

BUG BRAIN

TM

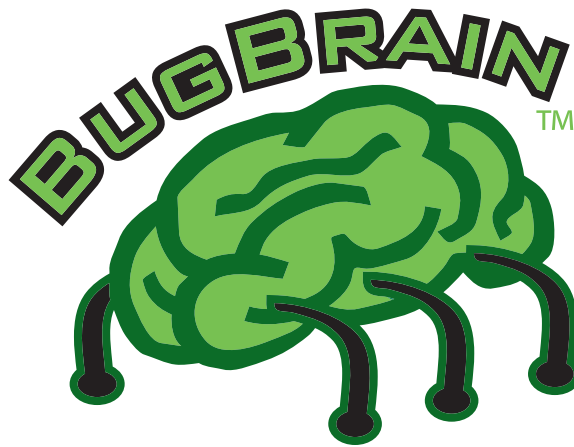


BugBrain: 101 Course Book



Yost Engineering, Inc.

BugBrain: 101 Course Book



Yost Engineering Inc.
630 Second Street
Portsmouth, Ohio 45662


www.YostEngineering.com

888.395.9029 | 740.355.9029
Fax: 888.565.1170 | 740.354.1170
Sales: sales@yostengineering.com
Support: support@yostengineering.com

SAMPLE PAGES

This class is an introduction to building and programming robots using Yost Engineering's BugBrain™ robot kits. Upon completion of this class, students will have built an autonomous walking robot and used a computer and programming software to make their robot interact with its environment.

The course includes a BugBrain™ robotics kit, BugBrain™ assembly manual, the student book "BugBrain 101," as well as a teacher manual, tests, and handouts. (Tools needed for assembly may be acquired with the Yost Engineering BugBrain Assembly Toolkit if not already available in the classroom.)



The BugBrain walking robot kit teaches problem solving, critical thinking skills, computer technology fundamentals, programming fundamentals, and many hands-on technical and fabrication skills. The BugBrain kit comes with all of the necessary mechanical and electronic components, detailed full-color fabrication and instruction manuals, a PC programming cable, all the necessary software, and even an AC adapter so that you don't waste batteries while perfecting your programs.

Yost Engineering, Inc.
630 Second Street
Portsmouth, Ohio 45662
U.S.A.

www.YostEngineering.com

BugBrain: 101 – Course Book

Copyright © 2006 by Yost Engineering Inc. All rights reserved.



SAMPLE PAGES

Table of Contents

Chapter 1: What Is A Computer?	1
The Silly Definition	1
A Great Idea	2
Remembering Numbers	2
Computing with Numbers	4
Input and Output	6
A Peek Inside	7
What is a microcontroller?	10
Chapter 1 Questions	12
Chapter 1 BugBrain Assembly	13
Chapter 2: Binary, Hexadecimal, and ASCII	15
Binary	15
Hexadecimal	16
ASCII	18
Printable ASCII Characters	19
Chapter 2 Questions	21
Chapter 2 BugBrain Assembly	22
Chapter 3: Introduction to Programming	23
What is Machine Code?	23
What is a Program?	26
What Does a Compiler Do?	27
Software and Hardware	28
Chapter 3 Questions	30
Chapter 3 BugBrain Assembly	30
Chapter 4: All About Operators	31
Ready to Operate?	31
Variables in Programming	31
Creating Variables: Variable Declaration and Data Types	32
Storing Data in Variables: The Assignment Operator	33
Operators: Getting the Computer to Compute	36
Arithmetic Operators	36
Relational Operators	37
Logical Operators	38
The String Operator	42
Unary Plus and Minus Operators	42
The Precedence Operator and Order of Operations	42
Other Ways to Manipulate Data	44
Chapter 4 Questions	46
Chapter 4 BugBrain Assembly	48

Chapter 5: Basic BasicX Programming Concepts	49
Variables Rule!	49
More About Variables and Data Types	50
Enumerations and Constants	53
Operators and Data Types	56
Type Conversion	56
Rounding	59
Variable Initialization	60
Comments	60
Chapter 5 Questions	62
Chapter 5 BugBrain Assembly	65
Chapter 6: Introduction to BasicX	67
BasicX Programming Environment	67
Intro to BasicX Code	72
Debug.Print	73
Goodbye Earth	75
PutPin and GetPin: Output and Input!	77
Chapter 6 Questions	83
Chapter 7: Control Structures	85
Introduction	85
If Statements	85
Introduction to Loops	90
For Loops	102
Chapter 7 Questions	105
Chapter 8: Sub Procedures and Functions	111
Subprograms	111
Sub Procedures	111
Functions	119
ByVal vs. ByRef	122
Chapter 8 Questions	126
Chapter 9: Speaker and Servos	131
Ahhh Freqout!	131
Take Your Bug for a Walk	134
Chapter 9 Questions	143

Appendix A: Hardware/Components Information 145

Appendix B: Glossary 149

Index 159

SAMPLE PAGES

Chapter 1: What Is A Computer?

The Silly Definition

Most introductory computer books answer this question with a definition something like this: "A computer is a device that receives input, stores and processes that input, and produces output." Sometimes this definition is also accompanied by an illustration such as that shown in Figure 1:

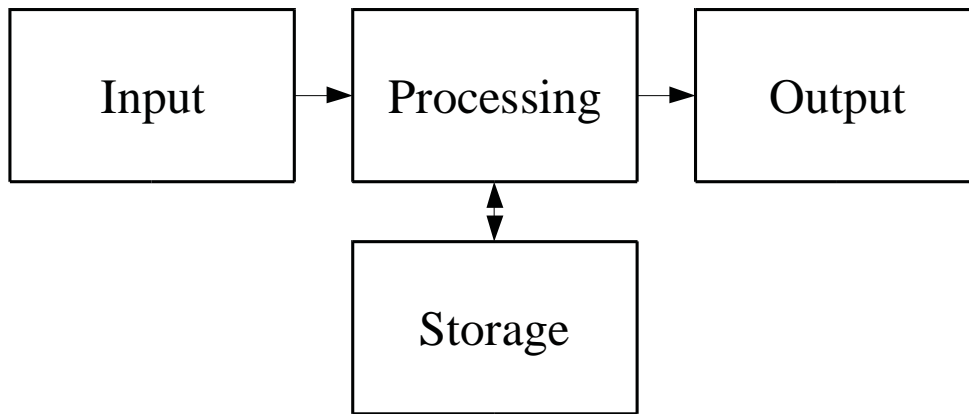
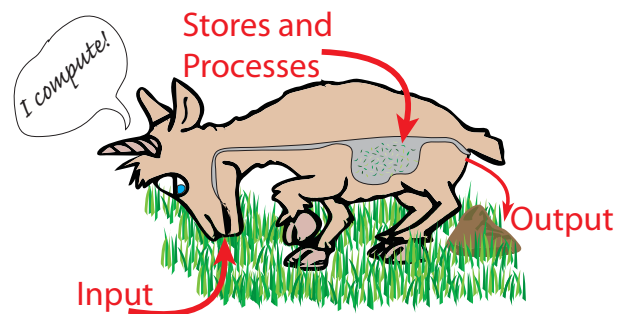


Figure 1 – How a Computer Works

While this definition isn't really wrong, it isn't exactly a very useful definition either because it is so abstract. In fact, looking at the definition and diagram above doesn't get you any closer to understanding how a computer really works or what a computer really does. To illustrate this definition's shortcomings, consider a goat: a goat takes input (grass), stores and processes that input (stomach and digestive system), and produces output (poop). Thus, by this definition a goat is a computer! By similar logic a lawnmower is a computer: a lawnmower takes input (gasoline), stores and processes that input (gas tank & engine), and produces output (exhaust and grass clippings).



The problem is that the definition and diagram above are simply too abstract. If we want to know how a computer really works then we need a more detailed and descriptive definition; one that explains the parts of a computer and how these parts interact with one another to do things. So, let's start at the beginning...

A Great Idea

Most great inventions in history, the computer included, were sparked by simple, yet powerful, ideas. The simple idea that inspired the invention of the computer is that of "digital memory". *Digital memory* is, simply put, the idea of having a machine store information using a series of switches that are each either on or off. Since each switch holds its last state and only changes its state when told to, each switch can be thought of as "remembering" what state it was told to be in. Thus, in your house, each light switch can be thought of as "remembering" if it is currently on or off.



The earliest computing devices used mechanical methods such as gears or relays to represent these switch settings while later devices used holes punched in paper tape or cards, magnetic beads, or vacuum tubes for memory. Modern computer systems usually use electronic integrated circuits for *main memory* and optical and magnetic systems for *storage memory*. Since the main memory of a computer is electronic in nature, the switches are cleared when the circuits lose power. Storage memory is like a person's long-term memory; a much larger amount of information can be contained in storage memory, but the data contained in storage must first be moved to immediate memory for the computer to use it, just as a person must move information from long-term to short-term memory in order to think about it. Some examples of *storage devices* are hard drives, CD/DVD drives, and USB memory sticks. Storage devices are just devices that provide access to storage memory.

But how, you ask, how does an idea as simple as an electrical switch holding its state spark an invention as powerful as the computer? Well, to truly understand this we need to take another step in the inspiration/invention chain...

Remembering Numbers

In order for a switch to become a useful way to "remember" things, a couple of things needed to be invented. First, a way of changing the state of the switch electrically rather than manually needed to be developed. This is what the vacuum tubes and relays did in old systems and what transistors do now in modern integrated circuits. Second, a method of storing useful data, such as numbers or letters or instructions, by using a group of these "switches" needed to be developed. The binary number system allows the storage of numbers as patterns of switch settings and the ASCII system