

Volume

1

A-WIT TECHNOLOGIES INC.

---

... a passion for execution ...

# C Stamp Infrastructure for the iRobot® Create

---

Version 1.1

A-WIT TECHNOLOGIES INC.

# C Stamp Infrastructure for the iRobot® Create - Manual

---

© A-WIT Technologies Inc.  
Phone (800) 985-AWIT • Fax (800) 985-2948

---

# Table of Contents

Getting Started	1
Notices	2
Getting Support	2
C Stamp Program Example for the iRobot	
Create	2
Downloading and Running Your Program	5
Developing Your Own Programs and	
Projects	5
WC Program Structure	6
iRC_power_on	6
iRC_start	6
iRC_baud	7
iRC_safe	7
iRC_full	7
iRC_demo	7
iRC_drive	8
iRC_drive_direct	8
iRC_light_LEDs	8
iRC_digital_outputs	8
iRC_pwm_low_side_drivers	9
iRC_low_side_drivers	9
iRC_send_IR	9
iRC_sensors	10
iRC_wait_time	10
iRC_wait_distance	10
iRC_wait_angle	10
iRC_wait_event	10
Terms and Conditions	11

---

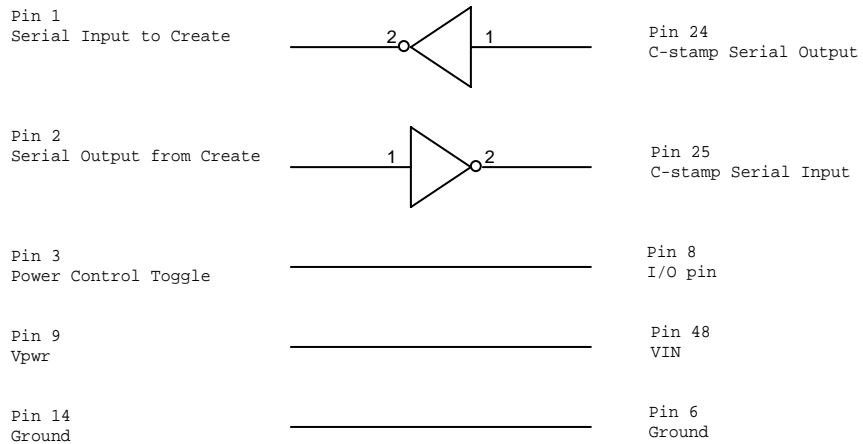


## Introduction

Although the iRobot Create robot can be purchased with a default microcontroller, it can be controlled with a C Stamp. The C Stamp offers the advantages of more expandability, better control, and a better learning experience through more access to all the hardware details. With a simple set of connection between the C Stamp and the Create robot, and this C Stamp software infrastructure users can easily control the Create robot with the C Stamp. This software infrastructure is based on the “iRobot Create Open Interface (OI) Specification”, and it consists of 20 functions.

## Getting Started

The following figure describes the connections between the iRobot Create and the C Stamp. Since the iRobot Create communicates via a serial port, it is relatively easy using the C Stamp serial port interface. The diagram below includes connections for power to the C Stamp. The left side of the diagram depicts the iRobot Create pins while the right side of the diagram depicts the C Stamp pins.



## Notices

CSTAMP™ and CSTAMP™ Related Hardware Products, Software Products and Documentation are developed and distributed by A-WIT Technologies, Inc. All rights reserved by A-WIT Technologies, Inc. A-WIT SOFTWARE OR FIRMWARE AND LITERATURE IS PROVIDED “AS IS,” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL A-WIT BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR FIRMWARE OR THE USE OF OTHER DEALINGS IN THE SOFTWARE OR FIRMWARE.

MPLAB C-18 and MPLAB C-18 Users Guide is reproduced and distributed by A-WIT Technologies, Inc. under license from Microchip Technology Inc. All rights reserved by Microchip Technology Inc. MICROCHIP SOFTWARE OR FIRMWARE AND LITERATURE IS PROVIDED “AS IS,” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL MICROCHIP BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR FIRMWARE OR THE USE OF OTHER DEALINGS IN THE SOFTWARE OR FIRMWARE.

## Getting Support

If possible, please check the C Stamp website [www.c-stamp.com](http://www.c-stamp.com) under SUPPORT for any updates to documentation, changes, or notices that may have become available since your Installation CD was produced. If you continue to have any issues for which a solution is not found in the aforementioned website, please e-mail [tech\\_support@a-wit.com](mailto:tech_support@a-wit.com) for help.

## C Stamp Program Example for the iRobot Create

The following program is a demo of how to use the C Stamp software infrastructure for the iRobot Create robot. The robot will drive until it hits a wall, and at that point, it will back up, turn slightly and then continue forward.

In terms of software development, it is always necessary to make the following global declarations to use this infrastructure.

```
// The amount of time to wait after a function call
// default 10 mS
BYTE delay;
```

```
// The serial communication baud rate
// default 57.6
float baud_rate;
```

The sample program is as follows:

```
#include "CS110000.h"
```

```
// The amount of time to wait after a function call
// default 10 mS
BYTE delay;
```

```
// The serial communication baud rate
// default 57.6
float baud_rate;
```

```
void main(void)
```

```
{
  RAM BYTE sensor_data = 0;
  RAM BYTE tmp = 0;
  RAM BYTE advance_play = 0;
  RAM BYTE count = 0;
  RAM BIT test = 0;
  RAM int adder = 1;
  RAM int speed_left = 200;
  RAM int speed_right = 200;
  RAM int requested_right = 0;

  iRC_power_on(8);

  tmp = iRC_start();
  tmp = iRC_safe();
  tmp = iRC_drive_direct(speed_right, speed_left);

  while (1){
    test = ~test;

    if (count == 255)
      adder = -1;
    if (count == 0)
      adder = 1;
```

```

count = count + adder;

if (test){
    STPIND(46, HIGH);
    advance_play = 0x02;
}
else{
    STPIND(46, LOW);
    advance_play = 0x08;
}

tmp = IRC_light_LEDs(advance_play, count, count);

sensor_data = IRC_sensors(7);
sensor_data = sensor_data & 0x03;

if (sensor_data == 3){ // both bumpers
    STPIND(40,HIGH);
    speed_right = speed_right * -1;
    speed_left = speed_left * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(500);
    speed_right = speed_right * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(1000);
    speed_left = speed_left * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
}
else if (sensor_data == 2){ // left bumper
    STPIND(41, HIGH);
    speed_right = speed_right * -1;
    speed_left = speed_left * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(300);
    speed_left = speed_left * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(600);
    speed_right = speed_right * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
}
else if (sensor_data == 1){ // right bumper
    STPIND(41, HIGH);
    speed_right = speed_right * -1;
    speed_left = speed_left * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
}

```

```

    PAUSE(150);
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(150);
    speed_right = speed_right * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(600);
    speed_left = speed_left * -1;
    tmp = IRC_drive_direct(speed_right, speed_left);
    PAUSE(100);
}
else{
    STPIND(40, LOW);
    STPIND(41, LOW);
    STPIND(42, LOW);
}
}

//power off
PAUSE(5000);
STPIND(8, HIGH);
PAUSE(100);
STPIND(8, LOW);
STPIND(46, HIGH);
STOP();
}

```

## Downloading and Running Your Program

Download and start the program by pushing and letting go of the RESET button while pushing the START button. Then you can let go of the START button.

## Developing Your Own Programs and Projects

Now that you have successfully developed and run your program, it is easy to move on to more complex and elaborate projects and circuits of your own. To do this, your first step should be to thoroughly study the next chapter of this manual: Infrastructure for the iRobot Create Reference. Also, reference the iRobot Create Open Interface (OI) Specification.

## Infrastructure for the iRobot Create Reference

This chapter describes the elements of the C Stamp Infrastructure for the iRobot Create robot, which consists of 18 functions. While this manual is generally written in a variable size font, code and its comments are written with a “fixed font”. Additionally, keywords, functions, and commands of the infrastructure are in “**bold**”. Generic syntax is in a “***bold and italics fixed font***”.

### WC Program Structure

The only requirement to use this infrastructure in a C Stamp program is to make the following global declarations:

```
// The amount of time to wait after a function call
// default 10 mS
BYTE delay;

// The serial communication baud rate
// default 57.6
float baud_rate;
```

NOTE: More information can be found on the previous functions by referencing the iRobot Open Interface documentation.

### iRC\_power\_on

```
void iRC_power_on(BYTE power_pin);
```

Powers on the robot using the specified I/O pin.

### iRC\_start

```
BYTE iRC_start(void);
```

This function must be called before commands can be sent to the robot.

## iRC\_baud

**BYTE iRC\_baud(BYTE baud\_code);**

Changes the rate of communication between the C Stamp and the robot according to the table below.

<i>baud_code Values</i>	
Code	Baud Rate (Kbps)
4	4.8
5	9.6
7	19.2
9	38.4
10	57.6
11	115.2

## iRC\_safe

**BYTE iRC\_safe(void);**

Puts the robot in “safe” mode.

## iRC\_full

**BYTE iRC\_full(void);**

Puts the robot in “full” mode.

## iRC\_demo

**BYTE iRC\_demo(BYTE demo\_number);**

Starts the requested robot built in demo.

## iRC\_drive

```
BYTE iRC_drive(int velocity, int radius);
```

Controls the robot's drive wheels with respect to the requested velocity and radius.

## iRC\_drive\_direct

```
BYTE iRC_drive_direct(int right_velocity,  
                      int left_velocity);
```

Controls the robot's drive wheels with respect to the requested left and right wheel velocities.

## iRC\_light\_LEDs

```
BYTE iRC_light_LEDs(BYTE advance_play,  
                   BYTE power_color,  
                   BYTE power_intensity);
```

Controls the three robot LEDs according to the following table.

<b>advance_play Value</b>	<b>Advance</b>	<b>Play</b>
0x02	off	on
0x08	on	off
0x00	off	off
0x0A	on	on

Also note that the power LED parameters (power\_color and power\_intensity) range from 0-255.

LED bits - send 8 for advance LED, 1 for play LED

Power color (0-255) 0-green, 255-red

Power intensity (0-255) 0-off, 255 -full intensity

## iRC\_digital\_outputs

```
BYTE iRC_digital_outputs(BYTE output_bits);
```

Controls the state of the three digital output pins on the 25 pin Cargo Bay Connector according to the following table.

<i>Value 0 → low (0V)</i>	<i>Value 1 → high (5V)</i>
pin 20 → digital-out-2 → bit 2	
pin 7 → digital-out-1 → bit 1	
pin 19 → digital-out-0 → bit 0	
- disregard all other bits -	

### iRC\_pwm\_low\_side\_drivers

```
BYTE iRC_pwm_low_side_drivers(
  BYTE low_side_driver_2_duty_cycle,
  BYTE low_side_driver_1_duty_cycle,
  BYTE low_side_driver_0_duty_cycle);
```

This function lets you control the three low side drivers with variable power.

### iRC\_low\_side\_drivers

```
BYTE iRC_low_side_drivers(BYTE driver_bits);
```

This function turns the three low side drivers off or on according to the following table.

<i>Value 0 → off</i>	<i>Value 1 → on</i>
pin 24 → side driver 2 → bit 2	
pin 22 → side driver 1 → bit 1	
pin 23 → side driver 0 → bit 0	
- disregard all other bits -	

### iRC\_send\_IR

```
BYTE iRC_send_IR(BYTE value);
```

Sends the requested byte out of low side driver 1.

## iRC\_sensors

**WORD iRC\_sensors(BYTE packed\_ID);**

Requests and receives the requested sensor information. This function is only implemented for sensor packets 7-42. Moreover, some sensor data may need to be put into to an integer after the function call. Each packet is outlined in the iRobot Create Open Interface documentation.

## iRC\_wait\_time

**BYTE iRC\_wait\_time(BYTE time);**

Causes the robot to wait for the specified amount of time in 15 mS intervals/resolution.

## iRC\_wait\_distance

**BYTE iRC\_wait\_distance(int distance);**

Causes the robot to wait for the specified distance in mm.

## iRC\_wait\_angle

**BYTE iRC\_wait\_angle(int angle);**

Causes the robot to wait for the specified angle in degrees.

## iRC\_wait\_event

**BYTE iRC\_wait\_event(BYTE event\_number);**

Causes the robot to wait for an event or the inverse of an event. To wait for the inverse of event, send a negative value.

## Terms and Conditions

### **Quality Assurance**

A-WIT has stringent quality control procedures in place to insure the best quality products.

### **90-Day Limited Warranty**

A-WIT Technologies, Inc warrants its products against defects in materials and workmanship for a period of 90 days. If you discover a defect, A-WIT Technologies, Inc. will, at its option, repair, replace, or refund the purchase price. After 90 days, products can still be sent in for repair or replacement, but there will be a \$10.00USD minimum inspection/labor/repair fee (not including return shipping and handling charges).

### **14-Day Money-Back Guarantee**

If, within 14 days of having received your product, you find that it does not suit your needs, you may return it for a refund. A-WIT will refund the purchase price of the product in the form of a check, excluding shipping/handling costs, once the product is received. This refund does not apply if the product has been altered or damaged. If you decide to return the products after the 14-day evaluation period, a 20% restocking fee will be charged against a credit.

### **Disclaimer**

Warranty does not apply if the product has been altered, modified, or damaged. A-WIT makes no other warranty of any kind, expressed or implied, including any warranty of merchantability, fitness of the product for any particular purpose even if that purpose is known to A-WIT, or any warranty relating to patents, trademarks, copyrights or other intellectual property. A-WIT shall not be liable for any injury, loss, damage, or loss of profits resulting from the handling or use of the product shipped.

### **How to Return a Product**

When returning, you must first e-mail [sales@a-wit.com](mailto:sales@a-wit.com) for a Return Merchandise Authorization number. No packages will be accepted without the RMA number clearly marked on the outside of the package. After inspecting and testing, we will return your product, or its replacement using the same shipping method used to ship the product to A-WIT within 30 days. In your package, please include a daytime telephone number and a brief explanation of the problem.

Please contact our Sales Department at [sales@a-wit.com](mailto:sales@a-wit.com) if you have any questions regarding our warranty policy or if you are requesting an RMA number.

