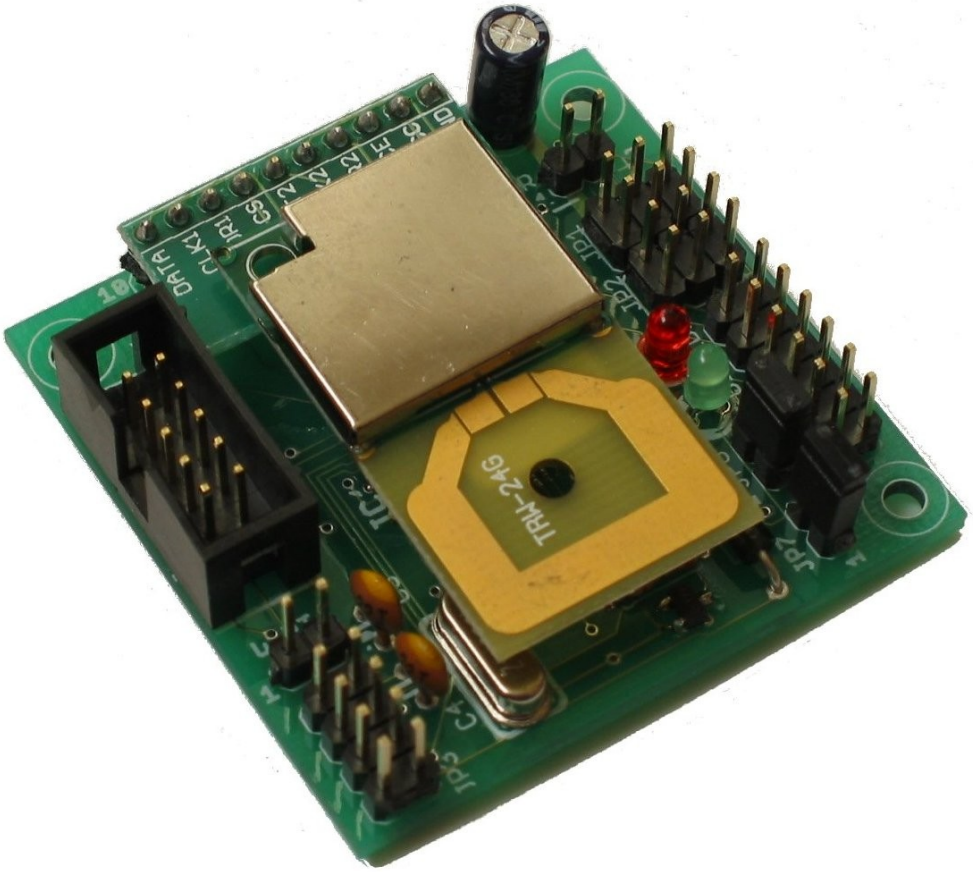
BNX Technologies Inc.

WLB-100 User's Manual



The products of BNX Technologies are designed and assembled with care and tested for functionality and quality before they are shipped. Our products are not rated for use in medical, life support, or systems where failure could result in the loss of life.

BNX Technologies shall not be liable for any direct, indirect, incidental, special, exemplary, or consequential damages of any kind, however incurred, through the use of our products.

WLB-100

The WLB-100 is a small, low-power, embedded microcontroller board with built-in wireless capability. The board is well suited for applications such as robotic control, remote data acquisition, wireless sensor network design, and home automation.

Features:

- Atmel ATMEGA32L 8MHz at 3.3V
  - 8-bit AVR microcontroller
  - 32KB reprogrammable flash memory
  - 2KB SRAM
  - 512 bytes of EEPROM
- Nordic RF2401 2.4-2.53 GHz wireless transceiver
  - 127 software selectable frequencies
  - 200us frequency switching time
  - up to 1Mbs transfer data rate
  - automatic CRC generation/checking
  - dual frequency receiver operation
  - built-in antenna (open space range of up to 125m)
- 4 servo outputs
  - accepts standard Futaba connectors
  - reconfigurable as digital I/O
- headers for UART, SPI, and I^2C
- 8 ADC inputs (each reconfigurable as digital I/O)
- 2 on-board program controlled LEDS (jumper configurable for use as digital I/O pins)
- 5V-8V DC power input

Microcontroller:

The WLB-100 features the Atmel ATMega32 microcontroller. This is an 8-bit microcontroller and is part of the AVR series of microcontrollers from Atmel. The microcontroller has 32KB of reprogrammable, non-volatile flash memory for program storage and

also has 512 bytes of EEPROM for data storage. Complete specifications on the ATMega32 can be obtained from the Atmel (www.atmel.com) and a datasheet has been included on the CD in the Datasheets directory.

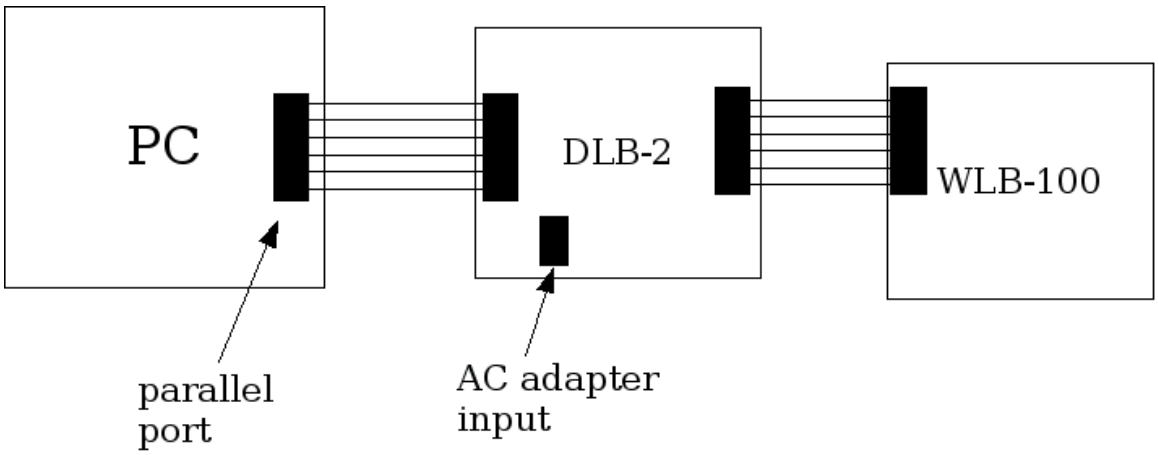
Wireless Transceiver:

The wireless transceiver is the Nordic RF2401. This is a 2.4-2.53 GHz wireless transceiver. A datasheet for the transceiver module is included on the CD. The provide samples routines provided easy function that allow transmitting and receiving data. The transceiver can be configured to use 1 of 127 possible frequencies between 2.4 and 2.53 GHz. Complete details regarding the operation of the wireless transceiver can be obtained through the data sheet provided on the CD. The transceiver unit on the board has a built-in antenna. This antenna cannot be modified.

Connecting the WLB-100 and DLB-2 download board to a computer:

In order to download a program to the WLB-100, the DLB-2 download board must be connected to a PC with a parallel port. Connect the DLB-2 to the parallel port on the computer using the supplied ribbon cable. Connect the DLB-2 to the WLB-100 using the supplied ribbon cable.

A diagram of the connections is shown below:



Note: The parallel port must be in EPP mode. Do not set the parallel port to bi-directional mode.

Powering the DLB-2:

The DLB-2 will accept a DC input voltage in the range of 7V-20V. The polarity of the input plug does not matter. The DC power supplied to the DLB-2 does not perform any charging function for the WLB-100 power source. It only powers the DLB-2.

Powering the WLB-100:

The WLB-100 will accept a DC input voltage in the range of 5-8V. The on-board regulator will power the components with 3.3V.

If servos are connected to the WLB-100, the power for the servos will be directly connected to the positive power terminal. The voltage for the servos does not come from a regulated source.

NOTE: Do not connect servos to the WLB-100 when using a power source with a voltage of greater than 6V. This will damage the servos.

Connections

All I/O connections must be 3.3V compatible. Even though the servo outputs are 3.3V, this is sufficient to control 5V servos.

SERVO1-4

These pins accept standard Futaba hobby servos. There are 3 pins for each servo connector: Ground, Power, and Signal. The Ground and Power pins connect directly to the board power and are not regulated.

Note: Do not plug in a servo backwards. This will damage the servo.

UART

The 2 pins on this connector are for the UART Receive and Transmit. These pins are directly connected to the UART RX and TX pins on the ATMega32 microcontroller.

PROG

The PROG connector is used for programming the board. When the board is not being programmed, the pins on the PROG connector can also be used as general purpose I/O or as an SPI port. Refer to the ATMega32 datasheet for information on how to configure the PROG pins for use in your program.

JP6 & JP7

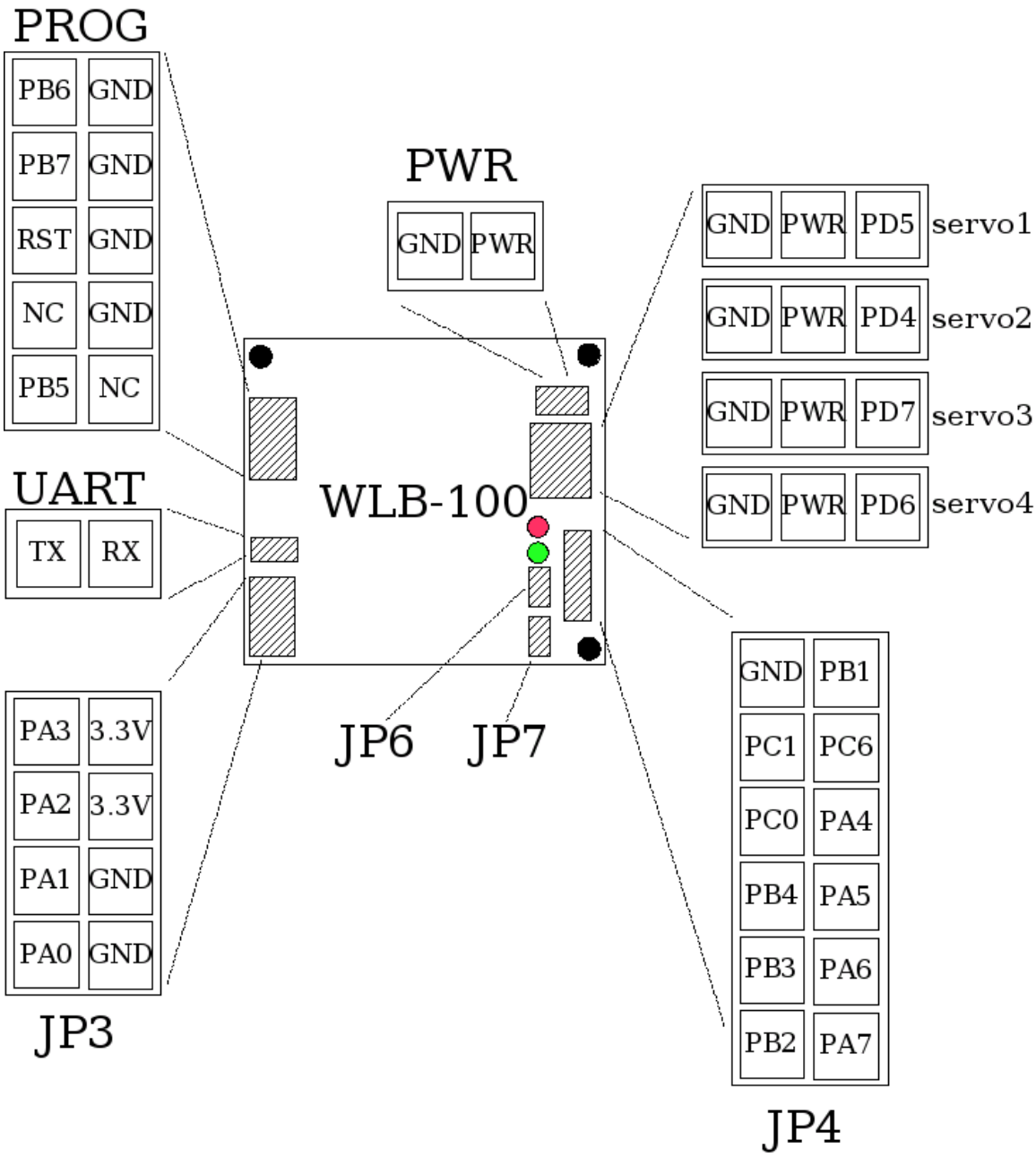
Connectors JP6 and JP7 are used to enable the on-board LEDs. Placing a jumper across JP7 will enable the red LED for software control. Placing a jumper across JP6 will enable the green LED for software control. Removing the jumpers will allow access to pins PB4 and PC6 on connector JP4. JP6 disconnects the green LED from PB4. JP7 disconnects the red LED from PC6.

JP3

This connector provides access to PA0-PA3. These are 4 inputs for the analog-digital converter.

JP4

This connector provides access to PA4-PA7, PB2-4, PC0, PC1, and PC6. PA4-7 are additional inputs for the analog-digital converter.



Software Development

The WLB-100 is most easily programmed in C using open source tools for the AVR family of microcontrollers. AVR-GCC is the GCC compiler for the AVR and is freely available for download and use. This compiler provides excellent performance and is well supported by the AVR development community. The latest version of AVR-GCC can be found at http://sourceforge.net/projects/winavr.

AVR-LIBC is a C library for the AVR microcontroller. The AVR-LIBC library provides excellent access to the functionality of the microcontroller through well-defined C function calls. The latest version of AVR-GCC can be found at http://www.avr-libc.net.

The compiler and library are available under the GNU Public License and are provided in the Software directory of the CD.

A Windows version of the AVR-GCC compiler is included on the CD. The version of WinAVR included on the CD is 200502015. A Linux version of AVR-GCC is also available from http://www.sourceforge.net.

Installing the WinAVR compiler:

Installing the compiler involves running the setup.exe binary and following the necessary steps during installation.

NOTE: After installing WinAVR, it is important that the file install_giveio.bat is executed so that avrdude is able to gain access to the parallel port. Otherwise, avrdude will not be able to reprogram the flash memory of the microcontroller.

Downloading a program to the board:

Avrdude (AVR downloaded/uploader) is the program that allows a compiled program to be downloaded to the flash on the WLB-100. Avrdude is installed during the WinAVR installation. It accepts file in the .hex format. To download the sample_transmit program, first compile the program by typing ‘make’ at the command line in the

sample_transmit directory. If WinAVR is installed correctly, this should produce a file name sample_transmit.hex. Then use the following command at the Windows command line to perform the download:

avrdude -p m32 -c stk200 -U flash:w:sample_transmit.hex

The '-p m32' flag specifies that the processor is an ATMega32 and the '-c stk200' flag specifies the pin out of the download board.

A GUI version of avrdude is also installed and may be used (avrdude-gui). At the time of this writing, the GUI version of avrdude is currently not supported by BNX Technologies.

Sample source code:

Sample routines are provided on the CD which provides servo control, wireless communication, and encryption. The routines are written using the AVR-LIBC library and the source code for the routines is provided. Documentation on how to use the routines in provided in the source code. The source code for these routines is released under the BSD license allowing free use of the code for commercial applications.

Using the sample code:

- sample_transmit - continuously transmits a data packet once a second
- sample_receive – turns on LED1 when a data packet is received
- sample_servo – moves each servo through its full range of motion
- sample_encrypt – encrypts and decrypts data, lights the red LED if the final data is same as the initial data.

To compile the sample programs, change to the appropriate program directory and type 'make'. To download the program, type 'make download.'

Provided C routines:

Using the provided C routines will allow you to quickly develop applications for your WLB-100. Routines have been included which allow easy control of up to 4 standard hobby servos, send and receive data using the wireless transceiver, and encrypt/decrypt data using the ARCFOUR algorithm with a 256-bit key.

Wireless communication routines:

- configure_wireless()
  - This function takes two parameters: an address and a frequency. The address must be between 0 and 255. The frequency must be between 0 and 127.
- change_frequency()
  - This changes the wireless transceiver to use a different frequency.
- set_transmit()
  - This function sets the wireless transceiver to transmit mode.
- set_receive()
  - This function sets the wireless transceiver to receive mode.
- get_data()
  - This function returns a 16-bit data value when a data packet is received
- send_data()
  - This function sends a 16-bit data value to a particular address.

Servo control routines:

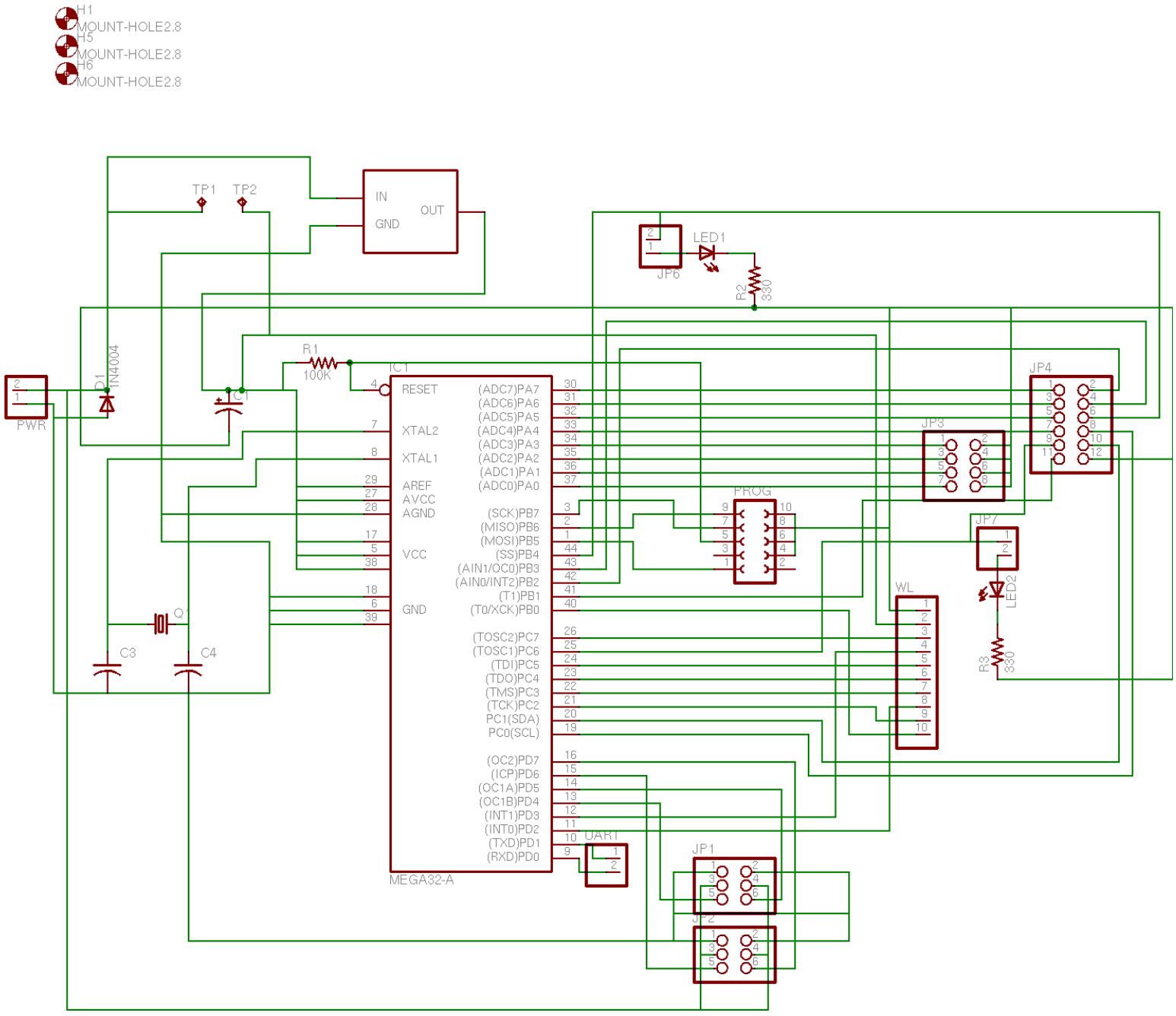
- servo_init()
  - This function initializes the servo routines.
- set_servo_position(u08 number, u08 position)
  - This function sets servo ‘number’ to a 'position', where position is between 0-255.

Encryption routines:

- arcfour_setup()
  - This function will set up the key structure for the ARCFOUR algorithm. The 256-bit key should be in the array key[]. This only needs to be called at the beginning of a program.
- arcfour_crypt()
  - This function will encrypt/decrypt the data in the array data[].

If this function is called on original data, after the function completes, the data in the data array will be encrypted. If data has been encrypted, calling this function on the data will decrypt the data leaving the original data in the data array.

Schematic


H1
MOUNT-HOLE2.8
H5
MOUNT-HOLE2.8
H6
MOUNT-HOLE2.8
TP1
TP2
IN
OUT
GND
LED1
JP6
R2
330
R1
100K
IC1
D1
IN4004
PWR
C1
RESET
XTAL2
XTAL1
AREF
AVCC
AGND
VCC
GND
(ADC7)PA7
(ADC6)PA6
(ADC5)PA5
(ADC4)PA4
(ADC3)PA3
(ADC2)PA2
(ADC1)PA1
(ADC0)PA0
(SCK)PB7
(MISO)PB6
(MOSI)PB5
(SS)PB4
(AIN1/OC0)PB3
(AIN0/INT2)PB2
(T1)PB1
(T0/XCK)PB0
(TOSC2)PC7
(TOSC1)PC6
(TDI)PC5
(TDO)PC4
(TMS)PC3
(TCK)PC2
PC1(SDA)
PC0(SCL)
(OC2)PD7
(ICP)PD6
(OC1A)PD5
(OC1B)PD4
(INT1)PD3
(INT0)PD2
(TXD)PD1
(RXD)PD0
MEGA32-A
Q
C3
C4
UART
JP1
JP2
PROG
JP3
JP4
JP7
LED2
R3
330
WL

Troubleshooting

Problem	Potential Solution
Board does not respond to avrdude check.	The file install_giveio.bat needs to be run to allow avrdude to access the parallel port under Windows.
Program does not download to board.	Check power to DLB-2 and WLB-100. Both need to have power applied during a flash download.
DLB-2 board is connected to a parallel port other than LPT1	The configuration for avrdude can be changed in avrdude.conf. Change the parallel port from LPT1 to the port currently in use.
Servo position cannot be set using set_servo_position.	Make sure that the servo_init function has been invoked before using the set_servo_position functions.