# ASUS eee PC HowTo

Although this howto was written specifically for the ASUS 1005HA netbook, it may be ported to similar machines.

With Slackware everything works pretty much out of the box for this Asus EEE-pc model. Function keys, however, are not enabled by default, which means additional configuration steps are required.

The scope of this document is to provide a guide for such steps. This document assumes some basic knowledge of Slackware configuration procedures available here.

## Powersaving and ACPI

Asus eee PCs are/were popular netbooks. Some linux distros were built specifically for these machines and specific scripts where ported to Slackware.

### Enabling ACPI

To enable ACPI, the following parameter should be passed to the kernel:

Append="Acpi_osi=Linux vt.default_utf8=0"

> 💡 I recommend to add this parameters using liloconfig logged as root.

### EEE PC acpi scripts

Eric Hameleers (aka AlienBOB) had created a Slackware package that will enable most of the FN keys functions; however, two FN key functions that I personally find impossible to live without based on my preferences and usage are the FN+F3 (disable touchpad) and FN+Space bar (Super Hybrid Engine).

These could be easily enabled by doing the following:

#### Touchpad toggle function

Create a text file in /etc/acpi/events/ containing the following code:

```
event=hotkey ATKD 00000037
action=/etc/acpi/actions/touchpad-toggle.sh
```

In my case, the file is named touchpad-toggle. This particular file identifies an action once an event is triggered. The "00000037" points specifically to the event of pressing FN+F3 keys combination of the model specified.

> **note** To identify the correct code for your keys use Acpi_listen

The **Action** file /etc/acpi/actions/touchpad-toggle.sh:

```bash
#!/bin/bash
#Toggle touchpad

# do nothing if package is removed
PKG=eeepc-acpi-scripts
FUNC_LIB=/usr/share/$PKG/functions.sh
DEFAULT=/etc/default/$PKG
[ -e $FUNC_LIB ] || exit 0

. $FUNC_LIB
. /etc/acpi/lib/notify.sh
action=$1


SYNSTATE=$(synclient -l | grep TouchpadOff | awk '{ print $3}')
# change state
   if [ $SYNSTATE = 0 ]; then
        synclient touchpadoff=1
        msg="Touchpad OFF"
            elif [ $SYNSTATE = 1 ]; then
        synclient touchpadoff=0
        msg="Touchpad ON"
   fi

notify touchpad "$msg"
```

> ⚠ Do not forget to give execute permissions to the actions files (as root) chmod +x /etc/acpi/actions/touchpad-toggle.sh

**Enabling SHE function**

"Super Hybrid Engine" as it's known under Windows has a significant effect on powersaving. This underclocks the FSB for powersave/overclocks for performance. This is achieved by laptop-mode-tools.

> 💡 I recommend using the SlackBuild script available at http://slackbuilds.org/repository/14.0/system/laptop-mode-tools/

Once laptop-mode-tools is installed we could proceed with enabling the control of the FSB by pressing FN+Space bar combo by creating a file in the /etc/acpi/events directory just as we did before:

```
event=hotkey ATKD 00000039
action=/etc/acpi/actions/she.sh
```

in my case, the file is /etc/acpi/events/cpu.

The **Action** file /etc/acpi/actions/she.sh:

```bash
#!/bin/bash
#Toggle touchpad

# do nothing if package is removed
PKG=eeepc-acpi-scripts
FUNC_LIB=/usr/share/$PKG/functions.sh
DEFAULT=/etc/default/$PKG
[ -e $FUNC_LIB ] || exit 0

. $FUNC_LIB
. /etc/acpi/lib/notify.sh
action=$1

she_names=("performance" "normal" "powersave")
she_value_performance=("0" "0x300")
she_value_normal=( "1" "0x301")
she_value_powersave=( "2" "0x302")

#find current
SYS_DEVICE="/sys/devices/platform/eeepc/cpufv"

she_current=`cat ${SYS_DEVICE}`

if [[ "${she_current}" == "${she_value_performance[1]}" ]]; then
she_next=${she_value_normal[0]}
echo $she_current

elif [[ "${she_current}" == "${she_value_normal[1]}" ]]; then
she_next=${she_value_powersave[0]}
echo $she_current

elif [[ "${she_current}" == "${she_value_powersave[1]}" ]]; then
she_next=${she_value_performance[0]}
echo $she_current
fi

function she_toggle() {
    if [[ "${she_names[${she_current/0x30/}]}" !=
"${she_names[${she_next}]}" ]]; then
msg="Super Hybrid Engine: ${she_names[${she_next}]}"
        echo ${she_next} > ${SYS_DEVICE} && msg="Super Hybrid Engine:
${she_names[${she_current/0x30/}]} to ${she_names[${she_next}]}" cpu &
    else
msg="Super Hybrid Engine: Already at ${she_names[${she_next}]}"
```

```
        fi
}


#############################################################
case $1 in
    "debug")
        print_generic_debug
    ;;
    "powersave")
        she_next=2
        she_toggle
    ;;
    "normal")
        she_next=1
        she_toggle
    ;;
    "performance")
        she_next=0
        she_toggle
    ;;
    *)
        she_toggle
    ;;
esac


notify cpu "$msg"
```

> ⚠️ Do not forget to give execute permissions to the actions files (as root) chmod +x /etc/acpi/actions/she.sh

# Video card speed tune up

I've recently learned that this Asus 1005HA Intel chip 945GM/GMS/GME is slightly under clocked in order to provide longer battery life according to the SHE profile. On Windows, the default clock-speed is 166MHz in powersaving mode and 200MHz in High/Super mode. This chip, however, is able to run at 400MHz.

> 📝 I've only found necessary to set the clock at high speed when using an external monitor with extended desktop while running KDE. I've left the video card clock speed settings separated form the SHE toggle acpi scripts for that specific reason.

The following script will set GMA chipset speeds (run it as root):

```
#!/bin/sh
```

```
case "$1" in
    250) val=31 ;;
    400) val=33 ;;
    200) val=34 ;;
    *)
        echo "Possible values are: 200, 250, 400." >&2
        exit 1
    ;;
esac

setpci -s 02.0 f0.b=00,60
setpci -s 02.0 f0.b=$val,05
```

In my case, I called it gma-clock… You just need to call it with the desired speed options you as argument. The only three valid options are 200, 250 and 400.

# Sources

1. Arch linux forums
2. linuxquestions

Originally written by weput aka kernel-P4N1C

howtos, asus, eee