Groups

- Group 1: Aleshinloye, Harrison, Lan, Van Duyne(*)
- Group 2: Terrill, Russell, Mills, Lee
- Group 3: Langston, Aguayo, Greenway, Knapp(*)
- Group 4: Robinson, Tankersley, Lanham, Couch
- Group 5: Verdin, Pawlowski, Rosa

* = new platform

Today: Project 2

Group work for today:

- First circuit
- Essential software pieces
  - Reading compass heading
  - Computing compass error
  - Computing compass derivative
  - Displaying heading or error with 4 LEDs
  - Displaying rotational velocity with 5 LEDs

As you complete one or two tasks, show them to Di or Me

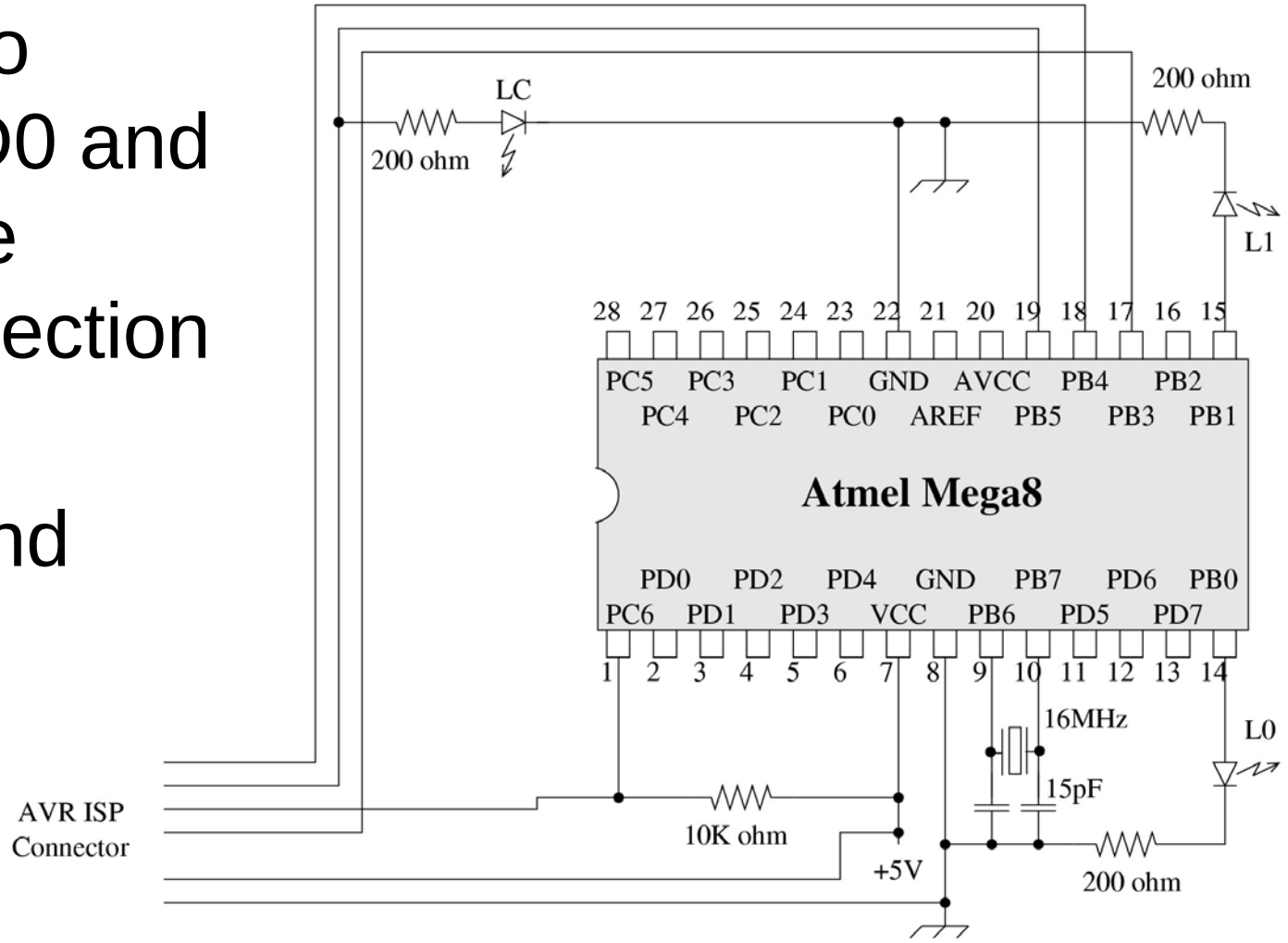
First Circuit

Pieces you need to assemble (after drawing a circuit diagram):

- Atmel with crystal
- Programming interface
- LEDs
  - Circle of LEDs for displaying heading or heading error
  - Line of LEDs for displaying heading velocity

Circuit Starting Point

- Remember to reserve PD0 and PD1 for the serial connection to the heli
- PB0, PB1, and PB2 are available



Reading the Compass

int16_t get_heading(void)

- Returns the heading in 10ths of a degree: values between -1799 and 1800

Reading the Compass

- Your atmel sends: ‘c’ (1 character)
- The heli responds with:

  “cDDDD\n\r”

  – There are always 4 decimal digits
  – Value is between 0000 and 3599

Reading the Compass

int16_t get_heading(void)

- Ask for the heading from the heli
- Translate the characters received from the heli into a number between 0 and 3599
- Translate this number to one that is between -1799 and 1800
  - Note: the heading that is represented must be the same after this transformation

Computing Error

int16_t compute_error(int16_t goal, int16_t heading)

Returns the heading error in 10^{th}s of a degree:

- error = goal - heading
- But: return value must be between -1799 and 1800

- Positive errors correspond to the craft being clockwise from the goal

Computing Velocity

int16_t compute_derivative(int16_t heading_last,
int16_t heading_current)

Returns the heading velocity in 10^{th}s of a degree per second:

- Assume the two samples are 100ms apart
- As with the error computation, you must handle the “wrap-around” cases

Displaying Orientation

void display_orient(int16_t theta)

Display either an absolute heading or a heading error using a set of LEDs

- Use 4 LEDs for this
- How do you decide when to turn on each of the LEDs given theta? Can you do better than 90 degree resolution?

Displaying Velocity

void display_derivative(int16_t velocity)

Display the rotational velocity using a set of LEDs

- 5 LEDs
- How do you decide when to turn on each of the LEDs given theta?

Main Loop

Wrap all of the pieces together:

- Every 100ms, take an orientation sample, compute error and derivative, update display
- See the code skeleton in the project specification