AN1575
APPLICATION NOTE

ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUs

by Microcontroller Division Applications

INTRODUCTION

This application note presents various ways of programming a Microcontroller (MCU) that has already been soldered on a PCB. This “on-board” programming process can be used to update either the whole firmware or only a set of data. The techniques described are fully supported features of the STMicroelectronics range of MCUs with embedded non-volatile memories (OTP, then FLASH) designed to meet the needs of equipment manufacturers and customers. The ever-growing importance of Surface Mounted Devices (SMD) with high pin count makes it more and more complex and costly to program with standard programming tools, while the constraints of time-to-market and flexibility make it necessary to be able to program or reprogram the products as late as possible in the production cycle, even at the final customer site. Aside from these manufacturing considerations, the capability of an MCU to be reprogrammed in-situ opens new application fields: customization, performance upgrades, remote maintenance, etc, are features that are valued by the end-customer.

Two main contexts have to be considered:

– Programming a MCU in the framework of a production line or when the application is not running (see Section 1).

– Programming the MCU ‘on the fly’ while it is running in the application, generally at the customer site (see Section 2).

1 APPLICATION NOT RUNNING DURING PROGRAMMING

This case covers applications that can be stopped during the programming operation: either because there is no need to keep the application running, or because the programming process does not require any special firmware to be run by the MCU. This last point means the application design is compatible with the programming method and tools used.

For this purpose, STMicroelectronics has defined an In Circuit Programming (ICP) method with a minimum of constraints on the application design.

1.1 ICP OVERVIEW

With this method, the MCU is switched by the programming tool into a special operating mode, called In Circuit Communication (ICC) mode, by an event sequence on the RESET, VPP/TEST, ICCCLK and ICCDATA pins: Throughout the entire programming operation, it remains under control of the programming tool and all transactions go through the ICCCLK and ICCDATA lines.

In order to be compatible with the ICP method, the application must therefore be equipped with a special connector for the RESET, VPP/TEST, ICCCLK and ICCDATA signals, while respecting some guidelines in order not to interfere with other parts of the application.

Note that this connector features a VPP/TEST line to allow to supply 12V externally on the VPP pin for HDFLASH-based devices.

When using the ICP method, the user has exactly the same features as on any programming tool: the whole program memory range, as well as the option bytes can be accessed.

1.2 ICP PRACTICAL IMPLEMENTATION

The ICP implementation is purely a hardware issue as shown in Figure 1. No embedded firmware needs to be developed. On the other words, the ICP method can be used to program a blank MCU.

The only sensitive point is sometimes to correctly isolate the RESET, ICCCLK and ICCDATA in order to avoid any conflict with the devices connected to these pins for purposes of the final application. This is generally ensured by simple resistors (more details can be found in the MCU datasheets or in the ST7 ICC Reference Manual).

During the ICP process, the MCU's embedded peripherals remain in reset state since no software is running to activate them. It should be noted however that the hardware watchdog is disabled.

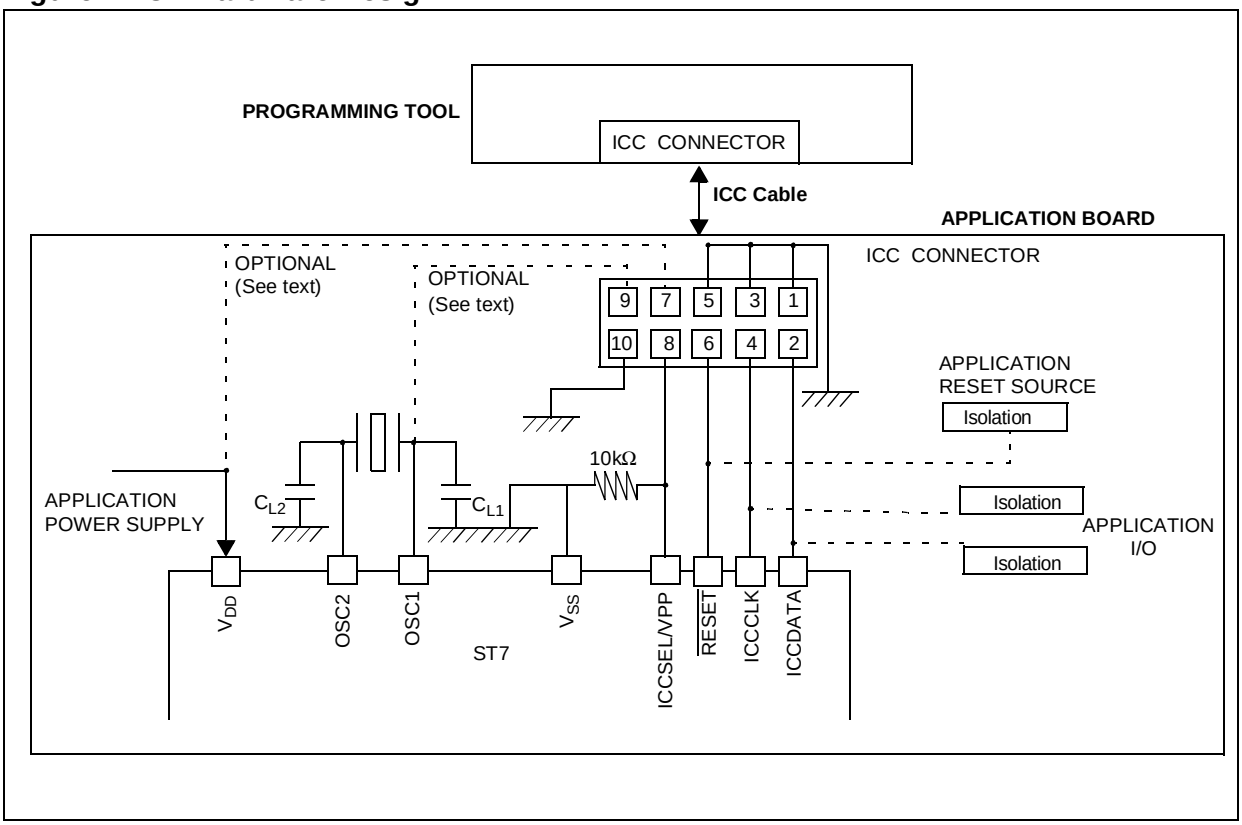
1.2.1 Application power supply

The application does not need to use its own power supply if the programming tool supplies power through pin 7 of the ICC connector. This is the case when using STMicroelectronics programming tools (Refer to the Programming Tool manual).

1.2.2 Application clock

In case the application does not have its own clock during the ICP process (Option byte not programmed yet,..), the ICC connection can be used to input an external clock from the programming tool.

Figure 1. ICP Hardware Design



2 APPLICATION RUNNING DURING PROGRAMMING

This method is primarily for applications that cannot be stopped and which have to keep some kind of software kernel running all the time (generally for security or safety reasons).

It also covers, and this is the most common reason for not using ICP, applications where the application designer chooses alternative means of implementing the programming process, specifically:

– The hardware connection to the programming tool, where the new firmware is stored

– The software protocol that defines the commands, status and data format.

A typical case is when a communication port already used in the application is available: It avoids implementing an additional connector just for the ICP process: this approach is widely used for devices with a CAN or USB interface.

Using your own protocol for on-board programming has the advantage of allowing flexibility in terms of hardware implementation. However, some firmware must already be present on the MCU in order to run the communication protocol: This method can thus be used only for re-programming, which implies the MCU has been programmed at least once before with another method. Another constraint is that it is not possible to reprogram the whole program memory of the MCU: nor can the reprogramming firmware itself be altered.

ST7 STMicroelectronics MCUs with HDFLASH and XFLASH have this capability of running user-specific firmware to perform In-Application Programming (IAP) of the MCU program memory.

This feature not only allows the application to keep some background tasks running, but also to allows you to use any type of communication protocol for the reprogramming process.

2.1 IAP OVERVIEW

The ST7 program memory is divided into several (up to 3) sectors. Only Sector 0 cannot be programmed or erased in normal (user mode) operation. So this sector can be used to execute the firmware handling the reprogramming process, without being altered.

Reprogramming any of the other sectors is possible, and is based on the execution of a specific software sequence, for which a software library, the IAP driver, is provided by STMicroelectronics.

The Sector 0 firmware handles the complete protocol and sequencing of the operations, and some tasks that cannot be stopped during the programming process.

In contrast to the ICP process, the embedded hardware peripherals can be activated by the Sector 0 firmware to carry out some background tasks. The only limitation is that the Interrupt Requests coming from these peripherals (Reception on UART, Timer overflow, ..) can be processed only after the IAP driver has completed the requested operations. A notable excep-

tion is the hardware watchdog that remains active: It must therefore be refreshed before performing any Program/Erase operations on the FLASH.

2.2 IAP PRACTICAL IMPLEMENTATION

2.2.1 General architecture

As already stated, a firmware must already have been loaded into Sector 0. This firmware has three purposes:

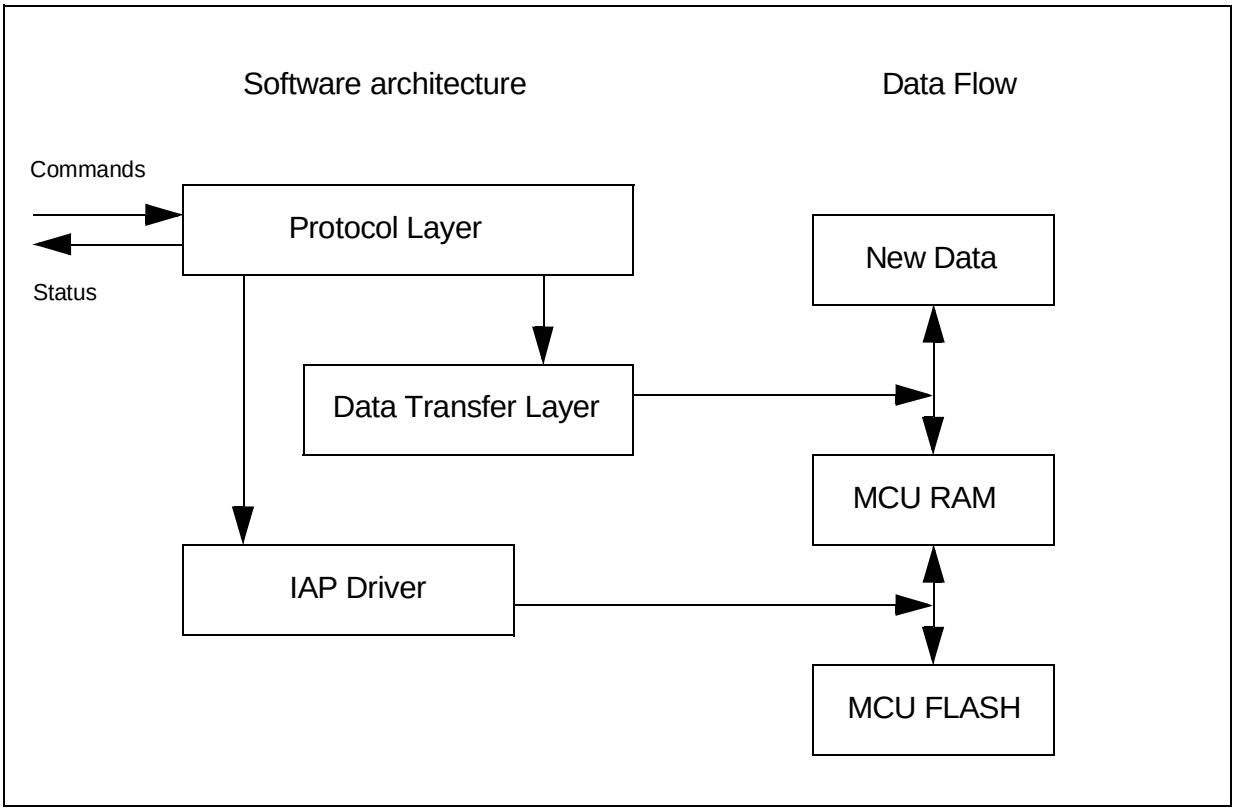
– Performing the Erase/Write operations on Sector 1 or 2 of program memory.

– Receiving the new data to be programmed.

– Managing the whole process protocol (Commands and status).

Therefore the firmware contained in the Sector 0 is generally based on three modules: Protocol layer, Data Transfer Layer and IAP Driver (Figure 2)

Figure 2. IAP Architecture



This chart only gives the generic scheme of an IAP application, however the various types of *New Data* sources, *Data Transfer* physical supports, and the *Protocol Layer* interfaces can lead to a wide range of practical implementations.

2.2.2 Implementation with no external interface

A particular implementation is the case where *New Data* source, the *Data Transfer* physical support, and the *Protocol Layer* interfaces are within the MCU itself. This is particularly suitable for calibration operations when the MCU needs to autonomously set or update some data in its program memory.

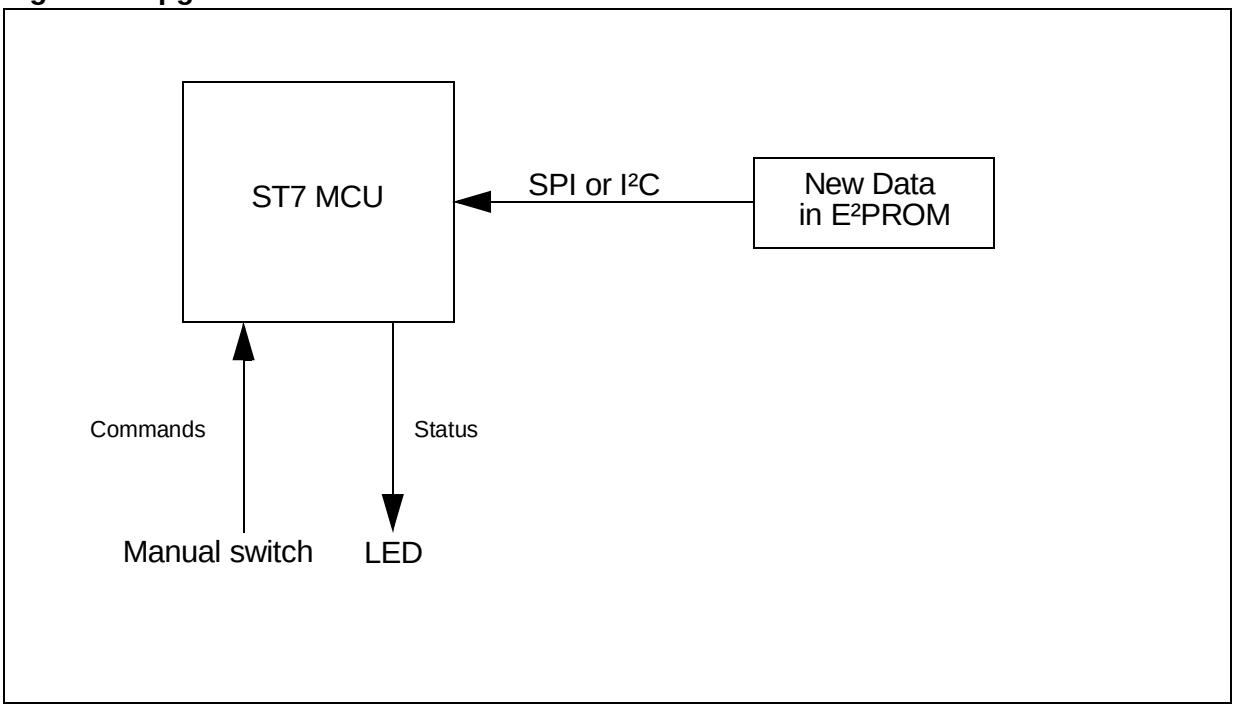
In cases where only one-time programming (during initialization for instance) is needed, IAP can save using an embedded or external data EEPROM module.

2.2.3 Upgrade from serial EEPROM

This example corresponds to STMicroelectronics Application Note AN1576: *In Application Programming drivers for ST7 HDFLASH OR XFLASH MCUs*.

In this AN1576, the *Data Transfer* module is based, as example, on a SPI or I²C driver.

Figure 3. Upgrade from serial EEPROM



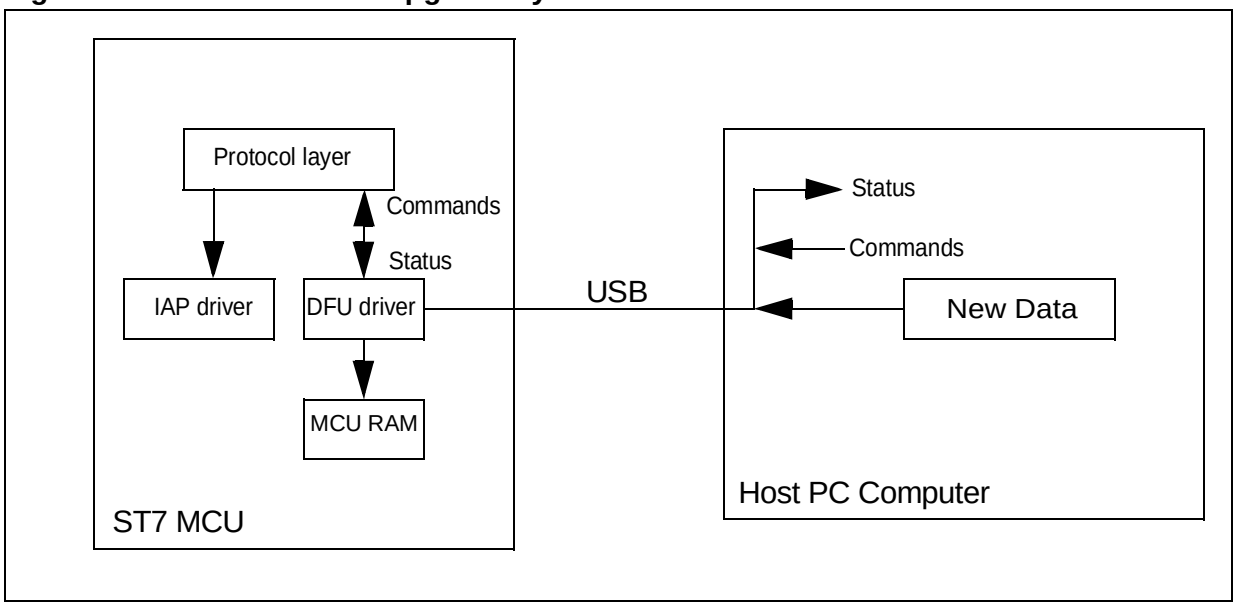
2.2.4 USB-based example

In this example, the USB link is used for the *Protocol Layer* interfaces and *Data Transfer* support.

This implementation is an ideal example of IAP because the fact that the USB link is already available means that you get a connection to the programming tool for free. On top of that, the USB protocol itself allows the same media support to be used for the New Data and the Command/Status transfer.

The final scheme is there slightly different from Figure 2 since the *Data Transfer* layer and the Protocol layer interfaces are merged into a DFU driver.

Figure 4. Device Firmware Upgrade by USB



2.2.5 Local Network example

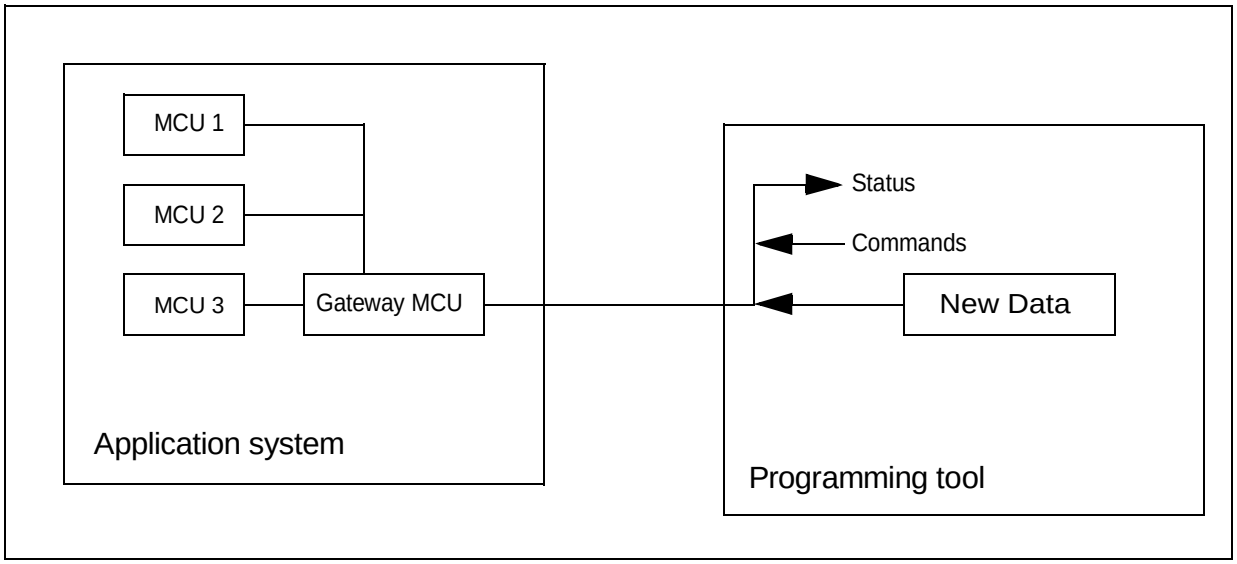
A multi-MCU application is another case where the IAP approach is particularly suitable, with two main architectural schemes: gateway or direct network connection.

2.2.5.1 Gateway MCU

The idea is to define one of the MCUs as the gateway to the programming tool, and to link (in a local network) all the other MCUs to this common node. As a consequence, only one external connector is needed to reprogram all the assembled MCUs (including the gateway), and the gateway MCU can provide additional security by preventing access to the programming lines of the other MCUs.

Ideally, the internal links already exist in the application (the CAN network is a good example), otherwise if they have to be implemented, multiplexed buses (hardware or software) such as I²C are obviously preferable as used for connecting MCU 1 & MCU 2 in Figure 5

Figure 5. IAP through a local network and gateway MCU

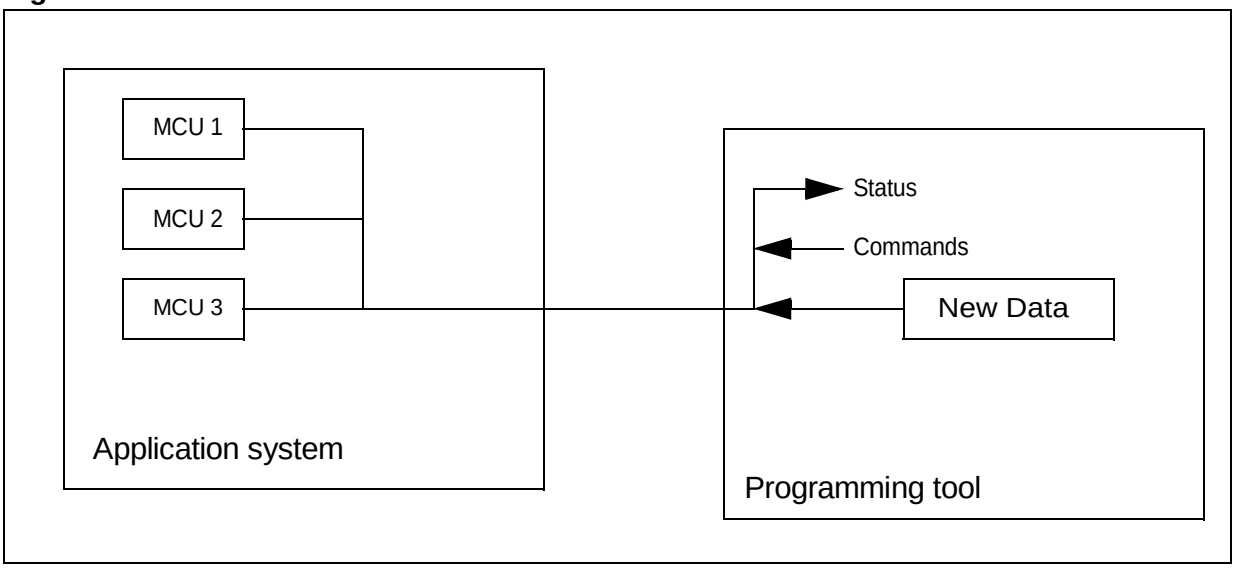


2.2.5.2 Connection to internal multiplexed bus

In this scheme there is no gateway MCU, and the programming tool is directly linked to the internal multiplexed bus of the system.

The advantage of this solution is to save some I/Os by removing the gateway MCU. It is a low cost solution that should be considered if an internal multiplexed bus is available. However, the drawback is that the connection to the programming tool also gives direct access to the whole internal system: the use of the gateway avoids this issue.

Figure 6. IAP with direct connection to the local network



3 FEATURES SUMMARY

Both ICP or IAP methods have their own advantages and drawbacks. Choosing the right solution for each application can be done based on the elements given in the Table 1.

Table 1. ICP compared to IAP

Topic	ICP	IAP
Requires a programming operation before ICP or IAP	No need	Sector 0 and option bytes need to be programmed first
Sector 0 upgrade	Possible	Not possible
Application kept running	No	Yes, from sector 0
Connection to programming tool	ICC compatibility	Vendor specific
Multiple MCU system	Multiple ICC connection	Gateway or multiplexed bus

4 GETTING MORE INFORMATION

For more detailed information on ICP and IAP please refer to:

AN1576: In-Application Programming drivers for ST7 HDFLASH or XFLASH MCUs

ST7 Flash Programming Reference Manual

ST7 ICC Reference Manual