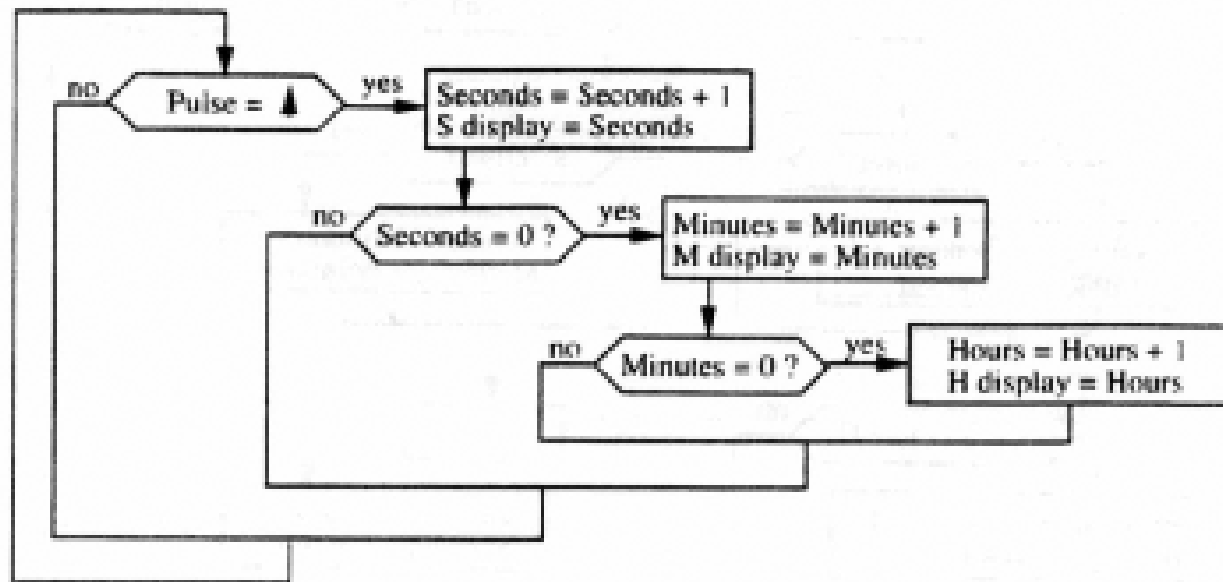


Fall2018 CSE 140L

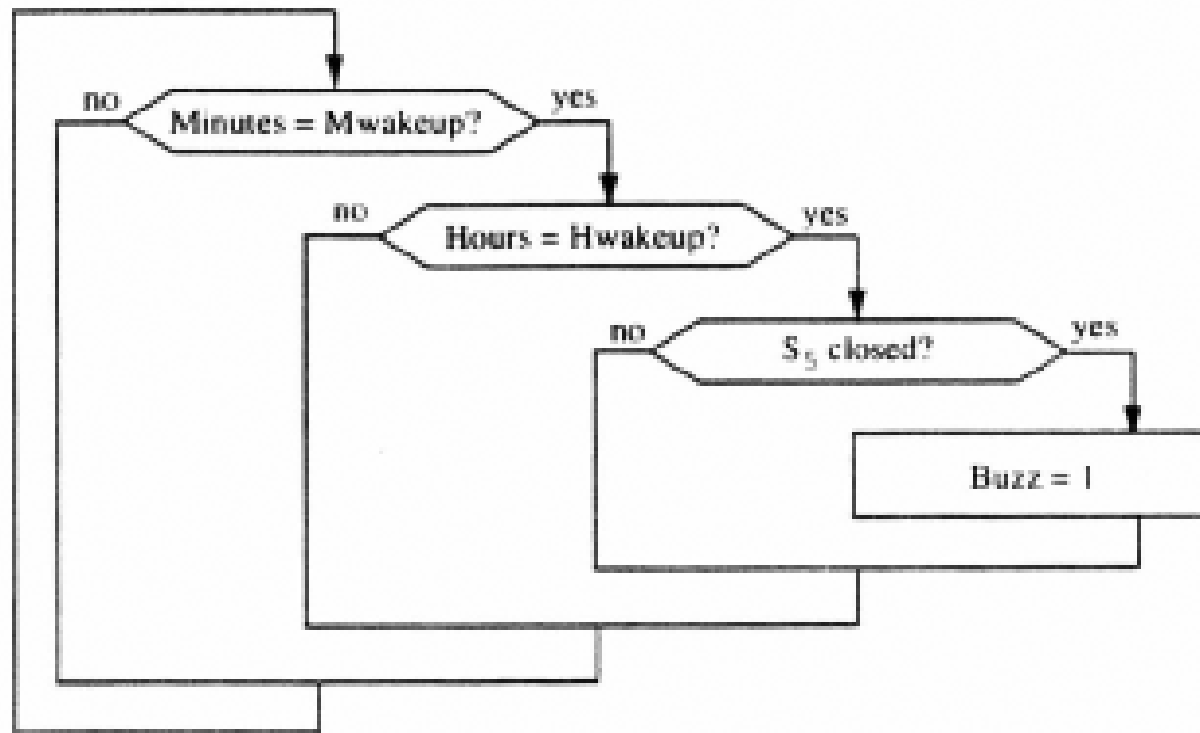
Lab 2 Assignment – UCSD Alarm Clock

Oct 2018

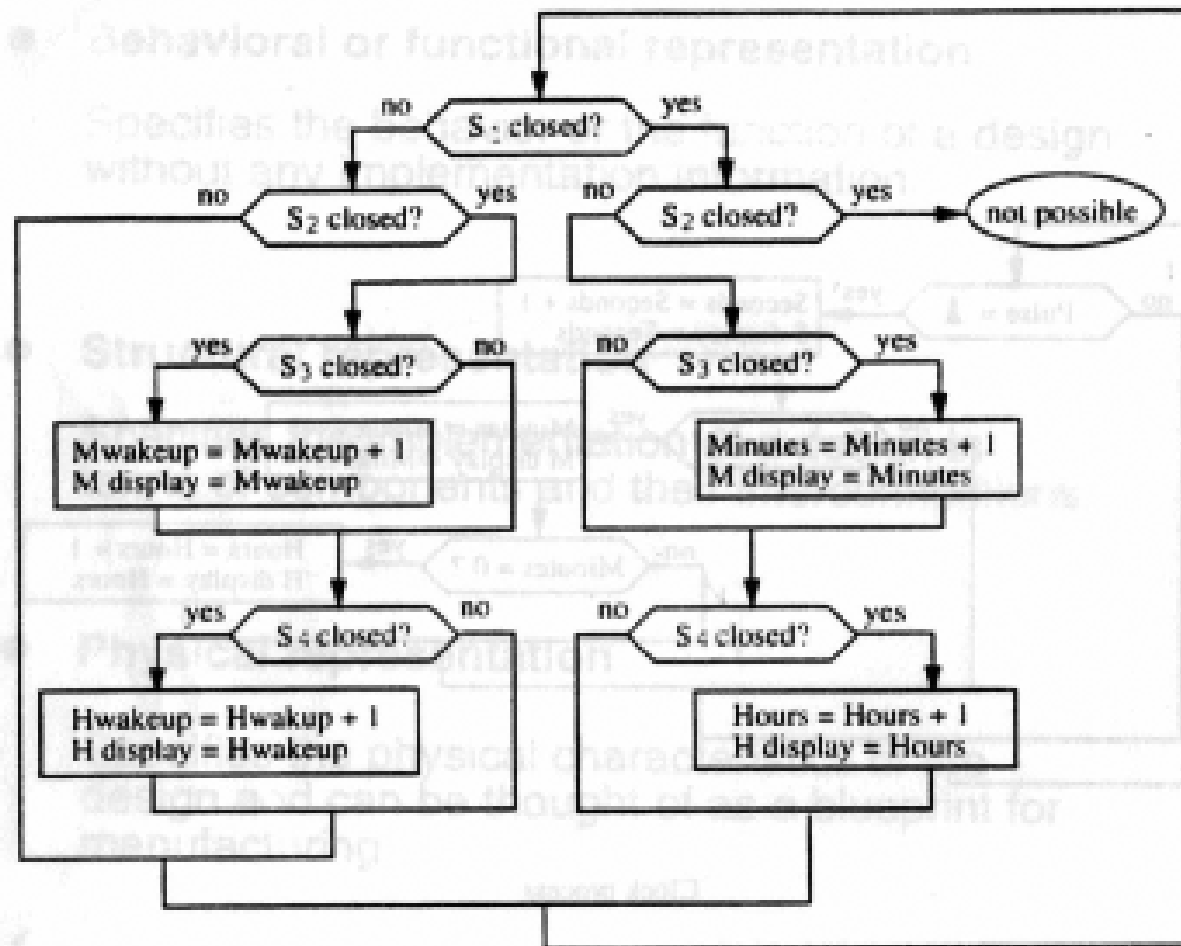
Clock Behavioral Diagram



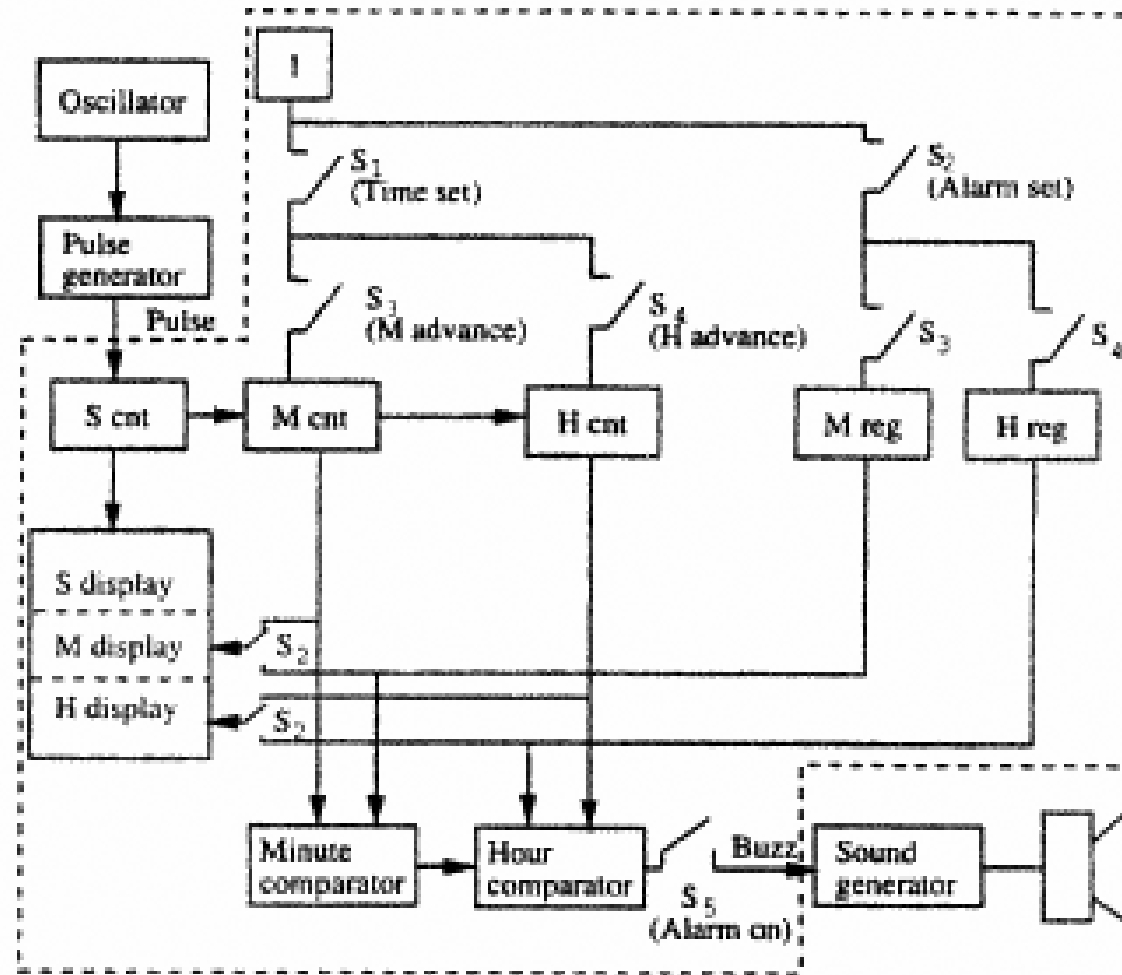
Alarm Behavioral Diagram



Combined Behavioral Diagram



Structural Diagram



Behavioral Description

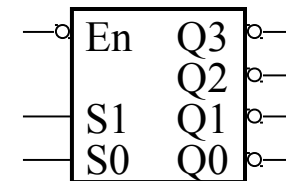
- **Part 1: Basic alarm clock**
 - User can adjust the clock time and the alarm time.
 - A single set of LCDs that display:
 - The alarm time when the user is setting the alarm;
 - The current time in all other cases.
 - User can turn the alarm on or off.
 - If the alarm is on, the buzzer goes high when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

Behavioral Description

- **Part 1: Basic alarm clock**
- **Part 2: Days and Months**
 - The alarm clock keeps track of the day of week, day of month, and month of year information, and displays it on 7-seg Disp I/O
 - Requires 6 new counters
 - DayOfWeek, DayOfMonth, MonthOfYear for current time
 - DayOfWeek, DayOfMonth, MonthOfYear for alarm time
 - Should be displayed in the same manner as in Part 1, where:
 - The alarm date is displayed when setting the alarm
 - The current date is displayed otherwise

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 (SystemVerilog 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 2

- **DayOfWeek**
 - Represents day of week.
 - Should use provided Counter-4 for basis of implementation.
 - (Current Time) Should count from 1 to 7.
 - (Current Time) Can be incremented via SPDT Pushbutton (independently of **DayOfMonth**) or through rollover of hours (in parallel with **DayOfMonth**).
 - (Alarm Time) Should count from 0 to 7.
 - (Alarm Time) Can be independently incremented via SPDT Pushbutton.
 - (Alarm Time) If set to 0, then the value is ignored when evaluating logic for the BUZZER.

Components to be Implemented for Part 2

- **DayOfMonth**
 - Represents day of month
 - Should use provided Counter-8 for basis of implementation
 - (Current Time) On months 1, 3, 5, 7, 8, 10, and 12, this should count from 1 to 31.
 - (Current Time) On months 4, 6, 9, and 11, this should count from 1 to 30.
 - (Current Time) On month 3, this should count from 1 to 28.
 - (Current Time) Can be incremented via SPDT Pushbutton (independently of **DayOfWeek**) or through a rollover of the hours (in parallel with **DayOfWeek**).
 - (Alarm Time) Should **always** count from 0 to 31
 - (Alarm Time) Can be independently incremented via SPDT Pushbutton
 - (Alarm Time) If set to 0, then the value is ignored when evaluating logic for the BUZZER

Components to be Implemented for Part 2

- **MonthOfYear**
 - Represents month of the year
 - Should use provided Counter-4 for basis of implementation
 - (Current Time) Should count from 1 to 12
 - (Current Time) Can be incremented only through rollover of **DayOfMonth**
 - (Alarm Time) Should count from 0 to 12
 - (Alarm Time) Can be independently incremented via SPDT Pushbutton
 - (Alarm Time) If set to 0, then the value is ignored when evaluating logic for the BUZZER

Component Increment Rules for Part 2

- **Current Time**
 - **DayOfWeek** and **DayOfMonth** are increment independently when setting the current time.
 - **DayOfWeek** and **DayOfMonth** are incremented in parallel upon a rollover of the hours.
 - **MonthOfYear** cannot be set directly.
 - Ensure always correct date by appropriate rollover.
- **Alarm Time**
 - All counters can be set independently, even to an invalid date.

Making Custom Components

To make a custom component:

1. Design a circuit that implements the component's functionality
2. Identify the component's inputs and outputs, attach ports to your circuit accordingly
 - You must name your ports: right click the port and click “name”
3. Open a new file (File → New → Device Symbol)
4. Import pin list from existing design
 1. Options → Subcircuit and Part Type → Create subcircuit symbol and select open circuit to attach to it
5. Draw your circuit
6. Place the pins on the circuit (from the left)
 - Click the appropriate buttons for the desired direction of pins
7. Save the component in a library of your choice