Last Time

Digital I/O and the Atmel Mega8s

• DDRx
• PORTx
• PINx
Today

- Project 1
- Bion programming
- Serial I/O
Schedule

• Project 1: Due March 5\textsuperscript{th} (1 week)
• HW 1 & 2: coming back early next week
• HW 3: out tonight. Due March 10\textsuperscript{th} at the beginning of class
• March 10\textsuperscript{th}: Midterm review
• March 12\textsuperscript{th}: Midterm
  – See prior class web pages for exams and solution sets
Bion

Sensor network:
• 1000 sensor nodes
• 3 miles of telephone cable

Wilhelm Reich
Project 1: Digital I/O and Timing

• Control of LEDs and Speaker
  – Precise timing requires timer use
• Respond to button presses
Part 1

• Internal 4-bit (software) counter
• Counter state is reflected by the LEDs
  – Bit 0 (LSB): Blue
  – Bit 1: Green
  – Bit 2: Red
  – Bit 3: Yellow
Part 1

• Each button release:
  – Increment counter
  – Show the new state of the counter with the LEDs
Part 2

• Generate tone with the speaker
  – Different tone for each counter state (higher frequencies for higher values)
  – This tone should be produced continuously (no pauses)

• Speaker is controlled by a digital I/O line
  – So: in one of two states
  – Tones are produced by producing a “square wave” at a given frequency
Required Components

• Modular code
  – E.g., implement a separate function that translates the current counter value into the LED state
Project Administrivia

Due on March 5\textsuperscript{th}

• Demonstrate to me, or Di Wang
• Documented code: hand-in on D2L
  – One copy per 2-person group
• Personal report: distribution of work
  – You will not receive a grade if this is not turned in
Bion Care

• Hold bions on the side of the board (don’t touch the components)
• Minimize the bending of the components
• Don’t let the bion come in contact with metal while it is powered on

• If things get hot: disconnect power immediately and ask for help
Getting Started

See:  http://www.cs.ou.edu/~fagg/classes/general/atmel/

Summary:

• (perhaps) Install AVRstudio
• Install WinAVR
• Plug the programmer into your computer
• Plug the programmer into the bion
• Plug the power into the bion
• Create a program
Downloads from Atmel HOWTO

- libou_atmega8.a
- oulib.h
- oulib_serial_buffered.h
- makefile (OSX and linux)
Compiling and Downloading (the easy way)

• Obtain a copy of the “makefile”
  – Modify the “TARGET” line for your program
• Type “make”
  – You should see no errors
• Type “make program”
  – This will download your code to the bion
  – Again, you should see no errors
Getting Started
Project Menu: New Project
Back to the OS…

Copy the following to your “firstproject” folder:

• oulib.h
• libou_atmega8.a
• (useful later): oulib_serial_buffered.h
Project Menu: Configuration Options

1. Use External Makefile
   - Target name must equal project name.
   - Clean/rebuild support requires "clean" target.
   - Makefile and target must exist in the same folder

2. Device: atmega8
   - Frequency: ___ Hz
   - Optimization: -Os
   - Options: Unsigned Chars (-funsigned-char),
              Unsigned Bitfields (-funsigned-bitfields),
              Pack Structure Members (-fpack-struct),
              Short Enums (-fshort-enums)

Create Hex File, Generate Map File, Generate List File
Project Menu: Configuration Options

1. General
2. Include Directories
3. Libraries
4. Memory Settings

2 (add a dot)
Project Menu: Configuration Options

1. Edit Custom Compilation Options
2. Add or remove options as needed
3. Set external tools for compilation and linking
Project Menu: Configuration Options

1. Custom Compilation Options
2. Add external tools
3. Edit external tools
4. Edit Custom Options
Right click to add oulib.h
Now for the code...

```
#include "oulib.h"

int main(void)
{
    DDRB = 1;

    while(1) {
        PORTB = 1;
        delay_ms(500);
        PORTB = 0;
        delay_ms(500);
    }
}
```
Build menu: Build
You should get this
Now We Are Ready…

• Plug the programmer into the bion (If it is not already)
• Power up the bion
• And download the program…
  – Tools Menu: AVR: Connect
```c
int main(void)
{
    DDRB = 7;
    while(1) {
        PORTB = 1;
        delay_ms(500);
        PORTB = 0;
        delay_ms(500);
    }
}
```

```
eeprom --set-section-flags=.eeprom="alloc,load" --change-section-lma .eeprom=0 -o ihex firstproject.elf firstproject.eep
```

6 bytes (39.4% Full)
+ .bootloader)
6 bytes (1.6% Full)
  .noinit)
Ideal: 2 MHz
(should only need to do this once)
Specify your hex file
Flashing?

Your program will start executing as soon as the download is complete …

Your green Light Emitting Diode should be blinking at 1 Hertz (once per second)