

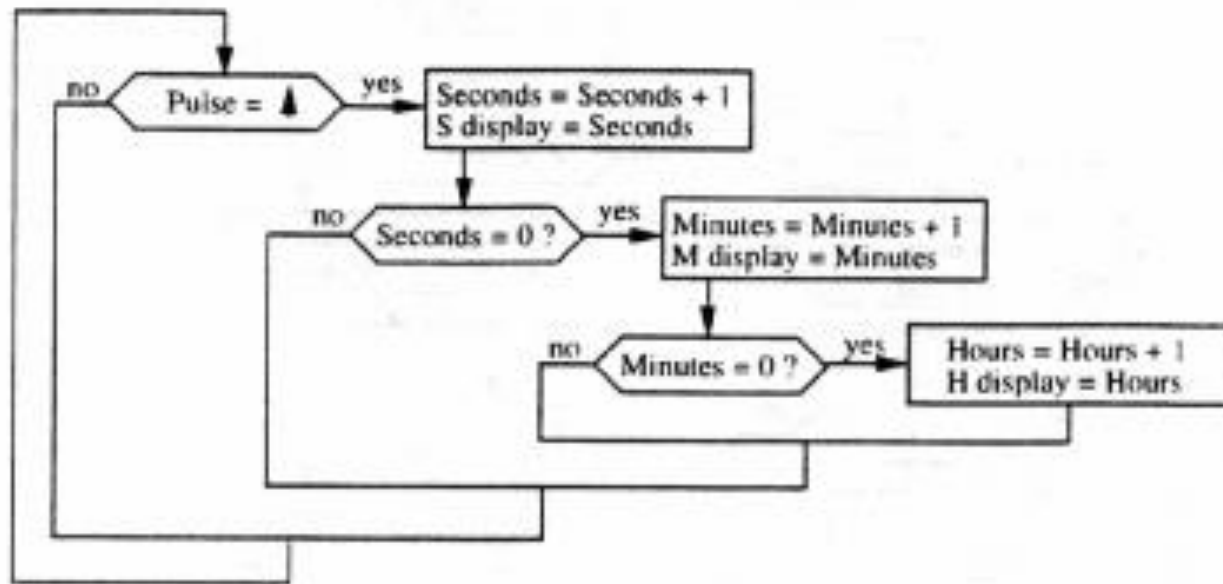
Fall 14 CSE 140L

Lab 1 Assignment – UCSD Alarm Clock

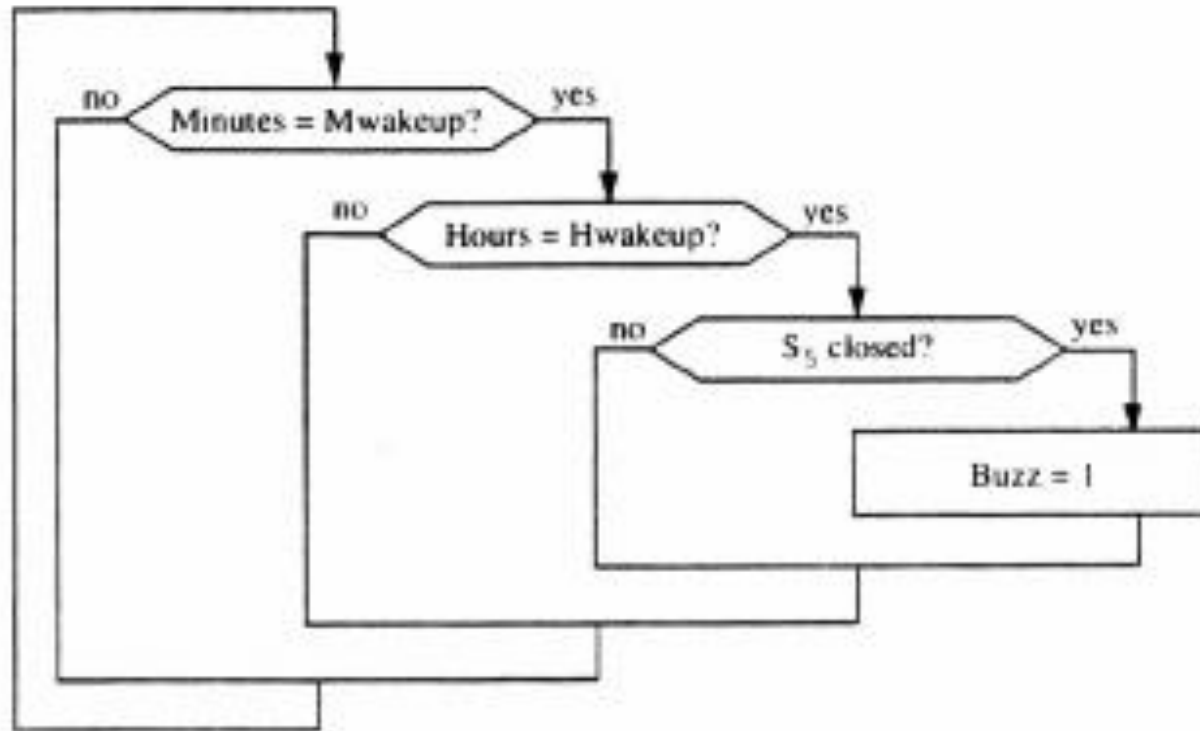
Alex Orailoglu

10/06/14

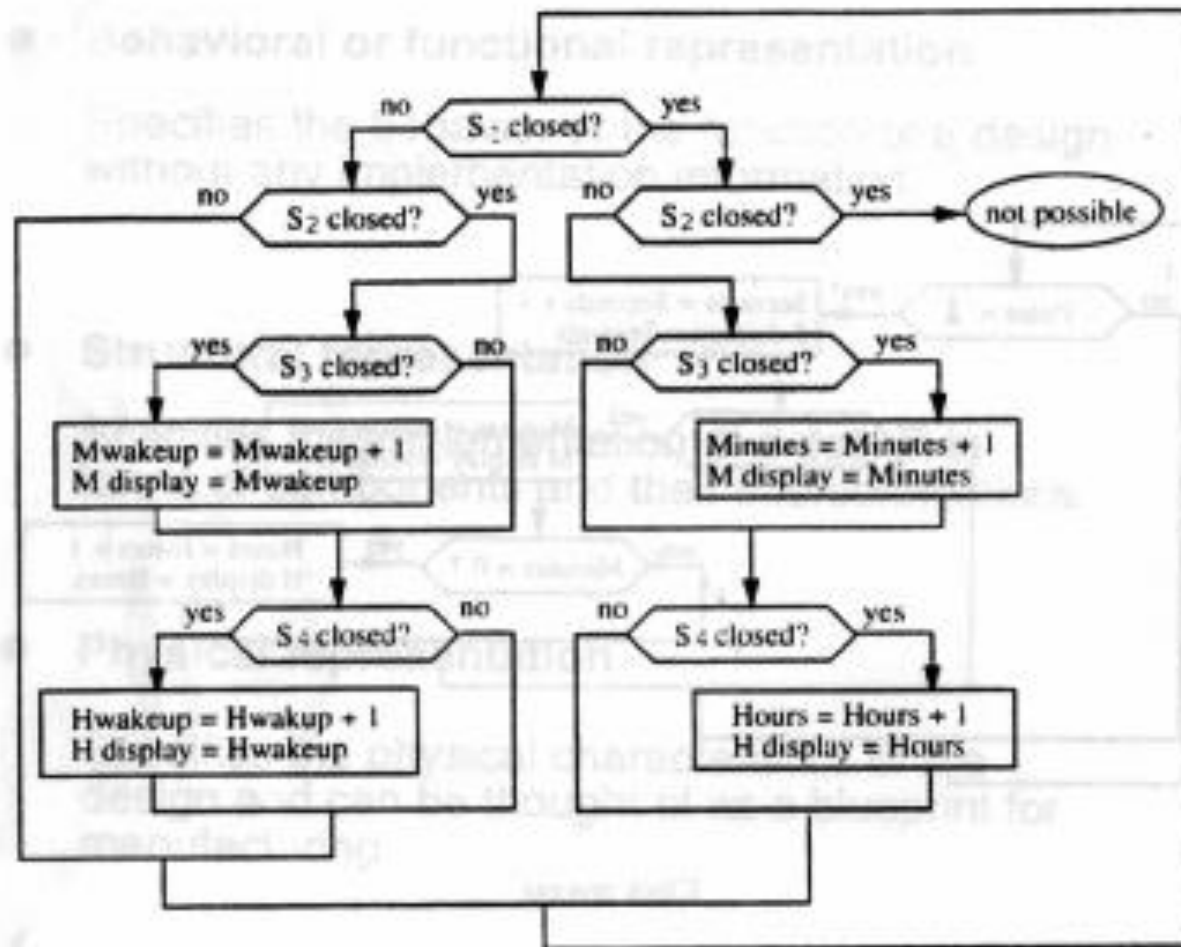
Clock Behavioral Diagram



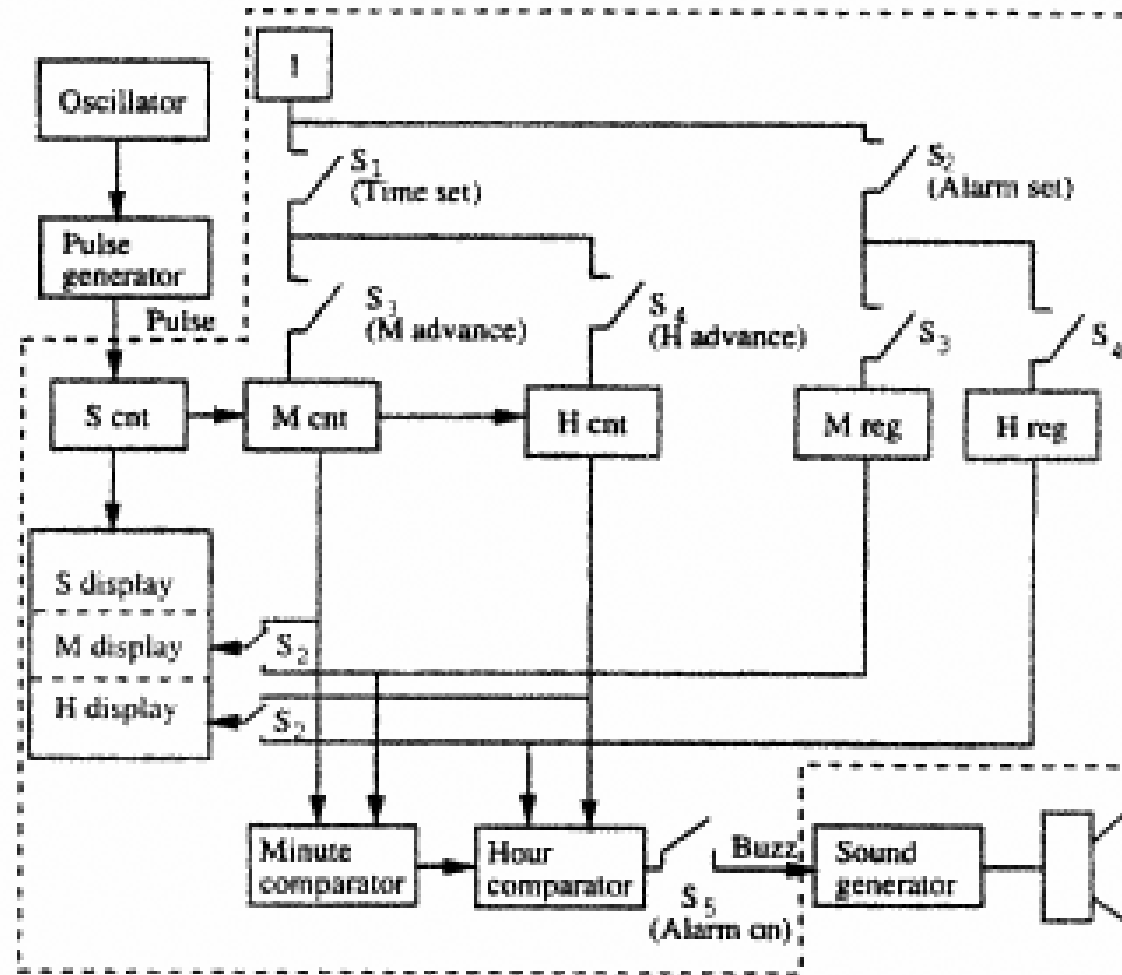
Alarm Behavioral Diagram



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

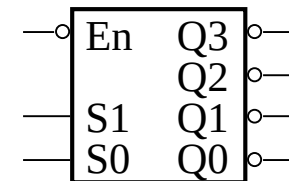
- Part 1: Basic alarm clock
- **Part 2: Days of the Week**
 - The alarm clock keeps track of the day of the week and displays it on a *7-seg LCD Display*.
 - Counts from 0 to 6 (each number represents a different day of the week, starting with 0 for Sunday and going to 6 for Saturday.)
 - Requires a new *DayOfWeek counter* for tracking the current time.
 - Days of the week for the alarm to go off can be set to any combination of days of the week.
 - While in the set alarm mode, *7 binary switches* (one for each day of the week) are used to select what days the alarm should go off, and the selected days are stored in a *Register-7* component;
 - A *Decoder-8* is used to convert the *DayOfWeek* counter value into a representation that can be easily compared to the days of the week the user selected for the alarm.

Behavioral Description

- Part 1: Basic alarm clock
- Part 2: Days of the Week
- **Part 3: A Second Alarm**
 - Add a second alarm that can be set to go off at a different time on a different set of days than the alarm you added in Part 2.
 - If no days are selected for the second alarm to go off, then the second alarm should not go off;
 - If no days are selected for the first alarm, it should still go off on every day like it did in Part 2.
 - Connections between components remain largely the same, however components need to be duplicated to store the alarm time and days for the second alarm.
 - Reuse parts when possible, such as displays and switches.
 - An additional *MUX-2x6* is needed to allow the user to see the time they are setting for the second alarm.

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 Provided for Part 2

- **Register-7:**
 - Stores 7 bits of data
 - Inputs D0..6 are loaded into the register while LD == 1
 - Outputs Q0..6 are the current value that is stored in the register
- **Decoder-8 (LogicWorks library component):**
 - If En == 1, all output = 1
 - If En == 0, For $i = 0$ to 7, $Q_i = 0$ if $S_2S_1S_0 == i$
- **Counter-4 (LogicWorks library component):**
 - A flexible 4-bit counter component that can be configured to increment or decrement on each CLK cycle.
- **Reuse**
 - In addition to the new components, you will need to use an additional **LCD Interface** component from Part 1.

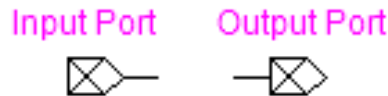
Components Reused for Part 3

- **Reuse**
 - You will need to use additional **AlarmDev** and **MUX** components from Part 1, and an additional **Register-7** component from Part 2

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