Binary Representations,
Serial Communication
and the Atmel 2560
Administration…

• Top Hat or Zyante problems?
Questions?
Atmel Mega2560 Microcontroller
Atmel Mega2560
Atmel
Mega2560

Pins are organized into 8-bit “Ports”:

- A, B, C … L
- But no “I”
Digital Input/Output

• Each port has three special-purpose registers that control its behavior.
  • Each pin can either be an input or output!

• For port B, they are:
  • DDRB: data direction register B
  • PORTB: port output register B
  • PINB: port input B
Data Direction Register: DDRx

- 8-bit wide register
  - Controls one pin with each bit
- 0 -> this is an input pin
- 1 -> this is an output pin

- Note: only configure pins as an output if you really mean it!
Port Output Register: PORTx

- Also one pin per bit
- If configured as an output:
  - 0 -> the pin is held at 0 V
  - 1 -> the pin is held at +5 V
Port INput register: PINx

- One pin per bit
- Reading from the register:
  - 0 -> the voltage of the pin is near 0 V
  - 1 -> the voltage of the pin is near +5 V
- If nothing is connected to the pin, then the pin will appear to be in a random state
A First Circuit
A First Program

Flash the LEDs at a regular interval

• How do we do this?
A First Program

main() {
    DDRC = ???;

    while(1)
    {
    }
}

Andrew H. Fagg: Embedded Real-Time Systems: Digital IO
A First Program

```c
main() {
    DDRC = 0x3;

    while(1) {
        PORTC = 0x1;       // sets PC0 to 1
        delay_ms(100);
        PORTC = 0x0;       // set PC0 to 0
        delay_ms(100);
    }
}
```
A Second Program

```c
main() {
    DDRC = 3;   // Set port C pins 0, and 1 as outputs

    while(1) {
        PORTC = 0x3;
        delay_ms(250);
        PORTC = 0x1;
        delay_ms(250);
        PORTC = 0x2;
        delay_ms(250);
        PORTC = 0x0;
        delay_ms(250);
    }
}
```

**What does this program do?**
A Second Program

```c
main() {
    DDRC = 3;  // Set port C pins 0, and 1 as outputs

    while(1) {
        PORTC = 0x3;
        delay_ms(250);
        PORTC = 0x1;
        delay_ms(250);
        PORTC = 0x2;
        delay_ms(250);
        PORTC = 0x0;
        delay_ms(250);
    }
}
```

Flashes LED on PC1 at 2 Hz on PC0: 1 Hz

Duty Cycle for each: 50%
Port-Related Registers

Some of the C-accessible registers for controlling digital I/O:

<table>
<thead>
<tr>
<th></th>
<th>Directional control</th>
<th>Writing</th>
<th>Reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port B</td>
<td>DDRB</td>
<td>PORTB</td>
<td>PINB</td>
</tr>
<tr>
<td>Port C</td>
<td>DDRC</td>
<td>PORTC</td>
<td>PINC</td>
</tr>
<tr>
<td>Port D</td>
<td>DDRD</td>
<td>PORTD</td>
<td>PIND</td>
</tr>
</tbody>
</table>
... go to Bit Manipulation
A Third Program

If switch reads zero, turn PC0 on and PC1 off

Otherwise, turn PC0 off and PC1 on
main() {
    DDRC = 0x3;
    while(1) {
    }
}
A Third Program

main() {
    DDRC = 0x3;

    while(1)
    {
        if(PINC & 0x80) {
            PORTC = 0x2;
        }else{
            PORTC = 0x1;
        }
    }
}

Arduino Mega Board

(see schematic)
Quiz
Input/Output Systems

Processor needs to communicate with other devices:
- Receive signals from sensors
- Send commands to actuators
- Or both (e.g., disks, audio, video devices, other processors)
I/O Systems

Communication can happen in a variety of ways:
- Binary parallel signal
- Analog
- Serial signals
Serial Communication

• Communicate a set of bytes using a single signal line

• We do this by sending one bit at a time:
  • The value of the first bit determines the state of a signal line for a specified period of time
  • Then, the value of the 2nd bit is used
  • Etc.
Serial Communication on the Mega2560

Our mega 2560 has FOUR Universal, Asynchronous serial Receiver/Transmitters (UARTs):

• Each handles all of the bit-level manipulation
  • Your software only worries about the byte level

• UART #1 is attached to the USB connection between the Arduino board and your laptop
Mega2560 UART C Interface

Lib C support (standard C):

- `char fgetc(fp)`: receive a character
- `fputc(’a’, fp)`: put a character out to the port
- `fputs(”foobar”, fp)`: put a string out to the port
- `fprintf(fp, ”foobar %d %s”, 45, ”baz”):` put a formatted string out to the port
Mega2560 UART C Interface

OUlib support:

```c
fp = serial_init_buffered(1, 38400, 40, 40)
```

Initialize port one for a transmission rate of 38400 bits per second (input and output buffers are both 40 characters long)

Note: declare fp as a global variable:

```c
FILE *fp;
```

```c
serial_buffered_input_waiting(fp)
```

Is there a character in the buffer?

See the Atmel HOWTO: examples_2560/serial
Reading a Character from the Serial Port

```c
int c;

c=fgetc(fp);
```

Note: `fgetc()` “blocks” until a byte is available
• Will only return with a value once a character is available to be returned
Processing Serial Input

int c;
while(1) {
    if(serial_buffered_input_waiting(fp)) {
        // A character is available for reading
        c = fgetc(fp);
        //do something with the character>
    }
    //do something else while waiting>
}

serial_buffered_input_waiting(fp) tells us whether a byte is ready to be read
Mega2560 UART C Interface

Also available:

- `fscanf()`: formatted input

See the LibC documentation or the AVR C textbook
Next Time

Project 0: compiling and downloading for the mega 2560