Xen block backend/frontend multiqueue support

Original reference:
http://wiki.xenproject.org/wiki/GSoc_2014#Xen_block_backend.2Ffrontend_multiqueue_support
Mentor: Konrad Rzeszutek Wilk (konrad.wilk@oracle.com)
Intern: Arianna Avanzini (avanzini.arianna@gmail.com)

As detailed by the mentor himself, the project will mainly concern the study and implementation of the block multi-queue API support in Xen's paravirtualized block I/O driver. The project will then include the following activities; please note that the indicated activity durations are only an approximation of the actual timeline.

1. Study of the currently-available documentation about the multi-queue API currently implemented in the Linux kernel, such as papers[1] and on-line documents[2].
   [1 week]
2. Study of the multi-queue implementation itself, as included in the Linux kernel's source code (block/blk-mq* files in the Linux kernel tree).
   [1 week]
3. Study of the multi-queue implementation already available in the virtio drivers[3].
   [1 week]
4. First implementation of a simplified model of the blk-mq* API in the frontend driver (xen-blkfront), using a single queue with a fixed queue depth (44 or higher, if indirect descriptors[4] are enabled).
   [3 weeks]
5. Extension of the previously-implemented simplified model to allow for multiple queues. More in detail, two sub-activities need to be completed.
   5.1. First, the frontend driver (xen-blkfront) would need to interrogate the backend driver (xen-blkback) to obtain information about the number of available hardware queues. The retrieved data would then need to be piped up to xen-blkfront via XenBus, so that the frontend driver can negotiate with the backend driver the number of queues in the driver.
   5.2. Secondly, it would be finally necessary to implement in xen-blkback support for multiple rings, with one ring corresponding to each of the queues previously negotiated between frontend and backend drivers.
   [5 weeks]
6. Testing of the implemented API. This activity, in fact, should include the following sub-activities.
   6.1. Functional evaluation of the implemented API.

[1] http://kernel.dk/systor13-final18.pdf
[2] https://lwn.net/Articles/552904/
[3] https://lkml.org/lkml/2013/10/25/202
[4] http://blog.xen.org/index.php/2013/08/07/indirect-descriptors-for-xen-pv-disks/

6.2. Survey of the currently-available benchmarks for block I/O. Selection and/or extension of an existing benchmark, or implementation of an ad-hoc benchmark for the performance analysis of the newly-implemented API.

6.3. Performance analysis of the extended paravirtualized driver with respect to both the original performance of the driver and the performance of block I/O on bare metal. The main parameters of the performance analysis would be the overall throughput of the driver and the service latency of I/O requests.

[2 weeks]