

CSE 140L

# Lab 1 Assignment – UCSD Alarm Clock

Alex Orailoglu

08/06/13

# Behavioral Description

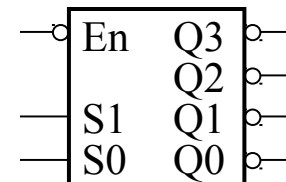
- **Part 1: Basic alarm clock**
  - User can adjust the clock time and the alarm time.
  - A single set of LCDs that display:
    - The alarm time when the user is setting the alarm;
    - The current time in all other cases.
  - User can turn the alarm on or off.
  - If the alarm is on, the buzzer goes high when
    - clock hour = alarm hour, and
    - clock minutes = alarm minutes.

# Behavioral Description

- **Part 1: Basic alarm clock**
- **Part 2a: Differentiating between MW vs. TuTh alarm times**
  - The alarm clock keeps track of the day information, and displays it on 7-seg Disp I/O
  - User can adjust the day
  - Two alarm times
    - One for **MonWedFriSun** and one for **TueThuSat**
    - Alarm clock automatically selects the effective alarm time for buzzing based on the day of the week information
    - Each alarm can be turned on/off and adjusted separately
- **Part 2b: Disabling Extended Weekend Alarm**
  - Friday & Weekend alarms automatically disabled

# Components Provided for Part 1

- **ClockDev:**
  - Seconds are incremented at each CLK cycle as in a regular clock. When its *Set* input is set, clock time can be adjusted by incrementing hours and minutes using *SM* (*Set Minute*) and *SH* (*Set Hour*) inputs.
- **AlarmDev:**
  - Very similar to **ClockDev** except that this block has neither a CLK input nor output for seconds.
- **LCD Interface:**
  - Takes a 6-bit input number (0 to 59) and displays it on 7-segment displays as two decimal digits.
- **6Comp:**
  - 6-bit comparator that gives a '1' output if the two inputs are the same.
- **MUX-2x6:**
  - 6-bit 2-to-1 multiplexer that selects one of the two 6-bit inputs and directs it to its output.
- **Decoder-4 (LogicWorks library component):**
  - If  $En == 1$ , all output = 1;
  - If  $En == 0$ , For  $i = 0$  to 3,  $Q_i = 0$  if  $S_1S_0 == i$ ;

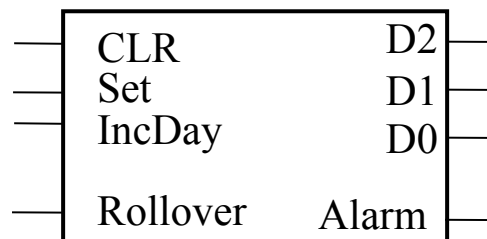


# Components to be Implemented for Part 2a

- You will be making a DayCounterA
  - A modulo-7 counter, counting from 0 to 6, then back to 0
  - Encoding of days:

| Day              | Monday    | Tuesday    | Wednesday | Thursday   | Friday    | Saturday   | Sunday    |
|------------------|-----------|------------|-----------|------------|-----------|------------|-----------|
| Counter Encoding | 0         | 1          | 2         | 3          | 4         | 5          | 6         |
| Alarm Selection  | First (0) | Second (1) | First (0) | Second (1) | First (0) | Second (1) | First (0) |

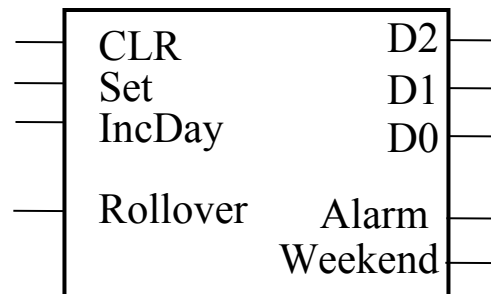
- If CLR = 1, counter is reset to 0 (Monday)
- If Set = 1, counter will increment if IncDay transitions from 0 to 1
- Counter will increment when Rollover transitions from 0 to 1 (unless CLR = 1)



# Part 2b: No Weekend Alarms

Mostly the same as Part A, but alarm should not buzz if the current day is Friday, Saturday, or Sunday

- Create a new component, called a DayCounterB, which is like a DayCounterA except that it has an additional “Weekend” output which indicates that the current day is Fri, Sat, or Sun



# Counter-4 UP wo/EN

Use this component to implement counting functionality.

- Inputs:

- CLK – Increments the counter when it transitions from 0 to 1
- CLR – If CLR = 1, the counter is reset
  - Counter can't increment until CLR = 0
- LD – Load from D inputs. (Not necessary for this lab)
- D3..0 – The value to load into the counter. (Not necessary)

- Outputs:

- Q3..0 – The current value of the counter.
- C0 – “Carry-out”, becomes 1 when counter rolls over (Not necessary for this lab)

- By default, the Counter-4 wo/EN counts to 7 before resetting.

- You'll have to add glue logic that resets after counting to 6.

