

# Bad to the Bone

Crafting Electronic Systems  
with BeagleBone and  
BeagleBone Black



Steven F. Barrett  
Jason Kridner



MORGAN & CLAYPOOL PUBLISHERS



# **Bad to the Bone**

**Crafting Electronic Systems with BeagleBone and BeagleBone Black**



# Synthesis Lectures on Digital Circuits and Systems

## Editor

**Mitchell A. Thornton**, *Southern Methodist University*

The Synthesis Lectures on Digital Circuits and Systems series is comprised of 50- to 100-page books targeted for audience members with a wide-ranging background. The Lectures include topics that are of interest to students, professionals, and researchers in the area of design and analysis of digital circuits and systems. Each Lecture is self-contained and focuses on the background information required to understand the subject matter and practical case studies that illustrate applications. The format of a Lecture is structured such that each will be devoted to a specific topic in digital circuits and systems rather than a larger overview of several topics such as that found in a comprehensive handbook. The Lectures cover both well-established areas as well as newly developed or emerging material in digital circuits and systems design and analysis.

## [Bad to the Bone: Crafting Electronic Systems with BeagleBone and BeagleBone Black](#)

Steven Barrett and Jason Kridner

2013

## [Introduction to Noise-Resilient Computing](#)

S.N. Yanushkevich, S. Kasai, G. Tangim, A.H. Tran, T. Mohamed, and V.P. Smerko

2013

## [Atmel AVR Microcontroller Primer: Programming and Interfacing, Second Edition](#)

Steven F. Barrett and Daniel J. Pack

2012

## [Representation of Multiple-Valued Logic Functions](#)

Radomir S. Stankovic, Jaakko T. Astola, and Claudio Moraga

2012

## [Arduino Microcontroller: Processing for Everyone! Second Edition](#)

Steven F. Barrett

2012

## [Advanced Circuit Simulation Using Multisim Workbench](#)

David Báez-López, Félix E. Guerrero-Castro, and Ofelia Delfina Cervantes-Villagómez

2012

[Circuit Analysis with Multisim](#)

David Báez-López and Félix E. Guerrero-Castro

2011

[Microcontroller Programming and Interfacing Texas Instruments MSP430, Part I](#)

Steven F. Barrett and Daniel J. Pack

2011

[Microcontroller Programming and Interfacing Texas Instruments MSP430, Part II](#)

Steven F. Barrett and Daniel J. Pack

2011

[Pragmatic Electrical Engineering: Systems and Instruments](#)

William Eccles

2011

[Pragmatic Electrical Engineering: Fundamentals](#)

William Eccles

2011

[Introduction to Embedded Systems: Using ANSI C and the Arduino Development Environment](#)

David J. Russell

2010

[Arduino Microcontroller: Processing for Everyone! Part II](#)

Steven F. Barrett

2010

[Arduino Microcontroller Processing for Everyone! Part I](#)

Steven F. Barrett

2010

[Digital System Verification: A Combined Formal Methods and Simulation Framework](#)

Lun Li and Mitchell A. Thornton

2010

[Progress in Applications of Boolean Functions](#)

Tsutomu Sasao and Jon T. Butler

2009

[Embedded Systems Design with the Atmel AVR Microcontroller: Part II](#)

Steven F. Barrett

2009

[Embedded Systems Design with the Atmel AVR Microcontroller: Part I](#)

Steven F. Barrett

2009

[Embedded Systems Interfacing for Engineers using the Freescale HCS08 Microcontroller II: Digital and Analog Hardware Interfacing](#)

Douglas H. Summerville

2009

[Designing Asynchronous Circuits using NULL Convention Logic \(NCL\)](#)

Scott C. Smith and JiaDi

2009

[Embedded Systems Interfacing for Engineers using the Freescale HCS08 Microcontroller I: Assembly Language Programming](#)

Douglas H. Summerville

2009

[Developing Embedded Software using DaVinci & OMAP Technology](#)

B.I. (Raj) Pawate

2009

[Mismatch and Noise in Modern IC Processes](#)

Andrew Marshall

2009

[Asynchronous Sequential Machine Design and Analysis: A Comprehensive Development of the Design and Analysis of Clock-Independent State Machines and Systems](#)

Richard F. Tinder

2009

[An Introduction to Logic Circuit Testing](#)

Parag K. Lala

2008

[Pragmatic Power](#)

William J. Eccles

2008

[Multiple Valued Logic: Concepts and Representations](#)

D. Michael Miller and Mitchell A. Thornton

2007

Finite State Machine Datapath Design, Optimization, and Implementation

Justin Davis and Robert Reese

2007

Atmel AVR Microcontroller Primer: Programming and Interfacing

Steven F. Barrett and Daniel J. Pack

2007

Pragmatic Logic

William J. Eccles

2007

PSpice for Filters and Transmission Lines

Paul Tobin

2007

PSpice for Digital Signal Processing

Paul Tobin

2007

PSpice for Analog Communications Engineering

Paul Tobin

2007

PSpice for Digital Communications Engineering

Paul Tobin

2007

PSpice for Circuit Theory and Electronic Devices

Paul Tobin

2007

Pragmatic Circuits: DC and Time Domain

William J. Eccles

2006

Pragmatic Circuits: Frequency Domain

William J. Eccles

2006

Pragmatic Circuits: Signals and Filters

William J. Eccles

2006



### High-Speed Digital System Design

Justin Davis

2006

### Introduction to Logic Synthesis using Verilog HDL

Robert B. Reese and Mitchell A. Thornton

2006

### Microcontrollers Fundamentals for Engineers and Scientists

Steven F. Barrett and Daniel J. Pack

2006

Copyright © 2013 by Morgan & Claypool

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopy, recording, or any other except for brief quotations in printed reviews, without the prior permission of the publisher.

Bad to the Bone: Crafting Electronic Systems with BeagleBone and BeagleBone Black

Steven Barrett and Jason Kridner

[www.morganclaypool.com](http://www.morganclaypool.com)

ISBN: 9781627051378      paperback

ISBN: 9781627051385      ebook

DOI 10.2200/S00500ED1V01Y201304DCS041

A Publication in the Morgan & Claypool Publishers series

*SYNTHESIS LECTURES ON DIGITAL CIRCUITS AND SYSTEMS*

Lecture #41

Series Editor: Mitchell A. Thornton, *Southern Methodist University*

Series ISSN

Synthesis Lectures on Digital Circuits and Systems

Print 1932-3166    Electronic 1932-3174

# Bad to the Bone

**Crafting Electronic Systems with BeagleBone and BeagleBone Black**

Steven Barrett  
University of Wyoming

Jason Kridner  
Texas Instruments

*SYNTHESIS LECTURES ON DIGITAL CIRCUITS AND SYSTEMS #41*



MORGAN & CLAYPOOL PUBLISHERS

## ABSTRACT

BeagleBone is a low cost, open hardware, expandable computer first introduced in November 2011 by BeagleBoard.org, a community of developers sponsored by Texas Instruments. Various BeagleBone variants, including the original BeagleBone and the new BeagleBone Black, host a powerful 32-bit, super-scalar ARM Cortex A8 processor operating from 720 MHz to 1 GHz. Yet, BeagleBone is small enough to fit in a small mint tin box. The “Bone” may be used in a wide variety of projects from middle school science fair projects to senior design projects to first prototypes of very complex systems. Novice users may access the power of the Bone through the user-friendly Bonescript environment, a browser-based experience, in MS Windows, the Mac OS X, or the Linux operating systems. Seasoned users may take full advantage of the Bone’s power using the underlying Linux-based operating system, a host of feature extension boards (Capes) and a wide variety of Linux community open source libraries. This book provides an introduction to this powerful computer and has been designed for a wide variety of users including the first time novice through the seasoned embedded system design professional. The book contains background theory on system operation coupled with many well-documented, illustrative examples. Examples for novice users are centered on motivational, fun robot projects while advanced projects follow the theme of assistive technology and image processing applications.

## KEYWORDS

BeagleBone, Linux, Ångström distribution, microcontroller interfacing, embedded systems design, Bonescript, ARM, open source computing

*For the students!*



# Contents

	<b>Preface</b> .....	<b>xxi</b>
	<b>Acknowledgments</b> .....	<b>xxv</b>
<b>1</b>	<b>Getting Started</b> .....	<b>1</b>
1.1	Welcome! .....	1
1.2	Overview .....	1
1.3	A brief Beagle history .....	3
1.4	BeagleBoard.org community .....	4
1.5	BeagleBone hardware .....	4
1.5.1	Open source hardware .....	6
1.6	Developing with Bonescript .....	6
1.7	BeagleBone Capes .....	6
1.8	Power requirements and capabilities .....	7
1.9	Getting started — success out of the box .....	8
1.9.1	Exercise 1: Programming with Bonescript through your browser .....	8
1.9.2	Exercise 2: Blinking an LED with Bonescript .....	9
1.9.3	Executing the blinked.js program .....	11
1.9.4	Exercise 3: Developing your own Boneyard — AROO! .....	11
1.10	Summary .....	12
1.11	References .....	12
1.12	Chapter Exercises .....	13
<b>2</b>	<b>System Design: Programming</b> .....	<b>15</b>
2.1	An Overview of the Design Process .....	15
2.2	Overview .....	16
2.3	Anatomy of a Program .....	16
2.3.1	Comments .....	18
2.3.2	Include files .....	20
2.3.3	Functions .....	21
2.3.4	Interrupt handler definitions .....	24

2.3.5	Program constants	24
2.3.6	Variables	24
2.3.7	Main function	25
2.4	Fundamental programming concepts	26
2.4.1	Operators	26
2.4.2	Programming constructs	30
2.4.3	Decision processing	33
2.5	Programming in JavaScript using Node.js	38
2.5.1	JavaScript	38
2.5.2	Event-driven programming	39
2.5.3	Node.js	39
2.6	Bonescript Development Environment	45
2.7	Application 1: Robot IR sensor	48
2.8	Application 2: Art piece illumination system	53
2.9	Application 3: Blinky 602A Autonomous Maze Navigating Robot	56
2.9.1	Blinky 602A robot	56
2.9.2	Requirements	58
2.9.3	Circuit diagram	58
2.9.4	Structure chart	58
2.9.5	UML activity diagrams	58
2.9.6	Bonescript code	63
2.10	Summary	64
2.11	References	65
2.12	Chapter Exercises	65
<b>3</b>	<b>BeagleBone Operating Parameters and Interfacing</b>	<b>67</b>
3.1	Overview	67
3.2	Operating Parameters	68
3.2.1	BeagleBone 3.3 VDC operation	68
3.2.2	Compatible 3.3 VDC logic families	69
3.2.3	Input/output operation at 5.0 VDC	69
3.2.4	Interfacing 3.3 VDC logic families to 5.0 VDC logic families	70
3.3	Input Devices	71
3.3.1	Switches	71
3.3.2	Switch Debouncing	73
3.3.3	Keypads	74



3.3.4	Sensors .....	74
3.3.5	Transducer Interface Design (TID) Circuit .....	81
3.3.6	Operational Amplifiers .....	82
3.4	Output Devices .....	84
3.4.1	Light Emitting Diodes (LEDs) .....	84
3.4.2	Seven Segment LED Displays .....	88
3.4.3	Tri-state LED Indicator .....	89
3.4.4	Dot Matrix Display .....	89
3.4.5	Liquid Crystal Display (LCD) .....	93
3.5	High Power Interfaces .....	93
3.5.1	High Power DC Devices .....	93
3.5.2	DC Motor Speed and Direction Control .....	94
3.5.3	DC Motor Operating Parameters .....	96
3.5.4	H-bridge direction control .....	96
3.5.5	DC Solenoid Control .....	98
3.5.6	Stepper motor control .....	98
3.6	Interfacing to Miscellaneous Devices .....	101
3.6.1	Sonalerts, Beepers, Buzzers .....	101
3.6.2	Vibrating Motor .....	101
3.6.3	DC Fan .....	101
3.7	AC Devices .....	102
3.8	Application: Equipping the Blinky 602A robot with a LCD .....	102
3.9	Application: the Blinky 602A interface on a custom cape .....	120
3.10	Summary .....	120
3.11	References .....	125
3.12	Chapter Exercises .....	126
<b>4</b>	<b>BeagleBone Systems Design .....</b>	<b>127</b>
4.1	Overview .....	127
4.2	What is an embedded system? .....	128
4.3	Embedded system design process .....	128
4.3.1	Project Description .....	128
4.3.2	Background Research .....	128
4.3.3	Pre-Design .....	130
4.3.4	Design .....	130
4.3.5	Implement Prototype .....	132

4.3.6	Preliminary Testing .....	133
4.3.7	Complete and Accurate Documentation .....	133
4.4	Submersible Robot .....	134
4.4.1	Requirements .....	134
4.4.2	Structure chart .....	134
4.4.3	Circuit Diagram .....	135
4.4.4	UML Activity Diagram .....	136
4.4.5	BeagleBone Code .....	137
4.4.6	Project Extensions .....	143
4.5	Mountain Maze Navigating Robot .....	143
4.5.1	Description .....	144
4.5.2	Requirements .....	144
4.5.3	Circuit diagram .....	144
4.5.4	Structure chart .....	144
4.5.5	UML activity diagrams .....	144
4.5.6	Bonescript code .....	144
4.5.7	Mountain Maze .....	150
4.5.8	Project extensions .....	152
4.6	Summary .....	153
4.7	References .....	153
4.8	Chapter Exercises .....	153
<b>5</b>	<b>BeagleBone features and subsystems .....</b>	<b>157</b>
5.1	Overview .....	157
5.2	Programming BeagleBone in Linux, C and C++ .....	157
5.2.1	Beagling in Linux .....	158
5.2.2	BeagleBone Linux releases .....	159
5.2.3	Bonescript processing in Linux .....	159
5.3	Updating your SD card or eMMC in Linux .....	163
5.3.1	Programming in C using the Ångström Toolchain .....	165
5.4	BeagleBone features and subsystems .....	166
5.5	Exposed functions .....	168
5.5.1	Expansion interface – original BeagleBone .....	168
5.5.2	Accessing pins via Linux 3.2 .....	174
5.6	Expansion Interface BeagleBone Black .....	186
5.6.1	Accessing pins with Device Tree Overlays –Linux 3.8 .....	186
5.6.2	Overview .....	186

5.6.3	Binary Tree .....	188
5.6.4	Device Tree Format .....	188
5.6.5	BeagleBone Device Tree–Linux 3.8 .....	189
5.7	Fundamental Examples Programming in C with BeagleBone Black–Linux 3.8 ..	193
5.8	Analog-to-Digital Converters (ADC) .....	199
5.8.1	ADC process: sampling, quantization and encoding .....	200
5.8.2	Resolution and Data Rate .....	201
5.8.3	ADC Conversion Technologies .....	202
5.8.4	BeagleBone ADC subsystem description –Linux 3.2 .....	203
5.8.5	ADC conversion via Linux 3.2 .....	204
5.8.6	ADC support functions in C Linux 3.2 .....	204
5.8.7	ADC support functions in C Linux 3.8 .....	209
5.9	Serial Communications .....	212
5.9.1	Serial Communication Terminology .....	212
5.9.2	Serial UART .....	215
5.9.3	Serial Peripheral Interface (SPI) .....	222
5.10	Precision Timing .....	224
5.10.1	Timing related terminology .....	225
5.10.2	BeagleBone timing capability system–Linux 3.2 .....	225
5.11	Pulse Width Modulation (PWM) .....	231
5.11.1	BeagleBone PWM subsystem (PWMSS) description .....	232
5.11.2	PWM configuration–Linux 3.2 .....	233
5.11.3	PWM C support functions–Linux 3.2 .....	233
5.11.4	PWM C support functions–Linux 3.8 .....	235
5.12	Networking .....	237
5.12.1	Inter-Integrated Circuit (I2C) bus .....	237
5.12.2	Controller Area Network (CAN) bus .....	238
5.12.3	Ethernet .....	239
5.13	Liquid Crystal Display (LCD) interface .....	239
5.13.1	C support functions .....	241
5.14	Interrupts .....	241
5.14.1	Bonescript interrupt support .....	241
5.15	Summary .....	244
5.16	References .....	244
5.17	Chapter Exercises .....	245

<b>6</b>	<b>BeagleBone “Off the Leash”</b>	<b>247</b>
6.1	Overview	247
6.2	Boneyard II: a portable Linux platform—BeagleBone unleashed	248
6.3	Application 1: Weather station in Bonescript	251
6.3.1	Requirements	251
6.3.2	Structure chart	251
6.3.3	Circuit diagram	253
6.3.4	UML activity diagrams	253
6.3.5	Bonescript code	253
6.4	Application 2: Speak-and-Spell in C	264
6.4.1	BeagleBone C Code	264
6.5	Application 3: Daguer 5 Treaded Robot	277
6.5.1	Description	277
6.5.2	Requirements	277
6.5.3	Circuit diagram	277
6.5.4	Structure chart	279
6.5.5	UML activity diagrams	281
6.5.6	BeagleBone C code	281
6.6	Application 4: Portable Image Processing Engine	292
6.6.1	Brief introduction to image processing	292
6.6.2	OpenCV Computer Vision Library	293
6.6.3	Stache Cam	293
6.7	Summary	303
6.8	References	303
6.9	Chapter Exercises	304
<b>7</b>	<b>Where to from here?</b>	<b>305</b>
7.1	Overview	305
7.2	Software Libraries	305
7.2.1	OpenCV	305
7.2.2	Qt	305
7.2.3	Kinect	306
7.3	Additional resources	306
7.3.1	OpenROV	306
7.3.2	Ninja Blocks	306
7.3.3	BeagleBoard.org Resources	307

7.3.4	Contributing to Bonescript .....	308
7.4	Summary .....	308
7.5	References .....	309
7.6	Chapter Exercises .....	309
<b>A</b>	<b>Bonescript functions .....</b>	<b>311</b>
<b>B</b>	<b>LCD interface for BeagleBone in C .....</b>	<b>315</b>
B.1	BeagleBone Original – Linux 3.2 .....	315
B.2	BeagleBone Black –Linux 3.8 .....	335
<b>C</b>	<b>Parts List for Projects .....</b>	<b>351</b>
<b>D</b>	<b>BeagleBone Device Tree .....</b>	<b>355</b>
D.1	am33xx.dtsi .....	355
D.2	am335x–bone–common.dtsi .....	372
D.3	am335x–bone.dts .....	384
D.4	am335x–boneblack.dts .....	384
D.5	am33xx_pwm–00A0.dts .....	386
D.6	bone_pwm_P8_13–00A0.dts .....	388
D.7	cape–bone–iio–00A0.dts .....	390
	<b>Authors’ Biographies .....</b>	<b>393</b>
	<b>Index .....</b>	<b>395</b>



# Preface

BeagleBone is a low cost, open hardware expandable computer first introduced in November 2011 by BeagleBoard.org, a community of developers sponsored by Texas Instruments. Various BeagleBone variants host a powerful 32-bit, super-scalar ARM Cortex A8 processor operating from 720 MHz to 1 GHz. Yet, BeagleBone is small enough to fit in a small mint tin box. The “Bone” may be used in a wide variety of projects from middle school science fair projects to senior design projects to first prototypes of very complex systems. Novice users may access the power of the Bone through the user-friendly browser-based Bonescript environment that may be operated in MS Windows, the Mac OS X, or the Linux operating systems. Seasoned users may take full advantage of the Bone’s power using the underlying Linux-based operating system, a host of feature extension boards (Capes) and a wide variety of Linux community open source libraries. This book provides an introduction to this powerful computer and has been designed for a wide variety of users including the first time novice through the seasoned embedded system design professional. The book contains background theory on system operation coupled with many well-documented, illustrative examples. Examples for novice users are centered on motivational, fun robot projects while advanced projects follow the theme of assistive technology and image processing applications.

Texas Instruments has a long history of educating the next generation of STEM (Science, Technology, Engineering and Mathematics) professionals. BeagleBone is the latest educational innovation in a long legacy which includes the Speak & Spell and Texas Instruments handheld calculators. The goal of the BeagleBone project is to place a powerful, expandable computer in the hands of young innovators. Novice users can unleash the power of the Bone through the user-friendly, browser-based Bonescript environment. As the knowledge and skill of the user develops and matures, BeagleBone provides more sophisticated interfaces including a complement of C-based functions to access the hardware systems aboard the ARM Cortex A8 processor. These features will prove useful for college-level design projects including capstone design projects. The full power of the processor may be unleashed using the underlying onboard Linux-based operating system. A wide variety of hardware extension features are also available using daughter card “Capes” and Linux community open source software libraries. These features allow for the rapid prototyping of complex, expandable embedded systems.

## A BRIEF BEAGLE HISTORY

The Beagle family of computer products include BeagleBoard, BeagleBoard xM, the original BeagleBone and the new BeagleBone Black. All have been designed by Gerald Coley, a long time expert Hardware Applications Engineer, of Texas Instruments. He has been primarily responsible for all hardware design, manufacturing planning, and scheduling for the product line. The goal of the en-

tire product line is to provide users a powerful computer at low cost with no restrictions. As Gerald describes it, his role has been to “get the computers out there and then get out of the way” to allow users to do whatever they want. The computers are intended for everyone and the goal is to provide “out of the box success.” He indicated it is also important to provide users a full suite of open source hardware details and to continually improve the product line by listening to customers’ desires and keeping the line fresh with new innovations. In addition to the computers, a full line of Capes to extend processor features is available. There are more than 20 different Capes currently available with 50 more on the way.

BeagleBone was first conceptualized in early Fall 2011. The goal was to provide a full-featured, expandable computer employing the Sitara ARM processor in a small profile case. Jason Kridner told Gerald the goal was to design a processor board that would fit in a small mint tin box. After multiple board revisions, Gerald was able to meet the desired form factor. BeagleBone enjoyed a fast development cycle. From a blank sheet of paper as a starting point, to full production was accomplished, in 90 days.

New designs at Texas Instruments are given a code name during product development. During the early development of the Beagle line, the movie “Underdog” was showing in local theaters. Also, Gerald had a strong affinity to Beagles based on his long time friend and pet “Jake.” Gerald dubbed the new project “Beagle” with the intent of changing the name later. However, the name stuck and the rest is history. Recently, an acronym was developed to highlight the Beagle features:

- **B**ring your own peripherals
- **E**ntry level costs
- **A**RM Cortex-A8 superscalar processor
- **G**raphics accelerated
- **L**inux and open source community
- **E**nvironment for innovators

## APPROACH OF THE BOOK

Concepts will be introduced throughout the book with the underlying theory of operation, related concepts and many illustrative examples. Concepts early in the book will be illustrated using motivational robot examples including an autonomous robot navigating about a two-dimensional maze based on the Graymark Blinky 602A robot platform, an autonomous four wheel drive (4WD) robot (DFROBOT ROBOT0003) in a three-dimensional mountain maze, and a SeaPerch Remotely Operated Vehicle (ROV).

Advanced examples follow a common theme of assistive technology. Assistive technology provides those with challenges the ability to meet their potential using assistive devices. One example covered in the book is a BeagleBone based Speak & Spell. Speak and Spell is an educational toy



developed by Texas Instruments in the mid-1970's. It was developed by the engineering team of Gene Franz, Richard Wiggins, Paul Breedlove and Larry Brannntingham.

This book is organized as follows. Chapter 1 provides an introduction to the BeagleBone processor and the Bonescript programming environment. Chapter 2 provides a brief introduction to programming and provides a detailed introduction to the Bonescript environment. The chapter concludes with an extended example of the Blink 602A robot project. Chapter 3 provides electrical interfacing fundamentals to properly connect BeagleBone to peripheral components. Chapter 4 provides an introduction to system level design, the tools used to develop a system and two advanced robot system examples.

Chapter 5 begins an advanced treatment of BeagleBone and the ARM Cortex A8 processor and an in depth review of exposed functions and how to access them using both Bonescript and C functions. In Chapter 6 the full power of BeagleBone is unleashed via several system level examples. Chapter 7 briefly reviews additional resources available to the BeagleBone user.

## A LITTLE BEAGLE LORE

The beagle dog has been around for centuries and was quite popular in Europe. They were known for their sense of smell and were used to track hare. They are also known for their musical, bugling voice. In “The Beagle Handbook,” Dan Rice, Doctor of Veterinary Medicine (DVM), indicates the Beagle was probably named for their size or voice. A combination of the French words for “open wide” and “throat” results in Begueule and might refer to the Beagle’s distinctive and jubilant bugle while on the hunt. Rice also notes “the word beagling isn’t found in Webster’s dictionary, but it’s well recognized in canine fancy. Used by the Beagle community for practically all endeavors that involve their beautiful dogs [Rice, 2000].” We too will use the term “beagling” to describe the fun sessions of working with BeagleBone that are ahead. It is also worth mentioning that Ian Dunbar, Member of the Royal College of Veterinary Surgeons (MRCVS) in “The Essential Beagle” indicates the “overall temperament of the Beagle is bold and friendly.” This seems like an appropriate description for the BeagleBone computer running Bonescript software.

## THE BEAGLEBOARD.ORG COMMUNITY

The BeagleBoard.org community has many members. What we all have in common is the desire to put processing power in the hands of the next generation of users. BeagleBoard.org, with Texas Instruments’ support, embraced the open source concept with the development and release of BeagleBone in late 2011. Their support will insure the BeagleBone project will be sustainable. BeagleBoard.org partnered with circuitco ([www.circuitco.com](http://www.circuitco.com)) to produce BeagleBone and its associated Capes. The majority of the Capes have been designed and fabricated by circuitco. Clint Cooley, President of circuitco, is most interested in helping users develop and produce their own ideas for BeagleBone Capes. Texas Instruments has also supported the BeagleBoard.org community by giving Jason Kridner the latitude to serve as the open platform technologist and evangelist for the BeagleBoard.org community. The most important members of the community are the BeagleBoard

and Bone users. Our ultimate goal is for the entire community to openly share their successes and to encourage the next generation of STEM practitioners.

### A LITTLE BIT ABOUT THE AUTHORS

Steve is a life time teacher. He has taught at a variety of age levels from middle school science enhancement programs through graduate level coursework. He served in the United States Air Force for 20 years and spent approximately half of that time as a faculty member at the United States Air Force Academy. Following military “retirement,” he began a second academic career at the University of Wyoming as an Assistant Professor. He now serves as a Professor of Electrical and Computer Engineering and the Associate Dean for Academic Programs. He is planning on teaching into his 80’s and considers himself a student first teacher. Most importantly, he has two “grand beagles,” Rory and Romper, fondly referred to as “the girls.”

Jason got an early start with computing at age 9 programming his mom’s Tandy Radio Shack TRS-80. He was also a big fan of Forrest Mim’s “Getting Started in Electronics.” Much of his allowance was spent developing projects. He really enjoyed the adventure of trying new hardware and software projects. His goal is to bring back this spirit of adventure and discovery to the Beagle-Board.org community. While still in high school, he worked extensively with AutoCAD at a leak and flow testing company. He joined Texas Instruments in 1992 after a co-op with them while a student at Texas A&M University. He started using Linux at about the same time. Since joining TI he has worked on a wide variety of projects including audio digital signal processing, modems, home theater sound, multi-dimensional audio and MP3 player development.

Steven Barrett and Jason Kridner  
April 2013

# Acknowledgments

The authors would like to thank Cathy Wicks and Larissa Swanland of Texas Instruments who proposed this collaboration. We also thank Gerald Coley of Texas Instruments who was interviewed for the book. We also thank Clint Cooley, President of circuitco for hosting a tour of his company where BeagleBone and its associated Capes are produced. We would also like to thank Joel Claypool of Morgan & Claypool Publishers for his support of this project and his permission to use selected portions from previous M&C projects. We would also like to thank our reviewers including Dimple Joseph a student in Houston, Texas; Richard (Andy) Darter and Luke Kaufman of the University of Wyoming; Dr. Derek Molloy of Dublin City University; Shawn Trail of the University of Victoria, Victoria, BC, Canada; and C. P. Ravikumar, Technical Director of University Relations at TI India. Also, a special thank you to Jonathan Barrett of Closer to the Sun International

Steven Barrett and Jason Kridner  
April 2013



## CHAPTER 1

# Getting Started

**Objectives:** After reading this chapter, the reader should be able to do the following:

- Provide a brief history of the Beagle computer line.
- Outline the different members of the BeagleBoard.org community.
- Appreciate the role of the BeagleBoard.org community.
- Describe BeagleBone concept of open source hardware.
- Diagram the layout of the original BeagleBone and BeagleBone Black computers.
- Summarize the differences between the original BeagleBone and BeagleBone Black computers.
- Describe BeagleBone Cape concept and available Capes.
- Define the power requirements for BeagleBone computer.
- Download, configure, and successfully execute a test program using the Cloud9 Integrated Development Environment (IDE) and the Bonescript software.
- Design and implement a BeagleBone Boneyard prototype area to conduct laboratory exercises.

## 1.1 WELCOME!

Welcome to the wonderful world of BeagleBone! Whether you are a first-time BeagleBone user or are seasoned at “Beagling,” this book should prove useful. Chapter 1 is an introduction to BeagleBone, its environment and the Beagle community. BeagleBone hosts the Linux operating system; however, the user-friendly Bonescript programming environment may be used in a wide variety of browser environments including: Microsoft Windows, MAC OS X, and Linux. We provide instructions on how to rapidly get up-and-operating right out of the box! Also, an overview of BeagleBone features and subsystems is provided. The chapter concludes with several examples to get you started.

## 1.2 OVERVIEW

BeagleBone is a low cost, open hardware, expandable computer first introduced in November 2011 by BeagleBoard.org, a community of developers started by Beagle enthusiasts at Texas Instruments. Various BeagleBone variants host a powerful 32-bit, super-scalar ARM<sup>®</sup> Cortex<sup>™</sup> –A8 processor

## 2 1. GETTING STARTED



**Figure 1.1:** BeagleBone. (Figures adapted and used with permission of [www.adafruit.com](http://www.adafruit.com).)

operating up to 1 GHz. This allows the “Bone” to be used in a wide variety of applications usually reserved for powerful, desktop systems. The Bone is a full featured computer small enough to fit in a small mint tin box as shown in Figure 1.1. The combination of computing power and small form factor allows the Bone to be used in a wide variety of projects from middle school science fair projects to senior design projects to first prototypes of very complex systems.

Novice users may access the power of the Bone through the user-friendly, browser-based Bonescript environment in MS Windows, Mac OS X, and Linux. Seasoned users may take full advantage of the Bone’s power using the underlying Linux-based operating system, a host of feature extension boards (Capes) and a wide variety of open source libraries.

Texas Instruments has supported BeagleBone development and has a long history of educating the next generation of STEM (Science, Technology, Engineering and Mathematics) professionals. BeagleBone is the latest education innovation in a long legacy which includes the Speak & Spell and Texas Instruments handheld calculators. The goal of BeagleBone project is to place a powerful, expandable computer in the hands of young innovators through the user-friendly, browser-based Bonescript environment. As the knowledge and skill of the user develops and matures, BeagleBone provides more sophisticated interfaces including a complement of C-based functions to access the hardware systems aboard the ARM Cortex A8 processor. These features will prove useful for college-

–level design projects including capstone design projects. The full power of the processor may be unleashed using the underlying onboard Linux–based operating system.

To assemble a custom system, the Bone may be coupled with a wide variety of daughter board “Capes” and open source libraries. These features allow for the rapid prototyping of complex, expandable embedded systems. A full line of Capes to extend processor features is available. There are over 35 different Capes currently available with at least as many more on the way. The Cape system is discussed later in this chapter. Open source libraries are discussed later in the book.

### 1.3 A BRIEF BEAGLE HISTORY

The Beagle family of computer products include BeagleBoard, BeagleBoard–xM, the original BeagleBone and the new BeagleBone Black. All have been designed by Gerald Coley, a long time expert Hardware Applications Engineer, of Texas Instruments. He has been primarily responsible for all hardware design, manufacturing planning, and scheduling for the product lines. The goal of the entire platform line is to provide users a powerful computer at a low cost with no restrictions. As Gerald describes it, his role has been to “get the computers out there and then get out of the way” to allow users to do whatever they want. The computers are intended for everyone and the goal is to provide out–of–box success.” He indicated it is also important to provide users a full suite of open source hardware details and to continually improve the product line by listening to customers’ desires and keeping the line fresh with new innovations.

BeagleBone was first conceptualized in early Fall 2011. The goal was to provide a full–featured, expandable computer employing the Sitara<sup>TM</sup> AM335x ARM<sup>®</sup> processor in a small profile case. Jason Kridner, a longtime expert Software Applications Engineer responsible for all Beagle software and co–author of this textbook, of Texas Instruments told Gerald the goal was to design a processor board that would fit in a small mint tin box. After multiple board revisions, Gerald was able to meet the desired form factor. BeagleBone enjoyed a fast development cycle. From a blank sheet of paper starting point to full production was accomplished in 90 days.

New designs at Texas Instruments are given a code name during product development. During the early development of the Beagle line, the movie “Underdog” was showing in local theaters. Also, Gerald had a strong affinity to Beagles based on his long time friend and pet, “Jake.” Gerald dubbed the new project “Beagle” with the intent of changing the name later. However, the name stuck and the rest is history.

The “Underdog” movie was adapted from the cartoon series of the same name. The series debuted in 1964 and featured the hero “Underdog” –who was humble and loveable “Shoeshine Boy” until danger called. Shoeshine Boy was then transformed into the invincible flying Underdog complete with a cape that allowed him to fly. His famous catch phrase was “Have no fear, Underdog is here!” This series is readily available on Amazon and is a lot of good fun. BeagleBone is a bit like Underdog. It can fit into an unassuming, mint tin box, yet is a full–featured, “fire–breathing,” computer equipped with the Linux operating system whose features are extended with Capes.

## 4 1. GETTING STARTED

### 1.4 BEAGLEBOARD.ORG COMMUNITY

The BeagleBoard.org community has thousands of members. What we all have in common is the desire to put processing power in the hands of the next generation of users. BeagleBoard.org with Texas Instruments support embraced the open source concept with the development and release of BeagleBone in late 2011. Their support will ensure BeagleBone project remains sustained. BeagleBoard.org partnered with Circuitco (a contract manufacturer) to produce BeagleBone and many of its associated Capes. The majority of the Capes have been designed and fabricated by Circuitco. Clint Cooley, President of Circuitco, is most interested in helping users develop and produce their own ideas for BeagleBone Capes. Texas Instruments has also supported the BeagleBoard.org community by giving Jason Kridner the latitude to serve as the open platform technologist and evangelist for the BeagleBoard.org community. The most important members of the community are the BeagleBoard and BeagleBone users (you). Our ultimate goal is for the entire community to openly share their successes and to encourage the next generation of STEM practitioners.

In the next several sections, an overview of BeagleBone hardware, system configuration and software is discussed.

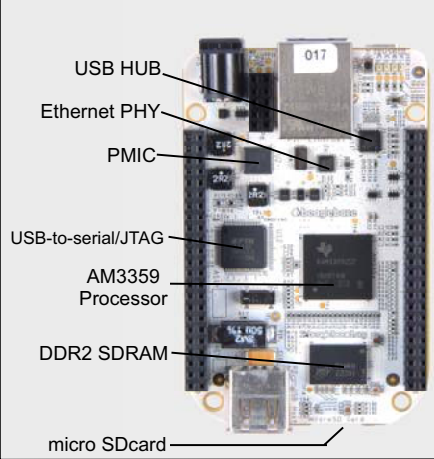
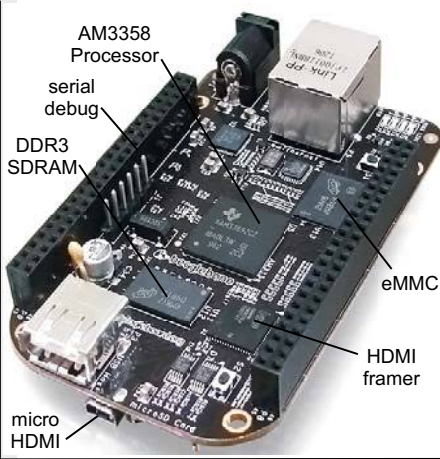


### 1.5 BEAGLEBONE HARDWARE

There are two different variants of BeagleBone: the original BeagleBone, released in late 2011, and BeagleBone Black released, in early 2013. The features of both Bones are summarized in Figure 1.2.

The two variants of BeagleBone are physically quite similar. They both fit inside a small mint box. The original BeagleBone is equipped with a Sitara<sup>TM</sup> AM3359 ARM<sup>®</sup> processor that operates at a maximum frequency of 720 MHz when powered from an external 5 VDC source. The original BeagleBone is also equipped with an SD/MMC micro SD card. The card acts as the “hard drive” for BeagleBone board and hosts the Ångström distribution operating system, Cloud9 Integrated Development Environment (IDE), and Bonescript. The original BeagleBone also features an integrated JTAG “and serial port” over the USB client connection [Coley].

BeagleBone Black hosts the Sitara<sup>TM</sup> AM3358 ARM<sup>®</sup> processor which operates at a maximum frequency of 1 GHz. The micro SD card provided with the original BeagleBone has been replaced with a 2 Giga byte (GB) eMMC (embedded MultiMedia Card) on BeagleBone Black. This provides for non-volatile mass storage in an integrated circuit package. The eMMC acts as the “hard drive” for BeagleBone Black board and hosts the Ångström distribution operating system, Cloud9 Integrated Development Environment (IDE) and Bonescript. The micro SD card connector is still available for expanding the storage. BeagleBone Black has also been enhanced with an HDMI (High-Definition Multimedia Interface) framer and micro connector. HDMI is a compact digital audio/interface which is commonly used in commercial entertainment products such as DVD players, video game consoles and mobile phones. Interestingly, BeagleBone Black costs approximately half of the original BeagleBone at approximately US \$45 [Coley].



	BeagleBone White	BeagleBone Black
		
Processor	AM3359 500 MHz - USB powered 720 MHz - DC powered	AM3358 1 GHz - USB powered 1 GHz - DC powered
Memory	256 MB DDR2 SDRAM SD/MMC micro SD connector	512 MB DDR3 SDRAM 2GB eMMC FLASH
Power Options	USB connection 5 VDC external jack	USB connection 5 VDC external jack
Power Mgt	TPS65217B Power Management IC (PMIC)	TPS65217B Power Management IC (PMIC)
Board Features	Single cable integrated JTAG, serial port and USB 10/100 Ethernet Power expansion header	HDMI with audio USB 10/100 Ethernet serial debug via external header
Processor Subsystems	 <ul style="list-style-type: none"> <li>176K ROM</li> <li>64K RAM</li> <li>3D graphics engine</li> <li>LCD and touchscreen controller</li> <li>PRU-ICSS</li> <li>Real Time Clock (RTC)</li> <li>USB ports (2)</li> <li>Ethernet</li> <li>Controller Area Network (CAN)</li> <li>UART (2)</li> <li>McASPs (2)</li> <li>McSPI (2)</li> <li>I2C (3)</li> <li>Analog-to-digital converter</li> <li>Enhanced Capture Module (3)</li> <li>Pulse width modulation (3)</li> <li>Crypto accelerator</li> </ul>	 <ul style="list-style-type: none"> <li>176K ROM</li> <li>64K RAM</li> <li>3D graphics engine</li> <li>LCD and touchscreen controller</li> <li>PRU-ICSS</li> <li>Real Time Clock (RTC)</li> <li>USB ports (2)</li> <li>Ethernet</li> <li>Controller Area Network (CAN)</li> <li>UART (2)</li> <li>McASPs (2)</li> <li>McSPI (2)</li> <li>I2C (3)</li> <li>Analog-to-digital converter</li> <li>Enhanced Capture Module (3)</li> <li>Pulse width modulation (3)</li> <li>Crypto accelerator</li> </ul>

[source: BeagleBone System Reference Manual]

**Figure 1.2:** BeagleBone comparison [Coley].(Figures adapted and used with permission of beagle-board.org.)

## 6 1. GETTING STARTED

### 1.5.1 OPEN SOURCE HARDWARE

To spur the development and sharing of new developments and features among the BeagleBoard community, all hardware details of BeagleBone boards are readily available as open source. This includes detailed schematics, bill-of-materials and printed circuit board (PCB) layout. This allows for custom designs to take full advantage of BeagleBone-based products at the lowest possible cost. Hardware details are readily available at [www.beagleboard.org](http://www.beagleboard.org).

### 1.6 DEVELOPING WITH BONESCRIPT

Bonescript helps to provide a user-friendly, browser-based programming environment for BeagleBone. Bonescript consists of a JavaScript library of functions to rapidly develop a variety of physical computing applications. Bonescript is accessed via a built-in web server that is pre-installed on the BeagleBone board. Because Bonescript programming is accessed over your browser, it may be developed from any of MS Windows, Mac OS X, and Linux. Exercise 1 provided later in the chapter provides easy-to-use instructions on how to quickly get Bonescript-based program up and operating. Although Bonescript is user-friendly, it may also be used to rapidly prototype complex embedded systems.

### 1.7 BEAGLEBONE CAPES

A wide variety of peripheral components may be easily connected to BeagleBone. A series of Capes have been developed to give BeagleBone “super powers” (actually, using these plug-in boards provide BeagleBone with additional functionality). Currently, there are over 20 different capes available with many more on the way. See [www.beaglebonecapes.com](http://www.beaglebonecapes.com) for the latest information on available capes. BeagleBone may be equipped with up to four Capes at once. Each Cape is equipped with an onboard EEPROM to store Cape configuration data. For example, Capes are available to provide BeagleBone:

- a seven inch, 800 x 400 pixel TFT LCD (liquid crystal display) (LCD7 Cape)
- a battery pack (Battery Cape)
- a Controller Area Network (CAN) interface (CANbus Cape)
- separate Capes for interface to a wide variety of devices including DVI-D, Profibus, RS-232, RS-485 and VGA standards
- an interface for two stepper motors

There is also a Cape equipped with a solderless breadboard area to provide a straight forward interface to external components. This Cape is illustrated in Figure 1.3 along with a representative sample of other Capes.



**Figure 1.3:** BeagleBone Capes. (top left) prototype Cape, (top right) Liquid Crystal Display LCD7 Cape, (bottom left) Controller Area Network Cape, (bottom right) motor control Cape [Circuitco].

Most of these Capes have been designed by BeagleBoard.org and are manufactured by Circuitco in Richardson, Texas. Both BeagleBoard.org and Circuitco are interested in manufacturing Cape designs submitted by the BeagleBoard.org community.

## 1.8 POWER REQUIREMENTS AND CAPABILITIES

BeagleBone may be powered from the USB cable, an external 5 VDC power supply or via the Battery Cape. When powered via the USB cable, the original BeagleBone operates at 500 MHz and BeagleBone Black at 1 GHz. When DC powered via an external source, the original BeagleBone operates up to 720 MHz and the Black up to 1 GHz. An external 5 VDC supply may be connected to the external power supply connector. The 5 VDC supply requires a minimum 1 amp current rating and must be equipped with a 2.1 mm center positive with a 5.5 mm outer barrel [Coley]. A BeagleBone-compatible 5 VDC, 2A power supply is available from Adafruit ([www.adafruit.com](http://www.adafruit.com)).



**Figure 1.4:** BeagleBone power sources (photo courtesy of J. Barrett, Closer to the Sun International).

## 1.9 GETTING STARTED — SUCCESS OUT OF THE BOX

It is important to the BeagleBoard designers that a novice user experience out-of-box success on first time use. In this section we provide a series of exercises to quickly get up and operating with BeagleBone and Bonescript.

### 1.9.1 EXERCISE 1: PROGRAMMING WITH BONESCRIPT THROUGH YOUR BROWSER

For the novice user, the quickest way to get up and operating with Bonescript is through your web browser. Detailed step-by-step quick-start instructions are provided in the online “BeagleBone Quick-Start Guide” available at [www.beagleboard.org](http://www.beagleboard.org). The guide consists of four easy to follow steps. The four steps are summarized below. They may be accomplished in less than 10 minutes to allow rapid out-of-box operation [[www.beagleboard.org](http://www.beagleboard.org)].

1. Plug BeagleBone into the host computer via the mini-USB capable and open the START.htm file, or README.htm for older software images.

2. Install the proper drivers for your system. MS Windows users need to first determine if the host computer is running 32 or 64-bit Windows. Instructions to do this are provided at <http://support.microsoft.com/kb/827218>.
3. For the original BeagleBone, EJECT the drive original (if you have not updated to the latest Bonescript software image).
4. Browse to “Information on BeagleBoard” information using Chrome or Firefox by going to <http://192.168.7.2>
5. Explore the Cloud9 IDE develop environment by navigating to <http://192.168.7.2:3000/> using Chrome or Firefox.

### 1.9.2 EXERCISE 2: BLINKING AN LED WITH BONESCRIPT

In this exercise we blink a light emitting diode (LED) onboard BeagleBone and also an external LED. Bonescript will be used along with the blinked.js program. The LED interface to the BeagleBone is shown in Figure 1.5. The interface circuit consists of a 270 ohm resistor in series with the LED. This limits the current from BeagleBone to a safe value below 8 mA. The LED has two terminals: the anode (+) and the cathode (−). The cathode is the lead closest to the flattened portion of the LED.

The blinked.js program is provided in the code listing below. The purpose of this program is to blink two different LEDs: on BeagleBone (LED USR3) and an external LED via header P8 pin 13. Only a brief description is provided. A detailed description of Bonescript is provided in the next chapter. In the first line of the code, the Bonescript library must be included. Several lines then follow to allow the program to work in version 0.0.1 (original) and 0.2.0 (black) of Bonescript. The pin designators are then assigned to variable names. The pins are then configured as general purpose outputs. The initial state of the LEDs are set to LOW turning the LEDs off. A JavaScript timer is then configured that runs the function “toggle” once every second (1000 ms). Within the function, the variable “state” is toggled between LOW and HIGH, then used to set the state of the pins driving the LEDs.

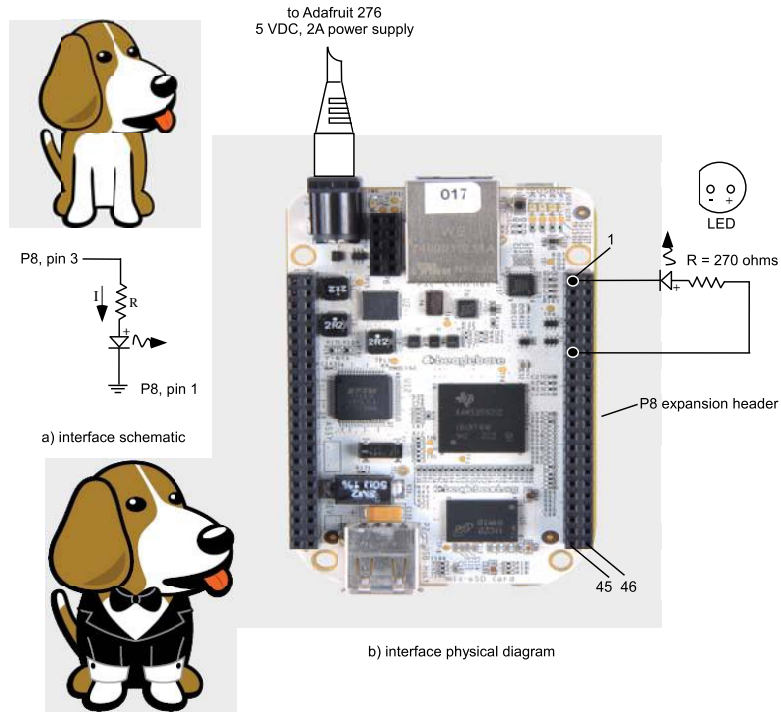
```
//*****
var b = require('bonescript');

//Old bonescript defines 'bone' globally
var pins = (typeof bone !== 'undefined') ? bone : b.bone.pins;

var ledPin = pins.P8_13;
var ledPin2 = pins.USR3;

b.pinMode(ledPin, b.OUTPUT);
```

## 10 1. GETTING STARTED



**Figure 1.5:** Interfacing an LED to BeagleBone. **Notes:** The current being supplied by the BeagleBone pin should not exceed 8 mA.

```
b.pinMode(ledPin2, b.OUTPUT);

var state = b.LOW;
b.digitalWrite(ledPin, state);
b.digitalWrite(ledPin2, state);

setInterval(toggle, 1000);

function toggle() {
    if(state == b.LOW) state = b.HIGH;
    else state = b.LOW;
    b.digitalWrite(ledPin, state);
    b.digitalWrite(ledPin2, state);
}

//*****
```

### 1.9.3 EXECUTING THE BLINKLED.JS PROGRAM

To execute the program, Cloud9 IDE is started as described earlier in the chapter. The `blinkled.js` program is selected and the “run” button is depressed. The LEDs will commence blinking!

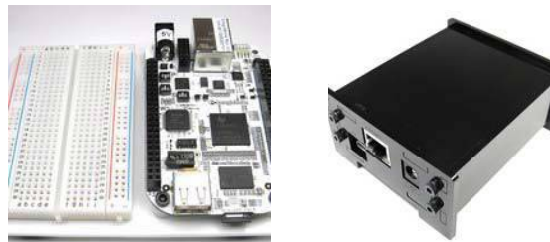
### 1.9.4 EXERCISE 3: DEVELOPING YOUR OWN BONEYARD — AROO!

In this exercise you develop a prototype board to exercise BeagleBone. We have dubbed this the “Boneyard.” In the spirit of “Do-it-yourself (DIY),” we encourage you to develop your own Boneyard by using your creativity and sense of adventure.

If your schedule does not permit time to design and build your own Boneyard, “way cool” prototype boards and cases are available from Adafruit and built-to-spec.com. Adafruit ([www.adafruit.com](http://www.adafruit.com)) provides a plethora of Beagle related products including an Adafruit proto plate and Bone Box pictured in Figure 1.6. Built-to-spec also offers a variety of BeagleBone products including prototype boards and enclosures. Several are shown in Figure 1.7.



**Figure 1.6:** Sample of Adafruit BeagleBone products [[www.adafruit.com](http://www.adafruit.com)].



**Figure 1.7:** Sample of Built-to-spec BeagleBone products [www.built-to-spec.com](http://www.built-to-spec.com)].

For our version of the Boneyard, we have used available off-the-shelf products as shown in Figure 1.8. The Boneyard includes the following:

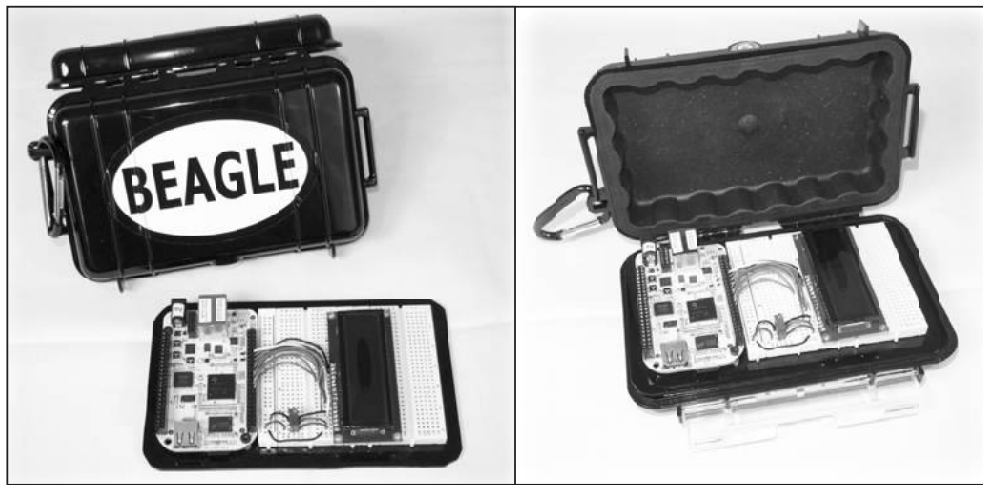
- A black Pelican Micro Case #1040,



## 12 1. GETTING STARTED

- A BeagleBone evaluation board,
- Two Jameco JE21 3.3 x 2.1 inch solderless breadboards and
- One piece of black plexiglass.

We purposely have not provided any construction details. Instead, we encourage you to use your own imagination to develop and construct your own BeagleBone Boneyard. Also, a variety of Beagle related stickers are available from Café Press [www.cafepress.com](http://www.cafepress.com) to customize your boneyard.



**Figure 1.8:** BeagleBone Boneyard (photo courtesy of J. Barrett, Closer to the Sun International).

### 1.10 SUMMARY

This chapter has provided an introduction to BeagleBone, its environment and Beagle community. Also, an overview of BeagleBone features and subsystem was provided. The chapter concluded with several examples to get you started. In the next chapter a detailed introduction to programming and programming tools is provided.

### 1.11 REFERENCES

- Coley, Gerald. *BeagleBone Rev A6 Systems Reference Manual*. Revision 0.0, May 9, 2012, beagleboard.org [www.beagleboard.org](http://www.beagleboard.org)
- Dulaney, Emmett. *Linux All-In-One for Dummies*. Hoboken, NJ: Wiley Publishing, Inc., 2010. Print.



- Octavio, “First steps with BeagleBone.” [www.borderhack.com](http://www.borderhack.com)
- “Adafruit Industries.” [www.adafruit.com](http://www.adafruit.com)
- “Built-to-Spec.” [www.built-to-spec.com](http://www.built-to-spec.com)

## 1.12 CHAPTER EXERCISES

1. What processor is hosted aboard the original BeagleBone? BeagleBone Black? Summarize the features of each processor.
2. What is the difference between BeagleBone computer and a microcontroller based hobbyist board?
3. What is the operating system aboard BeagleBone? What are the advantages of using this operating system?
4. Describe the Beagle community.
5. Where is the operating system stored on the original BeagleBone? BeagleBone Black?
6. What is the Cloud9 IDE?
7. Summarize the features of Bonescript.
8. Describe the concept of open source hardware.
9. What is a BeagleBone Cape? How many can be used simultaneously? How are conflicts prevented between Capes?
10. Modify the `blinkled.js` program such that one LED blinks at three times the rate of the other.