Overview

SLICTool is a new tool to insert a SLIC table using one of three main methods:

1) SSV3 - used for most AMI BIOSes. Does NOT work for some MSI boards. This is the safest for non-MSI boards.

2) SSV2 - used for MSI boards. Asus boards should use the SSV3 method, SSV2 is NOT needed.

3) Dynamic - The SLIC table is introduced into the BIOS so that, along with other ACPI tables, it's memory location is dynamically allocated by the BIOS. This is how manufacturers would insert a SLIC table, HOWEVER, it requires code modification to overcome locks - hence it is less safe than SSV2 or 3 which should be used in preference.

Notes:
- BIOS flashing is not for the faint hearted. There is ALWAYS an element of risk. Please ensure you are familiar with the procedure, and have necessary floppy disks etc., to allow a recovery in case of a flash failure.
- We assume you have a legitimate edition of Vista. Piracy is NOT condoned.
- Please remove any softmods. For Vistaloader this includes repairing the boot sector and MBR of the hard disk.
- There is an attempt to automatically identify the manufacturer from the BIOS. Please ensure the correct BIOS manufacturer is selected from the drop down list before modding. This is most important for MSI, BIOSTAR and XFX. If your manufacturer is not listed then it is safe to select Other and try an SSV2 or SSV3 mod. (core v8 mods only)

Flash recovery information:
- Read here - http://www.biosman.com/biosrecovery.html - note that sometimes the filename needs to be different - the boot block error message will tell you, however this does mean you need a second machine to be able to change the file name.
- Also read here - http://forums.mydigitallife.info/showthread.php?t=2105
- Ensure that the version on the disk is the (unmodified) version you are trying to flash with SLIC - avoids problems with bootblock flashing.

SLIC and SLP files - The program needs:
- A 374 byte SLIC file (ASUS included, others available from other tools and softmod archives). Checksum is automatically corrected.
- A one line text file containing the SLP string you wish to be inserted. Again, ASUS is included. See http://www.msfn.org/board/index.php?showtopic=71016&view=findpost&p=534097 - only the text string (without ') after the hex numbers is needed in the TXT file
- 911medic has kindly uploaded his collection of SLIC .BIN files here - http://slics.myftp.org/slic.rar.
- For 2.1 SLICs look in http://forums.mydigitallife.info/showthread.php?t=5952.

Vista Keys and Certificates
- Keys can be found here - http://forums.mydigitallife.info/showthread.php?t=2581
- Certificates can be found here - http://rapidshare.com/files/118119842/39-CERTS.zip

XP Keys and OEMBIOS files
- Keys can be foud here - http://forums.mydigitallife.info/showthread.php?t=2602

Method

1) Flash, using the manufacturers recommended method, the unmodified BIOS of the same version. This allows the bootblock etc. to be updated.
2) Mod the BIOS using the tool
3) Please ensure the correct BIOS manufacturer is selected from the drop down list before modding. This is most important for MSI, BIOSTAR and XFX. If your manufacturer is not listed then it is safe to select Other and try an SSV2 or SSV3 mod.
4) Flash the SLIC'd BIOS from DOS. This ensures the bootblock is left intact and maximises the chance of a successful recovery if a problem was to occur.
5) You then still need to use the appropriate key and certificate for Vista, and key and OEMBIOS files for XP.

Many thanks to Yen, Zort, 911medic and P4Spooky for their invaluable advice and feedback.

** This tool is freeware and may not be sold or redistributed for a charge **

General Options

Only alter RSDT and XSDT tables - To be compliant with ACPI specs the OEM ID of all ACPI tables should match. The OEM ID is taken from the SLIC table that is inserted. However, most early mods only changed these two tables.

Only replace OEM ID in additional tables - To be complaint with ACPI specs. If unticked then the OEM ID and the OEM Table ID is copied from the SLIC table to the other ACPI tables (either RSDT/XSDT or all depending on the state of ‘Only alter RSDT and XSDT tables’).

Replace dummy SLIC table ID with - If there is a SLIC table (usually filled with 0s) in the 1B module already it is necessary to change the table ID from ‘SLIC’ to something else. 'OEMS' is default and ACPI compliant.

Always shrink 1B module - The tool will run the same code used for the SSV2 method to try and shrink the 1B module for all methods. Occasionally with an SSV1/SSV3/Dynamic mod there is not enough space in the BIOS for the enlarged 1B module.

Use extra space to shrink - When shrinking the 1B module (either for SSV2 or if 'Always shrink 1B module' is ticked) there are 3 strings in the 1B module than can be nulled to increase compressibility. When this option is ticked another 3 strings are nulled, if needed. Note that this can cause problems when tools like EZ-Flash are used subsequently as some identifications strings may be destroyed. DOS flashing would still work.

Ignore checksum and module 80h warnings - Before and after modification the tool automatically checks main BIOS and boot block checksums. If these are invalid then the process fails. Likewise if using AMIMMWIN would damage the 80h (ROM information) module the process fails. Check this option to allow an override.

Repair AMIMMWIN damage - AMIMMWIN can damage the resulting image in a number of ways (overwrite initial bytes (ie. ASUSTEK string), damage module 80h, invalidate main BIOS checksum, damage Extended Bootblock).

Preserve boot block - Both AMIMMWIN and MMTool alter the bootblock. This option copies the bootblock and extended bootblock (if present) in entirety from the original image.

Preserve unlinked areas - If bytes in the ROM image are not part of a module, the bootblock or a ROM hole then AMIMMWIN and MMTool will not transfer them and replace them with FFh. This option will preserve them.

Automate MMTool - Automate the process of inserting/replacing modules with MMTool. Currently does not work on non-English systems.

Correct BIOS image 8 bit checksum - Ensures that the entire modified ROM image sums to the same 8 bit checksum as the original. Needed only for BIOSTAR ROMs.

Allow 8 bit checksum in NVRAM area – Allow the 8 bit checksum correction byte to be inserted in the NVRAM Reserved area.

Remove Intel lock – Remove the lock in Intel AMI BIOSes.

Replace pubkey/marker in main BIOS – Replace a pubkey/marker, if present in the main BIOS image.

Replace pubkey/marker in 1B module – Replace a pubkey/marker, if present in the 1B module.

Replace split OEM/Table ID strings – Replace any split OEM/Table ID strings in code with the OEM/Table ID from the new SLIC table.

Replace additional OEM/Table ID – Replace the specified OEM/Table ID (either split or complete) with that from the new SLIC table. Split IDs are removed only if 'Replace split OEM/Table ID strings' is selected.

Remove existing SLP string – Removes any existing SLP string.

SSV Options (Core v7 only)

Use alternate method – Use an alternate method of patching the ACPI code to insert the SLIC table address. Only used if normal method is unsuccessful.

Insert SLP string SSV2 style – Insert the SLP1.0 string into the main (02) module without altering its size

SSV3 Options

Adjust location of SLIC – It is normal to place the SLIC table 200h bytes before the AMIBIOS string in the 1B module. However in some BIOSes (ie. ASUS eeePC) this area is occupied. When ticked, the tool will move the SLIC table forward by up to 80h bytes.

Shrink 1B – The tool will run the same code used for the SSV2 method to try and shrink the 1B module for SSV3 and Dynamic methods. It will not fail if the size difference is not zero. This allows an SSV3 mod where there is very limited space for the module to expand.

SSV2 Options

Range for SLIC insertion - The SLIC table will only be inserted in the ROM image between these addresses.

Manual location for SLIC insertion - The SLIC table will be inserted at this address in the ROM image.

Ensure padding before SLIC in BIOS - Ensures there is at least 64 FFh bytes are present before the SLIC table.

Force insertion below AMIBIOS string - When ticked, the SLIC table will only be inserted before the AMIBIOSC header in the ROM image. This is needed for ASUS ROMs, but not for MSI or XFX ROMs.

Allow SLIC in NVRAM area – Allow the SLIC table to be inserted in the NVRAM Reserved area. The NVRAM area is almost always blank and just a space holder. Sometimes there is no other space. Mods with a SLIC here have been successful.

Adjust BIOS 32 bit checksum - If the SLIC table is inserted before the AMIBIOSC header then a corrective 32 bit sum is placed immediately after the table to ensure that the main BIOS checksum in the AMIBIOSC header is correct and unchanged. If this is not possible then the main BIOS checksum is corrected only if this is ticked. This should be unticked for MSI and XFX. It should be ticked for ASUS. Irrespective, the tool will not produce a modified ROM with an invalid main BIOS checksum.

SSV2 Options (Core v7 only)

Use LHA to compress module – Use LHA v2.55 to compress the modules, rather than MMTool2

Insert SLIC as module – Insert the SLIC table as a module using MMTool2, rather than inserting directly into BIOS.

Allow SLIC in bootblock – Allows the SLIC table to be inserted in the bootblock. This is more risky than a normal SSV2 mod, but has proven successful in one (the only) case.

Dynamic Options

Remove lock - Remove config lock from 1B module that prevents an inserted dynamic SLIC from being utilised. Usually found in ASUS BIOSes.

Replace FC module – Replace the FC (or other Fx) module rather than a dummy SLIC in the main module.

Replace 1B module - Replace the dummy SLIC in the 1B module

Included Tools

AMIMM.EXE – Used for core 6/7 mods when 12 byte headers are present.

AMIMMWIN.EXE – Used for core 8 mods (especially when finding module size in SSV2) as can be invoked silently with command line options.

MMTOOL.EXE – Used for SSV1/3 and Dynamic mods be default (AMIMMWIN can damage the BIOS in a number of ways). Automated keypresses as no command line capability.

MMTOOL2.EXE – Used for core 6/7 mods when 20 byte headers are present. Has command line capability.

LHA255.EXE – Optionally used to compress modules for core 7 SSV2 method

LHA0.EXE – A modified version of LHA2.67 to extract modules (modified to disregard CRC)