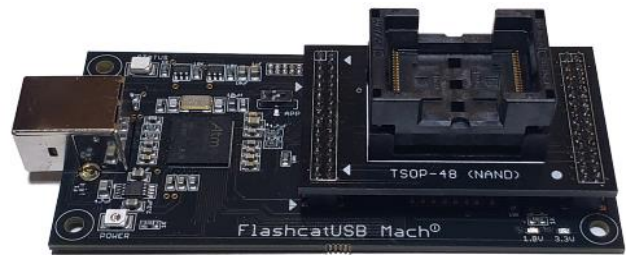
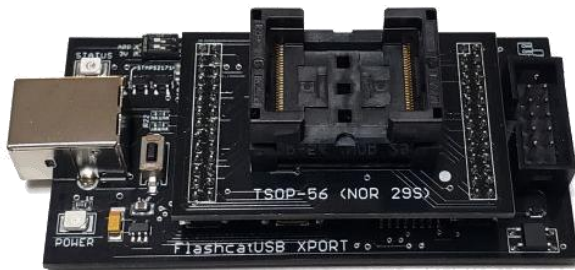
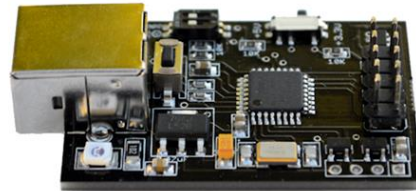


Embedded Computers

For professionals



FlashcatUSB

USER'S GUIDE

Classic, Professional, XPORT, and Mach¹

Website: www.embeddedcomputers.net/products/
Support email: contact@embeddedcomputers.net
Last updated: March, 2022



Table of Contents

WHAT'S NEW	3
INTRODUCTION	3
MEMORY HARDWARE SUPPORT	5
SOFTWARE REQUIREMENTS	6
LICENSING INFORMATION	6
DRIVER INSTALLATION	7
HARDWARE LAYOUT AND ORIENTATION	8
SOFTWARE OVERVIEW	11
BOOTLOADER MODE	15
MULTI-DEVICE PROGRAMMING.....	16
SPI MODE FOR FLASH PROGRAMMING.....	17
SPI FLASH WIRING DIAGRAMS.....	18
PROGRAMING MOTHERBOARDS USING SPI MODE.....	20
HOW TO PROGRAM I2C EEPROM.....	22
HOW TO PROGRAM MICROWIRE EEPROM	23
HOW TO PROGRAM ONE-WIRE EEPROM.....	24
JTAG MODE FOR ISP FLASH PROGRAMMING	25
SOFTWARE ECC FOR NAND MEMORY	26
USING BOUNDARY SCAN PROGRAMMER.....	28
USING THE COMMAND PROMPT / CONSOLE MODE.....	30
LIST OF SUPPORTED FLASH DEVICES	31

WHAT'S NEW

1. Parallel EEPROM memory support added (AT28C)
2. Console version updated
3. SPI NAND performance for Professional has been improved by 3.5 times.
4. NAND Wait feature has been added; you can select to use either hardware RB pins or software status-register.
5. Updated Flash memory database

INTRODUCTION

FlashcatUSB is a versatile multi-protocol Flash memory programming platform. This family of products includes: **Classic**, a low-cost 8-bit microcontroller based design, **Professional**, a higher performance 32-bit ARM based microcontroller with ISP logic, and **XPORT**, a programmer designed to interface with NOR/NAND memory in high-pin packages. **Mach¹** is the most advanced programmer with high-speed USB and a high-performance logic device to interface with the highest density memories available.

FlashcatUSB is a cost-effective hardware tool that is used both by industry professionals, semiconductor manufacturers, defense contractors, and hobbyists. By utilizing a USB microcontroller design, the hardware in conjunction with software and firmware can allow the device to configure its operation to meet the required protocols of a desired TAP (test-access point) or interface bus (such as a serial peripheral interface). Since the device uses a hardware-based USB interface, it can then perform its function at a much faster rate than traditional serial or parallel ports.

Classic hardware features:

ATMEGA32U2/U4 8-bit AVR microcontroller
Processor specifications: 16MHz clock, 8Mhz peripheral clock
USB PHY: Full-speed mode (12Mbps)
Memory: 1KB/2.5KB RAM, 32KB Flash (4KB reserved for bootloader)
Voltage output: 3.3v or 5.0v (current @ 1000ma)
Reverse voltage protection (STMP52171 MOSFET)
VCC output disable switch

Professional hardware features:

SAM3U 32-bit ARM microcontroller
Processor specifications: 96MHz, 48MHz peripheral clock
USB PHY: High-speed mode (480Mbps)
Memory: 36KB RAM, 128KB Flash (16KB reserved for bootloader)
Voltage output: 1.8v and 3.3v (current @ 500ma)
FPGA design with 3520 programmable logic elements
VCC output disable switch

XPORT hardware features:

ATMEL AT90USB646 8-bit AVR microcontroller
USB PHY: Full-speed mode (12Mbps)

Processor specifications: 16MHz clock, 8Mhz peripheral clock
Memory: 4KB RAM, 128KB Flash (8KB reserved for bootloader)
Voltage output: 3.3v or 5.0v (current @ 300ma)
+12V support for UV or OTP EPROMs
SPI and I2C protocols supported

Mach¹ hardware features:

ATMEL SAM3U 32-bit ARM microcontroller
Processor specifications: 96MHz, 48MHz peripheral clock
USB PHY: High-speed mode (480Mbps)
Memory: 52KB RAM, 256KB Flash (16KB reserved for bootloader)
Voltage output: 1.8v and 3.3v (current @ 500ma)
Integrated FPGA with 4000 logic elements

SPI Mode supported features:

Mode 0, 1, and 2 compatible
SPI-QUAD and SPI-DUAL IO supported
High density devices supported: 1 to 128 Mbit. (24-bit addressing)
Ultra-high-density devices supported: 256 Mbit to 2 Gbit (32-bit addressing)
Ability to program MCU's with on board Flash / NV memory
SPI-NAND devices are supported (up to 4Gbit)

I2C mode supported features:

Supports all I2C and TWI memory devices
Address selectable (A0, A1, A2)
Supports up to 400kHz data rates (1MHz for Pro)

JTAG mode (Classic/XPORT) supported features:

Programming of logic devices (i.e. CPLD) using SVF player

JTAG mode (Professional) supported features:

Clock speed: 40MHz, 20MHz and 10MHz selectable
Boundary Scan programming for parallel Flash devices.
ARM and MIPS processor designs supported

Parallel Flash mode supported features:

Extension Port: Parallel NOR devices (up to 512Mbit)
FlashcatUSB XPORT: Parallel NOR devices (up to 2Gbit)
NAND support: X8 IO, 3.3V, SDR
OTP EPROM (27 series) with +12V VPP support

Mach¹ mode supported features:

NAND support: X8/X16 IO, 1.8V/3.3V, SDR/DDR
HyperFlash S26KL/S26KS, 1.8V/3.3V

Software ECC algorithms supported:

Hamming (1-bit correction)
Reed Solomon (1-bit to 14-bit correction)
Binary BHC (1-bit to 14-bit correction)
Note: commercial license required

MEMORY HARDWARE SUPPORT

Below is a list of all the types of memory devices supported and which programmers can be used.

	Classic	XPORT	Professional	Mach ¹
SPI	✓	✓	✓	✓
SPI NAND	✓ (Note 1)	✓ (Note 1)	✓	✓
SPI QUAD IO	✓ (Note 5)	✓ (Note 5)	✓ (Note 4)	✓ (Note 4)
I2C	✓	✓	✓	
Microwire	✓	✓	✓	
1-Wire	✓		(Note 3)	
X8 NAND		✓ (Note 2)		✓
X16 NAND				✓
HyperFlash				✓
Parallel Flash		✓		✓
FWH Flash		✓		
JTAG (SVF)	✓	✓	✓	
JTAG (Advanced)			✓	
Voltage: 5V	✓	✓		
Voltage: 3.3V	✓	✓	✓	✓
Voltage: 1.8V			✓	✓

Notes:

- 1: Speed is limited to software implementation
- 2: SLC devices up to 4Gbit supported
- 3: Planned for future update
- 4: Supports single, dual, and quad operations
- 5: Speed limited to 200KB/s read and 100KB/s write

SOFTWARE REQUIREMENTS

A computer with at least a 1 GHz processor and 256 MB of free memory available, and a USB 2.0 port with full 5V (500mA) power. Operating systems supported: Windows 7, 8, 8.1,10, and 11. Supports both 32-bit and 64-bit versions.

Software is compiled using Microsoft .NET Framework 4.8.

LICENSING INFORMATION

The software is provided in both compiled binaries for Windows platform and as source code that is compilable with the free version of Visual Studio. The text file 'LICENSE_README.txt' included in this software, details the entire license and legal information.

This software is available for all lawful proposes, without the use of a license, for personal use. This also includes hobbyists. Colleges and universities need to obtain an academia license (contact us to register).

Commercial use is prohibited without a license after 30-days of use (evaluation period). There are limited exclusions, see the License file for exact terms and conditions.

Licenses are digitally issued and sold on an annual basis, you can purchase online at: <https://www.embeddedcomputers.net/software/> or via a purchase order, email to contact@embeddedcomputers.net.

Some features of the software will require a license to use. This includes:

- Multi-device programming support
- Boundary-scan programming mode (JTAG)
- Command-line (console mode) support
- ECC support for NAND devices

Perpetually licenses are available upon request.

In addition, the license system is applied enterprise wide. Meaning, once a company is a licensor, their license can be used on as many employee/engineer computers within the company. If you are an engineer who uses this software daily, please consider purchasing it legally, it really does help us out with further development and improvements.

NOTE: the software will always function, regardless of license status.

DRIVER INSTALLATION

When you connect FlashcatUSB to your PC for the first time, you should notice a "Found new hardware" notification pop-up, depending on what operating system you are using, you may have to manually install the driver. To do this:

1. Open the run command by pressing and holding the Windows key, then press the R key ("Run").
2. Type devmgmt.msc
3. The Device Manager window will now appear. Find the newly connected device, usually in the "Other devices" category.
4. Right-click the Unknown device and select "Update driver software".
5. You will be presented with two choices, "Search automatically..." and "Browse me computer...". Select the later to browse for the driver.
6. Select the folder "Drivers". You can use this driver for all FlashcatUSB firmware, including SPI, JTAG, NAND, and DFU.

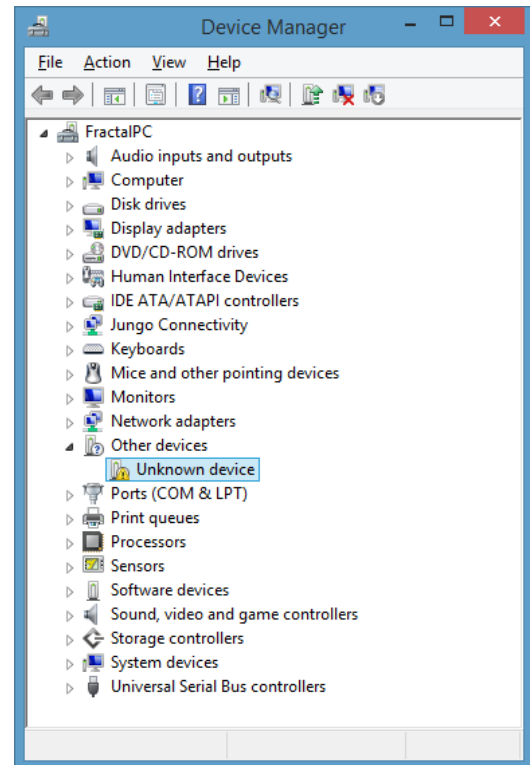


Figure 1 – Device Manager

You may need to do this for each device mode. You should use the same driver file for every FlashcatUSB device you connect to your PC.

HARDWARE LAYOUT AND ORIENTATION

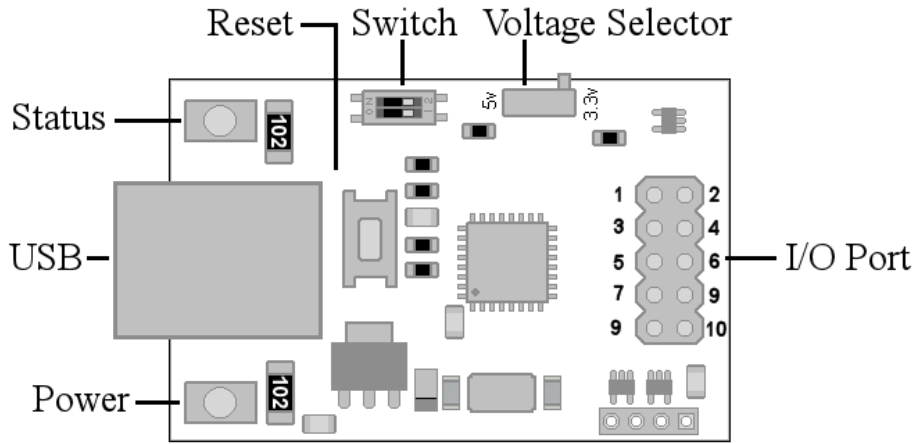


Figure 2 - Classic (PCB 2.2 / 2.3)

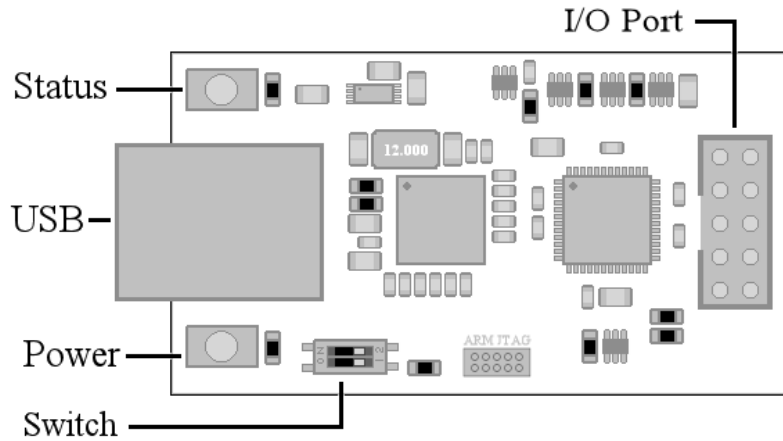


Figure 3 - Professional PCB 4.0 Shown, PCB 5.0 is similar.

Note: All FlashcatUSB products use the same I/O port pinout and numbering scheme for serial devices.

Definitions of diagram labels:

Status – This LED (blue) blinks when the device is reading or writing to memory. This LED will be solid to indicate that the board is successfully connected to the FlashcatUSB interfacing software.

Power – This LED (red) turns solid when the device is powered on and the on-board firmware application has started.

USB – This is the USB Type-B that needs to be connected to a USB cable that is then connected to your PC.

Reset – This button will reset the firmware application. If the switch #2 has been set to off, the board will instead boot into DFU mode.

Switch – This part has two switches, the first enables/disables the VCC out pin. The second pin is used to enable bootloader mode, if this pin is off, pressing the RESET button will cause the device to reset and start in bootloader mode. This mode is used to program the main application firmware.

*On newer revision boards, a label of **APP** and (**V**) are used indicate which pins are used for the application/bootloader mode and which pin is used to disable the VCC output pin.*

Voltage Selector – This switch controls the voltage output. If the switch is at the left position, the board will operate and output at 5v. If the switch is at the right position, the board will operate and output 3.3v.

I/O Port – This is the 10-pin port that is connected to your target device or Flash memory. The top left pin is a dedicated ground connection, the top right pin is the dedicated voltage supply.

All sockets and adapters connect the same way, with the text on the adapter reading the same direction as the text on the programmer.

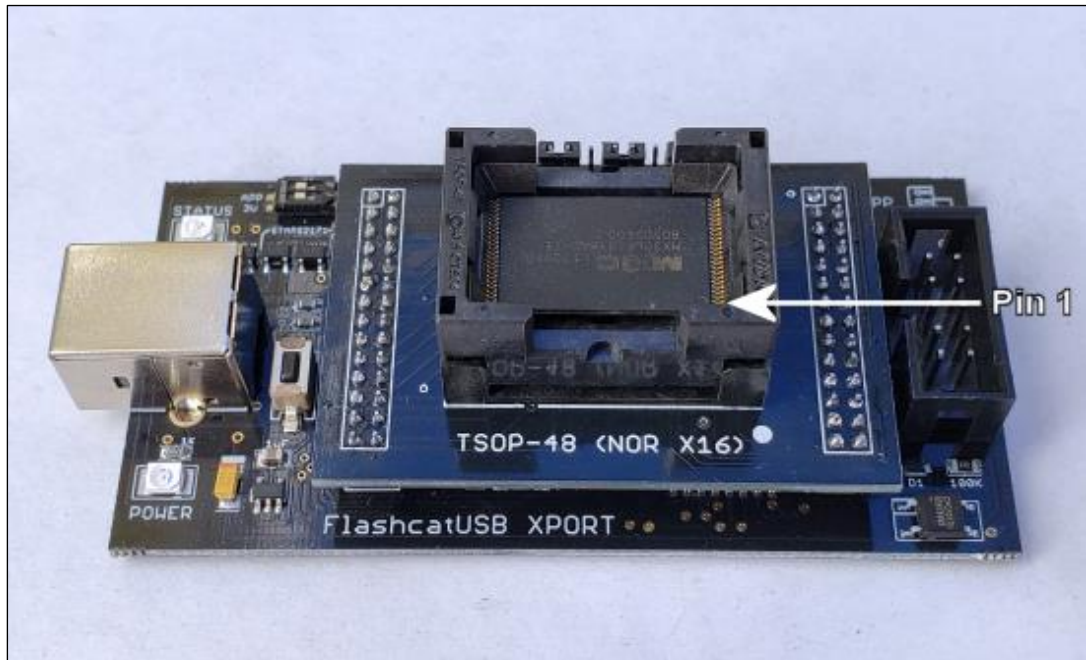


Figure 4 – XPORT PCB 2.0 with proper adapter and chip orientation.

Most sockets will have a “white-dot” placed on the socket adapter. This signifies how the Flash chip seated properly. PIN 1 of the Flash device (most chips will have an indented circle) needs to be placed closest to the “white-dot”. Notice that often (as with the TSOP-48 devices), the chip will appear to be upside down.

SOFTWARE OVERVIEW

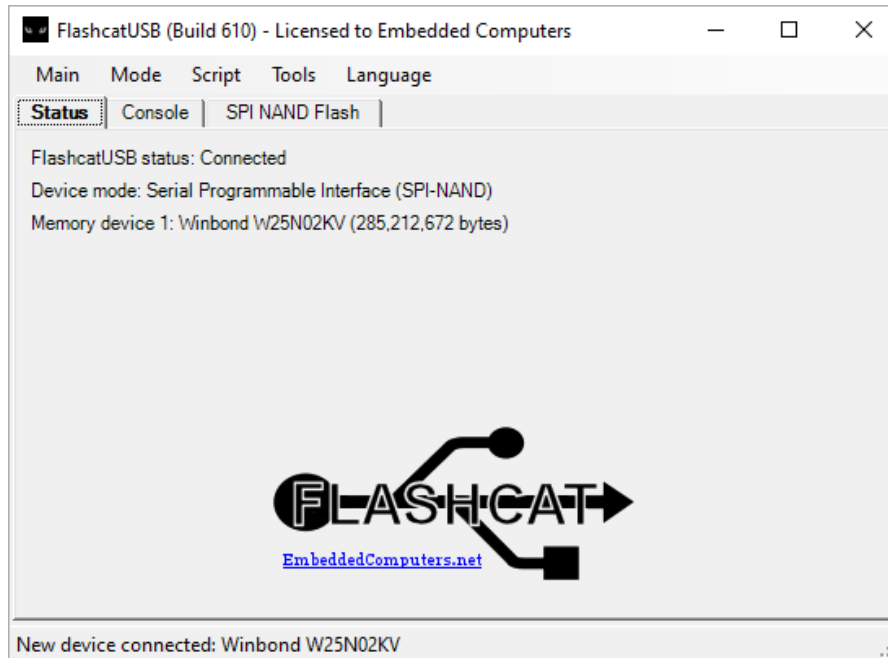


Figure 5 – Graphical User Interface of FlashcatUSB

The open-source software will run on Microsoft Windows operating system Windows 7, 8 and 10. Older versions of Windows, such as XP, are no longer supported.

If you already connect FlashcatUSB to your PC, when you run the software it will automatically attempt to connect to any attached Flash memory (using the operation mode last used). Otherwise, connect your device, set the operation mode and then click the Detect Device menu option.

Main Menu

Detect Device – will initialize the software and attempt to connect to a Flash using the current settings and operation mode.

Repeat write operation – will automatically perform the last Flash write operation. This feature is useful for programming multiple Flash devices using the same data.

Refresh flash device – will cause the current Flash tab and hex editor to refresh its data screen (performs a small read operation from the Flash device).

Exit – will close the software.

Settings Menu

Protocol Settings – this will bring up the window to set protocol specific settings, such as SPI clock frequency, i2c address, and NAND memory settings / bad block manager / layout.

Verify Programming – this setting will make all write operations verify the data written after each block/sector write by reading back the data and comparing it. If the comparison fails, the data is then re-written up to 3 times.

Bit Swapping – is a tool that will automatically swap the bits from data operations. The software supports swapping every 4-bits, 8-bits, and 16 bits. For example, 0x1F with 4-bit swapping would appear as 0xF1.

Endian Mode – is also a bit modification tool that will change the base endian (byte order) of the data. That is the MSB and LSB. For example, 0x31 with Little Endian mode will output 0x8C. This tool can be used if the Flash memory is used as a microprocessor boot device that executes code in a specific byte endian.

Script Menu

Current script – in JTAG mode, this will display all of the compatible scripts that are specified for your processor. You can also use it to change/select other compatible scripts.

Load script – this allows you to manually load a specified script.

Unload script – this will unload any current or loaded script.

Tools Menu

Erase chip – will perform a full chip erase of the selected Flash device. Please note, not all Flash devices support full chip erase, in which case, the software will erase all sectors.

Create image – this option will do two full chip reads, compare the data to ensure a good read, and create a single zip file with entire data compressed.

Write image – this tool will open an image created using the Create image tool and write the data into a new Flash device.

NAND memory map – will launch the NAND memory map tool, which will show a graphical grid of all the NAND blocks. You can also use this tool to perform a full block erase and random byte read to verify if the block is still good.

Vendor Features – some devices may have vendor specific menus. This menu item will bring up that menu to access / change these settings. Some examples include SPI devices that have non-vol registers for changing SPI to QUAD mode. In single-wire mode, this feature will allow you to read/write to the security and OTP memory area.

The Flash Tab

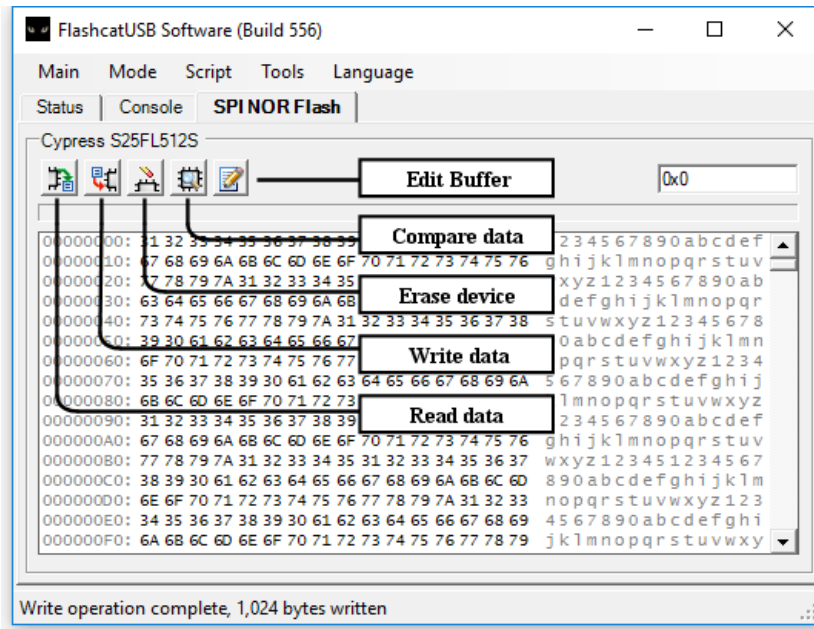


Figure 6 – Memory Device Interface

With a Flash device successfully detected, the software will create a Flash tab. This is the primary tool you will use to interact with a Flash device. This window will show you the name of the Flash device, as well as a hex data grid (and ASCII window) representation of the current data on the device. You can scroll down to see the data update in real-time. (Tip: pressing F5 will refresh the screen)

There are five buttons: read data, write data, erase all, compare, and edit mode. When you click the read data button, a window will appear to allow you to specify a base address and length. By default, the base address will be zero and the length will be the size of the entire Flash, so that if you do not change anything, the default operation will be to read the entire chip starting from the first byte. After clicking OK, a prompt will ask you where to save the data and it will allow you to save in three file formats: binary (*.bin), Intel HEX (*.hex) and Motorola S-Records (*.srec). This can be any location on your computer.

If you click the write data button, a file open prompt will appear asking for a file. By default it will filter for only binary files (*.bin), but you can use the drop down menu to also select Intel HEX files (*.hex) or Motorola S-Records (*.srec) which will automatically convert those file formats into binary data for writing. Upon selecting a file, you will now be able to set the base address and length of data to write, the default choice is the beginning of Flash and the length of the file or the Flash device, whichever is shorter.

The compare button will allow you to read a file from disk and compare it to the contents on the Flash. This will result in a window telling you how many bytes matched and a percentage.

The Edit Mode will allow you to edit the data buffer and write it to the Flash, while automatically filling the buffer with existing data. To use this feature, click the Edit Mode button and then move your cursor to any area in the hex editor or ASCII area and click.

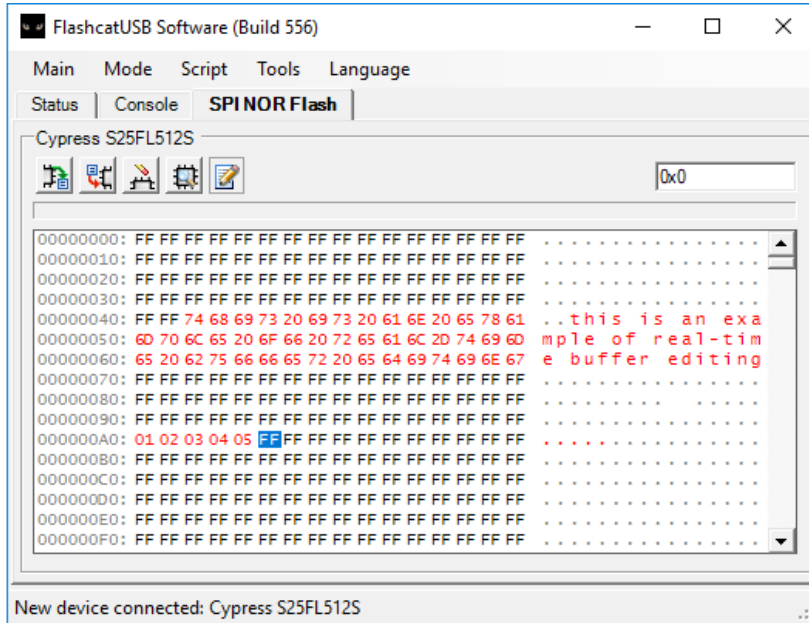


Figure 7 – Buffer Edit Mode Feature

If you edit the hex area, you can automatically change the next byte by pressing the Enter key.

In the ASCII area, you can type a single or multiple characters that will automatically be converted to hex using the standard ASCII to hexadecimal conversion table.

All changed data will be represented with red characters.

Finally, when you are satisfied with the edits, click the Edit Mode button once again which will prompt you if you want to write changes. Select Yes and the software will automatically update all sectors with changed data.

BOOTLOADER MODE

The FlashcatUSB product line is generally shipped with the newest firmware installed. However, sometimes stock from various vendors may contain out of data firmware. Since April of 2021, all FlashcatUSB products will update their firmware to the current version required by the software. But if you obtain older hardware, you may need to put the device into bootloader mode.

When the device is in bootloader mode, you may have to re-install the USB driver. FlashcatUSB Classic and XPORT ship with the standard ATMEL DFU loader, while Professional and MACH1 ship with our proprietary bootloader.



Figure 8 – Switch settings for Classic and Professional

To enable bootloader mode:

- 1) Unplug the FlashcatUSB device
- 2) Set the #2 switch to OFF position as shown in figure 8. (Note 1)
- 3) Connect the FlashcatUSB device back to your PC.
- 4) Classic and XPORT will need the button pressed.
- 5) Start the software to automatically begin the firmware update process.
- 6) After update, unplug the FlashcatUSB device and set the #2 switch back to ON.

Note 1: FlashcatUSB MACH1 only has a single switch which is used for bootloader mode.

MULTI-DEVICE PROGRAMMING

NOTE: This feature is only available to commercial license users.

Up to five FlashcatUSB devices can be connected to a single PC to allow for multi-device programming simultaneously (also known as gang programming).

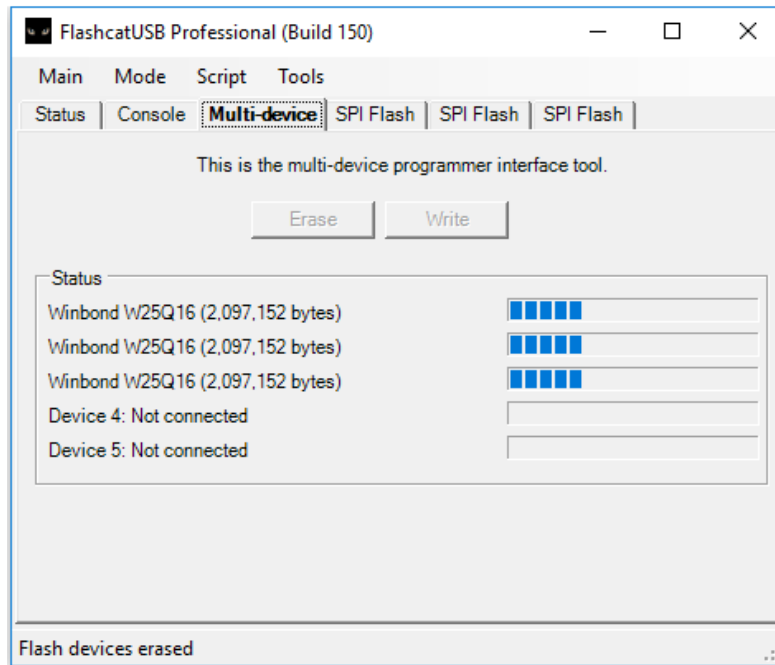


Figure 9 – Multi-device programming mode

When two or more FlashcatUSB are connected, the software will display the “Multi-device” tab as shown above. From this tab, you can erase or write to all detected memory devices at once. This is ideal for production environments that need to program hundreds, or even thousands of memory devices daily.

When using this mode, its best to connect the FlashcatUSB to native USB ports on your PC. This is because each device uses the maximum allowed current available in the USB specification (5v @ 500ma). So if you try and use a USB hub, it may not be able to provide enough current to power multiple devices, although it is safe to try and results may vary depending on the type of hub used.

Mountable programming fixtures are available, please contact us to receive a quote.

This feature is supported by all FlashcatUSB products. However, only multiple devices of the same model can be used (you cannot mix Classic/Pro for example).

SPI MODE FOR FLASH PROGRAMMING

With the software mode set to SPI NOR or SPI NAND you can use the device as a high-speed programmer for virtually every SPI and SQI compatible Flash memory device.

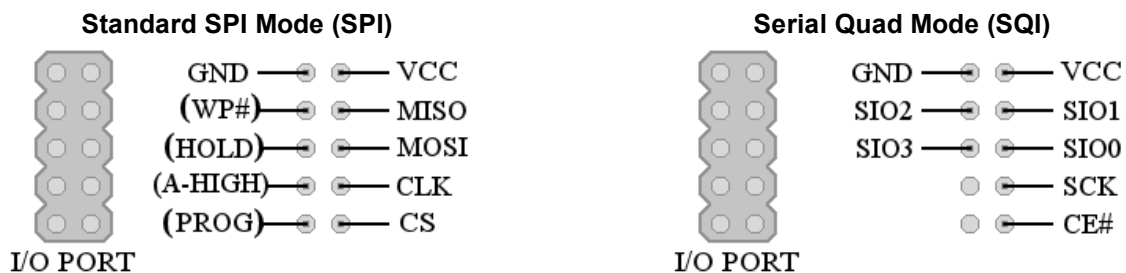


Figure 10 – Pinout for SPI and QUAD-SPI modes

GND – Ground connections [Pin 1]

VCC – Voltage out (can be disabled using SWITCH #1) [Pin 2]

MISO – Serial In (receives data from the SPI Flash) [Pin 4]

MOSI – Serial Out (sends data to the SPI Flash) [Pin 6]

CLK – Serial Clock [Pin 8]

CS – Chip-select, also called Chip-Enable [Pin 10]

WP# – Write-protect, some devices require a high-level to enable erasing [Pin 3]

HOLD# – Used to put a device into sleep-mode [Pin 5]

A-HIGH – Active high-level. Use this as an additional pin to set a logic input to high [Pin 7]

PROG – Programmable user pin. You can use the console command: SPI.PROG(1) to set the pin to high-level or SPI.PROG(0) to set it to low-level [Pin 9]

Note: For Nordic nRF24 products, connect the RST pin to (PROG), and the PROG pin on the Nordic device to the A-HIGH pin. When used in this configuration, the PROG pin will automatically pulse (resetting the device) and allowing it to enable SPI Slave mode.

Serial Quad Mode (SQI) supports both dual and quad operation mode.

SPI FLASH WIRING DIAGRAMS

The following diagrams illustrate how to manually connect the 10-pin IDC header on Classic, Professional and XPORT to various SPI memory packages. No additional resistors are needed.

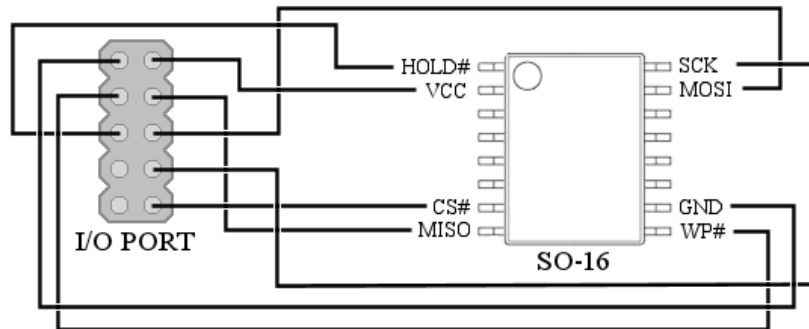


Figure 11 – SPI connection for SO-16 package memory

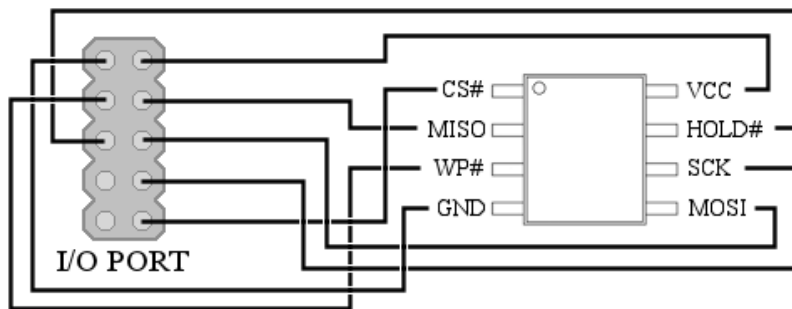


Figure 12 – SPI connection for SO-8 and DIP-8 package memory

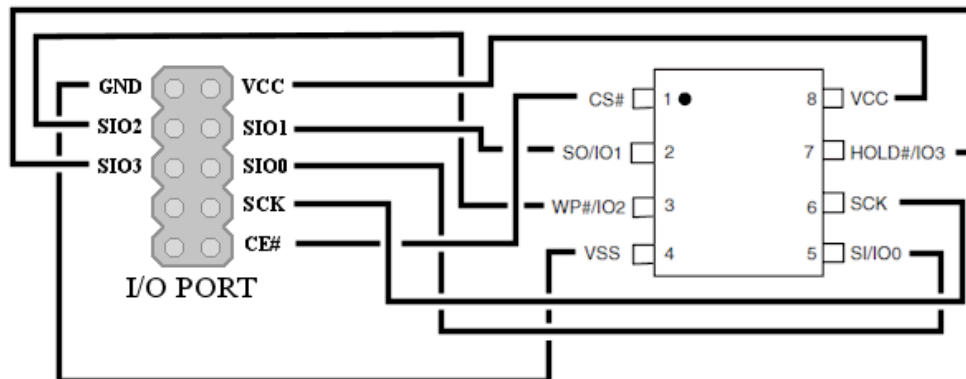


Figure 13 – QUAD SPI connection for SO-8 package memory

Various socket adapters are available for most packages. We can produce a socket for virtually any type of package, contact us for a quote.

SOIC-16 JEDEC (SPI)



TSSOP-8 (I2C)



SOIC-8 WIDE (SPI)



WSO8-8 8x6mm (SPI)



SOIC-8 NARROW (SPI)



WSO8 6x5mm (SPI)



DIP-8 (SPI and I2C)



USO8-8 2x3mm (SPI)



SOIC-8 NARROW (I2C)



BGA-24 (SPI)



Figure 14 – Various socket adapters are available

PROGRAMMING MOTHERBOARDS USING SPI MODE

Many motherboards, including circuit boards, contain small headers that are designed to program the soldered-on SPI Flash memory. Manufacturers such as MSI and ASUS will often have these ports located near the SPI Flash but will not be labeled or be mentioned in the motherboard's manual.

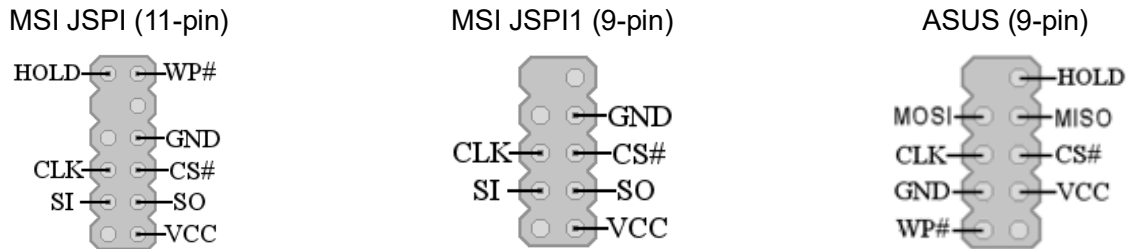


Figure 15 – Various pinout for motherboards

Connecting FlashcatUSB to this header, with the software in SPI NOR FLASH mode, should automatically detect the SPI Flash. Note: you should check the part number of the Flash and verify that it uses the correct voltage of the programmer. Some motherboards contain Flashes that are 1.8V and attempting to detect them with 3V could damage your motherboard. In this circumstance, you should use either FlashcatUSB Professional or use a level shifter circuit or adapter.

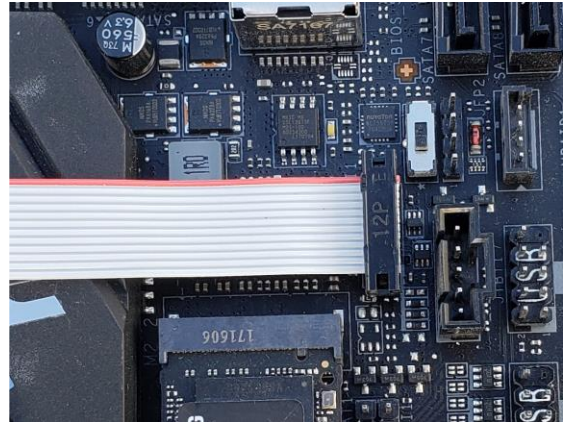
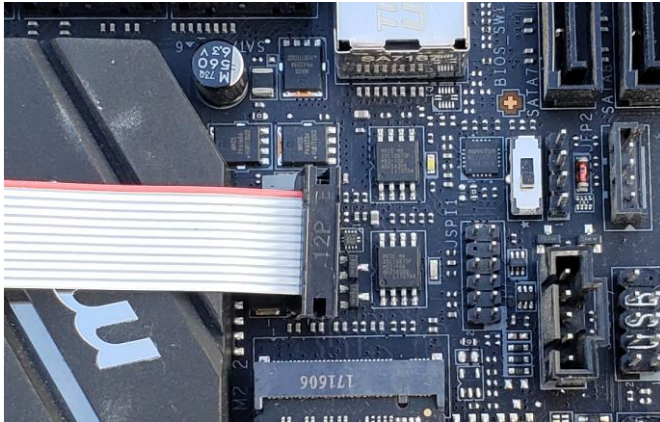
Tips for a successful connection

1. Make sure the SPI Clock speed is 8-10 MHz, too fast or too slow can lead to problems.
2. Remove the power supply and any attached peripherals to the motherboard. You can leave the CPU and memory connected, but we recommend everything else be removed.
3. Try using a premade adapter and if that does not work, try and use a SO-8 or SO-16 clip to connect directly to the SPI flash.

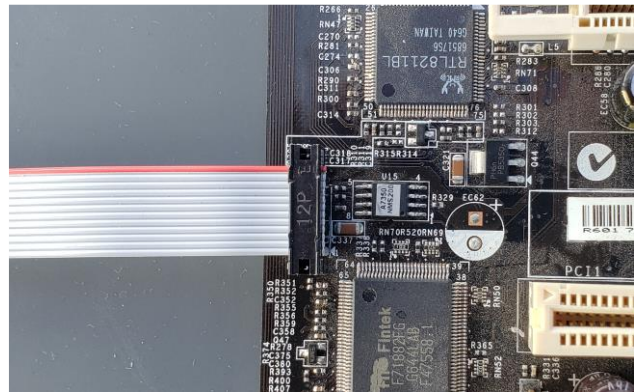
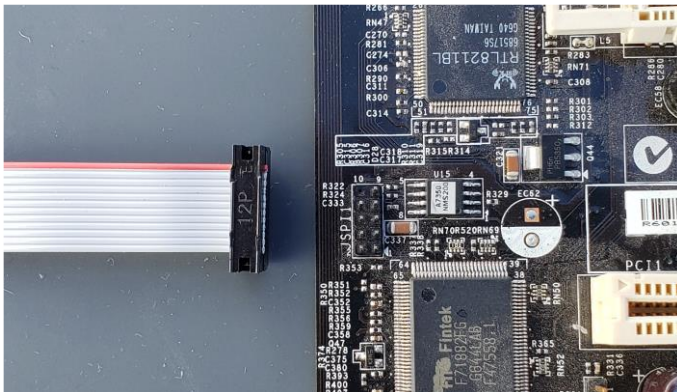
FlashcatUSB Classic can provide up to 1A of current at 3V and Professional can provide up to 500mA at 3V or 1.8V. Some motherboards may require more, in which case you need to provide that power and then disable the programmer's VCC output using the onboard switch.

Using an Embedded Computers MSI Adapter

The official MSI adapter that we produce can be used to program both 9-pin and 11-pin. Notice that you use the same 12-pin cable for both connections.



JSP11 (11-pin): Notice that the red pin on the cable is pointing up, and the missing pin is on the left.



JSP11 (9-pin): Notice that the top pin is red, and that it will overlap the motherboard, not connecting to anything. The missing pin is in the top left corner of the header.

HOW TO PROGRAM I2C EEPROM

I2C EEPROM (sometimes called TWI compatible) can be programmed by FlashcatUSB Classic, XPORT and Professional.

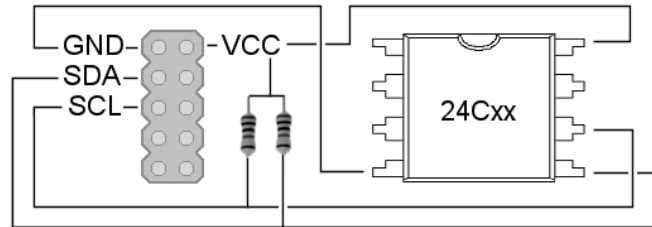


Figure 16 – I2C Wiring Diagram

The wiring connection only uses 4 pins from the FlashcatUSB I/O port. When connecting to an EEPROM device, the SDA and SCL lines must also have a resistor that connects to VCC. We recommend using 4.7K values, although any resistor from 4K to 10K should be suitable. The device address pins (A0, A1, A2) can be left unconnected or connected to ground. In-circuit devices may use the address pins, so you will need to check those boxes in the I2C / SWI settings (see Figure 17).

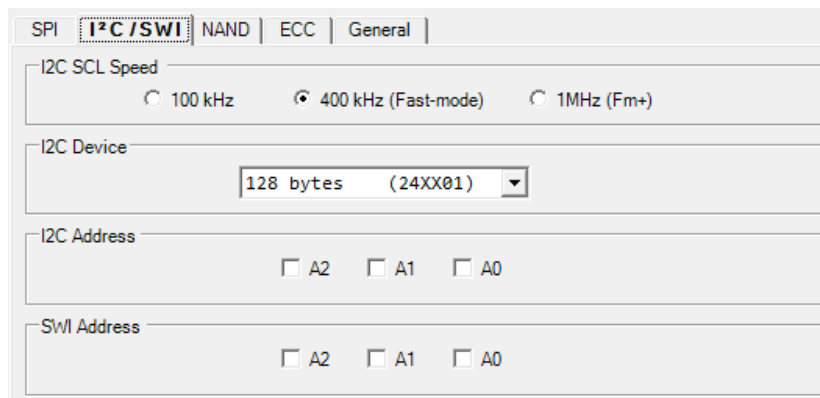


Figure 17 – I2C and SWI settings

The FlashcatUSB Software can not automatically detect I2C EEPROM devices. Therefore, once you have the device connected to the hardware, from the menu Mode→Protocol Settings, I2C / SWI tab (Figure 17), and select the device from the dropdown menu. Note: 400 kHz speed will meet the needs of nearly all I2C devices and does not need to be changed.

With the device successfully connected, run the software and from Setting menu, select 'I2C EEPROM Mode'. Then click 'Detect' from the 'Main' menu. A new tab will appear that will allow you to read and write to the EEPROM.

HOW TO PROGRAM MICROWIRE EEPROM

FlashcatUSB supports Microwire 3-wire EEPROM devices. This includes devices that have part numbers that match: 93xx46, 93xx56, 93xx66, 93xx76, 93xx86

Due to the limitations of the 93 series EEPROM devices, automatic detection is not available. So, you need to make sure the connection matches the diagram below. If you are using PCB 3.x, you will need to use PORT B.

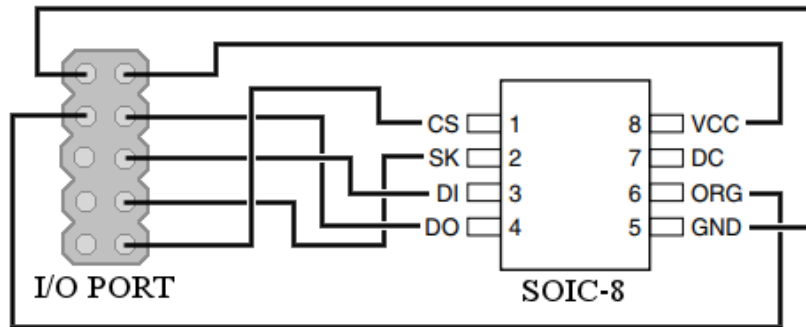


Figure 18 – Microwire Wiring Diagram

After connecting the device as shown above, open the software and go to the Mode-->Protocol Settings. Then select the MISC tab.

In this section, you need to select your device from the drop-down menu and select either X8 or X16 organization.

Close this menu and from the main Settings menu, select the “Microwire EEPROM” mode from the drop down. Finally, click Main-->Detect to have the software apply these settings and attempt to properly connect to this device. Note, if you see all 0s in the Flash tab, this may be an indication that you are not properly connected to the EEPROM device.

HOW TO PROGRAM ONE-WIRE EEPROM

FlashcatUSB Classic can be used to program single-wire EEPROMs. For example, the AT21CS01 and AT21CS11. First connect the device to the programmer like this:

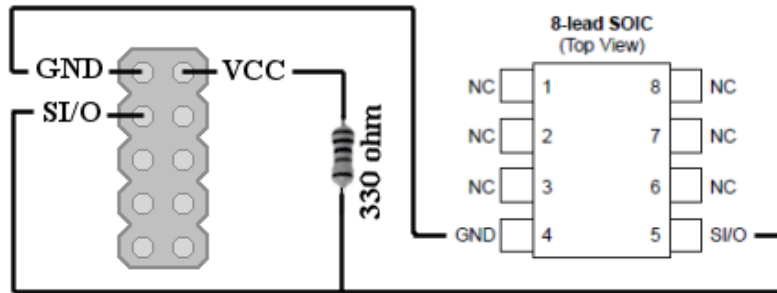


Figure 19 – 1-Wire EEPROM Wiring Diagram

You will need to connect a 330-ohm resistor to the SI/O line and the VCC. For VCC make sure the 3.3V switch is set.

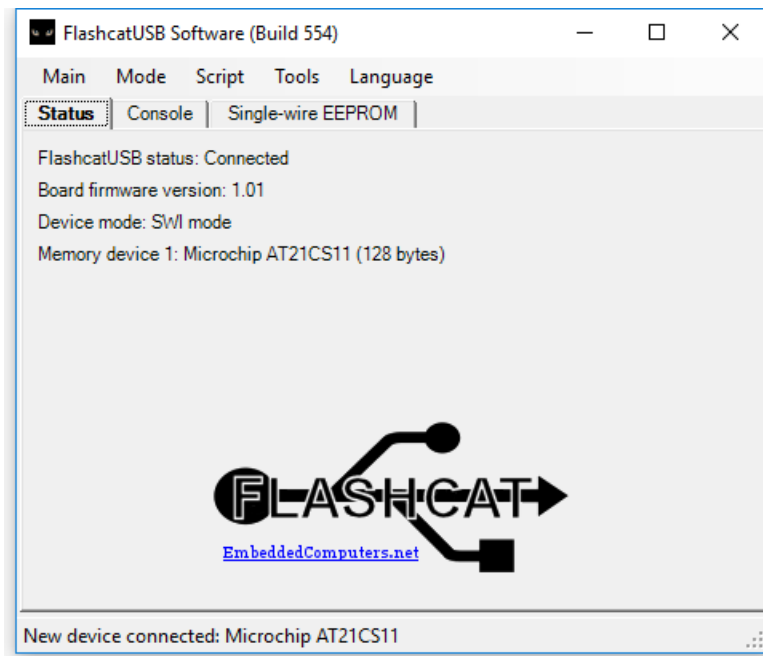


Figure 20 – Interface showing 1-wire EEPROM support

Finally, set the Mode to “1-Wire EEPROM” and the device will automatically be detected.

JTAG MODE FOR ISP FLASH PROGRAMMING

Many embedded devices with in-circuit non-volatile memory and programmable logic devices, such as CPLD, can be programmed using FlashcatUSB.

Complex Programmable Devices (CPLD) can be programmed using a standard SFV or XSVF format file. These devices are designed to be erased and programmed and will not typically be able to be “read” or “dumped”. Therefore, do not expect to use FlashcatUSB to clone an existing CPLD.

FlashcatUSB Classic is limited to SVF/XSVF programming only. FlashcatUSB Professional offers a wide range of tools, JTAG extensions and tools that expand programming to many kinds of hardware scenarios.

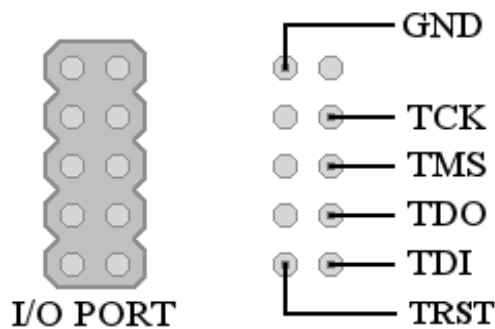


Figure 21 – Classic and Professional pinouts for JTAG

The image above shows you the pin outs of the 10-pin I/O port and how it should be connected to the test-access port (TAP) of your target device. FlashcatUSB will only act like a passive diagnostic tool for the target system, so the device will need to be powered on by itself. **Note: the TRST pin is optional.**

When a CPLD is connected to FlashcatUSB with JTAG mode enabled, upon detection, the software will automatically load the script 'SVF_Player.fcs' which will give you a new tab that you can use to program the CPLD using a SVF or XSVF file.

Support for various devices over JTAG is complex. If you need hardware support, please contact us, you may be required to provide sample equipment as well as a BSDL file.

SOFTWARE ECC FOR NAND MEMORY

NOTE: This feature is only available to commercial license users.

The FlashcatUSB software has support for ECC (Error Correction Code) for NAND devices. This feature is typically needed to successfully read or program NAND memory where bit errors are expected to occur.

The nature of NAND memory and the manufacturing process produces devices with varying levels of unreliability. NAND memory is very unreliable compared to NOR memory. In addition to factory bad blocks (areas of the device that are unusable), data stored on NAND memory will randomly change over time. To overcome this limitation, the spare area of the device is often used to store ECC data that can be used to automatically correct it.

However, different devices require different types of ECC implementations and there is no industry standard as to the types of ECC used or the location to store the ECC data. So, this means you will need to configure the software to be able to properly use this feature.

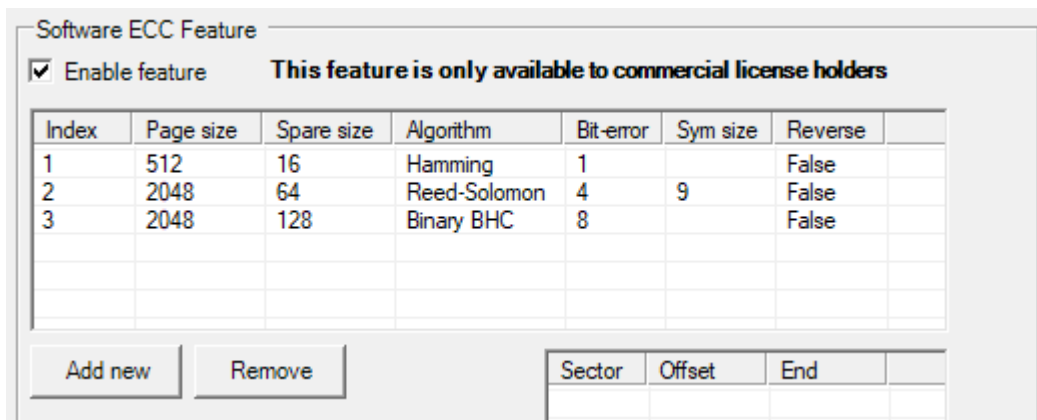


Figure 22 – ECC Settings

To enable this feature, open the Settings tab (Mode→Protocol Settings) and on the ECC you can check 'Enable feature'. By default, some common algorithms and settings are defined, but you may need to change them depending on the exact devices. The sector, offset, and end tables allow you to define where the ECC data is store/loaded.

For this feature to take effect, you need to initialize the device again using the Detect command.

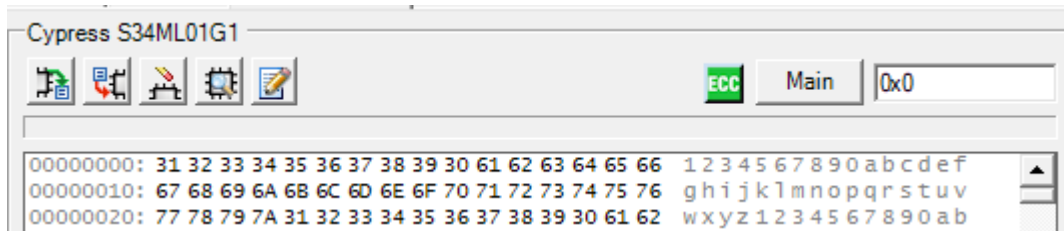


Figure 23 – Memory Interface with ECC enabled

Software ECC will auto-correct 'Main' data read from a device with the ECC data stored in the Spare area. When this operation is successful, a logo next to the area select button will be shown in green. If ECC data fails to load or the correction fails, the ECC logo will be a blue color. This icon will only be displayed when the memory area is set to Main.

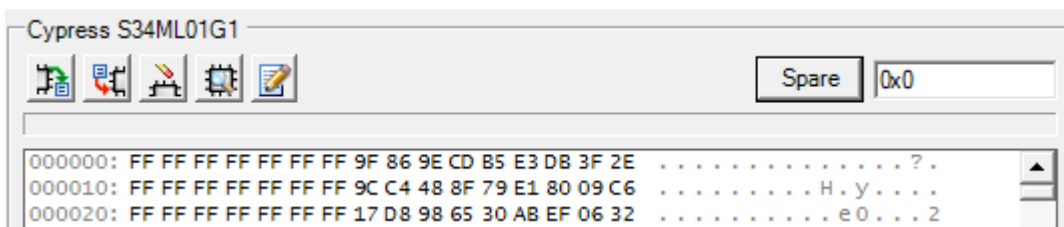


Figure 24 – Spare area shown where the ECC data is

The area select button allows you to change between Main, Spare and All. The Main area is where the stored application data should be, while the spare area will contain information such as bad block markers, wear-leveling data, and the ECC data. How and where exactly this data is arranged is very vendor specific and will most definitely vary. The 'All' will display every byte from the device.

Note 1: Serial NAND (SPI-NAND) or other 'managed NAND' devices, may have built-in ECC and therefore you should not use Software NAND.

Note 2: Software ECC uses a lot of resources and is very processor heavy. You should expect a huge performance downgrade when ECC is running. In the future, we plan to migrate our ECC libraries directly into the Mach¹ hardware to enable real-time ECC processing.

USING BOUNDARY SCAN PROGRAMMER

NOTE: This feature is only available to commercial license users.

FlashcatUSB Professional can be used to program a parallel NOR flash that is connected to a MCU, CPLD, or FPGA with a JTAG interface. This can be accomplished by using a little-known process called 'boundary scan programming', where the JTAG host controller puts the target device into an interconnect test (EXTEST) mode and continually updates all of the pins on the device to “simulate” the NOR programming interface. **Note: this feature is not available on Classic or XPORT.**

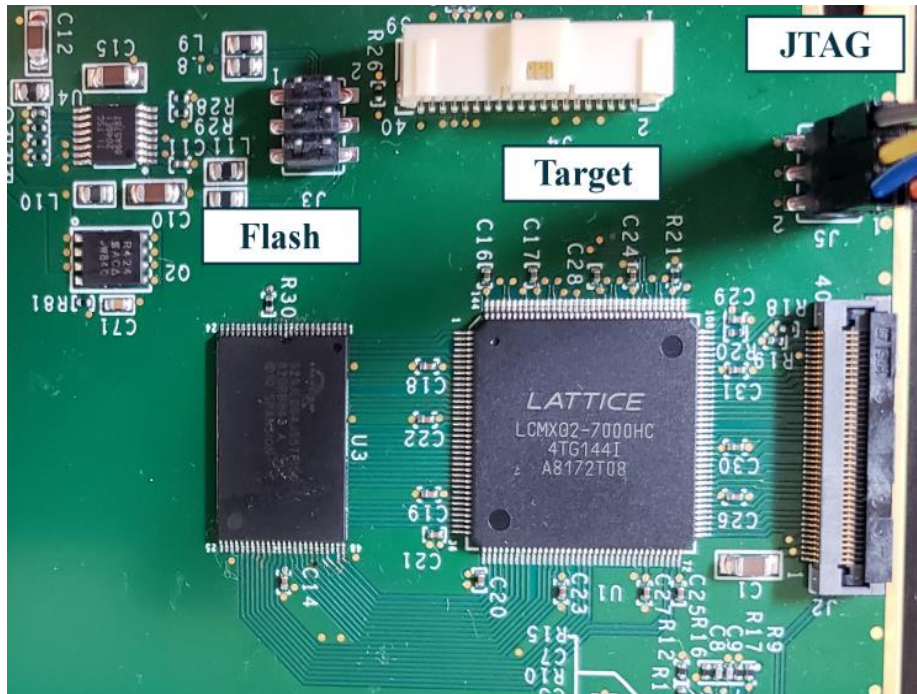


Figure 25 – A typical embedded platform with memory and JTAG support

The benefit of using this feature is that the external hardware does not need any special configuration or specific JTAG registers in order to program the memory and is considered the most versatile and universal method. However, there are drawbacks. First, data transfer speeds are going to be slow. For example, with the JTAG clock set to 20MHz you should expect a read speed of 5KB/s and a write speed of 2.5KB/s.

To use this feature, first you need to obtain the 'Target' BSDL file. This is a plain-text file that describes all the physical pins for a particular IC and all the internal pins and registers.

In the BSDL file, look for the line “attribute BOUNDARY_LENGTH” and at the end of this it will say “entity is <NUMBER>”; This number is the number of bits that the boundary scan register is.

Next, the file will then list all of the bits in the boundary scan register, starting with the bit index and then several parameters. The second parameter will be the pin description. Using this, you need to make a map of all the pins that connect to the Flash.

A typical NOR flash will have address pins, data pins, and control pins. You need to correctly create a list of all the pin indexes that are associated with the NOR flash pins.

For example, a 2mbit bootrom will have 20 address pins (labeled AD0 to AD19), 16 data pins (labeled DQ0 to DQ16) and 6 control pins labeled CE# (chip-enable), OE# (output-enable), WE# (write-enable), BYTE# (Byte/Word select), WP# (write-protect), and RESET#. RB# is also listed but you can ignore that as its not used. Some PCB designs may hardwire CE# to GND and RESET# and WP# to VCC, in which case you can ignore those too. But if they are wired to the Target IC, you need to include them. Note: datasheets will often vary on the pin terminology, some might use G instead of OE, A0 instead of AD, D0 instead of DQ0, etc.

In order to use this feature, you have to create a script file that defines the pin indexes (input/output/bi-dir) and assigns those pins to the NOR interface (“DQ0”, “DQ1”, etc.). Please see the **Boundary Scan Programmer commands** in the Script Engine PDF for all of the instructions needed. In addition, part of the commercial license program includes engineering support, so please contact us if you need help writing a compatible script for your target board.

When the script file is correct, when you run it, the software will give you an easy to use interface to access your in-circuit memory device:

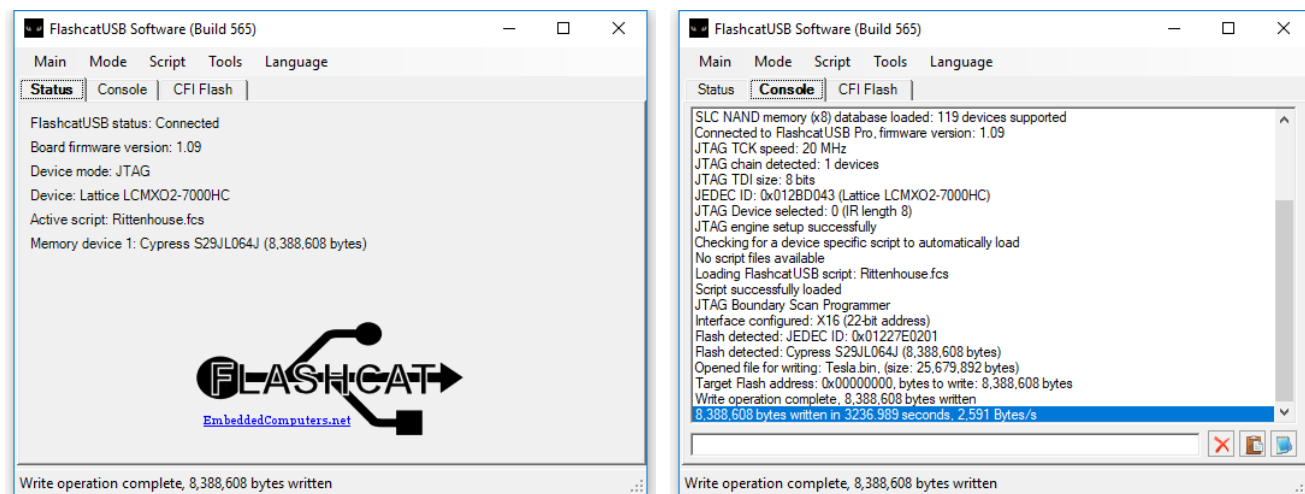
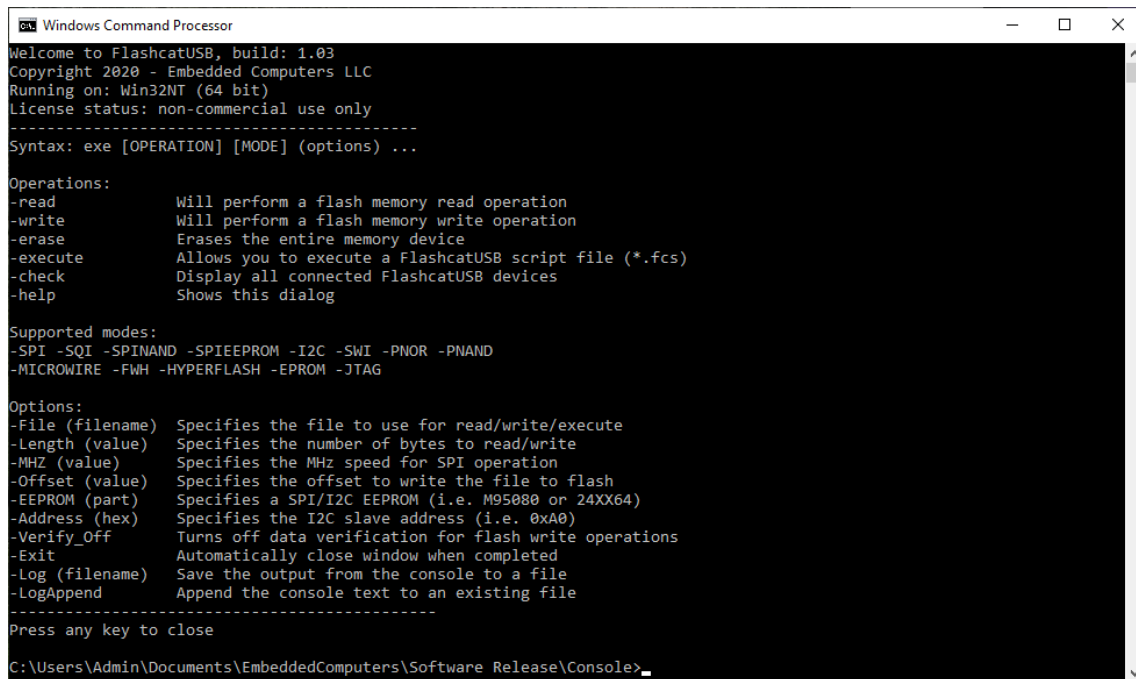


Figure 26 – Using the boundary programmer feature

USING THE COMMAND PROMPT / CONSOLE MODE

FlashcatUSB devices can also be controlled from a cross-platform console program. This software is written in .NET 5.0 Framework and is compatible with Windows, Linux and MacOS. Note: official software support is only available for Windows.

Compiled binaries for Windows are included in the 'Console' folder from the Windows software download. Other platforms will need to be compiled from the source code and configured to use LibUSB.



```
Windows Command Processor
Welcome to FlashcatUSB, build: 1.03
Copyright 2020 - Embedded Computers LLC
Running on: Win32NT (64 bit)
License status: non-commercial use only
-----
Syntax: exe [OPERATION] [MODE] (options) ...

Operations:
-read          Will perform a flash memory read operation
-write        Will perform a flash memory write operation
-erase        Erases the entire memory device
-execute      Allows you to execute a FlashcatUSB script file (*.fcs)
-check        Display all connected FlashcatUSB devices
-help         Shows this dialog

Supported modes:
-SPI -SQU -SPINAND -SPIEEPROM -I2C -SWI -PNOR -PNAND
-MICROWIRE -FWH -HYPERFLASH -EPROM -JTAG

Options:
-File (filename) Specifies the file to use for read/write/execute
-Length (value)  Specifies the number of bytes to read/write
-MHZ (value)     Specifies the MHz speed for SPI operation
-Offset (value)  Specifies the offset to write the file to flash
-EEPROM (part)   Specifies a SPI/I2C EEPROM (i.e. M95080 or 24XX64)
-Address (hex)   Specifies the I2C slave address (i.e. 0xA0)
-Verify_Off     Turns off data verification for flash write operations
-Exit           Automatically close window when completed
-Log (filename) Save the output from the console to a file
-LogAppend      Append the console text to an existing file
-----
Press any key to close
C:\Users\Admin\Documents\EmbeddedComputers\Software Release\Console>
```

To get started, from your computer's terminal, run the `fcusb_console` program. This will display the above prompt, which has the basic commands/operations and syntax to using the program.

To read an entire memory device and save it to disk:

```
fcusb_console -READ -PNAND -File nand_dump.bin
```

The first parameter `-READ` indicates that you want to read data. `-PNAND` is how you define which mode, in this case, Parallel NAND. `-FILE` tells the program where to write the data.

LIST OF SUPPORTED FLASH DEVICES

Here is an automatically generated list of part numbers from the software. This list is only updated one or twice a year, so current versions of the software may include devices not listed here. Please note that this list contains devices that operate with various voltage levels, so check the datasheet to make sure the programmer you have is able to operate at the correct voltage or consider using a level shifter.

Serial Peripheral Interface Flash Devices (SPI):

AT45DB641E (64Mbit)	W25Q512 (256Mbit)	PM25LV010 (1Mbit)
AT45DB642D (64Mbit)	W25Q256 (256Mbit)	PM25LV512 (512Kbit)
AT45DB321E (32Mbit)	W25Q128 (128Mbit)	PM25LD020 (2Mbit)
AT45DB321D (32Mbit)	W25Q64 (64Mbit)	PM25LD010 (1Mbit)
AT45DB161E (16Mbit)	W25Q32 (32Mbit)	PM25LD512 (512Kbit)
AT45DB161D (16Mbit)	W25Q16 (16Mbit)	A25LQ64 (64Mbit)
AT45DB081E (8Mbit)	W25Q16JV-DTR (16Mbit)	A25LQ32A (32Mbit)
AT45DB081D (8Mbit)	W25Q80 (8Mbit)	A25L032 (32Mbit)
AT45DB041E (4Mbit)	W25Q40 (4Mbit)	A25L016 (16Mbit)
AT45DB041D (4Mbit)	W25X64 (64Mbit)	A25LQ16 (16Mbit)
AT45DB021E (2Mbit)	W25X32 (32Mbit)	A25L080 (8Mbit)
AT45DB021D (2Mbit)	W25X16 (16Mbit)	A25L040 (4Mbit)
AT45DB011D (1Mbit)	W25X80 (8Mbit)	A25L020 (2Mbit)
AT25DF641 (64Mbit)	W25X40 (4Mbit)	A25L010 (1Mbit)
AT25DF321S (32Mbit)	W25X20 (2Mbit)	A25L512 (512Kbit)
AT25DF321 (32Mbit)	W25X10 (2Mbit)	A25LS512A (512Kbit)
AT25DF161 (16Mbit)	W25X05 (512Kbit)	FM25Q128 (32Mbit)
AT25DF081 (8Mbit)	W25M121AV (128Mbit/1Gbit)	FM25Q64A (32Mbit)
AT25DF041 (4Mbit)	W25Q256FW (256Mbit)	FM25Q32A (32Mbit)
AT25DF021 (2Mbit)	W25Q128FW (128Mbit)	FM25Q16A (16Mbit)
AT26DF321 (32Mbit)	W25Q64FW (64Mbit)	FM25Q08 (8Mbit)
AT26DF161 (16Mbit)	W25Q32FW (32Mbit)	FM25Q04 (4Mbit)
AT26DF161A (16Mbit)	W25Q16FW (16Mbit)	FM25Q02 (2Mbit)
AT26DF081A (8Mbit)	W25Q80EW (8Mbit)	FM25M04A (4Mbit)
AT25SF321 (32Mbit)	W25Q40EW (4Mbit)	FM25M08A (8Mbit)
AT25SF161 (16Mbit)	W25Q20EW (2Mbit)	FM25M16A (16Mbit)
AT25SF081 (8Mbit)	W25Q80BW (8Mbit)	FM25M32A (32Mbit)
AT25SF041 (4Mbit)	W25Q40BW (4Mbit)	FM25M64A (64Mbit)
AT25XV041 (4Mbit)	W25Q20BW (2Mbit)	FM25M4AA (4Mbit)
AT25XV021 (2Mbit)	MX66L1G45G (1Gbit)	DS25M4BA (4Mbit)
AT25SL128A (128Mbit)	MX25L51245G (512Mbit)	GD25Q256 (256Mbit)
AT25SL641 (64Mbit)	MX25L25655E (256Mbit)	GD25Q128 (128Mbit)
AT25SL321 (32Mbit)	MX25L256 (256Mbit)	GD25Q64 (64Mbit)
S70FL01GS (1Gbit)	MX25L12855E (128Mbit)	GD25Q32 (32Mbit)

S25FL512S (512Mbit)	MX25L128 (128Mbit)	GD25Q16 (16Mbit)
S25FL256S (256Mbit)	MX25L6455E (64Mbit)	GD25Q80 (8Mbit)
S25FL256S (256Mbit)	MX25L640 (64Mbit)	GD25Q40 (4Mbit)
FL127S/FL128S (128Mbit)	MX25L320 (32Mbit)	GD25Q20 (2Mbit)
S25FL128S (128Mbit)	MX25L3205D (32Mbit)	GD25Q10 (1Mbit)
S25FL127S (128Mbit)	MX25L323 (32Mbit)	GD25Q512 (512Kbit)
S25FS256S (256Mbit)	MX25L3255E (32Mbit)	GD25VQ16C (16Mbit)
S25FS128S (128Mbit)	MX25L1633E (16Mbit)	GD25VQ80C (8Mbit)
S25FS064S (64Mbit)	MX25L160 (16Mbit)	GD25VQ41B (4Mbit)
S25FS256S (256Mbit)	MX25L80 (8Mbit)	GD25VQ21B (2Mbit)
S25FS128S (128Mbit)	MX25L40 (4Mbit)	GD25LQ128 (128Mbit)
S25FS064S (64Mbit)	MX25L20 (2Mbit)	GD25LQ64 (64Mbit)
S25FL256L (256Mbit)	MX25L10 (1Mbit)	GD25LQ32 (32Mbit)
S25FL128L (128Mbit)	MX25L512 (512Kbit)	GD25LQ16 (16Mbit)
S25FL064L (64Mbit)	MX25L1021E (1Mbit)	GD25LQ80 (8Mbit)
S70FL256P (256Mbit)	MX25L5121E (512Kbit)	GD25LQ40 (4Mbit)
S25FL128P (128Mbit)	MX66L51235F (512Mbit)	GD25LQ20 (2Mbit)
S25FL128P (128Mbit)	MX25V8035 (8Mbit)	GD25LQ10 (1Mbit)
S25FL129P (128Mbit)	MX25V4035 (4Mbit)	IS25LP512 (512Mbit)
S25FL129P (128Mbit)	MX25V8035F (8Mbit)	IS25LP256 (256Mbit)
S25FL064 (64Mbit)	MX25R6435 (64Mbit)	IS25LP128 (128Mbit)
S25FL032 (32Mbit)	MX25R3235F (32Mbit)	IS25LP064 (64Mbit)
S25FL016A (16Mbit)	MX25R8035F (8Mbit)	IS25LP032 (32Mbit)
S25FL008A (8Mbit)	MX25L3235E (32Mbit)	IS25LP016 (16Mbit)
S25FL004A (4Mbit)	MX25L2005 (32Mbit)	IS25LP080 (8Mbit)
S25FL040A (4Mbit)	MX25L2006E (32Mbit)	IS25CD020 (2Mbit)
S25FL164K (64Mbit)	MX25L2026E (32Mbit)	IS25CD010 (1Mbit)
S25FL132K (32Mbit)	MX25L51245G (32Mbit)	IS25CD512 (512Kbit)
S25FL216K (16Mbit)	MX25UM51345G (512Mbit)	IS25CD025 (256Kbit)
S25FL116K (16Mbit)	MX25U25645G (256Mbit)	IS25CQ032 (32Mbit)
S25FL208K (8Mbit)	MX25U12873F (128Mbit)	IS25LQ032 (32Mbit)
S25FL204K (4Mbit)	MX25U643 (64Mbit)	IS25LQ016 (16Mbit)
S25HS256T (256Mbit)	MX25U323 (32Mbit)	IS25LQ080 (8Mbit)
S25HS512T (512Mbit)	MX25U3235F (32Mbit)	IS25LQ040B (4Mbit)
S25HS01GT (1Gbit)	MX25U1635E (16Mbit)	IS25LQ040 (4Mbit)
S25HL256T (256Mbit)	MX25U803 (8Mbit)	IS25LQ020 (2Mbit)
S25HL512T (512Mbit)	EN25Q128 (128Mbit)	IS25LQ020 (2Mbit)
S25HL01GT (1Gbit)	EN25Q64 (64Mbit)	IS25LQ010 (1Mbit)
S26HS256T (256Mbit)	EN25Q32 (32Mbit)	IS25LQ512 (512Kbit)
S26HS512T (512Mbit)	EN25Q16 (16Mbit)	IS25LQ025 (256Kbit)
S26HS01GT (1Gbit)	EN25Q80 (8Mbit)	IS25LD040 (4Mbit)
S26HL256T (256Mbit)	EN25Q40 (4Mbit)	IS25WP256 (256Mbit)
S26HL512T (512Mbit)	EN25QH128 (128Mbit)	IS25WP128 (128Mbit)

S26HL01GT (1Gbit)	EN25QH64 (64Mbit)	IS25WP064 (64Mbit)
MT25QL02GC (2Gbit)	EN25QH32 (32Mbit)	IS25WP032 (32Mbit)
N25Q00AA (1Gbit)	EN25QH16 (16Mbit)	IS25WP016 (16Mbit)
N25Q512A (512Mbit)	EN25QH80 (8Mbit)	IS25WP080 (8Mbit)
N25Q256A (256Mbit)	EN25P64 (64Mbit)	IS25WP040 (4Mbit)
NP5Q128A (128Mbit)	EN25P32 (32Mbit)	IS25WP020 (2Mbit)
N25Q128 (128Mbit)	EN25P16 (16Mbit)	IS25WQ040 (4Mbit)
N25Q064 (64Mbit)	EN25F32 (32Mbit)	IS25WQ020 (2Mbit)
N25Q032 (32Mbit)	EN25F16 (16Mbit)	IS25WD040 (4Mbit)
N25Q016 (16Mbit)	EN25F80 (8Mbit)	IS25WD020 (2Mbit)
N25Q008 (8Mbit)	EN25F40 (4Mbit)	F25L64QA (32Mbit)
N25Q00AA (1Gbit)	EN25F20 (2Mbit)	F25L32QA (32Mbit)
N25Q512A (512Mbit)	EN25T32 (32Mbit)	F25L16QA (32Mbit)
N25Q256A (256Mbit)	EN25T16 (16Mbit)	F25L14QA (32Mbit)
N25Q128A (128Mbit)	EN25T80 (8Mbit)	F25L08 (8Mbit)
N25Q064A (64Mbit)	EN25T40 (4Mbit)	F25L08 (8Mbit)
N25Q032 (16Mbit)	EN25T20 (2Mbit)	F25L04 (4Mbit)
N25Q016 (16Mbit)	EN25F10 (1Mbit)	F25L04 (4Mbit)
N25Q008 (8Mbit)	EN25S64 (64Mbit)	F25L64PA (16Mbit)
M25P128 (128Mbit)	EN25S32 (32Mbit)	F25L32PA (16Mbit)
M25P64 (64Mbit)	EN25S16 (16Mbit)	F25L16PA (16Mbit)
M25PX32 (32Mbit)	EN25S80 (8Mbit)	F25L08PA (8Mbit)
M25P32 (32Mbit)	EN25S40 (4Mbit)	F25L04PA (4Mbit)
M25PX16 (16Mbit)	EN25S20 (2Mbit)	F25L02PA (2Mbit)
M25PX16 (16Mbit)	EN25S10 (1Mbit)	LE25FU406B (4Mbit)
M25P16 (16Mbit)	SST26VF064 (64Mbit)	LE25FW406A (4Mbit)
M25P80 (8Mbit)	SST26VF064B (64Mbit)	BG25Q32A (32Mbit)
M25PX80 (8Mbit)	SST26VF032 (32Mbit)	XM25QH32B (32Mbit)
M25P40 (4Mbit)	SST26VF032 (32Mbit)	XM25QH64A (64Mbit)
M25P20 (2Mbit)	SST26VF032B (32Mbit)	XM25QH128A (128Mbit)
M25P10 (1Mbit)	SST26VF016 (16Mbit)	BY25D16 (16Mbit)
M25P05 (512Kbit)	SST26VF016 (16Mbit)	BY25Q32 (32Mbit)
M25PX64 (64Mbit)	SST26VF016B (16Mbit)	BY25Q64 (64Mbit)
M25PX32 (32Mbit)	SST25VF128B (128Mbit)	BY25Q128A (128Mbit)
M25PX16 (16Mbit)	SST25VF064C (64Mbit)	P25Q32H (32Mbit)
M25PE16 (16Mbit)	SST25VF032 (32Mbit)	P25Q16H (16Mbit)
M25PE80 (8Mbit)	SST25VF032B (32Mbit)	P25Q80H (8Mbit)
M25PE40 (4Mbit)	SST25VF016B (16Mbit)	P25D16H (16Mbit)
M25PE20 (2Mbit)	SST25VF080 (8Mbit)	P25D80H (8Mbit)
M25PE10 (1Mbit)	SST25VF080B (8Mbit)	P25D40H (4Mbit)
M45PE16 (16Mbit)	SST25VF040B (4Mbit)	P25D20H (2Mbit)
M45PE80 (8Mbit)	SST25VF020 (2Mbit)	P25D10H (1Mbit)
M45PE40 (4Mbit)	SST25VF020A (2Mbit)	P25D05H (512Kbit)

M45PE20 (2Mbit)	SST25VF010 (1Mbit)	AT25128B (128Kbit)
M45PE10 (1Mbit)	SST25VF010A (1Mbit)	AT25256B (256Kbit)
W25M512JV (512Mbit)	SST25VF512 (512Kbit)	AT25512 (512Kbit)
W25M512JW (512Mbit)	SST25PF040C (4Mbit)	AT25010A (1Kbit)
W25H02NW (2Gbit)	SST25LF020A (2Mbit)	AT25020A (2Kbit)
W25H01NW (1Gbit)	SST26WF064 (64Mbit)	AT25040A (4Kbit)
W25H512NW (512Mbit)	SST26WF032 (32Mbit)	AT25080 (8Kbit)
W25H02JV (2Gbit)	SST26WF016 (16Mbit)	AT25160 (16Kbit)
W25H01JV (1Gbit)	SST26WF080 (8Mbit)	AT25320 (32Kbit)
W25H512JV (512Mbit)	SST26WF040 (4Mbit)	AT25640 (64Kbit)
W25Q02NW (2Gbit)	SST25WF080B (8Mbit)	M95010 (1Kbit)
W25Q01NW (1Gbit)	SST25WF040 (4Mbit)	M95020 (2Kbit)
W25Q512NW (512Mbit)	SST25WF020A (2Mbit)	M95040 (4Kbit)
W25Q01NW (1Gbit)	SST25WF040B (4Mbit)	M95080 (8Kbit)
W25Q512NW (512Mbit)	SST25WF020 (2Mbit)	M95160 (16Kbit)
W25Q256FV (256Mbit)	SST25WF010 (1Mbit)	M95320 (32Kbit)
W25Q02JV (2Gbit)	SST25WF512 (512Kbit)	M95640 (64Kbit)
W25Q01JV (1Gbit)	25AA160A (16Kbit)	M95128 (128Kbit)
W25Q512JV (512Mbit)	25AA160B (16Kbit)	M95256 (256Kbit)
W25Q256JV (256Mbit)	25LC1024 (1Mbit)	M95512 (512Kbit)
W25Q128JV (128Mbit)	PM25LV016B (16Mbit)	M95M01 (1Mbit)
W25Q64JV (64Mbit)	PM25LV080B (8Mbit)	M95M02 (2Mbit)
W25Q32JV (32Mbit)	PM25LV040 (4Mbit)	X25650 (64Kbit)
W25Q01 (256Mbit)	PM25LV020 (2Mbit)	

Serial NAND Devices (SPI-NAND):

MT29F1G01ABA (1Gbit)	GD5F4GQ4UC (4Gbit)	TH58CVG3S0HRAIJ (8Gbit)
MT29F1G01ABB (1Gbit)	GD5F4GQ4RC (4Gbit)	TC58CYG0S3HRAIJ (1Gbit)
MT29F2G01AAA (2Gbit)	W25N512GV (512Mbit)	TC58CYG1S3HRAIJ (2Gbit)
MT29F2G01ABA (2Gbit)	W25N01GV (1Gbit)	TC58CYG2S0HRAIJ (4Gbit)
MT29F2G01ABB (2Gbit)	W25M02GV (2Gbit)	TH58CYG3S0HRAIJ (8Gbit)
MT29F4G01ADA (4Gbit)	W25N02KV (2Gbit)	PN26Q01AWSIUG (1Gbit)
MT29F4G01AAA (4Gbit)	W25N512GW (512Mbit)	PN26Q02AWSIUG (2Gbit)
GD5F1GQ4UB (1Gbit)	W25N01GW (1Gbit)	PN26G01AWSIUG (1Gbit)
GD5F1GQ4RB (1Gbit)	W25M02GW (2Gbit)	XT26G01AWSEGA (1Gbit)
GD5F2GQ4UB (2Gbit)	TC58CVG0S3 (1Gbit)	XT26G02AWSEGA (2Gbit)
GD5F2GQ4RB (2Gbit)	TC58CVG1S3 (2Gbit)	XT26G01BWSEGA (1Gbit)
GD5F4GQ4UA (4Gbit)	TC58CVG2S0 (4Gbit)	XT26G02BWSIGA (2Gbit)
GD5F4GQ4UB (4Gbit)	TC58CYG0S3 (1Gbit)	MX35LF1GE4AB (1Gbit)
GD5F4GQ4RB (4Gbit)	TC58CYG1S3 (2Gbit)	MX35LF2GE4AB (2Gbit)
GD5F1GQ4UC (1Gbit)	TC58CYG2S0 (4Gbit)	IS37/38SML01G1 (1Gbit)
GD5F1GQ4RC (1Gbit)	TC58CVG0S3HRAIJ (1Gbit)	F50L1G41A (1Gbit)

GD5F2GQ4UC (2Gbit)
GD5F2GQ4RC (2Gbit)

TC58CVG1S3HRAIJ (2Gbit)
TC58CVG2S0HRAIJ (4Gbit)

FM25G01 (1Gbit)
FM25G02 (2Gbit)

Parallel NOR Devices (Parallel Flash NOR/NAND):

28F064P30(T) (64Mbit)	AT49F020 (2Mbit)	M28W320FCB (32Mbit)
28F064P30(B) (64Mbit)	AT49F040 (4Mbit)	M28W320BT (32Mbit)
28F128P30(T) (128Mbit)	AT49F040T (4Mbit)	M28W320BB (32Mbit)
28F128P30(B) (128Mbit)	AT49BV/LV16X (16Mbit)	M28W640ECT (64Mbit)
28F256P30(T) (256Mbit)	AT49BV/LV16XT (16Mbit)	M28W640ECB (64Mbit)
28F256P30(B) (256Mbit)	MX29F040 (4Mbit)	M28W320FSU (32Mbit)
A28F512 (512Kbit)	MX29F080 (8Mbit)	M28W640FSU (64Mbit)
28F320J5 (32Mbit)	MX29F016 (16Mbit)	M58LW064D (64Mbit)
28F640J5 (64Mbit)	MX29F800T (8Mbit)	M29F200FT (2Mbit)
28F320J3 (32Mbit)	MX29F800B (8Mbit)	M29F200FB (2Mbit)
28F640J3 (64Mbit)	MX29F1610 (16Mbit)	M29F400FT (4Mbit)
28F128J3 (128Mbit)	MX29F1610MC (16Mbit)	M29F400FB (4Mbit)
28F256J3 (256Mbit)	MX29F1610MC (16Mbit)	M29F800FT (8Mbit)
28F800C3(T) (8Mbit)	MX29F1610A (16Mbit)	M29F800FB (8Mbit)
28F800C3(B) (8Mbit)	MX29F1610B (16Mbit)	M29F160FT (16Mbit)
28F160C3(T) (16Mbit)	MX29SL800CT (8Mbit)	M29F160FB (16Mbit)
28F160C3(B) (16Mbit)	MX29SL800CB (8Mbit)	M29W160ET (16Mbit)
28F320C3(T) (32Mbit)	MX29L3211 (32Mbit)	M29W160EB (16Mbit)
28F320C3(B) (32Mbit)	MX29LV040 (4Mbit)	M29W320DT (32Mbit)
28F640C3(T) (64Mbit)	MX29LV400T (4Mbit)	M29W320DB (32Mbit)
28F640C3(B) (64Mbit)	MX29LV400B (4Mbit)	M29W640GH (64Mbit)
28F008SA (8Mbit)	MX29LV800T (8Mbit)	M29W640GL (64Mbit)
28F400B3(T) (4Mbit)	MX29LV800B (8Mbit)	M29W640GT (64Mbit)
28F400B3(B) (4Mbit)	MX29LV160DT (16Mbit)	M29W640GB (64Mbit)
28F800B3(T) (8Mbit)	MX29LV160DB (16Mbit)	M29W128GH (128Mbit)
28F800B3(B) (8Mbit)	MX29LV320T (32Mbit)	M29W128GL (128Mbit)
28F160B3(T) (16Mbit)	MX29LV320B (32Mbit)	M29W256GH (256Mbit)
28F160B3(B) (16Mbit)	MX29LV640ET (64Mbit)	M29W256GL (256Mbit)
28F320B3(T) (32Mbit)	MX29LV640EB (64Mbit)	M29W512G (512Mbit)
28F320B3(B) (32Mbit)	MX29GL128F (128Mbit)	MT28EW128 (128Mbit)
28F640B3(T) (64Mbit)	MX29GL256F (256Mbit)	MT28EW256 (256Mbit)
28F640B3(B) (64Mbit)	MX29LV128DT (128Mbit)	MT28EW512 (512Mbit)
28F004B5(T) (4Mbit)	MX29LV128DB (128Mbit)	MT28EW01G (1Gbit)
28F200B5(T) (2Mbit)	S29AL004D(B) (4Mbit)	MT28FW02G (2Gbit)
28F400B5(T) (4Mbit)	S29AL004D(T) (4Mbit)	LHF00L15 (32Mbit)
28F800B5(T) (8Mbit)	S29AL008J(B) (8Mbit)	LH28F160S3 (16Mbit)
28F004B5(B) (4Mbit)	S29AL008J(T) (8Mbit)	LH28F320S3 (32Mbit)
28F200B5(B) (2Mbit)	S29AL016M(B) (16Mbit)	LH28F160BJE (16Mbit)

28F400B5(B) (4Mbit)	S29AL016M(T) (16Mbit)	LH28F320BJE (32Mbit)
28F800B5(B) (8Mbit)	S29AL016D(B) (16Mbit)	LH28F008SCT (8Mbit)
AM29F200(T) (2Mbit)	S29AL016D(T) (16Mbit)	LH28F016SCT (16Mbit)
AM29F200(B) (2Mbit)	S29AL016J(T) (16Mbit)	TC58FVT800 (8Mbit)
AM29LV002B(T) (2Mbit)	S29AL016J(B) (16Mbit)	TC58FVB800 (8Mbit)
AM29LV002B(B) (2Mbit)	S29AL032D (32Mbit)	TC58FVT160 (16Mbit)
AM29LV065D (64Mbit)	S29AL032D(B) (32Mbit)	TC58FVB160 (16Mbit)
AM29F040 (4Mbit)	S29AL032D(T) (32Mbit)	TC58FVT321 (32Mbit)
AM29F010B (1Mbit)	S29GL128 (128Mbit)	TC58FVB321 (32Mbit)
AM29F040B (4Mbit)	S29GL256 (256Mbit)	TC58FVM5T2A (32Mbit)
AM29F080B (8Mbit)	S29GL512 (512Mbit)	TC58FVM5B2A (32Mbit)
AM29F016B (16Mbit)	S29GL01G (1Gbit)	TC58FVM5T3A (32Mbit)
AM29F016D (16Mbit)	S29JL032J(T) (32Mbit)	TC58FVM5B3A (32Mbit)
AM29F032B (32Mbit)	S29JL032J(B) (32Mbit)	TC58FYM5T2A (32Mbit)
AM29F032B (32Mbit)	S29JL064J (64Mbit)	TC58FYM5B2A (32Mbit)
AM29LV200(T) (2Mbit)	S29GL032M (32Mbit)	TC58FYM5T3A (32Mbit)
AM29LV200(B) (2Mbit)	S29GL032M(B) (32Mbit)	TC58FYM5B3A (32Mbit)
AM29F200(T) (2Mbit)	S29GL032M(T) (32Mbit)	TC58FVM6T2A (64Mbit)
AM29F200(B) (2Mbit)	S29JL032J(B) (32Mbit)	TC58FVM6B2A (64Mbit)
AM29LV400(T) (4Mbit)	S29JL032J(T) (32Mbit)	TC58FVM6T5B (64Mbit)
AM29LV400(B) (4Mbit)	S29JL032J(B) (32Mbit)	TC58FVM6B5B (64Mbit)
AM29F400(T) (4Mbit)	S29JL032J(T) (32Mbit)	TC58FYM6T2A (64Mbit)
AM29F400(B) (4Mbit)	S29JL032J(B) (32Mbit)	TC58FYM6B2A (64Mbit)
AM29LV800(T) (8Mbit)	S29JL032J(T) (32Mbit)	TC58FVM7T2A (128Mbit)
AM29LV800(B) (8Mbit)	S29GL064M (64Mbit)	TC58FVM7B2A (128Mbit)
AM29F800(T) (8Mbit)	S29GL064M (64Mbit)	TC58FYM7T2A (128Mbit)
AM29F800(B) (8Mbit)	S29GL064M(T) (64Mbit)	TC58FYM7B2A (128Mbit)
AM29LV160B(T) (16Mbit)	S29GL064M(B) (64Mbit)	K8P1615UQB (16Mbit)
AM29LV160B(B) (16Mbit)	S29GL064M (64Mbit)	K8D1716UT (16Mbit)
AM29DL322G(T) (32Mbit)	S29GL128M (128Mbit)	K8D1716UB (16Mbit)
AM29DL322G(B) (32Mbit)	S29GL256M (256Mbit)	K8D3216UT (32Mbit)
AM29DL323G(T) (32Mbit)	S29GL032N (32Mbit)	K8D3216UB (32Mbit)
AM29DL323G(B) (32Mbit)	S29GL064N (64Mbit)	K8P3215UQB (32Mbit)
AM29DL324G(T) (32Mbit)	S29GL128N (128Mbit)	K8D6316UT (64Mbit)
AM29DL324G(B) (32Mbit)	S29GL256N (256Mbit)	K8D6316UB (64Mbit)
AM29LV320D(T) (32Mbit)	S29GL512N (512Mbit)	K8P6415UQB (64Mbit)
AM29LV320D(B) (32Mbit)	S29GL128S (128Mbit)	K8P2716UZC (128Mbit)
AM29LV320M(T) (32Mbit)	S29GL256S (256Mbit)	K8Q2815UQB (128Mbit)
AM29LV320M(B) (32Mbit)	S29GL512S (512Mbit)	K8P5516UZB (256Mbit)
W49F020 (2Mbit)	S29GL128P (128Mbit)	K8P5615UQA (256Mbit)
W49F002U (2Mbit)	S29GL256P (256Mbit)	HY29F040 (4Mbit)
W29EE512 (512Kbit)	S29GL512P (512Mbit)	HY29F080 (8Mbit)
W29C010 (1Mbit)	S29GL01GP (1Gbit)	HY29F400T (4Mbit)

W29C020 (2Mbit)	S29GL512T (512Mbit)	HY29F400B (4Mbit)
W29C040 (4Mbit)	S29GL01GS (1Gbit)	HY29F800T (8Mbit)
W29GL256S (256Mbit)	S29GL01GT (1Gbit)	HY29F800B (8Mbit)
W29GL256P (256Mbit)	S70GL02G (1Gbit)	HY29LV400T (4Mbit)
W29GL128C (128Mbit)	S29PL032J (32Mbit)	HY29LV400B (4Mbit)
W29GL064CT (64Mbit)	S29PL064J (64Mbit)	HY29LV800T (8Mbit)
W29GL064CB (64Mbit)	S29PL127J (128Mbit)	HY29LV800B (8Mbit)
W29GL032CT (32Mbit)	M29F200T (2Mbit)	HY29LV160T (16Mbit)
W29GL032CB (32Mbit)	M29F200B (2Mbit)	HY29LV160B (16Mbit)
29EE010 (1Mbit)	M29F400T (4Mbit)	HY29LV320T (32Mbit)
29LE010/29VE010 (1Mbit)	M29F400B (4Mbit)	HY29LV320B (32Mbit)
39VF401C/39LF401C (4Mbit)	M29F080A (8Mbit)	MBM29F400TA (4Mbit)
39VF402C/39LF402C (4Mbit)	M29F800T (8Mbit)	MBM29F400BA (4Mbit)
39SF512 (512Kbit)	M29F800B (8Mbit)	MBM29F800TA (8Mbit)
39SF010 (1Mbit)	M29W400DT (4Mbit)	MBM29F800BA (8Mbit)
39SF020 (2Mbit)	M29W400DB (4Mbit)	MBM29LV200TC (2Mbit)
39SF040 (4Mbit)	M29W800AT (8Mbit)	MBM29LV200BC (2Mbit)
39LF010 (1Mbit)	M29W800AB (8Mbit)	MBM29LV400TC (4Mbit)
39LF020 (2Mbit)	M29W800DT (8Mbit)	MBM29LV400BC (4Mbit)
39LF040 (4Mbit)	M29W800DB (8Mbit)	MBM29LV800TA (8Mbit)
39VF200 (2Mbit)	M29W160ET (16Mbit)	MBM29LV800BA (8Mbit)
39VF400 (4Mbit)	M29W160EB (16Mbit)	MBM29LV160T (16Mbit)
39VF800 (8Mbit)	M29D323DT (32Mbit)	MBM29LV160B (16Mbit)
39VF160 (16Mbit)	M29D323DB (32Mbit)	MBM29LV320TE (32Mbit)
39VF1681 (16Mbit)	M29W320DT (32Mbit)	MBM29LV320BE (32Mbit)
39VF1682 (16Mbit)	M29W320DB (32Mbit)	MBM29DL32XTD (32Mbit)
39VF1601 (16Mbit)	M29W320ET (32Mbit)	MBM29DL32XBD (32Mbit)
39VF1602 (16Mbit)	M29W320EB (32Mbit)	EN29LV400AT (4Mbit)
39VF3201 (32Mbit)	M29W640GH (64Mbit)	EN29LV400AB (4Mbit)
39VF3202 (32Mbit)	M29W640GL (64Mbit)	EN29LV800AT (8Mbit)
39VF6401 (64Mbit)	M29W640GT (64Mbit)	EN29LV800AB (8Mbit)
39VF6402 (64Mbit)	M29W640GB (64Mbit)	EN29LV160AT (16Mbit)
39VF6401B (64Mbit)	M29W128GH (128Mbit)	EN29LV160AB (16Mbit)
39VF6402B (64Mbit)	M29W128GL (128Mbit)	EN29LV320AT (32Mbit)
AT29C010A (1Mbit)	M28W160CT (16Mbit)	EN29LV320AB (32Mbit)
AT49F512 (512Kbit)	M28W160CB (16Mbit)	EN29LV640 (64Mbit)
AT49F010 (1Mbit)	M28W320FCT (32Mbit)	

SLC NAND Devices (Parallel Flash NOR/NAND):

NAND128W3A (128Mbit)	TC58BVG0S3HTA00 (1Gbit)	HY27US08121B (512Mbit)
NAND256R3A (256Mbit)	TC58DVG02D5TA00 (1Gbit)	HY27UF081G2A (1Gbit)
NAND256W3A (256Mbit)	TC58NVG0S3ETA00 (1Gbit)	HY27UF161G2A (1Gbit)

NAND256R4A (256Mbit)	TC58NVG0S3HTA00 (1Gbit)	H27U1G8F2B (1Gbit)
NAND256W4A (256Mbit)	TC58NVG0S3HTAI0 (1Gbit)	H27U1G8F2CTR (1Gbit)
NAND512R3A (512Mbit)	TC58BVG0S3HTAI0 (1Gbit)	HY27UF081G2M (1Gbit)
NAND512W3A (512Mbit)	TC58NVG1S3HTA00 (2Gbit)	HY27US081G1M (1Gbit)
NAND512R4A (512Mbit)	TC58BVG1S3HTA00 (2Gbit)	HY27SF081G2M (1Gbit)
NAND512W4A (512Mbit)	TC58NVG1S3HTAI0 (2Gbit)	HY27UF082G2B (2Gbit)
NAND01GR3A (1Gbit)	TC58NVG2S0HTA00 (4Gbit)	HY27UF082G2A (2Gbit)
NAND01GW3A (1Gbit)	TC58NVG2S3ETA00 (4Gbit)	H27UAG8T2M (16Gbit)
NAND01GR4A (1Gbit)	TC58NVG2S0HTAI0 (4Gbit)	H27UAG8T2B (8Gbit)
NAND01GW4A (1Gbit)	TH58NVG2S3HTA00 (4Gbit)	H27U2G8F2C (2Gbit)
NAND01GR3B (1Gbit)	TC58BVG2S0HTAI0 (4Gbit)	H27U2G8F2C (2Gbit)
NAND01GW3B (1Gbit)	TH58NVG3S0HTA00 (8Gbit)	H27U2G6F2C (2Gbit)
NAND01GR4B (1Gbit)	TH58NVG3S0HTAI0 (8Gbit)	H27S2G8F2C (2Gbit)
NAND01GW4B (1Gbit)	TC58NVG3S0FTA00 (8Gbit)	H27S2G6F2C (2Gbit)
NAND02GR3B (2Gbit)	TC58NVG3S0FTA00 (8Gbit)	H27UBG8T2B (32Gbit)
NAND02GW3B (2Gbit)	TC58NVG6D2HTA00 (64Gbit)	H27UBG8T2C (32Gbit)
NAND02GR4B (2Gbit)	TC58NVG2S0HBAI4 (4Gbit)	H27U4G8F2D (4Gbit)
NAND02GW4B (2Gbit)	W29N01GW (1Gbit)	H27U4G6F2D (4Gbit)
NAND04GW3B (4Gbit)	W29N01GV (1Gbit)	H27S4G8F2D (4Gbit)
NAND04GW3B (4Gbit)	W29N01HV (1Gbit)	H27S4G6F2D (4Gbit)
NAND08GW3B (8Gbit)	W29N02GV (2Gbit)	H27UCG8T2FTR (64Gbit)
MT29F1G08ABAEA (1Gbit)	W29N04GV (4Gbit)	HY27UG084G2M (4Gbit)
MT29F1G08ABBEA (1Gbit)	W29N08GV (8Gbit)	HY27UG084GDM (4Gbit)
MT29F1G08ABADAWP (1Gbit)	W29N08GV (8Gbit)	HY27UG164G2M (4Gbit)
MT29F2G08AAB (2Gbit)	W29N04GZ (4Gbit)	HY27UF084G2M (4Gbit)
MT29F2G08ABBFA (2Gbit)	W29N04GW (4Gbit)	S34ML01G1 (1Gbit)
MT29F2G08ABAF (2Gbit)	MX30LF1208AA (512Mbit)	S34ML02G1 (2Gbit)
MT29F2G08ABBEA (2Gbit)	MX30LF1GE8AB (1Gbit)	S34ML04G1 (4Gbit)
MT29F2G16ABBEA (2Gbit)	MX30UF1G18AC (1Gbit)	S34ML01G2 (1Gbit)
MT29F2G08ABAEA (2Gbit)	MX30LF1G18AC (1Gbit)	S34ML02G2 (2Gbit)
MT29F2G16ABAEA (2Gbit)	MX30LF1G08AA (1Gbit)	S34ML04G2 (4Gbit)
MT29F2G08ABBEAH4 (2Gbit)	MX30LF2G18AC (2Gbit)	S34ML01G2 (1Gbit)
MT29F2G16ABBEAH4 (2Gbit)	MX30UF2G18AC (2Gbit)	S34ML02G2 (2Gbit)
MT29F2G08ABAEAH4 (2Gbit)	MX30LF2G28AB (2Gbit)	S34ML04G2 (4Gbit)
MT29F2G16ABAEAH4 (2Gbit)	MX30LF2GE8AB (2Gbit)	S34ML04G3 (4Gbit)
MT29F4G08BAB (4Gbit)	MX30UF2G18AB (2Gbit)	S34MS01G200 (4Gbit)
MT29F4G08AAA (4Gbit)	MX30UF2G28AB (1Gbit)	S34MS02G200 (4Gbit)
MT29F4G08BABWP (4Gbit)	MX30LF4G18AC (4Gbit)	S34MS04G200 (4Gbit)
MT29F4G08ABA (4Gbit)	MX30UF4G18AB (4Gbit)	S34MS01G204 (4Gbit)
MT29F4G08ABADAWP (4Gbit)	MX30LF4G28AB (4Gbit)	S34MS02G204 (4Gbit)
MT29F4G08ABAEA (4Gbit)	MX30LF4GE8AB (4Gbit)	S34MS04G204 (4Gbit)
MT29F4G08ABAEA (4Gbit)	MX30UF4G28AB (2Gbit)	FMND1G08U3D (1Gbit)
MT29F4G08ABAEA (4Gbit)	MX60LF8G18AC (8Gbit)	FMND1G16U3D (1Gbit)

MT29F4G16ABADA (4Gbit)	MX60LF8G28AB (8Gbit)	FMND1G08S3D (1Gbit)
MT29F8G08DAA (4Gbit)	K9F2808U0C (128Mbit)	FMND1G16S3D (1Gbit)
MT29F8G08BAA (8Gbit)	K9K2G08U0M (2Gbit)	FMND2G08U3D (2Gbit)
MT29F8G08ABACA (8Gbit)	K9K1G08U0M (1Gbit)	FMND2G16U3D (2Gbit)
MT29F8G08ABABA (8Gbit)	K9F1208U0C (512Mbit)	FMND2G08S3D (2Gbit)
MT29F16G08CBACA (16Gbit)	K9F5608U0D (256Mbit)	FMND2G16S3D (2Gbit)
MT29F32G08CBACAWP (32Gbit)	K9F1G08U0A (1Gbit)	FMND4G08U3B (4Gbit)
MT29F32G08CBACAWP (32Gbit)	K9F1G08U0B (1Gbit)	FMND4G16U3B (4Gbit)
MT29F64G08CFACAWP (64Gbit)	K9F1G08U0C (1Gbit)	FMND4G08S3B (4Gbit)
MT29F64G08CEACAD1 (64Gbit)	K9F1G08U0D (1Gbit)	FMND4G16S3B (4Gbit)
MT29F128G08CXACAD1 (64Gbit)	K9F1G08U0C (1Gbit)	FMND4G08U3C (4Gbit)
MT29F64G08CECCBH1 (64Gbit)	K9F1G08B0C (1Gbit)	FMND4G16U3C (4Gbit)
MT29F64G08CBABA (64Gbit)	K9F1G08U0E (1Gbit)	FMND4G08S3C (4Gbit)
MT29F64G08CBABB (64Gbit)	K9F2G08X0 (2Gbit)	FMND4G16S3C (4Gbit)
MT29F64G08CBCBB (64Gbit)	K9F2G08U0C (2Gbit)	FMND4G08U3F (4Gbit)
MT29F64G08CEACA (64Gbit)	K9F2G08U0M (4Gbit)	FMND4G16U3F (4Gbit)
MT29F64G08CECCB (64Gbit)	K9G8G08U0B (8Gbit)	FMND4G08S3F (4Gbit)
MT29F64G08CFACA (64Gbit)	K9W8G08U1M (4Gbit)	FMND4G16S3F (4Gbit)
MT29F64G08CFACBWP (128Gbit)	K9F4G08U0A (4Gbit)	DSND8G08U3N (8Gbit)
MT29F128G08CKCCBH2 (128Gbit)	K9F4G08U0B (4Gbit)	DSND8G16U3N (8Gbit)
MT29F256G08CUCCBH3 (128Gbit)	K9GAG08U0E (16Gbit)	DSND8G08S3N (8Gbit)
MT29F128G08CECBB (128Gbit)	K9GAG08U0M (16Gbit)	DSND8G16S3N (8Gbit)
MT29F128G08CFABA (128Gbit)	K9K8G08U0A (8Gbit)	DSND8G08U3M (8Gbit)
MT29F128G08CFABB (128Gbit)	K9KAG08U0M (16Gbit)	DSND8G16U3M (8Gbit)
MT29F128G08CXACA (128Gbit)	K9WAG08U1A (8Gbit)	DSND8G08S3M (8Gbit)
MT29F256G08CJABA (256Gbit)	K9K8G08U0A (8Gbit)	DSND8G16S3M (8Gbit)
MT29F256G08CJABB (256Gbit)	K9N8G08U5A (8Gbit)	FS33ND01GS1 (1Gbit)
MT29F256G08CKCBB (256Gbit)	HY27US08281A (128Mbit)	A5U1GA31ATS (1Gbit)
MT29F256G08CMCBB (256Gbit)	HY27US08561A (256Mbit)	F59L1G81MA (1Gbit)
TC58DVM92A5TAI0 (512Mbit)	HY27US16561A (256Mbit)	F59L1G81LA (1Gbit)
TC58DVM92A5TAI0 (512Mbit)	HY27SS08561A (256Mbit)	F59L2G81A (2Gbit)
TC58NVG3D4CTGIO (1Gbit)	HY27SS16561A (256Mbit)	

EEPROM devices (SPI EEPROM):

Renesas X5083	Atmel AT25128B	ST M95256
Nordic nRF24LE1	Atmel AT25256B	ST M95512
Nordic nRF24LU1+	Atmel AT25512	ST M95M01
Nordic nRF24LU1+	ST M95010	ST M95M02
Atmel AT25010A	ST M95020	Microchip 25AA160A
Atmel AT25020A	ST M95040	Microchip 25AA160B
Atmel AT25040A	ST M95080	Microchip 25XX320

Atmel AT25080	ST M95160	Microchip 25XX640
Atmel AT25160	ST M95320	Microchip 25XX512
Atmel AT25320	ST M95640	Microchip 25XX1024
Atmel AT25640	ST M95128	XICOR X25650

HyperFlash devices (HyperFlash):

Cypress S26KS128S	Cypress S26KS256S	Cypress S26KS512S
Cypress S26KL128S	Cypress S26KL256S	Cypress S26KL512S

MCU Devices with internal programmable memory (SPI):

Nordic nRF24LE1	Altera EPCS1	Altera EPCS4
Nordic nRF24LU1+	Altera EPCS16	Altera EPCS64
Altera EPCS128		

MCU Devices with internal programmable memory (JTAG):

Xilinx XC2C32A	Xilinx XC2C64A	Xilinx XC2C128
Xilinx XC2C256	Xilinx XC2C384	Xilinx XC2C512
Xilinx XC9500XL	Xilinx XC95288XL	Xilinx XC95144XL
Xilinx XC9572XL	Xilinx XC9536XL	Lattice LC4032V
Lattice LC4064V	Lattice LCMXO256	Lattice LCMXO640
Lattice LCMXO1200	Lattice LCMXO2280	

Firmware Hub devices (FWH):

AT49LH002 (2Mbit)	SST49LF003A (3Mbit)	PM49FL002 (2Mbit)
AT49LH004 (4Mbit)	SST49LF004A (4Mbit)	PM49FL004 (4Mbit)
W39V040FA (4Mbit)	SST49LF008A (8Mbit)	PM49FL008 (8Mbit)
W39V080FA (8Mbit)	SST49LF080A (8Mbit)	
SST49LF002A (2Mbit)	SST49LF016C (16Mbit)	