Installation:

The following steps would help you to install and run the modified QEMU.

1. Setting Up ZFS

The modified QEMU uses ZFS to take and manage snapshots. Thus we need an installation of ZFS before we can use use qemu to take snapshots.
The following command installs zfs as a fuse module in ubuntu
$ **sudo apt-get install zfs-fuse**

The disk image we intend to use needs to be set up in a predefined manner to enable the qemu zfs plugin utilize it for storing snapshot information. The following needs to be done to set up a disk image.

i) *Create a raw disk image using dd*
$ ***dd if=/dev/zero of=zfsimage bs=512 count=2M***

ii) *Use the zpool command to create a zpool on it.*
For example the following command creates a zpool named *tankPool* on the disk *zfsimage* (Note that the absolute path of zfsimage has to be specified here).

zpool create tankPool /home/user/images/zfsimage

iii) *Create a zfs filesystem on it.*
The command below creates a filesystem named fs on tankPool

zfs create tankPool/fs

iv) *The place where the filesystem is mounted can be changed using the set mountpoint property*.We change the directory where the filesystem is mounted by default to a more convenient directory (linux-0.2) using the set mountpoint command. It is normally useful (though not required) to mount it under a directory named suitably (here linux-0.2).

zfs set mountpoint=/home/user/images/linux-0.2 tankPool/fs

v) *Create the following directories under the root directory where the filesystem is mounted.*
a) image
b) snapshot
mkdir /home/user/images/linux-0.2/image
 # mkdir /home/user/images/linux-0.2/snapshot

vi) *Once the directories are created, create the following files*

a) A regular text file within the root directory named **active.txt**
touch /home/user/images/linux-0.2/active.txt
b) A raw image file within **image** directory named as **disk**. This is the raw image format supported

by qemu. To convert an image file in a different format (say qcow2) to raw format, suitable qemu-img commands can be used.
The following command copies a pre existing **linux-0.2.img** file which already has a guest kernel image into the **image** directory.
cp linux-0.2.img /home/user/images/linux-0.2/image/disk

c) A regular file within the **snapshot** directory named as **snap**.
touch /home/user/images/linux-0.2/snapshot/snap

vii) Open the file **active.txt** in an editor, write the following lines and save it.
a) The filesystem name
b) The zpool name
For example, with the above set up, active.txt contains the following
cat /home/user/images/linux-0.2/active.txt
tankPool/fs
tankPool

2. Compiling QEMU
The qemu code included in the webpage comes as a part of the MARSS simulator (http://marss86.org/~marss86/index.php/Getting_Started). Compiling this source therefore requires the following steps.

i) *Setup Scons*
For compilation, the Scons compilation tool (version 1.2 or higher) is required. Scons can be easily set up using standard software installation tools such as apt-get or yum.
ii) *Compilation*
Once Scons is set up go to the directory containing the source code of MARSS and issue the compilation command.
For example
$ **cd $MARSS**
$ **scons -Q**

3. Clean Binaries
For cleaning up the compiled binaries, use
$ **scons -Q -c**

4. Applying the patch
If you decide on compiling the unmodified qemu and then apply the patch, then first go to the directory containing the source code of MARSS and then issue the patch command.
$ **cd $MARSS**
$ **patch -p2 < fast_snapshot.patch**

Running QEMU:

The modified qemu can be run in the following way.

./qemu/qemu-system-x86_64 -hda ***directory***

where ***directory*** is the directory where the zfs filesystem is mounted. Each such directory thus represents a single IDE hard disk 0/1 image. One thing to ensure here is that zfs shell commands should have permission to be executed. Running as the root user normally ensures that all required permissions are present.
The following command runs the linux image that we just created.

 # ./qemu/qemu-system-x86_64 -hda *~~/~~home/user/images/linux-0.2/*

The mechanism to capture and load snapshots remain the same with **savevm/loadvm** used the same way as they are in the original unmodified qemu version.

Loading from a specific snapshot:
Loading from a specific snapshot and preventing any changes to the underlying image can be done in the following way.

 # ./qemu/qemu-system-x86_64 -hda *directory* -loadvm *snapshot_name -loadsnap*

While the '-loadvm' option specifies the snapshot to load, the '-loadsnap' option ensures that all changes made to the underlying image while qemu runs are written only to temporary files and the original image remains unmodified.

FAQ:

1. Why do I need to have two raw image files, one on which a ZFS file system is created and another within the **image** directory?

Ans. The raw file on which the ZFS filesystem is created is an equivalent of any disk image file (such as .qcow2 or .img which is normally used by qemu). However this file, besides storing the guest image also saves the file that stores state information corresponding to a particular snapshot (snapshot/snap file).
Therefore the guest image information and snapshot state file are segregated into different directories under the filesystem root. The raw file created in step 1 is the one that stores both of these. The **disk** file (image/disk) on the other hand is one that contains just the actual guest image.

2. Can I start qemu by giving the filename on which the zfs filesystem was created as the hard disk image parameter instead of the **directory** name and expect the same behavior?

For example something like
./qemu/qemu-system-x86_64 -hda ***home/user/images/zfsimage***
instead of
./qemu/qemu-system-x86_64 -hda ***/home/user/images/linux-0.2/***
Ans. No You cannot. When qemu is started correctly with the root directory where the zfs filesystem is mounted as the parameter, the qemu zfs plugin handles reads and writes and everything works as expected. However, when just the raw file name is specified, it does not identify it as a file with a zfs filesystem on it. Failing to recognize which block driver to select, it selects the raw block driver by

default (the one used with raw .img files). Therefore things dont work as expected.

3. What behavior should I expect I take a snapshot when my qemu is running with the **loadsnap** option ?

Ans. When qemu is started with the **loadsnap** option, the original image file is expected to remain unchanged and writes affect only temporary files. However if a snapshot is taken at this stage, then the snapshot persists and can be loaded anytime in the future.

4. Can I run multiple qemu instances using the same image file at the same time ?

Ans. Yes. With the loadsnap option multiple qemu processes can be started with each specifying the same disk image. Since the writes go to temporary files, changes made in the image through one qemu process are *invisible* to others.

5. Can the **loadsnap** option be used without the **loadvm** option ?

Ans. The behavior of **loadsnap** without the loadvm option is undefined.