Using the Stamp Stretcher 1B

The Stamp Stretcher 1B is an input/output (I/O) extender for the BASIC Stamp single-board computer. The Stamp reads and writes the Stretcher's 16 I/O pins using the Serin and Serout commands. The Stamp can also read the Stretcher's onboard eight-bit analog-to-digital converter (ADC). The ADC is capable of measuring voltages between 0 and 4.7 Vdc and returning a proportional value from 0 to 255. The Stretcher 1B operates at 2400 or 9600 baud.

Configuring and connecting the Stretcher 1B to the BASIC Stamp

The Stretcher communicates with the BASIC Stamp through one Stamp I/O pin. All of the examples here use pin 0, but you may use any available Stamp pin, 0 through 7. You can power the Stretcher from the Stamp's voltage regulator, since the Stretcher draws less than 5 mA. However, if you are driving loads with the Stretcher and/or the Stamp, make sure that their total current draw doesn't exceed the regulator's 50-mA capacity. If it does, you will need an external voltage regulator. Figure 1 shows a typical Stamp-to-Stretcher connection, using power supplied by the Stamp's voltage regulator.

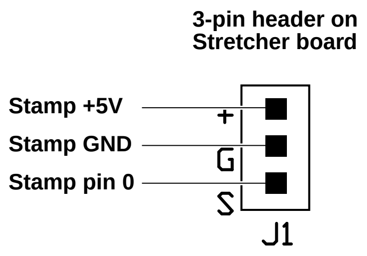


Figure 1. Power, serial hookup

The Stretcher normally operates at 2400 baud—the maximum speed of the original Stamp and BS1-IC. If you install a shorting jumper on the posts marked BPS, the Stretcher 1B will operate at 9600 baud. The Stretcher only checks BPS at startup; if you install or remove a jumper while the Stretcher is powered up, the change won't take effect until it is turned off and back on.

Using the digital I/O pins

The Stretcher's 16 I/O pins, numbered 0 through 15, are connected to header stakes at J3 and J6 on the circuit board as shown in figure 2.

The Stretcher's I/O lines are electrically identical to those of the Stamp. When set to output, a pin can source 20 mA and sink 25 mA, for a total of up to 40 mA (source) or 50 mA (sink) per port. Stretcher pins 0 through 7 comprise the "lower" port; pins 8 through 15 are the "higher" port.

When set to input, the Stretcher's pins have very high impedance, leaking no more than 1 µA of current in or out, provided input voltages aren't more negative than ground, or more positive than the + power supply.

Next to the I/O-pin headers are 10-conductor inline sockets J4 and J5. These sockets are also connected to the I/O pins, but have an additional connection at each end for +5 Vdc (top) and ground (bottom). This lets you conveniently install resistors between the I/O pins and +5 Vdc or ground

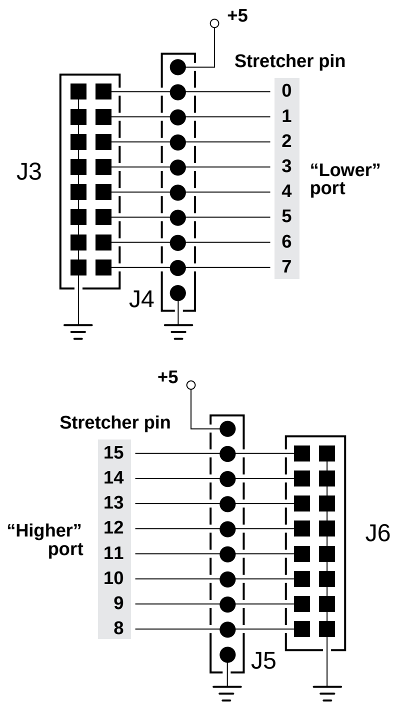


Figure 2. Stretcher I/O ports.

using SIP resistor packs. If you want to exclude one of the I/O pins from the pullup or pulldown connection, you can simply cut the corresponding pin off the resistor pack

Using and reconfiguring the ADC

The voltage to be measured with the ADC must be positive and referenced to the Stretcher's ground. Do not connect negative voltages, or positive voltages exceeding the ADC voltage reference voltage by more than 0.5 volts, to the ADC. The input to the ADC is J2. The input is marked +; the other pin is ground.

The ADC's voltage reference is normally wired to its positive power-supply connection, giving it a range of 0 to 4.7 Vdc. (The range is not 0 to 5 Vdc as you might expect, because the ADC gets its power from a Stretcher I/O pin, which has a small internal voltage drop.) If you want to reduce the upper limit of the ADC's range, you may install a zener diode at D1 and a 1k resistor at R1 (the orange jumper wire above J1 on assembled boards). A 3.3-volt zener at D1 will give the ADC a range of 0 to 3.3 Vdc. When you install the zener, make sure that its striped end lines up with the stripe on the printed symbol on the board.

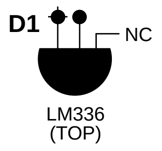


Fig. 3

The LM336 2.5-volt zener comes in a three-wire, TO-92 package. You can install it as shown in figure 3. The unused terminal is meant to allow fine adjustment of the zener voltage or connection of temperature-compensation circuitry. For most applications, it's not needed, since temperature drift over a range from subfreezing to boiling will cause about a 1-bit change in the ADC output.

Caution: Do not connect the ADC's reference pin to a voltage higher than its supply voltage (at pin 8). Doing so will damage the ADC.

Programming the Stretcher I

Your Stamp programs control the Stretcher by issuing commands to it using Serout (2400 baud, inverted). The commands are abbreviations of the corresponding Stamp commands. For example, to change Stretcher pin 13 to output/low, your program would say:

```
Serout 0,N2400,("L",13)
```

The commands must be in uppercase. Numbers sent to the Stretcher may be either constants (as shown), or variables. Do not precede them with the # character, since this causes the Stamp to convert them to ASCII text, which the Stretcher doesn't understand.

The accompanying table lists Stretcher commands with PBASIC examples. Note that input commands, such as Bit, require two PBASIC instructions. The first instruction is a Serout, to send the command to the Stretcher, while the second is a Serin, to receive the data.

Using sleep mode

The Stretcher's Sleep mode reduces current draw to less than 1 mA, not counting any loads it may be driving. The Stretcher awakens periodically to check its serial connection to the Stamp to determine whether it's time to wake up. If the serial line is low, as it normally is, the Stretcher goes back to sleep. If it is high, the Stretcher wakes up and waits for the serial input to go low again. When it does, the Stretcher returns to normal operation.

How often the Stretcher checks for the wake-up signal depends on the value sent with the Sleep (S) command as follows:

Stretcher Sleep Durations

S0	18 ms	S4	288 ms
S1	36 ms	S5	576 ms
S2	72 ms	S6	1152 ms
S3	144 ms	S7	2304 ms

Every time the Stretcher checks for the wake-up signal, any pins that are configured as outputs will float for approximately 18 ms. The internal reset mechanism that periodically wakes the Stretcher to allow it to check the serial line also briefly forces all pins to input mode, causing them to float. Make sure that any loads driven while the Stretcher is asleep can tolerate this glitching. (The Stamp does the same thing during *its* Sleep, Nap, and End modes, by the way.)

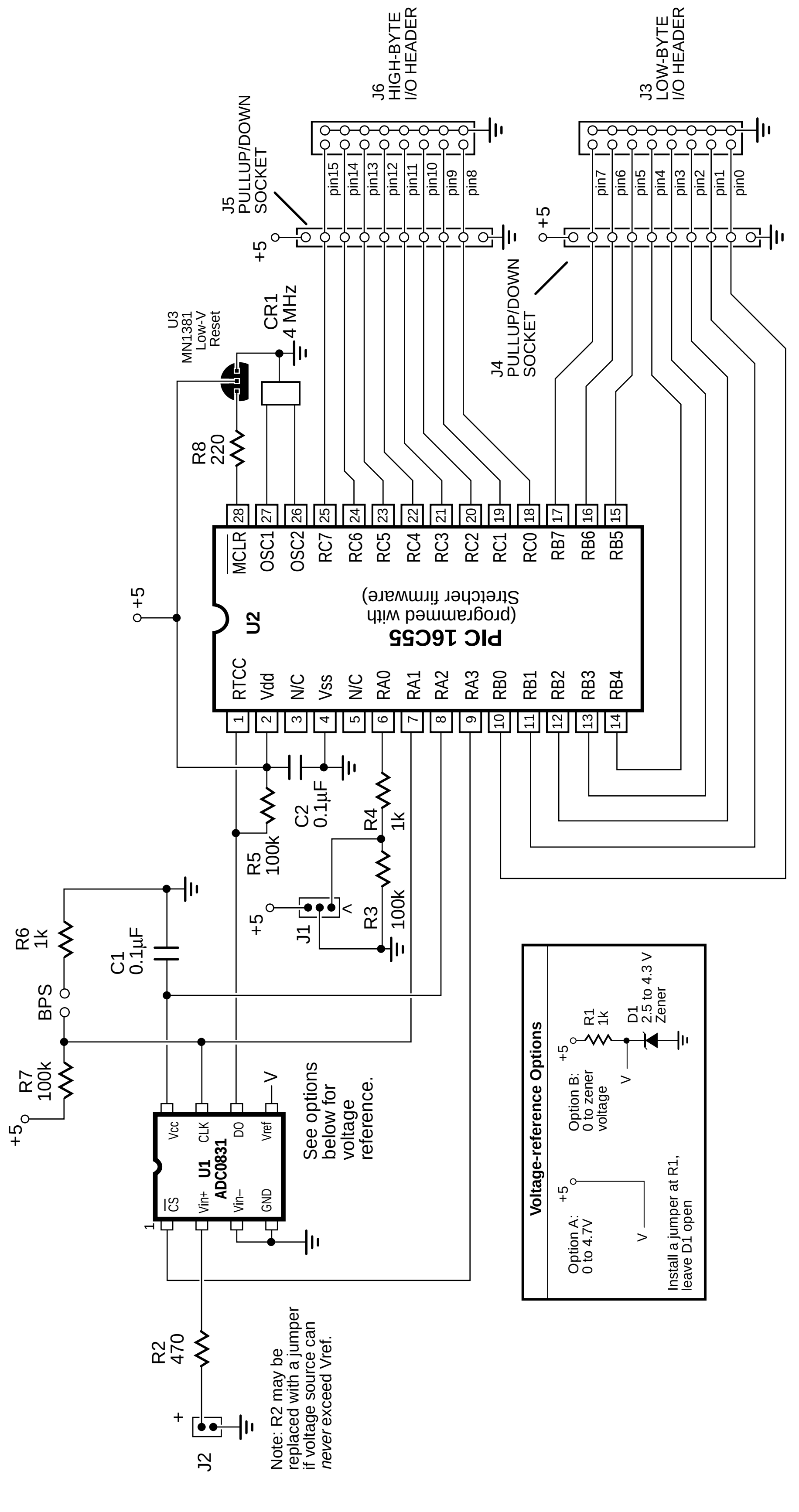
To wake up the Stretcher, the Stamp should output a high on the serial communication line for at least twice (preferably three times) the length of the Stretcher's sleep interval. For best ADC accuracy, don't request an analog reading for at least 50 ms after waking up the Stretcher.

Programming Tips

• Reprogramming the Stamp (ALT-R) does not reset the Stretcher. During program development, you may want to start your program with several reset commands (*) to put the Stretcher into the same state it is in when power is first applied. An alternative is to disconnect power from the Stretcher to cause it to forget previous states before reprogramming. If reprogramming the Stamp interrupts it while it's sending data, the Stretcher may receive the interrupted command incorrectly. Again, the solution is to get the Stretcher into a known state by cycling its power.

• At 2400 baud, it takes about 4.2 ms to send a byte of data. This limits the maximum rate at which you can write data to or get data from the Stretcher's I/O pins. For example, the Toggle instruction takes two bytes, so the fastest you can toggle a Stretcher bit is every 8.4 ms, or approximately 129 times a second (ignoring the Stamp's program-execution time). Try to divide up I/O responsibilities so that the fast jobs go to the Stamp's pins and the slow jobs to the Stretcher. (At 9600 baud, the situation is better, but the Stamp's own I/Os are still faster.)

• Take advantage of automatic data-direction settings. Just like the Stamp, the Stretcher wakes up with all of its pins configured as inputs. The commands High, Low, and Toggle all set the specified pin to output and leave it in that state. The Read and Write commands do not change the data directions of the pins. Values written to pins that are configured as inputs are stored in internal registers of the PIC controller. They don't affect the state of the pins until the pins' data direction is set to output. Likewise, pins that are configured as outputs can be read, but you will be reading the state most recently written to them.



Schematic diagram of the Stamp Stretcher 1B.

Stretcher Input/Output Commands

Command	**Stretcher Syntax**	**Action, Stamp Syntax Example**
High	H [0-15]	Turns the selected pin to an output and writes a 1 to it. `SEROUT 0,N2400,("H",9)  ' Sets Stretcher pin 9.`

Low	L [0-15]	Turns the selected pin to an output and writes a 0 to it.
		`SEROUT 0,N2400,("L",b2) ' Clears Stretcher pin`
		`' whose number is contained in b2.`

Toggle	T [0-15]	Turns the selected pin to an output and toggles (inverts) it. `SEROUT 0,N2400,("T",9)  ' Toggles Stretcher pin 9.`

Bit	B [0-15]	Returns the state (0 or 1) of the specified bit. Does not affect the pins' data direction settings (input or output).
		`SEROUT 0,N2400,("B",15) ' Request status of bit 15.`
		`SERIN 0,N2400,b2        ' Get bit status in b2.`

Analog	A	Returns a reading (0 to 255) from the ADC.
		`SEROUT 0,N2400,("A")    ' Request ADC reading.`
		`SERIN 0,N2400,b2        ' Get ADC data in b2.`

Input	I [0-15]	Makes the specified pin number an input.
		`SEROUT 0,N2400,("I",b2) ' Makes the pin whose number`
		`' is stored in b2, an input.`

Output	O [0-15]	Makes the specified pin number an output. Whatever state was last written to the pin (high or low) will be output when this occurs. If the pin must be in a particular state, use High or Low to make it an output. `SEROUT 0,N2400,("O",3)  ' Makes pin 3 an output.`

Direction, Both	DB	Writes the data direction registers of both of the Stretcher's I/O ports. A binary 1 in the data-direction register makes the corresponding pin an output; a 0 makes it an input. The Stretcher expects the lower port's data direction first.
		`SEROUT 0,N2400,("DB",%00001111,%11111111)`
		`' Make pins 0-3 outputs, 4-7 inputs. Make all of the`
		`' upper port pins outputs.`

Direction, Lower	DL	Writes the data direction register of the Stretcher's lower I/O port as in DB above.
		`SEROUT 0,N2400,("DL",255)`
		`' Make all of lower port outputs (255 = %11111111).`

Direction, Higher	DH	Writes the data direction register of the Stretcher's higher I/O port as in DB above.
		`SEROUT 0,N2400,("DH",255)`
		`' Make all of higher port outputs (255 = %11111111).`

Stretcher Input/Output Commands (cont)

Command	Stretcher Syntax	Action, Stamp Syntax Example

Read Both	RB	Reads both of the Stretcher's I/O ports. Does not affect the pins' data direction settings (input or output). Stretcher returns data as two bytes, lower byte first.
		`SEROUT 0,N2400,("RB")   ' Request read of ports.`
		`SERIN 0,N2400,b2,b3     ' Get Stretcher data.`

Read Lower	RL	Reads the Stretcher's lower I/O port (pins 0-7) as in RB above.
		`SEROUT 0,N2400,("RL")   ' Request read of port.`
		`SERIN 0,N2400,b2        ' Get Stretcher data.`

Read Higher	RH	Reads the Stretcher's higher I/O port (pins 8-15) as in RB above.
		`SEROUT 0,N2400,("RH")   ' Request read of port.`
		`SERIN 0,N2400,b2        ' Get Stretcher data.`

Write Both	WB	Writes both of the Stretcher's I/O ports. Does not affect the pins' data direction settings (input or output). If a pin is set to input at the time of a write, it will not be affected until it is made an output.
		`SEROUT 0,N2400,("WB",b2,b3)   ' Write contents of`
		`' variable b2 to lower port, b3 to higher.`

Write Lower	WL	Writes the Stretcher's lower I/O port (pins 0-7) as in WB above.
		`SEROUT 0,N2400,("WL",255)     ' Write 1s to all pins`
		`' of the lower port.`

Write Higher	WH	Writes the Stretcher's higher I/O port (pins 0-7) as in WB above.
		`SEROUT 0,N2400,("WH",255)     ' Write 1s to all pins`
		`' of the higher port.`

Sleep	S	Puts the Stretcher to sleep. At intervals of 18 ms to 2.3 seconds, the Stretcher will check its serial line for a high, signaling it to wake up.
		`SEROUT 0,N2400,("S",0)  ' Sleep. Look for wakeup`
		`                        ' every 18 ms.`

Reset	*	Returns the Stretcher's I/O pins to their power-up configuration; all inputs, and all internal latches cleared to 0s. Useful during programming to set the Stretcher to a known state. However, if the Stamp is interrupted in the middle of a multipart command, it may take as many as three resets to finish the interrupted command and actually reset the Stretcher. `SEROUT 0,N2400,("*")    ' Reset all pins/directions.`

Technical Support for the Stretcher 1B

Contact the manufacturer:

Scott Edwards Electronics
964 Cactus Wren Lane
Sierra Vista, AZ 85635
phone: 520-459-4802; fax: 520-459-0623
Internet: 72037.2612@compuserve.com
Compuserve: 72037,2612

Converting the program examples for use with the BASIC Stamp 2

The BASIC Stamp 2 (BS2) has Serin and Serout instructions that are similar to those of the BS1, but offer more options. First of all, you can send and receive data at arbitrary data rates up to 50,000 bps, so it's up to you to calculate the bit time and other communication parameters, called "baudmodes" in PBASIC. Here are the baudmodes that apply to communication with the Stretcher 1B:

```
$418d           ' Inverted, 2400 baud.
$4054           ' Inverted, 9600 baud.
```

We won't duplicate the entire instruction table here, just a couple of examples. Suppose you wanted to toggle pin 10 of the Stretcher, and that the Stretcher's serial pin is connected to BS2 pin 0. Assuming the Stretcher is set for 9600 bps (jumper installed on the posts marked BPS), the instruction would be:

```
serout 0,$4054,["T",10]       ' Toggle pin 10.
```

The same pattern applies to all of the other output-only instructions.

To issue an instruction that expects a serial response (B for bit and A for analog), you perform Serout to send the instruction followed by Serin to get the response. For instance, to read the state of Stretcher bit 15, assuming BS2 pin 0 for serial communication, 2400 baud (no jumper at BPS), and the existence of a byte variable named "result," your instructions would be:

```
serout 0,$418d,["B",15]       ' Request state of bit 15.
serin 0,$418d,[result]        ' Get the response.
```

The BS2 flavors of Serin and Serout have many other options. See the Parallax documentation for details.