Fomu - ICE40 FPGA Development Board

PRODUCT ID: 4332

DESCRIPTION

Only 13mm long, Fomu really puts the *micro* in microprocessor. Fomu is a fully open-source, programmable FPGA device that sits inside a USB Type-A port. It has four buttons, an RGB LED, and an FPGA that is compatible with a fully open source chain and capable of running a RISC-V core. Fomu comes in a custom plastic enclosure that slots perfectly into a USB port.

Fomu:

- has Python
    - With 128 kilobytes of RAM and a large amount of storage, Fomu is powerful enough to run Python natively. And since it lives in your USB port, installation is super simple. FPGAs are complicated, but the latest Python tools make it easy to use Fomu without any specialized training.

- runs RISC-V
    - Underneath the Python interpreter lies a RISC-V softcore running on the FPGA fabric. RISC-V is an up-and-coming processor architecture that is poised to take over everything from deeply-embedded chips to high-performance computing. Fomu's RISC-V softcore is a great introduction to the processor architecture of the future.
- is an FPGA
    - An FPGA is a piece of reconfigurable silicon. The default Fomu firmware exposes a USB bootloader running a RISC-V softcore, but you can load whatever you want. Softcores are also available for LM32 and OpenRISC. You can practice adding instructions to the CPU, or add new blocks such as LED blink patterns or better captouch hardware blocks.
- is entirely open
    - Developing with Fomu is incredibly easy: just load code via USB and go. Whether you're writing RISC-V code, Python code, or HDL, it's all uploaded to Fomu in the same way. The ICE40UP5K FPGA is supported with a fully open toolchain, meaning you can start development without creating an account, signing an NDA, or downloading a multi-gigabyte installer.

An FPGA is an Array of Gates that is Field-Programmable. When you buy a chip such as a CPU, the logic cells are all fixed in place. A CPU can run any amount of code, but if you want to do anything exotic you need to create software and, depending on what you want to do, that software can be very slow.

For example, many embedded projects use WS2812 LEDs such as NeoPixels that require a specialized timing signal. A CPU can generate this signal in software, but it can't do anything in the background while talking to the light. If the string of LEDs is very long, then the CPU wastes a lot of time and power generating the signal.

With an FPGA, it becomes possible to create an "LED driver" that allows the CPU to keep running while a hardware component handles the timing. The CPU could do other work, or could put itself in a low power state.

In fact, the “CPU” in the FPGA is created from a hardware description language, meaning it can be modified or swapped out. If you wanted, you could create a brand-new CPU instruction. Do you want to have fast 64-bit multiplies? Or maybe you want a way to get random numbers easily? With Fomu and its FPGA, you have the source code to the CPU itself.

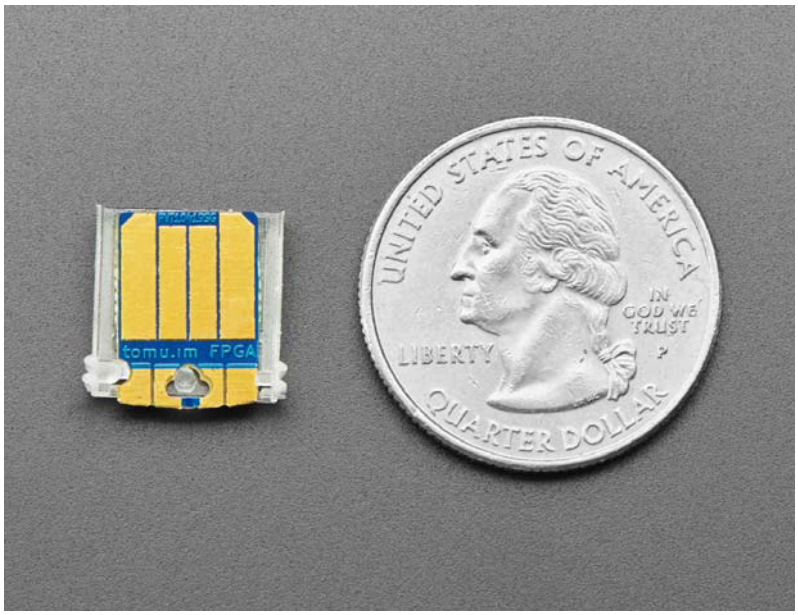
Comes with one fully assembled Fomu.

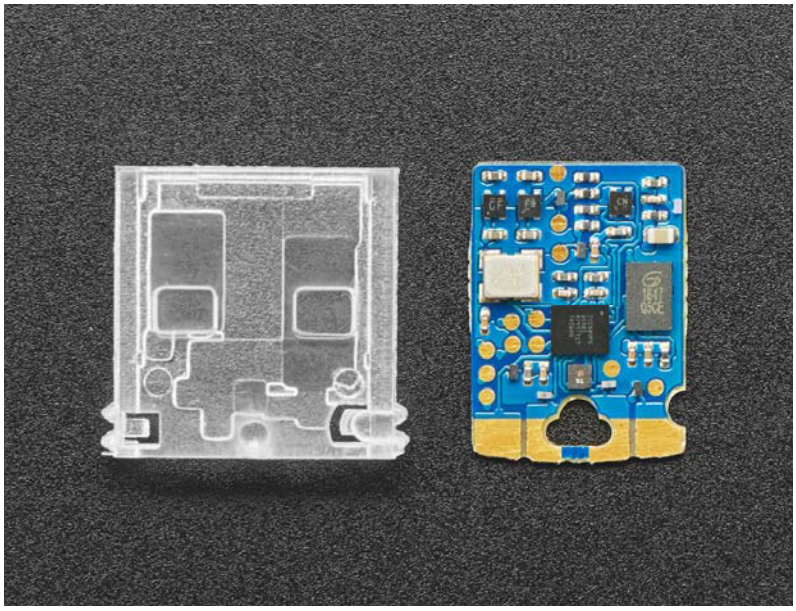
TECHNICAL DETAILS

Features & Specifications

- FPGA: Lattice ICE40UP5K (link to datasheet and reference manual directory)
- Speed: 48 MHz external oscillator
- RAM: 128 kB RAM for a soft CPU[1]
- Storage: 1 MB SPI flash[2]
- Connectivity: USB 2.0 FS (12 Mbps)
- Buttons: Four[3]
- LEDs: One RGB
- GitHub repository
- Fomu page

[1]: The FPGA has 1024 kilo-bits of memory available. A separate block of memory is used for things like the processor register file, in addition to temporary memory for things like USB buffers. The CPU can use 64 or 128 kilobytes of memory, depending on configuration. [2]: This is the minimum configuration amount — more may be available in the final version.
[3]: Fomu has four copper pads near the edge. Capacitive touch solutions have not yet been validated.





https://www.adafruit.com/product/4332/2-28-20