Project 5: Sensor Models

Questions?

Project 4

- Turn in using subversion
- Demos by Friday

Math on the Microcontrollers

- Inexpensive microcontrollers:
  - Do have hardware for integer math operations
  - **Do not** have hardware for floating point math. Instead, floating point operations are performed by functions (this is hidden from you)
- When time is important (in particular, for a control loop), we strive to do as much with integers as possible. But, one must be cautious…

Integers

- Division: lose the remainder
- Multiplication, addition, subtraction: must make sure that we do not end up with a value that does not fit in the size of integer that we are using

- We must worry about both for the intermediate values of a computation as well as the final values.

Division

```
int8_t a = 5;
int8_t b = 7;
int8_t c = a/b;
```

What is c?

Division

```
int8_t a = 5;
int8_t b = 7;
int8_t c = a/b;
```

c = 0

Addition

```
int8_t a = 125;
int8_t b = 115;
int8_t c = a + b;
```

What is c?

Addition

```
int8_t a = 125;
int8_t b = 115;
int8_t c = a + b;
```

c = -16

Mixing Operations

```
int8_t a = 125;
int8_t b = 115;
int8_t c = (a + b)/4;
```

What is c?

Mixing Operations

```
int8_t a = 125;
int8_t b = 115;
int8_t c = (a + b)/4;
```

c = -4

The intermediate value matters!

Why Not Pick the Largest Integer?

We could solve the overflow problem (in part) by picking the biggest available integer (int32_t)…

- Our microcontrollers can only work with 8 bits at a time
- To add two int32_t variables together requires 8 separate steps to pull these variables from memory in order to add them (and another 4 steps to write to memory)

- The lesson: always pick the size that is just right

Performing Floating Point Operations

```
int16_t a = 150;
```

How do we multiply a by 0.75?

Performing Floating Point Operations

```
int16_t a = 150;
int16_t b = (a * 75) / 100;
```

Performing Floating Point Operations

```
int16_t a = 150;
int16_t b = (a * 75) / 100;
```

But remember: a*75 must fit within an int16_t!

Fixed-Point Mathematics

- Instead of using an integer variable to represent units of “1”, we use the integer to represent units of “10ths” or “100ths” (or smaller)
- So, we can write:

```
int16_t a = 5;
```

to mean that a is capturing a value of 0.05

Addition Works

```
int16_t a = 5;          // 0.05
int16_t b = 130;        // 1.3
int16_t c = a + b;      // = 1.35
```

Multiplication

```
int16_t a = 5;          // 0.05
int16_t b = 130;        // 1.3
int16_t c = a * b;
```

What is c?

Multiplication

```
int16_t a = 5;          // 0.05
int16_t b = 130;        // 1.3
int16_t c = a * b;      // 6.50 ??
```

Multiplication

```
int16_t a = 5;                  // 0.05
int16_t b = 130;                // 1.3
int16_t c = a * b / 100;        // 0.06
```

Must take into account the extra factor of 100

Programming Embedded Systems

The book has a slightly different take on this:

- Instead of a variable representing a value in units of 10ths or 100ths, the variable represents the value in units of 2^{-k}
- So, in a “16.8” representation (where k=8):
  - A value of 1 means 1/256
  - A value of 2 means 2/256
  - A value of 512 means 2

Programming Embedded Systems

Come with questions on Thursday …

Project 5: Sensor Models

- Derive a sensor model given the data that you have collected
- Code: create a function that returns a calibrated distance
- Collect data and analyze

Component 1: Sensor Model

- Given the data you have collected, design a function that will return a distance in mm given the raw analog value that you read
- Approximate with a simple function
  - We won’t be able to capture all points perfectly
  - Capture the “most important” ones best & the others less well
    - In our case, the most important ones correspond to the distances at 5cm and slightly above

Component 2: Code

Functions to implement:

- `uint16_t read_distance(Sensor side)`
  - Read from one of the two sensors & return a calibrated distance value
  - Must use integer math
- `main()`
  - Repeatedly sample the two sensors & print out the distances.

Component 3:
Data Collection and Analysis

- At least 5 samples each for: 5, 6, 8, 10, 14, 20, 30, 40, 60, 80 cm.
- Plot:
  - Sensed distance as a function of distance (mm)
  - One set of points for each of two sensors

Next Time

Serial communication