



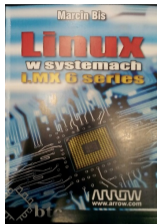
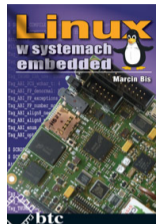
Exploring Linux Kernel Source Code with Eclipse and QtCreator

Marcin Bis

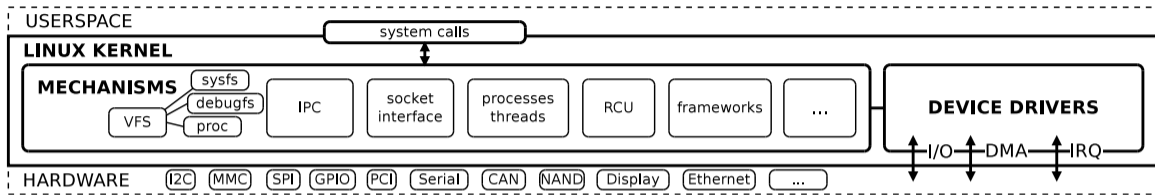
 **BIS-LINUX.COM**

2016.10.12
ELCE 2016, Berlin

- marcin@bis-linux.com
+48 607 075 848
- Training (<http://bis-linux.com>):
 - Embedded Linux
 - Yocto
 - **Linux Device Drivers Development**
 - Real-Time Linux
- Professional services - Technical support
located: Warszawa, Poland



The problem



Linux kernel consists of vast amount of source code.

- 1 ... need to have a quick look into some framework or driver to debug a problem
- 2 ... need to introduce the project to newcomers (training participants)

Kernel mechanisms

- Well designed code (with performance/latency/portability in mind)
- Finesse algorithms

Device Drivers

- Code follows specification (not so much space for fancy algorithms)
- Lots of boilerplate code

The solution

- VIM or EMACS
- exuberant-ctags
ctags parses source code and produces a sort of index mapping the names of significant entities (e.g. functions, classes, variables) to the location where that entity is defined
- cscope
Cscope is an interactive text screen based source browsing tool
- LXR - *Linux Cross Referece*
<http://lxr.free-electrons.com>
- IDE: Eclipse
- IDE: Qt Creator
- IDE: KDevelop

Non open-source solution:

- Commercial IDE (\$\$\$).

Commercial IDE example: Timesys Timestorm



- Eclipse based.
- Includes proprietary plugins.
- Per-seat licensing.

The screenshot shows an IDE window titled "C/C++ - TimeStorm". The "File" menu is open, and "Linux Kernel Project" is highlighted. Below it, a dialog box titled "Linux Kernel Project" is displayed. The dialog has the following fields and options:

- Project Name:** my-module
- Create project using:** SDK
- SDK Type:** Factory SDK Yocto SDK Other SDK
- SDK:** factory-wandboard_quad-armv7l-timesys-linux-gnueabi-gcc-4.8.3-20140626-17:45
- SDK Kernel Source:** /home/marcin/timesys/wandboard_quad/kernel-source/linux-3.0
- External Kernel Sources
- External Kernel Source:** [text input field] [Archive...] [Directory...]

At the bottom of the dialog, there are buttons for "< Back", "Next >", "Cancel", and "Finish".

Eclipse

Eclipse CDT - Eclipse IDE for C/C++ Developers

<http://eclipse.org/>

- Eclipse from Ubuntu package:

```
aptitude install eclipse-cdt eclipse-cdt-launch-remote
```

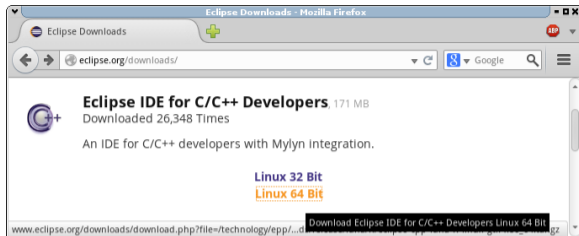
Ubuntu 14.04 LTS contains - Juno (3.8) ... Ubuntu 16.04 LTS contains - Juno (3.8) (the same)



- Use newer version:

Kepler (4.3), Luna (4.4), Mars (4.5), Neon (4.6)





```
$ aptitude install openjdk-8-jdk  
$ update-alternatives --config java
```

- System must have proper version of Java SDK installed:
 - for Kepler minimal version is 6
 - for Luna - 7
 - for Mars and Neon - 8

```
$ tar -xf eclipse-cpp-neon-1-linux-gtk-x86_64.tar.gz
```

```
$ source /opt/poky/2.1.1/\
environment-setup-cortexa9hf-neon-poky-linux-gnueabi
$ export export SWT_GTK3=0
$ ./eclipse/eclipse &
```

- Hint: set compiler `PATH` before running IDE - it will inherit exported settings. Just `export PATH=...` if using Linaro toolchain.
- Hint: in Xubuntu, SWT + GTK3 looks ugly, force using GTK2 (not needed in case of Debian, Fedora, SUSE)

Prerequisites

- Kernel sources has to be configured.
- Initial steps of build process has to be executed.

Why?

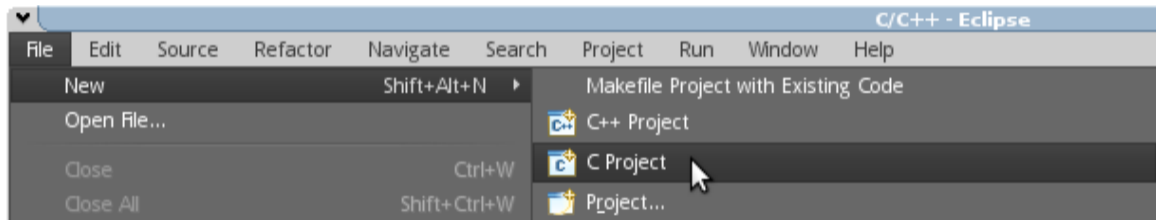
include/generated

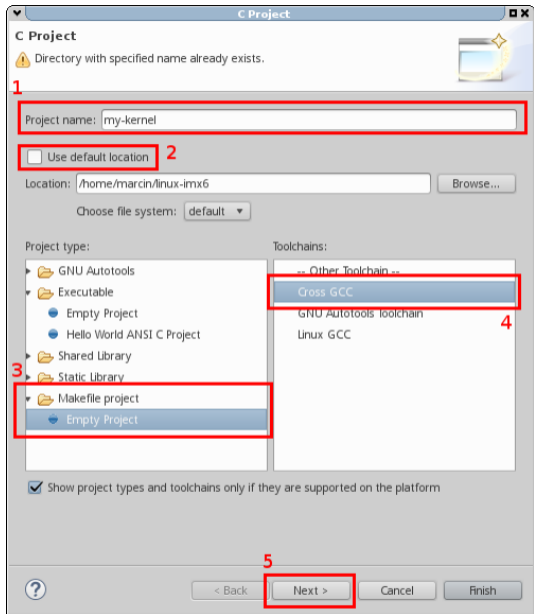
- include/generated/autoconf.h
Contains kernel configuration generated from `.config`:

```
#define CONFIG_RING_BUFFER 1
#define CONFIG_HAVE_ARCH_SECCOMP_FILTER 1
#define CONFIG_SND_PROC_FS 1
#define CONFIG_SCSI_DMA 1
#define CONFIG_KERNEL_GZIP 1
#define CONFIG_ATAGS 1
#define CONFIG_INPUT_KEYBOARD 1
...

```

New project

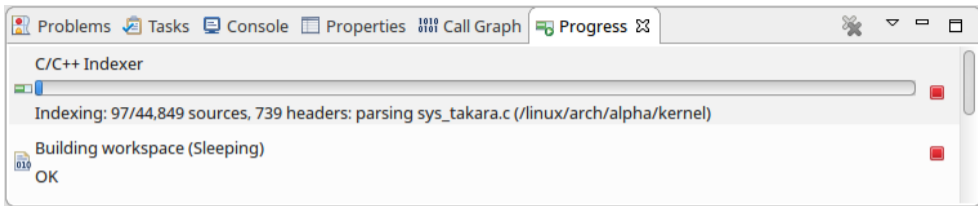




New -> C Project

- Set project name (it is just for Eclipse)
 - Project directory will be created in `workspace`, indexes will be stored here.
- Do not use default location (workspace).
Point to directory containing kernel sources
 - Project configuration will be stored here.
- Kernel is build using `make`, so choose `Makefile` project.
- ... and it is using cross-toolchain.

Project is created...



...but IDE is barely usable:

```
223 static struct platform_driver w1_gpio driver = {
224     .driver = {
225         .name    = "w1-gpio",
226         .pm      = &w1_gpio_pm_ops,
227         .of_match_table = of_match_ptr(w1_gpio_dt_ids),
228     },
229     .probe = w1_gpio_probe,
230     .remove = w1_gpio_remove,
231 };
232
233 module_platform_driver(w1_gpio driver);
234
235 MODULE_DESCRIPTION("GPIO w1 bus master driver");
236 MODULE_AUTHOR("Ville Syrjälä <syrjala@sci.fi>");
237 MODULE_LICENSE("GPL");
```

- ▶ w1-gpio.c
- ▶ built-in.o - [arm/le]
- ▶ w1-gpio.o - [arm/le]
- Kconfig
- Makefile
- modules.builtin
- modules.order
- ▶ slaves
- ▶ w1_family.c
- ▶ w1_family.h
- ▶ w1_int.c
- ▶ w1_int.h
- ▶ w1_io.c
- ▶ w1_log.h
- ▶ w1_netlink.c
- ▶ w1_netlink.h

```

113 static int w1_gpio_probe(struct platform_device *pdev)
114 {
115     struct w1_bus_master *master;
116     struct w1_gpio_platform_data *pdata;
117     int err;
118
119     if (of_have_populated_dt()) {
120         err = w1_gpio_probe_dt(pdev);
121         if (err < 0)
122             return err;
123     }
124
125     pdata = dev_get_platdata(&pdev->dev);
126

```

- linux/of_platform.h
- linux/of_gpio.h
- linux/err.h
- linux/of.h
- linux/delay.h
- ../w1.h
- ../w1_int.h
- ^S w1_gpio_set_pullup(v
- ^S w1_gpio_write_bit_dii

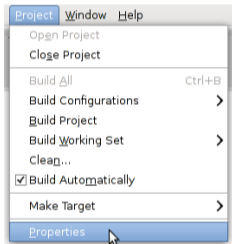
Problems Tasks Console Properties Call Graph Progress

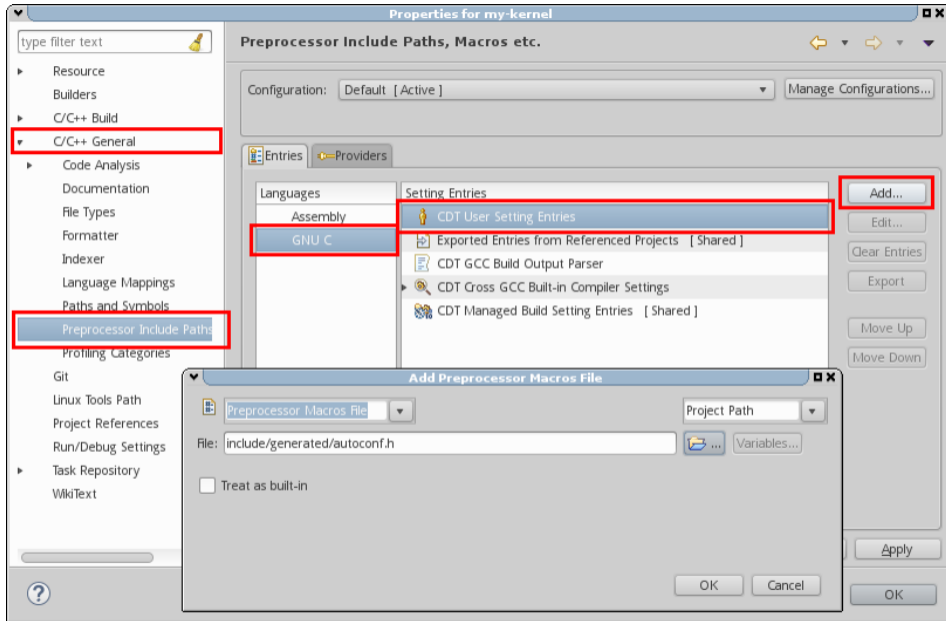
7 errors, 3 warnings, 0 others

Description	Resource	Path	Location	Type
▼ ✖ Errors (7 items)				
✖ Field 'data' could not be resolved	w1-gpio.c	/linux/drivers/w1/r	line 156	Semantic Error
✖ Field 'read_bit' could not be resolved	w1-gpio.c	/linux/drivers/w1/r	line 157	Semantic Error

Could not find symbol 'w1_bus_master' in index Writable Smart Insert 115 : 19

Modify project settings





Force-include
include/
generated/
autoconf.h
for every file
parsed by
indexer.

Add it as
Preprocessor
macros
file

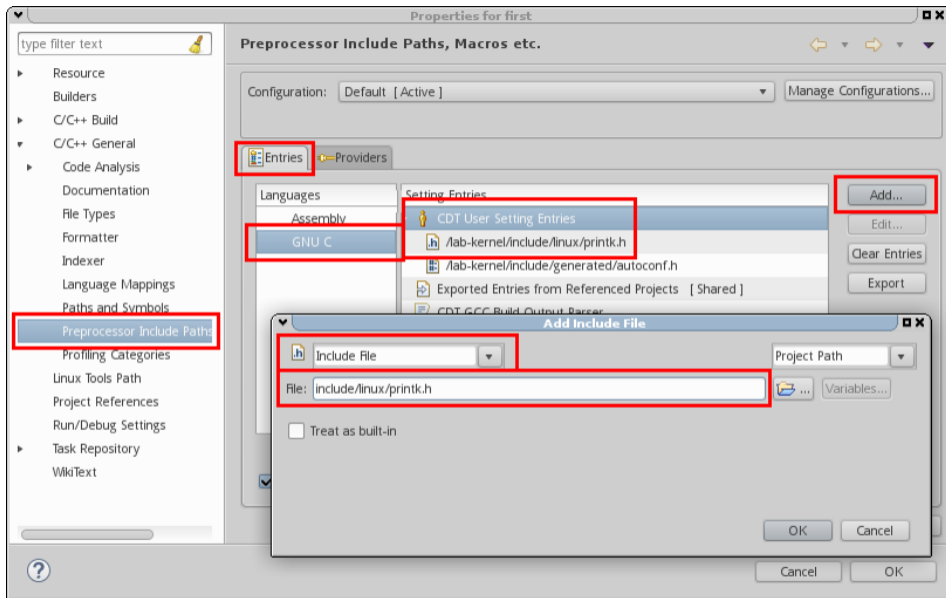
include/
linux/
kconfig.h
for newer
kenels

Before:

```
w1-gpio.c
68 }
69
70 #if defined(CONFIG_OF)
71 static const struct of_device_id w1_gpio_dt_ids[] = {
72     { .compatible = "w1-gpio" },
73     {}
74 };
75 MODULE_DEVICE_TABLE(of, w1_gpio_dt_ids);
76 #endif
77
78 static int w1_gpio_probe_dt(struct platform_device *pdev)
79 {
80     struct w1_gpio_platform_data *pdata = dev_get_platdata(&pdev->dev);
81     struct device_node *np = pdev->dev.of_node;
82     int gpio;
83
84     pdata = devm_kzalloc(&pdev->dev, sizeof(*pdata), GFP_KERNEL);
85     if (!pdata)
86         return -ENOMEM;
87
88     if (of_get_property(np, "linux,open-drain", NULL))
89         pdata->is_open_drain = 1;
```

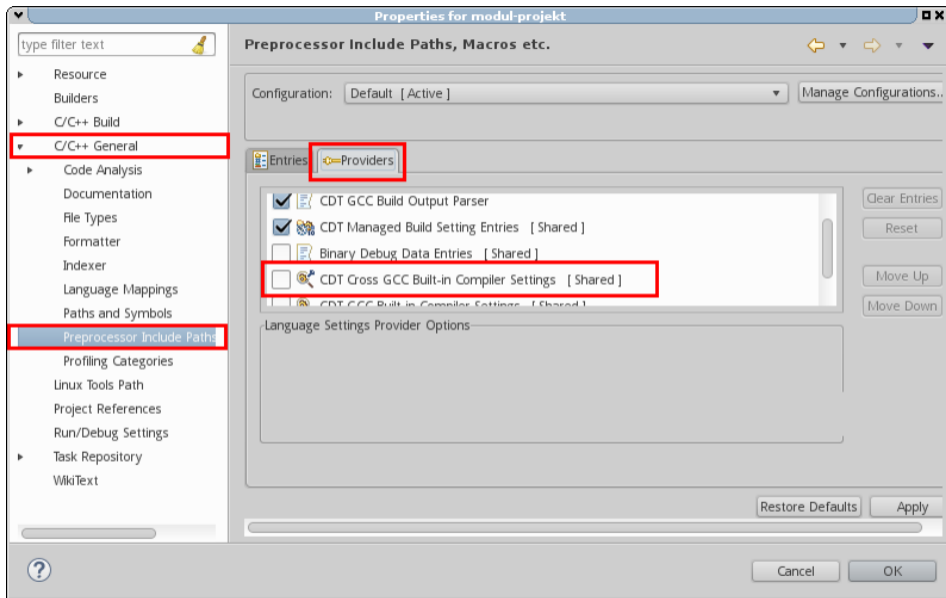
After:

```
w1-gpio.c
67     return gpio_get_value(pdata->pin) ? 1 : 0;
68 }
69
70 #if defined(CONFIG_OF)
71 static const struct of_device_id w1_gpio_dt_ids[] = {
72     { .compatible = "w1-gpio" },
73     {}
74 };
75 MODULE_DEVICE_TABLE(of, w1_gpio_dt_ids);
76 #endif
77
78 static int w1_gpio_probe_dt(struct platform_device *pdev)
79 {
80     struct w1_gpio_platform_data *pdata = dev_get_platdata(&pdev->dev);
```



Force-include
include/
linux/
printk.h for
every file
parsed by
indexer.

Depending on
the kernel
version, other
files may be
needed (e.g.
sched.h



Eclipse calls toolchain to get default include directory and defines.

Kernel does not use standard include directories.

```
$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi- zImage V=1
arm-linux-gnueabi-gcc -Wp,-MD,drivers/mmc/host/.sdhci.o.d -nostdinc -isystem
../lib/gcc/arm-linux-gnueabi/5.3.1/include -I./arch/arm/include
-I./arch/arm/include/generated/uapi -I./arch/arm/include/generated -I./include
-I./arch/arm/include/uapi -I./include/uapi -I./include/generated/uapi -include
./include/linux/kconfig.h -D__KERNEL__ -mlittle-endian -Wall -Wundef
-Wstrict-prototypes -Wno-trigraphs -fno-strict-aliasing -fno-common
-Werror-implicit-function-declaration -Wno-format-security -std=gnu89
-fno-dwarf2-cfi-asm -fno-omit-frame-pointer -mapcs -mno-sched-prolog
-fno-ipa-sra -mabi=aapcs-linux -mno-thumb-interwork -mcpu=cortex-a9 -mfpu=vfp -funwind-tables
-marm -D__LINUX_ARM_ARCH__=7 -march=armv7-a -msoft-float -Uarm
-fno-delete-null-pointer-checks -Wno-maybe-uninitialized -Os
--param=allow-store-data-races=0 -Wframe-larger-than=1024 -fstack-protector
-Wno-unused-but-set-variable -fno-omit-frame-pointer -fno-optimize-sibling-calls
-fno-var-tracking-assignments -g -pg -Wdeclaration-after-statement
-Wno-pointer-sign -fno-strict-overflow -fconserve-stack -Werror=implicit-int
-Werror=strict-prototypes -Werror=date-time -Werror=incompatible-pointer-types
-DCC_HAVE_ASM_GOTO -DKBUILD_BASENAME=' "sdhci"' -DKBUILD_MODNAME=' "sdhci"' -c -o
```

The screenshot shows the Eclipse IDE's 'Properties for my-kernel' dialog box, specifically the 'Paths and Symbols' tab. The 'Includes' sub-tab is active, displaying a list of languages with 'GNU C' selected. Below the languages list, two include directories are listed: '/my-kernel/include' and '/my-kernel/arch/arm/include'. The 'Add...' button on the right side of the dialog is highlighted with a red box, indicating the next step in the configuration process. The 'Configuration' dropdown is set to 'Default [Active]'. At the bottom of the dialog, there are buttons for 'Restore Defaults', 'Apply', 'Cancel', and 'OK'.

Set proper include directories.

"Good enough" configuration for working with external modules:

- `include`
- `arch/arm/include`
- **optional:** `arch/arm/mach-<MACH>/include`, `arch/arm/plat-<MACH>/include`

Some functions and macros will be marked as syntax error by indexer.

"Better" configuration:

- Add all directories, verbose build output contains.
- Add `-isystem` dir - it contains compiler specific options (eg. `va_list` definitions).

Properties for linux

type filter text

- ▼ C/C++ General
 - ▶ Code Analysis
 - Documentation
 - File Types
 - Formatter
 - Indexer
 - Language Mapping
 - Paths and Symbols**
 - Preprocessor Includes
 - Profiling Categories
 - Git
 - Linux Tools Path
 - Project References
 - Run/Debug Settings
 - ▶ Task Repository
 - Task Tags
 - ▶ Validation

Paths and Symbols

Configuration: Default [Active] Manage Configurations...

Includes Symbols Libraries Library Paths Source Location

Languages	Symbol	Value
GNU C	# _KERNEL_	
	# _LINUX_ARM_ARCH_	7

Add...
Edit...
Delete
Export

"Preprocessor Include Paths, Macros etc." property page may define additional entries

Show built-in values

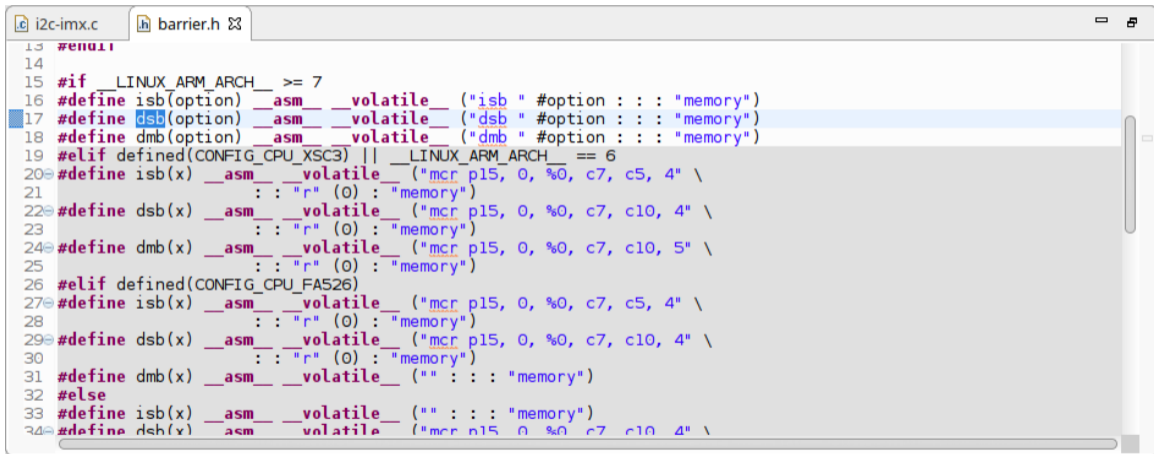
Import Settings... Export Settings...

Restore Defaults Apply

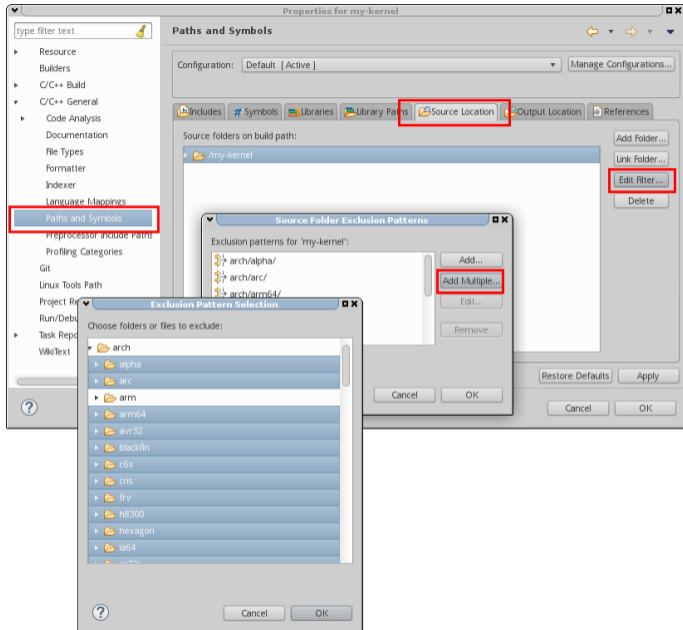
Cancel OK

Set symbols.

Looking at `writel()` or `iowriteX()` call in a driver source code, shows a proper implementation being used:

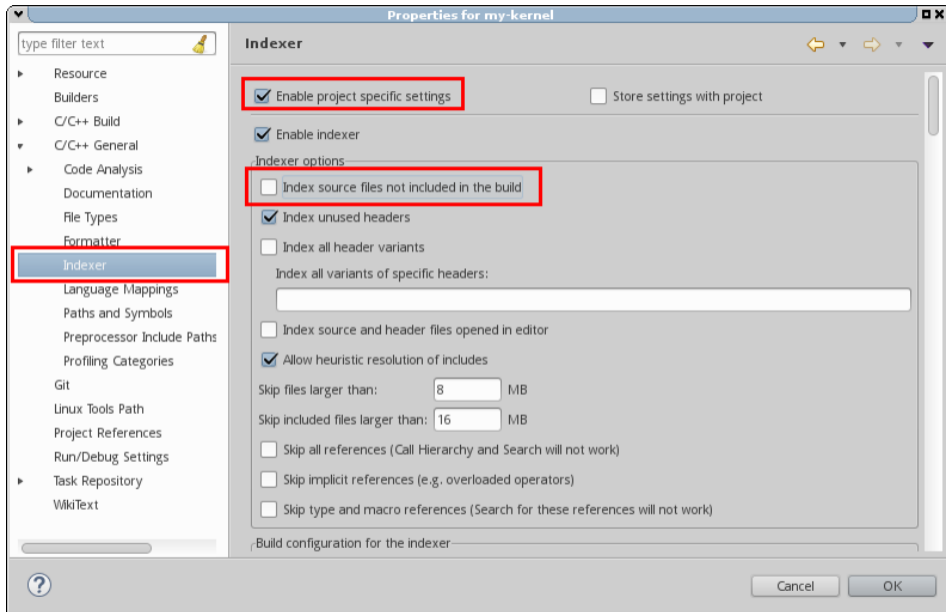


```
i2c-imx.c barrier.h
13 #endif
14
15 #if __LINUX_ARM_ARCH__ >= 7
16 #define isb(option) __asm__ __volatile__ ("isb " #option " : : "memory")
17 #define dsb(option) __asm__ __volatile__ ("dsb " #option " : : "memory")
18 #define dmb(option) __asm__ __volatile__ ("dmb " #option " : : "memory")
19 #elif defined(CONFIG_CPU_XSC3) || __LINUX_ARM_ARCH__ == 6
20 #define isb(x) __asm__ __volatile__ ("mcr p15, 0, %0, c7, c5, 4" \
21 : : "r" (0) : "memory")
22 #define dsb(x) __asm__ __volatile__ ("mcr p15, 0, %0, c7, c10, 4" \
23 : : "r" (0) : "memory")
24 #define dmb(x) __asm__ __volatile__ ("mcr p15, 0, %0, c7, c10, 5" \
25 : : "r" (0) : "memory")
26 #elif defined(CONFIG_CPU_FA526)
27 #define isb(x) __asm__ __volatile__ ("mcr p15, 0, %0, c7, c5, 4" \
28 : : "r" (0) : "memory")
29 #define dsb(x) __asm__ __volatile__ ("mcr p15, 0, %0, c7, c10, 4" \
30 : : "r" (0) : "memory")
31 #define dmb(x) __asm__ __volatile__ (" : : : "memory")
32 #else
33 #define isb(x) __asm__ __volatile__ (" : : : "memory")
34 #define dsb(x) __asm__ __volatile__ ("mcr p15, 0, %0, c7, c10, 4" \
```

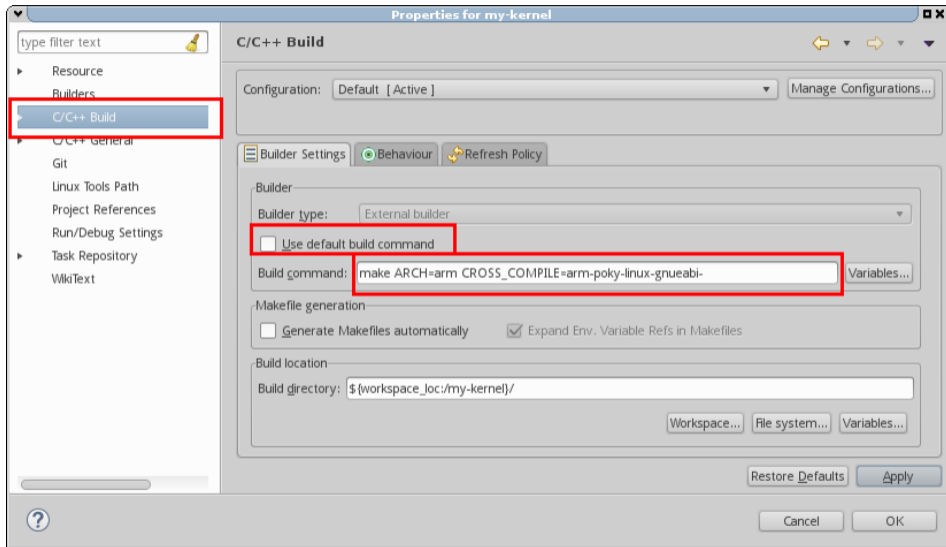


arch directory, contains architecture-specific files. The filter has to be defined to exclude all subdirectories for architectures not used in current configuration.

- select all subdirectories of arch/ but arch/arm
- select tools, scripts, Documentation

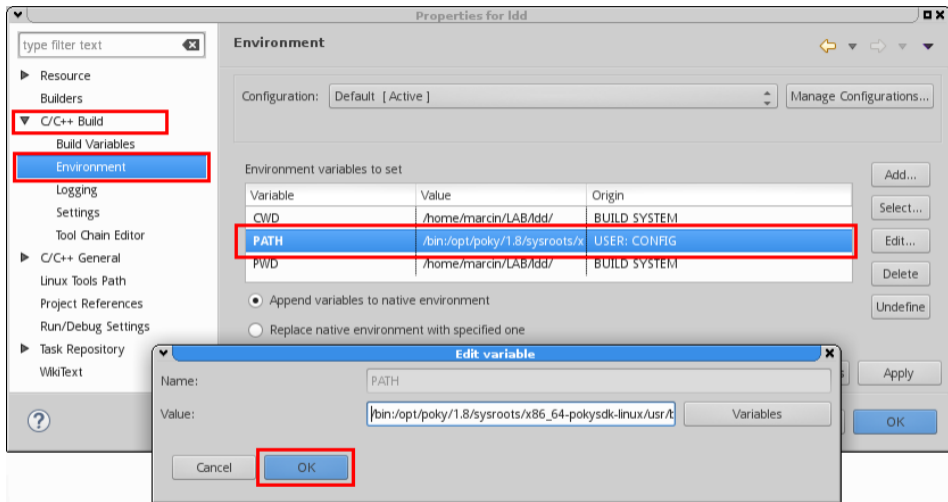


Do not index files not included in build - saves lots of time.



make
parameters to
build the kernel.

Setting
environment
variables may
be needed as
well.



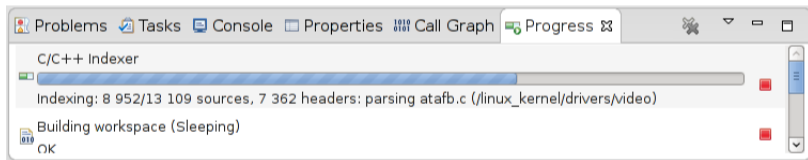
If you executed Eclipse from shell configured to use toolchain - patch is already set.

Saving it in Eclipse as USER: CONFIG will allow us to run IDE from any shell.

Rebuild the index:



This step usually resolves any errors.



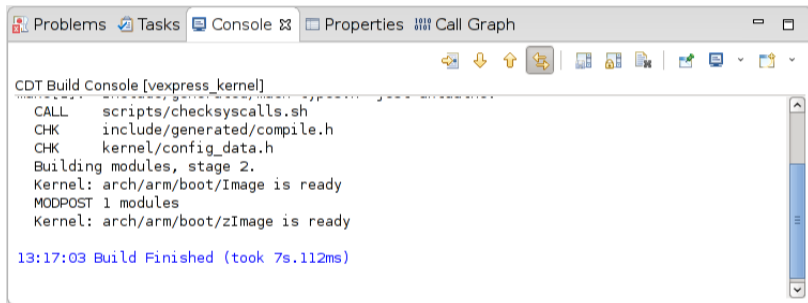
Indexing will take a few minutes. As for Linux 3.14: about 800-900MB of data will be created in workspace directory and about 16500 files will be parsed (exact figures depends on kernel version and configuration).

For 4.8: index takes 1.1GB, 19000 files

Building kernel

Now Eclipse can be used to build kernel.

Messages printed in console:



The screenshot shows the Eclipse IDE's console window for a CDT Build Console. The window title is "CDT Build Console [vexpress_kernel]". The console output shows the following messages:

```
CALL    scripts/checksyscalls.sh
CHK     include/generated/compile.h
CHK     kernel/config_data.h
Building modules, stage 2.
Kernel: arch/arm/boot/Image is ready
MODPOST 1 modules
Kernel: arch/arm/boot/zImage is ready

13:17:03 Build Finished (took 7s.112ms)
```

You may want to create additional `make` targets to make it easier to work with code.
e.g. `dtbs`

Out-of-tree kernel modules

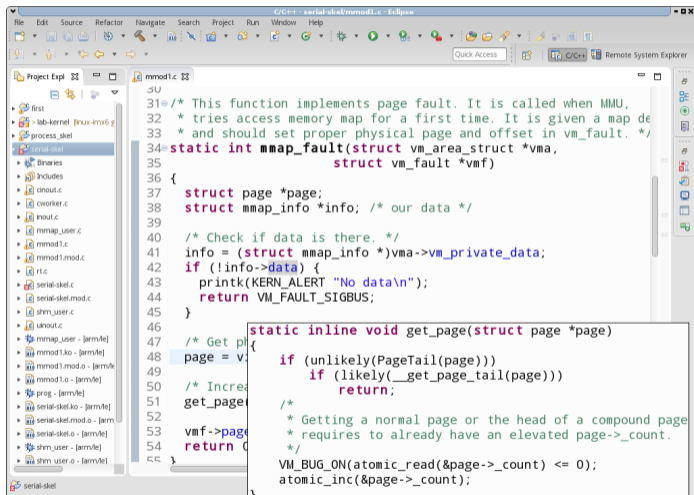
Our kernel module is build out-of-the-tree.

`Makefile` defines targets to build and clean code.

Configuration checklist:

- 1 Create new C project, pointing to existing module sources.
Use the same method as for kernel.
You may also use a shortcut: *Makefile C project using existing code.*
- 2 Indexer has to be informed about kernel configuration - include `autoconf.h` from kernel project.
- 3 Include `printk.h` if needed.
- 4 Turn off searching include paths build-in into toolchain.
- 5 Set proper include path instead (same as in case of kernel: `include, arch/arm/include`).
- 6 Define `__KERNEL__` and `__LINUX_ARM_ARCH__` (=7) symbols.
- 7 Do not define filter - not needed.
- 8 Define additional `make` targets.

The result



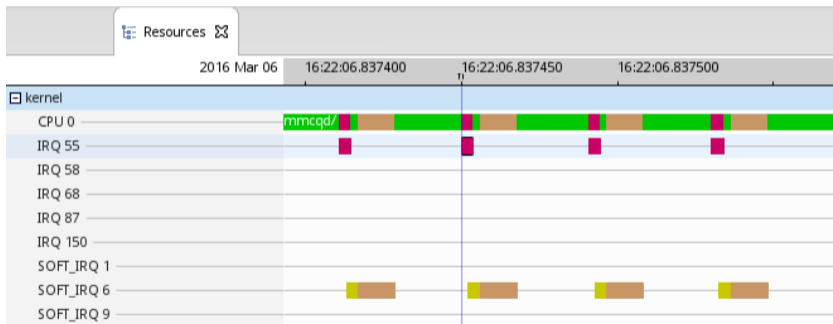
```
30
31 /* This function implements page fault. It is called when MMU,
32 * tries access memory map for a first time. It is given a map de
33 * and should set proper physical page and offset in vm_fault. */
34 static int mmmap_fault(struct vm_area_struct *vma,
35                       struct vm_fault *vmf)
36 {
37     struct page *page;
38     struct mmap_info *info; /* our data */
39
40     /* Check if data is there. */
41     info = (struct mmap_info *)vma->vm_private_data;
42     if (!info->data) {
43         printk(KERN_ALERT "No data\n");
44         return VM_FAULT_SIGBUS;
45     }
46
47     /* Get page */
48     page = v
49
50     /* Increa
51     get_page
52
53     vmf->page
54     return 0
55 }
```

```
static inline void get_page(struct page *page)
{
    if (unlikely(PageTail(page)))
        if (likely(__get_page_tail(page)))
            return;
    /*
     * Getting a normal page or the head of a compound page
     * requires to already have an elevated page->_count.
     */
    VM_BUG_ON(atomic_read(&page->_count) <= 0);
    atomic_inc(&page->_count);
}
```

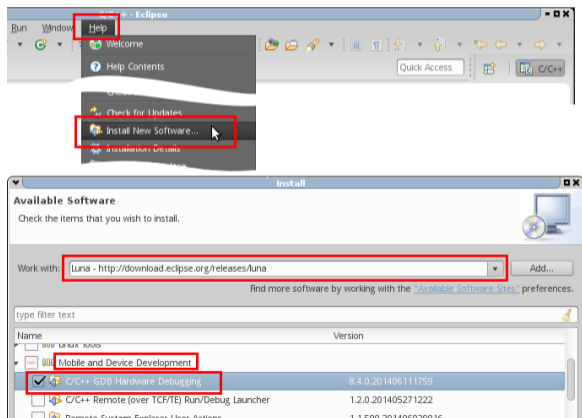
Demo

■ 3.14.56 - drivers/mmc/host/sdhci.h

```
2512 static irqreturn_t sdhci_irq(int irq, void *dev_id)
2561 tasklet_schedule(&host->card_tasklet);
```



Eclipse can be use as a front-end got `gdb`. Debugger loads kernel image from file and connects to system running the same kernel, which `KGDB` enabled. Both systems are connected via serial console. We have to install additional plugin to enable connectivity - C/C++ GDB Hardware Debugging.



Create, manage, and run configurations

Project does not exist



type filter text

- C/C++ Application
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- GDB Hardware Debugging**
 - New_configuration**
- Launch Group

Filter matched 7 of 7 items

Name: New_configuration

Main Debugger Startup Source Common

C/C++ Application:
vmlinux

Variables... Search Project... Browse...

Project:
my-kernel

Browse...

Build (if required) before launching

Build configuration: Use Active

Select configuration using 'C/C++ Application'

Enable auto build Disable auto build








Use workspace settings [Configure Workspace Settings...](#)

Using GDB (DSF) Hardware Debugging Launcher - [Select other...](#)

Apply Revert

Create, manage, and run configurations

 [Main]: Project does not existtype filter text 

-  C/C++ Application
-  C/C++ Attach to Applicati
-  C/C++ Postmortem Debu
-  C/C++ Remote Application
- ▼  GDB Hardware Debuggin
-  New_configuration
-  Launch Group

Name: New_configuration

 Main  Debugger  Startup  Source  Common

GDB Setup

GDB Command:

Browse...

Variables...

Remote Target

 Use remote target

JTAG Device: Generic Serial

GDB Connection String: Force thread list update on suspend

workspace - Debug - linux-rpi/kernel/debug/debug_core.c - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

Debug Σ

linux-rpi Default [GDB Hardware Debugging]

arch_kgdb_breakpoint() at debug_core.c:1,070 0x801b460c

Variables Breakpoints Registers Modules

Name	Type	Value

inout_irq.c inout_user.c Makefile m08_mmap_mod.c mm.h if.h e100.c snull.c sched.h kernel debug_core.c

```

1067=noninline void kgdb_breakpoint(void)
1068 {
1069     atomic_inc(&kgdb_setting_breakpoint);
1070     wmb(); /* Sync point before breakpoint */
1071     arch_kgdb_breakpoint();
1072     wmb(); /* Sync point after breakpoint */
1073     atomic_dec(&kgdb_setting_breakpoint);
1074 }
1075 EXPORT_SYMBOL_GPL(kgdb_breakpoint);
1076
1077=static int __init opt_kgdb_wait(char *str)
1078 {

```

Outline Σ

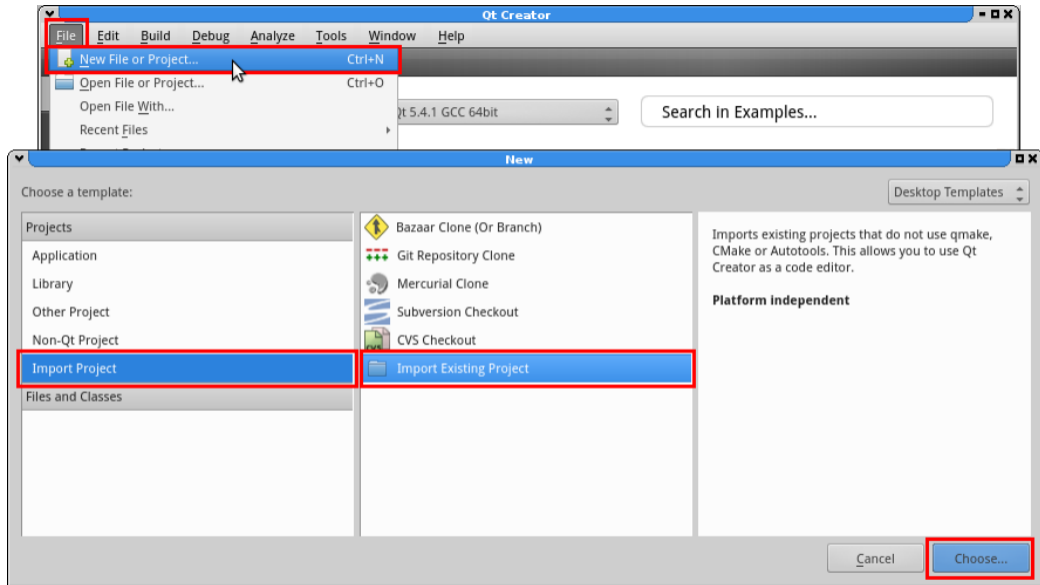
- /home/marcin/LAB/kernel-rpi/linux/include/generated/autoconf.h
- /home/marcin/LAB/kernel-rpi/linux/include/linux/printk.h
- # pr_fmt
- linux/pid_namespace.h
- linux/clocksource.h
- linux/serial_core.h
- linux/interrupt.h
- linux/spinlock.h
- linux/console.h
- linux/threads.h
- linux/uaccess.h
- linux/kernel.h
- linux/module.h
- linux/ptrace.h
- linux/trace.h

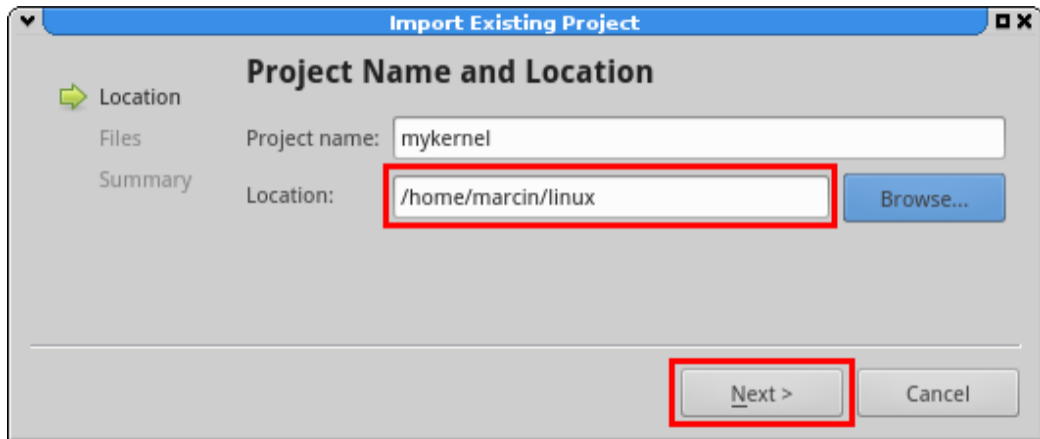
Console Σ Tasks Problems Executables Memory

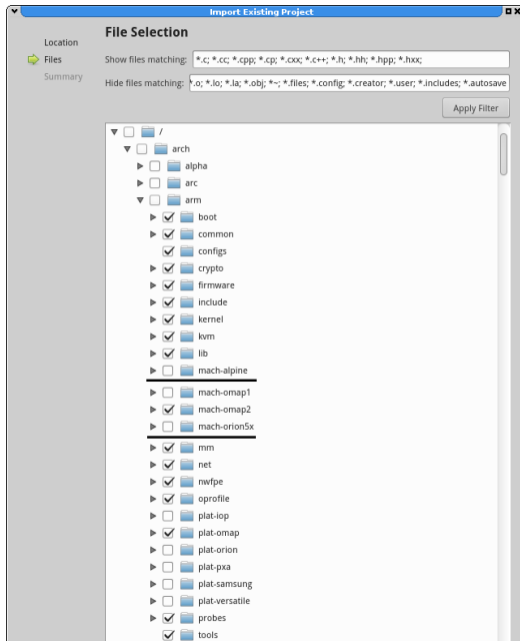
linux-rpi Default [GDB Hardware Debugging] /home/marcin/LAB/toolchain/gcc-linaro-5.3-2016.02-x86_64_arm-linux-gnueabi/bin/arm-linux-gnueabi-gdb (7.10.1.20160210)

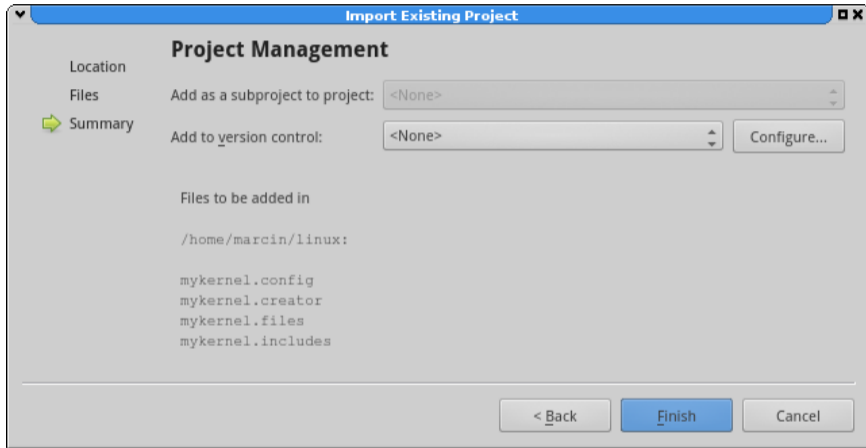
Launching linux-rpi Default: (93%)

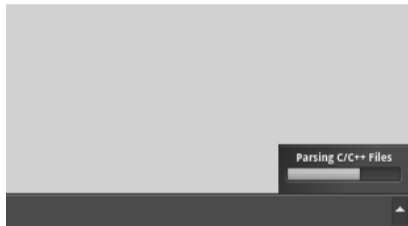
Qt Creator

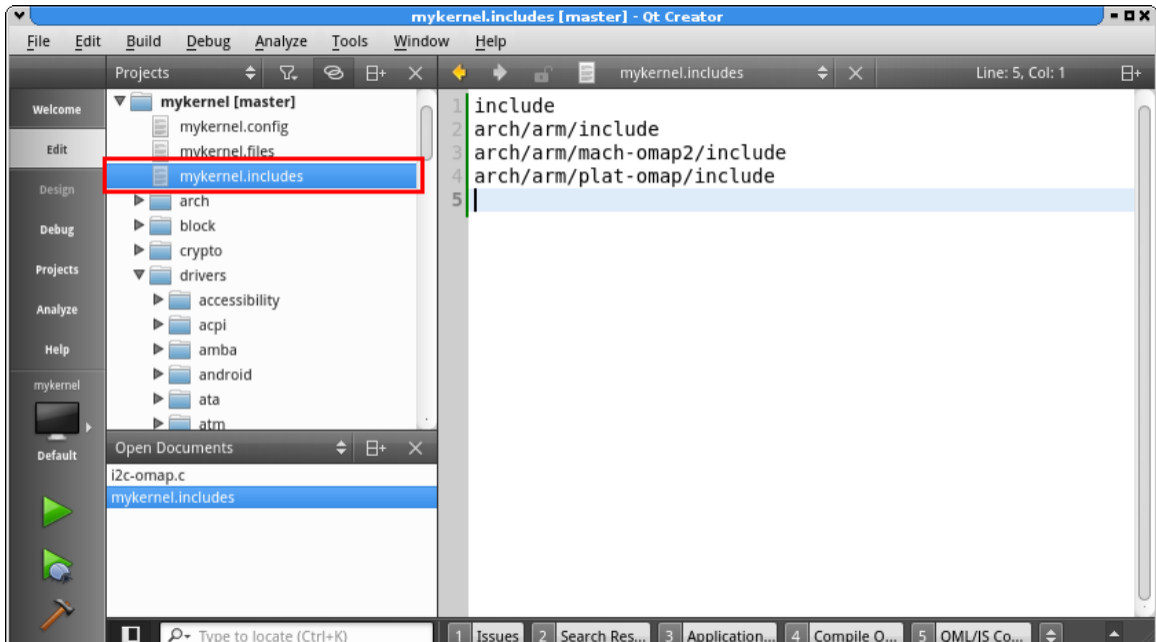


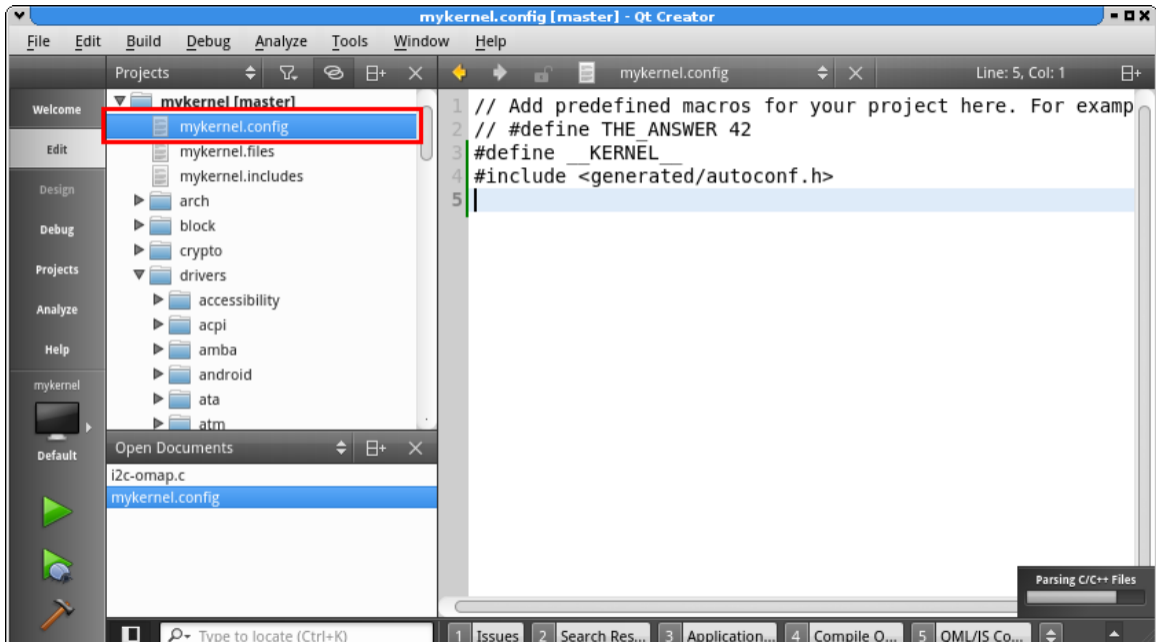


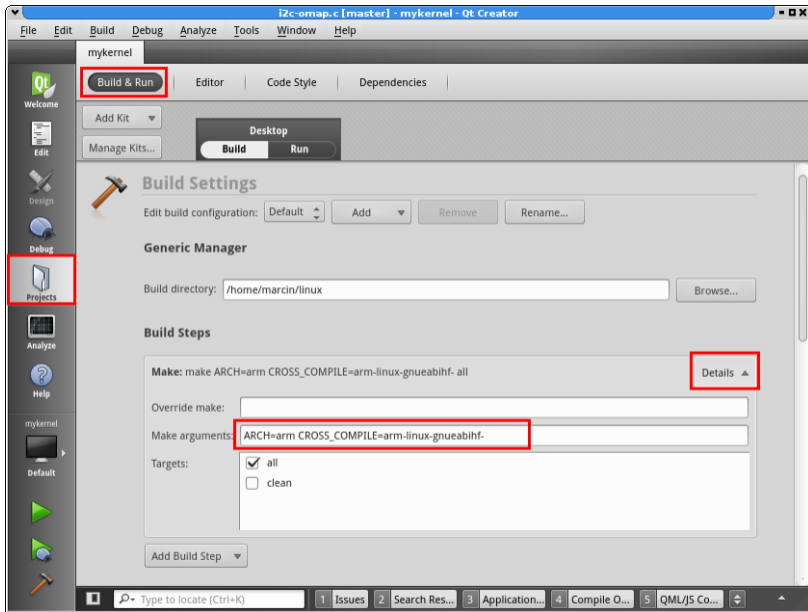












`/* */`

`||`

`?`

Additional files used during the presentation.

`http://bis-linux.com/ELCE/`