gpioirq

Version 1.0.0 – 16 June 2016
Kit Bishop

Document History		
Version	Date	Change Details
Version 1.0.0	16 June 2016	First released version

Contents

1. Overview

gpioirq is C++ program that will run user supplied commands on interrupts from Omega GPIO pins.

Notes:

- The **gpioirq** library uses of the **libnewgpio** library that is available and documented at https://github.com/KitBishop/Omega-GPIO-I2C-Arduino/tree/master/libnewgpio

gpioirq consists of single program in static and dynamic forms that runs the commands on the interrupts.

This program is described in more detail in this document.

The program was developed on a KUbuntu-14.04 system running in a VirtualBox VM and uses the OpenWrt toolchain for building the code:

The toolchain used can be found at:

- https://s3-us-west-2.amazonaws.com/onion-cdn/community/openwrt/OpenWrt-Toolchain-ar71xx-generic_gcc-4.8-linaro_uClibc-0.9.33.2.Linux-x86_64.tar.bz2

and details of its setup and usage can be found at:

- https://community.onion.io/topic/9/how-to-install-gcc/22

gpioirq comes with **NO GUARANTEES** ☺ but you are free to use it and do what you want with it.

2. Files Supplied

gpioirq is supplied in files in a GitHub repository at https://github.com/KitBishop/Omega-GPIO-I2C-Arduino/tree/master/gpioirq. This repository contains the following important directories and files:

- **gpioirq.pdf** – this documentation as a PDF file
- **Makefile** – the Makefile for **gpioirq** library
- **hdr** – directory containing header (***.h**) files for **gpioirq** library
- **src** – directory containing source (***.cpp**) files for **gpioirq** library
- **bin** – directory containing the built program code:
    - **dynamic/gpioirq** – the dynamically linked version of the program
    - **static/gpioirq** – the statically linked version of the program

3. Usage and Installation

Installing and using the program is simple. It primarily consists of linking the program and if needed (see below) the gpio library code.

3.1. Using gpioirq statically linked program

To use **gpioirq** statically linked program you simply need to copy the program to the Omega and run it.

3.2. Using and Installing gpioirq dynamically linked program

To use **gpioirq** dynamically linked program you need to copy it and the **libnewgpio** library to your Omega and then run the program.

Since **gpioirq** dynamically linked program makes use of **libnewgpio** library for the I2C communication, you will also need to dynamically link to copy **libnewgpio.so** to the **/lib** directory on your Omega

Alternatively, you can copy the library to any location that may be set up in any **LD_LIBRARY_PATH** directory on your Omega. For example, I use the following for testing:

- Created directory **/root/lib**
- Copied the libraries to **/root/lib**
- Added the following lines to my **/etc/profile** file:

```
LD_LIBRARY_PATH=/root/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
```

4. Using Makefile

A **Makefile** is supplied that can be used to build the library.

4.1. Modify Makefile

The **Makefile** will need modifying:

- You **<u>NEED</u>** to and **<u>MUST</u>** change **TOOL_BIN_DIR** to the "bin" directory of your OpenWrt uClibc toolchain. E.G. make appropriate change to **<xxxx>** in:

  TOOL_BIN_DIR=<xxxx>/OpenWrt-Toolchain-ar71xx-generic_gcc-4.8-linaro_uClibc-0.9.33.2.Linux-x86_64/toolchain-mips_34kc_gcc-4.8-linaro_uClibc-0.9.33.2/bin

- You **<u>MAY</u>** need to change **LIBNEWGPIO_DIR** to relative directory of libnewgpio if you are not using the sources as originally supplied.
  The default if using the standard **source** directory structure as supplied is:

  LIBNEW-GPIO_DIR=../libnewgpio

4.2. Makefile targets

The **Makefile** implements the following set of targets:

- **make**
  The default target. Performs a complete build of both static and dynamic link versions of the program.
  This is directly equivalent to:
  make static dynamic

- **make static**

Performs a complete build of just the static link version of the program.

- **make dynamic**
  Performs a complete build of just the dynamic link version of the program.

- **make clean**
  Removes all previous build files, both static and dynamic link versions.
  This is directly equivalent to:
  make clean-static clean-dynamic

- **make clean-static**
  Removes all previous build files for static link versions only
  .

- **make clean-dynamic**
  Removes all previous build files for dynamic link versions only

If the following is added to the **make** command line:

builddep=1

then **libnewgpio** library that the program depends on will also be built before building the program.

5. Running the gpioirq Program

The gpioirq program can be run from the directory where the program is placed:

```
./gpioirq <parameters>
```

The program parameters are documented by running the command:

```
./gpioirq help
```

Which gives the self-explanatory output:

```
Catches interrupts on given pin to run given command(s)
Usage
Commands:
- one of:
    ./gpioirq <pin> rising <command> <optional-debounce>
        Starts background process to run <command> on rising edge of pin
    ./gpioirq <pin> falling <command> <optional-debounce>
        Starts background process to run <command> on falling edge of pin
    ./gpioirq <pin> both <command1> <command2> <optional-debounce>
        Starts background process to run <command1> on rising edge of pin
                                     and <command2> on falling edge of pin
    ./gpioirq <pin> stop
        Terminates input handling on pin
    ./gpioirq help
        Displays this usage information

Where:
    <pin> is a valid GPIO pin number to trigger the interrupts
    <command>, <command1>, <command2> are the commands to be executed
```

```
        on the interrupt.  Must be enclosed in " characters if they contain
        spaces or other special characters
    <optional-debounce> is an optional parameter that specifies a debounce time
        in milliseconds to allow for potentially noisy mechanical switches.
        Any signal change that occurs within this time of the previous change
        will be ignored.  If absent or 0, no debounce processing will be applied.
```

When the program is run using one of the forms:

```
./gpioirq <pin> rising …
```

Or:

```
./gpioirq <pin> falling …
```

Or:

```
./gpioirq <pin> both …
```

A separate process is started in the background to keep interrupt processing active. This process continues until it is terminated using:

```
./gpioirq <pin> stop
```

With the same **<pin>** number.

6. Further Development

Development of **gpioirq** is on-going. There will be changes and additions to the code in the future.