i.MX6 series Firmware

1 Introduction

The purposes of the firmware library for the i.MX6 series are to:

- Provide example driver code that enables main features of a peripheral without the need for an operating system.
- Provide example usage for each driver to demonstrate their main features.
- Provide a solid and simple environment for quick board bringup.

The outcome is finally a test suite where each module can be individually tested. Based on a common initialization of the i.MX6 and the development board, a test for a specific peripheral is launched with the help of a serial console for human interaction.

2 Versions Available

	Version	Released Date	Description
Software (2011w39_mx61_evb_sdk_release.tar.gz)	v1.0.0	09/30/2011	Initial Release with following drivers IPU, ESAI, SSI, AUDMUX, I2C, HDMI, GPT, EPIT, GIC, SDMA, UART and USDHC.
Firmware Guide (IMX6FG_RevA.pdf)	Rev A	09/28/2011	Detailed driver documentation available with the release

3 Source code structure of the library:

Description	Location
Configuration file used to enable what should be supported by a board. e.g mx61_evb.conf	./configs
Source code related to the ARM Cortex-A9 core.	./src/cortex_a9
Common headers which are platform and i.MX independent.	./src/include
Headers specific to the i.MX6 series.	./src/include/mx61
Global library source file such main, system initialization,...	./src/init
Source code specific to the i.MX6 series.	./src/mx61/
Parent directory for all drivers, includes, and their unit tests and docs.	./sdk/

Example for a driver:

Description	Location
SDMA driver.	./sdk/sdma/drv
SDMA driver related includes.	./sdk/sdma/inc
SDMA driver unit tests.	./sdk/sdma/test

4 Build procedure

4.1 System requirements

To build the firmware library a Linux host or Windows + Cygwin host is necessary.

Cygwin can be installed and obtained from here:
http://www.cygwin.com/
Please follow the installation procedure documented on this site.

The code is compiled with the ARM-NONE-EABI GCC toolchain obtained here:
http://www.codesourcery.com/sgpp/lite/arm/portal/subscription3053
Click on "Download Sourcery G++ Lite 2011.03-42" and download the Windows or Linux tar ball or installer.

Install the package in "/opt" or any other local folder, and make sure that the PATH environment variable allows accessing these executables used for the build process:

```
CC= arm-none-eabi-gcc
AS = arm-none-eabi-as
AR = arm-none-eabi-ar
LN = arm-none-eabi-ld
```

Otherwise, it is possible to re-define the executables in this file ./install_path/make.def.

4.2 Source code installation

The source code package can be installed anywhere.

4.3 Build command

Each module can be tested by using a command like:
./tools/build_sdk –target mx61 –board evb –board_rev 1 –test uart

Note: the above command generates a makefile in the installation path.

Only the i.MX6 EVB rev1 is available in this release.

Tests list is:

Module	parameter
Audio : SSI and audiomux	- audio
EPIT	- epit
Generic Interrupt Controller - GIC	- gic
GPT	- gpt
HDMI	- hdmi
IPU	- ipu
SDMA	- sdma
UART	- uart
USDHC	- usdhc

Note: the warnings detected during the build process are considered as errors.

Once the build is finished a message should appear such:

```
************************************************
  Build has completed. ELF file available in:
  => /install_path/output/mx61/bin

Done ... Build script completed
```

The output binary and ELF files can be found in:
./output/mx61/bin/mx61ard-epit-sdk.elf
./output/mx61/bin/mx61ard-epit-sdk.elf

The file name contains the targeted CPU and board, as well as the tested module.

It is sometimes required to clean the build to take into account some changes. The following command will remove the directory ./install_path/output/mx61 before building:
./tools/build_sdk –target mx61 –board evb –board_rev 1 –test uart -clean

5 How to run a test

5.1 Setup required

An i.MX61 EVB board is necessary to run a test. Please refer to the appropriate board user guide to prepare this setup.

The user interaction and output information are available through the serial port UART4 connected to a host running a terminal such Teraterm, miniterm, hyperterminal,…
The configuration for the terminal is common: 115200bps, 8-bit, no parity, 1 stop bit.

5.2 Download with a JTAG probe

The code is downloaded and executed thanks to the help of a JTAG tool.
It is necessary to have a board initialization file that will initialize some of the clocks, the MMDC controller and its interface, as well as the DDR3.
The source package contains the inc file used by the ARM Realview Debugger to initialize the board.

5.3 Using SD boot

The SDK binary image can be programmed on an SD card using cfimager-imx.exe provided with the release package under folder tools/windows.
Command:
cfimager-imx.exe –o 0x0 –f ./output/mx61/evb/bin/mx61evb-ALL-sdk.bin –d <your drive letter for the SD card>

For example, if the drive letter of the SD card reader is “f”, type the following:
cfimager-imx.exe –o 0x0 –f ./output/mx61/evb/bin/mx61evb-ALL-sdk.bin –d f

There is also the option of formatting the card first before programming it, though be warned it will take more time for this operation to complete due to the card formatting.
To invoke the card formatting, simply append the above command with “-a”, as provided in the following example (again, assuming the drive letter is “f”):
cfimager-imx.exe –o 0x0 –f ./output/mx61/evb/bin/mx61evb-ALL-sdk.bin –d f -a

2. Use “dd” under linux.
dd if=output/mx61/evb/bin/mx61evb-ALL-sdk.bin of=/dev/sdx seek=2 skip=2 && sync
/dev/sdx is the driver letter for your SD card.

Once the SD card is programmed successfully with bootable SDK image, follow below instructions on how to run it on MX61 evb board:

- Insert SD card with SDK binary into SD slot.
- Configure boot dip switches for boot from SD – SW1: off-off-off-off-off-off-on-off; SW2: off-off-off-on-on-off-off-off; SW5: all off; SW8: all off.
- Power on board, and follow on screen prompt commands on host' terminal window.

6 How to add a new driver and test

A new driver should typically be added in the ./sdk/new_driver/drv and its test into ./sdk/new_driver/test .

In the board config file such mx61_evb.conf, the following lines must be added:
src/sdk/new_driver/drv
src/sdk/new_driver/test
Everything contained in these repositories will be compiled.

The main test file should contain a test function named: `int32_t new_driver_test(void)`.
Hence, when building, the parameter "– new_driver" can simply be used.