 <p><b>DAVE s.r.l.</b>  via Forniz, 2  33080 Porcia (PN) – Italy  Tel. +39.0434.921215  Fax. +39.0434.591631  Email: info@dave-tech.it</p>		
		DOCUMENT CODE:	<b>WP001</b>
		VERSION:	<b>1.0.0</b>
		NO. OF PAGES:	<b>4 pages</b>

*White paper*

# Flash programming techniques for production



<b>Version</b>	<b>Release date</b>	<b>Comments</b>
0.9.1	December 2004	first draft
1.0.0	February 2005	initial release

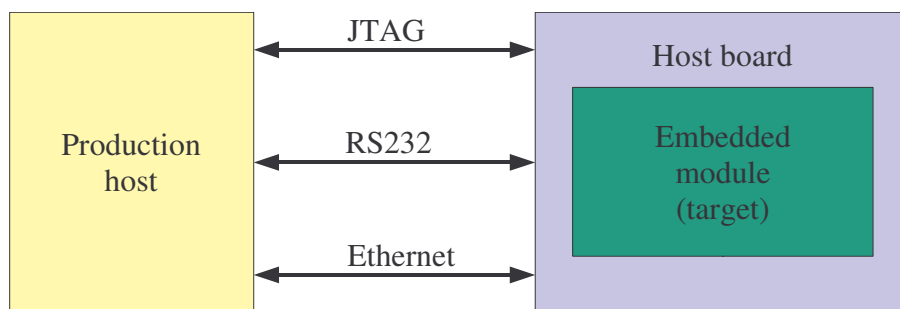
---

## Abstract

This document briefly describes the most common techniques used to program Flash memories mounted on embedded modules manufactured by DAVE Electronics System House. Special thanks go to our customers who provided qualified feedback and who contributed significantly to this document. When otherwise noted, what is described here applies to the following products:

- B2 (Samsung S3C44B0X - ARM7TDMI)
- PPChameleon (AMCC PowerPC 405EP)
- Zefeer (Cirrus Logic EP93xx – ARM920T).

The following figure shows the typical production environment that will be referred throughout the document.



*Fig. 1 - Production environment*

## Purely JTAG hardware interface

This strategy exploits the JTAG interface only. The host is connected to the embedded module through a hardware JTAG tool. The binary image is stored directly by the hardware tool under the control by an application running on the host. Burning speed may vary significantly depending on the tool characteristics (usually it is very close related to the cost). Typically these tools support two programming modes that we generically call here slow and fast. Slow mode may take up to tens of minutes to program few megabytes. Fast mode may reduce dramatically this time but it is more complex to implement because it uses some target's resources to speed up the process.

Generically speaking, the target requires some basic initializations in order to optimize the physical access to the memory to be burned.

The following list provides some examples of hardware tools:

1. Abatron BDI1000 / BDI2000 ([www.abatron.ch](http://www.abatron.ch))
2. Lauterbach TRACE32/ICD ([www.lauterbach.com](http://www.lauterbach.com))
3. MacRaigor OCDemon Wiggler, Raven ([www.macraigor.com](http://www.macraigor.com))

Technical details are clearly very tool-dependent. For this reason we suggest the interested reader to refer to vendor documentation for more information.

## External bootloader (U-Boot, Redboot, W@WBoot etc.)

This technique can be used when the target is able to run the bootloader and it does not require any JTAG interfacing. In this case production manager may write a host-side simple script to handle the flash programming. The script, by talking to the bootloader through the serial and/or Ethernet connection, can:

- download the binary image to be burned
- let the bootloader to burn it



- optionally, read id back to perform verification.

The achievable performances are usually comparable to the ones of the “fast mode” described in previous section.

## **JTAG/software flasher combined**

This technique is similar to the one used when debugging code on target through the JTAG interface. The basic operations are:

1. production manager builds a target executable program that
  - a) provides flash programming functions
  - b) embeds the binary image to be burned
  - c) in the “main” function:
    - (optional) erases the flash
    - programs the flash with the embedded binary image
    - (optional) reads it back to perform verification
    - (optional) reports error code
2. this executable is downloaded to the target through the JTAG interface
3. the executable is run on the target.

This strategy requires a JTAG hardware tool but, as the actual programming is performed by software running on the target, it usually is implemented with low performance, cheap tools. The basic hardware initialization may be performed by the JTAG tool or by the software itself.

## **Internal bootloader (Zefeer modules only)**

Some microcontrollers provide internal, rom-masked bootloader. When available, this can be used to program the flash by running on host side the proper application. Usually this application downloads the binary image through one of the microcontroller’s native communication interfaces such as serial ports, Ethernet or USB.