Introduction

For our hardware project, we decided to build a digital clock with functions such as the setting of the time, an alarm, setting of the alarm, a stopwatch, and perhaps a countdown timer. We decided upon this project because we wanted to simulate an ordinary wrist watch using digital circuitry, to see how complex a watch circuit could become, and to build something practical.

Design

The best way to organize the project was to break it down into smaller parts controlled by a finite state machine (See Appendix A, figure 1, figure 2). Following hierarchical design practices, the circuitry was decomposed into the following major components.

[a] 0-59 counter for seconds and minutes
[b] 1-12 counter for hours
[c] Moore finite state machine to control the entire circuit
[d] converters to change BCD (Binary Coded Decimal) outputs from counters to 7-segment display inputs
[e] multiplexer unit to change between the functions to be displayed (i.e. time, alarm, stopwatch, countdown timer)
[f] display unit made of six 7-segment displays (two each for seconds, minutes, and hours)
[g] comparator unit to determine if the alarm should be triggered

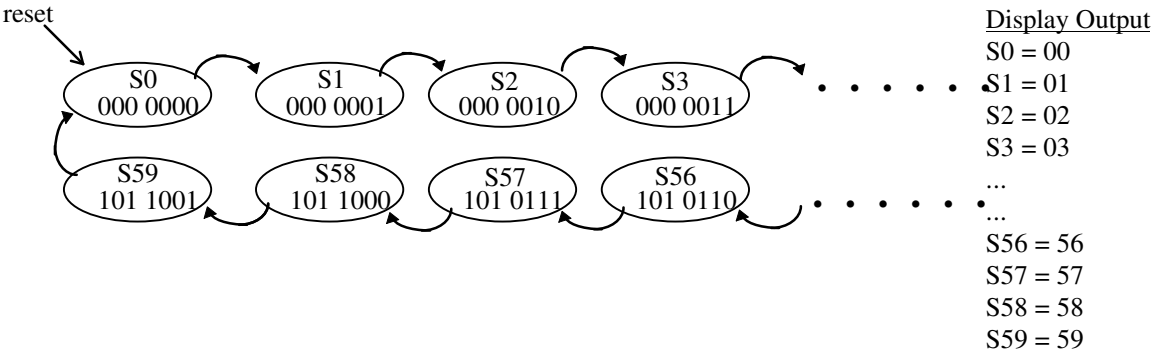
A: 0-59 Counter Design

The first idea was to design a synchronous ripple counter using T flip-flops. To implement this type of flip-flop on a standard P18CV8 PAL, each D flip-flop on the PAL was converted to a JK flip-flop and the inputs (J and K) were tied together to form a T flip-flop. Using this method, we designed a straight 0-59 counter, hoping that there was such a chip that converted seven bit binary numbers into two 7-segment display outputs. Unfortunately, these types of converters could not be found. Although PALs could have been programmed to perform this conversion task, this would have increased the number of parts required, and the circuit would have made inefficient use of PALs (compared to BCD to 7-segment display converter chips).

With this limitation in mind, a new set of requirements appeared. In order to display the time on 7-segment displays each digit had to have its own counter sub-circuit with 4-bit BCD output. To meet this requirement, we designed a 0-9 counter and a 0-5 counter, linking the resets of the flip-flops to a circuit that would allow for the cycling of each digit. Initially, these two counters were programmed in ABEL to fit onto one PAL

to the minimize cost and number of parts. Based upon all available documentation, this optimization seemed possible. However, upon testing the newly programmed PALs in the lab, we discovered that they could not support two resets for two different sets of flip-flops. Consequently, we could not use this design.

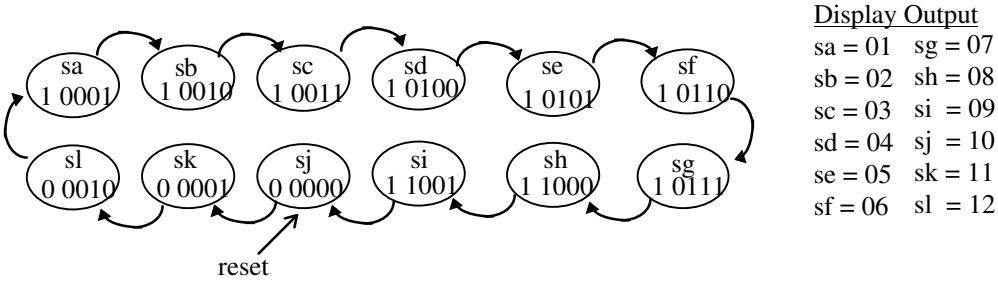
Finally, we decided to implement the 0-59 counter as a Moore finite state machine where the outputs were the states themselves. The outputs of the PAL were two BCD numbers, one set for the one's digit and the other set for the ten's digit. For the wiring schematic see Appendix A, figure 3. Below is a state diagram of the counter.



Considering the simplicity of the diagram and the similarity to a previous lab assignment the finite state machine was programmed directly into ABEL (See Appendix B, figures 1a, 1b, 1c).

B: 1-12 Counter Design

Learning from the previous design we decided to design this counter using a similar Moore finite state machine. For the wiring schematic see Appendix A, figure 4. Below is a state diagram of the counter.



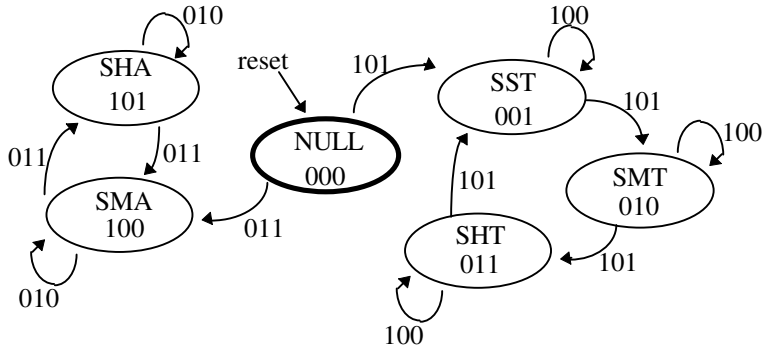
Again, for the same reasons as mentioned above, this circuit was programmed directly into ABEL (See Appendix B, figures 2a, 2b). Because one of our displays was common cathode (while the others were common anode) the BCD output for the ten's digit of this counter was handled differently. The last display was set to output a 1, and the last output of this counter determined whether this display was on or off. This had the added feature of displaying a 1 when the hour was between 10 and 12, and displaying nothing when the hour was between 1 and 9.

C: Clock Control Finite State Machine Design

The design of this subcircuit was implemented using the classic finite state machine design procedure. A Moore machine was designed with the following inputs.

- *time_set*: used to stop the clock and allow the setting of the time.
- *alarm_set*: used to switch to the alarm display (the clock would still run) and allow for the setting of the alarm.
- *shift*: used to cycle through the seconds, minutes, and hours display when setting either the time or alarm.

This finite state machine drove combinational logic that allowed either the standard clock frequency or the manual clock signal to enter the counters. For the wiring schematic see Appendix A, figure 5. For the ABEL program see Appendix B, figure 3. Below is a state diagram of the finite state machine.



Inputs:
time_set, alarm_set, shift

State Definitions
NULL: invalid inputs
SST: set seconds of the time
SMT: set minutes of the time
SHT: set hours of the time
SMA: set minutes of the alarm
SHA: set hours of the alarm

* all other input possibilities lead to the null state

D: BCD to 7-Segment Display Converters

Five 74LS47 TTL chips, specifically designed for common anode 7-segment displays, were purchased and utilized for this purpose. Although PALs could have been programmed for this purpose, the converter circuit would have made inefficient use of PAL resources because of the small logic task involved, and too many PALs would have been required.

E: Multiplexer Unit

Five 74LS157 TTL chips were used to toggle between the displaying of the time and the alarm setting. The selection input pins on these chips were wired to the alarm set input pin of the finite state machine to allow for the appropriate toggling of the display (See Appendix A, figure 2). In a similar case as mentioned above, PALs would have been inappropriate for this purpose.

F: Display Unit

Five of the common anode 7-segment displays received their inputs from the BCD to 7-segment converters. The sixth display unit, which was used to display the ten's digit of the hour, was common cathode and received its input from the ten's digit of the hours PAL via the multiplexer. As explained in section B, the common cathode display was set to output a 1 and was turned on and off by the hours PAL.

G: Comparator Unit

The design of this simple comparator unit utilized the 2-level exclusive NOR / AND prototype mentioned in our lectures (See Appendix A, figure 6). In the lab, this circuit was partitioned and distributed amongst several existing PALs, which had extra space for additional logic, to make maximum use of these chips. For the ABEL program see Appendix B, figures 1c, 2a, 2b.

Results of the Lab

As it is apparent, we did not implement a stopwatch or a countdown timer in our circuit. The reason being that the project grew rather quickly in size and cost. Also, adding additional features meant adding another three PALs for each feature and a larger multiplexer unit to handle the display routing.

In the first two lab periods the project was connected into three main sections, the display, multiplexer, and clock. Each section functioned correctly when tested separately. In the third lab period all three sections were connected together. At first the complete circuit did not function correctly. From experience, the 5 volt and ground pins on every chip was tested using the logic probe. Next, the inputs and outputs of the clock-control PAL and seconds PAL were checked using the probe. We found that the clock outputs were low and remained low regardless of the clock pulses that were fed into it. We then removed the PAL and tested it on another protoboard and found that it functioned correctly. Our next guess was to measure the voltage across the protoboard terminals using a voltmeter. From these measurements, we found that the voltage supplied was less than 5 volts (approximately 3.3 volts). To remedy this problem, we decided to supply power to each breadboard using separate power supplies (totaling three). After the connections were made, the circuit worked as expected.

Conclusions

We learned that the PAL's that we used were, to some extent, limited in what they could accomplish. From our experience, we could not implement more than one reset on a PAL. Also, though simple functions could be programmed into a PAL, these simple functions can often be found in inexpensive, mass produced, generic chips. In our case, instead of programming multiplexers and BCD to 7-segment converters, we purchased 74LS157 and 74LS47 chips that performed these specific tasks. We also learned that the 7-segment LED displays drained quite a lot of power, as we had to use a total of three separate power supplies to ensure the proper operation of our circuit. From this ordeal, we learned that the logic probe was useless when dealing with power consumption problems. We had to resort to a voltmeter to measure precise voltage values. We also gained practical experience on the voltage tolerances of chips (the voltage regions which are considered as high or low signals). Finally, we learned that simple devices can become rather complex when built out of medium-scale-integration components.

If we had to do our project over again, we would (1) use more sophisticated PALs, (2) use LCDs instead of LEDs to reduce power consumption, (3) improve the finite state machine that controlled the clock by having more user-friendly controls and having a display system for showing which parts of the time were being set by the user, (4) and have an AM/PM indicator.