SMH Technologies
UNIVERSAL PRODUCTION
IN–SYSTEM PROGRAMMING

03/06/2019 Samuele Stefanoni

DC11054

AN00182: flexible FRB management

FlashRunner 2.0 is a Universal In-System Programmer, which feature several options to integrate flashing into your test system. This Application Note describes how to explore and get the best in terms of system flexibility.

1. Introduction

One of the most appreciated FlashRunner 2.0 features is its “flexibility”. This terms is related to a set of appreciated features that lets users to easily integrate our product in their test system, setup their production line and startup it in short time. The integration step into a software system become critical when the flashing process is integrated with other testing machines which must be controlled by a complex operation sequence.

2. Software integration

One of the most common software used in test software machines is LabView®/Teststand® tool. Labview enable you to easily create testing procedures and control testing instruments with simple graphical steps.

LabView® is able to drive instruments in different ways, one of them is including a C/C++ library. By interfacing with this library you’ll be able to control your instruments and put all them inside your testing sequence

FlashRunner 2.0 features in standard System Software package (you can download it for free from this link). Once installed, you'll find in the installation path an "InterfaceLibrary" folder which contains all that you need to start with the FlashRunner 2.0 integration.

InterfaceLibrary folder contains:

- FR_COMM.dll, FR_COMM.h, FR_COMM.lib, standard C/C++ library
- VC80 project. A very basic example on how to integrate library inside a Visual Studio C++ application
- DotNetWrapper folder, which contains a wrapper able to let you integrate COMM.dll library in a C# project.

Full description of all the function is available on Programmer's Manual, chapter 8, available on the same installation path.

3. Flexibility

Project files are simply "collection of commands". This means that you can copy a project file directly into FlashRunner 2.0 storage memory and execute it through RUN command or you can execute every command included in the project file "one by one", or you can mix the two (with limitations depending on command order). Commands can be sent through terminal tool of FlashRunner 2.0 Workbench software or through the dll of course.

Moreover, each command (including RUN command) can be sent to a defined channel (to understand how please read chapter 4.2 of Programmer's Manual), so you can synchronize and manage each channel independently.

By doing this you can easily setup

- Serial numbering (described in Programmer's Manual chapter 5)
- Dynamic FRB selection and update

Both features above take advantage of mixing commands and project execution features, providing full flexibility to the production line.

4. Serial numbering

Although serial numbering feature is described in full details in chapter 5 Programmer's Manual, the main benefit here is to do it by fluidly use test software and dll management. For example, let's imagine you want to define a different serial number for each new board; achieving this goal requires:

- Having a standard project, valid for every board which do main flashing process with standard firmware;
- Having a process, provided by a server/external routine/Labview system process, which calculates and provide a serial number for every board;

How does it works the full process? It's very simple:

- Take the serial number externally provided by the "system".
- Provide it to FlashRunner with DYNMEMSET command on the selected channel
- Execute the project on the selected channel
- Clear the memory with DYNMEMCLEAR

5. Dynamic FRB selection and update

Often customer requires to update firmware version "in the process". Usually customer think that changing firmware to a new release requires an operator action, in order to create a new project and convert it again to FRB. Since firmware filename is defined inside the project this is the first conclusion that comes to a FlashRunner 2.0 user.

This process instead could be automatized in the following way and is not so far from the "mixing" process we used for serial numbering:

- Use Labview® to download the new firmware (.bin, .hex, .s19) from the server (or whatever customer requires)
- Use Labview® to call our command line frbconverter.exe tool in order to create frb file with the new firmware release
- Take our from the project #TPSETSRC and send it as a single command before project exectution. This way you can update your release and changing file name without need to modify and update the project file.
- Execute the project like it is (without #TPSETSRC directive)

If you need extra check you can add FRBREADCRC command just after firmware download to keep track if really modifications occurred between the last download or not. If no changes occurred no file upload needs to be done on FlashRunner 2.0 storage memory (saving time and SD card memory retention).