IntelliBrain™ 2

Robotics Controller

User Guide

www.ridgesoft.com

IMPORTANT NOTICE

Information in this document is furnished under license and may only be used in accordance with the terms of the license.

RIDGESOFT PRODUCTS ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT.

RIDGESOFT PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS WHERE FAILURE COULD RESULT IN THE LOSS OF LIFE OR HAVE SERIOUS LIFE THREATENING OR ECONOMIC IMPACT.

Regulatory Warning

IntelliBrain 2 robotics controllers, their constituent components, add-on boards and accessories sold by RidgeSoft, LLC are not FCC approved because they are not in finished product form. If you wish to obtain FCC approval, you must first design the controller and/or accessory into a product, and then seek FCC approval of the whole product.

Trademarks

RidgeSoft™, RoboJDE™ and IntelliBrain™ are trademarks of RidgeSoft, LLC.

Java™ and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All other brand or product names are trademarks of their respective owners.

Copyright

RidgeSoft, LLC
PO Box 482
Pleasanton, CA 94566
www.ridgesoft.com

Document Revision 1.0

Table of Contents

Introduction

Your IntelliBrain™ 2 robotics controller is a circuit board designed to enable you to build a robot and program it using the Java™ programming language.

Your IntelliBrain 2 robotics controller provides the following features, which are also shown in Figure 1:

- Java programmable
- 2 RS-232 serial ports
- 7 analog input ports
- 13 digital input/output ports
- 38 kHz infrared universal remote control receiver
- 38 kHz infrared transmitter
- 5 I^2C ports
- 8 servo ports (3 signal only)
- 2 pulse width modulated DC motor ports
- 16 x 2 character LCD display
- thumbwheel
- buzzer
- 2 push buttons
- 6 program controlled LEDs
- battery pack connector
- wall brick power connector
- power switch
- 4 screw mounting holes
- 4 Lego® brick grid compatible mounting holes
- 4.0 x 3.05 inch circuit board

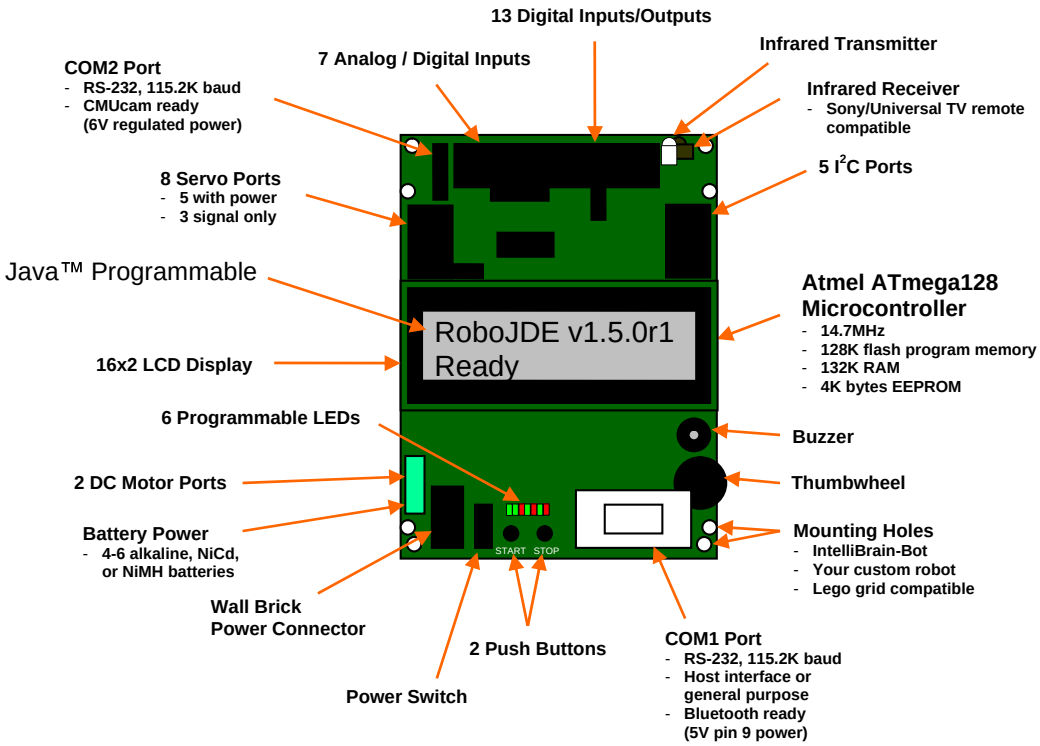

Figure 1 - IntelliBrain 2 Robotics Controller Features

Getting Started

The following sections describe the steps you will need to complete in order to get started with your IntelliBrain™ 2 robotics controller.

Connecting Power

Your IntelliBrain 2 robotics controller supports a variety power options, described in later sections. Use the following procedure to power your IntelliBrain 2 robotics controller with 4 AA batteries.

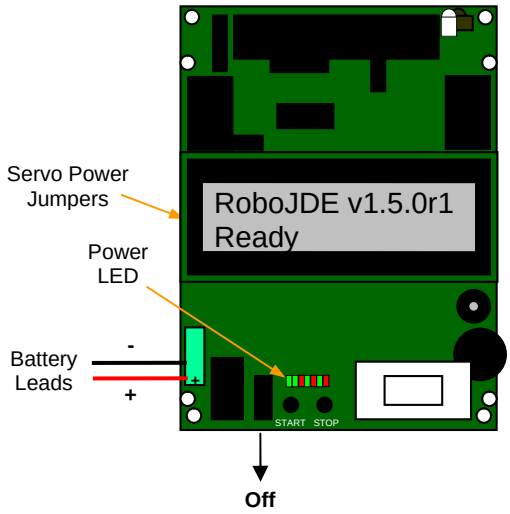

Figure 2 – Making Battery Connections

1. Check that the power switch is in the off position, as shown in Figure 2.

2. Attach the red lead from the battery holder to the positive terminal on the main board battery terminal block and the black lead from the battery holder to the negative terminal, as shown in Figure 2.

3. Insert four fresh AA NiMH (recommended), NiCd or alkaline batteries in the battery holder in the orientation indicated on the battery holder.

4. Switch the power switch on. The green power LED will illuminate and the RoboJDE version message will be displayed on the LCD screen, as shown in Figure 2.

Establishing a Host Connection

Your IntelliBrain 2 robotics controller connects to your host computer via a straight through DB9 female to DB9 male extension cable or via a Bluetooth serial adapter. Use the following procedure to connect your IntelliBrain 2 robotics controller to your host computer:

1. Attach the male end of the cable to the DB9 connector on your IntelliBrain 2 robotics controller.

2. Attach the female end of the cable to your host computer. If your host computer does not have a serial port, you will need to use a USB to serial adapter cable, such as the FTDI US232B USB to serial adapter cable, to add a serial port to your host computer.

3. Install the RoboJDE development environment on your host computer.

4. Configure the RoboJDE development environment's settings using the Tools->Settings dialog to: 1) select the controller type, "IntelliBrain", 2) set the COM port to which you have attached your IntelliBrain 2 robotics controller, and 3) set the baud rate to 115,200. For more information see the *RoboJDE User Guide*.

   Note: The default baud rate for your IntelliBrain 2 robotics controller is 115.2K baud. Your IntelliBrain 2 robotics controller displays the currently configured baud rate when in bootstrap mode (see the next section).

5. Following the instructions in the RoboJDE User Guide, build, load and run the "HelloWorld" example program.

Changing the Host Port Baud Rate

1. Start your IntelliBrain 2 robotics controller in bootstrap mode by holding the STOP button down while switching power on or by pressing the STOP button immediately after switching power on.

2. Observe the bootstrap loader's welcome message, "IntelliBrain 2", followed by the current baud rate setting. The factory default baud rate setting is 115.2K (115,200) baud.

   Note: If you experience problems with communications between your host computer and your IntelliBrain 2 robotics controller you may need to configure the RoboJDE development environment and your IntelliBrain 2 robotics controller to use a lower baud rate.

3. Press the STOP button repeatedly until the baud rate you would like to use is displayed.

4. Press the START button. Your IntelliBrain 2 robotics controller will store the new baud rate setting so it is retained when you power it off.

5. Press the START button, or switch the power off and back on again, to start the virtual machine.

6. Use the RoboJDE development environment's Tools->Settings menu to configure the same baud rate on the host.

7. Verify that the "VM State" displayed in the RoboJDE Run window is something other than "Not Responding".

Programming Your IntelliBrain 2 Robotics Controller

You will use the RoboJDE development environment to create Java programs, compile them, download them to your IntelliBrain 2 robotics controller and run them.

Using the Application Programming Interface (API) Documentation

The RoboJDE development environment includes full on-line documentation for its Application Programming Interface (API). This API allows your program to interface with your IntelliBrain 2 robotics controller hardware, as well as sensors and effectors attached to it.

To view the API documentation, click on the blue book icon on the tool bar in the RoboJDE development environment. Browse to the documentation for the `IntelliBrain` class for full documentation on the methods that allow your programs to access your IntelliBrain 2 robotics controller's features. The API documentation is also available on the RidgeSoft website and on the *Software and Documentation* CDROM that came with your IntelliBrain 2 robotics controller.

The RoboJDE development environment also includes an API quick reference graphic for your IntelliBrain 2 robotics controller. See the file IntelliBrain2API.pdf in the "docs" folder where you installed the RoboJDE development environment. This document is also available from the RidgeSoft website and on the *Software and Documentation* CDROM that came with your IntelliBrain 2 robotics controller.

This User Guide provides brief programming examples that demonstrate how you can use the IntelliBrain API to access features of your IntelliBrain 2 robotics controller. These programming examples are intended to provide you with basic information to get you started programming your IntelliBrain 2 robotics controller. Consult the API documentation for more in depth information about each feature.

Using the RoboJDE Development Environment

Use the *RoboJDE User Guide* to learn how to use the RoboJDE development environment.

Understanding Error Messages

All run-time errors, except internal virtual machine errors, are reported to your Java program as exceptions. Consult the *RoboJDE User Guide* for more information on exceptions.

If the RoboJDE virtual machine determines it has encountered an internal error it will illuminate the red fault LED, display "Internal Error" to the LCD screen and enter breakpoint mode.

When the RoboJDE virtual machine is in breakpoint mode, the LCD screen displays "Breakpoint" on the first line followed by "ip: <address>", where <address> is the hexadecimal value of the virtual machine's instruction pointer at the time of the breakpoint. This indicates the point of execution in your program where the breakpoint occurred. You can toggle between the breakpoint display screen and your program's output by pressing the STOP button. If the breakpoint is the result of an internal error, the program output screen will display the internal error message.

Internal errors should rarely or never occur. Such errors are the result of errors in the RoboJDE virtual machine. You can't fix internal errors by changing your program, though you may be able to avoid them by changing your program. If your program encounters an internal error, refer to the RidgeSoft web site for support information or send email to support@ridgesoft.com to report the error.

Mounting Your IntelliBrain 2 Robotics Controller on a Robot Chassis

Your IntelliBrain 2 robotics controller has four mounting holes located on the corners of the circuit board which allow you to fasten it to your robot chassis with screws. There are four additional mounting holes intended for mounting your IntelliBrain 2 robotics controller on a chassis constructed from Lego® bricks. These are the mounting holes without screw pads around them.

Mounting on an IntelliBrain-Bot Chassis

If you purchased an IntelliBrain-Bot kit, follow the directions in the *IntelliBrain-Bot Assembly Guide* to mount your IntelliBrain 2 robotics controller on your IntelliBrain-Bot chassis.

Mounting on a Lego Chassis

The Lego mounting holes in your IntelliBrain 2 robotics controller are spaced to fit the Lego brick grid. You can create mounting posts using four antenna whip pieces, shown in Figure 3. The wide base portion of the antenna fits snuggly in your IntelliBrain 2 robotics controller mounting holes, allowing your IntelliBrain 2 robotics controller to be mounted on a chassis constructed from Lego bricks. You can make an antenna into a short mounting post by cutting it at the point above the base where it narrows.

Slide an antenna through each hole from the bottom of the main board and push it snuggly into place. This will give the main board four Lego connectors that will align with other Lego components.

Figure 3 - Lego Antenna Whip 8H

These and many other Lego components can be found for sale on www.bricklink.com.

Mounting on a Custom Robot Chassis

You can mount your IntelliBrain 2 robotics controller on almost any chassis by drilling four 1/8 inch (3 mm) diameter mounting holes in the chassis in a 2.75x3.70 inch (70x94 mm) rectangular pattern. Fasten your IntelliBrain 2 robotics controller to your robot chassis using four 1" standoffs and eight screws (see Table 1).

Table 1 – Mounting Hardware

Part	Part Number	Source
1" round aluminum standoff, 4-40	3482K-ND	www.digikey.com
Pan head Phillips screw, 1/4", 4-40	H342-ND	www.digikey.com
Pan head Phillips screw, 3/8", 4-40	H781-ND	www.digikey.com

Connecting Sensors, Effectors and other Devices

Your IntelliBrain 2 robotics controller has 37 program accessible ports which allow you to connect a wide variety of sensors, effectors and other devices.

Port headers are arranged side-by-side along three edges of your IntelliBrain 2 robotics controller's circuit board, as shown in Figure 4. The ground pin for each port is always the pin nearest the edge of the board. The power pin is always the second pin from the edge of the board, next to the ground pin. The power pin is +5V for all ports except the servo ports (see servos section) and the COM2 port. The power pin on the COM2 port is +6V to supply power to a CMUcam or CMUcam2. The next pin–third from the board edge–is always a signal pin. On four-pin ports, the fourth pin is a second signal pin. Additional details for each port are provided in later sections.

Each port, except for the motor ports and COM1, has a 3 or 4 pin male header to provide power, ground and signal connections, as shown in Figure 5. You can construct sensor connectors using the parts and tools listed in Table 2.

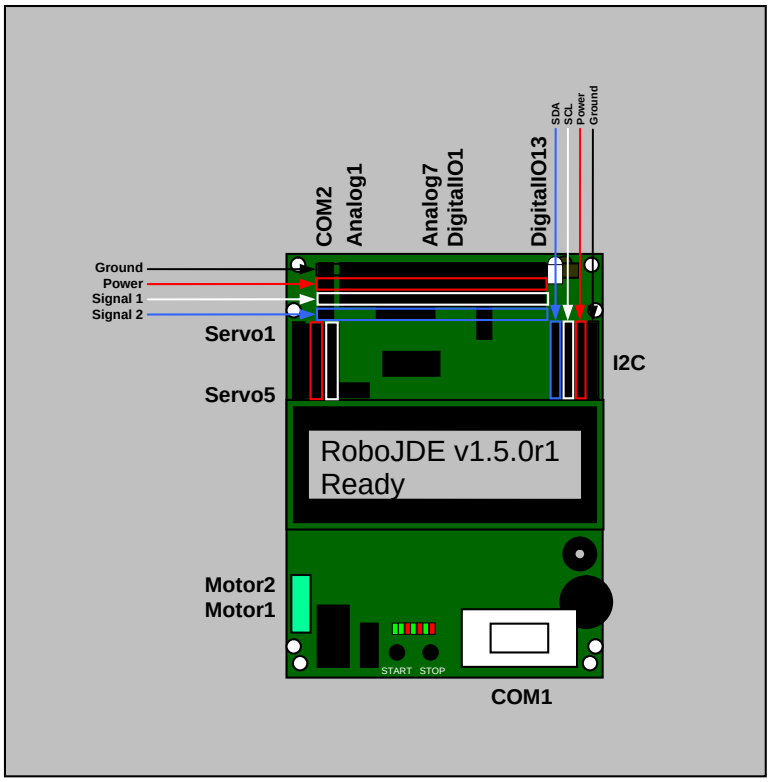

Figure 4 - Ports and Pin Arrangement

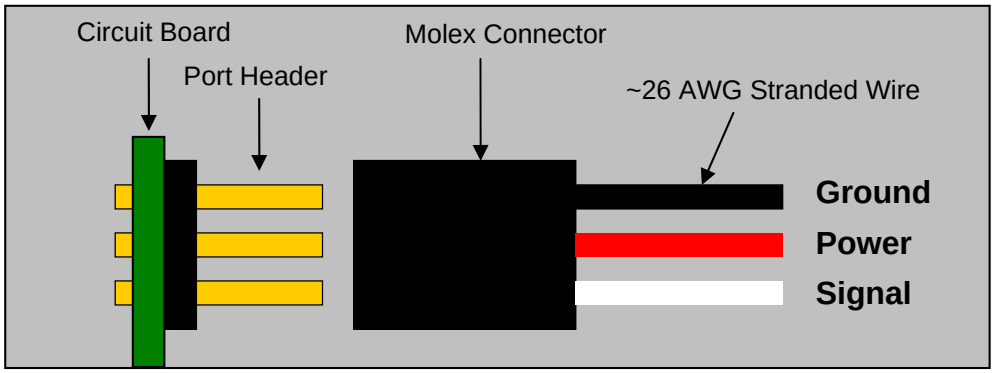

Figure 5 - Typical Sensor Connection

Table 2 – Sensor Connector Parts and Tools

Part	Part Number	Source
Crimp Terminals	WM2555-ND	www.digikey.com
3 Circuit Housing	WM2801-ND	www.digikey.com
4 Circuit Housing	WM2802-ND	www.digikey.com
Universal Crimp Tool	WM9999-ND	www.digikey.com
Insertion Tool	WM9911-ND	www.digikey.com

Connecting Handy Board Sensors

Most sensors designed for the MIT Handy Board controller will work with your IntelliBrain 2 robotics controller but require an adapter cable as shown in Figure 6. The adapter cable can be constructed using 26 AWG stranded wire and the parts listed in Table 2.

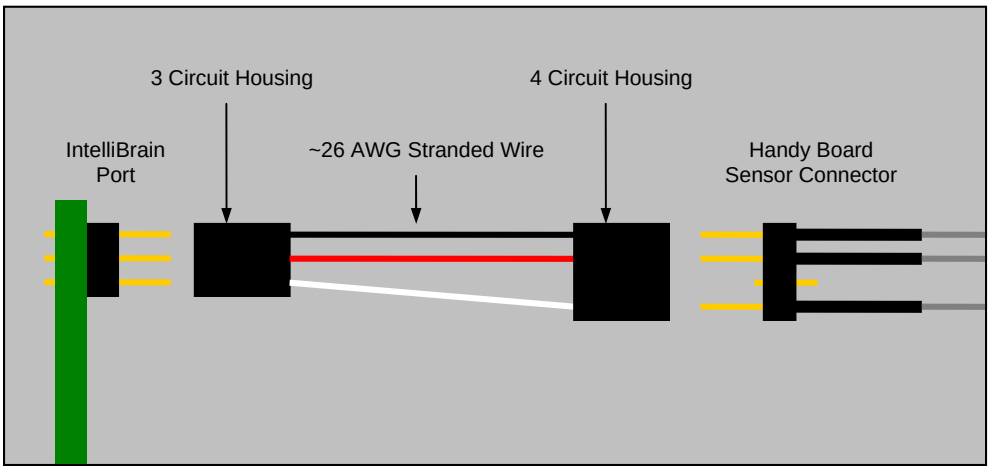

Figure 6 - Handy Board Sensor Adapter Cable

Feature Reference

This chapter will help you learn about the many features of your IntelliBrain™ 2 robotics controller and how to use them.

CPU and Memory

An Atmel ATmega128 microcontroller with 128K bytes of external RAM and a 14.7 MHz clock forms the core of your IntelliBrain 2 robotics controller.

Flash Memory

The ATmega128 includes 128K bytes of on-chip flash memory. Your IntelliBrain 2 robotics controller uses this memory for its bootstrap loader, the RoboJDE virtual machine and for 60K bytes of persistent storage for one Java™ program.

The flash memory has a limited life (minimum 10,000 writes according to ATmega128 specifications). Minimizing downloads to the flash memory by using RAM while developing, testing and debugging your Java programs will extend the life of your IntelliBrain 2 robotics controller.

RAM

The ATmega128 has 4K bytes of on-chip Random Access Memory (RAM). This memory is used by the bootstrap loader and the RoboJDE virtual machine.

Another 128K bytes of external RAM on your IntelliBrain 2 robotics controller provides space for your Java program data and for temporarily loading your Java program while developing and testing. You can load a Java program of up to 30K bytes into RAM, but it will be lost when power is switched off or the batteries become drained. Using RAM for program storage while developing your program reduces the wear on the flash memory and results in faster program downloading. However, RAM occupied by your program is not available for storing data such as Java objects and stacks. Downloading your program to flash allows you to load larger programs and have the maximum amount of memory available for objects and stacks.

When you have a program loaded in RAM, it will be the program that runs when you press the START button. If you want to run the program that is in flash memory when there is also a program stored in RAM, switch your IntelliBrain 2 robotics controller off then back on to erase the program in RAM. This will allow you to run the program stored in flash memory.

The RoboJDE virtual machine manages the allocation of memory as the program creates new objects. The RoboJDE virtual machine recovers memory by garbage collecting objects as they become un-referenced. The virtual machine's garbage collector uses a reference counting garbage collector rather than a mark-and-sweep algorithm to provide predictable real-time performance.

CPU EEPROM

The ATmega128 includes 4K bytes of EEPROM memory with a life of at least 100,000 writes, according to ATmega128 specifications. The first 96 bytes of the EEPROM memory are reserved and hidden from your programs. The remaining 4000 bytes are available to your programs.

Programming Example

```
static final int ADDRESS_OF_SOMEDATA = 0;
      :
EEPROM cpuEEPROM = IntelliBrain.getCpuEEPROM();
      :
cpuEEPROM.write(ADDRESS_OF_SOMEDATA, 1234);
      :
int someData = cpuEEPROM.readInt(ADDRESS_OF_SOMEDATA);
```

Direct Memory Access

Under most circumstances you should access features of your IntelliBrain 2 robotics controller via the Java API rather than attempting to access the microcontroller's registers or memory directly. The RoboJDE development environment does provide direct memory access via methods in the `com.ridgesoft.vm.VM` class; however, you should avoid using these methods. Should you choose to use the direct memory access methods, you must be careful to avoid interfering with the RoboJDE virtual machine. All RAM memory is managed by the virtual machine; therefore, it is not advisable to use the direct memory access methods to access RAM.

Power Connections

Your IntelliBrain 2 robotics controller provides connectors for two DC power sources, allowing you to power the board using a battery pack or a DC power supply (typically a wall brick). Using a DC power supply while developing and debugging your robot on the desktop will preserve the charge in the battery for times when you want to operate your robot untethered.

Connect a battery or battery holder to your IntelliBrain 2 robotics controller using the instructions in the *Connecting Power* section. You may choose to use either a rechargeable or non-rechargeable battery pack. If you use a rechargeable battery pack, you must recharge it with a separate battery charger. RidgeSoft recommends using NiMH rechargeable batteries.

You may connect your IntelliBrain 2 robotics controller to a DC power supply using the power connector next to the power switch (see Figure 1). When you connect a DC power supply, it does not charge the battery, it simply disconnects it.

Servo Power Regulator and Fuse

Your IntelliBrain 2 robotics controller includes a regulator to optionally regulate the voltage supplied to the servo ports to 5 volts. This allows you to use a higher voltage battery without damaging your servos. If you choose not to regulate servo power, the servos will be powered directly from the battery or DC power supply. Figure 7 shows how to set the servo regulator jumpers, which are located on the left side of the circuit board, under the LCD display.

A 1.5A auto-reset fuse protects the servo power circuit. The circuit is fused whether or not you regulate servo power.

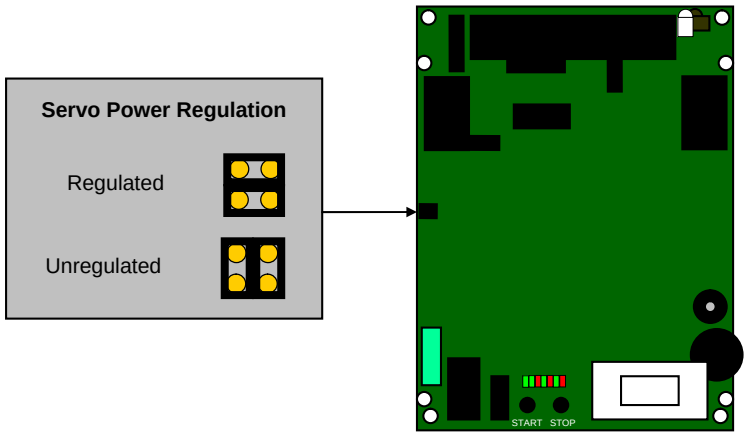

Figure 7 – Servo Power Regulator Jumper Settings

Choosing a Battery and/or a Power Supply

Your IntelliBrain 2 robotics controller includes a two stage voltage regulator designed to fully and efficiently use the charge in standard NiMH, NiCd, or alkaline battery cells. Depending on the sensor and effector load, the board can operate on two to six NiMH, NiCd or alkaline cells.

A battery consisting of four to six NiMH, NiCd or alkaline cells is suitable for most small robotics projects. RidgeSoft recommends using NiMH cells because they are inexpensive, rechargeable and they deliver more current than most other batteries.

If you use a battery or power supply that outputs more than 6 volts you should configure the servo power jumpers to regulate servo power. Failing to do so may result in damage to your servos.

Table 3 provides information to help you select a battery arrangement and a power supply to suit your robot.

Table 3 – Battery and Power Supply Recommendations

Power Source	Notes
Battery	▪ NiMH, NiCd or alkaline cells may be used. RidgeSoft recommends NiMH. ▪ 2 cells minimum when not using servos ▪ 4 cells minimum if using servos ▪ up to 6 cells maximum (maximum 9V) to support higher voltage motors and/or high power demand ▪ servo power must be regulated if greater than 6V
Power Supply	▪ 5V-9V DC output. RidgeSoft recommends using a regulated 5V-6V DC power supply. ▪ regulate servo power if power supply is unregulated or greater than 6V ▪ female, 2.1 mm x 5.5 mm, center positive plug

Main Power Supply Loading Considerations

There are many factors that affect the number of devices you can power from your IntelliBrain 2 robotics controller. One primary consideration is the total load the main power regulator can support. The main power regulator supplies regulated power to power the controller itself and most sensors and effectors. Servos and motors are not powered via the main power regulator.

The amount of current the main regulator can supply depends on the input voltage and the thermal limits of the regulator. If the power supply is overloaded it will either enter thermal shutdown or it will be unable to maintain the +5 volt output. In either case, the controller will reset.

If your IntelliBrain 2 robotics controller resets unexpectedly, it is an indication you need to use a larger battery or you have overloaded the main power regulator with sensors or other devices that draw too much power. If resets occur only when a servo or motor turns on, it is most likely that your battery is not large enough.

Table 4 provides approximate values of the maximum current output available from the main regulator after deducting the current consumed by the on-board circuitry.

Table 4 – Main Power Regulator Output

Main Battery	Approximate maximum total current available to sensors and effectors
4 alkaline cells	450 mA
4 NiMH or NiCd cells	600 mA
6 NiMH or NiCd cells	800 mA

COM1 (Host Port)

Your IntelliBrain 2 robotics controller provides a high speed RS-232 serial connection to your host computer for software download and debugging. This port can operate at baud rates between 115.2K baud and 9600 baud. The highest baud rate you can use may be limited by the host computer and the quality or length of the cable you use. See the *Establishing a Host Connection* section for more information on connecting IntelliBrain 2 robotics controller to your host computer.

A Bluetooth serial "cable", such as those manufactured by AirCable (www.aircable.net), may be used instead of a real cable to allow untethered interfacing with your host computer.

The COM1 port uses a standard female DB9 connector. Regulated power (+5V) is supplied to pin 9 of this connector, which is convenient to power devices such as an AirCable Bluetooth serial adapter.

Programming Example

```
// Print some text to the RoboJDE run window
System.out.println("some text");
```

By default `System.out` goes to both the LCD and the Run window. You can direct `System.out` to just the Run window as follows:

```
System.out = new PrintStream(VM.getDebugOutputStream());
```

Using COM1 to Interface with Other Devices

Your program may also take control of the COM1 port to interface with other devices, such as a GPS receiver.

Programming Example

```
SerialPort com1 = IntelliBrain.getCom1();
com1.setSerialPortParams(115200,
                         SerialPort.DATABITS_8,
                         SerialPort.STOPBITS_1,
                         SerialPort.PARITY_NONE);
InputStream inputStream = com1.getInputStream();
OutputStream outputStream = com1.getOutputStream();
   :
int inData = inputStream.read();    // receive data
   :
outputStream.write(outData);        // transmit data
```

LCD Display

The LCD display allows your program to output text messages to its two-line by sixteen character LCD screen.

Programming Example

```
// Address the display lines separately
Display display = IntelliBrain.getLcdDisplay();
display.print(0, "first line");
display.print(1, "second line");

// Write scrolling text to the LCD screen
System.out.println("This is more than 16 characters");
```

LEDs

Your IntelliBrain 2 robotics controller has seven visible LEDs and one infrared LED. The visible LEDs are arranged in a row above the START and STOP button (see Figure 1). The infrared LED is located near the front right corner the circuit board.

Power LED

The left most LED is a green power LED. It is illuminated when power is on.

Status and Fault LEDs

The second LED from the left is a green "status" LED. You may use this LED as you choose in your program. This LED always illuminates while the START button is pressed. The third LED is a red "fault" LED. The virtual machine illuminates this LED if it encounters a problem. You may also use this LED in your program, but you can't prevent it from being illuminated if the virtual machine detects a fault.

Programming Example

```
LED statusLED = IntelliBrain.getStatusLed();
LED faultLED = IntelliBrain.getFaultLed();
      :
statusLED.toggle();
      :
faultLED.on();
```

Motor / User LEDs

The rightmost four LEDs are two green-red pairs which, by default, indicate the state of the two motor ports. The pair on the left indicates the state of motor port 1. The pair on the right indicates the state of motor port 2. The motor port states are described in Table 5.

Table 5 - Motor Port States

Green LED	Red LED	Motor Port State
Off	Off	Off, coast
On	Off	On, forward
Off	On	On, reverse
On	On	On, brake

Your programs can take full control of any of the motor/user LEDs to use them for other purposes.

```
IntelliBrain.getUserLed(int ledNumber)
```

These LEDs (the four rightmost LEDs) are numbered 1 to 4 from left to right. The odd numbered LEDs are green and the even numbered LEDs are red.

Programming Example

```
LED greenLED = IntelliBrain.getUserLed(1);
greenLED.on();
```

Infrared Transmitter LED

The infrared transmitter LED is modulated at 38 kHz to be compatible with the infrared receiver. You may use this LED in conjunction with the infrared receiver to sense objects in front of your robot or to communicate between robots at low bandwidth.

If you use the infrared LED in conjunction with the infrared receiver for object detection, you will need to restrict the infrared light beam it transmits by placing a short piece (~1 inch) of heat shrink tubing over it and wrapping electrical tape around behind it. Be sure to slide the tubing all the way over the base of the LED and taped to fully block any infrared light leaking out. Even the slightest hole will allow enough infrared light out to be detected by the receiver.

Programming Example

```
LED irLED = IntelliBrain.getIrLed();
irLED.on();
      :
irLED.off();
```

Buzzer

You may use the buzzer to provide user feedback, such as clicking when a button is pressed, beeping when your robot detects certain events, or playing tunes to entertain observers.

Programming Example

```
Speaker buzzer = IntelliBrain.getBuzzer();
      :
buzzer.beep();
      :
buzzer.click();
      :
// play C note for one quarter second
buzzer.play(262, 250);
```

Push Buttons

The START and STOP buttons start and stop the execution of your program. These buttons are also used to interact with the bootstrap loader (see the *Bootstrap Loader* section).

Once your program is executing, you can use the START button for any purpose you choose. You can also gain control of the STOP button as follows:

```
IntelliBrain.setTerminateOnStop(false);
```

Programming Example

```
PushButton startButton = IntelliBrain.getStartButton();
      :
if (startButton.isPressed()) {
      // wait for the button to be released
      startButton.waitReleased();

      // do something based on START being pressed
      :
}
```

Thumbwheel

The thumbwheel provides an analog user input. You may use it for a variety of functions such as calibration, scrolling through a list of sensor readings or controlling the speed of a motor. Depending on its position, the thumbwheel will return a value between 0 and 1023 when your program samples it.

Programming Example

```
AnalogInput thumbwheel = IntelliBrain.getThumbWheel();
      :
// generate 0 – 9 index based on thumbwheel position
int index = (thumbwheel.sample() + 57) / 114;
```

Servo Ports

Your IntelliBrain 2 robotics controller can control up to eight servos. Five of the servo ports are arranged in a row along the left edge of the circuit board (see Figure 4). These ports provide power and ground connections in addition to the servo control signal arrange to facilitate direct connection to standard servos. The remaining three ports provide only the control signal (no power or ground) and are the fourth, fifth and sixth pins to the right of servo port 5.

You may use the servo ports to control positioning servos or continuous rotation servos. The ports support servos that are controlled using a 1 to 2 millisecond pulse width modulated signal (1.5ms center position).

Attach your servos to your IntelliBrain 2 robotics controller by plugging them into the servo port as shown in Figure 5. Be sure to orient the connector with the black (ground) wire on the pin nearest the left edge of the board and the signal wire to the third pin from the edge.

The servo ports are powered directly from the battery or DC power. If you use a battery or power supply that is greater than 6V you should enable the servo power regulator. See the *Servo Power Regulator and Fuse* section for more information.

The servo ports are fused through a single 1.5 amp auto-reset fuse.

Programming Example

```
// Positioning servo
Servo servo = IntelliBrain.getServo(1);
      :
servo.setPosition(75);

// Continuous rotation servo
Motor motor = new ContinuousRotationServo(
                        IntelliBrain.getServo(2),
                        false);
      :
motor.setPower(Motor.MAX_FORWARD);
```

Motor Ports

Your IntelliBrain 2 robotics controller has two pulse width modulated DC motor ports capable of supplying up to 1 amp per port at up to 9 volts. The motor ports provide braking as well as variable forward and reverse power.

Each motor port consists of two adjacent terminals on the motor terminal block. The motor ports are arranged on the terminal block as shown in Figure 4. Each port consists of two terminals. The two terminals in the middle of the terminal block are motor port 1. The two terminals nearest the front edge of the circuit board are motor port 2.

Connect your motors by inserting each motor's leads into the port's terminal connectors then tighten the screws. If you find a motor spins in the opposite direction of what your software intends, swap the positions of its leads in the terminal block.

By default, the four Motor / User LEDs show the state of the motor ports (see the *Motor / User LEDs* section).

Programming Example

```
Motor motor = IntelliBrain.getMotor(1);
      :
```

```
motor.setPower(Motor.MAX_FORWARD);
      :
motor.brake();
      :
motor.setPower(Motor.STOP);
```

COM2 (CMUcam Port)

The COM2 port (see Figure 4) is an RS232 serial port. (Note: The COM2 port uses RS232 signal levels, not TTL signal levels). In addition to transmit, receive and ground pins, the port also provides a regulated +6V power supply to pin 2 with a 1 amp auto-reset fuse to power the CMUcam or CMUcam2.

In addition to the CMUcam, the COM2 port can be used to interface with other devices that provide an RS232 interface, such as a Global Positioning System (GPS) receiver. Figure 8 shows how to construct an adapter cable to connect to a device that is designed to connect to a standard 9 pin COM port on a PC. Table 6 lists the parts and tool required to attach the 4 pin connector to the cable shown in Figure 8. In addition, you will need to solder the wires to the DB9 connector.

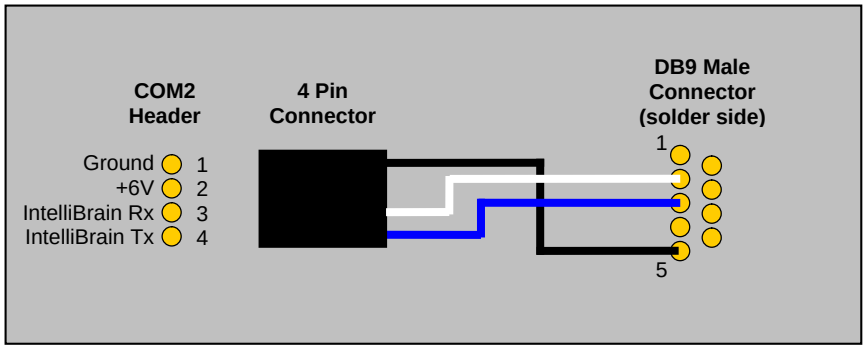

Figure 8 – COM2 DB9 Adapter Cable

Table 6 – DB9 Adapter Cable Parts and Tools

Qty	Description	Part Number	Source
4	Crimp Terminals	WM2555-ND	www.digikey.com
1	4 Circuit Housing	WM2802-ND	www.digikey.com
1	DB9 Male Connector	209M-ND	www.digikey.com
	26 AWG stranded wire	A3049x-100-ND	www.digikey.com
1	Universal Crimp Tool	WM9999-ND	www.digikey.com

Programming Example

```
SerialPort com2 = IntelliBrain.getCom2();
com2.setSerialPortParams(115200,
                         SerialPort.DATABITS_8,
                         SerialPort.STOPBITS_1,
                         SerialPort.PARITY_NONE);
```

```
InputStream inputStream = com2.getInputStream();
OutputStream outputStream = com2.getOutputStream();
   :
int inData = inputStream.read();    // receive data
   :
outputStream.write(outData);        // transmit data
```

Analog / Digital Input Ports

Seven ports on your IntelliBrain 2 robotics controller provide for Analog-to-Digital conversion (see Figure 4). All seven ports may also be used as digital inputs. All seven ports have a software controlled pull-up resistor built into the ATmega128 chip. When you sample an analog port it will return a value between 0 and 1023, which is proportional on the voltage (0–5 volts) on the signal pin (pin 3).

Analog Ports 1 - 3

Analog ports 1–3 include a .1 uF capacitor on the signal line for noise filtering. This is particularly useful when interfacing with Sharp infrared range sensors such as the GP2D12. Sensors should be connected to these ports as shown in Figure 5.

Analog Ports 4 - 7

Analog ports 4–7 have an additional pin and optional electronics on the signal pin (see Figure 9) to support interfacing to infrared emitter-detector sensors, such as the Fairchild QRB1134. The fourth pin connects to ground through a 220 ohm resistor to provide a current limited power circuit for the infrared emitter.

Figure 9 illustrates the additional electronics and shows how to connect QRB1134 sensors to these ports. The jumper block for these ports is located between the ports and the LCD display. The jumpers are arranged in the same order as the ports. Insert the corresponding jumper if you connect an infrared photo-reflector sensor to one of these ports.

You can connect other sensors to pins 1 through 3 of these ports, leaving pin 4 unconnected, just as you would with the other analog ports; however, you should remove the corresponding jumper.

Programming Example

```
AnalogInput lineSensor = IntelliBrain.getAnalogInput(4);
     :
int lineSensorReading = lineSensor.sample();
```

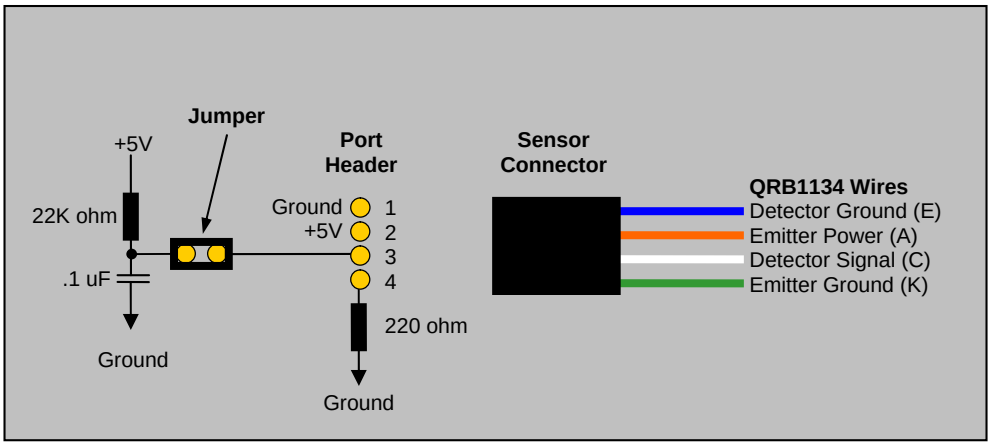

Figure 9 - Connecting QRB1134 Line Sensors to Analog Ports 4 - 7

Digital Input / Output Ports

There are thirteen digital input / output (IO) ports on your IntelliBrain 2 robotics controller (see Figure 4).

Each port is software configurable as an input or an output.

All ports have a software selectable pull-up resistor in the ATmega128 chip when configured as an input.

You can use digital IO ports 3 through 6 for measuring input pulse durations. You will find this useful for interfacing with sensors such as the Parallax Ping)))™ or Devantech SRF04 sonar range finders which use pulse duration modulation to indicate the measured value.

Sensors and effectors should be attached to these ports as shown in Figure 5.

Programming Example

```
// A digital input
IntelliBrainDigitalIO digitalIn = IntelliBrain.getDigitalIO(1);
digitalIn.setDirection(false);
digitalIn.setPullUp(true);
      :
if (digitalIn.isSet()) {
      // input is high
}

// A digital output
IntelliBrainDigitalIO digitalOut = IntelliBrain.getDigitalIO(2);
digitalOut.setDirection(true);
      :
// set the output high
digitalOut.set()
      :
// pull the output low
```

```
digitalOut.clear();
```

I²C Ports

Your IntelliBrain 2 robotics controller has a single I²C bus with 5 headers which allow you to interface it to a wide variety of I²C slave devices. Some popular robotics sensors and effectors that you may want to interface your IntelliBrain 2 robotics controller to are:

- Devantech SRF08 sonar range finder
- Devantech CMPS03 magnetic compass
- Devantech SP03 text to speech synthesizer
- Devantech MD03 motor driver
- Devantech MD22 dual motor driver

Your IntelliBrain 2 robotics controller is an I²C master and the sensors and effectors you attach are I²C slaves. Each I²C slave must have a unique device address which is used to communicate to the device. Your IntelliBrain 2 robotics controller treats the device address as 8 bits consisting of 7 address bits followed by a zeroed bit as the least significant bit. The least significant bit is used as a read-write indicator when your IntelliBrain 2 robotics controller communicates with slave devices. The RoboJDE virtual machine manages the read write bit so you only need to be concerned with the value of the 7 address bits when using the API. Consult the technical documentation for the I²C devices you use for more information on their I²C addresses.

Figure 10 shows how to connect your I²C devices to your IntelliBrain 2 robotics controller headers.

Programming Examples

```
// Devantech SP03 example
DevantechSP03 sp03 =
            new DevantechSP03(IntelliBrain.getI2CMaster());
sp03.speak(0, 5, 10, "Hello");


// Direct I2C device access example
I2CMaster i2cMaster = IntelliBrain.getI2CMaster();
      :
// read the byte in the device at "address" by
// writing the 2 byte address and reading back 1
// byte into resultBuffer array
byte[] writeBuffer = new byte[] {
                              (byte)(address >> 8),
                              (byte)address
                              };
byte[] resultBuffer = new byte[1];
i2cMaster.transfer(deviceAddress, writeBuffer, resultBuffer);
```

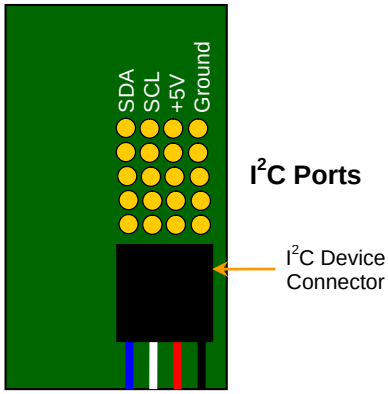

Figure 10 - Connecting I²C Devices

Infrared Receiver

The infrared receiver on your IntelliBrain 2 robotics controller is modulated at 38 kHz to be compatible with many television remote controls. The infrared transmitter LED is also modulated at 38 kHz allowing you to use the transmitter and receiver together for object sensing and low bandwidth communication between robots.

Programming Example

```
IrRemote irRemote =
            new SonyIrRemote(IntelliBrain.getIrReceiver());
      :
int data = irRemote.read();
if (data != -1) {
      // data received from IR receiver
      :
}
```

Bootstrap Loader

Bootstrap loader software was programmed into a protected area of flash memory when your IntelliBrain 2 robotics controller was manufactured. The bootstrap loader allows you to configure the host port baud rate, download the RoboJDE virtual machine to flash memory and download your Java programs into flash memory.

You may enter the bootstrap loader by pressing the STOP button while switching power on or by pressing the STOP button immediately after switching power on. The bootstrap loader displays “IntelliBrain 2” on the first line of the LCD screen followed by “Boot: <baudRate>” on the second line. The <baudRate> parameter is the currently configured baud rate (9.6K, 38.4K or 115.2K), used by both the bootstrap loader and the RoboJDE virtual machine to communicate with the host computer. Pressing the START button exits the bootstrap loader and starts the RoboJDE virtual machine.

In addition to setting the host port baud rate (see the *Changing the Host Port Baud* Rate section), the bootstrap loader allows you to revert to the configuration data in the reserved portion of the microcontroller's EEPROM to the factory settings. The bootstrap loader also allows you to view the main board's serial number.

Reverting to Factory Settings

Reverting to the factory settings will set the baud rate to 115.2K and clear all other configuration data in the reserved portion of the EEPROM. (Currently the baud rate is the only configurable parameter stored in EEPROM. If other parameters are added in the future, they will be cleared by this operation.) Use the following procedure to revert to the factory default configuration:

1. Switch the power off and back on.

2. Press the STOP button repeatedly until you see the message "CLEAR CONFIG?" displayed on the LCD screen.

3. Press the START button to return to the factory configuration settings.

Note: You may exit the configuration mode without making changes by pressing the STOP button repeatedly until the initial bootstrap loader message reappears on the LCD screen.

Viewing the Serial Number

You may view the board's serial number, hardware revision and model number using the following procedure:

1. Switch the power off and back on.

2. Press the STOP button repeatedly until you see the serial number displayed.

3. Press the STOP button again to return to exit the configuration mode and display the initial bootstrap loader message.

In-System Programming (ISP) Header

The header on your IntelliBrain 2 robotics controller just above the LCD display near the I^2C ports (see Figure 11) is an In-System Programming port. This port is used when the board is manufactured. You should not attempt to use it.

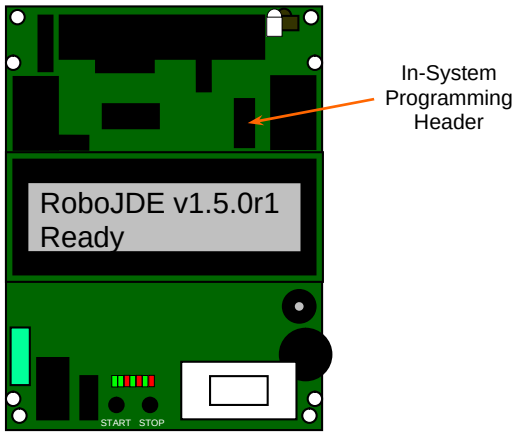

Figure 11 – In-System Programming Header

Appendix A - ATmega128 Port/Pin Usage

The following table documents the usage of each of the I/O pins on the ATmega128 microcontroller.

Table 7 - ATmega128 Pin Usage

Port	Usage
Port A0-7	External RAM & LCD
Port B0	Digital IO7
Port B1	Digital IO8
Port B2	Digital IO9
Port B3	Digital IO10
Port B4	Buzzer
Port B5	Digital IO11
Port B6	Servos
Port B7	Infrared LED
Port C0-7	External RAM
Port D0	I^2C SCL
Port D1	I^2C SDA
Port D2	COM2 receive
Port D3	COM2 transmit
Port D4	Infrared receiver
Port D5	LCD
Port D6	Digital IO12
Port D7	Digital IO13
Port E0	COM1 (host) receive
Port E1	COM1 (host) transmit
Port E2	Digital IO1
Port E3	Digital IO2
Port E4	Digital IO4
Port E5	Digital IO5
Port E6	Digital IO3
Port E7	Digital IO6
Port F0	Thumbwheel
Port F1	Analog 1
Port F2	Analog 2
Port F3	Analog 3
Port F4	Analog 4
Port F5	Analog 5
Port F6	Analog 6
Port F7	Analog 7
Port G0	External RAM
Port G1	External RAM
Port G2	External RAM

Port	Usage
Port G3	START button & green (status) LED
Port G4	STOP button & red (fault) LED