Lab 3(part 2): Reading of Sensors

Preparation

Reading

Lab Manual

Chapter 4 - The Silicon Labs C8051F020 and the EVB (sections on the *C8051 timer functions, interrupts, and the programmable counter array*)

Chapter 5 - Circuitry Basics and Components (The buffer)

Chapter 6 - Motor Control (DC Motors)

Objectives

General

Implement the sensor hardware and write code that can read the sensors. The ranger pair and the compass pair will write the code to read respective sensors.

Hardware

1. The respective teams will install ultrasonic ranger and electronic compass for data collection.

Software

2. Ultrasonic Ranger Pair - (1) request that the ranger starts an ultrasonic ping, (2) wait 65ms while the ranger "listens" for an echo, (3)request that the ranger value be sent on the I^2C bus.
3. Electronic compass pair - (1) obtain a reading and (2) control how often to get a reading.

Motivation

In the previous lab, pulse width modulation was introduced to change speed of drive-motor and steering of servo-motor. In this lab, a program reads the electronic compass or the ultrasonic ranger, which will be used respectively to control heading or altitude in Lab 6.

Lab Description & Activities - Ultrasonic Ranger Pair

Read the specification on ultrasonic ranger and wire the ranger according to the given circuit diagram. Note that the ranger is to be mounted on the small protoboard provided on the *Smart Car.* Don't move the rangers since those are also used in 3 other sections. The following pseudo-code describes the main function to read from the ranger

```
Initialize everything
start while (1) loop
if 80ms has passed
call ranger function
print range - (printing every 80ms may be more than needed)
```

1. Write a ranger function which reads the ultrasonic ranger.

2. Check the ranger for accuracy.
   – Hold an object above the car, measure the height with the ranger and with a tape measure.
   – Repeat for several distances
   – Repeat with a piece of carpet
   – What is the closest an object can be and you still get reliable readings?
3. Check the ranger for sensitivity to objects off to the side. Crudely estimate the angular sensitivity of the ranger.Record the information of ranger sensitivity in your lab notebook and have a TA verify the performance of the ranger.

Lab Description & Activities - Electronic Compass Pair

Read the specification on electronic compass and wire the compass according to the given circuit diagram. Note that the compass is to be mounted on the small protoboard provided on the *Smart Car* such that the car is pointing in the direction of magnetic north the compass read 0 degree. Don't move the compasses since those are also used in 3 other section.

The compass will return a result that is accurate to a tenth of a degree. This result is return as an integer, therefore you will need to perform any calculations based on the true reading. For example, if the compass was aligned with 48.3 degrees off north, the result the code will return is 483.

The following pseudo-code describes the main function to read from the ranger

```
Initialize everything
start while (1) loop
if 40ms has passed,
call heading function
print heading
```

1. Write a compass function which reads the electric compass.
2. Check the heading for accuracy.
   – The magnetic directions will indicated in an area of the room with low interference
   – Place the car on the floor near these markings and make reading with the car heading North, East, South, and West
   – Note the orientation of the module. Its reference isn't in line with the car, rather at right angles. So when the car is aligned North, the sensor should read either about 90 or 270 degrees.
   – Note that the room has a fair amount of magnetic interference. The direction of 'North' will depend on where you are located, especially if you are near an active power conduit.
3. If your readings are within 10 degrees of the lines, then show the results to a TA
4. If the readings vary by more than 10 degrees, then read the sensor should be calibrated.
5. Record the results in you lab notebook and have a TA verify the results, even if you don't get any better accuracy than what you started with.
6. Record these readings in your lab notebook

Software - Ultrasonic Ranger Pair

The pseudo-code of the ranger function, which reads the ultrasonic ranger, is:

```
while (1)
{if 80ms have passed
{
read the ranger
reset the 80ms flag
start a ping
print the range
}
}
```

Notes:

1. The time lag of 80 ms is used because we have already created a counter with overflows every 20 ms using the PCA timer overflow interrupts; 4 PCA timer overflows yield 80 ms.
2. The first reading will be wrong because we haven't requested a ping, but after 80ms we will have valid readings. The blimp doesn't travel very far in 80ms, the car travels even less.
3. To read data:

```
unsigned int ReadRanger()
{
       unsigned char Data[2];
       unsigned int range =0;
       unsigned char addr=0xE0// the address of the ranger is 0xE0
       i2c_read_data(addr, __, Data, _);  / / read two bytes, starting at reg 2
       range = (((unsigned int)Data[0] << 8) | Data[1]);
       return range;
}
```

4. To start a ping with the result in cm:

```
Data[0] = 0x51;                          //write 0x51 to reg 0 of the ranger:
i2c_write_data(addr, ___,  Data, _) ;  // write one byte of data to reg 0 at addr
```

5. To create a flag that indicates that 80ms have passed, put the following into the PCA ISR:

```
void PCA_ISR ( void ) interrupt 9
{
   if (CF) {
       ...
       r_count++;
       if(r_count>=4)
       {
          new_range=1;  // 4 overflows is about 80 ms
          r_count = 0;
       }

   }
}
```

6. The ‘new_range’ is the flag used to determine if a new range is ready to be read:

```
       if (new_range) { clear new_range, read range, start a new ping}
```

Software - Electric Compass Pair

To read the compass you simply need to read registers 2 and 3 of the compass module. Both this sensor and the ranger are set so that if you read register 2 and then request another read, you will get register 3. So, The pseudo-code of the compass function, which reads the electric compass, is:

```
unsigned int ReadCompass()
{
unsigned char addr = _____;// the address of the sensor, 0xC0 for the compass
unsigned char Data[2]; // Data is an array with a length of 2
unsigned int heading;// the heading returned in degrees between 0 and 3599.
i2c_read_data(addr, ____, Data, _);  / / read two byte, starting at reg 2
heading =(((unsigned int)Data[0] << 8) | Data[1]);  //combine the two values
//heading has units of 1/10 of a degree
return heading; // the heading returned in degrees between 0 and 3599.
}
```

Notes:

1. The electronic compass is easy to use because it is always updating itself. Any ‘read’ commands will result in a reply of the last valid reading. It is however a tricky thing to use inside the LITEC lab because there are stray magnetic fields. The central column in the studio appears to be magnetized, and looks like the North Pole when you get close. There is a lot of steel reinforcing in the columns and in the floor. So while it is easy to take magnetic readings, it isn't easy to really know which direction you are headed. This problem doesn't exist in the Armory. The object then is to make the compass work as best as you can in this room, with the expectation that things will get easier later.

2. You need to consider how often and when to read the compass. The compass takes about 35 ms to complete a reading. Our controller will need to look at the time rate of change of the heading, (the turning rate of the blimp.) We will determine the time rate of change by taking the difference of two compass readings and effectively dividing by the time difference:

   ```
   rate of change = (heading change)/(time change).
   ```

   For the above reason, we don't want to take two heading readings within 35ms. If both readings occur within one actual sensor update, both readings will be the same and our code will assume the blimp isn't turning, while it may well be turning very rapidly. The solution suggested here is to wait 40ms between request to the compass module for the present heading. 40ms is chosen because it is equal to time of 2 PCA timer overflows:

   ```
   void PCA_ISR ( void ) interrupt 9
   {
   if (CF) {
       ...
       h_count++;
       if(h_count>=2)
       {
           new_heading=1;  // 2 overflows is about 40 ms
           h_count = 0;
       }
    ...
   }
   }
   ```

In the main program, you will get a compass reading only if new_heading is set high in the PCA interrupt function above. The 'new_heading' is a flag that is set in the PCA ISR and cleared when you do a compass reading.

```
if (new_heading) // enough overflows for a new heading
{
heading = read_compass();
printf(...)  // print heading
/* Printing every compass reading will slow down the code significantly.
   Consider printing every 5th reading. */
new_heading = 0;
}
```

Lab Check-Off: Demonstration and Verification

1. Complete the required entries in your lab notebook and present it to your TA.
2. Your TA may ask you to explain how sections of the C code or circuitry you developed for this exercise work. To do this, you will need to understand the entire system.
3. Ultrasonic Ranger Pair - How would you change the commands to receive the results in inches or msec? How would you read the range results out of the ranger?
4. Electronic Compass Pair - How would you change the commands to receive the results as a number between 0 and 255?

Writing Assignment - Lab Notebook

Your Lab Notebook must be kept up to date by recording the work you and your partner do in the lab. This should include pseudo-code for your system, data graphs, and a copy of your final C code for this lab. Further information on keeping a good Lab Notebook can be found in *Writing Assignment Guidlines* section of the manual.