18-847: Wireless Software Systems Architecture

Lab 3:
Enabling Interrupts

Introduction

In the previous two labs, we learned about FreeRTOS and threaded programming and serial connections between embedded devices. Next, we will update the code developed in Lab 2 to enable interrupts on the PowerDue. Using the interrupts, we will develop a magnetic detector, capture the magnetic signals and do some simple processing on the signal to calculate the number of times a magnetic material is brought near the sensor.

The objectives of this lab are:

- Understanding the working of interrupts
- Enabling low power modes on PowerDue
- Learning how to read sensor datasheets
- Learning how to build and use device libraries
- Comparing the energy consumption of polling v/s interrupts

Part 1: Enabling interrupts

In this part, we will interface the PowerDue to the FXOS8700CQ magnetometer using the SPI protocol, which is part of the BRKT-STBC-AGM01 breakout board. We will build upon the code you used in Lab 2 and modify it to work in FreeRTOS.

Resources:

For this part of the lab, you will need:

1. The PowerDue connected to the BRKT-STBC-AGM01 breakout board on the left bench.
2. FXOS8700CQ datasheet
3. Lab 2 FXOS8700CQ driver and SPI driver code

Setting up the environment:

We will write all our code on the Arduino IDE. As you will be using your code developed in Lab 2, ensure that any statements or modifications you might have done for doing the energy calculations are reset to default behaviour.

Adding interrupt enable and disable functions:

Using the datasheet, write functions to enable, disable interrupts and read the magnetometer interrupt source register for the FXOS8700CQ. We will be using the INT1 pin of the FXOS8700CQ for generating interrupts

Calibrating the magnetometer:

While polling the magnetometer, observe the x,y and z axis values. Are they constantly changing? What is the reason for the constant variations in the values? The magnetometer can be set to trigger an interrupt if a signal above a threshold level is encountered. How do you set this level? If the threshold was zero, or less than the values that are observed without any magnetic material nearby, an interrupt would always be triggered. How do you ensure that the variable signal will itself will not trigger an interrupt? How do you adjust the default offset of the axis values? This process is known as calibration. Write a function to calibrate the magnetometer signal and set a useful interrupt threshold.

Enabling interrupts on the PowerDue:

On the PowerDue, set up the EXT_SPI_SS0 pin (Pin number 4) as your SPI chip select pin and GPIO pin (Pin 51) as the interrupt pin. You can look at the Arduino Reference to figure out how to attach an external interrupt to the Arduino. Refer to the datasheet to check the properties of the interrupt source from the FXOS8700CQ. In the interrupt service routine, you will be using the FreeRTOS ISR queues. Keep the ISR as short as possible. The ISR should call another task *colllectData()* to collect samples from the magnetometer. For how long would you collect data from the magnetometer? Define the capabilities of your system.

Processing the data:

The data captured by the *collectData()* now needs to be processed to determine the number of times. The number of times a magnetic material was brought close to the magnetometer will simply be the number of times *collectData()* is called. Create another task called *processData()* which will just increase the count by 1 everytime *collectData()* is called.