What You’ll Need…

- Minimum System Requirements
  - Windows XP/Vista/Win7, Mac not supported
  - PIII-450MHz+, 256MB+ RAM, 120MB Hard Disk Space
  - Administrator Access on the PC
  - 1 USB port available for Cortex programming

- Software & Installation
  - Software provided on CD
  - Installer auto runs from CD
  - 4-month license begins at installation, 3 seats
  - Updates available at [http://www.intellitekdownloads.com](http://www.intellitekdownloads.com)
What gets installed?

- Prolific USB to Serial Driver
- easyC Integrated Development Environment
- IFI/Intelitek Loader
- easyC libraries
- IFI VEXnet firmware upgrade*
- Many example projects
- 14 Integrated Tutorials

* Note: Firmware upgrade requires administrator mode on Windows VISTA. Right click on “IFI VEXnet Firmware Upgrade” in Programs Menu and select “Run as administrator”.
Key Software Locations

- Start Program Menu items are added
  - easyC V4 for Cortex
- Desktop icons

- Projects and Libraries are added here
  - My Documents\Intelitek\easyC V4 for Cortex
easyC Software Installation

- Must be in the administrator account (or administrator mode)
- Run the easyC_V4_for_Cortex4001.exe program
- Follow the on screen instructions
- Check the “Install Prolific USB to Serial adapter driver“ checkbox before clicking the Finish button, the driver installer will startup after a few seconds
- Sample files copied into a “Intelitek” subfolder in the Documents (or My Documents in XP) folder
- Sample files must be copied to each users folder if the software will be shared by multiple users on the same machine.
easyC Software Installation

- On the first startup of the software, there will be a prompt for registration code
- Enter the registration code provided with your CD
easyC Software Overview

- Block based programming
  - Drag & Drop C function blocks into a C program structure
  - Raw C code programming
- Create your code from scratch in the text editor
  - Import your own functions/libraries
  - Create user-code blocks that can be placed in the block programming window
- Lots of built-in functions/blocks; similar to past BRAIN
  - setservo(), setmotor(), getjoystick(), getdigitalinput()
- Compiles and downloads to your robot
VEX Cortex Review

- **USB**: Standard Serial Interfaces (UART, I2C)
- **Analog in**: 1
- **Digital in/out**: 8, 1, 12
- **Speaker Out**: SP
- **2-wire motor**: 1, 2, 9, 10
- **3-wire PWM servo/motor ctrl**: 2
- **Gnd (Blk)**, **+5v (Red)**, **Sig (Wht)**

Copyright © 2010 BEST Robotics, Inc. All rights reserved.

August 1, 2010
# BEST Default Program

<table>
<thead>
<tr>
<th>Motor/Servo Output</th>
<th>Joystick Channel</th>
<th>Motor Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Positive Direction</td>
</tr>
<tr>
<td>Motor 1 (Arcade Right)</td>
<td>Channel 1 (Lt, Rt) Channel 2 (Fwd/Rev)</td>
<td>None</td>
</tr>
<tr>
<td>Motor 2</td>
<td>Channel 1</td>
<td>Digital Input 1</td>
</tr>
<tr>
<td>Motor 3</td>
<td>Channel 2</td>
<td>Digital Input 3</td>
</tr>
<tr>
<td>Motor 3</td>
<td>Channel 4</td>
<td>Digital Input 5</td>
</tr>
<tr>
<td>Motor 4</td>
<td>Channel 5</td>
<td>Digital Input 7</td>
</tr>
<tr>
<td>Motor 3</td>
<td>Channel 6</td>
<td>None</td>
</tr>
<tr>
<td>Motor 7</td>
<td>Channel 3 Inverse</td>
<td>None</td>
</tr>
<tr>
<td>Motor 8</td>
<td>Channel 4</td>
<td>None</td>
</tr>
<tr>
<td>Motor 9</td>
<td>Channel 4 Inverse</td>
<td>None</td>
</tr>
<tr>
<td>Motor 10 (Arcade Left)</td>
<td>Channel 1 (Lt, Rt) Channel 2 (Fwd/Rev)</td>
<td>None</td>
</tr>
</tbody>
</table>
Programming Vocabulary

- **Compile** – changes your C program into object code that the linker understands.

- **Link** – combines your program’s object code with the Intelitek library and other libraries to create code that is executable on the Cortex processor.

- **Download / Bootload** – transfers the machine code version of your program from the PC to the Cortex where it will execute (the IFI/Intelitek Loader will perform the transfer via the PC USB cable)
Programming Flow

1. Write It.
   - Block-based
     - Drag & Drop (Menu Cfg)
   - From Scratch
     - User Created C Program

2. Save It.
   - Project Files

3. Compile & Link.
   - (aka Build)
   - Integrated Design Environment (IDE)
     - easyC V4 for Cortex
     - Libraries (C Functions)

4. Download It.

5. Test It.
   - Target (Cortex)
Programming (step-by-step)

1. Create a new project.
   - Select “New Standalone Project” from the File menu
   - Select “Joystick Project (wifi)” from the popup window

2. Create your program; save your program.
   - Drag & drop programming blocks to create your program

3. Build (i.e., compile and link) the code.
   - Use the “Build and Download” option from the Build and Download Menu
   - Errors will be reported in the “Outputs & Tasks” window

4. Download the program to the Cortex
   - Automatic if using the “Build and Download” option
   - IFI/Intelitek Loader window will popup showing progress
Creating a new project
Type of project
Navigating easyC (1)

This shows the easyC layout with a New Standalone Project started and all of the View options turned on.
Navigating easyC (2)
Navigating easyC (3)

```c
void main ( void )
{
    Config
    Globals
    BEGIN
    Variables
    END
}
```
Navigating easyC (4)
Navigating easyC (5)
Set Controller Configuration

The Controller Configuration window is used to identify what the various interfaces will be used for and whether the digital interfaces are configured as inputs or outputs. The Controller Configuration window can be accessed via the Project menu. The example shown is the BEST default program.
Creating a Program

Drag blocks from here to build your program. Context sensitive so you can’t make a program structure mistake.
The programming blocks in the Program Flow folder are primarily standard C language constructs.

The items unique to easyC are as follows:

- Timer: Gives access to six software timers.
- Wait: Suspends program execution.
- Assignment: For writing a custom expression.
- Print To Screen: For sending output to the terminal window while a program is running (usually used for debugging).
- Graphic Display: For sending output to the Graphic Display window while a program is running.
- Comment: For placing a comment statement in the code.
- User code: Can construct any valid C statement/s.
The switches used by BEST can be used in conjunction with the Digital Input or the Interrupt Watcher programming blocks. The Digital Input block will detect whether a switch is open (returns a 1) or closed (returns a 0). The Interrupt Watcher block detects whether a switch state has changed.

Note that the Bumper Switch, Limit Switch and Digital Input blocks are all functionally the same, only their names are different.
The Motor and Servo Module blocks can be used interchangeably. However it is suggested to use the Motor Module with motors and the Servo Module with servos just to make the code more readable.

BEST does not currently use any devices that would make use of the Digital Output block.
Many prewritten functions are available to interface with the Joysticks. These standard functions can be used or you can read the Joystick values into variables with the “Get Joystick” blocks and then manipulate these values before sending them to the motors or servos.
The Battery blocks can be used to detect the voltage of the main battery or the backup battery. The voltages can then be reported via the terminal window or the graphical display window.
easyC Mathematical Functions

```c
tvoid main ( void )
{
    Config
    
    Global
    
    BEGIN
    
    Variables
    
    END
}
```
The joystick and accelerometer limit return values are +127 and -127 as shown in the figure. The buttons have a return value of 0 when pressed and 1 when released.
### Whole Number Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>-127 to 127</td>
</tr>
<tr>
<td>unsigned char</td>
<td>0 to 255</td>
</tr>
<tr>
<td>signed char</td>
<td>-127 to 127</td>
</tr>
<tr>
<td>Int</td>
<td>-32,768 to 32,768</td>
</tr>
<tr>
<td>unsigned int</td>
<td>0 to 65,535</td>
</tr>
<tr>
<td>short</td>
<td>-32,767 to 32,767</td>
</tr>
<tr>
<td>unsigned short</td>
<td>0 to 65,535</td>
</tr>
<tr>
<td>short long</td>
<td>-8,388,608 to 8,388,607</td>
</tr>
<tr>
<td>unsigned short long</td>
<td>0 to 16,777,215</td>
</tr>
<tr>
<td>long</td>
<td>-2,147,483,647 to 2,147,483,647</td>
</tr>
<tr>
<td>unsigned long</td>
<td>0 to 4,294,967,295</td>
</tr>
</tbody>
</table>

### Decimal Number Variables (6 digits of precision)

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>1e-38 to 1e+38</td>
</tr>
<tr>
<td>double</td>
<td>1e-38 to 1e+38 (same as float in this environment)</td>
</tr>
</tbody>
</table>
The LCD blocks are for a VEX LCD device that is not used by BEST.

The User Functions folder has the functions that you have created. You can just drag and drop them into your code to call them.
Select “Build and Download” from the “Build and Download” menu to build (compile/link) your program and download it to your robot.

The IFI/Intellitek Loader will run after build is complete.
To determine where the errors are in your code, open the “C Programming” window from the Window menu. The line numbers of the code will be displayed.

Compile errors will be reported in the output & tasks window.
Successful Build

The HEX file was successfully built. Would you like to download the program?

Yes  No
Download the Program

Option 1: Direct USB Download

Uses only the A-A USB cable.

Battery is not needed.
Download the Program

Option 2: Tethered Download

Uses the programming cable with A-A USB tether.

Allows advanced debugging.
Download the Program

Option 3: Wireless Download

Uses the programming cable with WiFi USB Keys.

Battery is needed

Allows advanced debugging.
The first time the “IFI/intelitek Loader” runs, a message about updating the master code will pop up.

Simply check the “don’t show this message again” checkbox and continue.
Advanced Debugging

Advanced debugging functions in easyC are located under the Tools menu.

(Terminal Window, On-Line Window, Graphic Display)
The On-Line Control Window is a good way to verify hardware function.

It allows direct control of the robot’s I/O to verify hardware connections are correct.
The Graphic Display window is a customizable window for your program to display the value of internal variables in real-time.

If the Graphic Display is not active; select the Stop button, then mark the Graphic Display box and then select the Start button.

The output shown here is from the “Graphical Display Joystick Test” sample program.

```
Channel 1 = 0
Channel 2 = 0
Channel 3 = 0
Channel 4 = 0

Channel 5: Down = 0; Up = 0
Channel 6: Down = 0; Up = 0

Channel 7: Down = 0; Up = 0; Left = 0; Right = 0
Channel 8: Down = 0; Up = 0; Left = 0; Right = 0

Gamepad Accelerometer: AxisX = 31; AxisY = 0

Loop Time = 22 msec
```