Introduction to UEFI Graphics and HII

Kimon Berlin
Firmware Architect, HP Workstations

Week of June 12, 2007
Nanjing China

Overview

- UEFI Graphics
- Human Interface Infrastructure (HII)

UEFI Graphics

Why support graphics?

- Offer standard, rich pre-boot environment
  - Setup and configuration GUI
  - Logos
  - Localized output
  - GUI infrastructure for pre-boot apps (installation, recovery, update, diagnostics…)
- Unicode support
  - native UTF-16 (UCS-2) support, can display any language
  - only practical limit is ROM space
  - some glyph data can reside in the EFI partition (on disk)

Graphics-Related Protocols

- Simple Text Output (console output)
  - text-based, supports Unicode
  - remote-friendly
  - pseudo-text mode on local graphics console
- Simple Pointer
  - Mouse, trackball
- Absolute Pointer
  - Trackpad, touch screen
- Graphics Output
- EDID* Protocols
  - Monitor information

Chapter 11 of the UEFI specification

*: Extended Display Identification Data

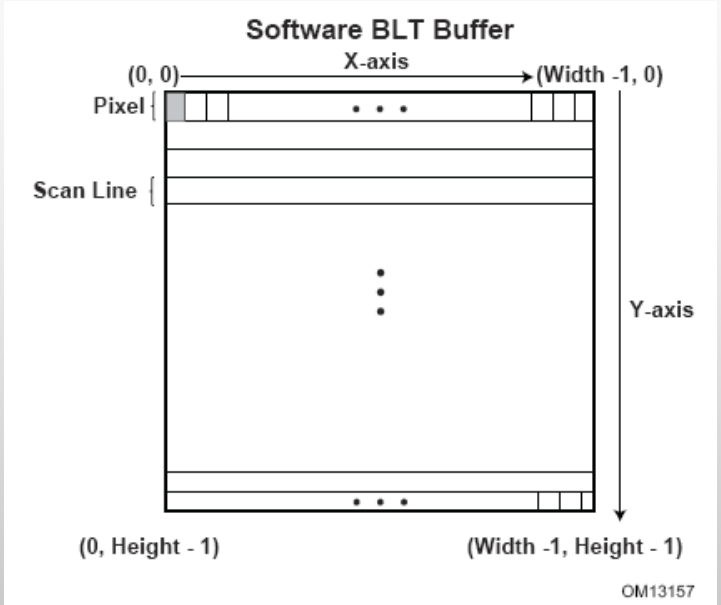
UEFI Graphics Output Protocol

- Replaces EFI UGA (and legacy VGA HW+BIOS)
- Pre-boot only today
  - HW frame buffer accessible to OS
- 800x600x32 minimum, no maximum
- No CSM or VGA dependencies
  - but these are required to boot a legacy OS
- GOP over VBE* possible
  - Uses existing VBE-compliant video BIOS
  - Requires CSM and VGA

*: VESA BIOS Extensions

Blt* Buffer

- Blt allows to read/write data to the graphic device's memory
- Blt buffer abstracts graphics HW implementation through a software buffer (in system memory)
- Standard XY array of pixels
- Different pixel formats
  - RGBr, BGRr, blt-only, …



*: **Bl**ock **t**ransfer

GOP Components

- QueryMode()
- SetMode()
- Blt()
- Mode structure
  - read-only
  - describes current mode
  - contents changed by using SetMode()

```
typedef struct {
  UINT32 MaxMode;
  UINT32 Mode;
  EFI_GRAPHICS_OUTPUT_MODE_INFORMATION *Info;
  UINTN SizeOfInfo;
  EFI_PHYSICAL_ADDRESS FrameBufferBase;
  UINTN FrameBufferSize;
} EFI_GRAPHICS_OUTPUT_PROTOCOL_MODE;

typedef struct {
  UINT32 Version;
  UINT32 HorizontalResolution;
  UINT32 VerticalResolution;
  EFI_GRAPHICS_PIXEL_FORMAT PixelFormat;
  EFI_PIXEL_BITMASK PixelInformation;
  UINT32 PixelsPerScanLine;
} EFI_GRAPHICS_OUTPUT_MODE_INFORMATION;
```

Blt Pixel Format

```
typedef struct {
  UINT8 Blue;
  UINT8 Green;
  UINT8 Red;
  UINT8 Reserved;
} EFI_GRAPHICS_OUTPUT_BLT_PIXEL;
```

Blt Operations

- EfiBltVideoFill()
- EfiBltVideoToBltBuffer()
- EfiBltBufferToVideo()
- EfiBltVideoToVideo()

Finding the GOP Handle

- Several handles can exist (e.g. console splitter)
  - Use LocateHandle() to find them
- The correct handle is associated with the graphics device and also has a device path
  - Use HandleProtocol() to select

```
Handle 7C (3E6C9418)
   Txtout (3CDCB960) Attrib 7
         mode 0: col  80 row  25
         mode 1: error Unsupported
      *  mode 2: col 128 row  40
   GraphicsOutput (3E6A5498)


Handle CB (3DE48B98)
   Txtout (3DC6A1B0) Attrib 7
         mode 0: col  80 row  25
         mode 1: error Unsupported
      *  mode 2: col 128 row  40
   GraphicsOutput (3DE49338)
   EdidDiscovered (3DE49358)
   EdidActive (3DE49368)
   ConOut (0)
   Dpath (3DE48C18)
'Acpi(PNP0A03,0)/Pci(4|0)/Pci(0|0)
/?'
```

GOP Demo

GOP Summary

- Consistent graphics abstraction
- Built-in
- Enables scaling with HW capabilities
- Practical limits:
  - ROM space (EFI partition can be used too)
  - No standardized remote graphics

Human Interface Infrastructure

HII Scope

- String and font management
- User input abstraction (keyboard, mouse, touchscreen…)
- Internal representation of Setup “forms”
  - concept borrowed from HTML forms
- External representation of forms to pass config info to/from runtime applications
  - runtime control of pre-boot config

HII Goals

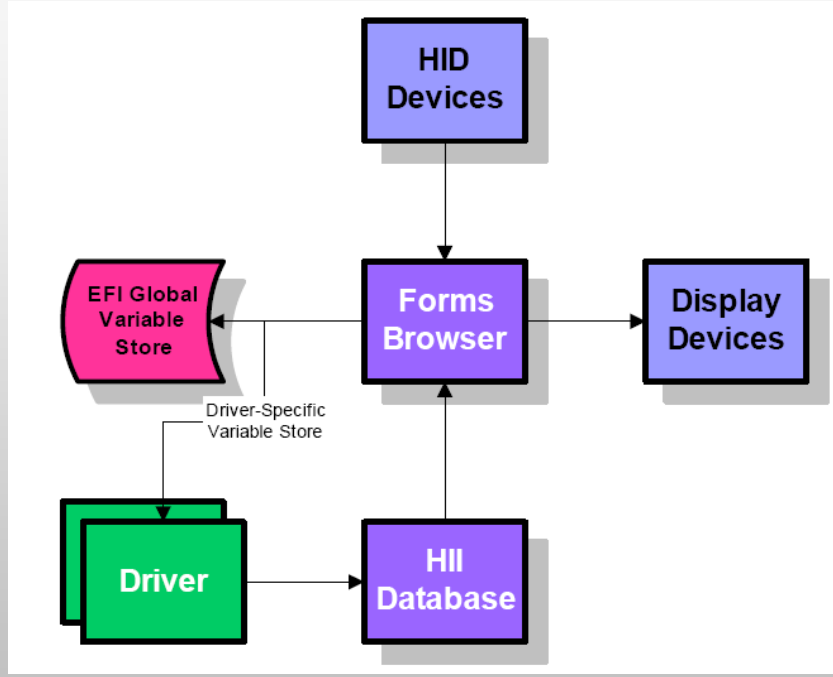
- Simplified localization
- Rich forms mechanism adapted to pre-boot ecosystem
- Configuration during boot and runtime
- Extensible
  - drivers and applications can contribute packages (forms, strings and fonts) to HII database
  - UI stays consistent
- Support UI on many types of display devices
  - text and graphics
  - local and remote

HII Components

- Strings
  - Unicode representation
- Fonts
  - Presumption of bitmap fonts for easier localization
- Human Input Devices (HIDs)
  - Keyboard: different keyboard mappings
  - Pointing devices
- Forms
  - Describe user interface layout for 'windowing' interfaces
  - An application that uses String and Font support
- Package
  - Self supporting data structure containing fonts, strings, and forms from a driver or set of drivers

HII Design

- No more option ROM-specific UIs
- Driver installs elements (fonts, strings, images, forms) into HII database
- Forms browser renders UI on display devices, receives user input from HIDs
- Changes are saved to EFI global store or to storage provided by drivers



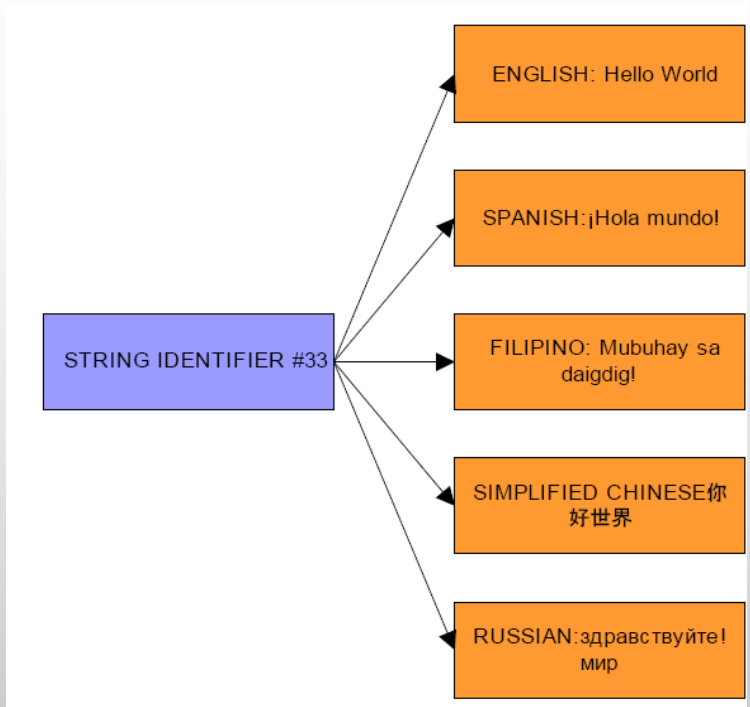
HII Protocols

- String
  - Handling of localized strings
- Font
  - Rendering glyphs on screen (frame buffer)
- Image
  - Create/get/set/draw images
- Database
  - Package management
  - Keyboard layouts

Chapter 28 of the UEFI specification

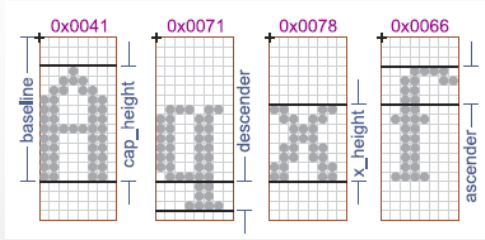
Strings

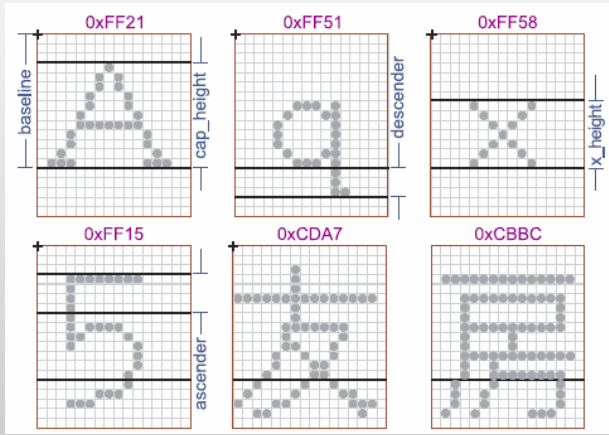
- All strings are UCS-2 (16-bit)
- Localization happens at the string level
  - Caller externs and passes in language independent string token (unique identifier per driver)
  - Forms brower selects actual text to display based on platform language setting
  - Separate packages for each language (add/remove independently)

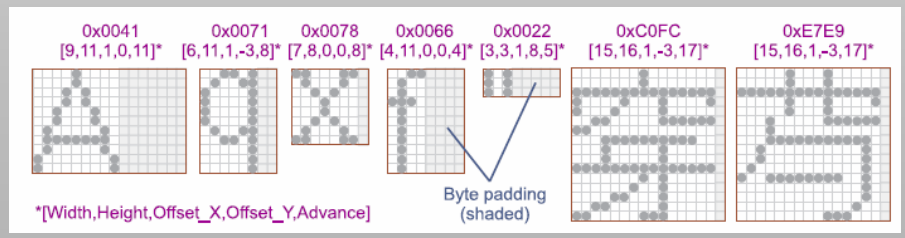


Fonts

- Standard font
  - "system"
  - Fixed-pitch
  - 8x19 and 16x19 glyphs
- Optional fonts
  - Fixed-pitch
  - Variable-pitch
- Efficient use of fonts in ROM
  - Unused characters can be stripped







Keyboards

- Support varying keyboards
  - Keyboard layouts vary widely
  - Adding support of other modifiers (e.g. Alt-GR, Dead-keys, etc)
- Keyboard Layout
  - Allow for a standardized mechanism to describe a keyboard layout and add to system database.
  - Allow for switching keyboard layouts.

Visual Forms Representation (VFR)

- Language used to describe what a page layout would be in a browser as well as the op-codes and string tokens to display
- Op-codes are defined for the following functions
  - FormSet and Forms definitions
  - Subtitle and other Text Fields
  - one of type questions with corresponding options
  - Checkbox fields
  - Numeric Fields
  - String Fields
  - Password fields
  - Boolean expressions in support of errors, suppression, and grayout

Form Example

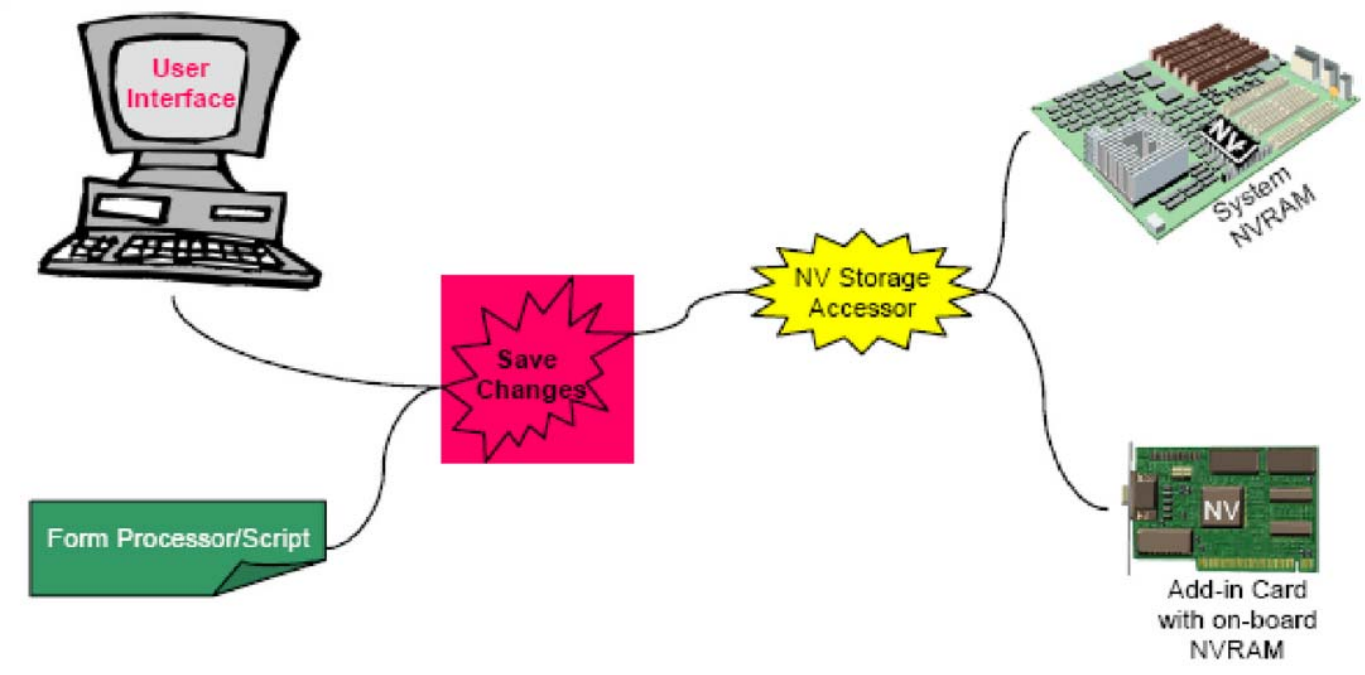
```
formset
  guid     = FORMSET_GUID,
  title    = STRING_TOKEN(STR_FRONT_PAGE_TITLE),

  form formid = 0x1000,
       title  = STRING_TOKEN(STR_FRONT_PAGE_TITLE);

checkbox
      varid  = D915G_SETUP_DATA.MemoryHoleEnable,
      prompt = STRING_TOKEN(STR_ENABLE_MEM_HOLE),
      help   = STRING_TOKEN(STR_ENABLE_MEM_HOLE_HELP),
      flags  = 0,
    endcheckbox;

 grayoutif
      NOT ideqval D915G_SETUP_DATA.MemDetectionMode == MEMORY_SETUP_MANUAL;
        help  = STRING_TOKEN(STR_DESIRED_SETTINGS_HELP),
        text  = STRING_TOKEN(STR_DESIRED_SETTINGS),
        text  = STRING_TOKEN(STR_BLANK_STRING),
        flags = 0,
        key   = 0;
      endif;

  endform;
```

IFR: User Interface

- Internal Forms Representation created by VFR to IFR tool
- Byte encoded operations (much smaller)
- String references abstracted as tokens
- Improved validation, visibility primitives
- At better level of presentation control for firmware
  - Tension between configuration driver and presentation driver over control of presentation format
- Easy to
  - Interpret for small Setup engine in desktop firmware
  - Translate into XHTML or JavaScript or …

NVRAM Storage

- Forms data encodes how to store the changes per configuration question
- System NVRAM and proprietary NVRAM can be used



HII Demo

Other localization issues

- Directional display
  - LTR, RTL
  - up to display engine
- Punctuation
  - handled in strings
- Date and time
  - format left to UI
- Numbers
  - print only integers
  - no separators
- Line breakage

Human Interface Summary

- Localization designed in from the start
- Localization is independent of display device
- Multi vendor repository for Fonts
- Maps setup easily into Web model
  - Setup replaced with a Markup Language
  - OEM can have unique look and feel
  - Browser defines look and feel
  - IFR maps to XML/HTML plus JavaScript

Q&A