DediProg User Manual

12/2017

NuProg-E

Engineering UFS/EMMC Programmer User Manual

Version 1.4



©DediProg Technology Co., Ltd 2017 All rights reserved.



Table of Contents

| ۱. | INTROD | UCTION | . 4 |
|------|-----------|--|-----|
| II. | PRODU | CT INFORMATION | . 4 |
| III. | SYSTEM | REQUIREMENT | . 5 |
| IV. | PRODU | CT DESCRIPTIONS | .6 |
| 4. | 1 Exterio | R | . 6 |
| 4. | 2 Install | Socket Adaptor | . 8 |
| v. | DEDIW | ARE QUICK INSTALLATION | .9 |
| 5. | 1 Softwa | RE INSTALLATION | . 9 |
| 5. | 2 Install | NUPROG-E PROGRAMMER | 12 |
| 5. | 3 NUPROG | 5_UFS INSTALLATION GUIDE. | 12 |
| | 5.3.1 Nı | IProg_UFS icon | 12 |
| | 5.3.2 So | ftware Introduction (UFS Part) | 13 |
| | А. | Main Menu and Functions | 13 |
| | В. | LOG Window | 27 |
| | С. | Programmer Information | 27 |
| | D. | IC Information | 28 |
| | Ε. | CheckSum Data | 28 |
| | <i>F.</i> | Batch Config Setting | 28 |
| | 5.3.3 Ex | amples | 29 |
| 5. | 4 Precau | TIONS WHEN USING NUPROG-E (FOR UFS) SOFTWARE | 37 |
| 5. | 5 NuProc | 6_eMMC Installation Guide | 38 |
| | 5.5.1 Nı | IProg_eMMC | 38 |
| | 5.5.2 | Software Introduction (eMMC part) | 38 |
| | А. | Main Menu and Functions | 39 |
| | В. | LOG Window | 50 |
| | С. | Programmer Information | 50 |
| | D. | IC Information | 50 |
| | Ε. | CheckSum Data | 51 |
| | F. | Batch Setting | 51 |
| VI. | REVISED | DEDITION | 52 |



Important notice:

This document is provided as a guideline and must not be disclosed without consent of DediProg. However, no responsibility is assumed for errors that might appear. DediProg reserves the right to make any changes to the product and/or the specification at any time without notice. No part of this document may be copied or reproduced in any form or by any means without prior written consent of DediProg.



I. Introduction

This user manual is mainly introducing the hardware specifications, applications and the software quick installation. NuProg-E is an engineering programmer that is innovated for programming mass flash storage, such as UFS and eMMC. For UFS, it provides the basic LUN and RPMB read and write as well as the settings for descriptors, attributes and flags. As for eMMC, it supports User Area, Boot 1/2 and RPMB basic read and write; also supports ExtCSD, GPP1~4 partition, read and write, as well as enhanced mode setting. With the high speed USB 3.0, the NuProg-E is the finest programmer for UFS and eMMC development. For more information, please visit our website.

www.dediprog.com/download

II. Product Information

High Speed Programming

With high speed processor, programming speed will increase greatly. UFS Write speeds: 20~50MB/s; Read speeds: 50~80MB/s (Depends on computer and IC performance).

Time reference: It will only take 25 seconds to program a 1GB file into a Toshiba UFS IC

Support UFS and EMMC/EMCP Settings and Programs

UFS part

- 1. Support Descriptors, Attributes, Flags settings
- 2. Support LUN configured and advance settings
- 3. RPMB

EMMC part

- 1. Support User Area, Boot1/2 Partition and Extend CSD
- 2. Support RPMB, GPP1~4 and Enhanced mode

Support all IC package

Support standard package of UFS and EMMC (BGA153 (11.5x13mm)) or special package of EMCP.

• Regular Software Update

Palm Size and Space Saving

Dimension: 132 x 75 x 30 mm Weight: 104g

Support USB 2.0 and USB 3.0 (Power adapter is required)



III. System Requirement

| CPU: | Intel i5 or Above |
|------------------|--|
| OS: | Windows 7 / 8 / 8.1 / 10 |
| USB Port: | USB 2.0 and USB 3.0 |
| Free Dish Space: | At least twice of the programming memory. |
| CD ROM: | It is necessary for installing the software. |

*Since UFS and EMMC have mass volumes, please reserve enough space for buffering. *Computer performances will affect the read and write of UFS, please choose a computer that has higher CPU and better performance.



IV. Product Descriptions

4.1 Exterior









A. Power Signal Light

The light indicates the programmer is powered on.

B. Operation Lights

Red LED (Error): Error; programming has failed.Yellow LED (Busy): The programmer is operating.Green LED (Pass): Passed; the programming has completed successfully.

C. Programming Socket Sites

Built-in high speed connector, which is for installing the socket adaptor

D. Start Button

Not available now.

E. Power Connector

External power inputs (*In order to make it more stable, please make sure the power adapter is connected.*)

F. USB 3.0 Port

For connecting programmer with the computer.



4.2 Install Socket Adaptor

Place an IC into the socket adaptor and attach the adaptor to the socket site.







V. Dediware Quick Installation

The software is provided with the purchase of NuProg-E programmer. The newest version will also be available on our website. **www.dediprog.com**

5.1 Software Installation

5.1.1 Install DediWare



Fig. 5-1



5.1.2 When you install NuProg-E software for the first time, please install the USB Driver. Otherwise, the computer will not be able to recognize the programmer.



Fig. 5-2



Fig. 5-3



5.1.3 After installation, **Dediware, NuProg_UFS** and **NuProg_eMMC** icons will appear on the desktop. The Dediware icon is for StarProg and ProgMaster series programmer while the NuProg_UFS and NuProg_eMMC are for NuProg-E programming.



There is another icon called DediWare_CLI; it is the Command Line software for StarProg Series programmers, so it does not support NuProg series.



Fig. 5-5



5.2 Install NuProg-E Programmer

5.2.1 Place an IC into the socket adaptor and attach it to the socket site.

5.2.2 Connect NuProg-E to the computer (USB 3.0 is recommended).

5.2.3 Once you open Dediware, it will detect a programmer automatically and will be available for programming.

*Using multiple USBs on the same computer may affect the charging currents, so it is **REQUIRED** to connect the USB with our *power adapter* for stable programming.

5.3 NuProg_UFS Installation Guide.

5.3.1 Double Click NuProg_UFS icon.



Fig. 5-6





5.3.2 Software Introduction (UFS Part)

Fig. 5-7

A. Main Menu and Functions



Fig. 5-8



A-1. Main Menu- Advance





- Language: English, Simplified Chinese and Traditional Chinese are provided.
- Log in: Set up the IP address for remote controls.
- General options: Set up a temporary file for saving buffers. Since large volume IC has a great demand of storage. Therefore, if C drive does not have enough space, please choose other drive.

| General options | | × |
|-----------------------------------|--------|---|
| Custom path for buffer file | | |
| C:\Users\user\AppData\Local\Temp\ | | |
| | | |
| | | |
| | | |
| | | |
| ОК | Cancel | |

Fig. 5-10



A-2. Main Menu- Help



Fig. 5-11

- **Firmware Manual Update:** Update the firmware version of the NuProg-E. Update the firmware and restart the programmer.
- Launch Calculator: It opens the calculator.
- User Manual: It links to DediProg's user manuals.



A-3. Functions (From Top to Bottom; Left to Right)





- Detect: Detect IC or choose the model number manually. If the data list has the corresponding model number that is supported, software will automatically import the values of the UFS, and the Log will appear as below (Fig. 5-13). If the IC model number that you need are not listed (Fig. 5-14), please feel free to contact us.
 - 06:44:38:Type THGLF2G8D4KBADR is applied.
 - (i) 06:44:46:Detecting chip...
 - 06:44:46:Found chip THGLF2G8D4KBADRby auto detecting
 - 06:44:47:Set operation:none
 - (i) 06:44:47:OptionBytes CheckSum :0x0
 - (i) 06:44:47:Chip CheckSum :0x0
 - ♦ 06:44:47:Select ic(UFS:THGLF2G8D4KBADR[TOSHIBA]) success.
 - 06:44:47:Type THGLF2G8D4KBADR is applied.

Fig. 5-13

| g Nu | Prog-E Engineering UFS/EMMC Programmer User M |
|---|---|
| Select Chip | × |
| Chip Type All Manufacture All KLUBG4G IBD-E0B 1[BGA095]-Samsung KLUBG40 ICE-80B 1[TFBGA153]-Samsung KLUDG831CB-80B 1[TFBGA153]-Samsung THGBF7G8K4LBATR8[TFBGA153]-TOSHIBA THGBF7G9L4LBATR[TFBGA153]-TOSHIBA THGBF7CBR4BATAB[TFBGA153]-TOSHIBA THGBF7CBR4BATAB[TFBGA153]-TOSHIBA | ✓ ✓ |
| THGLF2G8D4KBADR[TFBGA153]-TOSHIBA THGLF2G8J4LBATR[TFBGA153]-TOSHIBA | LUN0size: 30508 MB LUN1size: 4 MB LUN2size: 4 MB (i) 10:51:04:Type KLUBG4G1CE-B0B1 is applied. (i) 10:51:15:Detecting chip |
| | Δ 10:51:17:initialize fail Δ 10:51:17:Device unrecognizable or not found |

Fig. 5-14

> **Load:** Import the programming file and set the values according to your requirements.

| | | | | Load file | | | | × |
|--------------------------------|--|--|------------------------------|-------------------------------------|---------------|---|---|------------|
| File 1 | + | | | | | | | |
| FilePat File For FileChe | h: C:\U rmat: Binar eckSum: Byte | sers\user\Desktop\ ry(*.bin) v Acc v | 1420035668 UB FileOffset: | CD31_HWReset_eM 0x00 nk Value | IMC.t ♥ | Partition Name: SectorIndex: AutoSe SectorCount: | LUN0 0 et FileOffset:0x0 32768 | 0 |
| Hide Image | 25 | | | | | | ОК | Cancel |
| # | PartitionName | SectorIndex | FileOffset | SectorCount | FileFormat | CheckSumAlg | SkipBlank | FilePath |
| Image 01 | LUNO | 0 | 0x00 | 32768 | Binary(*.bin) | ByteAcc | NotSkip | C:\Users\u |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |





Value Descriptions:

- File Format: The format of the programming file.
- **File Checksum:** The file checksum's calculation method.
- File Offset: Appoint an address to start loading the buffer.
- File Path: Assign a path for the programming file.
- **Skip Blank Value:** Enable this function to analyze and skip the blank data before programming, which will write more efficiently and reduce the programming time.
- Partition Name: It depends on the partitioned sections.
 Note: This partition here will only show the amounts and settings according the

IC that has been detected.

- Sector Index: Set up the sector starting point
- Sector Count: Set up total programming sectors.
- **AutoSetFileOffset**: When setting the **Sector Index**, this function can automatically calculate the File Offset.
- +: When there is more than one programming file, please press "+" to add the files to the data list.
- Show Images and Hide Images: Open and close the data list.
- > Load Prj: Import previous programming files (Not Supported Yet).
- Save Prj: Save all the settings as a programming file (Not supported yet).



Buffer: Temporary files.

Select a partition to display the LUN and the related Option values of the UFS.

| LUN0 [Type:Flash] | | | | | | | | | | | | | | | | | \times |
|--|--------|------|------------|------------|------------|------|------------|------------|----|-------------|----|----|------------|------------|------------|----|----------|
| ⊖ Flag ⊖ | Attrib | oute | | C | De | scri | ptor | | OF | ₹ PM | в | | ۲ |) LUI | NO | | |
| O LUN1 O | LUN | 2 | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | |
| Address | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C | +D | +E | +F | - 1 |
| 0x000000000000000000 | 37 | 7A | BC | AF | 27 | 1C | 00 | 03 | 62 | 78 | 77 | 02 | CF | 23 | 16 | 00 | |
| 0x000000000000000000 | 00 | 00 | 00 | 00 | 75 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 13 | D2 | 0D | A3 | _ |
| 0x0000000000000000020 | 00 | 1E | 1A | 0 A | 86 | EF | 4A | D6 | 8E | 91 | F5 | EF | F5 | B6 | 19 | 02 | _ |
| 0x000000000000000000000000000000000000 | 82 | 56 | 33 | 6E | 29 | 90 | DB | 8B | 1E | 6C | 2C | 09 | 65 | A9 | CD | FC | |
| 0x000000000000000040 | 6A | 10 | 7A | 99 | E4 | D7 | 89 | 65 | FA | FE | 43 | 9E | 26 | BF | EE | C4 | |
| 0x000000000000000050 | 0D | E2 | 8 A | 77 | BE | 79 | C0 | 71 | E7 | 5D | 18 | 36 | B 3 | B8 | 17 | BC | |
| 0x000000000000000000 | 69 | 1E | 88 | EE | AA | DC | AB | D0 | 36 | C9 | 31 | 79 | 1B | 9F | 40 | DC | |
| 0x0000000000000000070 | 5B | 53 | 5A | 88 | 80 | EF | 5C | C3 | 91 | 18 | 7A | 64 | 95 | F9 | C7 | 0A | |
| 0x000000000000000000000000000000000000 | 55 | 4F | 66 | 33 | 2A | 46 | ED | B9 | 3E | 79 | CC | 54 | A 1 | 9A | 44 | BB | |
| 0x00000000000000000 | 4B | 98 | 96 | 51 | 4D | CA | 0E | A 8 | FB | AE | 23 | 03 | 16 | 60 | 04 | 77 | |
| 0x00000000000000000 | B8 | 75 | 0 A | 47 | A1 | BB | 47 | 92 | 7A | 91 | BB | 86 | 68 | 4E | 4B | 82 | |
| 0x000000000000000000B0 | 80 | 68 | 5E | 2D | 2A | FF | FA | CD | BC | 5E | 9E | F6 | A4 | FC | B 8 | 48 | |
| 0x000000000000000000000000000000000000 | 4E | F5 | 38 | F8 | 0 C | DE | 5A | 9E | 92 | E2 | DC | 82 | 9F | A 0 | EA | 81 | |
| 0x000000000000000000000000000000000000 | 43 | 94 | A 3 | 09 | 3C | 8F | A2 | D1 | 26 | B 2 | F5 | 67 | 10 | 9A | 71 | 85 | |
| 0x00000000000000000 | CC | 49 | 47 | 6B | CB | 5B | 7B | 7C | 33 | B 2 | CD | 3A | 6B | 75 | 2C | 17 | |
| 0x000000000000000000000000000000000000 | EF | E9 | 7A | F9 | 17 | 88 | ED | 42 | 2D | 84 | 10 | 4D | 79 | 82 | DF | AF | |
| 0x0000000000000100 | 11 | C3 | 1A | 60 | 33 | 09 | 33 | 11 | CA | 12 | 44 | B2 | CE | 41 | 10 | 69 | |
| 0x0000000000000110 | 0B | C6 | F8 | 44 | EC | 29 | 0 A | 19 | F4 | 78 | 42 | 20 | B 3 | 1E | EA | 0A | |
| 0x000000000000120 | 20 | 13 | A 1 | 34 | 18 | 86 | 26 | E5 | 59 | F7 | C7 | 2E | 0 A | 33 | 47 | 84 | - |
| Buffer Checksum 0x00 | 00000 | 0 | | | | | | | | | | | | | | | |
| Goto 0x | 00000 | 000 | | | | | | | | | | | | | | | |

Fig. 5-16

Buffer Checksum: It will calculate the assigned Partition checksum.

Goto: Assign a Buffer address



- Config: Programming Settings. It will read the contents of Descriptors/Attribute/Flags from the UFS and display on each window.
- Batch Setting: Double click the programming options that are listed in the Batch Operation box or click >> or << to move the options to the Operation Selected box. The AutoBatch will program according to the listed order in the Operation Selected box.

Note: You can add Write Flag and Write Attributes before or after Wipe LUN/Blank Check/Program Chip/Checksum Verify. When executing the Attributes, it is recommended to arrange the Write Attributes to the last step (especially when setting the frequency clock bRefClkFreq of UFS), shown as Fig 5-17.

| Config | | Х |
|------------------------------------|---------------------------------------|---|
| Batch | Batch Setting | |
| Batch | Batch Operation | Operation Selected |
| Descriptors Attributes Flags | Write Flag Wipe LUN Blank check | Vrite Descriptor Program dhip Checksum verify Write Attribute |
| | | OK Cancel |

Fig. 5-17



• **Descriptors**: It provides all UFS descriptor settings. Use **Write Descriptor** to write; test read is also available on this page. (Fig 5-18).

| Config | | | | _ | | × |
|---------------------------------|-----------------|--------------|-------------------------|---------------|--------|--------------------|
| Batch Descriptors | | | | | Able t | o assign different |
| Batch Configuration Device | UFS Geomertry | Unit Power p | parameters Interconnect | String Descri | | descriptors |
| Configuration descrip Header | tor | | Unit | | | |
| | bLength 0x | 90 | LUN NO. : | 0 | \sim | |
| Descriptors | iptorType 0x | 01 | bLUEnable | 0x01 | \sim | |
| Atte | Reserved | | bBootLunID | 0x00 | \sim | |
| ы | ootEnable 0x01 | ~ | bLUWriteProtect | 0x00 | \sim | |
| Attributes bDesc | AccessEn 0x00 | ~ | bMemoryType | 0x00 | \sim | |
| Fiage bInitPo | owerMode 0x01 | · · · | dNumAllocUnits | 0x 00001dcb | | |
| bHighP | riorityLUN 0x7F | ~ | bDataReliability | 0x00 | ~ | |
| Flags bSecureRen | movalType 0x00 | \sim | bLogicalBlockSize | 0x0C | ~ | .oad Setting |
| bInitActiv | eICCLevel 0x00 | ~ | bProvisioningType | 0x02 | ~ S | Save Setting |
| wPeriodicR | TCUpdate 0x | 0000 | wContextCapabilities | 0x 0000 | | Read |
| | Reserved | | Reserved | | | |
| | | | | | | |
| | | | | | _ | |
| | Able | to Save. | Load and Rea | ad the | | |
| | | | | | | |
| | CC | onfigurat | tion descripto | ors | ОК | Cancel |



Load Setting: Load the configuration descriptor settings.Save Setting: Save the configuration descriptor settings.Read: Read the configuration descriptor settings.

*Only the configuration descriptor can read and write, other descriptors are read-only. Note: Some descriptor values are one time programming, which cannot change once it is written. Please refer to the UFS datasheet that you use before changing any settings.



• Attributes: Provides set up and read for each LUN attribute values. Use Write Attribute to write; test read is also available on this page.

| Config | | | | × |
|-------------|---------------------|-----------|------------------------|---------------|
| Batch | Attributes | | | |
| Batch | LUN Number | ~ | Able to select | different LUN |
| Des | Attributes | | | |
| Descriptors | bBootLunEn | 0x01 ~ | wExceptionEventControl | 0x0000 ~ |
| Attr | Reserved | | wExceptionEventStatus | 0x0000 |
| 207 | bCurrentPowerMode | 0x11 | dSecondsPassed | 0x0000000 |
| Attributes | bActiveICCLevel | 0x00 ~ | wContextConf | |
| Flag | bOutOfOrderDataEn | 0x00 ~ | Bit[7:6] | ooh ~ |
| 203 | bBackgroundOpStatus | 0x00 | Bit[5:3] | |
| Flags | bPurgeStatus | 0x00 | Bit[1:0] | 00b ~ |
| | bMaxDataInSize | 0x 08 | Value: | 0x0000 |
| | bMaxDataOutSize | 08 | dCorrProPII/Num | 0x00000000 |
| | dDynCaoNeeded | 0x0000000 | Deserved | |
| | bRefClkFreq | 0x01 ~ | Reserved | |
| | bConfigDescrLock | 0x00 ~ | Load | Save Refresh |
| | bMaxNumOfRTT | 0x 02 | | |
| | | \sim | | |
| Ahle to | Save/Load/Ref | resh | | OK Cancel |
| | | | | |

Fig. 5-19

Assign a LUN NO according to your need and **Refresh** to read the attributes, please ensure the information is written correctly, and then save the settings for next time usage.

Some attribute values are read-only and some can read and write. The values that you can change are the ones that are able to write.

Note: Some attribute values are one time programming, which cannot change once it is written. Please refer to the UFS datasheet that you use before changing any settings.



• **Flags:** Set up the settings and write through the Write Flag; test read is also available on this page.

| Config | | X |
|------------|----------------------------|--|
| Batch | Flage | |
| | Flags | |
| Batch | Reserved | |
| Des | fDeviceInit | 0:Device initialization completed or not started yet. $$ |
| | fPemanentWPEn | 0:Permanent write protection disabled. $\qquad \lor$ |
| Atta | fPowerOnWPEn | 0:Power on write protection disabled. $\qquad \checkmark$ |
| ALL | fBackgroundOpsEn | 1:Device initiated background ops, is enabled. $\qquad \checkmark$ |
| Attributes | Reserved | |
| Flag | fPurgeEnable | 0:Purge operation is disabled. \checkmark |
| | Reserved | |
| Flags | fPhyResourceRemoval | 0:The device shall reset this flag to 0 after completion of dynamic (\checkmark |
| | fBusyRTC | 0:Device is not executing internal operation related to RTC $\qquad \lor$ |
| | Reserved | |
| | fPemanentlyDisableFwUpdate | 0:The UFS device firmware may be modified V |
| | | Load Save Refresh |
| | Save | /Load and refresh |

Fig. 5-20

Change the settings according to your needs, and then write through the Write Flag. Refresh the flags and ensure it is written correctly. Save the values for next time usage.

Some flag values are read-only and some can read and write, the values that you can change are the ones that are able to write.

Note: Some flag values are one time programming, which cannot chang once it is written. Please refer to the UFS datasheet that you use before changing any settings.



- > **IC Info:** Chip information and cautions (Not supported yet).
- > **ReadIC:** Read and display the IC data.

| Partition | ıs | File Area | Chip Area |
|------------------------------|--|----------------------------|-------------------------|
| UN0 [Type:Flash] | | | × |
| ○ Flag ○ Attribute | O Descriptor O RPMB 💿 LUNO | O LUN1 O LUN2 | |
| ſ | | Chip | |
| Address +0 +1 | +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C | +F +0 +1 +2 +3 +4 +5 +6 +7 | +8 +9 +B +C +D +E +F |
| 0x00000001A84DE400 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 |
| 0x00000001A84DE410 00 00 (| 00 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 |
| 0x00000001A84DE420 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x00000001A84DE430 00 00 0 | 10 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x00000001A84DE440 00 00 0 | 0 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | |
| 0x00000001A84DE450 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | |
| 0x00000001A84DE460 00 00 00 | | | |
| 0x00000001A84DE470 00 00 0 | | | |
| 0x00000001A84DE480 00 00 0 | | | |
| 0x00000001A84DE450 00 00 0 | | | |
| 0x00000001A84DE4R0 00 00 00 | | | |
| 0x00000001A84DE4C0 00 00 0 | | | |
| 0x0000001A84DE4D0 00 00 00 | 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x0000001A84DE4E0 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x0000001A84DE4F0 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x00000001A84DE500 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x0000001A84DE510 00 00 0 | 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x0000001A84DE520 00 00 (| 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 0x0000001A84DE530 00 00 (| 00 00 00 00 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 🖵 |
| Buffer Checksum 0x0000000 | | Chip Checksum 0x0000000 | |
| Goto 0x 0000000 | 7 | | |
| Save Memory From 0x 00000000 | Length:0x 772C00000 | | |
| Next Different | | | |
| | | | |
| | | | |
| | | | \sim |
| | Fig. F | 21 | Eunctions |
| | FIG. 5 | -21 | i unctions |

A. Partitions

The partition numbers are configured by the actual LUN partitions of the UFS. Select a partition to display its data.

B. File Area (Buffer)

This area will display the data of the imported files.

C. Chip Area

It will read the actual IC and compare with the file data. The abnormal parts will be high-lighted in red for analysis.



D. Functions

- **Buffer Checksum and Chip Checksum:** Show the buffer checksum and the chip checksum of the partition for verification.
- **Goto:** Enter the number line in order to go to the address for examination.
- Save Memory: Saves the actual IC data of the selected partition or select a range to save. However, it is not available to save all the partitions at once.
- Next Different: Search and compare the next loaded file and edited file.
- > Blank: Blank IC checks. Select all or a LUN



Fig. 5-22

> Wipe: Erase IC data. Select all or a LUN.



Fig. 5-23



Program: Program the data to the IC. Select all, one of the LUN, or Descriptors/Attributes/Flags Program.



Fig. 5-24

Verify: Verify and compare the programming file with the IC data. Select all, one of the LUN, or Descriptors/Attributes/Flags Program.



Fig. 5-25

AutoBatch: The programming procedure will follow the order listed in the Batch Settings, which is in the Config window.

| Batch Config Setting | |
|--|--|
| Write Descriptor Program chip Checksum verify Write Attribute | |





B. LOG Window: Display and record the entire process and the results.



Fig. 5-27

C. Programmer Information: Display the programmer name, the firmware version and the FPGA version.

| NuProg-Master FPGA Ver: 15 F/W Ver: 1.2.12 S/N: NUP | Master Pass: 1 Fail: 0 N 0 | PASS |
|---|--|------|
|---|--|------|

Fig. 5-28



D. IC Information: Display all the data of the selected IC.

| ChipInfo | | | |
|-----------|-----------------|-----------|------------------------------|
| Type: | UFS | ID: | |
| Manufact: | SAMSUNG | ADP P/N1: | N-UFS-050-FBGA153-115130-010 |
| Size: | 0x77380021f | ADP P/N2: | |
| Package: | TFBGA153 | ADP P/N3: | |
| PartNum: | KLUBG4G1CE-B0B1 | | |

Fig. 5-29

E. *CheckSum Data: Display programming file name, file size, corresponding partition, and file checksum.*

| Check Sum_ | | | | |
|---------------|-----------|-----------|--------------|--|
| File CheckSum | File Size | Partition | File Name | |
| 0xFFFF210 | 0x200000 | LUNO | Full_p_2.BIN | |
| | | | | |
| | | | | |

Fig. 5-30

F. Batch Config Setting: It displays the batch settings that are selected in the Config window. When you use AutoBatch, the system will follow the listed order when programming.



Fig. 5-31





5.3.3 Examples

5.3.3.1 Only update LUN0, LUN1, and LUN2 for the IC that has been initialized, please follow the steps below:

Step 1. Install an IC on the programmer and open the software. The software will automatically detect the IC model and the LUN status.

| 14:09:19:Detecting chip | 1 | | | | | | | |
|---|--|--|--|--|--|--|--|--|
| | | | | | | | | |
| 14:09:20:Found chip KLUBG4G1CE-B0B1 by auto detecting | | | | | | | | |
| 14:09:23:Set operation:none | | | | | | | | |
| 14:09:23:OptionBytes Check | 14:09:23:OptionBytes CheckSum :0x0 | | | | | | | |
| 14:09:23:FileName:Full_p_2.BIN,CheckSum:0xFFFF210 | | | | | | | | |
| 14:09:23:Chip CheckSum :0x0 | | | | | | | | |
| 14:09:23:Select ic(UFS:KLUB | G4G1CE-B0B1[SAMSUNG]) success. | | | | | | | |
| Chip Information | | | | | | | | |
| Reference clock: 26MHz | | | | | | | | |
| LUN number: 3 | LUN number: 3 | | | | | | | |
| LUN0size: 30508 MB | | | | | | | | |
| LUN1size: 4 MB | | | | | | | | |
| LUN2size: 4 MB | | | | | | | | |
| 14:09:23:Type KLUBG4G1CE | -B0B1 is applied. | | | | | | | |
| Save Log Clear Log | | | | | | | | |
| | | | | | | | | |
| ipInfo | | | | | | | | |
| pe: UFS | ID: | | | | | | | |
| anufact: SAMSUNG | ADP P/N1: N-UFS-050-FBGA153-115130-010 | | | | | | | |
| 2e: UX//380021f | ADP P/N2: | | | | | | | |
| ickage: IFBGA153 | ADP P/N3: | | | | | | | |

Fig. 5-32

As shown in Fig. 5-33, the LUN of this UFS has been partitioned into three sections and the reference clock is 26MHz.







Step 2. Load a programming file: load a programming file to each LUN partition separately (Fig. 5-34). Since LUNO file is bigger and more dispersed, please select the Skip Blank Value to reduce the programming time.

| FilePat File For FileChe | h: C:\U: mat: Binar eckSum: Byte | sers \user \Desktop y(*.bin) \ Acc \ | Customer \Pegat | ron\EMMC_Dony_V 0x00 nk Value | VIGP: ~ | Partition Name: SectorIndex: AutoS SectorCount: | LUN0 0 et FileOffset:0x0 1048576 | 0 |
|--------------------------------|--|--|-----------------|-------------------------------------|---|--|---|----------|
| Hide Image | 25 | | | | | | ОК | Cancel |
| -14 | PartitionName | SectorIndex | FileOffset | SectorCount | FileFormat | CheckSumAlg | SkipBlank | FilePath |
| # | | | | 10.10576 | | | | |
| # Image 01 | LUN0 | 0 | 0x00 | 1048576 | Binary(*.bin) | ByteAcc | SkipBlank | C:\User |
| # Image 01 Image 02 | LUN0 LUN1 | 0 | 0x00 0x00 | 39 39 | Binary(*.bin) Binary(*.bin) Binary(* bin) | ByteAcc ByteAcc ByteAcc | SkipBlank NotSkip NotSkip | C:\User |

Fig. 5-34

After import a file, the corresponding programming file and the File CheckSum will be shown in the Check Sum Area.

| File CheckSum | File Size | Partition | File Name |
|---------------|-----------|-----------|-----------------------------------|
| 0x101ADE3 | 0x4000000 | LUNO | Scatter_map.png |
| 0x58025B5C | 0x4000000 | LUNO | wipg3000 emmc applicationzone.rom |

Fig. 5-35



Step 3. For single programming function

In order to re-work IC, please write in the order of **wipe (*Note) > Program > Verify** LUN0~2. If IC has not been programmed yet, then skip **wipe** and start from **Program >** Verify. However, if you need batch programming, please go to step 4.

***Note:** Since UFS is a rewriteable IC, it will take a while to Wipe, so please use it deliberately.

| 10.03.20 Chie Charles - 0.0 | |
|---|--|
| 0 10:02:29:Chip CheckSum :0x0 | |
| 10:02:29:Load file success. | |
| 10:02:33:Wiping ALL | |
| 10:02:40:Master, Erase success, takes 3.951s | |
| 10:02:40:Master, Erase success, takes 0.23s | |
| 10:02:40:Master, Erase success, takes 0.25s | |
| \Lambda 10:02:40:Others operation is on going. | |
| 10:05:17:Master, Program success, takes 16.426s | |
| 10:05:23:Master, Program success, takes 0.93s | |
| 10:05:29:Master, Program success, takes 0.144s | |
| 10:06:08:Master, Verify success, takes 9.445s | |
| 10:06:15:Master, Verify success, takes 0.90s | |
| 10:06:18:Master Verify success takes 0.91s | |

Fig. 5-36



Step 4. Batch Programming

Step4-1. Set up **Batch Setting** in the **Config** window. Since you will only need to update the contents of LUN0~2, and the file is bigger than the original IC file, so it is not necessary to set up and program Flag, Attribute, and Descriptor again. Therefore, only need to select Program chip and Checksum verify.

| Config | | X |
|--|--|---------------------------------|
| Batch | Batch Setting | |
| Batch | Batch Operation | Operation Selected |
| Descriptors Descriptors Attributes | Write Flag Write Attribute Write Descriptor Wipe LUN Blank check | Program chip Checksum verify |
| | | Batch Config Setting |
| | | Program chip Checksum verify |

Fig. 5-37

Step 4-2. Click Auto Batch to start programming.





5.3.3.2 If the IC is not initialized, please follow the steps below:

Step 1. If the LUN number still remains at zero (Fig. 5-39) even after the IC and socket are installed, then it might indicates the LUN is unable to write, please partition the LUN first.



Fig. 5-39

Step 2. When you see a warning message shown as Fig. 5-40, please go to **Descriptors** in the **Config** window, set up the configuration descriptor dialogs, and then partition the UFS and the LUN.







| | Configuration descriptor Header | | | | Unit | | | |
|--------|------------------------------------|-----|------|---|----------------------|-------------|--------|--------------|
| | bLength | 0x | 90 | 1 | LUN NO. : | 0 | \sim | |
| iptors | bDescriptorType | 0x | 01 | | bLUEnable | 0x01 | \sim | |
| 2 | Reserved | | | | bBootLunID | 0x00 | \sim | |
| 3 | bBootEnable | 0x0 |)1 ~ | | bLUWriteProtect | 0x00 | \sim | |
| ites | bDescrAccessEn | 0x0 | 10 ~ | | bMemoryType | 0x00 | \sim | |
| 2 | bInitPowerMode | 0x0 | 1 ~ | | dNumAllocUnits | 0x 00001dcb | | |
| 3 | bHighPriorityLUN | 0x7 | rF ∨ | | bDataReliability | 0x00 | \sim | Lood Collins |
| S | bSecureRemovalType | 0x0 | 10 ~ | | bLogicalBlockSize | 0x0C | \sim | Load Setting |
| | bInitActiveICCLevel | 0x0 | io ~ | | bProvisioningType | 0x02 | \sim | Save Setting |
| | wPeriodicRTCUpdate | 0x | 0000 | | wContextCapabilities | 0x 0000 | | Read |
| | Reserved | | | | Reserved | | | |
| | | | | | | | | |

Fig. 5-41

After changing the settings, click **Write Descriptor** to start the programming process, and then **Detect** again; the Chip Information will display the updated status of the LUN.



Fig. 5-42

Next, follow the steps in section 5.3.3.1.



5.3.3.3 If there are only two partitions originally and you want to add a third section, please follow the steps below:

Step 1. Open the software and set up the configuration descriptor in the **Config** window. (Fig. 5-43)

| Descriptors | | | | | | |
|-----------------|-------------------------|------------------|---------------------|-------------------|--------|------------|
| Configuration | Device UFS Geomertry Ur | it Power paramet | ers Interconnect | String Descriptor | | |
| Configuratio | n descriptor | | | | | |
| Header | | Un | t | | | |
| | bLength 0x 90 | | LUN NO. : | 2 | \sim | |
| | bDescriptorType 0x 01 | | bLUEnable | 0x01 | \sim | |
| | Reserved | | bBootLunID | 0x02 | \sim | |
| | bBootEnable 0x01 | ~ | bLUWriteProtect | 0x00 | \sim | |
| | bDescrAccessEn 0x00 | ~// | bMemoryType | 0x03 | \sim | |
| | bInitPowerMode 0x01 | | dNumAllocUnits | 0x 00000002 | | |
| | bHighPriorityLUN 0x7F | | bDataReliability | 0x01 | ~ | |
| bS | ecureRemovalType 0x00 | | bLogicalBlockSize | 0x0C | ~ Lo | ad Setting |
| ь | InitActiveICCLevel 0 | ~ | bProvisioningType | 0x02 | ∼ Sa | ve Setting |
| vvF | PeriodicRTCUpdate | w | ContextCapabilities | 0x 0000 | | Read |
| | Rest | | Reserved | | | |
| | | | | | | |
| | | | | | | |
| | han aat un ath | | | | | |
| SLEUNZ, and t | nen set up otn | | | | | |
| elated values a | and volumes. | | | | 01/ | Control |
| Set bLUEna | able as 1 | | | l l | UK | Cancel |
| | | | | | | |
| | | | | | _ | 1 |
| | | | | Press C |)K | |
| | | Eia 5 12 | | -1000 0 | | |
| | | Fly. 5-45 | | | | |

Step 2. Finish set up and Write Descriptor, and then click Detect; the LUN Number will turn to three.





5.3.3.4 Duplicate the bootable master IC to other blank IC through Auto Batch programming.

Step 1. First, insert a master IC. Once the IC has been detected, save the contents of Descriptor/Attributes/Flags/LUN of the master IC.

Enter Config window, and then save values from the Descriptors, the Attributes, and the Flags (For next time usage).





> Use Save Memory in the read IC to save the LUN.

| Goto | 0x | 0000000 |] | |
|------------------|----|---------|-----------|----------|
| Save Memory From | 0x | 0000000 | Length:0x | 00400000 |
| Next Different | | | | |

Fig. 5-46

Step 2. Load the images that were saved in the LUN.

Step 3. The batch setting in the Config window should be Write Descriptor > Program > Checksum Verify

| Batch Config Setting | Check Sum | | | _ |
|----------------------|---------------|----------------------|---|-----|
| Write Descriptor | File CheckSum | File Size | ^ | |
| Program chip | 0x101ADE3 | 0x4000000 | | 4 |
| Checksum verity | 0x58025B5C | 0x4000000 | ~ | · |
| | < | | > | |
| | - | | | - Y |
| ProjectName: | | ProjectCheckSum:0x00 | | |

If Write Attributes and Write Flags are required, then add it behind Checksum verify.



Step 4. Insert a blank IC and then click Auto Batch to begin LUN partitioning and programming.



Fig. 5-47

If the image File is already existed, please follow the below steps:

Step 1. Detect master IC, Click Config to read Descriptors/Attributes/Flags and set up the Batch setting.

Step 2. Load the Image file.

Step 3. Insert a blank IC, and then click Auto batch to start programming process.

5.4 Precautions when using NuProg-E (For UFS) software

- Click Config will read the descriptor, attributes, and flags of the IC.
- The loaded Partition Name will be shown as the actual IC partition that has been detected.
- Execute Write Descriptor will format the IC's LUN and the original data will be gone. If descriptor needs to be written during the programming process, please use Write Descriptor instead of Wipe.
- Wipe will erase the entire IC data, so it will affect the IC's life span.
- When using Write Attribute's bRefClkFreq value to change the frequency clock of the UFS, it is recommended to arrange it to the last one to write.



5.5 NuProg_eMMC Installation Guide

5.5.1 Double Click the NuProg_eMMC icon.



5.5.2 Software Introduction (eMMC part)





A. Main Menu and Functions





A-1. Main Menu- Advance



Fig. 5-51

- Language: Provide English, Simplified Chinese and Traditional Chinese
- Log in: Set up the IP address for remote controls.
- **General options:** Set up a temporary file for saving buffers. Since large volume IC has a great demand of storage. Therefore, if C drive does not have enough space, please choose other drive.

| General options | × |
|--|-------|
| Custom path for buffer file Finable custom path for buffer file | |
| C:\Users\user\AppData\Local\Temp\ | |
| | |
| | |
| | |
| | |
| ОК Са | ancel |

Fig. 5-52



A-2. Main Menu- Help

| P | | | | | | |
|------------|------|--|--|--|--|--|
| Advance | Help | | | | | |
| | | Download Default FPGA(ALL) | | | | |
| N 199 | 1 | Firmware Manual Update(For Experts Only) | | | | |
| Detect | | LCD Firmware Update | | | | |
| 1 | I | Launch Calculater | | | | |
| ReadIO | | User Manual | | | | |
| - Log Wing | | About DediProg | | | | |

Fig. 5-53

- **Firmware Manual Update:** Update the firmware version of the NuProg-E. Update the firmware and restart the programmer.
- Launch Calculator: It opens the calculator.
- User Manual: It links to DediProg's user manuals.

A-3. Functions (From Top to Bottom; Left to Right)



Fig. 5-54



Detect: Detect IC or choose the model number manually. If the data list has the corresponding model number that is supported, software will automatically import the values of the eMMC and the Log will appear as below (Fig. 5-55). If the IC model number that you need is not listed (Fig. 5-56, please feel free to contact us.





| Select Chip | | × | |
|---|---|----------|--|
| Chip Type | EMMC/SD | ~ | |
| Manufacture | All | ~ | Log Window Image: Window < |
| BWAMGTB41A32G | [BGA 169]-Richtek | ^ | Programmer Information Type: NuProg-Master |
| EMMC04G-M627-A0 EMMC04G-M627-B0 | 01[BGA153]-Kingston 01U[BGA153]-Kingston | | Firmware: 1.3.6 |
| EMMC04G-M627-X0 EMMC04G-S100[BG | 01U[BGA153]-Kingston GA153]-Kingston | | PPGA: 24 00:16:53:Detecting chip |
| EMMC08G-M325-A EMMC08G-M325[FE | 51[BGA 153]-Kingston BGA 153]-Kingston | | 00:16:54:Set operation:none Detecting 00:16:54:OntionBytes CheckSum:00 |
| EMMC08G-S100-A0 EMMC16G-M525-A3 EMMC16G-M525[FE | J6[BGA 153]-Kingston 51[BGA 153]-Kingston 3GA 153]-Kingston | | 00:16:54:Chip CheckSum :0x0 |
| EMMC16G-S100-A0 EMMC16G-S100[BG | 06[BGA 153]-Kingston GA 153]-Kingston | | 00:16:54:Select ic(eMMC:KLMDGAGEAC-B001[BGA153][Samsunq]) success. 00:20:05:Detecting chip |
| EMMC16G-S325[BG EMMC32G-M525-A | GA153 7.6x11.1]-Kingston 51[BGA153]-Kingston | | ▲ 00:20:10:operation timeout |
| EMMC32G-S325[BG | GA153 7.6x11.1]-Kingston | ~ | |
| ОК | Cancel | | Save Log Clear Log |

Fig. 5-56



Load: Import the programming file and set the values according to your needs.

| | | | | Load file | | | | × |
|---------------------------------|-------------------------------------|--|------------------------------|-----------------------------------|---------------|--|--|------------|
| File1 | + | | | | | | | |
| FilePatl File For FileChe | h: C:\l mat: Bina eckSum: Byb | Jsers\user\Desktop\1 ary(*.bin) v eAcc v | 420035668 UBC FileOffset: | D31_HWReset_eN 0x00 k Value | IMC.t V | Partition Name: SectorIndex: AutoSet SectorCount: | LUN0 0 t FileOffset:0x0 32768 | 0 |
| Hide Image | es | | | | | | OK | Cancel |
| # | PartitionName | SectorIndex | FileOffset | SectorCount | FileFormat | CheckSumAlg | SkipBlank | FilePath |
| Image 01 | LUNO | 0 | 0x00 | 32768 | Binary(*.bin) | ByteAcc | NotSkip | C:\Users\u |
| | | | | | | | | |
| | | | | | | | | |
| _ | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| < | | | | | | | | > |

Fig. 5-57



Value Descriptions:

- File Format: The format of the programming file.
- **File Checksum:** The file checksum's calculation method.
- File Offset: Appoint an address to start loading the buffer.
- File Path: Assign the path for the programming file.
- **Skip Blank Value:** Enable this function to analyze and skip the blank data before programming, which will write more efficiently and reduce the programming time.
- Partition Name: It depends on the partitioned sections.
 Note: This partition here will only show the amounts and settings according the

IC that has been detected.

- Sector Index: Set up the sector starting point
- Sector Count: Set up total programming sectors.
- **AutoSetFileOffset**: When set up the **Sector Index**, this function can automatically calculate the File Offset.
- +: When there is more than one programming file, please press "+" to add the files to the data list.
- Show Images and Hide Images: Open and close the data list.
- > Load Prj: Import previous programming files (Not Supported Yet).
- Save Prj: Save all the settings as a programming file (Not supported yet).



Buffer: Temporary files.

Choose a partition to fully display its content.

| UserArea | 0 | Boot | 1Are | a | 0 | Bo | ot2A | rea | | OF | RPM | в | | С |) GP | PO | | |
|-----------------|---------------|-------|------|----|----|------|------|-----|----|-----|------|----|----|----|------|----|----|--|
|) GPP1 | 0 | GPP | 2 | | C |) GF | PP3 | | | 0 6 | extC | SD | | | | | | |
| Address | | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C | +D | +E | +F | |
| 0x000000000000 | 0000 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0010 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0020 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0030 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0040 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0050 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0060 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0070 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0800 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0090 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0 A 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 00B0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 00C0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 00D0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0 0E0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0 0F0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0100 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x0000000000000 | 0110 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x000000000000 | 0120 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| Buffer Checksum | 0x00 | 00000 | 0 | | | | | | | | | | | | | | | |
| Goto | 0x (| 00000 | 000 | | | | | | | | | | | | | | | |

Fig. 5-58

Buffer Checksum: It will calculate the assigned Partition checksum.

Goto: Assign a Buffer address



- Config: Programming Settings. It will read the contents of Descriptors/Attribute/Flags from the UFS and display on each window.
- Batch Setting: Double click the programming options that are listed in the Batch
 Operation box or click >> or << to move the options to the Operation Selected box. The
 AutoBatch will program according to the listed order in the Operation Selected box (As Fig. 5-59).

| Config | | | | × |
|----------------|---|----------|---------------------------|---|
| Batch | Batch Setting Batch Operation | | Operation Selected | |
| extCSD RPMB | extCSD Program chip Checksum verify | >> << | Erase chip Blank check | |
| | | - | OK Cancel | |

Fig. 5-59



• **Ext CSD Setting:** Provide Extend CSD setting for eMMC. Set up ExtCSD on this page, and then add this option in the Batch Setting.

| Config | | | | × |
|---|---------------------------|---|----|---|
| Batch | Option Bytes | | | |
| Batch Batch Desse extCSD RPMB | Option Bytes Address Oxa2 | Value 0x1 Ext CSD value setting | | Address: Ox a2 Value: Ox 1 ADD DELETE RESET |
| | | | | |
| | | | | |
| | | | ОК | Cancel |

Fig. 5-60

Add: Add a new ExtCSD value to the list.Delete: Delete all the selected ExtCSD value.Reset: Clean up all ExtCSD values.

X The address and the values will be in hexadecimal. In addition, Ext CSD will write in the listed order, therefore, set up Ext CSD for User Area's GPP and Enhanced setting according the JEDEC standard.



RPMB: Since RPMB needs a set of Key to read and write normally, so set up the RPMB key on this page.

| Config | X |
|--------|----------------------------|
| Batch | Option Bytes |
| Batch | Set RPMB Key |
| Des | KWP5 Key content (HEX): |
| extCSD | Key path: |
| Des | Load key Save key |
| КРМВ | |
| | Save and load the RPMB Key |
| | |
| | |
| | |
| | |
| | |
| | |
| | OK Carrel |
| | OK |

Fig. 5-61

- IC Info: Chip information and cautions. (Not supported yet)
- > **ReadIC:** Read and display the IC data.





A. Partitions

It will configure according the actual partition of eMMC, and then you switch to see different partitions.

B. File Area (Buffer)

This area will display the data of the imported files.

C. Chip Area

It will read the actual IC and compare with the file data. The abnormal parts will be high-lighted in red for analysis.

D. Functions

- **Buffer Checksum and Chip Checksum:** Show the buffer checksum and the chip checksum of the partition for verification.
- **Goto:** Enter the number line in order to go to the address for examination.
- Save Memory: Save the actual IC data of the selected partition or select a range to save. However, it is not available to save all the partitions at once.
- Next Different: Search and compare the next different file.
- **Blank:** Blank IC checks. Select **all** or choose one of the partitions.



> **Erase:** Erase IC data. Select **all** or choose one of the partitions.





Program: Program the project file to the IC. Select all or choose one of the partitions.



Fig. 5-65

Verify: Verify and compare the project file with the IC. Select all or choose one of the partitions.



Fig. 5-66

Auto Batch: It will program according to the listed order in the Batch Setting, which is in the Config window.



B. LOG Window: Display and record the entire process and the results.



Fig. 5-67

C. Programmer Information: Display the programmer name, the firmware version and the FPGA version, and the serial number.



Fig. 5-68

D. *IC Information: Display the part number and the relate information according to the selected chip.*

| ChipInfo | | | |
|-----------|-----------------|-----------|-------------------------|
| Type: | eMMC | ID: | 47 44 4d 15 00 43 47 41 |
| Manufact: | Samsung | ADP P/N1: | |
| Size: | 0x1d1f800200 | ADP P/N2: | |
| Package: | BGA153 | ADP P/N3: | |
| PartNum: | KLMDGAGEAC-B001 | | |

Fig. 5-69



E. CheckSum Data: Display project file name, file size, corresponding partition and file CheckSum.

| Check Sum | | | | |
|---------------|-----------|-----------|--------------|--|
| File CheckSum | File Size | Partition | File Name | |
| 0xFFFF210 | 0x200000 | LUNO | Full_p_2.BIN | |
| | | | | |
| | | | | |

Fig. 5-70

F. Batch Setting: The Batch setting is in the Config window, and AutoBatch will program in the listed order.

| Batch Config Setting | | | |
|---|--|--|--|
| Erase chip Program chip Checksum verify | | | |

Fig. 5-71



VI. Revised Edition

| Date | Versions | Changed |
|------------|----------|--|
| 2015/11/26 | 1.0 | First Edition |
| 2016/04/26 | 1.1 | Photos Changed. |
| 2016/08/17 | 1.2 | Software Images and some contents |
| 2016/10/2 | 1.3 | Added eMMC instruction |
| 2017/12/10 | 1.4 | Added some notes to the power adaptors and modified company information. |

DediProg Technology Co., Ltd

 Taiwan Headquarter
 TEL: 886-2-2790-7932
 FAX: 886-2-2790-7916

 4F., No.7, Ln. 143, Xinming Rd., Neihu Dist., Taipei City 114, Taiwan

China Office TEL: 86-21-5160-0157

Room 518, Building 66, Lane1333, Xinlong Road, Vanke Hongqiao CBD.Min Hang District, Shanghai, P.R.C. 201101

U. S. Office TEL: 1-909-274-8860

209 E Baseline RD, Suite E208 #8, Tempe, AZ, 85283, USA

Technical Support: support@dediprog.com Sales Support: sales@dediprog.com

Information furnished is believed to be accurate and reliable. However, DediProg assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties which may result from its use. Specifications mentioned in this publication are subject to change without notice.

This publication supersedes and replaces all information previously supplied.

All rights reserved Printed in Taiwan.