Freescale Semiconductor
Application Note

Document Number: AN3985
Rev. 0, 02/2010

Customizing the Freescale® Advanced Toolkit for i.MX Based Platforms

by *Multimedia Applications Division*
Freescale Semiconductor, Inc.
Austin, TX

The Advanced Toolkit (ATK) is a container of tools for the i.MX processor family. This software is used for developing and validating i.MX based platforms. The ATK can be used both by the manufacturing and engineering teams to assist in board bring up, diagnostics, and testing. Although the ATK is not a substitute for a Joint Test Action Group (JTAG) tool, it can replace some functionalities that a JTAG tool provides.

The ATK software provides the following features:

- A Flash tool for downloading data to the SDRAM for programming, dumping, and erasing data in Flash memories such as NOR, NAND, and SD.
- Ability to download and execute the code directly to the SDRAM without having any custom bootstrap code in the flash on board.
- An image conversion tool for converting the image file formats such as binary to S-record, S-record to binary, and Executable Linkable Format (ELF) to binary.
- A graphical user interface (GUI) to control the tools.

Contents

© Freescale Semiconductor, Inc., 2010. All rights reserved.

freescale semiconductor

The official ATK releases from Freescale support all i.MX System on a Chip (SoC) and the development platforms available at the time of release. However, the customers frequently want to use the ATK with their custom hardware designs for basic debugging/validation or development.

A typical custom design uses components such as NAND or DDR memories that are not supported by the standard ATK. Due to this reason, Freescale provides the toolkit source to adapt it accordingly.

This application note describes ATK customization. In particular, the following scenarios are covered:

- Supporting a new SDRAM device
- Supporting a new NAND Flash device
- Supporting a new NOR Flash device
- Modifying the GUI

1 Prerequisites

This section describes the prerequisites for ATK customization.

1.1 Installing the Tool and Source Code Packages

Download the latest ATK tool and source code packages from Freescale website. If they are not available in the website, contact the Freescale representative to get access to the following packages.

- FSL_ATK_STD_v_vv.zip
  - FSL_ATK_TOOL_STD_v_vv
    a) ATK User's Guide Standard Version.pdf
    b) FSL_ATK_TOOL_WINS_STD_INSTALL_v_vv.exe
  - FSL_ATK_SRC_STD_v_vv
    a) ATK Reference Manual Standard Version.pdf
    b) FSL_ATK_SOURCE_CODE_STD_v_vv.exe

NOTE

v_vv is the version number corresponding to the latest ATK release.

Install the packages using the setup (.exe) files. Note the folder path where the software is installed. In this application note, the installation directories are referenced as below:

- Path to the ATK source code installation folder: `<ATK_SRC_PATH>`
- Path to the ATK tool installation folder: `<ATK_TOOL_PATH>`

Before customizing the ATK, check if the hardware support is available in the latest ATK tool release. See the *ATK User Guide Standard Version* document for a list of supported i.MX platforms and hardware.

1.2 Setting Up the Development Environment

Table 1 shows the tools required to setup the development environment.

Table 1. Development Tools

Software	Download From
Cygwin	http://www.cygwin.com/ To install cygwin, follow the recommended installation instructions at http://cygwin.com/faq.html
GNU ARM gcc tool chain (4.1.1) for cygwin	http://www.gnuarm.com/bu-2.17_gcc-4.1.1-c-c++_nl-1.14.0_gi-6.5.exe To install GNU ARM gcc, follow the installation instructions and use the default settings.
Microsoft Visual C++ 2005 or 2008 (OPTIONAL)	Get it from the Microsoft download center. This is only required to build the host DLL and GUI

NOTE

After installing cygwin and GNU ARM tool chain, copy all `cygwin.dll` files from `C:\Program Files\GNUARM\bin` (or from the GNU ARM installation path) to `C:\cygwin\bin` (or to the directory where cygwin is installed).

2 ATK Components and Environment

Figure 1 illustrates the ATK components and the environment where the ATK tool is used.

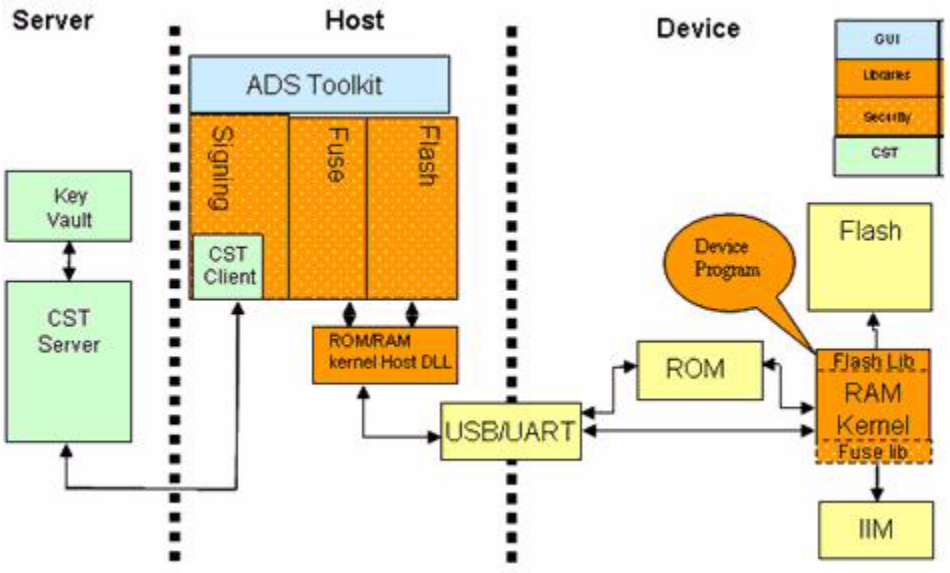

Figure 1. ATK Components and Environment

NOTE

The fuse and code signing functionalities are not available in the standard ATK package.

3 Supporting a New SDRAM Device

The i.MX family of processors incorporates a SDRAM controller (ESDRAMC), which is capable of supporting the following memory types:

- Single Data Rate (SDR) SDRAM
- Double Data Rate (DDR) SDRAM
- Low-Power (LP) DDR SDRAM (also referenced as mDDR)
- Double Data Rate 2 (DDR2) SDRAM

NOTE

Not all memory types are supported by all the i.MX SoCs. See the hardware reference manual of the i.MX device available for information about supported SDRAM types.

While using a SDRAM device, which is not supported by the ATK, a custom initialization file must be available. This initialization file configures the i.MX processor and initializes the SDRAM memory. This is analogous to a JTAG development tool initialization file.

3.1 Creating a Custom Initialization File for SDRAM Memory

The memory initialization file is a text (.txt) file in which each line (terminated with CR+LF) contains a register write operation to the i.MX processor. Each line in this file contains information organized in three columns. The comments in the file begin with #.

The format of the memory initialization file is described in Table 2.

Table 2. Memory Initialization File Format

Address (hexadecimal)	Space	Data (hexadecimal)	Space	Access Type (decimal)
Indicates the address of the register for writing. The address must be in hexadecimal and must begin with 0x	Single blank space	Indicates the value to be written to the register. The value must be in hexadecimal and must begin with 0x	Single blank space	Indicates the data access type such as 8-bit (8), 16-bit (16) or 32-bit (32)

The configuration in this file is sent to the i.MX processor, while it is executing the internal ROM code (bootstrap mode) out of the mask ROM embedded in the processor. This data is sent through the Universal Serial Bus (USB) or Universal Asynchronous Receiver/Transmitter (UART) links using a particular protocol. The i.MX internal ROM code writes these values to the respective memory addresses (registers) to configure the memory controller and the external SDRAM device.

3.1.1 Memory Initialization Example

Table 3 shows the memory initialization file format for the i.MX27 IP camera reference design. This reference design uses an LPDDR model MT46H16M32.

Table 3. Memory Initialization File Format

Description	Line(s) within the File
Enable LPDDR delay line measure unit and start a new measurement (using ESDMISC register).	0xD8001010 0x00000008 32
Set drive strength to high on address pads (A15-A0)	0x10027828 0x55555555 32
Set drive strength to high on data pads (SD15-SD0).	0x10027830 0x55555555 32
Set drive strength to high on data pads (SD31-SD16).	0x10027834 0x55555555 32
Set drive strength to high on the pads (SDBA1, SDBA0, CSD0, CSD1)	0x10027838 0x00005005 32
Set drive strength to high on the pads (SDQS3 - SDQS0, SDCLK, SDCKE1, SDCKE0, SDWE_B, CAS_B, RAS_B, MA10, DQM3-DQM0)	0x1002783C 0x15555555 32
Deassert the reset signal on the delay line unit and enable the Low Power DDR operation mode of the ESDRAMC	0xD8001010 0x00000004 32
Set DDR timing parameters (tXP, tWTR, tRP, tMRD, tWR, tRAS, tRRD, tCAS, tRCD, tRC).	0xD8001004 0x00795729 32
Set data size, number of columns and rows, enable the SDRAM controller and set the memory controller operating mode to precharge command to begin LPDDR initialization sequence.	0xD8001000 0x92120000 32
Send precharge command to DDR.	0xA0000F00 0x12121212 32
Change the memory controller operating mode to auto-refresh command.	0xD8001000 0xA2120000 32
Send autorefresh command to DDR.	0xA0000F00 0x34343434 32 0xA0000F00 0x56565656 32
Change the memory controller operating mode to load mode register command.	0xD8001000 0xB2120000 32
Configure standard and extended mode registers of DDR	0xA0000033 0xda 8 0xA1000000 0xff 8
Set the ESDRAMC refresh rate, power down timer, burst length, and precharge timer	0xD8001000 0x82128485 32
Initialization is done. Perform a dummy write to DDR memory.	0xA0000000 0xCAFECAFE 32

Some more examples of memory initialization files are available in the following path:

`<ATK_TOOL_PATH>\example\memory_init\`

Once the file editing is complete, use the ATK tool to verify if it works with the i.MX based board. Start the ATK tool and select the i.MX CPU that matches the design and provide the custom memory initialization file. See Figure 2.

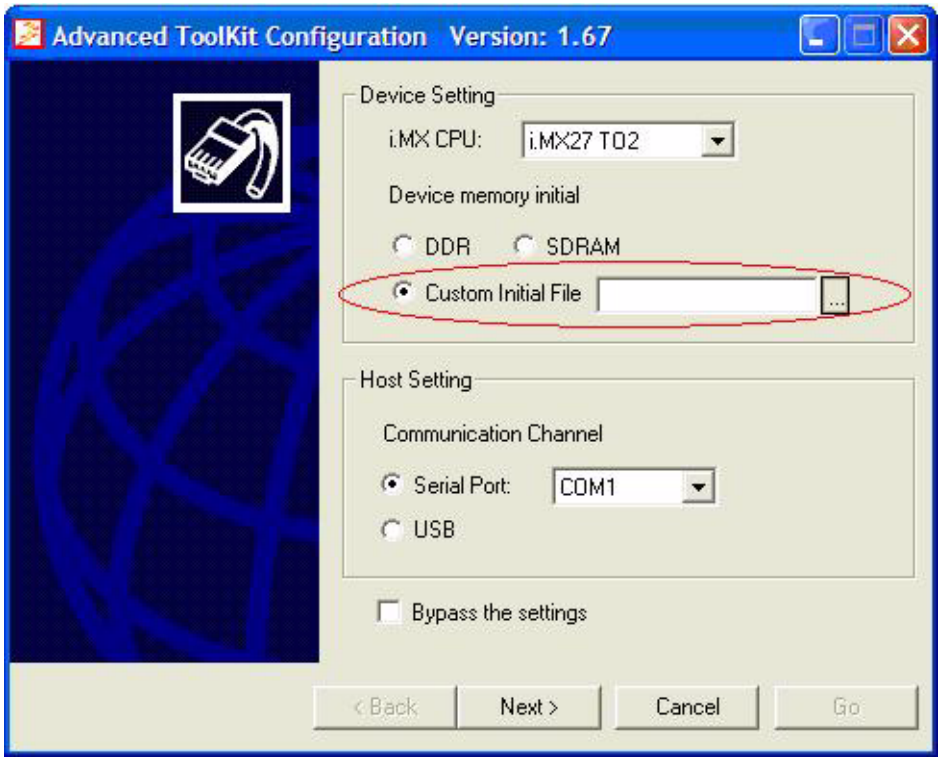

Figure 2. ATK Tool Custom Memory Initialization

4 Supporting a New NAND Flash Device

The i.MX family of processors incorporates a NAND flash controller (NFC) capable of supporting the following memory types:

- Single Level Cell (SLC) NAND
- Multi-Level Cell (MLC) NAND

NOTE

Not all NAND flash types are supported by all the i.MX SoCs. Refer to the available hardware reference manual of the i.MX device, for information about the supported NAND flash types.

While using a NAND flash device, which is not supported by the ATK, the device program's source code has to be adapted accordingly (in particular the flash library) and rebuilt. This new flash library binary contains the support for the new hardware and is downloaded to the i.MX device, allowing the in-circuit programming of the new NAND flash model.

4.1 Adding a NAND Flash Model to the Flash Library

To add a new NAND flash model to the flash library of the device program, search the file `nand_ids.c` in the following path:

`<ATK_SRC_PATH>/device_program/flash/nand_flash/src`

This file contains the NAND flash models that are currently supported by the ATK flash library. To support a new device, add a new entry to this file using the format shown in Table 4.

Table 4. NAND Flash Model Definition Used in the nand_ids.c File

{ man,	dev,	io,	ps,	oob,	mo,	po,	scan,	row,	blks,	ppb,	name },
Manufacturer ID	Device ID	Bus Width	Page Size	Spare Area Size	Offset for bad block indication within one page	Offset for the page which contains the bad block indication within one block	Number of pages needs to be scanned for the bad block indication	Row cycles	Number of blocks this NAND flash contains	Number of pages per one block	NAND flash model name

Example 1 shows how to add a NAND flash model to the flash library.

Example 1. Adding a NAND Flash Model to the Flash Library

```
{0xEC, 0x76, 8, 512, 16, 5, 0, 1, 3, 4096, 32, "myflashmodel"},
```

The above parameters are available in the manufacturer's datasheet of the NAND flash device. In some cases, the datasheet does not explicitly define some of these parameters such as mo, po, scan. But, these should be available, at least embedded as phrases in a paragraph in the bad block management section. If the corresponding parameters are not available, contact the NAND flash device manufacturer to get the required information.

5 Supporting a New NOR Flash Device (Spansion)

The Freescale i.MX family of processors incorporates an external memory interface that supports the NOR flash devices. While using a NOR flash device, which is not supported by the ATK, the device program’s source code has to be adapted accordingly (in particular the flash library) and rebuilt. This new flash library binary supports the new hardware and it is downloaded to the i.MX device, allowing the in-circuit programming of the new NOR flash model

5.1 Adding a NOR Flash Model to the Flash Library

Similar to the NAND flash device, the i.MX processor is also capable of interfacing additional NOR flash devices by customizing the ATK's fuse library.

Search the file `nor_flash.c` in the following path:

`<ATK_SRC_PATH>/device_program/flash/nor_flash/spansion/src`

This file contains the NOR flash models that are currently supported. To add a new device, create a new entry within this structure and modify the parameters accordingly.

See Table 5, for more information about the parameters.

Table 5. NOR Flash Model Definition used in the nor_flash.c File

Parameter	Description
device name	The NOR flash model or part number
device_size	The NOR flash size
max_wb_word	Maximum write buffer word
device_id	NOR flash ID read out by auto-select mode, concatenate the third and second word (in the same order) of the device ID available in the device's datasheet
sector_size	Sector size array, lists the supported sector sizes
sector_mask	Use the sector address to mask this value and get the sector size index in sector_size[]

Example 2 shows how to add a NOR flash model to the flash library.

Example 2. Adding a NOR Flash Model to the Flash Library

```
static struct nor_flash_model supported_model[] = {
{
.device_name = SG29GL512N,
.device_size = DEVICE_64M,
.max_wb_word = 16,
.device_id = 0x22012223,
.sector_size = { SECTOR_128K, 0 },
.sector_mask = 0,
},
};
```

Once the file editing is complete, follow the instructions in Section 6, “Building the Flash Library,” for building the new flash library and using the resulting binary with the ATK tool.

6 Building the Flash Library

The method described here works for a Microsoft Windows® environment using cygwin. This emulates a Linux layer and allows GNU ARM compilation but, the device program code can also be built using a true Linux host with a proper GNU ARM tool chain installed.

To build the ATK flash library follow the steps below:

1. Ensure that the required tools listed in Section 1.2, “Setting Up the Development Environment,” are installed.
2. Launch cygwin bash shell.
3. Change the directory to the ATK source path

```
$cd <ATK_SRC_PATH>/device_program/
```

For example: `$cd /cygdrive/d/atk_src/device_program/`

4. Enter the next build commands according to build targets
   — Delete the unused files

   ```
   $make clean
   ```

   — make the target

   ```
   $make MCU={user input} REV={user input} flashlib
   FLASH_TYPE={user input} FLASH_MODEL={user input}
   ```

   NOTE

   See Section 6.1, “Compiler Flags,” for information on filling the user input fields.

5. Copy the new flash library (either NAND or NOR) from the `<ATK_SRC_PATH>/device_program/bin` to the ATK tool installation path `<ATK_TOOL_PATH>`. Copying can be done manually using the Microsoft Windows Explorer. The default directory of RAM Kernel images is

   `<ATK_TOOL_PATH>/image`

   For example: `C:\Program Files\freescale\AdvancedToolKit-STD\image.`

   The RAM kernel images that comes with the ATK tool are available in this directory. To install the new RAM kernel image (that contains the new flash library), back up the existing file (renaming it is an easier and faster way to back it up), then copy the new file inside this directory with the same name as that of the original file. If the above steps are followed, then step 6,7, 8 can be skipped and the user can start using the ATK tool with support for the new flash hardware.

6. To add a new flash model to the ATK’s GUI combo box, add an entry to the `ADSToolkit.cfg` file. This file is located in the following path:

   `<ATK_TOOL_PATH>/config`

   For example: `C:\Program Files\freescale\AdvancedToolKit-STD\config`

   The new entry must follow the format below:

   TYPE OF MEMORY:MODEL:`<YOUR_BIN_LOCATION>:SIZE`

   For example, for the i.MX27 TO1 or TO2, the available flash memories are:

   ```
   [MX27]
   NOR:Spansion:image\mx27_nor_spansion.bin:0x(unknown)
   NAND:image\mx27_nand.bin:0x(unknown)
   ```

   `<YOUR_BIN_LOCATION>` is relative to `<ATK_TOOL_PATH>`

7. An alternative method to assign a new RAM kernel binary to the ATK is by using the Custom Model option available in the flash GUI combo box.

8. To modify the GUI and generate a new ATK tool package, install the built binary files in `<ATK_SRC_PATH>/device_program/bin` to the `<ATK_SRC_PATH>/gui_application/image` folder using the following command:

   ```
   $make install DEST=../gui_application/image/
   ```

6.1 Compiler Flags

This section describes the complier flags.

- MCU—Indicates the type of i.MX SoC.
  - mx31: i.MX31 chip
  - mx32: i.MX32 chip
  - mx27: i.MX27 chip
  - mx35: i.MX35 chip
  - mx37: i.MX37 chip
  - mx51: i.MX51 chip
  - mx25: i.MX25 chip
- REV—Indicates i.MX SoC revision version.
  - to1: Revision 1
  - to2: Revision 2
- FLASH_TYPE—Indicates the type of flash.
  - nor: NOR Flash
  - nand: NAND Flash
  - mmc: MMC
  - sd: SD

NOTE

mmc and sd are not available on all i.MX processors.

- FLASH_MODEL— Indicates the flash model
  - spansion: NOR Flash S71WS256ND0/SG29GL512N

NOTE

The FLASH_MODEL flag is obsolete on newer ATK packages (1.5.1 and above), use the FLASH_TYPE=nand instead. See `Makefile` available in `<ATK_SRC_PATH>device_program/flash/`, for more details about compiler flags.

7 Modifying the Graphical User Interface

This section describes the GUI modification.

7.1 Adding a New Tool to the GUI

The ATK's GUI can be customized to fit any new function or tool the customers would like to add to it. This section describes how to add a new tool to the ATK's GUI. Microsoft Visual C++ 2005 or 2008 with MFC support should be installed to edit and build the GUI and host DLL.

Execute the following steps to add a new tool to the GUI:

1. Open the `gui_application/ADSToolkit_std.dsw` project.
2. Using the solution explorer, double-click on ADSToolkit > Resource Files > ADSToolkit.rc2 to open the resource view and then search the IDW_TOOLSELECT panel and open it (double click). This panel is located in the path; ADSToolkit > ADSToolkit.rc > Dialog > IDW_TOOLSELECT.
3. Open the toolbox (if it does not appears automatically).
4. Select radio button control and draw a new button on the panel.
5. Modify the radio button properties. Change the caption text, in this example **My button** is used. Also, set the Push Like property to true. See Figure 3.

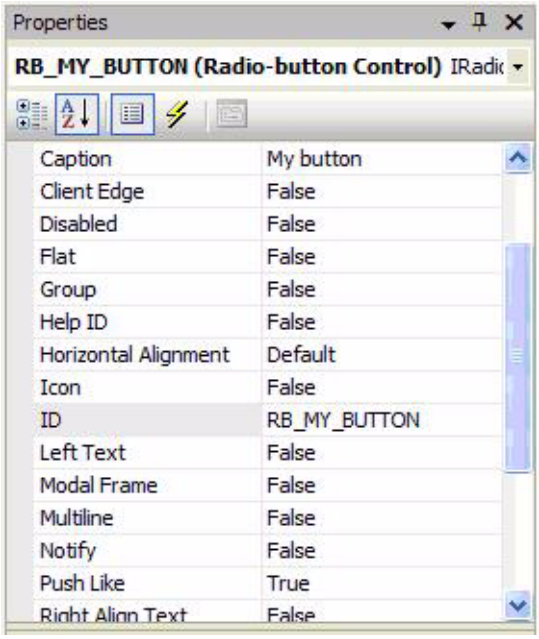

Figure 3. Radio Button Properties

The modified panel looks as shown in Figure 4.

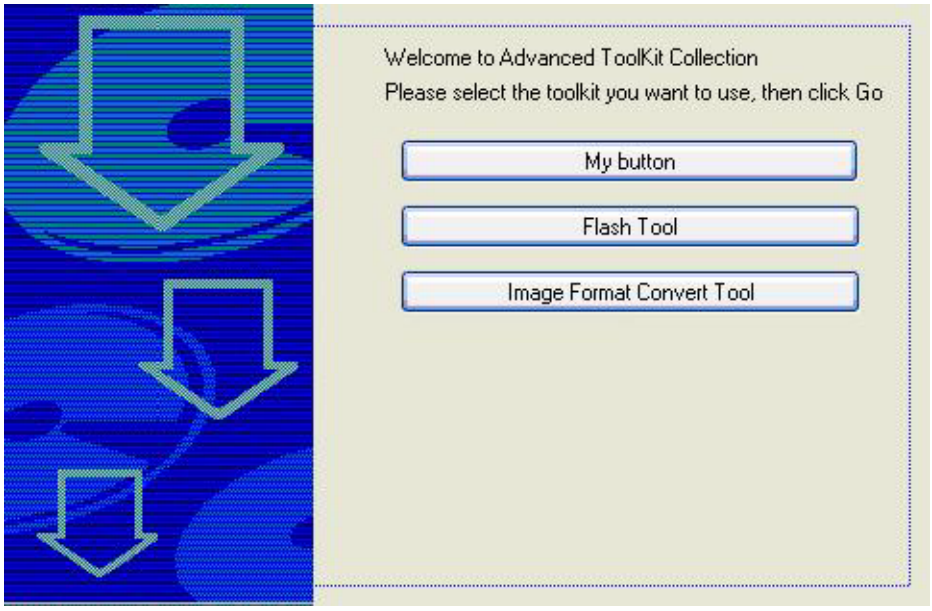

Figure 4. My Button Added to the Panel

6. Right-click the dialog folder of the resource view and select **insert dialog** to add a new dialog.
7. By default, the new dialog is named IDD_DIALOG1. The default name can be modified using the ID parameter on the properties window.
8. Right-click the dialog and select **add class** to attach a new class. An MFC wizard window is displayed.
9. Design the graphic environment of the new dialog according to the requirements. If required, other dialogs can be used as a base.
10. After completing the new tool design, connect it to the ATK so that the tool can be opened when selected. Open the `ToolSelectPage.cpp` source file, and execute the following:
    - Add the include file of the dialog, for example:

      ```
      #include "MyButtonDlg.h"
      ```
    - Search for the following line:

      ```
      ((CButton *)GetDlgItem(RB_FLASH_TOOL))->EnableWindow(TRUE);
      ```
    - Add an equivalent line with the new tool button:

      ```
      ((CButton *)GetDlgItem(RB_MY_BUTTON))->EnableWindow(TRUE);
      ```
    - Search for the following line:

      ```
      ((CButton *)GetDlgItem(RB_FLASH_TOOL))->SetCheck(BST_UNCHECKED);
      ```
    - Add an equivalent line with the new button:

      ```
      ((CButton *)GetDlgItem(RB_MY_BUTTON))->SetCheck(BST_UNCHECKED);
      ```
    - Add an element to the array pBtn in the OnWizardFinish function (remember to increase the size of the array). Modify the for loop that follows this array, so that it checks for one more element (for example, change i < 6 to i < 7).

11. Open the `NewWizPage.h` header file and add the tool name to the TOOL_PAGE_T typedef, in the same position where the pBtn array is added before. Example 3 shows how to add a tool name:

Example 3. Adding a Tool Name

```
typedef enum
{
     NONE_TOOL = -1,
     FLASH_TOOL = 0,
     FUSE_TOOL,
     CST_TOOL,
     IMAGE_BUILDER,
     DCD_BUILDER,
     IMAGE_CONVERT,
     MY_BUTTON,
} TOOL_PAGE_T;
```

12. Open the `NewWizDialog.cpp` source file and add the tool header file to the include list. Search the OnWizardFinish function and a new case entry to the switch of variable `m_iPage` as shown below:

```
case MY_BUTTON:
         {
         MyButtonDlg pDlg;
          if (pDlg.DoModal() == IDOK)
         {
         ShowWindow(SW_SHOW);
         } else {
         CDialog::EndDialog(0);
         }
         break;
         }
```

13. Once this is complete, a new GUI can be built. See Section 7.2, “Building the Host DLL, GUI Application, and Updating the ATK Tool Package,” for information on building a new GUI.

7.2 Building the Host DLL, GUI Application, and Updating the ATK Tool Package

The host DLL and GUI applications are built using Visual C++ 2005.

7.2.1 Building the Host DLL

.To build the host DLL, execute the following steps:

1. Open `<ATK_SRC_PATH>\host_dll\AtkHostApi_std.dsw` and set Build > Configuration Manager as Win32 Release for a release version or as Win32 Debug for a debug version.
2. Select Project > Properties > Configuration Properties > General > Project Defaults > Use of MFC as Use MFC in a Static Library.
3. Select Project > Properties > Configuration Properties > Linker > Input > Additional Dependencies to ensure `MXUsb.lib` is linked.
4. Select Build > Rebuild AtkHostApi. After building is complete, install the generated files in the path: `<ATK_SRC_PATH>\host_dll\Release\ to <ATK_SRC_PATH>\gui_application\Release`

Or if the debug environment is selected, install the files in the path:

<ATK_SRC_PATH>\host_dll\Debug\ to <ATK_SRC_PATH>\gui_application\Debug

7.2.2 Building the GUI Application

To build the GUI application, execute the following steps:

1. Open <ATK_SRC_PATH>\gui_application\ADSToolkit_std.dsw.
2. Select Build > Configuration Manager as Win32 Release for a release version or as Win32 Debug for a debug version.
3. Select Project > Properties > Configuration Properties > General > Project Defaults > Use of MFC as Use MFC in a Static Library.
4. Select Project > Properties > Configuration Properties > Linker > Input > Additional Dependencies to ensure that AtkHostApi_std.lib is linked to the right location. By default, AtkHostApi_std.lib is linked to one of the following path:

   ../host_dll/Release/AtkHostApi_std.lib

   ../host_dll/Debug/AtkHostApi_std.lib
5. Select Build > Rebuild ADSToolkit. The output file ADSToolkit_stdt.exe is located in one of the following path:

   <ATK_SRC_PATH>\gui_application\Release, or <ATK_SRC_PATH>\gui_application\Debug directory.

7.2.3 Updating the ATK Tool with the Newly Built Files

To run the ATK, execute the following steps:

1. If ADSToolkit.exe is located in the path; <ATK_TOOL_PATH>, then install AtkHostApi_std.dll in the path <ATK_TOOL_PATH>.
2. Copy the following directories to the path <ATK_TOOL_PATH>:

   <ATK_SRC_PATH>\gui_application\bin

   <ATK_SRC_PATH>\gui_application\config

   <ATK_SRC_PATH>\gui_application\image

8 Revision History

Table 6 provides a revision history for this application note.

Table 6. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	02/2010	Initial release.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale and the Freescale logo are trademarks or registered trademarks of Freescale Semiconductor, Inc. in the U.S. and other countries. All other product or service names are the property of their respective owners. ARM is the registered trademark of ARM Limited. ARMnnn is the trademark of ARM Limited.

© Freescale Semiconductor, Inc., 2010. All rights reserved.

Document Number: AN3985
Rev. 0
02/2010