Block Device Image (BDI) Interface for QL emulators

Specification of a simple and open interface for QL emulators, so native hardware drivers on the QL side can access block device images present on the emulator's host filesystem.

The original idea was to allow the driver for the QL-SD hardware to natively access SDHC cards via emulators also, with just simple changes.

BDI specifies a simple pseudo-hardware. The QL emulator emulates the presence of several registers, translating accesses into reads and writes of blocks within a file system image that could actually be on any drive.

Registers on a real hardware will differ. There are similarities, but especially the initialization remains completely different. The emulator doesn't have to care about an SDHC card at all, it simply provides read/write of 512-byte blocks into a normal, existing file.

(On QL hardware, QL-SD will only support SDHC cards. It used a native filesystem derived from an old QL harddisk driver called QL-HD. (This was replaced by a Qubide based driver.) An image of the filesystem is placed in the root directory of the first FAT32 partition of the SDHC card. Only one single unfragmented, fixed-size image per card, so things can be handled easily from from QL side.)

Register definitions:

BDI_SELECT, Offset $0, Byte, Write
Device Select Register

0=No device selected
1..8=Select device by number, BDI_SIZE and BDI_STATUS get updated

---

BDI_COMMAND, Offset $1, Byte, Write
Command Register

2=Execute Read at the block address contained in BDI_ADDRESS
3=Execute Write at the block address contained in BDI_ADDRESS
After the command, 512 accesses to BDI_DATA follow

BDI_STATUS, Offset $2, Byte, Read
Status Register
*** Optional. Not required for minimal implementation ***

Bit 0:
  0=Device image present
  1=Device image not attached
  Native driver could detect "medium change" by checking this regularly
Bit 1:
  0=OK
  1=Error, command could not be executed, or block address out of range

---

BDI_DATA, Offset $3, Byte, Read/Write
Data register

Following a Read/Write command, 512 accesses to this register actually
transfer a block of data.
Ignored, if block address was out of range or device not ready

---

BDI_ADDRESS, Offset $4, Long, Write
Block address register

Block address for an upcoming transfer, written before the command.
Size of block fixed at 512 Bytes ==> Theorectical capacity 2 Terabytes

---

BDI_SIZE, Offset $8, Long, Read
Size register

Size of device image, in blocks of 512 Bytes.
Zero, if device image not found.