The Mini Board Technical Reference

Fred G. Martin*

August 7, 1998

The *Mini Board 2.0* is a single-board computer optimized for controlling small DC motors and receiving data from various electronic sensors. Its miniature size (3.1 by 1.9 inches, smaller than a business card), low power operation, and programmability make it ideal for control of a small, mobile robot. It communicates with a desktop computer over a standard RS-232 serial line, making it suitable for desktop computer-based control as well.

The Mini Board 2.0 features:

- Control of four DC motors, at voltages of 6 to 36 volts and up to 600 mA of current. Using software-based *pulse width modulation*, motors may be controlled at 16 levels of speed from off to on in either direction.
- Eight analog inputs for continuously varying sensor devices.
- Eight digital inputs for on/off type sensor devices.
- Three or four programmable counter inputs and four or five timer outputs. One of the timer pins is used to drive a piezo beeper for sound output.
- An RS-232 compatible serial port for communicating with desktop computers.

---

*The Media Laboratory at the Massachusetts Institute of Technology, 20 Ames Street Room E15–320, Cambridge, MA 02139. E-mail: `fredm@media.mit.edu`. An electronic version of this document and the freely distributable software described herein are available via anonymous FTP from `cherupakha.media.mit.edu` (Internet 18.85.0.47). Acknowledgments: Much of the technology described in this document is based on prior designs created in collaboration with Randy Sargent (see *Background* section). Amy Bruckman provided helpful comments on the organization of this manual.

- A high-speed synchronous serial bus that may be used to implement a network of Mini Boards. The network may be controlled by a single desktop computer, a single Mini Board, or it may have distributed control.
- Single power input for powering electronics and motors. On-board voltage regulator allows board to be powered by any DC voltage source of 6 volts or greater.
- Motorola 6811 8-bit microprocessor with 256 bytes of internal RAM and 2K bytes of electrically erasable programmable ROM (program memory).
- 6811 software libraries provided for embedded control applications using an inexpensive C compiler.
- 6811 monitor program supplied for use of board as tethered controller operated by desktop computer.
- MS-DOS, Macintosh, and Unix software provided for downloading 6811 programs to board over serial line.

Background

The first incarnation of what is now the Mini Board 2.0 was a controller board designed by Randy Sargent and this author for the LEGO Robot Design Competition held at the Massachusetts Institute of Technology in 1990. This board had many of the same features as the current Mini Board, including a 2K EEPROM version of the 6811, two L293D motor driver chips, and an RS-232 serial port interface.

For the Robot Design contest in the following year, Randy and I designed a more sophisticated controller board with 32K of external battery-protected RAM, an LCD display, and several other new features. This board, still used in the M.I.T. design course, is now in its third revision.

Even after successfully using the 32K board, we perceived the value of the simpler controller board. In the middle of 1991, we revised the design used in the 1990 contest. We dubbed the new board the "Mini Board 1.5," because it was small (37% smaller than the 1990 board) and was derived from our first board. The version 1.5 board worked, but it had a couple of minor wiring bugs, and was never distributed publicly, lacking any documentation.

In early 1992, Randy and I were offered the chance to organize a robot-building workshop at the third Artificial Life conference being held in Santa Fe, New Mexico. It seemed like an ideal occasion to use Mini Boards, because they were easy to use yet powerful enough for introductory robotics work.

I set about revising the design and developed the current Mini Board 2.0. The new board is 13% smaller than the version 1.5 board, uses better-quality switches and connectors, and adds networking capability. The version 2.0 design, free of any layout bugs, "premiered" at the June 1992 Artificial Life conference, along with the first version of these notes.

Since then, several hundred copies of the board have been printed up and distributed by Greg Ratcliff at Ohio State University. This manual has incrementally improved to include corrections made by different readers as well as new material to better support the expanding user base.

Contents

List of Figures

1 Introduction to the Mini Board

This section explains the capabilities of the Mini Board, indicating the location and function of each connector port. Please refer both Figure 1, a photograph of the Mini Board (enlarged for clarity), and Figure 2, a diagram of the Mini Board's connectors, in the discussion that follows.

1.1 Power Connector

In the upper right corner of the Mini Board is the power connector. The upper pin is (+) and the lower pin is (−). Screw terminals accessible from the top of the connector are used to fasten wires inserted from the edge.

The Mini Board will operate on a DC supply of 6 to 36 volts. One hundred milliamperes of current are required when the motors are not operated. When motors are running, additional current will be required as per the demands of the motors.

When using motors, the power supply voltage should equal the ratings of the motors plus 1 to 2 volts. For example, when operating 12 volt motors, a power supply of 13 to 14 volts should be used.

1.2 Motor Ports

In the lower right corner of the Mini Board are the four *motor ports*, labelled *Motor 1* through *Motor 4*. When a motor is powered, one of the two of its status LEDs will light, depending on the direction it is driven.

The C function `motor(int m, int s)` sets motor `m` at speed `s`. Speeds range from -16 for full backward to $+16$ for full forward. The command `motor(m, 0)`, which turns a motor off, can alternately be specified by `off(m)`.

Each motor port is three pins wide; a motor is wired to the two outer pins, as in Figure 3. The center pin is used when controlling two non-directional devices, such as the lamps shown in Figure 4. When used to power lamps, only one of the lamps can be on at a time (depending on the direction of the motor port that is selected).

The motor ports are powered by a pair of L293D motor driver ICs. These ICs are rated for operation of up to 36 volts with a maximum current drive of 600 milliamperes.

The L293D chips are connected to the *Port B* output pins of the 6811 microprocessor. The upper four bits of Port B enable motor ports 4 through 1 (respectively) for on and off; the lower four bits determine the ports' directionality. Port B is not

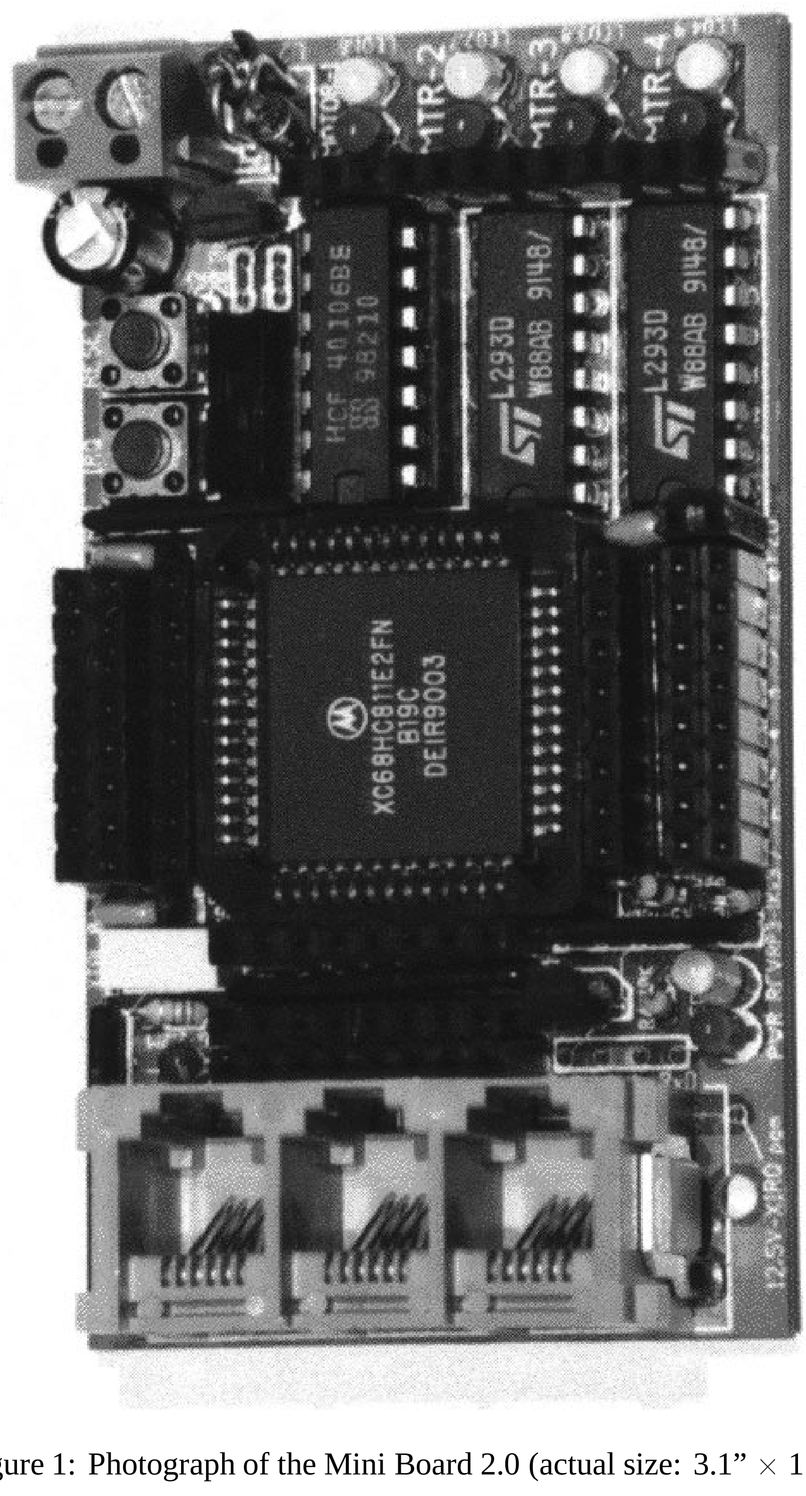


Figure 1: Photograph of the Mini Board 2.0 (actual size: 3.1” × 1.9”)

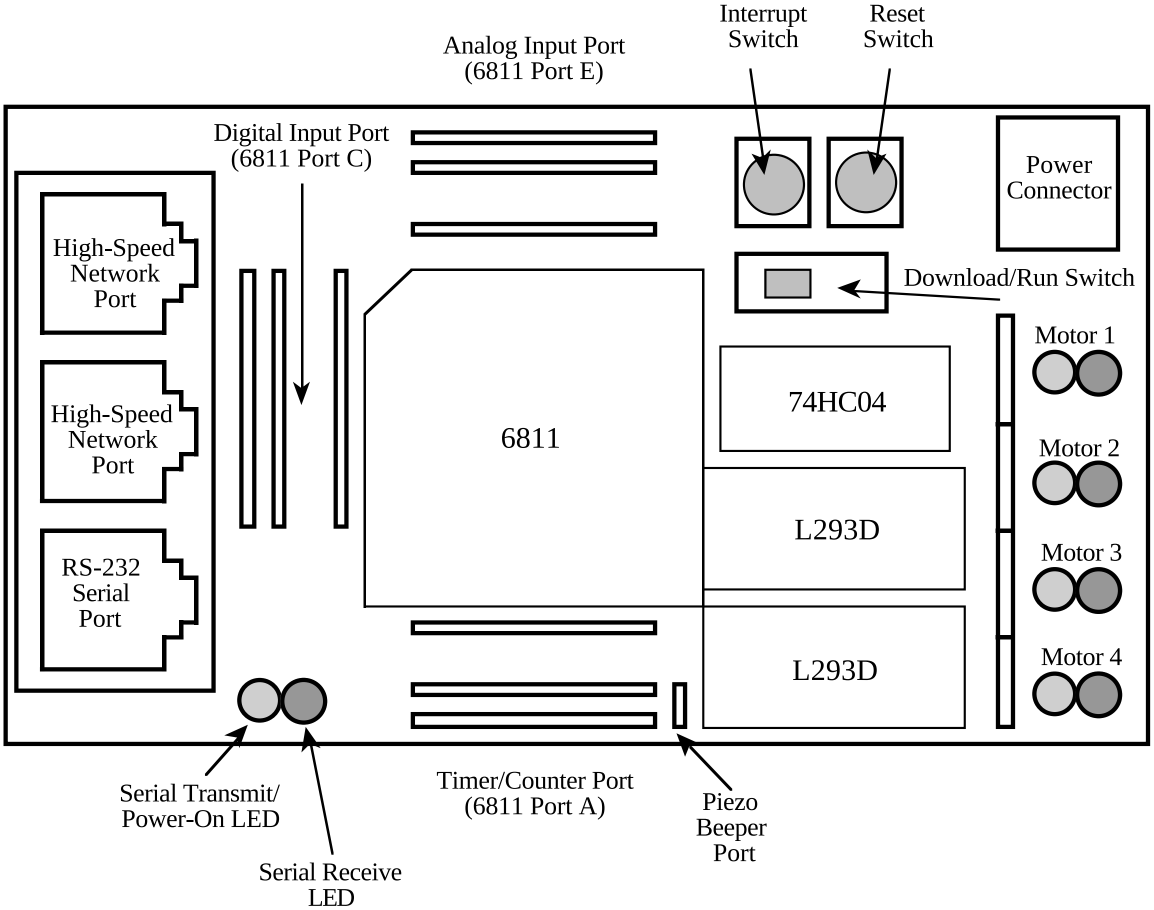


Figure 2: Mini Board 2.0 Connector Ports, Switches, and Indicators

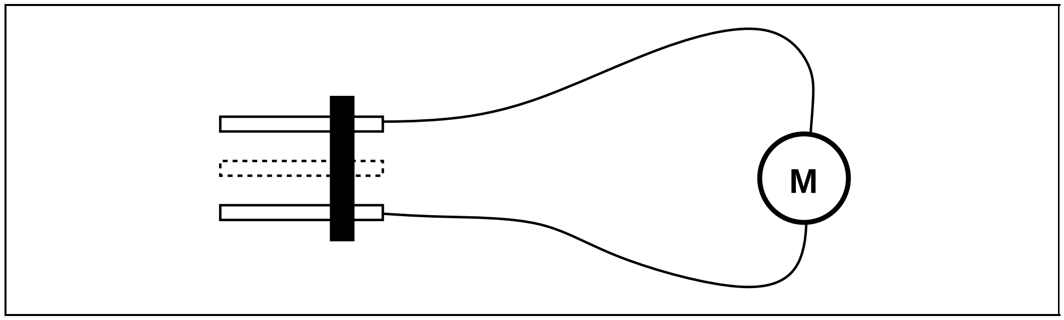


Figure 3: Motor Wiring Diagram

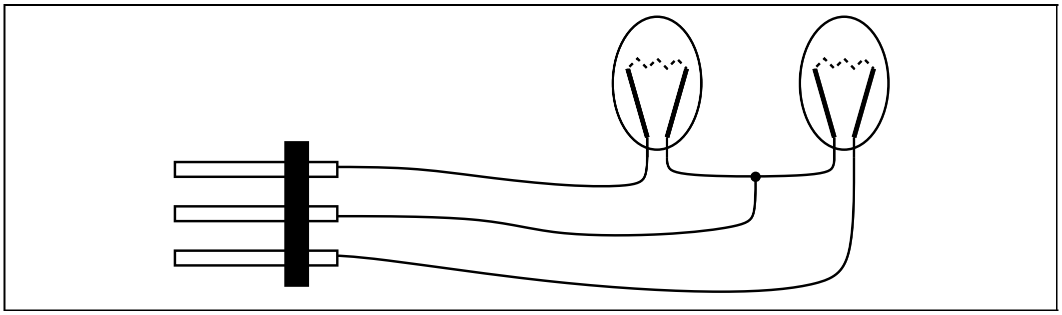


Figure 4: Wiring Two Lamps to a Motor Port

shown on the diagram because the signal wires are connected internally between the 6811 and the L293D chips rather than being broken out to a connector header.

1.3 Switches

1.3.1 Reset Switch

The pushbutton switch labelled *RESET* will reset the microprocessor. The board also has a "power-on reset," meaning that whenever power is first supplied to the board it will automatically reset.

1.3.2 Interrupt Request Switch

The switch labelled *IRQ* sends an interrupt signal to the board; this halts program operation for the duration that the switch is held down. The switch release can then be detected by software. The C function `button()` returns true (1) if the IRQ button has been depressed since the last call to `button()` and false (0) if not.

1.3.3 Run/Download Switch

The slide switch is used to select one of two operating modes when the microprocessor is reset. If the switch is to the left, the microprocessor will reset into "run mode" and execute the program that is resident in its internal memory. If the switch is to the right, the microprocessor will reset into a "bootstrap download" mode in which a program is read in over the serial line and executed. This mode is used to download a new program into the microprocessor's internal memory.

1.4 Analog Input Port

Eight analog inputs, numbered 0 through 7, are located in the middle of the top edge of the Mini Board. The C function `analog(int port)` returns a number from 0 to 255 indicating the voltage (0 to 5 volts) sensed on the desired analog input.

All sensor inputs, including the analog inputs, use a standard connector plug format. This connector is a four pin header with one middle pin removed, leaving a polarized three wire connector (see Figure 5). In addition to signal and ground, a power lead is provided, allowing the operation of active sensors that require a power source.

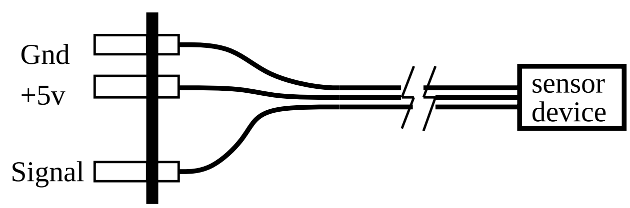


Figure 5: Generic Sensor Wiring

The analog input port is connected to the 6811's *Port E* (the analog inputs). 47K pullup resistors are present between each input pin and +5 volts, allowing most resistive devices to be wired between signal and ground with no additional circuitry needed.

Input 0 of the analog port can be used to implement battery level detection by installing resistors R1 and R2. This is explained in Section 4.2.

1.5 Digital Input Port

Eight digital inputs are located vertically on the left half of the Mini Board. These are numbered from 0 (top) to 7 (bottom).

The C function `digital(int port)` is used to return the value of a digital input. The function returns the logical inverse of the actual value detected by the hardware. This is because the pins are normally high and go low when the input is connected to ground (as by the action of a pushbutton switch). This is the reverse of the normal logical sense: one would like the switch to read "true" when it is depressed, not when it is released. Therefore the `digital()` function inverts the hardware reading, returning true or 1 when the input is at zero volts (as when the switch connects the input to ground).

The digital input port is connected to the 6811's *Port C*. Though they default to act as inputs, each of the pins of Port C may individually be configured as digital outputs under software control.

1.6 Timer and Counter Port

An eight-pin port located along the bottom edge of the Mini Board is wired to the 6811 *Port A*, a collection of special-function timer input and counter output pins. These pins may be used to generate rectangular waves or sample the period of incoming signals.

Pin 3 of this port is also connected to the piezo beeper connection labelled *PZO*. Audible tones may be generated on this pin using the C function `tone(int f, int d)`. The frequency `f` is specified in Hertz and the duration `d` is specified in milliseconds.

Because the use of these pins is generally for special purpose applications, no other C functions have been written for using the pins of this port. Instead, these pins may be accessed using memory reads and writes to the 6811 internal register bank; more information, consult Motorola documentation.

1.7 Status LEDs

Two LEDs located in near the lower left corner of the Mini Board provide helpful status information.

A green LED, labelled *RCV*, lights when the Mini Board is connected to a host computer's serial line. This indicates proper connection from the transmit line of the host computer to the receive line of the Mini Board. When the host transmits information, the RCV LED blinks off and on.

A red LED, labelled *PWR*, lights when the Mini Board is powered on and operating normally. It flashes off when the Mini Board transmits serial information to its host.

When the Mini Board is reset into the special bootstrap mode, the red LED will be completely off.

1.8 RS-232 Serial Port

The lower of the three RJ-11 jacks is the RS-232 compatible serial port for connecting the Mini Board to a host computer. Figure 6 shows the pin-out of this connector as viewed looking into the RJ-11 connector.

If the functionality of the high speed serial bus (see following) is not required, the Mini Board can be shortened so as to remove the board area required for

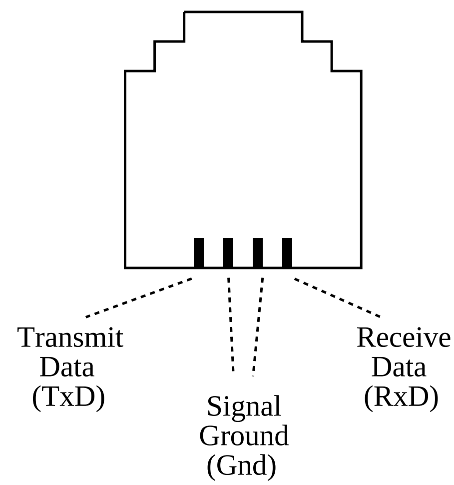


Figure 6: Mini Board 2.0 RJ-11 Serial Jack

mounting J2, the triple RJ-11 connector. In this case, an RS-232 RJ-11 jack can be wired to the four holes immediately above the PWR LED labelled *TxD*, *RxD*, and *GND*.

1.9 Serial Peripheral Interface Jacks

The upper two of the three RJ-11 jacks are connected to the Serial Peripheral Interface (SPI), a circuit built-in to the microprocessor that allows Mini Boards to be networked together. The two jacks are connected in parallel to the same internal hardware; Mini Boards would be connected in a daisy chain fashion (Figure 7).

Use of the SPI circuit is explained in Appendix Section D.4.

2 The Micro-C Compiler

The Mini Board can be programmed using the Micro-C C compiler for MS-DOS sold by Dunfield Development Systems, Inc.[1] To use the C compiler, first a C program is created using any text editor. Then, this program is compiled with Micro-C. The compilation involves several steps—compiling, linking, and assembly—but it can be done by invoking a single program. Finally, the resulting

[1]Information on obtaining the Dunfield C compiler can be found in Appendix E, “Suppliers.”

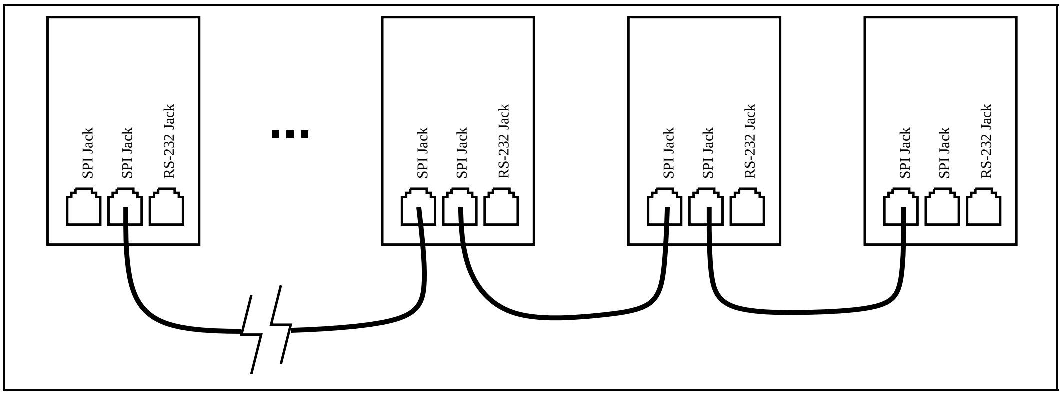


Figure 7: Sample Daisy Chain Configuration

binary code is downloaded to the Mini Board using the serial port of the host computer.

The Micro-C program `CC11` is the one-step compile, link, and assemble command. The individual programs that are invoked by this one are `MCC11`, the C compiler, `SLINK`, the source code linker (this links the compiled C code to a standard set of library files), and `ASM11`, a 6811 assembler program.

The program `DLM` (supplied by the author) is used to download the resulting binary file (with `.HEX` suffix) to the Mini Board.

2.1 Writing C Programs

Every C program must have a routine named `main`. This routine is called when the Mini Board is reset. `main` should never exit; instead, if the program wishes to terminate, it should end in an infinite loop. If `main` were to exit, the microprocessor would start executing garbage code.

A set of library routines callable as standard C functions are provided for interacting with the various features of the Mini Board. For example, the C function

```
motor(1, 16);
```

would turn Motor 1 on at power 16 (full power in the "forward" direction). Here's another Micro-C example using a sensor function:

```
if (analog(3) > 100) {off(1); off(3);}
```

When this code fragment executes, if the value of analog input 3 is greater than 100, the Motors 1 and 3 are turned off.

The Micro-C compiler uses K&R syntax rather than ANSI syntax for function and argument declaration. All functions are assumed to be of type `int`, so function declaration is not strictly necessary. Most C control structures including `if–else`, `while`, `for`, and `break` are supported. Arrays and structures are supported but floating point numbers are not. For full information, see the Micro-C manual on the Micro-C distribution diskettes.

> *NOTE: Micro-C must be obtained from Dunfield Development Systems, Inc. The software provided with this board does not include the Micro-C compiler system; only libraries for use with Micro-C and the code downloader may be obtained from this author.*
>
> *The preferred method for obtaining the free software is via anonymous FTP to the address 18.85.0.47 (*`cherupakha.media.mit.edu`*). See Appendix G for more details.*

When programming the Mini Board, keep in mind that the biggest constraint on the code is the overall code size. The 68HC811E2FN microprocessor has only 2048 bytes of program memory (EEPROM). Approximately 700 bytes are taken up by the code libraries. The rest is available for user programs.

2.2 Mini Board Micro-C Library

This section describes the standard C functions provided in the Micro-C library for the Mini Board. When Micro-C is properly installed (see following section), these functions will automatically link into any program compiled with the Micro-C compiler.

`analog(int port)`

> Returns value on analog input `port` as a number from 0 (corresponding to a 0 volt input) to 255 (5 volt input). Ports are numbered from 0 to 7 as labelled on the Mini Board.

`digital(int port)`

> Returns 1 or 0 (true or false) value from 6811 Port C. Open-circuit reading is 5 volts due to pull-up resistor present on Port C pins, and is reported as false (0) by the `digital` function.

`motor(int m, int s)`

Sets motor `m` at speed `s`. Motors are numbered from 1 to 4; speeds are numbered from -16 (full backward) to $+16$ (full forward).

`off(int m)`

Turns off motor `m`. Functionally equivalent to `motor(m, 0)` but uses less program storage space.

`button()`

Returns 1 (true) if IRQ button has been pressed and released since the last call to the `button` routine; returns false (0) otherwise. While IRQ button is held down, system will hang.

Button presses are automatically debounced.

`msleep(int msec)`

Performs a dead loop for `msec` milliseconds.

`tone(int freq, int duration)`

Generates a beep of frequency `freq` Hertz for a period of `duration` milliseconds on the designated piezo pin (Port A, bit 3). Frequencies are rounded off to the nearest 16 Hertz.

`printdec(int num)`

Prints `num` as a decimal number over the Mini Board's serial port. This is useful for debugging when a terminal emulator program is running on the host PC that the Mini Board is connected to: variable values can be printed over the serial line and displayed on the terminal screen. (The host emulation should be set to 9600, no parity, and one stop bit.)

`getch()`

Waits until a single character is received over the serial line, and then returns its value. A one-character queue is supported by the 6811 hardware. Serial port settings are the same as indicated in `printdec`.

`time`

> `time` is an integer C variable that keeps track of time passage in milliseconds. Upon board reset, this variable is set to 0 and begins counting 1000 counts per second. The value of this variable may be read or written at any point during program execution.
>
> For reasons of implementation efficiency, this timing feature is implemented as a variable declaration, rather than as a function call. To enable access to this variable, use the declaration "`extern int time;`" at the beginning of the C program.

`rand(int limit)`

> Returns a pseudo-random number modulo `limit`. This function is supplied in the Dunfield 6811 libraries.

2.3 Using the Code Downloader

`DLM`, the *D*own*L*oader for the *M*ini board, is used to download either the `.HEX` file produced by the Dunfield compiler or the `.S19` file produced by the Motorola assembler (more on this assembler in the section on `HEXMON`).

> *The instructions that follow assume use of the MS-DOS version of the software. Instructions that explain use of the Macintosh version are included with the distribution of the software itself.*

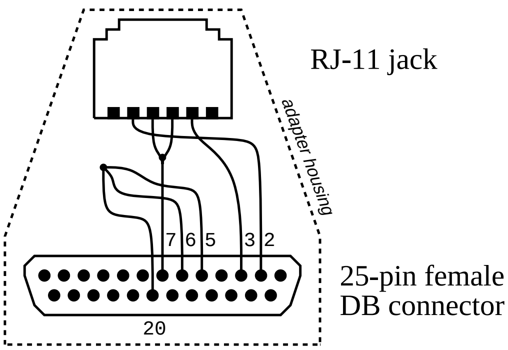
To use `DLM`, the Mini Board must be plugged in to the serial port of your computer. Standard 4-wire phone cable can be used with the appropriate modular-to-DB adapter. Refer to Figure 8, wiring instructions for an appropriate adapter for your computer.

Through the adapter, connect the Mini Board's RS-232 jack to your computer's serial port (either COM1 or COM2). If things are wired properly, the green serial receive LED (labelled "RCV") on the Mini Board will be lit. If this LED does not come on, check the wiring of the adapter and/or the connecting cable.

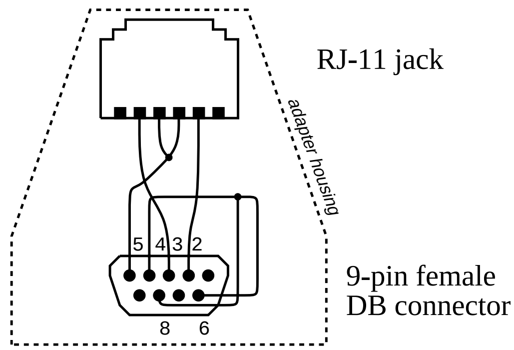
Flip the Mini Board's run/download switch into the download position (the position away from the 6811). Power on the board and press the reset button. *The red LED (labelled "PWR") should turn on dimly while the reset button is held down, and then turn off and stay off when it is released.*

If the red LED is now off, the board is in "download mode," ready to download new code. *If the red LED is on either fully or dimly, the board is not in download*

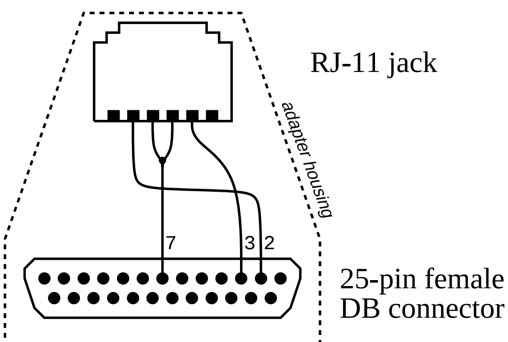
WIRING INSTRUCTIONS: RJ-11 jack is viewed peering into the jack.
If you have a 6-pin RJ-11 jack, the two outer pins are not wired.



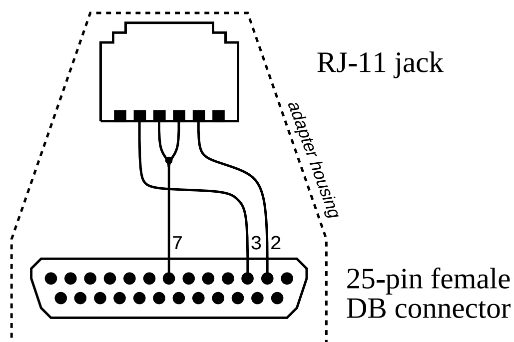
for IBM-XT style computer



for IBM-AT style computer



for Macintosh computer (with modem cable)



for Macintosh computer (with printer cable)

Figure 8: Wiring Diagram for Building Modular Jack Adapter

mode, and an attempt to download will surely fail. Check connections and switch settings and try again. *Note: the board will only go into download mode when it is reset* while hooked up *to a computer.*

When the board is waiting in download mode, simply type `DLM` with the name of the program you would like to download to the MS-DOS prompt. For example, to download a file named `FOO.HEX`, type the following that is underlined:

```
C:> DLM FOO
```

If you are using COM2, you must tell `DLM` to use that serial port:

```
C:> DLM FOO -PORT COM2
```

The `DLM` program will now commence downloading. This will proceed in two stages. First, a "bootstrap" program will be loaded into the 6811's internal RAM. During this stage a dot (".") will be printed as each byte is downloaded. Then this program will read the `.HEX` program into 6811 program memory. During this stage a dot will be printed for about every 16 bytes. Figure 9 shows a transcript of a normal downloading session.

```
D:\C>dlm car
DLM: 6811 File Downloader with Intelligent EEPROM burn by Fred
Martin
Version 2.0 Apr 13 1992 12:58:30
Downloading car.hex, press ESC to abort
Downloading eeprom loader to RAM at 1200 baud...
................................
................................
................................
................................
................................
................................
................................
........________________________
256 total bytes (boot loader done)
Synchronizing with board....OK
Sending car.hex at 9600 baud
.........................................................................
User program downloaded successfully.
```

Figure 9: Transcript of Typical Downloading Session

As you write and download programs, you may notice that during some portions of the download proceed faster than others. This occurs when the downloaded code matches the code already in memory: no changes need to take place, and the download can proceed to the next byte with no EEPROM "burn time."

This is particularly noticeable when downloading compiled code, because a good portion of this code is library code, which does not change between changes to the program code.

If you are downloading a program that has many changes from the code in memory, you may wish to use the bulk erase option. This option erases the entire memory in one step before downloading the program and is faster when many bytes have been changed. Use this option as follows:

```
C:> DLM FOO -BE
```

For a full list of `DLM` options, type "`DLM`" at the MS-DOS prompt with no arguments.

2.4 Installing Micro-C

This section explains how to install Micro-C and supporting software onto your hard drive. The software system for programming on the Mini Board consists of the following:

- *Micro-C executables.* These are the compiler, linker, and assembler. These programs are purchased from Dunfield Development Systems as mentioned earlier.
- *Micro-C libraries.* A custom set of library files, supplied by the author, is used with Micro-C to implement various functions like `motor()`, `analog()`, `tone()`, etc.
- *The downloader.* This program, supplied by the author, is used to download compiled programs to the Mini Board.

2.4.1 The Micro-C Directory

Follow the installation instructions included with your Micro-C distribution diskettes. You will create a directory named `MC` at the top level of your hard drive. Into this directory, the following files will reside:

`CC11.COM` This program is the command coordinator that invokes the entire compile/optimize/link/assemble operation.

MCC11.COM This is the C compiler program. It takes a C source file and produces assembly language output.

MCO11.COM This is the optimizer program. It takes compiled C code (i.e., assembly code) and removes redundancies.

SLINK.COM This is the source code linker. It links the assembled code to the library routines, places globals in RAM, and performs other related functions.

ASM11.COM This is the 6811 assembler. It takes the linked output from SLINK and produces a binary object file (a .HEX file).

MCP.COM This is the optional C code preprocessor.

MAKE.COM, SCONVERT.COM, SINDEX.COM, SLIB.COM, TOUCH.COM These programs are used to maintain the library files.

From the disk supplied by the author, copy the following files into the same MC directory:

DLM.EXE This is the code downloader. It takes a binary .HEX file and downloads it to the Mini Board's internal program memory.

CCDL.BAT This short batch file compiles and downloads a program with one command.

AS11.EXE This is a free 6811 assembler program distributed by Motorola.

Now that you have installed all of the required executables, make sure to add the MC directory to your execution path.

2.4.2 The Library Files Directory

From *within* the MC directory, a directory named LIB11 will be created. Install the following files, supplied by the author, to this directory:

EXTINDEX.LIB Before installing this file, make a backup of the file by this same name that already exists in the \MC\LIB11\ directory (copy it to another name). Now install the file from this author's disk.

This file is used by SLINK to organize the linking process.

6811RL.ASM The run-time library used by Micro-C.

`6811ST.ASM` The initialization (or STart) file linked into every compile.

`6811RST.ASM` The concluding, or ReST, file linked at the end of every compile.

`6811ZP.ASM` The file containing Zero Page variables used by the Mini Board libraries.

`6811REGS.ASM` This file declares the location of standard 6811 registers.

`MINILIB.PSC, MINILIB.SRC, MINILIB.ASM, MINILIB.FCN` These files are the main code libraries for the Mini Board.

`SERIAL.SRC, SERIAL.ASM, SERIAL.FCN` This library implements the functions for serial line communication.

`MAKEFILE` This file coordinates updates to the other library files.

2.5 Configuring the Library Files

The `MAKEFILE` file in the `LIB11` directory controls the configuration of the Mini Board library files. By setting flags on the commands in this file, several optional features may be enabled:

- Speaker output through Motor port 4
- Shaft encoder support

2.5.1 Speaker Output Through Motor Port 4

This feature causes any `tone` command to be mirrored through motor port 4. By hooking a small speaker to the motor 4 port, loud tones will be generated. (Motor port 4 can no longer be used to control motors when this feature is enabled).

To enable this feature, find the following lines of the `MAKEFILE` file:

```
6811st.asm : 6811st.src
    $mcdir\\mcp 6811st.src 6811st.asm
6811zp.asm : 6811zp.src
    $mcdir\\mcp 6811zp.src 6811zp.asm
minilib.src : minilib.psc
    $mcdir\\mcp minilib.psc minilib.src
```

Then, change these lines to appear as follows:

```
6811st.asm : 6811st.src
    $mcdir\\mcp 6811st.src 6811st.asm MOTOR_BEEP=
6811zp.asm : 6811zp.src
    $mcdir\\mcp 6811zp.src 6811zp.asm MOTOR_BEEP=
minilib.src : minilib.psc
    $mcdir\\mcp minilib.psc minilib.src MOTOR_BEEP=
```

Now, from within the `LIB11` directory, type the command `MAKE`. This runs the Micro-C utility `MAKE.COM`, which will rebuild the Mini Board library files to incorporate this feature.

At this point, Micro-C programs that use the `tone` command can be recompiled; the new final code will cause and sounds to be reproduced on motor port 4.

To disable the feature, remove the flags `MOTOR_BEEP=` from `MAKEFILE` and re-run the `MAKE` command.

2.5.2 Shaft Encoder Support

The Mini Board library files can support two shaft encoder type sensors. Interrupt-driven software will keep a 16-bit running count of transitions on these sensor ports.

To enable the shaft encoder operation, the sensor ports and sensor thresholds must be first hard-coded into the file named `6811ST.SRC`. The following section from this file shows how this is done:

```
#ifdef ENCODER
* equates for shaft encoder sensor ports
LTENCDR EQU     0
RTENCDR EQU     1

ENCDRLO EQU     25
ENCDRHI EQU     40
#endif
```

The symbols `LTENCDR` and `RTENCDR` are defined to the analog sensor ports to be used with the left and right encoders, respectively. In the example, they are defined for ports 0 and 1, respectively.

The symbols `ENCDRLO` and `ENCDRHI` are defined to be the lower limit and upper limit of the shaft encoder sensor. When the sensor reading falls below the lower limit, the encoder count is incremented; when the reading rises above the upper limit, the count is incremented again. After rising above the upper limit, the sensor reading must fall beneath the lower limit before a subsequent count will be recorded (and vice-versa).

After editing 6811ST.SRC, edit MAKEFILE to add the flag ENCODER= to the same three lines as indicated for the motor port tone sequence. Run the MAKE command, and new versions of the library files supporting shaft encoders will be created.

To access the shaft encoder values, add the following declarations to the beginning of a C program:

```
extern int lt_count;
extern int rt_count;
```

The C variables lt_count and rt_count will keep track of the shaft encoder sensors plugged into analog ports LTENCDR and RTENCDR, respectively.

3 The HEXMON Monitor Program

Instead of writing programs in Micro-C, you may run the HEXMON program on the Mini Board. Then, you can control motors and sensors via direct commands to the Mini Board's serial port.

HEXMON is a monitor program for the Mini Board written in 6811 assembly language. It provides services and an interaction protocol for control of motors and sensors over a serial line. This allows the Mini Board to act as a motor/sensor controller that can be operated by any computer with a serial port.

The current revision of the HEXMON program is version 4.0. This version provides the following features:

- Pulse-width modulation speed control of the Mini Board's four motor ports to 16 levels of speed in either direction.
- Reading of eight analog inputs.
- Internal timer accurate to one millisecond takes timestamps when any analog input exceeds or falls below a preset value.
- Peek and poke access to memory allows interaction with 6811 registers for further functionality.

The HEXMON program is so named because it uses an ASCII hexadecimal-based interaction protocol. This protocol was designed to allow a person to directly type commands to the program using a terminal emulator, if need be.

3.1 Overview

HEXMON communicates at 9600 baud, no parity, one stop bit. The Mini Board's serial line hardware provides an automatic echo of characters it receives; therfore any character sent to the Mini Board is immediated echoed back to the host. The HEXMON program does not provide an additional echo except when a complete command is received, as is explained following.

Upon reset, HEXMON sends its prompt character "$>$" (ASCII 62) over the serial line. At this point it is ready to accept a command.

Commands have the following format:

<command-character>
Data-3
Data-2
Data-1
Data-0
Addr-3
Addr-2
Addr-1
Addr-0

Each of these items is a single ASCII character, having the following function:

command-character is a lower-case letter indicating the command being issued.

Data-3 through *Data-0* are four hexadecimal digits forming a 2-byte data value. Depending on the nature of the command, both bytes, the lower byte, or neither are used.

Addr-3 through *Addr-0* are four hexadecimal digits forming a 2-byte address value. The whole address word is normally used, with *Addr-3* being the most significant digit and *Addr-0* being the least.

After receiving a complete command, HEXMON processes it and then sends back the command character as confirmation. If there is a result to be returned, it is sent next. Following this, the prompt character is sent, indicating that HEXMON is ready for another command.

The following table lists the set of commands supported:

Read Byte Command character: `r`. Two hexadecimal digits indicating the value of the byte at the address specified by the address digits are returned. The data digits are ignored.

Write Byte Command character: w. The byte formed by the two lower data digits is written to the memory location specified by the four address digits. No value is returned.

Read Word Command character: q. The word read from the memory location specified by the four address digits is returned.

The most significant byte is read from the address specified and the least significant byte is read from the following address (as per 6811 byte ordering standard.) The value is returned as four hexadecimal digits (most significant first).

Write Word Command character: z. The word formed by the four data digits is written to the memory location specified by the four address digits. No value is returned.

Reset/Synchronize Command character: s. The reset command is used to reset the HEXMON program to a known state. It also turns off all motors and otherwise initializes the program.

In order for the s command to be processed, the eight argument characters must be received. Hence the method to reset the board is to successively send single s characters, and wait for a double-s followed by the prompt to be returned (remember that the Mini Board hardware automatically provides a single character echo, so the second s/prompt will indicate that the reset command was actually processed).

3.2 Special Features

HEXMON performs a host of other services on a timed-interrupt basis, while also supporting serial line protocol just described.

These services are the following:

- Pulse-width modulation speed control of four DC motors.
- Sampling of all eight analog inputs.
- 32-bit real time counter incremented once per millisecond.
- Threshold detection of minimum or maximum analog input reading, with timestamp taken if reading falls below or exceeds threshold.

These services are interacted with by reads and writes to dedicated memory locations in the 6811's internal RAM memory. Figures 10 and 11 list the address and function of the memory mapped RAM locations.

Note: please do not confuse these memory mapped locations with the 6811 internal register bank (located from $1000 to $103F). This register bank is supported by hardware, while the following "registers" are supported only by the `HEXMON` *program.*

For example, writing a value to address $005D will have the effect of controlling motors if and only if *the HEXMON program is running. The actual motor hardware is connected to the 6811's Port B register at address $1004. The HEXMON program reads the byte stored at $005D to determine how to set motor state, and then writes the proper control information to address $1004.*

3.3 System Timer

The four bytes labelled "time" in Figure 10 are a free-running counter that counts at a rate of 1000 ticks per second (1000 Hertz). This timer may be used to time physical events. It is also used by the internal timestamping mechanism discussed below.

The timer may be reset by writing zeroes into its four bytes. To do this, it is recommended that the Write Word command be used in the following way:

1. Write $0000 to location $0002 (the less significant word).
2. Write $0000 to location $0000 (the more significant word).
3. Again write $0000 to location $0002.
4. The counter is now reset.

This method minimizes the possibility that the count can wrap and increment the higher word the during the interval between steps 2 and 3.

3.4 Motor Control

Motors are controlled in two ways:

- The *Motor Control Mask* (location $005D) determines whether a motor is on or off, and which direction it runs.
- Each motor's *PWM Bits* (Pulse-Width Modulation Bits) determine what portion of the time the motor is actually powered if it is on. If all of these bits are one, the motor runs at full power. If half of the bits are one, the motor runs at half power. If all of the bits are zero, the motor will stay off regardless of the state of its control mask.

Address (Hex)	Function	Comments
$0000	Time (highest byte)	4 bytes total.
$0001		
$0002		Incremented at
$0003	(lowest byte)	1000 Hertz.
$0004	Analog 0 minimum point	One byte each.
$0005	Analog 1	
$0006	Analog 2	
$0007	Analog 3	
$0008	Analog 4	
$0009	Analog 5	
$000A	Analog 6	
$000B	Analog 7	
$000C	Analog 0 maximum point	One byte each.
$000D	Analog 1	
$000E	Analog 2	
$000F	Analog 3	
$0010	Analog 4	
$0011	Analog 5	
$0012	Analog 6	
$0013	Analog 7	
$0014	Analog 0 min timestamp	4 bytes each.
$0018	Analog 1	
$001C	Analog 2	
$0020	Analog 3	
$0024	Analog 4	
$0028	Analog 5	
$002C	Analog 6	
$0030	Analog 7	

Figure 10: `HEXMON` Memory Mapped Control Locations, Page One

Address (Hex)	Function	Comments
$0034	Analog 0 max timestamp	4 bytes each.
$0038	Analog 1	
$003C	Analog 2	
$0040	Analog 3	
$0044	Analog 4	
$0048	Analog 5	
$004C	Analog 6	
$0050	Analog 7	
$0054	Analog 0 value	One byte each.
$0055	Analog 1 value	
$0056	Analog 2 value	Values are
$0057	Analog 3 value	updated at
$0058	Analog 4 value	1000 Hertz.
$0059	Analog 5 value	
$005A	Analog 6 value	
$005B	Analog 7 value	
$005C	Timestamp control mask	see notes
$005D	Motor control mask	see notes
$005E	Motor 1 PWM bits	2 bytes each.
$0060	Motor 2 PWM bits	1=on; 0=off
$0062	Motor 3 PWM bits	
$0064	Motor 4 PWM bits	see notes
$1003	Digital input bits	6811 register

Figure 11: HEXMON Memory Mapped Control Locations, Page Two

3.4.1 Motor Control Mask

The upper four bits of the Motor Control Mask determine whether a motor is on or off. The highest bit (bit 7) controls motor 4; bit 4 controls motor 1. If a bit is set, the motor is enabled to be on. If a bit is cleared, the motor will be off.

The lower four bits determine the direction a motor will run when on. Bit 3 controls motor 4's direction; bit 0 controls motor 1's direction. A direction bit being cleared corresponds to the motor direction in which the red LEDs are lighted; if the bit is set the green LEDs will light.

3.4.2 Motor PWM Bits

When a motor is enabled to be on, the 16-bit value formed by the motor's PWM bits determine what portion of the time the motor is actually powered.

System software cycles through the PWM bits at a rate of 1000 times per second. Each time the highest bit is tested. If this bit is set, the motor is powered for that $1/1000^{th}$ of a second (assuming the control bit indicated the motor should be on). If the bit is cleared, the motor is allowed to coast for that $1/1000^{th}$ of a second.

The motor itself performs a time-average of the power that is delivered to it. Thus a motor powered every other cycle will operate half as strongly as is the motor powered continuously. By varying the proportion of one's and zero's in the PWM bits, a motor's power level can thus be controlled.

The PWM Bits default to full power (all one's) when `HEXMON` is reset. To change the power level of a particular motor, simply write a value to its PWM bits with some proportion of 0's and 1's to determine the desired power level.

Figure 12 lists suggested bit patterns for power levels incrementing from full off to full on. The smoothness of the motor's operation is dependent upon the distribution of power in the PWM cycle; if the on-times and the off-times were clustered together, motors are likely to seem more jerky in operation. In the table, the recommended motor on-times and off-times are as evenly distributed as possible.

It is important that the PWM bits be changed with the Write Word command and not a sequence of two Write Byte commands. This is because the system software actually shifts the bits in place in the RAM locations. In between the execution of the two Write Byte commands, bits could shift out of the first byte and be written into the second, ready to be clobbered by the second write command. The simple solution is to use the Write Word command, which writes all 16 bits in one operation.

Power Level	Bit Pattern	Hexadecimal Value
0	0000000000000000	$0000
1	0000000000000001	$0001
2	0000000100000001	$0101
3	0000100001000001	$0841
4	0001000100010001	$1111
5	0001001001001001	$1249
6	0010100101001001	$2949
7	0010101010010101	$2A93
8	0101010101010101	$5555
9	1101010101010101	$D555
10	1101010111010101	$D5D5
11	1110110110110110	$EDB6
12	1110111011101110	$EEEE
13	1111011110111110	$F7BE
14	1111111011111110	$FEFE
15	1111111111111110	$FFFE
16	1111111111111111	$FFFF

Figure 12: Suggested Pulse Width Modulation Bit Patterns

3.5 Analog Inputs

The values of analog sensors can be determined by reading the values located at memory addresses $0054 to $005B, as indicated in Figure 11. These memory locations are updated with the current value of the respective sensor at a rate of 1000 updates per second.

3.6 Digital Inputs

The digital input bank is connected to the 6811 Port C register, and can be read by examining memory address $1003.

3.7 Timestamp and Thresholding

`HEXMON` can also check if an analog sensor exceeds a preset maximum value or falls below a preset minimum value. This is done by storing the thresholds in the maximum or minimum fields corresponding to the desired sensor, and then setting a bit in the *Timestamp Control Mask* (location $005C) (the high bit corresponds to the analog 7 input; the low bit, analog 0).

When the sensor reading either exceeds the maximum threshold or falls below the minimum threshold, the system timer value is copied into the sensor's timestamp location. In addition, the bit in the timestamp control mask is cleared, to indicate that either a maximum or minimum timestamp was taken.

By determining which timestamp was taken, it can be concluded that the sensor either exceeded the maximum or fell below the minimum threshold.

3.8 Creating `HEXMON`

`HEXMON` is written in 6811 assembly language; the current version of the source code is named `HEXMON40.ASM`. In order to create a downloadable file, the 6811 assembler program supplied by Motorola must be used (not the Dunfield assembler). This is because the source code was written for the Motorola assembler and is slightly incompatible with the Dunfield assembler.

The Motorola assembler program is named `AS11.EXE` (note that the Dunfield assembler is named `ASM11.COM`). It is free software and is available from Motorola's bulletin board (phone number 512–891–3733).

The output file of `AS11` has a `.S19` suffix, rather than the `.HEX` suffix produced by the Dunfield assembler. These two file formats are identical and the `DLM` downloader can download either one of them.

4 Assembly Instructions

This section describes how to construct the Mini Board. Basic soldering and electronic construction skills are assumed.

4.1 Main Assembly

As you go through the following steps, you may wish to check off each instruction as it is completed. These instructions propose a particular order that parts be installed onto the board. This ordering was designed to make the assembly as easy as possible.

Figure 16 is an enlargement of the component silkscreen that is printed on the Mini Board. Many values and markings will be more legible in this diagram than as actually printed on the board. Please refer to this diagram as often as necessary during assembly.

1–☐ Component Side vs. Solder Side. On the Mini Board, a component silkscreen of white printing aids in placing components properly. This silkscreen is on the *component side* of the board. This means that components are inserted down from this side of the board, and soldered from the unprinted *solder side* of the board.

2–☐ Resistor Packs. There are two basic styles of resistor packs: *common ground* and *isolated terminals*. The Mini Board uses a mixture of these two types.

The *common ground* type has a number of resistors that have one of their terminals tied to a common pin. The opposite pin for each resistor is connected to a separate pin of the package. These types must be installed in the proper orientation, because the common terminal pin must install into the board where it is expected.

The common ground position is indicated on the Mini Board with a band printed on the board where the common pin should go. On the resistor pack, the common pin is indicated with a thin band or dot.

The *isolated terminal* type is a set of electrically isolated resistors. Component orientation does not matter for this type of component because it is symettric from an orientation point of view.

It so happens that the Mini Board design uses one 1K $\times$ 3 isolated resistor pack and one 1K $\times$ 5 common ground pack (amongst others). Both of these resistor packs have six pins—thus making them nearly indistinguishable, since both also are of the same value. Yet, they are distinct devices and cannot be interchanged.

There are two ways to tell the difference between these two packs. The first is to use an ohmmeter and simply measure resistances to determine which is which. (This is much easier to do before the packs are installed.) The other way is to look for the following symbols: a "V" on the package indicates the isolated type, and an "E" on the package indicates the common ground type. *Please take care to install these devices properly, and to observe correct orientation for all common ground packs.*

Make sure resistor packs are mounted at right angles to the plane of the circuit board. A good soldering technique is to first solder the two end pins and then straighten the pack before soldering the rest of the pins.

◯ **RP1**–1KΩ×5, polarized, "E" marking
◯ **RP2**–1KΩ×4, non-polarized
◯ **RP3**–1KΩ×3, non-polarized, "V" marking
◯ **RP4**–47KΩ×7, polarized
◯ **RP5**–47KΩ×7, polarized
◯ **RP6**–47KΩ×9, polarized

3–☐ **Single Resistors.** Resistors **R3**, **R4**, and **R5** mount vertically while resistor **R6** mounts flat on the board. Note: if you have trouble discerning colors, please use an ohmmeter to confirm the proper values when installing.

◯ **R3**–10KΩ, *brown, black, orange* ◯ **R4**–10KΩ, *brown, black, orange*
◯ **R5**–100KΩ, *brown, black, yellow* ◯ **R6**–2.2MΩ, *red, red, green*

4–☐ **IC Sockets.** When soldering IC sockets, it is a good idea to first solder two diagonally opposite corners, check that the socket is fully flat to the board, and then proceed with the rest of the soldering.

◯ **U2 socket**–14 pin DIP

Orient notch in socket package above notch in component placement figure. This will serve as a reference when installing the IC into the socket.

◯ **U1 socket**–52 pin PLCC

It is imperative to *orient the notched corner of the PLCC socket above the notched corner of component placement figure.* The socket enforces an orientation on the installation of the IC; it is therefore necessary that the socket be installed correctly.

5–☐ Non-Polarized Capacitors. There are three 0.1μF capacitors on the board. These have not been labelled with part numbers on the board. Orientation does not matter when mounting these components.

◯ **C2**–0.1μF. Mount near **RP1**.
◯ **C4**–0.1μF. Mount near **XTAL**.
◯ **C5**–0.1μF. Mount near **U4**.

6–☐ XTAL, Ceramic Resonator. This is a small blue rectangular component labelled with the value "8.000 MHz." Orientation does not matter when installing this device.

7–☐ Q1, PNP Transistor. Install **Q1**, the 2N3905 or 2N3906 transistor, so that flat edge of device package is oriented above flat edge of component placement figure.

8–☐ LEDs. Please also note that the parts listing calls for the HLMP1700 series of LED (or equivalent). These LEDs are special low-current draw (2 mA) devices. In particular, **LED5** and **LED10** must be of this variety. Normal T1-sized LEDs may be substituted for the other LEDs if the value of **RP1** is changed to 330 or 470 ohms.

When installing LEDs, make sure that the shorter lead (the cathode) mounts into *shaded half* of component placement figure. If LEDs are installed backwards they will not work.

Note: **LED6** through **LED9** mount along the outside edge of the circuit board.

◯ **LED1**–red	◯ **LED2**–red	◯ **LED3**–red
◯ **LED4**–red	◯ **LED5**–red	◯ **LED6**–green
◯ **LED7**–green	◯ **LED8**–green	◯ **LED9**–green
◯ **LED10**–green		

9–☐ Switches.

◯ **SW1**–pushbutton ◯ **SW2**–pushbutton ◯ **SW3**–slide

Please note that the Digikey part number for **SW3** printed on the board is incorrect. The correct part number is *EG1903.*

10–☐ Direct Mount Integrated Circuits. U3 and **U4** are soldered directly to the board. When soldering, be careful not to apply too much heat. It is suggested that a cooling period be allowed after soldering half of each chip.

Make certain of correct orientation. The notch along a short edge of the IC package must be aligned above the notch in the component placement rectangle.

◯ **U3**–L293D ◯ **U4**–L293D

11–☐ Power Connector. J1 (Digikey part number ED1601) is a small rectangular terminal block with two screw-tightened connections. It is installed so that the wire slots face open over the outside edge of the circuit board.

In some copies of the Mini Board, the component holes for J1 may be drilled too small. If this is the case, it is suggested that the component leads of J1 be "whittled" down, using the sharp edge of a razor knife, until the component can be installed easily into its holes.

12–☐ Polarized Capacitors. The following two components are polarized: make sure to install the lead marked with a (+) into the hole marked (+) on the Mini Board. Depending on the capacitor manufacture, either the (+) lead or the (−) lead may be marked.

◯ **C1**–330μF, polarized ◯ **C3**–1μF, polarized

13–☐ U5, Power Regulator. The device labelled on the board is the *78L05.* This device would allow the Mini Board to be operated on voltages from 7 volts and up. A recommended substitution is the *LM2931Z-5.0.* This device allows the Mini Board to be operated on voltages as low as 5.6 volts.

Install so that flat edge of device package is oriented above flat edge of component placement figure.

14–☐ Inductors. Install upright, but mount so that the component bodies are not based in adjacent holes, to keep the two devices from touching each other.

◯ **L1**–1μH, 1 amp inductor ◯ **L2**–1μH, 1 amp inductor

15–☐ Strip Socket Header. Socket header comes in strips of 36 to 40 connections which must be cut into pieces of several lengths. Make sure to use *female socket header*, not *male pin header*.

To cut the socket header, two methods are suggested. The first is the use of a mitre box and an X-acto saw blade. If these tools are not available, a regular razor knife may be used. Score the header along the detent line at the place of desired separation. After scoring repeatedly on both sides of the header, the header will easily break at the desired line. *If it is attempted to break the header before it has been sufficiently scored, it is likely that one of the two end connections will be damaged.*

Cut seven 8-long pieces, two 9-long pieces, and two 12-long pieces. Install the headers as follows:

◯ **Analog Input Bank**–three 8-long pieces
◯ **Digital Input Bank**–two 8-long pieces, one 12-long
◯ **Port A Bank**–one 8-long piece, two 9-long
◯ **Motor Bank**–one 12-long piece

Note that the 12-long piece mounting in the Digital Input Bank extends to the additional serial connector, and the two 9-long pieces in the Port A Bank extended to the piezo connector.

16–☐ RJ11 Connector. Solder **J2** (Digikey part number H9117), the triple RJ11 jack into its location. Note that it will mount properly in only one orientation. Solder the metal mounting supports in addition to the electrical jack connections.

17–☐ Finish Assembly. Complete the assembly by looking over the connections, making sure that all joints are soldered well. Look for a shiny finish on the solder connections, not a dull or mottled finish. Retouch any connections as necessary.

Clip any excess component leads at this time.

4.2 Battery Level Indicator

A circuit may be installed on the board that allows monitoring of the battery level. To activate this circuit, two resistors must be installed and a circuit board trace be cut. This operation then feeds a ratio of the battery level into the Analog 0 input (analog 0 may not be used for an external sensor in this configuration).

Resistors **R1** and **R2** form a voltage divider that feeds a proportion of the input battery level before regulation to the analog 0 input. The suggested values printed on the circuit board are 10.0 KΩ for R1, and 20.0 KΩ for R2. This forms a voltage divider that passes one-third of the input voltage. The voltage to an analog input must never exceed five volts, so this choice of resistor values is appropriate for power supplies that will never exceed fifteen volts. Other values of resistors may be chosen for higher voltage ranges.

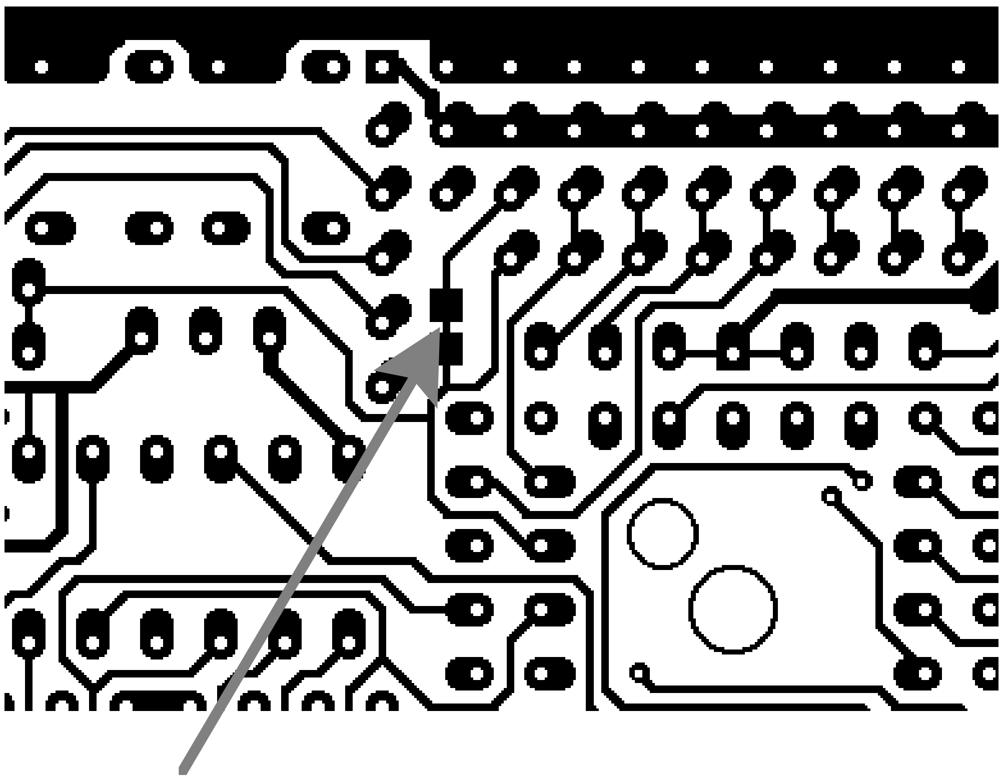
For best results, it is suggested to either use precision tolerance (1%) resistors, or hand-matched pairs of normal resistors.

After installing resistors R1 and R2, make the trace cut on the solder side of the board shown in Figure 13.

If it is later desired to disable the circuit, the two resistors can be removed and the trace can be jumpered with a solder patch across the two square pads on either side of the trace cut.

A Mini Board PC Layouts

Figures 14, 15, and 16 are double-sized reproductions of the component side, solder side, and silkscreen mask of the Mini Board 2.0. These images should be of sufficient quality to serve as artwork masters for PC board fabrication.



Cut trace between square pads after installing resistors R1 and R2 to enable battery level monitoring feature

Figure 13: Trace Cut on Solder Side of Board to Enable Battery Level Monitoring Feature

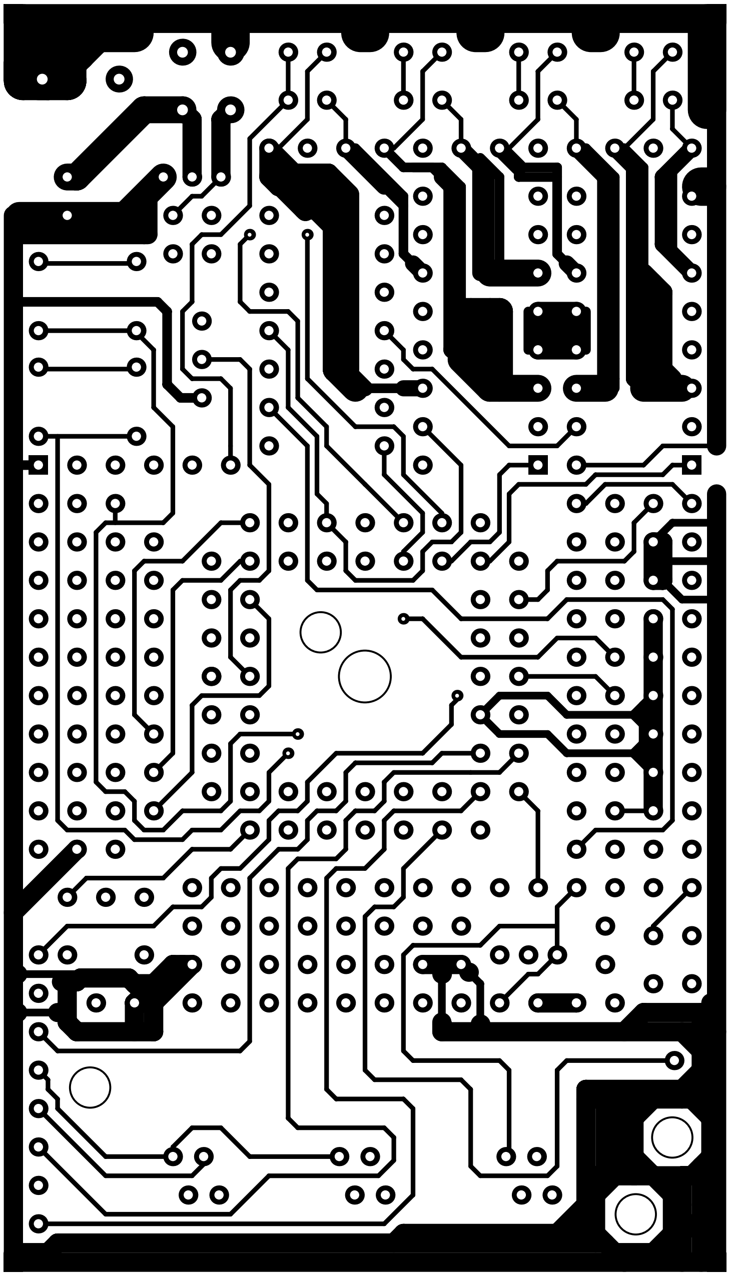


Figure 14: Mini Board 2.0, Component Side, 2:1 Enlargement

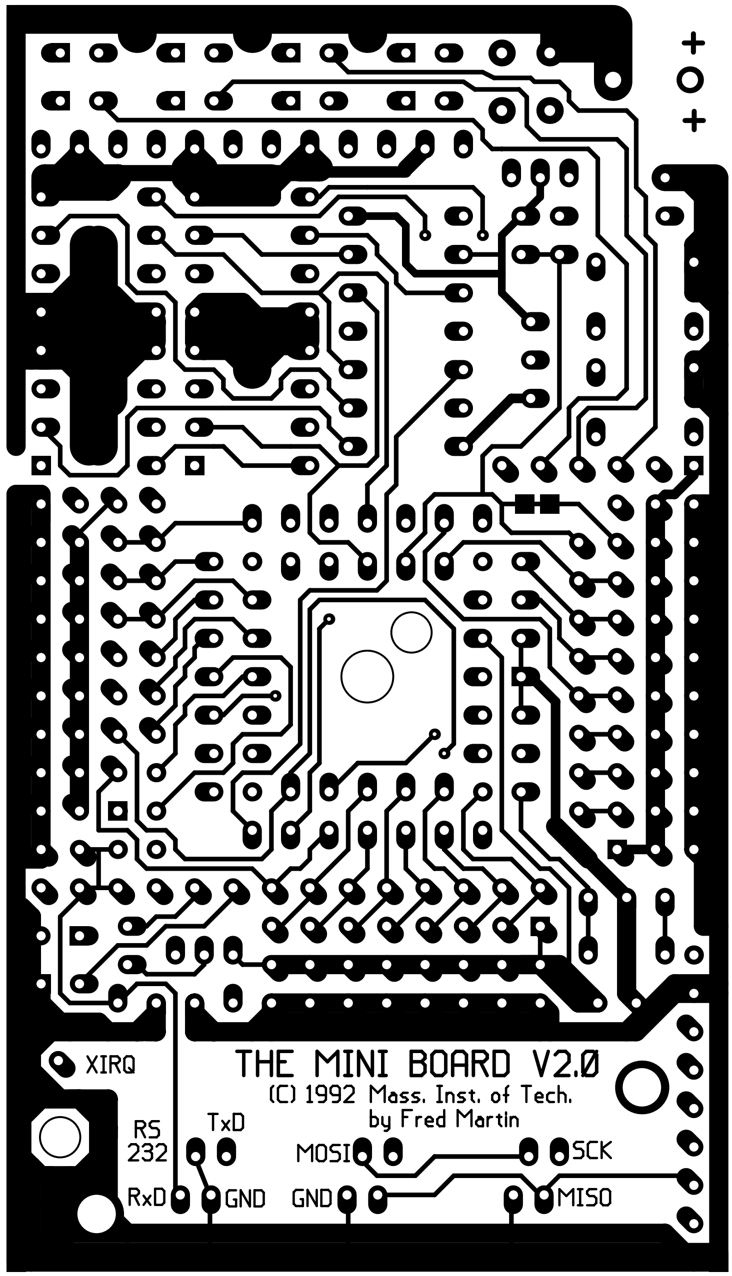


Figure 15: Mini Board 2.0, Solder Side, 2:1 Enlargement

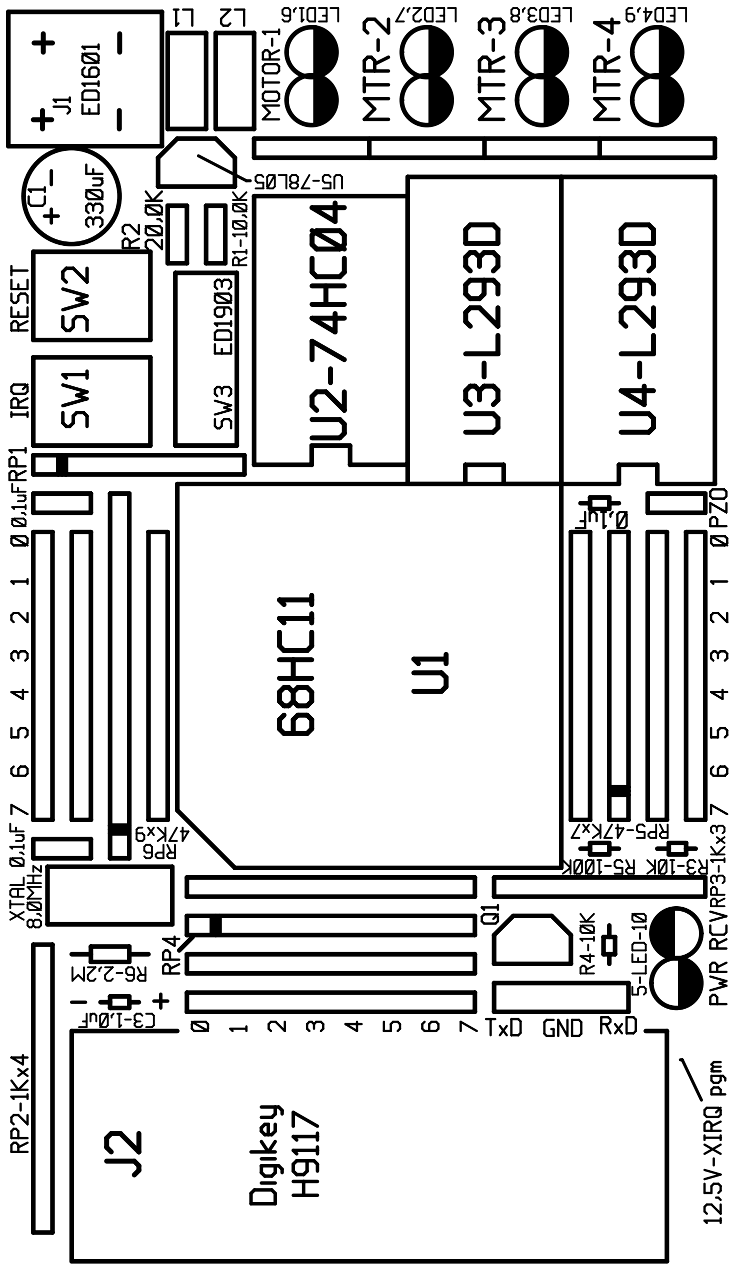
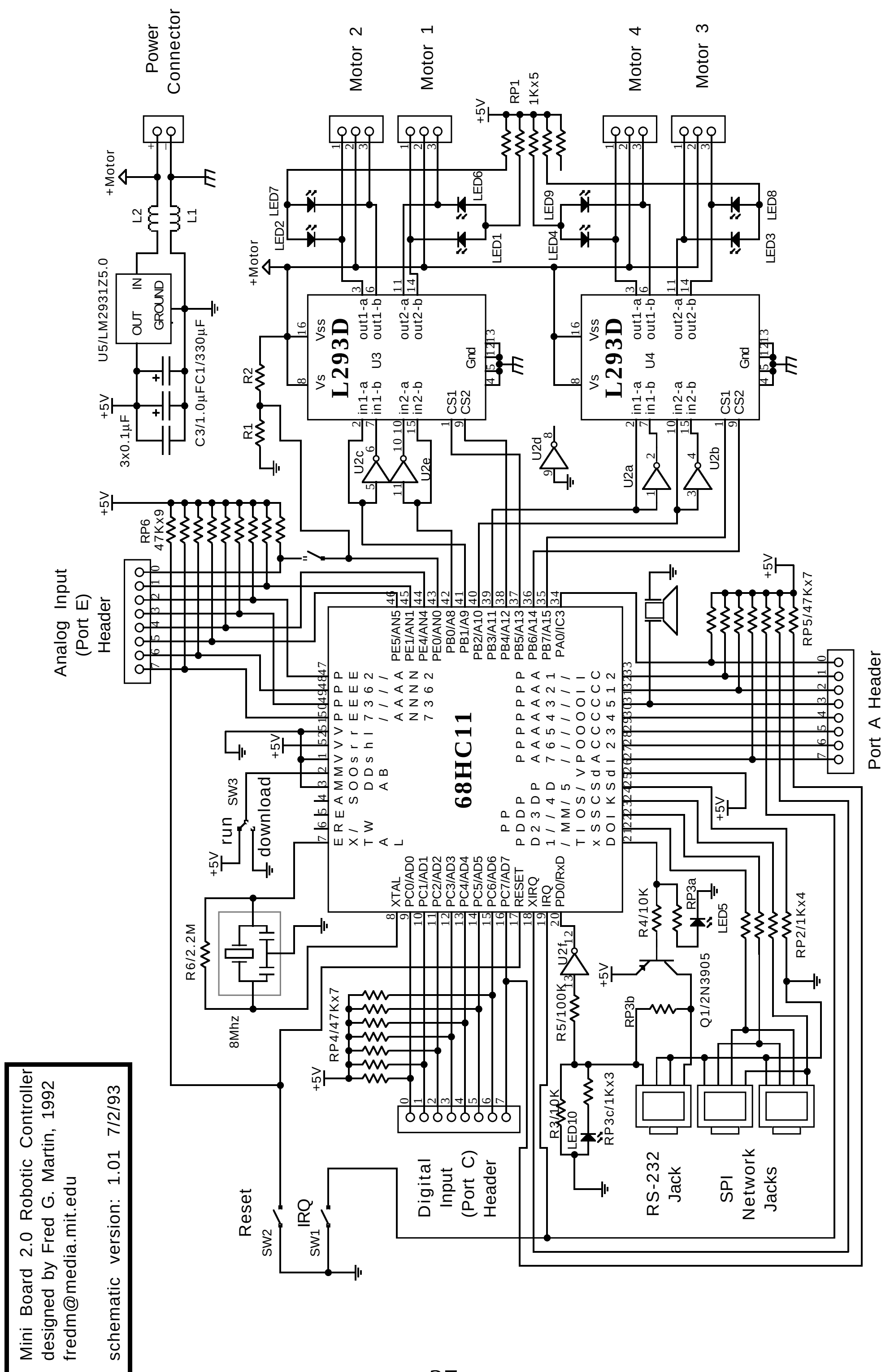


Figure 16: Mini Board 2.0, Component Side Silkscreen, 2:1 Enlargement

B Mini Board Schematic



C Mini Board Parts Listing

PART REF	QUA	DESCRIPTION	PRICE EA	PRICE TOT	CATALOG #	COMPANY	ALTERNATE #	ALTERNATE CO.	PRICE
		Total Price		**$60.19**					
		Integrated Circuits							
U1	1	MC68HC811E2FN 8-bit µproc	$20.00	$20.00		Active			
U2	1	74HC04 hex inverter	$0.13	$0.13	96-07404	EBC/ICC	MM74HC04N	Digikey	$0.28
U3, U4	2	SGS-Thomson L293D	$2.40	$4.80	L293D	Active	L293D	Arrow Elex	
U5	1	LM2931Z-5.0 regulator	$0.81	$0.81	LM2931Z-5.0	Digikey			
		SWITCHES							
SW1, SW2	2	Panasonic pushbutton switch	$0.18	$0.36	P8011S	Digikey			
SW3	1	SPDT miniature slide switch	$1.56	$1.56	EG1903-ND	Digikey			
		LEDs							
LED1-5	5	HLMP1700 high eff red LED	$0.10	$0.50	HLMP1700	R&D Elex	HLMP1700-ND	Digikey	$0.37
LED6-10	5	HLMP1790 high eff green LED	$0.15	$0.75	13503	Active	HLMP1790-ND	Digikey	$0.37
		Resistor Packs							
RP1	1	1K x 5 common terminal	$0.11	$0.11	20-15410	EBC/ICC	Q5102	Digikey	$0.16
RP2	1	1K x 4 isolated terminals	$0.14	$0.14	20-24410	EBC/ICC	Q4102	Digikey	$0.22
RP3	1	1K x 3 isolated terminals	$0.11	$0.11	20-23410	EBC/ICC	Q2102	Digikey	$0.16
RP4, RP5	2	47K x 7 common	$0.14	$0.28	20-17547	EBC/ICC	Q7473	Digikey	$0.22
RP6	1	47K x 9 common	$0.18	$0.18	20-19547	EBC/ICC	Q9473	Digikey	$0.27
		Individual Resistors							
R1	1	10.0K precision resistor	$0.10	$0.10	10.0X	Digikey			
R2	1	20.0K precision resistor	$0.10	$0.10	20.0X	Digikey			
R3, R4	2	10K, 1/8 w	$0.06	$0.11	10KE	Digikey			
R5	1	100K, 1/8 w	$0.06	$0.06	100KE	Digikey			
R6	1	2.2M, 1/8 w	$0.06	$0.06	2.2ME	Digikey			
		Capacitors							
C1	1	330 µF, 6.3v electrolytic	$0.14	$0.14	P6203	Digikey			
C3	1	1 µF tantalum	$0.13	$0.13	71-21036	EBC/ICC	P2105	Digikey	$0.21
C2,C4-5	3	0.1 µF monolithic	$0.19	$0.58	P4917	Digikey		Active	$0.08
		Connectors/Sockets							
J1	1	2 position terminal block	$0.34	$0.34	ED1601	Digikey			
J2	1	3 position RJ11 4 wire jacks	$4.77	$4.77	H9117	Digikey			
	2.3	36 position female header	$1.94	$4.53	929974-01-36-ND	Digikey			
	1	52 pin PLCC socket	$1.78	$1.78	A418-ND	Digikey			
	1	14 pin DIP socket	$0.07	$0.07	33-13144	EBC/ICC	A9214	Digikey	$0.18
		Miscellaneous							
XTAL	1	8 MHz ceramic resonator	$0.84	$0.84	PX800	Digikey			
L1, L2	2	high current inductor	$0.84	$1.68	M7010-ND	Digikey			
Q1	1	2N3905 PNP general purpose	$0.07	$0.07	52-03905	EBC/ICC	2N3905	Digikey	$0.23
	1	4 C cell battery holder	$0.75	$0.75	BH-4C	All Elex	270-390	Radio Shack	$1.49
	4	alkaline C cell	$0.75	$3.00					
	1	piezo beeper	$1.00	$1.00	PE-9	All Elex	273-073	Radio Shack	$1.49
	1	power switch	$0.33	$0.33	SSW-222	All Elex	275-401	Radio Shack	$0.60
	1	modular plug cable	$0.65	$0.65	MT-260	All Elex			
	1	DB-25 female to modular jack	$0.89	$0.89	AM1125F	C-Gate Elex			
	1	DB-9 female to modular jack	$0.89	$0.89	AM1109F	C-Gate Elex			
	1	9-wire rainbow ribbon cable	$0.26	$0.26	HC09M-ND	Digikey			
	2	3/32" heat shrink tubing	$0.56	$1.12	W182-ND	Digikey			
		Sensors							
	2	male strip header connector	$1.10	$2.20	929400-01-36-ND	Digikey	SHS-40	All Elex	$1.35
	2	reflective IR sensor	$0.50	$1.00	OSR-4	All Elex			
	2	phototransistor	$0.25	$0.50	PTS	All Elex			
	3	photocell	$0.25	$0.75	G990	Elex Goldmine			
	3	touch switch	$0.59	$1.77	14443	ADC			

D Mini Board Technical Notes

This appendix section presents some additional technical details about the Mini Board's operation.

D.1 Specifications

CATEGORY	DESCRIPTION	DATA
Microprocessor	Motorola 6811 series	
	68HC811E2FN	256 bytes RAM, 2048 bytes EEPROM
	68HC711E9FN	512 bytes RAM, 512 bytes EEPROM, 12 Kbytes EPROM
	others available	
External Memory	none supported	
Motor Outputs	4 motor ports	
	current capacity (each)	600 mA
	driver output states	"forward" direction, "backward" direction, off
Analog Inputs	8 bit resolution	8 inputs
Digital I/O	bidirectional: input or output	8 lines
Timer Port	programmable timers	4 or 5 outputs
	hardware counters	3 or 4 inputs
Power	supply required	5.6–36 volts DC
	current draw	80 mA (motors off)
	+5v output limit	80 mA
	connector	screw terminals
Communications	one RS-232 compatible	one RJ11 jack
	one Motorola serial peripheral interface (SPI)	two RJ11 jacks
Size	3.3×1.86 inches	

D.2 XIRQ Programming Voltage Input

At the lower edge of the board near the triple RJ11 jack is a pad labelled "*12.5V XIRQ pgm*" (this is easiest seen in Figure 16 on page 36). This pad may be used to supply the 12.5 volt programming voltage necessary when programming 6811's with EPROM (i.e., the 'HC711 series).

Please keep in mind that the `DLM` software discussed in this document is not capable of programming EPROM, even if the programming voltage were applied. `DLM` only works with EEPROM (in the 'HC811 series), which does not require a special programming voltage anyway. Motorola supplies MS-DOS software (named "PCBUG") which includes EPROM programming capability.

D.3 Serial Line Circuitry

This section details some idiosyncracies in the Mini Board's serial line circuitry.

Serial line voltages are normally $+12$ volts for logic low and -12 volts for logic high. Therefore some type of interface circuitry must be used for interfacing with TTL/CMOS internal logic levels, which are $+5$ volts for logic high and 0 volts for logic low.

There are a number of ways to obtain the voltages appropriate for a serial line. One way is to simply require an external power supply with $+$ and $-$ 12 volt lines. Another solution is to use a chip which can generate these voltages from a single $+5$ volt supply (the MAX232 chip by Maxim Corp. is an example). However, in a board as small as the Mini Board, even one extra chip is one chip too many, and the circuit shown in Figures 17 and 18 is used.[2]

D.3.1 Serial Output

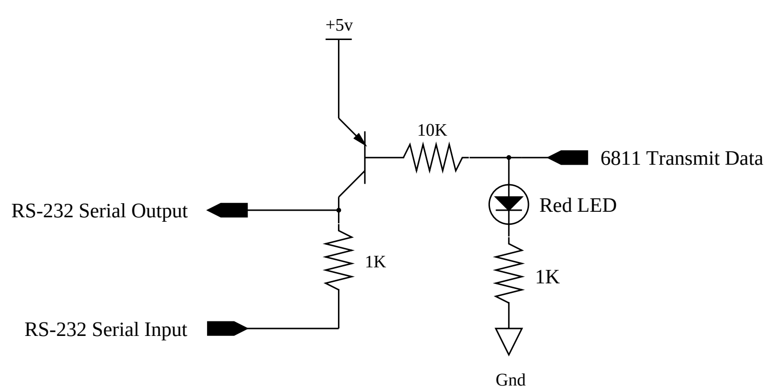


Figure 17: Mini Board Serial Line Transmit Circuit

[2]This circuit was originally conceived by Randy Sargent.

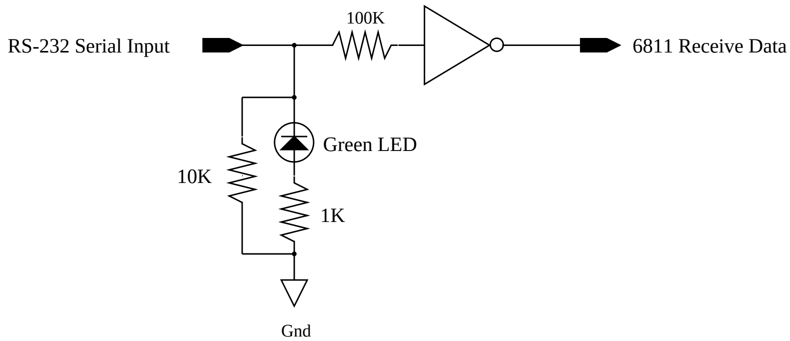


Figure 18: Mini Board Serial Line Receive Circuit

In the transmit circuit (Figure 17), a 1KΩ resistor is used to "loop back" a logic high signal from the host computer's own transmit line to its receive line. (Keep in mind that this is a -12 volt signal.) Because a relatively small resistive value is used, there is only a small voltage drop across the resistor, and the host computer is "fooled" into thinking it is receiving a valid logic high signal from the Mini Board, when really the voltage is coming from its own transmit line.

When the Mini Board needs to generate the logic low signal, the PNP transistor is turned on, which pulls the Mini Board's serial output (connected to the host's serial receive line) to $+5$ volts. This is quite obviously not up to the specification of $+12$ volts, but it happens that nearly all modern computers will interpret a $+5$ voltage as a logic low anyway.

D.3.2 Serial Input

To perform serial input, the Mini Board uses a spare CMOS inverter gate from the 'HC04 chip used in the motor control circuit to provide the necessary logic level inversion. Resistive coupling provides a sufficient load to the host's serial output and also protects the CMOS logic from excessive negative voltage (Figure 18).

D.3.3 Implications

There a couple of quirks associated with the use of this serial line interface. First, the Mini Board performs a "hardware echo" of any character that it receives, because of the loop-back resistor. Second, the Mini Board can't reliably transmit characters while the host is transmitting characters, because it needs the host's output to be at the -12 volt quiescent level in order to generate logic highs on its own output.

As long as these constraints are kept in mind, the Mini Board's serial circuit is quite functional and is compatible with nearly all types of computers.

D.4 Serial Peripheral Interface

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that is standard across many Motorola microprocessors and other peripheral chips. It provides support for a high bandwidth (1 megabaud) network connection amongst CPUs and other devices supporting the SPI.

The Mini Board's "SPI Network Jacks" allow multiple Mini Boards or other SPI devices to be connected using a four-wire RJ11 (telephone jack) connection. The Mini Board uses a particular configuration of the SPI system that requires a minimum of external hardware support. All that is needed to network Mini Boards or similarly configured other SPI circuits is a couple of resistors and wire. The resistors are located on the Mini Board itself, so multiple Mini Boards may be networked with no additional components other than the cabling itself.

It is beyond the scope of this document to provide a thorough introduction to the functionality of the SPI circuit. Instead, the reader should consult the 6811 literature mentioned in Appendix F. A brief review of the SPI system follows, along with a discussion of the particular configuration of the SPI on the Mini Board.

D.4.1 Introduction to the SPI

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPI's. During a data transfer, one SPI system acts as the "master" which controls the data flow, while the other system acts as the "slave" which has data shifted into and out of it by the master. Different CPU's can take turn being masters, and one master may simultaneously shift data into multiple slaves. However, only one slave may drive its output to write data back to the master at any given time.

The SPI system consists of two data lines and two control lines:

Master Out Slave In Abbreviated `MOSI`, this data line supplies the output data from the master which is shifted into the input(s) of the slave(s).

Master In Slave Out Abbreviated `MISO`, this data line supplies the output data from a slave to the input of the master. There may be no more than one slave which is transmitting data during any particular transfer.

Serial Clock Abbreviated SCLK, this control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates; the SCLK line cycles once for each bit that is transmitted.

Slave Select Abbreviated SS, this control line allows slaves to be turned on and off with hardware control.

D.4.2 Mini Board SPI Configuration

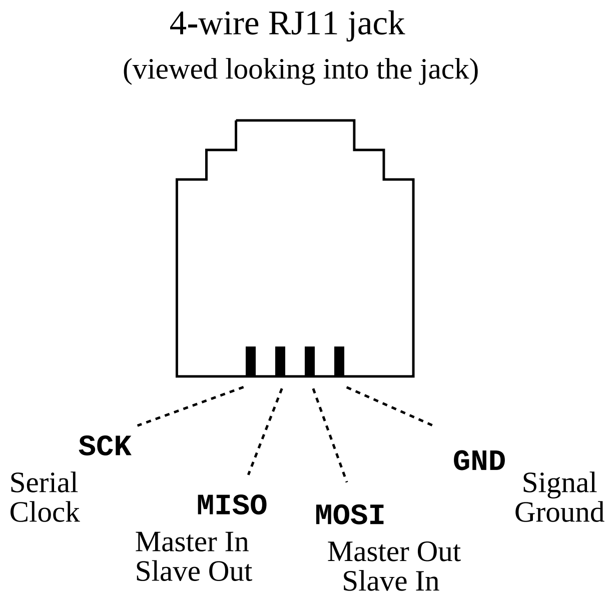


Figure 19: Serial Peripheral Interface Jack

The Mini Board uses a four-wire SPI connection that carries the MOSI, MISO, and SCLK lines in addition to a signal ground (see Figure 19). The Slave Select line is not used because use of this control line would require additional hardware. Instead, this line is wired to be always true (this line uses negative true logic, so it is wired to ground); software protocols must then be used to enable the activity of the SPI slave.

To connect the SPI circuits of multiple Mini Boards, each respective line of each Mini Board is tied together: MOSI to MOSI, MISO to MISO, etc. To make this happen, the connecting RJ11 cables must have half-twists in them. This configuration works because of the bi-directionality of each of the control lines. When a given SPI circuit is the master, it transmits on its MOSI line while slaves receive on their MOSI lines. Likewise a slave transmits on its MISO line and the master receives on its MISO line. The SCK line also is automatically configured properly (it is an output for the master and an input for the slave).

D.4.3 Suggested Protocols

Two sample protocols are presented. The former, a single master protocol, would be significantly easier to implement than the latter, a multiple master protocol.

Single Master Protocol. In this protocol, one CPU is always the master while all of the others are always slaves. One can imagine that the slave CPUs are simply input and output devices for the master.

The master might desire to exchange information with a particular slave. To do this, the master would send a slave address datum in each communication packet. The selected slave would then enable its MISO line to write information back to the master. All other slaves would be required to turn their MISO lines off so as to not conflict with the selected slave. This protocol could be extended to allow the master to broadcast a message to multiple slaves, but only one slave may respond at a time (otherwise there would be hardware contention on the MISO).

Multiple Master Protocol. This protocol allows CPUs to take turns being masters and request information from other CPUs at will. The trick here is to provide for some type of collision detection if more than one CPU decides to initiate a communication at the same time.

If the Slave Select input were being used, the 6811 would provide hardware support for collision detection. However, as previously explained, the Mini Board does not support hardware Slave Select. Thus, data integrity must be managed in software, presumably with a checksum or cyclic redundancy check (CRC) method.

At the time of this writing, a debugged and documented general purpose networking software package has not been prepared by this author. Its implementation is thus left to the reader as an exercise of practical import for those who desire to make use of the networking functionality. If anyone would like to contribute some debugged and documented networking software, please contact the author.

E Suppliers

This section provides information about electronic suppliers and other resources helpful in using the Mini Board technology.

Active Electronics

Several regional centers across the USA; call Telemarketing Department at (800) 677–8899

Leading retail dealer, oriented toward both small orders and large. Very good assortment of all necessities. Prices are sometimes high and sometimes very good. 33% discount on most items if $300 or more of any individual item is ordered.

All Electronics Corp

P.O. Box 567 • Van Nuys, CA 91408
orders only: (800) 826–5432
information: (818) 904–0524
fax: (818) 781–2653

Surplus dealer with good basic parts assortment. Many interesting surplus boards and assemblies, excellent motor selection, rechargeable batteries, switches, optoelectronics, etc. Prices very good.

Terms: $10 minimum order; MC/VISA/Discover accepted; flat shipping/handing charge of $3.50.

American Design Components

815 Fairview Avenue • POB 220 • Fairview, NJ 07022
orders and info: (800) 776–3700
local: (201) 941–5000
fax: (201) 941–7480

Surplus computer equipment, power supplies, lots of motors and batteries, switches, MOVIT robot kits. Overpriced basic parts assortment. Some other stuff very well priced.

Terms: $15 minimum order; MC/VISA/AMEX/ accepted; annoying shipping/handing charges of $3.00 plus 10% of cost of merchandise. Mentally add 10% to their catalog prices as you're going through their catalog and you won't be as annoyed by this excess "handling" charge.

Arrow Electronics

Catalog Division • 1860 Smithtown Avenue

Ronkonkoma, NY 11779
orders and info: (800) 93-ARROW

Huge retail dealer. Prices are high unless large quantities are involved. Excellent selection.

C-Gate International

3529 Ryder Street • Santa Clara, CA 95051
orders and info: (408) 730–0673
fax: (408) 730–0735

Computer accessories dealer. Unbelievably low prices on many items. How about a 6-outlet metal power strip for under $5 in single quantities? Or null modem adapters for $1.70? This is *the* ultra-cheap accessories warehouse. Look for their ad in *Computer Shopper* magazine.

Terms: no minimum order; MC/VISA accepted; you pay exact UPS shipping charge plus $2.95 handling charge per order.

Creative Learning Systems, Inc.

16510 Via Esprillo • San Diego, CA 92127–1708
orders and info: (800) 458–2880
fax: (619) 675–7707

This company publishes a wonderful, must-have catalog of educational technology titled "TransTech." It's loaded with books, software, and the wide assortment of building kits. Not only do they distribute LEGO Dacta products, but also other "universal" construction materials made by fischertechnik, Meccano, LASY, märklin, and Capsela, as well as a variety of dedicated kit products. They carry a variety of computer interface products as well.

Digi-Key Corporation

701 Brooks Ave. South • P.O. Box 677
Thief River Falls, MN 56701–0677
orders and info: (800) DIGI-KEY
fax: (218) 681–3380

Large retail parts and accessories assortment. Chips, displays, power supplies, cases, connectors. Extensive. Prices okay, better in quantity. The standard dealer for the hobbyist.

Terms: $5 handling charge for orders under $25; MC/VISA, check, money order or COD; customer pays shipping on credit card and PO orders. Increasing volume discounts for orders over $100.

Dunfield Development Systems, Inc.

P.O. Box 31044 • Nepean, Ontario K2B 8S8 (Canada)
Voice: (613) 256–5820 [Between 0800–1900 EST Please!]
Fax : (613) 256–5821 [0800–1900 EST Mon-Fri]
BBS : (613) 256–5821 [1900–0800 EST Mon-Fri, All day Sat-Sun]

David Dunfield is the developer of Micro-C. In addition to this product, he sells a number of software packages for embedded systems development.

The version of Micro-C used with the Mini Board is the *6811 Developer's Special*, priced at U.S.$99.95 plus $5.00 shipping and handling.

If you are planning to port Micro-C to a platform other than MS-DOS, inquire about the *Porting Package*, which includes source code to all of the Micro-C applications. This is available from Dunfield Development Systems for an additional $50 with the Micro-C purchase.

Call for more info.

Electronic Buyers' Club

1803 Northwest Lincoln Way
Toledo, OR 97391–1014
orders and info: (800) 325–0101

Organization with a $35 yearly membership fee. Unusual, but once you join you get incredibly low prices on basic electronics: semiconductors, ICs, resistors, caps, diodes, ceramic resonators, LEDs, more. Prices start low at single quantities and go down if you buy tens or hundreds. Beats any of the other retailers hands down for most of the parts that they stock.

Catalog includes many manufacturers' data sheets and is a reference work in and of itself. Quick delivery on stocked parts. If you will spend $100 or more on electronic supplies in the next year, the $35 fee will earn its price. Many parts sold are available at literally one-fourth the prices that Digikey (for example) advertises.

See also *International Components Corporation*, the twin of EBC with slightly higher prices but no membership fee.

Terms: No minimum order but $2 fee on orders less than $25; shipping charges are exact UPS rates.

The Electronic Goldmine

POB 5408 • Scottsdale, AZ 85261

orders and info: (602) 451–7454
fax: (602) 451–9495

Excellent catalog comprised mostly of packaged electronic kits, with a small amount of very good surplus stock mixed in.

Terms: $10 minimum order; MC/VISA accepted; minimum $3.50 for UPS shipping.

LEGO Dacta

555 Taylor Road • P.O. Box 1600
Enfield, CT 06083–1600
orders and info: (800) 527–8339
fax: (203) 763–2466

LEGO Dacta is the educational branch of the LEGO company (which has its U.S. headquarters in Enfield, CT). Dacta sells the *LEGO Technic* product line—the geared and motorized version of the LEGO system.

Call Dacta and get their “Gear Up for Learning” catalog, which has many LEGO Technic kits. Recommended kits are the *1038 Technic Universal Buggy* (a specialized kit for building a small LEGO vehicle with a dual motor drive; about $60), the *1032 Technic II with Motorized Transmission* (a small general-purpose kit including one motor and one battery pack; about $76), and the *9605 Technic Resource Set* (a large general-purpose kit including two motors and two battery packs; about $200).

International Components Corporation

1803 Northwest Lincoln Way
Toledo, OR 97391–1014
orders and info: (800) 325–0101

The non-membership version of *Electronic Buyers' Club*, above. This catalog offers slightly higher prices and smaller selection than members have; still, the prices are extremely good.

Terms: No minimum order but $2 fee on orders less than $25; shipping charges are exact UPS rates.

Motorola Semiconductor Inc.

Motorola Literature Distribution • P.O. Box 20912
Phoenix, AZ 85036

Motorola has excellent technical documentation and support that is free upon request. They often give away free samples of chips as well.

If you are a student working from a university, the University Support office (602 952 3855) will handle customer support. You can mail or fax parts requests to the following address: Motorola Semiconductor Parts Sector; University Support 56–106; P.O. Box 52073; Phoenix, AZ 85072 (fax number: 602 952 3621). You must enclose a photocopy of your student ID card as proof of university status with the request.

For other tech support, call Motorola's "Customer Responsiveness Center" at (512) 891–2628 to find out the Motorola sales office nearest to your location. For technical documentation requests, call (800) 521–6274. Finally, you may reach a Motorola engineer familiar with the MC6811 series by calling the Advanced Applications Group at (512) 891–6517.

Motorola maintains as a free BBS that supplies Motorola and other public domain software for PCs and Macs. This service can be reached via modem at (512) 891–3733. In addition, Stanford University maintains an anonymous FTP server that has copies of much of the public domain 6811 software on the Motorola BBS. This server can be reached over Internet at `calvin.stanford.edu` (address 36.14.0.43).

R&D Electronics
5363 Broadway • Cleveland, OH 44127
orders only: (800) 642-1123
info: (216) 441-5577
fax: (216) 621-8628

A small but very useful surplus assortment, good service, very good prices. Lots of switches, motors, transformers, LCD displays, wire, good other random stuff.

Terms: $10 minimum order; MC/VISA/Discover accepted; shipping charges are UPS rates.

F References

This appendix provides a brief list of other publications that may be of interest to the Mini Board user.

Flynn, Anita and Jones, Joe. *Mobile Robots: from Inspiration to Implementation.* ISBN 1-56881-011-3, published by AK Peters, Ltd., Wellesley, MA, 1993. A complete guide to building a mobile robot from scratch, including an overview of various sensor technology, electronic and digital hardware, and high-level control strategies.

Martin, Fred. *The 6.270 Robot Builder's Guide.* Course notes to 1991 M.I.T. LEGO Robot Design Competition. To obtain a copy, send a check for $15 made out to "MIT Epistemology and Learning" at 20 Ames Street Room E15-309, Cambridge, MA 02139.

McComb, Gordon. *The Robot Builder's Bonanza: 99 Inexpensive Robotics Projects.* Tab Books, Inc., Blue Ridge Summit, PA, 1987. Slightly outdated, but a good collection of interesting robotic projects.

Motorola. Literature reference MC68HC11E/D, "M68HC11 Reference Manual." This document is the "bible" of the 6811 and is a must-have for any serious 6811 programmer.

Motorola. Literature reference MC68HC11ERG/AD, "MC68HC811E2 Programming Reference Guide." A pocket-sized guide to the version of the 6811 used on the Mini Board, ownership of this handy reference is proof of being a true 6811 nerd.

G Other Resources

This appendix gives practical information about obtaining physical printed circuit boards, electronic parts, and software, an electronic discussion group for Mini Board users, and copyright information for use of the Mini Board technology.

G.1 Mini Board Kits

Two businesses supply Mini Board technology. Please keep in mind that these references are provided as a service to the reader; the author has no formal affiliation with these parties though he believes them to be reliable sources. It is advisable to inquire specifically about availability and expected delivery time when placing an order.

CW Technology

7328 Timbercreek Court • Reynoldsburg, OH 43068–1181

CW Technology is run by Wally Blackburn, and offers assembled Mini Board systems in quantities of five or more.

CW Technologies' WWW page is located at
`http://www.infinet.com/~cwtech/`.

Douglas Electronics

2777 Alvarado Street • San Leandro, California 94577
phone (510) 483–8770; fax (510) 483–6453; FirstClass BBS: (510) 483–6548

Douglas Electronics is an established printed circuit board (PCB) manufacturing company and creator of the *Professional Layout* Macintosh-based CAD tools for PCB design (which were used to develop the Mini Board).

Please see their Mini Board web page at
`http://www.douglas.com/hardware/pcbs/miniboard.html`.

G.2 Software

Freely available software can be obtained via anonymous FTP to the address 18.85.0.47 (`cherupakha.media.mit.edu`). If you have an electronic mail account, but you do not have FTP access, send a message containing the single word “help” to the address `ftpmail@decwrl.dec.com`. This machine is an automated mailer service that will mail you files retrieved from an FTP server that you specify.

If you do not have any electronic access, the author will provide you with a copy of the software. Please send a formatted, 3.5 inch diskette only along with a self-addressed stamped envelope to Fred Martin, MIT E15–320, Cambridge, MA 02139. Be sure to specify Macintosh or IBM version.

G.3 Mailing List

The Mini Board is often discussed on the Usenet newsgroup `comp.robotics.-misc`. In addition, Ken Hornstein maintains mailing list for especially for users of the Mini Board and other MIT robot controller boards.[3] This is the best way for a new user to be supported in getting up and running with a Mini Board.

To join the list, send an e-mail message with the line containing the following text in the *body* of the message (not the subject):

`subscribe robot-board` <*your e-mail address*>

to the address `majordomo@cmf.nrl.navy.mil`. You only need to add the e-mail address if your site tends to mangle outgoing return addresses, otherwise you can safely omit it. You will be automatically added to the list. To send a message to the people on the list, write e-mail to `robot-board@cmf.nrl.navy.mil`.

Please *do not* send administrative things to the main "robot-board" mailing list address. If you have trouble getting on the list, you can contact Ken directly at `kenh@cmf.nrl.navy.mil`.

G.4 The `icc11` C Cross-Compiler

A new 6811 C-language cross compiler has recently become available. Developed by ImageCraft (P.O. Box 64226, Sunnyvale, CA 94086–4226; phone: (408) 749–0702; e-mail: `imagecft@netcom.com`), the `icc11` compiler runs on MS-DOS, Linux, and OS/2 2.x systems. As of this writing, it is in general release version 2.0.

`icc11` is important because it was the first freely available high-quality 6811 C compiler. The current release version sells for US$50, but ImageCraft continues to distribute a prior version for free.

The compiler is available from the `cherupakha.media.mit.edu` FTP site, in directory `pub/incoming/icc11`. Also available is a version of the Mini Board libraries customized to work with `icc11`.

[3] This list was founded and maintained for several years by Greg Kulosa.

G.5 Copyright Information

The Mini Board technology, including the printed circuit board layout and supplied code libraries, is distributed under a free licensing policy. This agreement allows any party to use the Mini Board technology for any purpose without having to pay a licensing fee.

The technology is *not* public domain. The Massachusetts Institute of Technology reserves the copyright to the artwork and code. Any commercial use of the technology must include a reproduction of the copyright notice on the board itself, and must acknowledge the institutional source (MIT) and author (Fred Martin) of the technology in an appropriate fashion in any accompanying product documentation.

Given this policy, the author will distribute the printed circuit board artwork in electronic form to parties (like the above two small businesses) who wish to fabricate their own circuit boards. Note that this is only economical in units of one hundred or more; in smaller quantities it probably makes more sense to contact Douglas Electronics or Progressive Solutions.

The PC board artwork files are available on the `cherupakha.media.mit.edu` FTP site, in the `pub/project/miniboard/layout` directory.