# Building support for a media device

The first step is to download the Kernel's source code, either via a distribution-specific source file or via the Kernel's main git tree[1].

Please notice, however, that, if:

- you're a braveheart and want to experiment with new stuff;
- if you want to report a bug;
- if you're developing new patches

you should use the main media development tree `master` branch:

https://git.linuxtv.org/media_tree.git/

In this case, you may find some useful information at the LinuxTv wiki pages:

https://linuxtv.org/wiki/index.php/How_to_Obtain,_Build_and_Install_V4L-DVB_Device_Drivers

[1] The upstream Linux Kernel development tree is located at https://git.kernel.org/pub/scm/li nux/kernel/git/torvalds/linux.git/

## Configuring the Linux Kernel

You can access a menu of Kernel building options with:

```
$ make menuconfig
```

Then, select all desired options and exit it, saving the configuration.

The changed configuration will be at the `.config` file. It would look like:

```
...
# CONFIG_RC_CORE is not set
# CONFIG_CEC_CORE is not set
CONFIG_MEDIA_SUPPORT=m
CONFIG_MEDIA_SUPPORT_FILTER=y
...
```

The media subsystem is controlled by those menu configuration options:

```
Device Drivers --->
    <M> Remote Controller support  --->
    [ ] HDMI CEC RC integration
    [ ] Enable CEC error injection support
    [*] HDMI CEC drivers  --->
    <*> Multimedia support  --->
```

The `Remote Controller support` option enables the core support for remote controllers[2].

The `HDMI CEC RC integration` option enables integration of HDMI CEC with Linux, allowing to receive data via HDMI CEC as if it were produced by a remote controller directly connected to the machine.

The `HDMI CEC drivers` option allow selecting platform and USB drivers that receives and/or transmits CEC codes via HDMI interfaces[3].

The last option (`Multimedia support`) enables support for cameras, audio/video grabbers and TV.

The media subsystem support can either be built together with the main Kernel or as a module. For most use cases, it is preferred to have it built as modules.

> **Note**
>
> Instead of using a menu, the Kernel provides a script with allows enabling configuration options directly. To enable media support and remote controller support using Kernel modules, you could use:
>
> ```
> $ scripts/config -m RC_CORE
> $ scripts/config -m MEDIA_SUPPORT
> ```

[2] `Remote Controller support` should also be enabled if you want to use some TV card drivers that may depend on the remote controller core support.

[3] Please notice that the DRM subsystem also have drivers for GPUs that use the media HDMI CEC support.

Those GPU-specific drivers are selected via the `Graphics support` menu, under `Device Drivers`.

When a GPU driver supports HDMI CEC, it will automatically enable the CEC core support at the media subsystem.

## Media dependencies

It should be noticed that enabling the above from a clean config is usually not enough. The media subsystem depends on several other Linux core support in order to work.

For example, most media devices use a serial communication bus in order to talk with some peripherals. Such bus is called IÂ²C (Inter-Integrated Circuit). In order to be able to build support for such hardware, the IÂ²C bus support should be enabled, either via menu or with:

```
./scripts/config -m I2C
```

Another example: the remote controller core requires support for input devices, with can be enabled with:

```
./scripts/config -m INPUT
```

Other core functionality may also be needed (like PCI and/or USB support), depending on the specific driver(s) you would like to enable.

## Enabling Remote Controller Support

The remote controller menu allows selecting drivers for specific devices. It's menu looks like this:

```
--- Remote Controller support
<M>   Compile Remote Controller keymap modules
[*]   LIRC user interface
[*]     Support for eBPF programs attached to lirc devices
[*]   Remote controller decoders  --->
[*]   Remote Controller devices  --->
```

The `Compile Remote Controller keymap modules` option creates key maps for several popular remote controllers.

The `LIRC user interface` option adds enhanced functionality when using the `lirc` program, by enabling an API that allows userspace to receive raw data from remote controllers.

The `Support for eBPF programs attached to lirc devices` option allows the usage of special programs (called eBPF) that would allow aplications to add extra remote controller decoding functionality to the Linux Kernel.

The `Remote controller decoders` option allows selecting the protocols that will be recognized by the Linux Kernel. Except if you want to disable some specific decoder, it is suggested to keep all sub-options enabled.

The `Remote Controller devices` allows you to select the drivers that would be needed to support your device.

The same configuration can also be set via the `script/config` script. So, for instance, in order to support the ITE remote controller driver (found on Intel NUCs and on some ASUS x86 desktops), you could do:

```
$ scripts/config -e INPUT
$ scripts/config -e ACPI
$ scripts/config -e MODULES
$ scripts/config -m RC_CORE
$ scripts/config -e RC_DEVICES
$ scripts/config -e RC_DECODERS
$ scripts/config -m IR_RC5_DECODER
$ scripts/config -m IR_ITE_CIR
```

## Enabling HDMI CEC Support

The HDMI CEC support is set automatically when a driver requires it. So, all you need to do is to enable support either for a graphics card that needs it or by one of the existing HDMI drivers.

The HDMI-specific drivers are available at the `HDMI CEC drivers` menu[4]:

```
--- HDMI CEC drivers
< >   ChromeOS EC CEC driver
< >   Amlogic Meson AO CEC driver
< >   Amlogic Meson G12A AO CEC driver
< >   Generic GPIO-based CEC driver
< >   Samsung S5P CEC driver
< >   STMicroelectronics STiH4xx HDMI CEC driver
< >   STMicroelectronics STM32 HDMI CEC driver
< >   Tegra HDMI CEC driver
< >   SECO Boards HDMI CEC driver
[ ]     SECO Boards IR RC5 support
< >   Pulse Eight HDMI CEC
< >   RainShadow Tech HDMI CEC
```

[4] The above contents is just an example. The actual options for HDMI devices depends on the system's architecture and may vary on new Kernels.

## Enabling Media Support

The Media menu has a lot more options than the remote controller menu. Once selected, you should see the following options:

```
--- Media support
[ ] Filter media drivers
[*] Autoselect ancillary drivers
    Media device types --->
    Media core support --->
    Video4Linux options --->
    Media controller options --->
    Digital TV options --->
    HDMI CEC options --->
    Media drivers --->
    Media ancillary drivers --->
```

Except if you know exactly what you're doing, or if you want to build a driver for a SoC platform, it is strongly recommended to keep the `Autoselect ancillary drivers` option turned on, as it will auto-select the needed IÂ²C ancillary drivers.

There are now two ways to select media device drivers, as described below.

### `Filter media drivers` menu

This menu is meant to easy setup for PC and Laptop hardware. It works by letting the user to specify what kind of media drivers are desired, with those options:

```
[ ] Cameras and video grabbers
[ ] Analog TV
[ ] Digital TV
[ ] AM/FM radio receivers/transmitters
[ ] Software defined radio
[ ] Platform-specific devices
[ ] Test drivers
```

So, if you want to add support to a camera or video grabber only, select just the first option. Multiple options are allowed.

Once the options on this menu are selected, the building system will auto-select the needed core drivers in order to support the selected functionality.

> **Note**
>
> Most TV cards are hybrid: they support both Analog TV and Digital TV.
>
> If you have an hybrid card, you may need to enable both `Analog TV` and `Digital TV` at the menu.

When using this option, the defaults for the media support core functionality are usually good enough to provide the basic functionality for the driver. Yet, you could manually enable some desired extra (optional) functionality using the settings under each of the following `Media support` sub-menus:

```
Media core support --->
Video4Linux options --->
Media controller options --->
Digital TV options --->
HDMI CEC options --->
```

Once you select the desired filters, the drivers that matches the filtering criteria will be available at the `Media support->Media drivers` sub-menu.

### `Media Core Support` menu without filtering

If you disable the `Filter media drivers` menu, all drivers available for your system whose dependencies are met should be shown at the `Media drivers` menu.

Please notice, however, that you should first ensure that the `Media Core Support` menu has all the core functionality your drivers would need, as otherwise the corresponding device drivers won't be shown.

## Example

In order to enable modular support for one of the boards listed on :doc:`this table <cx231xx-cardlist>`, with modular media core modules, the `.config` file should contain those lines:

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\admin-guide\media\(linux-master)(Documentation)(admin-guide)(media)building.rst`, line 276); *backlink***
>
> Unknown interpreted text role "doc".

```
CONFIG_MODULES=y
CONFIG_USB=y
CONFIG_I2C=y
CONFIG_INPUT=y
CONFIG_RC_CORE=m
CONFIG_MEDIA_SUPPORT=m
CONFIG_MEDIA_SUPPORT_FILTER=y
CONFIG_MEDIA_ANALOG_TV_SUPPORT=y
CONFIG_MEDIA_DIGITAL_TV_SUPPORT=y
CONFIG_MEDIA_USB_SUPPORT=y
CONFIG_VIDEO_CX231XX=y
CONFIG_VIDEO_CX231XX_DVB=y
```

## Building and installing a new Kernel

Once the `.config` file has everything needed, all it takes to build is to run the `make` command:

```
$ make
```

And then install the new Kernel and its modules:

```
$ sudo make modules_install
$ sudo make install
```

## Building just the new media drivers and core

Running a new development Kernel from the development tree is usually risky, because it may have experimental changes that may have bugs. So, there are some ways to build just the new drivers, using alternative trees.

There is the Linux Kernel backports project, with contains newer drivers meant to be compiled against stable Kernels.

The LinuxTV developers, with are responsible for maintaining the media subsystem also maintains a backport tree, with just the media drivers daily updated from the newest kernel. Such tree is available at:

https://git.linuxtv.org/media_build.git/

It should be noticed that, while it should be relatively safe to use the `media_build` tree for testing purposes, there are not warranties that it would work (or even build) on a random Kernel. This tree is maintained using a "best-efforts" principle, as time permits us to fix issues there.

If you notice anything wrong on it, feel free to submit patches at the Linux media subsystem's mailing list: media@vger.kernel.org. Please add `[PATCH media-build]` at the e-mail's subject if you submit a new patch for the media-build.

Before using it, you should run:

```
$ ./build
```

> **Note**
>
> 1. you may need to run it twice if the `media-build` tree gets updated;
> 2. you may need to do a `make distclean` if you had built it in the past for a different Kernel version than the one you're currently using;
> 3. by default, it will use the same config options for media as the ones defined on the Kernel you're running.

In order to select different drivers or different config options, use:

```
$ make menuconfig
```

Then, you can build and install the new drivers:

```
$ make && sudo make install
```

This will override the previous media drivers that your Kernel were using.