Questions regarding operation and use:

What is MicroStamp11?

MicroStamp11 is a tiny microcontroller module which can be programmed using BASIC, C, assembler, Forth, or Pascal programming language (or any other language available for the 68HC11). It is about the size of a commemorative postage stamp, hence it was dubbed "stamp".

How does MicroStamp11 compare with the BASIC Stamp?

There are many fundamental differences. See comparison chart for details.

Does MicroStamp11 lose its program when I disconnect the battery or power supply?

No, your code is stored inside an EEPROM on-board MicroStamp11. EEPROMs provide non-volatile storage; they retain memory even without power. The EEPROM used in MicroStamp11 is guaranteed to function properly for at least 10 years and for 1,000,000 write cycles per memory location.

How big is MicroStamp11?

MicroStamp11 measures 1.4" (36 mm) L x 1.0" (25 mm) W x 0.4" (10 mm) H, excluding the connector.

How do I power MicroStamp11?

MicroStamp11 runs on 5 to 15 Volts DC. It features an on-board 5-Volt regulator which converts the input 6 to 15 Volts (on the Vin pin) to the 5 Volts that its components require. If your power supply is 6 to 15 Volts, you should connect it directly to the VIN and GND pins (10 and 11 on MicroStamp11).
If your power supply delivers a regulated 5 Volts, you should connect it directly to the +5V and GND pins (12 and 11 on MicroStamp11).

How much current does MicroStamp11 draw?

MicroStamp11 consumes approximately 15 mA in running mode and 20 uA in sleep mode not including any circuitry on the I/O pins.

Is MicroStamp11 sensitive to static electricity?

While many electronic devices, including MicroStamp11, can be damaged by static electricity, MicroStamp11 is generally more tolerant of static. We do, however, recommend taking all the usual precautions when handling MicroStamp11s in static prone environments.

Can MicroStamp11 be embedded into my product for volume manufacturing?

Yes. Technological Arts offers MicroStamp11 modules in volume at a discounted price for economical integration into your products. A variety of connector options is offered, to suit your needs.

What items do I need to get started with MicroStamp11?

The items you need are:

1. programming software to develop your application
2. Docking Module with serial cable
3. MicroStamp11 module
4. program which loads your application into MicroStamp11's memory
5. a ribbon cable/breadboard adapter assembly
6. manual

If you are new to MicroStamp11, it is best to purchase a MicroStamp11 Starter Package. This package includes all six of the items listed above at a lower cost than if you were to purchase them separately.

What kind of microcontroller chip is used in MicroStamp11?

MicroStamp11 uses an MC68HC11D0, a high-performance CMOS chip made by Freescale.

What kind of environment can MicroStamp11 operate in?

MicroStamp11 modules will work over a temperature range of 0° to +70° C, with up to 70%, non-condensing humidity. While the modules may continue to function outside these ranges, it is not guaranteed or recommended by Technological Arts. Additionally, it is best to keep MicroStamp11 modules away from, or shielded from, any nearby RF interference as this may impact
the accuracy of the I/O functions.

How many I/O pins does MicroStamp11 have?

MicroStamp11 has 14 multi-purpose I/O pins plus 2 hardware interrupt pins.

What can I use the I/O pins for?

MicroStamp11's I/O pins are perfectly suited to digital input and output with TTL/CMOS level (0 to 5 Volt) signals. In addition to basic I/O, all of the pins have alternate functions related to a powerful set of on-chip hardware subsystems. These subsystems include SCI, SPI, 16-bit timer/counter, and RTI.

Can the I/O pins be used to control relays, solenoids, motors, and other similar devices?

Yes, however, due to the demanding current and voltage requirements of some of these components, driver circuitry will be required in order to properly isolate the I/O pins from harmful effects.

Can I control LEDs with the I/O pins?

Yes. Simply use a 330 ohm resistor in series with the LED to limit the current draw through the I/O pin.

How much current can the I/O pins handle?

Each I/O pin is capable of sourcing or sinking a maximum of 25 mA; however, the total across all I/O pins should not exceed 80 mA source or sink at any given time.

Can MicroStamp11 generate analog voltages?

Generating an analog output is very simple, using a PWM waveform in conjunction with a resistor and capacitor. Alternatively, a DAC can be added via the SPI port (see below).

What can I use the SPI pins for?

The Serial Peripheral Interface (SPI- pronounced "spy") opens up a whole realm of possibilities for I/O expansion. Some possibilities are:
multi-channel analog-to-digital conversion, non-volatile data storage (serial EEPROM), real-time clock/calendar functions, LCD interface, and a virtually unlimited number of additional input and output lines.

How fast does MicroStamp11 execute code?

The MCU in atandard MicroStamp11 has a bus speed of 2MHz. This translates to an instruction cycle time of 500ns. Most 68HC11 instructions execute in 2 to 9 cycles, with an average of around 4 cycles. This means MicroStamp11 can execute code at an average speed of 500,000 instructions per second (0.5 MIPS). The true execution speed depends upon many factors, including the particular
set of instructions used and their addressing modes. A "Turbo" version of MicroStamp11, operating at 9.8304 MHz, is now available.
This gives a 23% speed increase.

What is the Vin pin used for?

The Vin (Voltage input) pin is used to power MicroStamp11 from a 6 to 15 Volt source. The Vin pin is the positive connection while the GROUND pin is the negative, or ground, connection. When powered from the Vin pin MicroStamp11 regulates the voltage down and outputs +5 Volts on the +5VDC pin.

How does the +5VDC pin work?

The +5VDC pin (pin 12) outputs 5 Volts when MicroStamp11 is powered by a 6 to 15 Volt source using the Vin and GROUND (pin 11) pins. The +5VDC pin can be used to power other circuitry provided that the overall current consumption is within the capabilities of MicroStamp11's regulator.

How does the reset pin (RESET*) work?

The reset pin is internally controlled by MicroStamp11's low-voltage inhibit circuit. It is normally high (+5V), which allows MicroStamp11 to run its program, and is pulled low when the power supply voltage drops below 4 Volts, which safely puts MicroStamp11 in a reset state. This pin can be monitored to detect when a reset condition occurs, or, you can pull the line to ground to force a reset (via the reset button on the Docking Module, for example). After the reset pin is allowed to rise to +5 Volts, MicroStamp11 wakes up and starts executing its program from the first line of code. Do not drive this pin high-- it should be left electrically disconnected (floating) when you want MicroStamp11 to run normally.

What are the slide switches for?

During programming, the MCU needs to be put in a Special Bootstrap Mode so that the program can be loaded in via the serial port. This position is labelled LOAD on the RUN/LOAD switch (SW2). The other switch (SW3) is for protecting the contents of EEPROM from corruption due to power glitches. During programming, slide the switch to the LOAD position. After the code has been loaded, slide both switches away from the LOAD position (ie. to RUN and PROT). Reset the board to cause the MCU to begin executing your program.

How much current can MicroStamp11's on-board regulator provide?

The 5-Volt regulator built into MicroStamp11 can supply 100 mA current if powered by a 6 Volt source. The limit is affected by operating temperature, so if the unit is running significantly above room temperature, the maximum current it can supply will be reduced. The amount it can supply will also be reduced if you use a higher input voltage. This is because the regulator drops the voltage to 5 Volts by dissipating the excess voltage as heat. MicroStamp11 draws approximately 15 mA in RUN mode, leaving 85 mA for use with other

circuitry via the +5VDC pin and the I/O pins.

Can MicroStamp11 generate a sine wave?

Yes. MicroStamp11 can be made to generate a sine wave by means of a continually varying pulse-width on an output pin. By adding an RC circuit to average out the PWM waveform over time, the result is a sine wave. The same technique can be used to generate other types of waveforms as well.

Can MicroStamp11 generate a waveform on two pins at once?

Yes. MicroStamp11 is a multi-tasking device, utilizing a hardware timer system and interrupts to generate waveforms independently, and thus can generate more than one waveform at a time (besides doing other tasks).

Can MicroStamp11 generate square waves?

Yes, it is very easy to do this using the Output Compare feature of the hardware timer subsystem.

Can MicroStamp11 vary the pulse-width of a square wave?

The duty cycle of the square wave can easily be varied to produce a PWM waveform. Varying the pulse-width can, for example, control the shaft position of a servo motor, the speed of a DC motor, the effective brightness of a light, or the temperature of a heater.

Does MicroStamp11 include a real time clock function?

Not exactly. However, it does have a real-time interrupt (RTI), which can be used to accurately measure the passage of time. In fact, a "Turbo" MicroStamp11's RTI rate is exactly 3.3333 ms, making it easy to keep track of time accurately to 1/100th of a second (ie. Every 3 interrupts equals 10 ms). If you are not using the "Turbo" version, or if you need to keep track of absolute time, especially dates and time, it is best to interface a real time clock/calendar chip to MicroStamp11. Many are available from various manufacturers. Those with a serial interface are usually the best choice, as they can be implemented with fewer MicroStamp11 I/O pins.

Does MicroStamp11 have a built-in timer?

Yes, it has a real-time interrupt, that is programmable in increments of 4.096 ms, 8.192 ms, 16.384 ms, or 32.768 ms. This can be used for such applications as task-scheduling, switch debouncing, or elapsed time measurement. It can also be used to implement a time-of-day clock/calendar. (The increments for a "Turbo" MicroStamp11 are 3.333 ms, 6.667 ms, 13.33 ms, and 26.67 ms.)

Does MicroStamp11 support interrupts?

Yes, there are 20 different interrupt sources on MicroStamp11! These are all user-enabled, so that an interrupt can be made to occur on such events as

* an input pin logic-level changes
* the hardware timer overflows
* the hardware free-running counter matches a user-defined value
* a byte has been received via the serial port

There are also two interrupt input pins. IRQ and XIRQ, which can be used to signal MicroStamp11 when some external event has occurred. Additionally, any of the interrupts can be used to trigger a wake-up of MicroStamp11 from sleep mode. This is a very important advantage that MicroStamp11 has over similarly-priced stamp-sized modules.

. . . . . .

Questions regarding operation and use:

How do I program MicroStamp11?

You can write your BASIC, C, or assembler program using any text editor (eg. Notepad or Edit) on a standard IBM compatible PC. After you compile and assemble the code for your application, you simply plug MicroStamp11 into a Docking Modle, connect the Docking Module to the computer's serial port, provide power to the Docking Module, slide both switches to LOAD, and press the reset button.
To download your program into MicroStamp11's EEPROM, use the utility included with the Starter Package (MicroLoad for Windows)
. As soon as the program has been downloaded successfully, slide the switches back to RUN and PROT,
press the reset button, and MicroStamp11 begins executing its new program, starting at the first line of code.

How does my program get stored in MicroStamp11?

Your code gets stored within MicroStamp11's EEPROM memory. This memory is non-volatile, meaning it retains its program even without power. Every time MicroStamp11 receives power or is reset, it starts running code directly out of EEPROM, starting with the first executable line. Source code elements like comments and constant and variable names are not stored in MicroStamp11, thus you may feel free to use as many comments and descriptive symbol names in your code as you like without worrying about increasing your code size.

How do I erase MicroStamp11's program space?

An erase cycle is performed automatically on every byte that is to be changed, so erasing is not necessary. Addresses which do not need to be changed are not erased.

How do I re-program MicroStamp11?

Simply re-connect it to the computer via the Docking Module, and run MicroLoad. The standard configuration of the Starter Package includes a ribbon cable and solderless breadboard adapter, so you can leave MicroStamp11 in the Docking Module while you're developing your application. Just plug the adapter into your solderless breadboard in place of MicroStamp11. Wire up your application circuits, and attach the ribbon cable between the adapter and the Docking Module.
When you are finished, and no longer need the Docking Module, remove the ribbon cable from the adapter, and substitute MicroStamp11.
The right-angle dual-row receptacle (connector option FRA) provided on the module can be used with any commonly available 20-pin dual-row male connector having .025" square pins. In fact, a couple are included in the Starter Package, for your convenience.

How big a program can I store in MicroStamp11?

MicroStamp11 has 8K or 32K bytes of program storage (depending on which version you purchased). Most MCU instructions require from 1 to 3 bytes of storage space. This means you have enough space for about 11,000 instructions on a 32K version, or 2,730 instructions on an 8K version. If you're writing your code in C or BASIC, the number of lines of code you can store will depend on the number of instructions it takes the compiler to implement a line of code. This number will vary greatly, depending on the line of code being compiled.

Can I expand the program memory?

Not directly; however, you can add non-volatile memory (EEPROM or Flash) via the SPI port for archiving programs or subroutines, and then have your main program load them into RAM whenever you wish to execute them.

Can I expand the data memory?

MicroStamp11 is also available in a 64K version, which adds 32K bytes of data memory by means of a RAM chip stacked on top of the EEPROM (32K version only). This is ideal for larger applications which require more variables, greater stack space, and structures such as arrays. Whether you have a 64K or standard MicroStamp11, you can still interface serial EEPROMs, or other memory devices via
MicroStamp11's I/O pins (eg. SPI) to gain more data storage area. You will have to include the appropriate code within your program to
interface to the particular device you choose.

How difficult is it to program MicroStamp11?

When compared to other programmable microcontrollers, MicroStamp11 is perhaps the easiest to use because of its powerful hardware subsystems, the variety of languages supported, and its straightforward method of downloading and debugging. If you have some experience programming in BASIC, C or Pascal, you should find the learning process with MicroStamp11 to be a fairly simple one, once you get to know
the MCU's hardware features. If, however, you have never had any programming experience, you may have to spend some extra time studying
the examples and application notes provided before you feel comfortable with MicroStamp11.

Can I program MicroStamp11 in Visual BASIC, TurboPascal, or QBASIC?

No. These languages produce programs that are meant to be executed on a PC. MicroStamp11 is a microcontroller, not a miniature PC. It does not have a graphical user interface, a hard drive, or a lot of memory.

What is SBASIC?

SBASIC is a 68HC11-specific form of the BASIC programming language which many people are familiar with. It is a Freeware compiler, written by Karl Lunt, and is intended for educational and personal use. It is not permitted to be used for the development of commercial applications. SBASIC has special commands to efficiently control the features of the 68HC11 MCU.

Can I embed assembly language routines in my SBASIC or C code?

Yes, SBASIC and C are both compilers (not interpreters), and they allow assembly language routines to be included wherever desired. You run the compiler on the completed program to produce assembly language code. Then you assemble it to produce a downloadable file called an s-record.

How do I know how much space my program consumes?

A quick estimate can be made by dividing the .s19 filesize by two. But the best way is to look

at the listing file generated by the assembler to see what is the last address it put code in.

How do I debug my programs?

There are two types of errors arising in programs:
1. Syntax errors (missing or incorrect punctuation and misspelled keywords, for example)
2. Logical errors (faulty logic, where the resulting flow control is not what was desired, for example)

Syntax errors can be corrected by examining the offending line(s) and comparing them with the definitions provided in the compiler user manual.

Logical errors are sometimes more difficult to find but may be more easily discovered by careful placement of print commands. For example, you can print the current status of specific variables in your program to the serial port as it is executed. Then monitor the serial port using any terminal program.

Can I read out the program which is already stored in MicroStamp11?

Yes, using PCBug11, you can examine and modify the contents of MicroStamp11's EEPROM (as well as RAM and the MCU registers). You can also disassemble it, displaying the contents in assembly language. However, there is no way to "reassemble" it into the original C or BASIC program.

Can my program store sensor values to the on-board EEPROM (ie. datalogging)?

No, on-board EEPROM is reserved for your program and any constant data (eg. text messages, conversion factors, etc.). After programming, you switch the WRITE PROT switch on, preventing EEPROM from being written to. To log data values, (eg. sensor values), you should store them in serial EEPROM or serial Flash that you interface to the SPI port.