Warsaw 03.04.2020

# Linux for Embedded Systems – Laboratory ex. 2

The student: Patryk Walczak

**Description of the assignment**

1. Prepare an application in C language, which uses the buttons and LEDs. When reading the buttons, please take into consideration the „contact bounce” effect.

2. It is required that application implements thorough error checking (you can’t assume that e.g. all hardware communication functions always succeed).

3. The application must respond to changes of state of the buttons without active waiting (should sleep waiting for the required change).
4. The application functionality should be defined by the student.
5. The application should be converted into the Buildroot package.
6. The student should demonstrate debugging of his application with the gdb debugger.

**Procedure to recreate the design from the attached archive**

I prepared:
- build.sh – script which downloads and prepares the configuration
- run_before – script which runs SwitchDemo
- runme – script which runs QEMU and starts buildroot

To reproduce my script:
1. run build.sh script
2. run run_before script
3. run runme script

To add my timer script to buildroot run build_script.sh, then in buildroot type „script”

To reproduce my package please read the description and repeat the steps to make the package.

**Description of the solution**

I prepared the C language program which uses the buttons and LEDs. It is the first option „The timer controlled by the buttons (with lap-time function). LEDs should be used to confirm the operation. The measured time should be displayed on the console.” and as it was required the program handles the „bounce effect”. To understand and find out the gpiodlib functions I used „https://github.com/brgl/libgpiod/blob/master/include/gpiod.h”.

Firstly program connect to the SwitchDemo.

```
chip = gpiod_chip_open("/dev/gpiochip1");
if (!chip)
   return EXIT_FAILURE;
```

Then connect to the buttons with one funtion.

```
if(gpiod_line_request_bulk_falling_edge_events(&buttons, "its_me")){
 printf("Error: line_request_bulk_falling_edge_events -> buttons\n");
 gpiod_chip_close(chip);
 return EXIT_FAILURE;
}
```

But leds are conneced one by one.

```
line1 = gpiod_chip_get_line(chip, 24);
if (!line1)
{
 gpiod_chip_close(chip);
 return EXIT_FAILURE;
}
```

Then open communication with leds.

```
gpiod_line_request_output(line1, "its_me", 0);
gpiod_line_request_output(line2, "its_me", 0);
```

After that program start main loop in which it handles all signals from the SwitchDemo. For „contact bounce” I prepared two funtions - „bulk_gpiod_line_event_without_bounce” and „gpiod_line_event_without_bounce”. The bulk function uses the 2nd recursive function.

*ev_val = gpiod_line_event_wait_bulk(&bulk, timeout, &result);*

In the line, the program waits set time to any signal from the buttons. If any button was clicked, function read what button was clicked and try to deal with bounce effect by the „gpiod_line_event_without_bounce” function. The function recursively checks the signals and return the right signal after omitting the bounce effect.

Then return appropriate event with set event_type set to the proper number. After click button „12” the timer starts and the first led is up. After the second click first button, the timer ends and print the time. The second button prints lab time and rise the second led. The „14” set off all LEDs and stop the timer application.

To add my package to the buildroot I create a directory “package” and put it in “BR_PATH/package” inside the directory I create a “Config.in” file which will serve as a description for Buildroot and “timer.mk” which will tell Buildroot how to make the package. Next, I add a line to “Config.in” file under “Target packages” in “package” directory: source “package/timer/Config.in”

I use CMake for building. I put sources outside the Buildroot directory (Also set the correct path to the sources in the “timer.mk” file). My sources are in the “timer_src” directory. Inside the directory, I put a “CMakeLists.txt” file and an “src” directory. Inside the directory, I create another “CMakeLists.txt” file and “script.c” file. All as described in the lecture slides.

Now I modify buildroot settings with $make menuconfig.
Target packages → Debugging,... → gdb → gdbserver
Toolchain → Build cross gdb for the host
Build options → build packages with debugging symbols
Target packages → timer

The instructions add timer as a package to the buildroot and enable the gdb debugger. In case the image was already built with my package and I made a change to the source files I have to do:

```
$ make timer-dirclean
```