

# Project 5: Sensor Models

# Questions?

# Project 4

- Finish demos
- Catme surveys are due on Monday
  - Everyone should have received email already

# Integer 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 = 150;  
int8_t b = 90;  
int8_t c = a + b;
```

What is c?

# Addition

```
int8_t a = 150;  
int8_t b = 90;  
int8_t c = a + b;
```

**c = -16**

# Mixing Operations

```
int8_t a = 150;  
int8_t b = 90;  
int8_t c = (a + b) / 4;
```

What is c?

# Mixing Operations

```
int8_t a = 150;  
int8_t b = 90;  
int8_t c = (a + b) / 4;
```

**c = -4**

The intermediate value matters!

# 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;
```

# Fixed-Point Mathematics

- Instead of using an integer variable to represent units of “1”, 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

# Project 5: Sensor Models

- Derive a sensor model given the data that you have collected
- Code: 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

# Component 2: Code

## Functions to implement:

- `uint16_t get_distance(DistanceSensor side)`
  - Read from one or two sensors & return a calibrated distance value
  - Must use integer math
- `void display_distance(uint16_t dist)`
  - Update the LED bar
- `main()`
  - Repeatedly sample the two sensors & report 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 (cm)
  - One set of points for each of two sensors

# New Hardware

To complete the project, you will need one more Sharp sensor. We have some now – others will be available in the next day.

# Next Time

Serial communication