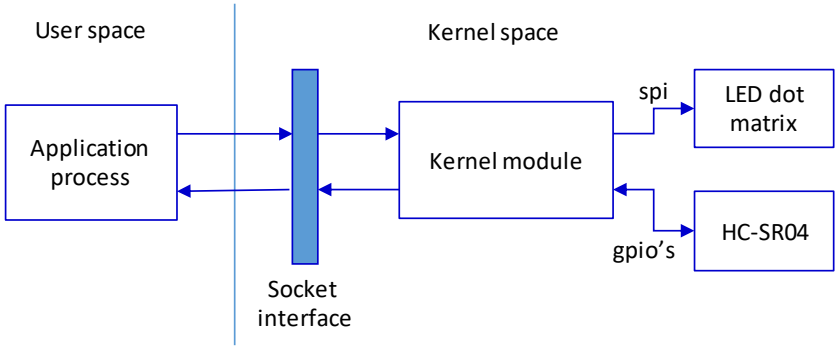
Assignment 3 Generic Netlink Socket and SPI Device Programming (150 points)

Exercise Objectives

1. To learn the basic programming technique in Linux spi and gpio kernel modules.
2. To learn the application of kernel thread and interrupt handler to manage spi and gpio operations
3. To learn socket programming using generic netlink in an application of distance measurement and animation display.

Programming Assignment

Linux socket enables a communication interface to various protocol channels between network nodes and between processes. In this assignment, you will exercise the mechanism provided by netlink socket for communication between user-space processes and the kernel. As shown in the following diagram, via receiving and sending messages, an application process interacts with a kernel module which controls a HC-SR04 sensor and a LED dot matrix to enable display animation based on measured distance.



In the assignment, you will develop an application program and a kernel module, and use a generic netlink socket to enable the communication between them. There are three types of message to be sent by the application process to the kernel module:

1. A display pattern for the LED dot matrix,
2. A configuration command for SPI CS pin, HC-SR04 trigger and echo pins,
3. A request to measure distance (which is the average of 5 samples collected from HC-SR04).

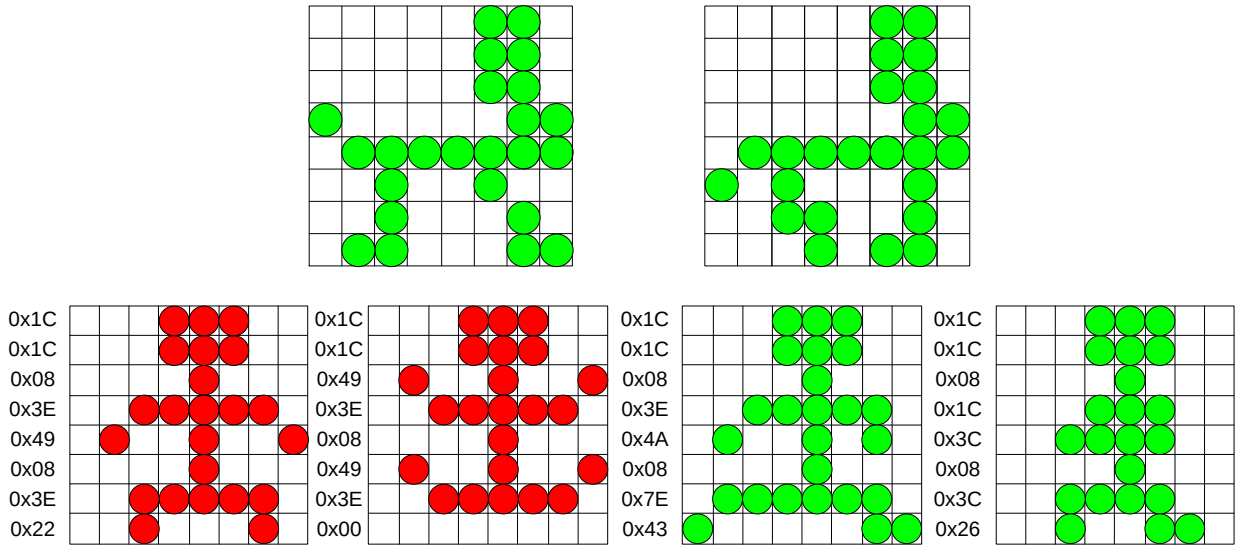
The application process can receive a message from the kernel module that contains a measured distance. Based on the measured distance, it should determine any updates of display pattern or request a new measurement after a fixed period. You should design a proper message format for the communication. Also, all IO operations must be down asynchronously with the communication operation.

To display patterns, you will use a simple 8X8 LED matrix which is equipped with a MAX7219 driver. In addition to source current for LED display, the driver chip can buffer display data and scan digits using an internal oscillator.

To display a pattern, 8 bytes of pattern should be written to the driver chip via SPI interface bus. Assume we will use the example definition of a pattern:

```
typedef struct
{
        uint8 led[8];     // 8 bytes (64 bits) for 64  leds
} PATTERN;
```

To create an animation, a sequence of patterns should be written to the LED matrix. For instance, by switching between the following two patterns, the LED matrix can show a running or walking dog. (This is an example display pattern. Please create something interesting, such as bouncing balls, for this assignment. For example, in the next diagram, you can show the red guy standing and jumping and the green one walking or running)



To make your application interesting, you should use the measured distance to determine how fast your display object is moving (from walking to running) and the direction it moves (left or right when the distance object moves to the sensor or away from the sensor).

Due Date

The due date is 11:59pm, April 17.

What to Turn in for Grading

- Create a working directory, named "EOSI-LastName-FirstInitial-assgn03", to include your source files (.c and .h), makefile(s), and readme.
- Compress the directory into a zip archive file named EOSI-LastName-FirstInitial-assgn03.zip and submit the zip archive to Canvas by the due date. Note that any object code or temporary build files should not be included in the submission.

- Please make sure that you comment the source files properly and the readme file includes a description about how to make and use your software. A sample result from your test run can be included in readme file. Don't forget to add your name and ASU id in the readme file.
- There will be 20 points penalty per day if the submission is late. Note that submissions are time stamped by Canvas. If you have multiple submissions, only the newest one will be graded. If needed, you can send an email to the instructor and TA to drop a submission.
- The assignment must be done individually. No collaboration is allowed, except the open discussion in the forum on Canvas. The instructor reserves the right to ask any student to explain the work and adjust the grade accordingly.
- Failure to follow these instructions may cause deduction of points.
- Here are few general rule for deductions:
    - No make file or compilation error -- 0 point for the part of the assignment.
    - Must have "–Wall" flag for compilation -- 5-point deduction for each warning.
    - 10-point deduction if no compilation or execution instruction in README file.
    - Source programs are not commented properly -- 10-20-point deduction.
- ASU Academic Integrity Policy (http://provost.asu.edu/academicintegrity), and FSE Honor Code (http://engineering.asu.edu/integrity) are strictly enforced and followed..