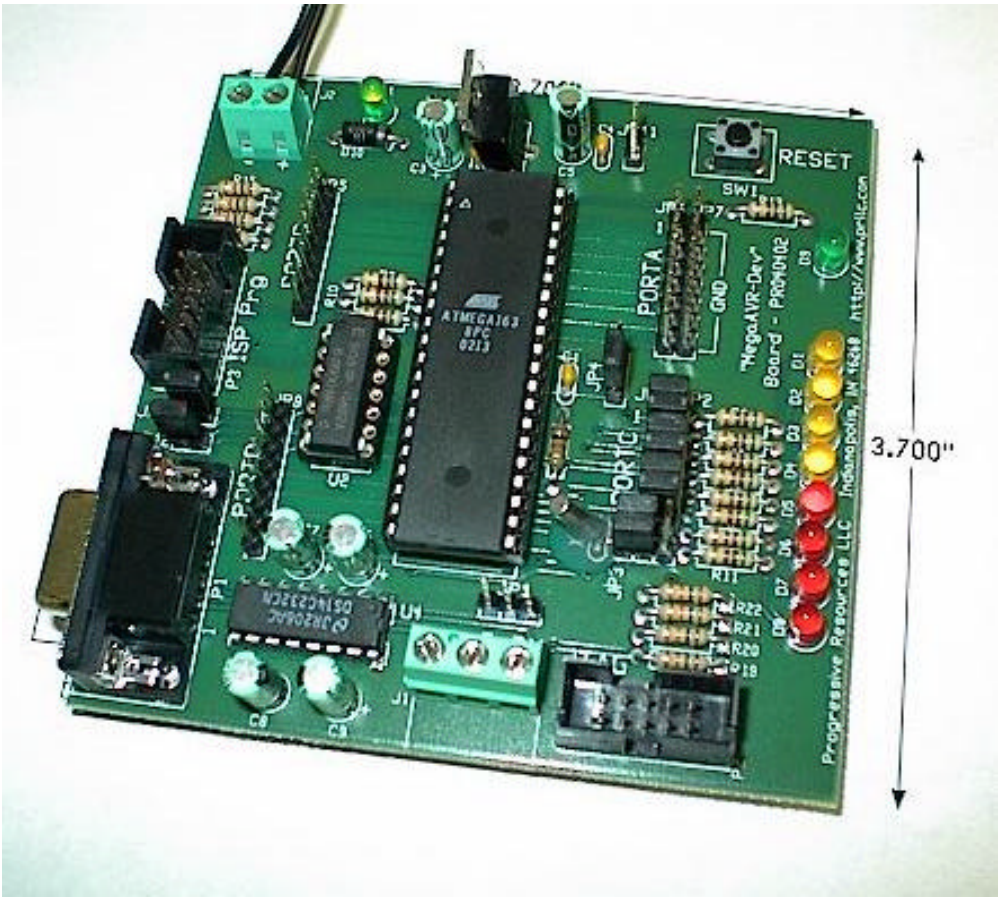
*"MegaAVR-DEV*elopment Board"

Progressive Resources LLC
4105 Vincennes Road
Indianapolis, IN 46268
(317) 471-1577
(317) 471-1580 FAX
http://www.prllc.com



GENERAL

The ***MegaAVR-Dev***elopment board is designed for prototyping and laboratory use. The board supports the AT90S8535, Mega16, Mega163, Mega32, or the Mega323 in a 40-pin DIP package.

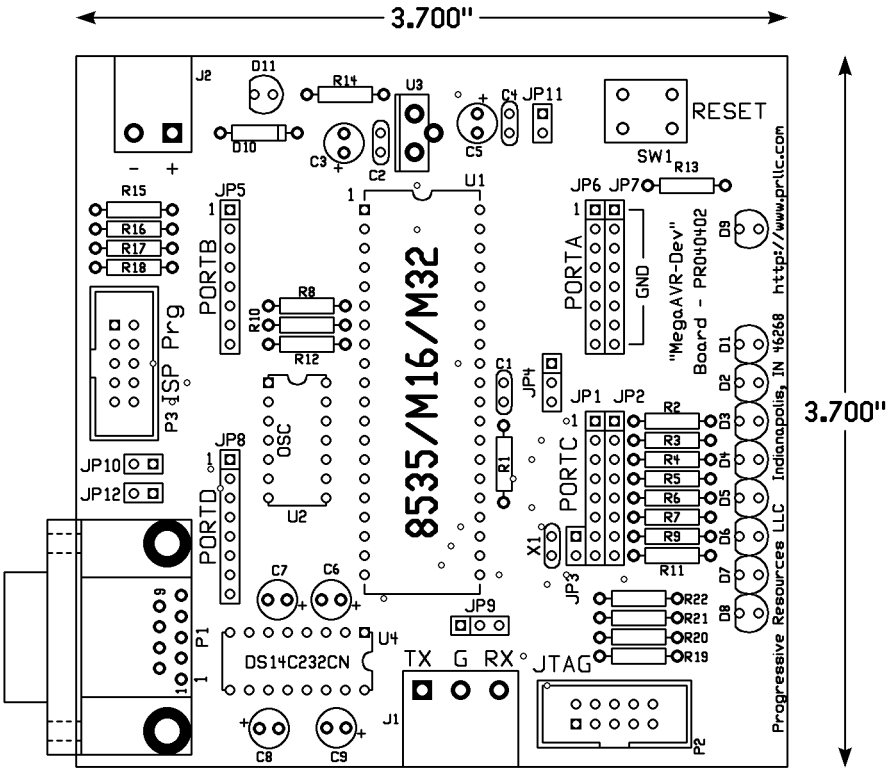
New Feature: **Mega 16/32 AVR® Boot Loader**

The ***MegaAVR-Dev***elopment board now comes programmed with the Mega 16/32 AVR® Boot Loader. The AVR Boot Loader (AVRBL) is a bootstrap loader that, once programmed into the AVR boot-block memory area, allows reprogramming of AVR microcontrollers without need for a chip programmer. The AVRBL makes use of the self-programming features of the AVR microcontrollers to allow in-circuit reprogramming.

Once the AVRBL is programmed into the microcontroller, it remains resident until the chip is erased. Application programs require only a very minimum software interface to use the AVRBL. The PC application to interface to the boot loader and more documentation about interfacing to the boot loader are available free at http://www.prllc.com. (Some notes are provided on the following pages.)

Other features of the development board include:

- RS232 through a 9-Pin D-Shell as well as screw terminals and a jumper header.
- Up to 32K of In-System Programmable FLASH memory with up to 1K of EEPROM and up to 1K of Internal RAM (depending on processor selection).
- Eight, 10 bit, Analog Inputs, using either internal or user supplied reference.
- 9 I/O controlled LEDs, 8 of which are jumper selectable.
- 32KHz “watch” crystal for on-board Real-Time operations.
- A universal clock socket allows for “canned oscillators”, as well as a variety of crystals, ceramic resonators, and passive terminations.
- 0.1” centered headers provide for simple connection to the processor special function pins and I/O.
- 10-pin, polarized, ISP and JTAG connections are provided for in-system programming and debugging. MegaAVRs can also be programmed through RS232 using an appropriate boot loader application.
- On-board regulation allows for power inputs from 8-38VDC with an LED power indicator.
- Termination is provided for 5VDC output at 250ma



SPECIFICATIONS

- Voltage range 8 V to 38 VDC
- Power consumption 250 mW (nominal)
- Dimensions: W 3.7 inches x H 3.7 inches
- Mounting Rubber feet, 4 places
- Weight ~3 OZ
- Operating temperature 0 deg. C to +60 deg. C
- Storage temperature 0 deg. C to +85 deg. C
- Humidity 0% to 95% at +50 deg. C (non-condensing)

APPLICATION NOTES:

Power

J2 (screw terminal connector) is the power input point. Acceptable voltages are 8 – 38 VDC (J1-1 is +, J1-2 is ground).

JP11 is an output of the regulated 5 VDC that may be used to power other devices. Note, however, that the LM7805 does not have a heat sink and so the actual available power output is somewhat limited, depending on the input voltage and power being consumed. Check the LM7805 regulator specification for details.

Serial Connection

P1 is a standard DB-9 connector usually used to connect to a PC. The TX signal is on P1-2 and the RX signal on P1-3. These are RS-232 level signals.

J1 and JP9 also provide connections for the RS-232 level serial signals. On each connector, pin 1 is the TX signal, pin 3 is the Rx signal and pin 2 is ground.

JP10 and JP12 are jumpers, which connect the processor serial signals (RXD and TXD) to/from the RS-232 driver chip. These jumpers must be in place for the RS-232 serial connections to work. Removing the jumpers allows use of the Port D, bits 0 and 1, for TTL-level I/O.

SPI Connection

Use of the SPI bus is possible via the programming connector, P3 or by making connections to the SPI bus signals on Port B.

Parallel Ports

Parallel ports A, B, C, and D are connected to JP6, JP5, JP1, and JP8, respectively (and labeled clearly on the board). Bits are connected sequentially with bit 0 on pin 1, bit 1 on pin2, etc. These are normal TTL-level signals with or without pull-ups depending on the port initialization set up in the software.

Port A (JP6) has a parallel row of ground pins next to it (JP7) so that enabling the built-in pull-up resistors and then using two-pin jumpers to ground any pins that need a logic 0 applied for input purposes can effect simple input signals. This also provides a convenient ground reference when measuring analog voltages with the internal A/D converter (ADC).

Port C (JP1) has a parallel row of pins (JP2), each of which is connected to an LED through a 510-ohm series resistor to +5 VDC. Jumping any of the pins of JP2 to the corresponding pin of JP1 allows the use of the on-board LED's as an output. Because the LED's are connected to +5 VDC and the port is sinking the LED current, the LED will be on for any pin that outputs logic 0.

A clock crystal (32.768 KHz) is connected to JP3. JP3 is located adjacent to JP1 (Port C) to provide an easy connection of the clock crystal to Port C bits 6 and 7 for use as a real-time clock.

System Clock

As supplied, the system clock is 6 MHz. U2 contains the crystal and caps necessary for the oscillator. Replacing U2 with a TTL, crystal, ceramic resonator, or RC oscillator, or a different integrated oscillator unit allows changing the system clock if necessary.

Analog-to-Digital Converter Connections

JP4 provides the connections to control the V_{ref} for the ADC. A new V_{ref} may be connected to JP4, pin 2, if desired. Or the internal connections to V_{ref} may be controlled by software.

Mega16/32 AVR ® Boot Loader

There are two methods of running the boot loader, upon reset and by a direct call from the application code. The boot loader runs on UART0 at 19200 baud, XON, XOFF handshaking, 8 data bits and 1 stop bit.

The boot loader code is executed upon reset. If the boot loader does not receive instructions to load a new file with 5 seconds, then it jumps to the application code.

To call the boot loader from your application code consider using a condition and jump instruction.
For example:

```
if (UDR0 == '$')        // did I receive a $ on the USART?
{                       // if true, then jump to the AVRBL
        #asm
        JMP   0x1C00 ; for Mega32, use JMP 0x3C00
        #endasm
}
```

This example code checks to see if the character received on USART is a '$'. If so, then the JMP instruction is executed, exiting the main application and entering the boot loader directly. If the boot loader does not receive any instructions within a selected amount of time it will jump to 0x0000, effectively resetting the processor and starting the main application code. Otherwise, if the boot loader does receive instructions, the opportunity is then provided to download new firmware. The PC application allows the user to establish what characters are used by the application to start the boot loader.

After the AVBRL is started (via a reset, a power-up, or a jump from the main application), the following protocol must be observed. The PC application provided handles the protocol for you with its default settings or you can create a custom application. In either case the protocol is:

1. Upon power-up, reset, or as a result of a jump from the main application, the AVRBL will send a '?' at your selected baud rate. This is the character used to tell the external program downloading software, or host, to send the file.

2. The host is then required to send the three-character entry sequence. This is used to prevent an inadvertent attempt of reprogramming from taking place. If the AVRBL does not receive these characters within the timeout period, the AVRBL will test to see if there is code located in the main application area of flash. If there is, the AVRBL will jump to it, otherwise, execution will stay within the AVRBL indefinitely, waiting for the entry sequence.

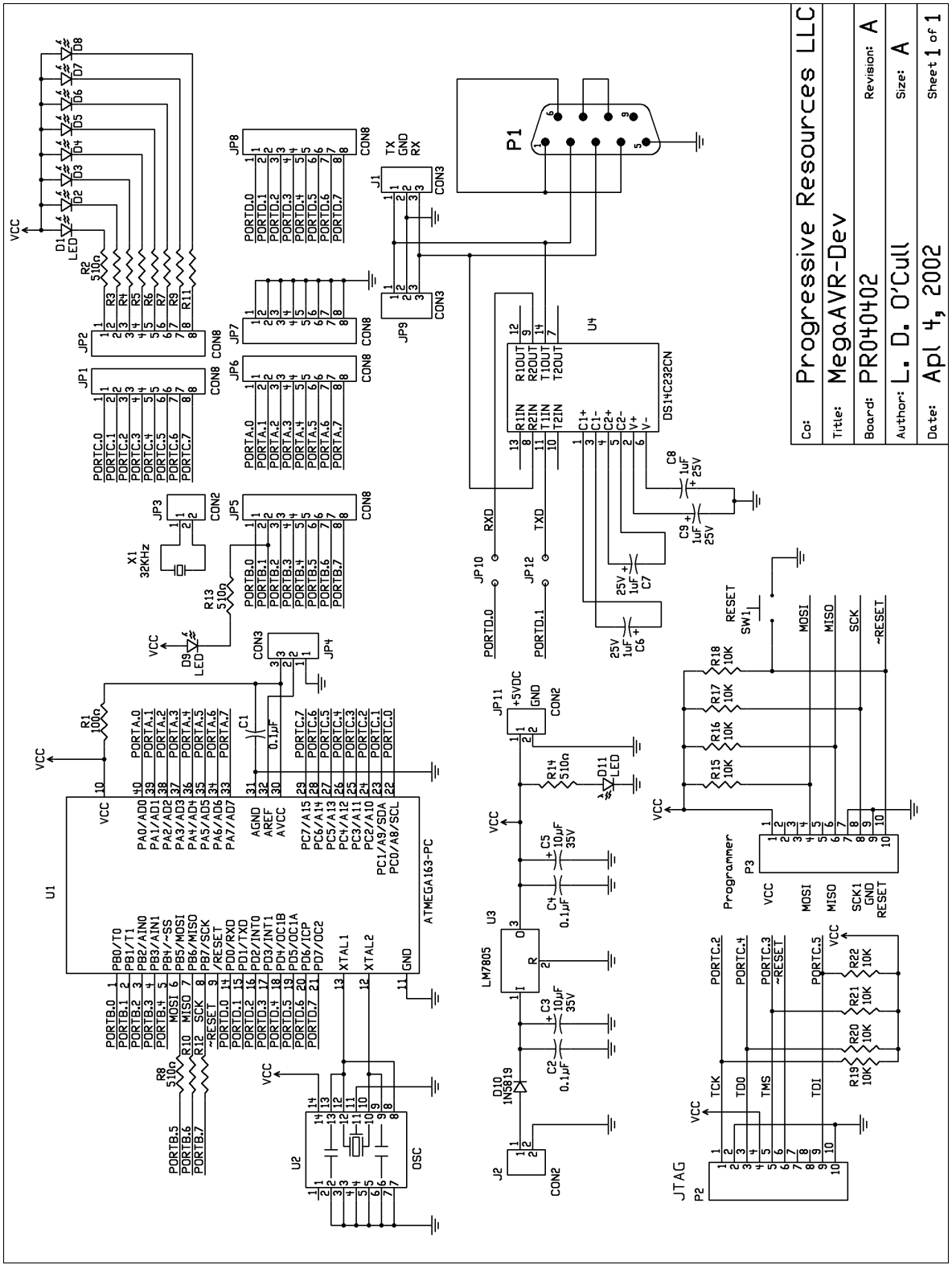
3. Once the three-character entry sequence has been sent, the programming software should now send the hex file for the new/updated application program observing an X-ON / X_OFF handshaking protocol to control data flow. The handshaking is very important as the flash memory area writes much more slowly than the serial port can send data. The programming software continues sending the hex file until it is all sent. After each line of ".hex" file is sent, a '^' is returned if an error was detected, otherwise a '%' is returned indicating acceptance of the line.

4. After the programming is complete, the AVRBL will send either a '#', meaning the programming is all right, or an '@' indicating that an error has occurred and the program did not load successfully. In most cases an error during programming means that the main application program is corrupted and will need to be resent.

5. The AVRBL will then start the newly programmed application software. As stated in step 2, the AVRBL will test to see if there is code located in the main application area of flash. If there is, the AVRBL will jump to it, otherwise, execution will stay within the AVRBL indefinitely, waiting for the entry sequence.

One final note about the **Mega16/32 AVR ® Boot Loader**, the boot loader source code is available for purchase and as such provides the opportunity for customized boot loader solutions. The AVRBL-16/32 boot loader (as well as others) is available at http://www.prllc.com.

SCHEMATIC


Co: Progressive Resources LLC
Title: MegaAVR-Dev
Board: PR040402
Revision: A
Author: L. D. O'Cull
Size: A
Date: Apl 4, 2002
Sheet 1 of 1
ATMEGA163-PC
DS14C232CN
LM7805
Programmer
JTAG
OSC
RESET
MOSI
MISO
SCK
~RESET
RXD
TXD
+5VDC