Project: security lock system

• Idea:
  – Create a 4bit password using switches
  – Send data serially (one bit at a time) using an LED
  – Receive the code serially and convert data into a 4 bit number
  – Compare the received data with original code
  – Unlock the key if it matches!

• Potential uses: TV/ DVD/Car unlock … remote control

• Components:
  – Timer -- Switch
  – Shift Register -- Comparators
  – Amplifier

• Optional: extending the system into 8-bit
LM555 Timer

- Used as an oscillator to drive a speaker
- **Trigger:** when < $\frac{1}{3}$ Vcc, the output is high (Vcc)
- **Threshold input:** when > $\frac{2}{3}$ Vcc and the trigger is > $\frac{1}{3}$ Vcc, the output is low (0V). If the trigger is < $\frac{1}{3}$ Vcc, it overrides the threshold input and holds the output high.
- **Reset input:** when less than about 0.7V, all other inputs are overridden and the output is low.
- **Discharge pin:** This is connected to 0V when the timer output is low and is used to discharge the timing capacitor in astable operation.
LM555 Timer as an oscillator

- Astable operation: The circuit oscillates on its own.
- With the output high, the capacitor C is charged by current flowing through $R_A$ and $R_B$.
- The threshold and trigger inputs monitor the capacitor voltage and when it reaches $\frac{2}{3}V_{cc}$ (threshold), the output becomes low and the discharge pin is connected to 0V.
- The capacitor discharges with current flowing through $R_B$ into the discharge pin. When the voltage falls to $\frac{1}{3}V_{cc}$ (trigger) the output becomes high again and the discharge pin is disconnected, allowing the capacitor to start charging again.
- Adjust duty cycle (time on : total time) by adjusting the ratio between $R_A$ and $R_B$.
- Note that pin 4 (reset) is held at Vcc here. You will need change the connection for light sensitivity.

LM555 Timer

• Some equations for astable operation:
The charge time (output high) is given by:
\[ t_1 = 0.693 \,(R_A + R_B) \, C \]
And the discharge time (output low) by:
\[ t_2 = 0.693 \,(R_B) \, C \]
Thus the total period is:
\[ T = t_1 + t_2 = 0.693 \,(R_A + 2R_B) \, C \]
The frequency of oscillation is:
\[ f = 1/T = 1.44/ \,(R_A + 2R_B) \, C \]
And the duty cycle is:
\[ D = t_1/(t_1 + t_2) = (R_A + R_B)/(R_A + 2R_B) \]
Comparator

- Built using an op-amp (a 741 will do)
- Compares it’s “+” and “-” inputs
  - If $V^+ > V^-$ then output = $V_{\text{High}}$ (a digital “1”)
  - If $V^+ < V^-$ then output = $V_{\text{low}}$ (a digital “0”)
- Useful for converting small analog voltages into big, digital signals
- To power up, attach $V_{\text{low}}$ to -6V, $V_{\text{high}}$ to +6V
- Test: attach output to LED in series with a 1kΩ resistor to ground
- Set $V^+$, $V^-$ with SMUs, confirm that LED turns on when $V^+ > V^-$
Shift register (1)

- A shift register is a kind of digital memory
- It has 6 data inputs:
  - Parallel data D0, D1, D2, D3
  - Serial data DSR, DSL
- It has three controls:
  - Shift controls, S0, S1
  - Clock
- It has 4 outputs:
  - Q0, Q1, Q2, Q3
  - These outputs change only when the clock changes from 0 to 1

Set VCC to 5V, VSS to 0V, pin 1 to 5V
Shift register (2)

• The shift register has 4 modes, set by S0, S1, and triggered by the clock
  • When S0=1, S1 =1,
    – Q0 = D0, Q1 = D1, etc
  • When S0 = 0, S1 = 0
    – Q0,Q1,Q2,Q3 hold their value
  • When S0 = 0, S1 = 1
    – Data shifts left: Q1 = Q0 (from before clock) Q2= Q1, etc
    – Q0 = DSR
  • When S0 = 1, S1 = 0
    – Data shifts right: Q2 = Q3 (from before clock) Q1= Q2, etc
    – Q3 = DSL

• Test:
  – attach outputs to 4 LEDs in series with 1kΩ resistors to ground
  – Set function generator to make a 5V square wave (2.5V offset) with frequency = 1Hz, attach it to the clock input
  – Short D0, D2, D3, SDR to ground, short D1, SDL to 5V

• Try different combinations of S0, S1.
• What happens?
  – You should see things shift left or right.
Comparator (1)

- Includes combination of AND and XOR gated:

- **AND:**
  - A
  - B
  - Y
  - Y = 1 if A = B = 1, otherwise Y = 0

- **XOR:**
  - A
  - B
  - Y
  - False if A = B
  - True if A ≠ B

- **NOT**
  - A
  - Y
  - If A = 1, Y = 0, if A = 0, Y = 1
Comparator (2)

• Comparing two 2-bit numbers:

\[ A = A_2, A_1 \]
\[ B = B_2, B_1 \]

1 when \( A = B \)
0 when \( A \neq B \)
Latch circuit

Question: write a truth table for this circuit: what happens when A=B=1?

Test: hook up and gates as you did yesterday, test its function: does it behave as you predicted?