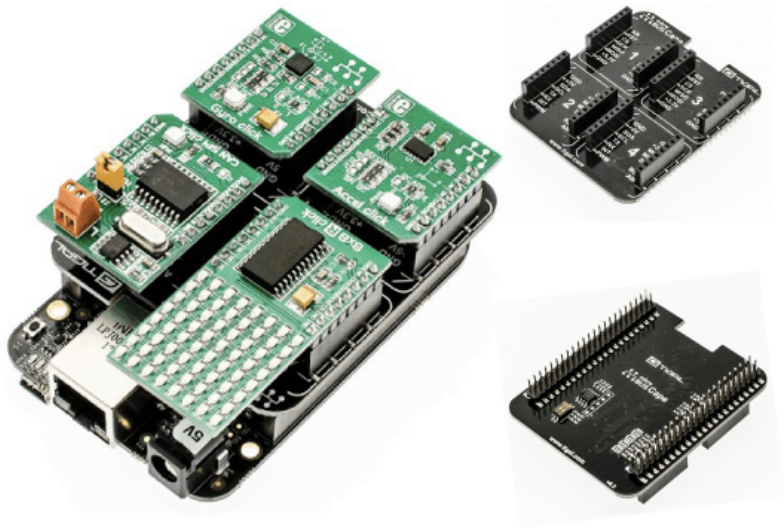


LEGEND
Power/Ground/Reset
Available Digital
Available PWM
Shared I2C Bus
Reconfigurable Digital
Analog Inputs (1.8V)

DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DAT0	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

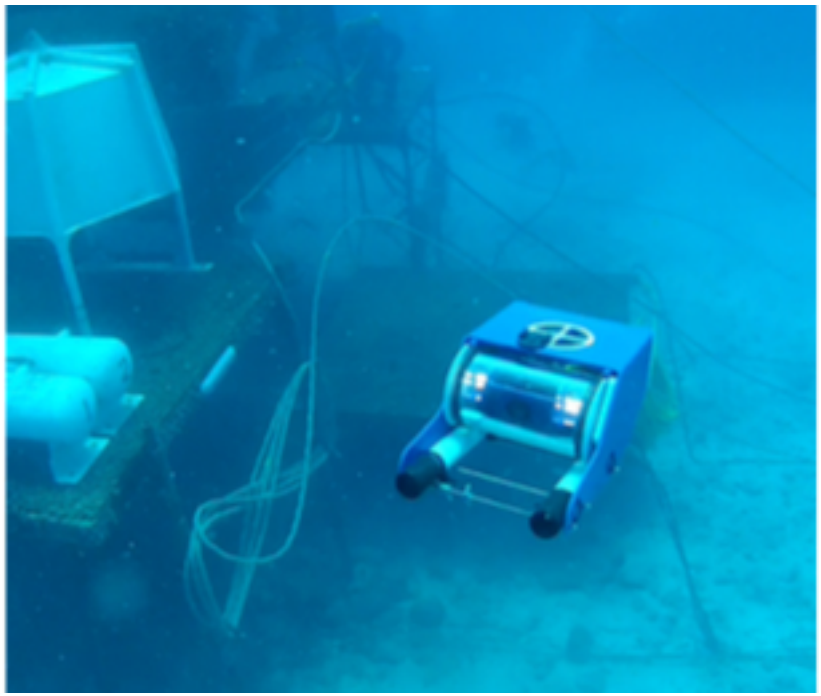
Tigal Mikrobus Cape and Click Boards

- "One Cape to Rule them All"
- Four Adaptable Capes in One
- Over 70 Click Boards Available and Counting



beagleboard.org

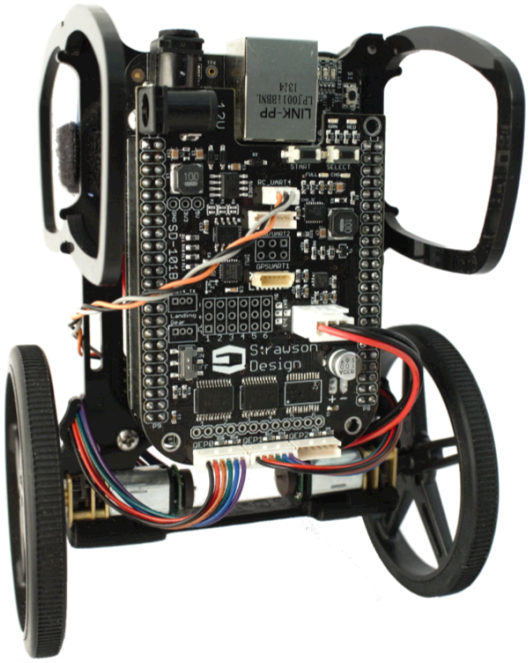
OpenROV



- Open-source underwater robot
- Community creating more accessible, affordable and awesome tools for underwater exploration
- Started by people wanting to explore an underwater cave
- Successfully Kickstarter'd

beagleboard.org

BeagleMIP



- Self-Balancing robot powered by the BeagleBone Black and the Novus Robotics Cape
- Hackable Open Source Robotics Platform for Fun and Education
- Developed at the University of California, San Diego to Teach Advanced Digital Control Systems

beagleboard.org

BeagleQuad

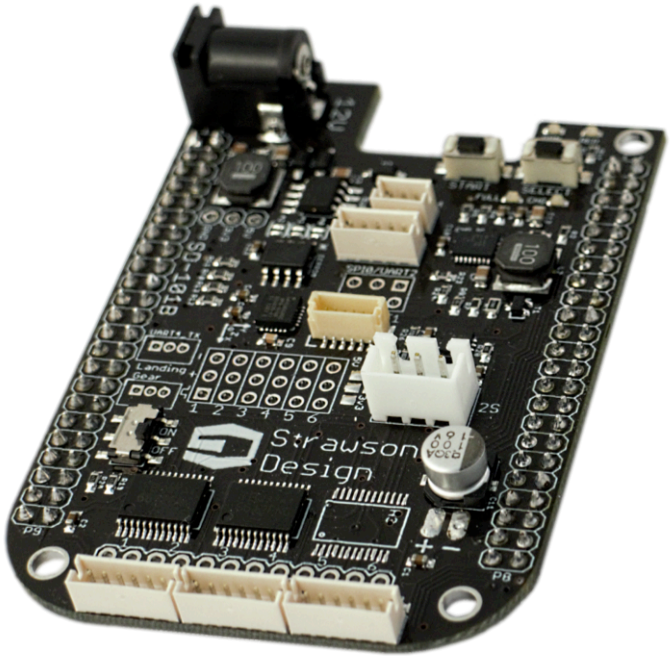
UCSD

The new ***Novus Robotics Cape*** sends your ***BeagleBone Black*** projects to the sky!

beagleboard.org

Novus Robotics Cape

STRAWSON DESIGN



- Bringing the power of the BeagleBone Black to your robotics project has never been easier.
- 2S LiPo Charger and Balancer
- 9-Axis IMU
- Drive 6 DC Motors
- Plug and Play Connections for
  - GPS
  - I2C
  - UART
  - Hobby Servos
  - Brushless ESCs
  - Spektrum RC Radio
- Open Source Libraries, Sample Code, and detailed documentation.

beagleboard.org

Georgia Tech course on mobile robots

- http://o-botics.org/
  - A place where roboticists can collaborate on robot designs, code, electronics, and hardware
- Build a robot from scratch using components from Sparkfun
- Learn about mobile robotics theory

beagleboard.org

Georgia Tech

Why is BeagleBone perfect for bots?

- Lots of I/Os (65 digital, 7 analog inputs, 8 PWMs...)
- PRUs (2 32-bit RISC microcontrollers
- Fast (1GHz) super-scalar armv7a processor
- Linux makes networking easy
- Ready to use out-of-the-box

beagleboard.org

Whats and whys on languages

- C/C++/Sketches
- Go
- Java
- Ruby
- Erlang
- Forth
- Lisp
- Perl
- Python
- JavaScript
- Scratch
- Blockly
- LabView / Simulink
- Etoys

Javascript and Python libraries

JavaScript

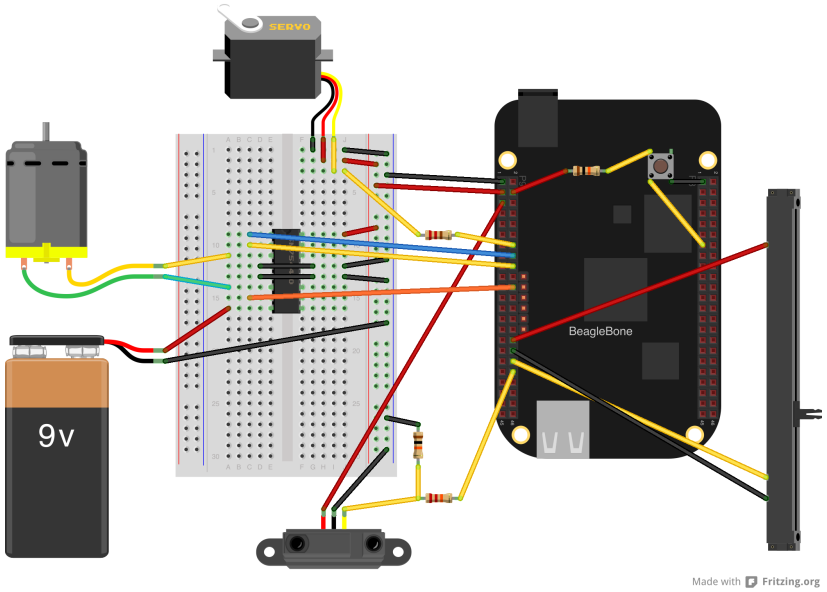
- BoneScript
- Johnny Five
- Cylon.js
- Node-RED
- BotSpeak

Python

- Adafruit_BBIO
- PyBBIO
- OpenCV

Some basic robotic components

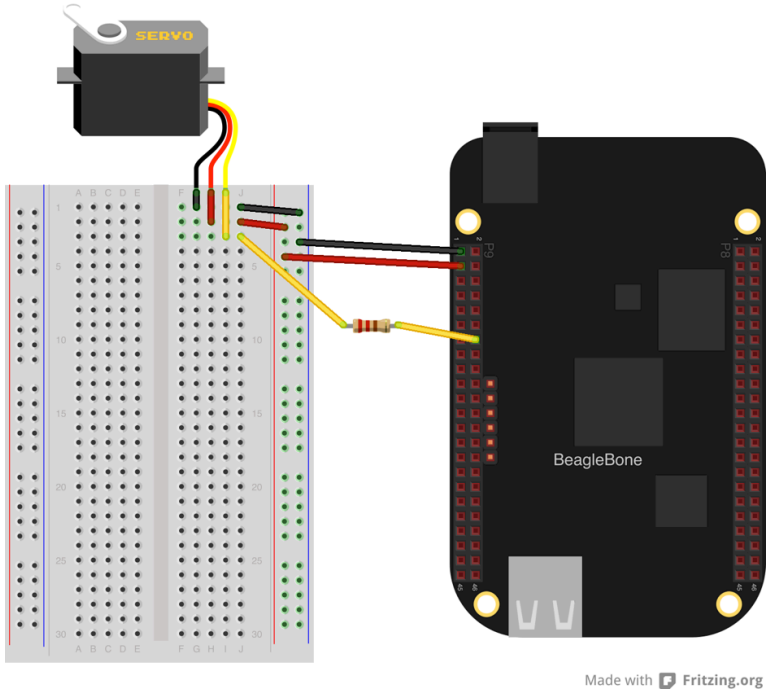
- Analog sensors
  - IR range finder
  - Potentiometer
- Digital sensor
  - Button
- Servo and DC motors



beagleboard.org

Wiring up a servo motor

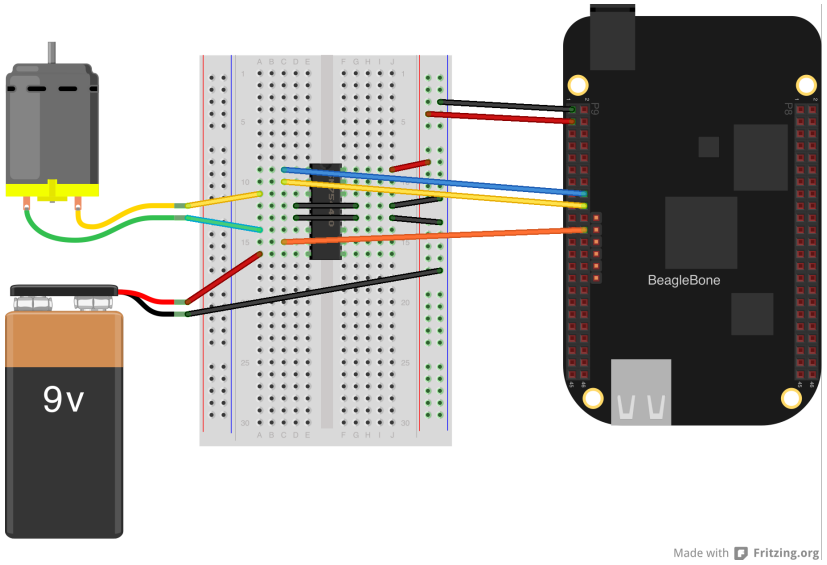
- Needs power and ground
- Connect to a PWM capable pin
- Resistor to protect the output pin



beagleboard.org

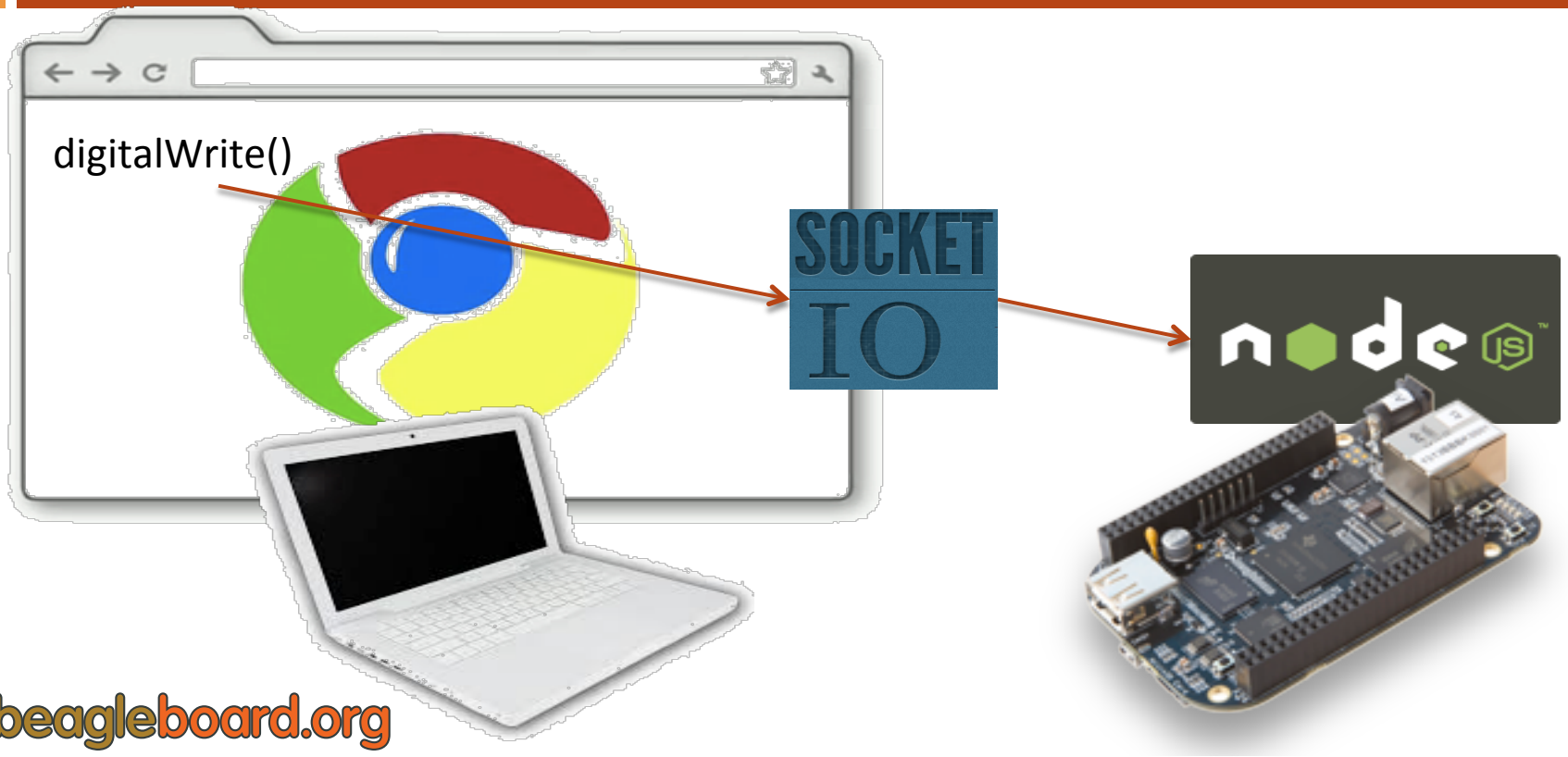
Wiring up a DC motor

- Use a transistor or H-bridge to provide enough power
- Use a PWM to be able to adjust
- Extra GPIOs to set direction

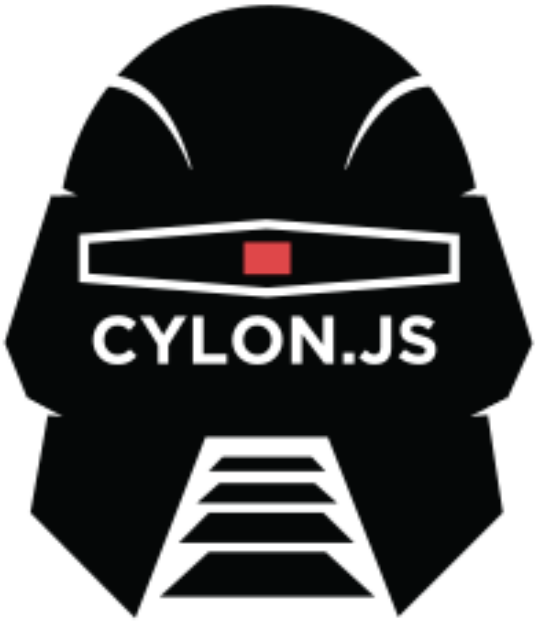


beagleboard.org

BoneScript in the browser

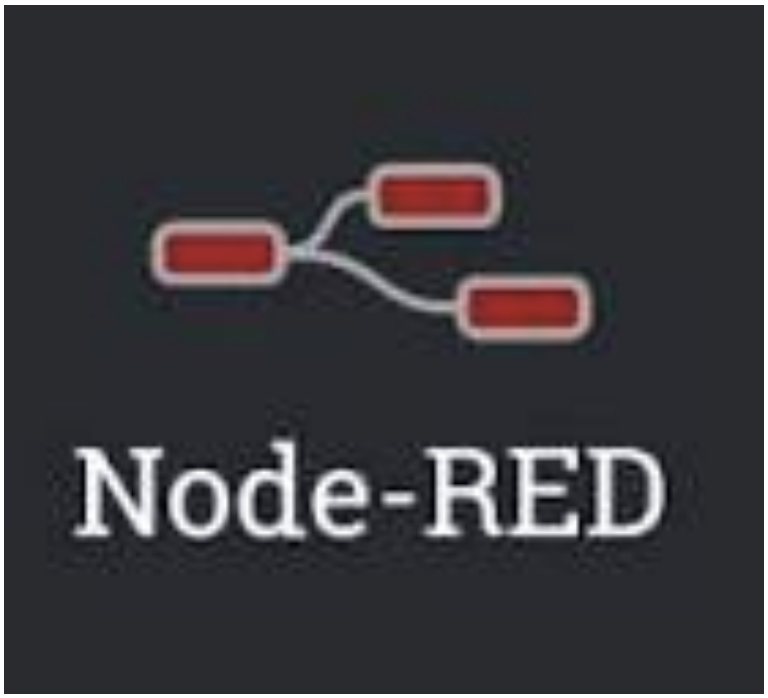


Cylon.js



beagleboard.org

Node-RED

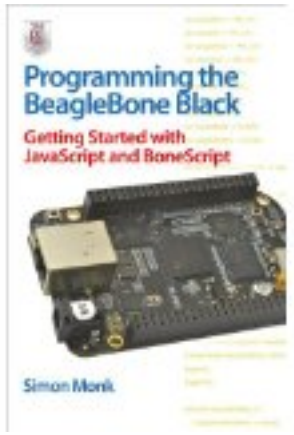


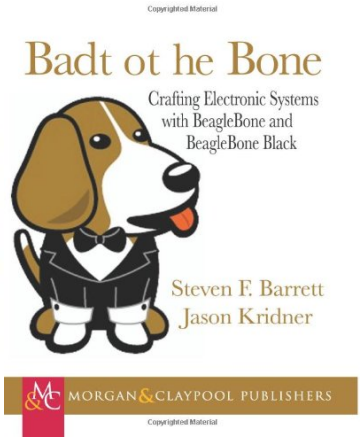
beagleboard.org

Upcoming libraries

- BotSpeak for PRU
- BeaglePilot

Some BeagleBone books





beagleboard.org