

USING MULTI-PROJECT FEATURE ON ELNEC DEVICE PROGRAMMERS

Application note

June 2021

an_elnec_multi_prj, version 1.2



Disclaimer:

This application note describes how to program NAND flash devices using Elnec device programmers. Before reading this document, user should be familiarized with NAND flash devices. There are plentiful sources available through the web containing detailed informations about NAND flash internal organization, errors in NAND flash, basic algorithms, etc. Study, please, your device datasheet thoroughly, at least.

This application note is provided by our technical support department to help our customers and is provided “as-is”, without warranty of any kind, either expressed or implied. We reserve the rights to make changes to the information available in this application note at any time and assume no liability for applications assistance, customer product design and any damages arising from the use of this application note.

This application note may refer to various products and brand names that may be claimed as property of their respective owners.



CONTENTS

- 1. INTRODUCING THE MULTI-PROJECT FEATURE.....5**
 - 1.1. TERMINOLOGY.....5
 - 1.2. PART NAMES CONVENTION.....6
 - 1.3. USING MULTI-PROJECT WIZARD.....6
 - 1.3.1. Building new Multi-Project file.....8
 - 1.3.2. Loading existing Multi-Project file.....12
 - 1.3.3. Running the multi-chip device operation.....13
 - 1.3.3.1. Single programming using Pg4uw.....14
 - 1.3.3.2. Multi-programming using Pg4uwMC.....16
 - 1.4. LIMITATIONS OF MULTI-PROJECT FEATURE.....17
- 2. USING THE MULTI-PROJECT FEATURE FOR VARIOUS TYPES OF PRACTICAL TASKS.....18**
 - 2.1. USING MULTI-PROJECT FEATURE FOR PROGRAMMING MULTI-CHIP DEVICES.....18
 - 2.2. USING MULTI-PROJECT FEATURE FOR PROGRAMMING MULTIPLY PARTITIONS INTO SINGLE NAND FLASH.....20
 - 2.3. USING MULTI-PROJECT FEATURE FOR PROGRAMMING MULTIPLY DAISY-CHAINED JTAG DEVICES.....22

PREFACE

Small form, fast speed and low power consumption – these are three most demanding requests from mobile devices manufacturers. The chip makers respond in simple way – by mixing various memories into single package, so called multi-chip device. There is a wide variety of mixtures of memory types (NOR, NAND, eMMC, OneNAND, SRAM, DRAM,...) and capacities (from several mega-bytes up to giga-bytes) available on the market.

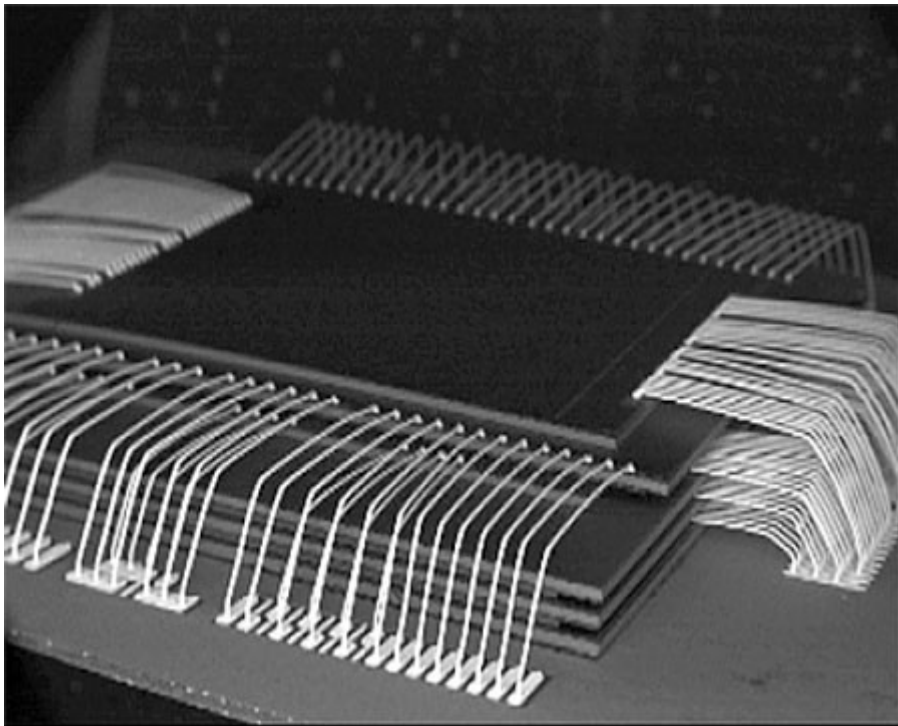


Figure 1: Look inside the MCP device.

These multi-chip devices put specific requirements on programming process. The programmer must be able to conduct various memory types and switch between them again and again. There must be a high level of understanding between programmer manufacturer and user / technician in programming centre, regarding the issues such as buffer organization, operation settings and similar.

Elnec can offer a solution for these problems – Multi-Project.

1. INTRODUCING THE MULTI-PROJECT FEATURE

Multi-Project feature was released in version 2.50 of Pg4uw control software. This chapter gives you the introduction to the topic and shows how to work with dedicated tool – Multi-Project Wizard.

1.1. TERMINOLOGY

There are several new terms that should be explained before starting any work using Multi-Project feature:

- **Multi-Project** – special feature designed to simplify the programming tasks for multi-chip devices.
- **Multi-chip device** – (memory) device with two or more independent chips (of the same or various types) in single package.
- **Sub-device** – an individual part of multi-chip device. Sub-device is selectable from Pg4uw device list. Once selected, you can work with respective device in fully manner. You can define, test and save the Project file for the particular device.
- **Master-device** – a multi-chip device unit, consists of Sub-devices. Master-device is selectable from Pg4uw device list, too. Once selected, you can use Multi-Project Wizard for building the Multi-Project file from individual Project files and save / load / execute it.
- **Project file** – a special file that combines buffer data, device operation options, special options and some level of safety features. It completely defines the way how to treat the device. Once saved, it can be reloaded at any time so the operation can be repeated exactly. Typically, Project files use .eproj file extension. Probably, you are familiar with Project files already from your previous experience with Pg4uw – Elnec device programmers control software.
- **Multi-Project file** – a special file that packs user selected Project files. Multi-Project file can include one or more Project files for individual Sub-devices. Typically, Multi-Project files use .eproj-m file extension.
- **Multi-Project Wizard** – a dedicated tool for working with Multi-project files. The wizard allows user to select Project files that have to be included in Multi-Project file and combine them into single Multi-Project file. The wizard also allows to run the multi-chip device operation(s) according to Project files included in Multi-Project file.

1.2. PART NAMES CONVENTION

Overview of convention used for Master-device and Sub-device part names in Pg4uw device list:

Master-device: Multi-chip_device_original_part_name [package_type]
 Sub-devices: Multi-chip_device_original_part_name [package_type] (part_1)
 Multi-chip_device_original_part_name [package_type] (part_2)
 ...
 Multi-chip_device_original_part_name [package_type] (part_n)

Examples:

Master-device: TV0057A002CAGD [FBGA107]
 Sub-devices: TV0057A002CAGD [FBGA107] (NAND)
 TV0057A002CAGD [FBGA107] (NOR)

Master-device: M30W0R6500T0 [LFBGA88]
 Sub-devices: M30W0R6500T0 [LFBGA88] (Flash1)
 M30W0R6500T0 [LFBGA88] (Flash2)

1.3. USING MULTI-PROJECT WIZARD

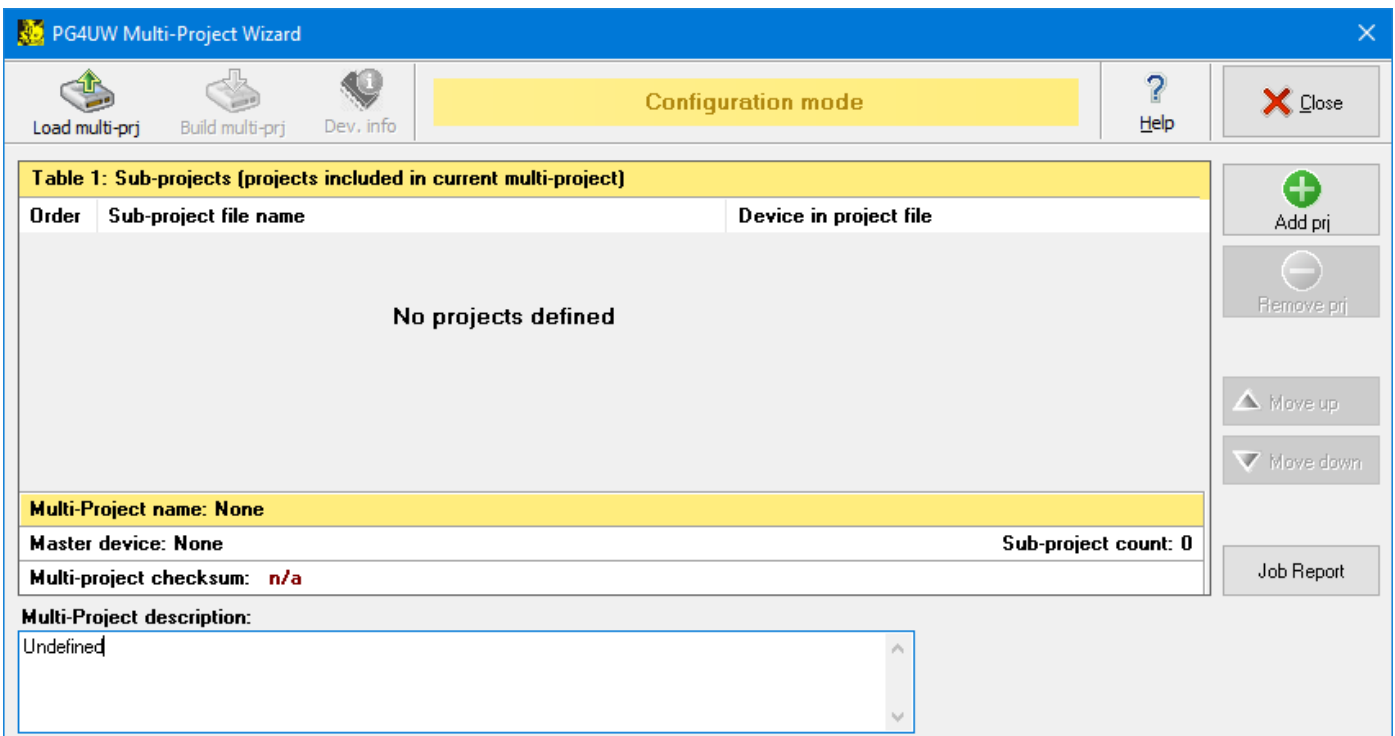


Figure 2: Empty Pg4uw Multi-Project Wizard window.

Multi-chip device operation requires the Multi-Project file, which contains partial Project files for all accessed Sub-devices of Master-device. The Multi-Project file can be created only using Multi-Project Wizard (see Figure 2). The Wizard can be accessed by selecting the Master-device from Pg4uw device list, or from Pg4uw menu **Options / Multi-Project Wizard**, or using a key short-cut **<Ctrl+M>**.

The Wizard allows following main features:

- building new Multi-Project file,
- loading existing Multi-Project file,
- running the multi-chip device operation, according to actual Multi-Project file.

The Multi-Project Wizard window contains following controls:



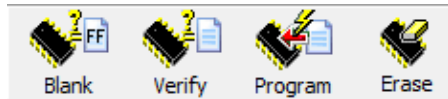
Button **Load Multi-Project file**
Loads existing Multi-Project file.



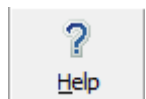
Button **Build Multi-Project file**
Builds new Multi-Project file from selected Project files.



Button **Device info**
Shows short device info (if available).



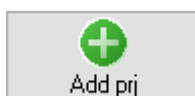
Buttons for device operations – **Blank, Verify, Program, Erase**
These buttons are used to run the selected multi-chip device operation according to Project files. Read operation is not supported in Multi-Project mode.



Button **Help**
Shows Help window with brief assistance on using Multi-Project Wizard.



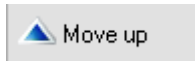
Button **Close**
Terminates the Multi-Project Wizard. After closing, the “unselected” device is automatically selected in Pg4uw software.



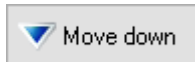
Button **Add project**
Adds existing Project file into selected Project files list, making it ready for later Multi-Project file building.

**Button Remove project**

Removes the Project file from selected Project files list.

**Button Move up**

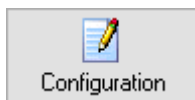
Moves the Project file one position up in selected Project files list. When running the Multi-Project file, individual Project files are processed in order according to selected Project files list.

**Button Move down**

Moves the Project file one position down in selected Project files list.

**Button Job Report**

Creates job report. For more information consult your programmer manual or **File / Make Job Report** paragraph in Pg4uw help.

**Button Configuration**

The button is displayed only if Multi-Project file is loaded and ready for operation. It switches the Wizard back to Multi-Project file building state.

1.3.1. BUILDING NEW MULTI-PROJECT FILE

Following steps are recommended when building new Multi-Project file:

1. Creating Project files

Project files are created in the same way as projects for devices in general:

- select Sub-device from Pg4uw device list,
- set device parameters, settings, and load required data into buffer using **Load file** command in Pg4uw,
- optionally perform test of device operation by running the device operation on real device,
- if everything is OK, the Project file can be created using **Save project** command in Pg4uw.

Creating the Project files for various tasks is in more details discussed in further chapters.

2. Launching Multi-Project Wizard

Multi-Project Wizard (see Figure 2) can be launched in two ways:

- selecting Master-device from Pg4uw device list (automatic launch), or
- using **<Ctrl+M>** shortcut or **Options / Multi-Project Wizard** menu.

3. Adding Project files

In Multi-Project Wizard, add required Project files using **Add project** button.

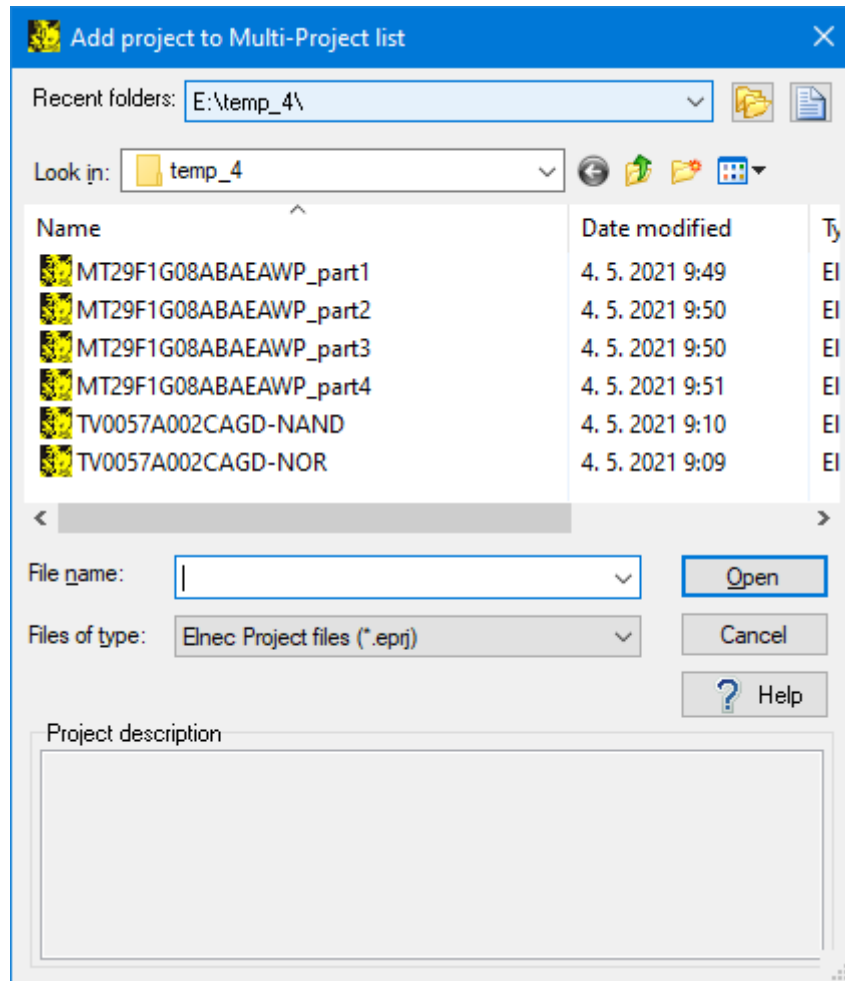


Figure 3: Add Project to Multi-Project list window.

4. Building final Multi-Project file

After completion of Project files selection, use **Build Multi-Project file** button to start the building process. A short warning window appears (see Figure 4), click **OK** to continue.

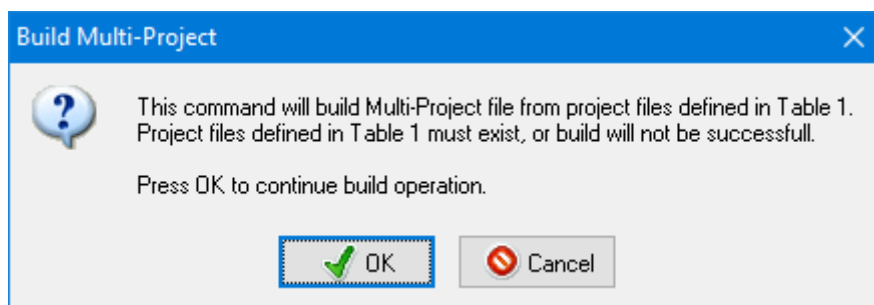


Figure 4: Build Multi-Project file warning window.

A **Build Multi-Project file As** window appears (see Figure 5), similar to the one for Save project command.

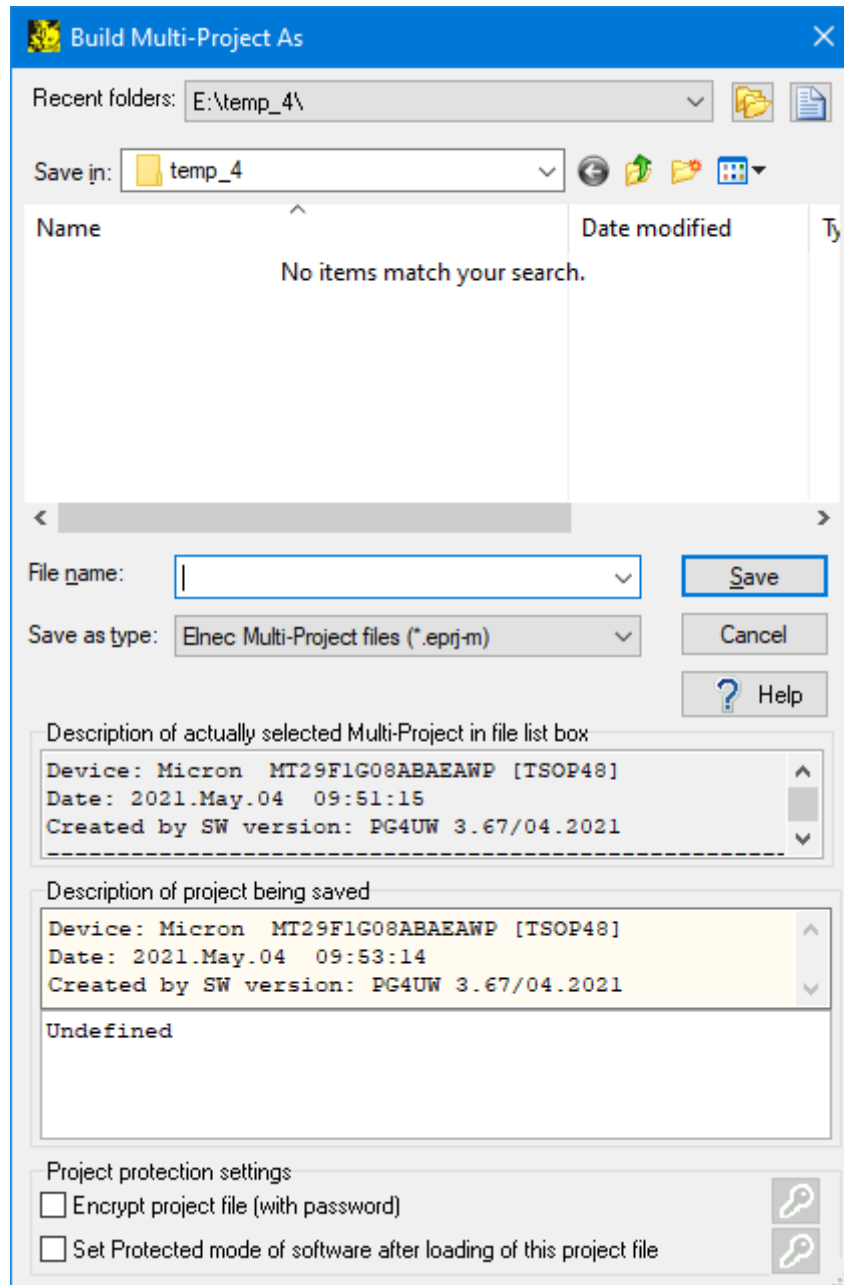


Figure 5: Build Multi-Project file As window.

Enter the Multi-Project file name and click Save button. The building process will start. During building, the individual Project files are compacted into single Multi-Project file. A small progress window is displayed on the top of Pg4uw windows stack (see Figure 6). After the building process is finished, the Multi-Project Wizard window is shown again, ready for running the multi-chip device operations (see Figure 7).

New Multi-project file carries all data specified in particular Project files. Those files are not further necessary for correct multi-chip device operation. Only Multi-Project file should be used / shared with other programmers.

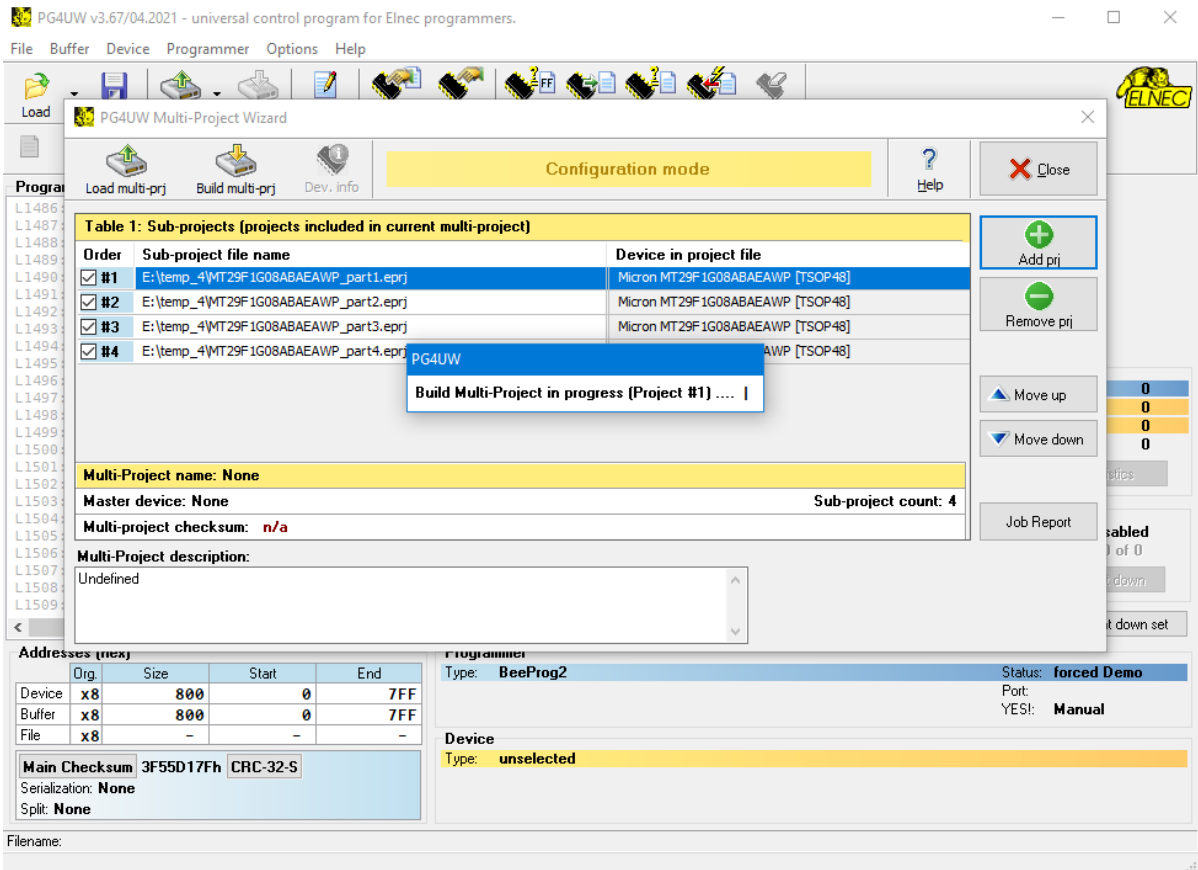


Figure 6: Pg4uw windows during building-up the Multi-Project file.

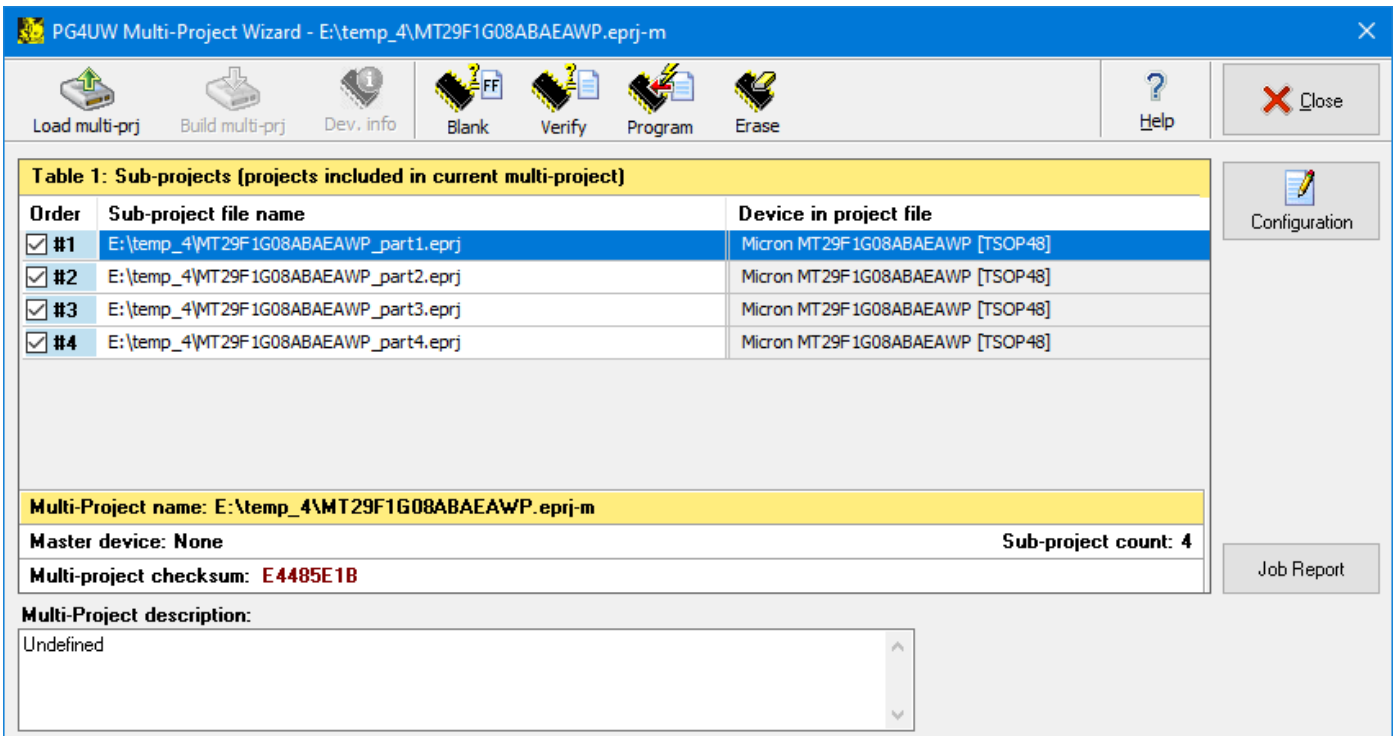


Figure 7: Multi-Project Wizard window with Multi-Project file loaded.

1.3.2. LOADING EXISTING MULTI-PROJECT FILE

Simply click the **Load Multi-Project** file button. A Load Multi-Project file window appears, as shown on Figure 8.

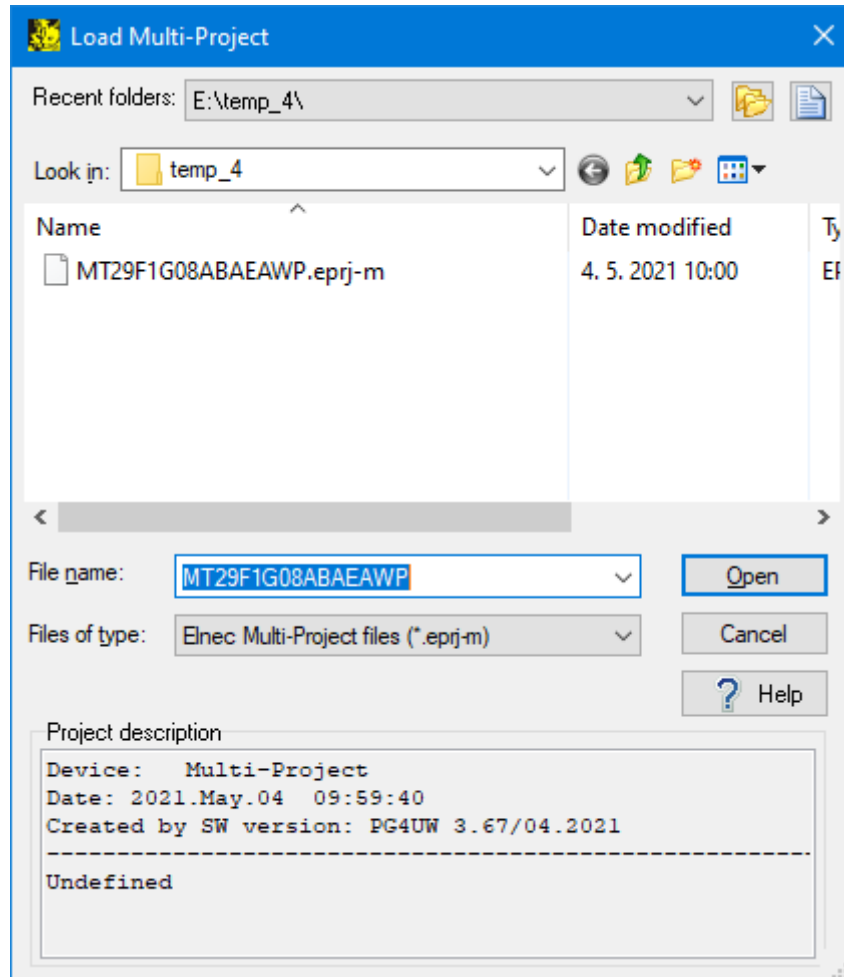


Figure 8: Load Multi-Project file window.

Select the required Multi-Project file and click **Open** button to load the file. During Multi-Project file loading, all included Project files are pre-loaded and internal Pg4uw environment is set up for running multi-chip device operation. During the loading, various messages can appear in Pg4uw Log window (see Figure 9). After loading completion, the Multi-Project Wizard window appears again (see Figure 7) and the programmer is ready to run the multi-chip device operation.

Alternatively, existing Multi-Project file can be loaded from Pg4uw software main menu using command **File / Load project**. In Load project window, select **Multi-Project files (*.eprj-m)** option in **Files of type** drop-down menu to view Multi-Project files instead of (single) Project files.

Then, select the desired Multi-Project file and click the **Open** button to load the file. A loading process is similar to that one using Multi-Project Wizard. After loading the Multi-Project file is finished, the Multi-Project Wizard window is shown (see Figure 7) and the programmer is ready to run the multi-chip device operation.

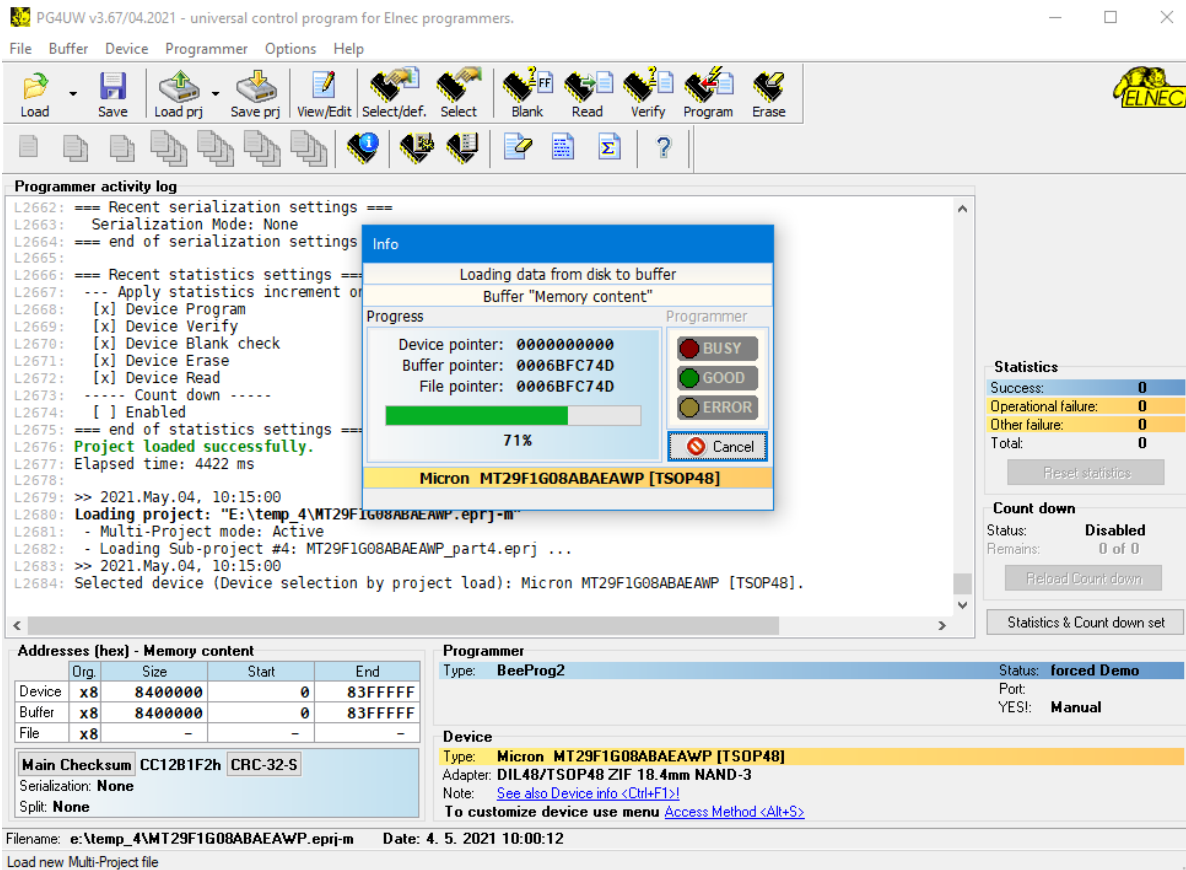


Figure 9: Pg4uw window during loading the Multi-Project file.

After loading the Multi-Project file, you can edit the Multi-Project settings, if necessary (after clicking the **Configuration** button). You can change the Project files order (determines the Project files execution order), add new Project files or remove existing Project files. After any change is made, the Multi-Project file must be built again to save performed changes. See previous pages for more information about building the Multi-Project file.

1.3.3. RUNNING THE MULTI-CHIP DEVICE OPERATION

Multi-Project feature is intended (but not restricted) to allow programming the multi-chip devices in single button click. Different techniques of using the Multi-Project feature for running various device programming tasks are explained in next chapters. However, the difference is concerned just to Multi-Project file preparation stage. After successful building the Multi-Project file, the procedure is always the same. The only difference on using the Multi-Project file for running the device operations is determined by programmer / control software used for performing the programming tasks.

1.3.3.1. SINGLE PROGRAMMING USING PG4UW

1. Load existing Multi-Project file using **File / Load project** menu command in Pg4uw main window or **Load Multi-Project file** button in the Multi-Project Wizard window. After the Multi-Project file is successfully loaded, the Multi-Project Wizard window is opened automatically.
2. In the Multi-Project Wizard window, run desired multi-chip device operation using one of available device operation buttons (Blank, Verify, Program, Erase). Mostly the programming operation is used. Selected device operation is executed as sequence of Project file loading and consequent Sub-device operation for each Sub-device specified in the Multi-Project file (see Figure 10 and Figure 11). This is the main purpose of Multi-Project feature – to automate the sequence of device operations for each chip in the multi-chip device.
3. After programming (verifying, ...) of all Sub-devices is finished (or error occurs), standard Repeat? window is displayed (see Figure 12). Programmed device can be removed from programmer socket and new device can be inserted. Pressing Yes button in Repeat? window or **YES!** button on programmer will start multi-chip device operation sequence again.

Note: If **Automatic YES!** feature is active, Repeat? window is not displayed after the completion of multi-chip device operation. Automatic YES! window is displayed instead. The window shows the status of programmer socket and notices to remove programmed device from programmer socket and insert the new one. After the new device is inserted, multi-chip device operation sequence will start automatically again. For more information on Automatic YES! feature consult your programmer manual or **Programmer / Automatic YES!** paragraph in Pg4uw help.

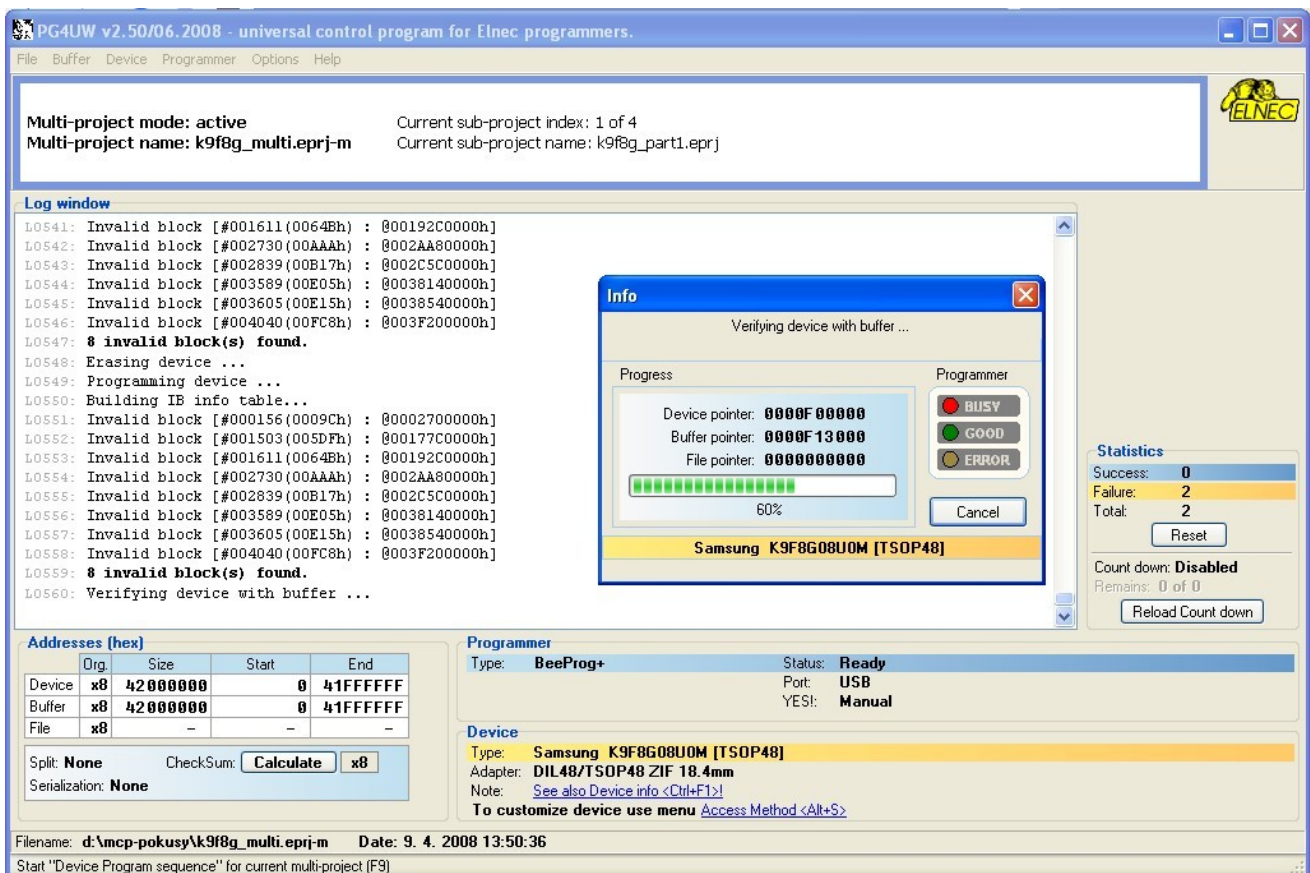


Figure 10: Processing the Multi-Project file – Project #1 of 4.



PG4UW v2.50/06.2008 - universal control program for Elnec programmers.

Multi-project mode: active Current sub-project index: 3 of 4
 Multi-project name: k9f8g_multi.eprj-m Current sub-project name: k9f8g_part3.eprj

Log window

```

L0618: Description: k9f8g - partition 3
L0619: -----
L0620: Project loaded successfully.
L0621: Elapsed time: 641 ms
L0622:
L0623: >> 20.06.2008, 09:37:11
L0624: Programming device: Samsung K9F8G08U0M [TSOP48].
L0625: Device insertion test ...
L0626: Checking device ID ...
L0627: Building IB info table...
L0628: Invalid block [#000156(0009Ch) : @0002700000h]
L0629: Invalid block [#001503(005DFh) : @00177C0000h]
L0630: Invalid block [#001611(0064Bh) : @00192C0000h]
L0631: Invalid block [#002730(00AAAh) : @002AA80000h]
L0632: Invalid block [#002839(00B17h) : @002C5C0000h]
L0633: Invalid block [#003589(00E05h) : @0038140000h]
L0634: Invalid block [#003605(00E15h) : @0038540000h]
L0635: Invalid block [#004040(00FC8h) : @003F200000h]
L0636: 8 invalid block(s) found.
L0637: Programming device ...
    
```

Info Programming device ...

Progress: Device pointer: 001FBCD000
 Buffer pointer: 0000815B80
 File pointer: 0000000000
 15%

Programmer: BUSHY GOOD ERROR

Samsung K9F8G08U0M [TSOP48]

Statistics
 Success: 0
 Failure: 2
 Total: 2
 Count down: Disabled
 Remains: 0 of 0

Addresses (hex)

Device	Org	Size	Start	End
x8	42000000		0	41FFFFFF
Buffer	x8	42000000		41FFFFFF
File	x8	-	-	-

Split: None CheckSum: Calculate x8
 Serialization: None

Programmer
 Type: BeeProg+ Status: Ready
 Port: USB
 YES!: Manual

Device
 Type: Samsung K9F8G08U0M [TSOP48]
 Adapter: DIL48/TSOP48 ZIF 18.4mm
 Note: See also Device info <Ctrl+F1>
 To customize device use menu Access Method <Alt+S>

Filename: d:\mcp-pokusy\k9f8g_multi.eprj-m Date: 9. 4. 2008 13:50:36
 Start "Device Program sequence" for current multi-project (F9)

Figure 11: Processing the Multi-Project file – Project #3 of 4.

PG4UW v2.50/06.2008 - universal control program for Elnec programmers.

Multi-project mode: active Current sub-project index: 4 of 4
 Multi-project name: k9f8g_multi.eprj-m Current sub-project name: k9f8g_part4.eprj

Log window

```

L0676: Invalid block [#002839(00B17h) : @002C5C0000h]
L0677: Invalid block [#003589(00E05h) : @0038140000h]
L0678: Invalid block [#003605(00E15h) : @0038540000h]
L0679: Invalid block [#004040(00FC8h) : @003F200000h]
L0680: 8 invalid block(s) found.
L0681: Programming device ...
L0682: Building IB info table...
L0683: Invalid block [#000156(0009Ch) : @0002700000h]
L0684: Invalid block [#001503(005DFh) : @00177C0000h]
L0685: Invalid block [#001611(0064Bh) : @00192C0000h]
L0686: Invalid block [#002730(00AAAh) : @002AA80000h]
L0687: Invalid block [#002839(00B17h) : @002C5C0000h]
L0688: Invalid block [#003589(00E05h) : @0038140000h]
L0689: Invalid block [#003605(00E15h) : @0038540000h]
L0690: Invalid block [#004040(00FC8h) : @003F200000h]
L0691: 8 invalid block(s) found.
L0692: Verifying device with buffer ...
L0693: Programming device - O.K.
L0694: Elapsed time: 0:00:01.1
L0695: Statistics info: Success:1 Failure:2
    
```

Info Programming device - O.K.

Progress: Device pointer: 003E840000
 Buffer pointer: 0000041000
 File pointer: 0000000000
 100%

Programmer: BUSHY GOOD ERROR

Samsung K9F8G08U0M [TSOP48]

Statistics
 Success: 1
 Failure: 2
 Total: 3
 Count down: Disabled
 Remains: 0 of 0

Repeat?
 Press "YES!" button or "Y" key to repeat last activity or press <Esc> key to exit
 Yes No >>

Addresses (hex)

Device	Org	Size	Start	End
x8	42000000		0	41FFFFFF
Buffer	x8	42000000		41FFFFFF
File	x8	-	-	-

Split: None CheckSum: Calculate x8
 Serialization: None

Device
 Type: Samsung K9F8G08U0M [TSOP48]
 Adapter: DIL48/TSOP48 ZIF 18.4mm
 Note: See also Device info <Ctrl+F1>
 To customize device use menu Access Method <Alt+S>

Filename: d:\mcp-pokusy\k9f8g_multi.eprj-m Date: 9. 4. 2008 13:50:36
 Start "Device Program sequence" for current multi-project (F9)

Figure 12: Processing the Multi-Project file – successful completion with Repeat? window.

1.3.3.2. MULTI-PROGRAMMING USING Pg4uWMC

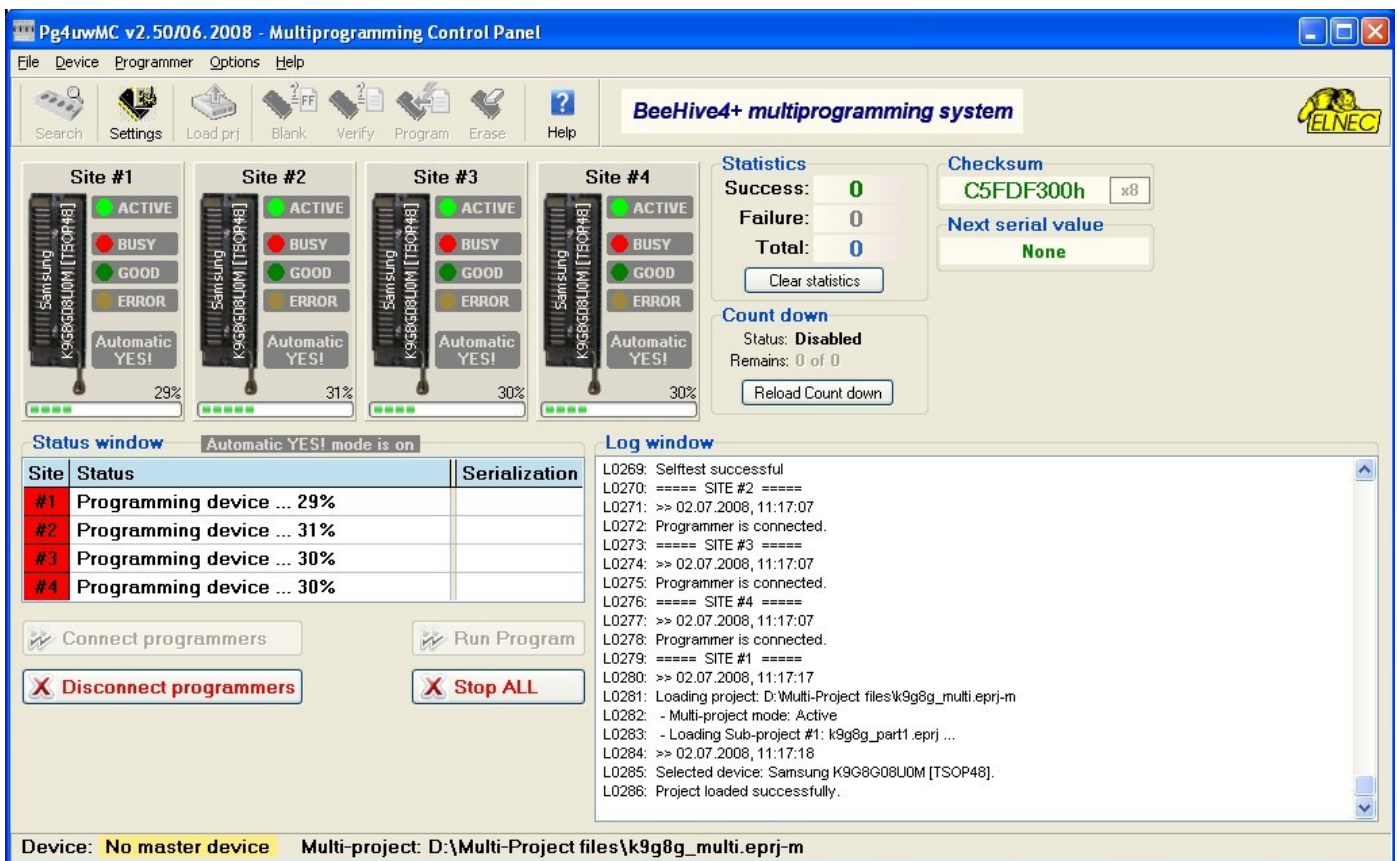
1. Load existing Multi-Project file using **Load project** menu.
2. Run desired multi-chip device operation using one of available device operation buttons (Blank, Verify, Program, Erase). Mostly the programming operation is used.

Selected device operation is executed as sequence of Project file loading and consequent Sub-device operation for each Sub-device specified in the Multi-Project file (see Figure 13).

This is the main purpose of Multi-Project feature – to automate the sequence of device operations for each chip in multi-chip device.

3. After programming (verifying, ...) of all Sub-devices is finished (or error occurs), information with result of device operation is displayed in Pg4uWMC. Programmed device can be removed from programmer socket and new device can be inserted. Pressing the operation button for the Site or **YES!** Button on the Site will start multi-chip device operation sequence again.

Note: If **Automatic YES!** feature is active, the multi-chip device operation sequence is automatically started again after removing the programmed device from programmer socket and inserting the new one. For more information on Automatic YES! feature consult your programmer manual or **Programmer / Automatic YES!** paragraph in Pg4uWMC help.



Pg4uWMC v2.50/06.2008 - Multiprogramming Control Panel

File Device Programmer Options Help

Search Settings Load prj Blank Verify Program Erase Help

BeeHive4+ multiprogramming system

Site #1: ACTIVE, 29%
 Site #2: ACTIVE, 31%
 Site #3: ACTIVE, 30%
 Site #4: ACTIVE, 30%

Statistics: Success: 0, Failure: 0, Total: 0
 Checksum: C5FDF300h x8
 Next serial value: None

Count down: Status: Disabled, Remains: 0 of 0

Site	Status	Serialization
#1	Programming device ... 29%	
#2	Programming device ... 31%	
#3	Programming device ... 30%	
#4	Programming device ... 30%	

Log window:

```

L0269: Selftest successful
L0270: ===== SITE #2 =====
L0271: >> 02.07.2008, 11:17:07
L0272: Programmer is connected.
L0273: ===== SITE #3 =====
L0274: >> 02.07.2008, 11:17:07
L0275: Programmer is connected.
L0276: ===== SITE #4 =====
L0277: >> 02.07.2008, 11:17:07
L0278: Programmer is connected.
L0279: ===== SITE #1 =====
L0280: >> 02.07.2008, 11:17:17
L0281: Loading project: D:\Multi-Project files\k9g8g_multi.eprj-m
L0282: - Multi-project mode: Active
L0283: - Loading Sub-project #1: k9g8g_part1.eprj ...
L0284: >> 02.07.2008, 11:17:18
L0285: Selected device: Samsung K9G8G08U0M [TSOP48].
L0286: Project loaded successfully.
  
```

Device: No master device Multi-project: D:\Multi-Project files\k9g8g_multi.eprj-m

Figure 13: Processing the Multi-Project file in Pg4uWMC.



1.4. LIMITATIONS OF MULTI-PROJECT FEATURE

- serialization is not supported in multi-programming mode (only single programming supports serialization),
- count-down function is not supported,
- maximum 16 Project files in single Multi-Project file are supported.

We are working on improving of the Multi-Project feature, so missing features may be supported in the future. Please, check for the newest version of control software.

2. USING THE MULTI-PROJECT FEATURE FOR VARIOUS TYPES OF PRACTICAL TASKS

Multi-Project feature is very versatile tool. Its primary designation is to simplify the processing of multi-chip devices. However, it can be used for various other kinds of complex programming tasks. The following chapters will provide more details on this topic.

2.1. USING MULTI-PROJECT FEATURE FOR PROGRAMMING MULTI-CHIP DEVICES

Imagine you have a multi-chip device that contains several memories. For example – NAND flash, NOR flash and PSRAM memories in single package. Generally, RAM memories are out of the scope, since they cannot be programmed permanently. So there are two independent chips remaining that need to be programmed – NAND flash and NOR flash. Having Multi-Project feature available, you can program both memories on single button click. Read further on how to proceed.

Note: We will use TV0057A002CAGD [FBGA107] from Toshiba in following example.

Step 1 – Defining the Project file for NAND part

1. Firstly, display the Select device window and list for the relevant Sub-device, i.e. for TV0057A002CAGD [FBGA107] (NAND) in this example (see Figure 14).
2. Select the device and configure all settings (Device operation option - <Alt+O> as well as Access Method - <Alt+S>). For more information about programming NAND flash memories using Elnec device programmers consult our application note freely available from our web-site https://www.elnec.com/sw/an_programming_nand_flash_using_elnec_programmers.pdf
3. Load the data file(s) into buffer.
4. Save the Project file for NAND flash part.
5. Test your Project file.

Step 2 – Defining the project for NOR part

1. Then, display Select device window again and list for other Sub-device, i.e. TV0057A002CAGD [FBGA107] (NOR) in this example (see Figure 15).
2. Select the device and configure all settings (Device operation options - <Alt+O> as well as Device settings - <Alt+S>).

3. Load the data file(s) into buffer.
4. Save the Project file for NOR flash part.
5. Test your Project file.

This way configure the Project files for all Sub-devices included in desired multi-chip device. After doing so you can finally build the Multi-Project file.

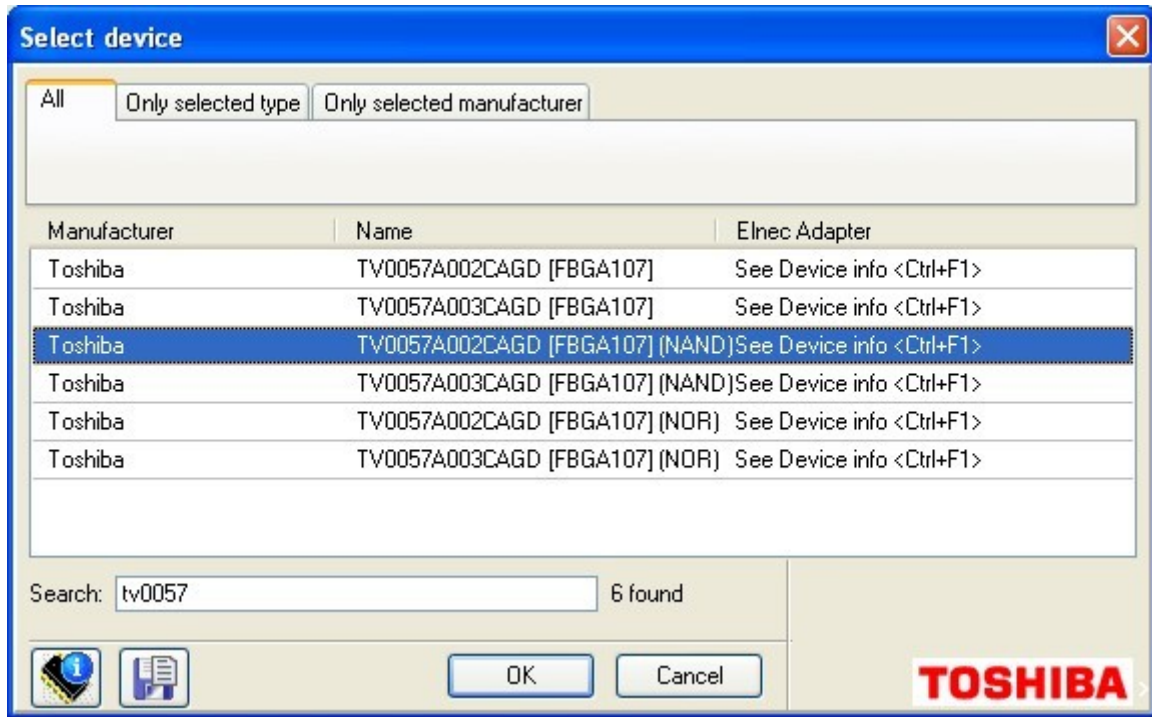


Figure 14: Selecting NAND part of multi-chip device.

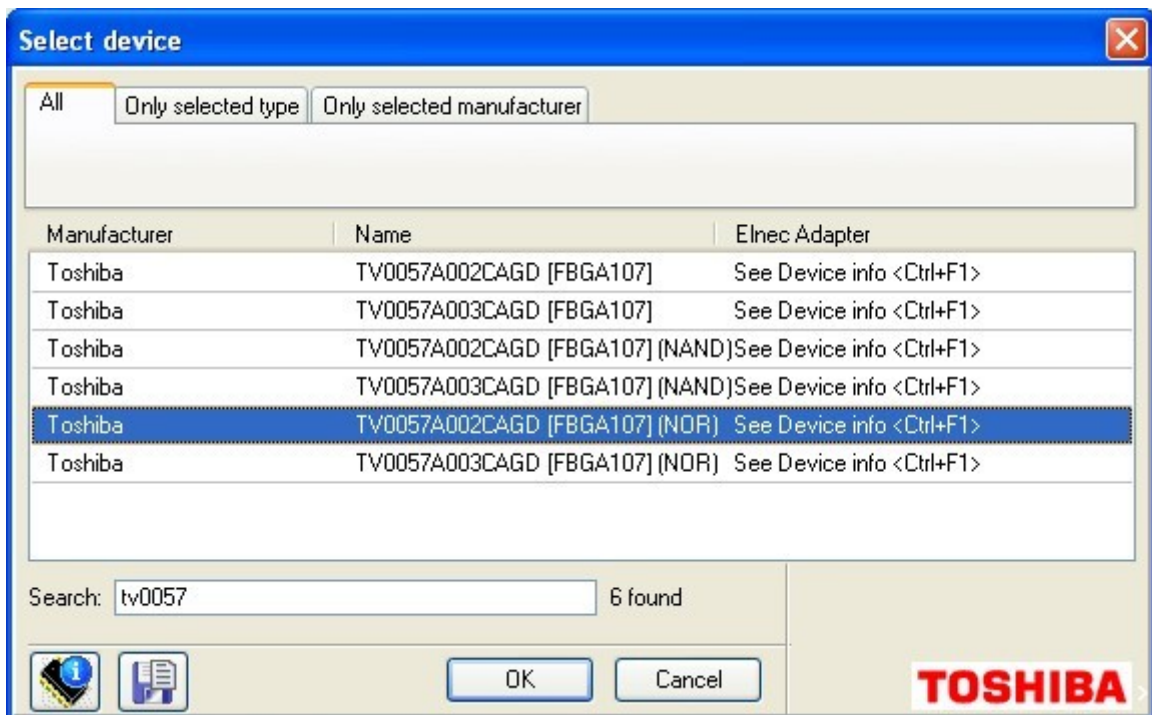


Figure 15: Selecting NOR part of multi-chip device.

Step 3 – Building the Multi-Project file

1. Having Project files available for all Sub-devices, you can finally build the Multi-Project file. Display Select device window again and select the Master-Device, i.e. TV0057A002CAGD [FBGA107] in this example (see Figure 16). Empty Multi-Project Wizard window displays, see Figure 2.
2. Add Project files saved in previous steps and build the Multi-Project file. The procedure was described in chapter **Building new Multi-Project file**.
3. Test your Multi-Project file.

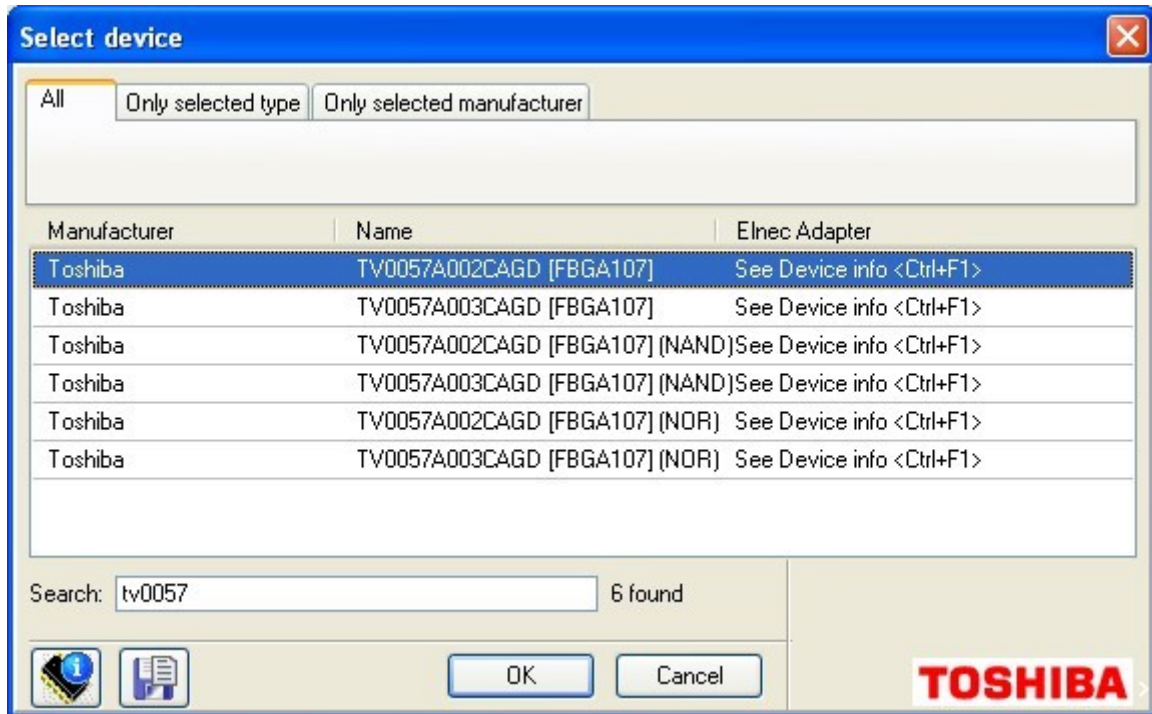


Figure 16: Selecting Master-device.

Known limitations:

All general limitations are effective in this mode.

2.2. USING MULTI-PROJECT FEATURE FOR PROGRAMMING MULTIPLY PARTITIONS INTO SINGLE NAND FLASH

Imagine your project need to arrange various data into single NAND flash memory. For example – boot record image, operating system image and file system image. Each image need to be placed at exactly specified address / block of NAND flash memory. It is not such easy to guarantee the exact block mapping in NAND flash device, due to a possibility of invalid block(s) occurrence. Having Multi-Project feature available, your images can start at exact block on single button click. Read further on how to proceed.

Notes:

This Multi-Project feature operation mode is primarily designated (but not limited to) for programming of multiply partitions into single NAND flash device.

If you need to program multiply partitions into NAND flash chip that stands for a Sub-device in multi-chip device, you can use this technique to save the Project files for NAND flash part. Then you can use common multi-chip approach described in previous chapter – just use multiply Project files for NAND part saved here instead of single Project file defined in previous chapter.

There are plenty of NAND flash devices in our support today. We will use K9F1208U0C [TSOP48] from Samsung in this example. The device has 512 Mbit / 64 Mbyte capacity and consists of 4096 blocks.

Step 1 – Defining the Project files for individual data images

1. Firstly, display the Select device window and list for desired NAND flash device, i.e. for K9F1208U0C [TSOP48] in this example. Select the device.
2. Display Access Method window, key short-cut <Alt+S>. Set appropriate Invalid Block Management method, Spare Area Usage method, User Area – Start Block (equivalent to partition start), User Area – Number of Blocks (equivalent to used partition size), User Area – Last Block (equivalent to partition end) and other options as necessary for the first data image. For more information about programming NAND flash memories using Elnec device programmers consult our application note available at https://www.elnec.com/sw/an_programming_nand_flash_using_elnec_programmers.pdf.
3. Display Device operation options window, key short-cut <Alt+O>. Select the proper settings.
4. Load the data file(s) into buffer.
5. Save the Project file for the first data image / partition.

This way prepare the Project files for all your data images. Once all Project files are saved you can build the Multi-Project file.

Step 2 – Building the Multi-Project file

1. Having Project files available for all data images, you can finally build the Multi-Project file. Open the Multi-Project Wizard window (see Figure 2) using **Options / Multi-Project Wizard** Pg4uw menu command or key short-cut <Ctrl-M>, add individual Project files and build the Multi-Project file, as described in chapter **Building new Multi-Project file**.
2. Test your Multi-Project file.

If you need to program multiply partitions into NAND flash chip that stands for a Sub-device in multi-chip device, use individual Project files saved in Step 1 together with Project file for NOR part for building the final Multi-Project file. The Multi-Project file built here in Step 2 can be used for processing only the NAND part of multi-chip device.

Known limitations:

All general limitations are effective in this mode.

Special notes:

- If Spare Area Usage setting is set to User Data for certain partition, a plenty of invalid blocks may be found by programmer when processing all subsequent partitions. The only effect will be a long listing in control software log window. This will not affect the quality of the programming process for subsequent partitions, since invalid blocks before User Area – Start Blocks are ignored, anyway. None workaround is necessary.
- Each time the **Erase before programming** option is enabled, a whole NAND flash device will be erased. This way data programmed in preceding partitions will be destroyed.
Workaround: Use this option carefully. Make sure that you have enabled **Erase before programming** only for the first Project file in a sequence (if any).

2.3. USING MULTI-PROJECT FEATURE FOR PROGRAMMING MULTIPLY DAISY-CHAINED JTAG DEVICES

Imagine you have a data acquisition board that incorporates several (e.g. 8) low-cost microcontrollers, that act as intelligent A/D converters. If these microcontrollers are provided with JTAG interface, you can connect them into single chain. Probably, you have already did so because of debug purposes. Having Multi-Project feature available, you can program all your chained microcontrollers on single button click. Read further on how to proceed.

Notes:

Please, test the feature with your board before you decide to start mass-production. This mode was not tested in a responsible manner due to a lack of suitable board. The maximum supported chain length is 256 bits. If you observe any problems using Multi-Project feature for programming multiply daisy-chained JTAG devices, please, contact your distributor. Sending your board to our lab may be necessary.

Although all microcontrollers used in following example are the same and use the same data file, this is not a limitation. From Multi-Project feature point of view, it is possible to combine any JTAG devices and an individual data image can be used for each device.

Actually, there are several ARM based microcontroller families in our support, that can be used as target device if connected in JTAG daisy chain. They are differentiable by part name extension of (ISP-JTAG CHAIN). We will use STM32F103RB (ISP-JTAG CHAIN) in this example.

Step 1 – Defining the Project files for individual devices

1. Firstly, display the Select device window and list for desired JTAG device supported in ISP-JTAG CHAIN mode, i.e. for STM32F103RB (ISP-JTAG CHAIN) in this example.
2. Display Device access configuration window, key short-cut <Alt+S>. Set desired device configuration options.
3. Display Device operation options window, key shortcut <Alt+O> (see Figure 17). The window consists of several sections of the settings, that you probably are familiarized with from your previous experience with in-circuit programming using Elnec device programmers. The sections Insertion test, Command execution, Target system power supply parameters and Target system parameters are out of the scope of this application note. If you need more information about those, consult your programmer manual or Pg4uw Help.

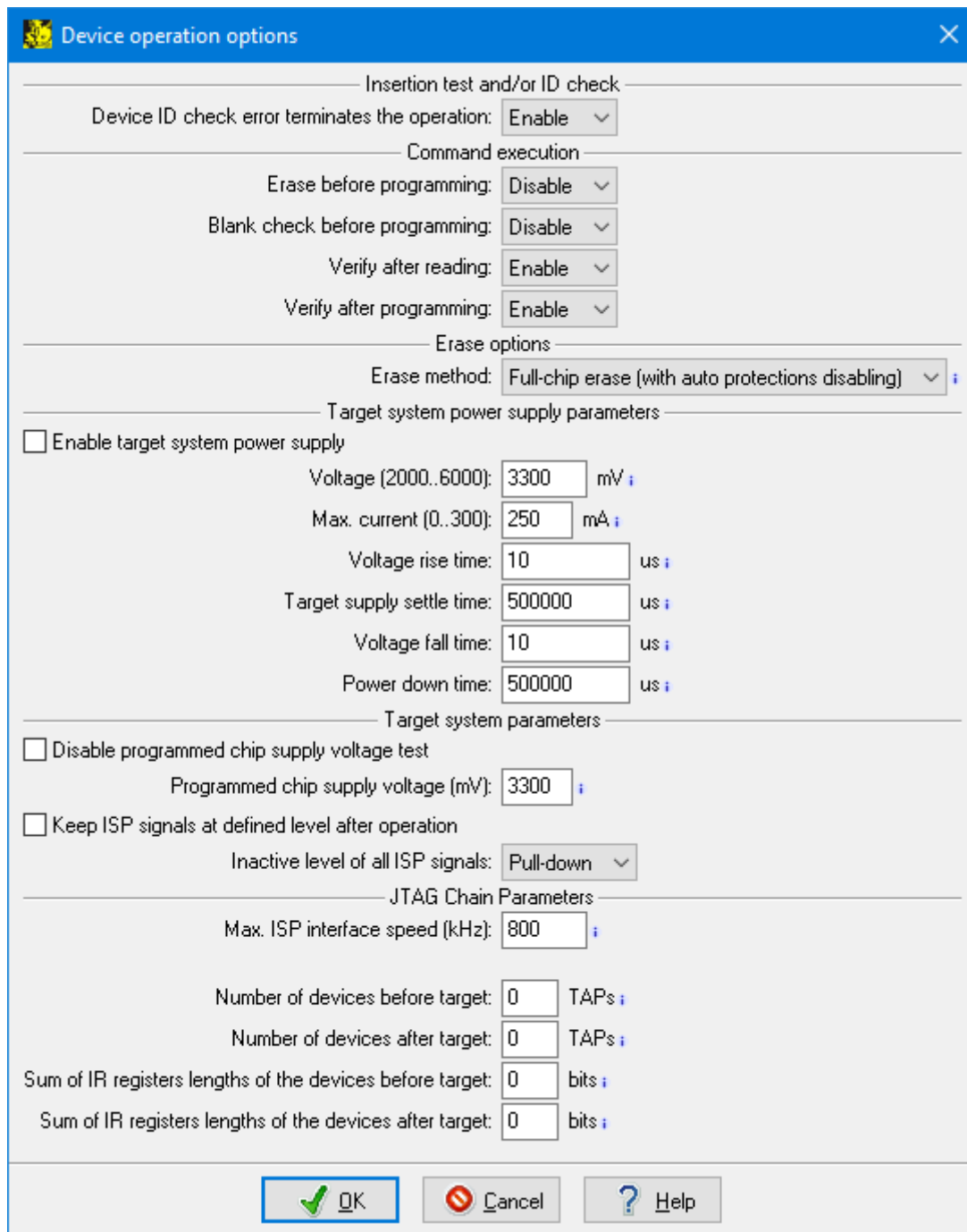


Figure 17: Device operation options for device supported in ISP-JTAG CHAIN mode.

- The section JTAG Chain Parameters is critical for this mode of operation. There are several settings available:

Max. ISP interface speed (kHz) – determines maximum clock frequency on JTCK pin allowed for use. It must comply with the slowest device in the chain. Consult the datasheets of connected devices for maximum allowed JTAG speed.

Number of devices before target – see Figure 18 for definition of the target device (STM32 in this example) and devices before the target device (devices between the programmer and the target device JTDI pin) and after the target device (devices between the target device JTDO pin and the programmer). Specify the number of devices before the target device in this field.

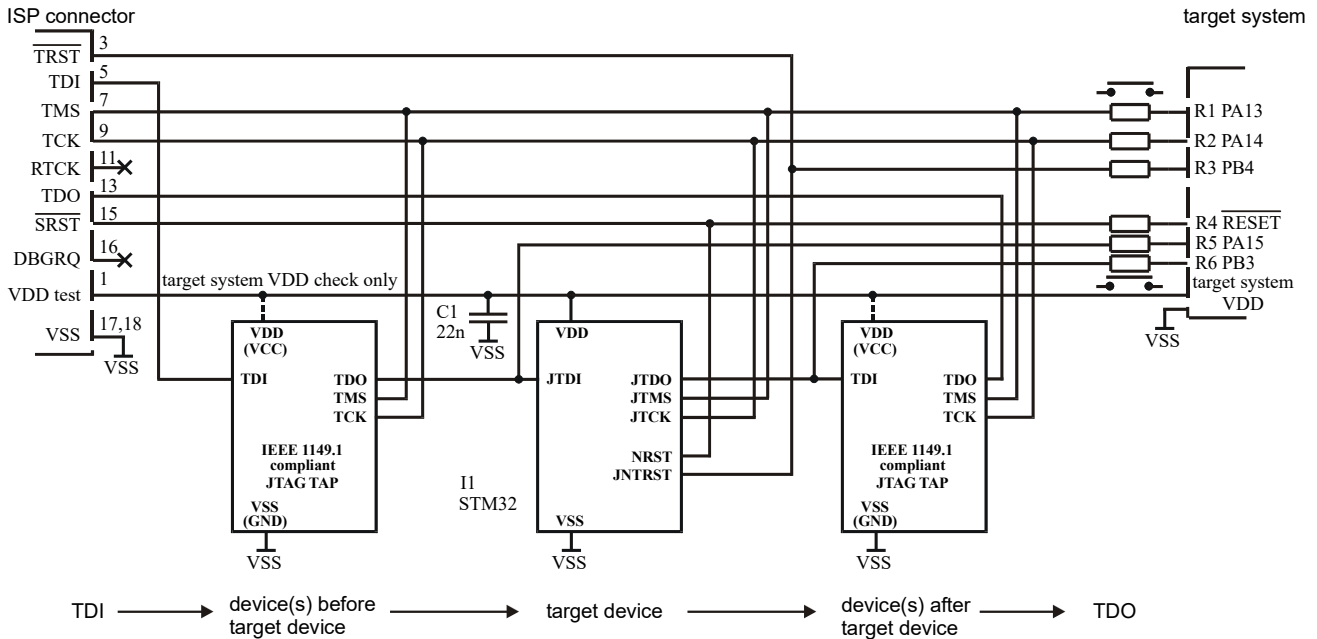


Figure 18: Example of JTAG chain connection.

Number of devices after target – similarly to previous field, specify the number of devices after the target device here.

Sum of IR registers lengths of the devices before target – each JTAG device has at least three registers – instruction register (IR), data register (DR) and bypass register (BYPASS). Typically, if device is not the target one, bypass instruction is written into IR. In consequence, the BYPASS register is connected to JTAG chain instead of DR one. The length of BYPASS register is known and compulsory – 1 bit. Also the bypass instruction is exactly defined – all ones (1). This way all devices in chain other than the target one are simply switchable to bypass mode, so the programmer affects only the target device. Consult the datasheets of connected devices for IR lengths. Calculate the sum of the IR lengths for devices that are connected before the target one and specify it in this field.

Sum of IR registers lengths of the devices after target – similarly to previous field, specify the sum of the IR lengths for devices that are connected after the target one here.

Note:

Some devices consist of several independent units that are chained internally – e.g. CPU core, flash memory, trace unit, etc. Although they are in single package, consider them being independent devices before / after the target device, having their own instruction registers, and reflect them in the chain settings. On the contrary, the target device should be considered as single device, internal units should not be included into the sums calculations.

- Having all settings properly set, load the data image(s) into buffer and save the Project file for this device.

This way configure and save the Project files for all desired devices in the chain.



Step 2 – Building the Multi-Project file

1. Having Project files available for all desired devices connected in the chain you can finally build the Multi-Project file. Open the Multi-Project Wizard window (see Figure 2) using **Options / Multi-Project Wizard** Pg4uw menu command or key short-cut **<Ctrl+M>**, add individual Project files and build the Multi-Project file, as described in chapter **Building new Multi-Project file**.
2. Test your Multi-Project file.

Known limitations:

All general limitations are effective in this mode.



VERSIONS HISTORY

Version 1.00 – July 2008

- initial release

Version 1.01 – June 2010

- User Area - Last Block option added
- minor Building new Multi-Project file chapter modification / clarification

Version 1.2 – June 2021

- figures updated
- duplicate and / or illustrative only figures removed
- some text clarifications
- document formatting changed
- disclaimer added