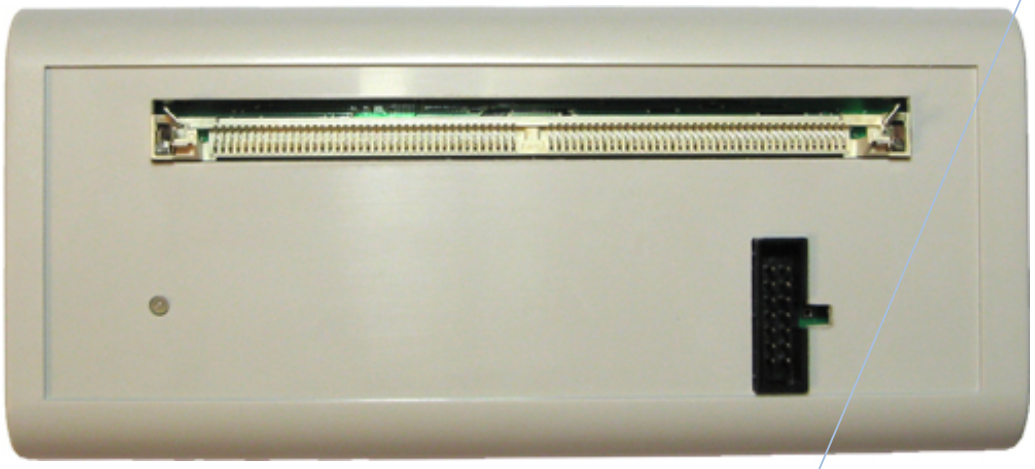


SIMM Flash Progger v3

User manual v3.2

Introduction

The programmer is designed for reading, erasing, programming, memory chip 32-bit standard modules SIMM80;

Read, erase, programming chip AT89S52, AT89S53, AT89S8252, AT89S8253 and other in-circuit via the SPI interface (ISP - In System Programming);

Reading and programming of the memory chips via the I2C interface

Technical characteristics

1. Dimensions 172x77x25mm
2. USB communication interface
3. No external power supply
4. Automatic detection algorithm of the module type and write depending on the chipset used in the module
5. Verification of the contents of the modules with a base configuration file for the three hashes - MD5, SHA1, AGI CRC, possibly independent add / edit / remove hashes
6. ISP programming - from 4 to 20 seconds, depending on the type of chip
7. Three-color indication of the programmer with the ability to change for each mode of operation

Package included

1. Programmer
2. I2C interface adapter
3. USB cable
4. Software and drivers

Software

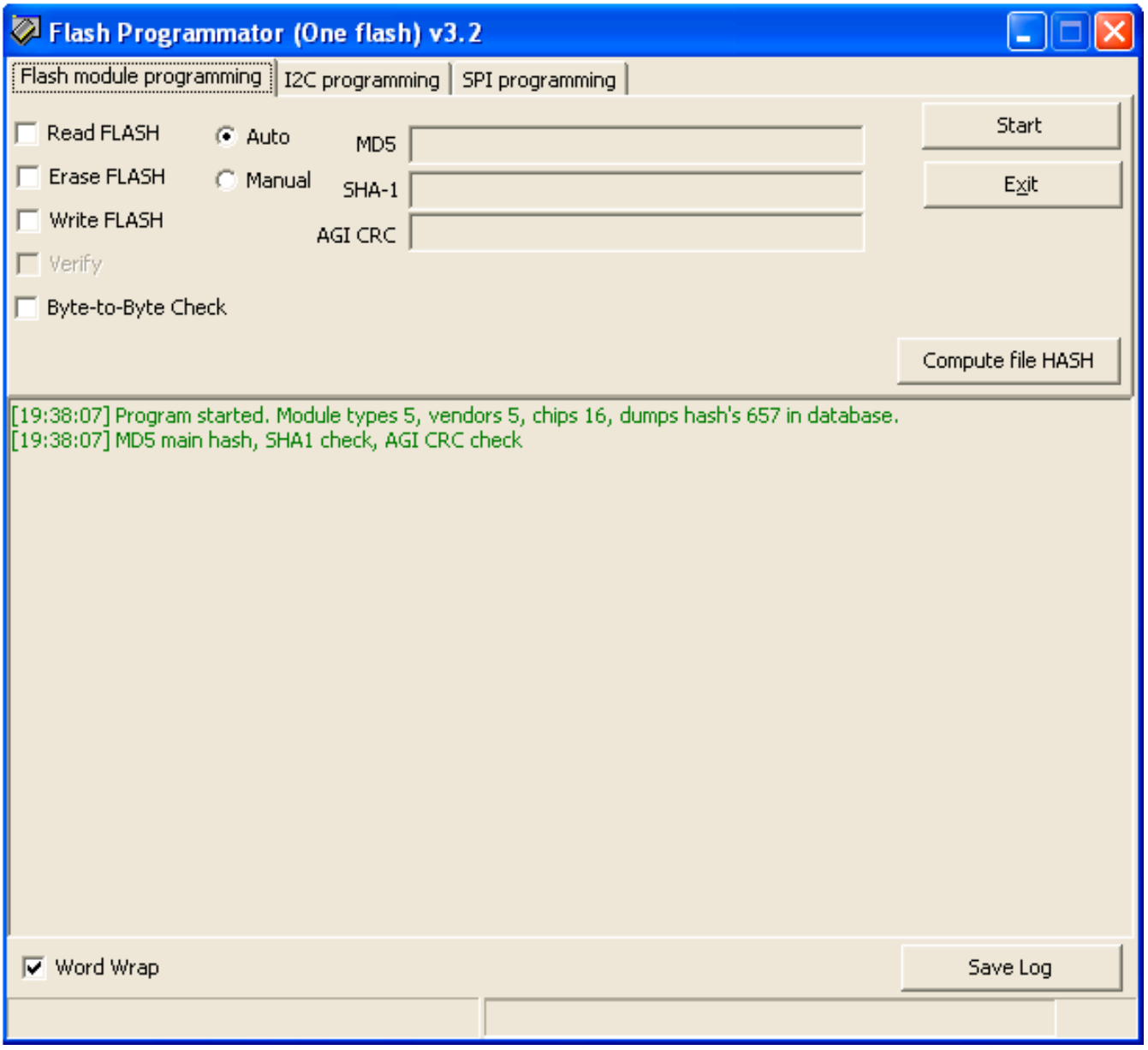
1. Installation

Run the file setup_FP1_v3.2.exe. Select the installation folder (in the future - [INSLALL_DIR])

Connect the programmer. The system detects the new device. Specify the location of the driver - [INSTALL_DIR] \ Driver

2. Programmer usage

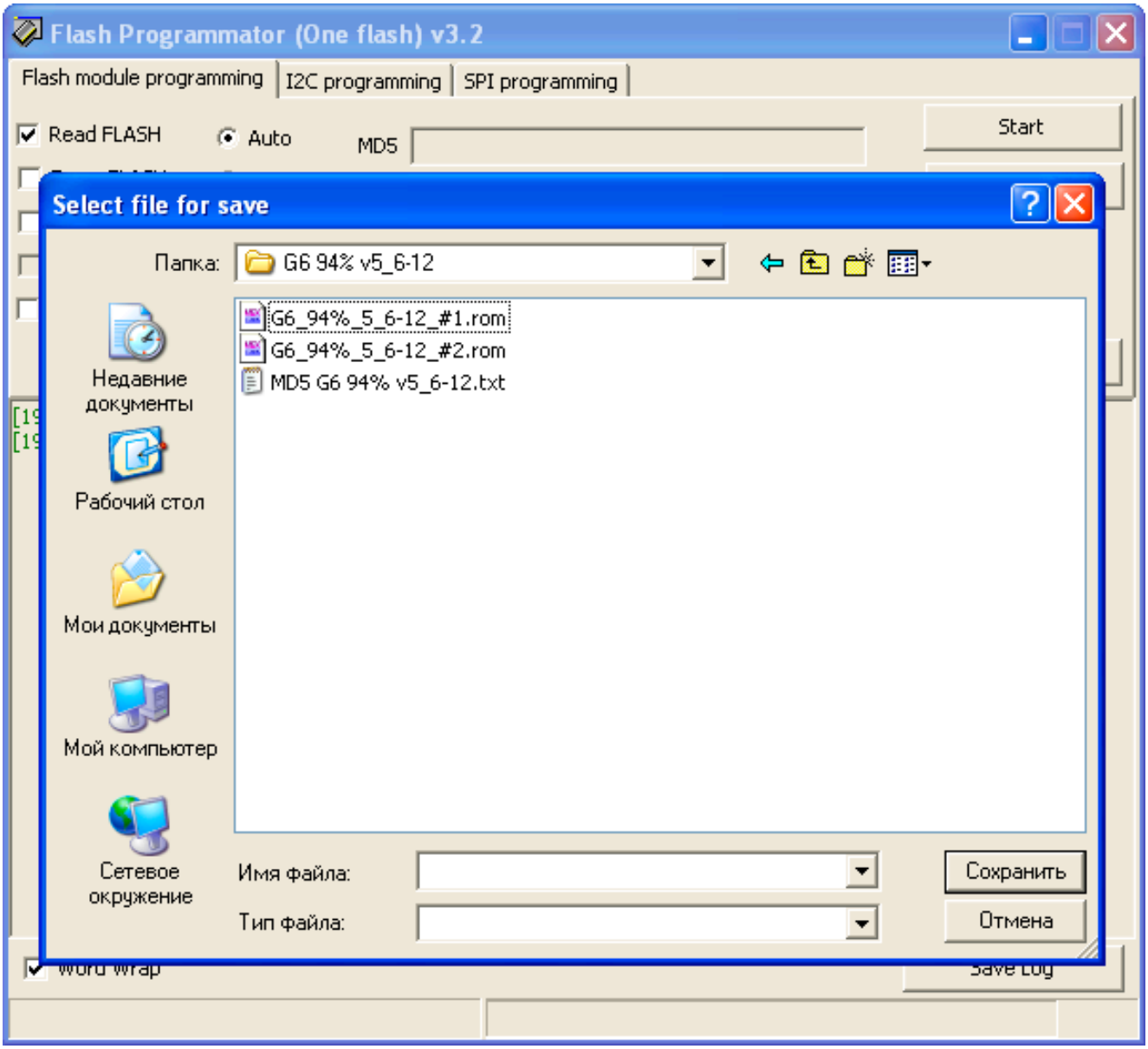
After starting the program opens a window (all figures are approximate and depend on the configuration INI file)



Flash Module Programming tab

Read Flash

Choosing this item opens a dialog box where you specify where the content of flash module will be stored:

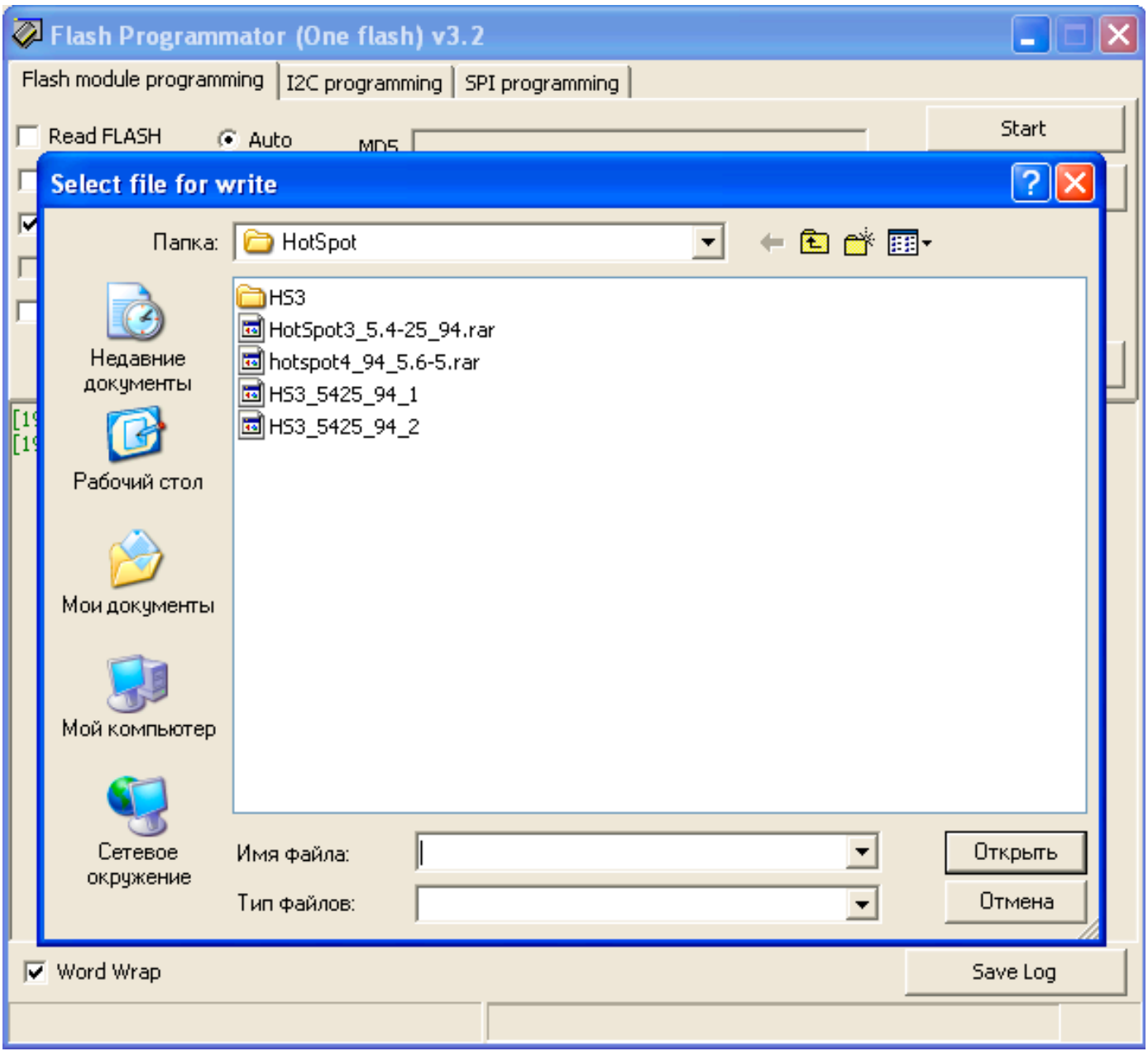


Erase Flash

Selecting this item includes a list of tasks erasing chip SIMM module

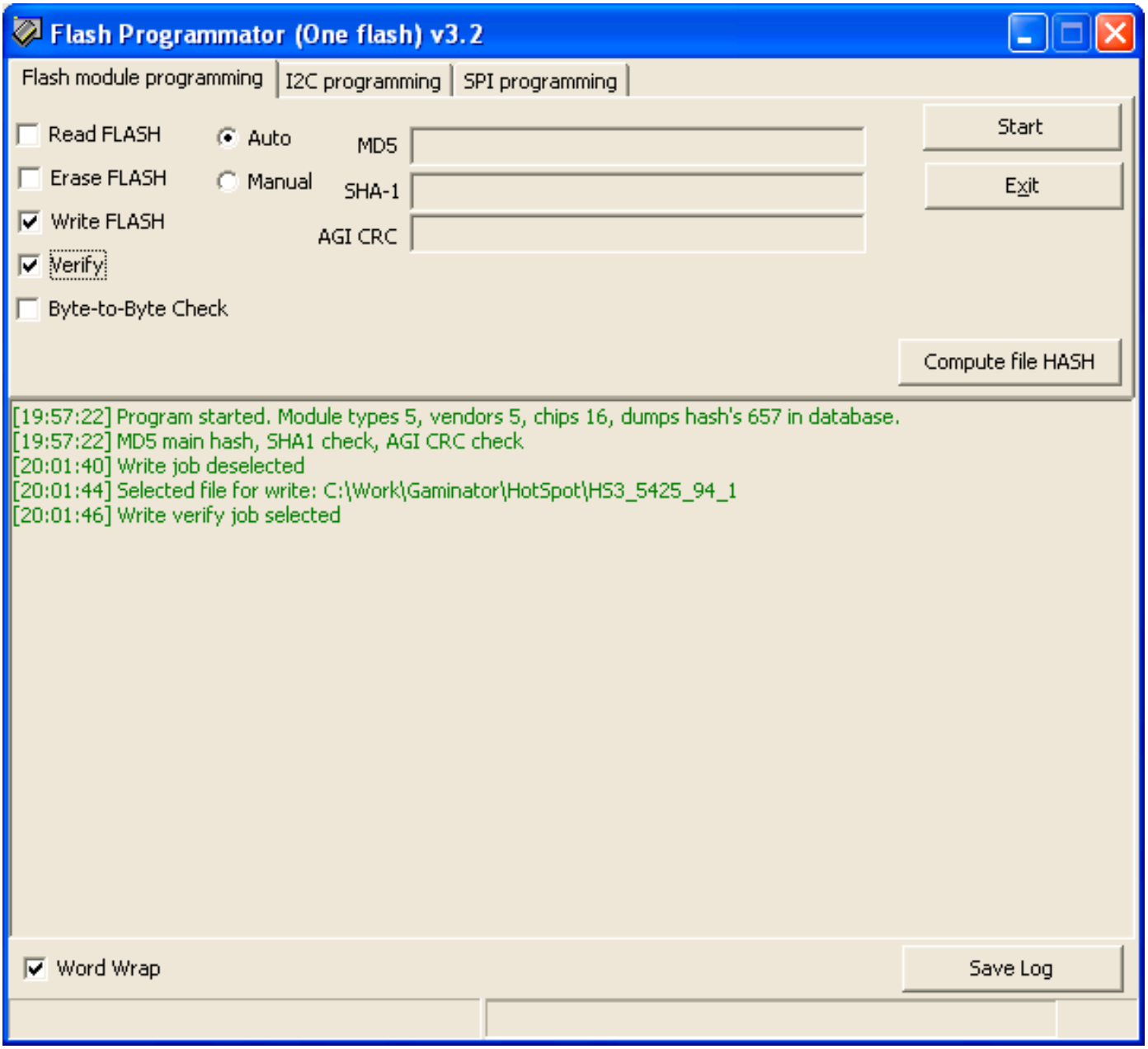
Write Flash

Choosing this item opens a dialog box where you specify the contents of the file will be written to the flash module



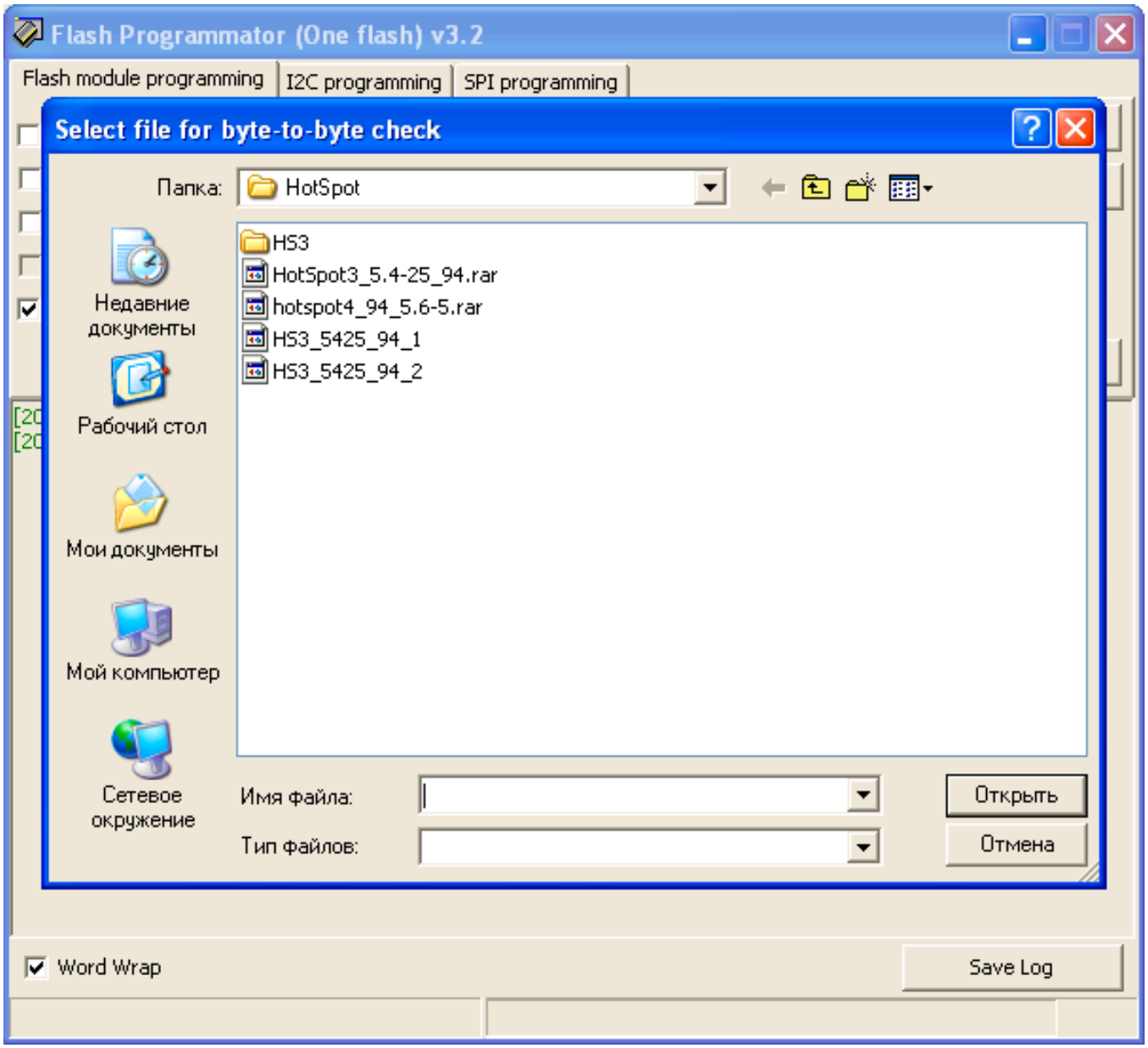
Verify

After selecting the **Write Flash**, opens the possibility of selecting the **Verify**. If this mode is enabled, after recording will be made byte-to-byte checking contents of the module with the firmware file.



Byte-to-byte check

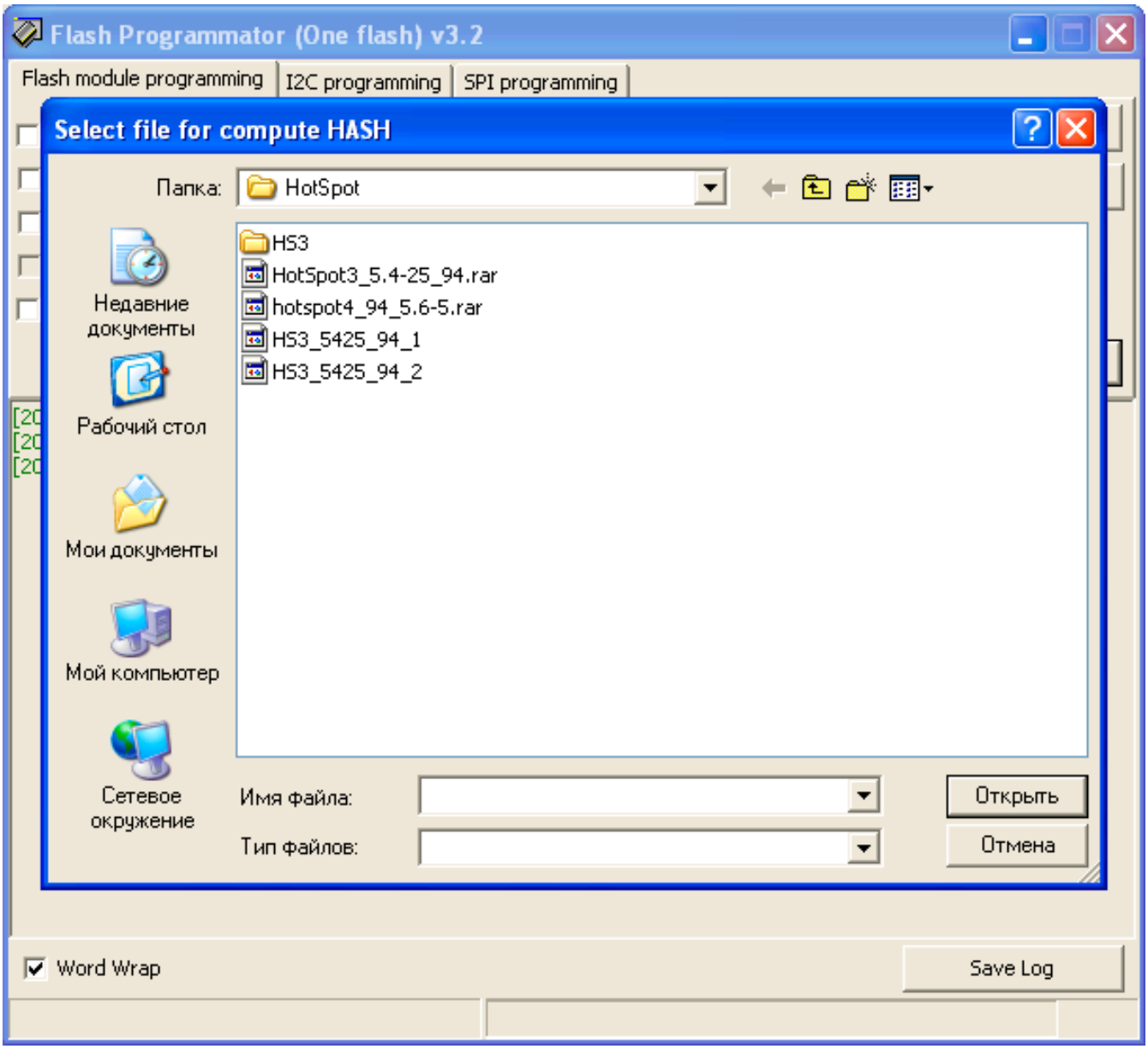
Choosing this item opens a dialog box where you specify which file will be done byte-to-byte checking with the contents of the module



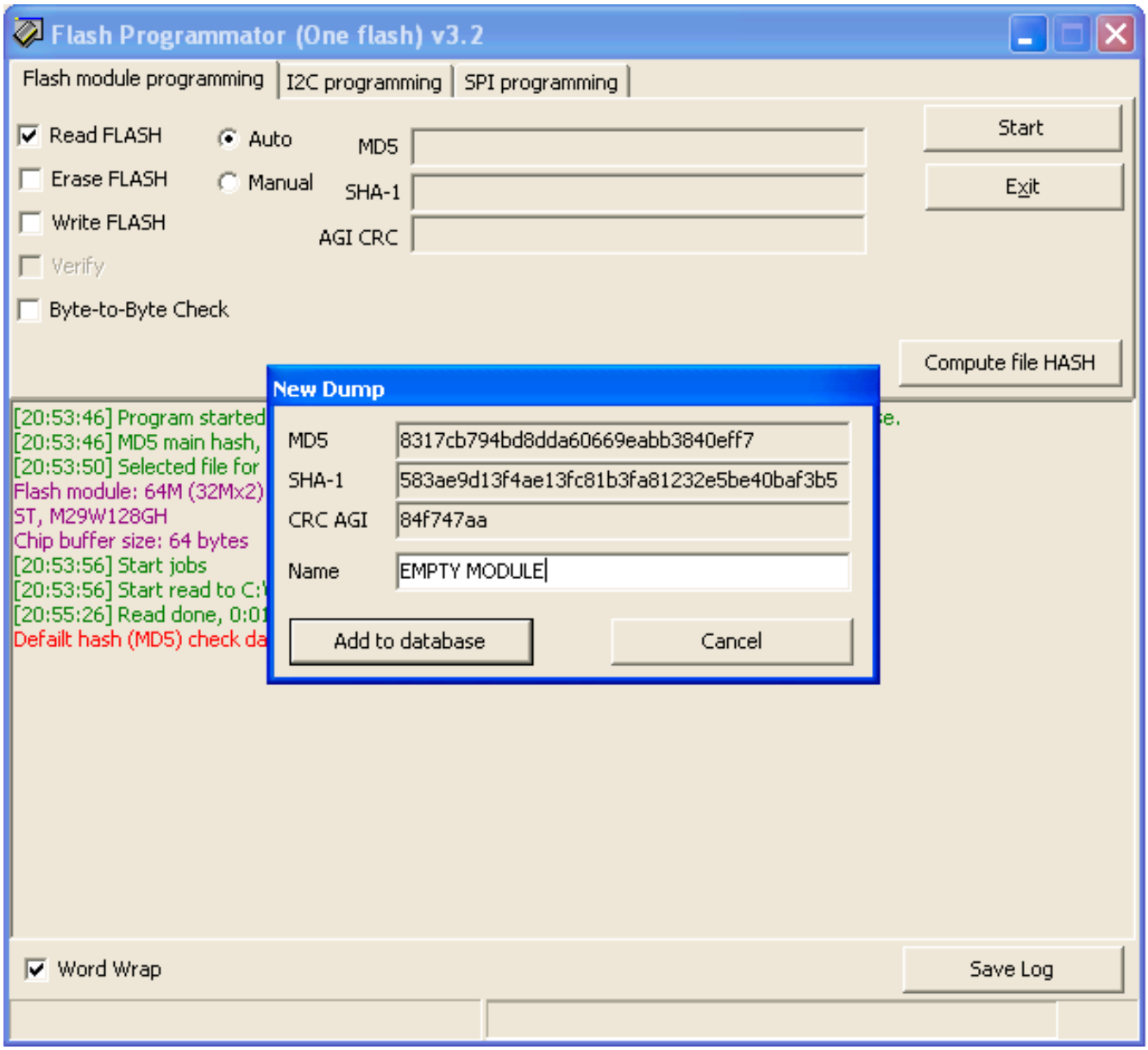
Compute HASH

After clicking on this button opens a dialog box where you specify a file for which checksum **MD5**, **SHA1** and **AGI CRC**.

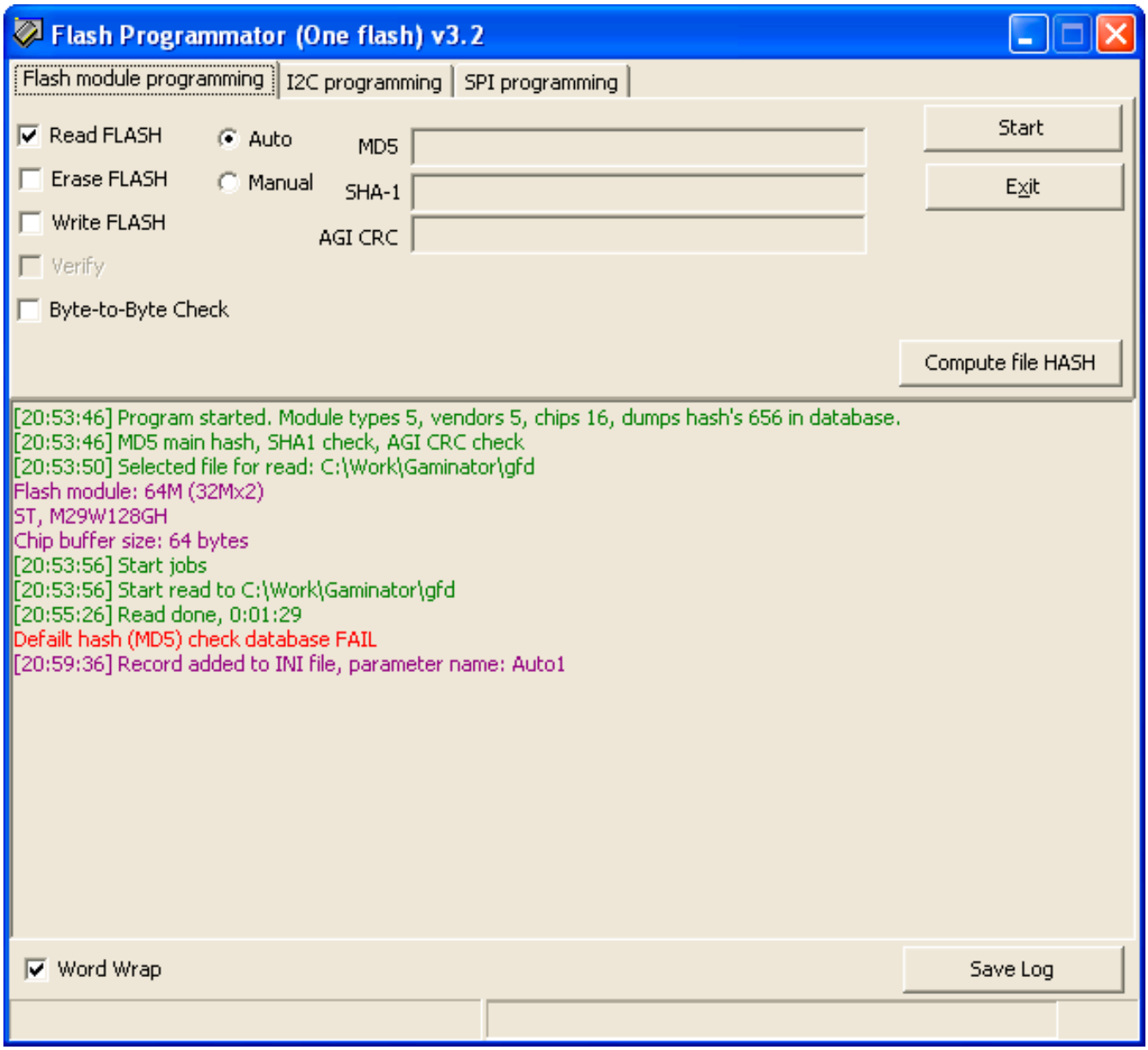
Attention! To execute the transaction programmer must be connected to the USB port!



After reading the module, or checksumming a file is checked with a database of checksums stored in the INI file. If no match is found, you can make an entry in the database, simply by specifying the name



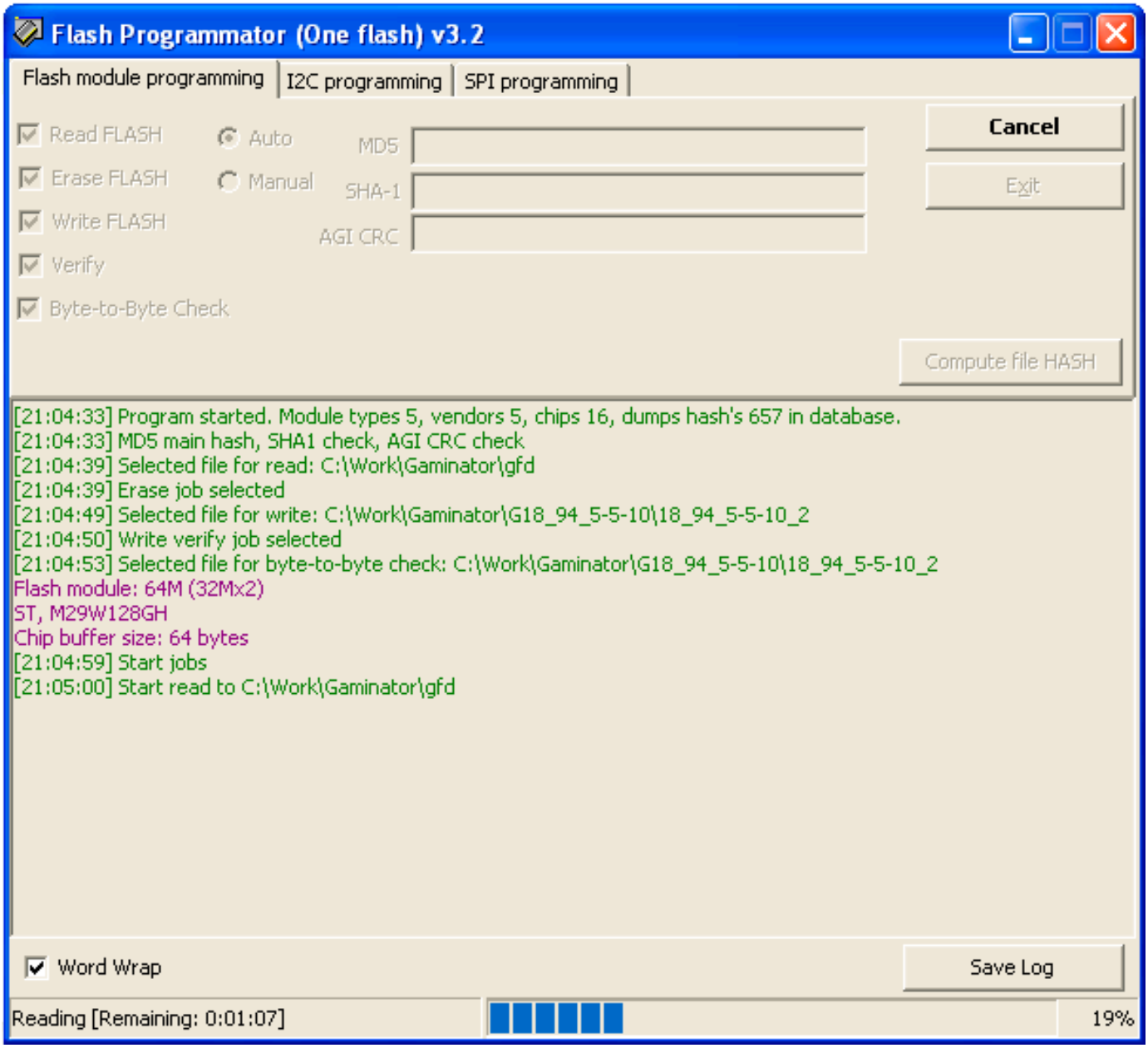
These checksums match the empty 64-megabyte module.

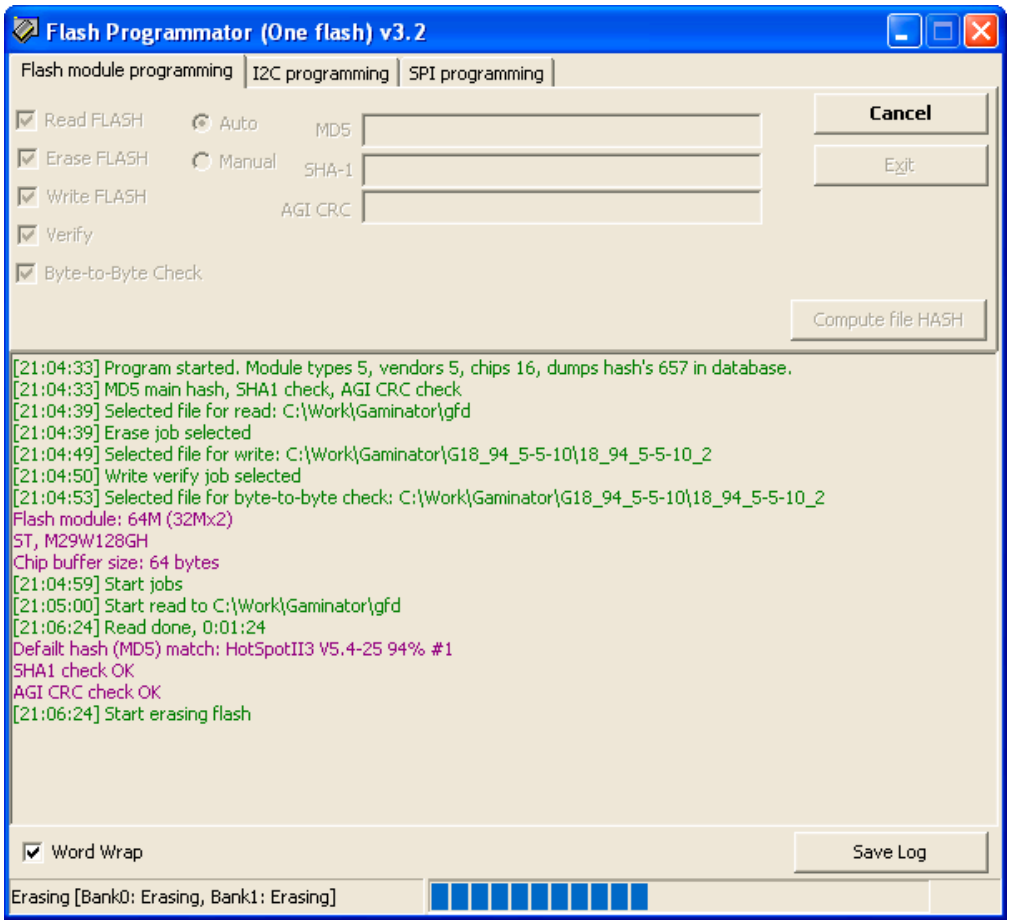


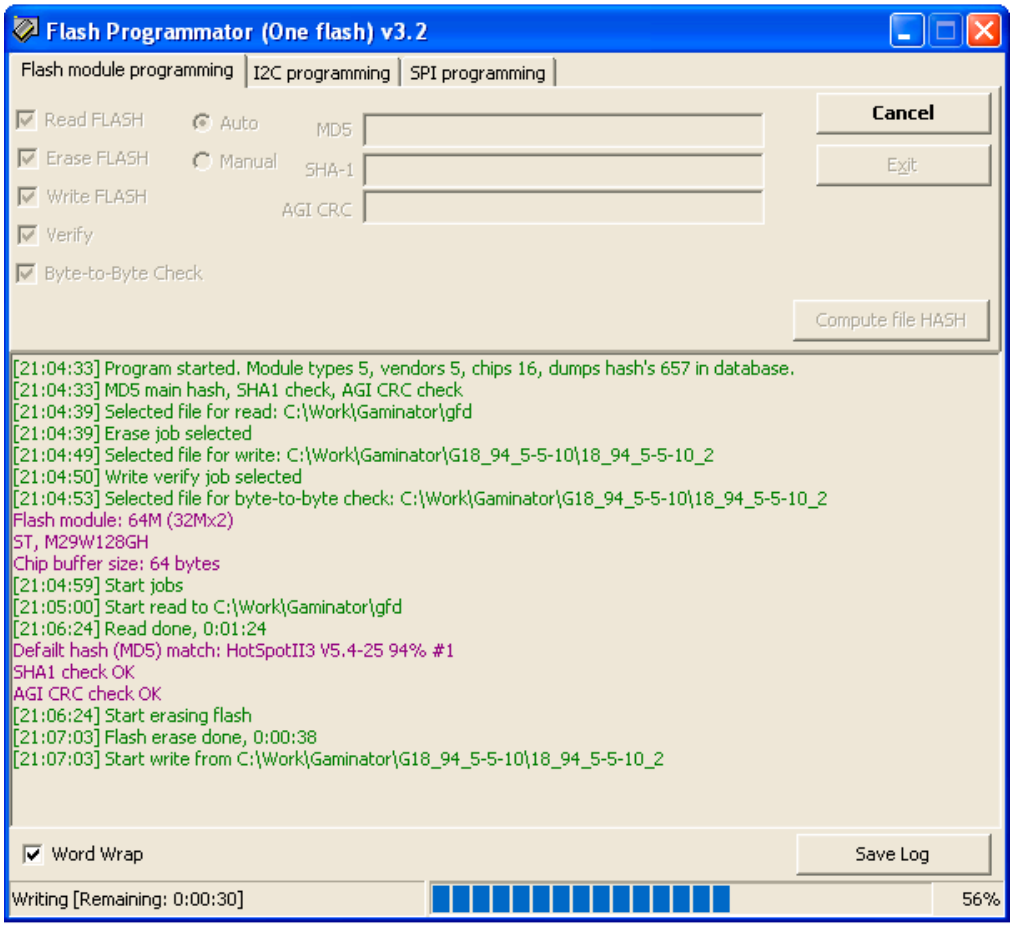
After adding the new entry is created in section **[HASH]** INI file name parameter is automatically determined (in this case - **Auto1**)

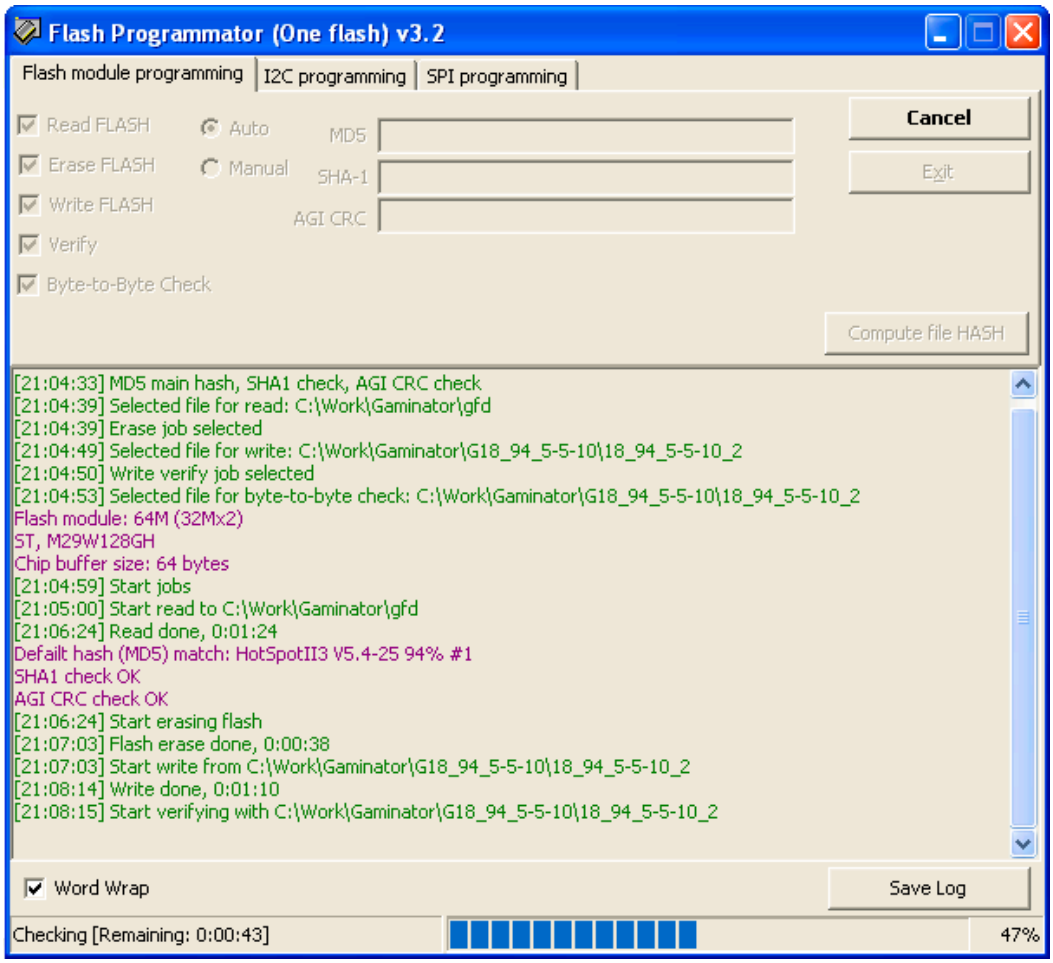
After selecting the required jobs, click on **"Start"** button

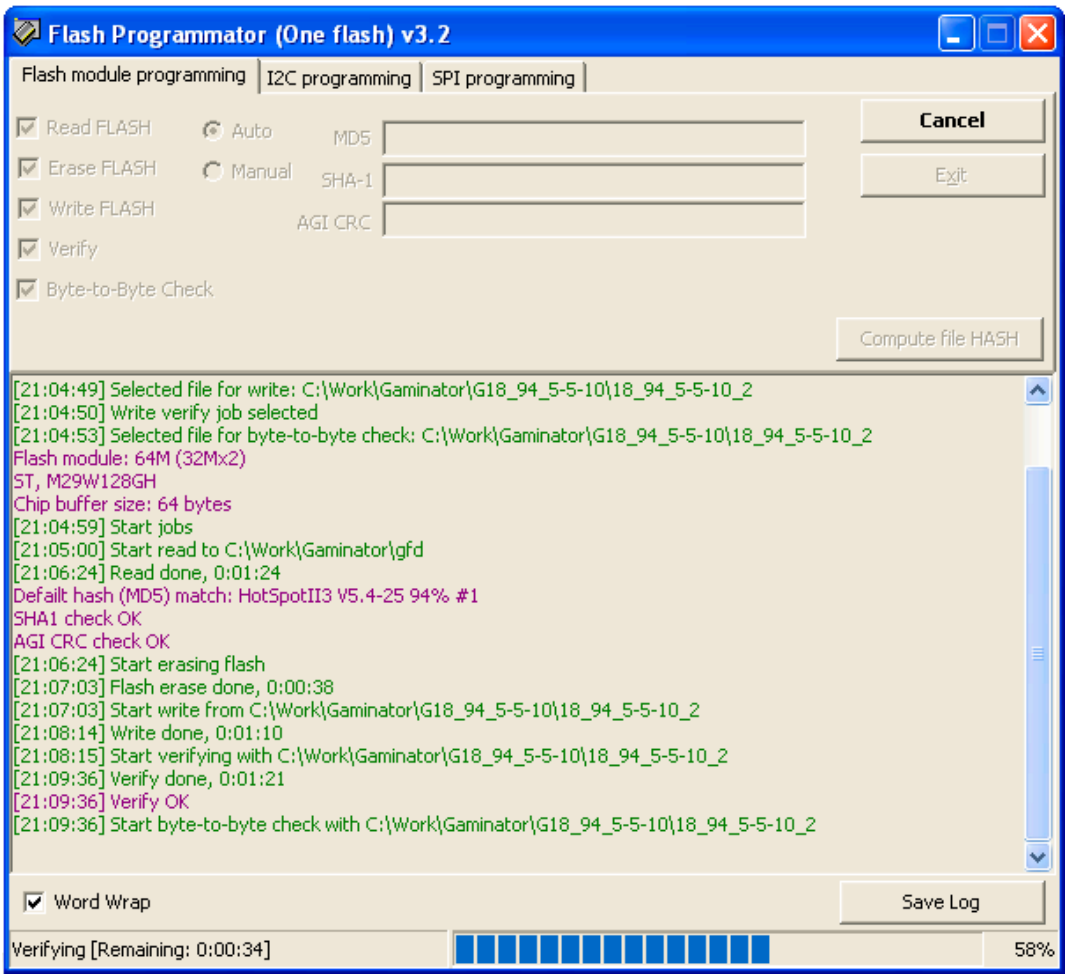
Jobs are executed sequentially, one after the other

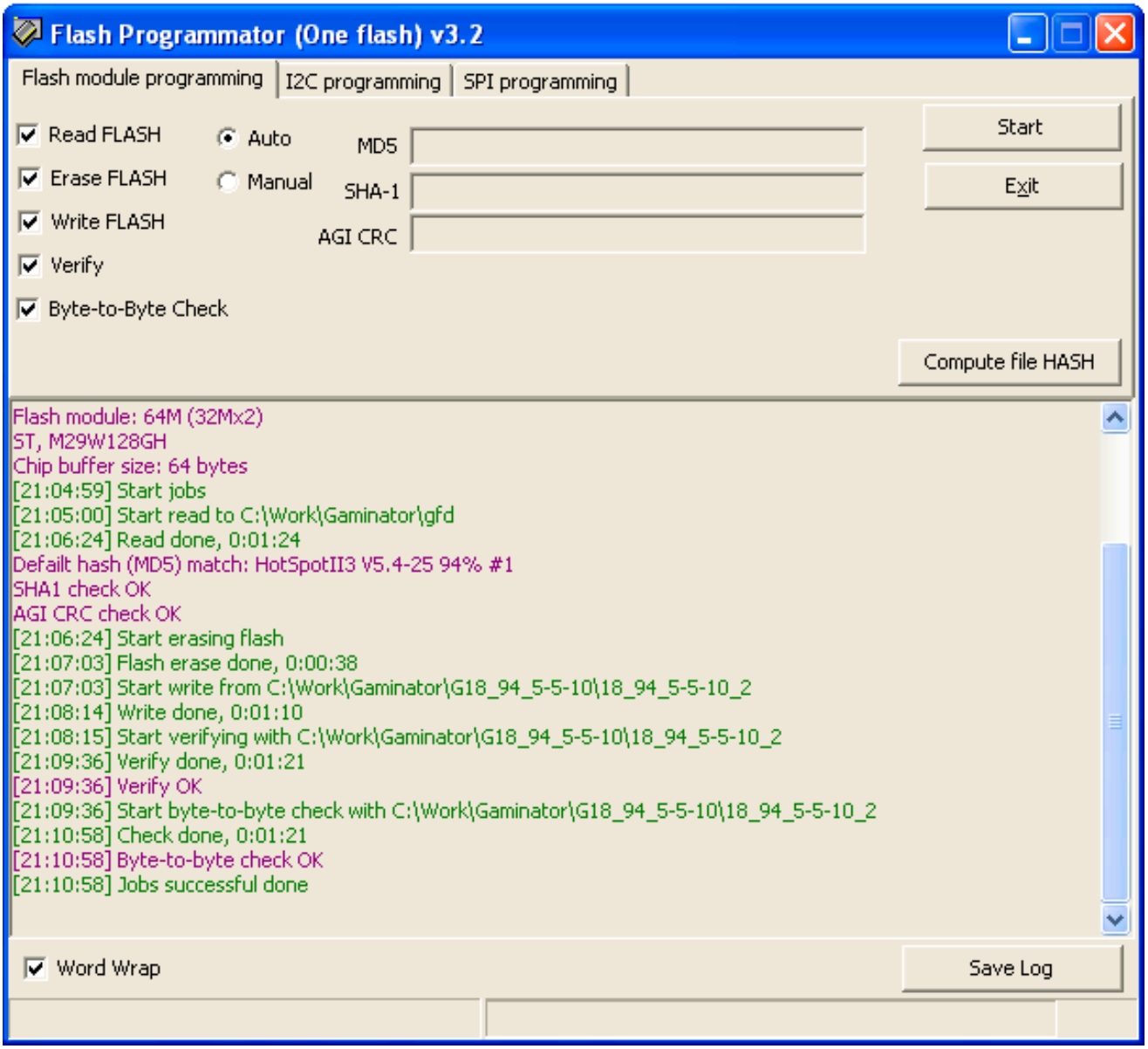



Flash Programmator (One flash) v3.2
Flash module programming
I2C programming
SPI programming
Read FLASH
Auto
Erase FLASH
Manual
Write FLASH
Verify
Byte-to-Byte Check
MD5
SHA-1
AGI CRC
Cancel
Exit
Compute file HASH
[21:04:33] Program started. Module types 5, vendors 5, chips 16, dumps hash's 657 in database.
[21:04:33] MD5 main hash, SHA1 check, AGI CRC check
[21:04:39] Selected file for read: C:\Work\Gaminator\gfd
[21:04:39] Erase job selected
[21:04:49] Selected file for write: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:04:50] Write verify job selected
[21:04:53] Selected file for byte-to-byte check: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Flash module: 64M (32Mx2)
ST, M29W128GH
Chip buffer size: 64 bytes
[21:04:59] Start jobs
[21:05:00] Start read to C:\Work\Gaminator\gfd
[21:06:24] Read done, 0:01:24
Defailt hash (MD5) match: HotSpotII3 V5.4-25 94% #1
SHA1 check OK
AGI CRC check OK
[21:06:24] Start erasing flash
Word Wrap
Save Log
Erasing [Bank0: Erasing, Bank1: Erasing]


Flash Programmator (One flash) v3.2
Flash module programming
I2C programming
SPI programming
Read FLASH
Auto
Erase FLASH
Manual
Write FLASH
Verify
Byte-to-Byte Check
MD5
SHA-1
AGI CRC
Cancel
Exit
Compute file HASH
[21:04:33] Program started. Module types 5, vendors 5, chips 16, dumps hash's 657 in database.
[21:04:33] MD5 main hash, SHA1 check, AGI CRC check
[21:04:39] Selected file for read: C:\Work\Gaminator\gfd
[21:04:39] Erase job selected
[21:04:49] Selected file for write: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:04:50] Write verify job selected
[21:04:53] Selected file for byte-to-byte check: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Flash module: 64M (32Mx2)
ST, M29W128GH
Chip buffer size: 64 bytes
[21:04:59] Start jobs
[21:05:00] Start read to C:\Work\Gaminator\gfd
[21:06:24] Read done, 0:01:24
Defailt hash (MD5) match: HotSpotII3 V5.4-25 94% #1
SHA1 check OK
AGI CRC check OK
[21:06:24] Start erasing flash
[21:07:03] Flash erase done, 0:00:38
[21:07:03] Start write from C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Word Wrap
Save Log
Writing [Remaining: 0:00:30]
56%


Flash Programmator (One flash) v3.2
Flash module programming
I2C programming
SPI programming
Read FLASH
Erase FLASH
Write FLASH
Verify
Byte-to-Byte Check
Auto
Manual
MD5
SHA-1
AGI CRC
Cancel
Exit
Compute file HASH
[21:04:33] MD5 main hash, SHA1 check, AGI CRC check
[21:04:39] Selected file for read: C:\Work\Gaminator\gfd
[21:04:39] Erase job selected
[21:04:49] Selected file for write: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:04:50] Write verify job selected
[21:04:53] Selected file for byte-to-byte check: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Flash module: 64M (32Mx2)
ST, M29W128GH
Chip buffer size: 64 bytes
[21:04:59] Start jobs
[21:05:00] Start read to C:\Work\Gaminator\gfd
[21:06:24] Read done, 0:01:24
Defailt hash (MD5) match: HotSpotII3 V5.4-25 94% #1
SHA1 check OK
AGI CRC check OK
[21:06:24] Start erasing flash
[21:07:03] Flash erase done, 0:00:38
[21:07:03] Start write from C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:08:14] Write done, 0:01:10
[21:08:15] Start verifying with C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Word Wrap
Save Log
Checking [Remaining: 0:00:43]
47%


Flash Programmator (One flash) v3.2
Flash module programming
I2C programming
SPI programming
Read FLASH
Erase FLASH
Write FLASH
Verify
Byte-to-Byte Check
Auto
Manual
MD5
SHA-1
AGI CRC
Cancel
Exit
Compute file HASH
[21:04:49] Selected file for write: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:04:50] Write verify job selected
[21:04:53] Selected file for byte-to-byte check: C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Flash module: 64M (32Mx2)
ST, M29W128GH
Chip buffer size: 64 bytes
[21:04:59] Start jobs
[21:05:00] Start read to C:\Work\Gaminator\gfd
[21:06:24] Read done, 0:01:24
Defailt hash (MD5) match: HotSpotII3 V5.4-25 94% #1
SHA1 check OK
AGI CRC check OK
[21:06:24] Start erasing flash
[21:07:03] Flash erase done, 0:00:38
[21:07:03] Start write from C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:08:14] Write done, 0:01:10
[21:08:15] Start verifying with C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
[21:09:36] Verify done, 0:01:21
[21:09:36] Verify OK
[21:09:36] Start byte-to-byte check with C:\Work\Gaminator\G18_94_5-5-10\18_94_5-5-10_2
Word Wrap
Save Log
Verifying [Remaining: 0:00:34]
58%



While doing the sounds reproduced, stored in a folder **[INSTALL_DIR]\Sound**

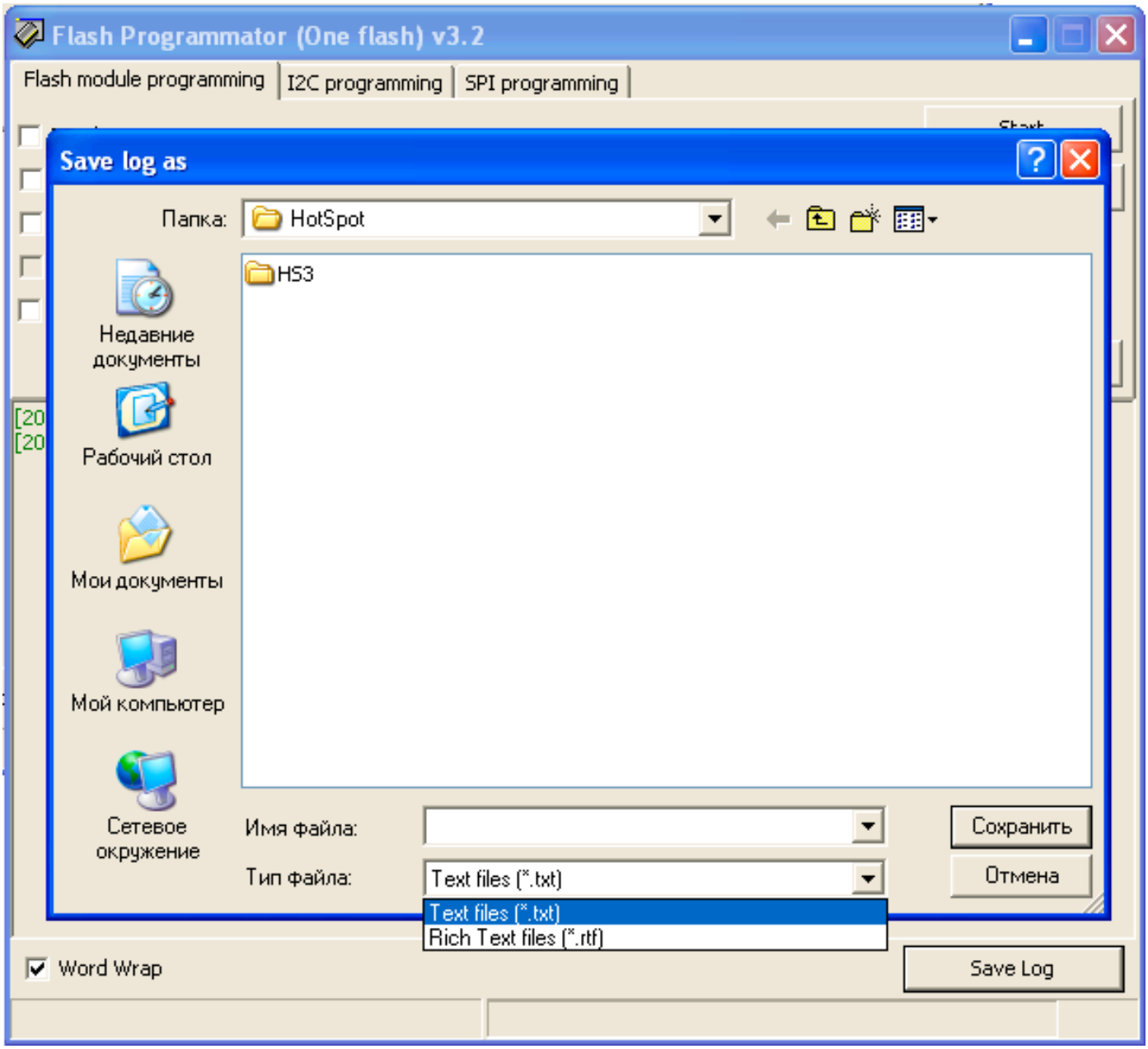
alarm.wav - failure of jobs

ding.wav - transition to the next job

tada.wav - successful completion of jobs

You can save the log by pressing **Save Log** button

Supported formats: TXT, RTF



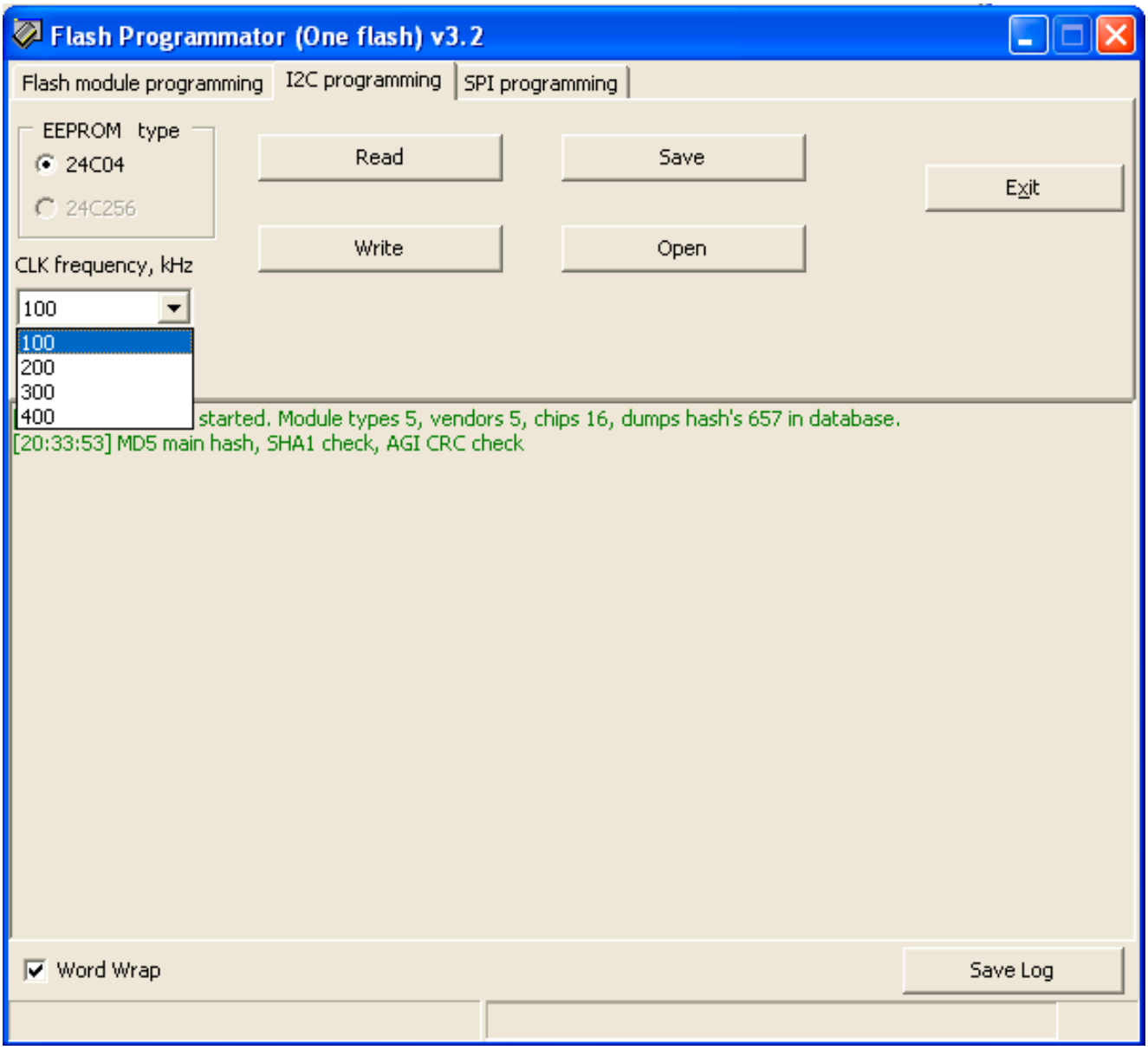
I2C programming tab

Read - read chip contents to the buffer

Write - write buffer contents to the chip

Open - open file to the buffer. Supported formats: BIN, Intel HEX

Save - save buffer contents to the file



You can choose the frequency of operation (100, 200, 300 or 400 kHz)

SPI programming tab

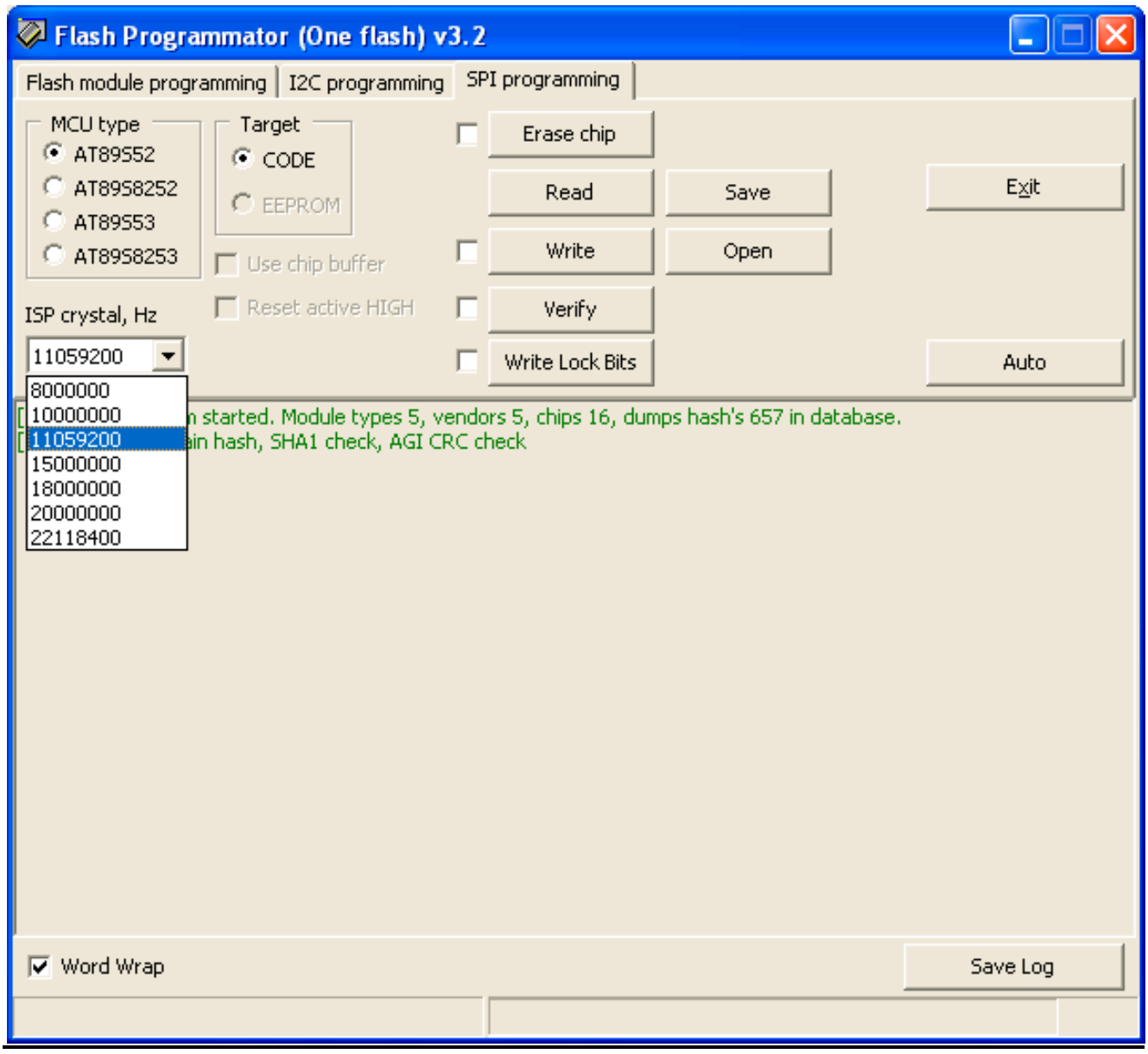
MCU type - select of MCU type

Target - working area of the microcontroller

Write Lock Bits - protection of the MCU contents for reading

Verify - verify content after write

Auto - sequential execution of operations marked



Вы можете выбрать частоту кварцевого резонатора, используемого в схеме. Заметьте, что иногда понижение реальной частоты позволит получить более стабильное программирование

The configuration INI file

Section [Modules]

```
Contains the modules description

[Modules]
module0 = ID,SIZE,TWO_BANKS
...
```

ID - the module ID in HEX

SIZE - the module capacity {16|32|64} MBytes

TWO_BANKS - two/one banks {1|0}

Section [Vendors]

Vendors description. Used for info only

```
[Vendors]
mnf0 = NAME,ID
...
```

NAME - Vendor

ID - vendor ID in HEX (ID0 from chip ID's)

Секция [Chips]

Chips description

```
[Chips]
chip0 = NAME,CHIP_ORG,WRITE_METHOD,USING_UBYPASS,ID0,ID1[,ID2,ID3]
...
```

NAME - chip name

CHIP_ORG - chip bus width :

- 0 - 8bit
- 1 - 16bit
- 2 - 32bit

WRITE_METHOD

- 0 - word by word
- 1 - use buffer
- 2 - use extended buffer
- 3 - use buffer size from CFI

USING_UBYPASS - chip can use short commands

0 - not using

1 - can used

IDx - chip ID's (started from Vendor ID). Number of bytes (2 <= IDs <= 4)

Section [Hash]

Hashes of dumps database

```
[Hash]
dump = NAME,HASH_MD5,HASH_SHA1,AGI_CRC
...
```

NAME - dump name

HASH_MD5 - 32 chars in HEX of MD5

HASH_SHA1 - 40 chars in HEX of SHA1

AGI_CRC - 8 chars in HEX of AGI CRC

The name value (in this case, **dump**) program is not processed, you can use any name

Section [Setup]

Contains settings check hashes in the database

```
[Setup]
MD5_check = M
SHA1_check = 1
AGI_CRC_check = 1
```

Parameter values can be {M|1|0}

One of the parameters must always be M (Main)

The validation method is that: The database hash sums sought the amount shown as Main. If successful, other types of search are checked against the hash referred to as "1". The hash labeled as "0" is not checked

Section [LED]

Contains a description of the color indication modes

```
[LED]
parameter = time
mode = COLOR,MODE[,COUNT]
...
```

Group frequency settings of flashes

Parameter	time (default) [1]	comment
ShortInterval	10	Time glow and pause mode fast flashing (Short)
LongInterval	70	Time glow and pause mode slow flashing (Long)
CounterActiveTime	20	Time glow in counting mode (Count)
CounterPassiveTime	20	Time pause in counting mode (Count)
CounterInterval	100	The time intervals between groups of flashes in the counting mode (Count)

[1] Real time is calculated as (time * 10) ms

Group of settings for each mode of the glow

COLOR - Green (G), Red (R) or Yellow (Y)
MODE - (L) - slow flashing, (S) - short flashing, (C) - defined number of flashes
COUNT - number of flashes in mode "C"

Default values

Mode	Value
StartReadFlash	G,C,1
FaultReadFlash	R,C,1
StartEraseFlash	G,C,2
FaultEraseFlash	R,C,2
StartWriteFlash	G,C,3
FaultWriteFlash	R,C,3
StartCheckFlash	G,C,4
FaultCheckFlash	R,C,4
StartBTBCheckFlash	G,C,5
FaultBTBCheckFlash	R,C,5
StartReadI2C	Y,S

FaultReadI2C	R,S
StartWriteI2C	Y,S
FaultWriteI2C	R,S
StartReadSPI	Y,S
FaultReadSPI	R,S
StartEraseSPI	Y,S
FaultEraseSPI	R,S
StartWriteSPI	Y,S
FaultWriteSPI	R,S
StartWriteLockbitsSPI	Y,S
FaultWriteLockbitsSPI	R,S

You can not add all the values in section **[LED]**, is sufficient to add only those parameters that specify the new values