



JED MICROPROCESSORS PTY LTD

173 Boronia Rd, Boronia, (PO Box 30), Victoria 3155 Australia

Phone: +61 3 9762 3588,

<http://www.jedmicro.com.au>

Fax: +61 3 9762 5499.

email: jed@jedmicro.com.au

JED AVR200 Single Board Computer

(Ed Schoell 29th Nov 2004)

Background

JED has designed industrial boards on a number of bus (and non-bus) architectures over the company's 25 year history.

This board is designed to provide users with a low cost board based on the Atmel ATmega32, a CPU with an 8-bit RISC architecture designed for very efficient programming in high-level languages such as C and BASIC, (but it is no slouch, and is well supported for assembly code as well.)

What's on the AVR200 board

The AVR200 board has as its heart the ATmega32 CPU in a 40-pin DIP package.

The board has a simple architecture, and uses most of the CPU ports (4 by 8 bits) as user interfaces, with some more complex functions such as serial I/O and I²C supported by dedicated hardware.

At one end is a power connector and voltage regulator.

At the other end are serial communications connectors (RS232 on a D9, RS485 on screw terminals and I²C connection).

Along the top and bottom edges are screw terminals for eight analog inputs and 20 digital I/O lines. These are driven or are inputs to CPU port pins. User I/O port pins used as inputs are provided with protection and pins used as outputs are provided with power FETs as output drivers to the off-board world. In most cases the ports can be loaded with either input or output components, making the board an "input rich" or an "output rich" configuration.

CPU summary

The ATmega32 CPU is a mid-level CPU in the AVR range, and has 131 instructions, most executing in just one clock cycle. This board is normally supplied with a 3.6864 MHz crystal, but can operate up to 16 MHz. Because of this, and the filtering components fitted to the board, radiated RF is at a very low level, especially as it is used here in internal memory mode, and no high-speed transactions use the port pins at high speed. There are 32 user registers (as well as 2 Kbytes of user RAM). Six registers of these registers can be combined to form three 16-bit pointer registers with auto-increment/decrement and indirect and indexed addressing modes. (The advantages of the AVR over architectures like the 8051 and PIC are immediately apparent when looking at this internal structure.)

There are 32 Kbytes of FLASH program memory and 1 Kbytes of EEPROM on chip. The CPU FLASH can be loaded from a programming adapter (such as the AVRISP) or via a serial port from a PC, using a boot loader.

In the peripheral area of the CPU, there are:

- A hardware UART for serial communications (RS232, RS485 or TTL level);
- A hardware "Two Wire Interface", identical to the Philips I²C system;
- A hardware "Serial Peripheral Interface" or "SPI" which allows two-way clocked data transfer;
- Two eight-bit, and one sixteen-bit counter with prescalers, and compare modes with PWM outputs;
- Hardware watchdog timer.

Port A analog input (or digital I/O)

The ATmega32 has a powerful analog input system which provides an eight-channel, 10-bit resolution, Analog to Digital Converter (ADC) sub-system. The AVR200 board adds a seven-way socket strip (for each channel), into which users can install a range of components to customise individual channels of the ADC to match desired input voltage ranges or specific transducer characteristics. The channels are connected to two screw terminal connectors, J2 and J3 where four input channels and a ground are provided on each. (J3 also has a +5 volt output to power off-board electronics.)

The full-scale voltage range of the ADC depends on the reference voltage selected. This can be 5V Vcc, an internal 2.56V reference, and an external 4.096 Volt adjustable reference, a MAX874CPA device (optionally) loaded at location U4. Trim pot VR1 allows this to be set accurately using a Digital Voltmeter connected from the output of U4 (Pin 6) to ground on J3. Typical temperature coefficient is 20 ppm/deg. C, which is less than 1 LSB of conversion error over a 40 deg. C. range. Note, whereas the external reference voltage can be set accurately, and has a good temperature coefficient, the internal reference can vary from 2.3V to 2.7V, cannot be trimmed, and has no quoted temperature coefficient. When 5V Vcc is used as a reference, the absolute voltage of the voltage regulator can be anywhere from 4.75V to 5.25V.

(It actually works quite well to use 5V Vcc as a reference if one is measuring positions of potentiometers, because the voltage range from a pot across J3 pins 1 ...6 will be ratiometric with the Vcc used as a reference.)

The possibilities of component placements in the seven-long socket strip are shown in diagrammatic form in the Port A schematics. They allow the following:

- A series resistor (Rs) for protection of port input (usually 10K);
- A termination resistor to ground (R-pd) for a current loop input (e.g. a 200 Ohm resistor converts a 4-20 mA current loop to a 4.00 Volt signal. (Rs is still used);
- A pullup resistor (R-pu) to 5.0V Vcc, used with resistive transducers to ground. (Rs is still used);
- A voltage-divider configuration with Rs, and Rd, a divider resistor to ground after Rs; or
- A series resistor (Rs) and a filter capacitor to ground for signal smoothing.

All input channels of Port A have Texas Instruments TL7726 hex-clamp protection devices to limit the input voltage range on pins to close to the ground to Vcc range. (Within 200mV sinking 25 mA).

Digital I/O on Port A

In parallel with the analog inputs of the CPU are a normal eight-bit port, Port A, an I/O structure which can be used for digital I/O on the same connectors. To use a Port A channel in this way, the resistors in the socket strips need to be installed appropriately.

If the digital port is used as **inputs**, resistors similar to the analog modes above can be installed:

- A series resistor (Rs) for protection of port input (usually 10K);
- A pull-down resistor to ground (R-pd). This allows voltage inputs of 5 to 12 volts to be sensed, with over-voltage protection limited by R-pd dissipation;
- A voltage-divider configuration with Rs, and Rd, a divider resistor to ground after Rs. Peak over-voltage protection of hundreds of volts are provided in this way (dissipation-limited by Rs);
- A switch or (external) opto-isolator input mode, with a pullup resistor (R-pu) to 5.0V Vcc.

If Port A is to be used as **outputs**, a direct CPU port output can be done simply by using a low (100 Ohm or lower) Rs series resistors, and setting Port A direction control register bits to "Output". Voltage output swing of 0 to 5V is available, with several mA of drive. (LEDs can easily be driven with Rs as a current limiting resistor)

If more current is needed from Port A, a different build replaces the socket strips with 4K7 pull-down resistors (in the Rd positions above) and then installs eight MDT3055 N-channel power FETs in the centre three holes of the socket-strip positions. This grounds the FET sources, connects the FET drains to screw terminals and drives the gates from Port A pins. (The resistors Rd ensures the FETs are turned OFF before the CPU starts to drive Port A pins as outputs.)

Port B SPI interface (and digital I/O)

The Serial Peripheral Interface is a three-wire interface (plus chip enables as required) which synchronously transfers data between the ATmega32 (in Master mode) and any number of peripheral chips. All four standard SPI modes are supported. The SPI interface can also be used between CPUs, where one operates in Master mode, and one in Slave mode. The SPI interface facilities include the ability of using variable bit rates, MSB or LSB first transfers, and interrupt generation on transfer complete.

The SPI interface uses three pins in port B: B5 is MOSI (Master Out Slave In), B6 is MISO (Master In Slave Out) and B7 is SCK (Serial Clock).

No peripheral chips on the AVR200 base board use the SPI interface system, but it is available on special connector J13 to provide two functions:

- A 6-pin Atmel STK500 or AVRISP 6-way programming cable interface. This allows direct programming of the CPU FLASH from the serial port of a PC running AVR Studio, CodeVision C or BASCOM. (A gap of four missing J13 pins allows for the 6-pin AVR-ISP cable header to fit onto the first six pins of J13); and
- A further four pins from CPU Ports B1, B2, B3 and B4 are on the other end of the connector: J13 pins 11,12,13, and 14, can be used as “Chip Selects” for peripherals.

Thus, J13 can be used in two ways:

- It can be used to go “upstairs” to a prototype or project board with SPI-interfaced peripherals or direct port-driven I/O. (Pin 2 is Vcc and Pin 5 is RESET and Pin 6 is Ground);
- It can be used with a 14-pin ribbon cable to go to an external board with either SPI or direct port-driven I/O. (Vcc, ground and reset support an off-board system via such a cable.)

Digital I/O on Port B

As well as the SPI functions above, all bits of Port B are available for general purpose I/O.

The standard build is to configure all eight bits as digital inputs, but bits 0 ... 3 can be loaded with alternate components and become power FET outputs (using MTD3055s).

Bits 0 ... 3

If used as **inputs**, socket strip RP5 is a set of sockets into which users can plug a 4-resistor, 5-pin, SIP resistor pack, with four 4K7 resistors. The end pins of the socket are Ground and +5V Vcc, so depending on orientation of the pack when plugged in, the four input bits 0 ... 3 are pulled down to ground, or are pulled up to 5 V Vcc.

Then, a 4-resistor, 4K7, 8-pin resistor pack is loaded into location RP15 as protection resistors in series with the inputs to the CPU Port B. (These CPU pins are protected by a TL7726 clamping device).

Note: Configuring Bits 0 ... 3 as inputs allows dedicated special functions connected to these port pins to be used:

- Bit 0 is Timer 0 input (T0);
- Bit 1 is Timer 1 input (T1);
- Bit 2 is INT2, a user interrupt. This can be linked via jumper L12 to the programmable square-wave output from the RTC;

If used as **outputs**, FETs 1 ... 4 are installed, and four gate pull-down resistors are installed as a 4-resistor, 5-pin, 4K7 SIP resistor pack into location RP1.

Bits 4 ... 7

These Port B bits can be used as digital inputs as an alternative to SPI. Socket strip RP6 has a 4-resistor, 5-pin SIP pull-up/pulldown array, and RP7 is the protection 4-resistor pack. Four Zener diode clamps protect the CPU inputs.

Port C Two-wire Serial Interface TWI (I²C) (and digital I/O)

This interface is used for expansion of devices accessible to the programmer, both on the AVR200 board (e.g. the real time clock (RTC), upstairs to the non-volatile memory board, and user I/O expansion (via connector J14), and off-board) via the buffered expansion port to ribbon cable connector J6.)

J14 can be used to go “upstairs” to a prototype or project board with TWI-interfaced peripherals, for example multi-channel Digital to Analog Converters (DACs) or higher precision, multi-channel A to D converters.

The hardware for this interface is one of the CPU on-chip peripherals, and is fully programmable regarding transmission rate, etc via five registers in the interface. The system used CPU Port C0 for line SCL, the TWI system clock (sent out from a master), and CPU Port C1 for SDA, the bi-directional data line.

The TWI interface hardware can generate service request interrupts to the CPU under a number of conditions.

Both the CodeVision C, and the BASCOM BASIC compilers for the AVR support this interface.

When used on the main or upstairs board, the TWI interface is used un-buffered, and chips (e.g. RTC and memory) drive this un-buffered bus directly. 1K5 pullup resistors and Zener diodes protect the internal bus and the CPU from ESD hits on the lines.

Normally, off-board TWI communications are buffered with U11, “times-10” current multiplier, which can extend the TWI bus off-board to many metres via a 10-way ribbon cable, using low-cost IDC connectors. The buffer, a PCF82B715, drives a 10-times lower-impedance bus with 330 ohm pullups and 10 ohm series resistors. EMC filters with ESD Transorbs protect these off-board connections. The 10-way bus also supports two 5-volt power lines, four ground lines and two interrupt lines. The cable pinout standard is shown in the following table. Note how ground or supply lines are alternated with signal lines:

| Pin No | Function | Pin No | Function |
|--------|---------------|--------|----------|
| Pin 1 | Vcc 5V | Pin 2 | Vcc 5V |
| Pin 3 | Buffered. SCL | Pin 4 | Ground |
| Pin 5 | Buffered. SDA | Pin 6 | Ground |
| Pin 7 | Int 0* | Pin 8 | Ground |
| Pin 9 | Int 1 * | Pin 10 | Ground |

(The same pinout is used for the upstairs connector, J14.)

If, (for example to communicate with unbuffered external interfaces) it is desired to operate the external interface in unbuffered mode, the buffer chip U11 is not loaded, and links L24 and L25 connect the unbuffered signals externally. Also, R9 and R10 are omitted, and R11 and R12 loaded with 100R resistors. (This might be done to interface to other supplier’s products which are usually unbuffered.)

TWI interface to RTC

The Real Time Clock on this board is a Dallas DS1307, an 8-pin device which sits on the board beneath the CPU. It uses a 32KHz crystal, and is powered in standby mode by a small Lithium battery. It provides time and date to users via the TWI bus, and also can provide a square-wave via link L12 to INT2, on Port B2. The frequency of this square wave can be set to 1 Hz, 4 KHz, 8 KHz or 32KHz. Fifty-six bytes of non-volatile RAM are also available in the RTC.

Digital I/O for bits C2 to C7

Normal board load for C2 to C7 is as **outputs**. Six MTD3055 power FETs drive screw terminals J4 and J5 (3 each, plus ground). RP8 and RP9 (4K7 SIPs) pull the gates LOW.

An alternate load is as **inputs**. In this mode, the RP8 and RP9 resistor pack positions are replaced with six Zener protection diodes. 4K7, 3 resistor, six pin SIPs are installed in RP13 and RP14 positions, and RP2 and RP3 are sockets for two 4-pin, 3-resistor 4K7 SIPs with a Ground and 5V Vcc on either end, as pull-up/pull-down options.

Port D UART driven serial ports (and digital I/O)

The ATmega32 hardware UART is a powerful device with receive buffers, its own baud rate generator, digital filters for noise reduction, multiple modes (including addressable 9-bit multidrop), and three independent interrupts (including an end-of-byte interrupt ideal for RS485 TX control).

The comparison with simpler CPUs which often don't even have UARTs, or tie up valuable CPU counters as baud rate generators, or have a combined RX and TX interrupt, is very much in the ATmega32's favour.

Simple RS232-only standard build

The standard build of the AVR200 board is to load just the components to support RS232 communications. This uses the minimum number of CPU pins for communications support: just Port D0 for RXD, and Port D1 for TXD.

In this simple, no-handshake, mode, links L20 and L21 are connected, connecting PD0 and PD1 to the RX and TX of the RS232 transceiver U8. The line side of the transceiver connects via link L13 to the D9, RS232 connector J8.

L13 allows changeover of TX and RX pins on the D9 connector to change the mode of the RS232 interface from DTE (Data Terminal Equipment) to DCE (Data Communications Equipment). If talking to a Modem, the AVR200 needs to look like a PC, i.e. a DTE; if talking to a PC, set the AVR200 up as a DCE.

If hardware handshake is needed (using the RTS/CTS pair) this can be incorporated in the design by linking L22 to drive the RTS transmitter in U8 from CPU Port D7. Then the CTS receiver can be linked into CPU Port D6 by linking L17 positions 2 ... 3.

L14 allows changeover of RTS/CTS pins on the D9 connector to swap between DTE and DCE modes. Linking L14 pins 5 ... 6 simply loops back user RTS to user CTS.

| Function | DTE setup | DCE setup |
|------------------|--------------|--------------|
| RX/TX | L13, 1 ... 2 | L13, 3 ... 4 |
| RTS/CTS loopback | L14, 5 ... 6 | |
| RTS/CTS active | L14, 1 ... 2 | L14, 3 ... 4 |

RS232/RS485 switchable (optional) build

In this build, links L20, L21 and L22 are not connected, and instead multiplexer U9 is installed, along with RS485 transceiver U10, LC2 and LC5 filters and screw terminal RS485 connector J7.

The multiplexer switches the serial interfaces from RS232 to RS485 mode, and this is controlled by L19, and optionally, CPU Port D4. Options are:

| Serial Mode | Links / drive |
|----------------------|-------------------|
| RS232 fixed | No L19 links |
| RS485 fixed | Link L19, 2 ... 3 |
| RS232/485 switchable | Link L19, 1 ... 2 |
| RS232 selected | Set PD4 HIGH |
| RS485 selected | Set PD4 LOW |

In RS232 mode, operation, including operation of RTS/CTS via CPU Port D7 and Port D6 respectively, is the same as RS232 above. In this mode, the multiplexers connect the RS232 receiver and transmitter to the CPU, disconnect the RS485 receiver, and force the RS485 transmitter to OFF (Tristated).

In RS485 mode, the multiplexers connect the RS485 receiver and transmitter to the CPU, and the control line from Port D7 (RTS in RS232 mode) is now used for TX drive enable (HIGH is TX enable, and LOW is TX disable.) RS232 TX and the RS232 RTS lines are forced LOW (-9) at RS232 levels.

Port D4 controls the switching of the mode if CPU control is desired: (HIGH is RS232 mode, LOW is RS485 mode.)

TTL/CMOS level serial build

By omitting the RS485 transceiver (but including the multiplexer U9), and installing J7B, a 4-pin strip connector, it is possible to connect to devices such as TTL/CMOS level radios or GPS receivers to the AVR200 board. (These signals are the reverse polarity to RS232 signals, i.e. the signal rests at +5 volts.)

Connections are as follows:

| Connections | Signals |
|-------------|---------------------------------|
| Pin 1 | Ground |
| Pin 2 | TTL TX out |
| Pin 3 | Port D7, a radio control signal |
| Pin 4 | TTL RX in |

Digital inputs for Port D Bits 2, 3 and 6

These bits are always inputs, usually via screw terminal J9, but each can be re-assigned by links to other functions:

- Bit 2, via L15, can be an external input, or can be assigned to be the INT0 interrupt line from the TWI interface;
- Bit 3, via L16, can be an external input, or can be assigned to be the INT1 interrupt line from the TWI interface;
- Bit 6, via L17, can be an external input, or can be assigned to the CTS input from the RS232 port. L18 on this interface bit allows this input to be pulled up (to 5V Vcc) or down to Ground.

Digital outputs (or inputs) for Port D Bits 4, 5 and 7

These bits are usually loaded as outputs. J10 is the connector, and FETs F11, F12 and F13 are MTD3055s. RP10 holds the three gate pull-down resistors in a 3-resistor 6-pin SIP.

As inputs, RP4 holds the 4K7 pull-up/pull-down 3 resistor, 4 pin SIP. RP16 holds the three 4K7 series protection resistors, and RP10 becomes three Zener clamp diodes.

Note: Bit 7 is used as the RTS source if the RS232 needs one, and as the TX-ON source in RS485 mode. Bit 4 is used as the multiplexer control bit if the switching between RS232 and RS485 has to be CPU controlled.

Note: Bits 4, 5 and 7 are all internally drive-able as PWM outputs from “Output compare” logic in the CPU.

Power supply

A simple, three-terminal voltage regulator is provided on the board; a low-drop-out LM2940T-5, 5-volt output device rated for use in automotive environments. It can operate with DC inputs as low as 5.5 volts and still produce a regulated 5 volts output, but can run up to 18 volts continuous DC volts in. (It has built-in reverse protection of -50 volts, and positive transient protection to +60 volts. This is to protect against “load dump” spikes from car battery systems and reverse connection of plug pack power supplies.)

The input power is filtered against EMC in or out with a ferrite/capacitor filter. Inductors L1 and L2 also prevent radiation of CPU noise back to the power system.