[UEFI Practice] HII vfr file

原创 jiangwei0512 Posted on 2022-02-08 15:08:03 Read 1w Collection 57 Likes 25

Category Column: UEFI Development Basics Article Tags: this setup uefi

vfr file

The implementation of HII involves many different types of files, among which vfr file is the most important one, which constitutes the structural style of the interface. This article mainly refers to "edk-ii-vfr- specification .pdf", which is referred to as reference document below.

The UEFI code operates the UI interface, not directly using the vfr file, but a binary called **IFR** (Internal Forms Representation). The vfr file is just a string representation of IFR, which makes it convenient to write the interface using recognizable strings as codes, and then compile the IFR binary through the VFR compiler, which is finally used by the UEFI code.

Taking the Front Page that appeared earlier as an example, the structure of the interface in the figure depends on the vfr file MdeModulePkg\Application\UiApp\FrontPageVfr.Vfr, which builds the skeleton of the Front Page page. The content of the file is as follows:

AI generated projects 登录复制

```
#define FORMSET_GUID  { 0x9e0c30bc, 0x3f06, 0x4ba6, 0x82, 0x88, 0x9, 0x17, 0x9b, 0x85, 0x5d, 0xbe }

#define FRONT_PAGE_FORM_ID             0x1000

#define LABEL_FRANTPAGE_INFORMATION    0x1000
#define LABEL_END                      0xffff

formset
  guid     = FORMSET_GUID,
  title    = STRING_TOKEN(STR_FRONT_PAGE_TITLE),
  help     = STRING_TOKEN(STR_EMPTY_STRING ),
  classguid = FORMSET_GUID,

  form formid = FRONT_PAGE_FORM_ID,
       title  = STRING_TOKEN(STR_FRONT_PAGE_TITLE);

    banner
      title = STRING_TOKEN(STR_FRONT_PAGE_COMPUTER_MODEL),
      line  1,
      align left;

    banner
      title = STRING_TOKEN(STR_FRONT_PAGE_CPU_MODEL),
      line  2,
      align left;

    banner
      title = STRING_TOKEN(STR_FRONT_PAGE_CPU_SPEED),
      line  2,
      align right;

    banner
      title = STRING_TOKEN(STR_FRONT_PAGE_BIOS_VERSION),
      line  3,
      align left;

    banner
      title = STRING_TOKEN(STR_FRONT_PAGE_MEMORY_SIZE),
      line  3,
      align right;

    banner
      title = STRING_TOKEN(STR_CUSTOMIZE_BANNER_LINE4_LEFT),
      line  4,
      align left;

    banner
      title = STRING_TOKEN(STR_CUSTOMIZE_BANNER_LINE4_RIGHT),
      line  4,
      align right;

    banner
      title = STRING_TOKEN(STR_CUSTOMIZE_BANNER_LINE5_LEFT),
      line  5,
      align left;

    banner
      title = STRING_TOKEN(STR_CUSTOMIZE_BANNER_LINE5_RIGHT),
      line  5,
      align right;

    label LABEL_FRANTPAGE_INFORMATION;
    //
    // This is where we will dynamically add a Action type op-code to show
    // the platform information.
    //
    label LABEL_END;

  endform;
```

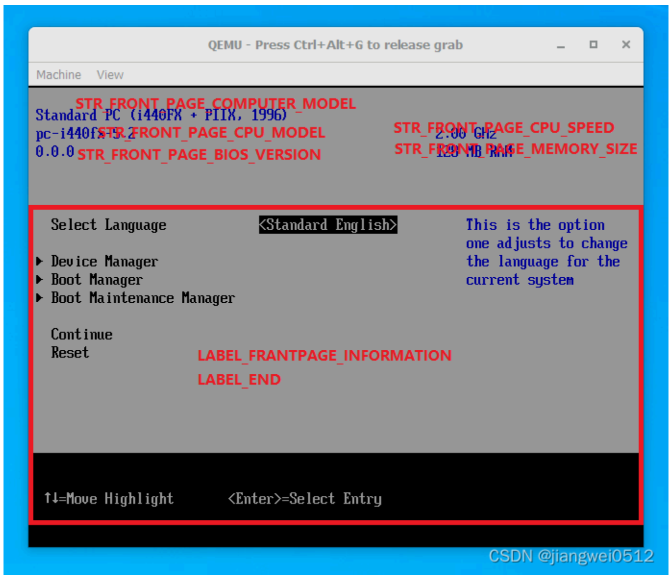
收起 ^

The basic structure of it is as follows:

bash AI generated projects 登录复制

```bash
STR_FRONT_PAGE_COMPUTER_MODEL
STR_FRONT_PAGE_CPU_MODEL                                    STR_FRONT_PAGE_CPU_SPEED
STR_FRONT_PAGE_BIOS_VERSION                                 STR_FRONT_PAGE_MEMORY_SIZE
STR_CUSTOMIZE_BANNER_LINE4_LEFT                             STR_CUSTOMIZE_BANNER_LINE4_RIGHT
STR_CUSTOMIZE_BANNER_LINE5_LEFT                             STR_CUSTOMIZE_BANNER_LINE5_RIGHT
LABEL_FRANTPAGE_INFORMATION
LABEL_END
```

Corresponding to the Front Page interface (the fourth and fifth lines are not used):


QEMU - Press Ctrl+Alt+G to release grab
Machine View
STR_FRONT_PAGE_COMPUTER_MODEL
STR_FRONT_PAGE_CPU_MODEL
STR_FRONT_PAGE_BIOS_VERSION
STR_FRONT_PAGE_CPU_SPEED
STR_FRONT_PAGE_MEMORY_SIZE
LABEL_FRANTPAGE_INFORMATION
LABEL_END
CSDN @jiangwei0512

```
Standard PC (i440FX + PIIX, 1996)
pc-i440fx-5.2                                   2.00 GHz
0.0.0                                           128 MB RAM

Select Language            <Standard English>    This is the option
                                                 one adjusts to change
▶ Device Manager                                 the language for the
▶ Boot Manager                                   current system
▶ Boot Maintenance Manager

Continue
Reset

↑↓=Move Highlight        <Enter>=Select Entry
```

The red font part is the tag defined in the vfr file, and part of the displayed string is defined in the uni file, and the red frame part is implemented by code, so the vfr file constitutes the static framework of the interface, and the code can be dynamically modified through the tags defined in the vfr file.

The vfr file is compiled by a specific VFR compiler, and the final result is an intermediate file. The intermediate file is a c file (FrontPageVfr.c) or an hpk file, which contains a variable used to represent the VFR resource:

c AI generated projects 登录复制 run

```c
unsigned char FrontPageVfrBin[] = {
  // ARRAY LENGTH，0x143加上下述的头部长度，就是0x147
  0x47,  0x01,  0x00,  0x00,

  // PACKAGE HEADER，对应的是EFI_HII_FORM_PACKAGE_HDR，0x13F加上下述的头部长度，就是0x143，占据3个字节；第4个字节0x02表示HII类型Form
  0x43,  0x01,  0x00,  0x02,

  // PACKAGE DATA，319个字节，十六进制就是0x13F，注释已经加上

  // 第1个操作码，对应结构体EFI_IFR_FORM_SET
  0x0E,  // EFI_IFR_FORM_SET_OP，它的值就是0xE
  0xA7,  // 前面7位表示长度，即0x27=39个字节，总长度到第二个FORMSET_GUID为止刚好39个字节；第8位表示的是scope，该位为1则表示开始一个新的scope
  // Guid: FORMSET_GUID
  0xBC,  0x30,  0x0C,  0x9E,  0x06,  0x3F,  0xA6,  0x4B,  0x82,  0x88,  0x09,  0x17,  0x9B,  0x85,  0x5D,  0xBE,
  0x02,  0x00,  // FormSetTitle: 字符串Token，在AutoGen中定义，对应的是STR_FRONT_PAGE_TITLE，值就是0x0002
  0x0C,  0x00,  // Help: 字符串Token，在AutoGen中定义，对应的是STR_EMPTY_STRING，值就是0x000C
  0x01, // Flags:
  // ClassGuid: FORMSET_GUID
  0xBC,  0x30,  0x0C,  0x9E,  0x06,  0x3F,  0xA6,  0x4B,  0x82,  0x88,  0x09,  0x17,  0x9B,  0x85,  0x5D,  0xBE,

  // 第2个操作码，对应结构体EFI_IFR_DEFAULTSTORE
  0x5C,  // 操作码EFI_IFR_DEFAULTSTORE_OP
  0x06,  // 长度6个字节
  0x00,  0x00,  // DefaultName: 字符串Token，似乎并不存在
  0x00,  0x00,  // DefaultId: 表示EFI_HII_DEFAULT_CLASS_STANDARD

  // 第3个操作码，对应结构体EFI_IFR_DEFAULTSTORE
  0x5C,  // 操作码EFI_IFR_DEFAULTSTORE_OP
  0x06,  // 长度6个字节
  0x00,  0x00,  // DefaultName: 字符串Token，似乎并不存在
  0x01,  0x00,  // DefaultId: 表示EFI_HII_DEFAULT_CLASS_MANUFACTURING
  // 前面两个操作码并没有在vfr文件中声明，但是却创建了。

  // 第4个操作码，对应结构体EFI_IFR_FORM
  0x01,  // 操作码示EFI_IFR_FORM_OP
  0x86,  // 长度6个字节，新建scope
  0x00,  0x10,  // FormId: 对应FRONT_PAGE_FORM_ID
  0x02,  0x00,  // FormTitle: 对应字符串Token，STR_FRONT_PAGE_TITLE

  // 第5个操作码，对应结构体EFI_IFR_GUID_BANNER
  0x5F,  // 操作码示EFI_IFR_GUID_OP
  0x18,  // 长度24个字节，其中16个是GUID
  // EFI_IFR_TIANO_GUID，表示GUIDed opcodes defined for EDKII implementation，定义在MdeModulePkg\Include\Guid\MdeModuleHii.h
  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF, 0x48,  0xCE,
  0x01,  // ExtendOpCode，0x01表示EFI_IFR_EXTEND_OP_BANNER
  0x03,  0x00,  // Title，对应字符串Token，0x0003对应的是STR_FRONT_PAGE_COMPUTER_MODEL
  0x01,  0x00,  // LineNumber
  0x00,  // Alignment

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x04,  0x00,  0x02,  0x00,  0x00,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x05,  0x00,  0x02,  0x00,  0x02,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x07,  0x00,  0x03,  0x00,  0x00,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x06,  0x00,  0x03,  0x00,  0x02,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x0E,  0x00,  0x04,  0x00,  0x00,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x0F,  0x00,  0x04,  0x00,  0x02,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x10,  0x00,  0x05,  0x00,  0x00,

  0x5F,  0x18,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x01,  0x11,  0x00,  0x05,  0x00,  0x02,

  // 第14个操作码，对应结构体EFI_IFR_GUID_LABEL
  0x5F,  // 操作码示EFI_IFR_GUID_OP
  0x15,  // 长度21个字节，其中16个是GUID
  // EFI_IFR_TIANO_GUID，表示GUIDed opcodes defined for EDKII implementation，定义在MdeModulePkg\Include\Guid\MdeModuleHii.h
  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,  0x48,  0xCE,
  0x00,  // ExtendOpCode，0x00表示EFI_IFR_EXTEND_OP_LABEL
  0x00,  0x10,  // Label Number.

  0x5F,  0x15,  0x35,  0x17,  0x0B,  0x0F,  0xA0,  0x87,  0x93,  0x41,  0xB2,  0x66,  0x53,  0x8C,  0x38,  0xAF,
  0x48,  0xCE,  0x00,  0xFF,  0xFF,

  0x29,  0x02,

  0x29,  0x02
};
```

收起 ^

These data are the IFR binaries, which are `HiiAddPackages()` installed via:

c AI generated projects 登录复制 run

```c
  //
  // Publish our HII data
  //
  gFrontPagePrivate.HiiHandle = HiiAddPackages (
                                  &mFrontPageGuid,
                                  gFrontPagePrivate.DriverHandle,
                                  FrontPageVfrBin,
                                  UiAppStrings,
                                  NULL
                                  );
```

收起 ^

This way you can `gFrontPagePrivate.HiiHandle` access the installed resources through.

Like uni files, vfr files can be used in two ways: one is to define variables in the intermediate file, and the other is to define the binary directly. The above code uses the intermediate file method, removing the first 4 bytes, and the remaining part corresponds to a structure:

c AI generated projects 登录复制 run

```c
///
/// The header found at the start of each package.
///
typedef struct {
  UINT32  Length:24;
  UINT32  Type:8;
  // UINT8  Data[...];
} EFI_HII_PACKAGE_HEADER;
///
/// The Form package is used to carry form-based encoding data.
///
typedef struct _EFI_HII_FORM_PACKAGE_HDR {
  EFI_HII_PACKAGE_HEADER       Header;
  // EFI_IFR_OP_HEADER         OpCodeHeader;
  // More op-codes follow
} EFI_HII_FORM_PACKAGE_HDR;
```

收起 ^

`EFI_HII_PACKAGE_HEADER` The following types are available `Type` , which represent all HII types, such as structure, font, string, etc.:

c AI generated projects 登录复制 run

```c
//
// Value of HII package type
//
#define EFI_HII_PACKAGE_TYPE_ALL             0x00
#define EFI_HII_PACKAGE_TYPE_GUID            0x01
#define EFI_HII_PACKAGE_FORMS                0x02
#define EFI_HII_PACKAGE_STRINGS              0x04
#define EFI_HII_PACKAGE_FONTS                0x05
#define EFI_HII_PACKAGE_IMAGES               0x06
#define EFI_HII_PACKAGE_SIMPLE_FONTS         0x07
```

```
#define EFI_HII_PACKAGE_DEVICE_PATH          0x08
#define EFI_HII_PACKAGE_KEYBOARD_LAYOUT      0x09
#define EFI_HII_PACKAGE_ANIMATIONS           0x0A
#define EFI_HII_PACKAGE_END                  0xDF
#define EFI_HII_PACKAGE_TYPE_SYSTEM_BEGIN    0xE0
#define EFI_HII_PACKAGE_TYPE_SYSTEM_END      0xFF
```

收起 ^

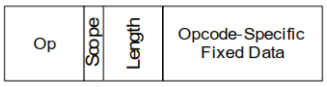
For the structure described by the vfr file, its type is of course `EFI_HII_PACKAGE_FORMS` and the value is 2.

The contents after the header are operation codes, and their structure is as follows:

c AI generated projects 登录复制 run

```c
typedef struct _EFI_IFR_OP_HEADER {
  UINT8                    OpCode;
  UINT8                    Length:7;
  UINT8                    Scope:1;
} EFI_IFR_OP_HEADER;
```

This structure is also of indefinite length, followed by data. Depending on the opcode, the data is different, as shown in the following figure:



The opcodes are listed in IFR opcodes ; the length includes the entire variable-length structure, that is, the length of the header; `Scope` if it is 1, it means that a new scope is opened until it is encountered `EFI_IFR_END_OP` .

Language Basics

`vfr` The file is also described using EBNF, which will not be introduced in detail here, but only the basic language foundation will be briefly explained.

- Annotate by `//` ;
- Support predefined instructions:
  - `#define` : It is similar to the usage in C language, that is, defining macros;
  - `#include` : Can include C language header files;
  - `#pragma` : It is used in C language header files. For example, for the structure in the header file, its alignment can be set.
- Supports basic data types and HII-specific data types, as well as structures:
  - `UINT8` , `UINT16` , `UINT32` , `UINT64` , `BOOLEAN` wait;
  - `EFI_STRING_ID` , `EFI_HII_DATA` , `EFI_HII_TIME` , `EFI_HII_REF` wait.

VFR Components

This article introduces commonly used components, such as buttons, selection boxes, labels, etc. These will be combined with IFR operation code descriptions. This section describes the most basic components of VFR.

form set

There is only one vfr file in each vfr file `formset` , which constitutes the main body of the interface. Other interface components are inside it. Its structure is as follows:

json AI generated projects 登录复制

```json
formset
  guid      = TCG_CONFIG_FORM_SET_GUID, // GUID
  title     = STRING_TOKEN(STR_TPM_TITLE), // uni文件中定义的标记
  help      = STRING_TOKEN(STR_TPM_HELP), // uni文件中定义的标记
  classguid = EFI_HII_PLATFORM_SETUP_FORMSET_GUID, // GUID, 可选
  class     = EFI_NETWORK_DEVICE_CLASS, // 数值, 可选
  subclass  = 0x03, // 数值, 可选

  // 剩余组件写在这里

endformset;
```

收起 ^

The remaining components can be images, variables, DisableIf, SuppressIf, extensions, etc., which can be expressed in BNF as follows:

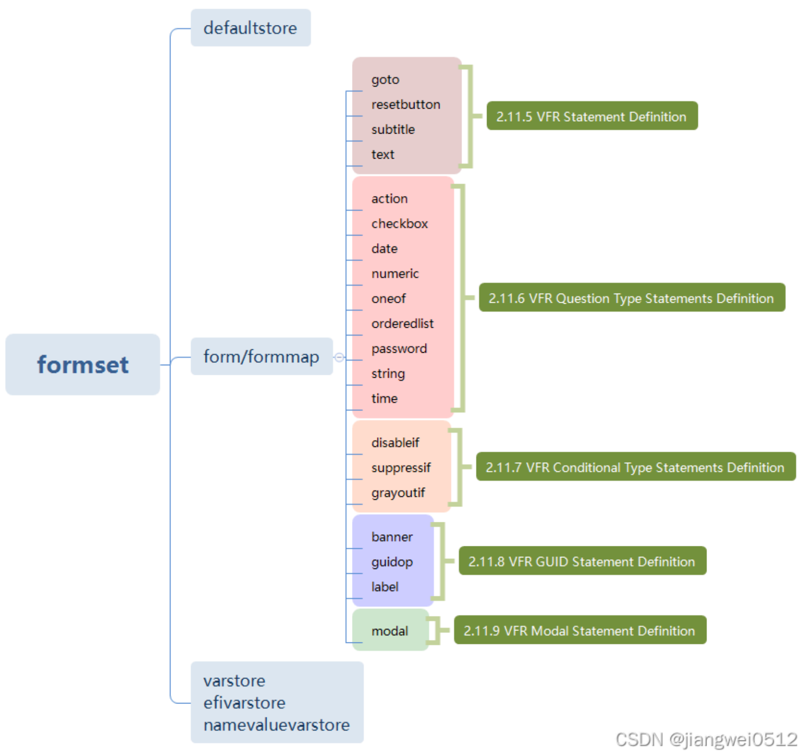
json AI generated projects 登录复制

```json
vfrFormSetList ::=
(
    vfrFormDefinition
    | vfrFormMapDefinition
    | vfrStatementImage
    | vfrStatementVarStoreLinear
    | vfrStatementVarStoreEfi
    | vfrStatementVarStoreNameValue
    | vfrStatementDefaultStore
    | vfrStatementDisableIfFormSet
    | vfrStatementSuppressIfFormSet
    | vfrStatementExtension
)*
```

收起 ^

`formset` And `endformset` pairing use.

The specific hierarchical relationship is as follows (the vfr components that can be found in the current source code):



Note that only the components that exist in the current EDK code are included here, there are many more that are not listed.

form

`formset` There can be several of them `form` , here is an example:

json AI generated projects 登录复制

```json
form formid = FORM_MAIN_ID,
  title = STRING_TOKEN(STR_FORM_MAIN_TITLE);
endform;
```

`formid` `formset` Must be unique within a ; `title` tag from the uni file.

variable

There are three variables used by VFR, namely `varstore` , , `efivarstore` and `namevaluevarstore` . Here is an example:

c AI generated projects 登录复制 run

```c
//
// Define a Buffer Storage (EFI_IFR_VARSTORE)
//
varstore DRIVER_SAMPLE_CONFIGURATION,     // 数据结构，在头文件中定义
  varid = CONFIGURATION_VARSTORE_ID,      // 变量ID，可选，在创建操作码的函数中会用到，比如HiiCreateOneOfOpCode()
  name  = MyIfrNVData,                    // 变量名，CHAR16     VariableName[] = L"MyIfrNVData";
  guid  = DRIVER_SAMPLE_FORMSET_GUID;     // 变量GUID，跟变量名共同确定了UEFI变量

//
// Define a EFI variable Storage (EFI_IFR_VARSTORE_EFI)
//
efivarstore MY_EFI_VARSTORE_DATA,  // 数据结构，在头文件中定义
  attribute = EFI_VARIABLE_BOOTSERVICE_ACCESS | EFI_VARIABLE_NON_VOLATILE,  // UEFI变量属性
  name  = MyEfiVar,  // 变量名，CHAR16     MyEfiVar[] = L"MyEfiVar";
  guid  = DRIVER_SAMPLE_FORMSET_GUID;  // 变量GUID，跟变量名共同确定了UEFI变量

//
// Define a Name/Value Storage (EFI_IFR_VARSTORE_NAME_VALUE)
//
namevaluevarstore MyNameValueVar,                // Define storage reference name in vfr
  name = STRING_TOKEN(STR_NAME_VALUE_VAR_NAME0), // Define Name list of this storage, refer it by MyNameValueVar[0]
  name = STRING_TOKEN(STR_NAME_VALUE_VAR_NAME1), // Define Name list of this storage, refer it by MyNameValueVar[1]
  name = STRING_TOKEN(STR_NAME_VALUE_VAR_NAME2), // Define Name list of this storage, refer it by MyNameValueVar[2]
  guid = DRIVER_SAMPLE_FORMSET_GUID;             // GUID of this Name/Value storage
```

收起 ^

The above variables can have default values set, and the default values can also be different, which can `defaultstore` be achieved by:

json AI generated projects 登录复制

```json
defaultstore MyStandardDefault,
  prompt      = STRING_TOKEN(STR_STANDARD_DEFAULT_PROMPT),
  attribute   = 0x0000;                         // Default ID: 0000 standard default

defaultstore MyManufactureDefault,
  prompt      = STRING_TOKEN(STR_MANUFACTURE_DEFAULT_PROMPT),
  attribute   = 0x0001;                         // Default ID: 0001 manufacture default
```

Usage example, you can use it in the above 3 variables:

json AI generated projects 登录复制

```
    numeric varid   = MyIfrNVData.HowOldAreYouInYears,  // varstore
            prompt  = STRING_TOKEN(STR_NUMERIC_STEP_PROMPT),
            help    = STRING_TOKEN(STR_NUMERIC_HELP2),
            minimum = 0,
            maximum = 243,
            step    = 1,
            default = 18, defaultstore = MyStandardDefault,     // This is standard default value
            default = 19, defaultstore = MyManufactureDefault,  // This is manufacture default value

    endnumeric;

    numeric varid   = MyEfiVar.Field8,                    // Reference of EFI variable storage
            questionid  = 0x1111,
            prompt  = STRING_TOKEN(STR_TALL_HEX_PROMPT),
            help    = STRING_TOKEN(STR_NUMERIC_HELP1),
            flags   = DISPLAY_UINT_HEX | INTERACTIVE,     // Display in HEX format (if not specified, default is in decimal format)
            minimum = 0,
            maximum = 250,
            default = 18, defaultstore = MyStandardDefault,     // This is standard default value
            default = 19, defaultstore = MyManufactureDefault,  // This is manufacture default value

    endnumeric;

    //
    // Define numeric using Name/Value Storage
    //
    numeric varid   = MyNameValueVar[0],     // This numeric take NameValueVar0 as storage
            prompt  = STRING_TOKEN(STR_NAME_VALUE_VAR_NAME0),
            help    = STRING_TOKEN(STR_NAME_VALUE_VAR_NAME0_HELP),
            //
            // Size should be defined for numeric when use Name/Value storage
            // Valid value for numerice size are: NUMERIC_SIZE_1, NUMERIC_SIZE_2, NUMERIC_SIZE_4 and NUMERIC_SIZE_8
            //
            flags   = NUMERIC_SIZE_1,        // Size of this numeric is 1 byte
            minimum = 0,
            maximum = 0xff,
            step    = 0,
            locked,
            default = 16, defaultstore = MyStandardDefault,     // This is standard default value
            default = 17, defaultstore = MyManufactureDefault,  // This is manufacture default value
    endnumeric;
```

收起 ^

control

There are three statements that control the display:

json AI generated projects 登录复制

```
    disableif ideqval MyIfrNVData.SuppressGrayOutSomething == 0x2;
      orderedlist
        varid       = MyIfrNVData.OrderedList,
        prompt      = STRING_TOKEN(STR_TEST_OPCODE),
        help        = STRING_TOKEN(STR_TEXT_HELP),
        flags       = RESET_REQUIRED,
        option text = STRING_TOKEN(STR_ONE_OF_TEXT1), value = 3, flags = 0;
        option text = STRING_TOKEN(STR_ONE_OF_TEXT2), value = 2, flags = 0;
        option text = STRING_TOKEN(STR_ONE_OF_TEXT3), value = 1, flags = 0;
        default     = {1,2,3},
      endlist;
    endif;

    grayoutif NOT ideqval MyIfrNVData.SuppressGrayOutSomething == 0x1;
      suppressif questionref(MyOneOf) == 0x0;

        checkbox varid   = MyIfrNVData.ChooseToActivateNuclearWeaponry,
                 prompt   = STRING_TOKEN(STR_CHECK_BOX_PROMPT),
                 help     = STRING_TOKEN(STR_CHECK_BOX_HELP),
                 //
                 // CHECKBOX_DEFAULT indicate this checkbox is marked with EFI_IFR_CHECKBOX_DEFAULT
                 // CHECKBOX_DEFAULT_MFG indicate EFI_IFR_CHECKBOX_DEFAULT_MFG.
                 //
                 flags    = CHECKBOX_DEFAULT | CHECKBOX_DEFAULT_MFG,
                 default  = TRUE,
        endcheckbox;
      endif;
    endif;
```

收起 ^

`disableif` Indicates not used, `suppressif` indicates not displayed, `grayoutif` indicates displayed but cannot be operated (cannot be selected).

There are also control jump statements:

json AI generated projects 登录复制

```
      goto FORM_BOOT_DEL_ID,
           prompt = STRING_TOKEN(STR_FORM_BOOT_DEL_TITLE),
           help = STRING_TOKEN(STR_FORM_BOOT_IMMEDIATE_HELP),
           flags = INTERACTIVE,
           key = FORM_BOOT_DEL_ID;
```

`FORM_BOOT_DEL_ID` Corresponding to `form` one `formid` :

json AI generated projects 登录复制

```
  form formid = FORM_BOOT_DEL_ID,
       title = STRING_TOKEN(STR_FORM_BOOT_DEL_TITLE);
```

```
        label FORM_BOOT_DEL_ID;
        label LABEL_END;
    endform;
```

expression

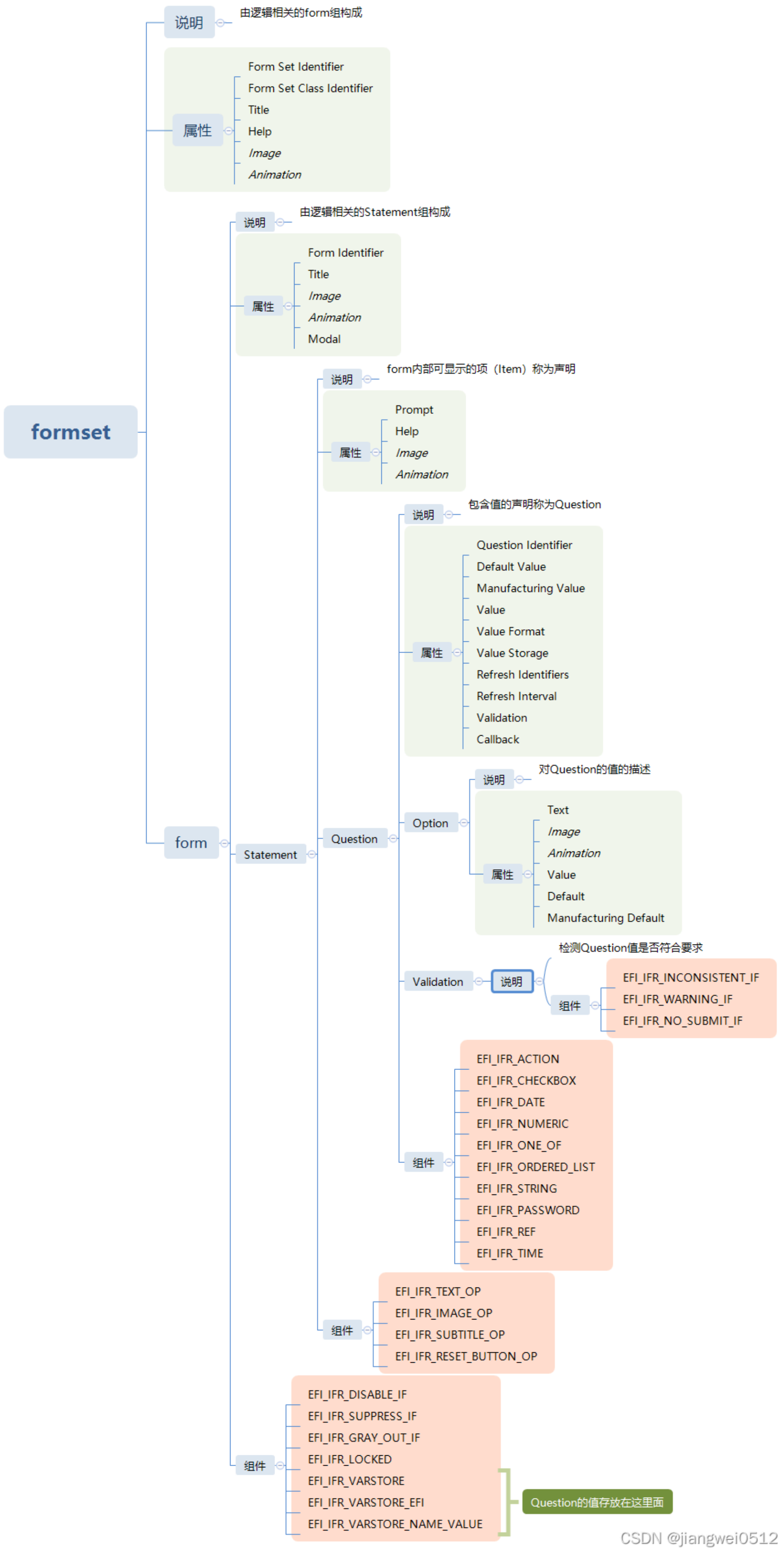
VFR can contain expressions, such as `OR` ,, `AND` and `NOT` so on, and its function is similar to that in C language.

IFR Opcodes

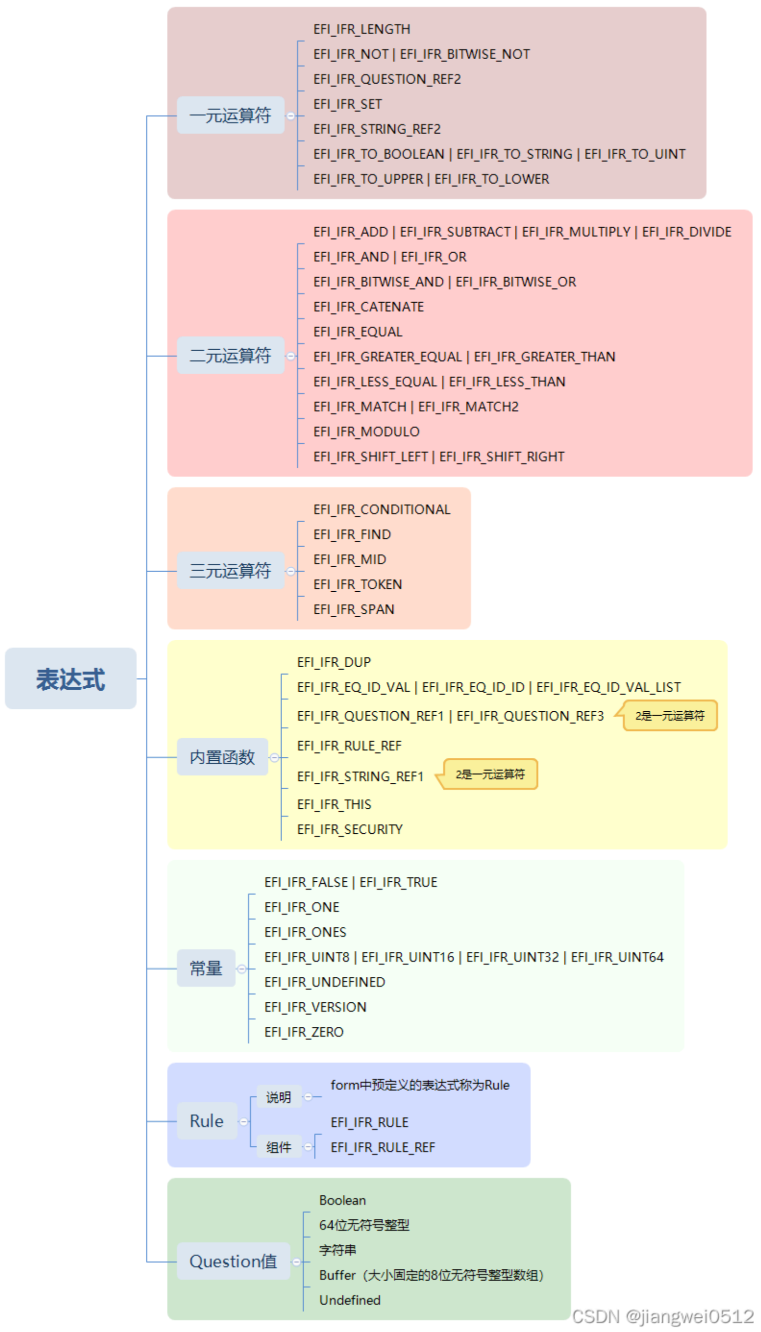
Operation Code	value	describe
EFI_IFR_FORM_OP	0x01	Form
EFI_IFR_SUBTITLE_OP	0x02	Subtitle statement
EFI_IFR_TEXT_OP	0x03	Static text/image statement
EFI_IFR_IMAGE_OP	0x04	Static image
EFI_IFR_ONE_OF_OP	0x05	One-of question
EFI_IFR_CHECKBOX_OP	0x06	Boolean question
EFI_IFR_NUMERIC_OP	0x07	Numeric question
EFI_IFR_PASSWORD_OP	0x08	Password string question
EFI_IFR_ONE_OF_OPTION_OP	0x09	Option
EFI_IFR_SUPPRESS_IF_OP	0x0A	Suppress if conditional
EFI_IFR_LOCKED_OP	0x0B	Marks statement/question as locked
EFI_IFR_ACTION_OP	0x0C	Button question
EFI_IFR_RESET_BUTTON_OP	0x0D	Reset button statement
EFI_IFR_FORM_SET_OP	0x0E	Form set
EFI_IFR_REF_OP	0x0F	Cross-reference statement
EFI_IFR_NO_SUBMIT_IF_OP	0x10	Error checking conditional
EFI_IFR_INCONSISTENT_IF_OP	0x11	Error checking conditional
EFI_IFR_EQ_ID_VAL_OP	0x12	Return true if question value equals UINT16
EFI_IFR_EQ_ID_ID_OP	0x13	Return true if question value equals another question value
EFI_IFR_EQ_ID_VAL_LIST_OP	0x14	Return true if question value is found in list of UINT16s
EFI_IFR_AND_OP	0x15	Push true if both sub-expressions returns true
EFI_IFR_OR_OP	0x16	Push true if either sub-expressions returns true
EFI_IFR_NOT_OP	0x17	Push false if sub-expression returns true, otherwise return true
EFI_IFR_RULE_OP	0x18	Create rule in current form
EFI_IFR_GRAY_OUT_IF_OP	0x19	Nested statements, questions or options will not be selectable if expression returns true
EFI_IFR_DATE_OP	0x1A	Date question
EFI_IFR_TIME_OP	0x1B	Time question
EFI_IFR_STRING_OP	0x1C	String question
EFI_IFR_REFRESH_OP	0x1D	Interval for refreshing a question
EFI_IFR_DISABLE_IF_OP	0x1E	Nested statements, questions or options will not be processed if expression returns true
EFI_IFR_ANIMATION_OP	0x1F	Animation associated with question statement, form or form set
EFI_IFR_TO_LOWER_OP	0x20	Convert a string on the expression stack to lower case
EFI_IFR_TO_UPPER_OP	0x21	Convert a string on the expression stack to upper case
EFI_IFR_MAP_OP	0x22	Convert one value to another by selecting a match from a list
EFI_IFR_ORDERED_LIST_OP	0x23	Set question
EFI_IFR_VARSTORE_OP	0x24	Define a buffer-style variable storage
EFI_IFR_VARSTORE_NAME_VALUE_OP	0x25	Define a name/value style variable storage
EFI_IFR_VARSTORE_EFI_OP	0x26	Define a UEFI variable style variable storage
EFI_IFR_VARSTORE_DEVICE_OP	0x27	Specify the device path to use for variable storage
EFI_IFR_VERSION_OP	0x28	Push the revision level of the UEFI Specification to which this Forms Processor is compliant
EFI_IFR_END_OP	0x29	Marks end of scope
EFI_IFR_MATCH_OP	0x2A	Push TRUE if string matches a pattern
EFI_IFR_GET_OP	0x2B	Return a stored value
EFI_IFR_SET_OP	0x2C	Change a stored value
EFI_IFR_READ_OP	0x2D	Provides a value for the current question or default
EFI_IFR_WRITE	0x2E	Change a value for the current question
EFI_IFR_EQUAL_OP	0x2F	Push TRUE if two expressions are equal

Operation Code	value	describe
EFI_IFR_NOT_EQUAL_OP	0x30	Push TRUE if two expressions are not equal
EFI_IFR_GREATER_THAN_OP	0x31	Push TRUE if one expression is greater than another expression
EFI_IFR_GREATER_EQUAL_OP	0x32	Push TRUE if one expression is greater than or equal to another expression
EFI_IFR_LESS_THAN_OP	0x33	Push TRUE if one expression is less than another expression
EFI_IFR_LESS_EQUAL_OP	0x34	Push TRUE if one expression is less than or equal to another expression
EFI_IFR_BITWISE_AND_OP	0x35	Bitwise-AND two unsigned integers and push the result
EFI_IFR_BITWISE_OR_OP	0x36	Bitwise-OR two unsigned integers and push the result
EFI_IFR_BITWISE_NOT_OP	0x37	Bitwise-NOT an unsigned integer and push the result
EFI_IFR_SHIFT_LEFT_OP	0x38	Shift an unsigned integer left by a number of bits and push the result
EFI_IFR_SHIFT_RIGHT_OP	0x39	Shift an unsigned integer right by a number of bits and push the result
EFI_IFR_ADD_OP	0x3A	Add two unsigned integers and push the result
EFI_IFR_SUBTRACT_OP	0x3B	Subtract two unsigned integers and push the result
EFI_IFR_MULTIPLY_OP	0x3C	Multiply two unsigned integers and push the result
EFI_IFR_DIVIDE_OP	0x3D	Divide one unsigned integer by another and push the result
EFI_IFR_MODULO_OP	0x3E	Divide one unsigned integer by another and push the remainder
EFI_IFR_RULE_REF_OP	0x3F	Evaluate a rule
EFI_IFR_QUESTION_REF1_OP	0x40	Push a question's value
EFI_IFR_QUESTION_REF2_OP	0x41	Push a question's value
EFI_IFR_UINT8_OP EFI_IFR_UINT16_OP EFI_IFR_UINT32_OP EFI_IFR_UINT64_OP	0x42 0x43 0x44 0x45	Push an 8-bit/16-bit/32-bit/64-bit unsigned integer
EFI_IFR_TRUE_OP	0x46	Push a boolean TRUE.
EFI_IFR_FALSE_OP	0x47	Push a boolean FALSE
EFI_IFR_TO_UINT_OP	0x48	Convert expression to an unsigned integer
EFI_IFR_TO_STRING_OP	0x49	Convert expression to a string
EFI_IFR_TO_BOOLEAN_OP	0x4A	Convert expression to a boolean
EFI_IFR_MID_OP	0x4B	Extract portion of string or buffer
EFI_IFR_FIND_OP	0x4C	Find a string in a string
EFI_IFR_TOKEN_OP	0x4D	Extract a delimited byte or character string from buffer or string
EFI_IFR_STRING_REF1_OP	0x4E	Push a string
EFI_IFR_STRING_REF2_OP	0x4F	Push a string
EFI_IFR_CONDITIONAL_OP	0x50	Duplicate one of two expressions depending on result of the first expression
EFI_IFR_QUESTION_REF3_OP	0x51	Push a question's value from a different form
EFI_IFR_ZERO_OP	0x52	Push a zero
EFI_IFR_ONE_OP	0x53	Push a one
EFI_IFR_ONES_OP	0x54	Push a 0xFFFFFFFFFFFFFFFF
EFI_IFR_UNDEFINED_OP	0x55	Push Undefined
EFI_IFR_LENGTH_OP	0x56	Push length of buffer or string
EFI_IFR_DUP_OP	0x57	Duplicate top of expression stack
EFI_IFR_THIS_OP	0x58	Push the current question's value
EFI_IFR_SPAN_OP	0x59	Return first matching/non-matching character in a string
EFI_IFR_VALUE_OP	0x5A	Provide a value for a question
EFI_IFR_DEFAULT_OP	0x5B	Provide a default value for a question
EFI_IFR_DEFAULTSTORE_OP	0x5C	Define a Default Type Declaration
EFI_IFR_FORM_MAP_OP	0x5D	Create a standards-map form
EFI_IFR_CATENATE_OP	0x5E	Push concatenated buffers or strings
EFI_IFR_GUID_OP	0x5F	An extensible GUIDed op-code
EFI_IFR_SECURITY_OP	0x60	Returns whether current user profile contains specified setup access privileges
EFI_IFR_MODAL_TAG_OP	0x61	Specify current form is modal
EFI_IFR_REFRESH_ID_OP	0x62	Establish an event group for refreshing a forms-based element
EFI_IFR_WARNING_IF	0x63	Warning conditional
EFI_IFR_MATCH2_OP	0x64	Push TRUE if string matches a Regular Expression pattern

The installation hierarchy describes the above opcodes and has the following structure:



Installing expressions to describe the above operators has the following structure:


表达式
一元运算符
EFI_IFR_LENGTH
EFI_IFR_NOT | EFI_IFR_BITWISE_NOT
EFI_IFR_QUESTION_REF2
EFI_IFR_SET
EFI_IFR_STRING_REF2
EFI_IFR_TO_BOOLEAN | EFI_IFR_TO_STRING | EFI_IFR_TO_UINT
EFI_IFR_TO_UPPER | EFI_IFR_TO_LOWER
二元运算符
EFI_IFR_ADD | EFI_IFR_SUBTRACT | EFI_IFR_MULTIPLY | EFI_IFR_DIVIDE
EFI_IFR_AND | EFI_IFR_OR
EFI_IFR_BITWISE_AND | EFI_IFR_BITWISE_OR
EFI_IFR_CATENATE
EFI_IFR_EQUAL
EFI_IFR_GREATER_EQUAL | EFI_IFR_GREATER_THAN
EFI_IFR_LESS_EQUAL | EFI_IFR_LESS_THAN
EFI_IFR_MATCH | EFI_IFR_MATCH2
EFI_IFR_MODULO
EFI_IFR_SHIFT_LEFT | EFI_IFR_SHIFT_RIGHT
三元运算符
EFI_IFR_CONDITIONAL
EFI_IFR_FIND
EFI_IFR_MID
EFI_IFR_TOKEN
EFI_IFR_SPAN
内置函数
EFI_IFR_DUP
EFI_IFR_EQ_ID_VAL | EFI_IFR_EQ_ID_ID | EFI_IFR_EQ_ID_VAL_LIST
EFI_IFR_QUESTION_REF1 | EFI_IFR_QUESTION_REF3
2是一元运算符
EFI_IFR_RULE_REF
EFI_IFR_STRING_REF1
2是一元运算符
EFI_IFR_THIS
EFI_IFR_SECURITY
常量
EFI_IFR_FALSE | EFI_IFR_TRUE
EFI_IFR_ONE
EFI_IFR_ONES
EFI_IFR_UINT8 | EFI_IFR_UINT16 | EFI_IFR_UINT32 | EFI_IFR_UINT64
EFI_IFR_UNDEFINED
EFI_IFR_VERSION
EFI_IFR_ZERO
Rule
说明
form中预定义的表达式称为Rule
组件
EFI_IFR_RULE
EFI_IFR_RULE_REF
Question值
Boolean
64位无符号整型
字符串
Buffer（大小固定的8位无符号整型数组）
Undefined
CSDN @jiangwei0512

Each type has a corresponding structure. The corresponding type macros and structures are defined in MdePkg\Include\Uefi\UefiInternalFormRepresentation.h. In addition, you need to pay special attention to the following:

c AI generated projects 登录复制 run

```c
#define EFI_IFR_GUID_OP                 0x5F
```

Its structure is defined as follows:

c AI generated projects 登录复制 run

```c
typedef struct _EFI_IFR_GUID {
  EFI_IFR_OP_HEADER        Header;
  EFI_GUID                 Guid;
  //Optional Data Follows
} EFI_IFR_GUID;
```

You can see that it is not actually a complete version. The real complete version is defined in MdeModulePkg\Include\Guid\MdeModuleHii.h, which has different types:

c AI generated projects 登录复制 run

```c
///
/// GUIDed opcodes defined for EDKII implementation.
///
#define EFI_IFR_TIANO_GUID \
  { 0xf0b1735, 0x87a0, 0x4193, {0xb2, 0x66, 0x53, 0x8c, 0x38, 0xaf, 0x48, 0xce} }

///
/// EDKII implementation extension opcodes, new extension can be added here later.
///
#define EFI_IFR_EXTEND_OP_LABEL       0x0
#define EFI_IFR_EXTEND_OP_BANNER      0x1
#define EFI_IFR_EXTEND_OP_TIMEOUT     0x2
#define EFI_IFR_EXTEND_OP_CLASS       0x3
#define EFI_IFR_EXTEND_OP_SUBCLASS    0x4
```

收起 ^

`Guid` Currently, only the following data is defined `EFI_IFR_TIANO_GUID` , and the corresponding structure is different according to different type values.