Innovation First, Inc.

EDU Default Software Reference Guide

For the Isaac16 EDU RC

```
'>>>>>>>>>>>>1 Joystick Drive - Optional Code - Install Jumper on SW5<<<<<<<<<

'   This code mixes the Y and X axis on Port 1 to allow one joystick drive.
'   Joystick forward  = Robot forward
'   Joystick backward = Robot forward
'   Joystick right    = Robot rotates right
'   Joystick left     = Robot rotates left

        if rc_sw5 = 0 then skipOption5
                  temp = drive_R
                  drive_R = (((2000 + temp  + drive_L - 127) Min 2000 Max 2254) - 2000)
                  drive_L = (((2000 + temp  - drive_L + 127) Min 2000 Max 2254) - 2000)
                  drive_L = 254 - drive_L
        skipOption5:


'>>>>>>>>>>>>Reverse Drive Direction - Optional Code - Install Jumper on SW6<<<<

'   This code simply reverses the drive outputs, reversing the robot direction.

        if rc_sw6 = 0 then skipOption6
                  temp = Drive_R
                  drive_R = drive_L
                  drive_L = temp
        skipOption6:


'>>>>>>>>>>>>Limit Switches - Optional Code - Install Jumper on SW8<<<<

'   This code uses the switch inputs to stop motor movement in a specific direction.
'   If a switch is ON, the value is 1.  If a switch is OFF, the value is 0.

        if rc_sw8 = 0 then skipOption8
                  if rc_sw1 = 0 then sw1_off
                            p3_x = p3_x MAX 127           'PWM1 wont go CCW if rc_sw1 is ON
                  sw1_off:
                  if rc_sw2 = 0 then sw2_off
                            p3_x = p3_x MIN 127           'PWM1 wont go CW  if rc_sw2 is ON
                  sw2_off:
                  if rc_sw3 = 0 then sw3_off
                            p3_wheel = p3_wheel MAX 127   'PWM3 wont go CCW if rc_sw3 is ON
                  sw3_off:
                  if rc_sw4 = 0 then sw4_off
                            p3_wheel = p3_wheel MIN 127   'PWM3 wont go CW  if rc_sw4 is ON
                  sw4_off:

                  relay1_fwd = p1_sw_trig &~ rc_sw1       'RLY1 wont go CW  if rc_sw1 is ON
                  relay1_rev = p1_sw_top  &~ rc_sw2       'RLY1 wont go CCW if rc_sw2 is ON
                  relay3_fwd = p3_sw_trig &~ rc_sw3       'RLY3 wont go CW  if rc_sw3 is ON
                  relay3_rev = p3_sw_top  &~ rc_sw4       'RLY3 wont go CCW if rc_sw4 is ON
        skipOption8:
```

FIRST SM
EDUrobotics Kit

Table of Contents

1. Overview

The Isaac16 EDU Robot Controller & Motherboard comes pre-programmed to perform basic driving and movement functions. There are four additional pre-programmed options that provide the most commonly used driving and control enhancements. The Isaac16 EDU RC can be re-programmed to create almost any custom motion based on operator and sensor inputs.

It is important to understand that the programs that are pre-programmed are located on the Isaac16 EDU Motherboard, not on the Isaac16 EDU RC (the black box). This means that each robot with a Motherboard can have a different program, while sharing the same Isaac16 EDU RC.

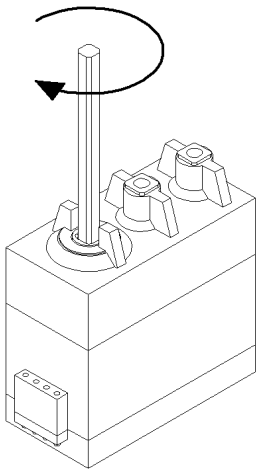
2. Default Pre-Programmed Software

The following table describes the functions of the default code that is pre-programmed into the Isaac16 EDU Motherboard. This default code requires a joystick connected to Port 1 and Port 3 of the Operator Interface. The main function of this software is to provide two-joystick drive, with Port 1 - Y axis controlling the left wheel(s) and Port 3 - Y axis controlling the right wheel(s).

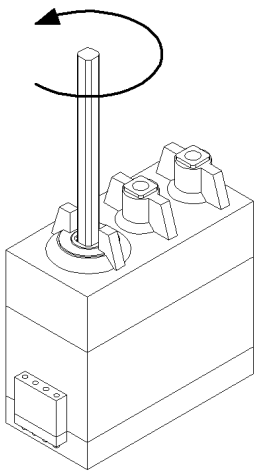
If you need to restore the default program on the motherboard, you can download the EDU Default Code at www.EduRobotics.com.

Input	Output	Notes (see CW and CCW below)
Port 1 – Y Axis	PWM1 and PWM3	Y-axis forward = CCW
Port 1 – X Axis	PWM5	X-axis right = CW
Port 1 – Wheel	PWM7	Wheel forward = CCW
Port 1 – Top Button	RLY1 and RLY2	Button Pressed = CCW
Port 1 – Trigger Button	RLY1 and RLY2	Button Pressed = CW
Port 3 – Y Axis	PWM2 and PWM4	Y-axis forward = CW
Port 3 – X Axis	PWM6	X-axis right = CW
Port 3 – Wheel	PWM8	Wheel forward = CCW
Port 3 – Top Button	RLY3 and RLY4	Button Pressed = CCW
Port 3 – Trigger Button	RLY3 and RLY4	Button Pressed = CW

Clockwise (CW) Rotation



Counter-Clockwise (CCW) Rotation



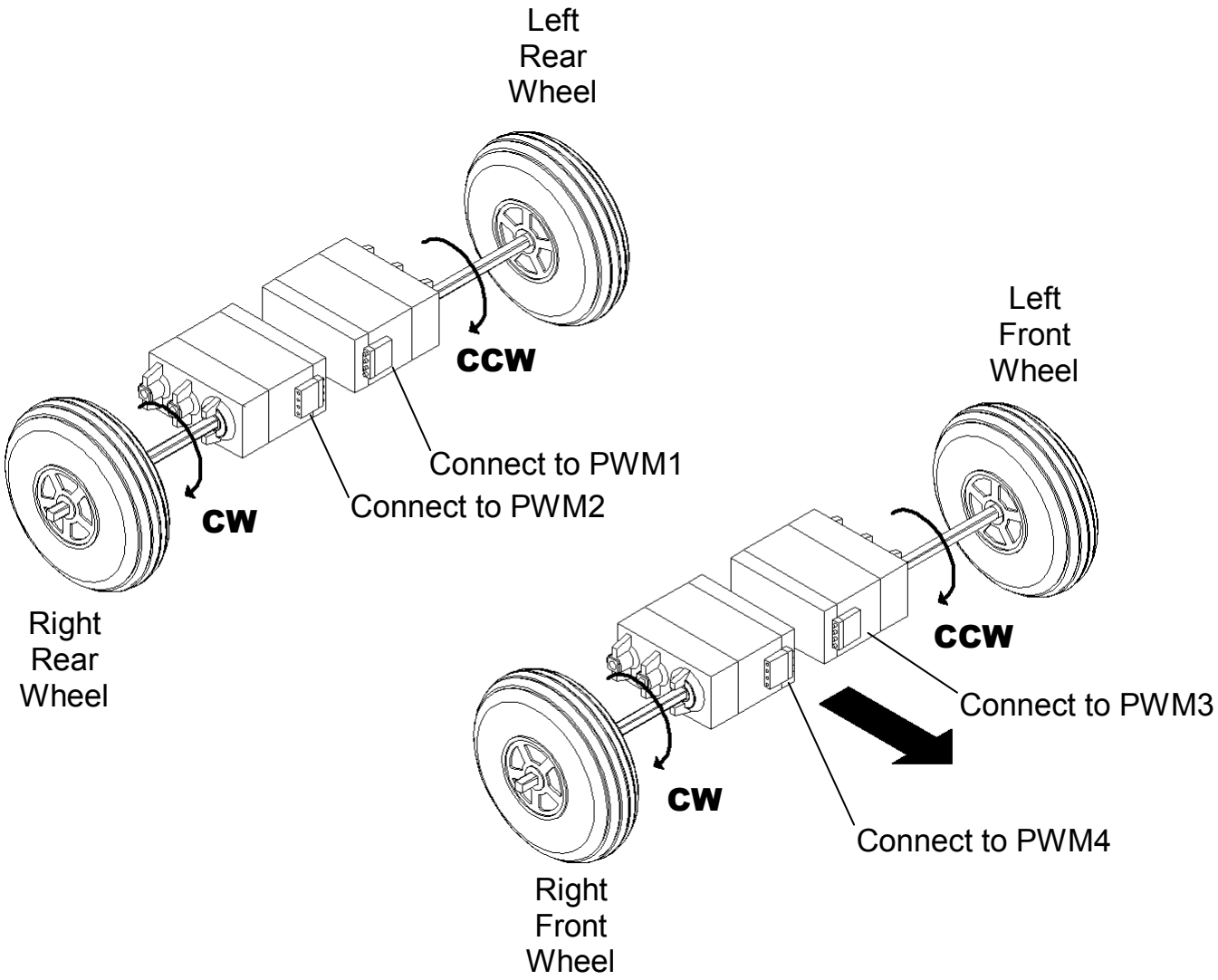
Example (using the default software): Press the Trigger button on the Port 1 Joystick to make a motor connected to RLY1 rotates clockwise. Press the Top button on the Port 1 Joystick to make a motor connected to RLY1 rotates counter-clockwise.

Example (using the default software): Move the Y axis on the Port 1 Joystick forward to make a motor connected to PWM1 turn counter-clockwise. The further you push the joystick, the faster the motor turns. From a neutral position move the Y axis on the Port 1 Joystick backwards to make a motor connected to PWM1 turn clockwise. Again, the further you push the joystick, the faster the motor turns.

3. Robot Drive Direction and Wheel Rotation

Typical two-wheel and four-wheel robots require the motors to rotate in different directions based on there mounting orientation. The diagram below shows the motors and wheels of a four-wheel robot driving forward. The left wheels must rotate counter-clockwise (CCW) and the right wheels must rotate clockwise (CW) to provide a forward movement of the robot. The EDU Default program on the motherboard accommodates this.

Input	Output	Notes (see CW and CCW below)
Left Rear Motor	PWM1	Port 1 Y-axis forward = CCW
Left Front Motor	PWM3	Port 1 Y-axis forward = CCW
Right Rear Motor	PWM2	Port 3 Y-axis forward = CW
Right Front Motor	PWM4	Port 3 Y-axis forward = CW



4. Optional Pre-Programmed Software

The following sections describe the functions of the optional code that is pre-programmed into the Isaac16 EDU Motherboard. The optional code is part of the EDU Default Code. Installing a 2-pin jumper on the motherboard switch inputs SW5-SW8 accesses the optional code. The 2-pin jumper shorts/connects both switch input pins together for that corresponding input.

If you need to restore the EDU Default program on the motherboard, you can download the default code at www.EduRobotics.com.

5. SW5 Option – One Joystick Drive

The one joystick drive option can be invoked by installing a 2-pin jumper, connecting both pins of SW5 on the Motherboard. The software allows Port 1 - X axis and Port 1 - Y axis to drive the robot. This is done by "mixing" the axes and creating a separate left and right drive command, drive_R and drive_L.

The One Joystick Drive code is shown below:

```
'   This code mixes the Y and X axis on Port 1 to allow one joystick drive.
'   Joystick forward  = Robot forward
'   Joystick backward = Robot forward
'   Joystick right    = Robot rotates right
'   Joystick left     = Robot rotates left

if rc_sw5 = 0 then skipOption5
    temp = drive_R
    drive_R = (((2000 + temp  - drive_L + 127) Min 2000 Max 2254) - 2000)
    drive_L = (((2000 + temp  + drive_L - 127) Min 2000 Max 2254) - 2000)
    drive_R = 254 - drive_R
skipOption5:
```

6. SW6 Option – Reverse Drive Direction

The reverse drive direction option can be invoked by installing a 2-pin jumper, connecting both pins of SW6 on the Motherboard. The software reverses the motor direction for PWM 1, 2, 3, and 4. This is most commonly used to change the forward driving direction of a robot.

The Reverse Drive Direction code is shown below:

```
if rc_sw6 = 0 then skipOption6
    temp = Drive_R                          'save right before it is changed below
    drive_R = drive_L                       'set right to left
    drive_L = temp                          'set left to the saved right
skipOption6:
```

7. SW7 Option – Speed Sensitivity Adjustment

The speed sensitivity adjustment option can be invoked by installing a 2-pin jumper, connecting both pins of SW7 on the Motherboard. The software uses the Port 1 Wheel to set the maximum speed of the motors. The Wheel in the full forward position corresponds to the motors at maximum speed.

The Speed Sensitivity Adjustment code is shown below:

```
if rc_sw7 = 0 then skipOption7
    p1_wheel = (((p1_wheel*154)/254)+100) max 254    'adjust wheel to 154-254
    if drive_R < 127 then drive_R_reverse:           'is the right drive forward
        drive_R = (drive_R - 127) min 0              'subtract 127 to get the forward value
        drive_R = (drive_R * p1_wheel)/254           'multiply by the wheel percentage
        drive_R = (drive_R + 127) max 254            'add 127 back for proper output
        goto drive_R_done:                           'exit the drive right section
    drive_R_reverse:                                 'the right drive is reverse
        drive_R = (127 - drive_R) min 0              'invert drive-R to get a forward value
        drive_R = (drive_R * p1_wheel)/254           'multiply by the wheel percentage
        drive_R = (127 - drive_R) min 0              'invert drive_R back to normal
    drive_R_done:                                    'drive_R section complete

    if drive_L < 127 then drive_L_reverse:           'is the left drive forward
        drive_L = (drive_L - 127) min 0              'subtract 127 to get the forward value
        drive_L = (drive_L * p1_wheel)/254           'multiply by the wheel percentage
        drive_L = (drive_L + 127) max 254            'add 127 back for proper output
        goto drive_L_done:                           'exit the left right section
        drive_L_reverse:                             'the left drive is reverse
        drive_L = (127 - drive_L) min 0              'invert drive-L to get a forward value
        drive_L = (drive_L * p1_wheel)/254           'multiply by the wheel percentage
        drive_L = (127 - drive_L) min 0              'invert drive_R back to normal
    drive_L_done:                                    'drive_L section complete
skipOption7:
```

8. SW8 Option – Limit Switches

The limit switches option can be invoked by installing a 2-pin jumper, connecting both pins of SW8 on the Motherboard. The software stops movement of a motor in a specific direction when a limit switch is pressed. This is commonly used to stop arms at the end of their normal or desired travel. This section of code requires limit switches be connected to the Digital Inputs SW1-SW4. In software, these 4 limit switches are referred to as rc_sw1 through rc_sw4.

The Limit Switch code is shown below:

```
if rc_sw8 = 0 then skipOption8
    if rc_sw1 = 0 then sw1_off
        p3_x = p3_x MAX 127                  'PWM6 wont go CCW if rc_sw1 is ON
    sw1_off:
    if rc_sw2 = 0 then sw2_off
        p3_x = p3_x MIN 127                  'PWM6 wont go CW  if rc_sw2 is ON
    sw2_off:
    if rc_sw3 = 0 then sw3_off
        p3_wheel = p3_wheel MAX 127          'PWM8 wont go CCW if rc_sw3 is ON
    sw3_off:
    if rc_sw4 = 0 then sw4_off
        p3_wheel = p3_wheel MIN 127          'PWM8 wont go CW  if rc_sw4 is ON
    sw4_off:
    relay1_fwd = p1_sw_trig &~ rc_sw1        'RLY1 wont go CW  if rc_sw1 is ON
    relay1_rev = p1_sw_top  &~ rc_sw2        'RLY1 wont go CCW if rc_sw2 is ON
    relay3_fwd = p3_sw_trig &~ rc_sw3        'RLY3 wont go CW  if rc_sw3 is ON
    relay3_rev = p3_sw_top  &~ rc_sw4        'RLY3 wont go CCW if rc_sw4 is ON
skipOption8:
```