What Beasties Cannot Do
And Some Things That They Can

Asher Hoskins
## Contents

1 Introduction 2

2 What’s in a Beastie? 2
  2.1 The Beastie Bus 2
  2.2 Inputs 4
  2.3 Outputs 4
    2.3.1 Producing Analogue Voltages 5
  2.4 Analogue Comparator 6
  2.5 Serial Interfaces 6
  2.6 Onboard External Memory 6
  2.7 Configuration Jumpers 6
  2.8 The Aerial 6

3 Power Supply and Consumption 7
  3.1 Supply 7
  3.2 Power on the Beastie Bus 7
  3.3 Consumption 7
  3.4 Batteries 9
  3.5 Lifetime 9

4 Running Programs 10
  4.1 Bits, Bytes, and KBytes 10
  4.2 How Much Can A Beastie Store? 11
  4.3 The Speed of Beasties 12

5 Wireless Networking 12
  5.1 Data Rates 12
  5.2 Transmission Range 13
  5.3 The Network 13

6 Connecting Beasties to Other Computers 13
  6.1 Serial Interface 13
  6.2 Internet Bridges 13
  6.3 Ethernet/WiFi Interfaces 14

7 Summary and Specifications 14
1 Introduction

This document is intended to give a non-technical overview of the capabilities of the Beastie One and Beastie OnePlus computers (shown in figure 1). What they can do, what they can’t do, and how they can (and cannot) be used. For a quick overview of their main features, see the specifications list in section 7.

2 What’s in a Beastie?

The ‘Beastie’ is a small computer designed for use as the core of a wireless sensor node. Its main functional blocks are shown in figure 2. At the moment there are two varieties of Beastie: the ‘One’ and ‘OnePlus’. The OnePlus is identical to the One except for the addition of 32 kB of non-volatile memory (memory that doesn’t lose its contents when the power is removed, as opposed to ‘volatile’ memory, which does) external to the processor.

2.1 The Beastie Bus

On its own, a Beastie doesn’t do very much. It can run programs and communicate with the wireless network but that’s about all (one of the few uses of a “naked” Beastie would be as a network repeater: extending the range of a network by receiving messages from the edge of one network and then resending them into another network). The Beastie Bus socket (a ‘bus’ is just a collection of related signals) on the back of each Beastie gives access to a wide range of inputs and outputs to which sensors and actuators can be attached. These are described in full in table 1 (a couple of the pin names in this table have bars over them, like $\overline{THIS}$; this notation just means that they are activated by a low voltage rather than a high voltage — you don’t have to worry about what this means unless you’re actually designing circuits to plug into a Beastie). Pin 1 is at the top of the bus when you’re looking at the Beastie from above, with the bus on the right. Note that most of the connections on the bus have two possible uses. Normally each connection will only be used in one of its possible connection modes at any given time (although it is possible, at the expense of more complex electronics, to have connections which can change mode in a running system).
Table 1: The Beastie Bus.
Peripheral boards, or ‘daughterboards’, are generally attached to the Beastie Bus via a ‘backplane’: a circuit board containing several rows of plugs which connect the bus of the Beastie to the buses of the daughterboards when they are stacked above each other as shown in figure 3.

2.2 Inputs

The Beastie’s ‘analogue’ inputs can measure any voltage between zero and five Volts in steps of about 5 mV (‘mV’ is ‘milli Volts’, thousandths of a Volt), the ‘digital’ inputs merely measure whether there are roughly zero Volts on the input or roughly five Volts. Digital inputs are much faster: with the Beastie running at 4 MHz (see section 4.3 on speed later on in this document) it can take up to 9,600 analogue measurements a second or up to 570,000 digital measurements.

If the Beastie is taking as many analogue measurements per second as possible its processor is actually spending most of its time asleep waiting for the measurement to take place (the analogue measurement hardware is a separate part of the Beastie’s processor chip to the main processor) and so it can continue to send and receive messages over the network and do other processing. If it is taking as many digital measurements per second as possible it will be spending all its time storing these measurements into memory and so won’t be able to do anything else at the same time.

Note that the analogue inputs PC4 and PC5 are also used to make up the I2C serial interface (see section 2.5 on serial interfaces). As well as being available on the Beastie Bus, the I²C interface is used to connect to the onboard external memory on a Beastie OnePlus (see section 2.6) and so these connections cannot be used as analogue inputs unless the memory is disabled by removing two configuration jumpers (see section 2.7).

2.3 Outputs

The Beastie has no analogue outputs (but this doesn’t mean that you can’t produce analogue voltages with a Beastie, see below): all its outputs are set to either zero Volts or five Volts. It can change these outputs at the same speed that it can take digital measurements, up to 570,000 times a second when running at 4 MHz. Each output can supply up to 40 mA (‘milli Amps’) current as long as the total output current does not exceed 100 mA for the pins PC0–PC5 and 100 mA for all the other output pins combined. An average LED light, for example, will take about 20 mA to light (although you can get low current LEDs which take as little as 2 mA to light). See section 3 on power for more about milli Amps and what they mean.

To drive high current outputs such as motors, relays have to be used. Two types of easily obtainable relay (‘reed’ relays which have a very lightweight switch inside them, and ‘solid state’
2.3.1 Producing Analogue Voltages

Even though the Beastie only has digital outputs there are a couple of ways in which it can produce analogue voltages. The first is by a method called ‘pulse width modulation’ (PWM). This method uses software to switch a digital output on and off quickly. A simple bit of electronics allows us to take the average value of this on/off signal to produce an analogue voltage. By changing the ratio of on to off in the series of pulses we can change the analogue voltage. For example, in the left hand side graph in figure 4 the output is on (five Volts) for the same amount of time that it is off (zero Volts). The average voltage is therefore 2.5 Volts (5 × 50% = 2.5). In the right hand graph the output is on for only a quarter of the time and therefore the average voltage is 1.25 Volts (5 × 25% = 1.25).

The second method for producing an analogue voltage uses a type of chip called a ‘digital to analogue convertor’ (DAC). These come in a variety of types, with varying precision and numbers of analogue outputs. They are connected to the digital outputs of the Beastie which sends instructions to them to set their output voltage. The number of digital outputs that are needed to control them varies from chip to chip but it can be as few as two for I²C connected DACs. Producing analogue voltages this way has the great advantage that the Beastie’s processor can idle once it has set the DAC to a particular level. With pulse width modulation it normally has to keep running in active mode so that it can switch the outputs on and off. If several analogue outputs are required then the software to produce PWM pulses for all of them can get complex, whereas with a DAC all that is required is a series of commands to set the voltages.
2.4 Analogue Comparator

The ‘analogue comparator’ can compare two voltages on the pins AIN0 (PD6) and AIN1 (PD7) and can tell whether or not the voltage on AIN0 is higher than the voltage on AIN1. Each voltage must be in the range zero to five Volts.

2.5 Serial Interfaces

There are three different types of serial interface on the Beastie Bus: RS232, I²C, and SPI. RS232 is the standard serial interface found on the back of a PC or older Mac. I²C and SPI are interfaces designed for linking chips together.

To use the RS232 serial port on a Beastie an accurate clock signal needs to be supplied via the bus (a small electronic module costing about £3 and drawing about 20 mA can do this¹) and the RXD and TXD signals must be put through a level convertor (another low cost chip) to convert them from the 0 V and 5 V bus levels to the −10 V to +10 V levels that the RS232 standard requires. It is hoped that a standard daughterboard for Beasties containing all the parts required for an RS232 interface will be produced shortly.

The I²C and SPI interfaces are designed for linking chips together (on a Beastie OnePlus the I²C interface is also what connects the onboard external memory to the processor) and as such are beyond the scope of this document. For reference though, with the Beastie running at 4 MHz it can run the I²C interface in the 100 kbit/s “standard” mode, running at 8 MHz it can run the interface in the 400 kbit/s “fast” mode; the SPI interface can run at up to 1 Mbit/s when the Beastie is running at 4 MHz. The SPI interface is also used to load programs into the Beastie and circuit designers who use the SPI bus should be aware of this.

2.6 Onboard External Memory

In addition to the memory built into the processor (see section 4.2), the Beastie OnePlus has an additional 32 kB memory chip. This memory is ‘non-volatile’, it doesn’t lose its contents when the power is switched off.

This memory is connected to the processor via the I²C interface (at address 1010000₂) and therefore the speed at which it can be accessed depends on the speed that this interface can run at. With the interface running in standard mode on a 4 MHz Beastie, accessing this memory is about 180 times slower than accessing the internal processor data memory. With the interface running in fast mode on an 8 MHz Beastie access is about 90 times slower.

2.7 Configuration Jumpers

At the front of a Beastie is a bank of four configuration jumpers. These are numbered from 1 to 4, with jumper 1 on the right hand side of the bank when looking at the front of the Beastie. See table 2 for their functions. A jumper is said to be ‘on’ when the shorting link is fitted, and ‘off’ when it is removed (see figure 5). Beasties are supplied with all jumpers ‘on’.

2.8 The Aerial

The aerial on a Beastie can plug directly into the aerial socket or be mounted remotely and connected to the socket via a length of coaxial cable. There are two main types of aerial that can be used with a Beastie: whip (just a piece of wire about 16 cm long) or helical (a 5 mm

¹For example, RS part number 226-1976.
<table>
<thead>
<tr>
<th>Jumper</th>
<th>Function when on</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Enable onboard power supply</td>
</tr>
<tr>
<td>2</td>
<td>Enable RESET signal from Beastie Bus</td>
</tr>
<tr>
<td>3 &amp; 4</td>
<td>Enable onboard external memory (OnePlus only)</td>
</tr>
</tbody>
</table>

Table 2: Configuration jumpers.

Figure 5: Beastie configuration jumpers. Jumpers 3 and 4 are off, jumpers 1 and 2 are on.

A diameter coil of wire about 3.5 cm long. Both types of aerial can be put inside a plastic or rubber moulding to make them more robust.

3 Power Supply and Consumption

3.1 Supply

Power is normally supplied to a Beastie through a flying lead attached to the top-left corner of the circuit board. This normally has a PP3 battery connector on it, although there’s no reason why a different form of connector couldn’t be substituted.

The input voltage must be between 7 V and 20 V DC. This is regulated onboard the Beastie so if it is to be run off the mains a cheap unregulated DC power supply can be used. The Beastie’s power supply is limited by a self-resetting fuse to 500 mA. If the DC power supply that is feeding the Beastie has a lower power rating than 500 mA then it may need to be independently protected so that the Beastie cannot overload it.

It is also possible to disable the onboard power supply and feed a regulated 5 V DC power supply in via the VCC and GND pins on the Beastie Bus.

3.2 Power on the Beastie Bus

The VCC and GND pins on the Beastie Bus supply +5 V and 0 V power connections respectively. The amount of power that can be drawn from these pins depends on the amount of power that is being used by the Beastie to run itself and the radio but it is normally safe to assume that at least 400 mA will be available.

3.3 Consumption

There are three main consumers of power on a Beastie: the processor, the radio module, and the LED indicator light.
The faster a Beastie runs, the more power its processor uses. When the processor is running program code it is said to be ‘active’ and uses the most power. It can also be switched into ‘idle’ mode to reduce power consumption. In this mode it stops running the current program until it is woken up by an event such as a timer overflow or an external interrupt. The amount of power the processor uses in these two modes for various clock speeds is shown in figure 6.

The processor can also be put into a power-saving mode in which it will stop completely, only waking on an external interrupt. The power usage in this mode is so tiny it wasn’t worth plotting it on the graph.

The standard processor on a Beastie (an Atmel ATmega8L) can run at 1, 2, 4, or 8 MHz without any external components. If the processor is changed to the higher power ATmega8 and an external oscillator is used then it can run at up to 16 MHz. Note that if an external oscillator is used then the power consumption of that (typically 20 mA) must be added to the power consumption figure shown on the graph. Any external oscillator will stay running (and consuming power) even if the processor is idling or power-saving.

The Beastie radio protocol is still being worked on. Any figures given for power consumption in this section should be considered extremely preliminary.

The data transmission capabilities of the Beastie’s radio module are described in section 5. This section will talk about power consumption only.

The radio module has three possible states: transmitting, receiving, and standby. When transmitting it draws 26 mA, when receiving it draws 12 mA, and when in standby it only draws 8 μA (8 ‘micro Amps’, or millionths of an Amp).

In order to conserve battery power, the Beastie does not have its receiver turned on all the time. It instead switches it on for just long enough to see if there is a radio message being transmitted, if not it switches it off again. At the moment (this is provisional), the receiver is on for about 25% of the time, giving an average power consumption of 3 mA. This power consumption will increase as network traffic increases since the Beastie will have to listen more since it has to receive every message that is transmitted within range in order to see if the message is addressed to it.

The LED on a Beastie has only two states: on or off. When it’s switched off it draws no
power and when it’s switched on it draws 10 mA. The LED is frequently used to show when the radio is transmitting as an aid to debugging communications problems.

### 3.4 Batteries

There are many different types of battery and the Beastie doesn’t care which type is used as long as it gets between 7 V and 20 V supplied to its power supply. A selection of common types and their storage capacities are shown in table 3. Battery storage capacities are measured in ‘milli Amp hours’ (mAh). Roughly, if a battery has a capacity of 100 mAh then it can supply 100 mA for one hour, 50 mA for two hours, 1 mA for one hundred hours, and so on. In reality things aren’t quite that neat but this model will do for us here. Batteries that supply less than 7 V can be wired together to increase the total voltage. For example, six 1.5 V AA batteries wired together will produce 9 V.

### 3.5 Lifetime

As standard, the Beastie has no way to measure its battery voltage, it’ll just stop running when the power runs out. It is possible to add battery measurement to the Beastie via a daughterboard though (the voltage can be scaled so that it fits into the 0–5 V range of the Beastie’s analogue inputs). It’s normally only worth doing this if the Beastie has some strategy for prolonging battery life once it sees it is running out.

Figure 7 gives some idea of the lifetimes that can be expected from a Beastie for various battery types and processor clock speeds. These graphs assume that a Beastie has its radio in receive mode for 25% of the time and that the processor is active for 10% of the time and idle the rest. It may seem strange that the Beasties are idle for 90% of the time but this is normal behaviour for a system that only needs to take measurements occasionally and which uses most of its battery power listening for messages on the wireless network. It’s assumed that the transmission of messages on the network takes place rarely enough that it’s not worth including in the calculations. For clock speeds over 8 MHz the additional 20 mA drain of an external oscillator is taken account of.

Power consumption will increase and lifetime will decrease if the Beastie has to do more processing (and so spend less time idle), if there is a lot of network traffic, or if the Beastie has

<table>
<thead>
<tr>
<th>Make</th>
<th>Type</th>
<th>Size</th>
<th>Voltage</th>
<th>Capacity (mAh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-rechargeable Duracell Procell</td>
<td>Alkaline</td>
<td>PP3</td>
<td>9</td>
<td>550</td>
</tr>
<tr>
<td>Duracell Procell</td>
<td>Alkaline</td>
<td>AAA</td>
<td>1.5</td>
<td>1175</td>
</tr>
<tr>
<td>Duracell Procell</td>
<td>Alkaline</td>
<td>AA</td>
<td>1.5</td>
<td>2700</td>
</tr>
<tr>
<td>Duracell Procell</td>
<td>Alkaline</td>
<td>C</td>
<td>1.5</td>
<td>7750</td>
</tr>
<tr>
<td>Duracell Procell</td>
<td>Alkaline</td>
<td>D</td>
<td>1.5</td>
<td>18000</td>
</tr>
<tr>
<td>Energizer</td>
<td>Lithium</td>
<td>AA</td>
<td>1.5</td>
<td>3000</td>
</tr>
<tr>
<td>Sonnenschein</td>
<td>Lithium</td>
<td>C</td>
<td>1.5</td>
<td>8500</td>
</tr>
<tr>
<td>Sonnenschein</td>
<td>Lithium</td>
<td>D</td>
<td>1.5</td>
<td>19000</td>
</tr>
<tr>
<td>Rechargeable Fuji</td>
<td>Lithium ion</td>
<td></td>
<td>3.6</td>
<td>1850</td>
</tr>
<tr>
<td>Yuasa</td>
<td>Lead-acid</td>
<td></td>
<td>12</td>
<td>65000</td>
</tr>
</tbody>
</table>

Table 3: Some common batteries capacities.
Figure 7: Beastie lifetime for various battery types.

Table 4: Some number types.

<table>
<thead>
<tr>
<th>Bytes</th>
<th>Bits</th>
<th>Name</th>
<th>Unsigned maximum</th>
<th>Signed range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>Byte</td>
<td>255</td>
<td>−128 – 127</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>Short</td>
<td>65535</td>
<td>−32768 – 32767</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
<td>Int</td>
<td>4294967295</td>
<td>−2147483648 – 2147483647</td>
</tr>
</tbody>
</table>

4 Running Programs

This section doesn’t cover actually writing programs for Beasties and uploading them but rather tries to give an idea of what sorts of programs Beasties can run. The processor in a Beastie is much smaller than the processor in a desktop computer and consumes a tiny fraction of the power. It therefore can’t do as much or do it as quickly.

4.1 Bits, Bytes, and KBytes

The smallest unit of information that a computer can handle is the binary digit, or ‘bit’. This can hold the values zero or one (contrast with the decimal digit which can hold the values zero through to nine). An eight digit binary number, or ‘8-bit’ number, is called a ‘byte’ and can hold the integer (whole number) values 0 to 255 (256 different values, since $2^8 = 256$; an eight digit decimal number can hold the values 0 to 99,999,999, or $10^8$ different values).

Since computers are typically able to hold thousands of millions of different bytes in their memory and on their hard discs several prefixes are used to keep the numbers manageable. A ‘kilobyte’, ‘kB’, or frequently just ‘k’, is 1024 bytes ($1024 = 2^{10}$); a ‘megabyte’, ‘MB’, or ‘M’, is 1024 kilobytes ($2^{20}$ bytes); and a ‘gigabyte’, ‘GB’, or ‘G’, is 1024 megabytes ($2^{30}$ bytes).

Since it’s frequently useful to be able to deal with numbers outside of the range 0–255 bytes can be combined together to make larger numbers. The ranges of these numbers when they are used for storing positive integers only (‘unsigned’ numbers) and positive or negative integers (‘signed’ numbers) are summarised in table 4.
There are two main standards for storing numbers with a fractional part (‘floating point’ numbers) which take up four or eight bytes per number. Unfortunately the Beastie’s processor cannot deal with these types of number directly and the programs to enable it to do so do not fit into its memory. This is rarely a disadvantage however since almost every algorithm can be rewritten to use integers only.

Storing text is another common requirement. Without compression (which can reduce storage sizes significantly depending upon the compression method used) it takes a single byte to store a single text character from a western European language. For languages with lots of characters like Japanese it is usually necessary to use two bytes per characters.

4.2 How Much Can A Beastie Store?

A Beastie has several different types of memory: program, data, EEPROM, and, on a OnePlus, onboard external memory. All the memory apart from the external memory is built into the Beastie’s processor.

Program code must be stored in the 8 kB ‘program memory’. Of this 8 kB about 6–7 kB will normally be available to the programmer (the rest is taken up with support routines for the wireless network and suchlike). Most program instructions are two bytes long and so this equates to 3000–3500 instructions, with each instruction generally able to handle a single byte of data. The program memory cannot be changed while the Beastie is running.

It’s very hard to say exactly what can be done with this amount of memory, especially since programmers rarely work at the machine instruction level, relying on compilers to generate these instructions for them from higher level languages. As a rough idea though, programs that capture perhaps 0.5 kB of data per second, process and filter it down to 30–40 bytes, and then transmit this over the network should be well within the capabilities of a Beastie.

A single Beastie can contain and run several “programs” at once (in reality they’ll have been put together as a single program), sharing resources between the programs: there’s no need to think of each Beastie as only being able to perform a single task.

If the limited program memory space in the Beastie becomes a problem then there is a solution at the expense of reduced program speed. It is possible to design a simple pseudo-machine language which can be stored in the onboard external memory or in a memory on a daughterboard (the maximum size of which is essentially unlimited). This pseudo-language can then be interpreted by a program running on the Beastie itself. Running a program this way is much slower due to the slow access speed of external memories but allows large programs with large datasets to run.

The ‘data memory’ of the Beastie can hold 1 kB of data. It does not retain data when the power is removed but can be read from and written to very quickly.

The ‘EEPROM’ (electrically-erasable programmable read-only memory) can hold 512 bytes of data, which is retained even if the power is removed. This memory is about four times slower than data memory for reading and about five times slower than data memory for writing. More seriously however, each location in the EEPROM can only be written about 100,000 times before it “wears out” (this is a design limitation common to all EEPROM memories). This must be taken account of by the programmer. There is no limit to the number of times the memory can be read from.

The Beastie OnePlus has an external memory chip which can hold 32 kB of data. It will retain this data when the power is removed and has no limit on the number of times it can be written to. It is however much slower than the data memory: 180 times slower on a 4 MHz Beastie and 90 times slower on an 8 MHz Beastie.
4.3 The Speed of Beasties

There’s been a lot of talk of ‘MHz’, or ‘mega Hertz’, in this document so far. The Hertz is a measure of frequency: 1 Hz means one cycle per second. For various obscure reasons ‘mega’, ‘giga’, and other prefixes are multiples of 1000, not 1024, when talking about time rather than memory, so 1 MHz is the same as 1,000,000 Hz.

All computers have a ‘clock’ which synchronises their internal operation. The faster this clock goes the faster they can perform computations. The standard Beastie runs at 4 MHz (compare this with the typical 2 GHz (2 giga Hertz) processor speed of a modern PC: 500 times faster but vastly more power consuming) but can go down as slow as 1 MHz or as fast as 16 MHz (speeds above 8 MHz require the processor chip to be replaced with a high-speed version and the addition of an external clock generator). The processor used in the Beasties performs about one instruction every clock cycle, so a 4 MHz Beastie will be able to execute about four million instructions every second.

As with the size of programs, it’s very hard to estimate just what can be done given a Beastie of a particular speed. It’s probably safe to assume that hundreds to a few thousand measurements can be taken every second and simple processing performed upon them by a standard 4 MHz Beastie. This means that measuring something like a flickering light is possible, as is processing text (as long as it fits in the data memory) but that receiving the input from a microphone at anything other than a low quality or processing video is not possible.

5 Wireless Networking

The Beastie radio protocol is still being worked on. Any figures given for data rates or power consumption in this section should be considered extremely preliminary.

5.1 Data Rates

The speed at which data can be transferred over a network is measured in ‘bits per second’ (bit/s) or ‘kilo bits per second’ (kbit/s). As when talking about clock speeds, the prefixes are multiples of 1000, so the Beastie’s data speed of 5 kbit/s is the same as 5000 bit/s.

To make things even more complex, you can’t just divide by eight to get the number of bytes per second of data that could be transferred. Each set of data transmitted by the Beastie needs to be wrapped up into a ‘packet’ with special information at the beginning and end so that other Beasties can recognise it as a packet and know whether or not it is addressed to them. A packet transmitted by a Beastie has 24 bytes of wrapping.

There’s also a limit of the amount of data that can be fitted into a single packet. For Beasties this is 128 bytes (made up of 104 bytes data and 24 bytes wrapping). Each packet of data has to be acknowledged by the device that the data is being sent to so that the Beastie knows that it has made it through the network successfully. An acknowledgement packet is 24 bytes long and is received after an average 20 bytes delay (for reasons to do with the technique that the Beasties use to allow multiple Beasties to share the same radio channel). If all this is added up we can see that to send 104 bytes of actual data requires 172 bytes to be transmitted over the radio.

Given these numbers (which, as stated above, are very provisional and subject to change) we can calculate a maximum data rate of 378 bytes per second. This is equivalent to 189 16-bit ‘short’ numbers, 94 32-bit ‘int’ numbers, or 378 text characters. This data rate only applies when
no other devices are trying to use the network and will drop as other devices share the radio channel to send their data. A rough guess at a worst case data rate would be about 25% of the maximum, or 94 bytes per second.

These data rates sound appalling to someone accustomed to the megabytes per second that a modern PC network delivers but remember that the radio on the Beasties is very low power and that the amount of data they will be transmitting will generally be low — the data output from a typical building sensor like a thermometer will be measured in bytes per hour.

5.2 Transmission Range

The radio on the Beasties has a stated maximum range of two hundred metres. This range could only be achieved if there was a direct line of sight between two Beasties with no obstacles in between. In reality the actual range is somewhat less than that. Indoors, surrounded by walls, furniture, and electrical equipment, the range is around twenty metres.

The maximum range in a given situation can only really be discovered through experimentation. It’s virtually impossible to calculate just how far radio waves can travel through a building since there are so many things that can absorb or bounce them around.

Fortunately, the network used by the Beasties allows packets of data to travel from one Beastie to another via intermediate Beasties, greatly extending the range over which packets can be sent.

5.3 The Network

Beasties are connected together in ‘networks’. These networks are self-configuring and adapt automatically when new Beasties are introduced or old Beasties removed. Packets can be sent from any Beastie on the network to any other. It is not possible, except in special circumstances, to send a single packet from one Beastie to multiple Beasties at the same time. It has to be resent to each Beastie in turn.

6 Connecting Beasties to Other Computers

6.1 Serial Interface

The easiest way to get data from a Beastie to another computer is to use a serial interface daughterboard. This provides a standard 9-pin RS232 socket which can be connected to the serial port of a PC. To connect this to modern Macs and laptops without an RS232 interface a USB to serial adaptor can be used.

The standard RS232 data rates which are possible at various clock speeds are shown in table 5. As well as these standard rates, Beasties can run at some higher nonstandard rates (up to 500 kbit/s on a 4 MHz Beastie). Whether or not these can be used depends on the computer that the Beastie is connected to.

6.2 Internet Bridges

If a Beastie is connected to a computer which is connected to the internet then it is possible for that computer to act as a ‘bridge’, taking packets received by the Beastie and transmitting them over the internet to another computer. If this computer is also connected to a Beastie then it can cause the Beastie to resend the packet that came over the internet into its own Beastie network.
In this way the area covered by a single Beastie network can be greatly extended, crossing towns or even countries with ease.

### 6.3 Ethernet/WiFi Interfaces

*The devices mentioned in this section have not yet been tried with Beasties.*

It is also possible to connect Beasties directly to ethernet or WiFi networks using an adaptor on a daughterboard. These adaptors are much smaller (generally only about a quarter of the area of a Beastie and about 15–20 mm thick) and cheaper than a desktop computer. Their power usage is such that the Beastie will probably have to be powered from a large battery or a mains adaptor in order to run for any sensible length of time.

The exact capabilities of these devices when connected to Beasties have still to be worked out but at the very least they should allow bridging between Beastie sub-networks via the internet.

### 7 Summary and Specifications

**Processor**
- **Type**: Atmel ATmega8L
- **Data width**: 8 bits
- **Speed**: 1–16 MHz

**Memory**
- **Program**: 8 kB
- **Data (SRAM)**: 1 kB
- **EEPROM**: 512 B
- **External data memory**: 32 kB

**Interfaces**
- **Digital input/outputs**: 16
- **Analogue inputs**: 6
- **Analogue comparator**: 1
- **Serial interfaces**: RS232, I^2^C, SPI

**Typical Battery Lifetimes (4 MHz)**
- **PP3**: 85 hours
- **AAA**: 185 hours

<table>
<thead>
<tr>
<th>Clock</th>
<th>2400</th>
<th>4800</th>
<th>9600</th>
<th>14.4k</th>
<th>19.2k</th>
<th>28.8k</th>
<th>38.4k</th>
<th>57.6k</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 MHz</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2 MHz</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4 MHz</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>8 MHz</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>16 MHz</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 5: RS232 rates possible at various clock speeds.
AA 420 hours
Lead acid 10155 hours (14 months)

Network
Range outdoors (maximum) 200 metres
Range indoors (typical) 20 metres
Data rate (maximum) 378 bytes/second
Data rate (busy network) 94 bytes/second

Notes
1. For speeds above 8 MHz the processor must be changed to an ATmega8 and an external clock oscillator must be used.
2. Only available on Beastie OnePlus.
3. Not all interfaces can be used at the same time since some share pins with others.