Joseph Henry & the Telegraph

- Albany Academy Experiment
- Assisted Morse at Princeton
- 1st Head of Smithsonian
- Unit of inductance: Henry

Morse Code

- Simple sequences of short and long clicks to represent letters and numbers
- Easier to generate than sound
- Easier to distinguish than sound

UPC Codes

- Bars come in four widths 1-2-3-4
- Start is 1-1-1 (black-white-black)

- 0 = 3-2-1-1 (space-bar-space-bar).
- 1 = 2-2-2-1 (space-bar-space-bar).
- 2 = 2-1-2-2 (space-bar-space-bar).
- 3 = 1-4-1-1 (space-bar-space-bar).
- 4 = 1-1-3-2 (space-bar-space-bar).
- 5 = 1-2-3-1 (space-bar-space-bar).
- 6 = 1-1-4-4 (space-bar-space-bar).
- 7 = 1-3-4-2 (space-bar-space-bar).
- 8 = 1-2-4-3 (space-bar-space-bar).
- 9 = 3-1-1-2 (space-bar-space-bar).

Communicating With Pulses

- PCM: Pulse Code Modulation
**PCM: Pulse Code Modulation**

<table>
<thead>
<tr>
<th>Signal</th>
<th>Digit 1</th>
<th>Digit 2</th>
<th>Digit 3</th>
<th>Digit 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0 0 1 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0 0 1 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0 1 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0 1 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0 1 1 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0 1 1 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1 0 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1 0 1 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1 0 1 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1 1 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1 1 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1 1 1 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1 1 1 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PWM: Pulse Width Modulation**

- Signal is compared to a sawtooth wave producing a pulse width proportional to amplitude.

---

**What Can Be Done With PWM?**

- Question: What happens if voltages like the ones above are connected to a light bulb?
- Answer: The longer the duty cycle, the longer the light bulb is on and the brighter the light.

- Average power can be controlled
- Average flows can also be controlled by fully opening and closing a valve with some duty cycle
PULSE WIDTH MODULATION

- Pulse Width Modulation (PWM) involves the generation of a series of pulses at a fixed period and frequency.
- The duty cycle defines the width of each pulse which is varied to generate waveforms.

Choosing the -3 dB point at 4 kHz, and using the Relation:

\[ RC = \frac{1}{2 \cdot \pi \cdot f} \]

we get \( R = 4 \, k \), if \( C \) is chosen as 0.01 mF:

- \( R = 4.0 \, k \)
- \( C = 0.01 \, mF \)

PULSE WIDTH MODULATION

- A simple low pass filter is then used to generate an output voltage directly proportional to the average time spent in the HIGH state.
- (i.e., 50% duty cycle is equal to 2.5 volts when \( VDD = 5.0V \)).
/* Controls a LED that can be directly connected from OC1A to GND. The brightness of the LED is controlled with the PWM. After each period of the PWM, the PWM value is either incremented or decremented */

int main (void)
{
    ioinit ();
    /* loop forever, the interrupts are doing the rest */
    for (;;)
    {
        return (0);    
    }
}

void ioinit (void)
{
    /* tmr1 is 10-bit PWM */
    TCCR1A = _BV (PWM10) | _BV (PWM11) | _BV (COM1A1);
    /* tmr1 running on full MCU clock */
    TCCR1B = _BV (CS10);
    /* set PWM value to 0 */
    OCR1A = 0;
    /* enable OC1 and PB2 as output */
    DDRD = _BV (PD5);
    timer_enable_int (_BV (TOIE1));
    /* enable interrupts */
    sei ();  
}
SIGNAL (SIG_OVERFLOW1)
{
    switch (direction)
    {
        case UP:
            if (++pwm == 1023)
                direction = DOWN;
            break;
        case DOWN:
            if (--pwm == 0)
                direction = UP;
            break;
    }
    OCR1A = pwm; /* triangle */
}