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
  – Implement proper debouncing
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

• Software debounce of the button (switch2)
• Modular code
  – E.g., implement a separate function that translates the current counter value into the LED state
Project Administrivia

Due on March 3rd

• Demonstrate to me, or Di Wang
  Documented code: hand-in on D2L
    – One copy per 2-person group

• Personal report: distribution of work
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
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_serialbuffered.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: [ ]
   - Optimization: [-Os]

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

1. Include Directories
2. Library Search Path (add a dot)
3. Available Link Objects: libu_atmega8.a
4. Link with These Objects: Add Library...
Project Menu: Configuration Options
Project Menu: Configuration Options

1. Custom Compilation Options
2. Add
3. External Tools
4. OK/Cancel
Right click to add oulib.h.
Code goes here
Now for the code...

```c
#include "oulib.h"

int main(void)
{
    DDRB = 1;

    while(1) {
        PORTB = 1;
        delay_ms(500);
        PORTB = 0;
        delay_ms(500);
    }
}
```
Build menu: Build

```c
int main(void)
{
    DCON = 1;
    while(1) {
        PORTS = 1;
        delay_ms(500);
        DCON = 0;
        delay_ms(500);
    }
}
```
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);
    }
}
```

```
.eprom --set-section-flags=.eeprom="alloc,load" --change-section-lma .eeprom=0 -0 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)
1. Specify your hex file.

```c
int main(void) {
    DDRB = 1;
    while(1) {
        PORTB = 1;
        delay_ms(500);
        PORTB = 0;
        delay_ms(500);
    }
}
```
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)