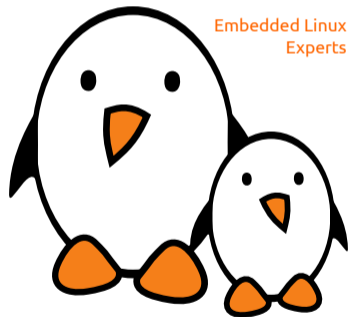# SD/eMMC: new speed modes and their support in Linux

Gregory CLEMENT
Free Electrons
gregory@free-electrons.com

Embedded Linux
Experts

- Embedded Linux engineer and trainer at **Free Electrons**
  - Embedded Linux **expertise**
  - **Development**, consulting and training
  - Strong open-source focus
- Open-source contributor
  - Contributing to **kernel support** for the Armada 370, 375, 38x, 39x and Armada XP ARM SoCs and Armada 3700, 7K/8K ARM64 SoCs from Marvell.
  - Co-maintainer of mvebu sub-architecture (SoCs from Marvell Engineering Business Unit)
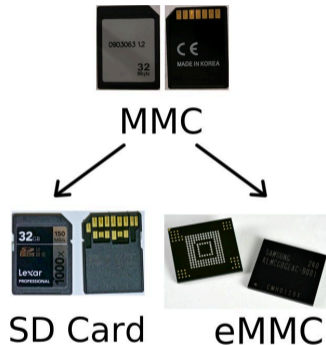 - Living near **Lyon**, France
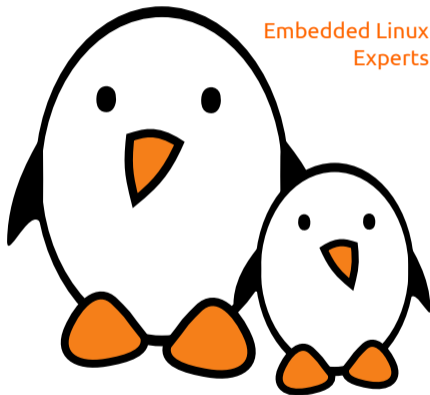
Embedded Linux Experts

SD card and eMMC

SD card and eMMC have common point:

- Both come from **MMC** (*MultiMediaCards*).
- Increase their bandwidth as new versions of the standards were released
  - Now they can reach more than 400MB/s in theory
- Supported in Linux though the **mmc subsystem**

MMC

SD Card   eMMC

Photographies from Wikipedia, credit:
MMC: Pixelk - CC BY 1.0
SD Card: Adryan R. Villanueva - CC BY-SA 4.0
eMMC: Toniperis - CC BY-SA 4.0

- Presentation of **SD Card** and **eMMC**
- Initial support in Linux
- The new speed modes
- State of the support for these new speed modes in Linux

SD card

- **SD** stands for *Secure Digital*
  - *"Secure" for copyright content*
- Introduced in 1999
- **MMC** extension
- Standardized by **SDA** (*SD Association* created in 2000)

SD card - Hardware

- ▶ Flash chip + small micro-controller in a card
- ▶ 9 pins: `CLK, CMD, DAT0-3, VDD, VSS1-2`
- ▶ SPI mode compatibility
    - ▶ `DAT3 -> CS, CMD -> DI, DAT0 -> DO`
- ▶ In initial release 25MHz clock

SD card - Protocol

SD Bus protocol

- ► Command and data bit stream
- ► Command and response on CMD line
- ► Data on the data lines
- ► Basic transaction command/response
- ► Some operations can have data token
- ► All communication initiated by the host
- ► Data transfer in block with CRC
- ► Multiple data block: always stop by a host command

SD card vs MMC

- Initial version: 1 data line for **MMC** vs 4 for **SD card**
- Nowadays **MMC** can go up to 8 data lines
- No DRM in **MMC**
- Command set diverged
- Both have SPI compatibility mode

eMMC

- other **MMC** extension from **MMCA** and **JDEC**
- **eMMC** stands for *embedded* MultiMedia Card
- Mentioned in the **MMC** spec v4.1 in 2007

eMMC - Hardware

- Flash chip + small micro-controller in BGA chip
- Pinout:
  - Since 4.1, 14 pins: CLK, CMD, DAT0-7, VccQ, VssQ, Vcc, Vss
  - With 4.4 version, one more pin: RST_n (Reset)
  - With 5.0 version, one more pin: DS (Data Strobe)
- No more SPI mode compatibility since 4.3
- In initial release: 52 MHz clock

Bus protocol same than the SD bus protocol (both came from **MMC**)

- ▶ Command, response on CMD line
- ▶ Data on the data lines
- ▶ Basic transaction command/response
- ▶ Some operation can have data token
- ▶ All communication initiate by the host
- ▶ Data transfer in block with CRC
- ▶ Multiple data blocks: always stop by a host command

eMMC vs MMC card

Both defined by the same specification but there still differences:

- ▶ **eMMC** BGA chip soldered to a board
- ▶ **MMC card** removable part
- ▶ Dedicated feature for **eMMC** such as partitioning, device information
- ▶ **eMMC** widely used whereas the MMC card are hard to find

- **MMC** Framework added in 2004 with **2.6.9** by **Russell King**
- **SD card** Support added in 2005 with **2.6.14** by **Pierre Ossman** (who became MMC maintainer in 2006)
- **SDHCI** (Secure Digital Host Controller Interface) added with **2.6.17** in 2006
- High Speed mode (clock up to 52MHz) for **MMC** added with **2.6.20**
- High Speed mode for **SD Card** added in the same release
- **SDIO** extension support with **2.6.24** in 2007

- ▶ Code located in `drivers/mmc` and headers in `include/linux/mmc/`
- ▶ Currently maintained by **Ulf Hansson** since 2014
- ▶ Code separated in two parts:
    - ▶ `core`: protocol for **MMC/eMMC** and **SD Card** as well as common functions for the framework
    - ▶ `host`: support for the controllers
        - ▶ `host/sdhci*` for the controller based on **SDHCI** maintained by **Adrian Hunter**
        - ▶ **SPI mode** supported in `host/mmc_spi.c` but currently without maintainer

- Maximum clock from 26MHz to 52MHz for **MMC**
- Maximum clock from 25MHz to 50MHz for **SD Card**
- Introduce the speed mode selection sequence using `CMD6`
- Introduce since **SD v2** and **MMC v4**

- ▶ Introduced with **SD 3.01** (2010)
- ▶ New speed modes (name are base on the bandwidth):
    - ▶ **SDR12** (max bandwidth: 12MB/s)
    - ▶ **SDR25** (max bandwidth: 25MB/s)
    - ▶ **SDR50** (max bandwidth: 50MB/s)
    - ▶ **SDR104** (max bandwidth: 104MB/s)
    - ▶ **DDR50** (max bandwidth: 50MB/s)
- ▶ All these new modes under 1.8V compared to the 3.3V for **DS** (Default Speed 25MHz) and **HS** (High Speed at 50MHz)
- ▶ New step in the switch sequence: modifying the voltage

- ▸ **SDR12**: simple data rate with clock at 25MHz (with 4 lines)
- ▸ **SDR25**: simple data rate with clock at 50MHz (with 4 lines)
- ▸ **SDR50**: simple data rate with clock at 100MHz (with 4 lines)
- ▸ **SDR104**: simple data rate with clock at 208MHz (with 4 lines).
  For this speed mode tuning (`CMD19`) is required
- ▸ **DDR50**: double data rate with clock at 50MHz. Data sample on each front of
  the clock

Speed mode improvement - **DDR** mode for **eMMC**

- Introduced with **MMC 4.4** (2009)
- Up to 52MHz (as high Speed mode)
- Configured with `CMD6` but with different arguments than **SD Card**
- Can be used at 3V
- At host controller level, same configuration used than for **DDR50**

Speed mode improvement - **HS-200**

- ▶ Introduced with **MMC 4.5** (2011)
- ▶ Up to 200MHz at single data rate
- ▶ Tuning command (**CMD21**) can be used to find optimal data sampling.
- ▶ Must be used at 1.8V or 1.2V

Speed mode improvement - **HS-400**

- Introduced with **MMC 5.0** (2013)
- Up to 200MHz at dual data rate
- New `DS` (Data Strobe) line: used during `DATA out` and `CRC response`
- Tuning command (**CMD21**) can be used to find optimal data sampling.
- Must be used at 1.8V or 1.2V
- With **MMC 5.1** (2014), **Enhanced Strobe** added: strobe also provided during `CMD Response`

Speed mode improvement - **UHS-II**

- ▶ Introduced with **SD 4.1** (2013)
- ▶ Completely new mode
- ▶ New set of signal: `RCLK+, RCLK-, D0+,D0-,D1+,D1-, VSS3-5, VDD1-2`
- ▶ 2 data lanes (`D0`, `D1`) using 2 differential signals
- ▶ `RCLK`: 26 to 52 MHZ
- ▶ Data x15 or x30 depending of the mode, up to 312MB/s
- ▶ Completely different protocol: exchange of packet messages on both way
- ▶ Each packet have header and payload data
- ▶ At transaction layer possibility to encapsulate SD packet
- ▶ At lower level still needed to be able to use the new protocol
- ▶ With **SD 6.0** (2017): **UHS-III** (624 MB/s)

- **DDR 50 mode** added with **2.6.37** in 2010
- **UHS-I** added with **3.0** in 2011
- **HS200** added with **3.10** in 2012
- **HS400** added with **3.16** in 2014
- **HS400 retuning** added with **4.2** in 2015
- **HS400es** (Enhanced Strobe) added with **4.8** in 2016

- ▶ Signal voltage switching needed for most of the new speed mode
  - ▶ Supported by the framework in `core.c`
  - ▶ Make use of the regulator framework
- ▶ Tuning used by **eMMC** and **SD Card**
  - ▶ Function present in the core
  - ▶ But implemented at controller driver level
- ▶ Switching sequence handled by the core but most of the steps can be customized for the host controller.

- **eMMC** speed mode support quite complete, most of the development now at driver level and specific to each controller.
- **SD Card**: no support at all for **UHS-II** (and **UHS III**), adding this new protocol would be a big task.

Questions?

Gregory CLEMENT
gregory@free-electrons.com

Slides under CC-BY-SA 3.0
http://free-electrons.com/pub/conferences/2017/elce/clement