

Sunday, September 2. 2007

far, far away

Nobody wants to control the multimedia system with an ordinary Keyboard, especially I've need only 6 keys to control the device. Not even I. So it's get time to install the asus digital home remote control. As I've mentioned already, with that remote and its usb-dongle, you can wakeup your system from suspend (from disk only!), it has also the 5 direction buttons (up, down, left, right, ok) and some more special ones, we can use for other things (like toggle fullscreen, quit elisa and display the menu). Generally for Remote Controls, there is the LIRC-Project, which is by elisa as an input device. That is very good, because this project also support this remote control. Perfect! Okay. Lets get it on! Set up LIRC generally The first thing to do, is to set up lirc itself. In older days, you had to recompile your kernel or at least compile certain modules. That was an ugly process. Today there are some remotes available for which the installation is really easy. Ours is one of them. Simply install lirc with this command-line: `sudo aptitude install lirc` and you'll get ask, which dongle you have. Select there Asus Digital Home, press enter and well, that it at first. This installation is putting a `lircd.conf` and some other files into the directory `/dev/lirc`. There is also one called `hardware.conf`. That one is very important. But first, let's test the remote control. To do this, there is a simple programm in the lirc-package. It is called `irw` and you may start it from the commandline/a shell. At first nothing happens, but If you press a button on the remote-control (expecting that the dongle is plugged in before [!!] starting `irw` and that you are pointing to this dongle), there should appear some lines, ending with `'AsusDH'`. If that is the case, everything is alright and you can go on with installing `pylirc`. Don't worry if it didn't for you, because it didn't for me either. That's bad, but there is a good reason. Okay, well, it is not `_good_` but it better than having the problem without any reason . The reason is, that HAL is detecting the device as a HID-Device and is loading the `hiddev`-driver (you can see that, if you use the `hal-device-manager`). For some would the solution be to prevent `hiddev` from beeing loaded system wide, but for me that is not a solution, because currently I've also a Keyboard plugged in, which is using the `hiddev`. In fact, HAL should not detect the `asusdh` as a hid-device but as an lirc-thing, but it does not yet, so let's change the lirc configuration to use the `hiddev` device the lirc device instead. That is very easy, just open the `/etc/lirc/hardware.conf` in your favorite editor and set the variable `'DEVICE'` to your `hiddev` device. Normally this should be `/dev/usb/hiddev0`, for me it was `/dev/usb/hiddev1` because my keyboard is loaded first. Don't forget that you've set this variable, because If you remove the keyboard, stick in another one HID-Device (like a keyboard) or you change the position of your USB-Dongle, the order may change and you'll wonder, why your remote control is not working anymore. Now restart lirc and test again with `irw`: `sudo /etc/init.d/lircd restart irw` If there is still nothing happening, you have not selected the correct device file. If it reacts, then go on with installing `pylirc`. Installing `pylirc` is really easy. You just go here and download the latest tarball (the `tar.gz`-file). Don't use the debian packages, they are too old for gutsy and you'll always get annoying messages. Unpack it, and go into the directory. You need a working build-essential enviroment (but I think I've already one for pigment). You also need the `lirc-dev` packages, which you can install by doing `sudo aptitude install liblircclient-dev` You also need the `python-setuptools` (and you should already have installed them for elisa!). Then you can do: `sudo python setup.py install` If there are no errors, then let's check the installation by doing starting the python shell (just type `python` in a shell) and in there you type (and press enter) `import pylirc` If nothing seems to happen, everything is alright and you can leave the shell with `quit()`. Otherwise the installation of `pylirc` didn't work and you should retry it. Now elisa is generally able to handle lirc-events. Thats cool. Set up lirc for elisa Now we replace the file `/etc/lirc/lircd.conf` with this (you have to be root to do that!):

```
## brand:      Asus# model:      Asus DH Deluxe IR#begin
remote name   AsusDH bits    32 pre_data_bits 32 pre_data 0xFF000000 post_data_bits 0 begin codes
POWER        0x01 QUICK_POWER 0x02 NOISE_OFF 0x03 WIFI 0x04 MENU
0x05 RED     0x06 UP      0x07 LEFT  0x08 OK    0x09 RIGHT 0x0A DOWN
0x0B end codesend remote
```

 That makes it easier for us, to use the file in elisa. We now don't have to change anything in the `elisa.conf`-file. I hope that this step is obsolete very soon because of new signal-naming, but actually it is necessary. Okay. restart lirc and try again with `irw`. It is working there, you can start your elisa (be sure that you didn't remove the lirc-stuff from the lirc-configuration file yet!) and use it with the asus DH: most keys have the place you would expect, but there are some you should know about: `power-key`: quits elisa (only) `ap-launch`: toggle the menu fullscreen: well, what would you expect? The other three keys are not yet used, because elisa does actually not know any more keys (and has no actions for them). Keep on playing! The elisa lirc configuration section Well, in most cases you should not need to set there anything for this remote control, but if I'm already speacking about lirc, I can not skip this . Currently (0.3.1) there are these values in the section `[base:lirc_input]` # filename of the LIRC config map to use `lirc_rc = 'streamzap.lirc'` `delay = '4'` `repeat = '1'` The first option is the `lirc_rc` file, which elisa should load to know, which key of the remote control should point to which elisa internal key. This is hopefully obsolete very soon, with the signal-naming project of the UMC! Then you'll not have more than one set of keys for all remotes and we can 'hardcode' them. The second and the third one are simply the things to overwrite in the configuration-file of lirc. That means, if you change the lirc-file in the first option, you should not forget to set the `dealy` and `repeat` option in there. You should only touch this

files, if you know what you are doing! But changing this parameters (for the default configuration) is very usefull. They are just overwriting the values of lirc-configuration and so I'll only quote, what they say about it:repeat: tells the program what shall happen if a key is repeated. A value of zero tells the program to ignore repeated keys. Any other positive value 'n' tells the program to pass the config string every 'n'-th time to the according application, when a key is repeated. The default for repeat is zero.delay: tells the program to ignore the specified number of key repeats before using the "repeat" configuration directive above. This is used to prevent double triggers of events when using a fast repeat rate. A value of zero, which also is the default, will disable the delay function. Both are really usefull! If you change the values your remote comes more or less sensitiv. The default values are already very usefull, but for certain remotes, you may want to change it (like for the apple-remote...).Have a lot of fun. And as always: do not hesitate to comment this article!

Posted by Benjamin Kampmann in chaos at 18:40