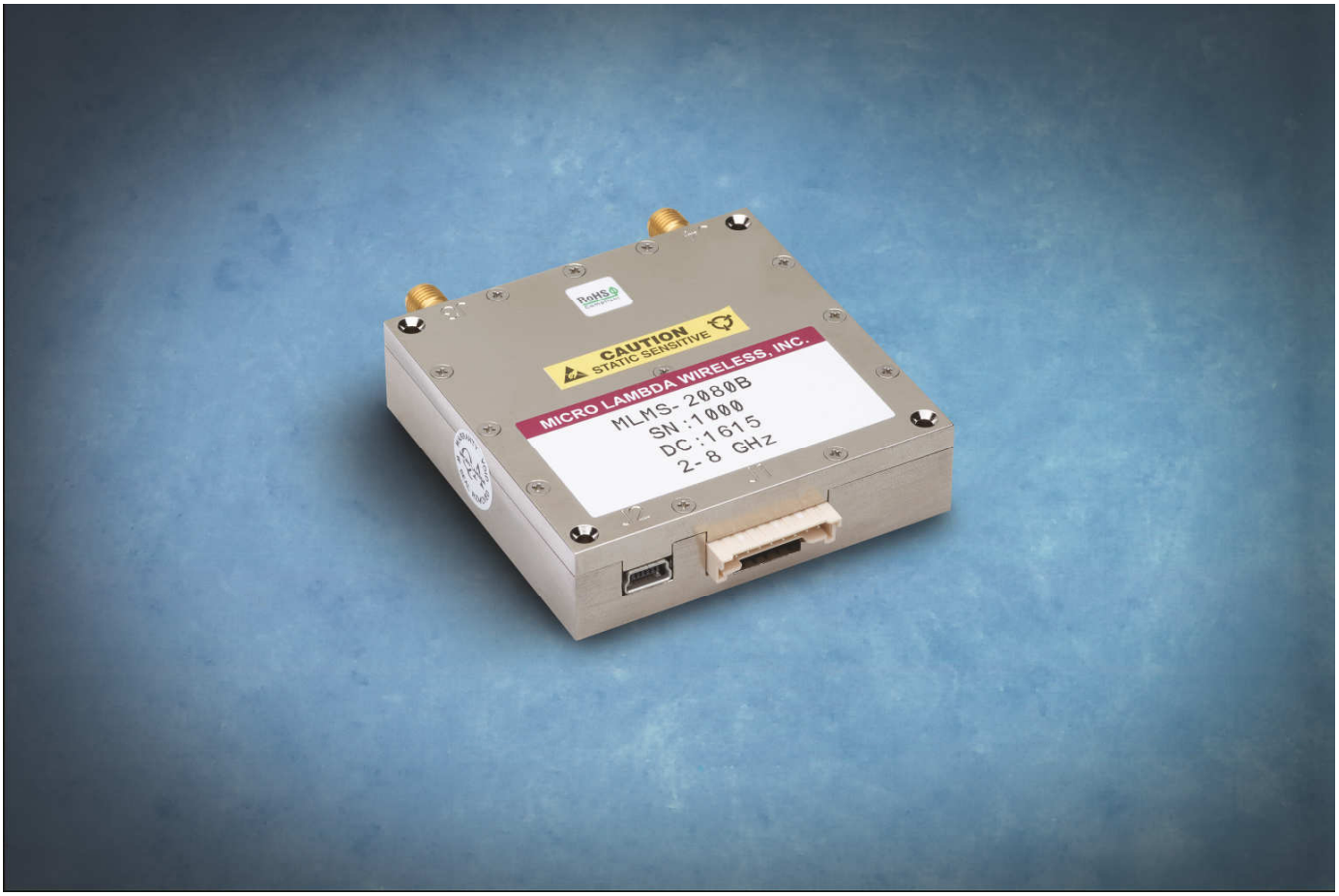
MICRO LAMBDA WIRELESS, INC.

YIG based Products



MLMS Synthesizer User Manual

MICRO LAMBDA WIRELESS, INC.

MLMS Synthesizer
User Manual

46515 Landing Pkwy. Fremont, CA 94538
Phone 510.770.9221 • Fax 510.770.9213

Rev A 7/15/2016

Table of Contents

1.0 Introduction

This manual describes the setup, operation and remote communication for the MLMS Synthesizer. The model and serial numbers are located on the label located on the top cover of the unit. Each unit has a separate, custom specification sheet for the particular model defining the synthesizer's frequency range, RF characteristics and options.

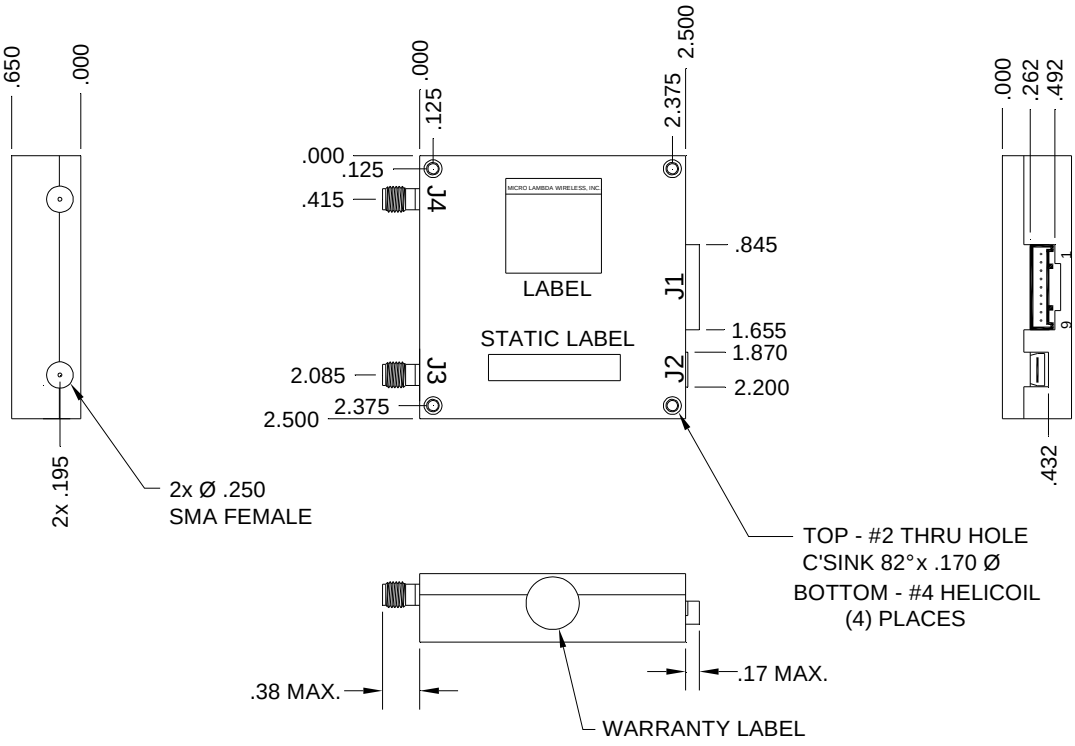
General operating/programming instructions are located herein.
The CD Rom supplied with the package contains a **SetupMLMS.msi** file, when executed, will create a folder named "MLMS Support Files" on the desktop containing short cuts to the manual, documentation and programs for interfacing the product with a personal computer. This **SetupMLMS.msi** file is compatible with Windows XP, Windows Vista and Windows 7, 8, and 10. The most current versions of these files, new offerings and standard synthesizer specifications can be downloaded at our web site: http://www.microlambdawireless.com

2.0 Package Contents

Item	Quantity
MLMS Series Synthesizer	Purchased Qty.
DC Power mating connector	Purchased Qty.
USB A male to USB Mini-B cable	Purchased Qty.
CD Rom (Contains manual, quick start guide and PC software)	1 each
MLMS Quick Start Guide (Printed)	1 each

3.0 General Overview of Product Capabilities

The MLMS series of YIG-Based synthesizers can be supplied in wideband or narrowband models and are ideal as the main local oscillators in receiving systems, frequency converters and test and measurement equipment. They provide 1 kHz frequency resolution over the 250 MHz to 16 GHz frequency range. Power levels of +8 to +13 dBm are provided throughout the series and full band tuning speed is 1 - 3 mSec. The units are 2.5" x 2.5" x 0.65" high and fit a 1 slot PXI chassis. Standard frequency ranges are 0.25 to 4.0 GHz, 0.25 to 8.0 GHz, 2 to 8 GHz, 6 to 13 GHz and 8 to 16 GHz. The MLMS can be configured to phase lock to an external reference signal in the 1 - 200 MHz range. Two types of interface come standard: USB for trouble free connection to a personal computer and a 5 wire serial interface for use in the customers system. Outline drawing 99-0211-001 shown on the next page defines the mechanical configuration. The drawing is displayed mainly for the mechanical and the connection information, many other configurations and options are available. See your model specification sheet or contact Micro Lambda Wireless, Inc. for details. Operating temperature ranges outside the 0 to 60 Deg. C range are also available.



NOTES :

-J1 MALE: - MOLEX# 0353630960

-J1 MATES WITH: - MOLEX# 0355070900

-CRIMP PIN: - MOLEX# 0502128100

- J1 POWER SUPPLY INPUTS REQUIRED FOR USB OPERATION
- J1 RECOMMENDED WIRE SIZE = A.W.G. 22-24
(*) ACTIVE LOW

CONNECTIONS			
CONN.	TYPE	PIN #	FUNCTION
J1	35363-0960	1	+15 VDC, +12V OPT.
J1	35363-0960	2	GROUND (PWR/LOGIC)
J1	35363-0960	3	+ 5 VDC
J1	35363-0960	4	SERIAL CLOCK
J1	35363-0960	5	SERIAL DATA IN
J1	35363-0960	6	SERIAL SELECT/ENABLE
J1	35363-0960	7	SERIAL BUSY
J1	35363-0960	8	LOCK ALARM OUT
J1	35363-0960	9	SERIAL DATA OUT

CONNECTIONS			
CONN.	TYPE	PIN #	FUNCTION
J2	USB MINI-B	1	+V
J2	USB MINI-B	2	D-
J2	USB MINI-B	3	D+
J2	USB MINI-B	4	GND
J2	USB MINI-B	5	GND
J3	SMA-FEMALE	1	REF. INPUT
J4	SMA-FEMALE	1	RF OUTPUT

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCE ARE :
FRACTIONS ± DECIMALS .xx ±.02 .xxx ±.010 ANGLES ±

WEIGHT: 4.0 Oz. / 113.4gr

FINISH: NICKEL

DO NOT SCALE DRAWING

CONTRACT NO.

APPROVALS	DATE
DRAWN N.NGUYEN	9/25/15
ENGR.	
MANUF.	
Q.A.	

MICRO LAMBDA WIRELESS, INC.

MLMS SYNTHESIZER, EXT. REF.

SIZE	CAGE No	DWG. NO.	REV.
	0RN63	99 - 0211 - 001	A

A simple block diagram is shown in Figure 1. The port configuration is shown in drawing 99-0211-001, the synthesizer provides a external reference input J3, a RF output J4, a DC power/serial interface port J1, and a USB Mini-B interface connector J2. The synthesizer is tuned via the J1 serial connection or the USB port J2. DC voltages of +5.0V and +15.0V (+12.0V optional), with adequate current to operate the product must be applied to J1 in order to communicate with the product.

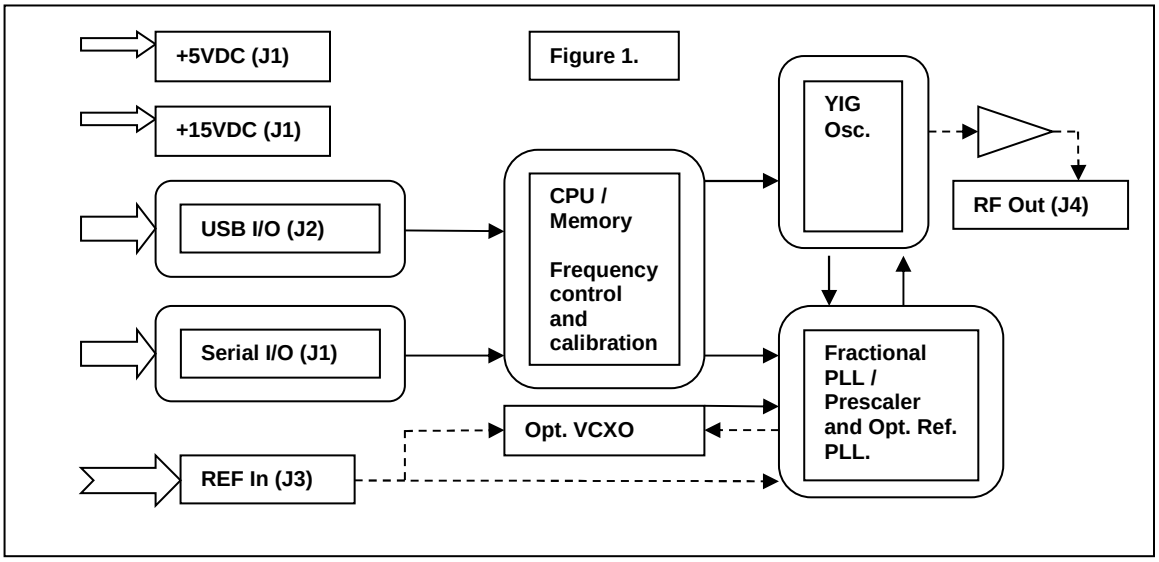


Figure 1.

4.0 Setup and Operation

This product is designed for a 0 to 60 Degree C environment and should not be subjected to humidity >95%. Use proper ESD handling procedures. Allow for a proper heat-sink able to dissipate the total wattage/heat generated at the highest frequency setting of the unit. Verify that all external RF/microwave cables and components connected to the unit are in good working condition.

4.1 Connections

Connect to the preferred interface port, Serial or USB. USB Mini-B to USB Male - A cable for connection to a host PC USB port, the USB interface is USB 1.1 and 2.0 compatible. The serial port is a 0-5V, CMOS/TTL compatible port; it is very similar to an SPI communication port. Clock, Data and Select/Enable operate in the typical serial communication format, except that the maximum clock speed is 10 MHz. The Busy line is for handshake to the controller, a High on this line tells the controller that the synthesizer is busy and should not be selected for communication at this time. The Data Out line is used to read data from the synthesizer if a command requested it. The data is clocked into the controller by clocking out data of 0's to the MLMS and the controller reads during this time. The Data Out line will be taken high by the MLMS, when data is ready. The Busy line must be monitored. See section 6.0/6.1 serial communication for details. The Lock Alarm signal J1 pin 8 is a hardware logic signal that shows the state of the internal phase locked loop circuits.

This line should be a TTL high for the majority of the time unless the unit is stepping frequency, then it will pulse low when switching between frequencies, typically it will be low for less than 1 millisecond.
Connect an external reference frequency (if required) to J3. Connect a +5.0 VDC and a +15 VDC power supply with adequate current to operate the product (see spec sheet) to the J1 connector; +5.0 VDC on J1 pin 3, and +15.0 VDC on J1 pin 1, and a common ground for both supplies on pin 2. Note the power supply inputs are protected from damage up to +20 VDC.

4.2 MLMS operation

Turn on the power supply voltages and verify that the current for each supply is below the maximum stated current in the specifications for your model. A 1 minute warm up is recommended before use. The unit should be operating at the last frequency it was set to before power down; this would typically be Fmin when shipped from the factory. The Lock Alarm line (J1 pin 8) should be high.

5.0 Controlling the MLMS using a personal computer

The MLMS Synthesizer can be controlled by a personal computer for Demo purposes. The requirements for this are as follows: A USB ver. 1.1 or 2.0 port on the PC, Windows XP, Windows Vista or Windows 7, 8, and 10 (32 or 64 bit versions), the programs included on the MLMS Support CD and a power supply capable of supplying the DC Voltage and Current required to operate the MLMS synthesizer.

5.1 Installing the documentation and control software

The CD ROM supplied with the MLMS contains the file named **SetupMLMS.msi**. Execute this file to install the manual, documentation and control programs for PC interface. The setup file, when run, will create a folder named “MLMS Support Files” on the computer desktop with short cuts to the documentation and interface program. (Note: This setup file must be executed for the USB PC interface program to operate correctly, dll files will be installed to the system directory.)

5.2 USB Interface

The MLMS product, when connected using the USB interface, appears as a USB HID device (Human Interface Device) to the Windows operating system. The USB HID driver is supplied with the windows operating system, and is installed automatically when the unit is connected to the PC’s USB port.
The MLMS may be controlled remotely via a USB connection using the supplied “MLMS PC interface.exe” program. A screen capture of this program is shown in Figure 2. Additional information is accessed via the program’s pull-down menus “File” and “Help”. Included in these menus are View/Print the configuration NOVO locations and data, a list of all commands that the unit will respond to, and a description of NOVO locations and what is stored at each location. On the program screen you will see some limited information about the unit. PLL Lock status is also shown; it is updated each time the

"Update Info" check box is selected. Commands may be sent to and received from the unit. The unit can also be stepped up and down in frequency using the "Step Up" and "Step Down" buttons, the frequency will increment and decrement based on the frequency shown in the step size box. This number can be changed to any valid step size within the frequency range limits of the unit. The current frequency setting is also shown. The program can be used to connect to multiple units; all units that are connected to the PC's USB ports will show up in the pull down list in the "Choose Unit #" box. If units are added after the program has been initiated or the MLMS(s) were powered down for a time, press the "Refresh" button to update the list. A sweep mode feature is included to allow the unit to sweep (Step and Dwell) at each frequency based on the step size and dwell time text boxes. Start and Stop frequency can also be adjusted for a narrower range, if needed. Four sweep modes are selectable; Auto = continuous sweep across the start/stop range retraces from stop frequency and repeats. Single = one single sweep from start to stop. Manual = unit will take one step up or down from the current frequency when the step up/down keys are pressed. Frequency List mode = frequency sweep based on a user generated list of frequencies. To configure this mode, click the Edit List button and input one frequency per line (In MHz, i.e. 2450.125) in the text editor and save/exit. The file is created and named for the serial number of the unit. Make sure the List Mode is selected and choose your desired sweep mode, then press Run. The unit will step the frequency based on your list of frequencies and the dwell time selected. Frequencies can be increasing, decreasing or random. Multiple versions of the program can be executed to sweep or communicate with more than one unit at a time.

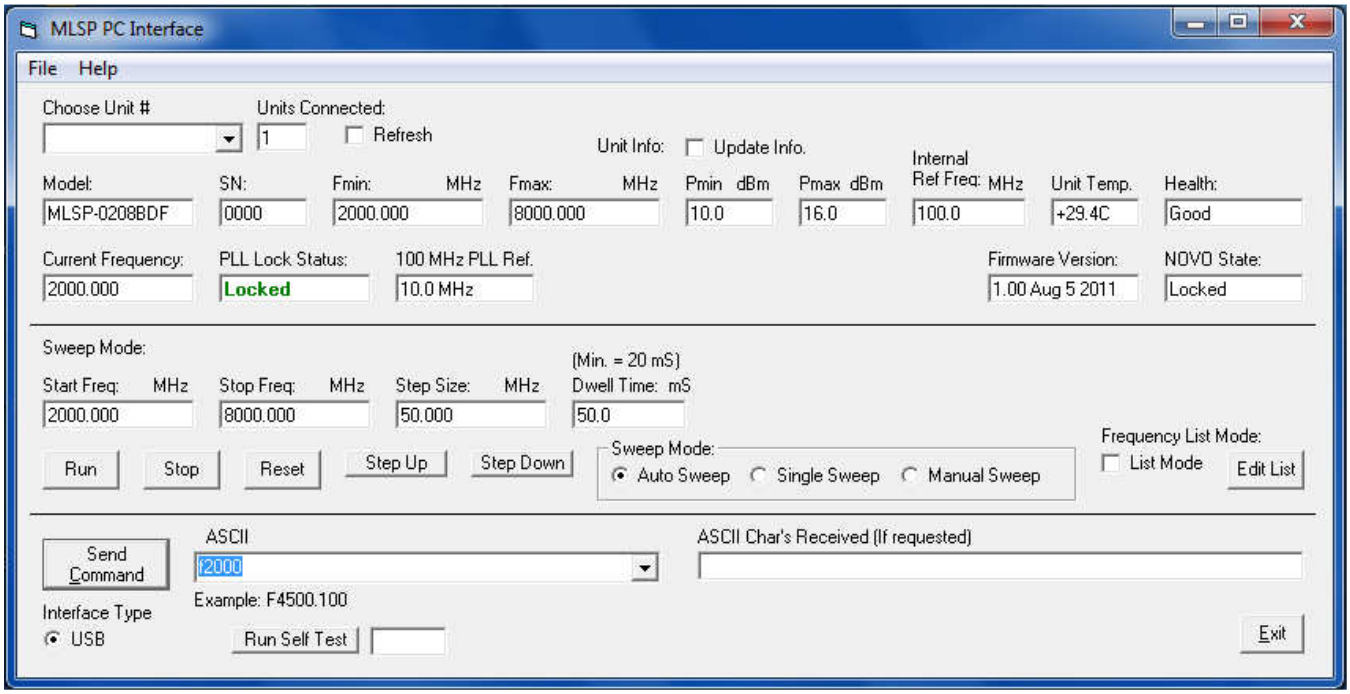


Figure 2

5.3 USB HID PC Interface example C# source code

Included in the installed support files folder is an example of a simple USB HID interface program written in C#. The project file and source code were written using Microsoft Visual Studio Express 2010, C Sharp. Visual Studio Express 2010, C Sharp can be downloaded for free at http://www.microsoft.com/visualstudio/en-us/products/2010-editions/visual-csharp-express. After you register this free version of the programming environment, you can create and edit C# programs.
This source code will allow you to get started integrating the MLMS USB communication into your own C programs. The example program is a simple interface in which you can send and receive ASCII characters. The interface tests for a connection to the MLMS synthesizer by looking for a return string identifying the MLMS unit after searching through all attached USB devices.
An executable version of the program is located on the CD in the directory - \MLMS USB HID PC Interface C# Example\bin\x86\Release\. The file name is – MLMS USB HID PC Interface C Example.exe. A screen shot of the interface is shown in figure 3.

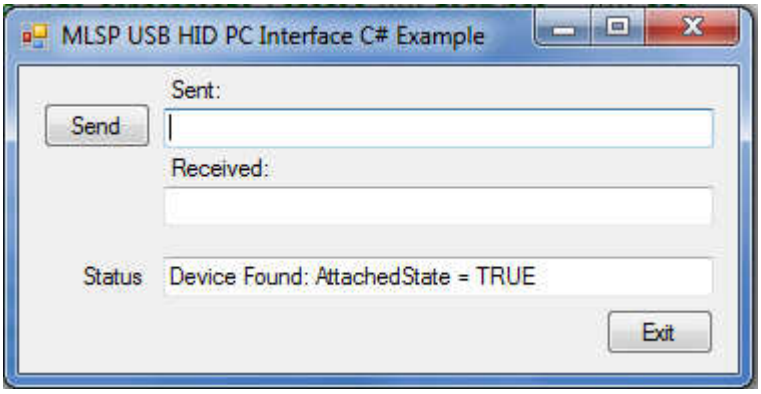


Figure 3

6.0 Serial interface

The MLMS can be programmed using a 5 wire serial bus. The timing diagram is shown in Figure 4. The Five control lines are as follows:

SELECT (J1 – Pin 6)	Input	Active Low. Enables the shifting of data into the internal command buffer. Also serves as a command terminator when it goes HIGH. The status of the BUSY line should be checked before activating this line. Note: All command + data, command strings must be sent using a single Select Line Cycle. When low, Busy is high.
CLOCK (J1 – Pin 4)	Input	Data is clocked into the unit on the rising edge (Positive Edge Clocked) and DATA OUT is valid at this time. The maximum clock rate of this line is 100 nsec. For best performance the status of the DATA OUT line should be checked before sending the first Clock. This line should be maintained in a LOW state at the application of Select to prevent confusion.
DATA IN (J1 – Pin 5)	Input	Input data pin. Data is sent MSB first. Data must be stable 100 nsec. Before the CLOCK line goes high and 100 nsec. After the clock goes low. (Setup / Hold time)
BUSY (J1 – Pin 7)	Output	This line is used to indicate that the unit is busy processing other commands or doing its internal housekeeping. Before sending a Select = TRUE the status of this line should be checked to ensure that it is LOW (NOT Busy). Any command initiated by setting Select Low while BUSY is High may result in lost data and uncertain results. Note: Busy goes high, once Select is set low. NOTE: The unit can be programmed without using this line if sufficient time is allowed between Clocks and between commands. The time required varies between commands. This mode is not recommended as there are some events that occupy the microcontroller other than the serial communications.
DATA OUT (J1 – Pin 9)	Output	This line is used to pass internal information from the synthesizer to the user. Data is guaranteed to be valid on the falling edge of the clock signal. Data is sent out MSB first. In addition, this line is used as a communication 'handshake' line. Once SELECT has gone LOW the DATA OUT line will be taken HIGH to indicate that the unit is listening. It will remain HIGH until the first data is sent out which is initiated by the first rising edge of CLOCK. DATA OUT will be returned to LOW after Select has been released. All DATA in/out is in ASCII format.

For the following information, please reference the above documentation regarding Busy and Data out.

Recommended send data sequence: The format is one select per command string. The string length is variable up to 16 ASCII characters. A decimal point is required for resolution less than 1.0 MHz. The unit will accept a frequency command with a resolution of 1.0 Hz, and it will try to get as close to the frequency requested as it can, typically <= 10.0 Hz accuracy. So the command to set the frequency F8000.1 would be sent as follows: Set select low, clock out 01000110 00111000 00110000 00110000 00110000 00101110 00110001 (Ignore spaces, only used to single out each ASCII char), then set select high. The unit should go to 8000.1 MHz. When select goes high, this tells the unit that the user is done and to starts interpreting the command. If the command is not understood, the unit will do nothing.

Recommended read data sequence: The returned data is variable in length; however it is recommended that the full 16 bytes of data be read to clear the buffer. Set select low, send the desired read command and set select high. The unit interprets the command and places the requested data in its buffer. Then set select low and clock out 16 ASCII nulls while clocking in the data, then set select high. Example: To read the units internal temperature, set select low and send ASCII T (01010100), set select high. Set select low and send 16 00000000 while reading the data line and clocking in bits. Set select high. The information should be similar to +25.0C, in ASCII. All of the memory locations in the unit can be read in this manner, using the R command.

In addition to the lines above there is a unit status line, LOCK ALARM status (J1- Pin 8), which is a static line. This TTL output (High = Locked) indicates the overall health of the unit – specifically, that all of the internal phase locked loops are locked.

The MLMS serial interface lines operate on internal 3.3V logic of a PIC microcontroller; this should allow the unit to communicate in systems operating with 2.5V, 3.3V and 5.0V serial control lines.

6.1 Serial interface timing diagram:

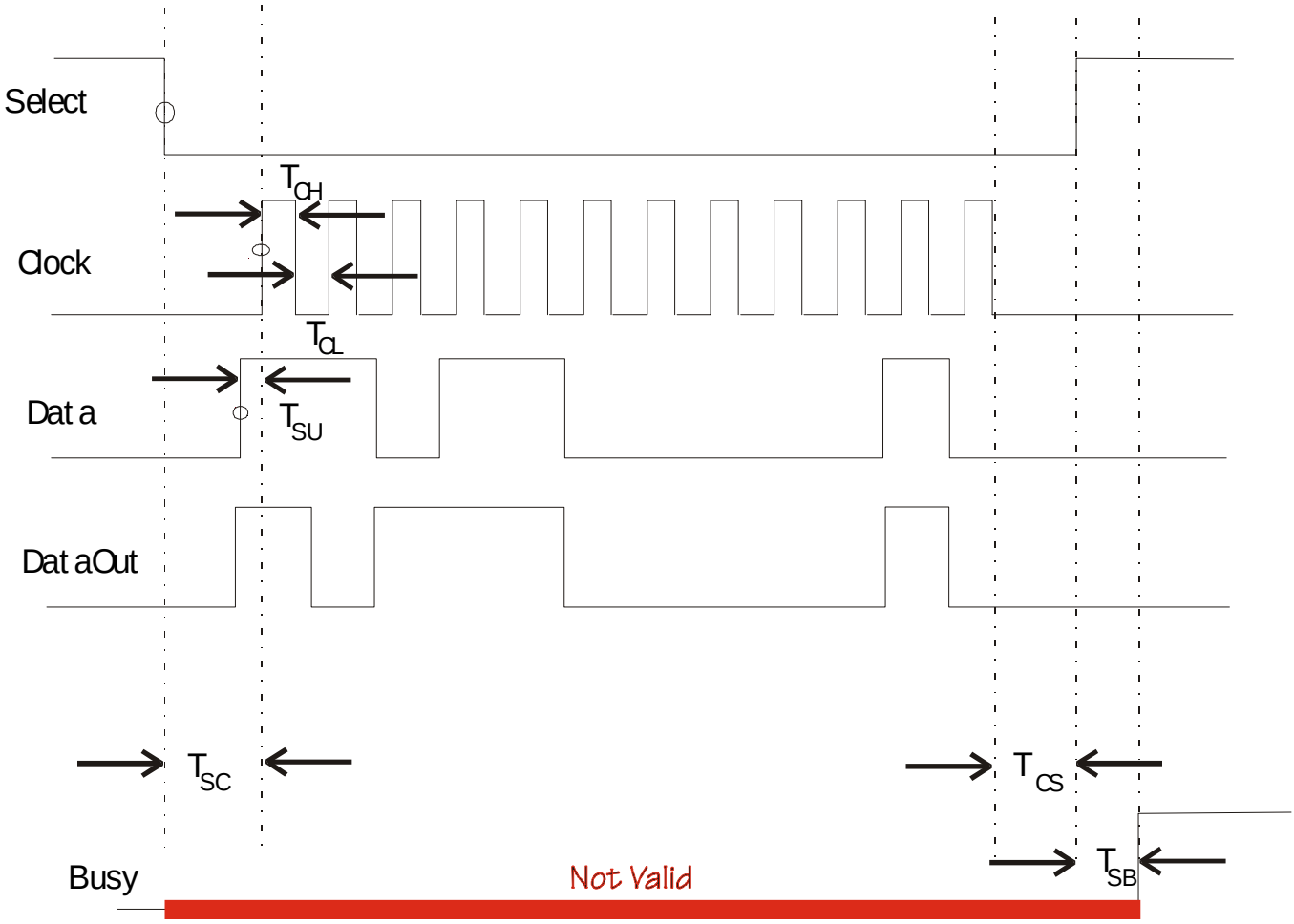


Figure 4

Tsc > 100 nsec select low before first clock
Tcs > 100 nsec clock low before chip select high
Tsu > 100 nsec data stable before rising edge of clock
Tch > 100 nsec minimum clock high time
Tcl > 100 nsec minimum clock low time
Tsb > 500 nsec (time to wait before sampling 'BUSY')

Data/Clock Setup time = >50 nsec.
Data/Clock Hold time = Tch + 50 nsec.

7.0 Communication syntax

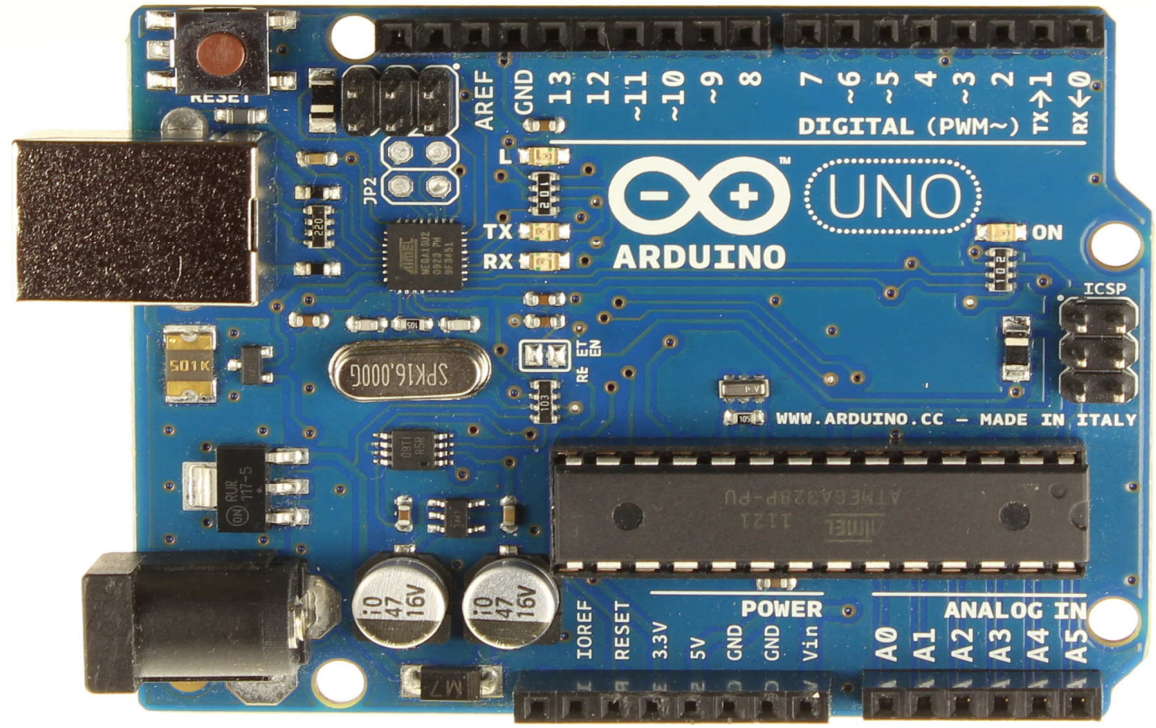
The following table describes the commands that the MLMS supports. This is a custom syntax created by Micro Lambda Wireless, Inc. All commands are sent and received in ASCII format. The commands are NOT case sensitive. These commands can be used with all forms of communication (USB and Serial).

Command	Function	Comment
?	Report Status – Bits, D0 = 100 MHz lock, D1 = YIG PLL, D6 = self test, D7 = NOVO lock	Typical return = 11000011
F	Frequency (ASCII) (Dec. #)	ASCII freq in MHz: xxxxx.xxxxxx; (example: F12345.678900)
MR	Recall a user saved frequency setting from memory location (MR25)	0-99, recalled from NOVO location 200-299
MS	Save current frequency setting of unit to memory location (MS75)	0-99, stored @ NOVO location 200-299
POWERON	Turns ON internal supplies related to +15V input	Turns ON YIG / PLL / Analog supplies.(default = on @ power up)
POWEROFF	Turns OFF internal supplies related to +15V input (Low power state)	Digital logic Supplies are always on.
R	Read a NOVO location	R0 returns model number
SP	Synthesizer preset to factory settings.	Copy NOVO Loc. 900-960 to 0-60, and reboot
SR	Software based Processor Reset (Reboot)	Reset PIC, clear var. run PIC code from start; (example: SR)
ST	Self Test	Execute internal test; 1 = Pass; (example: ST, then read data)
T	Read internal temp.	Returns ASCII chars, reading in Deg. C; (example: T, then read data) returns i.e. +25.0 C
V1	Read YIG PLLV (typical Range = 1V to 12V)	6.75V = normal; (example: V1, then read data)
V2	Read 100 MHz PLL V (typical Range = .5V to 3V)	1.65V = normal; (example: V2, then read data)
V3	Read internal +2.5V voltage	2.50V = normal; (example: V3, then read data)
V4	Read internal +3.3V voltage	3.30V = normal; (example: V4, then read data)
V5	Read internal +5.0V voltage	5.00V = normal; (example: V5, then read data)
V6	Read internal +6.75V voltage	6.75V = normal; (example: V6, then read data)
V7	Read internal +13.5V voltage (+11.0V for +12V Opt.)	13.50V = normal; (example: V7, then read data)
V8	Read internal -5.0V voltage	-5.0V = normal; (example: V8, then read data)
R0000	Model Number **(Example = R0) Read Location 0.**	MLMS-0208B ("W" blocked with NOVO locked.) (R/W = 16 Bytes)
R0001	Serial Number	0002
R0002		
R0003	Fmin, in MHz	2000 (unit is tunable 100.0 MHz below Fmin.)
R0004	Fmax, in MHz	8000 (unit is tunable 100.0 MHz above Fmax.)
R0005	Current Internal uW PLL Reference Frequency Setting - MHz	R# = 1 – 200 MHz, typ. = 100
R0006	RF min, in dBm	10.0
R0007	RF max, in dBm	15.0
R0008	Temp min, in Deg. C	0
R0009	Temp max, in Deg. C	60
R0010	Highest Temp reached, in Deg. C	59.8
R0011	NOVO State - Locked/Unlocked	Locked
R0012	Firmware Version & date	2.0 Nov 16 2015
R0013	Unit Health Status – “Good” or Self-Test failure information	Good or Fail V5 as example
R0014	Unit Calibration Status - Yes/No	Yes
R0015	Self-Test Results - Pass/Fail	Pass
R0016	Current Output Frequency setting - MHz	2500
R0017	Xtal Setting – Ext or ExtXtal	ExtXtal (2 modes; External and External with Int. Xtal.)
R0018	Current Downconverter divider setting - (1, 2, 4, 8)	2
R0019	Coarse DAC Fmin cal # (Hex)	0000-FFFF
R0020	Coarse DAC Fmax cal # (Hex)	0000-FFFF

Command	Function	Comment
R0021	Current Coarse DAC setting (Hex)	0000-FFFF
R0022	Current Fine DAC interpolated setting (Hex)	0000-FFFF, 8000 nominal
R0023		
R0024	Current Microwave Divider (DV) setting, 1, 2 ,4 ,8 (Dec)	2 (Sets uWave Prescaler to 1, 2, 4, or 8)
R0025	Current uW PLL Reference divider setting	2 Ref Divide #
R0026	Coarse Cal status; Yes / No	Yes
R0027	Fine Cal status; Yes / No	Yes
R0028	Current Downconverter Divide # setting (1, 2, 4, 8)	8
R0029	External reference freq. In MHz for 100 MHz PLL (ExtXtal mode)	i.e.: 10 = 10 MHz external reference. 1.0 MHz increments only
R0030		
R0031	Customer part number, if shown on P.O.	123-45-6789 (Shown on unit label as PN:)
R0032	Frequency resolution in MHz (or Step Size)	0.001 = 1.0 kHz
R0033	Spurious Spec., in dBc	-60
R0034	Harmonics Spec., in dBc	-12
R0035	Phase Noise Spec. @ 100 Hz Offset, in dBc/Hz	-80
R0036	Phase Noise Spec. @ 1 kHz Offset, in dBc/Hz	-95
R0037	Phase Noise Spec. @ 10 kHz Offset, in dBc/Hz	-100
R0038	Phase Noise Spec. @ 100 kHz Offset, in dBc/Hz	-120
R0039	Phase Noise Spec. @ 1 MHz Offset, in dBc/Hz	-140
R0040	Switching Speed, any step, in mS	3.0
R0041	+15V Supply current Max, in mA	450
R0042	+5V Supply current Max, in mA	450
R0043	Doubler mode enabled (X2)	Yes/No/Blank
R0044	Downconverter (Divider) mode enabled	Yes/No/Blank
R0045		
R0046		
R0047		
R0048		
R0049		
R0050		
R0051	Level flatness Spec. in +/- dB (+/- 2.0 = 4.0 total)	2.0
R0052		
R0053	uW Divider switch point in MHz	8100
R0054	+12V Option Installed	Yes / No
R0055	Low end limit for CP gain	43d
R0056	High end limit for CP gain	127d
R0057	Current CP setting #	91d
R0058	MLWI Sales (Job) number	21-0024
R0059	MLWI Product Outline Drawing # and Revision	211-001 A
R0060	Power State (Power supplies on or off) On power-up will default to ON!	"ON" or "OFF"(Low power) - Show status of "poweron" and "poweroff" commands.
….		
200-299	User Saved / Recalled frequency setting locations, (0-99)	Frequency stored in MHz
….		
R 900-960	Config data backup safe area, SF - save factory stores data here.	Backup copy of NOVO location 0000 to 0060 (Config Data)
….		
R1000-2047	DAC cal data, stored in 25 MHz increments, Fmin-100 to Fmax+100 MHz, 8000 Nom.	Stored in 16 bit HEX numbers (ASCII format)

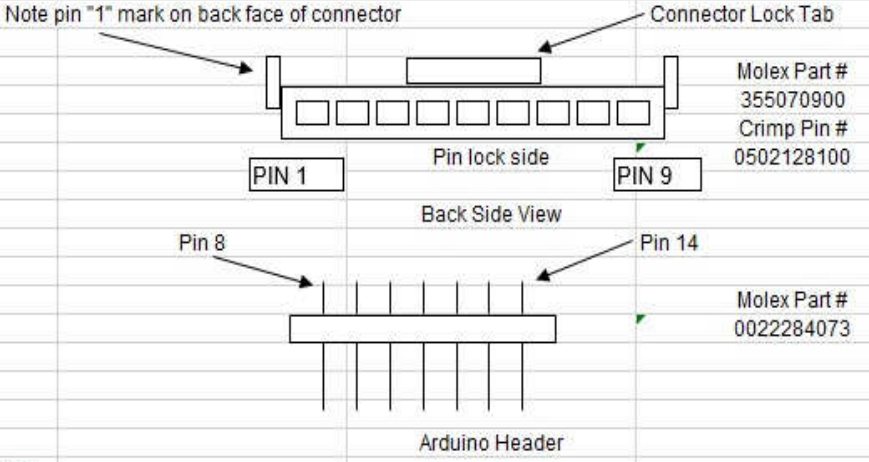
8.0 Arduino to MLMS Serial Interface Example

In this section we offer an example of serial interface using an Arduino UNO microcontroller, coded in C language. The Arduino series of educational prototype boards is a simple and effective way to communicate with the MLMS synthesizer. They are readily available for purchase on many web sites, including Amazon.com, for under $15.



The wiring interconnection between the Arduino and the MLMS is as follows:

WIRE	COLOR	FROM	TO	NOTES	A.W.G	LENGTH
1	BLUE	9 PIN FEMALE - 1	BLUE BANANA MALE	+15 VDC	22	36"
2	BLACK	9 PIN FEMALE - 2	BLACK BANANA MALE	GROUND	22	36"
3	BLACK	9 PIN FEMALE - 2	Arduino UNO - 14	LOGIC GROUND	22	36"
4	RED	9 PIN FEMALE - 3	RED BANANA MALE	+5 VDC	22	36"
5	ORANGE	9 PIN FEMALE - 4	Arduino UNO - 8	CLOCK	26	36"
6	BROWN	9 PIN FEMALE - 5	Arduino UNO - 11	DATA	26	36"
7	WHITE	9 PIN FEMALE - 6	Arduino UNO - 12	ENABLE (Select)	26	36"
8	GRAY	9 PIN FEMALE - 7	Arduino UNO - 13	BUSY	26	36"
9	BLACK	9 PIN FEMALE - 8	RED LED - Cathode (-)	Cathode to Lock Alarm	26	3"
10	RED	9 PIN FEMALE - 3	RED LED w 1.0K res. series	Anode to +5 VDC	26	3"
11	YELLOW	9 PIN FEMALE - 9	Arduino UNO - 10	DATA OUT	26	36"



Notes:

1. Crimp insulator and crimp wire and solder pin to wire.
2. Insert pin and wire with metal lock tab face down so pins will lock into place.
3. Do Not use Flux on connector or pins.
4. Heat shrink all pins on Adruino UNO connector.
5. Split power supply (w/ Banana plugs) and data wires into 2 bundles, cable tie each bundle every 6".
6. Add 1K res. To Anode of LED, hestshrink wire and resistor, then heatshrink both wires.

Arduino Code example:

```
/*
  Communicate with Micro Lambda Wireless, MLSP or MLMS Synthesizer, using the Arduino serial monitor.
  For wire harness diagram see - MLMS synthesizer harness w-LED for Arduino interface 36 inch.xls
  D.S 6/12/16
*/
const int clockPin = 8;//Serial Clock line
const int dataInPin = 10; //Serial Data (IN) line RX
const int dataOutPin = 11; //Serial Data (OUT)line TX
const int selectPin = 12;//Serial Select line
const int busyPin = 13; //Serial Busy line
//Pin 14 = Serial Logic Ground
String bitOrder = "MSBFIRST"; //serial stream bit order
int busy;
char sendArray[16];
char recvArray[16];
int counter;
int b = 0;
int i = 0;
int x = 0;

void setup() {
  //set pins to output/input serial data
  pinMode(clockPin, OUTPUT);
  pinMode(dataInPin, INPUT);
  pinMode(dataOutPin, OUTPUT);
  pinMode(selectPin, OUTPUT);
  pinMode(busyPin, INPUT);
  digitalWrite(selectPin, HIGH); //Init the selectPin high
  Serial.begin(9600);
  while (! Serial); // Wait untilSerial is ready - For Leonardo
  Serial.println("MLSP Arduino Synthesizer Interface - Enter a Command");
}

void loop() {
  while (Serial.available() > 0) {
    //read all char's to send from serial monitor
    delay(1); //wait for serial port
    sendArray[i] = Serial.read();
    i = i + 1;
    delay(1); //wait for serial port
    //when all char's received, call serial send frequency routine (sends ASCII #'s)
    if (Serial.available() == 0) {
      sendFreq();
    }
    //prints the command sent / received to the serial monitor window
    if (Serial.available() == 0) {
      printComm();
    }
  }
}

//send / receive command routine
void sendFreq() {
  do {
    busy = digitalRead(busyPin); //read busy line, if low continue
  } while (busy == 1);
```

```
//loop for array length of chars to send
for (counter = 0; counter != i; counter++)
{
  digitalWrite(selectPin, LOW); //selectPin Low and hold low for as long as you are transmitting
  delayMicroseconds(10);
  //shift out 8 bits with xx uS delay (Bit Bang)
  for (b = 0; b < 8; b++)  {
    if (bitOrder == LSBFIRST)
    { digitalWrite(dataOutPin, !!(sendArray[counter] & (1 << b)));
      delayMicroseconds(2);
    }
    else
    { digitalWrite(dataOutPin, !!(sendArray[counter] & (1 << (7 - b))));
      delayMicroseconds(2);
    }
    //toggle clock line
    digitalWrite(clockPin, HIGH);
    delayMicroseconds(2);
    digitalWrite(clockPin, LOW);
    delayMicroseconds(2);
  }
  delayMicroseconds(5); //delay between chars for byte visibility on Dig. scope
}
digitalWrite(selectPin, HIGH); //pull the selectPin to save the data
digitalWrite(dataOutPin, LOW); //set data out low when done
digitalWrite(clockPin, LOW); //set clock low when done

delayMicroseconds(10); //wait for a xuS between write/read

//read back data from unit if available
do {
  busy = digitalRead(busyPin); //read busy line, if low continue
} while (busy == 1);

digitalWrite(clockPin, HIGH); //clock line high to start for read

//read back data from unit if available, 16 bytes
for (counter = 0; counter < 16; counter++)
{
  digitalWrite(selectPin, LOW); //selectPin Low and hold low for as long as you are receiving
  delayMicroseconds(10);
  //shift in 8 bits with xx uS delay (Bit Bang)
  for (b = 0; b < 8; b++)  {
    if (bitOrder == LSBFIRST)
    { recvArray[counter] |= digitalRead(dataInPin) << b; //shift in bits
      delayMicroseconds(2);
    }
    else
    { recvArray[counter] |= digitalRead(dataInPin) << (7 - b); //shift in bits MSB first
      delayMicroseconds(2);
    }
    //toggle clock line
    digitalWrite(clockPin, LOW);
    delayMicroseconds(2);
    digitalWrite(clockPin, HIGH);
    delayMicroseconds(2);
  }
  delayMicroseconds(5); //delay between chars for byte visibility on Dig. scope
```

```
  }
  digitalWrite(selectPin, HIGH); //pull the selectPin to save the data
  digitalWrite(dataOutPin, LOW); //set data out low when done
  digitalWrite(clockPin, LOW); //set clock low when done

  delayMicroseconds(10); //wait for a 5uS between commands
}

//format and print command sent/received to serial monitor window
void printComm() {
  Serial.print("Command Sent = ");
  for (x = 0; x < 16; x++) {
    Serial.print(sendArray[x]);
  }
  Serial.println();
  Serial.print("Info Received = ");
  for (x = 0; x < 16; x++) {
    Serial.print(recvArray[x]);
  }

  Serial.println();
  //clear var.
  i = 0;
  x = 0;
  b = 0;
  memset(&sendArray[0], 0, sizeof(sendArray)); //clear array
  memset(&recvArray[0], 0, sizeof(recvArray)); //clear array
}
```

9.0 Hardware installation information

The unit may be installed into a system using four #2-56 X 0.875” long flat head screws, inserted from the top, in the 4 corner mounting holes. Alternately, four #4-40 X 0.25” screws can be attached from the bottom side through the mounting surface, in the 4 corner mounting holes. The Molex connector (J1) and the mating connector information are shown in the drawing number 99-0211-001 in this document. The USB connector is the standard Mini-B style.

10.0 Technical Support

For Technical support please contact:

Micro Lambda Wireless, Inc.
46515 Landing Pkwy.
Fremont, CA 94538
Ph: (510) 770-9221
Fax: (510) 770-9213

Email: sales@microlambdawireless.com

You can visit our website at http://www.microlambdawireless.com for updated information, specifications and downloads.

11.0 Warranty

Seller warrants for a period of twelve (12) months from the date of original shipment that the products will be free from defects in material and workmanship and design (if of Micro Lambda Wireless, Inc. design) and will be in conformity with applicable specifications and drawings and all other contractual requirements. However, this warranty shall not apply to any product which that has been subjected to misuse, misapplication, accident, improper installation, neglect, unauthorized repair, alteration, adjustment, inundation or fire. See the complete warranty and return policy document number 201-005 Rev- at our website at http://www.microlambdawireless.com.